Spatial Statistics

Adrian Baddeley
www.maths.uwa.edu.au/~adrian
CSIRO and University of Western Australia

ASC Workshop
Computing with R
2008
Types of spatial data
Three basic types of spatial data:
Three basic types of spatial data:

- geostatistical
Three basic types of spatial data:

- geostatistical
- regional
Three basic types of spatial data:

- geostatistical
- regional
- point pattern
**Geostatistical Data:**
The quantity of interest has a value at any location, ...
... but we only measure the quantity at certain sites. These values are our data.
REGIONAL DATA:
The quantity of interest is only defined for regions. It is measured/reported for certain *fixed* regions.
**Point Pattern Data:**
The main interest is in the *locations* of all occurrences of some event (e.g. tree deaths, meteorite impacts, robberies). Exact locations are recorded.
Points may also carry data (e.g. tree heights, meteorite composition)
Point pattern or geostatistical data?

POINT PATTERN OR GEOSTATISTICAL DATA?
Response variable: the quantity that we want to “predict” or “explain”

Explanatory variable: quantity that can be used to “predict” or “explain” the response.
Geostatistics treats the spatial locations as explanatory variables and the values attached to them as response variables. Spatial point pattern statistics treats the spatial locations, and the values attached to them, as the response.
“Temperature is increasing as we move from South to North” — **geostatistics**

“Trees become less abundant as we move from South to North” — **point pattern statistics**
Software Overview
For information on spatial statistics software:
For information on spatial statistics software:

■ go to cran.r-project.org
For information on spatial statistics software:

- go to cran.r-project.org
- find Task Views
For information on spatial statistics software:

- go to cran.r-project.org
- find Task Views --- Spatial
GIS = Geographical Information System
GIS = Geographical Information System

ArcInfo
GIS = Geographical Information System

*ArcInfo* proprietary [esri.com](http://esri.com)
GIS = Geographical Information System

<table>
<thead>
<tr>
<th>Software</th>
<th>Nature</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArcInfo</td>
<td>proprietary</td>
<td>esri.com</td>
</tr>
<tr>
<td>GRASS</td>
<td>open source</td>
<td>grass.osgeo.org</td>
</tr>
</tbody>
</table>
GRASS: Image data
GRASS: Vector data
GRASS: Multiple data layers

- Elevation
- Slope
- Landcover
- Training Areas
GRASS: Mixed layers
GRASS: Data integration
GRASS: I mean really integrated
GRASS: did I mention data integration
GRASS: unbelievably well integrated
GRASS: runs on anything
Recommendations:

---

Spatial Statistics

ASC Workshop 2008 – 30
Recommendations:
For visualisation of spatial data, especially for presentation graphics, use a GIS.
Recommendations:

For visualisation of spatial data, especially for presentation graphics, use a GIS.

For statistical analysis of spatial data, use R.
Recommendations:

For visualisation of spatial data, especially for presentation graphics, use a GIS.

For statistical analysis of spatial data, use R.

Establish two-way communication between GIS and R, either through a direct software interface, or by reading/writing files in mutually acceptable format.
Putting the pieces together
Interface between R and GIS (online or offline)
Direct interfaces between R and GIS:

- spgrass6: interface to GRASS 6
- RArcInfo: interface to ArcInfo
Direct interfaces between R and GIS:

- **spgrass6** interface to GRASS 6
- **RArcInfo** interface to ArcInfo

Start R and GRASS independently; then start `library(spgrass6)` to establish communication.
## Dominant formats for data files:

<table>
<thead>
<tr>
<th>Format</th>
<th>Software</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESRI “shapefiles”</td>
<td>ArcInfo software</td>
<td>esri.com</td>
</tr>
<tr>
<td>NetCDF</td>
<td>Unidata GIS standard</td>
<td>unidata.ucar.edu</td>
</tr>
</tbody>
</table>
Dominant formats for data files:

- ESRI “shapefiles”  ArcInfo software  esri.com
- NetCDF  Unidata GIS standard  unidata.ucar.edu

Libraries for reading/writing formats, etc:

- GDAL  geospatial data  gdal.org
- PROJ.4  map projections  remotesensing.org
Dominant formats for data files:
- ESRI "shapefiles"  ArcInfo software  esri.com
- NetCDF  Unidata GIS standard  unidata.ucar.edu

Libraries for reading/writing formats, etc:
- GDAL  geospatial data  gdal.org
- PROJ.4  map projections  remotesensing.org

R packages handling GIS data files:
- rgdal  shapefiles, GDAL, PROJ.4
- maps + mapproj  map databases
Support for spatial data: data structures, classes, methods
### R packages supporting spatial data classes:

<table>
<thead>
<tr>
<th>Package</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>sp</td>
<td>generic</td>
</tr>
<tr>
<td>maps</td>
<td>polygon maps</td>
</tr>
<tr>
<td>spatstat</td>
<td>point patterns</td>
</tr>
</tbody>
</table>
Putting the pieces together

Capabilities for statistical analysis
Multiple packages for different analyses
### R packages for geostatistical data

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gstat</td>
<td>classical geostatistics</td>
</tr>
<tr>
<td>geoR</td>
<td>model-based geostatistics</td>
</tr>
<tr>
<td>RandomFields</td>
<td>stochastic processes</td>
</tr>
<tr>
<td>akima</td>
<td>interpolation</td>
</tr>
</tbody>
</table>
**R packages for geostatistical data**

- gstat: classical geostatistics
- geoR: model-based geostatistics
- RandomFields: stochastic processes
- akima: interpolation

**R packages for regional data**

- spdep: spatial dependence
- spgwr: geographically weighted regression
**R packages for geostatistical data**

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gstat</td>
<td>classical geostatistics</td>
</tr>
<tr>
<td>geoR</td>
<td>model-based geostatistics</td>
</tr>
<tr>
<td>RandomFields</td>
<td>stochastic processes</td>
</tr>
<tr>
<td>akima</td>
<td>interpolation</td>
</tr>
</tbody>
</table>

**R packages for regional data**

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spdep</td>
<td>spatial dependence</td>
</tr>
<tr>
<td>spgwr</td>
<td>geographically weighted regression</td>
</tr>
</tbody>
</table>

**R packages for point patterns**

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spatstat</td>
<td>parametric modelling, diagnostics</td>
</tr>
<tr>
<td>splancs</td>
<td>nonparametric, space-time</td>
</tr>
</tbody>
</table>
Geostatistical data
The R package \texttt{gstat} does classical geostatistics: kriging, variograms etc.
> library(gstat)
Loading required package: sp
> data(meuse)
> class(meuse)
[1] "data.frame"
> names(meuse)
[1] "x"    "y"    "cadmium" "copper" "lead"   "zinc"   "elev"
[8] "dist"  "om"   "ffreq"   "soil"  "lime"   "landuse" "dist.m"
Convert raw data to spatial class

```r
> coordinates(meuse) = ~x+y
> class(meuse)
[1] "SpatialPointsDataFrame"
attr(,"package")
[1] "sp"
```
> bubble(meuse, "zinc",
    main="Zinc concentration (ppm)")
> data(meuse.grid)
> coordinates(meuse.grid) = ~x+y
> gridded(meuse.grid) = TRUE
> class(meuse.grid)
[1] "SpatialPixelsDataFrame"
attr(,"package")
[1] "sp"
> image(meuse.grid["dist"])
> title("distance to river")

distance to river
Naive Interpolation

```r
> zinc.idw = krige(zinc~1, meuse, meuse.grid)
[inverse distance weighted interpolation]
> class(zinc.idw)
[1] "SpatialPixelsDataFrame"
attr(,"package")
[1] "sp"
> spplot(zinc.idw["var1.pred"],
    main = "Inverse distance weighted interpolations")
```

zinc inverse distance weighted interpolations
> plot(zinc ~ dist, meuse)
> plot(log(zinc) ~ sqrt(dist), meuse)
> abline(lm(log(zinc) ~ sqrt(dist), meuse))
Variogram assuming constant mean:

```r
> lzn.vgm = variogram(log(zinc)~1, meuse)
> head(lzn.vgm)
   np   dist  gamma  dir.hor  dir.ver   id
 1  57  79.29244 0.1234479       0       0 var1
 2 299 163.97367 0.2162185       0       0 var1
 3 419 267.36483 0.3027859       0       0 var1
> lzn.fit = fit.variogram(lzn.vgm, model = vgm(1, "Sph", 900, 1))
> lzn.fit
   model   psill    range
1    Nug 0.05066243 0.00002
2    Sph 0.59060780 897.0209
> plot(lzn.vgm, lzn.fit)
```
Variogram of residuals from a fitted spatial trend:

```r
> lznr.vgm = variogram(log(zinc)~sqrt(dist), meuse)
> lznr.fit = fit.variogram(lznr.vgm, model = vgm(1, "Exp", 300, 1))
> lznr.fit

   model   psill  range
1  Nug 0.05712231 0.0000
2  Exp 0.17641559 340.3201
```

```r
> plot(lznr.vgm, lznr.fit)
```

![Variogram plot](image)
lzn.kriged = krige(log(zinc)~1, meuse, meuse.grid, model = lzn.fit)
spplot(lzn.kriged["var1.pred"])

Spatial Statistics
lzn.condsim = kriged(log(zinc)~1, meuse, meuse.grid, model = lzn.fit,
     nmax = 30, nsim = 4)
spplot(lzn.condsim, main = "four conditional simulations")
Regional data
The R package `spdep` analyses regional data using neighbourhood dependence statistics.
> library(spdep)
Loading required package: sp
Loading required package: tripack
Loading required package: maptools
Loading required package: foreign
Loading required package: SparseM
Loading required package: boot
> data(nc.sids)
> plot(sidspolys, forcefill=FALSE)
```
pmap <- probmap(nc.sids$SID74, nc.sids$BIR74)
brks <- c(0, 0.001, 0.01, 0.025, 0.05, 0.95, 0.975, 0.99, 0.999, 1)
cols <- rainbow(length(brks))
plot(sidspolys, col=cols[findInterval(pmap$pmap, brks)], forcefill=FALSE)
```
Define which regions are immediate neighbours according to some criterion.

```r
coords <- nc.sids[, c("east", "north")]
gg <- gabrielneigh(coords)
nb <- graph2nb(gg)
```
An index of spatial autocorrelation:

\[
I = \frac{n \sum_i \sum_j w_{ij} (y_i - \bar{y})(y_j - \bar{y})}{(\sum_i \sum_j w_{ij})(\sum_i (y_i - \bar{y})^2)}
\]

where \(w_{ij} = 1\) if sites \(i\) and \(j\) are neighbours, and \(0\) otherwise.
Convert neighbourhood relations to weights $w_{ij}$ between each pair of regions $i, j$.

```r
lw <- nb2listw(nb)
```

(Non-binary weights are possible too.)
> rates <- with(nc.sids, SID74/BIR74)
> moran.test(rates, listw=lw)

Moran’s I test under randomisation

data:  rates
weights:  lw

Moran I statistic standard deviate = 4.1051, p-value = 2.021e-05
alternative hypothesis: greater
sample estimates:
  Moran I statistic   Expectation    Variance
    0.222612195   -0.010101010   0.003213686
> lmr <- lm(rates ~ 1, data=nc.sids, weights=BIR74)
> res <- sp.correlogram(nb, residuals(lmr), order=5, method="I")
> plot(res)
Spatial point patterns
The R package **spatstat** supports statistical analysis for spatial point patterns.
A **point pattern** dataset gives the locations of objects/events occurring in a study region.

The points could represent trees, animal nests, earthquake epicentres, petty crimes, domiciles of new cases of influenza, galaxies, etc.
The points may have extra information called **marks** attached to them. The mark represents an “attribute” of the point.
The points may have extra information called **marks** attached to them. The mark represents an “attribute” of the point.

The mark variable could be *categorical*, e.g. species or disease status:
The mark variable could be *continuous*, e.g. tree diameter:
Our dataset may also include **covariates** — any data that we treat as explanatory, rather than as part of the ‘response’.
Covariate data may be a *spatial function* $Z(u)$ defined at all spatial locations $u$, e.g. altitude, soil pH, displayed as a pixel image or a contour plot:
Covariate data may be another *spatial pattern* such as another point pattern, or a line segment pattern, e.g. a map of geological faults:
Intensity
‘Intensity’ is the average density of points (expected number of points per unit area). Intensity may be constant (‘uniform’) or may vary from location to location (‘non-uniform’ or ‘inhomogeneous’).
> data(swedishpines)
> P <- swedishpines
> plot(P)
Divide study region into rectangles (‘quadrats’) of equal size, and count points in each rectangle.

```r
Q <- quadratcount(P, nx=3, ny=3)
Q
plot(Q, add=TRUE)
```
If the points have uniform intensity, and are completely random, then the quadrat counts should be Poisson random numbers with constant mean.
If the points have uniform intensity, and are completely random, then the quadrat counts should be Poisson random numbers with constant mean. Use the $\chi^2$ goodness-of-fit test statistic

$$X^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}}$$
If the points have uniform intensity, and are completely random, then the quadrat counts should be Poisson random numbers with constant mean. Use the $\chi^2$ goodness-of-fit test statistic

$$X^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}}$$

> quadrat.test(P, nx=3, ny=3)

Chi-squared test of CSR using quadrat counts

data:  P
X-squared = 4.6761, df = 8, p-value = 0.7916
\( \chi^2 \) test of uniformity

```r
> QT <- quadrat.test(P, nx=3, ny=3)
> plot(P)
> plot(QT, add=TRUE)
```

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7.9</td>
<td>+ 0.04</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>6</td>
<td>7.9</td>
<td>+ -0.67</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7+</td>
<td>7.9</td>
<td>+ 1.1</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>8+</td>
<td>7.9</td>
<td>+ 0.04</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>11</td>
<td>7.9</td>
<td>+ 1.1</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>9</td>
<td>7.9</td>
<td>+ + +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7.9</td>
<td>7.9</td>
<td>+ +</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Spatial Statistics
Kernel smoothed intensity

\[ \tilde{\lambda}(u) = \sum_{i=1}^{n} \kappa(u - x_i) \]

where \( \kappa(u) \) is the kernel function and \( x_1, \ldots, x_n \) are the data points.
Kernel smoothed intensity

\[ \tilde{\lambda}(u) = \sum_{i=1}^{n} \kappa(u - x_i) \]

where \( \kappa(u) \) is the kernel function and \( x_1, \ldots, x_n \) are the data points.

1. replace each data point by a square of chocolate
Kernel smoothed intensity

\[ \tilde{\lambda}(u) = \sum_{i=1}^{n} \kappa(u - x_i) \]

where \( \kappa(u) \) is the kernel function and \( x_1, \ldots, x_n \) are the data points.

1. replace each data point by a square of chocolate
2. melt chocolate with hair dryer
Kernel smoothed intensity

\[ \tilde{\lambda}(u) = \sum_{i=1}^{n} \kappa(u - x_i) \]

where \( \kappa(u) \) is the kernel function and \( x_1, \ldots, x_n \) are the data points.

1. replace each data point by a square of chocolate
2. melt chocolate with hair dryer
3. resulting landscape is a kernel smoothed estimate of intensity function
Kernel smoothing

den <- density(P, sigma=15)
plot(den)
plot(P, add=TRUE)
A more searching analysis involves fitting *models* that describe how the point pattern intensity $\lambda(u)$ depends on spatial location $u$ or on spatial covariates $Z(u)$. 
A more searching analysis involves fitting *models* that describe how the point pattern intensity $\lambda(u)$ depends on spatial location $u$ or on spatial covariates $Z(u)$.

Intensity is modelled using a “log link”. 
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTENSITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(P, ~1)</td>
<td>$\log \lambda(u) = \beta_0$</td>
</tr>
<tr>
<td>COMMAND</td>
<td>INTENSITY</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>ppm((P, \sim 1))</td>
<td>(\log \lambda(u) = \beta_0)</td>
</tr>
</tbody>
</table>

\(\beta_0, \beta_1, \ldots\) denote parameters to be estimated.
**Command** | **Intensity**
---|---
`ppm(P, \sim 1)` | \( \log \lambda(u) = \beta_0 \)
`ppm(P, \sim x)` | \( \log \lambda((x, y)) = \beta_0 + \beta_1 x \)

\( \beta_0, \beta_1, \ldots \) denote parameters to be estimated.
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTENSITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(P, ~1)</td>
<td>$\log \lambda(u) = \beta_0$</td>
</tr>
<tr>
<td>ppm(P, ~x)</td>
<td>$\log \lambda((x, y)) = \beta_0 + \beta_1 x$</td>
</tr>
<tr>
<td>ppm(P, ~x + y)</td>
<td>$\log \lambda((x, y)) = \beta_0 + \beta_1 x + \beta_2 y$</td>
</tr>
</tbody>
</table>

\(\beta_0, \beta_1, \ldots\) denote parameters to be estimated.
Swedish Pines data

> ppm(P, ~1)
Stationary Poisson process
Uniform intensity: 0.007
> ppm(P, ~x+y)
Nonstationary Poisson process
Trend formula: ~x + y
Fitted coefficients for trend formula:
  (Intercept)    x    y
-5.1237   0.00461  -0.00025
<table>
<thead>
<tr>
<th>Command</th>
<th>Intensity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(P, ~polynom(x,y,3))</td>
<td>3rd order polynomial in $x$ and $y$</td>
</tr>
</tbody>
</table>
Modelling intensity

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTENSITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(P, ~polynom(x,y,3))</td>
<td>3rd order polynomial in $x$ and $y$</td>
</tr>
<tr>
<td>ppm(P, ~I(y &gt; 18))</td>
<td>different constants above and below the line $y = 18$</td>
</tr>
</tbody>
</table>
fit <- ppm(P, ~x+y)
lam <- predict(fit)
plot(lam)

The `predict` method computes fitted values of intensity function $\lambda(u)$ at a grid of locations.
fit0 <- ppm(P, ~1)
fit1 <- ppm(P, ~polynom(x,y,2))
anova(fit0, fit1, test="Chi")
Likelihood ratio test

```r
fit0 <- ppm(P, ~1)
fit1 <- ppm(P, ~polynom(x,y,2))
anova(fit0, fit1, test="Chi")
```

Analysis of Deviance Table

Model 1: .mpl.Y ~ 1  
Model 2: .mpl.Y ~ polynom(x, y, 5)

| Resid. Df | Resid. Dev | Df | Deviance | P(>|Chi|) |
|-----------|------------|----|----------|---------|
| 1         | 699        | 408.10 |          |         |
| 2         | 694        | 400.62 | 5        | 7.48    | 0.19    |
fit0 <- ppm(P, ~1)
fitted0 <- fitted(fit0)

fit1 <- ppm(P, ~polynom(x,y,2))

anova(fit0, fit1, test="Chi")

Analysis of Deviance Table

Model 1: .mpl.Y ~ 1

Model 2: .mpl.Y ~ polynom(x, y, 5)

| Resid. Df | Resid. Dev | Df | Deviance | P(>|Chi|) |
|-----------|------------|----|----------|----------|
| 1         | 699        |    | 408.10   |          |
| 2         | 694        | 5  | 400.62   | 7.48     | 0.19     |

The p-value 0.19 exceeds 0.05 so the log-quadratic spatial trend is not significant.
diagnose.ppm(fit0, which="smooth")
Spatial covariates
A *spatial covariate* is a function $Z(u)$ of spatial location.
A *spatial covariate* is a function $Z(u)$ of spatial location.

- geographical coordinates
A *spatial covariate* is a function $Z(u)$ of spatial location.

- geographical coordinates
- terrain altitude
A spatial covariate is a function $Z(u)$ of spatial location.

- geographical coordinates
- terrain altitude
- soil pH
A *spatial covariate* is a function $Z(u)$ of spatial location.

- geographical coordinates
- terrain altitude
- soil pH
- distance from location $u$ to another feature
A **spatial covariate** is a function $Z(u)$ of spatial location.

- geographical coordinates
- terrain altitude
- soil pH
- distance from location $u$ to another feature
Covariate data may be another *spatial pattern* such as another point pattern, or a line segment pattern:
For a point pattern dataset with covariate data, we typically

- investigate whether the intensity depends on the covariates
- allow for covariate effects on intensity before studying dependence between points
A intensive mineralogical survey yields a map of copper deposits (essentially pointlike at this scale) and geological faults (straight lines). The faults can easily be observed from satellites, but the copper deposits are hard to find.

*Main question:* whether the faults are ‘predictive’ for copper deposits (e.g. copper less/more likely to be found near faults).
data(copper)
P <- copper$SouthPoints
Y <- copper$SouthLines
plot(P)
plot(Y, add=TRUE)
For analysis, we need a value \( Z(u) \) defined at each location \( u \).
For analysis, we need a value $Z(u)$ defined at each location $u$.
Example: $Z(u) =$ distance from $u$ to nearest line.
For analysis, we need a value $Z(u)$ defined at each location $u$.
Example: $Z(u) = \text{distance from } u \text{ to nearest line.}$

```
D <- distmap(Y)
plot(D)
```
We want to determine whether intensity depends on a spatial covariate $Z$. 
We want to determine whether intensity depends on a spatial covariate $Z$.
Plot $C'(z)$ against $z$, where $C'(z) =$ fraction of data points $x_i$ for which $Z(x_i) \leq z$. 
We want to determine whether intensity depends on a spatial covariate $Z$. 
Plot $C'(z)$ against $z$, where $C'(z) = \text{fraction of data points } x_i \text{ for which } Z(x_i) \leq z$. 
Also plot $C_0(z)$ against $z$, where $C_0(z) = \text{fraction of area of study region where } Z(u) \leq z$. 
We want to determine whether intensity depends on a spatial covariate $Z$.

Plot $C(z)$ against $z$, where $C(z) =$ fraction of data points $x_i$ for which $Z(x_i) \leq z$.

Also plot $C_0(z)$ against $z$, where $C_0(z) =$ fraction of area of study region where $Z(u) \leq z$.

lurking(ppm(P), Z)
We want to determine whether intensity depends on a spatial covariate $Z$.
Plot $C'(z)$ against $z$, where $C'(z) = \text{fraction of data points } x_i \text{ for which } Z(x_i) \leq z$.
Also plot $C_0(z)$ against $z$, where $C_0(z) = \text{fraction of area of study region where } Z(u) \leq z$.

\text{lurking(ppm(P), Z)}
Kolmogorov-Smirnov test

Formal test of agreement between $C(z)$ and $C_0(z)$. 
Formal test of agreement between $C(z)$ and $C_0(z)$.

> kstest(P, Z)

Spatial Kolmogorov-Smirnov test of CSR

data: covariate 'Z' evaluated at points of 'P'
   and transformed to uniform distribution under CSR
D = 0.1163, p-value = 0.3939
alternative hypothesis: two-sided
D <- distmap(Y)
ppm(P, ~Z, covariates=list(Z=D))

Fits the model

$$\log \lambda(u) = \beta_0 + \beta_1 Z(u)$$

where $Z(u)$ is the distance from $u$ to the nearest line segment.
D <- distmap(Y)
ppm(P, ~polynom(Z,5), covariates=list(Z=D))
fits a model in which $\log \lambda(u)$ is a 5th order polynomial function of $Z(u)$. 
fit <- ppm(P, ~polynom(Z,5), covariates=list(Z=D))
plot(predict(fit))
Dr <- summary(D)$range
Dvalues <- seq(Dr[1], Dr[2], length=100)
fakeZ <- data.frame(Z=Dvalues)
fakexy <- data.frame(x=rep(0,100), y=rep(0,100))
lambda <- predict(fit, locations=fakexy, covariates=fakeZ)
plot(Dvalues, lambda, type="l")

plots fitted curve of $\lambda$ against $Z$. 
Copper data

Distance to nearest line vs. intensity.
fit0 <- ppm(P, ~1)
fit1 <- ppm(P, ~polynom(Z,5), covariates=list(Z=D))
anova(fit0, fit1, test="Chi")
fit0 <- ppm(P, ~1)
fit1 <- ppm(P, ~polynom(Z, 5), covariates=list(Z=D))
anova(fit0, fit1, test="Chi")

Analysis of Deviance Table

Model 1: mpl.Y ~ 1
Model 2: mpl.Y ~ polynom(Z, 5)

| Resid. Df | Resid. Dev | Df | Deviance | P(>|Chi|) |
|-----------|------------|----|----------|----------|
| 1         | 682        | 5  | 372.32   |          |
| 2         | 677        | 5  | 370.04   | 2.28     | 0.81     |
fit0 <- ppm(P, ~1)
fit1 <- ppm(P, ~polynom(Z,5), covariates=list(Z=D))
anova(fit0, fit1, test="Chi")

Analysis of Deviance Table

Model 1: mpl.Y ~ 1
Model 2: mpl.Y ~ polynom(Z, 5)

| Resid. Df | Resid. Dev | Df | Deviance | P(>|Chi|) |
|-----------|------------|----|----------|----------|
| 1         | 682        | 677| 372.32   |          |
| 2         | 677        |    | 370.04   | 5        | 2.28     | 0.81     |

The \( p \)-value 0.81 exceeds 0.05 so the 5th order polynomial is not significant.
‘Interpoint interaction’ is stochastic dependence between the points in a point pattern. Usually we expect dependence to be strongest between points that are close to one another.
Example: spacing between points in Swedish Pines data
nearest neighbour distance = distance from a given point to the nearest other point
Summary approach:
Summary approach:

1. calculate average nearest-neighbour distance
Summary approach:

1. calculate average nearest-neighbour distance
2. divide by the value expected for a completely random pattern.

Clark & Evans (1954)
Summary approach:

1. calculate average nearest-neighbour distance
2. divide by the value expected for a completely random pattern.

Clark & Evans (1954)

```r
> mean(nndist(swedishpines))
[1] 7.90754
> clarkevans(swedishpines)
   naive Donnelly     cdf
1.360082 1.291069 1.322862
```
Summary approach:

1. calculate average nearest-neighbour distance
2. divide by the value expected for a completely random pattern.

Clark & Evans (1954)

```r
> mean(nndist(swedishpines))
[1] 7.90754
> clarkevans(swedishpines)
   naive Donnelly     cdf
1.360082 1.291069 1.322862
```

Value greater than 1 suggests a regular pattern.
Exploratory approach:
**Exploratory approach:**

- plot NND for each point

```r
P <- swedishpines
marks(P) <- nndist(P)
plot(P, markscale=0.5)
```
Exploratory approach:

- plot NND for each point
**Exploratory approach:**

- plot NND for each point
- look at empirical distribution of NND’s

```r
plot(Gest(swedishpines))
```
Modelling approach:
Modelling approach:

- Fit a stochastic model to the point pattern, with likelihood based on the NND’s.
Modelling approach:

- Fit a stochastic model to the point pattern, with likelihood based on the NND's.

```r
> ppm(P, ~1, Geyer(4,1))
```

Stationary Geyer saturation process
First order term:
  beta
0.00971209
Fitted interaction parameter gamma: 0.6335
Locations of 65 saplings of Japanese pine in a $5.7 \times 5.7$ metre square sampling region in a natural stand.

```r
data(japanesepines)
J <- japanesepines
plot(J)
```
fit <- ppm(J, ~polynom(x,y,3))
plot(predict(fit))
plot(J, add=TRUE)
If the intensity function \( \lambda(u) \) is known, or estimated from data, then some statistics can be adjusted by counting each data point \( x_i \) with a weight \( w_i = 1/\lambda(x_i) \).
Inhomogeneous $K$-function

```r
lam <- predict(fit)
plot(Kinhom(J, lam))
```
A point process model can also be defined through its *conditional intensity* $\lambda(u \mid x)$. This is essentially the conditional probability of finding a point of the process at the location $u$, given complete information about the rest of the process $x$. 
Strauss process

$\text{Strauss}(\gamma = 0.2)$

$\text{Strauss}(\gamma = 0.7)$
The command \texttt{ppm} will also fit Gibbs models, using the technique of ‘maximum pseudolikelihood’.
The command `ppm` will also fit Gibbs models, using the technique of ‘maximum pseudolikelihood’.

```r
data(swedishpines)
ppm(swedishpines, ~1, Strauss(r=7))
```
The command `ppm` will also fit Gibbs models, using the technique of ‘maximum pseudolikelihood’.

```r
data(swedishpines)
ppm(swedishpines, ~1, Strauss(r=7))
```

Stationary Strauss process

First order term:

```
    beta
0.02583902
```

Interaction: Strauss process

```
interaction distance:  7
Fitted interaction parameter gamma:  0.1841
```
The model can include both spatial trend and interpoint interaction.
The model can include both spatial trend and interpoint interaction.

```r
data(japanesepines)
ppm(japanesepines, ~polynom(x,y,3), Strauss(r=0.07))
```
The model can include both spatial trend and interpoint interaction.

```r
data(japanesepines)
ppm(japanesepines, ~polynom(x,y,3), Strauss(r=0.07))
```

Nonstationary Strauss process

Trend formula: `~polynom(x, y, 3)`

Fitted coefficients for trend formula:

<table>
<thead>
<tr>
<th>Term</th>
<th>Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>0.4925368</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x]</td>
<td>22.0485400</td>
</tr>
<tr>
<td>polynom(x, y, 3)[y]</td>
<td>-9.1889134</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x^2]</td>
<td>-14.6524958</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x.y]</td>
<td>-41.022232</td>
</tr>
<tr>
<td>polynom(x, y, 3)[y^2]</td>
<td>50.2099917</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x^3]</td>
<td>3.4935300</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x^2.y]</td>
<td>5.4524828</td>
</tr>
<tr>
<td>polynom(x, y, 3)[x.y^2]</td>
<td>23.9209323</td>
</tr>
<tr>
<td>polynom(x, y, 3)[y^3]</td>
<td>-38.3946389</td>
</tr>
</tbody>
</table>

Interaction: Strauss process

Interaction distance: 0.1

Fitted interaction parameter gamma: 0.5323
When we plot or predict a fitted Gibbs model, the first order trend $\beta(u)$ and/or the conditional intensity $\lambda(u \mid x)$ are plotted.

```r
fit <- ppm(japanesepines, ~x, Strauss(r=0.1))
plot(predict(fit))
plot(predict(fit, type="cif"))
```
A fitted Gibbs model can be simulated automatically using the Metropolis-Hastings algorithm (which only requires the conditional intensity).
Simulating the fitted model

A fitted Gibbs model can be simulated automatically using the Metropolis-Hastings algorithm (which only requires the conditional intensity).

```r
fit <- ppm(swedishpines, ~1, Strauss(r=7))
Xsim <- rmh(fit)
plot(Xsim)
```
A fitted Gibbs model can be simulated automatically using the Metropolis-Hastings algorithm (which only requires the conditional intensity).

```r
fit <- ppm(swedishpines, ~1, Strauss(r=7))
Xsim <- rmh(fit)
plot(Xsim)
```
Tests of goodness-of-fit can be performed by simulating from the fitted model.

\[
\text{plot(envelope(fit, Gest, nsim=19))}
\]
More powerful diagnostics are available.

\texttt{diagnose.ppm(fit)}
Marks
Each point in a spatial point pattern may carry additional information called a ‘mark’. It may be

**a continuous variate:** tree diameter, tree height

**a categorical variate:** label classifying the points into two or more different types (on/off, case/control, species, colour)

In spatstat version 1, the mark attached to each point must be a *single* value.
Categorical marks
A point pattern with categorical marks is usually called “multi-type”.

```r
> data(amacrine)
> amacrine
marked planar point pattern: 294 points
multitype, with levels = off   on
window: rectangle = [0, 1.6012] x [0, 1] units (one unit = 662 microns)
> plot(amacrine)
```

![Diagram of amacrine point pattern](image)
summary(amacrine)
summary(amacrine)

Marked planar point pattern: 294 points
Average intensity 184 points per square unit (one unit = 662 microns)
Multitype:

<table>
<thead>
<tr>
<th></th>
<th>frequency</th>
<th>proportion</th>
<th>intensity</th>
</tr>
</thead>
<tbody>
<tr>
<td>off</td>
<td>142</td>
<td>0.483</td>
<td>88.7</td>
</tr>
<tr>
<td>on</td>
<td>152</td>
<td>0.517</td>
<td>94.9</td>
</tr>
</tbody>
</table>

Window: rectangle = [0, 1.6012] x [0, 1] units
Window area = 1.60121 square units
Unit of length: 662 microns
Intensity of multitype patterns

\texttt{plot(split(amacrine))}

\texttt{split(amacrine)}

\begin{itemize}
  \item \textit{off}
  \begin{figure}
  \centering
  \includegraphics[width=0.4\textwidth]{off.png}
  \end{figure}
  \\
  \item \textit{on}
  \begin{figure}
  \centering
  \includegraphics[width=0.4\textwidth]{on.png}
  \end{figure}
\end{itemize}
data(lansing)
summary(lansing)
plot(lansing)
“Segregation” occurs when the intensity depends on the mark (i.e. on the type of point).

```r
plot(split(lansing))
```

split(lansing)
Intensity of multitype patterns

Let $\lambda(u, m)$ be the intensity function for points of type $m$ at location $u$. This can be estimated by kernel smoothing the data points of type $m$.

```r
plot(density(split(lansing)))
```
The probability that a point at location $u$ has mark $m$ is

$$p(m \mid u) = \frac{\lambda(u, m)}{\lambda(u)}$$

where $\lambda(u) = \sum_m \lambda(u, m)$ is the intensity function of points of all types.
D <- density(lansing)
Y <- density(split(lansing))
Dblackoak <- Y$blackoak
pBlackoak <- eval.im(Dblackoak/D)
plot(pBlackoak)
Interaction between types
In a multitype point pattern, there may be interaction between the points of different types, or between points of the same type.
Assume the points of type $i$ have uniform intensity $\lambda_i$, for all $i$. For two given types $i$ and $j$, the bivariate $G$-function $G_{ij}$ is

$$G_{ij}(r) = P(R_{ij} \leq r)$$

where $R_{ij}$ is the distance from a typical point of type $i$ to the nearest point of type $j$. 
plot(Gcross(amacrine, "on", "off"))
Bivariate \( G \)-function

```r
plot(alltypes(amacrine, Gcross))
```

array of Gcross function for amacrine.
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
</table>

Fitting Poisson models
For a *mutitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td></td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>$\log \lambda(u, m) = \beta$ constant.</td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td>log $\lambda(u, m) = \beta$ constant.</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>$\log \lambda(u, m) = \beta$ constant. Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td></td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>Command</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>$\log \lambda(u, m) = \beta$ constant.</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td>$\log \lambda(u, m) = \beta_m$</td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td>[ \log \lambda(u, m) = \beta \text{ constant.} ]</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks)</td>
<td>[ \log \lambda(u, m) = \beta_m ]</td>
</tr>
<tr>
<td></td>
<td>Different constant intensity for points of each type.</td>
</tr>
</tbody>
</table>

**Fitting Poisson models**
Fitting Poisson models

For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>[ \log \lambda(u, m) = \beta ] constant.</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td>[ \log \lambda(u, m) = \beta_m ]</td>
</tr>
<tr>
<td></td>
<td>Different constant intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks + x)</code></td>
<td></td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>( \log \lambda(u, m) = \beta ) constant.</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td>( \log \lambda(u, m) = \beta_m )</td>
</tr>
<tr>
<td></td>
<td>Different constant intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks + x)</code></td>
<td>( \log \lambda((x, y), m) = \beta_m + \alpha x )</td>
</tr>
</tbody>
</table>
### Fitting Poisson models

For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td>log $\lambda(u, m) = \beta$ constant.</td>
</tr>
<tr>
<td></td>
<td>Equal intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks)</td>
<td>log $\lambda(u, m) = \beta_m$</td>
</tr>
<tr>
<td></td>
<td>Different constant intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks + x)</td>
<td>log $\lambda((x, y), m) = \beta_m + \alpha x$</td>
</tr>
<tr>
<td></td>
<td>Common spatial trend</td>
</tr>
</tbody>
</table>
Fitting Poisson models

For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>$\log \lambda(u, m) = \beta$ constant. Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td>$\log \lambda(u, m) = \beta_m$</td>
</tr>
<tr>
<td></td>
<td>Different constant intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks + x)</code></td>
<td>$\log \lambda((x, y), m) = \beta_m + \alpha x$</td>
</tr>
<tr>
<td></td>
<td>Common spatial trend.</td>
</tr>
<tr>
<td></td>
<td>Different overall intensity for each type.</td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td>$\log \lambda(u, m) = \beta$ constant. Equal intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks)</td>
<td>$\log \lambda(u, m) = \beta_m$ Different constant intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks + x)</td>
<td>$\log \lambda((x, y), m) = \beta_m + \alpha x$ Common spatial trend Different overall intensity for each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks + x + marks:x)</td>
<td></td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ppm(X, ~1)</code></td>
<td>( \log \lambda(u, m) = \beta ) constant. Equal intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks)</code></td>
<td>( \log \lambda(u, m) = \beta_m ) Different constant intensity for points of each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks + x)</code></td>
<td>( \log \lambda((x, y), m) = \beta_m + \alpha x ) Common spatial trend Different overall intensity for each type.</td>
</tr>
<tr>
<td><code>ppm(X, ~marks + x + marks:x)</code></td>
<td>equivalent to</td>
</tr>
<tr>
<td><code>ppm(X, ~marks * x)</code></td>
<td></td>
</tr>
</tbody>
</table>
For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ppm(X, ~1)</td>
<td>( \log \lambda(u, m) = \beta ) constant.\nEqual intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks)</td>
<td>( \log \lambda(u, m) = \beta_m ) \nDifferent constant intensity for points of each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks + x)</td>
<td>( \log \lambda((x, y), m) = \beta_m + \alpha x ) \nCommon spatial trend \nDifferent overall intensity for each type.</td>
</tr>
<tr>
<td>ppm(X, ~marks + x + marks:x)</td>
<td>equivalent to ( \log \lambda((x, y), m) = \beta_m + \alpha_m x )</td>
</tr>
<tr>
<td>ppm(X, ~marks * x)</td>
<td></td>
</tr>
</tbody>
</table>
## Fitting Poisson models

For a *multitype* point pattern:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>INTERPRETATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{ppm(X, ~1)}</td>
<td>\log \lambda(u, m) = \beta \text{ constant.} \newline Equal intensity for points of each type.</td>
</tr>
<tr>
<td>\texttt{ppm(X, ~marks)}</td>
<td>\log \lambda(u, m) = \beta_m \newline Different constant intensity for points of each type.</td>
</tr>
<tr>
<td>\texttt{ppm(X, ~marks + x)}</td>
<td>\log \lambda((x, y), m) = \beta_m + \alpha x \newline Common spatial trend \newline Different overall intensity for each type.</td>
</tr>
<tr>
<td>\texttt{ppm(X, ~marks + x + marks:x)}</td>
<td>equivalent to \newline \log \lambda((x, y), m) = \beta_m + \alpha_m x \newline Different spatial trends for each type</td>
</tr>
</tbody>
</table>
Likelihood ratio test of segregation in Lansing Woods data:
Likelihood ratio test of segregation in Lansing Woods data:

```r
fit0 <- ppm(lansing, ~marks + polynom(x,y,3))
fit1 <- ppm(lansing, ~marks * polynom(x,y,3))
anova(fit0, fit1, test="Chi")
```
Likelihood ratio test of segregation in Lansing Woods data:

```r
fit0 <- ppm(lansing, ~marks + polynom(x,y,3))
fit1 <- ppm(lansing, ~marks * polynom(x,y,3))
anova(fit0, fit1, test="Chi")
```

Analysis of Deviance Table

<table>
<thead>
<tr>
<th>Model</th>
<th>.mpl.Y ~ marks + polynom(x, y, 3)</th>
<th>.mpl.Y ~ marks * polynom(x, y, 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resid.</td>
<td>Df Resid. Dev</td>
<td>Df Deviance</td>
</tr>
<tr>
<td>1</td>
<td>73515 17485.0</td>
<td>45 612.6 1.226e-100</td>
</tr>
</tbody>
</table>
fit1 <- ppm(lansing, ~marks * polynom(x,y,3))
plot(predict(fit1))
Inhomogeneous multitype $K$ function can be generalised to inhomogeneous multitype $K$ function.

```r
fit1 <- ppm(lansing, ~marks * polynom(x,y,3))
lamb <- predict(fit1)
plot(Kcross.inhom(lansing, "maple","hickory",
      lamb$markmaple, lamb$markhickory))
```
Multitype Gibbs models
The conditional intensity $\lambda(u, m \mid x)$ is essentially the conditional probability of finding a point of type $m$ at location $u$, given complete information about the rest of the process $x$. 
> ppm(amacrine, ~marks, Strauss(r=0.04))

Stationary Strauss process

First order terms:
beta_off beta_on
156.0724 162.1160

Interaction: Strauss process
interaction distance: 0.04
Fitted interaction parameter gamma: 0.4464
> rad <- matrix(c(0.03, 0.04, 0.04, 0.02), 2, 2)
> ppm(amacrine, ~marks,
      MultiStrauss(radii=rad,types=c("off", "on")))

Stationary Multitype Strauss process

First order terms:
beta_off  beta_on
120.2312  108.8413

Interaction radii:
   off   on
off 0.03  0.04
on  0.04  0.02

Fitted interaction parameters gamma_ij:
   off   on
off 0.0619 0.8786
on  0.8786 0.0000
www.spatstat.org