

38
ORNL/TM-8676
ENDF-332

ornl

OAK
RIDGE
NATIONAL
LABORATORY

UNION
CARBIDE

**User's Guide for ALEX:
Uncertainty Propagation from
Raw Data to Final Results
For Orela Transmission
Measurements**

Nancy M. Larson

OPERATED BY
UNION CARBIDE CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A06; Microfiche A01

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

**ORNL/TM-8676
ENDF-332**

**Engineering Physics Division
and Physics Division**

**USER'S GUIDE FOR ALEX: UNCERTAINTY PROPAGATION FROM RAW DATA
TO FINAL RESULTS FOR ORELA TRANSMISSION MEASUREMENTS**

**Nancy M. Larson
Computer Sciences Division**

NOTICE This document contains information of a preliminary nature.
It is subject to revision or correction and therefore does not represent a
final report.

Date Published - February 1984

**OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
DEPARTMENT OF ENERGY
Contract No. W-7405-eng-26**

TABLE OF CONTENTS

LIST OF TABLES	iv
ABSTRACT	1
1. INTRODUCTION	1
2. THEORETICAL DEVELOPMENT	2
2.1 SIMPLE CASE	2
2.2 GENERAL CASE	5
3. ORGANIZATION OF ALEX	14
3.1 BOOKKEEPER ROUTINES	14
3.2 ROUTINES TO EVALUATE σ_i, $\frac{\partial\sigma_i}{\partial c_i}$, $\frac{\partial\sigma_i}{\partial C_i}$, $\frac{\partial\sigma_i}{\partial \rho_k}$	15
3.3 EVALUATE σ^{sum} AND ITS PARTIAL DERIVATIVES	15
3.4 EVALUATE COVARIANCE MATRIX	18
3.5 EVALUATE UNCERTAINTY WHEN THE NUMBER OF BINS IS LARGE	18
4. SPECIFYING THE EXPERIMENTAL SITUATION FOR ALEX	19
4.1 SUBROUTINE INPUT	19
4.2 SUBROUTINE SETPAR	19
4.3 SUBROUTINE PREP	24
4.4 FUNCTION FUNC(IFUNC)	25
4.5 FUNCTION DERIV(IFUNC,KPAR)	25
4.6 SUBROUTINES WHICH READ RAW DATA	25
5. EXAMPLE	26
Step 1. Organize Raw Data	26
Step 2. Prepare Subroutines INPUT, SETPAR, PREP, FUNC, and DERIV	26
Step 3. Prepare Input File	26
Step 4. Execute ALEX	28
Step 5. Study the Output	28
Step 6. Rerun	34
Step 7. Additional Runs	34
6. SUMMARY AND PROGNOSIS	34
ACKNOWLEDGMENTS	34
REFERENCES	35
APPENDIX A - FORTRAN Listing of File ALEX.FOR	37
APPENDIX B - FORTRAN Listing of File ALXFIX.FOR	49
APPENDIX C - FORTRAN Listing of File ALXSUM.FOR	59
APPENDIX D - FORTRAN Listing of File ALXALL.FOR	71
APPENDIX E - FORTRAN Listing of File ALX10K.FOR	85
APPENDIX F - FORTRAN Listing of File ALXPAR.FOR	93

LIST OF TABLES

1. Summary of Data Files Required by ALEX	16
2. Description of Information Specified on the Input File	20
3. Input File FE.DAT for the Fe Example of Section 2	27
4. ALEX Prompts and User Responses for the Natural Iron Analysis. Underlined Portions are Typed by the Users; a Downward Arrow Indicates Carriage Return	29
5. Line-Printer Output for Natural Iron Analysis	30

USER'S GUIDE FOR ALEX: UNCERTAINTY PROPAGATION FROM RAW DATA TO FINAL RESULTS FOR ORELA TRANSMISSION MEASUREMENTS

Nancy M. Larson
Computer Sciences Division

ABSTRACT

This report describes a computer code (ALEX) developed to assist in *AnaLysis of EXperimental* data at the Oak Ridge Electron Linear Accelerator (ORELA). Reduction of data from raw numbers (counts per channel) to physically meaningful quantities (such as cross sections) is in itself a complicated procedure; propagation of experimental uncertainties through that reduction procedure has in the past been viewed as even more difficult - if not impossible. The purpose of the code ALEX is to correctly propagate all experimental uncertainties through the entire reduction procedure, yielding the complete covariance matrix for the reduced data, while requiring little additional input from the experimentalist beyond that which is required for the data reduction itself. This report describes ALEX in detail, with special attention given to the case of transmission measurements (the code itself is applicable, with few changes, to any type of data). Application to the natural iron measurement of D. C. Larson et al. is described in some detail.

1. INTRODUCTION

Extracting physically meaningful information (e.g., cross sections) from raw data (counts) is often a laborious procedure. Deadtime corrections must be made, backgrounds determined and subtracted, and other transformations applied. Whereas the uncertainty on the raw data (the so-called statistical uncertainty) is well understood and easily manipulated, the uncertainty on the *corrected* data involves both statistical uncertainty and other terms due to the corrections (the so-called systematic uncertainty). Carefully propagating the systematic uncertainties through all the required corrections and transformations is often perceived to be a hopelessly complex operation, as much as an order-of-magnitude more difficult than "merely" producing the corrected data. Adding to the complexity is the desire to generate off-diagonal elements of the data covariance matrix, for possible use in analysis codes such as SAMMY (NL80) or BAYES (NL82).

Thus the need was apparent for a black-box code into which the experimentalist could feed all the information he already has (raw data, deadtime corrections, backgrounds, etc.) and from which he could learn the uncertainties on the corrected data and the correlations among data points. Ideally the code would handle all the intermediate steps, determining all necessary derivatives and propagating all uncertainties to produce the final result. Moreover, the code should produce, store, and display results in an efficient manner, without requiring the entire resources of the computer.

The computer code ALEX (*AnaLyzE EXperiments*) was written in response to this need. In its present form, ALEX meets most of the requirements outlined above, for the specific case of a sample-in, sample-out measurement from which transmissions and cross sections are extracted. Extensions to other types of experiments will be introduced as required.

Since ALEX was developed for use at the Oak Ridge Electron Linear Accelerator (ORELA), data input and output are accomplished via ORELA data format (ODF) files (JC78). Conversion for use on other computer systems should, however, be straightforward since only input and output are ORELA-specific.

In Section 2 of this report is a discussion of the method used for propagating uncertainties, including relevant equations. A general description of the organization of the code ALEX is presented in Section 3, and details on those subroutines which must be changed for each special case are given in Section 4. In Section 5, we describe the application of ALEX to the natural iron measurement of D. C. Larson et al. (DL80). Concluding remarks are given in Section 6. FORTRAN listings of the code are given in the appendixes.

2. THEORETICAL DEVELOPMENT

In this section, we describe the method used to propagate uncertainties from the measured raw data, through all the necessary corrections and transformations, to the final corrected data. We shall assume that the experiment to be analyzed consists of sample-in and sample-out measurements, and that the desired "final corrected data" are in the form of cross sections. (The code ALEX was, however, designed to be far more general than this special case and is not inherently limited to transmission measurements. Generalizations to other types of experiments are straightforward, and will be incorporated into ALEX as they are needed.)

2.1 SIMPLE CASE

We first consider a simple transmission experiment for which the background level is independent of time for both sample-in and sample-out. This corresponds, for example, to the natural iron data of D. C. Larson et al. (DL80). In Subsection 2.2 we shall describe the more general case.

The experimental cross section obtained from the measurement described above is given by

$$\sigma_i = -\frac{1}{n} \ln T_i \quad (1)$$

where n is the sample thickness and T_i is the measured transmission for channel i . T_i , in turn, is given by

$$T_i = \frac{M (d_i c_i - g)}{m (D_i C_i - G)} \quad (2)$$

where m (M) is the number of monitor counts for sample-in (sample-out), c_i (C_i) the raw data or number of counts, d_i (D_i) the deadtime correction factor, and g (G) the constant background for sample-in (sample-out). Note the convention that capital letters refer to sample-out, lower case to sample-in.

The uncertainty on σ_i and the covariance between channels i and j are found by taking expectation values of small increments $\delta\sigma_i$. That is, the uncertainty $\Delta\sigma_i$ is given by

$$(\Delta\sigma_i)^2 = \langle \delta\sigma_i^2 \rangle \quad (3)$$

and the covariance by

$$Cov_{ij} = \langle \delta\sigma_i \delta\sigma_j \rangle \quad . \quad (4)$$

To obtain a small increment $\delta\sigma_i$, we first combine Eqs. (2) and (1) yielding

$$\sigma_i = - \frac{1}{n} \left\{ \ln M - \ln m + \ln (d_i c_i - g) - \ln (D_i C_i - G) \right\} \quad (5)$$

$\delta\sigma_i$ is then found in terms of small increments in parameters n , M , m , g , and G and in counts c_i and C_i as

$$\begin{aligned} \delta\sigma_i = & \frac{\partial\sigma_i}{\partial n} \delta n + \frac{\partial\sigma_i}{\partial m} \delta m + \frac{\partial\sigma_i}{\partial M} \delta M + \frac{\partial\sigma_i}{\partial g} \delta g \\ & + \frac{\partial\sigma_i}{\partial G} \delta G + \frac{\partial\sigma_i}{\partial c_i} \delta c_i + \frac{\partial\sigma_i}{\partial C_i} \delta C_i \end{aligned} \quad (6)$$

or, explicitly evaluating the derivatives,

$$\delta\sigma_i = - \frac{1}{n} \left\{ \sigma_i \delta n + \frac{1}{M} \delta M - \frac{1}{m} \delta m + \frac{d_i \delta c_i - \delta g}{d_i c_i - g} - \frac{D_i \delta C_i - \delta G}{D_i C_i - G} \right\} \quad . \quad (7)$$

The covariance between channel i and j is then given by

$$\begin{aligned}
\langle \delta\sigma_i \delta\sigma_j \rangle &= \frac{1}{n^2} \left\{ \sigma_i \sigma_j \langle \delta n^2 \rangle + \frac{1}{M^2} \langle \delta M^2 \rangle + \frac{1}{m^2} \langle \delta m^2 \rangle \right. \\
&+ \frac{d_i^2}{(d_i c_i - g)^2} \langle \delta c_i^2 \rangle \delta_{ij} + \frac{D_i^2}{(D_i C_i - G)^2} \langle \delta C_i^2 \rangle \delta_{ij} \\
&\left. + \frac{1}{d_i c_i - g} \langle \delta g^2 \rangle \frac{1}{d_j c_j - g} + \frac{1}{D_i C_i - G} \langle \delta G^2 \rangle \frac{1}{D_j C_j - G} \right\}
\end{aligned} \tag{8}$$

where δ_{ij} represents the Kronecker delta function. In Eq. (8) we have assumed that all cross terms (e.g., $\langle \delta n \delta m \rangle$, $\langle \delta n \delta c_j \rangle$, $\langle \delta \sigma_i \delta C_i \rangle$) are identically zero. That is, we have assumed that the determination of values for each of our five parameters was done independently of all other parameters, and that the raw data are also independently determined numbers.

Equation (8) involves only quantities which have been measured or which may be estimated. Because of the nature of counting statistics (Poisson distribution), the expectation values for the raw data are given by

$$\langle \delta c_i^2 \rangle = (\Delta c_i)^2 = c_i \tag{9}$$

and

$$\langle \delta C_i^2 \rangle = (\Delta C_i)^2 = C_i \tag{10}$$

The sample thickness has been determined to be $n \pm \Delta n$, so that

$$\langle \delta n^2 \rangle = (\Delta n)^2 \tag{11}$$

Likewise the monitor counts (m and M) are known to within a specified uncertainty (Δm and ΔM), and the backgrounds were measured or fitted to $g \pm \Delta g$ and $G \pm \Delta G$.

Substituting these uncertainties into Eq. (8) gives the covariance between channels i and j (with $i \neq j$) as

$$\begin{aligned} \langle \delta\sigma_i \delta\sigma_j \rangle &= \frac{1}{n^2} \left\{ \sigma_i \sigma_j (\Delta n)^2 + \left(\frac{\Delta M}{M} \right)^2 + \left(\frac{\Delta m}{m} \right)^2 \right. \\ &\quad \left. + \frac{\Delta g^2}{(d_i c_i - g)(d_j c_j - g)} + \frac{\Delta G^2}{(D_i C_i - G)(D_j C_j - G)} \right\} . \end{aligned} \quad (12)$$

The variance, or square of the uncertainty, for channel i is given by

$$\begin{aligned} (\Delta\sigma_i)^2 &= \langle \delta\sigma_i^2 \rangle = \frac{1}{n} \left\{ (\sigma_i \Delta n)^2 + \left(\frac{\Delta M}{M} \right)^2 + \left(\frac{\Delta m}{m} \right)^2 \right. \\ &\quad \left. + \left(\frac{\Delta g}{d_i c_i - g} \right)^2 + \left(\frac{\Delta G}{D_i C_i - G} \right)^2 + \frac{d_i^2 c_i}{(d_i c_i - g)^2} + \frac{D_i^2 C_i}{(D_i C_i - G)^2} \right\} . \end{aligned} \quad (13)$$

2.2 GENERAL CASE

An actual experiment will require considerably more parameters than the five indicated above, since backgrounds will in general not be time independent but will differ from channel to channel. For example, in the natural nickel experiment reported by D. C. Larson et al. (DL83), 26 parameters are needed to properly specify the background. Also, deadtime is not considered to be an exact correction, but rather involves a parameter which describes variations in flux intensity. Thus, including sample thickness and monitor counts, thirty parameters are required for the one experiment. Equations analogous to those of the previous subsection become unduly complicated when working with so many parameters; one entire page in the nickel report is devoted to writing the equation for the covariance matrix elements.

An alternative formulation is therefore desirable. This formulation should be one which is readily adapted for programming on a computer and which can be readily modified to accommodate changing experimental conditions. Entire pages of equations or of FORTRAN should not need to be rewritten every time the background is parameterized in a slightly different manner. The following formulation meets these requirements (note that, although transmission is specified here, ALEX can readily be extended to other types of measurements).

A general form for the transmission T_i is

$$T_i = \eta_i(\rho) - \frac{d_i(\rho) c_i - b_i(\rho)}{D_i(\rho) C_i - B_i(\rho)} \quad (14)$$

where, for the right-hand side of this equation, lower-case letters refer to sample-in quantities, capitals to sample-out, and Greek to either or both. The symbol ρ represents the entire set of parameters (e.g., sample thickness, backgrounds) which contribute to analysis of this experiment. The five functions η , d , b , D , and B are defined as

- η = normalization
- d, D = deadtime correction or other multiplicative correction
- b, B = background correction or other additive correction

For the example cited above, the set of parameters is $\rho = \{ \rho_1 \rho_2 \rho_3 \rho_4 \rho_5 \} = \{ n, m, M, g, G \}$. Normalization η_i is given by M/m . Deadtime corrections are d_i and D_i (independent of ρ), and background corrections are $b_i = g$, $B_i = G$.

To simplify the notation in the equations to follow, we introduce the symbol β_{li} , for $l=1$ to 5, to describe the functions appearing in Eq. (14):

$$\begin{aligned}\beta_{1i}(\rho) &= \eta_i(\rho) \\ \beta_{2i}(\rho) &= d_i(\rho) \\ \beta_{3i}(\rho) &= b_i(\rho) \\ \beta_{4i}(\rho) &= D_i(\rho) \\ \beta_{5i}(\rho) &= B_i(\rho)\end{aligned}\quad (15)$$

so that Eq. (14) becomes

$$T_i = \beta_{1i}(\rho) \frac{\beta_{2i}(\rho)c_i - \beta_{3i}(\rho)}{\beta_{4i}(\rho)C_i - \beta_{5i}(\rho)} \quad (16)$$

If Eq. (16) is substituted into Eq. (1) for σ_i in terms of T_i , we obtain

$$\sigma_i = -\frac{1}{n} \left[\ln \beta_{1i}(\rho) + \ln \left\{ \beta_{2i}(\rho)c_i - \beta_{3i}(\rho) \right\} - \ln \left\{ \beta_{4i}(\rho)C_i - \beta_{5i}(\rho) \right\} \right] \quad (17)$$

in which the dependence of σ_i on parameters ρ is explicitly displayed.

A small increment $\delta\sigma_i$ in the cross section σ_i can then be expressed in terms of small increments $\delta\rho_k$ in parameters ρ_k and small increments δc_i (δC_i) in raw data c_i (C_i) as

$$\delta\sigma_i = \sum_k \frac{\partial\sigma_i}{\partial\rho_k} \delta\rho_k + \frac{\partial\sigma_i}{\partial c_i} \delta c_i + \frac{\partial\sigma_i}{\partial C_i} \delta C_i . \quad (18)$$

The partial derivatives in Eq. (18) can be evaluated directly from Eq. (17) as

$$\frac{\partial\sigma_i}{\partial c_i} = -\frac{1}{n} \frac{\beta_{2i}}{\beta_{2i}c_i - \beta_{3i}} \quad (19)$$

$$\frac{\partial\sigma_i}{\partial C_i} = \frac{1}{n} \frac{\beta_{4i}}{\beta_{4i}C_i - \beta_{5i}} \quad (20)$$

and

$$\frac{\partial\sigma_i}{\partial\rho_k} = \sum_{l=1}^5 \frac{\partial\sigma_i}{\partial\beta_{li}} \frac{\partial\beta_{li}}{\partial\rho_k} - \frac{\sigma_i}{n} \delta_{k,nthick} \quad (21)$$

where we have identified parameter number "nthick" as the sample thickness (n). The term $\delta_{k,nthick}$ is the Kronecker delta, equal to 1 if $k=nthick$ and equal to 0 otherwise.

The partial derivatives $\partial\beta_{li}/\partial\rho_k$ in Eq. (21) depend on the specific functional form chosen for the five functions β_{li} ; these will be discussed in Section 4.

For our example, these partial derivatives are

$$\frac{\partial\beta_{1i}}{\partial\rho_1} = \frac{\partial(M/m)}{\partial n} = 0; \quad \frac{\partial\beta_{1i}}{\partial\rho_2} = \frac{\partial(M/m)}{\partial m} = -\frac{M}{m^2};$$

$$\frac{\partial\beta_{1i}}{\partial\rho_3} = \frac{\partial(M/m)}{\partial M} = \frac{1}{m}; \quad \frac{\partial\beta_{1i}}{\partial\rho_4} = \frac{\partial(M/m)}{\partial g} = 0; \quad \frac{\partial\beta_{1i}}{\partial\rho_5} = \frac{\partial(M/m)}{\partial G} = 0;$$

$\frac{\partial \beta_{2i}}{\partial \rho_k} = \frac{\partial d_i}{\partial \rho_k} = 0$ since d_i is independent of all parameters;

$$\frac{\partial \beta_{3i}}{\partial \rho_1} = \frac{\partial g}{\partial n} = 0 ; \quad \frac{\partial \beta_{3i}}{\partial \rho_2} = \frac{\partial g}{\partial m} = 0 ;$$

$$\frac{\partial \beta_{3i}}{\partial \rho_3} = \frac{\partial g}{\partial M} = 0 ; \quad \frac{\partial \beta_{3i}}{\partial \rho_4} = \frac{\partial g}{\partial g} = 1 ; \quad \frac{\partial \beta_{3i}}{\partial \rho_5} = \frac{\partial g}{\partial G} = 0 ;$$

$\frac{\partial \beta_{4i}}{\partial \rho_k} = \frac{\partial D_i}{\partial \rho_k} = 0$ since D_i is independent of all parameters;

$$\frac{\partial \beta_{5i}}{\partial \rho_1} = \frac{\partial G}{\partial n} = 0 ; \quad \frac{\partial \beta_{5i}}{\partial \rho_2} = \frac{\partial G}{\partial m} = 0 ;$$

$$\frac{\partial \beta_{5i}}{\partial \rho_3} = \frac{\partial G}{\partial M} = 0 ; \quad \frac{\partial \beta_{5i}}{\partial \rho_4} = \frac{\partial G}{\partial g} = 0 ; \quad \frac{\partial \beta_{5i}}{\partial \rho_5} = \frac{\partial G}{\partial G} = 1 .$$

The partial derivatives $\partial \sigma_i / \partial \beta_{li}$ in Eq. (21) can be found directly from Eq. (17):

$$\frac{\partial \sigma_i}{\partial \beta_{1i}} = - \frac{1}{n \beta_{1i}}$$

$$\frac{\partial \sigma_i}{\partial \beta_{2i}} = - \frac{1}{n} \frac{c_i}{\beta_{2i} c_i - \beta_{3i}}$$

$$\frac{\partial \sigma_i}{\partial \beta_{3i}} = \frac{1}{n} \frac{1}{\beta_{2i} c_i - \beta_{3i}} \quad (22)$$

$$\frac{\partial \sigma_i}{\partial \beta_{4i}} = \frac{1}{n} \frac{C_i}{\beta_{4i} C_i - \beta_{5i}}$$

and

$$\frac{\partial \sigma_i}{\partial \beta_{5i}} = - \frac{1}{n} \frac{1}{\beta_{4i} C_i - \beta_{5i}} .$$

In our example, these derivatives reduce to

$$\begin{aligned}
 \frac{\partial \sigma_i}{\partial \beta_{1i}} &= - \frac{m}{nM} \\
 \frac{\partial \sigma_i}{\partial \beta_{2i}} &= - \frac{1}{n} \frac{c_i}{d_i c_i - g} \\
 \frac{\partial \sigma_i}{\partial \beta_{3i}} &= \frac{1}{n} \frac{1}{d_i c_i - g} \\
 \frac{\partial \sigma_i}{\partial \beta_{4i}} &= \frac{1}{n} \frac{C_i}{D_i C_i - G} \\
 \frac{\partial \sigma_i}{\partial \beta_{5i}} &= - \frac{1}{n} \frac{1}{D_i C_i - G}
 \end{aligned} \tag{23}$$

Combining Eqs. (18) through (21) and (23) then produces

$$\begin{aligned}
 \delta \sigma_i &= - \frac{m}{nM} \left[- \frac{M}{m^2} [\delta m] + \frac{1}{m} [\delta M] \right] - \frac{1}{n} \frac{c_i}{d_i c_i - g} [0] \\
 &\quad + \frac{1}{n} \frac{1}{d_i c_i - g} [\delta g] + \frac{1}{n} \frac{C_i}{D_i C_i - G} [0] \\
 &\quad - \frac{1}{n} \frac{1}{D_i C_i - G} [\delta G] - \frac{\sigma_i}{n} [\delta n] \\
 &\quad - \frac{1}{n} \frac{d_i [\delta c_i]}{d_i c_i - g} + \frac{1}{n} \frac{D_i [\delta C_i]}{D_i C_i - G}
 \end{aligned} \tag{24}$$

which is precisely the form obtained directly in Eq. (7).

To obtain the variance-covariance matrix for the corrected data (σ_i), we form the product of $\delta\sigma_i$ and $\delta\sigma_j$ and take expectation values:

$$\begin{aligned}
 \langle \delta\sigma_i \delta\sigma_j \rangle &= \sum_{k,k'=1}^K \frac{\partial\sigma_i}{\partial\rho_k} \langle \delta\rho_k \delta\rho_{k'} \rangle \frac{\partial\sigma_j}{\partial\rho_{k'}} \\
 &\quad + \frac{\partial\sigma_i}{\partial c_i} \langle \delta c_i \delta c_j \rangle \frac{\partial\sigma_j}{\partial c_j} + \frac{\partial\sigma_i}{\partial C_i} \langle \delta C_i \delta C_j \rangle \frac{\partial\sigma_j}{\partial C_j} \\
 &\quad + \sum_{k=1}^K \frac{\partial\sigma_i}{\partial\rho_k} \langle \delta\rho_k \delta c_j \rangle \frac{\partial\sigma_j}{\partial c_j} + \sum_{k'=1}^K \frac{\partial\sigma_i}{\partial c_i} \langle \delta c_i \delta\rho_{k'} \rangle \frac{\partial\sigma_j}{\partial\rho_{k'}} \\
 &\quad + \sum_{k=1}^K \frac{\partial\sigma_i}{\partial\rho_k} \langle \delta\rho_k \delta C_j \rangle \frac{\partial\sigma_j}{\partial C_j} + \sum_{k'=1}^K \frac{\partial\sigma_i}{\partial C_i} \langle \delta C_i \delta\rho_{k'} \rangle \frac{\partial\sigma_j}{\partial\rho_{k'}} \\
 &\quad + \frac{\partial\sigma_i}{\partial c_i} \langle \delta c_i \delta C_j \rangle \frac{\partial\sigma_j}{\partial C_j} + \frac{\partial\sigma_i}{\partial C_i} \langle \delta C_i \delta c_j \rangle \frac{\partial\sigma_j}{\partial c_j}.
 \end{aligned} \tag{25}$$

In Eq. (25) all cross-terms are explicitly displayed. However, the raw data (c or C) are measured independently of everything else, and each channel is measured independently of every other channel. Thus Eq. (25) reduces to

$$\langle \delta\sigma_i \delta\sigma_j \rangle = \sum_{k,k'=1}^K \frac{\partial\sigma_i}{\partial\rho_k} \langle \delta\rho_k \delta\rho_{k'} \rangle \frac{\partial\sigma_j}{\partial\rho_{k'}} + \left[\left(\frac{\partial\sigma_i}{\partial c_i} \right)^2 c_i + \left(\frac{\partial\sigma_i}{\partial C_i} \right)^2 C_i \right] \delta_{ij} \tag{26}$$

in which we have used

$$\langle \delta c_i^2 \rangle = c_i \text{ and } \langle \delta C_i^2 \rangle = C_i \tag{27}$$

and where δ_{ij} is the Kronecker delta function. Note that the cross terms between two different parameters ρ_k and $\rho_{k'}$ are not necessarily zero. (E.g. ρ_k and $\rho_{k'}$ may be two parameters which describe a certain type of background. The least-squares fitting procedure which determined values for those parameters also determined correlations between the parameters.) Note also that the cross sections for two channels i and j are in general correlated, due to the systematic uncertainties [i.e., due to the summations over parameters in Eq. (26)].

An additional complication is introduced by "rebinning" the data. That is, what is usually reported is not the cross section σ_i for channel i (where i typically ranges from 1 to $\sim 60,000$) but rather an average over channels. We define the summed cross section σ_I^{sum} as

$$\sigma_I^{sum} = \sum_{i=Imin(I)}^{Imax(I)} \sigma_i \Delta_i \quad (28)$$

where Δ_i is the width of channel i , and where $Imin(I)$ and $Imax(I)$ are the minimum and maximum channel number to be included in this bin. The average cross section $\bar{\sigma}_I$ is found by normalizing σ_I^{sum} , giving

$$\bar{\sigma}_I = \sigma_I^{sum} \left[\sum_{i=Imin(I)}^{Imax(I)} \Delta_i \right]^{-1} \quad . \quad (29)$$

The covariance matrix for σ^{sum} is found by summing Eq. (26) over i and j :

$$\langle \delta\sigma_I^{sum} \delta\sigma_J^{sum} \rangle = \sum_{i=Imin(I)}^{Imax(I)} \sum_{j=Imin(J)}^{Imax(J)} \Delta_i \langle \delta\sigma_i \delta\sigma_j \rangle \Delta_j \quad (30)$$

which gives

$$\begin{aligned} \langle \delta\sigma_I^{sum} \delta\sigma_J^{sum} \rangle &= \sum_{k,k'=1}^K \left(\sum_{i=Imin(I)}^{Imax(I)} \Delta_i \frac{\partial\sigma_i}{\partial\rho_k} \right) \langle \delta\rho_k \delta\rho_{k'} \rangle \left(\sum_{j=Imin(J)}^{Imax(J)} \Delta_j \frac{\partial\sigma_j}{\partial\rho_{k'}} \right) \\ &+ \sum_{i=Imin(I)}^{Imax(I)} \left[\left(\frac{\partial\sigma_i}{\partial C_i} \right)^2 c_i + \left(\frac{\partial\sigma_i}{\partial C_i} \right)^2 C_i \right] \Delta_i^2 \delta_{IJ} \quad . \end{aligned} \quad (31)$$

Note that the " δ_{IJ} " in Eq. (31) expresses the implicit assumption that bins I and J are non-overlapping.

Equation (31) may be greatly simplified by, first, noting from the definition given in Eq. (28) that the summations within parentheses are partial derivatives of the summed cross sections:

$$\sum_{i=I\min(I)}^{I\max(I)} \Delta_i \frac{\partial \sigma_i}{\partial \rho_k} = \frac{\partial \sigma_I^{\text{sum}}}{\partial \rho_k} . \quad (32)$$

The second simplification occurs when we define sample-in statistical uncertainty as

$$\Delta \sigma_I^{\text{sum,in}} = \left[\sum_{i=I\min(I)}^{I\max(I)} \left(\frac{\partial \sigma_i}{\partial c_i} \right)^2 c_i \Delta_i^2 \right]^{1/2} \quad (33)$$

and sample-out statistical uncertainty as

$$\Delta \sigma_I^{\text{sum,out}} = \left[\sum_{i=I\min(I)}^{I\max(I)} \left(\frac{\partial \sigma_i}{\partial C_i} \right)^2 C_i \Delta_i^2 \right]^{1/2} \quad (34)$$

With these identities, the covariance matrix for the summed cross section becomes

$$\begin{aligned} \langle \delta \sigma_I^{\text{sum}} \delta \sigma_J^{\text{sum}} \rangle &= \sum_{k,k'=1}^K \frac{\partial \sigma_I^{\text{sum}}}{\partial \rho_k} \langle \delta \sigma_k \delta \sigma_{k'} \rangle \frac{\partial \sigma_J^{\text{sum}}}{\partial \rho_{k'}} \\ &+ \left[(\Delta \sigma_I^{\text{sum,in}})^2 + (\Delta \sigma_I^{\text{sum,out}})^2 \right] \delta_{IJ} . \end{aligned} \quad (35)$$

The first term in Eq. (35) is generally referred to as the "systematic uncertainty," squared. That is, the total uncertainty $\Delta \sigma_I^{\text{sum}}$ is given by the sum, in quadrature, of the systematic uncertainty, the sample-in statistical uncertainty, and the sample-out statistical uncertainty. Moreover, the systematic uncertainty due to parameter k alone may be expressed as

$$\Delta \sigma_I^{\text{sum},k} = \frac{\partial \sigma_I^{\text{sum}}}{\partial \rho_k} \Delta \rho_k \quad (36)$$

where $\Delta \rho_k$ is the measured uncertainty on parameter k , given by

$$(\Delta\rho_k)^2 = \langle \delta\rho_k \delta\rho_k \rangle \quad . \quad (37)$$

Note that the total systematic uncertainty is *not*, in general, simply the sum in quadrature of the uncertainties due to the individual parameters but rather involves the covariance between the parameters.

Finally, to obtain the covariance matrix for the average cross section $\bar{\sigma}_I$, we simply multiply by the appropriate normalization factor N_I , given from Eq. (29) as

$$N_I = \left[\sum_{i=I_{\min}(I)}^{I_{\max}(I)} \Delta_i \right]^{-1} \quad . \quad (38)$$

The covariance matrix for the rebinned data is then given by

$$\begin{aligned} \langle \delta\bar{\sigma}_I \delta\bar{\sigma}_J \rangle &= \sum_{k,k'=1}^K \frac{\partial\bar{\sigma}_I}{\partial\rho_k} \langle \delta\rho_k \delta\rho_{k'} \rangle \frac{\partial\bar{\sigma}_J}{\partial\rho_{k'}} \\ &\quad + \left[(\Delta\sigma_I^{in})^2 + (\Delta\sigma_I^{out})^2 \right] \delta_{IJ} \end{aligned} \quad (39)$$

where the statistical uncertainties are given by

$$\Delta\bar{\sigma}_I^{in} = N_I \Delta\sigma_I^{sum,in} \quad (40)$$

and

$$\Delta\bar{\sigma}_I^{out} = N_I \Delta\sigma_I^{sum,out} \quad (41)$$

and the partial derivatives by

$$\frac{\partial\bar{\sigma}_I}{\partial\rho_k} = N_I \frac{\partial\sigma_I^{sum}}{\partial\rho_k} \quad . \quad (42)$$

Note that the statistical uncertainties contribute only to the diagonal part of the data covariance matrix, while the systematic uncertainties contribute both on- and off-diagonal.

Equation (39) gives the covariance matrix for the average cross section. Often it is useful to convert to the correlation matrix, defined as

$$C_{IJ} = \frac{\langle \delta\bar{\sigma}_I \ \delta\bar{\sigma}_J \rangle}{\Delta\bar{\sigma}_I \ \Delta\bar{\sigma}_J}, \quad (43)$$

where the uncertainty $\Delta\bar{\sigma}_I$ is the square root of the diagonal element of the covariance matrix,

$$(\Delta\bar{\sigma}_I)^2 = \langle \delta\bar{\sigma}_I \ \delta\bar{\sigma}_I \rangle. \quad (44)$$

3. ORGANIZATION OF ALEX

The evaluation of cross sections and corresponding covariance matrix elements from the raw data and correction/transformation information can be accomplished in three distinct steps:

- (a) Evaluate σ_i from Eqs. (1) and (14), and $\partial\sigma_i/\partial c_i$, $\partial\sigma_i/\partial C_i$, and $\partial\sigma_i/\partial\rho_k$ from Eqs. (19) through (21).
- (b) Evaluate the summed data σ_i^{sum} from Eq. (28), and the derivatives $\partial\sigma_i^{sum}/\partial\rho_k$ from Eq. (32).
- (c) Evaluate the covariance matrix elements $\langle \delta\bar{\sigma}_I \delta\bar{\sigma}_J \rangle$ from Eq. (39) et al.; this matrix constitutes the final results.

The code ALEX is organized in such a way as to take maximum advantage of this three-step nature. ALEX is divided into six parts (i.e., six separate files on the ORELA PDP-10 computer), each of which is semiautonomous. Five of these six parts are described briefly in this section, and the sixth (which deals with information that varies from case to case) is discussed in some detail in Section 4. Again, the extension of ALEX to other types of data reduction is straightforward due to the modular structure of the program.

3.1 BOOKKEEPER ROUTINES

The main program in ALEX may be viewed as an overseer which controls operations but which performs no work itself. Three distinct operations are supervised by the main program. The first is allocation of array storage as it is needed. The second is interpretation of user-supplied information regarding parameter and data storage. The third is the actual calculation of the cross section and covariance matrix.

In order to minimize storage requirements in ALEX, space for many of the needed arrays is allocated dynamically as needed, and released when no longer needed. Allocation/deallocation is accomplished in functions IDIMEN (for real arrays, with storage taken from container array A) and JDIMEN (for integer arrays, using container array IA). Reallocation is accomplished in subroutine MOVE.

Subroutines relating to user-supplied information for experimental conditions are described fully in Section 4 of this report. Subroutine FIXPAR reorganizes experimental parameters into a form convenient for ALEX, and subroutine OUTPAR lists these parameters. Subroutines FILENM and START prompt the user for information regarding data storage (filenames) and the existence or nonexistence of intermediate results from earlier ALEX runs.

The main program and the subprograms indicated by name above are stored in file ALEX.FOR on the ORELA PDP-10 computer, project-programmer number [100,1006]. Users not on that computer should contact the Radiation Shielding Information Center (RA82) for access to this and other files. A FORTRAN listing of this file appears in Appendix A of this report.

3.2 ROUTINES TO EVALUATE σ_i , $\frac{\partial\sigma_i}{\partial c_i}$, $\frac{\partial\sigma_i}{\partial C_i}$, $\frac{\partial\sigma_i}{\partial \rho_k}$

Step (a) above was to evaluate the corrected/transformed cross section σ_i from Eqs. (1) and (14), and the partial derivatives $\partial\sigma_i/\partial c_i$, $\partial\sigma_i/\partial C_i$, and $\partial\sigma_i/\partial \rho_k$ from Eqs. (19) through (21). PDP-10 file ALXFIX.FOR contains many of the subroutines needed to accomplish this step. [The remaining subroutines for step (a) are those which vary from one experiment to the next; these are described in detail in Section 4.] File ALXFIX.FOR is listed in Appendix B of this report.

The first four routines (ZERØ, ZERØ2, SORT, and RESORT) in ALXFIX.FOR serve to interface between the user-supplied information (see Section 4) and the code ALEX.

Subroutine SIGMAC calculates the cross sections and partial derivatives, using values for β_{li} determined in FUNCTN, derivatives of β_{li} determined in DDERIV, and raw data located by subroutine GETRAW.

For each channel i , there are $3+NPAR$ quantities to generate, where NPAR is the number of parameters. All $NCHN \times (3+NPAR)$, where NCHN is the number of channels, of these quantities are stored in the ØDF file whose default name is ALXFIX.ØDF; the characters "FIX" imply that this file holds the corrected or "fixed" data. Subroutine PUTFIX generates this intermediate storage file. See Table 1 for a description of this and other data files required by ALEX.

3.3 EVALUATE σ_i^{sum} AND ITS PARTIAL DERIVATIVES

File ALXSUM.FOR holds the subroutines needed to evaluate the summed data σ_I from Eq. (28) and the derivatives $\partial\sigma_I^{sum}/\partial \rho_k$ from Eq. (32) [step (b) above]. Subroutine EXTRA is the bookkeeper for this step, calling the appropriate subroutine ESUM or CHNSUM to perform the summation over channels i for each bin I. Two such subroutines are needed, since in one case (ESUM) the bins are specified by energy limits and in the other (CHNSUM) the bins are specified by channel limits. Subroutine GETFIX locates channel limits, cross sections, and partial derivatives for use in ESUM; GETFXK performs the analogous function for CHNSUM.

Table 1. Summary of Data Files Required by ALEX

Unit Number	Unit Name, If Any	Default Filename If Any	Input, Intermediate, Or Output*	Subroutine In Which This File Is Read	Subroutine In Which This File Is Written	for ØDF Files		What's Stored In That File Or That Section of ØDF File
						Number of Channels	Section Number	
24	ALEXIN.DAT	input	INPUT, SETPAR					See Table 2.
23	IURAW ALXRAW.ØDF	input	GETRAW, GETRF, GETRFK	NCHN (~60,000)	1 2	Channel energy at mid-channel d = deadtime correction factor, sample-in		
25	ALXAUX.DAT	input	COUNT, EXTRA2		3 4 5	D = deadtime correction factor, sample-out c = raw counts, sample-in C = raw counts, sample-out		
22	IUFIX ALXFIX.ØDF	intermediate	GETFIX, GETFXK, GETRF, GETRFK	PUTFIX NCHNMX-NCHNMN (~60,000)	1 2 3 4 4+k	channel energy at end of channel cross section $\partial\sigma/\partial c$ = partial derivative of cross section with respect to sample-in counts $\partial\sigma/\partial C$ = partial derivative of σ w.r.t. sample-out counts $\partial\sigma/\partial\rho_k$ = partial derivative of σ w.r.t. parameter number k		
26	IUSUM ALXSUM.ØDF	intermediate	GETSUM	PUTSUM NBIN ($\leq 200^2$)	1 2 3 3+k 4+NPAR 5+NPAR 6+NPAR 7+NPAR	RH = upper energy limit to energy region RL = lower limit $\bar{\sigma}$ = average cross section for this region $\partial\bar{\sigma}/\partial\rho_k$ = partial derivative of σ w.r.t. parameter number k FRACH = fraction of channel KH contributing to this region FRACL = fraction of channel KL contributing to this region KH = smallest channel number in this energy region KL = largest channel number in this energy region		

Table 1. (Continued)

Unit Number	Default Filename If Any	Input, Intermediate, Or Output* If Any	Subroutine In Which This File Is Read	Subroutine In Which This File Is Written	for ϕ DF Files		What's Stored In That File Or That Section of ϕ DF File
					Number of Channels	Number of Section Number	
21	IU ϕ UT	ALX ϕ UT. ϕ DF	output	PUT ϕ UT	NBIN (≤ 200)	1 2	$\bar{\sigma}$ $\Delta\bar{\sigma}$ sample-in statistics $\Delta\bar{\sigma}$ sample-out statistics $\Delta\bar{\sigma}$ due to parameter k $\Delta\bar{\sigma}$ total
21	IU ϕ UT	ALX ϕ UT. ϕ DF	output	PUT ϕ K	NBIN ($\geq 10,000$)	1 2 3 4 5 6 7 8	energy (mid point between RH and RL) $\bar{\sigma}$ $\Delta\bar{\sigma}$ statistical $\Delta\bar{\sigma}$ systematic $\Delta\bar{\sigma}$ total % error, statistical % error, systematic % error, total
32		ALX ϕ UT.LPT	output	INPUT, SEIPAR, ϕ UTPD, ϕ UTALL, ϕ UTVX, ϕ UTV, ϕ UTSTR, SIGMAC, GETDAT			line printer output sample-in, sample-out, and total covariance matrices for debug
31		F ϕ R31.DAT	output	TALK			debug information
33		F ϕ R33.DAT	input	TALKK			teletype I/ ϕ
5			input, output	FILENM, START			

*"Intermediate" files are written in one part of ALEX and read in another. On the initial run of ALEX, all intermediate files are both written and read; on subsequent runs for the same experiment, an "intermediate" file may serve as "input" to avoid repeating calculations already completed.

If NBIN represents the number of bins, then there are $NBIN \times (1 + NPAR)$ quantities (cross section and partial derivatives) to be generated and stored. Subroutine PUTSUM stores these quantities in the QDF file whose default name is ALXSUM.QDF; the characters "SUM" indicate that quantities stored here are summed over channels to produce "bins". If steps (a) and (b) are to be skipped, subroutine GETSUM recalls these quantities from the storage file.

Subroutine QUTPD writes the summed cross section [Eq. (28)] and partial derivatives [Eq. (32)] onto the line-printer file whose default name is ALXQUT.LPT.

File ALXSUM.FOR is listed in Appendix C.

3.4 EVALUATE COVARIANCE MATRIX

File ALXALL.FOR (listed in Appendix D) stores the subroutines needed to evaluate and report the covariance matrix elements [step (c) above]. The complete covariance matrix for the summed data [i.e., Eq. (31)] is generated in subroutine FIXCQV, which calls auxiliary subroutine GETRF to locate raw data c_i and C_i and partial derivatives $\partial\sigma_i/\partial c_i$ and $\partial\sigma_i/\partial C_i$. FIXCQV also calls subroutines TALK and TALKK to output debug information into files FOR33.DAT and FOR31.DAT.

Subroutine FIXDER (with help from SETD) evaluates the several components of the covariance matrix, i.e., the systematic uncertainty on the summed data due to each parameter, the sample-in statistical uncertainty, the sample-out statistical uncertainty, and the total uncertainty. If the user so specifies, these components are reported in subroutine QUTBUG, which calls QUTALL, QUTV, and QUTSTR.

Normalization from summed cross section σ^{sum} to averaged or rebinned cross section $\bar{\sigma}$ is accomplished in subroutine DIVIDE. Results are reported by subroutine QUTFIN, which calls PUTQOUT, QUTALL, and QUTVX. Subroutine PUTQOUT generates the output QDF file whose default name is ALXQOUT.QDF, which stores the components of the uncertainty and which may, for example, be used to make plots comparing the contributions of two parameters. Subroutine QUTALL writes the components of the uncertainty onto the line-printer file ALXQOUT.LPT; subroutine QUTVX does the same for the covariance matrix.

For NBIN equal to the number of bins desired, the size of the covariance matrix is $NBIN \times (NBIN+1)/2$. [Actually, the size is $NBIN \times NBIN$, but because this is a symmetric matrix, only half of the off-diagonal elements need to be reported.] For large values of NBIN, say $NBIN > 200$, it is not practical to report all these numbers. What is done in that case is discussed in the next subsection.

3.5 EVALUATE UNCERTAINTY WHEN THE NUMBER OF BINS IS LARGE

In Subsection 3.3 we noted that $NBIN \times (1 + NPAR)$ quantities (σ^{sum} and $\partial\sigma^{sum}/\partial\rho_k$) need to be evaluated and stored. In Subsection 3.4, $NBIN \times (5 + NPAR)$ quantities ($\bar{\sigma}$, $\Delta\bar{\sigma}$, and the various components of $\Delta\bar{\sigma}$) must be evaluated and stored, and an additional $NBIN \times (NBIN+1)/2$ quantities evaluated and reported. For large NBIN, say $NBIN > 1000$, the computer rapidly runs out of storage space, and an alternative procedure is used.

For NBIN large, steps (b) and (c) are combined and results from step (b) are not stored but are discarded after use. Only the statistical uncertainty, the systematic uncertainty, and the total uncertainty on $\bar{\sigma}$, are evaluated. That is, the systematic uncertainty due to an individual parameter is not reported, and the statistical uncertainty is not broken up into sample-in and sample-out contributions. Off-diagonal elements of the covariance matrix for $\bar{\sigma}$ are not generated.

File ALX1OK.FOR (Appendix E) holds FORTRAN listings of the subroutines which generate $\bar{\sigma}$, $\Delta\bar{\sigma}^{stat}$, $\Delta\bar{\sigma}^{syst}$, and $\Delta\bar{\sigma}$. COUNT merely counts channels to determine array sizes. Subroutine EXTRA2 generates the cross sections and uncertainties, calling ZER03 to initialize arrays, GETRFK to obtain raw data and "fixed" data from step (a), and PUT0K to complete the calculation and to output final results onto file ALXOUT.ODF. Uncertainties are given both as absolute errors and as percent errors.

4. SPECIFYING THE EXPERIMENTAL SITUATION FOR ALEX

Routines which specify details about the experimental situation must, of course, be modified for each specific case. For that reason these routines will be discussed in some detail here. A FORTRAN listing of these five routines is kept in file ALXPAR.FOR on the ORELA PDP-10 and is reproduced in Appendix F of this report.

4.1 SUBROUTINE INPUT

Subroutine INPUT furnishes two basic types of information: constants which exactly describe the experimental situation (e.g., flight path length, channel widths), and indicators for the manner in which results are to be reported (e.g., energy intervals for summing cross sections). In its present form, subroutine INPUT reads this information from unit number 24 (whose unit name is specified by the user in response to ALEX's prompt "What is filename for input information?"). Variable names, meanings, and formats for this file are shown in Table 2.

4.2 SUBROUTINE SETPAR

Subroutine SETPAR sets the values of the parameters $\{\rho_k\}$ and of the parameter covariance matrix elements $\{<\delta\rho_k\delta\rho_{k'}>\}$. Information used to specify these arrays is also obtained from the input file (unit 24); see Table 2 for details. Because it is symmetric, the covariance matrix is stored in "packed" form via

$$<\delta\rho_i\delta\rho_j> = <\delta\rho_j\delta\rho_i> = COV(IJ)$$

where the index IJ is given by

$$IJ = (I * (I-1))/2 + J$$

and I is the greater of i and j , J is the lesser.

In our example, the five parameters are $\rho_1 = n$ = thickness, $\rho_2 = m$ = monitor counts for sample-in, $\rho_3 = M$ = monitor counts for sample-out, and $\rho_4 = g$ = background for sample-in, and $\rho_5 = G$ = background for sample-out. Since values of all five are determined independently, off-diagonal components of the covariance matrix are zero, card group number 16 in Tables 1 and 2 has no entries. The covariance matrix generated in SETPAR has the form

$COV(1) = (\Delta n)^2$				
$COV(2) = 0$	$COV(3) = (\Delta m)^2$			
$COV(4) = 0$	$COV(5) = 0$	$COV(6) = (\Delta M)^2$		
$COV(7) = 0$	$COV(8) = 0$	$COV(9) = 0$	$COV(10) = (\Delta g)^2$	
$COV(11) = 0$	$COV(12) = 0$	$COV(13) = 0$	$COV(14) = 0$	$COV(15) = (\Delta G)^2$

Table 2. Description of Information Specified on the Input File

Card Group Number	How Many Cards In This Group	Variable Name	Range of Values or Approximate Value	Default Value	Units	Format (Column Numbers)	Meaning
1	1	FILEQ				3A10 (1-30)	Name of file containing the raw counts and deadline correction.
		NCHNMX	≥ 0	(number of channels in FILEQ)		I10 (31-40)	Largest channel number to be used.
		NCHNMN	≥ 0	1		I10 (41-50)	Smallest channel number to be used.
2	1	(blank)					
3	1	TITLE	'TIME DELAY'			A10 (1-10)	The next card stores "time delay" and "flight-path length".
4	1	TDELAY	$\sim 800?$ ≥ 0		ns	F	Time delay.
5	1	FPL	$\sim 200000.$		mm	F	Flight-path length.
6	1	(blank)					
		TITLE	'CRUNCH B&U'			A10 (1-10)	The next cards give "crunch boundary" information.
7	NBINSZ (i.e., as many as needed)	M(K)	≥ 1		I		Number of channels in region K.
		C(K)	$1 \leq C \leq 1000?$		ns	F	Time-width of each channel in this region.
8	1	(blank)					

Table 2. Description of Information Specified on the Input File (contd)

Card Group Number	How Many Cards In This Group	Variable Name	Range of Values or Approximate Value	Default Value	Units	Format (Column Numbers)	Meaning
9	1	TITLE	'ENERGY INT'			A10 (1-10)	The next cards contain "energy intervals" for reporting results.
first alternative	WHATEV	'ME'				A2 (19-20)	Units for RL and RH (see card 10) are MeV.
		'KE'					Units are keV.
10	NENERG (i.e., as many as needed)	RL(K)	$\geq R_{H}(K-1)$				Lower limit for Kth energy interval.
		RH(K)	$\geq R_{L}(K)$	RL(K+1)	F		
9	1	TITLE	'CHANNEL IN'			A10 (1-10)	Upper (higher) limit for Kth energy interval.
		KMIN	≥ 1	I			The next cards contain channel intervals for reporting results.
10	as many as needed	KMAX	$\geq K_{MIN}$	I			First channel number.
		INC	$1 \leq INC \leq (K_{MAX}-K_{MIN}+1)$	1	I		Between channel KMIN and KMAX, INC channels will be combined into a single bin. The number of bins thus formed is $(K_{MAX}-K_{MIN}+1)/INC$.

Table 2. Description of Information Specified on the Input File (contd)

Card Group Number	How Many Cards In This Group	Variable Name	Range of Values or Approximate Value	Default Value	Units	Format (Column Numbers)	Meaning
9 third alternative	1	TITLE	'CHANNEL BΦ			A10 (1-10)	Channel boundaries and increments for rebinning are read on the auxiliary file (FILEC); only diagonal elements of covariance matrix will be generated.
10	MISSING!		— — — — —			A10 (52-61)	
11	1	(blank)					Values and uncertainties for parameters are on next cards.
12	1	TITLE	'PARAMETERS'				
13	NPAR (as many as needed)	INPAR(J)	1≤INPAR(J)≤NPAR	I			Index to count parameters; if set to zero, ALEX will ignore this parameter.
		PAR(J)		F			Value of parameter number J.
		DELPAR(J)	ABS(PAR(J)) *PERCNT(J)*100.	F			(Absolute) uncertainty on this parameter number J.
		PERCNT(J)		F			Percent uncertainty on this parameter.
		NAMPAR(J)				8A5(28-67)	Alphanumeric name for parameter number J.
14	1	(blank)					

Table 2. Description of Information Specified on the Input File (contd)

Card Group Number	How Many Cards In This Group	Variable Name	Range of Values or Approximate Value	Default Value	Units	Format (Column Numbers)	Meaning
15	1	TITLE	'CΦVARIANCE'			A10 (1-10)	Off-diagonal "covariance" matrix elements for parameters are given in the following cards.
16	as many as needed	IPAR	1≤IPAR≤NPAR			I	Parameter number.
		JPAR	1≤JPAR≤NPAR			I	Parameter number.
		CΦV	CΦV ≤ DELPAR(IPAR) *DELPAR(JPAR)	DELPAR(IPAR)* CΦRR*DELPAR(JPAR)		F	Covariance between parameters IPAR and JPAR.
		CΦRR	-1≤CΦRR≤1			F	Correlation between parameters IPAR and JPAR.
17	1	(blank)					

In some experiments additional information is needed in order to fully specify the situation. For example, in the natural nickel evaluation (DL83) one background had a shape which was linear at low times, proportional to $1/t^2$ for long time, and constant at intermediate times. Boundaries between these regions were specified in the input file; subroutine SETPAR called subroutine JUNK to read these values before reading parameter values.

4.3 SUBROUTINE PREP

Subroutine PREP is perhaps the most complicated of those routines which must be modified for each specific case. In PREP, the various corrections which will be applied to the raw data are described in a manner which code ALEX can interpret and use.

First, the total number NFUNC of correction functions is given, and the functions are numbered consecutively from 1 to NFUNC. Note that NFUNC need not be equal to 5, since the correction functions specified here are *not* the β_i given in Eqs. (14) and (15); rather, the β_i will be formed from these functions.

- (1) If the normalization is unity, it need not be listed with the correction functions. In this case, ALEX sets $\beta_1 = 1$.
- (2) Similarly for deadtime corrections, sample-in or sample-out, ALEX sets β_2 or β_4 equal to unity if they are not otherwise specified.
- (3) There may be several contributions to the background (e.g., a time-dependent background and a time-independent background), each of which may be counted separately.

For each correction function, the TYPE is specified as 'NORMALIZATION', 'SAMPLE IN' ; or 'SAMPLE OUT', and the USE is specified as 'DEADTIME Correction' or 'BACKGROUND'. Only those characters shown here in capital letters (the first ten characters) need actually be given.

For each correction function, the parameters which affect that function must be specified. This is accomplished by setting IDEP(1) equal to the number of a relevant parameter, IDEP(2) equal to the number of another relevant parameter, etc., where the "numbers" are as specified in subroutine SETPAR (i.e., in the input file).

Subroutine PREP specifies which parameter is sample thickness by setting NTHICK equal to that parameter number. This is necessary because sample thickness is treated differently from other parameters [see Eq. (21)], since it is involved in the transformation from transmission to cross section.

In our example NFUNC=5, and there is exactly one of each kind of function. The normalization function is given by M/m , so for function number 1 PREP sets TYPE = 'NORMALIZAT' (USE is irrelevant), IDEP(1) = 2, and IDEP (2) = 3. Function number 2 is the sample-in deadtime correction, for which TYPE='SAMPLE-IN', USE = 'DEADTIME C', and IDEP need not be set. Function number 3 is the sample-out deadtime correction with TYPE = 'SAMPLE-OUT' and USE = 'DEADTIME C'. Function number 4 (or 5) is the background correction with TYPE = 'SAMPLE-IN' ('SAMPLE-OUT'), USE = 'BACKGROUND', and IDEP(1) = 4 (5). Note that the *order* in which correction functions are numbered is irrelevant; what is important is that the numbering be consistent between this routine PREP and the two other subroutines FUNC and DERIV.

4.4 FUNCTION FUNC(IFUNC)

Values for the functions described in subroutine PREP are generated in function FUNC(IFUNC). The argument IFUNC is the indexing number of the function whose value is to be determined on this call to FUNC; this numbering system must be identical to that used in PREP.

Quantities such as raw data CIN and COUT (sample-in counts c_i and sample-out counts C_i), deadtime correction factors DTCIN and DTCOUT, channel energy ECHN, channel time TCHN, channel time width CHN, and channel number ICHN, any of which might be required for generating FUNC for a given channel are evaluated elsewhere (in subroutine GETRAW) and sent to FUNC via COMMON/CHANL/.

In our example, IFUNC = 1 would cause evaluation of the normalization $M/m = \text{PAR}(3)/\text{PAR}(2)$. For IFUNC = 2, FUNC is set equal to the sample-in deadtime correction factor. For IFUNC = 3, FUNC is set to the sample-out deadtime correction factor. For IFUNC = 4, FUNC is set to $g = \text{PAR}(4)$, and for IFUNC = 5, FUNC is set to $G = \text{PAR}(5)$.

4.5 FUNCTION DERIV(IFUNC,KPAR)

The partial derivative of the i th function with respect to the k th parameter ($\partial f_i / \partial p_k$) is evaluated in DERIV, where $i = \text{IFUNC}$ and $k = \text{KPAR}$. The numbering system for the functions must be consistent with that described in PREP and used in FUNC, and the numbering system for the parameters must be consistent with that given in SETPAR. Derivatives need be specified only when they are not identically zero; that is, DERIV will be called only for parameters KPAR which are relevant to function IFUNC (as indicated by array IDEP in subroutine PREP).

The required derivatives for our example are summarized in the table below. Only these derivatives are actually given in function DERIV; ALEX automatically assumes the others are zero.

IFUNC	KPAR	DERIV
1	2	$-\text{PAR}(3)/\text{PAR}(2)^{**2}$
1	3	$1.0/\text{PAR}(2)$
4	4	1.0
5	5	1.0

4.6 SUBROUTINES WHICH READ RAW DATA

Generally there will be several tens of thousands of channels of raw data. Only a few (currently 512) of these data may be held in core simultaneously, necessitating considerable I/O to and from storage files. Because the raw data are required at several different points in the calculation (i.e., several different subroutines in ALEX), it seemed expedient to assume a standard form for the storage file and to have program ALEX deal with only that standard form. It is likely easier to produce such a standard storage file for each application (writing an auxiliary FORTRAN code to do so if needed) than to program ALEX for the general case.

The standard form for the storage file is in ORELA Data Format (\emptyset DF) with Section 1 containing the channel energy, Section 2 the deadtime correction factor for sample-in, Section 3 the deadtime correction factor for sample-out, Section 4 the raw counts for sample-in, and Section 5 the raw counts for sample-out. Subroutines which read this "raw data" file are GETRAW, GETRF, and GETRFK, which are included in files ALXFIX.F \emptyset R, ALXALL.F \emptyset R, and ALX10K.F \emptyset R, respectively.

5. EXAMPLE

The "simple case" discussed in Section 2 of this report is in fact a description of the experimental situation for the natural iron measurement, from 2 to 80 MeV, of D. C. Larson et al. (DL80). This measurement will, therefore, serve to illustrate the manner in which ALEX is to be used; a person wishing to use ALEX for analysis of another experiment can simply repeat the steps outlined here, substituting his own specifics.

For guidance on choosing parameter values and covariances in a more complicated experimental situation, the reader is referred to the natural nickel uncertainty analysis (DL83).

Step 1. Organize Raw Data

An \emptyset DF file FERAW. \emptyset DF is generated, consisting of 10000 channels and 5 sections. Section 1 contains energy, Sections 2 and 3 hold dead time correction factors, and Sections 4 and 5 hold raw counts (see Table 1, unit 23). For this particular case only channels 938 through 4356 contain meaningful information.

Step 2. Prepare Subroutines INPUT, SETPAR, PREP, FUNC, and DERIV

For this measurement, these subroutines are exactly as given in Appendix F. For other experiments, review Section 4 of this report to learn what these routines must do.

Step 3. Prepare Input File

Since the only information to be read from the input file is from the five subroutines specified in Step 2 and these have not been modified, the input description given in Table 2 is correct for this data. The input file FE.DAT for the natural iron measurement is listed in Table 3.

Values for all quantities in the input file are provided by the experimenter; that is, the time delay, flight path length, sample thickness, monitor counts, and constant backgrounds were determined in conjunction with the measurement of the raw data. Uncertainties on sample thickness, monitor counts, and background levels either are determined or may be guesstimated. The "energy intervals for averaging" were, for this example, chosen arbitrarily. Note that any one bin (energy interval) covers many channels.

Table 3. Input File FE.DAT for the Fe Example of Section 2**FERAW.ODF****4356****938****TIME DELAY, FLIGHT PATH LENGTH**
249.,80187.**CRUNCH BOUNDARIES (NUMBER OF CHANNELS, WIDTHS IN NANOSEC)**
6000,1.
2000,4.
1000,16.
1000,1000.**ENERGY INTERVALS (KEV) FOR AVERAGING**2000.,2400.
2400.,3000.
3000.,4500.
4500.,8000.
8000.,11000.
11000.,25000.
25000.,50000.
50000.,80000.**PARAMETERS AND UNCERTAINTIES**1,0.4296,0.,0.2, **THICKNESS OF SAMPLE**
2,8770139.,0.,0.5, **MONITOR FOR SAMPLE IN**
3,7060770.,0.,0.5, **MONITOR FOR SAMPLE OUT**
4,2.7422,0.,5., **BACKGROUNDFOR SAMPLE IN**
5,2.5587,0.,5., **BACKGROUNDFOR SAMPLE OUT****COVARIANCES FOR PARAMETERS**

Step 4. Execute ALEX

To run the code ALEX on the ORELA PDP-10 computer, one simply types

EX @ ALEX↓

where ALEX.CMD is a command file which will be described below. ALEX then prompts the user for the information it requires. This includes file names, decisions regarding debug options, and specifics about which (if any) of the intermediate files have been created by earlier ALEX runs. See Table 1 in Section 3 for a description of the various files. Table 4 shows ALEX prompts and user responses for this example, assuming no intermediate files have been created.

The command file ALEX.CMD contains, on one line, the names of all files which together make up the code ALEX, plus the library file in which the ØDF subroutines (ØDFIØ, INØDF, ØUTØDF) are stored. For this example, ALEX.CMD reads

ALEX, ALXPAR, ALXFIX, ALXSUM, ALXALL, ALX10K, @ØRELIB.F10 [101,1010].

Step 5. Study the Output

The line-printer output for our example is given in Table 5. This output should be carefully examined to ensure accuracy of results:

1. Is the input correct? Check that all input quantities are correct in the units specified; i.e., that time delay is in ns, flight path length in mm, and crunch boundaries in ns for this example. Note that energy intervals are converted to MeV before being printed.
2. Are parameters and uncertainties what the user had intended? Be sure decimal points are in the appropriate positions for example.
3. Summed cross sections [which correspond to Eq. (28)] should agree with the user's intuition. Intuition may not be a reasonable guide for the partial derivatives [which correspond to Eq. (32)], but may indicate whether the signs of the derivatives are reasonable.
4. The average cross sections [Eq. (29)] should correspond even more closely with the user's intuition. The total uncertainty, sample-in and sample-out statistical uncertainty, and the systematic uncertainty due to each individual parameter can be examined to determine which are the more important contributions. Note that, in general, the total uncertainty will not be merely the summation in quadrature of the various components, since correlations between the parameters must also be incorporated. In this example, however, the parameters are given as uncorrelated.
5. The final covariance matrix for the average cross sections is printed as percent standard deviation ($100 \times \text{SIGMA} \div \text{UNCERT.}$) plus the correlation matrix [see Eq. (43)]. (The correlation matrix elements have been multiplied by 100 for ease of displaying.) Note the large off-diagonal elements. This is due to the large number of channels (hence high number of counts and low statistical uncertainties) in the energy intervals chosen for display. Uncertainties due to count rates (i.e., statistics) are strictly on the diagonal and are low here. Hence the systematic contributions, which are both on- and off-diagonal, assume relatively greater importance.
6. Plots can be made from the various input, intermediate, and final ØDF files using program FØRØDF (JC78) as further tests.

Table 4. ALEX prompts and user responses for the natural iron analysis. Underlined portions are typed by the user; a downward arrow indicates carriage return.

```
.EX @ ALEX↓
-----
LINK:    LOADING
[LNKXCT ALEX EXECUTION]
WHAT IS FILENAME FOR INPUT INFORMATION? FE.DAT↓
-----
WHAT IS FILENAME FOR UNSUMMED CROSS SECTION AND DERIVATIVES? FEUNSM.ODF↓
-----
WHAT IS FILENAME FOR SUMMED CROSS SECTION AND DERIVATIVES? FESUM.ODF↓
-----
WHAT IS FILENAME FOR LINE PRINTER OUTPUT? FE.LPT↓
-----
DO YOU WANT DEBUG PRINTOUT? N↓
-
DOES DERIVATIVE FILE ALREADY EXIST? N↓
-
```

```
*****
UNSUMMED DERIVATIVE FILE CREATED BY THIS RUN IS CALLED FEUNSM.ODF
SUMMED DERIVATIVE FILE CREATED BY THIS RUN IS CALLED FESUM.ODF
*****
```

```
WHAT IS NAME FOR ODF OUTPUT FILE? FEOUT.ODF↓
-----
```

```
MX, MANY=    7  512
KOUNT,MANYX=  1  512
KOUNT,MANYX=  2  512
KOUNT,MANYX=  3  512
KOUNT,MANYX=  4  512
KOUNT,MANYX=  5  512
KOUNT,MANYX=  6  512
KOUNT,MANYX=  7  512
```

```
REAL ARRAY SIZE USED FOR ALEX IS 512
INTEGER ARRAY SIZE USED FOR ALEX IS 401
STOP
```

```
END OF EXECUTION
CPU TIME: 31.75 ELAPSED TIME: 4:8.53
EXIT
```

Table 5. Line-Printer Output for Natural Iron Analysis

*****INPUT DATA IS FOUND ON FILE "FE.DAT

" *****

NCHN, NCHNMN= 4356 938
 TIME DELAY (NS) AND FLIGHT PATH LENGTH (MM) ARE 249.000 AND 80187.0

CRUNCH BOUNDARIES AND CHANNEL WIDTHS IN NANoseconds

6000	1.00000
2000	4.00000
1000	16.00000
1000	1000.00000

ENERGY INTERVALS IN MEV

1	2.000000	2.400000
2	2.400000	3.000000
3	3.000000	4.500000
4	4.500000	8.000000
5	8.000000	11.000000
6	11.000000	25.000000
7	25.000000	50.000000
8	50.000000	80.000000

PAR USED PARAMETER UNCERTAINTY % UNCERTAINTY
 NO. ???

1	1	0.429600	0.000859	0.200	THICKNESS OF SAMPLE
2	2	8770139.00000	43850.694800	0.500	MONITOR FOR SAMPLE IN
3	3	7060770.00000	35303.850100	0.500	MONITOR FOR SAMPLE OUT
4	4	2.742200	0.137110	5.000	BACKGROUNDFOR SAMPLE IN
5	5	2.558700	0.127935	5.000	BACKGROUNDFOR SAMPLE OUT

Table 5. (Continued)

*****CORRELATION MATRIX FOR PARAMETERS
OFF-DIAGONAL ELEMENTS OF COVARIANCE MATRIX ARE ALL ZERO.

*** STANDARD DEVIATION (SORT OF DIAGONAL ELEMENTS)

	STD. DEV.	STD. DEV.	STD. DEV.	STD. DEV.
(1)	8.5920E-04	(2)	4.3851E+04	(3)

UNSUMMED DERIVATIVE FILE CREATED BY THIS RUN IS CALLED FEUNSM.ODF

SUMMED DERIVATIVE FILE CREATED BY THIS RUN IS CALLED FESUM.0DF

SUMMED CROSS SECTION AND PARTIAL DERIVATIVES

EL	ER	SIGMA	DERIVATIVES AND INTEGRALS	1	2	3	4	5
			THICKNESS OF SAMPLE	MONITOR FOR SAMPLE	MONITOR FOR SAMPLE	BACKGROUND FOR SAMPLE	BACKGROUND FOR SAMPLE	BACKGROUND FOR SAMPLE
				IN	OUT	IN	IN	OUT
1	2.00000	2.40000	1.259	-2.930	1.0617E-07	-1.3187E-07	8.5723E-03	-2.6101E-03
2	2.40000	3.00000	2.055	-4.783	1.5925E-07	-1.9780E-07	1.1285E-02	-3.2023E-03
3	3.00000	4.50000	5.296	-12.33	3.9813E-07	-4.9451E-07	1.4802E-02	-4.0808E-03
4	4.50000	8.00000	12.55	-29.21	9.2896E-07	-1.1539E-06	6.5292E-03	-1.7284E-03
5	8.00000	11.00000	9.364	-21.80	7.9625E-07	-9.8902E-07	2.3552E-03	-7.5764E-04
6	11.00000	25.00000	33.31	-77.53	3.7158E-06	-4.6154E-06	6.2382E-03	-2.7684E-03
7	25.00000	50.00000	60.06	-139.8	6.6354E-06	-8.2418E-06	1.7534E-02	-7.6742E-02
8	50.00000	79.80661	69.85	-162.6	7.9112E-06	-9.8264E-06	0.1067	-4.9824E-01

Table 5. (Continued)

***** CROSS SECTIONS AND CONTRIBUTIONS TO UNCERTAINTIES

EL	EH	SIGMA	UNCERT.	(S-I)	(S-O)	DERIVATIVES WRT PAR,							
						1	2	3					
						THICKNESS	MONITOR	MONITOR					
						OF SAMPLE	FOR SAMPLE	FOR SAMPLE					
						IN	OUT	OUT					
1	2.00000	2.40000	3.14731	0.02253	0.01203	0.00657	6.2946E-03	1.1639E-02					
2	2.40000	3.00000	3.42435	0.02187	0.01090	0.00587	6.8487E-03	1.1639E-02					
3	3.00000	4.50000	3.53077	0.01930	0.00620	0.00337	7.0615E-03	1.1639E-02					
4	4.50000	8.00000	3.58572	0.01821	0.00265	0.00150	7.1714E-03	1.1639E-02					
5	8.00000	11.00000	3.12129	0.01787	0.00259	0.00164	6.2426E-03	1.1639E-02					
6	11.00000	25.00000	2.37921	0.01722	0.00144	0.00102	4.7584E-03	1.1639E-02					
7	25.00000	50.00000	2.40233	0.01740	0.00245	0.00163	4.8047E-03	1.1639E-02					
8	50.00000	79.80661	2.34329	0.01952	0.00773	0.00530	4.6866E-03	1.1639E-02					
						4	5						
						BACKGROUND	BACKGROUND						
						FOR SAMPLE	FOR SAMPLE						
						IN	OUT						
						2.9384E-03	8.3481E-04						
						2.5788E-03	6.8281E-04						
						1.3530E-03	3.4805E-04						
						2.5578E-04	6.3177E-05						
						1.0764E-04	3.2310E-05						
						6.1094E-05	2.5298E-05						
						9.6161E-05	3.9272E-05						
						4.9091E-04	2.1385E-04						

Table 5. (Continued)

	ELOW	EHIGH	% STD.DEV.	CORRELATION	1	2	3	4	5	6	7	8
1	2.00000	2.40000	0.72		100							
2	2.40000	3.00000	0.64		65	100						
3	3.00000	4.50000	0.55		74	77	100					
4	4.50000	8.00000	0.51		77	81	92	100				
5	8.00000	11.00000	0.57		77	80	91	97	100			
6	11.00000	25.00000	0.72		78	81	92	97	98	100		
7	25.00000	50.00000	0.72		77	80	91	96	97	98	100	
8	50.00000	79.80661	0.83		69	71	81	86	86	87	86	100

REAL ARRAY SIZE USED FOR ALEX IS 512

INTEGER ARRAY SIZE USED FOR ALEX IS 401

Step 6. Rerun

If Step 5 showed the results to be reasonable, proceed to Step 7. If not, make corrections in code (Step 2) or input (Step 3) as needed, and repeat Steps 4 and 5.

Step 7. Additional Runs

To change energy intervals or parameter uncertainties, it is not necessary to repeat the entire calculation. Unless parameter *values* are changed, the "unsummed derivative file" (FEUNSM.ODF in this example) need not be recreated with each ALEX run. To use an existing file in this way, merely answer "Y" (for "yes") to ALEX's prompt "DOES DERIVATIVE FILE ALREADY EXIST?" (see Table 4). ALEX will then ask the question "DOES SUMMED DERIVATIVE FILE ALREADY EXIST?" If energy intervals have been changed, the answer to this question is "N" (for "no"). If only parameter uncertainties or correlations have been changed, the "summed derivative file" (FESUM.ODF in this example) from the previous run can be used as input here (i.e., answer "Y" to this question).

6. SUMMARY AND PROGNOSIS

The computer code ALEX was developed as an aid to the uncertainty analysis of neutron transmission data at ORELA. One major application of ALEX is already being reported, that being the natural nickel experiment of D. C. Larson et al. (DL83). Others are in progress.

Extension of ALEX to other experimental situations should not require extensive re-programming and will be made as needed. It is expected that ALEX will someday be applicable to fission and capture data as well as transmission.

ACKNOWLEDGMENTS

The author is indebted to R. Gwin for suggesting this project and for providing data, advice, and encouragement in the early stages of this work. The financial support in the form of an in-house grant from A. Zucker enabled me to pursue the further development of this code (and incidentally inspired the naming of the code!). D. C. Larson provided both the nickel data which were used for the first major application of the techniques in ALEX and the iron data which were used for the example in this report. Thanks to Dr. Larson also for tolerating the seemingly endless discussions required in order to fully understand all the subtleties involved in an uncertainty analysis. Finally, thanks go to B. J. Waddell and B. R. Roth for accurately deciphering my longhand and patiently typing this manuscript.

REFERENCES

- DL80 D. C. Larson, "ORELA Measurements to Meet Fusion Energy Neutron Cross Section Needs," *Symposium on Neutron Cross Sections From 10 to 50 MeV*, BNL-NCS-51245, pg. 277, July 1980.
- DL83 D. C. Larson, N. M. Larson, J. A. Harvey, N. W. Hill, C. H. Johnson, *Application of New Techniques to ORELA Transmission Measurements and Their Uncertainty Analysis: The Case of Natural Nickel from 2 keV to 20 MeV*, ORNL/TM-8203, Oak Ridge National Laboratory, Oak Ridge, Tenn., 1983.
- JC78 J. G. Craven, *OPRQDF, A DECsystem-10 Data Manipulation Program for ORELA Data Formatted Files*, ORNL/CSD/TM-45, Oak Ridge National Laboratory, Oak Ridge, Tenn., May 1978.
- NL80 N. M. Larson and F. G. Perey, *User's Guide for SAMMY: A Computer Model for Multilevel R-Matrix Fits to Neutron Data Using Bayes' Equations*, ORNL/TM-7485, ENDF-297, Oak Ridge National Laboratory, Oak Ridge, Tenn., November 1980.
- NL82 N. M. Larson, *User's Guide for BAYES: A General-Purpose Computer Code for Fitting a Functional Form to Experimental Data*, Oak Ridge National Laboratory, Oak Ridge, Tenn., ORNL/TM-8185, ENDF-323, August 1982.
- RA82 Radiation Shielding Information Center, Oak Ridge National Laboratory, P. O. Box X, Oak Ridge, TN 37830.

**APPENDIX A
FORTRAN Listing of File ALEX.FOR**

```

C
C *** PROGRAM ALEX *** ANALYSIS OF EXPERIMENTAL DATA
C
C
C *** PROGRAM TO PROPAGATE UNCERTAINTIES THROUGH
C ***           CROSS SECTION GENERATION FROM SAMPLE-IN,
C ***           SAMPLE-OUT DATA
C
C *** SEPTEMBER, 1982
C *** N.M.LARSON
C
C **** **** **** **** **** **** **** **** **** ****
C
C      DESCRIPTION OF THE VARIOUS "UNITS" REQUIRED BY ALEX
C -----
C
C      UNIT      UNIT      DEFAULT
C      NUMBER    NAME      FILENAME      WHAT'S STORED IN FILE
C      -----    ----      -----      -----
C
C      21        IUOUT     ALXOUT.ODF      "FINAL" RESULTS
C      22        IUFIX     ALXFIX.ODF      UNSUMMED CROSS SECTIONS
C                                         AND DERIVATIVES
C      23        IURAW     ALXRAW.DCL      RAW COUNTS AND BACKGROUND
C                                         CORRECTION FACTORS
C      24          ALEXIN.DAT      ALPHANUMERIC AND NUMERIC
C                                         INFORMATION
C      25          ALXAUX.DAT      AUXILIARY CHANNEL INFO
C      26        IUSUM     ALXSUM.ODF      SUMMED CROSS SECTIONS
C                                         AND DERIVATIVES
C      31          ALXOUT.LPT      COVARIANCE MATRICES
C      32          ALXOUT.LPT      LINE-PRINTER OUTPUT
C      33          ALXOUT.LPT      INFO RE WHICH CHANNELS
C                                         CONTRIBUTE TO THE
C                                         ENERGY GROUPS
C
C **** **** **** **** **** **** **** **** ****
C
C      DOUBLE PRECISION TYPE, USE
C      LOGICAL DEBUG
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
COMMON /EXPAND/ A(10000)
COMMON /IEXPND/ NREAL, INTEG, IA(600)
COMMON /REAL/ TDELAY, FPL, C(20)
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
DATA NONO /2HNO/, NPARMX /30/
C
C
NREAL = 10000
INTEG = 600

```

```

C **** BEGIN INITIALIZATION STEP ***
C
C     CALL FILENM(DEBUG)
C         READ NAMES OF FILES, AMONG OTHER THINGS
NNN = 200
C             NNN IS A GUESS FOR THE NUMBER OF ENERGY REGIONS
C             (SPECIFIED BY RH AND RL) REQUIRED FOR THE
C             FINAL RESULTS, AND ULTIMATELY EQUAL TO NENERG
IRH = IDIMEN(NNN)
IRL = IDIMEN(NNN)
KH = JDIMEN(NNN)
KL = JDIMEN(NNN)
CALL INPUT(A(IRH), A(IRL), IA(KH), IA(KL), NNN)
C             TO GET CONSTANTS ETC -- WHATEVER'S NEEDED FOR
C             SPECIAL CASE
I = IDIMEN(-IRH)
I = JDIMEN(-KH)
IF (NENERG.LT.0) GO TO 10
CALL MOVE(A(IRH), A(IRL), NENERG, NENERG)
IRH = IDIMEN(NENERG)
IRL = IDIMEN(NENERG)
CALL MOVE(IA(KH), IA(KL), NENERG, NENERG)
KH = JDIMEN(NENERG)
KL = JDIMEN(NENERG)
C
C *** GET PARAMETERS ORGANIZED
10 NPARAM = NPARMX
NNN = (NPARAM*(NPARAM+1))/2
ICOVPA = IDIMEN(NNN)
IDELPA = IDIMEN(NPARAM)
NAMPAR = JDIMEN(NPARAM*8)
CALL SETPAR(A(ICOVPA), A(IDELPA), IA(NAMPAR))
C             INITIALIZE NUMBER AND VALUE OF PARAMETERS
C             AND COVARIANCE MATRIX ELEMENTS
C
CALL FIXPAR(A(ICOVPA), A(IDELPA), IA(NAMPAR))
C             DROP UNUSED PARAMETERS
NNN = (NPAR*(NPAR+1))/2
CALL MOVE(A(ICOVPA), A(IDELPA), NPAR, NNN)
I = IDIMEN(-ICOVPA)
ICOVPA = IDIMEN(NNN)
IDELPA = IDIMEN(NPAR)
JDUM = JDIMEN(NPAR)
CALL OUTPAR(A(ICOVPA), A(IDELPA), IA(JDUM), DEBUG)
C             WRITE OUT THE PARAMETER VALUES AND THE PARAMETER
C             COVARIANCE MATRIX
C
CALL START(NEW, NEW2)
C             DECIDE WHETHER PARTIAL DERIVATIVES ALREADY EXIST
C
C *** END INITIALIZATION STEP ***
C ****

```

```

IF (NEW.NE.NONO) GO TO 40
C
C
C **** BEGIN STEP A -- EVALUATE CROSS SECTION AND PARTIAL DERIVATIVES ***
C
C *** GET FUNCTIONS ORGANIZED, LABELED ETC
NNN = 100
C           NNN IS A GUESS FOR NFUNCT, THE NUMBER OF FUNCTIONS
C           TO BE INCLUDED
IDEP = JDIMEN(NPARAM)
KWHERE = JDIMEN(NNN)
CALL ZERO(IA(KWHERE), KNORM, NNN)
C
DO 20 JFUNC=1,NNN
  IFUNC = JFUNC
  CALL ZERO2(TYPE, USE, IA(IDEPE))
  CALL PREP(TYPE, USE, IA(IDEPE), IFUNC)
C           DESCRIBE WHAT THE VARIOUS FUNCTIONS DO
C
  CALL SORT(TYPE, USE, IA(IDEPE), IA(KWHERE), IFUNC,
*           KNORM, NNN)
C           REORGANIZE INPUT FROM PREP INTO MORE CONVENIENT FORM
  IF (IFUNC.GE.NFUNC) GO TO 30
  IF (NFUNC.GT.NNN) STOP 3
20 CONTINUE
C
30 I = JDIMEN(-IDEPE)
  CALL RESORT(IA(KWHERE), KNORM)
C           MORE REORGANIZATION
  IF (DEBUG) CALL BUGOUT
C
C *** NOW DONE WITH INITIALIZATION SO CAN GET TO WORK
C
  CALL SIGMAC
C           GENERATE SIGMA AND PARTIAL DERIVATIVES OF SIGMA
C           WRT THE PARAMETERS
C
C *** END STEP A ***
C **** BEGIN STEP B -- REBIN CROSS SECTIONS AND DERIVATIVES ***
C
  IESIGM = IDIMEN(NENERG)
  IEDSIG = IDIMEN(NPAR*NENERG)
  IFRACH = IDIMEN(NENERG)
  IFRACL = IDIMEN(NENERG)
  IDUM = IDIMEN(NENERG)

```

```

      IF (NEW2.EQ.NONO) CALL EXTRA(A(IRH), A(IRL), A(IESIGM),
*       A(IEDSIG), A(IFRACH), A(IFRACL), IA(KH), IA(KL),
*       A(IDUM))
C           FORM INTEGRAL SIGMA(E)DE AND WRITE ON "FILSUM"
C
      IF (NEW2.NE.NONO) CALL GETSUM(A(IRH), A(IRL), A(IESIGM),
*       A(IEDSIG), A(IFRACH), A(IFRACL), IA(KH), IA(KL),
*       A(IDUM))
      I = IDIMEN(-IDUM)
C
      CALL OUTPD(A(IRH), A(IRL), A(IESIGM), A(IEDSIG),
*       IA(NAMPAR))
C           WRITE OUT THE INTEGRATED PARTIAL DERIVATIVES
C
C *** END STEP B ***
C **** **** **** **** **** **** **** **** **** **** ****
C
C **** **** **** **** **** **** **** **** **** **** ****
C *** BEGIN STEP C -- GENERATE COV MATRIX ***
C
      JKM = (NENERG*(NENERG+1))/2
      IVI = IDIMEN(JKM)
      IVO = IDIMEN(JKM)
      IVTOT = IDIMEN(JKM)
C
      CALL FIXCOV(A(ICOVPA), A(IESIGM), A(IEDSIG), A(IFRACH),
*       A(IFRACL), IA(KH), IA(KL), A(IVI), A(IVO), A(IVTOT))
C           GENERATE COVARIANCE MATRIX OF INTEGRATED CROSS
C           SECTIONS, DUE TO ALL UNCERTAINTIES
C
      IDI = IDIMEN(NENERG)
      IDO = IDIMEN(NENERG)
      IDTOT = IDIMEN(NENERG)
      CALL FIXDER(A(IDELPA), A(IEDSIG), A(IVI), A(IVO),
*       A(IVTOT), A(IDI), A(IDO), A(IDTOT))
C           NORMALIZE PARTIAL DERIVATIVES BY TAKING
C           ABSOLUTE VALUES AND MULTIPLYING
C           BY PARAMETER UNCERTAINTIES
C           SET D = SQRT(DIAGONAL OF V)
C
      IC = JDIMEN(NENERG)
      IF (DEBUG) CALL OUTBUG(A(IRH), A(IRL), A(IESIGM),
*       A(IEDSIG), A(IVI), A(IVO), A(IVTOT), A(IDI), A(IDO),
*       A(IDTOT), IA(NAMPAR), IA(IC))
C           WRITE OUT THE RESULTS, INTEGRATED BUT NOT AVERAGED
C
      CALL DIVIDE(A(IRH), A(IRL), A(IESIGM), A(IEDSIG), A(IDI),
*       A(IDO), A(IDTOT))
C           NORMALIZE BY DIVIDING BY ENERGY-DIFFERENCE
C

```

```

IDUMMY = IDIMEN(NENERG)
CALL OUTFIN(A(IRH), A(IRL), A(IESIGM), A(IEDSIG),
*      A(IVTOT), A(IDI), A(IDO), A(IDTOT), A(IDUMMY),
*      IA(NAMPAR), IA(IC))

C
I = IDIMEN(0)
I = JDIMEN(0)

C
STOP

C
C *** END STEP C ***
C ****
C
C
C ****
C *** BEGIN STEPS B AND C, COMBINED
C
C
C *** HERE WE HAVE TOO MANY ENERGY-GROUPS TO DO FULL COVARIANCE
C ***      MATRIX. INSTEAD, JUST DO DIAGONAL PART, AND PUT RESULTS
C ***      ON ODF FILE.

50 CONTINUE
CALL COUNT(NENERG, NUMMAX)

C
MOST = 200
IF (NUMMAX.LT.MOST) MOST = NUMMAX
IECHN = IDIMEN(1+MOST)
ISIGMA = IDIMEN(MOST)
IDSIN = IDIMEN(MOST)
IDSOUT = IDIMEN(MOST)
IDSPAR = IDIMEN(NPAR*MOST)
ICIN = IDIMEN(MOST)
ICOUT = IDIMEN(MOST)
I = IDIMEN(0)
CALL EXTRA2(A(ICOVPA), A(IECHN), A(ISIGMA), A(IDSIN),
*      A(IDSOUT), A(IDSPAR), A(ICIN), A(ICOUT), NUMMAX,
*      MOST)

C *** GENERATE RESULTS
C
STORED IN FILE ALX10K
C
I = IDIMEN(0)
I = JDIMEN(0)
STOP

C *** END STEPS B AND C ***
C ****
END

```

```

FUNCTION IDIMEN(MANY)
C
C      PURPOSE -- KEEP TRACK OF DIMENSIONS OF REAL ARRAYS
C
COMMON /EXPAND/ A(1)
COMMON /IEXPND/ NREAL, INTEG, IA(1)
DATA KOUNT /1/, MAX /0/, MAXT /0/
IF (MANY.LT.0) GO TO 10
IDIMEN = KOUNT
KOUNT = KOUNT + MANY
IF (KOUNT.GT.MAX) MAX = KOUNT
IF (MANY.EQ.0) GO TO 20
IF (KOUNT.GT.NREAL) WRITE (5,99999) NREAL, KOUNT
RETURN
C
10 CONTINUE
KOUNT = -MANY
RETURN
C
20 CONTINUE
WRITE (5,99998) MAX
WRITE (32,99998) MAX
IF (MAX.GT.MAXT) MAXT = MAX
MAX = KOUNT
RETURN
C
99999 FORMAT (////26H AVAILABLE SIZE FOR REAL =, I5, 7H BUT YO,
*           6HU NEED, I6, 8H. ERROR )
99998 FORMAT (////33H REAL ARRAY SIZE USED FOR ALEX IS, I8)
END
C
C
C
FUNCTION JDIMEN(MANY)
C
C      PURPOSE -- KEEP TRACK OF DIMENSIONS OF INTEGER ARRAYS
C
COMMON /EXPAND/ A(1)
COMMON /IEXPND/ NREAL, INTEG, IA(1)
DATA KOUNT /1/, MAX /0/, MAXT /0/
IF (MANY.LT.0) GO TO 10
JDIMEN = KOUNT
KOUNT = KOUNT + MANY
IF (KOUNT.GT.MAX) MAX = KOUNT
IF (MANY.EQ.0) GO TO 20
IF (KOUNT.GT.INTEG) WRITE (5,99999) INTEG, KOUNT
RETURN
C

```

```

10 CONTINUE
KOUNT = -MANY
RETURN
C
20 CONTINUE
WRITE (5,99998) MAX
WRITE (32,99998) MAX
IF (MAX.GT.MAXT) MAXT = MAX
MAX = KOUNT
RETURN
C
99999 FORMAT (29H AVAILABLE SIZE FOR INTEGER =, I5, 9H BUT YOU ,
*      4HNEED, I6, 8H. ERROR )
99998 FORMAT (/36H INTEGER ARRAY SIZE USED FOR ALEX IS, I8)
END
C
C
SUBROUTINE FIXPAR(COVPAR, DELPAR, NAMPAR)
C
C *** PURPOSE -- DELETE UNUSED PARAMETERS
C
DIMENSION COVPAR(1), DELPAR(NPARAM), NAMPAR(8,NPARAM)
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JK, M(21), NCHNMN, NTHICK, NPARAM
C
KPAR = 0
DO 20 IPAR=1,NPARAM
  IF (INPAR(IPAR).LE.0) GO TO 20
  KPAR = KPAR + 1
  INPAR(KPAR) = IPAR
  DELPAR(KPAR) = DELPAR(IPAR)
  DO 10 K=1,8
    NAMPAR(K,KPAR) = NAMPAR(K,IPAR)
10      CONTINUE
20 CONTINUE
NPAR = KPAR
C
KL = 0
DO 40 KPAR=1,NPAR
  IPAR = INPAR(KPAR)
  II = (IPAR*(IPAR-1))/2
  DO 30 LPAR=1,KPAR
    JPAR = INPAR(LPAR)
    KL = KL + 1
    IJ = II + JPAR
    COVPAR(KL) = COVPAR(IJ)
30      CONTINUE
40 CONTINUE
RETURN
END

```

```

C
C
C
      SUBROUTINE OUTPAR(COVPAR, DELPAR, IDUM, DEBUG)
C
C *** PURPOSE -- OUTPUT PARAMETERS
C
      LOGICAL DEBUG
      DIMENSION COVPAR(1), DELPAR(NPAR), IDUM(NPAR)
      DIMENSION CORREL(10)
      COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
      *          JDEP(30,100), INPAR(30)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *          NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DATA CORREL /
      *          50HCORRELATION MATRIX FOR PARAMETERS
C
      IF (.NOT.DEBUG) GO TO 20
      WRITE (32,99999)
      DO 10 IPAR=1,NPAR
         WRITE (32,99998) IPAR, INPAR(IPAR),
      *                  PAR(INPAR(IPAR)),                   DELPAR(IPAR)
      10 CONTINUE
C
      20 CALL OUTV(CORREL, COVPAR, DELPAR, IDUM, NPAR)
      RETURN
C
      99999 FORMAT (/40H           PARAMETER           UNCERTAINTY)
      99998 FORMAT (2I5, 2G14.6)
      END
C
C
C
      SUBROUTINE MOVE(A, B, N, M)
C *** PURPOSE -- REORGANIZE STORAGE IN ARRAY A
      DIMENSION A(N), B(N)
      DO 10 I=1,N
         A(M+I) = B(I)
      10 CONTINUE
      RETURN
      END

```

```

SUBROUTINE FILENM(DEBUG)
C *** PURPOSE -- READ FILENAMES
LOGICAL DEBUG
DOUBLE PRECISION FILFIX, FILSUM, FILOUT
DOUBLE PRECISION FILE1, FILE2, FILE3, FILE4
DOUBLE PRECISION BLANK
DIMENSION FILE1(3), FILE2(3)
COMMON /ODF/ FILFIX, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
*      IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
*      NDSTRT
DATA FILE3 /10HALXFIX.ODF/, FILE4 /10HALXSUM.ODF/
DATA BLANK /10H          /, YES /1HY/

C
WRITE (5,99999)
READ (5,99998) FILE1
OPEN (UNIT=24, FILE='ALEXIN.DAT', ACCESS='SEQIN',
*      DIALOG=FILE1)

C
WRITE (5,99997)
READ (5,99998) FILFIX
IF (FILFIX.EQ.BLANK) FILFIX = FILE3

C
WRITE (5,99996)
READ (5,99998) FILSUM
IF (FILSUM.EQ.BLANK) FILSUM = FILE4

C
WRITE (5,99995)
READ (5,99998) FILE2
OPEN (UNIT=32, FILE='ALXOUT.LPT', ACCESS='SEQOUT',
*      DIALOG=FILE2)
IF (FILE1(1).NE.BLANK) WRITE (32,99994) FILE1
IF (FILE1(1).EQ.BLANK) WRITE (32,99993)

C
WRITE (5,99992)
READ (5,99991) ANS
DEBUG = .FALSE.
IF (ANS.EQ.YES) DEBUG = .TRUE.
RETURN

99999 FORMAT (41H WHAT IS FILENAME FOR INPUT INFORMATION? $)
99998 FORMAT (3A10)
99997 FORMAT (46H WHAT IS FILENAME FOR UNSUMMED CROSS SECTION A,
*      16HND DERIVATIVES? $)
99996 FORMAT (46H WHAT IS FILENAME FOR SUMMED CROSS SECTION AND,
*      14H DERIVATIVES? $)
99995 FORMAT (43H WHAT IS FILENAME FOR LINE PRINTER OUTPUT? $)
99994 FORMAT (/38H *****INPUT DATA IS FOUND ON FILE ", 3A10,
*      10H" *****///)
99993 FORMAT (/41H INPUT DATA IS FOUND ON FILE "ALEXIN.DAT")
99992 FORMAT (29H DO YOU WANT DEBUG PRINTOUT? $)
99991 FORMAT (A1)
END

```

```

C
C
      SUBROUTINE START(NEW1, NEW2)
C
C *** PURPOSE -- DETERMINE IF CROSS SECTION AND DERIVATIVES
C ***           WRT PARAMETERS HAVE ALREADY BEEN SET
C
      DOUBLE PRECISION FILFIX, FILSUM, FILOUT, BLANK, ALXOUT
      COMMON /ODF/ FILFIX, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
*           IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
*           NDSTRT
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK
      DATA BLANK /1H          /, NNO /1HN/, NONO /2HNO/
      DATA ALXOUT /10HALXOUT.ODF/
C
      WRITE (5,99999)
      READ (5,99998) NEW1
      IF (NEW1.EQ.NNO) NEW1 = NONO
      IF (NEW1.NE.NONO) GO TO 10
      WRITE (32,99997) FILFIX
      WRITE (5,99997) FILFIX
      NEW2 = NONO
      WRITE (32,99994) FILSUM
      WRITE (5,99994) FILSUM
      GO TO 30
10   WRITE (32,99996) FILFIX
      WRITE (5,99996) FILFIX
C
      IUFIX = 22
      NEW = 0
      CALL ODFIO(IUFIX, FILFIX, IFB, NEW, INSFIX, K, MODE,
*           NDSTRT, IENER, IRUN)
      IF (K.NE.NCHN-NCHNMN) STOP 201
C
      IF (NENERG.LT.0) GO TO 30
      WRITE (5,99995)
      READ (5,99998) NEW2
      IF (NEW2.EQ.NNO) NEW2 = NONO
      IF (NEW2.NE.NONO) GO TO 20
      WRITE (32,99994) FILSUM
      WRITE (5,99994) FILSUM
      GO TO 30
20   WRITE (32,99993) FILSUM
      WRITE (5,99993) FILSUM
C
      IUSUM = 26
      NEW = 0
      CALL ODFIO(IUSUM, FILSUM, IFB, NEW, INSSUM, KENERG, MODE,
*           NDSTRT, IENER, IRUN)
      IF (KENERG.NE.NENERG) STOP 202

```

```
C
30 CONTINUE
WRITE (5,99992)
READ (5,99991) FILOUT
IF (FILOUT.EQ.BLANK) FILOUT = ALXOUT
RETURN

C
99999 FORMAT (37H DOES DERIVATIVE FILE ALREADY EXIST? $)
99998 FORMAT (A1)
99997 FORMAT (///36H *****,*
*      49H UNSUMMED DERIVATIVE FILE CREATED BY THIS RUN IS ,
*      7HCALLED , A10)
99996 FORMAT (///36H *****,*
*      49H PRE-EXISTING DERIVATIVE FILE USED BY THIS RUN IS,
*      8H CALLED , A10)
99995 FORMAT (44H DOES SUMMED DERIVATIVE FILE ALREADY EXIST? $)
99994 FORMAT (46H SUMMED DERIVATIVE FILE CREATED BY THIS RUN IS,
*      8H CALLED , A10/36H *****,*
*      19HTHIS RUN IS CALLED , A10/
*      36H *****,*//)
99992 FORMAT (35H WHAT IS NAME FOR ODF OUTPUT FILE? $)
99991 FORMAT (A10)
END
```

**APPENDIX B
FORTRAN Listing of File ALXFIX.FOR**

```

C
C
C
      SUBROUTINE ZERO(KWHERE, KNORM, NNN)
C *** PURPOSE -- ZERO ARRAYS KWHERE AND JDEP
      COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
      *      JDEP(30,100), INPAR(30)
      DIMENSION KWHERE(NNN)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      KNORM = 0
      DO 20 J=1,NNN
         DO 10 I=1,NPAR
            JDEP(I,J) = 0
10      CONTINUE
20      CONTINUE
      DO 30 J=1,NNN
         KWHERE(J) = 0
30      CONTINUE
      RETURN
      END

C
C
C
      SUBROUTINE ZERO2(TYPE, USE, IDEP)
C *** PURPOSE -- ZERO ARRAY "IDEP" AND VARIABLES "TYPE" AND "USE"
      DOUBLE PRECISION TYPE, USE, BLANK
      DIMENSION IDEP(NPAR)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DATA BLANK /10H          /
      TYPE = BLANK
      USE = BLANK
      DO 10 I=1,NPAR
         IDEP(I) = 0
10      CONTINUE
      RETURN
      END
C
C

```

```

SUBROUTINE SORT(TYPE, USE, IDEP, KWHERE, IFUNC, KNORM,
*      NNN)
C *** PURPOSE -- REORGANIZE INFORMATION ABOUT THE FUNCTION
      DOUBLE PRECISION TYPE, USE, ANORM, SAMIN, SAMOUT, DTIME,
*      BACKG, BLANK
      DIMENSION IDEP(NPAR), KWHERE(NNN)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
      DATA ANORM /10HNORMALIZAT/, SAMIN /10HSAMPLE-IN /, SAMOUT
*      /10HSAMPLE-OUT/, DTIME /10HDEADTIME C/, BACKG /
*      10HBACKGROUND/, BLANK /10H
C
      IF (IDEPA(1).EQ.0) GO TO 30
      DO 20 I=1,NPARAM
         IF (IDEPA(I).EQ.0) GO TO 80
         DO 10 II=1,NPAR
            IF (IDEPA(I).EQ.INPAR(II)) GO TO 30
10      CONTINUE
20      CONTINUE
      GO TO 80
C      IE, THIS FUNCTION IS NEVER USED SO JUMP OUT OF
C      THIS SUBROUTINE
30      IF (TYPE.EQ.ANORM) KNORM = IFUNC
      IF (TYPE.EQ.ANORM) GO TO 40
      IF (TYPE.EQ.SAMIN .AND. USE.EQ.DTIME) KWHERE(IFUNC) = 2
      IF (TYPE.EQ.SAMIN .AND. USE.EQ.BACKG) KWHERE(IFUNC) = 3
      IF (TYPE.EQ.SAMOUT .AND. USE.EQ.DTIME) KWHERE(IFUNC) = 4
      IF (TYPE.EQ.SAMOUT .AND. USE.EQ.BACKG) KWHERE(IFUNC) = 5
      IF (TYPE.NE.ANORM .AND. TYPE.NE.SAMIN .AND.
*      TYPE.NE.SAMOUT) WRITE (5,99999) TYPE
      IF (TYPE.NE.ANORM .AND. TYPE.NE.SAMIN .AND.
*      TYPE.NE.SAMOUT) STOP 4
      IF (USE.NE.BLANK .AND. USE.NE.DTIME .AND. USE.NE.BACKG)
*      STOP 5
40      CONTINUE
      JJ = 0
      DO 70 I=1,NPARAM
         IPAR = IDEP(I)
         IF (IPAR.EQ.0) GO TO 80
         DO 50 II=1,NPAR
            IF (INPAR(II).EQ.IPAR) GO TO 60
50      CONTINUE
      GO TO 70
60      JJ = JJ + 1
      JDEP(JJ,IFUNC) = IPAR
70      CONTINUE
80      RETURN
99999 FORMAT (5HTYPE=, A10)
      END

```

```

SUBROUTINE RESORT(KWHERE, KNORM)
C *** PURPOSE -- RE-SORT IWHERE, WHICH TELLS WHICH FUNCTION IS WHICH
DIMENSION KWHERE(1)
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DO 10 I=1,NPAR
    IF (INPAR(I).EQ.NTHICK) GO TO 20
10 CONTINUE
GO TO 30
20 NTHICK = I
30 IWHERE(1) = KNORM
K = 2
IF (KNORM.EQ.0) K = 1
IWHERE(K) = 0
DO 50 J=2,5
    DO 40 IFUNC=1,NFUNC
        IF (KWHERE(IFUNC).NE.J) GO TO 40
        K = K + 1
        IWHERE(K) = IFUNC
40 CONTINUE
K = K + 1
IWHERE(K) = 0
50 CONTINUE
CX     DO 30 IFUNC=1,NFUNC
CX         IF ((KWHERE(IFUNC).EQ.0 .AND. KNORM.NE.IFUNC) .OR.
CX         *      (KWHERE(IFUNC).EQ.1) .OR. (KWHERE(IFUNC)
CX         *      .GT.NFUNC)) STOP 6
CX     30 CONTINUE
RETURN
END
C
C
C
SUBROUTINE BUGOUT
C *** PURPOSE -- DEBUG PRINTOUT
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
WRITE (5,99999)
DO 10 I=1,NFUNC
    WRITE (5,99998) I, (JDEP(J,I),J=1,NPARAM)
10 CONTINUE
WRITE (5,99997) (IWHERE(I),I=1,NFUNC+5)
RETURN
99999 FORMAT (/1H FUNC JDEP)
99998 FORMAT (I4, 1X, 30I2)
99997 FORMAT (/9H IWHERE =, 15I2)
END

```

```

SUBROUTINE SIGMAC
C
C *** PURPOSE -- EVALUATE CROSS SECTION AND ITS DERIVATIVES
C ***           WRT PARAMETERS
C
      DOUBLE PRECISION FILE, FILSUM, FILOUT
      COMMON /STORE/ ENERGY(512), SIGMA(512), DSIN(512),
      *          DSOUT(512), DSPAR(512,30)
      COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
      *          JDEP(30,100), INPAR(30)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *          NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      COMMON /REAL/ TDELAY, FPL, C(20)
      COMMON /ODF/ FILE, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
      *          IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
      *          NDSTR
      COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
      *          CHN, ICHN
      DATA MANY /512/, NPARMX /30/
C
      IF (NPAR.GT.NPARMX) GO TO 40
C
      IUFIX = 22
      IENER = -1
      IRUN = 0
      NEW = 1
      INSFIX = 4 + NPAR
      CALL ODFIO(IUFIX, FILE, IFB, NEW, INSFIX, NCHN-NCHNMN,
      *          MODE, NDSTR, IENER, IRUN)
C
      THICK = PAR(INPAR(NTHICK))
      MX = (NCHN-NCHNMN-1)/MANY + 1
      MANYX = MANY
      ICHN = NCHNMN
      WRITE (5,99999) MX, MANY
      DO 30 KOUNT=1,MX
          WRITE (5,99998) KOUNT, MANYX
          IF (KOUNT.EQ.MX) MANYX = NCHN - NCHNMN -
          *          (KOUNT-1)*MANY
C
      DO 20 KOUNTR=1,MANYX
          ICHN = ICHN + 1
          CALL GETRAW
C
          GETRAW SETS VALUES OF CIN,COUT,DTCIN,DTCOUT,ECHN
          IF (ECHN.EQ.0.0) GO TO 20
C
          CALL FUNCTN
          FUNCTN EVALUATES ALL THE FUNCTIONS (FFUNC) FOR
              THIS ICHN
C
          ENERGY(KOUNTR) = ECHN

```

```

SIN = CIN*FFUNC(2) - FFUNC(3)
SOUT = COUT*FFUNC(4) - FFUNC(5)
TRANS = FFUNC(1)*SIN/SOUT
IF (TRANS.GT.0.0) SIGMA(KOUNTR) =
      -ALOG(TRANS)/THICK
IF (TRANS.LE.0.0) SIGMA(KOUNTR) = 999.
DSIN(KOUNTR) = -FFUNC(2)/(SIN*THICK)
DSOUT(KOUNTR) = FFUNC(4)/(SOUT*THICK)

C
DO 10 KKK=1,NPAR
      KPAR = KKK
      DSPAR(KOUNTR,KPAR) = 0.E0
      CALL DDERIV(KPAR)
      EVALUATE CDELP=DERIV OF FFUNC WRT PARAMETER # KPAR

C
C ** FFUNC1
      IF (CDELP(1).NE.0.0D0) DSPAR(KOUNTR,KPAR)
      *           = -CDELP(1)/(FFUNC(1)*THICK)

C
C ** FFUNC2
      IF (CDELP(2).NE.0.0D0) DSPAR(KOUNTR,KPAR)
      *           = -CIN*CDELP(2)/(SIN*THICK)

C
C ** FFUNC3
      IF (CDELP(3).NE.0.0D0) DSPAR(KOUNTR,KPAR)
      *           = CDELP(3)/(SIN*THICK)

C
C ** FFUNC4
      IF (CDELP(4).NE.0.0D0) DSPAR(KOUNTR,KPAR)
      *           = CDELP(4)*COUT/(SOUT*THICK)

C
C ** FFUNC5
      IF (CDELP(5).NE.0.0D0) DSPAR(KOUNTR,KPAR)
      *           = -CDELP(5)/(SOUT*THICK)

10      CONTINUE
C
      DSPAR(KOUNTR,NTHICK) = -SIGMA(KOUNTR)/THICK +
      *           DSPAR(KOUNTR,NTHICK)

20      CONTINUE
C
      CALL PUTFIX(MANYX)

30 CONTINUE
      RETURN
C
40 WRITE (5,99997) NPAR, NPARMX
      STOP
99999 FORMAT (/9H MX,MANY=, 2I5)
99998 FORMAT (13H KOUNT,MANYX=, 2I5)
99997 FORMAT (13H NEED NPARMX=, I4, 24H IN ALXFIX; CURRENT VALU,
      *           1HE, 3H IS, I4/34H CHANGE IN COMMON/STORE/...DSPAR(X,
      *           5HXXX,N, 28HPARMX) AND IN DATA STATEMENT)
      END

```

```

SUBROUTINE FUNCTN
C
C *** PURPOSE -- EVALUATE ALL 5 KINDS OF FUNCTIONS
C
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
N = 0
DO 20 KKK=1,5
    FFUNC(KKK) = 0.0E0
10     N = N + 1
        IF (IWHERE(N).EQ.0) GO TO 20
        FFUNC(KKK) = FFUNC(KKK) + FUNC(IWHERE(N))
        GO TO 10
20 CONTINUE
IF (FFUNC(1).EQ.0.0E0) FFUNC(1) = 1.0E0
IF (FFUNC(2).EQ.0.0E0) FFUNC(2) = 1.0E0
IF (FFUNC(4).EQ.0.0E0) FFUNC(4) = 1.0E0
RETURN
END
C
C
SUBROUTINE DDERIV(KPAR)
C
C *** PURPOSE -- EVALUATE DSIGMA/DPAR = DSPAR
C
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JK, M(21), NCHNMN, NTHICK, NPARAM
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
N = 0
DO 30 KKK=1,5
    CDELP(KKK) = 0.
10     N = N + 1
        IF (IWHERE(N).EQ.0) GO TO 30
        DO 20 IPAR=1,NPARAM
            IF (JDEP(IPAR,IWHERE(N)).EQ.0) GO TO 10
C               IE, WE'VE CHECKED ALL PARAMETERS RELEVANT TO
C               FUNCTION IWHERE(N), AND DIDN'T FIND KPAR
            IF (JDEP(IPAR,IWHERE(N)).NE.INPAR(KPAR)) GO TO
*                20
C               IE, PARAMETER IPAR IS NOT THE ONE WE WANT
C
            HERE, JDEP=KPAR SO WE NEED THIS DERIVATIVE
            CDELP(KKK) = CDELP(KKK) + DERIV(IWHERE(N),
*                INPAR(KPAR))
            GO TO 10
20     CONTINUE
STOP 66666
30 CONTINUE
RETURN
END

```

```

C
C
      SUBROUTINE GETRAW
C
C *** PURPOSE -- READ ENERGIES "XECHN", DEAD TIME CORRECTION
C ***           FACTORS "XDTCIN" AND "XDTCOU", AND RAW COUNTS
C ***
C ***           "XCIN" AND "XCOUT" FROM "RAW" DATA FILE. ALSO,
C ***           FIGURE WHAT CHANNEL IS WANTED NOW AND WHAT ITS
C ***           ENERGY "ECHN", CHANNEL NUMBER "ICHN", TIME "TCHN",
C ***           AND TIME-WIDTH "CHN" ARE, ALONG WITH "DTCIN",
C ***           "DTCOUT", "CIN", AND "COUT".
C
      COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*      DSOUT, DSP(30)
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
*      IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
*      INSOUT, MODE, NDSTRT
      COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
*      CHN, ICHN
      COMMON /STORE/ XECHN(512), XDTCIN(512), XDTCOU(512),
*      XCIN(512), XCOUT(512), XDUMMY(512,29)
C ****   XCIN(512), XCOUT(512), XDUMMY(512,NPAR-1)
      REAL XECHN, XDTCIN, XDTCOU, XCIN, XCOUT
      COMMON /REAL/ TDELAY, FPL, C(20)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DATA MANY /512/, MANYX /512/, MINCHN /0/, MAXCHN /0/,
*      ICHNP /0/
      DATA C1 /1.59647E-03/, C2 /2.831906E-06/, C3 /0.07229855/
C
      IF (MAXCHN.GT.0) GO TO 30
      TOLD = -TDELAY
      MAXCHN = NCHNMN
      L = 1
      MQ = 0
      DO 20 ICHNP=1,NCHNMN
         IF (ICHNP.NE.M(L+1)) GO TO 10
         WRITE (5,99999) L, M(L+1)
         TOLD = TOLD + FLOAT(M(L+1)-M(L))*C(L)
         WRITE (5,99998) TOLD
         L = L + 1
         MQ = 0
10      MQ = MQ + 1
20  CONTINUE
      T = TOLD + FLOAT(MQ)*C(L)
      FPL = FPL*C3
      E = (FPL/T)**2
      EMAX = E*(1.+E*(C1+E*C2))
C

```

```

30 CONTINUE
  IF (ICHN.LE.MAXCHN) GO TO 40
  ICHNP = 0
  MINCHN = MAXCHN + 1
  MAXCHN = MAXCHN + MANY
  IF (MAXCHN.GT.NCHN) MAXCHN = NCHN
  IF (MAXCHN.EQ.NCHN) MANYX = NCHN - MINCHN + 1
C
C *** HERE WE NEED TO READ FROM ODF FILE TO GET A NEW CHUNCK OF DATA
C
  CALL INODF(IURAW, IFB, INSRAW, 1, MODE, NDstrt, MINCHN,
*      MANYX, XECHN, 1)
  CALL INODF(IURAW, IFB, INSRAW, 2, MODE, NDstrt, MINCHN,
*      MANYX, XDTCIN, 1)
  CALL INODF(IURAW, IFB, INSRAW, 3, MODE, NDstrt, MINCHN,
*      MANYX, XDTcou, 1)
  CALL INODF(IURAW, IFB, INSRAW, 4, MODE, NDstrt, MINCHN,
*      MANYX, XCIN, 1)
  CALL INODF(IURAW, IFB, INSRAW, 5, MODE, NDstrt, MINCHN,
*      MANYX, XCOUT, 1)
C
C
40 ICHNP = ICHNP + 1
  IF (ICHN.NE.M(L+1)) GO TO 50
  WRITE (5,99999) L, M(L+1)
  TOLD = TOLD + FLOAT(M(L+1)-M(L))*C(L)
  WRITE (5,99998) TOLD, T, EMIN
  MQ = 0
  L = L + 1
50 MQ = MQ + 1
  T = TOLD + FLOAT(MQ)*C(L)
  E = (FPL/T)**2
  EMIN = E*(1.+E*(C1+E*C2))
  TCHN = T*0.001
C
          CONVERT TO MICROSECONDS
  ECHN = EMIN
  CHN = C(L)
  IF (XECHN(ICHP).EQ.0.0) RETURN
  CIN = XCIN(ICHP)
  COUT = XCOUT(ICHP)
  DTcin = XDTCIN(ICHP)
  DTcOut = XDTcou(ICHP)
C
          RETURN
C
99999 FORMAT (10H L,M(L+1)=, I3, I7)
99998 FORMAT (12H TEXACT,T,E=, 3F20.10)
  .
  END
C

```

```

C
C
      SUBROUTINE PUTFIX(MANYX)

C *** PURPOSE -- PUT "ADJUSTED" DATA INTO "FIX"ED DATA FILE
C

      COMMON /STORE/ ENERGY(512), SIGMA(512), DSIN(512),
*      DSOUT(512), DSPAR(512,30)
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
*      IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
*      INSOUT, MODE, NDSTRT
      COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
*      CHN, ICHN
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DATA MINCHN /1/

C
C
      CALL OUTODF(IUFIX, IFB, INSFIX, 1, MODE, NDSTRT, MINCHN,
*      MANYX, ENERGY, 1)
      CALL OUTODF(IUFIX, IFB, INSFIX, 2, MODE, NDSTRT, MINCHN,
*      MANYX, SIGMA, 1)
      CALL OUTODF(IUFIX, IFB, INSFIX, 3, MODE, NDSTRT, MINCHN,
*      MANYX, DSIN, 1)
      CALL OUTODF(IUFIX, IFB, INSFIX, 4, MODE, NDSTRT, MINCHN,
*      MANYX, DSOUT, 1)
      DO 10 KPAR=1,NPAR
          CALL OUTODF(IUFIX, IFB, INSFIX, 4+KPAR, MODE,
*                  NDSTRT, MINCHN, MANYX, DSPAR(1,KPAR), 1)
10 CONTINUE
C
      MINCHN = MINCHN + MANYX
      RETURN
      END

```

**APPENDIX C
FORTRAN Listing of File ALXSUM.FOR**

```

C
C
C
SUBROUTINE EXTRA(RH, RL, ESIGMA, EDSIGM, FRACH, FRACL,
*      KH, KL, DUM)
C
C *** PURPOSE -- FIND INTEGRATED CROSS SECTION
C
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
COMMON /REAL/ TDELAY, FPL, C(20)
DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), FRACH(NENERG), FRACL(NENERG),
*      KH(NENERG), KL(NENERG), DUM(NENERG)
COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*      DSOUT, DSP(30)
C
C
IF (KH(1).GT.0) GO TO 20
ELAST = 1.E30
ENOW = 1.E30
J = NENERG + 1
DO 10 JJ=1,NENERG
   J = J - 1
   CALL ESUM(RH(J), RL(J), ESIGMA(J), EDSIGM(1,J),
*            FRACH(J), FRACL(J), KH(J), KL(J))
10 CONTINUE
GO TO 40
C
20 CONTINUE
DO 30 J=1,NENERG
   CALL CHNSUM(RH(J), RL(J), ESIGMA(J), EDSIGM(1,J),
*            FRACH(J), FRACL(J), KH(J), KL(J))
30 CONTINUE
C
40 CONTINUE
CALL PUTSUM(RH, RL, ESIGMA, EDSIGM, FRACH, FRACL, KH, KL,
*            DUM)
C
RETURN
END
C

```

```

        SUBROUTINE ESUM(EH, EL, ESIGMA, EDSIGM, FRACH, FRACTL, KH,
*      KL)
C
C *** PURPOSE -- GENERATE INTEGRAL (FROM EL TO EH) OF SIGMA(E)DE
C ***           AND OF (DSIGMA/DPAR)DE
C
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DIMENSION EDSIGM(NPAR)
COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*      DSOUT, DSP(30)
DATA K /0/
KH = 0
KL = 0
FRACH = 0.0
FRACTL = 0.0
ESIGMA = 0.0D0
DO 10 I=1,NPAR
    EDSIGM(I) = 0.
10 CONTINUE
C
IF (EMAX.EQ.0.) EMAX = 19.62242
IF (EMIN.EQ.0.) EMIN = .0019909497
IF (EL.GT.EMAX) GO TO 90
IF (EH.LT.EMIN) GO TO 90
IF (EH.GE.ENOW) GO TO 20
C
C ** FIND CHANNEL NUMBER K FOR FIRST CHANNEL (HIGHEST ENERGY)
KFIN = 1
CALL GETFIX(EH, EL, K, KFIN)
KFIN = 0
IF (K.EQ.0) GO TO 90
IF (K.EQ.1 .AND. EH.GT.ELAST) EH = ELAST
20 KH = K
DEL = EH - ENOW
FRACH = DEL
ESIGMA = SIG*DEL
DO 30 IPAR=1,NPAR
    EDSIGM(IPAR) = DSP(IPAR)*DEL
30 CONTINUE
C
C ** ADD UP DATA TWEEN FIRST AND LAST CHANNELS
40 CALL GETFIX(EH, EL, K, KFIN)
IF (ENOW.LT.0.) GO TO 60
IF (ENOW.LE.EL) GO TO 70
DEL = ELAST - ENOW
ESIGMA = SIG*DEL + ESIGMA
DO 50 IPAR=1,NPAR
    EDSIGM(IPAR) = DSP(IPAR)*DEL + EDSIGM(IPAR)
50 CONTINUE
GO TO 40

```

```

C ** ADD FINAL CHANNEL CONTRIBUTION
60 K = NCHN - NCHNMN
   EL = -ENOW
70 CONTINUE
   DEL = ELAST - EL
   ESIGMA = SIG*DEL + ESIGMA
   DO 80 IPAR=1,NPAR
      EDSIGM(IPAR) = DSP(IPAR)*DEL + EDSIGM(IPAR)
80 CONTINUE
   FRACL = DEL
   KL = K
90 RETURN

C
   RETURN
END

C
C
C
   SUBROUTINE GETFIX(EH, EL, K, KFIND)

C *** PURPOSE -- READ "FIXED" DATA FROM TEMPORARY FILE
C

COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
*      IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSSUM,
*      INSOUT, MODE, NDSTRT
COMMON /CHANNL/ CIN, COUT, DTCIN, DTOUT, ECHN, TCHN,
*      CHN, ICHN
COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DDSIN,
*      DDSOUT, DSP(30)
COMMON /STORE/ ENERGY(512), SIGMA(512), DSIN(512),
*      DSOUT(512), DSPAR(512,30)
COMMON /REAL/ TDELAY, FPL, C(20)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DATA MANY /512/, KK /0/

C
IF (KFIND.EQ.1) KBIG = 0
KK = KK + 1
K = K + 1
ELAST = ENOW
IF (K.GE.NCHN-NCHNMN) ENOW = -EMIN
IF (K.GT.NCHN-NCHNMN) RETURN
IF (K.GT.1 .AND. K.LE.MAXCHN .AND. KFIND.EQ.0) GO TO 80
IF (K.GT.1 .AND. K.LE.MAXCHN .AND. KFIND.NE.0) GO TO 40
IF (K.GT.1 .AND. K.GT.MAXCHN) GO TO 10
ELAST = EMAX
MINCHN = 1
MAXCHN = MANY
MANYX = MANY
GO TO 20

```

```

10 CONTINUE
  KK = 1
  MINCHN = MAXCHN + 1
  MAXCHN = MAXCHN + MANY
C
20 CONTINUE
  IF (MAXCHN.GT.NCHN-NCHNMN) MAXCHN = NCHN - NCHNMN
  IF (MAXCHN.EQ.NCHN-NCHNMN) MANYX = MAXCHN - MINCHN + 1
C
C *** HERE WE NEED TO READ FROM ODF FILE TO GET A NEW CHUNK OF DATA
C
  CALL INODF(IUFIX, IFB, INSFIX, 1, MODE, NDSTRT, MINCHN,
*      MANYX, ENERGY, 1)
  IF (EH.LE.ENERGY(MANYX)) GO TO 60
  CALL INODF(IUFIX, IFB, INSFIX, 2, MODE, NDSTRT, MINCHN,
*      MANYX, SIGMA, 1)
  DO 30 KPAR=1,NPAR
    CALL INODF(IUFIX, IFB, INSFIX, 4+KPAR, MODE, NDSTRT,
*      MINCHN, MANYX, DSPAR(1,KPAR), 1)
30 CONTINUE
C
  IF (KFIND.EQ.0) GO TO 80
40 IF (EH.LE.ENERGY(MANYX)) GO TO 60
  KKMIN = KK
  DO 50 KK=KKMIN,MANYX
    IF (EH.GT.ENERGY(KK)) GO TO 70
50 CONTINUE
60 ELAST = ENERGY(MANYX)
  IF (MAXCHN.EQ.NCHN-NCHNMN) K = 0
  IF (MAXCHN.EQ.NCHN-NCHNMN) RETURN
  GO TO 10
70 K = KK + MINCHN - 1
  IF (KK.GT.1) ELAST = ENERGY(KK-1)
C
80 ENOW = ENERGY(KK)
  SIG = SIGMA(KK)
  IF (SIG.GE.998. .AND. KBIG.EQ.0) WRITE (5,99999) ENOW
  IF (SIG.GE.998. .AND. KBIG.EQ.0) WRITE (32,99999) ENOW
  IF (SIG.GE.998. .AND. KBIG.EQ.0) KBIG = 1
  DO 90 KPAR=1,NPAR
    DSP(KPAR) = DSPAR(KK,KPAR)
90 CONTINUE
C
  RETURN
99999 FORMAT (42H **** ERROR. NEGATIVE TRANSMISSION,
*      19H FOUND AT ENERGY = , 1PE14.6, 16H AND POSSIBLY AT,
*      36H SMALLER ENERGIES IN THE SAME RANGE./9H YOU MUST,
*      43H CHANGE ENERGY INTERVALS TO AVOID BAD DATA]/
*      16H ****)
END

```

```

SUBROUTINE CHNSUM(EH, EL, ESIGMA, EDSIGM, FRACH, FRACL,
*      KH, KL)
C
C *** PURPOSE -- GENERATE INTEGRAL (FROM EL TO EH) OF SIGMA(E)DE
C ***          AND OF (DSIGMA/DPAR)DE, KNOWING KH AND KL
C
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DIMENSION EDSIGM(NPAR)
COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*      DSOUT, DSP(30)
DATA K /0/
C
FRACH = 0.0
FRACL = 0.0
ESIGMA = 0.0D0
C
DO 10 I=1,NPAR
    EDSIGM(I) = 0.
10 CONTINUE
C
KFINDD = 1
CALL GETFXK(KH, K, KFINDD)
KFINDD = 0
IF (K.EQ.0) GO TO 50
EH = ELAST
DEL = EH - ENOW
FRACH = DEL
ESIGMA = SIG*DEL
DO 20 IPAR=1,NPAR
    EDSIGM(IPAR) = DSP(IPAR)*DEL
20 CONTINUE
IF (KL.EQ.KH) EL = ENOW
IF (KL.EQ.KH) RETURN
C
C ** ADD UP DATA TWEEN FIRST AND LAST CHANNELS
DO 40 KKK=KH+1,KL
    CALL GETFXK(KH, K, KFINDD)
    DEL = ELAST - ENOW
    ESIGMA = SIG*DEL + ESIGMA
    DO 30 IPAR=1,NPAR
        EDSIGM(IPAR) = DSP(IPAR)*DEL + EDSIGM(IPAR)
30     CONTINUE
40 CONTINUE
C
C ** FIGURE FINAL CHANNEL CONTRIBUTION
FRACL = DEL
EL = ENOW
50 RETURN
C
END

```

```

      SUBROUTINE GETFXK(KH, K, KFIND)
C *** PURPOSE -- READ "FIXED" DATA FROM TEMPORARY FILE
C
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
      *          IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
      *          INSOUT, MODE, NDSTRT
      COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
      *          CHN, ICHN
      COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DDSIN,
      *          DDSOUT, DSP(30)
      COMMON /STORE/ ENERGY(512), SIGMA(512), DSIN(512),
      *          DSOUT(512), DSPAR(512,30)
      COMMON /REAL/ TDELAY, FPL, C(20)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *          NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DATA MANY /512/, KK /0/
C
      IF (KFIND.EQ.1) KBIG = 0
      KK = KK + 1
      K = K + 1
      ELAST = ENOW
      IF (K.GE.NCHN-NCHNMN) ENOW = -EMIN
      IF (K.GT.NCHN-NCHNMN) RETURN
      IF (K.GT.1 .AND. K.LE.MAXCHN .AND. KFIND.EQ.0) GO TO 50
      IF (K.GT.1 .AND. K.LE.MAXCHN .AND. KFIND.NE.0) GO TO 40
      IF (K.GT.1 .AND. K.GT.MAXCHN) GO TO 10
      ELAST = EMAX
      MINCHN = KH - 1
      MAXCHN = MANY + MINCHN - 1
      MANYX = MANY
      GO TO 20
C
      10 CONTINUE
      KK = 1
      MINCHN = MAXCHN + 1
      MAXCHN = MAXCHN + MANY
C
      20 CONTINUE
      IF (MAXCHN.GT.NCHN-NCHNMN) MAXCHN = NCHN - NCHNMN
      IF (MAXCHN.EQ.NCHN-NCHNMN) MANYX = MAXCHN - MINCHN + 1
C
      C *** HERE WE NEED TO READ FROM ODF FILE TO GET A NEW CHUNK OF DATA
C
      CALL INODF(IUFIX, IFB, INSFIX, 1, MODE, NDSTRT, MINCHN,
      *          MANYX, ENERGY, 1)
      CALL INODF(IUFIX, IFB, INSFIX, 2, MODE, NDSTRT, MINCHN,
      *          MANYX, SIGMA, 1)
      DO 30 KPAR=1,NPAR
      CALL INODF(IUFIX, IFB, INSFIX, 4+KPAR, MODE, NDSTRT,
      *          MINCHN, MANYX, DSPAR(1,KPAR), 1)
C
      30 CONTINUE

```

```

C
      IF (KFIN.D_EQ.0) GO TO 50
40  KK = KH - MINCHN + 1
      K = KH
      IF (KK.GT.1) ELAST = ENERGY(KK-1)
C
50  ENOW = ENERGY(KK)
      SIG = SIGMA(KK)
      IF (SIG.GE.999. .AND. KBIG.EQ.0) WRITE (5,99999) ENOW
      IF (SIG.GE.999. .AND. KBIG.EQ.0) WRITE (32,99999) ENOW
      IF (SIG.GE.998. .AND. KBIG.EQ.0) KBIG = 1
      DO 60 KPAR=1,NPAR
          DSP(KPAR) = DSPAR(KK,KPAR)
60  CONTINUE
C
      RETURN
99999 FORMAT (42H **** ERROR. NEGATIVE TRANSMISSION,
*      19H FOUND AT ENERGY = , 1PE14.6, 16H AND POSSIBLY AT,
*      36H SMALLER ENERGIES IN THE SAME RANGE./9H YOU MUST,
*      43H CHANGE ENERGY INTERVALS TO AVOID BAD DATA]/
*      16H ****)
      END
C
C
C
      SUBROUTINE PUTSUM(RH, RL, ESIGMA, EDSIGM, FRACH, FRACL,
*      KH, KL, DUM)
C
C *** PURPOSE -- STORE INTEGRATED CROSS SECTION IN "SUM" MED DATA FILE
C
      DOUBLE PRECISION FILFIX, FILSUM, FILOUT
      COMMON /ODF/ FILFIX, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
*      IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
*      NDSTR
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      COMMON /REAL/ TDELAY, FPL, C(20)
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), FRACH(NENERG), FRACL(NENERG),
*      KH(NENERG), KL(NENERG), DUM(NENERG)
C
      IUSUM = 26
      IENER = -1
      IRUN = 0
      NEW = 1
      INSSUM = 7 + NPAR
      CALL ODFIO(IUSUM, FILSUM, IFB, NEW, INSSUM, NENERG, MODE,
*      NDSTR, IENER, IRUN)

```

```

C
      CALL OUTODF(IUSUM, IFB, INSSUM, 1, MODE, NDSTRT, 1,
*      NENERG, RH, 1)
      CALL OUTODF(IUSUM, IFB, INSSUM, 2, MODE, NDSTRT, 1,
*      NENERG, RL, 1)
      CALL OUTODF(IUSUM, IFB, INSSUM, 3, MODE, NDSTRT, 1,
*      NENERG, ESIGMA, 1)
      DO 20 IPAR=1,NPAR
         DO 10 I=1,NENERG
            DUM(I) = EDSIGM(IPAR,I)
10       CONTINUE
         CALL OUTODF(IUSUM, IFB, INSSUM, 3+IPAR, MODE,
*                  NDSTRT, 1, NENERG, DUM, 1)
20     CONTINUE
C
      CALL OUTODF(IUSUM, IFB, INSSUM, 4+NPAR, MODE, NDSTRT, 1,
*      NENERG, FRACH, 1)
      CALL OUTODF(IUSUM, IFB, INSSUM, 5+NPAR, MODE, NDSTRT, 1,
*      NENERG, FRACL, 1)
      CALL OUTODF(IUSUM, IFB, INSSUM, 6+NPAR, MODE, NDSTRT, 1,
*      NENERG, KH, 1)
      CALL OUTODF(IUSUM, IFB, INSSUM, 7+NPAR, MODE, NDSTRT, 1,
*      NENERG, KL, 1)
      RETURN
      END
C
C
C
      SUBROUTINE GETSUM(RH, RL, ESIGMA, EDSIGM, FRACH, FRACL,
*      KH, KL, DUM)
C
C *** PURPOSE -- READ INTEGRATED CROSS SECTION FROM ODF FILE
C
      DOUBLE PRECISION FILFIX, FILSUM, FILOUT
      COMMON /ODF/ FILFIX, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
*      IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
*      NDSTRT
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), FRACH(NENERG), FRACL(NENERG),
*      KH(NENERG), KL(NENERG), DUM(NENERG)
C
      IUSUM = 26
      CALL ODFIO(IUSUM, FILSUM, IFB, NEW, INSSUM, NENERG, MODE,
*      NDSTRT, IENER, IRUN)
C

```

```

C
      CALL INODF(IUSUM, IFB, INSSUM, 1, MODE, NDSTRT, 1,
*      NENERG, RH, 1)
      CALL INODF(IUSUM, IFB, INSSUM, 2, MODE, NDSTRT, 1,
*      NENERG, RL, 1)
      CALL INODF(IUSUM, IFB, INSSUM, 3, MODE, NDSTRT, 1,
*      NENERG, ESIGMA, 1)
      DO 20 IPAR=1,NPAR
          CALL INODF(IUSUM, IFB, INSSUM, 3+IPAR, MODE, NDSTRT,
*          1, NENERG, DUM, 1)
          DO 10 I=1,NENERG
              EDSIGM(IPAR,I) = DUM(I)
10      CONTINUE
20      CONTINUE
C
      CALL INODF(IUSUM, IFB, INSSUM, 4+NPAR, MODE, NDSTRT, 1,
*      NENERG, FRACH, 1)
      CALL INODF(IUSUM, IFB, INSSUM, 5+NPAR, MODE, NDSTRT, 1,
*      NENERG, FRACL, 1)
      CALL INODF(IUSUM, IFB, INSSUM, 6+NPAR, MODE, NDSTRT, 1,
*      NENERG, KH, 1)
      CALL INODF(IUSUM, IFB, INSSUM, 7+NPAR, MODE, NDSTRT, 1,
*      NENERG, KL, 1)
      RETURN
      END
C
C
C
      SUBROUTINE OUTPD(RH, RL, ESIGMA, EDSIGM, NAMPAR)
C
C *** PURPOSE -- OUTPUT INTERMEDIATE RESULTS
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), NAMPAR(8,NPAR)
C
C
      WRITE (32,99999)
      WRITE (32,99998)
      NPARMX = MIN0(NPAR,7)
      WRITE (32,99997) (IPAR,IPAR=1,NPARMX)
      WRITE (32,99996) ((NAMPAR(J,IPAR),J=1,2),IPAR=1,NPARMX)
      WRITE (32,99996) ((NAMPAR(J,IPAR),J=3,4),IPAR=1,NPARMX)
      WRITE (32,99996) ((NAMPAR(J,IPAR),J=5,6),IPAR=1,NPARMX)
      WRITE (32,99996) ((NAMPAR(J,IPAR),J=7,8),IPAR=1,NPARMX)
      DO 10 I=1,NENERG
          WRITE (32,99995) I, RL(I), RH(I), ESIGMA(I),
*          (EDSIGM(J,I),J=1,NPARMX)
10      CONTINUE

```

```

C
20 CONTINUE
  NPARMN = NPARMX + 1
  NPARMX = MIN0(NPARMX+9,NPAR)
  IF (NPARMX.LT.NPARMN) GO TO 40
  WRITE (32,99994)
  WRITE (32,99993) (IPAR,IPAR=NPARMN,NPARMX)
  WRITE (32,99992) ((NAMPAR(J,IPAR),J=1,2),IPAR=NPARMN,
*      NPARMX)
  WRITE (32,99992) ((NAMPAR(J,IPAR),J=3,4),IPAR=NPARMN,
*      NPARMX)
  WRITE (32,99992) ((NAMPAR(J,IPAR),J=5,6),IPAR=NPARMN,
*      NPARMX)
  WRITE (32,99992) ((NAMPAR(J,IPAR),J=7,8),IPAR=NPARMN,
*      NPARMX)
  DO 30 I=1,NENERG
    WRITE (32,99991) I, (EDSIGM(J,I),J=NPARMN,NPARMX)
30 CONTINUE
  GO TO 20
C
40 CONTINUE
  RETURN
99999 FORMAT (//43H ***** SUMMED CROSS SECTION AND PARTIAL ,
*      11HDERIVATIVES)
99998 FORMAT (/45H          EL          EH          SIGMA        DERIVATIV,
*      17HES WRT PARAMETERS)
99997 FORMAT (29X, 7I12)
99996 FORMAT (35X, 7(1X, 2A5, 1X))
99995 FORMAT (I5, 2F9.5, 1P8G12.4)
99994 FORMAT (/31H DERIVATIVES WRT PAR, CONTINUED)
99993 FORMAT (8H PAR NUM, I3, I10, 7I12)
99992 FORMAT (5X, 9(1X, 2A5, 1X))
99991 FORMAT (I5, 1P9G12.4)
  END

```


**APPENDIX D
FORTRAN Listing of File ALXALL.FOR**

```

SUBROUTINE FIXCOV(COVPAR, ESIGMA, EDSIGM, FRACH, FRACL,
*      KH, KL, VI, VO, VTOT)
C
C *** PURPOSE -- FIX COVARIANCE MATRIX FOR SUMMED CROSS SECTIONS
C
      DIMENSION COVPAR(1), ESIGMA(NENERG), EDSIGM(NPAR,NENERG),
*      FRACH(NENERG), FRACL(NENERG), KH(NENERG),
*      KL(NENERG), VI(JKM), VO(JKM), VTOT(JKM)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      COMMON /REAL/ TDELAY, FPL, C(20)
      COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*      DSOUT, DSP(30)
      COMMON /CHANL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
*      CHN, ICHN
C
C
      CALL TALK(KH, KL, FRACH, FRACL, NENERG)
C
      DO 10 JK=1,JKM
         VI(JK) = 0.
         VO(JK) = 0.0E0
10 CONTINUE
C
      ICHN = 0
20 CONTINUE
      MISSED = 0
30 ICHN = ICHN + 1
      IF (ICHN.GT.NCHN-NCHNMN) GO TO 80
      DO 40 J=1,NENERG
         IF (ICHN.GE.KH(J) .AND. ICHN.LE.KL(J)) GO TO 50
40 CONTINUE
      MISSED = MISSED + 1
      GO TO 30
50 CALL GETRF(MISSED)
      DEL = ELAST - ENOW
C
      JK = 0
      DO 70 J=1,NENERG
         DELJ = 0.0E0
         IF (ICHN.EQ.KH(J)) DELJ = FRACH(J)
         IF (ICHN.EQ.KL(J)) DELJ = FRACL(J)
         IF (ICHN.GT.KH(J) .AND. ICHN.LT.KL(J)) DELJ = DEL
         IF (DELJ.EQ.0.0E0) JK = JK + J
         IF (DELJ.EQ.0.0E0) GO TO 70
         AI = DSIN**2*CIN
         AO = DSOUT**2*COUT
         IF (AI.EQ.0.0E0 .AND. AO.EQ.0.0E0) WRITE (5,99999)
*           ICHN, CIN, COUT, DSIN, DSO
         IF (AI.EQ.0.0E0 .AND. AO.EQ.0.0E0) GO TO 20
         BI = AI*DELJ
         BO = AO*DELJ
70 CONTINUE

```

```

DO 60 K=1,J
    JK = JK + 1
    DELK = 0.0E0
    IF (ICHN.EQ.KH(K)) DELK = FRACH(K)
    IF (ICHN.EQ.KL(K)) DELK = FRACTL(K)
    IF (ICHN.GT.KH(K) .AND. ICHN.LT.KL(K)) DELK =
*
        DEL
    IF (DELK.EQ.0.0E0) GO TO 60
    VI(JK) = VI(JK) + BI*DELK
    VO(JK) = VO(JK) + BO*DELK
60      CONTINUE
70      CONTINUE
    GO TO 20
C
80      CALL TALKK(VI, NENERG, 1)
        CALL TALKK(VO, NENERG, 2)
C
DO 90 JK=1,JKM
    VTOT(JK) = VI(JK) + VO(JK)
90      CONTINUE
    IL = 0
    DO 140 IPAR=1,NPAR
        DO 130 LPAR=1,IPAR
            IL = IL + 1
            IF (COVPAR(IL).EQ.0.) GO TO 130
            JK = 0
            DO 120 J=1,NENERG
                IF (EDSIGM(IPAR,J).EQ.0. .AND.
*
                    EDSIGM(LPAR,J).EQ.0.) JK = JK + J
                IF (EDSIGM(IPAR,J).EQ.0. .AND.
*
                    EDSIGM(LPAR,J).EQ.0.) GO TO 120
                DO 110 K=1,J
                    JK = JK + 1
                    IF (EDSIGM(LPAR,K).EQ.0. .OR.
*
                        EDSIGM(IPAR,J).EQ.0.) GO TO 100
                    VTOT(JK) = EDSIGM(IPAR,J)*COVPAR(IL)*
                        EDSIGM(LPAR,K) + VTOT(JK)
100             CONTINUE
                    IF (EDSIGM(LPAR,J).EQ.0. .OR.
*
                        EDSIGM(IPAR,K).EQ.0.) GO TO 110
                    IF (IPAR.NE.LPAR) VTOT(JK) =
                        EDSIGM(LPAR,J)*COVPAR(IL)*
                        EDSIGM(IPAR,K) + VTOT(JK)
110             CONTINUE
120             CONTINUE
130             CONTINUE
140             CONTINUE
C
        CALL TALKK(VTOT, NENERG, 3)
C
        RETURN
99999 FORMAT (26H ICHN,CIN,COUT,DSIN,DSOUT=, I8, 4G14.6)
END

```

```

C
C
C      SUBROUTINE GETRF(MISSED)
C
C *** PURPOSE -- READ RAW COUNTS "XCIN" AND "XCOUT" FROM "RAW"
C ***          DATA FILE, AND ENERGY "XECHN", PARTIAL WRT XCIN
C ***          "XDTCIN", AND PARTIAL WRT XCOUT "XDTCOU" FROM
C ***          "FIX"ED DATA FILE
C
C      COMMON /ODF/ IUFIX(2), FILSUM(2), FILOUT(2), IURAW,
*           IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
*           INSOUT, MODE, NDSTRT
C      COMMON /CHANN2/ EMAX, EMIN, ELAST, ENOW, SIG, DSIN,
*           DSOUT, DSP(30)
C      COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
*           CHN, ICHN
C      COMMON /STORE/ XECHN(512), XDTCIN(512), XDTCOU(512),
*           XCIN(512), XCOUT(512)
REAL XECHN, XDTCIN, XDTCOU, XCIN, XCOUT
C      COMMON /REAL/ TDELAY, FPL, C(20)
C      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DATA MANY /512/, MAXCHN /0/, ICHNP /512/
C
ICHNP = ICHNP + MISSED
IF (ICHN.LE.MAXCHN) GO TO 20
IF (MAXCHN.EQ.0) ENOW = EMAX
10 CONTINUE
ICHNP = ICHNP - MANY
MINCHN = MAXCHN + 1
MINCHP = MINCHN + NCHNMN
MAXCHN = MAXCHN + MANY
MANYX = MANY
IF (ICHN.GT.MAXCHN) GO TO 10
IF (MAXCHN.GT.NCHN-NCHNMN) MAXCHN = NCHN - NCHNMN
IF (MAXCHN.EQ.NCHN-NCHNMN) MANYX = NCHN - MINCHN + 1 -
*     NCHNMN
C
C *** HERE WE NEED TO READ FROM ODF FILE TO GET A NEW CHUNK OF DATA
C
CALL INODF(IUFIX, IFB, INSFIX, 1, MODE, NDSTRT, MINCHN,
*           MANYX, XECHN, 1)
CALL INODF(IUFIX, IFB, INSFIX, 3, MODE, NDSTRT, MINCHN,
*           MANYX, XDTCIN, 1)
CALL INODF(IUFIX, IFB, INSFIX, 4, MODE, NDSTRT, MINCHN,
*           MANYX, XDTCOU, 1)
CALL INODF(IURAW, IFB, INSRAW, 4, MODE, NDSTRT, MINCHP,
*           MANYX, XCIN, 1)
CALL INODF(IURAW, IFB, INSRAW, 5, MODE, NDSTRT, MINCHP,
*           MANYX, XCOUT, 1)
C

```

```

C
20 ICHNP = ICHNP + 1
CIN = XCIN(ICHPN)
COUT = XCOUT(ICHPN)
ELAST = ENOW
ENOW = XECHN(ICHPN)
DSIN = XDTCIN(ICHPN)
DSOUT = XDTCOU(ICHPN)
C
RETURN
END
C
C
C
SUBROUTINE TALK(KH, KL, FRACH, FRACTL, NENERG)
DIMENSION KH(NENERG), KL(NENERG), FRACH(NENERG),
*      FRACTL(NENERG)
DO 10 I=1,NENERG
      WRITE (33,99999) I, KH(I), KL(I), FRACH(I), FRACTL(I)
10 CONTINUE
RETURN
99999 FORMAT (21H I,KH,KL,FRACH,FRACTL=, 3I6, 2F16.10)
END
C
C
C
SUBROUTINE TALKK(V, NENERG, K)
DIMENSION V(NENERG)
JK = 0
IF (K.EQ.1) WRITE (31,99999)
IF (K.EQ.2) WRITE (31,99998)
IF (K.EQ.3) WRITE (31,99997)
DO 10 J=1,NENERG
      WRITE (31,99996) J
      WRITE (31,99995) (V(JK+K),K=1,J)
      JK = JK + J
10 CONTINUE
RETURN
99999 FORMAT (//35H COVARIANCE DUE TO SAMPLE-IN COUNTS)
99998 FORMAT (//36H COVARIANCE DUE TO SAMPLE-OUT COUNTS)
99997 FORMAT (//17H TOTAL COVARIANCE)
99996 FORMAT (/18H DATA POINT NUMBER, I3)
99995 FORMAT (7X, 1P10G11.3)
END

```

```

SUBROUTINE FIXDER(DELPAR, EDSIGM, VI, VO, VTOT, DI, DO,
*      DTOT)

C *** PURPOSE -- FIX EDSIGM TO INCLUDE DELPAR, GENERATE
C ***           SQRT OF DIAGONALS OF V'S

C
DIMENSION DELPAR(NPAR), EDSIGM(NPAR,NENERG), VI(JKM),
*       VO(JKM), VTOT(JKM), DI(NENERG), DO(NENERG),
*       DTOT(NENERG)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*       NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DO 20 NERGY=1,NENERG
    DO 10 IPAR=1,NPAR
        IF (EDSIGM(IPAR,NERGY).GT.0.0)
*            EDSIGM(IPAR,NERGY) = EDSIGM(IPAR,NERGY)*
*            DELPAR(IPAR)
        IF (EDSIGM(IPAR,NERGY).LT.0.0)
*            EDSIGM(IPAR,NERGY) = -EDSIGM(IPAR,NERGY)*
*            DELPAR(IPAR)
10    CONTINUE
20    CONTINUE
    CALL SETD(VI, DI, NENERG, JKM)
    CALL SETD(VO, DO, NENERG, JKM)
    CALL SETD(VTOT, DTOT, NENERG, JKM)
    RETURN
END

C
C
C
SUBROUTINE SETD(V, D, N, NN)
C PURPOSE -- GENERATE D'S
DIMENSION V(NN), D(N)
C
II = 0
DO 10 I=1,N
    II = II + I
    D(I) = SQRT(V(II))
10 CONTINUE
RETURN
END

```

```

        SUBROUTINE OUTBUG(RH, RL, ESIGMA, EDSIGM, VI, VO, VTOT,
*      DI, DO, DTOT, NAMPAR, IC)
C PURPOSE -- OUTPUT RESULTS
        COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
        DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), VI(JKM), VO(JKM), VTOT(JKM),
*      DI(NENERG), DO(NENERG), DTOT(NENERG),
*      NAMPAR(8,NPAR), IC(NENERG)
        DIMENSION COVIN(10), COVOUT(10), COVTOT(10)
        DATA COVIN /
*      50H COVARIANCE MATRIX DUE TO SAMPLE-IN COUNTS      /
        DATA COVOUT /
*      50H COVARIANCE MATRIX DUE TO SAMPLE-OUT COUNTS     /
        DATA COVTOT /
*      50H TOTAL COVARIANCE MATRIX                         /
C
        CALL OUTALL(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT, NAMPAR)
        CALL OUTV(COVIN, VI, DI, IC, NENERG)
        CALL OUTV(COVOUT, VO, DO, IC, NENERG)
        CALL OUTV(COVTOT, VTOT, DTOT, IC, NENERG)
        RETURN
        END
C
C
        SUBROUTINE OUTV(TITLE, V, S, IC, N)
C
C *** PURPOSE -- OUTPUT TRIANGULAR VARIANCE V AS STD. DEV. AND CORRELATI
C
C      NML, MAY 1981
C
        DOUBLE PRECISION TITLE, HEADR
        DIMENSION TITLE(5), V(1), S(N), IC(N)
        DIMENSION STDDEV(10)
        DATA STDDEV /
*      50H STANDARD DEVIATION (SQRT OF DIAGONAL ELEMENTS)   /
        DATA HEADR /10HSTD. DEV. /
C
        WRITE (32,99999) TITLE
        IL = 0
        DO 20 I=1,N
          DO 10 L=1,I
            IL = IL + 1
            IF (I.EQ.L) GO TO 10
            IF (V(IL).NE.0.) GO TO 30
10      CONTINUE
20      CONTINUE
C
        WRITE (32,99998)
        CALL OUTSTR(STDDEV, HEADR, S, N)
        RETURN

```

```

30 WRITE (32,99997)
    NN = 0
    MANY = 30
    M = N/MANY
    MM = M*MANY
    IF (MM.NE.N) M = M + 1
    IMIN = 1 - MANY
    IMAX = MIN0(MANY,N)
    DO 80 J=1,M
        IMIN = IMIN + MANY
        MAX = (IMIN*(IMIN-1))/2
        WRITE (32,99996) (I,I=IMIN,IMAX)
        IMAX = IMAX + MANY
        IMAX = MIN0(IMAX,N)
        DO 70 I=IMIN,N
            MIN = MAX + IMIN
            MAX = MAX + I
            JMAX = I
            MM = JMAX - IMIN + 1
            IF (MM.GT.MANY) JMAX = IMIN + MANY - 1
            SI = S(I)
            IL = MIN - 1
            DO 40 L=IMIN,JMAX
                IL = IL + 1
                D = V(IL)*100./(S(L)*SI)
                IF (D.GT.0.) D = D + 0.5
                IF (D.LT.0.) D = D - 0.5
                IC(L) = D
40          CONTINUE
            IF (J.EQ.1) GO TO 60
            NOT = 0
            JJ = JMAX
            IF (JMAX.EQ.I) JJ = JJ - 1
            JJP = IMIN
            IF (JJP.GT.JJ) GO TO 60
            DO 50 L=JJP,JJ
                IF (IC(L).EQ.0) GO TO 50
                NOT = 1
50          CONTINUE
            IF (NOT.EQ.1) GO TO 60
            NN = 1
            GO TO 70
60          WRITE (32,99995) I, SI, (IC(L),L=IMIN,JMAX)
70          CONTINUE
80 CONTINUE
    RETURN
99999 FORMAT (//6H *****, 5A10)
99998 FORMAT (46H OFF-DIAGONAL ELEMENTS OF COVARIANCE MATRIX AR,
    *      11HE ALL ZERO.)
99997 FORMAT (/29H      STD. DEV.      CORRELATION)
99996 FORMAT (/15X, 30I4)
99995 FORMAT (I3, 1PG12.5, 30I4)
    END

```

```

C
C
      SUBROUTINE OUTSTR(TITLE, T1, A1, N)
C
C *** PURPOSE -- OUTPUT I VS A1(I) IN COLUMNS
C
C       NML, MAY 1981
C
      DOUBLE PRECISION TITLE, T1
      DIMENSION TITLE(5), A1(N)
      DATA PL /1H(/, PR /1H)/
C
      WRITE (32,99999) TITLE
      M = 6
      IF (N.LT.M) M = N
      WRITE (32,99998) (T1,I=1,M)
C
      M = N/6
      MM = M*6
      IF (MM.NE.N) GO TO 10
      MM = M
      GO TO 20
10   M = M + 1
      K = MM + 6 - N
      IF (K.GE.M) GO TO 60
      MM = M - K
20   CONTINUE
      DO 30 I=1,MM
          WRITE (32,99997) ((PL,L,PR,L=I+(K-1)*M,I+(K-1)*M),
          *                               A1(I+(K-1)*M),K=1,6)
30   CONTINUE
      IF (MM.EQ.M) GO TO 50
      MM = MM + 1
      DO 40 I=MM,M
          WRITE (32,99997) ((PL,L,PR,L=I+(K-1)*M,I+(K-1)*M),
          *                               A1(I+(K-1)*M),K=1,5)
40   CONTINUE
50   CONTINUE
      RETURN
60   WRITE (32,99997) (PL,I,PR,A1(I),I=1,N)
      RETURN
99999 FORMAT (//6H *****, 5A10)
99998 FORMAT (/8X, 5(A10, 10X), A10)
99997 FORMAT ((1X, A1, I3, A1, 1PG12.4, 5(3X, A1, I3, A1,
          *           1PG12.4)))
      END
C
C

```

```

C
      SUBROUTINE DIVIDE(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT)
C
C *** PURPOSE -- DIVIDE EVERYTHING BY ENERGY-DIFFERENCE
C
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*           EDSIGM(NPAR,NENERG), DI(NENERG), DO(NENERG),
*           DTOT(NENERG)
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DO 20 NERGY=1,NENERG
         Q = RH(NERGY) - RL(NERGY)
         Q = 1.0/Q
         ESIGMA(NERGY) = ESIGMA(NERGY)*Q
         DI(NERGY) = DI(NERGY)*Q
         DO(NERGY) = DO(NERGY)*Q
         DTOT(NERGY) = DTOT(NERGY)*Q
         DO 10 IPAR=1,NPAR
            EDSIGM(IPAR,NERGY) = EDSIGM(IPAR,NERGY)*Q
10      CONTINUE
20      CONTINUE
      RETURN
      END

C
C
C
      SUBROUTINE OUTFIN(RH, RL, ESIGMA, EDSIGM, VTOT, DI, DO,
*           DTOT, DUMMY, NAMPAR, IC)
C *** PURPOSE -- "OUT"PUT RESULTS
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*           EDSIGM(NPAR,NENERG), VTOT(JKM), DI(NENERG),
*           DO(NENERG), DTOT(NENERG), DUMMY(NENERG),
*           NAMPAR(NPAR), IC(NENERG)
      DIMENSION COVFIN(10)
      DATA COVFIN /
*           50H FINAL COVARIANCE MATRIX
      /
C
      CALL PUTOUT(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT, DUMMY)
      CALL OUTALL(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT, NAMPAR)
      CALL OUTVX(COVFIN, RH, RL, ESIGMA, VTOT, DTOT, DUMMY, IC,
*           NENERG)
      RETURN
      END
C

```

```

      SUBROUTINE PUTOUT(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT,
     *                   DUMMY)
C *** PURPOSE -- "OUT"PUT RESULTS ONTO ODF FILE --
C ***
C ***          AVERAGE ENERGY "(RH+RL)/2",
C ***
C ***          AVERAGED CROSS SECTION "ESIGMA",
C ***
C ***          SAMPLE-IN STATISTICAL UNCERTAINTY "DI",
C ***
C ***          SAMPLE-OUT STATISTICAL UNCERTAINTY "DO",
C ***
C ***          CONTRIBUTION "EDSIGM" TO THE UNCERTAINTY
C ***
C ***                  DUE TO EACH PARAMETER, AND
C ***          THE TOTAL UNCERTAINTY "DTOT".
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
     *             NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
     *             EDSIGM(NPAR,NENERG), DI(NENERG), DO(NENERG),
     *             DTOT(NENERG), DUMMY(NENERG)
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
     *             IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSPFIX, INSSUM,
     *             INSOUT, MODE, NDSTRT
C
      IUOUT = 21
      IENER = -1
      IRUN = 0
      NEW = 1
      INSOUT = 5 + NPAR
      CALL ODFIO(IUOUT, FILOUT, IFB, NEW, INSOUT, NENERG, MODE,
     *           NDSTRT, IENER, IRUN)
C
      DO 10 I=1,NENERG
         DUMMY(I) = (RH(I)+RL(I))*0.5E0
10   CONTINUE
C
      CALL OUTODF(IUOUT, IFB, INSOUT, 1, MODE, NDSTRT, 1,
     *             NENERG, DUMMY, 1)
      CALL OUTODF(IUOUT, IFB, INSOUT, 2, MODE, NDSTRT, 1,
     *             NENERG, ESIGMA, 1)
      CALL OUTODF(IUOUT, IFB, INSOUT, 3, MODE, NDSTRT, 1,
     *             NENERG, DI, 1)
      CALL OUTODF(IUOUT, IFB, INSOUT, 4, MODE, NDSTRT, 1,
     *             NENERG, DO, 1)
C
      DO 30 IPAR=1,NPAR
         DO 20 I=1,NENERG
            DUMMY(I) = EDSIGM(IPAR,I)
20   CONTINUE
         CALL OUTODF(IUOUT, IFB, INSOUT, 4+IPAR, MODE,
     *             NDSTRT, 1, NENERG, DUMMY, 1)
30   CONTINUE
         CALL OUTODF(IUOUT, IFB, INSOUT, 5+NPAR, MODE, NDSTRT, 1,
     *             NENERG, DTOT, 1)
      RETURN
      END

```

```

SUBROUTINE OUTALL(RH, RL, ESIGMA, EDSIGM, DI, DO, DTOT,
*      NAMPAR)
C PURPOSE -- OUTPUT RESULTS IN LINE-PRINTER FORM
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DIMENSION RH(NENERG), RL(NENERG), ESIGMA(NENERG),
*      EDSIGM(NPAR,NENERG), DI(NENERG), DO(NENERG),
*      DTOT(NENERG), NAMPAR(8,NPAR)
WRITE (32,99999)
WRITE (32,99998)
NPARMX = MIN0(NPAR,5)
WRITE (32,99997) (IPAR,IPAR=1,NPARMX)
WRITE (32,99996) ((NAMPAR(J,IPAR),J=1,2),IPAR=1,NPARMX)
WRITE (32,99996) ((NAMPAR(J,IPAR),J=3,4),IPAR=1,NPARMX)
WRITE (32,99996) ((NAMPAR(J,IPAR),J=5,6),IPAR=1,NPARMX)
WRITE (32,99996) ((NAMPAR(J,IPAR),J=7,8),IPAR=1,NPARMX)
DO 10 I=1,NENERG
     WRITE (32,99995) I, RL(I), RH(I), ESIGMA(I),
*                  DTOT(I), DI(I), DO(I), (EDSIGM(J,I),J=1,NPARMX)
10 CONTINUE
20 CONTINUE
NPARMN = NPARMX + 1
NPARMX = MIN0(NPARMX+9,NPAR)
IF (NPARMX.LT.NPARMN) GO TO 40
WRITE (32,99994)
WRITE (32,99993) (IPAR,IPAR=NPARMN,NPARMX)
WRITE (32,99992) ((NAMPAR(J,IPAR),J=1,2),IPAR=NPARMN,
*      NPARMX)
WRITE (32,99992) ((NAMPAR(J,IPAR),J=3,4),IPAR=NPARMN,
*      NPARMX)
WRITE (32,99992) ((NAMPAR(J,IPAR),J=5,6),IPAR=NPARMN,
*      NPARMX)
WRITE (32,99992) ((NAMPAR(J,IPAR),J=7,8),IPAR=NPARMN,
*      NPARMX)
DO 30 I=1,NENERG
     WRITE (32,99991) I, (EDSIGM(J,I),J=NPARMN,NPARMX)
30 CONTINUE
GO TO 20
40 CONTINUE
RETURN
99999 FORMAT (////42H **** Cross Sections And Contribution,
*      19HNS To Uncertainties)
99998 FORMAT (//44H          EL          EH          SIGMA        UNCERT. ,
*      49H (S-I)      (S-O)      DERIVATIVES WRT PAR, TIMES DEL,
*      3HPAR)
99997 FORMAT (55X, 5I12)
99996 FORMAT (59X, 5(1X, 2A5, 1X))
99995 FORMAT (I5, 2F9.5, 4F9.5, 1P5G12.4)
99994 FORMAT (/45H DERIVATIVES WRT PAR, TIMES DELPAR, CONTINUED)
99993 FORMAT (11H PAR NUMBER, I3, I10, 7I12)
99992 FORMAT (5X, 9(1X, 2A5, 1X))
99991 FORMAT (I5, 1P9G12.4)
END

```

```

SUBROUTINE OUTVX(TITLE, RH, RL, ESIGMA, V, S, DUMMY, IC,
*                 N)
C
C *** PURPOSE -- OUTPUT TRIANGULAR VARIANCE V AS STANDARD DEVIATION
C ***                         AND CORRELATION
C
C      NML, JUNE 1982
C
      DOUBLE PRECISION TITLE
      DIMENSION TITLE(5), RH(N), RL(N), ESIGMA(N), V(1), S(N),
*              DUMMY(N), IC(N)
C
      WRITE (32,99999) TITLE
C
      DO 10 I=1,N
         DUMMY(I) = (S(I)/ESIGMA(I))*100.
10 CONTINUE
C
      WRITE (32,99998)
C
      NN = 0
      MANY = 23
      M = N/MANY
      MM = M*MANY
      IF (MM.NE.N) M = M + 1
      IMIN = 1 - MANY
      IMAX = MIN0(MANY,N)
      DO 60 J=1,M
         IMIN = IMIN + MANY
         MAX = (IMIN*(IMIN-1))/2
         WRITE (32,99997) (I,I=IMIN,IMAX)
         IMAX = IMAX + MANY
         IMAX = MIN0(IMAX,N)
         DO 50 I=IMIN,N
            MIN = MAX + IMIN
            MAX = MAX + I
            JMAX = I
            MM = JMAX - IMIN + 1
            IF (MM.GT.MANY) JMAX = IMIN + MANY - 1
            SI = S(I)*(RH(I)-RL(I))
            IL = MIN - 1
            DO 20 L=IMIN,JMAX
               IL = IL + 1
               D = 0.
               IF (V(IL).NE.0.) D = V(IL)*100./(S(L)*SI*
*                               (RH(L)-RL(L)))
               IF (D.GT.0.) D = D + 0.5
               IF (D.LT.0.) D = D - 0.5
               IC(L) = D
20      CONTINUE

```

```
IF (J.EQ.1) GO TO 40
NOT = 0
JJ = JMAX
IF (JMAX.EQ.I) JJ = JJ - 1
JJP = IMIN
IF (JJP.GT.JJ) GO TO 40
DO 30 L=JJP,JJ
    IF (IC(L).EQ.0) GO TO 30
    NOT = 1
30   CONTINUE
    IF (NOT.EQ.1) GO TO 40
    NN = 1
    GO TO 50
40   WRITE (32,99996) I, RL(I), RH(I), DUMMY(I),
    *           (IC(L),L=IMIN,JMAX)
50   CONTINUE
60 CONTINUE
RETURN

C
99999 FORMAT (//6H *****, 5A10)
99998 FORMAT (/45H          ELOW   EHIGH   % STD.DEV.   CORRELAT,
    *      3HION)
99997 FORMAT (/34X, 23I4)
99996 FORMAT (I4, 2F9.5, F7.2, 5X, 23I4)
END
```

**APPENDIX E
FORTRAN Listing of File ALX10K.FOR**

```

C
C
C
      SUBROUTINE COUNT(NENERG, NUMMAX)
C
C *** PURPOSE -- COUNT NUMBER OF CHANNELS IN OUTPUT ODF FILE
C
      NENERG = 0
      NUMMAX = 0
10 CONTINUE
      READ (25,99999,END=20) MIN, MAX, NUMBER
      IF (MIN.EQ.0) GO TO 20
      MM = MAX - MIN + 1
      NENERG = MM/NUMBER + NENERG
      IF (MM.GT.NUMMAX) NUMMAX = MM
      GO TO 10
20 WRITE (5,99998) NENERG, NUMMAX
      REWIND 25
      RETURN
99999 FORMAT (3I)
99998 FORMAT (16H NUMBER OF BINS=, I8, 21H      MAXIMUM NUMBER OF,
*           30H CHANNELS TO READ AT ONE TIME=, I8)
      END
C
C
C
      SUBROUTINE EXTRA2(COVPAR, ECHN, SIGMA, DSIN, DSOUT,
*           DSPAR, CIN, COUT, NUMMAX, MOST)
C
C *** PURPOSE -- AVERAGE OVER CHANNELS TO GIVE FINAL RESULTS
C
C
      DIMENSION COVPAR(1), ECHN(MOST), SIGMA(MOST), DSIN(MOST),
*           DSOUT(MOST), DSPAR(MOST,NPAR), CIN(MOST), COUT(MOST)
C
      COMMON /STORE/ ENERGY(512), ESIGMA(512), STAT(512),
*           SYST(512), EDSIGM(30,512)
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
C
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
*           IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSSUM,
*           INSOUT, MODE, NDSTRT
C
      DATA MANY /512/, MAXCHN /512/, MINCHN /1/
C

```

```

        IF (NPAR.GT.30) STOP 55555
        CALL ZERO3
        NEWCHP = 0
10    CONTINUE
        READ (25,99998,END=60) MIN, MAX, NUMBER
        WRITE (5,99999) MIN, MAX, NUMBER
        IF (MIN.EQ.0) GO TO 60
C
        N = 0
        MMM = MIN
        NUMMER = MAX - MIN + 1
        CALL GETRFK(N, MMM, NUMMER, MOST, ECHN, SIGMA, DSIN,
*              DSOUT, DSPAR, CIN, COUT)
C
        DO 50 MM=MIN,MAX,NUMBER
            NEWCHP = NEWCHP + 1
C
        DO 30 NNN=1,NUMBER
            N = N + 1
            IF (N.GT.MOST) CALL GETRFK(N, MMM, NUMMER,
*                  MOST, ECHN, SIGMA, DSIN, DSOUT, DSPAR,
*                  CIN, COUT)
            ENERGY(NEWCHP) = ECHN(N) + ENERGY(NEWCHP)
            ESIGMA(NEWCHP) = SIGMA(N) + ESIGMA(NEWCHP)
            STAT(NEWCHP) = (CIN(N)*DSIN(N)**2+COUT(N)*
*                  DSOUT(N)**2) + STAT(NEWCHP)
            DO 20 KPAR=1,NPAR
                EDSIGM(KPAR,NEWCHP) = DSPAR(N,KPAR) +
                EDSIGM(KPAR,NEWCHP)
20      CONTINUE
30      CONTINUE
C
        ENERGY(NEWCHP) = ENERGY(NEWCHP)/FLOAT(NUMBER)
        ESIGMA(NEWCHP) = ESIGMA(NEWCHP)/FLOAT(NUMBER)
        STAT(NEWCHP) = STAT(NEWCHP)/FLOAT(NUMBER)**2
        DO 40 KPAR=1,NPAR
            EDSIGM(KPAR,NEWCHP) = EDSIGM(KPAR,NEWCHP)/
*                  FLOAT(NUMBER)
40      CONTINUE
C
        IF (NEWCHP.EQ.MANY) CALL PUTOK(COVPAR, MANY, NEWCHP)
50    CONTINUE
        GO TO 10
C
        60 MANYX = NEWCHP
        CALL PUTOK(COVPAR, MANYX, NEWCHP)
        RETURN
99999 FORMAT (16H MIN,MAX,NUMBER=, 3I6)
99998 FORMAT (3I)
        END

```

```

C
      SUBROUTINE ZERO3
C
C ***** PURPOSE -- ZERO ARRAYS
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
      COMMON /STORE/ ENERGY(512), ESIGMA(512), STAT(512),
      *      SYST(512), EDSIGM(30,512)
      DATA MANY /512/, NPARMX /30/
C
      MM = (NPARMX+4)*MANY
      DO 10 K=1,MM
         ENERGY(K) = 0.
10 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE GETRFK(N, MMM, NUMMER, MOST, ECHN, SIGMA,
      *      DSIN, DSOUT, DSPAR, CIN, COUT)
C
C *** PURPOSE -- READ ECHN, CIN, AND COUT FROM "RAW" DATA FILE
C ***          READ SIGMA, DSIN, DSOUT, AND DSPAR FROM "FIX"ED
C ***          DATA FILE
C
      DIMENSION ECHN(MOST), SIGMA(MOST), DSIN(MOST),
      *      DSOUT(MOST), DSPAR(MOST,NPAR), CIN(MOST), COUT(MOST)
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
      *      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
C
      COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
      *      IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
      *      INSOUT, MODE, NDSTRRT
C
      IF (N.EQ.0) GO TO 10
      N = N - MOST
      NUMMER = NUMMER - MOST
      MMM = MMM + MOST
C
10 CONTINUE
      MOSTX = MOST
      IF (NUMMER.LT.MOSTX) MOSTX = NUMMER
      MMP = MMM - NCHNMN
C

```

```

      CALL INODF(IURAW, IFB, INSRAW, 1, MODE, NDSTRT, MMM,
*      MOSTX, ECHN, 1)
      CALL INODF(IUFIX, IFB, INSFIX, 2, MODE, NDSTRT, MMP,
*      MOSTX, SIGMA, 1)
      CALL INODF(IUFIX, IFB, INSFIX, 3, MODE, NDSTRT, MMP,
*      MOSTX, DSIN, 1)
      CALL INODF(IUFIX, IFB, INSFIX, 4, MODE, NDSTRT, MMP,
*      MOSTX, DSOUT, 1)
      DO 20 KPAR=1,NPAR
         CALL INODF(IUFIX, IFB, INSFIX, 4+KPAR, MODE, NDSTRT,
*                  MMP, MOSTX, DSPAR(1,KPAR), 1)
20 CONTINUE
      CALL INODF(IURAW, IFB, INSRAW, 4, MODE, NDSTRT, MMM,
*      MOSTX, CIN, 1)
      CALL INODF(IURAW, IFB, INSRAW, 5, MODE, NDSTRT, MMM,
*      MOSTX, COUT, 1)
C
      RETURN
      END
C
C
C
      SUBROUTINE PUTOK(COVPAR, MANYX, NEWCHP)
C
C *** PURPOSE -- WRITE FINAL "OUT"PUT FILE (FILOUT)
C
      DOUBLE PRECISION FILFIX, FILOUT, FILSUM
      DIMENSION COVPAR(1)
      COMMON /STORE/ ENERGY(512), ESIGMA(512), STAT(512),
*                 SYST(512), EDSIGM(30,512)
C
      COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*                 NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
C
      COMMON /ODF/ FILFIX, FILSUM, FILOUT, IURAW, IUFIX, IUSUM,
*                 IUOUT, IFB, INSRAW, INSFIX, INSSUM, INSOUT, MODE,
*                 NDSTRT
C
      DATA MANY /512/, MINCHN /1/, KKKKKK /0/
C
      IF (KKKKKK.NE.0) GO TO 10
      KKKKKK = 1
      IUOUT = 21
      IENER = -1
      IRUN = 0
      NEW = 1
      INSOUT = 8
      CALL ODFIO(IUOUT, FILOUT, IFB, NEW, INSOUT, NENERG, MODE,
*                 NDSTRT, IENER, IRUN)
10 CONTINUE
C

```

```

CALL OUTODF(IUOUT, IFB, INSOUT, 1, MODE, NDSTRT, MINCHN,
*      MANYX, ENERGY, 1)
CALL OUTODF(IUOUT, IFB, INSOUT, 2, MODE, NDSTRT, MINCHN,
*      MANYX, ESIGMA, 1)

C
C *** CALCULATE SYSTEMATIC UNCERTAINTIES FROM PARAMETER UNCERTAINTIES
IK = 0
DO 40 IPAR=1,NPAR
    DO 30 KPAR=1,IPAR
        IK = IK + 1
        IF (COVPAR(IK).EQ.0.) GO TO 30
        DO 20 K=1,MANYX
            IF (EDSIGM(IPAR,K).EQ.0. .OR.
*                 EDSIGM(KPAR,K).EQ.0.) GO TO 20
            SYST(K) = SYST(K) + EDSIGM(IPAR,K)*
*                 COVPAR(IK)*EDSIGM(KPAR,K)
            IF (IPAR.NE.KPAR) SYST(K) = SYST(K) +
*                 EDSIGM(IPAR,K)*COVPAR(IK)*EDSIGM(KPAR,
*                 K)
20         CONTINUE
30         CONTINUE
40 CONTINUE

C
DO 50 K=1,MANYX
    ENERGY(K) = SQRT(STAT(K)+SYST(K))
C
50 CONTINUE
C
DO 60 K=1,MANYX
    STAT(K) = SQRT(STAT(K))
60 CONTINUE
C
DO 70 K=1,MANYX
    SYST(K) = SQRT(SYST(K))
70 CONTINUE
C
CALL OUTODF(IUOUT, IFB, INSOUT, 3, MODE, NDSTRT, MINCHN,
*      MANYX, STAT, 1)
CALL OUTODF(IUOUT, IFB, INSOUT, 4, MODE, NDSTRT, MINCHN,
*      MANYX, SYST, 1)
CALL OUTODF(IUOUT, IFB, INSOUT, 5, MODE, NDSTRT, MINCHN,
*      MANYX, ENERGY, 1)

C
DO 80 K=1,MANYX
    STAT(K) = 100.*STAT(K)/ESIGMA(K)
80 CONTINUE
C
DO 90 K=1,MANYX
    SYST(K) = 100.*SYST(K)/ESIGMA(K)
90 CONTINUE
C

```

```
DO 100 K=1,MANYX
    ENERGY(K) = 100.*ENERGY(K)/ESIGMA(K)
100 CONTINUE
C
    CALL OUTODF(IUOUT, IFB, INSOUT, 6, MODE, NDSTRT, MINCHN,
*      MANYX, STAT, 1)
    CALL OUTODF(IUOUT, IFB, INSOUT, 7, MODE, NDSTRT, MINCHN,
*      MANYX, SYST, 1)
    CALL OUTODF(IUOUT, IFB, INSOUT, 8, MODE, NDSTRT, MINCHN,
*      MANYX, ENERGY, 1)
C
    NEWCHP = 0
    MINCHN = MINCHN + MANYX
    IF (MANYX.EQ.MANY) CALL ZERO3
    RETURN
    END
```


APPENDIX F
FORTRAN Listing of File ALXPAR.FOR

```

C
C      THIS IS FOR DUMMY CASE FOR ILLUSTRATION
C          FOR USE IN TM REPORT
C
C
C      SUBROUTINE INPUT(RH, RL, KH, KL, NNN)
C
C *** PURPOSE -- READ INPUT FROM "FILE2"
C
        DOUBLE PRECISION FILEQ, TITLE, FILEC(1)
        DOUBLE PRECISION TIMEDE, CRUNCH, ENGINT, CHANLB, CHNINT
        DIMENSION FILEQ(3)
        DIMENSION RH(NNN), RL(NNN), KL(NNN), KH(NNN)
        COMMON /REAL/ TDELAY, FPL, C(20)
        COMMON /ODF/ FILFIX(2), FILSUM(2), FILOUT(2), IURAW,
*           IUFIX, IUSUM, IUOUT, IFB, INSRAW, INSFIX, INSSUM,
*           INSOUT, MODE, NDSTRT
        COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*           NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
        DATA TIMEDE /10HTIME DELAY/, CRUNCH /10HCRUNCH BO/,
*           ENGINT /10HENERGY INT/, CHANLB /10HCHANNEL BO/,
*           CHNINT /10HCHANNEL IN/
        DATA WME /2HME/, WKE /2HKE/, WEV /2HEV/
C
C
C *** FILE NAME FOR RAW DATA
C
        READ (24,99999) FILEQ, NCHNXX, NCHNMN
        IURAW = 23
        CALL ODFIO(IURAW, FILEQ, IFB, NEW, INSRAW, NCHN, MODE,
*           NDSTRT, IENER, IRUN)
        IF (NCHNXX.NE.0) NCHN = NCHNXX
        IF (NCHNMN.EQ.0) NCHNMN = 1
        WRITE (32,99998) NCHN, NCHNMN
C        WRITE (5,99998) NCHN, NCHNMN
        READ (24,99999) TITLE
C
C
C *** TIME DELAY, FLIGHT PATH LENGTH
C
        READ (24,99999) TITLE
        IF (TITLE.NE.TIMEDE) STOP 31
        READ (24,99997) TDELAY, FPL
C *** TIME DELAY IN NS, FLIGHT PATH LENGTH IN MM
        WRITE (32,99996) TDELAY, FPL
        READ (24,99999) TITLE
C

```

```

C *** CRUNCH BOUNDARIES
C
      READ (24,99999) TITLE
      IF (TITLE.NE.CRUNCH) STOP 32
      WRITE (32,99995)
      L = 0
10   L = L + 1
      READ (24,99994) M(L), C(L)
C *** C(L)=CHANNEL WIDTH IN NANoseconds OF CRUNCH SECTION L
C *** M(L)=NUMBER OF CHANNELS IN CRUNCH SECTION L
C ***      (CHANGE TO STARTING CHANNEL NUMBER)
      IF (M(L).GT.0) GO TO 10
      NBINSZ = L - 1
      IF (NBINSZ.GT.20) STOP 33
      WRITE (32,99993) (M(L),C(L),L=1,NBINSZ)
      NBNSZ1 = NBINSZ + 1
      DO 20 LL=1,NBNSZ1
          L = 1 + NBINSZ - LL
          M(L) = M(L-1)
20   CONTINUE
      M(1) = 1
      MM = 0
      DO 30 LL=1,NBNSZ1
          MM = MM + M(LL)
          M(LL) = MM
30   CONTINUE
C     WRITE (05,99993) (M(L),C(L),L=1,NBINSZ,1)
C
C
C *** ENERGY INTERVALS FOR REPORTING RESULTS
C
      READ (24,99992) TITLE, WHATEV, FILEC
      IF (TITLE.NE.ENGIN) GO TO 100
C
C *** HERE INTERVALS ARE GIVEN AS "ENERIES" DIRECTLY
      J = 0
40   J = J + 1
      READ (24,99997) RL(J), RH(J)
C *** RL=LOWER ENERGY, RH=HIGHER ENERGY
      IF (RL(J).NE.0.) GO TO 40
      JRS = J - 1
      JRSS = JRS - 1
      DO 50 J=1,JRSS
          IF (RH(J).EQ.0.) RH(J) = RL(J+1)
50   CONTINUE
      IF (RH(JRS).EQ.0.) JRS = JRS - 1
      NENERG = JRS
      IF (NENERG.GT.NNN) STOP 34
      IF (WHATEV.EQ.WME) GO TO 90
      IF (WHATEV.EQ.WKE) GO TO 70
      IF (WHATEV.NE.WEV) STOP 35
C

```

```

C *** ENERGY INTERVALS ARE GIVEN IN EV.  CONVERT TO MEV
DO 60 J=1,NENERG
  RL(J) = RL(J)*0.000001
  RH(J) = RH(J)*0.000001
60 CONTINUE
GO TO 90
C
C *** ENERGY INTERVALS ARE GIVEN IN KEV.  CONVERT TO MEV
70 DO 80 J=1,NENERG
  RL(J) = RL(J)*0.001
  RH(J) = RH(J)*0.001
80 CONTINUE
C
C *** NOW ENERGY INTERVALS ARE IN MEV.
90 WRITE (32,99991)
  WRITE (32,99990) (J,RL(J),RH(J),J=1,NENERG)
RETURN
C
C
C *** CHANNEL NUMBER ARE GIVEN (MIN,MAX AND INCREMENT) TO USE
C *** AS ENERGY INTERVALS.  IE, DIVIDE CHANNEL NUMBERS KMIN TO
C *** KMAX INTO INTERVALS, EACH OF WHICH IS INC CHANNELS WIDE.
C
100 CONTINUE
  IF (TITLE.NE.CHNINT) GO TO 170
  J = 0
110 CONTINUE
  READ (24,99989) KMIN, KMAX, INC
  IF (KMIN.EQ.0) GO TO 140
120 CONTINUE
  IF (((KMAX-KMIN+1)/INC)*INC.NE.(KMAX-KMIN+1)) GO TO 160
  IF (KMAX.GT.NCHN) GO TO 160
  IF (KMIN.GT.KMAX) GO TO 140
  J = J + 1
  KH(J) = KMIN
  DO 130 K=KMIN+INC,KMAX,INC
    KL(J) = K - 1
    J = J + 1
    KH(J) = K
130 CONTINUE
  KL(J) = KMAX
  GO TO 110
140 NENERG = J
  IF (NENERG.GT.NNN) STOP 36
  DO 150 J=1,NENERG
    KH(J) = KH(J) - NCHNMN
    KL(J) = KL(J) - NCHNMN
150 CONTINUE
RETURN

```

```

160 CONTINUE
    IF (KMAX.GT.NCHN) KMAX = NCHN
    WRITE (5,99988) KMIN, KMAX, INC, NCHN
    N = (KMAX-KMIN+1)/INC
    KMAX = KMIN - 1 + N*INC
    WRITE (5,99987) KMIN, KMAX, INC, NCHN
    GO TO 120

C
C *** CHANNEL NUMBERS ARE TO BE READ FROM AUXILIARY FILE. THERE
C *** ARE LOTS OF BINS SO DON'T GENERATE COVARIANCE MATRIX,
C *** JUST THE DIAGONAL ELEMENTS.
C
170 IF (TITLE.NE.CHANLB) STOP 37
    READ (24,99999) TITLE
    NENERG = -1
    OPEN (UNIT=25, FILE='ALXAUX.DAT', ACCESS='SEQIN',
*          DIALOG=FILEC)
    RETURN

C
99999 FORMAT (3A10, 2I10)
99998 FORMAT (13H NCHN,NCHNMN=, 2I10)
99997 FORMAT (2F)
99996 FORMAT (46H TIME DELAY (NS) AND FLIGHT PATH LENGTH (MM) A,
*              2HRE, G14.6, 4HAND , G14.6)
99995 FORMAT (/45H CRUNCH BOUNDARIES AND CHANNEL WIDTHS IN NANO,
*              7HSECONDS)
99994 FORMAT (I, F)
99993 FORMAT (I8, F12.5)
99992 FORMAT (A10, 8X, A2, 31X, A10)
99991 FORMAT (/24H ENERGY INTERVALS IN MEV)
99990 FORMAT (I5, 2F16.6)
99989 FORMAT (3I)
99988 FORMAT (44H OOPS -- CHANNEL LIMITS ARE OUT OF BOUNDS --/
*              1X, 5I10)
99987 FORMAT (36H NEW VALUES OF CHANNEL LIMITS ARE --/1X, 5I10)
END

C
C
C
SUBROUTINE SETPAR(COVPAR, DELPAR, NAMPAR)
C
C *** PURPOSE -- INITIALIZE PARAMETERS AND COVARIANCE MATRIX
C
        DOUBLE PRECISION TITLE, PRMTRS, CVRNCS
        DIMENSION COVPAR(1), DELPAR(NPAR), NAMPAR(8,NPAR)
        COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*              JDEP(30,100), INPAR(30)
        COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*              NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
        DATA PRMTRS /10HPARAMETERS/, CVRNCS /10HCOVARIANCE/

```

```

C
READ (24,99999) TITLE
IF (TITLE.NE.PRMTRS) WRITE (5,99998) TITLE
IF (TITLE.NE.PRMTRS) STOP 36
C
WRITE (32,99997)
DO 10 IPAR=1,NPARAM
    READ (24,99996,END=20) INPAR(IPAR), PAR(IPAR),
*        DELPAR(IPAR), DDD, (NAMPAR(K,IPAR),K=1,8)
    IF (PAR(IPAR).EQ.0. .AND. DELPAR(IPAR).EQ.0. .AND.
*        DDD.EQ.0.) GO TO 20
    IF (DELPAR(IPAR).EQ.0.) DELPAR(IPAR) =
*        ABS(PAR(IPAR))*DDD*0.01
    IF (DDD.EQ.0.) DDD = DELPAR(IPAR)/ABS(PAR(IPAR))*100.
    WRITE (32,99995) IPAR, INPAR(IPAR), PAR(IPAR),
*        DELPAR(IPAR), DDD, (NAMPAR(K,IPAR),K=1,8)
10 CONTINUE
WRITE (5,99994)
IPAR = NPARAM + 1
READ (24,99996) J
C
20 NPARAM = IPAR - 1
C
IJ = 0
DO 40 IPAR=1,NPARAM
    DO 30 JPAPR=1,IPAR
        IJ = IJ + 1
        COVPAR(IJ) = 0.0D0
        IF (JPAPR.EQ.IPAR) COVPAR(IJ) = (DELPAR(IPAR))**2
30     CONTINUE
40 CONTINUE
C
READ (24,99999,END=60) TITLE
IF (TITLE.NE.CVRNCS) GO TO 60
50 CONTINUE
READ (24,99993,END=60) IPAR, JPAPR, COV, CORR
IF (IPAR.GT.NPARAM .OR. JPAPR.GT.NPARAM) STOP 310
IF (IPAR.LE.0 .OR. JPAPR.LE.0) GO TO 60
II = MAX0(IPAR,JPAPR)
JJ = MIN0(IPAR,JPAPR)
IJ = (II*(II-1))/2 + JJ
IF (COV.NE.0.0D0) COVPAR(IJ) = COV
IF (COV.EQ.0.0D0) COVPAR(IJ) = CORR*DELPAR(IPAR)*
*        DELPAR(JPAPR)
GO TO 50
C

```

```

60 CONTINUE
CLOSE (UNIT=24)
RETURN
99999 FORMAT (A10)
99998 FORMAT (1X, A10)
99997 FORMAT (//4H PAR USED PARAMETER UNCERTAINTY %,
*      5H UNCE, 7HRTAINTY/20H NO. ???? )
99996 FORMAT (I, 3F, T28, 8A5)
99995 FORMAT (1X, 2I5, F16.6, F13.6, F10.3, 3X, 8A5)
99994 FORMAT (/4H CAREFUL -- IF YOU HAVE MORE THAN 30 PARAMETE,
*      3HRS,/4H           THE CODE ALEX SEES ONLY 30 OF THEM])
*      /12X, 44HTO USE MORE, INCREASE DIMENSIONS IN /PARAM/ .
*      /)
99993 FORMAT (2I, 2F)
END
C
C
C
SUBROUTINE PREP(TYPE, USE, IDEP, IFUNC)
C
C *** PURPOSE -- DESCRIBE THE VARIOUS TYPES OF FUNCTIONS
C ***          AND GIVE RELEVANT CONSTANTS AND PARAMETERS
C ***          FOR EACH
C ***
C ***      TYPE=NORMALIZATION
C ***          SAMPLE-IN
C ***          SAMPLE-OUT
C ***
C ***      USE =BACKGROUND
C ***          DEADTIME CORRECTION
C ***
C ***      IDEP(1)=NUMBER OF PARAMETER WHICH IS RELEVANT TO THIS
C ***          FUNCTION
C ***      IDEP(2)=NUMBER OF SECOND RELEVANT PARAMETER
C ***          ETC.
C
DOUBLE PRECISION TYPE, USE, ANORM, SAMIN, SAMOUT, DTIME,
*      BACKG, BLANK
DIMENSION IDEP(NPAR)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
DATA ANORM /10HNORMALIZAT/, SAMIN /10HSAMPLE-IN /, SAMOUT
*      /10HSAMPLE-OUT/, DTIME /10HDEADTIME C/, BACKG /
*      10HBACKGROUND/, BLANK /10H           /
C
NFUNC = 9
GO TO (10, 20, 30, 40, 50), IFUNC
C

```

```

C *** OVERALL NORMALIZATION ON TRANSMISSION
10 TYPE = ANORM
    IDEP(1) = 2
C           MONITOR-IN
    IDEP(2) = 3
C           MONITOR-OUT
    NTHICK = 1
C           PARAMETER NUMBER FOR "THICKNESS OF SAMPLE"
    RETURN
C
C *** DEAD TIME CORRECTION
20 TYPE = SAMIN
    USE = DTIME
    RETURN
C
C *** DEAD TIME CORRECTION
30 TYPE = SAMOUT
    USE = DTIME
    RETURN
C
C *** BACKGROUND
40 TYPE = SAMIN
    USE = BACKG
    IDEP(1) = 4
    RETURN
C
C *** BACKGROUND FOR SAMPLE-OUT
50 TYPE = SAMOUT
    USE = BACKG
    IDEP(1) = 5
    RETURN
    END
C
C
C
FUNCTION FUNC(IFUNC)
C
C *** EVALUATE VALUE OF FUNCTION IFUNC FOR CHANNEL ICHN AT
C *** ENERGY ECHN AND TIME TCHN
C
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
C
COMMON /CHANNL/ CIN, COUT, DTCIN, DTOUT, ECHN, TCHN,
*      CHN, ICHN
C
GO TO (10, 20, 30, 40, 50), IFUNC
C
C *** NORMALIZATION
10 FUNC = PAR(3)/PAR(2)
    RETURN

```

```

C
C *** DEAD TIME CORRECTION FOR SAMPLE-IN
20 FUNC = DTCIN
    RETURN
C
C *** DEAD TIME CORRECTION FOR SAMPLE-OUT
30 FUNC = DTCOUT
    RETURN
C
C *** BACKGROUND FOR SAMPLE-IN
40 CONTINUE
    FUNC = PAR(4)*CHN
    RETURN
C
C *** BACKGROUND FOR SAMPLE-OUT
50 CONTINUE
    FUNC = PAR(5)*CHN
    RETURN
    END
C
C
C
FUNCTION DERIV(IFUNC, KPAR)
C
C *** EVALUATE DERIVATIVE OF FUNCTION IFUNC WRT PARAMETER K,
C *** AT CHANNEL ICHN WITH ENERGY ECHN AND TIME TCHN
C
COMMON /PARAM/ FFUNC(5), CDELP(5), PAR(30), IWHERE(35),
*      JDEP(30,100), INPAR(30)
COMMON /INTEGR/ NPAR, NFUNC, NCHN, NUMBER, NBINSZ,
*      NENERG, JKM, M(21), NCHNMN, NTHICK, NPARAM
COMMON /CHANNL/ CIN, COUT, DTCIN, DTCOUT, ECHN, TCHN,
*      CHN, ICHN
C
C
DERIV = 0.
GO TO (10, 50, 50, 30, 40), IFUNC
C
C *** NORMALIZATION
10 CONTINUE
IF (KPAR.NE.2) GO TO 20
DERIV = -PAR(3)/PAR(2)**2
RETURN
20 IF (KPAR.NE.3) GO TO 60
DERIV = 1.0/PAR(2)
RETURN
C
C *** BACKGROUND FOR SAMPLE-IN
30 CONTINUE
IF (KPAR.NE.4) GO TO 60
DERIV = CHN
RETURN

```

```
C
C *** BACKGROUND FOR SAMPLE-OUT
40 CONTINUE
  IF (KPAR.NE.5) GO TO 60
  DERIV = CHN
  RETURN

C
C *** DEADTIME CORRECTION
50 CONTINUE
  RETURN

C
60 WRITE (5,99999) ICHN, IFUNC, KPAR
  RETURN

C
99999 FORMAT (39H GOT LOST IN DERIV FOR ICHN,IFUNC,KPAR=, I8,
*           2I5)
END
```

ORNL/TM-8676
ENDF-332

INTERNAL DISTRIBUTION

1-2.	L. S. Abbott	35.	B. F. Maskowitz
3.	G. F. Auchampaugh (Consultant)	36.	G. S. McNeilly
4.	Z. W. Bell	37.	P. S. Meszaros
5.	R. F. Carlton (Consultant)	38.	B. D. Murphy
6.	H. P. Carter/G. E. Whitesides/ CS Library	39.	D. K. Olsen
7.	J. G. Craven	40.	R. W. Peele
8.	J. W. Dabbs	41.	C. M. Perey
9.	J. K. Dickens	42.	F. G. Perey
10.	G. de Saussure	43.	R. B. Perez
11.	C. Y. Fu	44.	S. Raman
12.	P. W. Gaffney	45.	M. Rhoades-Brown
13.	G. Goldstein (Consultant)	46.	R. Satchler
14.	P. Greebler (Consultant)	47.	R. O. Sayer
15.	R. Gwin	48.	R. R. Spencer
16.	J. A. Harvey	49.	R. E. Uhrig (Consultant)
17.	D. M. Hetrick	50.	L. W. Weston
18.	J. T. Holdeman	51.	R. Wilson (Consultant)
19.	D. J. Horen	52.	R. Winters (Consultant)
20.	C. H. Johnson	53.	A. Zucker
21-25.	D. C. Larson	54.	Central Research Library
26-30.	N. M. Larson	55.	K-25 Plant Library
31.	W. B. Lowenstein (Consultant)	56.	ORNL Y-12 Technical Library
32.	I. Y. Lee	57.	Document Reference Section
33.	R. L. Macklin	58-59.	Laboratory Records Department
34.	F. C. Maienschein	60.	ORNL Patent Office
		61.	Laboratory Records (RC)

EXTERNAL DISTRIBUTION

- 62. Chief, Mathematics and Geoscience Branch, DOE, Washington, DC 20545.
- 63. DOE Oak Ridge Operations, Reactor Division, P. O. Box E, Oak Ridge, TN 37830.
- 64. DOE Division of Reactor Research and Development, Washington, DC 20545.
- 65. Office of Assistant Manager, Energy Research and Development, Oak Ridge, TN 37830.
- 66-92. Technical Information Center, P. O. Box 62, Oak Ridge, TN 37830.