Package 'tensorFun'

November 1, 2022

Type Package

Title Basic Functions to Handle Tensor Data in Array Class

Version 0.1.1

Date 2022-10-31

Author Zetai Cen [aut, cre]

Maintainer Zetai Cen <z.cen@lse.ac.uk>

Description Basic functions to handle higher-order tensor data. See Kolda and Bader (2009) <doi:10.1137/07070111X> for details on tensor. While existing packages on tensor data extend the base 'array' class to some S4 classes, this package serves as an alternative resort to handle tensor only as 'array' class.
Some functionalities related to missingness and rearrangement, discussed in Bai and Ng (2021) <arXiv:1910.06677>, are also supported.

License GPL-3

Imports ClimProjDiags, psychTools, MASS

Encoding UTF-8

RoxygenNote 7.2.1

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-11-01 08:02:25 UTC

R topics documented:

`All_inv` *Inverse rearrangement algorithm on all modes*

Description

The inverse algorithm of rearrangement algorithm for all modes

Usage

```
All_inv(ten_bundle, mode = "all")
```

Arguments

`ten_bundle`	A list of two objects, a multi-dimensional array and a list of indices along the modes to reverse
`mode`	A vector implying the modes to reverse, correspondent to the list in 'ten_bundle', set as 'all' by default

Value

A multi-dimensional array

`All_rearrange` *Rearrangement algorithm on all modes*

Description

Rearrangement algorithm for all modes

Usage

```
All_rearrange(ten, except = "NA", key = "NA")
```

Arguments

`ten`	A multi-dimensional array
`except`	A vector of integers implying the modes not to rearrange, set as 'NA' by default
`key`	The value to which rearrange is according, set as 'NA' by default

Details

A rearrangement algorithm on higher order tensor to rearrange all missing entries to a corner block. The case of mode-2 tensor returns to the work of Bai and Ng in 2021.

Value

A list of two objects, a rearranged tensor and a list of indices rearranged (ordered by the rearranged modes)

H_trans *Mode-k Block-a Block-b transformation matrix*

Description

Defined for an ongoing working paper

Usage

```
H_trans(A, B)
```

Arguments

A	A matrix with n rows and l columns
B	A matrix with k rows and l columns, with k no larger than n

Value

A matrix

Mcut *Matrix cutting*

Description

Cutting the bottom few rows of a matrix B so that B has the same number of rows as matrix A

Usage

```
Mcut(A, B)
```

Arguments

A	A matrix with n rows and l columns
B	A matrix with k rows and l columns, with k no less than n

Value

A matrix with n rows and l columns

Examples

```
Mcut(matrix(1:4,nrow=2), matrix(1:6,nrow=3))
```

---

obs_ind *Largest index observed*

Description

Finding the largest index I along a tensor mode k such that the block until index I contains no NA

Usage

```
obs_ind(ten, k)
```

Arguments

ten	A multi-dimensional array
k	An integer specifying the tensor mode to check

Value

An integer specifying the largest possible index

---

rearrange *Rearrangement algorithm on mode k*

Description

Rearrangement algorithm on one mode

Usage

```
rearrange(ten, k, key = "NA")
```

Arguments

ten	A multi-dimensional array
k	An integer specifying the mode to arrange
key	The value to which rearrange is according, set as 'NA' by default

Details

A rearrangement algorithm on higher order tensor to rearrange all missing entries along mode k to the end.

Value

A list of two objects, a rearranged tensor and the indices rearranged

---

rearrange_inv *Inverse rearrangement algorithm on mode k*

Description

The inverse algorithm of rearrangement algorithm for one mode

Usage

```
rearrange_inv(ten, k, l)
```

Arguments

ten	A multi-dimensional array
k	An integer specifying the mode k to rearrange backwards
l	A vector specifying the original indices rearranged

Value

A multi-dimensional array

---

refold *Tensor refolding*

Description

Performing tensorization, which is the inverse process of unfolding

Usage

```
refold(unfolding, k, dim_vec)
```

Arguments

unfolding	A multi-dimensional array
k	An integer specifying the mode of array to unfold
dim_vec	A vector specifying the expected dimension of output array

Value

A multi-dimensional array

Examples

```
refold(matrix(1:9,nrow=3), 1, c(3,1,3))
```

`ttm` *Mode k product with matrix*

Description

Performing k-mode matrix product of a tensor to a matrix

Usage

```
ttm(ten, A, k)
```

Arguments

`ten`	A multi-dimensional array with the k mode dimension aaa
`A`	A matrix with dimension bbb by aaa
`k`	An integer specifying the tensor mode to perform k-mode matrix product

Value

A multi-dimensional array with the k mode dimension bbb

Examples

```
ttm(array(1:24,c(3,4,2)), matrix(1:4,nrow =2), 3)
```

`unfold` *Tensor unfolding*

Description

Performing tensor unfolding, also known as matricization

Usage

```
unfold(ten, k)
```

Arguments

`ten`	A multi-dimensional array
`k`	An integer specifying the mode of array to unfold

Value

A matrix

Examples

```
unfold(array(1:24, dim=c(3,4,2)), 2)
```

Index