

# Package ‘orloca’

January 2, 2012

**Type** Package

**Depends** methods

**Suggests** grDevices, graphics, snow

**Title** The package deals with Operations Research LOCational Analysis models

**Version** 3.2

**Date** 2010-05-22

**Author** Fernando Fernandez-Palacin <fernando.fernandez@uca.es> and  
Manuel Munoz-Marquez <manuel.munoz@uca.es>

**Maintainer** Manuel Munoz-Marquez <manuel.munoz@uca.es>

**Description** This version of the package deals with the min-sum ocation  
problem, also known as Fermat--Weber problem or center location  
problem. The min-sum location problem search for a point such  
that the weighted sum of the distances to the demand points are minimized.

**License** GPL (>= 2)

**URL** <http://knuth.uca.es/orloca>

**Repository** CRAN

**Date/Publication** 2010-05-26 16:55:10

## R topics documented:

orloca-package	2
as-methods	3
czsum	5
czsummin	5
loca.p-class	6
plot-methods	7
plot.zsum	8

rloca.p . . . . .	9
zsum . . . . .	10
zsuml2 . . . . .	11
zsuml2min . . . . .	12
zsumlp . . . . .	13
zsumlpmin . . . . .	14
zsummin . . . . .	15

<b>Index</b>	<b>16</b>
--------------	-----------

---

orloca-package	<i>The package deals with Operations Research LOCational Analysis models</i>
----------------	--

---

## Description

This version of the package deals with the min-sum location problem, also known as Fermat–Weber problem or center location problems. The min-sum location problem look for a point such that the weighted sum of the distances to the demand points are minimized.

## Details

Package: orloca  
 Type: Package  
 Version: 3.2  
 Date: 2010-05-22  
 License: GPL (>= 2)

The packages 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 travel by all the customers to the service.

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

The package is ready for internationalization. The authors ask for translated version of the `.mo` file to include in the package.

## Index:

`loca.p`: loca.p class description.  
`rloca.p`: random instances of loca.p class objects.  
`zsum`: function to compute objective function for min-sum location models.  
`zsummin`: function to look for the minimum for the optimization problem.  
`plot.loca.p`: to make plots of loca.p objects.  
`plot.zsum`: to make plots of objective function.

**Author(s)**

Fernando Fernandez-Palacin <fernando.fernandez@uca.es> and Manuel Munoz-Marquez <manuel.munoz@uca.es>

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

**References**

[1] Love, R. F., Morris, J. G., Wesolowsky, G. O. *Facilities Location: Chapter 2: Introduction to Single-Facility Location*, 1988, North-Holland

[2] <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 package `orloca.es` and see the help about [orloca.es-package](#)).

**Examples**

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

# Compute the objective function at point (3, 4)
zsum(o, 3, 4)

# Compute the objective function at point (3, 4) using lp norm
zsum(o, 3, 4, lp=2.5)

# Solve the optimization problem
zsummin(o)

# Contour plot
contour.loca.p(o)

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

**Description**

Methods to convert from and to `loca.p` class.

**Usage**

```

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

```

**Arguments**

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

**Details**

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

The `matrix` 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 w.

The `data.frame` to convert into `loca.p` must have at least an x column for x coordinates, and an y column for y coordinates. Optionally, it can have w column, as the values of w.

**Value**

If the arguments have valid values, 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)
m

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

```

---

czsum	<i>czsum and czsumgra at orloca package</i>
-------	---

---

**Description**

zsum and zsumgra functions computed at the given snow cluster.

**Usage**

```
czsum(o, cluster, x=0, y=0)
czsumgra(o, cluster, x=0, y=0)
```

**Arguments**

o	An object of loca.p class.
cluster	An snow cluster object.
x	The x coordinate of the point to be evaluated.
y	The y coordinate of the point to be evaluated.

**Value**

czsum 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 distances between  $a_i$  and the point  $(x, y)$ .

czsumgra returns the gradient vector of the function zsum.

The computations are made at the given snow cluster.

**See Also**

See <http://www.stat.uiowa.edu/~luke/R/cluster/cluster.html> for information about computers clusters.

See also [zsum](#), [zsumgra](#) and [zsummin](#).

---

czsummin	<i>czsummin at orloca package</i>
----------	-----------------------------------

---

**Description**

czsummin is the cluster version of the function zsummin.

**Usage**

```
czsummin(o, cluster, x=0, y=0, max.iter=100, eps=1.e-3,
  verbose=FALSE, algorithm="gradient")
```

**Arguments**

<code>o</code>	An object of <code>loca.p</code> class.
<code>cluster</code>	An snow cluster object.
<code>x</code>	The x coordinate of the starting point.
<code>y</code>	The y coordinate of the starting point.
<code>max.iter</code>	Maximum number of iterations allowed.
<code>eps</code>	The module of the gradient in the stop rule.
<code>verbose</code>	If TRUE then the function produces detailed output.
<code>algorithm</code>	The algorithm to be use. For this version of the package, the valid values are: "gradient" or "g" for a gradient based algorithm, and "search" or "s" for local search method. "gradient" is the default value.

**Value**

`czsummin` returns an array with the coordinates of the solution point.

These computation are made at the given snow cluster.

**See Also**

See <http://www.stat.uiowa.edu/~luke/R/cluster/cluster.html> for information about computers clusters.

See also [zsummin](#), [loca.p](#) and [zsum](#).

---

`loca.p-class`

*loca.p class for Operations Research LOCational Analysis*

---

**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.

**Details**

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 a summary of the `x` `loca.p` object.

## Generators

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

**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.

## See Also

See also [orloca-package](#).

## Examples

```
# 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")
```

---

plot-methods

*plot of loca.p class objects*

---

## Description

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

## Usage

```
## S3 method for class 'loca.p'
plot(x, xlab="", ylab="", main=gettext("Plot of loca.p object"), ...)
```

## Arguments

<code>x</code>	The <code>loca.p</code> object to plot.
<code>xlab</code>	The label for x axis.
<code>ylab</code>	The label for y axis.
<code>main</code>	The main title for the plot.
<code>...</code>	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.zsum](#).

**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)
```

---

plot.zsum

*Plots of the min-sum objective function*

---

**Description**

contour and persp provide two graphical representations of min-sum function (zsum).

**Usage**

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

**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.
...	Other options.

**Details**

If  $p < 1$  then  $l_p$  are not a norm, so only  $p \geq 1$  are valid values.

**Value**

contour.loca.p plots a contour like graphics and persp.loca.p a 3D plot.

**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 contour plot of min-sum function for loca (a loca.p object)
contour(loca)

# The 3D graphics
persp(loca)
```

---

rloca.p

*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, groups=numeric(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.
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 groups parameter is given, 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 = numeric(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 groups
rloca.p(1, groups=c(10, 7, 2))
```

---

zsum

*zsum and zsumgra at orloca package*

---

**Description**

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

**Usage**

```
zsum(o, x=0, y=0, lp=numeric(0))
zsumgra(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.

**Value**

zsum 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  $(x, y)$ .

zsumgra returns the gradient vector of the function zsum.

**See Also**

See also [orloca-package](#) and [zsummin](#).

**Examples**

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

# Evaluation of zsum at (0, 0)
zsum(loca)

# Evaluation of zsum at (1, 3)
zsum(loca, 1, 3)

# Compute the objective function at point (3, 4) using lp norm
zsum(loca, 3, 4, lp=2.5)

# The gradient function at (1,3)
zsumgra(loca, 1, 3)
```

---

zsuml2

*zsuml2 and zsuml2gra at orloca package*


---

**Description**

zsum and zsumgra functions for the Euclidean norm ( $l_2$ ). Mainly for internal use.

**Usage**

```
zsuml2(o, x=0, y=0)
zsuml2gra(o, x=0, y=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.
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.

**Value**

zsuml2 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 distances between  $a_i$  and the point  $(x, y)$ .

zsumgra returns the gradient vector of the function zsum.

**See Also**

See also [orloca-package](#), [zsum](#), [zsumgra](#) and [zsummin](#).

---

zsuml2min

*zsuml2min at orloca package*


---

**Description**

zsummin function for the Euclidean norm ( $l_2$ ). Mainly for internal use.

**Usage**

```
zsuml2min(o, x=0, y=0, max.iter=100, eps=1.e-3, verbose=FALSE,
          algorithm="weiszfeld")
```

**Arguments**

o	An object of loca.p class.
x	The x coordinate of the starting point.
y	The y coordinate of the starting point.
max.iter	Maximum number of iterations allowed.
eps	The module of the gradient in the stop rule.
verbose	If TRUE the function produces detailed output.
algorithm	The method to be use. For this version of the package, the valid values are: "gradient" or "g" for a gradient based method, "search" or "s" for local search method, and "weiszfeld" or "w" 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.

**Value**

zsuml2min returns an array with the coordinates of the solution point.

**See Also**

See also [orloca-package](#), [zsummin](#), [loca.p](#) and [zsum](#).

---

zsumlp

*zsumlp and zsumlpgra at orloca package*


---

**Description**

zsum and zsumgra functions with  $l_p$  norm. Mainly for internal use.

**Usage**

```
zsumlp(o, x=0, y=0, p=2)
zsumlpgra(o, x=0, y=0, p=2, 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.
p	The $l_p$ norm to use.
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**

If  $p < 1$  then  $l_p$  are not a norm, so only  $p \geq 1$  are valid values.

**Value**

zsumlp returns the objective function of the min-sum location problem with  $l_p$  norm,  $\sum_{a_i \in o} w_i d(a_i, (x, y))$ , where  $d(a_i, (x, y))$  gives the distances between  $a_i$  and the point  $(x, y)$  using  $l_p$  norm.

zsumlpgra returns the gradient vector of the function zsumlp.

**Note**

Since  $l_2$  norm is the Euclidean norm, when  $p = 2$  zsumlp are equal to zsum, and zsumlpgra are equal to zsumgra. But the computations involved are greater for the firsts form.

**See Also**

See also [zsum](#), [orloca-package](#) and [zsumlpmin](#).

zsumlpmin

*zsumlpmin at orloca package***Description**

zsummin function with  $l_p$  norm. Mainly for internal use.

**Usage**

```
zsumlpmin(o, x=0, y=0, p=2, max.iter=100, eps=1.e-3,
          verbose=FALSE, algorithm="weiszfeld")
```

**Arguments**

o	An object of loca.p class.
x	The x coordinate of the starting point.
y	The y coordinate of the starting point.
p	p value for $l_p$ norm.
max.iter	Maximum number of iterations allowed.
eps	The module of the gradient in the stop rule.
verbose	If TRUE, then the function produces detailed output.
algorithm	The method to be use. For this version of the package, the valid values are: "gradient" or "g" for a gradient based method, "search" or "s" for local search method, and "weiszfeld" or "w" 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.

**Details**

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

**Value**

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

**Note**

Since  $l_2$  norm is the Euclidean norm, when  $p = 2$  zsumlpmin are equal to zsummin. But the computations involved are greater for the first form.

**See Also**

See also [zsummin](#), [orloca-package](#), [loca.p](#) and [zsum](#).

---

zsummin *zsummin at orloca package*

---

### Description

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

### Usage

```
zsummin(o, x=0, y=0, lp=numeric(0), max.iter=100, eps=1.e-3,
        verbose=FALSE, algorithm="weiszfeld")
```

### Arguments

<code>o</code>	An object of <code>loca.p</code> class.
<code>x</code>	The x coordinate of the starting point.
<code>y</code>	The y coordinate of the starting point.
<code>lp</code>	If given, the $l_p$ norm will be used instead of the Euclidean norm.
<code>max.iter</code>	Maximum number of iterations allowed.
<code>eps</code>	The module of the gradient in the stop rule.
<code>verbose</code>	If TRUE the function produces detailed output.
<code>algorithm</code>	The algorithm to be use. For this version of the package, the valid values are: "gradient" or "g" for a gradient based method, "search" or "s" for local search method, and "weiszfeld" or "w" 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.

### Value

`zsummin` returns an array with the coordinates of the solution point.

### See Also

See also [orloca-package](#), [loca.p](#) and [zsum](#).

### Examples

```
# 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<-zsummin(loca)

# Show the result
sol

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

# Index

- \*Topic **classes**
  - as-methods, 3
  - czsum, 5
  - czsummin, 5
  - loca.p-class, 6
  - plot-methods, 7
  - plot.zsum, 8
  - zsum, 10
  - zsuml2, 11
  - zsuml2min, 12
  - zsumlp, 13
  - zsumlpmin, 14
  - zsummin, 15
- \*Topic **datagen**
  - rloca.p, 9
- \*Topic **hplot**
  - plot-methods, 7
  - plot.zsum, 8
- \*Topic **methods**
  - as-methods, 3
- \*Topic **optimize**
  - czsum, 5
  - czsummin, 5
  - loca.p-class, 6
  - orloca-package, 2
  - zsum, 10
  - zsuml2, 11
  - zsuml2min, 12
  - zsumlp, 13
  - zsumlpmin, 14
  - zsummin, 15
- \*Topic **package**
  - orloca-package, 2
- as, loca.p-method (as-methods), 3
- as-methods, 3
- as.data.frame (as-methods), 3
- as.data.frame, loca.p-method (as-methods), 3
- as.data.frame.loca.p (as-methods), 3
- as.loca.p (as-methods), 3
- as.loca.p, data.frame-method (as-methods), 3
- as.loca.p, matrix-method (as-methods), 3
- as.loca.p.data.frame (as-methods), 3
- as.loca.p.matrix (as-methods), 3
- as.matrix (as-methods), 3
- as.matrix, loca.p-method (as-methods), 3
- as.matrix.loca.p (as-methods), 3
- contour, loca.p-method (plot.zsum), 8
- contour.loca.p (plot.zsum), 8
- czsum, 5
- czsum, loca.p-method (czsum), 5
- czsumgra (czsum), 5
- czsumgra, loca.p-method (czsum), 5
- czsummin, 5
- czsummin, loca.p-method (czsummin), 5
- initialize, loca.p-method (loca.p-class), 6
- loca.p, 2, 4, 6, 8, 9, 13–15
- loca.p (loca.p-class), 6
- loca.p-class, 6
- orloca (orloca-package), 2
- orloca-package, 6–15
- orloca-package, 2
- orloca.es-package, 3
- persp, loca.p-method (plot.zsum), 8
- persp.loca.p (plot.zsum), 8
- plot, loca.p-method (plot-methods), 7
- plot-methods, 7
- plot.loca.p, 2, 9
- plot.loca.p (plot-methods), 7
- plot.zsum, 2, 8, 8
- print, loca.p-method (loca.p-class), 6
- print.loca.p (loca.p-class), 6

rloca.p, 2, 9

summary, loca.p-method (loca.p-class), 6  
summary-method (loca.p-class), 6

zsum, 2, 5, 6, 10, 12–15

zsum, loca.p-method (zsum), 10

zsumgra, 5, 12

zsumgra (zsum), 10

zsumgra, loca.p-method (zsum), 10

zsuml2, 11

zsuml2, loca.p-method (zsuml2), 11

zsuml2gra (zsuml2), 11

zsuml2gra, loca.p-method (zsuml2), 11

zsuml2min, 12

zsuml2min, loca.p-method (zsuml2min), 12

zsumlp, 13

zsumlp, loca.p-method (zsumlp), 13

zsumlpgra (zsumlp), 13

zsumlpgra, loca.p-method (zsumlp), 13

zsumlpmin, 13, 14

zsumlpmin, loca.p-method (zsumlpmin), 14

zsummin, 2, 5, 6, 11–14, 15

zsummin, loca.p-method (zsummin), 15