Topic 2: Spatial Point Processes

This topic introduces a new spatial model. We will define and investigate spatial point processes. In particular, we consider

- Introduction to **spatial point processes**.
- Investigating **intensity** in R.
- **Covariates** in spatial analysis.
- **Class of ppp objects.**
- **Converting between ppp and sp formats.**
- Creating **ppp objects** from csv file.

Spatial Point Processes

Spatial point processes are mathematical models for irregular or random point patterns. The points could represent trees, animal nests, earthquake epicenters, petty crimes, domiciles of new cases of influenza, galaxies, etc. The points might be situated in a region of the two-dimensional (2D) plane, or on the Earths surface, or a 3D volume, etc. They could be points in space-time (e.g. earthquake epicenter location and time).

> Aa point process on R^d is mathematically defined as a random variable X taking values in the space $\mathfrak{X}$, where $\mathfrak{X}$ is the family of all sequences $\{x_n\}$ of points of R^d satisfying the local finiteness condition, which means that each bounded subset of R^d contains only a finite number of points.

We consider only **simple spatial point processes**, i.e. $x_i \neq x_j$ if $i \neq j$.

Intensity

The **intensity measure** Λ of X is a characteristic analogous to the mean of a real-valued random variable. Its definition is

$$\Lambda(B) = E(X(B)) \quad \text{for} \quad B \in R^n.$$

Here $X(B)$ is the number of points of X in B. So $\Lambda(B)$ is the mean number of points in B.

If the measure $\Lambda(B)$ has a density then it can be written as

$$\boxed{\Lambda(B) = \int_B \lambda(x)dx.}$$

The density $\lambda(x)$ is called the **intensity function of the point process**.

A point process X is said to be **stationary** if its characteristics are invariant under translation: the processes $X = \{x_n\}$ and $X_a = \{x_n + a\}$ have the same distribution for all $a \in R^d$.

If X is stationary then the intensity measure simplifies; it is a multiple of d-dimensional volume $V(B)$, i.e.

$$\boxed{\Lambda(B) = \lambda \cdot V(B)}$$

for some non-negative constant λ, which is called the intensity of X. It can be interpreted as the mean number of points of X per unit volume. In this case the intensity function $\lambda(x) = \lambda$.

Let the point pattern dataset consist of n points $x_1, ..., x_n$ in a bounded spatial region $W \subset R^d$.
If we know that a point process is stationary, then the **empirical density of points**,

$$\boxed{\hat{\lambda} = \frac{X(W)}{V(W)} = \frac{n}{V(W)}}$$

is an unbiased estimator of the true intensity λ.

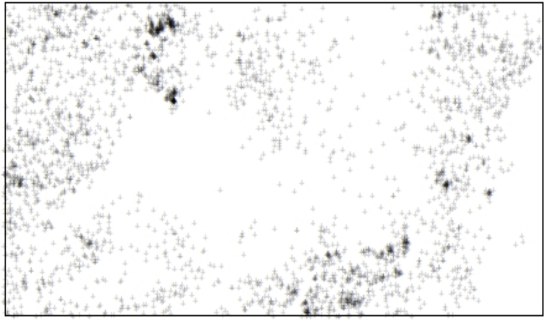
Investigating intensity in spatstat

The **dataset bei** gives the positions of 3605 trees of the species Beilschmiedia pendula (Lauraceae) in a 1000 by 500 metre rectangular sampling region in the tropical rainforest of Barro Colorado Island.
To compute the estimator $\hat{\lambda}$ in **spatstat,** use SUMMARY:

```
> library(spatstat)
> data(bei)
> summary(bei)
Planar point pattern:  3604 points
Average intensity 0.007208 points per square metre

Coordinates are given to 1 decimal place
i.e. rounded to the nearest multiple of 0.1 metres
Window: rectangle = [0, 1000] x [0, 500] metres
Window area = 5e+05 square metres
Unit of length: 1 metre
```

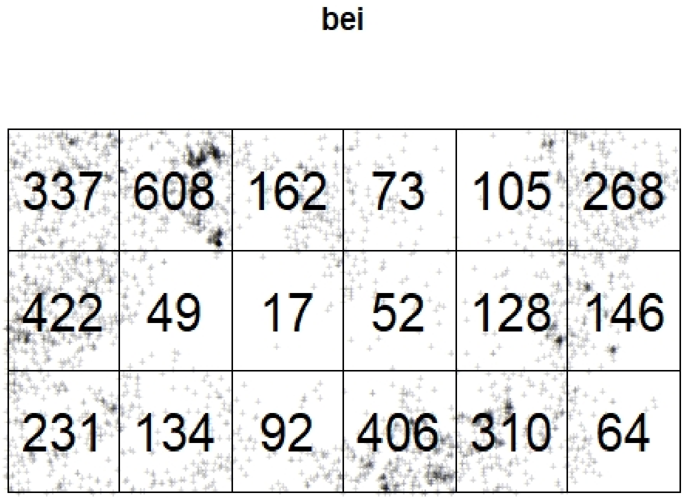
```
> lamb <- summary(bei)$intensity
> lamb
[1] 0.007208
> plot(bei, cex = 0.5, pch = "+")
```



If it is suspected that the intensity may be nonstationary, the intensity function or intensity measure can be estimated nonparametrically by techniques such as **quadrat counting** and **kernel smoothing**.

In **quadrat counting**, the window W is divided into subregions ("quadrats") $B_1, ..., B_m$ of equal area. We count the numbers of points falling in each quadrat, $n_j = X(B_j)$ for $j = 1, ..., m$. These are unbiased estimators of the corresponding intensity measure values $\Lambda(B_j)$.
In R quadrat counting is performed by

```
> quadratcount(bei, nx = 6, ny = 3)
xy             [0,167) [167,333) [333,500) [500,667)
[333,500]       337       608       162        73
[167,333)       422        49        17        52
[0,167)         231       134        92       406
...
> Q <- quadratcount(bei, nx = 6, ny = 3)
> plot(bei, cex = 0.5, pch = "+")
> plot(Q, add = TRUE, cex = 2)
```


bei
337 608 162 73 105 268
422 49 17 52 128 146
231 134 92 406 310 64

The **kernel estimator** of the intensity function is

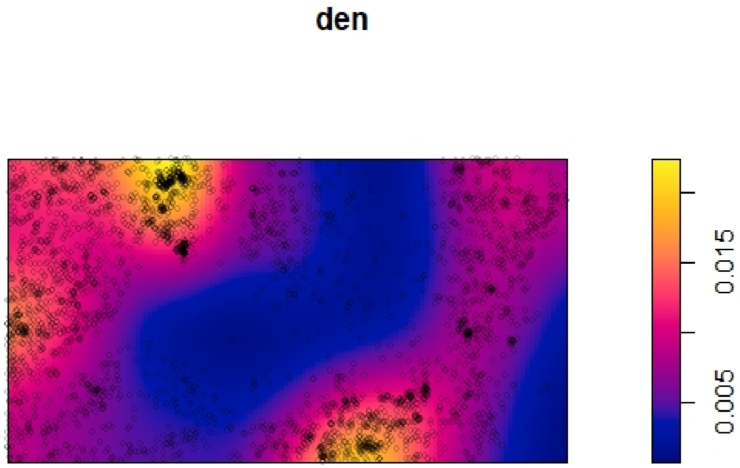
$$\boxed{\hat{\lambda}(u) = \frac{1}{||K(u)||} \sum_{i=1}^{n} K(u - x_i),}$$

where $K(u)$ is the kernel (an arbitrary probability density) and

$$||K(u)|| = \int_W K(u - v) dv.$$

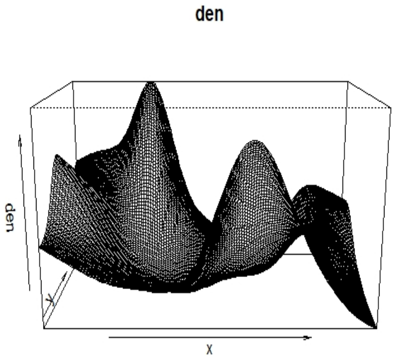
Intensity estimation using an isotropic Gaussian kernel is implemented by

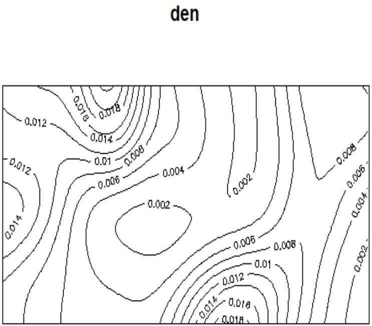
```
> den <- density(bei, sigma = 70)
> plot(den)
> plot(bei, add = TRUE, cex = 0.5)
```


den
0.005
0.015

Perspective and contour plots can be displayed by

```
> persp(den)
> contour(den)
```





Covariates

Datasets may also include **covariates**, i.e. any data that are treated as explanatory, rather than as part of the response.

Covariates data can be described by a spatial function $Z(u)$ defined at all spatial locations u, e.g. altitude, pollution level, soil pH, etc.

In quadrat counting methods, any choice of quadrats is permissible. From a theoretical viewpoint, the quadrats do not have to be rectangles of equal area, and could be regions of any shape.

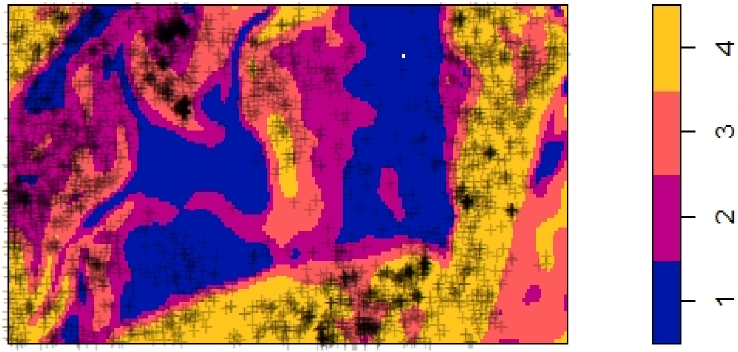
Quadrat counting is more useful if we choose the quadrats in a meaningful way. One way to do this is to define the quadrats using covariate information.

The tropical rainforest point pattern dataset BEI comes with an extra set of covariate data BEI.EXTRA, which contains a pixel image of terrain elevation BEI.EXTRA$ELEV and a pixel image of terrain slope BEI.EXTRA$GRAD.

It might be useful to split the study region into several sub-regions according to the terrain slope.

For example, to divide the study region into four zones of equal area according to the terrain slope, we use the quartiles of the slope values:

```
> Z <- bei.extra$grad
> b <- quantile(Z, probs = (0:4)/4)
> Zcut <- cut(Z, breaks = b, labels = 1:4)
> V <- tess(image = Zcut)
> plot(V)
> plot(bei, add = TRUE, pch = "+")
```

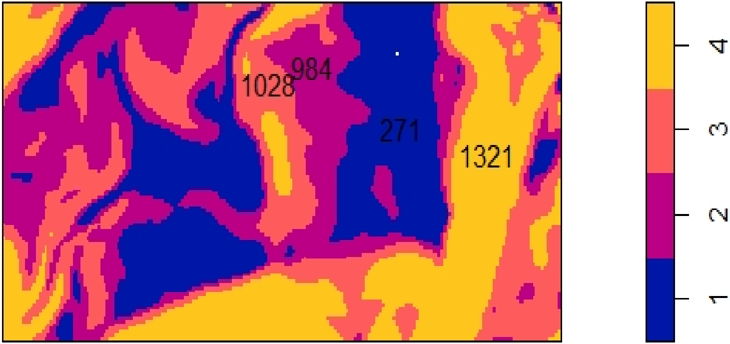

4
3
2
1

We can find the number of trees in each region by

```
> qb <- quadratcount(bei, tess = V)
> qb
tile
1    2    3    4
271  984 1028 1321
> plot(qb)
```

Since the four regions have equal area, the counts should be approximately equal if there is a uniform density of trees. Obviously they are not equal.

There appears to be a strong preference for steeper slopes.


1028
984
271
1321
4
3
2
1

Format of ppp objects

A point pattern is represented in SPATSTAT by the **class "ppp"**. It contains the coordinates of the points, optional marks values attached to the points, and a description of the study region or spatial window.

A point pattern object P has the following components:

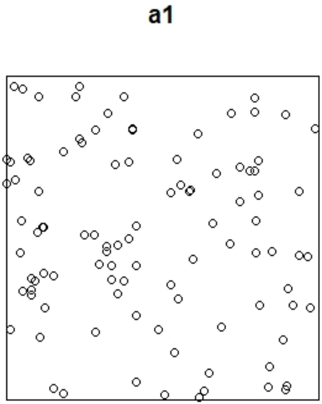
- $P\$n$ is the number of points (which may be zero).
- $P\$x$ is a numeric vector containing the x coordinates of the points. Its length equals $P\$n$ (and may be zero).
- $P\$y$ is a numeric vector containing the y coordinates of the points. Its length also equals $P\$n$.
- $P\$marks$ contains the marks. It is either NULL, or a vector of length $P\$n$ containing the mark values. The entries of $P\$marks$ may be of any atomic type (character, numeric,...).
- $P\$window$ is an object of class "OWIN" (observation window) determining the study region or spatial window.

Creating datasets

In SPATSTAT, each point in a point pattern can be marked with a single value (i.e. one mark value per point). The marks are stored in a vector, of the same length as the number of points. The marks can be of any atomic type: numeric, integer, character, factor, logical or complex.

Marks can be attached to an existing point pattern X using the function **marks()** or using the binary operator **%mark%.** These are convenient when you want to assign new marks to a dataset that are computed using another variable.

```
> library(spatstat)
> a1<-runifpoint(100)
> a1
Planar point pattern: 100 points
window: rectangle = [0, 1] x [0, 1] units
> plot(a1)
```

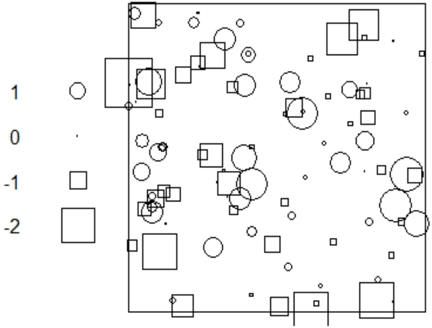



```
> m1<-rnorm(100)
> marks(a1)<-m1
> a1
```

```
Marked planar point pattern: 100 points
marks are numeric, of storage type  double
window: rectangle = [0, 1] x [0, 1] units
> Y1<-a1 %mark% m1
> summary(Y1)
Marked planar point pattern:  100 points
Average intensity 100 points per square unit
Coordinates are given to 8 decimal places
marks are numeric, of type double
Summary:
Min.  1st Qu.   Median     Mean  3rd Qu.
-2.06510 -0.58418 -0.06890  0.07298  0.77116
Max.
2.61851

Window: rectangle = [0, 1] x [0, 1] units
Window area = 1 square unit

> plot(Y1)
```



The marks can be extracted using the function MARKS:

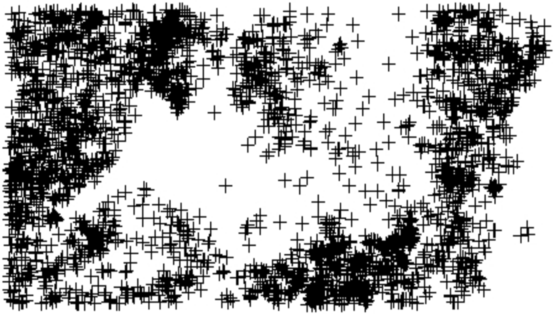
```
> m<-marks(Y1)
> m
[1]  2.42441833  0.82120049 -0.20682970
...
```

Converting to/from sp formats

To convert SPATSTAT point pattern dataset BEI to an object of class "SpatialPoints" use:

```
> library(sp)
> library(maptools)
> data(bei)
> bb<-as(bei,"SpatialPoints")
> summary(bb)
Object of class SpatialPoints
Coordinates:
min  max
[1,]    0 1000
[2,]    0  500
Is projected: NA
proj4string : [NA]
Number of points: 3604
```

```
> plot(bb)
```



To convert it in the opposite direction to the class "ppp" one can use:

```
> bb1<-as(bb,"ppp")
> summary(bb1)
Planar point pattern:  3604 points
Average intensity 0.007208 points per square unit

Coordinates are given to 1 decimal place
i.e. rounded to the nearest multiple of 0.1 units

Window: rectangle = [0, 1000] x [0, 500] units
Window area = 5e+05 square units

> plot(bb1)
```

Creating spatstat dataset from csv file

As an example let us use the dataset worldcities.csv. First, read itinto R using read.csv:

```
> mydata <- read.csv("worldcities.csv")
> mydata <- mydata[mydata$country == "Australia",]
> head(mydata)
```

You can see that the data are saved as a data frame.

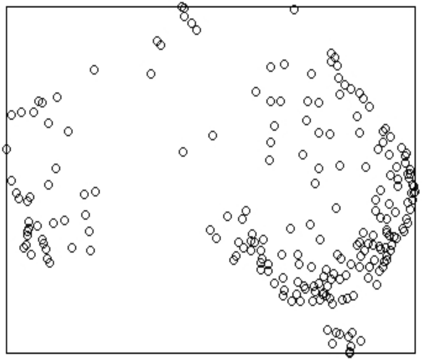
```
> str(mydata)
```

To convert the data from a data frame we first find the range of longitudes and latitudes for the points

```
> range(mydata$lng)
[1] 113.6501 153.6129
> range(mydata$lat)
[1] -42.9911 -12.4254
```

Then we created a point pattern object by using the command

```
> myppp <- ppp(mydata$lng, mydata$lat, c(113.6501, 153.6129),
+ c(-42.9911, -12.4254))
> plot(myppp)
```



Key R commands

quadratcount(X,nx,ny)	*divides window into quadrats and counts the numbers of points in each quadrat*
density(x,...)	*computes kernel density estimates*
persp(x, ...)	*draws perspective plots of a surface*
contour(x)	*creates a contour plot*
tess(image)	*creates a tessellation of a spatial region*
runifpoint(n)	*generates a random point pattern with n independent uniform random points*
marks(x)	*extracts or changes the marks*
ppp(x,y,,...)	*creates an object of class "ppp"*
bei	*data withlocations of 3605 trees in a tropical rain forest*