PUB-3038
c.1

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

RECEIVED
LAWRENCE
BERKELEY LABORATORY

JAN 3 1985

LIBRARY AND
DOCUMENTS SECTION

WC - USER'S GUIDE

S. Wald and P. Countryman

February 1984

For Reference

Not to be taken from this room

NUCLEAR SCIENCE DIVISION

PUB-3038
c.1

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

DISCLAIMER

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

PUB-3038

WC - USER'S GUIDE*

Shlomo Wald

88-Inch Cyclotron
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

and

Peter Countryman

High Energy Physics Laboratory and
Department of Physics
Stanford University
Stanford, CA 94305

February 1984

*This work was supported by the Director, U. S. Office of Energy Research, Division of Nuclear Physics of the Office of High Energy and Nuclear Physics, and by Nuclear Sciences of the Basic Energy Sciences Program of the U.S. Department of Energy under Contract Nos. DE-AC03-76SF00098 and DE-AM03-76SF00326.

Abstract

WC is an interactive code for analyzing (or simulating) multiparameter experiments. It enables the user to perform complex manipulations of the data on an event-by-event basis, generate output tapes, and display results (as tables or graphs) in a variety of formats. Many powerful features can be invoked interactively, such as definition of one- and two-dimensional (free-form) gates, certain kinds of "pseudo-parameters", spectrum definitons (gated or otherwise) and "virtual" displays (also gated). Still more complex operations can be done on an event-by-event basis by means of a subroutine which can be modified to the user's specifications and linked to the body of the program. Operations can be made with ease upon the accumulated spectra, including χ^2 fits of various functions to the one-dimensional spectra. The spectra produced and the fits can also be plotted.

		Page
I.	Introduction	1
II	The overlay structure	2
III.	General remarks and the normal operation of the code.	4
IV.	Overlay descriptions	12
	1. SETEXP	12
	A. Commands - 1) AN	13
	2) SE	13
	3) SO	15
	4) ST	16
	2. INITS	16
	A. Commands - 1) SWOV	17
	2) PARM	18
	3) FORM	19
	4) CNST	20
	5) COND	20
	6) PSPR	20
	7) 1DSP	21
	8) 2DSP	24
	9) WCND	24
	10) GAT2	25
	11) LINR	26
	12) TSTX	27
	13) SSTS	28
	14) ZALL	30

PAGE

15) ZSPC 30

16) REST 30

17) SVSP 30

18) SVSE 30

19) DISP 30

20) SETE 31

21) ANLD 31

22) PRTS 31

23) VIRS 31

3. ANLDAT 32

A. VIRD 33

1) SETV 34

2) REFR 34

3) CURS 34

4) GATE 34

5) WGAT,LGAT 36

6) LINR 36

7) HCOP 37

8) ANLD 37

9) CONT 37

10) Final remarks on VIRD. 38

B. YFSR - Your Favorite Sub Routine 39

1) PSPR 43

2) LINR 43

3) ISTEST 44

4) INCSPC 44

PAGE

5) PSPR1 45

6) LINR1 45

7) INC1DS 46

8) INC2DS 46

9) ICOND 46

10) IGAT2 46

11) ITEST 47

12) Random-number generators 48

13) D2SOR1 49

14) NRMSPC 49

15) SQRTSP 50

16) PWR2SP 50

17) DIVSPC 50

18) MULSPC 51

19) ADDSPC 51

20) AVRSPC 51

21) DEF1DS 51

22) DEF2DS 52

23) GET1DS 52

24) GET2DS 52

25) DPS1DS 53

26) DPS2DS 53

27) GET2DP 53

28) COMBIN 54

29) SMOOTH 55

30) DSMOTH 55

PAGE

31) VMIN 55

32) VMAX 56

33) CURFIT 56

34) LEGFIT 57

35) PMAT 60

36) PARR 61

37) Final important remarks on YFSR 61

4. DISP 63

A. DSP1 Commands - 1) SW 64

2) PO 64

3) VW 64

4) SP 64

5) FR 65

6) PL 66

7) ES 66

8) PG 66

9) LL 66

10) HD 67

11) MW 67

12) WD 67

13) CU 68

14) LB 68

15) DV 68

16) HC 68

17) ER 68

PAGE

18) CL 68

19) AN 68

20) IN 68

21) SE 69

22) OC 69

23) PF 69

24) SC 69

25) MC 69

5. PRTSPC 72

V. Common Block Descriptions 73

VI. Instructions for changing WC 84

1. Replacement or addition of commands 84

2. Changing the dimension of arrays 84

3. Moving to another computer 87

4. Changing default values 89

5. Quick Load of YFSR 89

VII. Acknowledgments 90

VIII. Examples 91

IX. Figures 123

X. Appendices: 138

1. A Listing of the default YFSR

2. A. Listing of the default setup file

B. Listing of a more complex setup file

3. List of all subroutines

4. List of Procedures

Feb. 84 WC-Newsletter 183

WC

I. Introduction

WC is an interactive code for analyzing mutiparameter experiments. The code is installed now on the MODCOMP-CLASSIC computers and knows how to handle CHAOS-like event-by-event tapes. The code is written in such a way that it is easy to operate, easy to modify and easy to move to another computer.

The program can be used for:

1) Filtering data: producing a new event-by-event CHAOS-like tape after selection and modification of events from an existing tape or from simulated data.

2) Analyzing data: The program provides subroutines and functions which enable the user to apply complicated tests event-by-event, increment one-dimensional and two-dimensional spectra, increment scalers, etc.

Many operations can be done on the final spectra. This "language" enables the user to write an analyzing routine directly from a flow chart without thinking about formats, plots, zeroing scalers, etc.

3) Simulation codes: The code can be used to create simulated events and to analyze them as if they were the measured data.

All the above functions are carried out by the user via a single subroutine "YFSR" (Your Favorite Sub-Routine) which can be compiled by itself and then linked to the rest of the code. There are procedures for handling YFSR, saving and restoring the setup of the experiment and the spectrum buffer, etc.

The code is written in FORTRAN and is designed to run on minicomputers as well as on big computers. It is segmented so all parts can be overlaid. The overlay structure does not significantly affect the flexibility of the program. Changing the space devoted to spectra is a trivial task.

II. The Overlay Structure

The overlay structure is shown in Figure 1.

WC controls the flow between the overlays. Whenever the user switches from one overlay to the other the computer prompts with the message:

IGOTO,IPASS n m,

followed by the prompt of the overlay. (n and m provide information about the status of the program. The explanation of this code will be given in Sect.(IV.2.A1)).

1. SETEXP-Defines the "Hardware" tools needed for running the program, e.g. assign magnetic tape drivers, advance or skip backward a file, write EOF, etc. It also asks for information about the data format on the input and output tapes and if the setup of the experiment already exists.

   The information accumulated in SETEXP is not saved for future use and should be re-entered each time WC is executed.

2. INITS - Defines the "Software" tools, i.e. constants, one-dimensional gates (conditions), two-dimensional rectangular gates (window conditions), two-dimensional complex gates (gates), linearization of curves in two-dimensional space, pseudoparameters, spectrum definitions, etc. (Remark: gates and linearizations can be defined in VIRD as well).

   The information accumulated in INITS (except for "simple tests") is saved for future use and can be restored the next time WC is executed. We refer to these data as the "setup file" of the experiment. (See examples of setup files in Appendix 2).

3. ANLDAT - Analyzes the event-by-event data. The overlay loads the event buffer from the input record and transmits it to the user-supplied subroutine YFSR. It increments scalers, looks for interrupts, and plots points on virtual displays.

Besides calling YFSR (at the beginning of the program, for every event, and at the end of the program) ANLDAT allows the user to switch to VIRD. In the VIRD overlay one can set two-dimensional virtual displays ("life displays") and define on them two-dimensional free-form gates and linearizations, read positions on the 2-D space using the cross-hair thumb wheels, etc.

4. DISP - Plots spectra (in overlay DSP1) and operates on the spectrum buffer (in overlay SPCOPR). The user can display one-dimensional spectra and projections of two-dimensional spectra, set windows and integrate or average over windows. It provides operations on spectra such as the square root and square of spectra, normalization, addition, multiplication, and division of two spectra, smoothing, fitting to various common functions (or to a user-supplied function with up to 10 parameters) etc.

5. PRTSPC - Prints and plots spectra on a line printer (Versatec). The format of the output is defined in INITS.

In section IV a detailed description of all the overlays is given.

III. General Remarks and the Normal Operation of the Code

The normal flow of WC is shown in Figure 2.

Before using WC on the MODCOMP, make sure that a binary library UL$ and a TASK/OVERLAY library have been assigned to you. The first library is needed for saving the binary version of WC. If the user can not have a binary library on the disk, one can assign BI to a magnetic tape which holds this library. The second library is needed in order to save the executable module on the disk for future execution of WC.

1. The COMPWC procedure

In order to create the binary library of WC on UL$ follow the instructions below:

a. Mount the WC removable disk (or magnetic tape)

b. Type $ASS USL RSL (ASS USL MTn)
   $COMPWC

These commands erase your UL$, compile WC and put the updated binary file on your UL$.

2. The LOADWC procedure

   Type $ LOADWC

   This command creates a task catalogued under the name WC on your load-module library (FIL TM in TOC)

3. The ANALY procedure.

Now the user can compile, link and load his own subroutine. The user must have on his USL (user library) a file which contains at least the YFSR; Your Favorite SubRoutine. The user can attribute any name to his own program, which resides on his USL. This name is also automatically attributed to the catalogued load-module on the TASK/OVERLAY library by the procedure $ANALY. On the WC.disk (and tape) a default program exists under the name WCOO. In order to copy it to the user's USL type :

```
$EXE SED
ASS SI RSL  (or ASS SI MTn)
POS WCOO
CAT WCOO
ASS SI SI
EXI
To modify the program (if the user wishs to do so)
$EXE SED
POS WCOO
RAN
:Enter your changes
CAT NAME  (catalogue it in USL under the name NAME)
EXI       (return to the file manager)
$
```

In order to maintain the normal flow of the procedure it is recommended (but not required) to have a setup file on the USL. A default setup file exists on the WC removable disk and magnetic tape under the name SETUP.

Execute the ANALY procedure by typing $ANALY NAME, NAME2, NAME3

The procedure performs the following steps:

a. It compiles the file NAME which exists on USL and links it with WC. If NAME is missing WCOO is taken.

b. It invokes TOC (The Task/Overlay cataloguer) and catalogues the executable load module under the name NAME.

c. The procedure WC is executed. (The $ANALY procedure includes the command $WC NAME2, NAME3.)

If the load module NAME already exists, the user can go directly to the WC procedure.

4. The WC procedure

The following steps are done:

a. The setup file NAME2 is restored, i.e. it is copied from the USL to the scratch file SCO. This is done because the MODCOMP can read/write just from/to scratch files. If NAME2 is missing SETUP is taken.

b. If an old spectrum buffer exists on the USL (a file which contains spectra accummulated in a previous run of WC) under the name NAME3 it can be restored as well. The spectrum buffer is copied to SC2. If NAME3 is missing no spectrum buffer is restored. (a. and b. are performed by the procedure RESTFI).

c. The default assignments of I/O devices are set as follows:

IO logical unit number		Device name
1	INPUT	TY - The user's terminal
2	OUTPUT	TY - The user's terminal
3	INPUT/OUTPUT	NST - TEKTRONIX 401
4	OUTPUT	PLA/NST - TEKTRONIX 4014
5	OUTPUT	VP - Versatec
6	Input setup file	SC0
7	Output setup file	SC1
8	Input spectrum buffer	SC2
9	Output spectrum buffer	SC2
MTW		MT2
MTW		MT3
DSP		PLT/DSQ

d. The WC task (which was previously created by the procedure LOADWC) is executed by:

$EXE WC TM

5. The program can run on a graphics terminal such as the TEKTRONIX 4014 (with 4096x4096 channel resolution) on VT100 with SELANAR 200 PLUS Tektronix emulator, or on a standard terminal or on two terminals simultaneously. The benefit of running on two terminals is that conversation with the program does not overwrite the display of the spectra on the graphics terminal. The default assignments are for the two-terminal operation. To change this, execute WC (by $ANALY or WC), and when the message SETEXP >> appears type

a.	↑C	(Press CONTROL and C simultaneously)
	>	(This is the file manager reponse)
b.	ASS 1 ASn, 2 ASn, 3 ASn	
	>	
c.	R	(Resume WC execution)
d.	<CR>	(Enter carriage return)

The prompt SETEXP>> will appear on terminal ASn. (The number n is the BATCH number of the standard terminal. The hard-copy SILENT 700 is TYO the Tektronix is NST. If running VT100, assign DSP to ASn).

In order to return to the two terminal mode at any stage of the program, assign devices 1,2,3 back to their defaults.

ASS 1 TY, 2 TY, 3 TY

When the program has finished, the procedures automatically enter the $JOB command, which changes all assignments to the MODCOMP default ones.

6. As can be seen from the flow chart in Figure 2, the program normally flows from SETEXP through INITS, ANLDAT (YFSR or VIRD) and optionally to DISP (DSP1 or SPCOPR) or PRTSPC, and back to SETEXP. The ST command (STOP) in SETEXP generates a call to PRTSPC (where spectra are printed on the line printer), then a call to INITS (where the current setup and spectrum buffer are dumped on scratch files) and finally it terminates WC.

The User can switch between overlays at any stage and modify the normal flow of WC.

Example 1 demonstrates a typical conversation with the progam. In this example we analyze four runs from a tape on physical device MT2. The data format is CAMAC with no zero-skip. The setup file contains all needed information. The reader can see that <CR> is entered as a response to most of the questions. This is an important feature! To every question with a list of possible answers the first answer given is the default one (unless another option is explicitly declared as the default). (The underlined commands are the ones entered by the user).

7. Every overlay has its own prompt, as can be seen in Example 1, so the user knows at all times to whom he or she is talking.

8. A short summary of the status of the analysis is given whenever the user leaves ANLDAT. A full summary is given on the line printer (Versatec) when the command ST (STOP) is entered in SETEXP.

9. When the run is terminated by ST, the new setup file is saved on I07 (which is assigned by WC to the scratch file SC1). The spectrum buffer is saved on I09 (assigned to SC2) which is the <u>same</u> scratch file from which the previous spectrum buffer was restored (See III.4). If the user wants to save those files, he or she should type.

a. $SAVEFI NEWSET (or $SAVEFI NEWSET, SC1)

b. $SAVEFI NEWSPC, SC2

9a. will save the new setup on your USL under the name NEWSET.

9b. will save the new spectrum buffer on the user's USL under the name NEWSPC.

10. For a detailed description of the overlays see Section IV. A list of procedures used in conjunction with WC and detailed descriptions of their parameters (and of other useful procedures) are given in Appendix 4.

IV. Overlay Descriptions

In this section detailed descriptions of all overlays are given. All overlays and most of the commands have the HELP option. The recommended way to become acquainted with the program is to run it, try to find your way using HELP, and test all options. In this section clarification and useful suggestions for each command are given.

1. SETEXP

The prompt message of this overlay is SETEXP>>. The user first enters SETEXP automatically while running WC. Then he can switch from INITS, ANLDAT or DISP to SETEXP with the command SETEXP entered in the above overlays.

In SETEXP the "hardware tools" for the analysis are prepared. The normal flow of a program starts with assigment of devices (AS), positioning of the tapes (AF,AR), then SE (set). The overlay is used also for the normal termination of WC by the command stop.

The list of commands is given in Example 2. All commands are two-letter commands. (The user can print more letters if he wishes, for example, HELP instead of HE.)

The magnetic-tape handling commands (for INPUT and OUTPUT) AS, AF, AR, ET, WE are explained by the HELP command. Comments are given here just on the AS (ASSIGN) command. The procedure $WC automatically asigns MT2 for the input tape and no unit is assigned for the output tape. So if one

uses this unit as described, the AS command is not needed. If the user wants to change only one of these assigments, he can hit <CR> when he is asked about the other one. A negative unit number means that the unit is not needed. Notice that if one does not need a Mag-Tape he does not have to enter a negative number in AS, because the program knows it according to the information given in the SE command.

A. Command Descriptions.

1) AN - Switches to ANLDAT, without changing the previous status of the program, given by IPASS (See. explanations in IV.2.A.1).

2) SE - (SET) - In this command the user specifies the exact way he wants to handle the data. This command must be executed at least once in the program. Follow Example 1 or 2 with the comments below:

a. Graphic terminal: The default is Tektronix. Enter No if you do not have the TEKTRONIX 4014 (or the VT100 + SELANAR 200P). T,V or <CR> (carriage return) enables the graphic device.

b. Debug options: Option 1 prints on the terminal one out of every 75 events as it is read from the input tape and as it is written on the output tape (if there is an output tape). The number 75 an be changed in YFSR (change NDBMAX).

Option 2 prints event-by-event

Option 0 (the default) prints nothing

c. Input from device 6:

The default is Yes. Device 6 is assigned by $WC to be the scratch file SCO. Furthermore , the procedure $WC copies the user (or the default) setup file from USL to SCO. If the answer is Yes, overlay INITS will read the setup from this device. The answer No is needed only if WC was started by typing $EXE WC, TM and not by $ANALY or $WC procedures, (see III.3). The reason is that the last two procedures execute the procedure $RESTFI NAME which restores the setup file (either the default or the user's file). Note that if the default setup is restored, it has no practical importance since INITS generates the same setup anyhow.

d. CAMAC Format: Type Yes or <CR> if the data were written at the 88" or HILAC MODCOMP using CHAOS with CAMAC. Enter No if the multiplexer has been used. (The multiplexer modifies the four most significant bits and the program trims them).

e. Zero-skip: The default is No; usually, no hardware zero-skip is done, that is, there is a fixed-length event buffer.

f. Generate OUTPUT tape: The default is No. If the answer is Yes the question e. is asked again about zero-skipping on the output tape.

The format (CAMAC/MULTIPLEXER) is the same as for the input tape.

g. Modify HEADER: The default is No. If an output tape is produced the user can modify the header on the output tape or copy the one which is on the input tape. If the default answer is entered the header on the output tape is set to be the same as on the input tape. (No example is given.).

h. EOF after each run: The default is Yes. If an output tape is produced the program can create a run on the output tape for each run on the input tape (Yes), or combine all the runs (declared in i) into a single run on the output tape (No)

i. Number of runs: Enter the number of runs on the input tape to be analyzed using the above conditons. If it is a negative number, no input tape is assumed to exist. This option can be used for simulation codes. An additional important usage of this option is when the user wants to manipulate a spectrum buffer which was created before, using DISP without performing any event-by-event calculations in ANLDAT. If <CR> is entered the program remains in SETEXP overlay. Otherwise WC transfers control to the INITS overlay.

3) SO - (SORT) - After NRUNS have been analyzed according to the SE command, the user can reposition (or replace) the tapes and continue analyzing event-by-event, with the same conditions set in the previous

SE command. Note that the scalers, the test counters, and spectra are not erased. The new runs are added to the previous one(s). If <CR> is entered the program remains in SETEXP overlay. Otherwise WC transfers control to the INITS overlay.

4) ST-(STOP) - Generates a sequence of operations which is ended by terminating WC. It switches to INITS and in INITS the setup file is saved on the scratch file assigned to I/O unit 7 (the default set in $WC is SC1). It saves the spectrum buffer on I/O unit 9 (the default is SC2). Then the program switches to PRTSPC, where spectra are printed and/or plotted. Finally WC is terminated and the $WC preocedure is ended with a $JOB command which sets all the assigments to the default ones. (See example 3).

2. INITS:

The prompt in this overlay is ENTER INIT COMMAND. For help enter HELP>>>

The user enters INITS from SETEXP as a result of the SE or SO commands, or by typing IN, and enters from ANLDAT and DISP by typing INIT.

In INITS the "software tools" for the analysis are prepared. Spectra, gates, constants, etc. are defined interactively or read from an input file ("setup file") on I/O unit 6. (The input setup file is read whenever the program enter INITS with the status code IPASS=1. See remarks in SWOV command below). It enables the zeroing of the spectrum buffer or the restoration of a previous spectrum buffer from I/O unit 8.

The routine provides the user with the possibility of defining, without recompiling YFSR, a "simple test" which is a complex gate that can be applied on the data event-by-event.

In this overlay the sophisticated user can modify the status of the program as he pleases, by using the SWOV (switch overlay) command.

Most of the "software tools" of INITS are saved at the terminaton of WC on I/O unit 7 (asigned to scratch file SC1 by $WC) for future use. We refer to this file as the setup file.

A list of the commands in INITS is given in Example 4. All commands are four-letter commands.

A. Command description

1) SWOV - (Example 5) SWOV switches between overlays and modifies the WC program status (IPASS). The user can change two parameters, IGOTO and IPASS. IGOTO is a number that determines the destination overlay; the codes are given in Example 5. IPASS tells what the status of the program is with respect to the tasks given to it (in SE

or SO commands in SETEXP). Example 5 gives the codes for IPASS and the overlays in which they are set in the normal flow of the program. IGOTO and IPASS are printed on the terminal whenever the program switches from one overlay to another.

Most important note: It is necessary to enter the overlays INITS and ANLDAT, at least once, with IPASS=1. This is most easily done by means of the SE command in SETEXP, followed by the ANLD command in INITS.

2) PARM - (Example 6) The program is designed to handle experiments with up to 100 parameters per event (either measured or calculated in WC). The number of parameters can be easily changed (see VI.2). This command is used to assign mnemomics and the number of channels for each parameter.

a. MODIFY I/O number of PARAMETERS - The maximum number of parameters on the input tape (IPARI) and the maximum number of parameters on the output tape (IPARO) can be modified. IPARI and IPARO appear in common EVNCOM. These numbers are the exact number of I/O parameters for a fixed event buffer (when the zero-skip option is not used). They are read from the run header by SE or SO (in SETEXP) and can be modified here. The importance of this function is in solving a special problem of CHAOS in which the number of parameters written in the header is

different from the actual number of parameters, if the latter is higher than 32 parameters. Note that these numbers are not saved on the setup file.

b. ENTER the number of PARAMETERS - This number is the total number of parameters either measured or created by the user in WC.

c. CHAN - Data are read from and written to the input and output tapes in INTEGER*2. Sometimes it is useful to use the number of channels per parameter for a fast check for overflows, etc. The program does not use this information but it is available to the user in YFSR through the common PARCOM, in the buffer LPARM (100). The setting of the default number of channels and the modifying of individual parameters is possible under the command PARM with subcommand CHAN.

d. MEMO - MEMO assigns four-letter mnemonics (labels) to each parameter. These labels appear on any plot or printout when the parameters are addressed.

e. END - END exits the PARM command.

3) FORM - FORM defines a variable event buffer format for input and output according to a TAG WORD. This is the way it is done in the LYNEN system (operating at the Max-Planck Institute fur Kernphysik -

Heidelberg and the GSI in W. Germany). The subroutines GETEVT and PUTEVT must be modified in order to use this variable format mode. The variable format information is saved in the commons VARCO1 and VARCO2.

4) CNST - CNST adds, changes or lists constants (See Example 7). These constants are transferred to YFSR by the common CNSCOM. Up to 100 constants can be defined.

5) COND - Conditions (or one-dimensional gates) can be added, changed, deleted or listed. Each condition is defined by a lower-limit (XL) an upper-limit (XU) and a default parameter number (NP). These conditions are transferred to the function ICOND(NPX,NC) by the common CONCOM (see description of ICOND in (IV.3.B.9)). The function checks if parameter NPX is within the closed interval [XL,XU].

6) PSPR - PSPR defines PSEUDO-PARAMETERS. (See Example 8). When starting the analysis of an experiment, it is recommended to use the default YFSR for setting all the calibrations, instead of changing and recompiling YFSR again and again. For this reason, the default YFSR was written in such a way that the user can do certain operations on the data without modifying YFSR. One of the operations is to modify the input parameters or to create new parameters from the input ones. We refer to these modified or new parameters as pseudo-parameters. Four types of pseudo-parameters exist (See Example 8). (The user can

add his or her own types very easily by adding them to subroutines PSPRS and PSPR1 in WC). C1 and C2 are constants. E, CHANNEL, M,T,ΔE etc. stand for parameters. Up to 100 pseudo-parameters can be defined, but the total number of measured and pseudo-parameters is limited to one hundred as well. The list given in Example 8 should be decoded in the following way:

a. 1st line: Pseudo-parameter 1 of type 1 (energy calibration). A new parameter 22 is defined which is the input parameter 3 after normalizaion: EVENT(22)=1.*(EVENT(3)+2048.)

b. 2nd line: Here we make use of pseudo-parameter 1 to define pseudo-parameter 2 (2 in the 1st column) of type 4 (adding parameters). The parameter number of the pseudo-parameter is 22, i.e. we are modifying an already-existing parameter. The result is : EVENT(22) = EVENT(22)- EVENT(4). The effect of both lines together is:

EVENT(22) = EVENT(3) - EVENT(4)+2048.

(This is the x-position calculated from the time signals from both sides of a position-sensitive, parallel-plate detector.

All the information from PSPR passes to the subroutine PSPR1 by the common PSPCOM.

7. 1DSP - 1DSP defines one-dimensional spectra (See Example 9.) A one-dimensional spectrum is defined on parameter n (NP). The lower

(XL) and upper (XU) limits and bin size (XBIN) must be given. The spectrum includes events in the half-open interval [XL,XU+XBIN) and the number of bins is: (XU-XL+XBIN)/XBIN.

The spectrum can be defined as INTEGER*2 (type 1) or REAL*4 (type 2). Use REAL*4 under either of two conditions: a. More than 2^{15} events are expected in certain bins. b. The spectrum is to be incremented by a fractional value. If an INTEGER*2 spectrum is incremented by a fractonal value, the value will be rounded to the closest integer value.

The spectrum can be gated by a test (See IV.2A.13 and I.V.3.B.3). Enter the test number of the test by which you want to gate the spectrum. The default value is test number 0 (zero) i.e. the spectrum is not gated by any test.

The OUPUT CONTROL option affects the output of the PRTSPC overlay which prints results on the lineprinter. The codes are given in Example 9. The maximum resolution for a one-dimensional spectrum on the line printer is 100 bins. if the number of bins is greater than 100, the full range is divided into 100 channels and the program (subroutine PLOT1D) averages over all bins within the same channel.

A default spectrum definition exists as follows: On parameter NP=1 XL=0, XU=LPARM(1) (i.e. the number of channels for parameter 1 as

determined in the INITS subcommand PARM (IV.2.A.2)), and XBIN is set so that there are 62 bins in the spectrum. The spectrum is defined as INTEGER*2 (type 1) the test number is 0 and the output control is -1,i.e. there is no output on the line-printer. Whenever one spectrum has been defined, the defaults for the next spectrum are the values of the previous one.
Note that whenever the user wants a default value to be entered between new values he or she has to enter a comma (,) instead of the value. if all values, or all those above a certain one should be the same as the default, it is enough to enter <CR> after the last new value.

After a spectrum is defined the command asks for a text (up to 40 letters) attributed to the spectrum. No default text is set.

If the user changes the spectrum definition, the change itself does <u>not</u> re-allocate all the other spectra in the spectrum buffer. So, if the modified spectrum has <u>more</u> bins than the old one, it can happen that this spectrum will overwrite other spectra. The way to rearrange the spectrum buffer is to zero it! After changing a spectrum, it is recommended to type <u>ZALL</u> in INITS. If ZALL is not typed, the program may complain that spectra defined exceed the available space. If this annoys you, type ZALL. If it persists, check the spectrum definitions.

Up to 100 one-dimensional spectra can be defined. The spectrum definitions are saved in the common D1SCOM.

8) 2DSP - 2DSP defines two-dimensional spectra. (See Example 10). All of the description of the 1DSP command is applicable here except for the following:

a. Here YL, YU and YBIN should also be entered.

b. The maximum resolution, if plotted on the lineprinter, is 120x120 channels.

c. The default spectrum definition , in case no spectra are defined, is meaningless. The default type is 1 (INTEGER*2) and the default output control code is 0 (i.e. just the total area is printed).

d. If changes are made, the final remark in 1DSP should be remembered (i.e. spectra should be zeroed if they have been re-defined.)

Up to 100 two-dimensional spectra can be defined. The spectrum definitions are saved in the common D2SCOM.

9) WCND - WCND sets window conditions (See Example 11.) WCND defines rectangular windows on a two-dimensional space. The window is determined by lower (XL), upper (XU) limits in the x-coordinate and in the y-coordinate (YL,YU). It covers

[XL,XU]X[YL,YU].

The window condition can be used to define a window for a virtual display (see IV.3A.1), or as a rectanglular gate (applied by the function IGAT2, see IV.3B.10). No default parameters are set. The effect is similar to the multiplication of two one-dimensional conditions.

The other purpose of WCND is as an aid in plotting two-dimensional virtual displays in VIRD. The effect of applying WCND to any two-parameter space is to assign the maximum available space on the screen to the limits defined by the WCND (in the parameter units, and not the screen units.)

Up to 10 window conditions can be defined. The window conditions are saved in the common WCNCOM.

10) GAT2 - GAT2 defines two-dimensional "free-form" gates. (See Example 11) Up to 50 gates can be set. Each gate is defined by up to 20 pairs of points XL_i, XU_i on the x-coordinate for different y-coordinates y_i. The y_i's should have increasing values, i.e., $y_{i+1}>y_i$, and for each i, $XU_i>XL_i$ should be true.

The user can enter default parameters for x and y coordinates and default window conditions. The default window condition is irrelevant since the absolute coordinates of the gate are used and not those with respect to the window condition. This number is just to remind the user which window condition (if any) he used when he set the gate.

Enter just ONE set of y_i,XL_i, XU_i for each line.

Notice that it is much more common to define the gates in VIRD directly on the two-dimensional virtual display. This command (GAT2) is not usually used.

The information on two-dimensional gates is saved in the common blocks GT2COM, GBOCO1, and GBCOM2.

11) LINR - LINR sents up linearizations.

On a two-dimensional space of parameter XOLD versus parameter Y two zig-zag lines can be defined. Each of them is determined by 25 points equally spaced in the Y-direction, including the lower (YL) and the upper (YU) limits.

The two zig-zag lines are mapped to two straight lines in the space of a new parameter XNEW versus the same y-parameter Y. Events in XOLD,Y space are linearly interpolated according to these lines while they are transformed to the XNEW, Y space, i.e. the ratio of the distances in X-COOR (for a point (x,y) in XOLD,Y space) from the two zig-zag lines is the same as the ratio for the (x,y) in XNEW,Y space from the two straight lines (XNEW is the linearized value of x). The equations of the two lines in the new space are X=XNEW1,and X=XNEW2, where XNEW1 and XNEW2 are constants. The user enters 26 numbers for each line; the first 25 are the x-coordinates in the X-OLD parameter, and the 26th is XNEW1(XNEW2). Enter a single x-value in each line.

As with GAT2 command, it is much more natural to define linearizations on the virtual display in VIRD.

Up to 16 linearizations can be set. The information about LINR is saved in the common LINCOM.

12) TSTX- TSTX attributes text to tests. (See Example 12.)

In the event-by-event user-supplied subroutine YFSR it is possible to define tests. (See (IV.3.B11)). A test can be any combination of one-dimensional conditions and/or two-dimensional conditions and gates etc. and can be applied to each event. These tests have counters in the program and whenever an event passes a certain test its counter is increased by one. Then the test flags are reset before analyzing the next event. The number of events which pass each test is printed on the terminal and on the lineprinter whenever the analysis is terminated. Using this command, the user can declare how many tests are defined in YFSR and give a text to any of the tests of up to 40 letters. If the first four letters in a test-text are blank the counter result for this test is not printed but the test still functions as a good logical test in YFSR, i.e., its flag is cleared and the counter is incremented (just not printed).

Up to 100 tests can be defined. The TSTX information is saved in the common TSTCOM.

13) SSTS - SSTS sets "simple" test. (See Example 12)

Since the setting of tests in YFSR requires modification and recompilation of YFSR, WC provides an additional way to define up to four simple tests which can be applied event-by-event while using the default YFSR. The building blocks (elements) of each simple test are one-dimensional conditions, two-dimensional free-form gates and already defined tests.

A simple test is a combination of one or two groups of elements. Each group consists of up to five elements, among which is a logical OR or AND operation. Then a logical AND or OR operation is performed on the two groups. The logical operation NOT can be applied to the final result. The user is asked for a test number (from 1 to 100) under which the simple test is to be saved. This completes the definition of this "simple" test which is performed by the function ISTEST in YFSR (IV.3.B.3). (The ISTEST function performs the calculation of all the simple tests defined - i.e. it sets the ITEST (n) to 1 or 0, if the test is fulfilled or not, respectively. The result of the function ISTEST itself is the multiplication of all ITEST (n)'s of the simple tests (an AND between all the simple tests).

If the number of elements in group two is zero (or <CR>) the test consists of the single group.

The program asks for the dimension of element n.

0 - means a TEST

1 - one-dimensional condition (See COND)

2 - two-dimensional free-form gate (See GAT2)

According to the user response the program asks for a test number, condition or gate number. If the element is a one-dimensional condition, the program asks for the parameter number on which the condition is to be applied. If the parameter number is set to zero, the parameter number is taken to be the default (as declared by COND).

If the dimension is two, the gate is applied on the default x and y parameters as defined by GAT2 (or by the VIRD subcommand GATE).

After the simple test is defined, a summary is printed on the terminal. Each element is marked by three numbers in brackets. The first is the dimension and the second is the TEST, COND or GAT2 number. In the case of a one-dimensional condition, the third number is the parameter number.

The simple tests are saved on the setup file. The information about SSTS is transferred to ISTEST in the common blocks STSCOM1, STSCOM2, STSCOM3 and STSCOM4.

14) ZALL-ZALL zeros all the spectra and reallocates the addresses of the spectra in the buffer. (See IV.2.A.7).

15) ZSPC - ZSPC zeros a selected group of spectra (or a single one).

16) REST - REST restores a spectrum buffer from I/O unit 8 (the default set by $WC is the scratch file SC2). A previous spectrum buffer can be copied to I/O 8 by the procedures $RESTFI or by $WC. Notice that the spectrum buffer is not restored unless this command is executed.

17) SVSP - SVSP saves the spectrum buffer on I/O unit 9. The user can save several spectrum files if he changes the assignment of I/O 9 each time. This command is automatically executed at the termination of WC.

18) SVSE - SVSE saves the setup information accumulated in INITS on I/O unit 7 (assigned to scratch file SC1 by $WC). This command is automatically executed at the termination of WC.

19) DISP - DISP switches to the DISP overlay.

20) SETE - SETE switches to the SETEXP overlay.

21) ANLD - ANLD switches to the ANLDAT overlay (the normal pattern of execution).

22) PRTS - PRTS switches to the PRTSPC overlay and enables the user to print selected spectra bin-by-bin no matter what the pre-set print controls in the spectrum definitions are.

23) VIRS - Define and change the frames of up to two setups of virtual displays. (See VIRD IV.3.A). Each setup can hold up to twelve frames of virtual displays. Each virtual display can be gated by a test (See IV.3.B.3). The program asks for the x and y parameter numbers for each frame, then for a window-condition number (IV.2.A.3). If the latter is greater then zero, the boundaries of the frame are determined by the WCND. If set equal to zero, the range is [0,4096)x[0,4096). If it is -1 (the default) the user is asked for the boundaries [XL,XU) and [YL,YU). The last question for each frame is the gating-test number. The default is 0 i.e. the virtual display is ungated.

3. ANLDAT

The prompt message of this overlay is ANLDAT>>>. The user enters ANLDAT from any overlay by typing the ANLD command and from VIRD by typing the CONT command (if continued accumulation of virtual displays is desired).

This overlay handles the event-by-event analysis of the data. It loads the user's favorite subroutine from the load-library (FIL TM in TOC) into the program. It then fills the event-buffer with an event and transfers control to YFSR. When it returns from YFSR it increments the test counters and resets the test flags. If an output tape is assigned and the event is accepted and loaded into the output-event-buffer (in YFSR) this buffer is dumped to the output tape. If any virtual displays are set, it adds the event to the display.

Interrupts in ANLDAT enable the user (if he or she has a graphics terminal) to switch to VIRD and set virtual two-dimensional displays. The list of commands is given in Example 13. The user gets the ANLDAT>>> prompt by pressing the letter A on the keyboard of the terminal on which he logged on. This interrupt is checked every 50 events (50 calls to YFSR). The list of four-letter commands is given below:

a. DISP - Go to the DISP overlay.

b. VIRD - Go to the VIRD sub-overlay and set virtual displays.

c. SETE - Go tothe SETEXP overlay.

d. INIT - Go to the INITS overlay.

e. CONT - Resume the event-by-event analysis.

A. VIRD

The prompt message of this overlay is VIRD>. The user enters VIRD from ANLDAT with the command VIRD, or by interrupting the event-by-event calculations by typing the letter V on the keyboard. It is impossible to enter VIRD unless the user has a graphics terminal.

This overlay enables the user to set up to two virtual two-dimensional displays at a time. While accumulating data in ANLDAT, it is possible to return to VIRD and set two-dimensional free-form gates and linearizations (see GAT2 and LINR in INITS (IV.2.A.10,11) using the cross-hair thumbwheels. The screen cursor is also used to retrieve the coordinates of points on the virtual displays.

VIRD has four-letter commands. (See Example 14.)

1) SETV - SETV sets virtual displays (See example 15.) The user is asked for virtual-display setup number. If one or two is entered,the applicable setup is displayed and accummulated. If zero (or <CR>, the program performs the subcommand VIRS of INIT (See IV.2.A.23) where the frames of the virtual displays are defined. Then the user returns to the VIRD prompt. The result of this command is the plotting of the frame at the display shown in Fig. 3.

2) REFR - REFR clears the screen and plots the frame again. ANLDAT continues, and the accumulation of the virtual display resumes.

3) CURS - CURS allows the user to use the screen cursor to retrieve the position of points on the screen. Use the cross-hair thumbwheels to position the cursor on the desired point. Then press any letter key on the keyboard. The point will become brighter on the screen and the coordinates will be printed on I/O device 3 (which is the "conversation" terminal).

If the coordinates of many points are required, enter a blank instead of a letter. The result is that the cursor appears again ready for the next point. (See Example 15.) Pressing a non-blank key terminates the cursor function.

4) GATE - GATE defines a new two-dimensional free-form gate. (See GAT2 in INITS (IV.2.A.10)), and can modify, delete or replot existing gates.

The response A enables the cursor so that the user can enter the gate limits by positioning the cursor and pressing a key on the keyboard. In Figure 5 a gate is shown (on the right). This gate is defined by three sets of points (three different y-coordinates). Start defining a gate with the lowest y-coordinate and continue with higher y-coordinates. The order of the points (XL and XU) for the same y is irrelevant; if the y-coordinate of the first x,y point is defined, then a repostion of the y-cursor does not matter, and for the second point just the second x-coordinate is retrieved. The program checks which x is the lower one and which is the upper one.

The user may define a gate with up to twenty sets of points, but you must use at least two sets. Whenever the user wishes to terminate the entering of gates, any letter key should be pressed while entering the last point. If a blank is entered, the program waits for the user to enter one more set (i.e. two more points) for an additional y-coordinate.

The response D to the GATE command deletes the last n defined two-dimensional gates. The program asks for n.

The reponse P to the GATE command plots existing gates on the screen. (See Example 15). The user can replot already existing gates by entering the gate number and the plot number. (If one display is set,

as in Figure 3, enter 1 or 0 for the plot number. If two plots exist, the one on the left is number 1 and the one on the right is number 2). In Figure 6 the gate number is marked at the corner points of the gate. All the gates can be plotted at once by entering zero as the gate number. If, for the plot number, the user enters zero, the plot number is picked up by the program according to the way the gates have been defined. Notice that: a. The user can either plot gates one-by-one or all of the gates. b. The position of the gate on the screen i.e. the plot number, is defined only if the gates were set in the current execution of WC, since the plot numbers are not saved on the setup file. In Figure 7 the last option is demonstrated.

5) WGAT, LGAT - These commands print on the lineprinter the coordinates of the last (WGAT) or all (LGAT) two-dimensional free-form gates.

6) LINR - LINR sets, modifies or deletes linearizations. (See Example 15 and the LINR command in INITS. (IV.3.H.11)).

On the left side of Figure 5 the setting of a linearization is demonstrated. The response A (or M) to the command enables the cursors, but just the x-cursor is read by the program.

Position the x (vertical) cursor to the lowest frame line on the proper x for the first point of line number one and press any key on

the keyboard. A dashed line will appear on the level of the next y. Position the vertical cursor on the proper x for the second point on line one and press any key to enter this point. Repeat this procedure till all 25 points of line 1 are entered. Then the program asks for the x-coordinate of this line in the new x-parameter space XNEW1. After the user enters XNEW1 the cursor reappears for the same procedure for the second line. The order of the lines (left or right first) is irrelevant. This procedure ends with the assignment of a parameter number to X-NEW, which can be the same X-OLD (i.e., the parameter will be overwritten) or any other number (less than or equal to 100).

Figure 8 demonstrates the effect of linearization. The linearization was set on the left position. The result is seen on plot number 2, where X-NEW vs. the same y is plotted.

The response D to the LINR command deletes the last defined linearizaton.

7) HCOP - HCOP makes a hard copy of the screen.

8) ANLD - ANLD returns to ANLDAT and disables the accumulation of points on the virtual displays.

9) CONT - CONT returns to ANLDAT and enables the virtual displays.

13) Final remarks on VIRD.

If the user enters VIRD, all commands except for ANLD, SETV, REFR and CONT leave the program under VIRD control. The way to return to ANLDAT is to type ANLD or CONT. If no virtual displays are going to be incremented in ANLDAT, enter ANLD. If accumulation is to be resumed, use CONT.

SETV and REFR return control to ANLDAT after plotting the frame. If the user entered VIRD using the VIRD command under ANLDAT>>>, the last prompt will reappear, and in order to resume data accumulation, CONT should be entered in ANLDAT. If the user entered VIRD by pressing the key V during the event-by-event session, the accumulation is resumed without any further command in ANLDAT.

If ANLD was entered by mistake, enter VIRD in ANLDAT then CONT in VIRD; accuisition will be resumed. The actual sorting of data on the virtual displays is done by the subroutine D2SORT under the overlay ANLDAT. The user can return to ANLDAT by ANLD and increment each of the virtual displays separately in YFSR using the routine D2SOR1 (See IV.3.B.13).

B YFSR - Your Favorite SubRoutine

This overlay is of great importance since it can be modified by the user to accomodate the analysis of specific experiments. Its name expresses its flexibility. A space of 7600 INTEGER* 4 words is devoted to YFSR. This space can be changed by changing the dimension of LDBUF (7552) in ANLDAT.

In Appendix 1 a list of the default (or "skeleton") subroutine is given.

Since all WC subroutines are already compiled, the user has only to compile YFSR and link it with the rest of the system, and save a short load-module on his load library (FIL TM in TOC). This is done by the procedure $ANALY as described in (III.3). Assume a load-module of your favorite subroutine has been catalogued on file TM under the name MYOVERLY. The first time the program enters ANLDAT, or when ANLDAT is entered with the staus code IPASS=1 (see remarks in SWOV command in INIT (IV.2.A.1)), the user is asked for the name of his load module (MYOVERLY), and for the file or which it is cataloged (TM). Then his favorite subroutine is loaded. Note that with this technique, one can easily switch between different analayzing routines.

Subroutine YFSR has three parameters. The first one, NPASS is an input parameter set by ANLDAT. The first time the user enters ANLDAT

after using the SE command in SETEXP (when IPASS=1, See Section (IV.2.A.1)), NPASS is set to 1 and YFSR is called. Following the list of the skeleton YFSR in Appendix 1, one can see that for NPASS=1 the program jumps to statement label 9000

In the opening section of the skeleton YFSR we set the following things: NCT- a counter which increases by one every event and has a value betwen one and NDBMAX- is set to its initial value. This counter is used if, in the SE command (in SETEXP), the user asks for DEBUG OPTION=1. (See (IV.1.A.3).) This causes the printout of each event buffer every NDBMAX events.

If no output tape is assigned (IANS4=N) IPARO - the maximum number of parameters on the output tape - is set to zero.

In case the user is running a simulation code, IANS5 is set to Y (Yes). (IANS5 is set to Yes if no input tape was assigned, or if the number of runs (NRUNS) was set to a negative number under the SE command in SETEXP (IV. 1. A.3)). In this case ANLDAT has to "know" when to end the program. The way to do it is to check, in the event-by-event step, the event counter (NCOUNT). If there are sufficient events, "tell" the program to terminate the event-by-event procedure. This is done by returning to ANLDAT by the RETURN 1 command, and this is the reason for the third parameter in YFSR - the asterisk * . In a simulation code, the RETURN 1 command should be moved to the event-by-event section.

In this opening part, the user can add any steps which he wishes to do just one time. We refer to this as "the first call to YFSR."

After all the runs (NRUNS) have been analyzed (IPASS=5), ANLDAT sets the input parameter NPASS to -1. In this case it is seen that the program jumps to statement label 9900. This section of the program is the closing part of the program-the last call to YFSR. This is the place where the user can make operations on the spectra accumulated during the analysis and print out special results and comments just the way he likes.

Notice that if the user returns to SETEXP after this, and enters the SE command again, the first call to YFSR will be executed again. If SO command is used, the beginning part will be skipped but the closing part will be called again at the end of the runs requested by SO. ANLDAT prompts the user with the question whether to perform the last call to YFSR. So the last call can be skipped until the user finishes all his analysis, if he so desires.

After the first call to YFSR, ANLDAT proceeds to the event-by-event section. In this part the following things are done in ANLDAT. The parameter NPASS is set to zero (and it is reset to zero only if the program switches from ANLDAT to another overlay and back to ANLDAT). After that:

a. An event is loaded into IVI(100)-the input event buffer. (IVI is an INTEGER*2 array.)

b. IPARI (the maximum number of parameters per event on the input tape) parameters are copied to the "working event buffer," EVENT(100). EVENT(i)=FLOAT(IVI(i)), for all i, $1 \leqslant i \leqslant$ IPARI. Both buffers are available to the user in YFSR via the common EVNCOM.

Steps a. and b. are skipped if IANS6=Yes (i.e. a simulation code is being run and/or there is no input tape).

c. YFSR is called.

Now let us return to the YFSR listing; we see that now the event-by-event section is entered (since NPASS is neither +1 nor -1). First we print the input event buffer if the debug option was requested in SETEXP.

Then we see that the second parameter of YFSR, i.e. NSKIP, is set to zero. NSKIP is an output parameter of the subroutine which enables the user to filter out events according to the tests done in the analysis. If NSKIP is set between zero and ten the event is accepted, i.e. the event is dumped on the output tape. If NSKIP is set to a negative number between -1 and -10, the event is rejected.

Two arrays exist in ANLDAT - NACC(10) and NREJ(10). On the return from YFSR to ANLDAT these arrays are incremented in the following way:

value of NSKIP	result
$1 \leq NSKIP \leq 10$	NACC(NSKIP)=NACC(NSKIP)+1
NSKIP=0	no array is incremented, but event is accepted
$-10 \leq NSKIP \leq -1$	NREJ(-NSKIP)=NREJ(-NSKIP)+1

These arrays are cleared just after the first call to YFSR (when NPASS=1, as described above) so they provide us with scalers for different cases of acceptance and rejection of events. These scalers are printed every time the program switches from ANLDAT to SETEXP. (See Example 20.)

1) PSPR - A call is made to PSPR. This subroutine calculates all the pseudo-parameters that have been determined in the INITS command PSPR (IV.2.A.6.). The pseudo-parameters are calculated one after the other, according to their pseudo-parameter numbers (and not their parameter numbers).

2) LINR - CALL LINR performs all linearizations, one after the other according to the order in which they were defined.

3) ISTEST - The "simple tests" defined in the SSTS command of INITS (IV.2.A.13) are carried out. ISTEST(NDUMMY) is a function which has a dummy integer parameter (0 in our case). The output of this function is 1 if the event passes all the simple tests (or if no simple test has been defined) and 0 if it does not. In the latter case we set NSKIP to -1, i.e. this event is to be rejected.

4) INCSPC - The call to INCSPC increments all the one-dimensional and two-dimensional spectra (defined using the 1DSP and 2DSP commands in INITS (IV.2.A.7,8)) by 1 (one). This command is called after ISTEST since the spectra can be gated by tests which have been defined interactively as "simple-tests".

Note that all four operations PSPR, LINR, ISTEST and INCSPC use the working event-buffer EVENT(100) and not IVI(100), since the new parameters produced can have any value and are not restricted to integer ones.

The next-to-last step is to copy the event buffer to the "output event-buffer," IVO(100). This buffer is filled when an output tape is produced by loading IPARO parameters (the maximum number of parameters in the output tape) into this INTEGER*2 buffer from the working event-buffer EVENT(100). The IVO buffer will be dumped on tape only if NSKIP$\geqslant$0, (and output tape was requested in SETEXP subcommand SE).

In the skeleton subroutine we load the first IPARO parameters from EVENT into IVO, but this is of course subject to user changes.

The last step in the event-by-event procedure is to print the output buffer IVO in case the debug option is requested in SETEXP. After this step, control returns to the overlay ANLDAT.

The event-by-event part, as well as other parts in YFSR,is usually changed by the user in order to suit his or her own analysis requirements. Some general tools are provided by WC in order to simplify the programming.

Sometimes it is not convenient to calculate all pseudo-parameters, then all linearizatons, etc., but to calculate selected numbers of pseudo-parameters, then make a single linearization then calculate additional pseudo-parameters using the linearized x-coordinate X-NEW, etc. For this reason, the following subroutines exist:

5) PSPR1 - CALL PSPR1(N) calculates only the single pseudo-parameter number N (N is its sequential number in the pseudo-parameter list).

6) LINR1 - CALL LINR1(N) performs linearization number N.

7) INC1DS - CALL INC1DS(NS,DEL) increments the one-dimensional spectrum number NS by the amount DEL. DEL can be any number (not just an integer). If the spectrum NS is gated by a test number n, the spectrum is incremented only if ITEST(n) $\neq$0 (See 11. for ITEST).

8) INC2DS - CALL INC2DS(NS,DEL) increments the two-dimensional spectrum number NS by the amount DEL. If the spectrumn NS is gated by a test number n, the spectrum is incremented only if ITEST(n)$\neq$0.

Writing your own YFSR eliminates the restriction to using just four "simple tests". The user's own "complex tests" can be set using the following functions (not subroutines):

9) ICOND - ICOND(NP,NC) checks if parameter NP fulfills the one-dimensional condition number NC, i.e. if XL$\leq$EVENT(NP)$\leq$XU, where XL and XU are the limits defined in the command COND in INITS (IV.2.A.5) for condition number NC. If NP=0 the default parameter number is used. The result of this test is ICOND=1 if the event falls inside the condition limits (TRUE) and ICOND=0 if it does not (FALSE). Notice that we test the closed interval [XL,XU]: the condition includes the boundaries.

10) IGAT2 - IGAT2(NPX,NPY,NG2,NWC) checks if parameters NPX and NPY (EVENT(NPX), EVENT(NPY)) are inside the two-dimensional free-form gate number NG2. The gate includes its boundaries. (Free form gates are

defined by using GAT2 in INITS (IV.2.A.10) or the command GATE in VIRD (IV.3.A.4). If NPX and/or NPY are zero, the default parameters x and/or y are taken for the gate definition. If NG2 - the gate number, is zero then the function checks if EVENT(NPX), EVENT(NPY) are in the rectangular window condition number NWC (defined in INITS by the WCND command (IV.2.A.9)). If NG2≠0 the window condition number NWC is ignored. IGAT2=1 if the result is TRUE (inside the gate) and IGAT2 = 0 if the result is FALSE (outside of the gate limits).

11) ITEST - Now we are ready to combine these functons into a complex test. OR logic is done by summation of these functions and AND logic is done by multiplication of these functions. The result of any such combination is either zero or non-zero. A zero result means that the test is not fulfilled (FALSE) and a non-zero result means that the test is fulfilled.

The test information is saved in the common TSTCOM. The number NTST is the number of tests to be calculated. (This number can be defined in INITS using the command TSTX (IV.2.A.12)). A test is established by ITEST(n)=(Any complex combination of ICOND, IGAT2 etc.)* The user can check the value of ITEST(n) in order to make a decision about the event tested. However, when the program returns to ANLDAT, the element n in the NTEST buffer NTEST(n) is incremented by 1 if ITEST(n)≠ 0, whether the event is accepted (NSKIP≥0) or rejected (NSKIP<0). Later, all the NTST tests are cleared i.e. ITEST(n) = 0.

---

*e.g. ITEST(1)=(IGAT2(22,23,9,1)+ICOND(3,2))*IGAT2(22,23,20,1)

OR AND

Be sure that the number of tests NTST is determined either in YFSR or by TSTX command in INITS. Otherwise the tests are not cleared in ANLDAT except for the simple tests which are cleared and set by the ISTEST function). Text can be attributed to the tests (using the command TSTX) so the test scalers, NTEST, with their text are printed whenever the program returns to SETEXP (See Example 20.) These scalers (NTEST) are cleared whenever the initial part of YFSR is executed (NPASS=1). This is usually a consequence of using the SE command in SETEXP.

In each event, after NTEST is incremented, all of the ITEST buffer is set to zero.

12) Random Number Generators - If a simulation code is run or the user wishes to smooth the data using a random number generator, two kinds of random number generators are available:

a. The internal MODCOMP funciton RANX(NDUMMY) produces uniformly distributed numbers between 0 and 1.

b. The WC function GAUSS(S,W) produces normally distributed numbers around a mean value W with a standard deviation S. The error function taken in GAUSS covers up to five standard deviations.

The tools discussed above are useful for event-by-event analysis. In the last call to YFSR (NPASS = -1), operations on spectra can be performed which are discussed below.

13) D2SOR1 - CALL D2SOR1(NV) adds the event to virtual display number NV (NV=1 or 2). If virtual display NV was not defined (by SETV in VIRD) nothing happens.

Note! If D2SOR1 is used, return to ANLDAT, after defining the virtual displays in VIRD, by the command ANLD and not by CONT. CONT causes ANLDAT to increase all virtual displays if the reutrn code NSKIP of YFSR is qreater than or equal to zero. ANLD stops the accumulation and enables the user to have direct control of the virtual display via YFSR.

14) NRMSPC - CALL NRMSPC (ND,NS1,ADD,DIV,NS) normalizes ND-dimensional spectrum number NS1 (ND=1 or 2) and puts the resulting spectrum into spectrum number NS. NS can be equal to NS1. (NS is referred to as the "TARGET SPECTRUM.") The spectrum is normalized thusly:

$$S(i) = \frac{S_1(i) + ADD}{DIV}$$

S(i) is the content of bin i in spectrum NS.

$S_1(i)$ is the content of bin i in spectrum NS1.

15) SQRTSP - CALL SQRTSP(ND,NS1,NS) takes the square root of the content of every bin in the ND-dimensional (ND=1 or 2) spectrum NS1 and deposits the result in spectrum NS, that is,

$$S(i) = \sqrt{S_1(i)}$$

Negative numbers are set to zero.

16) PWR2SP - CALL PWR2SP(ND,NS1,NS) takes the square of a spectrum. Thus:

$$S(i) = S_1(i)^2$$

(ND,NS1,NS are the same as for SQRTSP)

17) DIVSPC - CALL DIVSPC(ND,NS1,NS2,NS) divides two spectra. Thus

$$S(i) = \frac{S_1(i)}{S_2(i)}$$

S(i) is the content of bin i in spectrum NS.

$S_1(i)$ is the content of bin i in spectrum NS_1.

$S_2(i)$ is the content of bin i in spectrum NS_2.

Note that if $S_2(i)=0.$ than $S(i)=0$, not infinity.

It doesn't cause an error or an overflow.

18) MULSPC - CALL MULSPC(ND,ND1,NS2,NS) multiplies two spectra. Thus,

$$S(i) = S_1(i)*S_2(i).$$

19) ADDSPC - CALL ADDSPC(ND,NS1,NS2,NS) adds two spectra. Thus,

$$S(i) = S_1 = S_1(i) + S_2(i).$$

20) AVRSPC - CALL AVRSPC(NS,PL,PU,AVR,SIGMA,SUM) calculates the average value (AVR) standard deviation (SIGMA) and the integral ($\sum_i S(i)$) over one-dimensional spectrum number NS between the limits PL and PU. The subroutine takes an integer number of spectrum bins totally included between these limits, and does not correct for cases where PL or PU falls in the middle of a certain bin.

21) DEF1DS - CALL DEF1DS (NP,XL,XU,DX,NTYP,NS) defines a new one-dimensional spectrum. The output control is set to -1 i.e. no printout is made on the lineprinter, but space is assigned for this spectrum (See the 1DSP command in INITS (IV.2.A.2)). NP is the parameter number, XL, XU and DX are the lower, upper limits and the

bin size respectively. NTYP = 1 or 2 for INTEGER*2 or REAL*4 respectively. The subroutine allocates space for the spectrum in the spectrum buffer and the output parameter NS is the number of the new spectrum.

22) DEF2DS - CALL DEF2DS (NPX,XL,XU,DX,NPY,YL,YU,DY,NTYP,NS) is similar to DEF1DS but it defines a new two-dimensional spectrum. NPY,YL,YU, and DY are the parameter number, lower and upper limits and bin size for the y-coordinate. (See the 2DSP command in overlay INITS (IV.2.A.8).)

23) GET1DS - CALL GET1DS (NS,AREA,VECTOR,NBINS) "gets" a one-dimensional spectrum. This subroutine copies the content of the one-dimensional spectrum number NS into an array VECTOR. This VECTOR must be declared in YFSR in a DIMENSION statement or established via a common statement. The subroutine also supplies the output parameters AREA and NBINS. AREA = $\sum_i |S(i)|$, the sum over all bins, and NBINS is the number of bins in the original spectrum and of course in the array VECTOR.

24) GET2DS - CALL GET2DS(NS,AREA,VECTOR,NBINX,NBINY) is similar to GET1DS but copies two-dimensional spectrum number NS to VECTOR. NBINX and NBINY are the number of bins in the x- and y- parameters respectively.

The user can now perform <u>any</u> calculation on the array VECTOR, and then deposit the result back in the spectrum buffer with the following subroutines:

25) <u>DPS1DS</u> - CALL DPS1DS(NS,VECTOR) copies the array VECTOR to the space assigned to one-dimensional spectrum NS.

26) <u>DPS2DS</u> - CALL DPS2DS(NS,VECTOR) is similar to DPS1DS but for a two-dimensional spectrum.

27) <u>GET2DP</u> - CALL GET2DP(NS,NPRJ,XL,XU,YL,YU,AREA,VECTOR,NBINS) allows the user to project a rectangular window defined by XL,XU,YL and YU on the two-dimensional spectrum number NS on either of the axes. NPRJ is the parameter number of the axis on which the spectrum is to be projected; NPRJ is either NPX or NPY. GET2DP takes an integral number of bins in the x- and y- parameters which are totally included in the window's limits. The sum of the absolute values of all the bins' contents, $\sum_{i,j} |S(i,j)|$, is given by AREA. The projected spectrum is deposited in the array VECTOR (see the note about GET1DS) and NBINS is the number of bins in the projected parameter copied to VECTOR. If any of the limits XL, XU, YL or YU is out of the range covered by the spectrum definition, the limit of the parameter is taken from the spectrum definition. The limits of the projection will then be the intersection of the region defined by [XL,XU]x[YL,YU] and the region declared in the spectrum definition.

28) COMBIN - CALL COMBIN (NS1,NS2,NCOMP,KBINS,DX) compresses a 1-D spectrum (given by NS1) by a factor NCOMP. The first NCOMP bins of spectrum NS1 go into the first bin of spectrum NS2, the second NCOMP bins go into the second bin of spectrum NS2 etc. If NCOMP <2, spectrum NS1 is put directly into NS2. NS1=NS2 is allowed, and if NS2 is not within the range of currently defined spectra, a new spectrum is defined, and the compressed spectrum is stored in it. In this case, the spectrum number of the new spectrum will be determined by the number of spectra already defined, and so may not be equal to NS2. In any case, NS2 is not altered by the subroutine. KBINS is the number of bins in the output spectrum, and DX the size of a bin in the output spectrum in physical units. If the number of bins in the original spectrum is not an integer multiple of NCOMP, KBINS will be rounded up, so that the total number of counts in the final spectrum will equal the total number in the initial spectrum. So the number of counts in the final bin of the compressed spectrum may be the sum of fewer than NCOMP bins of the initial spectrum. If a new spectrum is created, the new spectrum will be of the same variable type (integer or real) as the original spectrum. Finally, while using COMBIN in the overlay DISP the compressed spectrum is left in the plotting buffer (the array DUMMY in the COMMON DUMCOM), but the virtual window is not changed (and therefore may not be appropriate to plot the compressed

spectrum). Since we can copy the content of a spectrum to an array and deposit it back into the spectrum buffer, WC provides the user with array operations.

Array Operations:

29) SMOOTH - CALL SMOOTH(A1,A2,NPOINTS,NBINS) smooths the array A1, with NBINS equal to the number of bins, by least square fitting, using NPOINTS. NPOINTS ranges from 3 to 25 and must be an odd number. The first and last (NPOINTS+1)/2 bins of A1 remain unaffected. The result is deposited in the array A2. (A2 can be A1). (For reference see note in DSMOTH).

30) DSMOTH - CALL DSMOTH(A1,A2,DX,NPOINTS,NBINS) smooths and differentiates the array A1. A2, NPOINTS and NBINS are the same as in SMOOTH. DX is the bin size (i.e. the derivative is taken with respect to x). Notice that DSMOTH is not a two-step process of smoothing, then differentiating, but a one step, improved algorithm.

For details about SMOOTH and DSMOTH, see Analytical Chemistry, Vol. 36., No. 8,July 1964, p 1627.

31) VMIN - CALL VMIN(A,NBL,NBU,AMIN,NBMIN,NBINS) returns the minimum value AMIN of the array A with NBINS, located between or in bin number NBL and bin number NBU. NBMIN is the bin number where the minimum

value AMIN is located. A(NBMIN)=AMIN. If the value of AMIN is found for several bins, NBMIN is the first to be found.

32) VMAX - CALL VMAX(A,NBL,NBU,AMAX,NBMAX,NBINS) returns the maximum value AMAX of the array A. NBL,NBU, and NBINS are as in VMIN.

If the value AMAX is found for several bins, NBMAX is the first to be found.

33) CURFIT - CALL CURFIT (X,Y, WEIGHT, YFIT, NPTS, ND, NTERMS, A, DELTAA, SIGMAA, FLAMDA, CHISQR, NFIT,NCH1, NCH2, FCURVE, FDERIV) performs a χ^2 fit to the data in array Y by conducting a search of parameter space. X is the array of data points for the independent variable, Y is the array of data points for the dependent variable, WEIGHT is the array of statistical weights assigned to the data (usu. the reciprocal of the variance of each point), YFIT is the array of values of the fitted function at each value of X. NPTS is the number of data points over which the fit is to be performed. ND is the total number of bins in YFIT. NTERMS is the number of parameters in the fitting function (NTERMS $\leq$ 10). A is the array of parameters, and is both an input and an output. DELTAA is an array of parameter increments which is used for numerical evaluation of the partial derivatives of the fitting function by the parameters. SIGMAA is the array which contians the standard deviations of the parameter values. FLAMDA is a number which determines the extent to which a gradient

search is mixed with the linearizaton search. It should end up smaller than it started. CHISQR is the reduced χ^2(or χ^2 per degree of freedom). NFIT is the number of iterations required by the search. The search will be terminated automatically after 200 iterations (sooner, of course, if χ^2 reaches a stable value in parameter space). NCH1 and NCH2 are the bin numbers of the lowest and highest bins included in the fit (NPTS=NCH2-NCH1+1). FCURVE is the name of the fitting function, and FDERIV is the subroutine which evaluates the partial derivatives of FCURVE by the parameters, A. The subroutine CURFIT is taken from Data Reduction and Error Analysis for the Physical Sciences by P. R. Bevington (1969), and a complete explanation of the methods used in the subroutine can be found there.

34) LEGFIT - CALL LEGFIT (THETA,Y, WEIGHT,YFIT,NTERMS,MODE,NPRIOR, NCH1,NCH2,A,SIGMAA,CHISQR,NFIT,NRSN,P) performs a χ^2 fit of the data to a linear combination of Legendre polynomials. Since the fitting fuction, $f(\theta) = \sum_{\ell} a_{\ell} P_{\ell}(\cos\theta)$ is linear in the parameters a_{ℓ}, the set of parameters which minimizes χ^2 is the solution of a system of linear equations. The solution is found directly by elementary methods, so a search of parameter space is not necessary, as it is in SUBROUTINE CURFIT.

THETA is the array of angles (in degrees), and Y is the array to be fitted by $f(\theta)$. Y, WEIGHT, YFIT, NCH1, NCH2, CHISQR, and NFIT are as

in CURFIT. NTERMS is the number of parameters to be used in the fitting function. Exactly which polynomials are to be used in the fitting function is specified in the array NPRIOR. NPRIOR contains the ℓ-values to be used in the fit. (The fit always includes ℓ=0, so NPRIOR(1) = 0. The subroutine never looks at NPRIOR(1), since it assumes it to be 0, but it is convenient to set NPRIOR(1) = 0 for purposes of output. See, e.g., the output section of the Legendre polynomial fit in SUBROUTINE SPCOPR.) NPRIOR(2) to NPRIOR(NTERMS) are integers from 1 to 20. Polynomials of order greater than ℓ=20 may not be used. If NTERMS = 1, only the ℓ=0 term is used.

The subroutine starts by fitting ℓ=0 to the data. It then adds additional ℓ's to the fit, in the order specified in NPRIOR, either until all NTERMS polynomials have been included, or until the fit becomes impossible to perform. The reason for the fit being terminated is given by NRSN. If NRSN=1, the fit was terminated because the system of linear equations could not be solved (the determinant vanished). If NRSN=2, the number of degress of freedom is no longer positive. If NRSN=3, the χ^2 was negative or zero. (If NRSN is 1 or 3, someting funny is going on. These possibilities were checked to avoid Fortran errors, and should never occur in practice). If NRSN=2, you're trying to fit more parameters than you have data points. If NRSN=5, all NTERMS polynomials were included in the fit. The number of terms included in the fit is given by NFIT. Therefore, the number of degress of freedom is NCH2+1-NCH1-NFIT.

MODE is an input parameter which determines how the standard deviations of the parameters are calculated. If mode is nonzero, SIGMAA is the square root of the diagonal elements of the covariance matrix (denoted by ε in Bevington's book). This is the correct procedure if the array WEIGHT is the reciprocal of the squares of the errors in the data. If one wishes to weight all points equally, one usually loads the array WEIGHT with the value 1. In this case, the SIGMAA should be multiplied by $\sqrt{\chi^2}$,since the calculated χ^2 is proportional to the errors on the data. So, if one decides to use equal weights, load one's into WEIGHT and set MODE = 0. This will ensure that the SIGMAA make sense.

A and SIGMAA are almost the same as in CURFIT, except that no starting values for A are required. Also, the ℓ-value associated with a given element of A is determined by NPRIOR. Hence, A(1) is the coefficient of P_0. If NPRIOR(I)=L, A(I) is the coefficient of P_L. SIGMAA(I) is the error in A(I).

The parameter P is a one-dimensional array which must be large enough to hold the Legendre polynomials for the points included in the fit. Therefore, its size must be at least (NTERMS-1)*(NCH2+1-NCH1) words of

REAL*4. The contents of P are usually of no interest to the user. P is simply a working space for the subroutine. It was included as a parameter so that this space could be allocated in the calling routine. It holds all of the Legendre polynomials specified in NPRIOR, evaluated at all of the points included in the fit. $P_0(\cos \theta_i)$ is <u>not</u> stored, since it has the constant value 1. That's why the space required is (NTERMS-1)*NPTS (where NPTS=NCH2+1-NCH1). The calling routine should check to see if sufficient space exists.

Legendre fits may be performed by invoking the OC directive "CURFIT", specifying "LEGENDRE" as the type of fit.

Printout Services

35) <u>PMAT</u> - CALL PMAT(A,TXT,XL,XU,DX,NCNTL,NBINS,NSUM) prints the array A bin by bin (NCNTL=1) or the logarithm of array A bin by bin (NCNTL=2). The cumulative sum ($S(i)=\sum_{j=1}^{i} A(j)$) can be printed (NSUM=0) or not (NSUM=1). If the bin number is correlated with some physical value x, the lowest value (XL), the highest value (XU) and the bin size DX can be given. Then the physical quantities corresponding to each bin will be printed to the left of the contents of that bin. Otherwise the user should set XL=1, XU=RNBINS, and DX=1.

(RNBINS=FLOAT(NBINS)). A text consisting of four letters can be given to identify the meaning of x. Note that the content of the array A is changed if NSUM=0.

36) PARR - CALL PARR (A,TXTX,XL,XU,DX,NBINSX,TXTY,YL,YU,DY,NBINSY,NCNR) prints the two-dimensional array A. The parameters are the same as for PMAT, but values are assigned for the second dimension as well.

The array operations are used by the overlay PRTSPC to print results on the lineprinter. Examples 21, and 23 show the formats of PMAT and PARR respectively. No limitations on the dimension (i.e. the number of bins) are imposed on these routines.

Many of the spectrum and array operations can be done interactively in overlay DSP1 using the subcommand of the DSP1 command OC. (See Section IV.4.A.22).)

37) Final Important Remarks on YFSR

a. Note the IMPLICIT INTEGER*2 (I) statement. This is a convention which is used in most (but not all) of the subroutines in WC. But all integer input and output parameters of the subroutines and functions described in this section are INTEGER*4. None of the input parameters of these subroutines and funcions, besides PMAT and CURFIT (See note in PMAT and CURFIT - (IV.3.B.30)), are modified in these subroutines.

As a result, the user can use numbers as well as expressions for input parameter values for all subroutines and functions.

b. If the user needs a buffer to accumulate his own arrays or for array operations, a big buffer is provided by the common statement COMMON /DUMCOM/ DUMMY (8424).
This buffer can be used through all the event-by-event analysis. However, WC uses this buffer in most of the spectrum manipulations (SQRTSP,PWR2SP,MULSPC,DIVSPC, ADDSPC and NRMSPC) and as a plotting buffer in the DSP1 and PRTSPC overlays. Before using these subroutines, the user can freely modify, read and write on this buffer.

c. All the common blocks in Appendix 1 are essential in order to avoid unexpected errors, due to the overlay structure of WC. More common blocks can be added. Note the common CNSCOM. This common transmits to YFSR the constants defined with the INITS command CNST (IV2.A.4).

Important! Because of the overlay stracture, only the values of variables saved in the WC common blocks listed in Appendix 1 are saved while switching from one overlay to another. Variables initialized by data statements will be reinitialized whenever the user jumps out from ANLDAT and back! In order to overcome this problem one can write all his or her special variables on a file, change the NPASS parameter of YFSR (from zero to any number different from 1). Then,

every event one may check if NPASS is equal to zero or not. If it is zero, that means the program has returned to ANLDAT from another overlay and it is time to read back all the variables from the file and set NPASS to NPASS≠0 again.

4. DISP

The prompt message of this overlay is DISP>>>. DISP is not in the normal flow of the program and the user has to ask for it explicitly by typing DISP while in the overlay INITS or ANLDAT. DISP invokes the overlays DSP1 and SPCOPR.

DSP1 provides a versatile way to plot one-dimensional spectra on a graphics terminal or on the TEKTRONIX plotter. SPCOPR enables the user to make operations on spectra and to modify the spectrum buffer.

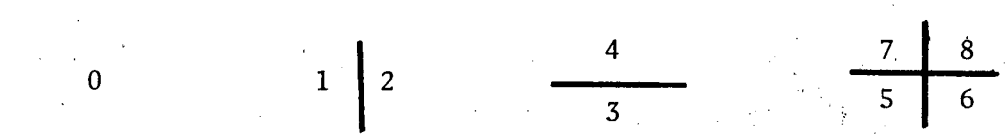
DSP1 uses the common array DUMMY in the comon DUMCOM as a "plotting buffer."

DSP1 has two-letter commands, SPCOPR the spectrum operation overlay, has six-letter commands. A list of commands is given in Example 16. To switch from DISP (or DSP1) to SPCOPR enter the command OC. To switch back hit the carriage return.

A. Command Description:

1) SW - SW sets screen window. (See Example 17). SW limits the plotting area available on the screeen to points between X-LOW, Y-LOW and X-UP, Y-UP. The extreme limits are 0,0, and 4096, 3200, respectively.

2) PO - PO selects one out of nine pre-assigned screen windows. the codes of these windows are as follows:



full screen half screen half screen quarter screen

3) VW - VW sets virtual windows. It maps the screen window into a virtual window. i.e. the limits x and y are in the user's favorite units. For example, if a differential cross secton is plotted, the X-LOW and X-UP might be mapped to -60° and $+60^{o}$ and Y-LOW, Y-UP might be set from 0 to 10^{-4}mb/sr. The part of the plotted spectrum outside of the window is chopped off. (See Example 18.)

4) SP - SP selects a single spectrum to be plotted. The spectrum is copied from the spectrum buffer into the plotting buffer. To specify the spectrum, one enters from one to three numbers. The first number entered is the spectrum-number of the spectrum containing the y-coordinates. The second and third numbers are optional. The second

number is the spectrum number of the error-spectrum. If it is zero, no error spectrum is defined (and no error bars plotted by the PL command). If it is less than zero, the errors are taken to be the square-roots of the y-coordinates. If it is greater than zero, the errors are taken from the specified spectrum. The third number is the x-spectrum number. If it is less than or equal to zero, the x-coordinates are taken from the definition of the y-spectrum. If it is greater than zero, the x-coordinates are taken from the specified spectrum and matched bin-by-bin to the y-coordinates. This feature was included so that one could have spectra with arbitrary spacing of x-coordinates. Such spectra will usually be generated in YFSR or entered through the OC command MODSPC. The x,y and error spectra are saved as NS,NXS and NES respectively in the common. The program checks the minimum (YL) and the maximum (YU) values of the spectrum and prints these values and the total area (area = $\sum_{i=1}^{ND} |Yi|$, ND is the dimension, i.e. number of bins in this spectrum). If the spectrum is empty, i.e. the area is zero, it returns to DSP1>>>. If it is not empty, the user is asked if he wishes to modify the virtual window according to the present XL,YL and XU,YU. The default answer is Yes. The No answer leaves the virtual window unchanged. The latter option is frequently taken, so that one may display several spectra one on top of the other, or inspect a limited portion of the spectrum. This command does not plot the spectrum.

5) FR - FR plots the frame and labels.

6) PL - PL plots the spectrum which is in the plotting buffer, and also plots the error bars if the error-spectrum number NES was entered as a second parameter in SP.

7) ES - The user selects the spectrum number which contains the error for the spectrum currently deposited in the plotting buffer with this command. ES copies the error spectrum into the plotting buffer and adds the error bars to the plot. (See Example 18 and Figure 13). If no error-spectrum number is entered, statistical errors are assumed, i.e. $\sigma_i = \sqrt{Yi}$.

8) PG - PG plots up to twelve consecutive spectra. It skips empty spectra. The last spectrum plotted remains in the plotting buffer. The screen window and virtual window are those of the last plot until the user changes them. This can be done by choosing the "current spectrum" to be any of those displayed on the screen by the command SG. The program will ask you which of the displayed spectra you choose, replot it and return to DSP1 prompt. (See Example 17 and Fig. 7.)

9) LL - LL switches between linear (Code=0) and logarithmic (Code=2) scales. If the spectrum has negative numbers for all y-values, the code is reset to zero (linear scale) by the program (See Example 17). The assignment stays unchanged till the next execution of the LL command.

10) HD - HD switches between histogram and dot modes. If dot mode is chosen, the spectrum points are marked by a symbol chosen by the user. (See Example 18.). Fig. 14 shows two plots, one on top of the other, in dot mode with different symbols.

11) MW - MW marks windows, enables the cursor (only the vertical (x) cursor is used) to set two limits on the current plot. These limits are marked by lines, on top of which the window number is printed. At the mean value of x, the mean value of x and the integral of the spectrum over the window are printed on the terminal. (See Fig. 7).

Many window-markers can be set on the same spectrum. If a blank is entered when the user positions the cursor, then more windows are expected. If any letter key is pressed, the mark window session is terminated.

12) WD - WD changes the virtual window (only for the x-coordinate) to the limits given by the last window marks. The screen is erased and the spectrum within the window is plotted. The plot-buffer is not changed. See Fig. 10 for the window marked in fig. 9. The same window is plotted after entering the SW command in Fig. 9, and after entering the LL command in logarithmic scale in Fig. 12.

13) CU - CU gets the cross-hair position. (The cursor is enabled.) The user positions the vertical cursor at any desired position on the current plot. By pressing any key on the keyboard, the user can get the location of the cursor printed on the terminal. If a blank is entered, the cursor is enabled again for retrieving additional positions.

14) LB - LB modifies the label on the x-axis, the spectrum label and the y-label. (The label on the y-axis is not plotted.)

15) DV - DV switches plotting between the TEKTRONIX 4014 and the DIGITAL TEKTRONIX PLOTTER.

16) HC - HC makes a hard copy of the screen.

17) ER - ER erases the screen.

18) CL - CL erases the screen and sets the screen window to the full screen size and the virtual window to its default values (not of any importance).

19) AN - AN goes to the ANLDAT overlay.

20) IN - IN goes to the INITS overlay

21) SE - SE goes to the SETEXP overlay

22) OC - OC switches to the spectrum manipulation mode (Overlay SPCOPR). The prompt message is ENTER OP-CODE::. A list of the six-letter commands is given in Example 16.

23) PF - PF plots the last fit performed by CURFIT command in SPCOPR.

24) SC - Stores the last marked window as a condition (1-D gate) (See IV.2.A.5). The default parameter number is the parameter on which the window was defined.

25) MC - Marks a window on the plotted spectrum using the limits of a 1-D condition (See IV.2.A.5)

All operations on the spectra are done only on the one-dimensional ones, but one can project a two-dimensional spectrum on any of its axes with the command GET2DP, plot it and/or use the command DPS1DS to deposit the projection in the spectrum buffer as a one-dimensional spectrum. (DPS1DS loads the plotting buffer into the spectrum space in the spectrum buffer). Furthermore, the user can define more one-dimensional spectra by typing DEF1DS and use them to store the projections in order to perform other operations on them.

All the OC subcommands (which are listed in example 16) make use of the subroutines with the same names as those described in Section IV.3.B(14-28) in YFSR. Therefore, no further comments are given on these subcommands. If curve-fit (CURFIT command) is used, standard fits to Gaussian, Maxwellian and Legendre polynomials are built into the program. If the user wants to add his or her own special function, the following steps should be followed:

a. The user function has to be called FUNCTION FCURVE (X,N,A,NFIT). DIMENSION X(1), A(1) command must be defined. x is the array of the function's argument and X(N) is the x for which y=FCURVE(X(N)...). A is the array of the parameters to be fitted. Up to 10 parameters are permitted. NFIT is a counter for the number of iterations CURFIT has made. The user can check if NFIT=0. In this case he can determine (if needed) the initial values of the parameters. (The initial values can also be entered manually during run time).

b. If your function is analytic and has analytical derivatives with respect to the parameters A, define the Subroutine SUBROUTINE DFCURV(X,N,A,DELTAA,NTERMS,DERIV,NFIT), with DIMENSION X(1), A(1), DELTAA(1), DERIV(1). X,N,A, NFIT are as previously defined. NTERMS is the number of parameters A to be fitted. DERIV(N) contains the calculated value of the function's derivative with respect to A(N). DELTAA(N) is the step Δ(A(N)) needed for numerical derivatives:

$\frac{f(x|A+\Delta)-f(x|A)}{\Delta}$. This array (DELTA) can be ignored for analylical derivatives.

c. WC provides a subroutine for numerical derivatives, SUBROUTINE FDERIV, with the same parameters as DFCURV. The user can modify this routine if he or she wishes. The implantation of your routine in WC should proceed as follows (ON MODCOMP):

(i) Replace the binary version of the default function FCURVE and subroutine DFCURV (or FDERIV) on your binary library UL using the procedures $DELFUL FCURVE (and $DELFUL DFCURV) and $ADDTUL NAME2, where NAME2 is the name of the file on your USL which contains the new version of your function and routine.

(ii) Produce a new task, new load-module, of WC (which can be saved under another name) using the procedure $LOADWC (see Appendix 4).

(iii) Execute WC by $ANALY or $WC as usual.

(iv) Try CURFIT in DISP. Good Luck!

Wherever it is mentioned in the HELP of a OC subommand that the result is in the plotting buffer, the execution of PL command (in DSP1) will plot it.

5. PRTSPC

The way the program goes to the PRTSPC overlay is by way of the command ST (stop) in the overlay SETEXP or PRTS command in INITS. (The user can of course switch to PRTSPC and back by using the SWOV command in INITS.)

PRTSPC prints and plots on the line printer all the spectra according to the instructions given by the user in the output control parameter. This parameter is defined in the one-dimensional and two-dimensional spectrum definition commands 1DSP and 2DSP in the overlay INITS (See (IV.2.A.7) and (IV.2.A.8)).

If PRTS command is entered in INITS, selected spectra can be printed bin-by-bin no matter how the output control parameter was set for these spectra.

The special subroutines called by PRTSPC are PLOT1D, PLOT2D, PMAT and PARR. These four subroutines are never used in other parts of WC.

PRTSPC uses the common array DUMMY in the common DUMCOM as a working space.

Examples of output are given in examples 20-24.

V. Common Block Descriptions

```
C
      IMPLICIT INTEGER*2 (I)
C ***   COMMON BLOCKS NEEDED FOR OVERLAYS.
C
      COMMON/IOCOM1/IO1, IO2, IO3, IO4, IO5, IO6, IO7, IO8,IO9
      COMMON/IOCOM2/IRUN,NRUNS,NDB, NDBMAX,NCOUNT,KCOUNT
      COMMON/IOCOM3/IWIN,IWOUT,NSKP(10),NACC(10),NRUNC
      COMMON/TEKCOM/INTEKT
      COMMON/SPCCOM/ISPEC(8424)
      COMMON/DFSCOM/KAVLSP,KSTADD,LEFTSP
      COMMON/DUMCOM/DUMMY(4212)
      COMMON/RANDOM/NSEED
      COMMON/PARCOM/NPARM,LPARM(100),TXTPR(100)
      COMMON/CNSCOM/NCNST,CNST(100)
      COMMON/CONCOM/NCOND,COND(100,2),ICONDP(100)
      COMMON/D1SCOM/ND1SP,ID1SP(100,4),D1SPC(100,3),TXT1D(100,10)
      COMMON/D1SCO2/KD1SP(100),LD1SP(100)
      COMMON/D2SCO1/ND2SP,ID2SP(100,5),D2SPC(100,6),TXT2D(100,10)
      COMMON/D2SCO2/KD2SP/KD2SP(100), LXD2SP(100),LYD2SP(100)
      COMMON/WCNCOM/NWCND,WCND(10,6)
      COMMON/GT2COM/NGATS2,MAXGAT,MAXPOP
      COMMON/GBOCO1/IBN(50),IPX(50),IPY(50),BY(1000),BXL(1000),BXU(1000)
      COMMON/GBCOM2/INWC(50)
      COMMON/TSTCOM/NTST,ITEST(100),NTEST(100),TSTXT(10,100)
```

```
21  COMMON/STSCO1/NSTEST,IGRP1(4),IGRP2(4),ISTSN(4)
22  COMMON/STSCO2/IOPR1(4),IOPR2(4),IOPR3(4)
23  COMMON/STSCO3/IG1D(4,5),IG1G(4,5),IG1P(4,5)
24  COMMON/STSCO4/IG2D(4,5),IG2G(4,5),IG2P(4,5)
25  COMMON/VARCO1/NBITPI,IBITPI(100),ICOORI(100),ISTCOI(100)
26  COMMON/VARCO2/NBITPO,IBITPO(100),ICOORO(100),ISTCOO(100)
27  COMMON/LINCOM/NLINR,LINPR(16,3),XLINDT(16,26,2),YLINL(16,2)
28  COMMON/VIRCO1/NVRSN,IVIR,NVRSET,NVRPLT
29  COMMON/VIRCO2/IVRS(2),IVRTST(12,2),IVRPRX(12,2)
30  COMMON/VIRCO3/VRXL(12,2),VRXU(12,2)VRYL(12,2),VRYU(12,2)
31  COMMON/VIRCO4/XVRL(12),YVRL(12),VRCX(12),VRCY(12)
32  COMMON/VIRCO5/IVRXL(12),IVRXU(12),IVRYL(12),IVRYU(12)
33  COMMON/VIRCO6/IVRPX(12),IVRPYU(12),IVRT(12)
34  COMMON/ANSCOM/IANS1,IANS2,IANS3,IANS4,IANS5,IANS6
35  COMMON/EVNCOM/IPARI,IVI(100),IPARO,IVO(100),EVENT(100),ICMC
36  COMMON/PSPCOM/NPSPR,IFPSPR,IPSPR(100,4),CPSPR(100,2)
37  COMON/QLOCOM/LDNAME(3),LDLIBN(2)


C   COMMON BLOCKS WHICH ARE
C   NEEDED FOR VIRTUAL TO SCREEN TRANSFORMATION.  (IN DISP AND VIRD)

38  COMMON/VR2AB1/XMIN,XMAX,CX,YMIN,YMAX,CY
39  COMMON/VR2AB2/NKEY,IMINX,IMINY,IMAXX,IMAXY
```

```
C   COMMON BLOCKS WHICH ARE
C   NEEDED FOR TRANSFERRING SPECTRUM INFORMATION BETWEEN DISP AND ITS
C   LOWER LEVEL OVERLAYS.

40  COMMON/DS1CO1/IDEV,NFRAM,NSFR,ILTYP,INOLAB,IKEY,NPG,IHDS
41  COMMON/DS1CO2/TEXTX(10),TEXTT(10)
42  COMMON/DS1CO3/NS,NXS,NES,NP,NTSSP,ND,XL,XU,DX,TXTPO,XWML,XWMU
43  COMMOM/DS1CO4/NS1,NS2,NS3,NS4,NPRJS,PL,PU,SL,SU,ADD,DIV,NDI
44  COMMON/FITCOM/FFNAME,A(10),NTERMS,NPRIOR(10)
```

1) IOCOM1 Contains input/output device assignments, which are defined in the main program (program MAIN attached to YFSR),and assigned by the procedure $OVERL.

IO1(IO2) - Reading (writing) from (to) the user terminal-(TY).

IO3 - Reading and writing on a graphics terminal. When working with two terminals the user should assign IO3 to the non-graphics one. The default assignment is to NST.

IO4 - Reading and writing on a graphics terminal <u>only</u> (NST).

IO5 - Printing to a lineprinter or a Versatec (VP).

IO6 - Reading the input (old) setup file. The default assignment is SCO.

IO7 - Writing the output (new) setup file. Assigned to SC1.

IO8 - Reading the spectrum buffer ISPEC. The default assignment is SC2.

IO9 - Writing the spectrum buffer ISPEC. The default assinment is SC2.

IO10 - Write-through-mode device for the TEKTRONIX 4014 (WRU). (will be added in the future).

2) IOCOM2 - IRUN - The current run number as read from the input tape.

NRUNS - The number of runs to be analyzed as determined with the SE or SO commands in the overlay SETEXP. (IV.1)

NDB - The debugging code. (See SE command in SETEXP (IV.1.A.3))

NDBMAX - If NDB=1, one out of NDBMAX events will be printed.

NCOUNT - The number of events analyzed.

KCOUNT - The number of accepted events (those for which NSKIP≥0 in YFSR (IV.3.B.))

3) IOCOM3 - IWIN - Twice the number of 16 bit words per record on the input tape. IWIN is used by the MODCOMP system subroutines GETEVT and MODHDR. The default is 2000.

IWOUT - Twice the number of 16 bit words per record on the output tape. IWOUT is used by the MODCOMP system subroutine PUTEVT, FLSEVT and MODHDR. The default is 2000.

NSKIP(10)-The scalers for rejected events (see YFSR IV.3.B)

NACC (10)-The scalers for accepted events (see YFSR IV.3.B)

NRUNC - The number of runs (out of NRUNS) that are already analyzed.

4) TEKCOM - INTEKT 0 or 1 indicates whether to initialize the graphics terminal or not.

5) SPCCOM - Contains the spectrum buffer. Its default size is 8424x2 bytes.

6) DFSCOM - Constrains the status of spectra defined in WC.

KAVLSP - The total space devoted to spectra (equal to the dimension of the array ISPEC in the common SPCCOM).

KSTADD - The first available address, relative to ISPEC(1), for a spectrum to be defined.

LEFTSP - The size of the buffer which is not yet assigned to spectra. i.e. LEFTSP = KAVLSP-KSTADD+1

7) DUMCOM - Contains a buffer devoted to spectrum manipulations. Used by the overlays DSP1, and PRTSPC, and optionally by the user in YFSR. The default size is the same size as the spectrum buffer. It does not have to be that large, but it must be at least twice the size of the largest spectrum defined.

8) RANCOM - Contains the seed for the random-number generator. (Set in the overlay SETEXP.)

9) PARCOM - NPARM - The number of parameters.

LPARM - The number of channels per parameter

TXTPR - The mnemonics for the parameters

(See PARM command in the overlay INITS (IV.2.A.2))

10) CNSCOM - NCNST - The number of constants defined using the CNST command in INITS (IV.2.A.4)

CNST - The constants so defined.

11) CONCOM - NCOND - The number of one-dimensional conditions defined using the COND command in the overlay INITS (IV.2.A.5)

COND - The lower and upper limits which define the conditions

ICONDP - The default parameter number for each condition.

12,13,) D1SC01, D1SC02 -

ND1SP - The number of one-dimensional spectra defined ($\leqslant$100).

ID1SP - In each spectrum: Parameter number, INTEGER/REAL code, Test number and Output-Control code.

D1SPC - $X_L X_U$ and DX for each spectrum.

TXT1D - Text describing the one-dimensional spectra.

KD1SP, LD1SP - The start address in the spectrum buffer and the length (number of bins) of each spectrum.

(See 1DSP command in the overlay INITS (IV.2.A.7))

14,15) D2SCOM - Contains the two-dimensional spectrum definitions. Explanations of the content of D1SPC and D2SPC are given under the 1DSP and 2DSP commands of the overlay INITS (IV.2.A.8)

16) WCNCOM - NWCND - The number of two-dimensional window conditions. (See the command WCND in the overlay INITS (IV.2.A.9).)

WCND - The limits of the rectangular windows and conversion factors of these limits to the range of 0. to 4096.

17) GT2COM - NGATS2 - The number of two dimenmsional free-form gates. (See GAT2 command in the overlay INITS (IV.2.A.10).)

MAXGAT - The maximum number of gates that can be defined. MAXGAT is set to the value 50 in subroutine INITS.

MAXPOP - The maximum number of different y-coordinates available for the definition of each gate. MAXPOP is set to 20, which is equivalent to 40 corner points per gate, in subroutine INITS

18) GBOCO1 - Contains the two-dimensional free-form gates' boundaries.

IBN - The number of different y-coordinates per gate (it must be less than or equal to MAXPOP).

IPX,IPY- The default x-coordinate and y-coordinate parameter numbers.

BY,BXL,BXU- The gates' boundaries

19) GBCOM2 - INWC - The window condition number of the window condition used while the gate has been defined in VIRD.

Comment regarding 11,15 and 16: Whenever MAXGAT and/or MAXPOP are/is changed keep the following relations:

a. In all the arrays with the dimension 50, change 50 to any number N so MAXGAT$\leqslant$N.

b. In all the arrays with the dimension 1000, change 1000 to any number M so that MAXGAT*MAXPOP ≤ M.

20) TSTCOM - NTST - The number of tests used in YFSR. This number can also be defined using the command TSTX in the overlay INITS (IV.2A.12).

ITEST - The test flags set in YFSR. If a flag is zero, the result of this test is FALSE and if it is non-zero the event passed the test (TRUE). this buffer is set to zero after NTEST is incremented in the overlay ANLDAT.

NTEST - The test scalers. Incremented by 1 for each event, for each test which is found to be TRUE.

TSTXT - Texts for the tests. The test scaler is printed just for tests with text.

21-24) STSC01 - STSC04 - Contains "simple test" definitions.

NSTEST - The number of tests defined as "simple tests" (4).

IGRP1,IGRP2 - The number of elements in groups 1 and 2. (See SSTS command in the overlay INITS (IV.2.A.13).)

IOPR1 - If IOPR1 is zero, it functions as an OR among the elements within the groups. If it is one it function as AND among the elements within the groups.

IOPR2 - If zero or one functions as OR or AND respectively among the two groups.

IOPR3 - If IOPR2 is zero it has no effect. If it is one

it applies NOT logic on the final result of the "simple test"

IG4D, IG2D - The elements' dimension: 0,1 or 2 for test, condition or gate respectively, for up to five elements in each group.

IG1G, IG2G - The test, condition or gate number

IG1P, IG2P - In case of 1-D condition; the parameter number on which the condition is applied.

25)and 26) VARCO1 and VARCO2 - Contains input and output variable format specifications. (See FORM command in the overlay INITS (IV.2.A.3).)

27) LINCOM NLINR - The numbers of linearizatons. (See LINR command in the overlay INITS (IV.2.A.11.)

XLINDT - The x-coordinates of the linearization lines.

28-33) VICO1-VIRCO6 -

Contains the description of the virtual display setups.

NVRSN - The number of setups defined ($\leqslant 2$).

IVIR - An internal flag set in VIRD. If 1 the virtual displays are plotted in ANLDAT.

NVRSET - The setup number <u>currently</u> plotted on the screen.

NVRPLT - The number of virtual displays within the current setup NVRSET ($\leqslant 12$).

IVRS - The number of virtual displays in each setup.

VRXL,VRXU,VRYL,VRYU - X-low, Y-up, X-low, and Y-up for all the displays within each setup.

XVRL,YVRL,VRCX,VRCY - X and Y lower limits and X and Y compression factor (from screen to virtual window) for each display within the current setup.

IVRTST - The test number of the tests gating the virtual display in each setup.

IVRT - The same as IVRTST but for the current setup.

34) ANSCOM - Contains the analysis setup mode set using the SE command in the overlay SETEXP (IV.1.A.3). The answer is a single letter either Y for yes or N for no.

IANS1=REWRITE HEADER

IANS2=INPUT FROM DEV 6

IANS3=CREATE OUTPUT FILE FOR EACH INPUT FILE

IANS4=GENERATE O/P TAPE

IANS5=NO INPUT TAPE. TO BE USED IN SIMULATION CODES

IANS6=GRAPHIC TERMINAL. (The answer is T,V, or N)

35) EVNCOM - The event-buffer description. (See YFSR overlay (IV.3.B).)

IPARI,IPARO- The maximum number of parameters on the input and output tapes (determined by the MODCOMP system routine MODHDR in the overlay SETEXP or by the user with the command PARM in the overlay INITS or in YFSR.)

IVI,IVO - The input/output event buffer (100 must be greater than or equal to IPARI and IPARO.)

EVENT - The "working event-buffer" which is set equal to IVI in the overlay ANLDAT. EVENT is a REAL*4 array. All event-by-event operations are done on this buffer.

ICMC- is + 1 for data written using the multiplexer and it is -1 for data written using CAMAC. ICMC is used by the MODCOMP system routines GETEVT and PUTEVT.

36) PSPCOM - NPSPR- The number of pseudo-parameters. (See PSPR command in the overlay INITS (IV.2.A.6).)

IFPSPR - IFPSPR - The default parameter number for the first pseudo-parameter to be defined i.e. IFPSPR=IPARI+1.

IPSPR - The pseudo-parameter types, parameter numbers, and the input (the modified) parameter numbers.

CPSPR - The constants used for pseudo-parameter calculations

37) QLOCOM - LDNAME - The name of YFSR on the load module library (ASCII)

LDLIBN - The name of the load module library (TM,TU etc.) (ASCII characters)

38,39) VR2AB1,VR2AB2- Internal common blocks in DSP1 for virtual to absolute coordinate transformations. (These transformation are needed since not all the PLOT10 routines function on the MODCOMP.)

VI. Instructions for Changing WC

One of the most important features of WC is that it is easy to modify. Four kinds of modifications are discussed: replacement or addition of commands, changing the dimensions of arrays, moving to another computer, and changing the default values of parameters.

1. The replacement or addition of commands

WC is written in a segmented way so that almost all commands are in separate subroutines or in isolated parts in the same subroutine. In many cases subroutines are self-contained, although this causes several parts of the program to appear several times. Since the last feature is not universal in WC, whenever the user changes certain subroutines it is recommended that the user search for all the calls to that subroutine and, if there are such calls, see how they are affected by the change.

To add a new command without interfering with the rest of the program is trivial. Just look at the beginning of any overlay and the logic is quite transparent.

2. Changing the dimension of arrays

Two kinds of arrays are discussed: those which are defined and loaded in the overlay INITS, and the rest of the buffers, including the event and the spectrum buffers.

A. Arrays loaded in overlay INITS:

Arrays which are defined in the commands FORM, PARM, CNST, COND, 1DSP, 2DSP, TSTX, WCND, LINR, GAT2 and PSPR should be looked for in three locations in the program:

1. The common block referring to the array to be changed.
2. The subroutine which reads the input. In the overlay INITS all subroutines have names similar to the command in INITS plus the letter S. (The exact names can be easily seen in subroutine INITS.) For example: GAT2 → GAT2S, FORM → FORMS. 1DSP → D1SPS, 2DSP → D2SPS. These subroutines make sure input is not written beyond the allowed dimensions.
3. In INITS itself, certain arrays are zeroed and constants are set to their initial values. For example the maximum number of free-form two dimensional gates, MAXGAT, is set to 50. (See V.14.)

<u>Example</u>: Change the number of one-dimensional spectra from 100 to n.

1) In the common D1SCOM change D1SPC(100,7), and TXT1D(100,10) to D1SPC(n,7) and TXT1D(n,10).
2) In subroutine D1SPS, search for the number 100. It only appears in the section where new spectra are added, in the statement

IF (NUM.LE.0.OR.NUM.GT.100) RETURN 1

Change 100 to n.

3) In the initialization section of INITS the texts of the spectra are set to blanks. Put this statement in a separate DO loop and change the DO-loop range from 100 to n.

B. Non-standard arrays:

1) If the number of parameters per event is to be be changed, check first to see if the MODCOMP system routines GETEVT, PUTEVT and MODHDR "know" how to handle the requested new number. Then 100 should be changed to n in at least two common blocks (EVNCOM and PARCOM). If the user wants to raise the limit on the number of parameters plus pseudo-parametes to n, refer to the places mentioned in subsection (VI.2.A) for the commands PARM and PSPR in INITS.)

The dimension of the event-buffer does not appear elsewhere.

2) Change the spectrum buffer size.

The default size is 8424 INTEGER*2 words, but this buffer size also appears as 8424/2=4212 in all the places where the INTEGER*2 buffer is in equivalence with the REAL*4 buffer. So both numbers should be changed simultaneously in the common SPCCOM and in the subroutines SPECTRA, INC1DS, INC2DS and

SAVESP. The numbers 4212 and 8424 are unique in the program, so only two global change commands are needed to change 4212 to n and 8424 to 2n. Choose your new n in such a way that further changes will be as simple.

Important remark (for MODCOMP): Changing arrays in WC or adding COMMON blocks in WC (not in YFSR) may change the addresses of the common blocks in the memory. Since we load our YFSR dynamically during WC execution, all correct common block must appear in YFSR. Furthermore, while producing the load-module of YFSR by the procedure $ANALY the start addresses of all the commons as given in the COM command in EXE M4EDIT should be the correct ones. (For new common blocks, ATR and COM command should be added). The procedure $LOADWC prints the load map of WC. Edit the procedure $ANALY and correct the addresses according to this load map. (List of the procedures are given in Appendix 4).

3) Moving to another computer

The program is written in such a way that it is portable, i.e., use of special features of the MODCOMP is avoided. However, it was impossible to ignore the special limitations of the MODCOMP (which are many). So in general, "beautification" projects are possible on other computers.

The program is written in standard FORTRAN language. All the plotting routines are those of PLOT10, the normal TEKTRONIX plotting language. Be sure that this package exists on your computer. On the MODCOMP, the Feb. 1976 version of PLOT10 has been installed, but many important subroutines like VWINDO, DWINDO, and all terminal status routines are not operative, so parts of them have been replaced by simplified subroutines like VR2AB, AB2VR, SMOVEA, and SDRAWA. If the user wishes to simplify the plotting parts in VIRD and DSP1, it is a relatively simple task.

The following subroutines should be replaced by those available on the user's system: GETEVT, PUTEVT, MODHDR and FLSEVT, which read or write records and events from or to tape or disk; and MTIO, AVRUN2, BKRUN, and AVEOT2, etc. which manipulate the magnetic tape drive.

Other routines to be replaced are TRAPON, TRAPOF, CHREAD and QLOAD. The first two enable and disable the run-time interrupts. Actually they queue a read command which is tested whenever the CHREAD routine is called. If the proper key is hit, the program branches to the interrupt handler, otherwise the program proceeds normally. Remarks on QLOAD are given in section 5 below.

4. Changing default values of parameters

The MODCOMP has an important feature which is used for setting defaults. Whenever the READ command in FORTRAN is performed (either formatted or not, e.g. READ(IO,*)), if nothing is entered (i.e. <CR> or ,, in an unformatted READ) the values of the variables read remain unchanged. So usually, in the line before a READ command all variables are set to their default values.

If such an option does not exist on the user's computer, the default setting should be redone.

5. Quick load of YFSR

The program assigns a space in ANLDAT overlay (the array LDBUF) for the load-module of YFSR. The load-module of YFSR is saved with relative (and not absolute) addresses and it is loaded dynamically at run-time into this buffer by the routine QLOAD. In order to accomplish this powerful technique, QLOAD should be rewritten if one is moving WC to another computer.

VII. Final Remarks

GOOD LUCK.

VIII. Acknowledgments

The authors wish to thank Mr. Bob Belshi for much helpful advice and for the first version of the free-form gates subroutine. We are grateful to Drs. R. McDonald and L. G. Sobatka for supplying us with their program EVENT (the first version of SETEXP) and their one-dimensional spectrum display subroutine (the first verison of DSP1).

We are greatly indebted to C. R. Albiston for offering useful advice on early drafts of this manuscript.

IX. Example Captions

1. A simple execution of WC - An example of user-terminal interaction.
2. The list of the subcommands of SETEXP.
3. Termination of WC by the ST subcommand of SETEXP.
4. The list of the subcommands of INITS.
5. The SWOV subcommand of INITS.
6. The PARM subcommand of INITS.
7. The CNST and COND subcommands of INITS.
8. The PSPR subcommand of INITS.
9. The 1DSP subcommand of INITS.
10. The 2DSP subcommand of INITS.
11. The WCND and GAT2 subcommands of INITS.
12. The TSTX and SSTS subcommands of INITS.
13. The subcommands of the overlay ANLDAT.
14. The list of the subcommands of the overlay VIRD, and SETV subcommand.
15. The subcommands CURS, HCOP, LINR and GATE of VIRD.
16. The list of the subcommands of DSP1D and the list of the subcommands of SPCOPR.
17. Selected subcommands of the overlay DSP1D.
18. Selected subcommands of the overlay DSP1D.
19. Combination of SPCOPR commands and DSP1D commands for setting an error-spectrum and plotting a Gaussion fit to a spectrum. Changing title and entering label through the terminal is demonstrated.
20. WC output: Event statistics and TEST results.
21. Printout of 1-D spectra bin by bin.
22. 1-D spectrum display on the line printer. (Output control code = 10 in the spectrum definition).
23. Printout of 2-D spectra bin by bin.
24. 2-D spectrum display on the line printer.

EXAMPLE 1a

```
JOB
$
WC SN250E                ←Execute the procedure - $WC using the setup file SN250E
$NOP
$RESTFI SN250E,SCO          $RESTFI copy the setup from USL to the scratch file SCO
$NOTE COP FIL (P1) TO SCn (P2). (PROTOCOL,SCO)   14:27:37
$NOTE COP FIL (P1) TO SCn (P2). (PROTOCOL,SCO)   14:27:37   /BAT007/RES
$AVR CI 2
$ASS SO SCO
$AVR CI 10
$EXE SED
.
REW SO
.
POS SN250E
.
NOCO
.
ASS LO NO
.
.
.
.
$NOTE END OF "RESTFI" PROC.    14:27:45
$NOTE END OF "RESTFI" PROC.    14:27:45   /BAT007/BATCH
$END  RESTFI
$NOP
$ASS 1=TY 2=TY 3=TY 4=PLA DSP=PLT 5=VP   ←Set the assignments
$ASS 6=SC0 7=SC1 8=SC2 9=SC2
$ASS MTW=MT2
$REW 6,7,8
$EXE WC TM                                ←Invoke the WC task


        *** WELCOME TO   W C , VERSION: AUG 83 ***
      WRITTEN BY SHLOMO WALD, LBL BLDG. 88,  X5088.

SETEXP >>>              ←——Entering SETEXP
AS
INPUT MT UNIT=
2
OUTPUT MT UNIT=
                        ←——— <CR>
NO OUTPUT TAPE.
SETEXP >>>
SE
GRAPHIC TERMINAL:T-TEKTRONIX,V-VT100,N-NON(T/V/N)
N
DEBUG OPTIONS: 0=> NONE, 1=> LIMITED, 2=> FULL

READ SETUP FILE FROM DEV 6? (Y OR N)

INPUT TAPE WRITTEN IN STANDARD CAMAC FORMAT ? (Y/N)

ZERO SKIP ON INPUT? (N OR Y)

GENERATE O/P TAPE? (N OR Y)

SETUP CONDITIONS:
```

```
                                (1) CHANGE HEADER? N
                                (2) DEBUG OPTION=  0
                                (3) Z-SKIP ON INPUT=N
                                (4) Z-SKIP ON OUTPUT=N
                                (5) WRITE FILE MARKS EACH RUN= Y
                                (6) WRITE O/P TAPE =N
                                (7) READ FROM DISK (UNIT 6)=Y
                                (8) NO INPUT TAPE (SIMULATION)=N
NUMBER OF RUNS TO SORT USING THESE CONDITONS
4
ORIGINAL RUN #=1705       5 PARAMETERS PER EVENT
HEADING= 250MEV 160+SN 12.5DEG
*** IGOTO,IPASS ***     2     1          <------ Entering INITS
ENTER INIT COMMAND. For help enter HELP >>>
ANLD
ANLD
*** IGOTO,IPASS ***     3     1          <------ Entering ANLDAT
ENTER "YFSR" LOAD-MODULE NAME. DEFAULT: WCOO
APBOX                          <------ Quick load of YFSR overlay from file TM
ENTER LOAD-LIBRARY FILE NAME. DEFAULT: TM

ENTER TELESCOP #          <------ This question is printed during the 1st
1                                 call of this specific YFSR

*** IGOTO,IPASS ***     1     4 <------ Return to SETEXP for reading the header
ORIGINAL RUN #=1706       5 PARAMETERS PER EVENT                of the new run
HEADING= 160+SN 250MEV 12.5 DEG.
*** IGOTO,IPASS ***     3     4 <------ Then, return to ANLDAT


*** IGOTO,IPASS ***     1     4
ORIGINAL RUN #=1707       5 PARAMETERS PER EVENT
HEADING= SAME AS BEFORE
*** IGOTO,IPASS ***     3     4


*** IGOTO,IPASS ***     1     4
ORIGINAL RUN #=1708       5 PARAMETERS PER EVENT
HEADING= 250MEV 016 + SN  12.5DEG
*** IGOTO,IPASS ***     3     4


*** PERFORM THE LAST CALL TO "YFSR"  ? (Y/N) *** <------ The analysis of 4
                                                        runs was completed
DUMP SECTION, NPASS=  -1

ANLDAT >>>

SETE                           <------ Return to SETEXP
   4      ***     W C      CODE    ***    RUN NUMBER 1708
EVENTS READ=      23556     EVENTS WRITTEN=      18901
 # OF BLOCKS READ=    118  # OF BLOCKS WRITTEN=      0
REJ.        0       0       0       0       0       0       0       1  25221       0
ACC.   119424       0       0       0       0       0       0       0       0       0

                 *** NUMBER OF EVENTS WHICH PASSED THE DEFINED TESTS ***
               TEST NUMBER                                   NUMBER OF EVENTS
                  1  FACE #1 & #7 FIRED TOGETHER.                        572
                  2  FACE #8 AND (FACE #1 OR #2) FIRED.                  881
```

```
                    3  N+M+K >= 4

*** IGOTO,IPASS ***     1     5
SETEXP >>>
RW                                          ←—— Rewind the tape
REWIND INPUT TAPE?? (Y OR N)

SETEXP >>>
IN                ←—— OOPS!, Spectra 1 and 2 have OUTPUT CONTROL code =-1 and I
*** IGOTO,IPASS ***     2     5    want to print them bin-by-bin.  So let us
DONE WITH "INIT"  ? (N/Y)          switch to INITS
                                                                    .
ENTER INIT COMMAND. For help enter HELP >>>                         .
PRTS                                                                .
*** IGOTO,IPASS ***     5     5    ←—— Then, switch to PRTSPC       .
ENTER DIMENSION OF SPECTRA TO BE PRINTED (1/2)                      .
2                                                                   .
ENTER SPECTRUM NUMBERS OF THE 1ST AND LAST SPECTRA                  .
TO BE PRINTED BIN-BY-BIN. DEFAULT:ALL SPECTRA
1 2                                                 And print specta 1ℓ2
*** IGOTO,IPASS ***     2     5                     on the line printer
DONE WITH "INIT"  ? (N/Y)

ENTER INIT COMMAND. For help enter HELP >>>
SETE                                        ←—— Return to SETEXP
*** IGOTO,IPASS ***     1     5
SETEXP >>>
ST                                          ←—— Anol STOP WC
*** IGOTO,IPASS ***     2   100
ENTER TITLE FOR OUTPUT-SETUP FILE. OLD TITLE:
SE13A - SETUP FOR RAW DATA OF TOF13 EXP. 11/1/83
SE13B - TOF13 EXP. SETUP WITH VIRTUAL DISPLAY & PSPR.
THIS SETUP IS SAVED NOW ON UNIT I07
ENTER OUTPUT-SPECTRUM-BUFFER TITLE >
SP13B - SPECRUM BUFFER OF INITIAL ANALYSIS OF TOF13 EXP.
THE SPECTRUM BUFFER IS SAVED NOW ON UNIT I09
*** IGOTO,IPASS ***     5   100
*** IGOTO,IPASS ***     1   100
       11/ 7/1983      10:55

*** END OF  W C  PROGRAM ***
*** IGOTO,IPASS ***    10   100
$AVR CI 8
$WEOF 7
$WEOF 9
$WEOF 5


$JOB END OF OVERL PROCEDURE.  10:55:41  /BAT000/BATCH
$END  WC
$
BYE
HAVE A NICE DAY
```

```
JOB
$                                   Using the procedure $SAVEFI to save the spectra
SAVEFI SPCBUF SC2 <------------ buffer on USL under the name SPCBUF
$NOTE COPY TO USL FILE (P1) FROM SCn (P2). (PROTOCOL,SC1)    15:01:36
$NOTE COPY TO USL FILE (P1) FROM SCn (P2). (PROTOCOL,SC1)    15  /BAT007/SAV
$AVR CI 2
$ASS SI SC2
$AVR CI 6
$EXE SED
.
REW SI
.
NOCO
.
CAT SPCBUF
.
EXI
$ASS SI SI
$NOTE END OF "SAVEFI" PROC.   15:01:42
$NOTE END OF "SAVEFI" PROC.   15:01:42  /BAT007/BATCH
$END  SAVEFI
$
```

```
$END  RESTFI
$NOP
$ASS 1=TY 2=TY 3=TY 4=PLA DSP=PLT 5=VP
$ASS 6=SC0 7=SC1 8=SC2 9=SC2
$ASS MTW=MT2
$REW 6,7,8
$EXE WC TM


        *** WELCOME TO   W C , VERSION: AUG 83 ***
      WRITTEN BY SHLOMO WALD, LBL BLDG. 88,  X5088.

SETEXP >>>
HELP
COMMANDS ARE: AF=ADVANCE (OR BACK) FILES (RUNS) ON MT UNITS
              AR=ADVANCE/BACK RECORDS ON INPUT MT UNIT
              AS=ASSIGN MT UNITS
              ET=GO TO THE END OF THE OUTPUT TAPE
              HE=HELP
              RW=REWIND TAPE
              SE=ENTER SETUP MODE
              SO=SORT DATA USING YFSR
              ST=STOP
              IN=GO TO INIT OVERLAY
              AN=GO TO ANLDAT OVERLAY.
              WE=WRITE END OF FILE ON OUTPUT TAPE
SE13A >>>
```

EXAMPLE 3

```
*** IGOTO,IPASS ***    2  100
ENTER TITLE FOR OUTPUT-SETUP FILE. OLD TITLE:
SE13A - SETUP FOR RAW DATA OF TOF13 EXP. 11/1/83
SE13B - TOF13 EXP. SETUP WITH VIRTUAL DISPLAY & PSPR.
THIS SETUP IS SAVED NOW ON UNIT I07
ENTER OUTPUT-SPECTRUM-BUFFER TITLE >
SP13B - SPECRUM BUFFER OF INITIAL ANALYSIS OF TOF13 EXP.
THE SPECTRUM BUFFER IS SAVED NOW ON UNIT I09
*** IGOTO,IPASS ***    5  100
*** IGOTO,IPASS ***    1  100
       11/ 7/1983     10:55

*** END OF  W C  PROGRAM ***
*** IGOTO,IPASS ***   10  100
$AVR CI 8
$WEOF 7
$WEOF 9
$WEOF 5


$JOB END OF OVERL PROCEDURE.  10:55:41  /BAT000/BATCH
$END  WC
$
BYE
HAVE A NICE DAY
```

```
IN
*** IGOTO,IPASS ***     2     2
DONE WITH "INIT"  ? (N/Y)

ENTER INIT COMMAND. For help enter HELP >>>
HELP
*** LIST OF INIT COMMANDS ***
HELP - GET THIS LIST
FORM - I/O FORMAT DEFINITION
PARM - SET PARAMETER LIST, MNEMONICS, ETC.
CNST - ENTER CONSTANTS
COND - 1 DIM. CONDITIONS
GAT2 - 2 DIM. GATES
WCND - 2 DIM. WINDOW CONDITIONS
SSTS - SET SIMPLE TEST WHICH WILL BE APPLIED ON EACH EVENT
PSPR - SET PSEUDO PARAMETERS
TSTX - TEXT DESCRIPTION OF TESTS
1DSP - 1 DIM. SPECTRUM DEFINITION
2DSP - 2 DIM. SPECTRUM DEFINITION
LINR - SET LINEARIZATION LINES
VIRS - SET VIRTUAL-DISPLAY SETUPS
ZSPC - ZERO SELECTED SPECTRA
ZALL - ZERO ALL SPECTRA
REST - RESTORE SPECTRUM BUFFER FROM UNIT IO8
SVSP - SAVE SPECTRUM BUFFER ON UNIT IO9
SVSE - SAVE SETUP ON UNIT IO7
SWOV - SWITCH BETWEEN OVERLAYS (CHANGE IGOTO,IPASS)
SETE - RETURN TO SETEXP
DISP - SWITCH TO "DISPS" OVERLAY - (DISPLAY SPECTRA)
PRTS - SWITCH TO "PRTSPC" - PRINT SPECTRA BIN-BY-BIN
ANLD - DONE WITH "INIT", GO TO "ANLDAT" OVERLAY

ENTER INIT COMMAND. For help enter HELP >>>
REST
INPUT-SPECTRUM-BUFFER TITLE:
20NE+26MG 220MEV TOF13     11/3/83
ENTER INIT COMMAND. For help enter HELP >>>
```

```
ENTER INIT COMMAND. For help enter HELP >>>
SWOV
*** IGOTO,IPASS CODES ***
IGOTO            OVERLAY
  1              SETEXP            "HARDWARE" DEFINITIONS
  2              INITS             "SOFTWARE" DEFINITIONS
  3              ANLDAT   EVENT-BY-EVENT AND DATA ANALYSIS
  4              DISPS    MANIPULATION AND DISPLAY OF SPECTRA
  5              PRTSPC   PRINT RESULTS AND SPECTRA ON PRINTER
  6              VIRD     SET 2 DIM. VIRTUAL DISPLAYS
 10              STOP WC. (SET BY ST-COMMAND IN SETEXP)
IPASS                                                    SET IN
 1  FIRST CALL TO SUBROUTINES.                           SETEXP
 2  IN THE MIDDLE OF A RUN                               ANLDAT
 3  AT THE END OF A RUN BUT NO EOF ON OUTPUT TAPE.       ANLDAT
 4  AT THE END OF A RUN WITH EOF ON OUTPUT TAPE.         ANLDAT
 5  AFTER SORTING THE SPECIFIED NUMBER OF RUNS.          ANLDAT
 100  CLOSING PROCEDURE. GOTO PRTSPC. (ST-COMMAND OF SETEXP
(*)  IGOTO=1,IPASS=0 RE-EXECUTE "WC" FROM THE BEGINNING.
(**) IGOTO=3,IPASS>5 ENABLE TO PERFORM THE LAST CALL TO "YFSR"
(***)IGOTO=5,IPASS>100 RESULTS WITH ALL PRINTOUT AS IN
     IPASS=100 BUT THE PROGRAM RETURNS TO "INITS".
     IPASS<100 IS THE SAME AS THE "PRTS"-SUBCOMMAND
ENTER NEW VALUES FOR IGOTO AND IPASS
PRESENT VALUES ARE    2    2
2 2
*** IGOTO,IPASS ***    2    2
DONE WITH "INIT"  ? (N/Y)

ENTER INIT COMMAND. For help enter HELP >>>
```

```
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
PARM
DO YOU WANT TO MODIFY I/O NUMBER OF PARAMETERS ? (N/Y)

ENTER NUMBER OF PARAMETERS (.LE.100) >>
30
PARM          30
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
HELP
* PARM SUBCOMMAND MENU *
HELP - GET THIS LIST
CHAN - DEFINE NUMBER OF CHANNELS FOR EACH PARAMETER
MEMO - ATRIBUTE NAMES TO PARAMETERS
END - RETURN TO THE "INIT" MENU
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
CHAN
ENTER DEFAULT # OF CHANNELS PER PARMETER. DEFAULT=4096
2048
DEFAULT NUMBER OF CHANNEL SETTED TO  2048
ENTER # OF PARM"S FOR THEM # OF CHAN"S ARE SPECIFFIED
2
ENTER     3 PAIRS OF "PARM #" & "# OF CHANNELS" (UNFORMATED)
1 1 2 64 21 1
  1   1  2   64 21    1
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
MEMO
LIST OR ADD PARAMETER"S MENMONICS ? (L/A) >
L
    1 WGR      2 BITR     3 X-F1     4 X-D1     5 Y-L1     6 Y-S1     7 X-F2
    8 X-D2     9 Y-L2    10 Y-S2    11 U-AN    12 D-AN    13 T-F1    14 T-F2
   15 DEST    16 DEF1    17 DEF2    18 DEAN    19 MON1    20 MON2    21 DUMM
   22 X-1     23 X-2     24 Y-1     25 Y-2     26 TETA    27         28
   29         30
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
MEMO
LIST OR ADD PARAMETER"S MENMONICS ? (L/A) >
A
ENTER THE NUMBER OF PARAM"S FOR WHICH MEMO"S ARE GIVEN
2
ENTER     2PAIRS OF "PARM #" & "MEMO". FORMAT 7(I5,1X,A4)
UP TO 7 PAIRS IN ONE ROW >
  #### MMMM #### MMMM #### MMMM #### MMMM #### MMMM #### MMMM #### MMMM
     2 CHAK    30 DEMO
     2 CHAK    30 DEMO
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
MEMO
LIST OR ADD PARAMETER"S MENMONICS ? (L/A) >
L
    1 WGR      2 CHAK     3 X-F1     4 X-D1     5 Y-L1     6 Y-S1     7 X-F2
    8 X-D2     9 Y-L2    10 Y-S2    11 U-AN    12 D-AN    13 T-F1    14 T-F2
   15 DEST    16 DEF1    17 DEF2    18 DEAN    19 MON1    20 MON2    21 DUMM
   22 X-1     23 X-2     24 Y-1     25 Y-2     26 TETA    27         28
   29         30 DEMO
ENTER PARM SUBCOMMAND. FOR HELP ENTER HELP >
END
PARM END
```

EXAMPLE 7

```
ENTER INIT COMMAND. For help enter HELP >>>
CNST
ADD,CHANGE OR LIST CONSTANTS (A/C/L) >
A
ENTER NUMBER OF CONSTANTS (.LE.100) >>
2
CNST          2
ENTER    2 CONSTANTS ONE-BY-ONE (FREE FORMAT)
ENTER CONSTANT #     2
3.14
ENTER CONSTANT #     3
6.28
   2: 3.1400E+00     3: 6.2800E+00
ENTER INIT COMMAND. For help enter HELP >>>
CNST
ADD,CHANGE OR LIST CONSTANTS (A/C/L) >
C
CHANGE CNST(I) FOR  IFIRST=<I=<ILAST.
ENTER  IFIRST AND ILAST >
3 3
ENTER CONSTANT #    3
11.
   3: 1.1000E+01
ARE THESE OK?(Y/N)

<CR>

ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
COND
ADD, CHANGE, DELETE, LIST CONDISIONS (A/C/D/L)
TO RETURN TO "INIT" ENTER <CR>
A
ENTER NUMBER OF CONDITIONS (.LE.100) >>
2
COND          2
  1: ENTER  X-LOW,X-UP  AND DEFAULT PARM.# >
20.,135.5  4
  P: ENTER  X-LOW,X-UP  AND DEFAULT PARM.# >
7.,7.,2
  2:(  20.000, 135.500)   3:(   7.000,   7.000)
    4:      2:
ADD, CHANGE, DELETE, LIST CONDISIONS (A/C/D/L)
TO RETURN TO "INIT" ENTER <CR>
L
*** LIST OF CONDITIONS ***
 CONDITION NUMBER:(X-LOW, X-UP)
  1:(   7.000,   7.000)   2:(  20.000, 135.500)   3:(   7.000,   7.000)
DEFAULT PARAMETER"S NUMBERS:
    3:      4:      2:
ADD, CHANGE, DELETE, LIST CONDISIONS (A/C/D/L)
TO RETURN TO "INIT" ENTER <CR>
D
ENTER NUMBER OF LAST DEFINED CONDITIONS TO BE DELETED
1
   2 CONDITIONS REMAIND
ADD, CHANGE, DELETE, LIST CONDISIONS (A/C/D/L)
TO RETURN TO "INIT" ENTER <CR>
L
*** LIST OF CONDITIONS ***
 CONDITION NUMBER:(X-LOW, X-UP)
  1:(   7.000,   7.000)   2:(  20.000, 135.500)
DEFAULT PARAMETER"S NUMBERS:
    3:      4:
ADD, CHANGE, DELETE, LIST CONDISIONS (A/C/D/L)
TO RETURN TO "INIT" ENTER <CR>
```

EXAMPLE 8

```
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
PSPR
ADD,CHANGE,DELETE OR LIST PSEUDO PARAMETERS (A/C/D/L)
RETURN TO INITS BY <CR>  >
A
ENTER PSEUDO-PARAMETER TYPE AND  # (DEF.# 11) >

*** PSEUDO-PARAMETER TYPES ***
1 - "ENERGY CALIBRATION",  E = C1* (CHANNEL - C2)
2 - MASS-FROM ENERGY AND T.O.F,  M = C1*(E-*C(T-C2)**2)
3 - PARTICLE I.D.(CHARGE), PID=C1*SQRT((E+DE)**C2-E**C2)
4 - ADD PARAMETERS, SUM= C1*DE + C2*E

ENTER PSEUDO-PARAMETER TYPE AND  # (DEF.# 11) >
2
MASS=C1*(E * (T-C2)**2)
   ENTER PARM.# OF E AND T >
16 13
ENTER C1 AND C2 (DEFAULT 1.0E-06 AND 0.) >
<CR>
  11    2  11  16  13    1.0000E-06    0.0000E-01
ADD,CHANGE,DELETE OR LIST PSEUDO PARAMETERS (A/C/D/L)
RETURN TO INITS BY <CR>  >
C
CHANGE PSPR. ENTER PSEUDO PARAMETER NUMBER >
11
*** ORIGINAL PSEUDO PARAMETER SETUP ***
      TYP PS# PR1 PR2       C1            C2
  11    2  11  16  13    1.0000E-06    0.0000E-01
ENTER PSEUDO-PARAMETER TYPE AND  # (DEF.# 11) >
2 30
MASS=C1*(E * (T-C2)**2)
   ENTER PARM.# OF E AND T >

ENTER C1 AND C2 (DEFAULT 1.0E-06 AND 0.) >
1.E-05,1024.
  11    2  30  16  13    1.0000E-05    1.0240E+03
ADD,CHANGE,DELETE OR LIST PSEUDO PARAMETERS (A/C/D/L)
RETURN TO INITS BY <CR>  >
L
*** PSEUDO PARAMETERS LIST ***
      TYP PS# PR1 PR2       C1            C2
   1    1  22   3   0    1.0000E+00   -2.0480E+03
   2    4  22  22   4    1.0000E+00   -1.0000E+00
   3    1  23   7   0    1.0000E+00   -2.0480E+03
   4    4  23  23   8    1.0000E+00   -1.0000E+00
   5    1  24   5   0    1.0000E+00   -2.0480E+03
   6    4  24  24   6    1.0000E+00   -1.0000E+00
   7    1  25   9   0    1.0000E+00   -2.0480E+03
   8    4  25  25  10    1.0000E+00   -1.0000E+00
   9    1  26  11   0    1.0000E+00   -2.0480E+03
  10    4  26  26  12    1.0000E+00   -1.0000E+00
  11    2  30  16  13    1.0000E-05    1.0240E+03
ADD,CHANGE,DELETE OR LIST PSEUDO PARAMETERS (A/C/D/L)
RETURN TO INITS BY <CR>  >
```

EXAMPLE 9a

```
ENTER INIT COMMAND. For help enter HELP >>>
1DSP
ADD, DELETE, LIST, CHANGE 1 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
A
ENTER NUMBER OF 1 DIM. SPECTRA (.LE.100) >>
2
1DSP          2
HELP FOR SPEC. DEFINITION AVAILABLE. ENTER ? >
?
*** SPECTRUM DEFINITION & OUTPUT CONTROL INDICES : ***
          1ST- PARAM NUM.
          2ND- LOWER LIMIT.
          3RD- UPPER LIMIT.
          4TH- BIN SIZE.
          5TH- 1 : INTEGER * 2 SPECTR.
               2 : REAL * 4 SPECTR.
          6TH- THE TEST # BY WHICH THE SPECR. IS GATED. 0-NO GATE.
          7TH- OUTPUT CONTROL.
              -1 : NOTHING IS PRINTED.
               0 : JUST TOTAL AREA IS PRINTED.
               1 : PRINT CH. BY CH.
               2 : PRINT CH. BY CH. -LOG. SCALE.
              10 : PLOT.
              20 : PLOT , LOG. SCALE.
              11 : PRINT + PLOT.
              22 : PRINT + PLOT , LOG. SCALE.
ENTER
PARAM #     XL          XU        XBIN   INT/REAL TEST    I/O
30 20.5 293.22 1.5,2,100,
   30    2.050E+01 2.932E+02 1.500E+00    2      100     -1
TEXT FOR SPECTRUM NUMBER  23 ::
MASS DISTRIBUTION OF HEAVY IONS IN TEL#1.
 23: MASS DISTRIBUTION OF HEAVY IONS IN TEL#1

<CR>

   30    2.050E+01 2.932E+02 1.500E+00    2      100     -1
TEXT FOR SPECTRUM NUMBER  24 ::
<CR>
 24:
ADD, DELETE, LIST, CHANGE 1 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
D
# OF 1 DIM SPECTRA TO BE DELETED >
2
   22 1 DIM. SPECTRA REMAIN
```

EXAMPLE 9b

```
ADD, DELETE, LIST, CHANGE 1 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
C
ENTER THE NUMBER OF THE SPECTRUM TO BE MODIFIED
2
OLD VALUES OF SPECTRUM   2
PARAM #     XL          XU          XBIN   INT/REAL TEST    I/O
   16    0.000E-01 5.000E+00 4.000E-02    2        2     -1
ENTER NEW VALUES. BEWARE IF YOU INCREASE THE SPACE OF THIS SPECTRUM !
,,,5.E-02
   16    0.000E-01 5.000E+00 5.000E-02    2        2     -1
  2: A=21
TEXT FOR SPECTRUM NUMBER   2 ::
A=21 GATED SPECTRUM
ADD, DELETE, LIST, CHANGE 1 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >

<CR>
```

```
2DSP          1
HELP FOR SPEC. DEFINITION AVAILABLE. ENTER ? >
?
* 2 DIM. SPECTRUM DEFINITION & OUTPUT CONTROL INDICES
                1ST- PARAM NUM. OF X-COORDINATE.
                2ND- LOWER LIMIT .
                3RD- UPPER LIMIT .
                4TH- BIN SIZE.
                5TH,-8TH ARE AS 1ST-4TH FOR Y-COORDINATE.
                9TH- 1 : INTEGER * 2 SPECTR.
                     2 : REAL * 4 SPECTR.
                10TH- THE NUMBER OF THE TEST WHICH GATES THE
                      SPECTRUM. IF 0 THE SPECTRUM IS NOT GATED.
                11TH- OUTPUT CONTROL .
                    -1 : NOTHING IS PRINTED.
                     0 : JUST TOTAL AREA IS PRINTED.
                     1 : PRINT CH. BY CH.
                     2 : PRINT CH. BY CH. -LOG. SCALE .
                    10 : PLOT
                    20 : PLOT , LOG. SCALE .
                    11 : PRINT + PLOT .
                    22 : PRINT + PLOT , LOG. SCALE .

ENTER
-X-     XL          XU        XBIN    -Y-     YL          YU        YBIN   IR T IO
22 1. 200. 50. 23 1 2000. 50. 1 99 10
 22 1.000E+00 2.000E+02 5.000E+01 23 1.000E+00 2.000E+03 5.000E+011 9910
TEXT FOR SPECTRUM NUMBER    1 ::
X-1 VS. Y-1
  1: X-1 VS. Y-1
ADD, DELETE, LIST, CHANGE 2 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
A
ENTER NUMBER OF 2 DIM. SPECTRA (.LE.100) >>
1
2DSP          1
HELP FOR SPEC. DEFINITION AVAILABLE. ENTER ? >
<CR>
ENTER
-X-     XL          XU        XBIN    -Y-     YL          YU        YBIN   IR T IO
1 2 2000 5 23 1 2000 50 2 99 -1
  1 2.000E+00 2.000E+03 5.000E+00 23 1.000E+00 2.000E+03 5.000E+012 99-1
WARNING *** 47180 BYTES ASSIGNED TO SPECTRA OVER THE AVAILABLE SPACE.
TEXT FOR SPECTRUM NUMBER    2 ::
<CR>
  2:
ADD, DELETE, LIST, CHANGE 2 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
C
ENTER THE NUMBER OF THE SPECRTUM TO BE MODIFIED
2
OLD VALUES OF SPECTRUM    2
-X-     XL          XU        XBIN    -Y-     YL          YU        YBIN   IR T IO
  1 2.000E+00 2.000E+03 5.000E+00 23 1.000E+00 2.000E+03 5.000E+012 99-1
ENTER NEW VALUES. BEWARE IF YOU INCREASE THE SPACE OF THIS SPECTRUM !
,,,50,,,,,1,
  1 2.000E+00 2.000E+03 5.000E+01 23 1.000E+00 2.000E+03 5.000E+011 99-1
  2:
TEXT FOR SPECTRUM NUMBER    2 ::
X-1 VS Y-1.
```

EXAMPLE 10b

```
ADD, DELETE, LIST, CHANGE 2 DIM. SPECTRA (A/D/L/C)
RETURN TO INITS BY <CR>.  >
<CR>
ENTER INIT COMMAND. For help enter HELP >>>
ZALL
ENTER INIT COMMAND. For help enter HELP >>>
```

EXAMPLE 11

```
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
WCND
ADD, DELETE, LIST WINDOW CONDITIONS (A/D/L)
RETURN TO INITS BY <CR>.  >
L
WCND       1
  0.0000E-01  2.0480E+03  0.0000E-01  2.0480E+03  2.0000E+00  2.0000E+00
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
WCND
ADD, DELETE, LIST WINDOW CONDITIONS (A/D/L)
RETURN TO INITS BY <CR>.  >
A
ENTER NUMBER OF 2 DIM. CONDITIONS (=<10) >>
1
ENTER X-LOW,X-UP,Y-LOW,Y-UP LIMITS FOR COND.  2  (FREE FORMAT) >
100. 1010.5 200.5 500.55
ADD, DELETE, LIST WINDOW CONDITIONS (A/D/L)
RETURN TO INITS BY <CR>.  >
L
WCND       2
  0.0000E-01  2.0480E+03  0.0000E-01  2.0480E+03  2.0000E+00  2.0000E+00
  1.0000E+02  1.0105E+03  2.0050E+02  5.0055E+03  4.4944E+00  1.3651E+01
```

```
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
GATE
ENTER NUMBER OF GATES >>
1
ENTER NUM. OF DIFFERENT Y-COOR. DEFINE GATE.  1 >
3
ENTER IPX AND IPY (PARAMETER # )  >
3 4
ENTER 2 DIM. WINDOW CONDITION #  >
1
ENTER   3 SETS OF COORD. (Y,XL,XU) (FREE FORMAT) >
111.1 20. 50.
500. 120.5 300.
600. 5.2 120.1
GAT2       1
GAT2       3     3     4     1
  1.1110E+02  2.0000E+01  5.0000E+01  5.0000E+02  1.2050E+02  3.0000E+02
  6.0000E+02  5.2000E+00  1.2010E+02
```

EXAMPLE 12a

```
ENTER INIT COMMAND. FOR HELP ENTER HELP >>>
TSTX
ENTER NUMBER OF TESTS (DEFAULT-  0) >
1
NUMBER OF TESTS SETTED TO   1
ENTER NUMBER OF TEST"S TEXTS TO BE DEFIENED >>
1
TSTX          1
ENTER "TEST NUMBER" AND "TEST-TEXT" FOR   1 TESTS
- FORMAT (I5,1X,10A4) >
 ##### TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
     1 COINCIDENCE EVENTS IN F1+F2 AND START-D.
    1 COINCIDENCE EVENTS IN F1+F2 AND START-D.


ENTER INIT COMMAND. For help enter HELP >>>
SSTS
ADD, CHANGE LIST OR DELETE SIMPLE TESTS (A/C/L/D) ?
<CR> - RETURN TO INIT.
L
SIMPLE TEST NUMBER:  1  APPLY ON EACH EVENT.
(ELEMENTS ARE GIVEN BELOW BY (DIM.,GATE#,PARM#) ).
TEST(100) =
 (1  1  1).AND.(1  1  4).
ADD, CHANGE LIST OR DELETE SIMPLE TESTS (A/C/L/D) ?
<CR> - RETURN TO INIT.
A
ENTER NUMBER OF ELEMENTS IN GROUP1 (DEFAULT: 0) >
3
ENTER DIMENSION OF ELEMENT #  1 (0/1/2) >
DEFAULT: *
0
ENTER TEST NUMBER (DEFAULT:  1) >
100
ENTER DIMENSION OF ELEMENT #  2 (0/1/2) >
DEFAULT: *
1
ENTER 1 DIM. COND # AND PARM # (DEFAULT: 1   0) >
2 5
ENTER DIMENSION OF ELEMENT #  3 (0/1/2) >
DEFAULT: *
2
ENTER 2 DIM. GATE #  (DEFAULT:  1) >
21
ENTER OPERATOR APPLIED WHITHIN THE ELEMENTS OF THE GROUP (OR/AND) >
DEFAULT: OR
AND
ENTER NUMBER OF ELEMENTS IN GROUP2 (DEFAULT: 0) >
1
ENTER DIMENSION OF ELEMENT #  1 (0/1/2) >
DEFAULT: *
2
ENTER 2 DIM. GATE #  (DEFAULT:  1) >
22
ENTER OPERATOR APPLIED BETWEEN THE GROUPS (AND/OR) >
DEFAULT: AND
OR
```

EXAMPLE 12b

```
APPLY .NOT. OPERATOR ON THE TEST JUST DEFINED ? (N/Y)
DEFAULT: N
Y
ENTER TEST # UNDER WHICH THE RESULT OF THIS SIMPLE TEST IS SAVED
(DEFAULT=  1) >
99
SIMPLE TEST NUMBER:  2  APPLY ON EACH EVENT.
(ELEMENTS ARE GIVEN BELOW BY (DIM.,GATE#,PARM#) ).
TEST( 99) =
          .NOT.
 (0100  0).AND.(1  2  5).AND.(2 21  0).
          .OR  .
 (2 22  0).
ADD, CHANGE LIST OR DELETE SIMPLE TESTS (A/C/L/D) ?
<CR> - RETURN TO INIT.
L
SIMPLE TEST NUMBER:  1  APPLY ON EACH EVENT.
(ELEMENTS ARE GIVEN BELOW BY (DIM.,GATE#,PARM#) ).
TEST(100) =
 (1  1  1).AND.(1  1  4).
SIMPLE TEST NUMBER:  2  APPLY ON EACH EVENT.
(ELEMENTS ARE GIVEN BELOW BY (DIM.,GATE#,PARM#) ).
TEST( 99) =
          .NOT.
 (0100  0).AND.(1  2  5).AND.(2 21  0).
          .OR  .
 (2 22  0).
ADD, CHANGE LIST OR DELETE SIMPLE TESTS (A/C/L/D) ?
<CR> - RETURN TO INIT.

ENTER INIT COMMAND. For help enter HELP >>>



×
ENTER INIT COMMAND. For help enter HELP >>>
VIRS
ADD,CHANGE OR LIST 2-DIM VIRTUAL DISPLAY SETUPS (A/C/L)
DEFAULT-RETURN TO THE CALLING ROUTINE
L

*** VIRTUAL-DISPLAY-SETUP NUMBER  1 ***
THE NUMBER OF DISPLAYS WITHIN THIS SETUP IS  2

* DISPLAY NUMBER  1 *
PARAMETER # 11-MASS VS. PARAMETER # 10-E-LG
RANGE:  [ 0.000E-01, 1.000E+02)×[ 0.000E-01, 2.200E+02)
THE DISPLAY IS NOT GATED BY ANY TEST
* DISPLAY NUMBER  2 *
PARAMETER # 21-     VS. PARAMETER # 10-E-LG
RANGE:  [ 0.000E-01, 1.000E+02)×[ 0.000E-01, 2.200E+02)
THE DISPLAY IS NOT GATED BY ANY TEST

*** VIRTUAL-DISPLAY-SETUP NUMBER  2 ***
THE NUMBER OF DISPLAYS WITHIN THIS SETUP IS  1

* DISPLAY NUMBER  1 *
PARAMETER # 21-     VS. PARAMETER #  4-E-LG
RANGE:  [ 0.000E-01, 1.000E+02)×[ 0.000E-01, 4.096E+03)
```

EXAMPLE 12c

```
THE DISPLAY IS GATED BY TEST NUMBER 100
ADD,CHANGE OR LIST 2-DIM VIRTUAL DISPLAY SETUPS (A/C/L)
DEFAULT-RETURN TO THE CALLING ROUTINE
C
ENTER VIRTUAL-DISPLAY-SETUP NUMBER TO BE CHANGED >
2
SETUP  2: ENTER NUMBER OF VIRTUAL DISPLAYS (N=<12)
DEFAULT VALUE =     1

VIRD #  1  ENTER: X AND Y PARAMETER NUMBER
DEFAULT VALUE =   21     4
21 10
ENTER W-COND #. NW=0: RANGE-[0,4096)x[0,4096)
IF NW<0 THE RANGE CAN BE ENTERED EXPLICITLY. (DEFAULT=-1) >
<CR>
ENTER X-LOW,X-UP,Y-LOW,Y-UP FOR PLOT # 1
DEFAULT VALUE =   0.000E-01   1.000E+02   0.000E-01   4.096E+03
0 100 0 220.
ENTER THE TEST NUMBER WHICH GATES THIS DISPLAY.
IF 0 THE DISPLAY IS NOT GATED.
DEFAULT VALUE =  100
<CR>
ADD,CHANGE OR LIST 2-DIM VIRTUAL DISPLAY SETUPS (A/C/L)
DEFAULT-RETURN TO THE CALLING ROUTINE
<CR>
ENTER INIT COMMAND. For help enter HELP >>>
```

EXAMPLE 13

```
ANLD
ANLD
*** IGOTO,IPASS ***     3     1
ENTER "YFSR" LOAD-MODULE NAME. DEFAULT: WCOO
CSN3RD
ENTER LOAD-LIBRARY FILE NAME. DEFAULT: TM
WRONG QUICK-LOAD MODULE NAMES !
ENTER "YFSR" LOAD-MODULE NAME. DEFAULT: WCOO
ENTER LOAD-LIBRARY FILE NAME. DEFAULT: TM

A
ANLDAT >>>

HELP
*** LIST OF COMMANDS ***
1) DISP - DISPLAY SPECTRA.
2) VIRD - SET 2 DIM. VIRTUAL DISPLAY.
3) CONT - CONTINUE
4) SETE - RETURN TO SETEXP ROUTINE
5) INIT - RETURN TO INITS ROUTINE

ANLDAT >>>
```

CSN3RD does not exist on FILE TM

Let us use the default YFSR

Interrupt - return to ANLDAT promp

EXAMPLE 14

```
CONT
V
*** IGOTO,IPASS ***     6     2
VIRD >
```

Interrupt - switch to VIRD overlay

```
SETV
ENTER VIRTUAL-DISPLAY SETUP NUMBER >
DEFAULT: ENABLE TO ADD, CHANGE, LIST AND  DELETE SETUPS.
1
*** VIRTUAL-DISPLAY-SETUP NUMBER  1 ***
 1: 11-MASS VS. 10-E-LG,TEST  0,[ 0.000E-01, 1.000E+02)x[ 0.000E-01, 2.200E+02
 2: 21-     VS. 10-E-LG,TEST  0,[ 0.000E-01, 1.000E+02)x[ 0.000E-01, 2.200E+02
*** IGOTO,IPASS ***  3     2
V
*** IGOTO,IPASS ***     6     2
VIRD >

HELP
*** VIRD MENU ***
SETV - SET VIRTUAL DISPLAY
REFR - REFRESH VIRTUAL DISPLAY
CURS - ENABLE CURSOR
GATE - SET, MODIFY, DELETE, OR REPLOT 2-DIM GATES
WGAT - WRITE COORD. OF LAST DEFINED GATE ON PRINTER
LGAT - LIST COORD. OF ALL GATES ON PRINTER

LINR - SET, MODIFY OR DELETE LINEARIZATION
CONT - CONTINUE PREVIOUS OPERATION
HCOP - PRODUCE HARD-COPY OF THE CURRENT PLOT
ANLD - RETURN TO ANLDAT
INIT - SWITCH TO INIT
DISP - SWITCH TO DISP (PLOT AND MANIPULATE SPECTRA)
HELP - GET THIS LIST
```

EXAMPLE 15

```
ANLDAT >>>
VIRD
VIRD >
CURS
 8.600E+02, 1.588E+03)
 9.360E+02, 1.380E+03)
 1.068E+03, 1.104E+03)
 1.196E+03, 9.520E+02)
 8.120E+02, 1.020E+03)
VIRD >
HCOP
VIRD >
LINR
ENTER X-NEW COOR. FOR LINE # 1 >
200.
ENTER X-NEW COOR. FOR LINE # 2 >
1800.
ENTER PARM. # FOR THE LIN. COORD. (DEF=  3) >
99
VIRD >
GATE
ADD,CHANGE,DELETE,OR REPLOT  A 2-DIM GATE ?
 (A/C/D/P - NO DEFAULT)
D
GATE DELETED, 32 GATES REMAIN.
VIRD >
GATE
ADD,CHANGE,DELETE,OR REPLOT  A 2-DIM GATE ?
 (A/C/D/P - NO DEFAULT)
P
ENTER GATE NUMBER. IF 0 ALL GATES WILL BE PLOTTED
33
ENTER PLOT NUMBER (IF 0 THE PRESENT ASSIGNMENT ARE ASSUMED)
2
VIRD >
HCOP
VIRD >
REFR
ANLDAT >>>
VIRD
VIRD >
SORT
ANLDAT >>>
CONT
VIRD >
PGAT

ENTER GATE NUMBER. IF 0 ALL GATES WILL BE PLOTTED
33
ENTER PLOT NUMBER (IF 0 THE PRESENT ASSIGNMENT ARE ASSUMED)
2
VIRD >
CONT
```

EXAMPLE 16a

```
*** IGOTO,IPASS ***     4     2

DISP >>>
HELP
*** DSP1 MENU ***
          SP = SELECT SPECTRUM
          CL = CLEAR ALL AND SET WINDOWS TO DEFAULT VALUES
          ER = ERASE THE SCREEN
          SW = SET SCREEN WINDOW
          PO = CHOOSE PRE-SET SCREEN WINDOW
          VW = SET VIRTUAL WINDOW
          LL = SET LOG-LIN OPTION
          HD = SET HISTOGRAM-DOT OPTION
          LB = SET LABELS FOR X AND Y AXES AND SPECTRUM TITLE
          NL = ENABLES TO OMIT LABELS AND NUMBERS FROM FRAME
          ML = ENABLES TO ENTER LABELS MANUALLY FROM TERMINAL
          FR = PLOT FRAME
          PL = PLOT DATA
          ES = PLOT ERROR-BARS ON THE CURRENT SPECTRUM
          PG = PLOT GROUP OF SPECTRA (UP TO 12 SPECTRA)
          SG = SELECT A SPEC FROM THE ABOVE GROUP TO WORK ON
          PF = PLOT THE LAST FIT DONE BY "CURFIT" IN "OC"
          HC = PRODUCE A HARD-COPY OF THE SCREEN
          DV = SELECT PLOTTING DEVICE
           ** ENTER <CR> TO CONTINUE

          CU = ENABLE CURSOR
          MW = MARK WINDOWS ON LAST PLOTTED SPECTRUM
          WD = DISPLAY SPECTRUM WITHIN LAST DEFINED WINDOW
          SC = STORE THE CURRENT WINDOW AS A 1 DIM. CONDITION
          MC = MARK 1D."COND" (SET WINDOW) ON PLOTTED SPECTRUM
          OC = SPECTRUM OPERATION-CODE READING
          HE = HELP, GET THIS LIST
          AN = SWITCH TO ANLDAT
          IN = GO TO INIT PROCEDURE
          SE = GO TO SETEXP PROCEDURE

DSP1 >
```

EXAMPLE 16b

```
DISP
*** IGOTO,IPASS ***    4     1

DISP >>>
OC

ENTER OP-CODE ::
HELP
*** SPECTRUM OPERATION-CODES ***
HELP     - GET THIS LIST
DEF1DS   - DEFINE A NEW 1-DIM SPECTRUM
DEF2DS   - DEFINE A NEW 2-DIM SPECTRUM
NRMSPC   - NORMALIZE SPECTRUM
SQRTSP   - SQRT OF A SPECTRUM
PWR2SP   - POWER OF 2
DIVSPC   - DIVIDE ONE SPECTRUM BY ANOTHER
MULSPC   - MULTIPLY TWO SPECTRA
ADDSPC   - ADD TWO SPECTRA
SMOOTH   - SMOOTHING OF A SPECTRUM
DSMOTH   - DIFFERENTIATE AND SMOOTH
AVRSPC   - AVERAGE OF A SPECTRUM
MODSPC   - MODIFY CONTENT OF SPECIFIC BINS IN A SPECTRUM
GET1DS   - COPY 1-D SPEC FROM SPECTRA-BUFFER TO PLOT-BUFFER
GET2DS   - COPY 2-D SPEC FROM SPECTRA-BUFFER TO PLOT-BUFFER
DPS1DS   - COPY SPECTRUM FROM PLOT-BUFFER TO SPECTRA-BUFFER
DPS2DS   - 2-DIM SPC FROM PLOT-BUFFER TO SPECTRA-BUFFER
GET2DP   - PROJECT 2-DIM SPEC. ON 1 AXIS (RESULT:PLOT-BUF)
CURFIT   - CURVE FIT TO SPECTRUM ALREADY IN THE PLOT-BUFFER
COMBIN - SQUEEZE N BINS OF INPUT SPEC. TO 1 BIN OF OUTPUT

ENTER OP-CODE ::


DISP >>>
```

EXAMPLE 17

```
DSP1 >
SP
ENTER SPECTRUM NUMBER >
TWO ADDITIONAL NUMBERS MAY BE ENTERED:
1 - THE NUMBER OF THE ERROR-SPECTRUM
2 - THE NUMBER OF THE SPECTRUM WHERE THE X-COOR. OF YOUR SPECTRUM RESIDES.
DEFAULT: THE X-COORDINATES ARE TAKEN FROM THE SPECTRUM DEFINITION.
IF THE ERR.-SPEC. NUMBER < 0, POISSON ERRORS ARE TAKEN.
13 -1

SPECTRUM  13 IS IN THE PLOTTING BUFFER
XL= 0.0000E-01 XU= 5.0000E+00 DX= 4.0000E-02
YL= 0.0000E-01 YU= 1.2717E+00
INTEGRAL OVER ABS(SPECTRUM)= 1.4895E+01
X-SPEC:   0      ERR-SPEC:  -1
MODIFY VIRTUAL WINDOW ? (Y/N)
<CR>
DSP1 >
PL

DSP1 >
PG
ENTER # OF 1ST AND LAST SPECT. TO BE PLOTTED
DEFAULT-SPECTRA   1 TO  12
13 22


DSP1 >
SG
ENTER # OF SPECTRUM YOU WISH TO WORK ON. PRESENT SEPC # IS  22
 DEFAULT: RETURN TO DSP1.
13


DSP1 >
MW


* WINDOW-MARKS NUMBER  1
* SPECTRUM  13 FROM- 0.000E-01 TO- 0.000E-01
               AVERAGE =  0.000E-01+- 0.000E-01
               INTEGRAL =  0.000E-01

DSP1 >
```

EXAMPLE 18

```
DSP1 >>>
LL
INPUT KEY: LIN-0,YLOG-2
0
DSP1 >>>
HD
SELECT HISTOGRAM OR DOT MODE (H/D) :
D
ENTER ANY CHARACTER TO BE USE AS A SYMBOL :
+
DSP1 >>>
ER
DSP1 >>>
PL
DSP1 >>>
VW
INPUT VIRTUAL:XMIN,XMAX,YMIN,YMAX
PRESENT VALUES ARE   1.0000E+00   2.0550E+03   0.0000E-01   2.9321E+03
0 8000 0 3000
DSP1 >>>
ER
DSP1 >>>
PL
DSP1 >>>
OC
ENTER OP-CODE ::
SQRTSP
ENTER SPECT#,TRGT-SPECT# AND N-DIM(1/2) :
CH => SQRT(CH)
12 13
ENTER OP-CODE ::
<CR>
DSP1 >>>
DSP1 >>>
SE
ENTER NUMBERE OF ERROR-SPECTRUM >
13
```

EXAMPLE 19a

```
DSP1 >>>
HC
DSP1 >>>
DC
ENTER OP-CODE ::
PWR2                    ←——This command does not exist. So, return to DSP1D
DSP1 >>>
DC
ENTER OP-CODE ::
PWR2SP
ENTER SPECT#,TRGT-SPECT# AND N-DIM(1/2) :
CH => CH**2
13 13
ENTER OP-CODE ::
<CR>
DSP1 >
HD
SELECT HISTOGRAM OR DOT MODE (H/D) :
D
*** ENTER DOT-TYPE CODE (1/2/3/4/5/6) ***
 1 - OPEN SQUARE
 2 - CLOSED SQUARE
 3 - OPEN CIRCLE
 4 - CLOSED CIRCLE
 5 - OPEN TRIANGLE
 6 - CLOSED TRIANGLE
 7 - A BAR  ( - )
4

DSP1 >
HD
SELECT HISTOGRAM OR DOT MODE (H/D) :
<CR>
*** ENTER LINE-TYPE CODE (0/1/2/3/4) ***
 0 - A SOLID LINE
 1 - A DOTTED LINE
 2 - A DASH-DOT LINE
 3 - A SHORT-DASHED LINE
 4 - A LONG-DASHED LINE

<CR>
DSP1 >
```

```
OC

ENTER OP-CODE ::
DEF1DS                                          ← Define new 1 Dim spectrum
ENTER: PARM.#,X-LOW,X-UP,DX,TYPE-INT/REAL(1/2)
BE CAREFUL ! INPUT PARAMETERS ARE NOT CHECKED FOR ERRORS:
3 0. 280. 3. 1
NEW SPECTRUM NUMBER IS     1

ENTER OP-CODE ::
DEF1DS
ENTER: PARM.#,X-LOW,X-UP,DX,TYPE-INT/REAL(1/2)
BE CAREFUL ! INPUT PARAMETERS ARE NOT CHECKED FOR ERRORS:
3 0. 280. 3. 1
NEW SPECTRUM NUMBER IS     2

ENTER OP-CODE ::
GET2DP                                          ← Project 2 Dim spet #6 on parm #3
ENTER 2-DIM SPECT#, PARM# OF PROJ-AXIS, X-LOW, X-UP, Y-LOW, Y-UP:
X-LOW,X-UP - THE LIMITS ON THE X-AXIS
Y-LOW,Y-UP - THE LIMITS ON THE Y-AXIS
DEFAULT VALUES    0    0  0.000E-01  0.000E-01  0.000E-01  0.000E-01

6 3 0 3 0 280

ENTER OP-CODE ::
GET2DP
ENTER 2-DIM SPECT#, PARM# OF PROJ-AXIS, X-LOW, X-UP, Y-LOW, Y-UP:
X-LOW,X-UP - THE LIMITS ON THE X-AXIS
Y-LOW,Y-UP - THE LIMITS ON THE Y-AXIS
DEFAULT VALUES    6    3  0.000E-01  3.000E+00  0.000E-01  2.800E+02
```

(NOTE that SPCOPR remembers the previous input)

```
ENTER OP-CODE ::
DPS1DS                                          ← Save the projected spectrum for
ENTER SPECT# :                                    further use
1

ENTER OP-CODE ::
GET2DP                                          ← Project 2-DIM spect #14, which contains
                                                  the variances of spectrum 6 on parm #3
ENTER 2-DIM SPECT#, PARM# OF PROJ-AXIS, X-LOW, X-UP, Y-LOW, Y-UP:
X-LOW,X-UP - THE LIMITS ON THE X-AXIS
Y-LOW,Y-UP - THE LIMITS ON THE Y-AXIS
DEFAULT VALUES    6    3  0.000E-01  3.000E+00  0.000E-01  2.800E+02
14

ENTER OP-CODE ::
DPS1DS
ENTER SPECT# :
2                                               ← Save it in 1-DIM spect #2

ENTER OP-CODE ::
SQRTSP                                          ← And take the SQRT of spect #2 to get the
ENTER SPECT#,TRGT-SPECT# AND N-DIM(1/2) :         errors on spect #1
CH => SQRT(CH)
2 2 1

ENTER OP-CODE ::
```

```
DSP1 >
SP
ENTER SPECTRUM NUMBER >
TWO ADDITIONAL NUMBERS MAY BE ENTERED:
1 - THE NUMBER OF THE ERROR-SPECTRUM
2 - THE NUMBER OF THE SPECTRUM WHERE THE X-COOR. OF YOUR SPECTRUM RESIDES.
DEFAULT: THE X-COORDINATES ARE TAKEN FROM THE SPECTRUM DEFINITION.
IF THE ERR.-SPEC. NUMBER < 0, POISSON ERRORS ARE TAKEN.
1

SPECTRUM   1 IS IN THE PLOTTING BUFFER
XL= 0.0000E-01 XU= 5.0000E+00 DX= 4.0000E-02
YL= 0.0000E-01 YU= 9.3142E+00
INTEGRAL OVER ABS(SPECTRUM)= 1.4970E+02
X-SPEC:   0     ERR-SPEC:   0
MODIFY VIRTUAL WINDOW ? (Y/N)
<CR>
   DSP1 >
   PO
   ENTER SPECTRUM POSITION, FOR INDEX ENTER NPOS<0 :
   -1
   *** SPECTRUM POSITION CODE ***
    0 - FULL SCREEN
    1 - HALF SCREEN, LEFT SIDE
    2 - HALF SCREEN, RIGHT SIDE
    3 - HALF SCREEN, BOTTOM
    4 - HALF SCREEN, TOP
    5 - QUARTER SCREEN, LEFT-BOTTOM
    6 - QUARTER SCREEN, RIGHT-BOTTOM
    7 - QUARTER SCREEN, LEFT-TOP
    8 - QUARTER SCREEN, RIGHT-TOP
   ENTER SPECTRUM POSITION, FOR INDEX ENTER NPOS<0 :
   1

   DSP1 >
   ER

   DSP1 >                          <------------ Plot spect #1
   PL

   DSP1 >
   PO
   ENTER SPECTRUM POSITION, FOR INDEX ENTER NPOS<0 :
   2

   DSP1 >
   ES                              <------------ Plot the errors saved in spect #2
   ENTER NUMBER OF ERROR-SPECTRUM.
   IF 0 STATISTICAL ERRORS ARE ASSUMED (SQRT(F(X)) >
   2


   DSP1 >
   LB                              <------------ Add labels
   CHANGE LABEL OF X, Y AXIS OR TITLE (X/Y/T) >
   RETURN TO "DISP" MENU BY <CR>.
   Y
   INPUT LABEL Y (40 CHAR)
   DEFAULT
                    C O U N T S
```

```
CHANGE LABEL OF X, Y AXIS OR TITLE (X/Y/T) >
RETURN TO "DISP" MENU BY <CR>.
T
INPUT TITLE (40 CHAR)
DEFAULT
PLOT OF THE ERRORS IN SPEC#1.
CHANGE LABEL OF X, Y AXIS OR TITLE (X/Y/T) >
RETURN TO "DISP" MENU BY <CR>.


DSP1 >                          <------------ Replot the frame
FR

DSP1 >
HC

DSP1 >
MW                              <------------ Mark window


* WINDOW-MARKS NUMBER  1
* SPECTRUM    1 FROM- 2.264E+01 TO- 2.191E+02
               AVERAGE =  1.229E+02+- 3.381E+01
               INTEGRAL =  1.118E+03

DSP1 >
FR

DSP1 >                          <------------ Switch to SPCOPR
OC
```

```
ENTER OP-CODE ::
CURFIT
ENTER SPECTRUM NUMBERS OF Y AND X SPECTRA.
DEFAULT: N(Y) =  13  N(X) =   0
<CR>
Y-SPEC. =  13  X-SPEC. =   0
ENTER NUMBER OF ERROR-SPECTRUM.
IF < 0 STATISTICAL ERRORS ARE ASSUMED
IF = 0 ALL BINS HAVE THE SAME PROBABILITY
NOTE: ERROR BARS IN PLOT BUFFER ARE REPLACED BY THE
CORRESPONDING WEIGHTS, BUT THE ERROR SPEC. NUMBER IS
REMEMBERED. DEFAULT =   0
-1
WEIGHTING MODE = POISSON
ENTER LOWER AND UPPER LIMITS OF THE RANGE OVER WHICH THE FIT WILL BE MADE.
AS A DEFAULT THE MARKED-WINDOW RANGE IS TAKEN
XL=XU THE FULL SPECTRUM RANGE IS TAKEN
1.1 3.02
SEARCH BETWEEN XL= 1.100E+00 (BIN   29) AND XU= 3.020E+00 (BIN   76)
ENTER FUNCTION :
GAUSS  : (A4+A5*X+A6*X**2) + A1*EXP(-.5*((X-A3)/A2)**2))
MAXWELL- A1*SQRT(X-A4) * EXP( (-(X-A4)+A3*(X-A4)**.5)/A2 )
LEGENDRE- A0 + SUM (AL*PL(COS X)), L < OR = 20
FCURVE   - USER-SUPPLIED FUNCTION (MUST BE CALLED FCURVE)
GAUSS
ENTER NUMBER OF TERMS TO BE FITTED. DEFAULT= 3 >
<CR>
THE NUMBER OF DEGREES OF FREEDOM IS    45
1ST CHOICE OF TERMS:
1.272E+00 3.373E-01 1.879E+00
DO YOU WISH TO MODIFY THESE TERMS ? (N/Y)

    INITIAL CHI-SQUARE =    8.999894E-01
    TOTAL NO. OF ATTEMPTS =      4 LAMBDA=  1.000E-05
*** CURVE-FIT RESULTS ***
A 1= 4.869E-01+- 2.253E-01      A 2= 2.684E-01+- 1.110E-01
A 3= 1.967E+00+- 1.233E-01      A
CHI-SQR/D.O.F= 1.509E-01    NUMBER OF ITERATIONS =    4
ENTER SPECTRUM NUMBER WHERE THE FIT WILL BE DEPOSITED
IF =< 0 THE FIT IS NOT SAVED !
<CR>
ENTER OP-CODE ::
<CR>
DISP >>>
PF

DSP1 >


OC
```

```
ENTER OP-CODE ::          <------------  Perform curve fit
CURFIT
ENTER LOWER AND UPPER LIMITS OF THE RANGE OVER WHICH THE FIT WILL BE MADE.
AS A DEFAULT THE MARKED-WINDOW RANGE IS TAKEN
XL=XU THE FULL SPECTRUM RANGE IS TAKEN

SEARCH BETWEEN XL= 2.264E+01 (BIN    8) AND XU= 2.191E+02 (BIN   74)
ENTER NUMBER OF ERROR-SPECTRUM.
IF 0 STATISTICAL ERRORS ARE ASSUMED (DEFAULT)
IF < 0 ALL BINS HAVE THE SAME PROBABILITY
2
ENTER FUNCTION :
GAUSS  : (A4+A5*X+A6*X**2) + A1*EXP(-.5*((X-A3)/A2)**2))
MAXWELL- A1*SQRT(X-A4) * EXP( (-(X-A4)+A3*(X-A4)**.5)/A2 )
FCURVE   - USER-SUPPLIED FUNCTION (MUST BE CALLED FCURVE)
GAUSS
ENTER NUMBER OF TERMS TO BE FITTED. DEFAULT= 3 >

THE NUMBER OF DEGREES OF FREEDOM IS    64
1ST CHOICE OF TERMS:
4.500E+01 3.381E+01 1.229E+02
DO YOU WISH TO MODIFY THESE TERMS ? (N/Y)

    INITIAL CHI-SQUARE =    2.577056E+00
    TOTAL NO. OF ATTEMPS =      3 LAMDA=  1.000E-04
*** CURVE-FIT RESULTS ***
A 1= 3.713E+01+- 1.720E+00      A 2= 3.193E+01+- 8.131E-01
A 3= 1.207E+02+- 1.243E+00      A
CHI-SQR/D.O.F= 1.389E+00    NUMBER OF ITERATIONS =    3
ENTER SPECTRUM NUMBER WHERE THE FIT WILL BE DEPOSITED
IF =< 0 THE FIT IS NOT SAVED !


ENTER OP-CODE ::
PF                        <------------  Plot the fit

DISP >>>
HC
                  )
                  )       <------------  Make two copies.  Why not?
DSP1 >            )
HC
                                         Enter interactively the desired label.
DSP1 >                    <------------
ML
ENTER POSITION (IX,IY-IN SCREEN UNITS)
2500 100
ENTER TEXT (UP TO 40 CHARACTERS)
                                         This is a wrong position.
                          <------------  Don't enter text,
ENTER CHARACTER SIZE (4/3/2/1) (4-SMALLEST)

X AND Y SIZE OF CHARACTERS (IN SCREEN UNITS)  31  48
ENTER ANGLE OF ROTATION (0 OR 90 DEG FOR TEK.)


DSP1 >
                          <------------  try again.
ML
ENTER POSITION (IX,IY-IN SCREEN UNITS)
3000 2800
ENTER TEXT (UP TO 40 CHARACTERS)
```

EXAMPLE 19f

```
CURFIT demonstration
ENTER CHARACTER SIZE (4/3/2/1) (4-SMALLEST)
3
X AND Y SIZE OF CHARACTERS (IN SCREEN UNITS)  34  53
ENTER ANGLE OF ROTATION (0 OR 90 DEG FOR TEK.)
<CR>
DSP1 >
HC                    <------ Make a hard copy

DSP1 >
HC                    <------ Make a second one.  Why not?

DSP1 >
IN
*** IGOTO,IPASS ***    2    2

ENTER INIT COMMAND. For help enter HELP >>>
ANLD
ANLD
*** IGOTO,IPASS ***    3    2
```

EXAMPLE 20

```
   1            U C      CODE    ***   RUN NUMBER  943
EVENTS READ=  21558      EVENTS WRITTEN=  20897
 # OF BLOCKS READ=   863  # OF BLOCKS WRITTEN=     0
REJECTED EVENTS      2     1   383   275     0     0     0     0     0     0
ACCEPTED EVENTS      0  5800 12332  2062     0   703     0     0     0     0

              *** NUMBER OF EVENTS WHICH PASSED THE DEFINED TESTS ***
            TEST NUMBER                                NUMBER OF EVENTS
               1  HI BIT REGISTER NOT SETTED PROPERLY, REJ          2
               2  BOTH HI TELESCOPES FIRED, REJ.                    1
               3  TEL # 1 : Z NOT DEFINED. REJ.                   383
               4  TEL # 2 : Z NOT DEFINED. REJ.                   275
               5  SC. BIT REG. DOES NOT AGREE WITH ADC,ACC       5004
               6  SC#1 AND #7 FIRED SIMULTANUSLY, ACC.            954
```

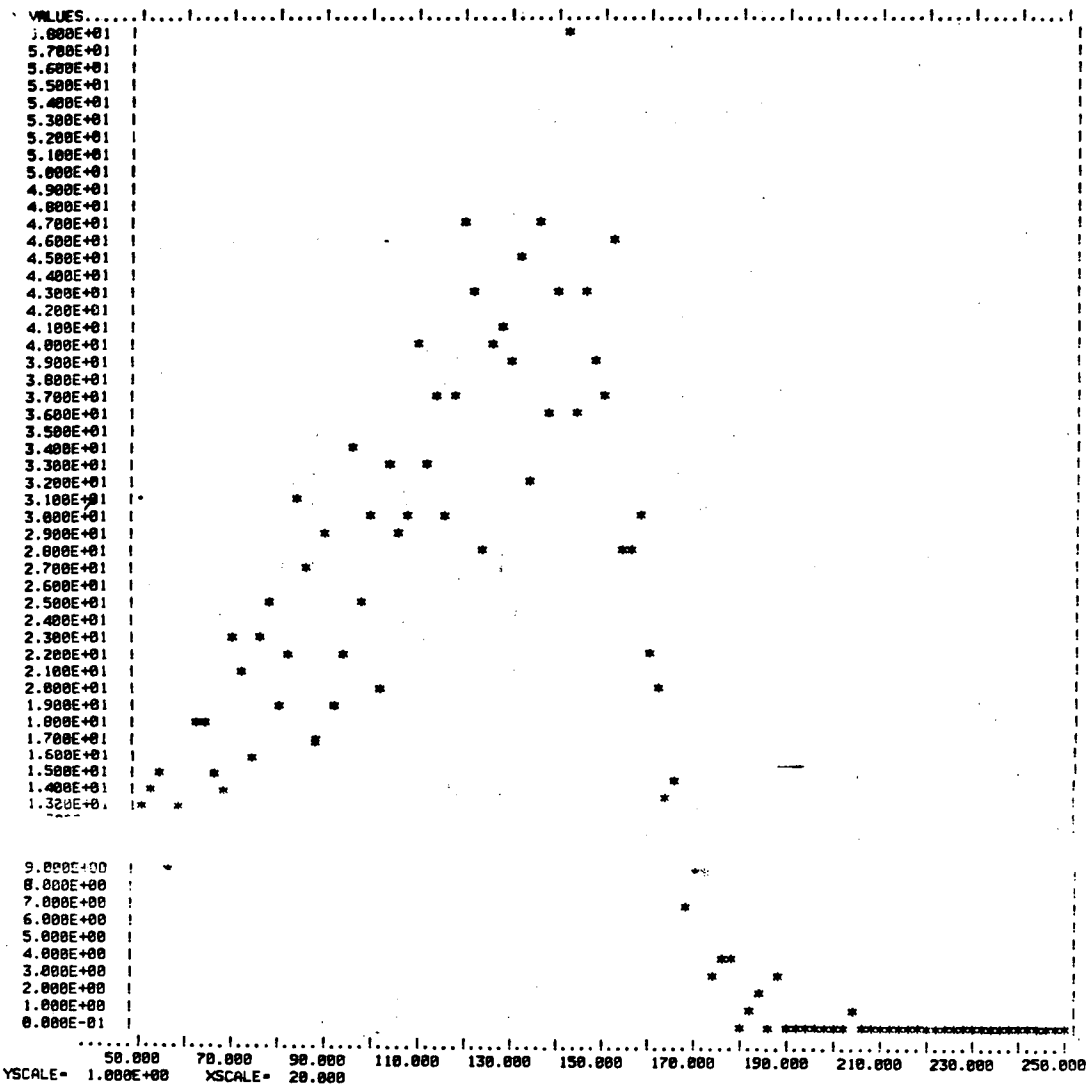
```
*** SPECTRUM NUMBER    1     HI - Z  DISTRIBUTION.                    ***
    ------------------------------------------------------------------

AREA-INTEGRATED OVER PARAMETER ( 31)  -Z-    FROM-    2.0000  TO-   10.0000  IN STEPS OF    1.0000     IS : 3.0301E+05

*** AVERAGE OF SPECTRUM NUMBER :    1 FROM: 2.0000E+00  TO: 1.0000E+01  IS::: 8.1543E+00+- 1.1307E+00
 -Z-  FROM          TO
    2.0000--    9.0000     6.4440E+03  2.5380E+03  2.1360E+03  4.1570E+03  1.1406E+04  1.0297E+04  2.6535E+05  6.8900E+02
   10.0000--   10.0000     0.0000E-01

ACCUMULATIVE SPECTRUM

    2.0000--    9.0000     6.4440E+03  8.9820E+03  1.1118E+04  1.5275E+04  2.6681E+04  3.6978E+04  3.0233E+05  3.0301E+05
   10.0000--   10.0000     3.0301E+05


*****************************************************************************************************************************
*** SPECTRUM NUMBER    2     HI - Z DIST. FOR PURE SINGLES.(BY B-REG) ***
    ------------------------------------------------------------------

AREA-INTEGRATED OVER PARAMETER ( 31)  -Z-    FROM-    2.0000  TO-   10.0000  IN STEPS OF    1.0000     IS : 2.6460E+05

*** AVERAGE OF SPECTRUM NUMBER :    2 FROM: 2.0000E+00  TO: 1.0000E+01  IS::: 8.3773E+00+- 5.6682E-01
 -Z-  FROM          TO
    2.0000--    9.0000     5.6200E+02  4.7600E+02  6.0300E+02  1.6850E+03  5.7970E+03  8.2730E+03  2.4660E+05  6.0400E+02
   10.0000--   10.0000     0.0000E-01

ACCUMULATIVE SPECTRUM

    2.0000--    9.0000     5.6200E+02  1.0380E+03  1.6410E+03  3.3260E+03  9.1230E+03  1.7396E+04  2.6399E+05  2.6460E+05
   10.0000--   10.0000     2.6460E+05


*****************************************************************************************************************************
*** SPECTRUM NUMBER    3     MULTIPLICITY. (BY B-REG.).               ***
    ------------------------------------------------------------------

AREA-INTEGRATED OVER PARAMETER ( 41) MULT    FROM-    0.0000  TO-    7.0000  IN STEPS OF    1.0000     IS : 3.0301E+05

*** AVERAGE OF SPECTRUM NUMBER :    3 FROM: 0.0000E-01  TO: 7.0000E+00  IS::: 6.5910E-01+- 4.6298E-01
MULT  FROM          TO
    0.0000--    7.0000     2.6460E+05  3.0771E+04  5.8470E+03  1.5050E+03  2.7500E+02  1.6000E+01  3.0000E+00  0.0000E-01

ACCUMULATIVE SPECTRUM

    0.0000--    7.0000     2.6460E+05  2.9537E+05  3.0122E+05  3.0272E+05  3.0300E+05  3.0301E+05  3.0301E+05  3.0301E+05
```

*** SPECTRUM NUMBER 9 E-HI [illegible] COUNT EVENTS. ***

EXAMPLE 22

AREA-INTEGRATED OVER PARAMETER (32) E-HI FROM- 50.0000 TO- 250.0000 IN STEPS OF 2.0000 IS : 1.7130E+03

*** AVERAGE OF SPECTRUM NUMBER : 9 FROM: 5.0000E+01 TO: 2.5000E+02 IS::: 1.1635E+02+- 3.0497E+01



HI FROM	TO								
50.0000--	64.0000	1.3000E+01	1.4000E+01	1.5000E+01	9.0000E+00	1.3000E+01	1.2000E+01	1.0000E+01	1.0000E+01
66.0000--	80.0000	1.5000E+01	1.4000E+01	2.3000E+01	2.1000E+01	1.6000E+01	2.3000E+01	2.5000E+01	1.9000E+01
82.0000--	96.0000	2.2000E+01	3.1000E+01	2.7000E+01	1.7000E+01	2.9000E+01	1.9000E+01	2.2000E+01	3.4000E+01
98.0000--	112.0000	2.5000E+01	3.0000E+01	2.0000E+01	3.3000E+01	2.9000E+01	3.0000E+01	4.0000E+01	3.3000E+01
114.0000--	128.0000	3.7000E+01	3.0000E+01	3.7000E+01	4.7000E+01	4.3000E+01	2.8000E+01	4.0000E+01	4.1000E+01
130.0000--	144.0000	3.9000E+01	4.5000E+01	3.2000E+01	4.7000E+01	3.6000E+01	4.3000E+01	5.0000E+01	3.6000E+01
146.0000--	160.0000	4.3000E+01	3.9000E+01	3.7000E+01	4.6000E+01	2.8000E+01	2.0000E+01	3.0000E+01	2.2000E+01
162.0000--	176.0000	2.0000E+01	1.4000E+01	1.5000E+01	7.0000E+00	9.0000E+00	9.0000E+00	3.0000E+00	4.0000E+00
178.0000--	192.0000	4.0000E+00	0.0000E-01	1.0000E+00	2.0000E+00	0.0000E-01	3.0000E+00	0.0000E-01	0.0000E-01
194.0000--	208.0000	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	1.0000E+00	0.0000E-01	0.0000E-01
210.0000--	224.0000	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
226.0000--	240.0000	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
242.0000--	250.0000	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01			

ACCUMULATIVE SPECTRUM

50.0000--	64.0000	1.3000E+01	2.7000E+01	4.2000E+01	5.1000E+01	6.4000E+01	7.6000E+01	9.4000E+01	1.1200E+02
66.0000--	80.0000	1.2700E+02	1.4100E+02	1.6400E+02	1.8500E+02	2.0100E+02	2.2400E+02	2.4900E+02	2.6900E+02
82.0000--	96.0000	2.9000E+02	3.2100E+02	3.4800E+02	3.6500E+02	3.9400E+02	4.1300E+02	4.3500E+02	4.6900E+02
98.0000--	112.0000	4.9400E+02	5.2400E+02	5.4400E+02	5.7700E+02	6.0600E+02	6.3600E+02	6.7600E+02	7.0900E+02
114.0000--	128.0000	7.4600E+02	7.7600E+02	8.1300E+02	8.6000E+02	9.0300E+02	9.3100E+02	9.7100E+02	1.0120E+03
130.0000--	144.0000	1.0510E+03	1.0960E+03	1.1200E+03	1.1750E+03	1.2110E+03	1.2540E+03	1.3120E+03	1.3480E+03
146.0000--	160.0000	1.3910E+03	1.4300E+03	1.4670E+03	1.5130E+03	1.5410E+03	1.5690E+03	1.5990E+03	1.6210E+03
162.0000--	176.0000	1.6410E+03	1.6550E+03	1.6700E+03	1.6770E+03	1.6360E+03	1.6950E+03	1.6980E+03	1.7020E+03
178.0000--	192.0000	1.7060E+03	1.7060E+03	1.7070E+03	1.7090E+03	1.7090E+03	1.7120E+03	1.7120E+03	1.7120E+03
194.0000--	208.0000	1.7120E+03	1.7120E+03	1.7120E+03	1.7120E+03	1.7120E+03	1.7130E+03	1.7130E+03	1.7130E+03
210.0000--	224.0000	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03
226.0000--	240.0000	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03
242.0000--	250.0000	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03	1.7130E+03			

```
*** TWO DIM. SPECTRUM NUMBER    1     MULTIPLICITY - Z VS. SC*.                    ***
-----------------------------------------------------------------------------------

AREA-INTEGRATING OVER PARAMETER ( 31)  -Z-    FROM-    2.0000  TO-    9.0000  IN STEPS OF    1.0000
                 AND PARAMETER ( 42) SC *     FROM-    1.0000  TO-    7.0000  IN STEPS OF    1.0000     IS : 3.9684E+04

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
:          :
                                                   - -Z- -
:    SC *   : 2.000E+00 : 3.000E+00 : 4.000E+00 : 5.000E+00 : 6.000E+00 : 7.000E+00 : 8.000E+00 : 9.000E+00 :
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
:     1.0000: 3.3150E+03 : 1.1750E+03 : 7.8100E+02 : 1.2570E+03 : 3.0720E+03 : 9.8500E+02 : 1.1801E+04 : 3.9000E+01 :
:     2.0000: 2.5420E+03 : 7.6800E+02 : 5.8500E+02 : 8.9500E+02 : 1.7620E+03 : 6.2900E+02 : 2.4310E+03 : 2.6000E+01 :
:     3.0000: 6.4400E+02 : 1.6800E+02 : 1.0600E+02 : 1.5200E+02 : 1.8400E+02 : 8.9000E+01 : 4.7600E+02 : 6.0000E+00 :
:     4.0000: 8.4600E+02 : 2.0900E+02 : 1.1800E+02 : 1.4300E+02 : 1.9200E+02 : 6.7000E+01 : 3.5800E+02 : 2.0000E+00 :
:     5.0000: 5.9200E+02 : 1.1000E+02 : 9.0000E+01 : 7.9000E+01 : 1.5700E+02 : 8.1000E+01 : 3.3100E+02 : 5.0000E+00 :
:     6.0000: 4.1400E+02 : 8.3000E+01 : 4.2000E+01 : 4.8000E+01 : 7.7000E+01 : 3.2000E+01 : 2.0100E+02 : 4.0000E+00 :
:     7.0000: 1.4300E+02 : 8.7000E+01 : 4.7000E+01 : 7.6000E+01 : 3.0300E+02 : 8.1000E+01 : 7.7500E+02 : 3.0000E+00 :
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


 -Z-  FROM          TO
    2.0000--    9.0000     8.4960E+03  2.6000E+03  1.7690E+03  2.6500E+03  5.7470E+03  1.9640E+03  1.6373E+04  8.5000E+01
SC *  FROM          TO
    1.0000--    7.0000     2.2425E+04  9.6380E+03  1.8250E+03  1.9350E+03  1.4450E+03  9.0100E+02  1.5150E+03
*****************************************************************************************************************************
*** TWO DIM. SPECTRUM NUMBER    2     REGION - 1, TAC - TRUE.                      ***
-----------------------------------------------------------------------------------

AREA-INTEGRATING OVER PARAMETER ( 31)  -Z-    FROM-    2.0000  TO-    9.0000  IN STEPS OF    1.0000
                 AND PARAMETER ( 42) SC *     FROM-    1.0000  TO-    7.0000  IN STEPS OF    1.0000     IS : 1.7465E+04

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
:          :
                                                   - -Z- -
:    SC *   : 2.000E+00 : 3.000E+00 : 4.000E+00 : 5.000E+00 : 6.000E+00 : 7.000E+00 : 8.000E+00 : 9.000E+00 :
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
:     1.0000: 1.7020E+03 : 5.4500E+02 : 3.3600E+02 : 5.2900E+02 : 1.1090E+03 : 3.7200E+02 : 4.7740E+03 : 9.0000E+00 :
:     2.0000: 1.3560E+03 : 3.8200E+02 : 2.9400E+02 : 4.0600E+02 : 8.3600E+02 : 2.5600E+02 : 1.0480E+03 : 1.0000E+01 :
:     3.0000: 4.9100E+02 : 1.0500E+02 : 7.6000E+01 : 9.1000E+01 : 9.2000E+01 : 2.6000E+01 : 8.4000E+01 : 2.0000E+00 :
:     4.0000: 6.4800E+02 : 1.5700E+02 : 9.3000E+01 : 9.5000E+01 : 1.1300E+02 : 2.5000E+01 : 8.7000E+01 : 1.0000E+00 :
:     5.0000: 1.0000E+00 : 3.0000E+00 : 0.0000E-01 : 0.0000E-01 : 3.0000E+00 : 1.0000E+00 : 1.0000E+02 : 0.0000E-01 :
:     6.0000: 1.0000E+00 : 1.0000E+00 : 1.0000E+00 : 0.0000E-01 : 8.0000E+00 : 3.0000E+00 : 6.6000E+01 : 1.0000E+00 :
:     7.0000: 1.3900E+02 : 8.6000E+01 : 4.7000E+01 : 7.2000E+01 : 2.8900E+02 : 7.1000E+01 : 4.1900E+02 : 3.0000E+00 :
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


 -Z-  FROM          TO
    [illegible]    [illegible]  4.3300E+03  1.2790E+03  8.4700E+02  1.1930E+03  2.4500E+03  7.5400E+02  6.5780E+03  2.6000E+01
```

**** TWO DIM. SPECTRUM NUMBER 1 T 1 ***

EXAMPLE 24

```
AREA-INTEGRATING OVER PARAMETER ( 1)      FROM-   0.0000  TO- 4096.0000  IN STEPS OF   64.0000
                 AND PARAMETER ( 2)      FROM-   0.0000  TO- 4096.0000  IN STEPS OF   64.0000    IS : 1.6912E+04


X-    = (COLUMN # -1) * 6.400E+01 +  0.000E-01
Y-    = ( ROW   # - 1) * 6.400E+01 +  0.000E-01

-Y- (     )
           10        20        30        40        50        60        70        80        90        100       110       120
  0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123
70-----------------------------------------------------------------------------------------------------------------------------70
69          !         !         !         !         !         !         !         !         !         !         !         !  69
68          !         !         !         !         !         !         !         !         !         !         !         !  68
67          !         !         !         !         !         !         !         !         !         !         !         !  67
66          !         !         !         !         !         !         !         !         !         !         !         !  66
65          !         !         !         !         !         !         !         !         !         !         !         !  65
64          !         !         !         !         !         !         !         !         !         !         !         !  64
63 9ZY72333!1113231232658666123337478559D84C5C86AA43566874I887D7B          !         !         !         !         !         !  63
62 7KXA 3151  11 214125 11 4332152A7459387DLIFF8CC5DFACA886866214          !         !         !         !         !         !  62
61 7APMB1 11222 2     132112 313C8AHJAKG87I8AFCC3749446I747442623          !         !         !         !         !         !  61
60-74CEME6111---1121112245456FEHJD7JDKDCF8A8A4367566763212431l1-1----------------------------------------------------------60
59 51 772H711 12 11 12198F8DD8GDCGEK8DGE5DDCE8582474 151      ! 1         !         !         !         !         !         !  59
58 8 1  GE6! 421 212558COHE9GMGHGD9C7AE7785862352633l111  11 !         !         !         !         !         !         !  58
57 31 17D64!1   12398EDCDCHHMFI86A888F786F94552211121 112     !         !         !         !         !         !         !  57
56 2112M4411  1153GFAJ6PEF6DFI9FH9DDC5888A4436 1   11  1      !         !         !         !         !         !         !  56
55 3  873111l124DI8DDFHD89098FHCBCG8DA8A868352142.1!     1    !         !         !         !         !         !         !  55
54 2 1981  225A598GD067GFJFAFFLFGFFHFFF4833511    !         !         !         !         !         !         !         !  54
53  17341121384A8C9GF58C7AGIKJJLNPEIGH8FG21213 1  !         !         !         !         !         !         !         !  53
52 1 7321  38769FF977H68CKQQXTZKRQJFC51842121     !!        !         !         !         !         !         !         !  52
51 128132I6C57734C78CCJVRNZXZXWWYNDC94322!1 1     !         !         !         !         !         !         !         !  51
50-18221-457AA8AC3989PVZZZXZZZWWKI8321211--------------------------------------------------------------------------------50
49 1341 235A448598ALVYZZZZZZVNKH941321  !         !         !         !         !         !         !         !         !  49
48 1834 56745576DEOPZZZZZZZRXG872       !         !         !         !         !         !         !         !         !  48
47  51  46852888KRZZZZZZZZWIC84!12 1    !         !         !         !         !         !         !         !         !  47
46 441 249829BCQXVZZZZZZZNEA 2 !        !         !         !         !         !         !         !         !         !  46
45  111728362DEQZZZZZZZWH871   !        !         !         !         !         !         !         !         !         !  45
44 32 211422CGUZZZZZZZTDA62    !1       !         !         !         !         !         !         !         !         !  44
43 1 1224443ASZZZZZZZI87       !        !         !         !         !         !         !         !         !         !  43
42 3 146237BQZZZZZZZQ92        !   1    !         !         !         !         !         !         !         !         !  42
41 2116437ADZZZZZZZCA3         !        !       1 !         !         !         !         !         !         !         !  41
40-2-43-179GZZZZZNC3-----------------------------------------------------------------------------------------------------40
39 1 2422BSZZZZXQ441 !         !        !         !         !         !         !         !         !         !         !  39
38 2148168QZZZZL621 1!         !        !         !         !         !         !         !         !         !         !  38
37 2 1747HZZZZOD3    !         !        !         !         !         !         !         !         !         !         !  37
36 3  52AUZZZKC 1    !         !1       !         !         !         !         !         !         !         !         !  36
35 1133 SZZZZB  1    !    1    !        !         !         !         !         !         !         !         !         !  35
34 42147HZZZK3  1    1         !        !         !         !         !         !         !         !         !         !  34
33 64325WZZ0822    1 !1   1    !        !         !         !         !         !         !         !         !         !  33
32 CL548ZZZ85    1   !11       !   11   !         !         !         !         !         !         !         !         !  32
31 6MB I[illegible]     1     ! 2 111  1!     1 !   1     11  1   !         !         !         !         !         !         !  31
30-CN88ZZ[illegible]4---------11-1112--------11---2--1------------------------------------------------------------------30
29 9IEKZZ[illegible] 1    2  11  2     111      !  1     !         :         :         :         :         :         :         :  29
28 7JH[illegible]    1      1!    1    !     2 ! 1   1  !         !         !         !         !         !         !         !  28
27 68EZZY[illegible]!      1   111       !    1!2 2 1 1  1 1    1         !         !         !         !         !         !  27
26 44DSZ05 21        1 1    1  ! 1 1    ! 111     !    1  1!         !         !         !         !         !         !  26
25 616ZZ831! 1    1  11        !        !     1   !         !         !         !         !         !         !         !  25
24 53DZZ663!    1    !   1    !11       !  1   1  !         !         !         !         !         !         !         !  24
23 631ZR411!1   1    !      1 1 !1      !         !         !         !         !         !         !         !         !  23
22 441ZG15 1       1111 1      !        !         !         !         !         !         !         !         !         !  22
21 54GZD64 !     2   !   1     !  1   1 !         !         !         !         !         !         !         !         !  21
20-55PZ25----------1--1--------------------------------------------------------------------------------------------------20
19 32YZ 1 1!     1   !1        !        !         !         !         !         !         !         !         !         !  19
18 22ZQ 2  1   1     !         !11    1 !         !         !         !         !         !         !         !         !  18
17  6ZJ 11 !  1      !   11    1        !     1   !         !         !         !         !         !         !         !  17
16 3AZE1   !         !         1   1    !         !         !         !         !         !         !         !         !  16
15 4AZ221  !         !         1        !         !         !         !         !         !         !         !         !  15
14 5CZ2    !         !1       1!       1!         !         !         !         !         !         !         !         !  14
13 2KWI1   !  1    1 ! 12     1!        !         !  1      !         !         !         !         !         !         !  13
12 7VV 1   !   1   1 !         !        !         !         !         !         !         !         !         !         !  12
11 30I   1 !        2!  1     1!        !         !         !         !         !         !         !         !         !  11
10-ZZG-------------1--1------------------------------------------------------------------------------------------------10
 9 4Z6     !     1 1 !  1      !        !         !         !         !         !         !         !         !         !   9
 8 4W61    !     3 1 11        !        !         !         !         !         !         !         !         !         !   8
 7 723 1   !    11   2    1    !        !         !         !         !         !         !         !         !         !   7
 6 [illegible]    ! !    11!   !        !         !         !         !         !         !         !         !         !   6
 5 [illegible]    !  !   1   111!       !         !         !         !         !         !         !         !         !   5
 4 [illegible]    !  !1  2211  !  1 1 1 !         !         !         !         !         !         !         !         !   4
 3 [illegible]       !!211   ! 1!        !         !         !         !         !         !         !         !         !   3
 2 [illegible]   2   1 1 !     !        !         !         !         !         !         !         !         !         !   2
 1         !         !         !        !         !         !         !         !         !         !         !         !   1
  0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123
           10        20        30        40        50        60        70        80        90        100       110       120
                                                          -X- (     )
```

:	0.000E-01	6.400E+01	1.280E+02	1.920E+02	2.560E+02	3.200E+02	3.840E+02	4.480E+02
0.0000	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
64.0000	0.0000E-01	5.0000E+00	2.9000E+01	0.0000E-01	0.0000E-01	0.0000E-01	2.0000E+00	0.0000E-01
128.0000	0.0000E-01	7.0000E+00	3.7000E+01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
192.0000	0.0000E-01	6.0000E+00	4.7000E+01	0.0000E-01	0.0000E-01	0.0000E-01	1.0000E+00	0.0000E-01
256.0000	0.0000E-01	2.0000E+00	3.1000E+01	3.0000E+00	0.0000E-01	0.0000E-01	1.0000E+00	0.0000E-01
320.0000	0.0000E-01	3.0000E+00	3.3000E+01	2.0000E+00	0.0000E-01	0.0000E-01	0.0000E-01	1.0000E+00
384.0000	0.0000E-01	7.0000E+00	4.0000E+01	3.0000E+00	0.0000E-01	1.0000E+00	0.0000E-01	0.0000E-01
448.0000	0.0000E-01	4.0000E+00	3.2000E+01	6.0000E+00	1.0000E+00	0.0000E-01	0.0000E-01	0.0000E-01
512.0000	0.0000E-01	4.0000E+00	3.9000E+01	6.0000E+00	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
576.000	0.0000E-01	2.0000E+00	4.6000E+01	1.6000E+01	0.0000E-01	0.0000E-01	0.0000E-01	0.0000E-01
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0000E-01	0.0000E-01	0.0000E-01	1.0000E+00

X. Figure Captions

1. The overlay structure of WC.
2. The normal flow of WC
3. A single virtual display in VIRD.
4. Two virtual displays.
5. Set linearization on plot 1 and a free-form 2-DIM gate on plot 2.
6. Re-plot the gate set on figure 5.
7. Setup of 15 free form gates in VIRD. Gates can overlap and be set in any order on any 2-DIM parameter space.
8. Demonstration of the effect of linearization. The x-axis in plot 2 is the new-linearized x parameter and the y-axis is the same as in plot 1.
9. Plot of a group of 12 spectra in DSP1 and a window marked on the last spectrum (2) by (MW command).
10. Re-plot of spectrum 12 seen in figure 9 betwen the limits of the window marked on the previous plot (fig. 9.) using the command WD.
11. The same spectrum as 10 but changing the screen window by SW command. (See also PO command).
12. The same spectrum as 11 but in logarithmic scale (LL command code 2).
13. Spectrum plotted in dot mode (HD command). On the right side the error bars of the same spectrum are shown (ES command).
    A Gaussian fit to the spectrum within the marked window done in SPCOPR (OC) by CURFIT command and plotted by PF command in DSP1. This is the result of the calculation shown in Example 19b-19f.

14. Several spectra, one on top of the other, plotted in histogram mode with different line type (HD command).
15. The same spectrum plotted in linear and log scale (using LL command) and at two different screen-window settings (using the pre-set windows in command PO)

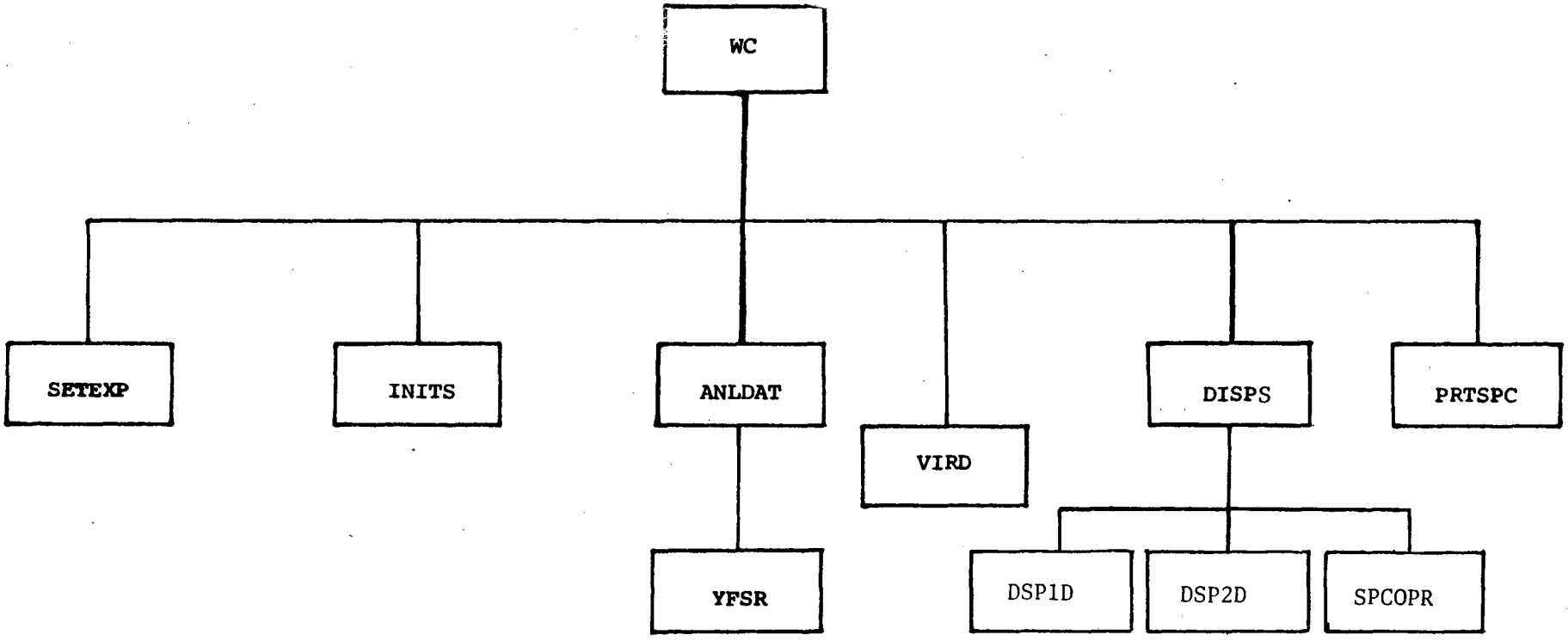

WC
SETEXP
INITS
ANLDAT
VIRD
DISPS
PRTSPC
YFSR
DSP1D
DSP2D
SPCOPR

Fig. 1

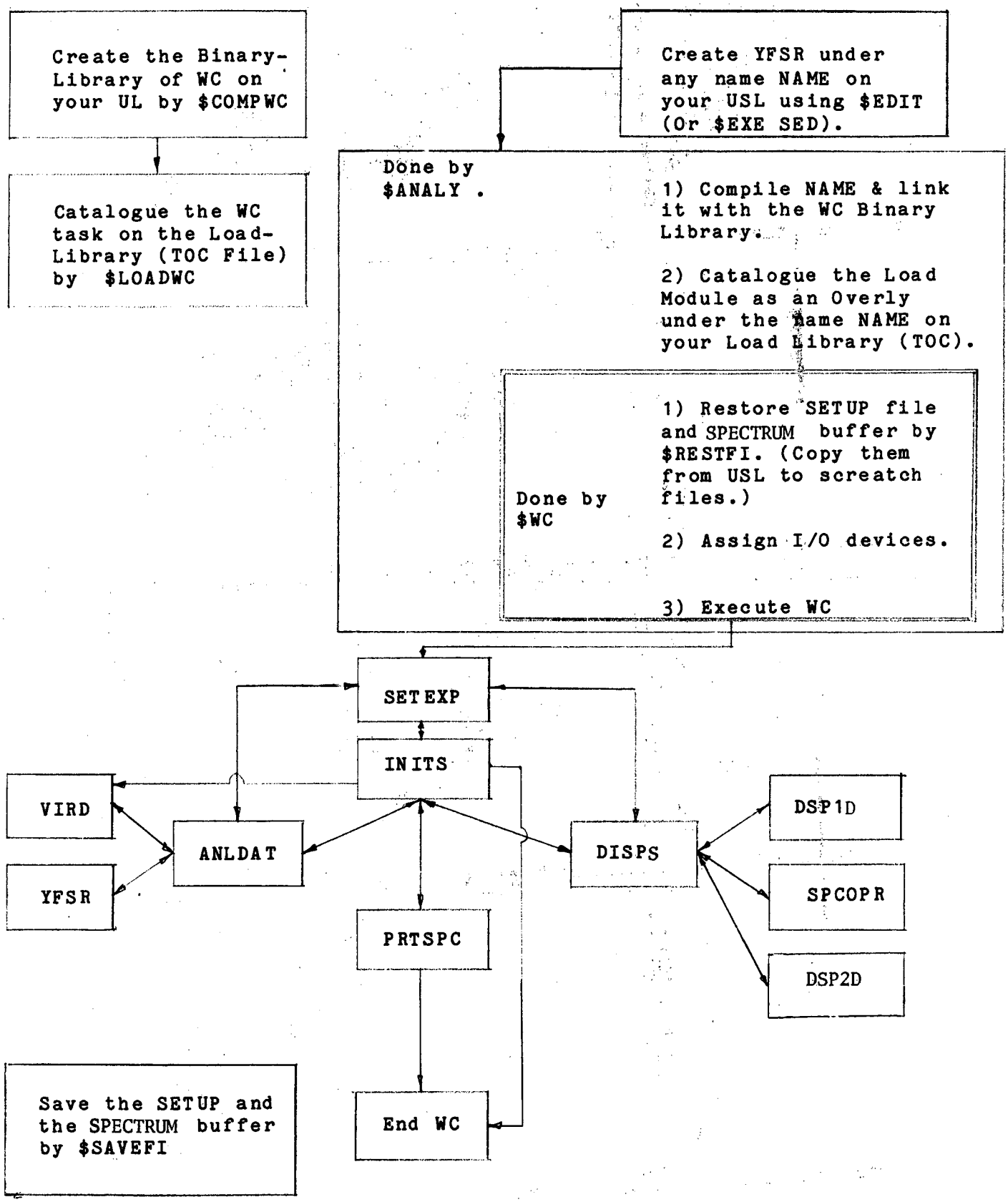

Create the Binary-Library of WC on your UL by $COMPWC
Catalogue the WC task on the Load-Library (TOC File) by $LOADWC
Create YFSR under any name NAME on your USL using $EDIT (Or $EXE SED).
Done by $ANALY .
1) Compile NAME & link it with the WC Binary Library.
2) Catalogue the Load Module as an Overly under the name NAME on your Load Library (TOC).
Done by $WC
1) Restore SETUP file and SPECTRUM buffer by $RESTFI. (Copy them from USL to screatch files.)
2) Assign I/O devices.
3) Execute WC
SETEXP
INITS
VIRD
YFSR
ANLDAT
DISPS
DSP1D
SPCOPR
DSP2D
PRTSPC
End WC
Save the SETUP and the SPECTRUM buffer by $SAVEFI

Fig. 2

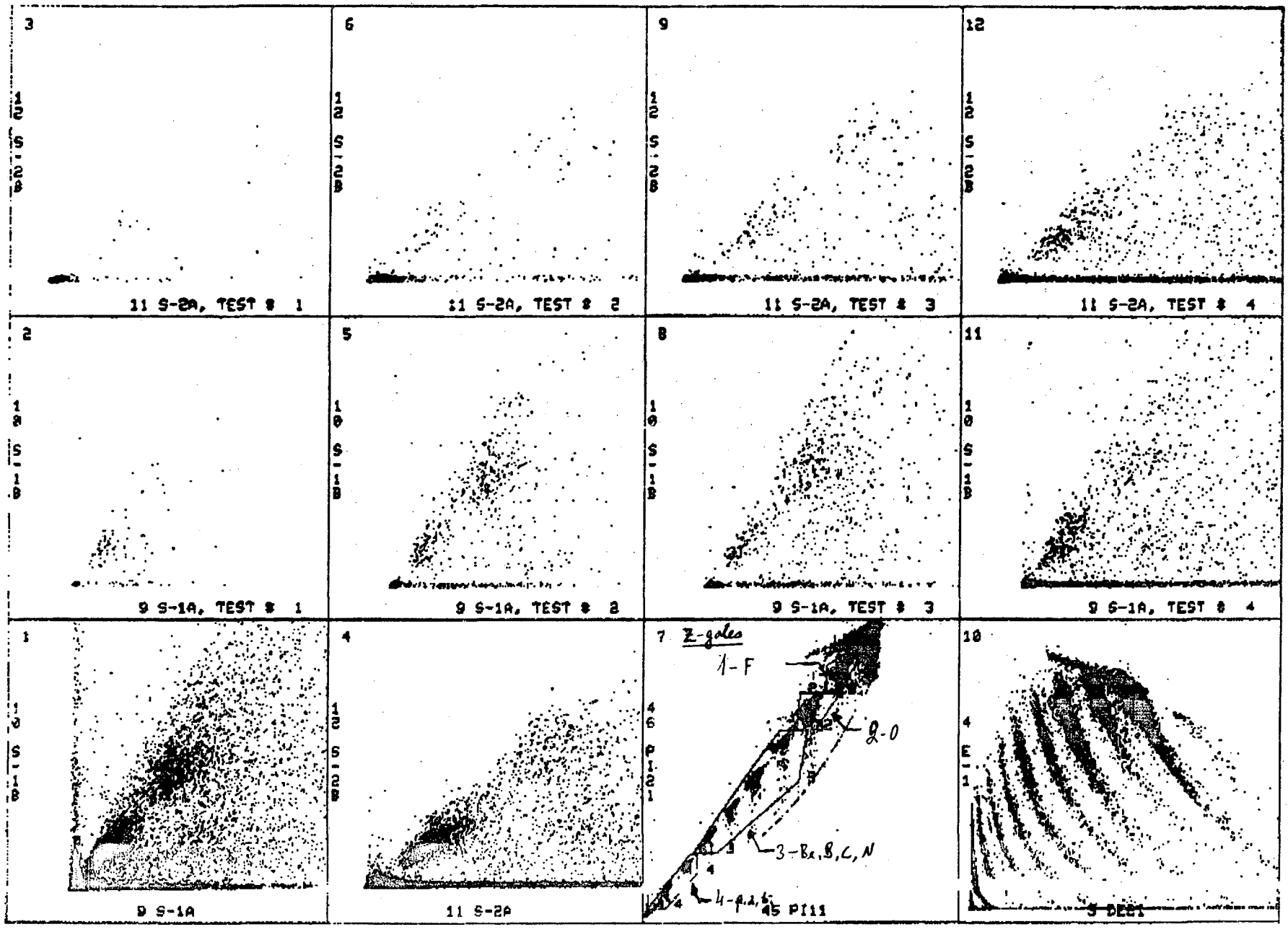

3
11 S-2A, TEST # 1
6
11 S-2A, TEST # 2
9
11 S-2A, TEST # 3
12
11 S-2A, TEST # 4
2
9 S-1A, TEST # 1
5
9 S-1A, TEST # 2
8
9 S-1A, TEST # 3
11
9 S-1A, TEST # 4
1
9 S-1A
4
11 S-2A
7
Z-gates
1-F
2-O
3-Be,B,C,N
4-p.s.
45 PIII
10
5 DEET

Fig. 3

```
- PLOT NUMBER 1
X-AXIS: PARAMETER  3  E2-1  XL-XU=  0.0000E-01  2.0480E+03
Y-AXIS: PARAMETER  4  E3-1  YL-YU=  0.0000E-01  2.0480E+03
- PLOT NUMBER 2
X-AXIS: PARAMETER  5  E1-2  XL-XU=  0.0000E-01  2.0480E+03
Y-AXIS: PARAMETER  6  E2-2  YL-YU=  0.0000E-01  2.0480E+03
```

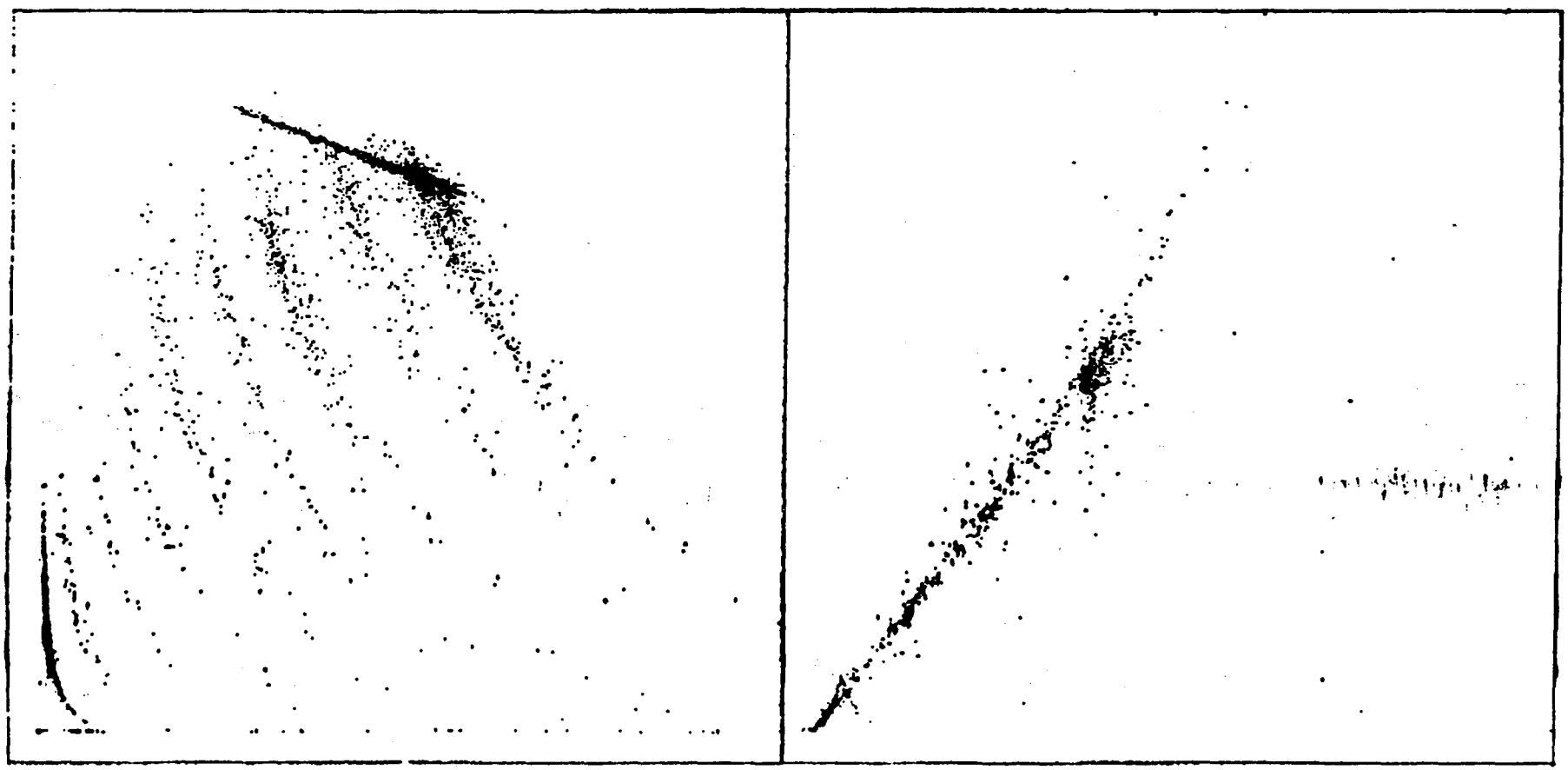


Fig. 4

```
- PLOT NUMBER 1
X-AXIS: PARAMETER  3  E2-1  XL-XU=  0.0000E-01  2.0480E+03
Y-AXIS: PARAMETER  4  E3-1  YL-YU=  0.0000E-01  2.0480E+03
- PLOT NUMBER 2
X-AXIS: PARAMETER  5  E1-2  XL-XU=  0.0000E-01  2.0480E+03
Y-AXIS: PARAMETER  6  E2-2  YL-YU=  0.0000E-01  2.0480E+03
```

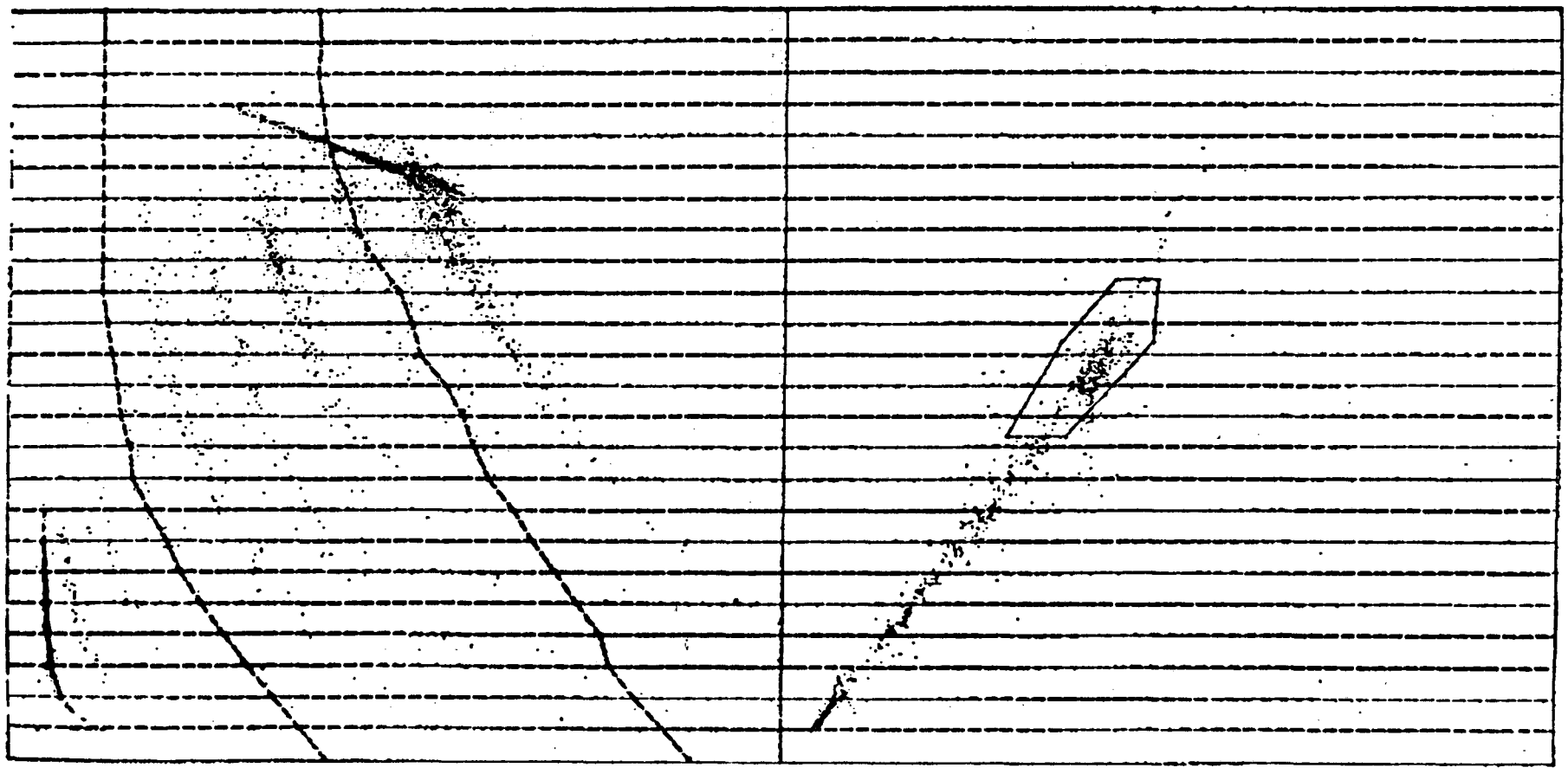
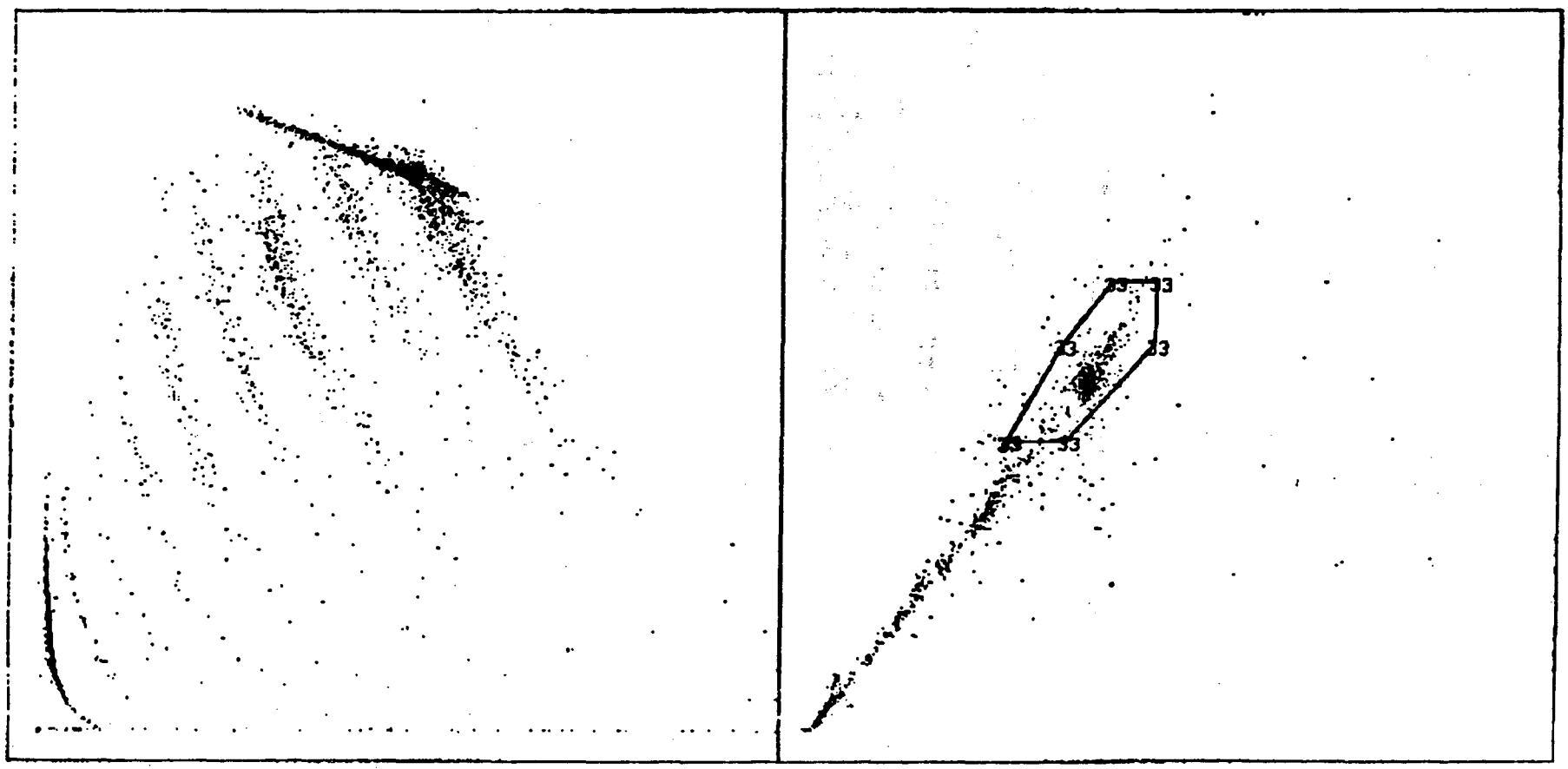


Fig. 5

- PLOT NUMBER 1
X-AXIS: PARAMETER 3 E2-1 XL-XU= 0.0000E-01 2.0480E+03
Y-AXIS: PARAMETER 4 E3-1 YL-YU= 0.0000E-01 2.0480E+03
- PLOT NUMBER 2
X-AXIS: PARAMETER 5 E1-2 XL-XU= 0.0000E-01 2.0480E+03
Y-AXIS: PARAMETER 6 E2-2 YL-YU= 0.0000E-01 2.0480E+03



*** SPECTRA BUFFER SIZE IS 16848 BYTES. 282 BYTES ARE NOT USED ***

Fig. 6

```
-ENTER PARAMETER 28                                                    169 BLOCKS SORTED.    0 ERROR BLOCKS
PGATE  PARAMETER 29
ABELEY PREDETERCONSTANT NUMBER :  CONSTANT IC(
TENTER GATE NUMBER30 IF 0 ALL GATES WILL BE PLOTTED
CONSTANT IC(10) =      0 NEW VALUE =
-ENTER PLOT NUMBER (IF 0 THE PRESENT ASSIGNMENT ARE ASSUMED)
2
```

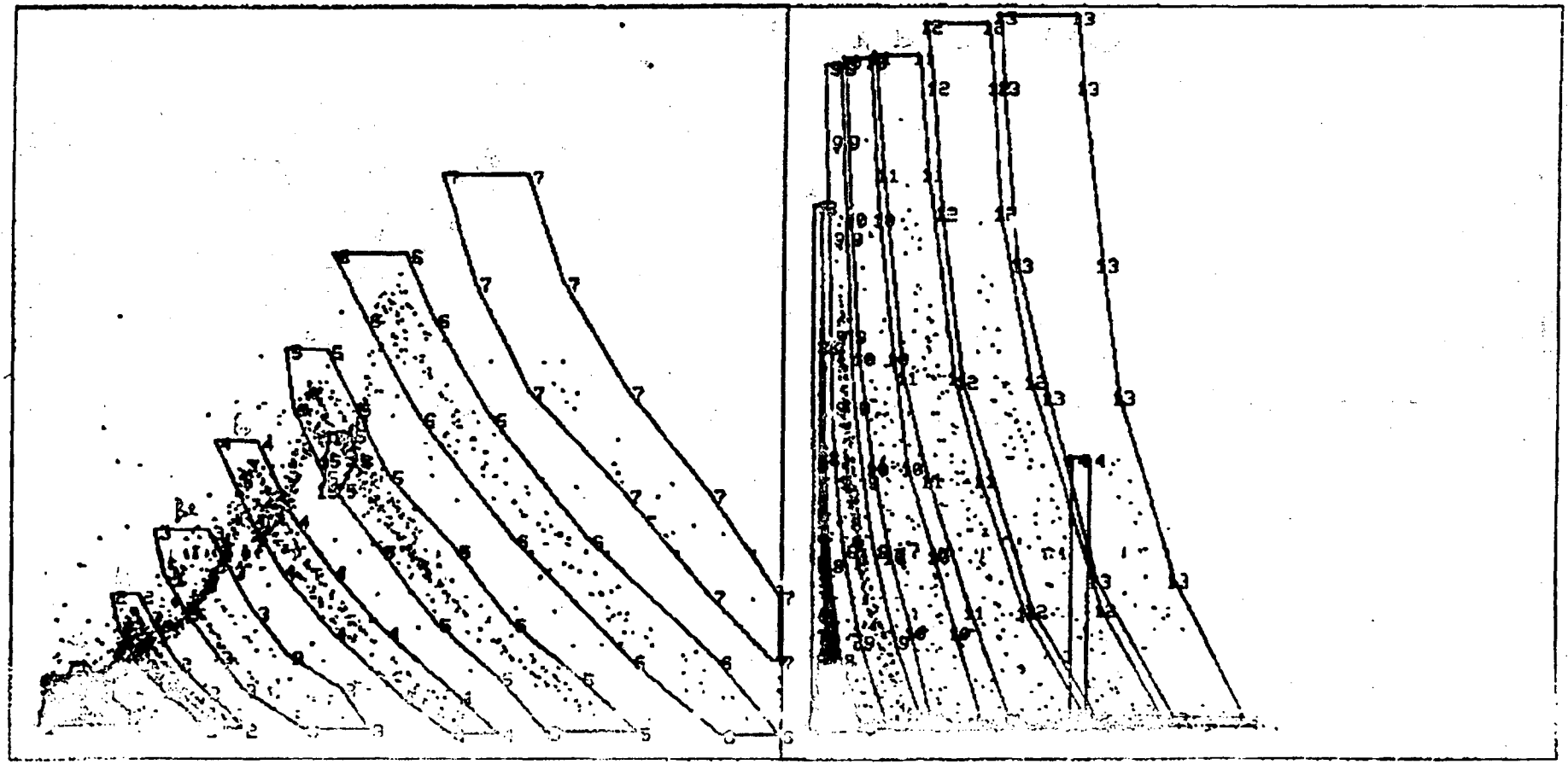


Fig. 7

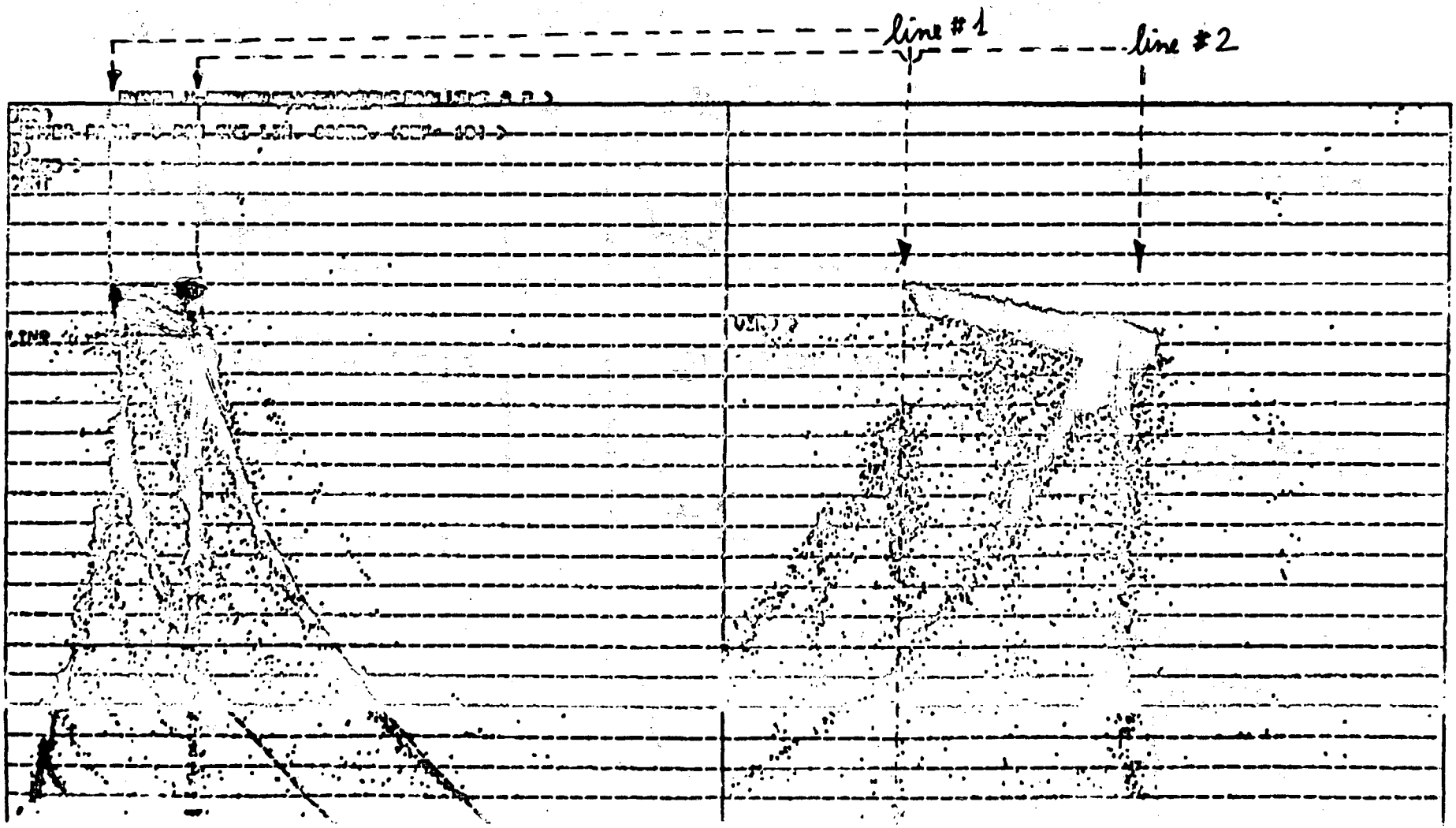
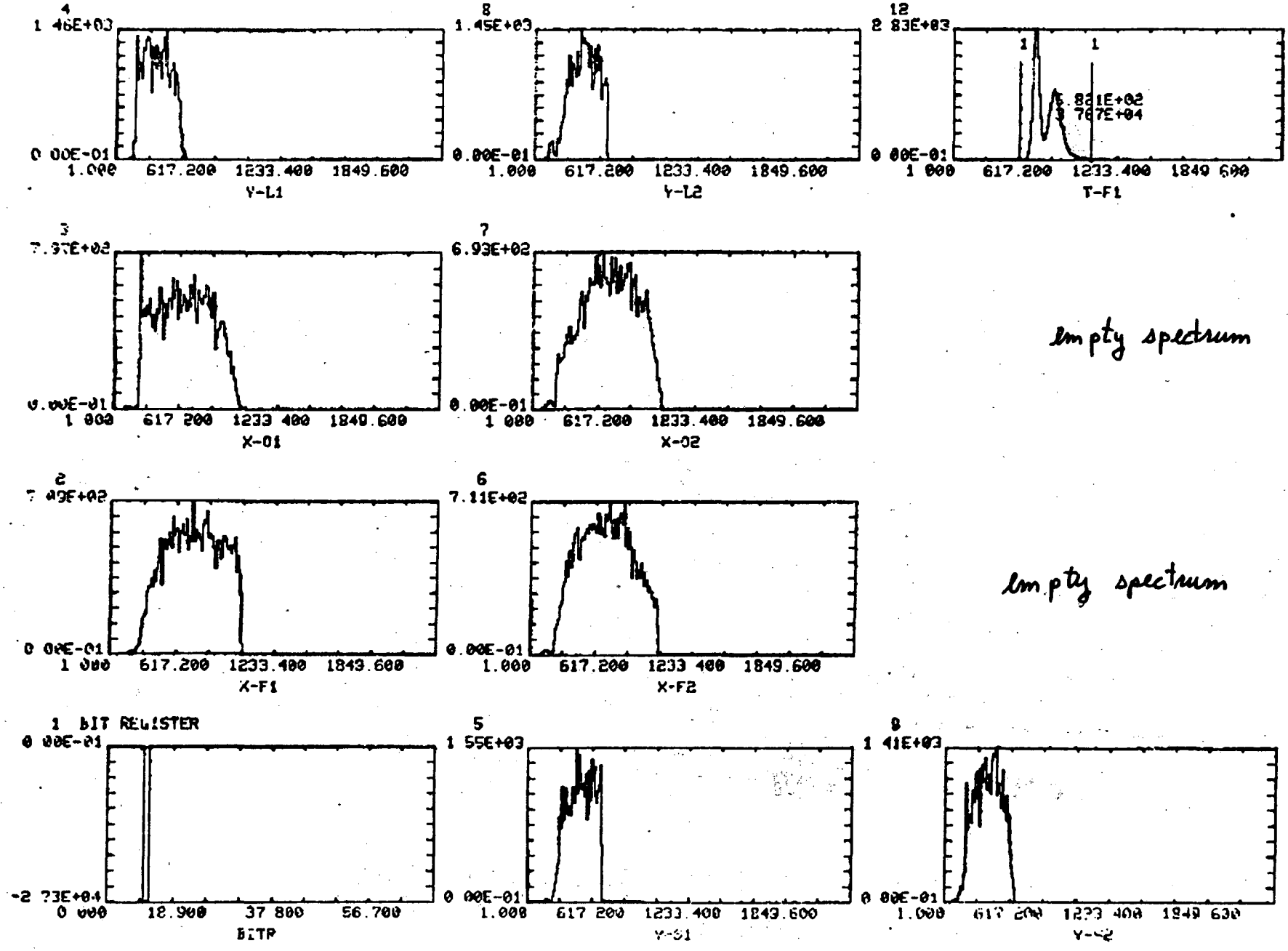

PLOT NUMBER 1
X-AXIS: PARAMETER 10 XL-XU= 0.0000E-01 2.0480E+03
Y-AXIS: PARAMETER 11 YL-YU= 0.0000E-01 2.0480E+03
- PLOT NUMBER 2
X-AXIS: PARAMETER 30 XL-XU= 0.0000E-01 2.0480E+03
Y-AXIS: PARAMETER 11 YL-YU= 0.0000E-01 2.0480E+03
line #1
line #2

Fig. 8



empty spectrum

empty spectrum

Fig. 9

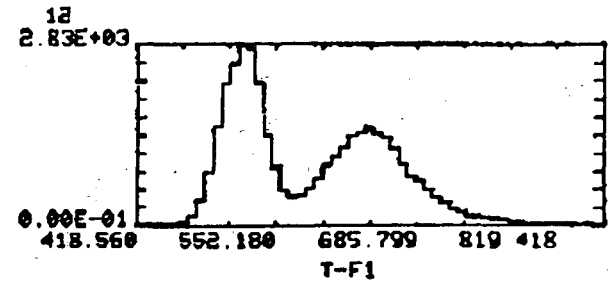


Fig. 10

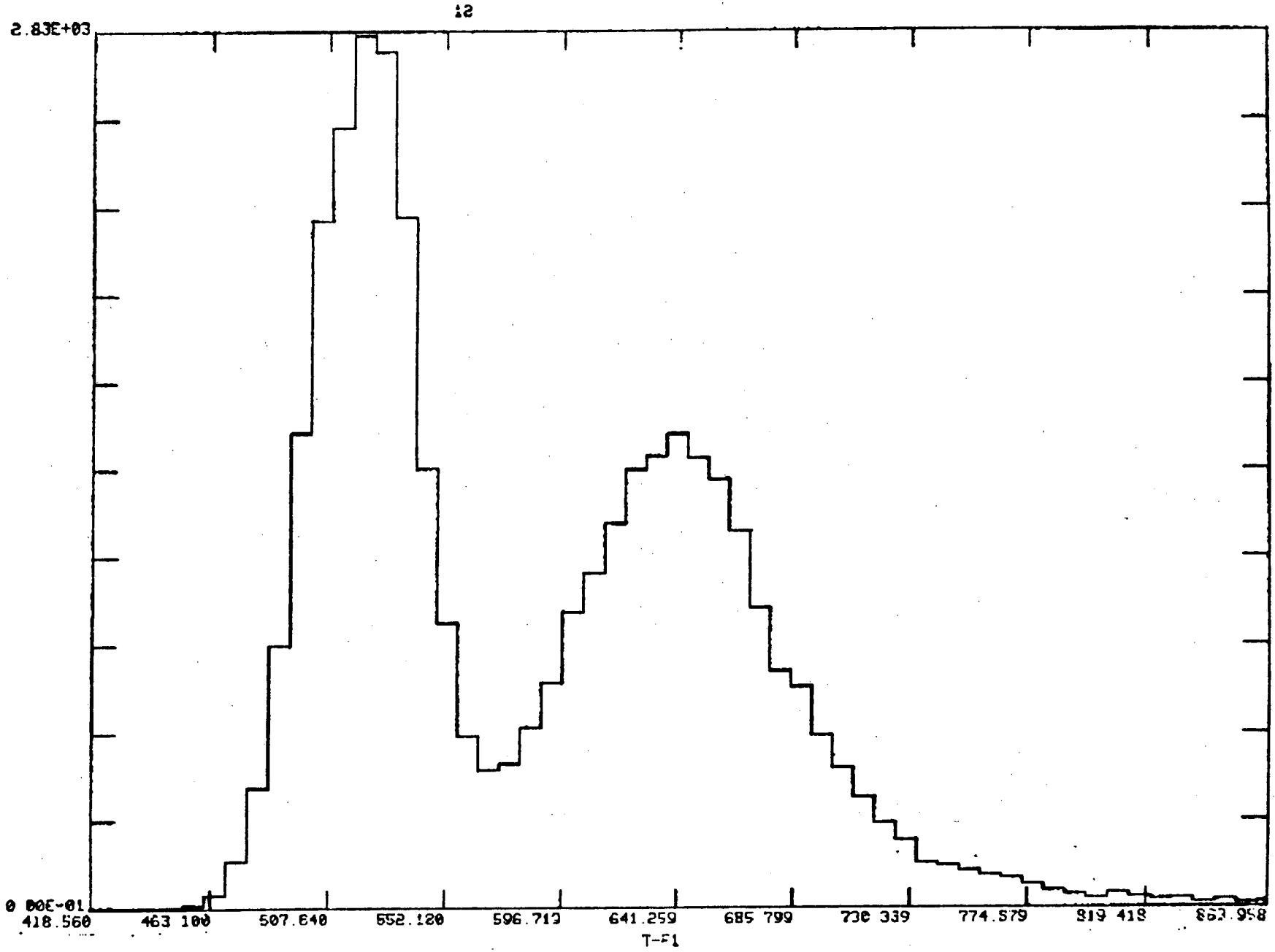

2.83E+03
0 00E-01
418.560
463 100
507.640
552.120
596.719
641.259
685 799
730 339
774.879
819 418
863.958
T-F1

Fig. 11

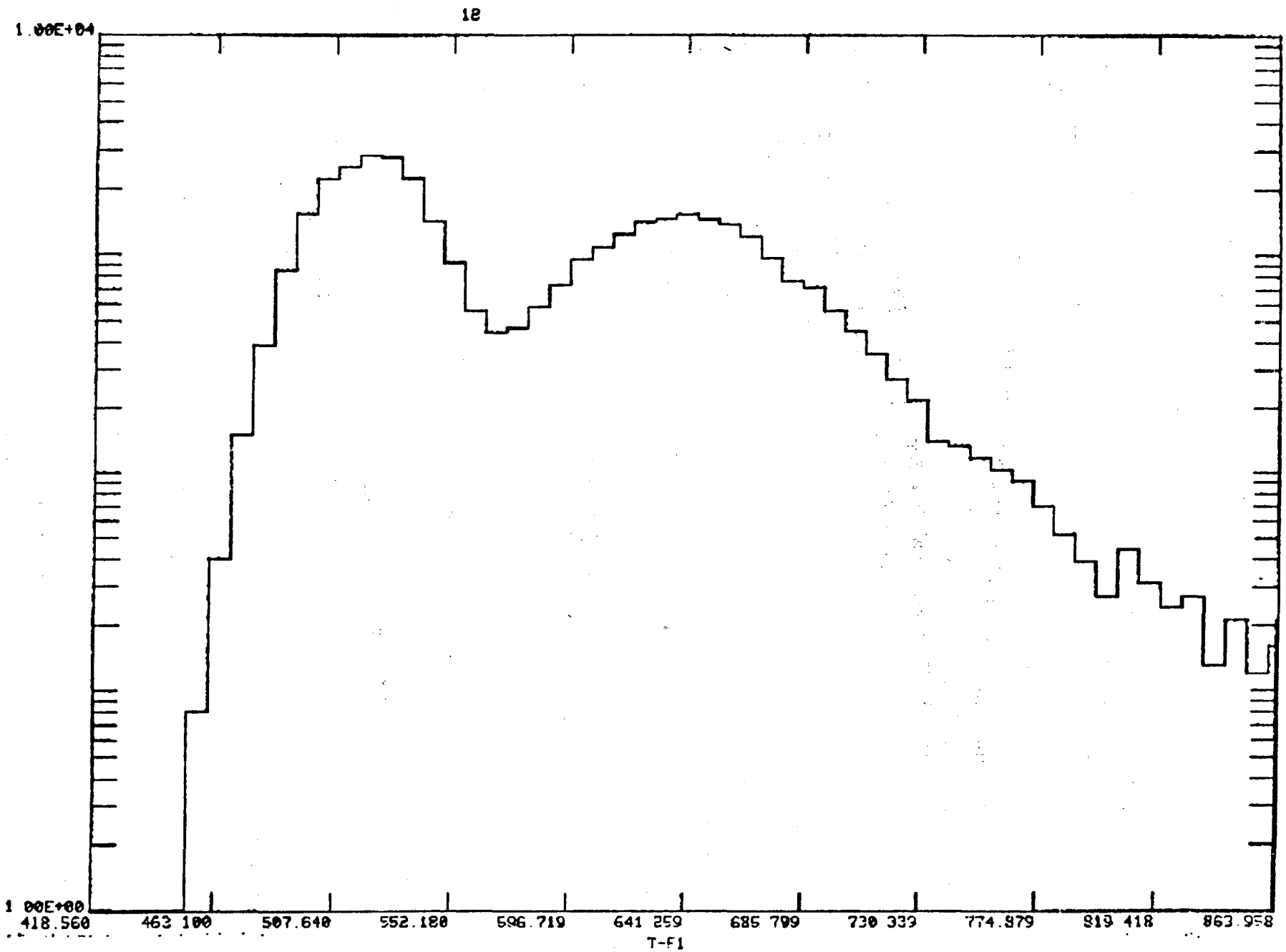

1.00E+04
1.00E+00
418.560
463.100
507.640
552.180
596.719
641.259
685.799
730.339
774.879
819.418
863.958
T-F1

Fig. 12

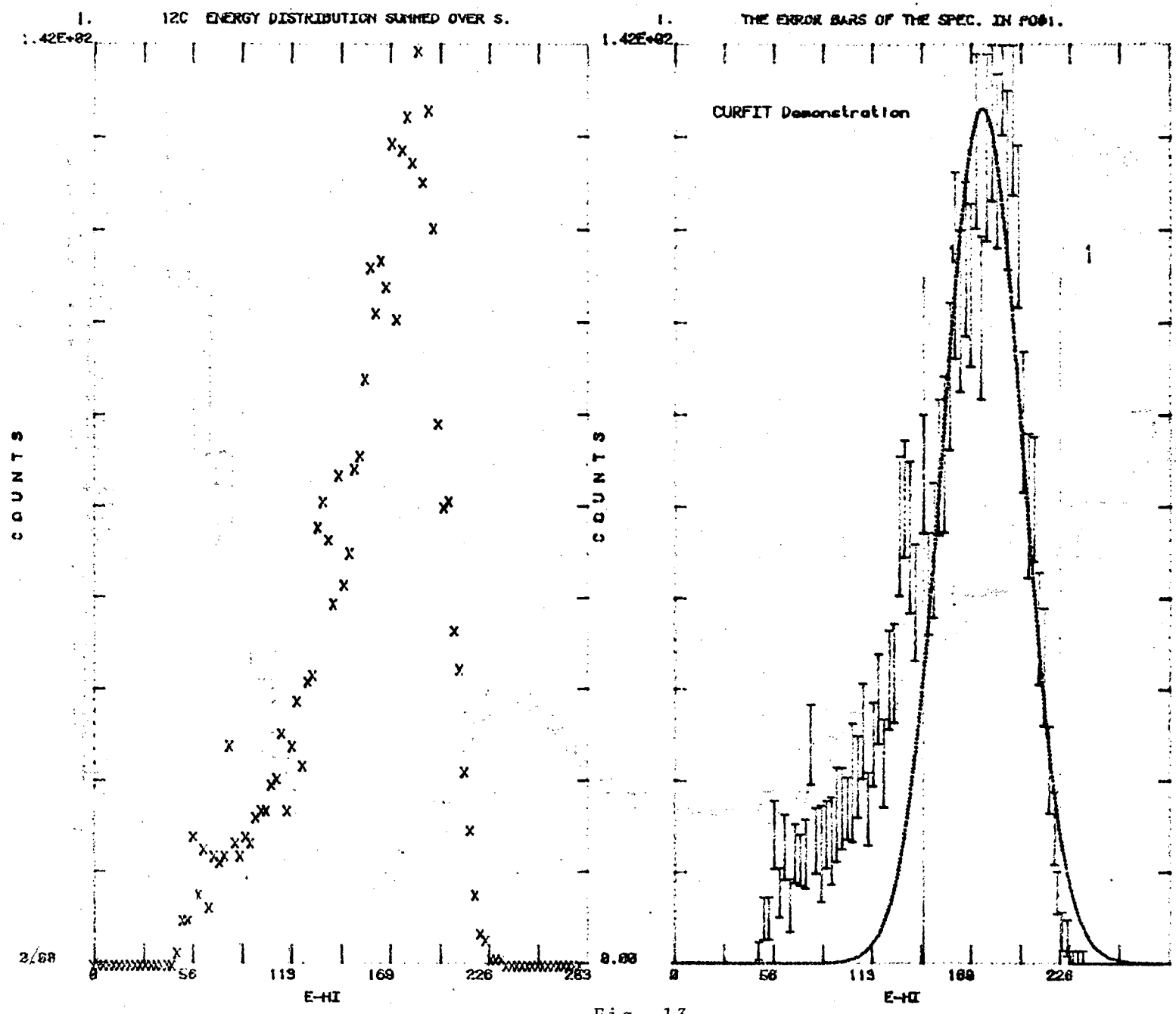

1. 12C ENERGY DISTRIBUTION SUMMED OVER S.
1.42E+02
COUNTS
2/68
0
56
113
169
226
283
E-HI
1. THE ERROR BARS OF THE SPEC. IN F081.
1.42E+02
CURFIT Demonstration
COUNTS
0.00
0
56
113
169
226
E-HI

Fig. 13

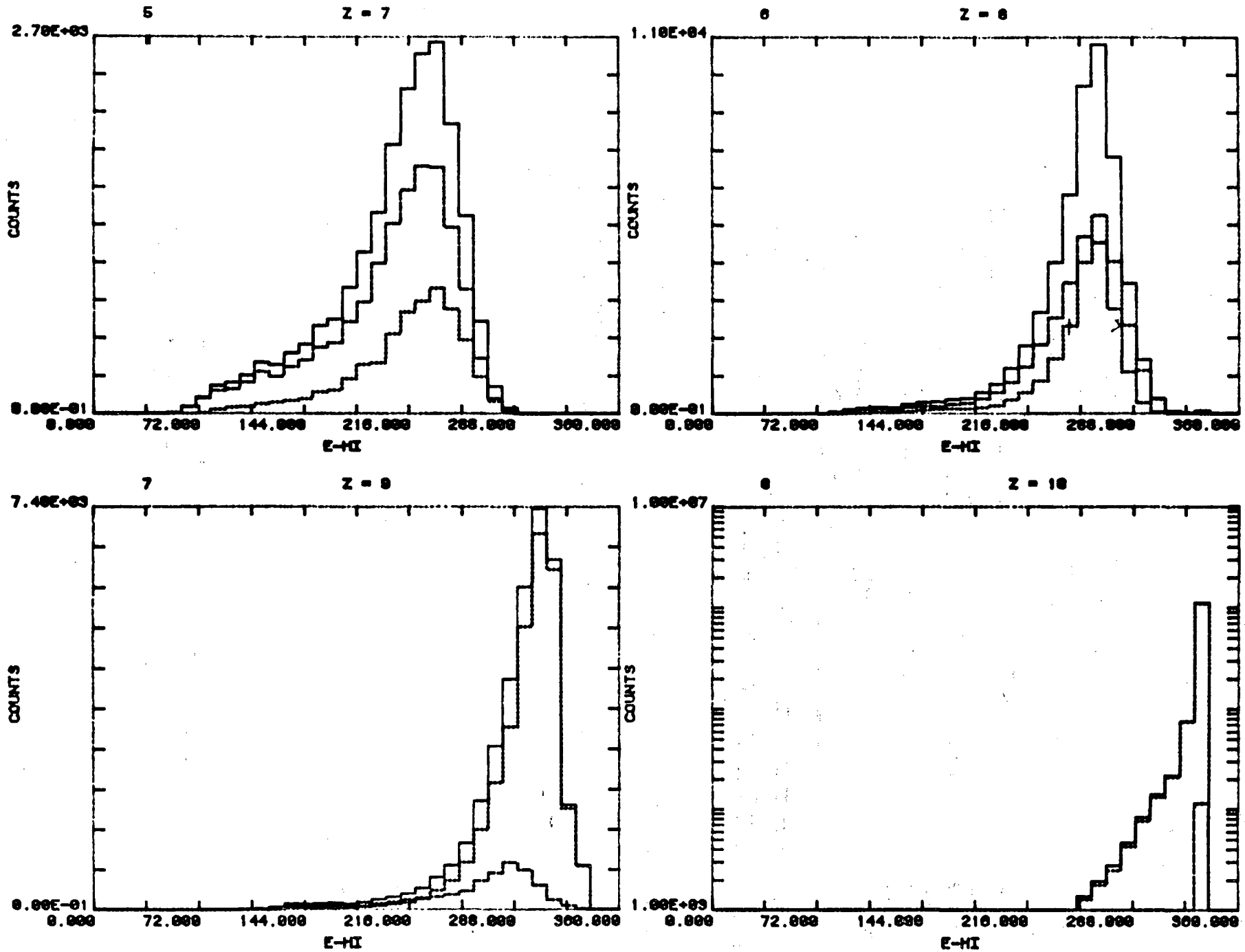

5
Z = 7
2.70E+03
0.00E-01
COUNTS
0.000
72.000
144.000
216.000
288.000
360.000
E-MI
6
Z = 8
1.10E+04
0.00E-01
COUNTS
0.000
72.000
144.000
216.000
288.000
360.000
E-MI
7
Z = 9
7.40E+03
0.00E-01
COUNTS
0.000
72.000
144.000
216.000
288.000
360.000
E-MI
8
Z = 10
1.00E+07
1.00E+03
COUNTS
0.000
72.000
144.000
216.000
288.000
360.000
E-MI

Fig. 14

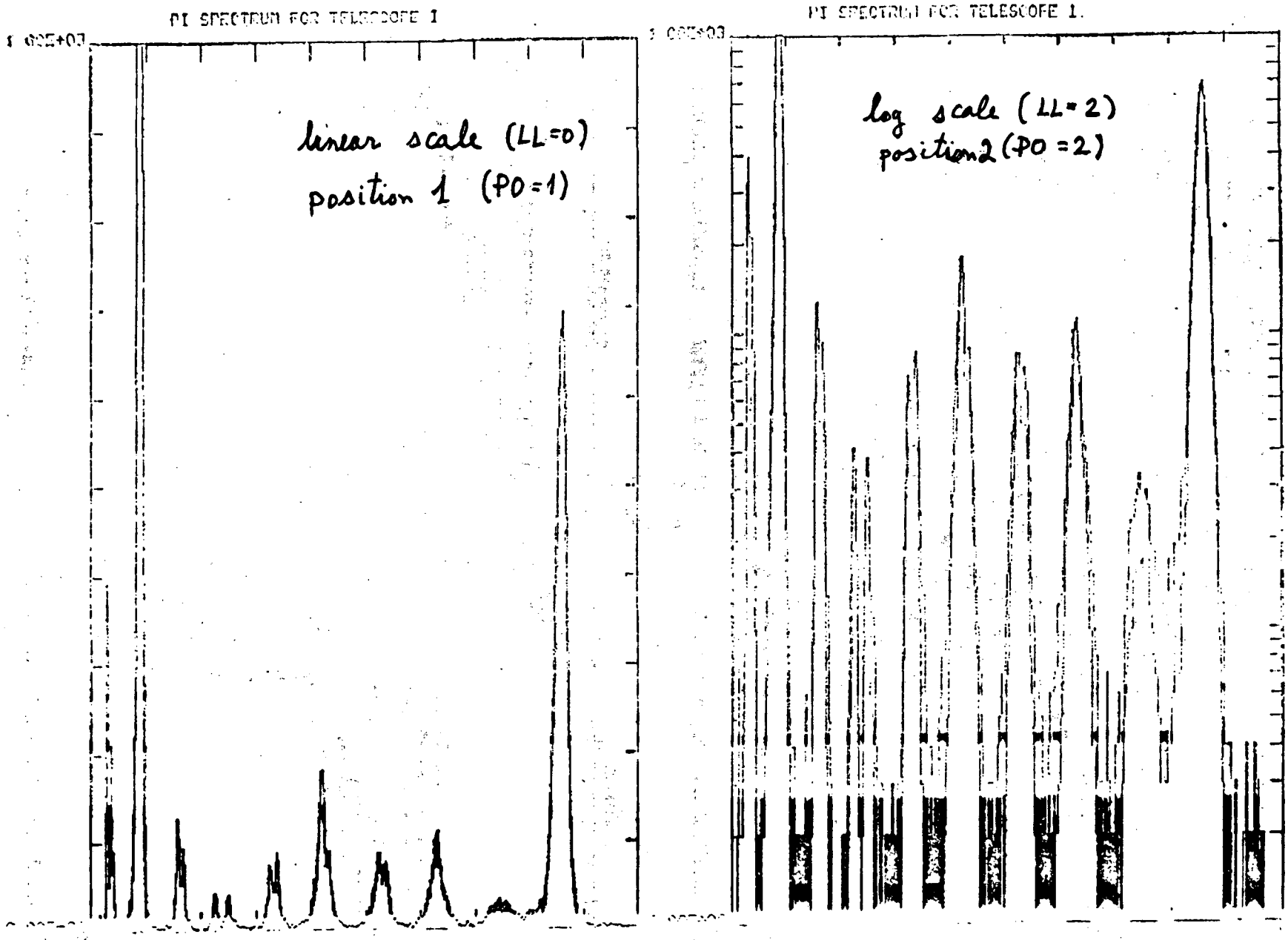

PI SPECTRUM FOR TELESCOPE 1
linear scale (LL=0)
position 1 (PO=1)
PI SPECTRUM FOR TELESCOPE 1.
log scale (LL=2)
position2 (PO=2)

Fig. 15

XI. Appendices

1. The default YRSR. 146

2.a. The default SETUP file. 150

b. An example of a more complex SETUP file. 151

3. List of subroutines. 154

4. List of procedures. 159

APPENDIX 1

```
C *** ****************************************************
C
C      S U B R O U T I N E      Y F S R
C
C      SKELETION SUBROUTINE OF YFSR.
C
C *** ****************************************************
      SUBROUTINE YFSR(NPASS,NSKIP,*)
C
      IMPLICIT INTEGER*2 (I)
C *** COMMON BLOCKS NEEDED FOR OVERLAYS.
C
      COMMON/IOCOM1/IO1,IO2,IO3,IO4,IO5,IO6,IO7,IO8,IO9
      COMMON/IOCOM2/IRUN,NRUNS,NDB,NDBMAX,NCOUNT,KCOUNT
      COMMON/IOCOM3/IWIN,IWOUT,NSKP(10),NACC(10),NRUNC
      COMMON/TEKCOM/ INTEKT
C     COMMON /SPCCOM/ ISPEC(8424)
      COMMON /DFSCOM/ KAVLSP,KSTADD,LEFTSP
      COMMON /DUMCOM/ DUMMY(4212)
      COMMON /RANDOM/ NSEED
      COMMON /PARCOM/ NPARM,LPARM(100),TXTPR(100)
      COMMON /CNSCOM/ NCNST,CNST(100)
      COMMON /CONCOM/ NCOND,COND(100,2),ICONDP(100)
      COMMON/D1SCO1/ND1SP,ID1SP(100,4),D1SPC(100,3),TXT1D(100,10)
      COMMON/D1SCO2/KD1SP(100),LD1SP(100)
      COMMON/D2SCO1/ND2SP,ID2SP(100,5),D2SPC(100,6),TXT2D(100,10)
      COMMON/D2SCO2/KD2SP(100),LXD2SP(100),LYD2SP(100)
      COMMON /WCNCOM/ NWCND,WCND(10,6)
      COMMON /GT2COM/ NGATS2,MAXGAT,MAXPOP
      COMMON/GBOCO1/IBN(50),IPX(50),IPY(50),BY(1000),BXL(1000),BXU(1000)
      COMMON/GBCOM2/INWC(50)
      COMMON /TSTCOM/ NTST,ITEST(100),NTEST(100),TSTXT(10,100)
      COMMON/STSCO1/NSTEST,IGRP1(4),IGRP2(4),ISTSN(4)
      COMMON/STSCO2/IOPR1(4),IOPR2(4),IOPR3(4)
      COMMON/STSCO3/IG1D(4,5),IG1G(4,5),IG1P(4,5)
      COMMON/STSCO4/IG2D(4,5),IG2G(4,5),IG2P(4,5)
      COMMON /VARCO1/NBITPI,IBITPI(100),ICOORI(100),ISTCOI(100)
      COMMON /VARCO2/NBITPO,IBITPO(100),ICOORO(100),ISTCOO(100)
      COMMON /LINCOM/ NLINR,LINPR(16,3),XLINDT(16,26,2),YLINL(16,2)
      COMMON/VIRCO1/NVRSN,IVIR,NVRSET,NVRPLT
      COMMON/VIRCO2/IVRS(2),IVRTST(12,2),IVRPRX(12,2),IVRPRY(12,2)
      COMMON/VIRCO3/VRXL(12,2),VRXU(12,2),VRYL(12,2),VRYU(12,2)
```

```
      COMMON/VIRCO4/XVRL(12),YVRL(12),VRCX(12),VRCY(12)
      COMMON/VIRCO5/IVRXL(12),IVRXU(12),IVRYL(12),IVRYU(12)
      COMMON/VIRCO6/IVRPX(12),IVRPY(12),IVRT(12)
      COMMON/ANSCOM/IANS1,IANS2,IANS3,IANS4,IANS5,IANS6
      COMMON/EVNCOM/IPARI,IVI(100),IPARO,IVO(100),EVENT(100),ICMC
      COMMON /PSPCOM/ NPSPR,IFPSPR,IPSPR(100,4),CPSPR(100,2)
      COMMON /QLOCOM/ LDNAME(3),LDLIBN(2)
C
      DATA INO,IYES/2HN ,2HY /
C ***
      DATA I0/0/
C
C      BEGINNING
C      =========
      IF(NPASS.EQ.1) GO TO 9000
      IF(NPASS.EQ.-1) GO TO 9900
C
C     WRITE INPUT EVENT IF DEBUG OPTION IS SET
C
      IF(NDB.EQ.0)GO TO 8001
      NCT=NCT+1
      IF(NCT.GT.NDBMAX) NCT=0
      IF((NDB.EQ.1).AND.(NCT.NE.NDBMAX))GO TO 8001
      IF(IPARI.LE.I0) GOTO 8001
      WRITE(IO2,8101) (IVI(I),I=1,IPARI)
      WRITE(IO2,8100) (IVI(I),I=1,IPARI)
8101   FORMAT(10(/,' HEX ',10(Z4,1X)))
8100   FORMAT(10(/,1X,10I7))
8001    CONTINUE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C               INPUT YOUR FAVORITE ANALYSIS CODE HERE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     DONT SAVE THE EVENT (THE DEFAULT OPTION)
C
C     NSKIP=-1
C
C     SAVE THE EVENT
C
      NSKIP=0
C *** CALCULATE PSEUDO PARAMETERS.
```

```
 85         CALL PSPR
 86   C*** DO ALL LINEARIZATIONS DEFINED.
 87         CALL LINR
 88   C *** AUTO CHACK OF SIMPLE TESTS. ISTEST IS THE PRODUCT OF ALL THE TESTS
 89   C     DEFINED AS A SIMPLE TEST WITHIN STST COMMAND OF "INITS".
 90         IF(ISTEST(0).EQ.I0) NSKIP=-1
 91   C *** AUTO SPECTRA INCREMENT.
 92         CALL INCSPC
 93   C     REJECT EVENT THAT DOES NOT PASS THE SIMPLE TEST.
 94   C     IF(NSKIP.LT.0) GOTO 8000
 95   C
 96   8000  CONTINUE
 97   CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
 98   C     DEFAULT O/P BUFFER CONTAINS EVENT(1) TO EVENT(IPARO)
 99   C     COPIED TO THE INTEGER*2 IVO BUFFER.
100   C
101         IF(IPARO.LE.0) GOTO 8002
102         IF(NSKIP.LT.0) GOTO 8002
103         DO 8003 I=1,IPARO
104   8003   IVO(I) = INT(EVENT(I))
105   C
106   CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
107   C     WRITE DEBUG OF OUTPUT IF DEBUG OPTION IS SET
108   C
109         IF(NDB.EQ.0)GO TO  8002
110         IF((NDB.EQ.1).AND.(NCT.NE.NDBMAX))GO TO  8002
111         WRITE(IO2,8101) (IVO(I),I=1,IPARO)
112         WRITE(IO2,8100) (IVO(I),I=1,IPARO)
113    8002    CONTINUE
114   C *** IN CASE OF DUMMY INPUT (SIMULATION CODE) STOP THE EVENT-BY-EVENT
115   C     CALCULATION BY "RETURN 1".
116         IF(IANS5.EQ.IYES) RETURN 1
117   C
118   C *** END OF EVENT-BY-EVENT PROCEDURE. ***
119         RETURN
120   C
121   CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
122   C     THIS SECTION IS EXECUTED ON THE 1ST CALL TO YFSR.
123   CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
124   C
125    9000 NDBMAX=75
126   C     NDBMAX = MAXIMUM NUMBER OF DEBUG EVENTS
```

```
127          NCT=NDBMAX-1
128    C *** IN CASE OF DUMMY INPUT (SIMULATION CODE) STOP THE EVENT-BY-EVENT
129    C     CALCULATION BY "RETURN 1".
130          IF(IANS5.EQ.IYES) RETURN 1
131    C
132          RETURN
133    C
134    CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
135    C     THIS SECTION IS EXECUTED ON THE LAST CALL TO YFSR.
136    CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
137    C
138    9900  CONTINUE
139    C
140          WRITE(IO2,9901)
141    9901  FORMAT(1X,'LAST CALL TO "YFSR"')
142    C INPUT CODE TO DUMP SPECTRA OR RESULTS OF CALCULATION
143    C
144    C *** PRINT RESULTS.
145          RETURN
146          END
147    C ***********************************************************************
TOTAL RECORDS WRITTEN =   148/     18
```

```
 1  PARM          100
 2  PARM LONG     100
 3    1 4096   2 4096   3 4096   4 4096   5 4096   6 4096   7 4096   8 4096
 4    9 4096  10 4096  11 4096  12 4096  13 4096  14 4096  15 4096  16 4096
 5   17 4096  18 4096  19 4096  20 4096  21 4096  22 4096  23 4096  24 4096
 6   25 4096  26 4096  27 4096  28 4096  29 4096  30 4096  31 4096  32 4096
 7   33 4096  34 4096  35 4096  36 4096  37 4096  38 4096  39 4096  40 4096
 8   41 4096  42 4096  43 4096  44 4096  45 4096  46 4096  47 4096  48 4096
 9   49 4096  50 4096  51 4096  52 4096  53 4096  54 4096  55 4096  56 4096
10   57 4096  58 4096  59 4096  60 4096  61 4096  62 4096  63 4096  64 4096
11   65 4096  66 4096  67 4096  68 4096  69 4096  70 4096  71 4096  72 4096
12   73 4096  74 4096  75 4096  76 4096  77 4096  78 4096  79 4096  80 4096
13   81 4096  82 4096  83 4096  84 4096  85 4096  86 4096  87 4096  88 4096
14   89 4096  90 4096  91 4096  92 4096  93 4096  94 4096  95 4096  96 4096
15   97 4096  98 4096  99 4096100 4096
16  PARM MEMO     100
17        1         2         3         4         5         6         7
18        8         9        10        11        12        13        14
19       15        16        17        18        19        20        21
20       22        23        24        25        26        27        28
21       29        30        31        32        33        34        35
22       36        37        38        39        40        41        42
23       43        44        45        46        47        48        49
24       50        51        52        53        54        55        56
25       57        58        59        60        61        62        63
26       64        65        66        67        68        69        70
27       71        72        73        74        75        76        77
28       78        79        80        81        82        83        84
29       85        86        87        88        89        90        91
30       92        93        94        95        96        97        98
31       99       100
32  PARM END
33   CNST          0
34   COND          0
35   1DSP          0
36   2DSP          0
37   WCND      0
38   GAT2      0
39   LINR          0
40   PSPR          0
41   TSTX          0
42   ANLD
```

*E.0+1 MODCOMP SOURCE EDITOR DATE 02/08/84 22:45:54

APPENDIX 2b

```
 1  EXAMPLE SETUP #1 JAN 31 1984
 2  PARM          100
 3  PARM CHAN     100
 4    1 4096  2 4096  3 4096  4 4096  5 4096  6 4096  7 4096  8 4096
 5    9 4096 10 4096 11 4096 12 4096 13 4096 14 4096 15 4096 16 4096
 6   17 4096 18 4096 19 4096 20 4096 21 4096 22 4096 23 4096 24 4096
 7   25 4096 26 4096 27 4096 28 4096 29 4096 30 4096 31 4096 32 4096
 8   33 4096 34 4096 35 4096 36 4096 37 4096 38 4096 39 4096 40 4096
 9   41 4096 42 4096 43 4096 44 4096 45 4096 46 4096 47 4096 48 4096
10   49 4096 50 4096 51 4096 52 4096 53 4096 54 4096 55 4096 56 4096
11   57 4096 58 4096 59 4096 60 4096 61 4096 62 4096 63 4096 64 4096
12   65 4096 66 4096 67 4096 68 4096 69 4096 70 4096 71 4096 72 4096
13   73 4096 74 4096 75 4096 76 4096 77 4096 78 4096 79 4096 80 4096
14   81 4096 82 4096 83 4096 84 4096 85 4096 86 4096 87 4096 88 4096
15   89 4096 90 4096 91 4096 92 4096 93 4096 94 4096 95 4096 96 4096
16   97 4096 98 4096 99 4096100 4096
17  PARM MEMO     100
18       1         2         3         4         5         6 DE      7 E
19       8         9        10        11        12        13        14
20      15        16        17        18        19        20        21
21      22        23        24        25        26        27        28
22      29        30        31        32        33        34        35
23      36        37        38        39        40        41        42
24      43        44        45        46        47        48        49
25      50        51        52        53        54        55        56
26      57        58        59        60        61        62        63
27      64        65        66 L-DE   67        68        69        70
28      71        72        73        74        75        76        77
29      78        79        80        81        82        83        84
30      85        86        87        88        89        90        91
31      92        93        94        95        96        97        98
32      99       100
33  PARM END
34   CNST          2
35     2.3450E+00   5.7890E+00
36   COND          2
37     3.500E+00  7.987E+00   5.777E+05  8.808E+05
38      2    4
39   1DSP          4
40       2   0.000E-01 1.001E+02 2.500E+00   1       0    -1
41     1: EXAMPLE
42       2   0.000E-01 1.001E+02 2.500E+00   2       0    -1
```

```
43      2: EXAMPLE BIN=2
44        6   1.000E+00 2.048E+03 1.600E+01   2       0    -1
45      3: NON LINEAR X
46       66   0.000E-01 1.000E+02 1.000E+00   2       0    -1
47      4: LINEARIZED X
48    2DSP          0
49    WCND          0
50    GAT2          2
51    GAT2#  1      6    6    7   -1
52     6.6412E+02  1.5485E+03  1.7167E+03  9.1105E+02  1.3363E+03  1.5605E+03
53     1.3968E+03  1.0761E+03  1.3083E+03  1.7170E+03  8.9587E+02  1.1401E+03
54     1.8445E+03  5.2351E+02  1.0921E+03  2.0019E+03  5.2752E+02  8.3582E+02
55    GAT2#  2      2   66    7   -1
56     4.3348E+02  4.0665E+01  4.9267E+01  1.7767E+03  4.0274E+01  5.0244E+01
57    LINR          1
58    LINR#    1    6    7   66   0.000E-01   2.048E+03
59    9.119E+02 8.118E+02 6.997E+02 6.196E+02 5.515E+02 5.115E+02 4.635E+02 4.234
60    3.954E+02 3.794E+02 3.553E+02 3.273E+02 2.993E+02 2.913E+02 2.793E+02 2.633
61    2.472E+02 2.432E+02 2.272E+02 2.232E+02 2.152E+02 2.152E+02 2.152E+02 2.152
62    2.112E+02 2.000E+01
63    1.689E+03 1.621E+03 1.508E+03 1.448E+03 1.360E+03 1.268E+03 1.200E+03 1.136
64    1.084E+03 1.028E+03 9.559E+02 9.159E+02 8.598E+02 8.278E+02 7.878E+02 7.597
65    7.157E+02 6.837E+02 6.636E+02 6.396E+02 6.196E+02 6.116E+02 5.996E+02 5.996
66    5.996E+02 6.050E+01
67    PSPR          0
68    TSTX          3
69        1 ELASTIC OR CARBON
70        2 ELSE
71        3 THE SAME AS 2
72    SIMP TEST     3
73    SIMP TEST#    1    2    0    1    0    0    1
74        2  1  0     2  2  0
75    SIMP TEST#    2    2    0    1    0    1    2
76        2  1  0     2  2  0
77    SIMP TEST#    3    1    0    1    0    1    3
78        0  1  0
79    VIRD          2
80    SETV NUM      6
81        2    3    0
82      0.000E-01   2.048E+03   0.000E-01   2.048E+03
83        3    4    0
84      0.000E-01   2.048E+03   0.000E-01   2.048E+03
```

```
   85        5    6    0
   86      0.000E-01   2.048E+03   0.000E-01   2.048E+03
   87        6    7    0
   88      0.000E-01   2.048E+03   0.000E-01   2.048E+03
   89        8    9    0
   90      0.000E-01   2.048E+03   0.000E-01   2.048E+03
   91        8    9    0
   92      0.000E-01   4.096E+03   0.000E-01   4.096E+03
   93   SETV NUM       5
   94        6    7    0
   95      0.000E-01   2.048E+03   0.000E-01   2.048E+03
   96      66    7    0
   97      0.000E-01   1.000E+02   0.000E-01   2.048E+03
   98        6    7    1
   99      0.000E-01   2.048E+03   0.000E-01   2.048E+03
  100        6    7    2
  101      0.000E-01   2.048E+03   0.000E-01   2.048E+03
  102        6    7    3
  103      0.000E-01   2.048E+03   0.000E-01   2.048E+03
  104   ANLD
TOTAL RECORDS WRITTEN =   105/     15
ASS LO LO

$$
```

APPENDIX 3

A list of subroutines, functions and entries is given here.

The list is segmented according to the overlay under which the subroutines are commonly called.

```
 198        SUBROUTINE WC


 284        SUBROUTINE SETEXP(IGOTO,IPASS)


1727        SUBROUTINE INITS(IGOTO,IPASS)

4377        SUBROUTINE SAVESP
               ENTRY DUMPSP
               ENTRY RESTSP
1976        SUBROUTINE ZSSPC
2022        SUBROUTINE PARMS(*,*)
               ENTRY RPARMS
               ENTRY WPARMS
2205        SUBROUTINE CNSTS(*,*)
               ENTRY RCNSTS
               ENTRY WCNSTS
2292        SUBROUTINE CONDS(*,*)
               ENTRY RCONDS
               ENTRY WCONDS
2396        SUBROUTINE D1SPS(*,*)
               ENTRY RD1SPS
               ENTRY WD1SPS
2593        SUBROUTINE D2SPS(*,*)
               ENTRY RD2SPS
               ENTRY WD2SPS
2799        SUBROUTINE VFORM(*,*)
               ENTRY RVFORM
               ENTRY WVFORM
5033        SUBROUTINE SSTEST
2912        SUBROUTINE TSTTXT(*,*)
               ENTRY WTSTXT
               ENTRY RTSTXT
2981        SUBROUTINE LINRS(*,*)
               ENTRY RLINRS
               ENTRY WLINRS
4747        SUBROUTINE WCNDS(*,*)
               ENTRY RWCNDS
               ENTRY WWCNDS
```

```
4857        SUBROUTINE GAT2S(*,*)
               ENTRY RGAT2S
               ENTRY WGAT2S
5246        SUBROUTINE PSPRS
               ENTRY RPSPRS
               ENTRY WPSPRS


 790        SUBROUTINE ANLDAT(IGOTO,IPASS)

7852        SUBROUTINE CHREAD(NCH,ICH,IOK)
               ENTRY  TRAPON(ICH)
               ENTRY  TRAPOF


7957         SUBROUTINE QLOAD(NAME, LFN, ADDR, STAT)
  69        SUBROUTINE YFSR(NPASS,NSKIP,*)

3896        FUNCTION ICOND(IP,IC)
4681        FUNCTION IGAT2(MPX,MPY,NGN,MWC)
5153        FUNCTION ISTEST(MDUMMY)
4641        FUNCTION GAUSS(S,W)
4114        SUBROUTINE LINR
4123        SUBROUTINE LINR1(L)
5400        SUBROUTINE PSPR
5417        SUBROUTINE PSPR1(N)
3351        SUBROUTINE INCSPC
3370        SUBROUTINE INC1DS(NS,DEL)
3406        SUBROUTINE INC2DS(NS,DEL)
3060        SUBROUTINE SPECTRA(VECTOR)
               ENTRY ZSPECS
               ENTRY GET1DS(NS,AREA,VECTOR,NDIM)
               ENTRY DPS1DS(NS,VECTOR)
               ENTRY GET2DS(NS,AREA,VECTOR,NDIMX,NDIMY)
               ENTRY DPS2DS(NS,VECTOR)
               ENTRY GET2DP(NS,NPRJ,XL,XU,YL,YU,AREA,VECTOR,MDIM)
3795        SUBROUTINE NRMSPC(ND,IS1,ADD,DIV,IS)
4507        SUBROUTINE DIVSPC(ND,IS1,IS2,IS)
4553        SUBROUTINE MULSPC(ND,IS1,IS2,IS)
4597        SUBROUTINE ADDSPC(ND,IS1,IS2,IS)
3826        SUBROUTINE SQRTSP(ND,IS1,IS)
```

```
3861        SUBROUTINE PWR2SP(ND,IS1,IS)
4402        SUBROUTINE DEF1DS(IP,XL,XU,DX,ITYP,N)
4451        SUBROUTINE DEF2DS(IPX,XL,XU,DX,IPY,YL,YU,DY,ITYP,N)
6744        SUBROUTINE COMBIN(NS1,NS2,NCOMP,KBINS,DX)
4342        SUBROUTINE AVRSPC(IS,PL,PU,AVR,SIGMA,SUM)
4315        SUBROUTINE AVA(Y,X,IXL,IXU,XAV,SIGMA,SUM,NDIM)
4179        SUBROUTINE SMOOTH(Y1,Y2,N,NDIM)
4246        SUBROUTINE DSMOTH(Y1,Y2,XBIN,N,NDIM)
3755        SUBROUTINE VMAX(X,NXL,NXU,XMAX,NXMAX,NDIM)
3775        SUBROUTINE VMIN(X,NXL,NXU,XMIN,NXMIN,NDIM)


1156        SUBROUTINE VIRD(IGOTO,IPASS)

4977        SUBROUTINE D2SORT
               ENTRY D2SOR1(NVD)
4954        SUBROUTINE MARK(IX,IY)
5022        SUBROUTINE DRWBOX(I)



3456        SUBROUTINE PRTSPC(IGOTO,IPASS)

3675        SUBROUTINE PMAT(DUMMY,TXT,XL,XU,DX,NCNTL,NDIM,NSUM)
3712        SUBROUTINE PARR(DUMMY,TXTX,XL,XU,DX,LX,TXTY,YL,YU,DY,LY,NCNTL
3916        SUBROUTINE PLOT1D(X,YFX,JI,KI,JJ,IYMAX,XL,XS,YL,YS,LOGS,NDIM)
4030        SUBROUTINE PLOT2D(DUMMY,TXTX,XL,XU,DX,LX,TXTY,YL,YU,DY,LY,ICN
                                                                        IS)


5459        SUBROUTINE DISPS(IGOTO,IPASS)

5603        SUBROUTINE VR2AB(X,Y,IX,IY)
5629        SUBROUTINE AB2VR(IX,X)
5646        SUBROUTINE SMOVEA(X,Y)
5656        SUBROUTINE SDRAWA(X,Y,ITYP)
5672        SUBROUTINE BELL
5685        SUBROUTINE BY2WR(TXT,IWT,NCHR)
7912        SUBROUTINE VT100
               ENTRY VTTTEK
               ENTRY TEKTVT
               ENTRY CLRTEK
```

```
5707        SUBROUTINE DSP1D(IOANS,*)

6662        SUBROUTINE MRKCND(*)
6714        SUBROUTINE STOCND(*)

6785        SUBROUTINE SPCOPR(IOANS,*)

7489        SUBROUTINE CURFIT(X,Y,WEIGHT,YFIT,NPTS,ND,
7623        FUNCTION FCHISQ(Y,WEIGHT,NPTS,NFREE,YFIT,NCH1,NCH2)
7641        SUBROUTINE MATINV(ARRAY,NORDER,DET)
7706        SUBROUTINE FDERIV(X,N,A,DELTAA,NTERMS,DERIV,NFIT)
7726        FUNCTION FCURVE(X,N,A,NFIT)
7733        SUBROUTINE DFCURV(X,N,A,DELTAA,NTERMS,DERIV,NFIT)
7742        FUNCTIØN FGAUSS(X,N,A,NFIT)
7760        SUBROUTINE DGAUSS(X,N,A,DELTAA,NTERMS,DERIV,NFIT)
7782        FUNCTION FMAXWL(X,N,A,NFIT)
7804        SUBROUTINE DMAXWL(X,N,A,DELTAA,NTERMS,DERIV,NFIT)
7834        SUBROUTINE POSDEF(V1,V2,ND,VMIN,XMIN)
```

APPENDIX 4

List of Procedures

1. $COMPWC - Compile WC and put the binary files on a binary library.
2. $LOADWC - Create the WC task on a load-module file.
3. $ANALY - Create the OVERLAY of YFSR.
4. $WC - Execute WC.
5. $RESTFI - Copy a file from USL to a scratch file.
6. $SAVEFI - Copy a file from a scratch file to the USL.
7. $SCRFI - Create a new scratch file.
8. $DELFUL - Delete an object file from a binary library.
9. $ADDTUL - Add an object file to a binary library.
10. $DELADD - Replace certain object file on a binary library by its new verison.
11. $LNAMUL - List names of all object files on a binary library.
12. $EDIT - Edit a file under SED, Remove the old version and catalogue the new one.
13. $EXCISE - Copy a segment of a file on USL to a different file on USL.
14. $INSERT - Copy a segment of a file on USL before a given line of another file on USL.
15. $CLG - Compile, link and execute a program by passing your UL
16. $CHKMT + The program CERT1FY - Check old magnetic tapes against parity errors before recycling.

```
*E.0+1  MODCOMP SOURCE EDITOR      DATE  02/08/84  21:49:28        PAGE   1


  1  *_  PRODEFAULT COMPWC USL,WCB,FR5,WCNAME
  2  *   RE-CREATE THE "WC" BINARY LIBRARY.
  3  *  P1 - THE LIBRARY ON WHICH THE SOURCE OF "WC" IS. DEFAULT-USL.
  4  *  P2 - WHERE TO CATALOGE THE BINARY LIBRARY. DEFAULTE-WCB. IF
  5  *       REMOVEABLE DISC IS USED SET P2 TO RDO.
  6  *  P3 - THE COMPILER TO USE. DEFAULTE-FORTRAN 5 (FR5).
  7  *_ SHLOMO WALD, LBL, MARCH 1983.
  8  $PRODEFAULT COMPWC USL,WCB,FR5,WC
  9  $ASS BI %2
 10  $REW BI
 11  $WEOF BI
 12  $REW BI
 13  $JOB RECREATE NEW BINARY LIBRARY.
 14  $ASS USL %1
 15  $IFN %3=FR5,P=$AVR CI 2
 16  $COMPL5 %4,NOLO,NOMAP
 17  $AVR CI 1
 18  $COMPIL %4,NOLO,NOMAP
 19  $EXE LIB
 20  ASS BI SCA BO %2
 21  REW BI BO
 22  LNAM
 23  COPY
 24  EXI
 25  $LNAMUL %2
 26  $JOB END COMPWC
 27  $END
TOTAL RECORDS WRITTEN =    28/      4
POS $LOADWC
LIST
```

```
 1  *_  PRODEFAULT LOADWC SU,WCB,COMPL5,NOLIST,WCNAME
 2  *     SAVE "WC"'S LOAD MODULE.
 3  *  P1 - ON WHICH FILE TO CATALOGE THE LOAD MODULE (IN TOC). DEFAULT-SU.
 4  *  P2 - THE FILE ON WHICH THE "WC" BINARY LIB. IS. DEFAULT-WCB.
 5  *       IF ON A REMOVEABLE DISC- SET P2 TO RDO  .
 6  *  P3 - WHICH COMPILER TO USE. DEFAULT-FORTRAN 5.
 7  *  P4 - IF PRESENT THE LOADER MAP IS LISTED ON THE VERSATEC.
 8  *  P5 - CHANGE THE NAME OF WC'S LOAD MODULE (DEFAULT-WC).
 9  *_ SHLOMO WALD, LBL, JULY 1983.
10  $PRODEFAULT LOADWC SU,WCB,,,
11  $IFP %4,P=$ASS LO VP
12  $EXE LIB
13  ASS BI %2
14  NOF
15  LNAM
16  GET MAIN
17  ASS BI BI
18  EXI
19  $WEOF BO
20  $REW BI BO
21  $ASS BI BO
22  $EXE M4EDIT
23  ASS GL ULB  PL %2 ML UL
24  LIB ML PL GL LB
25  INCLUDE MAIN,CHREAD
26  SEGMENT  1,SETEXP,
27  SEGMENT           2,ADVFIL,2,ADVREC,2,ASSMGT,2,REWMGT,2,GOTEOT,
28  SEGMENT           2,WRTEOT,
29  SEGMENT  1,INITS,
30  SEGMENT          2,PARMS,2,CNSTS,2,CONDS,2,D1SPS,2,D2SPS,2,FORDEF,
31  SEGMENT          2,RPARMS,2,RCNSTS,2,RCONDS,2,RD1SPS,2,RD2SPS,2,FORDEF,
32  SEGMENT          2,WPARMS,2,WCNSTS,2,WCONDS,2,WD1SPS,2,WD2SPS,2,FORDEF,
33  SEGMENT          2,LINRS,2,GAT2S,2,WCNDS,2,TSTTXT,
34  SEGMENT          2,RLINRS,2,RGAT2S,2,RWCNDS,2,RTSTXT,
35  SEGMENT          2,WLINRS,2,WGAT2S,2,WWCNDS,2,WTSTXT,
36  SEGMENT  1,ANLDAT,
37  SEGMENT  1,VIRD,
38  SEGMENT  1,DISPS,
39  SEGMENT     2,SPCOPR,
40  SEGMENT              3,CURFIT,3,LEGFIT,
41  SEGMENT     2,DSP1D,
42  SEGMENT     2,DSP2D,
```

```
   43  SEGMENT                3,PLOT2D
   44  SEGMENT  1,PRTSPC,
   45  SEGMENT        2,PLOT1D,
   46  SEGMENT        2,PLOT2D
   47  EDIT
   48  WEOF BO
   49  REW BO BI SC %1
   50  EXI
   51  $JOB SAVE THE LOAD MODULE
   52  $IFP %5,P=$AVR CI 8
   53  $EXE TOC
   54  FIL %1
   55  PEC CLM,PIP
   56  TASK WC,250,250
   57  ALL ALL
   58  CAT
   59  EXI
   60  $AVR CI 7
   61  $EXE TOC
   62  FIL %1
   63  PEC CLM,PIP
   64  TASK %5,250,250
   65  ALL ALL
   66  CAT
   67  EXI
   68  $JOB >>> LOADWC ENDED. GOOD BYE <<<
   69  $END
TOTAL RECORDS WRITTEN =    70/      8
POS $ANALY
LIST
```

```
 1  *_   PRODEFAULT ANALY YFSR,SETUP,SPECT,TM,SU,UL,FR5,NOLIST,WCB,WCNAME
 2  *** "QUIT" LOAD-MODULE OF "YFSR" IS CREATED.
 3  *** EXTENDED-MEMORY VERSION !.
 4  *   COMPILE "YFSR" AND LINK IT TO ALL THE NEEDED LIB. SAVE THE LOAD MODULE
 5  *  FOR FUTURE USE, AND EXECUTE "WC".
 6  *_ P1 - THE NAME, ON THE USL,  OF "YFSR" OR ANY SUBROUTIENS THE USER
 7  *_       WISHS TO LINK WITH "WC".
 8  *_       DEFAULT-THE DEFAULT YFSR, AND THE TASK IS CALLED "WC00".
 9  *_ P2 - THE NAME OF THE "SETUP" FILE ON THE USL. DEFAULT-SETUP.
10  *  P3 - THE NAME OF THE "SPECTRA-BUFFER" ON THE USL. NO DEFAULT VALUE.
11  *  P4 - ON WHICH FILE TO CATALOGE THE LOAD MODULE (IN TOC). DEFAULT-TM.
12  *  P5 - THE FILE ON WHICH THE LOAD MODULE OF WC RESIDES. DEFAULT-SU.
13  *  P6 - THE FILE ON WHICH THE USER BINARY LIB. IS (E.G.-UL).
14  *       IF NO USER BINARY LIBRARY EXISTS- SET P6 TO NOUL.
15  *  P7 - WHICH COMPILER TO USE. DEFAULT-FORTRAN 5 (FR5).
16  *  P8 - IF = LO THE LOADER MAP IS LISTED ON THE VERSATEC. DEFAULT-NOLO
17  *  P9 - THE FILE ON WHICH THE "WC" BINARY LIB. IS. DEFAULT-WCB.
18  *       IF ON A REMOVEABLE DISC- SET P9 TO RDO  .
19  *C P10- ASS IO3 -DEFAULT: TY. WITH SINGLE GRAPHIC TERMINAL SET TO NST.
20  *  P10- THE MANE OF WC-TASK ON THE LOAD-FILE GIVEN IN P5.
21  *_ SHLOMO WALD, LBL, MARCH 1983.
22  *** NOTE *** ALL COMMON BLOCKS MUST APPEAR IN YFSR.
23  *** NOTE *** COMPIL WITH $4E OPTION.
24  $PRODEFAULT ANALY WC00,SETUP,,TM,SU,UL,FR5,NOLO,WCB,WC
25  $NOTE COMPIL "YFSR" AND LINK IT TO ALL THE NEEDED LIB.
26  $IF %8=LO,P=$ASS LO VP
27  $IF %7=FR4,P=$AVR CI 2
28  $COMPL5 %1,NONE,$4E
29  $AVR CI 1
30  $COMPIL %1,NONE,$4E
31  $EXE M4EDIT
32  $IF %6=NOUL,P=$AVR CI 3
33  ASS GL ULB  WL %9 PL %6
34  LIB PL WL GL LB
35  AVR CI 2
36  ASS GL ULB  WL %9
37  LIB WL GL LB
38  ATT D2SCO1,ABS,IMAP,NOR,IPR
39  ATT D2SCO2,ABS,IMAP,NOR,IPR
40  ATT GBCOM2,ABS,IMAP,NOR,IPR
41  ATT ANSCOM,ABS,IMAP,NOR,IPR
42  ATT GT2COM,ABS,IMAP,NOR,IPR
```

```
43  ATT TSTCOM,ABS,IMAP,NOR,IPR
44  ATT RANDOM,ABS,IMAP,NOR,IPR
45  ATT PSPCOM,ABS,IMAP,NOR,IPR
46  ATT TEKCOM,ABS,IMAP,NOR,IPR
47  ATT DUMCOM,ABS,IMAP,NOR,IPR
48  ATT VARCO1,ABS,IMAP,NOR,IPR
49  ATT VARCO2,ABS,IMAP,NOR,IPR
50  ATT VIRCO1,ABS,IMAP,NOR,IPR
51  ATT VIRCO2,ABS,IMAP,NOR,IPR
52  ATT WCNCOM,ABS,IMAP,NOR,IPR
53  ATT VIRCO3,ABS,IMAP,NOR,IPR
54  ATT VIRCO4,ABS,IMAP,NOR,IPR
55  ATT CNSCOM,ABS,IMAP,NOR,IPR
56  ATT VIRCO5,ABS,IMAP,NOR,IPR
57  ATT VIRCO6,ABS,IMAP,NOR,IPR
58  ATT GBOCO1,ABS,IMAP,NOR,IPR
59  ATT CONCOM,ABS,IMAP,NOR,IPR
60  ATT QLOCOM,ABS,IMAP,NOR,IPR
61  ATT STSCO1,ABS,IMAP,NOR,IPR
62  ATT DFSCOM,ABS,IMAP,NOR,IPR
63  ATT STSCO2,ABS,IMAP,NOR,IPR
64  ATT PARCOM,ABS,IMAP,NOR,IPR
65  ATT STSCO3,ABS,IMAP,NOR,IPR
66  ATT IOCOM1,ABS,IMAP,NOR,IPR
67  ATT D1SCO1,ABS,IMAP,NOR,IPR
68  ATT STSCO4,ABS,IMAP,NOR,IPR
69  ATT IOCOM2,ABS,IMAP,NOR,IPR
70  ATT D1SCO2,ABS,IMAP,NOR,IPR
71  ATT EVNCOM,ABS,IMAP,NOR,IPR
72  ATT IOCOM3,ABS,IMAP,NOR,IPR
73  ATT LINCOM,ABS,IMAP,NOR,IPR
74  COM D2SCO1,,#0000
75  COM D2SCO2,,#0E76
76  COM GBCOM2,,#10CE
77  COM ANSCOM,,#1100
78  COM GT2COM,,#1106
79  COM TSTCOM,,#110C
80  COM RANDOM,,#1A0A
81  COM PSPCOM,,#1A0C
82  COM TEKCOM,,#1D2F
83  COM DUMCOM,,#1D30
84  COM VARCO1,,#3E18
```

```
 85  COM VARCO2,,#3F46
 86  COM VIRCO1,,#4074
 87  COM VIRCO2,,#407B
 88  COM WCNCOM,,#40C5
 89  COM VIRCO3,,#413F
 90  COM VIRCO4,,#41FF
 91  COM CNSCOM,,#425F
 92  COM VIRCO5,,#4329
 93  COM VIRCO6,,#4359
 94  COM GBOCO1,,#437D
 95  COM CONCOM,,#5B83
 96  COM QLOCOM,,#5D79
 97  COM STSCO1,,#5D83
 98  COM DFSCOM,,#5D91
 99  COM STSCO2,,#5D97
100  COM PARCOM,,#5DA3
101  COM STSCO3,,#5F35
102  COM IOCOM1,,#5F71
103  COM D1SCO1,,#5F7A
104  COM STSCO4,,#6B34
105  COM IOCOM2,,#6B70
106  COM D1SCO2,,#6B7B
107  COM EVNCOM,,#6D0B
108  COM IOCOM3,,#6E9E
109  COM LINCOM,,#6ECA
110  EDIT
111  WEOF BO
112  REW BO BI SC %4
113  EXI
114  $JOB SAVE THE LOAD MODULE FOR FUTURE USE
115  $EXE TOC
116  FIL %4
117  PEC REL
118  OVERLAY %1
119  CAT
120  EXI
121  $JOB EXECUTE "WC".
122  $WC %2,%3,%5,TY,%A
123  $JOB >>> ANALYZ ENDED. GOOD BYE <<<
124  $END
TOTAL RECORDS WRITTEN =   125/     15
POS $WC
```

```
 1  *_  PRODEFAULT WC SETUP,SPECT,SU,TY,WCNAME
 2  *_  GET SETUP (AND OPTIONALLY THE SPECTRA-BUFFER) FROM THE USL.
 3  *  SET ALL ASSIGNMENTS, AND EXECUTE THE OVERLAY THAT WSS CATALOGUED
 4  *  ON THE LOAD MODULE (BY TOC) UNDER THE NAME GIVEN BY PARAMETER#1
 5  *  USING THE PROCEDURE $ANALY.
 6  * P1 - THE NAME OF THE "SETUP" FILE ON THE USL. DEFAULT-SETUP.
 7  *  P2 - THE NAME OF THE "SPECTRA-BUFFER" ON THE USL. NO DEFAULT VALUE.
 8  *  P3 - THE LOAD MODULE ON WHICH "WC" TASK IS CATALOGED. DEFAULT-SU.
 9  *  P4 - THE ASSIGNMENT OF I/O UNIT 3. DEFAULT-TY.
10  *_ P5 - THE NAME OF "WC"-TASK ON THE LOAD-MODULE. DEFAULT-WC.
11  *_ SHLOMO WALD, LBL, MARCH 1983.
12  $PRODEFAULT WC SETUP,,SU,TY,WC
13  $IF %3=TU,P=$AVR CI 8
14  $RESTFI %1,SC0
15  $IFP %2,P=$RESTFI %2,SC2
16  $ASS 1=TY 2=TY 3=%4 4=PLA DSP=PLT 5=VP
17  $ASS 6=SC0 7=SC1 8=SC2 9=SC2
18  $ASS MTW=MT2
19  $REW 6,7,8
20  $EXE %5,%3
21  $AVR CI 8
22  $ASS USL RSL
23  $RESTFI %1,RR1
24  $IFP %2,P=$RESTFI %2,RR2
25  $ASS 1=TY 2=TY 3=%4 4=PLA DSP=PLT 5=VPP
26  $ASS 6=RR1 7=RR1 8=RR2 9=RR2
27  $ASS MTW=MT2
28  $REW 6,7,8
29  $EXE %5 TU
30  $WEOF 7
31  $WEOF 9
32  $WEOF 5
33  $JOB END OF OVERL PROCEDURE.
34  $END
TOTAL RECORDS WRITTEN =    35/      6
POS $RESTFI
LIST
```

```
  1  $PROCEDURE RESTFI LIBFIL SCRFIL
  2  $NOTE COP FIL (P1) TO SCn (P2). (PROTOCOL,SC0)
  3  $IFP %2,P=$AVR CI 2
  4  $ASS SO SC0
  5  $AVR CI 1
  6  $ASS SO %2
  7  $IFP %1,P=$AVR CI 10
  8  $EXE SED
  9  REW SO
 10  POS PROTOCOL
 11  NOCO
 12  ASS LO NO
 13  COPY
 14  WEOF SO
 15  REW SO
 16  EXI
 17  $AVR CI 9
 18  $EXE SED
 19  REW SO
 20  POS %1
 21  NOCO
 22  ASS LO NO
 23  COPY
 24  WEOF SO
 25  REW SO
 26  EXI
 27  $ASS LO LO SO SO
 28  $NOTE END OF "RESTFI" PROC.
 29  $END
TOTAL RECORDS WRITTEN =    30/      3
POS $SAVEFI
LIST
```

```
  1  $PROCEDURE SAVEFI LIBFIL SCRFIL
  2  $NOTE COPY TO USL FILE (P1) FROM SCn (P2). (PROTOCOL,SC1)
  3  $IFP %2,P=$AVR CI 2
  4  $ASS SI SC1
  5  $AVR CI 1
  6  $ASS SI %2
  7  $IFP %1,P=$AVR CI 6
  8  $EXE SED
  9  REW SI
 10  NOCO
 11  CAT PROTOCOL
 12  EXI
 13  $AVR CI 5
 14  $EXE SED
 15  REW SI
 16  NOCO
 17  CAT %1
 18  EXI
 19  $ASS SI SI
 20  $NOTE END OF "SAVEFI" PROC.
 21  $END
TOTAL RECORDS WRITTEN =    22/      3
POS $SCRFI
LIST
```

```
    1  $PRODEFAULT SCRFI,SCR
    2  $NOTE CREATE A SCREATCH FILE (P1). DEFAULT NAME-SCR.
    3  $CLO %1
    4  $FD0 INI VNA PACK02 TNA M1 ISA 1/T GEO 8 RSL 128 FSC 1/T AEI 1/T
    5  $FD0 FNA $TEMP:%1 FAC 1
    6  $CRE %1 FD0
    7  $OPE %1 FD0
    8  $END DO
TOTAL RECORDS WRITTEN =     9/      2
POS $DELFUL
LIST
```

```
    1  $PRODEFAULT DELFUL ,WCB
    2  $JOB DELETE SUBROUTINE "NAME" FROM THE BINARY LIB. (P2-DEFAULT UL)
    3  $EXE LIB
    4  ASS BI %2 BO SC0
    5  REW BI BO
    6  LNAM
    7  DEL %1
    8  COPY
    9  ASS BI SC0 BO %2
   10  REW BO BI
   11  COPY
   12  EXI
   13  $JOB END OF "DELFUL" PROC. GOOD LUCK.
   14  $END
TOTAL RECORDS WRITTEN =    15/      2
POS $ADDTUL
LIST
```

```
*E.0+1  MODCOMP SOURCE EDITOR      DATE  02/08/84  21:50:41        PAGE   1


  1  *   PRODEFAULT ADDTUL SUBNAME,WCB,FR5
  2  *  P1 - THE NAME OF THE SUB-PROGRAM TO BE COMPILED AND ADDED TO THE
  3  *       BINARY LIBRARY. NO DEFAULT.
  4  *  P2 - THE BINARY LIBRARY LOGICAL DEVICE NAME. DEFAULT-WCB.
  5  *  P3 - THE COMPILER TO BE USED. DEFAULT-FORTRAN 5.
  6  *  SHLOMO WALD, LBL, MARCH 1983.
  7  $PRODEFAULT ADDTUL ,WCB,
  8  $JOB ADD SUBROUTINES "NAME" TO THE BINARY LIB. P2.DEFAULT=WCB.
  9  $IFP %3,P=$AVR CI 2
 10  $COMPL5 %1,NOLO,NOMAP
 11  $AVR CI 1
 12  $COMPIL %1,NOLO,NOMAP
 13  $EXE LIB
 14  ASS BI %2 BO SC0
 15  REW BI BO
 16  NOF
 17  LNAM
 18  COPY
 19  ASS BI SCA
 20  FIL
 21  COPY
 22  ASS BI SC0 BO %2
 23  REW BO BI
 24  COPY
 25  EXI
 26  $JOB END OF "ADDTUL" PROC. GOOD LUCK.
 27  $END
TOTAL RECORDS WRITTEN =    28/      4
POS $DELADD
LIST
```

```
    1  *   PRODEFAULT DELADD ,WCB,
    2  *_ P1, P2 AND P3 AS DESCRIBED IN $ADDTUL.
    3  *_ SHLOMO WALD, LBL, MARCH 1983.
    4  $PRODEFAULT DELADD ,WCB,
    5  $JOB DELETE FILE AND ADD NEW VERSION TO BINARY LIB-P2. DEFAULT UL$
    6  $DELFUL %1,%2
    7  $ADDTUL %1,%2,%3
    8  $JOB END OF "DELADD" PROC. GOOD LUCK.
    9  $END
TOTAL RECORDS WRITTEN =    10/      3
POS $LNAMUL
LIST
```

```
*E.0+1  MODCOMP SOURCE EDITOR      DATE  02/08/84  21:51:02        PAGE   1


    1  $PRODEFAULT LNAMUL WCB
    2  $JOB LIST NAMES OF ALL SUBROUTINE ON THE BINARY LIB. (P1-DEFAULT WCB)
    3  $EXE LIB
    4  ASS BI %1 BO NO
    5  LNAM
    6  LIS
    7  EXI
    8  $JOB END OF "LNAMUL" PROC.
    9  $END
TOTAL RECORDS WRITTEN =    10/      2
POS $EDIT
LIST
```

```
 1  *_     EDIT[,NEW/NAME][,M/B/S]     EDIT "NAME" FROM USL IN RANDOM MODE;
 2  *                              SET TABS: (DEFAULT=FORTRAN)
 3  *                                        M=MACRO
 4  *                                        B=BCPL
 5  *                                        S=STRUCT FORT
 6  *                              IF "NAME" MISSING, DOES "RAN REC"!!!
 7  *                              IF "NAME" = "NEW", YIELDS EMPTY FILE.
 8  *  03C 23JAN80 CMI  USE REM AND CAT INSTEAD OF RECAT.
 9  *  03B 09JAN80 JLC  ADDED DATE OPTION
10  *  03A 20NOV79 RAB  CHANGES FOR SED VERSION E.0.A
11  *  02B 06AUG79 RAB  REMOVED UNNECESSARY ASS SI=SI:
12  *  02A 18APR79 RAB  ADDED HOLD OPTION
13  $PRO EDIT
14  $ASS SI=SI,SO=SO
15  $MOV CI=REP
16  $EXE SED,,NOLO
17  $IFP %2/P=CLE ALL
18  $IF %2=M/P=SET 9 25 41 73
19  $IF %2=B/P=SET 5  9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73
20  $IF %2=S/P=SET 8 13 18 23 28 33 38 43 48 53 58 63 68 73
21  OPT LIM DATE
22  MAC FF
23  F 22
24  P 22
25  ENDM
26  AVF REP,3
27  $IFP %1/P=AVF CI,1
28  RAN REC
29  AVF REP,1
30  ASS CI CI
31  $EOF            *** EDIT,NAME
32  $IF %1=NEW/P=AVF CI,1
33  ASS SI=SI
34  OPT HOL
35  POS %1
36  OPT NOHO
37  NOTE   Ready to edit "%1":  To save, type "ASS CI=REPLACE"
38  RAN
39  ASS CI=CI
40  $EOF            *** EDIT,NEW
41  ASS SI NO
42  RAN
```

```
   43  AVF REP,2
   44  ASS SI=SI
   45  ASS CI=CI
   46  $EOF
   47  OPT HOLD
   48  PAU            Type "R" to save "%1" in USL
   49  REM %1
   50  CAT %1
   51  EXI
   52  $NOP       "%1" has been removed and cataloged !!
   53  $ASS REP=REP,CI=CI,SI=SI,SO=SO
   54  $EOF
   55  PAU    Name unknown, use  "REC,NAME" !!
   56  BKF REP,1
   57  AVF REP,1
   58  ASS CI=CI
   59  $EOF
   60  PAU  This is a "NEW" file,  use "CAT,NAME" !!
   61  BKF REP,1
   62  AVF REP,1
   63  ASS CI=CI
TOTAL RECORDS WRITTEN =    64/      7
POS $EXCISE
LIST
```

```
    1  *- SHLOMO WALD, BLDG. 88,   MAY 1983
    2  $PROCEDURE EXCISE FILE1,FROM,TO,FILE2
    3  $JOB EXCISE FROM FILE1 (P1) LINES RANGE (P2,P3) TO FILE3 (P4).
    4  $EXE SED
    5  POS %1
    6  S %2
    7  ASS SO SC0 LO NO
    8  REW SO
    9  NOCO
   10  NOFIL
   11  G %3
   12  WEOF SO
   13  REW SO
   14  ASS SO SO SI SC0
   15  CAT %4
   16  EXI
   17  $JOB END OF EXCISE PROCEDURE.
   18  $END
TOTAL RECORDS WRITTEN =    19/      3
POS $INSERT
LIST
```

```
 1  *- PROCEDURE INSERT NAME1,LINE1,NAME2,LIN21,LIN22
 2  *  INSERT LINES LIN21 TO LIN22 FROM NAME2 PRIOR TO LINE
 3  *  LINE1 OF THE FILE NAME1.
 4  * THE USER CAN PUT A STRING (WITHOUT BLANKS) WITHIN ASTERISKS INSTEAD
 5  * OF A LINE NUMBER.
 6  *  DEFAULT FOR LINE1 AND LIN21 IS 1.
 7  *- DEFAULT FOR LIN22 IS THE LAST LINE IN NAME2.
 8  *- SHLOMO WALD, BLDG.88, X5080        AUG 1983.
 9  $PROCEDURE INSERT NAME1,LINE1,NAME2,LIN21,LIN22
10  $NOTE ! THE COMBINED FILE IS SAVED ON SC0 BEFORE IT IS CATALOGUED.
11  $EXE SED
12  ASS LO NO
13  POS %3
14  $IFP %4,P=AVR CI 2
15  S 1
16  AVR CI 1
17  S %4
18  ASS SO SC2
19  NOCO
20  NOFIL
21  $IFP %5,P=AVR CI 2
22  COPY
23  AVR CI 1
24  G %5
25  WEOF SO
26  ASS SO SO
27  POS %1
28  ASS SO SC0
29  NOCO
30  NOFIL
31  $IFP %2,P=AVR CI 1
32  AVR CI 1
33  G %2
34  ASS SI SC2
35  COPY
36  ASS SI SI
37  POS %1
38  $IFP %2,P=AVR CI 1
39  AVR CI 1
40  S %2
41  COPY
42  WEOF SO
```

```
   43  ASS SO SO
   44  ASS SI SI
   45  REM %1
   46  ASS SI SC0
   47  CAT %1
   48  EXI
   49  $JOB
   50  $END INSERT PROCEDURE.
TOTAL RECORDS WRITTEN =    51/      5
POS $CLG
LIST
```

*E.0+1 MODCOMP SOURCE EDITOR DATE 02/08/84 21:51:37 PAGE 1

```
    1  $PROCEDURE CLG FILNAM
    2  $JOB COMPILE FILENAME , LINK AND GO.
    3  $ASS 1,TY 2,TY 4,NST DSP,DSQ
    4  $ASS 3 TY
    5  $COMPL5 %1,NOLO,NOMAP
    6  $ASS UL ULB
    7  $REW BI BO UL
    8  $EXE MAIN,BI,LINK,NOMAP
    9  $END
TOTAL RECORDS WRITTEN =    10/      2
EXI

$$

$$
```

```
1  $PROCEDURE CHKMT
2  $JOB CERTIFY MAGENETIC TAPE
3  $COMPL5 CERTIFY,NOLO,NOMAP
4  $ASS UL ULB
5  $ASS 1 MT3 2 MT3 3 TY
6  $EXE MAIN,BI,LINK,NOMAP
7  $JOB >>> CERTIFY ENDED. GOOD BYE <<<
8  $END
```

```
      PROGRAM CERTIFY
C
C *** CERTIPY TAPES. VERSION 11/16/82.
C
      IMPLICIT INTEGER*2 (I)
      DIMENSION IV(8000)
      DATA IW,IR,IY,IANS2/'W ','R ','Y ','N '/
C     ASSIGNMENT OF I/O DEVICES.
C IO1-READ MT:
C IO2-WRITE MT:
C IO3-READ/WRITE TY:
C
C     FILL BUFFER.
      IO1=1
      IO2=2
      IO3=3
      DO 1 I=1,8000
1     IV(I)=(RANF(N)-.5)*65536.
      WRITE(IO3,10)
      WRITE(IO3,11) (IV(J),J=1,100)
10    FORMAT(/,' *** RANDOM BLOCK ***')
11    FORMAT(10I7)
      WRITE(IO3,12)
12    FORMAT(/,' *** CERTIFY TAPE PROCEDURE STARTS ***',/,
     + ' IF THE TAPE PASSES E.O.T. THE DRIVER WILL BE SET OFF LINE.'
     + ,/,' IF PARITY ERROR OCCURS, JUNK IT !!!.',/,
     + ' FULL PROCEDURE CONSISTS OF 2 STEPS-WRITE AND READ',/,
     + ' AFTER WRITE STEP THE TAPE IS REWINDED AND REREAD')
15    FORMAT(A2)
      IANS=IW
222   NR=0
2     WRITE(IO1,END=21,ERR=20) IV
      NR=NR+1
      GOTO 2
33    WRITE(IO3,100) NR
100   FORMAT(' # OF RECORDS WRITTEN IS',I6)
      READ(IO2,END=23,ERR=22) IV
      WRITE(IO3,11) (IV(J),J=1,100)
      NRR=NR-1
3     DO 333 N=1,NRR
333   READ(IO2,END=23,ERR=22) IV
      REWIND IO2
```

```
      GOTO 999
20    WRITE(IO3,16)
16    FORMAT(' >>> PARITY ERROR FOUND, CONTINUE ? (N/Y) >>> ')
      READ(IO3,15) IANS
      IF(IANS.EQ.IY) GOTO 2
      GOTO 900
21    GOTO 999
22    WRITE(IO3,17)
17    FORMAT(' >>> PARITY ERROR FOUND, CONTINUE ? (N/Y) >>> ')
      READ(IO3,15) IANS2
      IF(IANS2.EQ.IY) GOTO 3
      GOTO 900
23    GOTO 999
900   WRITE(IO3,18)
18    FORMAT(' *** THIS TAPE SHOULD BE JUNKED ***')
      GOTO 1000
999   WRITE(IO3,19) IANS
19    FORMAT(' *** THIS TAPE IS O.K. ***, (',A2,' STEP)')
      IF(IANS.EQ.IR) GOTO 1000
      IANS=IR
      REWIND IO1
      GOTO 33
C
1000  WRITE(IO3,1001)
1001    FORMAT(' CHACK ANOTHER TAPE ? (N/Y)')
      READ(IO3,15) IANS2
      IF(IANS2.NE.IY) GOTO 1010
      IANS=IW
      PAUSE NEWMT
      GOTO 222
1010  STOP
      END
```

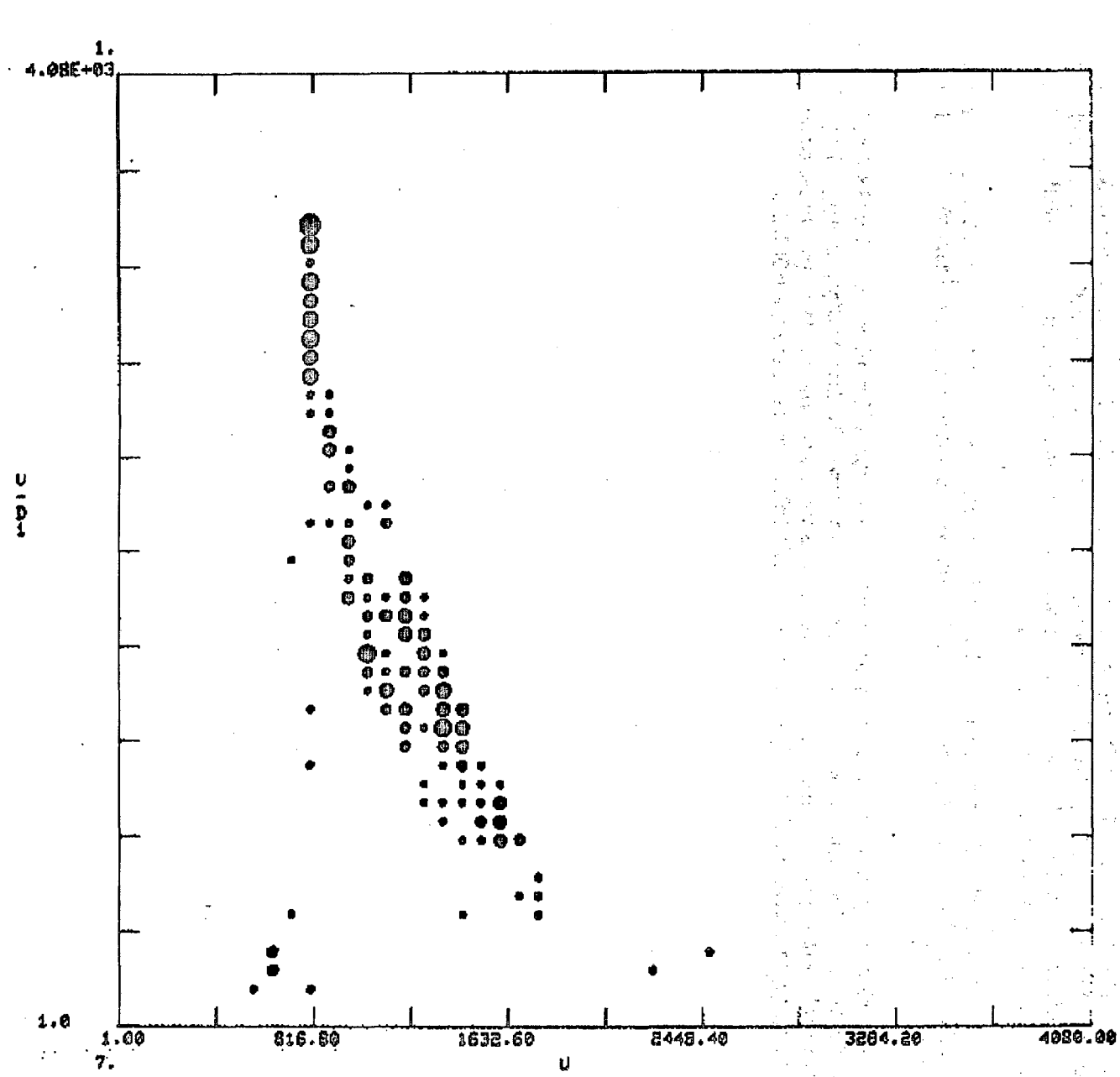
February 1984 WC Newsletter

1) Modcomp version is running on the NSD Modcomp CLASSICS, with extended memory available. It supports spectrum buffers of up to 131072 Integer*2 words (was previously 8424 words). 46568 bytes of memory are dedicated to the YFSR code, which was previously limited to 30,208 bytes.

2) A first version of an overlay for display of 2-D spectra now exists, called DSP2D. It is called by entering the subcommand D2 in response to either of the prompts DISP>>> or DSP1>. An example of a plotted 2-D spectrum is given in the attached figure. DSP2 commands are analogous to those of DSP1. Simply type "HELP" after entering the DSP2 overlay. A brief description of the DSP2 commands will appear on your terminal.

3) A version of WC is now available on VAX. WC is now operational on Modcop CLASSIC, IBM 3081 and 3083 (with VM/CMS operating system) and on VAX-11/780 under VMS.

4) The WC task and binary libraries are on public libraries at the 88-Inch Cyclotron Modcop B computer. The user can execute WC directly, using the $ANALY and $WC procedures. Note: If the user does not have a private binary library, parameter 6 of the procedure $ANALY should be set to NOUL.



Example of 2-D spectrum plotted by DSP2. Area of dots is proportional to value of spectrum at each point.

This report was done with support from the Department of Energy. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the Department of Energy.

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.

TECHNICAL INFORMATION DEPARTMENT
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720