ISSN 0917-8015

RESEARCH INSTITUTE FOR
ADVANCED SCIENCE AND TECHNOLOGY
OSAKA PREFECTURE UNIVERSITY
TECHNICAL REPORT No. 2

ALESQ, a Nonlinear Least-Squares Fit Code, and TSOLVE, a Nonlinear Best-Approximation Code

Second Revised Edition*

Tatsuo Tabata Rinsuke Ito

March 1997

RESEARCH INSTITUTE FOR ADVANCED SCIENCE AND TECHNOLOGY
OSAKA PREFECTURE UNIVERSITY
1-2 GAKUEN-CHO, SAKAI, OSAKA 593, JAPAN

---

* First edition was published in 1984 as Radiation Center of Osaka Prefecture Technical Report No. 4 authored by R. Ito and T. Tabata.

Research Institute for Advanced Science and Technology Osaka Prefecture University Technical Reports (RIAST-OPU-TR) [formerly Research Institute for Advanced Science and Technology University of Osaka Prefecture Technical Reports (RIAST-UOP-TR)] are issued irregularly. Copies are available upon request to the author(s) of each issue. Requests from libraries for continued supply are welcome; such requests should be addressed to Annual Report Committee, RIAST, OPU. The cost for subscription is free, but libraries of universities and institutes are requested to send similar publications, if possible, to the RIAST library in exchange.

RIAST was established in 1990 by incorporating the former Osaka Prefectural Radiation Research Institute (OPRRI). OPRRI Technical Reports were therefore discontinued with No. 11 (1990).

RIAST-OPU-TR 2
ALESQ, a Nonlinear Least-Squares Fit Code, and TSOLVE, a Nonlinear Best-Approximation Code / Tatsuo Tabata and Rinsuke Ito.

Request for a copy of this issue should be addressed to the first author.

Abstract

Algorithms for nonlinear least-squares fit and best approximation are described, and FORTRAN codes for these algorithms are given. The algorithm used for the least-squares fit is based on the maximum neighborhood algorithm developed by Levenberg and Marquardt; an improvement in strategy proposed by the present authors has been incorporated. The code for this algorithm is called ALESQ. For fitting the equation to the data with approximately equal relative uncertainties, a two-step method is proposed, and conveniences for this method are provided in ALESQ. The algorithm used for the best approximation is basically the one developed by Osborne and Watson; a method to obtain the solution for the continuous best-approximation problem with a discrete set of data has been incorporated. The code for this algorithm is called TSOLVE. A routine to estimate tolerances for roundoff of the solution has been developed and included in TSOLVE. Both ALESQ and TSOLVE can also be used for linear problems. The codes published originally in 1984 assumed the use of main-frame computers, and those in the present edition have been modified for the use on desk-top computers with some additional improvements. In Appendix I, procedures and methods to formulate empirical equations are discussed. In Appendix II, empirical and approximate formulas and other applications for which the codes ALESQ and TSOLVE were used are listed; the results are mostly related to the field of radiation physics.

Contents

1 Introduction

In various branches of science and technology the following problems frequently arise; (1) to determine values of parameters in a theoretical equation by fitting it to data; (2) to represent a mass of data by an empirical equation; and (3) to express results of laborious computation or a complicated mathematical function by a simple approximation. These problems are in most cases reduced to the problem of least-squares fit or the problem of best approximation in Tchebychev's sense. Coupled with the development of high-speed digital computers, effective algorithms have been devised to solve these problems.

The problem of least-squares fit is a special case of the problems of optimization, and is also one of the problems of data analysis; a number of monographs and text books are available on these problems (Ko68, Be69, Be70, Da71, Fl80, Gi81). Books have also been published on the best approximation (Ha55, Ri64, Ch66, Me67) However, the text books that describe algorithms of best approximation developed most recently are yet scarce.

Since the late 1960s we have been working to formulate empirical equations and approximations in the field of radiation physics using codes for least-squares fit and best approximation. In designing the codes, we have developed new methods and techniques to be incorporated into or to be jointly used with the algorithms adopted. This report gives description of these developments along with the explanation of the algorithms, and presents the FORTRAN codes for the algorithms.

In Sec. 2 terminology common to the two algorithms are given. Section 3 deals with an algorithm for nonlinear least-squares fit. It is based on the maximum-neighborhood algorithm developed by Levenberg (Le44) and Marquardt (Ma63) and technically modified by the present authors* (Ta75a). In Sec. 4 an algorithm for nonlinear best-approximation is described. It is basically the algorithm developed by Osborne and Watson (Os69), and utilizes the linear-programming technique in solving the linearized problem (Ke59, Os67). In Secs. 3 and 4 an outline of the codes for these algorithms are given. The FORTRAN codes and the results of sample runs are attached at the end of this report.

---

* The strategy proposed by the present authors was implemented on a Tektronix Graphic System [see for example: A. Campagnaro, A. Mantovani and P. Ugualiati, Inorganica Chimica Acta 99 L15 (1985)].

2 General Description

2.1 Terminology

The algorithms to be described are concerned with the minimization of the function E that represents a criterion of optimization. The function E is a measure of the error or the differences between the data y_i (i=1, 2, ..., n) and the values of the expression y. The expression y, in turn, is a function of independent variables and the parameters to be adjusted to minimize E. We denote the independent variables by $\mathbf{x}=(x_1, x_2, ..., x_m)^T$ and the parameters by $\mathbf{a}=(a_1, a_2, ..., a_p)^T$. (Here $\mathbf{x}$, for example, refers to a column vector; superscript T refers to transportation so that $\mathbf{x}^T$ refers to a row vector.) The values of the variables $\mathbf{x}$ corresponding to the ith data y_i is written as $\mathbf{x}_i=(x_{1i}, x_{2i}, ..., x_{mi})^T$.

In the nonlinear minimization problems treated here iterative procedure is used. For each iteration we have starting values $\mathbf{a}_0$ of the parameters. Except the first iteration, the values $\mathbf{a}_0$ are the best result up to the latest iteration. The optimum values of the increments $\delta\mathbf{a}$ of the parameters are sought in each iteration to obtain the optimum values $\mathbf{a}$ of the parameters, i.e.,

$$\mathbf{a} = \mathbf{a}_0 + \delta\mathbf{a}, \tag{1}$$

To linearize the problem we expand the function y to the first order in Taylor's series as a function of $\delta\mathbf{a}$, and write

$$y = y_0 + \delta\mathbf{a}^T\nabla y_0, \tag{2}$$

where y_0 is the value of y at $\mathbf{a}_0$, and ∇ denotes the gradient operator $(\partial/\partial a_1, \partial/\partial a_2, ..., \partial/\partial a_p)^T$.

2.2 Relation between ALESQ and TSOLVE

A good estimate of the starting values for a best approximation problem can usually be obtained by solving the same problem with the criterion of least-squares fit. Therefore, the code ALESQ for least-squares fit and the code TSOLVE for best approximation have been designed to use the same format for input variables and similar statements for the fitting and approximating functions. One of the subprogram called PUT is used in common.

3 ALESQ: An Algorithm for Nonlinear Least-Squares Fit

3.1 Formulation of the Problem

In the least-squares fit the error criterion E is given by

$$E = \sum_{i=1}^{n} w_i \left[y_i - y(\mathbf{x}_i; \mathbf{a}) \right]^2, \tag{3}$$

where w_i is the weight for the ith data. To the approximation of y given by eq. (2), E is expressed as

$$E = \sum_{i=1}^{n} w_i \left[y_i - y_0(\mathbf{x}_i) - \delta\mathbf{a}^{\mathrm{T}} \nabla y_0(\mathbf{x}_i) \right]^2. \tag{4}$$

Minimization of E in the space of parameter increments $\delta\mathbf{a}$ is made by setting the first derivatives of E with respect to the components of $\delta\mathbf{a}$ equal to zero. Then we have a set of p simultaneous equations

$$\boldsymbol{\beta} = \mathbf{A}\delta\mathbf{a}, \tag{5}$$

where $\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta} = \sum_{i=1}^{n} w_i \left[y_i - y_0(\mathbf{x}_i) \right] \nabla y_0(\mathbf{x}_i) \tag{6}$$

and $\mathbf{A}$ is the p×p matrix whose components A_{jk} are given by

$$A_{jk} = \sum_{i=1}^{n} w_i \frac{\partial y_0(\mathbf{x}_i)}{\partial a_j} \frac{\partial y_0(\mathbf{x}_i)}{\partial a_k}. \tag{7}$$

When the determinant of $\mathbf{A}$ is not equal to zero, the solution to eq. (5) can be written as

$$\delta\mathbf{a} = \mathbf{C}\boldsymbol{\beta} \tag{8}$$

where $\mathbf{C}=\mathbf{A}^{-1}$ is the inverse of $\mathbf{A}$. The matrix $\mathbf{A}$ is called the curvature matrix, and $\mathbf{C}$ the error matrix.

The iterative method that uses eq. (8) is called Gauss-Newton method. Another method relevant to the present algorithm is the steepest-descent method, in which the increments $\delta\mathbf{a}$ are chosen as follows:

$$\delta\mathbf{a} \propto -\nabla E = 2\boldsymbol{\beta} \tag{9}$$

3.2 Description of the Algorithm

3.2.1 The Maximum-Neighborhood Algorithm

The algorithm we use is the maximum-neighborhood algorithm developed by Levenberg (Le44) and Marquardt (Ma63). It uses the best features of Gauss-

Newton method and the steepest-descent method by interpolating between the two methods. Interpolation is achieved by multiplying the diagonal elements of the curvature matrix by $(1+\lambda)$. Then instead of eq. (5), we have

$$\boldsymbol{\beta} = \mathbf{A}'\delta\mathbf{a} \tag{10}$$

where $\mathbf{A}'$ is the p×p matrix whose components A'_{jk} are given by

$$\begin{aligned} A'_{jk} &= A_{jk}(1+\lambda) \quad \text{for } j = k \\ &= A_{jk} \qquad\quad \text{for } j \neq k. \end{aligned} \tag{11}$$

If the interpolation factor λ is much smaller than unity, the solution of eq. (10) approaches the solution by Gauss-Newton method. If λ is much greater than unity, the increments are given by

$$\delta a_j \cong \frac{\beta_j}{\lambda A_{jj}} \qquad j = 1,\ 2,\ \ldots,\ p, \tag{12}$$

and are in the direction of the steepest descent of E.

3.2.2 Choice of the Interpolation Factor

In the original recipe of the Marquardt's strategy the value of λ is initially set to 10^{-3}. After each iteration it is increased by the factor of 10 if $E(\mathbf{a})>E(\mathbf{a}_0)$, and decreased by the same factor if $E(\mathbf{a})\leq E(\mathbf{a}_0)$. The choice of λ in this recipe, therefore, is made with a logarithmically constant spacing. In the present algorithm we use a modified strategy which uses a variable spacing for the choice of λ (Ta75a).

To change the spacing of λ, the following five factors are provided in place of the single factor of 10: ν_1=1.33 ($\cong 10^{1/8}$), ν_2=1.78 ($\cong 10^{1/4}$), ν_3=3.16 ($\cong 10^{1/2}$), ν_4=10 and ν_5=10^2. One of these factors ν_i is selected for each iteration depending on *history* of minimization. The *history* is defined here as a sequence of the results of comparing $E(\mathbf{a})$ with $E(\mathbf{a}_0)$, and consists of the latest three results except at the earliest iterations. The scheme used in the present strategy to determine the value of the index i to the ν factor is given in Table 1.

Performance tests of the algorithm with the original and modified strategies have been made for six problems (Ta75a). The results have shown that the modified recipe requires smaller numbers of equivalent function evaluations (the total number of evaluations of the function and its derivatives) and shorter execution times than the original strategy except for a single problem.

Table 1. Scheme for determining the value of the index i to the v factor from the previous value i_0. Symbols D and I stand for the decrease and increase, respectively, of the error criterion E; DI, for example, means that E was decreased and then increased at the latest two iterations. (Cited from ref. Ta75a.)

Conditions		
History	i_0	i
At the start	...	4
DI, ID	...	i_0-1
DDI, IDI, IID	$i_0>1$	i_0-1
DDD	$i_0<5$	i_0+1
III	$i_0<3$	3
All the other cases		i_0

3.2.3 Iteration and Criteria for Convergence

The first iteration of our algorithm starts with $\lambda=0$, i.e. with Gauss-Newton method. The use of the maximum-neighborhood algorithm begins when the iteration with $\lambda=0$ has failed in decreasing the value of E. The value of λ is reset to zero when it has decreased below 10^{-8}, and Gauss-Newton method is again tried. When the number of function evaluation has reached $n_{\lim}\times(1+2p)$, computation is stopped to reconsider the problem, where $n_{\lim}$ is the approximate limit to the number of iteration (named NITLIM in the code).

When all the following conditions are satisfied, the solution is regarded as converged:

(i) $\lambda = 0$ (13)

(ii) $\left|\delta a_j / a_j\right| < \tau \quad j = 1,\ 2,\ ...,\ p,$ (14)

where τ is a parameter to indicate the number of correct figures desired in a_j. We use $\tau=5\times10^{-5}$ in our algorithm.

3.2.4 Weights

The values of the weight w_i in eq. (3) are calculated by

$$w_i = \frac{n/\sigma_i^{\,2}}{\sum_{i=1}^{n}\left(1/\sigma_i^{\,2}\right)}, \tag{15}$$

where σ_i is the uncertainty or the standard deviation of the ith data y_i. Either eq. (15) or one of the following special cases can be selected in the present algorithm by an input variable (see also Appendix I.2.2.)

(1) Absolute uncertainties are regarded as a constant (σ_i=const), or absolute errors ($y-y_i$) are expected to be rather uniform:

$$w_i = 1. \tag{16}$$

(2) Relative errors $(y-y_i)/y_i$ are expected to be rather uniform (equivalent to σ_i=const$\cdot y_i$):

$$w_i = \frac{n/y_i^2}{\sum_{i=1}^{n}\left(1/y_i^2\right)} \tag{17}$$

(3) Relative uncertainties can approximately be regarded as a constant (uncertainties of the data are larger than the systematic errors of the fitting formula):

If eq. (17) is used in this case, the data with smaller values are weighted more heavily than the data with larger values at the same values of the independent variables. Therefore, we propose to use a two-step method in this case (see also Appendix I.D.3). At the first step $\ln y$ is fitted to the data $\ln y_i$ with w_i=1. At the second step y is fitted to the data y_i with w_i given by

$$w_i = \frac{n/y^2\left(\mathbf{x}_i;\mathbf{a}_\mathrm{F}\right)}{\sum_{i=1}^{n}\left[1/y^2\left(\mathbf{x}_i;\mathbf{a}_\mathrm{F}\right)\right]}, \tag{18}$$

where $\mathbf{a}_\mathrm{F}$ represents the values of the parameters $\mathbf{a}$ determined in the first step.

(4) Uncertainties are statistical ($\sigma_i = y_i^{1/2}$):

$$w_i = \frac{n/y_i}{\sum_{i=1}^{n}\left(1/y_i\right)}. \tag{19}$$

3.3 Outline of the Code ALESQ

The code we have developed to solve the problems of nonlinear least-squarres fit by the algorithm described in the previous subsections is called ALESQ. It is written in FORTRAN, and consists of a main program ALESQ and seventeen subprograms.

The user should consult *COMMENT Statements for ALESQ* at the end of this report. The COMMENT statements give the program summary, and describe

the meaning of variables in the input file and in a PARAMETER statement, the function of the subprograms and the meaning of parameters in the subprograms.

In *FORTRAN Code for ALESQ*, the main program and subprograms are grouped in the following files:

(i) ALESQMN.f consisting of the main program ALESQ, and subprograms ALESQL, CALDER and WRITER
(ii) ALESQSUB.f consisting of the subprograms ALESQN, APUT, CHITEST (with attached function subprograms PCHISQ, GAMMA, and FACTOR), MATINV, PUT and WEIGHT
(iii) ALESQFUN.f consisting of CALFUN
(iv) ALESQOPT.f consisting of CALAIN, OPTION and PLOT
(v) ALESQRD.f consisting of RDDATA.

The last three files contain subprograms that are needed to be modified by the user most frequently. All the program in the files ALESQMN.f and ALESQFUN.f contain the PARAMETER statement.

The first three lines of the input file are read by the main program ALESQ. The next lines for the data to be fitted are read by the subprogram RDDATA. At the end of these lines, a line with input values for the parameters MDRUN and MKEND is required. The symbol * as a value of the parameter MKEND indicates the end of the data; the value of the parameter MDRUN indicates whether the present set of data or the present function is used in the next run. The line for the starting values of the parameters **a**, when provided, is read by the subprogram ALESQN.

The user have to provide FORTRAN statements for the fitting function as well as the input file. The subprogram in which the fitting function is to be written is the subroutine subprogram CALFUN. The following built-in function is provided for test purpose:

$$y = a_1 x\left(x + a_2\right)\Big/\left(x^2 + a_3 x + a_4\right). \tag{20}$$

In the subprogram ALESQN, the result is examined against the convergence criteria, the value of λ to be used is determined, and the subprogram ALESQL is called to solve the linearized problem. The subprogram ALESQL computes the increments $\delta\mathbf{a}$ of the parameters and the absolute error $\sigma_{\text{abs}}(\mathbf{a})$ of fit defined by

$$\sigma_{\text{abs}}(\mathbf{a}) \equiv \left[E(\mathbf{a})/(n-p)\right]^{1/2} \tag{21}$$

When the search for the solution has been terminated, the final result is printed. The output includes the values of the absolute error σ_{abs} of fit, the

relative error σ_{rel} of fit and the rms relative residual σ_{rms}. The relative error of fit σ_{rel} is defined by

$$\sigma_{\mathrm{rel}} \equiv \left(\frac{1}{n-p} \sum_{i=1}^{n} w_i' \left\{ \frac{y_i - y(\mathbf{x}_i)}{y_i} \right\}^2 \right)^{1/2}, \tag{22}$$

where

$$w_i' = \frac{n}{\sum_{i=1}^{n} w_i y_i^{\,2}} w_i y_i^{\,2}. \tag{23}$$

Putting eq. (23) into eq. (22), we have

$$\sigma_{\mathrm{rel}} = \frac{n}{\sum_{i=1}^{n} w_i y_i^{\,2}} w_i y_i^{\,2} \sigma_{\mathrm{abs}}. \tag{24}$$

The rms relative residual σ_{rms} is defined by

$$\begin{aligned} \sigma_{\mathrm{rms}} &\equiv \left\{ \frac{1}{n} \sum_{i=1}^{n} w_i' \left[\frac{y_i - y(\mathbf{x}_i)}{y_i} \right]^2 \right\}^{1/2} \\ &= \left(\frac{n-p}{n} \right)^{1/2} \sigma_{\mathrm{rel}}. \end{aligned} \tag{25}$$

The parameters σ_{rel} and σ_{rms} are useful as a measure of goodness of fit when the weights given by eq. (17) or (18) are used.

At the end of a run, the subprogram OPTION is called, which can be used for performing optional calculations or plotting the result of the fit. The user have to provide necessary statements for this subprogram.

Next to the list of the code, the input and output files of a test problem is given. The fitting function for the test problem used is called an enzime function (Ko67), and is provided as a built-in function in the subprogram CALFUN [NFUNC=1; eq. (20)]. The starting values of **a** are given by $\mathbf{a}^{\mathrm{T}}$=(0.25, 0.4, 0.4, 0.4).

4 TSOLVE: An Algorithm for Nonlinear Best-Approximation

4.1 Formulation of the Problem

In the best approximation the error criterion E is given by

$$E = \max\left|w_i\left[y_i - y(x_i; \mathbf{a})\right]\right|, \tag{26}$$

where w_i is the weight for the ith data, and the number m of the independent variables is assumed to be one ($x_i \equiv x_{1i}$). To the approximation of y given by eq. (2) in Sec. 2.1, E is expressed as

$$E = \max\left|w_i\left[y_i - y_0(x_i) - \delta\mathbf{a}^{\mathrm{T}}\nabla y_0(x_i)\right]\right|. \tag{27}$$

The linearized problem to find a set of solutions $\delta\mathbf{a}$ and E to eq. (27) for a finite set of points x_i is called the discrete T-problem. The problem can be restated:

Minimize E subject to the constraints

$$\left|w_i\left[y_i - y_0(x_i) - \delta\mathbf{a}^{\mathrm{T}}\nabla y_0(x_i)\right]\right| \le E \qquad i = 1,\ 2,\ \ldots,\ n. \tag{28}$$

This is equivalent to a linear programming problem (Ke59, Os67).

When a function $\eta(x)$ and $w(x)$ continuous in a given interval $a \le x \le b$ is considered instead of a finite set of data y_i and weights w_i, we have a problem to minimize E given by

$$E = \max\left|w(x)\left[\eta(x) - y(x; \mathbf{a})\right]\right| \tag{29}$$

in $a \le x \le b$. This is called the continuous T-problem. A continuous T-problem is reduced to a discrete T-problem by choosing n points x_i in $a \le x \le b$ and identifying $\eta(x_i)$ and $w(x)$ with y_i and w_i, respectively.

4.2 Description of the Algorithm

4.2.1 Osborne-Watson Algorithm

To solve the nonlinear best-approximation problem, the following iterative procedure, called Osborne-Watson algorithm (Os69), is used:

(i) Calculate $\delta\mathbf{a}$ to minimize E subject to the constraints (28). Use the linear programming technique to obtain the solution. Denote the minimum value of E by $\hat{E}$.

(ii) Calculate γ to minimize the maximum value of

$$\left|w_i\left[y_i - y(x_i; \mathbf{a}_0 + \gamma\delta\mathbf{a})\right]\right| \qquad i = 1,\ 2,\ \ldots.,\ n. \tag{30}$$

Let the minimum value be $\overline{E}$.

(iii) If $|E - E|$ is sufficiently small, the solution can be regarded as converged. If not, replace $\mathbf{a}_0$ by $\mathbf{a}_0+\gamma\delta\mathbf{a}$ and return to step (1).

The method of applying the linear programming technique to step (1) is described later. The optimum solution to γ in step (2) is expected to lie between 0 and 1 (Os69), and we solve this problem by the sequential search method starting from γ=1.

4.2.2 Exchange of Data in the Continuous T-Problem

The criterion for the discrete T-problem reduced from a continuous T-problem represents the criterion for the original continuous T-problem only approximately. To obtain a better approximation by using a moderate number of data points, we use a method of exchanging part of the data (Ta77b). When the mode for the continuous T-problem is selected by one of the input variables (MDINS), the data nearest to the extremal points of the error curve are replaced by the data at the extremal points in each iteration. The replacement data are computed by quadratic interpolation of the error curve given by the original data points.

4.2.3 Weights

The best approximation is usually sought for the problems in which the uncertainties of the data are negligible compared with the expected error of the approximating function. Therefore, we consider only the following two cases of the weights w_i:

(i) The maximum in magnitude of absolute errors $(y - y_i)$ are to be minimized:

$$w_i = 1. \tag{31}$$

(ii) The maximum in magnitude of relative errors $(y - y_i)/y_i$ are to be minimized:

$$w_i = \left(n/y_i^2\right)\Big/\sum_{i=1}^{n}\left(1/y_i^2\right). \tag{32}$$

4.2.4 Use of Linear Programming Technique

In the following description, expressions are given for the case of the weights given by eq. (31). Adaptation to the case of eq. (32) is simple.

The problem to minimize E subject to constraints (28) can then be rewritten as:

The primal problem: Maximize $g(\mathbf{v}) = \mathbf{v}^{\mathrm{T}}\mathbf{b} = -E$ subject to

$$\mathbf{v}^{\mathrm{T}}\mathbf{A} \le \mathbf{c}, \tag{33}$$

where **v** and **b** are the (p+l)-dimensional vectors given by

$$v = \left(\delta a_1, \delta a_2, \mathrm{K}, \delta a_p, -E\right)^{\mathrm{T}} \tag{34}$$

$$b = \left(0, 0, \mathrm{K}, 0, 1\right)^{\mathrm{T}}, \tag{35}$$

c is the (2n+1)-dimensional vector whose elements c_j are given by

$$c_j = y_0\left(x_j\right) - y_j \qquad j = 1,\ 2,\ \ldots,\ n \tag{36a}$$

$$c_{n+j} = -\left[y_0\left(x_j\right) - y_j\right] \qquad j = 1,\ 2,\ \ldots,\ n \tag{36b}$$

$$c_{2n+1} = 0, \tag{36c}$$

and **A** is the (p+1)×(2n+1) matrix whose elements A_{ij} are given by

$$A_{ij} = -\frac{\partial y_0\left(x_j\right)}{\partial a_i} \qquad i = 1,\ 2,\ \ldots,\ p;\ j = 1,\ 2,\ \ldots,\ n \tag{37a}$$

$$A_{i,n+j} = \frac{\partial y_0\left(x_j\right)}{\partial a_i} \qquad i = 1,\ 2,\ \ldots,\ p;\ j = 1,\ 2,\ \ldots,\ n \tag{37b}$$

$$A_{i,2n+1} = 0 \qquad i = 1,\ 2,\ \ldots,\ p \tag{37c}$$

$$A_{p+1,\,j} = 1 \qquad i = 1,\ 2,\ \ldots,\ 2n+1. \tag{37d}$$

This problem is equivalent to the following dual problem by the duality theorem of the linear programming (see for example Chapter 6 of ref. Ga75).

The dual problem: Minimize f($\boldsymbol{\xi}$)=$\mathbf{c}^{\mathrm{T}}\boldsymbol{\xi}$ subject to the conditions

$$\mathbf{A}\boldsymbol{\xi} = \mathbf{b} \tag{38}$$

and

$$\boldsymbol{\xi} \ge \mathbf{0}, \tag{39}$$

where $\boldsymbol{\xi}$ is a (2n+1)-dimensional vector, and **0** is a null vector of the same dimension.

This dual problem has a form of the general linear programming problem. According to the duality theorem, the optimal solution min[f($\boldsymbol{\xi}$)] of the dual problem is equal to the optimal solution max[g($\mathbf{v}$)] of the primal problem. We solve the dual problem using the revised-simplex method (see for example Chapter 5 of ref. Ga75).

To use the revised-simplex procedure, we set

$$M = p + 1 \tag{40}$$

$$N = 2n + 1, \tag{41}$$

and rewrite the dual problem in the form:

Maximize ξ_{N+M+1} subject to

$$\overline{\mathbf{A}}\boldsymbol{\xi} + \boldsymbol{\xi}' = \mathbf{b}' \tag{42}$$

and

$$\xi_i \geq 0 \qquad i = 1,\ 2,\ \ldots,\ N,\ N+1,\ \ldots, N+M. \tag{43}$$

Here $\overline{\mathbf{A}}$ is the $(M+2)\times N$ matrix whose elements $\overline{A}_{ij}$ are given by

$$\overline{A}_{ij} = A_{ij} \qquad i = 1,\ 2,\ \ldots,\ M;\ j = 1,\ 2,\ \ldots,\ N \tag{44}$$

$$\overline{A}_{M+1,\,j} = c_j \qquad j = 1,\ 2,\ \ldots,\ N \tag{45}$$

$$\overline{A}_{M+2,\,j} = -\sum_{i=1}^{M} \overline{A}_{ij} \qquad j = 1,\ 2,\ \ldots,\ N \tag{46}$$

$\boldsymbol{\xi}'$ is the $(M+2)$-dimensional vector whose components ξ_i' are denoted as

$$\xi_i' = \xi_{N+i} \qquad i = 1,\ 2,\ \ldots,\ M+2, \tag{47}$$

and $\mathbf{b}'$ is the $(M+2)$-dimensional vector given by

$$\mathbf{b}' = \left(b_1, b_2,\ \ldots, b_M, 0, b_{M+2}\right) \tag{48}$$

$$b_{M+2} = -\sum_{i=1}^{M} b_i\,. \tag{49}$$

In the general problem the following relations are assumed:

$$M < N \tag{50}$$

$$b_i \geq 0 \qquad i = 1,\ 2,\ \ldots,\ M. \tag{51}$$

Relations (51) do not place restrictions on the problem, because both sides of the component equations of (38) can be multiplied by -1 where necessary to satisfy these relations.

In our problem, we have special relations (35)–(37), (40) and (41); relation (50) is satisfied, because $n>p\geq$l; relations (51) are satisfied by definition (35).

We see from eqs. (45), (47) and component (M+1) of eq. (42) that ξ_{N+M+1} is equal to minus the objective function of the dual problem. Therefore, the optimum solution to ξ_{N+M+1} gives minus the optimum solution to the objective functions in the dual and primal problems, i.e., just the optimum value of E in our best-approximation problem.

The standard procedure for the revised-simplex method is started by preparing the followings: the matrix **A**; the initial solution of $\xi_{N+i}=b_i$ for i=1, ..., M; ξ_{N+M+1}=0; and $\xi_{N+M+2}=b_{M+2}$. The optimum solutions to δ**a** are obtained as minus the first p elements in row (M+1) of the final tableau for the revised simplex procedure (the matrix **U** in Chapter 5 of ref. Ga75).

4.2.5 Tolerances for Roundoff

When the tolerance εE to the best-approximation error E is given, tolerances $\Delta\mathbf{a}=(\Delta a_1, \Delta a_2, ..., \Delta a_p)^{\mathrm{T}}$ for roundoff of **a** can be estimated as follows (Ta77a):

For j=1, 2, ..., p, maximize $\Delta a_j'$ (>0), subject to

$$\left| y(x_i;\mathbf{a}) + \Delta a_j' \frac{\partial y(x_i;\mathbf{a})}{\partial a_j} - y_i \right| \leq (1+\varepsilon)E \qquad i = 1,\ 2,\ ...,\ n. \tag{52}$$

Then $\Delta a_j = \Delta a_j'/p$ gives *an* estimate of the tolerances, because

$$\begin{aligned} \left| y(x_i;\mathbf{a}+\Delta\mathbf{a}) - y_i \right| &\cong \left| y(x_i;\mathbf{a}) + \Delta\mathbf{a}^{\mathrm{T}} \nabla y(x_i;\mathbf{a}) - y_i \right| \\ &\leq (1/p)\sum_{i=1}^{p} \left| y(x_i;\mathbf{a}) + \Delta a_j' \frac{\partial y(x_i;\mathbf{a})}{\partial a_j} - y_i \right| \\ &\leq (1+\varepsilon)E \qquad i = 1,\ 2,\ ...,\ n. \end{aligned} \tag{53}$$

Solutions to $\Delta a_j'$ are given by

$$\Delta a_j' = \min\left(\left\{ \pm(1+\varepsilon)E - \left[y(x_i;\mathbf{a}) - y_i \right] \right\} \Big/ \frac{\partial y(x_i;\mathbf{a})}{\partial a_j} \right), \tag{54}$$

where the double sign corresponds to the relations

$$\frac{\partial y(x_i;\mathbf{a})}{\partial a_j} > \text{ or } < 0 \tag{55}$$

We use ε=0.005 in our algorithm.

The tolerances thus estimated do not always satisfy relations (53) because of the approximation used in the first line of these relations. Therefore, it is

sometimes necessary to retain more significant digits for a_i than indicated by Δa_i. For the convenience of checking the roundoff effect and cut-and-try adjustment of the number of significant digits, the present algorithm provides a mode of computing the error curve only.

4.3 Outline of the Code TSOLVE

The code we have developed to solve the problems of nonlinear best-approximation by the algorithm described in the previous subsections is called TSOLVE. It is written in FORTRAN, and consists of a main program TSOLVE and fourteen subprograms.

The user is advised to consult *COMMENT Statements for TSOLVE* at the end of this report. The COMMENT statements give the program summary, and describe the meaning of variables in the input file and in PARAMETER statements, the function of the subprograms and the meaning of parameters in the subprograms.

In *FORTRAN Code for TSOLVE*, the main program and subprograms are grouped in the following files:

(i) TSOLMN.f consisting of the main program TSOLVE, subprograms SELMAX, SIMPDT, SIMPLX, SOLVE, SRMXPT, START, WRITE1, FDERIV, FMAX, PUT and PUTI
(ii) TSOLFUN.f consisting of the subprogram FUNC
(iii) TSOLOPT.f consisting of the subprogram OPTION
(iv) TSOLRD.f consisting of the subprogram READ1

The last three files contain the subprograms that are needed to be modified by the user most frequently. The main program and the subprograms from OPTION to WRITE1 contain PARAMETER statements.

The first three lines of the input file are read by the subprogram START. The next lines for the data to be fitted are read by the subprogram READ1, which is called by START. At the end of the data, a line with input values for the parameters MDRUN and MKEND is required. The symbol * as a value of the parameter MKEND indicates the end of the data; the value of the parameter MDRUN indicates whether the present set of data or the present function is used in the next run. The line for the starting values of the parameters **a**, when provided, is read by the subprogram START.

The user have to provide FORTRAN statements for the approximating function as well as the input file. The subprogram in which the approximating

function is to be written is the function subprogram FUNC. The test function given by eq. (20) on page 7 is provided in FUNC as a built-in function.

After calling START, the main program calls the subprogram SELMAX, in which the maximum error of the approximating function for the starting values of the parameters is calculated. Then the subprograms WRITE1 and SOLVE are called repeatedly. In the subprogram SOLVE, the subprogram SIMPLX is called to solve the linearlized problem, optimization of γ is made, and the result is examined against the convergence criterion. When the convergence criterion has been satisfied, tolerances for roundoff of the parameters are calculated.

The final output includes the solution **a**, the tolerances $\Delta\mathbf{a}$ for roundoff, the list of the extremal points of the error curve, the input data, the values of the fitting function at the data points and errors at these points.

At the end of a run, the subprogram OPTION is called, which can be used for performing optional calculatins or plotting the result of the fit. The user have to provide necessary statements for this subprogram.

Next to the list of the code, the input and output files of a test problem is given. The approximating function for the test problem used is the same as the test function for ALESQ [eq. (20) on page 7]. It is provided as a built-in function in the subprogram FUNC (NFUNC=1). We take the starting values of **a** from the result of ALESQ: $\mathbf{a}^T$=(0.1928, 0.1913, 0.1231, 0.1361).

Appendix 1 Formulation of Empirical Equations

In this appendix, procedures and methods to formulate empirical equations from experimental data are briefly discussed mainly on the basis of the present authors' experience.*

A1.1 Outline of Procedures

Procedures used for formulating empirical equations constitute a number of recycles, and can best be explained with a flow diagram as given in Fig. 1 (a similar flow diagram containing more detailed explanation of each procedure is given in ref. Da71). The first procedure of collecting data is made by conducting an experiment, searching literature or combining the two methods. Procedures after collecting data are discussed here.

A1.2 Examinataion of Data

Let us consider that the values Y_i (i=1, 2, ..., n) of the dependent variable and σ_i of the standard deviation corresponding to the values $\mathbf{x}_i = (x_{1i}, x_{2i}, ..., x_{mi})^T$ of independent variables have been collected. Here n denotes the number of the data, and m the number of the independent variables. The case of unknown σ_i is discussed later. Items to be considered in examining the data collected are as follows:

(l) Check errors that might have been caused in the processes of reading-off from literature and making input to the computer. Plotting of the dependent variable as a function of each of the independent variables is useful for this purpose.

(2) Check whether there are some data sets showing large deviations from the rest of the data sets. If there are any such sets, try to find the possible cause of the deviations. Do not use the data set with the large deviations the cause of which is identifiable. [For a similar discussion, see the review article by Nakashima (Na81). See also the later discussion on adjustable weights.]

---

* This appendix is an extension of part of the talk given by one of the authors (T. T.) at the Research Meeting on Evaluation of Atomic Collision Cross Sections held at the Japan Atomic Energy Research Institute, Tokyo, in 1982.

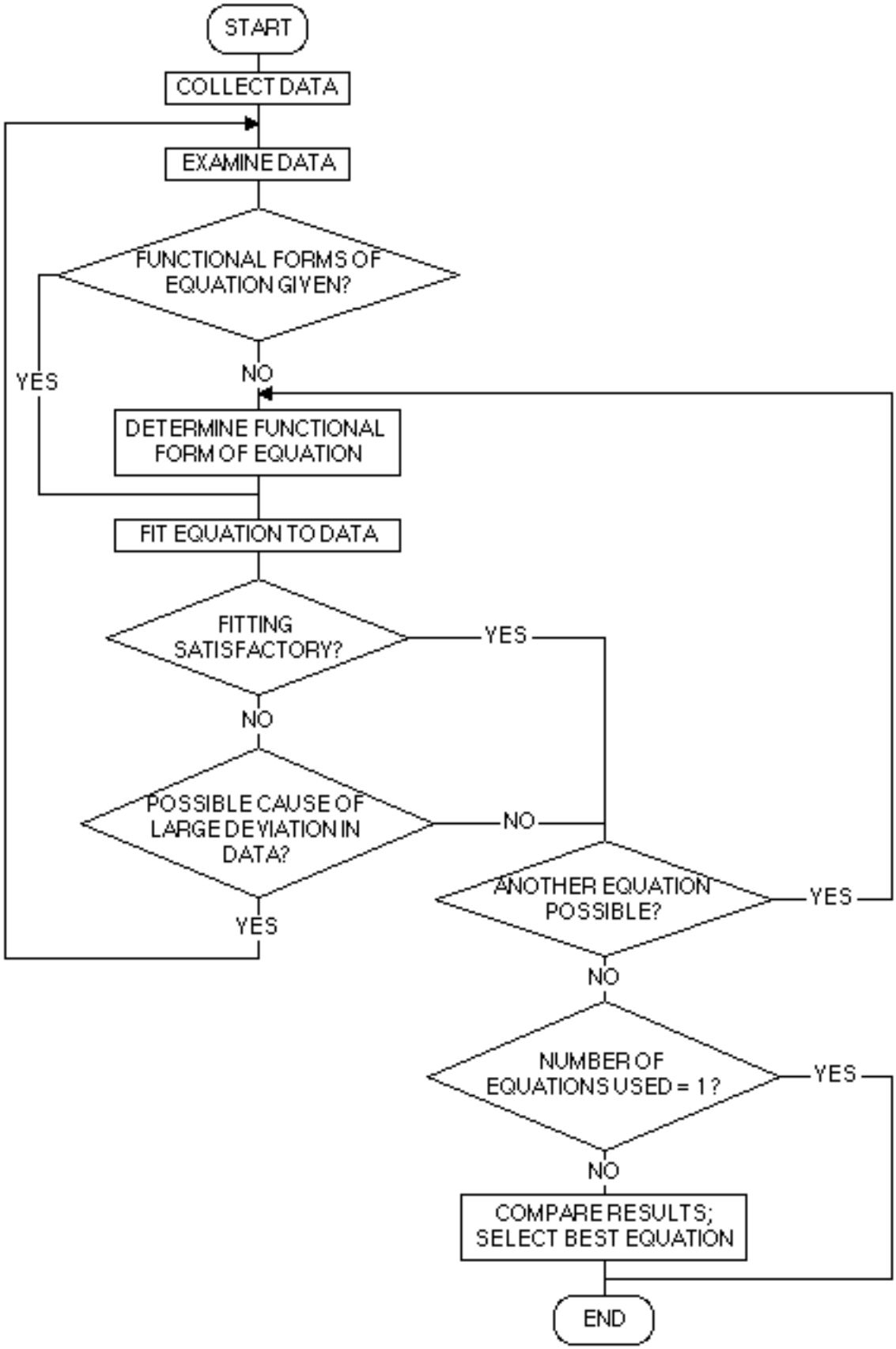


Fig. 1. Flow diagram for formulating empirical equations.

A1.3 Determination of the Functional Form of the Equation

A1.3.1 Methods to Derive the Functional Form

When we do not have the functional form of the equation beforehand, we have to make one. Methods to derive the functional form can be classified as follows:

(i) Theoretical approach.
   a. Derive an equation from a simple model of the phenomenon, and use unknown constants as adjustable parameters.
   b. Take the prediction of an asymptotic theory into the formulation of an equation.
   c. Use a scaling law theoretically predicted.
   d. Replace some constants in an approximate theoretical equation by adjustable parameters.
   e. Replace a theoretical equation by a simple equation showing a similar behavior.

(ii) Empirical approach.
   a. Find a possible function by plotting the data.
   b. Use a scaling law empirically found.

In the actual formulation, a combination of these methods are often used. In a formula devised to analyze a phenomenon, each parameter should have some theoretical meaning. However, there are cases in which a model that can lead to an analytic formula is too simple to describe the phenomenon with enough accuracy. In such cases, addition of parameters or modification of the formula on empirical grounds might be allowed.

Example 1: method (i)a. Weizsacker's famous formula for nuclear ground state energy E is written as (see for example ref. Bl52a)

$$E = -u_{\mathrm{v}} + 4u_{\mathrm{r}} T_{\zeta}^{\,2} \big/ A + 4u_{\mathrm{c}} Z(Z-1) A^{-1/3} + u_{\mathrm{s}} A^{2/3}, \qquad \text{(A1)}$$

where A and Z are the mass number and the atomic number of the nucleus, respectively, T_{ζ} is given by

$$T_{\zeta} = (N - Z)/2, \qquad \text{(A2)}$$

N is the number of neutrons in the nucleus, and u_{v}, u_{r}, u_{c} and u_{s} are constants to be determined by comparison with experiment. Each term on the right-hand side of eq. (A1) has theoretical implication: the first term expresses the volume

energy, the second term the symmetry energy, the third term the Coulomb energy, the fourth term the surface energy.

Example 2: methods (i)b and (i)e.

Green and Barth (Gr65) used the following formula to express excitation cross sections σ for the nitrogen molecule N_2 as a function of energy E of the incident electron:

$$\sigma(E) = a_1(1 - W/E)^{a_2}(W/E)^{1-a_3}. \tag{A3}$$

Here a_i (i=1, 2, 3) are adjustable parameters, and W is the excitation energy of the state involved in the inelastic collision. The factor $(1 - W/E)^{a_2}$ provides for the threshold behavior of the cross section. The factor $(W/E)^{1-a_3}$ with $a_3 \approx 1/4$ is used for analytic convenience to simulate the asymptotic dependence of the form $E^{-1} \ln E$ predicted by a theory.

Example 3: method (i)d.

Weber (We64) has derived semiempirical formulas for the stopping power of material for electrons and the ranges of electrons in material. He has considered the energy region where radiative energy-loss of electrons can be neglected, and has started from an approximate formula for the stopping power S:

$$S = \frac{kf(E)}{1 - \left(1 + E/m_0c^2\right)^{-2}}, \tag{A4}$$

where k is a constant independent of the energy E of the electrons, $f(E)$ is a function weakly depending on E, and m_0c^2 is the rest energy of the electron. He has replaced eq. (A4) by the following equation:

$$S = \frac{a}{1 - B(1 + CE)^{-2}}, \tag{A5}$$

where a, B and C are adjustable parameters. The continuous slowing-down approximation range R_0, defined by

$$R_0 = \int_0^E dE/S, \tag{A6}$$

represents the average path-length of electrons in material. Putting eq. (A5) into eq. (A6), Weber has obtained an empirical equation for R_0

$$R_0 = AE\left(1 - \frac{B}{1 + CE}\right), \tag{A7}$$

where A=1/a. While R_0 is theoretically computable, eq. (A7) is useful for rapid evaluation. More importantly, Weber has shown that the right-hand side of

eq. (A7) can also be used for expressing the extrapolated range R_{ex} as a function of E. The extrapolated range is a measure of the path-length projected on the axis along the depth, and is difficult to evaluate theoretically. Using an approximate formula for S including the radiative energy-loss, the present authors made a formula valid also for higher energies (Ta72a).

A1.3.2 Some Useful Functions

In using method (ii)a to determine the functional form, reference to a collection of functions is helpful. Useful collections of functions have been given by Hoerl (Ho54) and Daniel and Wood (Da71). Among these collections, Hoerl's special function defined by

$$y = a_1 x^{a_2} \exp(a_3 x) \tag{A8}$$

and the so-called logistic function expressed as

$$y = \frac{1}{a_1 + a_2 \exp(-a_3 x)} \tag{A9}$$

are especially useful.

Equation (A8) has originally been compiled by Hoerl and cited by Daniel and Wood; it can be linearlized by taking logarithms when a_1 and x are positive, and expresses the curve that starts from the origin, increases to a maximum and then dies away when a_1 and a_2 are positive and a_3 negative. The present authors have used the relation essentially same as eq. (A8) to express the correction factor to the radiative energy-loss of electrons as a function of the energy of the electrons (Ta77b).

Introducing an additional adjustable parameter into eq. (A8) and replacing a_3 by $-a_3$, we have

$$y = a_1 x^{a_2} \exp(-a_3 x). \tag{A10}$$

Assume that all the adjustable parameters in eq. (A10) are positive. Let the values of x in this equation be scaled so that y has the maximum at x=1. Then we have the relation $a_2 = a_3 a_4$. With this scaling, the area under the curve of eq. (A10) is normalized to unity by the relation $a_1 = a_3^{\,b} a_4 / \Gamma(b)$, where $b = a_3 + 1/a_4$. The present authors have used eq. (A10) to express the projected-range distribution of electrons (Ta71c).

Equation (A9) expresses an S-shaped or sigmoid curve. When y as a function of $\log x$ shows an S-shaped curve, we have a simpler form:

$$y = 1/\left(a_1 + a_2 x^{a_3}\right). \tag{A11}$$

The present authors have used eq. (A11) in formulating an empirical equation for the backscattering coefficient of electrons (Ta71a).

While eq. (A10) gives a variety of asymmetric distributions, another family of asymmetric distributions is obtained by Pearson distributions, which are generalizations of the Gaussian distribution

$$y = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-x^2/2\sigma\right). \tag{A12}$$

The Gaussian distribution satisfies the differential equation

$$\frac{dy}{dx} = -y\frac{x}{\sigma}. \tag{A13}$$

The Pearson distributions are based on the generalized differential equation

$$\frac{dy}{dx} = -y\frac{x - a_1}{a_2 + a_3 + a_4 x^2}. \tag{A14}$$

The solutions of this equation are given, for example, in refs. Ab70 and Ho75; examples of applications are seen in refs. Bl52b and Ho75.

A1.3.3 Plotting Data

To find an equation by plotting data, semilogarithmic or logarithmic graph papers are useful. Transformation of the variable in plotting the data sometimes helps to find a simple relation. Hastings (Ha55) has described the rational transformation defined by

$$\xi = \frac{b_0 + b_1 x}{1 + c_1 x}, \tag{A15}$$

which makes a curved line approach a straight line. To determine suitable values of the parameters b_0, b_1 and c_1, we divide the range $[y_0, y_2]$ considered of y into two roughly equal parts: $[y_0, y_1]$ and $[y_1, y_2]$, and then require that x_0, x_1 and x_2, corresponding to y_0, y_1 and y_2, respectively, transform to equally spaced values of, say, 0, 0.5, and 1.0.

A point-symmetric S-shaped curve in semilogarithmic plot can be fitted with eq. (A11). Some probability distributions become Gaussian in semilogarithmic plot (see, for example, ref. Fu77c).

A1.4 Fitting Equations to Data

After having determined the functional form of the equation, we have to find the most probable estimates of the adjustable parameters $\mathbf{a}=(a_1, a_2, \ldots, a_p)^T$ so as to minimize the discrepancy between the data y_i and the corresponding values $y(x_i)$

given by the equation. To do this, we have to solve the two problems: (i) establishing the criterion for minimizing the discrepancy and (ii) optimizing the estimates of the parameters according to this criterion. A simple and well established solution to these problems is the method of least squares, a brief description of which is given in the main sections of this report. Another solution is the best approximation in Tchebychev's sense, which is also treated in this report. In the followings, some important aspects of the method of least squares not described in the main sections are discussed.

A1.4.1 Conventional Method of Least Squares and Robust Estimation Method

The method of least squares is one of special cases of the generalized method of maximum likelihood estimation to obtain the solution **a** that minimizes

$$E = \sum_{i=1}^{n} \psi_i(\eta_i), \tag{A16}$$

where η_i is given by

$$\eta_i = y_i - y(\mathbf{x}_i; \mathbf{a}). \tag{A17}$$

In the conventional method of least squares, $\psi_i(\eta_i)$ is defined by

$$\psi_i(\eta_i) = v_i^2 \equiv w_i \eta_i^2, \tag{A18}$$

where v_i is called the normalized variance, and w_i is the weight proportional to $1/\sigma_i^2$.

To reduce the overwhelming effect of data with large deviations, there are techniques to use additional weighting factor $p(v_i)$ that depends on v_i; these are called the robust estimation method (for a brief description, see ref. Oy79). In this method, $\psi_i(\eta_i)$ is defined by

$$\psi_i(\eta_i) = p(v_i) v_i^2. \tag{A19}$$

The conventional method of least squares corresponds to the case of constant p:

$$p = 1. \tag{A20}$$

When p depends on v_i, it is called the adjustable weight. The adjustable weights often used are:

(i) Method of least squares with cutting;

$$p(v_i) = \begin{cases} 1 & |v_i| \le c \\ 0 & |v_i| > c, \end{cases} \tag{A21}$$

where c is a constant.

(ii) Huber's method;

$$p(v_i) = \begin{cases} 1 & |v_i| \le c \\ c/|v_i| & |v_i| > c. \end{cases} \tag{A22}$$

(iii) Biweight method;

$$p(v_i) = \begin{cases} \left[1-(v_i/c)^2\right]^2 & |v_i| \le c \\ 0 & |v_i| > c. \end{cases} \tag{A23}$$

A1.4.2 Determination of Weights

The relation of the weight w_i and the standard deviation σ_i of the ith data is given by eq. (15) of Sec. 3.2.3. The standard deviation, in turn, is determined by one of the following uncertainties involved:

(i) Instrumental uncertainty.
(ii) Statistical uncertainty.
(iii) Unknown uncertainty.

For statistical uncertainty, σ_i is equal to $\sqrt{y_i}$, and w_i is given by eq. (19) of Sec. 3.2.3. There are other cases in which the expression for w_i is simplified:

(i) $\sigma_i = \text{const}: \; w_i = 1.$
(ii) $\sigma_i = \text{const}:$ w_i is given by eq. (17) of Sec. 3.2.3.

Case (i) corresponds to the situation in which absolute uncertainties can be regarded as a constant. The relation $w_i = 1$ can also be used when absolute errors $(y - y_i)$ of fit are expected to be rather uniform. Case (ii) corresponds to the situation in which relative errors $(y - y_i)/y_i$ are expected to be rather uniform. In case (ii) uncertainties of the data are assumed to be negligible. The case in which the uncertainties are non-negligible and approximately equal in relative magnitude is considered below.

A1.4.3 Approximately Equal Relative Uncertainties and Two-Step Method

Uncertainties of approximately equal relative magnitude are usually assumed in fitting an equation to the data whose dependent variable ranges over a factor of ten or more. In principle, we can take this assumption into account by the use of the weights w_i:

$$w_i = 1/{y_{0i}}^2, \tag{A24}$$

where y_{0i} represents the true value of the dependent variable corresponding to the ith data. However, we do not know the best estimates of y_{0i} (i=1, 2, ..., n) before performing the fit. Using y_i or $y(x_i)$ instead of y_{0i}, we can consider criteria E_{I} and E_{II} to be minimized:

$$E_{\mathrm{I}} = \sum_{i=1}^{n} \frac{\left[y_i - y(\mathbf{x}_i)\right]^2}{{y_i}^2} \tag{A25}$$

$$E_{\mathrm{II}} = \sum_{i=1}^{n} \frac{\left[y_i - y(\mathbf{x}_i)\right]^2}{y^2(\mathbf{x}_i)}. \tag{A26}$$

The criterion E_{I} corresponds to the use of w_i given by eq. (17) [the constant factor in w_i has been dropped in eq. (A25) for simplicity]. To use E_{II}, the values of $y(\mathbf{x}_i)$ at the starting values of the parameters already have to be a good approximation to the true values y_{0i}. From the point of view of obtaining the result in which relative errors of the fit are rather uniform, we can consider a third criterion:

$$E_{\mathrm{III}} = \sum_{i=1}^{n} \left[\ln y_i - \ln y(\mathbf{x}_i)\right]^2. \tag{A27}$$

Although the results obtained by applying these criteria are almost the same for the data with small standard deviations, differences are appreciable for the data with large standard deviations. The best choice among the three criteria can be determined by considering a model problem as described below.

Consider a pair of data points (x_1, y_1) and (x_2, y_2) with $x_1 = x_2$. We expect the resulting $y(x_1)$ to be close to the arithmetic mean $a_0 = (y_1 + y_2)/2$, because relation (A24) gives $w_1 = w_2$ for these data. Let us find which of E_{I}, E_{II} or E_{III} yields the result closest to a_0 by solving the problem of fitting the equation $y = a$ to the pair of the data points, where a is a constant to be determined. The results obtained from the three criteria are given respectively by

$$a_{\mathrm{I}} = \frac{(y_1 + y_2)y_1y_2}{{y_1}^2 + {y_2}^2} \tag{A28}$$

$$a_{\mathrm{II}} = \frac{{y_1}^2 + {y_2}^2}{y_1 + y_2} \tag{A29}$$

$$a_{\mathrm{III}} = \sqrt{y_1y_2}\,. \tag{A30}$$

A simple arithmetic leads to the relations

$$a_{\mathrm{I}} \le a_{\mathrm{III}} \le a_0 \le a_{\mathrm{II}} \tag{A31}$$

$$a_0 - a_{\mathrm{III}} \le a_{\mathrm{II}} - a_0, \tag{A32}$$

where the signs of equality are valid when $y_1 = y_2$. Relations (A31) and (A32) indicate that a_{III} is the closest to a_0, so that we conclude that criterion E_{III} is the best among the three.

As relation (A31) indicates, the values of $y(\mathbf{x}_i)$ obtained by minimizing E_{III} are still systematically smaller for fluctuating data than the values expected from the weights of relation (A24). For further improvement, we use the two-step fit: In the first step E_{III} is minimized, and in the second step the least-squares fit with the weights given by

$$w_i \propto 1/y^2(\mathbf{x}_i; \mathbf{a}_{\mathrm{F}}) \tag{A33}$$

is applied, where a_{F} denotes the solution of a obtained by minimizing E_{III}. In using this method, it is necessary to confirm that $y(\mathbf{x}_i; \mathbf{a}_{\mathrm{F}})$ (i=1, 2, ..., n) are good estimates of y_{0i}; when $y(\mathbf{x}_i; \mathbf{a}_{\mathrm{F}})$ shows appreciable systematic deviations from y_i, relation (A33) does not approximates relation (A24).

A1.4.4 Algorithms for Optimizing Nonlinear Functions

To find a minimum of a given function nonlinear with respect to unknown parameters a, a number of methods are available. The methods can be classified into three categories:

(i) Direct search method.
   a. Grid search method.
   b. Rosenbrock's method.
   c. Simplex method.
   d. Monte Carlo method.
   etc.

(ii) Gradient method.
   a. Method of steepest descent.
   b. Davidon-Fletcher-Powell method (variable metric method).
   c. Powell's method without using derivatives.
   etc.

(iii) Method of least squares.
   a. Newton-Raphson method.
   b. Levenberg-Marquardt method.
   c. Powell's method of least squares.
   etc.

Derivatives of the function are not used in the methods of category (i), but are used in those of categroy (ii). Methods of category (iii) utilize special techniques applicable to the functions defined as the sum of squares. A brief explanation of these methods is given by Oyanagi (Oy73); for detailed descriptions, see the text books Ko68, Be70, Fl80 and Gi81.

Appendix 2 Work Done by Using ALESQ and TSOLVE

In this appendix, empirical and approximate formulas and other applications for which ALESQ or TSOLVE was used are listed. Results are mostly related to the field of radiation and atomic collision physics. The asterisk attached to the item number indicates the use of TSOLVE.

A2.1 The Passage of Fast Electrons through Matter

A2.1.1 Empirical Formulas and Algorithms

(1) Extrapolated range (Ta72a, Ta96a).
(2) Projected range (Ta68, Ta71b, Ta96b, Fe96).
(3) Most probable range (Ta71c).
(4) Projected-range distribution (Ta71c).
(5) Incident energy as a function of the extrapolated range (Ta71b).
(6) Number-transmission coefficient;
    normal incidence (Ta75c),
    oblique incidence for the Al absorber (Ta76b).
(7) Average residual energy at a given depth of the absorber (Ta73, Ta74).
(8) Number-backscattering coefficient;
    normal incidence (Ta71a, It92),
    point-isotropic source (Ta79b).
(9) Fractional energy of backscattered electrons (Ta97).
(10) Energy-deposition distribution;
    in water (Ta88a, Ta91),
    in any material (Ta90a, Ta95),
    bremsstrahlung component (Ta94a).
(11) Energy-reflection ratios of photons under electron irradiation (It93a).
(12) Radiation or photon yield by electrons;
    fitted to continuous slowing-down approximation data (Ta90b)
    fitted to Monte Carlo data (Ta97).

A2.1.2 Approximate Expressions for Theoretical Parameters

(1) Continuous slowing-down approximation range (It71, Ta94b).
(2) Energy dissipation in the infinite medium for the plane perpendicular source (Ta72b, Ta72c) (an interpolation formula for the results of the moments method computation published in ref. Sp59).

(3) Energy deposition in the semi-infinite medium for oblique incidence (Ta75b) [an interpolation formula for the results of ETRAN (Rs69) tabulated in ref. Wa71].

(4) Parameter B appearing in Moli ère's theory of multiple scattering (Ta76a).

(5)* Function $\cos\gamma$ used in the formula for Mott-to-Rutherford single scattering ratio (Ta78).

(6)* Landau's distribution function for the ionization energy loss (Ta79a).

(7)* The quantity $\phi_{\mathrm{rad}}/\overline{\phi}$ proportional to the radiative energy-loss divided by the total energy of the incident electron (Ta77b).

(8)* Empirical correction factor to be applied to the Born-approximation result for the bremsstrahlung cross section (Ta77b).

A2.2 Interaction of Ions with Solids

(1) Number- and energy-backscattering coefficients of light ions for solids;
normal incidence (Ta81, Mo84, Ta85a, It85),
oblique incidence (Ta85b, It85).

(2) Number- and energy-backscattering coefficients of light and heavy ions for solids (Ta89).

A2.3 Cross Sections for Atomic and Molecular Collisions

(1) Charge transfer of hydrogen atoms and ionscolliding with gaseous atoms and molecules (Na87, Ta90c).

(2) Charge transfer of helium atoms and ions colliding with gaseous atoms and molecules (Ta87).

(3) Charge transfer of hydrogen atoms and ions colliding with metal vapors (Ta88b).

(4) Collisions of H, H_2, He and Li atoms and ions with atoms and molecules (It93b, It94, It95, It96).

(5) Ionization of H, H_2 and He by multiply charged ions (Ta92).

(6) Sigle-electron capture of multiply charged ions colliding with H, H_2 and He targets (Na89).

(7) State-selective electron capture in collisions of C^{6+} and O^{8+} ions with atomic hydrogen (Ja93).

A2.4 Nuclear Physics

(1) Deuteron photodisintegration cross-section between 10 and 35 MeV (Fu77a).

(2) Cross section of the reaction Be(γ, n) near the threshold (Fu82). [Barker (Ba83) has later shown that the experimental data presented in Fu82 can well be fitted by the formula derived from the R-matrix theory.]

A2.5 Radiation Measurement

(1) Thermal neutron distribution in the water bath;
Ra-Be source (Fu75, Fu76).
Cf source (Fu77b).

A2.6 Other Applications

(1) Length distribution of browned part of inner buds in irradiated onions (Fu77c).
(2) Dose-response curve for seedling height of barley (In80).

References

Ab70 Abramowitz, M. and I. A. Stegun, ed. Handbook of Mathematical Functions. (Dover, New York, 1970).

Ba83 Barker, F. C. The first excited state of ^{9}Be. Can. J. Phys. **61**, 1371 (1983).

Be69 Bevington, P. R. Data Reduction and Error Analysis for the Physical Sciences. (McGraw-Hill, New York, 1969).

Be70 Beveridge, G. S. G. and R. S. Schechter. Optimization: Theory and Practice. (McGraw-Hill, New York, 1970).

Bl52a Blatt, J. M. and V. F. Weisskopf. Theoretical Nuclear Physics. (John Wiley & Sons, New York, 1952).

Bl52b Blunck, O. Zur Reichweite schneller Elektronen. Z. Phys. **131**, 354 (1952).

Ch66 Cheney, E. W. Introduction to Approximation Theory. (McGraw-Hill, New York, 1966).

Da71 Daniel, C. and F. S. Wood. Fitting Equations to Data. (Wiley-Interscience, New York, 1971).

Fe96 Fernándes-Varea, J. M., P. Andreo and T. Tabata. Detour factors in water and plastic phantoms and their use for range and depth scaling in electron-beam dosimetry. Phys. Med. Biol. **41**, 1119 (1996).

Fl80 Fletcher, R. Practical Methods of Optimization. (Wiley, New York, 1980).

Fu75 Fujishiro, M., T. Tabata and J. Furuta. Simplification of the water bath method for calibrating a Ra-Be neutron source. Nucl. Instrum. & Methods **131**, 259 (1975).

Fu76 Fujishiro, M., T. Tabata and J. Furuta. Fitting function for the thermal neutron distribution in water due to a Ra-Be source. Nucl. Instrum. & Methods **137**, 201 (1976).

Fu77a Fukuda, K. and T. Tabata. Summary of the experimental deuteron photodisintegration cross-section between 10 and 35 MeV. Hoshasen (Ionizing Radiation) **4**, 56 (1977). in Japanese.

Fu77b Fujishiro, M., T. Tabata, J. Furuta, E. Hiraoka, T. Tsujimoto, K. Okamoto and K. Katsurayama. Fitting function for the thermal neutron distribution in water due to a ^{252}Cf source. Radioisotopes **26**, 245 (1977).

Fu77c Furuta, J., E. Hiraoka, S. Okamoto, M. Fujishiro, T. Kanazawa, T. Ohnishi, Y. Tsujii, T. Tabata, S. Hori and T. Ojima. The gamma-ray dose-rate effect on the sprout inhibition of onion bulbs. Food Irradiat., Jpn. **12**, 1 (1977). in Japanese.

Fu82 Fujishiro, M., T. Tabata, K. Okamoto and T. Tsujimoto. Cross section of the reaction ^{9}Be(γ, n) near threshold. Can. J. Phys. **60**, 1672 (1982).

Ga75 Gass, S. I. Linear Programming. 4th ed. (McGraw-Hill, New York, 1975).

Gi81 Gill, P. E., W. Murray and M. H. Wright. Practical Optimization. (Academic, New York, 1981).

Gr65 Green, A. E. S. and C. A. Barth. Calculations of ultraviolet molecular nitrogen emissions from the aurora. J. Geophys. Res. **70**, 1083 (1965).

Ha55 Hastings, C., Jr. Approximations for Digital Computers. (Princeton Univ. Press, Princeton, 1955).

Ho54 Hoerl, A. E. Fitting curves to data. In *Chemical Business Handbook,* J. H. Perry, ed. (McGraw-Hill, New York, 1954).

Ho75 Hofker, W. K. Implantation of boron in silicon. Philips Res. Rep. Suppl. No. 8 (1975).

In80 Inoue, M., R. Ito, T. Tabata and H. Hasegawa. Varietal differences in the repair of gamma-radiation-induced lesions in barley. Environ. Exp. Bot. **20**, 161 (1980).

It71 Ito, R., T. Tabata and S. Okabe. Function fitting to range-energy relation of electrons. Ann. Rep. Radiat. Center Osaka Prefect. **12**, 49 (1971).

It85 Ito, R., T. Tabata, N. Itoh, K. Morita, T. Kato and H. Tawara. Data on Backscattering Coefficients of Light Ions from Solids (A Revision). Inst. Plasma Phys. Nagoya Univ. Rep. IPPJ-AM-41 (1985).

It92 Ito, R., P. Andreo and T. Tabata. Reflection ratios of electrons and photons from solids. Bull. Univ. Osaka Pref. A **41**, 69 (1992).

It93a Ito, R., P. Andreo and T. Tabata. Reflection of electrons and photons from solids bombarded by 0.1- to 100-MeV electrons. Radiat. Phys. Chem. **42**, 761 (1993).

It93b Ito, R., T. Tabata, T. Shirai and R. A. Phaneuf. Analytic Cross Sections for Collisions of H, H_2, He and Li Atoms and Ions with Atoms and

Molecules. I. Report JAERI-M 93-117, Japan Atomic Energy Res. Inst. (1993).

It94 Ito, R., T. Tabata, T. Shirai and R. A. Phaneuf. Analytic Cross Sections for Collisions of H, H_2, He and Li Atoms and Ions with Atoms and Molecules. II. Report JAERI-Data/Code 94-005, Japan Atomic Energy Res. Inst. (1994).

It95 Ito, R., T. Tabata, T. Shirai and R. A. Phaneuf. Analytic Cross Sections for Collisions of H, H_2, He and Li Atoms and Ions with Atoms and Molecules. III. Report JAERI-Data/Code 95-008, Japan Atomic Energy Res. Inst. (1995).

It96 Ito, R., T. Tabata, T. Shirai and R. A. Phaneuf. Analytic Cross Sections for Collisions of H, H_2, He and Li Atoms and Ions with Atoms and Molecules. IV. Report JAERI-Data/Code 96-024, Japan Atomic Energy Res. Inst. (1996).

Ja93 Janev, R. K., R. A. Phaneuf, H. Tawara and T. Shirai. Recommended cross sections for state-selective electron capture in collisions of C^{6+} and O^{8+} ions with atomic hydrogen. At. Data & Nucl. Data Tables **55**, 201 (1993).

Ke59 Kelly, J. E., Jr. An application of linear programming to curve fitting. J. Soc. Indust. Appl. Math. **6**, 15 (1959).

Ko68 Kowalik, J. and M. R. Osborne. Methods for Unconstrained Optimization Problems. (American Elsevier, New York, 1968).

Le44 Levenberg, K. A method for the solution of certain nonlinear problems in least squares. Quart. Appl. Math. **2**, 164 (1944).

Ma63 Marquardt, D. W. An algorithm for least squares estimation of nonlinear parameters. J. Soc. Indust. Appl. Math. **11**, 431 (1963).

Me67 Meinardus, G. Approximatin of Functions: Theory and Numerical Methods. (Springer, Berlin, 1967).

Mo84 Morita, K., T. Tabata and R. Ito. Universal relations for reflection of keV light ions from solid targets. J. Nucl. Mater. **128 & 129**, 681 (1984).

Na81 Nakasima, R. Review of nuclear data evaluation. Journal of Atomic Energy Society of Japan **23**, 231 (1981). in Japanese.

Na87 Nakai, Y., T. Shirai, T. Tabata and R. Ito. Cross sections for charge transfer of hydrogen atoms and ions colliding with gaseous atoms and molecules. At. Data & Nucl. Data Tables **37**, 69 (1987).

Na89 Nakai, Y., T. Shirai, T. Tabata and R. Ito. A semiempirical formula for single-electron-capture cross sections of multiply charged ions. Phys. Scr. **T28**, 77 (1989).

Os67 Osborne, M. R. and G. A. Watson. On the best linear Chebyshev approximation. Comput. J. **10**, 172 (1967).

Os69 Osborne, M. R. and G. A. Watson. An algorithm for minimax approximation in the nonlinear case. Comput. J. **12**, 63 (1969).

Oy73 Oyanagi, Y. Theory and practice of χ^2-fit. Buturi **28**, 1973 (1973). in Japanese.

Oy79 Oyanagi, Y. Robust estimation method and data analysis. Buturi **34**, 884 (1979). in Japanese.

Ri64 Rice, J. R. The Approximation of Functions. (Addison Wesley, Reading, 1964).

Rs69 RSIC. ETRAN: Monte Carlo Code System for Electron and Photon Transport. Computer Code Collection CCC-107, Oak Ridge Nat. Lab. (1969).

Sp59 Spencer, L. V. Energy Dissipation by Fast Electrons. NBS Monograph 1, Nat. Bureau of Standards (1959).

Ta68 Tabata, T. A simple calculation for mean projected range of fast electrons. J. Appl. Phys. **39**, 5342 (1968).

Ta71a Tabata, T., R. Ito and S. Okabe. An empirical equation for the backscattering coefficient of electrons. Nucl. Instrum. & Methods **94**, 509 (1971).

Ta71b Tabata, T., R. Ito, S. Okabe and Y. Fujita. Extrapolated and projected ranges of 4- to 24-MeV electrons in elemental materials. J. Appl. Phys. **42**, 3361 (1971).

Ta71c Tabata, T., R. Ito, S. Okabe and Y. Fujita. Projected-range straggling of 4- to 24-MeV electrons in elemental materials. Jpn. J. Appl. Phys. **10**, 1503 (1971).

Ta72a Tabata, T., R. Ito and S. Okabe. Generalized semiempirical equations for the extrapolated range of electrons. Nucl. Instrum. & Methods **103**, 85 (1972).

Ta72b Tabata, T., R. Ito and S. Okabe. A fitting function for energy dissipation curves of fast electrons. Nucl. Sci. & Eng. **49**, 505 (1972).

Ta72c Tabata, T., R. Ito and S. Okabe. An interpolation procedure for energy dissipation curves of fast electrons. Ann. Rep. Radiat. Center Osaka Prefect. **13**, 59 (1972).

Ta73 Tabata, T. and R. Ito. Current and energy losses of electrons in titanium foils. Ann. Rep. Radiat. Center Osaka Prefect. **14**, 27 (1973).

Ta74 Tabata, T. and R. Ito. An algorithm for the energy deposition by fast electrons. Nucl. Sci. & Eng. **53**, 226 (1974).

Ta75a Tabata, T. and R. Ito. Effective treatment of the interpolation factor in Marquardt's nonlinear least-squares fit algorithm. Comput. J. **18**, 250 (1975).

Ta75b Tabata, T. and R. Ito. Parametric representation of the energy deposition by fast electrons under oblique incidence. Int. J. Appl. Radiat. & Isotopes **26**, 411 (1975).

Ta75c Tabata, T. and R. Ito. A generalized empirical equation for the transmission coefficient of electrons. Nucl. Instrum. & Methods **127**, 429 (1975).

Ta76a Tabata, T. and R. Ito. An improved interpolation formula for the parameter B in Molière's theory of multiple scattering. Jpn. J. Appl. Phys. **15**, 1583 (1976).

Ta76b Tabata, T. and R. Ito. An empirical relation for the transmission coefficient of electrons under oblique incidence. Nucl. Instrum. & Methods **136**, 533 (1976).

Ta77a Tabata, T. Private communication to W. E. McBride (1977).

Ta77b Tabata, T. and R. Ito. Interpolation formulas for quantities related to radiative energy-loss of electrons. Nucl. Instrum. & Methods **146**, 435 (1977).

Ta78 Tabata, T. and R. Ito. Approximation to cosγ appearing in the formula for the Coulomb scattering of relativistic electrons. Nucl. Sci. & Eng. **65**, 414 (1978).

Ta79a Tabata, T. and R. Ito. Approximations to Landau's distribution functions for the ionization energy loss of fast electrons. Nucl. Instrum. & Methods **158**, 521 (1979).

Ta79b Tabata, T. and R. Ito. Empirical and approximate expressions related to back- and multiple scatterings of fast electrons. Ann. Rep. Radiat. Center Osaka Prefect. **20**, 87 (1979).

Ta81 Tabata, T., R. Ito, K. Morita and Y. Itikawa. Empirical formulas for the backscattering of light ions from solids. Jpn. J. Appl. Phys. **20**, 1929 (1981).

Ta85a Tabata, T., R. Ito, K. Morita and H. Tawara. Unified empirical formulas for the backscattering coefficient of light ions. Nucl. Instrum. Methods B **9**, 113 (1985).

Ta85b Tabata, T., R. Ito, K. Morita and H. Tawara. Empirical formulas for the backscattering coefficients of light ions obliquely incident on solids. Radiat. Eff. **85**, 45 (1985).

Ta87 Tabata, T., R. Ito, Y. Nakai and T. Shirai. Cross Sections for Charge Transfer of Helium Atoms and Ions Colliding with Gaseous Atoms and Molecules. Radiat. Center Osaka Prefect. Tech. Rep. 7 (1987).

Ta88a Tabata, T. and R. Ito. Precision fitting to depth–dose curves of electron beams in the water phantom. Jpn. Radiol. Phys. **8**, 73 (1988).

Ta88b Tabata, T., R. Ito, Y. Nakai, T. Shirai, M. Sataka and T. Sugiura. Analytic cross sections for charge transfer of hydrogen atoms and ions colliding with metal vapors. Nucl. Instrum. Methods B **31**, 375 (1988).

Ta89 Tabata, T. and R. Ito. Present Status of Data Compilation on Ion Backscattering. Inst. Plasma Phys. Nagoya Univ. Rep. IPPJ-AM-64. (1989) p. 84.

Ta90a Tabata, T., R. Ito and S. Tsukui. Semiempirical algorithms for dose evaluation in electron-beam processing. Radiat. Phys. Chem. **35**, 821 (1990).

Ta90b Tabata, T. and R. Ito. An analytic expression for the radiation yield from condensed materials bombarded by electrons. Bull. Univ. Osaka Pref. A **39**, 241 (1990).

Ta90c Tabata, T., R. Ito, Y. Nakai, T. Shirai and Y. Funatake. Partial Cross Sections for Single-Electron Capture of Hydrogen Ions. Osaka Prefect. Radiat. Res. Inst. Tech. Rep. 11 (1990).

Ta91 Tabata, T., P. Andreo and R. Ito. Analytic fits to Monte Carlo calculated depth-dose curves of 1- to 50-MeV electrons in water. Nucl. Instrum. & Methods **B58**, 205 (1991).

Ta92 Tabata, T., R. Ito, T. Shirai, Y. Nakai, H. T. Hunter and R. A. Phaneuf. Extended scaling of cross-sections for the ionization of H, H_2 and He by multiply charged ions. At. Plasma-Mater. Interaction Data for Fusion **2**, 91 (1992).

Ta94a Tabata, T., P. Andreo, K. Shinoda and R. Ito. Energy deposition through radiative processes in absorbers irradiated by electron beams. Nucl. Instrum. Methods B **93**, 447 (1994).

Ta94b Tabata, T., K. Shinoda and R. Ito. An analytic formula for continuous slowing-down approximation ranges of 1-keV to 1-GeV electrons. Bull. Univ. Osaka Pref. A **43**, 77 (1994).

Ta95 Tabata, T. Theoretical evaluation of absorbed doses in materials irradiated by low-energy electron beams: A short review. Bull. Univ. Osaka Pref. A **44**, 41 (1995).

Ta96a Tabata, T., P. Andreo and K. Shinoda. An analytic formula for the extrapolated range of electrons in condensed materials. Nucl. Instrum. Methods B **119**, 463 (1996).

Ta96b Tabata, T., P. Andreo, K. Shinoda and R. Ito. Range distributions and projected ranges of 0.1- to 100-MeV electrons in elemental absorbers. Nucl. Instrum. Methods B **108**, 11 (1996).

Ta97 Tabata, T., P. Andreo and K. Shinoda. Fractional energy of backscattered electrons and photon yield by electrons. Bull. Osaka Pref. Univ. A **45** (in press).

Wa71 Watts, J. W. and M. O. Burrell. Electron and Bremsstrahlung Penetration and Dose Calculation. NASA Tech. Note NASA TN D-6385 (1971).

We64 Weber, K.-H. Eine einfache Reichweite–Energie-Beziehung für Elektronen im Energiebereich von 3 keV bis 3 MeV. Nucl. Instrum. & Methods **25**, 261 (1964).

FORTRAN Code and Test Input and Output Files for ALESQ

B1.1 COMMENT Statements for ALESQ

```
C  DESCRIPTION OF "ALESQ"
C
C
C   PURPOSE: TO SOLVE THE PROBLEM OF NONLINEAR LEAST-SQUARES FIT
C
C   COMPUTER USED FOR TEST: POWER MACINTOSH 8500/150
C
C   OPERATING SYSTEM: MAC OS 7.5.5
C
C   PROGRAMMING LANGUAGE: FORTRAN 77
C
C   FORTRAN COMPILER USED: ABSOFT FORTRAN 77
C
C   PERIPHERALS USED: NONE
C
C   FORMAT FOR CARD INPUT
C
C     MDS,NFUNC,NU,NX,MDW,MDP,IS,MDCK,NITLIM,NPRM.PRM    FREE FIELD
C     COMNT                                              A80
C     (FINDX(J),J=1,NX)                                  8(A8,2X)
C     YDATA(I),SIGMA(I),(X(J,I),J=1,NX)                  10X,7F10.4
C     MDRUN,MKEND                                        A4,1X,A1
C     (A(I),I=1.NU)                                      FREE FIELD;
C
C   MEANING OF INPUT VARIABLES
C
C     MDS       PARAMETER TO SELECT MODE OF INPUT FOR STARTING VALUES
C               0  LINEAR LEAST-SQUARES FIT: STARTING VALUES NOT USED
C               1  NONLINEAR LEAST-SQUARES FIT: STARTING VALUES GIVEN BY
C                  THE DATA FILE
C               2  NONLINEAR LEAST-SQUARES FIT: STARTING VALUES TAKEN
C                  FROM THE PREVIOUS RESULT
C               3  NONLINEAR LEAST-SQUARES FIT: STARTING VALUES GIVEN BY
C                  SUBROUTTNE "CALAIN"
C     NFUNC     NUMBER ATTACHED TO THE FITTING FUNCTION
C     NU        NUMBER OF UNKNOWNS
C     NX        NUMBER OF INDEPENDENT VARIABLES
C     MDW       PARAMETER TO SELECT MODE OF WEIGHTING
C               0  STATISTICAL ERRORS TO BE ASSI~NED FOR "SlGMA~I)"
C               1  ABSOLUTE ERRORS GIVEN AS "SIGMA(I)"
C               2  ABSOLUTE ERRORS TO BE REGARDED AS THE SAME
C               3  RELATIVE ERRORS GIVEN AS "SIGMA(I)"
C               4  RELATIVE ERRORS TO BE REGARDED AS THE SAME
C               5  RELATIVE ERRORS APPROXIMATELY THE SAME; TWO-STFP FIT
C               6  STEP 2 FOR MDW=5; THIS VALUE IS SET AUTOMATICALLY IN
C                  THE PROGRAM.
C     MDP       PARAMETER TO SELECT MODE OF PRINT
C               0  PRINT FINAL RESULT ONLY
C               1  PRINT INTERMEDIATE AND FINAL RESULTS
C     IS        DUMMY PARAMETER (SIGNIFICANT IN THE CODE "TSOLVE")
C     MDCK      PARAMETER TO SELECT EITHER NORMAL OR CHECK MODE
C               0 NORMAL MODE (SOLVE LEAST-SQUARES FIT PROBLEM)
C               1 CHECK MODE (CALCULATE ERRORS FOR GIVEN VALUES OF "A")
C     NITLIM    LIMIT TO THE NUMBER OF ITERATIONS (PUT 100, 1000 OR AS
C               YOU LIKE.)
C     NPRM      NUMBER OF ELMENTS USED IN THE ARRAY "PRM" (MAXIMUM
C               NUMBER: 5)
C     PRM       ARRAY OF PARAMETERS IN THE FITTING FUNCTION WHOSE VALUES
C               ARE TO BE VARIED FROM ONE RUN TO THE NEXT
C     COMNT     COMMENT ON THE PROBLEM
C     FINDX     ARRAY OF NAMES FOR INDEPENDENT VARIABLES
```

```
C     YDATA    ARRAY OF DATA POINTS FOR DEPENDENT VARlABLE
C     SIGMA    ARRAY OF PROBABLE ERRORS FOR "YDATA"
C              WHEN MDW=0, LEAVE THE FIELDS BLANK.
C              WHEN MDW=1, PUT ABSOLUTE ERROR FOR EACH DATA POINT.
C              WHEN MDW=2, PUT ABSOLUTE ERROR INTO "SIGMA(1)," OR LEAVE
C                          THE FIELDS BLANK.
C              WHEN MDW=3, PUT RELATIVE ERROR FOR EACH DATA POINT.
C              WHEN MDW=4, PUT RELATIVE ERROR INTO "SIGMA(1)," OR LEAVE
C                          THE FIELDS BLANK.
C              WHEN MDW=5, LEAVE THE FIELDS BLANK.
C     X        ARRAY OF DATA POINTS FOR INDEPENDENT VARIABLES
C     MDRUN    PARAMETER TO SELECT MODE OF THE NEXT RUN
C              "ISDT"     THE NEXT RUN IS TO BE MADE WITH THE SAME DATA
C                         SET ("SDT" MEANING THE SAME DATA).
C                         DATA FOR THE NEXT RUN:
C                           MDS, ...
C                           MDRUN,MKEND
C                           (A(I),I=1,NU)
C              "ISDF"     THE NEXT RUN IS TO BE MADE WITH THE SAME DATA
C                         SET AND THE SAME FUNCTION, BUT WITH A DIF-
C                         FEREN WEIGHT OR DIFFERENT "PRM" VALUES ("SDF"
C                         MEANING THE SAME DATA AND FUNCTION).
C                         DATA FOR THE NEXT RUN:
C                           MDS, ...
C                           MDRUN,MKEND
C                           (A(I),I=1,NU)
C              "ISFN"     THE NEXT RUN IS TO BE MADE WITH THE SAME
C                         FUNCTION ("SFN" MEANING THE SAME FUNCTION).
C                         DATA FOR THE NEXT RUN:
C                           YDATA(I),SIGMA(I),(X(J,I),J=1,NX)
C                           MDRUN,MKEND
C                           (A(I),I=1,NU)
C              "ISTP"     STOP AFTER THE PRESENT RUN.
C              OTHERWISE  THE NEXT RUN IS TO BE MADE WITH A DIFFERENT
C                         FUNCTION AND A DIFFERENT DATA SET.
C     MKEND    VARIABLE TO INDICATE THE END OF "YDATA-SIGMA-X" SET;
C              "*" SHOULD BE PUT.
C     A        ARRAY OF UNKNOWNS; PUT STARTING VALUES; NECESSARY ONLY
C              FOR MDS=1 OR MDCK=1
C
C   MEANING OF VARIABLES IN PARAMETER STATEMENT
C
C     MAXND    MAXIMUM NUMBER OF DATA POINTS ALLOWED
C     MAXND1   MAXND+l
C     MAXNU    MAXIMUM NUMBER OF UNKNOWNS ALLOWED
C     MAXNX    MAXIMUM NUMBER OF INDEPENDENT VARIABLES ALLOWED
C              FOR VALUES GREATER THAN 5. MODIFY FORMAT STATEMENTS 111
C              AND 112 IN SUBROUTINE "WRITER"
C
C
C
C  DESCRIPTION OF SUBPRO&RAMS
C
C    SUBROUTINE ALESQL
C      SOLVE THE PROBLEM OF LINEAR OR LINEARIZED LEAST-SQUARES FIT.
C
C    SUBROUTINE ALESQN
C      SOLVES THE PROBLEM OF NONLINEAR LEAST-SQUARES FIT.
C
C    SUBROUTINE APUT
C      ADDS OR SUBTRACTS CORRESPONDING VALUES OF TWO ARRAYS.
C
C    SUBROUTINE CALAIN
C      CALCULATES INITIAL VALUES OF UNKNOWNS BY THE RELATION GIVEN BY
C      THE USER.
C
C    SUBROUTINE CALDER
C      EVALUATES DERIVATIVES OF THE FITTING FUNCTION WITH RESPECT TO
C      UNKNOWNS FROM RIGHT DIFFERENCES.
```

```
C
C    SUBROUTINE CALFUN
C      EVALUATES THE FITTING FUNCTION. THE USER SHOULD PROVIDE THE
C      FUNCTION TO BE USED IN THIS SUBPROGRAM.
C
C    SUBROUTINE CHITST
C      PERFORMS CHI-SQUARE TEST BY USING FUNCTION SUBPROGRAMS "PCHISQ",
C      "GAMMA", AND "FACTOR".
C
C    SUBROUTINE MATINV
C      INVERTS A SYMMETRIC MATRIX AND CALCULATES ITS DETERMINANT.
C      ADAPTED FROM P. R. BEVINGTON: DATA REDUCTION AND ERROR ANALYSIS
C      FOR THE PHYSICAL SCIENCES (MCGRAW-HILL, NEW YORK, 1969)
C
C    SUBROUTINE OPTION
C      PERFORMS OPTIONAL CALCULATIONS OR PLOTTING USING THE RESULT OF
C      LEAST-SQUARES FIT.  NECESSARY STATEMENTS SHOULD BE PROVIDED BY
C      THE USER.
C
C    SUBROUTINE PLOT
C      PLOT CURVES. TO USE THIS SUBROUTINE THE USER SHOULD PROVIDE
C      STATEMENTS IN THE SUBROUTINE "OPTION".
C
C    SUBROUTINE PUT
C      PUTS VALUES OF AN ARRAY INTO ANOTHER ARRAY.
C
C    SUBROUTINE RDDATA
C      READS VALUES OF DATA.
C
C    SUBROUTINE WEIGHT
C      CALCULATES WEIGHTS FOR DATA POINTS.
C
C    SUBROUTINE WRITER
C      WRITES INPUT VARIABLES, INTERMEDIATE AND FINAL RESULTS.
C
C
C
C  MEANING OF PARAMETERS IN THE SUBPROGRAMS (SEE ALSO INPUT VARIABLES
C     AND VARIABLES IN PARAMETER STATEMENT)
C
C
C   PARAMETERS WITH COMMON MEANING
C
C      A       ARRAY OF UNKNOWNS
C      CHI     RMS ABSOLUTE ERROR OF FIT
C      DA      ARRAY OF INCREMENTS OR ERRORS FOR UNKNOWNS "A"
C      FLAMB   PROPORTION OF GRADIENT SEARCH INCLUDED
C      ILL     INDICATOR FOR ABNORMAL STATUS
C               0  NORMAL
C               1  NUMBER OF FUNCTION EVALUATION LIMIT
C               2  LAMBDA TOO LARGE
C               3  ELEMENT OF ERROR MATRIX NEGATIVE
C               4  ERRORS INDEFINITE; ND=NU
C               5  ILL=1 AND ILL=3
C               6  ILL=1 AND ILL=4
C               7  ILL=2 AND ILL=3
C               8  ILL=2 AND ILL=4
C               9  FUNCTION UNDEFINED IN "FUNC"
C              10  VALUE OF "NX," "NU" OR "ND" NOT ALLOWED
C              11  "A" UNDEFINED IN "CALAIN"
C      ND      NUMBER OF DATA POINTS
C      YCAL    ARRAY OF CALCULATED VALUES OF DEPENDENT VARIABLE
C
C
C   PARAMETERS IN EACH SUBPROGRAM
C
C    SUBROUTINE ALESQL
C      MODE    PARAMETER TO SELECT THE MODE OF THE SUBROUTINE
C              1  CALCULATE INCREMENTS "DA" FOR "A"
```

```
C             2  CALCULATE "CHI"
C             3  CALCULATE ERRORS FOR "A"
C             4  CALCULATE "DA" WITHOUT CHANGING "PAA" AND "PAL"
C
C    SUBROUTINE ALESQN
C      AM      ARRAY OF THE BEST PREVIOUS VALUES FOR "A"
C
C    SUBROUTINE APUT
C      AL      OUTPUT ARRAY
C      AR1     INPUT ARRAY
C      AR2     INPUT ARRAY
C      NDIM    DIMENSION OF THE ARRAYS
C      MODE    PARAMETER TO SELECT OUTPUT
C              =0 OR >0  AL(I)=ARl(I)+AR2(I)
C              <0        AL(I)=ARl(I)-AR2(I)
C
C    SUBROUTINE CALDER
C      H       STEP TO CALCULATE THE DIFFERENCE OF THE FUNCTION (DEFAULT
C              VALUE IS 1.D-5 AS DETERMINED BY THE STATEMENT 16 OF
C              SUBROUTINE "ALESQL"
C
C    SUBROUTINE CALFUN
C      NCAL    NUMBER OF POINTS TO BE CALCULATED
C
C    SUBROUTINE CHITST
C      CHISQR  VALUE OF REDUCED CHI-SQUARE (INPUT)
C      DGFREE  DEGREE OF FREEDOM (INPUT)
C      PROB    INTEGRAL PROBABILITY ABOVE "CHISQR" (OUTPUT)
C      IERR    INDICATOR FOR ERROR (OUTPUT)
C
C    SUBROUTINE MATINV
C      ARRAY   INPUT MATRIX WHICH IS REPLACED BY ITS INVERSE
C      IK      ARRAY TO STORE ORDINAL NUMBERS OF ROWS
C      JK      ARRAY TO STORE ORDINAL NUMBERS OF COLUMNS
C      MAXORD  MAXIMUM OF "NORDER" ALLOWED
C      NORDER  DEGREE OF MATRIX (ORDER OF DETERMINANT)
C      DET     DETERMINANT OF INPUT MATRIX
C
C    SUBROUTINE PUT
C      AL      OUTPUT ARRAY
C      AR      INPUT ARRAY
C      NDIM    DIMENSION OF THE ARRAYS
C
C    SUBROUTINE WEIGHT
C      FND     REAL-TYPE VARIABLE OF "ND"
C      NDl     ND+l
C      P       ARRAY OF WEIGHTS FOR DATA POINTS
C
C    SUBROUTINE WRITER
C      MODE    PARAMETER TO SELECT THE MODE OF THE SUBROUTINE
C              1  WRITE INTERMEDIATE RESULTS
C              2  WRITE FINAL RESULT
C              3  WRITE COMMENT AND PARAMETERS AT THE START
C              4  WRITE NUMBER OF DATA AT THE START
C      ITR     NUMBER OF ITERATION
C      INITR   NUMBER OF INTERNAL ITERATION
```

B1.2 FORTRAN Code for ALESQ

B1.2.1 ALESQMN.f File

```
      PROGRAM ALESQ
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER *4 MDRUN,LCOMNT,FINDX*8,COMNT*80
      PARAMETER (MAXND=300,MAXND1=301,MAXNU=10,MAXNX=3)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
```

```
      COMMON/BLKARY/FINDX(MAXNX),X(MAXNX,MAXND1),P(MAXND),YDATA(MAXND1),
     +  SIGMA(MAXND1),A(MAXNU),AM(MAXNU),DA(MAXNU),YCAL(MAXND),
     +  YDATAM(MAXND),YCALM(MAXND),PAA(MAXNU,MAXNU),PAL(MAXNU)
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      COMMON/BLKNLN/NITLIM
      OPEN(5,FILE="",STATUS="OLD")
      OPEN(6,FILE="")
      MDRUN='ISTP'
    3 READ(5,*) MDS,NFUNC,NU,NX,MDW,MDP,IS,MDCK,NITLIM,NPRM,
     +  (PRM(J),J=1,NPRM)
      IF(MDRUN.EQ.'ISDT'.OR.MDRUN.EQ.'ISDF') GOTO 5
      READ(5,501) COMNT
    5 CALL WRITER(3,0,0.D0,0.D0,0)
      IF(NX.LE.0.OR.NX.GT.MAXNX) GOTO 18
      IF(NU.LE.0.OR.NU.GT.MAXNU) GOTO 18
      IF(MDRUN.EQ.'ISDT'.OR.MDRUN.EQ.'ISDF') GOTO 9
    7 READ(5,502) (FINDX(J),J=1,NX)
    8 CALL RDDATA(MAXNX,MAXND,MAXND1,YDATA,YDATAM,SIGMA,X,*18)
      CALL WRITER(4,0,0.D0,0.D0,0)
    9 CALL WEIGHT(ND,FND,ND1,P,YDATA,SIGMA,YCAL,MDW)
   10 CALL ALESQN(MAXNU,MAXND,MAXND1,MAXNX,A,AM,DA,YDATA,YCAL,YCALM,X,
     +  ILL,*13)
      IF(MDW.EQ.6) GOTO 8
   11 CALL OPTION(MAXNU,MAXND,MAXND1,MAXNX,A,DA,X,YDATA,YCAL,ILL)
   13 IF(MDRUN.EQ.'ISFN') GOTO 8
      IF(MDRUN.EQ.'ISTP') GOTO 20
      GOTO 3
   18 ILL=10
      CALL WRITER(2,0,0.D0,0.D0,ILL)
   20 CLOSE(5)
      CLOSE(6)
      STOP
  501 FORMAT(A80)
  502 FORMAT(8(A8,2X))
      END
C
      SUBROUTINE ALESQL(MODE,FLAMB,CHI,ILL,*)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER FINDX*8
      PARAMETER (MAXND=300,MAXND1=301,MAXNU=10,MAXNX=3)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKARY/FINDX(MAXNX),X(MAXNX,MAXND1),P(MAXND),YDATA(MAXND1),
     +  SIGMA(MAXND1),A(MAXNU),AM(MAXNU),DA(MAXNU),YCAL(MAXND),
     +  YDATAM(MAXND),YCALM(MAXND),PAA(MAXNU,MAXNU),PAL(MAXNU)
      COMMON/BLKDER/DERIV(MAXNU,MAXND)
      DIMENSION ARRAY(MAXNU,MAXNU),IK(MAXNU),JK(MAXNU)
      IF(MDCK.NE.0.AND.MODE.EQ.3) GOTO 81
      IF(MODE.EQ.4) GOTO 45
    5 IF(MODE.EQ.1.AND.MDS.EQ.0) GOTO 10
      IF(MODE.NE.2) GOTO 15
   10 CALL CALFUN(ND,ILL)
      IF(ILL.NE.0) RETURN 1
      NFE=NFE+1
      IF(MODE.EQ.2) GOTO 85
   15 CALL CALDER(1.D-5,ILL)
      DO 35 I=1,NU
      DO 35 J=I,NU
      IF(I.NE.1.OR.MODE.EQ.3) GOTO 20
      PAL(J)=0.D0
   20 PAA(I,J)=0.D0
      DO 35 L=1,ND
      AIL=DERIV(I,L)
      IF(J.EQ.I) GOTO 25
      AJL=DERIV(J,L)
      GOTO 30
   25 AJL=AIL
   30 PAA(I,J)=P(L)*AIL*AJL+PAA(I,J)
      IF(I.NE.1.OR.MODE.EQ.3) GOTO 35
```

```
      PAL(J)=(YDATA(L)-YCAL(L))*AJL*P(L)+PAL(J)
   35 CONTINUE
      NFE=NFE+NU
      DO 40 I=2,NU
      DO 40 J=1,I-1
   40 PAA(I,J)=PAA(J,I)
   45 DO 55 J=1,NU
      DO 50 K=1,NU
      S=DSQRT(PAA(J,J)*PAA(K,K))
   50 ARRAY(J,K)=PAA(J,K)/S
   55 ARRAY(J,J)=1.D0+FLAMB
      CALL MATINV(ARRAY,IK,JK,MAXNU,NU,S)
      IF(MODE.EQ.3) GOTO 65
   56 DO 60 J=1,NU
      DA(J)=0.D0
      DO 60 K=1,NU
      S=DSQRT(PAA(J,J)*PAA(K,K))
   60 DA(J)=PAL(K)*DBLE(ARRAY(J,K))/S+DA(J)
      RETURN
   65 DO 80 I=1,NU
      S=DBLE(ARRAY(I,I))/PAA(I,I)
      IF(S.LE.0.D0) GOTO 70
   66 DA(I)=DSQRT(S)*CHI
      GOTO 80
   70 ILL=3+2*ILL
      DA(I)=0.D0
   80 CONTINUE
      RETURN
   81 DO 82 I=1,NU
   82 DA(I)=0.D0
      RETURN
   85 PLL=0.D0
      DO 90 L=1,ND
   90 PLL=P(L)*(YDATA(L)-YCAL(L))**2+PLL
      IF(ND.LE.NU) GOTO 95
   93 CHI=DSQRT(PLL/DBLE(ND-NU))
      RETURN
   95 ILL=4+2*ILL
   96 RETURN 1
      END
C
      SUBROUTINE CALDER(H,ILL)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER FINDX*8
      PARAMETER (MAXND=300,MAXND1=301,MAXNU=10,MAXNX=3)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKARY/FINDX(MAXNX),X(MAXNX,MAXND1),P(MAXND),YDATA(MAXND1),
     +  SIGMA(MAXND1),A(MAXNU),AM(MAXNU),DA(MAXNU),YCAL(MAXND),
     +  YDATAM(MAXND),YCALM(MAXND),PAA(MAXNU,MAXNU),PAL(MAXNU)
      COMMON/BLKDER/DERIV(MAXNU,MAXND)
      DIMENSION YCALM(MAXND)
      DO 1 J=1,ND
    1 YCALM(J)=YCAL(J)
      DO 20 I=1,NU
      AI=A(I)
      DELTA=H
      IF(AI.EQ.0.D0) GOTO 10
    5 DELTA=AI*DELTA
   10 A(I)=AI+DELTA
      CALL CALFUN(ND,ILL)
      DO 15 J=1,ND
   15 DERIV(I,J)=(YCAL(J)-YCALM(J))/DELTA
      A(I)=AI
   20 CONTINUE
      DO 30 J=1,ND
   30 YCAL(J)=YCALM(J)
      RETURN
      END
```

```
C
      SUBROUTINE WRITER(MODE,ITR,CHI,FLAMB,ILL)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER *4 MDRUN,LCOMNT,COMNT*80
      CHARACTER *8 FINDX,FINDXN,FINDXM,BLNK,LBLM,WGTLET
      PARAMETER (MAXND=300,MAXND1=301,MAXNU=10,MAXNX=3)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKARY/FINDX(MAXNX),X(MAXNX,MAXND1),P(MAXND),YDATA(MAXND1),
     +  SIGMA(MAXND1),A(MAXNU),AM(MAXNU),DA(MAXNU),YCAL(MAXND),
     +  YDATAM(MAXND),YCALM(MAXND),PAA(MAXNU,MAXNU),PAL(MAXNU)
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      COMMON/BLKOPT/RMS
      COMMON/BLKNLN/NITLIM
      DIMENSION YD(MAXND),BLNK(MAXNX),FINDXN(4),FINDXM(4),LBLM(2)
      DATA BLNK/MAXNX*'        '/
      DATA FINDXN/'SIGMA   ',2*'        ','RESIDUAL'/
      DATA FINDXM/'Y DATA  ','LN(Y DT)','Y FITTED','LN(Y FT)'/
      DATA LBLM/' (ABS.) ',' (REL.) '/
      IF(MODE.NE.4) GOTO 3
      WRITE(6,104) ND
      RETURN
    3 IF(MODE.NE.3) GOTO 5
      WRITE(6,100) COMNT,MDS,NFUNC,NU,NX,MDW,MDP,MDCK,NITLIM,NPRM
      IF(NPRM.NE.0) WRITE(6,101) (PRM(I),I=1,NPRM)
      RETURN
    5 CONTINUE
      IF(MODE.EQ.2) GOTO 45
      IF(MDP.EQ.0) RETURN
   25 NUM=MIN0(NU,10)
      IF(ITR.EQ.0) WRITE(6,102) (J,J=1,NUM)
   30 WRITE(6,103) ITR,FLAMB,CHI,(A(J),J=1,NU)
      RETURN
   45 CONTINUE
   47 IF(ILL.EQ.0) GOTO 50
   48 GOTO (50,50,50,50,50,50,50,50,205,206,207,208),ILL
   50 CHIR=SIGMA(ND1)*CHI
      FRD=FND-DBLE(NU)
      IF(FRD.GT.0.D0) GOTO 52
   51 RMS=0.D0
      GOTO 55
   52 RMS=DSQRT(FRD/FND)*CHIR
   55 YDMAX=0.D0
      S=0.D0
      DO 65 I=1,ND
      YD(I)=YCAL(I)-YDATA(I)
      GOTO (58,58,57,57,58,57),MDW
   57 YD(I)=YD(I)/YDATA(I)
   58 ABSYD=DABS(YD(I))
      IF(ABSYD.LT.YDMAX) GOTO 62
   60 YDMAX=ABSYD
      LMAX=I
      IF(MDW.EQ.5) GOTO 65
   62 S=S+((YCAL(I)-YDATA(I))/YCAL(I))**2
   65 CONTINUE
      WRITE(6,105) ITR,NFE
      IF(MDW.EQ.0) GOTO 72
   70 GOTO (72,81,72,81,81,81),MDW
   72 CHISQR=0.D0
      DO 75 I=1,ND
   75 CHISQR=CHISQR+P(I)*((YCAL(I)-YDATA(I))/SIGMA(I))**2
      CHISQR=CHISQR/FRD
      CALL CHITST(CHISQR,FRD,PCHISQ,IERR)
      WRITE(6,106) CHISQR,PCHISQ
      IF(IERR.EQ.0) GOTO 81
   80 WRITE(6,107)
   81 WRITE(6,108) CHI,CHIR,RMS,YDMAX,LMAX
      IF(MDW.EQ.5) GOTO 83
      S=DSQRT(S/FND)
```

```
      WRITE(6,109) S
   83 WRITE(6,110) (J,A(J),DA(J),J=1,NU)
      WGTLET=LBLM(1)
      FINDXN(2)=FINDXM(1)
      FINDXN(3)=FINDXM(3)
      GOTO (90,90,85,85,88,85),MDW
   85 WGTLET=LBLM(2)
      GOTO 90
   88 FINDXN(2)=FINDXM(2)
      FINDXN(3)=FINDXM(4)
   90 WRITE(6,111) (FINDX(ID),ID=1,NX),FINDXN
      WRITE(6,112) (BLNK(ID),ID=1,NX),WGTLET
      DO 94 I=1,ND
   94 WRITE(6,113) I,P(I),(X(J,I),J=1,NX),SIGMA(I),YDATA(I),YCAL(I),
     +YD(I)
      IF(MDW.EQ.6) MDW=7
      IF(MDW.NE.5) GOTO 96
      MDW=6
      GOTO 96
   96 IF(ILL.EQ.0) GOTO 98
      GOTO (201,202,203,204,201,202,201,202,205,206,207,208),ILL
  201 WRITE(6,115) NFE
      GOTO (98,98,98,98,203,98,203,98,98,98,98),ILL
  202 WRITE(6,116)
      GOTO (98,98,98,98,98,204,98,204,98,98,98),ILL
  203 WRITE(6,117)
      GOTO 98
  204 WRITE(6,118)
      GOTO 98
  205 WRITE(6,119)
      GOTO 98
  206 WRITE(6,120)
      GOTO 98
  207 WRITE(6,121)
      GOTO 98
  208 WRITE(6,122)
   98 RETURN
  100 FORMAT(1H /' LEAST-SQUARES FIT PROGRAM ALESQ',4XA80//
     +'   MODE OF INPUT FOR STARTING VALUES      (MDS) =',I4/
     +'   FUNCTION NUMBER                      (NFUNC) =',I4/
     +'   NUMBER OF UNKNOWNS                     (NU) =',I4/
     +'   NUMBER OF INDEPENDENT VARIABLES        (NX) =',I4/
     +'   MODE OF WEIGHTING                     (MDW) =',I4/
     +'   MODE OF PRINT                         (MDP) =',I4/
     +'   MODE OF FIT/CHECK                    (MDCK) =',I4/
     +'   LIMIT TO THE NUMBER OF ITERATIONS   (NITLIM) =',I4/
     +'   NUMBER OF PARAMETERS                  (NPRM) =',I4)
  101 FORMAT('   PRM =',1P5E11.3)
  102 FORMAT(' '/' NO. LAMBDA  ERROR     VALUES OF UNKNOWNS A(I)'/1H ,
     +20X,10(5X,I2,4X))
  103 FORMAT(1H ,I3,1PE8.1,E9.2,10E11.3/1H ,20X,10E11.3)
  104 FORMAT('   NUMBER OF DATA                          (ND) =',I4)
  105 FORMAT(/'   NUMBER OF ITERATIONS                     =',I4/
     +'   NUMBER OF FUNCTION EVALUATIONS               =',I4/)
  106 FORMAT('   REDUCEDD CHI-SQUARE =',1PE10.3,'  PROBABILITY =',
     +0PF5.3/)
  107 FORMAT(1H ,10X,'DEGREE OF FREEDOM EXCEEDS 194 AND WAS SET AS 194')
  108 FORMAT('   RMS ABSOLUTE ERROR OF FIT                   =',1PE12.5
     +/'   RMS RELATIVE ERROR OF FIT                    =',E12.5/
     +'   RMS RELATIVE RESIDUAL                       =',E12.5/
     +'   MAXIMUM IN MAGNITUDE OF RESIDUAL            =',E10.3,' AT DAT
     +A NO.',I4/)
  109 FORMAT('   RELATIVE RMS DEVIATION OF DATA FROM EQUATION =',1PE9.2/
     +1H )
  110 FORMAT('   A(',I2,') =',1PE17.9,' +/- ',E8.1)
  111 FORMAT(' '/'  NO.   WEIGHT ',9(3X,A8))
  112 FORMAT(1H ,47X,6(3X,A8))
  113 FORMAT(1H ,I4,1P10E11.3)
  115 FORMAT(/' *** NUMBER OF FUNCTION EVALUATION',I6,' ***')
```

```
116 FORMAT(/' *** LAMBDA TOO LARGE ***')
117 FORMAT(/' *** ELEMENT OF ERROR MATRIX NEGATIVE ***')
118 FORMAT(/' *** ERRORS INDEFINITE; ND=NU ***')
119 FORMAT(/' *** FUNCTION UNDEFINED IN FUNC ***')
120 FORMAT(/' *** VALUE OF NX, NU OR ND NOT ALLOWED ***')
121 FORMAT(/' *** "A" UNDEFINED IN CALAIN ***')
122 FORMAT(/' *** "MDS=0 AND MDCK=1" NOT ALLOWED; FOR MDCK=1, PUT MDS=
   +1 EVEN WHEN THE FUNCTION IS LINEAR ***')
    END
```

B1.2.2 ALESQSUB.f File

```
    SUBROUTINE ALESQN(MAXNU,MAXND,MAXND1,MAXNX,A,AM,DA,YDATA,YCAL,
   +  YCALM,X,ILL,*)
    IMPLICIT DOUBLE PRECISION (A-H,O-Z)
    COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
   +  NFE
    COMMON/BLKNLN/NITLIM
    DIMENSION A(MAXNU),AM(MAXNU),DA(MAXNU),FACT(5),YDATA(MAXND1),
   +  X(MAXNX,MAXND1),YCAL(MAXND),YCALM(MAXND)
    DATA FACT/1.33,1.78,3.16,10.,100./
    ILL=0
    FLAMB=0.D0
    NFE=0
    IF(MDCK.NE.0) GOTO 35
  3 NFELIM=NITLIM*(1+NU)
    IF(MDS.NE.0) GOTO 30
  5 DO 10 I=1,NU
 10 A(I)=0.D0
 15 MODE=1
 20 CALL ALESQL(MODE,FLAMB,CHI,ILL,*205)
    CALL APUT(A,A,DA,NU,1)
    CALL ALESQL(2,FLAMB,CHI,ILL,*205)
    IF(MDS.NE.0) GOTO 60
 25 CALL ALESQL(3,0.D0,CHI,ILL,*205)
 26 CALL WRITER(2,ITR,CHI,FLAMB,ILL)
    RETURN
 30 ITR=0
    MHIST=0
    IF(MDW.EQ.6) GOTO 50
 35 IF(MDS.NE.0) GOTO 36
    ILL=12
    CALL WRITER(2,ITR,CHI,FLAMB,ILL)
    RETURN 1
 36 GOTO (40,50,45),MDS
 40 READ(5,*) (A(I),I=1,NU)
    GOTO 50
 45 CALL CALAIN(MAXNU,MAXND1,MAXNX,A,YDATA,X,ILL)
    IF(ILL.EQ.7) GOTO 205
 50 CALL ALESQL(2,FLAMB,CHI,ILL,*205)
    IF(MDCK.NE.0) GOTO 25
 53 CALL WRITER(1,ITR,CHI,FLAMB,ILL)
 55 CHIM=CHI
    FLAMBM=FLAMB
    CALL PUT(AM,A,NU)
    CALL PUT(YCALM,YCAL,ND)
    IF(FLAMB) 80,15,80
 60 IF(CHI.GT.CHIM) GOTO 120
 65 ITR=ITR+1
    CALL WRITER(1,ITR,CHI,FLAMB,ILL)
    IF(FLAMB) 55,70,55
 70 DO 75 I=1,NU
    IF(DABS(DA(I)/A(I)).GE.5.D-5) GOTO 55
 75 CONTINUE
    GOTO 25
 80 IF(MHIST.GE.0) GOTO 105
 85 IF(MHIST.NE.-2) GOTO 95
 90 IF(NFACT.LT.5) NFACT=NFACT+1
```

```
      GOTO 100
   95 MHIST=-2
  100 FLAMB=FLAMB/FACT(NFACT)
      IF(FLAMB.LT.1.D-8) FLAMB=0.D0
      GOTO 15
  105 IF(MHIST.NE.2) GOTO 115
  110 IF(NFACT.GT.1) NFACT=NFACT-1
  115 MHIST=-1
      GOTO 100
  120 IF(FLAMB.GT.1.D8) GOTO 195
  125 IF(NFE.GT.NFELIM) GOTO 190
  130 IF(FLAMB.NE.0.D0) GOTO 155
  135 IF(MHIST.NE.0) GOTO 150
  140 FLAMB=1.D-3
      NFACT=4
  145 CALL PUT(A,AM,NU)
      CALL PUT(YCAL,YCALM,ND)
      MODE=4
      GOTO 20
  150 FLAMB=FLAMBM
      GOTO 145
  155 IF(MHIST.GE.0) GOTO 170
  160 IF(NFACT.GT.1) NFACT=NFACT-1
      MHIST=1
  165 FLAMB=FLAMB*FACT(NFACT)
      GOTO 145
  170 IF(MHIST.NE.2) GOTO 180
  175 IF(NFACT.LT.3) NFACT=3
      GOTO 165
  180 MHIST=2
      GOTO 165
  190 ILL=1
      GOTO 200
  195 ILL=2
  200 CHI=CHIM
      CALL PUT(A,AM,NU)
      CALL PUT(YCAL,YCALM,ND)
      GOTO 25
  205 CALL WRITER(2,ITR,CHI,FLAMB,ILL)
      RETURN 1
      END
C
      SUBROUTINE APUT(AL,AR1,AR2,NDIM,MODE)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION AL(NDIM),AR1(NDIM),AR2(NDIM)
      DO 5 I=1,NDIM
      IF(MODE.LT.0) GOTO 4
    3 AL(I)=AR1(I)+AR2(I)
      GOTO 5
    4 AL(I)=AR1(I)-AR2(I)
    5 CONTINUE
      RETURN
      END
C
      SUBROUTINE CHITST(CHISQR,DGFREE,PROB,IERR)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      IERR=0
      IF(CHISQR.GT.1.76D2) GOTO 10
    5 IF(DGFREE.LE.1.76D2) GOTO 7
      IERR=1
    7 NFREE=IDINT(DGFREE)
      PROB=PCHISQ(CHISQR,NFREE)
      RETURN
   10 PROB=0.D0
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION PCHISQ(CHISQR,NFREE)
C       EVALUATE PROBABILITY FOR EXCEEDING CHI SQUARE
```

```
C       TAKEN FROM P. R. BEVINGTON, "DATA REDUCTION AND ERROR ANALYSIS
C       FOR THE PHYSICAL SCIENCES" (MCGRAW-HILL, NEW YORK, 1969)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
   11 IF(NFREE) 12,12,14
   12 PCHISQ=0.D0
      GOTO 60
   14 FREE=NFREE
      Z=CHISQR*FREE/2.D0
      NEVEN=2*(NFREE/2)
      IF(NFREE-NEVEN) 21,21,41
C       NUMBER OF DEGREES OF FREEDOM IS EVEN
   21 IMAX=NFREE/2
      TERM=1.D0
      SUM=0.D0
   31 DO 34 I=1,IMAX
      FI=I
      SUM=SUM+TERM
   34 TERM=TERM*Z/FI
   35 PCHISQ=SUM*DEXP(-Z)
      GOTO 60
C       NUMBER OF DEGREES OF FREEDOM IS ODD
   41 IF(Z-2.5D1) 44,44,42
   42 Z=CHISQR*(FREE-1.D0)/2.D0
      GOTO 21
   44 PWR=FREE/2.D0
      TERM=1.D0
      SUM=TERM/PWR
   51 DO 56 I=1,1000
      FI=I
      TERM=-TERM*Z/FI
      SUM=SUM+TERM/(PWR+FI)
   55 IF(DABS(TERM/SUM)-1.D-5) 57,57,56
   56 CONTINUE
   57 PCHISQ=1.D0-(Z**PWR)*SUM/GAMMA(PWR)
   60 RETURN
      END
C
      DOUBLE PRECISION FUNCTION GAMMA(X)
C       CALCULATES THE GAMMA FUNCTION FOR INTEGERS AND HALF-INTEGERS
C       TAKEN FROM P. R. BEVINGTON, "DATA REDUCTION AND ERROR ANALYSIS
C       FOR THE PHYSICAL SCIENCES" (MCGRAW-HILL, NEW YORK, 1969)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C       INTEGERIZE ARGUMENT
   11 N=X-2.5D-1
      XN=N
   13 IF(X-XN-7.5D-1) 31,31,21
C       ARGUMENT IS INTEGER
   21 GAMMA=FACTOR(N)
      GOTO 60
C       ARGUMENT IS HALF INTEGER
   31 PROD=1.77245385D0
      IF(N) 44,44,33
   33 IF(N-10) 41,41,51
   41 DO 43 I=1,N
      FI=I
   43 PROD=PROD*(FI-5.D-1)
   44 GAMMA=PROD
      GOTO 60
   51 SUM=0.D0
      DO 54 I=11,N
      FI=I
   54 SUM=SUM+DLOG(FI-5.D-1)
   55 GAMMA=PROD*6.393838623D5*DEXP(SUM)
   60 RETURN
      END
C
      DOUBLE PRECISION FUNCTION FACTOR(N)
C       CALCULATES FACTORIAL FUNCTION FOR INTEGERS
C       TAKEN FROM P. R. BEVINGTON, "DATA REDUCTION AND ERROR ANALYSIS
```

```
C       FOR THE PHYSICAL SCIENCES" (MCGRAW-HILL, NEW YORK, 1969)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
   11 FACTOR=1.D0
      IF(N-1) 40,40,13
   13 IF(N-10) 21,21,31
C       N LESS THAN 11
   21 DO 23 I=2,N
      FI=I
   23 FACTOR=FACTOR*FI
      GOTO 40
C       N GREATER THAN 10
   31 SUM=0.D0
      DO 34 I=11,N
      FI=I
   34 SUM=SUM+DLOG(FI)
   35 FACTOR=3.6288D6*DEXP(SUM)
   40 RETURN
      END
C
      SUBROUTINE MATINV(ARRAY,IK,JK,KMAX,NORDER,DET)
C       TAKEN FROM P. R. BEVINGTON, "DATA REDUCTION AND ERROR ANALYSIS
C       FOR THE PHYSICAL SCIENCES" (MCGRAW-HILL, NEW YORK, 1969)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION ARRAY(KMAX,KMAX),IK(KMAX),JK(KMAX)
   10 DET=1.D0
   11 DO 100 K=1,NORDER
      AMAX=0.D0
   21 DO 30 I=K,NORDER
      DO 30 J=K,NORDER
   23 IF(DABS(AMAX).GT.DABS(ARRAY(I,J))) GOTO 30
   24 AMAX=ARRAY(I,J)
      IK(K)=I
      JK(K)=J
   30 CONTINUE
   31 IF(AMAX.NE.0.D0) GOTO 41
   32 DET=0.D0
      GOTO 140
   41 I=IK(K)
      IF(I-K) 21,51,43
   43 DO 50 J=1,NORDER
      SAVE=ARRAY(K,J)
      ARRAY(K,J)=ARRAY(I,J)
   50 ARRAY(I,J)=-SAVE
   51 J=JK(K)
      IF(J-K) 21,61,53
   53 DO 60 I=1,NORDER
      SAVE=ARRAY(I,K)
      ARRAY(I,K)=ARRAY(I,J)
   60 ARRAY(I,J)=-SAVE
   61 DO 70 I=1,NORDER
      IF(I.EQ.K) GOTO 70
   63 ARRAY(I,K)=-ARRAY(I,K)/AMAX
   70 CONTINUE
   71 DO 80 I=1,NORDER
      DO 80 J=1,NORDER
      IF(I.EQ.K) GOTO 80
   74 IF(J.EQ.K) GOTO 80
   75 ARRAY(I,J)=ARRAY(I,K)*ARRAY(K,J)+ARRAY(I,J)
   80 CONTINUE
   81 DO 90 J=1,NORDER
      IF(J.EQ.K) GOTO 90
   83 ARRAY(K,J)=ARRAY(K,J)/AMAX
   90 CONTINUE
      ARRAY(K,K)=1.D0/AMAX
  100 DET=DET*DBLE(AMAX)
  101 DO 130 L=1,NORDER
      K=NORDER-L+1
      J=IK(K)
      IF(J.LE.K) GOTO 111
```

```
  105 DO 110 I=1,NORDER
      SAVE=ARRAY(I,K)
      ARRAY(I,K)=-ARRAY(I,J)
  110 ARRAY(I,J)=SAVE
  111 I=JK(K)
      IF(I.LE.K) GOTO 130
  113 DO 120 J=1,NORDER
      SAVE=ARRAY(K,J)
      ARRAY(K,J)=-ARRAY(I,J)
  120 ARRAY(I,J)=SAVE
  130 CONTINUE
  140 RETURN
      END
C
      SUBROUTINE PUT(AL,AR,NDIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION AL(NDIM),AR(NDIM)
      DO 5 I=1,NDIM
    5 AL(I)=AR(I)
      RETURN
      END
C
      SUBROUTINE WEIGHT(ND,FND,ND1,P,YDATA,SIGMA,YCAL,MDW)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION P(ND),YDATA(ND),SIGMA(ND1),YCAL(ND)
      SUMR=0.D0
      SUMP=0.D0
      IF(MDW.EQ.0) GOTO 10
    5 GOTO (20,60,50,50,60,70),MDW
   10 DO 15 I=1,ND
      SIGMA(I)=DSQRT(YDATA(I))
   15 P(I)=1.D0/YDATA(I)
      GOTO 30
   20 DO 25 I=1,ND
   25 P(I)=1.D0/SIGMA(I)/SIGMA(I)
   30 DO 35 I=1,ND
      GOTO (34,34,35,35,34,34),MDW
   34 SUMR=P(I)*YDATA(I)*YDATA(I)+SUMR
   35 SUMP=SUMP+P(I)
      SIGMA(ND1)=DSQRT(SUMP/SUMR)
      SUMP=FND/SUMP
      DO 40 I=1,ND
   40 P(I)=P(I)*SUMP
      RETURN
   50 IF(MDW.NE.4) GOTO 52
   51 P(1)=SIGMA(1)
   52 DO 55 I=1,ND
      IF(MDW.NE.4) GOTO 54
   53 SIGMA(I)=P(1)
   54 SUMR=1.D0/SIGMA(I)/SIGMA(I)+SUMR
   55 SIGMA(I)=SIGMA(I)*YDATA(I)
      GOTO 20
   60 DO 65 I=1,ND
      SIGMA(I)=SIGMA(1)
      SUMR=YDATA(I)*YDATA(I)+SUMR
   65 P(I)=1.D0
      SIGMA(ND1)=DSQRT(FND/SUMR)
      RETURN
   70 DO 75 I=1,ND
   75 SIGMA(I)=DEXP(YCAL(I))
      GOTO 20
      END
```

B1.2.3 ALESQFUN.f File

```
      SUBROUTINE CALFUN(NCAL,ILL)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER FINDX*8
```

```
      PARAMETER (MAXND=300,MAXND1=301,MAXNU=10,MAXNX=3)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKARY/FINDX(MAXNX),X(MAXNX,MAXND1),P(MAXND),YDATA(MAXND1),
     +  SIGMA(MAXND1),A(MAXNU),AM(MAXNU),DA(MAXNU),YCAL(MAXND),
     +  YDATAM(MAXND),YCALM(MAXND),PAA(MAXNU,MAXNU),PAL(MAXNU)
      IF(NFUNC.EQ.1) GOTO 10
      ILL=9
      RETURN
   10 DO 11 I=1,NCAL
   11 YCAL(I)=A(1)*X(1,I)*(X(1,I)+A(2))/(X(1,I)*(X(1,I)+A(3))+A(4))
      GOTO 9000
 9000 IF(MDW.NE.5) RETURN
      DO 9900 I=1,NCAL
 9900 YCAL(I)=DLOG(YCAL(I))
      RETURN
      END
```

B1.2.4 ALESQOPT.f File

```
      SUBROUTINE CALAIN(MAXNU,MAXND1,MAXNX,A,YDATA,X,ILL)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(MAXNU),YDATA(MAXND1),X(MAXNX,MAXND1)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      ILL=11
      RETURN
      END
C
      SUBROUTINE OPTION(MAXNU,MAXND,MAXND1,MAXNX,A,DA,X,YDATA,YCAL,ILL)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER *4 MDRUN,LCOMNT,COMNT*80
      DIMENSION A(MAXNU),DA(MAXNU),X(MAXNX,MAXND1),YDATA(MAXND1),
     +  YCAL(MAXND)
      DIMENSION XPLT(300),YPLT(300),ICLASS(300)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      COMMON/BLKOPT/RMS
      IF(NFUNC.EQ.1) GOTO 100
      RETURN
  100 NPT=51
      DO 101 I=1,ND
      XPLT(I)=LOG10(X(1,I))
  101 YPLT(I)=YDATA(I)
      XMIN=XPLT(ND)
      XMAX=XPLT(1)
      YMIN=0.D0
      YMAX=2.D-1
      DX=(XMAX-XMIN)/5.D1
      DO 102 I=1,NPT
      S=REAL(I-1)*DX+XMIN
  102 X(1,I)=10**S
      WRITE(6,601)
      CALL CALFUN(NPT,ILL)
  300 CALL PLOT(XMIN,DX,NPT,YMIN,YMAX,YCAL,MAXND,XPLT,YPLT,ICLASS,ND,
     +  MDAXIS)
      RETURN
  601 FORMAT(1H /1H ,'PLOT WITH LOGARITHMIC X AXIS; VALUES OF X IN THE P
     +LOT ARE ACTUALLY LOG10(X)')
      END
C
      SUBROUTINE PLOT(XMIN,DX,NPT,YMIN,YMAX,YCAL,MAXND,XPLT,YPLT,ICLASS,
     +  ND,MDAXIS)
C       PLOT DATA AND YCAL; TO BE USED IN OPTION WHEN NECESSARY
C       TAKEN FROM J. OIZUMI,"FORTRAN BASED ON JIS" (OHM-SHA, TOKYO,
C       1969) (IN JAPANESE) AND MODIFIED
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
```

```
      CHARACTER *2 LU,LO,LUNDER,LOVER,LBLANK*1,ILINE*1,LINE*1
      DIMENSION ILINE(101),LINE(101),XPLT(ND),YPLT(ND),ICLASS(ND),
     +  YCAL(MAXND)
      KF(S,S0,H)=IDINT(DABS((S-S0)/H)+5.D-1)+1
      DATA LBLANK,LU,LO/' ','UF','OF'/
      WRITE(6,100)
      WRITE(6,101) YMIN,YMAX
      DY=(YMAX-YMIN)/1.D2
      DO 20 I=1,101
   20 ILINE(I)=LBLANK
      GOTO (22,21), MDAXIS
   21 IF(YMAX*YMIN.GT.0.D0) GOTO 24
      I=KF(YMIN,0.D0,DY)
      GOTO 23
   22 I=1
   23 ILINE(I)='I'
   24 DO 10 I=1,ND
   10 ICLASS(I)=KF(XPLT(I),XMIN,DX)
      XV=XMIN
      DO 30 I=1,NPT
      DO 40 K=1,101
   40 LINE(K)=ILINE(K)
      LUNDER=LBLANK
      LOVER=LBLANK
      DO 70 J=1,ND
      IF(ICLASS(J).NE.I) GOTO 70
   61 IF(YPLT(J).LT.YMIN) GOTO 63
      IF(YPLT(J).GT.YMAX) GOTO 64
      K=KF(YPLT(J),YMIN,DY)
      IF(LINE(K).NE.LBLANK) GOTO 62
      LINE(K)='O'
      GOTO 70
   63 LUNDER=LU
      GOTO 70
   64 LOVER=LO
      GOTO 70
   62 LINE(K)='*'
   70 CONTINUE
      YCALI=SNGL(YCAL(I))
      IF(YCALI.LT.YMIN) GOTO 3
      IF(YCALI.GT.YMAX) GOTO 4
      K=KF(YCALI,YMIN,DY)
      IF(LINE(K).NE.LBLANK) GOTO 2
      LINE(K)='+'
      GOTO 50
    3 LUNDER=LU
      GOTO 50
    4 LOVER=LO
      GOTO 50
    2 LINE(K)='X'
   50 WRITE(6,110) XV,LUNDER,LINE,LOVER
   30 XV=XV+DX
      RETURN
  100 FORMAT(1H ,'PLOTTING SYMBOL',6X,'O',2X,'YDATA',6X,'+',2X,'YFITTED'
     +  )
  101 FORMAT(1H ,4X,1HX,3X1PE9.2,91XE9.2/1H ,12X,1HI,99(1H-),1HI)
  110 FORMAT(1H ,1PE9.2,1XA2,101A1,1XA2)
      END
```

B1.2.5 ALESQRD.f File

```
      SUBROUTINE RDDATA(MAXNX,MAXND,MAXND1,YDATA,YDATAM,SIGMA,X,*)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER *4 MDRUN,LCOMNT,MKEND*1,COMNT*80
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,MDP,MDCK,NPRM,PRM(5),ND,ND1,FND,
     +  NFE
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      DIMENSION YDATA(MAXND1),SIGMA(MAXND1),X(MAXNX,MAXND1),
```

```
     +  YDATAM(MAXND)
      IF(MDRUN.EQ.'ISD ') GOTO 40
      IF(MDW.EQ.6) GOTO 30
      ND=0
    5 IF(ND.GT.MAXND) RETURN 1
   10 ND=ND+1
   11 READ(5,501) MDRUN,MKEND,YDATA(ND),SIGMA(ND),(X(J,ND),J=1,NX)
   13 IF(ND.EQ.1) LCOMNT=MDRUN
   15 IF(MKEND.NE.'*') GOTO 5
   16 ND1=ND
      ND=ND-1
      FND=ND
   19 IF(SIGMA(1).NE.0.D0) GOTO 21
   20 SIGMA(1)=1.D0
   21 IF(MDW.NE.5) RETURN
      DO 25 I=1,ND
      YDATAM(I)=YDATA(I)
   25 YDATA(I)=DLOG(YDATA(I))
      RETURN
   30 DO 35 I=1,ND
   35 YDATA(I)=YDATAM(I)
      RETURN
   40 READ(5,501) MDRUN
  501 FORMAT(A4,1X,A1,4X,7F10.4)
      END
```

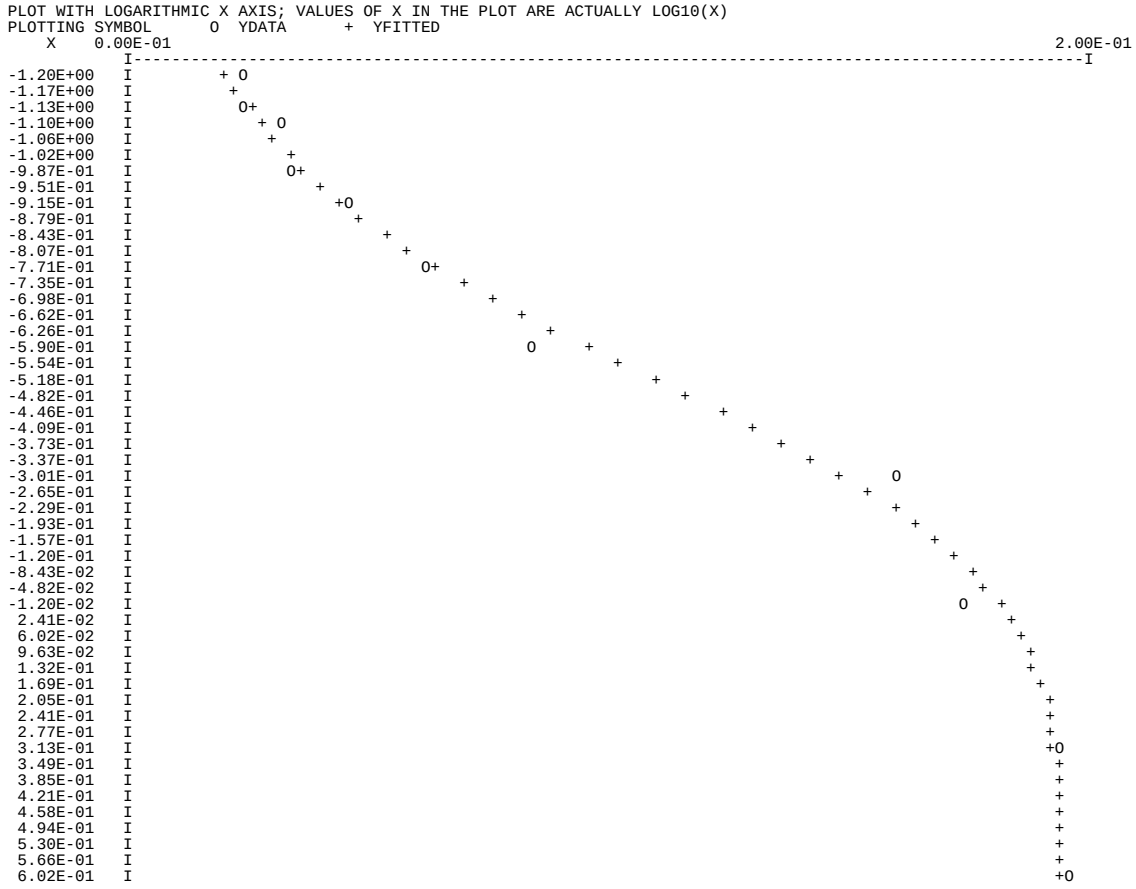
B1.3 Test Input and Output Files for ALESQ

B1.3.1 Test Input File for ALESQ

```
1,1,4,1,2,1,0,0,100,0
TEST PROBLEM: ENZIME FUNCTION
X
          .1957               4.00
          .1947               2.00
          .1735               1.00
          .1600               .500
          .0844               .250
          .0627               .167
          .0456               .125
          .0342               .100
          .0323               .0833
          .0235               .0714
          .0246               .0625
ISTP *
0.25,0.4,0.4,0.4
```

B1.3.2 Test Output File for ALESQ

```
LEAST-SQUARES FIT PROGRAM ALESQ    TEST PROBLEM: ENZIME FUNCTION

  MODE OF INPUT FOR STARTING VALUES      (MDS) =   1
  FUNCTION NUMBER                      (NFUNC) =   1
  NUMBER OF UNKNOWNS                     (NU) =   4
  NUMBER OF INDEPENDENT VARIABLES        (NX) =   1
  MODE OF WEIGHTING                     (MDW) =   2
  MODE OF PRINT                         (MDP) =   1
  MODE OF FIT/CHECK                    (MDCK) =   0
  LIMIT TO THE NUMBER OF ITERATIONS   (NITLIM) = 100
  NUMBER OF PARAMETERS                  (NPRM) =   0
  NUMBER OF DATA                          (ND) =  11

NO. LAMBDA  ERROR     VALUES OF UNKNOWNS A(I)
                          1          2          3          4
  0 0.0E-01 2.82E-02  2.500E-01  4.000E-01  4.000E-01  4.000E-01
  1 1.0E-02 1.95E-02  1.978E-01  3.734E-01  3.823E-01  1.225E-01
  2 3.2E-03 7.64E-03  1.924E-01  3.402E-01  2.375E-01  1.793E-01
  3 1.0E-03 6.78E-03  1.932E-01  1.607E-01  1.020E-01  1.284E-01
  4 1.0E-04 6.63E-03  1.923E-01  1.979E-01  1.217E-01  1.399E-01
  5 1.0E-06 6.63E-03  1.930E-01  1.859E-01  1.219E-01  1.336E-01
  6 1.0E-08 6.63E-03  1.927E-01  1.945E-01  1.236E-01  1.376E-01
  7 0.0E-01 6.63E-03  1.929E-01  1.892E-01  1.227E-01  1.351E-01
  8 0.0E-01 6.63E-03  1.927E-01  1.926E-01  1.233E-01  1.367E-01
  9 0.0E-01 6.63E-03  1.928E-01  1.905E-01  1.229E-01  1.357E-01
 10 0.0E-01 6.63E-03  1.928E-01  1.918E-01  1.232E-01  1.363E-01
 11 0.0E-01 6.63E-03  1.928E-01  1.910E-01  1.230E-01  1.359E-01
 12 0.0E-01 6.63E-03  1.928E-01  1.915E-01  1.231E-01  1.362E-01
 13 0.0E-01 6.63E-03  1.928E-01  1.912E-01  1.230E-01  1.360E-01
 14 0.0E-01 6.63E-03  1.928E-01  1.914E-01  1.231E-01  1.361E-01
 15 0.0E-01 6.63E-03  1.928E-01  1.912E-01  1.230E-01  1.360E-01
 16 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01
 17 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01
 18 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01
 19 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01
 20 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01
 21 0.0E-01 6.63E-03  1.928E-01  1.913E-01  1.231E-01  1.361E-01

  NUMBER OF ITERATIONS                          =  21
  NUMBER OF FUNCTION EVALUATIONS                = 112

  RMS ABSOLUTE ERROR OF FIT                     = 6.62792E-03
  RMS RELATIVE ERROR OF FIT                     = 5.70608E-02
  RMS RELATIVE RESIDUAL                         = 4.55187E-02
  MAXIMUM IN MAGNITUDE OF RESIDUAL              = 1.111E-02 AT DATA NO.   4

  RELATIVE RMS DEVIATION OF DATA FROM EQUATION = 7.88E-02

  A( 1) =  1.928070974E-01 +/-  1.1E-02
  A( 2) =  1.912787542E-01 +/-  2.0E-01
  A( 3) =  1.230558738E-01 +/-  8.1E-02
  A( 4) =  1.360606703E-01 +/-  9.0E-02

 NO.   WEIGHT    X          SIGMA      Y DATA     Y FITTED   RESIDUAL
                                                             (ABS.)
   1  1.000E+00  4.000E+00  1.000E+00  1.957E-01  1.944E-01 -1.306E-03
   2  1.000E+00  2.000E+00  1.000E+00  1.947E-01  1.928E-01 -1.876E-03
   3  1.000E+00  1.000E+00  1.000E+00  1.735E-01  1.824E-01  8.919E-03
   4  1.000E+00  5.000E-01  1.000E+00  1.600E-01  1.489E-01 -1.111E-02
   5  1.000E+00  2.500E-01  1.000E+00  8.440E-02  9.275E-02  8.352E-03
   6  1.000E+00  1.670E-01  1.000E+00  6.270E-02  6.253E-02 -1.735E-04
   7  1.000E+00  1.250E-01  1.000E+00  4.560E-02  4.563E-02  2.582E-05
   8  1.000E+00  1.000E-01  1.000E+00  3.420E-02  3.546E-02  1.262E-03
   9  1.000E+00  8.330E-02  1.000E+00  3.230E-02  2.878E-02 -3.524E-03
  10  1.000E+00  7.140E-02  1.000E+00  2.350E-02  2.412E-02  6.165E-04
  11  1.000E+00  6.250E-02  1.000E+00  2.460E-02  2.071E-02 -3.889E-03
```


PLOT WITH LOGARITHMIC X AXIS; VALUES OF X IN THE PLOT ARE ACTUALLY LOG10(X)
PLOTTING SYMBOL O YDATA + YFITTED
X 0.00E-01 2.00E-01
-1.20E+00
-1.17E+00
-1.13E+00
-1.10E+00
-1.06E+00
-1.02E+00
-9.87E-01
-9.51E-01
-9.15E-01
-8.79E-01
-8.43E-01
-8.07E-01
-7.71E-01
-7.35E-01
-6.98E-01
-6.62E-01
-6.26E-01
-5.90E-01
-5.54E-01
-5.18E-01
-4.82E-01
-4.46E-01
-4.09E-01
-3.73E-01
-3.37E-01
-3.01E-01
-2.65E-01
-2.29E-01
-1.93E-01
-1.57E-01
-1.20E-01
-8.43E-02
-4.82E-02
-1.20E-02
2.41E-02
6.02E-02
9.63E-02
1.32E-01
1.69E-01
2.05E-01
2.41E-01
2.77E-01
3.13E-01
3.49E-01
3.85E-01
4.21E-01
4.58E-01
4.94E-01
5.30E-01
5.66E-01
6.02E-01

FORTRAN Code and Test Input and Output Files for TSOLVE

B2.1 COMMENT Statements for TSOLVE

```
C  DESCRIPTION OF "TSOLVE"
C
C
C   PURPOSE: TO SOLVE THE PROBLEM OF NONLINEAR BEST APPROXIMATION
C
C   COMPUTER USED FOR TEST: POWER MACINTOSH 8500/150
C
C   OPERATING SYSTEM: MAC OS 7.5.5
C
C   PROGRAMMING LANGUAGE: FORTRAN 77
C
C   FORTRAN COMPILER USED: ABSOFT FORTRAN 77
C
C   PERIPHERALS USED: NONE
C
C   FORMAT FOR INPUT FILE
C
C     MDS,NFUNC,NU,NX,MDW,MDP,MDINS,MDCK,NPRM,NITLIM,PRM    FREE FIELD
C     COMNT                                                 A80
C     DUMMY LINE (SIGNIFICANT IN THE CODE "ALESQ")
C     YDATA(I),X(I)                                         2(10X10.4)
C     MDRUN,MKEND                                           A4,1X,A1
C     (A(I),I=1,NU)                                         FREE FIELD;
C
C   MEANING OF INPUT VARIABLES
C
C     MDS      PARAMETER TO SELECT THE MODE OF INPUT FOR STARTING VALUES
C              0  LINEAR BEST APPROXIMATION: STARTING VALUES NOT USED
C              1  NONLINEAR BEST APPROXIMATION: STARTING VALUES GIVEN BY
C                 THE INPUT FILE
C              2  NONLINEAR BEST APPROXIMATION: STARTING VALUES TAKEN
C                 FROM THE PREVIOUS RESULT
C     NFUNC    NUMBER ATTACHED TO THE APPROXIMATING FUNCTION
C     NU       NUMBER OF UNKNOWNS
C     NX       DUMMY PARAMETER (SIGNIFICANT IN THE CODE "ALESQ")
C     MDW      PARAMETER TO SELECT MODE OF APPROXIMATION
C              1 OR 2 MINIMIZE THE MAXIMUM OF ABSOLUTE ERRORS
C              3 OR 4 MINIMIZE THE MAXIMUM OF RELATIVE ERRORS
C     MDP      PARAMETER TO SELECT MODE OF PRINT
C              0  PRINT FINAL RESULT ONLY
C              1  PRINT INTERMEDIATE AND FINAL RESULTS
C     MDINS    PARAMETER TO SELECT MODE OF FINDING THE MAXIMUM ERROR
C              1  ONLY DATA POINTS ARE CONSIDERED IN FINDING THE MAXIMUM
C                 ERROR
C              2  QUADRATIC INTERPOLATION IS USED IN FINDING THE MAXIMUM
C                 ERROR
C     MDCK     PARAMETER TO SELECT EITHER NORMAL OR CHECK MODE
C              0  NORMAL MODE (SOLVE BEST-APPROXIMATION PROBLEM)
C              1  CHECK MODE (CALCULATE THE ERROR CURVE FOR GIVEN VALUES
C                 OF "A" )
C     NITLIM   DUMMY PARAMETER (SIGNIFICANT IN THE CODE "ALESQ")
C     NPRM     NUMBER OF ELEMENTS USED IN "PRM"
C     PRM      ARRAY OF PARAMETERS IN THE APPROXIMATING FUNCTION WHOSE
C              VALUES ARE TO BE VARIED FROM ONE RUN TO THE NEXT
C     COMNT    COMMENT ON THE PROBLEM
C     YDATA    ARRAY OF DATA POINTS FOR DEPENDENT VARIABLE
C     X        ARRAY OF DATA POINTS FOR INDEPENDENT VARIABLE
C     MDRUN    PARAMETER TO SELECT THE MODE OF THE NEXT RUN
C              "ISDT"     THE NEXT RUN IS TO BE MADE WITH THE SAME DATA
C                         ("SDT" MEANING THE SAME DATA).
C                         DATA FOR THE NEXT RUN:
```

```
C                          MDS, ...
C                          MDRUN,MKEND
C                          (A(I),I=1,NU)
C             "ISDF"     THE NEXT RUN IS TO BE MADE WITH THE SAME FUNC-
C                        TION AND THE SAME DATA ("SDF" MEANING THE SAME
C                        FUNCTION).
C                        DATA FOR THE NEXT RUN:
C                          MDS. ...
C                          MDRUN,MKEND
C                          (A(I).I=1,NU)
C             "ISFN"     THE NEXT RUN IS TO BE MADE WITH THE SAME FUNC-
C                        TION ("SDF" MEANING THE SAME FUNCTION & DATA.
C                        DATA FOR THE NEXT RUN:
C                          YDATA(I),X(I)
C                          MDRUN,MKEND
C                          (A(I).I=1,NU)
C             "ISTP"     STOP AFTER THE PRESENT RUN.
C             OTHERWISE  THE NEXT RUN IS TO BE MADE WITH A DIFFERENT
C                        FUNCTION AND A DIFFERENT DATA SET.
C     MKEND    VARIABLE TO INDICATE THE END OF "YDATA-X" SET;
C              "*" SHOULD BE PUT.
C     A        ARRAY OF UNKNOWNS; PUT STARTING VALUES; NECESSARY ONLY
C              FOR MDS=1 OR MDCK=1
C
C   MEANING OF VARIABLES IN PARAMETER STATEMENTS
C
C     MAXND    MAXIMUM NUMBER OF DATA ALLOWED
C     MAXND1   MAXND+1
C     MAXNDN   2*MAXND+1
C     MAXNU    MAXIMUM NUMBER OF UNKNOWNS ALLOWED
C     MAXNU3   MAXNU+3
C
C
C
C  DESCRIPTION OF SUBPROGRAMS
C
C    SUBROUTINE OPTION
C      PERFORMS OPTIONAL CALCULATIONS OR PLOTTING USING THE RESULT OF
C      BEST APPROXIMATION. NECESSARY STATEMENTS SHOULD BE PROVIDED BY
C      THE USER.
C
C    SUBROUTINE READ1
C      READS "YDATA" AND "X."
C
C    SUBROUTINE SELMAX
C      SELECTS MAXIMAL POINTS OF THE ERROR CURVE.
C
C    SUBROUTINE SIMPDT
C      SETS DATA FOR SUBROUTINE "SIMPLX."
C
C    SUBROUTINE SIMPLX
C      SOLVES THE SIMPLEX PROBLEM.
C
C    SUBROUTINE SOLVE
C      FINDS SOLUTION TO THE PROBLEM.
C
C    SUBROUTINE SRMXPT
C      FINDS AN ELEMENT OF MAXIMUM MAGNITUDE WITHIN A GIVEN RANGE IN AN
C      ARRAY.
C
C    SUBROUTINE START
C      READS INTUT VARIABLES AND SETS INITIAL VALUES.
C
C    SUBROUTINE WRITE1
C      WRITES INPUT VARIABLES, INTERMEDIATE AND FINAL RESULTS.
C
C    FUNCTION   FDERIV
C      EVALUATES DERIVATIVES OF THE APPROXIMATING FUNCTION FROM CENTRAL
C      DIFFERENCES.
```

```
C
C    SUBROUTINE FMAX
C      FINDS AN ELEMENT OF MAXIMUM MAGNITUDE IN AN ARRAY.
C
C    FUNCTION   FUNC
C      EVALUATES THE APPROXIMATING FUNCTION. THE USER SHOULD PROVIDE THE
C      FUNCTION TO BE USED IN THIS SUBPROGRAM.
C
C    SUBROUTINE PUT
C      PUTS VALUES OF AN ARRAY INTO ANOTHER ARRAY.
C
C    SUBROUTINE PUT1
C      PUTS VALUES OF AN INTEGER ARRAY INTO ANOTHER INTEGER ARRAY.
C
C
C
C  MEANING OF PARAMETERS IN THE SUBPROGRAMS (SEE ALSO INPUT VARIABLES
C     AND VARIABLES IN PARAMETER STATEMENTS)
C
C
C   PARAMETERS WITH COMMON MEANING
C
C      A       ARRAY OF UNKNOWNS
C      DYP     PARTIAL ARRAY OF DEVIATIONS OF APPROXIMATION FROM DATA
C
C
C   PARAMETERS IN EACH SUBPROGRAM
C
C    SUBROUTINE SELMAX
C      MDSEL   PARAMETER TO SELECT THE MODE OF THE SUBROUTINE
C              1  B, E AND F
C              2  B, C, D, E AND F
C              3  B, C, E AND F
C              4  A, B, C, E AND F
C              5  D,
C              WHERE THE SYMBOLS FROM A TO F MEAN:
C              A  RESTORE DATA
C              B  FIND MAXIMAL POINTS OF THE ERROR CURVE
C              C  INTERPOLATE THE ERROR CURVE
C              D  EXCHANGE DATA
C              E  SET "XMAX" AND "DYMAX"
C              F  FIND THE MAXIMUM ERROR
C      NDATA   DIMENSION OF "DYP" (ACTUAL ARGUMENT IS PUT EQUAL TO THE
C              MAXIMUM POSSIBLE VALUE "ND.")
C
C    SUBROUTINE SRMXPT
C      NI      DIMENSION OF "DYP" (ACTUAL ARGUMENT IS PUT EQUAL TO THE
C              NUMBER OF CONSECUTIVE DATA POINTS HAVING ERRORS OF THE
C              SAME SIGN.)
C      IB      THE NUMBER SUCH THAT (IB+1) IS EQUAL TO THE INDEX OF THE
C              ELEMENT OF THE ARRAY "DY" THAT CORRESPONDS TO THE FIRST
C              ELEMENT OF "DYP."
C
C    FUNCTION   FDERIV
C      I       INDEX OF THE UNKNOWN IN THE ARRAY "A" WITH RESPECT TO
C              WHICH DIFFERENTIATION IS MADE
C      J       INDEX OF THE DATA POINT AT WHICH DIFFERENTIATION IS MADE
C      H       HALF STEP TO CALCULATE DIFFERENCES OF THE FUNCTION
C
C    SUBROUTINE FMAX
C      X       ARRAY (INPUT)
C      N       DIMENSION OF "X" (INPUT)
C      XMAX    MAXIMUM IN MAGNITUDE OF VALUES IN "X" (OUTPUT)
C      NMAX    INDEX OF THE ELEMENT IN "X" WHOSE VALUE IS "XMAX"
C              (OUTPUT)
C
C    FUNCTION   FUNC
C      I       INDEX OF THE DATA POINT FOR WHICH FUNCTION IS TO BE
C              EVALUATED
```

```
C
C    SUBROUTINE PUT
C      AL      OUTPUT ARRAY
C      AR      INPUT ARRAY
C      NDIM    DIMENSION OF THE ARRAYS
C
C    SUBROUTINE PUT1
C      IAL     OUTPUT ARRAY
C      IAR     INPUT ARRAY
C      NDIM    DIMENSION OF THE ARRAYS
```

B2.2 FORTRAN Code for TSOLVE

B2.2.1 TSOLMN.f File

```
      PROGRAM TSOLVE
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      CHARACTER *4 MDRUN,LCOMNT,COMNT*80
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKSIM/ABAR(MAXNU3,MAXNDN),DELTA(MAXNDN),X0(MAXNU3),
     +   XK(MAXNU3),U(MAXNU3,MAXNU3),INDEX(MAXNU3),N,M,M1,M2,NCYC
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      DIMENSION DYP(MAXND)
      EQUIVALENCE (DYP,ABAR)
      OPEN(5,FILE='',STATUS='OLD')
      OPEN(6,FILE='')
      MDRUN='ISTP'
    5 CALL START
      MDSEL=MDINS
      GO TO (15,10),MDINS
   10 CALL PUT(XM,X,ND)
      CALL PUT(YM,YDATA,ND)
      IF(MDS.EQ.0.AND.MDINS.EQ.1) GO TO 19
      MDSEL=MDSEL+ICONV
   15 CALL SELMAX(MDSEL,DYP,ND)
      CALL PUT(AM,A,NU)
      DELMXM=DELMAX
   19 ICONV=ICONV+1
   20 CALL WRITE1(*25)
      CALL SOLVE
      GO TO 20
   25 CALL OPTION
      IF(MDRUN.NE.'ISTP') GO TO 5
      CLOSE(5)
      CLOSE(6)
      STOP
      END
C
      SUBROUTINE SELMAX(MDSEL,DYP,NDATA)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      DIMENSION VEC(3),VEC1(3),DYP(NDATA)
      DIMENSION DYP(NDATA)
      IF(MDSEL-4) 7,7,61
```

```
 5 CALL PUT(X,XM,ND)
   CALL PUT(YDATA,YM,ND)
 7 DO 12 I=1,ND
   YFIT(I)=FUNC(MAXNU,MAXND1,A,X,I)
   DY(I)=YFIT(I)-YDATA(I)
   IF(MDW.LE.2) GOTO 12
11 DY(I)=DY(I)/YDATA(I)
12 NSELEC(I)=0
   NCALL=NCALL+1
   IFIN=0
   DO 20 I=2,ND
   JJ=I-1
   IF(DY(JJ)*DY(I).GT.0.D0) GOTO 20
17 II=IFIN
   IFIN=I-1
   J=IFIN-II
   CALL SRMXPT(DYP,J,II)
20 CONTINUE
   J=ND-IFIN
   CALL SRMXPT(DYP,J,IFIN)
   II=0
   DO 27 I=1,ND
   IF(NSELEC(I).LE.0) GOTO 27
25 II=II+1
   NSELEC(II)=NSELEC(I)
27 CONTINUE
   NMAX=II
   IF(MDSEL.LT.2) GOTO 62
30 DO 45 I=1,NMAX
   II=NSELEC(I)
   IF(II-1) 37,34,33
33 IF(II.NE.ND) GOTO 37
34 XMAX(I)=X(II)
   DYMAX(I)=DY(II)
   GO TO 45
37 DO 40 J=1,3
   JJ=II+J-2
   VEC(J)=DY(JJ)
40 VEC1(J)=X(JJ)
   WS=VEC1(2)-VEC1(1)
   WS=(VEC(2)-VEC(1))/WS
   WS1=VEC1(3)-VEC1(1)
   WS2=VEC1(3)-VEC1(2)
   WS1=((VEC(3)-VEC(1))/WS1-WS)/WS2
   WS2=(-WS/WS1+VEC1(1)+VEC1(2))*.5D0
   XMAX(I)=WS2
   DYMAX(I)=WS2-VEC1(1)
   DYMAX(I)=((WS2-VEC1(2))*WS1+WS)*DYMAX(I)+VEC(1)
45 CONTINUE
   GO TO (60,51,60,60,51),MDSEL
51 DO 59 I=1,NMAX
   II=NSELEC(I)
   X(II)=XMAX(I)
   DY(II)=DYMAX(I)
   YFIT(II)=FUNC(MAXNU,MAXND1,A,X,II)
   IF(MDW.GT.2) GOTO 57
55 YDATA(II)=YFIT(II)-DY(II)
   GO TO 59
57 YDATA(II)=DY(II)+1.D0
   YDATA(II)=YFIT(II)/YDATA(II)
59 CONTINUE
   NCALL=NCALL+1
   IF(MDSEL.EQ.5) GOTO 61
60 CALL FMAX(DY,ND,DELMAX,JMAX)
61 RETURN
62 DO 65 I=1,NMAX
   II=NSELEC(I)
   XMAX(I)=X(II)
65 DYMAX(I)=DY(II)
```

```
      GO TO 60
      END
C
      SUBROUTINE SIMPDT
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKSIM/ABAR(MAXNU3,MAXNDN),DELTA(MAXNDN),X0(MAXNU3),
     +   XK(MAXNU3),U(MAXNU3,MAXNU3),INDEX(MAXNU3),N,M,M1,M2,NCYC
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      N=ND+ND+1
      M=NU+1
      M1=M+1
      M2=M1+1
      X0(M)=1.D0
      X0(M1)=0.D0
      X0(M2)=-1.D0
      ABAR(M,N)=1.D0
      ABAR(M1,N)=0.D0
      ABAR(M2,N)=-1.D0
      DO 30 I=1,NU
      X0(I)=0.D0
   30 ABAR(I,N)=0.D0
      DO 45 J=1,ND
      J1=ND+J
      ABAR(M1,J)=DY(J)
      ABAR(M1,J1)=-DY(J)
      ABAR(M,J)=1.D0
      ABAR(M,J1)=1.D0
      DO 40 I=1,NU
      ABAR(I,J1)=FDERIV(MAXNU,MAXND1,A,X,I,J,1.D-7)
      IF(MDW.LE.2) GOTO 40
   39 ABAR(I,J1)=ABAR(I,J1)/YDATA(J)
   40 ABAR(I,J)=-ABAR(I,J1)
      NCALL=NCALL+NU+NU
      ABAR(M2,J)=0.D0
      ABAR(M2,J1)=0.D0
      DO 45 I=1,M
      ABAR(M2,J)=ABAR(M2,J)-ABAR(I,J)
   45 ABAR(M2,J1)=ABAR(M2,J1)-ABAR(I,J1)
      RETURN
      END
C
      SUBROUTINE SIMPLX(*)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      COMMON/BLKSIM/ABAR(MAXNU3,MAXNDN),DELTA(MAXNDN),X0(MAXNU3),
     +   XK(MAXNU3),U(MAXNU3,MAXNU3),INDEX(MAXNU3),N,M,M1,M2,NCYC
      DATA TOL,NTOL/5.D-9,100/
      NCYC=0
      CALL SIMPDT
      DO 10 I=1,M2
      INDEX(I)=N+I
      DO 10 J=1,M2
      IF(I.NE.J) GOTO 9
    7 U(I,J)=1.D0
      GO TO 10
    9 U(I,J)=0.D0
   10 CONTINUE
      MP=1
      MM=M2
   16 IF(X0(M2)+TOL) 19,17,17
   17 MP=2
```

```
      MM=M1
   19 NCYC=NCYC+1
      IF(NCYC.GT.NTOL) GOTO 71
   21 DO 24 J=1,N
      DELTA(J)=0.D0
      DO 24 K=1,M2
   24 DELTA(J)=U(MM,K)*ABAR(K,J)+DELTA(J)
      K=1
      DO 29 J=2,N
      IF(DELTA(J).GE.DELTA(K)) GOTO 29
   28 K=J
   29 CONTINUE
      IF(DELTA(K)+TOL) 33,31,31
   31 GO TO (65,32),MP
   32 RETURN
   33 DO 35 I=1,M2
      XK(I)=0.D0
      DO 35 J=1,M2
   35 XK(I)=U(I,J)*ABAR(J,K)+XK(I)
      DO 38 L=1,M
      IF(XK(L).GT.TOL) GOTO 40
   38 CONTINUE
      GO TO (74,68),MP
   40 TH0=X0(L)/XK(L)
      IF(TH0.EQ.0.D0) GOTO 51
   42 IF(L.EQ.M) GOTO 51
   43 J=L+1
      DO 50 I=J,M
      IF(XK(I).LT.TOL) GOTO 50
   46 TH=X0(I)/XK(I)
      IF(TH.GE.TH0) GOTO 50
   48 L=I
      TH0=TH
   50 CONTINUE
   51 INDEX(L)=K
      DO 58 I=1,M2
      IF(I.NE.L) GOTO 57
   55 X0(I)=TH0
      GO TO 58
   57 X0(I)=-XK(I)*TH0+X0(I)
   58 CONTINUE
      DO 64 J=1,M2
      U(L,J)=U(L,J)/XK(L)
      DO 64 I=1,M2
      IF(I.EQ.L) GOTO 64
   63 U(I,J)=-U(L,J)*XK(I)+U(I,J)
   64 CONTINUE
      GO TO (16,19),MP
   65 WRITE(6,601)
  601 FORMAT(25H0**NO FEASIBLE SOLUTION**)
      RETURN 1
   68 WRITE(6,602)
  602 FORMAT(23H0**SOLUTION UNBOUNDED**)
      RETURN 1
   71 WRITE(6,603)
  603 FORMAT(19H0**NCYC TOO LARGE**)
      RETURN 1
   74 WRITE(6,604)
  604 FORMAT(13H0**SINGULAR**)
      RETURN 1
      END
C
      SUBROUTINE SOLVE
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
```

```
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKSIM/ABAR(MAXNU3,MAXNDN),DELTA(MAXNDN),X0(MAXNU3),
     +   XK(MAXNU3),U(MAXNU3,MAXNU3),INDEX(MAXNU3),N,M,M1,M2,NCYC
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      DIMENSION YFM(MAXND),DYM(MAXND),NSELM(120),AWK(MAXNU),XMAXM(120),
     +   DYMAXM(120),DYP(MAXND)
      EQUIVALENCE (YFM,DELTA),(DYM,DELTA(MAXND1)),(AWK,X0),(DYP,ABAR)
      DATA CRIT,TOL/5.D-9,5.D-3/
      ITR=ITR+1
      CALL SIMPLX(*98)
      INITR=INITR+NCYC
      GAMMA=1.D0
      DO 23 I=1,NU
   23 A(I)=AM(I)-U(M1,I)
      DELMXS=X0(M1)
      IF(MDS.EQ.0.AND.MDINS.EQ.1) GO TO 69
      MDSEL=MDINS*3-2
      CALL SELMAX(MDSEL,DYP,ND)
      MDSEL=MDSEL-MDINS+1
      DELGAM=1.D0
      DO 50 NORDER=1,7
      DELGAM=1.D-1*DELGAM
      DO 50 ITRG=1,9
      DELMX=DELMAX
      NMAXM=NMAX
      CALL PUT(YFM,YFIT,ND)
      CALL PUT(DYM,DY,ND)
      CALL PUTI(NSELM,NSELEC,NMAX)
      CALL PUT(XMAXM,XMAX,NMAX)
      CALL PUT(DYMAXM,DYMAX,NMAX)
      CALL PUT(AWK,A,NU)
      GAMMAM=GAMMA
      GAMMA=GAMMA-DELGAM
      DO 45 I=1,NU
   45 A(I)=-U(M1,I)*GAMMA+AM(I)
      CALL SELMAX(MDSEL,DYP,ND)
      IF(DELMX.GT.DELMAX) GOTO 50
   48 WS=DELMXM-DELMX
      IF(WS.GE.0.D0) GOTO 54
   50 CONTINUE
      WRITE(6,600)
  600 FORMAT(24H0**SEARCH DISCONTINUED**)
      ICONV=2
   54 DELMAX=DELMX
      NMAX=NMAXM
      CALL PUT(YFIT,YFM,ND)
      CALL PUT(DY,DYM,ND)
      CALL PUTI(NSELEC,NSELM,NMAX)
      CALL PUT(XMAX,XMAXM,NMAX)
      CALL PUT(DYMAX,DYMAXM,NMAX)
      CALL PUT(A,AWK,NU)
      GAMMA=GAMMAM
      IF(ICONV.EQ.2) GOTO 71
   65 WS=DABS(WS)
      WS1=DABS(DELMXS-DELMAX)
      IF(WS.GT.CRIT) GOTO 92
   68 IF(WS1-CRIT) 70,70,92
   69 CALL SELMAX(1,DYP,ND)
   70 ICONV=2
   71 DEL=(1.D0+TOL)*DELMXM
      IF(NU.GT.1) GOTO 75
   73 SIGINV=1.D0
      GO TO 76
   75 SIGINV=DSQRT(3.D0/DBLE(NU))
   76 DO 91 I=1,NU
      DO 85 J=1,ND
      WS=FDERIV(MAXNU,MAXND1,A,X,I,J,1.D-7)
      IF(WS) 80,83,84
```

```
   80 WS=-WS
      SS=1.D0
      GO TO 85
   83 WS=1.D-70
   84 SS=-1.D0
   85 YM(J)=(DY(J)*SS+DEL)/WS
      WS=YM(1)
      DO 90 J=2,ND
      IF(YM(J)-WS) 89,90,90
   89 WS=YM(J)
   90 CONTINUE
   91 DA(I)=WS*SIGINV
   92 DELMXM=DELMAX
      CALL PUT(AM,A,NU)
      IF(ICONV-2) 95,97,95
   95 IF(MDINS-1) 96,97,96
   96 CALL SELMAX(5,DYP,ND)
   97 RETURN
   98 ICONV=2
      RETURN
      END
C
      SUBROUTINE SRMXPT(DYP,NI,IB)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      DIMENSION DYP(NI)
      DO 5 J=1,NI
      JJ=IB+J
    5 DYP(J)=DY(JJ)
      CALL FMAX(DYP,NI,WS,JMAX)
      IF(WS) 30,30,20
   20 IS=IB+JMAX
      NSELEC(IS)=IS
   30 RETURN
      END
C
      SUBROUTINE START
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      CHARACTER *4 MDRUN,LCOMNT,COMNT*80,COMDUM*80
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      IF(MDRUN.EQ.'ISFN') GO TO 10
      READ(5,*) MDS,NFUNC,NU,NX,MDW,MDP,MDINS,MDCK,DUM,NPRM,(PRM(I),I=1,
     +NPRM)
      MDZ=(MDW+1)/2
      IF(MDRUN.EQ.'ISDT'.OR.MDRUN.EQ.'ISDF') GO TO 10
      READ(5,501) COMNT
      READ(5,501) COMDUM
   10 WRITE(6,600) COMNT
      WRITE(6,601) MDS,NFUNC,NU,MDW,MDP,MDINS,MDCK,NPRM
      IF(NPRM.GT.0) WRITE(6,602) (PRM(I),I=1,NPRM)
      CALL READ1
      WRITE(6,603) ND
      GO TO (12,15,16),MDS+1
   12 DO 13 I=1,NU
   13 A(I)=0.D0
      GO TO 16
   15 READ(5,*) (A(I),I=1,NU)
   16 ITR=0
```

```
      INITR=0
      NCALL=0
      ICONV=MDCK
      RETURN
  501 FORMAT(A80)
  600 FORMAT(1H /' BEST APPROXIMATION PROGRAM TSOLVE',4XA80/)
  601 FORMAT('   MODE OF INPUT FOR STARTING VALUES (MDS) =',I4/
     +'   FUNCTION NUMBER                (NFUNC) =',I4/
     +'   NUMBER OF UNKNOWNS                (NU) =',I4/
     +'   MODE OF WEIGHTING                (MDW) =',I4/
     +'   MODE OF PRINT                    (MDP) =',I4/
     +'   MODE OF FINDING MAXIMUM ERROR   (MDINS) =',I4/
     +'   MODE OF FIT/CHECK                (MDCK) =',I4/
     +'   NUMBER OF PARAMETERS             (NPRM) =',I4)
  602 FORMAT(1H ,20X,5HPRM =,1P5E13.4)
  603 FORMAT('   NUMBER OF DATA                    (ND) =',I4/)
      END
C
      SUBROUTINE WRITE1(*)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER OFF
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      CHARACTER *4 LWGT
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      DIMENSION LWGT(2)
      DATA LWGT,OFF/'ABS.','REL.',2/
      IF(ITR) 5,5,1
    1 GO TO (2,6), ICONV
    2 IF(MDP) 6,90,6
    5 NUM=MIN0(NU,10)
      WRITE(6,600) (I,I=1,NUM)
      IF(MDS.EQ.0) RETURN
    6 IF(ITR.GE.0) GOTO 7
      WRITE(6,601) ITR,DELMXM,(AM(I),I=1,NU)
      GOTO 15
    7 WRITE(6,602) ITR,DELMXS,GAMMA,DELMXM,(AM(I),I=1,NU)
   15 GOTO (90,20), ICONV
   20 WRITE(6,604) LWGT(MDZ)
      WRITE(6,605) (I,XMAX(I),DYMAX(I),I=1,NMAX)
      WRITE(6,606) ITR,INITR,NCALL
      WRITE(6,607) (I,AM(I),DA(I),I=1,NU)
      WRITE(6,608) LWGT(MDZ)
      DO 30 I=1,ND
      WRITE(6,609) I,X(I),YDATA(I),YFIT(I),DY(I)
   30 CONTINUE
   34 RETURN 1
   90 RETURN
  600 FORMAT(1H ,' NO. DELTAS   GAMMA   DELTAMAX VALUES OF UNKNOWNS A(I)
     +'/1H ,30X,10(4X,I2,4X))
  601 FORMAT(1H ,I3,18X1PE9.2,10E10.3/1H ,30X,10E10.3)
  602 FORMAT(1H ,I3,1PE10.2,E8.1,E9.2,10E10.3/1H ,30X,10E10.3)
  603 FORMAT(1H ,103X,1PE10.2,E8.1,E9.2)
  604 FORMAT(1H /25H  LIST OF EXTREMAL POINTS/1H ,11X,1HX,8X,A4,5H DEV.)
  605 FORMAT(1H ,I3,1P2E13.3)
  606 FORMAT(1H /21H  NO. OF ITERATIONS =,I3,4X,28HNO. OF INTERNAL ITERA
     +TIONS =,I6/1H ,30H NO. OF FUNCTION EVALUATIONS =,I6//39H  SOLUTION
     +S AND TOLERANCES FOR ROUNDOFF)
  607 FORMAT(1H ,14X,1HA,13X,2HDA/(1H ,I3,1PE20.10,E10.1))
  608 FORMAT(1H /1H ,1X'NO.',5X'X',10X'Y DATA',9X'Y FITTED',5X,A4,' DEV.
     +')
  609 FORMAT(1H ,2(I3,1PE11.3,2E16.8,E11.3,2X))
      END
C
```

```
      DOUBLE PRECISION FUNCTION FDERIV(MAXNU,MAXND1,A,X,I,J,H)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(MAXNU),X(MAXND1)
      AI=A(I)
      DELTA=H
      IF(AI) 5,10,5
    5 DELTA=AI*DELTA
   10 A(I)=AI+DELTA
      FPLUS=FUNC(MAXNU,MAXND1,A,X,J)
      A(I)=AI-DELTA
      FMINUS=FUNC(MAXNU,MAXND1,A,X,J)
      FDERIV=(FPLUS-FMINUS)*.5D0/DELTA
      A(I)=AI
      RETURN
      END
C
      SUBROUTINE FMAX(X,N,XMAX,NMAX)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION X(N)
      XMAX=DABS(X(1))
      NMAX=1
      IF(1-N) 5,6,6
    6 RETURN
    5 DO 8 I=2,N
      ABSX=DABS(X(I))
      IF(ABSX-XMAX) 8,8,9
    9 XMAX=ABSX
      NMAX=I
    8 CONTINUE
      RETURN
      END
C
      SUBROUTINE PUT(AL,AR,NDIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION AL(NDIM),AR(NDIM)
      DO 5 I=1,NDIM
    5 AL(I)=AR(I)
      RETURN
      END
C
      SUBROUTINE PUTI(IAL,IAR,NDIM)
      DIMENSION IAL(NDIM),IAR(NDIM)
      DO 5 I=1,NDIM
    5 IAL(I)=IAR(I)
      RETURN
      END
```

B2.2.2 TSOLFUN.f File

```
      DOUBLE PRECISION FUNCTION FUNC(MAXNU,MAXND1,A,X,I)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(MAXNU),X(MAXND1)
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      IF(NFUNC.EQ.1) GO TO 10
      WRITE(6,600)
      STOP
   10 FUNC=A(1)*X(I)*(X(I)+A(2))/(X(I)*(X(I)+A(3))+A(4))
      RETURN
  600 FORMAT('*** FUNCTION UNDEFINED IN FUNC ***')
      END
```

B2.2.3 TSOLOPT.f File

```
      SUBROUTINE OPTION
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
```

```
      CHARACTER *4 MDRUN,LCOMNT,COMNT*80
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKAGP/A(MAXNU),AM(MAXNU),DA(MAXNU),MDINS,ITR,INITR,ICONV,
     +   GAMMA,DELMXS,DELMXM,MDP
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      RETURN
      END
```

B2.2.4 TSOLRD.f File

```
      SUBROUTINE READ1
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (MAXND=100,MAXND1=101,MAXNDN=201)
      PARAMETER (MAXNU=10,MAXNU3=13)
      CHARACTER*4 MDRUN,LCOMNT,MKEND*1,COMNT*80
      COMMON/BLKDTG/X(MAXND1),YDATA(MAXND1),YFIT(MAXND),XMAX(20),
     +   DY(MAXND),DYMAX(20),NSELEC(MAXND),XM(MAXND),YM(MAXND),DELMAX,
     +   NMAX,MDZ,NCALL
      COMMON/BLKRUN/MDRUN,LCOMNT,COMNT
      COMMON/BLKPMT/MDS,NFUNC,NX,NU,MDW,NPRM,PRM(5),ND,ND1,FND
      IF(MDRUN.EQ.'ISDT') GOTO 20
      IF(MDRUN.EQ.'ISDF') GOTO 20
    5 ND=0
   10 ND=ND+1
   13 READ(5,501) MDRUN,MKEND,YDATA(ND),(X(ND),J=1,NX)
      IF(MKEND.NE.'*') GOTO 10
   16 ND=ND-1
      RETURN
   20 READ(5,501) MDRUN
      RETURN
  501 FORMAT(A4,1X,A1,4X,F10.0,10X,2F10.0)
      END
```

B2.3 Test Input and Output Files for TSOLVE

B2.3.1 Test Input File for TSOLVE

```
1,1,4,1,2,1,1,0,100,0
TEST PROBLEM: ENZYME FUNCTION
X
          .1957               4.00
          .1947               2.00
          .1735               1.00
          .1600               .500
          .0844               .250
          .0627               .167
          .0456               .125
          .0342               .100
          .0323               .0833
          .0235               .0714
          .0246               .0625
ISTP *
.1928,.1913,.1231,.1361
```

B2.3.2 Test Output File for TSOLVE

```
 BEST APPROXIMATION PROGRAM TSOLVE    TEST PROBLEM: ENZYME FUNCTION

   MODE OF INPUT FOR STARTING VALUES (MDS) =   1
   FUNCTION NUMBER                (NFUNC) =   1
   NUMBER OF UNKNOWNS                (NU) =   4
   MODE OF WEIGHTING                (MDW) =   2
   MODE OF PRINT                    (MDP) =   1
```

```
 MODE OF FINDING MAXIMUM ERROR   (MDINS) =   1
 MODE OF FIT/CHECK                (MDCK) =   0
 NUMBER OF PARAMETERS             (NPRM) =   0
 NUMBER OF DATA                     (ND) =  11

NO. DELTAS   GAMMA   DELTAMAX VALUES OF UNKNOWNS A(I)
                                    1         2         3         4
 0  0.00E-01 0.0E-01 1.11E-02 1.928E-01 1.913E-01 1.231E-01 1.361E-01
 1  8.12E-03 6.0E-01 8.68E-03 1.880E-01 1.205E-01 4.210E-02 1.141E-01
 2  8.09E-03 1.0E+00 8.14E-03 1.846E-01 1.044E-01 1.104E-02 1.115E-01
 3  8.08E-03 1.0E+00 8.08E-03 1.846E-01 1.052E-01 1.196E-02 1.118E-01
 4  8.08E-03 1.0E+00 8.08E-03 1.846E-01 1.052E-01 1.196E-02 1.118E-01
 5  8.08E-03 1.0E+00 8.08E-03 1.846E-01 1.052E-01 1.196E-02 1.118E-01

LIST OF EXTREMAL POINTS
          X        ABS. DEV.
 1    4.000E+00   -8.084E-03
 2    1.000E+00    8.084E-03
 3    5.000E-01   -8.084E-03
 4    2.500E-01    8.084E-03
 5    8.330E-02   -8.084E-03

NO. OF ITERATIONS =  5    NO. OF INTERNAL ITERATIONS =    40
NO. OF FUNCTION EVALUATIONS =   455

SOLUTIONS AND TOLERANCES FOR ROUNDOFF
             A             DA
 1    1.8463155137E-01   3.6E-05
 2    1.0520566876E-01   1.3E-04
 3    1.1964192157E-02   1.7E-04
 4    1.1178802848E-01   8.5E-05

NO.     X          Y DATA         Y FITTED     ABS. DEV.
 1  4.000E+00  1.95700000E-01  1.87615632E-01 -8.084E-03
 2  2.000E+00  1.94700000E-01  1.87966171E-01 -6.734E-03
 3  1.000E+00  1.73500000E-01  1.81584368E-01  8.084E-03
 4  5.000E-01  1.60000000E-01  1.51915632E-01 -8.084E-03
 5  2.500E-01  8.44000000E-02  9.24843684E-02  8.084E-03
 6  1.670E-01  6.27000000E-02  5.92415189E-02 -3.458E-03
 7  1.250E-01  4.56000000E-02  4.12145169E-02 -4.385E-03
 8  1.000E-01  3.42000000E-02  3.08066928E-02 -3.393E-03
 9  8.330E-02  3.23000000E-02  2.42156316E-02 -8.084E-03
10  7.140E-02  2.35000000E-02  1.97735153E-02 -3.726E-03
11  6.250E-02  2.46000000E-02  1.66197265E-02 -7.980E-03
```

List of
Research Institute for Advanced Science and Technology
Osaka Prefecture University Technical Reports
(RIAST-OPU-TR)

1 “Tables of Charge- and Energy-Deposition Distributions in Elemental Materials Irradiated by Plane-Parallel Electron Beams with Energies between 0.1 and 100 MeV,” Pedro Andreo, Rinsuke Ito and Tatsuo Tabata (1992) [a modified version has been published: T. Tabata, P. Andreo and R. Ito, At. Data Nucl. Data Tables **56**, 105 (1994).]

2 “ALESQ, a Nonlinear Least-Squares Fit Code, and TSOLVE, a Nonlinear Best-Approximation Code,” 2nd revised edition, Tatsuo Tabata and Rinsuke Ito (1997).