Package ‘nnTensor’

September 3, 2021

Type Package
Title Non-Negative Tensor Decomposition
Version 1.1.5
Date 2021-08-17
Suggests testthat
Depends R (>= 3.4.0)
Imports methods, fields, rTensor, plot3D, tagcloud, ggplot2
Description Some functions for performing non-negative matrix factorization, non-negative CANDE-
License Artistic-2.0
URL https://github.com/rikenbit/nnTensor
NeedsCompilation no
Author Koki Tsuyuzaki [aut, cre],
  Manabu Ishii [aut],
  Itoshi Nikaido [aut]
Maintainer Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>
Repository CRAN
Date/Publication 2021-09-03 04:50:31 UTC

R topics documented:
nnTensor-package ................................................. 2
GabrielNMF ...................................................... 4
jNMF ............................................................ 5
NMF ............................................................. 7
NTD .............................................................. 10
NTF .............................................................. 12
plot.NMF ......................................................... 14
Description


Details

The DESCRIPTION file:

Package:    nnTensor
Type:       Package
Title:      Non-Negative Tensor Decomposition
Version:    1.1.5
Date:       2021-08-17
Authors@R:  c(person("Koki", "Tsuyuzaki", role = c("aut", "cre"), email = "k.t.the-answer@hotmail.co.jp"), person("Manabu", "Ishii", role = "aut"), person("Itoshi", "Nikaido", role = "aut"))
Suggests:   testthat
Depends:    R (>= 3.4.0)
Imports:    methods, fields, rTensor, plot3D, tagcloud, ggplot2
License:    Artistic-2.0
URL:        https://github.com/rikenbit/nnTensor
Author:     Koki Tsuyuzaki [aut, cre], Manabu Ishii [aut], Itoshi Nikaido [aut]
Maintainer: Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Index of help topics:

GabrielNMF     Gabriel-type Bi-Cross-Validation for Non-negative Matrix Factorization
jNMF           Joint Non-negative Matrix Factorization Algorithms (jNMF)
NMF            Non-negative Matrix Factorization Algorithms (NMF)
nnTensor-package Non-Negative Tensor Decomposition
NTD            Non-negative Tucker Decomposition Algorithms
nnTensor-package

(NTD)

NTF Non-negative CP Decomposition Algorithms (NTF)
plot.NMF Plot function of the result of NMF function
plotTensor2D Plot function for visualization of matrix data structure
plotTensor3D Plot function for visualization of tensor data structure
recTensor Tensor Reconstruction from core tensor (S) and factor matrices (A)
siNMF Simultaneous Non-negative Matrix Factorization Algorithms (siNMF)
toyModel Toy model data for using NMF, NTF, and NTD

Author(s)

NA

Maintainer: NA

References


Xiaoxu Han. (2007). CANCER MOLECULAR PATTERN DISCOVERY BY SUBSPACE CONSENSUS KERNEL CLASSIFICATION

GabrielNMF


See Also

toyModel,NMF,NTF,NTD,recTensor,plotTensor3D

Examples

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

GabrielNMF

**Gabriel-type Bi-Cross-Validation for Non-negative Matrix Factorization**

Description

The input data is assumed to be non-negative matrix. GabrielNMF devides the input file into four matrices (A, B, C, and D) and perform cross validation by the prediction of A from the matrices B, C, and D.

Usage

```r
GabrielNMF(X, J = 3, nx = 5, ny = 5, ...)
```

Arguments

- `X` The input matrix which has N-rows and M-columns.
- `J` The number of low-dimension (J < N, M).
- `nx` The number of hold-out in row-wise direction (2 < nx < N).
- `ny` The number of hold-out in row-wise direction (2 < ny < M).
- `...` Other parameters for NMF function.
Value
  TestRecError : The reconstruction error calculated by Gabriel-style Bi-Cross Validation.

Author(s)
  Koki Tsuyuzaki

References
  Art B. Owen et. al., (2009). Bi-Cross-Validation of the SVD and the Nonnegative Matrix Factor-
  ization. The Annals of Applied Statistics

Examples
  if(interactive()){
    # Test data
    matdata <- toyModel(model = "NMF"

    # Bi-Cross-Validation
    BCV <- rep(0, length=5)
    names(BCV) <- 2:6
    for(j in seq(BCV)){
      print(j+1)
      BCV[j] <- mean(GabrielNMF(matdata, J=j+1, nx=2, ny=2)$TestRecError)
    }
    proper.rank <- as.numeric(names(BCV)[which(BCV == min(BCV))])

    # NMF
    out <- NMF(matdata, J=proper.rank)
  }

Description
  The input data objects are assumed to be non-negative matrices. jNMF decomposes the matrices to
  two low-dimensional factor matrices simultaneously.

Usage
  jNMF(X, M=NULL, pseudocount=1e-10,
       initW=NULL, initV=NULL, initH=NULL, fixW=FALSE, fixV=FALSE,
       fixH=FALSE,
       L1_W=1e-10, L1_V=1e-10, L1_H=1e-10,
       L2_W=1e-10, L2_V=1e-10, L2_H=1e-10,
       J = 3, w=NULL, algorithm = c("Frobenius", "KL", "IS", "PLTF"),
       p=1, thr = 1e-10, num.iter = 100, viz = FALSE, figdir = NULL, verbose = FALSE)
Arguments

X  A list containing input matrices (X_k, <N*Mk>, k=1..K).
M  A list containing the mask matrices (X_k, <N*Mk>, k=1..K). If the input matrix has missing values, specify the element as 0 (otherwise 1).
pseudocount  The pseudo count to avoid zero division, when the element is zero (Default: 1e-10).
initW  The initial values of factor matrix W, which has N-rows and J-columns (Default: NULL).
initV  A list containing the initial values of multiple factor matrices (V_k, <N*J>, k=1..K, Default: NULL).
initH  A list containing the initial values of multiple factor matrices (H_k, <Mk*J>, k=1..K, Default: NULL).
fixW  Whether the factor matrix W is updated in each iteration step (Default: FALSE).
fixV  Whether the factor matrices Vk are updated in each iteration step (Default: FALSE).
fixH  Whether the factor matrices Hk are updated in each iteration step (Default: FALSE).
L1_W  Parameter for L1 regularitation (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
L1_V  Parameter for L1 regularitation (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
L1_H  Parameter for L1 regularitation (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
L2_W  Parameter for L2 regularitation (Default: 1e-10).
L2_V  Parameter for L2 regularitation (Default: 1e-10).
L2_H  Parameter for L2 regularitation (Default: 1e-10).
J  Number of low-dimension (J < N, Mk).
w  Weight vector (Default: NULL)
algorithm  Divergence between X and X_bar. "Frobenius", "KL", and "IS" are available (Default: "KL").
p  The parameter of Probabilistic Latent Tensor Factorization (p=0: Frobenius, p=1: KL, p=2: IS)
thr  When error change rate is lower than thr, the iteration is terminated (Default: 1E-10).
num.iter  The number of iteration 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 windows.
Value

W : A matrix which has N-rows and J-columns (J < N, Mk). V : A list which has multiple elements containing N-rows and J-columns (J < N, Mk). H : A list which has multiple elements containing Mk-rows and J-columns matrix (J < N, Mk). RecError : The reconstruction error between data matrix and reconstructed matrix from W and H. TrainRecError : The reconstruction error calculated by training set (observed values specified by M). TestRecError : The reconstruction error calculated by test set (missing values specified by M). RelChange : The relative change of the error.

Author(s)

Koki Tsuyuzaki

References


Examples

matdata <- toyModel(model = "siNMF_Hard")
out <- jNMF(matdata, J=2, num.iter=2)

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 matrices. This function is also used as initialization step of tensor decomposition (see also NTF and NTD).
NMF

Usage


Arguments

X
The input matrix which has N-rows and M-columns.

M
The mask matrix which has N-rows and M-columns. If the input matrix has missing values, specify the element as 0 (otherwise 1).

pseudocount
The pseudo count to avoid zero division, when the element is zero (Default: 1e-10).

initU
The initial values of factor matrix U, which has N-rows and J-columns (Default: NULL).

initV
The initial values of factor matrix V, which has M-rows and J-columns (Default: NULL).

fixU
Whether the factor matrix U is updated in each iteration step (Default: FALSE).

fixV
Whether the factor matrix V is updated in each iteration step (Default: FALSE).

L1_U
Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.

L1_V
Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.

L2_U
Parameter for L2 regularization (Default: 1e-10).

L2_V
Parameter for L2 regularization (Default: 1e-10).

J
The number of low-dimension (J < N, M). If a numerical vector is specified (e.g. 2:6), the appropriate rank is estimated.

rank.method
The rank estimation method (Default: "all"). Only if the J option is specified as a numerical vector longer than two, this option will be active.

runtime
The number of trials to estimate rank (Default: 10).

algorithm

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 iteration 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 window.

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). J : The number of dimension (J < N, M). RecError : The reconstruction error between data tensor and reconstructed tensor from U and V. TrainRecError : The reconstruction error calculated by training set (observed values specified by M). TestRecError : The reconstruction error calculated by test set (missing values specified by M). RelChange : The relative change of the error. Trial : All the results of the trials to estimate the rank. Runtime : The number of the trials to estimate the rank. RankMethod : The rank estimation method.

Author(s)

Koki Tsuyuzaki

References


Examples

```r
if(interactive()){  
  # Test data
  matdata <- toyModel(model = "NMF")

  # Simple usage
  out <- NMF(matdata, J=5)

  # Rank estimation mode (single method)
  out2 <- NMF(matdata, J=2:10, rank.method="ccc", runtime=3)
  plot(out2)

  # Rank estimation mode (all method)
  out3 <- NMF(matdata, J=2:10, rank.method="all", runtime=10)
  plot(out3)
}
```
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 matrices (A).

Usage


Arguments

X
The input tensor which has I1, I2, and I3 dimensions.

M
The mask tensor which has I1, I2, and I3 dimensions. If the mask tensor has missing values, specify the element as 0 (otherwise 1).

pseudocount
The pseudo count to avoid zero division, when the element is zero (Default: 1e-10).

initS
The initial values of core tensor which has J1, J2, and J3 dimensions (Default: NULL).

initA
A list containing the initial values of multiple factor matrices (A_k, <Ik*Jk>, k=1..K, Default: NULL).

fixS
Whether the core tensor S is updated in each iteration step (Default: FALSE).

fixA
Whether the factor matrices Ak are updated in each iteration step (Default: FALSE).

L1_A
Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.

L2_A
Parameter for L2 regularization (Default: 1e-10).

rank
The number of low-dimension in each mode (J1, J2, J3, J1<I1, J2<I2, J3 < I3) (Default: c(3,3,3)).

modes
The vector of the modes on which to perform the decomposition (Default: 1:3 <all modes>).

algorithm
**nmf.algorithm**  
NMF algorithms, when the algorithm is "NMF", "Frobenius", "KL", "IS", "Pearson", "Hellinger", "Neyman", "Alpha", "Beta", "PGD", "HALS", "GCD", "Projected", "NHR", "DTPP", "Orthogonal", are "OrthReg" 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 iteration step (Default: 100).

**num.iter2**  
The number of NMF iteration step, when the algorithm is "NMF" (Default: 10).

**viz**  
If viz == TRUE, internal reconstructed tensor can be visualized.

**figdir**  
The directory for saving the figure, when viz == TRUE (Default: NULL).

**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.  
RecError : The reconstruction error between data tensor and reconstructed tensor from S and A.  
TrainRecError : The reconstruction error calculated by training set (observed values specified by M).  
TestRecError : The reconstruction error calculated by test set (missing values specified by M).  
RelChange : The relative change of the error.

**Author(s)**

Koki Tsuyuzaki

**References**


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 matrices (A).

Usage

NTF(X, M=NULL, pseudocount=1e-10, initA=NULL,
    fixA=FALSE, L1_A=1e-10, L2_A=1e-10, rank = 3,
                  "HALS", "Alpha-HALS", "Beta-HALS", "Alpha", "Beta"),
    init = c("NMF", "ALS", "Random"), Alpha = 1,
    Beta = 2, thr = 1e-10, num.iter = 100, viz = FALSE,
    figdir = NULL, verbose = FALSE)

Arguments

- **X** The input tensor which has I1, I2, and I3 dimensions.
- **M** The mask tensor which has I1, I2, and I3 dimensions. If the mask tensor has missing values, specify the element as 0 (otherwise 1).
- **pseudocount** The pseudo count to avoid zero division, when the element is zero (Default: 1e-10).
- **initA** A list containing the initial values of multiple factor matrices (A_k, <Ik*Jk>, k=1..K, Default: NULL).
- **fixA** Whether the factor matrices Ak are updated in each iteration step (Default: FALSE).
- **L1_A** Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
- **L2_A** Parameter for L2 regularization (Default: 1e-10).
- **rank** The number of low-dimension in each mode (J1=J2=J3, J1<I1, J2<I2, J3 < I3) (Default: 3).
- **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 thr, the iteration is terminated (Default: 1E-10).

num.iter

The number of iteration step (Default: 100).

viz

If viz == TRUE, internal reconstructed tensor can be visualized.

figdir

the directory for saving the figure, when viz == TRUE (Default: NULL).

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. RecError : The reconstruction error between data tensor and reconstructed tensor from S and A. TrainRecError : The reconstruction error calculated by training set (observed values specified by M). TestRecError : The reconstruction error calculated by test set (missing values specified by M). RelChange : The relative change of the error.

Author(s)

Koki Tsuyuzaki

References


See Also

plotTensor3D

Examples

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

Plot function of the result of NMF function

Description

Only if J is specified as a vector longer than 1, this function will be active.

Usage

plot(x, ...)

Arguments

x          The result of NMF function (NMF class).
...

Optional parameter for plot.

Value

A ggplot will be generated.

Author(s)

Koki Tsuyuzaki

References

Xiaoxu Han. (2007). CANCER MOLECULAR PATTERN DISCOVERY BY SUBSPACE CONSENSUS KERNEL CLASSIFICATION
**plotTensor2D**

**Examples**

```r
plot.NMF
```

---

**plotTensor2D**  
*Plot function for visualization of matrix data structure*

**Description**

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

**Usage**

```r
plotTensor2D(X = NULL, method=c("sd", "mad"),
                sign=c("positive", "negative", "both"), thr=2)
```

**Arguments**

- **X**: Matrix object.
- **method**: Cutoff method to focus on large/small value in the tensor data (Default: "sd").
- **sign**: Direction to cutoff the large/small value in the tensor data (Default: "positive").
- **thr**: Threshold of cutoff method (Default: 2).

**Author(s)**

Koki Tsuyuzaki

**Examples**

```r
tensordata <- toyModel(model = "CP")
out <- NTF(tensordata, rank=3, num.iter=2)
tmp <- tempdir()
png(filename=paste0(tmp, "/NTF.png"))
plotTensor2D(out$A[[1]])
dev.off()
```
plotTensor3D

Plot function for visualization of tensor data structure

Description

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

Usage

plotTensor3D(X = NULL, method=c("sd", "mad"),
              sign=c("positive", "negative", "both"), thr=2)

Arguments

X Tensor object, which is defined as S4 class of rTensor package.
method Cutoff method to focus on large/small value in the tensor data (Default: "sd").
sign Direction to cutoff the large/small value in the tensor data (Default: "positive").
thr Threshold of cutoff method (Default: 2).

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 tensor structure can be visualized.
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 multiplication (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 multiplied 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)

simNMF

Simultaneous Non-negative Matrix Factorization Algorithms (siNMF)

Description

The input data objects are assumed to be non-negative matrices. siNMF decompose the matrices to two low-dimensional factor matrices simultaneously.

Usage

siNMF(X, M=NULL, pseudocount=1e-10, initW=NULL, initH=NULL, fixW=FALSE, fixH=FALSE, L1_W=1e-10, L1_H=1e-10, L2_W=1e-10, L2_H=1e-10, J = 3, w=NULL, algorithm = c("Frobenius", "KL", "IS", "PLTF"), p=1, thr = 1e-10, num.iter = 100, viz = FALSE, figdir = NULL, verbose = FALSE)
Arguments

- **X**: A list containing the input matrices (X_k, <N*M_k>, k=1..K).
- **M**: A list containing the mask matrices (X_k, <N*M_k>, k=1..K). If the input matrix has missing values, specify the element as 0 (otherwise 1).
- **pseudocount**: The pseudo count to avoid zero division, when the element is zero (Default: 1e-10).
- **initW**: The initial values of factor matrix W, which has N-rows and J-columns (Default: NULL).
- **initH**: A list containing the initial values of multiple factor matrices (H_k, <M_k*J>, k=1..K, Default: NULL).
- **fixW**: Whether the factor matrix W is updated in each iteration step (Default: FALSE).
- **fixH**: Whether the factor matrices H_k are updated in each iteration step (Default: FALSE).
- **L1_W**: Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
- **L1_H**: Parameter for L1 regularization (Default: 1e-10). This also works as small positive constant to prevent division by zero, so should be set as 0.
- **L2_W**: Parameter for L2 regularization (Default: 1e-10).
- **L2_H**: Parameter for L2 regularization (Default: 1e-10).
- **J**: Number of low-dimension (J < N, M_k).
- **w**: Weight vector (Default: NULL).
- **algorithm**: Divergence between X and X_bar. "Frobenius", "KL", and "IS" are available (Default: "KL").
- **p**: The parameter of Probabilistic Latent Tensor Factorization (p=0: Frobenius, p=1: KL, p=2: IS)
- **thr**: When error change rate is lower than thr, the iteration is terminated (Default: 1E-10).
- **num.iter**: The number of iteration 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

- **W**: A matrix which has N-rows and J-columns (J < N, M_k).
- **H**: A list which has multiple elements containing M_k-rows and J-columns matrix (J < N, M_k).
- **RecError**: The reconstruction error between data matrix and reconstructed matrix from W and H.
- **TrainRecError**: The reconstruction error calculated by training set (observed values specified by M).
- **TestRecError**: The reconstruction error calculated by test set (missing values specified by M).
- **RelChange**: The relative change of the error.

Author(s)

Koki Tsuyuzaki
References


Examples

```r
matdata <- toyModel(model = "NMF_Easy")
out <- siNMF(matdata, J=2, num.iter=2)
```

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

Description

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

Usage

```r
toyModel(model = "CP", seeds=123)
```

Arguments

- **model**: Single character string is specified. "NMF", "CP", and "Tucker" are available (Default: "CP").
- **seeds**: Random number for setting set.seeds in the function (Default: 123).

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", seeds=123)
tensordata1 <- toyModel(model = "CP", seeds=123)
tensordata2 <- toyModel(model = "Tucker", seeds=123)
Index

* methods
  GabrielNMF, 4
  jNMF, 5
  NMF, 7
  NTD, 10
  NTF, 12
  plot.NMF, 14
  plotTensor2D, 15
  plotTensor3D, 16
  recTensor, 16
  siNMF, 17
  toyModel, 19

* package
  nnTensor-package, 2

GabrielNMF, 4

jNMF, 5

NMF, 4, 7, 19
nnTensor (nnTensor-package), 2
nnTensor-package, 2
NTD, 4, 10, 17, 19
NTF, 4, 12, 17, 19

plot (plot.NMF), 14
plot.NMF, 14
plotTensor2D, 15
plotTensor3D, 4, 11, 13, 16

recTensor, 4, 16

siNMF, 17

toyModel, 4, 19