27
7-8-80
240
SAND 80-0496
Unlimited Release

LSQFT -- A NONLINEAR LEAST SQUARES DATA FITTING
SUBROUTINE SUITABLE FOR MINICOMPUTERS

MASTER

W. D. Drotning

Prepared by Sandia Laboratories, Albuquerque, New Mexico 87185
and Livermore, California 94550 for the United States Department of
Energy under Contract AT(29-1)-789

May, 1980

Sandia National Laboratories

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Printed in the United States of America

Available from
National Technical Information Service
U. S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

Price: Printed Copy \$4.50; Microfiche \$3.00

M
LSQFT -- A NONLINEAR LEAST SQUARES DATA FITTING SUBROUTINE SUITABLE
FOR MINICOMPUTERS

W. D. Drotning
Thermophysical Properties Division 5824

Abstract

The FORTRAN subroutine LSQFT is described which uses the Gauss method to fit nonlinear functions to data. This particular program is suitable for use on a minicomputer which is utilized for real-time data acquisition and analysis. A sample problem is described which requires a nine parameter nonlinear fit to 200 data points. This problem serves both to illustrate the use of LSQFT and to compare its results with TJMAR1, a nonlinear parameter estimation program available in the Sandia MATHLIB, for the CDC 6600/7600 computers. The results of this comparison fit were identical to the significance of the input convergence criteria. The output of LSQFT includes the standard deviation of each fit parameter, which is useful for analysis of the significance of the parameters based on the input data, or for evaluation of functional models used to fit to the data. Examples are given of the types of graphical output which can be generated from the results of LSQFT. The program is documented and listed in its entirety, and a user's guide is provided.

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or approval by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

24

Table of Contents

I. Introduction	3
II. Nonlinear Least Squares Data Fitting	5
A. Mathematical Method	5
B. Program Description	7
III. Use of LSQFT	8
IV. Performance of LSQFT	10
V. Summary	14
Acknowledgements	15
References	15
Appendix: User's Guide to LSQFT	31

Figures

- Fig. 1. Listing of PROGRAM FIT, the main program which calls LSQFT.
- Fig. 2. Listing of SUBROUTINE FUNC, the user-supplied routine to evaluate the function to be fit.
- Fig. 3. Listing of LSQFT, the nonlinear least squares data fitting subroutine.
- Fig. 4. Output summary of fit from LSQFT. This output form occurs for any value of IPR. The summary shown here is for the sample problem described in the text, a nine parameter fit to 200 data points, using the functional form of two Gaussians and a quadratic polynomial.
- Fig. 5. Optional output from LSQFT when IPR = 1 or 3, giving data tabulation for each data point.
- Fig. 6. Optional output from LSQFT when IPR = 2 or 3, showing the parameter vector P(I) after each iteration. This example is from the sample problem described in the text and summarized in Fig. 4.
- Fig. 7. Sample problem described in the text. The rough curve is the data to be fit; the smooth curve is the initial guess.
- Fig. 8. The results of the fit from the sample problem. The rough curve is the data to be fit; the smooth curve through the data is the final fit from LSQFT, using the initial guess parameters displayed in Fig. 7. The other three curves show the components of the final fit: two Gaussians and a quadratic polynomial.
- Fig. 9. Scatter plot of residuals of the fit described in the text and shown in Figs. 4 and 8. The residual is the difference between dependent variable and calculated function, plotted here versus x, the independent variable.
- Fig. 10. Scatter plot of residuals of the fit to the sample problem, plotted versus fitted y, the functional value for each data point.

I. Introduction

As the minicomputer becomes increasingly commonplace in the experimental laboratory, there is a demand for software for these machines which duplicates the capabilities of much larger computers (e.g., CDC 6600 or 7600). Continuing advances in minicomputer technology, most notably increased memory size and speed at lower cost, make possible the use of sophisticated programs which are commonly restricted to the large computers. An example, which is the subject of this report, is the fitting of nonlinear functions to data. A common application of minicomputers in the experimental laboratory is real-time data acquisition. Very often one wants to extract parameters from the experimental data based on an available theoretical model or function which is nonlinear in the parameters to be estimated. It would be desirable in this case for the minicomputer to perform the data fitting and analysis functions as well as the data acquisition function.

Data fitting done by the minicomputer in the lab where the data is acquired is usually more convenient and may often be faster than processing the data by the large machines. While central processor speeds may be orders of magnitude slower on minicomputers (though these differences in speed are continually diminishing), one must consider the operator time spent in retrieving the data from the minicomputer on a software media and in a format acceptable to the large computer. One must add to this the time to transfer the data into the batch input stream, the actual processing time (large computers are heavily time-sliced), and the time to retrieve the results from the batch device. Two alternatives to this sequence are 1) a direct link between the data acquisition minicomputer and the computer which accomplishes the data fitting, and 2) data fitting done by the minicomputer. It is the latter approach which is addressed in this report.

Another advantage to data fitting with the minicomputer is real-time analysis. This feature can lead to novel control of experiments and more efficient experimentation. Long times spent in data acquisition may possibly be shortened, or real-time results may allow the operator to alter the future course of the experiment. As an example, consider an experiment where data is taken at a series of isothermal holds and several hours are required to attain each point because of long thermal equilibration times. In addition, suppose the experimenter has determined, either analytically or empirically, that the measured dependent variable approaches the value at the isotherm in a manner which can be modeled by, say, a constant plus an exponential in time. Real-time data fitting could be used to analyze the data as it is acquired and predict the final isothermal value. When the uncertainty in this predicted value is small enough, the experiment continues toward the next isotherm. In this manner, the time to perform a very long experiment could be dramatically shortened.

This report describes the FORTRAN subprogram LSQFT (least squares fit) for fitting nonlinear functions to experimental data using a least squares technique. The program is described and documented, and a program listing and user's manual are included for implementation of the routine. To illustrate the use of the program, a sample test was run which included 200 data points and a nine parameter fit. The performance of the routine using a minicomputer in fitting this sample data is compared to the same solution using the CDC 6600 and TJMAR1, the nonlinear least squares parameter estimation program in the Sandia Laboratories MATHLIB [1].

The subprogram LSQFT documented here was originally written by William H. Hardy at Princeton University in 1969, and is based on the method described by R. H. Moore and R. K. Zeigler in Los Alamos Scientific Laboratory report LA-2367, March, 1960 [2]. The present author brought it to Sandia Laboratories, where it

has been further modified for use on Hewlett Packard 1000 System minicomputers. The code, written in FORTRAN, can readily be adapted for use on other machines.

II. Nonlinear Least Squares Data Fitting

A. Mathematical Method

The program LSQFT employs the Gauss method to fit a nonlinear function to data. The following discussion provides a brief overview of the method; greater detail may be found in several references [2-6]. We will consider functions of a single independent variable, x , of the form

$$y = f(x; p_i) \quad , \quad (1)$$

where y is the dependent variable and p_i , $i = 1, 2, \dots, n$ are the n parameters to be estimated in the fitting procedure. The data to be fit are represented by the m observations (x_j, y_j) , $j = 1, 2, \dots, m$, where $m > n$. We also include weighting factors w_j associated with each y_j . The objective of the fitting procedure is to minimize the sum of the squares of the residuals, Q , or the deviations of the measured y_j from the function; thus, the method should minimize

$$Q = \sum_{j=1}^m w_j [y_j - f(x_j; p_i)]^2 \quad . \quad (2)$$

A common assumption is that the y_j are from a sample which is normally distributed about a mean for each independent variable x_j ; this leads to the definition of the weighted variance of the fit given by

$$s^2 = \frac{\sum_{j=1}^m w_j [y_j - f(x_j; p_i)]^2}{m - n} \quad . \quad (3)$$

The above definitions are common to linear and nonlinear least squares solutions. In the Gauss method for the nonlinear problem, the fitting function is linearized with respect to each of the parameters by a first order Taylor's series, and a linear least squares method is used to optimize the parameters of the fit.

The resulting linear problem can be shown to consist of the residuals $\Delta y_j = [y_j - f(x_j; p_1)]$ as the dependent variables, partial derivatives of the function with respect to the parameters $(\partial f / \partial p_1)$ as the independent variables, and Δp_1 , the increments in the parameters, are to be estimated [2]. The system of linear equations which results can be solved by a variety of methods; the program described here employs matrix inversion. In matrix notation, the problem is summarized as $\underline{B} = \underline{A}\underline{D}$, and the solution is $\underline{D} = \underline{A}^{-1}\underline{B}$. The elements of $\underline{D}$ are $D_1 =$

Δp_1 ; those of $\underline{B}$ are $B_1 = \sum_{j=1}^m w_j \Delta y_j \left(\frac{\partial f}{\partial p_1} \middle|_j \right)$, and the elements of $\underline{A}$ are given by

$$A_{1k} = A_{k1} = \sum_{j=1}^m w_j \left(\frac{\partial f}{\partial p_1} \middle|_j \right) \left(\frac{\partial f}{\partial p_k} \middle|_j \right).$$

The procedure, then, is to begin with a set of initial guesses to the parameters p_1 , and solve for the Δp_1 . This yields a new set p_1 with which to continue the process. Iteration continues until a convergence criteria is satisfied, e.g., when the values Δp_1 are less than a predetermined value. It can be shown that the elements of the inverse matrix $\underline{A}^{-1}$ yield σ_1^2 , the variance of each parameter p_1 , given by

$$\sigma_1^2 = s^2 (A^{-1})_{ii} \quad (4)$$

The technique also yields what may be called the variance of the predicted mean, defined by

$$s_i^2 = s^2 \sum_{j=1}^n \sum_{k=1}^n \left(\frac{\partial f}{\partial p_j} \right)_i \left(\frac{\partial f}{\partial p_k} \right)_i (A^{-1})_{jk} , \quad (5)$$

which is useful in prediction analysis [2].

B. Program Description

This section describes the implementation of the Gauss method described in the previous section into the FORTRAN subroutine LSQFT. Additional detail for use of the program and a listing are provided in a later section.

The data points to be fit and their weights are assumed to exist in arrays X, Y, and W. A vector PG is required for the initial guesses of the parameters. The parameter vector itself is P, and PC is the corrected parameter vector following each iteration. A separate, user-defined subroutine defines the function to be fit and the partial derivatives with respect to each parameter. The derivatives may be determined analytically or numerically. During each iteration, the matrix A and vector B are formed, the matrix is inverted, and the vector D is thereby generated (DP in program notation). The corrected parameter vector PC is generated by the algorithm

$$PC_i = P_i + H \times DP_i , \quad (6)$$

where H = 1.0 at the beginning and decreases by 0.1 every four iterations, until H = 0.1. Optionally, the user can specify that a parameter not be allowed to change sign, in which case the correction algorithm is

$$PC_i = (0.5) \times P_i / |DP_i| . \quad (7)$$

Iteration continues until $|DP_i/P_i| < \text{TEST}$, a user-defined convergence parameter, for each parameter P_i , or until ITMAX iterations have been completed. Failure of the matrix inversion routine generates a return from LSQFT with an error code.

Following a successful fit by LSQFT, the program generates a summary which includes the number of data points in the fit, the number of parameters in the fit, the value of TEST, the number of iterations, the weighted variance s^2 (Eq. 3), the unweighted sum of squares of the residuals, and the standard deviation of the fit s . For each of the i parameters, the initial guess, final fit value, and standard deviation σ_i (Eq. 4) are printed, as well as the final matrix $\underline{A}^{-1}$. Optional output includes the parameter vector after each iteration, and a tabulation of the data points, including (at each x_j) y_j , w_j , the calculated function value, the deviation between the dependent variable and the calculated function, and the standard deviation of the predicted mean s_j (Eq. 5).

III. Use of LSQFT

In order to illustrate the use of LSQFT, a sample problem was chosen which required a nine parameter fit to 200 data points. The form of the equation to fit to data points (x_j, y_j) was the sum of two Gaussians and a quadratic polynomial, given by

$$y = P_1 + P_2x + P_3x^2 + P_4 \exp[-(x - P_5)^2/2P_6^2] + P_7 \exp[-(x - P_8)^2/2P_9^2] \quad (8)$$

In addition to the main program which calls LSQFT, the user must provide a function subroutine FUNC which evaluates the function to be fit and the partial derivatives with respect to each of the fit parameters. The derivatives, returned in vector PART, may be numerically or analytically determined. All communication between the main program, FUNC, and LSQFT, is through subroutine parameter strings

or common blocks. A program listing of the sample problem is shown in Figs. 1-3, written in Hewlett Packard RTE Fortran IV.

The main program, PROGRAM FIT, is shown in Fig. 1. Lines 6-8 show the required COMMON and EXTERNAL declarations. The input data is read from disc into arrays X and Y in lines 12-22. Necessary input parameters are defined in lines 24-30. The initial guess parameter vector PG is defined in lines 31-35 prior to CALL LSQFT in line 37. The remainder of FIT is used for program timing and generation of files containing the residuals from the fit.

Figure 2 shows a listing of the subroutine FUNC for the particular functional form given by Eq. 8. Lines 67 and 68 show the required COMMON declaration. A call to FUNC with $I < 0$ generates a user-defined action -- in this case, the printing of a title, defined in lines 69-71. For $I > 0$, FUNC must return the functional value YT as a function of Z, the independent variable, and P, the current parameter vector. FUNC also returns PART, the vector of partial derivatives. These definitions are shown in lines 73-85 for the function in Eq. 8.

The FORTRAN listing of subroutine LSQFT is given in Fig. 3. Following an initialization section (to line 141), the vector B (designated BM) and matrix A are generated for each iteration in lines 145 to 161. The matrix A is inverted in lines 173 to 223 (retaining notation A), and the correction vector PC is computed (lines 227-245). Upon checking for convergence, the program either returns to iterate, or prints the desired output and returns to the main program.

The subroutine LSQFT, as shown in Fig. 3, is limited to 200 (or less) data points and a ten (or fewer) parameter fit. The user must guarantee that the number of data points exceeds the number of fit parameters. To increase the allowable number of data points, one can simply change all references to "200" in the DIMENSION and COMMON statements in lines 103-104.

The number of parameters to be fit can be increased by changing all references to "10" in lines 103-104. In addition, NSG must be dimensioned one greater than the number of parameters (line 105), and format 201 (line 106) should be changed to allow more than 10 fields of format E11.3.

A concise "User's Guide to LSQFT" containing all necessary parameter definitions, COMMON allocations, etc., is given in the Appendix.

The amount of printed output from LSQFT depends on the specification of the print parameter, IPR. The default ($IPR = \emptyset$) output is shown in Fig. 4 for the sample problem. The first line of output is generated by the user subroutine FUNC. The next several lines give a summary of the final fit to the data, followed by the list of the fit parameters, which includes initial and final values, and the standard deviation of each parameter. The final matrix $\underline{\underline{A}}^{-1}$ is also printed. For $IPR = 1$ or 3, the output shown in Fig. 5 is generated, which shows the tabulation for each of ND data points. This includes the input data, the final functional value for each independent variable, the deviation between the fit and the observed dependent variable, and the standard deviation of the predicted mean. For $IPR = 2$ or 3, the output shown in Fig. 6 results, which displays the parameter vector $P(I)$ after each of the iterations in LSQFT. This data is output during the fitting procedure, as each iteration is completed.

IV. Performance of LSQFT

The sample problem described in the preceding section, in addition to serving as an illustration in the use of LSQFT, provides a basis for comparison of the performance of LSQFT. This comparison was made between LSQFT run on a minicomputer and TJMAR1, a nonlinear parameter estimation routine, run on the Sandia CDC 6600.

The sample problem was generated as follows: two hundred "data" points, with x varying from 1 to 200 in increments of 1, were input to Eq. 8, using the following values for P_i :

$$P_1 = 0.2$$

$$P_2 = 0.001$$

$$P_3 = 1.5 \times 10^{-5}$$

$$P_4 = 3.0$$

$$P_5 = 68.0$$

$$P_6 = 13.0$$

$$P_7 = 2.0$$

$$P_8 = 100.0$$

$$P_9 = 18.0$$

In addition, "noise" was included at each point by the addition to $y(x)$ of a random number between 0.075 and -0.075. The resulting "noisy data" is shown as the rough curve in Fig. 7. This data set was used for all the comparison tests. A standard initial guess parameter vector was used for all tests, consisting of

$$P_1 = 0$$

$$P_2 = 0$$

$$P_3 = 0$$

$$P_4 = 4.0$$

$$P_5 = 60.0$$

$$P_6 = 15.0$$

$$P_7 = 1.5$$

$$P_8 = 90.$$

$$P_9 = 15.$$

These parameters yield the smooth curve in Fig. 7. An identical convergence criteria was used for the comparisons, based on $TEST = 10^{-5}$.

The minicomputer which utilized LSQFT is a Hewlett Packard 1000 System, consisting of a 21MX E-Series Processor with standard performance memory, fast FORTRAN processor firmware, and the Hewlett-Packard RTE-IVB operating system. The size of LSQFT itself takes approximately 2650 (decimal) words (16-bit) as an ASCII source subroutine; when PROGRAM FIT was loaded into the computer, the size of the entire binary program, including data, common, and all external references, required about 12 K (decimal) words of memory.

The comparison test was made using a FORTRAN program on the Sandia NOS time-sharing system, using a CDC 6600 computer. The main program made a call to TJMAR1, the nonlinear parameter estimation routine in the Sandia MATHLIB.

The results of the fit by LSQFT are shown numerically in Figs. 4 and 6, and graphically in Fig. 8. The graphical output shows the fit as the smooth curve through the noisy data; the three components of the fit (two Gaussians and a quadratic polynomial) are also shown in the figure. The goodness of the fit, besides the graphical display, is indicated by the low standard deviation of the fit, 0.043. (Recall that the peak-to-peak generated noise was 0.15.) From the standard deviations of the individual fit parameters, it can be seen that, except for $P(7)$, the fit values were all within one standard deviation of the parameters used to generate the data. All of the fit parameters from TJMAR1 differed from those in Fig. 4 by less than 4 in the sixth significant digit. Since the convergence criteria was satisfied when the fractional change in each parameter from one iteration to the next was less than 10^{-5} , the results of the fit from TJMAR1 and LSQFT were identical.

It is important to realize that a comparison test of this type makes no general statement about the performance of LSQFT versus TJMAR1. Because of the method which it uses, TJMAR1 is more likely to converge to a solution than is LSQFT, in the general case. In fact, the first order Taylor's series used by the Gauss method (LSQFT) requires that the residuals are small. For fitting to experimental data, however, the

user can generally make initial parameter estimates which are sufficiently close for LSQFT to converge on a solution. This is the case for the test problem in this report, and the resultant solution agrees with the solution provided by TJMAR1.

The same problem was also run on the Sandia CDC 7600 using NLWOOD. This program is similar to TJMAR1, but differs in one respect in that it generates, like LSQFT, a standard deviation for each fit parameter. These results from NLWOOD agreed with the parameter standard deviations from LSQFT. The parameter standard deviations can be very useful for data evaluation. If the functional form of the data is known, a large standard deviation of a particular parameter may indicate that the data is too inexact for accurate determination of the parameter, or the measured data may be insensitive to the parameter in question. For cases where the functional form of the data is not known, the parameter standard deviation may be used to decide which parameters have significance, or which are sensitive to the data.

A timing comparison was made, which consisted of the total CPU time spent in TJMAR1 or LSQFT. For TJMAR1 on the CDC 6600, the fit was accomplished in seven iterations, using 2.8 seconds of CPU time. The fit by LSQFT on the minicomputer took approximately 50 seconds and 13 iterations. (The same test was also timed using LSQFT on a Hewlett Packard 1000 System with a 2100S CPU; the fitting procedure required 87 seconds with this processor.)

Some users may be interested in reducing the program size or execution time of LSQFT. Significant reductions in size may be achieved by removing the WRITE statements for printed output, as FORTRAN formatters are typically quite sizeable. To examine the costs in time, some measurements were made on various sections of LSQFT. For the above problem (nine parameter fit to 200 data points), it was found that only 0.09 seconds was required (per iteration) for the inversion of the 9×9 parameter matrix, out of an average iteration time of $50/13 = 3.85$ seconds. A significant amount of time (3.15 seconds per iteration) was spent in the loop

between lines 150 and 161 (Fig. 3), for repeated calls to FUNC and evaluation of vector B and matrix A. The user could thus reduce fitting time by minimizing the number of data points to be fit and by simplifying functional evaluations in FUNC as much as possible. Additional time savings may of course be achieved by reducing the number of iterations, which can be done by: 1) improving the initial parameter estimate vector, and 2) relaxing the convergence criteria (using a larger value for TEST). Fig. 6 shows, for example, no change in the fourth significant digit in any parameter after the 10th iteration.

Besides a graphical display of the fit (as in Fig. 8) or an examination of the standard deviation of the fit, the goodness of fit may be examined by consideration of the randomness of the residuals of the fit. For the sample problem, Figs. 9 and 10 show the residuals plotted versus, respectively, x and fitted y , the functional value for each data point. For a good fit, one expects the residuals to be randomly positive and negative. Such a "scatter plot" can be used to quickly identify nonrandom trends in the residuals which indicate a poor fit.

There appears to be no definitive criteria for selection of the initial estimates of the parameters. The great diversity in functions which may be fit by such a routine requires that each problem be examined individually. For the sample problem described in this report, we note that fitting was successfully accomplished by using parameters which were chosen by a quick graphical examination of the input data. For other problems, only experience can dictate the required accuracy necessary for the initial parameter estimates.

V. Summary

The FORTRAN subroutine LSQFT for fitting nonlinear functions to data is described and documented, and has been found suitable for use on a minicomputer used for data acquisition and analysis. A sample problem was introduced which

required a nine parameter nonlinear fit to 200 data points; the data were fit using both LSQFT on a minicomputer and TJMAR1 on a CDC 6600. The resultant values of the fitted parameters were identical within the convergence criteria specified in the problem. The example problem and a program user's manual are included to serve as a guide in the use of LSQFT.

Acknowledgements

The author gratefully acknowledges J. Koski for running the sample test comparison using TJMAR1, E. Frost for assistance in running NLWOOD, and K. Hiebert for helpful discussions on the methods used in nonlinear least squares data fitting.

References

1. T. H. Jefferson, "TJMAR1 - A FORTRAN Subroutine for Nonlinear Least Squares Parameter Estimation," Sandia Laboratories, SLL-73-0705, March, 1974.*
2. R. H. Moore and R. K. Zeigler, "The Solution of the General Least Squares Problem with Special Reference to High-Speed Computers," Los Alamos Scientific Laboratory, LA-2367, March, 1960.
3. P. R. Bevington, Data Reduction and Error Analysis for the Physical Sciences, McGraw-Hill, NY, 1969, pp. 204-246.
4. G. Daniel and F. S. Wood, Fitting Equations to Data, Wiley-Interscience, New York, 1971.
5. W. Edwards Deming, Statistical Adjustment of Data, John Wiley and Sons, Inc., New York, 1943.
6. K. Hiebert, "A Comparison of Nonlinear Least Squares Software," SAND 79-0483, Sandia Laboratories, Albuquerque, NM, 1979.*

*Available from: National Technical Information Service, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161.


```

0001 FTN
0002 PROGRAM FIT
0003 DIMENSION IP(5),IDCB(144),NAM(3),IB(4),ISZ(2)
0004 DIMENSION ITMI(5),ITMO(5)
0005 EQUIVALENCE (XX,IB),(YY,IB(3))
0006 COMMON X(200),Y(200),PG(10),PAK(10),P(10),Z,Y1,W(200),ND,IPR,
0007 INP,TEST,LP,ITMAX,NSG(11)
0008 EXTERNAL FUNC
0009 CALL RMPAR(IP)
0010 LU=IP
0011 IF(LU.EQ.0)LU=1
0012 WRITE(LU,60)
0013 60 FORMAT("INPUT DATA FILENAME: ")
0014 READ(LU,61) NAM
0015 61 FORMAT(A2)
0016 CALL OPEN(IDCB,IER,NAM)
0017 CALL READF(IDCB,IER,IB,4)
0018 DO 10 I=1,200
0019 CALL READF(IDCB,IER,IB,4)
0020 X(I)=XX
0021 Y(I)=YY
0022 10 W(I)=1.
0023 CALL CLOSE(IDCB)
0024 ND=200
0025 ITMAX=40
0026 IPR=0
0027 TEST=1E-05
0028 LP=6
0029 NP=9
0030 NSG=0
0031 DO 20 J=1,3
0032 20 PG(J)=0.
0033 WRITE(LU,25)
0034 25 FORMAT("INPUT PG(I),I=4,9: ")
0035 READ(LU,*)PG(4),PG(5),PG(6),PG(7),PG(8),PG(9)
0036 CALL EXEC(11,ITMI)
0037 CALL LSQFT(FUNC,IER)
0038 CALL EXEC(11,ITMO)
0039 TIME=60.*(ITMO(3)-ITMI(3))+(ITMO(2)-ITMI(2))+.01*(ITMO(1)-ITMI(1))
0040 WRITE(LU,333)TIME
0041 333 FORMAT("ELAPSED TIME = "F6.3" SLC")
0042 ISZ=2
0043 NAM(1)=2H*K
0044 NAM(2)=2HES
0045 NAM(3)=2HX
0046 CALL CREAT(IDCB,IER,NAM,ISZ,3)
0047 CALL WRITEF(IDCB,IER,NAM,3)
0048 DO 800 I=1,200
0049 XX=X(I)
0050 Z=XX
0051 CALL FUNC(I)
0052 YY=(Y(I)-Y1)*100.
0053 800 CALL WRITEF(IDCB,IER,IB,4)
0054 CALL CLOSE(IDCB)
0055 NAM(3)=2HY

```

Fig. 1. Listing of PROGRAM FIT, the main program which calls LSQFT.

```
0056      CALL CREAT(IDC8,IER,NAM,ISZ,3)
0057      CALL WRITF(IDC8,IER,NAM,3)
0058      DO 801 I=1,200
0059      Z=X(I)
0060      CALL FUNC(I)
0061      YY=(Y(I)-YT)*100.
0062      XX=YT
0063 801   CALL WRITF(IDC8,IER,IH,4)
0064      CALL CLOSE(IDC8)
0065      END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00621 COMMON = 01282

Figure 1 (con't)

```

0066      SUBROUTINE FUNC(I)
0067      COMMON X(200),Y(200),PG(10),PART(10),P(10),Z,YT,W(200),ND,IPR,
0068      INP,TEST,LP,ITMAX,NSG(11)
0069      10  FORMAT(1H1" THE FUNCTION IS A QUADRATIC + TWO GAUSSIANS"/)
0070      IF(I)1,5,3
0071      1  WRITE(LP,10)
0072      GO TO 5
0073      3  EX1=P(4)*EXP(-(Z-P(5))**2/(2.*P(6)**2))
0074      EX2=P(7)*EXP(-(Z-P(8))**2/(2.*P(9)**2))
0075      Y1=P(1)+P(2)*Z+P(3)*Z**2+EX1
0076      YT=YT+EX2
0077      PART(1)=1.
0078      PART(2)=Z
0079      PART(3)=Z**2
0080      PART(4)=EX1/P(4)
0081      PART(7)=EX2/P(7)
0082      PART(5)=EX1*(Z-P(5))/P(6)**2
0083      PART(8)=EX2*(Z-P(8))/P(9)**2
0084      PART(6)=EX1*(Z-P(5))**2/P(6)**3
0085      PART(9)=EX2*(Z-P(8))**2/P(9)**3
0086      5  RETURN
0087      END

```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00388 COMMON = 01282

Fig. 2. Listing of SUBROUTINE FUNC, the user-supplied routine to evaluate the function to be fit.

```

0088 C
0089 C *****
0090 C
0091 C SUBROUTINE LSQFT(FUNC,ILRR)
0092 C
0093 C COMPUTES LEAST-SQUARES FIT TO DATA FOR NON-LINEAR FUNCTIONS.
0094 C FUNCTIONS AND DERIVATIVES WITH RESPECT TO THE PARAMETERS MUST BE
0095 C USER-SUPPLIED. BASED ON THE GAUSS METHOD. FOR MORE INFORMATION,
0096 C CONTACT W. DROTNING.
0097 C
0098 C LATEST REVISION: 791214 WDD
0099 C
0100 C *****
0101 C
0102 C SUBROUTINE LSQFT(FUNC,IERR)
0103 C DIMENSION A(10,10),BM(10),DP(10),PC(10),TEMP(10),STORL(200)
0104 C COMMON X(200),Y(200),PG(10),PART(10),P(10),Z,YLOW(200),ND,IPE,
0105 C INP,TEST,LP,ITMAX,NSG(11)
0106 C 201 FORMAT(1H,13,2X,1P10E11.3)
0107 C 202 FORMAT(1H1,20X" PARAMETERS BY ITERATION"// " ITER"20X"P(1)"//)
0108 C 207 FORMAT(/" NO. ITERATIONS "15/)
0109 C 208 FORMAT(/" NOTE: IT=ITMAX; ITERATION STOPPED"/)
0110 C 218 FORMAT(1H,10X"MATRIX INVERSION FAILED--MATRIX SINGULAR"//
0111 C 1" PARAMETER",16X,"MATRIX"//)
0112 C 217 FORMAT(1H,10("X"), " LSQFT SUMMARY "10("X")//
0113 C 1" NUMBER OF DATA POINTS "14/" NUMBER OF PARAMETERS "14
0114 C 2/" TEST "1PE9.2/" WEIGHTED VARIANCE "1PE12.4
0115 C 3/" UNWEIGHTED SUM OF SQUARES OF DEVIATIONS "1PE12.3
0116 C 4/" STD.DEV. OF FIT "1PE12.4)
0117 C 224 FORMAT(/" PARAMETER"5X"INITIAL GUESS"11X"FINAL"11X"SID, DEV."//)
0118 C 225 FORMAT(5X,12,7X,1PE12.3,3X,1PE14.7,6X,1PE10.3)
0119 C 234 FORMAT(1H1,18X"INDEPENDENT DEPENDENT CALCULATED"13X"SID,DEV."
0120 C 1"OF"/4X"1"5X"WEIGHT VARIABLE VARIABLE FUNCTION "
0121 C 2"DEVIATION PRED. MEAN"/)
0122 C 246 FORMAT(1H,14,1P6E12.3)
0123 C 205 FORMAT(10/" MATRIX A(I,J)"//)
0124 C
0125 C INITIALIZATION
0126 C
0127 C ITS=0
0128 C IERR=0
0129 C VAR=0.0
0130 C LIE=0
0131 C SS=0.0
0132 C LIF=0
0133 C NIT=0
0134 C H=1.0
0135 C DO 401 I=1,NP
0136 C DP(I)=0.0
0137 C PC(I)=PG(I)
0138 C 401 P(I)=PG(I)
0139 C LICK=0
0140 C IF=0
0141 C IF(IPR.GE.2)WRITE(LP,202)
0142 C

```

Fig. 3. Listing of LSQFT, the nonlinear least squares data fitting subroutine.

```

0143 C      RETURN TO ITERATE
0144 C
0145 502 DO 504 I=1,NP
0146      BM(I)=0.
0147      DO 504 J=1,I
0148 504 A(I,J)=0.0
0149      IT=IT+1
0150 600 DO 612 L=1,ND
0151      Z=X(L)
0152      CALL FUNC(L)
0153      DY=Y(L)-YT
0154      VAR=VAR+W(L)*DY**2
0155      SS=SS+DY**2
0156      IF(LIE.EQ.1)GO TO 612
0157      DO 611 I=1,NP
0158      BM(I)=BM(I)+PART(I)*DY*W(L)
0159      DO 611 J=1,I
0160 611 A(I,J)=A(I,J)+PART(I)*PART(J)*W(L)
0161 612 CONTINUE
0162      IF(LIE.EQ.1)GO TO 131
0163      IF(IPR.GE.2)WRITE(LP,201)IT,(P(1),I=1,NP)
0164      NIT=NIT+1
0165      IF(NIT.EQ.4)GO TO 270
0166      GO TO 275
0167 270 NIT=0
0168      H=H-0.1
0169      IF(H.LT.0.05)H=0.1
0170 C
0171 C      MATRIX INVERSION
0172 C
0173 275 R=1.0/SQRT(A(1,1))
0174      A(1,1)=R
0175      SIZE=0.0
0176      DO 810 JACK=1,NP
0177 810 SIZE = SIZE+A(JACK,JACK)
0178      SIZE=SIZE*1.0E-14/NP
0179      SIZE=MAX1(SIZE,1.0E-30)
0180      DO 802 I=2,NP
0181 802 A(I,1)=R*A(I,1)
0182      DO 806 J=2,NP
0183      DO 803 K=2,J
0184 803 A(J,J)=A(J,J)-A(J,K-1)**2
0185 C
0186 C      TEST FOR MATRIX INVERSION FAILURE
0187 C
0188      IF(A(J,J).LT.SIZE)GO TO 908
0189      R=1.0/SQRT(A(J,J))
0190      A(J,J)=R
0191      IF(J.EQ.NP)GO TO 806
0192      L=NP-1
0193      DO 805 I=J,L
0194      DO 804 K=2,J
0195 804 A(I+1,J)=A(I+1,J)-A(I+1,K-1)*A(J,K-1)
0196 805 A(I+1,J)=R*A(I+1,J)
0197 806 CONTINUE

```

Fig. 3 (con't)

```

0198      DO 903 J=2,NP
0199      L=J-1
0200      I=J-1
0201  901  R=0.0
0202      DO 902 K=1,L
0203  902  R=R+A(K+1,I)*A(K+1,J)
0204      A(I,J)=-R*A(I,1)
0205      I=I-1
0206      IF (I.GT.0)GO TO 901
0207  903  CONTINUE
0208      DO 905 J=1,NP
0209      DO 905 I=J,NP
0210      R=0.0
0211      DO 904 K=I,NP
0212  904  R=R+A(J,K)*A(I,K)
0213      A(I,J)=R
0214  905  A(J,I)=R
0215      GO TO 910
0216  C
0217  C      MATRIX INVERSION FAILED. RETURN TO MAIN
0218  C
0219  908  WRITE(LP,218)
0220      DO 911 J=1,NP
0221  911  WRITE(LP,201)J,(A(J,I),I=1,NP)
0222      IERR=1
0223      RETURN
0224  C
0225  C      CONTINUE
0226  C
0227  910  DO 102 I=1,NP
0228      TEMP(I)=0.0
0229      DO 102 J=1,NP
0230  102  TEMP(I)=TEMP(I)+A(I,J)*BM(J)
0231  C
0232  C      CALCULATE CORRECTION VECTOR PC(1)
0233  C
0234      DO 117 I=1,NP
0235      DP(I)=TEMP(I)
0236      PC(I)=P(I)+H*DP(I)
0237  117  CONTINUE
0238  C
0239  C      CALCULATE PC FOR NON-SIGN-CHANGING PARAMETERS
0240  C
0241      IF (NSG(1).EQ.0)GO TO 400
0242      DO 400 I=1,NSG(1)
0243      IF (PC(NSG(I+1))*P(NSG(I+1)).GE.0.0)GO TO 400
0244      PC(NSG(I+1))=0.5*P(NSG(I+1))/ABS(DP(NSG(I+1)))
0245  400  CONTINUE
0246      IF (L11.EQ.1)GO TO 126
0247      JERK=0
0248      DO 124 I=1,NP
0249      IF (P(I).EQ.0.0)GO TO 123
0250      IF (ABS(DP(I)/P(I))-TEST.GT.0.0)GO TO 126
0251      GO TO 124
0252  123  JLRK=JERK+1

```

Fig. 3 (con't)


```

0253 124 CONTINUE
0254 IF(JERK-NP.GE.0)GO TO 126
0255 C
0256 C CONVERGENCE TEST SATISFIED
0257 C
0258 125 LIT=1
0259 126 IF(ITS.EQ.ITMAX)GO TO 128
0260 DO 127 I=1,NP
0261 127 P(I)=PC(I)
0262 128 IF(LICK.EQ.1)GO TO 130
0263 IF(LIT.EQ.1)LICK=1
0264 IF(LIT.EQ.1.OR.IT-ITMAX+1.NE.0)GO TO 502
0265 C
0266 C FITTING COMPLETED
0267 C
0268 IT=0
0269 ITS=ITMAX
0270 GO TO 125
0271 130 VAR=0.0
0272 SS=0.0
0273 LIE=1
0274 GO TO 600
0275 131 IDF=ND-NP
0276 DF=IDF
0277 WVAR=VAR/DF
0278 I=-1
0279 CALL FUNC(I)
0280 C
0281 C PRINT SUMMARY
0282 C
0283 SDEV=SQRT(WVAR)
0284 WRITE(LP,217)ND,NP,TEST,WVAR,SS,SDEV
0285 IF(ITS.EQ.0)GO TO 134
0286 IT=ITS
0287 134 WRITE(LP,207)IT
0288 IF(IT.EQ.ITMAX)WRITE(LP,208)
0289 WRITE(LP,224)
0290 DO 223 I=1,NP
0291 SIG=SQRT(A(I,I)*WVAR)
0292 223 WRITE(LP,225)I,PG(I),P(I),SIG
0293 WRITE(LP,205)
0294 DO 251 J=1,NP
0295 251 WRITE(LP,201)J,(A(J,I),I=1,NP)
0296 IF(IPR.NE.1.AND.IPR.NE.3)GO TO 222
0297 WRITE(LP,234)
0298 222 DO 254 I=1,ND
0299 Z=X(I)
0300 CALL FUNC(I)
0301 AAA=0.0
0302 DY=(Y(I)-YT)
0303 STORE(I)=DY*W(I)
0304 DO 243 J=1,NP
0305 DO 243 JJ=1,NP
0306 243 AAA=AAA+PART(J)*PART(JJ)*A(J,JJ)
0307 AX=SQRT(AAA*WVAR)

```

Fig. 3 (con't)

PAGE 0008 LSQFT 10:06 AM FRI, 11 JAN., 1980

```
0308      IF(IPR.NE.1.AND.IPR.NE.3)GO TO 254
0309      WRITE(LP,246)I,W(I),X(I),Y(I),YT,OY,AX
0310  254  CONTINUE
0311      RETURN
0312      END
```

FTN4 COMPILER: HF92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 02642 COMMON = 01282

Fig. 3 (con't)

THE FUNCTION IS A QUADRATIC + TWO GAUSSIANS

***** LSQFT SUMMARY *****

NUMBER OF DATA POINTS 200

NUMBER OF PARAMETERS 9

TEST 1.00E-05

WEIGHTED VARIANCE 1.8818E-04

UNWEIGHTED SUM OF SQUARES OF DEVIATIONS 3.594E-01

STD. DEV. OF FIT 4.3380E-02

NO. ITERATIONS 13

PARAMETER	INITIAL GUESS	FINAL	STD. DEV.
1	0.000E+00	1.8839651E-01	1.115E-02
2	0.000E+00	1.038222E-03	4.488E-04
3	0.000E+00	1.4784198E-05	2.257E-06
4	4.000E+00	2.5921842E+00	5.215E-02
5	6.000E+01	6.7929733E+01	2.132E-01
6	1.500E+01	1.2901661E+01	1.316E-01
7	1.500E+00	2.6321722E+00	2.655E-02
8	9.000E+01	9.9743820E+01	5.470E-01
9	1.500E+01	1.8024463E+01	3.883E-01

MATRIX A(I,J)

1	6.622E-02	-1.997E-03	9.072E-06	-7.283E-02	-3.014E-01	-1.465E-01	6.289E-02	-6.838E-01	8.098E-01
2	-1.997E-03	1.061E-04	-5.310E-07	2.537E-03	1.549E-02	-1.855E-02	-3.537E-03	2.462E-02	-4.445E-02
3	9.072E-06	-5.310E-07	2.708E-09	-1.002E-05	-7.358E-05	7.343E-06	1.740E-05	-1.077E-04	2.138E-04
4	-7.283E-02	2.537E-03	-1.082E-05	1.445E+00	5.489E+00	2.637E+00	-5.298E-01	1.452E+01	-9.735E+00
5	-3.014E-01	1.549E-02	-7.358E-05	5.489E+00	2.530E+01	1.187E+01	-2.629E+00	5.992E+01	-3.542E+01
6	-1.465E-01	-1.855E-02	7.343E-06	2.637E+00	1.187E+01	9.206E+00	-1.244E+00	5.160E+01	-1.660E+01
7	6.289E-02	-3.537E-03	1.740E-05	-5.298E-01	-2.629E+00	-1.244E+00	3.749E+01	-5.992E+00	4.166E+00
8	-6.838E-01	2.462E-02	-1.077E-04	1.452E+01	5.992E+01	2.160E+01	-5.992E+00	1.590E+02	-9.884E+01
9	8.098E-01	-4.445E-02	2.138E-04	-9.735E+00	-3.942E+01	-1.660E+01	4.166E+00	-9.884E+01	8.032E+01

Fig. 4. Output summary of fit from LSQFT. This output form occurs for any

value of IPR. The summary shown here is for the sample problem

described in the text, a nine parameter fit to 200 data points, using

the functional form of two Gaussians and a quadratic polynomial.

1	WEIGHT	INDEPENDENT VARIABLE	DEPENDENT VARIABLE	CALCULATED FUNCTION	DEVIATION	STD. DEV. OF PRED. MEAN
1	1.000E+00	1.000E+00	2.010E-01	1.895E-01	1.157E-02	1.083E-02
2	1.000E+00	2.000E+00	1.407E-01	1.905E-01	-4.983E-02	1.051E-02
3	1.000E+00	3.000E+00	1.327E-01	1.917E-01	-5.895E-02	1.020E-02
4	1.000E+00	4.000E+00	2.066E-01	1.920E-01	1.381E-02	9.908E-03
5	1.000E+00	5.000E+00	1.529E-01	1.940E-01	-4.113E-02	9.625E-03
6	1.000E+00	6.000E+00	1.700E-01	1.952E-01	-2.519E-02	9.354E-03
7	1.000E+00	7.000E+00	2.507E-01	1.965E-01	5.445E-02	9.093E-03
8	1.000E+00	8.000E+00	2.419E-01	1.977E-01	4.417E-02	8.856E-03
9	1.000E+00	9.000E+00	1.441E-01	1.971E-01	-5.476E-02	8.623E-03
10	1.000E+00	1.000E+01	2.029E-01	2.004E-01	2.462E-03	8.416E-03
11	1.000E+00	1.100E+01	2.875E-01	2.018E-01	8.564E-02	8.217E-03
12	1.000E+00	1.200E+01	1.827E-01	2.033E-01	-2.054E-02	8.038E-03
13	1.000E+00	1.300E+01	2.490E-01	2.048E-01	4.420E-02	7.872E-03
14	1.000E+00	1.400E+01	2.272E-01	2.064E-01	2.080E-02	7.723E-03
15	1.000E+00	1.500E+01	1.982E-01	2.080E-01	-9.820E-03	7.590E-03
16	1.000E+00	1.600E+01	1.700E-01	2.098E-01	-3.984E-02	7.471E-03
17	1.000E+00	1.700E+01	2.592E-01	2.117E-01	4.754E-02	7.363E-03
18	1.000E+00	1.800E+01	1.996E-01	2.137E-01	-1.408E-02	7.279E-03
19	1.000E+00	1.900E+01	3.018E-01	2.159E-01	8.591E-02	7.202E-03
20	1.000E+00	2.000E+01	1.584E-01	2.183E-01	-5.992E-02	7.137E-03
21	1.000E+00	2.100E+01	2.494E-01	2.210E-01	2.840E-02	7.082E-03
22	1.000E+00	2.200E+01	2.796E-01	2.240E-01	5.557E-02	7.035E-03
23	1.000E+00	2.300E+01	2.285E-01	2.274E-01	1.053E-03	6.995E-03
24	1.000E+00	2.400E+01	2.714E-01	2.314E-01	4.005E-02	6.959E-03
25	1.000E+00	2.500E+01	1.792E-01	2.359E-01	-5.677E-02	6.926E-03
26	1.000E+00	2.600E+01	1.855E-01	2.413E-01	-5.576E-02	6.895E-03
27	1.000E+00	2.700E+01	2.493E-01	2.475E-01	1.811E-03	6.865E-03
28	1.000E+00	2.800E+01	2.186E-01	2.549E-01	-3.628E-02	6.836E-03
29	1.000E+00	2.900E+01	2.149E-01	2.636E-01	-4.874E-02	6.810E-03
30	1.000E+00	3.000E+01	2.644E-01	2.740E-01	-9.637E-03	6.788E-03
31	1.000E+00	3.100E+01	3.011E-01	2.862E-01	1.481E-02	6.775E-03
32	1.000E+00	3.200E+01	3.430E-01	3.008E-01	4.223E-02	6.776E-03
33	1.000E+00	3.300E+01	3.149E-01	3.179E-01	-2.940E-03	6.793E-03
34	1.000E+00	3.400E+01	3.358E-01	3.380E-01	-2.250E-03	6.850E-03

etc.

Fig. 5. Optional output from LSQFT when IPR = 1 or 3, giving data tabulation for each data point.

PARAMETERS BY ITERATION										
ITER	P(I)									
1	0.000E+00	0.000E+00	0.000E+00	4.000E+00	6.000E+01	1.500E+01	1.500E+00	9.000E+01	1.500E+01	
2	1.797E-01	4.434E-03	-3.373E-06	1.639E+00	6.324E+01	1.463E+01	1.856E+00	9.052E+01	2.672E+01	
3	2.251E-01	-1.107E-03	2.519E-05	1.577E+00	6.761E+01	9.808E+00	2.500E+00	8.715E+01	2.497E+01	
4	2.085E-01	-2.526E-03	3.345E-05	1.965E+00	6.642E+01	1.136E+01	2.430E+00	9.068E+01	2.504E+01	
5	2.217E-01	-9.648E-04	2.465E-05	2.293E+00	6.627E+01	1.166E+01	2.312E+00	9.272E+01	2.297E+01	
6	2.211E-01	7.506E-04	1.541E-05	2.787E+00	6.670E+01	1.290E+01	2.058E+00	9.777E+01	1.957E+01	
7	1.911E-01	1.089E-03	1.463E-05	3.009E+00	6.790E+01	1.300E+01	2.015E+00	9.996E+01	1.801E+01	
8	1.887E-01	1.044E-03	1.495E-05	2.994E+00	6.793E+01	1.291E+01	2.030E+00	9.976E+01	1.809E+01	
9	1.885E-01	1.040E-03	1.498E-05	2.992E+00	6.793E+01	1.290E+01	2.032E+00	9.975E+01	1.809E+01	
10	1.884E-01	1.039E-03	1.498E-05	2.992E+00	6.793E+01	1.290E+01	2.032E+00	9.974E+01	1.809E+01	
11	1.884E-01	1.039E-03	1.498E-05	2.992E+00	6.793E+01	1.290E+01	2.032E+00	9.974E+01	1.809E+01	
12	1.884E-01	1.039E-03	1.498E-05	2.992E+00	6.793E+01	1.290E+01	2.032E+00	9.974E+01	1.809E+01	
13	1.884E-01	1.039E-03	1.498E-05	2.992E+00	6.793E+01	1.290E+01	2.032E+00	9.974E+01	1.809E+01	

Fig. 6. Optional output from LSQFT when IPR = 2 or 3, showing the parameter vector P(I) after each iteration. This example is from the sample problem described in the text and summarized in Fig. 4.

Fig. 7. Sample problem described in the text. The rough curve is the data to be fit; the smooth curve is the initial guess.

Fig. 8. The results of the fit from the sample problem. The rough curve is the data to be fit; the smooth curve through the data is the final fit from LSQFT, using the initial guess parameters displayed in Fig. 7. The other three curves show the components of the final fit: two Gaussians and a quadratic polynomial.

Fig. 9. Scatter plot of residuals of the fit described in the text and shown in Figs. 4 and 8. The residual is the difference between dependent variable and calculated function, plotted here versus x , the independent variable.

Fig. 10. Scatter plot of residuals of the fit to the sample problem, plotted versus fitted y , the functional value for each data point.

APPENDIX

USER'S GUIDE TO LSQFT

A. Usage:

CALL LSQFT (FUNC, IERR)

Parameter FUNC must be declared EXTERNAL. All additional communication between MAIN and SUBROUTINE LSQFT is through COMMON.

B. Required Subroutines:

SUBROUTINE FUNC (I)

Communicates with MAIN and SUBROUTINE LSQFT through COMMON.

C. Description of Parameters:

X - Vector of independent variables of the data points (dimension ND).

Y - Vector of dependent variables of the data points (dimension ND).

W - Vector of weights for the data points (dimension ND).

PG - Vector of initial guesses for the parameter vector P (dimension NP).

P - Vector of resultant parameters of the fit (dimension NP).

FUNC - User-supplied subroutine which evaluates the function to be fit at each of ND data points, and the partial derivatives of the function with respect to the parameters to be found.

Z - Working independent variable passed to FUNC.

YT - Working dependent variable returned from FUNC, i.e., $YT = f(Z)$, where $f(Z)$ is the function to be fit to the data. YT is therefore user-supplied in FUNC.

PART - Vector of partial derivatives of the function to be fit with respect to the parameters to be found. PART is user-defined in SUBROUTINE FUNC. (dimension NP)

ND - Number of data points in the fit ($ND \leq 200$).

NP - Number of parameters in the fit ($NP \leq 10$).

TEST:- Convergence criteria for the fit. When the absolute value of the fractional change of $P(I)$ from one iteration to the next is less than TEST for each parameter I ($I = 1, NP$), iteration terminates, and the fitting is concluded.

IPR - Parameter which controls printed output.

IPR = 0 Summary only

IPR = 1 Summary + tabulation at each data point

IPR = 2 Summary + new parameter vector P at each iteration

IPR = 3 Combination of IPR = 1 and 2

LP - Device designation for output (e.g. line printer).

ITMAX - Maximum number of iterations allowed during fitting.

NSG - Vector (dimension NSG (1) + 1) which defines number and designation of parameters whose sign is not allowed to change during the fit. NSG(1) = number of parameters with no sign change allowed. NSG(I), (I = 2, 3, . . ., NSG(1), NSG(1) + 1) contains the indices of the parameters with no sign change allowed. If not used, NSG(1) = 0.

IERR - Error code on return from LSQFT.

IERR = 0 No error
IERR = 1 Matrix inversion failure

D. COMMON Declaration:

COMMON X(200), Y(200/), PG(10), PART (10), P(10), Z, YT, W(200), ND, IPR,
NP, TEST, LP, ITMAX, NSG(11)

This declaration must appear in both MAIN and SUBROUTINE FUNC, exactly as shown.

E. Input to LSQFT:

Prior to CALL LSQFT, MAIN must supply vectors X, Y, W, PG, and NSG; and parameters ND, NP, TEST, IPR, LP, and ITMAX.

F. Output from LSQFT:

IERR and vector P are returned following CALL LSQFT. Additional output depends on IPR.

G. Description of SUBROUTINE FUNC

1. Single integer dummy argument:

SUBROUTINE FUNC(I)

2. Include COMMON block declaration.

3. Action of subroutine when called:

I = -1 -- User-defined action on completion of fitting. For example, printout of a title.

I = 0 -- No action; return.

I > 0 -- Evaluate YT(Z), i.e., the functional value for Z = X(I). Also evaluate PART (J), J = 1, NP, at Z = X(I). The current parameter vector P is used. PART may be analytically or numerically determined.

H. Remarks:

1. User must insure that ND - NP > 0.
2. Output requires (11*NP + 5) or 76 printing columns, whichever is greater.

Distribution:

Mr. Larry B. Smith
Hewlett Packard
Neely Sales Div.
11300 Lomas Blvd., NE
Albuquerque, NM 87112

1521 W. W. Shurtleff
1521 J. L. Mortley
1523 R. E. Trellue
1523 B. T. Fox
2353 D. J. Mitchell
4242 K. W. Bieg
4731 C. E. Tyner
5640 G. J. Simmons
5642 K. L. Hiebert (3)
5800 R. S. Claassen
 Attn: 5810 R. G. Kepler
 5830 M. J. Davis
 5840 N. J. Magnani
5820 R. E. Whan
 Attn: 5821 N. E. Brown
 5822 K. H. Eckelmeyer
 5823 J. A. Borders
5822 W. F. Chambers
5823 G. C. Nelson
5824 J. N. Sweet
5824 P. E. Havey (4)
5824 E. P. Roth (4)
5824 J. A. Koski
5824 G. M. Haseman
5824 W. D. Drotning (4)
8330 G. W. Anderson
8332 T. H. Jefferson
3141 Tech. Library (5)
3151 W. L. Garner (3)
 for DOE/TIC
DOE/TIC (25)
 (R. P. Campbell, 3172-3)
8266 E. A. Aas (1)

3/51

* Recipient must initial on classified documents.