Package 'latticeDensity'

July 2, 2014

Type Package

Title Density estimation and nonparametric regression on irregular regions

Version 1.0.7

Date 2012-01-06

Author Ronald Barry <rpbarry@alaska.edu>

Maintainer Ronald Barry <rpbarry@alaska.edu>

Depends splancs, spdep, spatstat, spam

Description This package contains functions that compute the lattice-based density estimator of Barry and McIntyre, which accounts for point processes in two-dimensional regions with irregular boundaries and holes. The package also implements two-dimensional non-parametric regression for similar regions.

License GPL-2

URL www.r-project.org

Repository CRAN

Date/Publication 2012-01-07 05:44:22

NeedsCompilation no

R topics documented:

`latticeDensity-package`

Creates a 2-dimensional density estimate or nonparametric regression surface.

Description

Lattice Density can take a region bounded externally with a polygon and possibly containing polygonal holes, along with a point pattern in the region, and produces a 2-dimensional density map. Given a set of locations each associated with a response, `latticeDensity` can produce a 2-dimensional nonparametric regression surface.

Details

Package:	latticeDensity
Type:	Package
Version:	1.0.5
Date:	2011-06-07
License:	GPL-2
LazyLoad:	yes

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

out = crossvalDensity(formLatticeOutput,PointPattern=Pointdata, M=0.5,num.steps = 50)

densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=out$k,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

`addObservations` *Input observations for use in the lattice-based density estimator.*

Description

This function takes a `formLatticeOutput` object, which encodes a region possibly with and irregular boundary and holes, and adds point process observations. The observations should be in the form of a matrix or data frame. `addObservations` is used when the aim is to produce a density map from a point process. If, instead, you wish to produce a nonparametric regression surface given responses and their locations, you should use `addQuantVar` instead.

Usage

```
addObservations(formLatticeOutput, observations, will.plot=TRUE)
```

Arguments

`formLatticeOutput`	a formLatticeOutput object. This is returned by the functions formLattice and editLattice.
`observations`	a matrix with two columns or a data frame. Other arguments will be coerced (if possible) with as.matrix.
`will.plot`	logical. If TRUE (default), plots the actual point process along with the nearest nodes, to which they are relocated.

Details

Every node in the `formLatticOutput` object is assigned an initial density that is equal to the fraction of all observations that are nearest to that node. Note that this means observations can be outside the boundary of the region of interest - they will just be associated with the nearest node inside the region. The function returns a vector equal in length to the number of nodes that has the initial density for each node. This vector corresponds to p_0, the inital probability vector as in Barry and McIntyre (2011).

Value

a list with two elements.

`init.prob`	Numerical vector with the initial probability distribution
`which.nodes`	vector of nodes to which observations were assigned

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.01)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

out = crossvalDensity(formLatticeOutput,PointPattern=Pointdata,M=0.5,
num.steps = 50)

densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=out$k,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

addQuantVar *Input response variable for use in the createNparReg function*

Description

This function takes a formLatticeOutput object, which encodes a region possibly with and irregular boundary and holes. This and a matrix of locations where a response variable has been measured, and a vector of the responses, is used to create an initial distribution for use in the non-parametric regression function createNparReg. If, instead, you have a point process and wish to produce a density estimate, you should use the function addObservations.

Usage

```
addQuantVar(formLatticeOutput, Z, locations, will.plot=TRUE)
```

Arguments

formLatticeOutput	a formLatticeOutput object. This is returned by the functions formLattice and editLattice.
Z	a vector of the response variable.
locations	a matrix with two columns or a data frame. Other arguments will be coerced (if possible) with as.matrix.
will.plot	logical. If TRUE (default), plots the actual point process along with the nearest nodes, to which they are relocated.

Value

a list with three elements.

init.quantvar	Numerical vector of initial probability distribution multiplied by the response variable. This is used to compute the numerator of the kernel estimator.
init.prob	Numerical vector with the initial probability distribution based on locations. This is used to compute the denominator of the kernel estimator.
which.nodes	vector of nodes to which observations were assigned

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
data(nparExample)
attach(nparExample)
#
plot(rbind(polygon2,polygon2[1,]),type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
NparRegOut = createNparReg(formLatticeOutput,Z,PointPattern=grid2,k=2)
plot(NparRegOut)
```

---

areaRegion *Computes the area of a region*

Description

This function computes the area of a region by first finding the area of the bounding polygon, then subtracting the area of each hole.

Usage

```
areaRegion(formLatticeOutput)
```

Arguments

`formLatticeOutput`
: a `formLatticeOutput` object. This is returned by the functions `formLattice` and `editLattice`.

Details

Note that this program does not check to see if the holes are non-intersecting or if the holes intersect the polygon.

Value

Numerical area.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
data(areaRegionExample)
attach(areaRegionExample)
#
hole.list = list(hole1,hole2)
nodeFillingOutput = nodeFilling(poly=boundary,node.spacing=0.03,
 hole.list = hole.list)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

areaRegion(formLatticeOutput)
```

---

areaRegionExample *A region with two holes*

Description

Three 2-column matrices. The first is a set of vertices of a boundary polygon encompassing a region. The other two, hole1 and hole2, are holes in the region. This example dataset is used to illustrate the situation where there are holes in the region where a density or nonparametric regression is to be applied. It is used in the example of the function `areaRegion`.

Usage

```
areaRegionExample
```

Format

Three 2-column numerical matrices

---

createDensity *Generates a density using random walks on a lattice.*

Description

Given a lattice and a point pattern of observations, `createDensity` starts random walks at each observation. k steps are taken and the output is a `densityOut` object, which can be used to plot a density estimate. If you wish to perform non-parametric regression, you should use the functions `addQuantVar` and `createNparReg` instead.

Usage

```
createDensity(formLatticeOutput, PointPattern = NULL, M = 0.5, k,
intensity = FALSE, sparse=TRUE, ...)
```

Arguments

formLatticeOutput	a formLatticeOutput object, returned by formLattice or by editLattice.
PointPattern	a matrix of location of the point processes.
M	the probability that the random walk stays in the same location.
k	the smoothing parameter, k is the number of steps in the diffusion.
intensity	a logical. Either an intensity of a density map is produced.
sparse	a logical. If TRUE, function uses sparse matrix compuations.
...	arguments for the contour function.

Details

We start with an initial probability density p_0 where the ith entry in p_0 is the fraction of the point pattern that is nearest to the ith node. This is the empirical density function with no smoothing. If T is the transition matrix, and given a number of steps in the diffusion, $T^k p_0$ is the probability density of the diffusion after k steps. This is the major output of this function, along with information needed to produce a plot, including the polygons for the boundary and holes, and a vector of NS coordinates and EW coordinates used by the contour function. All of the necessary information for plotting is bundled in the object of class densityOutLBDE. Details of this process can be found in Barry and McIntyre (2011).

Value

an object of type densityOut

EW.locs	A vector of the EW coordinates for use in a countour plot
NS.locs	A vector of the NS coordinates for use in a contour plot.
boundaryPoly	The boundary of the region, a two-column numeric matrix
hole.list	A list of polygons, one for each hole in the region
PointPattern	A 2 column matrix of object locations
probs	probability distribution on the lattice after k steps
densityLBDE	A vector that is reformatted by the plot.DensityOut function to make a contour plot of the estimated density.
area	area of the region, accounting for holes in the region

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

out = crossvalDensity(formLatticeOutput,PointPattern=Pointdata,
  M=0.5,num.steps = 50)

densityOut = createDensity(formLatticeOutput,
  PointPattern=Pointdata, k=out$k,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

---

createNparReg *This function performs nonparametric regression.*

Description

This function takes the lattice from formLattice (which fills the region of interest) along with the list of responses and their locations, and creates a prediction surface. The approach is kernel nonparametric regression with the kernels created by a k-step diffusion on the lattice about each location where a response was collected.

Usage

```
createNparReg(formLatticeOutput,Z,PointPattern=NULL,M=0.5,k, ...)
```

Arguments

formLatticeOutput	a formLatticeOutput object, returned by formLattice or by editLattice.
Z	a numerical vector of N responses.
PointPattern	a 2xN matrix of locations where the response variables were collected.
M	probability that the random walk stays in the same location on a single step.
k	the smoothing parameter, k is the number of steps in the diffusion.
...	arguments for the contour function.

Details

We denote by $K_{ik}(s)$ the kernel obtained by assigning the node nearest to the ith response and then running a k-step diffusion on the lattice and evaluating the resulting density at location s. Then the estimator $\hat{f}(s) = (\sum_i K_{ik}(s) * Z_i) / \sum_i K_{ik}(s)$ which is the traditional kernal regression estimator with diffusion kernels. This approach leads to a non-parametric regression that respects the boundaries of the polygonal region. The construction of the kernels is detailed in Barry and McIntyre (2011). Using kernels to perform nonparametric regression is described in many publications, including Wasserman (2006).

Value

An object of type `NparRegOut`

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Larry Wasserman. All of Nonparametric Statistics. Springer Science + Business Media, Inc. N.Y. 2006.

Examples

```
data(nparExample)
attach(nparExample)
plot.new()
#  Simulate a response variable
index1 = (grid2[,2]<0.8)|(grid2[,1]>0.6)
Z = rep(NA,length(grid2[,1]))
n1 = sum(index1)
n2 = sum(!index1)
Z[index1] = 3*grid2[index1,1] + 4 + rnorm(n1,0,sd=0.4)
Z[!index1] = -2*grid2[!index1,1] + 4 + rnorm(n2,0,sd=0.4)
#
coords=rbind(polygon2,polygon2[1,])
plot(coords,type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
NparRegOut = createNparReg(formLatticeOutput,Z,PointPattern=grid2,k=2)
plot(NparRegOut)
```

`crossvalDensity` *UBC crossvalidation for the lattice-based density estimator.*

Description

A function to perform crossvalidation to determine the smoothing parameter for the lattice-based density estimator. It minimizes the UCV criterion.

Usage

```
crossvalDensity(formLatticeOutput, PointPattern, M = 0.5,num.steps = 200, sparse = TRUE)
```

Arguments

`formLatticeOutput`	an object of type formLatticeOutput as returned by functions formLattice and editLattice
`PointPattern`	a matrix or data frame of point process locations. Also, any object that can be coerced to a matrix by the function as.matrix.
`M`	governs how often the random walks stays in the same location.
`num.steps`	the maximum number of steps used in the crossvalidation.
`sparse`	logical. If TRUE (default), functions use sparse matrix computations from the package spam. If FALSE, functions use full matrices.

Details

The function computes the k-step diffusion $p_k = T^k p_0$, then computes the Unbiased CrossValidation (UCV) criterion of Sain, Baggerly and Scott (1994). This function can compute the UCV using either full matrix methods or sparse (default) matrix methods. The latter are almost always much faster, though it is possible that if the number of points in the point pattern is large compared to the number of nodes (an unlikely circumstance) that the full matrix method would be quicker. The sparse matrix approach typically uses less memory. The paper by Barry and McIntyre (2010) shows the approximation to the UCV used in this approach.

Value

`ucv`	a vector of ubc values, one for each step k
`k`	the number of steps that minimizes the ucv

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Crossvalidation of Multivariate Densities. Stephan R. Sain, Keith A. Baggerly, David W. Scott; Journal of the American Statistical Association, Vol. 89 (1994) 807-817

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

out = crossvalDensity(formLatticeOutput,PointPattern=Pointdata, M=0.5,num.steps = 100)

densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=out$k,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

---

crossvalNparReg *Crossvalidation for non-parametric regression.*

Description

Performs least-squares crossvalidation for the lattice-based non-parametric regression estimator.

Usage

```
crossvalNparReg(formLatticeOutput,Z,PointPattern,M=0.5,num.steps = 200,plots=
TRUE)
```

Arguments

formLatticeOutput	an object of type formLatticeOutput as returned by functions formLattice and editLattice
Z	the response variable
PointPattern	a matrix or data frame of point process locations. Also, any object that can be coerced to a matrix by the function as.matrix.
M	probability that the random walk stays in the same location in a single step.
num.steps	the maximum number of steps used in the crossvalidation.
plots	if FALSE, suppresses plots of the nodes and data locations, where each data location is colored along with the closes node to which it is assigned.

Value

a list consiting of

SumSq	A vector of crossvalidated sums of squares for k=1 to num.steps.
k	The number of steps that minimizes the crossvalidated sum of squares.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
data(nparExample)
attach(nparExample)
plot.new()
#  Simulate a response variable
index1 = (grid2[,2]<0.8)|(grid2[,1]>0.6)
Z = rep(NA,length(grid2[,1]))
n1 = sum(index1)
n2 = sum(!index1)
Z[index1] = 3*grid2[index1,1] + 4 + rnorm(n1,0,sd=0.4)
Z[!index1] = -2*grid2[!index1,1] + 4 + rnorm(n2,0,sd=0.4)
#
plot(rbind(polygon2,polygon2[1,]),type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
```

```
plot(formLatticeOutput)
hold = crossvalNparReg(formLatticeOutput,Z,
         PointPattern=grid2,M=0.5,num.steps = 75,plots=FALSE)
NparRegOut = createNparReg(formLatticeOutput,Z,PointPattern=grid2,k=hold$k)
plot(NparRegOut)
```

`deletedResid` *Deleted residuls for non-parametric regression.*

Description

Computes deleted residuals for the lattice-based non-parametric regression estimator.

Usage

```
deletedResid(formLatticeOutput,Z,PointPattern,M=0.5,k)
```

Arguments

`formLatticeOutput`	an object of type formLatticeOutput as returned by functions formLattice and editLattice
`Z`	the response variable
`PointPattern`	a matrix or data frame of point process locations. Also, any object that can be coerced to a matrix by the function as.matrix.
`M`	probability that the random walk stays in the same location in a single step.
`k`	number of steps in the random walk.

Value

a vector of deleted residuals.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
data(nparExample)
attach(nparExample)
plot.new()
#  Simulate a response variable
index1 = (grid2[,2]<0.8)|(grid2[,1]>0.6)
Z = rep(NA,length(grid2[,1]))
n1 = sum(index1)
n2 = sum(!index1)
Z[index1] = 3*grid2[index1,1] + 4 + rnorm(n1,0,sd=0.4)
Z[!index1] = -2*grid2[!index1,1] + 4 + rnorm(n2,0,sd=0.4)
#
plot(rbind(polygon2,polygon2[1,]),type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
hold = crossvalNparReg(formLatticeOutput,Z,
         PointPattern=grid2,M=0.5,num.steps = 75)
deletedResid(formLatticeOutput,Z,
         PointPattern=grid2,M=0.5,k=hold$k)
```

`editLattice` *Add or remove links in the lattice*

Description

`editLattice` is an interactive editor based on the function edit.nb from the package spdep. A `formLatticeOutput` object includes an automatically generated neighborhood structure. Occasionally this will either leave two nodes disconnected that should be connected or vice versa. `editLattice` allows the user to directly edit the plot of the lattice using mouseclicks to add or remove neighbor links between nodes.

Usage

```
editLattice(formLatticeOutput)
```

Arguments

`formLatticeOutput`
: a formLatticeOutput object, returned by `formLattice` or `editLattice`.

Value

a formLatticeOutput object, which contains

EWlocs	EW coordinates for use in contour
NSlocs	NS coordinates for use in contour
nodes	matrix of node locations
poly	matrix of vertices of the boundary polygon
latt	lattice object as generated by dnearneigh of package spdep

Author(s)

Ronald P. Barry

See Also

formLattice

Examples

```
  ## Not run:
plot.new()
data(polygon1)
nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.03)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
formLatticeOutput = editLattice(formLatticeOutput)
#
#  Paste the code above into R, then do the editing before
#  pasting the code below into R.
#
Pointdata = csr(polygon1,20)
densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=150,intensity=FALSE, sparse = TRUE)
plot(densityOut)

## End(Not run)
```

---

formLattice *Builds a neighbor structure on the nodes.*

Description

formLattice connects all nodes into a neighbor lattice by linking any two nodes that are within 1.5*node.spacing. Typically this will result in links in the E, W, N, S, NE, NW, SE, SW directions. The lattice object is created by the function dnearneigh from spdep.

Usage

```
formLattice(nodeFillingOutput)
```

Arguments

nodeFillingOutput	a nodeFillingOutput object, as produced by the function nodeFilling.

Details

When forming the lattice, the function does not check to see if any node is completely isolated from the rest of the nodes, nor does it check to see that paths exist between all pairs of nodes. Thus the lattice might be disconnected. You can still determine a nonparametric density in this case, but you need to think about whether it makes sense to allow disconnected sublattices. If you wish to connect isolated nodes to the lattice, use the editing function editLattice.

Value

formLatticeOutput object

EW.locs	EW coordinates for use by contour
NS.locs	NS coordinates for use by contour
nodes	matrix of node locations
poly	outer boundary
latt	neighbor lattice
hole.poly	list of hole polygons

Author(s)

Ronald P. Barry

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,80)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
```

```
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=20,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

---

homerange *Produces a homerange map.*

Description

homerange produces a map of the homerange, for any given percentage. The homerange contains the smallest number of nodes with total density greater than the percent. This function is illustrated in Barry and McIntyre (2011).

Usage

```
homerange(densityOut, percent = 0.95,output=FALSE)
```

Arguments

densityOut	a densityOut object, produced by createDensity.
percent	the sum of the probabilities of all nodes in the homerange exceeds this value.
output	if TRUE, the function returns a matrix containing, for each node, a location (first two columns) and whether the node is in the homerange.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.015)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

out = crossvalDensity(formLatticeOutput,PointPattern=Pointdata,
    M=0.5,num.steps = 50)

densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=out$k,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

---

makeTmatrix *Create the transition matrix for the diffusion.*

Description

This function generates a transition matrix for the diffusion process on the lattice.

Usage

```
makeTmatrix(formLatticeOutput, M = 0.5, sparse = TRUE)
```

Arguments

formLatticeOutput	a formLatticeOutput object, returned by the functions formLattice or by the function editLattice.
M	a smoothing parameter. It is the probability that the random walk stays on the same node in a single step. Since the number of steps k also determines smoothing, M is usually left at 0.5. Note that values of M = 1 or M = 0 can lead to pathological results. The paper of Barry and McIntyre (2011) shows the exact construction of the transition matrix.

sparse	logical. If TRUE, then uses sparse matrix computations from packages spdep and spam. If FALSE, uses full matrix computations. The use of sparse matrices is almost always more efficient.

Value

an NxN transition matrix, where N is the number of nodes.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
formLatticeOutput = formLattice(nodeFillingOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]

poly.area = areapl(polygon1)

init.prob = addObservations(formLatticeOutput, Pointdata)
T = makeTmatrix(formLatticeOutput, M = 0.5, sparse=TRUE)
p10 = Tkp(T,10,p=init.prob$init.prob)
cbind(init.prob$init.prob,p10)
```

---

nodeFilling *Produces a grid of locations inside the region boundary.*

Description

nodeFilling produces a grid of locations that are the nodes in the diffusion process.

Usage

```
nodeFilling(poly, node.spacing, hole.list)
```

Arguments

poly	a matrix that contains the vertices of the bounding polygon.
node.spacing	the distance between grid locations.
hole.list	optional list of holes to be removed from the region
...	arguments for the plot produced as a side effect.

Details

nodeFilling superimposes a square grid of points over the region, with spacing given by the parameter node.spacing. The points contained in the region are retained. The output, a nodeFillingOutput object, contains the boundaries of the region (and holes), the set of nodes, and EW and NS coordinates necessary for creating a contour plot.

Value

an object of type nodeFillingOutput is produced.

EW.locs	EW coordinates for the contour plot
NS.locs	NS coordinates for the contour plot
nodes	matrix of node locations
poly	matrix of vertices of boundary polygon
node.spacing	verticle and horizontal node spacing
hole.list	list of polygons representing holes in region

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.02)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)

Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)
```

```
densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=20,intensity=FALSE, sparse = TRUE)
plot(densityOut)

homerange(densityOut, percent = 0.95)
```

---

nparExample *Example boundary with a grid of locations*

Description

The first item, polygon2, is 35x2 matrix describing the boundary of a region. The second, grid2, is a set of 59 locations for simulated values of a response variable. The third item, Z, is a vector of responses. This dataset was created to test and illustrate the non-parametric lattice based regression estimator. See the example for function createNparReg.

Usage

```
nparExample
```

Format

Two matrices and a vector. One matrix is 35x2, the other is 59x2.

---

plot.densityOut *Plot the density map*

Description

Plots the boundary, all holes and the locations of all nodes along with the density contour map.

Usage

```
## S3 method for class 'densityOut'
plot(x,show.observations=FALSE,...)
```

Arguments

x	an object of type densityOut returned by createDensity.
show.observations	if TRUE, prints the data in the plot.
...	other arguments to be passed to functions plot, points, lines.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)
#
nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
#
Pointdata = csr(polygon1,100)
Pointdata = Pointdata[Pointdata[,1]<0.5,]
plot(rbind(polygon1,polygon1[1,]),type="l")
points(Pointdata,pch=19)

#
densityOut = createDensity(formLatticeOutput,PointPattern=Pointdata,
  k=60,intensity=FALSE, sparse = TRUE)
plot(densityOut)
#
homerange(densityOut, percent = 0.95)
```

---

plot.formLatticeOutput

Plot the lattice.

Description

This function plots the boundary, holes, nodes and neighbor lattice for the lattice based density or regression estimators. The plot can be examined to determine whether the lattice of connected nodes fills the region. If some nodes are connected when they should not be, or are disconnected when they should be connected, use `editLattice` to add or remove neighbor links.

Usage

```
## S3 method for class 'formLatticeOutput'
plot(x,...)
```

Arguments

x	an object of type formLatticeOutput returned by either formLattice or editLattice.
...	other arguments to be passed to functions plot, points, lines.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.015)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
```

---

plot.nodeFillingOutput

Plot a nodeFillingOutput *object.*

Description

Plots the boundary, all holes and the locations of all nodes. Should be used to decide if the nodes fill the region and are spaced closely enough to give good resolution in the plots. The only reason not to make the nodes too closely spaced is when the computing time or memory becomes too great.

Usage

```
## S3 method for class 'nodeFillingOutput'
plot(x,...)
```

Arguments

x	an object of type nodeFillingOutput returned by either nodeFilling or removeHole.
...	other arguments to be passed to functions plot, points, lines.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.01)
plot(nodeFillingOutput)
```

`plot.NparRegOut` *Plot the non-parametric regression surface.*

Description

Takes as input a NparRegOut object from the function `createNparReg`. This plotting function makes a countour plot of the non-parametric regression prediction surface.

Usage

```
## S3 method for class 'NparRegOut'
plot(x,...)
```

Arguments

`x`	an object of type `NparRegOut` returned by `createNparReg`.
`...`	other arguments to be passed to functions plot, points, lines.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
data(nparExample)
attach(nparExample)
plot.new()
#  Simulate a response variable
index1 = (grid2[,2]<0.8)|(grid2[,1]>0.6)
Z = rep(NA,length(grid2[,1]))
n1 = sum(index1)
n2 = sum(!index1)
Z[index1] = 3*grid2[index1,1] + 4 + rnorm(n1,0,sd=0.4)
Z[!index1] = -2*grid2[!index1,1] + 4 + rnorm(n2,0,sd=0.4)
#
plot(rbind(polygon2,polygon2[1,]),type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#  Following is the generation of the nonparametric
#  regression prediction surface.
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
```

```
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
NparRegOut = createNparReg(formLatticeOutput,Z,PointPattern=grid2,k=2)
plot(NparRegOut)
```

---

polygon1 *Example boundary with causeway*

Description

A 2x19 matrix of vertices for the boundary of a region representing a lake almost divided in half by a causeway. This was used in a simulation in the paper of Barry and McIntyre (2011).

Usage

```
polygon1
```

Format

A 2x19 numerical matrix

Source

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

---

predict.NparRegOut *Predictions at data locations from lattice-based non-parametric regression.*

Description

Takes as input a NparRegOut object from the function createNparReg. A vector of predicted values is produced corresponding to each location in the data.

Usage

```
## S3 method for class 'NparRegOut'
predict(object,new.pred=NULL,...)
```

Arguments

object	an object of type NparRegOut returned by createNparReg.
new.pred	if new.pred is left out, predictions are made at the locations of the point pattern. Otherwise, new.pred is a 2-column matrix of locations where you wish to obtain predictions
...	aditionally arguments affecting the predictions, of which there are none at this time.

Details

If new.pred is not used as an arguments, this function returns a vector of predictions at each node closest to an observations of the original point process. If you wish to make predictions at arbitrary locations, let new.pred be a 2-column matrix of those locations. Note that all predictions are actually at the nearest node to the desired locations. NOTE: Like all functions in this package, new locations are relocated to the nearest node in the region, even if they are outside the boundary. Thus you should ensure that your locations of interest are inside the boundary and that the density of nodes is high enough that the nearest node is close enough to the location you queried.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Examples

```
data(nparExample)
attach(nparExample)
plot.new()
#  Simulate a response variable
index1 = (grid2[,2]<0.8)|(grid2[,1]>0.6)
Z = rep(NA,length(grid2[,1]))
n1 = sum(index1)
n2 = sum(!index1)
Z[index1] = 3*grid2[index1,1] + 4 + rnorm(n1,0,sd=0.4)
Z[!index1] = -2*grid2[!index1,1] + 4 + rnorm(n2,0,sd=0.4)
#
plot(rbind(polygon2,polygon2[1,]),type="l")
points(grid2,pch=19,cex=0.5,xlim=c(-0.1,1))
text(grid2,labels=round(Z,1),pos=4,cex=0.5)
#  Following is the generation of the nonparametric
#  regression prediction surface.
nodeFillingOutput = nodeFilling(poly=polygon2,node.spacing=0.025)
plot(nodeFillingOutput)
formLatticeOutput = formLattice(nodeFillingOutput)
plot(formLatticeOutput)
NparRegOut = createNparReg(formLatticeOutput,Z,PointPattern=grid2,k=2)
plot(NparRegOut)
predict(NparRegOut)
```

removeHole *Removes holes from the region prior to density estimation.*

Description

If a hole in a region is specified as a polygon, the function removeHole removes all nodes in the nodeFillingOutput that are contained in the hole. This function is called by nodeFilling, so it is generally not needed by users.

Usage

```
removeHole(hole.poly, nodeFillingOutput)
```

Arguments

hole.poly	a numerical matrix of vertices of the hole polygon.
nodeFillingOutput	an object of type nodeFillingOutput, returned by codenodeFilling or removeHole.

Value

an object of type nodeFillingOutput.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

Tkp *Compute the vector T^k*p*

Description

T is the transition matrix of the random walk on the lattice. By multiplying by the probability density p at time t, you get the probability density at time t+1. Thus, to get the probability density after k steps, pk, compute $p_k = T^k p_1$. This application of finite Markov processes is described in Barry and McIntyre (2011).

Usage

```
Tkp(T, k, p)
```

Arguments

T	transition matrix returned by makeTmatrix.
k	the number of steps in the diffusion.
p	a numerical vector of length equal to the number of nodes, of initial probabilities.

Author(s)

Ronald P. Barry <rpbarry@alaska.edu>

References

Ronald P. Barry, Julie McIntyre. Estimation animal densities and home range in regions with irregular boundaries and holes: A lattice-based alternative to the kernel density estimator. Ecological Modelling 222 (2011) 1666-1672.

Examples

```
plot.new()
data(polygon1)
require(splancs)
require(spatstat)

nodeFillingOutput = nodeFilling(poly=polygon1,node.spacing=0.015)
formLatticeOutput = formLattice(nodeFillingOutput)

Pointdata = csr(polygon1,80)
Pointdata = Pointdata[Pointdata[,1]<0.5,]


init.prob = addObservations(formLatticeOutput, Pointdata)
T = makeTmatrix(formLatticeOutput, M = 0.5, sparse=TRUE)
p10 = Tkp(T,10,p=init.prob$init.prob)
cbind(init.prob$init.prob,p10)
```

Index