4. K-Function Analysis of Point Patterns

In the Bodmin Tors example above, notice from Figure 3.14a (p.20) that the clustering structure is actually quite different from that of the Redwood Seedling example in Figure 3.12a (p.12). Rather than small isolated clumps, there appear to be two large groups of points in the northwest and southwest, separated by a large empty region. Moreover, the points within each group are actually quite evenly spaced (locally dispersed). These observations suggest that the pattern of tors exhibits different structures *at different scales*. Hence the objective of the present section is to introduce a method of point pattern analysis that takes such scale effects into account, and in fact allows "scale" to become a fundamental variable in the analysis.

4.1 Wolf-Pack Example

To motivate the main ideas, we begin with a new example involving wolf packs. A map is shown in Figure 4.1a below representing the relative locations of wolf packs in a portion of the Central Arctic Region in 1998.[1] The enlarged portion in Figure 4.1b is a schematic map depicting individual wolves in four of these packs.

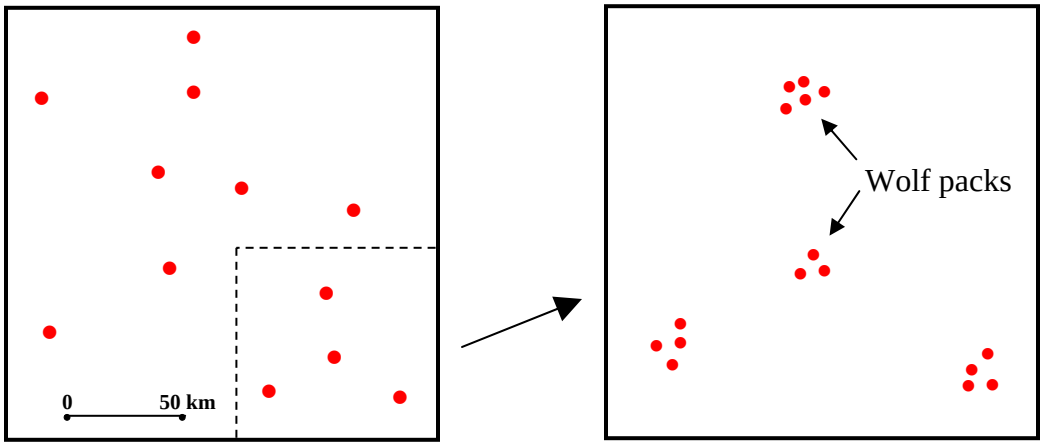

Fig.4.1a. Map of Wolf Packs **Fig.4.1b. Enlarged Portion**

At the level of individual wolf locations in Figure 4.1b, there is a pattern of isolated clumps that bears a strong resemblance to that of the Redwood seedlings above.[2] Needless to say, this pattern would qualify as strongly *clustered*. But if one considers the larger map in Figure 4.1a, a different picture emerges. Here, the dominant feature is the remarkable *dispersion* of wolf packs. Each pack establishes a hunting territory large enough for its survival (roughly 15 to 20 km in diameter), and actively discourages other

[1] This map is based on a more detailed map published in the *Northwest Territories Wolf Notes*, Winter 1998/99. See also http://www.nwtwildlife.rwed.gov.nt.ca/Publications/wolfnotes/wolf32.htm.
[2] The spacing of individual wolves is of course exaggerated to allow a representation at this scale.

packs from invading its territory.[3] Hence this pattern of wolf locations is very clustered at *small scales*, and yet very dispersed at *large scales*.

But if one were to analyze this wolf-location pattern using any of the nearest-neighbor techniques above, it is clear that only the *small-scale clustering* would be detected. Since each wolf is necessarily close to other wolves in the same dens, the spacing between dens would never be observed. In this simple example one could of course redefine wolf dens to be aggregate "points", and analyze the spacing between these aggregates at a larger scale. But there is no way to analyze multiple scales using nearest neighbors without some form of re-aggregation.[4]

4.2 K-Function Representations

To capture a range of scales in a more systematic way, we now consider what amounts to an extension of the quadrat (or cell-count) method discussed in section 1 above. In particular, recall that the quadrat method was criticized for being too dependent on the scale of individual cells. Hence the key idea of K-functions is to turn this dependency into a virtue by explicitly incorporating "scale" as a variable in the analysis. Thus, rather than fixing the scale and locations of cell grids, we now consider *randomly sampled* cells of *varying sizes*. While many sampling schemes of this type can be defined, we shall focus on the single most basic scheme which is designed to answer the following question for a given point process with density λ: *What is the expected number of point events within distance* h *from any randomly sampled point event*? Note that this expected number is not very meaningful without specifying the point density, λ, since it will of course increase with λ. Hence if we divide by λ in order to eliminate this obvious "density effect" then the quantities of interest take the form:

(4.2.1) $K(h) = \frac{1}{\lambda} E(\text{number of } \textit{additional} \text{ events within distance, } h, \text{ of an arbitrary event})$

If we allow the distance or *scale*, h, to vary then expression (4.2.1) is seen to define a function of h, designated as a *K-function*.[5] As with nn-distances, these values, $K(h)$, yield information about clustering and dispersion. In the wolf-pack example above, if one were to define $K(h)$ with respect to small distances, h, around each wolf in Figure 4.1b, then given the close proximity to other wolves in the same pack, these values would surely be *too high* to be consistent with CSR for the given density of wolves in this area. Similarly, if one were to define $K(h)$ with respect to much larger distances, h, around each wolf in Figure 4.1a, then given the wide spacing between wolf packs (and the relative uniformity of wolf-pack sizes[6]), these values would surely be *too low* to be

[3] Since wolves are constantly on the move throughout their hunting territories, the actual locations shown in Figure 1a are roughly at the centers of these territories.

[4] One could also incorporate larger scales by using *higher-order nearest neighbors* [as discussed for example in Ripley (1996, sec.6.2)]. But these are not only more complex analytically, they are difficult to associate with specific scales of analysis.

[5] This concept was popularized by the work of Ripley (1976,1977). Note also that following standard convention, we now denote distance by h to distinguish it from nn-distance, d.

[6] Wolf packs typically consist of six to eight wolves (see the references in footnote 1 above).

consistent with CSR for the given density of wolves. Hence if one can identify appropriate bench-mark values for $K(h)$ under CSR, then these K-functions can be used to test for clustering and dispersion at various scales of analysis. We shall consider these questions in more detail in Section 4.4 below.

But for the moment, there are several features of definition (4.2.1) that warrant further discussion. First, while the distance metric in (4.2.1) is not specified, we shall always refer to *Euclidean distance*, $d(s,v)$ between pairs of points, as defined expression (3.2.1) above. Hence with respect to any given point event, s, the expected number of point events within distance h of s is simply the expected number of such events a circle of radius h about s, as shown in Figure 4.2 below.

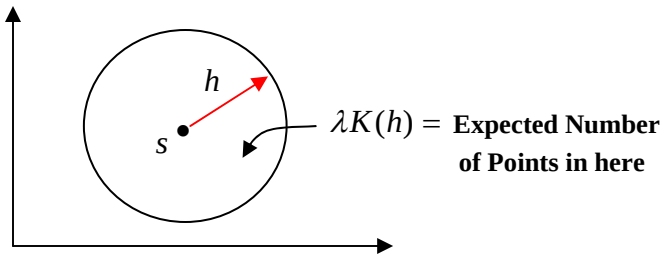

Fig.4.2. Interpretation of *K(h)*

This graphical image helps to clarify several additional assumptions implicit in the definition of $K(h)$. First, since this value is taken to depend only on the size of the circle (i.e., the radius h) and not its position (i.e., the coordinates of s) there is an implicit assumption of *spatial stationarity* [as in expression (2.5.1) above]. In other words, it is assumed that the *expected* number of additional points in this circle is the same regardless of where s is located. (This assumption will later be relaxed in our Monte Carlo applications of K-functions).

Observe next that the circularity of this region implicitly assumes that direction is not important, and hence that the underlying point process is *isotropic* (as in Figure 2.2 above). On the other hand, if the point process of interest were to exhibit some clear directionality, such as the vertical directionality in shown in Figure 2.3 above, then it might be more appropriate to use directional ellipses as defined by weighted Euclidean distances of the form:

$$(4.2.2) \qquad d(s,v) = \sqrt{w_1 \cdot (s_1 - v_1)^2 + w_2 \cdot (s_2 - v_2)^2}$$

where the weights w_1 and w_2 reflect relative sensitivities of point counts to movements in the horizontal or vertical direction, respectively.[7] More generally, if the relevant point

[7] One can also use appropriate quadratic forms to define *anisotropic distances* with any desired directional orientations. We shall consider such distances in more detail in the analysis of spatial variograms in Part II of this NOTEBOOK.

events occur in specific environments (such as the patterns of Philadelphia housing abandonments in Figures 1.4 and 1.5), then the relevant distances might be determined by these environments (such as travel distance on the Philadelphia street system).[8]

Finally, it is important to emphasize that the expected value in (4.2.1) is a *conditional* expected value. In particular, *given* that there is a point event, s, at the center of the circle in Figure 4.2 above, this value gives the expected number of *additional* points in this circle. This can be clarified by rewriting $K(h)$ in terms of conditional expectations. In particular if [as in Section 3.2.1 above] we now denote the circle in Figure 4.2 minus its center by

(4.2.3) $$C_h - \{s\} = \{v \in R : 0 < d(v,s) \le h\}$$

then $K(h)$ can be written more precisely as follows:

(4.2.4) $$K(h) = \tfrac{1}{\lambda} E[N(C_h - \{s\}) \mid N(s) = 1]$$

To see the importance of this conditioning, recall from expression (2.3.4) that for *any* stationary process (not just CSR processes) it must be true that the expected number of points in $C_h - \{s\}$ is simply proportional to its area, i.e., that

(4.2.5) $$E(C_h - \{s\}) = \lambda a(C_h - \{s\})$$

But this is *not* true of the conditional expectation above. Recall from the wolf-pack case, for example, that for small circles around any *given* wolf, the expected number of additional wolves is much larger than what would be expected based on area alone [i.e.,is larger than $\lambda a(C_h - \{s\})$]. These ideas will be developed in more detail in Section 4.4, where it is shown that such deviations from simple area proportionality form the basis for all K-function tests of the CSR Hypothesis.

4.3 Estimation of K-Functions

Given this general definition of K-functions as (conditional) expected values, we now consider the important practical question of *estimating* these values. To do so, we introduce the following notation analyzing for point counts. For any given realized point pattern, $S_n = (s_i : i = 1,..,n)$, and pair of points $s_i, s_j \in S_n$ we now denote the Euclidean distance between them by

(4.3.1) $$d_{ij} = d(s_i, s_j)$$

and for any distance, h, define the *indicator function*, I_h, for point pairs in S_n by

[8] Here it should be noted that tools are available in the **spatial analyst extension** of ARCMAP for constructing cost-weighted and shortest-paths distances. However, we shall not do so in this NOTEBOOK.

(4.3.2) $$I_h(d_{ij}) = I_h[d(s_i, s_j)] = \begin{cases} 1 & , d_{ij} \le h \\ 0 & , d_{ij} > h \end{cases}$$

From this definition it follows at once that for any given point $s_i \in S_n$, the total number of additional points s_j within distance h of s_i is given by the sum $\Sigma_{j \neq i} I_h(d_{ij})$. Hence, if i now refers to a randomly selected point generated by a point process on *R*, and if both the number and locations of points in *R* are treated as random variables, then in terms of (4.3.2) the K-function in (4.2.1) above can now be given the following equivalent definition:

(4.3.3) $$K(h) = \tfrac{1}{\lambda} E\left[\sum_{j \neq i} I_h(d_{ij})\right]$$

Observe also that for stationary point processes the value of $K(h)$ must be *independent* of the particular point event i chosen. So multiplying through by λ in (4.3.3) and summing over all point events $i = 1,..,n$ in region *R*, it follows that

(4.3.4) $$E\left[\sum_{j \neq i} I_h(d_{ij})\right] = \lambda K(h) \ , i = 1,..,n \ \Rightarrow \ \sum_{i=1}^{n} E\left[\sum_{j \neq i} I_h(d_{ij})\right] = n\lambda K(h)$$

$$\Rightarrow \ K(h) = \tfrac{1}{\lambda n} \sum_{i=1}^{n} E\left[\sum_{j \neq i} I_h(d_{ij})\right]$$

This "pooled" version of $K(h)$ motivates the following pooled estimate of $K(h)$, designated as the *sample K-function*,

(4.3.5) $$\hat{K}(h) = \tfrac{1}{\hat{\lambda} n} \sum_{i=1}^{n} \sum_{j \neq i} I_h(d_{ij})$$

where again, $\hat{\lambda} = n / a(R)$.[9] The advantage of this estimator is that uses *all* points of the given realized point pattern S_n in *R*. To interpret $\hat{K}(h)$, note that if we rewrite (4.3.5) as

(4.3.6) $$\hat{K}(h) = \tfrac{1}{\hat{\lambda}}\left[\tfrac{1}{n} \sum_{i=1}^{n} \left(\sum_{j \neq i} I_h(d_{ij})\right)\right]$$

then the expression in brackets is seen to be simply an average of the relevant point counts for each of the pattern points, $s_i \in S_n$. Hence, if the underlying process were truly *stationary* (and edge effects were small) then this sample K-function would be

[9] At this point it should be noted that our notation differs from [BG] where regions are denoted by a script $\mathcal{R}$ with area *R*. Here we use *R* for region, and make the area function, $a(R)$, explicit. In these terms, (4.3.5) is seen to be identical to the estimate on the top of p. 93 in [BG], where $1/(\hat{\lambda} n) = a(R)/n^2$.

approximately *unbiased* (and reasonably efficient) as an estimator of the common expected point count $E[\Sigma_{j\neq i} I_h(d_{ij})]$ in (4.3.3).[10]

However, since this idealization can never hold exactly in bounded regions *R*, it is necessary to take into account the *edge effects* created by the boundary of *R*. Unlike the case of nn-distances, where the expected values of nn-distances are *increased* for points near the boundary (as in Figure 3.16), the expected value of point counts are *reduced* for these points, as shown in Figure 4.3a below.

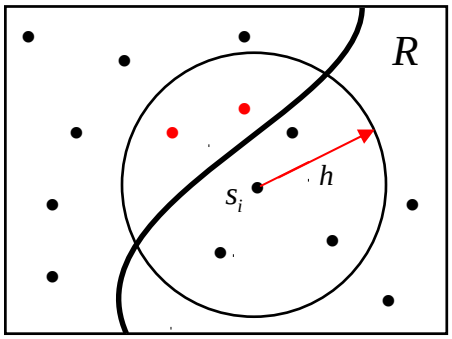

Fig.4.3a. Edge Effects for *K(h)*

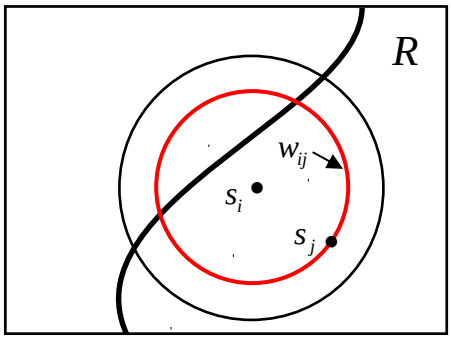

Fig.4.3b. Ripley's Correction

To counter this downward bias, Ripley (1976) proposed a "corrected" version of (4.3.5) that is quite effective in practice. His correction consists of weighting each point, s_j, in the count $\Sigma_{j\neq i} I_h(d_{ij})$ in a manner that inflates counts for points near the boundary. If one considers the circle about s_i passing through s_j (as shown in Figure 4.3b) and defines w_{ij} to be the fraction of its circumference that lies inside R, then the appropriate reweighting of s_j in the count for s_i is simply to divide $I_h(d_{ij})$ by w_{ij}, producing a new estimate known as *Ripley's correction*:

(4.3.7) $$\hat{K}(h) = \frac{1}{\hat{\lambda} n} \sum_{i=1}^{n} \sum_{j\neq i} \frac{I_h(d_{ij})}{w_{ij}}$$

One can gain some intuition here by observing in Figure 4.3b that weights will be unity unless circle about s_i passing through s_j actually leaves *R*. So only those point pairs will be involved that are close to the boundary of *R*, *relative* to distance h. Moreover, the closer that s_j is to the edge of *R*, the more of this circumference is outside *R*, and a hence the smaller w_{ij} becomes. This means that values $I_h(d_{ij})/w_{ij}$ are largest for points closest

[10] For further discussion of this approximate unbiasedness see Ripley (1977, Section 6).

to the edge, thus inflating $\hat{K}(h)$ to correct the bias. [An explicit derivation of Ripley's correction in given in Section 6 of the Appendix to Part I.]

It should be emphasized that while Ripley's correction is very useful for estimating the true K-function for a given stationary processes, this is usually not the question of most interest. As we have seen above, the key questions relate to whether this process exhibits structure other than what would be expected under CSR, and how this structure may vary as the spatial scale of analysis is increased. Here it turns out that in most cases, Ripley's correction is not actually needed. Hence this correction will not be used in the analysis to follow.[11]

4.4 Testing the CSR Hypothesis

To apply K-functions in testing the CSR Hypothesis, it is convenient to begin by ignoring edge effects, and considering the nature of K-functions under this hypothesis for points, $s \in R$ and distances, h, that are *not* influenced by edge effects. Hence, in contrast to Figure 4.3a above, we now assume that the set of locations, C_h, within distance h of s is entirely contained in R, i.e., that

(4.4.1) $$C_h = \{v \in R : d(s,v) \le h\} \subseteq R$$

Next recall from the basic independence assumption about individual point locations in CSR processes (Section 2.2 above) that for such processes, the expected number of points in $C_h - \{s\}$ does *not* dependent on whether or not there is a point event at s, so that

(4.4.2) $$E[N(C_h - \{s\}) \mid N(s) = 1] = E[N(C_h - \{s\})]$$

Hence from expression (4.2.3), together with the area formula for circles [and the fact that $a(C_h - \{s\}) = a(C_h)$], it follows that

(4.4.3) $$E[N(C_h - \{s\}) \mid N(s) = 1] = \lambda a(C_h - \{s\}) = \lambda a(C_h) = \lambda \pi h^2$$

which together with expression (4.2.4) yields the following simple K-function values:

(4.4.4) $$\boxed{K(h) = \tfrac{1}{\lambda}(\lambda \pi h^2) = \pi h^2}$$

Thus by standardizing with respect to density, λ, and ignoring edge effects as in (4.4.1), we see that the K-function reduces simply to *area* under the CSR Hypothesis. Note also that when $K(h) > \pi h^2$, this implies a mean point count *higher* than would be expected under CSR, and hence indicates some degree of clustering at scale h (as illustrated in

[11] Readers interested in estimating the true K-function for a given process are referred to Section 8.4.3 in Cressie (1993), and to the additional references found therein.

Section 4.2 above). Similarly, a value $K(h) < \pi h^2$ implies a mean point count *lower* than would be expected under CSR, and hence indicates some degree of dispersion at scale h. Thus for any given $h > 0$,

(4.4.5)
$$\boxed{\begin{array}{l} K(h) > \pi h^2 \;\Rightarrow\; \textit{clustering} \text{ at scale } h \\ \\ K(h) < \pi h^2 \;\Rightarrow\; \textit{dispersion} \text{ at scale } h \end{array}}$$

While these relations are adequate for testing purposes, area values are difficult to interpret directly. Hence it usually convenient to further standardize K-functions in a manner that eliminates the need for considering these values. If for each h we let

(4.4.6)
$$L(h) = \sqrt{\frac{K(h)}{\pi}} - h$$

then under CSR, this *L-function* has the property that

(4.4.7)
$$L(h) = \sqrt{\frac{\pi h^2}{\pi}} - h \;=\; h - h = 0$$

for all $h \geq 0$. In other words, this associated L-function is *identically zero* under CSR. Moreover, since $L(h)$ is an increasing function of $K(h)$, it follows that $L(h)$ is *positive* exactly when $K(h) > \pi h^2$, and is negative exactly when $K(h) < \pi h^2$. Hence the relations in (4.4.5) can be given the following simpler form in terms of L-functions:

(4.4.8)
$$\boxed{\begin{array}{l} L(h) > 0 \;\Rightarrow\; \textit{clustering} \text{ at scale } h \\ \\ L(h) < 0 \;\Rightarrow\; \textit{dispersion} \text{ at scale } h \end{array}}$$

Given the estimate, $\hat{K}(h)$, in (4.3.7) above, one can estimate $L(h)$ by

(4.4.9)
$$\hat{L}(h) = \sqrt{\frac{\hat{K}(h)}{\pi}} - h$$

and can in principle use (4.4.8) to test for clustering or dispersion.

4.5 Bodmin Tors Example

We can apply these testing ideas to Bodmin by using the MATLAB program, **k_function.m**. The first few lines of this program are shown below:

```
function C = k_function(loc,area,b,extent)

% K_FUNCTION computes the raw k-Function for a point pattern
%          and plots the normalized L-Function (without
%          edge corrections)

% Written by: TONY E. SMITH, 11/26/01

% INPUTS:
%    (i)  loc      = file of locations (xi,yi), i=1..m
%    (ii)  area    = area of region
%    (iii)  b      = number of bins to use in CDF (and plot)
%    (iv) extent  = 1 if max h = half of max pairwise distance (typical case)
%                 = 2 if max h = max pairwise distance to be considered
% DATA OUTPUTS: C = (1:b) vector containing raw Point Count
% SCREEN OUTPUTS:  Plot of L-Function over the specified extent.
```

To apply this program, again open the data file, **Bodmin.mat**, and recall that the tor locations are given in the matrix, **Bodmin**. As seen above, the program first computes $\hat{K}(h)$ for a range of distance values, h, and then coverts this to $\hat{L}(h)$ and plots these values against the reference value of zero. The maximum value of h for this illustration is chosen to be the maximum pairwise distance between pattern points (tors), listed as option **2** in input (iv) above. The number of intermediate distance values (bins) to be used is specified by input (iii). Here we set **b** = 20. Hence to run this program, type:

>> k_function(Bodmin,area,20,2);

The resulting plot is shown in Figure 4.4 to the right. Here the horizontal line indicates the "theoretical" values of $L(h)$ under the CSR Hypothesis. So it would appear that there is some degree of clustering at small scales, h. However, recall that the above analysis was predicated on the assumption of *no edge effects*. Since there are clearly strong edge effects in the Bodmin case, the real question here is how to incorporate these effects in a manner that will allow a meaningful test of CSR.

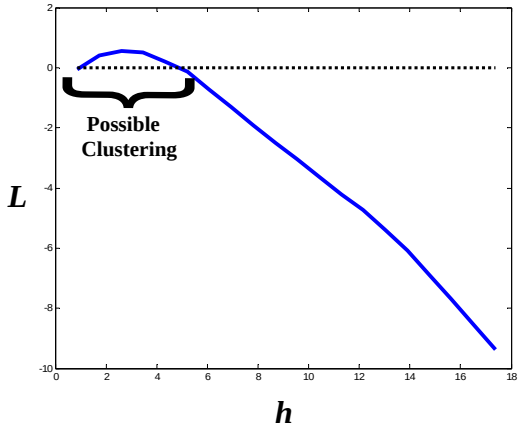

Fig.4.4. Bodmin L-function

One approach is suggested by recalling that *random* point pattern for Bodmin was also generated in Figure 3.14b above. Hence if the L-function for such a random pattern is plotted, then this can serve as a natural *benchmark* against which to compare the L-function for tors. This random pattern is contained in the matrix, **Bod_rn2**, of data file **Bodmin.mat** (and is also shown again in Figure 4.7 below). Hence the corresponding command, **k_function(Bod_rn2,area,20,2)**, now yields a comparable plot of this benchmark L-function as shown in Figure 4.5 below.

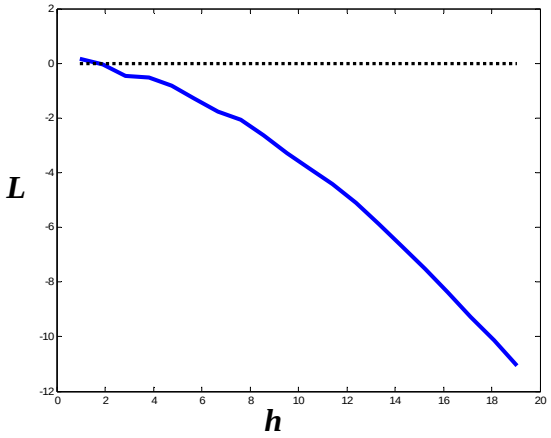

Fig.4.5. Random L-function

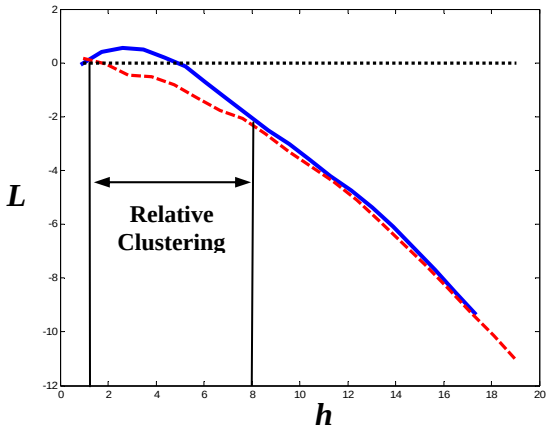

Fig.4.6. L-function Overlay

Here it is clear that the L-function for this random pattern is *not flat*, but rather is everywhere negative, and decreases at an increasing rate. Hence relative to zero, this pattern appears to exhibit more and more dispersion as the scale increases.

The reason for this of course is that the theory above [and expression (4.4.1) in particular] ignores those points near the boundary of the Bodmin region, such as the point shown in Figure 4.7. Here it is clear that for sufficiently small scales, h, there is little effect on $\hat{L}(h)$, so that values are close to zero for small h. But as this radius increases, it is also clear that most of the circle is eventually outside of R, and hence is mostly empty. Thus, given the estimated point density, $\hat{\lambda}$, for Bodmin tors *inside R*, point counts for large h start to look very small relative to the area πh^2. This is precisely the effect that *Ripley's correction* [expression (4.3.7)] attempts to eliminate.[12]

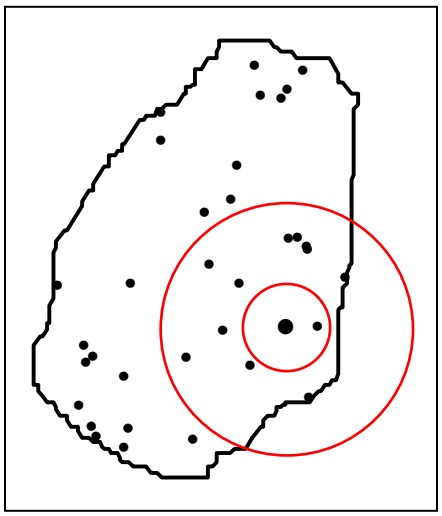

Fig.4.7. Bodmin Edge Effect

[12] A nice comparison of Ripley's correction with uncorrected L-functions (such as in Figure 4 above) is given in Figure 8.15 of Cressie (1993, p.617).

But if we now ignore the zero reference line and use this random L-function as a benchmark, then a perfectly meaningful comparison can be made by overlaying these two L-functions, as in Figure 4.6 above. Here one can see that the region of relative clustering is now considerably larger than in Figure 4.4, and occurs up to a scale of about $h = 8$ (see the scale shown in Figure 3.14). But observe even these benchmark comparisons have little meaning at scales so large that circles of radius h around all pattern points lie mostly outside the relevant region *R*. For this reason, the commonly accepted rule-of-thumb is that for any given point pattern, S_n, one should not consider h-values larger that half the maximum pairwise distance between pattern points. Hence if we now denote the maximum pairwise distance for S_n by, $h_{\max} = \max\{d(s_i, s_j) : s_i, s_j \in S_n\}$, and use $\bar{h}$ to indicate the largest value of h to be considered in a given case, then the standard rule-of-thumb is to set

(4.5.1) $$\bar{h} = h_{\max} / 2$$

This corresponds to option **1** for input (iv) of **k_function** above, and option **2** correspond to $\bar{h} = h_{\max}$. We shall have occasion to use (18) in many of our subsequent analyses, and in fact this will usually denote the "default" value of $\bar{h}$.

A more important limitation of this benchmark comparison is that (like the JMPIN version of the Clark-Evans test in Section 3.3.1 above) the results necessarily depend on the random point pattern that is chosen for a benchmark. Hence we now consider a much more powerful testing procedure using Monte Carlo methods.

4.6 Monte Carlo Testing Procedures

As we saw in Section 3.5 above, it is possible to use Monte Carlo methods to estimate the sampling distribution of nn-distances for any pattern size in a given region of interest. This same idea extends to the sampling distribution of *any* statistics derived from such patterns, and is of sufficient importance to be stated as a general principle:

> **SIMULATION PRINCIPLE:** *To test the CSR Hypothesis for any point pattern, S_n, of size n in a given region, R, one can simulate a large number of random point patterns, $\{S_n^{(i)} : i = 1,..,N\}$, of the same size, and compare S_n with this statistical population.*

Essentially, this simulation procedure gives us a clear statistical picture of what realized patterns from a CSR process on *R* should look like. In the case of K-function tests of CSR, we first consider the standard application of these ideas in terms of "simulation envelopes". This method is then refined in terms of a more explicit P-value representation.

4.6.1 Simulation Envelopes

The essential idea here is to simulate N random patterns as above and to compare observed estimate $\hat{L}(h)$ with the *range* of estimates $\hat{L}_i(h)$, $i = 1,..,N$ obtained from this simulation. More formally, if one defines the *lower-envelope* and *upper-envelope* functions respectively by

(4.6.1) $$\boldsymbol{L}_N(h) = \min\{\hat{L}_i(h) : i = 1,..,N\}$$

(4.6.2) $$\boldsymbol{U}_N(h) = \max\{\hat{L}_i(h) : i = 1,..,N\}$$

then $\hat{L}(h)$ is compared with $\boldsymbol{L}_N(h)$ and $\boldsymbol{U}_N(h)$ for each h. So for a given observed pattern, S_n, in region R the steps of this Monte Carlo testing procedure can be outlined as follows:

(i) Generate a number of random patterns, $\{S_n^{(i)} : i = 1,..,N\}$, of size n in region R (say $N = 99$).

(ii) Choose a selection of h-values, $H = \{h_1, h_2, .., \bar{h}\}$, and compute $\hat{L}_i(h)$ for each $h \in H$ and $i = 1,..,N$.

(iii) Form the *lower-* and *upper-envelope* functions, and $\boldsymbol{L}_N(h)$ and $\boldsymbol{U}_N(h)$ in (4.6.1) and (4.6.2).

(iv) Plot the L-values, $\hat{L}(h)$, for the observed pattern S_n along with the upper and lower values, $\boldsymbol{U}_N(h)$ and $\boldsymbol{L}_N(h)$, for each $h \in H$.

The result of this procedure is to yield a plot similar that shown in Figure 4.8 to the right. Here the blue region indicates the area in which the observed L-function, $\hat{L}(\cdot)$ is outside the range defined by the upper- and lower-envelope functions. In the case shown, this area is above the envelope, indicating that there is significant *clustering* relative to the simulated population under CSR.

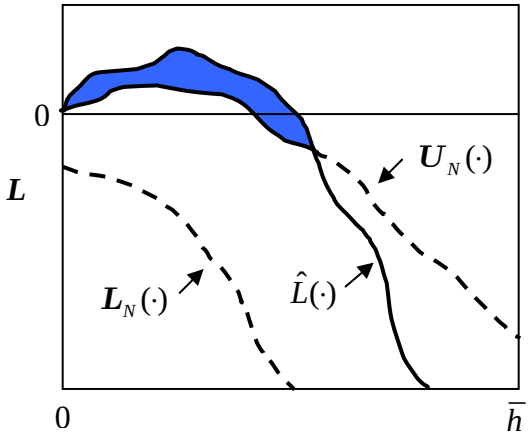

Fig.4.8. Simulation Envelope

The key difference between this figure and Figure 4.6 above is that, rather than a single benchmark pattern, we now have a *statistical population* of patterns for gauging the

significance of $\hat{L}(\cdot)$. This plot in fact summarizes a series of *statistical tests* at each scale of analysis, $h \in H$. In the case illustrated, if we consider any h under the blue area in Figure 4.8, then by definition, $\hat{L}(h) > \boldsymbol{U}_N(h)$. But if pattern S_n were just another sample from this population of random patterns, then every sample value $\{\hat{L}(h), \hat{L}_1(h), .., \hat{L}_N(h)\}$ would have the same chance of being the biggest. So the chance that $\hat{L}(h)$ is the biggest is only $1/(N+1)$. More formally, if pattern S_n is consistent with the CSR Hypothesis then:

(4.6.3) $$\Pr[\hat{L}(h) > \boldsymbol{U}_N(h)] = \frac{1}{N+1}\ ,\ h \in H$$

(4.6.4) $$\Pr[\hat{L}(h) < \boldsymbol{L}_N(h)] = \frac{1}{N+1}\ ,\ h \in H$$

These probabilities are thus seen to be precisely the *P-values* for one-tailed tests of the CSR Hypothesis against *clustering* and *dispersion*, respectively. For example, if $N = 99$ [as in step (i) above] then the chance that $\hat{L}(h) > \boldsymbol{U}_N(h)$ is only $1/(99+1) = .01$. Hence at scale, h, one can infer the presence of *significant clustering at the* .01*-level*. Similarly, if there were any $h \in H$ with $\hat{L}(h) < \boldsymbol{L}_N(h)$ in Figure 4.8, then at this scale one could infer the presence of *significant dispersion at the* .01*-level*. Moreover, higher levels of significance could easily be explored by simulating larger numbers of random patterns, say $N = 999$.

This Monte Carlo test can be applied to the Bodmin example by using the MATLAB program, **k_function_sim.m**, shown below.

```
function k_function_sim(loc,area,b,extent,sims,poly)

%  K_FUNCTION_SIM computes the raw k-Function for a point
%    pattern plus N random point patterns for a single polygon and
%    plots the normalized L-Function plus Upper and Lower envelopes

% INPUTS:
%    (i)  loc      = file of locations (xi,yi), i=1..n
%    (ii)  area    = area of region
%    (iii)  b      = number of bins to use in CDF (and plot)
%    (iv) extent  = 2 if max h = max pairwise distance to be considered
%                 = 1 if max b = half of max pairwise distance (typical case)
%    (v)  sims     = number of simulated random patterns
%    (vi)  poly    = polygon boundary file
```

Note that the two key additional inputs are the numbers of simulations (here denoted by **sims** rather that **N**) and the boundary file, **poly**, for the region, *R*. As with the program, **clust_sim**, in Section 3.5 above, **poly** is needed in order to generate random points in *R*.

To apply this program to Bodmin with **sims** = 99, be sure the data file, **Bodmin.mat**, in open in the Workspace, and write:

>> **k_function_sim(Bodmin,area,20,1,99,Bod_poly);**

The results of this program are shown in Figure 4.9 to the right. Notice first that there is again some clustering, and that now it can be inferred that this clustering is significant at the .01-level ($N = 99$). Notice also that the range of significant clustering is considerably smaller that that depicted in Figure 4.6 above. This will almost always be the case, since here the $\hat{L}(h)$ values must be bigger that 99 other random values, rather than just one "benchmark" value. Notice also that this scale, roughly $1.5 \le h \le 4.5$, appears to be more consistent with Figure 3.14a.

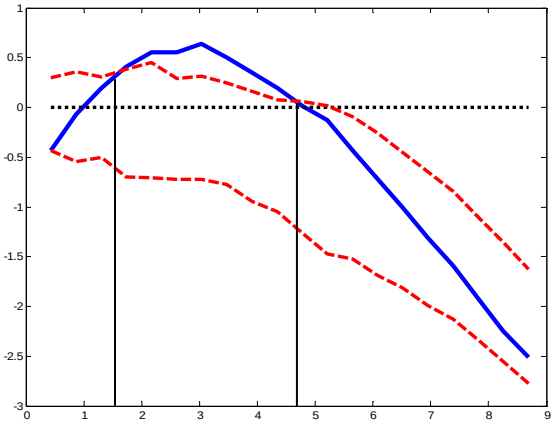

Fig.4.9. Bodmin Envelope Test

However, this approach is still rather limited in the sense that it provides information *only* about the relation of $\hat{L}(h)$ to the maximum and minimum simulated values $\boldsymbol{U}_N(h)$ and $\boldsymbol{L}_N(h)$ for each $h \in H$. Hence the following refinement of this approach is designed to make fuller use of the information obtained from the above Monte Carlo procedure.

4.6.2 Full P-Value Approach

By focusing on the maximum and minimum values, $\boldsymbol{U}_N(h)$ and $\boldsymbol{L}_N(h)$ for each $h \in H$, the only P-values that can be obtained are those in (4.6.3) and (4.6.4) above. But it is clear for example that values of $\hat{L}(h)$ that are just below $\boldsymbol{U}_N(h)$ are probably still very significant. Hence a natural extension of the above procedure is to focus directly on P-values for clustering and dispersion, and attempt to estimate these values on the basis of the given samples. Turning first to clustering, the appropriate P-value is given by the answer to the following question: *If the observed pattern were coming from a CSR process in region R, then how likely it would be to obtain a value as large as* $\hat{L}(h)$? To answer this question let the *observed L-value* be denoted by $l_0 = \hat{L}(h)$, and let the random variable, $L_{CSR}(h)$, denote the L-value (at scale h) obtained from a randomly sampled CSR pattern of size n on *R*. Then the answer to the above question

is given formally by the probability that $L_{CSR}(h)$ is at least as large as l_0, which we designate as the *clustering P-value*, $P_{clustered}(h)$, at scale h for the observed pattern, S_n:

(4.6.5) $$P_{clustered}(h) = \Pr[L_{CSR}(h) \geq l_0].$$

To estimate this probability, observe that our simulation has by construction produced a sample of N realized values, $l_i = \hat{L}_i(h)$, $i = 1,..,N$, of this random variable $L_{CSR}(h)$. Moreover, under the CSR Hypothesis the observed value, l_0, is just another sample, which for convenience we designate as sample $i = 0$. Hence the task is to estimate (4.6.5) on the basis of a random sample, $(l_0, l_1,..,l_N)$ of size $N+1$. The standard approach to estimating event probabilities is simply to count the number of times the event occurs, and then to estimate its probability by the relative frequency of these occurrences. In the present case, the relevant event is "$L_{CSR}(h) \geq l_0$". Hence if we now define the indicator variables for this event by

(4.6.6) $$\delta_0(l_i) = \begin{cases} 1 & , l_i \geq l_0 \\ 0 & , l_i < l_0 \end{cases} \quad , \quad i = 0,1,..,N$$

then the *relative-frequency estimator*, $\hat{P}_{clustered}(h)$, of the desired P-value is given by[13]

(4.6.7) $$\hat{P}_{clustered}(h) = \Pr[L_{CSR}(h) \geq l_0] = \tfrac{1}{N+1}\sum_{i=0}^{N} \delta_0(l_i)$$

To simplify this expression, observe that if $m_+(l_0)$ denotes the number of simulated samples, $i = 1,..,N$, that are *at least as large as* l_0 [i.e., with $\delta_0(l_i) = 1$], then this estimated P-value reduces to[14]

(4.6.8) $$\hat{P}_{clustered}(h) = \frac{m_+(l_0)+1}{N+1}$$

Observe that expression (4.6.3) above is now the special case of (4.6.8) in which $\hat{L}(h)$ happens to be bigger than all of the N simulated values. But (4.6.8) conveys a great deal more information. For example, suppose that $N = 99$ and that $\hat{L}(h)$ is only the fifth highest among these $N+1$ values. Then in Figure 4.9 this value of $\hat{L}(h)$ would be inside the envelope [probably much closer to $\boldsymbol{U}_N(h)$ than to $\boldsymbol{L}_N(h)$]. But no further information could be gained from this envelope analysis. However in (4.6.8) the estimated the chance of observing a value as large as $\hat{L}(h)$ is $5/(99+1) = .05$, so that

[13] This is also the *maximum-likelihood estimator* of $P_{cluster}(h)$. Such estimators will be considered in more detail in Part III of this NOTEBOOK.

[14] An alternative derivation of this P-value is given in Section 7 of the Appendix to Part I.

this L-value is still sufficiently large to imply some significant degree of clustering. Such examples show that the P-values in (4.6.8) are considerably more informative than the simple envelopes above.

Turning next to *dispersion*, the appropriate P-value is now given by the answer to the following question: *If the observed pattern were coming from a CSR process in region R, then how likely it would be to obtain a value as small as* $\hat{L}(h)$? The answer to this question is given by the *dispersion P-value*, $P_{dispersed}(h)$, at scale h for the observed pattern, S_n:

(4.6.9) $$P_{dispersed}(h) = \Pr[L_{CSR}(h) \leq l_0]$$

Here, if we let $m_{-}(l_0)$ denote the number of simulated L-values that are *no larger than* l_0, then exactly the same argument above [with respect to the event "$L_{CSR}(h) \leq l_0$"] now shows that the appropriate *relative-frequency estimate* of $P_{dispersed}(h)$, is given by

(4.6.10) $$\hat{P}_{dispersed}(h) = \frac{m_{-}(l_0)+1}{N+1}$$

To apply these concepts, observe first that (unless many l_i values are the same as l_0)[15] it must be true that $\hat{P}_{dispersed}(h) \approx 1 - \hat{P}_{clustered}(h)$. So there is generally no need to compute both. Hence we now focus on clustering P-values, $\hat{P}_{clustered}(h)$ for a given point pattern, S_n, in region *R*. Observe next that to determine $\hat{P}_{clustered}(h)$, there is no need to use L-values at all. One can equally well order the K-values. In fact, there is no need to normalize by $\hat{\lambda}$ since this value is the same for both the observed and simulated patterns. Hence we need only compute "raw" K-function values, as given by the bracketed part of expression (4.3.6). Finally, to specify an appropriate range of scales to be considered, we take the appropriate maximum value of h to be the default value $\bar{h} = h_{\max}/2$ in (4.5.1), and specify a number b of equal divisions of $\bar{h}$. The values of $\hat{P}_{clustered}(h)$ are then computed for each of these h values, and the result is plotted.

This procedure is operationalized in the MATLAB program, **k_count_plot.m**. This program will be discussed in more detail in the next section. So for the present, we simply apply this program to Bodmin (with **Bodmin.mat** in the Workspace), by setting $N = 99$, $b = 20$ and writing:

>> k_count_plot(Bodmin, 99,20,1,Bod_poly);

[15] The question of how to handle such ties is treated more explicitly in Section 7 of the Appendix to Part I.

(Simply ignore the fourth input "**1**" for the present.) The screen output of **k_count_plot** gives the value of $\bar{h}$ computed by the program, which in this case is **Dmax/2 = 8.6859**. The minimum pairwise distance between all pairs of points (**Dmin = 0.5203**) is also shown. This value is useful for interpreting P-values at small scales, since all values of h less that this minimum must have $\hat{K}(h) = 0$ and hence must be "maximally dispersed" by definition [since no simulated pattern can have smaller values of $\hat{K}(h)$].

The cluster P-value plot for Bodmin is shown in Figure 4.10. With respect to significant clustering, there is seen to be general agreement with the results of the envelope approach above. Here we see significant clustering at the .05 level (denoted by the lower dashed red line) for scale values in the range $1.3 \le h \le 6.1$ (remember that one will obtain slightly different values for each simulation).[16] But this figure clearly shows more. In particular, clustering at scales in the range $1.7 \le h \le 5.7$ is now seen to be significant at the .01 level, which by definition the highest level of significance possible for N = 99.

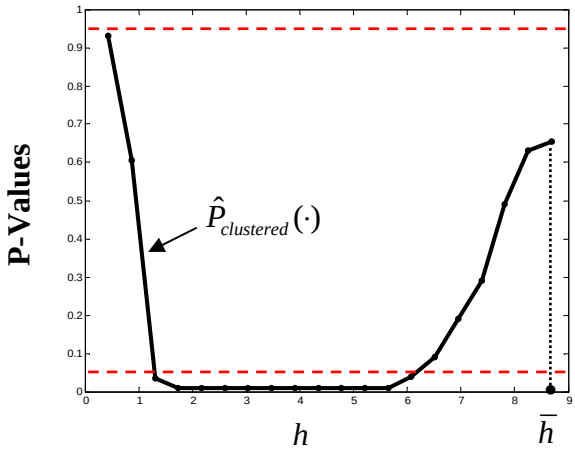

Fig.4.10. Bodmin Cluster P-Values

Here it is also worth noticing that the clustering P-value at scale $h = .5$ is so large (in fact .93 in the above simulation) that it shows *weakly significant dispersion* (where the upper dashed red line indicates significant dispersion at the .05 level). The statistical reason for this can be seen from the screen output that shows the minimum distance between any two tors to be .52. Hence at scale $h = .5$ it must be true that *no* circle of radius .5 about any tor can contain other tors, so that we must have $\hat{K}(.5) = 0$. But since random point patterns such as in Figure 3.14b often have at least one pair of points this close together, it becomes clear that there is indeed some genuine local dispersion here. Further reflection suggests that is probably due to the nature of rock outcroppings, which are often only the exposed portion of larger rock formations and thus cannot be too close together. So again we see that the P-value map adds information about this pattern that may well be missed by simply visual inspection.

4.7 Nonhomogeneous CSR Hypotheses

As mentioned in Section 2.4 above, it is possible to employ the Generalized Spatial Laplace Principle to extend CSR to the case of *nonhomogeneous* reference measures.

[16] Simulations with N = 999 yield about the same results as Figure 4.10, so this appears to be a more accurate range than given by the envelope in Figure 4.9.

While no explicit applications are given in [BG], we can illustrate the main ideas with the following housing abandonment example.

4.7.1 Housing Abandonment Example

As in the Philadelphia example of Section 1.2 above, suppose that we are given the locations of n currently abandoned houses in a given city, *R*, such as in Figure 4.11a below.

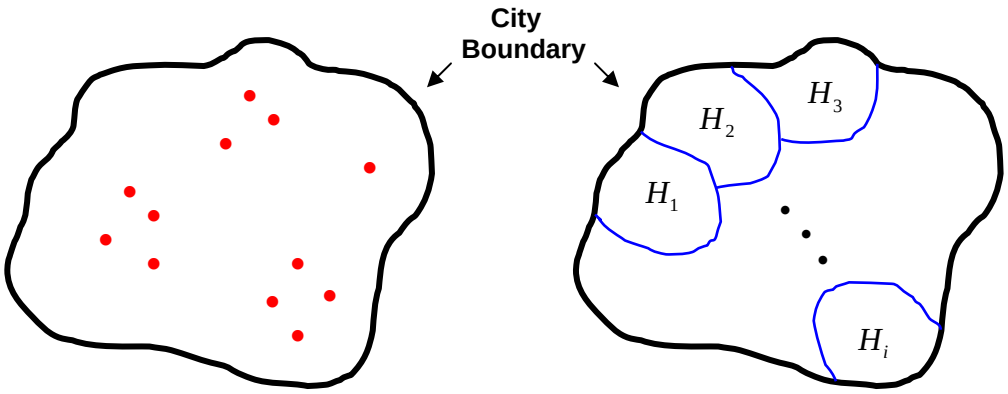

Fig.4.11a. Abandoned Houses **Fig.4.11b. Census Tract Data**

In addition, suppose that data on the number of *housing units*, $H_i = \rho(C_i)$, in each census tract, C_i, $i = 1,..,m$ within city *R* is also available, as in Figure 4.11b. If the number of *total housing units* in the city is denoted by

$$(4.7.1) \qquad H = \rho(R) = \sum_{i=1}^{m} \rho(C_i) = \sum_{i=1}^{m} H_i$$

then the probability that a randomly sampled housing unit will be located in tract i is given by

$$(4.7.2) \qquad P_i = \frac{H_i}{H} = \frac{\rho(C_i)}{\rho(R)}, \quad i = 1,..,m$$

Thus if these n housing abandonments were completely random events (i.e., with no housing unit more likely to be abandoned than any other) then one would expect the distribution of abandoned houses across census tracts to be given by n independent random samples from the distribution in (4.7.2).[17] More formally, this is an example of a *nonhomogeneous CSR hypothesis* with respect to a given *reference measure*, ρ.

[17] In particular, this would yield a *marginal* distribution of abandonments in each tract C_i given by the binomial distribution in expression (2.4.3) above with $C = C_i$.

4.7.2 Monte Carlo Tests of Hypotheses

To test such hypotheses, we proceed exactly the same way as in the homogeneous case. The only real difference here is that the probability distributions corresponding to nonhomogeneous spatial hypotheses are somewhat more complex. Using the above example as an illustration, we can simulate samples of n random abandonments from the appropriate distribution by the following *two-stage sampling procedure*:

(i) Randomly sample a census tract, C_{1i}, from the distribution in (4.7.2).

(ii) Randomly locate a point $s_1^{(i)}$ in C_{1i}.

(iii) Repeat (i) and (ii) n times to obtain a point pattern $S_n^{(i)} = (s_j^{(i)} : j = 1,..,n)$.

The resulting pattern $S_n^{(i)}$ corresponds to the above hypothesis in the sense that individual abandonment locations are *independent*, and the expected number of abandonments in each tract C_j is *proportional to the reference measure*, $H_j = \rho(C_j)$. However, this reference measure ρ is only an approximation to the theoretical measure, since the actual locations of *individual housing units* are not known. [This is typical of situations where certain key spatial data is available only at some aggregate level.[18]] Hence in step (ii) the location of a housing units in C_i is taken to be *uniformly* (homogeneously) distributed throughout this subregion. The consequences of this "local uniformity" approximation to the ideal reference measure, ρ, will be noted in the numerical examples below.

Given a point pattern, $S_n = (s_j : j = 1,..,n)$, such as the locations of n abandonments above, together with N simulated patterns $\{S_n^{(i)} : i = 1,..,N\}$ from the Monte Carlo procedure above, we are now ready to *test* the corresponding nonhomogeneous CSR hypothesis based on this reference measure ρ. To do so, we can proceed exactly as before by constructing K-counts, $\hat{K}(h)$, for the observed pattern, S_n, over a selected range of scales, h, and then constructing the corresponding K-counts, $\hat{K}^{(i)}(h)$, for each simulated pattern, $i = 1,..,N$.

This procedure is operationalized in the same MATLAB program, **k_count_plot** (which is more general than the Bodmin application above). Here the only new elements involve a partition of region *R* into subregions, $\{C_i : i = 1,..,m\}$, together with a specification of the appropriate reference measure, ρ, defined on this set of subregions.

[18] Such aggregate data sets will be treated in more detail in Part III of this NOTEBOOK.

4.7.3 Lung Cancer Example

To illustrate this testing procedure, the following example has been constructed from the Larynx and Lung Cancer example of Section 1.2 above. Here we focus only on Lung Cancer, and for simplicity consider only a random subsample of $n = 100$ lung cases, as shown in Figures 4.12 below.

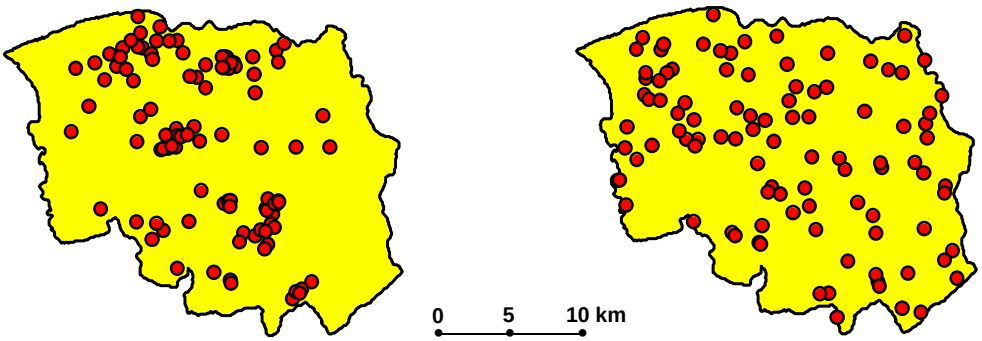

Fig.4.12. Subsample of Lung Cases **Fig.4.13. Random Sample of Same Size**

Note from Figures 1.7 and 1.8 that this is fairly representative of the full data set (917 lung cancers). To analyze this data set we begin by observing that in terms of area alone, the point pattern in Figure 4.12 is obviously quite clustered.

One can see this by comparison with a typical random pattern of the same size in Figure 4.13. This can be verified statistically by using the program **k_function_plot** (as in the Bodmin case) to conduct a Monte Carlo test for the homogenous case developed above. The results are shown in Figure 4.14 to the right. Here it is evident that there is *extreme* clustering. In fact, note from the scale in Figure 4.12 above that there is highly significant clustering up to a radius of $h = 20\,km$, which is large enough to encompass the *entire region*. Notice also that the significance levels here are as high as possible for the given number of simulations, which in this case was $N = 999$. This appears to be due to the fact that the overall pattern of points in Figure 4.12 is not only more clustered but is also more compact. So for the given common point density in these figures, cell counts centered at pattern points in Figure 4.12 tend to be uniformly higher than in Figure 4.13.

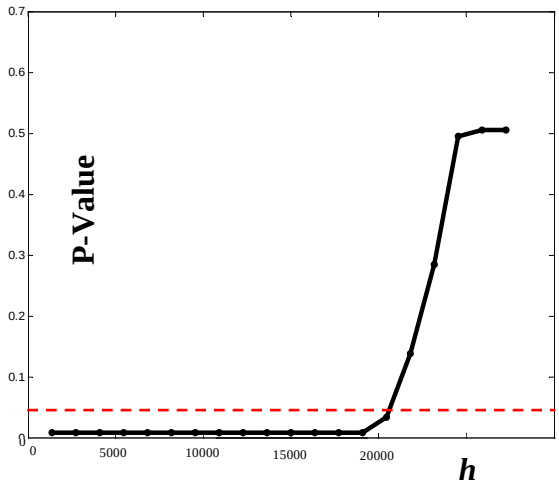

Fig.4.14. Test of Homogeneous Clustering

But the single most important factor contributing to this clustering (as observed in Section 2.4 above) is the conspicuous absence of an appropriate reference measure – namely *population*. In Figure 4.15 below, the given subsample of lung cases in Figure 4.12 above is now depicted on the appropriate population backcloth of Figure 1.8.

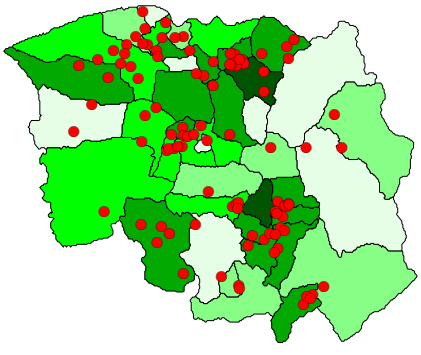

Fig.4.15. Subsample of Lung Cases

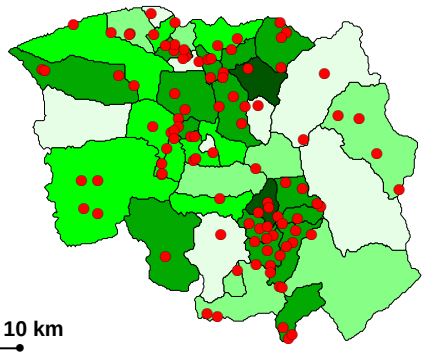

Fig.4.16. Random Sample from Population

Here it is clear that much of the clustering in Figure 4.12 can be explained by variations in population density. Notice also that the relative sparseness of points in the west and east are also explained by the lower population densities in these areas (especially in the east). For comparison, a random pattern generated using the two-stage sampling procedure above is shown in Figure 4.16. Here there still appears to be somewhat less clustering than in Figure 4.15, but the difference is now far less dramatic than above.

Using these parish population densities as the reference measure, ρ, a Monte Carlo test was run with $N = 999$ simulated patterns (including the one shown in Figure 4.16). The results of this test are plotted in Figure 4.17 to the right. Notice that the dramatic results of Figure 4.14 above have all but disappeared. There is now only significant clustering at the local scale (with $h \leq 2\ km$). Moreover, even this local clustering appears to be an artifact of the spatial aggregation inherent in the parish population density measure, ρ. As pointed out above, this aggregation leads to simulated point patterns under the nonhomogeneous CSR hypothesis that tend to be much too *homogeneous* at the parish level. This is particularly evident in the densely populated area of the south-central portion of the region shown. Here the tighter clustering of lung cancer cases seen in Figure 4.15 more accurately reflects local variations in population density than does the relatively uniform scattering of points in Figure 4.16. So in fact, a more

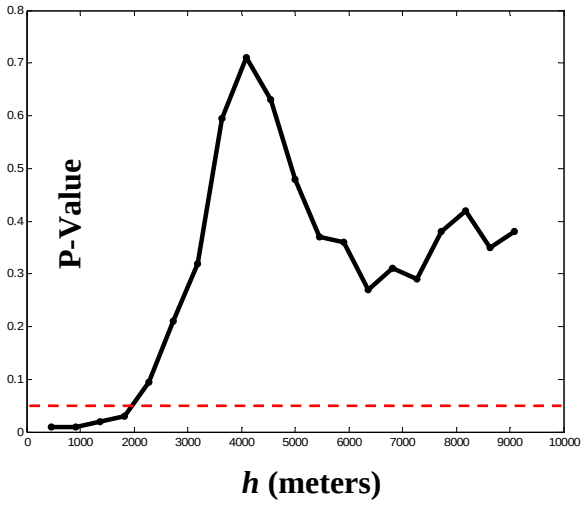

Fig.4.17. Test of Nonhomogeneous Clustering

disaggregated representation of population density would probably show that there is no significant clustering of lung cancer cases whatsoever.

4.8 Local K-Function Analysis

Up to this point we have only considered *global* properties of point patterns, namely the overall clustering or dispersion of patterns at various scales. However, in many cases interest focuses on more local questions of *where* significant clustering or dispersion is occurring. Here we begin by constructing local versions of K-functions, and then apply them to several examples.

4.8.1 Construction of Local K-Functions

Recall from expression (4.3.3) that K-functions were defined in terms of expected point counts for a *randomly* selected point in a pattern. But exactly the same definitions can be applied to each individual point in the pattern by simply modifying the interpretation of (4.3.3) to be a *given* point, i, rather than a randomly sampled point, and rewriting this expression as a *local K-function* for each point, i:

(4.8.1) $$K_i(h) = \tfrac{1}{\lambda} E\left[\sum_{j\neq i} I_h(d_{ij})\right]$$

Moreover, if we now *relax the stationarity assumption* used in (4.3.4) above, then these expected values may differ for each point, i. In this context, the pooled estimator (4.3.5) for the stationary case now reduces to the corresponding *local estimator*:

(4.8.2) $$\hat{K}_i(h) = \tfrac{1}{\hat{\lambda}} \sum_{j\neq i} I_h(d_{ij})$$

Hence to determine whether there is significant clustering about point i at scale h, one can develop local Monte Carlo testing procedures using these statistics.

4.8.2 Local Tests of Homogeneous CSR Hypotheses

In the case of homogenous CSR hypotheses, one can simply hold point i fixed in region R and generate N random patterns of size $n-1$ in R (corresponding to the locations of all other points in the pattern). Note that in the present case, (4.8.2) is simply a count of the number of points with distance h of point i, scaled by $1/\hat{\lambda}$. But since this scaling has no effect on Monte Carlo tests of significance, one can focus solely on point counts (which may be thought of as a "raw" K-function). For each random pattern, one can then simply count the number of points within distance h of point i. Finally, by comparing these counts with the observed point count, one can then generate *p-values* for each point $i = 1,..,n$ and distance, h, [paralleling (4.6.8) above]:

(4.8.3) $$\hat{P}_i(h) = \frac{m_i(h)+1}{N+1}$$

where $m_i(h)$ now denotes the number of simulated patterns with counts at distance h from i at least as large as the observed count. This testing procedure is operationalized in the MATLAB program, **k_count_loc.m**, shown below:

```
function [PVal,C0] = k_count_loc(loc,sims,D,M,poly)

% K_COUNT_LOC computes the raw K-function at each point in the
% pattern, loc, for a range of distances, D, and allows tests of non-
% homogeneous CSR hypotheses by including a set of polygons, poly, with
% reference measure, M.
%
% INPUTS:
%    (i)     loc  = population location file [loc(i)=(Xi, Yi),i=1:N]
%    (ii)   sims = number of simulations
%    (iii)    D   = set of distance values (in ASCENDING order)
%    (iv)    M   = k-vector of measure values for each of k polygons
%    (v)    poly = matrix describing boundaries of k polygons
```

Here the main output, **Pval**, is a matrix of *P-values* at each reference point and each distance value under the CSR Hypothesis. (The point counts for each point-distance pair are also in the output matrix, **C0**.) Notice that since homogeneity is simply a special case of heterogeneity, this program is designed to apply both to homogeneous and nonhomogeneous CSR hypotheses.

Application to Bodmin Tors

The homogeneous case can be illustrated by the following application to *Bodmin tors*. Recall that the location pattern of tors is given by the matrix, **Bodmin**, in the workspace **Bodmin.mat**. Here there is a single boundary polygon, **Bod_poly**. Hence the reference measure can be set to a constant value, say **M = 1**. So the appropriate command for **999** simulations in this case is given by:

>> [Pval,C0] = k_count_loc(Bodmin,999,D,1,Bod_poly);

In view of Figure 4.10 above, one expects that the most meaningful distance range for significant clustering will be somewhere between $h = 1$ and $h = 5$ kilometers. Hence the selected range of distances was chosen to be **D = [1,2,3,4,5]**. One key advantage of this type of *local* analysis is that since a p-value is now associated with each individual point, is now possible to *map* the results. In the present case, the results of this Monte Carlo analysis were imported to ARCMAP, and are displayed in **Bodmin.mxd**. In Figure 4.18 below, the p-value maps for selected radii of $h = 2,3,5$ *km* are shown. As

seen in the legend (lower right corner of the figure), the darker red values correspond to lower p-values, and hence denote regions of more significant clustering. As expected, there are basically two regions of significant clustering corresponding to the two large groupings of tors in the Bodmin field.

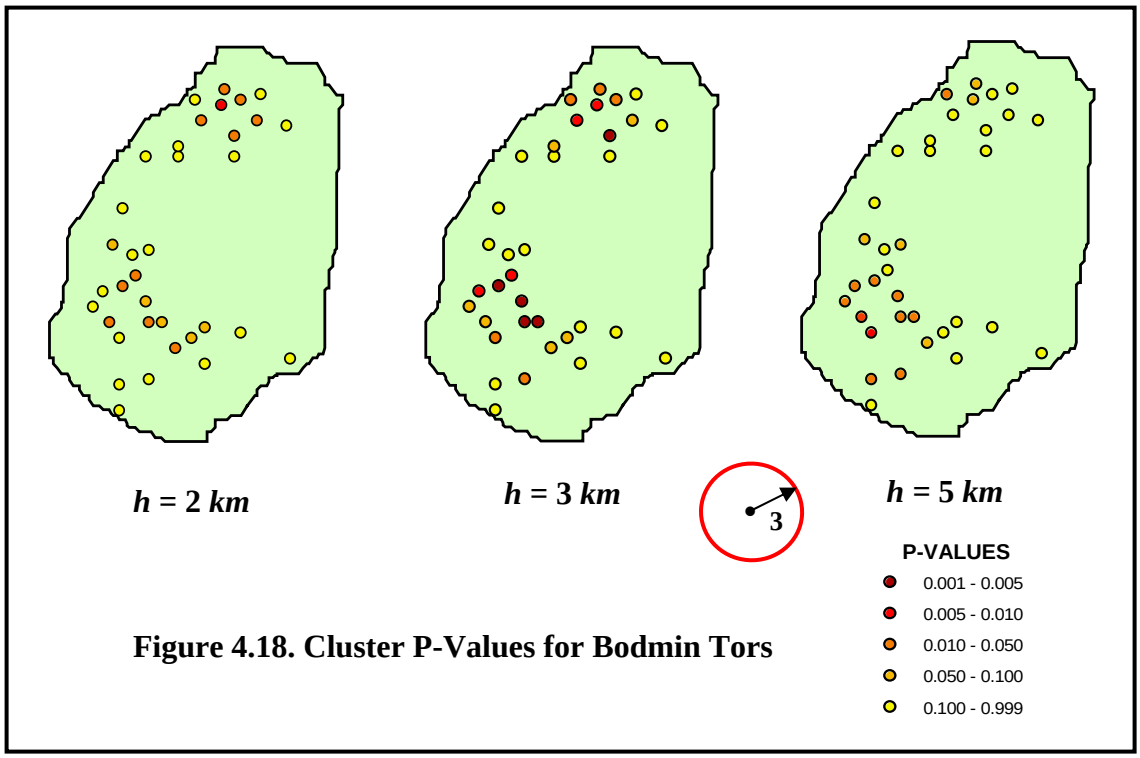

Figure 4.18. Cluster P-Values for Bodmin Tors

Notice here that clustering is much more pronounced at a radius of 3 *km* than at smaller or larger radii. (The red circle in the figure shows the actual scale of a 3 *km* radius.) This figure well illustrates the ability of local K-function analyses to pick up sharper variations in scale than global analyses such as Figure 4.10 above (where there appeared to be equally significant clustering at all three scales, $h = 2,3,5$ *km*). Hence it should be clear from this example that local analyses are often much more informative than their global counterparts.

Local Analyses with Reference Grids

The ability to map p-values in local analyses suggests one additional extension that is often more appropriate than direct testing of clustering at each individual point. By way of motivation, suppose that one is studying a type of tree disease by mapping the locations of infected trees in a given forest. Here it may be of more interest to distinguish diseased regions from healthy regions in the forest rather than to focus on individual trees. A simple way to do so is to establish a *reference grid* of locations in the forest, and then to estimate clustering p-values at each grid location rather than at each tree. (The construction of reference grids is detailed in Section 4.8.3 below.) Such a uniform grid of p-values can then be easily interpolated to produce a smoother visual

summary of disease clustering. An illustration of this reference-grid procedure is shown in Figure 4.19 below, where the red dots denote diseased trees in the section of forest shown, and where the white dots are part of a larger grid of reference points. In this illustration the diseased-tree count within distance h of the grid point shown is thus equal to 4.

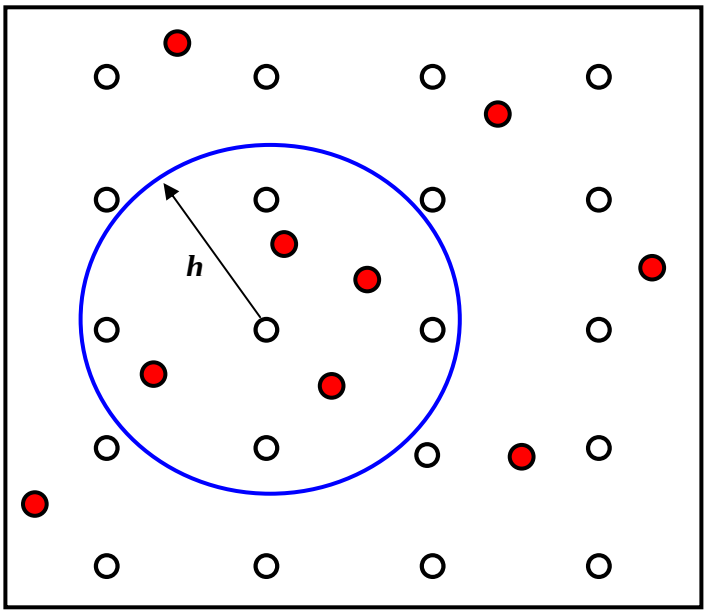

Figure 4.19. Reference Grid for Local Clustering

Assuming that the forest itself is reasonably uniform with respect to the spatial distribution of trees, the *homogeneous CSR hypothesis* would again provide a natural benchmark for identifying significant clustering of diseased trees. In this case, one would simulate random patterns of diseased trees and compare disease counts with those observed within various distances h of each grid point. Hence those grid points with low p-values at distance h would denote locations where there is significant disease clustering at scale h.

To develop the details of this procedure, it is convenient to construct a reference grid representation for *Bodmin*, so that the two approaches can more easily be compared. To do so, we start by constructing a reference grid for Bodmin. By inspecting the boundary of Bodmin in ARCMAP one can easily determine a box of coordinate values just large enough to contain all of Bodmin. In the present case, appropriate bounding X-values and Y-values are given by **Xmin = -5.2**, **Xmax = 9.5**, **Ymin = -11.5**, and **Ymax = 8.3**. Next one needs to choose a cell size for the grid (as exemplified by the spacing between white dots in Figure 4.19). One should try to make the grid fine enough to obtain a good interpolation of the p-values at grid points. Here the value of .5 *km* was chosen for spacing in each direction, yielding square cells with dimensions, **Xcell = .5 = Ycell**.

The construction of the corresponding reference grid is operationalized in the program **grid_form.m** with the command:

>> ref = grid_form(Xmin,Xmax,Xcell,Ymin,Ymax,Ycell);

This produces a 2-column matrix, **ref**, of grid point coordinates. (The upper left corner of the grid is displayed on the screen for a consistency check.). A plot of the full grid,

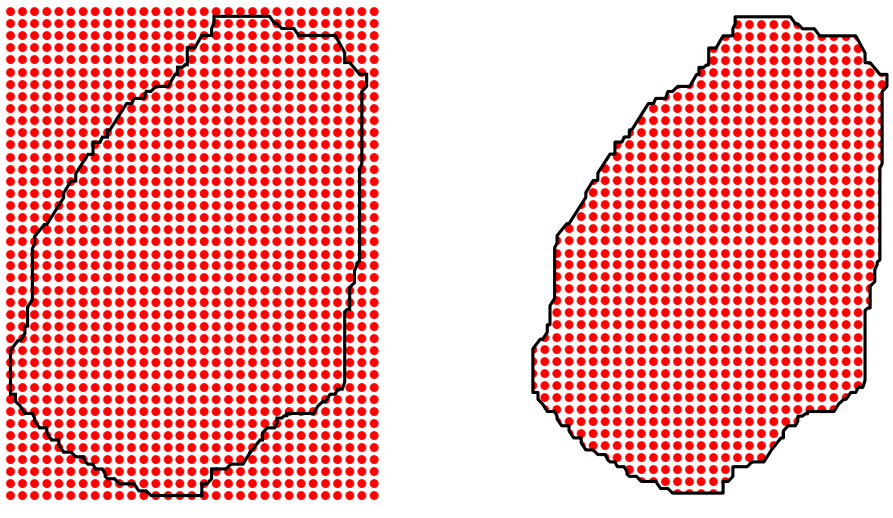

Figure 4.20. Reference Grid for Bodmin

ref, is shown on the left in Figure 4.20.[19] (In Section 8 of the Appendix to Part I a procedure is developed for obtaining this full grid representation directly in MATLAB.) While all of these grid points are used in the calculation, those outside of the Bodmin boundary are only relevant for maintaining some degree of smoothness in the interpolation constructed below. On the right, these grid points have been masked out in order to display only those points inside the Bodmin boundary. (The construction of such visual masks is quite useful for many displays, and is discussed in detail in Section 1.2.4 of Part IV in this NOTEBOOK.)

Given this reference grid, **ref**, the extension of **k_count_loc.m** that utilizes **ref** is operationalized in the MATLAB program, **k_count_loc_ref.m**. This program is essentially identical to **k_count_loc.m** except that **ref** is a new input. Here one obtains p-values for Bodmin at each reference point in **ref** with the command:

[19] Notice that the right side and top of the grid extend slightly further than the left and bottom. This is because the **Xmax** and **Ymax** values in the program are adjusted upward to yield an integral number of cells of the same size.

>> [Pval,C0] = k_count_loc_ref(Bodmin,ref,999,D,1,Bod_poly);

where the matrix **Pval** now contains one p-value for each grid point in **ref** and distance radius in **D**. The results of this Monte Carlo simulation were exported to ARCMAP and the p-values at each grid point inside Bodmin are displayed for $h = 3$ *km* on the left in Figure 4.21 below (again with a mask). By comparing this with the associated point

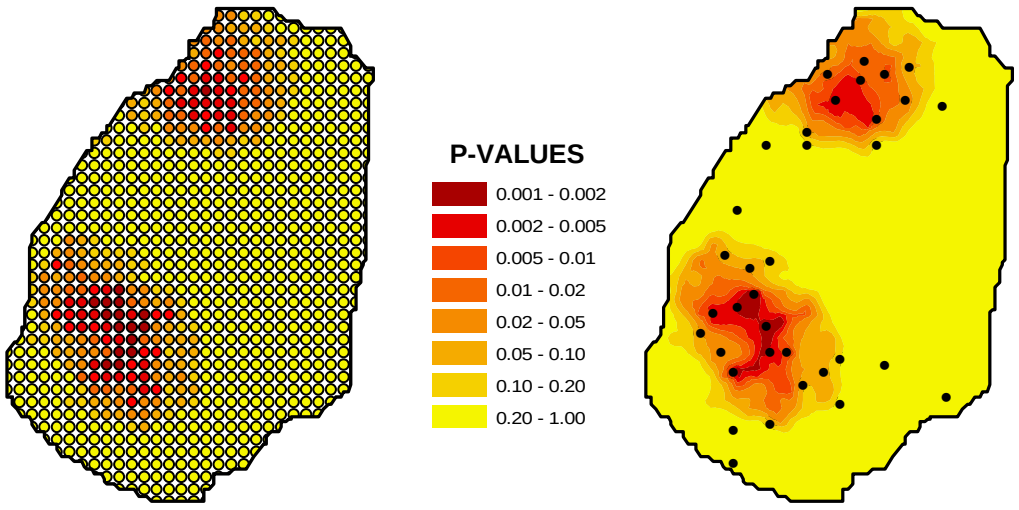

Figure 4.21. Interpolated P-Values for Bodmin

plot in the center of Figure 4.18, one can see that this is essentially a smoother version of the results depicted there. However, this representation can be considerably improved upon by interpolating these values using any of number of standard "smoothers" (discussed further in Part II). The interpolation shown on the right was obtained by the method known as *ordinary kriging.* This method of (stochastic) interpolation will be developed in detail in Section ??? of Part II in this NOTEBOOK.

4.8.3 Local Tests of Nonhomogeneous CSR Hypotheses

Next we extend these methods to the more general case of *nonhomogeneous* CSR hypotheses. As with all spatial Monte Carlo testing procedures, the key difference between the homogeneous and nonhomogeneous cases is the way in which random points are generated. As discussed in Section 4.7.2 above, this generation process for the nonhomogeneous case amounts to a *two-stage sampling procedure* in which a polygon is first sampled in a manner proportional to the given reference measure, **M**, and then a random location in this polygon is selected. Since this procedure is already incorporated into both the programs **k_count_loc.m** and **k_count_loc_ref.m** above, there is little need for further discussion at this point.

By way of illustration, we now apply **k_count_loc_ref.m** to a Philadelphia data set, which includes 500 incidents involving *inter-group conflict* (IGC) situations (such as housing discrimination) that were reported to the Community Service Division of the Philadelphia Commission on Human Relations from 1995-1996. [This data set is discussed in more detail in the project by Amy Hillier on the ESE 502 class web page.]

The locations of these 500 incidents are shown on the left in Figure 4.22 below, and are also displayed in the map document, **Phil_igc.mxd**, in ARCMAP. Here the natural null hypothesis would be that every individual has the same chance of reporting an "incident". But as with the housing abandonment example in Figure 4.11 above, individual location data is not available. Hence census tract population levels

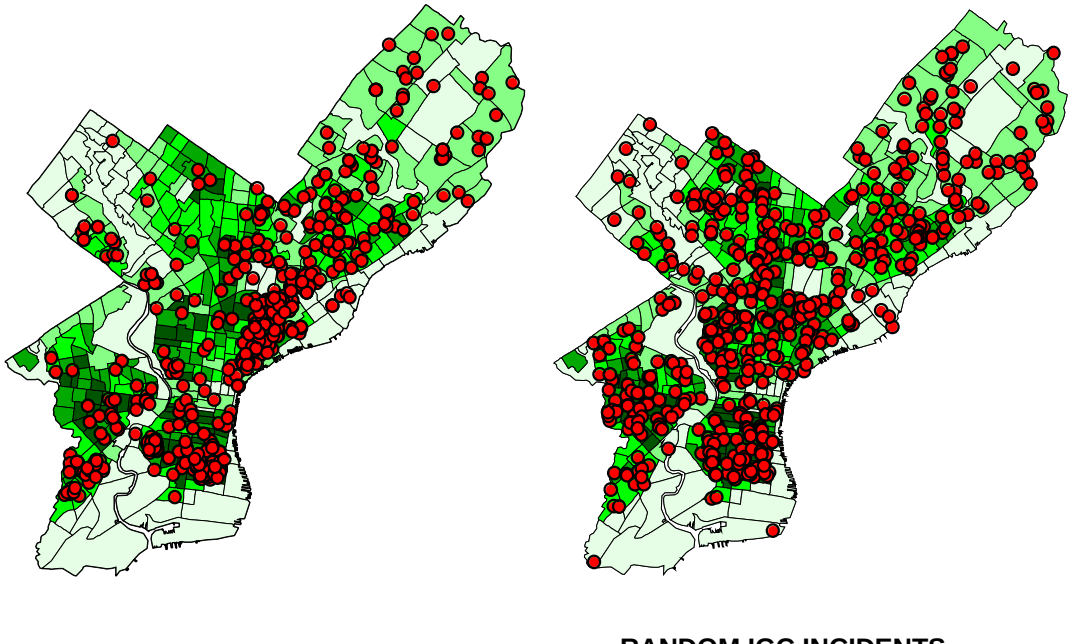

Figure 4.22. Comparison with IGC Random Incidents

will be used as an approximation to individual locations, so that the relevant *reference measure* is here taken to be population by census tract (with corresponding population densities shown in green in Figure 4.22). The relevant *nonhomogeneous CSR hypothesis* for this case is thus simply that the chance of any incident occurring in a given census tract is proportional to the population of that census tract. Under this hypothesis, a typical realization of 500 "random IGC incidents" is shown on the right. Here it is clear that incidents are more clustered in areas of high population density, such as in West Philadelphia and South Philadelphia. So clusters of actual data on the

left are only *significant* if they are more concentrated than would be expected under this hypothesis. Hence, even though there is clearly a cluster of cases in South Philadelphia, it is *not* clear that this is a significant cluster. Notice however that the Kensington area just Northeast of Center City does appear to be more concentrated than would be expected under the given hypothesis. But no conclusion can be reached on the basis of this visual comparison. Rather, we must simulate *many* realizations of random patterns and determine statistical significance on this basis.

To do so, a *reference grid* for Philadelphia was constructed, and is shown (with masking) on the left in Figure 4.23 below, in a manner similar to Figure 4.20 above. Here a range of distances was tried, and clustering was most apparent at a radius of 500 *meters* (in a manner similar to the radius of 3 *km* in Figure 4.18 above for the Bodmin example). The p-value results for this case are contained in the MATLAB workspace,

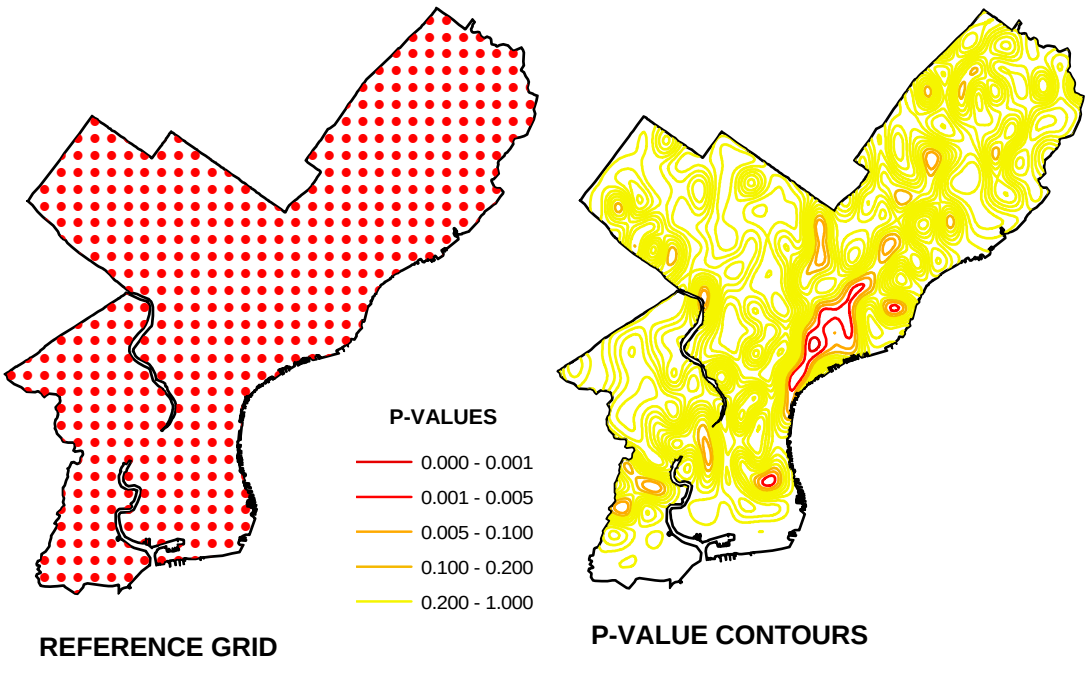

Figure 4.23. P-Value Map for ICG Clustering

phil_igc.mat, and were obtained using **k_count_loc_ref.m** with the command:

>> [Pval,C0] = k_count_loc_ref(loc,ref,999,D,pop,bnd);

Here **loc** contains the locations of the 500 IGC incidents, **ref** is the reference grid shown above, **D** contains a range of distances including the 500-meter case,[20] and **pop**

[20] The actual coordinates for this map were in *decimal degrees*, so that the value .005 corresponds roughly to 500 meters.

contains the populations of each census tract, with boundaries given by **bnd**. These results were imported to ARCMAP as a point file, and are displayed as **P-val.shp** in the data frame, "P-Values for Dist = .005", of **Phil_igc.mxd**. Finally, these p-values were interpolated using a different smoothing procedure than that of Figure 4.21 above. Here the spline interpolator in **Spatial Analyst** was used, together with the contour option. The details of this procedure are described in Section 8 of the Appendix to Part I.[21]

Here the red contours denote the most significant areas of clustering, which might be interpreted as IGC "hotspots". Notice in particular that the dominant hotspot is precisely the Kensington area mentioned above. Notice also that the clustering in West Philadelphia, for example, is now seen to be explained by population density alone, and hence is not statistically significant.

It is also worth noticing that there is a small "hotspot" just to the west of Kensington (toward the Delaware River) that appears hard to explain in terms of the actual IGC incidents in Figure 4.22. The presence of this hotspot is due to the fact that while there are only four incidents in this area, the population density is less than a quarter of that in the nearby Kensington area. So this incidence number is usually high given the low density. This raises the practical question of how many incidents are required to constitute a meaningful cluster. While there can be no definitive answer to this question, is important to emphasize that statistical analyses such as the present one should be viewed as providing only one type of useful information for cluster identification. [22]

[21] Notice also that this contour map of P-values is an updated version of that in the graphic header for the class web page. That version was based on only 99 simulations (run on a slower machine).

[22] This same issue arises in regression, where there is a need to distinguish between the *statistical* significance of coefficients (relative to zero) and the *practical* significance of their observed magnitudes in any given context.