Package ‘orloca’

May 16, 2023

Type Package
Depends methods, png, ucminf
Imports grDevices, graphics, knitr, rmarkdown, stats
Suggests orloca.es, testthat (>= 3.0.0)
ByteCompile true
VignetteBuilder knitr
Title Operations Research LOCational Analysis Models
Version 5.3
Date 2023-05-16
Encoding UTF-8
Author Manuel Munoz-Marquez <manuel.munoz@uca.es>
Maintainer Manuel Munoz-Marquez <manuel.munoz@uca.es>
Description Objects and methods to handle and solve the min-sum location problem, also known as Fermat-Weber problem. The min-sum location problem search for a point such that the weighted sum of the distances to the demand points are minimized. See "The Fermat-Weber location problem revisited" by Brimberg, Mathematical Programming, 1, pg. 71-76, 1995, <DOI:10.1007/BF01592245>, General global optimization algorithms are used to solve the problem, along with the ad-hoc Weiszfeld method, see "Sur le point pour lequel la Somme des distances de n points donnes est minimum", by Weiszfeld, Tohoku Mathematical Journal, First Series, 43, pg. 355-386, 1937 or "On the point for which the sum of the distances to n given points is minimum", by E. Weiszfeld and F. Plastria, Annals of Operations Research, 167, pg. 7-41, 2009. <DOI:10.1007/s10479-008-0352-z>.
Language en, es
License GPL (>= 3)
URL http://knuth.uca.es/orloca/
RoxygenNote 7.2.3
Repository CRAN
Repository/R-Forge/Project orloca
Repository/R-Forge/Revision 61
Description

Objects and methods to handle and solve the min-sum location problem, also known as Fermat-Weber problem.

Details

The min-sum location problem search for a point such that the weighted sum of the distances to the demand points are minimized. See "The Fermat-Weber location problem revisited" by Brimberg, Mathematical Programming, 1, pg. 71-76, 1995, doi:10.1007/BF01592245.

General global optimization algorithms are used to solve the problem, along with the adhoc Weiszfeld method, see "Sur le point pour lequel la Somme des distances de n points donnes est minimum", by E. Weiszfeld, Tohoku Mathematical Journal, First Series, 43, pg. 355-386, 1937 or "On the point for which the sum of the distances to n given points is minimum", by E. Weiszfeld and F. Plastria, Annals of Operations Research, 167, pg. 7-41, 2009, doi:10.1007/s104790080352z.
The package provides a class (loca.p) that represents a location problem with a finite set of demand points over the plane. Also, it is possible to plot the points and the objective function. Such objective function is the total weighted distances travelled by all the customers to the service. Non-planar location problems could be handle in future versions of the package.

For a demo, load the package with `library(orloca)`, and use `demo(orloca)`.

The package is ready for internationalization. The author asks for translated versions of the .mo file to include in the package.

**Author(s)**

Manuel Munoz-Marquez <manuel.munoz@uca.es>
Mantainer: Manuel Munoz-Marquez <manuel.munoz@uca.es>

**References**


[4] [http://knuth.uca.es/orloca/](http://knuth.uca.es/orloca/)

**See Also**

Para la versión en español, instale el paquete orloca.es y consulte la ayuda sobre orloca.es-package. (For the Spanish version, install the orloca.es package and see the help about orloca.es-package).

**Examples**

```r
# A new unweighted loca.p object
o <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Compute the sum of distances to point (3, 4)
distsum(o, 3, 4)

# Compute the sum of distances to point (3, 4) using lp norm
```
andalusia

```r
distsum(o, 3, 4, lp=2.5)

# Solve the optimization problem
distsummin(o)
# Contour plot
contour(o)

# Make a demo of the package
demo(orloca)
```

---

<table>
<thead>
<tr>
<th>andalusia</th>
<th>Cities of Andalusia</th>
</tr>
</thead>
</table>

### Description

The 'andalusia' data frame has 12 rows and 4 columns, which are the geographical position of the main capital cities of andalusia.

### Format

- **name**: The name of the city or relative position label.
- **x**: The x coordinate of points.
- **y**: The y coordinate of points.
- **city**: If yes the point is a city in other case is a limit.

### Usage

```r
data('andalusia')
```

### Source

The data are taken from wikipedia.

### See Also

See also `orloca-package`. 
Description

Conversions between loca.p class and some others classes

Arguments

x is the object to convert to the new class object.

row.names Unused.

optional Unused.

... Other arguments, unused.

Details

Methods to convert from and to loca.p class.

NA's values are not allowed in any of the arguments.

The matrix or data.frame to convert into loca.p must have at least two columns. The first column will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the values of weights w.

Value

It returns a new object of the new class.

See Also

See also loca.p

Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Conversion to matrix
m <- as.matrix(loca)

# Show matrix
m

# Conversion from matrix
as.loca.p(m)
as.data.frame.loca.p

## S3 method for class 'loca.p'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

### Description

Conversions between loca.p class and some others classes

### Usage

```r
## S3 method for class 'loca.p'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

### Arguments

- `x`: is the object to convert to the new class object.
- `row.names`: Unused.
- `optional`: Unused.
- `...`: Other arguments, unused.

### Details

Methods to convert from and to `loca.p` class.
NA's values are not allowed in any of the arguments.
The matrix or `data.frame` to convert into `loca.p` must have at least two columns. The first column will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the values of weights w.

### Value

It returns a new object of the new class.

### See Also

See also `loca.p`

### Examples

```r
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Conversion to matrix
m <- as.matrix(loca)

# Show matrix
m

# Conversion from matrix
as.loca.p(m)
```
as.loca.p

as.loca.p The following is for S3 compatibility, mainly for documentation check

Description
Conversions between loca.p class and some others classes

Usage
as.loca.p(x, ...)

Arguments
x is the object to convert to the new class object.
... Other arguments, unused.

Details
Methods to convert from and to loca.p class.
NA's values are not allowed in any of the arguments.
The matrix or data.frame to convert into loca.p must have at least two columns. The first column
will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the
values of weights w.

Value
It returns a new object of the new class.

See Also
See also loca.p

Examples
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Conversion to matrix
m <- as.matrix(loca)

# Show matrix
m

# Conversion from matrix
as.loca.p(m)
as.loca.p.data.frame  

as.loca.p.data.frame S3 method to convert from data.frame to loca.p

Description
Conversions between loca.p class and some others classes

Usage
as.loca.p.data.frame(x, ...)

Arguments
x is the object to convert to the new class object.
... Other arguments, unused.

Details
Methods to convert from and to loca.p class.
NA's values are not allowed in any of the arguments.
The matrix or data.frame to convert into loca.p must have at least two columns. The first column will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the values of weights w.

Value
It returns a new object of the new class.

See Also
See also loca.p

Examples
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Conversion to matrix
m <- as.matrix(loca)

# Show matrix
m

# Conversion from matrix
as.loca.p(m)
as.loca.p.matrix

as.loca.p.matrix S3 method to convert from matrix to loca.p

Description

Conversions between loca.p class and some others classes

Usage

as.loca.p.matrix(x, ...)

Arguments

x  is the object to convert to the new class object.
...
Other arguments, unused.

Details

Methods to convert from and to loca.p class.
NA's values are not allowed in any of the arguments.
The matrix or data.frame to convert into loca.p must have at least two columns. The first column will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the values of weights w.

Value

It returns a new object of the new class.

See Also

See also loca.p

Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Conversion to matrix
m <- as.matrix(loca)

# Show matrix
m

# Conversion from matrix
as.loca.p(m)
Description
Conversions between loca.p class and some others classes

Usage
## S3 method for class 'loca.p'
as.matrix(x, rownames.force = NA, ...)

Arguments
  x is the object to convert to the new class object.
  rownames.force If True the rownames is setted
  ... Other arguments, unused.

Details
Methods to convert from and to loca.p class.
NA's values are not allowed in any of the arguments.
The matrix or data.frame to convert into loca.p must have at least two columns. The first column
will be consider as the x coordinates, the second as the y coordinates, and the third (if given) as the
values of weights w.

Value
It returns a new object of the new class.

See Also
See also loca.p

Examples
  # A new unweighted loca.p object
  loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

  # Conversion to matrix
  m <- as.matrix(loca)

  # Show matrix
  m

  # Conversion from matrix
  as.loca.p(m)
contour.loca.p

Plots of the min-sum objective function

description

contour provides a graphical representations of min-sum function (distsum).

usage

```r
## S3 method for class 'loca.p'
contour(
x,
lp = numeric(0),
xmin = min(min(x@x), xleft),
xmax = max(max(x@x), xright),
ymin = min(min(x@y), ybottom),
ymax = max(max(x@y), ytop),
n = 100,
img = NULL,
xleft = min(x@x),
ybottom = min(x@y),
xright = max(x@x),
ytop = max(x@y),
...)
```

Arguments

- **x**: The loca.p object to compute the objective.
- **lp**: If given, then $l_p$ norm will be used instead of the Euclidean norm.
- **xmin**: The minimum value for x axis.
- **xmax**: The maximum value for x axis.
- **ymin**: The minimum value for y axis.
- **ymax**: The maximum value for y axis.
- **n**: The number of divisions for grid.
- **img**: A raster image to plot on background.
- **xleft**: The left position of the image.
- **ybottom**: The bottom position of the image.
- **xright**: The right position of the image.
- **ytop**: The top position of the image.
- **...**: Other options.
distsum

Details
If \( p < 1 \) then \( l_p \) is not a norm, so only \( p \geq 1 \) are valid values.

Value
contour.loca.p plots a contour plot of min-sum function (distsum).

See Also
See also orloca-package, plot.loca.p and loca.p.

Examples

```r
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# The contour plot of min-sum function for loca (a loca.p object)
contour(loca)
```

Description
The objective function and the gradient function for the min-sum location problem.

Usage

\[
\text{distsum}(o, x = 0, y = 0, l_p = \text{numeric}(0))
\]

Arguments

- \( o \quad \text{An object of loca.p class.} \)
- \( x \quad \text{The x coordinate of the point to be evaluated.} \)
- \( y \quad \text{The y coordinate of the point to be evaluated.} \)
- \( l_p \quad \text{If given, then } l_p \text{ norm will be used instead of the Euclidean norm.} \)

Details
The function zsum is deprecated and will be removed from new versions of the package.

Value
\[ \text{distsum} \text{ returns the objective function of the min-sum location problem, } \sum_{a_i \in o} w_i d(a_i, (x, y)), \]
where \( d(a_i, (x, y)) \) gives the euclidean or the \( l_p \) distances between \( a_i \) and the point \((x, y)\).
distsumgra

See Also

See also orloca-package and distsummin.

Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Evaluation of distsum at (0, 0)
distsum(loca)

# Evaluation of distsum at (1, 3)
distsum(loca, 1, 3)
# Compute the objective function at point (3, 4) using lp norm and p = 2.5
distsum(loca, 3, 4, lp=2.5)
# The gradient function at (1,3)
distsumgra(loca, 1, 3)

distsumgra  Computes the gradient of distsum function

Description

The gradient function for the min-sum location problem.

Usage

distsumgra(o, x = 0, y = 0, lp = numeric(0), partial = F)

Arguments

o  An object of loca.p class.
x  The x coordinate of the point to be evaluated.
y  The y coordinate of the point to be evaluated.
lp  If given, then \( l_p \) norm will be used instead of the Euclidean norm.
partial  If (x,y) is a demand point partial=T means ignore such point to compute the gradient. This option is mainly for internal use.

Details

The function zsumgra is deprecated and will be removed from new versions of the package.

Value

distsumgra returns the gradient vector of the function of the min-sum location problem, \( \sum_{a_i \in o} w_i d(a_i, (x, y)) \), where \( d(a_i, (x, y)) \) gives the euclidean or the \( l_p \) distances between \( a_i \) and the point \( (x, y) \).
distsummin

See Also

See also orloca-package and distsum.

Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Evaluation of distsum at (0, 0)
distsum(loca)

# Evaluation of distsum at (1, 3)
distsum(loca, 1, 3)
# Compute the objective function at point (3, 4) using lp norm and p = 2.5
distsum(loca, 3, 4, lp=2.5)
# The gradient function at (1,3)
distsumgra(loca, 1, 3)

distsummin

Returns the solution of the minimization problem

Description

Solve the min-sum location problem for a given loca.p class object.

Usage

distsummin(
  o,
  x = 0,
  y = 0,
  lp = numeric(0),
  max.iter = 1e+05,
  eps = 0.001,
  verbose = FALSE,
  algorithm = "Weiszfeld",
  ...
)

Arguments

  o            An object of loca.p class.
  x            The x coordinate of the starting point. It’s default value is 0.
  y            The y coordinate of the starting point. It’s default value is 0.
  lp           If given, the $l_p$ norm will be used instead of the Euclidean norm.
  max.iter     Maximum number of iterations allowed. It’s default value is 100000.
**distsummin**

- **eps**: The module of the gradient in the stop rule. Its default value is 1e-3.
- **verbose**: If TRUE the function produces detailed output. Its default value is FALSE.
- **algorithm**: The method to be use. For this version of the package, the valid values are: "gradient" for a gradient based method, "search" for local search method (this option is deprecated), "ucminf" for optimization with ucminf from ucminf package, and "Weiszfeld" for the Weiszfeld method or any of the valid method for optim function, now "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN". "Weiszfeld" is the default value.

... Other options for optimization algorithms.

**Details**

The algorithms Weiszfeld and gradient include and optimality test for demand points. The Weiszfeld version of the algorithm also implements slow convergence test and accelerator procedure.

If \( p < 1 \) thus \( l_p \) is not a norm, so, only \( p \geq 1 \) are valid values.

Since \( l_2 \) norm is the Euclidean norm, when \( p = 2 \) distsumlpmin are equal to distsummin. But the computations involved are greater for the first form.

max.iter for SANN algorithm is the number of evaluation of objective function, so this method usually requires large values of max.iter to reach optimal value.

The function zsummin is deprecated and will be removed from new versions of the package.

**Value**

distsummin returns an array with the coordinates of the solution point.

**See Also**

See also orloca-package, loca.p and distsum.

**Examples**

```r
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Compute the minimum
sol <- distsummin(loca)

# Show the result
sol

# Evaluation of the objective function at solution point
distsum(loca, sol[1], sol[2])
```
Description

An object of class `loca.p` represents a weighted location problem with a finite demand points set. The `orloca-package` is mainly devoted to deals with location problems.

Arguments

- `x` is a vector of the x coordinates of the demand points.
- `y` is a vector of the y coordinates of the demand points.
- `w` is a vector of weights of the demand points. If `w` is omitted then all weights are considered as 1.
- `label` If given, it is the label of the new object.

Details

The main generator of the `loca.p` class is `loca.p(x, y, w = numeric(0), label = "")`. An alternative form is `new("loca.p", x, y, w = numeric(0), label = "")`.

The lengths of `x` and `y` vector must be equals. The length of `w` must be equal to the previous ones or must be 0. NA’s values are not allowed at any of the arguments.

Value

If the arguments have valid values, it returns a new object of class `loca.p`, else it returns an error.

`summary(x)` returns a summary of the `x` `loca.p` object and `print(x)` prints the `x` `loca.p` object in table format.

See Also

See also `orloca-package`.

Examples

```r
# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# or
loca <- new("loca.p", x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# An example with weights and name
locb <- new("loca.p", x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1),
             w = c(1, 2, 1, 2), label = "Weighted case")
```
Description

persp provides a graphical representations of min-sum function (distsum).

Usage

```r
## S3 method for class 'loca.p'
persp(
x,
    lp = numeric(0),
    xmin = min(x@x),
    xmax = max(x@x),
    ymin = min(x@y),
    ymax = max(x@y),
    n = 10,
    ticktype = "detailed",
    ...
)
```

Arguments

- `x` The loca.p object to compute the objective.
- `lp` If given, then $l_p$ norm will be used instead of the Euclidean norm.
- `xmin` The minimum value for x axis.
- `xmax` The maximum value for x axis.
- `ymin` The minimum value for y axis.
- `ymax` The maximum value for y axis.
- `n` The number of divisions for grid.
- `ticktype` parameter to pass to low level function persp
- `...` Other options.

Details

If $p < 1$ then $l_p$ is not a norm, so only $p >= 1$ are valid values.

Value

A plot a 3D plot or min-sum function.

See Also

See also orloca-package, plot.loca.p and loca.p.
Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# The 3D graphics
persp(loca)

Description

This method provides a graphical representation of an object of class loca.p.

Usage

```r
## S3 method for class 'loca.p'
plot(
x,
xlab = "", ylab = "",
main = paste(gettext("Plot of loca.p", domain = "R-orloca"), ifelse(x@label == "", "", paste0(": \"", x@label, "\""))),
img = NULL,
xlim = c(min(xleft, min(x@x)), max(xright, max(x@x))), ylim = c(min(ybottom, min(x@y)), max(ytop, max(x@y))),
xleft = min(x@x), ybottom = min(x@y), xright = max(x@x), ytop = max(x@y), ...
)
```

Arguments

- `x` The loca.p object to plot.
- `xlab` The label for x axis.
- `ylab` The label for y axis.
- `main` The main title for the plot.
- `img` A raster image to plot on background.
- `xlim` Limit over the x axes of the plot.
- `ylim` Limit over the y axes of the plot.
- `xleft` The left position of the image.
- `ybottom` The bottom position of the image.
rloca.p

xright  The right position of the image.
ytop    The top position of the image.
...     Other graphical options.

Details

The function plots the demand points with automatic limits evaluation.

Value

The function plots the required graphics.

See Also

See also orloca-package, loca.p and plot.

Examples

# A new unweighted loca.p object
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# The plot of loca object
plot(loca)

rloca.p  Random instances generator of loca.p class object

Description

rloca.p function returns a random instance of loca.p class object at a given rectangular region.

Usage

rloca.p(
    n,
    xmin = 0,
    xmax = 1,
    ymin = 0,
    ymax = 1,
    wmin = 1,
    wmax = 1,
    label = "",
    groups = 0,
    xgmin = xmin,
    xgmax = xmax,
    ygmin = ymin,
    ygmax = ymax
)
Arguments

- **n**
  - The number of demand points.
- **xmin**
  - Minimum value for the x coordinates of the demand points.
- **xmax**
  - Maximum value for the x coordinates of the demand points.
- **ymin**
  - Minimum value for the y coordinates of the demand points.
- **ymax**
  - Maximum value for the y coordinates of the demand points.
- **wmin**
  - Minimum value for weights
- **wmax**
  - Maximum value for weights
- **label**
  - The label for the new loca.p object.
- **groups**
  - The number of (almost) equal size groups to generate, or a list size of the groups to generate. In the second case n will be ignored.
- **xgmin**
  - Minimum value for the x coordinate of demand points with respect to the group reference point.
- **xgmax**
  - Maximum value for the x coordinate of demand points with respect to the group reference point.
- **ygmin**
  - Minimum value for the y coordinate of demand points with respect to the group reference point.
- **ygmax**
  - Maximum value for the y coordinate of demand points with respect to the group reference point.

Details

- **n** must be at least 1.
- **xmin** must be less or equal than **xmax**.
- **ymin** must be less or equal than **ymax**.

If a non zero value is given for **groups** parameter, then a reference point for each group are generated. At second stage, the offset part for each demand point are generated, and added to the reference point generated at the first stage.

Note that **groups = 1** is not equivalent to the default value **groups = 0**, because in the first case a reference point are generated at the first stage.

Value

If the arguments are valid values, it returns a new object of loca.p class, else it returns an error.

See Also

See also orloca-package and loca.p.
Examples

# A random loca.p object at unit square with 5 demand points
rloca.p(5)
# At another region
rloca.p(10, xmin=-2, xmax=2, ymin=-2, ymax=2)
# Five groups
rloca.p(48, groups=5)
# Three unequal size groups
rloca.p(1, groups=c(10, 7, 2))

Description

The function zsum is deprecated and could be removed in next version of the package. Use distsum instead.

Usage

zsum(...)

Arguments

... Parameters passed to distsum

Description

The function zsumgra is deprecated and could be removed in next version of the package. Use distsumgra instead.

Usage

zsumgra(...)

Arguments

... Parameters passed to distsumgra
Description
The function zsummin is deprecated and could be removed in next version of the package. Use distsummin instead.

Usage
zsummin(...)  

Arguments
... Parameters passed to distsummin
Index

* andalusia
  andalusia, 4
* classes
  as-methods, 5
  contour.loca.p, 11
  distsum, 12
  distsumgra, 13
  distsummin, 14
  loca.p, 16
  persp.loca.p, 17
  plot, 18
* datagen
  rloca.p, 19
* data
  andalusia, 4
* deprecated
  zsum, 21
  zsumgra, 21
  zsummin, 22
* hplot
  contour.loca.p, 11
  persp.loca.p, 17
  plot, 18
* methods
  as-methods, 5
* optimize
  distsum, 12
  distsumgra, 13
  distsummin, 14
  loca.p, 16
  orloca-package, 2
* package
  orloca-package, 2

andalusia, 4
as-methods, 5
as.data.frame (as-methods), 5
as.data.frame.loca.p, 6
as.loca.p, 7
as.loca.p, data.frame-method
   (as-methods), 5
as.loca.p, matrix-method (as-methods), 5
as.loca.p.data.frame, 8
as.loca.p.matrix, 9
as.matrix (as-methods), 5
as.matrix.loca.p, 10
contour.loca.p, 11
distsum, 12, 14, 15, 21
distsum, loca.p-method (distsum), 12
distsumgra, 13, 21
distsumgra, loca.p-method (distsumgra), 13
distsummin, 13, 14, 22
distsummin, loca.p-method (distsummin), 14
loca.p, 5–10, 12, 15, 16, 17, 19
orloca-package, 2
persp.loca.p, 17
plot, 18, 19
plot.loca.p, 12, 17
print (loca.p), 16
rloca.p, 19
summary (loca.p), 16
zsum, 21
zsumgra, 21
zsummin, 22