TECHNICAL REPORT RSC-04

DISCUSSION OF THE LEAST SQUARES TECHNIQUE AND DEVELOPMENT OF A CURVE FITTING SUBROUTINE

by

T. A. Eppes

September 1969

supported by
National Aeronautics and Space Administration
NASA Grant ~~NsG 239-62~~ NGL-44-001-001
and
Texas Engineering Experiment Station

TEXAS A&M UNIVERSITY
REMOTE SENSING CENTER
COLLEGE STATION, TEXAS

DISCUSSION OF THE LEAST SQUARES TECHNIQUE AND DEVELOPMENT OF A CURVE FITTING SUBROUTINE

by

T.A. Eppes

I. INTRODUCTION

Attempts to interpret the relationship between backscatter measurements and the scattering elements have been made for over a decade. The most useful form of these data are as graphs of scattering coefficient as a function of incidence angle, for some applications it is sufficient to classify these graphs according to some characterizing factor such as slope of curve, etc. This general approach was employed by Rouse (1969) to categorize backscatter from Arctic ice. The data were fit to an equation developed by Hagfors (1964) based on the Kirchhoff method of describing scattering. The original work by Rouse used a manual technique of estimating the appropriate equation variable to fit the experimental data. This report describes a computer technique which improves the degree of "fit" and is readily adaptable to other scatterometer data analysis techniques.

II. LEAST SQUARES TECHNIQUE

In the final phase of an experiment, a decision must be made concerning the interpretation of the results.

This decision is often faciliated by previous experimentation and theoretical analysis of the problem. Often the decision is to "fit" the data to some desired mathematical model. Before the operation of this subroutine can be clearly understood, one needs a firm background in the theory of the least squares technique.

The method of least squares is an application of the science known as prediction analysis. Prediction analysis is a general method for predicting the accuracy of results that can be expected from an experiment. The following notation and derivation will be used in describing this technique (Wolberg, 1967).

Y_i = Observed values of the dependent variable.

σ_{yi} = Uncertainty (Standard Deriation) of Y_i.

X_{ji} = Observed value of the jth independent variable.

σ_{xji} = Uncertainty (Standard Deriation) of X_{ji}.

y_i = Calculated value of dependent variable.

x_{ji} = Calculated value of the jth independent variable.

n = Number of data points.

m = Number of independent variables.

a_k = Calculated values of the unknown parameters.

a_{ko} = Initial estimates of the unknown parameters.

p = Number of unknown parameters.

S = Squared error of data.

The derivation is begun by defining residuals and the squared error. The residual, R, and the squared error, S, are given by the following relations respectively:

$$R_{y_i} = Y_i - y_i$$

$$R_{x_{ji}} = X_{ji} - x_{ji}$$

and $$S = \sum_{i=1}^{n} \left(\frac{R_{y_i}^2}{\sigma_{y_i}^2} + \sum_{j=1}^{m} \frac{R_{x_{ji}}^2}{\sigma_{x_{ji}}^2} \right)$$

For any function the following is true given any set of calculated data:

$$F^i = 0 = y_i - f\,(x_{1i}, x_{2i}, \ldots, x_{mi}, a_1, a_2, \ldots, a_p)$$

for $i = 1, 2, \ldots, n.$

Since the experimental results will contain error, we can only estimate the functions F^i. We will denote $F_0{}^i$ as the estimates of the functions F^i for a given set of experimental data. In this case:

$$F_0{}^i = Y_i - f\,(X_{1i}, X_{2i}, \ldots, X_{mi}, a_{10}, a_{20}, \ldots, a_{p0}) \neq 0.$$

This equation will now be expanded in a Taylor's series, thereby making $F_0{}^i$ a function of the partials and the residuals of $F_0{}^i$. After expanding and neglecting the higher order terms, we have:

$$F_0{}^i = F_y{}^i R_{y_i} + \sum_{j=1}^{m} F_{x_j}{}^i R_{x_{ji}} + \sum_{k=1}^{p} F_{a_k}{}^i A_k \qquad (1)$$

where $$F_y{}^i = \frac{\partial F^i}{\partial y_i} ,$$

$$F_{x_j}{}^i = \frac{\partial F^i}{\partial x_{ji}} ,$$

$$F_{a_k}{}^i = \frac{\partial F^i}{\partial a_k} ,$$

and $R_{x_{ji}}$ and R_{y_i} are the previously defined residuals. A_k will be used to denote the variation around a_k ; namely $a_{ko} - a_k$. A relation must now be found between the residuals. This is accomplished by taking the differential of the S, squared error, equation. Since S is meant to be a minimum, small variations will not change the squared error. It therefore can be said that:

$$1/2\ \Delta S = \sum_{i=1}^{n} \left[\frac{R_{y_i} \Delta R_{y_i}}{\sigma^2_{y_i}} + \sum_{j=1}^{m} \frac{R_{x_{ji}} \Delta R_{x_{ji}}}{\sigma^2_{x_{ji}}} \right] = 0 \qquad (2)$$

Since $F_0{}^i$ is not a function of either R_{y_i} or $R_{x_{ji}}$ (see original equation of $F_0{}^i$), variation in them would not cause a change in $F_0{}^i$. Only by varying the a_{k0} set of parameters can $F_0{}^i$ be changed. In as much as A_k has been defined as the difference between the initial estimates and the true values of the variable parameters, $\Delta F_0{}^i$ can be set to zero and expressed as:

$$F_y{}^i \, \Delta R_{y_i} + \sum_{j=1}^{m} F_{x_j}{}^i \, \Delta R_{x_{ji}} + \sum_{k=1}^{p} F_{a_k}{}^i \, \Delta A_k = 0 \qquad (3)$$

for $i = 1, 2, \ldots, n.$

There now exists two sets of equations which are functions of the partials of F^i and the residuals of the experimental data. Equation (3) must now be multiplied by some multipler λ_i. This is done in order that we may subtract Equation (3) from Equation (2). This technique is known as the method of Lagrange multipliers. The n equations of (3) are each multiplied by a different multiplier $(\lambda_1, \lambda_2, \ldots, \lambda_i)$. The result of the subtraction of (3) from (2) is:

$$\sum_{i=1}^{n} \left(\frac{R_{y_i}}{\sigma^2_{y_i}} - \lambda_i F_y{}^i \right) \Delta R_{y_i} + \sum_{j=1}^{m} \sum_{i=1}^{n} \left(\frac{R_{x_{ji}}}{\sigma^2_{x_{ji}}} - \lambda_i F_{x_j}{}^i \right) \Delta R_{x_{ji}}$$

$$- \sum_{k=1}^{p} \left(\sum_{i=1}^{n} \lambda_i F_{a_k}{}^i \right) \Delta A_k = 0.$$

To satisfy this relation all of the coefficients of the variations must be equal to zero. Thus,

$$R_{y_i} = \sigma^2_{y_i} \, \lambda_i F_y{}^i, \; i = 1, 2, \ldots, n. \qquad (4)$$

$$R_{x_{ji}} = \sigma^2_{x_{ji}} \lambda_i F_{x_j}{}^{i}, \quad i = 1, 2, \ldots, n. \qquad (5)$$

and $$\sum_{i=1}^{n} \lambda_i F_{a_k}{}^{i} = 0, \qquad k = 1, 2, \ldots, p. \qquad (6)$$

Equations (4) and (5) give a relation between the residuals and the partials under least square conditions. The residuals can now be substituted for and thereby removed from the final results. Upon substituting Equations (5) and (6) into Equation (1) we have:

$$\sigma^2_{y_i} \lambda_i (F_y{}^{i})^2 + \sum_{j=1}^{m} \sigma_{x_{ji}} \lambda_i (F_{x_j}{}^{i})^2 + \sum_{k=1}^{p} F_{a_k}{}^{i} A_k = F_0{}^{i}, \quad (7)$$

for $i = 1, 2, \ldots, n$.

This equation can be simplified by the following substitution:

$$L_i \equiv \sigma^2_{y_i} (F_y{}^{i})^2 + \sum_{j=1}^{m} (\sigma_{x_{ji}} F_{x_j}{}^{i})^2,$$

for $i = 1, 2, \ldots, n$.

The partial of F^i with respect to y, $F_y{}^{i}$ is equal to one, and this further simplifies the substitution. Now:

$$L_i \equiv \sigma^2_{y_i} + \sum_{j=1}^{m} (\sigma_{x_{ji}} F_{x_j}{}^{i})^2,$$

for $i = 1, 2, \ldots, n$.

The result of the above substitution into Equation (7) is:

$$L_i \lambda_i = F_0{}^i - \sum_{k=1}^{p} F_{a_k}{}^i A_k \qquad (8)$$

It can now be seen that there exist two sets of equations which are functions of A_k and λ_i, Equations (6) and (8). Solving Equation (8) for λ_i we have:

$$\lambda_i = \frac{1}{L_i} [F_0{}^i - \sum_{k=1}^{p} F_{a_k}{}^i A_k],$$

$$\text{for} \quad i = 1, 2, \ldots, n.$$

Eliminating λ_i in Equation (6) by the above substitution yields a set of n + p equations. Namely,

$$A_1 \sum_{i=1}^{n} \frac{F_{a_1}{}^i F_{a_1}{}^i}{L_i} + A_2 \sum_{i=1}^{n} \frac{F_{a_1}{}^i F_{a_2}{}^i}{L_i} + \ldots$$

$$\vdots \qquad + A_p \sum_{i=1}^{n} \frac{F_{a_1}{}^i F_{a_p}{}^i}{L_i} = \sum_{i=1}^{n} \frac{F_{a_1}{}^i F_0{}^i}{L_i}$$

$$A_1 \sum_{i=1}^{n} \frac{F_{a_p}{}^i F_{a_1}{}^i}{L_i} + \ldots \qquad \vdots$$

$$+ A_p \sum_{i=1}^{n} \frac{F_{a_p}{}^i F_{a_p}{}^i}{L_i} = \sum_{i=1}^{n} \frac{F_{a_p}{}^i F_0{}^i}{L_i} .$$

This equation can be seen as functions of the partials of F^i with respect to the unknown parameters, a_k, the uncertainties and the partials of F^i with respect to independent variables, the variations in the unknown parameters, A_k, and the estimates of F^i as functions of the experimental data.

By adopting matrix notation and using the following substitutions we can greatly simplify the solution of A_k. Letting:

$$C_{kl} = \sum_{i=1}^{n} \left[\frac{F^i_{a_k} F^i_{a_l}}{L_i} \right], \qquad \begin{array}{l} k = 1, 2, \ldots, p. \\ l = 1, 2, \ldots, p. \end{array}$$

$$V_k = \sum_{i=1}^{n} \left[\frac{F^i_{a_k} F^i_0}{L_i} \right], \qquad k = 1, 2, \ldots, p.$$

we have:

$$\begin{array}{c} A_1C_{11} + A_2C_{12} + \cdots + A_pC_{1p} = V_1 \\ \vdots \qquad\qquad \vdots \\ A_1C_{p1} + A_2C_{p2} + \cdots + A_pC_{pp} = V_p \end{array}$$

This equation can be expressed as a simple linear matrix equation. Thus,

$$A = C^{-1}V$$

In matrix notation the C and V matrices can be expressed as:

$$C = (LF_a)^T F_a$$

and $$V = (LF_a)^T F_0.$$

The L matrix is a diagonal matrix of order n x n. It can be expressed as:

$$L = \begin{vmatrix} \frac{1}{L_1} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{L_2} & & & \\ \vdots & & \ddots & & \\ 0 & \cdots & & & \frac{1}{L_n} \end{vmatrix}_{n \times n}$$

The F_a matrix is the partial of F^i with respect to the variable parameters evaluated at each data point. Thus,

$$F_a = \begin{vmatrix} \frac{\partial F^1}{\partial a_1} & \frac{\partial F^1}{\partial a_2} & \cdots & \frac{\partial F^1}{\partial a_p} \\ \frac{\partial F^2}{\partial a_1} & & \ddots & \vdots \\ \vdots & & & \\ \frac{\partial F^n}{\partial a_1} & & \cdots & \frac{\partial F^n}{\partial a_p} \end{vmatrix}_{n \times p}$$

The F^0 matrix contain the estimates of F^i. That is:

$$F_0 = \begin{vmatrix} Y_1 - f(X_{11}, X_{21}, \ldots, X_{m1}, a_{10}, a_{20}, \ldots, a_{p0}) \\ Y_2 - f(X_{12}, X_{22}, \ldots, X_{m2}, a_{10}, a_{20}, \ldots, a_{p0}) \\ \vdots \qquad \vdots \qquad \vdots \\ Y_n - f(X_{1n}, X_{2n}, \ldots, X_{mn}, a_{10}, a_{20}, \ldots, a_{p0}) \end{vmatrix}_{n \times 1}$$

where Y_i = Experimental value of the ith dependent point,

X_{ji} = Experimental value of the jth independent variable at the ith data point,

and σ_{n0} = Initial estimates of the nth variable parameter.

The A matrix will be a p x 1 matrix containing the necessary increments to the initial guesses in obtaining a least squares fit. It can be expressed as:

$$A = \begin{vmatrix} A_1 \\ A_2 \\ \vdots \\ A_p \end{vmatrix}$$

where A_i = Necessary increment for ith variable parameter

The A matrix will be referred to as the incrementation matrix since it is the required correction needed to be added to initial guesses to obtain a least squares fit

Here one would assume that the job has been completed. However, this is true only for linear cases since the higher order terms were neglected in the Taylor's series expansion. By making the above procedure iterative, the values in the A matrix will approach zero as the true values of the parameters are reached. The higher order terms will prove to be less and less significant as each iteration is completed. This technique of guess, increment, and re-guess will in most cases converge. Many cases in which the initial estimates of the variable parameters were off by a magnitude of five have converged. On the other hand, estimates off by no more than 20% have failed to converge. In general, however the closer the initial estimates, the greater the probability of convergence.

To greatly simplify the teachnique of least squares, two assumptions were made regarding the uncertainties of the variables. First, the uncertainty of the independent variables was assumed to be 100%. Second, the uncertainty of the dependent variable was assumed to be the same for all data points. Data satisfying these assumptions are most common and the analysis is greatly simplified in as much as the L matrix becomes a scalar diagonal matrix and can be removed. The general incrementation matrix now becomes:

$$A = (F_a{}^T F_a)^{-1} F_a{}^T F_0 \qquad (9)$$

The subroutine herein discussed utilizes this equation in obtaining the per iteration correction term.

III.

Figures 1 and 2 provide a listing and a flow chart of the subroutine called NONLIN. The development of this subroutine can be classified into two categories. One, the formation of the array sizes and the derivation of a convergence testing criterion. Two, the ordering of the necessary matrix operations needed to satisfy Equation 9 This subroutine can be used for any number of independent variables or variable parameters. A vectorial storage technique is used; that is, any member of an array must be specified by one subscript. All data matrices and subroutine parameters must be supplied by the user.

Since this subroutine must provide for the general case, the results of all matrix operations are uniquely stored. The result of which is the need for much more storage area than would be absolutely necessary. By providing single storage capability, one may use the by-product matrices as output arrays in complex analyses. Two criteria are used in generating a "return", number of iteration completed and squared error. A third test could easily be implemented based on a comparison between the increment and the present value of all the variable parameters. This test would reveal percentage change information about the variable parameters. By applying all three of the above discussed criteria, an accurate estimate of the convergence possibilities could be determined, and an intelligent decision would be made.

Two function routines are required for the generation of the calculated data and the slope information. An array called SLOPE is iniated by evaluating the partials of function at every data point with respect to each variable parameter.

The function is then evaluated at all data points, and a calculated data array called SUBARR is created. This calculated data is subtracted from the experimental data array, DEXP, and an error matrix is formed called DDIFF. In the same loop in which DDIFF is created, the squared error is also calculated. The test sequence is placed after the squared error calculation, and a "return" will be generated if either of the two tests is satisfied.

Matrix operations are now performed upon the two arrays, SLOPE and DDIFF, according to Equation 9. The following matrix operating subroutines are needed: XGMTRA for transpositions, XGMPRD for multiplication, XGMINV for inversion, XGMSVB for subtraction, and XGMADD for addition. The incrementation array, DELTVP, is created by performing the necessary operations. After a "write" is executed, the previous variable parameters are altered by the addition of the DELTVP array. The "write" statement prints out four things: current variable parameter values, future variable parameter increments, the squared error, and the iteration number. The program now returns to calculate a new SLOPE and DDIFF arrays based on the new values for the variable parameters. After the matrix inversion, according to Equation 9, a test is performed to see if the determinant is zero, and an appropriate warning is printed if the test is affirmative.

IV.

In radar scatterometry analysis, a common equation used in curve fitting is based upon the Kirchhoff theory of scattering (Rouse, 1969). The reflection coefficient which relates received power to transmitted power can be expressed as a function of incidence angle. Accordingly:

$$P = -15\log_{10}|\cos^4\Theta + S\sin^2\Theta|$$

where Θ = Incidence angle,

P = Reflection coefficient,

and S = Surface roughness factor.

The family of curves formed by varying S from 2 to 200 is shown in Figure 3 (after Beckmann and Klemperer, 1965). P will be the dependent variable, Θ the independent variable, and S the variable parameter. The data in Table I are samples of experiment results (Rouse, 1969).

Table 1

Θ	P
(degrees)	(decibels)
2.5	0.0
6.7	- 8.7
15.0	-13.0
25.0	-15.3

From Figure 3 one can estimate the value of the variable S. Figure 4 is a printout of the following information for any given iteration: current value of S, change in S, the squared error, and the interation number. Figure 5 is a plot of experimental versus calculated data. Using the final value of S, as shown in Figure 4, a set of calculated data is created and is represented by the solid line curve. The experimental data is shown by the "+" characters.

If the mathematical model is altered with the same data, a unique effect is produced on the variable parameter S. Suppose the model had been:

$$P = -15\log_{10}|\cos^4\Theta + S\sin^2\Theta| + 2\log_{10}|\cos\Theta|$$

Figure 6 is a pointout similar to Figure 4. It can be seen that since the additional term forced the family of curves more negative, the S parameter must have been decreased in an effort to force a fit to the same data. Figure 7 is a plot of experimental versus calculated data; however, the effect of the additional term is difficult to see visually. It is demonstrated in the change of the variable S and the least square error.

Although the above example utilized a model equation with only one variable parameter, this subroutine is capable of handling many of them. The same is true for the number of independent variables and data points. The limitations are a function of the capability of the computer and the complexity of the squared error versus variable parameter contour. It is conceivable that one may never achieve an absolute minimum and the results may indeed be only relatively minimum. Ideally, the model equation's behavior as a function of all variable parameters should be clearly known before one attempts a least squares fit. The results may prove to be erroneous and misleading otherwise.

Figure 1 SUBROUTINE LISTING

```
  SUBROUTINE NONLIN(FUNC,PART,VARCOF,ERREXP,ERROBD,
C NL,DEXP,EXPI,MDATA,NVP,NIV,NI,DELTVP)
C      THE DATA MAY BE A FUNCTION OF MORE THAN ONE
C INDEPENDENT VARIABLE, HOWEVER ALL INDEPENDENT
C VARIABLES MUST BE SUPPLIED BY THE USER.  THE DEPEN-
C DENT VARIABLE OF EXP. DATA MUST ALSO BE SUPPLIED
C BY THE USER.  THE SUBROUTINE PROCEEDS TO INCREMENT
C THEM(UP OR DOWN) TO GIVE A LEAST SQUARES FIT.
C FUNC=  FUNCTION TO BE FITTED (IT IS A 'FUNCTION'
C      ROUTINE)
C PART=  PARTIAL OF FUNC WITH RESPECT TO ITH VARIABLE
C      PARAMETER.  IT IS ALSO A 'FUNCTION' ROUTINE.
C VARCOF=  VARIABLE PARAMETER ARRAY
C DELTVP=  INCREMENTATION OF VARCOF ARRAY
C ERREXP=  DESIRED LEAST SQUARED ERROR
C ERROBD=  OBTAINED LEAST SQUARED ERROR
C NL=  NUMBER OF ITERATIONS NOT TO BE EXCEEDED
C DEXP=  EXPERIMENTAL DATA (  DEPENDENT VARIABLE  FOR
C      ALL DATA POINTS)
C EXPI=  INPUT VARIABLES (INDEPENDENT VARIABLES FOR
C      ALL DATA POINTS)
C WARR=  WORKING ARRAY
```

SUBROUTINE LISTING (cont'd)

```
C  MDATA=  NUMBER OF MEMBERS OF EXP DATA (DEPENDENT
C       VARIABLE) ARRAY
C  DWORK=  WORKING ARRAY DIMENSIONED TO (NVP)
C  NVP=  NUMBER OF VARIABLE PARAMETERS
C  PRDARR=  WORKING ARRAY DIMENSIONED TO (NVP**2)
C  NI=  NUMBER OF ITERATIONS
C  MM=  WORKING ARRAY DIMENSIONED TO (NVP)
C  SWORK=  WORKING ARRAY DIMENSIONED TO (NVP*MDATA)
C  SLOPE=  WORKING ARRAY DIMENSIONED TO (NVP*MDATA)
C  DDIFF=  WORKING ARRAY DIMENSIONED TO (MDATA)
C  LL=  WORKING ARRAY DIMENSIONED TO (NVP)
C  NIV=  NUMBER OF DEPENDENT VARIABLES
C  VPINC=  INCREMENT OF VP'S DIMENSIONED TO (NVP)
C  SUBARR=  WORKING ARRAY FOR XGMSUB DIMENSIONED TO (MDATA)
C  FUNC AND PART MUST BE DECLARED EXTERNAL IN THE
C  MAIN PROGRAM.  ALL ARRAYS MUST BE STORED VECTORIALLY,
C  THAT IS, THEY ARE COLUMN MATRICES.
C  DIMENSION VARCOF(NVP),DELTVP(NVP),DEXP(MDATA),EXPI
C  (MDATA*NIV),WARR(NVP),SLOPE(MDATA*NVP),DDIFF(MDATA),
C  SUBARR(MDATA),SWORK(NVP*MDATA),PRDARR(NVP**2),DWORK
C  (NVP),VPINC(NVP),LL(NVP),MM(NVP)
   DIMENSION VARCOF(1),DELTVP(1),DEXP(4),EXPI(4),WARR(1),
   SLOPE(4),DDIFF(4),SUBARR(4),SWORK(4),PRDARR(1),DWORK(1),
```

SUBROUTINE LISTING (cont'd)

```
     VPINC(1),LL(1),MM(1)
     M= 1
     NI= 1
C    BEGINNING OF ITERATIVE PROCESS
   6 ERROBD= 0.0
C    NOW FILL UP ARRAY CONTAINING SLOPE INFORMATION
     DO 1 I=1,NVP
     DO 1 J=1,MDATA
     K= J+(I-1)*MDATA
   1 SLOPE(K)= PART(EXPI,VARCOF,MDATA,NIV,NVP,I,J)
C    NOW CALCULATE A DIFFERENCE OF EXPERIMENTAL AND CALCULATED DATA
C    AND PLACE IN THE ARRAY CALLED DDIFF
C    ALSO FROM THIS ARRAY CALCULATE THE SQUARED ERROR
     DO 2 I=1,MDATA
   2 SUBARR(I)= FUNC(EXPI,VARCOF,MDATA,NIV,NVP,I)
     CALL XGMSUB(DEXP,SUBARR,DDIFF,MDATA,M)
     DO 8 I=1,MDATA
   8 ERROBD= ERROBD+DDIFF(I)**2
C    TEST TO SEE IF LEAST SQUARED ERROR IS SATISFIED OR IF DESIRED
C    LIMIT OF ITERATIONS IS EXCEEDED
     IF(ERROBD.LE.ERREXP) GO TO 3
     IF(NI.EQ.NL) GO TO 4
C    PERFORM NECESSARY MATRIX OPERATIONS TO GET INCREMENTATION MATRIX
```

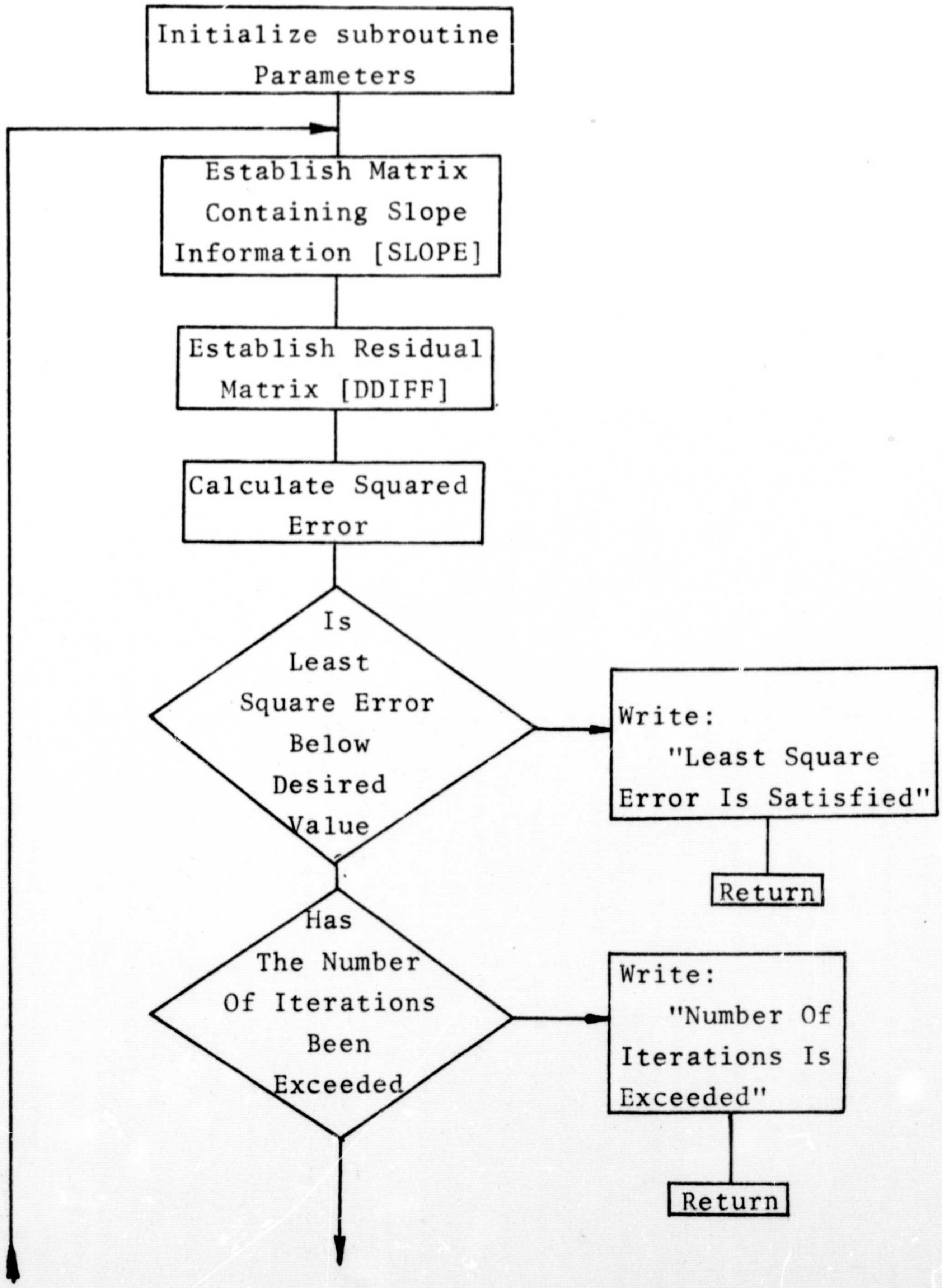
SUBROUTINE LISTING (cont'd)

```
    CALL XGMTRA(SLOPER,SWORK,MDATA,NVP,IER)
    ITEMP= NVP
    CALL XGMPRD(SWORK,SLOPER,PRDARR,NVP,MDATA,ITEMP)
    CALL XGMINV(PRDARR,NVP,DET,LL,MM)
C   IF DETERMINENT IS ZERO, THE MATRIX IS SINGULAR AND A MESSAGE
C   IS PRINTED
    IF(DET.EQ.0.0) GO TO 7
    CALL XGMPRD(SWORK,DDIFF,DWORK,NVP,MDATA,M)
    CALL XGMPRD(PRDARR,DWORK,DELTVP,NVP,ITEMP,M)
C   INCREMENT OLD VALUES OF VARIABLE PARAMETERS AND CREATE NEW
C   SET.  THIS SET SHOULD GIVE A BETTER LEAST SQUARES FIT THAN
C   THE OLD ONES
    WRITE(6,104) VARCOF(1),DELTVP(1),ERROBD,NI
104 FORMAT(' ',10X,F10.4,9X,E13.6,9X,E13.6,18X,I2/)
    CALL XGMADD(VARCOF,DELTVP,VPINC,NVP,M)
    DO 5 I=1,NVP
  5 VARCOF(I)= VPINC(I)
C   INCREASE LOOP PARAMETER
    NI= NI+1
C   RETURN TO EVALUATE THE NEW SET OF VARIABLE PARAMETERS
    GO TO 6
C   THE LEAST SQUARE ERROR IS SATISFIED
  3 WRITE(6,100)
```

SUBROUTINE LISTING (cont'd)

```
100 FORMAT(' '30X,'LEAST SQUARE ERROR IS SATISFIED')
    RETURN
C   THE MAXIMUM NUMBER OF LOOPS HAVE BEEN MADE
  3 WRITE (6,101)
101 FORMAT(' ',30X,'NUMBER OF ITERATIONS IS EXCEEDED')
    RETURN
C   THE MATRIX WAS A SINGULAR MATRIX
  7 WRITE(6,102)
102 FORMAT(' ',30X,'MATRIX WAS SINGULAR AND HAD NO INVERSE')
    RETURN
    END
    TOTAL MEMORY REQUIREMENTS 0006BC BYTES
```

Figure 2 SUBROUTINE LISTING


Initialize subroutine Parameters
Establish Matrix Containing Slope Information [SLOPE]
Establish Residual Matrix [DDIFF]
Calculate Squared Error
Is Least Square Error Below Desired Value
Write: "Least Square Error Is Satisfied"
Return
Has The Number Of Iterations Been Exceeded
Write: "Number Of Iterations Is Exceeded"
Return

SUBROUTINE LISTING (cont'd)

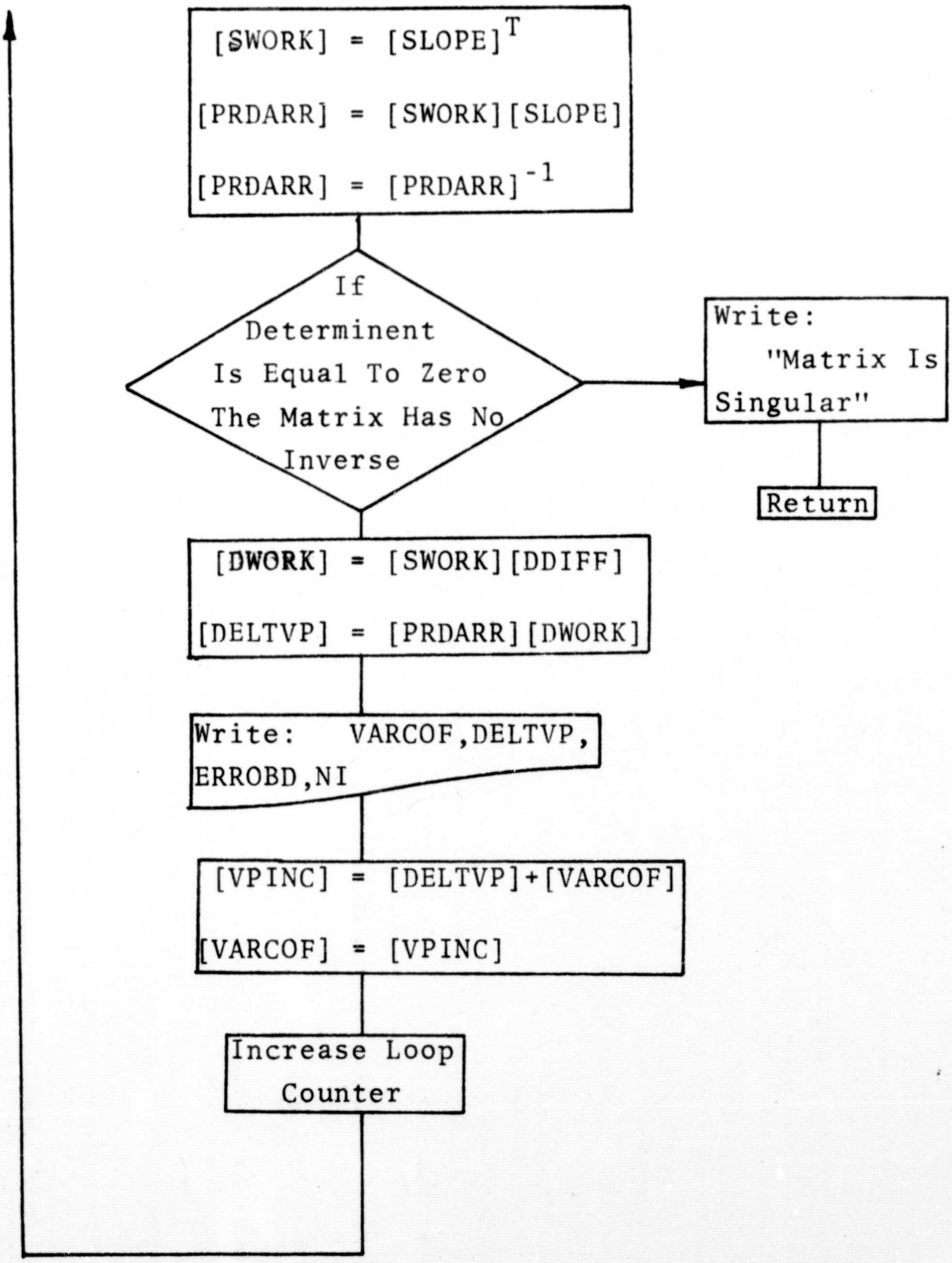

[SWORK] = [SLOPE]^T
[PRDARR] = [SWORK][SLOPE]
[PRDARR] = [PRDARR]$^{-1}$
If Determinent Is Equal To Zero The Matrix Has No Inverse
Write: "Matrix Is Singular"
Return
[DWORK] = [SWORK][DDIFF]
[DELTVP] = [PRDARR][DWORK]
Write: VARCOF,DELTVP, ERROBD,NI
[VPINC] = [DELTVP]+[VARCOF]
[VARCOF] = [VPINC]
Increase Loop Counter

Figure 3

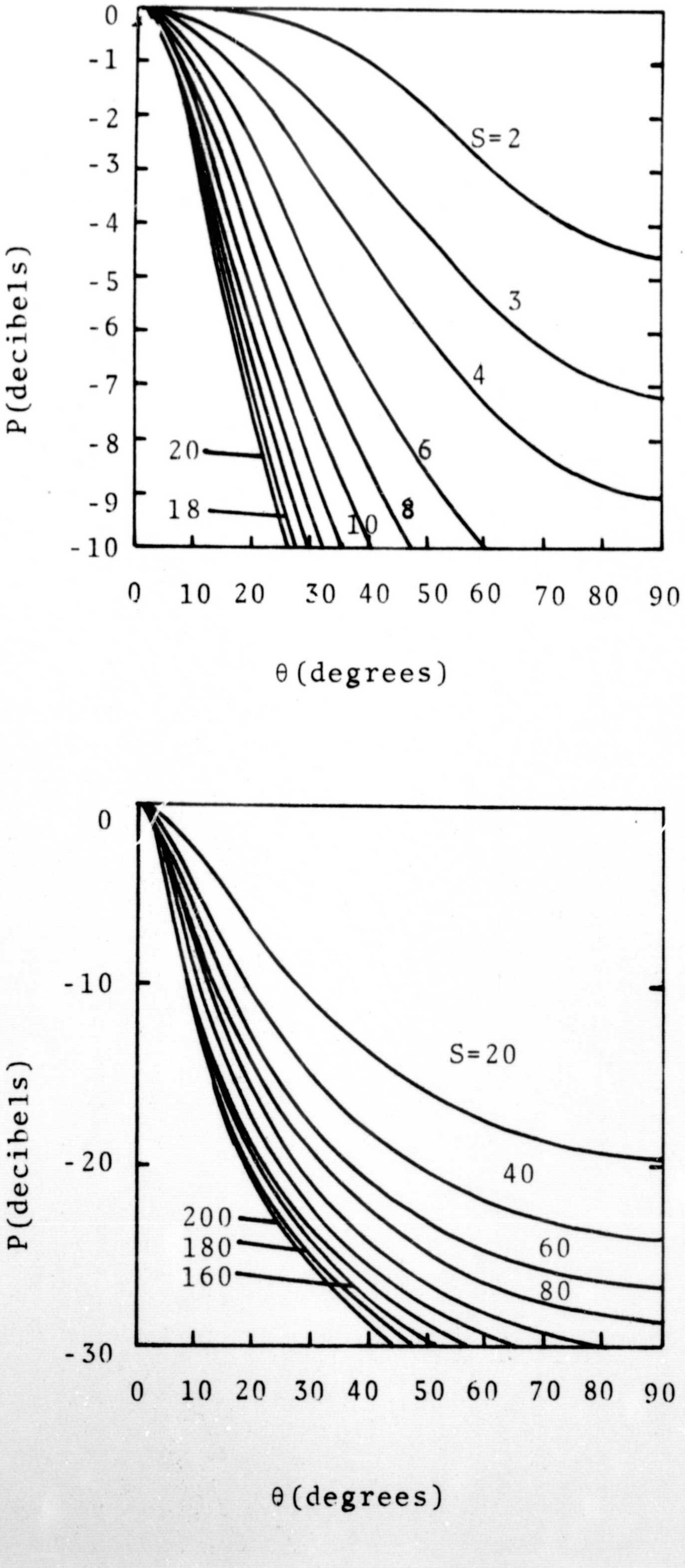

0
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
P(decibels)
S=2
3
4
6
8
10
18
20
0
10
20
30
40
50
60
70
80
90
θ(degrees)
0
-10
-20
-30
P(decibels)
S=20
40
60
80
160
180
200
0
10
20
30
40
50
60
70
80
90
θ(degrees)

Figure 4

```
VALUE OF S     CHANGE IN S     SQUARED ERROR     NUMBER OF ITERATIONS


  50.0000      0.111270E 02    0.431651E 02               1

  61.1270      0.831251E 01    0.310287E 02               2

  69.4395      0.571977E 01    0.260673E 02               3

  75.1592      0.372248E 01    0.241431E 02               4

  78.8817      0.233666E 01    0.234237E002               5

  81.2184      0.143401E 01    0.231609E 02               6

  82.6524      0.867864E 00    0.230665E 02               7

  83.5202      0.520789E 00    0.230329E 02               8

  84.0410      0.310993E 00    0.230209E 02               9

  84.3520      0.185086E 00    0.230168E 02              10

  84.5371      0.110026E 00    0.230153E 02              11

  84.6471      0.652887E-01    0.230148E 02              12


                   NUMBER OF ITERATIONS EXCEEDED LIMIT
```

Figure 5

Experimental Versus Calculated Data

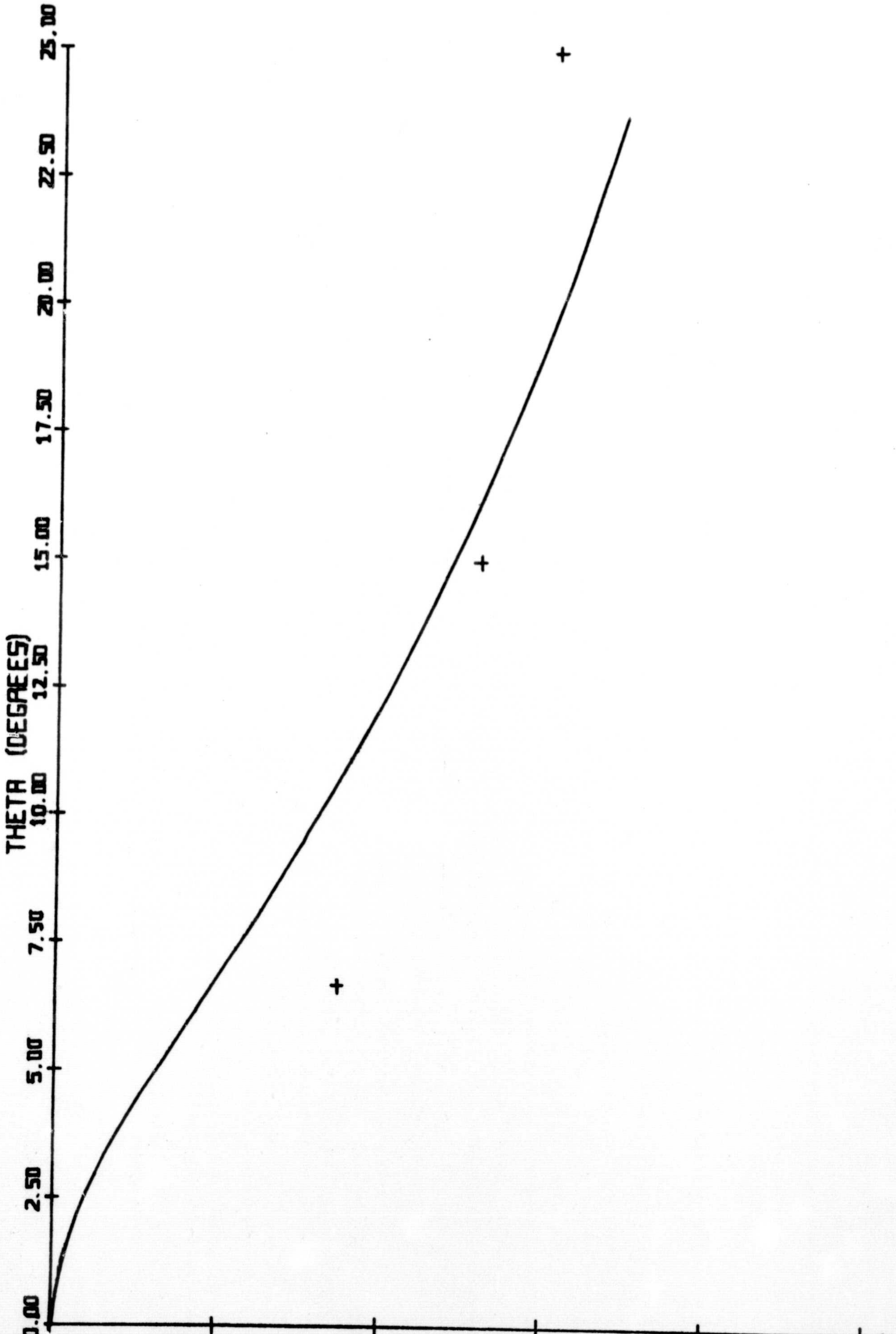
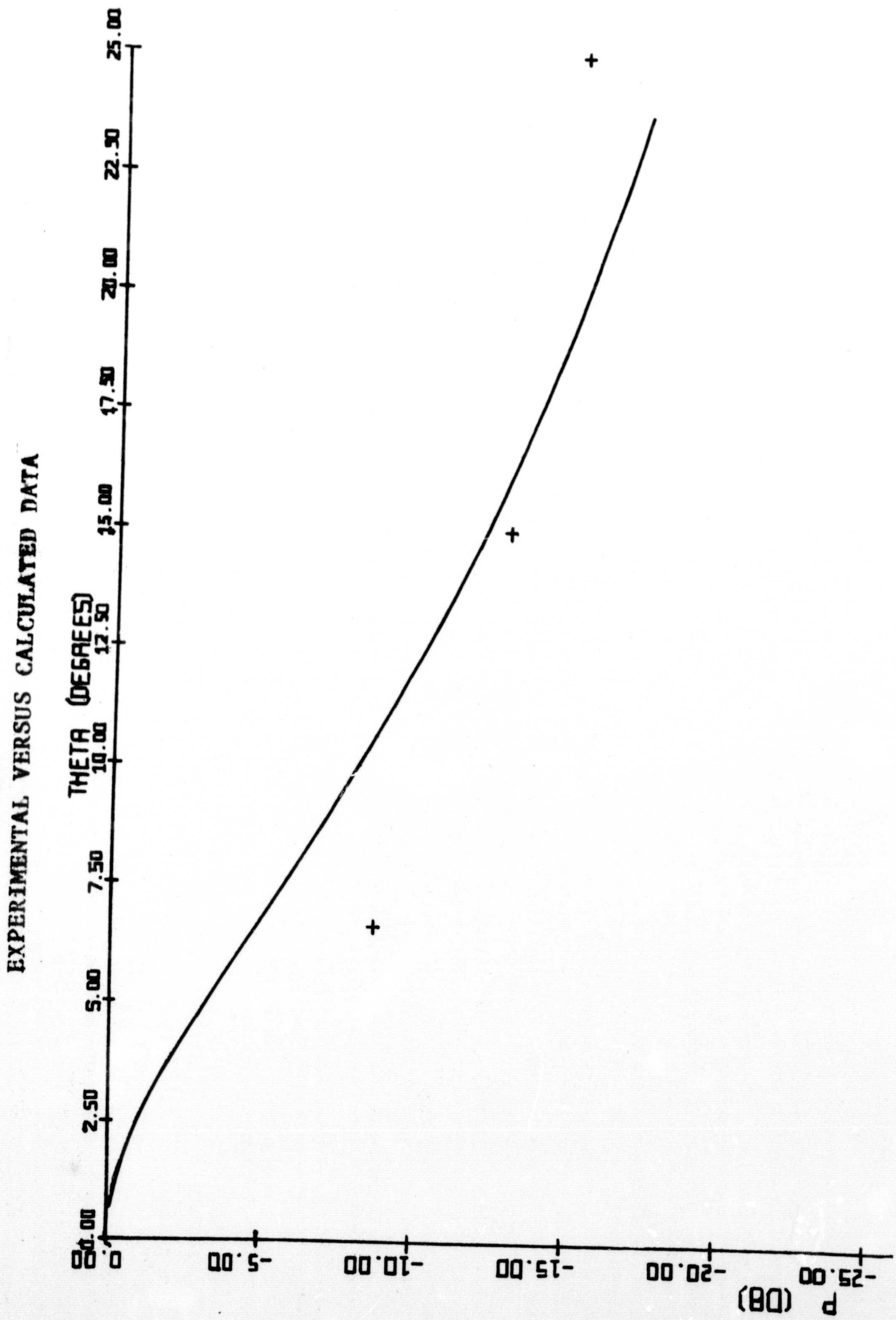


Figure 6

VALUE OF S	CHANGE IN S	SQUARED ERROR	NUMBER OF ITERATIONS
50.0000	0.109159E 02	0.427955E 02	1
60.9159	0.812778E 01	0.310999E 02	2
69.0436	0.558151E 01	0.263244E 02	3
74.6252	0.362872E 01	0.244732E 02	4
78.2539	0.227703E 01	0.237808E 02	5
80.5309	0.139746E 01	0.235278E 02	6
81.9284	0.846004E 00	0.234368E 02	7
82.7744	0.507918E 00	0.234044E 02	8
83.2823	0.303502E 00	0.233928E 02	9
83.5858	0.180777E 00	0.233888E 02	10
83.7665	0.107472E 00	0.233874E 02	11
83.8740	0.638265E-01	0.233869E 02	12

Figure 7


EXPERIMENTAL VERSUS CALCULATED DATA
THETA (DEGREES)
0.00
2.50
5.00
7.50
10.00
12.50
15.00
17.50
20.00
22.50
25.00
P (DB)
0.00
-5.00
-10.00
-15.00
-20.00
-25.00

REFERENCES

Aitken, A.C., Statistical Mathematics, Inerscience Publishers, New York, 1952.

Beckmann, P. and Klemperer, W.K., "Interpretation of the Angular Dependence of Backscattering From the Moon and Venus", Radio Science, Vol. 69D, pp. 1669-1676, December, 1965.

Guest, P.G., Numerical Methods of Curve Fitting, Cambridge University Press, New York, 1961.

Hagfors, T., "Backscatter From an Undulating Surface with Application to Radar Returns from the Moon", J. Geophys. Research, Vol. 69, No. 18, pp. 779-3784, 1964.

Horst, P., Factor Analysis of Data Matrices, Holt, Rinehart, and Winston, Inc., New York, 1965.

Rouse, J.W., "Artic Ice Type Identification by Radar", Proc. of the IEEE, Vol. 57, No. 4, pp. 605-611, April 1969.

Weld, L.D., Theory of Errors and Least Squares, The Macmillan Company, New York, 1916.

Wolberg, J.R., Prediction Analysis, D. Van Nostrand Company, Inc., Princetor, 1967.