NUCLEAR DATA AND MEASUREMENTS SERIES

ANL/NDM-128

A Least-Squares Computational "Tool Kit"

by

Donald L. Smith

April 1993

ARGONNE NATIONAL LABORATORY,
ARGONNE, ILLINOIS 60439, U.S.A.

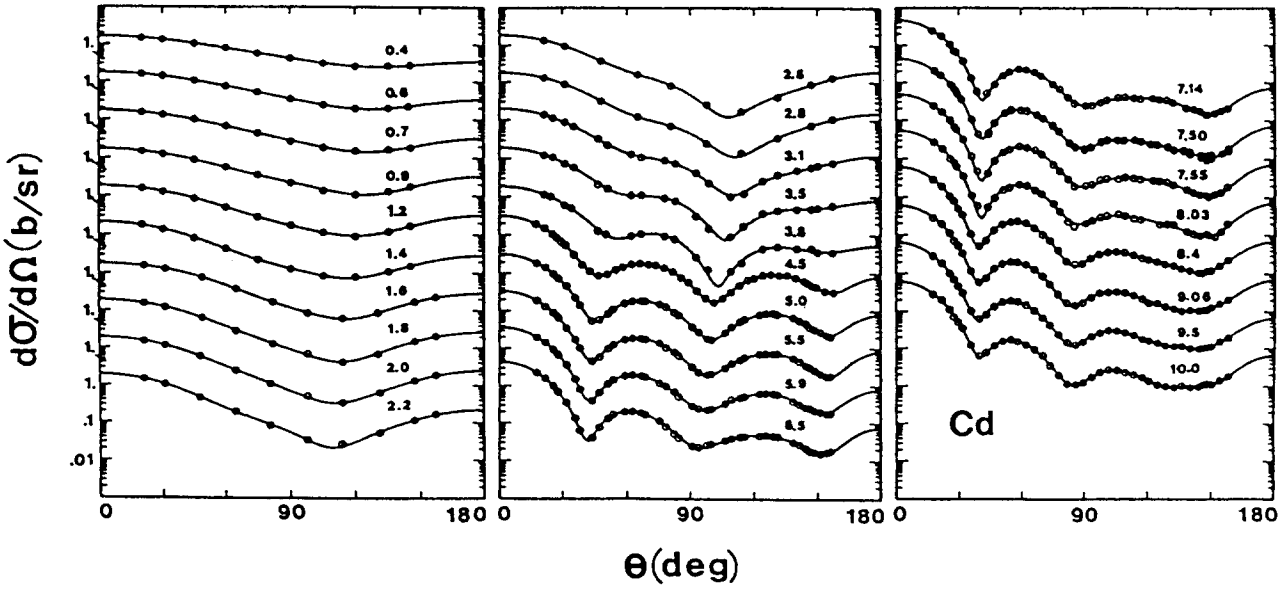
NUCLEAR DATA AND MEASUREMENTS SERIES

ANL/NDM-128

A LEAST-SQUARES COMPUTATIONAL "TOOL KIT"

Donald L. Smith

April 1993



ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Operated by THE UNIVERSITY OF CHICAGO
for the U. S. DEPARTMENT OF ENERGY
under Contract W-31-109-Eng-38

Reproduced from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831
Prices available from (615) 576-8401

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

ANL/NDM-128

A LEAST-SQUARES COMPUTATIONAL "TOOL KIT"[a]

by

Donald L. Smith

Engineering Physics Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439
U.S.A.

April 1993

STATISTICAL DATA ANALYSIS METHODS. Bayesian analysis. Simple least-squares analysis. Generalized least-squares analysis. Model parameters. Linear problems. Non-linear problems. Techniques of linearization. Design matrices. Correlated data. Covariance matrices. Data fitting. Normal distribution. Chi-square test. Matrix analysis. Computer algorithms. Application methods.

---

a This work was supported by the U.S. Department of Energy, Energy Research Programs, under contract W-31-109-Eng-38.

NUCLEAR DATA AND MEASUREMENTS SERIES

The Nuclear Data and Measurements Series presents results of studies in the field of microscopic nuclear data. The primary objective is the dissemination of information in the comprehensive form required for nuclear technology applications. This Series is devoted to: a) measured microscopic nuclear parameters, b) experimental techniques and facilities employed in measurements, c) the analysis, correlation and interpretation of nuclear data, and d) the evaluation of nuclear data. Contributions to this Series are reviewed to assure technical competence and, unless otherwise stated, the contents can be formally referenced. This Series does not supplant formal journal publication, but it does provide the more extensive information required for technological applications (e.g., tabulated numerical data) in a timely manner.

TABLE OF CONTENTS

INFORMATION ABOUT OTHER ISSUES OF THE ANL/NDM SERIES

A list of titles and authors for all the previous issues appears in each report of the series. The list for reports ANL/NDM-1 through ANL/NDM-75 appears in ANL/NDM-76, and ANL/NDM-91 contains the list for reports ANL/NDM-76 through ANL/NDM-90. Below is the list for ANL/NDM-91 up to the current report. Requests for a complete list of titles or for copies of previous reports should be directed to:

Donald L. Smith
Nuclear Data Program
Engineering Physics Division
Building 207
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439
U.S.A.

ANL/NDM-91 A.B. Smith, P.T. Guenther and R.D. Lawson, *On the Energy Dependence of the Optical Model of Neutron Scattering from Niobium*, May 1985.

ANL/NDM-92 Donald L. Smith, *Nuclear Data Uncertainties (Vol.-I): Basic Concepts of Probability*, December 1988.

ANL/NDM-93 D.L. Smith, J.W. Meadows and M.M. Bretscher, *Integral Cross-section Measurements for* $^{7}Li(n,n't)^{4}He$, $^{27}Al(n,p)^{27}Mg$, $^{27}Al(n,a)^{24}Na$, $^{58}Ni(n,p)^{58}Co$, *and* $^{60}Ni(n,p)^{60}Co$ *Relative to* ^{238}U *Neutron Fission in the Thick-target* $^{9}Be(d,n)^{10}B$ *Spectrum at* E_d = 7 *MeV*, October 1985.

ANL/NDM-94 A.B. Smith, D.L. Smith, P. Rousset, R.D. Lawson and R.J. Howerton, *Evaluated Neutronic Data File for Yttrium*, January 1986.

ANL/NDM-95 Donald L. Smith and James W. Meadows, *A Facility for High-intensity Neutron Irradiations Using Thick-target Sources at the Argonne Fast-neutron Generator*, May 1986.

ANL/NDM-96 M. Sugimoto, A.B. Smith and P.T. Guenther, *Ratio of the Prompt-fission-neutron Spectrum of Plutonium-239 to that of Uranium-235*, September 1986.

ANL/NDM-97 J.W. Meadows, *The Fission Cross Sections of* ^{230}Th, ^{232}Th, ^{233}U, ^{234}U, ^{236}U, ^{238}U, ^{237}Np, ^{239}Pu, *and* ^{242}Pu *Relative* ^{235}U *at 14.74-MeV Neutron Energy*, December 1986.

ANL/NDM-98 J.W. Meadows, *The Fission Cross Section Ratios and Error Analysis for Ten Thorium, Uranium, Neptunium and Plutonium Isotopes at 14.74-MeV Neutron Energy*, March 1987.

ANL/NDM-99 Donald L. Smith, *Some Comments on the Effects of Long-range Correlations in Covariance Matrices for Nuclear Data*, March 1987.

ANL/NDM-100 A.B. Smith, P.T. Guenther and R.D. Lawson, *The Energy Dependence of the Optical-model Potential for Fast-neutron Scattering from Bismuth*, May 1987.

ANL/NDM-101 A.B. Smith, P.T. Guenther, J.F. Whalen and R.D. Lawson, *Cobalt: Fast Neutrons and Physical Models*, July 1987.

ANL/NDM-102 D.L Smith, *Investigation of the Influence of the Neutron Spectrum in Determinations of Integral Neutron Cross-section Ratios*, November 1987.

ANL/NDM-103 A.B. Smith, P.T. Guenther and B. Micklich, *Spectrum of Neutrons Emitted From a Thick Beryllium Target Bombarded With 7-MeV Deuterons*, January 1988.

ANL/NDM-104 L.P. Geraldo and D.L. Smith, *Some Thoughts on Positive Definiteness in the Consideration of Nuclear Data Covariance Matrices*, January 1988.

ANL/NDM-105 A.B. Smith, D.L. Smith, P.T. Guenther, J.W. Meadows, R.D. Lawson, R.J. Howerton and T. Djemil, *Neutronic Evaluated Nuclear-data File for Vanadium*, May 1988.

ANL/NDM-106 A.B. Smith, P.T. Guenther and R.D. Lawson, *Fast-neutron Elastic Scattering from Elemental Vanadium*, March 1988.

ANL/NDM-107 P. Guenther, R. Lawson, J. Meadows, M. Sugimoto, A. Smith, D. Smith and R. Howerton, *An Evaluated Neutronic Data File for Elemental Cobalt*, August 1988.

ANL/NDM-108 M. Sugimoto, P.T. Guenther, J.E. Lynn, A.B. Smith and J.F. Whalen, *Some Comments on the Interaction of Fast Neutrons with Beryllium*, November 1988.

ANL/NDM-109 P.T. Guenther, R.D. Lawson, J.W. Meadows, A.B. Smith, D.L. Smith, and M. Sugimoto, *An Evaluated Neutronic Data File for Bismuth*, November 1989.

ANL/NDM-110 D.L. Smith and L.P. Geraldo, *A Vector Model for Error Propagation*, March 1989.

ANL/NDM-111 J.E. Lynn, *Fifty Years of Nuclear Fission*, June 1989.

ANL/NDM-112 S. Chiba, P.T. Guenther, and A.B. Smith, *Some Remarks on the Neutron Elastic- and Inelastic-scattering Cross Sections of Palladium*, May 1989.

ANL/NDM-113 J.E. Lynn, *Resonance Effects in Neutron Scattering Lengths*, June 1989.

ANL/NDM-114 A.B. Smith, R.D. Lawson and P.T. Guenther, *Ambiguities in the Elastic Scattering of 8-MeV Neutrons from Adjacent Nuclei*, October 1989.

ANL/NDM-115 A.B. Smith, S. Chiba, D.L. Smith, J.W. Meadows, P.T. Guenther, R.D. Lawson, and R.J. Howerton, *Evaluated Neutronic File for Indium*, January 1990.

ANL/NDM-116 S. Chiba, P.T. Guenther, R.D. Lawson and A.B. Smith, *Neutron Scattering from Elemental Indium, the Optical Model, and the Bound-state Potential*, June 1990.

ANL/NDM-117 Donald L. Smith and Luiz P. Geraldo, *An Evaluation of the Nb-93(n,n')Nb-93m Dosimeter Reaction for ENDF/B-VI*, November 1990.

ANL/NDM-118 J.W. Meadows, *Characteristics of the Samples in the FNG Fission Deposit Collection*, December 1990.

ANL/NDM-119 S. Chiba, P.T. Guenther, A.B. Smith, M. Sugimoto and R.D. Lawson, *Fast-neutron Interaction with Elemental Zirconium and the Dispersive Optical Model*, June 1991.

ANL/NDM-120 A.B. Smith, P.T. Guenther, J.F. Whalen and S. Chiba, *Fast-neutron Total and Scattering Cross Sections of Ni-58 and Nuclear Models*, April 1991.

ANL/NDM-121 Satoshi Chiba and Donald L. Smith, *A Suggested Procedure for Resolving an Anomaly in Least-squares Data Analysis Known as "Peelle's Pertinent Puzzle" and the General Implications for Nuclear Data Evaluations*", September 1991.

ANL/NDM-122 Dominique Feautrier and Donald L. Smith, *Development and Testing of a Deuterium Gas Target Assembly for Neutron Production via the H-2(d,n)He-3 Reaction at a Low-energy Accelerator Facility*, March 1992.

ANL/NDM-123 Donald L. Smith and Edward T. Cheng, *A Review of Nuclear Data Needs and Their Status for Fusion Reactor Technology with Some Suggestions on a Strategy to Satisfy the Requirements*, September 1991.

ANL/NDM-124 J.W. Meadows, *The Thick-target Be-9(d,n) Neutron Spectra for Deuteron Energies Between 2.6 and 7.0 MeV*, November 1991.

ANL/NDM-125 A.B. Smith and P.T. Guenther, *Fast-neutron Scattering Near Shell Closures: Scandium*, August 1992.

ANL/NDM-126 A.B. Smith, J.W. Meadows and R.J. Howerton, *A Basic Evaluated Neutronic Data File for Elemental Scandium*, November 1992.

ANL/NDM-127 A.B. Smith and P.T. Guenther, *Fast-neutron Interaction with Collective Cadmium Nuclei*, November 1992.

A LEAST-SQUARES COMPUTATIONAL "TOOL KIT"[a]

by

Donald L. Smith

Engineering Physics Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439
U.S.A.

ABSTRACT

The information assembled in this report is intended to offer a useful computational "tool kit" to individuals who are interested in a variety of practical applications for the least-squares method of parameter estimation. The fundamental principles of Bayesian analysis are outlined first and these are applied to development of both the simple and the generalized least-squares conditions. Formal solutions that satisfy these conditions are given subsequently. Their application to both linear and non-linear problems is described in detail. Numerical procedures required to implement these formal solutions are discussed and two utility computer algorithms are offered for this purpose (codes LSMOD and GLSMOD written in FORTRAN). Some simple, easily understood examples are included to illustrate the use of these algorithms. Several related topics are then addressed, including the generation of covariance matrices, the role of iteration in applications of least-squares procedures, the effects of numerical precision and an approach that can be pursued in developing data analysis packages that are directed toward special applications.

---

[a] This work was supported by the U.S. Department of Energy, Energy Research Programs, under contract W-31-109-Eng-38.

1. INTRODUCTION

An important aspect of data analysis is the derivation of best-estimate values for certain desired parameters, after having been provided with a set of data and, possibly, some prior information on the parameters themselves. An important technique used for this purpose is the method of least squares. This approach was first suggested by K. Gauss many years ago. Today, elementary versions of the least-squares method are commonly taught in mathematics and physics courses. They are widely applied by investigators. Frequently, these are incorporated into sophisticated computer software packages which also offer many options for handling and plotting data. However, there are more advanced and less frequently considered formulations of this method that ought to be considered. These offer possibilities for manipulation and utilization of data which are generally not appreciated adequately by the scientific community. In particular, it is possible to address non-linear problems, to include weighted and correlated data, to allow for the existence of prior information about the parameters being sought, and even to introduce constraints. However, it is rather difficult for the casual user (who has limited time available to develop the skills) to conveniently extract from the plethora of bewildering information in the literature (much of it highly mathematical in nature) exactly what is needed in order to address specific problems using advanced least-squares techniques. As a consequence of this "knowledge barrier", these advanced formulations of the least-squares method tend to be largely ignored by most investigators. This is unfortunate because they really are not that difficult to understand or formidable to apply, and the rewards of doing so are well worth expending extra effort to learn the skills.

The objective of this report is to gather conveniently in one place that information which is needed to correctly apply some of the more useful advanced least-squares methodologies. Although this goal is a practical one, some attention is given also to the theoretical foundations of the least-squares method. This report aims to bridge the above-mentioned "knowledge barrier". Since no new ground is broken here, this report does not target those investigators who are already experienced in the use of these techniques. However, researchers who routinely employ existing computer codes which incorporate least-squares analysis techniques, but do not possess a well-developed understanding of the underlying subject, will also profit from a study of this report. It will help them to utilize these various codes in a proper and effective manner.

Section 2 explores the foundations of the least-squares method from a Bayesian viewpoint and discusses models used to relate the sought-after parameters to the available data. It is shown that the "simple" least-squares algorithm results when there is no prior knowledge of the parameters sought (Section 3). The "generalized" least-squares algorithm is required when prior knowledge of the parameters and their uncertainties exists and must be combined with

new data to provide revised estimates for these parameters (Section 4). Versions of the least-squares method which allow for constraints between the parameters to be considered will not be treated in this report. Sections 3 and 4 have been prepared to serve as more or less self-contained instructional packages. In each section, the corresponding formalism is discussed, numerical procedures are described and some illustrative examples are presented. For example, a reader who is interested mainly in simple least-squares analysis could focus his attention principally on Section 3 and Appendix A. However, such a shallow approach to this subject is not recommended. Both the simple and generalized least-squares techniques are widely applicable. The experienced researcher ought to become conversant with both techniques and, therefore, able to discern which of them is most suitable for dealing with any particular problem he might encounter.

In order to gain proficiency in the use of these methods, the user will benefit from a study of the supplementary topics discussed in Section 5. Some thoughts are offered there on the generation of covariance matrices, on the role of iteration in use of least-squares methods, on the effects of numerical precision and on an approach toward the development of user-customized analysis packages derived from two basic algorithms (LSMOD and GLSMOD).

Two FORTRAN programs are provided here (LSMOD and GLSMOD), complete with test problems. The reader can, with some ingenuity, put these directly to use in solving problems without any further modifications. However, the reader need not be limited to using these codes. For example, there exist sophisticated commercial software packages which permit the use of symbolic expressions to evaluate functions, to carry out matrix operations, etc. These can also be employed profitably in solving least-squares problems. To this end, the fundamental equations upon which such analyses are based are clearly indicated in the text of Sections 3 and 4.

A few comments concerning notation are in order. Most of the equations which appear in this report involve matrices. The convention of denoting matrices by boldface letters will be followed here (e.g., $\mathbf{A}$, $\mathbf{V}$, $\mathbf{p}$, etc.). A vector is simply a matrix with unit dimension in at least one direction. A scalar is just a degenerate matrix with unit dimension in both directions. However, mathematical quantities which are always scalar in nature are never indicated using boldface symbols.

No attempt has been made to provide an exhaustive collection of references in this report. The reference list here is limited to a few documents which might prove of some interest to those readers who wish to explore the subject beyond the confines of this report.

2. FUNDAMENTAL PRINCIPLES

2.1 Bayesian Analysis

Least-squares conditions are generally introduced *ad hoc* in classical statistics. These conditions lead to algorithms that provide "estimators" for the parameters sought (e.g., Zeh70). Such estimators may possess some desirable properties, e.g., minimum variance [Zeh90]. It can be shown that these least-squares conditions result from very fundamental principles (e.g., Fro86 and Smi91). Their development involves consideration of conditional probability, Bayes theorem and the principle of maximum entropy. In view of the insight provided, it is worthwhile devoting some attention to these basic ideas. The logic is outlined below (see Smi91 for further details).

This discussion begins with conditional probability. First, we define the notation. Assume that E is an exhaustive collection of events (equivalent to sets) associated with a particular sampling procedure. Suppose that X and Y are arbitrary, non-trivial events (non-empty sets) belonging to E, i.e., $X \in E$ and $Y \in E$. We shall employ P to signify probability. Let $P(X|Y)$ denote the conditional probability of X given that Y has occurred while $P(X)$ is the unconditional probability of X. It is clear that $P(X) = P(X|E)$ since E is the certain event which always occurs. Probability must be normalized, thus $P(E) = P(E|E) = 1$, $P(X) \leq 1$ and $P(X|Y) \leq 1$. Now, let A_k (k=1,n) be a collection of non-trivial events (non-empty sets) belonging to E ($A_k \in E$). Suppose that these events A_k are mutually independent, i.e., that A_k and A_j have no elements in common for all $j \neq k$. Finally, suppose that $E = \Sigma_{k=1,n} A_k$, i.e., that E can be considered as formed by the union of all these A_k. Such a collection of events A_k is known as a partition of E. It is not unique since many partitions of E can be conceived with these properties. Then, let D be another arbitrary, non-trivial event belonging to E. It can be shown quite easily that

$$P(A_k|D) = P(D|A_k)P(A_k)/[\Sigma_{j=1,n} P(D|A_j)P(A_j)]. \qquad (2.1)$$

This formula is known as Bayes theorem. The validity of Eq. 2.1 is beyond question. However, there exists controversy in mathematical circles concerning interpretation of the various quantities which appear in this equation. We shall adopt the Bayesian interpretation. Then, A_k is an hypothesis (e.g., the proposition that certain values for a collection of parameters sought represent the best possible ones) and D represents an experimental data set. Thus, $P(A_k|D)$ is the posterior conditional probability that A_k is true given that the data set D has been acquired. On the other hand, $P(D|A_k)$ is the conditional probability (or likelihood) that the data set D would be generated if A_k were indeed true. Finally, $P(A_k)$ is the unconditional probability that A_k is true, based on our knowledge prior to the experiment which produced data set D. Note that the factor in the denominator serves to normalize the posterior probability, i.e., $P(E|D) = 1$.

Often, we must treat event spaces which are essentially infinite in extent. Then, Eq. 2.1 is not an appropriate form of Bayes theorem to consider. However, we can reformulate this equation by resorting to probability density functions, p. Then, Bayes theorem becomes

$$p(\mathbf{p}|D) = L(D|\mathbf{p})p_a(\mathbf{p})/[\int L(D|\mathbf{p}')p_a(\mathbf{p}')d\mathbf{p}']. \qquad (2.2)$$

Here, $\mathbf{p}$ represents a collection of parameters (in some contexts it signifies a set of specific values for these parameters). These parameters are random variables which represent events. Then, $p(\mathbf{p}|D)$ is the posterior probability density for $\mathbf{p}$, given that data set D was acquired in an experiment, $L(D|\mathbf{p})$ is the probability (likelihood) that parameter set $\mathbf{p}$ would yield data set D, and $p_a(\mathbf{p})$ is the unconditional probability density function for $\mathbf{p}$, based on our knowledge of the situation prior to the experiment which produced data set D. Again, the factor in the denominator serves to normalize the posterior probability. For convenience, we can simply express Eq. 2.2 as

$$p(\mathbf{p}|D) = CL(D|\mathbf{p})p_a(\mathbf{p}), \qquad (2.3)$$

where C represents the required normalization constant which can always be determined by integration.

It should be understood quite clearly that prior knowledge of the situation is characterized by an unconditional probability density function. After an experiment has been performed, and data are available, this probability density function must be revised. Eq. 2.3 gives us a procedure for carrying out this refinement process. The analysis can be repeated whenever a new data set is obtained. Such an iterative approach is consistent with common understanding of how knowledge grows out of experience. The Bayesian method simply offers a computational algorithm for the application of a common sense approach. Probability distributions provide the means for executing this process.

The problem is that we really don't care much about these probability distributions themselves. What we want to know are the parameter "best values" (or "results") to be deduced from such a "learning" exercise. Statistics provides us with a formal prescription for addressing this concern. If we have a probability distribution which describes a set of parameters, then best values for the m parameters p_k are just expected values, namely, those values $\langle p_k \rangle$ defined by

$$\langle p_k \rangle = \int p(\mathbf{p}'|D)p_k d\mathbf{p}' \qquad (k=1,m). \qquad (2.4)$$

Furthermore, the uncertainties (and their correlations) for these best values can be obtained from the covariance matrix V_p whose elements

are obtained from the formula

$$(V_p)_{kq} = \langle (p_k - \langle p_k \rangle)(p_q - \langle p_q \rangle) \rangle$$
$$= \int (p_k - \langle p_k \rangle)(p_q - \langle p_q \rangle) p(\mathbf{p}' | D) d\mathbf{p}' \qquad (k,q=1,m). \qquad (2.5)$$

Although this approach is completely rigorous, the calculations indicated by Eqs. 2.4 and 2.5 can be extremely cumbersome when the number of parameters is large. Therefore, this direct method for finding best values is generally impractical. What can be done about this? We continue to pursue the above line of reasoning. So far, we have said nothing about the likelihood function L which appears in Eqs. 2.2 and 2.3. For convenience, suppose that the data set represented by D consists of a collection of values $\mathbf{y}$. Furthermore, suppose that we also possess a covariance matrix $\mathbf{V}_y$ for these values. Then, suppose that $\mathbf{f}$ represents a collection of functions f_i (i=1,n) such that $f_i(\mathbf{p})$ is the calculated equivalent of y_i, given the parameters $\mathbf{p}$. By applying the principle of maximum entropy proposed by Shannon (e.g., Sha48), Jaynes (e.g., Jay78) showed that L takes the form

$$L(D|\mathbf{p}) \propto \exp\{(-1/2)[\mathbf{y}-\mathbf{f}(\mathbf{p})]^{+} \mathbf{V}_y^{-1} [\mathbf{y}-\mathbf{f}(\mathbf{p})]\}. \qquad (2.6)$$

The symbol " + " indicates matrix transposition and $\mathbf{V}_y^{-1}$ is the inverse matrix corresponding to $\mathbf{V}_y$. Eq. 2.6 indicates that knowledge of a set of data values and their covariance matrix suggests that the likelihood function should take the form of a normal distribution. However, observe that this distribution is <u>not</u> necessarily normal in the parameters $\mathbf{p}$! This issue rests upon the nature of $\mathbf{f}$. It is very important to recognize that Eq. 2.6 is a consequence of the existence of a data set <u>and</u> its covariance matrix. If only the data were available (with no uncertainty information provided), then this formula would not be appropriate.

We combine Eqs. 2.3 and 2.6. This leads to the expression

$$p(\mathbf{p}|D) = C\exp\{(-1/2)[\mathbf{y}-\mathbf{f}(\mathbf{p})]^{+} \mathbf{V}_y^{-1} [\mathbf{y}-\mathbf{f}(\mathbf{p})]\} p_a(\mathbf{p}), \qquad (2.7)$$

where C is the required normalization constant. Assumptions about the prior probability determine whether the analysis will involve the simple least-squares method or the generalized least-squares approach. This matter is discussed in Sections 3 and 4, respectively.

2.2 Linear and Non-linear Models

The nature of the model which relates the data $\mathbf{y}$ to the parameters $\mathbf{p}$ determines whether the least-squares condition is exactly comparable to the rigorous approach based on probability and expectation, as described above, or is an approximation which is actually equivalent to the maximum likelihood method of parameter estimation (e.g., Zeh70). This point is addressed in some detail in both Sections 3 and 4.

First, we discuss what is meant by a "linear" model. Suppose that each of the functions f_i can be expressed in the form

$$f_i(\mathbf{p}) = \Sigma_{k=1,m}\, a_{ik}p_k \qquad (i=1,n). \tag{2.8}$$

If each a_{ik} is a fixed constant which does not depend on $\mathbf{p}$, then the model $\mathbf{f}(\mathbf{p})$ which relates the data $\mathbf{y}$ to the parameters $\mathbf{p}$ is said to be explicitly linear. If $\mathbf{A}$ is defined as the matrix of these constants a_{ik} (i=1,n; k=1,m), then Eq. 2.8 takes the form

$$\mathbf{f}(\mathbf{p}) = \mathbf{A}\mathbf{p}. \tag{2.9}$$

$\mathbf{A}$ is known as the design matrix or sensitivity matrix for the problem under consideration. Since the data $\mathbf{y}$ are assumed to be approximated by $\mathbf{f}(\mathbf{p})$, one can also write the expression

$$\mathbf{y} \approx \mathbf{A}\mathbf{p}. \tag{2.10}$$

The symbol $\approx$ is used to emphasize the fact that a parameterized model <u>always</u> provides only an approximation to real data, owing to both errors in the data and model imperfections.

If no relationship akin to Eq. 2.8 is applicable, then the model is said to be "non-linear". However, if one is given prior values $\mathbf{p}_a$ for $\mathbf{p}$, or can make reasonable initial guesses for their values, then a non-linear model can be linearized by means of first-order Taylor series expansions. This leads to the approximate expression

$$f_i(\mathbf{p}) - f_i(\mathbf{p}_a) = \Sigma_{k=1,m}\, a_{ik}(p_k - p_{ak}) \qquad (i=1,n). \tag{2.11}$$

The equivalent to Eq. 2.11, in matrix form, is

$$\mathbf{f}(\mathbf{p}) - \mathbf{f}(\mathbf{p}_a) = \mathbf{A}(\mathbf{p} - \mathbf{p}_a). \tag{2.12}$$

This expression can be compared with Eq. 2.9, which is applicable in the case of an explicitly linear model. For non-linear problems, the parameters a_{ik} are obtained from the expression

$$a_{ik} = [\partial f_i(\mathbf{p})/\partial p_k]_a \qquad (i=1,n;\ k=1,m). \tag{2.13}$$

So, the elements of $\mathbf{A}$ are partial derivatives of the functions f_i with respect to the parameters p_k, evaluated at $\mathbf{p} = \mathbf{p}_a$. The matrix elements a_{ik} are no longer constants which are independent of the parameters $\mathbf{p}$. For such non-linear problems, it would be more precise to denote the matrix $\mathbf{A}$ in Eq. (2.12) by $\mathbf{A}_a$ to indicate explicitly that its elements depend upon $\mathbf{p}_a$, but we will not take this step. Let $\mathbf{y}_a = \mathbf{f}(\mathbf{p}_a)$. Since the data $\mathbf{y}$ are approximated by $\mathbf{f}(\mathbf{p})$, we can write the expression

$$\mathbf{y} - \mathbf{y}_a \approx \mathbf{A}(\mathbf{p} - \mathbf{p}_a). \tag{2.14}$$

For convenience, let $\mathbf{z} = \mathbf{y} - \mathbf{y}_a$ and $\mathbf{r} = \mathbf{p} - \mathbf{p}_a$. Then, we have

$$\mathbf{z} \approx \mathbf{A}\mathbf{r}, \tag{2.15}$$

which is quite comparable to Eq. 2.10.

In some cases it is possible to linearize a problem without resorting to Taylor series expansions. Usually, this involves transformation of the data and corresponding model to another reference frame where a linear relationship exists. The following example demonstrates the procedure.

Example 2.1: Exponential decay

Suppose that the data value y_i corresponds to measurement of the activity of a radioactive sample at time t_i. The model $f_i(\mathbf{p}) = f(p_1,p_2;t_i) = p_1\exp(-p_2t_i)$ is selected to represent these data. Then, $y_i \approx f(p_1,p_2;t_i)$. This problem is clearly non-linear. So, let us carry out the transformation $w_i = ln\ y_i$, $q_1 = ln\ p_1$ and $q_2 = -p_2$. Then, we obtain the expression $w_i \approx q_1 + q_2t_i$ which is a linear relationship, and we can apply the formalism described above. Keep in mind that such a "trick" cannot be found to deal with most of the non-linear problems encountered in physical research.

In this report, we treat only least-squares problems where the model that relates the data to the parameters is explicitly linear or can be linearized by the procedures described above. Therefore, for non-linear problems to be tractable, it is assumed, implicitly, that the region of significant probability density near $\mathbf{p}_a$, associated with the parameter set $\mathbf{p}$, is sufficiently small so that the first-order Taylor series approximations involved in Eqs. 2.11 - 2.13 are adequate (i.e., the higher-order terms of these expansions can be neglected).

3. SIMPLE LEAST-SQUARES ANALYSIS

3.1 Formalism

Consider Eq. 2.7. An expression for the likelihood function L has been provided but nothing has been stated concerning the prior probability function p_a. As indicated above, the simple least-squares method is appropriate for dealing with problems where there is essentially no prior information on the parameters. Then, p_a must be a non-informative prior probability function, i.e., it is essentially a constant independent of $\mathbf{p}$. For simplicity, we assume that $p_a(\mathbf{p}) = 1$. Then, we can write Eq. 2.7 as

$$p(\mathbf{p}|D) = C\exp\{(-1/2)[\mathbf{y}-\mathbf{f}(\mathbf{p})]^{*}\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})]\}. \tag{3.1}$$

It is absolutely essential that the matrix $\mathbf{V}_y$ be positive definite in order to represent physically reasonable uncertainties for data (e.g., GS88). Then, its inverse $\mathbf{V}_y^{-1}$ will also be positive definite and, as a consequence of the definition of positive definiteness, it follows directly that the quantity $[\mathbf{y}-\mathbf{f}(\mathbf{p})]^{*}\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})]$ is always a positive scalar. The principle of maximum likelihood (e.g., Zeh70) states that the best choice for the parameter set $\mathbf{p}$ is the one which maximizes the posterior probability. Given the nature of Eq. 3.1, this translates into the requirement

$$[\mathbf{y}-\mathbf{f}(\mathbf{p})]^{*}\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})] = \text{minimum}, \tag{3.2}$$

which is a manifestation of the least-squares condition, expressed in matrix notation. Suppose that the model which relates the data $\mathbf{y}$ to parameters $\mathbf{p}$ is either explicitly linear (Eq. 2.9) or has been linearized (Eq. 2.14 or 2.15). Then, we can write Eq. 3.1 as either

$$(\mathbf{y}-\mathbf{A}\mathbf{p})^{*}\mathbf{V}_y^{-1}(\mathbf{y}-\mathbf{A}\mathbf{p}) = \text{minimum} \tag{3.3}$$

or

$$(\mathbf{z}-\mathbf{A}\mathbf{r})^{*}\mathbf{V}_y^{-1}(\mathbf{z}-\mathbf{A}\mathbf{r}) = \text{minimum}. \tag{3.4}$$

Since it does not matter for the formalism whether we choose to use $\mathbf{y}$ and $\mathbf{p}$ or $\mathbf{z}$ and $\mathbf{r}$, no generality is lost by considering the former in addressing this mathematical problem. Therefore, we concentrate on finding a solution that satisfies Eq. 3.3. The details are discussed in Smi91. The results are summarized by following equations:

$$\mathbf{p} = \mathbf{V}_p\mathbf{A}^{*}\mathbf{V}_y^{-1}\mathbf{y}, \tag{3.5}$$

$$\mathbf{V}_p = (\mathbf{A}^{*}\mathbf{V}_y^{-1}\mathbf{A})^{-1}, \tag{3.6}$$

$$\chi^2 = (\mathbf{y}-\mathbf{A}\mathbf{p})^{*}\mathbf{V}_y^{-1}(\mathbf{y}-\mathbf{A}\mathbf{p}). \tag{3.7}$$

It is to these equations that reference must be made in all numerical calculations involving the simple least squares method. The matrix $\mathbf{V_p}$ serves as the covariance matrix for the solution parameters **p**. It provides their uncertainties and correlations. The scalar χ^2 is to be evaluated using solution values for parameters **p** (Eq. 3.5). It forms the basis for a chi-square test (e.g., see Smi91). In particular, the number of degrees of freedom f for the least-squares problem is given by the formula $f = n-m$. Obviously, the number of input data values n must exceed the number of parameters m to be determined or the method will fail. It is assumed that $(\chi^2)_{norm} = \chi^2/f$ follows a chi-square distribution with one degree of freedom. This is strictly true only if the data set **y** is normally distributed. However, for large data sets and modest numbers of parameters (i.e., if f is considerably larger than unity), these problems can be treated as asymptotically normal. If it happens that $(\chi^2)_{norm}$ is significantly larger than unity, then either the data are discrepant or the model used to represent these data is flawed. The origins of any data discrepancies need to be sought, and consideration should also be given to the validity of the model (e.g., see Smi91). If these investigations fail to identify the problem, an approach of last resort is to multiply the errors of all the solution parameters by the factor $(\chi^2/f)^{1/2}$. This is actually equivalent to multiplying all the errors for the input data by the same factor.

Another point to note is that so long as the model which relates the data **y** to the parameters **p** is linear, or has been linearized as discussed in Section 2.2, the solution provided by Eqs. 3.5 - 3.7 is entirely equivalent to what would have been obtained by the calculation of expected values directly from the posterior probability distribution, i.e., from Eqs. 2.4 and 2.5. This happens because the posterior probability distribution is then normal in the parameters **p**, and is therefore completely symmetric about the solution for **p**. This very important result conveniently ties together certain diverse aspects of the theory.

If we are required to treat a non-linear problem which has been linearized, it is easy to derive the solution **p** from **r**, i.e., $\mathbf{p} = \mathbf{p_a} + \mathbf{r}$. What is the corresponding relationship between $\mathbf{V_p}$ and $\mathbf{V_r}$? This question can be addressed by using the law of error propagation. This states that

$$\mathbf{V_p} = \mathbf{T}^*\mathbf{V_r}\mathbf{T}, \tag{3.8}$$

where **T** is the matrix involved in a transformation from variable set **r** to **p** (e.g., see Smi91). The elements of **T** are given by the expression

$$(\mathbf{T})_{kq} = \partial p_k/\partial r_q = \delta_{kq} \qquad (k,q=1,m), \tag{3.9}$$

since $\mathbf{p}_a$ is treated merely as a convenient constant for linearization in simple least-squares analysis. Therefore,

$$\mathbf{V}_p = \mathbf{V}_r \tag{3.10}$$

In order to handle non-linear problems, it is assumed that one is able to identify a good starting point for linearization, i.e., to select a set of parameter values $\mathbf{p}_a$ close enough to the final best set $\mathbf{p}$ which is consistent with the data provided. This can be a challenge if the parameter space involves several local least-squares minima, where the investigator may become inadvertently trapped in his analysis. If one begins near to any minimum of the chi-square surface, local or otherwise, the procedure described above will tend to converge to a solution at that closest minimum, in the neighborhood of $\mathbf{p}_a$. This is a generic problem of all non-linear fitting exercises. One possible remedy for this problem is to eventually make random jumps in parameter space to new starting points $\mathbf{p}_a'$. One thereby explores other possible minima, in order, hopefully, to eventually locate the "deepest" minimum which can then be assumed to correspond to the best solution. Another possibility is to introduce constraints which make it more likely that a reasonable solution will be obtained. As indicated previously, we have chosen to avoid consideration of constrained least-squares problems in this report. In my opinion, prior experience and intuition will generally serve an investigator well in selecting an initial parameter set $\mathbf{p}_a$ which is appropriate for application of this method. Of course, if this is possible, then it may also be feasible to provide reasonable estimates of the uncertainties for this prior set, i.e., to introduce a covariance matrix $\mathbf{V}_a$. Whenever this is actually the case, one probably ought to employ the generalized least-square method in the first place, and avoid the simple least-squares formulation altogether.

3.2 Numerical Procedures (LSMOD)

Code LSMOD offers a "bare-bones", simple least-squares computational tool. Figure 1 is a block diagram which shows the basic components of this code. Two versions of FORTRAN statements for this code plus the input and output for a test problem appear in Appendix A. The input must be sufficiently comprehensive to carry out the calculations indicated by Eqs. 3.5 to 3.7. If the problem has been linearized, then $\mathbf{z}$ replaces $\mathbf{y}$ and $\mathbf{r}$ replaces $\mathbf{p}$ in the input, as discussed in Section 3.1. There is no provision in LSMOD for preparation of this input from more-basic, problem-specific information. For example, the elements of $\mathbf{A}$ must be introduced explicitly. In some problems, these elements can be derived from specific functions. Thus, if the fitting model is a polynomial, these elements will originate from terms of that polynomial. However, all such calculations associated with preparation of the input must be done externally by the user of LSMOD. The same general characteristics hold for the output. A powerful feature of advanced least-squares

methods is that they offer the possibility to calculate uncertainties and their correlations for various quantities that can be derived from the fitted parameters. This is accomplished by employing the law of error propagation (see the discussion leading to Eq. 3.8 and 3.9). The user of LSMOD must take the direct output of this code and employ it externally to carry out such analyses. LSMOD could be tailored for special problems by altering both the input and output sections of the code, while leaving the computational portions largely untouched. This issue is discussed further in Section 5.

Finally, there is no provision in LSMOD (as presently written) to carry out iterations of the least-squares procedure. However, alterations could be made to this code that would allow this to be accomplished with insignificant modifications to the computational segment. This is an important consideration since the choice of prior parameters may be somewhat arbitrary in simple least-squares problems, and iteration might well be advisable in order to obtain reasonable solutions (see Section 5.2).

Key to Code Parameters Found in the Input and Output of LSMOD:

N $\Leftrightarrow$ n

M $\Leftrightarrow$ m

Y(I) $\Leftrightarrow$ y_i

EY(I) $\Leftrightarrow$ $(V_y)_{ii}^{1/2}$

VY(I,J) $\Leftrightarrow$ $(V_y)_{ij}$

CY(I,J) $\Leftrightarrow$ $(C_y)_{ij} = (V_y)_{ij}/[(V_y)_{ii}^{1/2}(V_y)_{jj}^{1/2}]$

A(I,J) $\Leftrightarrow$ $(A)_{ij} = a_{ij}$

P(I) $\Leftrightarrow$ p_i

EP(I) $\Leftrightarrow$ $(V_p)_{ii}^{1/2}$

VP(I,J) $\Leftrightarrow$ $(V_p)_{ij}$

CP(I,J) $\Leftrightarrow$ $(C_p)_{ij} = (V_p)_{ij}/[(V_p)_{ii}^{1/2}(V_p)_{jj}^{1/2}]$

CHI2 $\Leftrightarrow$ χ^2

CHI2NM $\Leftrightarrow$ $(\chi^2)_{norm} = \chi^2/f = \chi^2/(n-m)$

Input Formats for LSMOD:

Record 1.1: N,M (2I5)

Record 2.1 + ...

(Y(I),I=1,N) (6E12.6)

Record 3.1 + ...

```
VY(1,1)
VY(2,1),VY(2,2)
VY(3,1),VY(3,2),VY(3,3)                                   (6E12.6)
   ...
VY(N,1),VY(N,2), ... ,VY(N,N)
```

Record 4.1 + ...

```
A(1,1),A(1,2), ... ,A(1,M)
A(2,1),A(2,2), ... ,A(2,M)                                (6E12.6)
   ...
A(N,1),A(N,2), ... ,A(N,M)
```

Output Formats for LSMOD:

The output is well-labelled and self-explanatory.

Operation of Code LSMOD:

The input and output operations involve files, usually stored on a computer hard disk. The input is assigned to Unit 4 and the output is assigned to Unit 5. The output file to which the results are written must be a pre-existing (STATUS='OLD ') file. The I/O file identifiers are requested from the keyboard upon execution of the code.

3.3 Examples

Examples of least-squares calculations are available in a number of other reports (e.g., Man81 and Smi82). Here we shall consider three examples which should help in understanding the application of simple least-squares analysis to physical problems. Each of these examples has been analyzed using both single-precision (REAL*4 ⇒ 32 bits) and double-precision (REAL*8 ⇒ 64 bits) arithmetic on an IBM-compatible personal computer. The differences observed in the numerical results are negligible in these particular cases, but this will not be be true as a general rule. To be on the safe side, one should probably choose to always use double-precision arithmetic in such calculations, so long as there is no limitation of computer memory. However, in the present report only the single-precision results are shown.

Example 3.1: Averaging of unweighted data

Let us suppose that we have 10 equally weighted values and wish to average them. Of course we know how to do this without resorting to any least-squares methods. We can use a hand calculator and write down the answer. Nevertheless, it is beneficial to apply the methods of this report in order to gain some insight for a problem whose answer appears to be obvious. Here is the data set to consider:

i	y_i
1	10.48
2	11.02
3	9.97
4	10.31
5	10.79
6	11.20
7	10.55
8	11.10
9	9.92
10	10.63

The simple average of these values is 10.597. If we use the statistical option available in a scientific calculator, it will give us two values for the standard deviation, depending upon whether the population number is taken to be n or n-1. These values are $\sigma = [\Sigma(y_i-p)^2/n]^{1/2} = 0.422328$ and $s = [\Sigma(y_i-p)^2/(n-1)]^{1/2} = 0.445173$, respectively. Next, let us apply the present method. We are seeking one parameter, p (m=1). There are n=10 data values. Each data value is an approximation to this parameter, so our model states that $y_i \approx p$ (i=1,10). Therefore, the design matrix **A** has elements $a_{i1} = 1$ (i=1,10). The method requires a covariance matrix as input. This causes a problem because no error information is available. However, we do know that each data value is to be weighted equally. We can accomplish this by introducing a dummy covariance matrix $\mathbf{V}_y$ in which the absolute errors are unity and are uncorrelated. Thus $(\mathbf{V}_y)_{ij} = \delta_{ij}$. This completes preparation of the input for LSMOD.

The input file for LSMOD takes the following form:

```
   10    1
10.48        11.02        9.97         10.31        10.79        11.20
10.55        11.10        9.92         10.63
1.0
0.0          1.0
0.0          0.0          1.0
0.0          0.0          0.0          1.0
0.0          0.0          0.0          0.0          1.0
0.0          0.0          0.0          0.0          0.0          1.0
0.0          0.0          0.0          0.0          0.0          0.0
```

```
1.0
0.0          0.0          0.0          0.0          0.0          0.0
0.0          1.0
0.0          0.0          0.0          0.0          0.0          0.0
0.0          0.0          1.0
0.0          0.0          0.0          0.0          0.0          0.0
0.0          0.0          0.0          1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
```

The output obtained by running LSMOD is as follows:

```
 Y
 .104800E+02 .110200E+02 .997000E+01 .103100E+02 .107900E+02
 .112000E+02 .105500E+02 .111000E+02 .992000E+01 .106300E+02
 EY
 .100000E+01 .100000E+01 .100000E+01 .100000E+01 .100000E+01
 .100000E+01 .100000E+01 .100000E+01 .100000E+01 .100000E+01
 VY
 .100000E+01
 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 CY
 .100000E+01
 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .100000E+01
```

```
.000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
.000000E+00 .100000E+01
.000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
.000000E+00 .000000E+00 .100000E+01
.000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
.000000E+00 .000000E+00 .000000E+00 .100000E+01
.000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
.000000E+00 .000000E+00 .000000E+00 .000000E+00 .100000E+01
A
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
.100000E+01
P
.105970E+02
EP
.316228E+00
VP
.100000E+00
CP
.100000E+01
CHI2,CHI2NM
.178361E+01 .198179E+00
```

Clearly we get the same answer for p (the unweighted average) of this data set as was obtained using a very simple method. It would appear that the rest of the output is without significance since our choice for V_y was quite arbitrary. It is easy to show that χ^2 is simply $\Sigma(y_i-p)^2$. Therefore, if we divide CHI2 by n=10 and take the square root we do find that it equals 0.422328 which is the same as σ obtained with a calculator. Analogously, if we divide CHI2 by n-1=9 and take the square root we obtain 0.445175 which is the same as s obtained with a calculator. Note that this latter value is also equal to the square root of CHI2NM as it should be. How should we interpret EP as provided by LSMOD? In this calculation we assume that the absolute error of each data value is unity. According to the law of large numbers (e.g., see Smi91), the error in an average ought to be less than the error in any single value of the collection being averaged. In particular, the error in a simple average of a set of values each having the same absolute error should equal that fixed error divided by the square root of n. In the present case, this translates to an error value of 1/3.1622777 = 0.3162277, which is exactly what LSMOD yields for EP. EP clearly differs from both σ and s provided above. What do the data tell us that the true error for the derived result p should be? Since the value obtained for CHI2NM is significantly less than unity, we are led to the conclusion that the

error estimate of unity for each value y_i was probably too large. That is, the scatter in these values is, on the average, significantly smaller than unity. If we multiply each input error value by the square root of CHI2NM, i.e., if we assume that $(V_y)_{ij} = (0.198179)\delta_{ij}$, then LSMOD will generate a smaller value for EP. Without exhibiting all the details, the values obtained by carrying out this exercise are EP = 0.140776 and CHI2NM = 1.000000. In other words, if we reduce the errors in the y_i by the amount indicated we obtain a result which is consistent with the actual scatter of the given data. This has been done without compromising the original mandate that all input data points be equally weighted.

In many ways, the least-squares method, as embodied in code LSMOD, is too cumbersome for dealing with this simple problem. Of course, it does generate the same values for p, s and σ as the statistical option of a calculator normally provides. In addition, it yields information about consistency of the data and incorporates the law of large numbers in deriving an error estimate for the average based on the given input errors. We conclude that the advanced least-squares algorithm incorporated in LSMOD offers the user with opportunities for acquiring considerable insight into the nature of the data points being analyzed and the quality of fit provided by the particular model he has selected to represent them.

Example 3.2: Evaluation of cross sections

Let us suppose that our data set consists of six (n=6) neutron cross section values as follows:

i	Neutron Energy (MeV)	Cross Section y_i (millibarn)	Error in y_i (millibarn)
1	7.25	23.6	1.5
2	7.25	25.1	1.7
3	7.25	24.8	1.6
4	7.25	23.9	1.5
5	10.87	198.1	8.9
6	10.87	189.5	9.2

We also suppose that the errors are 50% correlated for all points measured at the same energy and 20% correlated otherwise. Since there are two distinct energies, we seek two distinct cross sections (m=2) for our evaluation, p_1 at 7.25 MeV and p_2 at 10.87 MeV. This problem is definitely beyond the scope of most hand calculators, although it could probably be treated by those units having the capacity to be programmed. The input data are cross sections and the solution parameters are also cross sections. Therefore, the design matrix A consists of the following elements: $a_{i1} = 1$ (i=1,4), $a_{i1} = 0$ (i=5,6), $a_{i2} = 0$ (i=1,4) and $a_{i2} = 1$ (i=5,6). The input required for LSMOD can be derived from this information.

The input file for LSMOD takes the following form:

```
    6    2
23.6          25.1          24.8          23.9          198.1         189.5
2.25
1.275         2.89
1.2           1.36          2.56
1.125         1.275         1.2           2.25
2.67          3.026         2.848         2.67          79.21
2.76          3.128         2.944         2.76          40.94         84.64
1.0           0.0
1.0           0.0
1.0           0.0
1.0           0.0
0.0           1.0
0.0           1.0
```

The output obtained by running LSMOD is as follows:

```
 Y
 .236000E+02 .251000E+02 .248000E+02 .239000E+02 .198100E+03
 .189500E+03
 EY
 .150000E+01 .170000E+01 .160000E+01 .150000E+01 .890000E+01
 .920000E+01
 VY
 .225000E+01
 .127500E+01 .289000E+01
 .120000E+01 .136000E+01 .256000E+01
 .112500E+01 .127500E+01 .120000E+01 .225000E+01
 .267000E+01 .302600E+01 .284800E+01 .267000E+01 .792100E+02
 .276000E+01 .312800E+01 .294400E+01 .276000E+01 .409400E+02
 .846400E+02
 CY
 .100000E+01
 .500000E+00 .100000E+01
 .500000E+00 .500000E+00 .100000E+01
 .500000E+00 .500000E+00 .500000E+00 .100000E+01
 .200000E+00 .200000E+00 .200000E+00 .200000E+00 .100000E+01
 .200000E+00 .200000E+00 .200000E+00 .200000E+00 .500000E+00
 .100000E+01
 A
 .100000E+01 .000000E+00
 .100000E+01 .000000E+00
 .100000E+01 .000000E+00
 .100000E+01 .000000E+00
 .000000E+00 .100000E+01
 .000000E+00 .100000E+01
```

```
P
.241816E+02 .193813E+03
EP
.123362E+01 .782780E+01
VP
.152181E+01
.280291E+01 .612744E+02
CP
.100000E+01
.290261E+00 .100000E+01
CHI2,CHI2NM
.210839E+01 .527099E+00
```

This is a very reasonable solution. Both evaluated cross-section values are obviously consistent with the input data. Based on $\chi^2/f = 0.527099$, it is clear that the input data are consistent and the model assumed for the evaluation is a valid one. The solution uncertainties are somewhat smaller than those for the input data and these errors are modestly correlated (about 29%). Recall that in the preceding example we argued, on the grounds that χ^2/f was significantly smaller than unity, that the error for the solution ought to be reduced. In that situation we had no explicit knowledge of the data uncertainties, and based our analysis entirely on the proposition that the errors should be equal so that the data values are equally weighted. Is it reasonable to do the same here since $\chi^2/f < 1$, i.e., should the data errors be reduced? The answer, generally, is that it is not the right thing to do. First, we note that while χ^2/f is indeed less than unity, it is not so by a very great amount. Furthermore, in realistic problems such as this one we have to assume that the error estimates provided are quite reasonable. The conservative approach is to let matters stand as they are and not decrease the errors.

Example 3.3: Fit of an harmonic function to data

In this example we fit an harmonic function to a set of six (n=6) data points y_i (i=1,6). The model is given by $f_i(\mathbf{p}) = p_1\cos(p_2\theta_i)$, i.e., there are two parameters (m=2) to be determined. These are p_1 (an amplitude parameter) and p_2 (an angular-scaling parameter). The model is linear in p_1 but not in p_2. Therefore, we need to linearize the model. The quantity θ_i is the angle to which the data point y_i corresponds. The input data are as follows:

i	θ_i (degrees)	y_i	Error in y_i
1	5	24.1	1.7
2	15	22.4	1.4
3	40	7.65	1.0
4	60	-7.8	0.9
5	75	-17.1	1.8
6	85	-22.0	1.6

We assume that the errors in these data are uncorrelated. The linearization process is carried out by assuming that $p_{a1} = 25.0$ and $p_{a2} = 1.8$ are values close to the solution we seek. Thus, the formalism embodied in code LSMOD is used to obtain a solution for **r** given **z** as input (in place of **y**). Then, **p** is computed from the formula $\mathbf{p} = \mathbf{p_a}+\mathbf{r}$, as described in Section 3.1. In order to calculate **z**, we need to evaluate $y_a = f(p_a)$. These calculations lead to the following table:

i	y_{ai}	z_i
1	24.692209	-0.592209
2	22.275163	0.124837
3	7.7254248	-0.0754248
4	-7.7254248	-0.0745752
5	-17.67767	0.57767
6	-22.275163	0.275163

The elements of the design matrix **A** can be calculated from the formulas $a_{i1} = \cos(p_{a2}\theta_i)$ and $a_{i2} = -p_{a1}\theta_i \sin(p_{a2}\theta_i)$, (i=1,6). These are given in the following table:

i	a_{i1}	a_{i2}
1	0.9876883	-0.341287
2	0.8910065	-2.9713609
3	0.3090169	-16.599068
4	-0.3090169	-24.898601
5	-0.7071067	-23.140015
6	-0.8910065	-16.837712

This provides all that is needed to analyze this problem using LSMOD.

The input file for LSMOD takes the following form:

```
    6    2
-0.592209   0.124837    -0.0754248  -0.0745752  0.57767     0.275163
2.89
0.0         1.96
0.0         0.0         1.0
0.0         0.0         0.0         0.81
0.0         0.0         0.0         0.0         3.24
0.0         0.0         0.0         0.0         0.0         2.56
0.9876883   -0.341287
0.8910065   -2.9713609
0.3090169   -16.599068
-0.3090169  -24.898601
-0.7071067  -23.140015
-0.8910065  -16.837712
```

The output obtained by running LSMOD is as follows:

Note: In the output, "Y" is to be interpreted as "Z" and "P" is equivalent to "R" in the context of the formalism.

```
 Y
-.592209E+00 .124837E+00-.754248E-01-.745752E-01 .577670E+00
 .275163E+00
 EY
 .170000E+01 .140000E+01 .100000E+01 .900000E+00 .180000E+01
 .160000E+01
 VY
 .289000E+01
 .000000E+00 .196000E+01
 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .810000E+00
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .324000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .256000E+01
 CY
 .100000E+01
 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .100000E+01
 .000000E+00 .000000E+00 .000000E+00 .000000E+00 .000000E+00
 .100000E+01
 A
 .987688E+00-.341287E+00
 .891007E+00-.297136E+01
 .309017E+00-.165991E+02
-.309017E+00-.248986E+02
-.707107E+00-.231400E+02
-.891007E+00-.168377E+02
 P
-.263394E+00 .853362E-03
 EP
 .885254E+00 .290236E-01
 VP
 .783675E+00
-.819147E-02 .842371E-03
 CP
 .100000E+01
-.318818E+00 .100000E+01
 CHI2,CHI2NM
 .181133E+00 .452834E-01
```

The adjustments to the parameters p_a to obtain **p** are quite small and the quality of the fit to the data is very good as judged by the value of χ^2/f obtained. In fact, the errors assigned to the data points seem to be too large considering the actual deviations from the fitted model. We make no decision on this point for the exercise.

4. GENERALIZED LEAST-SQUARES ANALYSIS

4.1 Formalism

Consider Eq. 2.7. An expression for the likelihood function L has been provided but nothing has been stated concerning the prior probability function p_a. The generalized least-squares method is appropriate for dealing with problems where there exists prior information on the parameters $\mathbf{p}$. In particular, one is given a set $\mathbf{p}_a$ and a corresponding covariance matrix $\mathbf{V}_a$. Once again, we apply the principle of maximum entropy, just as was done in the case of the likelihood function. Then, we have that

$$p_a(\mathbf{p}) \propto \exp[(-1/2)(\mathbf{p}-\mathbf{p}_a)^+\mathbf{V}_a^{-1}(\mathbf{p}-\mathbf{p}_a)]. \tag{4.1}$$

Then, we can write Eq. 2.7 as

$$\begin{aligned} p(\mathbf{p}|D) = C\exp\{&(-1/2)[\mathbf{y}-\mathbf{f}(\mathbf{p})]^+\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})] \\ &+ (-1/2)(\mathbf{p}-\mathbf{p}_a)^+\mathbf{V}_a^{-1}(\mathbf{p}-\mathbf{p}_a)\}. \end{aligned} \tag{4.2}$$

It is tacitly assumed that the new data are independent of the information that led to the prior knowledge of the parameters. Otherwise, Eq. 4.2 is not correct. It was shown in Section 2.1 that since $\mathbf{V}_y$ should be considered as positive definite, $\mathbf{V}_y^{-1}$ must also be positive definite. Consequently, the quantity $[\mathbf{y}-\mathbf{f}(\mathbf{p})]^+\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})]$ is always a positive scalar. Similar arguments lead us to suggest that $\mathbf{V}_a$ and $\mathbf{V}_a^{-1}$ are also positive definite. Thus, $(\mathbf{p}-\mathbf{p}_a)^+\mathbf{V}_a^{-1}(\mathbf{p}-\mathbf{p}_a)$ is also a positive scalar. The principle of maximum likelihood (e.g., Zeh70) states that the best choice for the parameter set $\mathbf{p}$ is the one which maximizes the posterior probability. Given the nature of Eq. 4.2, this translates into the requirement

$$[\mathbf{y}-\mathbf{f}(\mathbf{p})]^+\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{f}(\mathbf{p})] + (\mathbf{p}-\mathbf{p}_a)^+\mathbf{V}_a^{-1}(\mathbf{p}-\mathbf{p}_a) = \text{minimum}, \tag{4.3}$$

which is a manifestation of the generalized least-squares condition, expressed in matrix notation. We will deal with a model that has been linearized (Eq. 2.12). Thus, the elements of $\mathbf{A}$ are computed as described in Section 2.2 and $\mathbf{y}_a = \mathbf{f}(\mathbf{p}_a)$. Then, we can write Eq. 4.3 as

$$[\mathbf{y}-\mathbf{y}_a-\mathbf{A}(\mathbf{p}-\mathbf{p}_a)]^+\mathbf{V}_y^{-1}[\mathbf{y}-\mathbf{y}_a-\mathbf{A}(\mathbf{p}-\mathbf{p}_a)] + (\mathbf{p}-\mathbf{p}_a)^+\mathbf{V}_a^{-1}(\mathbf{p}-\mathbf{p}_a) = \text{minimum}. \tag{4.4}$$

The solution is given by the following four formulas:

$$\mathbf{p} = \mathbf{p}_a + \mathbf{V}_a\mathbf{A}^+(\mathbf{Q}+\mathbf{V}_y)^{-1}(\mathbf{y}-\mathbf{y}_a), \tag{4.5}$$

$$\mathbf{Q} = \mathbf{A}\mathbf{V}_a\mathbf{A}^+, \tag{4.6}$$

$$\mathbf{V}_p = \mathbf{V}_a - \mathbf{V}_a\mathbf{A}^+(\mathbf{Q}+\mathbf{V}_y)^{-1}\mathbf{A}\mathbf{V}_a, \tag{4.7}$$

$$(\chi^2)_{min} = (\mathbf{y}-\mathbf{y}_a)^+(\mathbf{Q}+\mathbf{V}_y)^{-1}(\mathbf{y}-\mathbf{y}_a). \tag{4.8}$$

It is to these equations that reference is made in all numerical calculations involving the generalized least squares method. The matrix $\mathbf{V}_p$ serves as the covariance matrix for the solution parameters **p**. It provides their uncertainties and correlations. It is interesting to note that the expression for the chi-square parameter which can be employed for chi-square testing is actually independent of the solution! It depends only on the experimental data and prior values for the parameters (used in conjunction with the model to calculate $\mathbf{y}_a$). This is a useful consequence of the theory for the treatment of large problems because it enables the input information to be tested for consistency before pursuing the analysis any further. The number of degrees of freedom f in generalized least-squares analysis just equals the number of input data points n. There is no constraint to the effect that the number of data points n must exceed the number of parameters m to be determined, as was the case for the simple least-squares method. One of the strengths of this formalism is that it can be applied even when only a single piece of new information is introduced, regardless of the number of parameters involved. We assume that $(\chi^2)_{norm} = \chi^2/f$ follows a chi-square distribution with one degrees of freedom. This is strictly true only if the data set **y** is normally distributed. However, for large data sets and modest numbers of parameters (i.e., if f is considerably larger than unity), such problems can be treated as asymptotically normal. One will probably not be led too far astray in interpreting the results of least-squares fitting in the context of a chi-square test even when f is of modest size. It is customary to do so. Let us suppose that $(\chi^2)_{norm}$ is significantly larger than unity. Then, it is the case that either the data are discrepant, the model used to represent the data is flawed, or the prior knowledge of the parameters is defective. The origins of any discrepancies in the data or prior parameters need to be sought and consideration should be given to the validity of the model (e.g., see Smi91). If these investigations fail to identify the problem, one possible approach is to increase the errors in the solution parameters by the factor $(\chi^2/f)^{1/2}$. This is equivalent to increasing all the errors in the input data by the same factor.

Another notable characteristic of the theory is that so long as the model which relates the data **y** to the parameters **p** is linear, or has been linearized as discussed in Section 2.2, the solution provided by Eqs. 4.5 - 4.8 is completely equivalent to what would have been obtained by calculation of expected values directly from the posterior probability distribution, i.e., from Eqs. 2.4 and 2.5. This happens because the posterior probability distribution is then normal in the parameters **p**. This is a very important result which conveniently ties together several diverse aspects of the theory.

Finally, although it was mentioned in Section 1 that a least-squares formalism with explicit constraints would not be considered here, it is possible to introduce and preserve certain *de facto* constraints within the generalized least-squares formalism by means of a clever selection of the elements in the prior-parameter correlation matrix, $\mathbf{V}_a$. In particular, one can prevent certain

parameters from experiencing significant adjustment by simply assigning them small prior errors. Strong prior correlations (positive or negative) also serve to constrain the adjustment process (e.g., Smi87a).

4.2 Numerical Procedures (GLSMOD)

Code GLSMOD offers a "bare-bones" generalized least-squares computational tool. Figure 2 is a block diagram which shows the basic components of this code. Two versions of FORTRAN statements for this code plus the input and output for a test problem appear in Appendix B. The input must be provided as required to carry out the calculations indicated by Eqs. 4.5 to 4.8. There is no need to make a distinction between linear and non-linear problems, other than to consider the impact on generation of the design matrix **A**. In generalized least-squares analysis, one always deals with prior estimates of the parameters as the starting point. There is no provision in GLSMOD for preparation of the input from more basic, problem-specific considerations. For example, the elements of **A** must be introduced explicitly. In some problems, these elements can be calculated from specific functions. Thus, if the fitting model is a polynomial, these elements will originate from terms of that polynomial. However, all such calculations associated with preparation of the input must be done externally by the user of GLSMOD. The same general characteristics hold for the output. A powerful feature of advanced least-squares methods is that they offer possibilities to calculate uncertainties and their correlations for various quantities that can be derived from the fitted parameters. This is accomplished by employing the law of error propagation (see the discussion leading to Eqs. 3.8 and 3.9). The user of GLSMOD must take the direct output of this code and employ it externally to carry out such analyses. GLSMOD could be tailored for special problems by altering both the input and output sections of the code, while leaving the computational portion largely untouched. This issue is discussed further in Section 5.

While iteration may play a useful role in simple least-squares analysis, it does not make any sense from a fundamental point of view in applications of generalized least squares. Why is this true? In generalized least-squares problems, one possesses prior knowledge of all the parameters, including their uncertainties. One must do a reasonable job of obtaining this prior information and then accepting its viability. To tamper with the process via iteration would violate the basic concepts upon which the theory rests. However, there is an exception to this edict. One may wish to use an approximate solution in order to better evaluate the errors in the input experimental data. One should then iterate to convergence to get a reasonable final solution that avoids the problems associated with Peelle's Pertinent Puzzle (e.g., see CS91). See Sections 5.1 and 5.2 for further discussion of this topic.

Key to Code Parameters Found in the Input and Output of GLSMOD:

N $\Leftrightarrow$ n

M $\Leftrightarrow$ m

Y(I) $\Leftrightarrow$ y_i

EY(I) $\Leftrightarrow$ $(V_y)_{ii}^{1/2}$

VY(I,J) $\Leftrightarrow$ $(V_y)_{ij}$

CY(I,J) $\Leftrightarrow$ $(C_y)_{ij} = (V_y)_{ij}/[(V_y)_{ii}^{1/2}(V_y)_{jj}^{1/2}]$

YA(I) $\Leftrightarrow$ y_{ai}

A(I,J) $\Leftrightarrow$ $(A)_{ij} = a_{ij}$

P(I) $\Leftrightarrow$ p_i

EP(I) $\Leftrightarrow$ $(V_p)_{ii}^{1/2}$

VP(I,J) $\Leftrightarrow$ $(V_p)_{ij}$

CP(I,J) $\Leftrightarrow$ $(C_p)_{ij} = (V_p)_{ij}/[(V_p)_{ii}^{1/2}(V_p)_{jj}^{1/2}]$

PA(I) $\Leftrightarrow$ p_{ai}

EA(I) $\Leftrightarrow$ $(V_a)_{ii}^{1/2}$

VA(I,J) $\Leftrightarrow$ $(V_a)_{ij}$

CA(I,J) $\Leftrightarrow$ $(C_a)_{ij} = (V_a)_{ij}/[(V_a)_{ii}^{1/2}(V_a)_{jj}^{1/2}]$

CHI2 $\Leftrightarrow$ χ^2

CHI2NM $\Leftrightarrow$ $(\chi^2)_{norm} = \chi^2/f = \chi^2/n$

Input Formats for GLSMOD:

Record 1.1: N,M (2I5)

Record 2.1 + ...

(Y(I),I=1,N) (6E12.6)

Record 3.1 + ...

(YA(I),I=1,N) (6E12.6)

Record 4.1 + ...

VY(1,1)
VY(2,1),VY(2,2)
VY(3,1),VY(3,2),VY(3,3) (6E12.6)
...
VY(N,1),VY(N,2), ... ,VY(N,N)

Record 5.1 + ...

(PA(I),I=1,M) (6E12.6)

Record 6.1 + ...

VA(1,1)
VA(2,1),VA(2,2)
VA(3,1),VA(3,2),VA(3,3) (6E12.6)
...
VA(M,1),VA(M,2), ... ,VY(M,M)

Record 7.1 + ...

A(1,1),A(1,2), ... ,A(1,M)
A(2,1),A(2,2), ... ,A(2,M) (6E12.6)
...
A(N,1),A(N,2), ... ,A(N,M)

Output Formats for GLSMOD:

The output is well-labelled and self-explanatory.

Operation of Code GLSMOD:

The input and output operations involve files, usually stored on a computer hard disk. The Input is assigned to Unit 4 and the output is assigned to Unit 5. The output file to which the results are written must be a pre-existing (STATUS='OLD ") file. The I/O file identifiers are requested from the keyboard upon execution of the code.

4.3 Examples

Examples of least-squares calculations are available in a number of other reports (e.g., Man81 and Smi82). Here we shall consider three examples which should help in understanding the application of generalized least-squares analysis to physical problems. Each of these examples has been analyzed using both single-precision (REAL*4 ⇒ 32 bits) and double-precision (REAL*8 ⇒ 64 bits) arithmetic on an IBM-compatible personal computer. The differences observed in the numerical results are negligible in these particular cases, but this

will not be be true as a general rule. To be on the safe side, one should probably choose to always use double-precision arithmetic in such calculations, so long as there is no limitation of computer memory. However, in the present report only the single-precision results are shown.

Example 4.1: Re-evaluation of a 14-MeV cross section

This problem is drawn from Ref. Smi91 (pp. 221-222). The only difference is that here we employ a somewhat different approach to calculate the input data covariance matrix, in order to avoid suffering the consequences of Peelle's Pertinent Puzzle (e.g., see Section 5.1 and Ref. CS91). The prior value for a single 14-MeV cross section (m=1) is available, namely, $p_a = \sigma_a = 1095 \pm 52$ mb ($\pm$ 4.7% error). The cross-section units "mb" signify millibarn. There are two new data values (n=2), namely, $y_1 = \sigma_1 = 1000 \pm 30$ mb ($\pm$ 3.0% error) and $y_2 = \sigma_2 = 1102 \pm 60$ mb ($\pm$ 5.4% error). We shall assume that these data are completely independent of each other, and are also independent of the prior result. Here, σ_a, σ_1 and σ_2 signify quantities which are equivalent to the revised evaluated cross section, $p = \sigma_e$, we seek to determine by this method. Consequently, the design matrix, **A**, is quite simple. It has dimension 2 x 1 and the elements $a_{11} = a_{21} = 1$. The covariance matrix, $\mathbf{V}_a$, for the prior result has dimension 1 x 1 (a scalar), and the value of its single element is $(52)^2 = 2704$. The data covariance matrix, $\mathbf{V}_y$, is 2 x 2. Its off-diagonal elements are $(\mathbf{V}_y)_{12} = (\mathbf{V}_y)_{21} = 0$, since the input data are uncorrelated. For the given input results, one would normally derive the values $(\mathbf{V}_y)_{11} = (30)^2 = 900$ and $(\mathbf{V}_y)_{22} = (60)^2 = 3600$. This is based upon use of the given absolute errors (in cross section units) in the calculation. However, there is an alternative approach, as discussed in Ref. CS91, whereby one employs fractional errors for the data and the same cross section value to calculate absolute errors. These reflect a more consistent weighting of the input data than emerges from use of the conventional approach. An obvious initial choice for this common cross section value is the prior value, $p_a = \sigma_a = 1095$ mb. Consequently, $(\mathbf{V}_y)_{11} = [(0.03)(1095)]^2 = 1079.1225$ and $(\mathbf{V}_y)_{22} = [(0.054)(1095)]^2 = 3496.3569$. This completes preparation of the input for GLSMOD.

The input file takes the form:

```
   2    1
1000.0       1102.0
1095.0       1095.0
1079.1125
0.0          3496.3569
1095.0
2704.0
1.0
1.0
```

The output obtained by running GLSMOD is as follows:

```
 YA
 .109500E+04 .109500E+04
 Y
 .100000E+04 .110200E+04
 EY
 .328498E+02 .591300E+02
 VY
 .107911E+04
 .000000E+00 .349636E+04
 CY
 .100000E+01
 .000000E+00 .100000E+01
 A
 .100000E+01
 .100000E+01
 PA
 .109500E+04
 EA
 .520000E+02
 VA
 .270400E+04
 CA
 .100000E+01
 P
 .104064E+04
 EP
 .251377E+02
 VP
 .631903E+03
 CP
 .100000E+01
 CHI2,CHI2NM
 .370020E+01 .185010E+01
```

This analysis yields 1040.6 ± 25.1 mb for the revised evaluated cross section. Since χ^2/f = 1.85010, there are clearly some inconsistencies concerning the new data and prior value. Since these are modest, we choose to simply enhance the solution error by the factor $(\chi^2/f)^{1/2} = (1.85010)^{1/2} = 1.36018$. Thus, $p = \sigma_e$ = 1040.6 ± 34.2 mb (± 3.2%). This result does not differ significantly from the one given in Ref. Smi91. The consideration of two new data points does lead to a change in the cross section relative to the prior value, and to some reduction of error. The latter was modest, however, because the new data were not very accurate and there are some inconsistencies that impact upon the error-reduction potential of the new information. We could consider another pass at this calculation, using the first-pass solution, 1040.6 mb, to calculate the elements of the covariance matrix, V_y. We shall not do so here in order to avoid cluttering this report with details. Also, the anticipated change in the solution is expected to be small.

Example 4.2: Re-analysis of Example 4.1 with a larger prior error

It is very interesting to see what happens if we assume that the error in the prior cross section $p_a = \sigma_a = 1095$ mb is quite large, say ± 30% (i.e., 328.5 mb). Then, the single element of V_a is equal to $(328.5)^2 = 107912.25$. Qualitatively, we would conclude that the prior value should have little influence on the re-evaluation. Furthermore, since the two new data points are uncorrelated, we might expect that the solution would be approximately be the weighted average of these two values, i.e., 1024.1 mb, with an error of ± 2.6% (i.e., 26.6 mb). This error follows from the expression $2.6 \approx [(1/3^2) + (1/5.4^2)]^{-1/2}$, as discussed in Ref. Smi81. In other words, when the prior information is very uncertain (an uninformative prior), the generalized least-squares problem essentially reduces to a simple least-squares problem. To check this out, we carry out a GLSMOD calculation with revised input.

The input file takes the form:

```
   2    1
1000.0       1102.0
1095.0       1095.0
1079.1125
0.0          3496.3569
1095.0
107912.25
1.0
1.0
```

The output obtained by running GLSMOD is as follows:

```
 YA
 .109500E+04 .109500E+04
 Y
 .100000E+04 .110200E+04
 EY
 .328498E+02 .591300E+02
 VY
 .107911E+04
 .000000E+00 .349636E+04
 CY
 .100000E+01
 .000000E+00 .100000E+01
 A
 .100000E+01
 .100000E+01
 PA
 .109500E+04
 EA
 .328500E+03
```

```
VA
.107912E+06
CA
.100000E+01
P
.102459E+04
EP
.286057E+02
VP
.818284E+03
CP
.100000E+01
CHI2,CHI2NM
.232015E+01 .116008E+01
```

We see that the GLSMOD solution is 1024.6 ± 28.6 mb (± 2.8%). Since $(\chi^2/f)^{1/2} = (1.16008)^{1/2} = 1.07707$, we should apply a slight enhancement factor to the error, thereby obtaining ± 30.8 mb (± 3.0%). This analysis yields results reasonably close to what was obtained from a very simple hand calculation, thereby confirming our expectations for this problem.

Example 4.3: Evaluation of a cross section at two energies

Let us suppose that we are interested in the values for a differential cross section at both 2.2 and 14.7 MeV. Furthermore, we have prior values for these as follows: p_{a1} = 210 mb (± 7%) and p_{a2} = 40 mb (± 8%). The errors are assumed to be 50%-correlated. The covariance matrix V_a can be determined in a straightforward manner from this information.

Next, we assume that we have two new data values as follows: y_1 = 205.6 mb (± 8%), corresponding to a direct measurement of the cross section at 2.2 MeV; y_2 = 0.209 (± 6%), corresponding to measurement of the ratio of the cross section at 14.7 MeV to that at 2.2 MeV. We assume that these data are only 20%-correlated. In order to avoid the Peelle's Pertinent Puzzle problem, as discussed in Example 4.1 and Section 5.1, we determine the elements of the covariance matrix V_y as follows: $(V_y)_{11} = [(0.08)(210)]^2 = 282.24$, $(V_y)_{22} = [(0.06)(40/210)]^2 = 0.0001306$ and $(V_y)_{12} = (V_y)_{21} = [(0.2)(0.08)(210)(0.06)(40/210)]^2 = 0.0384$. In other words, we use the prior values of the parameters and fractional errors in the data to evaluate the data covariance matrix.

Calculation of the design matrix elements, a_{ij} (i,j=1,2), is a bit more complicated. In terms of the parameters p_1 and p_2, they are given by $a_{11} = 1$, $a_{12} = 0$, $a_{21} = -p_2/p_1^2$ and $a_{22} = 1/p_1$. Actually, **A** must be evaluated by using the prior values of these parameters, namely, p_{a1} = 210 mb and p_{a2} = 40 mb.

The input file takes the form:

```
    2    2
205.6          0.209
210.0          0.1904761
282.24
0.0383999      0.0001306
210.0          40.0
216.09
23.52          10.24
1.0            0.0
-0.000907      0.0047619
```

The output obtained by running GLSMOD is as follows:

```
 YA
 .210000E+03 .190476E+00
 Y
 .205600E+03 .209000E+00
 EY
 .168000E+02 .114280E-01
 VY
 .282240E+03
 .383999E-01 .130600E-03
 CY
 .100000E+01
 .200009E+00 .100000E+01
 A
 .100000E+01 .000000E+00
-.907000E-03 .476190E-02
 PA
 .210000E+03 .400000E+02
 EA
 .147000E+02 .320000E+01
 VA
 .216090E+03
 .235200E+02 .102400E+02
 CA
 .100000E+01
 .500000E+00 .100000E+01
 P
 .204600E+03 .414010E+02
 EP
 .104885E+02 .255028E+01
 VP
 .110010E+03
 .190223E+02 .650393E+01
 CP
 .100000E+01
 .711146E+00 .100000E+01
 CHI2,CHI2NM
 .102432E+01 .512158E+00
```

The solution of this problem is $p_1 = 204.6$ (± 5.2%) and $p_2 = 41.4$ (± 6.2%). The solution covariance matrix indicates a correlation of 71% between these values. Since $\chi^2/f < 1$, there is no need to enhance the error. In this example, the data and prior values are mutually consistent and the new information leads to some reduction in the uncertainty of the evaluated result. For interest, this problem was subjected to an additional iteration, employing the solution values from the first pass to re-calculate only the elements of $\mathbf{V}_y$. Nothing else was altered. The result of this analysis was $p_1 = 204.7$ (± 5.1%) and $p_2 = 41.3$ (± 6.2%), with a correlation of 69%. These results differ insignificantly from those obtained from the first pass, demonstrating the point that such iteration is seldom needed in practice. Nevertheless, it is relatively simple to carry through, if only to satisfy curiosity.

Example 4.4: A variation of Example 4.3

In this problem, we start with the same prior information as was available in Example 4.3. The difference here is that we choose to introduce new values for the cross sections at 2.2 and 14.7 MeV, rather than one cross-section and one cross-section ratio. This assumption simplifies the design matrix, $\mathbf{A}$. Its values are now $a_{11} = a_{22} = 1$ and $a_{12} = a_{21} = 0$. The new data are as follows: $y_1 = 205.6$ mb (± 8%) and $y_2 = 42.3$ (± 6%). We again assume 20% correlation between these values. This time, however, we choose (for no particular reason) to compute the elements of $\mathbf{V}_y$ directly from the new data, thus: $(\mathbf{V}_y)_{11} = [(0.08)(205.6)]^2 = 270.5367$, $(\mathbf{V}_y)_{22} = [(0.06)(42.3)]^2 = 6.441444$ and $(\mathbf{V}_y)_{12} = (\mathbf{V}_y)_{21} = [(0.2)(0.08)(205.6)(0.06)(42.3)]^2 = 8.3490048$.

The input file takes the form:

```
   2    2
205.6        42.3
210.0        40.0
270.5367
8.3490048    6.441444
210.0        40.0
216.09
23.52        10.24
1.0          0.0
0.0          1.0
```

The output obtained by running GLSMOD is as follows:

```
 YA
 .210000E+03 .400000E+02
 Y
 .205600E+03 .423000E+02
 EY
 .164480E+02 .253800E+01
```

```
VY
.270537E+03
.834900E+01 .644144E+01
CY
.100000E+01
.200000E+00 .100000E+01
A
.100000E+01 .000000E+00
.000000E+00 .100000E+01
PA
.210000E+03 .400000E+02
EA
.147000E+02 .320000E+01
VA
.216090E+03
.235200E+02 .102400E+02
CA
.100000E+01
.500000E+00 .100000E+01
P
.209708E+03 .413301E+02
EP
.106827E+02 .197923E+01
VP
.114120E+03
.749131E+01 .391734E+01
CP
.100000E+01
.354308E+00 .100000E+01
CHI2,CHI2N
.498765E+00 .249383E+00
```

This analysis yields the solutions $p_1 = 209.7$ (± 5.1%) and $p_2 = 41.3$ (± 4.8%), with a correlation of 35%. What is interesting in comparing this result with that from Example 4.3 is that the solution correlation is much weaker. It appears that ratio data with good accuracy (Example 4.3) introduces strong correlations.

5. RELATED TOPICS

5.1 Generation of Covariance Matrices

Applications of the least-squares procedures described in this report call for the generation of various types of covariance matrices as input to the calculations. For applications of generalized least squares, one requires the matrix V_a for the prior parameters. There is little ambiguity in obtaining this matrix. Either it is given specifically, or it can be generated from the available prior values, their errors and their correlations, as described elsewhere (e.g., Smi81). Provision of the data covariance matrix, V_y, for both simple and generalized least-squares analysis is another matter. The errors in experimental data are often comprised of several distinct components, each with its own special correlation pattern. Generation of the covariance matrix for total errors involves error superposition, as discussed in Ref. Smi87b. These ideas will not be discussed further in this report since the reader can easily refer to the literature for guidance on the combination of partial errors in constructing a covariance matrix.

There is another issue, which has come to be known as Peelle's Pertinent Puzzle, which does need to be considered here. It was mentioned in conjunction with some of the examples presented in Section 4. The topic is explored in much greater detail in Ref. CS91. The basic idea is the following: Suppose that a collection of equivalent data values and their errors are provided, i.e., (y_i, E_{yi}) for i=1,n. Also, there is knowledge of the correlation matrix, C_y. Conventionally, one derives the elements of the covariance matrix, V_y, from the formula $(V_y)_{ij} = E_{yi}E_{yj}(C_y)_{ij}$.

However, it happens that in least-squares analyses a data point y_i tends to be weighted by the corresponding factor E_{yi}^{-2}. Let $f_{yi} = E_{yi}/y_i$ be the fractional error in y_i. When authors quote errors for the experimental values they report, they often give absolute errors (in units of the quantity reported) when, in fact, they probably ought to provide fractional errors as more consistent indicators of the true errors of their experiment. A fractional error is independent of the outcome, i.e., the specific value deduced for the physical quantity in question. For the sake of the present discussion, we suppose that all the fractional errors are the same for a particular set of equivalent data, i.e., that each reported value was considered to be measured to essentially the same degree of precision. In spite of this, it is inevitable that the various y_i will differ from each other, even when they are understood to be entirely equivalent. Then, the weighting factors E_{yi}^{-2} will also differ, so that the lower values of y_i will be more heavily weighted. If the data scatter considerably, and there are significant error correlations, least-squares analyses can often produce results which are intuitively quite unreasonable (one aspect of Peelle's Pertinent Puzzle). The main point from Ref. CS91 is that this problem can be mitigated by

calculating the elements of V_y in a modified fashion, i.e., by using the formula $(V_y)_{ij} = f_i y_i f_j y_j (C_y)_{ij}$. The quantity y is an approximation to the experimental value y which is more consistent with the underlying expectation for this physical quantity, i.e., a "universal" or "true" value. Since we do not know what this true value is at the outset, this introduces a dilemma. In the case of generalized least-squares analysis, it is quite reasonable to employ an equivalent value y_{ai} deduced from the prior parameters of the problem. For simple least-squares problems, there is no obvious choice and one is left to improvise. However, in both types of least-squares problems, it is possible to iterate the solution (see Section 5.2) in order to achieve consistency and avoid, to some extent, the arbitrariness associated with determination of V_y.

5.2 Some Thoughts about Iteration

The subject of iteration has already been mentioned in Sections 4.3 and 5.1. The purpose of this section is simply to present a few basic rules that ought to be followed in applying iterative techniques to least-squares problems.

Rule 1:

For both simple and generalized least-squares problems, it is quite reasonable to iterate any number of times in derivation of the covariance matrix V_y, until convergence is achieved, i.e., until the solution values for the parameters sought are essentially equal to those used in the calculation of V_y. Convergence will normally occur after very few iterations in most practical situations.

Rule 2:

In simple least-squares analysis applied to non-linear problems, it is reasonable to iterate any number of times in the calculation of any other quantities which are based on the choice of prior parameters used to linearize the problem, e.g., the design matrix elements. The point here is that the initial selection of these parameters is usually quite arbitrary, in the absence of concrete prior information. These prior values are chosen for convenience and because it is thought that the true solution might be found somewhere in that vicinity of parameter space. Actually, it is not unreasonable to consider some rather large jumps in the trial parameters along the way, just to make sure that one is not trapped near a local minimum of parameter space which is not the true minimum for the problem.

Rule 3:

In generalized least-squares one ought never to iterate the prior parameters or adjust those quantities derived from them, e.g., the design matrix elements (except as described under Rule 1 above). The prior parameter values are treated as "givens" in the problem. Their uncertainties are taken into consideration through the corresponding

covariance matrix. In other words, one should never re-introduce as prior information the results from any generalized least-squares calculation. The solution of a generalized least-squares problem takes on the status of "prior" information only when a new set of data is obtained which calls for a re-evaluation of the parameter set.

These rules are basically a reflection of ordinary common sense. Still, in this age of computers people are often tempted to manipulate data beyond a point that is fundamentally justified, simply because it is so easy to do. In my experience, faithful application of these rules will avoid many of the pitfalls that arise in typical least-squares calculations.

5.3 Numerical Precision

The calculations involved in applications of simple and generalized least-squares methods are quite involved. There are matrix multiplications, matrix inversions, etc. Clearly, there is a potential for error to be introduced by limited arithmetical precision. This is usually not a problem when the calculations are done on large computers, or even on work stations, where one typically encounters 48- to 64-bit arithmetic as the norm. However, even then it is worthwhile checking to see if there is a potential for such a problem to arise in calculations involving very large matrices, etc. Our concern here is mainly for calculations done on personal computers. For example, the standard single-precision arithmetic for real-numbers in FORTRAN programs compiled on an IBM-compatible PC is REAL*4 (i.e, 4 bytes or 32-bit arithmetic). Double-precision is generally also available, denoted REAL*8 (i.e., 8 bytes or 64-bit arithmetic). For all of the examples considered in Sections 3 and 4 (and Appendices A and B) this was found to not be a problem. Each of these examples was investigated with both single- and double-precision arithmetic. Any differences observed between the results were totally negligible. However, one cannot assume that this will always be the case. To demonstrate this, an example drawn from Smi82 (pp. 2-10) is presented here. This problem was addressed many years ago on a computer which employed 48-bit arithmetic (see results on pp. 9-10 of Ref. Smi82). Here, we will observe the results obtained using both 32- and 64-bit arithmetic.

Without going into the details (the reader who is interested can refer to Ref. Smi82), ten calibration data points (n=10) are fitted with a polynomial expansion involving m parameters. This analysis has been carried with a program named LLSFIT which is closely related to LSMOD but differs somewhat in the details of the input and output routines. Each data point consists of three values, namely, (x_i, y_i, E_{yi}), where x_i is an independent variable (no error), y_i is the measured calibration value and E_{yi} is its corresponding error. It is assumed that these data are uncorrelated. The data set is fitted by

the polynomial $y = \Sigma_{k=1,m}\ p_k x^{k-1}$. Thus, for each data point, $y_i \approx \Sigma_{k=1,m}\ p_k {x_i}^{k-1}$. The fit was carried out with five parameters (m=5).

Input data:

x_i	y_i	E_{yi}
31.9	565.0	4.0
35.9	813.0	4.0
37.6	930.0	4.0
39.6	1047.0	4.0
42.1	1205.0	4.0
54.4	1925.0	4.0
57.17	2078.0	4.0
74.25	3008.0	4.0
98.2	4220.0	4.0
117.4	5125.0	4.0

Solution parameters:

Single precision (REAL*4)

- - - - - -

```
 P,EP
 -.17809E+04   .95140E+02
  .85280E+02   .61376E+01
 -.41358E+00   .13905E+00
  .23470E-02   .13163E-02
 -.67543E-05   .44263E-05
 CP
  .10000E+01
 -.99628E+00   .10000E+01
  .98560E+00 -.99638E+00   .10000E+01
 -.96998E+00   .98672E+00 -.99688E+00   .10000E+01
  .95164E+00 -.97314E+00   .98878E+00 -.99746E+00   .10000E+01
 CHI2,CHI2NM
  .77737E+03   .15547E+03
```

- - - - - -

Double precision (REAL*8)

- - - - - -

```
 P,EP
 -.17893D+04   .97311D+02
  .84925D+02   .62835D+01
 -.42427D+00   .14245D+00
  .25480D-02   .13489D-02
 -.70070D-05   .45359D-05
```

```
 CP
  .10000D+01
 -.99642D+00  .10000D+01
  .98620D+00 -.99653D+00  .10000D+01
 -.97129D+00  .98731D+00 -.99701D+00  .10000D+01
  .95378D+00 -.97435D+00  .98929D+00 -.99756D+00  .10000D+01
 CHI2,CHI2NM
  .49475D+01  .98950D+00
```

- - - - - -

Only those portions of the input and output that are significant for the present discussion are presented above. The parameters of the output can be interpreted the same as for LSMOD and GLSMOD. Clearly there are noticeable differences in the results. The solution parameters (P) and their errors (EP) are qualitatively similar, but certainly distinct in detail. The same can be said for the correlation matrix (CP). However, the differences for CHI2 and CHI2NM are very large. Based on single-precision arithmetic, we would conclude that a five-parameter fit gives a very poor representation of these data. When the calculation is done in double precision, a totally different picture emerges. The double-precision results here agree quite well with those first reported in Ref. Smi82 based on 48-bit arithmetic.

Therefore, we can conclude that whenever there is a likelihood that arithmetic errors might develop from using single-precision arithmetic on a small computer it is advisable to run the problem using double precision. Since the speeds and memory capacities of small computers are growing very rapidly, this should present the user with few problems in practice.

5.4 Development of Routines for Special Applications

As indicated previously, both LSMOD and GLSMOD are "bare-bones" computer routines for the application of simple and generalized least-squares analysis, respectively. In that sense, there are two serious drawbacks in relying solely on these codes - as flexible as they may be - for routine analysis.

The first limitation is inconvenience. So long as the dimensions of a problem are small (a few data points and even fewer parameters), it is not too difficult to prepare the input for these codes externally, either using separate computer routines or even hand calculations. When the number of data points is large and there are several parameters, it can be very cumbersome to prepare the input. Also, if one chooses to iterate the solution process, as described in Section 5.2, it is also very inconvenient. Finally, one often wishes to proceed with further analysis once the best-estimate values for the parameters and their errors have been obtained. So, in practice, one will generally imbed the routines of LSMOD and GLSMOD in other programs designed for specific applications, thereby trading off

generality for convenience. This is not too difficult to do. For example, code LLSFIT (employed in the example discussed in Section 5.3 but not exhibited explicitly) is a typical extension of LSMOD. All that is required is modification of the input and output sections. The core of each code can be based on either LSMOD or GLSMOD, depending on the nature of the analysis.

The second limitation in using LSMOD and GLSMOD directly for applications has to do with arithmetical precision. This is fundamentally a more serious consideration than the convenience factor mentioned above. Among those quantities which must be introduced explicitly into LSMOD or GLSMOD are the data covariance matrix (VY), values equivalent to the experimental data which are calculated from the prior parameters (YA), the prior parameter covariance matrix (VA) and the design matrix (A). There is a practical limit to the precision allowed by the code data-input formats (generally no more than 6-8 significant figures). This limitation may be fatal in certain applications. Therefore, there is a great advantage from the point of view of precision to limiting the actual input to only the prior parameter values (PA), with associated error information, and the experimental data (Y), with its error information, and calculating everything else internally to assure adequate precision.

6. CONCLUSIONS

The following conclusions can be drawn from the present investigation:

1. The methods of simple and generalized least-squares analysis are applicable to many important problems in the physical sciences.

2. The algorithms incorporated in the generic codes LSMOD and GLSMOD are very convenient for addressing problems of modest size which involve simple and generalized least-squares analysis, respectively. Because of the general nature of their input, these codes can be applied to a variety of problems without regard to the specific physical features. However, extensive analysis external to these codes is required to prepare this input.

3. In practical applications of a larger scale, it is advisable to incorporate the algorithms of these codes into more specialized computer routines that are designed to deal with specific problems. There are two important reasons for this: i) convenience, and ii) avoidance of loss of precision.

4. For simple problems with few data and parameters, it is often adequate to employ single-precision arithmetic (e.g., REAL*4) on small computers. For larger problems with extensive data sets and several parameters, it is usually imperative to employ the higher precision afforded by the double-precision option (REAL*8) when performing calculations on small computers. To be on the safe side, one can opt to always use double precision. This choice demands greater memory capacity and longer running times, but this is seldom a problem for modern computers.

5. Care is required when calculating the data covariance matrix for least-squares problems, particularly when the data scatter considerably or show some inconsistencies (Peelle's Pertinent Puzzle phenomenon). To minimize this problem, the procedure described in Section 5.1 can be employed.

6. The technique of iteration can often be applied to advantage in least-squares analysis, but it is important to adhere to the three simple rules described in Section 5.2.

ACKNOWLEDGEMENTS

Drs. Donald G. Gardner and Maureen A. Gardner (Nuclear Chemistry Division, Lawrence Livermore National Laboratory) pointed out the importance of numerical precision in applications of least-squares methods, made results of their detailed studies available, and offered valuable comments on this manuscript. Mrs. Adrienne H. Novick (Engineering Physics Division, Argonne National Laboratory) provided alternative (more modern) versions of FORTRAN source coding for LSMOD and GLSMOD (see Appendices A.1 and B.1, respectively).

REFERENCES

CS91 Satoshi Chiba and Donald L. Smith, "A Suggested Procedure for Resolving an Anomaly in Least-squares Data Analysis Known as "Peelle's Pertinent Puzzle" and the General Implications for Nuclear Data Evaluation", Report ANL/NDM-121, Argonne National Laboratory, Argonne, Illinois, U.S.A. (1991).

Fro86 F.H. Froehner, "Principles and Methods of Data Evaluation", Report KfK-4099, Kernforschungszentrum Karlsruhe, Karlsruhe, Germany (1986).

GS88 Luiz P. Geraldo and Donald L. Smith, "Some Thoughts on Positive Definiteness in the Consideration of Nuclear Data Covariance Matrices", Report ANL/NDM-104, Argonne National Laboratory, Argonne, Illinois, U.S.A. (1988).

Jay78 E.T. Jaynes, "Where Do We Stand on Maximum Entropy?", The Maximum Entropy Formalism, eds. R.D. Legine and M. Tribus, M.I.T. University Press, Cambridge, Massachusetts, U.S.A. (1978).

Man81 W. Mannhart, "A Small Guide to Generating Covariances of Experimental Data", Report PTB-FMRB-84, Physikalisch-Technische Bundesanstalt, Braunschweig, Germany (1981).

Sha48 C.E. Shannon, Bell Syst. Tech. Journal 27, 379 and 623 (1948).

Smi81 D.L. Smith, "Covariance Matrices and Applications to the Field of Nuclear Data", Report ANL/NDM-62, Argonne National Laboratory, Argonne, Illinois, U.S.A. (1981).

Smi82 D.L. Smith, "Non-evaluation Applications for Covariance Matrices", Report ANL/NDM-67, Argonne National Laboratory, Argonne, Illinois, U.S.A. (1982).

Smi87a D.L. Smith, "Some Comments on the Effects of Long-range Correlations in Covariance Matrices for Nuclear Data", Report ANL/NDM-99, Argonne National Laboratory, Argonne, Illinois, U.S.A. (1987).

Smi87b D.L. Smith, Nuclear Instr. and Methods in Physics Research A257, 365 (1987).

Smi91 D.L. Smith, Probability, Statistics, and Data Uncertainties in Nuclear Science and Technology, American Nuclear Society, Lagrange Park, Illinois, U.S.A. (1991).

Zeh70 P.W. Zehna, Probability Distributions and Statistics, Allyn and Bacon, Inc., Boston, Massachusetts (1970).

APPENDIX A: LSMOD

This code has been compiled and used routinely on an IBM-compatible personal computer by the author. The source listing(s), executable program(s) and some test problems (input and output) can be obtained on a floppy disk by contacting the author.

A.1 FORTRAN Source

Original Version (D.L. Smith)

```
C
C     LSMOD
C
C     IBM PC - D.L. SMITH AND R.T. MAINARDI - 17 JULY 1988
C
C     REVISED 12/18/92. ALTERED I/O AND LEAST-SQUARES ROUTINE.
C
      DIMENSION Y(50),EY(50),CY(50,50),VY(50,50),VYI(50,50),
     1P(10),EP(10),CP(10,10),VP(10,10),VPI(10,10),A(50,10),
     2QN(50),WN(50,51),QM(10),WM(10,11)
C
C     INITIALIZATION AND CONTROL
C
      WRITE(*,1)
    1 FORMAT(' LSMOD'/)
      WRITE(*,2)
    2 FORMAT(' ENTER INPUT FILE NAME (UNIT 4)'/)
      OPEN(4,FILE=' ',STATUS='OLD ')
      WRITE(*,3)
    3 FORMAT(' ENTER OUTPUT FILE NAME (UNIT 5)'/)
      OPEN(5,FILE=' ',STATUS='OLD ')
C
C     READ INPUT FROM FILE (UNIT 4)
C
      READ(4,10) N,M
   10 FORMAT(16I5)
      READ(4,12) (Y(I),I=1,N)
   12 FORMAT(6E12.6)
      DO 17 I=1,N
   17 READ(4,12) (VY(I,J),J=1,I)
      DO 18 I=1,N
      DO 18 J=1,I
   18 VY(J,I)=VY(I,J)
      DO 19 I=1,N
   19 READ(4,12) (A(I,J),J=1,M)
C
C     ORDINARY LEAST-SQUARES ANALYSIS
C
      DO 1004 I=1,N
 1004 EY(I)=SQRT(VY(I,I))
```

```
      DO 1005 I=1,N
      DO 1005 J=1,N
 1005 CY(I,J)=VY(I,J)/EY(I)/EY(J)
      CALL MATINV(QN,VY,VYI,WN,NTEST,N,50,51)
      IF(NTEST.EQ.1) GO TO 1009
 1007 WRITE(*,1008)
 1008 FORMAT(' NO INV'/)
      STOP
 1009 DO 1010 I=1,M
      DO 1010 J=1,M
      VPI(I,J)=0.0
      DO 1010 K2=1,N
      DO 1010 K1=1,N
 1010 VPI(I,J)=VPI(I,J)+A(K2,I)*VYI(K2,K1)*A(K1,J)
      CALL MATINV(QM,VPI,VP,WM,NTEST,M,10,11)
      IF(NTEST.EQ.0) GO TO 1007
      DO 1011 I=1,M
      P(I)=0.0
      DO 1011 K3=1,M
      DO 1011 K2=1,N
      DO 1011 K1=1,N
 1011 P(I)=P(I)+VP(I,K3)*A(K2,K3)*VYI(K2,K1)*Y(K1)
      DO 1012 I=1,M
 1012 EP(I)=SQRT(VP(I,I))
      DO 1013 I=1,M
      DO 1013 J=1,M
 1013 CP(I,J)=VP(I,J)/EP(I)/EP(J)
      DO 1014 I=1,N
      QN(I)=Y(I)
      DO 1014 K1=1,M
 1014 QN(I)=QN(I)-A(I,K1)*P(K1)
      CHI2=0.0
      DO 1015 K2=1,N
      DO 1015 K1=1,N
 1015 CHI2=CHI2+QN(K2)*VYI(K2,K1)*QN(K1)
      CHI2NM=CHI2/FLOAT(N-M)
C
C     PRINT OUTPUT TO FILE (UNIT 5)
C
      WRITE(5,40)
   40 FORMAT(' Y')
   41 WRITE(5,12) (Y(I),I=1,N)
      WRITE(5,410)
  410 FORMAT(' EY')
      WRITE(5,12) (EY(I),I=1,N)
      WRITE(5,411)
  411 FORMAT(' VY')
      DO 412 I=1,N
  412 WRITE(5,12) (VY(I,J),J=1,I)
      WRITE(5,43)
   43 FORMAT(' CY')
      DO 44 I=1,N
   44 WRITE(5,12) (CY(I,J),J=1,I)
```

```
      WRITE(5,441)
  441 FORMAT(' A')
      DO 442 I=1,N
  442 WRITE(5,12) (A(I,J),J=1,M)
      WRITE (5,45)
   45 FORMAT(' P')
      WRITE(5,12) (P(I),I=1,M)
      WRITE(5,451)
  451 FORMAT(' EP')
      WRITE(5,12) (EP(I),I=1,M)
      WRITE(5,452)
  452 FORMAT(' VP')
      DO 453 I=1,M
  453 WRITE(5,12) (VP(I,J),J=1,I)
      WRITE(5,47)
   47 FORMAT(' CP')
      DO 48 I=1,M
   48 WRITE(5,12) (CP(I,J),J=1,I)
      WRITE(5,49)
   49 FORMAT(' CHI2,CHI2NM')
      WRITE(5,12) CHI2,CHI2NM
      STOP
      END
      SUBROUTINE MATINV(B,D,Q,E,NTEST,NS,NARA,NMAX)
      DIMENSION B(NARA),D(NARA,NARA),Q(NARA,NARA),E(NARA,NMAX)
      IP=NS+1
      BIG=0.0
      DO 555 I=1,NS
      DO 555 J=1,NS
      ABD=ABS(D(I,J))
      IF(ABD-BIG) 555,555,554
  554 BIG=ABD
  555 CONTINUE
      FACT=SQRT(BIG)
      I=1
    1 IF(I-NS) 2,2,20
    2 J=1
    3 IF(J-NS) 4,4,8
    4 K=1
    5 IF(K-NS) 6,6,7
    6 E(J,K)=D(K,J)/FACT
      K=K+1
      GO TO 5
    7 J=J+1
      GO TO 3
    8 L=1
    9 IF(L-NS) 10,10,14
   10 IF(L-I) 11,13,11
   11 E(L,IP)=0.0
   12 L=L+1
      GO TO 9
   13 E(L,IP)=1.0
      GO TO 12
```

```
   14 CALL JORDAN(B,E,NTEST,NS,NARA,NMAX)
      IF(NTEST) 15,15,16
   15 RETURN
   16 M=1
   17 IF(M-NS) 18,18,19
   18 Q(I,M)=E(M,IP)/FACT
      M=M+1
      GO TO 17
   19 I=I+1
      GO TO 1
   20 RETURN
      END
      SUBROUTINE JORDAN(B,C,INDEX,N,NARA,NMAX)
C
C     SUBROUTINE JORDAN SOLVES A SYSTEM OF LINEAR NONHOMOGENEOUS
C     EQUATIONS BY THE METHOD OF GAUSS-JORDAN REDUCTION. IF THE SYSTEM
C     FAILS TO HAVE A SOLUTION, A FLAG IS SET WHICH SIGNALS THE MAIN
C     PROGRAM.
C
      DIMENSION B(NARA),C(NARA,NMAX)
      K=1
    1 IF(K-N) 2,2,22
    2 IF(C(K,K)) 10,3,10
    3 L=K+1
    4 IF(L-N) 5,5,21
    5 IF(C(L,K)) 7,6,7
    6 L=L+1
      GO TO 4
    7 M=1
    8 IF(M-N-1) 9,9,2
    9 B(M)=C(K,M)
      C(K,M)=C(L,M)
      C(L,M)=B(M)
      M=M+1
      GO TO 8
   10 J=N+1
   11 IF(J-K) 13,12,12
   12 C(K,J)=C(K,J)/C(K,K)
      J=J-1
      GO TO 11
   13 I=1
   14 IF(I-N) 16,16,15
   15 K=K+1
      GO TO 1
   16 IF(I-K) 18,17,18
   17 I=I+1
      GO TO 14
   18 II=N+1
   19 IF(II-K) 17,20,20
   20 C(I,II)=C(I,II)-C(I,K)*C(K,II)
      II=II-1
      GO TO 19
   21 INDEX=0
```

```
      GO TO 23
   22 INDEX=1
   23 RETURN
      END
```

Alternative Version (A.H. Novick)

```
C     LSMOD
C
C     IBM PC - D.L. SMITH AND R.T. MAINARDI - 17 JULY 1988
C
C     REVISED 12/18/92. ALTERED I/O AND LEAST-SQUARES ROUTINE.
C
      DIMENSION Y(50), EY(50), CY(50, 50), VY(50, 50), VYI(50, 50),
     *    P(10), EP(10), CP(10, 10), VP(10, 10), VPI(10, 10),
     *    A(50, 10), QN(50), WN(50, 51), QM(10), WM(10, 11)
C
C     INITIALIZATION AND CONTROL
C
      WRITE (*, 1)
    1 FORMAT (' LSMOD'/)
      WRITE (*, 2)
    2 FORMAT (' ENTER INPUT FILE NAME (UNIT 4)'/)
      OPEN (4, FILE=' ', STATUS='OLD ')
      WRITE (*, 3)
    3 FORMAT (' ENTER OUTPUT FILE NAME (UNIT 5)'/)
      OPEN (5, FILE=' ', STATUS='OLD ')
C
C     READ INPUT FROM FILE (UNIT 4)
C
      READ (4, 10) N, M
   10 FORMAT (16I5)
      READ (4, 12) (Y(I), I = 1, N)
   12 FORMAT (6E12.6)
      DO 17 I = 1, N
   17 READ (4, 12) (VY(I, J), J = 1, I)
      DO 18 I = 1, N
        DO 18 J = 1, I
   18 VY(J, I) = VY(I, J)
      DO 19 I = 1, N
   19 READ (4, 12) (A(I, J), J = 1, M)
C
C     ORDINARY LEAST-SQUARES ANALYSIS
C
      DO 1004 I = 1, N
 1004 EY(I) = SQRT(VY(I, I))
      DO 1005 I = 1, N
        DO 1005 J = 1, N
 1005 CY(I, J) = VY(I, J) / EY(I) / EY(J)
      CALL MATINV(QN, VY, VYI, WN, NTEST, N, 50, 51)
      IF (NTEST .EQ. 1) GOTO 1009
```

```
1007 WRITE (*, 1008)
1008 FORMAT (' NO INV'/)
     STOP
1009 DO 1010 I = 1, M
       DO 1010 J = 1, M
         VPI(I, J) = 0.0
         DO 1010 K2 = 1, N
           DO 1010 K1 = 1, N
1010 VPI(I, J) = VPI(I, J) + A(K2, I) * VYI(K2, K1) * A(K1, J)
     CALL MATINV(QM, VPI, VP, WM, NTEST, M, 10, 11)
     IF (NTEST .EQ. 0) GOTO 1007
     DO 1011 I = 1, M
       P(I) = 0.0
       DO 1011 K3 = 1, M
         DO 1011 K2 = 1, N
           DO 1011 K1 = 1, N
1011 P(I) = P(I) + VP(I, K3) * A(K2, K3) * VYI(K2, K1) * Y(K1)
     DO 1012 I = 1, M
1012 EP(I) = SQRT(VP(I, I))
     DO 1013 I = 1, M
       DO 1013 J = 1, M
1013 CP(I, J) = VP(I, J) / EP(I) / EP(J)
     DO 1014 I = 1, N
       QN(I) = Y(I)
       DO 1014 K1 = 1, M
1014 QN(I) = QN(I) - A(I, K1) * P(K1)
     CHI2 = 0.0
     DO 1015 K2 = 1, N
       DO 1015 K1 = 1, N
1015 CHI2 = CHI2 + QN(K2) * VYI(K2, K1) * QN(K1)
     CHI2NM = CHI2 / FLOAT(N - M)
C
C    PRINT OUTPUT TO FILE (UNIT 5)
C
     WRITE (5, 40)
  40 FORMAT (' Y')
  41 WRITE (5, 12) (Y(I), I = 1, N)
     WRITE (5, 410)
 410 FORMAT (' EY')
     WRITE (5, 12) (EY(I), I = 1, N)
     WRITE (5, 411)
 411 FORMAT (' VY')
     DO 412 I = 1, N
 412 WRITE (5, 12) (VY(I, J), J = 1, I)
     WRITE (5, 43)
  43 FORMAT (' CY')
     DO 44 I = 1, N
  44 WRITE (5, 12) (CY(I, J), J = 1, I)
     WRITE (5, 441)
 441 FORMAT (' A')
     DO 442 I = 1, N
 442 WRITE (5, 12) (A(I, J), J = 1, M)
```

```
      WRITE (5, 45)
   45 FORMAT (' P')
      WRITE (5, 12) (P(I), I = 1, M)
      WRITE (5, 451)
  451 FORMAT (' EP')
      WRITE (5, 12) (EP(I), I = 1, M)
      WRITE (5, 452)
  452 FORMAT (' VP')
      DO 453 I = 1, M
  453 WRITE (5, 12) (VP(I, J), J = 1, I)
      WRITE (5, 47)
   47 FORMAT (' CP')
      DO 48 I = 1, M
   48 WRITE (5, 12) (CP(I, J), J = 1, I)
      WRITE (5, 49)
   49 FORMAT (' CHI2,CHI2NM')
      WRITE (5, 12) CHI2, CHI2NM
      STOP
      END
      SUBROUTINE MATINV(B, D, Q, E, NTEST, NS, NARA, NMAX)
      DIMENSION B(NARA), D(NARA, NARA), Q(NARA, NARA), E(NARA, NMAX)
      IP = NS + 1
      BIG = 0.0
      DO 555 I = 1, NS
        DO 555 J = 1, NS
          ABD = ABS(D(I, J))
          IF (ABD .GT. BIG) THEN
  554       BIG = ABD
          ENDIF
  555 CONTINUE
      FACT = SQRT(BIG)
      I = 1
    1 IF (I .LE. NS) THEN
    2   J = 1
    3   IF (J .LE. NS) THEN
    4     K = 1
    5     IF (K .LE. NS) THEN
    6       E(J, K) = D(K, J) / FACT
            K = K + 1
            GOTO 5
          ENDIF
    7     J = J + 1
          GOTO 3
        ENDIF
    8   L = 1
    9   IF (L .LE. NS) THEN
   10     IF (L .EQ. I) GOTO 13
   11     E(L, IP) = 0.0
   12     L = L + 1
          GOTO 9
   13     E(L, IP) = 1.0
          GOTO 12
        ENDIF
```

```
   14   CALL JORDAN(B, E, NTEST, NS, NARA, NMAX)
        IF (NTEST .LE. 0) THEN
   15     RETURN
        ENDIF
   16   M = 1
   17   IF (M .LE. NS) THEN
   18     Q(I, M) = E(M, IP) / FACT
          M = M + 1
          GOTO 17
        ENDIF
   19   I = I + 1
        GOTO 1
      ENDIF
   20 RETURN
      END
      SUBROUTINE JORDAN(B, C, INDEX, N, NARA, NMAX)
C
C     SUBROUTINE JORDAN SOLVES A SYSTEM OF LINEAR NONHOMOGENEOUS
C     EQUATIONS BY THE METHOD OF GAUSS-JORDAN REDUCTION. IF THE SYSTEM
C     FAILS TO HAVE A SOLUTION, A FLAG IS SET WHICH SIGNALS THE MAIN
C     PROGRAM.
C
      DIMENSION B(NARA), C(NARA, NMAX)
      K = 1
    1 IF (K .LE. N) THEN
    2   IF (C(K, K) .EQ. 0) THEN
    3     L = K + 1
    4     IF (L .GT. N) GOTO 21
    5     IF (C(L, K) .EQ. 0) THEN
    6       L = L + 1
            GOTO 4
          ENDIF
    7     M = 1
    8     IF (M - N - 1 .GT. 0) GOTO 2
    9     B(M) = C(K, M)
          C(K, M) = C(L, M)
          C(L, M) = B(M)
          M = M + 1
          GOTO 8
        ENDIF
   10   J = N + 1
   11   IF (J .GE. K) THEN
   12     C(K, J) = C(K, J) / C(K, K)
          J = J - 1
          GOTO 11
        ENDIF
   13   I = 1
   14   IF (I .GT. N) THEN
   15     K = K + 1
          GOTO 1
        ENDIF
   16   IF (I .NE. K) GOTO 18
   17   I = I + 1
```

```
      GOTO 14
18    II = N + 1
19    IF (II .LT. K) GOTO 17
20    C(I, II) = C(I, II) - C(I, K) * C(K, II)
      II = II - 1
      GOTO 19
21    INDEX = 0
   ELSE
22    INDEX = 1
   ENDIF
23 RETURN
   END
```

A.2 Test Problem Input

See Section 3.2 for a description of the input parameters and format.

```
   2    1
1.85          1.94
0.01232
0.008614      0.02409
1.0
1.0
```

A.3 Test Problem Output

See Section 3.2 for a description of the output parameters.

```
 Y
 .185000E+01 .194000E+01
 EY
 .110995E+00 .155210E+00
 VY
 .123200E-01
 .861400E-02 .240900E-01
 CY
 .100000E+01
 .500013E+00 .100000E+01
 A
 .100000E+01
 .100000E+01
 P
 .186739E+01
 EP
 .107722E+00
 VP
 .116040E-01
```

```
CP
.100000E+01
CHI2,CHI2NM
.422271E+00 .422271E+00
```

APPENDIX B: GLSMOD

This code has been compiled and used routinely on an IBM-compatible personal computer by the author. The source listing(s), executable program(s) and some test problems (input and output) can be obtained on a floppy disk by contacting the author.

B.1 FORTRAN Source

Original Version (D.L. Smith)

```
C     GLSMOD
C
C     IBM PC - D.L. SMITH  21 DECEMBER 1992
C
      DIMENSION Y(40),EY(40),VY(40,40),CY(40,40),Q(40,40),QVY(40,40),
     1QVYI(40,40),YA(40),P(10),EP(10),VP(10,10),CP(10,10),PA(10),
     2EA(10),VA(10,10),CA(10,10),A(40,10),QN(40),WN(40,41)
C
C     INITIALIZATION AND CONTROL
C
      WRITE(*,1)
    1 FORMAT(' GLSMOD'/)
      WRITE(*,2)
    2 FORMAT(' ENTER INPUT FILE NAME (UNIT 4)'/)
      OPEN(4,FILE=' ',STATUS='OLD ')
      WRITE(*,3)
    3 FORMAT(' ENTER OUTPUT FILE NAME (UNIT 5)'/)
      OPEN(5,FILE=' ',STATUS='OLD ')
C
C     READ INPUT FROM FILE (UNIT 4)
C
      READ(4,10) N,M
   10 FORMAT(16I5)
      READ(4,12) (Y(I),I=1,N)
   12 FORMAT(6E12.6)
      READ(4,12) (YA(I),I=1,N)
      DO 17 I=1,N
   17 READ(4,12) (VY(I,J),J=1,I)
      DO 18 I=1,N
      DO 18 J=1,I
   18 VY(J,I)=VY(I,J)
      READ(4,12) (PA(I),I=1,M)
      DO 19 I=1,M
   19 READ(4,12) (VA(I,J),J=1,I)
      DO 20 I=1,M
      DO 20 J=1,I
   20 VA(J,I)=VA(I,J)
      DO 30 I=1,N
   30 READ(4,12) (A(I,J),J=1,M)
```

```
C
C     GENERAL LEAST-SQUARES ANALYSIS
C
      DO 1040 I=1,N
 1040 EY(I)=SQRT(VY(I,I))
      DO 1050 I=1,N
      DO 1050 J=1,N
 1050 CY(I,J)=VY(I,J)/EY(I)/EY(J)
      DO 1051 I=1,M
 1051 EA(I)=SQRT(VA(I,I))
      DO 1052 I=1,M
      DO 1052 J=1,M
 1052 CA(I,J)=VA(I,J)/EA(I)/EA(J)
      DO 1053 I=1,N
      DO 1053 J=1,N
      Q(I,J)=0.0
      DO 1053 K2=1,M
      DO 1053 K1=1,M
 1053 Q(I,J)=Q(I,J)+A(I,K2)*VA(K2,K1)*A(J,K1)
      DO 1054 I=1,N
      DO 1054 J=1,N
 1054 QVY(I,J)=Q(I,J)+VY(I,J)
      CALL MATINV(QN,QVY,QVYI,WN,NTEST,N,40,41)
      IF(NTEST.EQ.1) GO TO 1090
      WRITE(*,1080)
 1080 FORMAT(' NO INV'/)
      STOP
 1090 CONTINUE
      DO 1100 I=1,M
      DO 1100 J=1,M
      VP(I,J)=VA(I,J)
      DO 1100 K4=1,M
      DO 1100 K3=1,N
      DO 1100 K2=1,N
      DO 1100 K1=1,M
 1100 VP(I,J)=VP(I,J)-VA(I,K4)*A(K3,K4)*QVYI(K3,K2)*A(K2,K1)*VA(K1,J)
      DO 1110 I=1,N
 1110 QN(I)=Y(I)-YA(I)
      DO 1200 I=1,M
      P(I)=PA(I)
      DO 1200 K3=1,M
      DO 1200 K2=1,N
      DO 1200 K1=1,N
 1200 P(I)=P(I)+VA(I,K3)*A(K2,K3)*QVYI(K2,K1)*QN(K1)
      DO 1300 I=1,M
 1300 EP(I)=SQRT(VP(I,I))
      DO 1400 I=1,M
      DO 1400 J=1,M
 1400 CP(I,J)=VP(I,J)/EP(I)/EP(J)
      CHI2=0.0
      DO 1600 K2=1,N
      DO 1600 K1=1,N
 1600 CHI2=CHI2+QN(K2)*QVYI(K2,K1)*QN(K1)
```

```
      CHI2NM=CHI2/FLOAT(N)
C
C     PRINT OUTPUT TO FILE (UNIT 5)
C
      WRITE(5,1700)
 1700 FORMAT(' YA')
      WRITE(5,12) (YA(I),I=1,N)
      WRITE(5,40)
   40 FORMAT(' Y')
   41 WRITE(5,12) (Y(I),I=1,N)
      WRITE(5,410)
  410 FORMAT(' EY')
      WRITE(5,12) (EY(I),I=1,N)
      WRITE(5,411)
  411 FORMAT(' VY')
      DO 412 I=1,N
  412 WRITE(5,12) (VY(I,J),J=1,I)
      WRITE(5,43)
   43 FORMAT(' CY')
      DO 44 I=1,N
   44 WRITE(5,12) (CY(I,J),J=1,I)
      WRITE(5,441)
  441 FORMAT(' A')
      DO 442 I=1,N
  442 WRITE(5,12) (A(I,J),J=1,M)
      WRITE(5,4000)
 4000 FORMAT(' PA')
      WRITE(5,12) (PA(I),I=1,M)
      WRITE(5,4100)
 4100 FORMAT(' EA')
      WRITE(5,12) (EA(I),I=1,M)
      WRITE(5,4200)
 4200 FORMAT(' VA')
      DO 4300 I=1,M
 4300 WRITE(5,12) (VA(I,J),J=1,I)
      WRITE(5,4400)
 4400 FORMAT(' CA')
      DO 4500 I=1,M
 4500 WRITE(5,12) (CA(I,J),J=1,I)
      WRITE (5,45)
   45 FORMAT(' P')
      WRITE(5,12) (P(I),I=1,M)
      WRITE(5,451)
  451 FORMAT(' EP')
      WRITE(5,12) (EP(I),I=1,M)
      WRITE(5,452)
  452 FORMAT(' VP')
      DO 453 I=1,M
  453 WRITE(5,12) (VP(I,J),J=1,I)
      WRITE(5,47)
   47 FORMAT(' CP')
      DO 48 I=1,M
   48 WRITE(5,12) (CP(I,J),J=1,I)
```

```
      WRITE(5,49)
   49 FORMAT(' CHI2,CHI2NM')
      WRITE(5,12) CHI2,CHI2NM
      STOP
      END
      SUBROUTINE MATINV(B,D,Q,E,NTEST,NS,NARA,NMAX)
      DIMENSION B(NARA),D(NARA,NARA),Q(NARA,NARA),E(NARA,NMAX)
      IP=NS+1
      BIG=0.0
      DO 555 I=1,NS
      DO 555 J=1,NS
      ABD=ABS(D(I,J))
      IF(ABD-BIG) 555,555,554
  554 BIG=ABD
  555 CONTINUE
      FACT=SQRT(BIG)
      I=1
    1 IF(I-NS) 2,2,20
    2 J=1
    3 IF(J-NS) 4,4,8
    4 K=1
    5 IF(K-NS) 6,6,7
    6 E(J,K)=D(K,J)/FACT
      K=K+1
      GO TO 5
    7 J=J+1
      GO TO 3
    8 L=1
    9 IF(L-NS) 10,10,14
   10 IF(L-I) 11,13,11
   11 E(L,IP)=0.0
   12 L=L+1
      GO TO 9
   13 E(L,IP)=1.0
      GO TO 12
   14 CALL JORDAN(B,E,NTEST,NS,NARA,NMAX)
      IF(NTEST) 15,15,16
   15 RETURN
   16 M=1
   17 IF(M-NS) 18,18,19
   18 Q(I,M)=E(M,IP)/FACT
      M=M+1
      GO TO 17
   19 I=I+1
      GO TO 1
   20 RETURN
      END
      SUBROUTINE JORDAN(B,C,INDEX,N,NARA,NMAX)
C
C     SUBROUTINE JORDAN SOLVES A SYSTEM OF LINEAR NONHOMOGENEOUS
C     EQUATIONS BY THE METHOD OF GAUSS-JORDAN REDUCTION. IF THE SYSTEM
```

```
C     FAILS TO HAVE A SOLUTION, A FLAG IS SET WHICH SIGNALS THE MAIN
C     PROGRAM.
C
      DIMENSION B(NARA),C(NARA,NMAX)
      K=1
    1 IF(K-N) 2,2,22
    2 IF(C(K,K)) 10,3,10
    3 L=K+1
    4 IF(L-N) 5,5,21
    5 IF(C(L,K)) 7,6,7
    6 L=L+1
      GO TO 4
    7 M=1
    8 IF(M-N-1) 9,9,2
    9 B(M)=C(K,M)
      C(K,M)=C(L,M)
      C(L,M)=B(M)
      M=M+1
      GO TO 8
   10 J=N+1
   11 IF(J-K) 13,12,12
   12 C(K,J)=C(K,J)/C(K,K)
      J=J-1
      GO TO 11
   13 I=1
   14 IF(I-N) 16,16,15
   15 K=K+1
      GO TO 1
   16 IF(I-K) 18,17,18
   17 I=I+1
      GO TO 14
   18 II=N+1
   19 IF(II-K) 17,20,20
   20 C(I,II)=C(I,II)-C(I,K)*C(K,II)
      II=II-1
      GO TO 19
   21 INDEX=0
      GO TO 23
   22 INDEX=1
   23 RETURN
      END
```

Alternative Version (A.H. Novick)

```
C     GLSMOD
C
C     IBM PC - D.L. SMITH  21 DECEMBER 1992
C
      DIMENSION Y(40), EY(40), VY(40, 40), CY(40, 40), Q(40, 40),
     *    QVY(40, 40), QVYI(40, 40), YA(40), P(10), EP(10),
     *    VP(10, 10), CP(10, 10), PA(10), EA(10), VA(10, 10),
     *    CA(10, 10), A(40, 10), QN(40), WN(40, 41)
```

```
C
C     INITIALIZATION AND CONTROL
C
      WRITE (*, 1)
    1 FORMAT (' GLSMOD'/)
      WRITE (*, 2)
    2 FORMAT (' ENTER INPUT FILE NAME (UNIT 4)'/)
      OPEN (4, FILE=' ', STATUS='OLD ')
      WRITE (*, 3)
    3 FORMAT (' ENTER OUTPUT FILE NAME (UNIT 5)'/)
      OPEN (5, FILE=' ', STATUS='OLD ')
C
C     READ INPUT FROM FILE (UNIT 4)
C
      READ (4, 10) N, M
   10 FORMAT (16I5)
      READ (4, 12) (Y(I), I = 1, N)
   12 FORMAT (6E12.6)
      READ (4, 12) (YA(I), I = 1, N)
      DO 17 I = 1, N
   17 READ (4, 12) (VY(I, J), J = 1, I)
      DO 18 I = 1, N
        DO 18 J = 1, I
   18 VY(J, I) = VY(I, J)
      READ (4, 12) (PA(I), I = 1, M)
      DO 19 I = 1, M
   19 READ (4, 12) (VA(I, J), J = 1, I)
      DO 20 I = 1, M
        DO 20 J = 1, I
   20 VA(J, I) = VA(I, J)
      DO 30 I = 1, N
   30 READ (4, 12) (A(I, J), J = 1, M)
C
C     GENERAL LEAST-SQUARES ANALYSIS
C
      DO 1040 I = 1, N
 1040 EY(I) = SQRT(VY(I, I))
      DO 1050 I = 1, N
        DO 1050 J = 1, N
 1050 CY(I, J) = VY(I, J) / EY(I) / EY(J)
      DO 1051 I = 1, M
 1051 EA(I) = SQRT(VA(I, I))
      DO 1052 I = 1, M
        DO 1052 J = 1, M
 1052 CA(I, J) = VA(I, J) / EA(I) / EA(J)
      DO 1053 I = 1, N
        DO 1053 J = 1, N
          Q(I, J) = 0.0
          DO 1053 K2 = 1, M
            DO 1053 K1 = 1, M
 1053 Q(I, J) = Q(I, J) + A(I, K2) * VA(K2, K1) * A(J, K1)
```

```
      DO 1054 I = 1, N
        DO 1054 J = 1, N
 1054 QVY(I, J) = Q(I, J) + VY(I, J)
      CALL MATINV(QN, QVY, QVYI, WN, NTEST, N, 40, 41)
      IF (NTEST .NE. 1) THEN
        WRITE (*, 1080)
 1080   FORMAT (' NO INV'/)
        STOP
      ENDIF
 1090 CONTINUE
      DO 1100 I = 1, M
        DO 1100 J = 1, M
          VP(I, J) = VA(I, J)
          DO 1100 K4 = 1, M
            DO 1100 K3 = 1, N
              DO 1100 K2 = 1, N
                DO 1100 K1 = 1, M
 1100 VP(I, J) = VP(I, J) - VA(I, K4) * A(K3, K4) * QVYI(K3, K2) *
     *    A(K2, K1) * VA(K1, J)
      DO 1110 I = 1, N
 1110 QN(I) = Y(I) - YA(I)
      DO 1200 I = 1, M
        P(I) = PA(I)
        DO 1200 K3 = 1, M
          DO 1200 K2 = 1, N
            DO 1200 K1 = 1, N
 1200 P(I) = P(I) + VA(I, K3) * A(K2, K3) * QVYI(K2, K1) * QN(K1)
      DO 1300 I = 1, M
 1300 EP(I) = SQRT(VP(I, I))
      DO 1400 I = 1, M
        DO 1400 J = 1, M
 1400 CP(I, J) = VP(I, J) / EP(I) / EP(J)
      CHI2 = 0.0
      DO 1600 K2 = 1, N
        DO 1600 K1 = 1, N
 1600 CHI2 = CHI2 + QN(K2) * QVYI(K2, K1) * QN(K1)
      CHI2NM = CHI2 / FLOAT(N)
C
C     PRINT OUTPUT TO FILE (UNIT 5)
C
      WRITE (5, 1700)
 1700 FORMAT (' YA')
      WRITE (5, 12) (YA(I), I = 1, N)
      WRITE (5, 40)
   40 FORMAT (' Y')
   41 WRITE (5, 12) (Y(I), I = 1, N)
      WRITE (5, 410)
  410 FORMAT (' EY')
      WRITE (5, 12) (EY(I), I = 1, N)
      WRITE (5, 411)
  411 FORMAT (' VY')
      DO 412 I = 1, N
  412 WRITE (5, 12) (VY(I, J), J = 1, I)
```

```
      WRITE (5, 43)
   43 FORMAT (' CY')
      DO 44 I = 1, N
   44 WRITE (5, 12) (CY(I, J), J = 1, I)
      WRITE (5, 441)
  441 FORMAT (' A')
      DO 442 I = 1, N
  442 WRITE (5, 12) (A(I, J), J = 1, M)
      WRITE (5, 4000)
 4000 FORMAT (' PA')
      WRITE (5, 12) (PA(I), I = 1, M)
      WRITE (5, 4100)
 4100 FORMAT (' EA')
      WRITE (5, 12) (EA(I), I = 1, M)
      WRITE (5, 4200)
 4200 FORMAT (' VA')
      DO 4300 I = 1, M
 4300 WRITE (5, 12) (VA(I, J), J = 1, I)
      WRITE (5, 4400)
 4400 FORMAT (' CA')
      DO 4500 I = 1, M
 4500 WRITE (5, 12) (CA(I, J), J = 1, I)
      WRITE (5, 45)
   45 FORMAT (' P')
      WRITE (5, 12) (P(I), I = 1, M)
      WRITE (5, 451)
  451 FORMAT (' EP')
      WRITE (5, 12) (EP(I), I = 1, M)
      WRITE (5, 452)
  452 FORMAT (' VP')
      DO 453 I = 1, M
  453 WRITE (5, 12) (VP(I, J), J = 1, I)
      WRITE (5, 47)
   47 FORMAT (' CP')
      DO 48 I = 1, M
   48 WRITE (5, 12) (CP(I, J), J = 1, I)
      WRITE (5, 49)
   49 FORMAT (' CHI2,CHI2NM')
      WRITE (5, 12) CHI2, CHI2NM
      STOP
      END
      SUBROUTINE MATINV(B, D, Q, E, NTEST, NS, NARA, NMAX)
      DIMENSION B(NARA), D(NARA, NARA), Q(NARA, NARA), E(NARA, NMAX)
      IP = NS + 1
      BIG = 0.0
      DO 555 I = 1, NS
        DO 555 J = 1, NS
          ABD = ABS(D(I, J))
          IF (ABD .GT. BIG) THEN
  554       BIG = ABD
          ENDIF
  555 CONTINUE
      FACT = SQRT(BIG)
```

```
      I = 1
    1 IF (I .LE. NS) THEN
    2   J = 1
    3   IF (J .LE. NS) THEN
    4     K = 1
    5     IF (K .LE. NS) THEN
    6       E(J, K) = D(K, J) / FACT
            K = K + 1
            GOTO 5
          ENDIF
    7     J = J + 1
          GOTO 3
        ENDIF
    8   L = 1
    9   IF (L .LE. NS) THEN
   10     IF (L .EQ. I) GOTO 13
   11     E(L, IP) = 0.0
   12     L = L + 1
          GOTO 9
   13     E(L, IP) = 1.0
          GOTO 12
        ENDIF
   14   CALL JORDAN(B, E, NTEST, NS, NARA, NMAX)
        IF (NTEST .LE. 0) THEN
   15     RETURN
        ENDIF
   16   M = 1
   17   IF (M .LE. NS) THEN
   18     Q(I, M) = E(M, IP) / FACT
          M = M + 1
          GOTO 17
        ENDIF
   19   I = I + 1
        GOTO 1
      ENDIF
   20 RETURN
      END
      SUBROUTINE JORDAN(B, C, INDEX, N, NARA, NMAX)
C
C     SUBROUTINE JORDAN SOLVES A SYSTEM OF LINEAR NONHOMOGENEOUS
C     EQUATIONS BY THE METHOD OF GAUSS-JORDAN REDUCTION. IF THE SYSTEM
C     FAILS TO HAVE A SOLUTION, A FLAG IS SET WHICH SIGNALS THE MAIN
C     PROGRAM.
C
      DIMENSION B(NARA), C(NARA, NMAX)
      K = 1
    1 IF (K .LE. N) THEN
    2   IF (C(K, K) .EQ. 0) THEN
    3     L = K + 1
    4     IF (L .GT. N) GOTO 21
    5     IF (C(L, K) .EQ. 0) THEN
    6       L = L + 1
            GOTO 4
          ENDIF
```

```
 7      M = 1
 8      IF (M - N - 1 .GT. 0) GOTO 2
 9      B(M) = C(K, M)
        C(K, M) = C(L, M)
        C(L, M) = B(M)
        M = M + 1
        GOTO 8
       ENDIF
10     J = N + 1
11     IF (J .GE. K) THEN
12       C(K, J) = C(K, J) / C(K, K)
         J = J - 1
         GOTO 11
       ENDIF
13     I = 1
14     IF (I .GT. N) THEN
15       K = K + 1
         GOTO 1
       ENDIF
16     IF (I .NE. K) GOTO 18
17     I = I + 1
       GOTO 14
18     II = N + 1
19     IF (II .LT. K) GOTO 17
20     C(I, II) = C(I, II) - C(I, K) * C(K, II)
       II = II - 1
       GOTO 19
21     INDEX = 0
     ELSE
22     INDEX = 1
     ENDIF
23   RETURN
     END
```

B.2 Test Problem Input

See Section 3.2 for a description of the input parameters and format.

```
   2    1
1000.0       1102.0
1095.0       1095.0
1079.1125
0.0          3496.3569
1095.0
2704.0
1.0
1.0
```

B.3 Test Problem Output

See Section 3.2 for a description of the output parameters.

```
 YA
 .109500E+04 .109500E+04
 Y
 .100000E+04 .110200E+04
 EY
 .328498E+02 .591300E+02
 VY
 .107911E+04
 .000000E+00 .349636E+04
 CY
 .100000E+01
 .000000E+00 .100000E+01
 A
 .100000E+01
 .100000E+01
 PA
 .109500E+04
 EA
 .520000E+02
 VA
 .270400E+04
 CA
 .100000E+01
 P
 .104064E+04
 EP
 .251377E+02
 VP
 .631903E+03
 CP
 .100000E+01
 CHI2,CHI2NM
 .370020E+01 .185010E+01
```

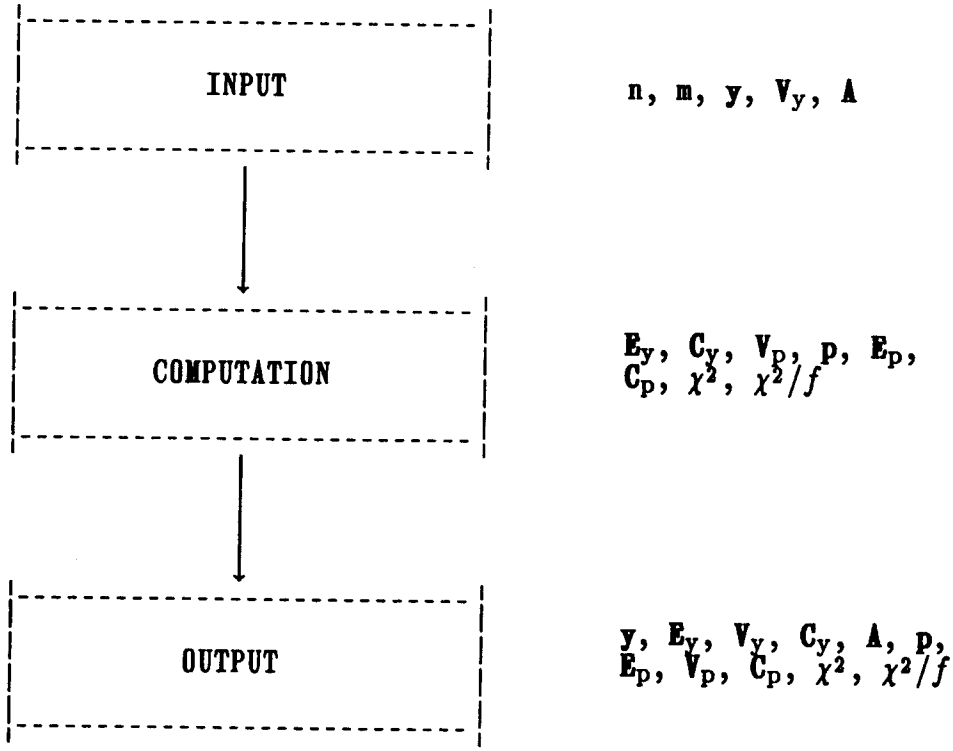


Figure 1: Flow diagram for the code LSMOD

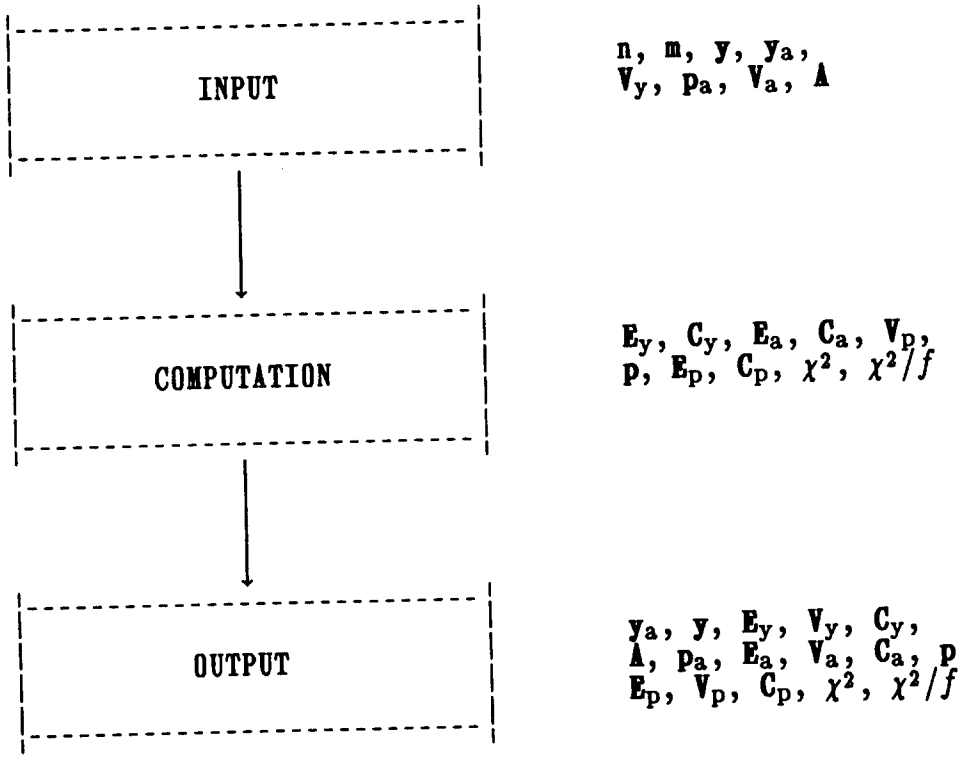


Figure 2: Flow diagram for the code GLSMOD