Package: gridOT (via r-universe)

October 13, 2024

Type Package

Title Approximate Optimal Transport Between Two-Dimensional Grids

Version 1.0.1

Date 2022-10-18

Description Can be used for optimal transport between two-dimensional grids with respect to separable cost functions of l^p form. It utilizes the Frank-Wolfe algorithm to approximate so-called pivot measures: one-dimensional transport plans that fully describe the full transport, see G. Auricchio (2021) <arXiv:2105.07278>. For these, it offers methods for visualization and to extract the corresponding transport plans and costs. Additionally, related functions for one-dimensional optimal transport are available.

License GPL-3

Imports Rcpp (>= 1.0.8.3)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.2.1

Encoding UTF-8

NeedsCompilation yes

Author Michel Groppe [aut, cre], Nicholas Bonneel [ctb], Egerváry Research Group on Combinatorial Optimization [cph]

Maintainer Michel Groppe <`michel.groppe@stud.uni-goettingen.de`>

Repository CRAN

Date/Publication 2022-10-18 23:05:13 UTC

Contents

dual1d 2
north_west_corner 3
otgrid 4
pivot_measure 5

plot.otgrid 7
transport_cost.numeric 8
transport_costmat 10
transport_df.numeric 11

Index **13**

`dual1d` *Dual Solution of one-dimensional Optimal Transport*

Description

Calculate an optimal dual pair for the optimal transport between discrete one-dimensional measures.

Usage

```
dual1d(a, b, wa, wb, p = 1, right.margin = 1e-15, sorted = FALSE)
```

Arguments

`a`	first vector of points.
`b`	second vector of points.
`wa`	weight vector of the first vector of points.
`wb`	weight vector of the second vector of points.
`p`	the power ≥ 1 of the cost function.
`right.margin`	small amount the points are moved by.
`sorted`	logical value indicating whether or not `a` and `b` are sorted.

Details

The pair f, g is an optimal dual pair if the optimal transport distance between the two distributions with respect to the cost function $c(x, y) = |x - y|^p$ is given by

$$\langle f, w_a \rangle + \langle g, w_b \rangle$$

and the condition $f_i + g_j \leq |a_i - b_j|^p$ holds.

Value

a list containing the dual vectors `pot.a` and `pot.b`.

Examples

```
set.seed(1)
a <- 1:5
wa <- rep(1/5, 5)
b <- 1:6
wb <- runif(6)
wb <- wb / sum(wb)

d <- dual1d(a, b, wa, wb, p = 1)

dc <- sum(d$pot.a * wa) + sum(d$pot.b * wb)
print(all.equal(dc, transport_cost(a, b, wa, wb, p = 1)))
```

`north_west_corner` *North-west-corner Rule*

Description

Calculate the transport plan obtained by the north-west-corner rule.

Usage

```
north_west_corner(wx, wy, threshold = 1e-15)
```

Arguments

`wx`	first weight vector.
`wy`	second weight vector.
`threshold`	small value that indicates when a value is considered to be zero.

Value

a matrix representing the transport plan obtained by the north-west-corner rule.

Examples

```
set.seed(1)
wx <- rep(1/5, 5)
wy <- runif(6)
wy <- wy / sum(wy)
north_west_corner(wx, wy)
```

otgrid *Two-dimensional Grid with Mass*

Description

Create an object that represents a probability measure that is supported on a two-dimensional grid.

Usage

```
otgrid(
  mass,
  x = NULL,
  y = NULL,
  sorted = FALSE,
  normalize = FALSE,
  remove.zeros = FALSE
)
```

Arguments

`mass`	matrix of non-negative weights.
`x`	vector of support points of the first marginal.
`y`	vector of support points of the second marginal.
`sorted`	logical value specifying whether or not the support points are sorted.
`normalize`	logical value specifying whether or not the total mass should be rescaled to 1.
`remove.zeros`	logical value specifying whether or not marginals with no mass should be removed from the grid.

Details

If `x` and `y` are not specified, then a equispaced unit grid is chosen.

Value

object of class `"otgrid"`. It contains the following elements:

`x`	vector of support points of the first marginal
`y`	vector of support points of the second marginal
`n`	number of support points of the first marginal
`m`	number of support points of the second marginal
`mass`	matrix of non-negative weights
`total`	total mass

Also note that the functions `print` and `plot` are available for objects of class `"otgrid"`.

See Also

plot `plot.otgrid`

Examples

```
x <- otgrid(cbind(1:2, 0, 3:4), remove.zeros = TRUE)

print(x) # note that it's only 2 x 2
plot(x)
```

`pivot_measure` *Pivot Measure*

Description

Calculate the pivot measure of the optimal transport between two-dimensional grids.

Usage

```
pivot_measure(x, ...)

## S3 method for class 'otgrid'
pivot_measure(
  x,
  y,
  p.1 = 2,
  p.2 = p.1,
  max.it = 100,
  tol = 1e-04,
  threads = 1,
  start.pivot = c("independent", "northwestcorner"),
  dual.method = c("discrete", "epsilon-discrete", "epsilon-histogram"),
  dual.params = list(),
  return.it = FALSE,
  ...
)

## S3 method for class 'data.frame'
pivot_measure(x, a, b, p.1 = 2, p.2 = p.1, ...)
```

Arguments

x	an object of class "`otgrid`" the mass is to be transported from or a data frame with columns `from`, `to` and `mass` specifying the optimal transport plan.
...	further arguments (currently unused).
y	an object of class "`otgrid`" the mass is to be transported to.

`p.1`	the first power ≥ 1 of the cost.
`p.2`	the second power ≥ 1 of the cost.
`max.it`	the maximum number of iterations.
`tol`	the desired accuracy.
`threads`	number of threads to use.
`start.pivot`	the start pivot to use: matrix representing an actual coupling or `"independent"` or `"northwestcorner"`.
`dual.method`	the name of the dual calculators to use: `"discrete"`, `"epsilon-discrete"` or `"epsilon-histogram"`.
`dual.params`	list of parameters for the dual calculators.
`return.it`	logical value specifying whether or not the costs and dual gaps in each iteration are to be returned .
`a`	an object of class `"otgrid"` the mass is to be transported from.
`b`	an object of class `"otgrid"` the mass is to be transported to.

Details

Denote with $X_1 \times X_2$ and $Y_1 \times Y_2$ the two-dimensional grids that `x` and `y` lie on, respectively. The pivot measure is a measure on $Y_1 \times X_2$ that specifies the whole transport between `x` and `y` given that the cost is of the separable form $c(x, y) = |x_1 - y_1|^{p_1} + |x_2 - y_2|^{p_2}$.

The pivot measure is approximated using the Frank-Wolfe algorithm. The algorithm starts with an initial guess (`start.pivot`), e.g., the independent coupling (`"independent"`) or the north-west-corner rule (`"northwestcorner"`). Then, in each iteration step, the dual solutions of multiple one-dimensional transport problems are calculated and combined to give the gradient. To ensure differentiability, the dual solutions must be unique. There are three different calculators for the dual solutions, of which the last two ensure uniqueness:

name	description
`"discrete"`	the distributions are basically unchanged, due to rounding errors the support points are moved by a
`"epsilon-discrete"`	additionally, the support is made connected by uniformly adding ε mass.
`"epsilon-histogram"`	additionally, each point mass is distributed uniformly in a bin around said point

A pivot measure can also be computed from an optimal transport plan.

Finally, the pivot measure can be used to calculate the full transport plan and cost.

Value

an object of class `"otgridtransport"` that contains the following elements:

`from`	an object of class `"otgrid"` the mass is transported from.
`to`	an object of class `"otgrid"` the mass is transported to.
`p.1`	the first power ≥ 1 of the cost function.
`p.2`	the second power ≥ 1 of the cost function.
`pivot`	a matrix representing the pivot measure.

If the Frank-Wolfe algorithm is used to approximate the pivot measure, the element `conv` indicates whether or not we can ensured that the error is less or equal to the given precision `tol`.

If `return.it` = `TRUE`, then it also contains the vectors `costs` and `dualgaps` of costs and dual gaps in each iteration.

Also note that the functions `print` and `plot` are available for objects of class `"otgridtransport"`.

See Also

transport plan `transport_df`, transport cost `transport_cost`, two-dimensional grid `otgrid`, plot `plot.otgridtransport`

Examples

```
x <- otgrid(rbind(c(0, 0, 0, 0, 0, 0, 0, 0, 0),
                 c(0, 0, 1, 0, 0, 0, 0, 0, 0),
                 c(0, 1, 2, 1, 0, 0, 0, 0, 0),
                 c(0, 0, 1, 0, 0, 0, 0, 0, 0),
                  0, 0, 0, 0, 0))
y <- otgrid(rbind(0, 0, 0, 0,
                 c(0, 0, 0, 0, 0, 0, 0, 0, 0),
                 c(0, 0, 0, 0, 0, 0, 1, 0, 0),
                 c(0, 0, 0, 0, 0, 1, 2, 1, 0),
                 c(0, 0, 0, 0, 0, 0, 1, 0, 0), 0))

pm <- pivot_measure(x, y, p.1 = 1, p.2 = 2, dual.method = "epsilon-discrete")

print(pm)
plot(pm)

# use pivot measure to calculate cost and plan
pm <- transport_cost(pm)
pm <- transport_df(pm)
print(pm)

# calculate pivot from plan
pm2 <- pivot_measure(pm$df, x, y)
plot(pm2)
```

`plot.otgrid` *Plots for two-dimensional Optimal Transport*

Description

Plot two-dimensional grids or visualize the pivot measure.

Usage

```
## S3 method for class 'otgrid'
plot(x, num.col = 256, useRaster = TRUE, ...)

## S3 method for class 'otgridtransport'
plot(x, num.col = 256, useRaster = TRUE, back.col = "lightblue", ...)
```

Arguments

`x`	an object of class `"otgrid"` or `"otgridtransport"`
`num.col`	number of colors (grey) to use.
`useRaster`	parameter passed to the `image` function.
`...`	further arguments (currently unused).
`back.col`	color of the background.

Details

For objects of class `"otgrid"`, the grid is plotted as a greyscale image.

For objects of class `"otgridtransport"`, the pivot measure is visualized as follows: the two grids of the transport are plotted in the left upper (`from`) and right lower (`to`) corner. The pivot measure is in the left lower corner such that the marginals match.

Value

No return value, called for side effects.

See Also

pivot measure `pivot_measure`, grid `otgrid`

`transport_cost.numeric`

Optimal Transport Cost

Description

Calculate the optimal transport cost.

Usage

```
## S3 method for class 'numeric'
transport_cost(x, y, wx, wy, p = 1, sorted = FALSE, threshold = 1e-15, ...)

transport_cost(x, ...)

## S3 method for class 'otgridtransport'
```

```
transport_cost(x, threshold = 1e-15, ...)

## S3 method for class 'otgrid'
transport_cost(x, ...)

## S3 method for class 'data.frame'
transport_cost(x, costm, ...)
```

Arguments

`x`	a vector of points; a data frame with columns `from`, `to` and `mass` specifying the optimal transport plan; an object of class `"otgridtransport"` or `"otgrid"`, in the latter case `...` must be the arguments of `pivot_measure`.
`y`	second vector of points.
`wx`	weight vector of the first vector of points.
`wy`	weight vector of the second vector of points.
`p`	the power ≥ 1 of the cost function.
`sorted`	logical value indicating whether or not `a` and `b` are sorted.
`threshold`	small value that indicates when a value is considered to be zero.
`...`	further arguments (for `pivot_measure` if `x` is an object of class `"otgrid"`).
`costm`	cost matrix of the transport

Details

In case of two-dimensional grids, the pivot measure is used to calculate the optimal transport cost.

For one-dimensional optimal transport, the cost function is given by $c(x, y) = |x - y|^p$. In this case, the north-west-corner algorithm is used.

Value

the optimal transport cost or, in case of two-dimensional case, an object of class `"otgridtransport"` with element `cost` that contains it.

See Also

pivot measure `pivot_measure`

Examples

```
## one-dimensional example
set.seed(1)
a <- 1:5
wa <- rep(1/5, 5)
b <- 1:6
wb <- runif(6)
wb <- wb / sum(wb)
transport_cost(a, b, wa, wb, p = 1)
```

```
## two-dimensional example
x <- otgrid(cbind(0:1, 1:0))
y <- otgrid(cbind(1:0, 0:1))

# first calculate pivot manually
pm <- pivot_measure(x, y)
pm <- transport_cost(pm)
print(pm$cost)

# or just
pm2 <- transport_cost(x, y)
print(pm2$cost)

# or from transport plan and cost matrix
costm <- transport_costmat(pm)
tp <- transport_df(pm)
print(transport_cost(tp$df, costm))
```

`transport_costmat` *Cost Matrix for two-dimensional Optimal Transport*

Description

Calculate the cost matrix for the optimal transport between two-dimensional grids with respect to the cost function

$$c(x, y) = |x_1 - y_1|^{p_1} + |x_2 - y_2|^{p_2}.$$

Usage

```
transport_costmat(x, ...)

## S3 method for class 'otgrid'
transport_costmat(x, y, p.1 = 2, p.2 = p.1, ...)

## S3 method for class 'otgridtransport'
transport_costmat(x, ...)
```

Arguments

`x`	an object of class `"otgridtransport"` or `"otgrid"`, in the latter case it gives the object the mass is to be transported from.
`...`	further arguments (currently unused).
`y`	an object of class `"otgrid"` the mass is to be transported from.
`p.1`	the first power ≥ 1 of the cost.
`p.2`	the second power ≥ 1 of the cost.

Value

a matrix giving the pairwise costs in column-mayor format.

Examples

```
x <- otgrid(cbind(0:1, 1:0))
y <- otgrid(cbind(1:0, 0:1))

transport_costmat(x, y, p.1 = 1, p.2 = 3)
```

`transport_df.numeric` *Optimal Transport Plan*

Description

Calculate the optimal transport plan.

Usage

```
## S3 method for class 'numeric'
transport_df(x, y, threshold = 1e-15, ...)

transport_df(x, ...)

## S3 method for class 'otgridtransport'
transport_df(x, ...)

## S3 method for class 'otgrid'
transport_df(x, ...)
```

Arguments

`x`	a vector of weights, an object of class `"otgridtransport"` or `"otgrid"`, in the latter case `...` must be the arguments of `pivot_measure`.
`y`	second weight vector.
`threshold`	small value that indicates when a value is considered to be zero.
`...`	further arguments (for `pivot_measure` if `x` is an object of class `"otgrid"`).

Details

In case of two-dimensional grids, the pivot measure is used to calculate the optimal transport plan.

For one-dimensional optimal transport, we assume that the optimal transport plan is the monotone plan. For example, this is the case for costs of the form $c(x, y) = |x - y|^p$ for some $p \geq 1$. In this case, the north-west-corner algorithm is used.

Value

a data frame representing the optimal transport plan. It has columns `from`, `to` and `mass` that specify between which points of the two grids how much mass is transported.

In the two-dimensional case, a point is given by the index in column-mayor format and the data frame is actually stored in the element `df` of an object of class `"otgridtransport"`.

See Also

pivot measure `pivot_measure`

Examples

```
## one-dimensional example
set.seed(1)
wa <- rep(1/5, 5)
wb <- runif(6)
wb <- wb / sum(wb)
transport_df(wa, wb)

## two-dimensional example
x <- otgrid(cbind(0:1, 1:0))
y <- otgrid(cbind(1:0, 0:1))

# first calculate pivot manually
pm <- pivot_measure(x, y)
pm <- transport_df(pm)

# or just
pm2 <- transport_df(x, y)
```

Index

dual1d, 2

image, *8*

north_west_corner, 3

otgrid, 4, *7, 8*

pivot_measure, 5, *8*, *9*, *11, 12*
plot.otgrid, *5*, 7
plot.otgridtransport, *7*
plot.otgridtransport *(*plot.otgrid*)*, 7

transport_cost, *7*
transport_cost
    *(*transport_cost.numeric*)*, 8
transport_cost.numeric, 8
transport_costmat, 10
transport_df, *7*
transport_df *(*transport_df.numeric*)*, 11
transport_df.numeric, 11