

Advanced R course on spatial point patterns

Adrian Baddeley / Ege Rubak
Curtin University / Aalborg University

SSAI 2017

Plan of Workshop

1. Introduction
2. Inhomogeneous Intensity
3. Intensity dependent on a covariate
4. Fitting Poisson models
5. Marked point patterns
6. Correlation
7. Envelopes and Monte Carlo tests
8. Spacing and nearest neighbours
9. Cluster and Cox models
10. Gibbs models
11. Multitype summary functions and models

These slides are a summary of
the most important concepts
in the workshop.

The slides are a summary

These slides are a summary of
the most important concepts
in the workshop.

Use them for review/reminder

Types of spatial data

Three basic types of spatial data:

Three basic types of spatial data:

- geostatistical

Three basic types of spatial data:

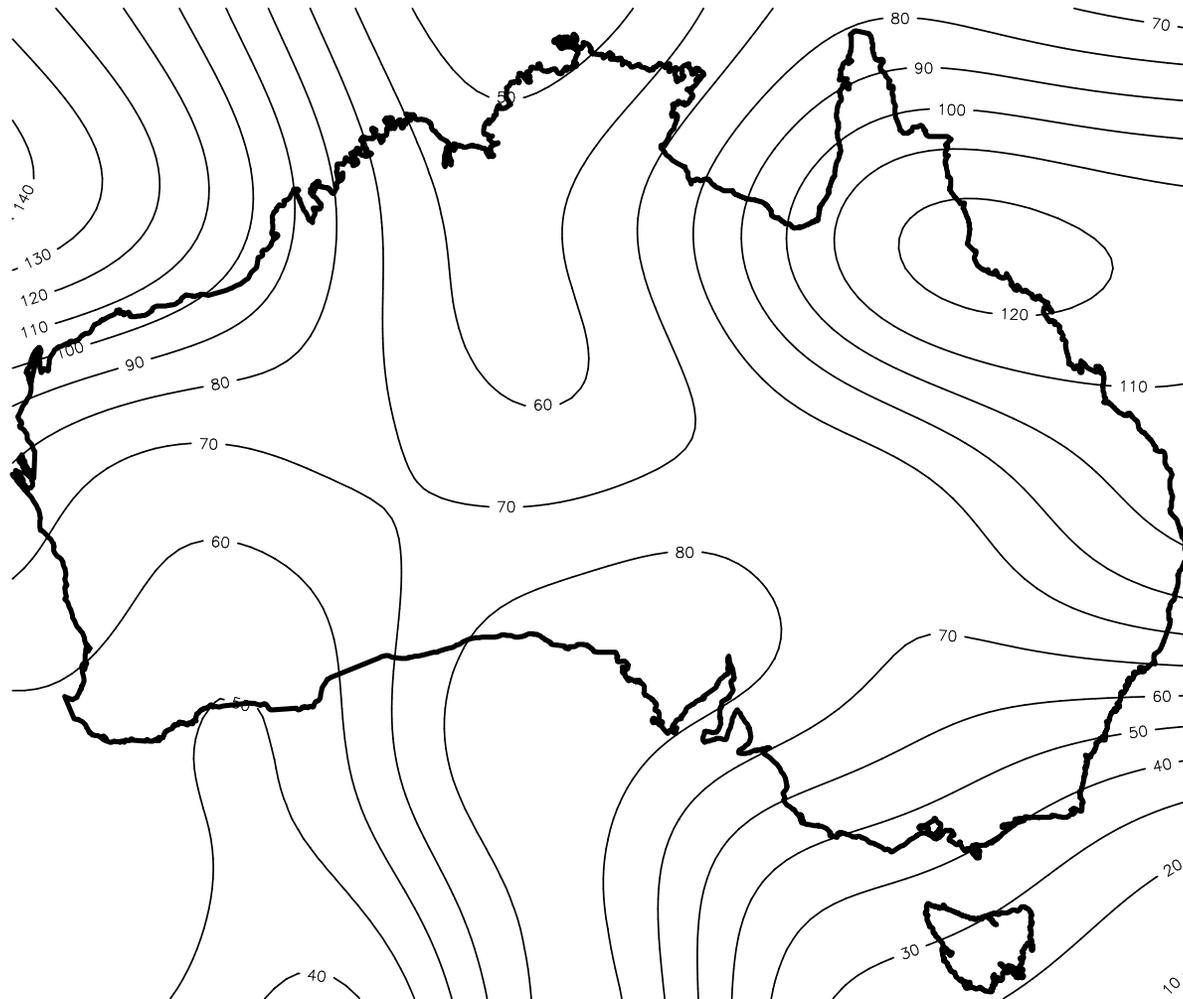
- geostatistical
- regional

Three basic types of spatial data:

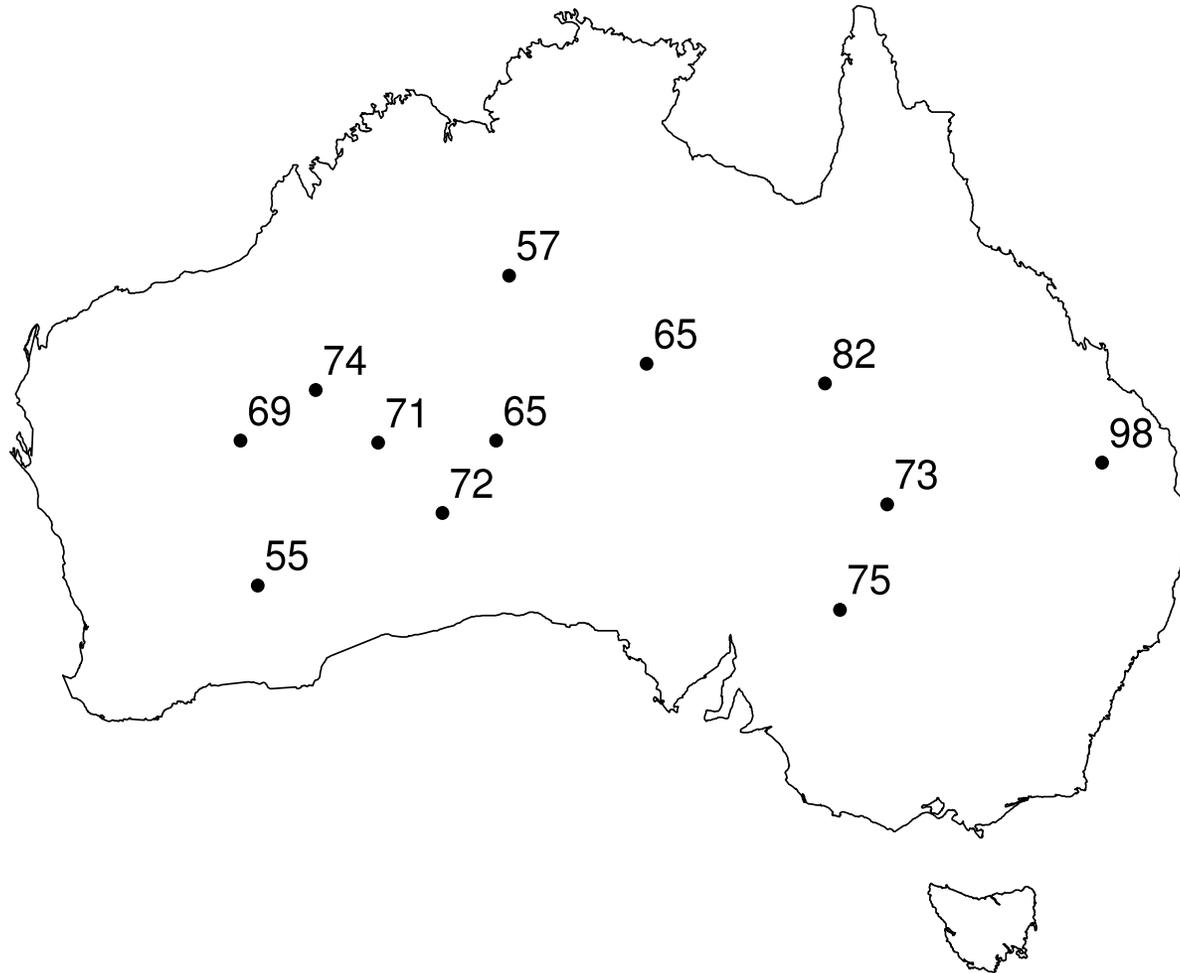
- geostatistical
- regional
- point pattern

GEOSTATISTICAL DATA:

The quantity of interest has a value at any location, ...

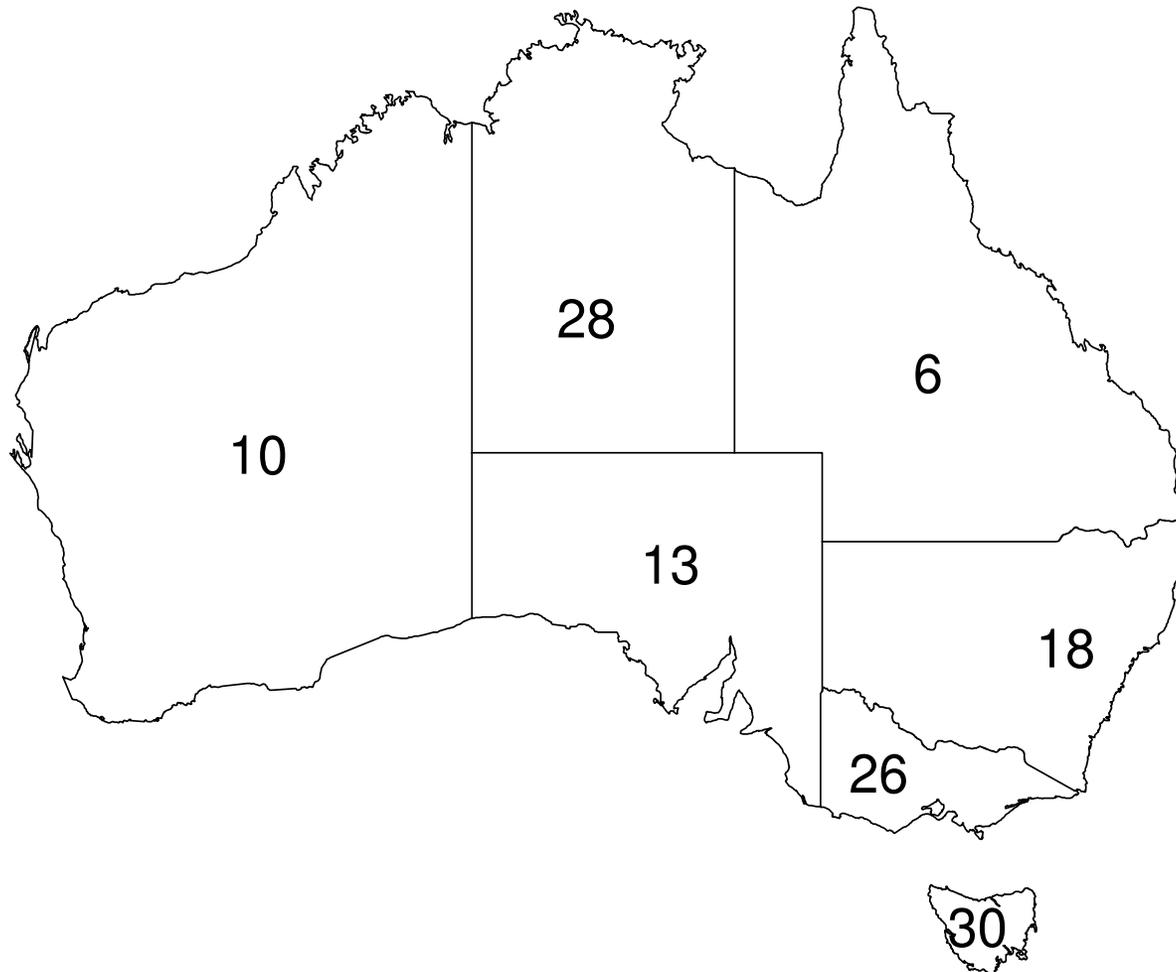


...but we only measure the quantity at certain sites. These values are our data.



REGIONAL DATA:

The quantity of interest is only defined for regions. It is measured/reported for certain *fixed* regions.



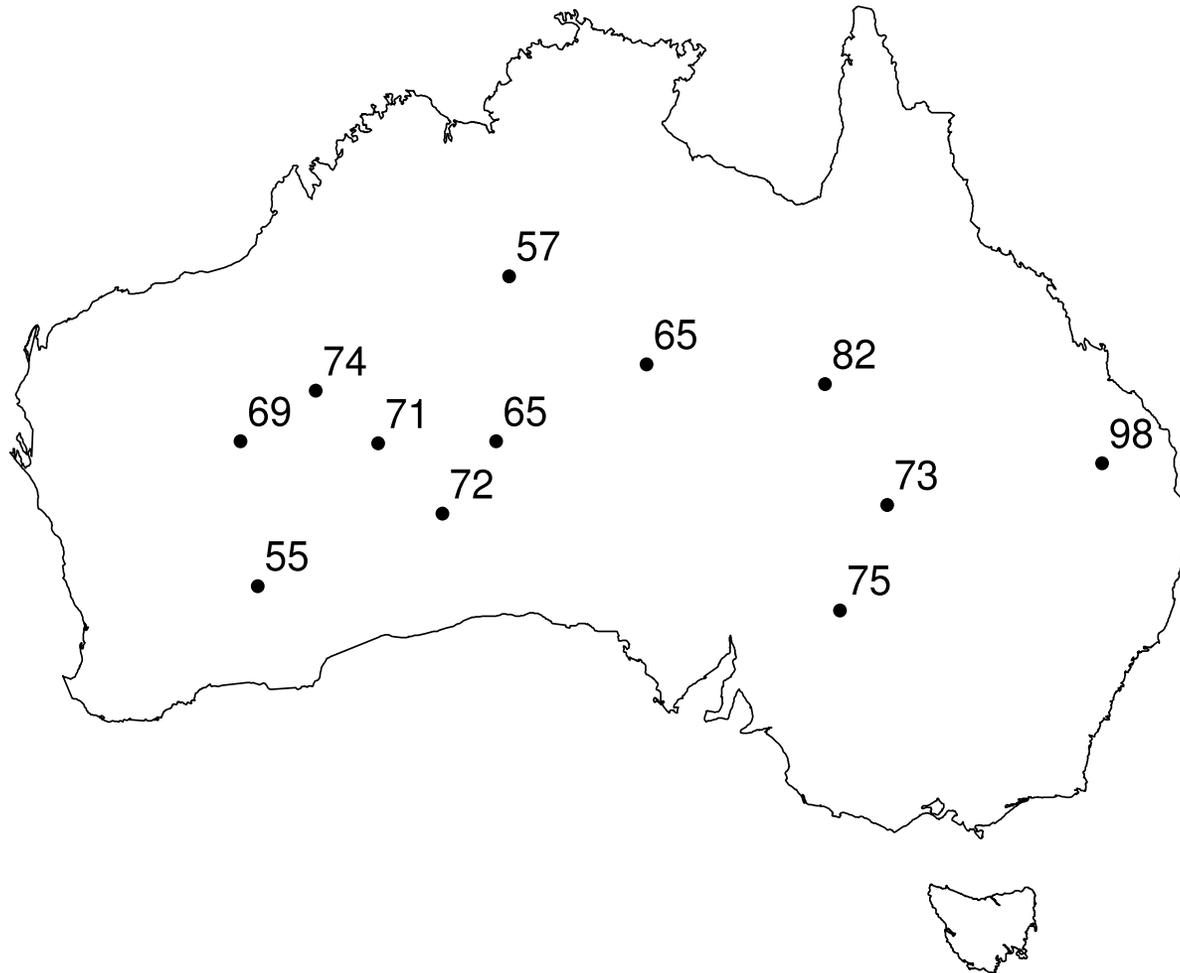
POINT PATTERN DATA:

The main interest is in the *locations* of all occurrences of some event (e.g. tree deaths, meteorite impacts, robberies). Exact locations are recorded.



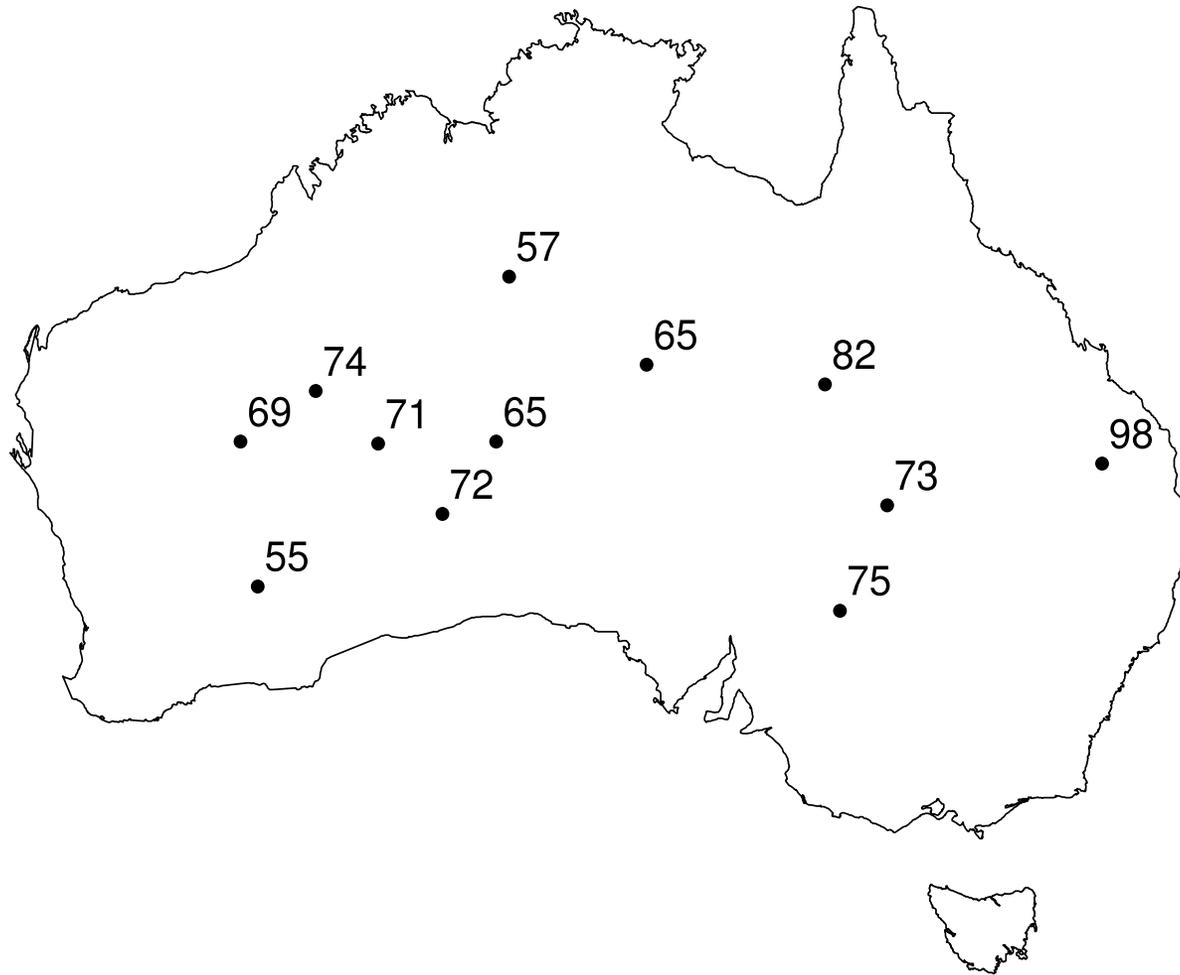
Points with marks

Points may also carry data (e.g. tree heights, meteorite composition)



Point pattern or geostatistical data?

POINT PATTERN OR GEOSTATISTICAL DATA?



Explanatory vs. response variables

Response variable: the quantity that we want to “predict” or “explain”

Explanatory variable: quantity that can be used to “predict” or “explain” the response.

Point pattern or geostatistical data?

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.

Point pattern or geostatistical data?

“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 in *R*:

For information on spatial statistics software in *R*:

- go to cran.r-project.org

For information on spatial statistics software in *R*:

- go to `cran.r-project.org`
- find `Task Views`

For information on spatial statistics software in *R*:

- go to `cran.r-project.org`
- find `Task Views --- Spatial`

GIS = Geographical Information System

GIS = Geographical Information System

ArclInfo

GIS = Geographical Information System

ArcInfo proprietary `esri.com`

GIS = Geographical Information System

ArcInfo proprietary `esri.com`

GRASS open source `grass.osgeo.org`



Recommendations:

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:

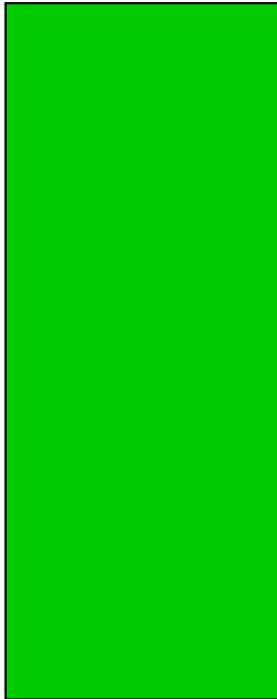
For visualisation of spatial data, especially for presentation graphics, use a **GIS**.

For statistical analysis of spatial data, use **R**.

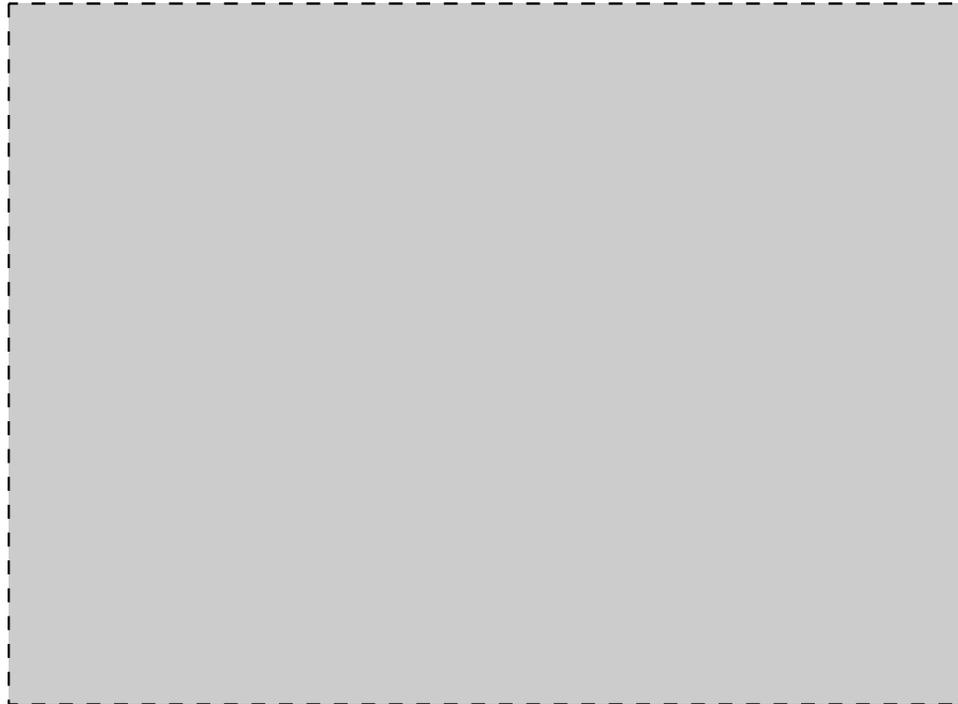
Establish two-way **communication** between GIS and R, either through a direct software interface, or by reading/writing files in mutually acceptable format.

Putting the pieces together

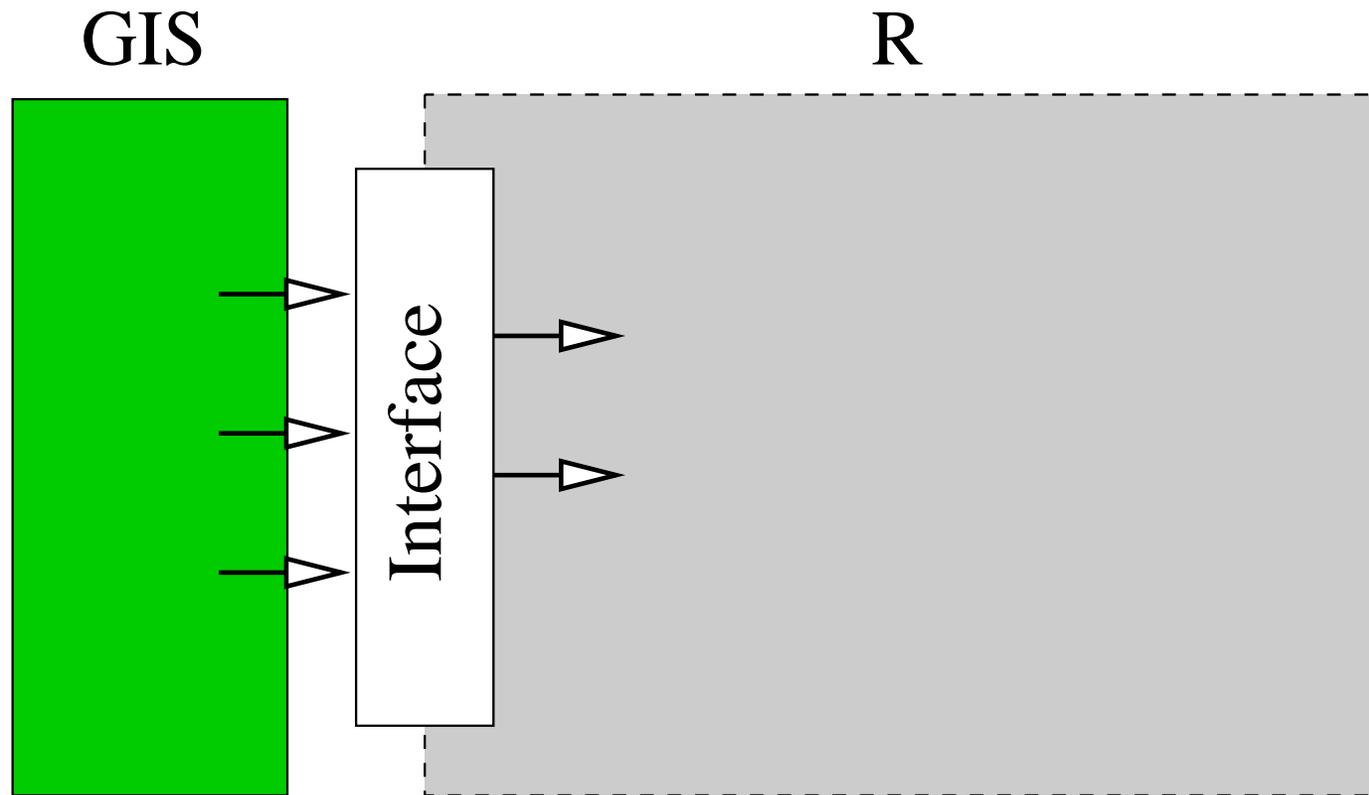
GIS



R

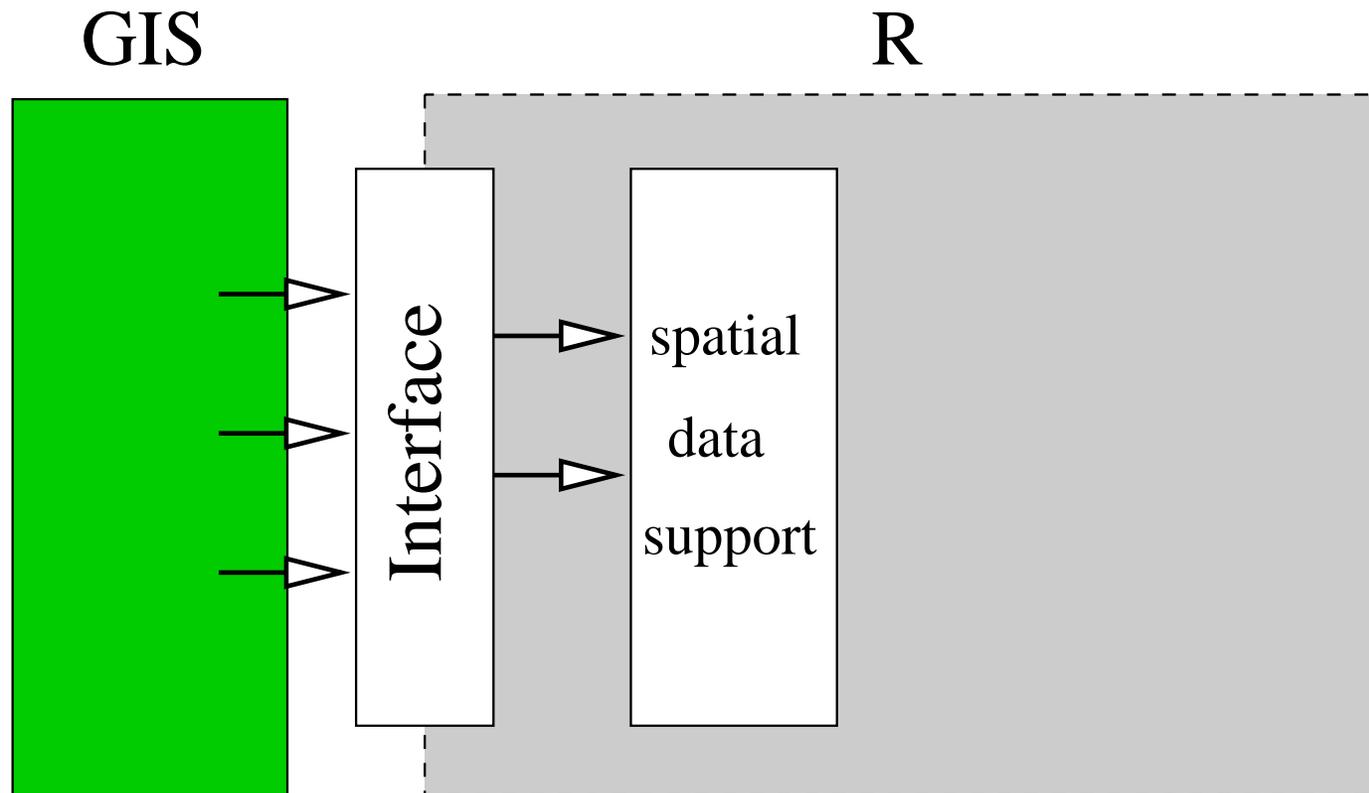


Putting the pieces together



Interface between R and GIS (online or offline)

Putting the pieces together



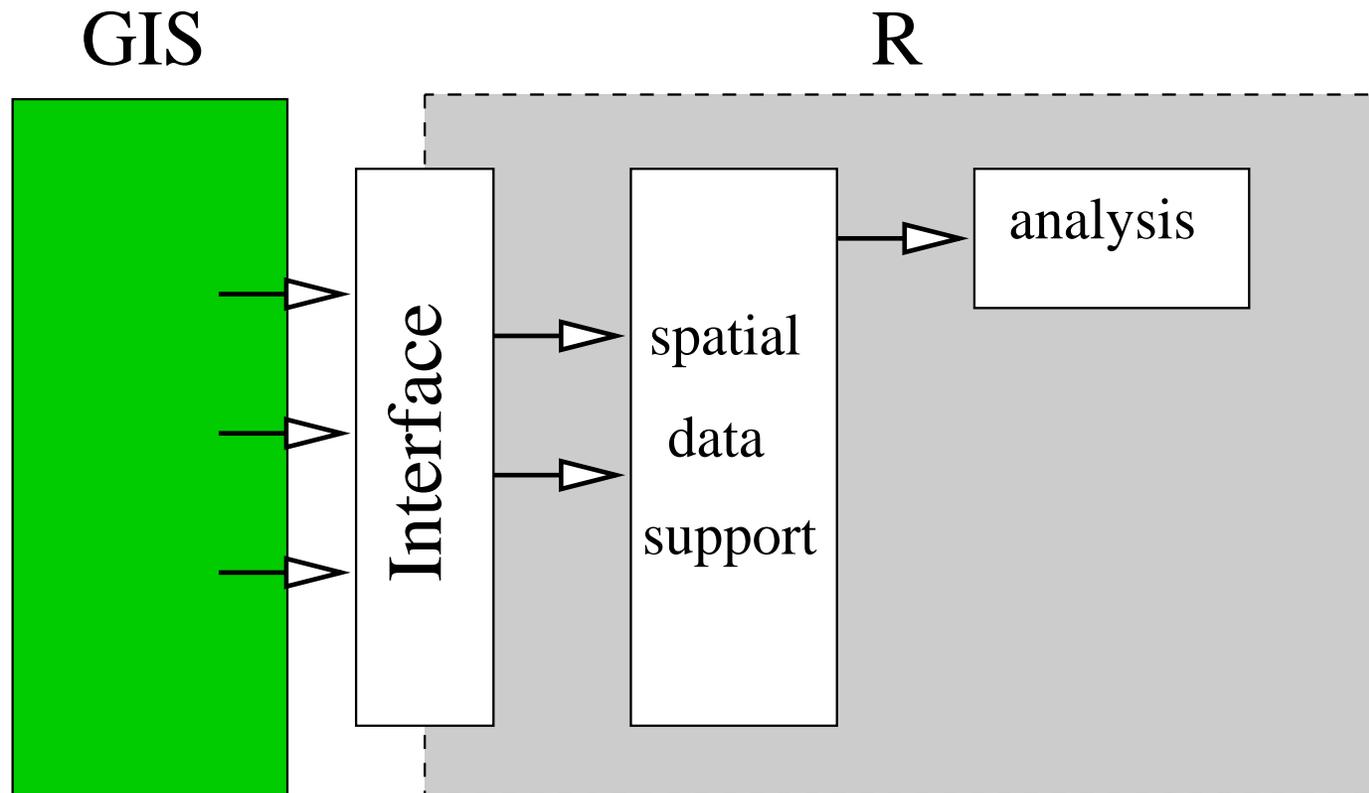
Support for spatial data: data structures, classes, methods

R packages supporting spatial data

R packages supporting spatial data classes:

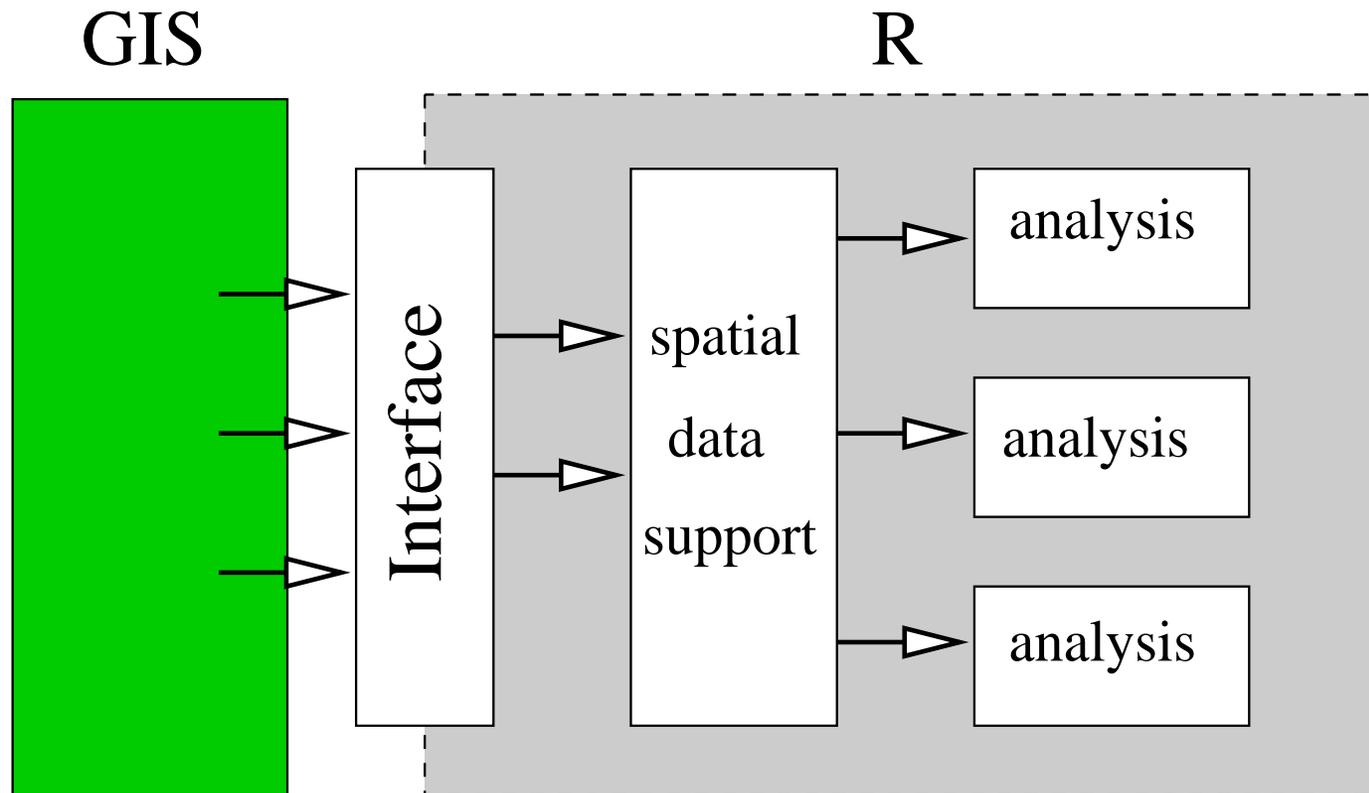
sp	generic
maps	polygon maps
spatstat	point patterns

Putting the pieces together



Capabilities for statistical analysis

Putting the pieces together



Multiple packages for different analyses

R packages for geostatistical data

<code>gstat</code>	classical geostatistics
<code>geoR</code>	model-based geostatistics
<code>RandomFields</code>	stochastic processes
<code>akima</code>	interpolation

R packages for geostatistical data

<code>gstat</code>	classical geostatistics
<code>geoR</code>	model-based geostatistics
<code>RandomFields</code>	stochastic processes
<code>akima</code>	interpolation

R packages for regional data

<code>spdep</code>	spatial dependence
<code>spgwr</code>	geographically weighted regression

R packages for geostatistical data

<code>gstat</code>	classical geostatistics
<code>geoR</code>	model-based geostatistics
<code>RandomFields</code>	stochastic processes
<code>akima</code>	interpolation

R packages for regional data

<code>spdep</code>	spatial dependence
<code>spgwr</code>	geographically weighted regression

R packages for point patterns

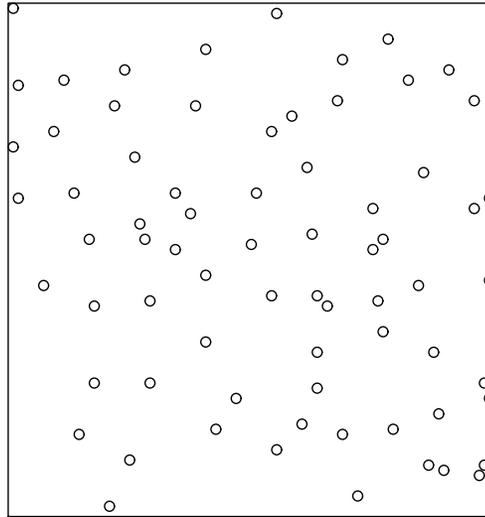
<code>spatstat</code>	parametric modelling, diagnostics
<code>splancs</code>	nonparametric, space-time

Spatial point patterns

The R package `spatstat` supports statistical analysis for spatial point patterns.

Point patterns

A **point pattern** dataset gives the locations of objects/events occurring in a study region.

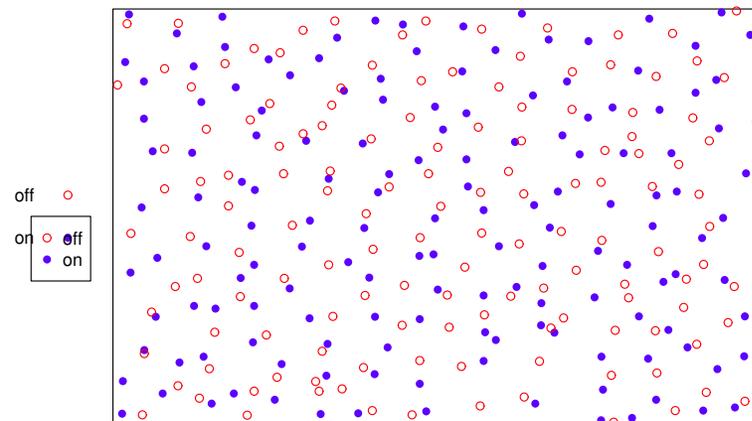


The points could represent trees, animal nests, earthquake epicentres, petty crimes, domiciles of new cases of influenza, galaxies, etc.

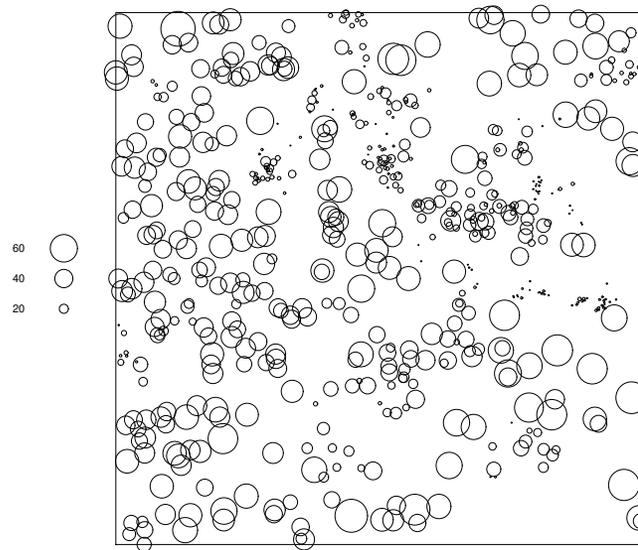
The points may have extra information called **marks** attached to them. The mark represents an “attribute” of the point.

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:

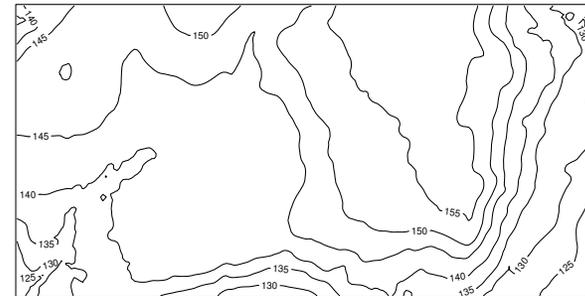
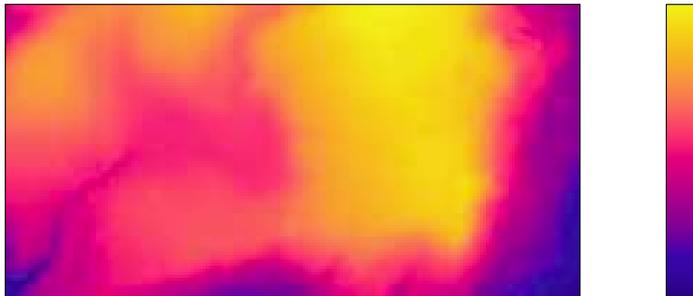


The mark variable could be *continuous*, e.g. tree diameter:

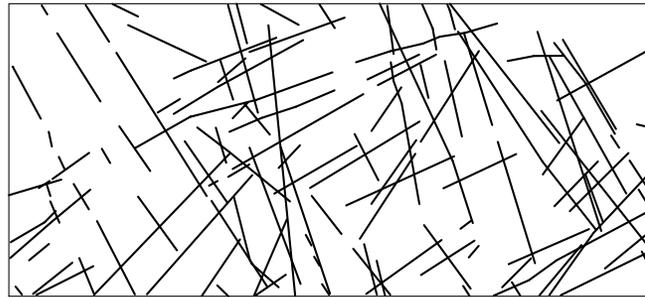


Our dataset may also include **covariates** — any data that we treat as explanatory, rather than as part of the ‘response’.

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:

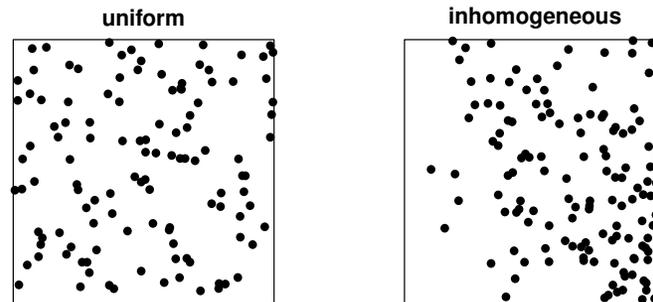


Covariate data may be another *spatial pattern* such as another point pattern, or a line segment pattern, e.g. a map of geological faults:

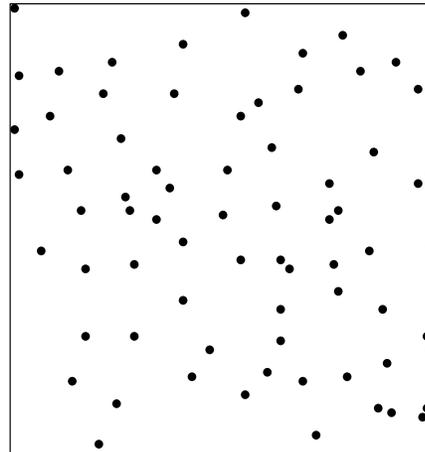


Intensity

'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)
```



Quadrat counts

Divide study region into rectangles ('quadrats') of equal size, and count points in each rectangle.

```
Q <- quadratcount(P, nx=3, ny=3)
```

```
Q
```

```
plot(Q, add=TRUE)
```

P

8	6	7
8	11	9
5	6	11

χ^2 test of uniformity

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 χ^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 χ^2 goodness-of-fit test statistic

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

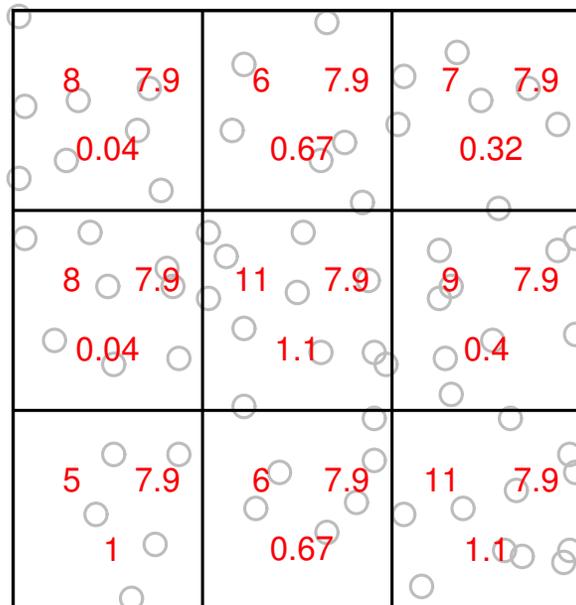
```
> quadrat.test(P, nx=3, ny=3)
```

Chi-squared test of CSR using quadrat counts

```
data: P
```

```
X-squared = 4.6761, df = 8, p-value = 0.7916
```

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



Kernel smoothed intensity

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

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

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

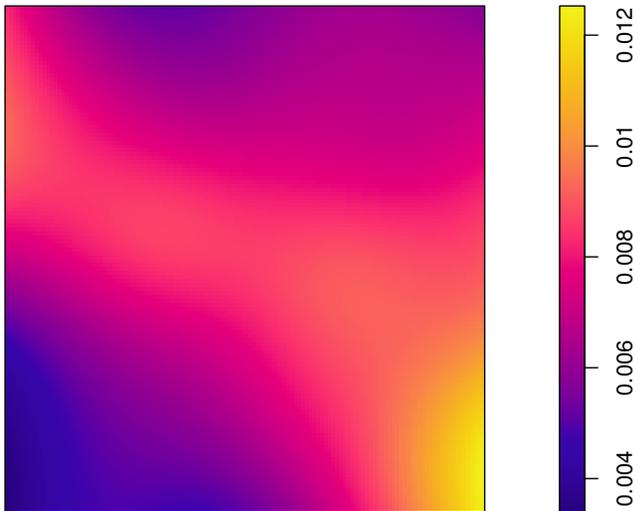
Kernel smoothed intensity

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

where $\kappa(u)$ is the kernel function and x_1, \dots, 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*

```
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”.

COMMAND

INTENSITY

ppm(P ~ 1)

$$\log \lambda(u) = \beta_0$$

COMMAND

INTENSITY

ppm(P ~ 1)

$$\log \lambda(u) = \beta_0$$

β_0, β_1, \dots denote parameters to be estimated.

COMMAND

INTENSITY

`ppm(P ~ 1)`

$$\log \lambda(u) = \beta_0$$

`ppm(P ~ x)`

$$\log \lambda((x, y)) = \beta_0 + \beta_1 x$$

β_0, β_1, \dots denote parameters to be estimated.

COMMAND

INTENSITY

`ppm(P ~ 1)`

$$\log \lambda(u) = \beta_0$$

`ppm(P ~ x)`

$$\log \lambda((x, y)) = \beta_0 + \beta_1 x$$

`ppm(P ~ x + y)`

$$\log \lambda((x, y)) = \beta_0 + \beta_1 x + \beta_2 y$$

β_0, β_1, \dots denote parameters to be estimated.

```
> 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

COMMAND

INTENSITY

`ppm(P ~polynom(x,y,3))`

exp(3rd order polynomial in x and y)

COMMAND

INTENSITY

`ppm(P ~polynom(x,y,3))`

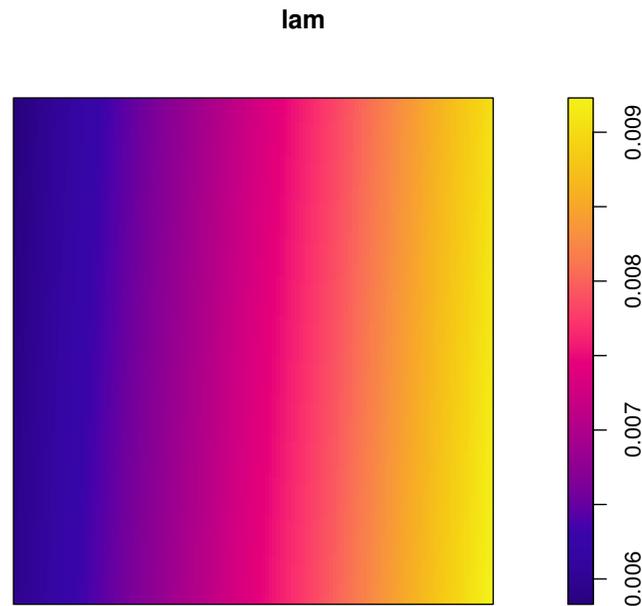
exp(3rd order polynomial in x and y)

`ppm(P ~I(y > 18))`

different constants above and below
the line $y = 18$

```
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")
```

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

Analysis of Deviance Table

Model 1: ~1

Poisson

Model 2: ~x + y + I(x^2) + I(x * y) + I(y^2)

Poisson

	Npar	Df	Deviance	Pr(>Chi)
--	------	----	----------	----------

1	1			
---	---	--	--	--

2	6	5	7.4821	0.1872
---	---	---	--------	--------

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

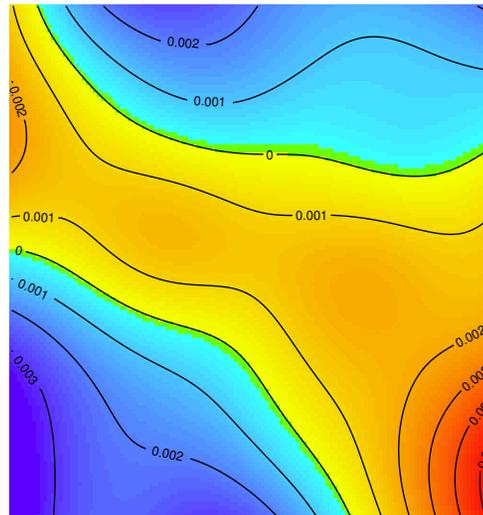
Analysis of Deviance Table

Model	Npar	Df	Deviance	Pr(>Chi)	
Model 1: ~1	1				Poisson
Model 2: ~x + y + I(x^2) + I(x * y) + I(y^2)	6	5	7.4821	0.1872	Poisson

The p -value 0.19 exceeds 0.05 so the log-quadratic spatial trend is *not significant*.

```
diagnose.ppm(fit0, which="smooth")
```

Smoothed raw residuals



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

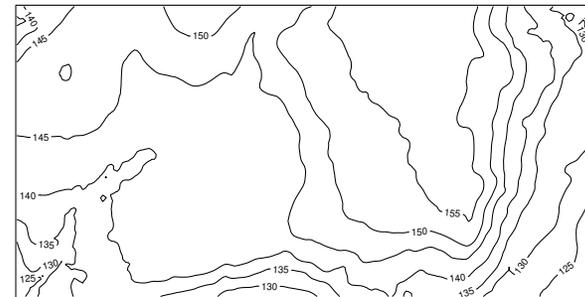
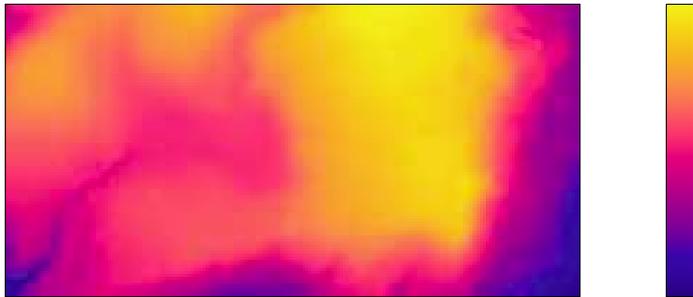
A *spatial covariate* is a function $Z(u)$ of spatial location.

- geographical coordinates
- terrain altitude
- soil pH
- distance from location u to another feature

Spatial covariates

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:

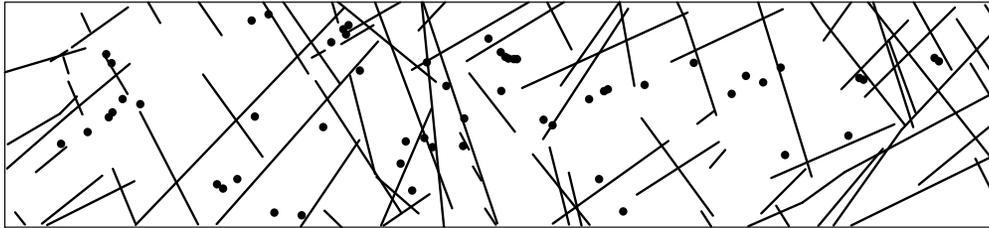


For a point pattern dataset with covariate data, we typically

- investigate whether the intensity depends on the covariates
- allow for covariate effects on intensity before studying dependence between points

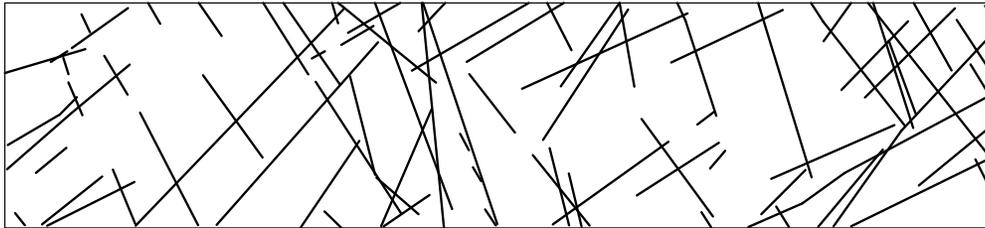
Example: Queensland copper data

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 .

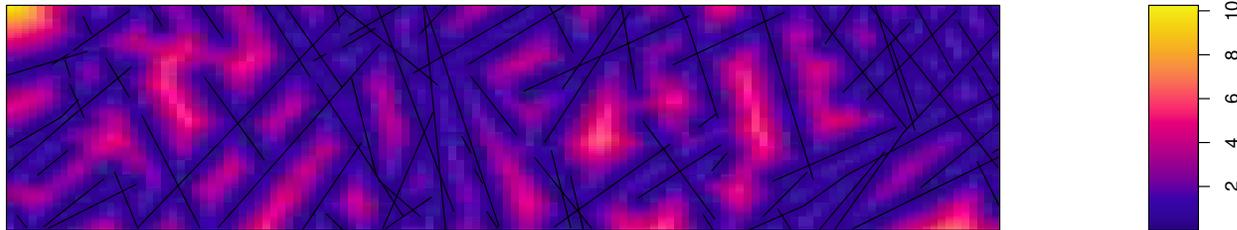
Example: $Z(u) = \text{distance from } u \text{ to nearest line.}$

For analysis, we need a value $Z(u)$ defined at each location u .

Example: $Z(u)$ = distance from u to nearest line.

```
Z <- distmap(Y)
```

```
plot(Z)
```



Lurking variable plot

We want to determine whether intensity depends on a spatial covariate Z .

Lurking variable plot

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 $Z(x_i) \leq z$.

Lurking variable plot

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 variable plot

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)

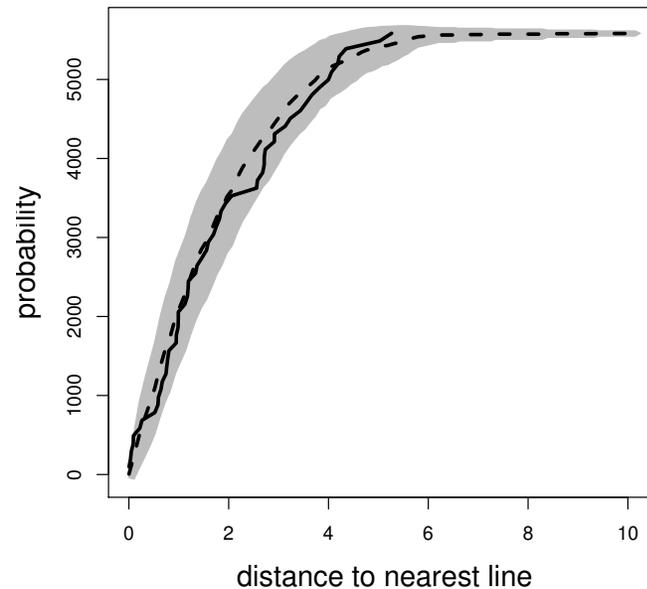
Lurking variable plot

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)



Kolmogorov-Smirnov test

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

Kolmogorov-Smirnov test

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
```

```
Z <- distmap(Y)  
ppm(P ~ Z)
```

Fits the model

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

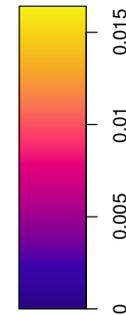
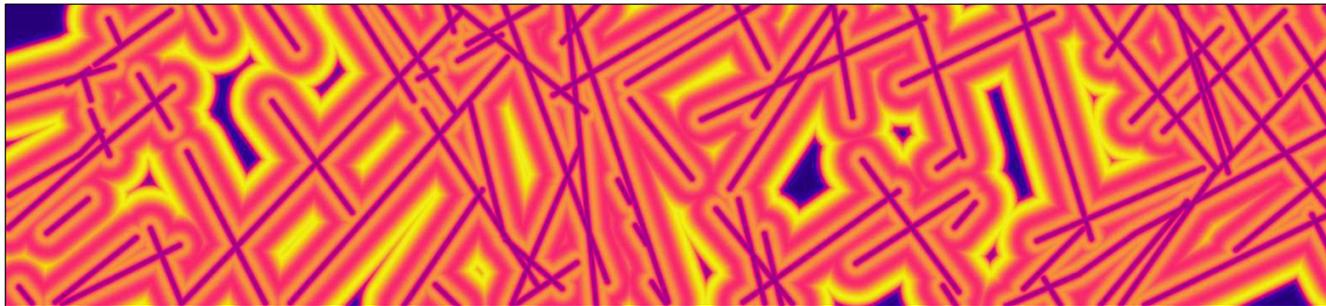
where $Z(u)$ is the distance from u to the nearest line segment.

```
Z <- distmap(Y)
```

```
ppm(P ~ polynom(Z,5))
```

fits a model in which $\log \lambda(u)$ is a 5th order polynomial function of $Z(u)$.

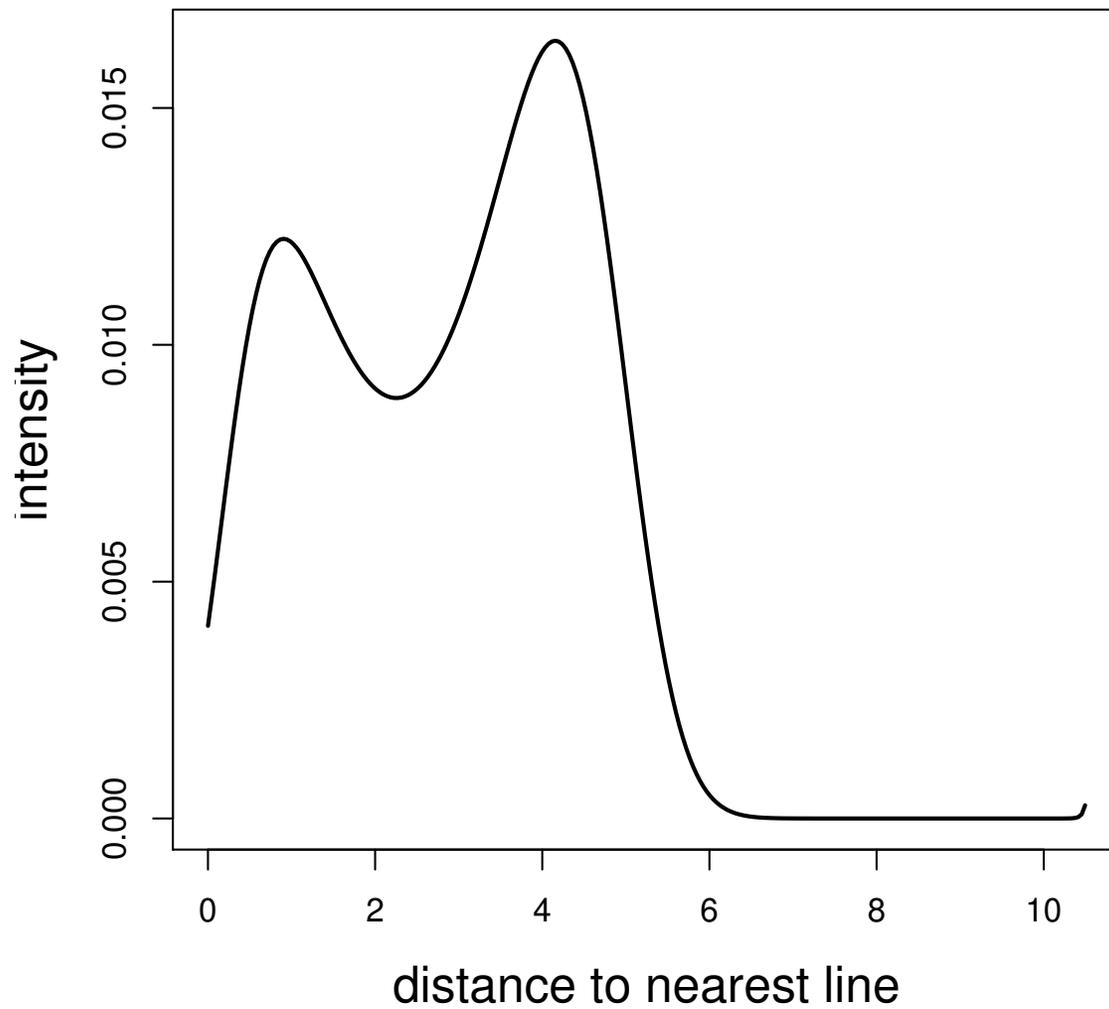
```
fit <- ppm(P ~polynom(Z,5))  
plot(predict(fit))
```



```
plot(effectfun(fit))
```

plots fitted curve of λ against Z .

effectfun fit



```
fit0 <- ppm(P ~1)
fit1  <- ppm(P ~polynom(Z,5))
anova(fit0, fit1, test="Chi")
```

```
fit0 <- ppm(P ~1)
fit1 <- ppm(P ~polynom(Z,5))
anova(fit0, fit1, test="Chi")
```

Analysis of Deviance Table

Model	1:	~1				Poisson
Model 2:	~Z + I(Z^2) + I(Z^3) + I(Z^4) + I(Z^5)					Poisson
	Npar	Df	Deviance	Pr(>Chi)		
1	1					
2	6	5	3.6476	0.6012		

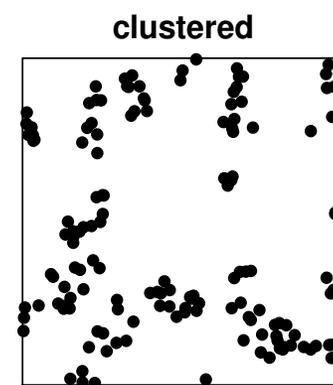
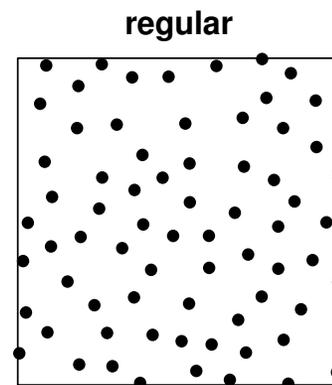
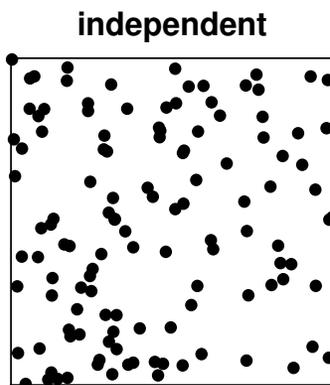
```
fit0 <- ppm(P ~1)
fit1  <- ppm(P ~polynom(Z,5))
anova(fit0, fit1, test="Chi")
```

Analysis of Deviance Table

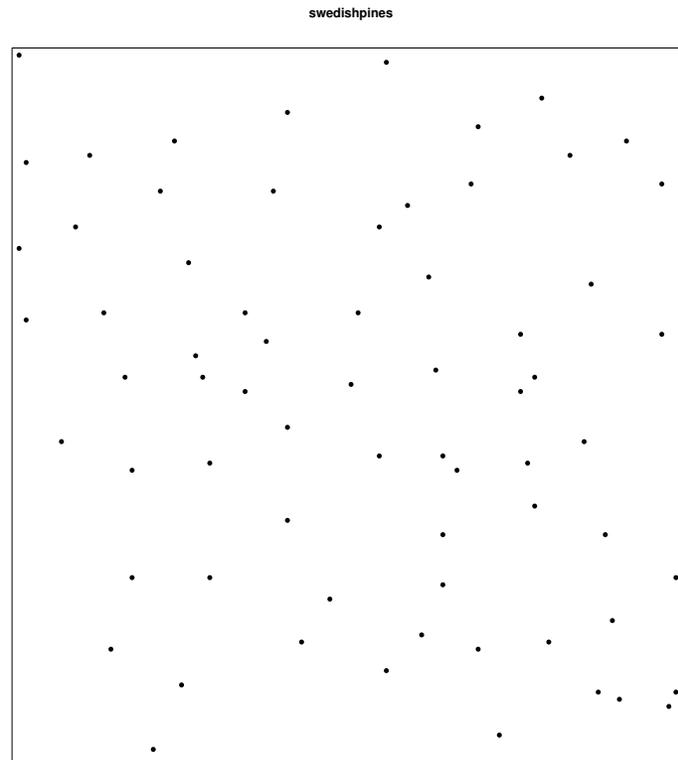
Model	1:	~1				Poisson
Model 2:	~Z + I(Z^2) + I(Z^3) + I(Z^4) + I(Z^5)					Poisson
	Npar	Df	Deviance	Pr(>Chi)		
1	1					
2	6	5	3.6476	0.6012		

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.

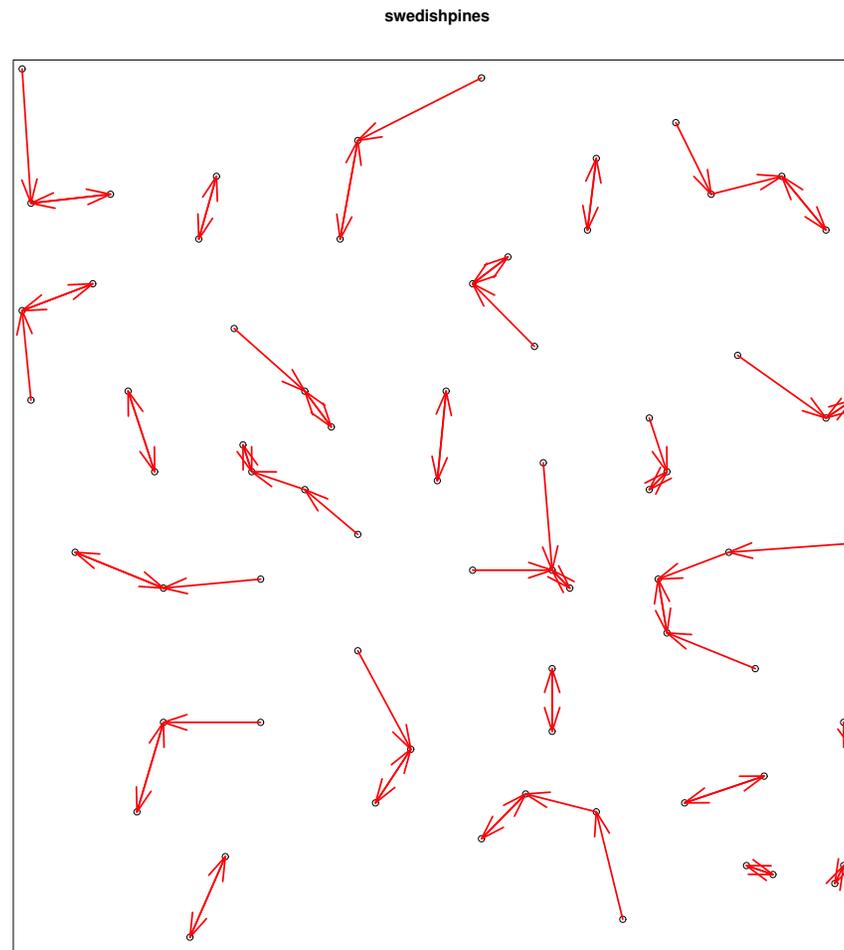


Example: spacing between points in Swedish Pines data



Example

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)

```
> 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)

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

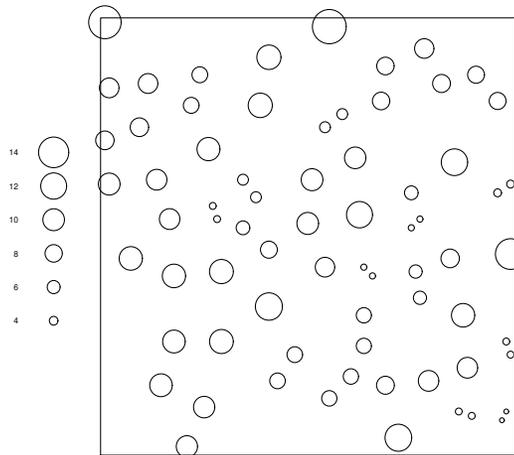
Value greater than 1 suggests a regular pattern.

Exploratory approach:

Exploratory approach:

- plot NND for each point

```
P <- swedishpines  
marks(P) <- nndist(P)  
plot(P, markscale=0.5)
```



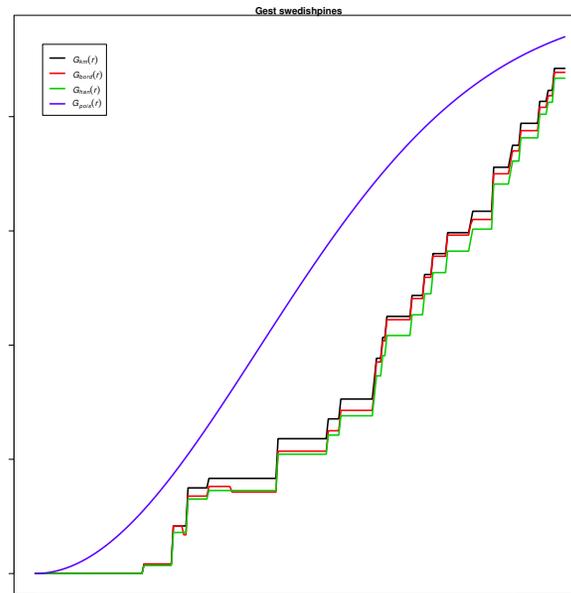
Exploratory approach:

- plot NND for each point

Exploratory approach:

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

```
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.

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

```
Stationary Geyer saturation process
```

```
First order term:
```

```
beta
```

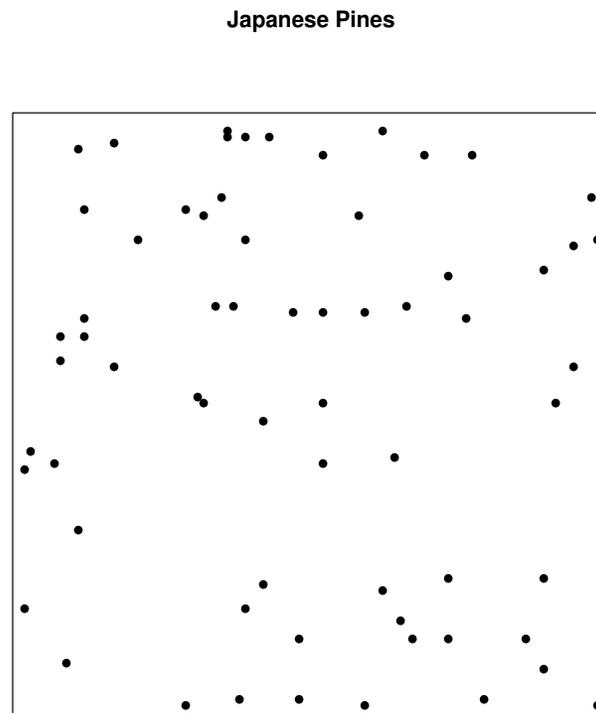
```
0.00971209
```

```
Fitted interaction parameter gamma: 0.6335
```

Example: Japanese pines

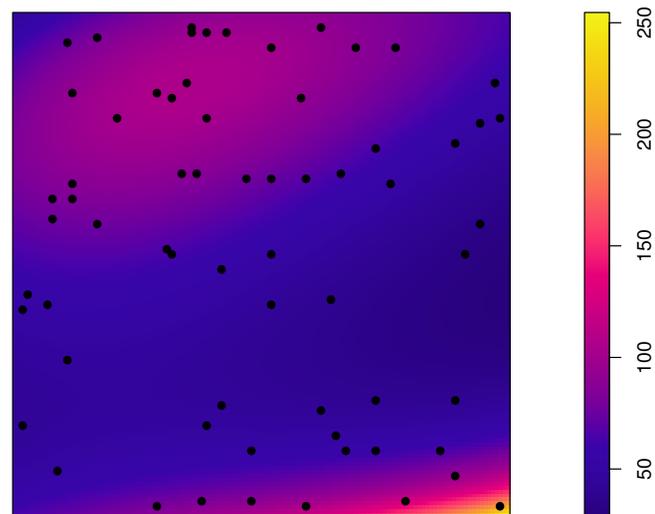
Locations of 65 saplings of Japanese pine in a 5.7×5.7 metre square sampling region in a natural stand.

```
data(japanesepines)  
J <- japanesepines  
plot(J)
```



```
fit <- ppm(J ~polynom(x,y,3))  
plot(predict(fit))  
plot(J, add=TRUE)
```

predict fit

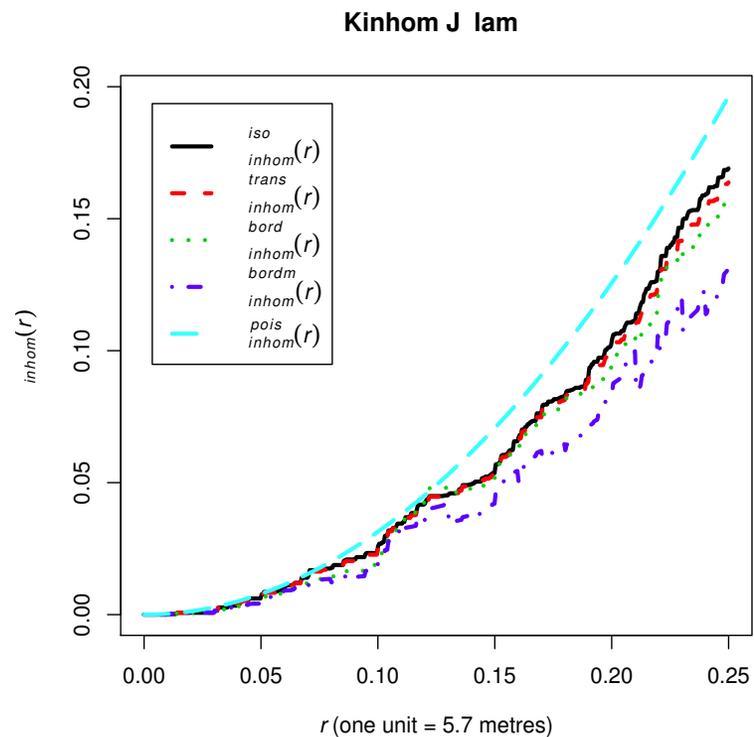


Adjusting for inhomogeneity

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

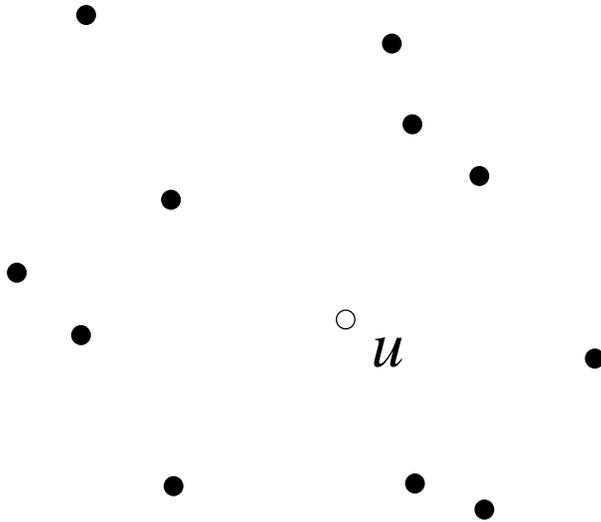
```
lam <- predict(fit)  
plot(Kinhom(J, lam))
```



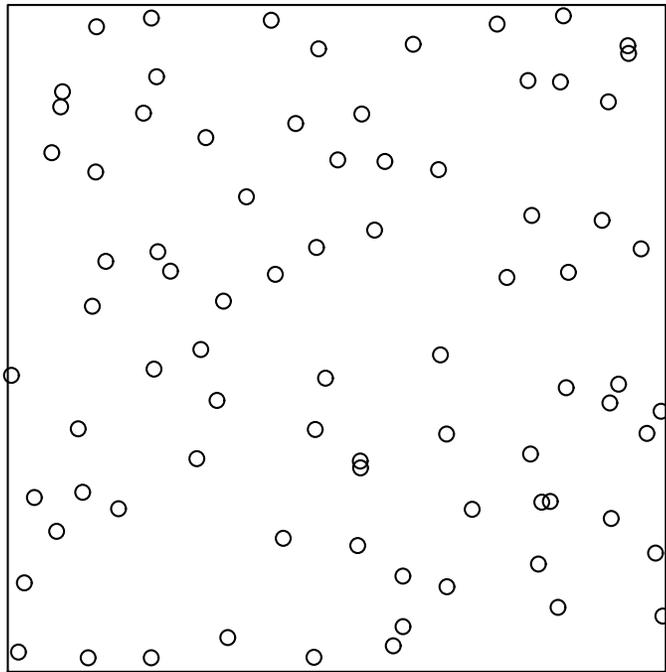
Conditional intensity

A point process model can also be defined through its *conditional intensity* $\lambda(u \mid \mathbf{x})$.

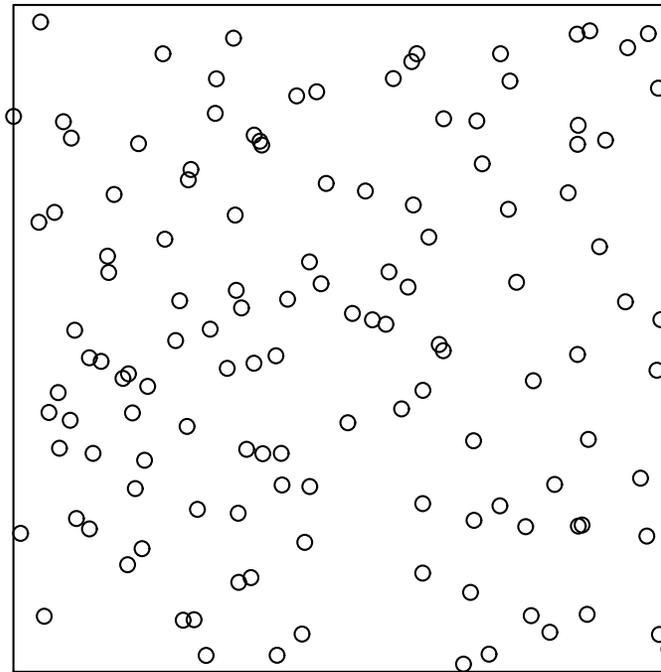
This is essentially the conditional probability of finding a point of the process at the location u , given complete information about the rest of the process \mathbf{x} .



Strauss($\gamma = 0.2$)



Strauss($\gamma = 0.7$)



Fitting Gibbs models

The command `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’.

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

The command `ppm` will also fit Gibbs models, using the technique of ‘maximum pseudolikelihood’.

```
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

Fitting Gibbs models

The model can include both spatial trend and interpoint interaction.

The model can include both spatial trend and interpoint interaction.

```
data(japanesepines)
```

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

Fitting Gibbs models

The model can include both spatial trend and interpoint interaction.

```
data(japanesepines)
```

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

Nonstationary Strauss process

Trend formula: \sim polynom(x, y, 3)

Fitted coefficients for trend formula:

(Intercept)	polynom(x, y, 3) [x]	polynom(x, y, 3) [y]
0.4925368	22.0485400	-9.1889134
polynom(x, y, 3) [x ²]	polynom(x, y, 3) [x.y]	polynom(x, y, 3) [y ²]
-14.6524958	-41.0222232	50.2099917
polynom(x, y, 3) [x ³]	polynom(x, y, 3) [x ² .y]	polynom(x, y, 3) [x.y ²]
3.4935300	5.4524828	23.9209323
polynom(x, y, 3) [y ³]		
-38.3946389		

Interaction: Strauss process

interaction distance: 0.1

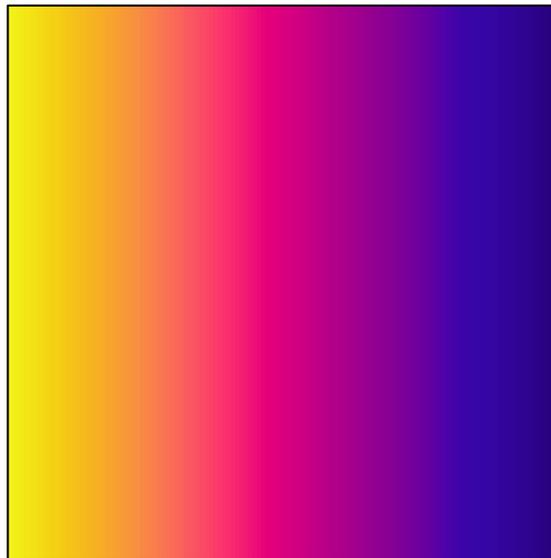
Fitted interaction parameter gamma: 0.5323

Plotting a fitted model

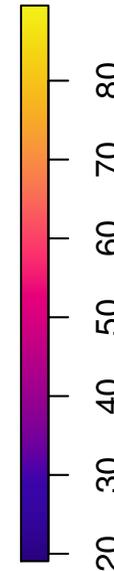
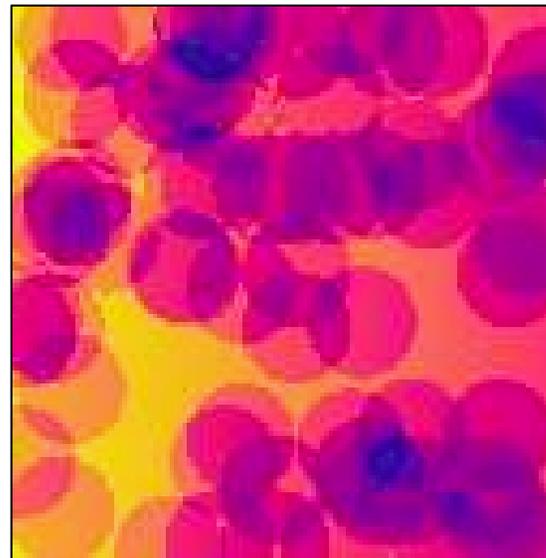
When we plot or predict a fitted Gibbs model, the first order trend $\beta(u)$ and/or the conditional intensity $\lambda(u \mid \mathbf{x})$ are plotted.

```
fit <- ppm(japanesepines ~x, Strauss(r=0.1))  
plot(predict(fit))  
plot(predict(fit, type="cif"))
```

predict fit



predict fit type cif



Simulating the fitted model

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

Simulating the fitted model

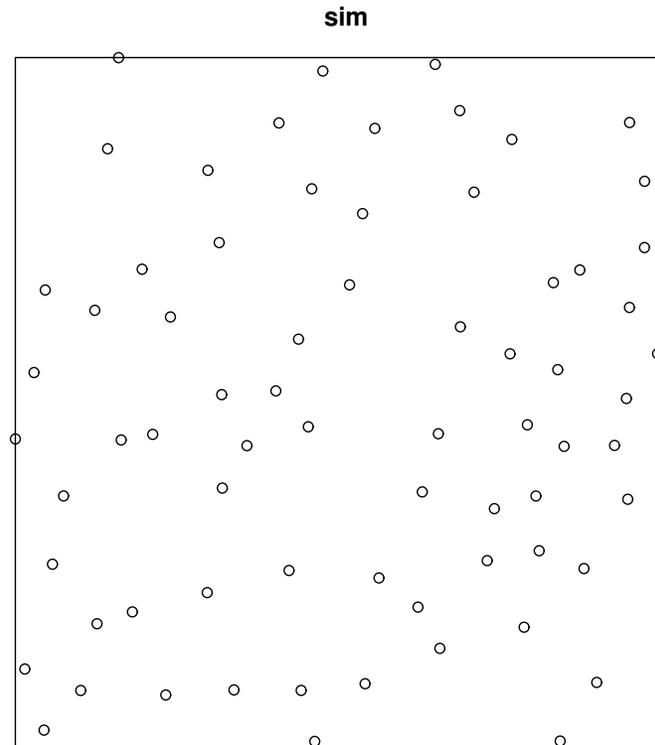
A fitted Gibbs model can be simulated automatically using the Metropolis-Hastings algorithm (which only requires the conditional intensity).

```
fit <- ppm(swedishpines ~1, Strauss(r=7))  
Xsim <- simulate(fit)  
plot(Xsim)
```

Simulating the fitted model

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

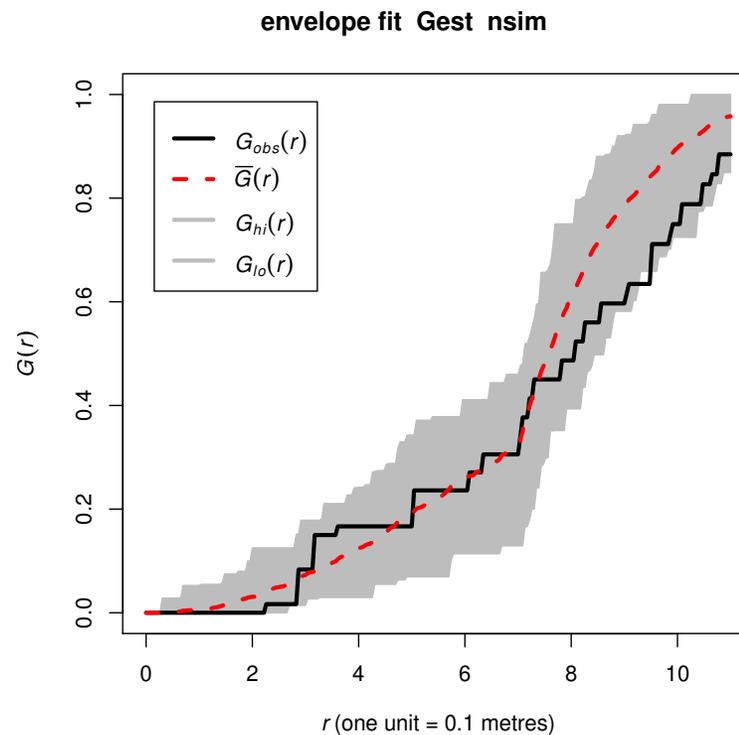
```
fit <- ppm(swedishpines ~1, Strauss(r=7))  
Xsim <- simulate(fit)  
plot(Xsim)
```



Simulation-based tests

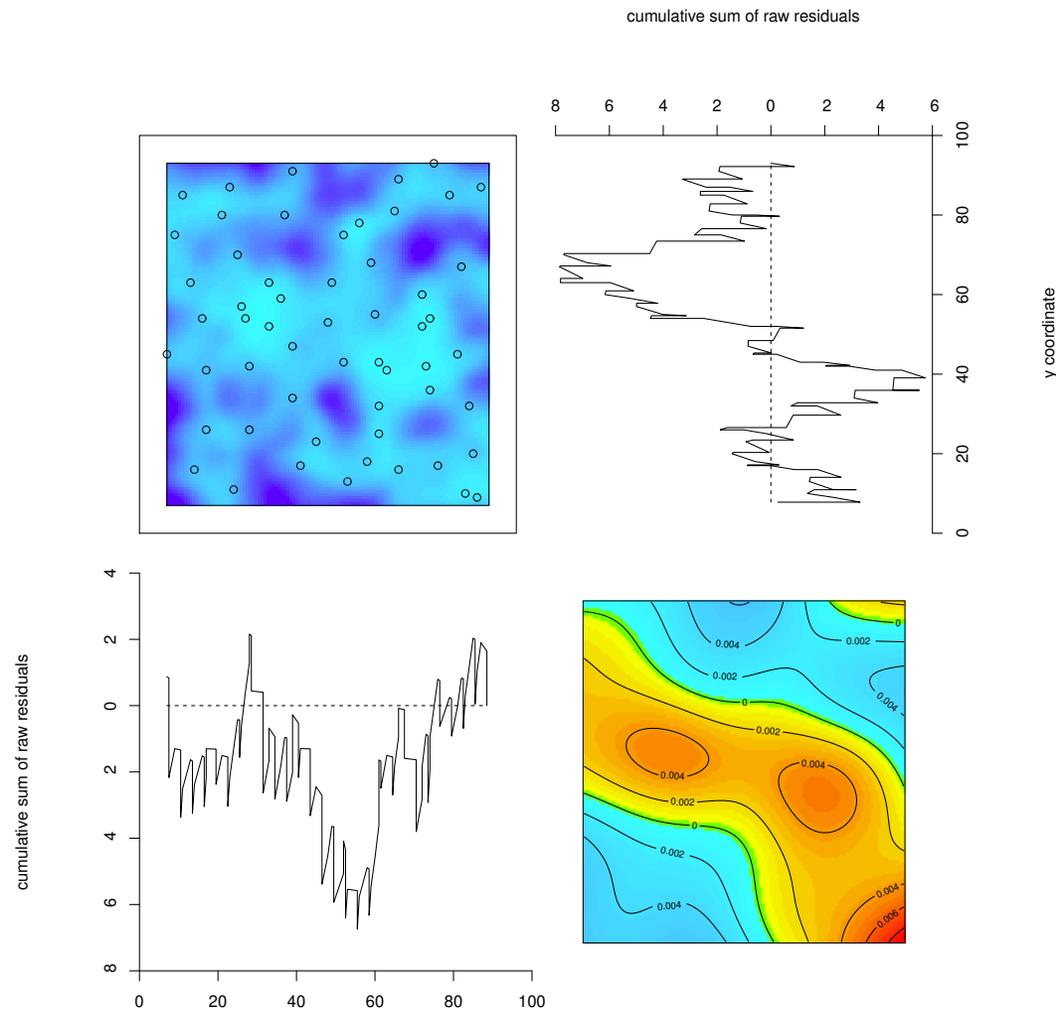
Tests of goodness-of-fit can be performed by simulating from the fitted model.

```
plot(envelope(fit, Gest, nsim=19))
```



More powerful diagnostics are available.

`diagnose.ppm(fit)`

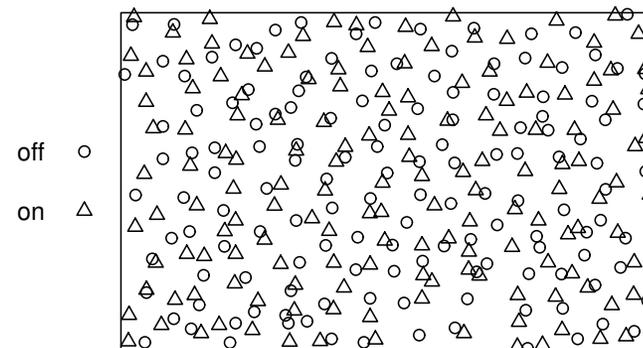
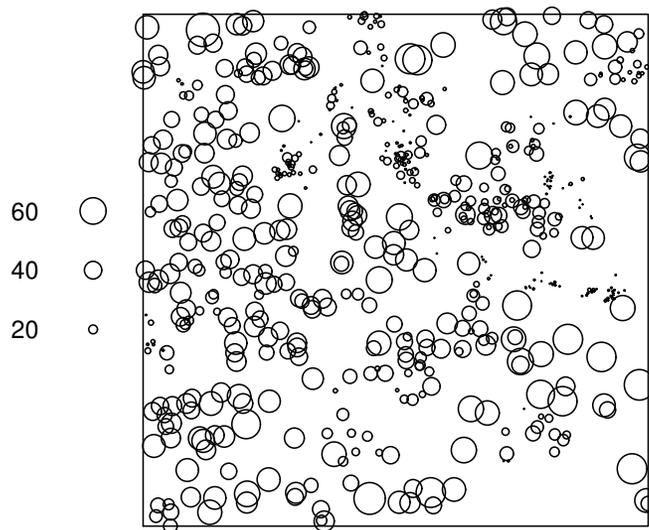


Marks

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”.

```
> data(amacrine)
```

```
> amacrine
```

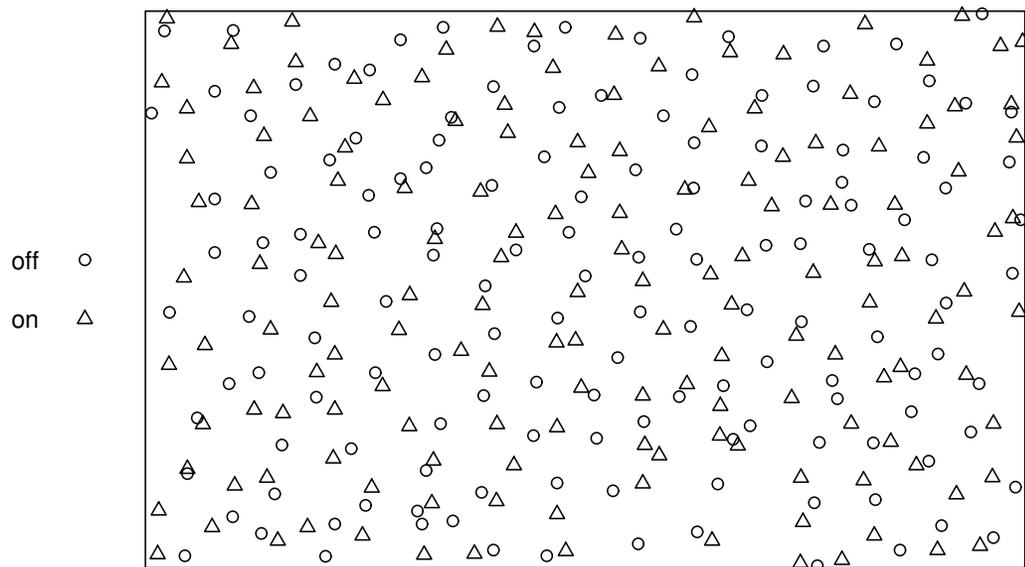
```
marked planar point pattern: 294 points
```

```
multitype, with levels = off    on
```

```
window: rectangle = [0, 1.6012] x [0, 1] units (one unit = 662 microns)
```

```
> plot(amacrine)
```

amacrine



summary(amacrine)

summary(amacrine)

Marked planar point pattern: 294 points

Average intensity 184 points per square unit (one unit = 662 microns)

Multitype:

	frequency	proportion	intensity
off	142	0.483	88.7
on	152	0.517	94.9

Window: rectangle = [0, 1.6012] x [0, 1] units

Window area = 1.60121 square units

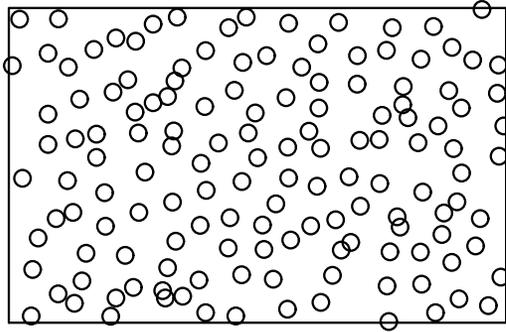
Unit of length: 662 microns

Intensity of multitype patterns

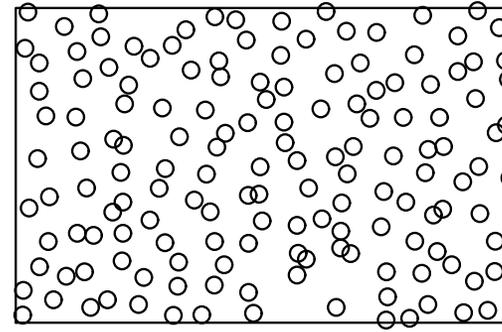
```
plot(split(amacrine))
```

split amacrine

off



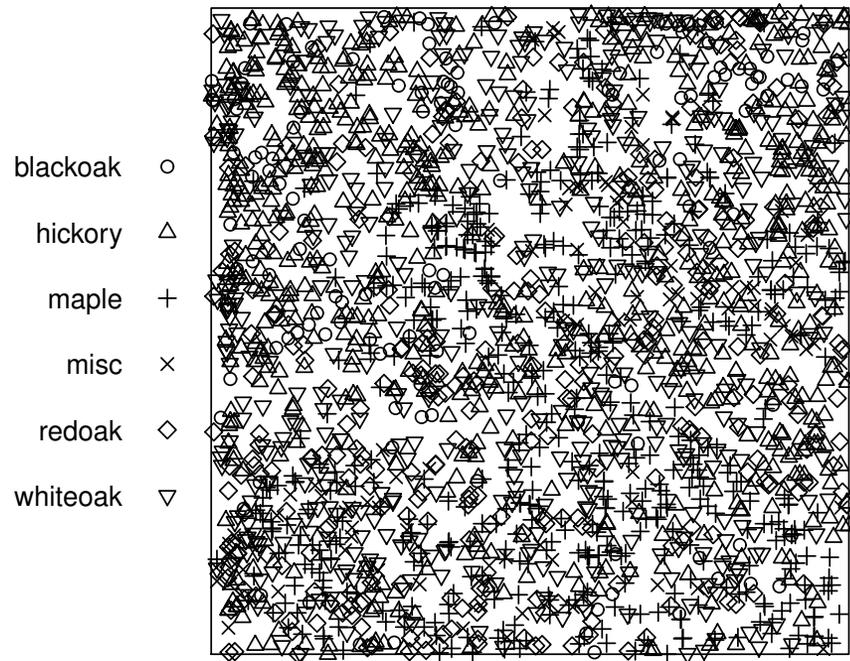
on



Intensity of multitype patterns

```
data(lansing)  
summary(lansing)  
plot(lansing)
```

lansing

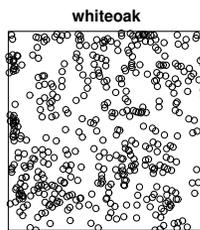
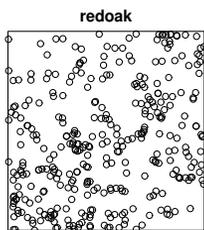
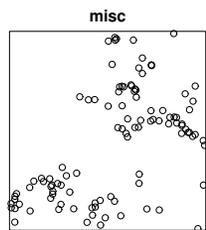
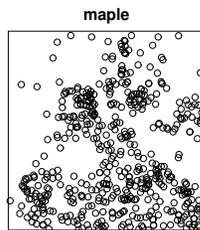
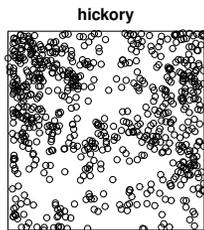
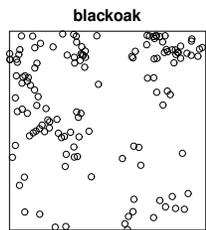


Intensity of multitype patterns

“Segregation” occurs when the intensity depends on the mark (i.e. on the type of point).

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

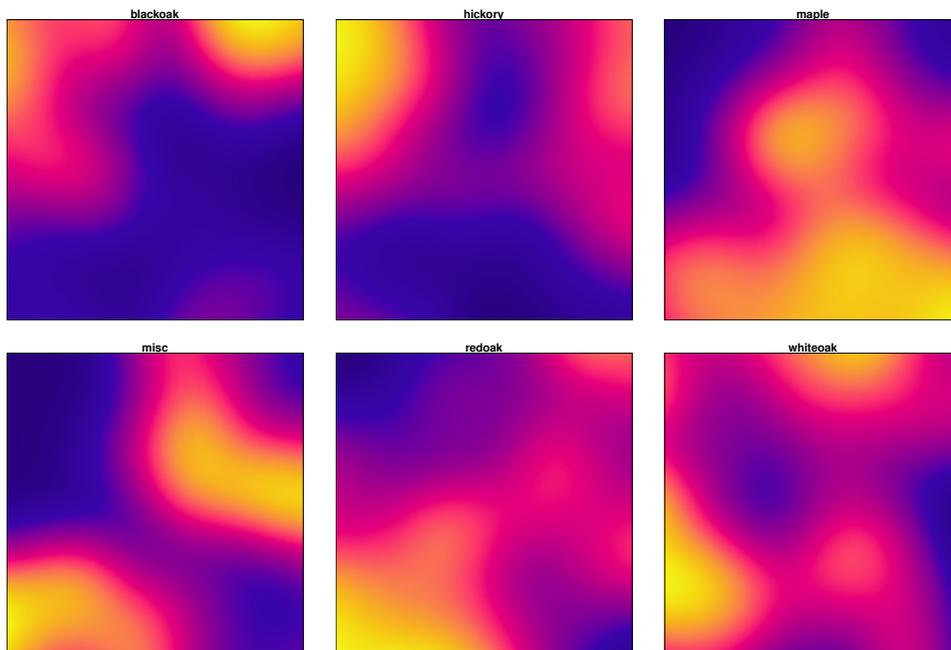
split lansing



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 .

```
plot(density(split(lansing)))
```

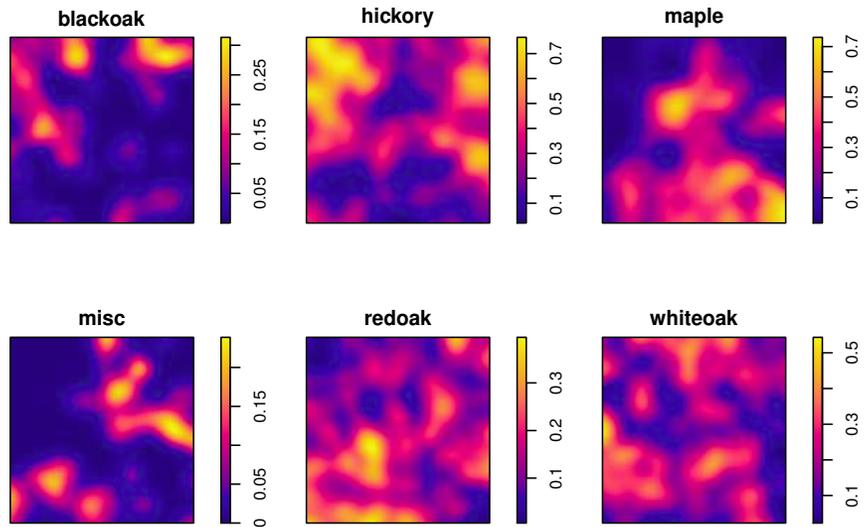


The probability that a point at location u has mark m is

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

where $\lambda(u) = \sum_m \lambda(u, m)$ is the intensity function of points of all types.

```
lansP <- relrisk(lansing)  
plot(lansP)
```

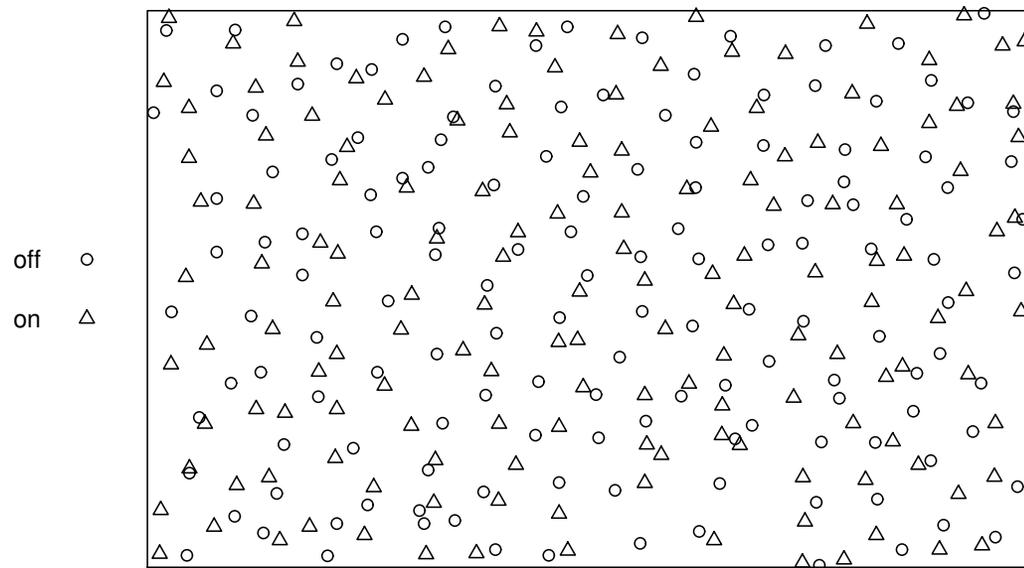


Interaction between types

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.

amacrine

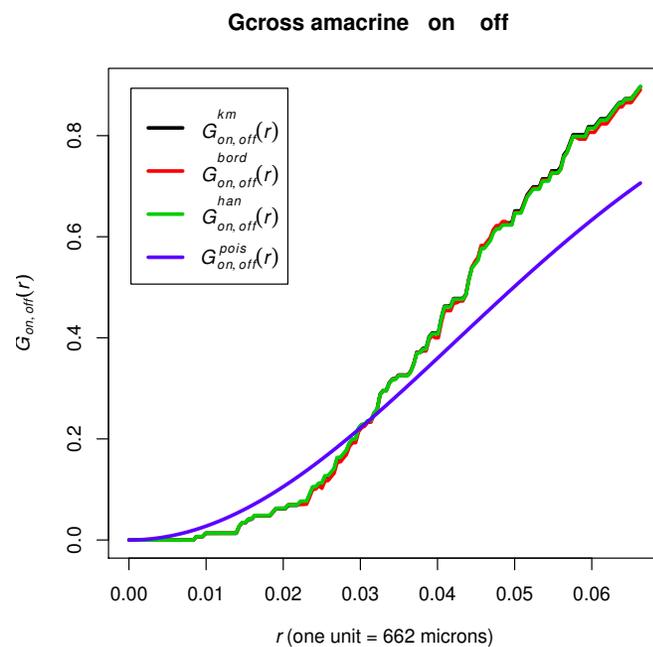


Assume the points of type i have uniform intensity λ_i , for all i .
For two given types i and j , the bivariate G -function G_{ij} is

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

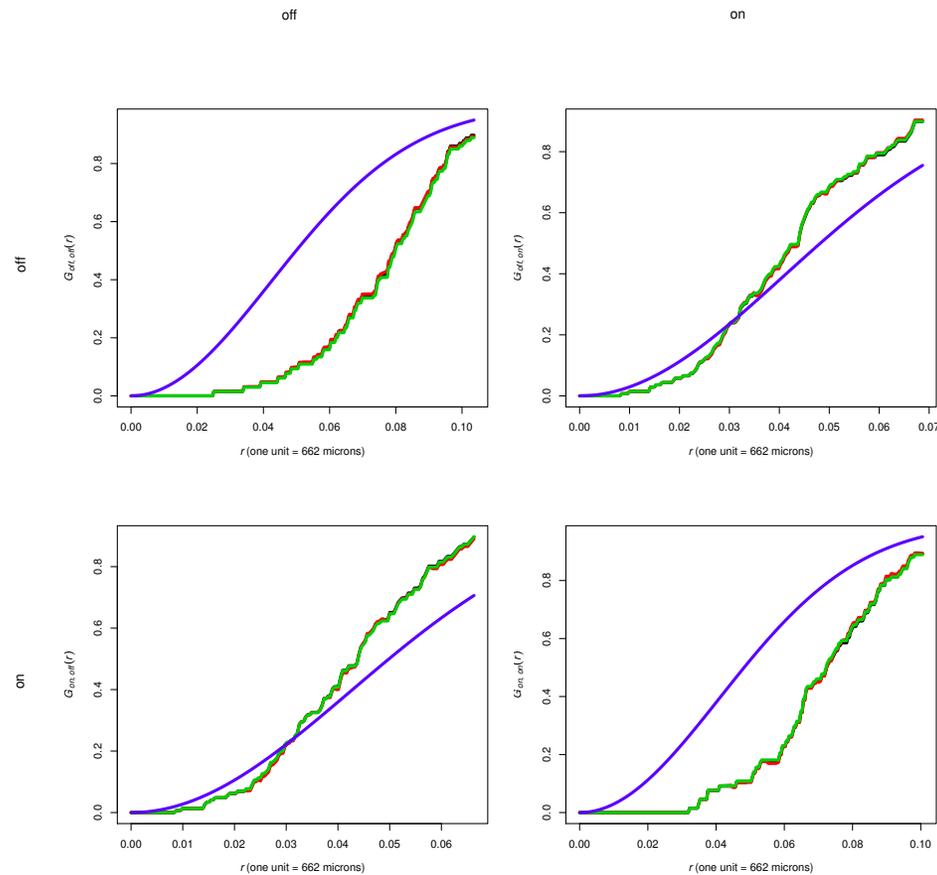
where R_{ij} is the distance from a typical point of type i to the nearest point of type j .

```
plot(Gcross(amacrine, "on", "off"))
```



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

array of Gcross functions for amacrine.



Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Different overall intensity for each type.

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Different overall intensity for each type.

`ppm(X ~ marks + x + marks:x)`

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Different overall intensity for each type.

`ppm(X ~ marks + x + marks:x)`

equivalent to

`ppm(X ~ marks * x)`

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Different overall intensity for each type.

`ppm(X ~ marks + x + marks:x)`

equivalent to

`ppm(X ~ marks * x)`

$\log \lambda((x, y), m) = \beta_m + \alpha_m x$

Fitting Poisson models

For a *multitype* point pattern:

COMMAND

INTERPRETATION

`ppm(X ~ 1)`

$\log \lambda(u, m) = \beta$ constant.

Equal intensity for points of each type.

`ppm(X ~ marks)`

$\log \lambda(u, m) = \beta_m$

Different constant intensity for points of each type.

`ppm(X ~ marks + x)`

$\log \lambda((x, y), m) = \beta_m + \alpha x$

Common spatial trend

Different overall intensity for each type.

`ppm(X ~ marks + x + marks:x)`

equivalent to

`ppm(X ~ marks * x)`

$\log \lambda((x, y), m) = \beta_m + \alpha_m x$

Different spatial trends for each type

Likelihood ratio test of segregation in Lansing Woods data:

Likelihood ratio test of segregation in Lansing Woods data:

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

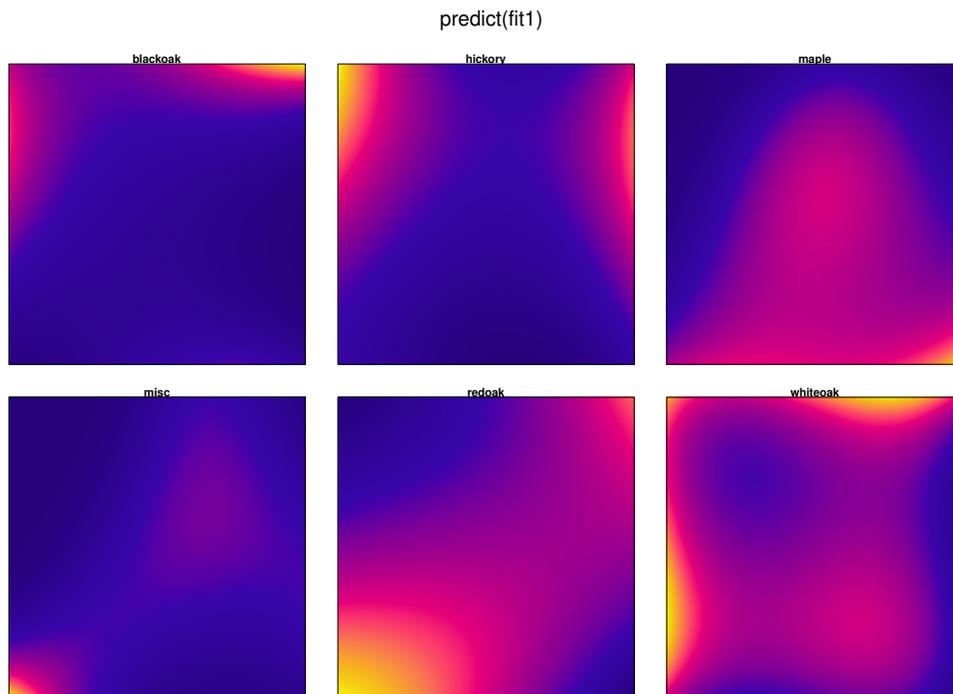
Likelihood ratio test of segregation in Lansing Woods data:

```
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

Model 1:	~marks + (x + y + I(x ²) + I(x * y) + I(y ²) + I(x ³) + I(x ² * y) + I(x * y ²) + I(y ³))	Poisson
Model 2:	~marks * (x + y + I(x ²) + I(x * y) + I(y ²) + I(x ³) + I(x ² * y) + I(x * y ²) + I(y ³))	Poisson
	Npar Df Deviance Pr(>Chi)	
1	15	
2	60 45 612.57 < 2.2e-16 ***	

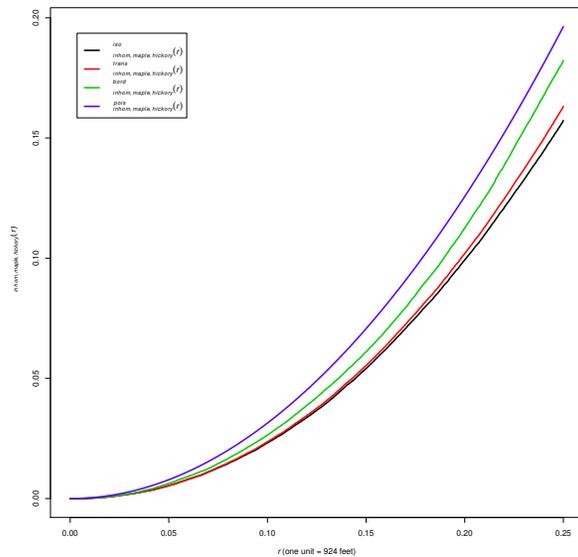
```
fit1 <- ppm(lansing ~marks * polynom(x,y,3))  
plot(predict(fit1))
```



Inhomogeneous multitype K function

Inhomogeneous K function can be generalised to inhomogeneous multitype K function.

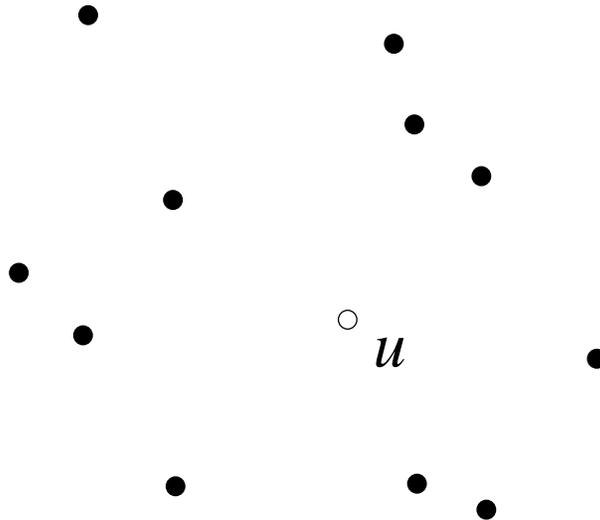
```
fit1 <- ppm(lansing ~marks * polynom(x,y,3))  
lamb <- predict(fit1)  
plot(Kcross.inhom(lansing, "maple", "hickory",  
  lamb$markmaple, lamb$markhickory))
```



Multitype Gibbs models

Conditional intensity

The conditional intensity $\lambda(u, m \mid \mathbf{x})$ is essentially the conditional probability of finding a point of type m at location u , given complete information about the rest of the process \mathbf{x} .



Multitype Strauss process

```
> 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
```

Multitype Strauss process

```
> 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`