SLAC Computation Group
Stanford, California

CGTM No. 146
January 1973

SLAC KIOWA

General Description

Jerry Friedman
Roger Chaffee

ABSTRACT

KIOWA is a data summarizing program which produces histograms, scatter plots, and Gaussian ideograms of data stored on magnetic tape files. The control quantities which define the plots for KIOWA are set up with FORTRAN coding in user-coded routines, and the plots are printed on a standard computer line printer. This paper describes KIOWA for the SLAC IBM 360/91. Versions of the program also exist for CDC 6600/6400, IBM 7044/7094, and IBM 360/75.

I. SLAC KIOWA, A GENERAL DESCRIPTION

A. Introduction

The principal motivation for coding KIOWA was to have a simple program to summarize data which executes very rapidly, consumes little storage, allows maximum versatility in setting up control quantities, is easy to use, and is easy to modify.

The program consists of a main routine coded by the user and a series of subroutines, some of which are also coded by the user. Thus, the user is required to have a rudimentary knowledge of the FORTRAN language.

The type, number, and control quantities (boundaries, bin widths, number of bins, labels) for the displays are set up by the user in the main routine with FORTRAN coding. The user may enter the control quantities with FORTRAN DATA statements, FORTRAN executable statements, or by reading data cards in formats of his own choosing.

The determination to enter some word(s) from a particular event on some particular plot(s) is made in a user-coded subroutine. This subroutine is called by the program for each event; the event record, as read from the input tape, is available to the routine. Using any FORTRAN coding desired, the user determines what is to be plotted and calls a subroutine to enter each quantity into the desired plot.

Since all testing is accomplished with FORTRAN logic and the control quantities for the plots are set up with FORTRAN coding, it is possible to produce a large number of plots with minimum effort on the part of the user.

Other subroutines are optionally coded by the user to superimpose curves on histograms and scatter plots, and to do any special initial and terminal processing desired.

B. General Flow

After setting up the control quantities in the main routine, the user transfers control to the executive subroutine with a FORTRAN CALL statement. This routine first checks all of the control quantities for inconsistencies. If an inconsistency is found, the program either resets the quantity in error to some nominal value, or terminates if further processing is impossible.

Next, KIOWA reads the input tape and processes values for the various plots. If enough core is available to contain all the plots, points are processed and binned at this time. If not, KIOWA writes a densely-packed scratch file containing just the information needed to make the plots at a later time. In either case, the input tape is read only once, and any subsequent processing is done from the scratch file.

Next, in the case of in-core storage, the plots are printed. If the scratch file was used, KIOWA makes as many plots as will fit in its "dynamic storage area", as set aside by the user in the main routine. These plots are then printed. Another pass is made, making and printing as many more plots as will fit in core. As many passes on the scratch file are made as are necessary to make all of the plots. Finally, KIOWA resets flags to the starting values and returns to the calling program.

C. Miscellaneous

KIOWA will superimpose user-coded curves on histograms and calculate a chi-square between the curve and the histogram. It will superimpose any number of user-coded curves on scatter plots. Histograms may be made with linear or logarithmic scaling.

The user need not specify the boundaries of his plots. At his request, KIOWA will adjust the boundaries of any plot to contain all of the events.

II. USAGE OF KIOWA

A. Introduction

There are three separate parts which the user must prepare in order to run KIOWA. In the main routine, before the FORTRAN statement CALL COCHIS, some quantities must be set to define the plots to be made. During the event accumulation, KIOWA calls subroutine GETUM, in which the user is expected to read an input tape and unpack the data for processing in the USER routine. GETUM subroutines are generally coded for classes of data tapes, and invoked, like KIOWA, in packaged form. On each call, GETUM unpacks the data for one event, and stores it for USER. Finally, SUBROUTINE USER is called to process the event data and select quantities to plot, which are transmitted to KIOWA by calls to HIST, PLOT, or GRAM.

In addition, subroutines START, FINISH, ALDONE, USRHST, USRGRM, UFUN, and UPLOT are called to allow special user options. Do-nothing routines are supplied by KIOWA, which can be overridden if desired.

B. The Main Program

The main program begins with the series of FORTRAN cards:

```
COMMON C(ndim)
COMMON/LENGTH/LEVENT,LBUFF,LSTOR
COMMON/TOTALS/NHISTS,NPLOTS,NGRAMS
COMMON/H1/HZ/H2/DH/H3/NH/H4/IH
COMMON/P1/XL/P2/XU/P3/YL/P4/YU/P5/NX/P6/NY
COMMON/G1/GZ/G2/DG/G3/NG
COMMON/ULABEL/ILABEL(3)
```

The array C is the general working area of the program and is used to hold such things as the event array, the data tape and scratch file buffers, the dynamic storage where the plots are accumulated, and temporary storage for the plotting routines. C, which is singly dimensioned (its dimension, ndim, is discussed below), is divided into three contiguous areas, whose individual lengths are specified by the user by setting the three quantities LEVENT, LBUFF, and LSTOR; this is generally done by a DATA statement:

DATA LEVENT,LBUFF,LSTOR/n_e,n_b,n_s/

C(1) through C(LEVENT) is the area where GETUM will place data for each event, and thus LEVENT must be at least as large as the largest event on the tape.

C(LEVENT+1) through C(LEVENT+LBUFF) is used by GETUM as the data tape buffer area, and thus LBUFF must be at least as large as the longest record on the tape.

C(LEVENT+LBUFF+1) through C(LEVENT+LBUFF+LSTOR) is used for the dynamic storage area in which the plots are accumulated. LSTOR should be as large as possible (subject to core limitations, or reasonableness in the case of multiprocessing machines) in order to reduce the number of passes on the scratch file.

In general, LEVENT and LBUFF are determined by the input data. In any case, KIOWA requires LEVENT+LBUFF $\geq$ 800, and LSTOR $\geq$ 1270.

Since the array C contains all of the three areas discussed, its dimensions, ndim, must be equal to the sum of the values used for LEVENT, LBUFF, and LSTOR.

Following the above COMMON and DATA cards, the user must set up the control quantities for the displays and store them in the locations described below. It is important that the names for the controls match those given in the COMMON blocks above. The values may be set up either with FORTRAN DATA statements, or by FORTRAN executable statements, or by being read in from data cards with FORTRAN READ statements.

The following quantities determine the number of each type of plot:

NHISTS is the number of histograms desired (integer ≤ 1023);
NPLOTS is the number of scatter plots desired (integer ≤ 1023);
NGRAMS is the number of Gaussian ideograms desired (integer ≤ 1023).

The following quantities control histogram displays <u>and must be dimensioned NHISTS</u> (or larger):

HZ(n) - lower limit of <u>n</u>th histogram (real);
DH(n) - bin width of <u>n</u>th histogram (real);
NH(n) - number of bins in <u>n</u>th histogram (integer ≤ 120);
IH(n) - indicator of which kind of plot is desired (integer ≤ 12);
- = 1 - histogram, plain and simple;
- = 2 - histogram with user function superimposed;
- = 3 - histogram with user function superimposed and chi-square calculated;
- = 4 - "big bin" histogram--uses 120 spaces on paper even though only 2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 24, 30, 40, or 60 bins are requested;
- = 5 - "big bin" histogram with user function;
- = 6 - "big bin" histogram with user function and chi-square.

The values 7 through 12 are equivalent to 1 through 6, but with a logarithmic vertical axis.

The following quantities control scatter plot displays <u>and must be dimensioned NPLOTS</u> (or larger):

XL(n) - lower limit of abscissa quantity for <u>n</u>th plot (real);
XU(n) - upper limit of abscissa quantity for <u>n</u>th plot (real);
XN(n) - number of bins for abscissa quantity for <u>n</u>th plot (integer ≤ 100);

YL(n) - lower limit of ordinate quantity for *n*th plot (real);
YU(n) - upper limit of ordinate quantity for *n*th plot (real);
NY(n) - number of bins for ordinate quantity for *n*th plot (integer ≤ 50).

The following quantities control ideogram displays *and must be dimensioned NGRAMS* (or larger):

GZ(n) - lower limit of *n*th ideogram (real);
DG(n) - bin width of *n*th ideogram (real);
NG(n) - number of bins in *n*th ideogram (integer ≤ 120).

The quantities HZ(n), XL(n), YL(n), and GZ(n) may be set by the user to 1HF, 4HFIND, or 7777.0 if it is desired that the program adjust the lower limit of the corresponding plot coordinate so as to contain the smallest data point. Similarly, DH(n), XU(n), YU(n), and DG(n) may be set to 1HF, 4HFIND, or 7777.0 if the user wishes the program to adjust the upper limit of the corresponding plot coordinate so as to contain the largest data point.

There is an optional provision for inserting user titles on histograms, scatter plots, and/or ideograms:

(1) To insert titles on all histograms, set ILABEL(1) equal to 1, and include the statement:

```
COMMON/HISTLB/HLAB(8*,nhists)
```

and store the label of the *n*th histogram in HLAB(I,n),I=1,8*.

(2) To insert titles on all scatter plots, set ILABEL(2) equal to 1, and include the statement:

```
COMMON/PLOTLB/PLAB(8*,nplots)
```

and store the label of the *n*th scatter plot in PLAB(I,n),I=1,8*.

(3) To insert titles on all ideograms, set ILABEL(3) equal to 1, and include the statement:

```
COMMON/GRAMLB/GLAB(8*,ngrams)
```

and store the label of the *n*th ideogram in GLAB(I,n),I=1,8*.

* = 5 for 7044/7094 versions, = 3 for CDC version

Each title is 8[*] words and, thus, 32[+] characters long. If all 32[+] characters are not used, the unused ones must be set to blanks. Also, if all titles for a particular type of plot are not used, the unused title locations must be set to blanks. Titles, like other control quantities, may be set up with FORTRAN DATA statements, FORTRAN executable statements, or FORTRAN READs from data cards.

Finally, the main program must call the working subroutine with the statement:

```
CALL COCHIS
```

and terminate with the FORTRAN END card.

C. The User Subroutine

The USER subroutine begins with the series of cards:

```
SUBROUTINE USER
COMMON C(500)
DIMENSION IC(500)
EQUIVALENCE (C,IC)
COMMON/EVINFO/NEVT,NWRDS
COMMON/STOP/ISTOP
```

USER is called once for each event on the data tape (except that tape reading may be terminated at any time by setting ISTOP non-zero). When USER is called, the event array will be found in C, the ordinal number of the event will be given by NEVT, and the number of words in the event will be given by NWRDS.

For each call to the USER routine (i.e., for each event) the user decides to which plot(s), if any, the event is to be contributed, how many times on each plot, and what quantity is to be plotted in each case. A quantity is entered into a plot by calling the appropriate plotting routine[**]:

```
CALL HIST(I,A,WT)        for histograms;
CALL PLOT(I,X,Y)         for scatter plots;
CALL GRAM(I.C,E,WT)      for Gaussian ideograms.
```

[*] = 5 for 7044/7094 versions, = 3 for CDC version

[**] The names H,P, and G may (but need not) be substituted for HIST, PLOT, and GRAM

[+] = 30 for CDC and 7044/7094 versions

The quantities in the calling sequence are as follows:

I = number of the plot of this type to which the event is to be entered (an integer);

WT = weight to be applied to the entry (a real number);

A = quantity to be histogrammed (real);

X = abscissa of point to be plotted (real);

Y = ordinate of point to be plotted (real);

C = central value of quantity to be ideogrammed (real);

E = standard deviation of quantity to be ideogrammed (real).

The calling sequence quantities may be any legal FORTRAN expression, such as constants, variables, arithmetic expressions, and calls to function subroutines. Some examples are:

```
CALL HIST(4,C(12),1.0)
CALL PLOT(2*J-1,C(14)**2,COS(C(16)))
CALL GRAM(16,DELSQ,DDELSQ,C(10))
```

The user may employ any type of FORTRAN coding in the USER routine to decide upon which plots an event should go and how many times, as well as to construct quantities to be plotted. If an event is not to be added to any plot, the user should simply return to the calling program. An example of a sequence of coding might be:

```
   IF(C(26).LT.C(3))CALL HIST(IC(2),SQRT(C(4)),0.5)
   NPLOT=(C(61)-13.30)/2.0+1.0
   IF(NPLOT.GT.10)GO TO 20
   CALL PLOT(NPLOT,C(5),C(6))
   CALL PLOT(NPLOT,C(5),C(7))
20 FMSQ=C(31)**2
   IF(FMSQ.LE.0.7)RETURN
   DO 30 I=1,10
   IF(C(I+40).GT.FMSQ)CALL HIST(I+5,C(I+50)+0.5*FMSQ,C(20))
30 CONTINUE
   RETURN
```

Multiplicities (plotting more than one word from the same event on the same plot) are accomplished by simply calling HIST, PLOT, or GRAM more than once for a given plot for the same event.

Finally, the USER routine ends with the usual FORTRAN END card.

D. The START, FINISH, and ALDONE Routines

The working program makes three special calls to user routines, in addition to calling USER once per event. Dummy versions of these routines are provided, and the user need insert his own versions only if he has need for them.*

SUBROUTINE START is called once, before any other processing, and may be used for any initialization procedure desired.

SUBROUTINE FINISH is called once, just after the reading of the data input tape, and may be used for any finalization procedures desired. Note that START, USER, and FINISH together may be used for accumulating and printing out any information of interest.

SUBROUTINE ALDONE is called once, just before the return to the calling program.

E. Plotting Functions over Histograms

In order to superimpose a curve on a data histogram, the user must do two things:

(1) Set IH(n) for the desired _n_th histogram to the appropriate value, as described above in the section on the main program; the legal values are IH(n)=2,3,5,6,8,9,11,12

(2) Code the routine UFUN with the following calling sequence:

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

N is the histogram number upon which the curve is to be superimposed, and X is the abscissa coordinate for which the function is to be evaluated. Both N and X are input to the routine; the user must execute a statement of the form

```
UFUN = some function of N and X
```

before returning to the calling program. The user function may have arbitrary normalization since the histogramming routine automatically normalizes it to the total number of events in the histogram.

* For versions of FORTRAN allowing multiple entry points, these routines may be made entry points to USER

F. Plotting Curves on Scatter Plots

In order to superimpose a curve on a scatter plot, the user must code the routine UPLOT with the following calling sequence:

```
FUNCTION UPLOT(X,Y,N)
```

N is the number of the scatter plot upon which the curve is to be superimposed; X and Y are the abscissa and ordinate for which the function is to be evaluated. All are input to the routine.

The calling program assumes the function UPLOT changes sign at those values of X and Y that define the curve. Thus, if the curve can be defined by

$$f(x,y)=0$$

where f is some functional form of x and y, then the FORTRAN statement

```
UPLOT=f(X,Y)
```

will serve to define the curve for the calling program. If the conditions which define the curve cannot be expressed in simple closed form, the user can set UPLOT=+1.0 or -1.0, depending upon whether the point (X,Y) is inside or outside the curve.

The calling program calls UPLOT for each plot. Thus, for those plots which are not to have a curve, the user should set UPLOT=1.0 before returning.

If several curves are desired on a single scatter plot, then the user should set

$$\mathrm{UPLOT} = \prod_{i=1}^{N_c} f_i(X,Y) \ ,$$

where N_c different curves are to be superimposed on the scatter plot and $f_i(X,Y)=0$ defines the *i*th curve.

G. KIOWA Printed Output

1. Histograms:

Two histogram formats are available with KIOWA, the standard "SUMX" format and the "big bin" format. The user selects which is used for a given histogram by setting the histogram flag IH(n) to the appropriate value as described in the section on the main routine.

The standard "SUMX" format allows one printer column per bin with a maximum of 120 bins. The number of events in each bin is printed vertically, below the pictorial column for the bin.

The "big bin" format uses more than one column per bin and always uses 120 columns to display the histogram. The number of bins for a "big bin" histogram display must divide evenly into 120. Legal values are 2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 24, 30, 40, and 60. At the base of the histogram the lower edge value of each bin in printed vertically below the lower edge of the bin, and the number of events in each bin is printed vertically below the center of the bin.

The sum of weights which underflow, which fall in the histogram, and which overflow, as well as the largest and smallest values and the mean and standard deviation, are printed at the bottom of the plot.

2. Scatter Plots:

The printed scatter plot displays the number of events which fall into each bin with either a blank (for no events), number (for 1 to 9 events), an alphabetic letter, A through U (for 10 to 30 events), or a letter V (for more than 30 events). The lower bin edge of both the ordinate and abscissa are printed at the edge of the plot, as well as the projections on the x and y axes. The various overflows are printed in the following manner:

PLOT STATISTICS=n1 n2 n3/n4 n5 n6/n7 n8 n9

where

n1 = number of events underflow x and overflow y;

n2 = number of events in x range and overflow y;

n3 = number of events overflow x and overflow y;

n4 = number of events underflow x and in y range;

n5 = number of events in x range and in y range;
(number of events plotted on scatter plot);

n6 = number of events overflow x and in y range;

n7 = number of events underflow x and underflow y;

n8 = number of events in x range and underflow y;

n9 = number of events overflow x and underflow y.

3. Gaussian Ideograms:

Gaussian ideograms are printed in the standard "SUMX" format described above. The sum of event contributions which underflow, which fall in the ideogram, and which overflow, as well as the largest and smallest values and the mean and standard deviation, are printed at the bottom of the plot.

H. User Coded Input Tape Read Routine

The user may enable KIOWA to read data tapes of his particular format by coding a routine called GETUM, beginning with the statements:

```
SUBROUTINE GETUM(B,LW)
COMMON C(500)
COMMON/LENGTH/LEVENT,LBUFF,LSTORE
DIMENSION B(1),IC(1)
EQUIVALENCE (C,IC)
COMMON/EVINFO/NEVT,NWRDS
```

The working program calls GETUM to obtain each event from the data input tape(s). It will keep calling this routine for events until it receives a flag indicating there are no more events.

Before returning to the calling program,GETUM must store the event data record in the array C, in blank common. LW is a flag to the calling program. For each event, LW must be set positive before returning to the calling program. When there are no more events, GETUM must set LW=0 before returning. This flags the end of the event data for the calling program, and it will not call GETUM again.

The array B is a utility storage area not used by KIOWA in the input tape reading phase. It is LBUFF in length. Thus, GETUM may use this storage area for input tape buffers or other utility purposes.

The ordinal number of the event, and the number of words in the event record may be stored in NEVT and NWRDS for use in the USER routine.

It should be noted that GETUM need not read a physical input tape(s). For example, the user may wish to plot Monte Carlo generated events; in this case, he can code GETUM to generate random events and return them to the calling program.

I. In-Core Operation

If the "dynamic storage area" in blank common is large enough, KIOWA can build the plots directly, without using the scratch file. This mode of operation is much more efficient than the scratch-file mode. The length of the "dynamic storage area" is set by the user in LSTOR, as described above.

The minimum space is given by

$$\text{MINIMUM} = \sum_{I=1}^{NHISTS} 8 + NH(I)$$

$$+ \sum_{I=1}^{NGRAMS} 8 + NG(I)$$

$$+ \sum_{I=1}^{NPLOTS} 11 + (NX(I) * NY(I))/6$$

$$+ 5$$

Another 120 words are needed if histogram or ideogram limits are to be found. The in-core mode is not allowed if scatterplot limits are to be found.

J. User Access to Histogram Contents

After all the event accumulation, and just before each histogram is printed, KIOWA executes the statement CALL USRHST(A,N). A is a dimensioned array containing the values which will be plotted, and N is the number of the histogram. The user could, but does not have to, code his own SUBROUTINE USRHST(A,N) to change or add to the histogram values, or punch them on cards.

Similarly, KIOWA calls SUBROUTINE USRGRM(A,N) just before plotting each Gaussian ideogram, so the user can modify the ideogram contents by writing his own subroutine.

A similar process for modifying scatterplot contents does not exist.

APPENDIX A

UTILITY ROUTINES FOR PLOTTING PHASE SPACE CURVES ON HISTOGRAMS AND DALITZ AND CHEW-LOW BOUNDARIES ON SCATTER PLOTS

For the user's convenience, the following four routines exist in KIOWA. PHASEP and PHASEQ calculate phase space curves for two particle invariant mass and mass squared distributions respectively, in three particle final states for use with the UFUN user routine. DALITZ and CHEWLO calculate Dalitz and Chew-Low boundaries, respectively, for use with the UPLOT user routine.

```
UFUN=PHASEP(X,ECM,A1,A2,A3)
UFUN=PHASEQ(X,ECM,A1,A2,A3)
UPLOT=DALITZ(X,Y,ECM,A1,A2,A3)
UPLOT=CHEWLO(X,Y,ECM,B1,B2,A1,A2,A3)
```

where

ECM = center of mass energy;
B1,B2 masses of initial state particles;
A1, A2, A3, = masses of final state particles;
X = invariant mass A1,A2 for PHASEP;
X = invariant mass squared A1, A2 for PHASEQ, DALITZ, and CHEWLO;
Y = invariant mass squared A1, A3 for DALITZ;
Y = momentum transfer squared B2 to A3 for CHEWLO.

APPENDIX B

AN EXAMPLE

In this section, detailed coding is presented for a rather complicated and involved sample problem in order to illustrate a few of the techniques for coding the various user subroutines in KIOWA.

Suppose the user wishes to study the reaction $K^-p \rightarrow \Lambda\pi^+\pi^-$ in the beam momentum range 2.0 to 3.0 GeV/c at ten different beam momentum settings. The following plots are required for all of the events as well as for each momentum sample separately.

1. Histograms
   Invariant mass of the three two-particle combinations;
   Invariant mass of $\Lambda\pi^+$ and $\Lambda\pi^-$ on same plot (multiplicity = 2);
   Production cosine of the three two-particle combinations;
   Decay cosine of the three two-particle combinations;
   Decay azimuth of the three two-particle combinations.
2. Scatter Plots
   Dalitz plot $\mu^2(\Lambda\pi^+)$ vs $\mu^2(\Lambda\pi^-)$
   Chew-Low Plot $\mu^2(\Lambda\pi^+)$ vs $\Delta^2(\Lambda\pi^+)$
   Chew-Low Plot $\mu^2(\Lambda\pi^-)$ vs $\Delta^2(\Lambda\pi^-)$
   Chew-Low Plot $\mu^2(\pi^+\pi^-)$ vs $\Delta^2(\pi^+\pi^-)$
3. Gaussian Ideograms
   Invariant mass of the three two-particle combinations;
   Resolution function* of the three two-particle combinations.

Thus, in all, 143 histograms, 44 scatter plots and 66 Gaussian ideograms are required. Suppose, further, phase space curves are to be plotted over the mass plots; the function $\cos^2\theta$ is to be plotted over the 33 decay cosine distributions; and the function $1-\cos 2\varphi$ is to be plotted over the 33 decay azimuth histograms, as well as the χ^2 evaluated between the curves and the data histograms. Also kinematic boundaries are to be put on the Dalitz and Chew-Low plots. For the Dalitz plot, where all beam momenta samples are plotted together, all ten boundaries are to be superimposed.

*A resolution function is a Gaussian ideogram where all events have the same central value.

The user coding could be:

```
      COMMON C(6000)
      CALL COCHIS
      STOP
      END
      BLOCK DATA
      COMMON /LENGTH/ LEVENT,LBUFF,LSTOR
      COMMON /TOTALS/ NHISTS,NPLOTS,NGRAMS
      COMMON /H1/HZ/H2/DH/H3/NH/H4/IH
      COMMON /P1/XL/P2/XU/P3/YL/P4/YU/P5/NX/P6/NY
      COMMON /G1/GZ/G2/DG/G3/NG
      COMMON /ULABEL/ILABEL(3)
C
      DIMENSION HZ(143),DH(143),NH(143),IH(143),GZ(66),DG(66),NG(66)
      DIMENSION XL(44),XU(44),NX(44),YL(44),YU(44),NY(44)
C
      DATA HZ,DH/44*1HF,66*-1.0,33*0.0,44*1HF,66*0.02,33 *3.6/
      DATA NH,IH /143*100,143*3/
      DATA XL,XU,YL,YU,NX,XY /176*1HF,44*100,44*50/
      DATA GZ,DG,NG /33*1HF,33*-1.0,33*1HF,33*0.04,33*100,33*50/
      DATA NHISTS,NPLOTS,NGRAMS /143,44,66/
      DATA LEVENT,LBUFF,LSTOR /500,1024,4476/
      END
      SUBROUTINE  USER
      COMMON C(100)
      REAL MASS(3).PANG(3).DANG(3).AZANG(3).DELSQ(3).ERRGR(3)
      EQUIVALENCE (C10),MASS),(C(13),ERROR),(C(16),DELSQ,(C(19),PANG),
     0 (C(22).DANG).(C(25).AZAN).(WEIGHT.C(8)).(PMOM.C(6))
      REAL MOM(10)
      DATA MOM /2.1,2.2,2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0/
      IF (WEIGHT.LT.1.0) RETURN
      DO 1 IMOM=1,10
      IF (PMOM.EQ.MOM(IMOM)) GO TO 2
1     CONTINUE
      RETURN
2     DO 3 I=1,3
      M = 0
      L = 0
5     CALL HIST(L+I,MASS(I),WEIGHT)
      CALL HIST(M+I+44,COS(PANG(I)),WEIGHT)
      CALL HIST(M+I+77,COS(DANG(I)),WEIGHT)
      CALL HIST(M+I+110,AZANG(I),WEIGHT)
      CALL PLOT(M+I+11,MASS(I)**2,DELSQ(I))
      CALL GRAM(M+I,MASS(I),ERROR(I),WEIGHT)
      CALL GRAM(M+I+33,0.0,ERROR(I),WEIGHT)
      IF (M.NE.0) GO TO 4
      M = 3*IMOM
      L = 4*IMOM
      GO TO 5
4     IF (I.GT.2) GO TO 3
      CALL HIST(4,MASS(I),WEIGHT)
      CALL HIST(L+4,MASS(I),WEIGHT)
3     CONTINUE
      CALL PLOT(1,MASS(1)**2,MASS(2)**2)
      CALL PLOT(IMOM+1,MASS(1)**2,MASS(2)**2)
      RETURN
      END
      FUNCTION UFUN(X,N)
      DIMENSION E(10)
      REAL LAM,K
      DATA K,P,LAM,PI /.494,.93826,1.1153,.13958/
      DATA E /2.274,2.314,2.353,2.392,2.430,2.467,2.505,2.542,2.578,
     0 2.613/
      M = (N-2)/3 + 1
      IF (MOD(N-1,3).NE.0) UFUN=PHASEP(X,E(M),LAM,PI,PI)
      IF (MOD(N-1,3).EQ.0) UFUN=PHASEP(X,E(M),PI,PI,LAM)
      IF (N.GT.77 .AND. N.LE.110) UFUN=X**2
      IF (N.GT.110) UFUN=1.0-COS(2.0*X/57.2958)
      RETURN
      END
```

```
      FUNCTION UPLOT(X,Y,N)
      DIMENSION E(10)
      REAL LAM,K
      DATA K,P,LAM,PI /.494,.93826,1.1153,.13958/
      DATA E /2.274,2.314,2.353,2.392,2.430,2.467,2.505,2.542,2.578,
     ) 2.613/
      IF (N.GT.1) GO TO 20
      UPLOT=1.0
      DO 10 I=1,120
10    UPLOT = UPLOT*DALITZ(X,Y,E(N-1)LAM,PI,PI)
      RETURN
20    IF (N.GT.11) GO TO 100
      UPLOT=DALITZ(X,Y,E(N-1),LAM,PI,PI)
      RETURN
100   IF (N.LE.14) RETURN
      M=(N-15)/3 + 1
      IF (MOD(N-14),3).NE.0) UPLOT=CHEWLU(X,Y,E(M),K,P,LAM,PI,PI)
      IF (MOD(N-14),3).EQ.0) UPLOT=CHEWLU(X,Y,E(M),K,P,PI,PI,LAM)
      RETURN
      END
```

APPENDIX C

JCL for the SLAC 360/91

Loadmods for KIOWA subroutines are on the library WYL.CG.RBC.LOADMODS. There are dummy subroutines for GETUM and USER, so the user must supply his own to override them. The following decks will work, if the proper data input and GETUM are supplied.

A. With the Linkage Editor and FORTRAN X.

```
//XYZKIOWA JOB XYZ$AB,CLASS=B,GRAPH=YES
/*SETUP U=TAPE9,V=XY1234
//STEP1 EXEC FORTXCLG,PARM.GO='ONLY,TERSE,TIMEOUT=5'
//FORT.SYSIN DD *
      COMMON C(6000)
      CALL COCHIS
               ...
      END
/*
//LKED.SYSLIB  DD DSNAME=WYL.CG.RBC.LOADMODS,DISP=SHR
//             DD DSNAME=WYL.CG.JPS.LOADMODS,DISP=SHR
//           DD DSNAME=SYS4.FORTLIB,DISP=SHR
//           DD DSNAME=SYS1.FORTLIB,DISP=SHR
//           DD DSNAME=SYS3.FORTLIB,DISP=SHR
//LKED.MYLIB DD DSN=WYL.AB.XYZ.GETUMLIB,DISP=SHR
//LKED.SYSIN DD *
      INCLUDE MYLIB(GETUM)
/*
//GO.FT14F001 DD DSN=DATATAPE,LABEL=(2,SL),UNIT=TAPE9,
//   VOL=SER=XY1234,DISP=(OLD,KEEP)
//GO.FT03F001 DD UNIT=(SYSDA,3),SPACE=(CYL,(1,20)),
//   DCB=(RECFM=VBS,LRECL=1805,BLKSIZE=3614),
//   DISP=(NEW,DELETE),DSNAME=&SCRATCH
//
```

B. With the Loader and FORTRAN G.

```
//XYZKIOWA JOB XYZ$AB,CLASS=B,REGION=150K,GRAPH=YES
/*SETUP U=TAPE9,V=XY1234
//STEP1 EXEC FORTGCG
//FORT.SYSIN DD *
      COMMON C(6000)
      CALL COCHIS
               ...
      END
/*
//GO.SYSLIB  DD DSNAME=WYL.CG.RBC.LOADMODS,DISP=SHR
//           DD DSNAME=SYS4.FORTLIB,DISP=SHR
//           DD DSNAME=SYS1.FORTLIB,DISP=SHR
//           DD DSNAME=SYS3.FORTLIB,DISP=SHR
//GO.SYSLIN2 DD DSN=WYL.AB.XYZ.GETUMLIB(GETUM),DISP=SHR
//GO.FT14F001 DD DSN=DATATAPE,LABEL=(2,SL),UNIT=TAPE9,
//   VOL=SER=XY1234,DISP=(OLD,KEEP)
//GO.FT03F001 DD UNIT=(SYSDA,3),SPACE=(CYL,(1,20)),
//   DCB=(RECFM=VBS,LRECL=1805,BLKSIZE=3614),
//   DISP=(NEW,DELETE),DSNAME=&SCRATCH
//
```

ADDENDA

A. The JCL card referring to the dataset WYL.CG.JPS.LOADMODS in the Appendix C example is to provide the external references for FORTRAN X. If you use FORTRAN H instead of FØRTRAN X, you should leave out reference to this card, and start SYSLIB with the next.

B. Histograms (but not Gaussian ideograms or scatter plots) may have any number of bins, up to the limit imposed by the size of the "dynamic storage space". They are printed along the paper instead of across it. Due to storage limitations, user curves and chi-squared values can be included only in histograms with 200 bins or fewer.

C. User supplied curves are usually normalized to the total contents of the histogram. To suppress this, a flag may be set in the following manner:

```
COMMON /NORMAL/ NONE
DATA NONE /4HNONE/
```

D. KIOWA will plot the ratio of two histograms. The ratio call is transmitted through a common block, as usual.   COMMON /HRATIO/ IRATE(2,m)
The dimension m must be at least n+1, where n is the number of pairs to be done. IRATE(1,1) must contain the Hollerith constant 4HRATS. IRATE(2,1) contains n, and IRATE(1,I+1) and IRATE(2,I+1) contain the histogram numbers for the numerator and denominator, respectively, of the I-th pair.

The division is done bin-by-bin, without reference to the abscissa values. For this reason, it may be unwise to find automatic limits for histograms involved in this calculation. If the denominator is zero for a particular bin, the ratio is given as zero. There is one limitation which may be oppressive. The contents of both histograms must be in core at the same time, that is, both histograms must be printed during the same pass on the scratch file. This is automatically satisfied if the scratch file is not used, but you may have to put the histograms used in ratios among the first to be plotted, if you make a large number of histograms.

An example may make it clearer:

```
COMMON /HRATIO/ IRATE(2,10)
DATA IRATE /4HRATS,3,1,5,2,5,5,4/
```

will cause the ratios to be plotted for histograms 1/5, 2/5, and 5/4.

E. The storage requirements for each scatter plot given on page 12 have been increased to NX(I)*NY(I)/6 + NX(I) + NY(I) + 11 *The projections onto each axis are now accumulated with full-*word accuracy. A histogram of each projection, for each scatter plot, is printed if the value 4HPROJ is put in the one-word common block /PROJ/:

```
COMMON /PROJ/ IPROJ
DATA IPROJ /4HPROJ/
```