DEPARTMENT OF STATISTICS
University of Wisconsin
1210 West Dayton St.
Madison, WI 53706

TECHNICAL REPORT NO. 847

December 1988

Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method [1]

by

Chong Gu and Grace Wahba

[1]Research supported in part by AFOSR under grant AFOSR-87-0171, by NASA under contract NAG5-316, and by NSF under grant ATM-8410373.

Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method *

Chong Gu and Grace Wahba

Department of Statistics

University of Wisconsin-Madison

December 1988

Abstract

The (modified) Newton method (Gill et al., 1981) is adapted to optimize GCV/GML scores with multiple smoothing parameters. The main concerns in solving the optimization problem are the speed and the reliability of the algorithm, as well as the invariance of the algorithm under transformations under which the problem itself is invariant. The proposed algorithm is believed to be the most efficient for the problem, though it is still rather expensive for large data sets, since its operational counts are $(2/3)kn^3 + O(n^2)$*, with* k *the number of smoothing parameters and* n *the number of observations. Sensible procedures for computing good starting values are also proposed, which should help in keeping the execution load to the minimum possible. The algorithm is implemented in the RKPACK (Gu, 1988) and illustrated by examples of fitting additive and interaction spline models. It is noted that the algorithm can also be applied to the ML and REML estimation of the variance component models.*

Key words and phrases: *additive/interaction spline models, gradient, Hessian, invariance, Newton method, smoothing parameters, starting values.*

*Research supported in part by AFOSR under grant AFOSR-87-0171, by NASA under contract NAG5-316, and by NSF under grant ATM-8410373.

1 Introduction

Suppose we observe

$$y_j = L_j f + \epsilon_j \quad j = 1, \cdots, n$$

where the L_j's are bounded linear functionals in a Hilbert space $\mathcal{H}$, and the ϵ_j's are *i.i.d.* Gaussian noise with possibly unknown variance σ^2. Such setups encompass a broad range of smoothing and indirect sensing problems. The solution f_λ to the variational problem

$$\min \frac{1}{n}\sum_{j=1}^{n}(y_j - L_j f)^2 + \lambda \|P_1 f\|^2 \tag{1}$$

is called a spline in a general sense (Kimeldorf and Wahba, 1971), where P_1 is a projection operator to a subspace $\mathcal{H}_1$ with codimension M, $\|\cdot\|$ is the norm in $\mathcal{H}$, and λ is the so-called smoothing parameter. The parameter λ controls the tradeoff between the residual sum of squares $\sum_{j=1}^{n}(y_j - L_j f_\lambda)^2$ and the penalty $\|P_1 f_\lambda\|^2$. f_λ is called a regularized estimate of f in the literature of ill-posed problems, and λ the regularization parameter, see, e.g., (O'Sullivan, 1986) and references cited therein.

Denote ξ_j as the representer of L_j, it is derived by (Kimeldorf and Wahba, 1971) that

$$f_\lambda = \sum_{j=1}^{n} c_j (P_1 \xi_j) + \sum_{\nu=1}^{M} d_\nu \phi_\nu, \tag{2}$$

where $\{\phi_\nu\}_{\nu=1}^{M}$ span $\mathcal{H}_0$, the null space of P_1. $\boldsymbol{c} = (c_1, \cdots, c_n)$, $\boldsymbol{d} = (d_1, \cdots, d_M)$ are solutions to the minimization problem

$$\min \frac{1}{n}\|\boldsymbol{y} - S\boldsymbol{d} - \tilde{Q}\boldsymbol{c}\|^2 + \lambda \boldsymbol{c}^T \tilde{Q} \boldsymbol{c}, \tag{3}$$

where

$$\begin{aligned} \tilde{Q} &= (< P_1\xi_{j_1}, P_1\xi_{j_2} >) \\ S &= (L_j \phi_\nu), \end{aligned} \tag{4}$$

with $< \cdot, \cdot >$ indicating the inner product in $\mathcal{H}$. It can be shown (Wahba, 1984) that the solution to the linear system

$$\begin{aligned} (\tilde{Q} + n\lambda I)\boldsymbol{c} + S\boldsymbol{d} &= \boldsymbol{y} \\ S^T \boldsymbol{c} &= 0 \end{aligned} \tag{5}$$

is a minimizer of (3), and when $\tilde{Q}$ is of full rank, it is the unique minimizer.

Consider an orthogonal decomposition of $\mathcal{H}$ into more than two components, $\mathcal{H} = \oplus_{i=0}^{k}\mathcal{H}_i$. A direct generalization of (1) is

$$\min \frac{1}{n}\sum_{j=1}^{n}(y_j - L_j f)^2 + \sum_{i=1}^{k}\lambda_i \|P_i f\|^2, \tag{6}$$

where the λ_i's are a set of smoothing parameters and P_i is the orthogonal projection operator onto $\mathcal{H}_i$. Writing $\lambda_i = \lambda/\theta_i$, We can rewrite (6) as

$$\min \frac{1}{n}\sum_{j=1}^{n}(y_j - L_j f)^2 + \lambda \|P_* f\|_{\boldsymbol{\theta}}^2, \tag{7}$$

where $P_* = \sum_{i=1}^{k} P_i$ is the projection operator onto $\mathcal{H}_* = \oplus_{i=1}^{k}\mathcal{H}_i$, and

$$\|f\|_{\boldsymbol{\theta}}^2 = \|P_0 f\|^2 + \sum_{i=1}^{k}\theta_i^{-1}\|P_i f\|^2$$

is a modified norm indexed by $\boldsymbol{\theta}$, where $\|\cdot\|$ is the original norm. It can be shown that the representer of L_j under the norm $\|\cdot\|_{\boldsymbol{\theta}}$ is

$$\xi_j^{\boldsymbol{\theta}} = (P_0\xi_j) + \sum_{i=1}^{k}\theta_i(P_i\xi_j),$$

where ξ_j is its representer under the norm $\|\cdot\|$. Denoting $<\cdot,\cdot>$, $<\cdot,\cdot>_{\boldsymbol{\theta}}$ as the inner products corresponding to the norms $\|\cdot\|$, $\|\cdot\|_{\boldsymbol{\theta}}$ respectively, we have

$$\tilde{Q}_*^{\boldsymbol{\theta}} = (<P_*\xi_{j_1}^{\boldsymbol{\theta}}, P_*\xi_{j_2}^{\boldsymbol{\theta}}>_{\boldsymbol{\theta}}) = \sum_{i=1}^{k}\theta_i\tilde{Q}_i, \tag{8}$$

where $\tilde{Q}_i = (<P_i\xi_{j_1}, P_i\xi_{j_2}>)$. Thus the solution to (6) can be written as

$$f_{\boldsymbol{\lambda}} = \sum_{j=1}^{n} c_j\xi_j^{\boldsymbol{\theta}} + \sum_{\nu=1}^{M} d_\nu\phi_\nu,$$

with $\boldsymbol{c}$, $\boldsymbol{d}$ determined by

$$\min \frac{1}{n}\|\boldsymbol{y} - S\boldsymbol{d} - \tilde{Q}_*^{\boldsymbol{\theta}}\boldsymbol{c}\|^2 + \lambda\boldsymbol{c}^T\tilde{Q}_*^{\boldsymbol{\theta}}\boldsymbol{c}. \tag{9}$$

Choosing appropriate smoothing parameters $\boldsymbol{\lambda}$ is crucial for effectively recovering truth functions from the data by fitting spline models. Two of commonly recognized data-driven methods for choosing smoothing parameters are the generalized cross validation (GCV) and the generalized maximum likelihood (GML) methods.

Writing

$$\tilde{Q}_*^{\boldsymbol{\theta}}\boldsymbol{c} + S\boldsymbol{d} = (L_1 f_{\boldsymbol{\lambda}}, \cdots, L_n f_{\boldsymbol{\lambda}})^T = A(\boldsymbol{\lambda})\boldsymbol{y},$$

the GCV method seeks $\boldsymbol{\lambda}$ to minimize

$$V(\lambda) = \frac{(1/n)\|(I - A(\boldsymbol{\lambda}))\boldsymbol{y}\|^2}{[(1/n)\mathrm{tr}(I - A(\boldsymbol{\lambda}))]^2}.$$

$A(\boldsymbol{\lambda})$ is the so-called influence matrix. Let

$$S = FR = (F_1, F_2)\begin{pmatrix} R_1 \\ 0 \end{pmatrix}$$

be the QR-decomposition of S, it can be shown (Wahba, 1984) that

$$I - A(\boldsymbol{\lambda}) = n\lambda F_2(\tilde{Q}_*^{\boldsymbol{\theta}} + n\lambda I)^{-1}F_2^T,$$

hence

$$V(\boldsymbol{\lambda}) = V(\lambda, \boldsymbol{\theta}) = n\frac{\boldsymbol{z}^T(Q_*^{\boldsymbol{\theta}} + n\lambda I)^{-2}\boldsymbol{z}}{[\mathrm{tr}(Q_*^{\boldsymbol{\theta}} + n\lambda I)^{-1}]^2}, \tag{10}$$

where $\boldsymbol{z} = F_2^T\boldsymbol{y}$ and

$$Q_*^{\boldsymbol{\theta}} = F_2^T\tilde{Q}_*^{\boldsymbol{\theta}}F_2 = \sum_{i=1}^{k}\theta_i(F_2^T\tilde{Q}_i F_2) = \sum_{i=1}^{k}\theta_i Q_i,$$

where $Q_i = F_2^T\tilde{Q}_i F_2$. The GCV method is proposed by (Craven and Wahba, 1979) and is shown to be asymptotically optimum for minimizing predictive mean square error ((Craven and Wahba, 1979) and (Li, 1986)), see also (Wahba and Wang, 1987), where it is shown that this method is also good for minimizing solution mean square error in a variety of circumstances.

Based on the Bayesian interpretation of the smoothing spline models (Wahba, 1978), Wahba also derives the GML method which seeks the minimizer of

$$M(\boldsymbol{\lambda}) = \frac{\boldsymbol{y}^T(I - A(\boldsymbol{\lambda}))\boldsymbol{y}/n}{[\det{}^+(I - A(\boldsymbol{\lambda}))]^{1/n_1}},$$

where $\det^+(I - A)$ is the product of the $n_1 = n - M$ nonzero eigenvalues of $I - A$, see (Wahba, 1985). In parallel to (10), we have

$$M(\boldsymbol{\lambda}) = M(\lambda, \boldsymbol{\theta}) = \frac{\boldsymbol{z}^T(Q_*^{\boldsymbol{\theta}} + n\lambda I)^{-1}\boldsymbol{z}/n}{[\det(Q_*^{\boldsymbol{\theta}} + n\lambda I)^{-1}]^{1/n_1}}. \tag{11}$$

A theoretical comparison of the GCV and GML methods can be found in (Wahba, 1985). The purpose of this article is to present a numerical algorithm for the computation of selecting the

smoothing parameters according to the GCV or GML criterion. Our main concern is to deal with problems with $k > 1$. The method is developed on the basis of the $k = 1$ algorithm presented by (Gu et al., 1988).

2 Preliminaries

To minimize the functions $V(\lambda, \boldsymbol{\theta})$ or $M(\lambda, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ and $n\lambda$, we wish to iterate on the following cycle:

1. For fixed $\boldsymbol{\theta}$, minimize $V(\lambda|\boldsymbol{\theta})$ or $M(\lambda|\boldsymbol{\theta})$ with respect to $n\lambda$.

2. Update $\boldsymbol{\theta}$ using information from the current estimates.

Step 1 above can be implemented through the single smoothing parameter GCV/GML algorithm based on the Householder tridiagonalization proposed by (Gu et al., 1988). To carry out step 2, we will evaluate the gradient and the Hessian of $V(\boldsymbol{\theta}|\lambda)$ or $M(\boldsymbol{\theta}|\lambda)$ with respect to $\boldsymbol{\eta} = \log(\boldsymbol{\theta})$, then apply the modified Newton method (Gill et al., 1981) to update the $\boldsymbol{\eta}$. In this section, we first discuss the choices for the scaling of the variables ($\boldsymbol{\eta}$) and the scaling of the objective functions ($V(\cdot)$ or $M(\cdot)$). Then we present the expressions of the gradient and the Hessian for later use.

We choose the variables $\boldsymbol{\eta}$ instead of $\boldsymbol{\theta}$ mainly for their *invariance*. Looking at the formulas of $V(\cdot)$ and $M(\cdot)$ ((10) and (11)), we can see that what really matter are the matrices $\theta_i Q_i$'s, while the "face values" of the θ_i's are subject to rescaling when the matrices Q_i's are multiplied by some positive constants. Notice that the problem itself is not changed by such transformations. Invariance under this kind of transformations is hence essential to any sensible algorithm. It is easy to see that the derivatives of $V(\cdot)$ and $M(\cdot)$ with respect to $\boldsymbol{\eta}$ are invariant, hence the methods based on these derivatives are invariant. Standard calculations show that the derivatives of $V(\cdot)$ and $M(\cdot)$ with respect to $\boldsymbol{\theta}$ are in general *not* invariant in this sense, which disqualifies them for serving as the basic variables. Another immediate numerical merit from adopting $\boldsymbol{\eta}$ instead of $\boldsymbol{\theta}$ is that we change a constrained optimization problem ($\boldsymbol{\theta} \geq 0$) to a unconstrained one, which allows much simpler treatment.

The objective functions can always be rescaled by monotone transformations without changing the optima of the problems. In our problem, it is observed that $\log V(\cdot)$ and $\log M(\cdot)$ have simpler derivative formulas. However, this transformation is not adopted for the following reason. To

efficiently *minimize* the objective function, we want the objective function to be as *convex* as possible, since most optimization methods are modeled after convex functions. When the convexity violated, we have to modify the methods and to suffer lower efficiency, and the methods may even fail to converge. Since $\log(\cdot)$ is concave, we can expect $\log f(\cdot)$ to be 'less convex' than $f(\cdot)$ for general $f(\cdot) > 0$. Actually, it can be shown that a positive-definite Hessian of $\log f(\cdot)$ implies a positive-definite Hessian of $f(\cdot)$, but the reverse is not true.

Having chosen the objective functions and the basic variables, we now collect the formulas of the gradient and the Hessian. Define $D = \sum_{i=1}^{k}(e^{\eta_i}Q_i) + n\lambda I = Q_*^{\boldsymbol{\theta}} + n\lambda I$, we write

$$V(\boldsymbol{\eta}|\lambda) = n\frac{\boldsymbol{z}^T D^{-2}\boldsymbol{z}}{(\mathrm{tr}D^{-1})^2},$$

and

$$M(\boldsymbol{\eta}|\lambda) = \frac{\boldsymbol{z}^T D^{-1}\boldsymbol{z}/n}{(\det D^{-1})^{1/n_1}}.$$

We have

Lemma 1

$$\frac{\partial V}{\partial \eta_i} = n(\frac{\dot{r}_i}{t^2} - 2\frac{r\dot{t}_i}{t^3}), \tag{12}$$

$$\frac{\partial^2 V}{\partial \eta_i \partial \eta_j} = n(\frac{\ddot{r}_{ij}}{t^2} - 2\frac{\dot{r}_i\dot{t}_j}{t^3} - 2\frac{\dot{t}_i\dot{r}_j}{t^3} - 2\frac{r\ddot{t}_{ij}}{t^3} + 6\frac{r\dot{t}_i\dot{t}_j}{t^4}), \tag{13}$$

where

$$\dot{r}_i = \frac{\partial r}{\partial \eta_i} = \frac{\partial}{\partial \eta_i}(\boldsymbol{z}^T D^{-2}\boldsymbol{z}) = -2\boldsymbol{z}^T D^{-2}(e^{\eta_i}Q_i)D^{-1}\boldsymbol{z}, \tag{14}$$

$$\dot{t}_i = \frac{\partial t}{\partial \eta_i} = \frac{\partial}{\partial \eta_i}(\mathrm{tr}D^{-1}) = -\mathrm{tr}(D^{-1}(e^{\eta_i}Q_i)D^{-1}), \tag{15}$$

$$\begin{aligned}\ddot{r}_{ij} = {} & 2[\boldsymbol{z}^T D^{-2}(e^{\eta_i}Q_i)D^{-1}(e^{\eta_j}Q_j)D^{-1}\boldsymbol{z} + \boldsymbol{z}^T D^{-1}(e^{\eta_i}Q_i)D^{-2}(e^{\eta_j}Q_j)D^{-1}\boldsymbol{z} \\ & + \boldsymbol{z}^T D^{-1}(e^{\eta_i}Q_i)D^{-1}(e^{\eta_j}Q_j)D^{-2}\boldsymbol{z}] + \delta_{|i-j|}\dot{r}_i,\end{aligned} \tag{16}$$

$$\ddot{t}_{ij} = 2\mathrm{tr}(D^{-1}(e^{\eta_i}Q_i)D^{-1}(e^{\eta_j}Q_j)D^{-1}) + \delta_{|i-j|}\dot{t}_i, \tag{17}$$

where

$$\delta_i = \begin{cases} 1 & i = 0, \\ 0 & i \neq 0. \end{cases}$$

Lemma 2

$$\frac{\partial M}{\partial \theta_i} = \frac{1}{n}\left(\frac{\dot{r}_i}{t^{1/n}} - \frac{1}{n}\frac{r\dot{t}_i}{t^{1+1/n}}\right), \tag{18}$$

$$\frac{\partial^2 M}{\partial \eta_i \partial \eta_j} = \frac{1}{n}\left(\frac{\ddot{r}_{ij}}{t^{1/n_1}} - \frac{1}{n_1}\frac{\dot{r}_i\dot{t}_j}{t^{1+1/n_1}} - \frac{1}{n_1}\frac{\dot{t}_i\dot{r}_j}{t^{1+1/n_1}} - \frac{1}{n_1}\frac{r\ddot{t}_{ij}}{t^{1+1/n_1}} + \frac{1}{n_1}\frac{n_1+1}{n_1}\frac{r\dot{t}_i\dot{t}_j}{t^{2+1/n_1}}\right), \tag{19}$$

where

$$\dot{r}_i = \frac{\partial r}{\partial \eta_i} = \frac{\partial}{\partial \eta_i}(\boldsymbol{z}^T D^{-1}\boldsymbol{z}) = -\boldsymbol{z}^T D^{-1}(e^{\eta_i}Q_i)D^{-1}\boldsymbol{z}, \tag{20}$$

$$\dot{t}_i = \frac{\partial t}{\partial \eta_i} = \frac{\partial}{\partial \eta_i}(\det D^{-1}) = -\det(D^{-1})\mathrm{tr}(e^{\eta_i}Q_i D^{-1}), \tag{21}$$

$$\ddot{r}_{ij} = 2\boldsymbol{z}^T D^{-1}(e^{\eta_i}Q_i)D^{-1}(e^{\eta_j}Q_j)D^{-1}\boldsymbol{z} + \delta_{|i-j|}\dot{r}_i, \tag{22}$$

$$\ddot{t}_{ij} = \det(D^{-1})\mathrm{tr}((e^{\eta_i}Q_i)D^{-1}(e^{\eta_j}Q_j)D^{-1}) + \det(D^{-1})\mathrm{tr}(e^{\eta_i}Q_i D^{-1})\mathrm{tr}(e^{\eta_j}Q_j D^{-1}) + \delta_{|i-j|}\dot{t}_i, \tag{23}$$

where δ_i is the same as in lemma 1.

The proofs to the above two lemmas are straightforward and tedious. Here we omit them.

3 Algorithms

In this section, we will specify the main algorithm and discuss its various aspects. More discussion will be collected in Section 5 as remarks.

Algorithm 1 *Assuming inputs of the null space matrix S, the response vector $\boldsymbol{y}$, the reproducing kernels $\tilde{Q}_i$, $i = 1, \cdots, k$, and the starting values $\boldsymbol{\eta}_0$, we propose the following algorithm:*

1. Initialization:
   (a) *Compute the QR-decomposition of* $S = FR = (F_1, F_2)\begin{pmatrix} R_1 \\ 0 \end{pmatrix}$.
   (b) *Compute* $\boldsymbol{z} = F_2^T\boldsymbol{y}$ *and* $Q_i = F_2^T\tilde{Q}_i F_2$.
   (c) *Set* $\Delta\boldsymbol{\eta} = 0$, $\boldsymbol{\eta}_- = \boldsymbol{\eta}_0$, $V_- = \infty$ *(or* $M_- = \infty$*).*
2. Iteration:
   (a) *For the current try values* $\boldsymbol{\eta} = \boldsymbol{\eta}_- + \Delta\boldsymbol{\eta}$, *collect* $\tilde{D} = Q_*^{\boldsymbol{\theta}} = \sum_{i=1}^k e^{\eta_i}Q_i$.

(b) *Compute* $\tilde{D} = U\tilde{T}U^T$, *where* U *is orthogonal and* $\tilde{T}$ *is tridiagonal. Compute* $\boldsymbol{x} = U^T\boldsymbol{z}$.

(c) *Minimize*

$$V(\lambda|\boldsymbol{\eta}) = \frac{n\boldsymbol{x}^T(\tilde{T}+n\lambda I)^{-2}\boldsymbol{x}}{[\mathrm{tr}(\tilde{T}+n\lambda I)^{-1}]^2} \tag{24}$$

or

$$M(\lambda|\boldsymbol{\eta}) = \frac{\boldsymbol{x}^T(\tilde{T}+n\lambda I)^{-1}\boldsymbol{x}/n}{\det(\tilde{T}+n\lambda I)^{-1/n_1}}. \tag{25}$$

If $V > V_-$ *(or* $M > M_-$*), set* $\Delta\boldsymbol{\eta} = \Delta\boldsymbol{\eta}/2$, *goto* (a); *else proceed.*

(d) *Evaluate the gradient* $\boldsymbol{g} = (\partial/\partial\boldsymbol{\eta})V(\boldsymbol{\eta}|\lambda)$ *(or* $(\partial/\partial\boldsymbol{\eta})M(\boldsymbol{\eta}|\lambda)$*) and the Hessian* $H = (\partial^2/\partial\boldsymbol{\eta}\partial\boldsymbol{\eta}^T)V(\boldsymbol{\eta}|\lambda)$ *(or* $(\partial^2/\partial\boldsymbol{\eta}\partial\boldsymbol{\eta}^T)M(\boldsymbol{\eta}|\lambda)$*). Calculate the increment* $\Delta\boldsymbol{\eta} = -\tilde{H}^{-1}\boldsymbol{g}$, *where* $\tilde{H} = H + \mathrm{diag}(\boldsymbol{e})$ *is positive definite. If* H *itself is positive definite "enough",* $\boldsymbol{e}$ *is simply 0.*

(e) *Check convergence conditions. If the conditions fail, set* $\boldsymbol{\eta}_- = \boldsymbol{\eta}$, $V_- = V$ *(or* $M_- = M$*), goto* (a); *else proceed.*

3. Calculate the optimal model:

(a) *If* $\Delta\eta_i < -\gamma$, *set* $\eta_i = -\infty$, *where* γ *is a "large" number, say,* $\gamma \in (.5, .9)$.

(b) *Collect* $\tilde{D} = \sum_{i=1}^{k} e^{\eta_i}Q_i$. *Calculate the model minimizing* $V(\lambda|\boldsymbol{\eta})$ *(or* $M(\lambda|\boldsymbol{\eta})$*).*

In the above specification, most items clearly explain themselves, except steps *2.*(d), *2.*(e) and *3.*(a), which we will explain in turn.

Step *2.*(d) is the major part of this algorithm. Define $D = \tilde{D} + n\lambda I$ (this definition is consistent with the one in Section 2), $T = \tilde{T} + n\lambda I$, and $K_i = e^{\eta_i}U^TQ_iU$, we can write (14)–(17) as

$$\begin{aligned}
\dot{r}_i &= -2\boldsymbol{x}^TT^{-2}K_iT^{-1}\boldsymbol{x} = -2(T^{-1}\boldsymbol{x})^T(T^{-1}K_i)(T^{-1}\boldsymbol{x}), && (26)\\
\dot{t}_i &= -\mathrm{tr}(T^{-2}K_i), && (27)\\
\ddot{r}_{ij} &= 2[(T^{-1}\boldsymbol{x})^T(T^{-1}K_i)(T^{-1}K_j)(T^{-1}\boldsymbol{x}) + (T^{-1}\boldsymbol{x})^T(T^{-1}K_i)^T(T^{-1}K_j)(T^{-1}\boldsymbol{x})\\
&\quad +(T^{-1}\boldsymbol{x})^T(T^{-1}K_i)^T(T^{-1}K_j)^T(T^{-1}\boldsymbol{x})] + \delta_{|i-j|}\dot{r}_i, && (28)\\
\ddot{t}_{ij} &= 2\mathrm{tr}((T^{-2}K_i)(T^{-1}K_j)) + \delta_{|i-j|}\dot{t}_i. && (29)
\end{aligned}$$

Similarly, (20)–(23) become

$$\begin{aligned}
\dot{r}_i &= -\boldsymbol{x}^T(T^{-1}K_i)(T^{-1}\boldsymbol{x}), && (30)\\
\dot{t}_i &= -\det(T^{-1})\mathrm{tr}(T^{-1}K_i), && (31)\\
\ddot{r}_{ij} &= 2\boldsymbol{x}^T(T^{-1}K_i)(T^{-1}K_j)(T^{-1}\boldsymbol{x}) + \delta_{|i-j|}\dot{r}_i, && (32)
\end{aligned}$$

$$\ddot{t}_{ij} = \det(T^{-1})\mathrm{tr}((T^{-1}K_i)(T^{-1}K_j)) + \det(T^{-1})\mathrm{tr}(T^{-1}K_i)\mathrm{tr}(T^{-1}K_j) + \delta_{|i-j|}\dot{t}_i. \quad (33)$$

The gradients and the Hessians presented in lemma 1 and lemma 2 will be calculated via (26)–(29) and (30)–(33). The modified Newton method based on the modified Cholesky decomposition, as described by (Gill et al., 1981), is adopted to calculate the update direction $\Delta\boldsymbol{\eta} = -\tilde{H}^{-1}\boldsymbol{g}$. Since the score evaluation is expensive, we choose not to perform a step length search, but to simply pick 1 as the default step length. In case it fails to provide an improved score, we keep dividing it by 2 until the first success, which is guaranteed by the fact that $-\tilde{H}^{-1}\boldsymbol{g}$ is a descent direction. In practice, such a failure trial rarely occurs except at places far away from the minimum.

Given a user supplied precision requirement ϵ_A, the algorithm is thought to have converged if at least one of the following criteria is satisfied:

1. (a) $V_- - V < \epsilon_A(1+V)$, and (b) $\|\boldsymbol{g}\|_\infty < \sqrt{\epsilon_A}(1+V)$.

2. $\|\boldsymbol{g}\|_\infty < \epsilon_A$.

The criteria are a modification of the suggestions by (Gill et al., 1981, pp.306-7). Our rule 1.(a) is identical to their U1, while our rule 1.(b) is expected to do what their U2–U3 do. In our problem some optimal η_i could be at $-\infty$. In such a situation, the algorithm will keep wanting to move towards the optimum with big steps even when the score values are well within the required precision of the optimal value. This is a typical ill-conditioned situation where their U2 will never be satisfied. By adopting a more stringent version of their U3 (our rule 1.(b)) instead of their U2–U3 combination, we can avoid such endless iteration and yet deliver qualified termination, since our rule 1.(b) always implies the satisfaction of their U2 when the problem is well-conditioned (Gill et al., 1981, p.307). Our rule 2 is simply their U4. Their U5 is discarded for the same reason discussed above. At this point, the motivation of step *3.*(a) of the algorithm should have become clear.

We now briefly discuss the operational counts of the algorithm. Steps *1.*(a), *1.*(b), and *2.*(a) can be executed with $O(n^2)$ operations, provided M (the rank of the null space matrix S) and k are both of constant order. Step *2.*(b) can be implemented via the Householder tridiagonalization algorithm, which in general takes about $(2/3)n_1^3$ operations, while some time-saving is possible through the distributed truncation proposed by (Gu et al., 1988). Step *2.*(c) is usually performed by a golden section search on $\log(n\lambda)$, each evaluation of the score functions $V(\cdot)$ or $M(\cdot)$ via

formulas (24) or (25) requires only $O(n)$ operations. As we mentioned earlier, step *2.*(d) is the major burden on the algorithm. To calculate each of the $K_i = e^{\eta_i} U^T Q_i U$, we need approximately the same number of operations needed for step *2.*(b). Making use of the identity $\sum_{i=1}^{k} K_i = \tilde{T}$, we need totally $(2/3)(k-1)n_1^3$ operations for the K_i's. Since the linear system $T\boldsymbol{x} = b$ can be solved with $O(n)$ operations, $T^{-1}K_i$'s and $T^{-2}K_i$'s can be obtained with $O(n^2)$ operations. Hence the total number of operations needed for each iteration is in general $(2/3)kn_1^3 + O(n^2)$. For each failure trial with $V > V_-$ or $M > M_-$, we have to spend $(2/3)n_1^3$ operations (step *2.*(b)) before discovering it. Step *3.*(b) needs another $(2/3)n_1^3$ to calculate the final results. The above operational counts are based on related discussions in (Dongarra et al., 1979) and (Golub and Van Loan, 1983), see also (Gu et al., 1988).

Good starting values are important to Newton-type iterative optimization methods. They are even more crucial to our algorithm since our iteration is extremely expensive for large n. Deriving good starting values is not a mere numerical problem but something to do with the original setup of the model. We will propose two sensible approaches below for obtaining good starting values for the algorithm. Both of them are based on the background problem formulation presented in Section 1.

The first proposal is built on the assumption that the optimal smoothing parameters λ_i's share approximately the same decreasing rate as the number of observations increases. We can then randomly select a subset of the observations available, calculate the optimal $\boldsymbol{\theta}$ for the subset, and use these as the starting value for some bigger subset or the complete data set. The idea here is to perform a "crude" search with a smaller problem size to save execution time. If this assumption is strongly believed then a large size iteration can be avoided by simply adopting the $\boldsymbol{\theta}$ obtained from a subset run and going ahead to perform only step *3.*(b) of the algorithm on the complete data set.

The second proposal is more involved. Suppose we knew the underlying truth function f_*, and it could be decomposed as the sum of projections onto orthogonal subspaces $\mathcal{H}_i$'s,

$$f_* = \sum_{i=0}^{k} P_i f_*,$$

where P_i denotes the projection operator onto subspace $\mathcal{H}_i$. When using the smoothing spline model to retrieve the truth function from data, it is sensible to adjust the face values of the norms $\|P_i f\|$ by the "strength" of the corresponding components of the truth function, $\|P_i f_*\|$, to balance the

roughness penalties put on different components. Specifically, when $\|P_1 f_*\| = .07$ and $\|P_2 f_*\| = 7$, say, we might regard $P_1 f$ with $\|P_1 f\| = .05$ as rough as $P_2 f$ with $\|P_2 f\| = 5$. This heuristic leads to the following one smoothing parameter formulation

$$\min \frac{1}{n}\sum_{j=1}^{n}(y_j - L_j f)^2 + \lambda \sum_{i=1}^{k} \frac{\|P_i f\|^2}{\|P_i f_*\|^2},$$

or equivalently,

$$\theta_i = \|P_i f_*\|^2.$$

θ chosen this way should be close to optimal, hence be a good starting value for the iteration. Of course in practice we will never know the truth function f_*, and in turn the factors $\|P_i f_*\|^2$. However, all we need here is just a set of starting values for the iterative algorithm, and some estimates of the factors $\|P_i f_*\|^2$ should suffice. The resulting starting value procedure, which is made default in the implementation, is thus

Algorithm 2 *If no starting values are specified, we calculate the default starting values as follows:*

1. *Set $\theta_i = (\mathrm{tr}(Q_i))^{-1}$, fit the one smoothing parameter spline model by minimizing $V(\lambda|\boldsymbol{\theta})$ or $M(\lambda|\boldsymbol{\theta})$, calculate the parameters $\mathbf{c}$.*
2. *Estimate $\|P_i f_*\|^2$ by $\theta_{i0} = \theta_i^2 \mathbf{c}^T \tilde{Q}_i \mathbf{c}$, and set the starting values of the algorithm 1 to be $\eta_{i0} = \log(\theta_{i0})$.*

The choice of θ_i's in step *1* above is arbitrary but *invariant* in the sense we discussed in Section 2, other invariant selections might be equally appropriate. The estimates of $\|P_i f_*\|^2$'s are simply by replacing $P_i f_*$ with $P_i f_{\boldsymbol{\lambda}} = \sum_j c_j \theta_i (P_i \xi_j)$, where ξ_j is the representer of the linear functional L_i under the original norm $\|\cdot\|$, remember that $< P_i\xi_{j_1}, P_i\xi_{j_2} >= (\tilde{Q}_i)_{j_1 j_2}$. Algorithm 2 takes about $(2/3)n^3 + O(n^2)$ operations for execution.

4 Examples: Additive/interaction spline models

To test the algorithms presented in the previous section, we apply them to fit the additive/interaction spline models, which were first proposed by (Barry, 1983; Barry, 1986) (see also (Wahba, 1986)), and were first illustrated by (Gu et al., 1988) in the $k = 2$ case where a grid search on the 1-dimension θ was feasibly conducted. In this section, we will report some of our numerical experiments with

the iterative multiple smoothing parameter algorithm proposed in this article. Various statistical aspects of the additive/interaction spline models are currently under study. The findings of the study will be presented elsewhere.

The formulation of the additive/interaction spline models is based on the tensor product Hilbert space formulation. Take the component Hilbert spaces (on $[0,1]$) as the reproducing kernel Hilbert space

$$W_2^m = \{f : f^{(\nu)} abs.cont., \nu = 0, \cdots, m-1, \int (f^{(m)})^2 < \infty\}$$

with norm

$$\|f\|^2 = \sum_{\nu=0}^{m-1} (\int_0^1 f^{(\nu)})^2 + \int_0^1 (f^{(m)})^2,$$

see (Craven and Wahba, 1979). We let $\mathcal{H}$ be the tensor product Hilbert space $\mathcal{H} = \otimes_{l=1}^d \mathcal{H}^l$, with $\mathcal{H}^l = W_2^m$. Readers are referred to (Aronszajn, 1950) for technical details on tensor products of reproducing kernel Hilbert spaces. We also note that the component spaces of the tensor product space need not to be of the same form, e.g., we might specify different m's for different $\mathcal{H}^l$'s for our formulation here, though we choose not to do so to keep the notation simple. Write $W_2^m = \mathcal{N} \oplus \mathcal{P}_{m-1} \oplus \mathcal{S}_m$, where $\mathcal{N}$ is the space of constant functions with square norm $(\int_0^1 f)^2$, $\mathcal{P}_{m-1}$ is the space of all polynomials with degrees less than m which integrate to zero, with square norm $\sum_{\nu=1}^{m-1} (\int_0^1 f^{(\nu)})^2$, and $\mathcal{S}_m$ is the space of functions with square integrable mth derivative and satisfy $\int_0^1 f^{(\nu)} = 0$, $\nu = 0, \cdots, m-1$, with square norm $\int_0^1 (f^{(m)})^2$. When $m = 1$, the space $\mathcal{P}_0$ vanishes. (Further decomposition of $\mathcal{P}_{m-1}$ for $m > 2$ is sometimes useful, see (Craven and Wahba, 1979), though we will not get into too much detail here.) From the direct sum decompositions of the component spaces formulated above, the space

$$\mathcal{H} = \otimes_{l=1}^d \mathcal{H}^l = \otimes_{l=1}^d (\mathcal{N}^l \oplus \mathcal{P}_{m-1}^l \oplus \mathcal{S}_m^l)$$

can be represented as the direct sum of 3^d orthogonal subspaces when $m > 1$, and 2^d subspaces when $m = 1$. Various statistical model formulations can be specified from this structure. E.g., the additive models are obtained by eliminating the subspaces with more than one non-constant tensor component (i.e., with less than $d-1$ $\mathcal{N}$'s as components).

We first present an simulated additive spline example on $[0,1]^4$. The truth function is specified by

$$f(\mathbf{x}) = f(x_1, x_2, x_3, x_4) = 10 \sin(\pi x_2) + \exp(3x_3) + 10^6 x_4^{11}(1-x_4)^6 + 10^4 x_4^3 (1-x_4)^{10}.$$

The bounded linear functionals L_j's are chosen as the evaluation functionals $[\mathbf{x}_j]f = f(\mathbf{x}_j)$. We generated 100 sampling points in $[0,1]^4$ randomly, and computed the observations by

$$y_j = f(\mathbf{x}_j) + \epsilon_j,$$

where ϵ_j's are independent Gaussian pseudo random numbers with mean 0 and variance 1. The sampling points were generated by the Fortran routine **uni** of the Core Mathematics Library (Cmlib) from the National Bureau of Standards, with **mdig**=32 and seed 2375, the first 400 (after a null call to pass the seed) random variables were cut into 4 segments of length 100, and formed the 1st to 4th coordinates of the sampling points in the natural order. The ϵ_j's were generated by the routine **rnor** of Cmlib with **mdig**=32, we took the first 100 outcomes after the null call which passed the seed 5732 to the routine. The scatter plot matrix of $\mathbf{x}$ and y is shown in Figure 1. We fitted models with $m = 2, 1$ respectively. For $m = 2$, we selected $\mathcal{H}_0 = \text{span}\{1, x_1 - .5, x_2 - .5, x_3 - .5, x_4 - .5\}$, where 1 spans $\otimes_{l=1}^4 \mathcal{N}^l$, $x_i - .5$ span $(\otimes_{l\neq i}\mathcal{N}^l) \otimes \mathcal{P}_1^i$), $i = 1, \ldots, 4$. The penalized spaces were $\mathcal{H}_i = (\otimes_{l\neq i}\mathcal{N}^l) \otimes \mathcal{S}_2^i$, $i = 1, \cdots, 4$. For $m = 1$, we chose $\mathcal{H}_0 = \{1\}$ and $\mathcal{H}_i = (\otimes_{l\neq i}\mathcal{N}^l) \otimes \mathcal{S}_1^i$, $i = 1, \cdots, 4$. Other subspaces were deleted. The evaluation formulas for the matrices S, $\tilde{Q}_i$, $i = 1, \cdots, 4$, and for the representers $P_i\xi_j$ can be found from (Gu et al., 1988) for the $m = 2$ setting directly, and the $m = 1$ formulas can be similarly derived. Figure 2 and Figure 3 illustrate the fitted additive components (solid lines) compared to the true additive components (dotted lines) for $m = 2, 1$ respectively, with the vertical positions of the solid lines being adjusted to make the solid and dotted lines of the same frames integrate to the same number, as they should. The $m = 1$ curves (broken lines) look more wiggly than the $m = 2$ curves (cubic splines), although the main features are the same. The criterion for selecting the smoothing parameters was the GCV score. The GCV scores and the corresponding mean square errors (evaluated at the sampling points) of each iteration are listed in Table 1. In Table 1, the iterations numbered 0 correspond to the starting values, and the last iterations (number 5 for $m = 2$ and number 3 for $m = 1$) in the two cases correspond to the models illustrated in Figure 2 and Figure 3 respectively. It appears that the starting value procedure (Algorithm 2) worked pretty well, and the convergence of the iteration was very fast. It can also be seen that both V and MSE are smaller for $m = 2$, indicating that $m = 2$ should have been chosen as a better model according to the GCV score, and actually it *is* better according to the MSE. An execution for the $k = 4$, $n = 100$ case with 5 iterations (first column of Table 1, the amount of work is that of 6 main loop executions) ran 724 cpu seconds on

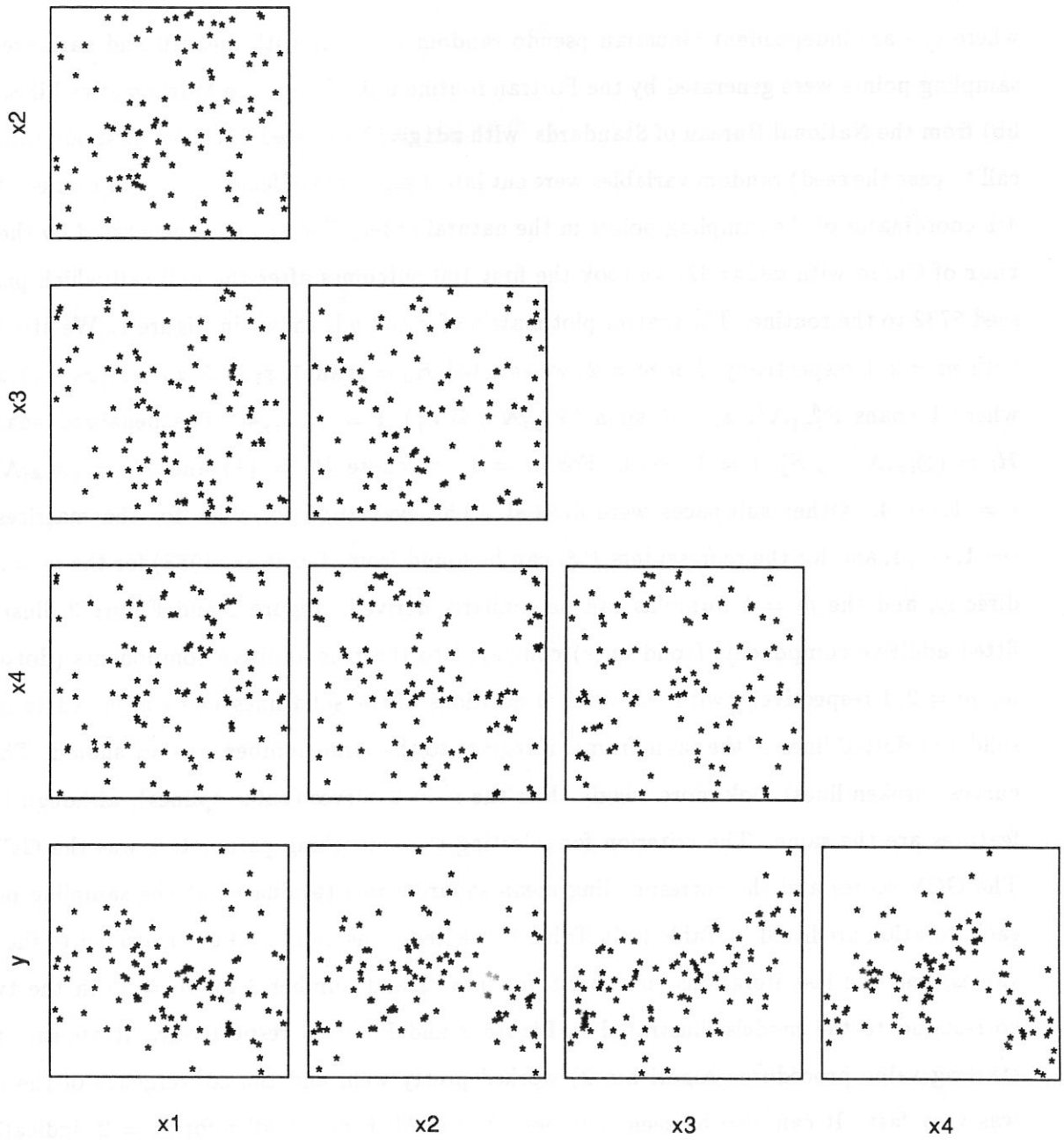

Figure 1: Scatter plot matrix: Additive model

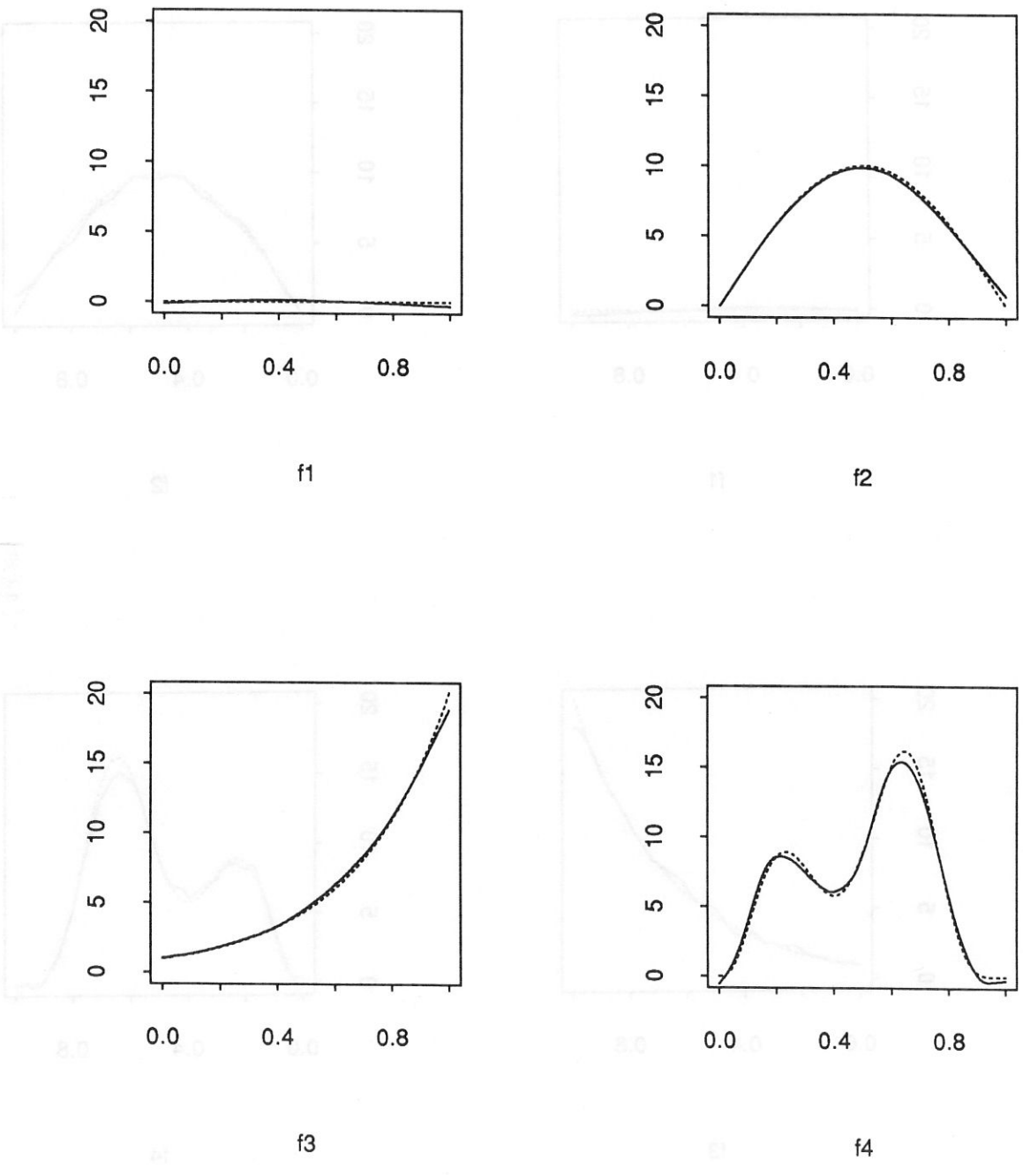

Figure 2: Additive model: $m = 2$

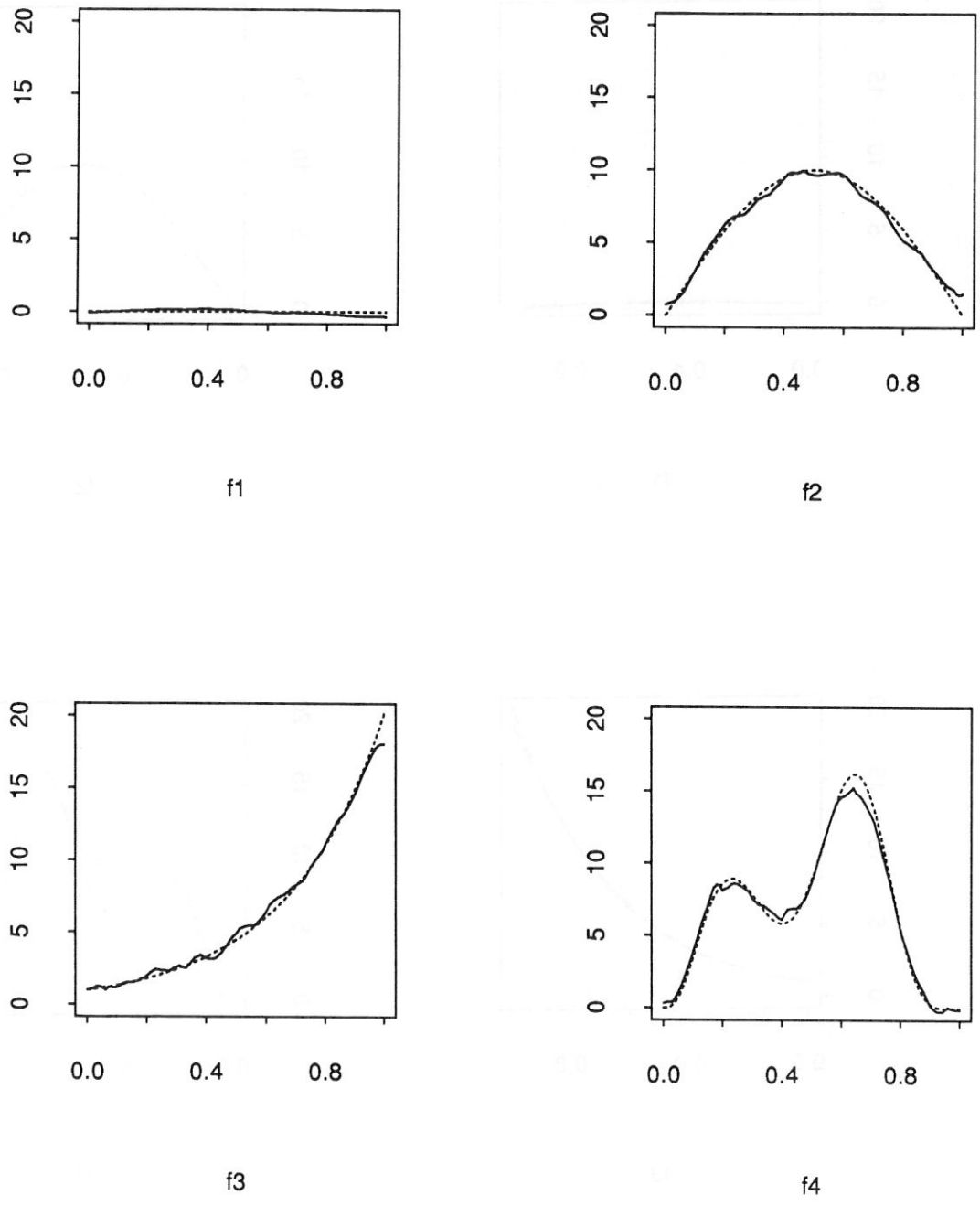

Figure 3: Additive model: $m = 1$

Table 1: Iterations of the additive models

$m = 2$			$m = 1$		
Iter.No.	V	MSE	Iter.No.	V	MSE
0	1.50409	.291176	0	1.80939	.519140
1	1.50180	.232195	1	1.75559	.462696
2	1.45412	.273181	2	1.74514	.446254
3	1.41040	.243224	3	1.74504	.444939
4	1.40893	.234954			
5	1.40893	.234726			

a Sun-3/280 (without a floating point accelerator) in the Department of Statistics, University of Wisconsin-Madison. We have also tried $n = 200$ and $n = 400$ for the same model. An $n = 200$ execution with 4 iterations ran 3628 cpu seconds on the same machine, and an $n = 400$ run with 3 iterations took 16371 cpu seconds.

The second example we wish to present is a one with a non null interaction component. This time we worked on the unit cube $[0, 1]^3$, with truth function

$$f(\mathbf{x}) = f(x_1, x_2, x_3) = 10 \sin(\pi x_2) + \exp(3x_3) + 5 \cos(2\pi(x_1 - x_2)),$$

and

$$y_j = f(\mathbf{x}_j) + \epsilon_j.$$

We used the similar procedures and same seeds to generate the sampling points and noise with the same mean and variance as in the additive model example, with $n = 400$ and $d = 3$. The scatter plot matrix of x_i's and y is omitted since it is not more informative than what we saw in Figure 1. We chose $m = 1$ for the interaction model example for its simpler formulation, though we would expect the plots of the estimation look rather wrinkled. Eliminating the 3 factor interaction, we were left with 7 subspaces of the tensor product Hilbert space $\mathcal{H} = \otimes_{l=1}^{3} \mathcal{H}^l$, namely $\mathcal{H}_0 = \{1\} = \otimes_{l=1}^{3} \mathcal{N}^l$, $\mathcal{H}_i = (\otimes_{l \neq i} \mathcal{N}^l) \otimes \mathcal{S}_1^i$, $i = 1, 2, 3$, and $\mathcal{H}_{(i)} = (\otimes_{l \neq i} \mathcal{S}_1^l) \otimes \mathcal{N}^i$, $i = 1, 2, 3$, where $\mathcal{H}_0$ was the null space and the other $k = 6$ subspaces were penalized. We know that the truth function has null projections in $\mathcal{H}_1$, $\mathcal{H}_{(2)}$, and $\mathcal{H}_{(1)}$. Projecting the estimated function to the subspaces, we can compare the truth and the estimated components individually. For the subspace $\mathcal{H}_1$, the estimated component $f_1(x_1)$ ranged between $[-.14, .12]$ on grids $x_1 = 0(.01)1$. On $\mathcal{H}_{(2)}$, the estimated component $f_{1,3}(x_1, x_3)$ ranged between $[-.38, .38]$ on the tensor product mesh $x_1, x_3 = 0(.033)1$. And on $\mathcal{H}_{(1)}$, $f_{2,3}(x_2, x_3)$

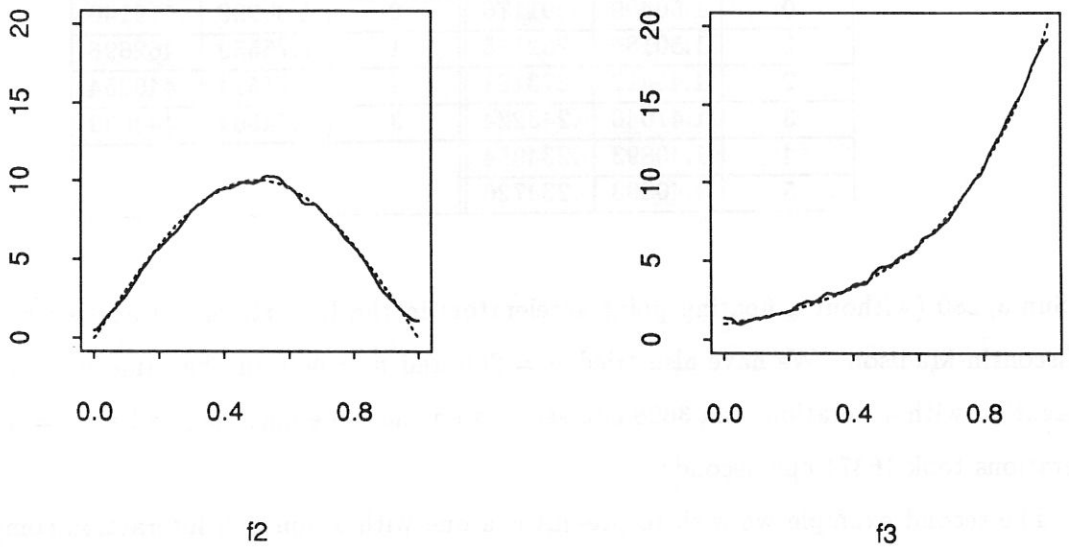

Figure 4: Interaction model: Additive components

ranged between $[-.18, .15]$ on the tensor product mesh $x_2, x_3 = 0(.033)1$. All three null component estimates were well below the standard deviation of the ϵ_j's. The other two additive components $f_2 \in \mathcal{H}_2$ and $f_3 \in \mathcal{H}_3$ are plotted in Figure 4 in the same manner as in Figure 3, and the truth and the estimation of the interaction component $f_{1,2} \in \mathcal{H}_{(3)}$ are plotted in Figure 5. The iteration sequence of this example is shown in Table 2. The iteration number 0 is again the starting value

Table 2: Iterations of the interaction model

Iter.No.	V	MSE	Iter.No.	V	MSE
0	1.40266	.376505	2	1.29839	.291201
1	1.31241	.303801	3	1.29585	.285957

model. The algorithm converged at the 3rd iteration in the sense that the optimal GCV value was within .1% of the achieved one. The total execution time was 25859 cpu seconds.

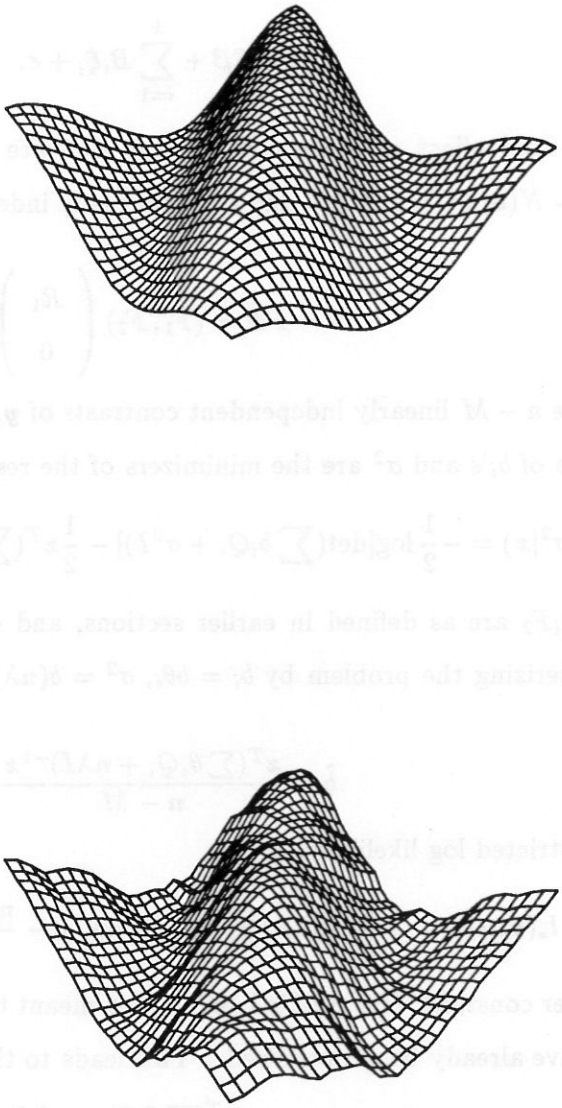

Figure 5: Interaction model: Interaction component

5 More about the algorithms

5.1 Variance component model

Consider the variance component model (Rao, 1973, Sect.4j)

$$\boldsymbol{y} = S\boldsymbol{\beta} + \sum_{i=1}^{k} B_i \boldsymbol{\xi}_i + \boldsymbol{\epsilon}, \tag{34}$$

where $S\boldsymbol{\beta}$ is the fixed effect of dimension M, the B_i's are known matrices with $B_i B_i^T = \tilde{Q}_i$, $\boldsymbol{\xi}_i \sim N(0, b_i I)$, $\boldsymbol{\epsilon} \sim N(0, \sigma^2 I)$, and $\boldsymbol{\xi}_i$'s and $\boldsymbol{\epsilon}$ are mutually independent. Let the QR-decomposition of S be

$$S = FR = (F_1, F_2) \begin{pmatrix} R_1 \\ 0 \end{pmatrix}.$$

Since $\boldsymbol{z} = F_2^T \boldsymbol{y}$ are $n - M$ linearly independent contrasts of $\boldsymbol{y}$, The restricted maximum likelihood (REML) estimates of b_i's and σ^2 are the minimizers of the restricted log likelihood

$$l_*(\boldsymbol{b}, \sigma^2 | \boldsymbol{z}) = -\frac{1}{2} \log[\det(\sum b_i Q_i + \sigma^2 I)] - \frac{1}{2} \boldsymbol{z}^T (\sum b_i Q_i + \sigma^2 I)^{-1} \boldsymbol{z} + C_1, \tag{35}$$

where $Q_i = F_2^T \tilde{Q}_i F_2$ are as defined in earlier sections, and C_1 is a constant, see also (Harville, 1977). Reparameterizing the problem by $b_i = b\theta_i$, $\sigma^2 = b(n\lambda)$, and maximizing $l_*(\cdot)$ with respect to b, we get

$$\hat{b} = \frac{\boldsymbol{z}^T (\sum \theta_i Q_i + n\lambda I)^{-1} \boldsymbol{z}}{n - M}, \tag{36}$$

and the profile restricted log likelihood is

$$l_*(\boldsymbol{\theta}, \lambda | \hat{b}) = -\frac{1}{2} \log[\det(\sum \theta_i Q_i + n\lambda I)] - \frac{n - M}{2} \log(\hat{b}) - C_2, \tag{37}$$

where C_2 is another constant. (By profile likelihood is meant the likelihood function where some of the parameters have already been optimized.) This leads to the equivalent criterion of minimizing

$$M(\boldsymbol{\theta}, \lambda | \boldsymbol{z}) = \frac{\boldsymbol{z}^T (\sum \theta_i Q_i + n\lambda I)^{-1} \boldsymbol{z} / n}{[\det(\sum \theta_i Q_i + n\lambda I)^{-1}]^{1/(n-M)}}.$$

Hence, our algorithm is directly applicable to the REML estimation of the usual variance component models.

For the maximum likelihood (ML) estimation, writing $\boldsymbol{r} = \boldsymbol{y} - S\boldsymbol{\beta}$, we have the log likelihood

$$l(\boldsymbol{b}, \sigma^2 | \boldsymbol{\beta}) = -\frac{1}{2} \log[\det(\sum b_i \tilde{Q}_i + \sigma^2 I)] - \frac{1}{2} \boldsymbol{r}^T (\sum b_i \tilde{Q}_i + \sigma^2 I)^{-1} \boldsymbol{r} + C_3, \tag{38}$$

where C_3 is a constant. After the reparameterization $b_i = b\theta_i$, $\sigma^2 = b(n\lambda)$, we can again solve for b explicitly, and lead to the minimization of a log likelihood $l(\lambda, \boldsymbol{\theta}, \boldsymbol{\beta})$. We can then compute the ML estimation by alternating parameters as

1. Maximize $l(\lambda, \boldsymbol{\beta}|\boldsymbol{\theta})$: Tridiagonalization.
2. Maximize $l(\boldsymbol{\theta}|\lambda, \boldsymbol{\beta})$: Newton update.

In step 1, we can perform an inner iteration loop to optimize λ by grid search and $\boldsymbol{\beta}$ by Gauss-Markov estimation. Specifically, we perform a tridiagonalization $\sum \theta_i \tilde{Q}_i = UTU^T$, and compute $\boldsymbol{x} = U^T\boldsymbol{y}$, $W = U^TS$. The optimal λ given $\boldsymbol{\beta}$ is the minimizer of the equivalent score

$$M(\lambda|\boldsymbol{\beta}, \boldsymbol{\theta}) = \frac{(\boldsymbol{x} - W\boldsymbol{\beta})^T(T + n\lambda I)^{-1}(\boldsymbol{x} - W\boldsymbol{\beta})/n}{\det(T + n\lambda I)^{-1/n}}.$$

And the Gauss-Markov estimator of $\boldsymbol{\beta}$ given λ can be computed through

$$\boldsymbol{\beta} = [W^T(T + n\lambda I)^{-1}W]^{-1}W^T(T + n\lambda I)^{-1}\boldsymbol{x}.$$

Overall, step 1 is a $(2/3)n^3 + O(n^2)$ operation. Step 2 is as before a $(2/3)(k-1)n^3$ operation. The corresponding ML algorithm is trivial to specify.

Various properties of the ML and REML estimations of variance component models and their relation with other estimation methods are discussed in (Harville, 1977), see also (Rao and Kleffe, 1988). Surprisingly, the simple variable transformation $\eta = \log\theta$, which results in the invariant and constraint free numerical procedures for the REML and ML estimations, is not recognized in previous works summarized by (Harville, 1977) and (Rao and Kleffe, 1988). Numerically, we should admit that our method is too general to make use of possible special structures of B_i (hence Q_i) to reduce computational load, as well as too restrictive to handle models where the variance-covariance matrix of $\boldsymbol{y}$ depends nonlinearly on the unknown parameters, though in the later case the model is no longer a variance component model literally.

5.2 Remarks

We now briefly remark on some further points of interest about the algorithms.

1. *Methodology*: Our iterative method is different from standard derivative based optimization methods. It does not operate on all parameters simultaneously. Instead, it cycles between the

two sets of parameters ($\boldsymbol{\theta}$ and λ), conditioning one on the latest version of the other, and uses different strategies for updating the two sets of parameters. Loosely speaking, each iteration targets on the profile score with respect to $\boldsymbol{\theta}$ where λ is being optimized, though the updates of $\boldsymbol{\theta}$ are not based on the derivatives of the profile score function since they are not available. Hence the method may be considered as a hybrid method adapted from the Newton methods on either the profile score or the score itself. In general, the idea of alternating parameters illustrated in this article may prove useful in tackling other statistical computing problems.

2. *Invariance*: The invariance property we are concerned about in this article is the invariance of the algorithm. It is different from the invariance properties encountered in most of the statistical literature which enforce the invariance of the end results of statistical procedures. In contrast, our concern is to enforce the invariance of numerical iteration sequences which we hope will converge to the invariantly defined end results.

3. *Numerical efficiency*: Our iterative algorithm is very expensive for large n, but it is believed to be the most efficient for the problem. The algorithm is efficient in the sense that its operational counts, $(2/3)kn^3 + O(kn^2)$ flops per iteration, is pretty much the least people can reasonably expect, given the fact that the single evaluation of the profile score is a $(2/3)n^3 + O(n^2)$ flop operation, which is itself the best available in general smoothing spline settings, see (Gu et al., 1988). Some Monte Carlo approximations to the derivatives (read the trace terms in (15), (17), (21), and (23)), which require only $O(n^2)$ extra operations beyond the profile score evaluation, have also been tried. They are eventually discarded for the reasons that the derivative based methods are very sensitive to the errors in the derivative evaluations (Gill et al., 1981), and that the approximations will lose most or all significant digits due to cancellation as the optimum is being approached.

4. *Other applications and the starting values*: It is obvious that our iterative algorithm applies to all settings which result in minimization problems with similar form as formulas (10) or (11). However, our starting value procedures are derived from the setting of (6), hence may not be appropriate for other settings such as the variance component models.

5. *Local minimum and saddle point*: The iterative algorithm will stop at any stationary point of the score function it first reaches, it may also halt at very flat regions far away from any

minimum, as most optimization methods will do. The existence of a unique minimum and the convergence to the unique minimum in various settings are beyond the scope of this article.

Acknowledgement

The original idea of the starting value procedure Algorithm 2 was inspired by a conversation with Zehua Chen to whom we owe thanks. We also wish to acknowledge the effort of our Computer Committee in the Department of Statistics, University of Wisconsin-Madison, led by Doug Bates, for maintaining a very comfortable computing environment in the department.

References

Aronszajn, N., Theory of reproducing kernels, *Trans. Amer. Math. Soc.*, 337 – 404, 1950.

Barry, D., *Nonparametric Bayesian Regression,* PhD thesis, Yale University, 1983.

Barry, D., Nonparametric Bayesian regression, *Ann. Statist.*, 934 – 953, 1986.

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

Dongarra, J., C. Moler, J. Bunch, and G. Stewart, *LINPACK User's Guide,* SIAM, Philadelphia, 1979.

Gill, P., W. Murray, and M. Wright, *Practical Optimization,* Academic Press, 1981.

Golub, G. and C. Van Loan, *Matrix Computation,* The Johns Hopkins University Press, Baltimore, 1983.

Gu, C., *RKPACK – A general purpose minipackage for spline modeling,* Technical Report 832, Department of Statistics, University of Wisconsin, Madison, 1988, under revision.

Gu, C., D. Bates, Z. Chen, and G. Wahba, *The computation of GCV functions through Householder tridiagonalization with application to the fitting of interaction spline models,* Technical Report 823, Department of Statistics, University of Wisconsin, Madison, 1988, tentatively accepted by *SIAM J. Matrix Anal. Applic.*

Harville, D. A., Maximum likelihood approaches to variance component estimation and to related problems, *J. Amer. Statist. Assoc., 72,* 320 – 340, 1977.

Kimeldorf, G. and G. Wahba, Some results on Tchebycheffian spline functions, *J. Math. Anal. Applic., 33,* 82–85, 1971.

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

O'Sullivan, F., A statistical perspective on ill-posed inverse problems, *Statist. Sci.,* 502 – 527, 1986.

Rao, C. R., *Linear Statistical Inference and Its Applications,* Wiley, 1973.

Rao, C. R. and J. Kleffe, *Estimation of variance components and applications,* North-Holland, 1988.

Wahba, G., Improper priors, spline smoothing and the problem of guarding against model errors in regression, *J. R. Statist. Soc. B, 40,* 364 – 372, 1978.

Wahba, G., Surface fitting with scattered noisy data on enclidean d-space and on the sphere, *Rocky Mountain J. Math., 14,* 281 – 299, 1984.

Wahba, G., A comparison of GCV and GML for choosing the smoothing parameter in the generalized spline smoothing problem, *Ann. Statist., 13,* 1378 – 1402, 1985.

Wahba, G., Partial and interaction splines for the semiparametric estimation of functions of several variables, in *Computer Science and Statistics: Proceedings of the 18th Symposium on the interface,* edited by T. J. Boardman, pp. 75 – 80, Amer. Statist. Assoc., Washington, D.C., 1986.

Wahba, G. and Y. Wang, *When Is the Optimal Regularization Parameter Insensitive to the Choice of the Loss Function,* Technical Report 809, Department of Statistics, University of Wisconsin, Madison, 1987.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) Technical Report No. 847	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION University of Wisconsin-Madison 6b. OFFICE SYMBOL (*If applicable*)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (*City, State, and ZIP Code*) Department of Statistics University of Wisconsin-Madison 1210 W. Dayton St., Madison, WI 53706	7b. ADDRESS (*City, State, and ZIP Code*)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION AFOSR/PKZ 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*) Building 410 Bolling AFB, Washington, D.C. 20332-6448	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. 2304/A5 / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (*Include Security Classification*)
MINIMIZING GCV/GML SCORES WITH MULTIPLE SMOOTHING PARAMETERS VIA THE NEWTON METHOD

12. PERSONAL AUTHOR(S)
Chong Gu and Grace Wahba

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Scientific Interim	FROM _____ TO _____	December 1988	24

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

see reverse side

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (*Include Area Code*) / 22c. OFFICE SYMBOL

Abstract

The (modified) Newton method (Gill et al., 1981) is adapted to optimize GCV/GML scores with multiple smoothing parameters. The main concerns in solving the optimization problem are the speed and the reliability of the algorithm, as well as the invariance of the algorithm under transformations under which the problem itself is invariant. The proposed algorithm is believed to be the most efficient for the problem, though it is still rather expensive for large data sets, since its operational counts are $(2/3)kn^3 + O(n^2)$*, with* k *the number of smoothing parameters and* n *the number of observations. Sensible procedures for computing good starting values are also proposed, which should help in keeping the execution load to the minimum possible. The algorithm is implemented in the RKPACK (Gu, 1988) and illustrated by examples of fitting additive and interaction spline models. It is noted that the algorithm can also be applied to the ML and REML estimation of the variance component models.*

Key words and phrases: *additive/interaction spline models, gradient, Hessian, invariance, Newton method, smoothing parameters, starting values.*