science Applications, INC.

MASTER

DISCLAIMER

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

DISCLAIMER

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

DISCLAIMER

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

ANALYSIS OF AIRBORNE RADIOMETRIC DATA

VOLUME 2

DESCRIPTION, LISTING, AND OPERATING
INSTRUCTIONS FOR THE CODE
DELPHI/MAZAS

FINAL REPORT

by

M. Sperling
D. C. Shreve

December 1, 1978

Science Applications, Inc.
P. O. BOX 2351
La Jolla, California 92037

Work performed under Bendix Field Engineering
Corporation, Grand Junction Operations
Subcontract No. 77-041-L

prepared for

U. S. Energy Research and Development Administration
Grand Junction, Colorado 81501

SCIENCE APPLICATIONS, LA JOLLA, CALIFORNIA
ALBUQUERQUE • ANN ARBOR • ARLINGTON • ATLANTA • BOSTON • CHICAGO • HUNTSVILLE
LOS ANGELES • McLEAN • PALO ALTO • SANTA BARBARA • SUNNYVALE • TUCSON

P.O. Box 2351, 1200 Prospect Street, La Jolla, California 92037

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

"This report was prepared as an account of work sponsored in part by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of the contractors, subcontractors, or 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."

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
1. INTRODUCTION	1
2. OPERATING INSTRUCTIONS	7
3. DELPHI GRAPHICAL OUTPUT.	16
4. SPECIAL INPUT CHARACTERISTICS OF RESPONSE FUNCTIONS AND DATA FILES	29
4.1 Response Functions.	29
4.2 Data Files.	29
5. LISTING OF CODE.	30

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

1. INTRODUCTION

The computer code DELPHI is an interactive English language command system for the analysis of airborne radio-metric data. The code includes modules for data reduction, data simulation, time filtering, data adjustment and graphical presentation of the results.

At the present time, DELPHI is implemented in FORTRAN on a DEC-10 computer. Operator interaction is by means of a Tektronix graphics terminal, primarily a Tektronix 4006. However, modular architecture has been utilized with microcomputer applications in view. This is illustrated in Fig.1 which shows the program file structure. In addition to DELPHI itself, the major functional components are as follows:

1. INPUT: Six subroutines for altering the command vocabulary. Four subroutines for transferring vectors and matrices. Three subroutines for altering control parameters. One subroutine for disk transfer.
2. OUTPUT: Seven subroutines for graphical displays of vectors, including discrete line spectra. Five subroutines for graphical displays of matrices, including columns as vector sequences, and surfaces with hidden line suppression. Two subroutines for displaying control parameters. One subroutine for displaying the command vocabulary. One subroutine for disk transfer.
3. UNFOLD: Thirteen subroutines for analyzing data with admixtures of continuum and discrete components.
4. SIMULATION: Thirteen subroutines for simulating response matrices and data.
5. GENERATION: Presently empty file, but allocated for subroutines under separate development for expeditious construction of actual spectrometer responses.
6. VECTOR: Eleven short, fast subroutines for vector algebra.

LEVEL

FIGURE 1. PROGRAM FILE STRUCTURE

7. MATRIX: Five subroutines for matrix algebra, including Gram-Schmidt orthogonalization.

DELPHI contains six subroutines, four for command interpretation, one for interpretation of switch parameters (explained below), and one for command recognition and initiation of utility sequences.

The hierarchy of data structures is illustrated in Fig. 2 . Processes are initiated by commands, and commands are sequences or vectors of vocabulary entries, their proper order being decipherable by grammatic considerations. However, even when a process is specified, it can have many variants, depending upon values assumed by parameters that are either scalars or vectors of low dimensionality. Four types of parameters are distinguished in DELPHI, as follows:

1. Transfer Parameters: Specifications relevant to I/O operations, such as the name of the input file.
2. Display Parameters: Specifications relevant to output graphics, such as the perspective for viewing a matrix surface.
3. Unfold Parameters: Specifications relevant for employment of the data reduction algorithms, such as the number of data channels.
4. Simulation Parameters: Specifications relevant for construction of data and responses, such as the assumed detector size.

Within each parameter block, there are eight sixteen-position switches capable of altering modes of operation such as the functionally active unfolding algorithm. In practice, these switches have been better used for distinguishing between binary states, and they should perhaps be reconfigured to permit 32 (or actually 35-36 with the DEC-10 36 bit word) binary positions.

FIGURE 2. DATA BUFFERS

The command structure of DELPHI is reflexive in that it permits insertion or deletion of vocabulary data, including the definition of synonyms. For example, to display the part of the detector response matrix associated with discrete spectral lines, one might command

DISPLAY THE DISCRETE RESPONSE

However, if it were first commanded

MAKE 'K' SYNONYMOUS WITH 'RESPONSE'

an equivalent display command could be stated as

DISPLAY K(DISCRETE)

The number of lines represented in the discrete response is not stated explicitly, but rather is implicit in the parameter structure, which can also be changed by command. Hence, the viewpoint that the data architecture is hierarchical.

Command sentences are organized into phrases as follows: (1) verb, (2) object phrase, (3) adverbial phrase, (4), and (5) two prepositional phrases. The verb is always the first word reading from left to right whose vocabulary information permits interpretation as a verb. The adverbial and prepositional phrases are located by their being preceded with an adverb or preposition. The object phrase is then found by a process of elimination. A phrase consists of an identifier, either an adverb or preposition, a noun object, and an adjective modifier; except for the object phrase itself, which can consist of two noun-adjective pairs joined by conjunction. This structure was arrived at by writing down as many potential commands as could be invented, and then devising the minimum syntactical range capable of containing them. While hardly infinite, or

even literate by human standards, it has proved quite flexible for man to machine communication within the scope of the unfolding problem.

Interpretation is initiated by a carriage return. The words of the command sentence are first examined and categorized by parts of speech. Numbers are translated into their appropriate floating point or integer form. The sentence is then parsed according to the grammatical rules described in the preceding paragraph. If the object phrase is the word "it" or "them", it is replaced by the object phrase of the previous command. Synonyms are replaced by "root words", the first word of the synonym group to have been entered into the vocabulary. Upon completion of these steps, there exists a fourteen-word vector (some of whose locations may be blank), which is passed through a decision tree to initiate action.

The structure of the software is such that action is usually initiated by a single subroutine call. This subroutine may be at an executive level, in turn calling lower level routines. For unfolding, the "bottom level", the level at which more than 99% of the CPU time is spent, is the level of vector algebra. Consequently, the subroutines in the VECTOR module are short--usually two or three FORTRAN lines--making them readily implementable in assembly language, microcode, or by any means that permits taking advantage of fast arithmetic circuitry.

This volume will give a brief set of operations instructions, samples of the output obtained from hard copies of the display on a Tektronix terminal and finally a listing of the code.

2. OPERATING INSTRUCTIONS

The following is a set of operating instructions. It gives information on how to run the code, how to input and analyze data, and how to display the results.

IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTTTTTTTTTTTTT
IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTTTTTTTTTTTTT
IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTTTTTTTTTTTTT
III	NNN	NNN	SSS	TTT
III	NNN	NNN	SSS	TTT
III	NNN	NNN	SSS	TTT
III	NNNNNN	NNN	SSS	TTT
III	NNNNNN	NNN	SSS	TTT
III	NNNNNN	NNN	SSS	TTT
III	NNN	NNN	SSSSSSSSSS	TTT
III	NNN	NNN	SSSSSSSSSS	TTT
III	NNN	NNN	SSSSSSSSSS	TTT
III	NNN	NNNNNN	SSS	TTT
III	NNN	NNNNNN	SSS	TTT
III	NNN	NNNNNN	SSS	TTT
III	NNN	NNN	SSS	TTT
III	NNN	NNN	SSS	TTT
III	NNN	NNN	SSS	TTT
IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTT
IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTT
IIIIIIIII	NNN	NNN	SSSSSSSSSSSS	TTT

DDDDDDDDDDDD		AAAAA		TTTTTTTTTTTTTT
DDDDDDDDDDDD		AAAAA		TTTTTTTTTTTTTT
DDDDDDDDDDDD		AAAAA		TTTTTTTTTTTTTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAAAA	AAA	TTT
DDD	DDD	AAAAA	AAA	TTT
DDD	DDD	AAAAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDD	DDD	AAA	AAA	TTT
DDDDDDDDDDDD		AAA	AAA	TTT
DDDDDDDDDDDD		AAA	AAA	TTT
DDDDDDDDDDDD		AAA	AAA	TTT

LPTSPL Version 6(344) Running on LPT010
 START User M0SC-UNIVAC [3270,3270] Job INSTR Seq. 5391 Date 15-Nov-78 21:18:13
 Request created: 15-Nov-78 13:08:17
 File: SALLIO:INSTR.DAT[3270,3270] Created: 15-Nov-78 15:36:00 <255> Printed: 15-Nov-78
 QUEUE Switches: /PRINT:ARROW /FILE:F0RT /COPIES:1 /SPACING:1 /LIMIT:40 /FORMS:8X1

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

INSTRUCTIONS FOR RUNNING DELPHI

STARTING THE PROGRAM:

DELPHI CAN BE STARTED IN A NUMBER OF WAYS DEPENDING ON WHETHER MODIFICATIONS HAVE BEEN MADE TO ANY OF THE SUBROUTINES. IN MOST CASES ONE WILL HAVE AN EXECUTABLE VERSION OF THE CODE IN THE USERS AREA AND THE CODE CAN BE STARTED BY SAYING

RUN DELPHI

TO RUN A NEW OR UPDATED VERSION OF THE CODE:
TO COMPILE:

COMPILE @D
D IS A ONE LINE PROGRAM CONTAINING THE NAMES OF THE FILES CALLED BY DELPHI. THE FILES ARE
DELPHI
INPUT
OUTPUT
VECTOR
MATRIX
UNFOLD
ANALY
PLOT10 (GRAPHICS PACKAGE)

TO LOAD: LOAD @D

TO SAVE: SAVE

THIS SEQUENCE OF COMMANDS PRODUCES
A FILE DELPHI.EXE WHICH CAN BE RUN BY SAYING
RUN DELPHI.

ONE CAN ALSO RUN OR EXECUTE DELPHI BY SAYING
EX @D

THIS IS SOMETIMES HELPFUL IF ONE IS WORKING ON
MODIFICATIONS TO THE CODE.

NOTE: IF ONE USES THE COMMANDS

ANALYZE
CHANGE
SIMULATE A SURVEY

ONE MUST ASSIGN DISK UNITS 25 TO 29 IN ORDER TO AVOID ERROR MESSAGES. IF ERROR MESSAGES DO OCCUR BECAUSE THE UNITS ARE NOT ASSIGNED ONE CAN SIMPLY ENTER THE PROPER DISK UNIT AND THE FILE NAME AND CONTINUE.

DELPHI WILL RESPOND WHEN READY WITH THE RESPONSE
<<DELPHI IS READY

ONE CAN NOW INPUT THE VARIOUS COMMANDS IN DELPHI'S
ENGLISH LANGUAGE COMMAND STRUCTURE. THE CODE WILL
ONLY ACCEPT ONE COMMAND AT A TIME. FOR EXAMPLE
IF ONE WANTS TO READ SOME DATA, UNFOLD IT AND COMPUTE CHI-SQUARED,
ONE MUST USE THREE COMMANDS:

READ *FILE.EXT FROM RECORD N1 TO RECORD N2
UNFOLD THE DATA
COMPUTE CHI-SQUARED

ONE CANNOT COMMAND

UNFOLD THE DATA AND COMPUTE CHI-SQUARED
ONLY THE FIRST COMMAND WILL BE RECOGNIZED.
THE CODE USES ONLY THE FIRST FIVE LETTERS OF EACH WORD.
COMMANDS LIKE

SHOW THE URANIUM FIT
CAN BE WRITTEN AS
SHOW THE URANI FIT

COMMANDS: ONE CAN READ DATA, CHANGE PARAMETERS, UNFOLD DATA AND SHOW MOST
OF THE VARIOUS QUANTITIES WHICH ARE USED IN DELPHI.

THIS SET OF INSTRUCTIONS WILL DESCRIBE THE MORE IMPORTANT OF THESE
COMMANDS.

READ:

READ IS USED TO INPUT DATA INTO THE CODE.

EXAMPLE:

READ "LKMSAD.DAT" FROM RECORD 5 TO RECORD 25
CAUSES RECORD 5 THROUGH 24 TO BE READ FROM THE
THE DATA FILE LKMSAD.DAT AND SUMMED AS DATA

INPUT:

INPUT IS USED TO ENTER EITHER RESPONSE
FUNCTIONS OR NEW VOCABULARY WORDS INTO THE CODE.

EXAMPLE:

INPUT K(DISCRETE)
THE PROGRAM WILL RESPOND WITH
<<FILE.EXP?
ONE NOW INPUTS THE NAME OF THE FILE CONTAINING
THE DISCRETE RESPONSE MATRIX.
AT PRESENT THE RESPONSE MATRIX IS ON FILE
URF.DAT

NOTE: ABOUT 100 RECORDS OF DATA SHOULD BE IN
THE DATA FILE WHEN THE RESPONSE FUNCTION IS
INPUT.

INPUT "NEWWORD" INTO VOCABULARY
THIS COMMAND WILL ENTER THE WORD NEWWORD INTO THE
VOCABULARY. THE CODE WILL RESPOND WITH A
QUESTION REGARDING THE PART OF SPEECH.

DELETE:

ONE CAN DELETE A WORD FROM THE VOCABULARY BY SAYING
DELETE "OLDWORD" FROM VOCABULARY

MAKE:

ONE CAN MAKE WORDS SYNONYMOUS BY SAYING
MAKE "WORD1" SYNONYMOUS WITH "WORD2" ,

UNFOLD:

THE COMMAND
UNFOLD THE DATA
OR SIMPLY
UNFOLD
WILL CAUSE THE PROGRAM TO UNFOLD THE DATA USING MAZAS.

COMPUTE:

THE COMMAND:

COMPUTE CHI-SQUARED

CAUSES THE CODE TO CALCULATE CHI-SQUARED AND TYPE THE RESULT.

SET:

ALL OF THE PARAMETERS IN DELPHI CAN BE CHANGED USING THE SET COMMAND. FOR EXAMPLE, ONE CAN CHANGE THE PARAMETER TAU0 BY SAYING

SET TAU0 EQUAL TO 100.

IT SHOULD BE NOTED THAT SETTING A PARAMETER DOES NOT CHANGE THE OPERATION OF MAZAS UNTIL THE DISCRETE RESPONSE MATRIX IS INPUT.

ONE CAN SET SWITCHES TO OBTAIN EITHER MAZAS OR WINDOW ANALYSIS OF THE DATA IN THE SURVEY PORTION OF THE CODE BY THE COMMANDS:

SET MAZAS

OR

SET WINDOW

ALSO ONE CAN SET SWITCHES TO GIVE EITHER A RUNNING SUM OR DELPHI TIME FILTER ANALYSIS OF THE DATA IN THE SURVEY PORTION OF THE CODE BY SAYING:

SET SUM TO N1

OR

SET FILTER

IN THE COMMAND SETTING THE SUM, N1 IS THE NUMBER OF SECONDS TO BE USED IN THE RUNNING SUM ANALYSIS. N1 MUST BE AN ODD INTEGER. THE DEFAULT VALUE OF N1 IS SEVEN.

IN ADDITION ONE CAN OBTAIN GRAPHICAL OUTPUT USING DISPLAY BY THE COMMAND:

SET COPY

THE CODE WILL ASK FOR A FILE NAME AT THIS POINT. THE GRAPHICAL DATA WILL BE WRITTEN INTO THIS FILE FOR POST PROCESSING USING A CODE SUCH AS VTCPOP. CHECK THE USERS MANUAL FOR THE NAME OF THE POST PROCESSING CODE IF THIS OPTION IS TO BE USED.

SHOW:

THE COMMAND SHOW OR DISPLAY WILL CAUSE THE TEKTRONIX TERMINAL TO EITHER LIST THE PARAMETERS REQUESTED OR DRAW A GRAPH OF THE DESIRED QUANTITIES. THE MOST COMMON OF THE QUANTITIES WHICH CAN BE SHOWN ARE LISTED BELOW.

SHOW THE VOCABULARY

NOONS

VERBS

ADVERBS

ADJECTIVES

PREPOSITIONS

WILL LIST THE ENTIRE VOCABULARY OR JUST THE DESIRED PART OF THE VOCABULARY.

SHOW THE DATA

WILL PRODUCE A LINEAR GRAPH OF THE DATA.

SHOW THE DATA AND THE EXPECTED DATA

WILL PRODUCE A GRAPH OF THE DATA AND THE
FIT OBTAINED USING THE UNFOLDING
ROUTINE MAZAS. THIS GRAPH WILL BE ON A SQUAR-
ROOT SCALE.

SHOW THE CONTINUUM FIT

DISCRETE
URANIUM
THORIUM
POTASIUUM

WILL PRODUCE GRAPHS OF THE DATA AND THE REQUESTED
PORTION OF THE UNFOLDED SPECTRUM.

SHOW THE CONTINUUM RESIDUALS

DISCRETE
URANIUM
THORIUM
POTASIUUM

WILL PRODUCE A LINEAR GRAPH OF THE DATA AND
THE DIFFERENCE BETWEEN THE DATA AND THE
INDIVIDUAL SPECIFIED COMPONENT.
IF NO INDIVIDUAL COMPONENT IS SPECIFIED THE
DISPLAY WILL BE OF THE RESIDUALS BETWEEN THE
DATA AND THE FIT PRODUCED BY MAZAS.

SHOW K(I)

WILL PRODUCE A LINEAR GRAPH OF THE ITH COMPONENT
OF THE RESPONSE MATRIX. THE FIRST FOUR COMPONENTS
ARE THE CONTINUUM COMPONENTS, THE NEXT IS THE
POTASIUUM RESPONSE, THE NEXT 9 ARE THE RESPONSES
FOR URANIUM AND THE NEXT 8 ARE THE RESPONSES
FOR THORIUM. THE LAST THREE RESPONSES ARE FOR
CS-137, ANNALILATION RADIATION AND U-238.

SHOW O(I)

WILL PRODUCE A LINEAR GRAPH OF THE CONTENTS
OF THE ITH O BUFFER. THESE
ARE THE UNFOLDING MARTIX COMPONENTS.

SHOW THE UNFOLDING PARAMETERS

SIMULATION PARAMETERS
TRANSFER

LISTS THE CURRENT VALUES OF THE PARAMETERS.
IF THE TYPE OF PARAMETER IS NOT SPECIFIED,
ALL OF THE PARAMETERS WILL BE SHOWN.

SHOW PHI

GIVES BOTH THE VALUES OF THE UNFOLDED LINE
INTENSITIES AND A GRAPHICAL PRESENTATION OF THEIR
VALUES.

SHOW THE ELAPSED TIME

TYPES THE ELAPSED CPU TIME SINCE THE LAST REQUEST
TO SHOW THE TIME.

SIMULATE:

SIMULATE DATA WITH N COUNTS

WILL PRODUCE A DATA SET (BASED ON THE LAST SET
OF PARAMETERS WHICH WERE SAVED FOR DATA SIMULATION)

WITH N TOTAL COUNTS.
IF THE NUMBER OF COUNTS IS NOT SPECIFIED, THE
CODE WILL SIMULATE DATA WITH THE NUMBER OF COUNTS
IN THE LAST UNFOLDED DATA SET.

SIMULATE A SURVEY

THIS COMMAND WILL PRODUCE A SET OF 101 SECONDS OF
SIMULATED DATA. THE CODE WILL ASK FOR THE QUANTITY TO BE
VARIED. ONE CAN CHANGE THE CONCENTRATION OF URANIUM,
CONTINUUM OR THE GAIN WITH THE RESPONSE FUNCTION URF.DAT
CHANGES IN THE OTHER CONCENTRATIONS WILL REQUIRE
DIFFERENT RESPONSES. THE MAGNITUDE OF THE GAIN CHANGE
IS CONTROLLED BY THE PARAMETER WOBBLE.
A VALUE OF WOBBLE = 1 GIVE A ONE CHANNEL CHANGE IN
THE VALUE OF ZERO. A VALUE OF 2 GIVES A 2 CHANNEL
CHANGE.

SAVE:

SAVE THE CONTINUUM PARAMETERS FOR DATA SIMULATION
THIS COMMAND SAVE THE CONTINUUM PARAMETERS
OBTAINED IN THE LAST UNFOLDING FOR SIMULATION
OF DATA.

SAVE THE DISCRETE PARAMETERS FOR DATA SIMULATION
SAVES THE LAST SET OF UNFOLDED
DISCRETE PARAMETERS.

SAVE THE PARAMETERS FOR DATA SIMULATION
THIS COMMAND SAVES ALL OF THE PARAMETERS FOR
DATA SIMULATION.

SURVEY:

SURVEY IS A GENERAL COMMAND FOR ANALYZING A
SEQUENCE OF DATA. THIS COMMAND CAN BE USED IN CONNECTION WITH
THE COMMANDS SET MAZAS, SET WINDOW, SET SUM AND SET FILTER
TO PRODUCE TIME SEQUENCES OF ANALYZED DATA.
IF ONE WISHES TO OBTAIN A FIVE SECOND RUNNING SUM ANALYSIS
OF THE FILE LKMEAD.DAT FROM RECORD 20 TO RECORD 130 ONE
USE THE FOLLOWING COMMANDS:

SET SUM TO 5
SET WINDOW
SURVEY *LKMEAD.DAT FROM RECORD 20 TO RECORD 130

THE CODE WILL RESPOND WITH THE AVERAGE VALUES
AND THE RMS DEVIATIONS OF THE POTASSIUM, URANIUM AND
THORIUM ANALYSES. ONE CAN NOW DISPLAY THE VARIOUS
TIME AVERAGED QUANTITIES BY COMMANDING:

SHOW THE POTASSIUM SURVEY
URANIUM SURVEY
THORIUM SURVEY

ONE CAN OBTAIN OTHER COMBINATIONS OF WINDOW,
MAZAS, WINDOW AND FILTER RESULTS BY SETTING THE PROPER SWITCHES.

ANALYZE:

ANALYZE *FILE.EXT FROM RECORD N1 TO RECORD N2

THIS OPTION WILL PRODUCE A RUNNING SUM ANALYSIS
OF THE DATA FILE FILE.EXT FROM RECORD N1 TO
RECORD N2. THE NUMBER OF SECONDS TO

BE AVERAGED IS ASKED FOR BY THE PROGRAM
AND IS VARIABLE FROM 2 SECONDS TO 99
SECONDS.

NOTE: AT PRESENT THE CODE WILL NOT DO A 1 SEC
RUNNING SUM! SINGLE SECOND ANALYSES CAN BE OBTAINED
USING THE SURVEY COMMAND.

THE PROGRAM WILL RESPOND WITH
<< ENTER NUMBER OF SECONDS TO BE AVERAGED
THIS PARAMETER IS TO BE INPUT IN I2 FORMAT
NOTE:

05 WILL PRODUCE A 5 SECOND RUNNING SUM
AVERAGE. 5 WILL PRODUCE A 50 SECOND RUN-
NING SUM AVERAGE!

THE CODE THEN ASKS FOR AN OUTPUT FILE NAME.
ALL OF THE DATA WILL BE WRITTEN ON THIS FILE.
THE RESULT OF THESE COMMANDS IS A DISPLAY OVER
HALF OF THE SCREEN OF THE RESULT OF THE RUNNING
SUM ANALYSIS OF THE 1.764 MEV WINDOW ANALYSIS
(JOINED BY LINES) AND THE MAZAS RESULTS FOR
THE URANIUM MAZAS ANALYSIS(POINTS), THE WINDOW
ANALYSIS OF THE THORIUM 2.614 MEV LINE(XS)
AND THE MAZAS THORIUM 2.614 MEV LINE(CROSSES).
THE DISPLAY IS LOG TO ALLOW COMPARISON OF THE
RESULTS WHICH ARE NOT NORMALIZED TO EACH OTHER.
THE LEFT SIDE OF THE DISPLAY
GIVES THE PEAK NUMBERS, THE PEAK ENERGIES,
THE AVERAGE VALUES OF THE MAZAS RESULTS AND THE
RMS DEVIATIONS FOR THE MAZAS RESULT FROM THESE
AVERAGES. THE LAST 4 LINES OF THE TYPE OUT GIVE
THE AVERAGE VALUES OF THE POTASSIUM, URANIUM,
THORIUM WINDOW ANALYSES, THEIR RMS DEVIATIONS
FROM THE AVERAGE AND THE AVERAGE VALUE OF
CHI-SQUARED CALCULATED USING MAZAS, RESPECTIVELY.
AT PRESENT, THE CODE WILL PRODUCE A SECOND GRAPH AFTER
A CARRAGE RETURN HAS BEEN GIVEN. THE SECOND GRAPH
WILL GIVE A LOG DISPLAY OF THE LAST SURVEY MADE.

ADJUST:

ADJUST THE GAIN

THIS ROUTINE ADJUSTS THE GAIN OF THE DATA IN
CORE UNTIL CHI-SQUARED IS MINIMIZED. THE VALUE
OF THE SLOPE AND INTERCEPT OF THE ADJUSTED GAIN
ARE TYPED OUT.

CHANGE:

CHANGE K

THIS PORTION OF THE CODE WILL REBIN THE DISCRETE
RESPONSE FUNCTIONS INTO THE GAIN OF THE DATA.
THE PROCESS FOR USING THIS PORTION OF THE CODE
IS TO FIRST ADJUST THE GAIN. THE CODE WILL RESPOND WITH
THE NEW VALUES OF THE SLOPE AND ZERO AND THE NEW VALUE
OF CHI-SQUARED. IF THE RESPONSE FUNCTIONS ARE TO BE
REBINED THE NEXT COMMAND IS

CHANGE K

THE CODE WILL RESPOND AND ASK FOR THE FILE.EXT OF THE OLD RESPONSE FUNCTIONS. (IF THE RESPONSE FUNCTIONS HAVE BEEN REBINED THE NAME OF THE LAST SET SHOULD BE USED.) AFTER THE RESPONSE FUNCTIONS ARE REBINED THE CODE WILL ASK FOR THE FILE.EXT OF THE NEW RESPONSES. THE REBINNED RESPONSES ARE WRITTEN INTO THIS NEW FILE.

IT SHOULD BE POINTED OUT THAT ONE SHOULD NOW INPUT THE NEW RESPONSES SINCE THE CODE HAS NOT BEEN SET UP FOR THE ADJUSTED GAIN. THE SEQUENCE OF COMMANDS TO USE IN SETTING UP THE CODE FOR A NEW SET OF DATA ARE:

READ "DATA.DAT FROM RECORD N TO RECORD N+100

<<COMPLETE READ DATA

ADJUST THE GAIN

<<CHI-SQUARED = , XERO = , SLOPE =

CHANGE K

<<FILE.EXT?

ENTER OLD RESPONSE FUNCTION FILE NAME

<<OUTPUT FILE NAME

ENTER NEW RESPONSE FUNCTION FILE NAME

<<COMPLETE CHANGE K

READ "DATA.DAT FROM RECORD N TO RECORD N+100

<<COMPLETE READ DATA

INPUT K(DISCRETE)

<<FILE.EXT?

ENTER NAME OF THE NEW RESPONSE FUNCTION FILE

<<COMPLETE INPUT K

EXIT:

IF ONE HAS SET COPY ONE MUST USE THE EXIT COMMAND TO TERMINATE THE PROGRAM.

STOP:

THE COMMAND STOP WILL BOTH STOP THE PROGRAM AND SAVE ALL OF THE PARAMETERS AND DATA IN THE CODE AT THE TIME OF THE COMMAND.

IF THE PARAMETERS HAVE BEEN CHANGED AND ONE DOES NOT WISH TO SAVE THEN THE PROGRAM SHOULD BE STOPPED BY USING EITHER THE EXIT COMMAND IF THE COPY SWITCH HAS BEEN SET OR TERMINATE EXECUTION BY THE USE OF TWO CONTROL C'S.

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

3. DELPHI GRAPHICAL OUTPUT

This section gives several examples of the displays available using the DELPHI commands. These graphs were generated on the Tectronix terminal using the following commands:

<u>Figure</u>	<u>Command</u>
3.	SHOW THE UNFOLDING PARAMETERS
4.	SHOW THE SIMULATION PARAMETERS
5.	SHOW THE DATA AND THE EXPECTED DATA
6.	SHOW THE CONTINUUM FIT
7.	SHOW THE POTASSIUM FIT
8.	SHOW THE URANIUM FIT
9.	SHOW THE THORIUM FIT
10.	ANALYZE 'SIMDAT.DAT FROM RECORD 1 TO RECORD 61 05 (second running sum)
11.	SECOND GRAPH GENERATED BY ANALYZE COMMAND
12.	ANALYZE 'LKMEAD.DAT FROM RECORD 1 TO RECORD 131 05 (second running sum)
13.	SECOND GRAPH OF ANALYZE COMMAND
14.	SHOW PHI

DIMENSION: N{.....}					
DATA 233	SPECTRUM 25	LINES 21	POTASSIUM 1	URANIUM 9	THORIUM 3

SLOPE 1.1635E-02 MEVS/CHANNEL	TAUC 0.00E+00 MEV	FWHM=RESOLUTION*E*(.6616/E)**POWER =(9.55E-02)*E*(.6616/E)**(3.42E-01) (MEV)			
ZERO 2.1609E-01 MEV	TAUD 1.25E+02	COUNTS 3.77E+05	ALTITUDE 3.50E+02 FEET	ETIME 7.54E+01 SECONDS	

I	ELINE(I)	I	ELINE(I)	I	ELINE(I)	I	ELINE(I)
POTASSIUM:		5	9.34E-01	THORIUM:		16	1.25E+00
1	1.46E+00	6	1.12E+00	11	3.39E-01	17	1.63E+00
URANIUM:		7	1.38E+00	12	4.63E-01	18	2.61E+00
2	3.51E-01	8	1.76E+00	13	5.83E-01	19	5.11E-01
3	6.09E-01	9	2.20E+00	14	7.27E-01	20	6.61E-01
4	7.68E-01	10	2.45E+00	15	9.11E-01	21	1.00E+00
I	KUT(I)	I	KUT(I)	I	KUT(I)	I	KUT(I)
POTASSIUM:		5	3.50E-01	THORIUM:		16	2.80E-01
1	2.55E+00	6	3.00E-01	11	5.20E-01	17	2.50E-01
URANIUM:		7	2.80E-01	12	4.40E-01	18	2.00E-01
2	5.00E-01	8	2.40E-01	13	4.00E-01	19	4.20E+01
3	3.90E-01	9	2.15E-01	14	3.60E-01	20	3.80E+01
4	3.60E-01	10	2.10E-01	15	3.30E-01	21	3.20E+01

Figure 3

SHOW THE SIMUL PARAMS

<<DIAMETER = 11.50 INCHES, THICKNESS = 4.00 INCHES
[GAIN] WOBBLE = 1.00 PER CENT
CTOD [CONTINUUM/DISCRETE COUNT RATIO] = 3.88
RESOLUTION COMPONENTS, NGAUSS = 2
ACONTINUUM (1) = 3.014E + 05, ACONTINUUM (2) = -1.672E + 04
ACONTINUUM (3) = 2.048E + 04, ACONTINUUM (4) = 5.280E + 04
ELINE (1) = 1.461, ALINE (1) = 1.32E + 03
ELINE (2) = 0.351, ALINE (2) = 3.73E + 02
ELINE (3) = 0.609, ALINE (3) = 6.20E + 02
ELINE (4) = 0.768, ALINE (4) = 7.19E + 02
ELINE (5) = 0.934, ALINE (5) = 7.94E + 02
ELINE (6) = 1.120, ALINE (6) = 8.54E + 02
ELINE (7) = 1.377, ALINE (7) = 9.15E + 02
ELINE (8) = 1.764, ALINE (8) = 9.85E + 02
ELINE (9) = 2.204, ALINE (9) = 1.05E + 03
ELINE (10) = 2.447, ALINE (10) = 1.08E + 03
ELINE (11) = 0.339, ALINE (11) = 2.19E + 01
ELINE (12) = 0.463, ALINE (12) = 2.70E + 02
ELINE (13) = 0.583, ALINE (13) = 4.51E + 02
ELINE (14) = 0.727, ALINE (14) = 6.17E + 02
ELINE (15) = 0.911, ALINE (15) = 7.82E + 02
ELINE (16) = 1.246, ALINE (16) = 1.01E + 03
ELINE (17) = 1.631, ALINE (17) = 1.20E + 03
ELINE (18) = 2.614, ALINE (18) = 1.53E + 03
ELINE (19) = 0.511, ALINE (19) = 2.83E + 02
ELINE (20) = 0.661, ALINE (20) = 1.90E + 02
ELINE (21) = 1.001, ALINE (21) = 1.71E + 02
<<COMPLETED DISPL PARAM

Figure 4

Figure 5

Figure 6

Figure 7

Figure 8

Figure 9

1	1.461	0.135 E + 02	0.479 E + 01
2	0.351	0.466 E + 01	0.421 E + 02
3	0.609	0.601 E + 01	0.208 E + 02
4	0.768	0.742 E + 01	0.155 E + 02
5	0.934	0.808 E + 01	0.121 E + 02
6	1.120	0.869 E + 01	0.998 E + 01
7	1.377	0.933 E + 01	0.880 E + 01
8	1.764	0.102 E + 02	0.906 E + 01
9	2.204	0.110 E + 02	0.102 E + 02
10	2.447	0.113 E + 02	0.109 E + 02
11	0.339	0.118 E + 01	0.130 E + 03
12	0.463	0.275 E + 01	0.719 E + 02
13	0.583	0.442 E + 01	0.431 E + 02
14	0.727	0.617 E + 01	0.260 E + 02
15	0.911	0.796 E + 01	0.160 E + 02
16	1.246	0.104 E + 02	0.821 E + 01
17	1.631	0.125 E + 02	0.475 E + 01
18	2.614	0.162 E + 02	0.472 E + 01
K		0.202 E + 03	0.370 E + 01
U		0.242 E + 02	0.135 E + 02
TH		0.691 E + 02	0.398 E + 01
CHI		0.934 E + 00	0.101 E + 02

Figure 10

TIME FILTERED ANALYSIS OF SIMDAT.DAT
KBAR = 13.406 ERR 0.084
UBAR = 9.103 ERR 0.261
TBAR = 16.184 ERR 0.102
<<COMPLETED ANALY SIMDA

Figure 11

1	1.461	0.135 E + 02	0.557 E + 02
2	0.351	0.451 E + 01	0.597 E + 02
3	0.609	0.730 E + 01	0.361 E + 02
4	0.768	0.842 E + 01	0.329 E + 02
5	0.934	0.926 E + 01	0.316 E + 02
6	1.120	0.994 E + 01	0.310 E + 02
7	1.377	0.106 E + 02	0.308 E + 02
8	1.764	0.114 E + 02	0.311 E + 02
9	2.204	0.121 E + 02	0.317 E + 02
10	2.447	0.124 E + 02	0.321 E + 02
11	0.339	0.229 E + 01	0.115 E + 03
12	0.463	0.371 E + 01	0.856 E + 02
13	0.583	0.525 E + 01	0.644 E + 02
14	0.727	0.690 E + 01	0.517 E + 02
15	0.911	0.856 E + 01	0.472 E + 02
16	1.246	0.108 E + 02	0.464 E + 02
17	1.631	0.127 E + 02	0.475 E + 02
18	2.614	0.159 E + 02	0.502 E + 02
K	0.212	0.212 E + 03	0.565 E + 02
U	0.267	0.267 E + 02	0.435 E + 02
Th	0.707	0.707 E + 02	0.518 E + 02
CHI	0.108	0.108 E + 01	0.120 E + 02

Figure 12

TIME FILTERED ANALYSIS OF LKMEAD.DAT
 KBAR = 14.447 ERR 7.752
 UBAR = 10.997 ERR 2.935
 TBAR = 16.114 ERR 7.657
 <<COMPLETED ANALY LKMEA

Figure 13

ENERGY	INTENSITY	ENERGY	INTENSITY
1.461	1.315E+03	0.583	4.505E+02
0.351	3.728E+02	0.727	6.171E+02
0.609	6.204E+02	0.911	7.818E+02
0.768	7.192E+02	1.246	1.006E+03
0.934	7.935E+02	1.631	1.196E+03
1.120	8.539E+02	2.614	1.528E+03
1.377	9.151E+02	0.511	2.828E+02
1.764	9.850E+02	0.661	1.903E+02
2.204	1.047E+03	1.001	1.713E+02
2.447	1.076E+03		
0.339	2.186E+01		
0.463	2.702E+02		

Figure 14

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

4. SPECIAL INPUT CHARACTERISTICS OF RESPONSE FUNCTIONS AND DATA FILES

4.1 RESPONSE FUNCTIONS

The response functions are read from a file written in 'image' mode with the following structure:

- Discrete potassium response { NPOTAS = 1
Continuum potassium response { each 256 channels
- Discrete uranium response { NURANI responses
each 256 channels
(NURANI at present=9)
- Continuum uranium response 256 channels
- Discrete thorium response { NTHORI responses
each 256 channels
(NTHORI at present=8)
- Continuum thorium response 256 channels
- Other discrete responses { 256 channels each
(3 other lines used
at present)

The gain of these responses should be close to that of the data to be analyzed. The first LOCHAN channels of the responses and the data will be truncated to eliminate problems with the very low energy spectrum.

4.2 DATA FILES

DELPHI reads compressed data files with each word actually containing two pieces of data. Also, the first word read contains the height and cosmic background information. The last word contains the detector live time.

Compressed data is generated by multiplying an octical 777777 (decimal 262144) times the first word and adding the second word. This shifts the first word by 18 binary bits to the left and adds the second word. The following gives the structure of the compressed file IDAT in terms of the data, IDATA; the time, IT; the cosmic counts above 3 MeV, ICOS; and the elevation, IH.

```
IDAT (1) = 262144 * IH + ICOS
IDAT (2) = 262144 * IDATA (3) + IDATA (4)
IDAT (3) = 262144 * IDATA (5) + IDATA (6)
      .
      .
      .
IDAT (127) = 262144 * IDATA (253) + IDATA (254)
IDAT (128) = IT
```

At present, IH corresponds to the Geodata, Inc. radar altimeter recording in millivolts and must be converted in DELPHI into an elevation in feet.

5. LISTING OF CODE

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

1[illegible]

37

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

PROGRAM DELPHI

- * SPEAK AND DELPHI WILL ANSWER.
- * DELPHI WAS CREATED BY DR. MARTIN SPERLING
- * FOR SCIENCE APPLICATIONS, INC., IN 1977.

```

      DIMENSION IDAT(128)
      COMMON PRED,OBJ1,OBMOD1,OBJ2,OBMOD2,ADPRED,ADOB,
;      ADMOD,PREP1,PROB1,PRMOD1,PREP2,PROB2,PRMOD2
      DIMENSION SYNTAX(14)
      EQUIVALENCE (PRED,SYNTAX(1)),(FMOD,PRMOD1)
      INTEGER SYNTAX,PRED,OBJ1,OBMOD1,OBJ2,OBMOD2,ADPRED,
;      ADOB,ADMOD,PREP1,PROB1,PRMOD1,PREP2,PROB2,PRMOD2
      INTEGER CHAR(72),VOCAB(80,6),VEND(40),VERB(80),NEND(80),
;      NOUN(160),ADVEND(40),ADVERB(80),ADJEND(40),
;      ADJCTV(80),PRPEND(40),PREPOS(80)
      INTEGER WORD(14),POINT,NVRT,NNRT,NADVRT,NADJRT,NPPRRT
      DIMENSION KPART(14)
      EQUIVALENCE (VOCAB(1,1),VERB(1)),(VOCAB(1,2),NOUN(1)),
;      (VOCAB(1,4),ADVERB(1)),(VOCAB(1,5),ADJCTV(1)),
;      (VOCAB(1,6),PREPOS(1))
      COMMON/SCRATC/TEMP(1022),QUOTE,DISCRE,DUMMY(256)
      COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
      COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
      COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
      COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
      INTEGER INSW,UNFSW,SIMSW,OUTSW,RECORD
      DOUBLE PRECISION DWORD,DATFIL
      EQUIVALENCE (DWORD,OBJ1)
      COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
      COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),
;      DEXP(256),G(256),O(256,32),K(256,32)
      DIMENSION PTASI(256),URANI(256),THORI(256),KGK(676)
      EQUIVALENCE (PTASI,O(1,27)),(URANI,U(1,28)),(THORI,O(1,29))
      EQUIVALENCE (KGK,O(1,30))
      REAL K,KUT

      OPEN(UNIT=1,DEVICE='DSKC',FILE='VOCAB.DAT',ACCESS='RANDOM',
;      MODE='IMAGE',BUFFER COUNT=1,RECORD SIZE=128)
      READ(1#1) NVRT,VEND,VERB
      READ(1#2) NNRT,(NEND(I),I=1,40),(NOUN(I),I=1,80)
      READ(1#3) NADVRT,ADVEND,ADVERB
      READ(1#4) NADJRT,ADJEND,ADJCTV
      READ(1#5) NPPRRT,PRPEND,PREPOS
      READ(1#6) NNRT,(NEND(I),I=41,80),(NOUN(I),I=81,160)
      CLOSE(UNIT=1)

      CALL ERRSET(0)
      CALL READAT
      CALL SETSW(UNFSW,2,1)
      CALL SFTSW(INFSW,6,1)
      CALL SETSW(OUTSW,1,0)
      CALL INITT(400)
      CALL TERM(1,1024)
8001 TYPE 9000
9000 FORMAT(' <<DELPHI IS READY-G')
      GO TO 1

```

2 TYPE 9001,PRED,OBJ1
9001 FORMAT(' <<COMPLETED ',A5,' ',A5,'-G')
1 CALL SECOND(T2)
DTIME=T2-T1
ACCEPT 9002,CHAR
9002 FORMAT(72R1)

```

POINT=1
DO 3 II=1,14
I=II
3 CALL CATGRY(POINT,CHAR,WORD(I),KPART(I),VOCAB)
CALL REFER(KPART,WORD,SYNTAX,VOCAB,$8000)
CALL PARSE(KPART,WORD,$1)
DO 7 II=1,14
I=II
IF(SYNTAX(I).EQ.' ') GO TO 7
IF(I.EQ.1) CALL REDUCE(SYNTAX,1,NVRT,VEND,VERB,40,80)
IF(I.EQ.6) CALL REDUCE(SYNTAX,6,NADVRT,ADVEND,ADVERB,40,80)
IF(I.EQ.9.OR.I.EQ.12) CALL REDUCE
(SYNTAX,I,NPRPRT,PRPEND,PREPOS,40,80)
IF(I.EQ.2.OR.I.EQ.4.OR.I.EQ.7.OR.I.EQ.10.OR.I.EQ.13) GO TO 4
GO TO 7
4 IF(SYNTAX(I).GT.0) GO TO 5
CALL REDUCE(SYNTAX,I,NVRT,VEND,NOUN,80,160)
CALL REDUCE(SYNTAX,I+1,NADJRT,ADJEND,ADJECTV,40,80)
GO TO 7
5 SYNTAX(I)=-SYNTAX(I)
7 CONTINUE

IF(PRED.NE.'STOP') GO TO 10
CALL WRIDAT
GO TO 11
10 IF(PRED.NE.'EXIT') GO TO 12
11 CALL ERASE
CALL GETSW(OUTSW,1,I)
IF(I.EQ.1) CALL DONEPL
STOP

12 IF(PRED.NE.'CLEAR') GO TO 1000
CALL NEWPAG
GO TO 8001

1000 CALL SECOND(T1)
IF(PRED.NE.'INPUT') GO TO 1200
IF(POB1.NE.'VOCAB') GO TO 1010
1001 CALL EXIST(SYNTAX,2,VOCAB,KPART(2),$1)
CALL GETPOS(KPART(2),NREC,$1003,$1004,$1005,$1006,$1007,$1)
1003 CALL NEWORD(SYNTAX,2,NVRT,VEND,VERB,40,80,NREC,$1,$2)
1004 CALL NEWORD(SYNTAX,2,NVRT,VEND,NOUN,80,160,NREC,$1,$2)
1005 CALL NEWORD(SYNTAX,2,NADVRT,ADVEND,ADVERB,40,80,NREC,$1,$2)
1006 CALL NEWORD(SYNTAX,2,NADJRT,ADJEND,ADJECTV,40,80,NREC,$1,$2)
1007 CALL NEWORD(SYNTAX,2,NPRPRT,PRPEND,PREPOS,40,80,NREC,$1,$2)

1010 IF(OBJ1.NE.'K') GO TO 1100
IF(OBMOD1.NE.'CONTI') GO TO 1012
CALL MATIN(1,NCONT)
GO TO 2
1012 IF(OBMOD1.NE.'DISCR') GO TO 8000
CALL MATIN(33-NLINES,32)
CALL SETUP
QUOTE=.FALSE.
GO TO 2

1100 IF(OBJ1.EQ.' ') GO TO 8000
CALL NEWDAT(DWORD)
IF(PREP1.EQ.'TO') CALL XCHANG(9,12,3)
NREC=1

```



```

IF(PREP1,EQ,'FROM',AND,PREP2,EQ,'TO') NREC=PRMOD2-PRMOD1+1
IF(NREC,LT,1) GO TO 8000
IF(PREP1,EQ,'') GO TO 1125
IF(PRMOD1,LT,1) GO TO 8000
CALL MOVDISK(PRMOD1)
1125 CALL DATIN(NREC)
GO TO 2

1200 IF(PRED,NE,'SAVE') GO TO 1325
IF(PROB1,NE,'DATA') GO TO 1325
IF(PRMOD1,NE,'SIMUL') GO TO 1325
IF(OBMOD1,EQ,'') GO TO 1205
IF(OBMOD1,EQ,'CONTI') GO TO 1210
IF(OBMOD1,EQ,'DISCR') GO TO 1215
IF(OBMOD1,GE,1,AND,OBMOD1,LE,32) GO TO 1220
GO TO 8000
1205 CALL GETSW(UNPSW,3,I)
IF(I,EQ,0) CALL HAZASC
CALL MULS(PHI,PHISIM,1.0,NC)
GO TO 2
1210 CALL GETSW(UNFSW,3,I)
IF(I,EQ,0) CALL HAZASC
CALL MULS(PHI,PHISIM,1.0,NCONT)
CALL CONDIS
GO TO 2
1215 CALL MULS(PHI(NCONT+1),PHISIM(NCONT+1),1.0,NLINES)
GO TO 2
1220 PHISIM(OBMOD1)=PHI(OBMOD1)
GO TO 2

1325 IF(PRED,NE,'DELET') GO TO 1450
IF(PREP1,NE,'FROM') GO TO 8000
IF(PROB1,NE,'VOCAB') GO TO 8000
IF(OBJ1,EQ,'') CALL EXIST(SYNTAX,2,VOCAB,KPART(2),S1)
CALL POS(OBJ1,NREC,VOCAB)
IF(NREC,EQ,0) GO TO 8000
GO TO (1326,1327,1328,1329,1330) NREC
1326 CALL XWORD(SYNTAX(2),NVRT,VEND,VERB,40,80,NREC,S1,S2)
1327 CALL XWORD(SYNTAX(2),NNRT,NEND,NOUN,80,160,NREC,S1,S2)
1328 CALL XWORD(SYNTAX(2),NADVRT,ADVEND,ADVERB,40,80,NREC,S1,S2)
1329 CALL XWORD(SYNTAX(2),NADJRT,ADJEND,ADJCTV,40,80,NREC,S1,S2)
1330 CALL XWORD(SYNTAX(2),NPRPRT,PRPEND,PREPOS,40,80,NREC,S1,S2)

1450 IF(PRED,NE,'MAKE') GO TO 1500
IF(ADPRED,NE,'SYNON') GO TO 8000
CALL EXIST(SYNTAX,2,VOCAB,KPART(2),S1)
IF(OBJ1,NE,'') AND,PROB1,NE,'') GO TO 1451
TYPE 9105
9105 FORMAT(' <<I NEED BOTH WORDS QUOTED-G')
GO TO 1
1451 CALL POS(SYNTAX(10),NREC,VOCAB)
IF(NREC,EQ,0) GO TO 8000
GO TO (1452,1453,1454,1455,1456) NREC
1452 CALL REDUCE(SYNTAX,10,NVRT,VEND,VERB,40,80)
CALL SYNON(SYNTAX,2,10,NVRT,VEND,VERB,40,80,NREC,S1,S2)
1453 CALL REDUCE(SYNTAX,10,NNRT,NEND,NOUN,80,160)
CALL SYNON(SYNTAX,2,10,NNRT,NEND,NOUN,80,160,NREC,S1,S2)
1454 CALL REDUCE(SYNTAX,10,NADVRT,ADVEND,ADVERB,40,80)
CALL SYNON(SYNTAX,2,10,NADVRT,ADVEND,ADVERB,40,80,NREC,S1,S2)
1455 CALL REDUCE(SYNTAX,10,NADJRT,ADJEND,ADJCTV,40,80)

```

```

1456 CALL SYNON(SYNTAX,2,10,NADJRT,ADJEND,ADJCTV,40,80,NREC,S1,S2)
CALL REDUCE(SYNTAX,10,NPRPRT,PRPEND,PREPOS,40,80)
CALL SYNON(SYNTAX,2,10,NPRPRT,PRPEND,PREPOS,40,80,NREC,S1,S2)

1500 IF(PRED.NE.'MOVE') GO TO 2000
CALL NEWDAT(DWORD)
IF(ADPRED.EQ.'') GO TO 1504
IF(ADPRED.NE.'FORWA') GO TO 1502
ADMOD=RECORD+ADMOD
1501 CALL MOVDSK(ADMOD)
GO TO 2
1502 IF(ADPRED.NE.'BACKW') GO TO 8000
ADMOD=RECORD-ADMOD
GO TO 1501
1504 IF(PREP1.NE.'') GO TO 1506
ADMOD=RECORD+1
GO TO 1501
1506 IF(PREP1.NE.'TO') GO TO 8000
CALL MOVDSK(PRMOD1)
GO TO 2

2000 IF(PRED.NE.'DISPL') GO TO 3000
IF(OBJ1.NE.'K') GO TO 2010
IF(OBMOD1.NE.'CONTI') GO TO 2005
CALL SURFAC(K,NR,NCONT,2)
GO TO 2
2005 IF(OBMOD1.NE.'DISCR') GO TO 2008
CALL SURFAC(K(1,NCONT+1),NR,NLINES,1)
GO TO 2
2008 IF(OBMOD1.LT.1.OR.OBMOD1.GT.NC) GO TO 8000
IF(OBMOD1.LE.NCONT) CALL GRAFMX
; ('K-BUFFERS','ENERGY(MEV)S',K(1,OBMOD1),NR,1,1)
IF(OBMOD1.GT.NCONT) CALL GRAFVP
; ('K-BUFFERS','ENERGY(MEV)S',K(1,OBMOD1),EDATA,NR)
GO TO 2

2010 IF(OBJ1.NE.'O') GO TO 2200
IF(OBMOD1.NE.'CONTI') GO TO 2015
CALL SURFAC(O,NR,NCONT,2)
GO TO 2
2015 IF(OBMOD1.NE.'DISCR') GO TO 2018
CALL SURFAC(O(1,NCONT+1),NR,NLINES,1)
GO TO 2
2018 IF(OBMOD1.LT.1.OR.OBMOD1.GT.NC) GO TO 8000
CALL GRAFMX('O-BUFFERS','ENERGY(MEV)S',O(1,OBMOD1),NR,1,1)
GO TO 2

2200 IF(OBJ1.NE.'PHI') GO TO 2250
CALL DISPEC(1)
GO TO 2
2250 IF(OBJ1.NE.'ERRCK') GO TO 2300
CALL DISPEC(2)
GO TO 2

2300 IF(OBJ1.NE.'DATA') GO TO 2400
IF(OBJ2.EQ.'DATA') GO TO 2310
CALL GRAFVP('AIRBORNE DATA','ENERGY(MEV)S',DATA,EDATA,NR)
GO TO 2
2310 CALL GETSW(UNFSW,1,II)

```

```

IF(II,EQ,0) TEMP(1)=CHISQ(X)
CALL GRAFPR('AIRBORNE DATA AND REFOLDED SPECTRUMS',
, 'ENERGY(MEV)S',EDATA,DATA,DEXP,NR,1)
GO TO 2

2400 IF(OBJ1,NE,'DELTA') GO TO 2500
IF(OBMOD1,NE,' ') GO TO 2401
TEMP(1)=CHISQ(X)
CALL MULB(TEMP,TEMP,0,0,NR)
CALL GRAFPR('RESIDUALS', 'ENERGY(MEV)S',EDATA,DELTA,TEMP,NR,0)
GO TO 2
2401 CALL DISRES(OBMOD1,$8000)
GO TO 2

2500 IF(OBJ1,NE,'G') GO TO 2600
DO 2501 I=1,NR
2501 TEMP(I)=SQRT(G(I))
CALL GRAFVP(TEMP,EDATA,NR)
GO TO 2

2600 IF(OBJ1,NE,'DEXP') GO TO 2700
CALL DISFIT(OBMOD1,$8000)
GO TO 2

2700 IF(OBJ1,NE,'SURVE') GO TO 2800
DO 2701 I=1,NSEC
2701 TEMP(I)=RECORD=NSEC+I-1
CALL THINDO(0,1023,0,300)
IF(OBMOD1,EQ,'POTAS') CALL GRAFP
, ('POTASSIUM SURVEYS','SECONDDS',POTASI,TEMP,NSEC)
IF(OBMOD1,EQ,'URANI') CALL GRAFP
, ('URANIUM SURVEYS','SECONDDS',URANI,TEMP,NSEC)
IF(OBMOD1,EQ,'THORI') CALL GRAFP
, ('THORIUM SURVEYS','SECONDDS',THORI,TEMP,NSEC)
CALL RESTOR
GO TO 2

2800 IF(OBJ1,NE,'PARAM') GO TO 2805
IF(OBMOD1,EQ,' ') CALL DISPAR(0)
IF(OBMOD1,EQ,'UNFOL') CALL DISPAR(1)
IF(OBMOD1,EQ,'SIMUL') CALL DISPAR(2)
IF(OBMOD1,EQ,'INPUT') CALL DISPAR(3)
IF(OBMOD1,EQ,'OUTPU') CALL DISPAR(4)
GO TO 2
2805 IF(OBJ1,NE,'SWITC') GO TO 2850
CALL DISPAR(5)
GO TO 2

2850 IF(OBJ1,NE,'TIME') GO TO 2900
IF(OBMOD1,NE,'ELAPS') GO TO 8000
TYPE 9285,OTIME
9285 FORMAT(' <<ELAPSED TIME=',F6.2,' SECONDS-G')
GO TO 1

2900 IF(OBJ1,NE,'VOCAB') GO TO 2910
CALL DISVOC(NVRT,VEND,VERB,40,80,1)
CALL DISVOC(NNRT,NEND,NOUN,80,160,2)
CALL DISVOC(NADVRT,ADVEND,ADVERB,40,80,3)
CALL DISVOC(NADJRT,ADJEND,ADJECTV,40,80,4)
CALL DISVOC(NPRPRT,PRPEND,PREPOS,40,80,5)

```

```

GO TO 2
2910 IF(OBJ1,NE,'VERBS') GO TO 2920
CALL DISVOC(NVRT,VEND,VERB,40,80,1)
GO TO 2
2920 IF(OBJ1,NE,'NOUNS') GO TO 2930
CALL DISVOC(NNRT,NEND,NOUN,80,160,2)
GO TO 2
2930 IF(OBJ1,NE,'ADVER') GO TO 2940
CALL DISVOC(NADVRT,ADVEND,ADVERB,40,80,3)
GO TO 2
2940 IF(OBJ1,NE,'ADJEC') GO TO 2950
CALL DISVOC(NADJRT,ADJEND,ADJCTV,40,80,4)
GO TO 2
2950 IF(OBJ1,NE,'PREPO') GO TO 8000
CALL DISVOC(NPRORT,PRPEND,PREPOS,40,80,5)
GO TO 2
3000 IF(PRED,NE,'SIMUL') GO TO 4000
IF(OBJ1,NE,'K') GO TO 3050
CALL SIMULK
GO TO 2
3050 IF(OBJ1,NE,'DATA') GO TO 3060
IF(PRMOD1,EQ,' ') GO TO 3051
COUNTS=FMOD
3051 CALL SIMDAT
GO TO 2
3060 IF(OBJ1,NE,'SURVE') GO TO 8000
CALL CONDIS
CALL DATSIM
GO TO 2

4000 IF(PRED,NE,'CHECK') GO TO 4200
CALL OKMPY(TEMP(1))
TYPE 9400,TEMP(1)
9400 FORMAT(' <<THE RATIO OF RMS DIAGONAL TERMS'/
, 3X,'TO RMS OFF-DIAGONAL TERMS IS ',E8.3,'-G')
GO TO 1

4200 IF(PRED,NE,'ORDER') GO TO 4500
CALL ERGORD
GO TO 2

4500 IF(PRED,NE,'COMPU') GO TO 5000
IF(OBJ1,NE,'CHI') GO TO 8000
IF(PRMOD1,NE,'SQUAR') GO TO 8000
TEMP(1)=CHISQ(X)
TYPE 9450,TEMP(1)
9450 FORMAT(' <<CHI-SQUARED=',F7.3,' EXPECTATIONS-G')
GO TO 1

5000 IF(PRED,NE,'SET') GO TO 5100
CALL NEWPAR(SYNTAX(2),WORD,WORD,$8000,$2)
5100 IF(PRED,NE,'RESET'.AND.OBJ1,NE,'COPY') GO TO 5500
CALL SETSW(OUTSW,1,0)
CALL DONEPL
GO TO 2
5500 IF(PRED,NE,'ADJUS') GO TO 6000
CALL GAIN
GO TO 2

6000 IF(PRED,NE,'UNFOL') GO TO 6100

```

```

        IF(PROB1,NE,'MAZAS') GO TO 6050
        CALL SETSW(UNFSW,2,1)
        GO TO 6051
6050    IF(PROB1,NE,'') GO TO 8000
6051    CALL GETSW(UNFSW,2,II)
        IF(II,EQ,1) CALL MAZAS0
        GO TO 2

6100    IF(PRED,NE,'SURVE') GO TO 6900
6150    IF(OBJ1,EQ,'') GO TO 6151
        CALL NEWDAT(DWORD)
6151    IF(PREP1,EQ,'TO') CALL XCHANG(9,12,3)
        IF(PREP1,EQ,'FROM') GO TO 6152
        IF(PREP1,EQ,'') GO TO 6154
        GO TO 8000
6152    IF(PRM0D1,LT,1) GO TO 8000
        CALL MOVDSK(PRM0D1)
        IF(PREP2,EQ,'TO') GO TO 6153
        IF(PREP2,EQ,'') GO TO 6154
        GO TO 8000
6153    NSEC=PRM0D2-PRM0D1+1
        IF(NSEC,GT,256) NSEC=256
        GO TO 6155
6154    NSEC=256
6155    II=0
        DO 6156 I=1,NSEC
        CALL DATIN(I)
        IF(RECORD,LT,PRM0D1+I) GO TO 6157
        CALL MAZAS(II)
6156    II=II+1
6157    NSEC=II
        CALL MAZAS(II)
        GO TO 1

6900    IF(PRED,NE,'CHANG')GO TO 6950
        CALL CHANG
        GO TO 2
6950    IF(PRED,NE,'ANALY') GO TO 8000
        DATFIL=0WORD
        CALL RUNSUM(PRM0D1,PRM0D2)
        GO TO 2

8000    TYPE 9800
9800    FORMAT(' <<I DON'T UNDERSTAND~G')
        GO TO 1

END

```

SUBROUTINE CATGRY(POINT,CHAR,WORD,KPART,VOCAB)

- * HARVEST 'WORD' FROM CHARACTER STRING 'CHAR'
- * BEGINNING AT CHARACTER NUMBER 'POINT'
- * THEN DETERMINE PART OF SPEECH
- * WORD CONTAINS FIVE OR FEWER CHARACTERS LEFT JUSTIFIED

```
COMMON/SCRATC/IBUF(1022),QUOTE,DISCRE,DUMMY(256)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCDNT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
```

```
INTEGER CHAR(72),VOCAB(80,6),POINT,WORD,BLANK
LOGICAL QUOTE
EQUIVALENCE (IWORD,FWORD)
DATA BLANK,QUOTE,DISCRE/'      ',.FALSE.,.FALSE./
```

* FIND BEGINNING OF WORD

```
15  IF(POINT.GT.72) GO TO 9
    DO 1  I=POINT,72
    IF(CHAR(I).EQ.43.OR.CHAR(I).EQ.46) GO TO 2
    IF(CHAR(I).EQ.45.AND.CHAR(I+1).LT.64) GO TO 2
    IF(CHAR(I).GT.47.AND.CHAR(I).LT.59) GO TO 2
    IF(CHAR(I).GT.64.AND.CHAR(I).LT.91) GO TO 2
1   CONTINUE
    GO TO 9
2   IMIN=I
```

* FIND END OF WORD

```
DO 3  I=IMIN,72
IF(CHAR(I).EQ.43) GO TO 3
IF(CHAR(I).EQ.45.AND.CHAR(I+1).LT.64) GO TO 3
IF(CHAR(I).EQ.46.AND.CHAR(I+1).GT.64) GO TO 3
IF(CHAR(I).EQ.46.AND.CHAR(I-1).GT.64) GO TO 4
IF(CHAR(I).EQ.46) GO TO 3
IF(CHAR(I).GT.47.AND.CHAR(I).LT.58) GO TO 3
IF(CHAR(I).EQ.58) GO TO 103
IF(CHAR(I).GT.64.AND.CHAR(I).LT.91) GO TO 3
GO TO 4
3   CONTINUE
    I=73
    GO TO 4
103 IF(CHAR(IMIN).EQ.58.OR.CHAR(IMIN).EQ.68) DISCRE=.TRUE.
    POINT=I+1
    GO TO 13
4   POINT=I
    WORD=0
```

* TEXT OR NUMBER?

```
IF(CHAR(IMIN).LT.58.AND.(.NOT.QUOTE)) GO TO 7
```

* PACK FIRST FIVE CHARACTERS INTO WORD

```

      IMAX=IMIN+4
      DO 6 I=IMIN,IMAX
      IF(I,LT,POINT) GO TO 5
      WORD=128*WORD+32
      GO TO 6
5     WORD=128*WORD+CHAR(I)
6     CONTINUE

      WORD=WORD-17179869184
      WORD=2*WORD
      WORD=WORD-34359738367-1
      GO TO 10

```

* CONVERT TO INTEGER OR DETECT DECIMAL POINT

```

7     IMAX=POINT-1
      DO 8 I=IMIN,IMAX
      IF(CHAR(I).EQ.46) GO TO 19
8     WORD=10*WORD+CHAR(I)-48
      KPART=4
      IF(DISCRE) WORD=WORD+NCONT
      DISCRE=.FALSE.
      RETURN

```

* NO WORD WAS FOUND

```

9     WORD=BLANK
      KPART=8
      RETURN

```

* CATEGORIZE WORD AS A PART OF SPEECH:

KPART	PART OF SPEECH
0	UNRECOGNIZED
1	VERB
2	NOUN
3	ADVERB
4	ADJECTIVE
5	PREPOSITION
6	CONJUNCTION
8	BLANK
9	VERB AND NOUN
10	VERB AND ADJECTIVE

* KPART IS THE PART IDENTIFIER
 * WORD IS A NOUN IF IT IS QUOTED

```

10  IF(QUOTE) GO TO 110
    IMIN=IMIN-1
    IF(IMIN.EQ.0) GO TO 11
    IF(CHAR(IMIN).NE.34.AND.CHAR(IMIN).NE.39) GO TO 11
    KPART=2
    WORD=WORD
    IF(POINT.LE.(IMIN+6)) RETURN
    POINT=IMIN+6
    QUOTE=.TRUE.
    RETURN
110  KPART=4
    QUOTE=.FALSE.
    RETURN

11  IF(WORD.NE.'AND'.AND.WORD.NE.'OR') GO TO 12
    KPART=6
    RETURN

12  IF(WORD.EQ.'THE') GO TO 13
  
```


ENTRY POS(WORD,KPART,VOCAB)

```

KPART=0
DO 16 KK=1,6
K=KK
IF(K.GT.2) K=K-1
DO 14 I=1,80
IF(VOCAB(I,KK).EQ.BLANK) GO TO 16
IF(WORD.EQ.VOCAB(I,KK)) GO TO 15
14 CONTINUE
GO TO 16
15 IF(K.EQ.3.OR.K.EQ.5) GO TO 17
KPART=KPART+K
16 CONTINUE
IF(KPART.EQ.0) RETURN
IF(KPART.EQ.3) KPART=9
IF(KPART.EQ.5) KPART=10
GO TO 16

17 KPART=K
18 IF(KPART.NE.4.AND.KPART.NE.10) RETURN
IF(WORD.EQ.'ZERO') WORD=0
IF(WORD.EQ.'ONE') WORD=1
IF(WORD.EQ.'TWO') WORD=2
IF(WORD.EQ.'THREE') WORD=3
IF(WORD.EQ.'FOUR') WORD=4
IF(WORD.EQ.'FIVE') WORD=5
IF(WORD.EQ.'SIX') WORD=6
IF(WORD.EQ.'SEVEN'.AND.CHAR(IMIN+6).EQ.32) WORD=7
IF(WORD.EQ.'EIGHT'.AND.CHAR(IMIN+6).EQ.32) WORD=8
IF(WORD.EQ.'NINE') WORD=9
IF(WORD.EQ.'TEN') WORD=10
IF(WORD.EQ.'ELEVE') WORD=11
IF(WORD.EQ.'TWELV') WORD=12
IF(WORD.EQ.'THIRT'.AND.CHAR(IMIN+6).EQ.69) WORD=13
IF(WORD.EQ.'FOURT'.AND.CHAR(IMIN+6).EQ.69) WORD=14
IF(WORD.EQ.'FIFTE') WORD=15
IF(WORD.EQ.'SIXTE') WORD=16
IF(WORD.EQ.'SEVEN'.AND.CHAR(IMIN+7).EQ.69) WORD=17
IF(WORD.EQ.'EIGHT'.AND.CHAR(IMIN+6).EQ.69) WORD=18
IF(WORD.EQ.'NINET'.AND.CHAR(IMIN+6).EQ.69) WORD=19
IF(WORD.EQ.'TWENT') WORD=20
IF(WORD.EQ.'THIRT'.AND.CHAR(IMIN+6).EQ.89) WORD=30
IF(WORD.EQ.'FORTY') WORD=40
IF(WORD.EQ.'FIFTY') WORD=50
IF(WORD.EQ.'SIXTY') WORD=60
IF(WORD.EQ.'SEVEN'.AND.CHAR(IMIN+7).EQ.89) WORD=70
IF(WORD.EQ.'EIGHT'.AND.CHAR(IMIN+6).EQ.89) WORD=80
IF(WORD.EQ.'NINET'.AND.CHAR(IMIN+6).EQ.89) WORD=90
RETURN

```

* CONVERT TO FLOATING POINT

```

19  WORD=0
    II=IMIN
    DO 23 I=1,3
      IBUF(I)=0
      DO 21 J=1,5
        IF(II.LT.POINT) GO TO 20
        IBUF(I)=128*IBUF(I)+32
        GO TO 21
      20  IBUF(I)=128*IBUF(I)+CHAR(II)
      21  II=II+1
        IF(IBUF(I).LT.17179869184) GO TO 22
        IBUF(I)=IBUF(I)-17179869184
        IBUF(I)=2*IBUF(I)
        IBUF(I)=IBUF(I)-34359738367-1
        GO TO 23
      22  IBUF(I)=2*IBUF(I)
      23  CONTINUE
    DECODE(15,9900,IBUF) FWORD
9000  FORMAT(G)
      WORD=IWORD
      KPART=4
      RETURN

    END

```

SUBROUTINE PARSE(KPART,WORD,*)

* DECOMPOSE COMMAND SENTENCE INTO FUNCTIONAL COMPONENTS

INDEX	FUNCTION
1	VERB
2	OBJECT ONE
3	OBJECT ONE MODIFIER
4	OBJECT TWO
5	OBJECT TWO MODIFIER
6	ADVERB
7	ADVERBIAL OBJECT
8	ADVERBIAL OBJECT MODIFIER
9	PREPOSITION ONE
10	PREPOSITION ONE OBJECT
11	PREPOSITION ONE MODIFIER
12	PREPOSITION TWO
13	PREPOSITION TWO OBJECT
14	PREPOSITION TWO MODIFIER

* SYNTAX(INDEX) IS EQUAL TO THE WORD PERFORMING
* FUNCTION(INDEX)

COMMON SYNTAX(14)
INTEGER KPART(14),SYNTAX,WORD(14)
DO 1 I=1,14
SYNTAX(I)=0

1

* FIND VERB

```

      DO 2 I=1,14
      IF(KPART(I).EQ.1) GO TO 3
      IF(KPART(I).EQ.9) GO TO 3
      IF(KPART(I).EQ.10) GO TO 3
2     CONTINUE
      TYPE 9000
9000  FORMAT(' <<I CAN'T FIND A VERB-G')
      RETURN 1
3     SYNTAX(1)=I
      DO 5 L=1,14
      IF(L.EQ.1) GO TO 4
      IF(KPART(L).EQ.9) KPART(L)=2
      IF(KPART(L).EQ.10) KPART(L)=4
      GO TO 5
4     KPART(I)=1
5     CONTINUE

```

* FIND PREPOSITIONAL PHRASES

```

      IMIN=1
      L=-1
10    L=L+1
      IF(L.EQ.2) GO TO 20

      DO 11 I=IMIN,13
      IF(KPART(I).EQ.5) GO TO 12
11    CONTINUE
      GO TO 20

12    IMIN=I+1
      IF(KPART(IMIN).EQ.7) IMIN=IMIN+1
      IF(IMIN.GT.14) GO TO 20
      IF(KPART(IMIN).EQ.2) GO TO 13
      IMIN=IMIN+1
      IF(IMIN.GT.14) GO TO 20
      IF(KPART(IMIN).NE.2) GO TO 10
      INOUN=IMIN
      IAD=IMIN-1
      IF(KPART(IAD).NE.4) GO TO 15
      GO TO 14

```

```

13  INOUN=IMIN
    IMIN=IMIN+1
    IF(IMIN.GT.14) GO TO 15
    IF(KPART(IMIN).NE.4) GO TO 15
    IAD=IMIN
14  IMIN=IMIN+1
    SYNTAX(11+3*L)=IAD
15  SYNTAX(10+3*L)=INOUN
    SYNTAX(9+3*L)=I
    IF(IMIN.GT.13) GO TO 20
    GO TO 10

```

* FIND VERBAL OBJECT

```

20  DO 21 I=1,14
    IF(KPART(I).NE.2) GO TO 21
    IF(SYNTAX(10).EQ.I) GO TO 21
    IF(SYNTAX(13).EQ.I) GO TO 21
    GO TO 22
21  CONTINUE
    GO TO 40

22  L=-1
23  L=L+1
    IF(L.EQ.2) GO TO 30
    INOUN=I
    IF(INOUN.EQ.1) GO TO 28
    IF(KPART(INOUN-1).EQ.4) GO TO 24
28  IF(INOUN.EQ.14) GO TO 27
    IF(KPART(INOUN+1).EQ.4) GO TO 25
    GO TO 27

```

```

24   IAD=INOUN-1
      IF(SYNTAX(11).EQ.IAD) GO TO 28
      IF(SYNTAX(14).EQ.IAD) GO TO 28
      GO TO 26
25   IAD=INOUN+1
      I=I+1
26   SYNTAX(3+2*L)=IAD
27   SYNTAX(2+2*L)=INOUN
      I=I+1
      IF(I.GT.13) GO TO 30
      IF(KPART(I).NE.6) GO TO 30
      I=I+1
      IF(KPART(I).EQ.2) GO TO 23
      I=I+1
      IF(I.GT.14) GO TO 40
      IF(KPART(I).EQ.2) GO TO 23

```

* FIND ADVERB AND ADVERBIAL OBJECT

```

30   IMIN=I
      DO 31 I=1,14
      IF(KPART(I).EQ.3) GO TO 32
31   CONTINUE
      GO TO 40

32   SYNTAX(6)=I
      DO 33 I=IMIN,14
      IF(KPART(I).NE.2) GO TO 33
      IF(SYNTAX(2).EQ.I) GO TO 33
      IF(SYNTAX(4).EQ.I) GO TO 33
      IF(SYNTAX(10).EQ.I) GO TO 33
      IF(SYNTAX(13).EQ.I) GO TO 33
      GO TO 34
33   CONTINUE
      GO TO 40

```

```

34  SYNTAX(7)=I
    IAD=I-1
    IF(KPART(IAD),NE,4) GO TO 40
    IF(SYNTAX(3),EQ,IAD) GO TO 40
    IF(SYNTAX(5),EQ,IAD) GO TO 40
    IF(SYNTAX(11),EQ,IAD) GO TO 40
    IF(SYNTAX(14),EQ,IAD) GO TO 40
    SYNTAX(8)=IAD

```

* SUBSTITUTE WORD CONTENT FOR INDICES

```

40  DO 42 I=1,14
    IF(SYNTAX(I),EQ,0) GO TO 41
    INOUN=SYNTAX(I)
    SYNTAX(I)=WORD(INOUN)
    GO TO 42
41  SYNTAX(I)= ' '
42  CONTINUE
    RETURN

```

- * DISTINCTIONS BETWEEN VERBAL OBJECT AND ADVERBIAL OBJECT,
- * BETWEEN OBJECT ONE AND OBJECT TWO, AND BETWEEN PREPOSITION
- * ONE AND PREPOSITION TWO, CAN BE AMBIGUOUS. THIS ENTRY
- * POINT ALLOWS THEM TO BE INTERCHANGED.

```

    ENTRY XCHANG(I1,I2,NUMBER)

    IMAX=NUMBER-1
    DO 50 I=0,IMAX
    INOUN=SYNTAX(I1+I)
    SYNTAX(I1+I)=SYNTAX(I2+I)
50  SYNTAX(I2+I)=INOUN

    RETURN
    END

```

```

SUBROUTINE REDUCE(WORD,IWORD,NROOT,PEND,PART,NPEND,NPART)
* RECOGNIZE A SYNONYMOUS ROOT WORD AND SUBSTITUTE IT FOR WORD(IWORD).
  INTEGER WORD(14),PEND(NPEND),PART(NPART)

  KMAX=0
  DO 1 I=1,NROOT
    KMIN=KMAX+1
    KMAX=PEND(I)
    DO 1 K=KMIN,KMAX
      IF(WORD(IWORD).EQ.PART(K)) GO TO 2
1    CONTINUE

  RETURN

2  WORD(IWORD)=PART(KMAX)
  RETURN
  END

```


SUBROUTINE REFER(KPART,WORD,SYNTAX,VOCAB,*)

* 'IT' OR 'THEM' REFERS TO THE OBJECT OF THE PRECEDING COMMAND

INTEGER WORD(14),SYNTAX(14),KPART(14),VOCAB(80,6)

DO 1 I=1,14
IF(WORD(I).EQ.'IT') GO TO 2
IF(WORD(I).EQ.'THEM') GO TO 2
1 CONTINUE
RETURN

2 IMIN=I+4
IF(IMIN.GT.14) RETURN 1
DO 3 I=14,IMIN,-1
KPART(I)=KPART(I-3)
3 WORD(I)=WORD(I-3)
DO 4 I=1,4
WORD(IMIN-I)=SYNTAX(6-I)
4 CALL PD3(WORD(IMIN-I),KPART(IMIN-I),VOCAB)

RETURN
END

SUBROUTINE GETSW(SWITCH,N,VALUE)

- * SWITCH IS AN INTEGER CONTAINING EIGHT 4-BIT VALUES
- * VALUE(SWITCH,N) IS THE N-TH VALUE, 0-15

INTEGER SWITCH,VALUE

VALUE=SWITCH/16**(N-1)
VALUE=MOD(VALUE,16)
RETURN
END

SUBROUTINE SETSW(SWITCH,N,NEWVAL)

- * NEWVAL BECOMES THE N-TH VALUE OF SWITCH

INTEGER SWITCH,VALUE
CALL GETSW(SWITCH,N,VALUE)
NEWVAL=MOD(NEWVAL,16)
VALUE=(NEWVAL-VALUE)*16**(N-1)
SWITCH=SWITCH+VALUE
RETURN
END

★ SWITCH DIRECTORY

★	★	★	★
★	SWITCH	POSITION	STATE
★	★	★	★
★	UNFSW(1)	1	SPECTRUM PROJECTED FORWARD
★	UNFSW(2)	0	ENERGY WINDOW METHOD
★	UNFSW(2)	1	MAZAS
★	UNFSW(3)	1	CONTINUUM UNFOLDED
★	UNFSW(4)	1	TEMPORAL RESPONSE ON
★	UNFSW(5)	1	COMPUTE VARIANCE MATRIX
★	UNFSW(6)	0	2*UNFSW(7)+1-SEC RUNNING SUM
★	UNFSW(6)	1	TIME FILTER
★	UNFSW(7)	0-15	SUM 2*UNFSW(7)+1 SECONDS
★	UNFSW(8)		
★	SIMSW(1)	1	DATA SIMULATED
★	SIMSW(2)		
★	SIMSW(3)		
★	SIMSW(4)		
★	SIMSW(5)		
★	SIMSW(6)		
★	SIMSW(7)		
★	SIMSW(8)		
★	OUTSW(1)	0	NO HARD COPY
★	OUTSW(1)	1	HARD COPY
★	OUTSW(2)	1	UNDER SURFACE VISIBLE
★	OUTSW(3)		
★	OUTSW(4)		
★	OUTSW(5)		
★	OUTSW(6)		
★	OUTSW(7)		
★	OUTSW(8)		
★	INSW(1)	1	DATA FILE OPEN
★	INSW(2)	1	CHANNEL STRUCTURE ALTERED
★	INSW(3)		
★	INSW(4)		
★	INSW(5)		
★	INSW(6)		
★	INSW(7)		
★	INSW(8)		

★ ZERO IS THE STANDARD OFF POSITION
 ★ PLEASE ENTER NEW STATES AS THEY ARE CODED

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

SUBROUTINE ADDV(VCTR1,VCTR2,VCTR3,N)
* VCTR3=VCTR1+VCTR2
DIMENSION VCTR1(N),VCTR2(N),VCTR3(N)
DO 1 I=1,N
1 VCTR3(I)=VCTR1(I)+VCTR2(I)
RETURN
END

SUBROUTINE SUBV(VCTR1,VCTR2,VCTR3,N)
* VCTR3=VCTR1-VCTR2
DIMENSION VCTR1(N),VCTR2(N),VCTR3(N)
DO 1 I=1,N
1 VCTR3(I)=VCTR1(I)-VCTR2(I)
RETURN
END

SUBROUTINE MULS(VCTR1,VCTR2,SCALAR,N)
*** VCTR2=SCALAR*VCTR1**
DIMENSION VCTR1(N),VCTR2(N)
DO 1 I=1,N
1 VCTR2(I)=SCALAR*VCTR1(I)
RETURN
END

SUBROUTINE MULSP(VCTR1,VCTR2,SCALAR,N)
* VCTR2=VCTR2+SCALAR*VCTR1
DIMENSION VCTR1(N),VCTR2(N)
DO 1 I=1,N
1 VCTR2(I)=VCTR2(I)+SCALAR*VCTR1(I)
RETURN
END

SUBROUTINE MULSM(VCTR1,VCTR2,SCALAR,N)
*** VCTR2=VCTR2-SCALAR*VCTR1**
DIMENSION VCTR1(N),VCTR2(N)
DO 1 I=1,N
1 VCTR2(I)=VCTR2(I)-SCALAR*VCTR1(I)
RETURN
END

FUNCTION SUMV(VECTOR,N)
*** SUM THE COMPONENTS OF VECTOR, DIMENSION N**
DIMENSION VECTOR(N)
SUMV=0.0
DO 1 I=1,N
1 SUMV=SUMV+VECTOR(I)
RETURN
END

```

FUNCTION DOTG(G,VCTR1,VCTR2,N)
* DOT PRODUCT (VCTR1,VCTR2)
* VCTR1 AND VCTR2 ARE VECTORS, DIMENSION N
* G IS THE DIAGONAL METRIC, DIMENSION N

    DIMENSION G(N),VCTR1(N),VCTR2(N)

    DOTG=0.0
    DO 1 I=1,N
1   DOTG=DOTG+G(I)*VCTR1(I)*VCTR2(I)

    RETURN
END

```

SUBROUTINE PNORM(VECTOR,SCALAR,N)
*** RENORMALIZE VECTOR, DIMENSION N, SO IT SUMS TO UNITY**
DIMENSION VECTOR(N)
SCALAR=SUMV(VECTOR,N)
SCALAR=SCALAR(-1.)**
CALL MULS(VECTOR,VECTOR,SCALAR,N)
RETURN
END

```

      FUNCTION DOT(VCTR1,VCTR2,N)
      * DOT PRODUCT (VCTR1,VCTR2) IN CARTESIAN COORDINATES
      * VCTR1 AND VCTR2 ARE VECTORS, DIMENSION N

      DIMENSION VCTR1(N),VCTR2(N)

      DOT=0.0
      DO 1 I=1,N
1      DOT=DOT+VCTR1(I)*VCTR2(I)

      RETURN
      END

```

```

SUBROUTINE RENORM(G,VECTOR,SCALAR,N)
* RENORMALIZE VECTOR, DIMENSION N, TO UNIT LENGTH
* G IS THE DIAGONAL METRIC, DIMENSION N

DIMENSION G(N),VECTOR(N)

SCALAR=DOTG(G,VECTOR,VECTOR,N)
SCALAR=SCALAR**(-.5)

CALL MULS(VECTOR,VECTOR,SCALAR,N)

RETURN
END

```

```

SUBROUTINE POSLIM(VECTOR,N,VMIN)
★ VECTOR IS GREATER THAN OR EQUAL TO VMIN
  DIMENSION VECTOR(N)
  DO 1 I=1,N
    IF(VECTOR(I).LT.VMIN) VECTOR(I)=VMIN
1  CONTINUE
  RETURN
END

```


**THIS PAGE
WAS INTENTIONALLY
LEFT BLANK**

```

MMM      MMM      AAAAAAAAAA      TTTTTTTTTTTTTTTT      RRRRRRRRRRRR
MMM      MMM      AAAAAAAAAA      TTTTTTTTTTTTTTTT      RRRRRRRRRRRR
MMM      MMM      AAAAAAAAAA      TTTTTTTTTTTTTTTT      RRRRRRRRRRRR
MMMMMM   MMMMM   AAA      AAA      TTT      TTT      RRR      RRR
MMMMMM   MMMMM   AAA      AAA      TTT      TTT      RRR      RRR
MMMMMM   MMMMM   AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAAAAAAAAA      TTT      TTT      RRR      RRR
MMM      MMM      AAAAAAAAAA      TTT      TTT      RRR      RRR
MMM      MMM      AAAAAAAAAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR
MMM      MMM      AAA      AAA      TTT      TTT      RRR      RRR

```

```

FFFFFFFFFFFFFF      00000000      RRRRRRRRRRRR
FFFFFFFFFFFFFF      00000000      RRRRRRRRRRRR
FFFFFFFFFFFFFF      00000000      RRRRRRRRRRRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFFFFFFFFFFFFF      000      000      RRRRRRRRRRRR
FFFFFFFFFFFFFF      000      000      RRRRRRRRRRRR
FFFFFFFFFFFFFF      000      000      RRRRRRRRRRRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR
FFF      000      000      RRR      RRR

```

LPTSPL VERSION 6(344) RUNNING ON LOT011
 START USER BENDIX-ERDA #2670,26700 JOB XFR SEQ. 3130 DATE 26-OCT-78 11:16:33 M
 REQUEST CREATED: 26-OCT-78 08:41:22
 FILE: DSKC0:MATRIX.FOP#2670,26700 CREATED: 25-SEP-78 16:08:00 <255> PRINTED: 26-O
 QUEUE SWITCHES: /PRINT:ARROW /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:200 /FORMS:8

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

SUBROUTINE BASIS(R,N)
*** CREATE AN ORTHONORMAL BASIS IN N-DIMENSIONAL SPACE**
DIMENSION R(N,N)
DO 1 K=1,N
DO 2 J=1,N
2 R(J,K)=0.0
1 R(K,K)=1.0
RETURN
END

```

SUBROUTINE EQUATE(KOLD,KNEW,NR,NC)
* SET MATRIX KNEW = MATRIX KOLD, DIMENSION NR-BY-NC
  DIMENSION KOLD(256,NC),KNEW(256,NC)
  REAL KOLD,KNEW

  DO 1 J=1,NC
  DO 1 I=1,NR
1  KNEW(I,J)=KOLD(I,J)

  RETURN
  END

```

SUBROUTINE ORTHO(G,O,R,NR,NC)

* ORTHOGONALIZE O BY GRAM-SCHMIDT PROCESS
* R IS THE REDUCED RESPONSE, INITIALLY R=IDENTITY
* G IS THE DIAGONAL METRIC, DIMENSION NR

DIMENSION O(NR,NC),R(NC,NC),G(NR)

DO 3 J=1,NC

DO 1 K=1,NC

IF(K.EQ.J) GO TO 2

SCALAR=DOTG(G,O(1,K),O(1,J),NR)

CALL MULSM(O(1,K),O(1,J),SCALAR,NR)

1 CALL MULSM(R(1,K),R(1,J),SCALAR,NC)

2 CALL RENORM(G,O(1,J),SCALAR,NR)

3 CALL MULS(R(1,J),R(1,J),SCALAR,NC)

RETURN

END

SUBROUTINE ORTHOT(G,O,R,S,NR,NC,TAU)

- * ORTHOGONALIZE O BY GRAM-SCHMIDT PROCESS
- * R IS THE A PRIORI CORRELATION MATRIX
- * S IS THE REDUCED RESPONSE, INITIALLY S=IDENTITY
- * G IS THE DIAGONAL METRIC, DIMENSION NR
- * TAU IS THE REGULARIZATION PARAMETER

COMMON/UPARAM/X(2),N
 DIMENSION O(256,NC),R(N,NC),S(NC,NC),G(NR)

DO 3 J=1,NC

DO 1 K=1,NC

IF(K.EQ.J) GO TO 2

SCALAR=DOTG(G,O(1,K),O(1,J),NR)

SCALAR=SCALAR+TAU*DOT(R(1,K),R(1,J),NC)

CALL MULSM(O(1,K),O(1,J),SCALAR,NR)

CALL MULSM(R(1,K),R(1,J),SCALAR,NC)

1 CALL MULSM(S(1,K),S(1,J),SCALAR,NC)

2 SCALAR=DOTG(G,O(1,J),O(1,J),NR)

SCALAR=SCALAR+TAU*DOT(R(1,J),R(1,J),NC)

SCALAR=SCALAR**-.5

CALL MULS(O(1,J),O(1,J),SCALAR,NR)

CALL MULS(R(1,J),R(1,J),SCALAR,NC)

3 CALL MULS(S(1,J),S(1,J),SCALAR,NC)

RETURN

END

```

SUBROUTINE INVERT(S,N)
* INVERT MATRIX S, DIMENSION N LESS THAN 32
COMMON/SCRATC/X(29),I,J,K,SS(961),G(31)
DIMENSION S(N,N)

CALL BASIS(SS,N)
DO 1 I=1,N
1  G(I)=1.0
   CALL ORTHO(G,S,SS,N,N)

DO 3 I=1,N
DO 2 K=1,N
G(K)=S(I,K)
2  S(I,K)=0.0
DO 3 J=1,N
DO 3 K=1,N
3  S(I,J)=S(I,J)+SS(J+(K-1)*N)*G(K)

RETURN
END

```



```

SUBROUTINE ROMPY(R)
* MULTIPLY/TRANSPOSE R TIMES O(TRANSPOSE)

COMMON/UPARAM/UNFSH, NR, NC
COMMON/DETECT/EDATA(256), DATA(256), DELTA(256),
DEXP(256), G(256), O(256,32), RESPON(256,32)
DIMENSION R(NC, NC)

DO 2 I=1, NR
DO 1 K=1, NC
DELTA(K)=O(I, K)
1 O(I, K)=0.0

DO 2 J=1, NC
DO 2 K=1, NC
2 O(I, J)=O(I, J)+R(J, K)*DELTA(K)

RETURN
END

```


THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

```

      SUBROUTINE PROJFD(DEXP,K,PHI,NR,NC)
      * FORWARD PROJECTION, DEXP=K(NU)*PHI(NU)
      * WHERE K(NU) IS THE NU-TH RESPONSE VECTOR
      * AND DEXP IS THE EXPECTED DATA VECTOR

      DIMENSION DEXP(NR),K(256,NC),PHI(NC)
      REAL K

      CALL MULS(DEXP,DEXP,0,0,NR)

      DO 1 NU=1,NC
1      CALL MULSP(K(1,NU),DEXP,PHI(NU),NR)

      DO 2 I=1,NR
      IF(DEXP(I).GE.1.0) GO TO 2
      DEXP(I)=EXP(DEXP(I)-1.)
2      CONTINUE

      RETURN
      END

```

```

      SUBROUTINE PROJ8K(PHI,O,DELTA,G,NR,NC)
      * BACK PROJECTION, PHI(NU)=O(NU)*DELTA
      * WHERE O(NU) AND DELTA ARE NR-DIMENSIONAL VECTORS
      * G IS THE DIAGONAL METRIC, DIMENSION NR
      DIMENSION PHI(NC),O(256,NC),DELTA(NR),G(NR)
      DO 1 NU=1,NC
1    PHI(NU)=DOTG(G,O(1,NU),DELTA,NR)
      RETURN
      END

```

SUBROUTINE DKMPY(RATIO)

- * MULTIPLY O(TRANPOSE) TIMES K
- * RATIO IS THE RMS DIAGONAL ELEMENT DIVIDED BY
- * THE RMS OFF-DIAGONAL ELEMENT

```
COMMON/UPARAM/UNFSW,NR,NC
COMMON/DETECT/X(256,4),G(256),O(256,32),K(256,32)
REAL K
```

```
DENUM=0.0
DENOM=0.0
DO 5 I=1,NC
DENOM=DENOM+DOT(O(1,I),K(1,I),NR)**2
DO 5 J=1,NC
IF(I.EQ,J) GO TO 5
DENOM=DENOM+DOT(O(1,I),K(1,J),NR)**2
5 CONTINUE
```

```
DENUM=SQRT(DENUM/NC)
DENOM=SQRT(DENOM/(NC*(NC-1)))
RATIO=DENUM/DENOM
```

```
RETURN
END
```

FUNCTION CHISQ(X)

* CHI-SQUARED

COMMON/UPARAM/UNFSW,NR,NC
COMMON/SOURCE/EPHI(256),PHI(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
G(256),O(256,32),K(256,32)
REAL K

CALL GETSW(UNFSW,3,I)
IF(I.EQ.0) CALL MAZASC

CALL PROJFD(DEXP,K,PHI,NR,NC)

DO 1 I=1,NR

IF(DEXP(I).GE.1.0) GO TO 1

DEXP(I)=EXP(DEXP(I)-1.)

1 DELTA(I)=(DATA(I)-DEXP(I))/SQRT(DEXP(I))

CHISQ=0.0

DO 2 I=1,NR

2 CHISQ=CHISQ+DELTA(I)**2

CHISQ=CHISQ/NR

CALL SETSW(UNFSW,1,1)

RETURN

END

SUBROUTINE MAZASO

- * A MAZE UNFOLDING ROUTINE ORIGINALLY DEVELOPED
- * FOR AIRBORNE SURVEYS OF K,U,T SPECTRA
- * BY ERDA/BENDIX AND THEIR SUBCONTRACTORS

```

COMMON/SCRATC/A(1024)
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;   WOBBLE,CTOD,NGAUSS,XNORM
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;   TAU,NCONT,NLINES,COUNTS,EMAX
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;   G(256),O(256,32),K(256,32)
INTEGER VALUE,SIMSW,UNFSW
REAL K

CALL MULS(PHI(NCONT+1),PHI(NCONT+1),0,0,NLINES)
DO 1 NU=NCONT+1,NC
1  PHI(NU)=PHI(NU)+DOT(O(1,NU),DATA,NR)
DO 2 NU=NCONT+1,NC
  IF(PHI(NU).GE.1.0) GO TO 2
  PHI(NU)=EXP(PHI(NU)-1.)
2  CONTINUE

CALL SETSW(UNFSW,1,0)
CALL SETSW(UNFSW,3,0)
RETURN

ENTRY MAZASC.

* CONTINUUM UNFOLDING

CALL MULS(PHI,PHI,0,0,NCONT)
DO 3 NU=1,NCONT
3  PHI(NU)=PHI(NU)+DOT(O(1,NU),DATA,NR)

CALL SETSW(UNFSW,3,1)
RETURN
END

```


SUBROUTINE MAZAS(ETIME)

- * UNFOLD AND TIME FILTER A SEQUENCE OF KUT DATA
- * UP TO AND INCLUDING 256 SECONDS

```

COMMON/SCRATCH/NDIM, N, N2, NHALF, W(252), KAL(256), HAL(256),
; KBAR, UBAR, TBAR, SIGMAK, SIGMAU, SIGMAT,
; FK(42), FU(42), FT(42), F(42), H(42),
; I1, I2, MIN, MAX, ALPHA(3), BETA(3), MODE2, MODE6
COMMON/SPARAM/SKIPS(24), ELINE(32)
COMMON/UPARAM/UNFSW, NR, NC, SLOPE, ZERO, RESOLU, POWER,
; TAUD, NCONT, NLINES, COUNTS, TAUC, NSEC, NPOTAS, NURANI,
; NTHORI, ALTITU, ETIME, KUT(32), SEQUEN(32)
COMMON/SOURCE/EPHI(256), PHI(256)
COMMON/DETECT/SKIP(256, 4), G(256), O(256, 26),
; POTASI(256), URANI(256), THORI(256), KGK(676), WEIGHT(32)
COMMON/IPARAM/INSW(3), RECORD
DIMENSION XMU(18)
INTEGER VALUE, SIMSW, UNFSW, RECORD
REAL KGK, KBAR
DATA XMU/2.262E-03, 4.350E-03, 3.524E-03, 3.132E-03, 2.871E-03,
; 2.610E-03, 2.262E-03, 2.088E-03, 1.827E-03, 1.697E-03,
; 4.437E-03, 3.437E-03, 3.132E-03, 2.784E-03, 2.697E-03,
; 2.436E-03, 2.088E-03, 1.653E-03/

IF(ETIME.GT.0) GO TO 1
CALL MULS(POTASI, POTASI, 0, 0, NSEC)
CALL MULS(URANI, URANI, 0, 0, NSEC)
CALL MULS(THORI, THORI, 0, 0, NSEC)
CALL GETSW(UNFSW, 2, MODE2)
CALL GETSW(UNFSW, 6, MODE6)
CALL GETSW(UNFSW, 7, NHALF)
N=2*NHALF
NDIM=N+1
IF(MODE6.EQ.0) GO TO 1
CALL CTRANS(42)
CALL MULS(ALPHA, ALPHA, 0, 0, 3)
BETA(1)=1.0
BETA(2)=1.0
BETA(3)=1.0

1 IF(ETIME.EQ.NSEC) GO TO 4
CALL MULS(H(2), H, 1.0, N)
CALL MULS(FK(2), FK, 1.0, N)
CALL MULS(FU(2), FU, 1.0, N)
CALL MULS(FT(2), FT, 1.0, N)
H(NDIM)=ALTITU
IF(MODE2.EQ.0) CALL WINDO(FK(NDIM), FU(NDIM), FT(NDIM), S4)
IF(MODE2.EQ.1) CALL MAZASO
PATH=ALTITU-400

FK(NDIM)=PHI(NCONT+1)*EXP(+XMU(1)*(PATH))/ETIME

MIN=NCONT+2
MAX=MIN+NURANI-1
FU(NDIM)=0.0
DO 2 MU=MIN, MAX
2 FU(NDIM)=FU(NDIM)+WEIGHT(MU)*PHI(MU)*EXP(+XMU(MU-NCONT)*PATH)
FU(NDIM)=FU(NDIM)/ETIME

```

```

MIN=MAX+1
MAX=MIN+NTHORI-1
FT(NDIM)=0.0
DO 3 MU=MIN,MAX
3 FT(NDIM)=FT(NDIM)+WEIGHT(MU)*PHI(MU)*EXP(+X*MU(MU=NCONT)*PATH)
FT(NDIM)=FT(NDIM)/ETIME

4 IF(ETIME.LT.N) RETURN
IF(MODE6.EQ.0) GO TO 7
IF(ETIME.EQ.NSEC) GO TO 6

5 ALTITU=SUMV(H,NDIM)/FLOAT(NDIM)
CALL GETSW(UNFSW,4,I)
CALL ALTCOR(I,ALTITU)

6 CALL FILTER(ETIME,FK,G,F,POTASI,ALPHA(1),BETA(1))
CALL FILTER(ETIME,FU,G,F,URANI,ALPHA(2),BETA(2))
CALL FILTER(ETIME,FT,G,F,THORI,ALPHA(3),BETA(3))
GO TO 9

7 CALL RUNAVG(ETIME,FK,POTASI)
CALL RUNAVG(ETIME,FU,URANI)
CALL RUNAVG(ETIME,FT,THORI)

9 IF(ETIME.LT.NSEC) RETURN
I1=RECORD-NSEC
I2=RECORD-1
CALL MULS(KBAR,KBAR,0.0,6)
KBAR=SUMV(POTASI,NSEC)/NSEC
UBAR=SUMV(URANI,NSEC)/NSEC
TBAR=SUMV(THORI,NSEC)/NSEC
DO 10 I=1,NSEC
SIGMAK=SIGMAK+(POTASI(I)-KBAR)**2
SIGMAU=SIGMAU+(URANI(I)-UBAR)**2
10 SIGMAT=SIGMAT+(THORI(I)-TBAR)**2
SIGMAK=SQRT(SIGMAK/NSEC)
SIGMAU=SQRT(SIGMAU/NSEC)
SIGMAT=SQRT(SIGMAT/NSEC)

TYPE 9000,I1,I2,KBAR,SIGMAK,UBAR,SIGMAU,TBAR,SIGMAT
9000 FORMAT (' <<RECORD',I4,' TO RECORD',I4/
, 16X,'MEAN',3X,'SIGMA'/
, '<<POTASSIUM',2F8.3/
, '<<URANIUM ',2F8.3/
, '<<THORIUM ',2F8.3,'-G')

RETURN
END

```

```

SUBROUTINE ALTCDR(MODE,ALTITU)
* ALTITUDE-DEPENDENT TEMPORAL RESPONSE

COMMON/SCRATC/NDIM,SKIP(255),KAL(256),HAL(256)
REAL KUT,KAL

IF(MODE,EQ,0) GO TO 3
DO 1 I=1,NDIM
S=SQRT(((I-1)*205.333)**2+ALTITU*ALTITU)
KAL(I)=EXP(-1.903E-03*S)
KAL(I)=KAL(I)*(.05+.95*ALTITU/S)
1 KAL(I)=KAL(I)/S**2

CALL FTRANS(KAL,HAL)

DO 2 I=NDIM,1,-1
2 HAL(I)=HAL(I)/HAL(1)
RETURN

3 DO 4 I=1,NDIM
4 HAL(I)=1.0
RETURN

END

```

SUBROUTINE WINDO(SUMK,SUMU,SUMT,*)

* WINDOW METHOD

```
COMMON/SCRATC/SKIP(1010),N1,N2,N3,N4,N5,N6,A,B,G
COMMON/UPARAM/UNFSH,NR,NC,SLOPE,ZERO,RESOLU,POWER,
,      TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPOTAS,NURANI,
,      NTHORI,ALTITU,ETIME
COMMON/DETECT/EDATA(256),DATA(257),COSMIC,DELTA(254),DEXP(256)
DATA ALFA1,BETA1,GAMA1/.4,.52,.91/
DATA ALFA2,BETA2,GAMA2/.43,.56,.95/
DATA BGK,BGU,BGTH/.04668,.04131,.0498/

N1=(1.361-ZERO)/SLOPE+.5
N2=(1.561-ZERO)/SLOPE+.5
N3=(1.665-ZERO)/SLOPE+.5
N4=(1.865-ZERO)/SLOPE+.5
N5=(2.414-ZERO)/SLOPE+.5
N6=(2.814-ZERO)/SLOPE+.5

SUMK=0.
SUMU=0.
SUMT=0.
DO 10 I=N1,N2
10  SUMK=SUMK+DATA(I)
DO 20 I=N3,N4
20  SUMU=SUMU+DATA(I)
DO 30 I=N5,N6
30  SUMT=SUMT+DATA(I)

SUMK=(SUMK-BGK*COSMIC)/ETIME
SUMU=(SUMU-BGU*COSMIC)/ETIME
SUMT=(SUMT-BGTH*COSMIC)/ETIME

A=ALFA1+(ALTITU*.3048-50.)*(ALFA2-ALFA1)/75.
B=BETA1+(ALTITU*.3048-50.)*(BETA2-BETA1)/75.
G=GAMA1+(ALTITU*.3048-50.)*(GAMA2-GAMA1)/75.
SUMU=SUMU-A*SUMT
SUMK=SUMK-B*SUMT-G*SUMU

SUMK=SUMK*EXP(2.3E-03*(ALTITU-400.))
SUMU=SUMU*EXP(1.7E-03*(ALTITU-400.))
SUMT=SUMT*EXP(1.7E-03*(ALTITU-400.))

RETURN 1
END
```

SUBROUTINE CTRANS(MDIM)

* COSINE TRANSFORM, DIMENSIONALITY=MDIM=NDIM < 127
 * F IS THE SPATIAL VECTOR, G THE WAVE AMPLITUDE VECTOR

```

COMMON/SCRATC/NDIM,N,N2,NHALF,W(252)
DIMENSION F(NDIM),G(NDIM)
DATA PI/3.14159265/
NDIM=MDIM
N=NDIM-1
N2=2*N
NHALF=NDIM/2

PHINRM=PI/N
W(1)=2.0
PHI=0.0
DO 1 I=2,N2
  PHI=PHI+PHINRM
  W(I)=2.*COS(PHI)
1  RETURN
  
```

ENTRY FTRANS(F,G)

* FORWARD COSINE TRANSFORM

```

DO 2 I=1,NDIM
  II=I-1
  G(I)=F(1)+F(NDIM)*(-1)**II
  IJ=I
  DO 2 J=2,N
    G(I)=G(I)+W(IJ)*F(J)
    IJ=IJ+II
    IF(IJ.GT.N2) IJ=IJ-N2
  2  CONTINUE

  RETURN
END
  
```

SUBROUTINE FILTER(ITIME,FU,G,F,URANI,ALPHA,BETA)

* COSINE-SPACE FILTER

```

COMMON/SCRATC/NDIM,N,N2,SKIP(509),HAL(256)
COMMON/UPARAM/UNFSH,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPDAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
DIMENSION A(2,2),FU(NDIM),G(NDIM),F(NDIM),URANI(NSEC)

CALL FTRANS(FU,G)
IF(ITIME.EQ.0) MAX=3
IF(ITIME.GT.0) MAX=1
DO 3 ITER=1,MAX

  A(1,1)=0.0
  A(1,2)=0.0
  A(2,2)=0.0
  DO 1 I=2,NDIM
    X=(HAL(I)/(I-1))**2
    W=1./(ALPHA*X+BETA)
    A(1,1)=A(1,1)+X**W
    A(1,2)=A(1,2)+W*X
    A(2,2)=A(2,2)+W
    DET=A(1,1)*A(2,2)-A(1,2)**2
    W=A(2,2)
    A(2,2)=+A(1,1)/DET
    A(1,1)=+      A/DET
    A(1,2)=-A(1,2)/DET

  SUM1=0.0
  SUM2=0.0
  DO 2 I=2,NDIM
    X=(HAL(I)/(I-1))**2
    W=1./(ALPHA*X+BETA)
    SUM1=SUM1+(A(1,1)*X+A(1,2))*W**G(I)**2
    SUM2=SUM2+(A(2,2)+A(1,2)*X)*W**G(I)**2
  2

  ALPHA=SUM1
  BETA=SUM2
  3

  DO 4 I=2,NDIM
    X=(HAL(I)/(I-1))**2
    W=G(I)**2/BETA
    W=W/(W+((ALPHA*X+BETA)/(ALPHA*X))**2)
    W=W/HAL(I)
    G(I)=W**G(I)
    CALL BTRANS(F,G)
  4

  IF(ITIME.EQ.NSEC) GO TO 12
  IF(ITIME.GT.N) GO TO 11
  DO 10 I=1,16
    URANI(I)=F(I)
    URANI(17)=9.950469E-01*F(17)
    URANI(18)=9.804689E-01*F(18)
    URANI(19)=9.101563E-01*F(19)
    URANI(20)=7.460938E-01*F(20)
    URANI(21)=5.000000E-01*F(21)
    URANI(22)=2.539063E-01*F(22)
    URANI(23)=8.984375E-02*F(23)
  10

```

```

URANI(24)=1.953125E-02*F(24)
URANI(25)=1.953125E-03*F(25)
11  URANI(ITIME=N+17)=URANI(ITIME=N+17)+1.953125E-03*F(17)
    URANI(ITIME=N+18)=URANI(ITIME=N+18)+1.757813E-02*F(18)
    URANI(ITIME=N+19)=URANI(ITIME=N+19)+7.031250E-02*F(19)
    URANI(ITIME=N+20)=URANI(ITIME=N+20)+1.640625E-01*F(20)
    URANI(ITIME=N+21)=URANI(ITIME=N+21)+2.460938E-01*F(21)
    URANI(ITIME=N+22)=URANI(ITIME=N+22)+2.460938E-01*F(22)
    URANI(ITIME=N+23)=URANI(ITIME=N+23)+1.640625E-01*F(23)
    URANI(ITIME=N+24)=URANI(ITIME=N+24)+7.031250E-02*F(24)
    URANI(ITIME=N+25)=URANI(ITIME=N+25)+1.757813E-02*F(25)
    URANI(ITIME=N+26)=URANI(ITIME=N+26)+1.953125E-03*F(26)
    RETURN

12  URANI(NSEC=NDIM+18)=URANI(NSEC=NDIM+18)+1.953125E-03*F(18)
    URANI(NSEC=NDIM+19)=URANI(NSEC=NDIM+19)+1.953125E-02*F(19)
    URANI(NSEC=NDIM+20)=URANI(NSEC=NDIM+20)+8.984375E-02*F(20)
    URANI(NSEC=NDIM+21)=URANI(NSEC=NDIM+21)+2.539063E-01*F(21)
    URANI(NSEC=NDIM+22)=URANI(NSEC=NDIM+22)+5.000000E-01*F(22)
    URANI(NSEC=NDIM+23)=URANI(NSEC=NDIM+23)+7.460938E-01*F(23)
    URANI(NSEC=NDIM+24)=URANI(NSEC=NDIM+24)+4.101563E-01*F(24)
    URANI(NSEC=NDIM+25)=URANI(NSEC=NDIM+25)+9.804688E-01*F(25)
    URANI(NSEC=NDIM+26)=URANI(NSEC=NDIM+26)+9.980469E-01*F(26)
    DO 13 I=25,NDIM
13  URANI(NSEC=NDIM+I)=F(I)
    RETURN
    END

```

SUBROUTINE RUNAVG(ITIME,F,URANI)

* RUNNING AVERAGE TIME FILTER

```

COMMON/SCRATC/NDIM,N,N2,NHALF,I1,I2,ICTP
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
,      TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC
DIMENSION F(NDIM),URANI(NSEC)

IF(ITIME.EQ.NSEC) GO TO 12
IF(ITIME.GT.N) GO TO 10
I1=1
I2=NHALF+1
ICTR=I2
GO TO 14
10  I1=ITIME+1-NHALF
    I2=I1
    ICTR=I2
    GO TO 14
12  IF(NHALF.EQ.0) RETURN
    I1=NSEC+1-NHALF
    I2=NSEC
    ICTR=I1-1
14  DO 16 I=I1,I2
      DO 16 IDEL=-NHALF,+NHALF,+1
        II=(I-ICTR)+IDEL+(NHALF+1)
        IF(II.LT.1.OR.II.GT.NDIM) II=II-2*IDEL
16  URANI(I)=URANI(I)+F(II)/NDIM

RETURN
END

```


SUBROUTINE BTRANS(F,G)

* BACKWARD COSINE TRANSFORM

COMMON/SCRATC/NDIM,N,N2,NHALF,W(252)
DIMENSION F(NDIM),G(NDIM)

```
10  DO 10 J=1,NDIM  
    F(J)=G(1)  
  
    DO 20 I=2,NHALF  
      II=I-1  
      IJ=1  
      DO 20 J=1,NDIM  
        F(J)=F(J)+G(I)*W(IJ)  
        IJ=IJ+II  
      IF(IJ,GT,N2) IJ=IJ-N2  
20  CONTINUE  
  
30  DO 30 J=1,NDIM  
    F(J)=F(J)/(2*N)  
  
    RETURN  
    END
```

SUBROUTINE LAGUER(M,PN,PN1,PN2,E,N)

* GENERATE M-TH ORDER LAGUERRE VECTOR PN
* FROM (M-1)-TH ORDER PN1 AND (M-2)-TH ORDER PN2
* AT N ENERGIES E(I), I=1,N

COMMON/UPARAM/PARAM(8),NCONT,NLINES,COUNTS,EMAX
DIMENSION PN(N),PN1(N),PN2(N),E(N)

IF(M,LT.2) GO TO 2

AMN1=2*M-1

AMN2=M-1

AMD=M

AMD=1./AMD

AMN2=AMN2*AMD

DO 1 I=1,N

X=NCONT*E(I)/3.

1 PN(I)=(AMN1-X)*AMD*PN1(I)-AMN2*PN2(I)
RETURN

2 IF(M.EQ.1) GO TO 4

DO 3 I=1,N

3 PN(I)=1.0.

RETURN

4 DO 5 I=1,N

X=NCONT*E(I)/3.

5 PN(I)=1.-X

RETURN

END

```

SUBROUTINE REBIN(DATA1,EDATA1,SLOPE1,NR1,
DATA2,EDATA2,SLOPE2,NR2)
,
* TRANSFORM NR1-DIMENSIONAL DATA1 WITH CHANNEL CENTERS EDATA1
* INTO NR2-DIMENSIONAL DATA2 WITH CHANNEL CENTERS EDATA2

DIMENSION DATA1(NR1),EDATA1(NR1),DATA2(NR2),EDATA2(NR2)

DELTA=SLOPE2/SLOPE1
DELCUB=DELTA**3/12.

IC=1
IM=1
IP=2
DELERG=EDATA1(2)-EDATA1(1)
DO 3 I=1,NR2

X=EDATA2(I)
1 IF(ABS(EDATA1(IC)-X).LE.(ABS(EDATA1(IP)-X))) GO TO 2
IC=IP
IM=IC-1
IP=IC+1
IF(IP.GT.NR1) IP=NR1
DELERG=(EDATA1(IP)-EDATA1(IM))/(IP-IM)
GO TO 1

2 X=(X-EDATA1(IC))/DELERG
XSQ=X**2
T=(-.08333333+XSQ)*DELTA+DELCUB
S=X*DELTA
DATA2(I)=(T-S)*DATA1(IM)+(T+S)*DATA1(IP)
3 DATA2(I)=.5*DATA2(I)+((1.0833333-XSQ)*
DELTA-DELCUB)*DATA1(IC)

RETURN
END

```

FUNCTION DATBIN(AI,SLOPE,ZERO)

- * COMPUTE DATA ENERGY AT CHANNEL POSITION AI
- * INTEGER VALUES ARE ASSUMED TO BE AT CHANNEL CENTERS
- * PARAMETERS ARE SLOPE MEV/CHANNELS
- * AND ZERO ENERGY IN MEV AT AI=0

PHU=SLOPE*AI+ZERO
IF(PHU.GT.3.0) GO TO 1

DATBIN=-.01387+PHU*(1.03+PHU*(-.05439
+PHU*(.0835+PHU*(-.03701+.00519*PHU))))

RETURN

1 DATBIN=1.034827*PHU

RETURN
END

```

      SUBROUTINE BINDAT(EDATA,NR,SLOPE,ZERO)
      * CONSTRUCT CHANNEL CENTER ENERGY VECTOR, EDATA
      DIMENSION EDATA(NR)
      DO 1 I=1,NR
      AI=I
1      EDATA(I)=DATBIN(AI,SLOPE,ZERO)

      RETURN
      END

```

SUBROUTINE SETUP

* INITIALIZE THE UNFOLDING PROBLEM

```

COMMON/SCRATC/SUM,LAMBDA
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;   NOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;   TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPOTAS,NURANI,
;   NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/IPARAM/INSH,DATFIL,RECORD,LOCHAN
COMMON/SOURCE/EPHI(256),PHI(256),TMP(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),
;   DEXP(256),G(256),O(256,32),K(256,32)
DIMENSION KKG(676),WEIGHT(32)
EQUIVALENCE (S,SUM),(R,EDATA(1)),(KKG,O(1,30)),(WEIGHT,O(165,32))
REAL K,KUT,KKG
DOUBLE PRECISION DATFIL

```

* CONSTRUCT THE K-MATRIX

```

NC=32
NCONT=NC-NLINES
CALL BINDAT(EDATA,NR,SLOPE,ZERO)

IF(NCONT.EQ.3) GO TO 4
CALL LAGUER(0,K(1,4),K(1,4),K(1,4),EDATA,NR)
IF(NCONT.GT.4) CALL LAGUER(1,K(1,5),K(1,5),K(1,5),EDATA,NR)
IF(NCONT.LT.6) GO TO 2
DO 1 LAMBDA=6,NCONT
1   CALL LAGUER(LAMBDA-4,K(1,LAMBDA),K(1,LAMBDA-1),K(1,LAMBDA-2),
;   EDATA,NR)

2   DO 3 I=1,NR
      IF(I.EQ.1) DELERG=EDATA(2)-EDATA(1)
      IF(I.GT.1.AND.I.LT.NR) DELERG=.5*(EDATA(I+1)-EDATA(I-1))
      IF(I.EQ.NR) DELERG=EDATA(NR)-EDATA(NR-1)
      DO 3 LAMBDA=4,NCONT
3     K(I,LAMBDA)=K(I,LAMBDA)*DELERG

```

* CONSTRUCT THE O-BUFFER

```

4   CALL MULS(DATA,TMP,1.0,256)
      CALL MULS(DATA,G,1.0,NR)
      CALL POSLIM(G,NR,1.0)
      DO 11 L=1,3
      IF(L.EQ.1) GO TO 6

      CALL MAZASO
      CALL MAZASC
      CALL PROJFD(G,K,PHI,NR,NC)

6   SUM=0.0
      DO 7 I=1,NR
      SUM=SUM+G(I)
7   G(I)=1./G(I)
      CALL MULS(G,G,SUM,NR)
      IF(L.LT.3) GO TO 9

8   CALL MAZASC

```

```

      IF(NCONT.GT.3) CALL MULS(K(1,4),K(1,4),PHI(4),NR)
      IF(NCONT.LT.5) GO TO 82
      DO 81 J=5,NCONT
81     CALL MULSP(K(1,J),K(1,4),PHI(J),NR)
82     CALL EQUATE(K(1,NCONT+1),K(1,5),NR,NLINES)
      NCONT=4
      NC=NLINES+4

9      DO 92 J=1,NCONT
      DO 91 I=1,NCONT
      IF(I.EQ.J) GO TO 92
91     CALL MULSM(K(1,I),K(1,J),DOTG(G,K(1,I),K(1,J),NR),NR)
92     CALL RENORM(G,K(1,J),SUM,NR)
      CALL PRIOR(R)
      CALL BASIS(S,NC)
      CALL EQUATE(K,D,NR,NC)
      CALL ORTHOT(G,D,R,S,NR,NC,TAUD*NR)
      CALL ROMPY(S)
      DO 10 J=1,NC
      DO 10 I=1,NR
10     O(I,J)=O(I)*O(I,J)
      IF(L.LT.3) GO TO 11
      CALL PRIOR(R)
      CALL ERROR(KGX,WEIGHT,R,TAUD*NR)
11     CALL MULS(TMP,DATA,1.0,256)

      CALL BINDAT(EDATA,NR,SLOPE,ZERO)
      CALL MULS(EPHI,EPHI,0.0,NCONT)
      CALL MULS(ELINE,EPHI(NCONT+1),1.0,NLINES)

      RETURN
      END

```

```

      SUBROUTINE ERROR(KGK,WEIGHT,R,TAU)
* COMPUTE COVARIANCE MATRIX

      COMMON/SCRATC/MU,NU
      COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;          TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPOTAS,NURANI,
;          NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
      COMMON/DETECT/DATA(1024),G(256),O(256,32),K(256,32)
      DIMENSION KGK(NC,NC),WEIGHT(NC),R(NC,NC)
      REAL K,KGK

      CALL GETSW(UNFSW,5,MU)
      IF(MU.EQ.0) GO TO 2
      CALL MULS(KGK,KGK,0.0,NC*NC)

      DO 1 MU=1,NC
      DO 1 NU=1,NC
1      KGK(MU,NU)=KGK(MU,NU)
;          +DOTG(G,K(1,MU),K(1,NU),NR)
;          +TAU*DOT(R(1,MU),R(1,NU),NC)
      CALL INVERT(KGK,NC)

* COMPUTE WEIGHT VECTOR
* FOR COMBINING PHI(URANIUM) AND PHI(THORIUM)

2      CALL MULS(WEIGHT,WEIGHT,0.0,NC)
      WEIGHT(NCONT+4)=1./6
      WEIGHT(NCONT+5)=1./6
      WEIGHT(NCONT+6)=1./6
      WEIGHT(NCONT+7)=1./6
      WEIGHT(NCONT+8)=1./6
      WEIGHT(NCONT+9)=1./6
      WEIGHT(NCONT+16)=1./3
      WEIGHT(NCONT+17)=1./3
      WEIGHT(NCONT+18)=1./3

      RETURN
      END

```


SUBROUTINE GAIN

* ADJUST THE DATA SLOPE AND ZERO TO MINIMIZE CHI-SQUARED

```

COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
; UNFTAU,NCDNT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
; NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),
; DEXP(256),G(256),O(256,32),K(256,32)
COMMON/IPARAM/INSH,DATFIL(2),RECORD,LOCHAN
COMMON/SCRATC/DATAIN(256),EDATA1(256),EDATA2(256),TAU,
; W(8),ALPHA(5),DZ(4),DS(4),SLOPIN,ZEROIN,
; SLOPE0,ZERO0,DET,CENTER
DIMENSION DAT0(256)

DZ(2)=.1*SLOPE
DZ(1)=.5*DZ(2)
DZ(3)=.5*DZ(2)

DS(2)=.0000000
DS(1)=.8660254*DZ(2)*(4./NR)
DS(3)=-DS(1)

CALL MULS(DATA,DAT0,1.0,NR)
CALL MULS(DATA(NR+1),DATA,1.0,LOCHAN-1)
CALL MULS(DAT0,DATA(LOCHAN),1.0,NR)
CALL MULS(DATA,DAT0,1.0,256)

TAU=.08
CENTER=(LOCHAN-1)+.5*(NR+1)
SLOPIN=SLOPE
ZEROIN=ZERO-SLOPE*(LOCHAN-1)
CALL BINDAT(EDATA2,256,SLOPIN,ZEROIN)
CALL MAZASO
W(7)=CHISQ(X)

1 IF(LOCHAN.EQ.1) GO TO 2
CALL MULS(DATA,DAT0,1.0,NR)
CALL MULS(DATA(NR+1),DATA,1.0,LOCHAN-1)
CALL MULS(DAT0,DATA(LOCHAN),1.0,NR)

2 CALL MULS(DATA,DAT0,1.0,256)
SLOPE0=SLOPIN
ZERO0=ZEROIN
DO 3 I=1,6
IF(I.LE.3) SLOPIN=SLOPE0+DS(I)
IF(I.LE.3) ZEROIN=ZERO0+DZ(I)-CENTER*DS(I)
IF(I.GE.4) SLOPIN=SLOPE0-DS(I-3)
IF(I.GE.4) ZEROIN=ZERO0-DZ(I-3)+CENTER*DS(I-3)
CALL BINDAT(EDATA1,256,SLOPIN,ZEROIN)
CALL REBIN(DAT0,EDATA1,SLOPIN,256,
; DATA,EDATA2,SLOPE,256)
CALL MULS(DATA(LOCHAN),DATA,1.0,NR)
CALL MAZASO
3 W(I)=CHISQ(X)

ALPHA(1)=.25*(W(1)+W(2)+W(3)-W(4)-W(5)-W(6))
ALPHA(2)=.2886751*(W(1)-W(3)-W(4)+W(6))
ALPHA(3)=.5773503*(W(1)-W(3)+W(4)-W(6))
ALPHA(4)=.5*(W(2)+W(5)-W(7))

```

```

ALPHA(5)=.3333333*(W(1)-.5*W(2)+W(3)+W(4)-.5*W(5)+W(6))-W(7)
ALPHA(4)=ALPHA(4)+TAU
ALPHA(5)=ALPHA(5)+TAU

DET=ALPHA(3)**2-4.*ALPHA(4)*ALPHA(5)
DZ(4)=2.*ALPHA(1)*ALPHA(5)-ALPHA(2)*ALPHA(3)
DZ(4)=(DZ(4)/DET)*DZ(2)
DS(4)=2.*ALPHA(2)*ALPHA(4)-ALPHA(1)*ALPHA(3)
DS(4)=(DS(4)/DET)*DZ(2)*(4./NR)

SLOPIN=SLOPE0+DS(4)
ZEROIN=ZERO0+DZ(4)-CENTER*DS(4)
CALL BINDAT(EDATA1,256,SLOPIN,ZEROIN)
CALL REBIN(DAT0,EDATA1,SLOPIN,256,
           DATA,EDATA2,SLOPE,256)
CALL MULS(DATA,DELTA,1.0,LOCHAN-1)
CALL MULS(DATA(LOCHAN),DATA,1.0,NR)
CALL MULS(DELTA,DATA(NR+1),1.0,LOCHAN-1)
CALL MAZAS0
W(8)=W(7)
W(7)=CHISQ(X)

IF(W(7).LT..995*W(8)) GO TO 1
IF(W(7).LT.W(8)) GO TO 4
W(7)=W(8)
CALL MULS(DATIN(LOCHAN),DATA,1.0,NR)
4      TYPE 9000,SLOPIN,ZEROIN,W(7)
9000   FORMAT(' <<INPUT GAIN: SLOPE=',1PE12.6,', ZERO=',1PE13.6,',',
           /3X,'YIELDS CHI-SQUARED=',0PF7.3)
      RETURN
      END

```

SUBROUTINE PRIOR(R)

* COMPUTE THE A PRIORI CORRELATION MATRIX

```

COMMON/SPARAM/SIMSH,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
DIMENSION R(NC,NC)

CALL BASIS(R,NC)
MIN=1
MAX=NCONT
IF(NCONT.EQ.0) GO TO 2
DO 1 I=MIN,MAX
1  R(I,I)=0.0

2  IF(NPOTAS.EQ.0) GO TO 4
MIN=MAX+1
MAX=MIN+NPOTAS-1
R(MIN,MIN)=0.0
R(MAX,MAX)=0.0
IF(NPOTAS.LE.2) GO TO 4
DO 3 I=MIN+1,MAX-1
W=ALOG(ELINE(I+1-NCONT)/ELINE(I-1-NCONT))
W=ALOG(ELINE(I-NCONT)/ELINE(I-1-NCONT))/W
R(I,I-1)=1.-W
R(I,I)=-1.
3  R(I,I+1)=W

4  IF(NURANI.EQ.0) GO TO 6
MIN=MAX+1
MAX=MIN+NURANI-1
R(MIN,MIN)=0.0
R(MAX,MAX)=0.0
IF(NURANI.LE.2) GO TO 6
DO 5 I=MIN+1,MAX-1
W=ALOG(ELINE(I+1-NCONT)/ELINE(I-1-NCONT))
W=ALOG(ELINE(I-NCONT)/ELINE(I-1-NCONT))/W
R(I,I-1)=1.-W
R(I,I)=-1.
5  R(I,I+1)=W

6  IF(NTHORI.EQ.0) GO TO 8
MIN=MAX+1
MAX=MIN+NTHORI-1
R(MIN,MIN)=0.0
R(MAX,MAX)=0.0
IF(NTHORI.LE.2) GO TO 8
DO 7 I=MIN+1,MAX-1
W=ALOG(ELINE(I+1-NCONT)/ELINE(I-1-NCONT))
W=ALOG(ELINE(I-NCONT)/ELINE(I-1-NCONT))/W
R(I,I-1)=1.-W
R(I,I)=-1.
7  R(I,I+1)=W

8  IF(NLINES.LE.NPOTAS+NURANI+NTHORI) RETURN
MIN=MAX+1
MAX=NC

```

SUBROUTINE SIMULK

* SIMULATE A DETECTOR RESPONSE MATRIX

```

COMMON/SCRATC/A(256),P(256),EFF
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;   WOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;   TAU,NCONT,NLINES,COUNTS,EMAX
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),
;   DEXP(256),G(256),O(256,32),K(256,32)
REAL K

CALL BINDAT(EDATA,NR,SLOPE,ZERO)
CALL MULS(EPHI,EPHI,0.0,NCONT)
CALL MULS(ELINE,EPHI(NCONT+1),1.0,NLINES)

IF(NCONT.GT.0) CALL LAGUER(0,K,K,K,EDATA,NR)
IF(NCONT.GT.1) CALL LAGUER(1,K(1,2),K(1,2),K(1,2),EDATA,NR)
IF(NCONT.LT.3) GO TO 2
DO 1 LAMBDA=3,NCONT
1  CALL LAGUER(LAMBDA-1,K(1,LAMBDA),K(1,LAMBDA-1),K(1,LAMBDA-2),
;   EDATA,NR)

2  SCALAR=0.0
   DO 4 I=1,NR
   IF(I.EQ.1) DELERG=EDATA(2)-EDATA(1)
   IF(I.GT.1.AND.I.LT.NR) DELERG=.5*(EDATA(I+1)-EDATA(I-1))
   IF(I.EQ.NR) DELERG=EDATA(NR)-EDATA(NR-1)
   G(I)=EXP(-6.*EDATA(I)/EMAX)*DELERG
   SCALAR=SCALAR+G(I)
   G(I)=1./G(I)
   IF(NCONT.EQ.0) GO TO 4
   DO 3 LAMBDA=1,NCONT
3  K(I,LAMBDA)=K(I,LAMBDA)*DELERG
4  CONTINUE
   CALL MULS(G,G,SCALAR,NR)

   IF(NCONT.GT.0) CALL ORTHO(G,K,A,NR,NCONT)
   IF(NCONT.GT.0) CALL ORTHO(G,K,A,NR,NCONT)

   CALL PROJED(G,K,PHISIM,NR,NCONT)
   CALL POSLIM(G,NP,.375)
   CALL PNDKM(G,SCALAR,NR)
   DO 8 I=1,NR
8  G(I)=1./G(I)

   IF(NLINES.EQ.0) GO TO 6
   LAMBDA=NCONT+1
   DO 5 I=1,NLINES
   CALL LINE(K(1,LAMBDA),NR,EPHI(LAMBDA),EFF,DIAMET,THICKN,
;   EDATA,P,RESOLU,POWER,NGAUSS)
5  LAMBDA=LAMBDA+1

6  CALL BASIS(A,NC)
   CALL EQUATE(K,O,NR,NC)
   CALL ORTHO(G,O,A,NR,NC)
   CALL POMPY(A)

```

```
7      DO 7 J=1,NC  
        DO 7 I=1,NR  
          O(I,J)=G(I)*O(I,J)  
        RETURN  
      END
```

SUBROUTINE SIMDAT

* SIMULATE A DATA SET WITH NUMBER OF EVENTS = 'COUNTS'

```
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLER,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX
COMMON/IPARAM/INSW
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;      G(256),O(256,32),K(256,32)
COMMON/SCRATC/A(256),P(256),EFF
INTEGER SIMSW,UNFSW,INSW
REAL K
```

* SIMSW(1)=1 MEANS DATA IS SIMULATED

```
1      CALL SETSW(INSW,1,1)
```

* GENERATE EXPECTATION VALUES

```
2      CALL MULS(EPHI,EPHI,0.0,NCONT)
      CALL MULS(ELINE,EPHI(NCONT+1),1.0,NLINES)
      CALL MULS(PHI,PHI,0.0,NC)
      CALL MULS(ERROR,ERROR,0.0,NC)

3      CALL PROJFD(DATA,K(1,NCONT+1),PHISIM(NCONT+1),NR,NLINES)
      CALL PROJFD(DEXP,K,PHISIM,NR,NCONT)
      A(1)=SUMV(DATA,NR)*(1.+CTOD)
      A(2)=SUMV(DEXP,NR)*(1.+CTOD)/CTOD
      CALL MULS(DEXP,DEXP,1./A(2),NR)
      CALL MULSP(DATA,DEXP,1./A(1),NR)
      XNORM=A(1)/COUNTS
```

* GENERATE MULTINOMIAL STATISTICS .

```
      CALL MULS(DEXP,DATA,COUNTS,NR)
      DO 7 I=1,257
      IF(I.GT.NR) GO TO 6
      CALL RANDP(DATA(I))
      GO TO 7
6      CALL RANDP(X)
7      CONTINUE

      RETURN
      END
```

SUBROUTINE COND15

* COMPUTE THE C-TO-D RATIO

```

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
      WOBBLER,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
      TAU,NCONT,NLINES,COUNTS,EMAX
COMMON/IPARAM/INSW
COMMON/SOURCE/EPHI(256),PHI(256),ERRDP(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
      G(256),O(256,32),K(256,32)
COMMON/SCRATC/A(256),P(256),EFF
INTEGER SIMSW,UNFSW,INSW
REAL K

CALL PROJFD(DATA,K(1,NCONT+1),PHISIM(NCONT+1),NR,NLINES)
CALL PROJFD(DEXP,K,PHISIM,NR,NCONT)
CTOD=SUMV(DEXP,NR)/SUMV(DATA,NR)

RETURN
END

```

IIIIIIIIII
IIIIIIIIII
IIIIIIIIII

NNN
NNN
NNN

NN
 NN
 NN

PPP
PPP
PPP

UUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUU

000
 0000
 0000
 0000

FFF
FFt
FFt

0000000000
0000000000
0000000000

KPR RRR
RPF RRR
RPF RRR

QUEUE SWITCHES: /PRINT:ARROW /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:128 /FORMS:8)

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

```

DIMENSION NUCDAT(645),IDATA(128)
DIMENSION HDR(50)
EQUIVALENCE (HDR(1),NUCDAT(1))

OPEN(UNIT=23,DEVICE='DSKC',FILE='GEODAT,A',
;      ACCESS='SEQIN',MODE='IMAGE',BUFFER COUNT=1,
;      ASSOCIATE VARIABLE=I23)
OPEN(UNIT=1,DEVICE='DSKC',FILE='GEODAT,B',
;      ACCESS='SEQOUT',MODE='IMAGE',BUFFER COUNT=1,
;      ASSOCIATE VARIABLE=I1)

READ(23) HDR

DO 10 ICYCLE=1,18
DO 10 IREC=1,19
IF(IREC.LT.19) READ(23) (NUCDAT(I),I=1,174)
IF(IREC.EQ.19) READ(23) NUCDAT
DO 1 I=1,127
1 IDATA(I)=NUCDAT(I+2)
IDATA(128)=NUCDAT(2)/262144
WRITE(1) IDATA
IF(IREC.LT.19) GO TO 10
DO 2 I=1,127
2 IDATA(I)=NUCDAT(I+514)
IDATA(128)=NUCDAT(514)/262144
WRITE(1) IDATA
10 CONTINUE

CLOSE(UNIT=23)
CLOSE(UNIT=1,PROTECTION='557')

STOP,END

```

SUBROUTINE EXIST(WORD,IWORD,VOCAB,KPART,*)

* CHECK FOR PRESENCE OF WORD(IWORD) IN THE VOCABULARY

INTEGER VOCAB(80,6),WORD(14)

IF(WORD(IWORD).NE.' ') GO TO 1

TYPE 9000

9000 FORMAT(' <<QUOTE THE WORD YOU WANT ME TO RECOGNIZE~G')
RETURN 1

1 CALL POS(WORD(IWORD),KPART,VOCAB)

IF(KPART.EQ.3) GO TO 3

IF(KPART.EQ.5) GO TO 5

IF(KPART.EQ.9) GO TO 9

IF(KPART.EQ.10) GO TO 10

RETURN

3 TYPE 9003

9003 FORMAT(' <<IT IS ALREADY AN ADVERB~G')
RETURN 1

5 TYPE 9005

9005 FORMAT(' <<IT IS ALREADY A PREPOSITION~G')
RETURN 1

9 TYPE 9009

9009 FORMAT(' <<IT IS ALREADY BOTH VERB AND NOUN~G')
RETURN 1

10 TYPE 9010

9010 FORMAT(' <<IT IS ALREADY BOTH VERB AND ADJECTIVE~G')
RETURN 1

END

SUBROUTINE XCLUDE(KPART,NREC,*)

* PREVENT DIS-ALLOWED COMBINATIONS OF MULTIPLY-DEFINED WORDS
* FROM BEING INPUT TO THE VOCABULARY. KPART IS THE PREVIOUSLY
* DEFINED PART OF SPEECH, NREC THE DESIRED NEW PART.

IF(KPART.EQ.0) RETURN
IF(KPART.EQ.1.AND.NREC.EQ.2) RETURN
IF(KPART.EQ.1.AND.NREC.EQ.4) RETURN
IF(KPART.EQ.2.AND.NREC.EQ.1) RETURN
IF(KPART.EQ.4.AND.NREC.EQ.1) RETURN

IF(KPART.NE.NREC) TYPE 9104
9104 FORMAT(' <<VERB/NOUN AND VERB/ADJECTIVE'/
; 3X,'ARE THE ONLY COMBINATIONS ALLOWED-G')
IF(KPART.EQ.NREC) TYPE 9105
9105 FORMAT(' <<IT HAS ALREADY BEEN ENTERED-G')
RETURN 1

END

SUBROUTINE GETPOS(KPART,NREC,*,*,*,*,*,*)

* GET THE DESIRED PART OF SPEECH AND BRANCH ACCORDINGLY
 * THE FIRST FIVE RETURNS ARE ACTION BRANCHES
 * THE SIXTH IS AN ERROR RETURN

```

      TYPE 9000
9000  FORMAT(' <<WHAT PART OF SPEECH?-G')
      7    ACCEPT 9001,NREC
9001  FORMAT(A5)
      IF(NREC.EQ.'VERB') NREC=1
      IF(NREC.EQ.'NOUN') NREC=2
      IF(NREC.EQ.'ADVER') NREC=3
      IF(NREC.EQ.'ADJEC') NREC=4
      IF(NREC.EQ.'PREPO') NREC=5
      CALL XCLINE(KPART,NREC,36)
      GO TO (1,2,3,4,5) NREC
      TYPE 9002
9002  FORMAT(' <<VERB, NOUN, ADVERB, ADJECTIVE, OR PREPOSITION?-G')
      GO TO 7

1      RETURN 1
2      RETURN 2
3      RETURN 3
4      RETURN 4
5      RETURN 5
6      RETURN 6

      END

```

```

      SUBROUTINE SYNON
      , (WORD, INEW, IOLD, NROOT, PEND, PART, NPEND, NPART, NREC, *, *)
      * ADD WORD(INEW) AS A SYNONYM FOR WORD(IOLD).
      * PEND IS THE LIST OF ROOT POINTERS, PART IS THE VOCABULARY,
      * THERE ARE NROOT ROOTS AND NWORD WORDS,
      * NREC IS THE RECORD NUMBER OF THE VOCABULARY
      * FOR THE AFFECTED PART OF SPEECH.

      COMMON/SCRATC/TEMP, ORDER, XTRA(1022)
      INTEGER TEMP
      LOGICAL ORDER
      INTEGER WORD(14), PEND(NPEND), PART(NPART)

      NWORD=PEND(NROOT)
      IF(NROOT.EQ.NPEND) GO TO 5
      IF(NWORD.EQ.NPART) GO TO 5

      DO 1 I=1, NROOT
      KMAX=PEND(I)
      IF(WORD(IOLD).EQ.PART(KMAX)) GO TO 2
1      CONTINUE
      RETURN 2
2      IMIN=I

      DO 3 I=IMIN, NROOT
      PEND(I)=PEND(I)+1
      DO 4 I=NWORD, KMAX, -1
      PART(I+1)=PART(I)
4      PART(KMAX)=WORD(INEW)
      I=IMIN
      IMIN=2
      IF(I.GT.1) IMIN=PEND(I-1)+2
      IMAX=PEND(I)-1
      IF(IMAX.LT.IMIN) GO TO 8
      GO TO 6

5      TYPE 9000
9000  FORMAT(' <<THERE ISN'T ENOUGH ROOM-G')
      RETURN 1

6      ORDER=.TRUE.
      DO 7 I=IMIN, IMAX
      IF(PART(I).GT.PART(I-1)) GO TO 7
      ORDER=.FALSE.
      TEMP=PART(I)
      PART(I)=PART(I-1)
      PART(I-1)=TEMP
7      CONTINUE
      IF(ORDER) GO TO 8
      GO TO 6

      * WRITE TO DISK

8      OPEN(UNIT=1, DEVICE='DSKC', FILE='VOCAB.DAT',
      , ACCESS='RANDOM', MODE='IMAGE', BUFFER COUNT=1,
      , RECORD SIZE=128)
      CLOSE(UNIT=1, PROTECTION='227')
      OPEN(UNIT=1, DEVICE='DSKC', FILE='VOCAB.DAT',
      , ACCESS='RANDOM', MODE='IMAGE', BUFFER COUNT=1,
      , RECORD SIZE=128)

```

```

IF(NREC.EQ.2) GO TO 9
WRITE(1#NREC) NROOT,PEND,PART
GO TO 10
9 WRITE(1#2) NROOT,(PEND(I),I=1,40),(PART(I),I=1,80)
WRITE(1#6) NROOT,(PEND(I),I=41,80),(PART(I),I=81,160)
10 CLOSE(UNIT=1,PROTECTION="557")
RETURN 2

END

```

```

      SUBROUTINE NEWWORD
      , (WORD, INEW, NROOT, PEND, PART, NPEND, NPART, NREC, *, *)
      * ADD WORD(INEW) TO THE VOCABULARY.
      * IT IS NOT SYNONYMOUS WITH AN EXISTING ENTRY.

      INTEGER WORD(14), PEND(NPEND), PART(NPART)

      NWORD=PEND(NROOT)
      IF(NROOT.EQ.NPEND) GO TO 20
      IF(NWORD.EQ.NPART) GO TO 20

      * PUT WORD IN ALPHABETICAL POSITION
      * FIND POSITION

      DO 1 I=1,NROOT
      K=PEND(I)
      IF(PART(K).GT.WORD(INEW)) GO TO 2
1      CONTINUE
      IMIN=NROOT+1
      K=NWORD+1
      GO TO 5
2      IMIN=I

      * MAKE AN OPENING

      DO 3 I=NROOT,IMIN,-1
3      PEND(I+1)=PEND(I)+1
      DO 4 I=NWORD,K,-1
4      PART(I+1)=PART(I)

      * INSERT NEW VALUES

      IF(IMIN.GT.1) GO TO 5
      PEND(IMIN)=1
      GO TO 6
5      PEND(IMIN)=PEND(IMIN-1)+1
6      PART(K)=WORD(INEW)
      NROOT=NROOT+1

      * WRITE TO DISK

      OPEN(UNIT=1,DEVICE='DSKC',FILE='VOCAB.DAT',
      , ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
      , RECORD SIZE=128)
      CLOSE(UNIT=1,PROTECTION='227')
      OPEN(UNIT=1,DEVICE='DSKC',FILE='VOCAB.DAT',
      , ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
      , RECORD SIZE=128)
      IF(NREC.EQ.2) GO TO 10
      WRITE(1#NREC) NROOT,PEND,PART
      GO TO 11
10      WRITE(1#2) NROOT,(PEND(I),I=1,40),(PART(I),I=1,80)
      WRITE(1#6) NROOT,(PEND(I),I=41,80),(PART(I),I=81,160)
11      CLOSE(UNIT=1,PROTECTION='557')
      RETURN 2

20      TYPE 9000
9000      FORMAT(' <<THERE ISN'T ENOUGH ROOM-G')
      RETURN 1

```


END

```

      SUBROUTINE XWORD
      ; (WORD,NROOT,PEND,PART,NPEND,NPART,NREC,*)
* DELETE 'WORD' FROM THE VOCABULARY
      INTEGER WORD,PEND(NPEND),PART(NPART)
* IS WORD A ROOT?
      NWORD=PEND(NROOT)
      DO 1 I=1,NROOT
      K=PEND(I)
      IF(WORD.EQ.PART(K)) GO TO 10
1      CONTINUE
* IS WORD A SYNONYM?
      DO 2 K=1,NWORD
      IF(WORD.EQ.PART(K)) GO TO 3
2      CONTINUE
      RETURN 2
* IT IS A SYNONYM OR A ROOT WORD
* WHOSE POINTER HAS ALREADY BEEN RE-POSITIONED
3      K=K+1
      DO 4 I=K,NWORD
4      PART(I-1)=PART(I)
      PART(NWORD)= ' '
      DO 5 I=1,NROOT
      IF(PEND(I).GE.K) PEND(I)=PEND(I)-1
5      CONTINUE
      GO TO 30
* IT IS A ROOT
10     IF(I.EQ.1.AND.K.GT.1) GO TO 20
      IF(I.GT.1.AND.(K-PEND(I-1)).GT.1) GO TO 20
      IF(I.EQ.NROOT) GO TO 12
      NROOT=NROOT-1
      IMIN=I
      DO 11 I=IMIN,NROOT
11     PEND(I)=PEND(I+1)
      PEND(NROOT+1)=0
      GO TO 3
12     PEND(NROOT)=0
      PART(K)= ' '
      NROOT=NROOT-1
      GO TO 30
20     TYPE 9000
9000   FORMAT(' <<DELETE SYNONYMS BEFORE ROOT WORDS-G')
      RETURN 1
* WRITE TO DISK
30     OPEN(UNIT=1,DEVICE='DSKC',FILE='VOCAB.DAT',
      ; ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
      ; RECORD SIZE=128)
      CLOSE(UNIT=1,PROTECTION='227')

```

```

OPEN(UNIT=1,DEVICE='DSKC',FILE='VOCAB.DAT',
,      ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
,      RECORD SIZE=128)
IF(NREC.EQ.2) GO TO 31
WRITE(1#NREC) NROOT,PEND,PART
GO TO 32
31  WRITE(1#2) NROOT,(PEND(I),I=1,40),(PART(I),I=1,80)
    WRITE(1#6) NROOT,(PEND(I),I=41,80),(PART(I),I=81,160)
32  CLOSE(UNIT=1,PROTECTION='557')
    RETURN 2
END

```

```

SUBROUTINE READAT
* READ FILES 'PARAMS.DAT', 'SOURCE.DAT', AND 'DETECT.DAT'

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLER,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)

COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
COMMON/SOURCE/EPhi(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;      G(256),O(256,32),K(256,32)

REAL K,KUT
INTEGER UNFSW,SIMSW,OUTSW,INSW,RECORD
DOUBLE PRECISION DATFIL

OPEN(UNIT=20,DEVICE='DSKC',FILE='PARAMS.DAT',
;      ACCESS='RANDOM',MODE='IMAGE',
;      BUFFER COUNT=1,RECORD SIZE=32)
READ(20#1) UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME
NCONT=NC-NLINES
READ(20#2) KUT
READ(20#3) SEQUEN
READ(20#4) SIMSW,MSTATE,NTRY,DIAMET,THICKN,
;      WOBBLER,CTOD,NGAUSS,XNORM
READ(20#5) ELINE
READ(20#6) PHISIM
READ(20#7) OUTSW,PHI1,THETA,PHI2,ROTATE
READ(20#8) INSW,DATFIL,RECORD,LOCHAN
CLOSE(UNIT=20)

ENTRY VECDAT
OPEN(UNIT=21,DEVICE='DSKC',FILE='SOURCE.DAT',
;      ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
;      ASSOCIATE VARIABLE=I21,RECORD SIZE=NC)
READ(21#1) (EPhi(I),I=1,NC)
READ(21#2) (PHI(I),I=1,NC)
READ(21#3) (ERROR(I),I=1,NC)

OPEN(UNIT=22,DEVICE='DSKC',FILE='DETECT.DAT',
;      ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
;      ASSOCIATE VARIABLE=I22,RECORD SIZE=NR)
READ(22#1) (EDATA(I),I=1,NR)
READ(22#2) (DATA(I),I=1,NR)
READ(22#3) (DELTA(I),I=1,NR)
READ(22#4) (DEXP(I),I=1,NR)
READ(22#5) (G(I),I=1,NR)
NREC=5
DO 2 J=1,32
NREC=NREC+1
2 READ(22#NREC) (O(I,J),I=1,NR)
DO 3 J=1,NC
NREC=NREC+1
3 READ(22#NREC) (K(I,J),I=1,NR)

```

CALL SETSW(INSW,1,0)

RETURN
END

SUBROUTINE NEWPAR(PARAM,WORD,FWORD,*,*)

* UPDATE PARAMETER VALUES

```

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLER,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      YAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
INTEGER PARAM(2),WORD(14),SIMSW,UNFSW,OUTSW,INSW,RECORD
DOUBLE PRECISION DATFIL
DIMENSION FWORD(14)
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),
;      DEXP(256),G(256),O(256,32),K(256,32)
COMMON/SCRATC/M1(8),M2(8),TEMP,ORDER,E1,E2
REAL K,KUT

I=0
J=PARAM(2)
1 I=I+1
  IF(WORD(I).EQ.'TO') GO TO 2
  IF(I.LT.13) GO TO 1
  IF(PARAM(1).EQ.'HAZAS') GO TO 2
  IF(PARAM(1).EQ.'WINDO') GO TO 2
  IF(PARAM(1).EQ.'FILTE') GO TO 2
  IF(PARAM(1).EQ.'SUM') GO TO 2
  IF(PARAM(1).EQ.'COPY') GO TO 2
  RETURN 1
2 I=I+1

```

```

IF(PARAM(1).EQ.'NR') GO TO 100
IF(PARAM(1).EQ.'NRDWS') GO TO 100
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'ROWS') GO TO 100
IF(PARAM(1).EQ.'NC') GO TO 110
IF(PARAM(1).EQ.'NCOLU') GO TO 110
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'COLUM') GO TO 110
IF(PARAM(1).EQ.'TAU') GO TO 120
IF(PARAM(1).EQ.'NLINES') GO TO 130
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'LINES') GO TO 130
IF(PARAM(1).EQ.'SLOPE') GO TO 140
IF(PARAM(1).EQ.'ZERO') GO TO 150
IF(PARAM(1).EQ.'DIAME') GO TO 160
IF(PARAM(1).EQ.'THICK') GO TO 170
IF(PARAM(1).EQ.'RESOL') GO TO 180
IF(PARAM(1).EQ.'POWER') GO TO 190
IF(PARAM(1).EQ.'COUNT') GO TO 200
IF(PARAM(1).EQ.'TAUC') GO TO 210
IF(PARAM(1).EQ.'INSH') GO TO 220
IF(PARAM(1).EQ.'UNFSW') GO TO 230
IF(PARAM(1).EQ.'WINDO') GO TO 2320
IF(PARAM(1).EQ.'HAZAS') GO TO 2321
IF(PARAM(1).EQ.'SUM') GO TO 2360
IF(PARAM(1).EQ.'FILTE') GO TO 2361
IF(PARAM(1).EQ.'WDBBL') GO TO 240
IF(PARAM(1).EQ.'CTOD') GO TO 250
IF(PARAM(1).EQ.'ACONT') GO TO 260
IF(PARAM(1).EQ.'ELINE') GO TO 270
IF(PARAM(1).EQ.'ALINE') GO TO 280
IF(PARAM(1).EQ.'NGAUS') GO TO 290
IF(PARAM(1).EQ.'SIMSW') GO TO 300
IF(PARAM(1).EQ.'OUTSW') GO TO 310
IF(PARAM(1).EQ.'COPY'.AND.PARAM(2).EQ.'NU') GO TO 3110
IF(PARAM(1).EQ.'COPY'.AND.PARAM(2).EQ.' ') GO TO 3111
IF(PARAM(1).EQ.'PHI1') GO TO 320
IF(PARAM(1).EQ.'THETA') GO TO 330
IF(PARAM(1).EQ.'PHI2') GO TO 340
IF(PARAM(1).EQ.'LJCHA') GO TO 350
IF(PARAM(1).EQ.'NPOTA') GO TO 360
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'PUTAS') GO TO 360
IF(PARAM(1).EQ.'NURAN') GO TO 370
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'URANI') GO TO 370
IF(PARAM(1).EQ.'INTHOR') GO TO 380
IF(PARAM(1).EQ.'N'.AND.PARAM(2).EQ.'THORI') GO TO 380
IF(PARAM(1).EQ.'ALTIT') GO TO 390
IF(PARAM(1).EQ.'ETIME') GO TO 400
IF(PARAM(1).EQ.'KUT') GO TO 410
IF(PARAM(1).EQ.'SEQUE') GO TO 420
IF(PARAM(1).EQ.'G') GO TO 430
RETURN 1

```

```

100  I=WORD(I)
      IF(I,GT,256) RETURN 1
      NR=I
      GO TO 1003
110  I=WORD(I)
      IF(I,GT,32) RETURN 1
      NC=I
      GO TO 1003
120  TAU=FWORD(I)
      RETURN 2
130  I=WORD(I)
      IF(I,GT,32) RETURN 1
      NLINES=I
      GO TO 1002
140  SLOPE=FWORD(I)
      RETURN 2
150  ZERO=FWORD(I)
      RETURN 2
160  DIAMET=FWORD(I)
      RETURN 2
170  THICKN=FWORD(I)
      RETURN 2
180  RESOLU=FWORD(I)
      RETURN 2
190  POWER=FWORD(I)
      RETURN 2
200  COUNTS=FWORD(I)
      RETURN 2
210  EMAX=FWORD(I)
      RETURN 2
220  CALL SETSW(INSW,PARAM(2),WORD(I))
      RETURN 2
230  CALL SETSW(UNFSW,PARAM(2),WORD(I))
      RETURN 2
2320 CALL SETSW(UNFSW,2,0)
      RETURN 2
2321 CALL SETSW(UNFSW,2,1)
      RETURN 2
2360 CALL SETSW(UNFSW,6,0)
      IF(I,EQ,14) RETURN 2
      IF(WORD(I).LT.1.OR.WORD(I).GT.31) RETURN 1
      IF(MOD(WORD(I),2).EQ.0) RETURN 1
      WORD(I)=WORD(I)/2
      CALL SETSW(UNFSW,7,WORD(I))
      RETURN 2
2361 CALL SETSW(UNFSW,6,1)
      RETURN 2
240  WOBBLE=FWORD(I)
      RETURN 2
250  CTOD=FWORD(I)
      RETURN 2
260  IF(J,GE,1.AND,J,LE,NCONT) GO TO 261
      IF(J,EQ,' ') GO TO 262
      RETURN 1
261  PHISIM(J)=FWORD(I)
      RETURN 2
262  DO 263 KK=1,NCONT
263  PHISIM(KK)=FWORD(I)
      RETURN 2
270  IF(J,GE,1.AND,J,LE,32) GO TO 271

```



```

        IF(J.EQ.' ') GO TO 272
        RETURN 1
271    ELINE(J)=FWORD(I)
        RETURN 2
272    DO 273 KK=1,NLINES
273    ELINE(KK)=FWORD(I)
        RETURN 2
280    IF(J.GE.1.AND.J.LE.32) GO TO 281
        IF(J.EQ.' ') GO TO 282
        RETURN 1
281    PHISIM(NCONT+J)=FWORD(I)
        RETURN 2
282    DO 283 KK=1,NLINES
283    PHISIM(NCONT+KK)=FWORD(I)
        RETURN 2
290    NGAUSS=WORD(I)
        IF(NGAUSS.LT.0) NGAUSS=0
        IF(NGAUSS.GT.2) NGAUSS=2
        RETURN 2
300    CALL SETSW(SIMSW,PARAM(2),WORD(I))
        RETURN 2
310    CALL SETSW(OUTSW,PARAM(2),WORD(I))
        IF(PARAM(2).EQ.1) GO TO 311
        GO TO 1005
311    IF(WORD(I).EQ.0) CALL DONEPL
        IF(WORD(I).EQ.0) RETURN 2
        CALL COMPRS
        GO TO 1006
3110    CALL SETSW(OUTSW,1,0)
        CALL DONEPL
        RETURN 2
3111    CALL SETSW(OUTSW,1,1)
        CALL COMPRS
        GO TO 1006
320    PHI1=FWORD(I)
        GO TO 1005
330    THETA=FWORD(I)
        GO TO 1005
340    PHI2=FWORD(I)
        GO TO 1005
350    ZERO=ZERO+(WORD(I)-LOCHAN)*SLOPE
        LOCHAN=WORD(I)
        RETURN 2
360    I=WORD(I)
        IF(I.GT.32) RETURN 1
        NPUTAS=I
        GO TO 1000
370    I=WORD(I)
        IF(I.GT.32) RETURN 1
        NURANI=I
        GO TO 1000
380    I=WORD(I)
        IF(I.GT.32) RETURN 1
        NTHORI=I
        GO TO 1000
390    ALTITU=FWORD(I)
        RETURN 2
400    ETIME=FWORD(I)
        RETURN 2
410    IF(J.GE.1.AND.J.LE.32) GO TO 411

```

```

      IF(J.EQ.' ') GO TO 412
      RETURN 1
411  KUT(J)=FWORD(I)
      RETURN 2
412  DO 413 KK=1,NLINES
413  KUT(KK)=FWORD(I)
      RETURN 2
420  IF(J.GE.1.AND.J.LE.32) GO TO 421
      IF(J.EQ.' ') GO TO 422
      RETURN 1
421  SEQUEN(J)=FWORD(I)
      RETURN 2
422  DO 423 KK=1,NLINES
423  SEQUEN(KK)=FWORD(I)
      RETURN 2
430  DO 431 KK=1,NR
431  G(KK)=NR
      RETURN 2

```

* INSW(2)=1 MEANS CHANNEL STRUCTURE HAS BEEN ALTERED

```

1000  I=NPOTAS+NURANI+NTHORI
      IF(I.EQ.0) RETURN 2
      IF(I.LE.32) GO TO 1001
      TYPE 9000
9000  FORMAT(' <<WARNING: '/
      ,      3X,'N(POTASSIUM)+N(URANIUM)+N(THORIUM) EXCEEDS 32-G')
      RETURN 2
1001  NLINES=I
1002  IF(NLINES.LE.NC) GO TO 1004
      NC=NLINES
1003  CALL SETSW(INSW,2,1)
1004  NCONT=NC-NLINES
      RETURN 2
1005  CALL SETROT
      RETURN 2
1006  CALL BGNPL(-1)
      CALL AREA2D(6.5,4.0)
      CALL HEADIN('DELPHI PLOTS',-12,-7,1)
      CALL ENOPL(-1)
      RETURN 2

      END

```

SUBROUTINE ERGORD

* ORDER LINE ENERGIES IN MONOTONICALLY INCREASING SEQUENCE

```

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
,      WOBBLER,CTDD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
,      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
,      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/SCRATC/IMIN,IMAX,ORDER,E1,E2,TEMP
LOGICAL ORDER
REAL KUT

IMAX=0
DO 3 L=1,3
IMIN=IMAX+2
IF(L.EQ.1) IMAX=NPOTAS
IF(L.EQ.2) IMAX=IMAX+NURANI
IF(L.EQ.3) IMAX=NLINES
IF(IMAX.LT.IMIN) GO TO 3

1  ORDER=.TRUE.
DO 2 I=IMIN,IMAX
IF(ELINE(I-1).LE.ELINE(I)) GO TO 2
ORDER=.FALSE.
TEMP=ELINE(I-1)
ELINE(I-1)=ELINE(I)
ELINE(I)=TEMP
TEMP=PHISIM(NCONT+I-1)
PHISIM(NCONT+I-1)=PHISIM(NCONT+I)
PHISIM(NCONT+I)=TEMP
TEMP=KUT(I-1)
KUT(I-1)=KUT(I)
KUT(I)=TEMP
2  CONTINUE
IF(.NOT.ORDER) GO TO 1

3  CONTINUE

RETURN
END

```

```

SUBROUTINE SETROT
* COMPUTE SURFACE PROJECTION MATRIX

COMMON/OPARAM/DUTSH,PHI1,THETA,PHI2,ROTATE(2,3)
DATA TORAD/.01745329/

C1=COS(TORAD*PHI1)
CT=COS(TORAD*THETA)
S1=SIN(TORAD*PHI1)
ST=SIN(TORAD*THETA)

ROTATE(1,1)=C1
ROTATE(2,1)=CT*S1
ROTATE(1,2)=-S1
ROTATE(2,2)=CT*C1
ROTATE(1,3)=0.0
ROTATE(2,3)=ST

RETURN
END

```

```

SUBROUTINE MATIN(MIN,MAX)
* READ THE RESPONSE MATRIX FROM A SEQUENTIAL DISK FILE
COMMON/SCRATC/JMIN,JMAX
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;       TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;       NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/IPARAM/INSH,DATFIL(2),RECORD,LOCHAN
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;       G(256),O(256,32),K(256,32)
REAL K
DOUBLE PRECISION A

TYPE 9000
9000  FORMAT(' <<FILE.EXT?-G')
ACCEPT 9001,A
9001  FORMAT(A10)

ENTRY MATIN2

OPEN(UNIT=23,FILE=A,ACCESS='SEQUIN',MODE='IMAGE',BUFFER COUNT=1)
JMAX=MIN+NPOTAS-1
READ(23) ((K(I,J),I=1,256),J=MIN,JMAX)
READ(23) (K(I,1),I=1,256)
JMIN=JMAX+1
JMAX=JMIN+NURANI-1
READ(23) ((K(I,J),I=1,256),J=JMIN,JMAX)
READ(23) (K(I,2),I=1,256)
JMIN=JMAX+1
JMAX=JMIN+NTHORI-1
READ(23) ((K(I,J),I=1,256),J=JMIN,JMAX)
READ(23) (K(I,3),I=1,256)
JMIN=JMAX+1
READ(23) ((K(I,J),I=1,256),J=JMIN,MAX)
CLOSE(UNIT=23)

DO 1 J=1,3
1  CALL MULS(K(LOCHAN,J),K(1,J),1.0,256-LOCHAN+1)
DO 2 J=MIN,MAX
2  CALL MULS(K(LOCHAN,J),K(1,J),KUT(J-MIN+1),256-LOCHAN+1)

RETURN
END

```

```

      SUBROUTINE NEWDAT(DWORD)
* OPEN A NEW DATA FILE IF NECESSARY

      COMMON/IPARAM/INSH,DATFIL,RECORD,LOCHAN
      INTEGER INSH,VALUE,RECORD
      DOUBLE PRECISION DWORD,DATFIL

      CALL GETSW(INSH,1,VALUE)
      IF(DWORD.EQ.'') GO TO 2
      IF(DWORD.EQ.'DATA') GO TO 2

      DATFIL=DWORD
      IF(VALUE.EQ.0) GO TO 1
      CLOSE(UNIT=23)

1      CALL SETSW(INSH,1,1)
      RECORD=1
      OPEN(UNIT=23,FILE=DATFIL,ACCESS='SEQIN',MODE='IMAGE',
        , BUFFER COUNT=1)
      RETURN

2      IF(VALUE.EQ.1) RETURN
      GO TO 1

      END

```

SUBROUTINE MOVDSK(NREC)

* MOVE THE SEQUENTIAL DATA FILE TO RECORD 'NREC'

```
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;      G(256),O(256,32),K(256,32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORT,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
DIMENSION IDATA(128)
EQUIVALENCE (DELTA(1),IDATA(1))
INTEGER INSW,RECORD
REAL K
DOUBLE PRECISION DATFIL
DATA MASK/'000000777777'/
DATA IDVSR/262144/
```

```
IF(NREC.EQ.RECORD) RETURN
IF(NREC.GT.RECORD) GO TO 1
RE=IND 23
RECORD=1
IF(NREC.EQ.1) RETURN
```

```
1  READ(23,ERR=13,END=14) IDATA
   IF(IDATA(128).GT.1000) GO TO 1
   RECORD=RECORD+1
   IF(RECORD.LT.NREC) GO TO 1
   RETURN
```

ENTRY DATIN(NREC)

* READ 'NREC' RECORDS

```
DO 2 I=1,256
2  DATA(I)=0.0
   ETIME=0.0
```

```
DO 6 N=1,NREC
3  READ(23,ERR=13,END=14) IDATA
   IF(IDATA(128).GT.1000) GO TO 3
   ETIME=ETIME+IDATA(128)/1000.
   RECORD=RECORD+1
```

```
II=0
4  DO 5 I=1,127
   II=II+1
   IDATA(128)=(IDATA(I)/IDVSR).AND.MASK
   DATA(II)=DATA(II)+IDATA(128)
   II=II+1
   IDATA(128)=IDATA(I).AND.MASK
5  DATA(II)=DATA(II)+IDATA(128)
```

```
IF(N.GT.1) GO TO 6
IF(DATA(1).LT.2644.) ALTITU=.1856*DATA(1)-38.5
IF(DATA(1).GE.2644.) ALTITU=1.075*DATA(1)-2390.
```

```
6  CONTINUE
   IF(ALTITU.LT.0.0) ALTITU=400.
```

```

7      IF(LOCHAN.EQ.1) GO TO 11
      DO 8  I=1,LOCHAN-1
8      DELTA(I)=DATA(I)
      DO 9  I=1,NR
9      DATA(I)=DATA(I+LOCHAN-1)
      DO 10 I=1,LOCHAN-1
10     DATA(I+NR)=DELTA(I)
11     COUNTS=0.0
      DO 12 I=1,NR
12     COUNTS=COUNTS+DATA(I)
      RETURN

13     TYPE 9000
9000   FORMAT(' <<AN ERROR CONDITION WAS ENCOUNTERED-G')
      GO TO 7
14     TYPE 9001
9001   FORMAT(' <<THE END-OF-FILE WAS ENCOUNTERED-G')
      GO TO 7
      END

```


THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

```

      SUBROUTINE COPY1(ITITLE,IAXIS,XMIN,X,XMAX,YMIN,Y1,YMAX,N)
* HARD COPY, SINGLE GRAPH LINE
      DIMENSION ITITLE(20),IAXIS(20),X(N),Y1(N),Y2(N)
      DATA NPLOT/=-2/
      MODE=1
      GO TO 1

      ENTRY COPY2(ITITLE,IAXIS,XMIN,X,XMAX,YMIN,Y1,Y2,YMAX,N,L)
* HARD COPY, DOUBLE GRAPH LINE
      MODE=2
1      CALL BGNPL(NPLOT)

      W=ALOG10(XMAX-XMIN)
      CHAR=AIN(T(W)
      IF(W.LT.0.0) CHAR=CHAR-1
      W=10**(W-CHAR)
      IF(W.GT.7.00) GO TO 5
      IF(W.GT.3.50) GO TO 4
      IF(W.GT.1.75) GO TO 3
      IF(W.GT.1.40) GO TO 2
      XSTP=.2
      CALL XTICKS(4)
      GO TO 61
2      XSTP=.25
      GO TO 6
3      XSTP=.50
      GO TO 6
4      XSTP=1.0
      GO TO 6
5      XSTP=2.0
6      CALL XTICKS(5)
61     XSTP=XSTP*10**CHAR
      XORIG=AIN(T(XMIN/XSTP)
      IF(XMIN.LT.0.0) XORIG=XORIG-1.0
      XORIG=XSTP*XORIG

      W=ALOG10(YMAX-YMIN)
      CHAR=AIN(T(W)
      IF(W.LT.0.0) CHAR=CHAR-1
      W=10**(W-CHAR)
      IF(W.GT.5.00) GO TO 9
      IF(W.GT.2.50) GO TO 8
      IF(W.GT.1.25) GO TO 7
      YSTP=.25
      GO TO 10
7      YSTP=.50
      GO TO 10
8      YSTP=1.0
      GO TO 10
9      YSTP=2.0
      CALL YTICKS(4)
      GO TO 11
10     CALL YTICKS(5)
11     YSTP=YSTP*10**CHAR
      IF(YMIN.GE.0.0) GO TO 12
      YORIG=AIN(T(YMIN/YSTP)-1.0

```

```

YORIG=YSTP*YORIG

12  IF(MODE.EQ.2.AND.L.EQ.1) GO TO 14
    CALL TITLE(ITITLE,100,IAXIS,100,' ',1,6.5,4.333)
    GO TO 16
14  CALL TITLE(ITITLE,100,IAXIS,100,'SQUARE-ROOTS',100,6.,4.333)
16  CALL GRAF(XORIG,XSTP,XMAX,YORIG,YSTP,YMAX)
    CALL FRAME
    CALL CURVE(X,Y1,N,0)
    IF(MODE.EQ.1) GO TO 20
    CALL SCLPIC(1/7.)
    CALL CURVE(X,Y2,N,-1)

20  CALL ENOPL(-1)
    NPLOT=NPLOT-1
    RETURN
    END

```

```

SUBROUTINE GRAFVP(ITITLE,IAXIS,ORDINA,ABSCIS,N)
* LINEAR GRAPH OF A POSITIVE VECTOR
  DIMENSION ITITLE(20),IAXIS(20),ORDINA(N),ABSCIS(N)

  CALL ERASE
  CALL TWINDO(0,1023,45,780)
  CALL GRAFP(ITITLE,IAXIS,ORDINA,ABSCIS,N)
  CALL XSCAL(ABSCIS(1),ABSCIS(N),32)

  ENTRY RESTOR
  CALL TWINDO(0,1023,0,780)
  CALL BELL
  CALL ANMODE
  ACCEPT 9000,STATUS
9000  FORMAT(A1)

  CALL NEWPAG
  RETURN
END

```

SUBROUTINE GRAFF(ITITLE,IAXIS,ORDINA,ABSCIS,N)

* LINEAR GRAPH WITH ZERO MINIMUM ORDINATE

COMMON/OPARAM/OUTSW

DIMENSION ITITLE(20),IAXIS(20),ORDINA(N),ABSCIS(N)

XMIN=ABSCIS(1)

XMAX=ABSCIS(N)

YMIN=0.

YMAX=0.

DO 1 I=1,N

IF(ORDINA(I).GT.YMAX) YMAX=ORDINA(I)

CONTINUE

YMAX=1.05*YMAX

CALL DMINDU(XMIN,XMAX,YMIN,YMAX)

CALL MOVEA(XMIN,YMAX)

CALL DRAWA(XMIN,YMIN)

CALL DRAWA(XMAX,YMIN)

CALL DRAWA(XMAX,YMAX)

CALL DRAWA(XMIN,YMAX)

CALL MOVEA(ABSCIS(1),ORDINA(1))

DO 2 I=2,N

CALL DRAWA(ABSCIS(I),ORDINA(I),3)

CALL GETSW(OUTSW,1,I)

IF(I.EQ.0) RETURN

CALL COPY1(ITITLE,IAXIS,XMIN,ABSCIS,XMAX,YMIN,ORDINA,YMAX,N)

RETURN

END

```

SUBROUTINE GRAFMX(ITITLE,IAXIS,A,NR,NC,ISCAL)

* GRAPH MATRIX A BY COLUMNS.
* IF ABS(ISCAL)=1, ABSCISSA=EDATA.
* IF ABS(ISCAL)=2, ABSCISSA=EPI.
* IF ABS(ISCAL)=3, ABSCISSA=COLUMN INDEX.

    DIMENSION ITITLE(20),IAXIS(20),A(256,NC)
    COMMON/SCRATC/ABSCIS(256)
    COMMON/SOURCE/EPI(256)
    COMMON/DETECT/EDATA(256)
    COMMON/OPARAM/OUTSW

    CALL GETSW(OUTSW,1,MODE)
    XMIN=1.E30
    YMIN=XMIN
    XMAX=-XMIN
    YMAX=XMAX

    DO 1 I=1,NR
    IF(ABS(ISCAL).EQ.1) ABSCIS(I)=EDATA(I)
    IF(ABS(ISCAL).EQ.2) ABSCIS(I)=EPI(I)
    IF(ABS(ISCAL).EQ.3) ABSCIS(I)=I
1    CONTINUE

    DO 2 I=1,NR
    IF(ABSCIS(I).LT.XMIN) XMIN=ABSCIS(I)
    IF(ABSCIS(I).GT.XMAX) XMAX=ABSCIS(I)
    DO 2 J=1,NC
    IF(A(I,J).LT.YMIN) YMIN=A(I,J)
    IF(A(I,J).GT.YMAX) YMAX=A(I,J)
2    CONTINUE
    YMAX=1.01*YMAX

    CALL DWINDD(XMIN,XMAX,YMIN,YMAX)
    DO 4 J=1,NC
    CALL ERASE
    CALL MOVEA(XMIN,YMAX)
    CALL DRAWA(XMIN,YMIN)
    CALL DRAWA(XMAX,YMIN)
    CALL DRAWA(XMAX,YMAX)
    CALL DRAWA(XMIN,YMAX)

    CALL MOVEA(ABSCIS(1),A(1,J))
    IF(MODE.EQ.1) CALL COPY1
    , (ITITLE,IAXIS,XMIN,ABSCIS,XMAX,YMIN,A(1,J),YMAX,NR)
    CALL BELL
    DO 3 I=2,NR
3    CALL DRAWA(ABSCIS(I),A(I,J))
4    CALL BELL

    CALL BELL
    CALL ANMODE
    ACCEPT 9000,STATUS
9000  FORMAT(A1)
    CALL NEWPAG
    RETURN
    END

```


SUBROUTINE SURFAC(MATRIX,NR,NC,ISCAL)

* DISPLAY 'MATRIX' AS A THREE-DIMENSIONAL SURFACE

```

REAL MATRIX(256,NC)
COMMON/UPARAM/PARAM(8),NCONT,NLINES
COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/SOURCE/EPHI(256)
COMMON/DETECT/EDATA(256)
COMMON/SCRATC/ALPHA,BETA,GAMMA,IDEL,JDEL,VISIBL,RELHT(3,2),
;      MARK(2),LIMIT(2),IVERTX,HORIZN(248,2,2),JOINT(3,2),
;      X(3),Y(3),DX,DY
INTEGER VISIBL

```

* SUPPRESS UNDER SIDE UNLESS OUTSW(2)=1

CALL GETSW(OUTSW,2,VISIBL)

```

IDEL=(NR-1)/64+1
JDEL=(NC-1)/64+1
IMAX=((NR-1)/IDEL)*IDEL+1
JMAX=((NC-1)/JDEL)*JDEL+1

```

```

ALPHA=EDATA(NR)-EDATA(1)
ZMIN=1.E10
ZMAX=-ZMIN
DO 1 I=1,IMAX,IDEL
DO 1 J=1,JMAX,JDEL
IF(MATRIX(I,J).LT.ZMIN) ZMIN=MATRIX(I,J)
IF(MATRIX(I,J).GT.ZMAX) ZMAX=MATRIX(I,J)
CONTINUE
GAMMA=ALPHA/(ZMAX-ZMIN)
ALPHA=ALPHA/(NC-1)
BETA=EDATA(1)-ALPHA
GAMMA=GAMMA*ROTATE(2,3)

```

```

IF(ISCAL.EQ.1) XMIN=ROTATE(1,1)*EDATA(1)
;      +ROTATE(1,2)*EPHI(NCONT+JMAX)
IF(ISCAL.EQ.2) XMIN=ROTATE(1,1)*EDATA(1)
;      +ROTATE(1,2)*(ALPHA*JMAX+BETA)
IF(ISCAL.EQ.1) XMAX=ROTATE(1,1)*EDATA(IMAX)
;      +ROTATE(1,2)*EPHI(NCONT+1)
IF(ISCAL.EQ.2) XMAX=ROTATE(1,1)*EDATA(IMAX)
;      +ROTATE(1,2)*(ALPHA+BETA)

```

```

YMIN=+1.E10
YMAX=-1.E10
DO 2 I=1,IMAX,IDEL
DO 2 J=1,JMAX,JDEL
IF(ISCAL.EQ.1) Y(1)=EPHI(NCONT+J)
IF(ISCAL.EQ.2) Y(1)=ALPHA*J+BETA
Y(1)=ROTATE(2,1)*EDATA(I)+ROTATE(2,2)*Y(1)+GAMMA*MATRIX(I,J)
IF(VISIBL.EQ.1.AND.Y(1).LT.YMIN) YMIN=Y(1)
IF(Y(1).GT.YMAX) YMAX=Y(1)
IF(I.GT.1.AND.J.GT.1) GO TO 2
IF(Y(1).LT.YMIN) YMIN=Y(1)
CONTINUE

```

```

DX=.5*(XMAX-XMIN)
DY=.5*(YMAX-YMIN)
IF(780.*DX.GT.1024.*DY) GO TO 3
T=.5*(XMIN+XMAX)
DX=1024.*DY/780.
XMIN=T-DX
XMAX=T+DX
GO TO 4
3 T=.5*(YMIN+YMAX)
DY=780.*DX/1024.
YMIN=T-DY
YMAX=T+DY

4 CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
CALL TWINDO(0,1023,0,779)
CALL ERASE
DX=(XMAX-XMIN)/1024.
DY=(YMAX-YMIN)/780.

IF(ISCAL.EQ.1) Y(1)=EPI(NCONT+JMAX)
IF(ISCAL.EQ.2) Y(1)=ALPHA*JMAX+BETA
X(1)=ROTATE(1,1)*EDATA(1)+ROTATE(1,2)*Y(1)
Y(1)=ROTATE(2,1)*EDATA(1)+ROTATE(2,2)*Y(1)+GAMMA*MATRIX(1,JMAX)
DO 9 I=1,2
LIMIT(I)=1
HORIZN(1,1,I)=X(1)-2.*DX
9 HORIZN(1,2,I)=Y(1)
CALL MOVEA(X(1),Y(1))
DO 10 JJ=JMAX,1,-JOEL
J=JJ
10 CALL OUTLIN(1,J,MATRIX,NR,NC,ISCAL)
DO 11 II=IDEL+1,IMAX,IDEL
I=II
11 CALL OUTLIN(I,1,MATRIX,NR,NC,ISCAL)
DO 12 I=1,2
LIMIT(I)=LIMIT(I)+1
HORIZN(LIMIT(I),1,I)=X(1)+2.*DX
12 HORIZN(LIMIT(I),2,I)=Y(1)

IF(VISIBL.EQ.1) GO TO 14
YMIN=YMIN-2.*DY
DO 13 I=1,LIMIT(1)
13 HORIZN(I,2,1)=YMIN

```

```

14      DO 19 JJ=JDEL+1,JMAX,JDEL
        J=JJ
        MARK(1)=1
        MARK(2)=1
        IVERTX=1
        CALL PROJEX(1,J,MATRIX,NR,NC,ISCAL)

16      DO 17 II=IDEL+1,IMAX,IDEL
        I=II
17      CALL CORNER(I,J,MATRIX,NR,NC,ISCAL)

19      CONTINUE

        CALL BELL
        CALL ANMODE
        ACCEPT 9000,STATUS
9000    FORMAT(A1)
        CALL NEWPAGE

        RETURN
        END

```

```

SUBROUTINE CORNER(I,J,MATRIX,NR,NC,ISCAL)
* DRAW ONE SURFACE CORNER FROM (I-IDEL,J) TO (I,J) TO (I,J-JDEL)

REAL MATRIX(256,NC)
COMMON/UPARAM/PARAM(8),NCONT,NLINES
COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/SOURCE/EPHI(256)
COMMON/DETECT/EDATA(256)
COMMON/SCRATC/ALPHA,BETA,GAMMA,IDEL,JDEL,VISIBL,RELHT(3,2),
;      MARK(2),LIMIT(2),IVERTX,HORIZN(248,2,2),JOINT(3,2),
;      X(3),Y(3),DX,DY
INTEGER VISIBL

DO 2 K=1,2
RELHT(2,K)=RELHT(1,K)
JOINT(2,K)=JOINT(1,K)
2 X(3*K-1)=X(3*K-2)
IVERTX=1
CALL PROJEX(I,J,MATRIX,NR,NC,ISCAL)

VISIBL=0
DO 20 IVERTX=2,3
IF(IVERTX.EQ.3) CALL PROJEX(I,J-JDEL,MATRIX,NR,NC,ISCAL)
DO 18 K=2,1,-1
IF(RELHT(1,K).LT.0.0) GO TO 18
IF(RELHT(IVERTX,K).LE.-DY) GO TO 12

IF(IVERTX.EQ.3) GO TO 6
II=MARK(K)
4 IF(HORIZN(II,1,K).LE.X(2)) GO TO 10
W=(HORIZN(II,1,K)-X(1))/(X(2)-X(1))
IF((1.-W)*Y(1)+W*Y(2).LT.HORIZN(II,2,K)) GO TO 12
II=II-1 ; GO TO 4
6 II=ABS(JOINT(1,K))+1
8 IF(HORIZN(II,1,K).GE.X(3)) GO TO 10
W=(HORIZN(II,1,K)-X(1))/(X(3)-X(1))
IF((1.-W)*Y(1)+W*Y(3).LT.HORIZN(II,2,K)) GO TO 12
II=II+1 ; GO TO 8

10 CALL MOVEA(X(IVERTX),Y(IVERTX))      ISEGMENT IS VISIBLE
CALL DRAWA(X(1),Y(1))
VISIBL=K ; GO TO 20

12 DELX=DX                                ISEGMENT CROSSES HORIZON
IF(IVERTX.EQ.2) DELX=-DELX
DELY=DELX*(Y(IVERTX)-Y(1))/(X(IVERTX)-X(1))
W=X(IVERTX) ; X(IVERTX)=X(1)
Y(IVERTX)=Y(1)
14 X(IVERTX)=X(IVERTX)+DELX
Y(IVERTX)=Y(IVERTX)+DELY
CALL UPDATE
IF(ABS(X(IVERTX)-W).LT.DX) GO TO 16
IF(RELHT(IVERTX,K).GT.DY) GO TO 14
16 IF(K.EQ.1) Y(IVERTX)=Y(IVERTX)+RELHT(IVERTX,K)
IF(K.EQ.2) Y(IVERTX)=Y(IVERTX)-RELHT(IVERTX,K)
GO TO 10

18 CONTINUE                                ISEGMENT IS NOT VISIBLE
20 CONTINUE

```

```

* MARK(1): LOWER HORIZON POINTER
* MARK(2): UPPER HORIZON POINTER
* LIMIT(1): LOWER HORIZON LENGTH
* LIMIT(2): UPPER HORIZON LENGTH
* IF VISIBL=0: (I,J) IS OBSCURED
* IF VISIBL=1: (I,J) IS VISIBLE BELOW THE LOWER HORIZON
* IF VISIBL=2: (I,J) IS VISIBLE ABOVE THE UPPER HORIZON

      IF(VISIBL.EQ.0) RETURN
      IGAP=ABS(JOINT(2,VISIBL))-ABS(JOINT(3,VISIBL))+2
      IF(JOINT(3,VISIBL).GT.0) GO TO 30
      JOINT(3,VISIBL)=JOINT(3,VISIBL)-1
30     IF(JOINT(2,VISIBL).GT.0) GO TO 31
      IGAP=IGAP+1
31     IF(IGAP.EQ.0) GO TO 37
      IF(IGAP.LT.0) GO TO 35
33     LIMIT(VISIBL)=LIMIT(VISIBL)+IGAP      IMAKE ROOM
      DO 34 II=LIMIT(VISIBL),ABS(JOINT(3,VISIBL))+IGAP,-1
      HORIZN(II,1,VISIBL)=HORIZN(II-IGAP,1,VISIBL)
34     HORIZN(II,2,VISIBL)=HORIZN(II-IGAP,2,VISIBL)
      GO TO 37
35     LIMIT(VISIBL)=LIMIT(VISIBL)+IGAP      ICOMPRESS
      DO 36 II=ABS(JOINT(3,VISIBL))+IGAP,LIMIT(VISIBL)
      HORIZN(II,1,VISIBL)=HORIZN(II-IGAP,1,VISIBL)
36     HORIZN(II,2,VISIBL)=HORIZN(II-IGAP,2,VISIBL)
37     K=ABS(JOINT(2,VISIBL))+1
      IF(JOINT(2,VISIBL).GT.0) GO TO 38
      HORIZN(K,1,VISIBL)=X(2)
      HORIZN(K,2,VISIBL)=Y(2)
      K=K+1
38     HORIZN(K,1,VISIBL)=X(1)
      HORIZN(K,2,VISIBL)=Y(1)
      REL-IT(1,VISIBL)=0.0
      JOINT(1,VISIBL)=K
      IF(JOINT(3,VISIBL).GT.0) RETURN
      K=K+1
      HORIZN(K,1,VISIBL)=X(3)
      HORIZN(K,2,VISIBL)=Y(3)
      RETURN

```

```

ENTRY OUTLIN(I,J,MATRIX,NR,NC,ISCAL)

40  GO TO (41,43) ISCAL
41  X(1)=ROTATE(1,1)*EDATA(I)+ROTATE(1,2)*EPI(NCONT+J)
    Y(1)=ROTATE(2,1)*EDATA(I)+ROTATE(2,2)*EPI(NCONT+J)
    ;      +GAMMA*MATRIX(I,J)
    GO TO 45
43  X(1)=ROTATE(1,1)*EDATA(I)+ROTATE(1,2)*(ALPHA*J+BETA)
    Y(1)=ROTATE(2,1)*EDATA(I)+ROTATE(2,2)*(ALPHA*J+BETA)
    ;      +GAMMA*MATRIX(I,J)
45  CALL DRAW(X(1),Y(1))
48  DO 49  K=1,2
    IF(X(1)-HORIZN(LIMIT(K),1,K).LT.DX) GO TO 49
    LIMIT(K)=LIMIT(K)+1
    HORIZN(LIMIT(K),1,K)=X(1)
49  HORIZN(LIMIT(K),2,K)=Y(1)

RETURN
END

```

```

      SUBROUTINE PROJEX(I,J,MATRIX,NR,NC,ISCAL)

* PROJECT A POINT ONTO THE VIEWING PLANE
* IF ISCAL=1, Y=EPhi(NCONT+J)
* IF ISCAL=2, Y=ALPHA*J+BETA

      REAL MATRIX(256,NC)
      COMMON/UPARAM/PARAM(8),NCONT,NLINES
      COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
      COMMON/SOURCE/EPHI(256)
      COMMON/DETECT/EDATA(256)
      COMMON/SCRATC/ALPHA,BETA,GAMMA,IDEI,JDEI,VISIBL,RELHT(3,2),
;         MARK(2),LIMIT(2),IVERTX,HORIZN(248,2,2),JOINT(3,2),
;         X(3),Y(3),DX,DY
      INTEGER VISIBL

      IF(ISCAL.EQ.1) Y(IVERTX)=EPHI(NCONT+J)
      IF(ISCAL.EQ.2) Y(IVERTX)=ALPHA*J+BETA

      X(IVERTX)=ROTATE(1,1)*EDATA(I)+ROTATE(1,2)*Y(IVERTX)
      Y(IVERTX)=ROTATE(2,1)*EDATA(I)+ROTATE(2,2)*Y(IVERTX)
;         +GAMMA*MATRIX(I,J)

* DETERMINE HEIGHT RELATIVE TO THE HORIZON

      ENTRY UPDATE

      DO 5 K=1,2

1      IF(HORIZN(MARK(K),1,K).GT.X(IVERTX)) GO TO 2
      MARK(K)=MARK(K)+1
      GO TO 1
2      MARK(K)=MARK(K)-1
      IF(HORIZN(MARK(K),1,K).LE.X(IVERTX)) GO TO 3
      GO TO 2

3      IF(X(IVERTX)-HORIZN(MARK(K),1,K).LT.DX) GO TO 4
      JOINT(IVERTX,K)=-MARK(K)
      W=(X(IVERTX)-HORIZN(MARK(K),1,K))/
;         (HORIZN(MARK(K)+1,1,K)-HORIZN(MARK(K),1,K))
      RELHT(IVERTX,K)=Y(IVERTX)
;         -(1.-W)*HORIZN(MARK(K),2,K)
;         -W*HORIZN(MARK(K)+1,2,K)
      IF(K.EQ.1) RELHT(IVERTX,K)=-RELHT(IVERTX,K)
      GO TO 5

4      JOINT(IVERTX,K)=MARK(K)
      RELHT(IVERTX,K)=Y(IVERTX)-HORIZN(MARK(K),2,K)
      IF(K.EQ.1) RELHT(IVERTX,K)=-RELHT(IVERTX,K)

5      CONTINUE

      RETURN
      END

```

```

SUBROUTINE GRAFPR(ITITLE, IAXIS, X, YDOT, YLINE, N, L)
* DISPLAY AN ORDINATE PAIR, DIMENSION N
* IF L=1, ORDINATE SCALE IS SQUARE ROOT

DIMENSION ITITLE(20), IAXIS(20), X(N), YDOT(N), YLINE(N)
COMMON/OPARAM/OUTSW
COMMON/SCRATC/TEMP(512), DOT(256), LINE(256)
REAL LINE

XMIN=X(1)
XMAX=X(N)
YMIN=1.E10
YMAX=-YMIN
DO 3 I=1, N
IF(L.EQ.1) GO TO 1
DOT(I)=YDOT(I)
LINE(I)=YLINE(I)
GO TO 2
1 DOT(I)=SQRT(YDOT(I))
LINE(I)=SQRT(YLINE(I))
2 IF(DOT(I).GT.YMAX) YMAX=DOT(I)
IF(LINE(I).GT.YMAX) YMAX=LINE(I)
IF(DOT(I).LT.YMIN) YMIN=DOT(I)
IF(LINE(I).LT.YMIN) YMIN=LINE(I)
3 CONTINUE
IF(YMIN.LT.0.0) GO TO 4
YMIN=0.0
YMAX=1.01*YMAX
GO TO 5
4 CENTER=.5*(YMIN+YMAX)
DY=.51*(YMAX-YMIN)
YMIN=CENTER-DY
YMAX=CENTER+DY
5 CALL DWINDO(XMIN, XMAX, YMIN, YMAX)
CALL TWINDO(0, 1023, 45, 780)
CALL ERASE
CALL MOVEA(XMIN, YMAX)
CALL DRAWA(XMIN, YMIN)
CALL DRAWA(XMAX, YMIN)
CALL DRAWA(XMAX, YMAX)
CALL DRAWA(XMIN, YMAX)

CALL BELL
DO 6 I=1, N
6 CALL POINTA(X(I), DOT(I))
CALL MOVEA(X(1), LINE(1))
DO 7 I=2, N
7 CALL DRAWA(X(I), LINE(I))
CALL XSCAL(XMIN, XMAX, 32)
CALL TWINDO(0, 1023, 0, 780)
CALL GETSW(OUTSW, 1, I)
IF(I.EQ.1) CALL COPY2
; (ITITLE, IAXIS, XMIN, X, XMAX, YMIN, LINE, DOT, YMAX, N, L)
CALL BELL

CALL ANMODE
ACCEPT 9000, STATUS
9000 FORMAT(A1)

```


CALL NEWPAG
RETURN
END

```

SUBROUTINE DISRES(IWORD,*)
* DISPLAY CONTINUUM AND DISCRETE RESIDUALS

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
; NOBBLE,CTOD,NGAUSS,XNORM
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
; TAU,NCONT,NLINES,COUNTS,EMAX
COMMON/IPARAM/INSH,DATFIL,RECORD
COMMON/SOURCE/EPHI(256),PHI(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
; G(256),O(256,32),K(256,32)
COMMON/SCRATC/DATAC(256),DEXPC(256),W
INTEGER VALUE,SIMSW,UNFSW,INSH,RECORD
DOUBLE PRECISION DATFIL
REAL K

IF(IWORD.EQ.'DISCR') GO TO 1
IF(IWORD.EQ.'CONTI') GO TO 2
RETURN 1

1 CALL GETSW(UNFSW,3,1)
IF(I.EQ.0) CALL MAZASC
CALL PROJFD(DEXP,K,PHI,NR,NCONT)
GO TO 3

2 CALL PROJFD(DEXP,K(1,NCONT+1),PHI(NCONT+1),NR,NLINES)

3 CALL SUBV(DATA,DEXP,DELTA,NR)
CALL MULS(DATAC,DATAC,0.0,NR)
IF(IWORD.EQ.'DISCR') CALL GRAFPR
; ('DISCRETE RESIDUALS','ENERGY(MEV)S',EDATA,DELTA,DATAC,NR,0)
IF(IWORD.EQ.'CONTI') CALL GRAFPR
; ('CONTINUUM RESIDUALS','ENERGY(MEV)S',EDATA,DELTA,DATAC,NR,0)

RETURN
END

```

SUBROUTINE DISFIT(IWORD,*)

* DISPLAY CONTINUUM AND DISCRETE FITS

```

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLE,CTOD,NGAUSS,XNORM
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI
COMMON/IPARAM/INSW,DATFIL,RECORD
COMMON/SOURCE/EPHI(256),PHI(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;      G(256),D(256,32),K(256,32)
COMMON/SCRATC/DATAC(256),DEXPC(256),N
INTEGER VALUE,SIMSW,UNFSW,INSW,RECORD
DOUBLE PRECISION DATFIL
REAL K

IF(IWORD.EQ.'CONTI') GO TO 1
IF(IWORD.EQ.'DISCR') GO TO 2
IF(IWORD.EQ.'POTAS') GO TO 3
IF(IWORD.EQ.'URANI') GO TO 4
IF(IWORD.EQ.'THORI') GO TO 5
RETURN 1

1  CALL GETSW(UNFSW,3,I)
   IF(I.EQ.0) CALL MAZASC
   CALL PROJFD(DEXP,K(1,NCONT+1),PHI(NCONT+1),NR,NCONT)
   CALL GRAFPR('CONTINUUM FITS','ENERGY(MEV)S',EDATA,DATA,DEXP,NR,1)
   RETURN

2  CALL PROJFD(DEXP,K(1,NCONT+1),PHI(NCONT+1),NR,NLINES)
   CALL GRAFPR('DISCRETE FITS','ENERGY(MEV)S',EDATA,DATA,DEXP,NR,1)
   RETURN

3  CALL PROJFD(DEXP,K(1,NCONT+1),PHI(NCONT+1),NR,NPOTAS)
   CALL GRAFPR('POTASSIUM FITS','ENERGYS',EDATA,DATA,DEXP,NR,1)
   RETURN

4  I=NCONT+NPOTAS+1
   CALL PROJFD(DEXP,K(1,I),PHI(I),NR,NURANI)
   CALL GRAFPR('URANIUM FITS','ENERGYS',EDATA,DATA,DEXP,NR,1)
   RETURN

5  I=NCONT+NPOTAS+NURANI+1
   CALL PROJFD(DEXP,K(1,I),PHI(I),NR,NTHORI)
   CALL GRAFPR('THORIUM FITS','ENERGY(MEV)S',EDATA,DATA,DEXP,NR,1)
   RETURN

END

```

SUBROUTINE DISPAR(NU)

* DISPLAY PARAMETERS

NU	DISPLAY
1	UNFOLDING PARAMETERS
2	SIMULATION PARAMETERS
3	TRANSFER PARAMETERS
4	DISPLAY PARAMETERS
5	SWITCH POSITIONS

* IF NU HAS ANY OTHER VALUE, ALL ARE DISPLAYED

```
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAUD,NCONT,NLINES,COUNTS,TAUC,NSEC,NPOTAS,NURANI,
;      NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)
COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
COMMON/SCRATC/M1(8),M2(8),M3(8),M4(8),INDEX(35),IMAX,JUMP
INTEGER SIMSW,UNFSW,OUTSW,INSW,RECORD
DOUBLE PRECISION DATFIL
REAL KUT
```

```
IF(NU.LT.2.OR.NU.GT.5) GO TO 10
GO TO (10,20,30,40,50) NU
```

* UNFOLDING PARAMETERS

10 CALL NEWPAG

```
CALL DRAWX(1)
CALL DRAWX(3)
CALL DRAWX(6)
CALL DRAWY(3,1,6)
CALL DRAWY(14,3,6)
CALL DRAWY(25,3,6)
CALL DRAWY(35,3,6)
CALL DRAWY(47,3,6)
CALL DRAWY(59,3,6)
CALL DRAWY(71,1,6)
```

```
CALL DRAWX(7)
CALL DRAWX(11)
CALL DRAWX(15)
CALL DRAWY(3,7,15)
CALL DRAWY(18,7,15)
CALL DRAWY(31,7,15)
CALL DRAWY(45,11,15)
CALL DRAWY(58,11,15)
CALL DRAWY(71,7,15)
```

```
CALL DRAWX(16)
CALL DRAWX(18)
IMAX=NLINES
IF(NPOTAS.GT.0) IMAX=IMAX+1
```

```

IF(NURANI.GT.0) IMAX=IMAX+1
IF(NTHORI.GT.0) IMAX=IMAX+1
JUMP=(IMAX+3)/4
I=JUMP+19
CALL DRAWX(I)
CALL DRAWY(3,16,I)
CALL DRAWY(20,16,I)
CALL DRAWY(37,16,I)
CALL DRAWY(54,16,I)
CALL DRAWY(71,16,I)

CALL DRAWX(I+2)
II=JUMP+(I+3)
CALL DRAWX(II)
CALL DRAWY(3,I,II)
CALL DRAWY(20,I,II)
CALL DRAWY(37,I,II)
CALL DRAWY(54,I,II)
CALL DRAWY(71,I,II)

CALL HOME
TYPE 9010
9010 FORMAT(/6X,'DIMENSION: N(.....)')
TYPE 9011
9011 FORMAT(/7X,'DATA',5X,'SPECTRUM',4X,'LINES',4X,
; 'POTASSIUM',4X,'URANIUM',5X,'THORIUM')
TYPE 9012,NR,NC,NLINES,NPOTAS,NURANI,NTHORI
9012 FORMAT(8X,I3,7X,I3,2(8X,I3),2(9X,I3)/)

TYPE 9013
9013 FORMAT(/9X,'SLOPE',9X,'TAUC',5X,
; 'FWHM=RESOLUTION=E*(.6616/E)**POWER')
TYPE 9014,SLOPE,TAUC,PESOLU,POWER
9014 FORMAT(6X,1PE10.4,5X,1PE8.2,7X,
; '=(',1PE8.2,')*E*(.6616/E)**(',1PE8.2,')'/
; 5X,'MEVS/CHANNEL',7X,'MEV',10X,'<MEV>')
TYPE 9015
9015 FORMAT(/9X,'ZERO',10X,'TAUD',8X,'COUNTS',
; 7X,'ALTITUDE',6X,'ETIME')
TYPE 9016,ZERO,TAUD,COUNTS,ALTITU,ETIME
9016 FORMAT(5X,1PE11.4,2(5X,1PE8.2),1X,2(5X,1PE8.2)/
; 10X,'MEV',37X,'FEET',7X,'SECONDS'//)

TYPE 9017
9017 FORMAT(3X,4(3X,'I ELINE(I)',2X))
II=1
DO 14 I=1,IMAX
IF(I.EQ.1.AND.NPOTAS.GT.0) GO TO 11
IF(I.EQ.(NPOTAS+2).AND.NURANI.GT.0) GO TO 12
IF(I.EQ.(NPOTAS+NURANI+3).AND.NTHORI.GT.0) GO TO 13
INDEX(I)=II
II=II+1
GO TO 14
11 INDEX(I)='POTAS'
GO TO 14
12 INDEX(I)='URANI'
GO TO 14
13 INDEX(I)='THORI'
14 CONTINUE

```

```

9018  FORMAT(/3X,S)
9101  FORMAT(1H+,2H  , 'POTASSIUM:',5X,S)
9102  FORMAT(1H+,2H  , 'URANIUM:',7X,S)
9103  FORMAT(1H+,2H  , 'THORIUM:',7X,S)
9104  FORMAT(1H+,2H  , I2,3X,1PE8.2,2X,S)

      II=1
      DO 15  I=1,IMAX
      IF(II,LE,JUMP) TYPE 9018
      IF(INDEX(II).EQ.'POTAS') TYPE 9101
      IF(INDEX(II).EQ.'URANI') TYPE 9102
      IF(INDEX(II).EQ.'THORI') TYPE 9103
      IF(INDEX(II).GT.0) TYPE 9104,INDEX(II),ELINE(INDEX(II))
      II=II+JUMP
      IF(II,GT,IMAX) II=II-4*JUMP+1
      IF(II,LT,1) II=II+JUMP
15    CONTINUE

      TYPE 9019
9019  FORMAT(/3X,4(3X,'I      KUT(I)',2X))

      II=1
      DO 16  I=1,IMAX
      IF(II,LE,JUMP) TYPE 9018
      IF(INDEX(II).EQ.'POTAS') TYPE 9101
      IF(INDEX(II).EQ.'URANI') TYPE 9102
      IF(INDEX(II).EQ.'THORI') TYPE 9103
      IF(INDEX(II).GT.0) TYPE 9104,INDEX(II),KUT(INDEX(II))
      II=II+JUMP
      IF(II,GT,IMAX) II=II-4*JUMP+1
      IF(II,LT,1) II=II+JUMP
16    CONTINUE

      ACCEPT 9200,STATUS
9200  FORMAT(A1)
      CALL NEWPAG
      IF(NU,NE,1) GO TO 20
      RETURN

* SIMULATION PARAMETERS

20    TYPE 9020,DIAMET,THICKN,WOBBLE,CTOD,NGAUSS
9020  FORMAT(' <<DIAMETER=',F5.2,' INCHES, THICKNESS=',
;          F5.2,' INCHES'/
;          3X,'NGAINE WOBBLE=',F5.2,' PER CENT'/
;          3X,'CTOD #CONTINUUM/DISCRETE COUNT RATIO=',F5.2/
;          3X,'RESOLUTION COMPONENTS, NGAUSS=',I1)

      DO 21  I=1,NCONT,2
      II=I+1
      TYPE 9021,I,PHISIM(I),II,PHISIM(II)
21    TYPE 9021,I,PHISIM(I),II,PHISIM(II)
9021  FORMAT(3X,'ACONTINUUM(',I2,')=',1PE10.3,
;          ', ACONTINUUM(',I2,')=',1PE10.3)

      DO 22  I=1,NLINES
      IF(ELINE(I).LT.1.E-10.AND.PHISIM(NCONT+I).LT.1.E-10)
      GO TO 23
22    TYPE 9025,I,ELINE(I),I,PHISIM(NCONT+I)
9025  FORMAT(3X,'ELINE(',I2,')=',0PF6.3,' , ALINE(',I2,')=',1PE8.2)
23    IF(NU,EQ,2) RETURN

```

* DATA TRANSFER PARAMETERS

```

30      TYPE 9030,RECORD,DATFIL,LOCHAN
9030    FORMAT(' <<RECORD',I4,' IN FILE ',A10,' ')/
      ;      3X,'LOCHANNEL=',I2)
      IF(NU.EQ.3) RETURN

```

* DISPLAY PARAMETERS

```

40      TYPE 9040,PHI1,THETA
9040    FORMAT(' <<PHI1=',F6.2,' , THETA=',F6.2,' DEGREES')
      IF(NU.EQ.4) RETURN

```

* SWITCH POSITIONS

```

50      DO 51 I=1,8
      N=I
      CALL GETSW(UNFSW,N,M1(N))
      CALL GETSW(SIMSW,N,M2(N))
      CALL GETSW(INSW,N,M3(N))
51      CALL GETSW(OUTSW,N,M4(N))

      TYPE 9050,M1,M2,M3,M4
9050    FORMAT(' <<SWITCH POSITIONS: ' /
      ;      3X,'UNFSW=(',7(I2,' '),I2,')' /
      ;      3X,'SIMSW=(',7(I2,' '),I2,')' /
      ;      3X,'INSW=(',7(I2,' '),I2,')' /
      ;      3X,'OUTSW=(',7(I2,' '),I2,')' )
      RETURN
      END

```

```

SUBROUTINE DRAWX(I)
* DRAW A HORIZONTAL LINE FROM CHARACTER POSITION (3,I)
* TO CHARACTER POSITION (71,I)
* WHERE I=1,34 FROM THE TOP LINE DOWN

    IY=22*(35-I)+28
    IX=14*3-10
    CALL MOVABS(IX,IY)
    IX=14*71-10
    CALL DRWABS(IX,IY)
    GO TO 1

    ENTRY DRAWY(I,ILO,IHI)
* DRAW A VERTICAL LINE AT HORIZONTAL CHARACTER POSITION I
* FROM LINE ILO TO LINE IHI

    IX=14*I-10
    IY=22*(35-IHI)+28
    CALL MOVABS(IX,IY)
    IY=22*(35-ILO)+28
    CALL DRWABS(IX,IY)

1    CALL ANMODE
    RETURN
    END

```


SUBROUTINE DISVOC(NROOT,PEND,PART,NPEND,NPART,NREC)
 * DISPLAY VOCABULARY

```

      INTEGER PEND(NPEND),PART(NPART)

      TYPE 9000
      FORMAT(' <<',S)
      GO TO (1,2,3,4,5) NREC
      1   TYPE 9001
      9001 FORMAT(1H+,'VERBS:',S)
      GO TO 6
      2   TYPE 9002
      9002 FORMAT(1H+,'NOUNS:',S)
      GO TO 6
      3   TYPE 9003
      9003 FORMAT(1H+,'ADVERBS:',S)
      GO TO 6
      4   TYPE 9004
      9004 FORMAT(1H+,'ADJECTIVES:',S)
      GO TO 6
      5   TYPE 9005
      9005 FORMAT(1H+,'PREPOSITIONS:',S)

      6   IMAX=PEND(NROOT)
      IROOT=1
      DO 8 I=1,IMAX
      IF(MOD(I,5).EQ.1) TYPE 9006
      9006 FORMAT(/3X,S)
      IF(I.EQ.IMAX) GO TO 9
      IF(I.EQ.PEND(IROOT)) GO TO 7
      TYPE 9007,PART(I)
      9007 FORMAT(1H+,A5,',',S)
      GO TO 8
      7   TYPE 9008,PART(I)
      9008 FORMAT(1H+,A5,',',S)
      IROOT=IROOT+1
      8   CONTINUE
      9   TYPE 9009,PART(I)
      9009 FORMAT(1H+,A5)

      RETURN
      END
  
```

```

SUBROUTINE DISPEC(NPLOT)
* IF NPLOT=1, PLOT DISCRETE SPECTRUM
* IF NPLOT=2, PLOT SPECTRAL ERROR

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;      WOBBL,E,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;      NTHORI

COMMON/IPARAM/INSW,DATFIL,RECORD
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256)
INTEGER VALUE,SIMSW,UNFSW,INSW,RECORD
DOUBLE PRECISION DATFIL

CALL GETSW(SIMSW,1,VALUE)
IF(VALUE.EQ.1) CALL MULS(PHI(NCONT+1),PHI(NCONT+1),XNORM,NLINES)
XMIN=EDATA(1)
XMAX=EDATA(NR)
YMIN=0.
IF(NPLOT.EQ.2) GO TO 2
J=NCONT
YMAX=0.0
DO 1 I=1,NLINES
J=J+1
IF(PHI(J).GT.YMAX) YMAX=PHI(J)
1 CONTINUE
YMAX=1.03*YMAX
GO TO 3

2 IF(VALUE.NE.0) RETURN
YMAX=1.0
IMIN=NCONT+1
CALL MULS(PHISIM(NCONT+1),ERROR(IMIN),1.0,NLINES)
RMS=0.0
DO 20 I=IMIN,NC
ERROR(I)=(PHI(I)-ERROR(I))/ERROR(I)
20 RMS=RMS+ERROR(I)**2
RMS=100.*SQRT(RMS/NLINES)

3 CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
CALL TWINDO(0,1023,429,780)
CALL ERASE
CALL MOVEA(XMIN,YMAX)
CALL DRAWA(XMIN,YMIN)
CALL DRAWA(XMAX,YMIN)
CALL DRAWA(XMAX,YMAX)
CALL DRAWA(XMIN,YMAX)

J=NCONT
DO 6 I=1,NPOTAS+NURANI+NTHORI
J=J+1
CALL MOVEA(EPHI(J),0.0)
GO TO (4,5) NPLOT
4 Y=PHI(J)
GO TO 60
5 Y=ERROR(J)
IF(Y.LT.0.0) Y=-Y

```

```

        IF(Y.GT.1.0) Y=1.0
60      IF(I.LE.NPOTAS) GO TO 61
        IF(I.LE.(NPOTAS+NURANI)) GO TO 62
        IF(I.LE.(NPOTAS+NURANI+NTHORI)) GO TO 63
61      CALL DRAWA(EPhi(J),Y)
        GO TO 6
62      CALL DASHA(EPhi(J),Y,2)
        GO TO 6
63      CALL DASHA(EPhi(J),Y,3)
6      CONTINUE

        CALL TWINDO(0,1023,0,780)
        CALL MOVABS(0,351)
        CALL DRWABS(0,0)
        CALL DRWABS(1023,0)
        CALL DRWABS(1023,351)
        CALL DRWARS(0,351)
        CALL MOVARS(0,300)
        CALL DRWARS(1023,300)
        CALL MOVABS(511,351)
        CALL DRWABS(511,0)

        CALL MOVABS(0,317)
        CALL ANMODE
        IF(NPLOT.EQ.1) TYPE 9000
        IF(NPLOT.EQ.2) TYPE 9001
9000     FORMAT(1H+,2(5X,'ENERGY',11X,'INTENSITY',6X)/S)
9001     FORMAT(1H+,2(5X,'ENERGY',11X,'ERROR %Z%',6X)/S)

        J=NCOUT
        DO 7 L=1,2
        CALL RECOVER
        CALL MOVABS(0,267)
        CALL ANMODE
        DO 7 I=1,12
        J=J+1
        IF(J.GT.(NC+1)) GO TO 8
        IF(L.EQ.2.AND.I.EQ.12) GO TO 8
        IF(NPLOT.EQ.1) Y=Phi(J)
        IF(NPLOT.EQ.2) Y=100.*ERRR(J)
        IF(L.EQ.1.AND.J.LE.NC) TYPE 9002,EPhi(J),Y
        IF(L.EQ.2.AND.J.LE.NC) TYPE 9003,EPhi(J),Y
        IF(J.EQ.(NC+1)) TYPE 9004
7        CONTINUE
9002     FORMAT(1H+,T7,F6.3,10X,1PE10.3/S)
9003     FORMAT(1H+,T44,F6.3,10X,1PE10.3/S)
9004     FORMAT(/S)
9005     FORMAT(1H+,T42,'STANDARD DEVIATION',1PE10.3/S)
9006     FORMAT(1H+,T5,'STANDARD DEVIATION',1PE10.3/S)

8        IF(NPLOT.EQ.1) GO TO 10
        IF(L.EQ.1) GO TO 9
        TYPE 9005,RMS
        GO TO 10
9        TYPE 9006,RMS
10       CALL XSCAL(XMIN,XMAX,416)
        CALL BELL
        IF(VALUE.EQ.1) CALL MULS
           (Phi(NCONT+1),Phi(NCONT+1),1./XNORM,NLINES)
        ACCEPT 9007,STATUS

```

9007 FORMAT(A1)
 CALL NEWPAC

RETURN
END

SUBROUTINE XSCAL(XMIN,XMAX,IY)

* DISPLAY A HORIZONTAL SCALE WITH VALUES XMIN TO XMAX
* AT SCREEN HEIGHT IY

```

      DX=1.0
      DO 1 MU=1,32
      I=(XMAX-XMIN)/DX
      IF(I,GE,32) GO TO 2
1     DX=DX/2.
      RETURN

2     CALL MOVABS(0,IY)
      CALL DRWABS(1023,IY)

      X=0.
      I=0
3     X=X+DX
      I=I+1
      IF(I,EQ,8) I=0
      IF(X,GT,XMIN) GO TO 4
      GO TO 3

4     IX=1023.*(X-XMIN)/(XMAX-XMIN)
      IF(I,GT,0) GO TO 5
      CALL MOVABS(IX,IY+8)
      CALL DRWREL(0,-17)
      IF(IX,GT,925) GO TO 6
      CALL MOVREL(-45,-23)
      CALL ANMODE
      TYPE 9000,X
9000  FORMAT(1H+,F6.2,3)
      CALL RECOVER
      GO TO 6
5     CALL MOVABS(IX,IY+4)
      CALL DRWREL(0,-9)
6     X=X+DX
      I=I+1
      IF(I,EQ,8) I=0
      IF(X,LT,XMAX) GO TO 4

      CALL MOVABS(950,IY-32)
      CALL ANMODE
      TYPE 9001
9001  FORMAT(1H+,' MEV'/3)
      CALL RECOVER
      CALL HOME

      RETURN
      END

```

```

SUBROUTINE WRIDAT
* WRITE FILES 'PARAMS.DAT', 'SOURCE.DAT', AND 'DETECT.DAT'

COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
; WOBBLE,CTOD,NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
; TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
; NTHORI,ALTITU,ETIME,KUT(32),SEQUEN(32)

COMMON/OPARAM/OUTSW,PHI1,THETA,PHI2,ROTATE(2,3)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
COMMON/SOURCE/EPHI(256),PHI(256),ERROR(256)
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
; G(256),O(256,32),K(256,32)

REAL K,KUT
INTEGER UNFSW,SIMSW,OUTSW,INSW,RECORD,VALUE
DOUBLE PRECISION DATFIL

CALL GETSW(INSW,2,VALUE)
IF(VALUE.NE.1) GO TO 1
CLOSE(UNIT=21,PROTECTION="007")
OPEN(UNIT=21,DEVICE='DSKC',FILE='SOURCE.DAT',
; ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
; ASSOCIATE VARIABLE=I21,RECORD SIZE=NC)
CLOSE(UNIT=21,DISPOSE='DELETE')
OPEN(UNIT=21,DEVICE='DSKC',FILE='SOURCE.DAT',
; ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
; ASSOCIATE VARIABLE=I21,RECORD SIZE=NC)
1 WRITE(21#1) (EPHI(I),I=1,NC)
WRITE(21#2) (PHI(I),I=1,NC)
WRITE(21#3) (ERROR(I),I=1,NC)
IF(VALUE.NE.1) CLOSE(UNIT=21)
IF(VALUE.EQ.1) CLOSE(UNIT=21,PROTECTION="227")

IF(VALUE.NE.1) GO TO 2
CLOSE(UNIT=22,PROTECTION="007")
OPEN(UNIT=22,DEVICE='DSKC',FILE='DETECT.DAT',
; ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
; ASSOCIATE VARIABLE=I22,RECORD SIZE=NR)
CLOSE(UNIT=22,DISPOSE='DELETE')
OPEN(UNIT=22,DEVICE='DSKC',FILE='DETECT.DAT',
; ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
; ASSOCIATE VARIABLE=I22,RECORD SIZE=NR)
2 WRITE(22#1) (EDATA(I),I=1,NR)
WRITE(22#2) (DATA(I),I=1,NR)
WRITE(22#3) (DELTA(I),I=1,NR)
WRITE(22#4) (DEXP(I),I=1,NR)
WRITE(22#5) (G(I),I=1,NR)
NREC=5
DO 3 J=1,32
NREC=NREC+1
3 WRITE(22#NREC) (O(I,J),I=1,NR)
DO 4 J=1,NC
NREC=NREC+1
4 WRITE(22#NREC) (K(I,J),I=1,NR)
IF(VALUE.NE.1) CLOSE(UNIT=22)
IF(VALUE.EQ.1) CLOSE(UNIT=22,PROTECTION="227")

* RESET INSW(2) TO ZERO

```

```

CALL SETSW(INSW,2,0)

OPEN(UNIT=20,DEVICE='DSKC',FILE='PARAMS.DAT',
;   ACCESS='RANDOM',MODE='IMAGE',
;   BUFFER COUNT=1,RECORD SIZE=32)
CLOSE(UNIT=20,PROTECTION='227')
OPEN(UNIT=20,DEVICE='DSKC',FILE='PARAMS.DAT',
;   ACCESS='RANDOM',MODE='IMAGE',
;   BUFFER COUNT=1,RECORD SIZE=32)
WRITE(20#1) UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;   TAU,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,
;   NTHORI,ALTITU,ETIME
WRITE(20#2) KUT
WRITE(20#3) SEQUEN
WRITE(20#4) SIMSW,MSTATE,NTRY,DIAMET,THICKN,
;   WOBBLE,CTOD,NGAUSS,XNORM
WRITE(20#5) ELINE
WRITE(20#6) PHISIM
WRITE(20#7) OUTSW,PHI1,THETA,PHI2,ROTATE
WRITE(20#8) INSW,DATFIL,RECORD,LOCHAN
CLOSE(UNIT=20,PROTECTION='557')

RETURN
END

```

```
9      DO 9 I=MIN,MAX  
        R(I,I)=0.0  
      RETURN  
    END
```


THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

```

SUBROUTINE RUNSUM(NRECLD,NRECHI)
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,
;   WOBBLER,CTOD,NGAUSS,XNORM,ELINE(32)
COMMON/SCRATC/DJM(256,3),KBAR,UBAR,TBAR,
;   SIGMAK,SIGMAU,SIGMAT
COMMON/ANA/HA,ETIM,COSMIC
COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;   TAUD,NCONT,NLINES,COUNTS,TAUC
REAL K
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256)
;   ,G(256),O(256,32),K(256,32)
COMMON /SOURCE/EPHI(256),PHI(256),URATIM(256)
INTEGER INSW,RECORD,UNFSW
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
DOUBLE PRECISION DATFIL,FILNM
DIMENSION ETIME(20),SLT(21),H(20),XMU(18),TEMP(1),DTH(9),
;   DU(9),CHI(200),AK(200),AU(200),ATH(200),IDATA(128)
;   ,IMPDAT(257,20),COSM(20)
DIMENSION PHIRUN(200,21),RMS(21),SUMR(21)
DATA(XMU(I),I=1,18)/
;   .052,.100,.081,.072,.066,.060,.052,.048,.042,.039,
;   .102,.079,.072,.064,.062,.056,.048,.038/
OPEN(UNIT=26,FILE=DATFIL,MODE='IMAGE',
;   ACCESS='SEQIN')
DATA IDVSR/'000001000000/'
DATA MASK/'000000777777/'
CALL MULS(ERROR,ERROR,0.0,256)
REWIND 25
TYPE 7000
7000 FORMAT(' <<ENTER NUMBER OF SECONDS TO BE AVERAGED FORMAT=I2')
ACCEPT 7001, NAVE
7001 FORMAT(I2)
M0=400.
RECORD =0
IF(NRECLD.EQ.1) GO TO 25
27 READ(26,ERR=13,END=14) IDATA
IF(IDATA(128).GT.1000) GO TO 27
RECORD =RECORD+1
IF (RECORD.LT.NRECLD-1) GO TO 27
25 CONTINUE
DO 24 KK=1,18
SLT(KK)=0.
24 CONTINUE
COSMIC=0.
HA=0.
ETIM=0.
DO 28 I=1,256
28 DATA(I)=0.
7006 FORMAT(A10)
TYPE 7005
ACCEPT 7006,FILNM
7005 FORMAT(' ENTER OUTPUT FILE NAME')
OPEN(UNIT=27,FILE=FILNM,MODE='ASCII',ACCESS='SEQOUT')
WRITE(27,7007)DATFIL,NAVE,TAUD,TAUC
7007 FORMAT(' RUNNING SUM ANALYSIS OF ',A10/
;   ' NAVE=',I3,' TAUD=',F8.2,' TAUC=',F7.3)
CALL ERASE
YMIN=ALOG10(1.)
YMAX=ALOG10(1000.)
XMAX=NRECHI-NRECLD

```

```

XMIN=XMAX/2.
CALL DWINDO(0.,XMAX,YMIN,YMAX)
XOLD=XMIN
CALL MOVEA(XMIN,YMAX)
CALL DRAWA(XMIN,YMIN)
CALL DRAWA(XMAX,YMIN)
CALL DRAWA(XMAX,YMAX)
CALL DRAWA(XMIN,YMAX)
DO 229 I=1,3
DO 229 J=1,10
AJ=J
Y=YMAX*(I-1)/3.+ALOG10(AJ)
X1=XMIN
X2=XMIN+.2
CALL MOVEA(X1,Y)
CALL DRAWA(X2,Y)
229 CONTINUE
CALL HOME
TYPE 2841
2841 FORMAT(
;
TYPE 2842
2842 FORMAT(
;
TYPE 2843
2843 FORMAT(
;
TYPE 2844
2844 FORMAT(
;
DO 30 N=1,NAVE
RECORD=RECORD+1
23 READ(26,ERR=13,END=14)IDATA
IF(IDATA(128).GT.1000)GO TO 23
IT=IDATA(1).AND. MASK
COSM(N)=IT
IT=(IDATA(11)/IDVSR).AND. MASK
COSMIC=COSMIC+COSM(N)
ETIME(N)=IDATA(128)/1000.
ETIM=ETIM+ETIME(N)
H(N)=1.075*IT-239.
IF(IT.LT.2644)H(N)=.1856*IT-38.5
IF(H(N).LE.0.1 H(N)=400.
HA=HA+H(N)/NAVE
DO 29 KK=1,18
29 SLT(KK)=SLT(KK)+ETIME(N)*EXP(-XMU(KK)*4.35E-02*(H(N)-H0))
II=2
DO 31 J=2,127
II=II+1
IMPDAT(II,N)=(IDATA(J)/IDVSR).AND. MASK
II=II+1
IMPDAT(II,N)=IDATA(J).AND. MASK
31 CONTINUE
DO 32 J=1,NR
32 DATA(J)=DATA(J)+IMPDAT(J+LDCHAN-1,N)
30 CONTINUE
376 FORMAT('      N    CHI-SQ    K          U          TH    KW
; UW          THW')
DO 34 LM=1,NLINES
34 SUMR(LM)=0.

```

```

IND=0
SUMK=0.
SUMTH=0.
SUMU=0.
SUMC=0.
35 CALL MAZASO
CALL WINDOW(DATA,SK,SU,STH)
NRC=RECORD-NAVE/2
DK=PHI(NCONT+1)/SLT(1)
M1=NCONT+2
M2=NCONT+10
M3=NCONT+11
M4=NCONT+18
DO 36 KK=M1,M2
36 DU(KK-M1+1)=PHI(KK)/SLT(KK-NCONT)
DO 37 KK=M3,M4
37 DTH(KK-M3+1)=PHI(KK)/SLT(KK-NCONT)
IND=IND+1
DO 39 LM=1,NLINES
PHIRUN(IND,LM)=PHI(LM+NCONT)/SLT(LM)
39 SUMR(LM)=SUMR(LM)+PHIRUN(IND,LM)
AK(IND)=SK
AU(IND)=SU
ATH(IND)=STH
SUMK=SUMK+SK
SUMU=SUMU+SU
SUMTH=SUMTH+STH
TEMP(1)=CHISO(X)
CHI(IND)=TEMP(1)
SUMC=SUMC+CHI(IND)
NS=RECORD-NAVE
X=XMIN+(RECORD-NRECLO-NAVE+1)*.5
Y1=ALOG10(SU)
IND1=MINO(IND,256)
URATIM(IND1)=(DU(5)+DU(6)+DU(7)+DU(8))/4.
Y2=ALOG10(URATIM(IND1))
Y3=ALOG10(DTH(8)*4.)
Y4=ALOG10(STH*3)
PY=(YMAX-YMIN)/200.
PX=XMAX/300.
CALL MOVEA(X,Y3+PY)
CALL DRAWA(X,Y3-PY)
CALL MOVEA(X-PX,Y3)
CALL DRAWA(X+PX,Y3)
CALL MOVEA(X-PX,Y4-PY)
CALL DRAWA(X+PX,Y4+PY)
CALL MOVEA(X-PX,Y4+PY)
CALL DRAWA(X+PX,Y4-PY)
CALL MOVEA(XOLD,YOLD)
CALL DRAWA(X,Y1)
CALL MOVEA(X,Y2)
CALL POINTA(X,Y2)
CALL ANMODE
XOLD=X
YOLD=Y1
377 FORMAT(15,7E10.3)
WRITE(27,301)NS,RECORD,TEMP(1),DK
WRITE(27,300)(DU(KK),KK=1,9)
WRITE(27,300)(DTH(KK),KK=1,8)
WRITE(27,300)SK,SU,STH

```

```

301  FORMAT(2I5,2E12.3)
300  FORMAT(5E12.3)
      RECORD = RECORD+1
      MMM=NRECHI+NAVE
      IF(RECORD.GT.MMM) GO TO 200
      ETIM=ETIM-ETIME(1)
      HA=HA-H(1)/NAVE
      COSMIC=COSMIC-COSM(1)
      DO 60 J=1,NR
200  DATA(J)=DATA(J)-IMPDAT(J+LCCHAN-1,1)
      DO 161 I=1,NAVE-1
      DO 161 J=1,254
      IMPDAT(J,I)=IMPDAT(J,I+1)
161  CONTINUE
      DO 61 I=1,NAVE-1
      H(I)=H(I+1)
      ETIME(I)=ETIME(I+1)
      COSM(I)=COSM(I+1)
61  CONTINUE
      DO 59 KK=1,18
      SLT(KK)=SLT(KK)+ETIME(1)*EXP(-XMU(KK)*4.35E-02*(H(1)-H0))
59  CONTINUE
22  READ(26,ERR=13,END=14)IDATA
      IF(IDATA(128).GT.1000) GO TO 22
      IT=IDATA(1).AND.MASK
      COSM(NAVE)=IT
      COSMIC=COSMIC+COSM(NAVE)
      ETIME(NAVE)=IDATA(128)/1000.
      IT=(IDATA(1)/IDVSR).AND.MASK
      H(NAVE)=1.075*IT-2390.
      IF(IT.LT.2644)H(NAVE)=0.1856*IT-38.5
      IF(H(NAVE).LE.0.) H(NAVE)=400.
      ETIM=ETIM+ETIME(NAVE)
      HA=HA+H(NAVE)/NAVE
      DO 62 KK=1,18
      SLT(KK)=SLT(KK)+ETIME(NAVE)*EXP(-XMU(KK)*4.32E-02*(H(NAVE)-H0))
62  CONTINUE
      II=2
      DO 261 J=2,127
      II=II+1
      IMPDAT(II,NAVE)=(IDATA(J)/IDVSR).AND.MASK
      II=II+1
      IMPDAT(II,NAVE)=IDATA(J).AND.MASK
261  CONTINUE
      DO 262 J=1,NR
262  DATA(J)=DATA(J)+IMPDAT(J+LCCHAN-1,NAVE)
      GO TO 35
200  CONTINUE
      MAM=N(LINES)-3
      DO 210 LM=1,MAM
      RMS(LM)=0.
210  SUMR(LM)=SUMR(LM)/IND
      SUMC=SUMC/IND
      SUMTH=SUMTH/IND
      SUMK=SUMK/IND
      RMSU=0.
      RMSK=0.
      RMSTH=0.
      RMSC=0.
      SUMU=SUMU/IND

```

```

DO 220 INDEX=1,IND
RMSU=RMSU+(SUMU-AU(INDEX))**2
RMSC=RMSC+(SUMC=CHI(INDEX))**2
RMSK=RMSK+(SUMK-AK(INDEX))**2
RMSTH=RMSTH+(SUMTH-ATH(INDEX))**2
DO 220 LM=1,MAM
220 RMS(LM)=RMS(LM)+(SUMR(LM)-PHIRUN(INDEX,LM))**2
RMSC=SQRT(RMSC/IND)/(.01*SUMC)
RMSU=SQRT(RMSU/IND)/(.01*SUMU)
RMSK=SQRT(RMSK/IND)/(.01*SUMK)
RMSTH=SQRT(RMSTH/IND)/(.01*SUMTH)
DO 230 LM=1,MAM
230 RMS(LM)=SQRT(RMS(LM)/IND)/(.01*SUMR(LM))
MR=NRECHI-NRECLD
WRITE(27,298)
298 FORMAT('EEEEEEEEEE')
302 FORMAT(2E12.3)
WRITE(27,299)MR
299 FORMAT(I5)
WRITE(27,300)(D(I,27),I=1,4R)
WRITE(27,300)(D(I,28),I=1,4R)
WRITE(27,300)(D(I,29),I=1,4R)
WRITE(27,302)(RMS(LM),LM=1,MAM)
WRITE(27,302)SUMK,SUMU,SUMTH
WRITE(27,302)RMSK,RMSU,RMSTH
WRITE(27,302)SUMC,RMSC
CALL HOME
DO 240 LM=1,MAM
TYPE 7631,L4,ELINE(LM),SUMR(LM),RMS(LM)
7631 FORMAT( I5,F6.3,2E10.3)
240 CONTINUE
TYPE 7632,SUMK,RMSK,SUMU,RMSU,SUMTH,RMSTH,SUMC,RMSC
7632 FORMAT(' K ',2E13.3/' U ',2E13.3/' TH ',2E13.3/
, CHI ',2E13.3)
CLOSE(UNIT=26)
CLOSE(UNIT=27)
ACCEPT 7001,ICD
CALL ERASE
CALL DWINDO(0.,XMAX,YMIN,YMAX)
CALL MOVEA(XMIN,YMAX)
CALL DRAWA(XMIN,YMIN)
CALL DRAWA(XMAX,YMIN)
CALL DRAWA(XMAX,YMAX)
CALL DRAWA(XMIN,YMAX)
DO 8229 I=1,3
DO 8229 J=1,10
AJ=J
Y=YMAX*(I-1)/3.+ALOG10(AJ)
X1=XMIN
X2=XMIN+.2
CALL MOVEA(X1,Y)
CALL DRAWA(X2,Y)
8229 CONTINUE
CALL HOME
TYPE 8841
8843 FORMAT(
,
TYPE 8842
8842 FORMAT(
,

```

- THX4')

. U')


```

      TYPE 8843
8841  FORMAT(
      ,
      MR=NRECHI-NRECLO
      Y=ALOG10(10.*O(1,27))
      CALL MOVEA(XMIN,Y)
      DO 8763 I=2,MR
      Y=ALOG10(10.*O(I,27))
      X=.5*I+XMIN
8763  CALL DASHA(X,Y,3)

      Y=ALOG10(O(1,28))
      CALL MOVEA(XMIN,Y)
      DO 8764 I=2,MR
      Y=ALOG10(O(I,28))
      X=.5*I+XMIN
8764  CALL PUINTA(X,Y)

      Y=ALOG10(4.*O(1,29))
      CALL MOVEA(XMIN,Y)
      DO 8765 I=2,MR
      Y=ALOG10(4.*O(I,29))
      X=.5*I+XMIN
8765  CALL DRAWA(X,Y)

      CALL HOME
      TYPE 8766,DATFIL,KBAR,SIGMAK,UBAR,SIGMAU,TBAR,SIGMAT
8766  FORMAT(' TIME FILTERED ANALYSIS OF ',1A10//
      , ' KBAR=',F8.3,' RMSD ',F8.3/
      , ' UBAR=',F8.3,' RMSD ',F8.3/
      , ' TBAR=',F8.3,' RMSD ',F8.3)

* WRITE URANIUM TIME SERIES ONTO DISK

      OPEN(UNIT=27,DEVICE='DSKC',FILE='TIME.DAT',
      , ACCESS='RANDOM',MODE='IMAGE',BUFFER COUNT=1,
      , RECORD SIZE=256)
      WRITE(27#1) UPA*IM
      CLOSE(UNIT=27)

      RETURN
13  TYPE 305
      RETURN
14  TYPE 306
      RETURN
305  FORMAT(' << ERROR ENCOUNTERED IN READING DATA TAPE')
306  FORMAT(' << EOF ENCOUNTERED' )
      END
      SUBROUTINE CHANG
      COMMON/UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
      , TAU,NCONT,NLINES,COUNTS,EMAX,NSEC,NPOTAS,NURANI,NTHORI,ALTITU,
      , ETIME,KUT(32),SEQUEN(32)
      COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
      INTEGER INSW,RECORD
      COMMON /SCRATC/ DATIN(256),EDATA1(256),EDATA2(256),XM,
      , W(8),ALPHA(5),DZ(4),DS(4),SLOPIN,ZEROIN
      DIMENSION KN(256,32)
      DOUBLE PRECISION DATFIL,DWJRD
      COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),G(256)

```

```

      ,O(256,32),K(256,32)
      REAL K,KUT,KN
      DIMENSION ENEW(256),IDATA(128),DATOLD(256),EOLD(256),DATNEW(256)
700  FORMAT(' <<ENTER OUTPUT FILE NAME')
701  FORMAT(1A10)
      TYPE 200
200  FORMAT(' <<OLD RF FILE.EXT?')
      ACCEPT 701, DWORD
      OPEN(UNIT=28,ACCESS='SEQIN',FILE=DWORD,MODE='IMAGE')
      READ(28)((K(J,I),J=1,256),I=1,24)
      DO 10 I=1,NLINES+3
      CALL REBIN(K(1,I),EDATA2,SLOPE,256,
      KN(1,I),EDATA1,SLOPIN,256)
10  CONTINUE
      CLOSE(UNIT=28)
      TYPE 700
      ACCEPT 701,DWORD
      OPEN(UNIT=28,ACCESS='SEQOUT',FILE=DWORD,MODE='IMAGE')
      SLOPE=SLOPIN
      ZERO=ZEROIN+SLOPE*(LOCHAN-1)
      WRITE(28) KN
      CLOSE(UNIT=28)
      RETURN
      END

```

```

SUBROUTINE WINDOW(DATA,SUMK,SUMU,SUMT)
DIMENSION DATA(1)
COMMON /ANA/H,TIME,COSMIC
COMMON/UPARAM/
; UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,TAU,NCONT
INTEGER UNFSW
REAL KUT
DATA ALFA1,BETA1,GAMA1/.4,.52,.91/
DATA ALFA2,BETA2,GAMA2/.43,.56,.95/
DATA BGK,BGU,BGTH/.04668,.04131,.0498/
N1=(1.361-ZERO)/SLOPE+.5
N2=(1.561-ZERO)/SLOPE+.5
N3=(1.665-ZERO)/SLOPE+.5
N4=(1.865-ZERO)/SLOPE+.5
N5=(2.414-ZERO)/SLOPE+.5
N6=(2.814-ZERO)/SLOPE+.5
SUMK=0.
SUMU=0.
SUMT=0.
DO 10 I=N1,N2
10 SUMK=SUMK+DATA(I)
DO 20 I=N3,N4
20 SUMU=SUMU+DATA(I)
DO 30 I=N5,N6
30 SUMT=SUMT+DATA(I)
SUMK=(SUMK-BGK*COSMIC)/TIME
SUMU=(SUMU-BGU*COSMIC)/TIME
SUMT=(SUMT-BGTH*COSMIC)/TIME
A=ALFA1+(H*.3048-50.)*(ALFA2-ALFA1)/75.
B=BETA1+(H*.3048-50.)*(BETA2-BETA1)/75.
G=GAMA1+(H*.3048-50.)*(GAMA2-GAMA1)/75.
SUMU=SUMU-A*SUMT
SUMK=SUMK-B*SUMT-G*SUMU
SUMU=SUMU*EXP(1.7E-03*(H-400.))
SUMT=SUMT*EXP(1.7E-03*(H-400.))
SUMK=SUMK*EXP(2.3E-03*(H-400.))
RETURN
END

```

```

SUBROUTINE DATSIM
REAL K
DOUBLE PRECISION DWORD,DATFIL,FWORD
COMMON/DETECT/EDATA(256),DATA(256),DELTA(256),DEXP(256),
;      G(256),O(256,32),K(256,32)
COMMON /UPARAM/UNFSW,NR,NC,SLOPE,ZERO,RESOLU,POWER,
;      TAU,NCONT,NLINES,COUNTS,EMAX,ITER,
;      NPOTAS,NURANI,NTHORI
COMMON/SPARAM/SIMSW,MSTATE,NTRY(16),DIAMET,THICKN,WOBBLER,CTOD,
;      NGAUSS,XNORM,ELINE(32),PHISIM(32)
COMMON/IPARAM/INSW,DATFIL,RECORD,LOCHAN
INTEGER RECORD,INSW,SIMSW
DIMENSION PHIO(32),ENEW(256)
TYPE 3
3  FORMAT(' <<PARAMETER TO BE VARRIED; POTAS,THORI,URANI,GAIN,
;  CONTI?')
ACCEPT 2,DWORD
TYPE 1
1  FORMAT(' << SIMUL FILE.EXT?')
ACCEPT 2, FWORD
2  FORMAT(1A10)
OPEN(UNIT=29,MODE='IMAGE',FILE=FWORD,ACCESS='SEQUENT')
IGP=0
IF(DWORD.EQ.'CONTI') IGP=2
IF(DWORD.EQ.'GAIN') IGP=1
IF(DWORD.EQ.'POTAS') GO TO 10
IF(DWORD.EQ.'THORI') GO TO 20
NST=NCONT+NPOTAS+1
NEND=NCONT+NPOTAS+NURANI
GO TO 50
10  NST=NCONT+1
NEND=NCONT+NPOTAS
GO TO 50
20  NST=NCONT+NPOTAS+NURANI+1
NEND=NCONT+NPOTAS+NURANI+NTHORI
50  CONTINUE
51  FORMAT (2I5)
TYPE 51,NST,NEND

COUNTS=3611
X=10.
CALL PROJFD(DEXP,K,PHISIM,NR,NC)
A=SUMV(DEXP,NR)
C0=COUNTS/A
DO 60 I=1,NC
60  PHIO(I)=PHISIM(I)
DO 100 NRUN=1,101
IF(IGP.GT.0) GO TO 71
PHISIM(1)=PHIO(1)*(1+.5/(1.+EXP((60.-NRUN)/5.)))
DO 70 I=NST,NEND
70  PHISIM(I)=PHIO(I)*(1.+1./(1.+EXP((60.-NRUN)/5.)))
71  IF(IGP.LE.1) GO TO 72

DO 73 I=1,4
73  PHISIM(I)=PHIO(I)*(1.+1./(1.+EXP((60.-NRUN)/5.)))

72  CONTINUE
SLOPEP=SLOPE+WOBBLER*NRUN*SLOPE/22220.
ZEROP=ZERO+WOBBLER*NRUN*SLOPE/101.

```

```

*      WOBBLER=1 GIVES A ONE CHANNEL SHIFT IN EZERO
*      WITH ENERGY OF CHANNEL 222 REMAINING THE SAME
75      CALL PROJFD(DEXP,K,PHISIM,NR,NC)
        A=SUMV(DEXP,NR)
        CALL MULS(DATA,DATA,0,0,NR)
        CALL MULS(DEXP,DEXP,C0,NR)

        IF(IGP.NE.1) GO TO 76
        CALL BINDAT(ENEW,NR,SLOPEP,ZEROP)
        CALL REBIN(DEXP,EDATA,SLOPE,NR,
                   DATA,ENEW,SLOPEP,NR)
        TYPE 150,NRUN,ENEW(238),EDATA(238),DATA(20),DEXP(20),
        SLOPEP,SLOPE,ZEROP,ZERO
150      FORMAT(1H,15,8F7.3)
        GO TO 77
76      CALL MULS(DEXP,DATA,1,NR)
77      CONTINUE

        CONTINUE
        DO 95 I=1,257
        IF(I.GT.NR) GO TO 92
        CALL RANDP(DATA(I))
        GO TO 95
92      CALL RANDP(X)
95      CONTINUE

        CALL WRDT(DATA,NR,LOCHAN)

100     CONTINUE
        CALL MULS(PHIS,PHISIM,1,NC)
        CLOSE(UNIT=29)
        RETURN
        END
        SUBROUTINE WRDT(DATA,NR,LOCHAN)
        DIMENSION DATA(256),IDAT(128)
        M=1,NC/NR/2
        DO 10 I=1,NR,2
        M=M+1
        I1=DATA(I)+.5
        I1=I1*262144
        I2=DATA(I+1)+.5
        IDAT(M)=I1+I2
10      CONTINUE
        ICOS=RANDP(140.)
        IT=438./185
        IDAT(1)=IT*262144+ICOS
        IDAT(128)=999
        WRITE(29) IDAT
        RETURN
        END

```


THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

```

      SUBROUTINE BITS(NSTATE)
* GENERATE A RANDOM BIT SEQUENCE (0 AND 1)

      DIMENSION NPRIME(16)
      COMMON/SPARAM/SIMSW,MSTATE,NTRY(16)

      DATA (NPRIME(I),I=1,16)/2,3,5,7,11,13,17,19,29,
        31,37,41,43,47,53,59/
      DATA MSTATE/0/

      DO 1 I=1,16
        NTRY(I)=NTRY(I)+1
        IF(NTRY(I).NE.NPRIME(I)) GO TO 1
        NTRY(I)=0
        MSTATE=1-MSTATE
1      CONTINUE

      NSTATE=MSTATE

      RETURN
      END

```


SUBROUTINE RANDU(X)

* GENERATE A RANDOM NUMBER FROM AN ENSEMBLE
* UNIFORMLY DISTRIBUTED ON (0,1)

```
      X=0.0  
      SCALAR=1.0  
      DO 1 I=1,23  
        SCALAR=.5*SCALAR  
        CALL BITS(NSTATE)  
        IF(NSTATE.EQ.1) X=X+SCALAR  
1      CONTINUE  
  
      RETURN  
      END
```

SUBROUTINE RANDP(X)

- * MAKE X OUTPUT A POISSON VARIATE WITH
- * MEAN X INPUT GREATER THAN OR EQUAL TO ZERO

CALL RANDU(U)

XBAR=X
X=0.0
PLOG=-XBAR
SUM=EXP(PLOG)

1 IF(SUM.GE.U) GO TO 2
X=X+1.0
PLOG=PLOG+ALOG(XBAR/X)
SUM=SUM+EXP(PLOG)
GO TO 1

2 RETURN
END

FUNCTION DETEFF(E,T)

* APPROXIMATE DETECTION EFFICIENCY AT ENERGY
* E MEV FOR A NAI DETECTOR T INCHES THICK

DETEFF=10.0*EXP(-22.5*E)
DETEFF=DETEFF+.4092*EXP(-6.297*E)
DETEFF=DETEFF+.07217*EXP(-1.092*E)
DETEFF=DETEFF+.0324*EXP(.01286*E)

DETEFF=1.-EXP(-9.1675*DETEFF*T)

RETURN
END

```

      FUNCTION CMPNRH(E,EMIN)
      * INTEGRAL OF THE DEPOSITED ENERGY DISTRIBUTION
      * FOR SINGLE COMPTON SCATTER
      * EMIN IS THE MINIMUM ENERGY OF THE SCATTERED GAMMA

      DATA EM,EM2/.511006,.261127/

      GAMMA=EM/(EM+2.*E)
      EMIN=GAMMA*E

      CMPNRH=.5*E+EM2/(GAMMA*E)
      CMPNRH=CMPNRH*(1.+GAMMA)+1.022012
      CMPNRH=CMPNRH*(1.-GAMMA)
      CMPNRH=CMPNRH+(1.022012*(E+EM)/E-E)*ALOG(GAMMA)

      RETURN
      END

```

```

      FUNCTION CMPTN(E,EDEP,EMIN,CNORM)
      * COMPUTE SINGLE COMPTON RESPONSE COMPONENT
      * E #MEV# IS THE GAMMA ENERGY
      * EDEP #MEV# IS THE DEPOSITED ENERGY
      * EMIN #MEV# IS THE MINIMUM SCATTERED ENERGY
      * CNORM IS THE NORMALIZATION MULTIPLIER

```

```

      EDEL=E-EDEP
      IF(EDEL.LT.EMIN) GO TO 1

```

```

      CMPTN=(.511006/E)**2+EDEL/E
      -1.022012*EDEL/(E*EDEP)
      CMPTN=2.+CMPTN*(EDEP/EDEL)**2
      CMPTN=CNORM*CMPTN

```

```

      RETURN

```

```

1      CMPTN=0.0
      RETURN

```

```

      END

```

```

      FUNCTION FOFRAK(E,D,T)
      * APPROXIMATE PHOTOFRACTION FOR A D-BY-T
      * NAI DETECTOR AT ENERGY E
      * E #MEV, D AND T #INCHES#

      X=.0302*D

      FOFRAK=SQRT((X+.65*X**2)/(1.+X**2))
      FOFRAK=FUFRAK*(1.24-EXP(-4.752*T/(D+8.)))
      FOFRAK=FUFRAK/(FOFRAK+(1.-FOFRAK)*E/2.5)

      RETURN
      END

```

```

SUBROUTINE ADDPK(A,NR,EDATA,PEAK,E,
, RESO,POWER,NGAUSS)

```

```

* ADD DISCRETE COMPONENT TO RESPONSE VECTOR A
* WITH AMPLITUDE 'PEAK' AT ENERGY E

```

```

DIMENSION A(NR),EDATA(NR)

```

```

DELSIG=5.
IF(NGAUSS.EQ.2) DELSIG=20.
SIGMA=.424661*RESO*E*(.6616/E)**POWER
API=.398942*PEAK/SIGMA
IF(NGAUSS.EQ.2) API=.853048*API

```

```

ELD=E-DELSIG*SIGMA
EMI=E+DELSIG*SIGMA
DO 3 I=1,NR
IF(EDATA(I).GT.EMI) GO TO 4
IF(EDATA(I).LT.ELD) GO TO 3
IF(NGAUSS.GT.0) GO TO 1

```

```

IF(I.EQ.1) DELERG=EDATA(2)-EDATA(1)
IF(I.GT.1.AND.I.LT.NR) DELERG=.5*(EDATA(I+1)-EDATA(I-1))
IF(I.EQ.NR) DELERG=EDATA(NR)-EDATA(NR-1)
IF(EDATA(I).LT.E) X=1.-(E-EDATA(I))/DELERG
IF(EDATA(I).GE.E) X=1.-(EDATA(I)-E)/DELERG
IF(X.LT.0.0) X=0.0
API=X*PEAK/DELERG
A(I)=A(I)+API
GO TO 3

```

```

1 X=((E-EDATA(I))/SIGMA)**2
IF(NGAUSS.EQ.2) GO TO 2
A(I)=A(I)+API*EXP(-.5*X)
GO TO 3

```

```

2 IF(X.LE.25.) A(I)=A(I)+API*(EXP(-.5*X)+.038*EXP(-.02433*X))
IF(X.GT.25.) A(I)=A(I)+API*(.038*EXP(-.02433*X))

```

```

3 CONTINUE
4 RETURN
END

```

```

      SUBROUTINE LINE(A,NR,E,EFF,D,T,EDATA,P,
      , RESO,POWER,NGAUSS)
* COMPUTE A RESPONSE FUNCTION AT ENERGY E #MEV#.
* IT IS REPRESENTED AS THE NR-DIMENSIONAL VECTOR A,
* WITH EFFICIENCY 'EFF'. DETECTOR DIMENSIONS ARE
* SPECIFIED BY THE DIAMETER D #INCHES# AND THICKNESS
* T #INCHES#, CHANNEL STRUCTURE BY THE NR-DIMENSIONAL
* CHANNEL CENTER ENERGY VECTOR, EDATA. P IS FOR
* TEMPORARY STORAGE OF THE UNBLURRED RESPONSE.
* RESOLUTION IS SPECIFIED BY 'RESO', THE FRACTIONAL
* RESOLUTION FWHM/E AT E=.6616 #MEV#, AND BY
* 'POWER', THE EXPONENT IN THE RESOLUTION EQUATION
*
*      RES=RESO*(.6616/E)**POWER
*
* NGAUSS IS THE NUMBER OF GAUSSIAN COMPONENTS (0,1,2).
      DIMENSION A(NR),P(NR),EDATA(NR)
      PEAK=FOFRAK(E,D,T)
      EFF=(1.-PEAK)/CMPNR*(F,EMIN)
      DO 1 I=1,NR
1      P(I)=CHPTN(E,EDATA(I),EMIN,EFF)
      CALL GAUSS(A,NR,EDATA,P,RESO,POWER,NGAUSS)
      CALL ADDPK(A,NR,EDATA,PEAK,E,RESO,POWER,NGAUSS)

```



```

RHO=0.0
DO 2 I=1,NR
IF(I.EQ.1) DELERG=EDATA(2)-EDATA(1)
IF(I.GT.1.AND.I.LT.NR) DELERG=.5*
      (EDATA(I+1)-EDATA(I-1))
IF(I.EQ.NR) DELERG=EDATA(NR)-EDATA(NR-1)
A(I)=A(I)*DELERG
2  RHO=RHO+A(I)

EFF=DETEFF(E,T)

CALL MULS(A,A,EFF,NR)

EFF=RHO*EFF

RETURN
END

```

SUBROUTINE GAUSS(A,NR,EDATA,P,RESO,POWER,NGAUSS)

* INCORPORATE RESOLUTION EFFECTS INTO THE RESPONSE VECTOR.
 * ON INPUT, P CONTAINS THE GAMMA INTERACTION EFFECTS.
 * ON OUTPUT, A CONTAINS THE RESPONSE. EDATA IS THE CHANNEL
 * CENTER ENERGY VECTOR MeV . A AND P ARE NR-DIMENSIONAL,
 * AS IS EDATA. RESO IS THE FRACTIONAL RESOLUTION FWHM/ENERGY
 * AT $.6616 \text{ MeV}$. POWER IS THE EXPONENT OF THE RATIO
 * $.6616/\text{EDEP}$ IN THE RESOLUTION EQUATION. NGAUSS IS THE
 * NUMBER OF COMPONENTS (0,1,2).

DIMENSION A(NR),P(NR),EDATA(NR)

IF(NGAUSS.EQ.0) GO TO 7
 DELSIG=5.
 IF(NGAUSS.EQ.2) DELSIG=20.

DO 1 I=1,NR
 1 A(I)=0.0

EDEP=EDATA(1)
 SIGMA=.424661*RESO*EDEP*(.6616/EDEP)**POWER
 DELERG=EDATA(2)-EDATA(1)
 IMIN=1.-DELSIG*SIGMA/DELERG

DO 6 I=IMIN,NR

IF(I.GT.0) GO TO 2
 API=1-I
 EDEP=EDATA(1)-DELERG*API
 API=P(1)+(P(1)-P(2))*API
 SIGMA=.424661*RESO*EDEP*(.6616/EDEP)**POWER
 API=.398942*API/SIGMA
 GO TO 3

```

2   EDEP=EDATA(I)
    SIGMA=.424661*RESQ*EDEP*(.6616/EDEP)**POWER
    API=.398942*P(I)/SIGMA
    IF(I.EQ.1) DELERG=EDATA(2)-EDATA(1)
    IF(I.GT.1.AND.I.LT.NR) DELERG=.5*
      (EDATA(I+1)-EDATA(I-1))
    IF(I.EQ.NR) DELERG=EDATA(NR)-EDATA(NR-1)

3   IF(NGAUSS.EQ.2) API=.853048*API

    ELO=EDEP-DELSIG*SIGMA
    EHI=EDEP+DELSIG*SIGMA
    DO 5 II=1,NR

    IF(EDATA(II).GT.EHI) GO TO 6
    IF(EDATA(II).LT.ELO) GO TO 5

    X=((EDEP-EDATA(II))/SIGMA)**2

    IF(NGAUSS.EQ.2) GO TO 4
    A(II)=A(II)+API*EXP(-.5*X)*DELERG
    GO TO 5

4   IF(X.LE.25.) A(II)=A(II)+API*(EXP(-.5*X)
      +.038*EXP(-.024330*X))*DELERG
    IF(X.GT.25.) A(II)=A(II)+.038*API*EXP(-.02433*X)*DELERG

5   CONTINUE
6   CONTINUE
    RETURN

7   DO 8 I=1,NR
8   A(I)=P(I)
    RETURN
    END

```