

# Package ‘mwTensor’

October 12, 2021

**Type** Package

**Title** Multi-Way Component Analysis

**Version** 0.99.5

**Date** 2021-10-12

**Suggests** testthat

**Depends** R (>= 4.1.0)

**Imports** methods, MASS, rTensor, nnTensor, ccTensor, ica, igraph

**Description** For single tensor data, any matrix factorization method can be specified the matrixed tensor in each dimension by Multi-way Component Analysis (MWCA). An originally extended MWCA is also implemented to specify and decompose multiple matrices and tensors simultaneously (CoupledMWCA). See the reference section of GitHub README.md <<https://github.com/rikenbit/mwTensor>>, for details of the methods.

**License** Artistic-2.0

**URL** <https://github.com/rikenbit/mwTensor>

**NeedsCompilation** no

**Author** Koki Tsuyuzaki [aut, cre]

**Maintainer** Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

**Repository** CRAN

**Date/Publication** 2021-10-12 08:00:02 UTC

## R topics documented:

mwTensor-package . . . . .	2
CoupledMWCA . . . . .	4
CoupledMWCAParams-class . . . . .	5
CoupledMWCAResult-class . . . . .	6
MWCA . . . . .	8
MWCAParams-class . . . . .	9
MWCAResult-class . . . . .	9
myALS_SVD . . . . .	10

myCX . . . . .	11
myICA . . . . .	12
myNMF . . . . .	13
mySVD . . . . .	14
plotTensor3Ds . . . . .	14
toyModel . . . . .	15

<b>Index</b>	<b>16</b>
--------------	-----------

---

mwTensor-package	<i>Multi-Way Component Analysis</i>
------------------	-------------------------------------

---

## Description

For single tensor data, any matrix factorization method can be specified the matricised tensor in each dimension by Multi-way Component Analysis (MWCA). An originally extended MWCA is also implemented to specify and decompose multiple matrices and tensors simultaneously (CoupledMWCA). See the reference section of GitHub README.md <<https://github.com/rikenbit/mwTensor>>, for details of the methods.

## Details

The DESCRIPTION file:

```

Package:      mwTensor
Type:         Package
Title:        Multi-Way Component Analysis
Version:      0.99.5
Date:         2021-10-12
Authors@R:   c(person("Koki", "Tsuyuzaki", role = c("aut", "cre"), email = "k.t.the-answer@hotmail.co.jp"))
Suggests:    testthat
Depends:      R (>= 4.1.0)
Imports:      methods, MASS, rTensor, nnTensor, ccTensor, ica, igraph
Description:  For single tensor data, any matrix factorization method can be specified the matricised tensor in each dimension
License:     Artistic-2.0
URL:         https://github.com/rikenbit/mwTensor
Author:      Koki Tsuyuzaki [aut, cre]
Maintainer:  Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

```

Index of help topics:

```

CoupledMWCA          Coupled Multi-way Component Analysis
                     (CoupledMWCA)
CoupledMWCAParams-class  Class "CoupledMWCAParams"
CoupledMWCAResult-class  Class "CoupledMWCAResult"

```

MWCA	Multi-way Component Analysis (MWCA)
MWCAParams-class	Class "MWCAParams"
MWCAResult-class	Class "MWCAResult"
mwTensor-package	Multi-Way Component Analysis
myALS_SVD	Alternating Least Square Singular Value Decomposition (ALS-SVD) as an example of user-defined matrix decomposition.
myCX	CX Decomposition as an example of user-defined matrix decomposition.
myICA	Independent Component Analysis (ICA) as an example of user-defined matrix decomposition.
myNMF	Independent Component Analysis (ICA) as an example of user-defined matrix decomposition.
mySVD	Singular Value Decomposition (SVD) as an example of user-defined matrix decomposition.
plotTensor3Ds	Plot function for visualization of tensor data structure
toyModel	Toy model of coupled tensor data

**Author(s)**

NA

Maintainer: NA

**References**

- Andrzej Cichocki et al., (2016). Tensor Networks for Dimensionality Reduction and Large-scale Optimization: Part 1 Low-Rank Tensor Decompositions
- Andrzej Cichocki et al., (2015). Tensor Decompositions for Signal Processing Applications, *IEEE SIGNAL PROCESSING MAGAZINE*
- Gene H. Golub et al., (2012). Matrix Computation (Johns Hopkins Studies in the Mathematical Sciences), *Johns Hopkins University Press*
- Madeleine Udell et al., (2016). Generalized Low Rank Models, *Foundations and Trends in Machine Learning*, 9(1).
- Andrzej CICHOCKI, et. al., (2009). Nonnegative Matrix and Tensor Factorizations.
- A. Hyvarinen. (1999). Fast and Robust Fixed-Point Algorithms for Independent Component Analysis. *IEEE Transactions on Neural Networks*, 10(3), 626-634.
- Petros Drineas et al., (2008). Relative-Error CUR Matrix Decompositions, *SIAM Journal on Matrix Analysis and Applications*, 30(2), 844-881.

**See Also**

[mySVD](#), [myALS\\_SVD](#), [myNMF](#), [myICA](#), [myCX](#), [MWCA](#), [CoupledMWCA](#), [plotTensor3Ds](#)

**Examples**

```
ls("package:mwTensor")
```

---

CoupledMWCA

*Coupled Multi-way Component Analysis (CoupledMWCA)*

---

### **Description**

The input is assumed to be a CoupledMWCAParams object.

### **Usage**

```
CoupledMWCA(params)
```

### **Arguments**

params            CoupledMWCAParams object

### **Value**

CoupledMWCAResult object.

### **Author(s)**

Koki Tsuyuzaki

### **See Also**

[CoupledMWCAParams-class](#) and [CoupledMWCAResult-class](#).

### **Examples**

```
if(interactive()){  
# Test data (multiple arrays)  
  Xs <- toyModel("coupled_CP_Easy")  
  
  params <- new("CoupledMWCAParams", Xs=Xs)  
  out <- CoupledMWCA(params)  
}
```

---

 CoupledMWCAParams-class

 Class "CoupledMWCAParams"
 

---

### Description

The parameter object to be specified against CoupledMWCA function.

### Objects from the Class

Objects can be created by calls of the form `new("CoupledMWCAParams", ...)`.

### Slots

MWCAParams has four settings as follows. For each setting, the list must have the same structure.

*1. Data-wise setting* Each item must be a list object that is as long as the number of data and is named after the data.

A list containing multiple high-dimensional arrays.

**mask:** A list containing multiple high-dimensional arrays, in which 0 or 1 values are filled to specify the missing elements.

**weights:** A list containing multiple high-dimensional arrays, in which some numeric values are specified to weigh each data.

*2. Common Model setting* Each item must be a nested list object that is as long as the number of data and is named after the data.

**common\_model:** Each element of the list must be a list corresponding the dimension name of data and common factor matrices name.

*3. Common Factor matrix-wise setting* Each item must be a list object that is as long as the number of common factor matrices and is named after the factor matrices.

**common\_initial:** The initial values of common factor matrices. If nothing is specified, random matrices are used.

**common\_algorithms:** Algorithms used to decompose the matricised tensor in each mode.

**common\_iteration:** The number of iterations.

**common\_decomp:** If FALSE is specified, unit matrix is used as the common factor matrix.

**common\_fix:** If TRUE is specified, the common factor matrix is not updated in the iteration.

**common\_dims:** The lower dimension of each common factor matrix.

**common\_transpose:** Whether the common factor matrix is transposed to calculate core tensor.

**common\_coretype:** If "CP" is specified, all the core tensors become diagonal core tensors. If "Tucker" is specified, all the core tensors become dense core tensors.

*4. Specific Model setting* Each item must be a nested list object that is as long as the number of data and is named after the data.

**specific\_model:** Each element of the list must be a list corresponding the dimension name of data and data specific factor matrices name.

5. *Specific Factor matrix-wise setting* Each item must be a list object that is as long as the number of data specific factor matrices and is named after the factor matrices.

**specific\_initial:** The initial values of data specific factor matrices. If nothing is specified, random matrices are used.

**specific\_algorithms:** Algorithms used to decompose the matricised tensor in each mode.

**specific\_iteration:** The number of iterations.

**specific\_decomp:** If FALSE is specified, unit matrix is used as the data specific factor matrix.

**specific\_fix:** If TRUE is specified, the data specific factor matrix is not updated in the iteration.

**specific\_dims:** The lower dimension of each data specific factor matrix.

**specific\_transpose:** Whether the data specific factor matrix is transposed to calculate core tensor.

**specific\_coretype:** If "CP" is specified, all the core tensors become diagonal core tensors. If "Tucker" is specified, all the core tensors become dense core tensors.

6. *Other option* Each item must to be a vector of length 1.

**specific:** Whether data specific factor matrices are also calculated.

**thr:** The threshold to stop the iteration. The higher the value, the faster the iteration will stop.

**viz:** Whether the output is visualized.

**figdir:** When viz=TRUE, whether the plot is output in the directory.

**verbose:** Whether the process is monitored by verbose messages.

## Methods

**CoupledMWCA** Function to perform CoupledMWCA.

## See Also

[CoupledMWCAResult-class](#), [CoupledMWCA](#)

---

CoupledMWCAResult-class

Class "CoupledMWCAResult"

---

## Description

The result object generated by CoupledMWCA function.

**Slots**

**weights:** weights of CoupledMWCAParams.  
**common\_model:** common\_model of CoupledMWCAParams.  
**common\_initial:** common\_initial of CoupledMWCAParams.  
**common\_algorithms:** common\_algorithms of CoupledMWCAParams.  
**common\_iteration:** common\_iteration of CoupledMWCAParams.  
**common\_decomp:** common\_decomp of CoupledMWCAParams.  
**common\_fix:** common\_fix of CoupledMWCAParams.  
**common\_dims:** common\_dims of CoupledMWCAParams.  
**common\_transpose:** common\_transpose of CoupledMWCAParams.  
**common\_coretype:** common\_coretype of CoupledMWCAParams.  
**common\_factors:** Common factor matrices of CoupledMWCA.  
**common\_cores:** Common core tensors of CoupledMWCA.  
**specific\_model:** specific\_model of CoupledMWCAParams.  
**specific\_initial:** specific\_initial of CoupledMWCAParams.  
**specific\_algorithms:** specific\_algorithms of CoupledMWCAParams.  
**specific\_iteration:** specific\_iteration of CoupledMWCAParams.  
**specific\_decomp:** specific\_decomp of CoupledMWCAParams.  
**specific\_fix:** specific\_fix of CoupledMWCAParams.  
**specific\_dims:** specific\_dims of CoupledMWCAParams.  
**specific\_transpose:** specific\_transpose of CoupledMWCAParams.  
**specific\_coretype:** specific\_coretype of CoupledMWCAParams.  
**specific\_factors:** Data specific factor matrices of CoupledMWCA.  
**specific\_cores:** Data specific core tensors of CoupledMWCA.  
**specific:** specific of CoupledMWCAParams.  
**thr:** thr of CoupledMWCAParams.  
**viz:** viz of CoupledMWCAParams.  
**figdir:** figdir of CoupledMWCAParams.  
**verbose:** verbose of CoupledMWCAParams.  
**rec\_error:** The reconstructed error.  
**train\_error:** Training Error.  $\text{train\_error} + \text{test\_error} = \text{rec\_error}$ .  
**test\_error:** Test Error.  $\text{train\_error} + \text{test\_error} = \text{rec\_error}$ .  
**rel\_change:** The relative change of each iteration step.

**See Also**

[CoupledMWCAParams-class](#), [CoupledMWCA](#)

---

MWCA

*Multi-way Component Analysis (MWCA)*

---

### Description

The input is assumed to be a `MWCAParams` object.

### Usage

```
MWCA(params)
```

### Arguments

`params`            `MWCAParams` object

### Value

`MWCAResult` object.

### Author(s)

Koki Tsuyuzaki

### References

Andrzej Cichocki et al., (2016). Tensor Networks for Dimensionality Reduction and Large-scale Optimization: Part 1 Low-Rank Tensor Decompositions

Andrzej Cichocki et al., (2015). Tensor Decompositions for Signal Processing Applications, *IEEE SIGNAL PROCESSING MAGAZINE*

### See Also

[MWCAParams-class](#) and [MWCAResult-class](#).

### Examples

```
if(interactive()){  
  # Test data (single array)  
  X <- nnTensor::toyModel("Tucker")@data  
  
  params <- new("MWCAParams", X=X)  
  out <- MWCA(params)  
}
```



---

MWCAParams-class	Class "MWCAParams"
------------------	--------------------

---

### Description

The parameter object to be specified against MWCA function.

### Objects from the Class

Objects can be created by calls of the form `new("MWCAParams", ...)`.

### Slots

**X:** A high-dimensional array.

**mask:** A mask array having the same dimension of X.

**algorithms:** Algorithms used to decompose the matricised tensor in each mode.

**dims:** The lower dimension of each factor matrix.

**transpose:** Whether the factor matrix is transposed to calculate core tensor.

**viz:** Whether the output is visualized.

**figdir:** When viz=TRUE, whether the plot is output in the directory.

### Methods

**MWCA** Function to perform MWCA.

### See Also

[MWCAResult-class](#), [MWCA](#)

---

MWCAResult-class	Class "MWCAResult"
------------------	--------------------

---

### Description

The result object generated by MWCA function.

**Slots**

**algorithms:** algorithm of MWCAParams.

**dims:** dims of MWCAParams.

**transpose:** transpose of MWCAParams.

**viz:** viz of MWCAParams.

**figdir:** figdir of MWCAParams.

**factors:** The factor matrices of MWCA.

**core:** The core tensor of MWCA.

**rec\_error:** The reconstructed error.

**train\_error:** Training Error.  $\text{train\_error} + \text{test\_error} = \text{rec\_error}$ .

**test\_error:** Test Error.  $\text{train\_error} + \text{test\_error} = \text{rec\_error}$ .

**See Also**

[MWCAParams-class](#), [MWCA](#)

---

myALS\_SVD

*Alternating Least Square Singular Value Decomposition (ALS-SVD)  
as an example of user-defined matrix decomposition.*

---

**Description**

The input data is assumed to be a matrix. When algorithms of MWCAParams and CoupledMWCA-Params are specified as "myALS\_SVD", This function is called in MWCA and CoupledMWCA.

**Usage**

```
myALS_SVD(Xn, k, L2=1e-10, iter=30)
```

**Arguments**

Xn	The input matrix which has N-rows and M-columns.
k	The rank parameter ( $k \leq \min(N,M)$ )
L2	The regularization parameter (Default: 1e-10)
iter	The number of iteration (Default: 30)

**Value**

The output matrix which has N-rows and k-columns.

**Author(s)**

Koki Tsuyuzaki

**References**

Madeleine Udell et al., (2016). Generalized Low Rank Models, *Foundations and Trends in Machine Learning*, 9(1).

**Examples**

```
if(interactive()){
  # Test data
  matdata <- matrix(runif(10*20), nrow=10, ncol=20)
  myALS_SVD(matdata, k=3, L2=0.1, iter=10)
}
```

---

myCX	<i>CX Decomposition as an example of user-defined matrix decomposition.</i>
------	---

---

**Description**

The input data is assumed to be a matrix. When algorithms of MWCAParams and CoupledMWCAParams are specified as "myCX", This function is called in MWCA and CoupledMWCA.

**Usage**

```
myCX(Xn, k)
```

**Arguments**

Xn	The input matrix which has N-rows and M-columns.
k	The rank parameter ( $k \leq \min(N,M)$ )

**Value**

The output matrix which has N-rows and k-columns.

**Author(s)**

Koki Tsuyuzaki

**References**

Petros Drineas et al., (2008). Relative-Error CUR Matrix Decompositions, *SIAM Journal on Matrix Analysis and Applications*, 30(2), 844-881.

**Examples**

```
if(interactive()){
  # Test data
  matdata <- matrix(runif(10*20), nrow=10, ncol=20)
  myCX(matdata, k=3)
}
```

---

myICA	<i>Independent Component Analysis (ICA) as an example of user-defined matrix decomposition.</i>
-------	---

---

**Description**

The input data is assumed to be a matrix. When algorithms of MWCAParams and CoupledMWCAParams are specified as "myICA", This function is called in MWCA and CoupledMWCA.

**Usage**

```
myICA(Xn, k)
```

**Arguments**

Xn	The input matrix which has N-rows and M-columns.
k	The rank parameter ( $k \leq \min(N,M)$ )

**Value**

The output matrix which has N-rows and k-columns.

**Author(s)**

Koki Tsuyuzaki

**References**

A. Hyvarinen. (1999). Fast and Robust Fixed-Point Algorithms for Independent Component Analysis. *IEEE Transactions on Neural Networks*, 10(3), 626-634.

**Examples**

```
if(interactive()){  
  # Test data  
  matdata <- matrix(runif(10*20), nrow=10, ncol=20)  
  myICA(matdata, k=3)  
}
```

---

myNMF	<i>Independent Component Analysis (ICA) as an example of user-defined matrix decomposition.</i>
-------	---

---

### Description

The input data is assumed to be a matrix. When algorithms of MWCAParams and CoupledMWCAParams are specified as "myNMF", This function is called in MWCA and CoupledMWCA.

### Usage

```
myNMF(Xn, k, L1=1e-10, L2=1e-10)
```

### Arguments

Xn	The input matrix which has N-rows and M-columns.
k	The rank parameter ( $k \leq \min(N,M)$ )
L1	The regularization parameter to control the sparseness (Default: 1e-10)
L2	The regularization parameter to control the overfit (Default: 1e-10)

### Value

The output matrix which has N-rows and k-columns.

### Author(s)

Koki Tsuyuzaki

### References

Andrzej CICHOCK, et. al., (2009). Nonnegative Matrix and Tensor Factorizations.

### Examples

```
if(interactive()){  
  # Test data  
  matdata <- matrix(runif(10*20), nrow=10, ncol=20)  
  myNMF(matdata, k=3, L1=1e-1, L2=1e-2)  
}
```

---

mySVD	<i>Singular Value Decomposition (SVD) as an example of user-defined matrix decomposition.</i>
-------	---

---

**Description**

The input data is assumed to be a matrix. When algorithms of MWCAParams and CoupledMWCAParams are specified as "mySVD", This function is called in MWCA and CoupledMWCA.

**Usage**

```
mySVD(Xn, k)
```

**Arguments**

Xn	The input matrix which has N-rows and M-columns.
k	The rank parameter ( $k \leq \min(N,M)$ )

**Value**

The output matrix which has N-rows and k-columns.

**Author(s)**

Koki Tsuyuzaki

**Examples**

```
if(interactive()){  
  # Test data  
  matdata <- matrix(runif(10*20), nrow=10, ncol=20)  
  mySVD(matdata, k=3)  
}
```

---

plotTensor3Ds	<i>Plot function for visualization of tensor data structure</i>
---------------	---

---

**Description**

Multiple multi-dimensional arrays and matrices are visualized simultaneously.

**Usage**

```
plotTensor3Ds(Xs)
```

**Arguments**

`Xs` A List object containing multi-dimensional array (or matrix) in each element.

**Author(s)**

Koki Tsuyuzaki

**See Also**

[plotTensor3D](#) and [plotTensor2D](#).

**Examples**

```
Xs <- toyModel(model = "coupled_CP_Easy")

tmp <- tempdir()

png(filename=paste0(tmp, "/couled_CP.png"))
plotTensor3Ds(Xs)
dev.off()
```

---

toyModel

*Toy model of coupled tensor data*

---

**Description**

A list object containing multiple arrays are generated.

**Usage**

```
toyModel(model = "coupled_CP_Easy", seeds=123)
```

**Arguments**

`model` "coupled\_CP\_Easy", "coupled\_CP\_Hard", "coupled\_Tucker\_Easy", "coupled\_Tucker\_Hard", "coupled\_Complex\_Easy", or "coupled\_Complex\_Hard" can be specified (Default: "coupled\_CP\_Easy").

`seeds` The seed of random number (Default: 123).

**Author(s)**

Koki Tsuyuzaki

**Examples**

```
Xs <- toyModel(model = "coupled_CP_Easy", seeds=123)
```

# Index

## \* **methods**

- CoupledMWCA, [4](#)
- MWCA, [8](#)
- myALS\_SVD, [10](#)
- myCX, [11](#)
- myICA, [12](#)
- myNMF, [13](#)
- mySVD, [14](#)
- plotTensor3Ds, [14](#)
- toyModel, [15](#)

## \* **package**

- mwTensor-package, [2](#)

CoupledMWCA, [3](#), [4](#), [6](#), [7](#)

CoupledMWCA, CoupledMWCAParams-method  
(CoupledMWCA), [4](#)

CoupledMWCAParams-class, [5](#)

CoupledMWCAResult-class, [6](#)

MWCA, [3](#), [8](#), [9](#), [10](#)

MWCA, MWCAParams-method (MWCA), [8](#)

MWCAParams-class, [9](#)

MWCAResult-class, [9](#)

mwTensor (mwTensor-package), [2](#)

mwTensor-package, [2](#)

myALS\_SVD, [3](#), [10](#)

myCX, [3](#), [11](#)

myICA, [3](#), [12](#)

myNMF, [3](#), [13](#)

mySVD, [3](#), [14](#)

plotTensor2D, [15](#)

plotTensor3D, [15](#)

plotTensor3Ds, [3](#), [14](#)

toyModel, [15](#)