Introduction to Spatstat

1 OVERVIEW

This library provides functions for the statistical analysis of spatial point patterns and binary images* in two dimensions. It supports

- creation, manipulation and plotting of point patterns
- exploratory data analysis using the statistics F, G, J and K
- parametric model-fitting by maximum pseudolikelihood

The window of observation for the point pattern may have arbitrary shape and the points may have marks.

The point process models may be quite general; they may include spatial trend, dependence on covariates, and interpoint interactions (which are not restricted to pairwise interactions).

[* The current version 0.6 only handles spatial point pattern data.]

2 EXAMPLE

The following code will read in a data file containing a point pattern in the unit square, plot it, compute and plot the estimate of the nearest neighbour function G, fit the Strauss point process model by maximum pseudolikelihood, and plot the fitted conditional intensity.

```
xy <- scan("myfile", what=list(x=0,y=0))
pp <- point.pattern(xy$x, xy$y, 0:1, 0:1)
plot(pp)
G <- Gest(pp)
plot(G$r, G$km, type="l")
fit <- mpl(pp, ~1, Strauss(0.07), rbord=0.07)
fit
plot(fit, trend=F)
```

3 FUNCTIONS

Here is a more detailed listing of the functions available.

3.1 CREATION, MANIPULATION AND PLOTTING OF POINT PATTERNS

The library supports a class "point.pattern" representing two dimensional point patterns and a class "owin" representing the window of observation. A window may have arbitrary shape. The points of a point pattern may carry marks.

To create a window:

`make.rectangle`	Defines a rectangle in R^2.
`make.raster`	Defines a raster (a rectangle divided into pixels)
`convex.poly.mask`	Define a convex polygonal window
`make.mask`	Define a window of arbitrary shape. A mask is a raster containing a binary image (i.e. a logical matrix) whose pixel values are T wherever the pixel is inside the window.
`matrix.to.mask`	
`coerce.to.mask`	
`coerce.to.raster`	
`as.owin`	Convert various other kinds of data to an "owin".

To manipulate a window:

`plot.owin`	plot a window. (a method for "plot")
`area.owin`	compute window's area
`diameter`	compute window frame's diameter
`erode.mask`	erode window by a distance r
`erode.owin`	" " " " " "
`eroded.areas`	compute areas of eroded windows
`inside.owin`	determine whether a point is inside a window
`nearest.raster.point`	map continuous coordinates to raster locations
`negate.mask`	invert (inside $\leftrightarrow$ outside)

To create a point pattern:

`point.pattern`	create a point pattern from (x, y) and window
`as.point.pattern`	convert other types of data to a point.pattern
`rpoispp`	generate an inhomogeneous Poisson point pattern
To manipulate a point pattern:	
`plot.point.pattern`	plot a point pattern (a method for "plot")
`"[.point.pattern"`	extract a subset of a point.pattern pp[subset] pp[, subwindow]

3.2 EXPLORATORY DATA ANALYSIS

Functions:

Summary statistics for a point pattern:

`Fest()`	empty space function F
`Gest()`	nearest neighbour distribution function G
`Kest()`	Ripley's K-function
`Jest()`	estimate the J-function $J = (1 - G)/(1 - F)$

Summary statistics for a marked point pattern:

`Gcross,Gdot,Gmulti`	counterparts of Gest
`Jcross,Jdot,Jmulti`	counterparts of Jest
`Kcross,Kcross.multi`	counterparts of Kest

Supporting functions:

`bdist`	Distance from (x, y) to boundary of window
`bdry.dist.image`	Distance from each pixel to boundary of window
`exactdt`	Exact distance transform of a point pattern. A rectangular array of values giving the Euclidean distance from each array location to the nearest point of the point pattern.
`exactPdt`	Exact distance transform of a binary image.

Details:

The library will compute estimates of the summary statistics

$F(r)$, the empty space function

$G(r)$, the nearest neighbour distance distribution function

$K(r)$, the reduced second moment function ("Ripley's K")

$J(r)$, the J function of Van Lieshout and Baddeley (1995)

for a point pattern, and their analogues for marked point patterns.

These estimates can be used for exploratory data analysis and in formal inference about a spatial point pattern.

The point pattern has to be assumed to be "stationary" (statistically homogeneous under translations) in order that the functions F, G, J, K be well-defined and the corresponding estimators approximately unbiased.

The empty space function F of a stationary point process X is the cumulative distribution function of the distance from a fixed point in space to the nearest point of X. The nearest neighbour function G is the c.d.f. of the distance from a point *of the pattern* X to the nearest other point of X. The J function is the ratio $J(r) = (1 - G(r))/(1 - F(r))$. The K function is defined so that $\lambda K(r)$ equals the expected number of additional points of X within a distance r of a point of X, where λ is the intensity (expected number of points per unit area).

In exploratory analyses, the estimates of F, G, J and K are useful statistics. F summarises the sizes of gaps in the pattern; G summarises the clustering of close pairs of points; J is a comparison between these two effects; and K is a second order measure of spatial association.

For inferential purposes, the estimates of F, G, J, K are usually compared to their true values for a completely random (Poisson) point process, which are

$$\begin{aligned} F(r) &= 1 - \exp(-\lambda \pi r^2) \\ G(r) &= 1 - \exp(-\lambda \pi r^2) \\ J(r) &= 1 \\ K(r) &= \pi r^2 \end{aligned}$$

where again λ is the intensity. Deviations between the empirical and theoretical curves may suggest spatial clustering or spatial regularity.

3.3 MODEL FITTING

FUNCTIONS:

To fit a point process model:

`mpl`	Fit a point process model to a two-dimensional point pattern
`plot.ppm`	Plot the fitted model
`predict.ppm`	Compute the spatial trend and conditional intensity of the fitted point process model
`quadscheme`	generate a Berman-Turner quadrature scheme for use by `mpl`

To specify a point process model:

`Poisson()`	the Poisson point process
`Strauss()`	the Strauss process
`StraussHard()`	the Strauss/hard core point process
`Softcore()`	pairwise interaction, soft core potential
`PairPiece()`	pairwise interaction, piecewise constant
`Pairwise()`	pairwise interaction, user-supplied potential
`Geyer()`	Geyer's saturation process
`Saturated()`	Saturated pair model, user-supplied potential
`OrdThresh()`	Ord process, threshold potential
`Ord()`	Ord model, user-supplied potential

DETAILS:

The function `mpl()` estimates the "exponential family" type parameters of a model for a point process (values for parameters not of this type must be specified a priori, as arguments to the interaction function). The estimation procedure is based on the maximum pseudolikelihood technique as discussed in Baddeley & Turner (2000).

The usage of `mpl()` is analogous to that of `glm()`. To fit a point process model, we specify its systematic component ("spatial trend" or spatial covariate effects) by an Splus formula, and its random component (the interpoint interaction) by a an "interaction family". For example the stationary Strauss process is fitted by

```
mpl(pp, ~1, Strauss(r=0.07), ....)
```

while the non-stationary Strauss process with spatial trend of the form $b(x, y) = \exp(a + bx)$ is fitted by

```
mpl(pp, ~x, Strauss(r=0.07), ....)
```

The formula `~x~` can be replaced by any Splus formula (with empty left hand side) in terms of the Cartesian coordinates x, y or in terms of some spatial covariates which you must then supply.

The expression `Strauss(..)` can be replaced by any of several interaction structures such as

`Poisson()` .... Poisson process
`StraussHard()` Strauss process with a hard core
`Softcore()` ... Pairwise interaction, soft core potential
`PairPiece()` .. Pairwise interaction, piecewise constant potential
`Geyer()` ..... Geyer's saturation process
`OrdThresh()` .. Ord process with threshold potential

The following will accept "user-defined potentials" in the form of an arbitrary Splus function. They effectively allow arbitrary point process models of the given type.

`Pairwise()` . Pairwise interaction, user-supplied potential
`Ord()` ..... Ord model, user-supplied potential
`Saturated()` Saturated pairwise model, user-supplied potential

The brave user may also generate completely new point process models using the foregoing as templates.

The function `mpl()` prefers to be provided with a "quadrature scheme" as its first argument (although it will make do with a point pattern and calculate a default quadrature scheme, grumbling all the while). Thus one might proceed as follows:

```
Q.simdat <- quadscheme(simdat,gridcentres(simdat,50,50),
                                  nx=40,ny=40)
fit.simdat <- mpl(Q.simdat,~poly(x,y,3),Softcore(0.5),
                           correction="periodic")
```

The resulting object fit.simdat is of class "ppm". A method for `predict()` (`predict.ppm`) to calculate values of the conditional intensity function corresponding to a fitted model is provided.

Plotting methods `plot.point.process` and `plot.ppm` are provided. One can say, e.g.

```
plot(simdat)
```

to obtain a scatterplot of the point pattern, and

```
plot(fit.simdat)
```

to obtain plots of the fitted trend and the conditional intensity function surfaces.

References

[1] A. Baddeley and R. Turner. Practical maximum pseudolikelihood for spatial point patterns. *Australian and New Zealand Journal of Statistics* **42** (2000) 283–322.

[2] A.J. Baddeley and R.D. Gill. Kaplan-Meier estimators for interpoint distance distributions of spatial point processes. *Annals of Statistics* **25** (1997) 263–292.

[3] M.B. Hansen, R.D. Gill and A.J. Baddeley. Kaplan-Meier type estimators for linear contact distributions. *Scandinavian Journal of Statistics* **23** (1996) 129–155.

[4] M.B. Hansen, A.J. Baddeley and R.D. Gill. First contact distributions for spatial patterns: regularity and estimation. *Advances in Applied Probability (SGSA)* **31** (1999) 15–33.

[5] M.N.M. van Lieshout and A.J. Baddeley. A non-parametric measure of spatial interaction in point patterns. *Statistica Neerlandica* **50** (1996) 344–361.

[6] M.N.M. van Lieshout and A.J. Baddeley. Indices of dependence between types in multivariate point patterns. *Scandinavian Journal of Statistics* **26** (1999) 511–532.