ALGORITHMS FOR MULTIDIMENSIONAL SEMIPARAMETRIC GLM'S

Brian S. Yandell

Departments of Statistics and Horticulture
University of Wisconsin–Madison

Key Words and Phrases: generalized linear model; generalized residuals; iteratively reweighted least squares; partial thin plate smoothing spline; penalized likelihood; semiparametric model.

Purpose and Description

Purpose

These Fortran-77 subroutines provide tools for penalized likelihood estimation and model checking for generalized linear models (GLMs) in which the model has a semi-parametric form. The routines build on GCVPACK (Bates et al., 1987) and are designed to use the generalized cross-validation criteria (Craven and Wahba, 1979) to determine the degree of data smoothing. These problems include smoothed GLMs (O'Sullivan, Yandell and Raynor, 1986), iteratively reweighted least squares (Green, 1984), and general nonlinear problems. We present some of the problems PGLMPACK is designed for and describe the structure of the routines.

General Problem: A variety of penalized nonlinear problems can be solved by an iterative scheme in which the inner step involves a linear model approximation,

$$y_i \approx \theta_i + \varepsilon_i\ ,\ i=1,\cdots,n\ ,$$

with $\mathbf{y} = (y_1,\cdots,y_n)^T$ the working values, $\boldsymbol{\theta} = (\theta_1,\cdots,\theta_n)^T$ the linearized model and $\boldsymbol{\varepsilon} = (\varepsilon_1,\cdots,\varepsilon_n)^T$ a random vector with zero mean and covariance $\mathbf{W}^{-2}$, which is often diagonal. (The matrix $\mathbf{W}$ is referred to as the working weights.) In many situation, a semiparametric model is appropriate, such as

$$\theta_i = \mathbf{s}_i^T\alpha + f(\mathbf{x}_i)\ ,\ i=1,\cdots,n\ , \tag{1.1}$$

in which $\mathbf{s}_i$ is a c-vector of covariates with corresponding parameter vector α, $\mathbf{x}_i$ is a d-vector of variates and $f(\cdot)$ is some "smooth" function. Smoothness can be enforced by a "roughness penalty'', $J(f)$, with a common choice being the integrated squared mth derivative (*cf.* Bates *et al.*, 1987). The solution to such a penalized linear model minimizes

$$\frac{1}{n} \|\mathbf{W}(\mathbf{y}-\boldsymbol{\theta})\|^2 + \lambda J(f) \tag{1.2}$$

for some fixed λ, leading to a solution of the linear model as $\hat{\boldsymbol{\theta}} = \mathbf{W}^{-1}\mathbf{A}(\lambda)\mathbf{W}\mathbf{y}$, with $\mathbf{A}(\lambda)$ the ''hat'' matrix. One can then iterate on the nonlinear problem to convergence.

The choice of the tuning constant λ has been a subject of considerable discussion (Rice, 1984; Härdle and Marron, 1985 a,b). We limit discussion to choices based on minimizing the generalized cross validation (GCV) criterion (Craven and Wahba, 1979)

$$V(\lambda) = \frac{n \, \|(\mathbf{I}-\mathbf{A}(\lambda))\mathbf{W}\mathbf{y}\|^2}{[tr(\mathbf{I}-\mathbf{A}(\lambda))]^2} \ . \tag{1.3}$$

However, our development could be easily modified for any data-driven criterion based on $\mathbf{A}(\lambda)$. What we propose to do is to iterate on both $\boldsymbol{\theta}$ *and* λ to find the $\hat{\lambda}$ which minimizes (1.3) with $\hat{\boldsymbol{\theta}}$ minimizing (1.2). It is not known whether such a procedure will converge, but we conjecture that, if the GCV minimizer is bounded away from 0 and ∞ and the nonlinear problem is suitably convex, then it does converge.

If the penalty is chosen so that the estimate of f is a member of a reproducing kernel Hilbert space then the penalty, and hence (1.2), can be expressed in a quadratic form (Aronszajn, 1950). Such a space can be partitioned into a ''smooth'' space which is defined by the penalty, and a ''null'' space which is annihilated by the penalty. The semiparametric model (1.1) can be written in matrix form as

$$\boldsymbol{\theta} = \mathbf{S}\boldsymbol{\alpha} + \mathbf{T}\boldsymbol{\beta} + \mathbf{K}\boldsymbol{\delta} \ ,$$

with $\mathbf{S}$ the $n \times c$ covariate matrix with rows $\mathbf{s}_i^{\mathrm{T}}$, $\mathbf{T}$ an $n \times t$ matrix whose columns span the null space, and $\mathbf{K}$ an $n \times k$ matrix spanning the smooth space. If J penalizes the integrated squared mth derivative, then the ith column of $\mathbf{T}$ contains the low order polynomials in $\mathbf{x}_i$ of total order at most $m-1$ and the ijth entry of $\mathbf{K}$ is proportional to $\|\mathbf{x}_i - \mathbf{x}_j\|^{2m-d}$ (*cf.* Bates *et al.*, 1987).

Let $\mathbf{K}_U$ be the $k \times k$ matrix corresponding to the quadratic penalty for J and let $\mathbf{T}_U$ be the $k \times t$ matrix spanning the null space. Typically $\mathbf{K}_U$ and $\mathbf{T}_U$ are either derived from the unique design points or from a set of user-supplied basis nodes (see Appendix 2 of Bates et al. (1987)). The objective function (1.2) can be expressed as

$$\frac{1}{n} \|\mathbf{W}(\mathbf{y}-\mathbf{S}\boldsymbol{\alpha}-\mathbf{T}\boldsymbol{\beta}-\mathbf{K}\boldsymbol{\delta})\|^2 + \lambda\boldsymbol{\delta}^{\mathrm{T}}\mathbf{K}_U\boldsymbol{\delta} \tag{1.4}$$

subject to $\mathbf{T}_U^{\mathrm{T}}\boldsymbol{\delta} = \mathbf{0}$. We propose computational solutions when matrices and working vectors in (1.4) may depend on the unknown parameters. Some problems of interest include: (a) semiparametric generalized linear models, in which the matrices $\mathbf{S}$, $\mathbf{T}$, $\mathbf{K}$ and $\mathbf{K}_U$ are constant while the working values $\mathbf{W}$ and $\mathbf{y}$ may change with each iteration; (b) iteratively reweighted least squares, in which only $\mathbf{K}_U$ remains constant; and (c) general nonlinear problems (remote sensing, for example), in which all matrices may change with each iteration.

One would like to decompose any constant matrices exactly once and to keep decompositions of the changing matrices as cheap as possible. The method proposed here combines the advantages of the singular value decomposition (SVD) (Dongarra et al., 1979, chapter 10) in locating the GCV choice of λ with Cholesky decompositions (CDs) (Dongarra et al., 1979, chapter 8) which are relatively cheap once λ is fixed. While the decompositions suggested are not new, the combination of approaches appears to be an unexplored area. The basic strategy is as follows:

(1) choose an initial guess of λ, e.g., $\lambda=\infty$;
(2) find estimates of $(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\delta})$ by iteration using CDs;
(3) linearize the problem based on the iterated solution;
(4) use SVD to diagonalize $\mathbf{A}(\lambda)$;
(5) choose new λ using GCV or another method;
(6) interate through (2)–(5) until convergence.

Convergence criteria can include absolute or relative convergence of the regularization functional and/or the parameter estimates, and absolute convergence of $\log(n\lambda)$. The number of iterations in (2) may be restricted, leading to rough estimates which are fed into (3).

We do not assume any special structure to the design or the matrices, except that we suppose that $\mathbf{W}$ is of full rank, symmetric and computationally invertible. In many cases, $\mathbf{W}$ is actually diagonal, but this will not be explicitly used in the linear algebra. The algorithms below are extensions of Bates et al. (1987), building on their Fortran77 package, GCVPACK.

Semiparametric Generalized Linear Models: Semiparametric generalized linear model parameter estimation can be formulated as the problem of minimizing, for fixed λ,

$$S_\lambda(\boldsymbol{\theta}) = -2L(\boldsymbol{\theta}) + \lambda J(f) \ . \tag{2.1}$$

in which $\boldsymbol{\theta}$ is of the form (1.1), $L(\boldsymbol{\theta})$ is the log likelihood and J is the smoothing penalty (see O'Sullivan, Yandell and Raynor, 1986; Green and Yandell, 1985). If $L(\boldsymbol{\theta})$ is suitably convex and $J(\boldsymbol{\theta})$ has a quadratic form, then $S_\lambda(\boldsymbol{\theta})$ has a unique minimum for each λ. These conditions appear to hold for many generalized linear models.

The log likelihood can be written in an iterative form using the score vector $\mathbf{u}$, the working-weights $\mathbf{W}$ and the working-values $\mathbf{y}$,

$$\mathbf{u} = \left[\frac{\partial L}{\partial \theta}\right]_{\theta^\circ} ,$$

$$\mathbf{W}^2 = E\left[-\frac{\partial^2 L}{\partial \theta \theta^T}\right]_{\theta^\circ} , \tag{2.2}$$

$$\mathbf{y} = \theta^\circ + \mathbf{W}^{-2}\mathbf{u} ,$$

based on θ° from the previous iteration. Note that for the independent normal model, $\mathbf{W}^{-1}$ is a diagonal matrix of the standard deviations and $\mathbf{y}$ is the vector of observed responses. The log likelihood is approximated by a quadratic based on a two-term Taylor series expansion (*cf.* Yandell and Hogg, 1987),

$$L(\theta) \approx L(\theta^\circ) + \tfrac{1}{2} \, \|\mathbf{W}^{-1}\mathbf{u}\|^2 - \tfrac{1}{2} \, \|\mathbf{W}(\mathbf{y}-\theta)\|^2 .$$

This allows one to locally approximate the penalized likelihood by (1.2). It is well known (Green, 1984) that under regularity conditions the iteratively reweighted least squares solution based on (1.2) is the same as the maximizer of (2.1).

We first decompose the constant matrices. Locating the unique design points $\mathbf{T}_U$ and the corresponding unique covariates $\mathbf{S}_{1U}$ (if any) we form a QR decomposition (Dongarra et al., 1979, chapter 9)

$$[\mathbf{T}_U : \mathbf{S}_{1U}] = \bar{\mathbf{F}}\bar{\mathbf{G}} = [\bar{\mathbf{F}}_1 : \bar{\mathbf{F}}_2]\begin{bmatrix}\bar{\mathbf{G}}_1\\ \mathbf{0}\end{bmatrix} = \bar{\mathbf{F}}_1\bar{\mathbf{G}}_1 .$$

From this we construct the (unweighted) design

$$\mathbf{X} = [\mathbf{T} : \mathbf{S} : \mathbf{K}\bar{\mathbf{F}}_2] \tag{2.3}$$

and penalty

$$\Sigma = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \bar{\mathbf{F}}_2^T\mathbf{K}_U\bar{\mathbf{F}}_2\end{bmatrix} . \tag{2.4}$$

We decompose Σ using a pivoted Cholesky followed by a QR decomposition,

$$\mathbf{E}^T\Sigma\mathbf{E} = \mathbf{L}^T\mathbf{L} \quad \text{and} \quad \mathbf{L}^T = \mathbf{Q}\mathbf{R} = \mathbf{Q}_1\mathbf{R}_1 , \tag{2.5}$$

and construct

$$\mathbf{Z} = [\mathbf{Z}_1 : \mathbf{Z}_2] = \mathbf{X}\mathbf{E}\mathbf{Q}\begin{bmatrix}\mathbf{R}_1^{-T} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} . \tag{2.6}$$

Note that $\bar{\mathbf{F}}_2^T\mathbf{K}_U\bar{\mathbf{F}}_2$ is of rank $q \le k - t$, and $\mathbf{L}^T$, $\mathbf{Q}_1$, $\mathbf{R}_1$ and $\mathbf{Z}_1$ all have q columns. The original parameters are transformed to

$$\begin{bmatrix}\beta\\ \alpha\\ \bar{\mathbf{F}}_2^T\delta\end{bmatrix} = \mathbf{E}\mathbf{Q}\begin{bmatrix}\mathbf{R}_1^{-T} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\gamma\\ \omega\end{bmatrix} \tag{2.7}$$

with $\boldsymbol{\gamma}$ of length q and $\boldsymbol{\omega}$ of length $t+c$. The objective function becomes

$$\frac{1}{n} \| \mathbf{W}(\mathbf{y} - \mathbf{Z}_2\boldsymbol{\omega} - \mathbf{Z}_1\boldsymbol{\gamma}) \|^2 + \lambda \boldsymbol{\gamma}^T \boldsymbol{\gamma} . \tag{2.8}$$

At this point, we have done all the "one-time" decompositions. The following steps must be redone each time $\mathbf{W}$ and $\mathbf{y}$ change. We form a QR decomposition of

$$\mathbf{W}\mathbf{Z}_2 = \mathbf{F}\mathbf{G} = \mathbf{F}_1\mathbf{G}_1 ,$$

and create

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{F}_1^T \\ \mathbf{F}_2^T \end{bmatrix} \mathbf{W}\mathbf{Z}_1 ,$$

with $\mathbf{J}_1$ being $(t+c)\times q$ and $\mathbf{J}_2$ being $(n-t-c)\times q$. This leads to the final form of the objective function,

$$\begin{aligned} &\frac{1}{n} \| \mathbf{F}_1^T\mathbf{W}\mathbf{y} - \mathbf{G}_1\boldsymbol{\omega} - \mathbf{J}_1\boldsymbol{\gamma} \|^2 + \\ &\frac{1}{n} \| \mathbf{F}_2^T\mathbf{W}\mathbf{y} - \mathbf{J}_2\boldsymbol{\gamma} \|^2 + \lambda\boldsymbol{\gamma}^T\boldsymbol{\gamma} , \end{aligned} \tag{2.9}$$

in which the first term can be made zero by solving for $\boldsymbol{\omega}$, with any given $\boldsymbol{\gamma}$,

$$\mathbf{G}_1\boldsymbol{\omega} = \mathbf{F}_1^T\mathbf{W}\mathbf{y} - \mathbf{J}_1\boldsymbol{\gamma} . \tag{2.10}$$

The latter two terms of (2.9) comprise a ridge regression (Golub, Heath, and Wahba, 1979), with the estimate of $\boldsymbol{\gamma}$ found by solving

$$\mathbf{M}\boldsymbol{\gamma} = \mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} , \tag{2.11}$$

where

$$\mathbf{M} = \mathbf{J}_2^T\mathbf{J}_2 + n\lambda\mathbf{I} .$$

The "hat" matrix can be formally written as

$$\mathbf{A}(\lambda) = \mathbf{F} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T \end{bmatrix} \mathbf{F}^T . \tag{2.12}$$

Naturally, one would iterate to new working-values and working-weights using (2.2) and repeat the minimization of the objective function (2.8). At convergence, one can obtain the estimates of the original parameters via (2.7).

Singular Value or Cholesky Decomposition?: One may approach the solution of (2.11) for $\boldsymbol{\gamma}$ and the hat matrix (2.12) in different ways, depending on whether one wishes to select a new λ or whether one wishes to leave λ fixed. One way to automate choice of a new λ is based on GCV for the linearized problem (2.9). We can diagonalize $\mathbf{A}(\lambda)$ with a singular value decomposition (SVD) to simplify the search (Golub, Heath, and Wahba, 1979). Decompose

$$\mathbf{J}_2 = \mathbf{U}\mathbf{D}\mathbf{V}^T ,$$

where $\mathbf{D}$ is diagonal of size $a = min(q, n-t-c)$ and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal of sizes $(n-t-c)\times a$ and $a \times q$, respectively. The parameter estimates are

$$\hat{\gamma} = \mathbf{V}(\mathbf{D}^2 + n\lambda\mathbf{I})^{-1}\mathbf{D}\mathbf{U}^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} ,$$

with $\hat{\omega}$ determined by (2.10). The hat matrix is diagonalized as

$$\mathbf{A}(\lambda) = \mathbf{F}\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{U}\mathbf{D}^2(\mathbf{D}^2 + n\lambda\mathbf{I})^{-1}\mathbf{U}^T \end{bmatrix}\mathbf{F}^T .$$

These leads to a rational polynomial representation of the GCV criterion (1.3), which can easily by minimized by a golden section search, as in GCVPACK.

If instead λ is fixed, one can take the cheaper approach of a Cholesky decomposition (CD) of

$$\mathbf{M} = \mathbf{C}^T\mathbf{C} ,$$

leading to the estimate of γ by solving

$$\mathbf{C}^T\mathbf{C}\hat{\gamma} = \mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} .$$

The hat matrix becomes

$$\mathbf{A}(\lambda) = \mathbf{F}\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2\mathbf{C}^{-1}\mathbf{C}^{-T}\mathbf{J}_2^T \end{bmatrix}\mathbf{F}^T . \qquad (2.13)$$

Diagnostics: The diagonal elements of the hat matrix have been used for diagnostics in GLMs (Pregibon, 1981) as well as in smoothing spline models (Eubank 1984, 1985). Recently they have been extended to semiparametric GLMs (Yandell and Green, 1986). The diagonal elements of (2.12) can be computed formally as

$$\{\mathbf{A}(\lambda)\}_{ii} = \|\mathbf{F}_1^T\mathbf{e}_i\|^2 + \|\mathbf{M}^{-1/2}\mathbf{J}_2^T\mathbf{F}_2^T\mathbf{e}_i\|^2$$

in which $\mathbf{e}_i$ is the n-vector with a 1 in the i-th position and 0's elsewhere. For the SVD approach this is simply

$$\{\mathbf{A}(\lambda)\}_{ii} = \|\mathbf{F}_1^T\mathbf{e}_i\|^2 + \|\mathbf{D}(\mathbf{D} + n\lambda\mathbf{I})^{-1/2}\mathbf{U}^T\mathbf{F}_2^T\mathbf{e}_i\|^2 ,$$

and for the Cholesky approach (*cf.* O'Sullivan (1985)),

$$\{\mathbf{A}(\lambda)\}_{ii} = \|\mathbf{F}_1^T\mathbf{e}_i\|^2 + \|\mathbf{C}^{-T}\mathbf{J}_2^T\mathbf{F}_2^T\mathbf{e}_i\|^2 .$$

The trace of $\mathbf{A}(\lambda)$ can be quickly computed if one is not interested in the diagonal entries by noting that

$$tr(\mathbf{A}(\lambda)) = t + c + tr(\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-1}) = q + t + c - n\lambda tr(\mathbf{M}^{-1}) .$$

For the SVD approach (see Bates et al. (1987)) this is

$$tr(\mathbf{A}(\lambda)) = t + c + \sum_{i=1}^{a} d_i^2/(d_i^2 + n\lambda)$$

$$= a + t + c - \sum_{i=1}^{a} (1 + d_i^2/(n\lambda))^{-1} \ .$$

For the CD approach we have

$$tr(\mathbf{A}(\lambda)) = q + t + c - n\lambda tr(\mathbf{C}^{-1}\mathbf{C}^{-T})$$

$$= q + t + c - n\lambda \sum_{i=1}^{q} \| \mathbf{C}^{-T}\mathbf{e}_i \|^2 \ ,$$

in which $\mathbf{e}_i$ here is of length q.

Covariance matrices can be computed by noting that $COV(\mathbf{y}) = \mathbf{W}^{-2}$. Considering first the linear model estimates of (1.1), we find from (2.12) that

$$COV(\hat{\boldsymbol{\theta}}) = \mathbf{W}^{-1}\mathbf{F}\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T \end{bmatrix}\mathbf{F}^T\mathbf{W}^{-1} \ .$$

Hence, the variances are

$$VAR(\theta_i) = \| \mathbf{F}_1^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 + \| \mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 \ .$$

For the SVD approach, this becomes

$$VAR(\theta_i) = \| \mathbf{F}_1^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 + \| \mathbf{D}^2(\mathbf{D}^2 + n\lambda\mathbf{I})^{-1}\mathbf{U}^T\mathbf{F}_2^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 \ .$$

Noting that

$$\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-1} = \mathbf{M}^{-1} - n\lambda\mathbf{M}^{-2} \ , \tag{3.1}$$

the variances under the Cholesky approach can be written as

$$VAR(\theta_i) = \| \mathbf{F}_1^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 + \| \mathbf{C}^{-T}\mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}^{-1}\mathbf{e}_i \|^2$$

$$- n\lambda \| \mathbf{C}^{-1}\mathbf{C}^{-T}\mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}^{-1}\mathbf{e}_i \|^2 \ .$$

The covariance among the coefficients can be derived, using (2.7), (2.10) and (2.11) as

$$COV\begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\alpha} \\ \mathbf{F}_2^T\boldsymbol{\delta} \end{bmatrix} = \mathbf{E}\mathbf{Q}_2\mathbf{G}_1^{-1}\mathbf{G}_1^{-T}\mathbf{Q}_2^T\mathbf{E}^T +$$

$$\mathbf{E}\mathbf{Q}\begin{bmatrix} \mathbf{R}_1^{-T} \\ -\mathbf{G}_1^{-1}\mathbf{J}_1 \end{bmatrix}\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-1}\begin{bmatrix} \mathbf{R}_1^{-T} \\ -\mathbf{G}_1^{-1}\mathbf{J}_1 \end{bmatrix}^T\mathbf{Q}^T\mathbf{E}^T \ .$$

In many situations we may be only interested in $COV(\boldsymbol{\alpha})$. Further, if the penalty $\boldsymbol{\Sigma}$ is of the proper rank, then (2.7) essentially permutes and rotates the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ into $\boldsymbol{\omega}$. Let $\tilde{\mathbf{e}}_i$ =

$\mathbf{G}_1^{-T}\mathbf{Q}_2^T\mathbf{E}^T\mathbf{e}_{t+i}$, $i=1,\cdots,c$, be the transformed index for α_i. For the SVD approach,

$$VAR(\alpha_i) = \|\bar{\mathbf{e}}_i\|^2 + \|\mathbf{D}(\mathbf{D}^2+n\lambda\mathbf{I})^{-1}\mathbf{V}^T\mathbf{J}_1^T\bar{\mathbf{e}}_i\|^2 .$$

For the Cholesky approach, using (3.1),

$$VAR(\alpha_i) = \|\bar{\mathbf{e}}_i\|^2 + \|\mathbf{C}^{-T}\mathbf{J}_1^T\bar{\mathbf{e}}_i\|^2 - n\lambda\|\mathbf{C}^{-1}\mathbf{C}^{-T}\mathbf{J}_1^T\bar{\mathbf{e}}_i\|^2 .$$

Tests: One may test parameters using the covariance matrix given above. One can perform stepwise tests in nested semiparametric GLMs using score tests which are computationally more appealing than tests based on deviances (Pregibon, 1982, Yandell and Green, 1986). One can also test whether the f is parametric using analogues to recent results of Cox et al. (1987).

Consider testing a full model

$$\tilde{\theta}_i = \mathbf{s}_i^T\alpha + \bar{\mathbf{s}}_i^T\bar{\alpha} + f(\mathbf{x}_i)$$

against the reduced model

$$\theta_i = \mathbf{s}_i^T\alpha + f(\mathbf{x}_i) .$$

In other words, one tests whether the r-vector $\bar{\alpha}=0$. The score statistic is

$$\mathbf{u} = \left[\frac{\partial L}{\partial \theta}\right]_{\hat{\theta}} = \mathbf{W}^2(\mathbf{y}-\hat{\theta}) = \mathbf{W}(\mathbf{I}-\mathbf{A})\mathbf{W}\mathbf{y}$$

$$= \mathbf{W}\mathbf{F}_2(\mathbf{I}-\mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T)\mathbf{F}_2^T\mathbf{W}\mathbf{y} = \mathbf{H}\mathbf{y} .$$

The score test for $\bar{\alpha}$ is

$$\begin{aligned} T &= \mathbf{u}^T\bar{\mathbf{S}}(\bar{\mathbf{S}}^T\mathbf{H}\bar{\mathbf{S}})^{-1}\bar{\mathbf{S}}^T\mathbf{u} \\ &= \mathbf{y}^T\mathbf{H}\bar{\mathbf{S}}(\bar{\mathbf{S}}^T\mathbf{H}\bar{\mathbf{S}})^{-1}\bar{\mathbf{S}}^T\mathbf{H}\mathbf{y} , \end{aligned} \tag{3.2}$$

with $\bar{\mathbf{S}}$ the $n\times r$ matrix with rows $\bar{\mathbf{s}}_i^T$. This test is conjectured to have approximately a χ^2 distribution with r degrees of freedom when $\bar{\alpha}=0$. For the SVD approach define the $q\times r$ matrix

$$\mathbf{B} = (\mathbf{I}+\mathbf{D}^2/(n\lambda))^{-1/2}\mathbf{U}^T\mathbf{F}_2^T\mathbf{W}\bar{\mathbf{S}}$$

and transformed working values

$$\bar{\mathbf{y}} = (\mathbf{I}+\mathbf{D}^2/(n\lambda))^{-1/2}\mathbf{U}^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} .$$

Form the QR decomposition of $\mathbf{B} = \bar{\mathbf{Q}}\bar{\mathbf{R}} = \bar{\mathbf{Q}}_1\bar{\mathbf{R}}_1$. The score test (3.2) can then be written as

$$T = \bar{\mathbf{y}}^T\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\bar{\mathbf{y}} = \|\bar{\mathbf{Q}}_1^T\bar{\mathbf{y}}\|^2 .$$

For the CD approach,

$$\mathbf{H} = \mathbf{W}\mathbf{F}_2(\mathbf{I}-\mathbf{J}_2\mathbf{C}^{-1}\mathbf{C}^{-T}\mathbf{J}_2^T)\mathbf{F}_2^T\mathbf{W} .$$

Forming the Cholesky decomposition of

$$\bar{\mathbf{S}}^T\mathbf{H}\bar{\mathbf{S}} = \mathbf{L}^T\mathbf{L} ,$$

the score test (3.2) becomes

$$S = \| \mathbf{L}^{-T}\tilde{\mathbf{S}}^T\mathbf{H}\mathbf{y} \|^2 .$$

Note that when $r = 1$, calculations for both the SVD and the CD approaches simplify greatly, obviating the need for the extra decomposition in either case.

Cox and Koh (1986) proposed a test for f parametric in the simple spline model, which was later extended to partial and generalized spline models by Cox et al. (1987). One can readily show that the nonlinear analogue of that test is

$$T = \| \mathbf{D}\mathbf{U}^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} \|^2 = \| \mathbf{J}_2^T\mathbf{F}_2^T\mathbf{W}\mathbf{y} \|^2 .$$

This can be easily computed after convergence is reached. In fact, the computation overlap with some of those needed for the score test, which can lead to some time savings. Unfortunately, the distribution of T is a weighted sum of χ^2 statistics which has no closed form (Cox and Koh, 1986).

Other Nonlinear Models: Iteratively reweighted least squares (IRLS) models differ from semiparametric GLMs in that only the penalty matrix remains fixed (Green, 1984). The likelihood parameter $\boldsymbol{\theta}$ can be locally linearized, but the **S**, **T**, and **K** matrices are no longer fixed. For instance, with a penalized likelihood of the form (2.1),

$$\mathbf{S} = \frac{\partial L}{\partial \alpha} , \ \mathbf{T} = \frac{\partial L}{\partial \beta} \text{ and } \mathbf{K} = \frac{\partial L}{\partial \delta} ,$$

which may depend on the unknown parameters. We still only need form and decompose Σ as in (2.4) and (2.5) exactly once. However, the (unweighted) design (2.3) may change with each iteration. Hence, the remaining computations need to be done at each iteration. One could proceed in the same manner as for the generalized linear models, but reconstructing **X**, and hence **Z** and **J**, each time.

General nonlinear problems could proceed in the same manner as for IRLS, except that $\mathbf{K}_U$ changes each time. Thus most computations need to be redone. It may be possible for some nonlinear problems to reparameterize them as SGLM or IRLS problems to eliminate this difficulty.

Description

The package has one main driver, *dpglm*, for penalized general linear models. The subroutine *dpglm* calls *dmksx* to make the penalty Σ and the design matrix **X** using GCVPACK routines *dmaket*, *dmakek* and *dctsx*. The penalty Σ is decomposed by a call to the GCVPACK routine *dsgdc*. Then the matrix **Z** is created by the GCVPACK routine *dcrtz*.

The routine *dnrfs* handles all computations for the iterations. The model is initialized by a call to *dmodel*, which is one of *dbin*, *dpois* or *dnorm*, depending on the model selected: binomial, poisson or normal. This routine handles evaluation of the likelihood and updating of working-values. The algorithms are set up for a diagonal $\mathbf{W}$ matrix, and would have to be slightly modified for more general•working-weights. At each iteration for either the CD or the SVD approach, *dmodel* is called to update working values, and *dcheck* is called at the end to check the convergence criteria. For the SVD approach, the GCVPACK routines *dzdc* and *dgcv* are called to decompose $\mathbf{Z}$ and to locate a new λ by GCV. For the CD approach, $\mathbf{Z}$ is decomposed in the routine *dchrr*.

Once convergence is established, *dnrfs* computes the predictive MSE (if requested) and back-transforms the predicted values to the original units. It computes the diagonal of the hat matrix (if requested) by *dcdiag* for the CD approach or by the GCVPACK routine *ddiag* for the SVD approach. The variances of the parameter estimates (if requested) are computed in *dvar* for both the SVD and the CD approaches. The test statistic for parametric f is then computed by *dnrfs*. If score tests are requested, then *dsvst* or *dchst* computes the overall test of $\bar{\alpha}=0$, along with single tests for for each of the r elements of $\bar{\alpha}$. Once *dnrfs* returns, *dpglm* does some final cleanup using LINPACK routines.

The user can control whether λ is to be considered fixed or to be automatically chosen, how many CD iterations are done each loop, and how many over CD and SVD iterations are performed.

Related Algorithms

The numerical linear algebra in our routines is performed using the LINPACK (Dongarra et al., 1979) routines. The linear algebra for generalized cross validation is performed using GCVPACK (Bates et al., 1987). The introductory comments of each PGLMPACK routine list which GCVPACK, LINPACK and BLAS (Basic Linear Algebra Subroutines) routines are called directly or indirectly. There is one machine–dependent constant, the relative machine precision, which is used in these routines to determine error conditions caused by ill-conditioning, but that constant is computed each time it is needed.

The present work generalizes GCVPACK algorithms for linear models of Bates et al. (1987) and references therein. It would also be possible to take advantage of block diagonal forms (Yandell, 1987) to realize further savings of time and storage space.

O'Sullivan, Yandell and Raynor (1986) developed algorithms for smooth generalized linear models based on the Cholesky decomposition. Green (1985) and Green and Yandell (1985) presented algorithms for penalized likelihood schemes which include generalized linear models and other iteratively reweighted least squares methods. Green and Yandell (1985) present a one-dimensional algorithm based on Reinsch (1967) and a general algorithm based on the Cholesky decomposition. See also O'Sullivan (1985). Yandell (1985) developed an earlier version of the present multidimensional algorithms. Hastie and Tibshirani (1986) and Buja, Hastie, and Tibshirani (1987) developed algorithms for generalized additive models using the "backfitting algorithm" pioneered by Friedman and Stuetzle (1981).

If one follows Elden (1984) to stop the singular value decomposition after the bidiagonalization, considerable time can be saved since the effort to diagonalize is magnified by the number of iterations. Earlier work on GCVPACK (Bates et al., 1987) indicated that half of the singular value decomposition time may be spent on bidiagonalization. Of course, once convergence is reached, one could complete the diagonalization, doing this only once, to easily derive the diagonal of the "hat" matrix. Such a savings in computation would further reduce the advantage of iterating via Cholesky with fixed λ (see Test Results section).

Test Results

The package and drivers have been tested for internal consistency and for accuracy against other known algorithms. Here we present some timing results to show that the methods are feasible for relatively large data sets and to offer insight into which portions of the code should be improved, if possible.

All timing runs were performed on a Vax-11/750 computer with a floating point accelerator and running the 4.2 BSD *UNIX*™ operating system. All timing was performed using GCVPACK with the standard BLAS of LINPACK (Dongarra et al., 1979).

We focus our investigations upon the Poisson and binomial special cases of the semiparametric generalized linear model as these are potentially of wide interest and easy to formulate. We allowed up to c initial iterations of the Cholesky decomposition (CD) for $\lambda = \infty$ (perfectly smooth case), and up to c CDs following each SVD, where c was 1, 2, or 10. No case required more than 7 CD following an SVD, or more than 7 SVD overall.

We simulated data which we thought might be "cumbersome" for the numerical algorithms. Simulations were conducted for $n = 50$ and 100. The simulations were Poisson with a normal shaped curve of θ = log(mean value), with peak height of between $\theta = 1.5$ and 20. Binomial simulations used a similar normal shaped curve for θ = logit(mean value), with peak height of between $\theta = \text{logit}(.05)$ and logit(.3) for $n = 50$ and between. $\theta = \text{logit}(.005)$ and logit(.3) for $n = 100$.

The simulations showed that when the "signal" is small relative to the "noise", the CDs seem to stabilize the minimization problem, reducing the number of SVDs required and cutting the run time. Table 1(a) present the combined CD and SVD run times, while Table 1(c) present the numbers of SVDs and CDs. As the height of the Poisson peak rises, the CD iterations have a reduced impact on convergence. However, note that on several occassions iteration with only one CD increased the number of SVDs required. Allowing more than 2 CD steps only seemed to increase the overall run time; the number of SVDs was reduced in only a few instances. In addi-

UNIX is a trademark of AT&T Bell Laboratories

Table 1(a). Poisson Run Times

	n = 50				n = 100			
peak	c=0	c=1	c=2	c=10	c=0	c=1	c=2	c=10
1.5	134	120	94	103	974	848	885	904
2	163	150	130	141	950	834	880	933
2.5	134	148	126	134	1149	1051	1098	932
3	132	148	125	138	759	824	659	718
4	159	178	155	142	956	1048	882	967
5	158	180	157	144	955	1069	1100	988
6	131	173	155	120	970	1244	915	1006
7	133	159	127	161	938	1038	873	970
8	131	175	157	141	939	1053	1105	1043
9	135	178	158	144	955	1280	1138	1026
10	157	204	188	174	1129	1245	1106	1371
15	134	180	187	181	941	1252	1109	762
20	158	207	189	175	962	1276	1131	1143

Table 1(b). Poisson Decomposition Count (SVD.CD)

	n = 50				n = 100			
peak	c=0	c=1	c=2	c=10	c=0	c=1	c=2	c=10
1.5	5.0	4.4	3.5	3.10	5.0	4.4	4.6	4.10
2	6.1	5.6	4.8	4.12	5.0	4.4	4.7	4.12
2.5	5.0	5.5	4.7	4.12	6.0	5.5	5.8	4.11
3	5.0	5.5	4.7	4.13	4.0	4.4	3.6	3.11
4	6.0	6.6	5.8	4.15	5.0	5.5	4.7	4.13
5	6.0	6.7	5.9	4.15	5.0	5.6	5.8	4.14
6	5.0	6.6	5.9	3.16	5.1	6.6	4.9	4.15
7	5.0	5.5	4.7	4.19	5.0	5.5	4.7	4.14
8	5.0	6.6	5.9	4.14	5.0	5.6	5.9	4.17
9	5.1	6.6	5.9	4.15	5.0	6.7	5.9	4.16
10	6.0	7.7	6.10	5.16	6.0	6.6	5.9	5.23
15	5.1	6.7	6.10	5.18	5.0	6.6	5.9	3.13
20	6.0	7.7	6.10	5.16	5.0	6.6	5.9	4.19

Table 2(a). Binomial Run Times

		n = 50				n = 100			
size	prob	c=0	c=1	c=2	c=10	c=0	c=1	c=2	c=10
10	.3	108	87	90	91	943	827	671	692
	.2	106	118	125	131	970	829	858	882
	.1	133	118	92	97	968	860	885	937
	.05	135	148	130	135	1171	1064	898	977
	.01	-	-	-	-	1166	1046	1097	935
20	.3	109	91	92	96	743	604	632	635
	.2	137	119	123	127	760	617	636	645
	.1	109	120	124	129	780	838	650	680
	.05	165	151	159	168	795	849	681	742
	.01	-	-	-	-	1351	1261	1103	1225
	.005	-	-	-	-	1513	1676	1536	1683

Table 2(b). Binomial Decomposition Count (SVD.CD)

		n = 50				n = 100			
size	prob	c=0	c=1	c=2	c=10	c=0	c=1	c=2	c=10
10	.3	5.0	4.4	3.5	3.8	5.0	4.4	3.6	3.8
	.2	4.0	4.4	4.6	4.9	5.0	4.4	4.6	4.8
	.1	4.0	3.3	3.5	3.6	5.0	4.5	4.7	4.11
	.05	5.0	5.5	4.8	4.11	6.0	5.5	4.7	4.12
	.01	-	-	-	-	6.0	5.5	5.8	4.11
20	.3	4.0	4.4	4.6	4.9	4.0	3.3	3.5	3.6
	.2	5.1	4.4	4.6	4.8	4.1	3.3	3.5	3.7
	.1	4.1	3.4	3.5	3.7	4.1	4.4	3.5	3.8
	.05	6.0	5.5	5.8	5.12	4.0	4.4	3.6	3.10
	.01	-	-	-	-	7.0	6.6	5.8	5.14
	.005	-	-	-	-	8.0	8.8	7.11	7.20

tion, a few simulations, not shown here, converged when up to 2 CDs per SVD were allowed, but did not converge when 0 or up to 10 were allowed. Similar statements can be made about the binomial simulations (Table 2).

Since we know that the estimates converge for fixed λ (O'Sullivan, Yandell, and Raynor, Jr., 1986), a few iterations for fixed λ may guard against nonlinearity in the penalized likelihood. It is not known at this time what conditions are required on the penalized likelihood, as a function of λ, to insure convergence in the SVD-only approach.

Acknowledgements

This research has been supported in part by United States Department of Agriculture CSRS grant 511-100, and National Sciences Foundation grant DMS-8404970. Computing was performed on the UW-Madison Statistics VAX 11/750 Research Computer.

Bibliography

Aronszajn, N. (1950), "Theory of Reproducing Kernels," *Trans. Am. Math. Soc.*, 68, 337-404.

Bates, D. M., Lindstrom, M. J., Wahba, G., and Yandell, B. S. (1987), "GCVPACK - Routines for Generalized Cross Validation," *Communications in Statistics - Simulation and Computation*, 16, 263-297. (Algorithms Section)

Buja, A., Hastie, T., and Tibshirani, R. (1987) "Linear Smoothers and Additive Models." Technical Report#50, April 1987, AT&T Bell Laboratories.

Cox, D. D., and Koh, E. (1986) "A Smoothing Spline Based Test of Model Adequacy in Polynomial Regression." Technical Report#787, Dept. of Statistics, U. of Wisconsin.

Cox, D. D., Koh, E., Wahba, G., and Yandell, B. S. (1986) "Testing the (Parametric) Null Model Hypothesis in (Semiparametric) Partial and Generalized Spline Models." Technical Report#790, Dept. of Statistics, U. of Wisconsin.

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," *Numerische Mathematik*, 31, 377-403.

Dongarra, J. J., Bunch, J. R., Moler, C. B., and Stewart, G. W. (1979), *Linpack Users' Guide*, Philadelphia: SIAM.

Elden, L. (1984), "A Note on the Computation of the Generalized Cross-Validation Function for Ill-Conditioned Least Squares Problems," *BIT*, 24, 467-472.

Eubank, R. L. (1984), "The Hat Matrix for Smoothing Splines," *Statist. and Prob. Letters*, 2, 9-14.

Eubank, R. L. (1985), "Diagnostics for Smoothing Splines," *Journal of the Royal Statistical Society, Ser. B*, 47, 332-341.

Friedman, J. H., and Stuetzle, W. (1981), "Projection Pursuit Regression," *Journal of the American Statistical Association*, 76, 817-823.

Golub, G. H., Heath, M., and Wahba, G. (1979), "Generalised Cross Validation as a Method for Choosing a Good Ridge Parameter," *Technometrics*, 21, 215-224.

Green, P. J. (1984), "Iteratively Reweighted Least Squares for Maximum Likelihood Estimation and Some Robust and Resistant Alternatives (with Discussion)," *Journal of the Royal Statistical Society, Ser. B*, 46, 149-192.

Green, P. J. (1985) "Penalized Likelihood for General Semi-Parametric Regression Models." Technical Report#2819, Math. Research Center, U. of Wisconsin.

Green, P. J., and Yandell, B. S. (1985), "Semi-Parametric Generalized Linear Models," in *GLIM85: Proceedings of the International Conference on Generalized Linear Models, September 1985*, ed. R. Gilchrist Lecture Notes in Statistics, vol. 32, Springer-Verlag.

Härdle, W., and Marron, J.S. (1985a), "Asymptotic Nonequivalence of Some Bandwidth Selectors in Nonparametric Regression," *Biometrika*, 72, 481-484.

Härdle, W., and Marron, J.S. (1985b), "Optimal Bandwidth Selection in Nonparametric Regression Function Estimation," *Annals of Statistics*, 13, 1465-1481.

Hastie, T.J., and Tibshirani, R.J. (1986), "Generalized Additive Models," *Statist. Science*, 1, 297-318. (with discussion)

O'Sullivan, F. (1985), "Contribution to the Discussion of the Paper by Silverman," *Journal of the Royal Statistical Society, Ser. B*, 47, 39-40.

O'Sullivan, F., Yandell, B.S., and Raynor, Jr., W.J. (1986), "Automatic Smoothing of Regression Functions in Generalized Linear Models," *Journal of the American Statistical Association*, 81, 96-103.

Pregibon, D. (1981), "Logistic Regression Diagnostics," *Annals of Statistics*, 9, 705-724.

Pregibon, D. (1982), "Score Tests in GLIM," in *Proc. GLIM82 Conf.*, ed. R. Gilchrist, New York: Springer-Verlag.

Reinsch, C.H. (1967), "Smoothing by Spline Functions," *Numer. Math.*, 10, 177-183.

Rice, J.R. (1984), "Bandwidth Choice for Nonparametric Regression," *Annals of Statistics*, 12, 1215-1230.

Yandell, B.S. (1985), "Graphical Analysis of Proportional Poisson Rates," in *Proceedings of the 17th Symposium on the Interface, 17-19 March 1985, Lexington*, ed. D.M. Allen, New York: North Holland, 283-287.

Yandell, B.S., and Green, P.J. (1986) "Semi-Parametric Generalized Linear Model Diagnostics." *Proceedings of the Statistical Computing Section, ASA, Chicago*, 48-53.

Yandell, B.S. (1987) "Block Diagonal Smoothing Splines." Technical Report#812, Dept. of Statistics, U. of Wisconsin.

Yandell, B.S., and Hogg, D.B. (1987) "Determining Egg-Laying Rates of an Insect Using Splines." Technical Report#807, Dept. of Statistics, U. of Wisconsin.

Comments for Driver Routine

```
      subroutine dpglm(des,lddes,cov,ldcov,node,ldnode,resp,ldresp,ivec,
     * adiag,avar,pred,lamlim,dvec,coef,svals,tbl,ldtbl,auxtbl,score,
     * ldscor,stest,work,lwa,iwork,liwa,job,info)
      integer lddes,ldcov,ldnode,ldresp,ivec(13),ldtbl,ldscor,lwa,
     * liwa,iwork(liwa),job,info
      double precision des(lddes,*),cov(ldcov,*),node(ldnode,*),
     * resp(ldresp,2),adiag(*),avar(*),pred(*),lamlim(2),dvec(8),
     * coef(*),svals(*),tbl(ldtbl,3),auxtbl(3,3),score(ldscor,*),
     * stest(*),work(lwa)
c
c Purpose: determine the generalized cross validation estimate of the
c      smoothing parameter and fit model parameters for a penalized
c      general linear model.
c
c On Entry:
c   des(lddes,dim)  design for the variables to be splined
c   lddes           leading dimension of des as declared in the
c                   calling program
c   cov(ldcov,ncov1+ncov2) design for the covariates
c                   first ncov1 columns contain covariates which
c                     duplicate the replication structure of des
c                   next ncov2 columns contain covariates which
c                     do not duplicate the replication structure of
c                     des
c   ldcov           leading dimension of cov as declared in the
c                   calling program
c   node(ldnode,dim+ncov1) nodes for basis splines
c                   or unique design points and covariates
c   ldnode          leading dimension of node as declared in the
c                   calling program
c   resp(ldresp,2)  response vector, weight vector
c   ldresp          leading dimension of resp as declared in the
c                   calling program
c   ivec(13)        contains:
c                    1 nobs   number of observations
c                    2 dim   number of columns in des
c                    3 m     order of the derivatives in penalty
c                    4 ncov1  number of covariates which duplicate
c                            the replication structure of des
c                            = 0 if using basis splines
c                    5 ncov2  number of covariates which do not
c                            duplicate the replication structure
c                            of des
c                    6 nuobs  number of nodes if using basis splines
c                            or number of unique design points
c                    7 ntbl        number of evenly spaced values for
c                            ln(nobs*lambda) to be used in the
c                            initial grid search for lambda hat
c                            if ntbl = 0 only a golden ratio search
c                            will be done and tbl is not referenced
c                            if ntbl > 0 tbl will have ntbl rows
c                    8-10    (not used)
c                   11 maxin  maximum iterations for inner loop
c                   12 maxout maximum iterations for outer loop
c                   13 nscor  number of covariables for score test
c   adiag(nobs)           "true" y values on entry if predictive mse is
c                   requested
c   lamlim(2)       limits on lambda hat search if user input limits
```

```
c                   are requested
c                   if lamlim(1) = lamlim(2) then lamhat is set to
c                   lamlim(1)
c   dvec(8)         contains:
c                   1  lambda   first guess for lamhat
c                              (required in except for normal job)
c                              = -1.0 for lambda = infinity
c                   2-6        returned on exit
c                   7  mintol   minimum tolerence for convergence
c                   8  minlam   minimum difference in log(lamhat)
c   ldtbl           leading dimension of tbl as declared in the
c                   calling program
c   score(ldscor,nscor)    covariables for score test (destroyed)
c   ldscor          leading dimension of score
c   job                    integer with decimal expansion abcdef
c                      if a is nonzero then compute predictive mse
c                         using adiag as true y
c                      if b is nonzero then user input limits on
c                         search for min lambda hat are used
c                      if c is odd then diagonal of the hat
c                         matrix is calculated
c                         c > 1 compute variance of (beta:alpha)
c                      if d is nonzero then use svd
c                         d = 0 only use cholesky (fixed lambda)
c                      if e = 0 use linear model (normal)
c                         e = 1 use logistic model (binomial)
c                         e = 2 use log-linear model (poisson)
c                      if f is nonzero then pred already initialized
c                         f = 0 then initialize pred
c
c On Exit:
c   des(lddes,dim)  unique rows of des
c   pred(nobs)             predicted values
c   adiag(nobs)            diagonal elements of the hat matrix if requested
c   avar((nnull*(nnull+1)/2))            covariance (alpha:beta)
c   lamlim(2)       limits on lambda hat search
c   dvec(8)         contains:
c                   1  lamhat   generalized cross validation
c                              estimate of the smoothing parameter
c                   2  penlty   smoothing penalty
c                   3  rss      residual sum of squares
c                   4  tr(I-A)  trace of I - A
c                   5  like          log likelihood
c                   6  obj     like+lamhat*penlty
c                   7  gcv     generalized cross validation
c                   8  test     test of lambda = infinity
c   ivec(13)        contains:
c                   1-7        (same as on entry)
c                   8  npsing   number of positive singular values
c                   9  npar          number of parameters
c                              (npar = nuobs + nnull)
c                   10 nnull    size of the null space of sigma
c                   11 itin     iterations for inner loop
c                   12 itout    iterations for outer loop
c                   13         (same as on entry)
c   coef(npar)             coefficient estimates [alpha:beta:delta]
c                   coef must have a dimension of at least
c                   nuobs+nnull+ncov2
c   svals(npsing)   singular values
c                   svals must have a dimension of at least
c                   nuobs-nnull
c   tbl(ldtbl,3)    column contains
c                     1    grid of ln(nobs*lambda)
c                     2    V(lambda)
```

```
c                     3     R(lambda) if requested
c   auxtbl(3,3)             auxiliary table
c                   1st row contains:
c                       ln(nobs*lamhat), V(lamhat) and  R(lamhat) if
c                       requested
c                       where lamhat is the gcv estimate of lambda
c                   2nd row contains:
c                       0, V(0) and  R(0) if requested
c                   3rd row contains:
c                       0, V(infinity) and R(infinity) if requested
c   stest(1+nscor)  overall and single score test if nscor > 0
c   info            error indicator
c                      0 : successful completion
c                     -1 : ln(n*lambda hat) <= lamlim(1) (not fatal)
c                     -2 : ln(n*lambda hat) >= lamlim(2) (not fatal)
c                      1 : dimension error
c                      2 : lwa (length of work) is too small
c                      3 : liwa (length of iwork) is too small
c                      4 : error in dmaket (in dmksx)
c                      5 : sigma is rank deficient (in dsgdc)
c                      6 : R is singular (dtrsl error in dxeqr)
c                      7 : ldcaux (length of dcaux) is too small
c                      8 : error in ntbl
c                      9 : lamlim(1) > lamlim(2)
c                     10 : weight (resp(,2)) is zero
c                     20 < info <  30 : 20 + nonzero info from chol
c                     30 < info <  40 : 30 + nonzero info from dzdc
c                    100 < info < 140 : 100 + nonzero info from dgcv
c                    200 < info < 210 : 200 + nonzero info from dchst
c
c Working Storage:
c   work(lwa)       double precision work vector
c   lwa                    length of work as declared in the calling
c                   program must be at least lwa1 + lwa2 where
c                         lwa1 = nctsl*(nuobs+1) + nnull*(nnull-1)
c                                + 2*npar*(npar-nnull+1)
c                                + npar*(nobs+max(nobs,npar))
c                         lwa2 = (npar-nnull)*(nobs-nnull) - nnull
c                                + 2*(nobs+npar)
c   iwork(liwa)            integer work vector
c   liwa            length of the iwork as declared in the calling
c                   program
c                   must be at least 3 * nobs - nctsl
c
c Subprograms Called Directly:
c      Pglmpack - dbin dmksx dnorm dnrfs dpois
c      Gcvpack  - dcrtz dsgdc
c      Linpack  - dqrsl
c      Blas     - dcopy
c      Other   - dset
c
c Subprograms Called Indirectly:
c      Pglmpack - dcdiag dcheck dchrr dchst dmodel(=dbin,dnorm,dpois)
c                dsvst dvar
c      Gcvpack  - dctsx dcfcr ddiag dgcv dmakek dmaket dpdcr dpmse
c                drsap dtsvdc dvl dvlop dvmin dzdc mkpoly
c      Linpack  - dchdc dposl dqrdc dqrsl dsvdc dtrco dtrsl
c      Blas     - daxpy dasum dcopy ddot dgemv dnrm2 dscal dswap
c      Other   - dcpmut dprmut dset dftkf fact pcheck
c
```

Received by Editorial Board member 1987; Approved October, 1987.

Recommended by William J. Kennedy, Iowa State University.