

# Package ‘spatgraphs’

November 12, 2009

**Type** Package

**Title** Graphs for spatial point patterns

**Version** 2.30

**Date** 2009-11-11

**Author** Tuomas Rajala

**Maintainer** Tuomas Rajala <tuomas.a.rajala@jyu.fi>

**Depends**

**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** 2009-11-12 11:33:24

## R topics documented:

|                                   |    |
|-----------------------------------|----|
| spatgraphs-package . . . . .      | 2  |
| spatgraph-shake . . . . .         | 4  |
| spatgraphs-adj2sg . . . . .       | 4  |
| spatgraphs-cut.sg . . . . .       | 5  |
| spatgraphs-mailer . . . . .       | 5  |
| spatgraphs-other . . . . .        | 5  |
| spatgraphs-plot.sg . . . . .      | 6  |
| spatgraphs-print.sg . . . . .     | 7  |
| spatgraphs-runif3d . . . . .      | 7  |
| spatgraphs-sg . . . . .           | 7  |
| spatgraphs-sg2dxf . . . . .       | 8  |
| spatgraphs-sg2igraph . . . . .    | 8  |
| spatgraphs-sgc . . . . .          | 9  |
| spatgraphs-shortestPath . . . . . | 9  |
| spatgraphs-spatcluster . . . . .  | 10 |
| spatgraphs-spatgraph . . . . .    | 10 |

---

spatgraphs-package *Graphs for spatial point patterns*

---

### Description

Compute various graph edge sets 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   < d(x) + d(y)$       |
| Mark crossing          | $  x-y   < m(x) + m(y)$       |
| k-Nearest neighbour    | $x \text{ in } \text{knn}(y)$ |
| Relative Neighbourhood | see refs.                     |
| Radial spanning tree   | see refs.                     |
| Minimum spanning tree  | see refs.                     |
| Gabriel graph          | see refs.                     |
| Class cover catch      | see refs.                     |
| Delauney triangulation | see refs.                     |
| Signal-to-noise-ratio  | see refs.                     |

where

$||.||$  ~ euclidian distance  
 $m(x)$  ~ mass, size i.e. real mark of  $x$   
 $d(x)$  ~ the distance to the nearest neighbour of  $x$ .  
 $\text{knn}(x)$  ~ 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. For adjacency matrices there is a class sgadj. This is mainly unused due to the memory requirements.

Also included are the following functions (3d stuff requires rgl):

| Function name | Description                                |
|---------------|--|
| spin3d        | Little program to animate the rgl-3d image |
| runif3d       | Simple 3d uniform pp generation            |

```

spatcluster      Compute clusters from adjacency matrix
tosgdadj         Convert to adjacency matrix form
tosg             Convert to edge list from sgadj
cut              Cut edges longer than given R>0
prune           Prunes the graph, aimed for MST
sg2dxf           Write graph to a dxf file
shortestPath     Find the shortest edgepath between two points
sg2igraph        Change the sg-object to igraph-object of package igraph

```

### Author(s)

Tuomas Rajala  
 University of Jyvaskyla, Finland  
 tarajala@maths.jyu.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/Delauney handling, see the package 'tripack'

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()

#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,
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()
#spin3d()
```

---

spatgraph-shake     *shake*

---

### Description

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

### Details

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

---

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

---

### Description

Maps between edgelist and the adjacency matrix representation.

### Usage

```
sg2adj(x)
adj2sg(x)
```

### Arguments

x                    sg-object or sgadj-object.

### 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` such that  $\text{distance}(x[i], x[j]) = x[d]$

**Usage**

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

**Arguments**

|                  |  |
|------------------|--|
| <code>x</code>   | sg-object.   |
| <code>way</code> | If 1, use (xy OR yx) rule, if 2 or anything else use (xy AND yx) rule. |
| <code>pp</code>  | Point pattern, for distances.  |
| <code>...</code> | ignored  |

**Details**

Date: 2009-09-14  
License: GPL v2 or later

---

spatgraphs-plot.sg *plot.sg*

---

**Description**

Plot the edges of graph, or color the clusters.

**Arguments**

|                       |  |
|-----------------------|--|
| <code>x</code>        | spatgraph/spatcluster object   |
| <code>pp</code>       | point pattern  |
| <code>directed</code> | Draw arrows with this size. If 0, no arrows.                               |
| <code>...</code>      | color, linesize etc. for corresponding function (lines, points, points3d). |

**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.

**Details**

Date: 2007-10-24  
License: GPL v2 or later

---

spatgraphs-runif3d *runif3d and spin3d*

---

**Description**

Simple simulation of uniform 3d point pattern.  
spin3d simply spins the rgl-scenery.

**Details**

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

---

spatgraphs-sg *sg*

---

**Description**

Edge list-of-lists class for spatgraphs. Methods: print, plot.  
Older version used adjacency matrix, this class is now called sgadj.

tosg and tosgadj convert back and forth.

See also `sg2sym`

### Details

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

---

`spatgraphs-sg2dxf` *sg2dxf*

---

### Description

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

### Arguments

|                   |                   |
|-------------------|-------------------|
| <code>x</code>    | spatgraph object. |
| <code>pp</code>   | point pattern.    |
| <code>file</code> | 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**

Clusterlist-of-lists class for spatgraphs. Methods: print, plot.

**Details**

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

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).

**Arguments**

pp                    Point pattern, as ppp in spatstat.  
 i                     The start node of the path to find.  
 j                     The target node of the path to find.  
 g=NULL              Graph which defines the edges. If NULL computes one for given additional parameters.  
 . . .                 Passed on to spatgraph if g is not given.

**Details**

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

Make sure the graph is symmetric. Value is the sum of Euclidian lengths of the edges in the shortest path. 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](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

---

spatgraphs-spatgraph  
*spatgraph*

---

**Description**

Compute an adjacency list-of-lists for a given 2D- or 3D- point pattern.

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

## Usage

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

## Arguments

|                          |   |
|--------------------------|---|
| <code>pp</code>          | Point pattern with members <code>x,y,n,window</code> . Window must have <code>x-</code> and <code>y-</code> limits according to given <code>x,y</code> . see package <code>spatstat</code> , class <code>ppp</code> . |
| <code>type</code>        | One of the supported graph types, see below.  |
| <code>par</code>         | Parameter(s) for the graph, see below.  |
| <code>preprocessR</code> | If $>0$ first compute geometric graph and then the <code>type</code> graph using the preprocessed edgelist. Useful to narrow the search space for bigger <code>pp</code> 's.  |
| <code>dbg</code>         | Boolean, print additional information of the execution.   |
| <code>doDists</code>     | Boolean, default <code>FALSE</code> . If true, precompute the distance diagonal matrix. Speeds up quite a lot but takes $O(n^2)$ memory!  |
| <code>preDists</code>    | Optional precalculated distance matrix for the points.  |
| <code>toroidal</code>    | Make a toroidal distance calculation. Not useful when visualizing but useful for edge correction in summary calculations.   |

## Details

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

|                  |   |
|------------------|---|
| 'geometric'      | <code>par=numeric&gt;0</code> . Geometric graph, <code>par</code> = connection radius.  |
| 'knn'            | <code>par=integer&gt;0</code> . <code>k</code> -nearest neighbours graph, <code>par</code> = <code>k</code> .                               |
| 'mass_geometric' | Connect two points if $\ x-y\  < m(x)$ .  |
| 'gabriel'        | Gabriel graph. Additional parameter for allowing <code>par=k</code> instead of 0 points in the circle.                                      |
| 'delauney'       | Delauney triangulation. !Only 2D!   |
| 'MST'            | Minimal spanning tree.  |
| 'markcross'      | Connect two points if $\ x-y\  < m(x)+m(y)$ .   |
| 'SIG'            | Spheres of Influence.   |
| 'RST'            | <code>par=c(x0,y0,z0)</code> . Radial spanning tree, <code>par</code> =origin of radiation.   |
| 'RNG'            | Relative neighbourhood graph.   |
| 'CCC'            | <code>par=integer</code> (or string). Class-Cover-Catch, <code>par</code> =target type.   |
| 'STIR'           | <code>par=c(noise,alpha,beta,gamma)</code> . Signal-To-Noise-Ratio, <code>par</code> : background noise, signal attenuation ( $1/r^alpha$ ) |

where

$m(x) \sim$  real valued mark for `x` (size, mass, diameter, transmission power...)

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.

### **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.

# Index

## \*Topic **spatial**

- spatgraph-shake, 4
  - spatgraphs-adj2sg, 4
  - spatgraphs-cut.sg, 4
  - spatgraphs-mailer, 5
  - spatgraphs-other, 5
  - spatgraphs-package, 1
  - spatgraphs-plot.sg, 6
  - spatgraphs-print.sg, 6
  - spatgraphs-runif3d, 7
  - spatgraphs-sg, 7
  - spatgraphs-sg2dxf, 8
  - spatgraphs-sg2igraph, 8
  - spatgraphs-sgc, 9
  - spatgraphs-shortestPath, 9
  - spatgraphs-spatcluster, 10
  - spatgraphs-spatgraph, 10
- adj2sg (*spatgraphs-adj2sg*), 4
- cut.sg (*spatgraphs-cut.sg*), 4
- edgeLengths (*spatgraphs-other*), 5
- igraph2sg (*spatgraphs-sg2igraph*),  
8
- mailer (*spatgraphs-mailer*), 5
- plot.sg (*spatgraphs-plot.sg*), 6
- plot.sgadj (*spatgraphs-plot.sg*), 6
- plot.sgc (*spatgraphs-plot.sg*), 6
- print.sg (*spatgraphs-print.sg*), 6
- print.sgadj  
(*spatgraphs-print.sg*), 6
- print.sgc (*spatgraphs-print.sg*), 6
- prune (*spatgraphs-cut.sg*), 4
- runif3d (*spatgraphs-runif3d*), 7
- sg (*spatgraphs-sg*), 7
- sg.edgelenlengths  
(*spatgraphs-other*), 5
- sg2adj (*spatgraphs-adj2sg*), 4
- sg2dxf (*spatgraphs-sg2dxf*), 8
- sg2igraph (*spatgraphs-sg2igraph*),  
8
- sg2sym (*spatgraphs-other*), 5
- sg\_default\_par  
(*spatgraphs-spatgraph*), 10
- SG\_GRAPH\_PARS  
(*spatgraphs-spatgraph*), 10
- sg\_modify\_pp  
(*spatgraphs-spatgraph*), 10
- SG\_SUPPORTED\_GRAPHS  
(*spatgraphs-spatgraph*), 10
- sg\_verify\_parameters  
(*spatgraphs-spatgraph*), 10
- sgadj (*spatgraphs-sg*), 7
- sgc (*spatgraphs-sgc*), 9
- shake (*spatgraph-shake*), 4
- shortestPath  
(*spatgraphs-shortestPath*),  
9
- spatcluster  
(*spatgraphs-spatcluster*),  
10
- spatgraph (*spatgraphs-spatgraph*),  
10
- spatgraph-shake, 4
- spatgraphs (*spatgraphs-package*), 1
- spatgraphs-adj2sg, 4
- spatgraphs-cut.sg, 4
- spatgraphs-mailer, 5
- spatgraphs-other, 5
- spatgraphs-package, 1
- spatgraphs-plot.sg, 6
- spatgraphs-print.sg, 6
- spatgraphs-runif3d, 7
- spatgraphs-sg, 7

spatgraphs-sg2dxf, 8  
spatgraphs-sg2igraph, 8  
spatgraphs-sgc, 9  
spatgraphs-shortestPath, 9  
spatgraphs-spatcluster, 10  
spatgraphs-spatgraph, 10  
spin3d(*spatgraphs-runif3d*), 7  
summary.sg(*spatgraphs-sg*), 7  
summary.sgc(*spatgraphs-sg*), 7  
took(*spatgraphs-mailer*), 5  
verifyclass(*spatgraphs-sg*), 7