A Generalized Approximate Cross Validation for Smoothing Splines with Non-Gaussian Data

by

Dong Xiang and Grace Wahba

---

1Supported by the National Science Foundation under Grant DMS-9121003 and the National Eye Institute under Grant R01 EY09946. e-mail xiang@stat.wisc.edu, wahba@stat.wisc.edu
A Generalized Approximate Cross Validation for Smoothing Splines with Non-Gaussian Data

Dong Xiang  Grace Wahba
Department of Statistics
University of Wisconsin at Madison

September 5, 1994

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-one-out 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 $\frac{1}{n}$ times the trace. A numerical simulation with Bernoulli data is used to compare to the smoothing parameter $\lambda$ chosen by this approximation procedure with the $\lambda$ 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.

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

1 Introduction

We are concerned with the important problem of the adaptive choice of smoothing parameter in penalized log likelihood smoothing spline models for nonparametric regression with non-Gaussian data from an exponential family. We suppose that $y_i, i = 1, 2, ..., n$ are independent observations from an exponential family with density of the form

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

(1.1)
where $a, b$ and $c$ are given, with $b$ a strictly convex function of $\eta$ on any bounded set, the $x_i$ are vectors of covariates, $\phi$ 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 will 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 (1983). 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 GLM 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 more flexibility than is 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 $H$ of smooth functions, and estimating $\eta(\cdot)$ by minimizing a penalized log likelihood. Assuming that $a(\phi) = 1$ (or, is absorbed into $\lambda$ 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 $H$ of

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

(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 $H$. Since $H$ 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 $H$ or a seminorm in $H$ with low dimensional null space (the ‘parametric part’) satisfying some conditions, then it is well known that $\eta_\lambda$, the minimizer of (1.2), is in a known $n$-dimensional subspace $H_n$ in $H$ with basis functions that are known functions of the reproducing kernel for $H$ 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 $\lambda$, we will assume that (1.2) will be minimized numerically in some
\( N \leq n \) dimensional space \( \mathcal{H}_B \), that is,

\[
\eta_\lambda(\cdot) = \sum_{j=1}^{N} \theta_j B_j(\cdot),
\]

(1.3)

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 \( \lambda \), the computational problem is then to find \( \theta = (\theta_1, ..., \theta_N)^T \) to minimize

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

(1.4)

where

\[
\eta_i(\theta) = \sum_{j=1}^{N} \theta_j B_j(x_i)
\]

and \( \Sigma_\theta \) 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 \( \theta \) via a Newton iteration. Define \( w_i = -d^2 l_i / d\eta_i^2 \), \( u_i = -d l_i / d\eta_i \). Each iteration for \( \theta \) is equivalent to finding \( \theta \) to minimize

\[
\min_{\theta} \frac{1}{n} \sum w_i (\hat{\eta}_i - \eta_i(\theta))^2 + \lambda \theta^T \Sigma_\theta \theta.
\]

(1.5)

where \( \hat{\eta}_i = \hat{\eta}_i - \tilde{u}_i / \tilde{w}_i \) and \( \tilde{\eta}_i, \tilde{u}_i, \tilde{w}_i \) are the values of \( \eta_i, u_i \) and \( w_i \) based on the last iteration. The \( \hat{\eta}_i \) will be called the pseudo data here. This problem will have a unique minimizer provided \( \Sigma_\theta \theta = 0 \) and \( \eta_i(\theta) = 0, i = 1, ..., n \Rightarrow \theta = 0 \). 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 \( E y_i = u_i \), and \( var y_i = w_i \).

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

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. 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 \( \theta \). The GCV score
they proposed is
\[ V_1(\lambda) = \frac{\frac{1}{n} \|\hat{W}^{-1/2}(Y - \hat{u})\|^2}{[\frac{1}{n} \text{tr}(I - \hat{A}(\lambda))]^2}, \]  
(1.6)

where \( Y = (y_1, ..., y_n)^T, \hat{u} = (\hat{w}_1, ..., \hat{w}_n)^T, \hat{W} = \text{diag}(\hat{w}_1, ..., \hat{w}_n), \hat{A}(\lambda) \) is the influence matrix relating \( \hat{u} \) to \( Y \), and the "\( \cdot \)" indicates that these quantities are evaluated at the final step of the Newton iteration for \( \theta \), 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
\[ V(\lambda|\hat{y}) = \frac{\frac{1}{n} \|\hat{W}^{1/2}(I - \hat{A}(\lambda))\hat{W}^{1/2}\hat{y}\|^2}{[\frac{1}{n} \text{tr}(I - \hat{A}(\lambda))]^2}, \]  
(1.7)

where \( \hat{y} = (\hat{y}_1, ..., \hat{y}_n)^T \). Here the matrix \( \hat{A}(\lambda) \) satisfies \( (\hat{w}_1^{1/2} \hat{\eta}_\lambda(x_1), \ldots, \hat{w}_1^{1/2} \hat{\eta}_\lambda(x_n))^T = A(\lambda)(\hat{w}_1^{1/2} \hat{y}_1, \ldots, \hat{w}_1^{1/2} \hat{y}_n)^T \), and \( \hat{\eta}_\lambda = \sum_{j=1}^N \bar{\theta}_j B_j \) where \( \bar{\theta} \) is the minimizer of (1.5), \( \hat{W} = \text{diag}(\hat{w}_1, ..., \hat{w}_n) \) and the "\( \cdot \)" means that these quantities are evaluated at the iteration indexed by "\( \cdot \)" in (1.5). (To see the relation between these two scores, note that \( du_i/d\eta_i = w_i \). Since \( \hat{A}(\lambda) \) and \( \hat{W} \) vary with the iteration, a decision must be made as to how to evaluate \( V \). Gu compared two algorithms:

**Algorithm 1** For fixed \( \lambda \)'s on a grid, iterate on (1.5) until convergence. Compute \( V(\lambda|\hat{y}_\lambda) \) at the last iteration where \( \hat{y}_\lambda \) are the pseudo data based on \( \eta_\lambda \). Choose the \( \lambda \) minimizing \( V(\lambda|\hat{y}_\lambda) \) on the grid and return the corresponding estimate \( \eta_\lambda \).

**Algorithm 2** Iterate on (1.5). In the \( k \)th iteration choose a possibly new \( \lambda^{(k)} \) minimizing \( V(\lambda|\hat{y}_\lambda^{(k-1)}) \) where \( \hat{y}^{(k-1)} \) are the pseudo data based on the previous estimate \( \eta^{(k-1)} \). Stop when \( \lambda \) is stabilized and the iteration for this stabilized \( \lambda \) has converged. Use this \( \lambda \) to compute \( \eta_\lambda \) and return.

Algorithm 1 involves comparison of \( V \)'s based on different pseudo data; and \( V \) based on different data are not comparable (See Gu, 1992). A theoretical justification for Algorithm 1 in the non-Gaussian case has not emerged, and several authors have observed unsatisfactory performance with it \(^1\), see Gu (1992), Cox and Chang (1990). On the other hand, Gu (1992) gave an argument to support Algorithm 2. His simulation results suggest that Algorithm 2 is better than Algorithm 1 in terms of obtaining an estimate which is closer to the true function, as measured by the Kullback-Leibler distance.

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 estimate in Craven and Wahba (1979).

\(^1\)Algorithm 1 is given in Wahba (1990), but is not generally recommended.
for Gaussian data for choosing \( \lambda \), which is

\[
U(\lambda | \hat{y}) = \frac{1}{n} \| (I - \tilde{A}(\lambda))^\frac{1}{2} \hat{y} \|^2 + \frac{2}{n} \text{tr} \tilde{A}(\lambda). \tag{1.8}
\]

He believed that (1.8) 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, 1994) based on RKPACK (Gu, 1992) and successfully used in data analysis (Wahba et al (1994 a, b, c), Wang (1994)).

Although it appears that the Algorithm 2 computation using \( U \) generally converges, it is not guaranteed to do so, since changing \( \lambda \) 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 other comparative loss function) between the spline fit \( \eta_\lambda(\cdot) \) for a particular \( \lambda \) and the true \( \eta \). Suppose \( \eta_\lambda(\cdot) \) is the estimate of \( \eta \). The Kullback-Leibler distance \( KL(\eta, \eta_\lambda) \) is defined by

\[
KL(\eta, \eta_\lambda) = \sum_{i=1}^{n} E_{\eta} \log \left( \frac{f(y_i, \eta(x_i))}{f(y_i, \eta_\lambda(x_i))} \right), \tag{1.9}
\]

where \( E_{\eta} \) denotes expectation under \( \eta \).

Wong (1992) showed that for \( y_i \) having a Poisson distribution \( b(\eta) = e^\eta, E y_i = e^{\eta(x_i)} \), a unique unbiased estimator for the comparative \( KL \) loss \( KL(\eta, \eta_\lambda) - \sum (e^{\eta(x_i)} - e^{\eta_\lambda(x_i)}) \) is:

\[
\sum_{i=1}^{n} (e^{\eta_i(x_i)} - y_i \eta_i^\lambda(x_i)),
\]

where \( \eta_i^\lambda \) is the smoothing spline fit (that is, the minimizer of (1.4)) with respect to data \( (y_1, \cdots, y_{i-1}, y_i - 1, y_{i+1}, \cdots, y_n) \). Wong's estimate is very elegant, however, it is computationally expensive, requiring \( n \) solutions of the variational problem of (1.4) to evaluate the comparative loss function for each \( \lambda \).

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 exactly unbiased estimates of the Kullback-Leibler distance or other loss functions. In the case \( y_i \) is Binomial \((m_i,p_{y_i}(x_i))\), Wong proved that when \( \eta_\lambda \), 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 the Kullback-Leibler distance. 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 will 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 (1993). Liu 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 key steps in our derivation. For completeness, we have left in our derivation, but we will note which steps could have been obtained from Liu. We remark that the arguments here will 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 \( CV(\lambda) \),

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

(2.1)

where \( \eta_{\lambda}^{(i)}(\cdot) \) is the minimizer of (1.2) with the \( i \)th data point omitted. \( CV(\lambda) \) can be expected to be at least roughly unbiased if \( \eta \) is “smooth” and the data are dense. For any fixed \( \lambda \), in order
to evaluate $CV(\lambda)$, we have to get $n$ leaving-out-one estimates $\eta^{(-i)}_\lambda(x_i)$ $i = 1, \ldots, n$. 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 $\eta^{(-i)}_\lambda(x_i)$. Using $CV(\lambda)$ is almost infeasible for large data sets. We will introduce an approximation for $CV(\lambda)$ via several first order Taylor series expansions.

From (2.1), we have

$$CV(\lambda) = \frac{1}{n} \sum_i [ - b(\eta_\lambda(x_i)) ]$$

$$= \frac{1}{n} \sum_i [ - y_i \eta_\lambda(x_i) + b(\eta_\lambda(x_i)) ] + y_i \eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)$$

$$= L(\lambda) + \frac{1}{n} \sum y_i \left( \frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)} \right) \left( y_i - \mu^{(-i)}_\lambda(x_i) \right)$$

$$= L(\lambda) + \frac{1}{n} \sum y_i \left( \frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)} \right) \left( y_i - \mu^{(-i)}_\lambda(x_i) \right) \frac{1 - \frac{\mu^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)}}{1 - \frac{\mu^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)}}$$

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

$$\frac{\mu_\lambda(x_i) - \mu^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)} = \frac{b'(\eta_\lambda(x_i)) - b'(\eta^{(-i)}_\lambda(x_i))}{y_i - \mu^{(-i)}_\lambda(x_i)}$$

$$\approx b''(\eta_\lambda(x_i)) \frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)}.$$ 

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

$$CV(\lambda) \approx L(\lambda) + \frac{1}{n} \sum_i \frac{y_i \left( \frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)} \right)}{1 - \frac{\mu^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)}} \frac{y_i - \mu_\lambda(x_i)}{1 - b''(\eta_\lambda(x_i)) \frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)}}$$

$$= L(\lambda) + \frac{1}{n} \sum \left( \frac{y_i (y_i - \mu_\lambda(x_i))}{\mu_\lambda(x_i) - \mu^{(-i)}_\lambda(x_i)} - b''(\eta_\lambda(x_i)) \right). \quad (2.2)$$

To avoid the calculation of

$$\frac{\eta_\lambda(x_i) - \eta^{(-i)}_\lambda(x_i)}{y_i - \mu^{(-i)}_\lambda(x_i)} \quad (2.3)$$

explicitly in (2.2), we will develop an approximation for this ratio. Before obtaining an approximation for (2.3), 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 \( I_{\lambda}(\eta, Y) = -l(y_i, \eta(x_i)) - \sum_{j \neq i} l(y_j, \eta(x_j)) + \frac{\lambda}{2} J(\eta) \). Suppose \( h_\lambda(i, z, \cdot) \) is the minimizer in \( H \) or \( H_B \) 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 \( -\sum_{j \neq i} l(y_j, \eta(x_j)) + \frac{\lambda}{2} J(\eta) \), 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 \( i \)th observation \( y_i \) by \( \mu_\lambda^{(-i)}(x_i) \), the minimizer of \( I_{\lambda} \) 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 \( I_{\lambda} \), 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 \( (\eta_\lambda, Y) \) 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,
\[
\frac{\partial I_{\lambda}(\eta, Z)}{\partial \eta}(\eta_\lambda, Y) = \frac{\partial I_{\lambda}}{\partial \theta}(\eta_\lambda, Y) = 0
\]
and
\[
\frac{\partial I_{\lambda}(\eta, Z)}{\partial \eta}(\eta_\lambda^{(-i)}, Y^{(-i)}) = \frac{\partial I_{\lambda}}{\partial \theta}(\eta_\lambda^{(-i)}, Y^{(-i)}) = 0.
\]

From
\[
I_{\lambda} = -\sum_{j=1}^n l(y_j, \eta(x_j)) + \frac{\lambda}{2} \eta^T \sigma \eta = \sum_{j=1}^n [-y_j \eta(x_j) + b(\eta(x_j))] + \frac{\lambda}{2} \eta^T \sigma \eta,
\]
the second derivative of \( I_{\lambda} \) with respect to \( \eta \) 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 $\sigma_{ij}$ is the $ij$th element of $\Sigma$.

Hence, we have
\[
\frac{\partial^2 I_\lambda}{\partial \eta^T \partial \eta} = W + n\lambda \Sigma, \quad \frac{\partial^2 I_\lambda}{\partial Y^T \partial \eta} = -I,
\]
where $W(\eta) = \text{diag}(b'(\eta(x_1)), \ldots, b'(\eta(x_n))) = \text{diag}(w_1, \ldots, w_n)$.

Using a first-order Taylor expansion to expand $\partial I_\lambda/\partial \eta(\eta\lambda, Y^{(-i)})$ at the point $(\eta\lambda, Y)$, we have the following equation:
\[
0 = \frac{\partial I_\lambda}{\partial \eta}(\eta\lambda^{(-i)}, Y^{(-i)}) = \frac{\partial I_\lambda}{\partial \eta}(\eta\lambda, Y) + \frac{\partial^2 I_\lambda}{\partial \eta^T \partial \eta}(\eta\lambda, Y^*)(\eta\lambda^{(-i)} - \eta\lambda) + \frac{\partial^2 I_\lambda}{\partial Y^T \partial \eta}(\eta\lambda, Y^*)(Y^{(-i)} - Y),
\]
or
\[
\eta\lambda - \eta\lambda^{(-i)} = (W(\eta\lambda) + n\lambda \Sigma)^{-1} (Y - Y^{(-i)}),
\]
where $(\eta\lambda^*, Y^*)$ is a point somewhere between $(\eta\lambda, Y)$ and $(\eta\lambda^{(-i)}, Y^{(-i)})$.

Approximate $W(\eta\lambda^*)$ by $W(\eta\lambda)$ and notice that $Y - Y^{(-i)} = (0, \ldots, 0, y_i - \mu\lambda^{(-i)}(x_i), 0, \ldots, 0)^T$.

We have
\[
\begin{pmatrix}
\eta\lambda(x_1) - \eta\lambda^{(-i)}(x_1) \\
\vdots \\
\eta\lambda(x_i) - \eta\lambda^{(-i)}(x_i) \\
\vdots \\
\eta\lambda(x_n) - \eta\lambda^{(-i)}(x_n)
\end{pmatrix}
= (W(\eta\lambda) + n\lambda \Sigma)^{-1}
\begin{pmatrix}
y_i - \mu\lambda^{(-i)}(x_i) \\
\vdots \\
0
\end{pmatrix},
\]
i.e.
\[
\frac{\eta\lambda(x_i) - \eta\lambda^{(-i)}(x_i)}{y_i - \mu\lambda^{(-i)}(x_i)} = h_{ii},
\]
where $H = [W(\eta\lambda) + n\lambda \Sigma]^{-1}$ is the inverse Hessian of $I_\lambda(\eta, Y)$ with respect to $\eta$ and $h_{ii}$ is the $i$th diagonal element of $H$.

Combining (2.2) and (2.4), we have an Approximate Cross Validation function
\[
ACV(\lambda) = \frac{1}{n} \sum_{i=1}^{n} (-y_i \eta\lambda(x_i) + b(\eta\lambda(x_i))) + \frac{1}{n} \sum_{i=1}^{n} h_{ii} y_i (y_i - \mu\lambda(x_i)) / \left(1 - h_{ii} b'^i(\eta\lambda(x_i))\right).
\]

In (2.5), replacing $h_{ii}$ by $\text{tr}(H)/n$ and replacing $h_{ii} b'^i(\eta\lambda(x_i))$ by $\text{tr}(W^{1/2} H W^{1/2})/n$, we have a generalized form for the approximate cross validation
\[
GACV(\lambda) = \frac{1}{n} \sum_{i=1}^{n} (-y_i \eta\lambda(x_i) + b(\eta\lambda(x_i))) + \frac{\text{tr}(H)}{n} \frac{\sum_{i=1}^{n} y_i (y_i - \mu\lambda(x_i))}{n - \text{tr}(W^{1/2} H W^{1/2})}.
\]

\footnote{This is essentially a special case of Liu's Equation (6)}
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 \( W = \text{diag}(p_\lambda(x_1)(1 - p_\lambda(x_1)), \ldots, p_\lambda(x_n)(1 - p_\lambda(x_n))) \). Then the GACV function will be

\[
GACV(\lambda) = \frac{1}{n} \sum_{i=1}^{n} (-y_i \eta_\lambda(x_i) + \log(1 + e^{\eta_\lambda(x_i)})) + \frac{\text{tr}(H)}{n} \sum_{i=1}^{n} y_i(y_i - p_\lambda(x_i)) \frac{n - \text{tr}(W^{1/2}HW^{1/2})}{n}.
\] (2.7)

### 3 Simulation Results

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

#### 3.1 Computation of \( \eta_\lambda, \Sigma \) 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

\[
\eta_\lambda(\cdot) = \sum_{\nu=1}^{m} d_\nu \phi_\nu(\cdot) + \sum_{i=1}^{n} c_i \xi_i(\cdot),
\] (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 \( i^{th} \) 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 \( ij^{th} \) entry \( K(x_i, x_j) \). See Wahba (1990). Thus to find \( \eta_\lambda \) to minimize (1.2), we only need to find \( d = (d_1, \ldots, d_m)^T \) and \( c \) to minimize

\[
- \sum_{i=1}^{n} u_i \left( \sum_{\nu=1}^{m} d_\nu \phi_\nu(x_i) + \sum_{j=1}^{n} c_j \xi_j(x_i) \right) + c^T Q c.
\] (3.2)

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

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

\[
\Sigma = \Delta (\Delta Q \Delta^T)^\dag \Delta^T.
\] (3.3)
If $Q$ is of full rank, we can write

$$\Sigma = Q^{-1} - Q^{-1}S(S^TQ^{-1}S)^{-1}S^TQ^{-1}.$$  \hspace{1cm} (3.4)

Proof: See Appendix B.

We remark that in large problems the computation of $H$ and especially $\Sigma$ 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.

For most of our experiments we took $H$ as the Sobolev space $W_2 = \{\eta : \eta, \eta' \text{ abs. cont.}, \eta'' \in L_2\}$ and

$$J(\eta) = \int_0^1 (\eta''(x))^2dx.$$  

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 $n$th Bernoulli polynomial and $[\tau]$ is the fractional part of $\tau$. In one example, we assumed that $\eta$ was periodic, in this case, $m = 1$ and $\phi_2$ is deleted from the above representation.

We have chosen to use the representation (3.1) for our simulation studies in order to use the code RKPAC, which is used as a subroutine at each step of the iteration in (1.5). Since $\eta_\lambda(\cdot)$ of our example is known to be a cubic spline, a $B$-spline basis could have been used. 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 contains two typical GACV(\lambda) and KL(\eta, \eta_\lambda) curves from an example using logistic regression for Bernoulli data, that is $y_i$ is 1 or 0 with $E y_i = p(x_i)$ and $\eta(x) = \logit(p(x)) = \log(p(x)/(1 - p(x)))$, $b(\eta(x)) = \log(1 + e^{\eta(x)})$. In this figure $\eta(x) = 2\sin(2\pi x)$ and $x_i = (i - .5)/100$, $i = 1, \ldots, 100$ are equally spaced from 0 to 1. $\eta_\lambda$ here is a periodic function in $W_2$. The figure shows that the minima of GACV(\lambda) and KL(\eta, \eta_\lambda) are very close in these two examples.

Figure 3.2 is the average of GACV(\lambda) and KL(\eta, \eta_\lambda) curves over two hundred replicates of curves generated as in Figure 3.1, which may be thought of as approximations of $E(GACV(\lambda))$ and $E(KL(\eta, \eta_\lambda))$. Note that the scale in Figure 3.2 has been expanded and the curves appear to be very close.

### 3.3 Compare \lambda from GACV(\lambda), U(\lambda) and V(\lambda)

In this subsection, we are going to use simulations to compare the $\lambda$ chosen from GACV(\lambda), U(\lambda) and $V(\lambda)$. As for the implementation of $V$ and $U$, we will use Algorithm 2 since Gu has pointed
Figure 3.1: Two $GACV(\lambda)$ (solid lines) and $KL(\eta, \eta_\lambda)$ (dotted lines) curves

out that Algorithm 2, in general, will produce a better estimation than Algorithm 1.

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

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

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 $x_i = (i - .5)/100$, $i = 1, \cdots, 100$, Bernoulli data were generated according to the logit functions. Calculating $\eta_\lambda$ by minimizing (1.2) on a grid of
Figure 3.2: Average $GACV(\lambda)$ (solid lines) and $KL(\eta, \eta_\lambda)$ (dotted lines) curves

$log_{10}\lambda = -6.080$, and evaluating $GACV(\lambda)$ on the same grid to find the minimizing $\hat{\lambda}_{GACV}$. To obtain $trH$, we will 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 $tr(HW)$, we have to calculate exactly $H$ and then $HW$ before we get $Tr(HW)$. Also from $V(\lambda)$ and $U(\lambda)$, we have $\lambda_{GCV}$, $\lambda_{UBR}$ 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.

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

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

Figure 3.4 shows the boxplots of efficiency from those three $\lambda$'s. It is noticed that the distribution of the efficiencies from the $GCV$ $\lambda$'s (from $V(\lambda)$) have a relatively long tail. As for the $UBR$ $\lambda$'s, (from $U(\lambda)$), the results are similar to Gu (1992). It was expected that the $UBR$ $\lambda$'s will be better than $GCV$ $\lambda$'s for Bernoulli data, because it takes advantage of the lack of a nuisance parameter dispersion. But in cases (c) and (d) where the true functions are not smooth functions, $UBR$ performs as bad as $GCV$. In all the cases we tried, the $GACV$ $\lambda$ provides the best fitting among these three $\lambda$'s, especially for the case (f) when the true $\eta(\cdot)$ is only a linear function. And in general, it appears to have fewer outliers than either of its two competitors.
Figure 3.5 shows the pairwise comparison of GACV and UBR methods, where KL_GACV is plotted against KL UBWR. A point on the diagonal line means the two methods tied each other whereas a point in the right lower corner means KL_GACV is smaller than KL UBWR, which suggests GACV performed better than UBR on that set of data. From this figure, we can see that many of the cases the KL_GACV are almost the same as KL UBWR, but for each function, there exists 10% to 20% of KL UBWR still worse than KL_GACV.

4 Discussion

In this paper we have proposed a proxy, GACV(λ) for the Kullback-Leibler Information distance KL(η, η_λ), 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(λ), appears from simulations to be an excellent proxy for the Kullback-Leibler Information distance, in the sense that the minimizer of GACV(λ) is close to the minimizer of KL(η, η_λ), 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 U for large data sets, stable numerical methods for n ≈ 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(1993).

We have tried other proxies starting with a leaving-out-one expressions, and using different approximations at certain stages. For example, if we start with using mean square error for our cross validation function, replacing negative loglikelihood by the mean square error in (2.1), the same derivation will lead us to the weighted GCV function,

$$GCV(\lambda) = \frac{\|Y - \mu_\lambda\|^2}{[tr(I - HW)]^2},$$

(4.1)

which is identical to GCV(λ) for the Gaussian case if the noise are from identical normal distributions.

Since

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

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

14
another way to take the generalization step from the above \( ACV(\lambda) \) will give us

\[
GACV_2(\lambda) = \frac{1}{n} \sum_{i=1}^{n} (-y_i \eta_\lambda(x_i) + b(\eta_\lambda(x_i))) + \frac{tr(W^{1/2}HW^{1/2})}{n} \sum_{i=1}^{n} \frac{(y_i - \mu_\lambda(x_i))^2}{n - tr(W^{1/2}HW^{1/2})}.
\]

(4.2)

In particular, for Bernoulli data,

\[
GACV_2(\lambda) = \frac{1}{n} \sum_{i=1}^{n} (-y_i \eta_\lambda(x_i) + b(\eta_\lambda(x_i))) + \frac{tr(W^{1/2}HW^{1/2})}{n} \sum_{i=1}^{n} \frac{(y_i - p_\lambda(x_i))^2}{n - tr(W^{1/2}HW^{1/2})}.
\]

(4.3)

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

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

(4.4)

where \(k = n/(n - tr(W^{1/2}HW^{1/2}))\). This version of GACV is very similar to that proposed by Gu in (1.8), where in (1.8), the first part is an approximation of the loglikelihood in (4.4). But for the examples we studied in this paper, simulation suggests that GACV is better than GACV_2.

A Proof of Lemma 2.1

First define \(Y^{-i} = (y_1, \cdots, y_{i-1}, \mu_\lambda^{(-i)}(x_i), y_{i+1}, \cdots, y_n)\). Since

\[-l(\mu_\lambda^{(-i)}(x_i), \tau) = -u_\lambda^{(-i)}(x_i)\tau + b(\tau),\]

we have that

\[-l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) \leq -l(\mu_\lambda^{(-i)}(x_i), \eta(x_i)).\]

(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 \(\eta_\cdot\)

\[
I_\lambda(\eta, Y^{-i}) = -l(\mu_\lambda^{(-i)}(x_i), \eta(x_i)) - \sum_{j \neq i} l(y_j, \eta(x_j)) + n^\lambda_2 J(\eta)
\]

\[
\geq -l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) - \sum_{j \neq i} l(y_j, \eta(x_j)) + n^\lambda_2 J(\eta)
\]

\[
\geq -l(\mu_\lambda^{(-i)}(x_i), \eta_\lambda^{(-i)}(x_i)) - \sum_{j \neq i} l(y_j, \eta_{\lambda}^{(-i)}(x_j)) + n^\lambda_2 J(\eta_{\lambda}^{(-i)}).
\]

The first inequality is because of (A.1), the second inequality is due to the fact that \(\eta_{\lambda}^{(-i)}\) is the minimizer of \(- \sum_{j \neq i} l(y_j, \eta(x_j)) + n^\lambda_2 J(\eta)\). Thus we have

\[
h_\lambda(i, \mu_\lambda^{(-i)}) = \eta_{\lambda}^{(-i)}.
\]
B Proof of Lemma 3.1

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

\[
Qc + Sd = \eta \\
S^Tc = 0.
\]  
(B.1)

Let \( c = \Delta \gamma \) for some \( n - m \) dimensional vector \( \gamma \), where \( \Delta \) is as defined in the text. That is necessary and sufficient to insure that \( S^Tc = 0 \). Then \( c^TQc = \gamma^T \Delta^TQ\Delta \gamma \). Substituting into (B.1) gives

\[
(\Delta^TQ\Delta)\gamma = \Delta^T \eta.
\]  
(B.2)

If \( \Delta^TQ\Delta \) is not of full rank, then \( \gamma \) is not uniquely defined by (B.2), nevertheless, \( \gamma^T \Delta^TQ\delta \gamma \) depends only on that the projection of \( \gamma \) onto the column space of \( \Delta^TQ\Delta \). Therefore we can take \( \gamma \) as \( \gamma = (\Delta^TQ\Delta)\Delta^T \eta \) and the result follows. If \( Q \) is of full rank then formulas for the block inverse of a matrix gives the unique solution to the system (B.1), with

\[
c = (Q^{-1} - Q^{-1}S(S^TQ^{-1}S)^{-1}S^TQ^{-1})\eta
\]  
(B.3)

and direct substitution of (B.3) into \( c^TQc \) and collection of terms gives the result.

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.

References

Cox, D. & 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.


Association, Washington, DC.
Figure 3.3: The true $p(x)$ and a set of data with $p(x)$ depend on (a): $\eta_1$, (b): $\eta_2$, (c): $p_3$, (d): $p_4$, (e): $\eta_5$ and (f): $\eta_6$
Figure 3.4: Boxplots of the Inefficiency for 6 different examples, for GCV(algorithm 2), UBR (algorithm 2) and GACV
Figure 3.5: Pairwise comparison for GACV and UBR.