A NONLINEAR REGRESSION PROCEDURE FOR EVOKE POTENTIAL DATA ANALYSIS(U) DESMATICS INC STATE COLLEGE PA C A MAURO JUN 85 TR-112-18 N00014-79-C-0128
A NONLINEAR REGRESSION PROCEDURE FOR EVOKED POTENTIAL DATA ANALYSIS

by

Carl A. Mauro

STATISTICS
OPERATIONS RESEARCH
MATHEMATICS

DESMATICS, INC.
P.O. Box 618
State College, PA 16804

DISTRIBUTION STATEMENT A
Approved for public release; Distribution Unlimited

8 5 0 7 0 1 1 2 3
A NONLINEAR REGRESSION
PROCEDURE FOR EVOKED
POTENTIAL DATA ANALYSIS

by

Carl A. Mauro

TECHNICAL REPORT NO. 112-18

June 1985

This study was supported by the Office of Naval Research under Contract No. N00014-79-C-0128

Reproduction in whole or in part is permitted for any purpose of the United States Government

Approved for public release; distribution unlimited
ABSTRACT

The recording and subsequent analysis of evoked potential activity has proven useful for the evaluation of neural dysfunction resulting from impact acceleration injury involving the head and neck. In animal impact acceleration experiments involving Rhesus monkeys, somatosensory evoked potentials showed an increase in latency following nonlethal experiments. In order to assess quantitatively and objectively the amplitude and duration of the latency effect following impact acceleration, a nonlinear mathematical model has been proposed. This technical report describes a nonlinear regression procedure for fitting the proposed model directly to empirical latency data. A FORTRAN computer program listing is provided.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT.</td>
<td>i</td>
</tr>
<tr>
<td>1. INTRODUCTION.</td>
<td>1</td>
</tr>
<tr>
<td>2. METHODS</td>
<td>5</td>
</tr>
<tr>
<td>3. EXPONENTIAL REGRESSION WITH AN ASYMPTOTE.</td>
<td>8</td>
</tr>
<tr>
<td>4. MINIMIZING $V(s,b)$</td>
<td>16</td>
</tr>
<tr>
<td>5. AN EXAMPLE</td>
<td>23</td>
</tr>
<tr>
<td>6. REFERENCES.</td>
<td>29</td>
</tr>
<tr>
<td>APPENDIX.</td>
<td>30</td>
</tr>
</tbody>
</table>
An important consequence of impact injury involving the head and neck is the disruption of normal functioning of the central nervous system. As part of its research effort on impact acceleration injury prevention, the Naval Biodynamics Laboratory (NBDL) has conducted a series of experiments designed to test the neurophysiological effects of indirect or inertial head-neck impact acceleration. In these experiments, unanesthetized Rhesus monkeys were subjected to peak sled accelerations in the \(-X\) direction and somatosensory evoked potentials (EPs) were recorded prior, during, and subsequent to impact. These experiments are discussed in detail in Berger and Weiss [1].

A primary objective of these experiments was to determine the extent to which impact produced shifts in latency of various peaks of the (cervical) evoked potentials. Shifts in latency of each peak were plotted as a function of time over the experiment, relative to impact. Figure 1 gives an example of such a plot. In order to quantitatively assess the amplitude and duration of the latency effect following impact, the following exponential model was proposed:

\[
y = B + St + h(t)D + h(t)A\exp(t/T) + \varepsilon(t) \tag{1}
\]

where:

- \(y\) is the value of the shift in latency with respect to the preimpact baseline average evoked potential (AEP)
CERVICAL AEP LATENCY SHIFT

RUN LX3009. 796 M/s². PEAK 8.46 MS. N-10

Figure 1. Example Plot of the Shift in Latency as a Function of Time, Relative to Impact. (Taken from Berger and Weiss [1].)
t is the time relative to impact
S is the slope of the linear baseline shift
B is the amplitude value of the baseline instantaneously prior to impact
D is the "permanent" shift induced by impact
A is the exponential amplitude
T is the exponential time coefficient
h(t) is the unit step function with value zero for t<0, and unity for t>0
ε(t) is the noise.

In Berger and Weiss [1], the linear baseline shift parameters, S and B, were determined by applying simple linear regression to the preimpact data. The linear equation was extrapolated in the postimpact region, and residual values were computed by subtracting out the linear equation component. However, a regression procedure which would fit the model $D + A \exp(t/T)$ to the postimpact residuals was not available. Consequently, an additional exponential term was introduced in lieu of the D term, and the residuals were then subjected to a polyexponential regression procedure. More specifically, the (postimpact) residuals were fit to a regression function of the following form:

$$A_1 \exp(t/T_1) + A_2 \exp(t/T_2).$$

The estimate of the "permanent" shift due to impact, D, was
obtained from the amplitude coefficient corresponding to the
exponential term having a time coefficient ($T_i$) that was very
long in relation to the duration of the analysis interval. In
some cases a constant value was added to the data to ensure that a
term with a long time coefficient would be obtained.

In this Desmatics technical report we present an
alternative computational procedure for fitting model (1)
directly to empirical latency data. In addition, we give a
FORTRAN program implementing the proposed computational method.
Finally, we include an example which illustrates a direct
application of the methodology.
2. METHODS

We can rewrite model (1) as

\[ y_i = f(t_i, \theta) + \epsilon(t_i) \quad (i = 1, 2, ..., n) \]

where \( y_i \) is the measured latency shift corresponding to the known time \( t_i \), \( \epsilon(t_i) \) is the random error component, \( n \) denotes the total number of data points, and the vector \( \theta = (S, B, D, A, T) \) consists of the 5 parameters to be estimated. Assuming that the errors \( \epsilon(t_i) \) all have zero mean, the expected responses, denoted by \( E(y_i) \), can be represented as:

\[ E(y_i) = f(t_i, \theta) \]

where \( f \) is of the form

\[ f(t, \theta) = B + St + h(t)D + h(t)Aexp(t/T). \]  

If we assume further that the errors \( \epsilon(t_i) \) all have the same variance and are uncorrelated, then it is reasonable to use the least squares method to fit equation (2) to the data. That is, we wish to find the estimate of \( \theta \), say \( \hat{\theta} \), which minimizes the sum of squares

\[ Q(\theta) = \sum_{i=1}^{n} \left[ y_i - f(t_i, \theta) \right]^2. \]
The proposed procedure is based on the following approach. For known values of \( S \) and \( B \), say \( s \) and \( b \), we define

\[
y_i(s,b) = y_i - b - st_i.
\]

Note that

\[
E[y_i(s,b)] = \begin{cases} 
0 & \text{if } t_i < 0 \\
D + A\exp(t_i/T) & \text{if } t_i > 0.
\end{cases}
\]

We will assume henceforth that the data has been ordered and indexed so that \( t_1 < t_2 < t_3 < \ldots < t_n \). In addition, we let \( n_1 \) and \( n_2 \) denote the total number of preimpact \((t_i < 0)\) and postimpact \((t_i > 0)\) data points, respectively.

It is clear that for known values of \( S \) and \( B \) the parameters \( D, A, \) and \( T \) can be estimated from an analysis of the postimpact residuals \( y_i(s,b), i > n_1 \). Accordingly, let \( \hat{D}(s,b), \hat{A}(s,b), \) and \( \hat{T}(s,b) \) denote the least squares estimates obtained from fitting the exponential function \( D + A\exp(t_i/T) \) to the residuals \( y_i(s,b), i > n_1 \). [We assume temporarily that we have a method at our disposal for computing \( \hat{D}(s,b), \hat{A}(s,b), \) and \( \hat{T}(s,b) \).]

If we define

\[
V(s,b) = Q(\hat{\theta}(s,b)) = \sum_{i=1}^{n} [y_i - f(t_i,\hat{\theta}(s,b))]^2
\]

where \( \hat{\theta}(s,b) = (s, b, \hat{D}(s,b), \hat{A}(s,b), \hat{T}(s,b)) \) we have the following key result:
Let $s$ and $b$ denote the values of $S$ and $B$, respectively, which minimize $V(s,b)$. Then $\hat{\theta}$, the value of $\theta$ which minimizes $Q(\theta)$, is given by $\hat{\theta} = (\hat{s}, \hat{b}, \hat{D}(\hat{s}, \hat{b}), \hat{A}(\hat{s}, \hat{b}), \hat{T}(\hat{s}, \hat{b}))$.

The significance of this result is that the stated problem of minimizing a five-variable function can be reduced to that of minimizing the two-variable function $V(s,b)$. Of course, an analysis of $V$ will require that we have available a method for fitting the exponential model $D + A\exp(t_i/T)$ to the residuals $y_i(s,b)$ for some $s$ and $b$, where $i > n_1$. In Section 3 a general method for fitting an exponential growth and decay model with an asymptote will be described. In Section 4 a method for minimizing $V(s,b)$ will be presented.
3. EXponential regression with an Asymptote

In this section we address the problem of finding the estimate of \( \lambda \), say \( \hat{\lambda} = (\hat{\lambda}_1, \hat{\lambda}_2, \hat{\lambda}_3) \), which minimizes the sum of squares

\[
F(\lambda) = \sum_{i > n_1} [z_i - w(t_i, \lambda)]^2 ,
\]

where \( w(t_i, \lambda) = \lambda_1 + \lambda_2 \exp(\lambda_3 t_i) \) and \( z_i = y_i(s, b) \), with \( t_i > 0 \). The summation in (3) contains \( n_2 \) terms, one for each postimpact data point. We will use a modified Newton's method to compute the vector \( \hat{\lambda} \).

Newton's method is a widely used, iterative method for minimization and requires use of both the gradient vector and the Hessian matrix in computations. The gradient vector consists of all first-order partial derivatives of the objective function with respect to the unknown parameters; the Hessian matrix consists of all second-order partial derivatives. [For the sake of discussion, we will assume that all first- and second-order derivatives exist and are continuous.]

To provide further description of Newton's method, consider a Taylor series expansion of the objective function, \( F(\lambda) \), about the point \( \lambda_{-1} \). This takes the form

\[
F(\lambda) = F(\lambda_{-1}) + (\lambda - \lambda_{-1})'g_{-1} + \frac{1}{2}(\lambda - \lambda_{-1})'G_{-1}(\lambda - \lambda_{-1}) + R_{1}
\]
approximate the function $V(s,b)$ in the design region. If the fitted surface is an adequate approximation of $V$, then analysis of the fitted surface will be approximately equivalent to an analysis of the actual function. An indication of lack of fit about the fitted surface can be obtained by considering variation between the observed responses (i.e., function evaluations) and the predicted responses (i.e., fitted values) at the nine points which comprise the CCD. The predicted responses $\hat{v} = (\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_9)'$ are given by

$$\hat{v} = X\hat{\beta},$$

where $X$ and $\hat{\beta}$ are as defined previously. A statistic which can be used to measure lack of fit is the mean absolute residual (MAR) defined by

$$\text{MAR} = \left( \frac{1}{9} \sum_{i=1}^{9} |v_i - \hat{v}_i| \right) / 9$$

where $v_1, v_2, \ldots, v_9$ are the nine observed responses. The deviations $v_i - \hat{v}_i$ represent the difference between the observed and fitted values based on the second-order regression. The closer are the $v_i$ to the $\hat{v}_i$, the smaller is the value of MAR, and hence the better the approximation by the second-order response surface.
minimum on the estimated response surface is given by

\[ \hat{x} = -\frac{1}{2} B^{-1} C \]

where \( C = (\hat{\beta}_1, \hat{\beta}_2)' \) and

\[ B = \begin{bmatrix} \hat{\beta}_3 & \hat{\beta}_5/2 \\ \hat{\beta}_5/2 & \hat{\beta}_4 \end{bmatrix}. \]

In the original space, we have

\[ \hat{s} = \hat{s} + F_s \hat{x}_1 \]

and

\[ \hat{b} = \hat{b} + F_b \hat{x}_2. \]

It should be noted that under proper scaling of the design region, the point \((\hat{x}_1, \hat{x}_2)\) should be located somewhere within the design region (i.e., \( \hat{x}_1^2 + \hat{x}_2^2 \leq 2 \)). However, if this point should fall outside the design region, then a new CCD design should be set up with \((\hat{s}, \hat{b})\), as defined above, as the center.

**Lack of Fit**

The second-order response surface given in (4) is used to
is fit to the nine responses (i.e., function evaluations) \( v_1, v_2, \ldots, v_9 \) corresponding to the nine points in the CCD. In matrix notation we have

\[
\mathbf{v} = \mathbf{X} \mathbf{\beta},
\]

where \( \mathbf{v} = (v_1, v_2, \ldots, v_9)' \), \( \mathbf{\beta} = (\beta_0, \beta_1, \ldots, \beta_5)' \), and

\[
\mathbf{X} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\
1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & -1 & 1 & 1 & -1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 & 0 \\
1 & -\sqrt{2} & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
1 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & -\sqrt{2} & 0 & 2 & 0 & 0 & 0 & 0
\end{bmatrix}.
\]

It follows directly that the estimate of the regression coefficients is given by

\[
\hat{\mathbf{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{v}.
\]

We can also show that the point \( \hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2) \) which results in the
where $F_s$ and $F_b$ represent scaling factors. In terms of the coded variables, we have

$$s = \tilde{s} + F_s x_1$$

and

$$b = \tilde{b} + F_b x_2.$$ 

Thus, for example, the coded design point $(x_1, x_2) = (\sqrt{2}, 0)$ corresponds to the uncoded design point $(s, b) = (\tilde{s} + \sqrt{2} F_s \tilde{b})$.

Initial parameter estimates of the linear baseline shift parameters, $S$ and $B$, can readily be determined by applying simple linear regression to the preimpact data. From this linear regression one can also determine the standard errors of the estimates $\tilde{s}$ and $\tilde{b}$. These standard errors can be used to scale the design region. It seems reasonable to define the scaling factors $F_s$ and $F_b$ as one-tenth the respective standard errors of $\tilde{s}$ and $\tilde{b}$. Additional comments on the scaling of the design region will be given later in this section.

As we indicated earlier, a second-order response surface in the coded variables

$$v_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{1i}^2 + \beta_4 x_{2i}^2 + \beta_5 x_{1i} x_{2i} \quad (4)$$
Figure 3. Central Composite Design For Two Variables
explores the parameter space by means of a geometric configuration of points. Function evaluations are made at each point in the design configuration, and a second-order response surface is fit to the responses (i.e., evaluations) to approximate $V(s,b)$ in the design region. An analysis of the fitted surface is then made to determine the point which results in the minimum of the estimated response function.

The statistical design we will use to determine $(s,b)$ is called a central composite design (CCD). A CCD for two variables, $x_1$ and $x_2$, is illustrated geometrically in Figure 3. The design consists of the nine following coded design points: $(1,1)$, $(-1,1)$, $(-1,-1)$, $(1,-1)$, $(0,0)$, $(0,\alpha)$, $(-\alpha,0)$, $(\alpha,0)$, and $(0,-\alpha)$, where $\alpha = \sqrt{2}$. The first four points are the usual factorial points for fitting a first order model for two variables. The fifth point is the center point of the design, and the four remaining points are the axial points of the square. Notice that each variable is measured at five coded levels: $0$, $1$, $-1$, $\alpha$, and $-\alpha$.

The center point of the CCD design corresponds to our initial estimates, which we will denote by $\tilde{s}$ and $\tilde{b}$, of $S$ and $B$. The coded variables $x_1$ and $x_2$ are defined as

$$x_1 = (s - \tilde{s}) / F_s$$

and

$$x_2 = (b - \tilde{b}) / F_b,$$
4. MINIMIZING $V(s,b)$

In this section we consider the problem of estimating the parameters $S$ and $B$, the slope of the linear baseline shift and the amplitude value of the baseline instantaneously prior to impact, respectively. In Section 3 we assumed fixed known values, $s$ and $b$, for these parameters and outlined an iterative method for obtaining least squares estimates of the postimpact parameters $D$, $A$, and $T$, based on an analysis of the residuals $y_i(s,b) = y_i - b - st_i$, with $t_i > 0$. Following the notation used in Section 2, let $\hat{D}(s,b)$, $\hat{A}(s,b)$, and $\hat{T}(s,b)$ denote the least squares estimates obtained from fitting the exponential function $D + A\exp(t_i/T)$ to the residuals $y_i(s,b)$, $i>n_1$. We wish now to find the estimates $\hat{s}$ and $\hat{b}$ which minimize the function

$$V(s,b) = Q(\hat{\theta}(s,b)) = \sum_{i=1}^{n} [y_i - f(t_i,\hat{\theta}(s,b))]^2$$

where $\hat{\theta}(s,b) = (s, b, \hat{D}(s,b), \hat{A}(s,b), \hat{T}(s,b))$ and the summation extends over all the data.

Because $V(s,b)$ cannot be differentiated analytically and because function evaluations (of $V$) are computationally demanding, to determine $\hat{s}$ and $\hat{b}$ we will use a numerical method which (1) makes no use of any of the derivatives of $V$, and (2) is highly efficient with regard to the number of function evaluations it requires. The proposed method is based on the use of a statistical design which
interval in which the minimum is known to lie by a factor of $r$.

The width of the target interval at each iteration is called the tolerance. A prescribed accuracy, $a$, can be achieved for estimating $\lambda_3$ by repeating the golden section procedure until the tolerance is less than or equal to $a$. 

-15-
Figure 2. Example of a Unimodal Function $f(u)$. 
interval \([x_l, x_u]\). A function \(f(x)\) is unimodal in the interval 
\([x_l, x_u]\) if it has a unique minimum (maximum), occurring at \(x^*\),
in the interval, is a decreasing (increasing) function in the
interval \([x_l, x^*]\), and is an increasing (decreasing) function in
the interval \([x^*, x_u]\).

A complete discussion of the method of golden section can
be found in Keifer [3]. However, to illustrate the basic technique,
consider a unimodal function \(f(u)\) with a minimum in the interval
\([u_1, u_4]\) and consider the four points \(u_1 < u_2 < u_3 < u_4\) that
satisfy the spacing equations

\[
u_3 - u_1 = u_4 - u_2 = r(u_4 - u_1)
\]

where \(r = 2/(1 + \sqrt{5}) \approx 0.618034\). The two limits \(u_1\) and \(u_4\) are
fixed in advance and are known to bracket the minimum.

By testing the relative values of \(f(u_1), f(u_2), f(u_3),\) and
\(f(u_4),\) it is possible, since \(f\) is unimodal [see Figure 2, for
example], to determine in which of the two intervals \([u_1, u_3]\)
or \([u_2, u_4]\) the minimum lies. Without loss of generality (note
that the intervals are of identical length), assume that the
minimum lies in the interval \((u_1, u_3)\). The process is then
repeated in the target interval \((u_1, u_3)\) by considering the
addition of a new interior point which satisfies the basic
spacing equations. Hence each subsequent iteration, which
involves one function evaluation, reduces the previous target
convergence process, especially with quadratically convergent methods such as Greenstadt's. We will now proceed to describe a general method for obtaining initial parameter estimates of $\lambda_1$, $\lambda_2$, and $\lambda_3$.

Once again, consider the regression function

$$w(t, \lambda) = \lambda_1 + \lambda_2 \exp(\lambda_3 t),$$

which we wish to fit to the set of data $z_i = y_i(s, b), i > n_1$. Observe that for a fixed known value of $\lambda_3$, say $\hat{\lambda}_3$, estimates of $\lambda_1$ and $\lambda_2$, call them $\hat{\lambda}_1'$ and $\hat{\lambda}_2'$, can easily be determined via a simple linear regression of the $z_i$ on $\lambda_1 + \lambda_2 \hat{\tau}_i$ where $\hat{\tau}_i = \exp(\hat{\lambda}_3 t_i)$. Accordingly, define

$$W(\hat{\lambda}_3) = \sum_{i>n_1} [z_i - \hat{\lambda}_1' - \hat{\lambda}_2' \exp(\lambda_3 t_i)]^2,$$

where $\hat{\lambda}_1'$ and $\hat{\lambda}_2'$ depend on $\hat{\lambda}_3$. Thus, an initial estimate of $\lambda_3$ can be obtained by finding the value of $\hat{\lambda}_3$ which minimizes $W(\hat{\lambda}_3)$. The corresponding values of $\hat{\lambda}_1'$ and $\hat{\lambda}_2'$ will provide initial parameter estimates of $\lambda_1$ and $\lambda_2$.

To minimize the function $W(\hat{\lambda}_3)$, we will use the method of golden section, which is a commonly used method for solving optimization problems in one variable. This method assumes that the objective function is a unimodal function in a specified
\[ d(i,j) = \begin{cases} 
1 & \text{for } j = 1 \\
\exp(\lambda_3 t_i) & \text{for } j = 2 \\
t_1 \lambda_2 \exp(\lambda_3 t_i) & \text{for } j = 3 
\end{cases} \]

and

\[ h(i,j,k) = \begin{cases} 
t_1 \exp(\lambda_3 t_i) & \text{for } j = 2, k = 3, \text{ and } j = 3, k = 2 \\
t_1 \lambda_3 \exp(\lambda_3 t_i) & \text{for } j = 3, k = 3 \\
0 & \text{otherwise}. 
\end{cases} \]

Using these results, the components of the gradient vector of \( F(\lambda) \) have the form

\[ \frac{\partial F(\lambda)}{\partial \lambda_j} = -2 \sum_{i > n_1} (z_i - \omega(t_i, \lambda))d(i,j), \]

and the components of the Hessian matrix of \( F(\lambda) \) have the form

\[ \frac{\partial^2 F(\lambda)}{\partial \lambda_i \partial \lambda_j} = -2 \sum_{i > n_1} (z_i - \omega(t_i, \lambda))h(i,j,k) + 2 \sum_{i > n_1} d(i,j)d(i,k). \]

**Initial Parameter Estimates**

As we have already noted, the proposed method of estimating \( \lambda \) requires initial parameter estimates of \( \lambda_1, \lambda_2, \) and \( \lambda_3. \)

Reasonably good initial estimates should be provided, however, since poor starting estimates can have a detrimental effect on the
eigenvector, with \( v'_i, v'_{i+1} = 1 \). Define the matrix \( G_i^* \) as

\[
G_i^* = \sum_{j=1}^{P} \gamma_{ij} v'_{ij} v'_{ij}
\]

where \( \gamma_{ij}^* = \max \{ |\gamma_{ij}|, \delta \} \) and \( \delta \) is a small positive number.

The matrix \( G_i^* \) is positive definite with inverse given by

\[
(G_i^*)^{-1} = \sum_{j=1}^{P} (\gamma_{ij})^{-1} v'_{ij} v'_{ij}.
\]

Under the proposed modification, the general iterative scheme is given by

\[
\lambda_{i+1} = \lambda_i - \alpha i (G_i^*)^{-1} g_i
\]

where the scalar \( \alpha i > 0 \), is a step size factor introduced to help avoid stepping across a minimum. In general, \( \alpha i \) is selected so that \( F(\lambda_{i+1}) < F(\lambda_i) \); usually, \( \alpha i = 1 \) will suffice. Iterations are carried out according to the general scheme until none of the components of the vector \( \lambda \) change by more than a negligible amount.

For use in Greenstadt's method, we will now derive the gradient vector and Hessian matrix of the objective function \( F(\lambda) \) given in equation (3). Let

\[
d(i,j) = \partial w(t_i, \lambda) / \partial \lambda_j \quad \text{and} \quad h(i,j,k) = \partial^2 w(t_i, \lambda) / \partial \lambda_j \partial \lambda_k.
\]

It can easily be checked that
where \( g_1 \) is the gradient vector evaluated at \( \lambda_1 \), \( G_1 \) is the Hessian matrix evaluated at \( \lambda_1 \), and \( R_1 \) is the remainder. The terms in the Taylor expansion, ignoring \( R_1 \), define a quadratic function which will approximate \( F(\lambda) \) when \( \lambda \) is close to \( \lambda_1 \). The stationary point (i.e., a point at which the gradient vector vanishes) of this quadratic function is given by the solution to the linear system of equations

\[
G_1(\lambda - \lambda_1) = -g_1.
\]

If \( G_1 \) is positive definite and therefore invertible, this suggests the general iterative scheme

\[
\lambda_{i+1} = \lambda_i - G_1^{-1} g_1.
\]

This recursive relation defines Newton's method, although one must generally provide an initial starting point \( \lambda_0 \). The method normally displays a quadratic rate of convergence near a stationary point.

A potential problem with Newton's method is that the Hessian matrix may not be positive definite at each iteration; consequently, several modifications have been suggested. Greenstadt [2] suggested the following variant of Newton's method. Let \( \gamma_{ij} \) be the jth eigenvalue of the \( p \times p \) matrix \( G_1 \) and \( v_{ij} \) its corresponding
5. AN EXAMPLE

In this section we illustrate by means of an example a direct application of the methodology developed in Sections 2, 3, and 4. In the Appendix we give a FORTRAN-77 listing of the proposed nonlinear regression method. This program runs interactively and was implemented on the Zenith-100 microcomputer. The program can easily be converted to run on a mainframe computer or another microcomputer which supports a FORTRAN compiler.

Cervical EP latency data was obtained from the Texas Research Institute of Mental Sciences.\(^1\) Latency data of a selected EP peak from Experiment #LX3009 (796 m/sec\(^2\)) is presented in Table 1. For this data, each test AEP spanned 10.2627 seconds and was computed by averaging a sequence of 50 EPs with a stimulus sampling rate of 4.872 Hz. Latency values are given starting at six minutes prior to impact, continuing through impact, and ending fifteen minutes after impact. The data is plotted in Figure 4. It should be noted that the three data points marked by an asterisk (*) in Table 1 were identified as spurious data values and omitted from subsequent analyses.

A baseline latency value of 3.865 ms was computed by averaging the preimpact latency values. The shift in latency at each time point was determined by calculating the difference between the

---

\(^1\)The author is indebted to Mr. William D. Burton, Systems Analyst at the Texas Research Institute of Mental Sciences, for supplying the latency data.
<table>
<thead>
<tr>
<th>Relative Time (sec)</th>
<th>Latency Shift (μs)</th>
<th>Latency (ms)</th>
<th>Relative Time (sec)</th>
<th>Latency Shift (μs)</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-356.000</td>
<td>3.875</td>
<td>10.</td>
<td>-64.772</td>
<td>4.025</td>
<td>160.</td>
</tr>
<tr>
<td>-345.737</td>
<td>3.900</td>
<td>35.</td>
<td>-75.034</td>
<td>3.950</td>
<td>85.</td>
</tr>
<tr>
<td>-181.534</td>
<td>3.825</td>
<td>-40.</td>
<td>-239.238</td>
<td>3.925</td>
<td>60.</td>
</tr>
<tr>
<td>* -7.067</td>
<td>3.925</td>
<td>60.</td>
<td>-413.704</td>
<td>3.875</td>
<td>10.</td>
</tr>
<tr>
<td>44.246</td>
<td>4.000</td>
<td>135.</td>
<td>-465.018</td>
<td>3.900</td>
<td>35.</td>
</tr>
</tbody>
</table>

Table 1. Cervical EP Latency Data of a Selected Peak: Animal AR - 0761, Experiment #LX3009, Acceleration 796 m/sec² →X direction.
latency value in Table 1 and the baseline latency value. These shifts (see Table 1) and corresponding times were then analyzed using the FORTRAN program presented in the Appendix.

In addition to the data, the user must also supply the program with two values which bracket $T^{-1}$, where $T$ is the exponential time coefficient in model (1). A lower bound of $-1$ sec$^{-1}$ and an upper bound of 0 sec$^{-1}$ were specified for $T^{-1}$. This is tantamount to specifying that $T \leq -10$ seconds. There were 119 data points, of which 34 were preimpact and 85 were postimpact.

The final estimates of the five model parameters were as follows:

$$\hat{S} = -0.04658660 \ \mu\text{sec/sec}$$
$$\hat{B} = -8.25491618 \ \mu\text{sec}$$
$$\hat{D} = 66.28035173 \ \mu\text{sec}$$
$$\hat{A} = 223.59247934 \ \mu\text{sec}$$
$$\hat{T} = -65.27566084 \ \text{sec}$$

The estimate of root mean square deviation was 19.017 \mu\text{sec}. The lack of fit statistic, MAR, for the CCD was $\text{MAR} = 663.6 \ (\mu\text{sec})^2$, which was 1.6% of the residual sum of squares [i.e., $V(\hat{s}, \hat{b})$].

A plot of the fitted curve is shown in Figure 5.

The data was reanalyzed using the seven minute time interval starting at two minutes preimpact and ending five minutes postimpact. There were 39 data points in this interval, of which 11 were preimpact and 28 were postimpact. The estimated parameters were
as follows:

\[ \hat{S} = -0.10731488 \ \mu\text{sec/sec} \]
\[ \hat{B} = -6.45739421 \ \mu\text{sec} \]
\[ \hat{D} = 78.29595060 \ \mu\text{sec} \]
\[ \hat{A} = 205.91875792 \ \mu\text{sec} \]
\[ \hat{T} = -63.04820536 \ \text{sec}. \]

The estimate of root mean square deviation was 19.768 \mu\text{sec}.

The lack of fit statistic, MAR, for the CCD was \( \text{MAR} = 372.6 \) \( (\mu\text{sec})^2 \), which was 2.8\% of the residual sum of squares.
6. REFERENCES


APPENDIX

IMPLICIT REAL*8(A-H,P-Z)
REAL*8 XD(600), YD(600), T(3), XX(600), YY(600), Z(6,9), T3LIM(2),
& X1(9), X2(9), YC(9), BETA(6), TPRE(3)
COMMON DELT, E1, E2, MAXIT, MPRE, MPOST, XD, YD, T3LIM, TOL
DATA Z(0.0D0, 2.0D0, 2.0D0, 1.0D0, 1.0D0, 4.0D0,
& 0.0D0, -2.0D0, 2.0D0, 1.0D0, 1.0D0, 4.0D0,
& 0.0D0, 2.0D0, -2.0D0, 1.0D0, 1.0D0, -4.0D0,
& 16.0D0, 0.0D0, 0.0D0, -8.0D0, -8.0D0, 0.0D0,
& 0.0D0, 0.0D0, 2.0D0, -1.0D0, 3.0D0, 0.0D0,
& 0.0D0, -2.0D0, 0.0D0, 3.0D0, -1.0D0, 0.0D0,
& 0.0D0, 2.0D0, 0.0D0, 3.0D0, -1.0D0, 0.0D0,
& 0.0D0, 0.0D0, -2.0D0, -1.0D0, 3.0D0, 0.0D0/.
Z(2, 2) = Z(2, 2) * DSQRT(2.0D0)
Z(2, 8) = Z(2, 8) * DSQRT(2.0D0)
Z(3, 6) = Z(3, 6) * DSQRT(2.0D0)
OPEN (6, FILE='PRN')
Z(3, 9) = Z(3, 9) * DSQRT(2.0D0)
DO 10 I = 1, 6
DO 11 J = 1, 9
11 Z(I, J) = Z(I, J) / 16.0D0
10 CONTINUE
WRITE(*, 7)
7 FORMAT('A NONLINEAR REGRESSION PROCEDURE FOR EVOKED POTENTIAL',
& 'DATA ANALYSIS BY CARL A. MAURO, DESMATICS INC. ',
& 'PO BOX 618, STATE COLLEGE, PA 16804 (814-238-9621).',
& 'BASED ON DESMATICS TECHNICAL REPORT NO. 112-18',
& 'PROGRAM FITS A REGRESSION FUNCTION OF THE FORM',
& 'Y = B + S + h(t) \cdot [D + A \cdot \exp(t/T)]',
& 'WHERE:',
& 't = TIME RELATIVE TO IMPACT(SECS)',
& 'Y = LATENCY SHIFT(MICROSECS) RELATIVE TO BASELINE AEP',
& 'h(t) = ZERO FOR t<0 AND UNITY FOR t>=0',
& 'B = AMPLITUDE INSTANaneously PRIOR TO IMPACT(MICROSECS)',
& 'S = SLOPE OF LINEAR BASELINE SHIFT(MICROSECS/SEC)',
& 'D = SHIFT INDUCED BY IMPACT(MICROSECS)',
& 'A = EXPONENTIAL AMPLITUDE(MICROSECS)',
& 'T = EXPONENTIAL TIME COEFFICIENT(SECS)'/)
PAUSE
WRITE(*, 8)
8 FORMAT('NOTE: (1) DATA IS READ FREE-FORMAT FROM THE FILE',
& 'WITH FILENAME DATA.EP',
& ' (2) EACH RECORD IN DATA.EP MUST CONTAIN THE TIME',
& 'IN SECONDS OF THE DATA POINT, FOLLOWED BY THE',
& 'CORRESPONDING Y VALUE IN MICROSECONDS',
& ' (3) DATA ARRAYS ARE DIMENSIONED TO ACCOMODATE',
& 'A MAXIMUM OF 600 DATA POINTS',
& ' (4) IN ADDITION TO THE DATA, USER MUST SUPPLY TWO',
& 'VALUES WHICH BRACKET THE RECIPROCAL OF t',
& 'PLEASE INPUT LOWER LIMIT FOR 1/t (IN 1/SECS)' )
READ(*, *) T3LIM(1)
WRITE(*, 53)
53 FORMAT('PLEASE INPUT UPPER LIMIT FOR 1/t (IN 1/SECS)' )
READ(*, *) T3LIM(2)
IF (T3LIM(2).LE.T3LIM(1)) GO TO 6
SF=10.0D0
TOL=.0001
DELT=2.**( -8)
E1=.0001
E2=.001
MAXIT=25
MPRE=0
MPOST=0
C )))) INPUT DATA
WRITE(*,54)
54 FORMAT(/' READING DATA ....')
OPEN(1,FILE='DATA.FR')
DO 5 I=1,601
READ(1,* ,END=15) XD(I),YD(I)
5 CONTINUE
15 M=I-1
CLOSE (1)
C )))) SORT DATA
ND=M
NM=M-1
DO 20 I=1,NM
ND=ND-I
DO 21 J=1,ND
IF (XD(J).LE.XD(J+1)) GOTO 21
A=XD(J)
B=YD(J)
XD(J)=XD(J+1)
YD(J)=YD(J+1)
XD(J+1)=A
YD(J+1)=B
21 CONTINUE
20 CONTINUE
C )))) SEPARATE PRE FROM POST IMPACT DATA
DO 30 I=1,M
IF (XD(I).GE.0.) GO TO 31
MPRE=MPRE+1
XX(MPRE)=XD(I)
YY(MPRE)=YD(I)
GO TO 30
31 MPOST=MPOST+1
30 CONTINUE
C )))) CHECK FOR SUFFICIENT DATA POINTS
IF (MPRE.LT.3.OR.MPOST.LT.6) STOP 'INSUFFICIENT DATA'
WRITE(*,55) MPRE,MPOST
55 FORMAT(/' NUMBER OF PREIMPACT DATA POINTS= ',I4,
" NUMBER OF POSTIMPACT DATA POINTS= ',I4," COMPLETING ....")
C )))) COMPUTE INITIAL ESTIMATES OF BASELINE SLOPE AND AMPLITUDE(INT)
CALL SREG(MPRE,XX,YY,BZ,B1)
C )))) COMPUTE ESTIMATED STANDARD ERRORS OF INITIAL BZ AND
B1(SDBZ,SDB1)
PSSE=0.0D0
XXBAR=0.0D0
DO 35 I=1,MPRE
XXBAR=XXBAR+XX(I)
35 PSSE=PSSE+(YY(I)-BZ-B1*XX(I))**2
-31-
XXBAR = XXBAR / MPRE
SD = DSQRT (PSSE / (MPRE - 2.0D0))
SXX = 0.0D0
DO 36 I = 1, MPRE
  36 SXX = SXX + (XX(I) - XXBAR)**2
A = 1. / (1.0D0 * MPRE) + (XXBAR**2) / SXX
SDB1 = SD / DSQRT (SXX)
SDB2 = SD / DSQRT (A)
C >>> START CENTRAL COMPOSITE DESIGN
C >>> DEFINE DESIGN MATRIX
40 X1(1) = B1 + SDB1 / SF
X1(2) = B1 - SDB1 / SF
X1(3) = X1(2)
X1(4) = X1(1)
X1(5) = B1
X1(6) = B1
X1(7) = B1 - DSQRT (2.0D0) * SDB1 / SF
X1(8) = B1 + DSQRT (2.0D0) * SDB1 / SF
X1(9) = B1
X2(1) = BZ + SDB2 / SF
X2(2) = X2(1)
X2(3) = BZ - SDB2 / SF
X2(4) = X2(3)
X2(5) = BZ
X2(6) = BZ + DSQRT (2.0D0) * SDB2 / SF
X2(7) = BZ
X2(8) = BZ
X2(9) = BZ - DSQRT (2.0D0) * SDB2 / SF
DO 100 I = 1, 9
CALL EGAD (X1(I), X2(I), T, YC(I))
IF (I .NE. 5) GOTO 100
TPRE(1) = T(1)
TPRE(2) = T(2)
TPRE(3) = 1.0D0 / T(3)
100 CONTINUE
C >>> DETERMINE BETAS FOR QUADRATIC FIT
DO 105 I = 1, 6
  BEI = BETA(I) = 0.0D0
DO 110 J = 1, 9
  110 BETA(I) = BETA(I) + Z(I, J) * YC(J)
105 CONTINUE
C >>> DETERMINE STATIONARY POINT
STAT1 = BETA(2) * BETA(5) - BETA(3) * BETA(6) / 2.0D0
STAT1 = STAT1 / (2.0D0 * DETER)
STAT2 = BETA(3) * BETA(4) - BETA(2) * BETA(6) / 2.0D0
STAT2 = STAT2 / (2.0D0 * DETER)
WDB = DSQRT ((STATZ**2 + STAT1**2) / 2.0D0)
C >>> TRANSFORM TO ORIGINAL SPACE
STAT1 = B1 + STAT1 * SDB1 / SF
STATZ = BZ + STATZ * SDBZ / SF
CALL EGAD (STAT1, STATZ, T, FSTAT)
RMS = DSQRT (YC(5) / (M - 5.0D0))
WRITE (6, 150) B1, BZ, TPRE(1), TPRE(2), TPRE(3), YC(5), RMS
150 FORMAT (/// 'CENTRAL POINT OF CENTRAL COMPOSITE DESIGN:',)
& /' S='F15.8,'/ B='F15.8,'/ CONDITIONAL ESTIMATES:,',
& /' D='F15.8,'/ A='F15.8,'/ T='F15.8,
& /' SUM OF SQUARED RESIDUALS='F15.6,
& /' ROOT MEAN SQUARE='F15.7)
RMS=DSQRT(FSTAT/(M-5.00))
XMAR=0.0000
DO 153 I=1,9
   YCP=BETA(1)+BETA(2)*X1(I)+BETA(3)*X2(I)+BETA(4)*X1(I)*X1(I)
& +BETA(5)*X2(I)*X2(I)+BETA(6)*X1(I)*X2(I)
   XMAR=XMAR+DABS(YC(I)-YCP)
153 CONTINUE
XMAR=XMAR/9.0000
WRITE(6,160)
STAT1,STATZ,T(1),T(2),1.000/T(3),FSTAT,RMS,XMAR
160 FORMAT('////'FINAL ESTIMATES OF MODEL PARAMETERS:,',
& /' S='F15.8,'/ B='F15.8,'/ D='F15.8,'/ A='F15.8,
& /' T='F15.8,'/ SUM OF SQUARED RESIDUALS='F15.6,
& /' ROOT MEAN SQUARE='F15.7,
& /' LACK OF FIT STATISTIC FOR CENTRAL COMPOSITE DESIGN:,',
& ' MAR='F15.6/
   IF (WDB.LE.DSQRT(2.00)) GO TO 200
   WRITE(6,59)
59 FORMAT('NOTE: STATIONARY POINT IS OUTSIDE BOUNDARIES OF,',
& '/' CENTRAL COMPOSITE DESIGN. DESIGN WILL BE RECENTERED,',
& '/' AND NEW RESPONSE SURFACE FIT TO (S,B) SPACE:,',
& '/' COMPUTING ......')
B1=STAT1
B2=STAT2
GO TO 40
200 STOP
END
SUBROUTINE EGADEGM(PSLP,PINT,T,FUN)
IMPLICIT REAL*8(A-H,P-Z)
REAL*8 X(600),Y(600),T(3),TH(3),YR(600),JM(600,3),TT(3),
& JR(3),G(3,3),D(3),Z(3,3),WK(3),V(3,3,3),GMINV(3,3),
& XSL(600),XD(600),YD(600),T3LIM(2)
COMMON DELTE1,E2,MAXIT,MPRE,MPRE,PSLP,XDYD,T3LIM,TOL
C )))))))) FORM ADJUSTED POST IMPACT DATA ARRAYS
DO 4 I=1,MPRE
X(I)=XD(I+MPRE)
Y(I)=YD(I+MPRE)-PINT-PSLP*XD(I+MPRE)
4 CONTINUE
C )))))))) COMPUTE STARTING ESTIMATES OF T(1),T(2),T(3)
A=T3LIM(1)
B=T3LIM(2)
XMIN = A
IER = 129
IF (B .LE. A) GO TO 24
IER = 130
IF (TOL .GE. (B-A)) GO TO 24
IER = 0
C = (3.000-DSQRT(5.00))/2.000
H = C*(B-A)
V1 = A+H
V2 = B-H
FA = F(A,X,Y)
FB = F(B,X,Y)
FV1 = F(V1,X,Y)
FV2 = F(V2,X,Y)

5 CONTINUE
    IF (A .GE. V1 .OR. V1 .GE. V2 .OR. V2 .GE. B) GO TO 23
    IF (FV1 .GE. FV2) GO TO 10
    IF (FV2 .GT. FB) GO TO 25
    B = V2
    IF (TOL .GE. (B-A)) GO TO 15
    FB = FV2
    V2 = V1
    FV2 = FV1
    H = C*(B-A)
    V1 = A+H
    FV1 = F(V1,X,Y)
    GO TO 5

10 IF (FV1 .GT. FA) GO TO 21
    A = V1
    IF (TOL .GE. (B-A)) GO TO 20
    FA = FV1
    V1 = V2
    FV1 = FV2
    H = C*(B-A)
    V2 = B-H
    FV2 = F(V2,X,Y)
    GO TO 5

15 XMIN = V1
    IF (FA .LT. FV1) XMIN = A
    GO TO 27

20 XMIN = V2
    IF (FB .LT. FV2) XMIN = B
    GO TO 27

25 XMIN = V2
    A = V1
    GO TO 22

21 XMIN = V1
    B = V2

22 IER = 131
    GO TO 24

23 IER = 132
    XMIN = A
    IF (FB .LT. FA) XMIN = B

24 CONTINUE
    IF (IER .EQ. 0) GO TO 27
    WRITE(*,26) IER

26 FORMAT(' TERMINAL ERROR FROM GOLDSER. ERROR CODE= ',I3)
    STOP

27 CONTINUE
    T(3) = XMIN
    DO 31 I=1,MPOST

31 XSL(I) = DEXP(T(3)*X(I))
    CALL SREG(MPOST,XSL,Y,BINT,BSLP)
    T(1) = BINT
    T(2) = BSLP

C ) ) ) ) ) START ITERATION LOOP
DO 100 IIT=1,MAXIT
   >>> COMPUTE RESIDUAL VECTOR (YR) AND JACOBIAN MATRIX (JM)
   SS=0.
   DO 30 J=1,MPOST
   YR(J)=Y(J)-T(1)-T(2)*DEXP(T(3)*X(J))
   SS=SS+YR(J)*YR(J)
   JM(J,1)=-1.
   JM(J,2)=-DEXP(T(3)*X(J))
   JM(J,3)=-X(J)*T(2)*DEXP(T(3)*X(J))
30 CONTINUE
   >>> FORM MATRIX PRODUCT JR=(TRAN OF JM)*YR
   JR(1)=0.
   JR(2)=0.
   JR(3)=0.
   DO 35 J=1,MPOST
   JR(1)=JR(1)-YR(J)
   JR(2)=JR(2)+JM(J,2)*YR(J)
   JR(3)=JR(3)+JM(J,3)*YR(J)
35 CONTINUE
   >>> COMPUTE HESSIAN MATRIX (G)
   DO 40 I=1,3
   DO 45 J=1,3
   G(I,J)=0.
   DO 50 L=1,MPOST
   G(I,J)=G(I,J)+JM(L,I)*JM(L,J)
   IF (I+J.EQ.6) G(I,J)=G(I,J)-YR(L)*X(L)*X(L)*T(2)*DEXP(T(3)*X(L))
   IF (I+J.EQ.5) G(I,J)=G(I,J)-YR(L)*X(L)*DEXP(T(3)*X(L))
40 CONTINUE
45 CONTINUE
40 CONTINUE
   >>> CALCULATE INVERSE OF MODIFIED HESSIAN MATRIX
   CALL EIGRS(G,3,11,D,Z,3,WK,IER)
   DO 60 J=1,3
   D(J)=DMAX1(ABS(D(J)),DELT)
60 CONTINUE
   DO 62 I=1,3
   DO 63 J=1,3
   DO 64 K=1,3
   V(I,J,K)=Z(J,1)*Z(K,1)
63 CONTINUE
62 CONTINUE
64 CONTINUE
   DO 65 J=1,3
   DO 66 K=1,3
   GMINV(J,K)=0.
66 CONTINUE
65 CONTINUE
64 CONTINUE
   GMINV(J,K)=GMINV(J,K)+V(I,J,K)/D(I)
66 CONTINUE
65 CONTINUE
   >>> UPDATE ESTIMATES
   DO 70 I=1,3
   TH(I)=0.
67 CONTINUE
   TH(I)=TH(I)+GMINV(I,J)*JR(J)
70 CONTINUE
   >>>> STEP SIZE CALCULATION
   ALP=2.
I=1
ALP=ALP/2.
DO 81 J=1,3
TT(J)=T(J)-ALP*TH(J)
CONTINUE
SST=0.
DO 82 J=1,MPOST
S1=Y(J)-TT(1)-TT(2)*DEXP(TT(3)*X(J))
SST=SST+S1*S1
CONTINUE
IF (SST.LT.SS) GOTO 85
I=I+1
IF (I.EQ.10) GOTO 85
GOTO 80
5 CONTINUE
)))) CHECK FOR TERMINATION
ITERM=1
DO 90 I=1,3
F1=ABS(T(I)-TT(I))
F2=E1*(ABS(T(I)))+E2)
IF (F1.GT.F2) ITERM=0
T(I)=TT(I)
0 CONTINUE
IF (ITERM.EQ.1) GOTO 101
DO CONTINUE
80 CONTINUE
81 IF (ITERM.EQ.1) NITT=IIT
IF (ITERM.EQ.0) STOP 'MAX ITERATIONS EXCEEDED IN POST MODULE'
FUNV=0.
DO 150 I=1,MPRE
FUNV=FUNV+(YD(I)-PINT-PSLP*XD(I))**2
50 CONTINUE
DO 160 J=1,MPOST
I=J+MPRE
A=PINT+PSLP*XD(I)
B=T(1)+T(2)*DEXP(T(3)*XD(I))
FUNV=FUNV+(VD(I)-A-B)**2
50 CONTINUE
RETURN
END
SUBROUTINE SREG(N,X,Y,BINT,BSLP)
IMPLICIT REAL*8(A-H,P-Z)
REAL*8 X(600),Y(600)
XBAR=0.
YBAR=0.
DO 5 I=1,N
XBAR=XBAR+X(I)
YBAR=YBAR+Y(I)
CONTINUE
XBAR=XBAR/N
YBAR=YBAR/N
A=0.
B=0.
DO 10 I=1,N
C=X(I)-XBAR
A=A+Y(I)*C
-36-
B = B + C**2

10 CONTINUE
BSLP = A/B
BINT = YBAR - BSLP*XBAR
RETURN
END

SUBROUTINE EIGRS (A, N, JOBN, D, Z, IZ, WK, IER)
DOUBLE PRECISION A(9), D(3), WK(3), Z(3, 3)
INTEGER N, JOBN, IZ, IER, IJ, JI
INTEGER JER, NA, ND, IIZ, I, J, K
DOUBLE PRECISION TEN, ZERO, ONE, THOUS
DATA IER = 0
JER = 0
K = 1
JI = N - 1
IJ = 1
DO 10 J = 1, N
   DO 5 I = 1, J
      A(K) = A(IJ)
      IJ = IJ + 1
      K = K + 1
   5 CONTINUE
   IJ = IJ + JI
   JI = JI - 1
10 CONTINUE
NA = (N*(N+1))/2
ND = 1
CALL EHOUSS (A, N, D, WK, WK)
IIZ = IZ
DO 55 I = 1, N
   DO 50 J = 1, N
      Z(I, J) = ZERO
   50 CONTINUE
   Z(I, J) = ONE
55 CONTINUE
CALL EQR2S (D, WK, N, Z, IIZ, JER)
IF (JER .GT. 128) GO TO 9000
CALL EHOBS (A, N, I, N, Z, IZ)
9000 CONTINUE
IF (JER .EQ. 0) GO TO 9005
STOP 'NOT ALL EIGENVALUES COULD BE COMPUTED'
9005 RETURN
END
SUBROUTINE EHOBS (A, N, M1, M2, Z, IZ)
DIMENSION A(9), Z(3, 3)
DOUBLE PRECISION A, Z, H, S
DO 25 I = 2, N
   L = I - 1
   IA = (I*L)/2
   H = A(IA+1)
   IF (H .EQ. 0.0D0) GO TO 25
   DO 20 J = M1, M2
      S = 0.0D0
      DO 10 K = 1, L
         ...
S = S + A(IA+K) * Z(K,J)

10 CONTINUE
S = S / H
DO 15 K = 1, L
Z(K,J) = Z(K,J) - S * A(IA+K)
15 CONTINUE
20 CONTINUE
25 CONTINUE
RETURN
END

SUBROUTINE EHOUS (A,N,D,E,E2)
DIMENSION A(9), D(3), E(3), E2(3)
DOUBLE PRECISION A, D, E, E2, ZERO, H, SCALE, ONE, SCALE1, F, G, HH
DATA ZERO / 0.0D0/, ONE / 1.0D0/
NP1 = N + 1
NN = (N * NP1) / 2 - 1
NBEG = NN + 1 - N
DO 70 II = 1, N
I = NP1 - II
L = I - 1
H = ZERO
SCALE = ZERO
IF (L .LT. 1) GO TO 10
NK = NN
DO 5 K = 1, L
SCALE = SCALE + DABS(A(NK))
NK = NK - 1
5 CONTINUE
IF (SCALE .NE. ZERO) GO TO 15
10 E(I) = ZERO
E2(I) = ZERO
GO TO 65
15 NK = NN
SCALE1 = ONE / SCALE
DO 20 K = 1, L
A(NK) = A(NK) * SCALE1
H = H + A(NK) * A(NK)
NK = NK - 1
20 CONTINUE
E2(I) = SCALE * SCALE * H
F = A(NN)
G = -DSIGN(DSORT(H), F)
E(I) = SCALE * G
H = H - F * G
A(NN) = F - G
IF (L .EQ. 1) GO TO 55
F = ZERO
JK1 = 1
DO 40 J = 1, L
G = ZERO
IK = NBEG + 1
JK = JK1
DO 25 K = 1, J
G = G + A(JK) * A(IK)
JK = JK + 1
40 CONTINUE
55 E2(I) = SCALE * SCALE * H
F = A(NN)
G = -DSIGN(DSORT(H), F)
E(I) = SCALE * G
H = H - F * G
A(NN) = F - G
IF (L .EQ. 1) GO TO 55
F = ZERO
JK1 = 1
DO 40 J = 1, L
G = ZERO
IK = NBEG + 1
JK = JK1
DO 25 K = 1, J
G = G + A(JK) * A(IK)
JK = JK + 1
40 CONTINUE
55
CONTINUE
JP1 = J+1
IF (L .LT. JP1) GO TO 35
JK = JK+J-1
DO 30 K = JP1, L
   G = G+A(JK)*A(IK)
   JK = JK+K
   IK = IK+1
30    CONTINUE
35    E(J) = G/H
      F = F+E(J)*A(NBEG+J)
      JK1 = JK1+J
40    CONTINUE
   HH = F/(H+H)
   JK = 1
   DO 50 J = 1, L
      F = A(NBEG+J)
      G = E(J)-HH*F
      E(J) = G
      DO 45 K = 1, J
         A(JK) = A(JK)-F*E(K)-G*A(NBEG+K)
         JK = JK+1
45     CONTINUE
50     CONTINUE
55     DO 60 K = 1, L
      A(NBEG+K) = SCALE*A(NBEG+K)
60     CONTINUE
65    D(I) = A(NBEG+I)
      A(NBEG+I) = H*SCALE*SCALE
      NBEG = NBEG-I+1
      NN = NN-I
70    CONTINUE
RETURN
END

SUBROUTINE EQRT2S (D, E, N, IZ, IER)
DIMENSION D(3), E(3), Z(3, 3)
DATA ZERO, ONE / 0.0, 1.0/
IER = 0
RDELP=2.**(52)
DO 5 I=2, N
   E(I-1)=E(I)
5  CONTINUE
E(N) = ZERO
B = ZERO
F = ZERO
DO 60 L=1, N
   J = 0
   H = RDELP*(DABS(D(L))+DABS(E(L)))
   IF (B.LT.H) B = H
   DO 10 M=L, N
      K=M
      IF (DABS(E(K)).LE. B) GO TO 15
10    CONTINUE
\begin{verbatim}
15 M = K
   IF (M.EQ.L) GO TO 55
20 IF (J.EQ. 30) GO TO 85
   J = J+1
   L1 = L+1
   G = D(L)
   P = (D(L1)-G)/(E(L)+E(L))
   R = DSQRT(P*P+ONE)
   D(L) = E(L)/(P+DSIGN(R,P))
   H = G-D(L)
   DO 25 I = L1,N
      D(I) = D(I)-H
   CONTINUE
   F = F+H
   P = D(M)
   C = ONE
   S = ZERO
   MM1 = M-1
   MM1PL = MM1+L
   IF (L.GT.MM1) GO TO 50
   DO 45 II=L,MM1
      I = MM1PL-II
      G = C*E(I)
      H = C*P
      IF (DABS(P).LT.DABS(E(I))) GO TO 30
      C = E(I)/P
      R = DSQRT(C*C+ONE)
      E(I+1) = S*P*R
      S = C/R
      C = ONE/R
      GO TO 35
   30 C = P/E(I)
      R = DSQRT(C*C+ONE)
      E(I+1) = S*E(I)*R
      S = ONE/R
      C = C*S
   35 P = C*D(I)-S*G
      D(I+1) = H+S*(C*G+S*D(I))
   IF (IZ .LT. N) GO TO 45
   DO 40 K=I,N
      H = Z(K,I+1)
      Z(K,I+1) = S*Z(K,I)+C*H
      Z(K,I) = C*Z(K,I)-S*H
   CONTINUE
   45 CONTINUE
   50 E(L) = S*P
      D(L) = C*P
      IF (DABS(E(L)) .GT. B) GO TO 20
   55 D(L) = D(L) + F
   60 CONTINUE
   DO 80 I=1,N
      K = I
      P = D(I)
      IP1 = I+1
      IF (IP1.GT.N) GO TO 70
\end{verbatim}
DO 65 J=IP1,N
   IF (D(J) .GE. P) GO TO 65
   K = J
   P = D(J)
65 CONTINUE
70 IF (K.EQ.I) GO TO 80
   D(K) = D(I)
   D(I) = P
   IF (IZ .LT. N) GO TO 80
   DO 75 J = 1,N
      P = Z(J,I)
      Z(J,I) = Z(J,K)
      Z(J,K) = P
75 CONTINUE
80 CONTINUE
GO TO 9005
85 IER = 128+L
9000 CONTINUE
9005 RETURN
END

DOUBLE PRECISION FUNCTION F(A,X,Y)
IMPLICIT REAL*8 (A-H,P-Z)
REAL*8 X(600),Y(600),XX(600)
COMMON DELT,El,E2,MAXIT,MPRE,MPOST,XD,YD,T3LIM,TOL
   YBAR=0.0D0
   DO 5 I=1,MPOST
      XX(I)=DEXP(A*X(I))
   5   F=0.0D0
      IF (A.EQ.0.0D0) GO TO 100
      CALL SREG(MPOST,XX,Y,BINT,BSLP)
   DO 10 I=1,MPOST
      F=F+(Y(I)-BINT-BSLP*XX(I))**2
   10 CONTINUE
   RETURN
100 YBAR=YBAR/MPOST
   DO 20 I=1,MPOST
      F=F+(Y(I)-YBAR)**2
   20 RETURN
END
**Title:** A NONLINEAR REGRESSION PROCEDURE FOR EVOKE POTENTIAL DATA ANALYSIS

**Authors:** Carl A. Mauro

**Performing Organization:** Desmatics, Inc.
P. O. Box 618
State College, PA 16804

**Contract or Grant Number(s):** N00014-79-C-0128

**Performing Organization Report Number:**

**Report Date:** June 1985

**Number of Pages:** 41

**Security Classification:** Unclassified

**Distribution Statement:** Distribution of this report is unlimited.

**Key Words:** Impact Acceleration Injury, Exponential Decay, Evoked Potential, Nonlinear Regression

**Abstract:**

The recording and subsequent analysis of evoked potential activity has proven useful for the evaluation of neural dysfunction resulting from impact acceleration injury involving the head and neck. In animal impact acceleration experiments involving Rhesus monkeys, somatosensory evoked potentials showed an increase in latency following nonlethal experiments. In order to assess quantitatively and objectively the amplitude and duration of the latency effect following impact acceleration, a nonlinear mathematical model has been proposed. (over)
20) This technical report describes a nonlinear regression procedure for fitting the proposed model directly to empirical latency data. A FORTRAN computer program listing is provided.