Package ‘kantorovich’

April 23, 2024

Type Package
Title Kantorovich Distance Between Probability Measures
Version 3.2.0
Date 2024-04-23
Author Stéphane Laurent
Maintainer Stéphane Laurent <laurent_step@outlook.fr>
Description Computes the Kantorovich distance between two probability measures on a finite set. The Kantorovich distance is also known as the Monge-Kantorovich distance or the first Wasserstein distance.
License GPL-3
URL https://github.com/stla/kantorovich
BugReports https://github.com/stla/kantorovich/issues
Depends R (>= 4.0.0)
Imports CVXR, gmp, lpSolve, methods, rcdd, Rglpk, slam, utils, ompr, ompr.roi, ROI.plugin.glpk
Suggests knitr, rmarkdown, testthat (>= 3.0.0)
VignetteBuilder knitr
Encoding UTF-8
RoxygenNote 7.3.1
SystemRequirements GMP (https://gmplib.org/)
NeedsCompilation no
Repository CRAN
Date/Publication 2024-04-23 07:40:06 UTC

R topics documented:

kantorovich-package .................................................. 2
edistances ............................................................... 2
ejoinings ................................................................. 3

Description

Computes the Kantorovich distance between two probability measures on a finite set, also known as the earth mover’s distance. The Kantorovich distance is not a "unique" distance: it is defined by a given distance on the two finite sets (generally equal). Note that the default distance is the 0-1 distance and with this choice the Kantorovich computation is totally useless (see the vignette). Computing the Kantorovich distance is a linear programming problem, and several methods are provided in the package. In particular there is an exact method available when the probability weights are rational numbers and when the distances are rational numbers as well. A benchmark suggests that the faster methods are those using the 'CVXR' package.

To learn more, start with the vignettes: browseVignettes(package="kantorovich").

If you encounter a bug, or if you have a suggestion to improve the package, please file an issue on the Github repo https://github.com/stla/kantorovich.

Details

Package: kantorovich
Type: Package
Version: 3.1.0
Date: 2023-08-22
License: GPL-3

Author(s)

Stéphane Laurent


ejoinings

Description

Compute the distances at the extreme joinings.

Usage

edistances(mu, nu, dist = NULL, ...)

Arguments

- **mu**: (row margins) probability measure in numeric or bigq/character mode
- **nu**: (column margins) probability measure in numeric or bigq/character mode
- **dist**: function or matrix, the distance to be minimized on average. If NULL, the 0-1 distance is used.
- **...**: arguments passed to dist

Value

A list with two components: the extreme joinings in a list and the distances in a vector.

Note

This function, called by kantorovich, is rather for internal purpose.

---

ejoinings  Extreme joinings

Description

Return extreme joinings between mu and nu.

Usage

ejoinings(mu, nu, zeros = FALSE)

Arguments

- **mu**: (row margins) probability measure in numeric or bigq/character mode
- **nu**: (column margins) probability measure in numeric or bigq/character mode
- **zeros**: logical; in case when mu and nu have different lengths, set FALSE to remove lines or columns full of zeros

Value

A list containing the extreme joinings (matrices).
Examples

```r
mu <- nu <- c(0.5, 0.5)
ejoinings(mu, nu)
# use exact arithmetic
library(gmp)
mu <- nu <- as.bigq(c(0.5, 0.5))
ejoinings(mu, nu)
# different lengths example
mu <- setNames(as.bigq(c(1,2,4), 7), c("a", "b", "c"))
nu <- setNames(as.bigq(c(3,1), 4), c("b", "c"))
ejoinings(mu, nu)
```

---

**kantorovich**  
*Kantorovich distance*

Description

Compute the Kantorovich distance between two probability measures on a finite set.

Usage

```r
kantorovich(mu, nu, dist = NULL, details = FALSE, ...)
```

Arguments

- `mu` (row margins) probability measure in numeric or bigq/character mode
- `nu` (column margins) probability measure in numeric or bigq/character mode
- `dist` function or matrix, the distance to be minimized on average; if NULL, the 0-1 distance is used.
- `details` prints the joinings achieving the Kantorovich distance and returns them in the "joinings" attribute of the output
- `...` arguments passed to `dist` (only if it is a function)

Details

The function firstly computes all the extreme joinings of `mu` and `nu`, then evaluates the average distance for each of them, and then returns the minimal one.

Value

The Kantorovich distance between `mu` and `nu`. 
Examples

```r
mu <- c(1/7, 2/7, 4/7)
nu <- c(1/4, 1/4, 1/2)
kantorovich(mu, nu)
library(gmp)
mu <- as.bigq(c(1,2,4), 7)
nu <- as.bigq(c(1,1,1), c(4,4,2))
kantorovich(mu, nu)
mu <- c("1/7", "2/7", "4/7")
nu <- c("1/4", "1/4", "1/2")
kantorovich(mu, nu, details=TRUE)
```

---

**kantorovich_CVX**

Computes Kantorovich distance with CVX

---

Description

Kantorovich distance using the CVXR package

Usage

```r
kantorovich_CVX(
  mu,
  nu,
  dist,
  solution = FALSE,
  stop_if_fail = TRUE,
  solver = "ECOS",
  ...
)
```

Arguments

- `mu` (row margins) probability measure in numeric mode
- `nu` (column margins) probability measure in numeric mode
- `dist` matrix defining the distance to be minimized on average
- `solution` logical; if TRUE the solution is returned in the "solution" attributes of the output
- `stop_if_fail` logical; if TRUE, an error is returned in the case when no solution is found; if FALSE, the output of `psolve` is returned with a warning
- `solver` the CVX solver, passed to `psolve`
- `...` other arguments passed to `psolve`
Examples

```r
x <- c(1.5, 2, -3)
mu <- c(1/7, 2/7, 4/7)
y <- c(4, 3.5, 0, -2)
nu <- c(1/4, 1/4, 1/4, 1/4)
M <- outer(x, y, FUN = function(x, y) abs(x - y))
kantorovich_CVX(mu, nu, dist = M)
```

kantorovich_glpk  
*Computes Kantorovich distance with GLPK*

Description

Kantorovich distance using the Rglpk package

Usage

```r
kantorovich_glpk(mu, nu, dist, solution = FALSE, stop_if_fail = TRUE, ...)
```

Arguments

- `mu` (row margins) probability measure in numeric mode
- `nu` (column margins) probability measure in numeric mode
- `dist` matrix defining the distance to be minimized on average
- `solution` logical; if TRUE the solution is returned in the "solution" attributes of the output
- `stop_if_fail` logical; if TRUE, an error is returned in the case when no solution is found; if FALSE, the output of `Rglpk_solve_LP` is returned with a warning
- `...` arguments passed to `Rglpk_solve_LP`

Examples

```r
x <- c(1.5, 2, -3)
mu <- c(1/7, 2/7, 4/7)
y <- c(4, 3.5, 0, -2)
nu <- c(1/4, 1/4, 1/4, 1/4)
M <- outer(x, y, FUN = function(x, y) abs(x - y))
kantorovich_glpk(mu, nu, dist = M)
```
**kantorovich_lp**  
*Computes Kantorovich distance with lp_solve*

**Description**

Kantorovich distance using the lpSolve package

**Usage**

```r
kantorovich_lp(mu, nu, dist, solution = FALSE, lp.object = FALSE, ...)
```

**Arguments**

- `mu` (row margins) probability measure in numeric mode
- `nu` (column margins) probability measure in numeric mode
- `dist` matrix defining the distance to be minimized on average
- `solution` logical, to use only if `lp.object=FALSE`; if TRUE the solution is returned in the “solution” attributes of the output
- `lp.object` logical, if FALSE, the output is the Kantorovich distance; if TRUE, the output is a `lp.object`
- `...` arguments passed to `lp`

**Examples**

```r
x <- c(1.5, 2, -3)  
mu <- c(1/7, 2/7, 4/7)  
y <- c(4, 3.5, 0, -2)  
nu <- c(1/4, 1/4, 1/4, 1/4)  
M <- outer(x, y, FUN = function(x, y) abs(x - y))  
kantorovich_lp(mu, nu, dist = M)
```

**kantorovich_ompr**  
*Computes Kantorovich distance with 'ompr'*

**Description**

Kantorovich distance using the ompr package

**Usage**

```r
kantorovich_ompr(mu, nu, dist, solution = FALSE, stop_if_fail = TRUE)
```
Arguments

- **mu**: (row margins) probability measure in numeric mode
- **nu**: (column margins) probability measure in numeric mode
- **dist**: matrix defining the distance to be minimized on average
- **solution**: logical; if TRUE the solution is returned in the "solution" attributes of the output
- **stop_if_fail**: logical; if TRUE, an error is returned in the case when no solution is found; if FALSE, the output of `solve_model` is returned with a warning

Note

The glpk solver is the one used to solve the problem.

Examples

```r
x <- c(1.5, 2, -3)
mu <- c(1/7, 2/7, 4/7)
y <- c(-4, 3.5, 0)
nu <- c(1/4, 1/4, 1/2)
M <- outer(x, y, FUN = function(x, y) abs(x - y))
kantorovich.ompr(mu, nu, dist = M)
```

---

**names.bigq**

### Names for bigq vectors

**Description**

Names for bigq vectors

**Usage**

```r
## S3 method for class 'bigq'
names(x)
```

**Arguments**

- **x**: a bigq vector

**Value**

the names of x
Index

edistances, 2
ejoinings, 3

kantorovich, 3, 4
kantorovich-package, 2
kantorovich_CVX, 5
kantorovich_glpk, 6
kantorovich_lp, 7
kantorovich_ompr, 7

lp, 7
lp.object, 7

names.bigq, 8

psolve, 5

Rglpk_solve_LP, 6

solve_model, 8