Feature Article:

Maximum Likelihood Estimation in the New Computing Environment

Linda Williams Pickle
National Center for Health Statistics

1. Introduction

Many articles have been published recently celebrating the tenth anniversary of the IBM PC. Just as the personal computer (PC) transferred computational power from the computer room to the desktop, so has the subsequent explosion in hardware and software development empowered the statistician with new analytic tools. The increase in speed more than 30-fold during the 1980s has encouraged the development of computationally-intensive statistical methods and interactive graphical tools not duplicated on the mainframe.

Although some would argue that classical statistical methods are now outdated, parametric methods are still needed for statistical inference. New methods such as robust regression and exploratory graphics in fact are a welcome addition to the statistical modeler's toolbag, as an aid in model selection and verification.

Because of these recent computational advances, we may now estimate parameters from models heretofore considered too complex for everyday analysis, such as generalized additive models (Efron and Tibshirani, 1991), overdispersed Poisson models (Efron 1986), and mixtures of distributions. The same estimation methods can also be used to estimate parameters resulting from various smoothing algorithms (e.g., Thisted, Chapter 6, 1988) or any other function minimization. In this article, we summarize generalized optimization methods appropriate for maximum likelihood estimation when analytic solutions and model-specific software are unavailable, and offer the reader practical advice on the selection and use of these methods.

2. Statistical software development during the 1980s

Ten years ago, the data analyst had but a few generalized statistical packages available on the mainframe computer; the use of "nonstandard" methods required customized programming in Fortran. Today, these and many other statistical packages have been written for the PC, including software to compute maximum likelihood estimates (MLEs) for particular nonlinear models (e.g., logistic). Many packages now include routines for general nonlinear parameter estimation. For more complex problems, new symbolic mathematics and matrix programming languages permit programming nearly at the conceptual level, so that customized programs are quick to develop and less error-prone. All of the acceptable optimization algorithms are publicly available as pseudo-code (Dennis and Schnabel 1983) or in Fortran, PASCAL, C and Basic (e.g., Boisvert et al. 1990; Kahaner et al. 1989, Press et al. 1986).

During this same time advances in numerical analysis led to an increase in the number of viable optimization algorithms and implementation options. In the past, many statisticians were not trained in numerical methods and so may have difficulty choosing from the confusing array of optimization methods. This article will provide information to help the statistician take advantage of these powerful analytic tools.

3. Requirements for statistical modeling

The goal of the type of analysis considered here is to identify a parametric model that adequately represents the probability distribution of a random variable (Y) and to estimate the effects of potential explanatory covariates (X) on its prediction. In general, parametric modeling requires methods for:

- selecting the form of the model,
- estimating model parameters,
- model verification and reduction.

Problems resulting from misspecification of the model have been extensively covered in the regression literature (e.g., Bates and Watts 1988). However, the virtual explosion of statistical graphics in the past 10 years has provided data analysts with new graphical tools for model selection and verification, such as smoothed density plots, scatter plot matrices, and 3-D point cloud rotations. Visualization of higher dimensional space via projections and motion graphics will be a welcome tool in the future (e.g., Miller and Wegman, 1989; Hurley and Buja, 1990).

The remainder of this article will focus on parameter estimation for nonlinear models, for which analytic solutions don't often exist, although the above requirements also hold for linear models.

4. Estimating model parameters

4.1 Background

The most commonly used methods of parameter estimation are least squares and maximum likelihood estimation. For simplicity, discussion will be limited to univariate, fixed-effects models; i.e., let $f(Y;X,B)$ denote the probability density of the random variable Y, given the covariate vector $X=(X_1,X_2,\ldots,X_I)'$ and parameter vector $B=(B_1,B_2,\ldots,B_I)'$, where $E(Y|X)=g(X,B)$, some continuous function of the covariates and parameters. Stated in terms of a general optimization problem, the goal for each estimation method is to identify the values of the vector B which minimize a function, called the objective function, given the observed data $\{Y_j\}$. For the (unweighted) least squares method, we seek to minimize the sum of squared errors, i.e.,

$$SSE=\sum_{j=1}^{n}[Y_j-g(X_j,B)]^2 \qquad (1)$$

whereas the MLE $\hat{B}$ results from the maximization of the likelihood, or equivalently minimizing the negative of the logarithm of the likelihood, of the observed data,

$$L=-l(B;X,Y)=-\sum_{j=1}^{n}\ln[f(Y_j;X_j,B)] \qquad (2)$$

where $f(Y_j;X_j,B)$ is the probability density function of Y_j. Several statistical software packages ask the user to define a "loss" function, which is just the individual term of the objective function to be summed over the observations. For maximum likelihood estimation, the loss is the negative of the log likelihood function. A penalized log likelihood may be constructed as

$$L+(\text{penalty})*(\text{CONDITION}),$$

where CONDITION is 1 if true, 0 if false.

Assuming that $g(X_j,B)$ and $f(Y_j;X_j,B)$ are twice continuously differentiable and that $f(Y_j;X_j,B)>0$ for all Y_j, then the optimal parameter estimator $\hat{B}$ will be the solution to the following differential equations:

$$\frac{\partial SSE}{\partial B_i}=\sum_{j=1}^{n}2*[Y_j-g(X_j,B)]\left[-\frac{\partial g(X_j,B)}{\partial B_i}\right]=0 \qquad (3)$$

or

$$\frac{\partial L}{\partial B_i}=-\sum_{j=1}^{n}\frac{1}{f(Y_j;X_j,B)}\left[\frac{\partial f(Y_j;X_j,B)}{\partial B_i}\right]=0 \qquad (4)$$

for $i=1,\ldots,I$, for the least squares or maximum likelihood method, respectively.

Analytic solutions exist to the least squares equations (3) when the expected value of Y is linear in the parameters, i.e., when $g(X_j,B)=X'B$. Even in this simple situation, maximum likelihood estimation is a nonlinear optimization problem because of the usual nonlinearity of the density function $f(Y;X,B)$. However, for some distributions, including members of the exponential family, (4) simplifies to provide analytic solutions if the model is linear in the parameters. For all other cases an iterative algorithm is required to solve for the optimal parameter vector by either least squares (3) or maximum likelihood estimation (4). We will confine the following discussion of optimization methods to maximum likelihood estimation.

4.2 Basics of optimization

Many years ago when optimization methods were not readily available to the data analyst, MLEs were more often computed by solving the system of nonlinear equations (4), rather than direct optimization of equation (2). In difficult

cases, linearizing or other simplifying approximations were substituted into (4) to make the solution computationally tractable. Now that we have greater computational power, why are the more complex optimization algorithms preferable?

First, without further checking of the second derivatives of the log likelihood, we can't tell whether the solution to the nonlinear system of equations (4) is a minimum, maximum, or saddle point. Also, optimization algorithms are generally faster than are those for solving a nonlinear system of equations. Finally, if the second derivative (Hessian) matrix is not directly available, some optimization algorithms can provide an approximation to it. This is critical for maximum likelihood estimation, as the negative of the inverse Hessian matrix evaluated at the MLE parameter vector is the estimated asymptotic variance matrix for the parameters.

Most generalized optimization programs can handle constraints on the parameters. As long as the solution $\hat{B}$ is not on a boundary of the parameter space, $\hat{B}$ is still the MLE of B. Constraints are often necessary to prevent the algorithm from diverging into an invalid parameter subspace or to ensure that the parameters are uniquely specified. If the available software does not permit constraints, we may optimize a penalized log likelihood function, constructed by adding a very large number to the objective function when the parameter is out of the permissible range. The need for parameter constraints is frequent enough that software should be chosen that will permit either direct constraints or conditional penalties.

4.3 Newton-type algorithms

Newton's method: Newton's method for unconstrained optimization was derived by considering the iterative process that would solve the quadratic Taylor's series expansion approximation of the true objective function. Thus Newton's method will converge to the true MLE in a single iteration if the log likelihood function is quadratic. The method provides an update of the estimated MLE, denoted B^{k+1}, as:

$$B^{k+1} = B^k - (H^{-1})^k S^k \qquad (5)$$

where

$H = \nabla^2 L$, the (Hessian) matrix of second partial derivatives of L with respect to the parameters B_i,

$S = \nabla L$, the (gradient) vector of first partial derivatives,

L = the negative of the log likelihood,

and the superscript k indicates evaluation at the previous parameter estimate B^k. Since $-H^{-1}$ evaluated at $\hat{B}$ is the asymptotic variance matrix for $\hat{B}$, denoted Σ, it may be more intuitive to rewrite (5) as:

$$B^{k+1} = B^k + \Sigma^k S^k \ . \qquad (6)$$

The primary advantage of Newton's method is its fast convergence when the initial parameter estimates are near the solution. Unfortunately, the method has several disadvantages that make it impractical for many problems:

- It may converge to a maximum or saddle point rather than a minimum,
- the analytic first and second derivatives of the log likelihood must be provided,
- the inverse of the Hessian matrix must be computed at each iteration, and
- if the Hessian is ill-conditioned, the method may not converge at all.

Simple variations of the Newton method to avoid these problems include the addition of a positive constant to the H diagonal to ensure that it is a positive definite matrix, the use of numerical rather than analytic second derivatives, and the substitution of a generalized inverse for H^{-1}.

Several variations of the Newton method were developed specifically for least squares optimization. These methods, e.g., Gauss-Newton and Levenberg-Marquardt, are most efficient for minimizing the sum of squared errors (1). Use of these algorithms to maximize the log likelihood will be inefficient unless the log likelihood has a quadratic form.

Algorithms representative of classes of methods are described below. Detailed descriptions of these algorithms may be found in McCormick (1983), Dennis and Schnabel (1983), Kahaner et al (1988), and Thisted (1988).

In order to discuss modifications to the Newton method, it is helpful to write (5) in a more general way:

$$B^{k+1} = B^k - \lambda^k \Delta^k \qquad (7)$$

where λ^k is the length and Δ^k is the direction of the "step" from B^k to B^{k+1}. For the pure Newton method (5), λ^k is always equal to 1.

Step size: One variation of the Newton algorithm is the reduction of the step length λ^k to a value > 0 for which L is improved (i.e., lowered). Theoretically we would like to choose the value of λ^k that minimizes L^{k+1} for a change in B^k in the Δ^k direction. However, the choice of step size at each iteration is itself an optimization problem with the conflicting goals of minimizing the computational time to solve for the step size while minimizing the number of iterations required for convergence by choosing the optimal step size. Much work has been done on this problem, also referred to as the line search method. The simplest solution is to halve the step size, beginning with size 1, until $L^{k+1} < L^k$, but the use of a Fibonacci series of step size reductions provides faster convergence of B^k for nearly the same computational cost. The Golden Section Search method, a variant of the Fibonacci method, is faster yet and guaranteed to find the optimum step length. Cubic interpolation has also proven successful for step size optimization and, for likelihoods where ∇L^{k+1} can be computed at little cost while computing L^{k+1}, can be easily used for each iteration (Dennis and Schnabel, 1983). A step size search is sometimes incorporated into a Newton algorithm, but is nearly always included for other methods. In general, a reduced step size is required when L is not well approximated by a quadratic approximation at B^{k+1}.

Descent methods: Descent algorithms, also called Cauchy or gradient algorithms, require specification of ∇L^k, the first partial derivative vector, but not of the Hessian matrix. Algorithms in this class force the step from B^k "downhill" on the negative log likelihood surface toward $\hat{B}$, with step size selected as described above. The method of steepest descent steps in the direction $\Delta^k = \nabla L^k$. Although these methods will move B^k in the direction of at least a local minimizer, they do so slowly and the rate of convergence is sensitive to the scaling of the parameter vector B. With improved computational speed, a more complex quasi-Newton algorithm is now preferred.

4.4 Quasi-Newton methods

A quasi-Newton algorithm resembles the true Newton method (5), but with an approximated inverse Hessian matrix (M) and a computed step size. That is, the Newton step direction is approximated by $\Delta^k = (M^k)\nabla L^k$. Secant methods estimate M^k by using a linear approximation to L^k based on the last few estimates of B (McCormick 1983). A newer class of inverse Hessian approximations was developed, beginning with Davidon (1959), that saves information from all previous iterations. These algorithms update M^k by a low-rank matrix, thus saving the computational effort required to approximate M^k fully at each iteration. After convergence, M provides a direct estimate of the asymptotic variance matrix of $\hat{B}$ without inversion. In addition, if the initial estimate M^0 is positive definite, subsequent M^ks are also positive definite, guaranteeing a proper descent direction for each step. These advantages, along with speed of convergence, make these newer algorithms (e.g., Davidon-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS)) the method of choice.

Quasi-Newton methods may be modified to approximate the first derivative vector as well as the inverse Hessian. However, a poor estimate of the gradient vector will affect the accuracy of the final parameter estimate, more so than a poor Hessian approximation (Kahaner et al 1989).

4.5 Trust regions

The classes of algorithms discussed so far have all moved the parameter estimates B^k in the direction exactly or approximately defined by the Newton method. Trust region methods, developed during the 1970's, relax this requirement and modify both the direction and step length. First, a hyper-dimensional parameter subspace is defined, centered on B^k, within which we "trust" L is well approximated by a quadratic function. Using this region to bound the step length, a step in a descent direction is chosen. The double dogleg method, an efficient strategy for choosing the direction when the derivative vector values are easily computed,

yields a direction between those of the steepest descent and quasi-Newton methods (Dennis and Schnabel, 1983).

4.6 Direct search methods

When all else fails, a brute force grid search will work, albeit slowly, and does not require any derivative evaluations. The simplex method (Spendley et al, 1962) is one of the more efficient direct search methods. To solve for an I-dimensional parameter vector B, a geometric simplex is first formed from $I+1$ points in I-dimensional space. Then the values of L at each point are computed and used to adjust the size and location of the simplex for the next iteration (Everitt 1987). This is usually much less work than the r^I evaluations of L required for a crude grid search of r values for each of I parameters. Although slower than the Newton or quasi-Newton methods, this method is robust and may converge for complex likelihoods when others methods fail.

4.7 Comparison of iterative methods

Convergence rates: The primary measure used to compare optimization algorithms has been the theoretical convergence rate, defined as the power p such that $\lim_{k\to\infty} \|B^{k+1}-\hat{B}\|/\|B^k-\hat{B}\|^p = C$, a constant. If $p=2$, as for the Newton method under certain regularity conditions, the rate of convergence of the algorithm is said to be quadratic, an approximate doubling of the number of accurate significant digits with each iteration. The linear ($p=1$) convergence of descent methods is considered slow, compared to the super-linear ($p=1.62$) convergence of quasi-Newton methods.

It is important to keep in mind that this theoretical convergence rate provides a comparison of the number of iterations, not of the total computational time, that different algorithms require to solve the same problem. With time per iteration now only a few seconds for moderately-sized MLE problems, it may often be worth a few extra iterations to use an algorithm that is more robust or easier to implement than other, slightly faster, methods. Kahaner et al. (1989) estimate the number of calculations required for several algorithmic components, but there are too many variations of implementation for each class of algorithms to estimate total time expected per iteration. Benchmark comparisons of actual convergence times for representative likelihoods using well tested implementations of algorithms would provide a more useful ranking of methods.

Accuracy of the solution: Unfortunately, even when it appears that convergence has occurred, the results can be incorrect. The apparent solution may be at a parameter boundary or it may be a local, but not global, MLE. Sufficient conditions for a vector B^k to be an unconstrained local MLE are that $\nabla L^k = 0$ (i.e., that B^k solves equations (4)) and that the Hessian matrix is positive definite (McCormick 1983). Furthermore, for a concave likelihood function, the solution vector will be the unique, global MLE. While the common densities generate nicely-behaved concave likelihoods, problems for which we need the power and flexibility of the methods described here are not always well behaved. Determining concavity and positive definiteness requires knowledge of the second partial derivatives. Unless symbolic mathematics software is available, these are often difficult to derive.

The data analyst's most useful tool in judging the accuracy of the solution is a 3-dimensional plot of the likelihood surface. A series of surface plots, varying two parameters at a time, can quickly point out problem areas such as ridges or multiple local maxima. The presence of "valleys" or "potholes" in the surface identifies a nonconcave likelihood, a warning that multiple local maxima may exist.

Estimation of variance matrix: The negative inverse Hessian matrix, evaluated at the MLE $\hat{B}$, is the asymptotic variance matrix (Σ) for $\hat{B}$. Because the optimization methods described here were first developed for deterministic models, accuracy of the solution vector, not Σ, was of concern. Only when the same algorithms began to be used for statistical models did the accuracy of the inverse Hessian become important. However, some algorithms may provide a poor estimate of Σ while others, notably the descent and direct search methods, do not provide an estimate at all. For example, the step direction for the Newton ($(H^{-1})^k S^k$) or quasi-Newton ($(M^k)\nabla L^k$) method can be optimal even when the factors H^{-1} or M are inaccurate due to inversion or update problems. Error may also be introduced by reporting the estimated Σ evaluated at B^k rather than at the final

MLE B^{k+1}. Many users report variance estimates provided by their software packages without an awareness of these potential errors.

Little work has been done to validate estimation methods for Σ. An examination of quasi-Newton Σ estimators showed at least linear convergence using two representative likelihoods (Pickle and McCormick 1988). This is slower than the superlinear convergence of quasi-Newton parameter estimates and suggests the need for stricter convergence criteria to ensure an accurate variance matrix estimate. Because it is difficult to judge when the matrix has converged, an alternative method is to compute the estimated variance matrix using numerical second derivatives evaluated at several of the final B^k iterates. With today's improved processor speed, this is a feasible method and has been shown to work well (Wilkinson and Fleury 1989).

If software to compute numerical derivatives is not available, the data analyst must be aware of the slower rate of convergence for the variance matrix and adjust the convergence tolerance or number of iterations accordingly. In no case should the number of iterations for an I-dimensional parameter vector be less than I when using quasi-Newton algorithms. This is the minimum number required to produce sufficient information for the inverse Hessian using this matrix updating method (McCormick 1983).

4.8 Recommended strategy: Choice of software and user control of the algorithm

The following is a checklist of options to look for in a generalized program for maximum likelihood estimation and recommended choices for these options.

Specification of the loss or objective function: The user should be able to specify that it is the negative log likelihood function that is to be minimized. Without this option, you are limited to a few common likelihood forms or the program is finding the least squares, rather than maximum likelihood, estimator.

Optional entry of derivatives: If the first and/or second partial derivatives of the log likelihood are easy to derive, the program should permit their use.

Choice of optimization method: The program should offer a direct search method as well as a more efficient method. A good combination is a quasi-Newton algorithm, such as BFGS or DFP, and the simplex method.

Choice of step size selection: The program should solve for the optimal step size, if required by the algorithm. Step halving is the most common method offered, but other methods such as the Golden Section Search are more efficient. This option is not as critical as the choice of the optimization algorithm.

Specification of convergence criteria: It should be possible to define convergence in terms of the change both in the parameter estimates and in the log likelihood function. If the log likelihood surface is relatively flat in one or more parameters, the objective function may converge earlier than the parameter estimates. Conversely, if the log likelihood surface is sharply peaked in one or more parameters, the range of "highly likely" parameter values may be narrow although the iterates have not quite converged to the maximum likelihood values.

Specification of the tolerance for convergence: The maximum change allowed in the parameters or log likelihood before convergence is declared should be a relative, not absolute, measure. A maximum relative change of 10% is very crude, 0.1% is probably reasonable for most problems.

Limiting the number of iterations: The software should allow the user to limit the number of iterations. Otherwise, a simple error in typing the loss function or initial values can be a time-consuming mistake. Also, iterating a few times using a direct search method can provide suitable initial values for a more efficient algorithm.

Bounding the parameter space: There should be some way to restrict the optimization search to an acceptable parameter range. Bounds may be required to conform to distributional assumptions or to avoid identifiability or computational problems, such as division by zero. If bounds cannot be set directly, it should be possible to add a penalty to the log likelihood instead.

Setting initial parameter estimates: The user should be able to specify initial parameter values. A poor choice here can result in divergence or

convergence to a nonglobal solution. Subject matter knowledge often will suggest plausible starting values or, if not, reasonable values can be determined from marginal density plots.

Computation of asymptotic variance matrix: The software must provide an estimated variance matrix for computation of parameter confidence limits or significance tests. Use of numerical derivatives using central differences based on a number of values near the MLE seems to be the preferred method (Thisted 1988). Although this approach adds to the total computational time required, it avoids the problem of defining convergence of the matrix. A measure of ill conditioning of the estimated matrix (e.g., the condition number or eigenvalues) would be useful, but these can be expensive computations for large matrices.

Plotting capabilities: Graphical software should be available to examine the data distribution and the association between Y and any covariates (X), and to plot the marginal and 3-D log likelihood functions. Simple density plots prior to modeling can suggest initial values and point to potentially troublesome parameters. If the marginal log likelihood for a parameter has a rather flat peak or the apparent maximum is near a boundary, consider a parameter transformation that will yield a more moderate curvature away from the boundaries. After convergence, a plot of the log likelihood surface for each pair of parameters can identify unforeseen problems or confirm that $\hat{B}$ is the MLE.

5. Example

To illustrate the strategy outlined above, data from a mixture of two normal distributions were simulated and then analyzed using commercially available software (SYSTAT; Wilkinson 1990) with DOS 5.0 on an AST 386/20 PC with a Cyrix 83D87 math coprocessor. A mixture of 200 randomly-generated values from a $N(1,1)$ and 100 values from a $N(3,1)$ distribution were analyzed (the *N11* and *N13* components, respectively). A Davidon-Fletcher-Powell quasi-Newton algorithm was used with numerically calculated first derivatives.

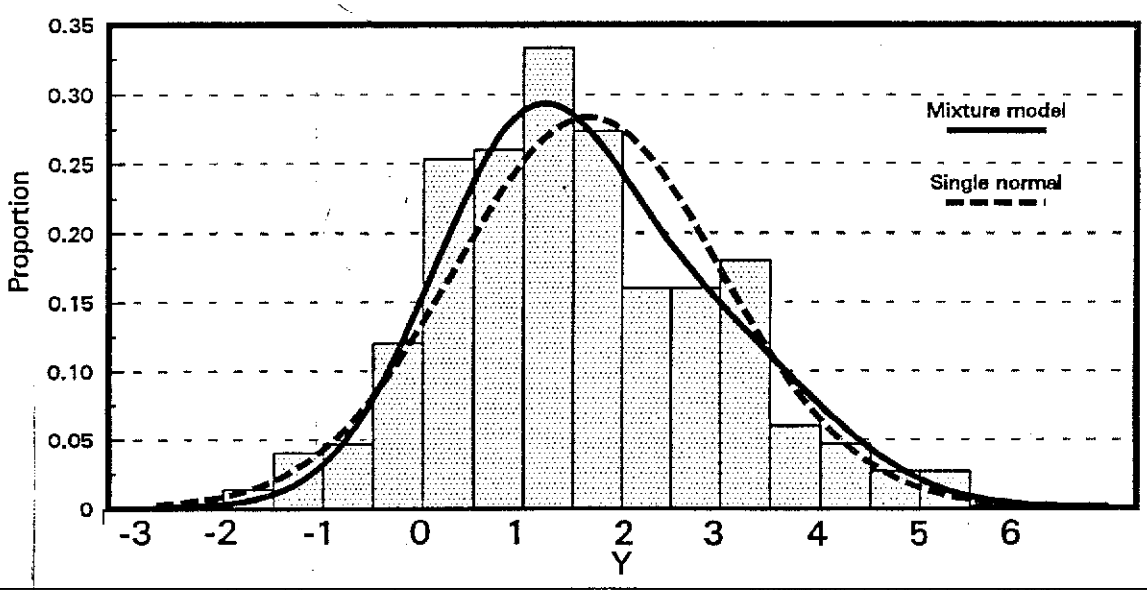
A histogram (Figure 1) suggested a mixture of at least 2 overlapping normal distributions. The negative log likelihood function (L) for the sum of 2 normal probability densities was specified as the loss. In syntax typical of statistical software,

$$\text{Loss} = -Ln((P/S1)*\text{Exp}(-((Y-M1)\text{^}2)/(2*S1\text{^}2)) + ((1-P)/S2)*\text{Exp}(-((Y-M2)\text{^}2)/(2*S2\text{^}2)))$$

where *M1* and *M2* are the means and *S1* and *S2* are the standard deviations of the two component distributions, with mixing parameter P, for the random variable Y. No covariates were included.

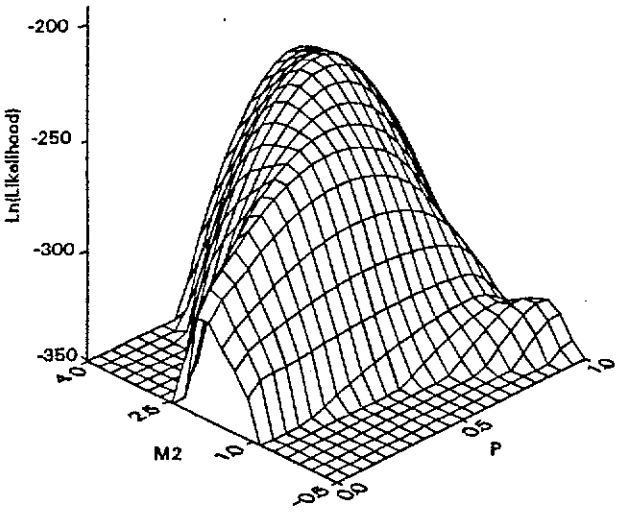
Results: The computed means and standard deviations of the simulated data were 0.87 and 0.94 for *N11* and 2.93 and 1.03 for *N13*, respectively. For $B = (P, M1, M2, S1, S2)'$ set to initial values $(0.5, 0.5, 5, 0.5, 0.5)'$, P quickly exceeded its limits. After adding a penalty ($+1E35*(P < 0.01$ or $P > 0.99)$) to the loss to constrain $0 < P < 1$, *S2*

Figure 1-Observed and Fitted Densities for Simulated Data



diverged and P neared its upper bound after 30 iterations. A 3-D plot of the log likelihood surface for $M2$ and P, with $M1$, $S1$, and S2 each set to 1.0, showed several local maxima (Figure 2). Looking at this surface from the $M2$ direction (Figure 3) showed a local maximum when $M2 < M1$. Restarting with a revised penalty ($+1E35*(P < 0.01$ or $P > 0.99$ or $M1 > M2)$) to ensure a unique solution, 10 simplex iterations ended with $B' = (0.49, 0.77, 2.33, 0.93, 1.30)$ and $L = 239.96$. Using these as initial values, the quasi-Newton algorithm then converged in 15 iterations to $B' = (0.68, 0.94, 2.87, 0.99, 1.11)$ with asymptotic standard errors = (0.28, 0.36, 0.98, 0.14, 0.36) and $L = 239.85$.

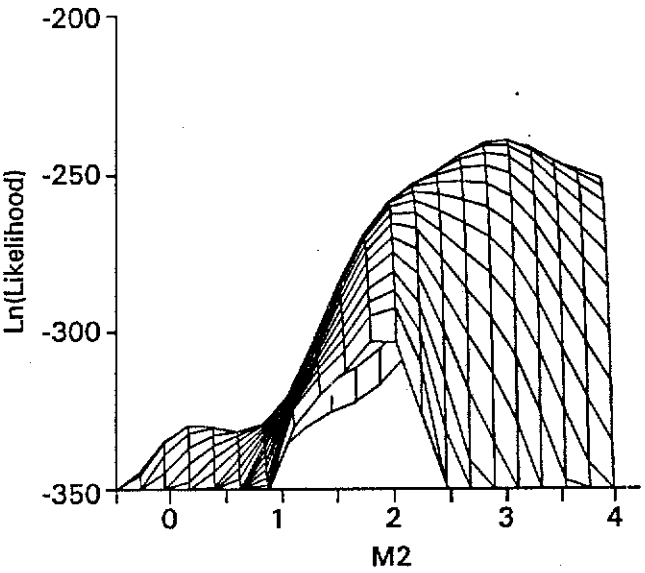
Figure 2 - Log likelihood surface: Example



The log likelihood surface has greater curvature for $M2$ than P near the maximum (Figures 3,4), with a ridge along the $M2-P$ direction because of the strong correlation of these parameters ($r = 0.98$). This is due to the poor power of this small dataset to distinguish among many nearly equally likely $M2-P$ combinations for the two overlapping normal components.

Fitting a single normal distribution to these data resulted in $\hat{B} = (M1, S1) = (1.56, 1.37)$ and $L = 244.2$. The likelihood ratio chi-square statistic ($2*(244.20-239.85) = 8.7$, 3 d.f.) indicates a significant improvement of the mixture model over the single normal model ($p = 0.03$); both fitted models are shown in Figure 1.

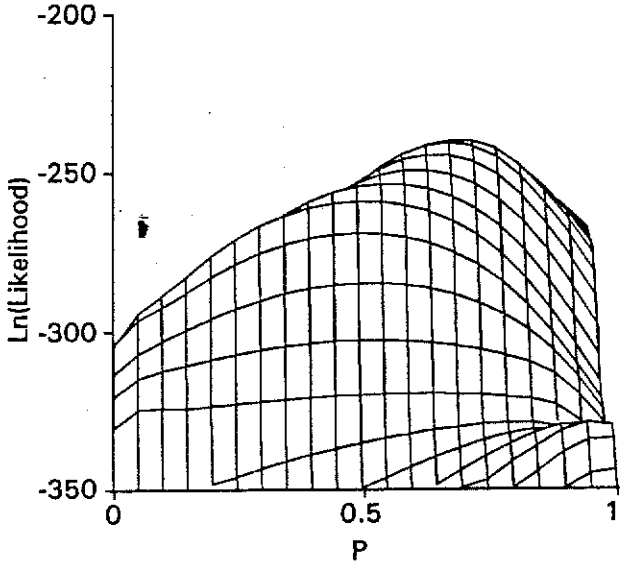
Figure 3 - Log likelihood from M2 perspective



This analysis was started as an unconstrained problem to illustrate problems with convergence and how to correct them. After adding a penalty to correct the identifiability problem, the simplex method was used to select initial values for the quasi-Newton algorithm. Although the parameter estimates appear far from $\hat{B}$ after the simplex iterations, the log likelihood value was near its maximum, where quasi-Newton methods converge quickly. The iterations averaged 44 seconds each for the simplex method and 20 seconds each for the quasi-Newton method. Although the mixture model is a significant improvement over the single normal model, asymptotic standard errors of the parameters and plots of the log likelihood surface show the difficulty of estimating the parameters precisely with this fairly small sample and high collinearity.

As a historical note, metabolic ratio data ($n = 245$) were reanalyzed in 1987 using a similar model (Caporaso et al. 1989). Iterations took several hours each on an 8088 PC without a math coprocessor, and 3 minutes each on a 386 PC with 80287 math chip. The original authors concluded that the metabolic rates were not genetically determined because the proportions of persons in categories defined by the histogram antimodes were inconsistent with theoretical predictions (Ayesh et al. 1984). However, recategorizing the rates using results of fitting a normal mixture model did in fact show that the data supported inherited metabolism.

Figure 4 - Log likelihood from P perspective



6. Summary and comments

Ten years ago, in order to compute MLEs for complex models, the data analyst chose between using one of a few mainframe statistical packages with limited optimization algorithms and options, or writing a customized Fortran program. These difficulties restricted the types of models considered by many analysts. Now we have a choice of many competing software packages for the PC, including higher-level and matrix-based programming languages, and new graphical tools. Not only can statisticians now easily estimate parameters from complex models, but estimates from several competing models can be quickly compared. Interactive computing environments currently being developed will bring together graphical, numerical, and statistical methods to facilitate complex data analysis in the future (Gale 1986).

Along with this newfound personal computational power comes the responsibility for the modeling strategy and choice of appropriate methods. Failure to accept this responsibility can lead to incorrect estimation of parameters, their variances, and as a result, incorrect inferences. Statisticians need to understand methods developed in other computational disciplines in order to make these decisions intelligently.

References

Ayesh, R., Idle, J., Ritchie, J.C., et al. (1984), "Metabolic oxidation phenotypes as markers for susceptibility to lung cancer," *Nature*,.312, 169–170.

Bates, D.M., and Watts, D.G. (1988), *Nonlinear Regression Analysis and Its Applications*, John Wiley and Sons, New York.

Boisvert, R.F., Howe, S.E., Kahaner, D.K., Springmann, J.L. (1990), *Guide to Available Mathematical Software*, NISTIR 90–4237, National Institute of Standards and Technology, March, 1990.

Caporaso, N., Pickle, L.W., Bale, S., et al (1989), "The distribution of debrisoquine metabolic phenotypes and implications for the suggested association with lung cancer risk," *Genetic Epidemiology* 6, 517–24.

Davidon, W.C. (1959), "Variable metric methods for minimization," *AEC Research and Development Report* ANL-5990, Argonne National Laboratory, IL.

Dennis, J.E., Jr. and Schnabel, R.B. (1983), *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

Efron, B. (1986), "Double exponential families and their use in generalized linear regression," *JASA* 81, 709–721.

Efron, B. and Tibshirani, R. (1991), "Statistical data analysis in the computer age," *Science* 253, 390–395.

Everitt, B.S. (1987), *Introduction to Optimization Methods and their Application in Statistics*, Chapman and Hall, London.

Gale, W.A. (ed.) (1986), *Artificial Intelligence & Statistics*, Addison-Wesley, Reading, MA.

Hurley, C. and Buja, A. (1990), "Analyzing high-dimensional data with motion graphics," *SIAM J. Sci. Stat. Comput.*, 11, 1193–1211.

Jezek, L.K. (1989), "Eenie, meenie, minie, mo: Choosing statistical software for your microcomputer," in K. Berk and L. Malone, (Eds.), *Computing Science and Statistics:*

Proceedings of the 21st Symposium on the Interface, 32-37, American Statistical Association, Alexandria, VA.

Kahaner, D., Moler, C., and Nash, S. (1989), *Numerical Methods and Software*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

McCormick, G.P. (1983), *Nonlinear Programming: Theory, Algorithms, and Applications*, John Wiley and Sons, New York.

Miller, J.J. and Wegman, E.J. (1989), "Construction of line densities for parallel coordinate plots," in K. Berk and L. Malone, (Eds.), *Computing Science and Statistics: Proceedings of the 21st Symposium on the Interface*, 191-199, American Statistical Association, Alexandria, VA.

Pickle, L.W. and McCormick, G.P. (1988), "Estimation of the variance matrix for maximum likelihood parameters using quasi-Newton methods," in E.J. Wegman, D.T. Gantz, and J.J. Miller, (Eds.), *Computer Science and Statistics: Proceedings of the 20th Symposium on the Interface*, 505-510, American Statistical Association, Alexandria, VA.

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. (1986), *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Cambridge.

Spendley, W., Hext, G.R., and Humsworth, F.R. (1962), "Sequential applications of simplex designs in optimization and evolutionary operation," *Technometrics* 4, 441–461.

Thisted, R.A. (1988), *Elements of Statistical Computing*, Chapman and Hall, New York.

Wilkinson, L. (1990), *SYSTAT: The System for Statistics*, SYSTAT, Inc., Evanston, IL.

Wilkinson, L. and Fleury, P. (1989), "Maximum likelihood estimation using SYSTAT," in K. Berk and L. Malone, (Eds.), *Computing Science and Statistics: Proceedings of the 21st Symposium on the Interface*, 402-404, American Statistical Association, Alexandria, VA.