UC-405

LBL-28009
Preprint

Center for Advanced Materials

CAM

Submitted to Nuclear Insturments and Methods in Physics Research, B

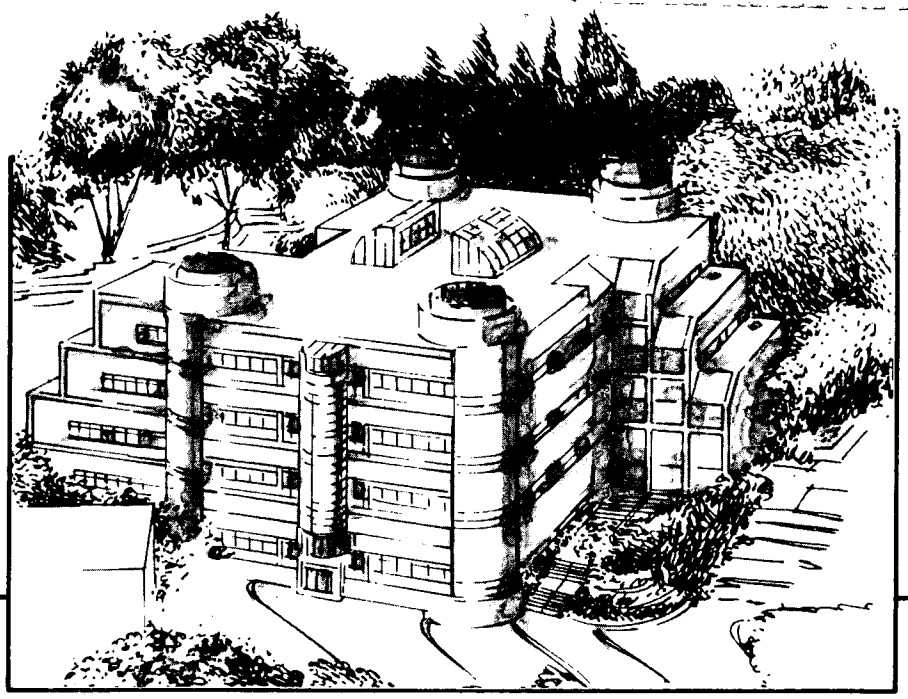
Mössbauer Spectrum Curve Fitting with Personal Computer

Z. Mei and J.W. Morris, Jr.

October 1989

For Reference

Not to be taken from this room



Materials and Chemical Sciences Division

Lawrence Berkeley Laboratory • University of California

ONE CYCLOTRON ROAD, BERKELEY, CA 94720 • (415) 486-4755

Prepared for the U.S. Department of Energy under Contract DE-AC03-76SF00098

LBL-28009 Copy 1
Bldg. 50 Library.

DISCLAIMER

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

LBL 28009

Mössbauer Spectrum Curve Fitting with Personal Computer

Z. Mei and J. W. Morris, Jr.

Center for Advanced Materials
Materials and Chemical Sciences Division
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720
and
Department of Materials Science and Mineral Engineering
University of California

October 1989

This work is supported by the Director, Office of Energy Research, Office of Fusion Energy, Development and Technology Division of the U. S. Department of Energy under Contract No. *DE-AC03-76SF00098*

Mössbauer Spectrum Curve Fitting with Personal Computer

Z. Mei and J. W. Morris, Jr.

Center for Advanced Materials, Lawrence Berkeley Laboratory and
Department of Materials Science and Mineral Engineering,
University of California at Berkeley

A method to perform Mössbauer spectrum curve fitting with a personal computer is presented. A modified Gauss-Newton algorithm for the least squares determination of nonlinear parameters is used in order to write computer codes that perform satisfactorily within the small memory sizes of personal computers. Mössbauer spectrum curve fitting with an IBM AT personal computer takes about 20 minutes of computing time and the quality of the curve fitting is identical with that fitted with a mainframe computer. The combination of the least square curve fitting with graphics capability of the personal computer makes the selection of the initial values of the curve fitting parameters much more convenient.

I. INTRODUCTION

Today most Mössbauer spectrometers are interfaced with personal computers that store spectrum data and perform all the spectrum data processing except for the spectrum curve fitting. This is because Mössbauer spectrum curve fitting involves the determination of nonlinear parameters by least squares calculations. These calculations are usually done on a mainframe computer using generic mathematical library routines that require large amounts of program memory. A specific routine that runs satisfactorily on the same personal computer used for data collection would eliminate the need for expensive mainframes in Mössbauer spectrum curve fitting.

In order to run satisfactorily on a personal computer, a routine that determines nonlinear parameters by the least square method must have modest memory requirements. For example, designating the number of spectrum channels as n (usually 1024 or 2048) and m as the number of parameters to be determined (usually 20-30), standard mainframe routines require several n x m matrices to perform the calculations. Such program memory requirements are beyond the capacity of personal computers and are in fact unnecessary. Presented below is a modified Gauss-Newton algorithm for the least square determination of nonlinear parameters that requires only several m x m matrices for computation. This reduced program memory requirement allows the computation to be performed on a personal computer. The algorithm satisfactorily fits 25 parameters on an IBM-AT in 20 minutes.

II. STATEMENT OF PROBLEM

A Mössbauer spectrum records detector counts, y, at Doppler velocity values, x, into n data pairs:

$$x_1, x_2, \ldots, x_n$$
$$y_1, y_2, \ldots, y_n.$$

This spectrum, Y(x), is defined as the superposition of a baseline, B(x), and peaks, P(x),

$$Y(x) = B(x) + P(x). \tag{1}$$

The baseline is assumed to depend on x quadratically [1],

$$B(x) = B_1 + B_2 x + B_3 x^2 \tag{2}$$

and B_1, B_2, and B_3 are parameters to be determined. In the limit of thin specimens, all the peaks can be assumed to have shapes described by the Lorentzian function,

$$P_I(x) = \frac{H_1}{1 + \frac{(x - C_1)^2}{W_1^2}} = P_I(x; H_1, W_1, C_1) \tag{3}$$

where the parameters to be determined H_1, W_1, and C_1 are the peak height, peak half width at half height, and peak center, respectively.

In the event of quadruple splitting of a nuclear energy level, two peaks with the same height and width will be located symmetrically with respect to a center. Only four parameters* are needed to describe this kind of doublet peak,

$$P_{II}(x) = \frac{H_2}{1 + \frac{(x - C_2 + D_2)^2}{W_2^2}} + \frac{H_2}{1 + \frac{(x - C_2 - D_2)^2}{W_2^2}}$$

$$= P_I(x; H_2, W_2, C_2 - D_2) + P_I(x; H_2, W_2, C_2 + D_2),$$

$$= P_{II}(x; H_2, W_2, C_2, D_2) \tag{4}$$

where D_2 is the symmetric position with respect to the center C_2.

Additionally, nuclear Zeeman splitting can be responsible for the formation of a sextet that may be described as three doublets (peaks 1 & 6, peaks 2 & 5, and peaks 3 & 4),

---

* If the heights of two peaks are not equal due to the Goldanskii-Karyagin Effect, we consider them as two singlets.

$$P_{VI}(x) = P_{II}(x; H_{16}, W_{16}, C_6, D_{61} + 2 * D_{62})$$

$$+ P_{II}(x; H_{25}, W_{25}, C_6, D_{61} + D_{62}) + P_{II}(x; H_{34}, W_{34}, C_6, D_{61}),$$

$$= P_{VI}(x;\ H_{16}, H_{25}, H_{34}, W_{16}, W_{25}, W_{34}, C_6, D_{61}, D_{62}) \qquad (5)$$

where H_{16}, H_{25}, H_{34} are the parameters describing the heights of doublets peaks 1 & 6, 2 & 5, 3 & 4 respectively. Similar definition applies for W_{16}, W_{25}, and W_{34}. D_{61} is the symmetric position of peaks 3 & 4 with respect to center C_6. The distances between peak 1 and 2, 2 and 3, 4 and 5, and 5 and 6 are the same + and designated as D_{62}.

In summary, it is assumed that P(x) can be sufficiently described as

$$P(x) = \sum_{i=1}^{n_I} P_I{}^i(x;\ H_I{}^i, W_I{}^i, C_I{}^i) + \sum_{i=1}^{n_{II}} P_{II}{}^i(x;\ H_{II}{}^i, W_{II}{}^i, C_{II}{}^i, D_{II}{}^i)$$

$$+ \sum_{i=1}^{n_{VI}} P_{VI}{}^i(x;\ H_{16}, H_{25}, H_{34}, W_{16}, W_{25}, W_{34}, C_6, D_{61}, D_{62}), \qquad (6)$$

where n_I, n_{II}, and n_{VI} are the number of the singlets, doublets, and sextets, respectively.

A Mössbauer spectrum with n data pairs is a n time sampling of Equation. (1). But every sampling involves an absolute statistic error of the range $\sqrt{y_i}$. Mössbauer spectrum curve fitting involves the determination of the parameters (B_1, B_2, B_3, H_I, W_I, C_I, H_{II}, W_{II}, C_{II}, D_{II}, H_{16}, H_{25}, H_{34}, W_{16}, W_{25}, W_{34}, C_6, D_{61}, and D_{62}) in (2) and (6) by minimizing the weighted sum of the squares of the residuals, i.e.,

$$\min \sum_{i=1}^{n} [Y(x_i) - y_i]^2 / y_i$$

$$= \min \sum_{i=1}^{n} [Y(x_i; C_1, C_2, \ldots, C_m) - y_i]^2 / y_i, \qquad (7)$$

where C_1, C_2, ..., C_m are the m parameters to be determined.

+ Here a pure Zeenman splitting is assumed, if quadruple interaction is present at same time, the centers of three doublets are not coincedent, then we consider them as three independent doublets.

III. ALGORITHM

The problem to be solved in (7) is the least squares determination of the unknown parameters. If $Y(x; C_1, C_2, ..., C_m)$ is a linear function of $C_1, C_2, ..., C_m$, then there exists a closed form solution. If, however, $f(x; C_1, C_2, ..., C_m)$ is not a linear function of $C_1, C_2, ..., C_m$, then it may be Taylor expanded with respect to $C_1, C_2, ..., C_m$ about initially guessed values $C_1^{(1)}, C_2^{(1)}, ..., C_m^{(1)}$. If only linear terms are kept in the Taylor expansion, then f(x) is approximated as a linear function of $C_1 - C_1^{(1)}, C_2 - C_2^{(1)}, ..., C_m - C^{(1)}$. Hence, $C_1, C_2, ..., C_m$ can be determined from the linear least square solution. Thus produced $C_1, C_2, ..., C_m$ are then set as the new reference point for the Taylor expansion, and the linear least square calculation is executed again to produce another new set of $C_1, C_2, ..., C_m$. This iterating method continues until a satisfactory precision is met. In this section, we will discuss the linear least square solution, the algorithm for nonlinear least square problem, and then the modification to the algorithm.

Linear Least Square Solution

For convenience and simplicity, vectorial formulation is used. If the following definitions are made,

$$\mathbf{y} = [\sqrt{y_1}, \sqrt{y_2}, ..., \sqrt{y_n}]^T. \tag{8}$$

$$\mathbf{C} = [C_1, C_2, ..., C_m]^T, \tag{9}$$

$$\mathbf{Y(C)} = [Y(x_1; \mathbf{C}) / \sqrt{y_1}, Y(x_2; \mathbf{C}) / \sqrt{y_2}, ..., Y(x_n; \mathbf{C}) / \sqrt{y_n}], \tag{10}$$

then expression (7) can be rewritten as

$$\min \ (\mathbf{Y(C)} - \mathbf{y})^T (\mathbf{Y(C)} - \mathbf{y}) = \min \mathbf{f(C)}^T \mathbf{f(C)}. \tag{11}$$

If $\mathbf{f(C)}$ is a linear function of $\mathbf{C}$, then we can write $\mathbf{f(C)}$ as

$$\mathbf{f(C)} = \mathbf{A\ C} + \mathbf{b}, \tag{12}$$

where $\mathbf{A}$ is a n x m matrix, $\mathbf{b}$ is a n dimensional vector $[b_1, b_2, ..., b_n]^T$. Therefore,

$$\mathbf{f(C)}^T \mathbf{f(C)} = (\mathbf{A\ C} + \mathbf{b})^T (\mathbf{A\ C} + \mathbf{b})$$

$$= \mathbf{C}^T \mathbf{A}^T \mathbf{A\ C} + 2\, \mathbf{b}^T \mathbf{A\ C} + \mathbf{b}^T \mathbf{b}. \tag{13}$$

Differentiating $\mathbf{f(C)}^T \mathbf{f(C)}$ with respect to $\mathbf{C}$, one finds

$$\nabla[\ \mathbf{f(C)}^T \mathbf{f(C)}] = 2\, \mathbf{A}^T \mathbf{A\ C} + 2\, \mathbf{A}^T \mathbf{b}. \tag{14}$$

The solution to (11), $\mathbf{C}^*$, is found by solving the simultaneous linear equations,

$$\mathbf{A}^T \mathbf{A}\, \mathbf{C}^* = -\, \mathbf{A}^T \mathbf{b}. \tag{15}$$

Gauss-Newton Algorithm

Unfortunately, $\mathbf{f}(\mathbf{C})$ is not a linear function for most of elements in $\mathbf{C}$ as seen in Equations (2)-(5). Therefore Equation (11) is a nonlinear least square problem. The three most common algorithms [2] used to solve the problem are those of steepest descent, Gauss-Newton, and Marquardt [3]. The steepest descent method has the advantage of certainty of convergency and the drawback of progressively slow convergent speed after first few iterations. The Gauss-Newton method has the advantage of high convergent speed but may diverge if the selection of initial solution of $\mathbf{C}$ is far away from the true value. The Marquardt method has both the certainty of convergency and high convergent speed, but the algorithm is complicated and difficult to program with the small memory size of a personal computer. We adopt the modified Gauss-Newton method [4] that has the certainty of convergency and high convergent speed, and is easy to program with small memory size.

The essence of the Gauss-Newton method is to use *linear* least square method to approach *nonlinear* least square solution by iteration. During the kth iteration, the first order Taylor expansion of $\mathbf{f}(\mathbf{C})$ at $\mathbf{C}^k$ may be written as, ($\mathbf{C}^k$ represents result of $\mathbf{C}$ produced by the (k-1)th iteration)

$$\mathbf{f}(\mathbf{C}) \cong \mathbf{f}(\mathbf{C}^k) + \mathbf{A}(\mathbf{C}^k)\,(\mathbf{C} - \mathbf{C}^k). \tag{16}$$

where the Jacobian of $\mathbf{f}(\mathbf{C})$ at $\mathbf{C}^k$, $\mathbf{A}(\mathbf{C}^k)$, is defined as

$$\mathbf{A}(\mathbf{C}^k) = \begin{pmatrix} \frac{\partial f_1(\mathbf{C}^k)}{\partial C_1} & \cdots & \frac{\partial f_1(\mathbf{C}^k)}{\partial C_m} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_n(\mathbf{C}^k)}{\partial C_1} & \cdots & \frac{\partial f_n(\mathbf{C}^k)}{\partial C_m} \end{pmatrix} = \begin{pmatrix} \frac{\partial Y(x_1;\mathbf{C}^k)}{\sqrt{y_1}\partial C_1} & \cdots & \frac{\partial Y(x_1;\mathbf{C}^k)}{\sqrt{y_1}\partial C_m} \\ \cdots & \cdots & \cdots \\ \frac{\partial Y(x_n;\mathbf{C}^k)}{\sqrt{y_n}\partial C_1} & \cdots & \frac{\partial Y(x_n;\mathbf{C}^k)}{\sqrt{y_n}\partial C_m} \end{pmatrix} \tag{17}$$

The solution to the linear least square problem,

$$\min\, [\, \mathbf{f}(\mathbf{C}^k) + \mathbf{A}(\mathbf{C}^k)\,(\mathbf{C} - \mathbf{C}^k)]^T\, [\mathbf{f}(\mathbf{C}^k) + \mathbf{A}(\mathbf{C}^k)\,(\mathbf{C} - \mathbf{C}^k)]$$

$$= \min\, [\, \mathbf{f}(\mathbf{C}^k) + \mathbf{A}(\mathbf{C}^k)\, \mathbf{P}^k]^T\, [\mathbf{f}(\mathbf{C}^k) + \mathbf{A}(\mathbf{C}^k)\, \mathbf{P}^k] \tag{18}$$

can be obtained by solving the simultaneous linear equations,

$$\mathbf{A}(\mathbf{C}^k)^T\, \mathbf{A}(\mathbf{C}^k)\, \mathbf{P}^k = -\, \mathbf{A}(\mathbf{C}^k)^T\, \mathbf{f}(\mathbf{C}^k), \tag{19}$$

and the new value of $\mathbf{C}$ will be

$$\mathbf{C}^{k+1} = \mathbf{C}^k + \mathbf{P}^k. \tag{20}$$

Modification to Gauss-Newton Algorithm

If during kth iteration, the determinant of $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$ becomes zero, there will be no unique solution to (19). In addition, even if the determinant of $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$ is not zero, the solution $\mathbf{P}^k$ determined from (19) does not guarantee $\mathbf{f}(\mathbf{C}^{k+1})^T\,\mathbf{f}(\mathbf{C}^{k+1})$ being less than $\mathbf{f}(\mathbf{C}^k)^T\,\mathbf{f}(\mathbf{C}^k)$. This becomes apparent when the chosen initial value $\mathbf{C}^0$ is far away from the true solution to (11) as the iteration then diverges.

It has been shown [4] that although $\mathbf{P}^k$ can not guarantee $\mathbf{f}(\mathbf{C}^{k+1})^T\,\mathbf{f}(\mathbf{C}^{k+1})$ being less than $\mathbf{f}(\mathbf{C}^k)^T\,\mathbf{f}(\mathbf{C}^k)$, $\mathbf{P}^k$ is along the direction that makes $\mathbf{f}(\mathbf{C})^T\,\mathbf{f}(\mathbf{C})$ decrease. Therefore, after solving (19) to get $\mathbf{P}^k$, we do not use Equation (20) to determine $\mathbf{C}^{k+1}$, but use $\mathbf{P}^k$ as a linear search direction,

$$\mathbf{C}^{k+1} = \mathbf{C}^k + \lambda \mathbf{P}^k, \tag{21}$$

determined by minimizing $\mathbf{f}(\mathbf{C}^k)^T\,\mathbf{f}(\mathbf{C}^k)$ with respect the scalar λ.

If the determinant of $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$ becomes zero, then $\mathbf{P}^k$ is set as $-\,\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{f}(\mathbf{C}^k)$ which is the steepest decent direction [2], and steepest decent method is then used for this iteration.

IV. PROGRAMMING

The central part of the curve fitting program is to solve the simultaneous linear equations in Equation (19). $\mathbf{A}(\mathbf{C}^k)$ is a n x m matrix, where n is usually 1024 or 2048, m is about 20 or 30 depending on the number of peaks in a spectrum. But $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$ is a matrix of only m x m. The small memory size of a personal computer is insufficient to store $\mathbf{A}(\mathbf{C}^k)$, but is quite adequate for storage of $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$. In addition, further memory is free from the fact that $\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)$ is a symmetrical matrix. Therefore only a single m x m array, F_{ij}, and a 1 x m array, b_i, need to be defined in the program codes. Equation (19) can now be represented as

$$\sum_{j=1}^{m} F_{ij}\,P_j = -\,b_i, \qquad (i = 1, 2, \ldots, m) \tag{22}$$

with F_{ij} calculated as

$$F_{ij} = [\mathbf{A}(\mathbf{C}^k)^T\,\mathbf{A}(\mathbf{C}^k)]_{ij}$$

$$= \frac{1}{\sqrt{y_i}\,\sqrt{y_j}} \sum_{l=1}^{n} \frac{\partial Y(x_l;\mathbf{C}^k)}{\partial C_i}\frac{\partial Y(x_l;\mathbf{C}^k)}{\partial C_j}. \tag{23}$$

The expression $\frac{\partial Y(x_l; \mathbf{C}^k)}{\partial C_i}$ is the value of the derivative of Y(x) defined in equation (1) with respect to ith parameter C_i when $\mathbf{C} = \mathbf{C}^k$. The formula of the derivative of Y(x) with respect to ith parameter C_i can be solved by simply differentiating $Y(x; C_1, C_2, ..., C_m)$ with respective to C_i. For example, if C_i stands for H_1, referring to Equation (3), $\frac{\partial Y(x; \mathbf{C})}{\partial H_1}$ will be

$$\frac{\partial Y(x; \mathbf{C})}{\partial H_1} = \frac{1}{1 + \frac{(x - C_1)^2}{W_1{}^2}} \cdot \tag{24}$$

The value of $\frac{\partial Y(x_l; \mathbf{C}^k)}{\partial H_1}$ is calculated from (24) by inserting the values of x (= x_l), C_1 (= $C_1{}^k$), and W_1 (= $W_1{}^k$).

The value of b_i is calculated as

$$b_i = [\mathbf{A}(\mathbf{C}^k)^T \, \mathbf{f}(\mathbf{C}^k)]_i$$

$$= \sum_{j=1}^{n} \frac{1}{y_j} \frac{\partial Y(x_j; \mathbf{C}^k)}{\partial C_i} (Y(x_j; \mathbf{C}^k) - y_j). \tag{25}$$

After F_{ij} and b_i are constructed, Equations (22) can be solved using the Gauss elimination method with pivotal condensation [5].

The flow chart of the program that embodies the algorithm is shown in Fig. 1. The fitting curve process starts with an initial guess of the parameters $\mathbf{C}^0$, calculates F_{ij} and b_i with $\mathbf{C}^0$ using Equations (23) and (25), then solves the simultaneous Equations (22) to get $\mathbf{P}^1$, linearly searches (Equation (21)) along $\mathbf{P}^1$ to obtain the new solution $\mathbf{C}^1$, reconstructs Equation (22) with $\mathbf{C}^1$ and solves it again. The iteration continues until an appropriate stop criterion is satisfied.

The stop criterion we used is

$$\max\{ \, | C_i^{k+1} - C_i^k | \, / | C_i^k | \, \} < d \tag{26}$$

where d is a very small number (e.g. = 0.0001). Satisfaction of (26) simply means stabilization of the iteration results. The quality of the fitting is indicated by the "goodness" parameter χ^2 [1],

$$\chi^2 = (\sum_{i=1}^{n} [Y(x_i) - y_i]^2 / y_i \,) \, / \, (n - m). \tag{27}$$

If χ^2 is close to 1, the fitting curve is close to experimental Mössbauer spectrum; if χ^2 is very much larger than 1, the fitting curve deviates from the experimental curve, the fitting must be re-done by choosing different initial values, C_1^0, C_2^0, ..., C_m^0.

V. RESULTS

We have been using our curve fitting program (written in FORTRAN, compiled by Microsoft Fortran Compiler and run in an IBM AT personal computer) for one and half years. The performance is rather satisfactory. Although the calculating speed is much slower than that of a mainframe computer, (for example, fitting a 1024 channel pure Fe spectrum with 6 singlets of 21 parameters, the program runs about 20 minutes in the personal computer and only a few seconds in a VAX mainframe computer), the speed is acceptable. The quality of the curve fitting with the personal computer is exactly the same as that with a mainframe computer. Fig. 2 (a) and (b) show two examples of Mössbauer spectra fitted with the personal computer.

Perhaps one of the advantages of using a personal computer for curve fitting over using a mainframe computer is convenience to choose the initial values of the parameters. Since any curve fitting program locates a local minimum point rather than the global minimum point, the choice of the initial values of the parameters is critical and is usually done by try and error, especially when peaks in a spectrum overlap one another very much. Our curve fitting program has two steps, "eye fit" and least square fit. The "eye fit" makes use of graphic package available for personal computer, and plots both the spectrum curve and the fitted curve on the computer monitor screen. The program can be quested to change any parameter (e.g. peak height, width ...) and replot the curves. After several interactions, approximate values of the parameters are found. Those values of the parameters are then set as the initial values of the parameters for the least square curve fitting to find exact values of the parameters. Since the mainframe computer is usually designed for fast and complicated scientific calculation, its graphics capability is not as easily employed as that of a personal computer.

VI. ACKNOWLEDGEMENT

The authors appreciate the helpful discussion with Dr. Chuck Violet, Lawrence Livermore National Laboratory, and Mr. Mark McCormack, Lawrence Berkeley Laboratory. This work was supported by the Director, Office of Energy Research, Office of Fusion Energy, Development and Technology Division of the U. S. Department of Energy, under Contract No. DE-AC03-76SF00098.

REFERENCE

[1] G. J. Long, ed., *Mössbauer Spectroscopy Applied to Inorganic Chemistry,* vol. 1, Plenum, New York and London, 1984.

[2] J. E. Dennis, Jr, and Robert B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[3] D. W. Marquardt, *J. Soc. Indust. Appl. Math*, 11 (1963) 431.

[4] J. Xue, *Optimization Principles and Methods*, (in Chinese), Chapter V, vol. 1, North East Engineering Institute, China, 1981.

[5] D. D. McCracken and W. S. Dorn, *Numerical Methods and Fortran Programming*, John Wiley and Sons Inc., New York, 1964.

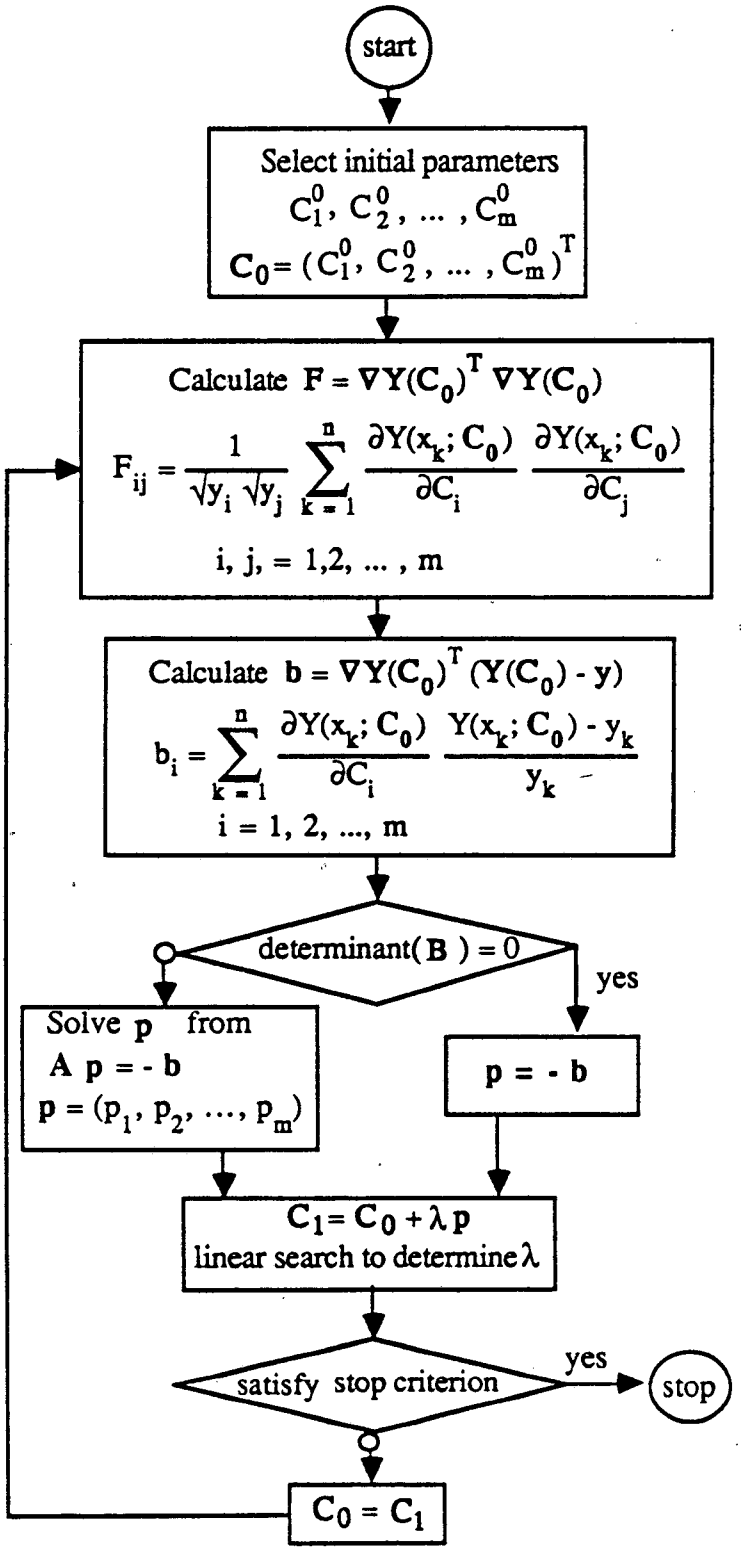


Figure 1. The flow chart of the curve fitting program.

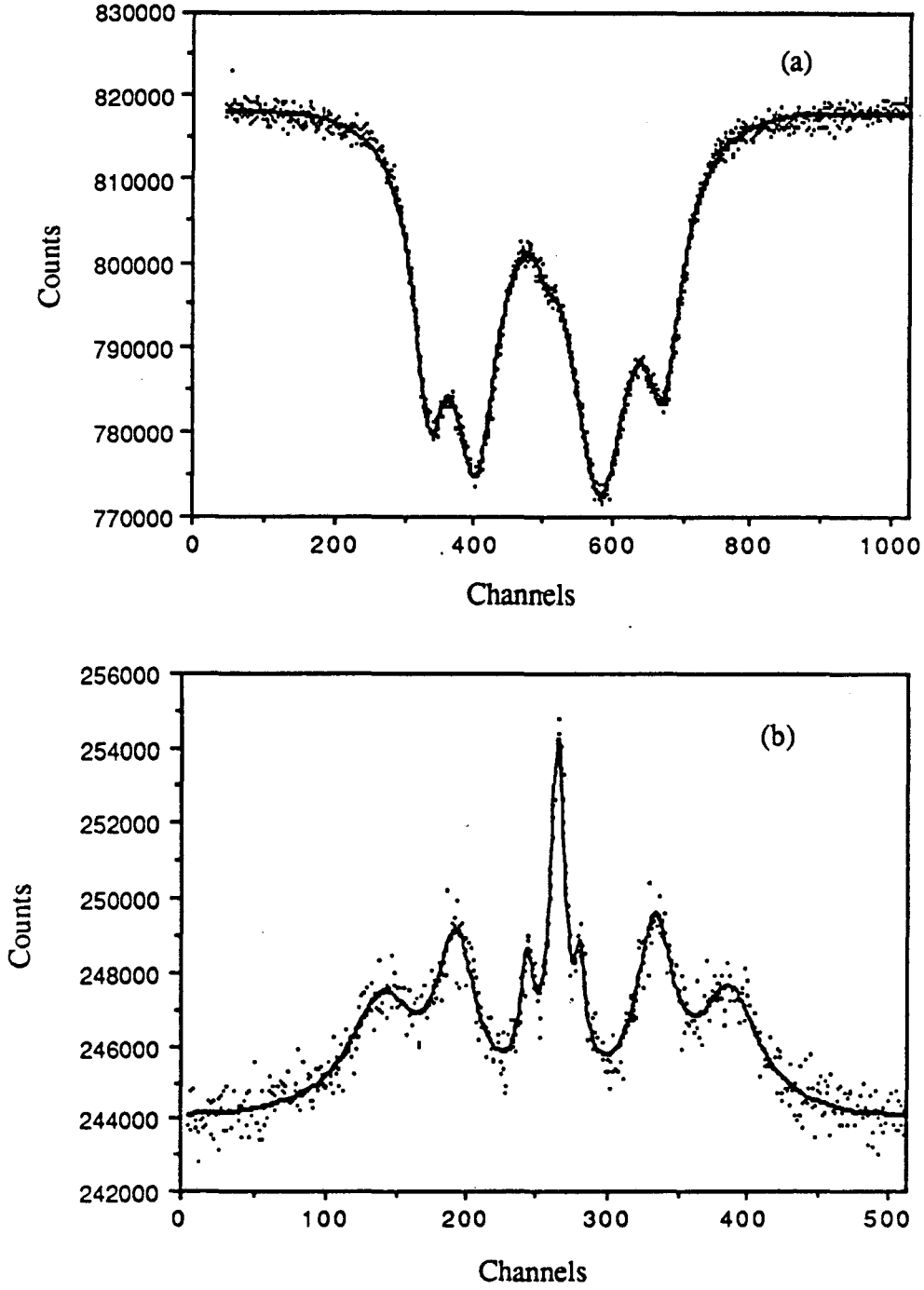


Figure 2: Two examples of Mössbauer spectra curve-fitted with an IBM AT personal computer, (a) transmission spectrum of $HoBa_2(Cu_{0.95}Fe_{0.05})_3O_{6.8}$;(b) backscattering spectrum of the deformation induced martensite on a J_{IC} test fracture surface of 304 stainless steel.

LAWRENCE BERKELEY LABORATORY
CENTER FOR ADVANCED MATERIALS
1 CYCLOTRON ROAD
BERKELEY, CALIFORNIA 94720