2. Modeling the Spatial Structure of Areal Units

Aside from data aggregations, the second major difference between continuous and areal data models concerns the representation of *spatial structure* itself. In particular, while "distance between points" for any given units of measure (straight-line distance, travel distance, travel time, etc.) is fairly unambiguous, the same is not true for "distance between areal units". As mentioned above, the standard convention here is to identify representative points for areal units, the most typical being areal *centroids* (as defined formally below). In fact, these centroids serve as the default option in ARCMAP for constructing such representative points [refer to Section 1.2.9 in Part IV of this NOTEBOOK]. But in spite of the fact that these points constitute the so-called "geometric centers" of each areal unit, they can sometimes be quite misleading in terms of distance relations between areal units.

An example is given in Figure 1.13 below, which involves three areal units, R_1, R_2, and R_3. Here it might be argued that since units R_2 and R_3 are spatially separated, but are each adjacent to R_1, they are both "closer" to (or exhibit a "stronger tie" to) unit R_1 than to each other. However, the centroids of these three units, shown by the black dots in Figure 1.14, are *equidistant* from one another. Thus all of these spatial relations are lost when "closeness" is summarized by centroid distances.
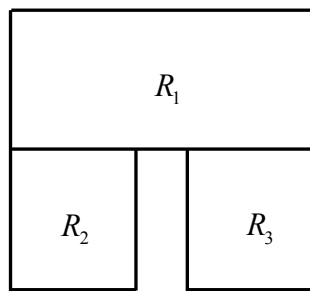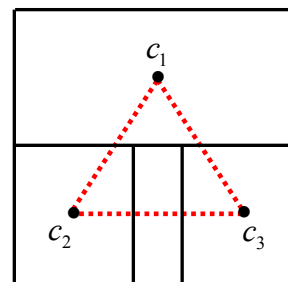

Figure 1.13. Areal Units   **Figure 1.14. Centroid Distances**

In particular, this suggests that the *shapes* of areal units also contain important information about their relative proximities, even though they are much more difficult to quantify. We shall return to this question below.

In addition to these geometric issues, there are other non-spatial properties of areal units that influence their "closeness" in terms of human interactions. For example, it is often observed that the opposite coasts in the US are relatively "close" to one another in terms of human interactions (such as phone calls or emails). More generally, there tends to be more interaction between states with large cities (such as those shown in Figure 1.15) than would be expected on the basis of their separation in geographical space. For example, such cities tend to contain relatively large professional populations conducting business between cities.

Figure 1.15. Cities above 500,000
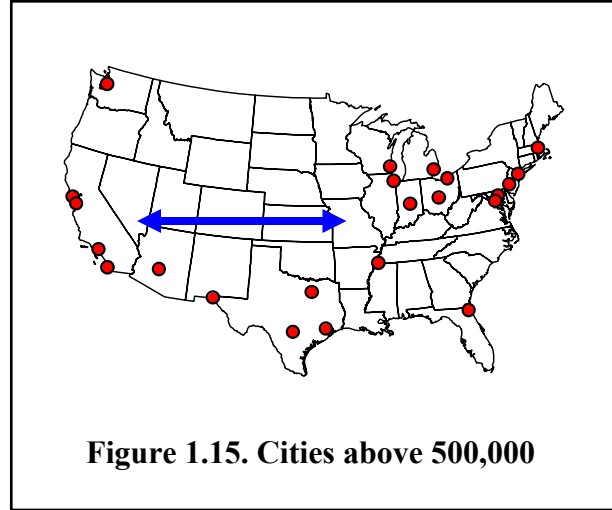
But while such socio-economic linkages between areal units may indeed be relevant for many applications, we shall restrict our present analysis to purely geometric notions of "closeness". The main justification for this is that we are primarily interested in modeling *unobserved residual effects* in regression models involving areal units. So these measures of closeness are designed solely to capture possible *spatial autocorrelation* effects. Indeed, it can be argued that potentially relevant socio-economic interactions between units (such a communication and travel flows) should be part of the *model*, and not the residuals.

2.1 Spatial Weights Matrices

To model spatial relations between areal units, we now let *n* denote the number of units to be considered, so that the region of interest, say $R =$ Continental US, is partitioned into areal units, $R = \{R_i : i = 1,..,n\}$, say the $n = 48$ states in R (as in Figure 1.15 above). Our basic hypothesis is that the relevant degree of "closeness" or "proximity" of each areal unit R_j to unit R_i (or alternatively, the "spatial influence" of R_j on R_i) can be represented by a *numerical weight*, $w_{ij} \geq 0$. where higher values of w_{ij} denote higher levels of proximity or spatial influence. Under this hypothesis, the full set of such spatial relations can be represented by a single nonnegative *weight matrix*:

$$(2.1.1) \qquad W = \begin{pmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{pmatrix}$$

Notice in particular that while the distance between a *point* and itself is naturally zero, this need not be true for areal units. For example, if w_{ij} were to represent the average distance between all cities in states i and j (possibly weighted by population sizes) then since the average distance between cities within each state i is certainly positive, one

must have $w_{ii} > 0$ for all $i = 1,..,n$ .. So in general, *any* nonnegative matrix can be a spatial weights matrix.

However, certain special structural properties of such matrices are quite common. For example, if distance itself is measured symmetrically, i.e., if $d(x,y) = d(y,x)$ for all locations *x* and y (as with Euclidean distance), then weight measures such as the average distance between cities in states i and j will also be symmetric, i.e., $w_{ij} = w_{ji}$. So, much like covariance matrices, many spatial weights matrices will be *symmetric* matrices.

Moreover, while diagonal weights, w_{ii}, can in principle be positive (as in the city example above), it will often be convenient for analysis to set $w_{ii} = 0$ for all $i = 1,..,n$. In particular, when w_{ij} is taken to reflect some notion of the "spatial influence" of unit *j* on unit i, then we set $w_{ii} = 0$ in order to avoid "self-influence". This will become clearer in the development of spatial autoregression models in Section 3 below.

Many of the most common spatial weights are based on distances between point representations of areal units. So before developing these weight functions, it is convenient to begin with a more detailed consideration of point representations themselves.

2.1.1 Point Representations of Areal Units

If distances between areal units, R_i, $i = 1.,,n$, are to be summarized by distances between representative "central" points, $c_i \in R_i$, then it is natural to require that c_i be "close" to all other points in R_i. This leads to certain well posed mathematical definitions of such representative points.[1] Perhaps the simplest is the "spatial median" of an areal unit, R, which is which is defined to be the point, c, with minimum average distance to all points in R. If the area of R is denoted (as in Section 2.1 of Part I) by

$$(2.1.2) \qquad a(R) = \int_R dx \quad,$$

then the *spatial median*, c, of R (with respect to Euclidean distance) is given by the solution to

$$(2.1.3) \qquad \boxed{\min_c \tfrac{1}{a(R)} \int_R \| x - c \| dx}$$

But while this point is well defined and is easily shown (from the convexity properties of this programming problem) to be unique, it is not identifiable in closed form. Even if R is approximated by a finite grid of points, the solution algorithms for determining spatial

---

[1] Here we ignore other possible reference points (such as the capital cities of states or countries) that might be relevant in certain applications.

medians are computationally intensive. For this reason, we shall not use spatial medians for reference points. However, it is still of interest to note that if R were approximated by some finite grid of points, $R_n = \{x_i : i = 1,..,n\}$, (say the set of raster pixels inside an ARCMAP representation of R), then the spatial median of this set, R_n, can in fact be calculated in ARCMAP using the **ArcToolbox** command: *Spatial Statistics Tools > Measuring Geographic Distributions > Median Center*.

Spatial Centroids

But in view of these computational complexities, a far more popular choice is the spatial "centroid" of R, which minimizes the average *squared distance* to all points in R. More formally, the *centroid*, c, of R is given by the solution to:

(2.1.4) $\boxed{\min_c \frac{1}{a(R)} \int_R \| x - c \|^2 dx}$

The advantage of using squared distances is that this minimization problem is actually solvable in closed form. In particular, by recalling that $\| x - c \|^2 = x'x - 2x'c + c'c$, and that the minimum of (2.1.4) is given by its first-order conditions [as for example in Section A2.7 of the Appendix to Part II], we see in particular that

(2.1.5) $0 = \nabla_c \left[\frac{1}{a(R)} \int_R (x'x - 2x'c + c'c) dx \right]$

$\Rightarrow \quad 0 \ = \ \int_R \nabla_c (x'x - 2x'c + c'c) dx$

$= \ \int_R (-2x + 2c) dx \ = \ 2\left[-\int_R x \, dx + c \int_R dx \right]$

$\Rightarrow \int_R x \, dx \ = \ c \int_R dx \ = \ c \, a(R)$

$\Rightarrow \boxed{c \ = \ \frac{1}{a(R)} \int_R x \, dx}$

which is simply the *average* over all locations, $x \in R$. So the coordinate values of $c = (c_1, c_2)$ are precisely the average values of coordinates, $x = (x_1, x_2)$, over R. In more practical terms, if one were to approximate R by a finite grid of points, $R_n = \{x_i : i = 1,..,n\}$, in R as mentioned above for spatial medians, then the centroid coordinates, $c = (c_1, c_2)$, are well approximated by

(2.1.6) $\boxed{c_i \approx \frac{1}{n} \sum_{x \in R_n} x_i \ , \ i = 1, 2}$

For this reason, the centroid of R is also called the *spatial mean* of R. Such spatial means can be calculated (for finite sets of points) in ARCMAP using the **ArcToolbox command:** *Spatial Statistics Tools > Measuring Geographic Distributions > Mean Center*.

Computation of Centroids

But while this view of centroids is conceptually very simple and intuitive, there is in fact a much more efficient and exact way to calculate centroids in ARCMAP. In particular, since areal units R are defined as *polygon features* with finite sets of vertices (in a manner paralleling the matrix representations of polygon boundaries in MATLAB discussed in Section 3.5 of Part I), one can actually calculate the exact centroids of these polygons with rather simple geometric formulas. Since the derivation of these formulas is well beyond the scope of these notes, we simply record them for completeness.[2] If we proceed in a clockwise direction around a given polygon, R, and denote its vertex points by (x_{1i}, x_{2i}), $i = 1, .., n$ [where by definition, $(x_n, y_n) = (x_1, y_1)$], then the *area* of R is given by

$$(2.1.7) \qquad \boxed{a(R) = \tfrac{1}{2} \sum_{i=1}^{n-1} (x_{1,i+1} x_{2i} - x_{1i} x_{2,i+1})}$$

and the *centroid coordinates*, $c = (c_1, c_2)$, are given by

$$(2.1.8) \qquad \boxed{c_j = \frac{1}{6\,a(R)} \sum_{i=1}^{n-1} (x_{1i} + x_{2i})(x_{1,i+1} x_{2i} - x_{1,i} x_{2,i+1}) \;,\; j = 1, 2}$$

These formulas are implemented in the MATLAB program, **centroids_areas.m**. If the boundary file (in MATLAB format) for a given system of areal units, $R = \{R_i : i = 1, .., n\}$, is denoted by **bnd_R**, then the $n \times 2$ matrix, **C**, of centroid coordinates and the n-vector, **A**, of corresponding areas can be obtained with the command,[3]

>> [C,A] = centroids_areas(bnd_R);

These are precisely the same formulas used for calculating areas and centroids in the "Calculate Geometry" option in ARCMAP, using the procedures outlined in Sections 1.2.8 and 1.2.9 of Part IV of this NOTEBOOK.

Displaying Centroids

One can display these centroids in ARCMAP by opening the Attribute Table containing the centroids calculated above and using **Table Options > Export…** to save this table as

---

[2] Full derivations of (2.7) and (2.8) require an application of Green's Theorem, and are given in expressions (31),(33) and (34) of Steger (1996). Here it should be noted that the signs in Steger are reversed, since it is there assumed that vertices proceed in a counterclockwise direction.

[3] A more general MATLAB program of this type can be downloaded at the web site:
http://www.mathworks.com/matlabcentral/fileexchange/319-polygeom-m.

_____

say **centroids.dbf**. When prompted to add this data to the existing map, click **OK**. If you right click on this new entry in the Table of Contents and select **Display XY Data**, then the centroids will now appear on the map. If you wish to save these centroids, right click on the new "centroid events" entry in the Table of Contents and use **Data > Export Data**. Finally, if you save to the map as **centroids.shp**, then you can edit this copy as a permanent file. This procedure was carried out for the Eire map in Figure 1.7 above, and is shown in Figure 1.16 below.
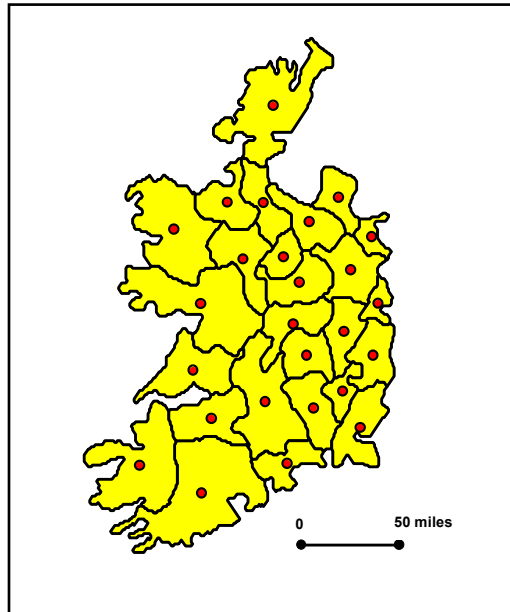

Figure 1.16. Eire Centroids

Before proceeding to spatial weights based on centroid distances, it is important to stress some of the limitations of this centroid-distance characterization of closeness between areal units. As was illustrated in Figures 1.13 and 1.14 above, such point representations can often ignore important shape relations between areal units. As seen in the Eire case, for example, the actual boundary relations among these counties are quite complex. In addition, while we usually refer to the centroid of a given areal unit by writing, $c_i \in R_i$, it *not* necessarily true that point, c_i, is actually an element of R_i. This is obvious for cases like the state of Hawaii, where the relevant areal unit is itself a string of disconnected islands. But in fact such problems may exist even for *spatially connected* areal units, such as the example of a "river shore" area, R, shown (in yellow) in Figure 1.17 below. Here the centroid of this area (shown in red) is not only outside of R, but is actually on the other side of the river. So it is important to remember that while such locations are indeed closest (in squared distance) to all points of R, the *shape* of R itself may dictate that such locations lie outside of R. However, it must also be stressed that these are very exceptional cases. Indeed, while the county boundaries in Eire are very complex, each centroid in Figure 1.16 is seen to be contained in its respective county.
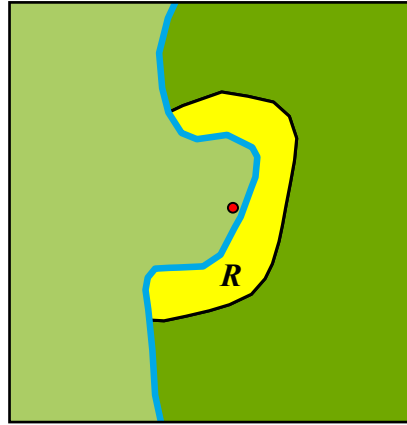
_____

Figure 1.17. Exterior Centroid Example

2.1.2 Spatial Weights based on Centroid Distances

While we shall implicitly assume that point representations, c_i, of areal units, R_i, are based on *centroids*, the following definitions hold intact for any relevant sets of points (such as state capitals or county seats).   Moreover, while *centroid distances*, $d_{ij} = d(c_i, c_j)$, are implicitly assumed to be Euclidean distances, $\| c_i - c_j \|$, the present definitions of spatial weights are readily extendable to other relevant notions of distance (such as travel distance or travel time). But it should also be stressed that our present conventions are in fact used in most areal data analyses. The following examples of spatial weights based on centroid distances extend the list given in [BG. p.261].

k-Nearest-Neighbor Weights

Recall from Section 3.2 in Part I that the nearest-neighbor distances defined within and between point patterns are readily extendable to centroid distances. However, such distance relations can be very restrictive for modeling spatial relations between areal units. This is again well illustrated by the Eire example above, where the neighbors of Laoghis county are shown in Figure 1.18 below.
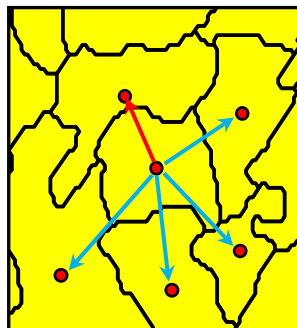

Figure 1.18. Nearest Neighbors Example

Here it turns out that the nearest neighbor to Laoghis county in centroid distance is Offaly county to the north (shown by the red arrow). But it is clear that the neighbors adjacent to Laoghis county in all other directions may be of equal importance it terms of spatial relations. We shall be more explicit about such adjacency relations below. But in the present case, it is clear that we can achieve the same effect by considering the *five* nearest neighbors to this county. (See Section 7.5.1 below for further discussion of this example.)

So to formalize such multiple-neighbor relations, let the centroid distances from each areal unit i to all other units $j \neq i$ be ranked as follows: $d_{ij(1)} \leq d_{ij(2)} \leq \cdots \leq d_{ij(n-1)}$. Then for each $k = 1,..,n-1$, the set $N_k(i) = \{j(1), j(2),.., j(k)\}$ contains the k areal units closest to i (where for simplicity we ignore ties). For each given value of k, the *k-nearest neighbor* weight matrix, W, is then defined to have spatial weights of the form:

(2.1.9)
$$w_{ij} = \begin{cases} 1 & , \ j \in N_k(i) \\ 0 & , \ otherwise \end{cases}$$

Note in particular that the values of w_{ij} for the *k*-nearest neighbors of i are higher than for other areal units, signifying that these neighbors are deemed to have greater proximity to i (or greater spatial influence on i) than other spatial units. Similar conventions will be used for all weights discussed below. Note also that the common value of these weights implicitly assumes that levels of proximity or influence are the same for all *k*-nearest neighbors. This constancy assumption will be relaxed for other types of spatial weights.

Before proceeding to other weighting schemes, it is also important to note that such nearest-neighbor relations are generally *asymmetric* in nature. For if j is a *k*-nearest neighbor of i, then it need not be true that i is a *k*-nearest neighbor of j, i.e., one may have $w_{ij} \neq w_{ji}$. As seen in Figure 1.19 below, this is true even for $k = 1$, where R_2 is the nearest neighbor of R_1, but R_3 is the nearest neighbor of R_2:
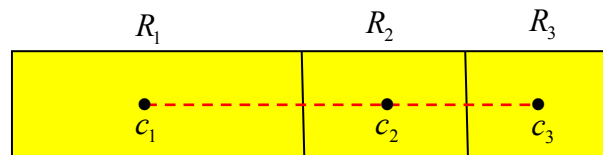

Figure 1.19. Asymmetric Nearest Neighbors

But in some applications it might be argued that as long as either i or j is an "influential neighbor" of the other, then i and j are "spatially related" in this sense. This *symmetric k*-nearest neighbor relation can be formalized as follows:

_____

(2.1.10)
$$w_{ij} = \begin{cases} 1 & , \quad j \in N_k(i) \; or \; i \in N_k(j) \\ 0 & , \qquad\quad otherwise \end{cases}$$

Radial Distance Weights

In some cases, distance itself is an important criterion of spatial influence. For example, locations "within walking distance" or "within one-hour driving distance" may be relevant. Such proximity criteria are usually more relevant for comparing actual point locations (such as distances to shopping opportunities or medical services), but are sometimes also used for areal data. If d denotes some *threshold distance* (or *bandwidth*) beyond which there is no "direct spatial influence" between spatial units, then the corresponding *radial distance* weight matrix, W, has spatial weights of the form:
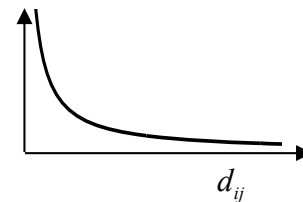
(2.1.11)
$$w_{ij} = \begin{cases} 1 & , \quad 0 \le d_{ij} \le d \\ 0 & , \quad d_{ij} > d \end{cases}$$
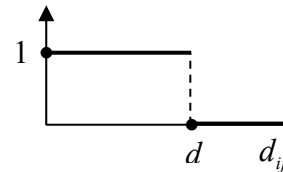
Power-Distance Weights

In the radial distance weights above there is no diminishing effect of distance up to threshold d. However, if there are believed to be diminishing effects, one common approach is to assume that weights are a *negative power function* of distance of the form

(2.1.12)
$$w_{ij} = d_{ij}^{-\alpha}$$

where α is some positive exponent, typically $\alpha = 1$ (as in the graph) or $\alpha = 2$. Note that expression (2.1.12) is precisely the same as expression (5.2.4) in the interpolation discussion of Section 5.2 in Part II. Thus all of the discussion in that section is relevant here as well.

Exponential-Distance Weights

As in expression (5.2.5) of Part II, the *negative exponential* alternative to negative power functions is also relevant here, and is again defined by:

_____

(2.1.13) $\boxed{w_{ij} = \exp(-\alpha\, d_{ij})}$
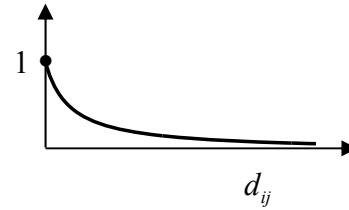
for some positive exponent, α (such as $\alpha = 1$ in the graph). As discussed in that section, the negative exponential version is better behaved for short distances, but converges rapidly to zero for larger distances.

Double-Power-Distance Weights

A somewhat more flexible family incorporates finite bandwidths with "bell shaped" taper functions.   If d again denotes the maximum radius of influence (*bandwidth*) then the class of *double-power* distance weights is defined for each positive integer *k* by

(2.1.14) $\boxed{w_{ij} = \begin{cases} \left[1 - \left(d_{ij}/d\right)^{k}\right]^{k} & , \ 0 \le d_{ij} \le d \\ 0 & , \quad d_{ij} > d \end{cases}}$
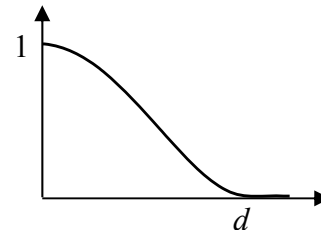
where typical values of k are 2, 3 and 4. Note that w_{ij} falls continuously to zero as d_{ij} approaches d, and is defined to be zero beyond d. The graph shows the case of a *quadratic distance function* with $k = 2$ (see also [BG, p.85]).

2.1.3 Spatial Weights Based on Boundaries

The advantage of the distance weights above is that such distances are easily computed. But in many cases the *boundaries* shared between spatial units can play in important role in determining degree of "spatial influence". The case of Eire in Figures 1.16 is a good example. In particular, recall that *k*-nearest-neighbor weights were in fact motivated by an effort to capture the counties surrounding Laoghis county in Figure 1.18. But such neighbor distances can at best only approximate spatial contiguity relations (especially since areal units can each have different numbers of contiguous neighbors). A better approach is of course to identify such contiguities directly. The main difficulty here is that the identification of contiguities requires the manipulation of boundary files, which are considerably more complex than simple point coordinates. We shall return to this issue in Section 2.2.2 below. But for the moment, we focus on the formal task of defining contiguity relations.

Spatial Contiguity Weights

The simplest contiguity weights indicate only whether pairs of areal units share a boundary or not. If the set of boundary points of unit R_i is denoted by $bnd(i)$ then the so-called *queen contiguity weights* are defined by

(2.1.15)
$$w_{ij} = \begin{cases} 1 & , \ bnd(i) \cap bnd(j) \neq \varnothing \\ 0 & , \ bnd(i) \cap bnd(j) = \varnothing \end{cases}$$

However, this allows the possibility that spatial units share only a single boundary point (such as a corner point shared by diagonally adjacent cells on a chess board).[4] Hence a stronger condition is to require that some *positive* portion of their boundary be shared. If l_{ij} denotes the *length* of shared boundary, $bnd(i) \cap bnd(j)$, between i and j, then these so-called *rook contiguity weights* are defined by

(2.1.16)
$$w_{ij} = \begin{cases} 1 & , \ l_{ij} > 0 \\ 0 & , \ l_{ij} = 0 \end{cases}$$

A simple example of a contiguity weight matrix, W, is given in expression (2.1.22) below.

Shared-Boundary Weights

As a sharper form of comparison, note that if l_i defines the *total boundary length* of $bnd(i)$ that is shared with other spatial units, i.e., $\Sigma_{j \neq i} \, l_{ij}$, then fraction of this length shared with any particular unit j is given by l_{ij} / l_i. These fractions themselves yield a potentially relevant set of *shared boundary weights*, defined by

(2.1.17)
$$w_{ij} = \frac{l_{ij}}{l_i} = \frac{l_{ij}}{\sum_{k \neq i} l_{ik}}$$

---

[4] In fact, the present use of the terms "queen" and "rook" in expressions (2.1.15) and (2.1.16) refer precisely to the possible moves of queen and rook pieces on a chess board, where rooks can only move through faces between adjacent squares, but the queen can also move diagonally through corners.

2.1.4 Combined Distance-Boundary Weights

Finally, it should be evident that in many situations spatial closeness or influence may exhibit aspects of *both* distance and boundary relations. One classical example of this is given in the original study of spatial autocorrelation by Cliff and Ord (1969). In analyzing the Eire blood-group data, they found that the best weighting scheme for capturing spatial autocorrelation effects was given by the following combination of *power-distance* and *boundary-shares*,

(2.1.18)
$$w_{ij} = \frac{l_{ij}d_{ij}^{-\alpha}}{\sum_{k \neq i} l_{ik}d_{ik}^{-\alpha}}$$

with simple inverse-distance, $\alpha = 1$. We shall return to this example in Section 7.5 below.

2.1.5 Normalizations of Spatial Weights

Having defined a variety of spatial weights, we next observe that for modeling purposes it is generally convenient to normalize these weights in order to remove dependence on extraneous scale factors (such as the particular units of distance employed in exponential and power weights). Here there are two standard approaches:

Row-Normalized Weights

Recall that the ith row of W contains all spatial weights influencing areal unit, i, namely $(w_{ij} : j = 1,..,n)$ [possibly with $w_{ii} = 0$]. So if the positive weights in each row are normalized to have unit sum, i.e., if

(2.1.19)
$$\sum_{j=1}^{n} w_{ij} = 1 \ , \ i = 1,..,n$$

then this produces what called the *row normalization* of W.[5] Note that each row-normalized weight, w_{ij}, can then be interpreted as the *fraction* of all spatial influence on unit i attributable to unit j. The appeal of this interpretation has led to the current wide-spread use of row-normalized weight matrices. In fact, many of the spatial weight definitions above are often implicitly defined to be row normalized. The most obvious example is that of shared boundary weights in (2.1.17), which by definition are seen to be

---

[5] In cases where $w_{ii} = 0$ by definition, it is possible that isolated units, i, may have *all-zero* rows in W. So condition (2.1.19) is only required to hold for those rows, i, with $\Sigma_j w_{ij} > 0$.

row normalized. [Also the combined example in (2.1.18) was defined by Cliff and Ord (1969) to be in row-normalized form.] Another simple example is provided by the *k-nearest neighbor weights* in (2.1.9) above, which are often defined using weights $1/k$ rather than 1 to ensure row normalization. A more interesting example is provided by the *power distance weights* in (2.1.12) which have the row-normalized form,

$$(2.1.20) \qquad w_{ij} = \frac{d_{ij}^{-\alpha}}{\sum_{k \neq j} d_{ik}^{-\alpha}}$$

These normalized weights are seen to be precisely the *Inverse Distance Weighting* (IDW) scheme employed in Spatial Analyst for spatial interpolation (as mentioned in Section 5.2 of Part II). A similar example is provided by *exponential distance weights*, with row-normalized form,

$$(2.1.21) \qquad w_{ij} = \frac{\exp(-\alpha \, d_{ij})}{\sum_{k \neq j} \exp(-\alpha \, d_{ik})}$$

These weights are also used for spatial interpolation. In addition, it should be noted that such normalized weights are commonly used in spatial interaction modeling, where (2.1.20) and (2.1.21) are often designated, respectively, as *Newtonian* and *exponential* models of spatial interaction intensities or probabilities.

Scalar Normalized Weights

In spite of its popularity, row-normalized weighting has its drawbacks. In particular, row normalization alters the internal weighting structure of W so that comparisons between rows become somewhat problematic. For example, consider *spatial contiguity weighting* with respect to the simple three-unit example shown below:

$$(2.1.22) \qquad \boxed{R_1 \mid R_2 \mid R_3} \qquad W = \begin{pmatrix} 0 & w_{12} & w_{13} \\ w_{21} & 0 & w_{23} \\ w_{31} & w_{32} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

As represented in the contiguity weight matrix, W, on the right, unit 2 is influenced by both 1 and 3, while units 1 and 3 are each influenced only by the single unit 2. Hence it might be argued that 2 is subject to *more* spatial influence than either 1 or 3. But row

normalization of W changes this relation, as seen by its row-normalized form, W_{rn}, below:

$$(2.1.23) \qquad W_{rn} = \begin{pmatrix} 0 & 1 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 1 & 0 \end{pmatrix}$$

Here the "total" influence on each unit is by definition the same, so that unit 1 now influences 2 only "half as much" as 2 influences 1. While the exact meaning of "influence" is necessarily vague in most applications, this effect of row-normalization can hardly be considered as neutral.[6]

In view of this limitation, it is natural to consider simple *scalar* normalizations, where W is multiplied by a single number, say $\alpha \cdot W$, that removes any measure-unit effects but preserves relations between all rows of W. For example, if w_{max} denotes the largest element of matrix, W, then the choice,

$$(2.1.24) \qquad \boxed{\alpha = \frac{1}{w_{max}} > 0}$$

provides one such normalization that has the advantage of ensuring that the resulting spatial weights, w_{ij}, are all between 0 and 1, and thus can still be interpreted as *relative influence intensities*.

However, for theoretical reasons, it is often more convenient to divide W by the maximum eigenvalue, λ_W, of W (to be discussed in Section 3.3.2 below) and hence to set

$$(2.1.25) \qquad \boxed{\alpha = \frac{1}{\lambda_W} > 0}$$

2.2 Construction of Spatial Weights Matrices

Our primary interest here is to show how spatial weight matrices can be constructed for applications in MATLAB. We begin with those spatial weights based on centroid

---

[6] A more detailed discussion of this problem can be found in Kelejian and Prucha (2010).

distances as in Section 2.1.2 above, and illustrate their construction in MATLAB. Next we consider certain of the spatial contiguity weights in Section 2.1.3, which require initial calculations to be made on shapefiles in ARCMAP.

2.2.1 Construction of Spatial Weights based on Centroid Distances

All spatial weights defined in Section 2.1.2 can be constructed in MATLAB using the program **dist_wts.m**. By opening this program, one can see that the inputs include a matrix, **L**, of centroid coordinates together with a MATLAB structure, **info**, containing information about the specific spatial weights desired. The use of this program can be illustrated by an application to the Eire centroid data in the workspace, **eire.mat**. Here **L** is a 26×2 matrix containing the centroid coordinates for the 26 counties in Eire. If one chooses to construct a weight matrix containing the five nearest neighbors for each county, say **W_5nn**, then by looking at the top of the program, one sees that *k-nearest neighbors* corresponds to the first of six types of spatial weights that can be created. In particular, by setting **info.type** = [1,5], one specifies a 5-nearest-neighbor matrix. Thus the appropriate commands for this case are given by:

>> info.type = [1,5];
>> W_nn5 = dist_wts(L,info);

To understand the matrix which is produced, we again consider the case of Laoghis county in Figure 1.18 above. By using the **Identify** tool in ARCMAP, one sees that the FID of Laoghis county is **10**, so that its centroid coordinates correspond to row **11** in **L** (remember that FID numbers start at **0** rather than **1**). Similarly, one can verify that the five surrounding counties (which are also its five nearest neighbors) have FID values **(0,8,9,18,21).** So their row numbers in **L** are given by **(1,9,10,19,22).** These numbers should thus correspond to the "1" values in row **11**. This can be verified by displaying the positive column numbers of all positive elements in row number **11** of **W_nn5** using the **find** command in MATLAB as follows:

>> find(W_nn5(11,:) > 0)

ans =

  1   9   10   19   22

It is also important to emphsize that this matrix is constructed to be in *sparse* form, which means that only nonzero values are recorded. This can be seen by attempting to display the first 5 rows and columns of **W_nn5** as follows:

>> W_nn5(1:5,1:5)

ans =

  (5,2)   1

The result displayed says that the only nonzero element here is in (row **5**, column **2**) and has value **1**. This is a particularly powerful format in MATLAB since spatial weight matrices tend to have many zero values, and can thus be stored and manipulated very efficiently in sparse form. If one wants to obtain a full matrix version of **W_nn5**, say **Wnn5**, then use the command:

>> Wnn5 = full(W_nn5);

The above 5×5 display then yields:

>> Wnn5(1:5,1:5)

ans =

```
  0  0  0  0  0
  0  0  0  0  0
  0  0  0  0  0
  0  0  0  0  0
  0  1  0  0  0
```

and shows in particular that all elements other than (5,2) are indeed zero.

2.2.2 Construction of Spatial Weights based on Boundaries

As mentioned above, the construction of spatial weights based on boundaries is inherently more complex from a computational viewpoint. While there are a number of available procedures for doing so, we focus here on methods that can be done by combining ARCMAP and MATLAB procedures. In particular, spatial weights matrices based on both queen and rook contiguities [expressions (2.1.15) and (2.1.16), respectively] are directly available in ARCMAP. So our present focus is on how to obtain these results, and import them to MATLAB. Here it should be mentioned that *boundary share weights* [expression (2.1.17)] can also be constructed, but require more complex procedures (as developed in Sections 3.2.2 and 3.2.3 of Part IV).

Here again we use the Eire data as an example, and assume that the shapefile, **Eire.shp**, is currently displayed in ARCMAP. The desired spatial weights can be obtained in **ArcToolbox** using the command sequence:

Spatial Statistics Tools > Modeling Spatial Relationships
                        > Generate Spatial Weights Matrix

(i) In the window that opens, first set:

Inputs Feature Class = "Eire.shp"

by opening the browser on the right and selecting "Eire.shp" from the Home directory.

(ii) One then needs to have a unique identifier for each boundary polygon (county) in Eire. If none are present, then the simplest procedure is to construct a new field, **ID**, calculated as "[FID] + 1" and to set:

Unique ID Field = "ID"

(iii) Here we calculate queen contiguity weights. So to specify the **Output Spatial Weights Matrix File**, open the browser on the right and navigate to directory your own working directory. Set File Name = **Queen_W**, and click Save. The path to this file (**Queen_W.swm**) should now appear in the Output window.

(iv) Queen contiguities are then specified by:

Conceptualization of Spatial Relationships = "CONTIGUITY_EDGES_CORNERS"

Note that a number of other spatial weight matrices can also be constructed:

"CONTIGUITY_EDGES_ONLY" = rook weights (2.1.16)

'k_NEAREST_NEIGHBORS" = *k*-nearest neighbors (2.1.9)

"FIXED_DISTANCE" = radial distance (2.1.11)

"INVERSE_DISTANCE" = power distance (2.1.12) [with exponent options]

► Before leaving this window be *sure* to remove the check on **Row Standardization**, unless you want row standardized values.

(v) Now click **OK** and the file **Queen_W.swm** should appear in the directory specified.

►Note this file is a *binary* file that is only useful inside ARCMAP. To use this data in MATLAB, it must be transformed into a suitable text file. To do so:

(i) Again in **ArcToolbox**, start with the command sequence:

Spatial Statistics Tools > Utilities > Convert Spatial Weights Matrix to Table

(ii) In the window that opens, set:

Input Spatial Weights Matrix File = "Path/Queen_W.swm"

The **Output Table** location will then be automatically specified, and you may ignore this.

(iii) Click **OK**, and the file **Queen_W_Table** should appear in the ARCMAP Table of Contents.

(iv) To convert this to an EXCEL file, open

ArcToolbox > Conversion Tools > Excel > Table to Excel

In the **Input Table**, this file should now appear as one of the options. Click on it, and in the Output Excel File window, remove the default location and use the path to your own working directory and save as **Queen_W.xls**.

(v) Open the file in EXCEL, and delete the first column (containing zeros), so that only three columns remain ("ID" "NID" "WEIGHT"). The save again.

(vi) To import this file to MATLAB, use

Home > Import Data

and open **Queen_W_Table.xls**.

(vii) In the IMPORT Window, change the default "Column vector" setting in the IMPORTED DATA box to "Matrix", and click

Import Selection > Import Data

The file will now appear in the workspace as a 112x3 matrix, **QueenW**. You can rename this as **W_queen** by right clicking on the workspace entry.

As a check to be sure this procedure was successful, one may compare **W_queen** with the ARCMAP representation. In particular, by repeating the procedure for **W_nn5** above, we now see that:

>> rows = find(W_queen(:,1)==11); W_queen(rows,2)'

ans =

    19   9   1   10   22

so that, as seen in Figure 1.18 above, the five contiguous neighbors to Loaghis county are indeed its five nearest neighbors with respect to centroid distance.