Using Spatstat for Distance-based Point Pattern Analysis

1. Let's look at nearest neighbor analysis using spatstat first.

2. Again, read the swedishpines ppp into the object X

```
> X <- swedishpines
```

3. You can find pairwise distances between all points in X using the pairdist() function

```
> xdistmatrix <- pairdist(X)
```

4. And check the resulting object is a matrix of dimension 71x71 of inter-point distances

```
> str(xdistmatrix)  num [1:71, 1:71] 0 27 37 15 54.3 ...
```

5. Find the nearest neighbor distances for each point directly

```
> xnndist <- nndist(X) > str(xnndist)  num
```

[1:71] 15.03 8.54 10.05 9.06 10.77 ...

6. These nearest neighbor distances can then be used in your test statistic to perform the nearest neighbor test.

```
> dbarobs <-sum(xnndist)/length(xnndist)
> dbarobs
```

[1] 7.907541

7. We know from the summary(X) function, that the area of the swedishpines point pattern covers 9600 (0.1 unit meters) and includes n=71 points. This allows us to find the expected value of the nearest neighbor index

```
> dbarexp <- 0.5/sqrt(71/9600)
> dbarexp
```

[1] 5.814019

8. And now we just have to find the standard deviation of the dbar expected value

```
> sddbarexp <- .26136/sqrt((71^2)/9600)
> sddbarexp
```

[1] 0.3606753

9. Our nearest neighbor index value R = 7.908/5.814 = 1.36 which looks close to the R ~ 1 value consistent with an IRP.

10. However, we have to test the null hypothesis that the nearest neighbor distance is consistent with an IRP. The Z-test statistic for our nearest neighbor analysis of the swedishpines data is

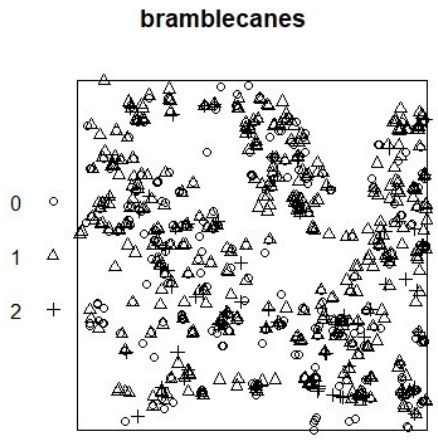
$$Z^* = \frac{\bar{d}_{obs} - \bar{d}_{exp}}{\sigma_{\bar{d}}} = 5.801$$

11. This value suggests that at any usual significance level we would reject the null hypothesis. With dbarobs >dbarexp, this suggests a dispersed point pattern. (Note that nearest neighbor analysis is very sensitive to edge effects that are not controlled for here. These would be points located just outside the study region which might be nearest neighbors to points inside the study region.)

**NB Using the K-function for the swedishpines data suggest some evidence of clustering at small values and then a basically random pattern. This is quite inconsistent with the nearest neighbor results above!)

12. Now let us turn to think about the K function. And let us develop the K function for the bramblecane data examined in Lecture 5. Let's read in the data

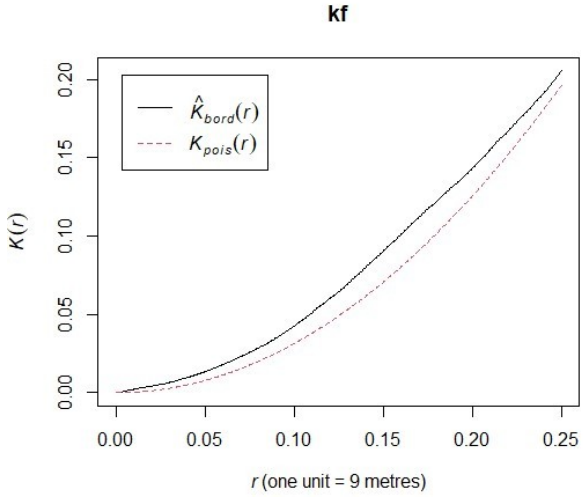
```
> data(bramblecanes)
> X <- bramblecanes
> plot(X)
```



13. Now let us find the K function for the bramblecanes data

```
> kf <-Kest(bramblecanes, correction='border')
> plot(kf)
```

Note that edge corrections should be used with the K function. The simplest to use is the correction='border' method that eliminates some points near the study region border to compute the observed K function. Better for rectangular study regions is the correction='Ripley'. There are others with different properties. Check the help function.

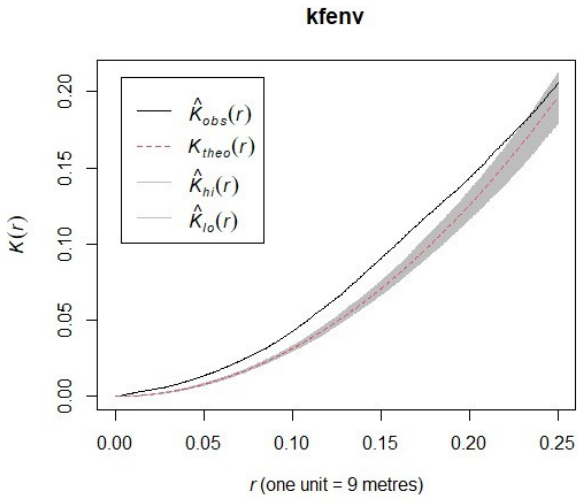


This plot shows the observed K function *K*and the expected value of the K function consistent with an IRP generated using the Poisson distribution (K_{pois}). (K_{bord} means K with border correction.) With the value of the observed function well above that of the expected function, this is evidence of a clustered point pattern at most distances (here distance is given as *r* rather than *d*). It is evidence of a clustered pattern because we observe more points around events for most distance bands than we expect.

Note from lecture 5 that when we sample from the population of all events in a study region, some sampling error might be responsible for deviations between observed and expected K functions even if the observed function was generated by an independent random process. We can correct for this possibility by fitting an envelope (sort of like a confidence interval) around the expected K function. The function is fit by randomly permuting the locations of the input data. When the sample size if high the envelope will be relatively narrow and vice versa. If our observed K function lies outside the envelope, then we have more evidence that departures from randomness are not simple the result of sampling error. To fit the envelope function

```
> kfenv <- envelope(bramblecanes, Kest,correction='border')
> plot(kfenv)
```

The plot below shows that for all except the highest distance values, the observed K function lies outside the envelop providing more evidence of a clustered point pattern.



14. And, finally, we can perform a hypothesis test of the difference between the observed and expected K functions using the maximum absolute deviation (MAD) statistic. This is generated within R as

> mad.test(bramblecanes, Kest)

Generating 99 simulations of CSR ...
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74,
75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99.

Done.

Maximum absolute deviation test of CSR
Monte Carlo test based on 99 simulations
Summary function: K(r)
Reference function: theoretical
Alternative: two.sided
Interval of distance values: [0, 0.25] units (one unit = 9 metres)
Test statistic: Maximum absolute deviation
Deviation = observed minus theoretical

data: bramblecanes
mad = 0.016159, rank = 1, p-value = 0.01 – suggesting we would reject the null hypothesis for 0.01