

Package ‘locationgamer’

May 8, 2026

Type Package

Title Identification of Location Game Equilibria in Networks

Version 0.1.0

Author Maximilian Zellner

Maintainer Maximilian Zellner <zellnermaximilian@gmail.com>

Description Identification of equilibrium locations in location games (Hotelling (1929) <doi:10.2307/2224214>). In these games, two competing actors place customer-serving units in two locations simultaneously. Customers make the decision to visit the location that is closest to them. The functions in this package include Prim algorithm (Prim (1957) <doi:10.1002/j.1538-7305.1957.tb01515.x>) to find the minimum spanning tree connecting all network vertices, an implementation of Dijkstra algorithm (Dijkstra (1959) <doi:10.1007/BF01386390>) to find the shortest distance and path between any two vertices, a self-developed algorithm using elimination of purely dominated strategies to find the equilibrium, and several plotting functions.

License MIT + file LICENSE

Imports graphics

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-12-18 09:30:02 UTC

Contents

createDistance	2
dijkstra	2

euclidDistance	3
lgsolve	4
plotDijkstra	5
plotNetwork	5
plotPrim	6
primDistance	7
randomCoordinates	7

Index	9
--------------	----------

createDistance	<i>Create distance matrix for a completely connected network</i>
----------------	--

Description

Create distance matrix for a completely connected network

Usage

```
createDistance(coordMatrix)
```

Arguments

coordMatrix A matrix containing all the x and y coordinates of the network vertexes

Value

A square matrix containing the Euclidean distances between all vertexes, assuming that the network is completely connected.

Examples

```
coordMatrix <- matrix(c(0,10,15,20,30,30,15,15),ncol = 2)
createDistance(coordMatrix)
```

dijkstra	<i>Shortest path through network using dijkstra's algorithm</i>
----------	---

Description

This function finds the shortest path from a starting node to an end node in a network specified by an edge matrix and vertex coordinates. Position i,j of the edge matrix is one if there is an edge between the i th and j th vertex, zero otherwise. The function returns the path NA with length infinity if the network is disconnected, i.e. if no shortest path can be found.

Usage

```
dijkstra(edgeMatrix, coordMatrix, initialNode, endNode, nNodes)
```

Arguments

edgeMatrix	A square matrix consisting of zeros and ones. Has to be zero on the diagonals
coordMatrix	A data frame containing the x and y coordinates of each network vertex
initialNode	A number corresponding to the start node/ vertex
endNode	A number corresponding to the end node/ vertex
nNodes	The number of vertices/ nodes in the network

Value

A list consisting of a vector with the vertices/ nodes visited by the shortest path and the length of the shortest path.

Examples

```
initialNode <- 1
endNode <- 4
nNodes <- 4
edgeMatrix <- matrix(0, nrow = 4, ncol = 4)
edgeMatrix[,1] <- c(0,1,0,0)
edgeMatrix[,2] <- c(1,0,1,1)
edgeMatrix[,3] <- c(0,1,0,0)
edgeMatrix[,4] <- c(0,1,0,0)
coordMatrix <- matrix(c(0,10,15,20,30,30,15,15), ncol = 2)
dijkstra(edgeMatrix, coordMatrix, initialNode, endNode, nNodes)
```

euclidDistance	<i>Euclidean distance between two points</i>
----------------	--

Description

Euclidean distance between two points

Usage

```
euclidDistance(x1, y1, x2, y2)
```

Arguments

x1	x-coordinate of point 1
y1	y-coordinate of point 1
x2	x-coordinate of point 2
y2	y-coordinate of point 2

Value

The Euclidean distance between points 1 and 2 as a number

lgsolve

Equilibrium locations of location game

Description

Function finds the equilibrium locations of a location game, similar to a hotelling game. Clients choose the location closest to them.

Usage

```
lgsolve(edgeMatrix, coordMatrix, nPlayers = 2, demandLoc)
```

Arguments

edgeMatrix	A square matrix consisting of zeros and ones. Has to be zero on the diagonals
coordMatrix	A data frame containing the x and y coordinates of each network vertex
nPlayers	Number of players in the location game. Default is set to 2, which is the only number of players supported right now.
demandLoc	A vector containing the demand or profit at each vertex of the network

Value

A list with two components. A matrix with zeros and ones, where a one symbolizes a equilibrium location. The row index denotes the location of player 1, and the column index the location chosen by player 2. The second entry is a summary of all equilibrium locations and the payoffs for player 1 and 2.

Examples

```
edgeMatrix <- matrix(0, nrow = 6, ncol = 6)
edgeMatrix[,1] <- c(0,1,0,0,0,0)
edgeMatrix[,2] <- c(1,0,1,0,1,0)
edgeMatrix[,3] <- c(0,1,0,0,0,0)
edgeMatrix[,4] <- c(0,0,0,0,1,0)
edgeMatrix[,5] <- c(0,1,0,1,0,1)
edgeMatrix[,6] <- c(0,0,0,0,1,0)
coordMatrix <- matrix(c(0,3,0,2,0,1,1,3,1,2,1,1), nrow = 6, ncol = 2, byrow = TRUE)
demandLoc <- c(100, 100, 100, 100, 100, 100)
lgsolve(edgeMatrix, coordMatrix, 2, demandLoc)
```

plotDijkstra	<i>Plot shortest path between two points in a network</i>
--------------	---

Description

This function plots the entire network and shortest path between two points. The parameter `dijkstraPath` is obtained by the function `dijkstra`, in which one has to specify the initial and end node of the path.

Usage

```
plotDijkstra(edgeMatrix, coordMatrix, dijkstraPath)
```

Arguments

<code>edgeMatrix</code>	A matrix containing zeros and ones if an edge between two vertexes is absent or not
<code>coordMatrix</code>	A data frame containing the x and y coordinates of each vertex of the network
<code>dijkstraPath</code>	A vector of numbers corresponding to the vertexes of the shortest path through the network

Value

Function outputs a two-dimensional plot

Examples

```
edgeMatrix <- matrix(0, nrow = 4, ncol = 4)
edgeMatrix[,1] <- c(0,1,0,0)
edgeMatrix[,2] <- c(1,0,1,1)
edgeMatrix[,3] <- c(0,1,0,0)
edgeMatrix[,4] <- c(0,1,0,0)
coordMatrix <- matrix(c(0,10,15,20,30,30,15,15), ncol = 2)
dijkstraPath <- c(4,2,1)
plotDijkstra(edgeMatrix, coordMatrix, dijkstraPath)
```

plotNetwork	<i>Plotting a network consisting of edges and vertexes</i>
-------------	--

Description

Plotting a network consisting of edges and vertexes

Usage

```
plotNetwork(edgeMatrix, coordMatrix)
```

Arguments

edgeMatrix	A matrix containing zeros and ones if an edge between two vertexes is absent or not
coordMatrix	A data frame containing the x and y coordinates of each vertex of the network

Value

A plot of the connected network `edgeMatrix <- matrix(0, nrow = 4, ncol = 4)` `edgeMatrix[1,1] <- c(0,1,0,0)` `edgeMatrix[,2] <- c(1,0,1,1)` `edgeMatrix[,3] <- c(0,1,0,0)` `edgeMatrix[,4] <- c(0,1,0,0)` `coordMatrix <- matrix(c(0,10,15,20,30,30,15,15), ncol = 2)` `plotNetwork(edgeMatrix, coordMatrix)`

plotPrim	<i>Plotting minimum spanning tree connecting all vertexes</i>
----------	---

Description

Plotting minimum spanning tree connecting all vertexes

Usage

```
plotPrim(minimumSp, coordMat)
```

Arguments

minimumSp	A data frame in which each row corresponds to an edge between two numbered vertexes Use function <code>primDistance</code> to obtain minimum spanning tree using Prim's algorithm.
coordMat	A matrix containing all the x and y coordinates of the network vertexes.

Examples

```
minimumSp <- matrix(c(1,4,4,3,2,3), ncol = 2)
coordMatrix <- matrix(c(0,10,15,20,30,30,15,15), ncol = 2)
plotPrim(minimumSp, coordMatrix)
```

primDistance	<i>Minimum spanning tree using Prim's algorithm</i>
--------------	---

Description

Minimum spanning tree using Prim's algorithm

Usage

```
primDistance(distMatrix)
```

Arguments

`distMatrix` A square matrix containing the distances between all vertexes of a network

Value

A matrix with rows describing which vertex is connected to which other vertex.

Examples

```
distMatrix <- matrix(c(0,10,20,30,10,0,40,60,20,40,0,30,30,60,30,0),  
  nrow = 4, ncol = 4, byrow = TRUE)  
primDistance(distMatrix)
```

randomCoordinates	<i>Create random coordinates for network vertexes</i>
-------------------	---

Description

Create random coordinates for network vertexes

Usage

```
randomCoordinates(nNodes, xMax, xMin, yMax, yMin)
```

Arguments

<code>nNodes</code>	The number of vertexes/ nodes in the network
<code>xMax</code>	The maximum x-coordinate of the nodes in the network
<code>xMin</code>	The minimum x-coordinate of the nodes in the network
<code>yMax</code>	The maximum y-coordinate of the nodes in the network
<code>yMin</code>	The minimum y-coordinate of the nodes in the network

Value

A data frame with dimensions nNodes x 2 containing the x and y coordinates of the network's vertexes

Examples

```
nNodes <- 10
xMax <- 2000
xMin <- 0
yMax <- 3000
yMin <- 200
randomCoordinates(nNodes, xMax, xMin, yMax, yMin)
```

Index

`createDistance`, [2](#)

`dijkstra`, [2](#)

`euclidDistance`, [3](#)

`lgsolve`, [4](#)

`plotDijkstra`, [5](#)

`plotNetwork`, [5](#)

`plotPrim`, [6](#)

`primDistance`, [7](#)

`randomCoordinates`, [7](#)