Package 'nnTensor'

June 4, 2018

Type Package

Title Non-Negative Tensor Decomposition

Version 0.99.1

Date 2018-06-01

Author Koki Tsuyuzaki, Manabu Ishii, Itoshi Nikaido

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

Suggests testthat

Depends R (>= 3.4.0), fields, rTensor, plot3D, tagcloud

Imports methods

Description Some functions for performing non-negative matrix factorization, non-negative CANDECOMP/PARAFAC (CP) decomposition, non-negative Tucker decomposition, and generating toy model data. See Andrzej Cichock et al (2009) <doi:10.1002/9780470747278> and the reference section of GitHub README.md <https://github.com/rikenbit/nnTensor>, for details of the methods.

License Artistic-2.0

URL https://github.com/rikenbit/nnTensor

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-04 08:48:01 UTC

R topics documented:

nnTensor-package *Non-Negative Tensor Decomposition*

Description

Some functions for performing non-negative matrix factorization, non-negative CANDECOMP/PARAFAC (CP) decomposition, non-negative Tucker decomposition, and generating toy model data. See Andrzej Cichock et al (2009) <doi:10.1002/9780470747278> and the reference section of GitHub README.md <https://github.com/rikenbit/nnTensor>, for details of the methods.

Details

The DESCRIPTION file:

Package:	nnTensor
Type:	Package
Title:	Non-Negative Tensor Decomposition
Version:	0.99.1
Date:	2018-06-01
Author:	Koki Tsuyuzaki, Manabu Ishii, Itoshi Nikaido
Maintainer:	Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>
Suggests:	testthat
Depends:	R (>= 3.4.0), fields, rTensor, plot3D, tagcloud
Imports:	methods
Description:	Some functions for performing non-negative matrix factorization, non-negative CANDECOMP/PARAFAC (CI
License:	Artistic-2.0
URL:	https://github.com/rikenbit/nnTensor

Index of help topics:

NMF	Non-negative Matrix Factorization Algorithms (NMF)
NTD	Non-negative Tucker Decomposition Algorithms (NTD)
NTF	Non-negative CP Decomposition Algorithms (NTF)
nnTensor-package	Non-Negative Tensor Decomposition
plotTensor3D	Plot function for visualization of tensor data structure
recTensor	Tensor Reconstruction from core tensor (S) and factor matrices (A)
toyModel	Toy model data for using NMF, NTF, and NTD

Author(s)

Koki Tsuyuzaki, Manabu Ishii, Itoshi Nikaido

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

References

Andrzej CICHOCK, et. al., (2009). Nonnegative Matrix and Tensor Factorizations. *John Wiley & Sons, Ltd*

Keigo Kimura, (2017). A Study on Efficient Algorithms for Nonnegative Matrix/Tensor Factorization. *Hokkaido University Collection of Scholarly and Academic Papers*

Andrzej CICHOCKI et. al., (2007). Non-negative Tensor Factorization using Alpha and Beta Divergence. *IEEE ICASSP 2007*

Anh Huy PHAN et. al., (2008). Multi-way Nonnegative Tensor Factorization Using Fast Hierarchical Alternating Least Squares Algorithm (HALS). *NOLTA2008*

Andrzej CICHOCKI et. al., (2008). Fast Local Algorithms for Large Scale Nonnegative Matrix and Tensor Factorizations. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*

Yong-Deok Kim et. al., (2007). Nonnegative Tucker Decomposition. *IEEE Conference on Computer Vision and Pattern Recognition*

Yong-Deok Kim et. al., (2008). Nonneegative Tucker Decomposition With Alpha-Divergence. *IEEE International Conference on Acoustics, Speech and Signal Processing*

Anh Huy Phan, (2008). Fast and efficient algorithms for nonnegative Tucker decomposition. *Advances in Neural Networks - ISNN2008*

Anh Hyu Phan et. al. (2011). Extended HALS algorithm for nonnegative Tucker decomposition and its applications for multiway analysis and classification. *Neurocomputing*

See Also

`toyModel`,`NMF`,`NTF`,`NTD`,`recTensor`,`plotTensor3D`

Examples

```
ls("package:nnTensor")
```

`NMF` *Non-negative Matrix Factorization Algorithms (NMF)*

Description

The input data is assumed to be non-negative matrix. NMF decompose the matrix to two low-dimensional factor matices. This function is also used as initialization step of tensor decomposition (see also NTF and NTD).

Usage

```
NMF(X, J = 3, algorithm = "Frobenius", Alpha = 1,
  Beta = 2, eta = 1e-04, thr1 = 1e-10, thr2 = 1e-10,
  tol = 1e-04, num.iter = 100, viz = FALSE, figdir = ".", verbose = FALSE)
```

Arguments

X	The input Matrix which has N-rows and M-columns.
J	Number of low-dimension (J < N, M).
algorithm	NMF algorithms. "Frobenius", "KL", "IS", "Pearson", "Hellinger", "Neyman", "HALS", "PGD", "Alpha", "Beta", and "GCD" are available (Default: "Frobenius").
Alpha	The parameter of Alpha-divergence.
Beta	The parameter of Beta-divergence.
eta	The stepsize for PGD algorithm (Default: 0.0001).
thr1	When error change rate is lower than thr1, the iteration is terminated (Default: 1E-10).
thr2	If the minus-value is generated, replaced as thr2 (Default: 1E-10). This value is used within the internal function .positive().
tol	The tolerance parameter used in GCD algorithm.
num.iter	The number of interation step (Default: 100).
viz	If viz == TRUE, internal reconstructed matrix can be visualized.
figdir	the directory for saving the figure, when viz == TRUE.
verbose	If verbose == TRUE, Error change rate is generated in console windos.

Value

U : A matrix which has N-rows and J-columns (J < N, M). V : A matrix which has M-rows and J-columns (J < N, M).

Author(s)

Koki Tsuyuzaki

References

Andrzej CICHOCK, et. al., (2009). Nonnegative Matrix and Tensor Factorizations. *John Wiley & Sons, Ltd*

Keigo Kimura, (2017). A Study on Efficient Algorithms for Nonnegative Matrix/Tensor Factorization. *Hokkaido University Collection of Scholarly and Academic Papers*

Examples

```
library("fields")
matdata <- toyModel(model = "NMF")
out <- NMF(matdata, J=2, algorithm="Frobenius", num.iter=2)
```

NTD *Non-negative Tucker Decomposition Algorithms (NTD)*

Description

The input data is assumed to be non-negative tensor. NTD decompose the tensor to the dense core tensor (S) and low-dimensional factor matices (A).

Usage

```
NTD(X, rank = c(3, 3, 3), algorithm = "KL", init = "NMF", Alpha = 1,
  Beta = 2, thr = 1e-10, num.iter = 100, viz = FALSE,
  figdir = ".", verbose = FALSE)
```

Arguments

X	The input tensor which has I1, I2, and I3 dimensions.
rank	The number of low-dimension in each mode (J1, J2, J3, J1<I1, J2<I2, J3 < I3) (Default: c(3,3,3)).
algorithm	NTD algorithms. "Frobenius", "KL", "IS", "Pearson", "Hellinger", "Neyman", "HALS", "Alpha", and "Beta" are available (Default: "Frobenius").
init	The initialization algorithms. "NMF", "ALS", and "Random" are available (Default: "NMF").
Alpha	The parameter of Alpha-divergence.
Beta	The parameter of Beta-divergence.
thr	When error change rate is lower than thr1, the iteration is terminated (Default: 1E-10).
num.iter	The number of interation step (Default: 100).
viz	If viz == TRUE, internal reconstructed tensor can be visualized.
figdir	the directory for saving the figure, when viz == TRUE.
verbose	If verbose == TRUE, Error change rate is generated in console windos.

Value

S : Tensor object, which is defined as S4 class of rTensor package. A : A list containing three factor matrices.

Author(s)

Koki Tsuyuzaki

References

Yong-Deok Kim et. al., (2007). Nonnegative Tucker Decomposition. *IEEE Conference on Computer Vision and Pattern Recognition*

Yong-Deok Kim et. al., (2008). Nonneegative Tucker Decomposition With Alpha-Divergence. *IEEE International Conference on Acoustics, Speech and Signal Processing*

Anh Huy Phan, (2008). Fast and efficient algorithms for nonnegative Tucker decomposition. *Advances in Neural Networks - ISNN2008*

Anh Hyu Phan et. al. (2011). Extended HALS algorithm for nonnegative Tucker decomposition and its applications for multiway analysis and classification. *Neurocomputing*

See Also

plotTensor3D

Examples

```
tensordata <- toyModel(model = "Tucker")
out <- NTD(tensordata, rank=c(2,2,2), algorithm="Frobenius", init="Random", num.iter=2)
```

NTF *Non-negative CP Decomposition Algorithms (NTF)*

Description

The input data is assumed to be non-negative tensor. NTF decompose the tensor to the diagonal core tensor (S) and low-dimensional factor matices (A).

Usage

```
NTF(X, rank = 3, algorithm = "KL", init = "NMF", Alpha = 1,
  Beta = 2, thr = 1e-10, num.iter = 100, viz = FALSE,
  figdir = ".", verbose = FALSE)
```

Arguments

X	The input tensor which has I1, I2, and I3 dimensions.
rank	The number of low-dimension in each mode (J1=J2=J3, J1<I1, J2<I2, J3 < I3) (Default: 3).
algorithm	NTF algorithms. "Frobenius", "KL", "IS", "Pearson", "Hellinger", "Neyman", "HALS", "Alpha-HALS", "Beta-HALS", "Alpha", and "Beta" are available (Default: "Frobenius").
init	The initialization algorithms. "NMF", "ALS", and "Random" are available (Default: "NMF").
Alpha	The parameter of Alpha-divergence.
Beta	The parameter of Beta-divergence.

thr	When error change rate is lower than thr1, the iteration is terminated (Default: 1E-10).
num.iter	The number of interation step (Default: 100).
viz	If viz == TRUE, internal reconstructed tensor can be visualized.
figdir	the directory for saving the figure, when viz == TRUE.
verbose	If verbose == TRUE, Error change rate is generated in console windos.

Value

S : Tensor object, which is defined as S4 class of rTensor package. A : A list containing three factor matrices.

Author(s)

Koki Tsuyuzaki

References

Andrzej CICHOCKI et. al., (2007). Non-negative Tensor Factorization using Alpha and Beta Divergence. *IEEE ICASSP 2007*

Anh Huy PHAN et. al., (2008). Multi-way Nonnegative Tensor Factorization Using Fast Hierarchical Alternating Least Squares Algorithm (HALS). *NOLTA2008*

Andrzej CICHOCKI et. al., (2008). Fast Local Algorithms for Large Scale Nonnegative Matrix and Tensor Factorizations. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*

See Also

plotTensor3D

Examples

```
tensordata <- toyModel(model = "CP")
out <- NTF(tensordata, rank=3, algorithm="Beta-HALS", num.iter=2)
```

---

plotTensor3D *Plot function for visualization of tensor data structure*

Description

Combined with recTensor function and the result of NTF or NTD, the reconstructed tesor structure can be visullized.

Usage

```
plotTensor3D(X = NULL)
```

Arguments

X	Tensor object, which is defined as S4 class of rTensor package.

Author(s)

Koki Tsuyuzaki

Examples

```
tensordata <- toyModel(model = "CP")

out <- NTF(tensordata, rank=3, algorithm="Beta-HALS", num.iter=2)

tmp <- tempdir()

png(filename=paste0(tmp, "/NTF.png"))
plotTensor3D(recTensor(out$S, out$A))
dev.off()
```

---

recTensor *Tensor Reconstruction from core tensor (S) and factor matrices (A)*

Description

Combined with plotTensor3D function and the result of NTF or NTD, the reconstructed tesor structure can be visullized.

Usage

```
recTensor(S = NULL, A = NULL, idx = 1:3, reverse = FALSE)
```

Arguments

S	Tensor object, which is defined as S4 class of rTensor package.
A	A list containing three factor matrices.
idx	The direction of mode-n muliplication (Default: 1:3). For example idx=1 is defined. S x_1 A is calculated (x_1 : mode-1 multiplication).
reverse	If reverse = TRUE, t(A[[n]]) is multiplicated to S (Default: FALSE).

Value

Tensor object, which is defined as S4 class of rTensor package.

Author(s)

Koki Tsuyuzaki

See Also

Tensor-class, NTF, NTD

Examples

```
tensordata <- toyModel(model = "CP")
out <- NTF(tensordata, rank=3, algorithm="Beta-HALS", num.iter=2)
rec <- recTensor(out$S, out$A)
```

---

toyModel *Toy model data for using NMF, NTF, and NTD*

Description

The data is used for confirming the algorithm are properly working.

Usage

```
toyModel(model = "CP")
```

Arguments

model	Single character string is specified. "NMF", "CP", and "Tucker" are available (Default: "CP").

Value

If model is specified as "NMF", a matrix is generated. Otherwise, a tensor is generated.

Author(s)

Koki Tsuyuzaki

See Also

NMF, NTF, NTD

Examples

```
matdata <- toyModel(model = "NMF")
tensordata1 <- toyModel(model = "CP")
tensordata2 <- toyModel(model = "Tucker")
```

Index