ELSEVIER

Statistics & Probability Letters 45 (1999) 317–324

STATISTICS & PROBABILITY LETTERS

www.elsevier.nl/locate/stapro

A new algorithm for 5-band Toeplitz matrix inversion with application to GCV smoothing spline computation ☆

Jyh-Jen Horng Shiau[1]

Institute of Statistics, National Chiao Tung University, Hsinchu, Taiwan 300

Received October 1998; received in revised form March 1999

Abstract

A new algorithm is developed for computing any entry of the inverse of a 5-band Toeplitz matrix. After a linear-time overhead, each entry can be computed in constant time. As an application of this algorithm, we present a way to compute the generalized cross validated smoothing spline in linear time for the equally spaced data case.

MSC: 62G07; 65D07; 65D10; 65D15

Keywords: Band matrix; Toeplitz matrix; Efficient algorithms; Smoothing splines; Generalized cross validation; Divided difference

1. Introduction

A Toeplitz matrix is a matrix whose entries are constant along each diagonal. A band matrix is a matrix whose nonzero entries are all fairly near the main diagonal. These two types of matrices arise in many applications. The formal definitions (from Golub and Van Loan, 1989) of these two terms are given in the following.

Definition 1. The $n \times n$ matrix $A = (a_{ij})$ is Toeplitz if there exist scalars $r_{-n+1}, \ldots, r_0, \ldots, r_{n-1}$ such that $a_{ij} = r_{j-i}$ for all i and j.

Definition 2. The $n \times n$ matrix $A = (a_{ij})$ is a band matrix with upper bandwidth q if $a_{ij} = 0$ whenever $j > i + q$ and lower bandwidth p if $a_{ij} = 0$ whenever $i > j + p$.

☆ This work was partially supported by the National Science Council of Republic of China under Grant No. NSC87-2118-M-009-004.

[1] Presently visiting scholar, Department of Statistics, Stanford University, Stanford, CA 94305-4065.

E-mail address: jyhjen@stat.nctu.edu.tw (J.-J. Horng Shiau)

0167-7152/99/$ - see front matter
PII: S0167-7152(99)00073-5

Because of the special structures of these two types of matrices, substantial economics can be realized in many matrix computations, such as inversion or solving linear system $A\boldsymbol{x}=\boldsymbol{b}$, where $\boldsymbol{x}$ and $\boldsymbol{b}$ are $n\times 1$ vector. For more information, see Golub and Van Loan (1989) and Dongarra et al. (1979).

In this paper, we consider the problem of inverting a 5-band Toeplitz matrix, which is of the form

$$\begin{bmatrix} r_0 & r_1 & r_2 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot \\ r_1 & r_0 & r_1 & r_2 & 0 & \cdot & \cdot & \cdot & \cdot \\ r_2 & r_1 & r_0 & r_1 & r_2 & 0 & \cdot & \cdot & \cdot \\ 0 & r_2 & r_1 & r_0 & r_1 & r_2 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 0 & r_2 & r_1 & r_0 & r_1 & r_2 & 0 \\ \cdot & \cdot & \cdot & 0 & r_2 & r_1 & r_0 & r_1 & r_2 \\ \cdot & \cdot & \cdot & \cdot & 0 & r_2 & r_1 & r_0 & r_1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 & r_2 & r_1 & r_0 \end{bmatrix}. \tag{1}$$

The most common way of inverting band matrices could be:

1. First perform triangular factorization of matrices, say, LU decomposition, Cholesky factorization, etc.
2. Solve iteratively the triangular systems. e.g., first solve $L\boldsymbol{y}_i=\boldsymbol{e}_i$, for $i=1,\ldots,n$, where $\boldsymbol{e}_i$ is the ith column vector of the identity matrix; and then solve $U\boldsymbol{x}_i=\boldsymbol{y}_i$, for $i=1,\ldots,n$. Then $\boldsymbol{x}_i$ is the ith column of A^{-1}.

The triangular factorizations used in step 1 are efficient and require only $O(n)$ operations due to the band structure the matrix A. Furthermore, the resulting triangular matrices are also banded and of the same bandwidths as that of A. Solving each triangular system in step 2 takes also $O(n)$ operations. Therefore, the whole matrix inversion takes $O(n^2)$ operations.

For Toeplitz matrices, one of the nice properties of a symmetric positive-definite Toeplitz matrix is that its complete inverse can be calculated in $O(n^2)$ operations (Golub and Van Loan, 1989, p. 188). We will not explore how it is done further.

In this paper, we develop a new algorithm that can compute the inverse of the 5-band Toeplitz matrix (1) in $O(n^2)$ operations. This feature is no better than — but at least as good as — the two above-mentioned methods. However, one feature that the other two methods do not have is the ability that this new algorithm can compute the inverse entry by entry. After a liner-time overhead has been performed, computing each entry only requires constant time. This feature makes this algorithm valuable when there are only a small number of the entries of the inverse needed. One application of this algorithm is demonstrated in this paper. By applying this algorithm, we are able to develop a way to compute the generalized cross validated (GCV) smoothing spline in linear time for the equally spaced data case.

This paper is organized as follows. In Section 2, we present the new algorithm. In Section 3, we review smoothing splines and demonstrate how this algorithm is used in computing spline estimates. We conclude the paper by a brief summary in Section 4.

2. The new algorithm

This algorithm is derived by brute force. Recall that the (j,i)th entry of A^{-1} can be computed as the ratio of the cofactor of the element A_{ij} (defined as the product of $(-1)^{i+j}$ and the determinant of the matrix obtained from A by deleting the ith row and jth column) and the determinant of A. Utilizing the band pattern of the matrix A, we are able to compute any entry of A^{-1} efficiently by the following algorithm.

Let A^k be the $k \times k$ symmetric 5-band matrix of the same form as A. Also define $\tilde{A}^k$ to be the $k \times k$ matrices obtained by removing the last row and the kth column of A^{k+1}. Let $A^k(j)$ be the $(k-1) \times (k-1)$ matrix obtained by removing the first row and the jth column of A^k. Denote α_k, $\tilde{\alpha}_k$, and $\alpha_{k,j}$ the determinants of matrices A^k, $\tilde{A}^k$, and $A^k(j)$, respectively. We also need to compute β_k, the determinant of the $k \times k$ matrix $F^k(=A^{k+1}(k+1))$ obtained by deleting the first row and the last column of the matrix A^{k+1}.

Algorithm (Computing any entry of the inverse of A^n).

1. Recursively compute $\{\alpha_k,\ k = 1,2,\ldots,n\}$ and $\{\tilde{\alpha}_k,\ k = 1,2,\ldots,n-1\}$ as follows:

$$\alpha_k = r_0\alpha_{k-1} - r_1\tilde{\alpha}_{k-1} + r_2 r_1\tilde{\alpha}_{k-2} - r_0 r_2^2\alpha_{k-3} + r_2^4\alpha_{k-4},$$

$$\tilde{\alpha}_k = r_1\alpha_{k-1} - r_1 r_2\alpha_{k-2} + r_2^2\tilde{\alpha}_{k-2}$$

with initial conditions

$$\alpha_{-3} = \alpha_{-2} = \alpha_{-1} = 0, \quad \alpha_0 = 1,$$

$$\tilde{\alpha}_{-1} = \tilde{\alpha}_0 = 0.$$

2. Recursively compute $\{\beta_k,\ k = 1,2,\ldots,n-1\}$ by

$$\beta_k = r_1\beta_{k-1} - r_2 r_0\beta_{k-2} + r_2^2 r_1\beta_{k-3} - r_2^4\beta_{k-4}$$

with initial conditions $\beta_{-3} = \beta_{-2} = \beta_{-1} = 0,\ \beta_0 = 1$.

3. To compute (j,i)th entry of $(A^n)^{-1}$ for $j \geqslant i$, we first compute the cofactor of the element A^n_{ij}, $\mathrm{Cof}\,A^n_{ij}$, as

$$\mathrm{Cof}\,A^n_{ii} = \alpha_{i-1}\alpha_{n-i} - r_2^2\alpha_{i-2}\alpha_{n-i-1} \quad \text{for } 1 \leqslant i \leqslant n,$$

$$\mathrm{Cof}\,A^n_{ij} = (-1)^{i+j}\{\alpha_{i-1}\alpha_{n-i+1,j-i+1} - r_2\tilde{\alpha}_{i-1}\alpha_{n-i,j-i} + r_2^3\alpha_{i-2}\alpha_{n-i-1,j-i-1}\} \quad \text{for } 1 \leqslant i < j \leqslant n,$$

where $\alpha_{k,j} = \beta_{j-1}\alpha_{k-j} - r_2\beta_{j-2}\tilde{\alpha}_{k-j} + r_2^3\beta_{j-3}\alpha_{k-j-1} \quad \text{for } 1 \leqslant j \leqslant k \leqslant n.$

4. Then $(A^n)^{-1}_{ji} = (\mathrm{Cof}\,A^n_{ij})/\alpha_n$.

We remark that the correctness of this algorithm was verified by a computer program.

The complexity of this algorithm is easily seen to be $O(n)$ in steps 1 and 2, and computing one entry in step 3 is $O(1)$. Therefore, if we want to get the inverse of A, the best we can do from this algorithm is $O(n^2)$, since the inverse of a band matrix is in general a full matrix. However, in some applications, we may need only some entries of A^{-1}. In Section 3, we will give an example that only a multiple of n entries of A^{-1} are needed in the computation.

3. An application

In this section, we give an example that can utilize the nice feature of the algorithm mentioned in Section 2 to achieve computational efficiency. We show that the smoothing spline estimate, when the smoothing parameter is chosen by the generalized cross validation method (described below), can be computed in $O(n)$ operations for equally spaced data.

3.1. Smoothing splines

Given noisy data $\{y_i,\ i = 1,2,\ldots,n\}$ observed from an unknown function g at $\{t_i,\ i = 1,2,\ldots,n\}$ in the interval [0,1], we consider the following nonparametric regression model:

$$y_i = g(t_i) + \varepsilon_i \quad \text{for } i = 1,2,\ldots,n,$$

where ε_i's are uncorrelated errors with mean zero and common variance σ^2.

The smoothing spline estimate g_λ is the minimizer of the following variational problem:

$$\min_{f\in H}\frac{1}{n}\sum_{i=1}^{n}(y_i-f(t_i))^2+\lambda\int_0^1(f^{(m)}(t))^2\,\mathrm{d}t, \tag{2}$$

where H is the Sobolev space $W_2^m=\{f|f^{(v)}$ are absolutely continuous for $v=1,2,\ldots,m-1$ and $f^{(m)}\in L_2[0,1]\}$. The smoothing parameter λ controls the trade-off between "closeness" to the data as measured by the first term and "roughness" of the solution measured by the second term. The estimate g_λ of g can be shown (Wahba, 1990) to be

$$g_\lambda=\sum_{i=1}^{n}c_i\xi_i+\sum_{j=1}^{m}d_j\phi_j,$$

where the c_i's and d_j's are real numbers, and

$$\xi_i(t)=\int_0^1\frac{(t-u)_+^{m-1}(t_i-u)_+^{m-1}}{((m-1)!)^2}\,\mathrm{d}u\quad\text{for } i=1,2,\ldots,n,$$

$$\phi_j(t)=\frac{t^{j-1}}{(j-1)!}\quad\text{for } j=1,2,\ldots,m \tag{3}$$

with $(x)_+=\max(x,0)$.

Let Σ be the $n\times n$ matrix with (i,j)th entry $\xi_j(t_i)$ and T be the $n\times m$ matrix with (i,j)th entry $\phi_j(t_i)$. Note that the polynomial basis $\{\phi_j\}$ are in the null space of the smoothing penalty functional $J(f)=\int_0^1(f^{(m)}(t))^2\,\mathrm{d}t$. Letting $\boldsymbol{c}=(c_1,c_2,\ldots,c_n)^{\mathrm{t}}$, $\boldsymbol{d}=(d_1,d_2,\ldots,d_m)^{\mathrm{t}}$, and $\boldsymbol{y}=(y_1,y_2,\ldots,y_n)^{\mathrm{t}}$, (2) then can be shown to be equivalent to

$$\min_{\boldsymbol{c},\boldsymbol{d}}\frac{1}{n}\|\boldsymbol{y}-(\Sigma\boldsymbol{c}+T\boldsymbol{d})\|^2+\lambda\boldsymbol{c}^{\mathrm{t}}\Sigma\boldsymbol{c},$$

which is equivalent to solving the following linear system of equations:

$$\begin{aligned}&(\Sigma+n\lambda I)\boldsymbol{c}=\boldsymbol{y}-Td,\\&T^{\mathrm{t}}\boldsymbol{c}=0.\end{aligned} \tag{4}$$

Let $M=\Sigma+n\lambda I$. Then the solution of (4) can be expressed explicitly by

$$\begin{aligned}&\boldsymbol{d}=(T^{\mathrm{t}}M^{-1}T)^{-1}T^{\mathrm{t}}M^{-1}\boldsymbol{y},\\&\boldsymbol{c}=M^{-1}(I-T(T^{\mathrm{t}}M^{-1}T)^{-1}T^{\mathrm{t}}M^{-1}\boldsymbol{y}).\end{aligned} \tag{5}$$

Further, it can be shown that the solution is unique provided that T is of full column rank. See, for example, Wahba (1990).

Denote $\boldsymbol{g}_\lambda=(g_\lambda(t_1),g_\lambda(t_2),\ldots,g_\lambda(t_n))^{\mathrm{t}}$. From (5), it can be easily shown that $\boldsymbol{g}_\lambda$ is a linear estimate since it can be expressed by

$$\boldsymbol{g}_\lambda=A(\lambda)\boldsymbol{y},$$

where the influence matrix (or "hat matrix")

$$A(\lambda)=\Sigma M^{-1}(I-T(T^{\mathrm{t}}M^{-1}T)^{-1}T^{\mathrm{t}}M^{-1})+T(T^{\mathrm{t}}M^{-1}T)^{-1}T^{\mathrm{t}}M^{-1}.$$

3.2. Special structure of Σ

Shiau (1998) showed that the matrix Σ can be transformed into a symmetric $(2m-1)$-band matrix. To describe the transformation, we first define an $(n-m)\times n$ matrix Δ that can transform the vector $\boldsymbol{g} = (g(t_1), g(t_2), \ldots, g(t_n))^{\mathrm{t}}$ to an $(n-m)$-vector corresponding to the second divided difference of g. Here we adopt the definition and the notation used in de Boor (1978).

Denote the mth divided difference of a function g at points $t_i, t_{i+1}, \ldots, t_{i+m}$ by $[t_i, \ldots, t_{i+m}]g$. Assume that t_i's are all distinct and $t_i < t_j$ for $i < j$. We remark that the problem of repeated observations can be resolved by averaging repeated observations and assigning appropriate weights to data points. Let Δ be the $(n-m)\times n$ with (i,j)th entry

$$(\Delta)_{ij} = \begin{cases} \displaystyle\prod_{k=i,\,k\neq j}^{i+m} (t_j - t_k)^{-1} & \text{for } i \leqslant j \leqslant i+m, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Note that Δ is an upper triangular matrix of upper bandwidth $q = m$. Then

$$\Delta \begin{bmatrix} g(t_1) \\ g(t_2) \\ \cdot \\ \cdot \\ g(t_n) \end{bmatrix} = \begin{bmatrix} [t_1, \ldots, t_{1+m}]g \\ [t_2, \ldots, t_{2+m}]g \\ \cdot \\ \cdot \\ [t_{n-m}, \ldots, t_n]g \end{bmatrix}.$$

For example, letting $m = 2$, $n = 5$, and $t_i = i/5$, we have, from (6),

$$\Delta = \frac{5^2}{2} \begin{bmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix}.$$

Having defined Δ, we now quote a proposition given in Shiau (1998).

Proposition 1 (Shiau, 1998). *$\Delta\Sigma\Delta^{\mathrm{t}}$ is a symmetric $(2m-1)$-band matrix.*

3.3. Generalized cross validation

It is also well known that the choice of the smoothing parameter is crucial to spline estimates as well as to that in any smoothing techniques. There are many ways of choosing λ. One of the most popular ones is the GCV method proposed by Craven and Wahba (1979). They proposed to estimate λ by the minimizer of the following GCV function:

$$V(\lambda) = \frac{n\,\|(I - A(\lambda))\boldsymbol{y}\|^2}{(\mathrm{tr}(I - A(\lambda)))^2}. \tag{7}$$

The GCV method has been proven to provide a nice estimate of λ theoretically and numerically. For the theoretical results on the efficiency of GCV estimate of λ, see Craven and Wahba (1979), Speckman (1982), and Li (1985, 1986).

Numerous algorithms are available for computing smoothing splines. Reinsch's algorithm (Reinsch, 1967) is an $O(n)$ algorithm for computing the spline estimate g_λ for fixed λ. The difficulty of computing the GCV function defined in (7) lies on the computation of the trace in the denominator.

Utreras (1980) proposed an approximation to the trace of $A(\lambda)$ in the case of equally spaced data. This approximation requires $O(n)$ operations for its calculation, so the GCV function can be computed cheaply. Utreras (1981) considered the case of not necessarily equally spaced data and obtained an approximation that has an initial overhead of finding the lowest n eigenvalues of a $2n \times 2n$ band matrix of bandwidth 5 (for $m = 2$) that requires $O(n^2)$ operations. Based on the special structure of cubic splines, Silverman (1984) modified Utreras' approximation and developed a linear-time procedure called "Asymptotic Generalized Cross-Validation" to obtain the smoothing parameter. Girard (1989) and Hutchinson (1989) proposed Monte Carlo approximations for the trace of $I - A(\lambda)$. More recently, this Monte Carlo approach has been popularized by Wahba and her collaborators. In this approach, the GCV function (7) is replaced by a randomized GCV function defined by

$$\text{ran } V(\lambda) = \frac{n\|(I - A(\lambda))\boldsymbol{y}\|^2}{(\boldsymbol{\xi}^{\mathrm{t}}(I - A(\lambda))\boldsymbol{\xi})^2},$$

where $\boldsymbol{\xi}$, an $n \times 1$ random vector, is normally distributed with mean $\mathbf{0}$ and covariance matrix I. See Wahba et al. (1995) and Gong et al. (1998) for some successful examples using this randomized GCV method. Note that these fast algorithms involve some kind of approximation. They either truncate some smaller eigenvalues, or approximate the trace of $A(\lambda)$, or both.

Based on Reinsch's algorithm, Hutchinson and de Hoog (1985) developed a linear time procedure to compute the GCV smoothing spline in the general, not necessarily equally spaced or uniformly weighted case. They provided a method for computing the trace of $A(\lambda)$, which requires just order $m^2 n$ operations. O'Sullivan (1985) proposed a simple algorithm to compute the trace of $A(\lambda)$, which requires $O(n)$ operations after an overhead of performing a Cholesky decomposition of a band matrix. Hutchinson and de Hoog (1985) and O'Sullivan (1985) appeared to be the first to provide "exact" linear time algorithms for computing GCV smoothing splines. We note that their approach, although completely different from ours, is actually based on a very similar structure.

We develop another linear time algorithm in the next subsection with no intention to compete with the above mentioned algorithms, but simply as an illustrative example on potential usefulness of the algorithm proposed in Section 2. We believe there are many other potential applications.

3.4. Efficient algorithms for smoothing splines

Let $p(t)$ be any polynomial of degree less than or equal to $m - 1$ and $\boldsymbol{p} = (p(t_1), p(t_2), \ldots, p(t_n))^{\mathrm{t}}$. One well-known property of the divided difference is that $\Delta \boldsymbol{p} = \mathbf{0}$. Thus, $\Delta T = 0$, by the definition of T and (3).

Since $\Delta T = 0$ and $T^{\mathrm{t}}\boldsymbol{c} = 0$, by the assumption that T is of rank m, we can express $\boldsymbol{c}$ uniquely as $\Delta^{\mathrm{t}}\boldsymbol{\gamma}$, for some $(n - m)$-vector $\boldsymbol{\gamma}$. Multiplying both sides of the first equation in (4) by Δ and plugging $\boldsymbol{c}$ by $\Delta^{\mathrm{t}}\boldsymbol{\gamma}$, the system of equations (4) can be simplified into

$$\Delta(\Sigma + n\lambda I)\Delta^{\mathrm{t}}\boldsymbol{\gamma} = \Delta \boldsymbol{y}, \tag{8}$$

since $\Delta T = 0$. Let $W = \Delta\Sigma\Delta^{\mathrm{t}} + n\lambda\Delta\Delta^{\mathrm{t}}$. Then (8) becomes

$$W\boldsymbol{\gamma} = \Delta \boldsymbol{y}. \tag{9}$$

First note that $\Delta \boldsymbol{y}$ can be computed in $O(n)$ operations by the band structure of Δ. Further, by Proposition 1 and the fact that $\Delta\Delta^{\mathrm{t}}$ is a $(2m + 1)$-band matrix, W is a symmetric $(2m + 1)$-band matrix. Thus, $\boldsymbol{\gamma}$ can be solved in linear time by the method similar to the one described in Section 1 for band matrices. The solution of (4) becomes

$$\begin{aligned} \boldsymbol{c} &= \Delta^{\mathrm{t}}\boldsymbol{\gamma}, \\ \boldsymbol{d} &= (T^{\mathrm{t}}T)^{-1}T^{\mathrm{t}}(\boldsymbol{y} - \Sigma\boldsymbol{c}). \end{aligned} \tag{10}$$

Then it is easy to see that $\boldsymbol{c}$ can be computed in linear time since Δ^{t} is a $(m+1)$-band matrix. However, if $\boldsymbol{d}$ is computed by (10), then it needs $O(n^2)$ operations since computing $\Sigma\boldsymbol{c}$ takes $O(n^2)$ operations. Instead, by (4), we can consider solving the following the system of equations:

$$T\boldsymbol{d} = \boldsymbol{y} - \Sigma\boldsymbol{c} - n\lambda\boldsymbol{c}. \tag{11}$$

At the first glance, it seems that the computation will take $O(n^2)$ operations since it also involves computing $\Sigma\boldsymbol{c}$. Fortunately, by noting that $\boldsymbol{d}$, the unknown vector to solve, has only m entries, and there are n equations in system (11), it is found that we only need m equations, say, the first m equations, to solve for $\boldsymbol{d}$. Then, we only need the first m rows of $\Sigma\boldsymbol{c}$, and each row takes $O(n)$ operations to compute. In this way, $\boldsymbol{d}$ can be obtained in linear time.

Now, for computing the GCV function $V(\lambda)$, observe that, by (4) and (9), we have

$$(I - A(\lambda))\boldsymbol{y} = \boldsymbol{y} - \Sigma\boldsymbol{c} - T\boldsymbol{d} = n\lambda\boldsymbol{c} = n\lambda\Delta^{\mathrm{t}}\boldsymbol{\gamma} = n\lambda\Delta^{\mathrm{t}}W^{-1}\Delta\boldsymbol{y}.$$

Thus, $I - A(\lambda)$ and $V(\lambda)$ can be simplified, respectively, into

$$I - A(\lambda) = n\lambda\Delta^{\mathrm{t}}W^{-1}\Delta$$

and

$$V(\lambda) = \frac{n\|\boldsymbol{c}\|^2}{(\mathrm{tr}(\Delta^{\mathrm{t}}W^{-1}\Delta))^2}.$$

For equally spaced data, W has the same pattern as that of the A^n matrix described in the previous section. Since $\mathrm{tr}(\Delta^{\mathrm{t}}W^{-1}\Delta) = \mathrm{tr}(W^{-1}\Delta\Delta^{\mathrm{t}})$, we actually only need to compute the central $(2m+1)$ bands of W^{-1} since only the central $(2m+1)$ bands of $\Delta\Delta^{\mathrm{t}}$ are nonzero. Thus, $V(\lambda)$ can be computed in $O(n)$ operations.

For unequally spaced data, the matrix W, which is banded, but unfortunately is not a Toeplitz matrix. Note that it is still true that only the central $(2m+1)$ bands of W^{-1} are needed in this case. Hutchinson and de Hoog (1985) provided a linear-time algorithm for computing the central $(2m+1)$ bands of the inverse of band matrices like W. Therefore, by using their algorithm to compute the central $(2m+1)$ bands of W^{-1}, we can still have a linear-time algorithm for unequally spaced data. However, this is not an application of the algorithm described in Section 2.

For the partial spline models discussed in Chapter 6 of Wahba (1990), this algorithm can be generalized with some modifications since the difference between the ordinary smoothing spline and the partial spline is the matrix T. The modifications are needed because we no longer have the property that $\Delta T = 0$, since in partial splines the T matrix has extra columns for the parametric component in the partial spline model that cannot be annihilated by the divided difference matrix Δ. For some cases that ΔT is sparse, the same efficiency can be obtained. Shiau (1998) discussed in details how this can be done for computing GCV smoothing splines for estimating functions with discontinuities, a problem can be modeled by a partial spline model. In this particular model, the augmented columns of T are piecewise polynomials of degree less than m, hence ΔT is very sparse. However, the W matrix is not a Toeplitz matrix; it has a few irregular entries. So the algorithms developed in this paper for the ordinary smoothing splines cannot be directly applied to that application. With some modifications, Shiau (1998) provided a linear-time algorithm for equally spaced data and a quadratic time algorithm for unequally spaced data for computing GCV spline solutions to this particular problem.

4. Summary

In this paper, we have developed a new algorithm that can compute the inverse of a 5-band Toeplitz matrix. This algorithm has a new feature that other efficient algorithms do not have – it can compute individual entries of the inverse efficiently. After a linear-time overhead has been performed, each entry of the inverse can be computed in constant time. Therefore, this algorithm may provide computational efficiency when the number

of the entries of the inverse needed is small compared to n^2. This may happen when some of the matrices or vectors in the matrix computation are sparse. We have demonstrated in the paper that the GCV smoothing spline computation is one example of such instances.

Five-band Toeplitz matrices arise in many applications. It is very likely that this new algorithm will find itself useful in many other applications.

Acknowledgements

The author would like to express her gratitude to the editor and a referee for helpful suggestions. This work was partially supported by National Science Council of Republic of China under Grant No. NSC87-2118-M-009-004.

References

Craven, P., Wahba, G., 1979. Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. Numer. Math. 31, 377–403.

de Boor, C., 1978. A Practical Guide to Splines. Springer, New York.

Dongarra, J., Bunch, J., Mosler, C., Stewart, G., 1979. Linpack Users' Guide. SIAM, Philadelphia, PA.

Girard, D., 1989. A fast Monte Carlo cross-validation procedure for large least squares problems with noisy data. Numer. Math. 56, 1–23.

Golub, G.H., Van Loan, C.F., 1989. Matrix Computations, 2nd Edition. The Johns Hopkins University Press, Baltimore, MD.

Gong, J., Wahba, G., Johnson, D.R., Tribbia, J., 1998. Adaptive tuning of numerical weather prediction models: simultaneous estimation of weighting, smoothing and physical parameters. Monthly Weather Rev. 125, 210–231.

Hutchinson, M.F., 1989. A stochastic estimator for the trace of the influence matrix for Laplacian smoothing splines. Comm. Statist.: Simulation Comput. 18, 1059–1076.

Hutchinson, M.F., de Hoog, F.R., 1985. Smoothing noisy data with spline functions. Numer. Math. 47, 99–106.

Li, K.C., 1985. From Stein's unbiased risk estimates to the method of generalized cross-validation. Ann. Statist. 13, 1352–1377.

Li, K.C., 1986. Asymptotic optimality of C_L and generalized cross-validation in ridge regression with application to spline smoothing. Ann. Statist. 14, 1101–1112.

O'Sullivan, F., 1985. Comments on "some aspects of the spline smoothing approach to non-parametric regression curve fitting" by B. Silverman. J. Roy. Statist. Soc. Ser. B 47, 39–40.

Reinsch, C., 1967. Smoothing by spline functions. Numer. Math. 10, 177–183.

Shiau, J.-J.H., 1998. Efficient algorithms for generalized cross validated smoothing splines with discontinuity. Manuscript, Institute of Statistics, National Chiao Tung University.

Silverman, B.W., 1984. A fast and efficient cross-validation method for smoothing parameter choice in spline regression. J. Amer. Statist. Assoc. 79, 584–589.

Speckman, P., 1982. Efficient nonparametric regression with cross-validation smoothing splines. Manuscript, Dept. of Statistics, University of Missouri-Columbia.

Utreras, F., 1980. Sur le choix du parametre D'adjustement dans le lissage par fonctions spline. Numer. Math. 34, 15–28.

Utreras, F., 1981. Optimal smoothing of noisy data using spline functions. SIAM J. Scientific Statist. Comput. 2, 349–362.

Wahba, G., 1990. Spline Models for Observational Data. SIAM, Philadelphia, PA.

Wahba, G., Johnson, D.R., Gao, F., Gong, J., 1995. Adaptive tuning of numerical weather prediction models: randomized GCV in three and four dimensional data assimilation. Monthly Weather Rev. 123, 3358–3369.