Statistica Sinica **6**(1996), 675-692

A GENERALIZED APPROXIMATE CROSS VALIDATION FOR SMOOTHING SPLINES WITH NON-GAUSSIAN DATA

Dong Xiang and Grace Wahba

University of Wisconsin- Madison

Abstract: In this paper, we propose a Generalized Approximate Cross Validation (GACV) function for estimating the smoothing parameter in the penalized log likelihood regression problem with non-Gaussian data. This GACV is obtained by, first, obtaining an approximation to the leaving-out-one function based on the negative log likelihood, and then, in a step reminiscent of that used to get from leaving-out-one cross validation to GCV in the Gaussian case, we replace diagonal elements of certain matrices by $1/n$ times the trace. A numerical simulation with Bernoulli data is used to compare the smoothing parameter λ chosen by this approximation procedure with the λ chosen from the two most often used algorithms based on the generalized cross validation procedure (O'Sullivan et al. (1986), Gu (1990, 1992)). In the examples here, the GACV estimate produces a better fit of the truth in term of minimizing the Kullback-Leibler distance. Figures suggest that the GACV curve may be an approximately unbiased estimate of the Kullback-Leibler distance in the Bernoulli data case; however, a theoretical proof is yet to be found.

Key words and phrases: Generalized Approximate Cross Validation, generalized cross validation, Kullback-Leibler distance, penalized likelihood regression, smoothing spline.

1. Introduction

We are concerned with the problem of the adaptive choice of the smoothing parameter in penalized log likelihood smoothing spline models for nonparametric regression with non-Gaussian data from an exponential family. We suppose that $y_i, i = 1, \ldots, n$ are independent observations from an exponential family with density of the form

$$f(y_i, \eta(x_i), \phi) = \exp\{(y_i\eta(x_i) - b(\eta(x_i)))/a(\phi) + c(y_i, \phi)\}, \tag{1.1}$$

where a, b and c are given, with b a strictly convex function of η on any bounded set, the x_i are vectors of covariates, ϕ is a nuisance parameter, and $\eta(x_i)$ is the so-called canonical parameter. The goal is to estimate $\eta(\cdot)$. For the purposes of exposition, we assume that x_i is on the real line, but our arguments extend to more general domains for x. A wide variety of distributions can be put in the form of (1.1) (see McCullagh and Nelder (1989)). In the particular case of Bernoulli

data, which we will study by Monte Carlo methods, $a(\phi) = 1, b(\eta) = \log(1 + e^{\eta})$, $c(y, \phi) = 0$, and y_i is 1 or 0 with probability $p_{\eta(x_i)} = e^{\eta(x_i)}/(1 + e^{\eta(x_i)})$. The Bernoulli case is of particular interest because of its applicability in risk factor estimation.

In the usual parametric GLIM models, $\eta(\cdot)$ is assumed to be of parametric form, and then maximum likelihood methods may be used to estimate and assess the fitted models. A variety of approaches have been proposed to allow for more flexibility than that inherent in simple parametric models. We will not review the general literature, other than to note that regression splines have been used for this purpose by, for example, Friedman (1991), Stone (1994) and others. O'Sullivan (1983), O'Sullivan, Yandell and Raynor (1986), Gu (1990), Wahba (1990) and references cited there, and others allow $\eta(\cdot)$ to take on a more flexible form by assuming that $\eta(\cdot)$ is an element of some (reproducing kernel Hilbert) space $\mathcal{H}$ of smooth functions, and estimating $\eta(\cdot)$ by minimizing a penalized log likelihood. Assuming that $a(\phi) = 1$ (or, is absorbed into λ below), define $l(y_i, \eta(x_i))$ by

$$l(y_i, \eta(x_i)) = y_i\eta(x_i) - b(\eta(x_i)).$$

The smoothing spline (or penalized log likelihood) estimate $\eta_\lambda(\cdot)$ of $\eta(\cdot)$ is the minimizer in $\mathcal{H}$ of

$$-\sum_{i=1}^{n} l(y_i, \eta(x_i)) + \frac{n\lambda}{2} J(\eta), \tag{1.2}$$

where the smoothing parameter $\lambda \geq 0$ balances the tradeoff between minimizing the negative log likelihood function

$$L = -\sum_{i=1}^{n} l(y_i, \eta(x_i))$$

and the "smoothness" $J(\eta)$. Here J is a quadratic penalty functional defined on $\mathcal{H}$. Since $\mathcal{H}$ is infinite dimensional the log likelihood may be maximized by interpolating the data, in the Bernoulli case for example resulting in $p_{\eta(x_i)} \approx y_i$. If $J^{1/2}(\cdot)$ is a norm in $\mathcal{H}$ or a seminorm in $\mathcal{H}$ with low dimensional null space (the "parametric part") satisfying some conditions, then it is well known that η_λ, the minimizer of (1.2), is in a known n-dimensional subspace $\mathcal{H}_n$ in $\mathcal{H}$ with basis functions that are known functions of the reproducing kernel for $\mathcal{H}$ and a basis for the null space of J. See Wahba (1990), O'Sullivan (1983), Kimeldorf and Wahba (1971), and below. For the purposes of discussing our estimate for λ, we assume that (1.2) will be minimized numerically in some $N \leq n$ dimensional space $\mathcal{H}_B$, that is, $\eta_\lambda(\cdot) = \sum_{j=1}^{N} \theta_j B_j(\cdot)$, where the B_j are suitable basis functions which may span $\mathcal{H}_n$, or may constitute a convenient, sufficiently rich, (linearly

independent) approximation to a spanning set. See Wahba (1990), Chapter 7, and references cited there.

Given λ, the computational problem is then to find $\theta = (\theta_1, \ldots, \theta_N)^T$ to minimize

$$I_\lambda = -\sum_{i=1}^{n} l(y_i, \eta_i(\theta)) + \frac{n\lambda}{2}\theta^T \Sigma_\theta \theta, \tag{1.3}$$

where $\eta_i(\theta) = \sum_{j=1}^{N} \theta_j B_j(x_i)$ and Σ_θ is defined by $\theta^T \Sigma_\theta \theta = J(\sum_{j=1}^{N} \theta_j B_j)$.

Letting $l_i(\cdot) = l(y_i, \cdot)$ and using the fact that all $l_i(\cdot)$ are strictly concave with respect to "$\cdot$", we may compute θ via a Newton iteration. Define $w_i = -d^2 l_i / d\eta_i^2$, $u_i = -dl_i/d\eta_i$. Each iteration for θ is equivalent to finding θ to minimize

$$\min_\theta \frac{1}{n} \sum \tilde{w}_i (\tilde{y}_i - \eta_i(\theta))^2 + \lambda \theta^T \Sigma_\theta \theta, \tag{1.4}$$

where $\tilde{y}_i = \tilde{\eta}_i - \tilde{u}_i/\tilde{w}_i$ and $\tilde{\eta}_i$, $\tilde{u}_i$, $\tilde{w}_i$ are the values of η_i, u_i and w_i based on the last iteration. The $\tilde{y}_i$ will be called the pseudo data here. This problem will have a unique minimizer provided $\Sigma_\theta \theta = 0$ and $\eta_i(\theta) = 0, i = 1, \ldots, n \Rightarrow \theta = 0$. (See O'Sullivan et al. (1986), Gu (1990).) For reference below, recall that by the properties of the exponential family, if $\eta(x_i)$ is the true canonical parameter evaluated at x_i, then $Ey_i = u_i$, and $\mathrm{Var}\,(y_i) = w_i$.

If $\mathcal{H}_B = \mathcal{H}_n$, or $\mathcal{H}_B$ is sufficiently large, then a sufficiently small λ allows the η_i to effectively interpolate the data while a sufficiently large λ forces the estimate to the null space of $J(\cdot)$ in $\mathcal{H}_B$.

With respect to the choice of λ, in the case of Gaussian data with unknown variance, Generalized Cross Validation (GCV) was proposed by Craven and Wahba (1979) and its properties have been extensively studied, see, for example Li (1986). In the Gaussian case with known variance, an unbiased risk estimate based on Mallows C_L was also proposed in Craven and Wahba (1979). In the GLIM context, O'Sullivan et al. (1986) adapted GCV to the non Gaussian case by considering the quadratic approximation to the negative log likelihood available at the final stage of their Newton iteration for θ. The GCV score they proposed is

$$V_1(\lambda) = \frac{\frac{1}{n}\|\hat{W}^{-1/2}(Y - \hat{u})\|^2}{[\frac{1}{n}\,\mathrm{tr}(I - \hat{A}(\lambda))]^2}, \tag{1.5}$$

where $Y = (y_1, \ldots, y_n)^T, \hat{u} = (\hat{u_1}, \ldots, \hat{u_n})^T, \hat{W} = \mathrm{diag}(\hat{w_1}, \ldots, \hat{w_n}), \hat{A}(\lambda)$ is the influence matrix relating $\hat{u}$ to Y, and the "$\hat{}$" indicates that these quantities are evaluated at the final step of the Newton iteration for θ, based on the quadratic approximation available then. It was suggested in Yandell (1986) to evaluate the GCV score as the iteration proceeded. Gu (1992) proposed a similar GCV score

$$V(\lambda|\tilde{y}) = \frac{\frac{1}{n}\|(I - \tilde{A}(\lambda))\tilde{W}^{1/2}\tilde{y}\|^2}{[\frac{1}{n}\,\mathrm{tr}(I - \tilde{A}(\lambda))]^2}, \tag{1.6}$$

where $\tilde{y} = (\tilde{y}_1, \ldots, \tilde{y}_n)^T$. Here the matrix $\tilde{A}(\lambda)$ satisfies $(\tilde{w}_1^{1/2}\tilde{\eta}_\lambda(x_1), \ldots, \tilde{w}_1^{1/2}\tilde{\eta}_\lambda(x_n))^T = A(\lambda)(\tilde{w}_1^{1/2}\tilde{y}_1, \ldots, \tilde{w}_1^{1/2}\tilde{y}_n)^T$, and $\tilde{\eta}_\lambda = \sum_{j=1}^N \tilde{\theta}_j B_j$ where $\tilde{\theta}$ is the minimizer of (1.4), $\tilde{W} = \mathrm{diag}(\tilde{w_1}, \ldots, \tilde{w_n})$ and the "˜" means that these quantities are evaluated at the iteration indexed by "˜" in (1.4). (To see the relation between these two scores, note that $du_i/d\eta_i = w_i$). Since $\tilde{A}(\lambda)$ and $\tilde{W}$ vary with the iteration, a decision must be made as to how to evaluate V. Gu (1992), by simulation studies and a theoretical argument, demonstrated that it was preferable to update λ at each iteration by minimizing $V(\lambda)$ (called Algorithm 2), as opposed to iterating to convergence and then evaluating and minimizing $V(\lambda)$ (called Algorithm 1, Algorithm 1 is given in Wahba (1990), but is not generally recommended), see Gu (1992).

In the case of Bernoulli data, there is no unknown variance or nuisance parameter. Using this fact, Gu (1992) gave a criteria similar to the unbiased risk (UBR) estimate in Craven and Wahba (1979) for Gaussian data for choosing λ, which is

$$U(\lambda|\tilde{y}) = \frac{1}{n}\|(I - \tilde{A}(\lambda))\tilde{W}^{1/2}\tilde{y}\|^2 + \frac{2}{n}\operatorname{tr}\tilde{A}(\lambda). \tag{1.7}$$

He believed that (1.7) is a proxy for the symmetrized Kullback-Leibler distance between $\eta_\lambda(\cdot)$, and the true $\eta(\cdot)$, summed over the x_i, and demonstrated via some simulations, that the U criteria, computed via Algorithm 2, gave more favorable results than V (also computed via Algorithm 2).

Algorithms for the estimation of multiple smoothing parameters via an Algorithm 2 iteration of U have been developed, (Wang (1995)) based on RKPACK (Gu (1989)) and successfully used in data analysis (Wahba et al. (1994a,b, 1995), Wang (1994)).

Although it appears that the Algorithm 2 computation using U generally converges, it is not guaranteed to do so, since changing λ along the iteration also changes the optimization problem. From a theoretical point of view, given that the algorithm converges, the goal function that is being minimized is not explicitly known, and so it is hard to analyze theoretically.

These considerations, as well as the widely discussed proposal of Moody (1991) in the neural net literature concerning a possible general form for an explicitly defined goal function, spurred our search for an explicit, computable, unbiased-risk-like proxy for the Kullback-Leibler distance between $\eta_\lambda(\cdot)$ and the true $\eta(\cdot)$.

One approach is to attempt to obtain directly an unbiased estimate for the Kullback-Leibler distance (or some comparative loss function) between the spline fit $\eta_\lambda(\cdot)$ for a particular λ and the true η. Suppose $\eta_\lambda(\cdot)$ is the estimate of η. The Kullback-Leibler distance $KL(\eta, \eta_\lambda)$ is defined by

$$KL(\eta, \eta_\lambda) = \frac{1}{n}\sum_{i=1}^n E_\eta \log\left(\frac{f(y_i, \eta(x_i))}{f(y_i, \eta_\lambda(x_i))}\right), \tag{1.8}$$

where E_η denotes expectation under η, and the comparative KL loss $CKL(\lambda)$, defined by

$$\begin{aligned} CKL(\lambda) &= KL(\eta, \eta_\lambda) - \frac{1}{n}\sum_{i=1}^{n}[-E_\eta y_i \eta(x_i) + b(\eta(x_i))] \\ &\equiv \frac{1}{n}\sum_{i=1}^{n}[-E_\eta y_i \eta_\lambda(x_i) + b(\eta_\lambda(x_i))] \end{aligned} \tag{1.9}$$

differs from the Kullback-Leibler distance by a quantity which does not depend on λ.

Wong (1992) showed that for y_i having a Poisson distribution ($b(\eta) = e^\eta, E_\eta y_i = e^{\eta(x_i)}$), a unique unbiased estimator for $CKL(\lambda)$ is

$$\frac{1}{n}\sum_{i=1}^{n}[-y_i \eta_\lambda^i(x_i) + e^{\eta_\lambda(x_i)}], \tag{1.10}$$

where η_λ^i is the smoothing spline fit (that is, the minimizer of (1.3)) with respect to data $(y_1, \ldots, y_{i-1}, y_i - 1, y_{i+1}, \ldots, y_n)$. Wong's estimate is very elegant; however, it is computationally expensive, requiring n solutions of the variational problem of (1.3) to evaluate (1.10) for each λ.

Wong also obtained an exact unbiased risk estimate for y from a gamma distribution with known shape parameter and unknown scale parameter. The unbiased estimate for the Gaussian case with known variance has been referred to already. However, in general, it is not straightforward to obtain an exactly unbiased estimates of the Kullback-Leibler distance or other loss functions. In the case y_i is Binomial $(m_i, p_\eta(x_i))$, Wong proved that when η_λ, considered as a function of y_i, is a polynomial of degree greater than $m_i - 1$, there does not exist an unbiased estimator for the mean square error. In particular, for $m_i = 1$ (Bernoulli data), there does not exist an unbiased estimate for the mean square error loss function and it is evident that the same techniques can be used to show that there also does not exist an exactly unbiased estimate for $CKL(\lambda)$. Thus we can only have approximately unbiased estimates. This, no doubt, explains why smoothing parameter selection with Bernoulli data has resisted a final, definitive answer so far.

In this paper, we apply first leaving-out-one cross validation to the likelihood function, which amounts to a comparative KL loss function. Since using the exact cross validation in this case is not computationally feasible for large data sets, we use a first order approximation for the cross validation of the likelihood function and get an approximate leaving-out-one cross validation function. Then, in a step reminiscent of the step in Craven and Wahba (1979) which gets to GCV from leaving-out-one cross-validation, we replace diagonal entries from certain

matrices with their averages. The end result is what might be considered an explicit form of GCV as opposed to iterative methods based on Algorithm 2. A small simulation study here with Bernoulli data shows that the estimate performs better in the examples tried than either V or U based on Algorithm 2. Theoretical justification for these promising numerical results remains to be found.

While this paper was being prepared, we become aware of Liu (1995). He gives a formula which approximates a leaving-out-one estimate under general circumstances, including when the estimate is a neural net. His formula is one of the steps in our derivation. For completeness, we have left in our derivation, but will note, which step may also be found in Liu. We remark that the arguments here also apply to a neural net estimate with weight penalties, but details are omitted.

2. Generalized Approximate Cross Validation Function

Define the ordinary, or leaving-out-one cross validation function $CV(\lambda)$,

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}[-y_i\eta_\lambda^{(-i)}(x_i) + b(\eta_\lambda(x_i))], \tag{2.1}$$

where $\eta_\lambda^{(-i)}(\cdot)$ is the minimizer of (1.2) with the ith data point omitted. $CV(\lambda)$ can be expected to be at least roughly unbiased for $CKL(\lambda)$ of (1.9) if η is "smooth" and the data are dense. For any fixed λ, in order to evaluate $CV(\lambda)$, we have to get n leaving-out-one estimates $\eta_\lambda^{(-i)}(x_i)$ $i = 1, \ldots, n$. Cox and Chang (1990) used an iterated state space algorithm to calculate the $CV(\lambda)$ function. But their algorithm can only be applied to one covariate. In general, it will be very expensive to compute $\eta_\lambda^{(-i)}(x_i)$. Using $CV(\lambda)$ is almost infeasible for large data sets. We introduce an approximation for $CV(\lambda)$ via several first order Taylor series expansions.

From (2.1), we have

$$\begin{aligned}
CV(\lambda) &= \frac{1}{n}\sum[-y_i\eta_\lambda^{(-i)}(x_i) + b(\eta_\lambda(x_i))] \\
&= \frac{1}{n}\sum[-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))] + y_i[\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)] \\
&= L(\lambda) + \frac{1}{n}\sum y_i[\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)] \qquad (2.2) \\
&= L(\lambda) + \frac{1}{n}\sum y_i\frac{(\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i))}{y_i - \mu_\lambda^{(-i)}(x_i)}(y_i - \mu_\lambda^{(-i)}(x_i)) \\
&= L(\lambda) + \frac{1}{n}\sum y_i\frac{(\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i))}{y_i - \mu_\lambda^{(-i)}(x_i)}\frac{(y_i - \mu_\lambda(x_i))}{1 - \frac{\mu_\lambda(x_i) - \mu_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)}}\ .
\end{aligned}$$

Using $\mu_\lambda(x_i) = b'(\eta_\lambda(x_i))$ gives

$$\frac{\mu_\lambda(x_i) - \mu_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)} = \frac{b'(\eta_\lambda(x_i)) - b'(\eta_\lambda^{(-i)}(x_i))}{y_i - \mu_\lambda^{(-i)}(x_i)} \approx b''(\eta_\lambda(x_i)) \frac{\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)} .$$

Therefore, $CV(\lambda)$ can be approximated by

$$\begin{aligned} CV(\lambda) &\approx L(\lambda) + \frac{1}{n}\sum_{i=1}^{n} y_i \frac{(\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i))}{y_i - \mu_\lambda^{(-i)}(x_i)} \frac{y_i - \mu_\lambda(x_i)}{1 - b''(\eta_\lambda(x_i)) \frac{\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)}} \\ &= L(\lambda) + \frac{1}{n}\sum_{i=1}^{n} \frac{y_i(y_i - \mu_\lambda(x_i))}{\frac{y_i - \mu_\lambda^{(-i)}(x_i)}{\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)} - b''(\eta_\lambda(x_i))} . \end{aligned} \tag{2.3}$$

To avoid the calculation of

$$\frac{\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)} \tag{2.4}$$

explicitly in (2.3), we develop an approximation for this ratio. Before obtaining an approximation for (2.4), we need to generalize the leaving-out-one lemma of Craven and Wahba (1979).

Lemma 2.1. (*Leaving-out-one lemma*) *Let* $-l(y_i, \eta(x_i)) = -y_i\eta(x_i) + b(\eta(x_i))$ *and* $I_\lambda(\eta, Y) = -l(y_i, \eta(x_i)) - \sum_{j\neq i} l(y_j, \eta(x_j)) + \frac{n\lambda}{2} J(\eta)$. *Suppose* $h_\lambda(i, z, \cdot)$ *is the minimizer in* $\mathcal{H}$ *or* $\mathcal{H}_B$ *of* $I_\lambda(\eta, Z)$, *where* $Z = (y_1, \ldots, y_{i-1}, z, y_{i+1}, \ldots, y_n)^T$, *then*

$$h_\lambda(i, \mu_\lambda^{(-i)}(x_i), \cdot) = \eta_\lambda^{(-i)}(\cdot),$$

where $\eta_\lambda^{(-i)}(\cdot)$ *is the minimizer of* $-\sum_{j\neq i} l(y_j, \eta(x_j)) + \frac{n\lambda}{2} J(\eta)$, *and* $\mu_\lambda^{(-i)}(\cdot)$ *is the mean corresponding to* $\eta_\lambda^{(-i)}(\cdot)$.

Proof. See Appendix A.

What this lemma says is that replacing the ith observation y_i by $\mu_\lambda^{(-i)}(x_i)$, the minimizer of I_λ with respect to $\eta(\cdot)$ will be $\eta_\lambda^{(-i)}(\cdot)$.

For the argument below we first observe that if $\eta_\lambda(\cdot)$ is a minimizer of I_λ, it is in a certain linear space of dimension at most n, and then $J(\eta_\lambda)$ can be written as a quadratic form in its values at x_i. With some abuse of notation we will sometimes write below $J(\eta) = \eta^T\Sigma\eta$, where, in this context, we are letting $\eta = (\eta(x_1), \ldots, \eta(x_n))^T$.

Let

$$\eta_\lambda = (\eta_\lambda(x_1), \ldots, \eta_\lambda(x_n))^T \quad \text{and} \quad \eta_\lambda^{(-i)} = (\eta_\lambda^{(-i)}(x_1), \ldots, \eta_\lambda^{(-i)}(x_n))^T,$$

also,

$$Y = (y_1, \ldots, y_n)^T \quad \text{and} \quad Y^{(-i)} = (y_1, \ldots, y_{i-1}, \mu_\lambda^{(-i)}(x_i), y_{i+1}, \ldots, y_n)^T.$$

Because (η_λ, Y) and $(\eta_\lambda^{(-i)}, Y^{(-i)})$ are two local minimizers of $I_\lambda(\eta, Z)$, $\partial I_\lambda / \partial\theta$ equal zero on those two points. Thus,

$$\frac{\partial I_\lambda(\eta, Z)}{\partial \eta}(\eta_\lambda, Y) = \frac{\partial I_\lambda}{\partial \theta}\frac{\partial \theta}{\partial \eta}(\eta_\lambda, Y) = 0$$

and

$$\frac{\partial I_\lambda(\eta, Z)}{\partial \eta}(\eta_\lambda^{(-i)}, Y^{(-i)}) = \frac{\partial I_\lambda}{\partial \theta}\frac{\partial \theta}{\partial \eta}(\eta_\lambda^{(-i)}, Y^{(-i)}) = 0.$$

From

$$I_\lambda = -\sum_{j=1}^{n} l(y_j, \eta(x_j)) + \frac{\lambda n}{2}\eta^T\Sigma\eta = \sum_{j=1}^{n}[-y_i\eta(x_j) + b(\eta(x_j))] + \frac{\lambda n}{2}\eta^T\Sigma\eta,$$

the second derivative of I_λ with respect to η will be

$$\frac{\partial^2 I_\lambda}{\partial\eta(x_i)\partial\eta(x_j)} = \begin{cases} b''(\eta(x_i)) + n\lambda\sigma_{ii}, & \text{if } i = j, \\ n\lambda\sigma_{ij}, & \text{if } i \neq j, \end{cases}$$

where σ_{ij} is the ijth element of Σ.

Hence, we have

$$\frac{\partial^2 I_\lambda}{\partial\eta\partial\eta^T} = W + n\lambda\Sigma, \quad \frac{\partial^2 I_\lambda}{\partial Y\partial\eta^T} = -I,$$

where $W(\eta) = \operatorname{diag}(b''(\eta(x_1)), \ldots, b''(\eta(x_n))) = \operatorname{diag}(w_1, \ldots, w_n)$.

Using a first-order Taylor expansion to expand $(\partial I_\lambda/\partial\eta)(\eta_\lambda^{(-i)}, Y^{(-i)})$ at the point (η_λ, Y), we have the following equation:

$$\begin{aligned} 0 &= \frac{\partial I_\lambda}{\partial\eta}(\eta_\lambda^{(-i)}, Y^{(-i)}) \\ &= \frac{\partial I_\lambda}{\partial\eta}(\eta_\lambda, Y) + \frac{\partial^2 I_\lambda}{\partial\eta\partial\eta^T}(\eta_\lambda^*, Y^*)(\eta_\lambda^{(-i)} - \eta_\lambda) + \frac{\partial^2 I_\lambda}{\partial Y\partial\eta^T}(\eta_\lambda^*, Y^*)(Y^{(-i)} - Y), \end{aligned}$$

or

$$\eta_\lambda - \eta_\lambda^{(-i)} = (W(\eta_\lambda^*) + n\lambda\Sigma)^{-1}(Y - Y^{(-i)}),$$

where (η_λ^*, Y^*) is a point somewhere between (η_λ, Y) and $(\eta_\lambda^{(-i)}, Y^{(-i)})$.

Approximate $W(\eta_\lambda^*)$ by $W(\eta_\lambda)$ and note that $Y - Y^{(-i)} = (0, \ldots, 0, y_i - \mu_\lambda^{(-i)}(x_i), 0, \ldots, 0)^T$. We have

$$\begin{pmatrix} \eta_\lambda(x_1) - \eta_\lambda^{(-i)}(x_1) \\ \vdots \\ \eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i) \\ \vdots \\ \eta_\lambda(x_n) - \eta_\lambda^{(-i)}(x_n) \end{pmatrix} \simeq (W(\eta_\lambda) + n\lambda\Sigma)^{-1} \begin{pmatrix} 0 \\ \vdots \\ y_i - \mu_\lambda^{(-i)}(x_i) \\ \vdots \\ 0 \end{pmatrix}, \tag{2.5}$$

i.e.

$$\frac{\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)}{y_i - \mu_\lambda^{(-i)}(x_i)} \simeq h_{ii}, \tag{2.6}$$

where $H = [W(\eta_\lambda) + n\lambda\Sigma]^{-1}$ is the inverse Hessian of $I_\lambda(\eta, Y)$ with respect to η and h_{ii} is the ith diagonal element of H. The derivation of (2.5) follows that of Liu's Equation (6).

Combining (2.3) and (2.6), we have an Approximate Cross Validation function

$$ACV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))) + \frac{1}{n}\sum_{i=1}^{n}\frac{h_{ii}y_i(y_i - \mu_\lambda(x_i))}{1 - h_{ii}b''(\eta_\lambda(x_i))}\ . \tag{2.7}$$

In (2.7), replacing h_{ii} by $\mathrm{tr}(H)/n$ and replacing $h_{ii}b''(\eta_\lambda(x_i)$ by $\mathrm{tr}(W^{1/2}H W^{1/2})/n$, we have a generalized form for the approximate cross validation

$$GACV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))) + \frac{\mathrm{tr}(H)}{n}\frac{\sum_{i=1}^{n} y_i(y_i - \mu_\lambda(x_i))}{n - \mathrm{tr}(W^{1/2}HW^{1/2})}\ . \tag{2.8}$$

As an example, in the Bernoulli case, $b(\eta_\lambda(x_i)) = \log(1 + e^{\eta_\lambda(x_i)})$, $\mu_\lambda(x_i) = p_\lambda(x_i)$ and $b''(\eta_\lambda(x_i)) = p_\lambda(x_i)(1-p_\lambda(x_i))$, and $W = \mathrm{diag}(p_\lambda(x_1)(1-p_\lambda(x_1)), \ldots, p_\lambda(x_n)(1 - p_\lambda(x_n))$. Then the GACV function will be

$$GACV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + \log(1 + e^{\eta_\lambda(x_i)})) + \frac{\mathrm{tr}(H)}{n}\frac{\sum_{i=1}^{n} y_i(y_i - p_\lambda(x_i))}{n - \mathrm{tr}(W^{1/2}HW^{1/2})}\ . \tag{2.9}$$

3. Simulation Results

In this section, we are going to perform several simulations to study the GACV curve and compare the λ chosen from $GACV(\lambda), U(\lambda)$ and $V(\lambda)$.

3.1. Computation of η_λ, Σ and the GACV function

Finite representations for the (exact) minimizer of (1.2) are well known when $J(\eta)$ is a seminorm in a reproducing kernel space $\mathcal{H}$. A popular example is $J(\eta) = \int_0^1 (\eta''(x))^2 dx$. We have chosen to use the exact representation in our simulations. If $\mathcal{H}$ is decomposed into $\mathcal{H}_0 \oplus \mathcal{H}_1$, where $\mathcal{H}_0$ is the null space of J, then the (exact) minimizer of (1.2) in $\mathcal{H}$ has a representation

$$\eta_\lambda(\cdot) = \sum_{\nu=1}^{m} d_\nu \phi_\nu(\cdot) + \sum_{i=1}^{n} c_i \xi_i(\cdot), \tag{3.1}$$

where the $\{\phi_\nu\}$ span the null space of J in $\mathcal{H}$, and it is being assumed that the $n \times m$ matrix S with $i\nu$th entry $\phi_\nu(x_i)$ is of full column rank. (Otherwise the minimizer is not necessarily unique.) $\xi_i(x) = K(x, x_i)$, where $K(x, y)$ is the reproducing kernel for $\mathcal{H}_1$, and $c = (c_1, \ldots, c_n)^T$ satisfies the m conditions $S^T c = 0$. Furthermore $J(\eta_\lambda) = c^T Q c$ where Q is the $n \times n$ matrix with ijth entry $K(x_i, x_j)$. See Wahba (1990). Thus to find η_λ to minimize (1.2), we only need to find $d = (d_1, \ldots, d_m)^T$ and c to minimize

$$-\sum_{i=1}^{n} l_i\Big(\sum_{\nu=1}^{m} d_\nu \phi_\nu(x_i) + \sum_{j=1}^{n} c_j \xi_j(x_i)\Big) + \boldsymbol{c}^T Q \boldsymbol{c}. \tag{3.2}$$

In order to compute $GACV(\lambda)$ we need to find Σ satisfying $\eta_\lambda^T \Sigma \eta_\lambda = c^T Q c$. Q may not be of full rank, despite the fact that η_λ is unique. (This will happen if, for example, if the x_i are not distinct.) We have the following lemma:

Lemma 3.1. *Let Δ be any $n \times (n - m)$ matrix of orthogonal vectors whose columns are all perpendicular to the columns of S, and let $\dagger$ be the Moore-Penrose generalized inverse. Then*

$$\Sigma = \Delta(\Delta Q \Delta^T)^\dagger \Delta^T. \tag{3.3}$$

If Q is of full rank, we can write

$$\Sigma = Q^{-1} - Q^{-1} S (S^T Q^{-1} S)^{-1} S^T Q^{-1}. \tag{3.4}$$

Proof. See Appendix B.

We remark that in large problems the computation of H and especially Σ may be unstable, but we encountered no problems in our examples below with $n = 100$ nicely spaced x_i, where Q^{-1} was computed via the eigenvalue-eigenvector decomposition. (See Note Added in Proof.)

For most of our experiments we took $\mathcal{H}$ as the Sobolev space $W_2 = \{\eta : \eta, \eta' \text{ abs. cont}, \eta'' \in \mathcal{L}_2\}$ and $J(\eta) = \int_0^1 (\eta''(x))^2 dx$. In this case, $m = 2, \phi_1(x) =$

$1, \phi_2(x) = x - 1/2$ and $\xi_i(x) = K(x, x_i)$ where $K(u, v) = k_2(u)k_2(v) - k_4([u - v])$, where $n!k_n(u)$ is the nth Bernoulli polynomial and $[\tau]$ is the fractional part of τ. In one example, we assumed that η was periodic, in this case, $m = 1$, ϕ_2 is deleted from the above representation, and $K(u, v)$ becomes $-k_4([u - v])$. We have chosen to use the representation (3.1) for our simulation studies in order to use the code RKPACK, which is used as a subroutine at each step of the iteration in (1.4), although other representations are available. We defer discussion of efficient numerical methods appropriate for large data sets for a later paper.

3.2. The $GACV(\lambda)$ curve

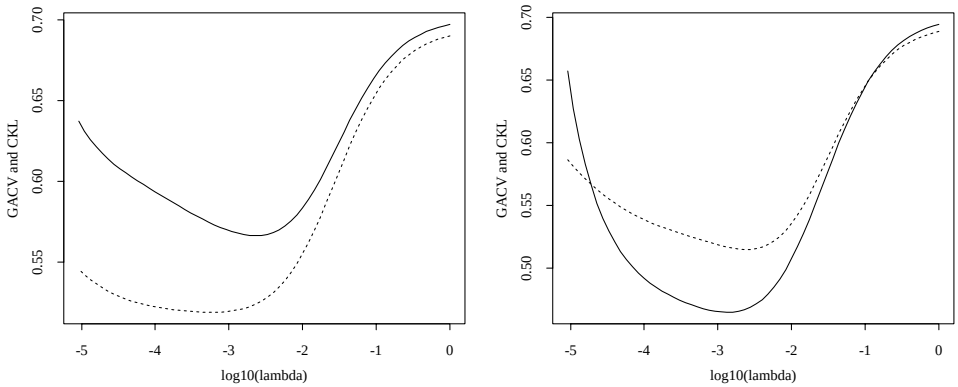

Figure 3.1. Two $GACV(\lambda)$ (solid lines) and $CKL(\lambda)$(dotted lines) curves.

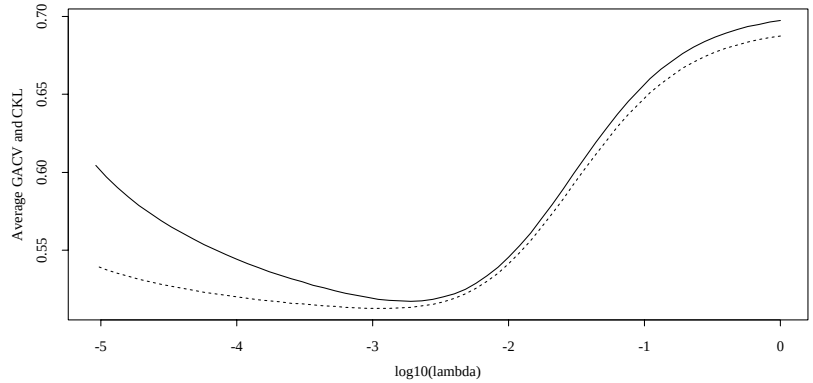

Figure 3.2. Average $GACV(\lambda)$(solid lines) and $CKL(\lambda)$ (dotted lines) curves.

Figure 3.1 contains two typical $GACV(\lambda)$ and $CKL(\lambda)$ curves from an example using logistic regression for Bernoulli data, that is y_i is 1 or 0 with $Ey_i = p(x_i)$ and $\eta(x) = \text{logit}(p(x)) = \log(p(x)/(1-p(x)))$, $b(\eta(x)) = \log(1 + e^{\eta(x)})$. In this fig-

ure $\eta(x)=2\sin(2\pi x)$ and $x_i=(i-.5)/100$, $i=1,\ldots,100$ are equally spaced from 0 to 1, and $KL(\eta,\eta_\lambda)$ may be obtained from $CKL(\lambda)$ by subtracting the constant 0.51157. The figure shows that the minima of $GACV(\lambda)$ and $CKL(\lambda)$ are very close in these two examples. Figure 3.2 gives the average of $GACV(\lambda)$ and $CKL(\lambda)$ curves over two hundred replicates of curves generated as in Figure 3.1.

3.3. Compare λ from $GACV(\lambda)$, $U(\lambda)$ and $V(\lambda)$

In this subsection, we are going to use simulations to compare the λ chosen from $GACV(\lambda)$, $U(\lambda)$ and $V(\lambda)$, with the U and V implementation via Algorithm 2.

Four different logistic or probability curves, which were used in Cox and Chang (1990), are reused in this section, they are

$$\begin{aligned}
\eta_1(x) &= 3-(5x-2.5)^2\\
\eta_2(x) &= 2\sin(10x)\\
p_3(x) &= \begin{cases} -1.6x+.9, & \text{if } x\le .5,\\ +1.6x-.7, & \text{if } x>.5,\end{cases}\\
p_4(x) &= \begin{cases} 3.5x/3, & \text{if } x\le .6,\\ .7, & \text{if } x>.6.\end{cases}
\end{aligned}$$

Also, we include a periodic function, $\eta_5(x)=2\sin(2\pi x)$, and a linear function, $\eta_6(x)=0.218-4.312x$, in the simulations. For the periodic function, we will minimize (1.2) in the space of periodic functions in W_2.

The experiments are conducted as follows: On $x_i=(i-.5)/100$, $i=1,\ldots,100$, Bernoulli data were generated according to the logit functions. Calculating η_λ by minimizing (1.2) on a grid of $\log_{10} n\lambda=-6(.08)0$, and evaluating $GACV(\lambda)$ on the same grid to find the minimizing $\hat\lambda_{GACV}$. To obtain $\operatorname{tr} H$, we use EISPACK to do the eigenvalue-eigenvector decomposition of $(W(\eta_\lambda)+n\lambda\Sigma)$ to find the eigenvalues of H, call them $\gamma_1,\ldots,\gamma_n$. $Tr(H)=\sum 1/\gamma_i$. For $\operatorname{tr}(HW)$, we have to calculate exactly H and then HW before we get $Tr(HW)$. Also from $V(\lambda)$ and $U(\lambda)$, we have λ_{GCV}, λ_{UBR} available. Then from $\eta_{GACV},\eta_{GCV},\eta_{UBR}$, we calculated three Kullback-Leibler Distances, $KL(\eta,\eta_{GACV}),KL(\eta,\eta_{GCV}),KL(\eta,\eta_{UBR})$, where $KL(\eta,\eta_\lambda)=\sum_{i=1}^n p(x_i)(\eta(x_i)-\eta_\lambda(x_i))-\log(1+e^{\eta(x_i)})+\log(1+e^{\eta_\lambda(x_i)})$. The true $p(x)$ curves of the six test functions above, and a set of data generated from each test function are plotted in Figure 3.3.

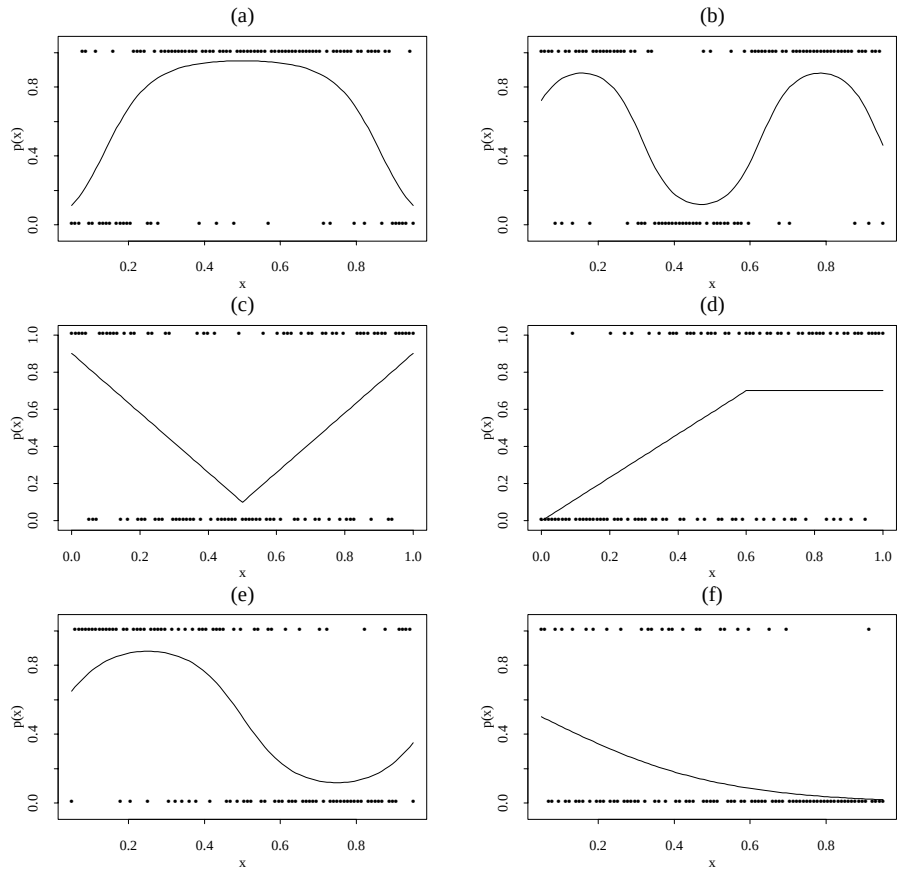

Figure 3.3. The true $p(x)$ and a set of data, for the six cases with $p(x)$ determined by by (a):η_1,(b):η_2,(c):p_3,(d):p_4,(e):η_5 and (f):η_6.

To evaluate the effectiveness of the methods, 200 sets of data for each function were generated and relative efficiencies were calculated based on

$$eff(\hat{\eta}) = \frac{\min_\lambda KL(\eta, \eta_\lambda)}{KL(\eta, \hat{\eta})}.$$

Figure 3.4 shows the boxplots of efficiency for the three methods of estimating λ. The example of Figure 3.3(b) appears to be the closest example to Gu's (1992) example, and the boxplots for V and U in Figure 3.3(b) appear to be roughly comparable to the V and U boxplots in Gu (1992), Figure 3. In all the cases we tried, the $GACV$ λ provides the best fitting among these three λ's, especially for the case (f) when the true $\eta(\cdot)$ is only a linear function. In general, the distribution of efficiencies for the GACV estimate appears to have a higher median and a shorter tail than either of its two competitors.

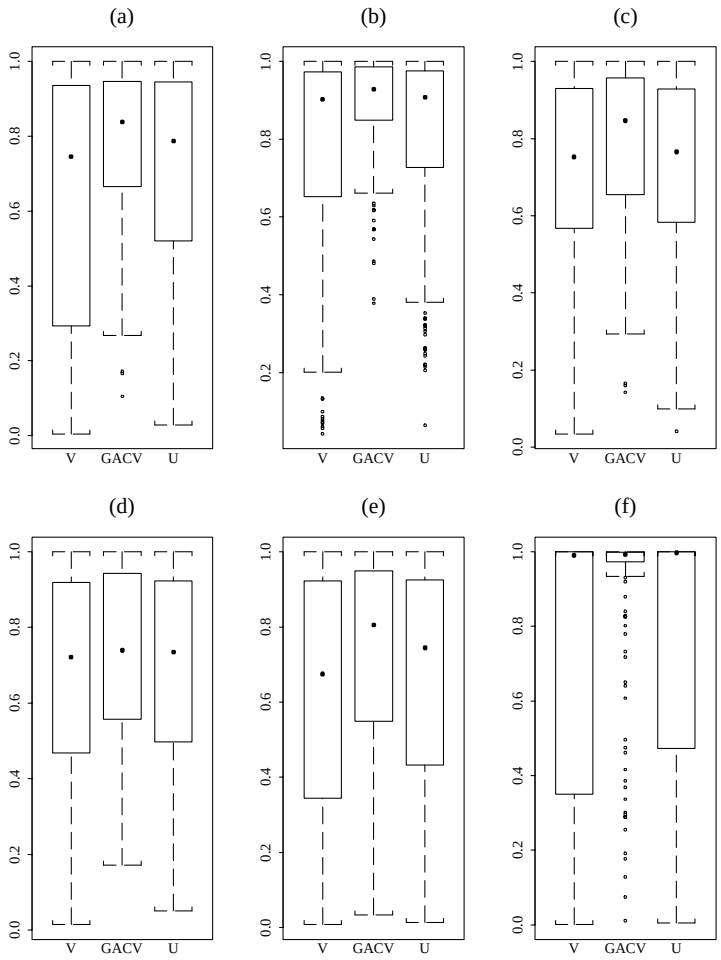

Figure 3.4. Boxplots of the efficiency for 6 different examples, for GCV (Algorithm 2), UBR (Algorithm 2) and GACV.

4. Discussion

In this paper we have proposed a proxy, $GACV(\lambda)$ for the comparative Kullback-Leibler distance $CKL(\lambda)$, by starting with a leaving-out-one proxy, approximating it by repeated use of a Taylor series expansion, and, finally, replacing individual diagonal entries in certain matrices by the average diagonal entry. The end result, the $GACV(\lambda)$, appears from simulations to be an excellent proxy for the Kullback-Leibler distance, in the sense that the minimizer of $GACV(\lambda)$ is close to the minimizer of $CKL(\lambda)$; furthermore, in the examples tried, the estimates of λ appeared superior to the popular and successful V and U estimates computed via Algorithm 2. A theoretical explanation of these results remains to be found. Also, in order for this method to be competitive with

Gu's (Algorithm 2) U for large data sets, stable numerical methods for $n \approx 1000$ must be found. We remark that the GACV can also be used in the context of choosing regularization parameters in a neural net where there is a penalty on the net weight (see Moody (1991), Liu (1995)).

We have tried other proxies starting with a leaving-out-one expression, and using different approximations at certain stages. For example, if we start with using mean square error for our cross validation function, replacing negative log likelihood by the mean square error in (2.1), the same derivation will lead us to the weighted GCV function, $\|Y - \mu_\lambda\|^2/[\mathrm{tr}(I - HW)]^2$, which is identical to $GCV(\lambda)$ for the Gaussian case if the noise are from identical normal distributions.

Consider a slightly different leaving-out-one, say

$$CV_2(\lambda) = \frac{1}{n}\sum_{i=1}^{n}[-y_i\eta_\lambda^{(-i)}(x_i) + b(\eta_\lambda^{(-i)}(x_i))];$$

then, by using the approximation $b(\eta_\lambda^{(-i)}(x_i)) - b(\eta_\lambda(x_i)) \approx -b'(\eta_\lambda(x_i))[\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)] = -\mu_\lambda(x_i)[\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)]$ we have an expression similar to (2.2), namely,

$$CV_2(\lambda) = L(\lambda) + \frac{1}{n}\sum(y_i - \mu_\lambda(x_i))[\eta_\lambda(x_i) - \eta_\lambda^{(-i)}(x_i)]. \tag{4.1}$$

The same argument as that following (2.2) results in

$$ACV_2(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\frac{h_{ii}(y_i-\mu_\lambda(x_i))^2}{1-h_{ii}b''(\eta_\lambda(x_i))} + \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i)+b(\eta_\lambda(x_i)))$$
$$= \frac{1}{n}\sum_{i=1}^{n}\frac{h_{ii}b''(\eta_\lambda(x_i))(y_i-\mu_\lambda(x_i))^2/b''(\eta_\lambda(x_i))}{1-h_{ii}b''(\eta_\lambda(x_i))} + \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i)+b(\eta_\lambda(x_i)));$$

another way to take the generalization step from the above $ACV(\lambda)$ will give us

$$\begin{aligned} GACV_2(\lambda) \;=\; & \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))) \\ & + \frac{\mathrm{tr}(W^{1/2}HW^{1/2})}{n}\,\frac{\sum_{i=1}^{n}(y_i - \mu_\lambda(x_i))^2/b''(\eta_\lambda(x_i))}{n - \mathrm{tr}(W^{1/2}HW^{1/2})}. \end{aligned} \tag{4.2}$$

In particular, for Bernoulli data,

$$\begin{aligned} GACV_2(\lambda) = & \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))) \\ & + \frac{\mathrm{tr}(W^{1/2}HW^{1/2})}{n}\,\frac{\sum_{i=1}^{n}(y_i-p_\lambda(x_i))^2/(p_\lambda(x_i)(1-p_\lambda(x_i)))}{n-\mathrm{tr}(W^{1/2}HW^{1/2})}. \end{aligned} \tag{4.3}$$

Since $(y_i - p_\lambda(x_i))^2/(p_\lambda(x_i)(1-p_\lambda(x_i)) \approx 1$, we have

$$GACV_2(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(-y_i\eta_\lambda(x_i) + b(\eta_\lambda(x_i))) + \frac{k}{n}\operatorname{tr}(W^{1/2}HW^{1/2}), \tag{4.4}$$

where $k = n/(n-\operatorname{tr}(W^{1/2}HW^{1/2}))$. This version of GACV is very similar to that proposed by Gu in (1.7), where in (1.7), the first part is an approximation of the log likelihood in (4.4). But for the examples we studied in this paper, simulation suggests that $GACV$ is better than $GACV_2$

Acknowledgement

This work is supported in part by the National Science Foundation under Grant DMS-9121003 and the National Eye Institute under Grant R01 EY09946.

Appendix

A. Proof of Lemma 2.1

First define $Y^{-i} = (y_1, \ldots, y_{i-1}, \mu_\lambda^{(-i)}(x_i), y_{i+1}, \ldots, y_n)$. Since $-l(\mu_\lambda^{(-i)}(x_i), \tau)$ $= -u_\lambda^{(-i)}(x_i)\tau + b(\tau)$, we have

$$-l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) \le -l(\mu_\lambda^{(-i)}(x_i), \eta(x_i)) \ . \tag{A.1}$$

This follows since setting

$$\frac{\partial l(\mu_\lambda^{(-i)}(x_i), \tau)}{\partial \tau} = -\mu_\lambda^{(-i)}(x_i) + b'(\tau) = 0$$

and using the fact that $b''(\tau) > 0$, implies that $l(u_\lambda^{(-i)}(x_i), \eta)$ achieves its (unique) minimum for $b'(\eta) = \mu_\lambda^{(-i)}(x_i)$. Thus for any η,

$$\begin{aligned} I_\lambda(\eta, Y^{-i}) &= -l(\mu_\lambda^{(-i)}(x_i), \eta(x_i)) - \sum_{j\neq i} l(y_j, \eta(x_j)) + n\frac{\lambda}{2}J(\eta) \\ &\ge -l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) - \sum_{j\neq i} l(y_j, \eta(x_j)) + n\frac{\lambda}{2}J(\eta) \\ &\ge -l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) - \sum_{j\neq i} l(y_j, \eta_\lambda^{(-i)}(x_j)) + n\frac{\lambda}{2}J(\eta_\lambda^{(-i)}). \end{aligned}$$

The first inequality is because of (A.1), the second inequality is due to the fact that $\eta_\lambda^{(-i)}$ is the minimizer of $-\sum_{j\neq i} l(y_j, \eta(x_j)) + n\frac{\lambda}{2}J(\eta)$. Thus we have $h_\lambda(i, \mu_\lambda^{(-i)}) = \eta_\lambda^{(-i)}$.

B. Proof of Lemma 3.1

Since $\eta(x_i) = \sum_{\nu=1}^{m} d_\nu \phi(x_i) + \sum_{j=1}^{n} c_j K(x_i, x_j)$,

$$\begin{aligned} Qc + Sd &= \eta \\ S^T c &= 0. \end{aligned} \tag{B.1}$$

Let $c = \Delta\gamma$ for some $n-m$ dimensional vector γ, where Δ is as defined in the text. This is necessary and sufficient to insure that $S^T c = 0$. Then $c^T Q c = \gamma^T \Delta^T Q \Delta \gamma$. Substituting into (B.1) gives $(\Delta^T Q \Delta)\gamma = \Delta^T \eta$. Then $c^T Q c = \gamma^T (\Delta^T Q \Delta)\gamma = \gamma^T (\Delta^T Q \Delta)(\Delta^T Q \Delta)^+ (\Delta^T Q \Delta)\gamma = \eta^T \Delta (\Delta^T Q \Delta)^+ \Delta^T \eta$. If Q is of full rank then formulas for the block inverse of a matrix gives the result.

Note Added in Proof

We have recently shown that the calculation of matrix inverses as described following (3.4) can be avoided by using the randomized trace method to estimate $\mathrm{tr}(H)$ and $\mathrm{tr}(W^{1/2} H W^{1/2})$, see Xiang, D. (1996), Model fitting and testing for non-Gaussian data with large data sets. (PhD thesis.) Technical Report 957. Dept. of Statistics, University of Wisconsin, Madison, WI.

References

Cox, D. and Chang, Y. (1990). Iterated state space algorithms and cross validation for generalized smoothing splines. Technical Report 49, University of Illinois, Dept. of Statistics, Champaign, IL.

Craven, P. and 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.

Friedman, J. (1991). Multivariate adaptive regression splines. *Ann. Statist.* **19**, 1-141.

Gu, C. (1989). RKPACK and its applications: Fitting smoothing spline models. In *Proceedings of the Statistical Computing Section*, 42-51. American Statistical Association.

Gu, C. (1990). Adaptive spline smoothing in non-Gaussian regression models. *J. Amer. Statist. Assoc.* **85**, 801-807.

Gu, C. (1992). Cross-validating non-Gaussian data. *J. Comput. Graph. Stats.* **1**, 169-179.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *J. Math. Anal. Appl.* **33**, 82-95.

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.

Liu, Y. (1995). Unbiased estimate of generalization error and model selection in neural network. *Neural Networks* **8**, 215-219.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models,* Second Edition, Chapman and Hall, London.

Moody, J. (1991). The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In *Advances in Neural Information Processing Systems 4*, (Edited by J. Moody, S. Hanson and R. Lippman), 847-854, Kaufmann, San Mateo.

O'Sullivan, F. (1983). The analysis of some penalized likelihood schemes. (PhD. Thesis) Technical Report 726, Department of Statistics, University of Wisconsin, Madeson, WI.

O'Sullivan, F., Yandell, B. and Raynor, W. (1986). Automatic smoothing of regression functions in generalized linear models. *J. Amer. Statist. Assoc.* **81**, 96-103.

Stone, C. (1994). The use of polynomial splines and their tensor products in multivariate function estimation. *Ann. Statist.* **22**, 118-184.

Wahba, G. (1990). *Spline Models for Observational Data*. SIAM. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 59.

Wahba, G., Gu, C., Wang, Y. and Chappell, R. (1995). Soft classification, a. k. a. risk estimation, via penalized log likelihood and smoothing spline analysis of variance. In *The Mathematics of Generalization*, Vol. XX, (Edited by D. Wolpert), 329-360, Santa Fe Institute Studies in the Sciences of Complexity, Proc. Addison-Wesley, Reading, MA.

Wahba, G., Wang, Y., Gu, C., Klein, R. and Klein, B. (1994a). Smoothing spline ANOVA for exponential families, with application to the Wisconsin Epidemiological Study of Diabetic Retinopathy. Technical Report 940, Department of Statistics, University of Wisconsin, Madison, WI, to appear, *Ann. Statist.* Dec. 1995.

Wahba, G., Wang, Y., Gu, C., Klein, R. and Klein, B. (1994b). Structured machine learning for "soft" classification with smoothing spline ANOVA and stacked tuning, testing and evaluation. In *Advances in Neural Information Processing Systems* 6, (Edited by J. Cowan, G. Tesauro and J. Alspector), 415-422, Morgan Kauffman.

Wang, Y. (1994). Smoothing spline analysis of Variance of Data from Exponential Families. PhD thesis, Technical Report 928, University of Wisconsin-Madison, Madison, WI.

Wang, Y. (1995). GRKPACK: Fitting smoothing spline analysis of variance models to data from exponential families. Technical Report 942, Dept. of Statistics, University of Wisconsin, Madison, WI.

Wong, W. (1992). Estimation of the loss of an estimate. Technical Report 356. Dept. of Statistics, University of Chicago, Chicago, IL.

Yandell, B. (1986). Algorithms for nonlinear generalized cross-validation, In *Computer Science and Statistics:* 18*th Symposium on the Interface* (Edited by T. Boardman), American Statistical Association, Washington, DC.

Department of Statistics, University of Wisconsin- Madison, 1210 W. Dayton St., Madison, WI 53706. U.S.A.

(Received September 1994; accepted October 1995)