UCRL-20160 c.2

USERS MANUAL FOR THE OPTIME SYSTEM

Philippe H. Eberhard and Werner O. Koellner (X-5711)

February 1971

AEC Contract No. W-7405-eng-48

LAWRENCE RADIATION LABORATORY
UNIVERSITY of CALIFORNIA BERKELEY

DISCLAIMER

CONTENTS

USERS MANUAL FOR THE OPTIME SYSTEM*

Philippe H. Eberhard and Werner O. Koellner

Lawrence Radiation Laboratory
University of California
Berkeley, California 94720

February 1971

ABSTRACT

This is a handbook for the use of the OPTIME SYSTEM FOR FITTING THEORETICAL EXPRESSIONS. It is based on Version 3.1 of the OPTIME system. The procedures of maintaining and distributing the program as well as the programming requirements on the part of the user are described.

Several optional features, among them easily useable subroutines to perform Monte Carlo integrations, to calculate error matrices, and to histogram selected quantities of the user's data, make the OPTIME system a reliable and versatile tool for the solution of fitting a distribution of data points to a mathematical expression.

Procedures to minimize computer time and some technical information about the structure of the system are also given.

Sections 1, 2, and 3 are probably sufficient for the user who wants to do a relatively simple fit. Other sections are important for more sophisticated or non-standard usage.

1.0 GENERAL PURPOSE AND DESCRIPTION

OPTIME is a system to perform and to check fits by maximum-likelihood or least-square methods.

Given experimental events with specification $|\xi|_k$, and given a theoretical distribution $y(x, a)$ such that $y(x, a)\,dx$ is an expectation value for the number of events in the hypervolume dx, OPTIME permits a user to find the values $|a|$ that represent the experimental data best. The value of the function $y(x, a)$ should be furnished by a user-coded routine for different sets of values of $|x|$ and different sets of values of $|a|$. The fitting is performed by maximizing a function $w(a)$ whose expression depends on the data $|\xi|_k$, on the user-coded function $y(x, a)$, and on the type of fit that the user may select. The maximization is performed by using stepping procedures described in [1].

The program contains options to produce printed or CRT histograms of selected quantities of the user's data, to superpose fitted curves on them, to calculate integrals of functions, and to compute error matrices.

1.1 The Structure of OPTIME

The Optime system consists of an assembly of subroutines. The user must add a main program and several other subroutines to make it an operational fitting program. It operates in three phases, which are in part determined by the requirements of the user. These phases are pre-fit processing, fitting, and post-fit processing. They will be described very briefly in the following sections.

In the pre-fit processes the experimental data points are read from a tape and only the pertinent information is kept for the fits and the post-fit processes. Integration points should be generated at that time if they are needed later. Also histograms should be prepared in that phase of the program.

The fitting begins with calling a routine MAXIME with some initial values for the parameters in a transfer vector. This vector will be returned filled with the best values found by the fit.

The post-fit processing contains optional features that the user may want to use to check his fits or to interpret the value of them. Plotting of histograms of datapoints along with a theoretical prediction, computation of

error matrices, and making integrals of the function are all part of the post-fit processing.

1.2 Pre-fit Processing

Pre-fit processing entails the preparation of data for the fitting routine and, if desired, the preparation of data to be histogrammed. It involves the reading and manipulation of experimental data as well as the preparation of integration points which may be generated by Monte Carlo methods.

1.21 Experimental data

The subroutine IMME controls the reading of experimental data. To read one experimental event IMME calls a subroutine GETUM1. The standard version contained in the OPTIME system reads a format called "ARROW" format [2] at Berkeley. For other formats the routine must be replaced. IMME makes one datapoint at a time available to the user supplied subroutine DOME. There the user may select, reject, or modify the information. For each datapoint that the user wishes to consider during the fitting, he writes all pertinent information into a transfer vector. This vector will be called one datapoint from here on. Unless the user specifies otherwise, a weight of 1.0 will be assumed for a datapoint.

1.22 Integration points

For all types of fit, except for type 11 (see under KTYPE, Section 3.2) integrations are required. The subroutine INTIME, called by the user from the main program, controls the generation of integration points and makes them also available to the subroutine DOME, similar to the procedure for experimental data. For generation of the integration points for use in numerical integrations a subroutine GETUM2 is called by INTIME.

For the cases that deal with particle interactions, a version of GETUM2 is incorporated in the system. That GETUM2 calls SAGE [3], a Monte Carlo event generator, to generate such events. The user specifies the number of events to be generated and other input quantities, like the number of particles in the final state, masses, and energy (see also sections 3.2 and 4.3).

The number of integration points that are required to make the integration statistically accurate can be reduced by providing so-called reference functions. This option, which will cause the overall execution time to be reduced, is described in Section 4.5.

The user can also supply his own integral as described in Section 7.32. A single point of integration is then generated for which the user supplies the value of the integral as he knows it.

1.23 Preparation of histograms

To prepare data for histogramming after the fitting phase, the user provides a subroutine HISTME, from which he calls a subroutine HIST after having stored in a transfer vector the quantity to be histogrammed for this datapoint and the histogram number. Superposition of a fitted curve to a histogram of experimental quantities in the post-fit period makes it necessary to generate integration points in the pre-fit process, because integration points are used for the display of the curve. Therefore, HIST must be called for the integration points as well as for the experimental events.

1.3 Fitting

The fitting consists of adjusting the parameters $|a|$ so that the theoretical function $y(|x|, |a|)$ fits the distribution of the experimental datapoints. The value of the function $y(x, a)$ is given by the user-coded subroutine HUME for the point of coordinate $|x|_\ell$ and for the parameters $|a|$.

The fitting is performed following a call from the user-supplied main program to the subroutine MAXIME. The routine makes the fit by maximizing a function w(a) that depends on the type of fit (estimator), on the datapoints, and on the function $y(x, a)$. By setting a common variable KTYPE to a specific value, the user may choose a particular stepping procedure and a particular type of estimator from those available in OPTIME. Different estimators correspond to different functions w(a) as described in [1]. For instance, for KTYPE=11, w(a) is the logarithm of the likelihood $\Sigma \log y(|x|_\ell, |a|)$.

Advice about what type should be used can be found under "KTYPE" in Section 3.2.

1.31 The user function

The function y(x, a) is referred to as the user function, a function of the variable parameters $|a|$ and the variable $|x|$, whose value is either the datapoint $|\xi|_k$ or the integration point $|x|_\ell$. The user must provide a subroutine HUME in which he calculates y from $|x|$ and $|a|$ given to him in transfer vector. He has to store the result in a proper location in COMMON.

The function y(x, a) is the expectation value of the distribution of datapoints in the space of $|x|$ if $|a|$ were the true values of the parameters (see Ref. [1], Section 1.2). When integration points with weight Ω_ℓ are involved in the fit, it is important that $\sum_\ell \Omega_\ell\, y(x_\ell, a)$ represent the true integral of y.

For instance, if SAGE is used, the weights of the integration points reconstitute phase space. To obtain the correct result, y(x, a) should be the square of the matrix element corresponding to the parameters $|a|$ and to the variable $|x|$.

For types 11 to 14, the function y has just the shape of the expected distribution and the fit won't try to normalizs it to the experimental data.

For types 3, 4 and 20 the function y is being adjusted so that the normalization of y to the total number of datapoints is of some influence in the fit.

1.32 Variable and constant parameters

Prior to calling MAXIME the user stores in a transfer vector PARME all parameters, fixed and variable, and any other information that he may want to have available in HUME, but that does not depend on the datapoint. In a companion transfer vector $|LOC|$ the user specifies which of these parameters are to be variable during this fit; he also must specify how many parameters are to be varied. If the fit is good, the values of the variable parameters $|a|$ that have been declared good are printed and left in a vector in blank COMMON. Those values are used for some optional calculations and for the computations of the error matrix. While checking and accepting that last set of values of $|a|$, MAXIME computes the next step. Without further checking that next step, MAXIME will return the value of PARME in the transfer vector with that last step included.

MAXIME may be called repeatedly to fit different sets of variable parameters, each time using the same vector PARME in the calling sequence. In that case the final values of the vectors PARME, UDA, and WGT from one fit will automatically become the initial values for the next fit, unless the user alters these values.

1.4 Post-fit Processing

Post-fit processing includes the optional calling of several available subroutines, for example, to calculate an error matrix, or to normalize and print histograms, if histogram data was prepared during the pre-fit processing.

The user may call the subroutine AROME to calculate and print the error matrix for the variable parameters. The subroutines NORME or ENORME may be called to calculate and print the integral of the function y by using the integration points. If the user wants a fitted curve on his histograms, FAME must be called. FAME will multiply a weight $\Omega_l^{(o)}$ associated with an integration point by the function y(x, a) given by the user-function routine HUME. Histogramming the integration points with this new weight will represent the fitted curve. To print histograms, the user finally calls the subroutine COCHIS [4].

2.0 EXAMPLE OF AN OPTIME PROGRAM

This section contains an example of an OPTIME user's program and the corresponding printout. It was coded for use at LRL on a 6600 computer (having characteristics described in Section 10.1). The card images for this same deck are also stored in the second file of the source program tape, which is available on request from Werner Koellner. The new user might well punch this deck as a starting point for the use of OPTIME.

2.1 Program Deck

```
                      PROGRAM SAMPLE(INPUT,OUTPUT,TAPE1=1,TAPE2=1,TAPE4=1,TAPE5=1,
                     .TAPE20=1,TAPE29=1,TAPE30=1)
000002                COMMON        Z(1)
000002                DIMENSION     IZ(1)
000002                EQUIVALENCE   (Z,IZ)
                C
000002                COMMON/HISTME/
                     .              DUMMY(3), IEVTH
000002                COMMON/NORME/
                     .              NEV,      WTEV,     NINT,     WTINT,    WTRAME,
                     .              WTHIST
000002                COMMON/WRIME/
                     .              KTEV,     KTINT,    KTSCR,    KTHISTR,  KTHISTF,
                     .              KTREAL,   KTFAKE,   IWRIME,   KWERR,    KTAPE,
                     .              NWKD
000002                COMMON/IMME/
                     .              LEVENT,   LBUFF,    LSTOR,    ITAPE,    NOEVR,
                     .              NOEVF,    LWORDL,   LEVF,     LEVR
                C
000002                DATA ITAPE/0/
000002                DATA LEVENT,LBUFF,KTEV,KTINT,KTSCR,KTREAL,KTYPE,KTHISTR/
                     X500,1027,1,2,30,5,13,20/
000002                DATA KTHISTF/29/
                C
000002                COMMON/GAME/
                     .              KTYPE,    KEND,     TPRINT
000002                 LOGICAL       TPRINT

                C
000002                COMMON/HUME/
                     .              Y,        ETA(2),   KDER,     KPT,      NW,
                     .              MULT,     KIND,     ICASE
                     .              /H1/HZ    /H2/DH    /H3/NH    /H4/IH    /TOTALS/NHISTS
000002                DIMENSION HZ(1),DH(1),NH(1),IH(1)
000002                DATA (NHISTS=1)
000002                DATA HZ,DH,NH,IH/-5.,0.5,20,5/

                C
000002                DIMENSION PARME(4),LOC(4),WGT(4),UDA(4)
000002                DATA PARME/
                     X 5.,0.,0.5,0./
000002                DATA LOC,WGT,UDA/1,3,2,4,4*1.,4*.01/
000002                DATA MSTEP,NBOUND,NPAR,MAXPAR,REF/
                     X 10,1,2,4,1.0/
                C
000002                COMMON/CASME/
                     .              NCASE
000002                COMMON/TOME/
                     .              HFAC,     NTOTF(20)
000002                COMMON/NUNAME/
                     .              XXX(3),   T2ND,     TCONST,   WI,       WR
000002                LOGICAL       T2ND,     TCONST
                C
000002                COMMON/NPR/NOVA
                C
000002                HZ = -10./3.
000003                DH = 1./3.
                C
```

```
C         .READ AND PROCESS REAL DATA
C
000005          CALL IMME
C
C         .GENERATE ALTOGETHER 1500 DATAPOINTS
C         .USE 500 DATAPOINTS FOR THE FIT AND ALL FOR HISTOGRAMMING
C
000006          NTOTF=500                                                    C
000007          HFAC=3.0                                                     C
000011          CALL INTIME(1)
C
C         .INITIALIZE OPTIME STORAGE (FOR CALLING NORME BEFORE MAXIME)
C
000012          CALL PREPME(PARME,LOC,WGT,UDA,MSTEP,NBOUND,NPAR,MAXPAR)
C
C         .CALCULATE THE INTEGRAL Y(X,PARME) DX AND STORE IN WI FOR FAME
C
000022          CALL NORME(WI,PARME,6HTEST1 )
C
C         .NORMALIZE HISTOGRAM WEIGHTS AND PLOT TO SEE DISTRIBUTION
C         .BEFORE THE FIT
C
000025          CALL FAME(PARME)
000027          CALL COCHISE
000030        2 CONTINUE
C
C         .DO A FIT OF NPAR (=2) PARAMETERS
C
000030          CALL MAXIME(PARME,LOC,WGT,UDA,MSTEP,NBOUND,NPAR,MAXPAR)
C
C         .CALCULATE THE ERROR MATRIX
C
000040          CALL AROME(PARME)
C
C         .NORMALIZE THE HISTOGRAM WEIGHTS USING HTE HUME FUNCTION
C         .WITH PARAMETERS ADJUSTED TO PLOT A FITTED CURVE
C
000042        1 CONTINUE
000042          CALL FAME(PARME)
000044          CALL COCHIS
000045          STOP
000047          END


PROGRAM LENGTH INCLUDING I/O BUFFERS
002371

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000043     2       -  000031

BLOCK NAMES AND LENGTHS
        -  000001     HISTME  -  000004     NORME   -  000006     WRIME   -  000013     IMME    -  000011     GAME    -  000003
HUME    -  000011     H1      -  000001     H2      -  000001     H3      -  000001     H4      -  000001     TOTALS  -  000001
CASME   -  000001     TUME    -  000025     NUNAME  -  000007     NPR     -  000001
```

VARIABLE ASSIGNMENTS

DH	-	000000C11	DUMMY	-	0CCC00C02	ETA	-	000001C07	HFAC	-	000000C16	HZ	-	000000C10	IH	-	000000C13
ITAPE	-	000003C05	IZ	-	000000C01	KTEV	-	C00000C04	KTHISTF	-	000004C04	KTHISTR	-	000003C04	KTINT	-	000001C04
KTREAL	-	000005C04	KTSCR	-	000002C04	KTYPE	-	C00000C06	LBUFF	-	000C01C05	LEVENT	-	000000C05	LOC	-	000106
MAXPAR	-	000125	MSTEP	-	000122	NBOUND	-	000123	NH	-	000000C12	NHISTS	-	C00000C14	NPAR	-	000124
NTOTF	-	000001C16	PARME	-	000102	REF	-	000126	TCONST	-	000004C17	TPRINT	-	000002C06	T2ND	-	000003C17
UDA	-	000116	WGT	-	000112	WI	-	000005C17	XXX	-	000000C17	Z	-	000000C01			

START OF CONSTANTS-000052 TEMPS--000100 INDIRECTS-000102

ROUTINE COMPILES IN 041400

```
                SUBROUTINE DUME(DATA)
000002          COMMON Z(1)
000002          DIMENSION IZ(1)
000002          EQUIVALENCE (Z,IZ)
000002          COMMON/HUME/Y,ETA(2),KDER,KPT,NW,MULT,KIND,ICASE
000002          COMMON/IMME/X(7),LEVF,LDATA
000002          DATA (IQUIT=4HQUIT )
000002          DIMENSION DATA(1)
000002          NW = 2
000003          ETA = 1.
000004          IF (KIND.EQ.1) GO TO 5
          C
          C        .KIND=2 - GENERATE INTEGRATION POINTS FLAT
          C                          BETWEEN -3.333 AND 3.333
          C        .GETUM2 IS NOT USED (SEE NEXT PAGE)
          C
000007          DATA(1) = 6.666*(RGEN(R)-0.5)
000014          DATA(2) = 0.8                                                  C
000015          MULT = 1
000016          RETURN
000017        5 CONTINUE
          C
          C        .KIND=1 - PROCESS REAL DATA
          C
000017          XK = WORD(31)
          C
          C        .MAKE A CUT ON THE DATA (IN THIS TEST ALL WILL PASS)
          C
000021          IF (XK.GT.3.333) RETURN
000026          IF (XK.LT.-3.333) RETURN
          C
          C        .MAKE ONE DATA POINT
          C
000031          DK=0.8                                                         C
000033          DATA(1) = XK
000034          DATA(2) = DK
000036          MULT = 1
000037          RETURN
000037          END


SUBPROGRAM LENGTH
000066

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
5       -  000020

BLOCK NAMES AND LENGTHS
        -  000001    HUME   -  000011    IMME   -  000011

VARIABLE ASSIGNMENTS
DK      -  000065    ETA    -  000001CC2 IQUIT  -  000062    IZ     -  000000C01 KIND   -  000007C02 MULT   -  000006C02
NW      -  00000SC02 R      -  000063    X      -  000000C03 XK     -  000064    Z      -  000C00C01

START OF CONSTANTS-000042       TEMPS--000052    INDIRECTS-000060
```

```
                SUBROUTINE GETUM2(IB,LW)
          C        .DUMMY ROUTINE TO ELIMINATE THE STANDARD GETUM2
000004            RETURN
000004            END


SUBPROGRAM LENGTH
000012

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS-000007        TEMPS--000010    INDIRECTS-000012

ROUTINE COMPILES IN 041000
```

```
                SUBROUTINE HISTME(DATA)
000002          COMMON Z(1)
000002          DIMENSION IZ(1)
000002          EQUIVALENCE (Z,IZ)
000002          COMMON/HUME/Y,ETA(2),KDER,KPT,NW,MULT,KIND,ICASE
000002          COMMON/NPR/NOVA
000002          DIMENSION PARME(1),DATA(1)
          C
          C         .PREPARE ONE HISTOGRAM OF THE QUANTITY STORED IN DATA(1)
          C
000002          CALL H(1,DATA(1))
000005          RETURN
000006          END


SUBPROGRAM LENGTH
000017

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
        - 000001    HUME    - 000011    NPR     - 000001

VARIABLE ASSIGNMENTS
ETA     - 000001002 IZ      - 000000001 PARME   - 000016    Z       - 000000001

START OF CONSTANTS-000011       TEMPS--000013    INDIRECTS-000015

ROUTINE COMPILES IN 041000
```

```
                SUBROUTINE HUME(PARME,DATA)
000004          COMMON Z(1)
000004          DIMENSION IZ(1)
000004          EQUIVALENCE (Z,IZ)
000004          COMMON/HUME/Y,ETA(2),KDER,KPT,NW,MULT,KIND,ICASE
000004          COMMON/NPR/NOVA
000004          DIMENSION PARME(1),DATA(1)
000004          COMMON/POW/A,B,CP,D
          C
          C         .SET UP PARAMETERS FOR THE FUNCTION XLIKE
          C
000004          A = PARME(1)
000004          B = PARME(2)
000006          CP= PARME(3)
000007          D = PARME(4)
000011          XK = DATA(1)
000012          DK = DATA(2)
          C
          C         .CALCULATE THE USER-FUNCTION Y(X,PAR)
          C                   CODED IN XLIKE
          C
000014          Y=XLIKE(XK,DK)
000017          RETURN
000017          ENTRY ALARME
          C
          C         .DEFINE BOUNDARY CONDITIONS
          C
000026          DATA(1) = 1.
000030          IF (PARME(1).LE.2.) DATA(1) = -1.
000034          TALARME=DATA(1).EQ.-1.0
000040          RETURN
000041          END


SUBPROGRAM LENGTH
000066

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
        - 000001    HUME    - 000011    NPR     - 000001    POW     - 000004

VARIABLE ASSIGNMENTS
A       - 000000C04 ALARME  - 000062    B       - 000001C04 CP      - 000002C04 D       - 000003C04 DK      - 000064
ETA     - 000001C02 IZ      - 000000C01 TALARME - 000065    XK      - 000063    Y       - 000000C02 Z       - 000000C01

START OF CONSTANTS-000044       TEMPS--000050    INDIRECTS-000054

ROUTINE COMPILES IN 041000
```

```
                      FUNCTION XLIKE(XK,DK)
000004                COMMON/POW/A,B,CP,D
000004                DIMENSION ALX(200),ALO(200)
000004                DATA NS,XNS,XKZ/29,30.,6.666/
000004                DATA NTLOG/0/
000004                IF (NTLOG.NE.0) GO TO 2
000005                NTLOG = 1
000005                XP = XKZ/(2.*XNS)
000010                DO 3 I=1,NS
000012                XKP = XP*(2*I-1)
000015                ALX(I) = ALOG(XKP)
000022                ALO(I) = ALOG(XKP+1.)
000031              3 CONTINUE
000033              2 CONTINUE
000033                RI = 0.
000034                DO 1 I=1,NS
000037                XKP = XP*(2*I-1)
000042                XL = ALX(I)
000043                XQ = ALO(I)
000045                RIS = EXP(XL*(A-2.-B*XQ+D*XQ**2))
000055                RIS=RIS*(EXP(-.5*((XK+XKP)/DK)**2)+CP*EXP(-.5*((XK-XKP)/DK)**2))
000077                RI = RI+RIS
000101              1 CONTINUE
000103                XLIKE=RI/DK
000104                RETURN
000105                END
```

SUBPROGRAM LENGTH
001006

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
2 - 000034

BLOCK NAMES AND LENGTHS
POW - 000004

VARIABLE ASSIGNMENTS

A	-	000000C01	ALO	-	000463	ALX	-	000153	B	-	000001C01	CP	-	000002C01	D	-	000003C01
I	-	001000	NS	-	000773	NTLOG	-	000776	RI	-	001002	RIS	-	001005	XKP	-	001001
XKZ	-	000775	XL	-	001003	XLIKE	-	000152	XNS	-	000774	XP	-	000777	XQ	-	001004

START OF CONSTANTS-000110 TEMPS--000114 INDIRECTS-000150

ROUTINE COMPILES IN 041000

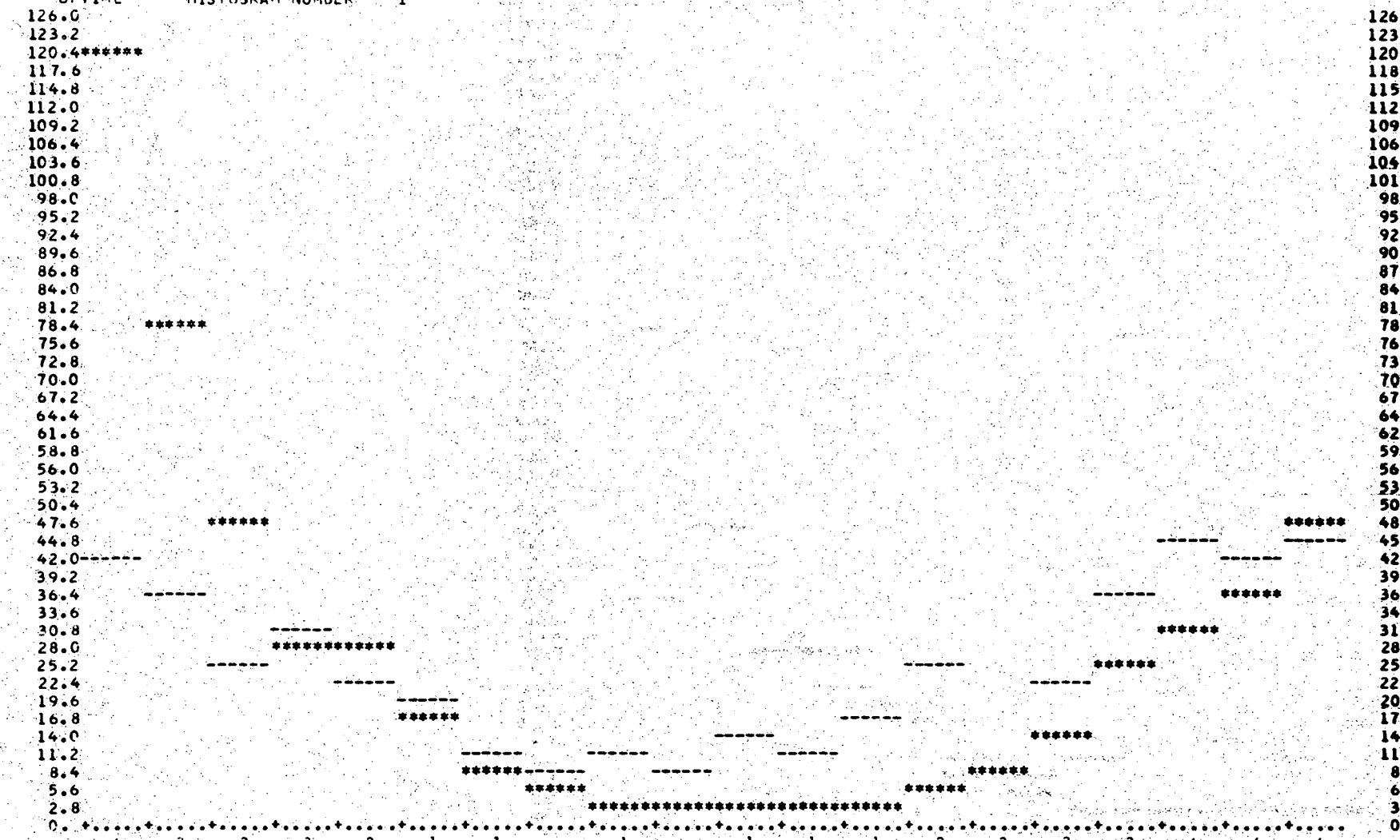
```
ZERO.2                                          IDENT  ZERO
                                  000016    PROGRAM LENGTH

                                            BLOCKS

                              000000  000016    PROGRAM*   LOCAL

                                            ENTRY POINTS

                                  000001 ZERO

ZERO.3                                                      ENTRY ZERO
ZERO.4          000000  3205221700000C000002      +         VFD  24/0HZERO,36/2
ZERO.5          000001  000000000CCC000000000     RETN      DATA 0
ZERO.6          000002  56120                               SA1  B2
ZERO.7                       43271                          MX2  57
ZERO.8                            11221                     BX2  X2*X1
ZERO.9                                63211                 SB2  X1+B1
ZERO.10         000003  6122777776                          SB2  B2-1
ZERO.11                           76600                     SX6  B0
ZERO.12         000004  0302000014 +                        ZR   X2,SOOP
ZERO.13                           63321                     SB3  X2+B1
ZERO.14         000005  6133777775                          SB3  B3-1
ZERO.15         000006  56610                     LOOP      SA1  B1
ZERO.16                      516  00001                     SA1  B1+1
ZERO.17         000007  5161000002                          SA6  B1+2
ZERO.18                           5161000003                SA6  B1+3
ZERO.19         000010  5161000004                          SA6  B1+4
ZERO.20                           51610C0005                SA6  B1+5
ZERO.21         000011  5161000006                          SA6  B1+6
ZERO.22                           5161000007                SA6  B1+7
ZERO.23         000012  6111000010                          SB1  B1+8
ZERO.24                           0713000006 +              LT   B1,B3,LOOP
ZERO.25         000013  0721000001 +                        LT   B2,B1,RETN
ZERO.26         000014  56610                     SOOP      SA6  B1
ZERO.27                      6111000001                     SB1  B1+1
ZERO.28         000015  0621000014 +                        GE   B2,B1,SOOP
ZERO.29                           0400000001 +              EQ   B0,B0,RETN
ZERO.30                             0000001 +   ZERO      EQU  RETN
ZERO.31         000016                                      END

                                  030652    UNUSED STORAGE          30 STATEMENTS      4  SYMBOLS


  LOOP      0000006     PROGRAM*        000012
  RETN      0000001     PROGRAM*        000013,   000015,   000016
  SOOP      0000014     PROGRAM*        000004,   000015
```

2.2 Printout

```
INPUT TAPE LABEL =          3          1       4444       50007012111539          4          0          0         42          9          1
THIS TAPE ENCOUNTERED          0 PARITY ERRORS,          0 RECORD CHECK SUM ERRORS AND          0 EVENT CHECK SUM ERRORS.
        500     EVENTS WERE READ. (TOTAL)

NUMBER OF REAL EVENTS ACCEPTED FOR FIT      500
SUM OF THE WEIGHTS       500.000
```



```
 THERE  ARE   500  POINTS  OF  INTEGRATION
AND  1 REFERENCE  FUNCTIONS  WHOSE  INTEGRALS  ARE
          1.000E+00 FOR REF. FCT.   1

NUMBER OF FAKED EVENTS ACCEPTED FOR FIT     500
SUM OF THE WEIGHTS         1.000
SUM OF THE WEIGHTS FROM RAME       500.000

    TEST1

 INTEGRAL OF Y =   1.05168E+02
```



TOTAL CONTENTS = 5.000000E+02 UNDERFLOW = 0. OVERFLOW = 0.

PROGRAM CONSTANTS

CHILIM= 1.0E-02 RLIM= 1.0E-10 ELIM= 1.0-200 TMAX= 9.0E-01 TMIN= 1.0E-01

TDIAG= F TCONST= F T2ND= F TPRINT= T TDIRCO= F

FIT NUMBER 1

NUMBER OF VARIABLE PARAMETERS LPARME= 2
HIGHEST ORD.NO. OF VAR.PARAMS LMAX= 4
MAX.NUMBER OF ATTEMPTED STEPS MSTEME= 10

	PARME ARRAY TO LMAX	VARIABLE PARAMETER LOCATION	UNWEIGHTED DERIVATIVE INCREMENTS	WEIGHT FOR DERIVATIVE
1)	5.000E+00	1	1.000E-02	1.00
2)	0.	3	1.000E-02	1.00
3)	5.000E-01			
4)	0.			

TYPE OF FIT MAX.LOG KTYPE= 13
NUMBER OF EQUATIONS FOR BOUNDS 1
NUMBER OF TAPES (FILES) USED 6
KTEV= 1 KTINT= 2 KTSCR= 30 KTHISTR= 20 KTHISTF= 29 KTREAL= 5 KTFAKE= X

AT STEP NUMB 0 FIT NUMBER 1

PARAMETERS

LOCATION IN PARME ARRAY	CENTRAL VALUE	INCREMENT FOR DERIVATIVE	DERIV.FLAG W W D D P M P M	PARTIAL DNS	COMMENT	NEXT PROPOSED STEP	BEST VALUE SO FAR
1	5.000E+00	1.000E-02		1.052E+02		-1.318E+00	5.000E+00
3	5.000E-01	1.000E-02		5.406E+01		3.169E-01	5.100E-01

FUNCTION SCM= 211

CENTRAL VALUE	TOTAL DNS	GRADIENT DOT STEP	BEST VALUE
-81.66	1.592E+02	1.592E+02	-79.57

****

AT STEP NUMB 1 FIT NUMBER 1

PARAMETERS

LOCATION IN PARME ARRAY	CENTRAL VALUE	INCREMENT FOR DERIVATIVE	DERIV.FLAG W W D D P M P M	PARTIAL DNS	COMMENT	NEXT PROPOSED STEP	BEST VALUE SO FAR
1	3.682E+00	1.472E-01		2.930E+01		-6.729E-01	3.535E+00
3	8.169E-01	4.310E-02		1.730E+01		3.122E-01	8.169E-01

FUNCTION SCM= 221

CENTRAL VALUE	TOTAL DNS	GRADIENT DOT STEP	BEST VALUE
33.13	4.660E+01	4.660E+01	38.79

****

AT STEP NUMB 2 FIT NUMBER 1

LOCATION IN PARME ARRAY	CENTRAL VALUE	INCREMENT FOR DERIVATIVE	DERIV.FLAG W W D D P M P M	PARTIAL DNS	COMMENT	NEXT PROPOSED STEP	BEST VALUE SO FAR
1	3.009E+00	1.304E-01		3.729E-01		-6.665E-02	3.017E+00
3	1.129E+00	7.506E-02		9.524E-01		1.041E-01	1.204E+00

FUNCTION

SCM= 221

CENTRAL VALUE	TOTAL DNS	GRADIENT DOT STEP	BEST VALUE
59.74	1.325E+00	1.325E+00	60.14

****

AT STEP NUMB 3 FIT NUMBER 1

PARAMETERS

LOCATION IN PARME ARRAY	CENTRAL VALUE	INCREMENT FOR DERIVATIVE	DERIV.FLAG W W D D P M P M	PARTIAL DNS	COMMENT	NEXT PROPOSED STEP
1	2.943E+00	1.148E-01		1.752E-06	ADJUSTED	7.244E-05
3	1.233E+00	1.066E-01		3.020E-03	ADJUSTED	6.416E-03

FUNCTION

SCM= 321

CENTRAL VALUE	TOTAL DNS	GRADIENT DOT STEP
60.41	3.022E-03	3.022E-03

****

END OF FIT NUMBER 1

BEST FIT FOUND UNDER ABOVE CONDITIONS

NUMBER OF STEPS ACCEPTED	3	NUMBER OF DATA POINTS	500
NUMBER OF ATTEMPTED STEPS	4	NUMBER OF INTEGRATION POINTS	500

THE RETURNED PARAMETERS CORRESPOND TO STEP 4

	PARME ARRAY TO LMAX	VARIABLE PARAMETER LOCATION	UNWEIGHTED DERIVATIVE INCREMENTS	WEIGHT FOR DERIVATIVE
1)	2.943E+00	1	1.148E-01	1.00
2)	0.	3	1.066E-01	1.00
3)	1.240E+00			
4)	0.			

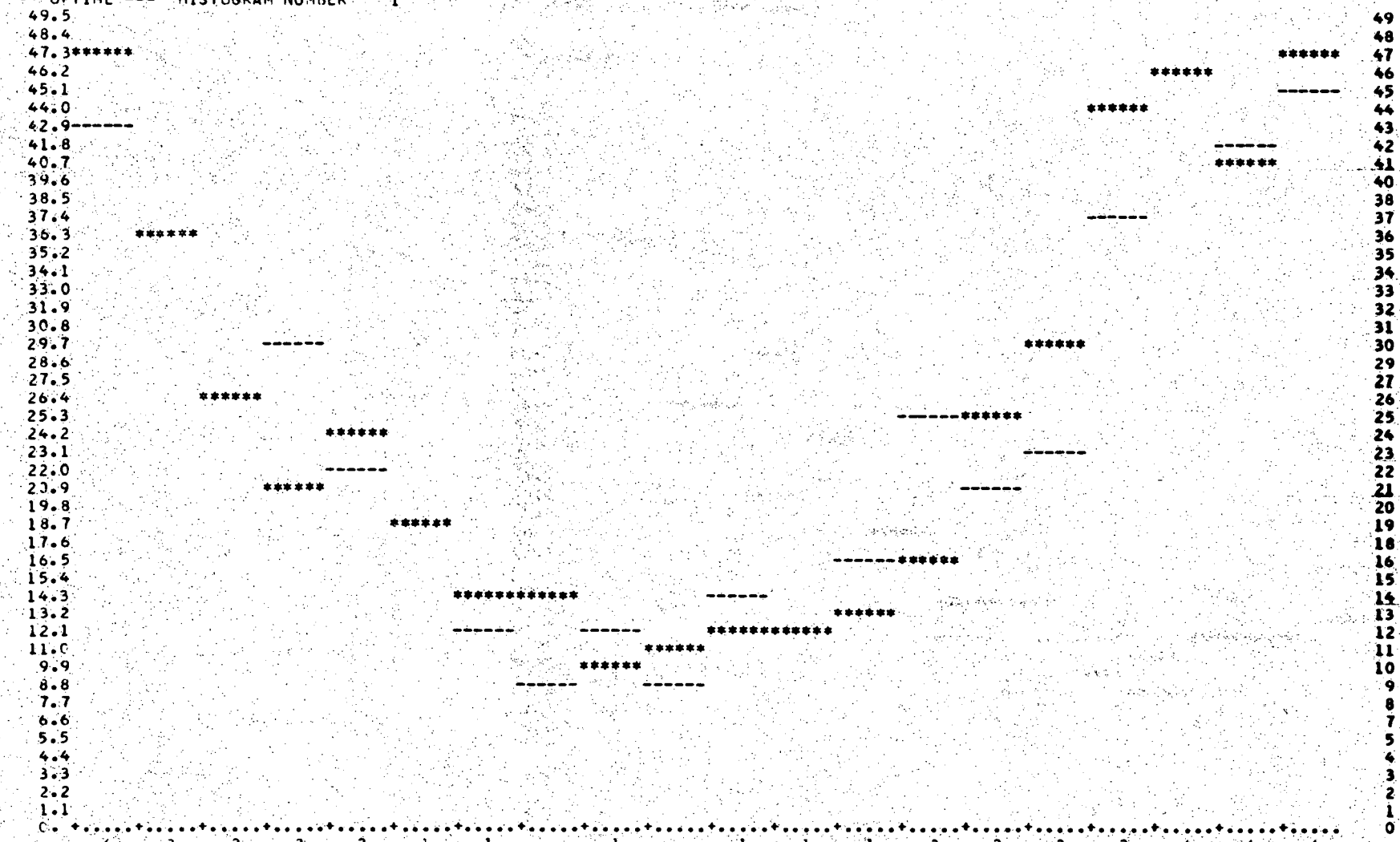
****

```
XXXX  X   X  XXXX        XXX   XXXX       XXXX  XXX  XXXXX
X     XX  X  X   X      X   X  X          X     X      X
XXXX  X X X  X   X      X   X  XXXX       XXXX  X      X
X     X  XX  X   X      X   X  X          X     X      X
XXXX  X   X  XXXX        XXX   X          X     XXX    X
```

****

ERROR MATRIX

1.27464E-02 4.71150E-04
4.71150E-04 1.36297E-02


--- OPTIME --- HISTOGRAM NUMBER 1
49.5
48.4
47.3
46.2
45.1
44.0
42.9
41.8
40.7
39.6
38.5
37.4
36.3
35.2
34.1
33.0
31.9
30.8
29.7
28.6
27.5
26.4
25.3
24.2
23.1
22.0
20.9
19.8
18.7
17.6
16.5
15.4
14.3
13.2
12.1
11.0
9.9
8.8
7.7
6.6
5.5
4.4
3.3
2.2
1.1
0.
TOTAL CONTENTS = 5.CCCCCE+C2 UNDERFLOW = 0. OVERFLOW = 0.

```
01.45.27. SAMPL00. SAMPL00*12,500,60C00.410702,KOELLNER
01.45.27. SAMPL00. **BKY43A*B*12/16/70.6012
01.45.28. SAMPL00.  OUTPUT  CC 77 0C31 0341
01.45.29. SAMPL00. FLOOR(5)
01.45.35. SAMPL00. REQUEST A,17980.
01.49.34. SAMPL00.  A       MT 5C M 17980
01.49.42. SAMPL00. COPY(A/RB,1F,TAPE5/BR)
01.49.42. SAMPL00.  TAPE5   DB 02
01.49.46. SAMPL00. COPY(A/C,1F,IN1/BR)
01.49.46. SAMPL00.  IN1     DA 00
01.49.53. SAMPL00. COPY(A/C,1F,NULL)
01.49.53. SAMPL00.  NULL    NE 03
01.50.07. SAMPL00. COPY(A/C,1F,IN2/BR)
01.50.08. SAMPL00.  IN2     DB 02
01.50.08. SAMPL00.  DAYFILE DA CC
01.53.4C. SAMPL00. RETURN(A)
01.53.40. SAMPL00.  MT5C, M  022205R 0000RP  0CC000W 0000WP
01.53.40. SAMPL00.  A       RETURNED
01.53.41. SAMPL00. RUNF(S,,1001,IN1,,,200000)
01.53.43. SAMPL00.  LGO     DA 00
01.53.43. SAMPL00.          COMPILING  SAMPLE
01.53.48. SAMPL00.  FL NEEDED BY RUNF= 041400
01.53.48. SAMPL00.  TIME 000.647 SEC.  RUNF(26AUG.41)
01.53.48. SAMPL00. RUNF(S,,1001,IN2,,LIBRY,200000)
01.53.50. SAMPL00.  LIBRY   DA 00
01.55.05. SAMPL00.  CMPSCR  DA 00
01.55.53. SAMPL00.  FL NEEDED BY RUNF= 050500
01.55.53. SAMPL00.  TIME 041.317 SEC.  RUNF(26AUG.41)
01.55.54. SAMPL00. REWIND(LIBRY)
01.55.54. SAMPL00. LODE(I=LGO,L=LIBRY,F=0,M=LMAP)
01.55.57. SAMPL00. XEQ.
01.55.57. SAMPL00.  RANDOMA DB 02
01.56.20. SAMPL00.  LMAP    DB 02
01.56.21. SAMPL00.  XEQ FL   036750     LODE FL  051131
01.56.21. SAMPL00. *GPA**GPA* OPTIME
01.56.21. SAMPL00.  NEW FL = 046000.
01.56.21. SAMPL00.  TAPE1   DA 00
01.56.21. SAMPL00.  TAPE2C  DB 02
01.56.24. SAMPL00.  TAPE5   RETURNED
01.56.24. SAMPL00.  NEW FL = 043100.
01.56.25. SAMPL00.  TAPE3C  DB 02
01.56.25. SAMPL00.  TAPE29  DA 00
01.56.27. SAMPL00.  TAPE2   DA 00
01.56.28. SAMPL00.  NEW FL = 046000.
02.01.18. SAMPL00. STOP       SAMPLE
02.01.18. SAMPL00. EXIT.
02.01.19. SAMPL00.  PRIME AUS LEFT -  PRI      RUSH
02.01.19. SAMPL00.                  60,421    16,827
02.01.20. SAMPL00. CP     165.776 SEC.
02.01.20. SAMPL00.       10,957 REQ.
02.01.20. SAMPL00.       342.071 AUS.
02.01.20. SAMPL00.        22.715 $$$
```

3.0 DESCRIPTION OF USER-FURNISHED ROUTINES

This section contains a detailed description of the requirements for the routines that must be provided by the user to make OPTIME an operational fitting program. Section 4.0 describes routines that may be furnished in some cases. Calls to these routines are normally satisfied by dummy routines within the OPTIME system when not supplied by the user.

3.1 Notation

The notation used in the following descriptions makes implicit reference to FORTRAN coventions. On the left-hand side we usually indicate the form of the coding that is required, thus

DIMENSION WGT (NPAR) means that a Fortran Dimension Declaration is required that reserves at least NPAR words for the variable WGT.

DATA (KTEV=1) means that the variable KTEV may be set to the value 1 by a Fortran Data Declaration, and DATA (LOC (i)) means that the vector |LOC| may be filled with a Fortran Data Declaration.

The word "fill" or "set," as for example in "set NCASE=k," will mean that the operation is to be done with an executable statement.

COMMON // means that the variables in the list are declared in Blank Common. COMMON/NAME/ signifies a labelled COMMON.

On the right-hand side of the page you will find explanations or descriptions of the effect that the coding statement will have.

All examples are coded to run on the computers at LRL.

3.2 The MAIN Program

```
 PROGRAM NAME(INPUT,OUTPUT, TAPE1=1, TAPE2=1, TAPE4=1,
CTAPE5=1, TAPE20=1, TAPE29=1, TAPE30=1)
 COMMON/CASME/
C                NCASE
 COMMON/GAME/
C                KTYPE,   KEND,    IPRINT
 COMMON/IMME/
C                LEVENT,  KBUFF,   LSTOR,   ITAPE,   NOEVR,
C                NOEVF,   LWORDL,  NEVF,    NDATA
```

```
 COMMON/NUNAME/
C                NMAT,    NBUFF,    DACUTM,  T2ND,     TCONST,
C                WI,      WR
 LOGICAL         T2ND,    TCONST
 COMMON/RENO/
C                NP,      ECM,      AMASS(10)
 COMMON/SAGME/
C                PLAB,    PCM(4,11),P(8)
 COMMON/TOME/
C                HFAC,    NTOTF(20)
 COMMON/WRIME/
C                KTEV,    KTINT,    KTSCR,   KTHISTR, KTHISTF,
C                KTREAL,  KTFAKE,   KWRIME,  KWERR,   KTAPE,
C                NWRD
 COMMON/ULABEL/ILABEL(3)                                        needed
 COMMON/HISTLB/HLAB(3,i)                                        for
 COMMON / H1/HZ / H2/DH / H3/NH / H4/IH / TOTALS/NHISTS         histo-
                                                                gramming
 DIMENSION       PARME(MAXPAR)      where MAXPAR = total number
                                    of parameters.
     "           LOC(NPAR)
     "           WGT(NPAR)          NPAR = number of variable
                                    parameters.
     "           UDA(NPAR)
     "           RES(NPAR + 1)
     "     HZ(i), DH(i), NH(i), IH(i)   for histogramming; i = max
                                    number of histograms.
 DATA            (ITAPE = k)        Controls reading of input file
                                    KTREAL (experimental data)
                                    by the standard subroutine
                                    GETUM1.
                                    k = 0 → only one reel of data;
                                      file will be returned at end
                                    k = 1 → if more than 1 reel of
                                      data
```

		k = 2 → one reel of data; will be left rewound at end.
DATA	(LEVENT = j)	length of unpacked real event; j must not be less than 500 when data in ARROW format is read. This area will be filled by unpacking routines (WORD, etc.).
		The following seven variables map the file-names defined on the above program card into OPTIME. The values of these variables and which ones need be set is in part determined by the files declared on the program card. The following examples correspond to the above program card.
DATA	(KTEV = 1)	Necessary when experimental data are used during fitting (written by IMME)
"	(KTINT = 2)	when integration points are used (written by INTIME).
"	(KTSCR = 30)	when integration points are used (written by INTIME).
"	(KTREAL = 5)	when experimental data are to be read (read by GETUM1).
"	(KTFAKE = 4)	when integration points are already prepared on tape (requires special version of GETUM2).
"	(KTHISTR = 20)	when histogramming experimental data (written by HIST).

DATA	(KTHISTF = 29)	when histogramming integration points to produce a fitted normalized curve (read by FAME).
"	(IPRINT = 0)	when it is desired to suppress printing during stepping (only initial and final print will appear).
"	(NTOTF(1) = k)	k is the number of integration points that are to be used for fitting and histogramming (see also HFAC below).
"	(PARME(i))	with starting values for all parameters.
"	(LOC(i))	with integers pointing to the location of the variable parameters in PARME. The numbers in LOC need not be ordered in a particular way, thus LOC can be used to fit parameters in a different order than they are stored in PARME. For example, to vary PARME(8) and PARME(2) one might have DATA (LOC=8, 2, 1, 3, 4, 5, 6, 7) and then in the call to MAXIME specify NPAR=2. Computing time can sometimes be saved if LOC(i) is filled with a negative pointer. Then the derivative for the (ABS(LOC(i)))th parameter in PARME is evaluated only in one (positive) direction. See

		also Ref. [1], Section II. 3.
DATA	WGT(i)	with the weights for the derivative increments. They are involved in the computation of the derivative increments and should usually be set to 1.0. If the user supplies his own derivatives he must set WGT(j)=0.0 to indicate to OPTIME that he furnishes the derivatives for the parameter specified by $\|LOC\|_j$ (see Sections 7.1 and 7.31).
DATA	UDA(i)	with the initial derivative increments. Note that when filling WGT and UDA one should keep in mind the way LOC was filled, since WGT(i), for example, will apply to the (ABS(LOC(i)))th parameter in PARME.
DATA	(KTYPE = k)	this is set to an integer to select a type of fit. This variable allows choosing of a function w(a) and a method of computing the step. The exact description of those types of fit are given in Ref. [1]. A complete fit of the function y(x, a) to the data with shape and normalization of the distribution requires type 3, 4, or 20. Type 20 is in general faster but less accurate than

type 3 or 4. Types 3 or 4 correspond to the same function w(a), but different mathematics for the step computation. If y is reasonably linear in $|a|$, type 4 is faster on the computer. Otherwise type 3 may be preferable. Types 11, 12, 13 and 14 are a fit of the shape of the distribution y only. Types 11 and 12 use the same function w(a); they can be used only if the integral of y is constant as a function of the parameters $|a|$. Types 13 and 14 correspond to the same function w(a). They compute the integral of the function y and its derivative at each step. Again types 12 and 14 have a faster convergence if y is reasonably linear. Otherwise types 11 and 13 may be preferable. All types except type 11 require integration points.

When histograms are being made by the user the following variables should be filled for use in COCHIS:

DATA(NHISTS=i) — i is the total number of histograms desired.

DATA(HZ(n) = r_n), $1 \leq n \leq i$ — specify lower limit of n^{th} histogram.

DATA(DH(n) = t_n), $1 \leq n \leq i$ — specify bin width of n^{th} histogram.

DATA (NH(n) = l_n), $1 \leq n \leq i$

specify number of bins in n^{th} histogram.

DATA (IH(n) = k_n), $1 \leq n \leq i$

specify form of n^{th} histogram.

where IH(n) = 1

histogram only experimental data.

= 2

histogram experimental data with fitted curve superposed.

= 3

same as 2 plus χ^2 printed.

IH(n) = 4, 5, 6

so-called "big-bin" histograms, with specifications as IH = 1, 2, 3; see also Ref. [5].

DATA ILABEL/1/

ILABEL(1) = 1 signals that headings (labels) are to be printed on all histograms. ILABEL(2) and ILABEL(3) are not used here.

DATA (HLAB(j, i), $1 \leq j \leq 3$, $1 \leq i \leq$NHISTS)

when ILABEL(1) = 1 fill HLAB with headings to be printed for each histogram. Each heading will consist of up to three ten-character words in Hollerith (A10) format.

If desired set T2ND = . TRUE.

Normally T2ND = . FALSE. and the standard procedure during fitting is to calculate the function w($|a|$), all its derivatives, and the E matrix for each attempted step before checking if the function has improved. This procedure takes the least amount of time as long as the function is well behaved and most attempted steps are accepted. If, however, many

steps are attemped unsuccessfully, i.e., refused for lack of improvement of w(a), the user should set T2ND = .TRUE. before calling MAXIME. Then before the derivatives are computed a subroutine CALME will be called to calculate only the function and it will check immediately if there has been an improvement or not.

Set HFAC = t

$1.0 \leq t$; setting HFAC is necessary only when $t > 1.0$ because the default value of 1.0 is set by OPTIME with a data statement.

The total number of integration points that will be generated is NTOTF(1)*HFAC. Of this, NTOTF(1) integration points will be considered for use in the fit, while all points will be considered for histogramming. Thus, HFAC determines the number of extra integration points that will be generated to produce smoother fitted curves in the histogram.

If desired set NBUFF = n ($n > 1027$)

Normally NBUFF is set to 1027 by a data statement in OPTIME. This determines the size of the buffer, and thereby the max.

size of the logical records, namely (NBUFF-3)/2, for all files written by OPTIME except those for COCHIS. In order to decrease the cost of computing, by decreasing the number of monitor requests it might be advantageous to increase NBUFF.

Pre-fit processing:
Preparation of experimental data:
CALL IMME

This routine processes experimental data and prepares it in a format suitable for the fit.
It calls a routine GETUM1 which reads the data from the file KTREAL (see also Section 4.2), and presents one event at a time to the user-subroutine DOME. There the user decides if the event should be used in the fit and/or in the histogram, or not at all.
After DOME the user-subroutine HISTME is called, from where the user may call the routine HIST to prepare histograms.
The events that were selected for the fit are finally written to the file KTEV, while histogram quantities for those events to be used in the histogramming are written to the file KTHISTR.

Preparation of integration data (not necessary for the fit when KTYPE=11):

Set NP =	to the number of particles in the final state.
fill AMASS(i) =	with their masses.
CALL INITL (B, T, ECM, PLAB, X)	This routine need be called here only when the user chooses to use the standard routine GETUM2 which generates fake particle interaction and decay final states, in the c.m. frame, flat in masses and angles (see also Section 4.3).
Specify: B =	Mass of the beam particle
T =	Mass of the target
PLAB =	Laboratory beam momentum
	The routine will return the c.m. energy in ECM, which is a variable in the COMMON/RENO/; also, COMMON/INIT/BM(4), TG(4) will be filled with beam and target 4-vectors respectively.
CALL INTIME(NREF)	This routine controls the generation and preparation of integration points.
Specify: NREF =	number of so-called reference functions that the user might provide via the user-subroutine FLAME (see Section 4.5). If the user does not program his own routine FLAME, to provide "reference functions" he should call INTIME with NREF = 1. INTIME calls the routine

GETUM2, which performs the generation with the aid of the subroutine package SAGE [3], and then presents one event at a time to the user subroutine DOME and HISTME just as described for the experimental data.

Finally the integration points that were selected for the fit are written to file KTINT, while all integration points to be used for histogramming fitted curves together with their histogram quantities are written to the file KTSCR.

Begin fitting:

CALL MAXIME(PARME, LOC, WGT, UDA, MSTEP, NBOUND, NPAR, MAXPAR)

Specify: MSTEP = maximum number of steps to be attempted in this fit.

NBOUND = number of boundary conditions that the subroutine ALARME will test when testing for boundary violations. See Section 4.1.

NPAR = number of variable parameters

MAXPAR = number of total parameters in PARME array (i.e., length of the vector PARME).

All other input quantities were already filled with data state ments.

If NBOUND > 0, ALARME will be called during the fitting and the returned values used as described

in [1]. Fitting proceeds then with the routine printing information at each step and at the end of the process.
At the conclusion of the fit the user will find in PARME(i), WGT(i), UDA(i) the final values for those variables. Thus if MAXIME were called again without altering the transfer vector, the starting values for that fit would be the final values of the last fit.
The variable KEND in COMMON/GAME/ indicates the type of ending of the fit that occurred. Upon successful conclusion of the fit the value of KEND will be 1. Other values and their meanings are described in Section 6.1.

Post-fit processing:
CALL AROME

This routine may be called by the user immediately after MAXIME to calculate and print the error matrix (ERR). If the fit is meaningless the routine will just return with a comment.
If the fit is good the matrix (ERR) is calculated and AROME will print it and store it in blank COMMON. Please see Section 5.3 for details.

Note: The following post-fit routines may be called before or without fitting. In that case, the subroutine PREPME must be called ahead of them at least once to initialize some post-fit requirements (see Section 5.1).

CALL NORME(RES, PARME, ICOM)

This routine may be called by the user to numerically calculate $\int y(|x|, |a|)\, dx$ in the space of x, where $y(|x|_\ell, |a|)$ is the function programmed in HUME. Obviously, integration points must have been prepared.

The answer is returned in RES(1) but will not be printed when ICOM = 0. If the user sets ICOM equal to a comment of up to 10 Hollerith characters, e.g., CALL NORME(RES, PAR, 10HFIRST CALL), the comment together with the answer from NORME will be printed (see Section 5.2).

of CALL ENORME(RES, PARME, ICOM)

This routine calculates not only the integral of $y(|x|, |a|)$ but also the NPAR derivatives of the integral of y, where NPAR is the number of variable parameters of the previous fit or transmitted by PREPME. The integral of y will returned in RES(1), while the derivatives of the integral will be in RES(j), $2 \leq j \leq$ NPAR+1. All other comments under "CALL NORME" are true here also.

For histogramming:
CALL FAME(PARME)

This routine must be called if the user made calls to HIST from HISTME for integration data, i.e., if he wanted fitted curves superposed in the histograms.

FAME and COCHIS may be called several times, e.g., with different values in the PARME array. FAME calls the subroutine HUME with that PARME in the transfer vector and with the variable KIND set to 3. The user function y(x, a) is used to calculate a new weight for each integration point. For types 11, 12, 13, and 14, where the fit is independent of the normalization of y, FAME also uses the integral WI = $\int f(x)\,dx$, among other quantities, to correct the weights of the integration points that are used in displaying fitted curves. They will be normalized to the experimental events in the region which is covered by the fit (see Ref.[1], Section III. 4). Note that, when KTYPE = 11, WI is not calculated during the fit, since no integrations are done there, and the user must fill WI with the value of the integral

of y prior to calling FAME. NORME may be used to calculate WI.

CALL COCHIS

This routine is called to print histograms of the experimental data with or without fitted curves superposed according to the way they were prepared during the pre-fit processing by the user's calls to HIST from HISTME (see Section 3.32). Note: The user may also say "CALL COCHISE."

STOP
END

3.3 User Routines Called During Pre-fit Processing

3.31 Subroutine DOME

This routine is called once per event either from IMME when experimental events are being processed or from INTIME when integration data is being generated.

The purpose of this routine is to allow the user: to make selections (cuts) from the set of experimental or fake events that are on hand, to make preliminary fit-independent calculations, and to store in a data array-- for each event that is to be used in the fit or for histogramming--only those quantities that are required in the later calculation of his function in HUME.

Any calculation that involves the data but is independent of the parameters $|a|$ of the function $y(|x|, |a|)$ should be done here rather than in the subroutine HUME. This is because DOME is called once or twice per fit while HUME is called many times.

Normally, experimental data is given in packed (ARROW) format [2]. Several functions and subroutines are available in OPTIME to unpack this data [5]. If the input data is in another format the user must furnish his own subroutine GETUM1 (see Section 4.2).

Integration data, from the standard routine GETUM2, is given stored beginning in word ISBUF3 of blank COMMON (see below).

The variable KIND is 1 when an experimental event is at hand and it is 2 when a fake event is at hand.

```
      SUBROUTINE DOME(DATA)
      COMMON/    /
C                     Z(2)
      DIMENSION       IZ(2)
      EQUIVALENCE     (Z,IZ)
      COMMON/HUME/
C                     Y,        ETA(2),   KDER,     IPT,      NW,
C                     MULT,     KIND,     ICASE
      COMMON/IMME/
C                     LEVENT,   KBUFF,    LSTOR,    ITAPE,    NOEVR,
C                     NOEVF,    LWORDL,   NEVF,     NDATA
      COMMON/SAGME/
C                     PLAB,     PCM(4,11), P(8)
      DIMENSION DATA(1)
```

Upon entry, the variables MULT, NW, and the vector DATA are set to zero (0) by OPTIME, while ETA(1) is set to 1.0 for experimental data or to the weight of the integration point for integration data. The variable NDATA specifies the maximum length for the DATA array for both faked and experimental events. It is preset to NDATA = 20 and may be changed with an executable statement prior to calling IMME or INTIME.

When KIND = 1 we have a real event beginning at

Z(LEVENT + 1). If the data is in ARROW format it can be unpacked into the area beginning at Z(1) with the use of the functions

WORD(i) or W (i)
IWORD(i) or IW(I)
} to unpack the desired word i and also store it in Z(i)

or the subroutines

PARTEV(i, j) to unpack words i to j into locations Z(i) to Z(j)

or

ALEVENT to unpack the whole event and store it beginning at Z(1).

When KIND = 2 an integration data point is present. If this event was generated by the standard routine GETUM2, it is stored beginning at Z(ISBUF3) in the following format (ISBUF3 currently is 1):

Z(1)	ICASE	--always 1
Z(2)	PLAB	--the lab beam momentum
Z(3)	ECM	--the overall c.m. energy
Z(4)	ETA(1)	--phase-space weight [3]
Z(5)	ETA(2)	--ETA(1)*frequency distribution function [3]
Z(6)	PCM(1, 1)	4-vector for particle 1
Z(7)	PCM(2, 1)	
Z(8)	PCM(3, 1)	Repeated for maximum of 10 particles
Z(9)	PCM(4, 1)	

let k=6+4*NP — NP = No. of particles

Z(k) EMASS(1) --invariant mass of particle 1

Z(k+1) EMASS(2) --invariant mass of particles 1 and 2

Z(k+NP-2) --invariant mass of the sum of particles 1 through NP-1

The quantities in Z(6) through Z(k-1) are also stored in PCM(i, j) in COMMON/SAGME/.

Make calculations and tests.

Decide and do one or more of the following:

RETURN — with MULT still set at 0. This means this event is not to be used at all, i.e., it is outside the space to be considered in the fit or the histograms.

Set MULT = 1 — means this event falls in the space considered in the fit. It will be used in the fit and in the histograms. Integration points that are generated for histogramming only, by HFAC being >1.0, will be automatically excluded from the set used for the fit even if MULT = 1. When histogramming a fitted curve over real data, only events for which MULT = 1 will be considered in the normalization to the real data.

Set MULT = -1	means this event is outside the space considered in the fit; however, it will be available for histogramming.
Set MULT = 4HQUIT	means the previous event was the last one to be read or generated. This will stop the process before the EOF is reached on the input file.
Set NW = No. of words to be filled in DATA array	i.e., length of vector $\|x\|$ that will be available in HUME when evaluating the function $y(\|x\|, \|a\|)$.
Possibly reset ETA(1) = weight	For KIND = 1, i.e., for real events, this weight had been set by OPTIME to 1.0 before calling DOME and need be reset only if the user wishes it to be different. For KIND = 2, i.e., for integration points, this number is supposed to be the proper weight for the integration points (Ω^{o}, in Ref. [1], Section III. 3). If the standard routine GETUM2 is used it has been set, with a call to WT, in such a way as to reconstitute phase space (see also Section 4.3).
Fill DATA(i), $1 \leq i \leq NW$	with values that will form the vector $\|x\|$, i.e., one data point.

RETURN
END

3.32 Subroutine HISTME

This routine need be supplied only when histograms are to be produced.

HISTME is called for all events for which MULT $\neq$ 0 during pre-fit processing, immediately after DOME has been called; i.e., once per real event and once per generated fake event. The data words to be histogrammed may be chosen either from the vector $|x|$ that was just made in DOME or from other quantities still stored in core. Remember, however, that for types 11 to 14 the fitted curve produced from integration data will be normalized by the subroutine FAME only in the region covered by the fit.
For experimental events the histogram data is written to file KTHISTR, while for integration points the data is written to file KTHISTF. After normalizing the data on KTHISTF by FAME, both will be processed later by the subsystem COCHIS.

```
 SUBROUTINE HISTME (DATA)
 COMMON/     /
C                    Z(2)
 DIMENSION           IZ(2)
 EQUIVALENCE         (Z,IZ)
 COMMON/HUME/
C                    Y,          ETA(2),    KDER,    IPT,    NW,
C                    MULT,       KIND,      ICASE
 DIMENSION  DATA(1)
```

The following variables are available for testing:

If KIND = 1	A real event is ready for use (may be partially unpacked from coding in DOME).
KIND = 2	An integration data point is ready.
MULT = 1	The event is in the space considered in the fit.
MULT = -1	The event is outside the space considered in the fit.

The transfer vector

DATA(i)	contains the OPTIME data point exactly in the form in which it was filled in DOME.

then

CALL HIST(n, word)

to histogram the quantity in "word" in histogram number n. The user may also say CALL H(n, word).

Examples:

a) To histogram the 2nd word of the OPTIME datapoint in histogram number 3:
   CALL H(3, DATA(2)).

b) To histogram word 10 of an experimental ARROW-format event, not yet unpacked, in histogram number 4:
   TEMP = WORD(10)
   CALL H(4, TEMP).

c) To histogram under a certain condition:
   IF(DATA(1). LT. 3.0)
   CALL H (N2, 2.0*DATA(2)).

Note that the weight used for each point is automatically the same that was assigned in DOME.

RETURN
END

3.4 User Routines Called During Fitting

During the fitting phase the user must calculate the value of the so-called user function $y = f(|x|, |a|)$, described already in Section 1.31. Whenever OPTIME needs the value of this function it calls a subroutine HUME with $|x|$ and $|a|$ in the transfer vector. The user must supply this subroutine and store the answer, the value of y, in a COMMON. To save computer time, calculations that do not depend on $|a|$ should not be done here but rather

should be performed in DOME during pre-fit processing, since HUME is called many times while DOME is called only once or twice. The answers of those pre-fit calculations can always be stored in $|x|$.

3.41 Subroutine HUME

```
 SUBROUTINE HUME(PAR, DATA)
 COMMON/HUME/
C                    Y,        ETA(2),    IDER,     IPT,      NW,
C                    MULT,     KIND,      ICASE
 COMMON/HUME1/
C                    LMAXI,    TDER,      TALARME, TEMERG
 LOGICAL                        TDER,      TALARME, TEMERG
 DIMENSION PAR(1), DATA(1)
```

Upon entry, the vector PAR contains the current values of all parameters, fixed and variable, while the vector DATA contains either $|\xi|_\ell$, the specifications of the ℓ^{th} experimental event, or $|x|_\ell$, the coordinates of the ℓ^{th} integration point, just as they were filled in DOME.

The following variables may be tested:

when KIND = 1	An experimental event is at hand, i.e., DATA contains $\|\xi\|_\ell$.
when KIND = 2	an integration point is at hand and DATA contains $\|x\|_\ell$.
when KIND = 3	just like KIND = 2 except that the call comes from FAME.
when KDER = 0	PAR contains the central values of the parameters.
when KDER = -j	PAR contains $\|a\| - DA_j$
when KDER = j	PAR contains $\|a\| - DA_j$

where DA_j is the j^{th} column of the derivative increment matrix (DA).

```
fill Y

RETURN
END
```

The last two cases will occur during numerical calculation of the derivatives. with the value for y(|DATA|, |PAR|).

Note: Considerable amounts of computer time may be saved if the user can supply analytic derivatives (see Sections 7.1 and 7.31).

4.0 OPTIONAL USER-FURNISHED ROUTINES

All optional subroutines are contained in the standard system, most of them in the form of dummy routines. Loading a user-furnished routine of the same name into core ahead of the standard routines will prevent loading of the standard routine.

4.1 Subroutine ALARME

This subroutine is called whenever it might be desirable to check whether any parameter violates some user-defined boundary condition or to give the user a chance to indicate that the function $y(|x|, |a|)$ would make no sense, e.g., if the probability would turn out to be negative. The routine must be supplied by the user if he set NBOUND > 0 in the call to MAXIME. It may be made an entry point in the subroutine HUME.

```
      SUBROUTINE ALARME(PAR, BF)
      COMMON/HUME1/
C                     LMAXI,    TDER,     TALARME,     TEMERG
      LOGICAL                   TDER,     TALARME,     TEMERG
      DIMENSION       PAR(1),   BF(1)
```

Upon entry PAR contains all parameters, fixed and variable, just like in the call to HUME. In addition the variable TALARME = .FALSE.

Calculate boundary conditions
fill BF(i), $1 \leq i \leq$ NBOUND

with values of < 0.0 when the i^{th} boundary condition was violated or with values ≥ 0.0 otherwise.
The variable TALARME should be left .FALSE. as long as the user function $y(|x|, |a|)$ makes sense with the given parameters. This will allow calculating the the function during derivative

calculations even if some given parameter(s) a $\pm$ DA_j are just outside the user's boundary. The central values of the variable parameters, |a|, are not allowed outside the user's boundary regardless of the setting of TALARME.

```
RETURN
END
```

4.2 Subroutine GETUM1

The standard OPTIME system contains a subroutine GETUM1 that is called from the subroutine IMME to read, during pre-fit processing, input records in ARROW format [2], and point to one event at a time. If the user's experimental data are in a different format he must supply his own subroutine GETUM1. The standard routine employs so-called switch buffering and reads data into an area beginning at blank COMMON(LEVENT+1) and of length KBUFF. LEVENT and KBUFF in COMMON/IMME/ are usually set to 500 and 1027 respectively.

The call to GETUM1 once per every experimental event from IMME is set up as follows

```
COMMON/ / Z(1)
  .
  .
  .
CALL GETUM1(Z(LEVENT+1), LWORD)
IF(LWORD.EQ.0) quit reading any further
```

As can be seen, the argument LWORD must be set to 0 by GETUM1 after the last event has been processed.

4.3 Non-standard Integration Points, Subroutine GETUM2

If the integration points do not represent particle interactions the user

must provide his own routine GETUM2 to make them. The standard OPTIME system contains a subroutine GETUM2 that generates integration points representing particle final states by Monte Carlo methods. These integration points are generated flat in masses and angles. If weighted with the weight returned in the variable ETA(1), those events represent phase space.

In cases where the function y is highly structured to fit the real distribution, the integration can often be made more accurate, or fewer integration points may be required, when the points are generated according to sophisticated distributions. The Monte Carlo event generator SAGE [3], which is incorporated in OPTIME, allows for generation in three ways:

1) According to Breit-Wigner distributions of the type $1/\{[(\mu-E)^2/(\Gamma/2)]^2+1\}$, where E is the energy of two or more particles in their c.m. system, μ the mass and Γ the width of a resonance.

2) According to an expression $e^{q\Delta^2}$, where Δ^2 is a momentum transfer squared and q a parameter.

3) According to a flat distribution.

As a rule of thumb one often recommends a distribution of integration points that would approximate the real distribution. To take advantage of these features, the user must also supply his own subroutine GETUM2, using the routines provided by the SAGE system. If more than one distribution is to be generated the user must have set in the main program the variable NCASE to the number of distributions, and filled NTOTF(i), $1 \leq i \leq$ NCASE, with the number of points to be generated for each case. In addition the subroutine MIME is required (see Section 4.4).

For example, suppose a user would like to generate, to be used in the fit, 1000 points for case 1, 2000 points for case 2, and 1000 points for case 3. In order to have a smooth fitted curve in his histograms, he would like to use there three times the number of points. He would have to set the following variables in the main program:

```
DATA NTOTF/1000, 2000, 1000/
HFAC = 3.0
NCASE = 3
```

Then the total number of points that will be generated is

$$\text{NUMBER} = \sum_{i=1}^{\text{NCASE}} \text{NTOTF(i)*HFAC} = 12\,000$$

of which 8000 will be used for histogramming only. GETUM2 will be called for all points of one case before the next case is begun. OPTIME takes care of separating the points for the histogram only from those for fit and histogram.

Below is shown the general arrangement of a routine GETUM2 to generate special or more than one distribution, using SAGE [3].

The first argument in the calling sequence to GETUM2, BUF, begins an area in blank COMMON of length NEVF that may be filled by the user. NEVF is set by OPTIME with a data declaration to 55. BUF(1) is equivalent to blank COMMON(ISBUF3) where ISBUF3 is currently set to 1.

```
      SUBROUTINE GETUM2(BUF,LW)
      COMMON/       /
C                   Z(2)
      DIMENSION     IZ(2)
      EQUIVALENCE   (Z,IZ)
      COMMON/HUME/
C                   Y,        ETA(2),   KDER,     IPT,      NW,
C                   MULT,     KIND,     ICASE
      COMMON/IMME/
C                   LEVENT,   KBUFF,    LSTOR,    ITAPE,    NOEVR,
C                   NOEVF,    LWORDL,   NEVF,     NDATA
      COMMON/RAME/
C                   ISD,      ISBUF1,   ISBUF2,   ISBUF3
      COMMON/RENO/
C                   NP,       ECM,      AMASS(10)
      COMMON/SAGME/
C                   PLAB,     PCM(4,11), P(8)
      COMMON/VEGAS
C                   BEAM,     TARGET,   CRAP(6)
      COMMON/INIT/  BM(4),    TG(4)
      DIMENSION     BUF(1)
```

The variable BUF in the transfer vector and (ISBUF3) point to the same core location, namely word ISBUF3 of blank COMMON, currently set to 1. The variable ICASE in COMMON/HUME/ ranges from 1 to NCASE and upon entry is set by OPTIME to the number of the distribution to be generated. The variable IPT is initially 0 and is incremented by INTIME each time after an event is generated.

```
GO TO (10, 20, ···, n), ICASE
10 CALL INITL(...)
```

to initialize SAGE, only required before first event is generated.

```
CALL RY(...)
```

to input quantities necessary to generate a two-body decay vertex.

```
CALL GO(...)
```

returns the generated quantities.

```
GO TO 90
20 CALL INITL(...)
```

this may be necessary when generating another distribution.

```
   CALL RY(...)
90 CALL WT(ETA)
```

this call fills ETA(1) with the "phase space" weight [3], ETA(2) with ETA(1)*"phase space" frequency distribution; it also signals to SAGE the end of an event [3].

if desired fill BUF(i), ISBUF3≤ i≤ NEVF

with quantities as described in

one may set LW = 0

Section 3.31.
This is not necessary but would make core look as if the standard routine GETUM2 had been used.
this will terminate calls to GETUM2 from OPTIME prior to generating

$$\sum_{ICASE=1}^{NCASE} NTOTF(ICASE)*HFAC$$

events; only IPT events will have been generated.

```
RETURN
END
```

If a routine GETUM2(BUF, LW) is supplied by the user to read integration points from a file KTFAKE rather than generating them, then the reading should be done into BUF(i) starting at location BUF(ISBUF3). In that case the user must set the variable NEVF in COMMON/IMME/ with an executable statement to the maximum length of his logical records. Also, LW must be set to zero when the end of file is encountered.

4.4 Subroutine MIME

This routine must be provided by the user if he chose to generate more than one distribution, i.e., if NCASE > 1, in his subroutine GETUM2. The purpose of this routine is to calculate and return the values of the NCASE generating functions. The form of these functions must be the same as those used by SAGE [3].

```
SUBROUTINE MIME(GF, DATA)
COMMON/HUME/Y, ETA(2), KDER, IPT, NW, MULT, KIND, ICASE
COMMON/RENO/NP, ECM, AMASS(10)
DIMENSION GF(1), DATA(1)
```

The routine is called following DOME and upon entry DATA

contains the integration point as filled in DOME.

calculate the values of the NCASE generating functions.

fill GF(i), $1 \leq i \leq$ NCASE — with the values of these generating functions.

RETURN
END

4.5 Subroutine FLAME

This routine allows the user to provide so-called reference functions in order to make the integrations more exact (see Ref. [1], Section III. 3). Reference functions are functions for which the integrals are known. For a given accuracy of the integration this procedure may allow use of fewer integration points, resulting in a saving of computer time.

SUBROUTINE FLAME(FR, DATA)
COMMON/FLAME/NREF, IFLAME
DIMENSION FR(1), DATA(1)

This routine is called several times during the process of preparing integration points for OPTIME.
Upon entry NREF is equal to the number of reference functions that the user will provide, as specified in the call to INTIME(NREF), and the variable IFLAME is either greater or less than zero.
DATA contains the integration point as it was filled in DOME.

when IFLAME > 0 — fill FL(i), $1 \leq i \leq$ NREF, with the value(s) of the reference function(s).

when IFLAME < 0 fill FL(i) with the integral(s) of the reference function(s).

RETURN
END

4.6 Program Constants, Subroutine SHAME

The routine SHAME is called by MAXIME just prior to starting a fit. The standard routine in the OPTIME system is just a dummy routine. The user may, however, supply a routine SHAME of his own which he could program, for example, to reset some of OPTIME's program constants instead of resetting them in his main program. Usually, however, the user need not reset these constants and they are listed below only for the purpose of giving some explanation.

CHILIM = 0.01 This constant is used when deciding if a fit is good enough. There are NPAR numbers, called "derivative norm squared" printed under the heading "PARTIAL DNS." If $(\text{PARTIAL DNS})_j <$ CHILIM the parameter a_j is called adjusted. One condition for the fit to stop, i.e., for its being good enough, is that all parameters must be adjusted (for an explanation of DNS see Ref. [1], Section II. 7).

RLIM = 1.0E-10 This constant is used in the subroutine DIAME to determine when a parameter is called correlated.

ELIM = 1.0E-200 This constant is also used in the subroutine DIAME. If a diagonal element of the matrix

E, $E_{j,j}$ is less than ELIM, the parameter j is called insensitive. This corresponds to the fact that the j^{th} eigenvalue of E does not exist (See Ref. [1], Section II. 8).

TMAX = 0.9
TMIN = 0.1

In calculations involving corrections to the stepsize, where $|v|$ is the proposed step and $t|v|$ is the corrected step, t is set such that TMIN $\leq t \leq$ TMAX, if it turned out to be outside of those limits.

TDIAG = .FALSE.
TCONST = .FALSE.

These two constants define alternative procedures for calculating the derivative increment matrix (DA) in the subroutine DAME. Normally (DA) is calculated according to (DA) = WT*WGT*(R); see [1]. When TDIAG = .T. and TCONST = .F. DA is a diagonal matrix ~~(DA)~~. From one step to another ~~(DA)~~ is modified according to $(\cancel{DA})_{new} = (\cancel{DA})_{old}$ *WT*WGT*(EDIAG)$^{-1}$, where (EDIAG) is the matrix consisting of only the diagonal elements of the matrix E^*. When in addition to TDIAG also TCONST = .T. then (DA) remains constant and diagonal, ~~(DA)~~ = |UDA|.

TPRINT = .TRUE. — If the user sets TPRINT = .FALSE. before calling MAXIME all printing during the stepping process will be suppressed.

```
      SUBROUTINE SHAME (PARME)
      COMMON/GAME/
     C                KTYPE,    KEND,     TPRINT
      LOGICAL          TPRINT
      COMMON/NUNAME/
     C                NMAT,     NBUFF,    DACUTM,  T2ND,    TCONST,
     C                WI,       WR
      LOGICAL          T2ND,     TCONST
      COMMON/PRIME/
     C                CHILIM,   RLIM,     ELIM,    TMAX,    TMIN,
     C                TCORME,   TDIAG,    TDIRCO,  WT
      LOGICAL          TDIAG,    TDIRCO,   TCORME
      Reset some constants
      RETURN
      END
```

4.7 Subroutine VARME

This subroutine is called by the subroutine DECIME once per attempted step immediately after returning from the subroutine COMME; see Section 8.1. It is a dummy routine in the standard OPTIME system but may, for example, be supplied for conveniently printing some quantities when debugging a program.

5.0 USER CALLS TO OPTIONAL ROUTINES

5.1 Initializing Necessary Program Constants

Some user-callable post-fit sub routines require that particular program constants be initialized. This is automatically accomplished when MAXIME is called. When the user wishes to calculate the integral $\int y(x, a)\, dx$ before he begins the fit, by calling the subroutines NORME or ENORME, or when he wishes to make histograms before the fit, by calling the subroutines FAME and COCHIS, he must first, at least once, call the subroutine PREPME. This routine is actually an entry point within the subroutine MAXIME and has the same calling sequence (see Section 3.2). PREPME will not fit anything but only initialize constants to allow execution of the post-fit processes.

Call PREPME(PARME, LOC, WGT, UDA, MSTEP, NBOUND, NPAR, MAXPAR)

5.2 Calculating Integrals

As already mentioned in Section 3.2 the user may call the subroutine NORME(RES, PAR, COM) which will return in RES the integral of y(x, a) in the space of x. The integral is actually calculated by summing over the integration points used in the fit:

$$Y(a) = \int y(|x|, |a|)\, dx = \sum_{\ell=1}^{NINT} \Omega_\ell\, y(|x|_\ell, |a|)$$

where $|x|_\ell$ is the vector specifying the ℓ^{th} integration point with Ω_ℓ being its weight (see also [1], Section III. 2), and $|a|$ is the vector containing the variable parameters stored in PAR, specified by the vector $|LOC|$. The subroutine NORME expects to find integration points stored on the file KTINT and it makes NINT calls to HUME with KIND = 2 to obtain the value of the user function.

Calls may be made with the values in PAR or the vector $|LOC|$ changed to calculate different integrals.

By calling the subroutine ENORME(RES, PAR, COM) the user may also calculate the derivatives of the integral with respect to $|a|$ in addition to the integral. Again the computation will be performed as a summation over the NINT integration points:

$$\partial\, Y(a) / \partial\, a = \sum_{l=1}^{NINT} \partial \left[\int y(|x|_l, |a|)\, dx \right] / \partial\, a.$$

RES(1) will contain the integral while RES(j), $2 \leq j \leq$ NPAR + 1, will contain the derivatives. Obviously RES must have been dimensioned NPAR + 1. If COM = 0 nothing will be printed. If COM is filled with a comment of up to 10 Hollerith characters this comment and all answers will be printed, e.g., CALL NORME (RES, PAR, 10HFIRST CALL).

5.3 Calculating the Error Matrix

To calculate and print the matrix (ERR) for the last accepted parameters $|a_o|$, fitted in the previous call to MAXIME, the user may call the subroutine AROME. This subroutine should be called immediately after MAXIME since it uses some quantities calculated by MAXIME in its last step which otherwise might get overwritten in core. AROME will calculate the matrix (ERR) only if all variable parameters $|a_o|$ have either been declared adjusted or have been fitted against the user-defined boundary, i.e., if KEND = 1 or 10. For all types of fit the matrix is calculated by performing a summation over the NEV experimental data points. The answer is an approximation of the error matrix and is defined as

$$(ERR) \approx \left\langle \left(|\alpha| - |A| \right) \overbrace{\left(|\alpha| - |A| \right)} \right\rangle$$

where $\langle\ \rangle$ is the expectation value, $|A|$ is the true value of the parameters $|a|$, and $|\alpha|$ is the fitted value of the parameters $|a|$, i.e., $|a_o|$, such that $y(|x|, |\alpha|)$ approximates the true distribution $y(|x|, |A|)$. The diagonal elements of (ERR) are the errors squared while the off-diagonal elements are the correlations (see also [1], Section III. 1). These values are printed and they are also stored sequentially row by row beginning in word LERR of blank COMMON. If we consider

```
COMMON Z(1)
COMMON/STORME/XX(40), LERR
```

we find the element in row i, column j in

```
ERR(i, j) = Z(LERR - 1 + j + NPAR*(i-1)).
```

As mentioned above, for the calculation of (ERR), the values $|a_o|$ of

the variable parameters are used. This vector is stored in blank COMMON, beginning at Z(LA0). When all parameters have been declared adjusted the difference between $|a_o|$ stored in Z(LA0) and $|a|$ returned in |PARME| should not matter for all practical purposes.

6.0 END OF FIT AND MESSAGES

In this section the conditions under which a fit may stop and the various messages that OPTIME prints are described. Some messages are to inform the user of some non-fatal condition, others may imply some grave error.

6.1 The Variable KEND, End of Fit

```
COMMON/GAME/KTYPE, KEND, TPRINT
LOGICAL TPRINT
```

The variable KEND is set by OPTIME to signify the status of the fit. It may be tested in the main program after returning from MAXIME and has the following meaning:

KEND	
0	fit still in progress.
1	successful fit; central values of all variable parameters are adjusted inside the physical region.
2	central values of the initial parameters are outside the user-defined boundary.
3	unable to calculate derivatives; derivative increment additions, $(\|a\| \pm (DA))$, for all parameters causes TALARME be set . TRUE. by the user during boundary tests.
4	serious problem; in MODE = 2 when the parameters $\|a\|$ are replaced by the best ones so far, $\|b\|$, the function calculated now, w(a), does not correspond to the value previously calculated for the same values of the parameters, w(b). Usually caused by a user error in HUME.
5	maximum numbers of proposed steps has been reached.
6	for max. log. fit (KTYPE < 20), some experimental data-point(s) causes the function y $(\|\xi\|, \|a\|)$ for the central values of the parameters $\|a\|$ to become zero or negative even though $\|a\|$ is within the boundary and the step cannot be reduced further.
7	read or write error from the OPTIME I/O routine usually causes ABORT with a comment from the routine ERME.

8	E matrix is not useable (singular or negative); all parameters are called correlated or insensitive; usually a user error.
9	when T2ND = . TRUE. two calculations of the function w(a) for the same parameters do not agree.
10	central values of parameters at the boundary, new parameters outside, corrected step is too small to be worthwhile.
11	pathological case when trying to correct step near the boundary.

6.2 Messages

Most messages that OPTIME prints are generated by the subroutine ERME and are printed with trace-back information. ERME also makes a dayfile message.

MESSAGE	ABORT	STOP	MEANING
Error in fit-type		Yes	KTYPE was not set to 3, 4, 11, 12, 13, 14, 20, or 21.
Inconsistent No. of real events			The number of real events read during fitting is different from NEV, determined by IMME.
Inconsistent No. of Monte Carlo events			The number of integration points read during fitting is different from NINT, determined by INTIME.
Inconsistent No. of histogram events			When FAME normalizes the fake histogram data, the number of events read differs from that specified earlier.
Y zero or negative	Yes		See KEND = 6
Inconsistent calculation of function	Yes		See KEND = 9
Read error	Yes		Caused when a file is read that is empty when it should not be, or when data is lost due to read error.

MESSAGES	ABORT	STOP	MEANING
Write error (histogram)	Yes		When writing histogram data either data are lost, or the end of the output file has been reached, or no data were written to a particular file, e.g., when no events are accepted for the fit in DOME.
Write error	Yes		Same as previous comment except when writing data other than histogram data.
Apparent user error			The user set the max. number of parameters (MAXPAR) lower than some absolute value(s) in the vector LOC.
Error in initialization		Yes	In the call to MAXIME the user probably wrongly specified the number of boundary functions, or the number of variable parameters, or the max. number of parameters.
Need subroutine MIME		Yes	The user set NCASE > 1 when using the SAGE Monte Carlo generator, via a call to INTIME, but did not supply the subroutine MIME.
Need subroutine ALARME		Yes	The user set NBOUND > 0 in the call to MAXIME but did not supply the subroutine ALARME.
Initial parameters bad		Yes	See KEND = 2.
Normalization impossible, WI indefinite		Yes	for KTYPE = 11, 12, 13, or 14 the integral WI is indefinite when FAME was called.
Generating less than 3 particles		Yes	The standard routine GETUM2 was called with NP < 3.

Other self-explanatory messages may be issued by the subroutines BUFOUT, BUFIN, CORME, and NORME.

7.0 REDUCING COMPUTER TIME

Once the programming of a particular problem of fitting with OPTIME has been accomplished, an additional goal becomes reduction of the time it takes to execute this program. Obviously one should take advantage of techniques that are already available in OPTIME as standard features, in particular, providing reference functions (see Section 4.5) can be helpful. Minimizing of the calculations in HUME is also very important. In this section some other standard and nonstandard features that may help reduce computer time are discussed.

7.1 Furnishing Derivatives for Linear Parameters

For any parameter |a| that enters linearly in the user function y (|x|, |a|) in HUME, derivatives may be programmed in HUME with the aid of the subroutine DIME. The procedure is as follows:

a) in the main program:

if PARME (n) enters linearly in the HUME function set UDA(j) = 0.0 or set WGT(j) = 0.0

where j is such that LOC(j) = n

b) in the subroutine HUME:

add DIMENSION LINP(NLIN), DPAR(NLIN)

where NLIN = number of programmed derivatives

then fill DATA LINP/k_1, k_2, $\cdots$, k_n/

where the k's are integers similar to LOC, giving the ordinal numbers of the linear parameters in the PARME vector in the order in which DPAR will be filled.

then fill DPAR(j), $1 \leq j \leq$ NLIN, with the coefficients of the linear parameters

then CALL DIME (DPAR, LINP, NLIN)

DIME performs the following computations:

$$Y = 0.0$$

$$Y = \sum_{j=1}^{NLIN} PAR(LINP(j)) * DPAR(j)$$

and the derivatives

H(i) = DPAR(j)

and stores all quantities in their proper locations.

If not all derivatives are provided by the user he can then finish the calculation of his function by

$$Y = Y + \cdots .$$

Providing derivatives for non-linear parameters is described in Section 7.31.

7.2 Eliminating Pre-fit Processing

The pre-fit processing phase of OPTIME involves the processing of experimental events and most often also the generation of integration data and the preparation of histograms with or without fitted curves superposed.

If a user wants to repeat a fit many times without changing the way data points are made in the subroutine DOME and without changing the quantities to be histogrammed or the number of histograms to be made, then there is no need to repeat the pre-fit work. Files with data that were written to the disc during the initial run can be saved and made available to MAXIME in subsequent runs.

Let us assume that a user wants to repeat a fit several times, making changes only in his routine HUME or associated routines, and that his filenames are: KTEV = 1, KTINT = 2, KTHISTR = 20, and KTHISTF = 29. He uses integration points and makes histograms with fitted curves superposed.

The following procedure may then be used:

The main program must contain

```
COMMON/NORME/NEV, NINT, WTINT, WTRAME, WTHIST
COMMON/HISTME/XXX(3), IEVTH
```

a) in the initial run:

```
      .
      .
CALL IMME
CALL INTIME(N)
PRINT1, NEV, NINT, IEVTH, WTEV, WTINT, WTRAME, WTHIST
1 FORMAT(10X, 3I6, 4O20)
      .
      .
```

These printed quantities are necessary for subsequent runs. In the control cards, at the end of execution, request one or more magnetic tapes and copy the four disc files TAPE1, TAPE2, TAPE20, and

TAPE29 to magnetic tape to be available for subsequent runs.

b) in subsequent runs:
In the control cards, before exection, request the magnetic tape that was saved and create on disc the four files TAPE1, TAPE2, TAPE20, and TAPE29.
Do not call IMME or INTIME(N).
Fill the COMMON variables NEV, NINT, IEVTH, WTEV, WTINT, WTRAME, and WTHIST with the values printed in the initial run.

```
      .
      .
      .
CALL MAXIME(...)
      .
      .
      .
CALL FAME(PARME)
CALL COCHIS
      .
      .
```

7.3 Other Techniques

In this section we discuss techniques that need not but can be used in OPTIME, requiring a minimum of effort on the part of the user. Some of them may reduce the amount of computer time, or increase the accuracy, or may be useful for other reasons.

7.31 Providing Derivatives for Non-linear Parameters

If some derivatives can be computed analytically in less time than required by the standard numerical computation of MAXIME, the user may save time by providing those derivatives. Remember, however, that by complicating the HUME routine additional errors may be introduced and more computer time may be used than by taking a longer but sometimes safer route.

If in the user's judgement it is better to provide derivatives the following procedure may be used:

a) In the main program, before calling MAXIME:

.
.
.

Set WGT(i) = 0.0

for those parameters for which derivatives are going to be provided. The WGT are in the order of LOC; e.g., if the derivative for parameter number n, in PARME, is provided and if LOC(k) = n then set WGT(k) = 0.0.

b) in the subroutine HUME add the following:

```
COMMON // Z(1)
DIMENSION IZ(1)
EQUIVALENCE (Z,IZ)
COMMON/HUME1/LMAXI, TDER, TALARME, TEMERG
LOGICAL                TDER, TALARME, TEMERG
COMMON/STORME/XX(9), LLOCM1, XXX(6), LH, XXXX(24)
```

.
.
.

Calculate the function y as before

```
IF(.NOT. TDER)RETURN
```

.
.
.

Supply derivatives in addition to the function when TDER = .TRUE. and store them as follows:

```
K = IZ(LLOCM1 + N - 1)
Z(LH + K - 1) = derivative N
                in PARME
```

MAXIME has filled LOCM1 with the inverse of LOC, therefore K is such that N = LOC(K). Using the pointer K will order the derivatives properly like LOC.

.
.
.

```
RETURN
```

7.32 Providing the integral $\int f(|x|)\,dx$ for KTYPE 3 or 13

All fits except KTYPE = 11 involve expressions with integrals over

the space of $|x|$. Instead of letting OPTIME calculate these integrals numerically during each fit with many integration points, the user may provide the integral to OPTIME when KTYPE = 3 or 13. This will probably save computer time if the user can make the necessary calculations faster and with equal or better accuracy than OPTIME does them numerically.

The current experimental procedure for providing the integral is to generate one fake event for the fit. This event serves only the purpose of later transmitting the integral to MAXIME from the subroutine HUME. The data from GETUM2 can be ignored but it is important that the first fake event that is generated be not rejected in the subroutine DOME.

Assuming the user wants a fitted curve superposed in his histograms, fake events for the histograms may be generated by setting HFAC > 1.0. It is probably best not to include in the histograms the first event that is generated. This can be accomplished by not calling HIST from the subroutine HISTME when KIND = 2 and IPT = 1 (both variables are in COMMON/HUME/). The following procedure is suggested:

a) in the main program:

```
set NTOTF = 1                    . This would generate 2000
HFAC = 2000.0                      events for the fitted curve.
    .
    .
    .
CALL IMME                        As usual
CALL INTIME(1)
```

b) in DOME, when KIND = 2:
for the first fake event (when IPT = 1)
store in the DATA array the parameter-independent quantities that will be needed for the integral.
Set NW to the appropriate number of words
set MULT = 1
for subsequent fake events make the data array and set MULT and NW as usual.

c) in HISTME: just return when KIND = 2 and IPT = 1, call HIST as usual for all other events.

d) in HUME, when derivatives are not provided:
when HUME is called during the fit with KIND = 2, complete the calculation of the integral $\int f(x)\,dx$ and set

$$Y = \int f(x)\,dx.$$

After the fit the user probably calls FAME to normalize the fake histogram events. FAME will call HUME once for each fake event that was prepared for histogramming with the variable KIND = 3. When KIND = 3 the user must return the function in Y and not the integral.

e) in HUME, when also some derivatives are provided;
If the user chooses to provide some or all derivatives of the function in addition to the integral he must also provide the corresponding derivatives of the integral. Then the following procedure should be used in HUME:
When KIND = 2 or KIND = 3
and when TDER = .F.
the procedure is as described under d).
When KIND = 1 and TDER = .T.
the user stores the derivatives of the function by using the same procedures as described in Sections 7.1 or 7.31.
When KIND = 2 and TDER = .T.
in addition to providing the integral in Y the user also must store the derivatives of the integral in the same locations as for real data for those parameters for which he chose to provide derivatives, namely:
K=IZ(LLOCM1 + N - 1)
$Z(LH + K - 1) = \int h_N(x, a)\,dx$
where $h_N(x, a)$ is the drivative for the N^{th} parameter in PARME (see also Section 7.31).

7.33 Plotting a fitted curve when KTYPE = 11

When a user makes a fit of KTYPE = 11 he usually does not call the subroutine INTIME because integration points are not needed for the fit. If, however, he wants to plot a curve that corresponds to some theory, superposed on the histograms, he must call INTIME, because integration points are needed for making this curve. In general, they are needed for the

post-fit processes performed by NORME, FAME, and COCHIS. The procedure should then be just like for any other fit, i.e., the user would set variables in COMMON to signal to COCHIS that a curve is to be superposed and in HISTME call HIST for fake data as well as for real data (see Section 3.32).

The post-fit routine FAME not only needs integration points but also the value of the integral WI = $\int f(x, a)\, dx$ to perform the normalization of the curve to the experimental data. Therefore, before calling FAME from the main program, the user must store the integral in WI in COMMON/NUNAME/XX(5), WI, XXX. If the user does not know the value of that integral he can call NORME to calculate the integral numerically, e.g., CALL NORME(WI, PAR, 0).

7.34 Plotting fake data only without fitting

In order to understand the influence of the parameters on the theoretical distribution it is often desirable to just plot fake data weighted by a function y ($|x|$, $|a|$) of HUME for values of $|a|$ that are not the fitted values. For this purpose the following procedure may be used:

a) In the main program don't call IMME or MAXIME,
set KTYPE = 3
set KTSCR = 20
set parameters for the generation of fake data
.
.
.
CALL INTIME(1) — will call GETUM2, DOME, and HISTME with KIND = 2
.
.
.
CALL PREPME(...) — (see Section 5.1)
fill PARME(i) with desired values
CALL FAME(PARME) — will call HUME with KIND = 3.
CALL COCHIS

b) In the COMMON/H4/ (see Section 3.2). — set IH(i) = 1 or 4. This will signal later to COCHIS that no curve is to be superposed.

PREPME is the routine that allows use of post-fit routines like NORME, FAME, and COCHIS without actually doing a fit.

7.35 Plotting an analytic curve over the histograms

It is possible for the user to provide his own curve for superposition over the experimental data instead of letting OPTIME construct it from Monte Carlo data points.

The procedure that is currently available requires that the user set variables in COMMON so that COCHIS, later, assumes that a curve is to be constructed (see Section 3.2). However, no fake data is necessary for histogramming and no calls to HIST for fake data need be made. Also the subroutine FAME must not be called.

The user replaces the function UFUN, which normally calculates the ordinates of the fitted curve, with a function of his own. The function UFUN will be called from COCHIS once per bin per histogram and looks as follows:

```
FUNCTION UFUN(X, N)
```

where X = abscissa of the center of the bin, N = histogram number.

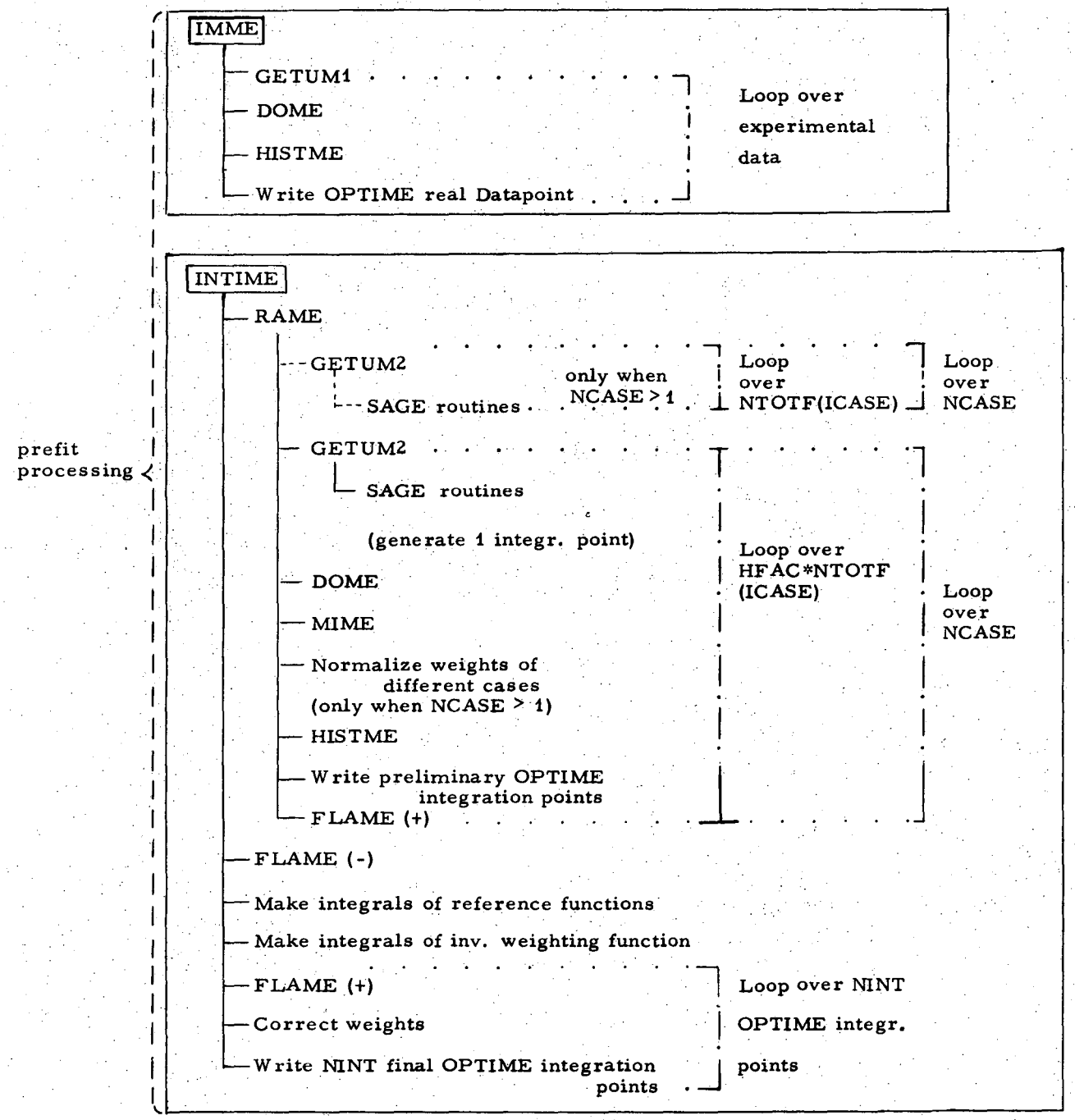
```
Calculate ordinate of curve
at abscissa X, for histogram N
and set
UFUN = that value

RETURN
END
```

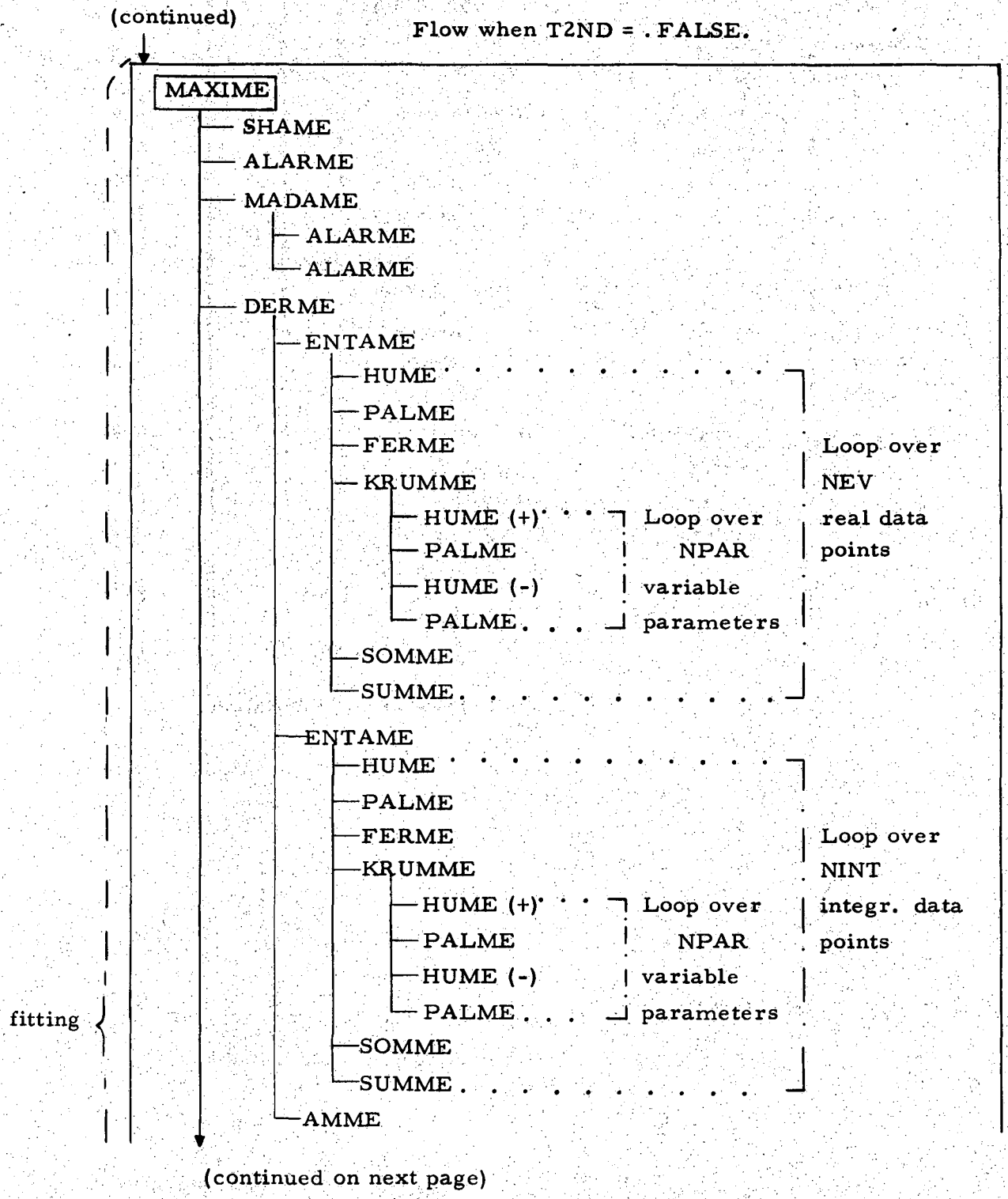
8.0 TECHNICAL INFORMATION

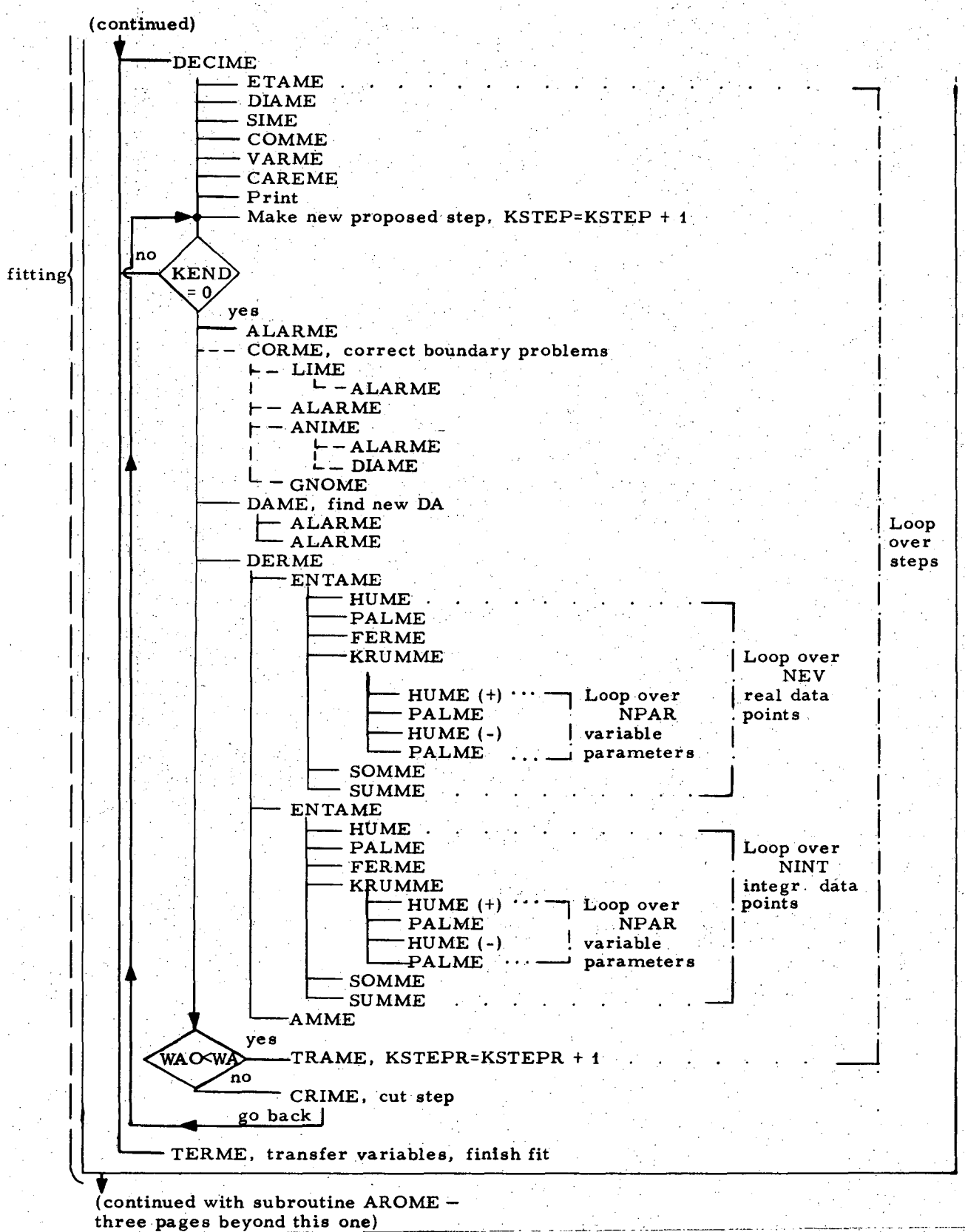
The information in this section deals mainly with some technical details of the OPTIME system. It may be useful when changes of the standard version are necessary, when debugging, or when just trying to understand the program better.

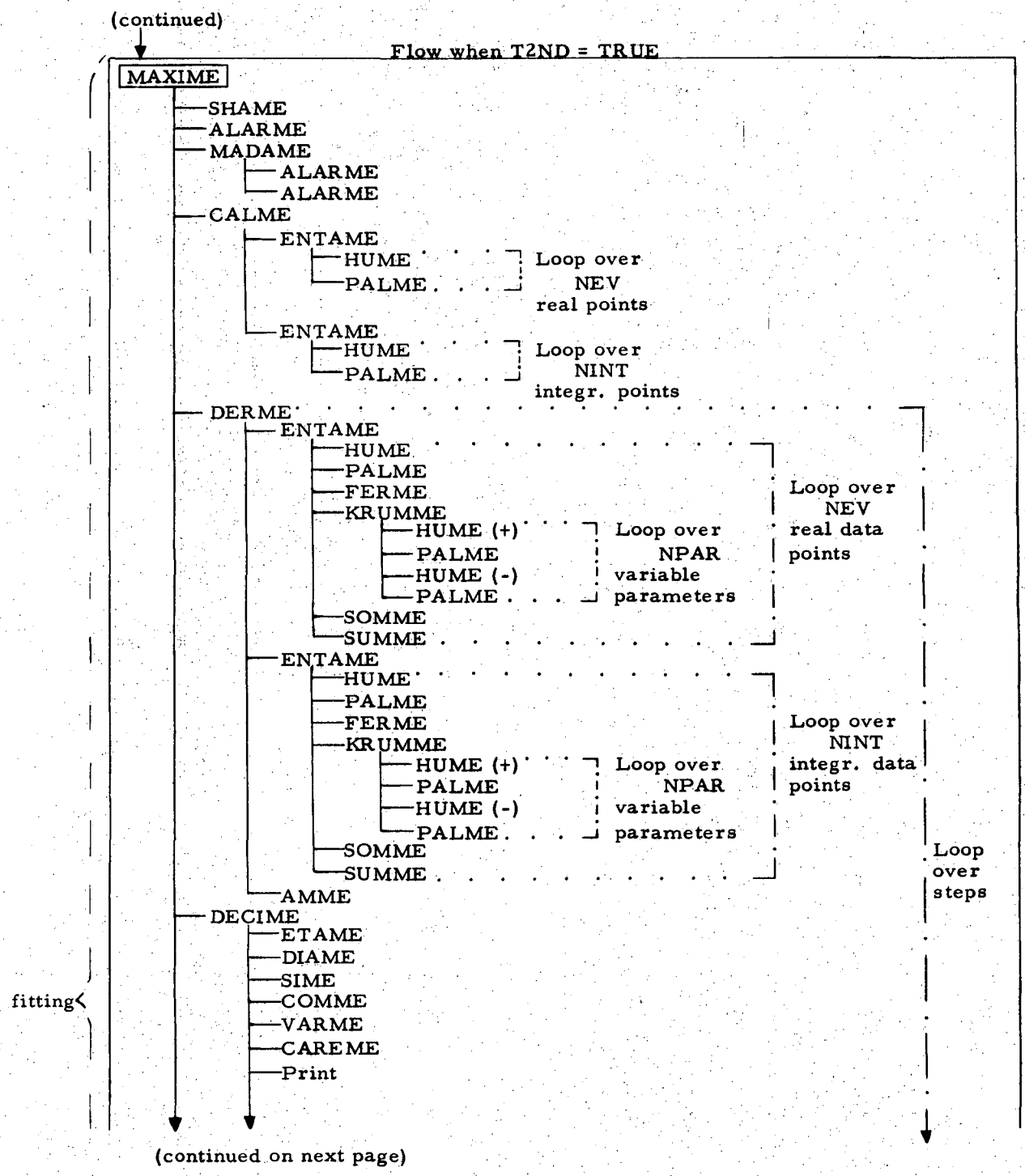
8.1 General Flow Diagram

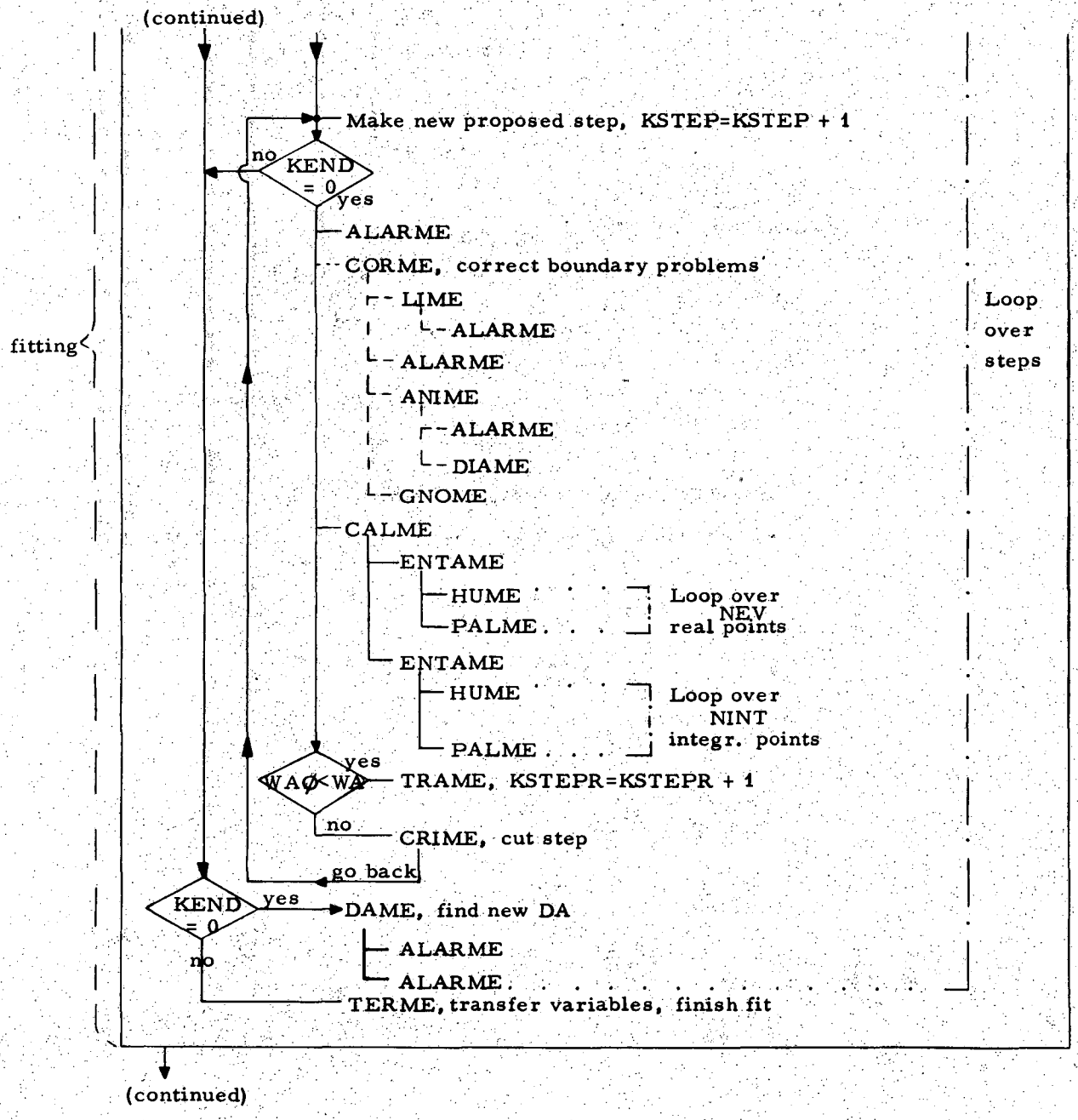
MAIN PROGRAM

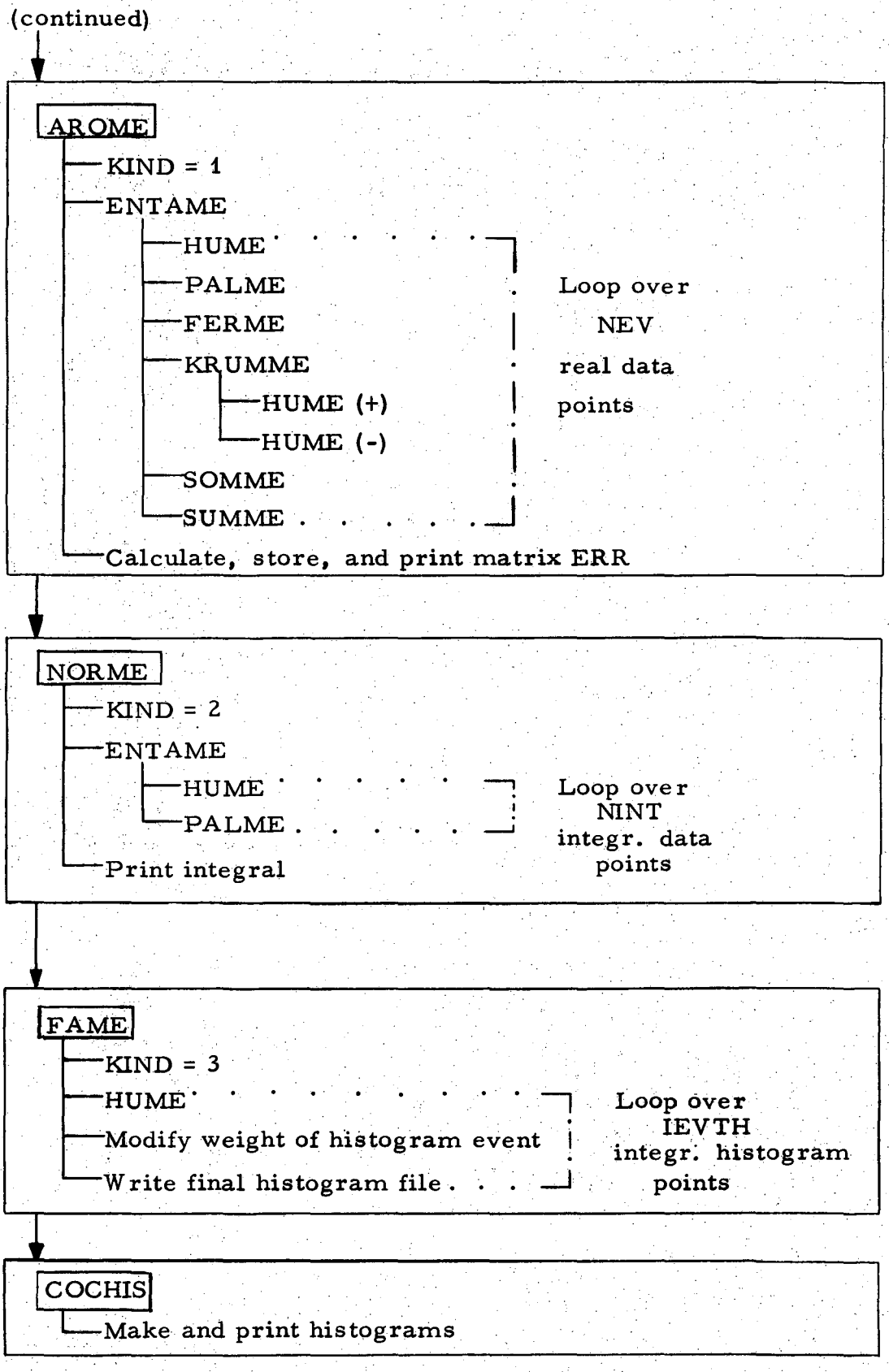



(continued)
Flow when T2ND = .FALSE.
MAXIME
SHAME
ALARME
MADAME
ALARME
ALARME
DERME
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NEV real data points
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NINT integr. data points
fitting
AMME
(continued on next page)


(continued)
DECIME
ETAME
DIAME
SIME
COMME
VARME
CAREME
Print
Make new proposed step, KSTEP=KSTEP + 1
fitting
no
KEND = 0
yes
ALARME
CORME, correct boundary problems
LIME
ALARME
ALARME
ANIME
ALARME
DIAME
GNOME
DAME, find new DA
ALARME
ALARME
DERME
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NEV real data points
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NINT integr. data points
AMME
yes
WAO<WA
TRAME, KSTEPR=KSTEPR + 1
no
CRIME, cut step
go back
Loop over steps
TERME, transfer variables, finish fit
(continued with subroutine AROME — three pages beyond this one)


(continued)
Flow when T2ND = TRUE
MAXIME
SHAME
ALARME
MADAME
ALARME
ALARME
CALME
ENTAME
HUME
PALME
Loop over NEV real points
ENTAME
HUME
PALME
Loop over NINT integr. points
DERME
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NEV real data points
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
PALME
HUME (-)
PALME
Loop over NPAR variable parameters
SOMME
SUMME
Loop over NINT integr. data points
AMME
Loop over steps
DECIME
ETAME
DIAME
SIME
COMME
VARME
CAREME
Print
fitting
(continued on next page)


(continued)
fitting
Make new proposed step, KSTEP=KSTEP + 1
no
KEND = 0
yes
ALARME
CORME, correct boundary problems
LIME
ALARME
ALARME
ANIME
ALARME
DIAME
GNOME
CALME
ENTAME
HUME
PALME
Loop over NEV real points
ENTAME
HUME
PALME
Loop over NINT integr. points
yes
WA∅ < WA
TRAME, KSTEPR=KSTEPR + 1
no
CRIME, cut step
go back
KEND = 0
yes
DAME, find new DA
ALARME
ALARME
no
TERME, transfer variables, finish fit
Loop over steps
(continued)


(continued)
AROME
KIND = 1
ENTAME
HUME
PALME
FERME
KRUMME
HUME (+)
HUME (-)
SOMME
SUMME
Loop over NEV real data points
Calculate, store, and print matrix ERR
NORME
KIND = 2
ENTAME
HUME
PALME
Loop over NINT integr. data points
Print integral
FAME
KIND = 3
HUME
Modify weight of histogram event
Write final histogram file
Loop over IEVTH integr. histogram points
COCHIS
Make and print histograms

8.2 COMMON Blocs

```
 COMMON //
C                Z(1)
 DIMENSION       IZ(1),      PAR(1),     LOC(1),     A0(1),      DA(1),
D                TDP(1),     TDM(1),     TWP(1),     TWM(1),     ESTAR(1),
D                U(1),       UI(1),      WP(1),      WM(1),      WPI(1),
D                WMI(1),     H(1),       BUFF(1),    V0(1),      BOUND(1),
D                R(1),       RST(1),     B(1),       A(1),       SIM(1),
D                V0ST(1),    V(1),       VSTST(1),   V02ST(1),   V12ST(1),
D                V2ST(1),    V3(1),      EDIAG(1),   KDIA(1),    PDNS(1)
D                AMAT(1),    VEC(1),     DATA(1),    U2(1),      LOCM1(1)
 EQUIVALENCE     (Z, IZ, PAR, LOC, A0, DA, TDP, TDM, TWP, TWM, ESTAR, U, UI,
E                 WP, WM, WPI, WMI, H, BUFF, V0, BOUND, R, RST, B, A, SIM, V0ST,
E                 V, VSTST, V02ST, V12ST, V2ST, V3, EDIAG, KDIA, PDNS, AMAT,
E                 VEC, DATA, U2, LOCM1)
 LOGICAL         TDP,        TDM,        TWP,        TWM
 COMMON/CAREME/
C                IS,         IS0,        MODE,       MODEP,      KASE
 COMMON/CASME/
C                NCASE
 COMMON/COMME/
C                WB,         WA,         WA0,        WA00,       V0TEV0,
C                DFT0,       DFT1,       UTV,        T,          TPR,
C                TDPR,       CHISQ,      WINC,       FINT0,      TEST,
C                VSTSQ
 LOGICAL         TEST
 COMMON/CORME/
C                LSSTST,     NREL,       LLIST,      LLISTE,     T2,
C                LQ,         LRQ,        UTPV
 COMMON/COUNT/
C                NEVT,       KF,         NGEV
 COMMON/CSYMB/
C                X(44)
 COMMON/ENTAME/
C                LACENT,     LMAT,       LVEC,       LDATA,      LFP,
C                LFM,        LAF,        KPALME,     KMAT,       KVEC,
C                FACMAT,     FACVEC,     FSCAL,      TMAT,       TVEC,
C                TSCAL,      TWPM,       TFPM
 LOGICAL         TMAT,       TVEC,       TSCAL,      TWPM,       TFPM
 COMMON/ERME/
C                TABORT,     TSTOP
 LOGICAL         TABORT,     TSTOP
 COMMON/ERRR/
C                IECSUM,     LCSUM,      NRTT
 COMMON/ETAME/
C                M2PTR(2)
```

```
 COMMON/FLAME/
C                NREF,          IFLAME
 COMMON/FUME/
C                IFUME
 COMMON/GAME/
C                KTYPE,         KEND,          TPRINT
 LOGICAL         TPRINT
 COMMON/GET/
C                MS,            ML,            NL
 COMMON/HCHI/
C                HCHI(100)
 COMMON/HISTLB/
C                HLAB(3,1)
 COMMON/HISTME/
C                IBEG,          IHIST,         IDATA,         IEVTH
 COMMON/HUME/
C                Y,             ETA(2),        KDER,          IPT,           NW,
C                MULT,          KIND,          ICASE
 COMMON/HUME1/
C                LMAXI,         TDER,          TALARME,       TEMERG
 LOGICAL                        TDER,          TALARME,       TEMERG
 COMMON/H1/
C                HZ(1)
 COMMON/HZ/
C                DH(1)
 COMMON/H3/
C                NH(1)
 COMMON/H4/
C                IH(1)
 COMMON/IMME/
C                LEVENT,        KBUFF,         LSTOR,         ITAPE,         NOEVR,
C                NOEVF,         LWORDL,        NEVF,          NDATA
 COMMON/INDXS/
C                M1,            M2,            LNBOUT
 COMMON/INIT/
C                PB(4)          PT(4)
 COMMON/MAXIME/
C                NPAR,          NMAX,          KSTEPM,        JOB,           KSTEP,
C                KSTEPR,        NFIXED,        NBOUND,        ACFIL
 LOGICAL         ACFIL
 COMMON/MDEL/
C                T,             COSINE
 COMMON/MIME/
C                ETABAR(20),    FMIME(20),     WTSAGE(20),    WTSQ(20)
 COMMON/NORME/
C                NEV,           WTEV,          NINT,          WTINT,         WTRAME,
C                WTHIST
```

```
 COMMON/NUNAME/
C                NMAT,          NBUFF,         DACUTM,        T2ND,          TCONST,
C                WI,            WR
 LOGICAL         T2ND,          TCONST
 COMMON/PRIME/
C                CHILIM,        RLIM,          ELIM,          TMAX,          TMIN,
C                TCORME,        TDIAG,         TDIRCO,        WT
 LOGICAL         TDIAG,         TDIRCO,        TCORME
 COMMON/RAME/
C                ISD,           ISBUF1,        ISBUF2,        ISUBF3
 COMMON/RGENCOM/
C                GEN(5),        K1
 COMMON/RENO/
C                NP,            ECM,           AMASS(10)
 COMMON/SAGEIN/
C                XX(18)
 COMMON/SAGELL/
C                XXX(3)
 COMMON/SAGME/
C                PLAB,          PCM(4,11),     P(8)
 COMMON/SAGEWT/
C                XXXX(5)
 COMMON/SCRAT/
C                NSCR
 COMMON/SHAT/
C                FAC0,          FAC(30)
 COMMON/STOR/
C                LSTO
 COMMON/STORME/
C                LPAR,          LLOC,          LA0,           LDA,           LTDP,
C                LTDM,          LTWP,          LTWM,                         LESTAR,
C                LLOCM1,        LU,            LUI,           LWP,           LWM,
C                LWPI,          LWMI,          LH,            LBUFF,
C                LV0,           LBOUND,        LWORK0,        LR,            LWORK,
C                LRST,          LB,            LA,            LSIME,         LV0ST,
C                LV,            LVSTST,        LV02ST,        LV12ST,        LV2ST,
C                LV3,           LEDIAG,        LKDIA,         LPDNS,         LU2,
C                LV4,           LVST,          LERR
 COMMON/SWITCH/
C                ISW
 COMMON/TIME/
C                ISTAT(7),      ISTAT0(7),     AU,            CP,            TMR,
C                TTIME
 LOGICAL         TTIME
 COMMON/TOME/
C                HFAC,          NTOTF(20)
 COMMON/TOTALS/
C                NHISTS
 COMMON/ULABEL/
C                ILABEL(3)
```

```
 COMMON/VEGAS/
C                BEAM,      TARGET,    CRAP(6)
 COMMON/WRIME/
C                KTEV,      KTINT,     KTSCR,     KTHISTR,  KTHISTF,
C                KTREAL,    KTFAKE,    KWRIME,    KWERR,    KTAPE,
C                NWRD
```

8.3 Input/Output

OPTIME normally requires one or more files of input data and writes, usually to a disc, one or more files of data suitable to be processed during the fitting and post-fit processing.

Some or all of the outputted data may be saved so that in subsequent runs of OPTIME the pre-fit processing may be eliminated. See Section 7.2 for details.

Printed output consists primarily of relevant information describing the state of affairs at each step, which may be suppressed, and of histograms, if the user chose to produce them.

8.31 Standard input files and formats

The standard input to OPTIME is usually one or more tapes of experimental data which will be read during the pre-fit processing by a subroutine GETUM1. The user chooses a filename for this tape and stores this name in the variable KTREAL. At LRL the standard choice is TAPE5 for the filename and consequently KTREAL = 5.

The format of this tape that the standard GETUM1 will accept is the so-called ARROW format [2]. If other formats are to be read the subroutine GETUM1 must be replaced.

For all types of fit, except for KTYPE = 11, integration data are required also. These data usually are not read from an input file but are generated, one datapoint at a time. A Monte Carlo method event generator, called SAGE, part of the OPTIME system, may be used for this purpose [3]. If these data were prepared outside of OPTIME and saved on tape, then they may be read if the subroutine GETUM2 is replaced. The user would choose a filename for this tape and store it in the variable KTFAKE. At LRL the choice is usually the filename TAPE4 and correspondingly KTFAKE = 4.

For all types of fit, except KTYPE = 11, two files of data are read repeatedly during the fitting phase. These files, KTEV, containing experimental data, and KTINT, containing integration data, are usually prepared by IMME and INTIME via the user-subroutine DOME. At LRL the choice for the filenames is usually TAPE1 and KTEV = 1, and TAPE2 and KTINT = 2. The format of these files is as follows:

	Word	
Event 1	1	No. of words in event, excluding this one (= n + 1)
	2	Weight for this event
	3	Data word 1
	4	Data word 2
	:	:
	n+2	Data word n

These datapoints are packed into logical records of max. (NBUFF-3)/2 60-bit words.

When histogramming experimental and integration data, two files, containing these data, are read by the subroutine COCHIS. At LRL the choice for these filenames is usually TAPE20 and TAPE30, and consequently KTHISTR = 20 and KTSCR = 30. The format of these files is as follows:

2 words/datapoint
max. 510 60-bit words/log. record

	Word		12 bits
1 data point	1	Data word	2 nnn
	2	Weight for this data point	0000

where nnn = histogram number.

8.32 Standard output files and formats

During the pre-fit processing the files KTEV and KTINT are written as already described in the previous section. In addition intermediate integration data is written by the subroutine INTIME to a scratchfile KTSCR whose filename is usually TAPE30 (KTSCR = 30). This file is also used later for histogramming by the subroutine COCHIS.

When histogramming is done the subroutine HIST, called by the user from the subroutine HISTME, writes for experimental data a file KTHISTR, already described in the previous section, and for integration data a file KTHISTF, whose filename is usually TAPE29 (KTHISTF = 29). This file contains the unnormalized integration histogram data and after the fit is read by the subroutine FAME. The format of this file is as follows:

		Word		
Event 1	"flag"	1	No. of words to follow	(= 1)
		2	all bits on	
	data point	3	No. of words to follow	(=NW of DOME+1)
		4	Preliminary weight for this event	
		5	Data word 1	
		6	Data word 2	
		⋮	⋮	
		NW+4	Data word NW	
	histogram information	.	No. of words to follow	(= 2)
		.	Data word to be histogrammed	2 nnn
		.	weight for this histogram point	0000
		.	No. of words to follow	(= 2)
		.	Data word to be histogrammed	2 nnn
		.	weight for this histogram point	0000
		⋮	⋮	

12 bits

max. (NBUFF-3)/2 60-bit words/log. record , nnn = histogram number

9.0 CRT HISTOGRAMS

By replacing some standard subroutines it is possible to make CRT histograms instead of printed histograms. For the experimental data an envelope will be drawn while the fitted curve will be represented by a superposed dot (.) in the center of each bin.

In addition to plot titles, labels for the x and y axes may be given with the following procedure (see also [6]):

```
COMMON/ULABEL/ILABEL(3)
COMMON/HISTLX/HX(3, NHISTS)
COMMON/HISTLY/HY(3, NHISTS)
```

Set ILABEL(1) to	meaning
0	no plot or axis labels
1	plot labels, no axis labels
2	plot labels, x-axis labels, no y-axis labels
3	plot labels and axis labels

then fill HX and HY with up to 3 words each of up to 10 Hollerith characters each for each histogram.

The production of CRT histograms is effected by replacing the routines COCHIS and HISTUM, plus those which are called by them, by CRT versions of these routines that are available from Werner Koellner.

At Berkeley, replacement of those routines will be done automatically by the selective loader if those routines are loaded into core ahead of the other routines.

The CRT routines require about 7_8K more locations of core.

10.0 OPTIME MAINTENANCE AND CHARACTERISTICS

10.1 Programming Characteristics

For the time being the official version is written with the following characteristics:

Language: FORTRAN and COMPASS

Computer: CDC 6000 series or higher

Core requirement: 40_8K to 70_8K of 60-bit words, depending on user requirements. Blank COMMON is increased or decreased at various times during the execution of OPTIME. This feature must be disabled if blank COMMON is not located behind the program.

No. of files required (disc or tape): Min. 1, max. 6, depending on user requirements.

At LRL, compiling and loading into core of a program is binary subroutine oriented. The compiler and assembler produce relocatable binary code, therefore the static part of the OPTIME system is precompiled. The user compiles the necessary user subroutines, and with the use of a selective loader only the minimally required code is loaded into core [7].

10.2 Program Maintenance

The official version of the program is maintained by:

Werner Koellner
U.C. Lawrence Radiation Laboratory
Group A Programming
Berkeley, CA. 94720

It is maintained with the aid of a library maintenance program, called UPDATE [8], which assigns to each source card image, in columns 73 through 80, a unique identifier.

If a user finds a programming bug or has an idea how to improve the program, we wish he would communicate this information to Werner Koellner, who will make the correction to the official version if it is possible.

10.21 Distribution

Distribution of the program and of program changes will also be

handled by Werner Koellner. We would like to recommend that potential users get the official version of the program from him directly, because he will supply a version with most of the mistakes communicated to him corrected.

Mailing lists will be maintained to provide general information about the system and to communicate program changes.

10.22 Version identification

At the beginning of each subroutine there will be a comment card specifying the version of the current code by date. We urge that anyone who would like to make changes of his own in the program insert some comment, dating it, to help identify the unofficial versions.

ACKNOWLEDGMENTS

The authors are indebted to O. I. Dahl, J. H. Friedman, and A. Rittenberg for generous programming assistance and for supplying us with parts of the KIOWA program. In addition, J. H. Friedman has provided the SAGE Monte-Carlo event generator, and P. L. Hoch, T. S. Mast and S. M. Flatté have reviewed the manuscript. The authors thank them as well as members of the Group A Programming Staff and many users of the program for valuable help.

REFERENCES AND FOOTNOTES

* Work done under the auspices of the U.S. Atomic Energy Commission.

1. P. H. EBERHARD and W. O. KOELLNER, "The OPTIME System for Fitting Theoretical Expressions," Lawrence Radiation Laboratory Report UCRL-20159, Lawrence Radiation Laboratory, Berkeley, 1970.
2. N. L. GOULD, "6600 Arrow System," Group A Programming Note P-186, Lawrence Radiation Laboratory, Berkeley, 1969.
3. J. FRIEDMAN, "Sage," Group A Programming Note P-189, Lawrence Radiation Laboratory, Berkeley, 1969. J. FRIEDMAN, "Random Event Generation with Nonconstant Frequency Distributions," Lawrence Radiation Laboratory Report UCRL-19206, Lawrence Radiation Laboratory, Berkeley, 1969 (to be published in J. Comp. Phys.). J. FRIEDMAN, "Sage 3," Group A Physics Note, Lawrence Radiation Laboratory, Berkeley, 1970 (received from the author).
4. Subroutines used here are adapted from the program KIOWA, see J. FRIEDMAN and A. RITTENBERG, "KIOWA, A General Description," Group A Programming Note P-171, Lawrence Radiation Laboratory, Berkeley, 1968.
5. O. DAHL, J. FRIEDMAN, and A. RITTENBERG," Alvarez Group Tape Read Routines for Use With Kiowa," Group A Programming Note P-172, Lawrence Radiation Laboratory, Berkeley, 1968.
6. J. FRIEDMAN and A. RITTENBERG, "C. R. T. Kiowa For The CDC-6600," Group A Programming Note P-173, Lawrence Radiation Laboratory, Berkeley, 1968.
7. D. F. STEVENS, editor, "BKY Users' Handbook," 3rd ed., Lawrence Radiation Laboratory, Berkeley, 1970.
8. UPDATA is a library maintenance program furnished by Control Data Corporation and adapted for use at LRL. The basic program is described in the CDC SCOPE 3.0 system manual.

TECHNICAL INFORMATION DIVISION
LAWRENCE RADIATION LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720