The following mansuscript appeared as,

Pace, R. Kelley, and Ronald Barry, "Fast CARs," *Journal of Statistical Computation and Simulation*, Volume 59, Number 2, 1997, p. 123-147.

We gratefully thank the *Journal of Statistical Computation and Simulation* and copyright owner OPA (Overseas Publishers Association) NV for allowing us to distribute this electronic version of the manuscript.

FAST CARS

By

R. KELLEY PACE
Department of Finance, E.J. Ourso College of Business Administration, Louisiana State University, Baton Rouge, LA 70802 USA

RONALD P. BARRY
Department of Mathematical Sciences, University of Alaska, Fairbanks, AK 99775 USA[*]

This paper develops methods for quickly computing maximum likelihood conditional autoregressions (CARs). By using sparse matrix methods, reorganizing the sum-of-squared errors function to avoid unnecessary calculations, and precomputing a set of determinants, simulations of large CARs become possible. As an illustration of the power of these approaches, a simulation of 250 CARs of 2,905 observations can take fewer than three minutes on a personal computer, despite the necessity of evaluating 100 determinants of 2,905 by 2,905 matrices. The computation of each estimate via examining the profile likelihood sampled at 100 points avoids problems of local optima. Simulating estimates avoids other problems associated with the traditional information matrix approach to inference.

KEY WORDS: Conditionally specified gaussian, Jacobian, GIS, vectorized profile likelihoods, spatial statistics, sparse matrices.

1. INTRODUCTION

The advent of the global positioning system (GPS), which can measure precisely geographic locations, coupled with the continuing development of geographic information systems (GIS) have led to an explosion of spatial data. For example, inexpensive phone books on CD-ROM now provide the latitude and longitude of over 100 million US residences with various demographic variables. The Bureau of the Census provides extensive demographic data for over 250,000 geographic points (census block groups). Remote sensing via satellites and digital imaging yields even vaster quantities of spatial information.

While the availability of spatial information has grown tremendously, the ability to process it has not grown *pari passu*. The necessary reliance on a n by n matrix, where n

[*]R. Kelley Pace is Louisiana Real Estate Commission Professor of Finance and Ronald P. Barry is Associate Professor of Statistics.

represents the number of observations, greatly impedes traditional spatial statistics from handling larger sample sizes. Spatial estimators rely upon either a n by n variance-covariance matrix (*e.g.*, kriging), the inverse of the variance-covariance matrix (*e.g.*, conditional autoregressions (CARs)), or the square root of the inverse variance-covariance matrix (*e.g.*, simultaneous autoregressions (SARs)). As n becomes large, the size of these matrices becomes astronomical. For example, a 50,000 observation spatial regression would require a 50 gigabyte variance-covariance matrix (double precision). Moreover, all of these estimators use computations such as equation solutions, inverses, or determinants which employ order of n cubed operations ($O(n^3)$) when operating on the full matrix.

Various ways of attacking this problem have been proposed. For example, Zimmerman (1989) devised a computational acceleration for the Gaussian covariance structure over a regular parallelogram lattice. Sone and Griffith (1995) discussed trade-offs from approximating the difficult normalizing constant (determinant) in maximum likelihood estimation of spatial statistical models. Li (1995) used a supercomputer to accelerate computations of the determinant term.

Ideally, one would like to quickly compute maximum likelihood spatial statistics for any sample size. To achieve this end, we assume that the direct effect of nearby observations decays to 0 after some distance.[*] This leads to a sparse covariance or inverse covariance matrix, which can potentially aid computations (*e.g.*, Ho and Klotz, 1992). In fact, Pace and Barry (1997) illustrated how to accelerate computations in SAR models via sparse matrices. As an illustration, they computed a simultaneous autoregression (SAR) on 20,640 observations using sparse matrices in under 19 minutes despite evaluating a determinant of 20,640 by 20,640 matrix ten times. Furthermore, Barry and Pace (1997) showed how to apply sparse matrix methods to the kriging problem via a mining data set. In addition, they performed a number of timing experiments showing the potential computational gains for plausible values of the spherical variogram.

In the realm of lattice models, many prefer conditional autoregressions (Besag 1974, 1975), Cressie 1993, p. 407-410). Accordingly, this paper provides a number of computational accelerations applicable to conditional autoregressions (CARs).

[*] This assumption has been employed both in a geostatistical context as represented by finite range variograms (see Barry and Ver Hoef (1997) for more information) and in a lattice context as represented by a limited number of neighbors. Even though the direct effect of a far away observations may be zero, each observation may yet have an indirect effect upon all other observations due to the dependency of each observation upon nearby observations.

The sparse matrix technology greatly accelerates computation of the normalizing constant, the major bottleneck in past algorithms. The acceleration of the difficult normalizing constant highlights other bottlenecks in the computation of CARs. We address these other bottlenecks by modifying the CAR computations. First, since many evaluations of the determinant term occur in the computation of each estimate, we reuse the same determinants by evaluating a set of these prior to estimation. Second, we greatly simplify the sum-of-squared errors (SSE) expression within the likelihood. This substantially reduces the number of computations needed. Third, we vectorize the computations. Many computer architectures favor the use of vectorized code. In addition, many languages such as Fortran 90, Gauss, Matlab, and S-Plus run faster with vectorized code.[†]

Moreover, to avoid problems of multiple local optima reported by Warnes and Ripley (1987), Ripley (1988), and Mardia and Watkins (1989), we evaluate the entire profile likelihood for the spatial differencing parameter. This ensures robustness and aids inference. In fact, we show computing profile likelihoods and simulating the estimator provides a low-cost means of inference superior to the traditional information matrix approach. Finally, we demonstrate an interesting invariance of CARs to a change in the magnitude, but not the direction, of the true errors.

As an illustration of the efficacy of these techniques, we perform a Monte Carlo simulation of the CAR estimator using 2,905 observations. Normally, even one spatial autoregression of this size could prove challenging. For example, Li (1995, p. 130) took 8566.5 seconds on an IBM RS6000 to compute a single 2,500 observation SAR. In contrast, we can compute 250 CARs on 2905 observations in 135.78 seconds (43.28 for 100 determinants and 92.5 for the other computations) using a 133 megahertz Pentium.

Section 2 discusses various facets of CAR computations, section 3 details the Monte Carlo study, and section 4 concludes with the key results.

2. CAR COMPUTATIONS

Section A begins by discussing the CAR model, section B provides details on the construction of the crucial spatial weighting matrix, section C discusses the SSE used in maximum likelihood, section D demonstrates the invariance of the estimate $\widetilde{\phi}_{ml}$ to the magnitude of the errors (but not their direction), section E discusses vectorized ways of computing the sum-of-squares, section F details the advantages of precomputing

[†] These interpreters clearly prefer the use of vectorized code. However, many compilers, such as Fortran 90, prefer such code because it allows the use of many optimizations.

determinants, section G presents the profile likelihood for an entire simulation, and section H gives ways of conducting inference without computing n by n information matrices.

A. THE SPATIAL CONDITIONAL AUTOREGRESSIVE ERRORS MODEL

Suppose the errors arise out of the following spatial conditional autoregressive error process,

$$Y = X\beta + (I - \phi C)^{-\frac{1}{2}}\varepsilon \tag{1}$$

where C represents a symmetric n by n weighting matrix with 0s on the diagonal and non-negative off-diagonals, the vector ε of length n denotes a $N(0, \sigma^2 I)$ error term, the n by k matrix X contains the independent variables, and the vector Y of length n contains the dependent variable. Since most interest centers on positive spatial autocorrelation, we use the restriction $\phi \geq 0$. We will normalize the elements in C to yield a maximum eigenvalue of 1. Coupled with the previous restriction, this normalization will ensure the positive definiteness of C as long as $\phi < 1$. A positive entry in the jth column of the ith row of C indicates that the jth observation directly affects the ith observation ($i \neq j$).

B. CONSTRUCTION OF THE SPATIAL WEIGHT MATRIX

As an illustration of how to construct C, compare the distance d_{ij} between every pair of observations j and i to $d_{max\,i}$, the distance from observation i and its mth nearest neighbor. Assign a weight of 1 to all observations in the non-normalized matrix $C^{(u)}$ whenever d_{ij} is less than or equal to $d_{max\,i}$ as in (2),

$$0 \leq d_{ij} \leq d_{max\,i} \leftrightarrow C_{ij}^{(u)} = 1\,. \tag{2}$$

Naturally, this yields a weight of 1 for the observation itself ($d_{ij} = d_{ii} = 0$) and 0 for each observation j more than $d_{max\,i}$ distance from observation i. To prevent the observation from predicting itself, we set $C_{ii}^{(u)} = 0$. The combination of these two procedures allows for observations j colocated with observation i ($i \neq j$) to have a weight of 1.

It seems reasonable to set to 0 the direct influence of distant observations upon a particular observation. This assumption can lead to rather sparse forms for C which will greatly accelerate the computation of the maximum likelihood estimator, as explored in 3.C.

As it stands, $C^{(u)}$ is neither symmetrical nor normalized. Let S represent an n by n diagonal matrix containing the square roots of the reciprocals of the sum of the columns and rows of $C^{(u)}$.

$$diag(S) = \left[\left(C^{(u)} + C^{(u)\prime}\right)[1]\right]^{-\frac{1}{2}} \qquad (3)$$

To normalize and make C symmetrical, we pre and post-multiply by S.

$$C = S\left(C^{(u)} + C^{(u)\prime}\right)S \qquad (4)$$

This operation will ensure the largest eigenvalue of C equals 1. See Ord (1975) for more details.

C. THE MAXIMUM LIKELIHOOD SUM-OF-SQUARED ERRORS

The conventional approach to computing CAR estimates relies upon solving the CAR normal equations (5). However, the solution to the equations depends upon the unobservable parameter ϕ.

$$\left[X'(I - \phi C)X\right]\beta = X'(I - \phi C)Y \qquad (5)$$

Due to its favorable statistical properties, maximum likelihood (ML) is the preferred way to determine the optimal value of ϕ. Substituting the solution of the normal equations into the likelihood function converts it into the univariate profile likelihood of the unknown parameter, ϕ. The optimal value of ϕ determines the ML estimate, $\tilde{\phi}_{ml}$.

Following the conventional approach becomes somewhat tedious when computing the estimate for different realizations of the dependent variable, Y, or when one desires the entire likelihood as a function of the unknown parameter, ϕ, also known as the profile likelihood. In either situation, one may need to solve equation (5) many thousands of times.

To increase the computational performance of the CAR estimator, reparameterize it to orthogonalize the independent variables as in (6). Hence, let Z represent a set of orthogonal independent variables.

$$Z\theta_{car} = X\beta \qquad Z'Z = I \qquad (6)$$

One can write the CAR normal equations in (5) as,

$$\left[Z'(I - \phi C)Z\right]\theta_{car} = \left[(I - \phi Z'CZ)\right]\theta_{car} = Z'(I - \phi C)Y\,. \qquad (7)$$

Substitute the spectral decomposition of $Z'CZ = U\Lambda U'$ in (7), where U represents a k by k matrix of eigenvectors and Λ represents the k by k diagonal matrix of eigenvalues. Since the eigenvectors are orthogonal, $I = UU'$. Neither the spectral decomposition of $Z'CZ$ or the orthogonal reparameterization of $Z'Z$ cost much computationally, since they work with a k (as opposed to n) dimensional problem. Substituting both of these relations yields,

$$\left[(UU' - \phi U\Lambda U')\right]\theta_{car} = \left[U(I-\phi\Lambda)U'\right]\theta_{car} = Z'(I-\phi C)Y \,. \tag{8}$$

Solving for θ_{car} yields,

$$\hat{\theta}_{car} = U(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)Y \tag{9}$$

Hence, the raw residuals from the above fit become,

$$r_{car} = Y - Z\hat{\theta}_{car} = \left(I - ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y \tag{10}$$

One can form the SSE for ML in (11) as in Cressie (1993, p. 437),

$$SSE_{ml} = r'_{car}(I-\phi C)r_{car} \tag{11}$$

Substituting (10) into (11) yields,

$$SSE_{ml} = Y'\left(I-(I-\phi C)ZU(I-\phi\Lambda)^{-1}U'Z'\right)(I-\phi C)\left(I-ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y \tag{12}$$

Expanding (12) gives,

$$\begin{aligned} SSE_{ml} = {} & Y'Y - \phi Y'CY - 2Y'(I-\phi C)\left(ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y + \\ & Y'\left((I-\phi C)ZU(I-\phi\Lambda)^{-1}U'Z'\right)(I-\phi C)\left(ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y \end{aligned} \tag{13}$$

Reorganizing (13) highlights the term in brackets within (14),

$$\begin{aligned} SSE_{ml} = {} & Y'Y - \phi Y'CY - 2Y'(I-\phi C)\left(ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y + \\ & Y'\left((I-\phi C)ZU(I-\phi\Lambda)^{-1}U'\right)\left[Z'Z-\phi Z'CZ\right]\left(U(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right)Y \end{aligned} \tag{14}$$

Recall $\left[Z'Z - \phi Z'CZ\right] = U(I-\phi\Lambda)U'$. Because of the orthogonality of U, a number of terms cancel, thus simplifying (14) greatly.

$$SSE_{ml} = Y'\left[(I-\phi C) - (I-\phi C)ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)\right]Y \tag{15}$$

D. INVARIANCE TO THE MAGNITUDE OF THE ERRORS

Take the expression for SSE_{ml} in (15) and factor $(I-\phi C)^{1/2}$ out of the main body of SSE_{ml}. Hence,

$$SSE_{ml} = Y'(I-\phi C)^{1/2}\left[I-(I-\phi C)^{1/2}ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)^{1/2}\right](I-\phi C)^{1/2}Y \qquad (16)$$

Assume, as CAR does, that,

$$Y = Z\theta_{car} + (I-\phi C)^{-1/2}\varepsilon \qquad (17)$$

Substituting (17) into the (16) yields,

$$SSE_{ml} = \left(\varepsilon+\theta'_{car}Z'(I-\phi C)^{1/2}\right)\left[I-(I-\phi C)^{1/2}ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)^{1/2}\right]\left((I-\phi C)^{1/2}Z\theta_{car}+\varepsilon\right) \qquad (18)$$

The terms involving θ_{car} vanish, thus leaving,

$$SSE_{ml} = \varepsilon'\left[I-(I-\phi C)^{1/2}ZU(I-\phi\Lambda)^{-1}U'Z'(I-\phi C)^{1/2}\right]\varepsilon \qquad (19)$$

Hence, scaling the errors by a positive constant will not affect where SSE attains its minimum or where the likelihood attains its maximum.

E. COMPUTATIONAL FORMS FOR THE SUM-OF-SQUARED ERRORS

Expanding the basic SSE equation in (15) yields,

$$SSE_{ml} = Y'Y-\phi Y'CY-Y'ZU\left[I-\phi\Lambda\right]^{-1}U'Z'Y-Y'CZU\phi^2\left[I-\phi\Lambda\right]^{-1}U'Z'CY+2Y'ZU\phi\left[I-\phi\Lambda\right]^{-1}U'Z'CY$$

One has the choice of using a vector of values of ϕ but using only one vector of *Y* each iteration or using a matrix of values of *Y* and looping over ϕ. In an estimation context with a particular *Y*, the first approach seems more natural.[‡] In a simulation context with multiple realizations of *Y*, the second approach seems more natural and thus we follow this approach below.

For use with multiple realizations of *Y* as in simulations redefine Y_r as $[Y_1, Y_2,\ldots Y_{iter}]$, an *n* by *iter* matrix. Define,

$$\underset{n\times k}{a_1 = ZU}, \underset{n\times iter}{a_2 = CY_r}, \underset{k\times iter}{a_3 = a_1'Y_r}, \underset{k\times iter}{a_4 = a_1'a_2}, \underset{k\times 1}{a_5 = diag\left(\left[I-\phi_i\Lambda\right]^{-1}\right)}$$
$$\underset{1\times iter}{a_6 = [1]'(Y_r\cdot Y_r)}, \underset{1\times iter}{a_7 = [1]'(Y_r\cdot a_2)}, \underset{k\times iter}{a_{33} = a_3\cdot a_3}, \underset{k\times iter}{a_{44} = a_4\cdot a_4}, \underset{k\times iter}{a_{34} = 2a_3\cdot a_4}$$

[‡] The algorithm using a vector of values of ϕ works well, but has a somewhat more complicated form owing to the number of places ϕ appears in the SSE_{ml}.

where the symbol $\cdot$ denotes the use of Hadamard element by element multiplication.

$$\underset{1\times iter}{SSE_{ml}(\phi_i, Y_r)} = \underset{1\times iter}{a_6} - \phi_i \underset{1\times iter}{a_7} - \underset{1\times k}{(a_5')}\underset{k\times iter}{(a_{33} - \phi_i a_{34} + \phi_i^2 a_{44})}$$

Let ϕ_i take on values $[\phi_1 \quad \phi_2 \quad \phi_3 \quad \ldots \quad \phi_l]$. Loop over $\underset{1\times iter}{SSE_{ml}(\phi_i, Y_r)}$ l iterations of ϕ to define $\underset{l\times iter}{SSE_{ml}(\phi, Y_r)}$.

F. PRECOMPUTING LOG-DETERMINANTS

The log-determinant, $\ln|I-\phi C|$, plays an important part in the likelihood function. If ϕ takes on only the values $[\phi_1 \quad \phi_2 \quad \phi_3 \quad \ldots \quad \phi_l]$, we can precompute an l by 1 vector of log-determinants. Several advantages accrue to computing these as a group. First, the ordering algorithms for sparse matrices apply to all matrices with a given pattern of sparsity. This pattern of zeros and non-zeros for $\ln|I-\phi C|$ would apply for all positive ϕ. Hence, computing the determinants as a group amortizes over l iterations the cost of computing the optimal ordering. Second, as this could prove a time consuming step for some problems, computing them as a group allows the computations to occur during non-peak usage. Third, most fitting exercises involve exploratory or diagnostic variations in the model which would affect the specification of *X* and hence *Z* but not affect $\ln|I-\phi C|$. Thus, changes in the specification of *X* further amortize the cost of precomputing $\ln|I-\phi C|$.

G. COMPUTING THE CAR LIKELIHOOD

Given the various computations, we can define twice the profile log-likelihood over the entire simulated Y_r.

$$2\begin{bmatrix} L(\phi_1|Y_1) & L(\phi_1|Y_2) & \ldots & L(\phi_1|Y_{iter}) \\ L(\phi_2|Y_1) & L(\phi_2|Y_2) & \ldots & L(\phi_2|Y_{iter}) \\ \vdots & \vdots & \ddots & \vdots \\ L(\phi_l|Y_1) & L(\phi_l|Y_2) & \ldots & L(\phi_l|Y_{iter}) \end{bmatrix} \propto \begin{bmatrix} \ln|I-\phi_1 C| \\ \ln|I-\phi_2 C| \\ \vdots \\ \ln|I-\phi_l C| \end{bmatrix} \underset{1\times iter}{[1]} - n\ln \begin{bmatrix} SSE(\phi_1, Y_1) & \ldots & SSE(\phi_1, Y_{iter}) \\ SSE(\phi_2, Y_1) & \ldots & SSE(\phi_2, Y_{iter}) \\ \vdots & \ddots & \vdots \\ SSE(\phi_l, Y_1) & \ldots & SSE(\phi_l, Y_{iter}) \end{bmatrix}$$

For each column of twice the profile-likelihood, find the value of ϕ which maximizes each column, ϕ^*. Hence, ϕ^* represents the maximum likelihood estimate, $\widetilde{\phi}_{ml}$, for the dependent variable associated with that column. Each row provides information for inference on $\widetilde{\phi}_{ml}$ across different realizations *Y*. Hence, the matrix of profile likelihoods across ϕ and *Y* provides a wealth of information.

H. INFERENCE

Computation of the observed or expected information matrix becomes expensive in a spatial context. Inspection of the information matrix in Cressie (1993, p. 484) shows it requires, among other computations, an n by n inverse ($O(n^3)$) and multiplication of n by n matrices ($O(n^3)$). The inversion of the n by n matrix defeats CAR's computational advantage of modeling the inverse of the variance-covariance matrix as opposed to kriging's approach of modeling the variance-covariance matrix directly.[§]

In addition, the information matrix approach works best for a profile likelihood quadratic in ϕ. However, most plots of this type of profile likelihood display substantial asymmetry (*e.g.*, see Ripley 1988, p. 14) which seems natural given the limited range of ϕ. In such cases, as Meeker and Escobar (1995) as well as others forcefully argue, profile likelihood techniques can outperform the information matrix approach. Finally, the information matrix approach requires enough "smoothness" to make second derivatives well-behaved. As Ripley (1988, p. 11-15) documented, such behavior does not occur universally.

If interest centers only upon inference concerning ϕ, the profile likelihood contains a wealth of information. With this one can test hypotheses and construct confidence intervals.

If interest centers upon both ϕ and β, one could proceed via two routes. First, one could use restricted least squares to conduct likelihood ratio tests. Because the problem has been reparameterized in terms of θ, this complicates the construction of hypotheses. Alternatively, one could simulate the problem. Simulation of the dependent variable values as in (20) below and finding the maximum likelihood point estimates $\tilde{\theta}_{ml}$, $\tilde{\phi}_{ml}$ requires remarkably little time. Given a set of $\tilde{\theta}_{ml}$ from a simulation, converting these back into $\tilde{\beta}_{ml}$ also takes little time. The simulated set of $\tilde{\phi}_{ml}$, $\tilde{\beta}_{ml}$ potentially provides better confidence intervals than the traditional quadratic approximation based upon the observed or expected information matrix. A simulation should provide better information as to the small sample properties of the problem.

[§] While sparsity could greatly help in computing the observed or expected information matrix, the fill-in produced by inversion runs counter to the spirit of attempting to maximize sparsity.

As an added bonus, one could use this procedure to very easily perform Bayesian inference. For example, inequality restricted Bayesian estimation arises naturally out of simulation.[**] Section 3.F illustrates the use of simulation for inference concerning ϕ.

3. SIMULATED CARS

This section examines a simulation of 2,50 regressions each using 2,905 observations. Section A discusses the data, section B defines the role of sparsity, section C contains the timings of the simulation computations, section D discusses the statistical results from the simulation, while section E illustrates the use of simulation for inference.

A. MONTE CARLO DATA

To provide verisimilitude to the simulation, we chose an actual set of locations for use in forming *C*, the spatial weighting matrix. Specifically, we used the geographic centroids from all the census block groups in Connecticut from the 1990 Census. This yielded a matrix *C* with 2,905 rows and 2,905 columns.

In the simulation, we:

1. Generated uniform random variables for nine columns of *X* and used a constant for the other column.
2. Set β to a vector of ones.
3. Let ϕ equal [.1, .25, .5, .75, .9].
4. Generated a common set of 250 N(0,1) vectors of 2,905 elements each using the Matlab normal random number generator. We perform this operation once for the entire simulation. Scaling the common N(0,1) errors by σ generates the N(0, σ^2) random variables. This practice, referred to as "correlated sampling" (Rubinstein, 1981) greatly reduces the variance in Monte Carlo experiments.

We subsequently generated the autocorrelated dependent variable, *Y,* according to (20),

$$Y = X\beta + \sigma(I - \phi C)^{-\frac{1}{2}}u \tag{20}$$

where *u* represents an *n* by *iter* matrix of N(0,1) random variates. In actuality, we solved the corresponding equation system in (21) for the coefficients A_S rather than computing the inverse as this goes much faster.

[**] See Gilley and Pace (1995) for a Monte Carlo study of the inequality restricted Bayesian estimator. See Pace and Gilley (1993) for more information on inequality restrictions in regression settings.

$$\underset{n\times n}{(I-\phi C)^{\frac{1}{2}}}\ \underset{n\times iter}{A_S} = \underset{n\times iter}{u}$$
$$\underset{n\times iter}{A_S} = \underset{n\times n}{(I-\phi C)^{-\frac{1}{2}}}\ \underset{n\times iter}{u} \tag{21}$$

Compare this to the usual Cholesky decomposition and inverse formulation,

$$\underset{n\times n}{(I-\phi C)^{\frac{1}{2}}}\ \underset{n\times n}{A_I} = \underset{n\times n}{I}$$
$$\underset{n\times n}{A_I} = \underset{n\times n}{(I-\phi C)^{-\frac{1}{2}}} \tag{22}$$
$$\underset{n\times iter}{A_S} = \underset{n\times n}{A_I}\ \underset{n\times iter}{u} = \underset{n\times n}{(I-\phi C)^{-\frac{1}{2}}}\ \underset{n\times iter}{u}$$

Relative to (21), (22) requires solving a larger system (*n* by *n* instead of *n* by *iter* and subsequently multiplying an *n* by *n* matrix by a *n* by *iter* matrix ($O(n^2 iter)$). Since *iter* usually is much smaller than *n*, the inverse method takes substantially longer for the same results.

Since efficient computation of the determinant required by maximum likelihood estimation uses the Cholesky decomposition of $(I-\phi C)$, this reduces the additional cost of simulating the random numbers. However, solving the 2,905 equations by 2,905 unknowns for *iter* right hand sides would prove quite difficult without resorting to sparse matrix techniques as described in 3.C below.

B. SPARSITY IN SPATIAL PROBLEMS

If differencing an observation with its nearby neighbors removes most of the effects of autocorrelation, the spatial weighting matrix *C* can be quite sparse. For example, if an observation displays error dependency only with its nearest *m* neighbors, only *m* non-zero entries exist per row of *C*. Thus, *C* will contain *nm* non-zero elements out of n^2 possible ones. This produces a *m/n* proportion of non-zero elements, a popular measure of sparsity. For example, with this problem we used four neighbors for each observation in computing $C^{(u)}$. This resulted in *C* having on average 5.08 neighbors and a sparsity of 5.08/2905 (0.21%). This represents a very high level of sparsity which grows as *n* increases.

Sparsity results in a number of computational gains. First, it dramatically decreases the storage needed for *C* and $(I-\phi C)$. Using traditional dense techniques, *C* requires 67.5 MB of storage (double precision). Using sparse matrix techniques, *C* requires less than 200 KB of storage. Naturally, this divergence grows with *n*.

Second, sparsity greatly accelerates computations. For example, multiplying the *n* by *n* matrix *C* with the *n* by *k* matrix *X* requires $O(kn^2)$ operations using dense matrices. Barring computational bookkeeping, the equivalent sparse operation requires $O(knm)$ operations, a much smaller number. The real benefits come when computing determinants, inverses, or solving systems of equations. All of these operations can build upon the Cholesky decomposition of a matrix and all require O(n^3) operations when using dense matrices. Sparsity, however, can totally change the order of the number of operations required in these computations. For example, if $(I-\phi C)$ had a band structure with half-bandwidth p, the Cholesky decomposition of $(I-\phi C)$, would require O($n(p^2+3p)$) operations (Golub and Van Loan (1989, p. 154)). Hence, for fixed bandwidths the computations grow linearly with n, the number of observations.

Unfortunately, the existence of a pure band structure does not arise very often. Figure 1a shows the actual plot of the non-zero elements in $(I-\phi C)$. The existence of such dispersed off-diagonals could make it difficult to achieve computational gains.

However, one can permute the rows or columns of $(I-\phi C)$ to reduce bandwidth or to achieve other optimizations. A variety of such permutations exist (see George and Liu (1981) for more on the various orderings). For example, the reverse Cuthill-McKee algorithm attempts to reorder the rows and columns of the matrix to create a variable band matrix as shown in Figure 1b. Figure 1b makes the gains of exploiting sparsity obvious. Less obviously, Figure 1c shows the plot of $(I-\phi C)$ permuted using the column minimum degree algorithm while Figure 1d shows the plot of $(I-\phi C)$ permuted using the symmetric minimum degree algorithm.[††]

Table I shows the timings associated with computing the log-determinants using the original, random, reverse Cuthill-McKee, column minimum degree orderings, and symmetric column minimum degree orderings for $(I-\phi C)$. As Table I makes clear, the ordering of the rows and columns matters, with the symmetric minimum degree ordering reducing execution times by 86% over the original ordering. Intriguingly, the more intuitive reverse Cuthill-McKee ordering actually performed worse than the original ordering but much better than the random ordering. As a worst case scenario, the random ordering produced computational times worse than the optimal ordering by a factor of 572. All computations used the Matlab language running on a 133Mhz Pentium computer.

To place these results in perspective, Li (1995) took the eigenvalue route to computing determinants.[‡‡] Li used an IBM RS6000 Model 550 and a CM5 parallel processing

[††] The use of column minimum degree ordering destroys symmetry. Hence, the determinant computations use LU decomposition.

[‡‡] The sparse matrix technology makes it much faster to compute the necessary determinants than to find the determinant via eigenvalues, the standard practice in this area (Ord (1975)). While sparse eigenvalue routines exist, these still require more time usually than the direct computation of a set of determinants.

supercomputer. The CM5 had 32 processors each with 32MB of local memory and four vector units. For a 2500 by 2500 spatial weight matrix the RS6000 required 8515.07 seconds while the CM5 required 45.78 seconds. In contrast, it took only 43.28 seconds to compute 100 different determinants of a 2,905 by 2,905 matrix on a 133Mhz Pentium computer using Matlab. Hence, the sparse technology employed here allowed a personal computer on a larger problem to exceed the performance of a supercomputer on a smaller problem!

Finally, even supercomputers have memory limitations. The use of sparse matrix technology has allowed us to handle problems with 20,640 observations (Pace and Barry (forthcoming)). A dense spatial weight matrix would have required 3.58 gigabytes (double precision), which would have further exacerbated the overall computational difficulties.

C. MONTE CARLO EXPERIMENT TIMINGS

Simulating a CAR need not take very long. To show this, we measured times in the different computational stages for simulating 250 regressions for a particular case ($\phi=.5$). It took 4.07 seconds from the time the matrix C and the log-determinants were loaded to generate X and other matrices needed and to perform the decompositions. It took 17.52 additional seconds to generate the 250 spatially autocorrelated realizations of Y. This involved computing a Cholesky decomposition of $(I-\phi C)$ and solving a 2,905 by 2,905 system 250 times. Sparsity provides an incredible boost to these computations. The computation of the 250 maximum likelihood estimates took only 70.91 seconds longer. Hence, the total time for computing 250 was 92.5 seconds (conditional upon the precomputation of the determinants and C).

D. MONTE CARLO EXPERIMENT RESULTS

The simulation results in Table II match some of those reported in the literature using regular lattices.[§§] First, the maximum likelihood estimator slightly underestimates the true differencing parameter, ϕ. However, both estimators show very small amounts of bias, especially when examining the median values of $\widetilde{\phi}_{ml}, \widetilde{\phi}_{egls}$. Second, the variance of both estimators decreases with rising ϕ. The inequality restrictions do not seem to affect the results much except for when ϕ is .1. In this case, the inequality aids both estimators.

[§§] See Haining (1990, p. 135-137) and Cressie (1993, p. 477) for a review of studies finding these effects.

E. PROFILE LIKELIHOOD PLOTS

As an example of the use of simulation for inference, Figures 2a, 2b, 2c, and 2d show the profile likelihood plots for $\tilde{\phi}_{ml}$ across 25 iterations for ϕ = .05, .25, .5, .95. The asterisk on each profile likelihood curve denotes the maximum point. The vertical line segment from the lowest to the highest curve shows the location of the true value of ϕ. The large dot denotes the mean while the symbol $\times$ denotes the median for all the profile likelihoods. Figures 2a and 2d (ϕ = .05, .95) illustrate how the information matrix approach could run into problems. Note, in Figure 2c (ϕ = .5) the profile likelihood is asymmetric. Figure 2b shows an intermediary case (ϕ = .25).

The simulated profile likelihoods distill great amounts of information about the maxima of the likelihoods and their global behavior across possible realizations (simulation trials). One can immediately obtain a feel for the robustness of inference for the important spatial differencing parameter, ϕ.

4. CONCLUSION

In a spatial context, maximum likelihood techniques have a reputation of being slow to compute, prone to error through local optima, and offering only asymptotic inference. The approach adopted here to computing conditional autoregressions attacks each of these problems. First, the use of sparse matrices and restructuring the sum-of-squared errors function greatly accelerates computations. Second, the evaluation of the likelihood over a grid of values of the important spatial differencing parameter greatly enhances robustness to local optima. Third, the quick simulation of the conditional autoregression coupled with the profile likelihood information provides a wealth of information for conducting inference in all sample sizes.

As an illustration of the efficacy of these techniques, we computed (a) 250 simulated conditionally autoregressive vectors of Y of 2905 observations; (b) 100 determinants of a 2,905 by 2,905 matrix; and (c) computed 250 estimates with the profile likelihood values. This took 135.8 seconds (43.3 seconds for the determinants and 92.5 for the estimates). In contrast, Li (1995) on a smaller problem took longer on a supercomputer to calculate the determinants than we accomplished on a personal computer.

Hopefully, additions like these to the spatial statistics toolbox will enable users to help cope with the increasing flow of spatial information.

ACKNOWLEDGEMENTS

We both would like to thank the University of Alaska for its generous research support. In addition, Pace would like to acknowledge support from the Center for Real Estate and Urban Economic Studies, University of Connecticut. Finally, we would like the thank Bill Hughes for having discussed the paper at the American Real Estate Society meetings.

Bibliography

Barry, Ronald P., and R. Kelley Pace. (1997). "Kriging with Large Data Sets Using Sparse Matrix Techniques," *Communications in Statistics*: *Computation and Simulation,* **26**.

Barry, Ronald P., and Jay M. Ver Hoef. (1997). "Blackbox Kriging — Kriging without Specifying Variogram Models," *Journal of Agricultural, Biological, and Environmental Statistics*.

Besag, Julian E. (1974). "Spatial Interaction and the Statistical Analysis of Lattice Systems," *Journal of the Royal Statistical Society, B* **36**, 192-236.

Besag, Julian E. (1975). "Statistical Analysis of Non-lattice Data," *The Statistician*, **24**, 179-95.

Cressie, Noel A.C. (1993). *Statistics for Spatial Data,* Revised ed. New York: John Wiley.

George, Alan, and Joseph Liu. (1981). *Computer Solution of Large Sparse Positive Definite Systems*, Englewood Cliffs: Prentice-Hall.

Gilley, O.W., and R. Kelley Pace. (1995). "Using Inequality Restrictions to Tame Multicollinearity in Hedonic Pricing Models," *Review of Economics and Statistics,* **77**, 609-621.

Golub, G. H., and C. F. Van Loan. (1989). *Matrix Computations*, second edition, John Hopkins.

Haining, Robert. (1990). *Spatial Data Analysis in the Social and Environmental Sciences*, Cambridge University Press.

Ho, Shu-Yen, and Jerome Klotz. (1992). "Sparse Matrix Methods for Unbalanced Multifactor Analysis of Variance and Covariance," *Journal of Statistical Computation and Simulation*, **41**, 55-72.

Li, Bin. (1995). "Implementing Spatial Statistics on Parallel Computers," in: Arlinghaus, S., ed. *Practical Handbook of Spatial Statistics*, (CRC Press, Boca Raton), 107-148.

Mardia, K.V., and A.J. Watkins. (1989). "On Multimodality of the Likelihood in the Spatial Linear Model," *Biometrika*, **76**, 289-295.

Meeker, W.Q., and L.A. Escobar. (1995). "Teaching about Approximate Confidence Regions Based on Maximum Likelihood Estimates," *The American Statistician,* **49**, 48-53.

Ord, J.K. (1975). Estimation Methods for Models of Spatial Interaction, *Journal of the American Statistical Association*, **70**, 120-126.

Pace, R. Kelley, and Ronald Barry. (1997). "Sparse Spatial Autoregressions," *Statistics and Probability Letters*, **33**, 291-297.

Pace, R. Kelley, and O.W. Gilley. (1993). "Improving Prediction and Assessing Specification Quality in Non-Linear Statistical Valuation Models," *Journal of Business and Economics Statistics,* **11**, 301-310.

Ripley, Brian D. (1981). *Spatial Statistics,* New York: John Wiley.

Ripley, Brian D., (1988). *Statistical Inference for Spatial Processes,* Cambridge: Cambridge University Press.

Rubinstein, R.V. (1981). *Simulation and the Monte Carlo Method*, New York: Wiley.

Sone, Akio, and Daniel Griffith. (1995). "Trade-Offs Associated with Normalizing Constant Computational Simplifications for Estimating Spatial Statistical Models," *Journal of Statistical Computation and Simulation*, **51,** 165-184.

Warnes, J.J., and Brian Ripley. (1987). "Problems with Likelihood Estimation of Covariance Functions of Spatial Gaussian Processes," *Biometrika*, **74**, 640-642.

Zimmerman, Dale. (1989). "Computationally Exploitable Structure of Covariance Matrices and Generalized Covariance Matrices in Spatial Models," *Journal of Statistical Computation and Simulation*, **32**, 1-15.

Table I — Determinant Computation Times

Ordering	Time per 100 Determinants
Random	24,772.00
Original	310.33
Reverse Cuthill-McKee	338.78
Column Minimum Degree	65.03
Symmetric Minimum Degree	43.28

Table II — ML CAR Simulation Statistics

ϕ	$mean(\widetilde{\phi}_{ml})$	$median(\widetilde{\phi}_{ml})$	$\sigma(\widetilde{\phi}_{ml})$	$range(\widetilde{\phi}_{ml})$
0.1000	0.0971	0.1000	0.0515	0.2700
0.2500	0.2463	0.2500	0.0493	0.3000
0.5000	0.4969	0.4950	0.0394	0.2400
0.7500	0.7477	0.7450	0.0262	0.1450
0.9000	0.8981	0.8950	0.0151	0.0800

Figure 1a − Original Ordering

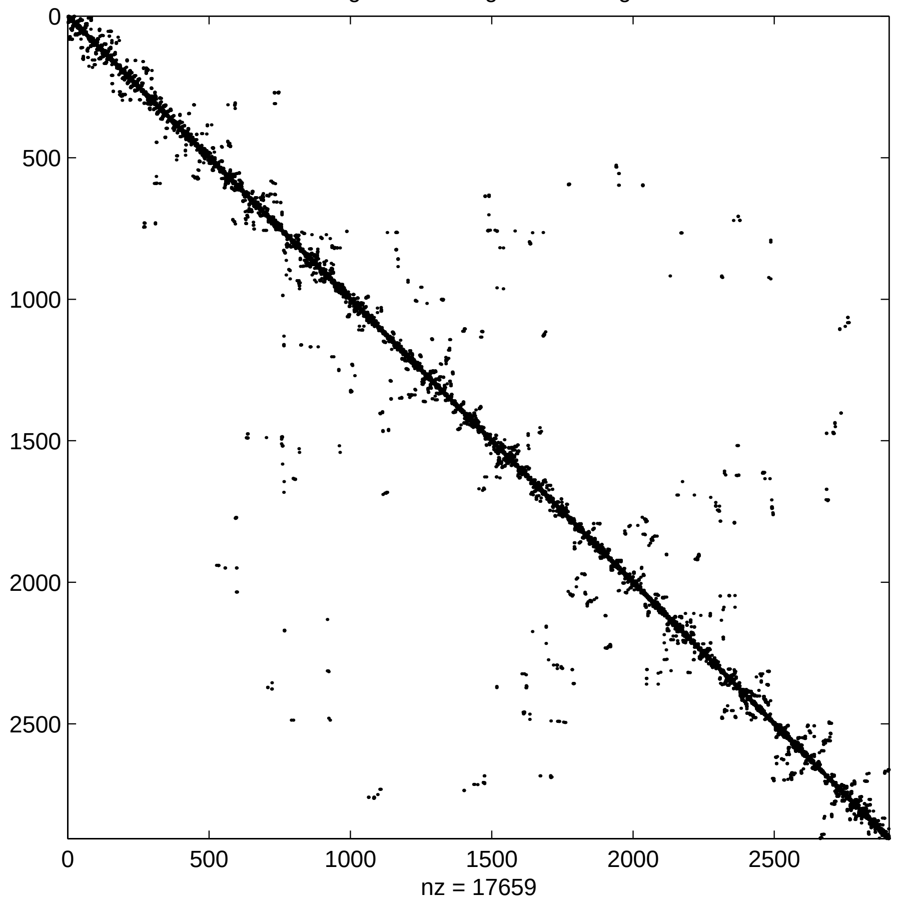


Figure 1b − Reverse Cuthill−McKee Ordering

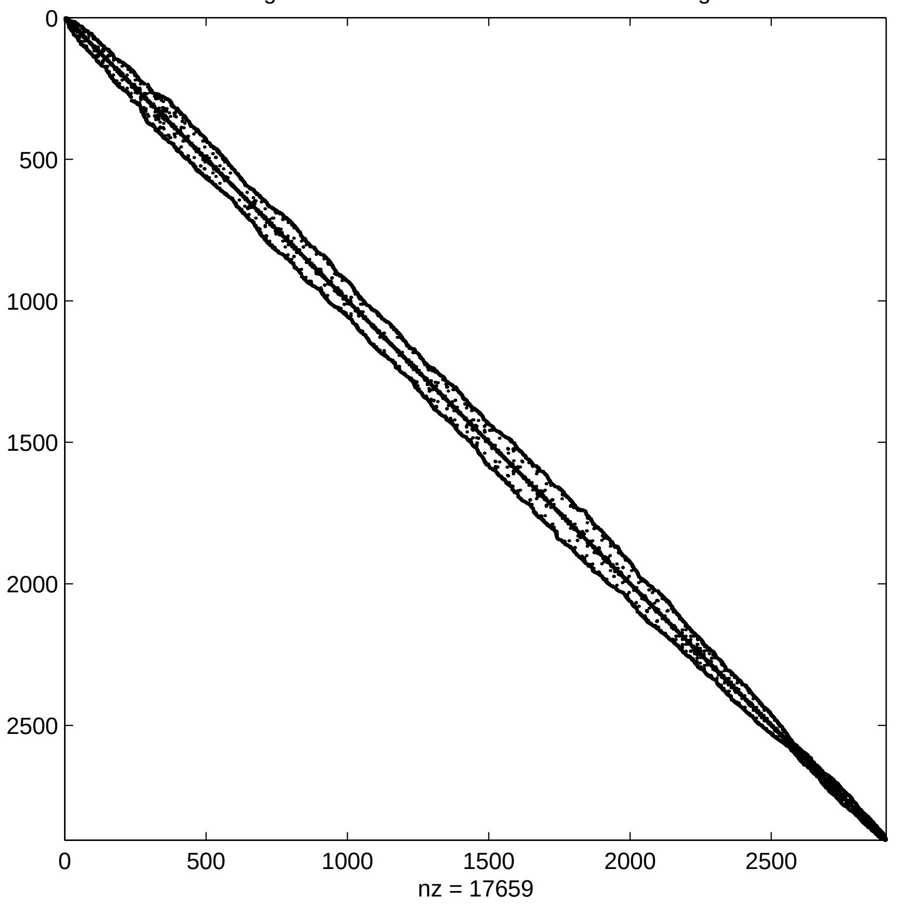


Figure 1c − Column Minimum Degree Ordering

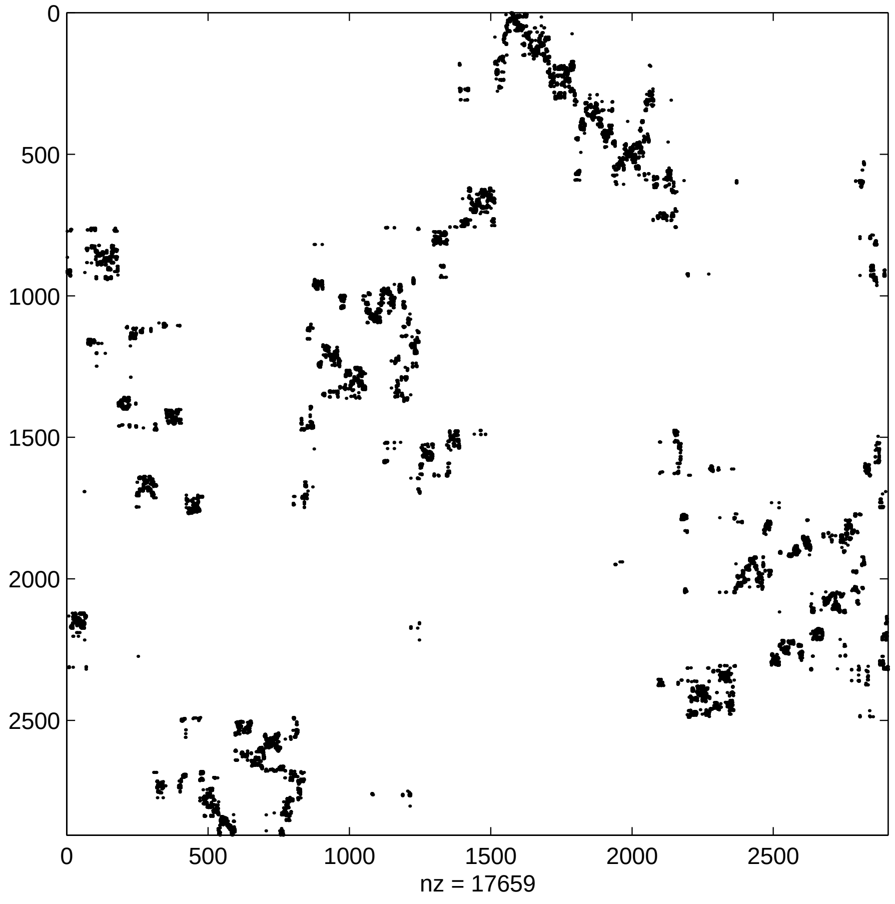


Figure 1d − Symmetric Minimum Degree Ordering

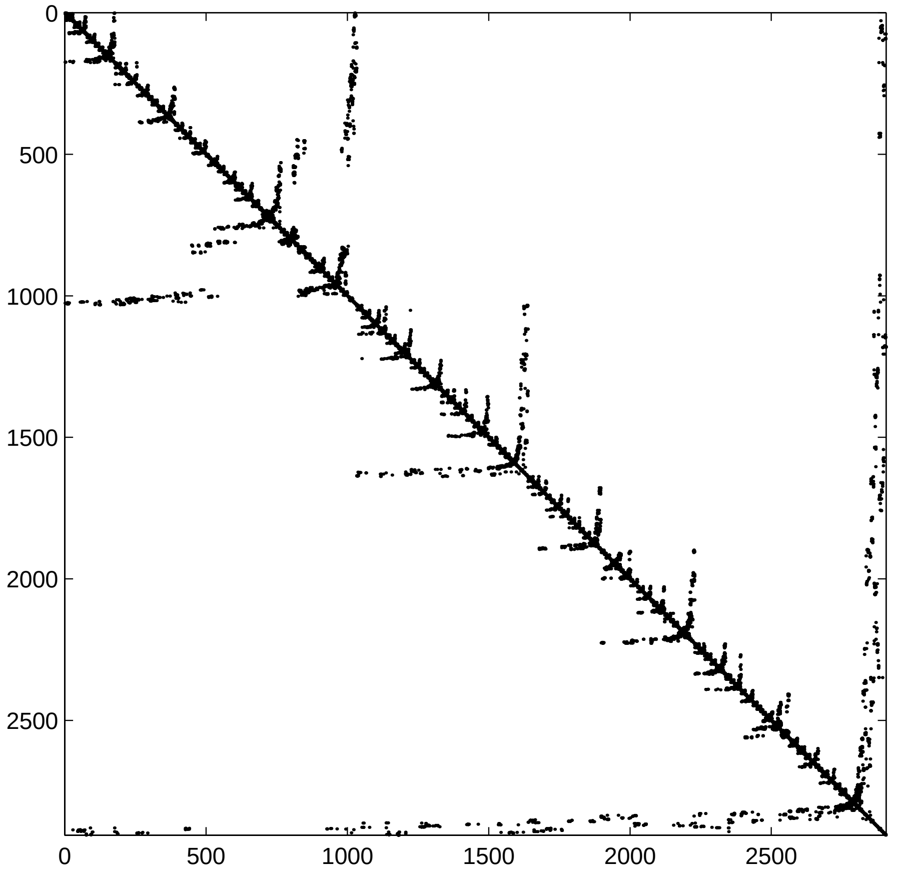


Figure 2a − Profile Likelihood for ϕ of 0.05

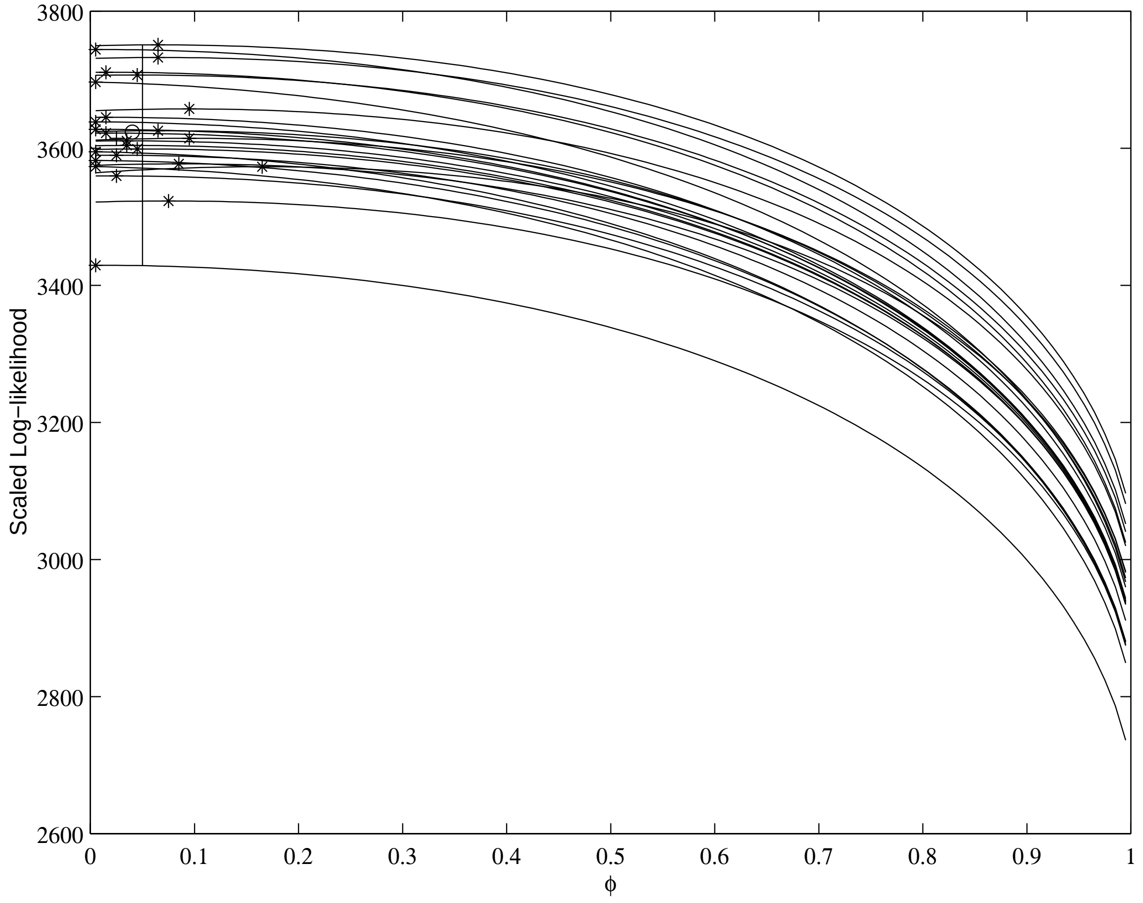


Figure 2b − Profile Likelihood for ϕ of 0.25

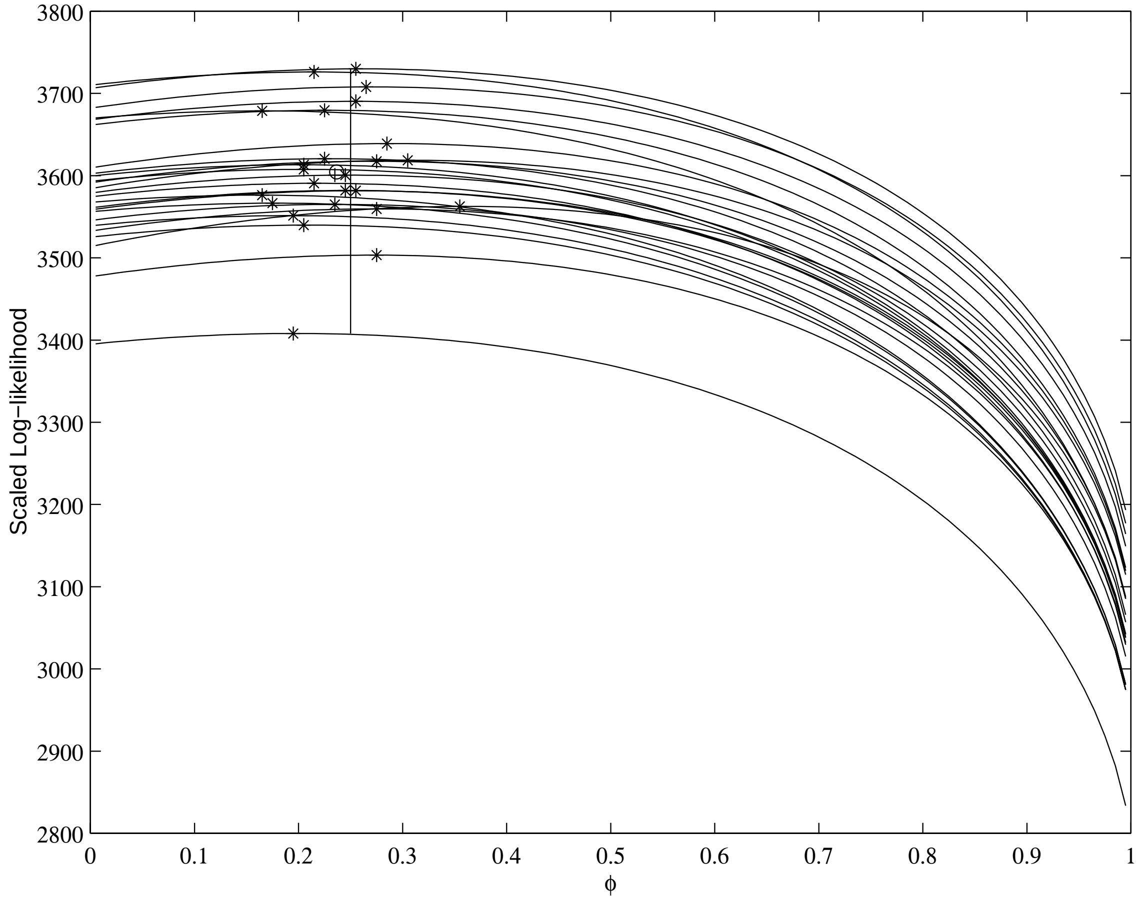


Figure 2c − Profile Likelihood for ϕ of 0.50

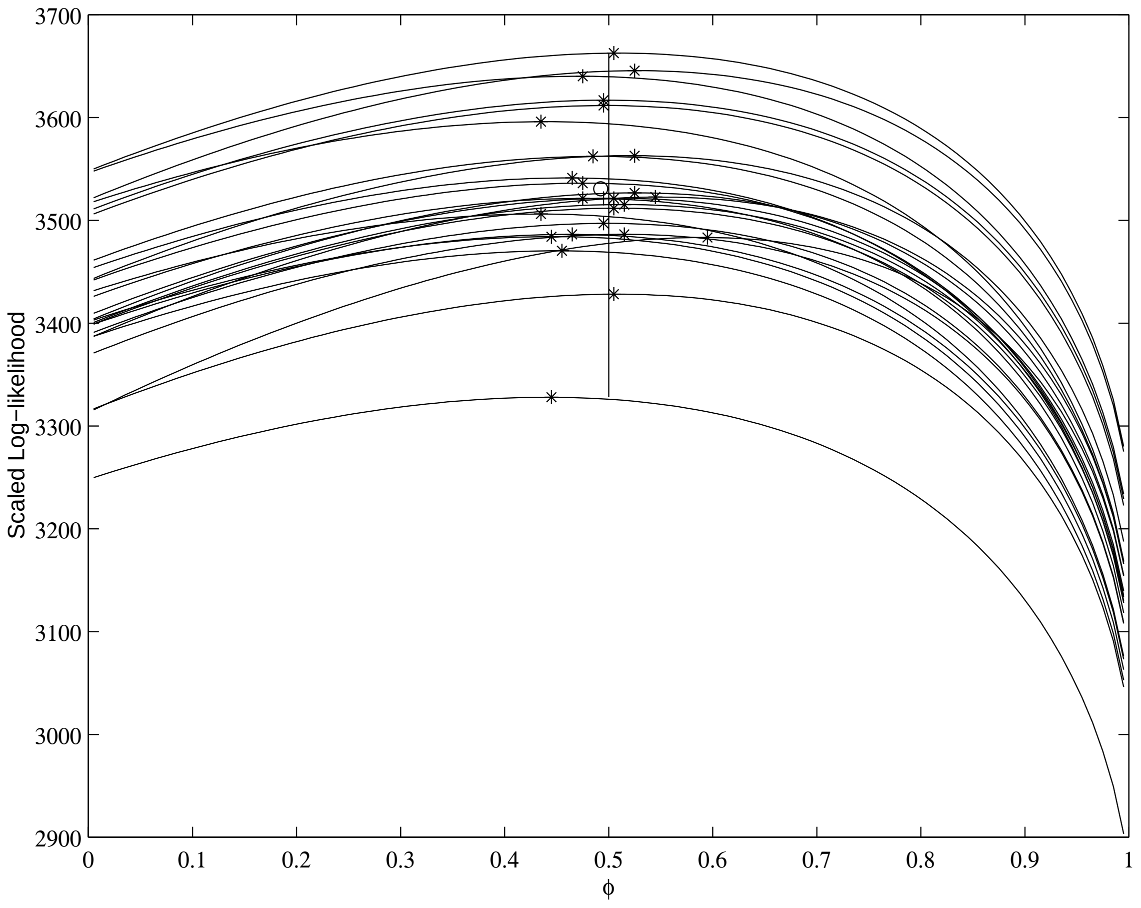
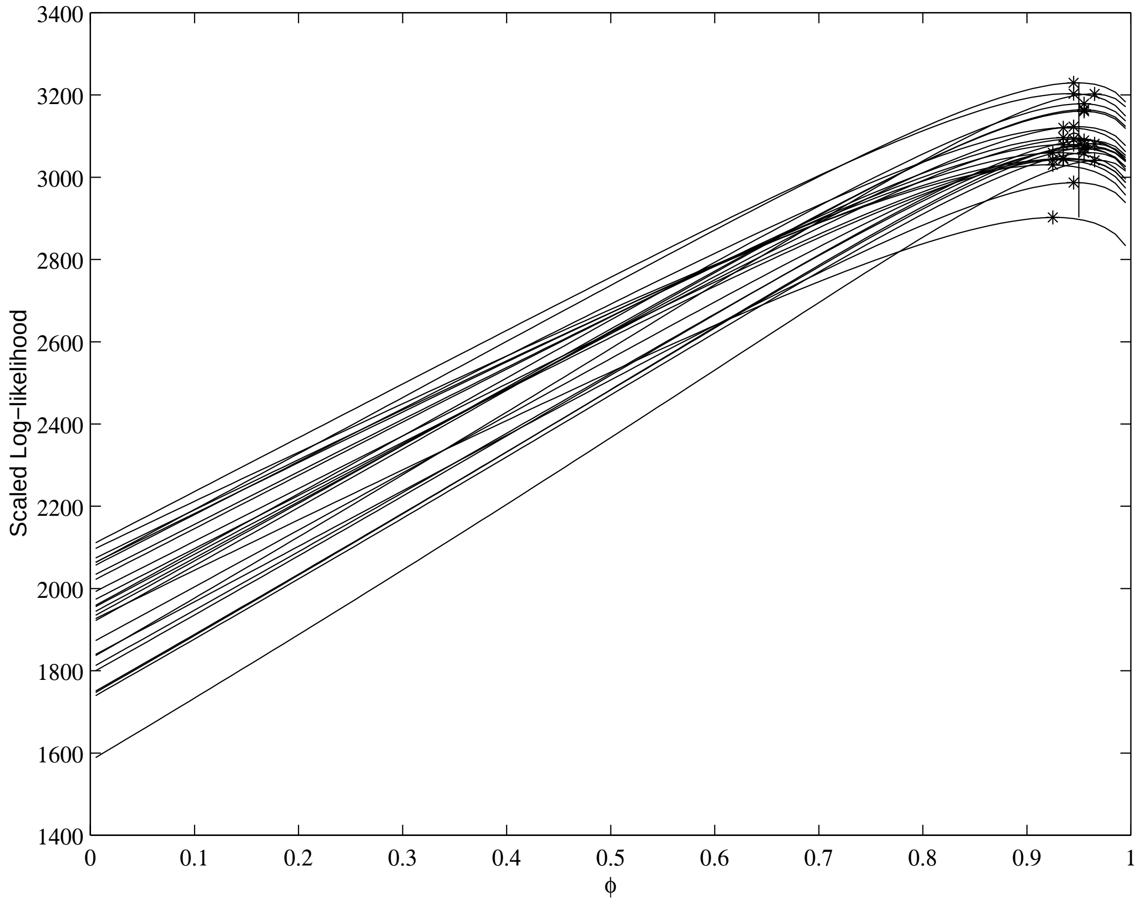


Figure 2d − Profile Likelihood for ϕ of 0.95


Scaled Log−likelihood
ϕ