Package: ttTensor (via r-universe)

May 7, 2026

Type Package

Title Tensor-Train Decomposition

Version 1.0.2

Date 2025-08-25

Suggests testthat

Depends R (>= 3.5.0)

Imports methods, rTensor, PTAk, Matrix

Description Tensor-train is a compact representation for higher-order tensors. Some algorithms for performing tensor-train decomposition are available such as TT-SVD, TT-WOPT, and TT-Cross. For the details of the algorithms, see I. V. Oseledets (2011) <doi:10.1137/090752286>, Yuan Longao, et al (2017) <doi:10.48550/arXiv.1709.02641>, I. V. Oseledets (2010) <doi:10.1016/j.laa.2009.07.024>.

License MIT + file LICENSE

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

Repository https://rikenbit.r-universe.dev

Date/Publication 2025-09-06 04:43:25 UTC

RemoteUrl https://github.com/rikenbit/tttensor

RemoteRef HEAD

RemoteSha 641368ddc60db9f7516addb48602336cb119f1f9

Contents

`ttTensor-package` *Tensor-Train Decomposition*

Description

Tensor-train is a compact representation for higher-order tensors. Some algorithms for performing tensor-train decomposition are available such as TT-SVD, TT-WOPT, and TT-Cross. For the details of the algorithms, see I. V. Oseledets (2011) <doi:10.1137/090752286>, Yuan Longao, et al (2017) <doi:10.48550/arXiv.1709.02641>, I. V. Oseledets (2010) <doi:10.1016/j.laa.2009.07.024>.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Koki Tsuyuzaki [aut, cre], Manabu Ishii [aut], Itoshi Nikaido [aut]

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

References

I. V. Oseledets, (2011). Tensor-Train Decomposition. *SIAM J. SCI. COMPUT.*

Yuan, Longhao, et. al., (2017). Completion of high order tensor data with missing entries via tensor-train decomposition. *International Conference on Neural Information Processing*

I. V. Oseledets, et. al., (2010). TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*

Ali Civril, et. al., (2009). On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*

See Also

`TTSVD`,`TTWOPT`,`TTCross`,`skeleton.decomp`,`maxvol`

Examples

```
ls("package:ttTensor")
```

as_sptensor *Convert to Simple Sparse Tensor*

Description

Converts an array or matrix to a simple sparse tensor format. This is a minimal implementation to replace the tensorr dependency.

Usage

```
as_sptensor(x)
```

Arguments

x	An array or matrix to convert

Details

This function provides a minimal sparse tensor implementation to support the TTCross function without requiring the archived tensorr package. For production use with actual sparse data, consider using specialized sparse tensor packages.

Value

A simple_sparse_tensor object

Examples

```
# Create a 3D array
x <- array(rnorm(24), dim = c(2, 3, 4))

# Convert to sparse tensor format
sparse_x <- as_sptensor(x)
```

dtensor *Dense Tensor Creation*

Description

Creates a dense tensor representation. This is a compatibility function that simply returns the input as-is.

Usage

```
dtensor(x)
```

Arguments

x	An array or matrix

Details

This function is provided for compatibility with code that previously used the tensorr package. It simply returns the input without modification.

Value

The input array or matrix unchanged

Examples

```
# Create a 3D array
x <- array(rnorm(24), dim = c(2, 3, 4))

# Create dense tensor (returns x unchanged)
dense_x <- dtensor(x)
```

---

maxvol *maxvol algorithm*

Description

maxvol finds the r*r submatrix of maximal volume in C (n*r) by greedily searching the vector of max norm, and subtractction of its projection from the rest of rows. See also http://tensorly.org/stable/_modules/tensorly/contr

Usage

```
maxvol(C)
```

Arguments

C	The input sparse matrix.

Value

row_idx : The indices of rows, which make the determinant as large

Author(s)

Koki Tsuyuzaki

References

Ali Civril, et. al., (2009). On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*

See Also

skeleton.decomp

Examples

```
library("Matrix")
# Matrix data
X3 <- matrix(runif(10*20), nrow=10)
X3 <- as(X3, "sparseMatrix")
# Skeleton Decomposition
out.SKD <- skeleton.decomp(X3, r=3, num.iter=2, thr=1E-5)
```

---

skeleton.decomp *Skeleton Decomposition*

Description

skeleton.decomp decomposes the input sparse matrix (n*m) and return the three matrices C (n*r), U (r*r), and R (r*m). Only sparse matrix defined by the Matrix package is acceptable as the input.

Usage

```
skeleton.decomp(A, r, thr=1E-10, num.iter=30)
```

Arguments

A	The input sparse matrix.
r	Rank parameter to specify the lower dimension (r <= min(A)).
thr	The threshold to determine the convergence (Default: 1E-10).
num.iter	The number of iteration (Default: 30).

Value

C : A[I, :] U : inverse(A[I, J]) R : A[:, J] rowidx :The indices of rows colidx : The indices of columns RecError : The reconstruction error between data matrix and reconstructed matrix from C, U, and R RelChange : The relative change of the error

Author(s)

Koki Tsuyuzaki

References

I. V. Oseledets, et. al., (2010). TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*

See Also

maxvol

Examples

```
library("Matrix")
# Matrix data
X3 <- matrix(runif(10*20), nrow=10)
X3 <- as(X3, "sparseMatrix")
# Skeleton Decomposition
out.SKD <- skeleton.decomp(X3, r=3, num.iter=2, thr=1E-5)
```

TTCross *Tensor-Train Decomposition by TRCross*

Description

TTCross incrementaly decomposes the input tensor by skeleton decomposition algorithm. The algorithm only select the row/column indices and any large temporal matrix are genrated in the process. Therefore, this method is suitable for the sparse tensor.

Usage

```
TTCross(A, Ranks=NULL, thr=1E-10, num.iter=30)
```

Arguments

A	The input sparse tensor.
Ranks	TT-ranks to specify the lower dimensions.
thr	The threshold to determine the convergence (Default: 1E-10).
num.iter	The number of iteration (Default: 30).

Value

G : Core tensors

Author(s)

Koki Tsuyuzaki

References

I. V. Oseledets, et. al., (2010). TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*

Examples

```
# TTCross requires sparse tensor input
# Creating a simple example
library("rTensor")
X1 <- array(rnorm(3*4*5), c(3,4,5))
X1 <- as.tensor(X1)
# Convert to sparse format
X2 <- as_sptensor(dtensor(X1@data))
# TT-ranks (should be less than dimensions)
Ranks <- c(p=2, q=2)
# Note: TTCross is designed for sparse tensors
# and may have numerical issues with some inputs
tryCatch({
  out.TTCross <- TTCross(X2, Ranks, num.iter=2)
  print("TTCross completed")
}, error = function(e) {
  print("TTCross encountered an error - this function is experimental")
})
```

TTSVD *Tensor-Train Decomposition by TTSVD*

Description

TTSVD incrementaly decomposes the input tensor by singular value decomposition (SVD).

Usage

```
TTSVD(A, Ranks=NULL, accuracy=NULL)
```

Arguments

A	The input tensor.
Ranks	TT-ranks to specify the lower dimensions.
accuracy	The accuracy of the compression.

Value

G : Core tensors

Author(s)

Koki Tsuyuzaki

References

I. V. Oseledets, (2011). Tensor-Train Decomposition. *SIAM J. SCI. COMPUT.*

Examples

```
library("rTensor")
# Tensor data
X1 <- array(rnorm(3*5*7*9*11), c(3,5,7,9,11))
dimnames(X1) <- list(
    I=paste0("i", 1:3),
    J=paste0("j", 1:5),
    K=paste0("k", 1:7),
    L=paste0("l", 1:9),
    M=paste0("m", 1:11)
    )
X1 <- as.tensor(X1)
# TT-ranks
Ranks <- c(p=2, q=4, r=6, s=8)
# TTSVD
out.TTSVD <- TTSVD(X1, Ranks)
out.TTSVD <- TTSVD(X1, accuracy=1E-10)
```

TTWOPT *Tensor-Train Decomposition by Tensor-train Weighted OPTimization*

Description

TTWOPT incrementaly decomposes the input tensor by gradient desecent. The tensor with missing entries is also specified with weight tensor W.

Usage

```
TTWOPT(X, Ranks, W=NULL, eta=1E-7, thr=1E-10, num.iter=100)
```

Arguments

X	The input tensor.
Ranks	TT-ranks to specify the lower dimensions.
W	The weight tensor to specify the missing entries (0: missing, 1: existing). The size must be same as that of X.
eta	The learning rate parameter of the gradient descent algorithm (Default : 1E-10).
thr	The threshold to determine the convergence (Default: 1E-10).
num.iter	The number of iteration (Default: 30).

Value

G : Core tensors RelChange : The relative change of the error f : The values of the object function RecError : The reconstruction error between data tensor and reconstructed tensor from C, U, and R

Author(s)

Koki Tsuyuzaki

References

Yuan, Longhao, et. al., (2017). Completion of high order tensor data with missing entries via tensor-train decomposition. *International Conference on Neural Information Processing*

Examples

```
library("rTensor")
# Tensor data
X1 <- array(rnorm(3*5*7*9*11), c(3,5,7,9,11))
dimnames(X1) <- list(
    I=paste0("i", 1:3),
    J=paste0("j", 1:5),
    K=paste0("k", 1:7),
    L=paste0("l", 1:9),
    M=paste0("m", 1:11)
    )
X1 <- as.tensor(X1)
# TT-ranks
Ranks <- c(p=2, q=4, r=6, s=8)
# TTWOPT
out.TTWOPT <- TTWOPT(X1, Ranks, eta=1E-7)
```

`unfold` *Unfold a Tensor*

Description

Unfolds a tensor along a specified mode into a matrix representation.

Usage

```
unfold(x, mode)
```

Arguments

`x`	A tensor object (simple_sparse_tensor, Tensor, array, or matrix)
`mode`	The mode along which to unfold the tensor

Details

This function unfolds a tensor along the specified mode into a matrix. It supports simple_sparse_tensor objects, rTensor Tensor objects, and regular arrays/matrices. The function uses rTensor's rs_unfold internally.

Value

A matrix representation of the unfolded tensor

Examples

```
library(rTensor)

# Create a 3D tensor
x <- array(rnorm(24), dim = c(2, 3, 4))
tensor_x <- as.tensor(x)

# Unfold along mode 1 (using ttTensor's unfold function)
unfolded <- ttTensor::unfold(tensor_x, mode = 1)
```

Index