Package 'otTensor'

May 13, 2026

Type Package

Title Optimal Tensor Transport

Version 0.99.0

Depends R (>= 3.4.0)

Imports methods,

Suggests rTensor, knitr, rmarkdown, testthat

Description An optimal transport (OT) method, which can handle tensors of any order by learning possibly multiple transport plans. For the details of the methods, see Kerdoncuff et al. (2022) <doi:10.1609/aaai.v36i7.20695>.

License MIT + file LICENSE

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

VignetteBuilder knitr

NeedsCompilation no

Author Koki Tsuyuzaki [aut, cre]

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

Repository CRAN

Date/Publication 2026-05-13 07:30:17 UTC

Contents

otTensor-package *Optimal Tensor Transport*

Description

An optimal transport (OT) method, which can handle tensors of any order by learning possibly multiple transport plans. For the details of the methods, see Kerdoncuff et al. (2022) <doi:10.1609/aaai.v36i7.20695>.

Details

The DESCRIPTION file:

Package:	otTensor
Type:	Package
Title:	Optimal Tensor Transport
Version:	0.99.0
Authors@R:	c(person("Koki", "Tsuyuzaki", role = c("aut", "cre"), email = "k.t.the-answer@hotmail.co.jp"))
Depends:	R (>= 3.4.0)
Imports:	methods,
Suggests:	rTensor, knitr, rmarkdown, testthat
Description:	An optimal transport (OT) method, which can handle tensors of any order by learning possibly multiple tra
License:	MIT + file LICENSE
URL:	https://github.com/rikenbit/otTensor
VignetteBuilder:	knitr
Author:	Koki Tsuyuzaki [aut, cre]
Maintainer:	Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Index of help topics:

```
OTT                     Optimal Tensor Transport
otTensor-package        Optimal Tensor Transport
```

Author(s)

NA

Maintainer: NA

References

Kerdoncuff, T. et al., (2022). Optimal Tensor Transport. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7), 7124-7132.

See Also

OTT

Examples

```
ls("package:otTensor")
```

OTT *Optimal Tensor Transport*

Description

Transport plans to align two tensors X and Y are estimated.

Usage

```
OTT(X, Y, f, ps=NULL, qs=NULL,
    loss=.absolute_error, num.sample=1000,
    num.iter=200, epsilon=1e-10, verbose=FALSE)
```

Arguments

X	The first tensor data ('rTensor' object). The order must be the same as that of Y.
Y	The second tensor data ('rTensor' object). The order must be the same as that of X.
f	Affectation function to assign transport plan to each pair of mode of X and Y.
ps	Row-wise weight vectors for transport plans (Default: NULL, which means uniform distribution).
qs	Column-wise weight vectors for transport plans (Default: NULL, which means uniform distribution).
loss	Loss function (Default: .absolute_error).
num.sample	Number of samples to calculate the gradient (Default: 1000).
num.iter	Number of iterations (Default: 200).
epsilon	Regularization parameter (Default: 1e-10).
verbose	Verbose option (Default: FALSE).

Value

Ts : A list contains transport plans.

Author(s)

Koki Tsuyuzaki

Examples

```
library("rTensor")
D <- 3
A <- 2
Is <- c(4, 4, 5)
Ks <- c(6, 6, 7)
f <- c(1, 1, 2)
arrX <- array(rep(0, prod(Is)), Is)
arrY <- array(rep(0, prod(Ks)), Ks)

for (i1 in 1:Is[1]) {
    for (i2 in 1:Is[2]) {
        for (i3 in 1:Is[3]) {
            arrX[i1, i2, i3] <- i1 + i2 + i3
        }
    }
}
for (k1 in 1:Ks[1]) {
    for (k2 in 1:Ks[2]) {
        for (k3 in 1:Ks[3]) {
            arrY[k1, k2, k3] <- k1 + k2 + k3
        }
    }
}

ps <- list()
for (a in 1:A) {
    ds <- which(f == a)
    d <- ds[1]
    length_of_p_a <- dim(arrX)[d]
    ps[[a]] <- rep(0.01, length_of_p_a); ps[[a]][c(1, 3)] <- 1
    ps[[a]] <- ps[[a]] / sum(ps[[a]])
}
qs <- list()
for (a in 1:A) {
    ds <- which(f == a)
    d <- ds[1]
    length_of_q_a <- dim(arrY)[d]
    qs[[a]] <- rep(1, length_of_q_a); qs[[a]][c(2, 3)] <- 0
    qs[[a]] <- qs[[a]] / sum(qs[[a]])
}

# Test Dataset
X <- as.tensor(arrX)
Y <- as.tensor(arrY)

# This is just for an example.
# In real data analysis,
# please specify larger num.sample and num.iter such as 1000 and 200, respectively.
OTT(X = X, Y = Y, f = f,
    ps=ps, qs=qs, num.sample=10,
    loss = function (x, y) {abs(x - y)},
```

```
        num.iter=2, epsilon=1e-10)
```

Index