Package 'spatgraphs'

July 2, 2014

Type Package

Title Graphs for spatial point patterns

Version 2.62

Date 2012-09-26

Author Tuomas Rajala

Maintainer Tuomas Rajala <tuomas.rajala@iki.fi>

Depends

Suggests Matrix, igraph, rgl

Description Graphs, graph visualization and graph component
calculations, ment to be used as a tool in spatial point
pattern analysis. See package 'spatstat' for more info about spatial point patterns.

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-09-26 08:59:54

NeedsCompilation yes

R topics documented:

`spatgraphs-package` *Graphs for spatial point patterns*

Description

Compute various spatial graphs for 2D and 3D spatial point patterns such as the ppp-objects in R-package `spatstat`. Also capable of cluster/component computation and visualization.

Details

This package provides the following graph computations, all handled by the `spatgraph`-function:

Graph	relation x~y
Geometric	||x-y||<R
Mass geometric	||x-y||<m(x)
Spheres of Influence	||x-y||<dn(x)+dn(y)
Mark crossing	||x-y||<m(x)+m(y)
k-Nearest neighbour	y in knn(x)
Relative Neighbourhood	see refs.
Radial spanning tree	see refs.
Minimum spanning tree	see refs.
Gabriel graph	see refs.
Class cover catch	see refs.
Delaunay triangulation	see refs.
Signal-to-noise-ratio	see refs.

where
    ||.|| ~ Euclidian distance
    m(x) ~ mass, size i.e. real mark of x
    dn(x) ~ the distance to the nearest neighbour of x.
    knn(x) ~ the k nearest neighbours set of x

The minimum spanning tree is computed using Prim's algorithm.

The classes sg and sgc are defined, with their own plot- and print-methods. 3D plotting requires package `rgl`.

For adjacency matrices see functions sg2adj and adj2sg.

In addition to the main workhorse spatgraph-function are the following functions:

```
Function name          Description
-----------------------------------------------------------
spatcluster           Compute clusters (connected components)
shortestPath          Find the shortest edge-path between two points
edgeLengths           Lenghts of edges in a graph
sg2sym                Make the edges symmetric
sg2adj                Convert to an adjacency matrix form
cut.sg                Cut edges longer than given R>0
prune.sg              Prunes the graph, aimed for MST
clip.sg             Clip edges crossing the borders of a window. Useful for non-convex regions.
sg2dxf                Write a graph to a dxf file
sg2igraph           Change the sg-object to igraph-object of package \code{\link{igraph}}
sg2sparse             Convert between sg-object and Matrix-package sparseMatrix
runif3d               Simple 3d uniform pp generation
spectral.sg           Spectral clustering
```

Author(s)

Tuomas Rajala
University of Jyvaskyla, Finland
tuomas.rajala@iki.fi

References

Dousse, O., Baccelli, F. \& Thiran, P.: Impact of Interferences on Connectivity in Ad Hoc Networks. IEEE/ACM Transactions on Networking,13 (2),p. 425-436,2005.

Marchette, D.: Random Graphs for Statistical Pattern Recognition, Wiley 2004.

See Also

Spatial point processes in general, see the package `spatstat`.

For more versatile Voronoi/Delaunay handling, see the package `tripack` or `deldir`.

The package `rgl` is required for 3D-plotting.

Examples

```
graph_example2d<-function(n=50,k=2,R=0.2)
{
 pp2d<-list(x=runif(n),y=runif(n),n=n,window=list(x=c(0,1),y=c(0,1)))
 e1<-spatgraph(pp2d,"geometric",par=R)
 e2<-spatgraph(pp2d,"knn",par=k)
 e3<-spatgraph(pp2d,"MST")
```

```
   A<-spatcluster(e2)
   par(mfrow=c(1,3))
   plot(pp2d,main=paste("Geometric,R =",R))
   plot(e1,pp2d)
   plot(pp2d,main=paste("k-nn, k =",k))
   plot(e2,pp2d)
   plot(A,pp2d,pch=19)
   plot(pp2d, main="Minimum spanning tree")
   plot(e3,pp2d)
  }
  graph_example2d()

## Not run:
## 3d example, requires library rgl
  library(rgl)
  graph_example3d<-function(n=200)
  {
w<-c(0,1)
phi<-runif(n,0,pi);tau<-runif(n,0,2*pi);r<-runif(n)^0.33
pp3d<-list(x=r*sin(tau)*cos(phi),y=r*cos(phi)*cos(tau),z=r*cos(phi),n=n,window=list(x=w,y=w,z=w))
e<-spatgraph(pp3d,"RST",par=c(x=0,y=0,z=0))
plot3d(pp3d,size=2, main="Radial spanning tree",col="black")
plot(e,pp3d,col="plum")
  }
  graph_example3d()

## End(Not run)
```

---

spatgraph-shake *shake*

Description

Shake (displace) the points a little. Grid like sampling gives problems with e.g. Delaunay graph (collinearity). Similar to `jitter` in `spatstat`

Details

Date:	2008-07-30
License:	GPL v2 or later

---

spatgraph-spectral *Spectral clustering*

Description

Spectral clustering: Given a weighted adjacency matrix W and G=diag(rowSums(W)), the Laplacian of the graph is L=G-W. We compute the eigenvalues of L, and for the 2:(m+1) smallest eigenvalues, we use the corresponding eigenvectors to do a K-means clustering.

Usage

```
spectral.sg(x, pp, m=2, K=3, diagplot=FALSE, ...)
```

Arguments

x	Weighted graph, sg-object. Will be transformed to W using sg2wadj().
pp	Point pattern.
m	Number of eigenvectors to use.
K	Number of clusters to look for using K-means.
diagplot	Plot some diagnostics: Result: first 20 eigenvalues and m eigenvectors.
...	Not used at the moment.

Details

Date:	2011-04-15
License:	GPL v2 or later

Value

List with $id element denoting each points cluster id, and $sgc is a spatcluster-object with the clusters.

---

spatgraphs-adj2sg *Adjacency matrix to edgelist and vice versa*

Description

Maps between edgelist and the adjancency matrix representation. Works also with sparse matrices using Matrix-package.

Usage

```
sg2adj(x)
adj2sg(x)
sg2sparse(x)
sparse2sg(x)
```

Arguments

x	sg-object, sgadj-object, or sparseMatrix-object.

---

spatgraphs-clip.sg *Clip edges that cross window border*

Description

Cuts off the edges that cross pattern's window border. Needs 'owin'-object from spatstat.

Usage

```
clip.sg(x, pp, window=NULL)
```

Arguments

x	spatgraph object.
pp	Point pattern from which x is computed.
window	Optional owin-object if the one in pp is not suitable.

Details

Date:	2008-07-30
License:	GPL v2 or later

---

spatgraphs-cut.sg *cut and prune*

Description

Cut and prune MST (or others).

Cut the edges of graph with length more that R>0.

Prune cuts away all branches of the graph which are shorter than level>0.

Details

Date:	2008-07-30
License:	GPL v2 or later

spatgraphs-mailer *mailer*

Description

Mailer: Unix/Linux only, needs the program `mail`: send an e-mail. Useful for notifying ending of long calculation on remote computers.

Took: One formatting of time lapsed since given Sys.time-object.

Details

Date:	2008-07-30
License:	GPL v2 or later

spatgraphs-other *Make the graph symmetric/compute edge lengths*

Description

`sg2sym` makes the graph symmetric.

`edgeLengths` returns the distances of edge-connected points as a list x=(i,j,d) such that distance(x$i[k],x$j[k])=x$d[k], k=1,...,NumbeOfEdges.

Usage

```
sg2sym(x, way=1)
edgeLengths(x, pp, ...)
```

Arguments

`x`	sg-object.
`pp`	Point pattern, for distances.
`way`	(in sg2sym). If 1, use (xy OR yx) rule, if anything else use (xy AND yx) rule.
`...`	ignored

spatgraphs-plot.sg *plot.sg*

Description

Plot the edges of graph, or color the clusters.

Usage

```
   ## S3 method for class 'sg'
plot(x, pp, add=TRUE, which=NULL, directed=0, add.points=FALSE,
        points.col="black", points.pch=1, points.cex=1, lines.col="gray30", ...)
```

Arguments

x	spatgraph/spatcluster object
pp	point pattern
add	Add the lines to an existing plot (such as plot(pp)) or draw a new plot.
which	Vector of indices or a Boolean vector of size n: Draw only edges starting at these points.
directed	Draw arrows with this size. If 0, no arrows.
add.points	Should we draw the points after the lines are drawn. Uses points, only 2D.
points.col	Color(s) for the points if they are added.
points.pch	Plotting character for points if they are added.
points.cex	Plotting size of point character if points are added.
lines.col	Color(s) for the lines.
...	line size etc. for corresponding function (lines, arrows, rgl.lines).

Details

Date:	2008-07-30
License:	GPL v2 or later

spatgraphs-print.sg *print.sg*

Description

Print method of sg, sgadj and sgc-object of package spatgraphs.

`spatgraphs-runif3d` *runif3d*

Description

Simple simulation of uniform 3d point pattern.

Usage

```
runif3d(n = c(10), window = list(x = c(0, 1), y = c(0, 1), z = c(0, 1)))
```

Arguments

`n`	Vector of point counts, e.g. c(10,10) is two type pattern with 10 points per type.
`window`	Rectangular cuboid window limits.

Details

Simulates uniformly distributed points in 3d rectangle.

`spatgraphs-sg` *sg*

Description

Edge list-of-lists class for spatgraphs. Methods: print, plot, summary, t.

Transposing is reversing edges.

Details

The object is a list with members

'edges'	a list with each element edges[[i]] being a vector of point i's neighbours' indices. So i->j iff j in edges[[i]].
'N'	cardinality of the point pattern.
'symmetric'	Boolean. Is the graph symmetric. Might also be '?' (would not trust this one).
'type'	Type of the graph as in the function call.
'parameters'	Parameters, as in the function call.

`spatgraphs-sg2dxf` *sg2dxf*

Description

Write the graph to an AutoCAD Drawing Exchange Format or dxf file.

Arguments

x	spatgraph object.
pp	point pattern.
file	output filename.

Details

Date:	2008-07-30
License:	GPL v2 or later

---

spatgraphs-sg2igraph *sg2igraph*

Description

Convert spatgraph-object to an igraph-object, and vice versa.

Usage

```
sg2igraph(g, pp=NULL)
igraph2sg(g)
```

Arguments

g	The object to be converted.
pp	point pattern. If none given, no details of the points will survive.

Details

Date:	2009-04-29
License:	GPL v2 or later

spatgraphs-sgc *sgc*

Description

Cluster-object class for spatgraphs. Methods: print, plot. A cluster is a connected component.

Details

A list with elements

'clusters'	A list with N vectors, each containing the indices of corresponding cluster's points. N clusters.
'nclusters'	Number of clusters
'N'	Number of points in the pattern
'type'	'type' of the original graph
'parameters'	'parameters' of the original graph

spatgraphs-shortestPath

Shortest path between nodes i and j

Description

Find the shortest edgeconnected path between two given nodes/points with indices i and j (in pp).

Usage

```
shortestPath(i, j, g, pp=NULL, dbg=FALSE)
```

Arguments

i	The start node of the path to find.
j	The target node of the path to find.
g	Graph which defines the edges.
pp	Point pattern. If given, the edges are of Euclidian length, otherwise each edge is of length 1.
dbg	Print runtime messages.

Details

Date:	2008-09-25
License:	GPL v2 or later

Returns the distance and edges along the shortest path.

Make sure the graph is symmetric.

The algorithm is Dijkstra\'s algorithm.

References

E. W. Dijkstra: A note on two problems in connexion with graphs. 'Numerische Mathematik', 1 (1959), p. 269-271.

http://en.wikipedia.org/wiki/Dijkstra's_algorithm

`spatgraphs-spatcluster` *spatcluster*

Description

Compute the list of clusters i.e. connected components given a graph-object from `spatgraph`.

Arguments

`x`	spatgraph-object
`dbg`	Boolean, print additional messages
`sym`	Boolean, symmetricise the graph first?

Details

Date:	2008-07-29
License:	GPL v2 or later

Value

An object of class `sgc`.

`spatgraphs-spatgraph` *spatgraph*

Description

Compute a spatial graph for a given 2D- or 3D- point pattern.

Usage

```
spatgraph(pp, type="knn", par=NULL, preprocessR=0, dbg=FALSE,
        doDists=FALSE, preDists=NULL, preGraph=NULL, toroidal=FALSE, include=NULL)
```

Arguments

pp	Point pattern with members x,y,n,window. Window must have x- and y-limits according to given x,y. see package spatstat, class ppp.
type	One of the supported graph types, see below.
par	Parameter(s) for the graph, see below.
preprocessR	If >0 first compute geometric graph and then the type graph using the preprocessed edgelist. Useful for narrowing down the search space for bigger pp's.
dbg	Boolean, print additional information during the execution.
doDists	Boolean, default FALSE. If true, precompute and store the pairwise distances. Speeds things up quite a lot but takes O(n\^2) memory!
preDists	Optional: precalculated distance matrix for the points. Can be used in case of non-standard metric spaces.
preGraph	Precalculated graph to be used as search space for neighbours. Useful for large computation of e.g. Gabriel graphs.
toroidal	Make a toroidal distance calculation. Not useful when visualizing but useful for edge correction in summary calculations.
include	A 0-1-vector describing which points' neighbourhoods to calculate, i.e. include[i]=1 => compute neighbourhood of pp[i].

Details

The following 'type' values are accepted, note that some of them need also the 'par':

'geometric'	par=numeric>0. Geometric graph, par = connection range.
'knn'	par=integer>0. k-nearest neighbours graph, par = k.
'mass_geometric'	Connect two points if ||x-y||<m(x).
'gabriel'	Gabriel graph. Optional parameter par=k: Allow k points in the shared circle between two points. Norma
'delaunay'	Delaunay triangulation. Note! Only 2D, and no check for collinearity: use rjitter from spatstat if uns
'MST'	Minimal spanning tree.
'markcross'	Connect two points if ||x-y||<m(x)+m(y).
'SIG'	Spheres of Influence. Connect x and y iff ||x-y|| < dn(x) + dn(y), i.e. if their 'individual domains' intersect
'RST'	par=c(x0,y0,z0). Radial spanning tree, par=origin of radiation. z0 i.e. z-coordinate can be omitted if in 2I
'RNG'	Relative neighbourhood graph. Connect x and y if the lens between them is empty. Lens in this case mean
'CCC'	par=integer (or string). Class-Cover-Catch, par=target type. This needs a multivariate point pattern.
'STIR'	par=c(noise,alpha,beta,gamma). Signal-To-Noise-Ratio, par: background noise, signal attenuation (1/r^al

where

```
m(x) ~ real valued mark for x (size, mass, diameter, transmission power...)
dn(x) ~ distance to the nearest neighbour of x
```

The graphs 'mass_geometric', 'markcross' and 'STIR' use scalar marks, 'CCC' class marks (e.g. integer). If given 'pp' has no marks it will be marked with 1.0's.

If your window is non-convex polygon, you can use `clip.sg` to cut extra edges that cross the border.

Value

An object of class `sg`.

References

Dousse, O., Baccelli, F. \& Thiran, P.: Impact of Interferences on Connectivity in Ad Hoc Networks. IEEE/ACM Transactions on Networking,13 (2),p. 425-436,2005.

Marchette, D.: Random Graphs for Statistical Pattern Recognition, Wiley 2004.

`spatgraphs-weight` *Weight edges*

Description

Weight edges with a given function of node-node Euclidian distances.

Usage

```
weight.sg(x, pp, f=function(x) exp(-x^2/scale), scale=1, ...)
sg2wadj(x)
```

Arguments

`x`	Graph.
`pp`	Point pattern.
`f`	R-function for weight computation; a function of distance.
`scale`	Scaling factor used in (possibly) in f.
`...`	Further parameters (ignored at the moment.)

Details

Date:	2011-04-14
License:	GPL v2 or later

Value

returns the graph x with additional compontent $weights that contains the weights for edges listed in $edges. For example, if x$edges[[1]] is c(2,3), and x$weights is c(1, 0.5), then the weight of edge 1->2 is 1 and edge 1->3 is 0.5.

Index