Lab Exercise #3

Please install the **spatstat** package
> install.packages("spatstat")
Once the package is installed, to use its functions, you need to load it by typing:
> library(spatstat)

Part 1: Simulation of homogeneous Poisson process
To make a point pattern using one of these processes, type:
> pp <-rpoispp(100) # CSR with intensity 100 in the unit square
and then to see the pattern:
> plot(pp)
You can combine generating and plotting a pattern into a single command like this:
> plot(rpoispp(100))
although it may be more sensible to create the pattern first then plot it, since this allows you to make patterns in memory and keep the interesting ones, like this:
> p1 <-rpoispp(100)
> plot(p1)
then use the up-arrow to repeatedly generate p1 and plot it, until you get an interesting realization of the process. Once you have an interesting one, you can then move on to
> p2 <- rpoispp(100)
and so on. If you need more than one graphic display output to compare patterns, then use
> x11()
to open a new window. You need to click back in the console window after this to type further commands. Also, now you have more than one graphic output window, you may need to switch between them. To do this, note the window device number in the title bar, and use
> dev.set(which=2)
where the value you set 'which' to is the device number. The next plot command will then write the active window.
An alternative to many graphic output windows is to create a grid of plots in a single window. Here's how:
> par(mfrow=c(2, 3))
this sets the current display window to expect two rows, by three columns of plots, so if you then type:
> plot(p1)
> plot(p2)
and so on to
> plot(p6)
these will appear in a 2 row by three column array in the active output window.

Q1. Take a screenshot of the above 6 figures. Do all these IRP/CSR patterns appear 'random'? If not, how do they seem to differ from 'random' patterns? (10points)

Q2. Do some exhibit clustering of events? Or, conversely, empty regions of the map? Provide an example from your explorations, and explain how it is possible for such

seemingly 'non-random' patterns to be generated by a completely random process. (10points)

Now, generate a point pattern with intensity 1 in a 10 x 10 square
>pp.1.10 <- rpoispp(1, win=owin(c(0,10),c(0,10)))
>par(mfrow=c(1,2))

>plot(rpoispp(100),main = "intensity = 100, unitsquare") # # uniform Poisson process with intensity 100 in the unit square
>plot (pp.1.10, main = "intensity = 1, 10 x 10 square")

plots should look similar

Q3. Redo exactly the same commands and see the different realizations of the same process under CSR. Take a screenshot of the first 3 realizations. (10points)

Q4. Change the intensity parameter to 10,20, and 50 in the unit square and see what happens. Take a screenshot of your figures. (10points)

Q5. When the intensity is 10 in the unit square, does each simulation show exactly 10 points in the area? why not? (10points)

Part 2: Simulation of inhomogeneous Poisson process

This is a spatial point process which generates events based on an underlying intensity that varies from place to place. In other words, it departs from IRP due to a *first-order trend* across the study area.
To introduce a trend, we have to define a function in place of the fixed intensity value of the standard Poisson process. When you typed:
> pp <- rpoispp(100)
the 100 specified a fixed intensity across the whole study area. We can instead specify a function for the intensity:
> pp <- rpoispp(function(x,y) {100*x + 100*y})
This tells R that the intensity of the Poisson point process varies with the x and y coordinates according to the function specified. Make a few patterns based on this intensity surface and have a look at them. You might also want to plot the density of points in each realization:
> plot(density(pp))
> plot(pp, add=T)
The first command here plots the density as a raster grid, while the second command plots the point pattern itself on top, with the 'add=T' parameter telling R to add the point pattern on top of the existing density surface ('T' signifies 'True'). Another aid to visualizing the density of each point process realization is using contours:
> plot(density(pp))
> contour(density(pp), add=T)
> plot(pp, add=T)

Q6. Does this pattern appear random? How does it differ from the pure random patterns you generated in the previous section, if at all? Can you tell the difference every time, just by visual inspection? (10points)

Now change the intensity function
with intensity lambda(x,y) = 100 * exp(-3*x)
Intensity is bounded by 100
>pp <- rpoispp(function(x,y) {100 * exp(-3*x)}, 100)

Q7. Redo the above statement 10 times and plot each realization (including density and point pattern on top). Take a screenshot of your figures. Are these 10 realizations the same? If not, how do they differ? (10points)

Part 3: Regular pattern point processes
To see the range of spatial processes available in **spatstat**, type
> help(spatstat)
and navigate down to the 'To simulate a random point pattern' section. You will see a selection of spatial processes to explore. Processes that can produce evenly-spaced patterns
are rMaternI(), rMaternII() and rSSI(), while the rMatClust() and rThomas() processes can produce clustered patterns.
In all cases, there are various types of interaction between events in the pattern, so all these processes introduce *second-order* interaction effects. Let's try:
> pp <- rSSI(0.05, 100)
This is the Simple Sequential Inhibition process. The first parameter (here 0.05) is the *inhibition distance*. Events are randomly located in the study area (which by default is a 1-by-1 square, with x and y coordinates both in the range 0 to 1), but, if an event is closer to an already existing event than the inhibition distance, it is discarded. This process continues until the required number of events (the second parameter, here equal to 100) have been placed.

Q8. Redo the above statement 6 times and plot each realization. Take a screenshot of your figures. (10points)

Q9. Change the inhibition distance to 0.08 and number of events to 50. Generate 6 realizations and take a screenshot of your figures. (10points)

Part 4: Clustering pattern point processes
By contrast, a clustering process, such as the Thomas process:
> pp <- rThomas(10, 0.1, 10)
Here the first parameter specifies the intensity of a homogeneous Poisson process (i.e., IRP/CSR) which produces cluster centers. The second parameter specifies the size of clusters as the standard deviation of a normally distributed distance from each cluster center. This means that events are most likely to occur close to cluster centers, with around 95% falling within twice this distance of centers, and very few falling more than 3 times this distance from cluster centers. Finally, the third parameter specifies the expected number of events in each cluster.

Q10. Redo the above statement 6 times and plot each realization. Take a screenshot of your figures. (10points)

To save your work in R, make sure that when you exit the program you OK the option to save the workspace image. You can also save the data at any time using File - Save workspace... option and also the commands you have used with the File - Save history... option. These will ask you to specify .Rdata or .Rhistory filenames and these can be retrieved at a later time by loading them from the same menu. Neither of these options will save any plots you make. If you want that capability, you should consider using the RStudio program to drive R.

Your answers should be in a Word document with each answer numbered. The format for naming your labs is:

lastname_ lab#.doc ie Smith_ Lab3.doc