To the Graduate Council:

I am submitting herewith a thesis written by Jane Elizabeth Rall entitled "Interactive Least Squares." I recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mathematics.

R. E. Funderlic

Robert E. Funderlic, Major Professor

We have read this thesis
and recommend its acceptance:

Randall E Cline

Accepted for the Council:

L Evans Peck

Vice Chancellor
Graduate Studies and Research

Thesis
T
R352
cop. 2

INTERACTIVE LEAST SQUARES

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Jane Elizabeth Rall

December 1977

1346265

ACKNOWLEDGMENTS

The author would like to thank her major professor, Dr. Robert E. Funderlic, for his interest, his counsel and assistance throughout the course of this project. I also wish to thank several staff members of the Oak Ridge National Laboratory for their assistance: Tom Hebble, for his suggestions and statistical expertise; Sarah Watson, for her thoughtful comments from a user's point of view; Bert Rust, for his suggestions on interactive programming; and Gary Westley, for his advice and loan of materials. Finally, I wish to thank my husband, Bill, for his comments and help in preparing this manuscript, and his support during the course of this work.

ABSTRACT

An interactive program has been developed for solving either linear or nonlinear least squares problems. Designed for use from a remote computer terminal, the program provides the user with instant feedback and the opportunity to make changes or corrections during execution of the program.

The least squares method used is based on mathematical work by Golub and Pereyra [10]. It takes special advantage of the structure of the model equation by treating the unknown parameters as "linear" or "nonlinear" depending on how they appear in the model. Initial estimates need to be provided only for the nonlinear parameters.

The program is especially designed for use by people with a minimal knowledge of computer programming.

TABLE OF CONTENTS

CHAPTER		PAGE
INTRODUCTION		1
I.	THE METHOD OF LEAST SQUARES	2
	An Introduction	2
	Linear Least Squares	4
	Nonlinear Least Squares	7
	Weighted Least Squares	9
II.	THE VARIABLE PROJECTION METHOD AND ITS IMPLEMENTATION	11
	The Variable Projection Functional	11
	Implementing the Marquardt Method	12
	Calculation of the Variable Projection Functional and its Derivative	13
	The Two-Stage Factorization Process	14
	The Variable Projection Algorithm	16
III.	DESCRIPTION OF THE PROGRAM PACKAGE	18
	The Main Program	18
	The Subroutines	24
	Variables	25
	Coding Restrictions	39
	Storage and Time Considerations	39
IV.	USING THE PROGRAM	43
	A Sample Problem	43
	Writing the Subroutines CALC, DERIV and WT	43
	Execution of the Program	46
	Statistical Considerations	61

CHAPTER PAGE

LIST OF REFERENCES 65

APPENDIXES 68

APPENDIX A. LISTINGS OF THE FORTRAN PROGRAMS 69

APPENDIX B. DESCRIPTION OF THE VARPRO SUBROUTINES. 124

APPENDIX C. CHANGING THE CODING RESTRICTIONS 128

VITA 136

LIST OF TABLES

TABLE		PAGE
1.1	Computations Required for Linear Techniques	7
3.1	List of Command Codes	20
3.2	Description of Variables.	27
3.3	Times Required for Sample Problems.	42

LIST OF FIGURES

FIGURE		PAGE
3.1	Flow Chart of Initialization Segment	19
3.2	Flow Chart of Command Segment	21
3.3	Hierarchy of Subroutines	26
4.1	Example of Subroutines CALC, DERIV and WT	45
4.2	A Sample Terminal Session	48

INTRODUCTION

The program package described herein can be used for solving either linear or nonlinear least squares problems. It has been written, for use in an interactive mode, in FORTRAN for the DEC PDP-10, a remote terminal system that is available both at the University of Tennessee and at the Oak Ridge National Laboratory. It is designed for relatively small curve-fitting problems and is written for use by people with a minimal knowledge of computers.

Several least squares methods were considered in developing this program package. These are described in Chapter I, which provides a general introduction to the subject of least squares.

The method selected for use in this program package is described in detail in Chapter II. This method takes special advantage of the structure of the model by separating the unknown parameters into those which appear linearly and those which appear nonlinearly. Initial estimates need only be provided for the nonlinear parameters. The program incorporates much of the routine VARPRO, written by John Bolstad at Stanford University [3].

A detailed description of the program package is found in Chapter III. It includes particular information about the main program, the provided subroutines, the coding restrictions and storage and time considerations.

Chapter IV should prove the most useful to someone who is contemplating using the program. It contains a detailed description of the user-supplied subroutines, a sample terminal session and some comments on statistical considerations.

CHAPTER I

THE METHOD OF LEAST SQUARES

An Introduction

The method of least squares can be used for a variety of problems encountered in science, engineering and statistics. By far the most common application is to problems of curve fitting.

A scientist might run an experiment making observations y_i at various times, temperatures or other independent variables t_i. In order to reduce the data to a more convenient form, and possibly use it to predict future observations, it is often useful to find a model equation that describes the given set of observations. By various techniques including plotting the data and using information of the physical nature of the problem, one can often get a general idea of the class of function which should be used for the model and what parameters need to be found.

For example, the fraction y of a radioisotope remaining t seconds after the isotope's formation is given by $y = e^{xt}$ where the parameter x is a physical constant proportional to the instantaneous rate of decay of the isotope. The magnitude of x is unknown, but can be estimated by finding the function which fits the data as well as possible [4].

There are several ways one might approach the problem of finding the values of the unknown parameters, one of which is the method of least squares. Generally, this method seeks to minimize the sum of squares of the difference between the observed value and the value predicted by the model equation at each observation point. (The

difference is called the residual.) That is, one seeks to minimize the residual sum of squares

$$\psi(x) = \sum_{i=1}^{m} (y_i - \eta_i)^2 \tag{1.1}$$

where the y_i are the observed values and the η_i are the predicted values for observations i = 1 to m. Each η_i is a function of the unknown parameters x and the independent variables t_i.

Least squares problems fall into the categories of linear or nonlinear depending on how the parameters fit into the model. A model such as

$$\eta(x,t) = x_1 + x_2 t + x_3 t^3 + x_4 e^t \tag{1.2}$$

would be considered linear since the unknown parameters x_i can be written as coefficients of functions which depend only on the independent variable t. Geometrically, the model is linear in the sense that all first order partial derivatives with respect to the unknown parameters are constants (and consequently all higher order derivatives are zero). A model which cannot be written in this form would be considered nonlinear, e.g.

$$\eta(x,t) = x_1 + x_2 e^{-x_3 t} + x_4 \sin(x_5 t) + e^{-x_6 t}. \tag{1.3}$$

Some more recent techniques for solving least squares problems take better advantage of the way the unknown parameters fit into the model [10]. This is done not only by taking advantage of the sum of squares form of the problem but also by considering the models in separable form, i.e. separating the unknown parameters into ones which are linear and ones which are nonlinear. Thus we write our models in the form

$$\eta(a,\alpha;t) = \sum_{j=1}^{n} a_j \phi_j(\alpha;t) + \phi_{n+1}(\alpha;t) \tag{1.4}$$

where the linear parameters $a \in R^n$ are coefficients of functions of the nonlinear parameters $\alpha \in R^k$ and the independent variables t. The separable form can be used for considering linear problems (no unknown α's) and strictly nonlinear problems (no unknown a's) as well as a combination of the two. To emphasize the difference between the unknown linear and nonlinear parameters, example (1.3) would be rewritten as

$$\eta(a,\alpha;t) = a_1 + a_2 e^{-\alpha_1 t} + a_3 \sin(\alpha_2 t) + e^{-\alpha_3 t}.$$

Linear Least Squares

The linear least squares problem is often written in the following form:

(1.5) Find $\hat{x}$ such that $||A\hat{x} - y|| = \min_{x \in R^n} ||Ax - y||$,

where A is the mxn matrix of the functions of the independent variables (i.e. the coefficients of the unknown parameters) at each point, x is the n-vector of unknown parameters and y is the m-vector of observed values. The notation $||\cdot||$ will refer to the Euclidean norm throughout this paper. That is,

$$||x||^2 = x^T x = \sum_{i=1}^{m} x_i^2.$$

For the linear example (1.2) the i-th row of the matrix A would be $(1, t_i, t_i^3, e^{t_i})$ and the vector x would be $(x_1, x_2, x_3, x_4)^T$. A variety of methods have been developed for solving problem (1.5). Four of these will be discussed briefly here.

(1) Singular Value Decomposition:

Find the orthogonal matrices U and V such that

$$U^T A V = \text{diag}(\mu_i)$$

where μ_i are the singular values of A and are ordered so that

$\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r$ (where r is the rank of A). Then

$$\hat{x} = \sum_{i=1}^{r} v_i (u_i^T y)/\mu_i$$

where the vectors u_i and v_i are the i-th columns of U and V respectively.

(2) QR Methods:

Find the orthogonal matrix Q such that

$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix}$$

where R is nxn upper triangular. Since orthogonal matrices preserve norms (i.e. $Q^T Q = I$ implies $||Qy|| = ||y||$),

$$\begin{aligned} ||Ax - y||^2 &= ||Q^T Ax - Q^T y||^2 \\ &= \left|\left|\left| \begin{pmatrix} R \\ 0 \end{pmatrix} x - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right|\right|\right|^2 \\ &= ||Rx - b_1||^2 + ||b_2||^2. \end{aligned}$$

Then

$$\hat{x} = R^{-1} b_1$$

and one can solve a triangular system of equations for $\hat{x}$ [11]. The QR method can also be used with pivoting. Then

$$Q^T AP = \begin{pmatrix} R \\ 0 \end{pmatrix}$$

where P is a product of permutation matrices.

(3) Normal Equations Method:

If A has full rank, then

$$\hat{x} = (A^T A)^{-1} A^T y. \tag{1.6}$$

This arises from the fact that if $\hat{x}$ is a solution of the linear least squares problem (1.5), the residual vector $r = A\hat{x} - y$ satisfies the relation $A^T r = 0$, since the residual lies in the orthogonal complement

of the range of A [20]. More generally, for A of arbitrary rank,

$$\hat{x} = A^{+}y$$

where A^{+} is the Moore-Penrose inverse of A [2]. Thus to find $\hat{x}$ using the normal equations, one could: (i) Form $C = A^{T}A$; (ii) Factor $C = LL^{T}$; and (iii) Solve $Lz = A^{T}y$ and $L^{T}x = z$, where L is the lower triangular Cholesky factor [9].

(4) Gram-Schmidt Methods:

Compute the factorization A = GS where G has orthogonal columns and S is an nxn upper triangular matrix with ones on the diagonal. Then from (1.6)

$$\hat{x} = S^{-1}D^{-1}G^{T}y$$

where $D = \text{diag}(||g_i||^2) = G^{T}G$. The modified Gram-Schmidt algorithm finds S a row at a time and has stability properties similar to the QR method [21].

If one considers the sensitivity of the least squares problem, one might compare the solution $x = A^{+}y$ to the solution of a slightly perturbed problem $\bar{x} = (A+E)^{+}y$. It can be shown that the relative error in x depends on the condition number of the matrix A. In fact

$$\frac{||x - \bar{x}||}{||x||} = 2K(A)\frac{||E_1||}{||E||} + 4K^2(A)\frac{||E_2||}{||A||}\cdot\frac{||y_2||}{||y_1||}$$

where the condition number $K(A) \equiv ||A||\cdot||A^{+}||$ and where the projections of E and y onto the range of A are E_1 and y_1 and onto the orthogonal complement of the range of A are E_2 and y_2. Moreover, when $r = Ax - y$ is not zero, the $K^2(A)$ term is dominant in the upper bound [20].

In using the normal equations method it is necessary to form the matrix $A^{T}A$. But $K(A^{T}A) = K^2(A)$, significantly enlarging the upper bound on the relative error. This fact is often given as a disadvantage of

the normal equations approach. The more modern methods do not form the matrix A^TA and thus avoid the problem.

If one considers the amount of computation required by each of the algorithms described above one would find that, ignoring low order terms, the number of multiplicative operations would be as in Table 1.1 [15].

TABLE 1.1

COMPUTATIONS REQUIRED FOR LINEAR TECHNIQUES

Method	No. of Multiplications
(1) Singular Value Decomposition	$2mn^2 + 4n^3$
(2) QR Method (Using Householder Transformations)	$mn^2 - n^3/3$
(3) Normal Equations Method	$mn^2/2 + n^3/6$
(4) Modified Gram-Schmidt	mn^2

Thus the QR algorithm appears to be the best on the basis of stability and the amount of work required and was therefore selected for use in the linear portion of the program package described herein.

Nonlinear Least Squares

Nonlinear least squares problems are often written in the form

$$\psi(x) = r^Tr = \sum_{i=1}^{m} r_i^2$$

where $r_i = y_i - \eta(x,t_i)$.

Nonlinear least squares procedures are iterative since in general there is no rational inverse to nonlinear operators. Beginning from an

initial guess x_i one seeks to determine a vector v_i in the direction of the proposed i-th step and a scalar ρ_i such that $h = \rho_i v_i$ is acceptable, i.e.

$$\psi(x_{i+1}) < \psi(x_i)$$

where

$$x_{i+1} = x_i + \rho_i v_i = x_i + h. \tag{1.7}$$

The process continues until some termination criterion is met. The various methods differ in the manner in which the v_i and ρ_i are chosen. A brief discussion of several of these follows.

If we define the gradient vector

$$q(x) \equiv \nabla\psi(x) = J^T r$$

where the k,j element of J is $\partial r_k/\partial x_j$, i.e.

$$J = \left(\frac{\partial r_k}{\partial x_j} \right)$$

then v is an acceptable direction if $q_i^T < 0$. Thus v is acceptable if and only if there is a positive definite matrix R so that $v = -Rq_i$ (i.e. v leads downhill if it forms an angle of greater than 90° with the gradient q_i). Thus gradient methods include iterations of the form

$$x_{i+1} = x_i - \rho_i R_i q_i$$

where the R_i and q_i are chosen according to some scheme [1].

Several techniques are based on Newton's method, which arise from the Taylor series expansion of $\psi(x)$ [21]. The Newton-Raphson method has $R_i = H_i^{-1}$ where

$$H_i \equiv J^T J + \sum_{i=1}^{m} r_i G_i \qquad \text{and} \qquad G \equiv \left(\frac{\partial r_k}{\partial x_i \partial x_j} \right)$$

(i.e. $H_i = \psi''(x)$). Each iteration would then be

$$x_{i+1} = x_i - \rho_i H_i^{-1} q_i.$$

Since $h = x_{i+1} - x_i$ (from (1.7)) we could say

$$h = -\rho_i (J^T J + \sum_{i=1}^{m} r_i G_i)^{-1} J^T r.$$

Thus at each step we need to find the $\hat{h}$ which minimizes

$$||Jh + r||^2 + h^T (\sum_{i=1}^{m} r_i G_i) h.$$

The Gauss-Newton algorithm can be derived from the Newton-Raphson method by ignoring the term

$$\sum_{i=1}^{m} r_i G_i.$$

This approach seems reasonable since the function values r_i should be small close to the solution. The Gauss-Newton iteration would then be

$$x_{i+1} = x_i - \rho_i J^+ r$$

(where again J^+ is the Moore-Penrose inverse of J). Each step of the Gauss-Newton algorithm would involve finding $\hat{h}$ to minimize

$$||Jh + r||^2.$$

Difficulties can arise with the Gauss-Newton method at a point where the Jacobian J is rank deficient when J^+ is assumed to be $(J^T J)^{-1} J^T$. The adding of a positive definite matrix νF to the $J^T J$ term circumvents the problem [19]. This is the Levenberg-Marquardt method, or

$$x_{i+1} = x_i - (J^T J + \nu F)^{-1} J^T r.$$

(Note that $\rho_i = 1$ since the step size can be determined indirectly through the choice of the auxiliary (Marquardt) parameter ν). At each iteration we need to find $\hat{h}$ that will yield

$$\min (||Jh + r||^2 + \nu ||Fh||^2) = \min \left|\left| \begin{pmatrix} J \\ \nu F \end{pmatrix} h + \begin{pmatrix} r \\ 0 \end{pmatrix} \right|\right|^2.$$

Weighted Least Squares

It is often useful to modify the least squares problem by taking into consideration the "goodness" of the observations. Thus instead of

(1.1) we will consider

$$\psi(x) = \sum_{i=1}^{m} w_i (y_i - \eta_i)^2$$

where w_i is a positive weight assigned to the i-th observation (e.g. $w_i = 1/\sigma_i$ where σ_i is the standard deviation of the uncertainty in y_i). Thus the problem (1.5) would be to minimize $||W(Ax - y)||$ over all x in R^n, where $W = \text{diag}\ (w_i)$. The solution can be computed by any of the previously mentioned techniques if one replaces A by WA and y by Wy. Weights can also be used to solve equality constrained problems [21].

CHAPTER II

THE VARIABLE PROJECTION METHOD AND ITS IMPLEMENTATION

The Variable Projection Functional

The program package described herein is especially designed for least squares problems whose variables separate. Instead of minimizing the norm of the residual

$$\psi(a,\alpha) = \sum_{i=1}^{m} (y_i - \eta(a,\alpha;t_i))^2 \tag{2.1}$$

we make use of the variable projection functional $\psi_2(\alpha)$ described by Golub and Pereyra [10]. The modified residual sum of squares $\psi_2(\alpha)$ depends only on the nonlinear parameters. Thus we will be able to minimize first over the nonlinear parameters to obtain optimal values $\hat{\alpha}$ and reduce the problem to a linear one for determining the optimal parameters $\hat{a}$.

In order to obtain the modified residual $\psi_2(\alpha)$ we first define the matrix $\{\Phi(\alpha)\}_{ij} = \phi_j(\alpha,t_i)$ where the ϕ_j are those found in the model equation (1.4). Then equation (2.1) can be written as

$$\psi(a,\alpha) = ||y - \Phi(\alpha)a||^2. \tag{2.2}$$

If the optimal nonlinear parameters $\hat{\alpha}$ were known, linear least squares analysis would tell us that $a = \Phi^+(\alpha)\, y$. Thus substituting into (2.2) gives us

$$\begin{aligned} ||y - \Phi(\alpha)a||^2 &= ||y - \Phi(\alpha)\Phi^+(\alpha)y||^2 \\ &= ||[I - \Phi(\alpha)\Phi^+(\alpha)]y||^2 \\ &= ||P_{\Phi(\alpha)}^{\perp} y||^2 \\ &= ||r_2(\alpha)||^2 \\ &\equiv \psi_2(\alpha) \end{aligned}$$

where $||\cdot||$ represents the Euclidean norm and $P_{\Phi(\alpha)}^{\perp}$ is the projector on the orthogonal complement of the column space of $\Phi(\alpha)$.

Implementing the Marquardt Method

We are thus faced with the problem of minimizing $\psi_2(\alpha) = ||P_{\Phi(\alpha)}^{\perp} y||^2$. Since the residual depends strictly on nonlinear parameters it is necessary to select one of the nonlinear least squares algorithms. A variation of Osborne's modification [18] of the Marquardt method was chosen for use in this program package.

For our problem, $J = Dr_2(\alpha)$ and $r = r_2(\alpha)$, where D indicates the Frechet derivative [17]. We define the partitioned matrices and vectors

$$K_i \equiv \begin{pmatrix} Dr_2(\alpha_i) \\ \nu_i F_i \end{pmatrix} \Big\} k \quad \text{and} \quad \hat{r}_i \equiv \begin{pmatrix} r_2(\alpha_i) \\ 0 \end{pmatrix} \Big\} k$$

where k is the number of nonlinear parameters and F is a diagonal matrix consisting of the column lengths of $Dr_2(\alpha)$. Thus at each iteration we need to find the h_i which minimizes

$$||K_i h_i + \hat{r}_i||^2,$$

which is a problem of linear least squares. That is, we need to find $h_i = -K_i^{+} \hat{r}_i$.

Therefore, each step of the Marquardt procedure involves solving the linear least squares problem

$$Kh = \begin{pmatrix} -r_2(\alpha) \\ 0 \end{pmatrix} \tag{2.3}$$

for h, steping to the new point $\alpha_{i+1} = \alpha_i + h$ and testing to see whether the new norm of the residual is acceptable. If it is (i.e. the new norm is smaller than the previous one), the step is accepted and ν_{i+1} is set to $.5\nu_i$. If not the step is retracted and ν_{i+1} is enlarged (set to $1.5\nu_i$).

A two-stage factorization process is used in solving for h so that certain information can be saved in case a step is retracted.

Calculation of the Variable Projection Functional and its Derivative

Before beginning the two-stage factorization process it is necessary to calculate $r_2(\alpha)$ and its derivative. Calculation of the modified residual $r_2(\alpha)$ involves determining the orthogonal matrix $Q(\alpha)$ used in reducing $\Phi(\alpha)$ to upper triangular form and computing

$$\bar{y} = Q(\alpha)y = \begin{pmatrix} \bar{y}_1 \\ \bar{y}_2 \end{pmatrix} \begin{matrix} \} \ n \\ \} \ m-n \end{matrix}$$

Then the residual vector is

$$r_2(\alpha) = \begin{pmatrix} \bar{y}_2 \\ 0 \end{pmatrix}.$$

Calculation of the derivatives is somewhat more complicated. Golub and Pereyra [10] provide an extensive discussion of the Frechet derivative of orthogonal projectors which leads to expressions for the derivatives of the vector $r_2(\alpha) = P_{\Phi(\alpha)}^{\perp} y$ and that of the pseudo-inverse of $\Phi(\alpha)$. They derive the expression

$$(2.4) \qquad Dr_2(\alpha) = P_{\Phi(\alpha)}^{\perp} y = -(P_{\Phi}^{\perp} D\Phi)\Phi^{B} y - (\Phi^{B})^{T}(P_{\Phi}^{\perp} D\Phi)^{T} y$$

where Φ^{B} is the matrix

$$P \begin{pmatrix} T_{11}^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q$$

in which the orthogonal matrix Q and the permutation matrix P are those used to reduce Φ to upper triangular form T_{11}. Kaufman [13] has proposed that the second term of the expression in (2.4) is unnecessary, thus reducing the amount of computation required.

The basic procedure used in subroutines VARPRO et. al. to calculate both $r_2(\alpha)$ and $Dr_2(\alpha)$ is that of Kaufman's algorithm. Although the theory

has been developed for any matrix of constant rank, the subroutines assume the matrix to be of full rank. The steps would be as follows:

(1) Compute $\Phi(\alpha)$ and $D\Phi(\alpha)$ and form the $m \times (n+p+1)$ array

$$G \equiv \left(\Phi(\alpha) \; ; \; y \; ; \; D\Phi(\alpha) \right)$$

where $D\Phi(\alpha)$ contains only the p nonzero columns of partial derivatives (i.e. all zero columns have been omitted).

(2) Use Householder reflections to obtain the orthogonal matrix Q (in factored form) and the permutation matrix P such that

$$Q\Phi P \equiv \begin{pmatrix} T_{11} \\ 0 \end{pmatrix}$$

where T_{11} is $n \times n$ upper triangular.

(3) Set

$$v = \begin{pmatrix} v_1 \} & n \\ v_2 \} & m-n \end{pmatrix} = Qy.$$

Then

$$\psi_2(\alpha) = ||r_2(\alpha)||^2 = ||v_2||^2.$$

(4) Solve $T_{11}b = v_1$ for b by back substitution; then set $c = Pb$.

(5) Set $W = -QD\Phi(\alpha)c$ and partition

$$W = \begin{pmatrix} W_1 \} & n \\ W_2 \} & m-n \end{pmatrix}.$$

Then $Dr_2(\alpha) = W_2$. Thus the matrix K used in the Marquardt iteration would be

$$K = \begin{pmatrix} W_2 \\ \nu F \end{pmatrix}.$$

The Two-Stage Factorization Process

The two-stage factorization process used in solving (2.3) has been modified from Osborne's general procedure [18] by Golub and Pereyra [10] to take advantage of the structure of the problem. If we set $DP = Dr_2(\alpha)$,

the first stage consists of determining an mxm orthogonal matrix Q (using Householder reflections to obtain it in factored form) so that

$$QDP = R_1 = \begin{pmatrix} R_1' \\ 0 \end{pmatrix}$$

where R_1' is nxn upper triangular. Then if we define

$$Q_1 \equiv \begin{pmatrix} Q & 0 \\ 0 & I_k \end{pmatrix}$$

it is clear that

$$Q_1 K = Q_1 \begin{pmatrix} DP \\ \nu F \end{pmatrix} = \begin{pmatrix} R_1' \\ \nu F \end{pmatrix}.$$

Also define $\bar{r}$ by

$$\begin{pmatrix} \bar{r} \\ 0 \end{pmatrix} \equiv Q_1 \begin{pmatrix} -r_2(\alpha) \\ 0 \end{pmatrix}.$$

R_1' and $\bar{r}$ are saved for future use (in case a step is retracted).

In the second stage we determine an (m+k)x(m+k) orthogonal matrix Q_2 (again using Householder reflections) such that

$$Q_2 \begin{pmatrix} R_1 \\ \nu F \end{pmatrix} = \begin{pmatrix} R_2 \\ 0 \end{pmatrix}$$

where R_2 is in upper triangular form. (Here especially we take advantage of the structure of R_1 and of νF to eliminate computation when a band of zeros is preserved.)

Then set

$$Q_2 \begin{pmatrix} \bar{r} \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{r} \\ 0 \end{pmatrix}$$

and solve the upper triangular system

$$Q_2 Q_1 Kh = Q_2 Q_1 \begin{pmatrix} -r_2(\alpha) \\ 0 \end{pmatrix}$$

or equivalently, $R_2 h = \hat{r}$, to determine the step h to be taken.

The Variable Projection Algorithm

The basic algorithm used in the routines VARPRO et. al. consists of the following steps:

(1) Set $\nu = 1$.

(2) Begin outer iteration loop to update α (the vector of nonlinear parameters). Calculate the norm of $r_2(\alpha)$ and the derivative of $r_2(\alpha)$ during the first iteration; calculate only the derivative in subsequent iterations. (The norm and the derivative are calculated in subroutine DPA.)

(3) Begin two-stage orthogonal factorization. (The first reduction is done in subroutine ORFAC1 and the upper triangular form and residual are saved.)

(4) Begin inner loop for generating new α and testing it for acceptance. (The second factorization is done in subroutine ORFAC2.)

(5) Solve the upper triangular system for h. (Subroutine BACSUB does the back substitution.)

(6) Step to the new point α_{i+1} and compute the new norm of the residual, RNEW.

(7) If RNEW $\geq$ RNORM retract the step just taken:

$\nu = 1.5\nu$

Retrieve the upper triangular form and residual.

Return to step (4).

If RNEW < RNORM

Calculate norm (h) / norm (α)

$\nu = .5\nu$

If convergence criteria are not met, return to step (2).

(8) Calculate the final quantities (in subroutines DPA and POSTPR) including the linear parameters, norm of residual and estimated variance.

A further description of the subroutines ORFAC1, ORFAC2, DPA, BACSUB, POSTPR, COV, VARERR and ADA is found in Appendix B.

CHAPTER III

DESCRIPTION OF THE PROGRAM PACKAGE

The Main Program

The main program is divided into two segments. The first is the Initialization Segment in which values are entered for the number of data points, number of linear parameters, number of nonlinear parameters and number of independent variables, followed by the data points themselves (independent variables, dependent variables and weights), initial values for the nonlinear parameters, some information about the model (which terms depend on particular nonlinear parameters) and increments for the derivatives if they are to be calculated numerically. A flow chart of the Initialization Segment is shown in Figure 3.1.

Once these values have been initialized one enters the Command Segment. Here one is given a choice of four options: (1) Print--examine the current values of the initialized data, (2) Change--correct any initialized value and either add or delete data points, (3) Run--run the least squares program (to find the optimal parameters) or run a statistical analysis of the data returned or (4) Stop--end execution of the program either with or without saving the initialized data. To indicate which step is to be performed the user should enter the appropriate command code as described in Table 3.1. (Should the table not be readily accessible one can obtain a listing of the command codes by entering H for the next step.) A flow chart of the Command Segment is found in Figure 3.2.

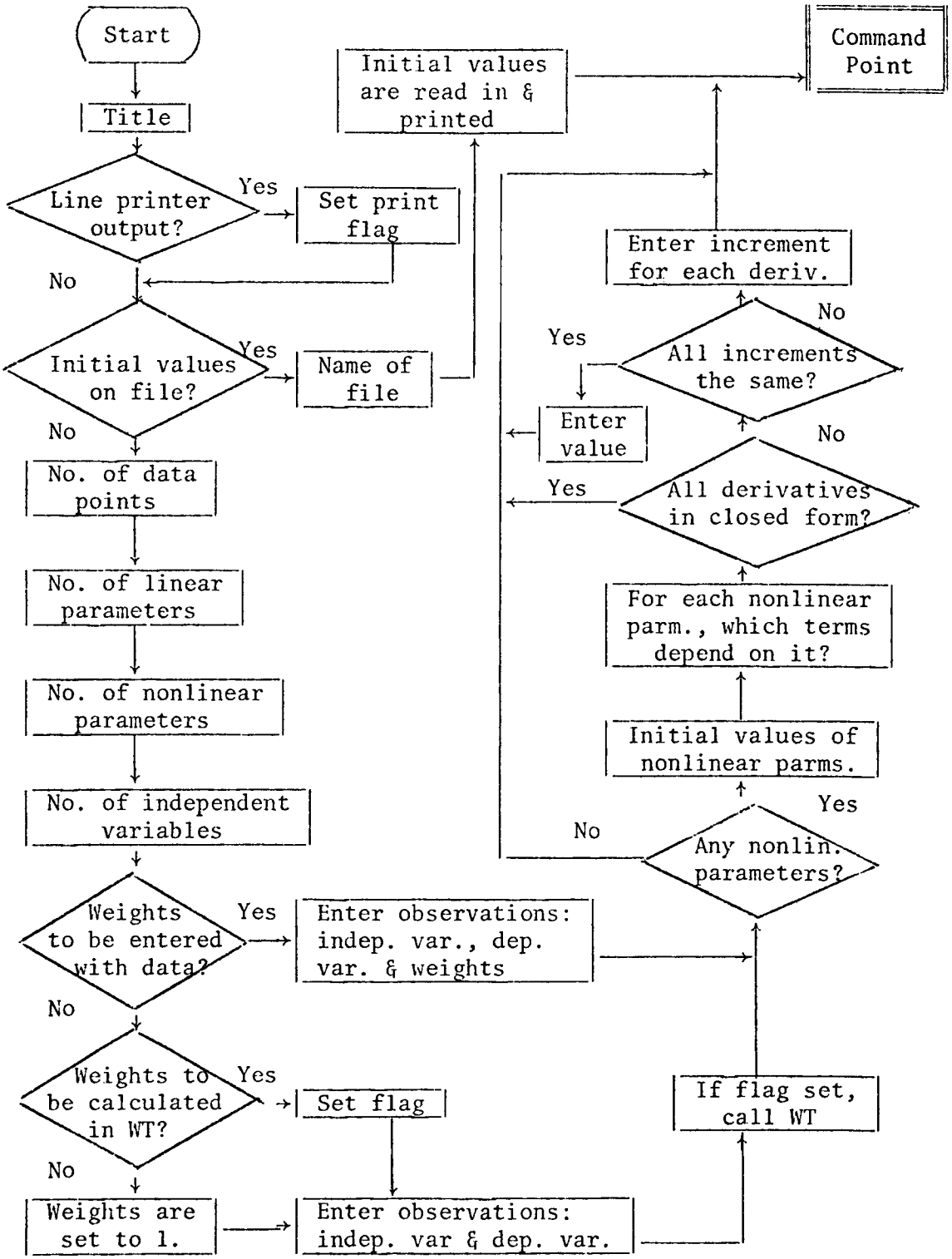


Figure 3.1

Flow Chart of Initialization Segment

TABLE 3.1

LIST OF COMMAND CODES

COMMAND CODES FOR PRINTING

PM	PRINT THE NUMBER OF DATA POINTS
PN	PRINT THE NUMBER OF LINEAR PARAMETERS
PKG	PRINT THE NUMBER OF NONLINEAR PARAMETERS
PIV	PRINT THE NUMBER OF INDEPENDENT VARIABLES
PTW	PRINT THE DATA POINTS AND WEIGHTS
PALF	PRINT THE VALUES OF THE NONLINEAR PARAMETERS
PNDI	PRINT THE NUMERICAL DERIVATIVE INCREMENTS
PINC	PRINT WHICH TERMS DEPEND ON PARTICULAR NONLINEAR PARAMETERS
LP	TURN ON THE LINE PRINTER FLAG
NP	TURN OFF THE LINE PRINTER FLAG

COMMAND CODES FOR CHANGING

CM	CHANGE THE NUMBER OF DATA POINTS
CN	CHANGE THE NUMBER OF LINEAR PARAMETERS
CKG	CHANGE THE NUMBER OF NONLINEAR PARAMETERS
CIV	CHANGE THE NUMBER OF INDEPENDENT VARIABLES
CT	CHANGE ONE OR MORE OF THE DATA POINTS
CW	CHANGE ONE OR MORE OF THE WEIGHTS
CALF	CHANGE ONE OR MORE OF THE NONLINEAR PARAMETERS
CNDI	CHANGE ONE OR MORE OR THE NUMERICAL DERIVATIVE INCREMENTS
CINC	CHANGE WHICH TERMS DEPEND ON PARTICULAR NONLINEAR PARAMETERS

COMMAND CODES FOR RUNNING

RV	RUN THE LEAST SQUARES PROGRAM
RS	RUN THE STATISTICAL ANALYSIS OF THE DATA RETURNED

COMMAND CODES FOR STOPPING

SD	SAVE THE CURRENT VALUES OF THE DATA AND STOP
SO	STOP WITHOUT SAVING ANY DATA

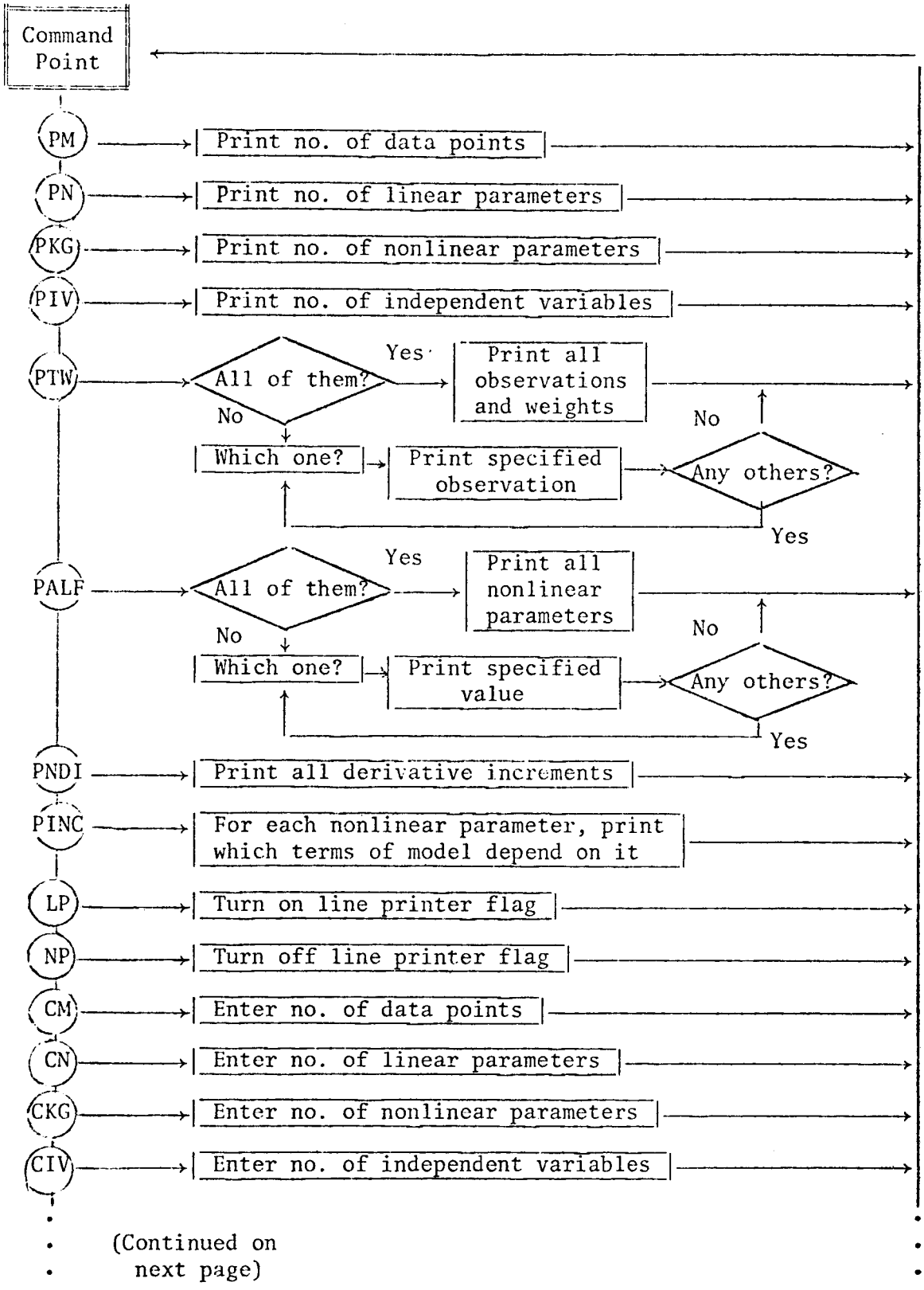


Figure 3.2

Flow Chart of Command Segment

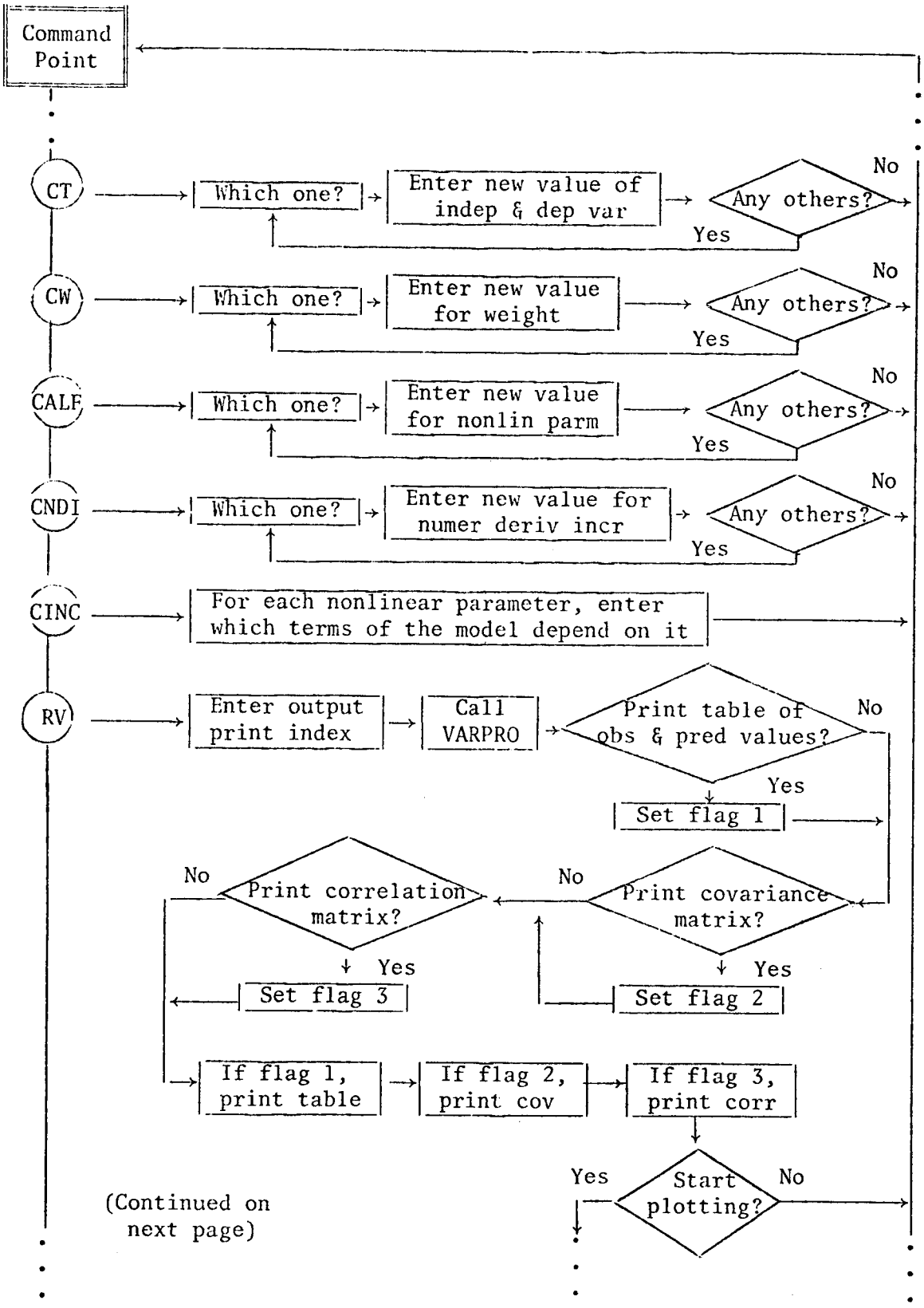


Figure 3.2 (Continued)

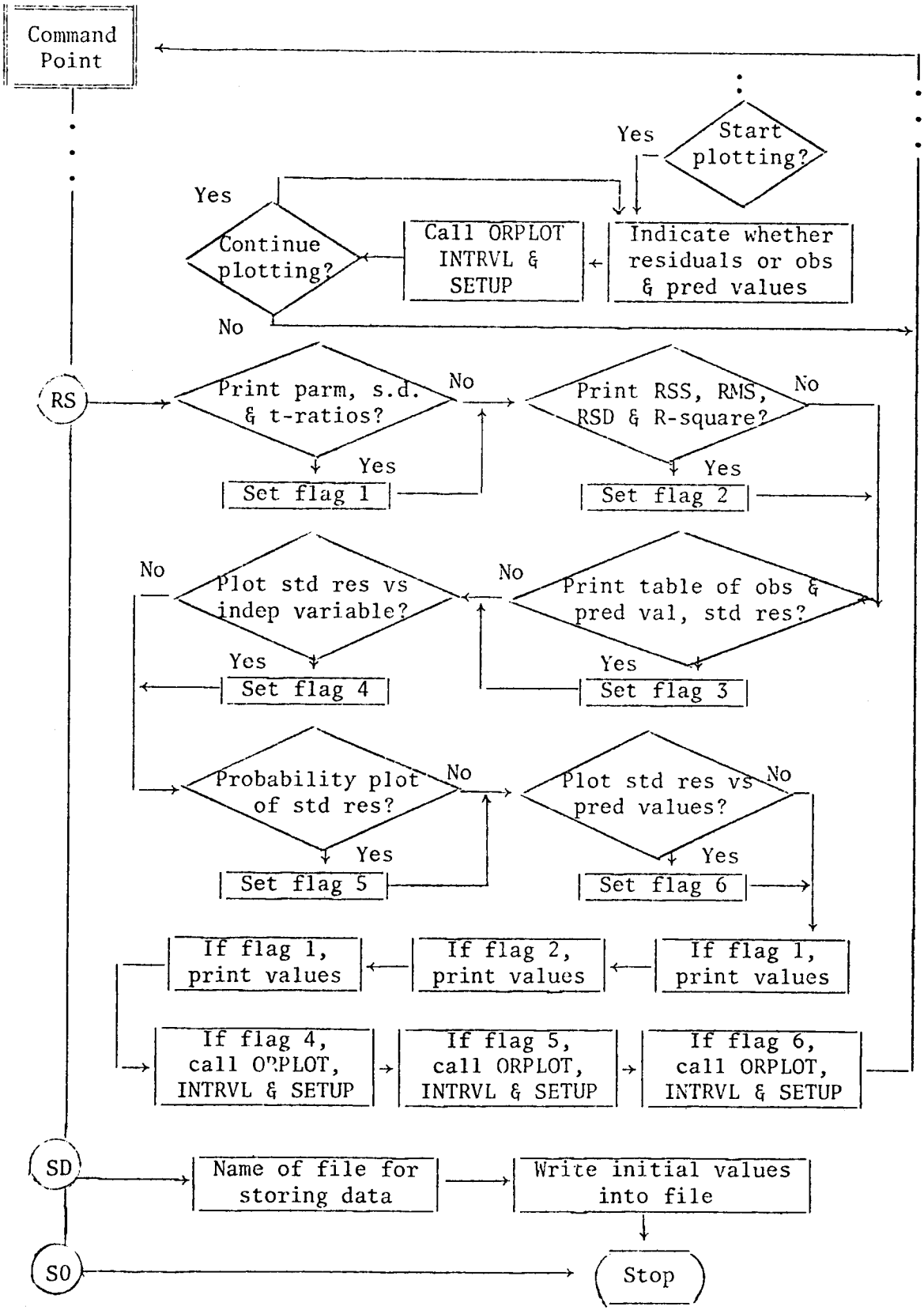


Figure 3.2 (Continued)

The Subroutines

Two types of subroutines are used in the package: (1) built-in subroutines and (2) user-supplied subroutines.

The built-in subroutines can be grouped into five task categories. The first includes subroutines VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR, COV and VARERR. These were written by John Bolstad at Stanford University and are the routines which are used to "solve" the least squares problem, i.e. to find the optimal values of the parameters. For further information about these routines one should consult their listings in Appendix A and their descriptions in Appendix B.

A second category includes the subroutines ADA and NUMDER. The subroutine ADA, which is required by VARPRO, is no longer provided by the user, but instead ADA calls the user-supplied subroutines CALC and DERIV to obtain the required user-defined information. ADA evaluates the functions ϕ and their partial derivatives $\frac{d\phi}{d\alpha}$. Subroutine NUMDER determines whether the derivatives to be calculated for ADA are to be calculated from a closed form specified in DERIV, and if not, will calculate them numerically using a forward difference quotient.

Another group of subroutines includes CH, PR and STAT. These are auxiliary to the main program and are used to execute the commands to change the values of the data points, weights, nonlinear parameters and numerical derivative increments; to print any or all of the initialized values; and to provide further statistical information about the data returned by VARPRO.

Four subroutines are used for producing the plotted output. These include ORPLOT, INTRVL, SETUP and PLOT. The subroutine ORPLOT arranges

the specified independent variables in ascending order and stores the indexing in an array. INTRVL determines the scale to be used for the independent variable, i.e. the step size between successive points. SETUP determines the actual arrays to be plotted by inserting zero values for the dependent variable at the interval step points where the true value is unknown. It also calls the last subroutine, PLOT, to do the plotting. PLOT can graph either one or two dependent variables against a specified independent variable. The plots are rotated 90° with the independent variable axis appearing vertically and the smaller values at the top. Should two or more observations have the same independent variable value, only the first will be plotted.

Finally, ARRAYD is an output printing subroutine which is used for producing the covariance and correlation matrices.

The user is asked to provide a subroutine CALC to describe the model equation to be used in the least squares fit. Subroutine DERIV should be provided if the derivatives are to be calculated from the closed form. One also has the option of using subroutine WT to calculate weights for weighted least squares problems. A further description of the user-supplied subroutines can be found in Chapter IV.

Figure 3.3 indicates the hierarchy of the subroutines.

Variables

A complete list of the variable names used in the program package is found in Table 3.2. A short description of each variable is provided along with a list of the routines in which it appears.

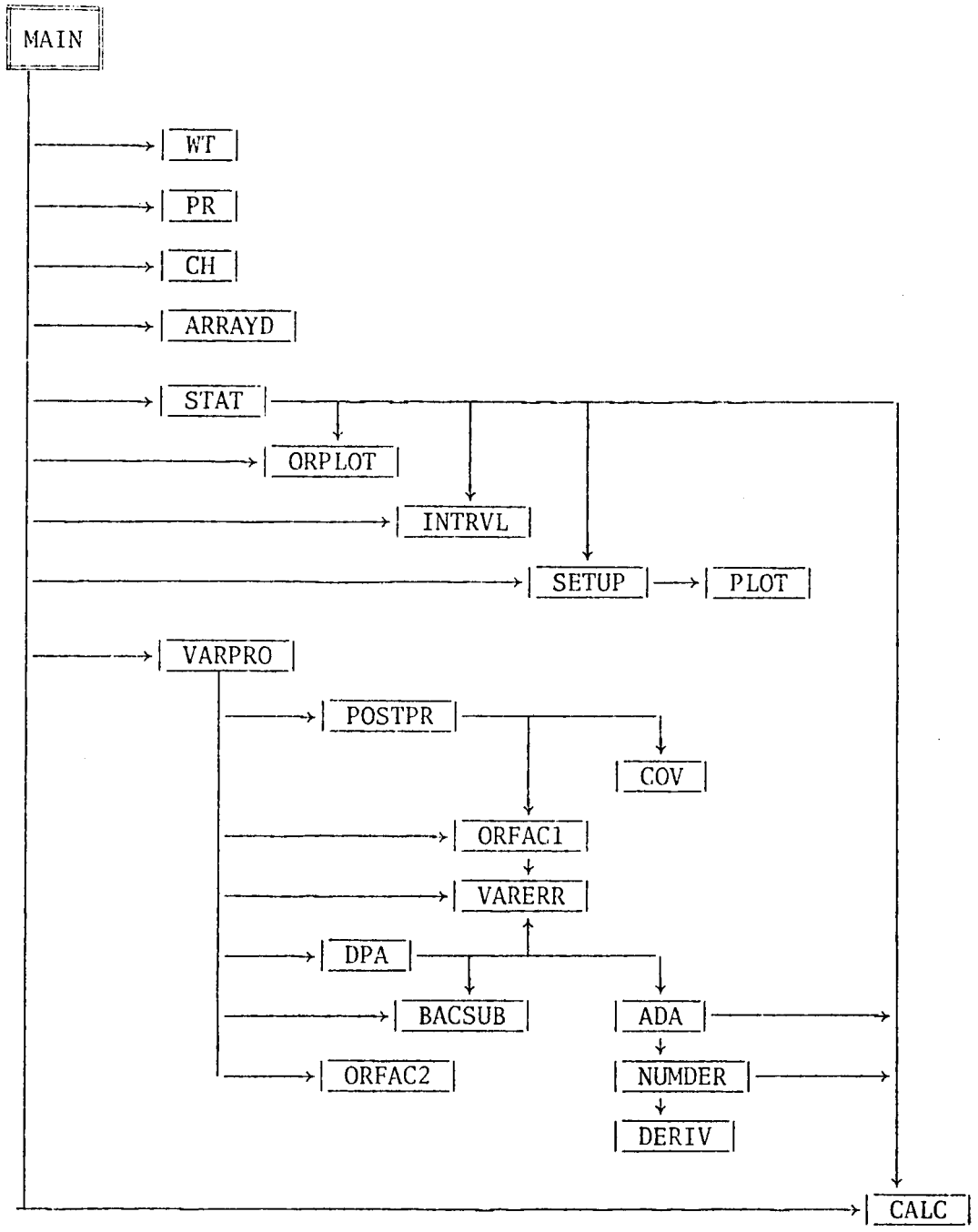


Figure 3.3

Hierarchy of Subroutines

TABLE 3.2

DESCRIPTION OF VARIABLES

Variable Name	Description
A(100,24)	On input to VARPRO the array contains the PH(J)s and their derivatives. On output, the first N+KG rows and columns contain an approximation to the (weighted) covariance matrix at the solution (the first N rows correspond to the linear parameters, the last KG to the nonlinear ones), column N+KG+1 contains the weighted residuals, A(1,N+KG+2) contains the (Euclidean) norm of the weighted residual and A(2,N+KG+2) contains an estimate of the (weighted) variance of the observations. [MAIN, VARPRO, DPA, POSTPR, COV, ADA, STAT]
ACUM	Variable used to aid in computation, e.g. accumulate a sum. [VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR]
AI	Absolute value of the I-th diagonal element of A. [STAT]
AJ	Absolute value of the J-th diagonal element of A. [STAT]
AL	Smallest value of an array. [INTRVL, PLOT]
ALF(12)	Vector of nonlinear parameters. [MAIN, CH, PR, VARPRO, DPA, POSTPR, ADA, NUMDER, STAT]
ALFNRM	Norm of ALF. [VARPRO]
ALPHA	Factor used in calculating Householder reflections. [ORFAC1, ORFAC2, DPA]
AN	Dummy variable. [STAT]
ANS	User response to question asking for double precision value. [MAIN]
ASAVE	Value of ALF(I) when computing numerical derivative. [NUMDER]
A1(100,24)	Triangular matrix used in solving system of equations. [BACSUB]
B(100,13)	Matrix which contains the derivative portion of A. [ORFAC1, ORFAC2]

TABLE 3.2 (Continued)

Variable Name	Description
BA(1)	Singly-dimensioned array to be printed as a matrix. [ARRAYD]
BB	Square root of AI or AJ. [STAT]
BETA	Factor used in calculating Householder reflections. [ORFAC1, ORFAC2, DPA]
BH	Largest value of an array. [INTRVL, PLOT]
BLKBM	Common block containing BM. [POSTPR, STAT]
BLKDPH	Common block containing DPH. [MAIN, NUMDER]
BLKINC	Common block containing INC. [MAIN, PR, DPA, ADA]
BLKNDI	Common block containing NDI. [MAIN, CH, PR, NUMDER]
BLKPH	Common block containing PH. [MAIN, ADA, STAT]
BLOKIP	Common block containing IP. [MAIN, CH, PR, VARPRO, POSTPR, VARERR, STAT, PLOT, ARRAYD]
BM(100,19)	Matrix of predictor variables used in calculating the standardized residuals. [POSTPR, STAT]
B1	N+2, used to identify the first column of A which contains derivatives of PH(J). [VARPRO]
C(7)	Vector of linear parameters. [MAIN, VARPRO, ADA, STAT]
CC(7)	Vector used to identify which PH(J) is to be used in calculating the numerical derivative. [NUMDER]
COLUMN	Data variable containing "COLUMN ". [ARRAYD]
CR(19,19)	Correlation matrix. [MAIN]
CTSS	Corrected total sum of squares. [STAT]
DALFNM	Norm of the change in ALF from the previous iteration to the present one. [VARPRO]
DELR	Change in the norm of the residual from the previous iteration to the present one. [VARPRO]

TABLE 3.2 (Continued)

Variable Name	Description
DK	Numerical derivative increment. [NUMDER]
DPH(8,12)	DPH(I,J) is the derivative of PH(I) with respect to ALF(J). [MAIN, NUMDER]
DV	Derivative value to be entered in A. [NUMDER]
EPS1	Absolute tolerance for the norm of the projection of the residual onto the range of the Jacobian of the variable projection functional. [VARPRO]
EPS2	Relative and absolute tolerance for the change in the norm of the residual. [VARPRO]
ERRNO	Error number (absolute value of IERR). [VARERR]
ETA(100)	Vector of predicted values of the dependent variable. [MAIN, ADA, STAT, SETUP]
FILE	Name of file for storing data. [MAIN]
FIRSTC	First column to which Householder reflections should be applied. [DPA]
FIRSTR	First row to which Householder reflections should be applied. [DPA]
GNSTEP	Step size for Gauss-Newton iteration. [VARPRO]
GN(100)	Vector containing differences between successive independent variable values. [INTRVL]
H	Variable used in arranging indexing of IOR. [ORPLOT]
HN	Variable used in arranging indexing of IOR. [ORPLOT]
I	Dummy subscript. [MAIN, CH, PR, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR, COV, ADA, NUMDER, STAT, ORPLOT, INTRVL, SETUP, PLOT]
IA(8)	Vector used in determining INC matrix. [MAIN, PR]
IAXIS	Character string for drawing axis in plots. [PLOT]
IB	Dummy subscript. [MAIN]

TABLE 3.2 (Continued)

Variable Name	Description
IBACK	Dummy subscript. [MAIN]
IBLANK	Character string for spacing in plots. [PLOT]
ICR	Flag to indicate correlation matrix should be printed. [MAIN]
ICROSS	Symbol "X" for plotting. [PLOT]
IERR	Integer error flag. If the value returned is ≥ 0 successful convergence. -1 terminated for too many iterations. -2 terminated for ill-conditioning (Marquardt parameter too large). -4 input error in parameter M, N, KG, NPP2, or MMAX. -5 INC matrix improperly specified. -6 weights not nonnegative. -7 "constant" column was computed more than once. -8 catastrophic failure--a column of matrix A has become zero. [MAIN, VARPRO, ORFAC1, POSTPR, VARERR]
II	Dummy subscript. [MAIN, DPA, ADA, STAT]
II1	Parameter used to place first dependent variable on plot. [PLOT]
II2	Parameter used to place second dependent variable on plot. [PLOT]
IK	Subscript used to identify placement of derivative in matrix A. [ADA]
IL	Dummy subscript. [CH, PR]
IM1	I-1 (dummy subscript). [SETUP]
IN	Subscript to identify which numerical derivative increment should be used. [ADA, NUMDER]
INC(12,8)	Incidence matrix for partial derivatives. INC(K,J)=1 if nonlinear parameter ALF(K) appears in PH(J). Otherwise, INC(K,J)=0. [MAIN, PR, DPA, ADA]
INCKJ	INC(K,J). [DPA]

TABLE 3.2 (Continued)

Variable Name	Description
IND	Data variable containing "IND ". [MAIN, PR, STAT]
INV	Number of independent variable used for plotting. [MAIN]
IOR(100)	Vector containing indexing of independent variable values arranged in ascending order. [MAIN, STAT, ORPLOT, INTRVL, SETUP]
IP	Flag to indicate line printer output. [MAIN, CH, PR, VARPRO, POSTPR, VARERR, STAT, PLOT, ARRAYD]
IPI	Flag to indicate whether plot of residuals or of predicted and observed values. [MAIN, STAT, SETUP]
IPIBLK	Common block containing IPI. [MAIN, STAT, SETUP]
IPIV	Flag to indicate plot of standardized residuals versus independent variables. [STAT]
IPLUS	Symbol "+" for plotting. [PLOT]
IPP	Flag to indicate probability plot of standardized residuals. [STAT]
IPPV	Flag to indicate plot of standardized residuals versus predicted values. [STAT]
IPRINT	Integer used to control printed output from VARPRO. If IPRINT > 0, the nonlinear parameters, norm of residual and Marquardt parameter will be printed every IPRINT-th iteration. If IPRINT = 0, only the final quantities will be printed. [MAIN, VARPRO, ORFAC1, DPA, POSTPR, VARERR]
IPT	Flag to indicate t-ratios whould be printed. [STAT]
IP1	I+1 (dummy subscript). [BACSUB, COV, INTRVL]
IQ	Flag to indicate Command Segment. [MAIN]
IR	Flag to indicate residual sum of squares, residual mean square and residual standard error, and coefficient of determination should be printed. [STAT]
ISEL	Flag to indicate what should be computed. [DPA, ADA]

TABLE 3.2 (Continued)

Variable Name	Description
ISTAR	Symbol "*" for plotting. [PLOT]
ISUB	Dummy subscript. [VARPRO]
ITB	Flag to indicate table should be printed. [MAIN, STAT]
ITER	Number of iterations. [VARPRO]
ITERIN	Control variable to determine if a step was retracted. [VARPRO]
ITMAX	Maximum number of iterations. [VARPRO]
IV	Number of independent variables per observation. [MAIN, CH, PR, VARPRO, DPA, ADA, NUMDER, STAT]
IVP	Number of columns to be printed in table. [PR]
IWT	Flag to indicate weights calculated in subroutine WT. [MAIN]
IX	Subscript used in arranging indexing of IOR. [ORPLOT]
IY(56)	Vector containing line of plot to be printed. [PLOT]
IZERO	Parameter for determining where to draw axis for plots. [PLOT]
J	Dummy subscript. [MAIN, CH, PR, VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR, COV, ADA, NUMDER, STAT, ORPLOT, SETUP, PLOT, ARRAYD]
JJ	Dummy subscript. [ADA, NUMDER]
JM1	J-1 (dummy subscript). [COV]
JP1	J+1 (dummy subscript). [POSTPR, SETUP]
JSUB	Dummy subscript. [VARPRO, ORFAC1]
K	Dummy subscript. [MAIN, PR, VARPRO, ORFAC1, ORFAC2, DPA, POSTPR, VARERR, ADA, NUMDER, STAT, INTRVL, SETUP]
KBACK	Dummy subscript. [POSTPR]

TABLE 5.2 (Continued)

Variable Name	Description
KG	Number of nonlinear parameters. [MAIN, CH, PR, VARPRO, ORFAC1, ORFAC2, DPA, POSTPR, ADA, NUMDER, STAT]
KGPK	KG+K. [ORFAC2]
KGPKM1	KG+K-1. [ORFAC2]
KGP1	KG+1. [VARPRO, ORFAC1, ORFAC2]
KG2	2(KG). [ORFAC2]
KG23	2(KG)+3. [VARPRO, ORFAC1, ORFAC2]
KK	Dummy subscript. [DPA, ADA]
KMN	Index of minimum value of independent variable. [INTRVL]
KMX	Index of maximum value of independent variable. [INTRVL]
KP1	K+1 (dummy subscript). [ORFAC1, ORFAC2, DPA, POSTPR, INTRVL, SETUP]
KSUB	Dummy subscript. [VARPRO, DPA]
L	Dummy subscript. [DPA, COV, ARRAYD]
LASTC	Last column to which Householder reflections should be applied. [DPA]
LI	Dummy subscript. [ARRAYD]
LVV	Integer first dimension in calling program of matrix to be printed. [ARRAYD]
M	Number of observations. [MAIN, CH, PR, VARPRO, ORFAC1, DPA, POSTPR, ADA, NUMDER, STAT, ORPLOT, INTRVL, SETUP]
MAX	Index corresponding to maximum value of independent variable. [ORPLOT]
MH	Highest column number to be printed on a given line. [ARRAYD]

TABLE 3.2 (Continued)

Variable Name	Description
MHM1	One less than the single dimension index corresponding to the first element of column MH. [ARRAYD]
ML	Lowest column number to be printed on a given line. [ARRAYD]
MLM1	One less than the single dimension index corresponding to the first element of column ML. [ARRAYD]
MMAX	Maximum number of observations. [MAIN, CH, PR, VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR, COV, ADA, NUMDER, STAT]
MMH	Single dimension index corresponding to column MH. [ARRAYD]
MML	Single dimension index corresponding to column ML. [ARRAYD]
MM1	M-1. [ORPLOT, INTRVL]
MODIT	Variable to determine whether information is to be printed at a given iteration (based on IPRINT). [VARPRO]
N	Number of linear parameters. [MAIN, PR, VARPRO, ORFAC1, DPA, POSTPR, ADA, NUMDER, STAT]
NAM(40)	Array which reads keywords from saved data file. [MAIN]
NAME	Heading to be used for matrix. [ARRAYD]
NC	Flag to skip unnecessary computation. [MAIN, ADA, NUMDER]
NCBLK	Common block containing NC. [MAIN, ADA, NUMDER]
NCC	Number of columns in matrix to be printed. [ARRAYD]
NCON	Number of "constant" functions in A. [DPA]
NCONP1	NCON+1. [DPA, ADA]
NDI(15)	Vector of numerical derivative increments. [MAIN, CH, PR, NUMDER]

TABLE 3.2 (Continued)

Variable Name	Description
NH	Highest row number to be printed on a page. [ARRAYD]
NKGM1	N+KG-1. [COV]
NKG1	N+KG+1. [POSTPR]
NKG2	N+KG+2. [MAIN, VARPRO, DPA, POSTPR, STAT]
NL	Lowest row number to be printed on a page. [ARRAYD]
NN	Number of points (lines) to be plotted. [SETUP, PLOT]
NPH	Subscript to identify column number for last function PH. [ADA]
NPK	N+K. [ORFAC1]
NPKG	N+KG. [MAIN, ORFAC1, POSTPR, COV, STAT]
NPP1	N+P+1. [DPA]
NPP2	N+P+2. [VARPRO, ADA]
NP1	N+1. [MAIN, PR, VARPRO, ORFAC1, DPA, BACSUB, POSTPR, ADA, NUMDER, STAT]
NP2	N+2. [DPA]
NRR	Number of rows in matrix to be printed. [ARRAYD]
NU	Marquardt parameter. [VARPRO, ORFAC2]
N1	Dimension of triangular matrix used in solving system of equations. [BACSUB]
OUTPUT	Unit number to determine where output should be printed. [VARPRO, POSTPR, VARERR]
P	Number of nonvanishing partial derivatives of the PH(J) (i.e. number of ones in INC). [MAIN, PR, DPA]
PH(8)	Vector containing the values of the functions PH(J) of the model equation. [MAIN, ADA, STAT]
PHINP1	Flag to indicate whether PH(NP1) is in the model equation. [DPA, ADA]

TABLE 3.2 (Continued)

Variable Name	Description
PI	π. [STAT]
PI4	$\pi/4$. [STAT]
PI8	$\pi/8$. [STAT]
PP	Variable used in calculating probability of standardized residuals. [STAT]
PRJRES	Norm of the projection of the residual onto the range of the Jacobian of the variable projection functional. [VARPRO, ORFAC1, POSTPR]
Q	Variable to determine whether the next independent variable value is more than one and one half step sizes away. [SETUP]
R(100)	Vector of residuals (Y - ETA). [MAIN, DPA, POSTPR, STAT]
RMS	Residual mean square. [STAT]
RNEW	Norm of residual after current iteration. [VARPRO]
RNORM	Norm of residual. [VARPRO, DPA, POSTPR]
RSD	Residual standard error. [STAT]
RSS	Residual sum of squares. [STAT]
R2	Coefficient of determination. [STAT]
SAV(100)	Vector used to store independent variable values. [ORPLOT]
SAVE	Value of A(1,NCON), i.e. first item in last "constant" column of A. [DPA]
SAVE1	Variable used in moving values from one location to another. [POSTPR]
SCALE	Dependent variable scaling parameter. [PLOT]
SIGMA2	Estimated variance, i.e. (norm of residual)^2/ degrees of freedom. [COV]

TABLE 3.2 (Continued)

Variable Name	Description
SKIP	Flag to indicate whether values should be printed at a given iteration. [VARPRO]
SR(100)	Vector of standardized residuals. [STAT]
SRV(100)	Vector of standardized residuals arranged in ascending order. [STAT]
STP	Interval step size between independent variables in plotted output. [MAIN, STAT, INTRVL, SETUP]
SUM	Variable used in computing a sum. [COV, STAT]
SYH(100)	Vector of probabilities of standardized residuals. [STAT]
T(100,6)	Array of independent variables. T(I,J) contains the value of the I-th observation of the J-th independent variable. [MAIN, CH, PR, VARPRO, DPA, ADA, NUMDER, ORPLOT, INTRVL, SETUP]
TITLE(35)	Title given to problem. [MAIN]
TR	T-ratios. [STAT]
U(7)	Vector containing information on Householder reflections; linear parameters. [DPA, POSTPR]
U1	Factor used in computing Householder reflections. [ORFAC1, ORFAC2]
VAR	Data variable containing "VAR ". [MAIN, PR, STAT]
W(100)	Vector of nonnegative weights. [MAIN, CH, PR, VARPRO, DPA, POSTPR, STAT]
X(200)	Vector of independent variable values to be plotted. [SETUP, PLOT]
XX(100,19)	Matrix product of BM and A(NPKG,NPKG) used in computing standardized residuals. [STAT]
Y(100)	Vector of observed values of dependent variable. [MAIN, CH, PR, VARPRO, DPA, STAT, SETUP]

TABLE 3.2 (Continued)

Variable Name	Description
YB	Mean of the observed values Y. [STAT]
YC	Value of PH(J) using current values of ALF in computing numerical derivatives. [NUMDER]
YD	Value of PH(J) using incremented values of ALF in computing numerical derivatives. [NUMDER]
YG(200)	Vector of first dependent variable values to be plotted. [SETUP, PLOT]
YH(200)	Vector of second dependent variable values to be plotted. [SETUP, PLOT]
YP1	YG(I). [PLOT]
YP2	YH(I). [PLOT]
Z(1)	Solution vector from solving system of equations. [BACSUB]

Coding Restrictions

Before attempting to use the program package one should make note of the following coding restrictions. Should they not suit the user's needs, a detailed account of how they may be changed is provided in Appendix C.

(1) Number of characters in title: 0 to 35.

(2) Number of characters in file name (for saving data): 1 to 5.

(3) Number of observations: 1 to 100.

(4) Number of independent variables per observation: 1 to 6.

(5) Number of linear parameters: 0 to 7.

(6) Number of nonlinear parameters: 0 to 12.

(7) Number of nonzero partial derivatives: 0 to 15.

For the plotted output,

(8) Maximum number of lines printed: 200.

(9) Maximum number of characters per line: 56.

For the convergence criteria,

(10) Maximum number of iterations (function and derivative evaluations): 50.

(11) Absolute tolerance for the norm of the projection of the residual onto the range of the Jacobian of the variable projection functional: .000001.

(12) Relative and absolute tolerance for the change in the norm of the residual: .000001.

(13) Maximum value allowed for the Marquardt parameter: 100.

Storage and Time Considerations

The main program and built-in subroutines are stored in six files. (1) File INLSPV contains the main program and subroutines CH and PR,

occupying 37 blocks of storage. (2) File VAR contains VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR and VARERR and occupies 59 blocks. (3) File ADA has subroutines ADA and NUMDER and occupies 4 blocks. (4) File PLOT contains subroutines ORPLOT, INTRVL, SETUP and PLOT, occupying a total of 6 storage blocks. (5) File STAT, containing subroutine STAT, occupies 8 blocks and (6) file ARY, containing subroutine ARRAYD, occupies 4 blocks. Thus the supplied program package requires a total of 117 blocks of storage. If none of the coding restrictions are to be changed, one might consider storing only the compiled form of the program and save the cost of recompilation. However, the compiled form will require 192 storage blocks. If the user-supplied subroutines CALC, DERIV and WT are place in one file, say CALC1, they should occupy only one additional storage block.

To run the program, one would give the command

.EX INLSPV,VAR,ADA,PLOT,STAT,ARY,CALC1.

If the command is issued more than once, the temporary file 0##.SVC will be added to one's directory.

If on ending execution of the program one decides to save the initialized data and store it in a file named SAVE1, the file SAVE1.DAT will be added to one's directory. Thus one could edit the file SAVE1.DAT using a text-editting language (e.g. TECO, SOS) to change the values of any initialized data instead of changing them during execution of the program. This is especially useful for adding or deleting data points. The size of the file will usually vary between 2 and 20 storage blocks.

If one requests line printer output during execution of the program a file Q##.LPT will be created containing the information to be printed.

When one logs off the terminal, this file will be queued for the line printer and deleted from one's directory. The print-out will be entitled FORLPT.

Execution time will vary depending on the problem being solved and the options selected while running the program. The choice of line printer output, for example, will significantly increase the CPU time required. To give one a rough idea of the time requirements, five problems were executed both with and without requiring line printer output. For each one, all of the plotting and statistical options were chosen except the plot of the standardized residual versus the predicted values.

Problem (1): Exponential Fitting

$$\eta = a_1 + a_2 e^{-\alpha_1 t} + a_3 e^{-\alpha_2 t}$$

where the number of linear parameters n = 3, the number of nonlinear parameters k = 2 and the number of observations m = 33. This example and test data are taken from Osborne [18]. Using the initial nonlinear parameter values provided, the problem took 8 iterations.

Problem (2): Exponential Fitting

$$\eta = \alpha_1 + \alpha_2 e^{-\alpha_4 t} + \alpha_3 e^{-\alpha_5 t}$$

where n = 0, k = 5 and m = 33. This is the same as Problem (1) except all the parameters are treated as nonlinear. Using the initial nonlinear parameter values provided, the problem took 27 iterations.

Problem (3): Fitting Gaussians Plus an Exponential Background

$$\eta = a_1 e^{-\alpha_1 t} + a_2 e^{-\alpha_2 (t-\alpha_5)^2} + a_3 e^{-\alpha_3 (t-\alpha_6)^2} + a_4 e^{-\alpha_4 (t-\alpha_7)^2}$$

where n = 4, k = 7 and m = 80. Data was generated using the model equation and rounding values to three decimal places. With the values

used in the generating equation as initial values for the nonlinear parameters, the problem required 8 iterations to converge.

Problem (4): Thermistor Problem

$$\eta = a_1 e^{\alpha_1/(t+\alpha_2)}$$

where n = 1, k = 2 and m = 16. This example and test data are from Meyer [16]. Using the provided initial values for the nonlinear parameters, the problem took 13 iterations.

Problem (5): Linear Example

$$\eta = a_1 + a_2 t$$

where n = 2, k = 0 and m = 5. The test data are taken from reference [12]. Since the problem is linear, no iterations were required.

Table 3.3 summarizes the results of the times required for the five sample problems.

TABLE 3.3

TIMES REQUIRED FOR SAMPLE PROBLEMS

Problem	Without line printer output	With line printer output
(1)	3.50 CPU sec.	4.13 CPU sec.
(2)	6.06 CPU sec.	6.94 CPU sec.
(3)	13.17 CPU sec.	14.78 CPU sec.
(4)	2.03 CPU sec.	2.43 CPU sec.
(5)	.68 CPU sec.	.85 CPU sec.

CHAPTER IV

USING THE PROGRAM

A Sample Problem

Several tasks must be completed before the user begins a session at the terminal. First, he must determine a model equation to be used and write it in separable form. That is, the model should be of the form

$$\eta = \sum_{i=1}^{n} a_i \phi_i(\alpha;t) + \phi_{n+1}(\alpha;t). \tag{4.1}$$

Consider as an example the thermistor problem described by R. R. Meyer [16] where for unknown parameters x_1, x_2 and x_3 the model equation is

$$\eta = x_1 e^{x_2/(t+x_3)}$$

for independent variable t. Instead of treating all the unknown parameters the same we will separate them into those that appear linearly (x_1) and those that appear in a nonlinear fashion (x_2 and x_3). Thus the model could be rewritten as

$$\eta = a_1 e^{\alpha_1/(t+\alpha_2)}$$

where the a_i are the linear parameters and the α_i are the nonlinear parameters. One can then easily determine the functions $\phi_i(\alpha;t)$. In this example, $\phi_1 = e^{\alpha_1/(t+\alpha_2)}$ and $\phi_2 = 0$. Note that the number of functions is n+1, or one more than the number of linear parameters. This allows one to have a function which depends entirely on the nonlinear parameters.

Writing the Subroutines CALC, DERIV and WT

Once the model has been determined and written in separable form

the user will need to write a subroutine which describes the model. The first four lines of CALC should be as in the example (Figure 4.1). Here ALF is the vector of nonlinear parameters, C is the vector of linear parameters and T is the array of independent variables. T(K,J) is the value of the J-th independent variable at the K-th observation. (If there is only one independent variable for each observation, one could change the dimensioning T(MMAX,IV) to T(MMAX) and refer to T(K,1) more simply as T(K).)

The separated functions PH(I) (i.e. ϕ_i) which fit into the model (4.1) should be next. All of the PH(I) which do not depend on any of the nonlinear parameters (ALF) should appear first. To eliminate unnecessary computation one should insert the statements

```
COMMON /NCBLK/ NC
LOGICAL NC
IF (NC) GOTO 5
```

after the first four lines and include the label 5 on the first PH(I) which does depend on ALF. The ϕ_{n+1} term will be assumed to be zero unless an expression for it appears in the coding; thus PH(2) is not needed in this example. The separated functions PH(I) should be followed by their linear combination ETA (η) and a RETURN and END.

Subroutine DERIV should be used when all or some of the partial derivatives of the PH(I) are to be calculated from the closed form. (If all of the derivatives are to be calculated numerically, a dummy subroutine must be provided.) The first four lines of DERIV should be as in the example and T may be singly-dimensioned when the number of independent variables per observation is one. DERIV should include the expressions for DPH(I,J) which represent the value of the partial

```
      SUBROUTINE CALC (K, KG, N, MMAX, IV, C, ALF, T, ETA)
      DOUBLE PRECISION ALF(KG), T(MMAX,IV), C(N), ETA
      COMMON /BLKPH/ PH
      DOUBLE PRECISION PH(8)
      PH(1) = DEXP(ALF(1)/(T(K,1)+ALF(2)))
      ETA = C(1)*PH(1)
      RETURN
      END


      SUBROUTINE DERIV (K, T, MMAX, IV, ALF, KG, NP1)
      DOUBLE PRECISION ALF(KG), T(MMAX,IV)
      COMMON /BLKDPH/ DPH
      DOUBLE PRECISION DPH(8,12)
      DOUBLE PRECISION T1, T2
      DO 10 J = 1, KG
         DO 10 I = 1, NP1
10          DPH(I,J) = 0.
      T2 = T(K,1)+ALF(2)
      T1 = DEXP(ALF(1)/T2)
      DPH(1,1) = T1/T2
      DPH(1,2) = -T1*ALF(1)/T2**2
      RETURN
      END


      SUBROUTINE WT (M, W, Y)
      DOUBLE PRECISION Y(M), W(M)
      DO 20 I = 1, M
20    W(I) = 1./Y(I)
      RETURN
      END
```

Figure 4.1

Example of Subroutines CALC, DERIV and WT

derivative of the I-th function PH with respect to the J-th nonlinear parameter. For example, DPH(1,2) is the derivative of PH(1) with respect to ALF(2) (evaluated at the K-th observation).

Subroutine WT can be used to calculate the weights, e.g. based on the observed values of the dependent variable. Here Y represents the vector of observed values and W represents the vector of weights. (If WT is not used, a dummy subroutine must be provided.) The user will be given the option of calculating the weights in WT during execution of the program.

One should note that all of the routines are written in double precision arithmetic. Therefore, one must be sure that any additional variables used in subroutines CALC, DERIV or WT are specified on a DOUBLE PRECISION statement. (For example, see variables T1 and T2 in Figure 4.1).

Execution of the Program

In order to execute the program, the user will first need to have the files INLSPV, VAR, ADA, PLOT, STAT and ARY copied into his work area. Once this has been done (and assuming the three subroutines CALC, DERIV and WT are all stored in one file named CALC7) one would need to issue the command

```
.EX INLSPV,VAR,ADA,PLOT,STAT,ARY,CALC7
```

to execute the program.

During execution the user will need to supply the answers to a series of questions printed out by the routine. All yes or no questions should be answered Y for yes and any other character (or no character at all) for no. Each response must be followed by a carriage return. If two or more items of information are to be entered on the same line, they

should be separated by either blanks or commas.

The following section headings correspond to the questions which will be asked. The reader may find it helpful to consider the sample terminal session in Figure 4.2 while reading this section. (Here, file CALC7 contains the three subroutines listed in Figure 4.1.) In Figure 4.2 all user responses have been underlined.

(1) ENTER TITLE: The user is asked to supply a title for the problem (up to 35 characters long). The title will be printed later as a heading for each run of the least squares program.

(2) DO YOU WANT LINE PRINTER OUTPUT? If the answer is Y, the program gives the user his output both at the terminal and at the line printer in the computer room. The latter provides a hard copy for future reference.

(3) ARE YOUR INITIAL VALUES ALREADY ON FILE? At the end of each session the user has the option of saving the values of all initialized data. If the data has been saved, the user can enter Y and will then be asked for the name of the data file in which it was stored. The initial information will then be printed out and control will shift to the command point (step (16)).

(4) NO. OF DATA POINTS: The number of observations (maximum 100) should be entered.

(5) NO. OF LINEAR PARAMETERS: Enter the number of linear parameters (maximum 7).

(6) NO. OF NONLINEAR PARAMETERS: Enter the number of nonlinear parameters (maximum 12).

(7) NO. OF INDEPENDENT VARIABLES: Enter the number of independent variables per observation (maximum 6).

```
.EX INLSPV,VAR,ADA,PLOT,STAT,ARY,CALC7
LINK: Loading
[LNKXCT INLSPV Execution]

ENTER TITLE:  SAMPLE PROBLEM
DO YOU WANT LINE PRINTER OUTPUT? (TYPE "Y" FOR YES)  _
ARE YOUR INITIAL VALUES ALREADY ON FILE?  _
NO. OF DATA POINTS:  15
NO. OF LINEAR PARAMETERS:  1
NO. OF NONLINEAR PARAMETERS:  2
NO. OF INDEPENDENT VARIABLES:  1
ARE WEIGHTS TO BE ENTERED WITH DATA?
ARE WEIGHTS TO BE CALCULATED IN SUBROUTINE WT?  _
WEIGHTS ARE SET TO 1

DATA POINTS:  LIST INDEPENDENT VARIABLES AND
DEPENDENT VARIABLE, SEPARATED BY BLANKS
 1  50 34780
 2  55 28610
 3  60 23650
 4  65 19630
 5  70 16370
 6  75 13720
 7  80 11540
 8  85 9744
 9  90 8261
10  95 7030
11  100 6005
12  105 5147
13  110 4427
14  115 3820
15  120 3307
INITIAL VALUES OF NONLINEAR PARAMETERS
 1  40000
 2  2500
WHICH TERMS OF THE MODEL DEPEND ON NONLINEAR PARAMETER  1:  1
WHICH TERMS OF THE MODEL DEPEND ON NONLINEAR PARAMETER  2:  1
ALL DERIVATIVES IN CLOSED FORM?  Y

NEXT STEP? (TYPE "H" FOR HELP)  CM
NO. OF DATA POINTS:  16

NEXT STEP? (TYPE "H" FOR HELP)  CT
WHICH DATA POINT IS TO BE CHANGED?
(TYPE ITS INDEX NO. AS IT APPEARS IN THE LIST)  16
ENTER NEW INDEP & DEP VARIABLES FOR DATA POINT 16:
125 2872
DO YOU WANT TO CHANGE ANY OTHERS?  _
```

Figure 4.2

A Sample Terminal Session

```
NEXT STEP? (TYPE "H" FOR HELP)  CW
WHICH WEIGHT IS TO BE CHANGED?
(TYPE ITS INDEX NO. AS IT APPEARS IN THE LIST)  16
ENTER NEW WEIGHT FOR DATA POINT 16:  1
DO YOU WANT TO CHANGE ANY OTHERS?  _

NEXT STEP? (TYPE "H" FOR HELP)  PTW
ALL OF THE DATA POINTS?  Y
INDEX      WEIGHT          DEP VAR          IND VAR  1
 1      0.1000000E+01   0.3478000E+05   0.5000000E+02
 2      0.1000000E+01   0.2861000E+05   0.5500000E+02
 3      0.1000000E+01   0.2365000E+05   0.6000000E+02
 4      0.1000000E+01   0.1963000E+05   0.6500000E+02
 5      0.1000000E+01   0.1637000E+05   0.7000000E+02
 6      0.1000000E+01   0.1372000E+05   0.7500000E+02
 7      0.1000000E+01   0.1154000E+05   0.8000000E+02
 8      0.1000000E+01   0.9744000E+04   0.8500000E+02
 9      0.1000000E+01   0.8261000E+04   0.9000000E+02
10      0.1000000E+01   0.7030000E+04   0.9500000E+02
11      0.1000000E+01   0.6005000E+04   0.1000000E+03
12      0.1000000E+01   0.5147000E+04   0.1050000E+03
13      0.1000000E+01   0.4427000E+04   0.1100000E+03
14      0.1000000E+01   0.3820000E+04   0.1150000E+03
15      0.1000000E+01   0.3307000E+04   0.1200000E+03
16      0.1000000E+01   0.2872000E+04   0.1250000E+03

NEXT STEP? (TYPE "H" FOR HELP)  RV

OUTPUT PRINTING INDEX
(SEE DOCUMENTATION.  "1" IS RECOMMENDED):  1
```

SAMPLE PROBLEM

```
    0  NORM OF RESIDUAL =    0.3093451E+05
    NU =  0.1000000E+01
%FRSAPR Floating overflow          PC= 75775

%FRSAPR Floating overflow          PC= 75775

  CATASTROPHIC FAILURE -- COLUMN    1 IS ZERO.  SEE DOCUMENTATION.
```

Figure 4.2 (Continued)

```
NEXT STEP? (TYPE "H" FOR HELP)  CALF
WHICH NONLINEAR PARAMETER IS TO BE CHANGED?
(TYPE ITS INDEX NO. AS IT APPEARS IN THE LIST)  1
ENTER NEW VALUE FOR PARAMETER  1:  4000
DO YOU WANT TO CHANGE ANY OTHERS?  Y
WHICH NONLINEAR PARAMETER IS TO BE CHANGED?
(TYPE ITS INDEX NO. AS IT APPEARS IN THE LIST)  2
ENTER NEW VALUE FOR PARAMETER  2:  250
DO YOU WANT TO CHANGE ANY OTHERS?  _

NEXT STEP? (TYPE "H" FOR HELP)  RV

OUTPUT PRINTING INDEX
(SEE DOCUMENTATION.  "1" IS RECOMMENDED):  2


                    SAMPLE PROBLEM


  0  NORM OF RESIDUAL =  0.2641466E+04
  NU =  0.1000000E+01

ITERATION   2    NONLINEAR PARAMETERS

0.3841692E+04  0.2566715E+03

  1  NORM OF RESIDUAL =  0.3819947E+03
  NU =  0.2500000E+00
  NORM(DELTA-ALF) / NORM(ALF) =   0.120E-01

ITERATION   4    NONLINEAR PARAMETERS

0.3887998E+04  0.2591363E+03

  1  NORM OF RESIDUAL =  0.3591494E+03
  NU =  0.6250000E-01
  NORM(DELTA-ALF) / NORM(ALF) =   0.106E-01

ITERATION   6    NONLINEAR PARAMETERS

o.4482350E+04  0.2839304E+03

  1  NORM OF RESIDUAL =  0.2531463
  NU =  0.1562500E-01
  NORM(DELTA-ALF) / NORM(ALF) =   0.988E-01
```

Figure 4.2 (Continued)

```
ITERATION   8    NONLINEAR PARAMETERS

0.5946845E+04  0.3380683E+03

  1  NORM OF RESIDUAL =  0.1238100E+03
  NU =  0.3906250E-02
  NORM(DELTA-ALF) / NORM(ALF) =   0.113E+00

ITERATION  10    NONLINEAR PARAMETERS

0.6181032E+04  0.3452136E+03

  1  NORM OF RESIDUAL =  0.9378404E+01
  NU =  0.9765625E-03
  NORM(DELTA-ALF) / NORM(ALF) =   0.303E-02

ITERATION  12    NONLINEAR PARAMETERS

0.6181346E+04  0.3452236E+03

  1  NORM OF RESIDUAL =  0.9377945E+01
  NU =  0.2441406E-03
  NORM(DELTA-ALF) / NORM(ALF) =   0.184E-06

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

 LINEAR PARAMETERS

0.5609636E-02

 NONLINEAR PARAMETERS

0.6181346E+04  0.3452236E+03

 NORM OF RESIDUAL =  0.9377945E+01

 ESTIMATED VARIANCE =  0.6765066E+01

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
INDICATE WHICH OF THE FOLLOWING SHOULD BE PRINTED
BY TYPING "Y":
  TABLE OF OBSERVED & PREDICTED VALUES, WTD RESIDUAL:  Y
  COVARIANCE MATRIX:  Y
  CORRELATION MATRIX:  Y
```

Figure 4.2 (Continued)

COVARIANCE MATRIX

ROW	COLUMN 1	COLUMN 2	COLUMN 3
1	2.461100D-08	-3.655635D-03	-1.229928D-04
2	-3.655635D-03	5.433105D+02	1.828957D+01
3	-1.229928D-04	1.828957D+01	6.160068D-01

INDEX	IND VAR 1	OBS VAL	PRED VAL	WTD RESID	WEIGHTS
1	5.00000E+01	3.47800E+04	3.47820E+04	-1.95030E+00	1.00000E+00
2	5.50000E+01	2.86100E+04	2.86086E+04	1.43114E+00	1.00000E+00
3	6.00000E+01	2.36500E+04	2.36446E+04	5.37484E+00	1.00000E+00
4	6.50000E+01	1.96300E+04	1.96330E+04	-2.98164E+00	1.00000E+00
5	7.00000E+01	1.63700E+04	1.63751E+04	-5.12902E+00	1.00000E+00
6	7.50000E+01	1.37200E+04	1.37170E+04	3.02242E+00	1.00000E+00
7	8.00000E+01	1.15400E+04	1.15383E+04	1.71649E+00	1.00000E+00
8	8.50000E+01	9.74400E+03	9.74473E+03	-7.33962E-01	1.00000E+00
9	9.00000E+01	8.26100E+03	8.26199E+03	-9.88692E-01	1.00000E+00
10	9.50000E+01	7.03000E+03	7.03117E+03	-1.16988E+00	1.00000E+00
11	1.00000E+02	6.00500E+03	6.00543E+03	-4.30013E-01	1.00000E+00
12	1.05000E+02	5.14700E+03	5.14733E+03	-3.26635E-01	1.00000E+00
13	1.10000E+02	4.42700E+03	4.42680E+03	1.95691E-01	1.00000E+00
14	1.15000E+02	3.82000E+03	3.81964E+03	3.64209E-01	1.00000E+00
15	1.20000E+02	3.30700E+03	3.30621E+03	7.87706E-01	1.00000E+00
16	1.25000E+02	2.87200E+03	2.87060E+03	1.39961E+00	1.00000E+00

CORRELATION MATRIX

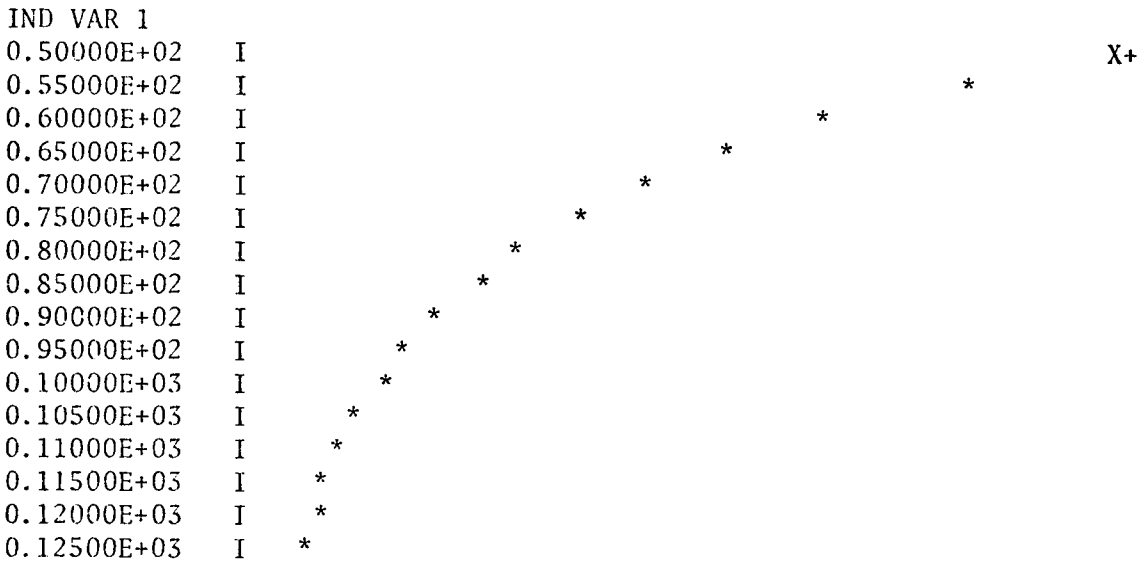
ROW	COLUMN 1	COLUMN 2	COLUMN 3
1	1.000000D+00	-9.997102D-01	-9.989011D-01
2	-9.997102D-01	1.000000D+00	9.997393D-01
3	-9.989011D-01	9.997393D-01	1.000000D+00

```
PLOT ROUTINE CURRENTLY SET NOT TO DO ANY PLOTS.
DO YOU WANT TO START PLOTTING? Y

IF YOU WANT TO PLOT RESIDUALS, TYPE "1"
IF YOU WANT TO PLOT OBSERVED & PREDICTED VALUES, TYPE "2": 2
```

Figure 4.2 (Continued)

PLOT OF OBSERVED (X) & PREDICTED (+) VALUES
(* INDICATES OBSERVED & PREDICTED VALUES THE SAME)

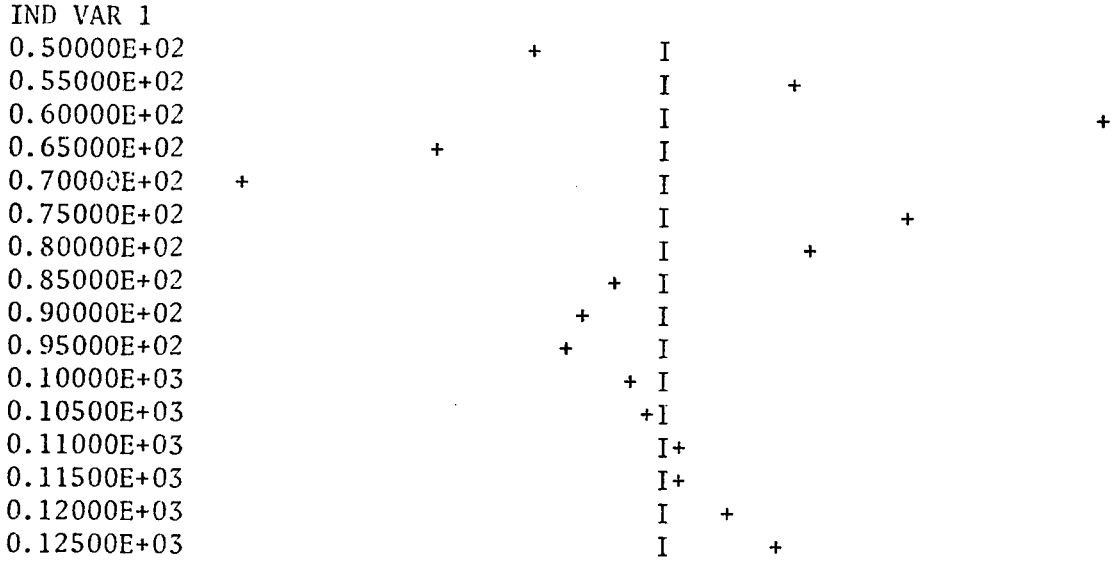


DO YOU WANT TO CONTINUE PLOTTING? Y

IF YOU WANT TO PLOT RESIDUALS, TYPE "1"
IF YOU WANT TO PLOT OBSERVED & PREDICTED VALUES, TYPE "2": 1

Figure 4.2 (Continued)

PLOT OF RESIDUALS




```
DO YOU WANT TO CONTINUE PLOTTING? _

NEXT STEP? (TYPE "H" FOR HELP)  RS
INDICATE WHICH OF THE FOLLOWING SHOULD BE PRINTED:
  PARAMETERS, THEIR STD DEV & T-RATIO:  Y
  RES SUM OF SQS, RES MEAN SQ, RES STD DEV &
      COEFF OF DETERMINATION:  Y
  TABLE OF OBS, PRED VAL, RESID & STD RESID:  Y
  PLOT OF STD RES VS INDEP VAR:  Y
  PLOT OF STD RES VS PRED VAL:
  PROBABILITY PLOT OF STD RES:  Y

PARAMETER          S.D. OF PARM.       T-RATIO

 0.5609636E-02   0.1568789E-03   0.3575774E+02
 0.6181346E+04   0.2330902E+02   0.2651912E+03
 0.3452236E+03   0.7848610E+00   0.4398532E+03


WTD RESIDUAL SUM OF SQUARES:     0.8794586E+02
WTD RESIDUAL MEAN SQUARE:     0.6765066E+01
WTD RESIDUAL STANDARD ERROR:     0.2600974E+01
```

Figure 4.2 (Continued)

COEFFICIENT OF DETERMINATION (R-SQUARE): 0.9999999E+00

INDEX	IND VAR 1	OBS VAL	PRED VAL	RESID	STD RES
1	5.00000E+01	3.47800E+04	3.47820E+04	-1.95030E+00	-7.17783E-04
2	5.50000E+01	2.86100E+04	2.86086E+04	1.43114E+00	1.78966E-03
3	6.00000E+01	2.36500E+04	2.36446E+04	5.37484E+00	8.13929E-03
4	6.50000E+01	1.96300E+04	1.96330E+04	-2.98164E+00	-5.44056E-03
5	7.00000E+01	1.63700E+04	1.63751E+04	-5.12902E+00	-1.12230E-02
6	7.50000E+01	1.37200E+04	1.37170E+04	3.02242E+00	7.89432E-03
7	8.00000E+01	1.15400E+04	1.15383E+04	1.71649E+00	5.32794E-03
8	8.50000E+01	9.74400E+03	9.74473E+03	-7.33962E-01	-2.69585E-03
9	9.00000E+01	8.26100E+03	8.26199E+03	-9.88692E-01	-4.27958E-03
10	9.50000E+01	7.03000E+03	7.03117E+03	-1.16998E+00	-5.94401E-03
11	1.00000E+02	6.00500E+03	6.00543E+03	-4.30013E-01	-2.55481E-03
12	1.05000E+02	5.14700E+03	5.14733E+03	-3.26635E-01	-2.26089E-03
13	1.10000E+02	4.42700E+03	4.42680E+03	1.95691E-01	1.57250E-03
14	1.15000E+02	3.82000E+03	3.81964E+03	3.64209E-01	3.38599E-03
15	1.20000E+02	3.30700E+03	3.30621E+03	7.87706E-01	8.44471E-03
16	1.25000E+02	2.87200E+03	2.87060E+03	1.39961E+00	1.72477E-02

PLOT OF STANDARDIZED RESIDUALS VS INDEPENDENT VARIABLE 1

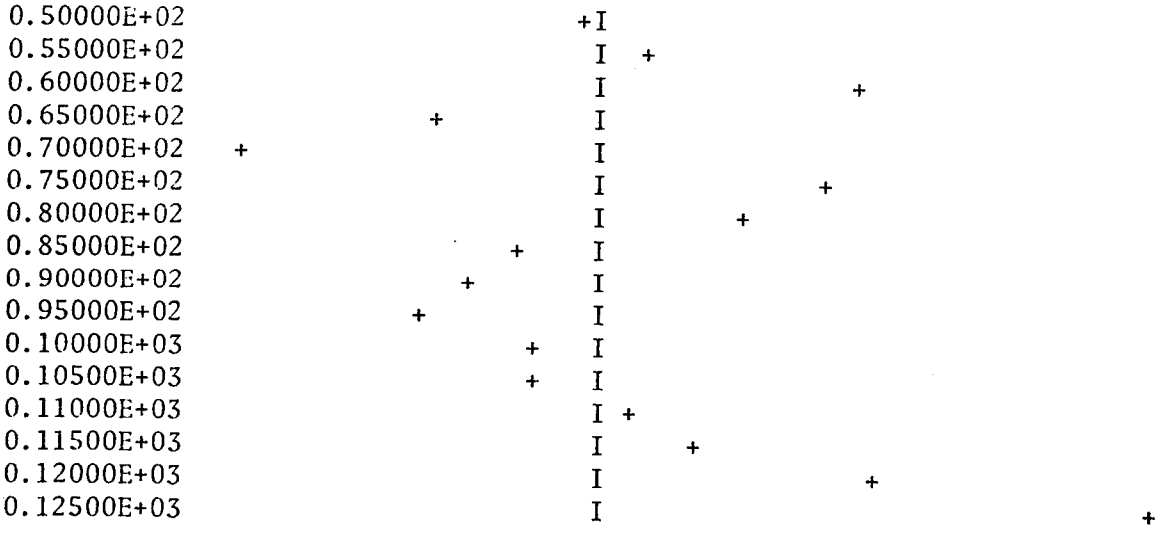
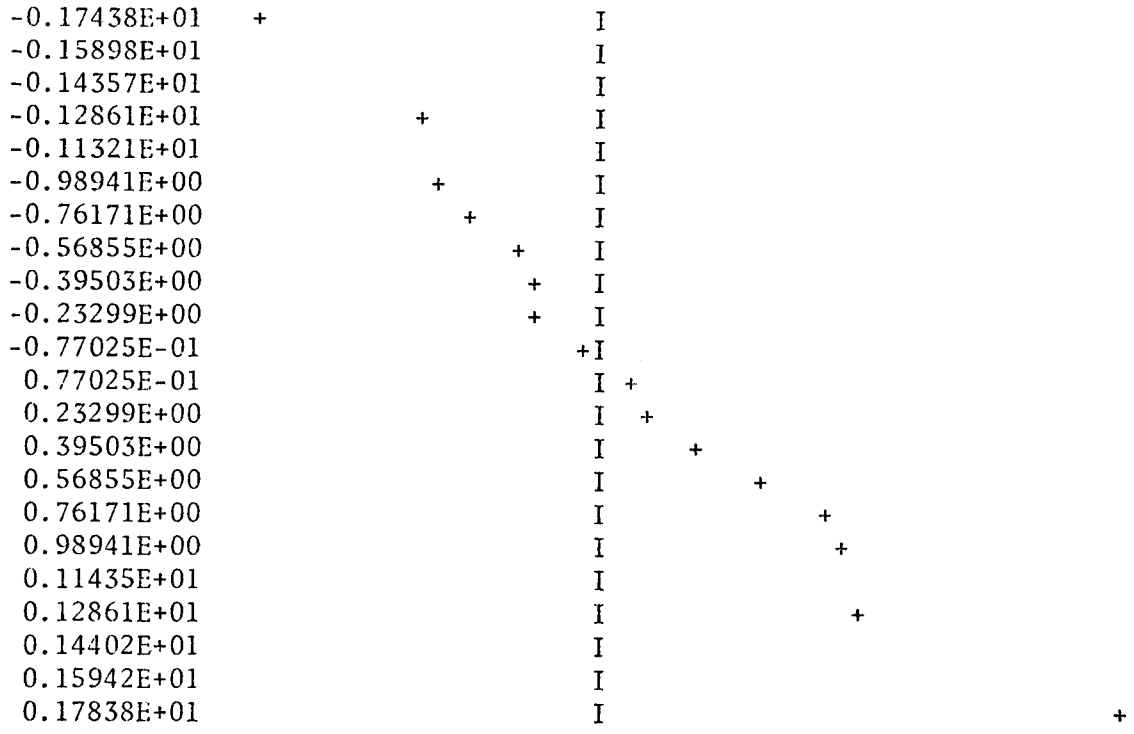


Figure 4.2 (Continued)

PROBABILITY PLOT OF STANDARDIZED RESIDUALS




```
NEXT STEP? (TYPE "H" FOR HELP)  SO
STOP

END OF EXECUTION
CPU TIME: 2.12  ELAPSED TIME: 13.20.28
EXIT

.
```

Figure 4.2 (Continued)

(8) ARE WEIGHTS TO BE ENTERED WITH DATA? If the answer is Y, continue with step (10).

(9) ARE WEIGHTS TO BE CALCULATED IN SUBROUTINE WT? If the answer is Y, the weights will be calculated as specified in the subroutine. If not, all weights will be set to one.

(10) DATA POINTS: LIST INDEPENDENT VARIABLES, DEPENDENT VARIABLE (& WEIGHTS) SEPARATED BY BLANKS. For each point, an index number will be printed. The user should then supply the values of the independent variables, the dependent variable, and if the answer to question (8) was Y, the weight. The values can be entered either with or without decimal points and with or without exponents (e.g. 125, 125.1 or 1.25E+02).

(11) INITIAL VALUES OF NONLINEAR PARAMETERS: An index number will be printed for each nonlinear parameter. The user should then enter the initial value for that parameter.

(12) WHICH TERMS OF THE MODEL DEPEND ON NONLINEAR PARAMETER #: Question (12) will be asked for each nonlinear parameter. If more than one term of the model equation depends on a particular nonlinear parameter the term numbers should be entered on the same line separated by blanks or commas. For example, for the model $\eta = a_1 + a_2 e^{-\alpha_1 t} + a_3 e^{(\alpha_1+t)/\alpha_2}$, one would enter "2 3" in response to the first question and "3" in response to the second since terms two and three depend on the first nonlinear parameter and term three depends on the second.

(13) ALL DERIVATIVES IN CLOSED FORM? If the closed form has been specified in subroutine DERIV for all derivatives $d\phi_i/d\alpha_j$ which are not zero (i.e. one for each of the numbers entered in (12)) one should answer Y and continue with step (16).

(14) ALL NUMERICAL DERIVATIVE INCREMENTS THE SAME? If the user

wishes to have the derivatives calculated numerically he will need to supply the increment values for the forward difference quotient. If the increment values are to be the same for each derivative he should answer Y. He will then be asked to enter the increment value and continue with step (16).

(15) NUMERICAL DERIVATIVE INCREMENTS: An index number will be printed for each increment value. The order of the increments will be by nonlinear parameter and then by term of the model. Thus for the example given in step (12) the increments would correspond in order to DPH(2,1), DPH(3,1) and DPH(3,2). If any of these is to be calculated from the closed form specified in DERIV, its increment value should be set to zero.

(16) NEXT STEP? (TYPE "H" FOR HELP) This is the command point described in Chapter III. Here the user has a choice as to what should be done next. To specify the next step the user will need to enter the appropriate command code as listed in Table 3.1 (page 20). (Should the table not be readily accessible he can enter H to obtain a listing of all the command codes.) At this point the user has the option of correcting any values that have been entered incorrectly. This is done in the sample terminal session in Figure 4.2 (page 48). The data points were miscounted; there should have been 16. Thus the command CM is used to change the number of data points. Then the command CT is used to change the values of the sixteenth data point from zero to 125 and 2872. When a data point is added, one will also need to enter its weight. The command CW is used to accomplish this. To insure that all the data points are now entered correctly, the command PTW is issued.

Once all initial values have been entered correctly one can run the least squares program by entering the command RV. Two examples of the

possible results from this command are given in the sample problem in Figure 4.2 (pages 49 and 50). The first execution of the RV command leads to the error message "CATASTROPHIC FAILURE -- COLUMN 1 IS ZERO. SEE DOCUMENTATION." and a return to the command point. If the error had been less severe (e.g. the problem terminated for ill-conditioning or excessive iterations) the program would have continued and not returned to the command point.

Convergence failures may occur as a result of incorrect coding in subroutines CALC or DERIV. If the subroutines are coded correctly and error messages of ill-conditioning or catastrophic failure persistently occur, a variety of factors may be responsible. One is a poor scaling of the parameters (i.e. improper choice of weights). Another is simply a poor choice of initial values for the nonlinear parameters. Or, the problem might be a poor choice of model equation.

To correct the error in the sample problem it was necessary to change the initial values of the nonlinear parameters. Thus the command code CALF was entered. It was decided to change the first nonlinear parameter to 4000 and the second to 250.

The second execution of the RV command in the sample terminal session demonstrates the normal flow of the program. First the user is asked to supply an output printing index. If the index is positive the nonlinear parameters, norm of the residual and Marquardt parameter will be printed every index-th iteration (as well as at the first and final iterations). The linear parameters will be printed at the final iteration along with the norm of the residual and estimated variance. Any error messages will also be printed. If the index is zero, only the final quantities will be printed along with any error messages.

Once the optimal parameters have been calculated the user will be given a choice as to what further information should be printed. The options include the covariance matrix, the correlation matrix, and a table of the independent variables, observed and predicted values of the dependent variable, the weighted residual and weights for each data point. The user will also be given the opportunity to plot the residuals and the observed and predicted values at each data point.

To obtain further statistical information one should enter the command RS. The statistics available will then include the parameters, their standard deviation and t-ratios; the weighted residual sum of squares, weighted residual mean square, weighted residual standard error and coefficient of determination; a table of the independent variables, the observed and predicted values of the dependent variables, the residuals, and the standardized residuals for each data point; plots of the standardized residuals versus the independent variables or versus the predicted values of the dependent variable; and a probability plot of the standardized residuals. Again the user is asked to indicate which of these should be printed. For a further detailed description of the statistics, the reader should refer to the following section.

Finally, to end execution of the program the user will need to enter either SD or S0. If the user enters SD he will be asked to supply the name of a file (up to five characters) for storing the initialized data. He will then have the option of answering Y to question (3) the next time he uses the program package. Entering S0 (S-zero) will lead to termination of the program without saving the initialized data.

Statistical Considerations

A variety of statistics are available after the optimal parameter values have been calculated. These will be discussed in the order that they appear in the program. One should bear in mind that for nonlinear least squares many statistics can only be estimated, and their values should be treated accordingly.

The first statistical option is a table of the independent variables, observed and predicted values of the dependent variable, weighted residual and weights. The weighted residual is the result of multiplying the residual by the appropriate weighting factor, i.e.

$$w_i(y_i - \eta_i) = w_i r_i.$$

The variance-covariance matrix provides an estimate of the variance (or covariance) of the calculated parameters. Thus the diagonal elements are the variances of the parameters (i.e. the average of the squared deviations). The off-diagonal element x-y is the covariance of the parameters x and y, defined as the average cross-product of the x and y deviations from their means $\bar{x}$ and $\bar{y}$ [8]. The covariance matrix is calculated, as described by Lawson and Hanson [15], from the formula

$$V = \sigma^2 T^{-1} T^{-T}$$

where σ^2 is a scalar factor such that

$$\sigma^2 = \frac{(\text{norm of residual})^2}{\text{degrees of freedom}}.$$

(The degrees of freedom is the number of observations minus the number of unknown parameters.) The matrix T is the final reduced form of the matrix of functions ϕ and $\frac{d\phi}{d\alpha}$ used in calculating the optimal parameters.

The correlation matrix is a scaled version of the covariance matrix. Its x-y element is

$$\frac{\text{cov } (x,y)}{\sigma_x \sigma_y}$$

where σ_i represents the standard deviation of the i-th parameter. Thus the diagonal elements are all ones and the off-diagonal elements are the correlation coefficients and have absolute value between zero and one. The closer the values are to one, the stronger the relationship is between the two parameters. A negative coefficient indicates an inverse relationship, i.e. y decreases as x increases; a positive coefficient indicates that x and y tend to increase (or decrease) together.

Plots of the residuals and of the observed and predicted values versus the independent variable give a pictoral view of the data in the table. One should note, however, that the plot is of the actual residuals, and not of the weighted residuals.

On execution of the RS command several other statistical options are available. The first is a table of the parameters, their standard deviations and the t-ratio (the parameters divided by their standard deviations). The t-ratio follows the Student's t-distribution if the sample population is normal or near-normal [8]. It can be used to determine the level of significance of the hypothesis that the mean of the population takes on a particular value (e.g. the parameter takes on a particular value) or to construct a confidence interval for the mean of a normal population (e.g. the parameter) [5].

The weighted residual sum of squares (RSS) is the quantity which we have attempted to minimize, i.e. the sum of the squares of the weighted residuals. The residual mean square (RMS), sometimes called the estimated variance, is the residual sum of squares divided by the degrees of freedom. The residual standard error is the square root of the residual mean square.

The coefficient of determination (also referred to as R^2 or the multiple correlation coefficient) is another measure used to determine the goodness of the fit. It is calculated from the formula

$$R^2 = 1 - RSS/CTSS$$

where the corrected total sum of squares is found by

$$CTSS = \sum_{i=1}^{m} w_i (y_i - \bar{y})^2$$

where $\bar{y}$ is the mean of the observed values

$$\bar{y} = (\sum_{i=1}^{m} y_i) / m.$$

The coefficient of determination is thus a normalized residual sum of squares and will always have a value between zero and one. The closer the value is to one, the better the fit. For example, a coefficient of .995 would indicate that 99.5% of the total sum of squares is accounted for by the fitted equation [8].

One also has the option of printing a table of values of the independent variables, the observed and predicted values of the dependent variable, the residual and the standardized residual. The standardized residuals are the residuals divided by their standard deviations and are calculated from the formula

$$r_i / (RMS/w_i - s_i^2)^{\frac{1}{2}}$$

where s_i is the standard deviation of the predicted value of the i-th dependent variable [12].

Several plotting features are also available. The standardized residual is used to provide a common scale and to avoid distortion caused by differences in the standard deviations of the residuals.

The first is a plot of the standardized residuals versus the

independent variable(s). If the values do not appear to lie in a vertical band about the axis one of several things may be indicated. If the variance steadily increases (or decreases) over the range of values to form a wedge-shaped graph, the implication is that weighted least squares analysis should have been used. If the values lie in a linear band that is not vertical, a linear term should have been included in the model. If the band of values is curved, linear and quadratic terms should have been included in the model. One should realize that combinations of these defects can also occur [7].

The standardized residuals are also plotted against the predicted values. Here again the analysis is similar to that above. Non-randomness might indicate the need to use weights or the omission of some important term from the model equation [12].

A probability plot of the standardized residuals versus the expected value of the standardized residuals is also provided, in which it is assumed that the measurement errors are normally and independently distributed. If the model adequately represents the data, the points will lie roughly on a straight line [12].

Thus the available statistics should provide the user with sufficient information to determine the adequacy of the model equation used.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Bard, Y., Nonlinear Parameter Estimation, Academic Press, New York (1974).

[2] Ben-Israel, A. and T.N.E. Greville, Generalized Inverses: Theory and Applications, Wiley, New York (1974).

[3] Bolstad, J., "VARPRO," Computer Science Department, Stanford University, Stanford, California (1974).

[4] Bolstad, J. and M. Cordes, "Numerical Solution of Least Squares Problems," Computer Science Department, Stanford University, Stanford, California (1975).

[5] Burington, R.S. and D.C. May, Handbook of Probability and Statistics, McGraw-Hill, New York (1970) 239-240, 243-244.

[6] DECSYSTEM10 Fortran 10 Programmer's Reference Manual, DEC-10-LFORA-C-D, Digital Equipment Corp., Maynard, Mass. (1975).

[7] Draper, N.R. and H. Smith, Applied Regression Analysis, Wiley, New York (1966).

[8] Ehrenberg, A.S.C., Data Reduction, Wiley, London (1975).

[9] Forsythe, G.E. and C.B. Moler, Computer Solution of Linear Algebraic Systems, Prentice-Hall, New Jersey (1967).

[10] Golub, G.H. and V. Pereyra, "The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate," SIAM J. Numer. Anal. 10 (1973) 413-432.

[11] Golub, G.H., "Numerical Methods for Solving Linear Least Squares Problems," Numer. Math. 7 (1965) 206-216.

[12] Hogben, D., S.T. Peavy and R.N. Varner, OMNITAB II User's Reference Manual, NBS Tech. Note 552 (1971) 138-146.

[13] Kaufman, L., "A Variable Projection Method for Solving Separable Nonlinear Least Squares Problems," BIT 15 (1975) 49-57.

[14] Krogh, F.T., "Efficient Implementation of a Variable Projection Algorithm for Nonlinear Least Squares Problems," Comm. ACM 17 (1974) 167-169.

[15] Lawson, C.L. and R.J. Hanson, Solving Least Squares Problems, Prentice-Hall, New Jersey (1974).

[16] Meyer, R.R., "Theoretical and Computational Aspects of Nonlinear Regression," in Rosen, et. al. eds., Nonlinear Programming, Academic Press, New York (1970) 465-486.

[17] Ortega, J.M. and W.C. Rheinbolt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York (1970) 59-66.

[18] Osborne, M.R., "Some Aspects of Non-linear Least Squares Calculations," in Lootsma, ed., Numerical Methods for Non-linear Optimization, Academic Press, London (1972) 171-189.

[19] Ramsin, H. and P.A. Wedin, "A Comparison of Some Algorithms for the Nonlinear Least Squares Problem," LU/CS-75:06, Computer Science Department, Lund University, Lund, Sweden (1975).

[20] Stewart, G.W., Introduction to Matrix Computations, Academic Press, New York (1973).

[21] Van Loan, C., "Lectures in Least Squares," TR76-279, Computer Science Department, Cornell University, Ithaca, New York (1976).

APPENDIXES

APPENDIX A

LISTINGS OF THE FORTRAN PROGRAMS

```
C       THE FOLLOWING PROGRAM AND SET OF SUBROUTINES
C       AN INTERACTIVE APPROACH FOR SOLVING EITHER
C       LINEAR OR NONLINEAR LEAST SQUARES PROBLEMS.
C
C       THE PACKAGE CONSISTS OF A MAIN PROGRAM AND
C       18 SUBROUTINES:  CH, PR, ARRAYD, STAT, PLOT,
C       VARPRO, ORFAC1, ORFAC2, DPA, BACSUB, POSTPR,
C       COV, VARERR, ADA, AND NUMDER.  IN ADDITION
C       THE USER IS ASKED TO SUPPLY THREE SUBROUTINES:
C       CALC, DERIV AND WT.
C
C       THE METHOD USED FOR SOLVING THE LEAST SQUARES
C       PROBLEM IS DESCRIBED IN THE COMMENTS PRECEDING
C       SUBROUTINE VARPRO.  ALSO INCLUDED ARE A
C       DESCRIPTION OF THE PARAMETERS, COMMENTS ON
C       CONVERGENCE FAILURES AND A LIST OF REFERENCES.
C       IN READING THESE COMMENTS, ONE SHOULD BEAR IN
C       MIND THAT THE USER WILL NOT NEED TO PROVIDE
C       THE SUBROUTINE ADA.  THUS THE SECTION ON THE
C       SUBROUTINES REQUIRED MAY BE SKIPPED.
C
C       FOR FURTHER INFORMATION ON THE USER-SUPPLIED
C       SUBROUTINES CALC, DERIV AND WT AS WELL AS
C       A MORE COMPLETE LIST OF THE CODING RESTRICTIONS,
C       ONE SHOULD REFER TO THE REPORT "INTERACTIVE
C       LEAST SQUARES" BY JANE E. RALL
C
C
        DIMENSION TITLE(35),NAM(40)
        DOUBLE PRECISION A(100,24)
        DOUBLE PRECISION T(100,6),Y(100),W(100)
        DOUBLE PRECISION ALF(12),C(7),NDI(15)
        DOUBLE PRECISION PH(8),DPH(8,12),ETA(100),R(100)
        DOUBLE PRECISION STP,CR(19,19),ANS
        INTEGER IOR(100),IA(8)
        INTEGER INC (12,8)
        INTEGER P
        LOGICAL IP,IQ,NC,IWT,ITB,ICV,ICR
        COMMON /BLOKIP/IP
        COMMON /BLKNDI/ NDI
        COMMON /BLKINC/INC
        COMMON /IPIBLK/ IPI
        COMMON /NCBLK/ NC
        COMMON /BLKPH/ PH
        COMMON /BLKDPH/ DPH
        DATA MMAX/100/
        DATA IP,IQ,NC,IWT,ITB,ICV,ICR/7*.FALSE./
        DATA IND/4HIND /, VAR/4HVAR /
```

```
C*************************************************************
C
C              INITIALIZATION SEGMENT OF PROGRAM
C
C*************************************************************
          TYPE 5
5         FORMAT (/' ENTER TITLE:  '$)
          ACCEPT 10,(TITLE(I),I=1,35)
10        FORMAT (35A1)
          TYPE 20
20        FORMAT (' DO YOU WANT LINE PRINTER OUTPUT?',
     *       ' (TYPE "Y" FOR YES)  '$)
          ACCEPT 30,IANS
30        FORMAT (A1)
          IF (IANS .EQ. 1HY) IP=.TRUE.
          IF (IP) PRINT 1105,TITLE
          TYPE 80
80        FORMAT (' ARE YOUR INITIAL VALUES ALREADY ON',
     *        ' FILE?  '$)
          ACCEPT 30,IANS
          IF (IANS. EQ. 1HY) GOTO 320
89        TYPE 90
90        FORMAT (' NO. OF DATA POINTS:  '$)
          ACCEPT 60,M
60        FORMAT (I)
          IF (IP) PRINT 91,M
91        FORMAT (' NO. OF DATA POINTS: ',I3)
          IF (IQ) GOTO 700
99        TYPE 100
100       FORMAT (' NO. OF LINEAR PARAMETERS:  '$)
          ACCEPT 60,N
          IF (IP) PRINT 101,N
101       FORMAT (' NO. OF LINEAR PARAMETERS: ',I2)
          IF (IQ) GOTO 700
109       TYPE 110
110       FORMAT (' NO. OF NONLINEAR PARAMETERS:  '$)
          ACCEPT 60,KG
          IF (IP) PRINT 111,KG
111       FORMAT (' NO. OF NONLINEAR PARAMETERS: ',I2)
          IF (IQ) GOTO 700
129       TYPE 130
130       FORMAT (' NO. OF INDEPENDENT VARIABLES:  '$)
          ACCEPT 60,IV
          IF (IP) PRINT 131,IV
131       FORMAT (' NO. OF INDEPENDENT VARIABLES: ',I2)
          IF (IQ) GOTO 700
C
C  INITIALIZE DATA POINTS & WEIGHTS
C
          TYPE 132
132       FORMAT (' ARE WEIGHTS TO BE ENTERED WITH DATA?  '$)
          ACCEPT 30,IANS
```

```
      IF (IANS .NE. 1HY) GOTO 1500
      TYPE 140
      IF (IP) PRINT 140
140   FORMAT(' DATA POINTS:  LIST INDEPENDENT VARIABLES,'/
     *  'DEPENDENT VARIABLE & WEIGHT, SEPARATED BY BLANKS')
      DO 145 J=1,M
      TYPE 70,J
70    FORMAT (' ',I2,'  '$)
      ACCEPT 150,(T(J,I),I=1,IV),Y(J),W(J)
150   FORMAT (8D)
      IF (IP) PRINT 151,J,(T(J,I),I=1,IV),Y(J),W(J)
151   FORMAT (I3,3X,8E)
145   CONTINUE
      GOTO 1601
1500  TYPE 1501
1501  FORMAT (' ARE WEIGHTS TO BE CALCULATED IN',
     *     ' SUBROUTINE WT?  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 1505
      DO 1504 J=1,M
1504  W(J)=1.
      TYPE 1508
      IF (IP) PRINT 1508
1508  FORMAT (' WEIGHTS ARE SET TO 1  '/)
      GOTO 1550
1505  CONTINUE
      IWT=.TRUE.
      IF (IP) PRINT 1509
1509  FORMAT (' WEIGHTS CALCULATED IN SUBROUTINE WT '/)
1550  TYPE 1551
1551  FORMAT (' DATA POINTS:  LIST INDEPENDENT VARIABLES',
     *   ' AND'/' DEPENDENT VARIABLE, SEPARATED BY BLANKS')
      DO 1601 I=1,M
         TYPE 70,I
         ACCEPT 150,(T(I,J),J=1,IV),Y(I)
         IF (IP) PRINT 151,I,(T(I,J),J=1,IV),Y(I)
1601  CONTINUE
      IF (IWT) CALL WT(M,W,Y)
      IF (KG .EQ. 0) GOTO 700
      TYPE 160
      IF (IP) PRINT 160
160   FORMAT (' INITIAL VALUES OF NONLINEAR PARAMETERS')
      DO 165 J=1,KG
      TYPE 70,J
      ACCEPT 170,ALF(J)
170   FORMAT (D)
      IF (IP) PRINT 171,J,ALF(J)
171   FORMAT (I3,3X,E)
165   CONTINUE
C
C  DETERMINE INCIDENCE MATRIX
C
```

```
      NP1=N+1
      PH(NP1)=0.
      DO 201 J=1,NP1
         DO 201 K=1,KG
201         INC(K,J)=0
      DO 204 J=1,KG
         TYPE 202,J
202      FORMAT (' WHICH TERMS OF THE MODEL DEPEND ON ',
     *      'NONLINEAR PARAMETER ',I2,':  '$)
         ACCEPT 203,(IA(K),K=1,NP1)
203      FORMAT (8I)
         DO 204 K=1,NP1
            IF(IA(K) .EQ. 0) GOTO 204
            IB=IA(K)
            INC(J,IB)=1
204   CONTINUE
      P=0
      DO 205 J=1,NP1
        DO 205 I=1,KG
205        IF (INC(I,J) .EQ. 1)  P=P+1
      IF (IQ) GOTO 700
      IQ=.TRUE.
C
C  INITIALIZE NUMERICAL DERIVATIVE INCREMENTS
C
      TYPE 180
180   FORMAT (' ALL DERIVATIVES IN CLOSED FORM?  '$)
      ACCEPT 30, IANS
      IF (IANS .EQ. 1HY) GOTO 700
      TYPE 184
184   FORMAT (' ALL NUMERICAL DERIVATIVE INCREMENTS THE',
     *   ' SAME?  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 196
      TYPE 190
190   FORMAT (' NUMERICAL DERIVATIVE INCREMENTS'/
     *  ' (ENTER 0 FOR THOSE TO BE FOUND FROM CLOSED FORM)')
      IF (IP) PRINT 191
191   FORMAT (' NUMERICAL DERIVATIVE INCREMENTS')
      DO 195 K=1,P
      TYPE 70,K
      ACCEPT 170,NDI(K)
195   IF (IP) PRINT 171,K,NDI(K)
      GOTO 700
196   CONTINUE
      TYPE 197
197   FORMAT (' INCREMENT VALUES:  '$)
      ACCEPT 170,ANS
      DO 199 I=1,P
199   NDI(I)=ANS
      GOTO 700
320   CONTINUE
```

```
C
C  READ IN DATA THAT HAS BEEN SAVED
C
      TYPE 330
330   FORMAT (' TYPE NAME OF FILE:  '$)
      ACCEPT 340,FILE
340   FORMAT (A5)
      CALL IFILE(3,FILE)
      READ (3,360) (NAM(I),I=1,11),M,(NAM(I),I=1,20),N,
     1          (NAM(I),I=1,23),KG,(NAM(I),I=1,24),
     2          IV,(NAM(I),I=1,32)
360   FORMAT (11A1,I/20A1,I/23A1,I/24A1,I/32A1)
      DO 350 I=1,M
350   READ (3,150) (T(I,J),J=1,IV),Y(I),W(I)
      READ (3,370) (NAM(I),I=1,23),(ALF(I),I=1,KG)
370   FORMAT (23A1,12E)
      NP1=N+1
      PH(NP1)=0.
      DO 391 J=1,NP1
         DO 391 K=1,KG
391         INC(K,J)=0
      DO 392 J=1,KG
      READ (3,390) (NAM(I),I=1,38),(IA(K),K=1,NP1)
390   FORMAT (38A1,8I2)
      DO 392 I=1,NP1
         IF(IA(I) .EQ. 0) GOTO 392
         IB=IA(I)
         INC(J,IB)=1
392   CONTINUE
      P=0
      DO 393 J=1,NP1
         DO 393 I=1,KG
393         IF (INC(I,J) .EQ. 1) P=P+1
      IQ=.TRUE.
      READ (3,380) (NAM(I),I=1,34),(NDI(I),I=1,P)
380   FORMAT (34A1,15E)
      REWIND 3
      CALL PR (1,MMAX,M,N,KG,P,IV,T,Y,W,ALF,NDI)
C*************************************************************
C
C                     COMMAND POINT
C
C*************************************************************
700   CONTINUE
      TYPE 780
780   FORMAT (/' NEXT STEP? (TYPE "H" FOR HELP)  '$)
      ACCEPT 800,I
800   FORMAT (A4)
      IF (I .NE. 1HH) GOTO 860
      TYPE 820
      IF (IP) PRINT 820
820   FORMAT (1H1,'THE COMMAND CODES FALL INTO FOUR MAIN',
```

```
     *   ' CATEGORIES'//
     1   '      PRINT-  TO EXAMINE CURRENT VALUES OF',
     *                  ' INITIALIZED DATA'/
     2   '      CHANGE- TO CORRECT ANY INITIALIZED VALUE AND',
     *                  ' EITHER'/
     3   '              ADD OR DELETE DATA POINTS'/
     4   '      RUN-    TO RUN THE LEAST SQUARES PROGRAM TO',
     *                  ' FIND THE'/
     5   '              OPTIMAL PARAMETERS, OR TO RUN A ',
     *                  'STATISTICAL'/
     6   '              ANALYSIS OF THE DATA RETURNED'/
     7   '      STOP-   TO STOP THE PROGRAM, WITH OR WITHOUT',
     *                  ' SAVING'/
     8   '              THE INITIALIZED DATA.'//
     9   ' TYPE "P","C","R", OR "S" FOR THE SPECIFIC',
     *   ' COMMANDS IN'/' THE ABOVE CATEGORIES.   TYPE "A"',
     *   ' TO OBTAIN A LISTING OF'/' ALL COMMAND CODES.   '$)
          ACCEPT 30,IANS
          IF (IANS .EQ. 1HC) GOTO 830
          IF (IANS .EQ. 1HR) GOTO 840
          IF (IANS .EQ. 1HS) GOTO 850
          TYPE 825
          IF (IP) PRINT 825
825       FORMAT (' COMMAND CODES FOR PRINTING'//
     1   '      PM     PRINT THE NO. OF DATA POINTS'/
     2   '      PN     PRINT THE NO. OF LINEAR PARAMETERS'/
     3   '      PKG    PRINT THE NO. OF NONLINEAR PARAMETERS'/
     4   '      PIV    PRINT THE NO. OF INDEPENDENT VARIABLES'/
     5   '      PTW    PRINT THE DATA POINTS & WEIGHTS'/
     6   '      PALF   PRINT THE VALUES OF THE NONLINEAR',
     *                  ' PARAMETERS'/
     7   '      PNDI   PRINT THE NUMERICAL DERIVATIVE',
     *                  ' INCREMENTS'/
     8   '      PINC   PRINT WHICH TERMS DEPEND ON NONLIN',
     *                  ' PARAMETERS'/
     9   '      LP     TURN ON THE LINE PRINTER FLAG'/
     *   '      NP     TURN OFF THE LINE PRINTER FLAG'///)
          IF (IANS .NE. 1HA) GOTO 700
830       CONTINUE
          TYPE 835
          IF (IP) PRINT 835
835       FORMAT (' COMMAND CODES FOR CHANGING'//
     1   '      CM     CHANGE NO. OF DATA POINTS'/
     2   '      CN     CHANGE NO. OF LINEAR PARAMETERS'/
     3   '      CKG    CHANGE NO. OF NONLINEAR PARAMETERS'/
     4   '      CIV    CHANGE NO. OF INDEPENDENT VARIABLES'/
     5   '      CT     CHANGE ONE OR MORE DATA POINTS'/
     6   '      CW     CHANGE ONE OR MORE WEIGHTS'/
     7   '      CALF   CHANGE ONE OR MORE NONLINEAR',
     *                  ' PARAMETERS'/
     8   '      CNDI   CHANGE ONE OR MORE NUMER DERIV',
     *                  ' INCREMENTS'/
```

```
     9  '      CINC  CHANGE WHICH TERMS DEPEND ON NONLIN',
     *             ' PARAMETERS'///)
        IF (IANS .NE. 1HA) GOTO 700
840     CONTINUE
        TYPE 845
        IF (IP) PRINT 845
845     FORMAT (' COMMAND CODES FOR RUNNING'//
     1  '      RV    RUN THE LEAST SQUARES PROGRAM'/
     2  '      RS    RUN A STATISTICAL ANALYSIS OF',
     *             ' RESULTS'///)
        IF (IANS .NE. 1HA) GOTO 700
850     CONTINUE
        TYPE 855
        IF (IP) PRINT 855
855     FORMAT (' COMMAND CODES FOR STOPPING'//
     1  '      SD    SAVE CURRENT VALUES OF DATA & STOP'/
     2  '      S0    STOP WITHOUT SAVING'///)
        GOTO 700
860     CONTINUE
        IF (I .EQ. 2HRV) GOTO 1100
        IF (I .EQ. 2HRS) GOTO 1200
        IF ((I .EQ. 2HPM) .OR. (I .EQ. 2HPN) .OR.
     1    (I .EQ. 3HPKG) .OR. (I .EQ. 3HPIV) .OR.
     2    (I .EQ. 3HPTW) .OR. (I .EQ. 4HPALF) .OR.
     3    (I .EQ. 4HPNDI) .OR. (I .EQ. 2HLP) .OR.
     4    (I .EQ. 2HNP) .OR. (I .EQ. 4HPINC)) GOTO 2100
        IF (I .EQ. 2HCM) GOTO 89
        IF (I .EQ. 2HCN) GOTO 99
        IF (I .EQ. 3HCKG) GOTO 109
        IF (I .EQ. 3HCIV) GOTO 129
        IF (I .EQ. 4HCINC) GOTO 165
        IF ((I .EQ. 2HCT) .OR. (I .EQ. 2HCW) .OR.
     *     (I .EQ. 4HCALF) .OR. (I .EQ. 4HCNDI)) GOTO 1300
        IF (I .EQ. 2HSD) GOTO 3200
        IF (I .EQ. 2HS0) GOTO 3300
        GOTO 3100
C
C ********** RUN THE LEAST SQUARES PROGRAM **********
C
1100    CONTINUE
        TYPE 1102
1102    FORMAT (/' OUTPUT PRINTING INDEX'/
     *    ' (SEE DOCUMENTATION.  "1" IS RECOMMENDED):  '$)
        ACCEPT 60,IPRINT
        IF (IP) PRINT 1103,IPRINT
1103    FORMAT (' OUTPUT PRINTING INDEX:',I3)
        TYPE 1105,TITLE
        IF (IP) PRINT 1105,TITLE
1105    FORMAT (1H1,20X,35A1//)
        CALL VARPRO (N,KG,M,MMAX,N+P+2,IV,T,Y,W,A,
     *    IPRINT,ALF,C,IERR)
        IF (IERR .LE. -4) GOTO 700
```

```
      NC=.FALSE.
      NPKG=N+KG
      TYPE 1150
1150  FORMAT (' INDICATE WHICH OF THE FOLLOWING SHOULD',
     *   ' BE PRINTED'/' BY TYPING "Y": '/'   TABLE OF',
     *   ' OBSERVED & PREDICTED VALUES, WTD RESIDUAL: '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) ITB=.TRUE.
      TYPE 1152
1152  FORMAT ('   COVARIANCE MATRIX:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) ICV=.TRUE.
      TYPE 1154
1154  FORMAT ('   CORRELATION MATRIX:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) ICR=.TRUE.
      IF (.NOT. ICV) GOTO 1159
      ICV=.FALSE.
      TYPE 1110
      IF (IP) PRINT 1110
1110  FORMAT ('///15X,'COVARIANCE MATRIX')
      CALL ARRAYD (A,NPKG,NPKG,MMAX,'     ')
1159  DO 1168 I=1,M
         II=I
1168     CALL CALC (II,KG,N,MMAX,IV,C,ALF,T,ETA(I))
      IF (.NOT. ITB) GOTO 1175
      ITB=.FALSE.
      TYPE 1160
      IF (IP) PRINT 1160
1160  FORMAT (//' INDEX',2X,'IND VAR 1',4X,'OBS VAL',
     *   6X,'PRED VAL',5X,'WTD RESID',5X,'WEIGHTS'/)
      DO 1170 I=1,M
      TYPE 1165,I,T(I,1),Y(I),ETA(I),A(I,NPKG+1),W(I)
      IF (IP) PRINT 1165,I,T(I,1),Y(I),ETA(I),
     *   A(I,NPKG+1),W(I)
1165  FORMAT (I4,1P5E13.5)
1170  CONTINUE
      IF (IV .EQ. 1) GOTO 1175
      TYPE 1173,(IND,VAR,J,J=2,IV)
      IF (IP) PRINT 1173,(IND,VAR,J,J=2,IV)
1173  FORMAT (/' INDEX',3X,5(2A4,I2,4X))
      DO 1175 I=1,M
      TYPE 1174,I,(T(I,J),J=2,IV)
      IF (IP) PRINT 1174,I,(T(I,J),J=2,IV)
1174  FORMAT (I3,3X,1P5E13.5)
1175  CONTINUE
      IF (.NOT. ICR) GOTO 1169
      ICR=.FALSE.
      DO 1167 I=1,NPKG
         DO 1167 J=I,NPKG
            CR(I,J)=A(I,J)/(DSQRT(A(I,I))*DSQRT(A(J,J)))
1167        CR(J,I)=CR(I,J)
```

```
      TYPE 1166
      IF (IP) PRINT 1166
1166  FORMAT (///15X,' CORRELATION MATRIX')
      CALL ARRAYD (CR,NPKG,NPKG,19,'     ')
1169  DO 1172 I=1,M
1172  R(I)=Y(I)-ETA(I)
      TYPE 1171
1171  FORMAT (//' PLOT ROUTINE CURRENTLY SET NOT TO DO',
     *  ' ANY PLOTS.'/' DO YOU WANT TO START PLOTTING? '$)
      ACCEPT 30,IANS
      IF (IANS .NE. 1HY) GOTO 700
      IF (IV .NE. 1) GOTO 1186
      INV=1
      GOTO 1190
1186  CONTINUE
      TYPE 1188
1188  FORMAT (/' WHICH INDEPENDENT VARIABLE SHOULD BE',
     *  ' USED?'/' (TYPE ITS NUMBER)  '$)
      ACCEPT 60,INV
1190  CONTINUE
      CALL ORPLOT (T(1,INV),M,IOR)
      CALL INTRVL (T(1,INV),M,IOR,STP)
1191  CONTINUE
      TYPE 1192
1192  FORMAT (/' IF YOU WANT TO PLOT RESIDUALS, TYPE "1"'/
     *    ' IF YOU WANT TO PLOT OBSERVED & PREDICTED',
     *    ' VALUES, TYPE "2":  '  $)
      ACCEPT 60,IPI
1193  CONTINUE
      IF (IPI .NE. 1) GOTO 1195
      TYPE 1194
1194  FORMAT (1H1,20X,'PLOT OF RESIDUALS'//)
      IF (IP) PRINT 1194
      TYPE 1199,INV
1199  FORMAT ('  IND VAR',I2)
      IF (IP) PRINT 1199,INV
      CALL SETUP (T(1,INV),M,Y,R,IOR,STP)
      GOTO 1196
1195  TYPE 1400
1400  FORMAT (1H1,10X,'PLOT OF OBSERVED (X) & PREDICTED',
     *  ' (+) VALUES'/8X,'(* INDICATES OBSERVED &',
     *  ' PREDICTED VALUES THE SAME)'//)
      IF (IP) PRINT 1400
      TYPE 1199,INV
      IF (IP) PRINT 1199,INV
      CALL SETUP (T(1,INV),M,Y,ETA,IOR,STP)
1196  TYPE 1197
1197  FORMAT (/' DO YOU WANT TO CONTINUE PLOTTING?  '$)
      ACCEPT 30,IANS
      IF (IANS .NE. 1HY) GOTO 700
      IF (IV .EQ. 1) GOTO 1191
      TYPE 1198
```

```
1198      FORMAT (/' USE SAME INDEPENDENT VARIABLE?  '$)
          ACCEPT 30,IANS
          IF (IANS .NE. 1HY) GOTO 1186
          GOTO 1191
C
C ********** RUN THE STATISTICAL PACKAGE **********
C
1200      CONTINUE
          NKG2=N+KG+2
          CALL STAT (MMAX,M,N,NKG2,KG,Y,W,C,ALF,A,IV,T,ETA,R)
          GOTO 700
C
C ********** CALL SUBROUTINE TO CHANGE **********
C
1300      CONTINUE
          CALL CH(I,MMAX,M,KG,IV,T,Y,W,ALF)
          GOTO 700
C
C ********** CALL SUBROUTINE TO PRINT **********
C
2100      CONTINUE
          CALL PR(I,MMAX,M,N,KG,P,IV,T,Y,W,ALF)
          GOTO 700
C
C ********** PRINT COMMAND CODE ERROR MESSAGE **********
C
3100      CONTINUE
          TYPE 3110
3110      FORMAT (' COMMAND CODE WAS INVALID'//)
          GOTO 700
C
C ********** SAVE CURRENT VALUES OF DATA & STOP **********
C
3200      CONTINUE
          TYPE 3210
3210      FORMAT (' TYPE NAME OF FILE FOR STORING DATA:  '$)
          ACCEPT 340,FILE
          CALL OFILE (3,FILE)
          WRITE (3,3260) M,N,KG,IV
3260      FORMAT (' DATA PTS: ',I/' LINEAR PARAMETERS: ',I/
     1     ' NONLINEAR PARAMETERS: ',I/
     2     ' INDEPENDENT VARIABLES: ',I/
     3     ' INDEP. VAR., DEP. VAR. & WEIGHTS: ')
          DO 3220 I=1,M
3220      WRITE (3,150) (T(I,J),J=1,IV),Y(I),W(I)
          WRITE (3,3270) (ALF(I),I=1,KG)
3270      FORMAT (' NONLINEAR PARAMETERS: ',12E)
          DO 3289 I=1,KG
             DO 3281 K=1,NP1
3281            IA(K)=0.
             K=1
             DO 3282 J=1,NP1
```

```
                IF (INC(I,J).NE.1) GOTO 3282
                IA(K)=J
                K=K+1
3282          CONTINUE
              WRITE (3,3285) I,(IA(K),K=1,NP1)
3285          FORMAT ('TERMS WHICH DEPEND ON NONLIN PARM',I2,
     *          ':  ',8I2)
3289        CONTINUE
            WRITE (3,3280) (NDI(I),I=1,P)
3280        FORMAT (' NUMERICAL DERIVATIVE INCREMENTS: ',15E)
            REWIND 3
            IF (IP) PRINT 3290,FILE
3290        FORMAT (//' DATA IS STORED IN FILE:  ',A5/)
C
C ********** STOP WITHOUT SAVING DATA **********
C
3300        CONTINUE
            STOP
            END
```

```
      SUBROUTINE CH(I,MMAX,M,KG,IV,T,Y,W,ALF)
C
C   CHANGE THE VALUE OF DATA POINTS, WEIGHTS, NONLINEAR
C   PARAMETERS OR NUMERICAL DERIVATIVE INCREMENTS
C   (DEPENDING ON THE COMMAND CODE ENTERED).
C
      DOUBLE PRECISION T(MMAX,IV),Y(M),W(M),ALF(KG)
      DOUBLE PRECISION NDI(15)
      LOGICAL IP
      COMMON /BLKNDI/ NDI
      COMMON /BLOKIP/ IP
60    FORMAT (I)
30    FORMAT (A1)
150   FORMAT (8E)
170   FORMAT (E)
      IF (I .EQ. 2HCT) GOTO 1800
      IF (I .EQ. 2HCW) GOTO 1850
      IF (I .EQ. 4HCALF) GOTO 1900
      IF (I .EQ. 4HCNDI) GOTO 2000
C
C ********** CHANGE ONE OR MORE DATA POINTS **********
C
1800  CONTINUE
      TYPE 1810
1810  FORMAT (' WHICH DATA POINT IS TO BE CHANGED?')
      TYPE 1811
1811  FORMAT (' (TYPE ITS INDEX NO. AS IT APPEARS IN THE',
     *  ' LIST)  '$)
      ACCEPT 60,IL
      TYPE 1820,IL
1820  FORMAT (' ENTER NEW INDEP & DEP VARIABLES FOR DATA',
     *  ' POINT',I3,': ')
      ACCEPT 150,(T(IL,J),J=1,IV),Y(IL)
      IF (IP) PRINT 1821,IL,(T(IL,J),J=1,IV),Y(IL)
1821  FORMAT (' NEW INDEP & DEPENDENT VARIABLES FOR ',
     *  'POINT',I3,': '/7E)
      TYPE 1830
1830  FORMAT (' DO YOU WANT TO CHANGE ANY OTHERS?  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 1800
      GOTO 700
C
C ********** CHANGE ONE OR MORE WEIGHTS **********
C
1850  CONTINUE
      TYPE 1851
1851  FORMAT (' WHICH WEIGHT IS TO BE CHANGED?')
      TYPE 1811
      ACCEPT 60,IL
      TYPE 1852,IL
1852  FORMAT (' ENTER NEW WEIGHT FOR DATA POINT',I4,': '$)
      ACCEPT 170,W(IL)
```

```
      IF (IP) PRINT 1853,IL,W(IL)
1853  FORMAT (' NEW WEIGHT FOR DATA POINT',I4,': ',E)
      TYPE 1830
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 1850
      GOTO 700
C
C ******** CHANGE ONE OR MORE NONLINEAR PARAMETERS *********
C
1900  CONTINUE
      TYPE 1910
1910  FORMAT (' WHICH NONLINEAR PARAMETER IS TO BE',
     *  ' CHANGED?')
      TYPE 1811
      ACCEPT 60,IL
      TYPE 1920,IL
1920  FORMAT (' ENTER NEW VALUE FOR PARAMETER',I4,': '$)
      ACCEPT 170,ALF(IL)
      IF (IP) PRINT 1921,IL,ALF(IL)
1921  FORMAT (' NEW VALUE FOR NONLINEAR PARAMETER',I4,
     *  ': ',E)
      TYPE 1830
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 1900
      GOTO 700
C
C ******* CHANGE ONE OR MORE NUMER DERIV INCREMENTS ********
C
2000  CONTINUE
      TYPE 2010
2010  FORMAT (' WHICH NUMER. DERIV. INCREMENT IS TO BE',
     *  ' CHANGED?')
      TYPE 1811
      ACCEPT 60,IL
      TYPE 2020,IL
2020  FORMAT (' NEW INCREMENT FOR DERIVATE',I4,': '$)
      ACCEPT 170,NDI(IL)
      IF (IP) PRINT 2021,IL,NDI(IL)
2021  FORMAT (' NEW INCREMENT FOR DERIVATIVE',I4,': ',E)
      TYPE 1830
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 2000
700   RETURN
      END
```

```
      SUBROUTINE PR(I,MMAX,M,N,KG,P,IV,T,Y,W,ALF)
C
C   PRINT THE VALUES OFANY OR ALL OF THE INITIALIZED
C   PARAMETERS.  THIS SUBROUTINE IS CALLED IN
C   RESPONSE TO ONE OF THE "PRINT" COMMANDS OR
C   WHEN THE INITIALIZED DATA IS OBTAINED FROM A
C   SAVED FILE.
C
      DOUBLE PRECISION T(MMAX,IV),Y(M),W(M),ALF(KG)
      DOUBLE PRECISION NDI(15)
      INTEGER INC(12,8),IA(8),P
      LOGICAL IP
      COMMON /BLKINC/ INC
      COMMON /BLKNDI/ NDI
      COMMON /BLOKIP/ IP
      DATA IND/4HIND /,VAR/4HVAR /
60    FORMAT (I)
30    FORMAT (A1)
1811  FORMAT (' (TYPE ITS INDEX NO. AS IT APPEARS IN THE',
     *  ' LIST)'$)
      IF (I .EQ. 1) GOTO 3050
      IF (I .EQ. 2HPM) GOTO 2100
      IF (I .EQ. 2HPN) GOTO 2200
      IF (I .EQ. 3HPKG) GOTO 2300
      IF (I .EQ. 4HPINC) GOTO 2400
      IF (I .EQ. 3HPIV) GOTO 2500
      IF (I .EQ. 3HPTW) GOTO 2600
      IF (I .EQ. 4HPALF) GOTO 2700
      IF (I .EQ. 4HPNDI) GOTO 2800
      IF (I .EQ. 2HLP) GOTO 2900
      IF (I .EQ. 2HNP) GOTO 3000
C
C ********** PRINT THE NO. OF DATA POINTS **********
C
2100  CONTINUE
      TYPE 2110,M
      IF (IP) PRINT 2110,M
2110  FORMAT (' NO. OF DATA POINTS:   ',I4)
      GOTO 700
C
C ********** PRINT THE NO. OF LINEAR PARAMETERS **********
C
2200  CONTINUE
      TYPE 2210,N
      IF (IP) PRINT 2210,N
2210  FORMAT (' NO. OF LINEAR PARAMETERS:   ',I4)
      GOTO 700
C
C ********* PRINT THE NO. OF NONLINEAR PARAMETERS *********
C
2300  CONTINUE
      TYPE 2310,KG
```

```
      IF (IP) PRINT 2310,KG
2310  FORMAT (' NO. OF NONLINEAR PARAMETERS:    ',I4)
      GOTO 700
C
C ******** PRINT WHICH TERMS DEPEND ON NONLIN PARM *********
C
2400  CONTINUE
      NP1=N+1
      DO 2440 I=1,KG
      K=1
      DO 2405 J=1,NP1
2405     IA(J)=0
         DO 2410 J=1,NP1
            IF (INC(I,J).NE.1) GOTO 2410
            IA(K)=J
            K=K+1
2410     CONTINUE
      TYPE 2430,I,(IA(K),K=1,NP1)
2430  FORMAT (' TERMS WHICH DEPEND ON NONLIN PARM',
     *   I2,':  ',8I2)
2440  IF (IP) PRINT 2430,I,(IA(K),K=1,NP1)
      GOTO 700
C
C ********* PRINT THE NO. OF INDEPENDENT VARIABLES *********
C
2500  CONTINUE
      TYPE 2510,IV
      IF (IP) PRINT 2510,IV
2510  FORMAT (' NO. OF INDEPENDENT VARIABLES:    ',I4)
      GOTO 700
C
C ********** PRINT THE DATA POINTS & WEIGHTS **********
C
2600  CONTINUE
      TYPE 2605
2605  FORMAT (' ALL OF THE DATA POINTS?  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) GOTO 2640
2620  CONTINUE
      TYPE 2606
2606  FORMAT (' WHICH DATA POINT?  ')
      TYPE 1811
      ACCEPT 60,IL
      TYPE 2607,IL
      IF (IP) PRINT 2607,IL
2607  FORMAT (' FOR POINT',I4,'VALUES OF INDEP. VAR,',
     *   ' DEP VAR & WT ARE')
      TYPE 2608,(T(IL,J),J=1,IV),Y(IL),W(IL)
      IF (IP) PRINT 2608,(T(IL,J),J=1,IV),Y(IL),
     *   W(IL)
2608  FORMAT (8E,/)
      TYPE 2609
```

```
2609      FORMAT (' DO YOU WANT TO EXAMINE ANY OTHERS?  '$)
          ACCEPT 30,IANS
          IF (IANS .EQ. 1HY) GOTO 2620
          GOTO 700
2640      CONTINUE
          IVP=MIN0(IV,2)
          TYPE 2610,(IND,VAR,J,J=1,IVP)
          IF (IP) PRINT 2610,(IND,VAR,J,J=1,IVP)
2610      FORMAT (' INDEX',5X,'WEIGHT',9X,'DEP VAR',8X,
     *     2(2A4,I2,5X))
          DO 2650 K=1,M
          TYPE 2651,K,W(K),Y(K),(T(K,J),J=1,IVP)
          IF (IP) PRINT 2651,K,W(K),Y(K),
     *     (T(K,J),J=1,IVP)
2651      FORMAT (I3,3X,4E)
2650      CONTINUE
          IF (IV .LE. 2) GOTO 700
          TYPE 2655,(IND,VAR,J,J=3,IV)
2655      FORMAT (/' INDEX',3X,4(2A4,I2,5X))
          IF (IP) PRINT 2655,(IND,VAR,J,J=3,IV)
          DO 2660 K=1,M
          TYPE 2658,K,(T(K,J),J=3,IV)
          IF (IP) PRINT 2658,K,(T(K,J),J=3,IV)
2658      FORMAT (I3,3X,4E)
2660      CONTINUE
          GOTO 700
C
C ****** PRINT THE VALUES OF THE NONLINEAR PARAMETERS ******
C
2700      CONTINUE
          TYPE 2705
2705      FORMAT (' ALL THE NONLINEAR PARAMETERS?  '$)
          ACCEPT 30,IANS
          IF (IANS .EQ. 1HY) GOTO 2740
2720      CONTINUE
          TYPE 2706
2706      FORMAT (' WHICH PARAMETER?  ')
          TYPE 1811
          ACCEPT 60,IL
          TYPE 2707,IL,ALF(IL)
          IF (IP) PRINT 2707,IL,ALF(IL)
2707      FORMAT (' NONLINEAR PARAMETER',I4,' IS',E)
          TYPE 2609
          ACCEPT 30,IANS
          IF(IANS .EQ. 1HY) GOTO 2720
          GOTO 700
2740      CONTINUE
          TYPE 2710
          IF (IP) PRINT 2710
2710      FORMAT (' NONLINEAR PARAMETERS:   ')
          DO 2750 K=1,KG
          TYPE 2751,K,ALF(K)
```

```
          IF (IP) PRINT 2751,K,ALF(K)
2751      FORMAT (I3,3X,E)
2750      CONTINUE
          GOTO 700
C
C ********** PRINT THE NUMER DERIV INCREMENTS **********
C
2800      CONTINUE
          TYPE 2810
          IF (IP) PRINT 2810
2810      FORMAT (' NUMERICAL DERIVATIVE INCREMENTS:')
          DO 2830 J=1,P
          TYPE 2751,J,NDI(J)
          IF (IP) PRINT 2751,J,NDI(J)
2830      CONTINUE
          GOTO 700
C
C ********** TURN ON THE LINE PRINTER FLAG **********
C
2900      CONTINUE
          IP=.TRUE.
          GOTO 700
C
C ********** TURN OFF THE LINE PRINTER FLAG **********
C
3000      CONTINUE
          IP=.FALSE.
          GOTO 700
C
C ********** PRINT ALL INITIAL VALUES **********
C
3050      CONTINUE
          TYPE 2110,M
          IF (IP) PRINT 2110,M
          TYPE 2210,N
          IF (IP) PRINT 2210,N
          TYPE 2310,KG
          IF (IP) PRINT 2310,KG
          TYPE 2510,IV
          IF (IP) PRINT 2510,IV
          IVP=MIN0(IV,2)
          TYPE 2610,(IND,VAR,J,J=1,IVP)
          IF (IP) PRINT 2610,(IND,VAR,J,J=1,IVP)
          DO 3055 K=1,M
          TYPE 2651,K,W(K),Y(K),(T(K,J),J=1,IVP)
3055      IF (IP) PRINT 2651,K,W(K),Y(K),
     *     (T(K,J),J=1,IVP)
          IF (IV .LE. 2) GOTO 3056
          TYPE 2655,(IND,VAR,J,J=3,IV)
          IF (IP) PRINT 2655,(IND,VAR,J,J=3,IV)
          DO 3056 K=1,M
          TYPE 2658,K,(T(K,J),J=3,IV)
```

```
      IF (IP) PRINT 2658,K,(T(K,J),J=3,IV)
3056  CONTINUE
      IF (KG .EQ. 0) GOTO 700
      TYPE 2710
      IF (IP) PRINT 2710
      DO 3060 K=1,KG
      TYPE 2751,K,ALF(K)
      IF (IP) PRINT 2751,K,ALF(K)
3060  CONTINUE
      TYPE 2810
      IF (IP) PRINT 2810
      DO 3065 K=1,P
      TYPE 2751,K,NDI(K)
      IF (IP) PRINT 2751,K,NDI(K)
3065  CONTINUE
      GOTO 2400
700   RETURN
      END
```

```
      SUBROUTINE STAT (MMAX,M,N,NKG2,KG,Y,W,C,ALF,A,
     *    IV,T,ETA,R)
C
C   PROVIDE THE INDICATED INFORMATION IN RESPONSE TO
C   THE RS COMMAND:
C     (1)  STANDARD DEVIATION OF THE PARAMETERS
C     (2)  T-RATIO
C     (3)  RESIDUAL SUM OF SQUARES
C     (4)  RESIDUAL STANDARD ERROR
C     (5)  COEFFICIENT OF DETERMINATION
C     (6)  STANDARDIZED RESIDUALS
C     (7)  PLOT OF STANDARDIZED RESIDUALS VS INDEPENDENT
C          VARIABLES
C     (8)  PLOT OF STANDARDIZED RESIDUALS VS PREDICTED
C          VALUES
C     (9)  PROBABILITY PLOT OF STANDARDIZED RESIDUALS.
C
      DOUBLE PRECISION Y(M),C(N),ALF(KG),A(MMAX,NKG2)
      DOUBLE PRECISION T(MMAX,IV),W(M),ETA(M),R(M)
      DOUBLE PRECISION AI,AJ,AN,STP,RSS,RMS,RSD
      DOUBLE PRECISION CTSS,R2,BB,TR,YB,SUM
      DOUBLE PRECISION BM(100,19),XX(100,19),PH(8)
      DOUBLE PRECISION SYH(100),SR(100),SRV(100)
      LOGICAL IP,IPT,IR,ITB,IPIV,IPPV,IPP
      DIMENSION IOR(100)
      COMMON /IPIBLK/ IPI
      COMMON /BLOKIP/ IP
      COMMON /BLKPH/ PH
      COMMON /BLKBM/ BM
      DATA PI/3.141592654/
      DATA IND/4HIND /, VAR/4HVAR /
      IPI=1
      NP1=N+1
      NPKG=N+KG
C
C  DETERMINE WHICH STATISTICS ARE TO BE PRINTED
C
      TYPE 1100
1100  FORMAT (' INDICATE WHICH OF THE FOLLOWING SHOULD',
     *    ' BE PRINTED:'/'   PARAMETERS, THEIR STD DEV',
     *    ' & T-RATIO:  '$)
      ACCEPT 30,IANS
30    FORMAT (A1)
      IF (IANS .EQ. 1HY) IPT=.TRUE.
      TYPE 1102
1102  FORMAT ('   RES SUM OF SQS, RES MEAN SQ, RES ',
     *    'STD DEV &'/'       COEFF OF DETERMINATION:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) IR =.TRUE.
      TYPE 1104
1104  FORMAT ('   TABLE OF OBS, PRED VAL, RESID & STD',
     *    ' RESID:  '$)
```

```
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) ITB=.TRUE.
      TYPE 1106
1106  FORMAT('   PLOT OF STD RES VS INDEP VAR:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) IPIV=.TRUE.
      TYPE 1108
1108  FORMAT ('   PLOT OF STD RES VS PRED VAL:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) IPPV=.TRUE.
      TYPE 1110
1110  FORMAT ('   PROBABILITY PLOT OF STD RES:  '$)
      ACCEPT 30,IANS
      IF (IANS .EQ. 1HY) IPP=.TRUE.
      IF (.NOT. IPT) GOTO 1241
C
C  PRINT PARAMETERS, THEIR STD DEV & T-RATIOS
C
      IPT=.FALSE.
      TYPE 1210
      IF (IP) PRINT 1210
1210  FORMAT (//' PARAMETER',8X,'S.D. OF PARM.',6X,
     *  'T-RATIO'/)
      IF (N .EQ. 0) GOTO 1230
      DO 1230 I=1,N
      AI=DABS(A(I,I))
      BB=DSQRT(AI)
      TR=C(I)/BB
      TYPE 1220,C(I),BB,TR
      IF (IP) PRINT 1220,C(I),BB,TR
1220  FORMAT (3E)
1230  CONTINUE
      IF (KG .EQ. 0) GOTO 1241
      DO 1240 I=1,KG
      J=N+I
      AJ=DABS(A(J,J))
      BB=DSQRT(AJ)
      TR=ALF(I)/BB
      TYPE 1220,ALF(I),BB,TR
1240  IF(IP) PRINT 1220,ALF(I),BB,TR
1241  IF (.NOT. IR) GOTO 1275
C
C  PRINT RESIDUAL SUM OF SQUARES, RES MEAN SQUARE &
C     RES STD ERROR.
C  PRINT COEFFICIENT OF DETERMINATION.
C
      IR=.FALSE.
      RSS=(A(1,NKG2))**2
      RMS=A(2,NKG2)
      RSD=DSQRT(DABS(RMS))
      TYPE 1250,RSS
      IF (IP) PRINT 1250,RSS
```

```
1250      FORMAT (//' WTD RESIDUAL SUM OF SQUARES:  ',E)
          TYPE 1252,RMS
          IF (IP) PRINT 1252,RMS
1252      FORMAT (' WTD RESIDUAL MEAN SQUARE:  ',E)
          TYPE 1254,RSD
          IF (IP) PRINT 1254,RSD
1254      FORMAT (' WTD RESIDUAL STANDARD ERROR:  ',E//)
          YB=0.
          DO 1260 I=1,M
1260      YB=YB+Y(I)
          YB=YB/M
          CTSS=0.
          DO 1270 I=1,M
1270      CTSS=CTSS+W(I)*(Y(I)-YB)**2
          R2=1.-RSS/CTSS
          TYPE 1272,R2
          IF (IP) PRINT 1272,R2
1272      FORMAT (' COEFFICIENT OF DETERMINATION',
     *     ' (R-SQUARE): ',E//)
C
C  CALCULATE STANDARDIZED RESIDUALS
C
1275      IF (N .EQ. 0) GOTO 1301
          DO 1300 I=1,M
             II=I
             CALL CALC (II,KG,N,MMAX,IV,C,ALF,T,AN)
             DO 1300 J=1,N
1300            BM(I,J)=PH(J)
1301      DO 1310 J=1,NPKG
             DO 1310 I=1,M
                SUM=0.
                DO 1315 K=1,NPKG
1315               SUM=SUM+BM(I,K)*A(K,J)
1310            XX(I,J)=SUM
          DO 1320 I=1,M
             SUM=0.
             DO 1325 K=1,NPKG
1325            SUM=SUM+XX(I,K)*BM(I,K)
             SYH(I)=DSQRT(DABS(SUM))
1320         SR(I)=R(I)/DSQRT(DABS(RMS/W(I)-SUM))
1321      IF (.NOT. ITB) GOTO 1344
C
C  PRINT TABLE OF OBS & PRED VALUES ALONG WITH STD RES
C
          ITB=.FALSE.
          TYPE 1330
          IF (IP) PRINT 1330
1330      FORMAT (/' INDEX',2X,'IND VAR 1',4X,'OBS VAL',
     *      5X,'PRED VAL',6X,'RESID',8X,'STD RES'/)
          DO 1340 I=1,M
             TYPE 1335,I,T(I,1),Y(I),ETA(I),R(I),SR(I)
             IF (IP) PRINT 1335,I,T(I,1),Y(I),ETA(I),
```

```
     *         R(I),SR(I)
1335      FORMAT (I4,1P5E13.5)
1340      CONTINUE
          IF (IV .EQ. 1) GOTO 1344
          TYPE 1342,(IND,VAR,J,J=2,IV)
          IF (IP) PRINT 1342,(IND,VAR,J,J=2,IV)
1342      FORMAT (/' INDEX',3X,5(2A4,I2,4X))
          DO 1344 I=1,M
          TYPE 1343,I,(T(I,J),J=2,IV)
          IF (IP) PRINT 1343,I,(T(I,J),J=2,IV)
1343      FORMAT (I3,3X,1P5E13.5)
1344      CONTINUE
          IF (.NOT. IPIV) GOTO 1350
C
C   PLOT STD RES VS INDEP VAR
C
          IPIV=.FALSE.
          IPI=1
          DO 1350 I=1,IV
             TYPE 1345,I
             IF (IP) PRINT 1345,I
1345         FORMAT (1H1,5X,'PLOT OF STANDARDIZED RESIDUAL',
     *          ' VS INDEPENDENT VARIABLE',I2//)
             CALL ORPLOT (T(1,I),M,IOR)
             CALL INTRVL (T(1,I),M,IOR,STP)
             CALL SETUP (T(1,I),M,Y,SR,IOR,STP)
1350      CONTINUE
          IF (.NOT. IPPV) GOTO 1360
C
C   PLOT STD RES VS PRED VAL
C
          IPPV=.FALSE.
          TYPE 1355
          IF (IP) PRINT 1355
1355      FORMAT (1H1,10X,'PLOT OF STANDARDIZED RESIDUAL VS',
     *       ' PREDICTED VALUES'//)
          CALL ORPLOT (ETA,M,IOR)
          CALL INTRVL (ETA,M,IOR,STP)
          CALL SETUP (ETA,M,Y,SR,IOR,STP)
1360      CONTINUE
          IF (.NOT. IPP) GOTO 1390
C
C   DO PROBABILITY PLOT OF STD RES
C
          IPP=.FALSE.
          IF (M .GT. 10) GOTO 1370
          DO 1365 I=1,M
             PP=(3.*I-1.)/(3.*M+1.)
1365         SYH(I)=4.91*(PP**(.14)-(1.-PP)**(.14))
          GOTO 1375
1370      PP=2./(3.*M+1.)
          SYH(1)=4.91*(PP**(.14)-(1.-PP)**(.14))
```

```
      PI8=PI/8
      PI4=PI/4
      DO 1375 I=2,M
         PP=(I-PI8)/(M+1.-PI4)
         SYH(I)=4.91*(PP**(.14)-(1.-PP)**(.14))
1375  CONTINUE
      CALL ORPLOT (SR,M,IOR)
      DO 1378 I=1,M
1378     SRV(I)=SR(IOR(I))
      TYPE 1380
      IF (IP) PRINT 1380
1380  FORMAT (1H1,10X,'PROBABILITY PLOT OF STANDARDIZED',
     *  ' RESIDUALS'//)
      CALL ORPLOT (SYH,M,IOR)
      CALL INTRVL (SYH,M,IOR,STP)
      CALL SETUP (SYH,M,Y,SRV,IOR,STP)
1390  RETURN
      END
```

```
      SUBROUTINE ORPLOT (T,M,IOR)
C
C  ARRANGE THE INDEPENDENT VARIABLES TO BE USED IN ASCENDING
C  ORDER.  THE INDEXING IS STORED IN ARRAY IOR.
C
      DIMENSION IOR (M)
      DOUBLE PRECISION T(M),SAV(100),HN,H
      DO 5 I=1,M
         SAV(I)=T(I)
5     CONTINUE
      H=SAV(1)
      IX=1
      DO 10 I=2,M
         HN=DMAX1(SAV(I),H)
         IF (HN .NE. H) IX=I
         H=HN
10    CONTINUE
      IOR(M)=IX
      MAX=IX
      MM1=M-1
      DO 50 J=1,MM1
      H=SAV(1)
      IX=1
         DO 20 I=2,M
            HN=DMIN1(SAV(I),H)
            IF (HN .NE. H) IX=I
         H=HN
20       CONTINUE
         IOR(J)=IX
         SAV(IX)=SAV(MAX)
50    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTRVL(T,M,IOR,STP)
C
C  DETERMINE THE LENGTH OF THE INTERVAL, STP, TO BE
C  USED BETWEEN SUCCESSIVE POINTS TO BE PLOTTED.
C
      DIMENSION IOR(M)
      DOUBLE PRECISION T(M),GN(100),AL,BH,STP
      MM1=M-1
      DO 10 I=1,MM1
         K=IOR(I)
         IP1=I+1
         KP1=IOR(IP1)
         GN(I)=T(KP1)-T(K)
10    CONTINUE
      BH=0.
      AL=1.E12
      DO 20 I=2,MM1
         IF (GN(I) .EQ. 0) GOTO 20
         AL=DMIN1(GN(I),AL)
         BH=DMAX1(GN(I),BH)
20    CONTINUE
      IF (AL .EQ. BH) GOTO 50
      KMN=IOR(1)
      KMX=IOR(M)
      GN(1)=T(KMX)-T(KMN)
      GN(2)=GN(1)/AL
      IF (GN(2) .LE. 200.) GOTO 50
40    STP=.5
      RETURN
50    STP=AL
      RETURN
      END
```

```
      SUBROUTINE SETUP (T,M,Y,ETA,IOR,STP)
C
C  SET UP ACTUAL ARRAYS TO BE PLOTTED.  INSERT ZEROS FOR
C  STEP POINTS WHOSE DEPENDENT VARIABLE IS NOT KNOWN.
C  CALL PLOT ROUTINE.
C
      DOUBLE PRECISION T(M),Y(M),ETA(M)
      DOUBLE PRECISION STP,Q,YG(200),YH(200),X(200)
      DIMENSION IOR(M)
      COMMON /IPIBLK/ IPI
      J=1
      K=IOR(J)
      X(1)=T(K)
      YG(1)=ETA(K)
      YH(1)=Y(K)
      DO 50 I=2,200
10       JP1=J+1
         KP1=IOR(JP1)
         IF (T(KP1) .NE. T(K)) GOTO 20
         J=JP1
         IF (J .EQ. M) GOTO 60
         GOTO 10
20       IM1=I-1
         X(I)=X(IM1)+STP
         IF((YG(IM1) .NE. 0.) .OR. (YH(IM1) .NE. 0.))J=J+1
         K=IOR(J)
         Q=(.5*STP)+X(I)
         IF (T(K) .GT. Q) GOTO 40
         YG(I)=ETA(K)
         YH(I)=Y(K)
         X(I)=T(K)
         IF (J .EQ. M) GOTO 60
         GOTO 50
40       CONTINUE
         YG(I)=0.
         YH(I)=0.
50    CONTINUE
60    NN=I
      IF (IPI .EQ. 2) GOTO 70
      DO 70 I=1,NN
         YH(I)=0.
70    CONTINUE
      CALL PLOT (X,YG,YH,NN)
      RETURN
      END
```

```
      SUBROUTINE PLOT (X,YG,YH,NN)
C
C  PLOT TWO DEPENDENT VARIABLES AGAINST INDEPENDENT VARIABLE.
C  NOTE:  PLOTS ARE ROTATED 90 DEGREES, WITH INDEPENDENT
C         VARIABLE AXIS APPEARING VERTICALLY (SMALLER VALUES
C         AT THE TOP).
C
      LOGICAL IP
      COMMON /BLOKIP/IP
      DOUBLE PRECISION X(NN),YG(NN),YH(NN),BH,AL,
     *   SCALE,YP1,YP2
      DIMENSION IY(56)
      DATA IBLANK /'     '/
      DATA IPLUS /'+    '/
      DATA ICROSS /'X    '/
      DATA ISTAR /'*    '/
      DATA IAXIS /'I    '/
C
C
      BH=0.
      AL=1.E12
      DO 40 I=1,NN
         AL=DMIN1(AL,YG(I),YH(I))
         BH=DMAX1(BH,YG(I),YH(I))
40    CONTINUE
      IF (AL .GE. 0.) GOTO 42
      IF (BH .LE. 0.) GOTO 45
C
C  AL<0 AND BH>0
C
      IZERO=IDINT(((DABS(AL))/(BH-AL))*54)+1
      SCALE=54./(BH-AL)
      GOTO 50
C
42    CONTINUE
C
C ALL POSITIVE VALUES
C
      IZERO=1
      SCALE=54./BH
      GOTO 50
C
45    CONTINUE
C
C  ALL VALUES NEGATIVE
C
      IZERO=55
      SCALE=54./(-AL)
C
50    CONTINUE
C
C   ENTER LOOP FOR PLOTTING
```

```
C
      DO 120 I=1,NN
         DO 51 J=1,56
            IY(J)=IBLANK
51       CONTINUE
         IY(IZERO)=IAXIS
         YP1=YG(I)
         YP2=YH(I)
         II1=IZERO+IDINT(YP1*SCALE)
         II2=IZERO+IDINT(YP2*SCALE)
         IF (II1 .NE. II2) GOTO 52
         IF (II1 .NE. IZERO) IY(II1)=ISTAR
         GOTO 55
52       CONTINUE
         IF (II1 .EQ. IZERO) GOTO 53
         IY(II1)=IPLUS
53       IF (II2 .EQ. IZERO) GOTO 55
         IY(II2)=ICROSS
55       CONTINUE
         TYPE 60,X(I),IY
60       FORMAT (' ',E12.5,3X,56A1)
         IF (IP) PRINT 60,X(I),IY
120   CONTINUE
      TYPE 140
140   FORMAT (////)
      IF (IP) PRINT 140
      RETURN
      END
```

```
      SUBROUTINE ARRAYD (BA,NRR,NCC,LVV,NAME)
C
C        PROGRAM AUTHORS  L. FELDMAN AND J DUNPHEY FORMERLY
C        WITH COMPUTING SCIENCES DIVISION, UNION CARBIDE,
C        NUCLEAR DIVISION, OAK RIDGE
C
C
C  ARRAYD IS AN OUTPUT SUBROUTINE WHICH PRINTS A
C  DOUBLE PRECISION MATRIX.  THE CALLING SEQUENCE
C  IS
C          CALL ARRAYD (A,NR,NC,ID,'....')
C  WHERE
C      A       IS A DOUBLE PRECISION ARRAY TO BE
C              LISTED.
C      NR      IS THE INTEGER NUMBER OF ROWS OF THE
C              MATRIX A DESIRED IN THE OUTPUT.
C      NC      IS THE INTEGER NUMBER OF COLUMNS OF THE
C              MATRIX A DESIRED IN THE OUTPUT.
C      ID      IS THE INTEGER FIRST DIMENSION OF A IN
C              THE CALLING PROGRAM, I.E. A(ID,*) OR A(ID)
C      '....' IS A FOUR-CHARACTER HOLLERITH ARGUMENT WHICH
C              WILL BE THE HEADING FOR THE MATRIX A.
C
      DOUBLE PRECISION COLUMN,BA(1)
         LOGICAL IP
         COMMON /BLOKIP/IP
      DATA COLUMN /8HCOLUMN  /
      DO 3 ML = 1,NCC,4
      MH = ML + 3
      IF (NCC .GE. MH) GO TO 1
      MH = NCC
    1 MLM1 = (ML-1)*LVV
      MHM1 = (MH-1)*LVV
      DO 3 LI = 1,NRR,50
      NL = LI
      NH = NL + 49
      IF (NRR .GE. NH) GO TO 2
      NH = NRR
    2 TYPE 100,     NAME
         IF (IP) PRINT 100,NAME
  100 FORMAT (A4)
      TYPE 200,      (COLUMN,J,J=ML,MH)
         IF (IP) PRINT 200,(COLUMN,J,J=ML,MH)
  200 FORMAT (4H0ROW,6X,4(A6,I5,4X))
      TYPE 300
         IF (IP) PRINT 300
  300 FORMAT (1H0)
      DO 3 L=NL,NH
      MML = MLM1 + L
      MMH = MHM1 + L
      TYPE 400,     L,(BA(J),J=MML,MMH,LVV)
3        IF (IP) PRINT 400,L,(BA(J),J=MML,MMH,LVV)
```

```
400 FORMAT (1H ,I3,3X,1P4D15.6)
    RETURN
    END
```

```
      SUBROUTINE VARPRO (N,KG,M,MMAX,NPP2,IV,T,Y,W,A,
     X IPRINT,ALF,C,IERR)
C
C          GIVEN A SET OF M OBSERVATIONS, CONSISTING OF
C       VALUES Y(1), Y(2),...,Y(M) OF A DEPENDENT VARIABLE
C       Y, WHERE Y(I) CORRESPONDS TO THE IV INDEPENDENT
C       VARIABLE(S) T(I,1), T(I,2),..., T(I,IV), VARPRO
C       ATTEMPTS TO COMPUTE A WEIGHTED LEAST SQUARES FIT TO
C       A FUNCTION ETA (THE 'MODEL') WHICH IS A LINEAR
C       COMBINATION
C                          N
C      ETA(C,ALF;T)   =  SUM C * PHI  (ALF;T) + PHI   (ALF;T)
C                         J=1  J      J              N+1
C
C       OF NONLINEAR FUNCTIONS PHI(J) (E.G. A SUM OF
C       EXPONENTIALS AND/OR GAUSSIANS),  THAT IS, DETERMINE
C       THE LINEAR PARAMETERS C(J) AND THE VECTOR OF
C       NONLINEAR PARAMETERS ALF BY MINIMIZING
C
C                      2     M                          2
C       NORM(RESIDUAL)   =  SUM W * (Y - ETA(C,ALF;T ))  .
C                           I=1  I    I              I
C
C       THE (N+1)-ST TERM IS OPTIONAL, AND IS USED WHEN IT
C       IS DESIRED TO FIX ONE OR MORE OF THE C'S (RATHER
C       THAN LET THEM BE DETERMINED).  VARPRO REQUIRES
C       FIRST DERIVATIVES OF THE PHI'S.
C
C                          NOTES:
C
C       A)  THE ABOVE PROBLEM IS ALSO REFERRED TO AS
C       'MULTIPLE NONLINEAR REGRESSION'.  FOR USE IN
C       STATISTICAL ESTIMATION, VARPRO RETURNS THE
C       RESIDUALS, THE COVARIANCE MATRIX OF THE LINEAR AND
C       NONLINEAR PARAMETERS, AND THE ESTIMATED VARIANCE
C       OF THE OBSERVATIONS.
C
C       B)  AN ETA OF THE ABOVE FORM IS CALLED 'SEPARABLE'.
C       THE CASE OF A NONSEPARABLE ETA CAN BE HANDLED BY
C       SETTING N = 0 AND USING PHI(N+1).
C
C       C)  VARPRO MAY ALSO BE USED TO SOLVE LINEAR LEAST
C       SQUARES PROBLEMS (IN THAT CASE NO ITERATIONS ARE
C       PERFORMED).  SET KG = 0.
C
C       D)  THE MAIN ADVANTAGE OF VARPRO OVER OTHER LEAST
C       SQUARES PROGRAMS IS THAT NO INITIAL GUESSES ARE
C       NEEDED FOR THE LINEAR PARAMETERS.  NOT ONLY DOES
C       THIS MAKE IT EASIER TO USE, BUT IT OFTEN LEADS
C       TO FASTER CONVERGENCE.
C
C
```

```
C   DESCRIPTION OF PARAMETERS
C
C         N         NUMBER OF LINEAR PARAMETERS C (MUST BE
C                   .GE. 0).
C         KG        NUMBER OF NONLINEAR PARAMETERS ALF (MUST
C                   BE .GE. 0).
C         M         NUMBER OF OBSERVATIONS, M MUST BE GREATER
C                   THAN KG + N (I.E., THE NUMBER OF OBSERVA-
C                   TIONS MUST EXCEED THE NUMBER OF PARAMETERS).
C         IV        NUMBER OF INDEPENDENT VARIABLES T.
C         T         REAL M BY IV ARRAY OF INDEPENDENT VARIABLES.
C                   T(I,J) CONTAINS THE VALUE OF THE I-TH
C                   OBSERVATION OF THE J-TH INDEPENDENT
C                   VARIABLE.
C         Y         M-VECTOR OF OBSERVATIONS, ONE FOR EACH
C                   ROW OF T.
C         W         M-VECTOR OF NONNEGATIVE WEIGHTS.  SHOULD
C                   BE SET TO 1'S IF WEIGHTS ARE NOT DESIRED.
C         INC       KG X (N+1) INTEGER INCIDENCE MATRIX.
C                   INC(K,J) = 1 IF NONLINEAR PARAMETER ALF(K)
C                   APPEARS IN THE J-TH FUNCTION PHI(J),
C                   OTHERWISE, INC(K,J) = 0.  IF PHI(N+1) IS
C                   NOT INCLUDED IN THE MODEL, THE (N+1)-ST
C                   COLUMN OF INC SHOULD BE SET TO ZERO.  IF
C                   PHI(N+1) IS INCLUDED, THE APPROPRIATE
C                   ELEMENTS OF THE (N+1)-ST COLUMN SHOULD BE
C                   SET TO 1'S.  INC IS NOT NEEDED WHEN N = 0 OR
C                   KG = 0.  CAUTION:  THE DECLARED ROW DIMENSION
C                   OF INC (IN ADA) MUST CURRENTLY BE SET TO 12.
C                   SEE 'RESTRICTIONS' BELOW.
C         MMAX      THE DECLARED ROW DIMENSION OF THE MATRICES
C                   A AND T.  IT MUST BE AT LEAST MAX(M,2*KG+3).
C         NPP2      N+P+2, WHERE P IS THE NUMBER OF ONES IN
C                   MATRIX INC..  THE DECLARED COLUMN
C                   DIMENSION OF A MUST BE AT LEAST NPP2.
C                   (IF N = 0, SET NPP2 = KG+2.  IF KG = 0,
C                   SET NPP2 = N+2.)
C         A         REAL MATRIX OF SIZE MAX(M,2*KG+3) BY N+P+2.
C                   ON INPUT IT CONTAINS THE PHI(J)'S AND
C                   THEIR DERIVATIVES (SEE BELOW).  ON OUTPUT
C                   THE FIRST N+KG ROWS AND COLUMNS OF A WILL
C                   CONTAIN AN APPROXIMATION TO THE (WEIGHTED)
C                   COVARIANCE MATRIX AT THE SOLUTION (THE
C                   FIRST N ROWS CORRESPOND TO THE LINEAR
C                   PARAMETERS, THE LAST KG TO THE NONLINEAR
C                   ONES), COLUMN N+KG+1 WILL CONTAIN THE
C                   WEIGHTED RESIDUALS (Y - ETA), A(1,N+KG+2)
C                   WILL CONTAIN THE (EUCLIDEAN) NORM OF THE
C                   WEIGHTED RESIDUAL, AND A(2,N+KG+2) WILL
C                   CONTAIN AN ESTIMATE OF THE (WEIGHTED)
C                   VARIANCE OF THE OBSERVATIONS,
C                   NORM(RESIDUAL)**2/(M-N-KG).
```

```
C         IPRINT  INPUT INTEGER CONTROLLING PRINTED OUTPUT.
C                 IF IPRINT IS POSITIVE, THE NONLINEAR
C                 PARAMETERS, THE NORM OF THE RESIDUAL, AND
C                 THE MARQUARDT PARAMETER WILL BE OUTPUT
C                 EVERY IPRINT-TH ITERATION (AND INITIALLY,
C                 AND AT THE FINAL ITERATION).  THE LINEAR
C                 PARAMETERS WILL BE PRINTED AT THE FINAL
C                 ITERATION.  ANY ERROR MESSAGES WILL ALSO
C                 BE PRINTED.  (IPRINT = 1 IS RECOMMENDED
C                 AT FIRST.)  IF IPRINT = 0, ONLY THE FINAL
C                 QUANTITIES WILL BE PRINTED, AS WELL AS ANY
C                 ERROR MESSAGES.  IF IPRINT = -1, NO
C                 PRINTING WILL BE DONE.  THE USER IS THEN
C                 RESPONSIBLE FOR CHECKING THE PARAMETER
C                 IERR FOR ERRORS.
C         ALF     KG-VECTOR OF ESTIMATES OF NONLINEAR
C                 PARAMETERS (INPUT).  ON OUTPUT IT WILL
C                 CONTAIN OPTIMAL VALUES OF THE NONLINEAR
C                 PARAMETERS.
C         C       N-VECTOR OF LINEAR PARAMETERS (OUTPUT ONLY).
C         IERR    INTEGER ERROR FLAG (OUTPUT):
C                 .GT. 0 - SUCCESSFUL CONVERGENCE. IERR IS
C                     THE NUMBER OF ITERATIONS TAKEN.
C                 -1  TERMINATED FOR TOO MANY ITERATIONS.
C                 -2  TERMINATED FOR ILL-CONDITIONING
C                     (MARQUARDT PARAMETER TOO LARGE.) ALSO
C                     SEE IERR = -8 BELOW.
C                 -4  INPUT ERROR IN PARAMETER N, M, KG,
C                     NPP2 OR MMAX.
C                 -5  INC MATRIX IMPROPERLY SPECIFIED, OR
C                     P DISAGREES WITH NPP2.
C                 -6  WEIGHTS NOT NONNEGATVE.
C                 -7  'CONSTANT' COLUMN WAS COMPUTED MORE
C                     THAN ONCE.
C                 -8  CATASTROPHIC FAILURE - A COLUMN OF THE
C                     A MATRIX HAS BECOME ZERO.  SEE
C                     'CONVERGENCE FAILURES' BELOW.
C
C                 (IF IERR .LE. -4, THE LINEAR PARAMETERS,
C                 COVARIANCE MATRIX, ETC. ARE NOT RETURNED.)
C
C   SUBROUTINES REQUIRED
C
C          SEVEN SUBROUTINES, DPA, ORFAC1, ORFAC2, BACSUB,
C       POSTPR, COV AND VARERR ARE PROVIDED.  IN ADDITION,
C       THE USER MUST PROVIDE A SUBROUTINE (CORRESPONDING
C       TO THE ARGUMENT ADA) WHICH, GIVEN ALF, WILL EVALUATE
C       THE FUNCTIONS PHI(J) AND THEIR PARTIAL DERIVATIVES
C       D PHI(J)/D ALF(K), AT THE SAMPLE POINTS T(I).
C       ITS CALLIING SEQUENCE IS
C
C       SUBROUTINE ADA (N+1 KG,M,MMAX,NPP2,IV,A,INC,T,ALF,
```

```
C       ISEL)
C
C          THE USER SHOULD MODIFY THE EXAMPLE SUBROUTINE
C       'ADA' (GIVEN ELSEWHERE) FOR HIS OWN FUNCTIONS.
C
C          THE VECTOR SAMPLED FUNCTIONS PHI(J) SHOULD BE
C       STORED IN THE FIRST M ROWS AND FIRST N+1 COLUMNS OF
C       THE MATRIX A, I.E., A(I, J) SHOULD CONTAIN
C       T(PHI(J, ALF; T(I,1), T(I,2),...,I,IV)),
C       I = 1,..., M; J = 1, ..., N (OR N+1). THE (N+1)-ST
C       COLUMN OF A CONTAINS PHI(N+1) IF PHI(N+1) IS IN THE
C       MODEL, OTHERWISE IT IS RESERVED FOR WORKSPACE.
C       THE 'CONSTANT FUNCTIONS (THESE ARE FUNCTIONS PHI(J)
C       WHICH DO NOT DEPEND UPON ANY NONLINEAR PARAMETERS
C       ALF, E.G., T(I)**J) (IF ANY) MUST APPEAR FIRST,
C       STARTING IN COLUMN 1. THE COLUMN N-VECTORS OF
C       NONZERO PARTIAL DERIVATIVES D PHI(J) / D ALF(K)
C       SHOULD BE STORED SEQUENTIALLY IN THE MATRIX A IN
C       COLUMNS N+2 THROUGH N+P+1. THE ORDER IS
C
C         D PHI(1)  D PHI(2)          D PHI(N)  D PHI(N+1)
C         --------, --------, ...,   --------, ----------,
C         D ALF(1)  D ALF(1)          D ALF(1)    D ALF(1)
C
C         D PHI(1)         D PHI(2)          D PHI(N+1)
C         --------, ...,   ----------, ...,   ---------, ...,
C         D ALF(2)          D ALF(2)          D ALF(2)
C
C         D PHI(1)          D PHI(N+1)
C         ---------, ...,   ----------,
C         D ALF(KG)         D ALF(KG)
C
C       OMITTING COLUMNS OF DERIVATIVES WHICH ARE ZERO, AND
C       OMITTING PHI(N+1) COLUMNS IF PHI(N+1) IS NOT IN THE
C       MODEL. NOTE THAT THE LINEAR PARAMETERS C ARE NOT
C       USED IN THE MATRIX A. COLUMN N+P+2 IS RESERVED
C       FOR WORKSPACE.
C
C       THE CODING OF ADA SHOULD BE ARRANGED SO THAT:
C
C       ISEL = 1  (WHICH OCCURS THE FIRST TIME ADA IS
C                 CALLED) MEANS:
C                 A.  FILL IN THE INCIDENCE MATRIX INC.
C                 B.  STORE ANY CONSTANT PHI'S IN A.
C                 C.  COMPUTE NONCONSTANT PHI'S AND PARTIAL
C                     DERIVATIVES.
C            = 2  MEANS COMPUTE ONLY THE NONCONSTANT
C                 FUNCTIONS PHI.
C            = 3  MEANS COMPUTE ONLY THE DERIVATIVES.
C
C       (WHEN THE PROBLEM IS LINEAR (KG = 0) ONLY ISEL = 1
C       IS USED, AND DERIVATIVES ARE NOT NEEDED.)
```

```
C
C     RESTRICTIONS
C
C          THE SUBROUTINES DPA (AND ADA) CONTAIN THE LOCALLY
C        DIMENSIONED MATRIX INC, WHOSE DIMENSIONS ARE
C        CURRENTLY SET FOR MAXIMA OF N+1 = 8, KG = 12.
C        THEY MUST BE CHANGED FOR LARGER PROBLEMS.
C        DATA PLACED IN ARRAY A IS OVERWRITTEN ('DESTROYED').
C        DATA PLACED IN ARRAYS T, Y AND INC IS LEFT INTACT.
C        THE PROGRAM RUNS IN WATFIV, EXCEPT WHEN N = 0
C        OR KG = 0.
C
C          IT IS ASSUMED THAT THE MATRIX PHI(J, ALF; T(I))
C        HAS FULL COLUMN RANK.  THIS MEANS THAT THE FIRST N
C        COLUMNS OF THE MATRIX A MUST BE LINEARLY
C        INDEPENDENT.
C
C          OPTIONAL NOTE:  AS WILL BE NOTED FROM THE SAMPLE
C        SUBPROGRAM ADA, THE DERIVATIVES D PHI(J)/D ALF(K)
C        (ISEL = 3) MUST BE COMPUTED INDEPENDENTLY OF THE
C        FUNCTIONS PHI(J) (ISEL = 2), SINCE THE FUNCTION
C        VALUES ARE OVERWRITTEN AFTER ADA IS CALLED
C        WITH ISEL = 2.  THIS IS DONE TO MINIMIZE STORAGE,
C        AT THE POSSSIBLE EXPENSE OF SOME RECOMPUTATION
C        (SINCE THE FUNCTIONS AND DERIVATIVES FREQUENTLY
C        HAVE SOME COMMON SUBEXPRESSIONS).  TO REDUCE THE
C        AMOUNT OF COMPUTATION AT THE EXPENSE OF SOME
C        STORAGE, CREATE AN ARRAY B OF DIMENSION MMAX BY
C        N+1 IN ADA, AND AFTER THE COMPUTATION OF THE PHI'S
C        (ISEL = 2), COPY THE VALUES INTO B.  THESE VALUES
C        CAN THEN BE USED TO CALCULATE THE DERIVATIVES
C        (ISEL = 3).  (THIS MAKES USE OF THE FACT THAT WHEN
C        A CALL TO ADA WITH ISEL = 3 FOLLOWS A CALL WITH
C        ISEL = 2, THE ALFS ARE THE SAME.)
C
C          TO CONVERT TO OTHER MACHINES, CHANGE THE OUTPUT
C        UNIT IN THE DATA STATEMENTS IN VARPRO, DPA, POSTPR,
C        AND VARERR.  THE PROGRAM HAS BEEN CHECKED FOR
C        PORTABILITY BY THE BELL LABS PFORT VERIFIER.  FOR
C        MACHINES WITHOUT DOUBLE PRECISION HARDWARE, IT
C        MAY BE DESIRABLE TO CONVERT TO SINGLE PRECISION.
C        THIS CAN BE DONE BY CHANGING (A) THE DECLARATIONS
C        'DOUBLE PRECISION' TO 'REAL', (B) THE PATTERN '.D'
C        TO '.E' IN THE 'DATA' STATEMENT IN VARPRO, AND (C)
C        DSIGN AND DSQRT TO SIGN AND SQRT, RESPECTIVELY.
C
C     CONVERGENCE FAILURES
C
C          IF CONVERGENCE FAILURES OCCUR, FIRST CHECK FOR
C        INCORRECT CODING OF THE SUBROUTINE ADA.  CHECK
C        ESPECIALLY THE ACTION OF ISEL, AND THE COMPUTATION
C        OF THE PARTIAL DERIVATIVES.  IF THESE ARE CORRECT,
```

```
C        TRY SEVERAL STARTING GUESSES FOR ALF.  IF ADA
C        IS CODED CORRECTLY, AND IF ERROR RETURNS IERR = -2
C        OR -8 PERSISTENTLY OCCUR, THIS IS A SIGN OF ILL-
C        CONDITIONING, WHICH MAY BE CAUSED BY SEVERAL THINGS.
C        ONE IS POOR SCALING OF THE PARAMETERS; ANOTHER IS
C        AN UNFORTUNATE INITIAL GUESS FOR THE PARAMETERS,
C        STILL ANOTHER IS A POOR CHOICE OF THE MODEL.
C
C     ALGORITHM
C
C           THE RESIDUAL R IS MODIFIED TO INCORPORATE, FOR
C        ANY FIXED ALF, THE OPTIMAL LINEAR PARAMETERS FOR
C        THAT ALF.  IT IS THEN POSSIBLE TO MINIMIZE ONLY ON
C        THE NONLINEAR PARAMETERS.  AFTER THE OPTIMAL VALUES
C        OF THE NONLINEAR PARAMETERS HAVE BEEN DETERMINED,
C        THE LINEAR PARAMETERS CAN BE RECOVERED BY LINEAR
C        LEAST SQUARES TECHNIQUES (SEE REF. 1).
C
C           THE MINIMIZATION IS BY A MODIFICATION OF
C        OSBORNE'S (REF. 3) MODIFICATION OF THE LEVENBERG-
C        MARQUARDT ALGORITHM.  INSTEAD OF SOLVING THE NORMAL
C        EQUATIONS WITH MATRIX
C
C                 T      2
C              (J J + NU  * D),     WHERE  J = D(ETA)/D(ALF),
C
C        STABLE ORTHOGONAL (HOUSEHOLDER) REFLECTIONS ARE
C        USED ON A MODIFICATION OF THE MATRIX
C
C                                  (   J   )
C                                  (-------) ,
C                                  ( NU*D  )
C
C        WHERE D IS A DIAGONAL MATRIX CONSISTING OF THE
C        LENGTHS OF THE COLUMNS OF J.  THIS MARQUARDT
C        STABILIZATION ALLOWS THE ROUTINE TO RECOVER FROM
C        SOME RANK DEFICIENCIES IN THE JACOBIAN.  OSBORNE'S
C        EMPIRICAL STRATEGY FOR CHOOSING THE MARQUARDT
C        PARAMETER HAS PROVEN REASONABLY SUCCESSFUL IN
C        PRACTICE.  (GAUSS-NEWTON WITH STEP CONTROL CAN BE
C        OBTAINED BY MAKING THE CHANGE INDICATED BEFORE THE
C        INSTRUCTION LABELED 1).  A DESCRIPTION CAN BE FOUND
C        IN REF. (3), AND A FLOW CHART IN (2), P. 22.
C
C        FOR REFERENCE, SEE
C
C        1.  GENE H. GOLUB AND V. PEREYRA, 'THE DIFFEREN-
C            TIATION OF PSEUDO-INVERSES AND NONLINEAR LEAST
C            SQUARES PROBLEMS WHOSE VARIABLES SEPARATE,'
C            SIAM J. NUMER. ANAL. 10, 413-432 (1973).
C        2.  ------, SAME TITLE, STANFORD C.S. REPORT 72-261,
C            FEB. 1972.
```

```
C         3.  OSBORNE, MICHAEL R., 'SOME ASPECTS OF NON-
C             LINEAR LEAST SQUARES CALCULATIONS,' IN LOOTSMA,
C             ED., 'NUMERICAL METHODS FOR NON-LINEAR
C             OPTIMIZATION,' ACADEMIC PRESS, LONDON, 1972.
C         4.  KROGH, FRED, 'EFFICIENT IMPLEMENTATION OF A
C             VARIABLE PROJECTION ALGORITHM FOR NONLINEAR
C             LEAST SQUARES PROBLEMS,' COMM. ACM 17, PP. 167-
C             169 (MARCH, 1974).
C         5.  KAUFMAN, LINDA, 'A VARIABLE PROJECTION METHOD
C             FOR SOLVING SEPARABLE NONLINEAR LEAST SQUARES
C             PROBLEMS', B.I.T. 15,49-57 (1975).
C
C                   JOHN BOLSTAD
C                   COMPUTER SCIENCE DEPT., SERRA HOUSE
C                   STANFORD UNIVERSITY
C                   SEPTEMBER, 1976
C
C     ..................................................
C
      DOUBLE PRECISION A(MMAX,NPP2),C(N),ALF(KG),T(MMAX,IV),
     2 W(M),Y(M),ACUM,ALFNRM,DALFNM,DELR,DSQRT,EPS1,EPS2,
     3 GNSTEP,NU,RNORM,RNEW,PRJRES
      INTEGER B1, OUTPUT
      LOGICAL SKIP,IP
      COMMON /BLOKIP/ IP
      DATA ITMAX /50/, EPS1 /1.D-6/, EPS2 /1.D-6/
      DATA OUTPUT /5/
C
C        THE FOLLOWING THREE PARAMETERS ARE USED IN
C        THE CONVERGENCE TEST:  EPS1 IS AN ABSOLUTE
C        TOLERANCE FOR THE NORM OF THE PROJECTION OF
C        THE RESIDUAL ONTO THE RANGE OF THE JACOBIAN
C        OF THE VARIABLE PROJECTION FUNCTIONAL.  EPS2
C        IS A RELATIVE AND ABSOLUTE TOLERANCE FOR THE
C        CHANGE IN THE NORM OF THE RESIDUAL.  ITMAX IS
C        THE MAXIMUM NUMBER OF FUNCTION AND DERIVATIVE
C        EVALUATIONS ALLOWED.  CAUTION:  EPS1 AND
C        EPS2 MUST NOT BE SET SMALLER THAN 10 TIMES
C        THE UNIT ROUND-OFF OF THE MACHINE.
C
      IERR = 1
      ITER = 0
      NP1 = N + 1
      B1 = N + 2
      NKG2 = N + KG + 2
      KGP1 = KG + 1
      KG23 = 2*KG + 3
      SKIP = .FALSE.
      MODIT = IPRINT
      IF (IPRINT .LE. 0) MODIT = ITMAX + 2
      NU = 0.
C
```

```
C         IF GAUSS-NEWTON IS DESIRED, REMOVE THE
C         NEXT STATEMENT.
C
      NU = 1.
C
C         CHECK FOR VALID INPUT
C
      IF (N .GE. 0 .AND. KG .GE. 0 .AND. N+KG .LT. M
     X  .AND. NKG2 .LE. NPP2 .AND. KG23 .LE. MMAX
     X  .AND. M .LE. MMAX .AND. IV .GT. 0
     X .AND. .NOT. (KG .EQ. 0 .AND. N .EQ. 0)) GOTO 10
       IERR = -4
       CALL VARERR (IPRINT, IERR, 1)
       GO TO 99
C
C                 BEGIN OUTER ITERATION LOOP TO UPDATE ALF.
C                 CALCULATE THE NORM OF THE RESIDUAL AND THE
C                 DERIVATIVE OF THE MODIFIED RESIDUAL THE
C                 FIRST TIME, BUT ONLY THE DERIVATIVE IN
C                 SUBSEQUENT ITERATIONS.
C
   10 CALL DPA (N,KG,M,MMAX,NPP2,IV,T,Y,W,ALF,IERR,
     X IPRINT,A,C,A(1,NP1),RNORM)
         GNSTEP = 1.0
         ITERIN = 0
         IF (ITER .GT. 0) GO TO 5
            IF (KG .EQ. 0) GO TO 90
            IF (IERR .NE. 1) GO TO 99
C
            IF (IPRINT .LE. 0) GO TO 5
            WRITE (OUTPUT,207) ITERIN, RNORM
            IF (IP) PRINT 207, ITERIN, RNORM
            WRITE (OUTPUT,200) NU
            IF (IP) PRINT 200, NU
C                 BEGIN TWO-STAGE ORTHOGONAL FACTORIZATION
    5 CALL ORFAC1(KGP1,MMAX,M,N,IPRINT,A(1,B1),PRJRES,IERR)
      IF (IERR .LT. 0) GOTO 99
      IERR = 2
      IF (NU .EQ. 0.) GO TO 30
C
C                   BEGIN INNER ITERATION LOOP FOR GENERATING
C                   NEW ALF AND TESTING IT FOR ACCEPTANCE.
C
   25   CALL ORFAC2(KGP1,MMAX,NU,A(1,B1))
C
C                   SOLVE A KG X KG UPPER TRIANGULAR SYSTEM
C                   FOR DELTA-ALF.  THE TRANSFORMED RESIDUAL
C                   (IN COL. NKG2 OF A) IS OVERWRITTEN BY
C                   THE RESULT DELTA-ALF.
C
   30   CALL BACSUB (MMAX,KG,A(1,B1),A(1,NKG2))
         DO 35 K=1,KG
```

```
   35       A(K,B1) = ALF(K) + A(K,NKG2)
C
C           NEW ALF(K) = ALF(K) + DELTA-ALF(K)
C
C                STEP TO THE NEW POINT NEW ALF, AND COMPUTE
C                THE NEW NORM OF THE RESIDUAL.  NEW ALF
C                IS STORED IN COLUMN B1 OF A.
C
   40     CALL DPA (N,KG,M,MMAX,NPP2,IV,T,Y,W,A(1,B1),
     X        IERR,IPRINT,A,C,A(1,NP1),RNEW)
          IF (IERR .NE. 2) GO TO 99
           ITER = ITER + 1
           ITERIN = ITERIN + 1
           SKIP = MOD(ITER, MODIT) .NE. 0
           IF (SKIP) GO TO 45
           WRITE (OUTPUT,203) ITER
           IF (IP) PRINT 203, ITER
           WRITE (OUTPUT,216) (A(K,B1),K = 1,KG)
           IF (IP) PRINT 216,(A(K,B1),K=1,KG)
           WRITE (OUTPUT,207) ITERIN, RNEW
           IF (IP) PRINT 207,ITERIN,RNEW
C
   45     IF (ITER .LT. ITMAX) GO TO 50
           IERR= -1
           CALL VARERR (IPRINT, IERR, 1)
           GO TO 95
   50     IF (RNEW .LT. RNORM) GO TO 75
C
C                RETRACT THE STEP JUST TAKEN
C
           IF (NU .NE. 0) GO TO 60
C               GAUSS-NEWTON OPTION ONLY
           GNSTEP = 0.5*GNSTEP
           IF (GNSTEP .LT. EPS1) GO TO 95
           DO 55 K = 1, KG
   55         A(K, B1) = ALF(K) + GNSTEP*A(K,NKG2)
           GO TO 40
C                ENLARGE THE MARQUARDT PARAMETER
   60     NU = 1.5*NU
           IF (.NOT. SKIP) WRITE (OUTPUT,206) NU
           IF ((.NOT. SKIP) .AND. IP) PRINT 206,NU
           IF (NU .LE. 100) GO TO 65
           IERR = -2
           CALL VARERR (IPRINT, IERR, 1)
           GO TO 95
C                RETRIEVE UPPER TRIANGULAR FORM
C                AND RESIDUAL OF FIRST STAGE.
   65     DO 70 K = 1,KG
              KSUB = NP1 + K
              DO 70 J = K, KGP1
                 JSUB = NP1 + J
                 ISUB = KGP1 + J
```

```
   70          A(K, JSUB) = A(ISUB, KSUB)
       GO TO 25
C
C                END OF INNER ITERATION LOOP
C
C       ACCEPT THE STEP JUST TAKEN
C
   75 DELR = RNORM - RNEW
      RNORM = RNEW
      ALFNRM = 0.
      DALFNM = 0.
      DO 80 K = 1, KG
          ALF(K) = A(K, B1)
          ALFNRM = ALFNRM + ALF(K)**2
   80     DALFNM = DALFNM + A(K, NKG2)**2
      ALFNRM = DSQRT(ALFNRM)
      DALFNM = GNSTEP*DSQRT(DALFNM)
      ACUM = DALFNM/ALFNRM
C
C           IF ITERIN IS GREATER THAN 1, A STEP WAS
C           RETRACTED DURING THIS OUTER ITERATION.
C
      IF (ITERIN .EQ. 1) NU = 0.5*NU
      IF (SKIP) GO TO 85
      WRITE (OUTPUT, 200) NU
         IF (IP) PRINT 200,NU
      WRITE (OUTPUT, 208) ACUM
         IF (IP) PRINT 208,ACUM
   85 IERR = 3
      IF (PRJRES .GT. EPS1 .OR. DELR .GT. (RNORM + 1.D0)*EPS2)
     X   GO TO 10
C           END OF OUTER ITERATION LOOP
C
C           CALCULATE FINAL QUANTITIES -- LINEAR PARAMETERS,
C           RESIDUALS, COVARIANCE MATRIX, ETC.
C
   90 IERR = ITER
   95 IF (KG .GT. 0) CALL DPA(N,KG,M,MMAX,NPP2,IV,T,Y,W,ALF,
     X  4,IPRINT,A,C,A(1,NP1),RNORM)
       CALL POSTPR(N,KG,M,MMAX,NKG2,RNORM,IPRINT,ALF,W,A,
     X A(1,NP1),C,IERR)
   99 RETURN
C
  200 FORMAT (9H      NU =, E15.7)
  203 FORMAT (12H0  ITERATION, I4, 14H    NONLINEAR ,
     X   10HPARAMETERS)
  206 FORMAT (25H      STEP RETRACTED, NU =, E15.7)
  207 FORMAT (1H0, I5, 20H  NORM OF RESIDUAL =, E15.7)
  208 FORMAT (34H      NORM(DELTA-ALF) / NORM(ALF) =, E12.3)
  216 FORMAT (1H0, 7E15.7)
      END
C
```

```
      SUBROUTINE ORFAC1 (KGP1,MMAX,M,N,IPRINT,B,PRJRES,IERR)
C
C             STAGE 1:  HOUSEHOLDER REDUCTION OF
C
C                       (    .    )       ( DR'. R3 )     KG
C                       ( DR . R2 )  TO   (----. -- ),
C                       (    .    )       (  0 . R4 )   M-N-KG
C
C                         KG    1           KG    1
C
C          WHERE DR = -D(Q2)*Y IS THE DERIVATIVE OF THE MODI-
C          FIED RESIDUAL PRODUCED BY DPA, R2 IS THE TRANS-
C          FORMED RESIDUAL FROM DPA, AND DR' IS IN UPPER
C          TRIANGULAR FORM (AS IN REF. (2), P. 18).
C          DR IS STORED IN ROWS N+1 TO M AND COLUMNS N+2 TO
C          N + KG + 1 OF THE MATRIX A (I.E.,COLUMNS 1 TO KG
C          OF THE MATRIX B).  R2 IS STORED IN COLUMN
C          N + KG + 2 OF THE MATRIX A (COLUMN KG + 1 OF B).
C          FOR K = 1, 2, ..., KG, FIND REFLECTION
C          I - U * U' / BETA WHICH ZEROES B(I, K),
C          I = N+K+1, ..., M.
C
C     ..................................................
C
      DOUBLE PRECISION ACUM,ALPHA,B(MMAX,KGP1),BETA,DSIGN,
     X DSQRT,U1,PRJRES
C
      KG = KGP1 - 1
      KG23 = 2*KG + 3
      NP1 = N + 1
      NPKG = N + KG
C
      DO 30 K = 1, KG
         ALPHA = 0.0
         NPK = N + K
         DO 11 I = NPK, M
   11       ALPHA = ALPHA + B(I, K) ** 2
         ALPHA = DSIGN(DSQRT(ALPHA), B(NPK, K))
         U1 = B(NPK, K) + ALPHA
         B(NPK, K) = U1
         BETA = ALPHA * U1
         IF (ALPHA .NE. 0.0) GO TO 13
C                              COLUMN WAS ZERO
         IERR = -8
         CALL VARERR (IPRINT, IERR, NP1 + K)
         GO TO 99
C                       APPLY REFLECTIONS TO REMAINING COLUMNS
C                       OF B AND TO RESIDUAL VECTOR.
   13    KP1 = K + 1
         DO 25 J = KP1, KGP1
            ACUM = 0.0
            DO 20 I = NPK, M
```

```
   20          ACUM = ACUM + B(I, K) * B(I, J)
               ACUM = ACUM / BETA
               DO 25 I = NPK, M
   25          B(I, J) = B(I, J) - B(I, K) * ACUM
   30       B(NPK, K) = -ALPHA
C
         PRJRES = 0.
         DO 35 J = NP1, NPKG
   35       PRJRES = PRJRES + B(J, KGP1)**2
         PRJRES = DSQRT(PRJRES)
C
C              SAVE UPPER TRIANGULAR FORM AND TRANSFORMED
C              RESIDUAL, FOR USE IN CASE A STEP IS RETRACTED.
C              ALSO COMPUTE COLUMN LENGTHS.
C
         IF (IERR .EQ. 4) GO TO 99
         DO 40 K = 1, KG
            NPK = N + K
            ACUM = 0.
            DO 37 J = NP1, NPK
   37          ACUM = ACUM + B(J, K) **2
            B(KG23, K) = DSQRT(ACUM)
            DO 40 J = K, KGP1
               JSUB = KGP1 + J
               B(K, J) = B(NPK, J)
   40          B(JSUB, K) = B(NPK, J)
C
   99 RETURN
      END
```

```
      SUBROUTINE ORFAC2 (KGP1,MMAX,NU,B)
C
C        STAGE 2:  SPECIAL HOUSEHOLDER REDUCTION OF
C
C             KG      ( DR' . R3 )       (DR'' . R5 )
C                     (-----. -- )       (-----. -- )
C        M-N-KG       (  0  . R4 )  TO   (  0  . R4 )
C                     (-----. -- )       (-----. -- )
C             KG      (NU*D . 0  )       (  0  . R6 )
C
C                        KG    1            KG    1
C
C        WHERE DR', R3, AND R4 ARE AS IN ORFAC1, NU IS THE
C        MARQUARDT PARAMETER, D IS A DIAGONAL MATRIX
C        CONSISTING OF THE LENGTHS OF THE COLUMNS OF DR',
C        AND DR'' IS IN UPPER TRIANGULAR FORM.  DETAILS IN
C        (1), PP. 423-424.  NOTE THAT THE (M-N-KG) BAND OF
C        ZEROES, AND R4, ARE OMITTED IN STORAGE.
C
C     ..................................................
C
      DOUBLE PRECISION ACUM,ALPHA,B(MMAX,KGP1),BETA,DSIGN,
     X DSQRT,NU,U1
C
      KG = KGP1 - 1
      KG2 = 2*KG
      KG23 = KG2 + 3
      DO 30 K = 1, KG
         KP1 = K + 1
         KGPK = KG + K
         KGPKM1 = KGPK - 1
         B(KGPK, K) = NU * B(KG23, K)
         ALPHA = B(K, K)**2
         DO 10 I = KGP1, KGPK
   10       ALPHA = ALPHA + B(I, K)**2
         ALPHA = DSIGN(DSQRT(ALPHA), B(K, K))
         U1 = B(K, K) + ALPHA
         BETA = ALPHA * U1
         B(K, K) = -ALPHA
C              THE K-TH REFLECTION MODIFIES ONLY ROWS K,
C              KG+1, KG+2, ..., KG+K, AND COLUMNS K TO KG+1.
         DO 30 J = KP1, KGP1
            B(KGPK, J) = 0.
            ACUM = U1 * B(K,J)
            DO 20 I = KGP1, KGPKM1
   20          ACUM = ACUM + B(I,K) * B(I,J)
            ACUM = ACUM / BETA
            B(K,J) = B(K,J) - U1 * ACUM
            DO 30 I = KGP1, KGPK
   30          B(I,J) = B(I,J) - B(I,K) * ACUM
C
      RETURN
      END
```

```
      SUBROUTINE DPA (N,KG,M,MMAX,NPP2,IV,T,Y,W,ALF,ISEL,
     X IPRINT,A,U,R,RNORM)
C
C        COMPUTE THE NORM OF THE RESIDUAL (IF ISEL = 1 OR 2),
C        OR THE (M-N) X KG DERIVATIVE OF THE MODIFIED
C        RESIDUAL (M-N) VECTOR Q2*Y (IF ISEL = 1 OR 3).
C        HERE Q * PHI = S, I.E.,
C
C     N      ( Q1 ) (     .   .          )   ( S  . R1 .  F1  )
C            (----) ( PHI . Y . D(PHI) ) = (--- . -- . ---- )
C     M-N    ( Q2 ) (     .   .          )   ( 0  . R2 .  F2  )
C
C              M        N    1      P           N     1     P
C
C        WHERE Q IS M X M ORTHOGONAL, AND S IS N X N UPPER
C        TRIANGULAR.  THE NORM OF THE RESIDUAL = NORM(R2),
C        AND THE DESIRED DERIVATIVE ACCORDING TO REF.(5), IS
C
C                                                    -1
C                  D(Q2 * Y) = -Q2 * D(PHI)* S   * Q1* Y.
C
C     ..................................................
C
      COMMON /BLKINC/ INC
      DOUBLE PRECISION A(MMAX,NPP2),ALF(KG),T(MMAX,IV),W(M),
     X Y(M),ACUM,ALPHA,BETA,RNORM,DSIGN,DSQRT,SAVE,R(M),U(N)
      INTEGER FIRSTC,FIRSTR,P,INC(12,8)
      LOGICAL PHINP1
C
      NP1 = N + 1
      CALL ADA (NP1,N,KG,M,MMAX,NPP2,IV,A,T,ALF,MIN0(ISEL,
     X 3),PHINP1,NCONP1)
      GO TO (5, 4, 3, 3),ISEL
C                                          ISEL = 3 OR 4
    3 FIRSTC = NP2
      LASTC = NPP1
      FIRSTR = (4 - ISEL)*N + 1
      GO TO 50
C                                          ISEL = 2
    4 FIRSTC = NCONP1
      LASTC = NP1
      IF (NCON .EQ. 0) GO TO 30
      IF (A(1, NCON) .EQ. SAVE) GO TO 30
         ISEL =-7
         CALL VARERR (IPRINT,ISEL,NCON)
         GOTO 99
C                                  ISEL = 1 (FIRST TIME ONLY)
    5 NP2 = N + 2
      NKG2 = N + 2 + KG
      NPP1 = NPP2 - 1
      FIRSTC = 1
      LASTC = NPP1
      FIRSTR = NP1
```

```
      DO 8 I = 1, M
         IF (W(I) .GE. 0.) GO TO 8
C                                              ERROR IN WEIGHTS
         ISEL = -6
          II=I
         CALL VARERR (IPRINT, ISEL, II)
         GO TO 99
    8    W(I) = DSQRT(W(I))
C
      NCON = N
      NCONP1 = NP1
      IF (KG .EQ. 0) GO TO 32
C
      IF (N .NE. 0) GO TO 10
      PHINP1 = .TRUE.
      GO TO 40
C                              DETERMINE NUMBER OF CONSTANT FCNS.
   10 P = 0
      DO 11 J = 1, NP1
         IF (P .EQ. 0) NCONP1 = J
         DO 11 K = 1, KG
            INCKJ = INC(K, J)
            IF (INCKJ .EQ. 1) P = P + 1
   11       CONTINUE
C
      NCON = NCONP1 - 1
C                     DETERMINE IF PHI(N+1) IS IN THE MODEL.
   20 PHINP1 = .FALSE.
      DO 25 K = 1, KG
         IF (INC(K, NP1) .EQ. 1) PHINP1 = .TRUE.
   25    CONTINUE
C                                              ISEL = 1 OR 2
   30 IF (PHINP1) GO TO 40
   32    DO 35 I = 1, M
   35       R(I) = Y(I)
         GO TO 50
   40    DO 45 I = 1, M
   45       R(I) = Y(I) - R(I)
C                                     WEIGHT APPROPRIATE COLUMNS
   50 DO 55 I = 1, M
         ACUM = W(I)
         DO 55 J = FIRSTC, LASTC
   55       A(I, J) = A(I, J) * ACUM
C
C           COMPUTE ORTHOGONAL FACTORIZATIONS BY HOUSEHOLDER
C           REFLECTIONS.  IF ISEL = 1 OR 2, REDUCE PHI
C           (STORED IN THE FIRST N COLUMNS OF THE MATRIX A)
C           TO UPPER TRIANGULAR FORM, (Q*PHI = S), AND
C           TRANSFORM Y (STORED IN COLUMN N+1), GETTING
C           Q*Y = R.  IF ISEL = 1, ALSO TRANSFORM J = D PHI
C           (STORED IN COLUMNS N+2 THROUGH N+P+1 OF THE
C           MATRIX A), GETTING Q*J = F.  IF ISEL = 3 OR 4,
```

```
C           PHI HAS ALREADY BEEN REDUCED, TRANSFORM ONLY J.
C           S, R, AND F OVERWRITE PHI, Y, AND J, RESPEC-
C           TIVELY AND A FACTORED FORM OF Q IS SAVED IN U
C           AND THE LOWER TRIANGLE OF PHI.
C
      IF (N .EQ. 0) GO TO 75
      DO 70 K = 1, N
         KP1 = K + 1
         IF (ISEL .GE. 3 .OR. (ISEL .EQ. 2 .AND.
     X       K .LT.NCONP1)) GO TO 66
         ALPHA = 0.
         DO 62 I = K, M
   62       ALPHA = ALPHA + A(I, K)**2
         ALPHA = DSIGN(DSQRT(ALPHA), A(K, K))
         U(K) = A(K, K) + ALPHA
         A(K, K) = -ALPHA
         FIRSTC = KP1
         IF (ALPHA .NE. 0.0) GO TO 66
         ISEL = -8
          KK=K
         CALL VARERR (IPRINT, ISEL, KK)
         GO TO 99
C                                APPLY REFLECTIONS TO COLUMNS
C                                FIRSTC TO LASTC.
   66    BETA = -A(K, K) * U(K)
         DO 70 J = FIRSTC, LASTC
            ACUM = U(K)*A(K, J)
            DO 68 I = KP1, M
   68          ACUM = ACUM + A(I, K)*A(I, J)
            ACUM = ACUM / BETA
            A(K,J) = A(K,J) - U(K)*ACUM
            DO 70 I = KP1, M
   70          A(I, J) = A(I, J) - A(I, K)*ACUM
C
   75 IF (ISEL .GE. 3) GO TO 85
      RNORM = 0.
      DO 80 I = NP1, M
   80    RNORM = RNORM + R(I)**2
      RNORM = DSQRT(RNORM)
      IF (ISEL .EQ. 2) GO TO 99
      IF (NCON .GT. 0) SAVE = A(1, NCON)
C
C           F2 IS NOW CONTAINED IN ROWS N+1 TO M AND COLUMNS
C           N+2 TO N+P+1 OF THE MATRIX A.  NOW SOLVE THE
C           N X N UPPER TRIANGULAR SYSTEM S*C = R1 FOR THE
C           LINEAR PARAMETERS C.  C OVERWRITES R1.
C
   85 IF (N .GT. 0) CALL BACSUB (MMAX, N, A, R)
C
C           MAJOR PART OF KAUFMAN'S SIMPLIFICATION OCCURS
C           HERE.  COMPUTE THE DERIVATIVE OF ETA WITH
C           RESPECT TO THE NONLINEAR PARAMETERS
```

```
C
C     T  D ETA       T   N         D PHI(J)    D PHI(N+1)
C    Q *--------- = Q *(SUM C(J) --------- + ----------) =  F2*C
C       D ALF(K)         J=1       D ALF(K)    D ALF(K)
C
C            AND STORE THE RESULT IN COLUMNS N+2 TO N+KG+1.
C            IF ISEL NOT = 4, THE FIRST N ROWS ARE OMITTED.
C            THIS IS -D(Q2)*Y.  IF ISEL NOT = 4 THE RESIDUAL
C            R2 = Q2*Y (IN COL. N+1) IS COPIED TO COLUMN
C            N+KG+2.  OTHERWISE ALL OF COLUMN N+1 IS COPIED.
C
      DO 95 I = FIRSTR, M
         IF (N .EQ. NCON) GO TO 95
         L = NP1
         DO 90 K = 1, KG
            ACUM = 0.
            DO 88 J = NCONP1, N
               IF (INC(K, J) .EQ. 0) GO TO 88
               L = L + 1
               ACUM = ACUM + A(I, L) *R(J)
   88          CONTINUE
            KSUB = NP1 + K
            IF (INC(K, NP1) .EQ. 0) GO TO 90
            L = L + 1
            ACUM = ACUM + A(I, L)
   90       A(I, KSUB) = ACUM
   95    A(I, NKG2) = R(I)
C
   99 RETURN
      END
C
```

```
      SUBROUTINE BACSUB (MMAX, N1, A1, Z)
C
C        BACKSOLVE THE N1 X N1 UPPER TRIANGULAR SYSTEM
C        A1*Z = B.  THE SOLUTION Z OVERWRITES THE RIGHT
C        SIDE B.
C
      DOUBLE PRECISION A1(MMAX, N1), Z(N1), ACUM
C
      Z(N1) = Z(N1) / A1(N1, N1)
      IF (N1 .EQ. 1) GO TO 30
      NP1 = N1 + 1
      DO 20 IBACK = 2, N1
         I = NP1 - IBACK
C           I = N-1, N-2, ..., 2, 1
         IP1 = I + 1
         ACUM = Z(I)
         DO 10 J = IP1, N1
   10       ACUM = ACUM - A1(I,J)*Z(J)
   20    Z(I) = ACUM / A1(I,I)
C
   30 RETURN
      END
```

```
      SUBROUTINE POSTPR(N,KG,M,MMAX,NKG2,RNORM,IPRINT,ALF,W,
     X  A,R,U,IERR)
C
C        CALCULATE RESIDUALS, SAMPLE VARIANCE, AND
C        COVARIANCE MATRIX.  ON INPUT, U CONTAINS
C        INFORMATION ABOUT HOUSEHOLDER REFLECTIONS FROM DPA.
C        ON OUTPUT, IT CONTAINS THE LINEAR PARAMETERS.
C
      DOUBLE PRECISION A(MMAX,NKG2),ALF(KG),R(M),U(N),W(M),
     X ACUM,PRJRES,RNORM,SAVE1
      INTEGER OUTPUT
         LOGICAL IP
         DOUBLE PRECISION BM(100,19)
         COMMON /BLKBM/ BM
         COMMON /BLOKIP/ IP
      DATA OUTPUT /5/
C
      NP1 = N + 1
      NPKG = NKG2 - 2
      NKG1 = NPKG + 1
      DO 10 I = 1, M
   10    W(I) = W(I)**2
C
C              UNWIND HOUSEHOLDER TRANSFORMATIONS TO GET
C              RESIDUALS AND MOVE THE LINEAR PARAMETERS
C              FROM R TO U.
C
      IF (N .EQ. 0) GO TO 30
      DO 25 KBACK = 1, N
         K = NP1 - KBACK
         KP1 = K + 1
         ACUM = 0.
         DO 20 I = KP1, M
   20       ACUM = ACUM + A(I, K) * R(I)
         SAVE1 = R(K)
         R(K) =  ACUM / A(K, K)
         ACUM = -ACUM / (U(K) * A(K, K))
         U(K) = SAVE1
         DO 25 I = KP1, M
   25       R(I) = R(I) - A(I, K)*ACUM
         IF (KG .EQ. 0) GOTO 45
         DO 27 J=NP1,NPKG
            JP1=J+1
            DO 27 I=1,M
27             BM(I,J)=A(I,JP1)/DSQRT(W(I))
C
C              THE FIRST N COLUMNS OF THE MATRIX HAVE BEEN
C              REDUCED TO UPPER TRIANGULAR FORM IN DPA.
C              FINISH BY REDUCING ROWS N+1 TO M AND COLUMNS
C              N+2 THROUGH N+KG+1 TO TRIANGULAR FORM.  THEN
C              SHIFT COLUMNS OF DERIVATIVE MATRIX OVER ONE
C              TO THE LEFT TO BE ADJACENT TO THE FIRST N
```

```
C              COLUMNS.
C
   30 CALL ORFAC1(KG+1,MMAX,M,N,IPRINT,A(1,N+2),PRJRES,4)
      DO 40 I = 1, M
         A(I, NKG2) = R(I)
         DO 40 K = NP1, NKG1
   40       A(I, K) = A(I, K+1)
C                                 COMPUTE COVARIANCE MATRIX
   45 A(1, NKG2) = RNORM
      ACUM = RNORM*RNORM/(M - N - KG)
      A(2, NKG2) = ACUM
      CALL COV(MMAX, NPKG, ACUM, A)
C
      IF (IPRINT .LT. 0) GO TO 99
      WRITE (OUTPUT, 209)
         IF (IP) PRINT 209
      IF (N .GT. 0) WRITE (OUTPUT, 210) (U(J), J = 1, N)
         IF ((N .GT. 0) .AND. (IP)) PRINT 210,(U(J),J=1,N)
      IF (KG .GT. 0) WRITE (OUTPUT, 211) (ALF(K), K = 1, KG)
      IF ((KG .GT. 0) .AND. (IP)) PRINT 211,(ALF(K),K=1,KG)
      WRITE (OUTPUT, 214) RNORM, ACUM
         IF (IP) PRINT 214,RNORM,ACUM
      WRITE (OUTPUT, 209)
         IF (IP) PRINT 209
   99 RETURN
C
  209 FORMAT (1H0, 50(1H'))
  210 FORMAT (20H0  LINEAR PARAMETERS // (4E15.7))
  211 FORMAT (23H0  NONLINEAR PARAMETERS // (4E15.7))
  214 FORMAT (21H0  NORM OF RESIDUAL =, E15.7/
     X  23H0  ESTIMATED VARIANCE =, E15.7 )
      END
```

```
      SUBROUTINE COV(MMAX, NPKG, SIGMA2, A)
C
C           COMPUTE THE UNSCALED COVARIANCE MATRIX OF THE
C        N + KG PARAMETERS.  THIS INVOLVES COMPUTING
C
C                                2     -1    -T
C                          SIGMA  *  T    * T
C
C        WHERE THE (N+KG) X (N+KG) UPPER TRIANGULAR MATRIX T
C        IS DESCRIBED IN SUBROUTINE POSTPR.  THE RESULT
C        OVERWRITES THE FIRST N+KG ROWS AND COLUMNS OF THE
C        MATRIX A.  THE RESULTING MATRIX IS SYMMETRIC.
C
C           REFERENCE:  LAWSON AND HANSON, SOLVING LEAST
C        SQUARES PROBLEMS, PRENTICE-HALL, 1974, PP. 67-
C        70, 281.
C
C     ..................................................
C
      DOUBLE PRECISION A(MMAX, NPKG), SUM, SIGMA2
C
      DO 10 J = 1, NPKG
   10    A(J, J) = 1./A(J, J)
C
C                    INVERT T UPON ITSELF
C
      IF (NPKG .EQ. 1) GO TO 70
      NM1 = NPKG - 1
      DO 60 I = 1, NM1
         IP1 = I + 1
         DO 60 J = IP1, NPKG
            JM1 = J - 1
            SUM = 0.
            DO 50 L = I, JM1
   50          SUM = SUM + A(I, L) * A(L, J)
   60       A(I, J) = -SUM * A(J, J)
C
C                    NOW FORM THE MATRIX PRODUCT
C
   70 DO 90 I = 1, NPKG
         DO 90 J = I, NPKG
            SUM = 0.
            DO 80 L = J, NPKG
   80          SUM = SUM + A(I, L) * A(J, L)
            SUM = SUM * SIGMA2
            A(I, J) = SUM
   90       A(J, I) = SUM
C
      RETURN
      END
```

```
      SUBROUTINE VARERR (IPRINT, IERR, K)
C
C        PRINT ERROR MESSAGES
C
         LOGICAL IP
         COMMON /BLOKIP/ IP
      INTEGER ERRNO, OUTPUT
      DATA OUTPUT /5/
C
      IF (IPRINT .LT. 0) GO TO 99
      ERRNO = IABS(IERR)
      GO TO (1, 2, 99, 4, 5, 6, 7, 8) ERRNO
C
    1 WRITE (OUTPUT,101)
      IF (IP) PRINT 101
      GO TO 99
    2 WRITE (OUTPUT,102)
      IF (IP) PRINT 102
      GO TO 99
    4 WRITE (OUTPUT,104)
      IF (IP) PRINT 104
      GO TO 99
    5 WRITE (OUTPUT,105)
      IF (IP) PRINT 105
      GO TO 99
    6 WRITE (OUTPUT,106) K
      IF (IP) PRINT 106,K
      GO TO 99
    7 WRITE (OUTPUT,107) K
      IF (IP) PRINT 107,K
      GO TO 99
    8 WRITE (OUTPUT,108) K
      IF (IP) PRINT 108,K
C
   99 RETURN
  101 FORMAT ('   PROBLEM TERMINATED FOR EXCESSIVE',
     X  ' ITERATIONS'//)
  102 FORMAT ('   PROBLEM TERMINATED BECAUSE OF ILL-',
     X  'CONDITIONING'//)
  104 FORMAT (/ '  INPUT ERROR IN PARAMETER N,KG,M,',
     X   'NPP2, OR MMAX.' /)
  105 FORMAT ('   ERROR -- INC MATRIX IMPROPERLY SPECIFIED',
     X   ' OR DISAGREES WITH NPP2.' /)
  106 FORMAT ('   ERROR -- WEIGHT(',I4,') NOT',
     X   ' NONNEGATIVE.' /)
  107 FORMAT ('   ERROR -- CONSTANT COLUMN',I3,
     X   ' MUST BE COMPUTED ONLY WHEN ISEL = 1.' /)
  108 FORMAT ('   CATASTROPHIC FAILURE -- COLUMN',I4,
     X   ' IS ZERO. SEE DOCUMENTATION.' /)
      END
```

```
      SUBROUTINE ADA (NP1,N,KG,M,MMAX,NPP2,IV,A,T,ALF,
     *   ISEL,PHINP1,NCONP1)
C
C  EVALUATE THE FUNCTIONS PHI AND THEIR PARTIAL DERIVATIVES
C  D PHI(J)/D ALF(K) AT THE SAMPLE POINTS T(I).
C       ISEL=1 MEANS COMPUTE BOTH THE FUNCTIONS AND
C              DERIVATIVES
C       ISEL=2 MEANS COMPUTE ONLY THE FUNCTIONS PHI
C       ISEL=3 MEANS COMPUTE ONLY THE DERIVATIVES
C
      DOUBLE PRECISION A(MMAX,NPP2),ALF(KG),T(MMAX,IV)
      DOUBLE PRECISION C(7),ETA
      INTEGER INC (12,8)
      LOGICAL PHINP1,NC
      COMMON /NCBLK/ NC
      COMMON /BLKINC/INC
      COMMON /BLKPH/ PH
      DOUBLE PRECISION PH(8)
      GOTO (10,20,30) ISEL
C
C  CALCULATE FUNCTION VALUES & STORE THEM IN FIRST (N+1)
C  COLUMNS OF A.
C
10    PHINP1=.TRUE.
      NPH=NP1
      NCONP1=1
20    IF (.NOT. PHINP1) NPH=N
      DO 22 I=1,M
         II=I
         CALL CALC (II,KG,N,MMAX,IV,C,ALF,T,ETA)
         DO 22 J=NCONP1,NPH
            A(I,J)=PH(J)
22       CONTINUE
      NC=.TRUE.
      IF (ISEL .EQ. 2) GOTO 35
      IF (KG .EQ. 0) GOTO 35
C
C  CALCULATE NONVANISHING PARTIAL DERIVATIVES & STORE THEM
C  IN COLUMNS (N+2) THROUGH (N+P+2) OF A.
C
30    IK=NP1
      DO 31 I=1,KG
         II=I
         DO 31 J=1,NP1
            JJ=J
            IF (INC(I,J) .NE. 1) GOTO 31
            IK=IK+1
            IN=IK-NP1
            DO 31 K=1,M
               KK=K
               CALL NUMDER (II,JJ,KK,IN,N,KG,IV,M,
     *            MMAX,T,ALF,A(K,IK))
```

```
31          CONTINUE
35          RETURN
            END
```

```
      SUBROUTINE NUMDER (I,J,K,IN,N,KG,IV,M,MMAX,
     *     T,ALF,DV)
C
C  GIVEN AN ARRAY OF NUMERICAL DERIVATIVE INCREMENTS,
C  CALCULATE DERIVATES NUMERICALLY IF INCREMENT IS NOT ZERO,
C  USE CLOSED FORM IN SUBROUTINE DERIV IF INCREMENT IS ZERO.
C
      DOUBLE PRECISION T(MMAX,IV),ALF(KG),NDI(15)
      DOUBLE PRECISION DV,ASAVE,YC,YD,DK
      LOGICAL NC
      COMMON /NCBLK/ NC
      COMMON /BLKNDI/ NDI
      COMMON /BLKDPH/ DPH
      DOUBLE PRECISION DPH(8,12),CC(7)
      DK=NDI(IN)
      IF (DK .EQ. 0.) GOTO 195
C
C  DERIVATIVE TO BE CALCULATED NUMERICALLY USING FORWARD
C  DIFFERENCE QUOTIENT.
C
      NP1=N+1
      DO 200 JJ=1,N
200      CC(JJ)=0.
      IF (J .NE. NP1) CC(J)=1.
      NC=.FALSE.
      CALL CALC (K,KG,N,MMAX,IV,CC,ALF,T,YC)
      ASAVE=ALF(I)
      ALF(I)=ASAVE+DK
      CALL CALC (K,KG,N,MMAX,IV,CC,ALF,T,YD)
      NC=.TRUE.
      DV=(YD-YC)/DK
      ALF(I)=ASAVE
      RETURN
C
C  DERIVATIVE TO BE CALCULATED FROM CLOSED FORM IN
C  SUBROUTINE DERIV.
C
195   CALL DERIV(K,T,MMAX,IV,ALF,KG,NP1)
      DV=DPH(J,I)
      RETURN
      END
```

APPENDIX B

DESCRIPTION OF VARPRO SUBROUTINES

ADA

Evaluates ϕ and $\frac{d\phi_i}{d\alpha_k}$ and stores them in the matrix A. A is (MMAX) by (N+P+2) or (N+KG+2). That is,

$$A \equiv \underset{N \quad 1 \quad P \quad 1}{\left(\Phi \mid\mid D\Phi \right)} \quad M$$

ORFAC1

Stage 1 of Householder reduction:

$$\underset{KG \quad 1}{M\text{-}N \left(DR \mid R2 \right)} \qquad \text{to} \qquad \underset{KG \quad 1}{\left(\begin{array}{c|c} DR' & R3 \\ 0 & R4 \end{array} \right)} \begin{array}{l} \} \ KG \\ \} \ M\text{-}N\text{-}KG \end{array}$$

where DR = -D(Q2)Y is the derivative of the modified residual produced by DPA; R2 is the transformed residual from DPA. DR' is in upper triangular form. [Note: DR is stored in rows N+1 to M and columns (N+2) to (N+KG+1) of the matrix A (columns 1 to KG of B). R2 is stored in column (N+KG+2) of A (column (KG+1) of B).]

For K = 1, ... , KG find the reflection I -UU'/BETA which zeros B(I,K), I = N+K+1, ... , M. Apply reflections to remaining columns and to residual vector. Save upper triangular form and transformed residual for use in case a step is retracted. Also compute column lengths.

ORFAC2

Stage 2 of Householder reduction:

$$\begin{array}{r} KG \\ M\text{-}N\text{-}KG \\ KG \end{array}\left\{\begin{array}{c|c} DR' & R3 \\ 0 & R4 \\ nu*D & 0 \end{array}\right. \qquad \text{to} \qquad \left.\begin{array}{c|c} DR'' & R5 \\ 0 & R4 \\ 0 & R6 \end{array}\right\}\begin{array}{l} KG \\ M\text{-}N\text{-}KG \\ KG \end{array}$$

$$\begin{array}{c} KG \quad 1 \end{array} \qquad\qquad \begin{array}{c} KG \quad 1 \end{array}$$

where D is diagonal consisting of the lengths of the columns of DR'. DR'' is in upper triangular form. [Note: The K-th reflection modifies only rows K, KG+1, KG+2, ... , KG+K and columns K to KG+1.]

DPA

Compute the norm of the residual if ISEL = 1 or 2.

Compute the (M-N) by (KG) derivative of the modified residual (M-N) vector (Q2)Y if ISEL = 1, 3, or 4. (ISEL = 4 on last pass.)

$$\begin{array}{r} N \\ M\text{-}N \end{array}\left\{\begin{array}{c} Q1 \\ Q2 \end{array}\right. \cdot \left(\begin{array}{c|c|c} \Phi & Y & D\Phi \end{array}\right) \qquad \text{to} \qquad \left(\begin{array}{c|c|c} S & R1 & F1 \\ 0 & R2 & F2 \end{array}\right) \begin{array}{l} M \end{array}$$

$$\begin{array}{ccccc} M & & N & 1 & P \end{array} \qquad\qquad \begin{array}{ccc} N & 1 & P \end{array}$$

Here $Q\Phi = S$; Q is MxM orthogonal, S is upper triangular. [Note: norm of residual = norm (R2); desired derivative $D[(Q2)Y] = -Q2(D\Phi)S^{-1}(Q1)Y$ (according to Kaufman).]

The steps involved in subroutine DPA are as follows:

(a) Determine number of constant functions.

(b) Weight appropriate columns (FIRSTC to LASTC, depending on ISEL).

(c) Compute orthogonal factorizations by Householder reflections.

(d) If ISEL = 1 or 2:

Reduce Φ to upper triangular form, $Q\Phi = S$.

Transform Y to QY = R

Calculate norm (R2).

If ISEL = 1:

Transform $J = D\Phi$ to QJ = F.

If ISEL = 3 or 4:

Φ has already been reduced. Transform only $J = D\Phi$ to $QJ = F$. S, R and F overwrite Φ, Y and $D\Phi$ in A.

(e) A factored form of Q is saved in U and the lower triangle of Φ.

(f) Apply reflections to columns FIRSTC to LASTC (determined by ISEL).

(g) F2 is contained in rows (N+1) to (M) and columns (N+2) to (N+P+1) of A. Now solve the NxN upper triangular system SC = R1 for the linear parameters C. (C overwrites R1.)

(h) Use Kaufman's simplification to compute $\frac{d\eta}{d\alpha_k}$.

$$Q^T \frac{d\eta}{d\alpha_k} = Q^T \left[\sum_{j=1}^{n} \left(c_j \frac{d\phi_j}{d\alpha_k}\right) + \frac{d\phi_{n+1}}{d\alpha_k} \right] = (F2)C$$

The result is stored in columns (N+2) to (N+KG+1) (overwrites F).

(i) If ISEL $\neq$ 4:

The first N rows are omitted. This is -[D(Q2)]Y.

The residual R2 = (Q2)Y (in column N+1) is copied to column (N+KG+2).

If ISEL = 4:

All of column (N+1) is copied to column (N+KG+2).

BACSUB

Solves the N1xN1 upper triangular system AZ = B. The solution overwrites B.

POSTPR

Calculates residuals, sample variance and covariance matrix. (On input U contains information of Householder reflections; on output the linear parameters.)

The steps involved in subroutine POSTPR are as follows:

(a) Unwind Householder transformations to get residuals and move linear parameters from R to U.

(b) The first N columns have been reduced to upper triangular form in DPA. Finish by reducing rows (N+1) to (M) and columns (N+2) to (N+KG+1) to triangular form (location F2). Then shift columns of derivative matrix over one to the left to be adjacent to the first N columns.

$$\begin{array}{c} \left(\begin{array}{c|c|c|c} S & F1 & & R1 \\ 0 & F2 & & R2 \end{array}\right) \\ \begin{array}{cccc} N & KG & 1 & 1 \end{array} \end{array}$$

(c) Compute the covariance matrix.

(d) Print linear parameters, nonlinear parameters, norm of residual and estimated variance.

COV

Compute the unscaled covariance matrix of the (N+KG) parameters. This involves computing ($sigma^2$ T^{-1} T^{-T}) where the (N+KG) by (N+KG) upper triangular matrix T is that of POSTPR. The result overwrites the first (N+KG) rows and columns of A (and is symmetric).

VARERR

Prints error messages.

APPENDIX C

CHANGING THE CODING RESTRICTIONS

(1) To change the maximum number of characters in the title from 35 to $\bar{t}$:

In MAIN, replace

```
DIMENSION TITLE (35), NAM (40)
```

by

```
DIMENSION TITLE (t̄), NAM (40)
```

(2) To change the maximum number of characters in the file name (for storing data) from 5 to $\bar{f}$:

In MAIN, include

```
INTEGER FILE (f̄)
```

and replace

```
      340 FORMAT (A5)
```

by

```
      340 FORMAT (f̄A1).
```

(3) To change the maximum number of observations from 100 to $\bar{m}$:

In MAIN, replace

```
DOUBLE PRECISION A(100,24)
DOUBLE PRECISION T(100,6), Y(100), W(100)
DOUBLE PRECISION PH(8), DPH(8,12), ETA(100), R(100)
INTEGER IOR(100), IA(8)
DATA MMAX /100/
```

by

```
DOUBLE PRECISION A(m̄,24)
DOUBLE PRECISION T(m̄,6), Y(m̄), W(m̄)
DOUBLE PRECISION PH(8), DPH(8,12), ETA(m̄), R(m̄)
```

IOR($\bar{m}$), IA(8)

DATA MMAX /$\bar{m}$/.

In ORPLOT, replace

DOUBLE PRECISION T(M), SAV(100), HN, H

by DOUBLE PRECISION T(M), SAV($\bar{m}$), HN, H.

IN INTRVL, replace

DOUBLE PRECISION T(M), GN(100), AL, BH, STP

by DOUBLE PRECISION T(M), GN($\bar{m}$), AL, BH, STP.

In STAT, replace

DOUBLE PRECISION BM(100,19), XX(100,19), PH(8)

DOUBLE PRECISION SYH(100), SR(100), SRV(100)

DIMENSION IOR(100)

by

DOUBLE PRECISION BM($\bar{m}$,19), XX($\bar{m}$,19), PH(8)

DOUBLE PRECISION SYH($\bar{m}$), SR($\bar{m}$), SRV($\bar{m}$)

DIMENSION IOR($\bar{m}$).

In POSTPR, replace

DOUBLE PRECISION BM(100,19)

by DOUBLE PRECISION BM($\bar{m}$,19).

(4) To change the maximum number of independent variables per observation from 6 to $\bar{iv}$:

In MAIN, replace

DOUBLE PRECISION T(100,6), Y(100), W(100)

by DOUBLE PRECISION T(100,$\bar{iv}$), Y(100), W(100).

Note: The user may want to modify the printing formats for the table of values in MAIN (statement 1173) and in subroutine STAT (statement 1342).

(5) To change the maximum number of linear parameters from 7 to $\bar{n}$ (with maximum number of nonlinear parameters k and nonzero partials p):

In MAIN, replace

```
      DOUBLE PRECISION A(100,24)
      DOUBLE PRECISION ALF(12), C(7), NDI(15)
      DOUBLE PRECISION PH(8), DPH(8,12), ETA(100), R(100)
      DOUBLE PRECISION STP, CR(19,19), ANS
      INTEGER IOR(100), IA(8)
      INTEGER INC(12,8)
```

by

```
      DOUBLE PRECISION A(100, n̄+p+2)
      DOUBLE PRECISION ALF(k), C(n̄), NDI(15)
      DOUBLE PRECISION PH(n̄+1), DPH(n̄+1,k), ETA(100), R(100)
      DOUBLE PRECISION STP, CR(n̄+k,n̄+k), ANS
      INTEGER IOR(100), IA(n̄+1)
      INTEGER INC(k,n̄+1)
```

In PR, replace

```
      INTEGER INC(12,8), IA(8), P
```

by

```
      INTEGER INC(k,n̄+1), IA(n̄+1), P.
```

In DPA, replace

```
      INTEGER FIRSTC, FIRSTR, OUTPUT, P, INC(12,8)
```

by

```
      INTEGER FIRSTC, FIRSTR, OUTPUT, P, INC(k,n̄+1).
```

In ADA, replace

```
      INTEGER INC(12,8)
      DOUBLE PRECISION C(7), ETA
      DOUBLE PRECISION PH(8)
```

by

INTEGER INC(k,$\bar{n}$+1)

DOUBLE PRECISION C($\bar{n}$), ETA

DOUBLE PRECISION PH($\bar{n}$+1).

In NUMDER, replace

DOUBLE PRECISION DPH(8,12), CC(7)

by DOUBLE PRECISION DPH($\bar{n}$+k), CC($\bar{n}$).

In STAT, replace

DOUBLE PRECISION BM(100,19), XX(100,19), PH(8)

by DOUBLE PRECISION BM(100,$\bar{n}$+k), XX(100,$\bar{n}$+k), PH($\bar{n}$+1).

(6) To change the maximum number of nonlinear parameters from 12 to $\bar{k}$ (with maximum number of linear parameters n):

In MAIN, replace

DOUBLE PRECISION ALF(12), C(7), NDI(15)

DOUBLE PRECISION PH(8), DPH(8,12), ETA(100), R(100)

DOUBLE PRECISION STP, CR(19,19), ANS

INTEGER INC(12,8)

by

DOUBLE PRECISION ALF($\bar{k}$), C(n), NDI(15)

DOUBLE PRECISION PH(n+1), DPH(n+1,$\bar{k}$), ETA(100), R(100)

DOUBLE PRECISION STP, CR(n+$\bar{k}$,n+$\bar{k}$), ANS

INTEGER INC($\bar{k}$,n+1).

In PR, replace

INTEGER INC(12,8), IA(8), P

by INTEGER INC($\bar{k}$,n+1), IA(n+1), P.

In NUMDER, replace

DOUBLE PRECISION DPH(8,12), CC(7)

by DOUBLE PRECISION DPH(n+1,$\bar{k}$), CC(n).

In ADA, replace

INTEGER INC(12,8)

by INTEGER INC($\bar{k}$,n+1).

In DPA, replace

INTEGER FIRSTC, FIRSTR, OUTPUT, P, INC(12,8)

by INTEGER FIRSTC, FIRSTR, OUTPUT, P, INC($\bar{k}$,n+1).

In POSTPR, replace

DOUBLE PRECISION BM(100,19)

by DOUBLE PRECISION BM(100,n+$\bar{k}$).

In STAT, replace

DOUBLE PRECISION BM(100,19), XX(100,19), PH(8)

by DOUBLE PRECISION BM(100,n+$\bar{k}$), XX(100,n+$\bar{k}$), PH(n+1)

(7) To change the maximum number of nonzero partials from 15 to $\bar{p}$ (with maximum number of linear parameters n):

In MAIN, replace

DOUBLE PRECISION A(100,24)

DOUBLE PRECISION ALF(12), C(7), NDI(15)

by

DOUBLE PRECISION A(100,n+$\bar{p}$+2)

DOUBLE PRECISION ALF(12), C(n), NDI($\bar{p}$).

In CH, replace

DOUBLE PRECISION NDI(15)

by DOUBLE PRECISION NDI($\bar{p}$).

In PR, replace

DOUBLE PRECISION NDI(15)

by DOUBLE PRECISION NDI($\bar{p}$).

In NUMDER, replace

```
      DOUBLE PRECISION T(MMAX,IV), ALF(KG), NDI(15)
```

by

```
      DOUBLE PRECISION T(MMAX,IV), ALF(KG), NDI(p̄).
```

(8) To change the maximum number of lines of plotted output from 200 to $\bar{I}$:

In SETUP, replace

```
      DOUBLE PRECISION STP, Q, YG(200), YH(200), X(200)
      DO 50 I = 2,200
```

by

```
      DOUBLE PRECISION STP, Q, YG(Ī), YH(Ī), X(Ī)
      DO 50 I = 2,Ī.
```

In INTRVL, replace

```
      IF (GN(2) .LE. 200.) GOTO 50
```

by

```
      IF (GN(2) .LE. Ī) GOTO 50.
```

(9) To change the maximum number of characters per line of plotted output from 56 to $\bar{c}$:

In PLOT, replace

```
      DIMENSION IY(56)
      SCALE = 54./(BH-AL)
      SCALE = 54./BH
      IZERO = 55
      SCALE = 54./(-AL)
      DO 51 J = 1, 56
   60 FORMAT (' ',E12.5,3X,56A1)
```

by

```
      DIMENSION IY(c̄)
```

```
      SCALE = (c̄-2)/(BH-AL)
      SCALE = (c̄-2)/BH
      IZERO = (c̄-1)
      SCALE = (c̄-2)/(-AL)
      DO 51 J = 1, c̄
   60 FORMAT (' ',E12.5,3X,c̄A1).
```

(10) To change the maximum number of iterations from 50 to $\bar{i}$:

In VARPRO, replace

```
      DATA ITMAX /50/, EPS1 /1.D-6/, EPS2 /1.D-6/
```

by

```
      DATA ITMAX /ī/, EPS1 /1.D-6/, EPS2 /1.D-6/.
```

(11) To change the tolerance for the norm of the projection of the residual onto the range of the Jacobian of the variable projection functional from 1.D-6 to $\bar{v}$:

In VARPRO, replace

```
      DATA ITMAX /50/, EPS1 /1.D-6/, EPS2 /1.D-6/
```

by

```
      DATA ITMAX /50/, EPS1 /v̄/, EPS2 /1.D-6/.
```

Caution: $\bar{v}$ must not be less than ten times the unit roundoff of the machine.

(12) To change the tolerance for the change in the norm of the residual from 1.D-6 to $\bar{r}$:

In VARPRO, replace

```
      DATA ITMAX /50/, EPS1 /1.D-6/, EPS2 /1.D-6/
```

by

```
      DATA ITMAX /50/, EPS1 /1.D-6/, EPS2/r̄/.
```

Caution: $\bar{r}$ must not be less than ten times the unit roundoff of the machine.

(13) To change the maximum value allowed for the Marquardt parameter from 100 to $\bar{M}$:

In VARPRO, replace

```
IF (NU .LE. 100) GOTO 65
```

by

```
IF (NU .LE. M̄) GOTO 65.
```

VITA

Jane Elizabeth Rall (nee Nothdurft) was born in Lima, Peru on April 29, 1952. After attending the American Community School in Buenos Aires, Argentina for nine years she moved to Garden City, New York where she graduated from high school in June 1969. The following September she entered the State University College at New Paltz, New York, and in June 1973 received a Bachelor of Arts degree Magna Cum Laude with a major in Mathematics in the Secondary Education Curriculum.

After working for two years, she was awarded a Graduate Teaching Assistantship at the University of Tennessee. She received the Master of Science degree with a major in Mathematics and minor in Computer Science in December 1977.

The author is a member of the Honor Society of Phi Kappa Phi, the National Council of Teachers of Mathematics and the Mathematical Association of America.