

# Package ‘orloca.es’

April 1, 2025

**Type** Package

**Depends** orloca (>= 5.3)

**Language** es

**Title** Spanish version of orloca package. Modelos de localizacion en investigacion operativa

**Version** 5.5

**Date** 2025-04-01

**Encoding** UTF-8

**Description** Help and demo in Spanish of the orloca package. Ayuda y demo en espanol del paquete orloca. Objetos y metodos para manejar y resolver el problema de localizacion de suma minima, tambien conocido como problema de Fermat-Weber. El problema de localizacion de suma minima busca un punto tal que la suma ponderada de las distancias a los puntos de demanda se minimice. Vease ``The Fermat-Weber location problem revisited" por Brimberg, Mathematical Programming, 1, pag. 71-76, 1995. <[DOI:10.1007/BF01592245](https://doi.org/10.1007/BF01592245)>. Se usan algoritmos generales de optimizacion global para resolver el problema, junto con el metodo especifico Weiszfeld, vease ``Sur le point pour lequel la Somme des distance de n points donnees est minimum", por Weiszfeld, Tohoku Mathematical Journal, First Series, 43, pag. 355-386, 1937 o ``On the point for which the sum of the distances to n given points is minimum", por E. Weiszfeld y F. Plastria, Annals of Operations Research, 167, pg. 7-41, 2009. <[DOI:10.1007/s10479-008-0352-z](https://doi.org/10.1007/s10479-008-0352-z)>.

**License** GPL (>= 3)

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

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** Manuel Munoz-Marquez [aut, cre]

**Maintainer** Manuel Munoz-Marquez <[manuel.munoz@uca.es](mailto:manuel.munoz@uca.es)>

**Repository** CRAN

**Date/Publication** 2025-04-01 11:30:02 UTC

## Contents

orloca.es-package . . . . .	2
andalusia . . . . .	4
as-methods . . . . .	4
as.data.frame.loca.p . . . . .	5
as.loca.p . . . . .	6
as.loca.p.data.frame . . . . .	7
as.loca.p.matrix . . . . .	8
as.matrix.loca.p . . . . .	9
contour.loca.p . . . . .	10
distsum . . . . .	11
distsumgra . . . . .	12
distsuml2min . . . . .	13
distsumlp . . . . .	14
distsumlpmin . . . . .	15
distsummin . . . . .	16
loca.p-class . . . . .	17
persp.distsum . . . . .	18
plot . . . . .	19
rloca.p . . . . .	20
zsum . . . . .	22
zsumgra . . . . .	22
zsummin . . . . .	22

<b>Index</b>	<b>23</b>
--------------	-----------

---

orloca.es-package	<i>Spanish version of orloca package - Versión española del paquete orloca</i>
-------------------	--

---

## Description

Ayuda y demo en español del paquete orloca.

Objetos y métodos para manejar y resolver el problema de localización de suma ponderada mínima, también conocido como problema de Fermat-Weber.

## Detalles

El problema de localización de suma mínima busca un punto tal que la suma ponderada de las distancias a los puntos de demanda se minimice. Véase "The Fermat-Weber location problem revisited" por Brimberg, *Mathematical Programming*, 1, pag. 71-76, 1995. <DOI: 10.1007/BF01592245>.

Se usan algoritmos generales de optimización global para resolver el problema, junto con el método adhoc Weiszfeld, véase "Sur le point pour lequel la Somme des distance de n points donne est minimum", por Weiszfeld, *Tohoku Mathematical Journal, First Series*, 43, pag. 355-386, 1937 o "On the point for which the sum of the distances to n given points is minimum", por E. Weiszfeld y F. Plastria, *Annals of Operations Research*, 167, pg. 7-41, 2009. <DOI:10.1007/s10479-008-0352-z>.

Package: orloca.es  
Type: Package  
Version: 5.5  
Date: 2025-04-01  
License: GPL (>= 3)

El paquete proporciona una clase, `loca.p`, que representa un problema de localización con un conjunto finito de puntos de demanda sobre el plano. También es posible representar los puntos y la función objetivo. Dicha función objetivo representa la suma de los desplazamientos de los usuarios al servicio.

El problema de localización no plano será abordado en futuras versiones del paquete.

Para una demostración, cargue el paquete con `library(orloca.es)` y use `demo(orloca)`.

El paquete está preparado para su internacionalización. Las traducciones de los ficheros `.mo` recibidas serán añadidas en próximas versiones del paquete.

### Autor

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

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

### Referencias

[1] Brimberg, J. *The Fermat-Weber location problem revisited*, *Mathematical Programming*, 1, pg. 71-76, 1995. doi:10.1007/BF01592245.

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

[3] <http://knuth.uca.es/orloca/>

### Véase también

For the English version of the package see [orloca-package](#).

### Ejemplos

```
# Un objeto loca.p no ponderado
o <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Calcula la funcion objetivo en el punto (3, 4)
zsum(o, 3, 4)
# Calcula ls suma de las distancias al punto (3, 4) usando la norma lp
zsum(o, 3, 4, lp=2.5)
# Resuelve el problema de localizacion
zsummin(o)
# Curvas de nivel
contour(o)
```

```
# Ejecuta una demo del paquete
demo(orloca)
```

---

andalusia

*Ciudades de Andalucia*

---

### Description

El conjunto de datos 'andalusia' tiene 12 filas y 4 columnas, que son la posición geográfica de las capitales de provincia andaluzas.

### Format

name: El nombre de una ciudad o de una etiqueta relativa de posición  
x: La coordenada x de los puntos  
y: La coordenada y de los puntos  
city: Si yes entonces el punto es una ciudad, en otro caso es un límite.

### Uso

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

### Fuente

Los datos se han tomado de wikipedia.

### See Also

Véase también [orloca.es-package](#).

---

as-methods

*as-methods*

---

### Description

Conversiones entre la clase loca.p y algunas otras

### Argumentos

**x:** es el objeto a convertir a la nueva clase.  
**row.names:** Sin uso.  
**optional:** Sin uso.  
**...:** Otros argumentos, sin uso.

**Valor**

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

**Detalles**

Métodos para convertir desde y a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada x, y la segunda como la coordenada y, y la tercera (si se ha suministrado) serán los valores de w.

El `data.frame` a convertir a `loca.p` debe tener al menos una columna x para la coordenada x, y una columna y para la coordenada y. Opcionalmente, puede tener una columna w para los valores de w.

**Ejemplos**

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
```

```
# Conversion a matrix m <- as.matrix(loca)
```

```
# Muestra la matrix m
```

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

**Véase también**

Véase también [loca.p](#)

---

as.data.frame.loca.p    *as.data.frame.loca.p* Método S3 para convertir de *loca.p* a *data.frame*

---

**Description**

Conversiones entre la clase `loca.p` y algunas otras

**Argumentos**

**x:** es el objeto a convertir a la nueva clase.

**row.names:** Sin uso.

**optional:** Sin uso.

**...:** Otros argumentos, sin uso.

**Valor**

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

## Detalles

Métodos para convertir desde y a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` o el `data.frame` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada x, y la segunda como la coordenada y, y la tercera, si se ha suministrado, serán los valores de los pesos w.

## Ejemplos

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Conversion a matrix m <- as.matrix(loca)
# Muestra la matrix m
# Conversion desde matrix as.loca.p(m)
```

## Véase también

Véase también [loca.p](#)

---

<code>as.loca.p</code>	<i>as.loca.p Método para compatibilidad con clases S3, principalmente para el test de documentación</i>
------------------------	---

---

## Description

Conversiones entre la clase `loca.p` y algunas otras

## Argumentos

**x:** es el objeto a convertir a la nueva clase.

**row.names:** Sin uso.

**optional:** Sin uso.

**...:** Otros argumentos, sin uso.

## Valor

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

## Detalles

Métodos para convertir desde y a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada x, y la segunda como la coordenada y, y la tercera (si se ha suministrado) serán los valores de w.

El `data.frame` a convertir a `loca.p` debe tener al menos una columna x para la coordenada x, y una columna y para la coordenada y. Opcionalmente, puede tener una columna w para los valores de w.

## Ejemplos

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Conversion a matrix m <- as.matrix(loca)
# Muestra la matrix m
# Conversion desde matrix as.loca.p(m)
```

## Véase también

Véase también [loca.p](#)

---

as.loca.p.data.frame    *as.loca.p.data.frame Método S3 para convertir de data.frame a loca.p*

---

## Description

Conversiones entre la clase `loca.p` y algunas otras

## Argumentos

**x:** es el objeto a convertir a la nueva clase.

**row.names:** Sin uso.

**optional:** Sin uso.

**...:** Otros argumentos, sin uso.

## Valor

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

## Detalles

Métodos para convertir desde y a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada x, y la segunda como la coordenada y, y la tercera (si se ha suministrado) serán los valores de w.

El `data.frame` a convertir a `loca.p` debe tener al menos una columna x para la coordenada x, y una columna y para la coordenada y. Opcionalmente, puede tener una columna w para los valores de w.

## Ejemplos

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Conversion a matrix m <- as.matrix(loca)
# Muestra la matrix m
# Conversion desde matrix as.loca.p(m)
```

## Véase también

Véase también [loca.p](#)

---

as.loca.p.matrix	<i>as-methods</i>
------------------	-------------------

---

## Description

Conversiones entre la clase `loca.p` y algunas otras

## Argumentos

**x:** es el objeto a convertir a la nueva clase.

**row.names:** Sin uso.

**optional:** Sin uso.

**...:** Otros argumentos, sin uso.

## Valor

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

## Detalles

Métodos para convertir desde y a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada x, y la segunda como la coordenada y, y la tercera (si se ha suministrado) serán los valores de w.

El `data.frame` a convertir a `loca.p` debe tener al menos una columna x para la coordenada x, y una columna y para la coordenada y. Opcionalmente, puede tener una columna w para los valores de w.

## Ejemplos

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Conversion a matrix m <- as.matrix(loca)
# Muestra la matrix m
# Conversion desde matrix as.loca.p(m)
```

## Véase también

Véase también [loca.p](#)

---

as.matrix.loca.p

*as.matrix.loca.p Método para convertir de loca.p a matriz.*

---

## Description

Conversiones entre la clase `loca.p` y algunas otras

## Argumentos

**x:** es el objeto a convertir a la nueva clase.

**row.names:** Sin uso.

**optional:** Sin uso.

**...:** Otros argumentos, sin uso.

## Valor

Si el argumento tiene un valor válido devuelve un nuevo objeto de la nueva clase.

**Detalles**

Métodos para convertir desde `y` a la clase `loca.p`.

No se permiten valores NA en ningún argumento.

La `matrix` `a` a convertir en `loca.p` debe tener al menos dos columnas. La primera será considerada como la coordenada `x`, y la segunda como la coordenada `y`, y la tercera (si se ha suministrado) serán los valores de `w`.

El `data.frame` `a` a convertir a `loca.p` debe tener al menos una columna `x` para la coordenada `x`, y una columna `y` para la coordenada `y`. Opcionalmente, puede tener una columna `w` para los valores de `w`.

**Ejemplos**

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Conversion a matrix m <- as.matrix(loca)
# Muestra la matrix m
# Conversion desde matrix as.loca.p(m)
```

**Véase también**

Véase también [loca.p](#)

---

contour.loca.p

*Gráfica de la función objetivo min-sum*

---

**Description**

`contour` proporcionan la representación gráfica de la función del problema min-sum (`zsum`).

**Uso**

```
## Método S3 para la clase '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), ...)
```

**Argumentos**

**x:** El objeto `loca.p` para calcular el objetivo.

**lp:** Si se proporciona, entonces se usa la norma  $l_p$  en vez de la euclídea.

**xmin:** El valor mínimo del eje `x`.

**xmax:** El valor máximo del eje `x`.

**ymin:** El valor mínimo del eje `y`.

**ymax:** El valor máximo del eje `y`.

- n:** El número de divisiones para la rejilla.
- img:** Una imagen en formato raster para el fondo.
- xleft:** Posición del borde izquierdo de la imagen.
- ybottom:** Posición del borde inferior de la imagen.
- xright:** Posición del borde derecho de la imagen.
- yttop:** Posición del borde superior de la imagen.
- ...:** Otras opciones.

### Detalles

Si  $p < 1$  entonces  $l_p$  no es norma, por tanto, sólo  $p \geq 1$  es válido.

### Valor

La función `contour.loca.p` representa un gráfico de curvas de nivel de la función min-sum (`zsum`).

### Véase también

Véase también [orloca.es-package](#), [plot.loca.p](#) y [loca.p](#).

### Ejemplos

```
# Un objeto loca.p sin pesos
loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# El grafico de curvas de nivel de la fucion min-sum para el objeto
contour(loca)
```

---

distsum

*distsum y distsumgra del paquete orloca*

---

### Description

La función objetivo para el problema de localización min-sum.

### Uso

```
distsum(o, x=0, y=0, lp=numeric(0))
```

### Argumentos

- o:** Un objeto de la clase `loca.p`
- x:** La coordenada x del punto a ser evaluado
- y:** La coordenada y del punto a ser evaluado
- lp:** Si se proporciona, entonces se usa la norma  $l_p$  en vez de la euclídea

**Valor**

distsum devuelve la función objetivo para el problema de localización min-sum,  $\sum_{a_i \in o} w_i d(a_i, (x, y))$ , donde  $d(a_i, (x, y))$  es la distancia euclídea o la distancia  $l_p$  entre  $a_i$  y  $(x, y)$ .

**Detalles**

La función zsum está marcada como obsoleta y será borrada de nuevas versiones del paquete.

**Véase también**

Véase también [orloca.es-package](#) y [distsummin](#).

**Ejemplos**

```
# Un nuevo objeto loca.p sin pesos loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Evaluacion de distsum en (0, 0) distsum(loca)
# Evaluacion de distsum at (1, 3) distsum(loca, 1, 3)
# Calculo de la fucion objetivo en el punto (3, 4) usando la norma lp con p = 2.5 distsum(loca, 3, 4, lp=2.5)
# La funcion gradiente en (1,3) distsumgra(loca, 1, 3)
```

---

distsumgra

*Calculo del gradiente de la funcion distsum*

---

**Description**

distsumgra calcula el gradiente de la función distsum

**Uso**

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

**Argumentos**

- o** Un objeto de clase loca.p.
- x** La coordenada x del punto a evaluar.
- y** La coordenada y del punto a evaluar.
- lp** Si se proporciona, la norma  $l_p$  será usada en vez de la norma euclídea.
- partial** Si  $(x,y)$  es un punto de demanda `partial=T` significa que se ignore dicho punto para el cálculo del gradiente. Esta opción es principalmente para uso interno.

**Valor**

distsumgra devuelve el vector gradiente de la función min-sum del problema de localización,  $\sum_{a_i \in o} w_i d(a_i, (x, y))$ , donde  $d(a_i, (x, y))$  da la distancia euclídea o la distancia  $l_p$  entre  $a_i$  y el punto  $(x, y)$ .

**Detalles**

La función `zsumgra` está marcada como obsoleta y será borrada de nuevas versiones del paquete.

**Ejemplos**

```
# Un nuevo objeto loca.p no ponderado loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Evaluacion de distsum en (0, 0) distsum(loca)
# Evaluacion de distsum en (1, 3) distsum(loca, 1, 3)
# Calculo de la funcion objetivo en el punto (3, 4) usando la norma lp con p = 2.5 distsum(loca, 3,
4, lp=2.5)
# El gradiente de la funcion en el punto (1,3) distsumgra(loca, 1, 3)
```

**See Also**

Véase [orloca-package](#) y [distsum](#).

---

distsuml2min

*distsuml2min en el paquete orloca*

---

**Description**

La función `distsummin` para la norma euclídea ( $l_2$ ). Principalmente para uso interno.

**Uso**

```
distsuml2min(o, x=0, y=0, max.iter=100, eps=0.001, verbose=FALSE, algorithm="Weiszfeld", ...)
```

**Argumentos**

- o** Un objeto de la clase `loca.p`.
- x** La coordenada  $x$  del punto inicial.
- y** La coordenada  $y$  del punto inicial.
- max.iter** Número máximo de iteraciones permitido.
- eps** La norma del gradiente en la regla de parada.
- verbose** Si es TRUE la función proporciona salida detallada.
- algorithm** El algoritmo a utilizar. Los valores válidos son: "gradient" para un algoritmo de gradiente, "search" para un algoritmo de búsqueda local (esta opción está obsoleta), "Weiszfeld" para el algoritmo de Weiszfeld o cualquiera de los métodos válidos para la función `optim`, a saber, "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN". "Weiszfeld" es el valor por defecto.
- ...** Otras opciones para los algoritmos de optimización.

**Valor**

`distsummin` devuelve un vector con las coordenadas del punto solución.

## Detalles

Los algoritmos de Weiszfeld y gradiente incluyen un test de optimalidad para los puntos de demanda. El algoritmo de Weiszfeld también implementa un test de convergencia lenta y un procedimiento acelerador.

Si  $p < 1$  entonces  $l_p$  no es una norma, por ello, solo se admiten valores  $p \geq 1$ .

$l_2$  es la norma euclídea, cuando  $p = 2$  `distsumlpmin` es igual a `distsuml2min`. Pero los cálculos involucrados en la primera forma son mucho mayores.

`max.iter` en el algoritmo SANN es el número de evaluaciones de la función objetivo, por lo que este método requiere de valores grandes de `max.iter` para alcanzar el óptimo.

La función `zsuml2min` está marcada como obsoleta y será borrada de nuevas versiones del paquete.

## Véase también

Vea también [orloca.es-package](#), [loca.p](#) y [distsum](#).

---

distsumlp

*distsumlp y distsumlpgra del paquete orloca*

---

## Description

Las funciones `distsum` y `distsumgra` con norma  $l_p$ . Principalmente para uso interno.

## Uso

`distsumlp(o, x=0, y=0, p=2)` `distsumlpgra(o, x=0, y=0, p=2, partial=F)`

## Argumentos

**o** Un objeto de clase `loca.p`.

**x** La coordenada x del punto a ser evaluado.

**y** La coordenada y del punto a ser evaluado.

**p** La norma  $l_p$  a usar.

**partial** Si  $(x,y)$  es un punto de demanda, `partial=T` significa que se ignore dicho punto para el cálculo del gradiente. Principalmente para uso interno.

## Valor

`distsumlp` devuelve el valor de la función objetivo del problema de localización min-sum con norma  $l_p$ ,  $\sum_{a_i \in o} w_i d(a_i, (x, y))$ , donde  $d(a_i, (x, y))$  es la distancia entre  $a_i$  y el punto  $(x, y)$  usando la norma  $l_p$ .

`distsumlpgra` devuelve el vector gradiente de la función `distsumlp`.

**Details**

Si  $p < 1$  entonces  $l_p$  no es una norma, por tanto, sólo valores  $p \geq 1$  son válidos.

Dado que  $l_2$  es la norma euclídea, cuando  $p = 2$  distsumlp es igual a distsum, y distsumlpgra es igual a distsumgra. Pero los cálculos necesarios son mayores para la primera forma.

La función zsumlp está marcada como obsoleta y será borrada de nuevas versiones del paquete.

**Véase también**

Véase también [distsum](#), [orloca.es-package](#) y [distsumlpmin](#).

---

 distsumlpmin

*distsumlpmin en el paquete orloca*


---

**Description**

La función distsummin con norma  $l_p$ . Principalmente para uso interno.

**Uso**

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

**Argumentos**

**o** Un objeto de la clase `loca.p`.

**x** La coordenada x del punto inicial.

**y** La coordenada y del punto inicial.

**p** Valor de p para la norma  $l_p$ .

**max.iter** Número máximo de iteraciones permitido.

**eps** El módulo del gradiente para la regla de parada.

**verbose** Si es TRUE la función proporciona salida detallada.

**algorithm** El algoritmo a utilizar. Para esta versión del paquete, los valores válidos son: "gradient" para un algoritmo de gradiente, "search" para un algoritmo de búsqueda local (esta opción está obsoleta), "Weiszfeld" para el algoritmo de Weiszfeld o cualquiera de los métodos válidos para la función `optim`, a saber, "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN". "Weiszfeld" es el valor por defecto.

... Otras opciones para los algoritmos de optimización.

**Detalles**

Si  $p < 1$  entonces  $l_p$  no es una norma, por tanto, sólo valores  $p \geq 1$  son válidos.

Dado que  $l_2$  es la norma euclídea, para  $p = 2$  distsumlpmin es equivalente a distsummin. Pero los cálculos involucrados son mayores en la primera forma.

La función zsumlpmin está marcada como obsoleta y será borrada de nuevas versiones del paquete.

**Valor**

distsummin devuelve un vector con las coordenadas del punto solución.

**Ejemplos**

```
# Un nuevo objeto loca.p loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Calcular el minimo sol<-distsummin(loca)
# Mostrar el resultado sol
# Evaluar la funcion en el punto solucion distsum(loca, sol[1], sol[2])
```

**Véase también**

Véase también [distsummin](#), [orloca.es-package](#), [loca.p](#) y [distsum](#).

---

distsummin

*distsummin en el paquete orloca*

---

**Description**

Resuelve el problema de localización min-sum para un objeto dado de la clase `loca.p`.

**Uso**

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

**Argumentos**

- o** Un objeto de la clase `loca.p`.
- x** La coordenada x del punto inicial.
- y** La coordenada y del punto inicial.
- lp** Si se proporciona, la norma  $l_p$  se usa en vez de la norma euclídea.
- max.iter** Número máximo de iteraciones permitido.
- eps** La norma del gradiente en la regla de parada.
- verbose** Si es TRUE la función proporciona salida detallada.
- algorithm** El algoritmo a utilizar. En esta versión del paquete los valores válidos son: "gradient" o "g" para el método basado en gradiente, "search" o "s" para el método de búsqueda local, "ucminf" o "u" para usar optimizar usando ucminf del paquete ucminf, y "weiszfeld" o "w" para el método de Weiszfeld o cualquier otro método válido para la función optim, ahora "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN". "weiszfeld" es el valor por defecto.
- ... Otras opciones para los algoritmos de optimización.

**Detalles**

Los algoritmos de Weiszfeld y gradiente incluyen un test de optimalidad para los puntos de demanda. El algoritmo de Weiszfeld también implementa un test de convergencia lenta y un procedimiento acelerador.

Si  $p < 1$  entonces  $l_p$  no es una norma, por tanto, sólo  $p \geq 1$  es válido.

$l_2$  es la norma euclídea, cuando  $p = 2$  `distsumlpmin` es igual a `distsuml2min`. Pero los cálculos involucrados en la primera forma son mucho mayores.

`max.iter` en el algoritmo SANN es el número de evaluaciones de la función objetivo, por lo que este método requiere de valores grandes de `max.iter` para alcanzar el óptimo.

La función `zsummin` está marcada como obsoleta y será borrada de nuevas versiones del paquete.

**Valor**

`distsummin` devuelve un vector con las coordenadas del punto solución.

**Véase también**

Véase también [orloca.es-package](#), [loca.p](#) y [distsum](#).

**Examples**

```
# Un objeto loca.p sin pesos
loca <- new("loca.p", x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))

# Calcula el minimo
sol <- distsummin(loca)

# Muestra el resultado
sol

# Evaluacion de la funcion objetivo en el punto solucion
distsum(loca, sol[1], sol[2])
```

---

loca.p-class

*Clase de objetos loca.p para Localizacion en Investigacion Operativa*


---

**Description**

Un objeto de la clase `loca.p` representa un problema de localización ponderado con un conjunto finito de puntos de demanda en el plano. El [orloca.es-package](#) está principalmente dedicado a abordar problemas de localización plana.

**Argumentos**

**x:** es un vector con las coordenadas x de los puntos de demanda

**y:** es un vector con las coordenadas y de los puntos de demanda

**w:** es un vector de pesos de los puntos de demanda. Si w se omite entonces todos los pesos se consideran iguales a 1

**label:** Si se explicita, es la etiqueta del nuevo objeto

**Detalles**

El principal generador es `loca.p(x, y, w = numeric(0), label = "")` o alternativamente `new("loca.p", x, y, w = numeric(0), label = "")`.

Las longitudes de los vectores x e y deben ser iguales. La longitud de w debe ser igual a los anteriores o 0. Los valores NA no están permitidos en ninguno de los argumentos.

**Valor**

Si los argumentos son valores válidos, devuelve un objeto de la clase `loca.p`, en caso contrario devuelve un error. `summary(x)` devuelve un resumen del objeto x de la clase `loca.p` y `print(x)` imprime el objeto x de la clase `loca.p` en formato tabla.

**Véase también**

Véase también [orloca.es-package](#).

**Ejemplos**

```
# Un objeto loca.p sin pesos loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# o loca <- new("loca.p", x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# Un ejemplo con pesos y nombre locb <- new("loca.p", x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1), w =
c(1, 2, 1, 2), label = "Caso Ponderado")
```

---

persp.distsum

*Grafica de la funcion objetivo min-sum*

---

**Description**

persp proporciona la representación gráfica de la función objetivo del problema min-sum (`distsum`).

**Uso**

```
## Metodo S3 para la clase 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, ...)
```

**Argumentos**

- x:** El objeto `loca.p` para calcular el objetivo
- lp:** Si se proporciona, entonces se usa la norma  $l_p$  en vez de la euclídea
- xmin:** El valor mínimo del eje x
- xmax:** El valor máximo del eje x
- ymin:** El valor mínimo del eje y
- ymax:** El valor máximo del eje y
- n:** El número de divisiones para la rejilla
- ... Otras opciones

**Detalles**

Si  $p < 1$  entonces  $l_p$  no es norma, por tanto, sólo  $p \geq 1$  es válido.

**Valor**

Un gráfico 3D de la función min-sum.

**Véase también**

Véase también [orloca.es-package](#), [plot.loca.p](#) y [loca.p](#).

**Ejemplos**

```
# Un objeto loca.p sin pesos loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# El grafico 3D del objeto loca.p persp(loca)
```

---

plot

*Grafico de un objeto de la clase loca.p*

---

**Description**

Este método proporciona una representación gráfica de un objeto de la clase `loca.p`.

**Uso**

```
## Metodo S3 para la clase 'loca.p'
plot(x, xlab = "", ylab = "", main = gettext("Plot of loca.p object", domain = "R-orloca"), 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), ...)
```

**Argumentos**

- x:** El objeto `loca.p` a representar.
- xlab:** La etiqueta para el eje x.
- ylab:** La etiqueta para el eje y.
- main:** El título principal del gráfico.
- img:** Una imagen en formato raster para el fondo.
- xlim:** Límite sobre el eje x del gráfico.
- ylim:** Límite sobre el eje y del gráfico.
- xleft:** Posición del borde izquierdo de la imagen.
- ybottom:** Posición del borde inferior de la imagen.
- xright:** Posición del borde derecho de la imagen.
- yttop:** Posición del borde superior de la imagen.
- ...:** Otras opciones gráficas.

**Detalles**

Gráfico de los puntos de demanda con límites de evaluación automáticos.

**Valor**

La representación gráfica de los puntos de demanda.

**Véase también**

Véase también [orloca.es-package](#), [loca.p](#) y [plot](#).

**Ejemplos**

```
# Un objeto de la clase loca.p sin pesos loca <- loca.p(x = c(-1, 1, 1, -1), y = c(-1, -1, 1, 1))
# El grafico del objeto loca.p plot(loca)
```

---

rloca.p

*Generador de instancias aleatorias de objetos de la clase loca.p*

---

**Description**

Devuelve una instancia aleatoria de un objeto de la clase `loca.p` en una región dada.

**Uso**

```
rloca.p(n, xmin = 0, xmax = 1, ymin = 0, ymax = 1, groups = 0, xgmin = xmin, xgmax = xmax,
        ygmin = ymin, ygmax = ymax)
```

### Argumentos

**n:** El número de puntos de demanda.

**xmin:** Mínimo valor para la coordenada x de los puntos de demanda.

**xmax:** Máximo valor para la coordenada x de los puntos de demanda.

**ymin:** Mínimo valor para la coordenada y de los puntos de demanda.

**ymax:** Máximo valor para la coordenada y de los puntos de demanda.

**groups:** El número de grupos (de aproximadamente igual tamaño) o una lista con los tamaños de los grupos a generar. En el segundo caso n se ignora.

**xgmin:** Mínimo valor para la coordenada x de los puntos de demanda respecto del punto referencia del grupo.

**xgmax:** Máximo valor para la coordenada x de los puntos de demanda respecto del punto referencia del grupo.

**ygmin:** Mínimo valor para la coordenada y de los puntos de demanda respecto del punto referencia del grupo.

**ygmax:** Máximo valor para la coordenada y de los puntos de demanda respecto del punto referencia del grupo.

### Detalles

n debe ser al menos 1.

xmin debe ser menor o igual que xmax.

ymin debe ser menor o igual que ymax.

Cuando se suministra valor no nulo para groups los puntos se generan en dos fases, en la primera se genera un punto de referencia, en la segunda se genera un desplazamiento sobre dicho punto de referencia que se suma a éste.

Obsérvese que groups = 1 no es equivalente a groups = 0, debido a que en el primer caso se genera un punto de referencia en la primera etapa.

### Valor

Si los argumentos son valores válidos, devuelve un nuevo objeto de la clase loca\_p, en otro caso informa de un error.

### Véase también

Véase también [orloca-package](#) y [loca.p](#)

### Ejemplos

```
# Un objeto aleatorio loca.p en el cuadrado unidad con 5 puntos de demanda rloca.p(5)
```

```
# En otra region rloca.p(10, xmin=-2, xmax=2, ymin=-2, ymax=2)
```

```
# Cinco grupos rloca.p(48, groups=5)
```

```
# Tres grupos de distinto tamaño rloca.p(1, groups=c(10, 7, 2))
```

*zsum**zsum*

---

**Description**

La función `zsum` esta obsoleta y podría ser suprimida en sucesivas versiones del paquete. Use [distsum](#) en su lugar.

**Usage**`zsum(...)`**Arguments**

...            Parámetros pasados a `distsum`

---

*zsumgra**zsumgra*

---

**Description**

La función `zsumgra` esta obsoleta y podría ser suprimida en sucesivas versiones del paquete. Use [distsumgra](#) en su lugar.

**Usage**`zsumgra(...)`**Arguments**

...            Parámetros pasados a `distsumgra`

---

*zsummin**zsummin*

---

**Description**

La función `zsummin` esta obsoleta y podría ser suprimida en sucesivas versiones del paquete. Use [distsummin](#) en su lugar.

**Usage**`zsummin(...)`**Arguments**

...            Parámetros pasados a `distsummin`

# Index

- \* **Andalucia**
  - andalusia, 4
- \* **Andalusia**
  - andalusia, 4
- \* **classes**
  - as-methods, 4
  - as.data.frame.loca.p, 5
  - as.loca.p, 6
  - as.loca.p.data.frame, 7
  - as.loca.p.matrix, 8
  - as.matrix.loca.p, 9
  - contour.loca.p, 10
  - distsum, 11
  - distsumgra, 12
  - distsuml2min, 13
  - distsumlp, 14
  - distsumlpmin, 15
  - loca.p-class, 17
  - persp.distsum, 18
  - plot, 19
- \* **datagen**
  - rloca.p, 20
- \* **data**
  - andalusia, 4
- \* **hplot**
  - contour.loca.p, 10
  - persp.distsum, 18
  - plot, 19
- \* **methods**
  - as-methods, 4
  - as.data.frame.loca.p, 5
  - as.loca.p, 6
  - as.loca.p.data.frame, 7
  - as.loca.p.matrix, 8
  - as.matrix.loca.p, 9
- \* **obsoleta**
  - zsum, 22
  - zsumgra, 22
  - zsummin, 22
- \* **optimize**
  - distsum, 11
  - distsumgra, 12
  - distsuml2min, 13
  - distsumlp, 14
  - distsumlpmin, 15
  - loca.p-class, 17
  - orloca.es-package, 2
- \* **package**
  - orloca.es-package, 2
- andalucia (andalusia), 4
- andalusia, 4
- as-methods, 4
- as.data.frame.loca.p, 5
- as.loca.p, 6
- as.loca.p.data.frame, 7
- as.loca.p.matrix, 8
- as.matrix.loca.p, 9
- contour, loca.p-method (contour.loca.p), 10
- contour.loca.p, 10
- distsum, 11, 13–17, 22
- distsum, loca.p-method (distsum), 11
- distsumgra, 12, 22
- distsumgra, loca.p-method (distsumgra), 12
- distsuml2min, 13
- distsuml2min, loca.p-method (distsuml2min), 13
- distsumlp, 14
- distsumlp, loca.p-method (distsumlp), 14
- distsumlpgra (distsumlp), 14
- distsumlpgra, loca.p-method (distsumlp), 14
- distsumlpmin, 15, 15
- distsumlpmin, loca.p-method (distsumlpmin), 15

`distsummin`, [12](#), [16](#), [16](#), [22](#)  
`distsummin`, `loca.p`-method (`distsummin`),  
[16](#)

`initialize` (`loca.p`-class), [17](#)

`loca.p`, [5–11](#), [14](#), [16](#), [17](#), [19–21](#)  
`loca.p` (`loca.p`-class), [17](#)  
`loca.p`-class, [17](#)

`orloca` (`orloca.es`-package), [2](#)  
`orloca`-package, [3](#)  
`orloca.es`-package, [2](#)

`persp`, `loca.p`-method (`persp.distsum`), [18](#)  
`persp.distsum`, [18](#)  
`persp.loca.p` (`persp.distsum`), [18](#)  
`plot`, [19](#), [20](#)  
`plot.loca.p`, [11](#), [19](#)  
`print` (`loca.p`-class), [17](#)

`rloca.p`, [20](#)

`summary` (`loca.p`-class), [17](#)

`zsum`, [22](#)  
`zsumgra`, [22](#)  
`zsummin`, [22](#)