Computer Physics Communications 10 (1975) 343–367

MINUIT – A SYSTEM FOR FUNCTION MINIMIZATION AND ANALYSIS OF THE PARAMETER ERRORS AND CORRELATIONS

F. JAMES
Data Handling Division, European Organization for Nuclear Research, CH-1211 Geneva 23, Switzerland

and

M. ROOS
Department of Nuclear Physics, University of Helsinki, Sf-00170 Helsinki, Finland

Received 15 August 1975

PROGRAM SUMMARY

Title of program: MINUIT

Catalogue number: ACWH

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland (see application form in this issue)

Computer: CDC 7600; *Installation*: CERN, Geneva, Switzerland

Operating system: SCOPE 2.1.2 or 2.0

Programming language used: ANSI FORTRAN

High-speed storage required: 12000 words

Number of bits in a word: 60

Overlay structure: None

Number of magnetic tapes required: None

Other peripherals used: Card reader, line printer, card punch

Number of cards in combined program and test deck: 2600

Card punching code: ANSI

Keywords: General purpose, minimization, fitting, error analysis, correlation, simplex method, variable metric method, global minimum, contours.

Nature of physical problem
The final stages of the analysis of experimental data often consists of determining (estimating) a certain number of physical parameters and establishing the errors (confidence regions) of these parameters. This in turn generally involves one or more stages of function minimization to determine the best parameter values and various techniques for investigating the shape of the function near the minimum to determine the errors. In this essentially statistical problem the function involved is usually a chisquare or negative log-likelihood function, however many other physical problems of a non-statistical nature, such as finding the minimum energy configuration of a molecule, can be treated using the same techniques.

Method of solution
We have chosen to offer the user a large number of different techniques which can be conveniently invoked by the use of command cards. For example, three different minimization algorithms are available (a Monte Carlo search [1], the simplex method of Nelder and Mead [2], and the variable metric method of Fletcher [3]) and they may be "guided" by fixing and restoring variable parameters in between minimization commands, and by putting limits on the values allowed for different parameters. Similarly, error analysis may be carried out using the covariance matrix of the function, or by calculating exact MINOS confidence intervals, or by plotting function contours. If the function is suspected of having more than one local minimum, a global minimization can be attempted using a simplified version of the algorithm of Goldstein and Price [4].

Restriction of the complexity of the problem
The current version is dimensioned for a maximum of 30 function parameters, of which not more than 15 may be variable.

Typical running time
In most cases nearly all the actual running time is spent in the

user-supplied subroutine FCN which is the function being analyzed by MINUIT. Typical applications usually require a few hundred or a few thousand function evaluations, which could in turn require much less than a second or many minutes on the CDC 7600 depending on the complexity of FCN.

Unusual features of the problem

MINUIT is not merely a minimization program, but is more properly a system for analysis of functions in the sense that its essential element is its structure rather than any of the algorithms that are actually implemented. This structure also allows in a very natural way the inclusion of global logic connecting the different algorithms, for example switching from one minimization method to another if the first does not converge rapidly enough.

References

[1] F. James, Monte Carlo for Particle Physicists (Section 6.1) in: Methods in Subnuclear Physics, Ed. M. Nikolic (Gordon and Breach Publ.) Vol. IV, Part 3.
[2] J.A. Nelder and R. Mead, Comput. J. 7 (1965) 308.
[3] R. Fletcher, Comput. J. 13 (1970) 317.
[4] A.A. Goldstein and J.F. Price, Math. Comp. 25 (1971) 569.

LONG WRITE-UP

1. Introduction

A large class of problems in many different fields of research can be reduced to the problem of finding the smallest value taken on by a function of one or more variable parameters. The classic example is the estimation of unknown parameters in a scientific theory by minimizing the difference (chi-square) between theory and experimental data, theory being represented by a function $F(X)$, where X are the unknown parameters of the theory.

The function $F(X)$ need not be known analytically, but it is specified by giving its value at any point X in the space of the parameters. The space may be limited by physical restrictions on the allowed values of the parameters (constrained minimization). Sometimes additional information about the function may be available, such as the numerical values of the derivatives $\partial F/\partial X$ at any point X, or analytic expressions to evaluate them. Minimization always proceeds by evaluating $F(X)$ repeatedly at different points X determined by the minimization algorithm(s) used, until some minimum value is attained.

At that point the user often wants to have an idea about how sensitive the solution is to variations in the parameters: how steeply does $F(X)$ increase away from $X_{\min}$ in X-space? Physicists call this information the "errors of the parameters X", and we shall term in error analysis.

Whenever the function may have more than one local minimum, new problems arise in addition to the problem of local minimization. The user must then decide whether it is sufficient to know the location of any one local minimum, or whether knowledge of the global minimum is required.

Such general considerations on function minimization and descriptions of the most important algorithms may be found in ref. [1].

MINUIT is a system of programs to solve the problems outlined above. As MINUIT is in principle designed to handle any function $F(X)$, it is quite general, and it may suit the needs of quite different users. A user, however, who spends much time minimizing a restricted class of functions, may well be able to write faster minimizing routines by making use of known properties of his functions. For such a user MINUIT may still offer a suitable organizational framework in which to embed his own routines as subroutines.

1.1. Minimization to a local minimum

To the standard user MINUIT offers the possibility within one program of catering to different kinds of functions since it incorporates three different minimization methods, each of which may be used alone or in combination with the others depending on the behaviour of the function and the requirements of the user. The three minimizing subroutines, SEEK, SIMPLX and MIGRAD, may be briefly characterized as follows (for details of theory see section 2):

i) SEEK – a Monte Carlo searching subroutine. It may be used at the beginning of a fit when no reasonable starting point is known, or when it is suspected that there are several minima. It must not, however, be expected to converge in the usual sense.

ii) SIMPLX – a minimizing subroutine using a simplex method by Nelder and Mead [2]. It is very "safe" and reasonably fast when far from minimum, and may also be used to converge to the exact minimum. It does not compute the covariance matrix, but gives order-of-magnitude estimates of its diagonal elements (the parameter errors).

iii) MIGRAD – a minimization subroutine based on a variable metric method by Fletcher [3]. It is extremely fast near a minimum or in any "nearly-quadratic" region but slower if the function is badly behaved. It uses the first derivatives of the function, which may either be supplied by the user or estimated by MINUIT.

Some "global" logic is built into the program. For example, if MIGRAD fails, it automatically causes SIMPLX to be called to make another attempt. In addition, the minimization can be guided or separated into steps by the use of FIX, RELEASE, and RESTORE commands which cause a variable to be fixed at a constant value or restored to variable status in between minimization steps. The program can also be instructed to force the value of any variable parameter(s) to stay within given limits during the minimization(s).

1.2. Error analysis

MINUIT is usually used to minimize a "chisquare" function or a negative log likelihood function. The error analyses assume that the function has such a statistical meaning, so that the inverse of the second derivative matrix can be associated (within a constant factor) with what the scientist calls the "error matrix" of the parameters. More generally, it is assumed that the point where the function takes on its lowest value FMIN determines the most likely or best-fit parameter values, and that the region over which the function takes on values smaller than (FMIN + UP) corresponds to a confidence interval or confidence region for which the confidence level is determined by the value of the positive constant UP. For example, in the simplest case where the function is a "chisquare" function of one variable parameter, a value of UP = 1.0 determines "one-standard-deviation" errors corresponding to a confidence level of 68%. MINUIT performs the following error analyses:

i) The *covariance matrix* (or error matrix) is a by-product of the MIGRAD minimizer. It may however be calculated separately (and independently using second derivatives) by subroutine HESSE. Based on this matrix, the program prints both the individual correlation coefficients and the global correlation coefficient [4] for each variable parameter.

ii) Exact *confidence intervals* [4] for any parameter(s) may be calculated by MINOS. These may differ from the "parabolic" errors derived from the covariance matrix if the model being fitted is highly non-linear.

iii) *Function contours* may be plotted in the space of any two variables at a time. This gives the most detailed description of the shape of the function, but only when the number of variables is very few.

1.3. Global minimum search

The user in quest of the global minimum may not be satisfied with the local minimum found, and he may even be interested in finding all local minima. The problem of global minimization is very complex, and we do not know of any really satisfactory algorithm to solve it. In view of the importance of the problem MINUIT offers the following options:

i) The IMPROVE command invokes a very elegant algorithm of Goldstein and Price [5] which, starting from a known local minimum with known covariance matrix, "removes" this minimum by dividing out the quadratic part and looks for a minimum of the transformed function.

ii) Mapping the function by the command CONTOUR may reveal the existence of further minima, appearing as bulges on the contours.

iii) MINOS can be forced to take arbitrarily large steps away from a local minimum, and since it then minimizes in the subspace of the NPAR-1 remaining variables, it has the capacity of finding new minima (if UP happens to be chosen right). When this happens MINOS gives a signal and the program automatically proceeds to evaluate the properties of the new minimum.

1.4. Internal and external parameters

In order to eliminate the problem of physical

boundaries on parameters, as well as to simplify the organization of the program, two lists of parameters have been created, called internal and external. The external parameter list contains all parameters, fixed and variable, arranged according to the way they are referenced by the user in FCN. Those parameters which are variable have corresponding entries in the internal parameters list, to which they are connected by transformations so that as the internal parameters approach $\pm n\pi/2$ the external values approach their physical limits if bounded. The minimizing subroutines SIMPLX and MIGRAD see only the internal values, which behave smoothly, even near the physical limits. The function FCN sees only the external values, which remain always within the physical limits.

2. MINUIT organization

2.1. History

MINUIT has been under continuous development at CERN since 1967. During this time the program has had enormous exposure to users at CERN and elsewhere. Although MINUIT is a framework capable in principle of offering any number of different algorithms, practical considerations require us to limit the number of possibilities. Over the years we have incorporated several different methods into standard MINUIT versions, and have withdrawn those which compare less favourably in the light of experience. Thus we have replaced the Rosenbrock (TAUROS) method [6] by Nelder and Mead's simplex technique [2] and we have replaced Davidon's rank-one variable-metric method (old MIGRAD) [7] by Fletcher's "switching" variation [3] of the more stable Davidon–Fletcher–Powell [8] variable-metric method (new MIGRAD). This does not mean that functions cannot be found for which our "rejected" methods will perform better than the "accepted" ones, but merely reflects our experience and theoretical understanding.

The present version of MINUIT is dimensioned for a maximum number of free parameters NPAR = 15. Redimensioning for larger problems is straightforward, and indeed another standard version (known as MINUITL) exists for 55 free parameters and requires about twice as much storage.

2.2. General organization

Control of MINUIT resides in the two subroutines MINNEW and COMAND. MINNEW takes care of all the automatic initialization procedures, whereupon it calls COMAND, according to the flow scheme in fig. 1. COMAND takes orders from the user in the form of command cards and executes them.

Below we list the subroutines in alphabetical order, specifying briefly their function. In section 3 we give more details under the description of command cards.

2.3. Functions and subroutines

Subroutine BINSIZ (A1, A2, NAA, BL, BH, NB, SWID)
Subroutine to determine reasonable intervals given absolute upper and lower bounds A1 and A2 and desired maximum number of bins NAA.

Function CALFCN (PVEC)
Called only from IMPROV. Transforms the function FCN by dividing out the quadratic part in order to find further minima. Calculates (F-FMIN)/(X-XMIN) *V*(X-XMIN).

Subroutine COMAND
Reads the command cards and takes appropriate action, either directly by skipping to the corresponding code in COMMAND, or by setting up a call to a subroutine. For command cards, see section 3.2.

Subroutine CONTOU
Finds points lying on contours of a given FCN value, as a function of two variable parameters specified by the CONTOUR command. The contours are then plotted by PLTCON.

Subroutine DERIVE (GG, GG2)
If first derivatives of the function are required (i.e. if MIGRAD is used or called by MINOS) they are calculated by subroutine DERIVE. If they are not supplied by the user, a finite-difference approximation is used, and we have chosen the symmetric formula:

$$\left.\frac{\partial F}{\partial X}\right|_{X_0} \sim \frac{F(X_0+d)-F(X_0-d)}{2d}$$

which requires $2n$ function calls for n parameters, but

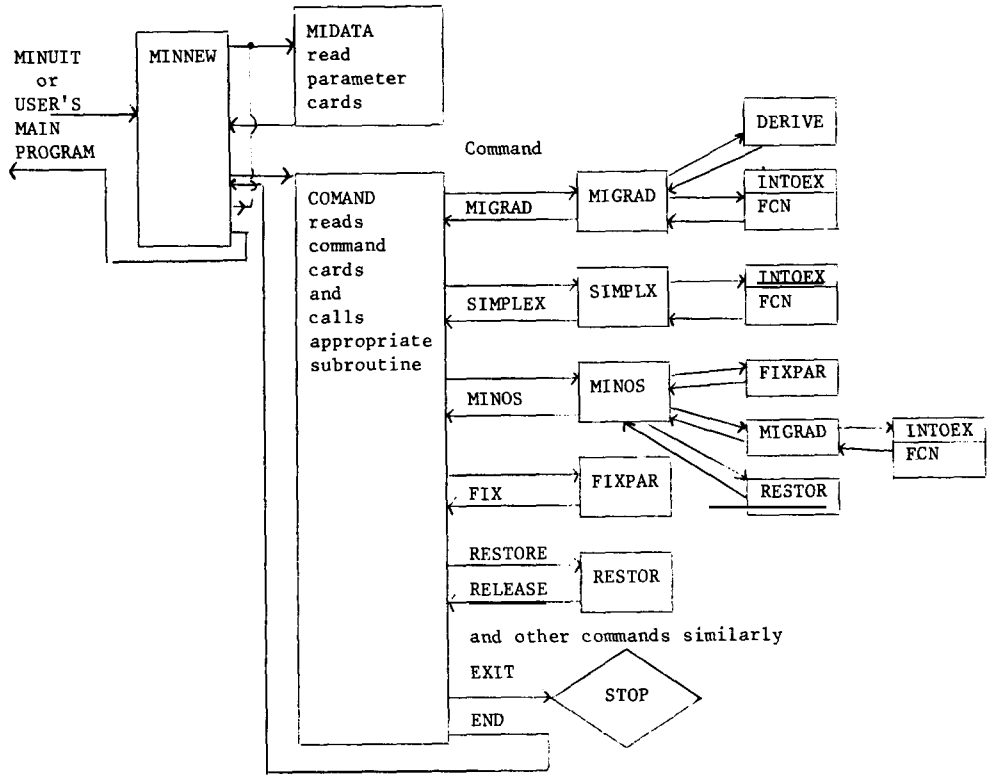


Fig. 1. Simplified flow scheme of MINUIT.

for which the error is, to lowest order, independent of the step-size d. An asymmetric formula requiring only $n+1$ points would result in an error proportional to d. The symmetric formula thus allows us to avoid the difficult problem of choosing d optimally, and makes MIGRAD behave well near the minimum where the first derivatives approach zero. It also offers an estimate of the error on the derivatives as a by-product.

If the user calculates first derivatives inside FCN, then DERIVE simply transforms these derivatives (if necessary) from external to internal coordinates.

Subroutine EXTOIN (PINT)
Transforms the external parameter values X to internal values in the dense array PINT. Function PINTF is used.

Subroutine FCN (NPAR, G, F, X, IFLAG)
User subroutine, see section 4.

Subroutine FIXPAR(I2, KODE, ILAX)
Removes parameter I2 from the internal (variable) parameter list, and arranges the rest of the list to fill the hole. If KODE = 0, I2 is an external number, otherwise internal. ILAX is returned as the external number of the parameter.

Function FTIME (BIDON)
Gives the elapsed job time in floating-point minutes by calling some installation-dependent subroutine.

Subroutine HESSE
Calculates the full second-derivative matrix of FCN by taking finite differences. Includes some safeguards against non-positive-definite matrices, and may set off-diagonal elements to zero in attempt to force positivity.

Subroutine IMPROV
Attempts to improve on a good local minimum by finding a better one. The quadratic part of FCN is

removed by CALFCN and this transformed function is minimized using the SIMPLEX method from several random starting points [5].

Subroutine INTOEX (PINT)
Transforms from internal coordinates (PINT) to external parameters (U). The minimizing routines which work in internal coordinates call this routine before calling FCN.

Subroutine MATOUT (TRACE, KODE)
Prints the covariance matrix V. Calculates and prints the individual correlation coefficients and global correlations.

Subroutine MIDATA
Reads the data cards (title card and parameter cards) and sets up the starting parameter lists. Control then passes to COMAND for reading the command cards.

Subroutine MIGRAD
Performs a local function minimization using a gradient supplied by the user (GRADIENT command) or calculated by subroutine DERIVE.

The algorithm used is Fletcher's "switching" variation [3] of the original Davidon–Fletcher–Powell algorithm [8]. It belongs to the class of variable metric methods which are characterized by the following general approach:

a) Starting values are given for the parameters X(I), the first derivatives GS(I) and the covariance matrix V(I, J). Although GS(I) is supposed to be a very good approximation for the gradient at X(I), the starting matrix V(I, J) may be only a diagonal matrix or even the unit matrix.
b) A "Newton's step" is taken to $X' = X - V*GS$ which would be the minimum if F were quadratic and if V(I, J) were the true covariance matrix. Since X'(I) is not generally the position of the minimum it is usual to perform a linear search along this direction, finding the α which minimizes $F(X - \alpha*V*GS)$. Let this new point be called X'(I), and let the gradient calculated at X'(I) be GS'(I).
c) The matrix V(I, J) is "corrected" using an updating formula of the general form $V' = V + f(V, X, X', GS, GS')$. Then GS is replaced by GS', X by X', and V by V', and steps (a) and (b) are repeated until some convergence criteria are satisfied.

The updating formula used in MIGRAD is either the original Davidon formula [8] or Fletcher's dual formula [3], depending on Fletcher's "switching" criterion [3]. Recent theoretical studies have shown that there is in fact little practical difference between the two formulas (no difference at all in the limit that the linear minimizations are exact), and both formulas are in the Fletcher "convex class" of stable formulas guaranteeing monotonic convergence of V toward the true covariance matrix. (The Davidon rank-one formula of 1968 [7] used in earlier version of MINUIT, is not in the "convex class" and indeed experience has shown it to be highly unstable.)

The linear minimization performed by MIGRAD in the direction of the Newton step is a rather rough one, using at most two points in addition to the Newton point. The gradient is of course required only at the best of these points which becomes the starting point for the next Newton step.

The convergence criteria are based on the "estimated distance to minimum" (EDM), which is just

$$EDM = GS^T*V*GS.$$

This would be exactly twice the vertical distance (F-Fmin) if the function were exactly quadratic with covariance matrix V. Since this estimate is only as good as the estimate of V, an additional criterion is applied, namely that the successive estimates of V be not very different. The measure of difference used is (VTEST = the average fractional change in the diagonal elements). These criteria may be set by the user (see under the MIGRAD command). Default values currently are

$$\text{both: } \begin{cases} EDM < EPSI*UP = 0.1*UP \\ VTEST < 0.01 \end{cases}$$

or

$$\text{alone: } EDM < 10^{-5}*EPSI*UP = 10^{-6}*UP$$

where UP is the error definition constant (default value = 1.0). The convergence criteria used are printed at the beginning of each MIGRAD minimization.

Subroutine MINNEW
This is the main program, disguised as a subroutine for reasons of compatibility between systems. It initializes some constants in COMMON (including the

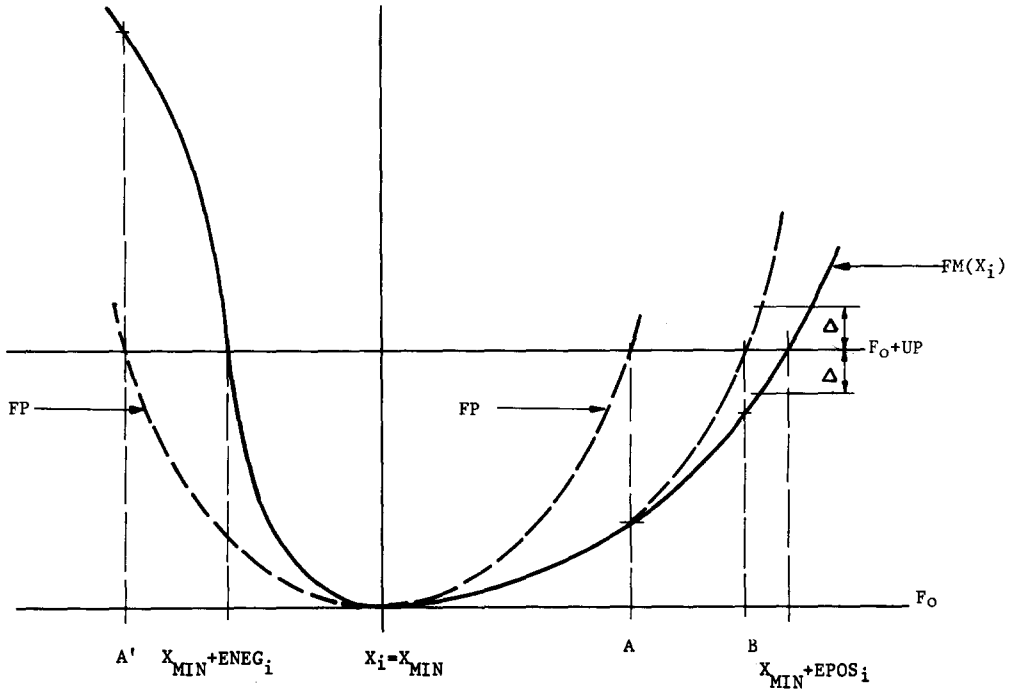


Fig. 2. Calculation of MINOS errors of parameter i. The (symmetric) dotted parabola FP is predicted from the covariance matrix, but the nonlinearity of the problem results in the solid curve FM which gives the asymmetric errors EPOS and ENEG (see text).

logical I/O unit nos.) which would have to be in block data for many compilers. Then verifies that FCN gives the same value when called twice with the same arguments, and passes control to COMAND.

Subroutine MINOS

Finds the true confidence intervals (errors) on the parameters by examining the exact behaviour of the (likelihood or chisquare) function over the interval in question. Only the numerical method will be described here; the theory that the intervals so defined do indeed correspond to confidence intervals is given in reference 4, pages 203–205. We point out here that in the limit that the model being fitted is linear, or in the limit of infinite data (asymptotic statistics), the MINOS errors will be symmetric and equal to the errors derived from the covariance matrix (so-called "parabolic" errors).

The MINOS errors are defined in terms of the size (probability content) of the confidence interval as specified by the user through the ERROR DEF command which sets the value of UP (default value = 1.0). MINOS then determines where the function FM(X_i) attains the value F_0 + UP, where FM(X_i) is the minimum of F with respect to all parameters except the ith parameter X_i, and F_0 is the overall minimum value of F. The MINOS error are then given by these crossing points as indicated in fig. 2 by EPOS$_i$ and ENEG$_i$.

The details of the procedure are as follows: first, on the basis of the known covariance matrix of the function, MINOS predicts that the crossing point will be at A, and it also predicts the values of the other parameters at the minimum FM(A). It then determines FM(A) by using FIXPAR to fix parameter i at X_i = A and using MIGRAD to minimize F with respect to the remaining variable parameters (if any!). If the case is not too pathological, the minimization proceeds very quickly since MINOS supplies MIGRAD with the reduced covariance matrix calculated from the full covariance matrix with parameter i fixed. In the case shown in fig. 2, the prediction A was not very good, and another point B must be tried. This time the starting values for the variable parameters

are chosen by extrapolating from their values at X_{min} through A to B. Point B is chosen by a parabolic extrapolation from A corrected to avoid the well-known instability whereby A could predict B and B could predict A, etc. The starting value for the covariance matrix for MIGRAD at B is the one resulting from the minimization at A. The procedure terminates when one predicted point of FM is within a certain tolerance of F_0 + UP (currently this tolerance is 0.1*UP). When the program has difficulty converging it plots the curve of fig. 2 as determined by all the points attempted so that the user can get a feeling for the nonlinearity of his problem.

Subroutine MPRINT (IKODE, FVAL)
Prints the values of the parameters at the time of the call. Also prints other relevant information such as function value, estimated distance to minimum, parameter errors, step sizes. According to the value of IKODE, the printout is long format, short format, or MINOS format (0, 1, 2).

Subroutine MPUNCH
Punches current parameter values and step sizes onto cards in format which can be reread by MINUIT for restarting. The covariance matrix is also punched if it exists.

Function PINTF (PEXIT, I)
Calculates the internal parameter value PINTF corresponding to the external value PEXT for parameter I.

Subroutine PLTCON (NSPT, SPT)
Plots points in array SPT onto page with labelled axes NSPT is the number of points to be plotted.

Subroutine RAZZIA (YNEW, PNEW)
Called only by SIMPLX (and IMPROV) to add a new point and remove an old one from the current simplex, and get the estimated distance to minimum.

Subroutine RESTOR (K)
Restores a fixed parameter to variable status by inserting it into the internal parameter list at the appropriate place.

Subroutine SEEK
The Monte Carlo technique [9] is used here in preference to the earlier methods of searching over a rectangular grid because of its higher efficiency in many variables. The function $F(X)$ is evaluated by calling FCN a given number of times (specified by the user). For each call, all of the variable parameter values X_i are chosen randomly according to uniform distribution at the best previous values with widths equal to the starting parameter errors $\sigma(X_i)$. When finished, the program prints out the best function value and corresponding parameter values. Depending on the PRINTOUT option chosen, every tenth new minimum, or every minimum may also be printed during the search.

Subroutine SIMPLX
In the minimization method of Nelder and Mead [2] the information about the function $F(X_1, ..., X_n)$ consists in $n+1$ points, forming a simplex. (A simplex is the smallest n-dimensional geometrical figure with $n+1$ corners: a triangle for $n = 2$, a tetrahedron for $n = 3$, etc.). By reflecting one point in the hyperplane of the other points, one continually forms new simplices. The simplex adapts itself to the local function landscape, elongating down long inclined planes, changing direction on encountering a valley at an angle, and contracting in the neighbourhood of a minimum.

In SIMPLX the initial simplex is formed by coordinate variation. From the starting point (read in from cards), the program proceeds to find a local minimum along each coordinate axis. These NPAR local minima and the starting point define the initial simplex. This procedure assures that the simplex has a reasonable size in each direction. (If no minimum is found, the search will not go beyond certain limits, depending on the starting error read in from cards.) Depending on how good $F(P^*)$ is, relative to the known simplex points, the program either replaces P_H by P^*, or evaluates F at a new point P^{**} on the line $(P_H, \bar{P})$. P^{**} may be beyond P^* or between P_H and $\bar{P}$. Depending on how good $F(P^*)$ and $F(P^{**})$ are, relative to the known simplex points, the pro-

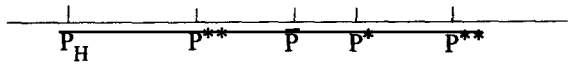


Fig. 3.

are, relative to the known simplex points, the program may now
- replace P_H by P^*, or
- replace P_H by P^{**}, or
- replace P_H by the point where a parabola through P_H, P^* and P^{**} has a minimum, or
- reduce the simplex linearly in every direction by a factor 0.5, keeping only the best point P_L, or
- use the best point P_L as a starting and evaluate a new starting simplex, as described above.

The program always has a rough idea of the parameter errors, of the order of the size of the simplex, and of the vertical distance to the minimum, EDM, of the order of $F(P_H) - F(P_L)$. Convergence is considered to be attained when EDM = $F(P_H) - F(P_L)$ < EPSI.

In the last step, SIMPLX evaluates $F(\bar{P})$, which should be smaller than $F(P_L)$. The minimum value of the function is then min $F(P_L), F(\bar{P})$, and the last EDM printed is

$$\text{EDM} = F(P_H) - \min[F(P_L), F(\bar{P})] .$$

If it then turns out that EDM > 2 EPSI, the minimization recommences.

MIGRAD of course has the advantage of producing a full covariance matrix, whereas SIMPLX only gives estimates (which may be poor) of the diagonal elements. Thus the choice between MIGRAD and SIMPLX depends on what information one requires. On the other hand, a good estimate of the covariance matrix can usually be obtained after SIMPLX simply by the command MATOUT or HESSE.

SIMPLX does not use derivatives. The diagonal error matrix which results from SIMPLX is not considered sufficient for use in MINOS.

Subroutine STAND
Optional user-supplied subroutine is called whenever the command "STANDARD" appears. See section 4.

Subroutine UCOPY (FROM, TO, N)
Copies N words from array FROM to array TO.

Subroutine VERMIN (A, L, M, N, IFAIL)
Inverts a symmetric matrix. The matrix is first scaled to have all ones on the diagonal (equivalent to change of units) but no pivoting is done since the matrix is positive-definite.

2.4. Internal storage of parameters and errors

The user may find it convenient to have direct access to some of the COMMON blocks used internally by MINUIT to store lists of parameter values, errors, etc. We therefore list below the contents of the blocks most likely to be of use.
In MINUITS:
COMMON blocks
/PAREXT/ U(30), NAM(30), WERR(30), MAXEXT, NU
/VARIAN/ V(15, 15)
/MINERR/ ERP(30), ERN(30)
contain:
U(I) = value of external parameter I
NAM(I) = name (A10 format)
WERR(I) = parabolic error
ERP(I) = positive MINOS error if > 0
ERN(I) = negative MINOS error if < 0
V(J, K) = internal covariance matrix element of internal parameters J and K
MAXEXT = 30
NU = highest external parameter used.

Warnings:
1. The contents of these COMMON blocks must not be changed by the user.
2. The contents are constantly changing during minimizations and may at any time be undefined, meaningless, or inconsistent. The safest time to use them is when IFLAG = 3.

3. Data cards and command cards

3.1. Data cards

The order of data cards is as follows:
1. Title card.
2. Parameter definition cards, one card per parameter.
3. Blank card.
4. Cards read by FCN, if any.
5. Command cards, controlling the actions taken by MINUIT.

The above sequence of cards constitutes one data block, and as many data blocks as desired may be executed, one after the other (using the same function FCN). A more detailed description of these cards

TABLE 1

MINUIT COMMANDS

(D.V. = default value)

COLUMNS	1 - 10	11 - 20	21 - 30	31 - 40	40 -
PURPOSE	COMMAND	ARG 1	ARG 2	ARG 3	ARGS 4 - 7
Causes a call to FCN	CALL_FCN	IFLAG			
Read in covariance matrix from cards	COVARIANCE	NPAR followed by Covariance matrix elements			
Traces contours of constant FCN value	CONTOUR	Param No. (X-AXIS)	Param No. (Y-AXIS)	No.of contours (D.V.=2)	
Signals end of data block	END	(See text)			
Signals end of data block. Control returns to main program	END_RETURN	(see text)			
Sets value of UP, defining errors	ERROR_DEF	UP(D.V.=1.0)			
Signals end of run	EXIT	(see text)			
Makes a param.constant	FIX	Extern.Param No.			
Indicates derivatives calculated by FCN	GRADIENT	(see text)			
Estimates and prints covariance matrix	HESSE	Max.No.calls (D.V.=1000)			
If minimum is found, looks for another one	IMPROVE	Max.No.calls (D.V.=1000)	No.of tries (D.V.=NPAR+2)		
Prints covariance matrix	MATOUT				
Does a MIGRAD minimization	MIGRAD	Max.No.calls (D.V.=1000)	Tolerance (D.V.=0.1)	VTEST (D.V.=0.1)	
Minimize FCN (=SIMPLEX + MIGRAD)	MINIMIZE	Max.No.calls (D.V.=1000)	(D.V.=0.1)		
Perform MINOS error analysis	MINOS	Max.No.calls (D.V.=1000)	Param. No. (see text)		
Sets ISW(5) controlling printout	PRINTOUT	Printout level (D.V.=1)			
Causes parameters & error matrix to be punched on cards	PUNCH				
Restores one or more previously fixed parameters to variable status	RELEASE	Extern.param.No.	External param. No.	External param. No.	External param. No etc
	RESTORE	= 0 for all parameters = 1 for last param. fixed			
Does Monte Carlo minimization	SEEK	No.calls to FCN			
Does a SIMPLX minimization	SIMPLEX	Max.No.calls (D.V.=1000)	Tolerance (D.V.=0.1)		
Calls subroutine STAND	STANDARD	Parameters available to STAND through COMMON block			

follows:

1. *Title card* – Any alphanumeric characters, serving as the title for the printout.
2. *Parameter cards* are in the format:
   Col. 1–10 Parameter number as referenced in FCN ($\leqslant 30$)
   Col. 11–20 Alphanumeric name for the parameter
   Col. 21–30 Starting value
   Col. 31–40 Approximate error or step size (if zero, parameter is constant)
   Col. 41–50 Lower bound on parameter
   Col. 51–60 Upper bound on parameter
   } if *both* blank, not bounded.

   Remaining columns are not used. All numbers may be punched either as right-adjusted integers or with decimal point. Not more than 15 parameters may be variable.
3. One *blank card* signals end of parameter cards.
4. Next come the *cards read by FCN,* if any. These may be followed by one or more blank cards if desired.
5. The *COMMAND cards* are all in the format A10, 7F10.0, that is, left-adjusted ten-letter command, followed by several numeric arguments which may be punched either as right-adjusted integers or as floating point.

3.2. Command cards

Each of the commands recognized by MINUIT is described in this section in some detail. Since most commands correspond to MINUIT subroutines, additional information on these commands may sometimes be found in section 2.2, and indeed the separation between commands and subroutines is necessarily somewhat arbitrary.

Table 1 contains a summary of the recognized commands and their formats.

Each command, as it is read, is stored temporarily, with its arguments in COMMON/CARD/ as follows:

```
COMMON /CARD/ CWORD, CWORD2, CWORD3,
  WORD7(7)
```

where

CWORD = the first four characters of the command (A4)
CWORD2 = the second four characters of the command (A4)
CWORD3 = the last two characters of the command (A2)
WORD7(I) = the Ith argument in floating point.

The contents of this COMMON block always refer to the last command card read. (This COMMON block is not normally of interest to the user.)

CALL FCN Command

This causes a call to FCN, with a given value of IFLAG indicated as the first argument of the command. Example

```
CALL FCN          3 .
```

The above example causes FCN to be called with IFLAG = 3. This is usually used to indicate to FCN that it should print our final curves, tables, etc. If the above command *does not* appear explicitly before the end of a block of command cards, an automatic call to FCN with IFLAG = 3 is made upon encountering the END or EXIT command (see description of END and EXIT).

MINUIT uses only IFLAG = 1, 2, 3, and 4 and reserves 5 for possible future use. Any other value of IFLAG may be used to cause any special action desired by the user in FCN. In order to permit the greatest possible flexibility, it is allowed in FCN, when IFLAG is greater than 5, to change any parameter values or internal constants (but *not* NPAR) which affect the value of the function (such changes are not normally permitted for obvious reasons). Thus a command CALL FCN with IFLAG greater than 5, causes MINUIT to "forget" where it was before, to call FCN again with IFLAG = 4 in order to get the new function value, and to arrange internal parameter lists to take account of any possible parameter changes. (Note that the *number* of variable parameters (NPAR) can only be changed through RELEASE, FIX and RESTORE commands.) Example of possible use of CALL FCN command: Suppose it is desired to fit a theoretical model to a mass spectrum between 600 and 800 MeV, then to use the resulting parameter values as starting values for a similar fit over the range 500 to 900 MeV. A possible command sequence is:

```
PRINTOUT   2
CALL FCN   8
```

```
MIGRAD
CALL FCN   3
CALL FCN   9
MIGRAD
CALL FCN   3
EXIT
```

Inside FCN, when IFLAG = 3, fitted curves would be printed as usual. When IFLAG = 8, the mass range for the fit would be set at 600–800 MeV, and when IFLAG = 9, the range would be set to 500–900 MeV.

CONTOUR Command
The CONTOUR command has the following format:

CONTOUR i j k

where i and j are parameter numbers and k is the desired number of contours. This command causes the program to trace contours of constant value of FCN as a function of the two parameters i and j (all others being fixed at their value at that time). It is assumed that FCN has already been minimized, and it is required that a covariance matrix exist, so that CONTOUR will normally be preceded by a MIGRAD command. Parameters i and j must be variable parameters at the time of the command. If $k \leqslant 0$, it is taken = 2. If $k > 5$, it is taken = 5.

The first (innermost) contour corresponds to an FCN value F = FMIN + UP, where FMIN is the current value (assumed to be a local minimum) and UP is the error definition previously specified (see B–230). If more than one contour is requested, then Nth contour corresponds to F = FMIN + UP*N**2. The resulting contours are plotted on one page using the line printer, but the actual coordinates of the points are printed only if the PRINTOUT level $\geqslant 2$. (See PRINTOUT command.)

COVARIANCE Command
This causes the program to read in a covariance matrix from cards, to be used in MIGRAD minimization or MINOS error analysis. The first argument gives the number of parameters to which the matrix corresponds, that is the square root of the number of matrix elements to be read. Example:

COVARIANCE	2		
21.443	17.212	17.212	46.28

The matrix elements are read in FORMAT (7 F10.0), but normally one need not worry about the format since one uses only cards which are punched by previous MINUIT runs, which are of course punched in the proper format. Since the matrix *must* be symmetric, it does not matter whether it is read by rows or by columns. It is, however, important that the matrix should correspond to the parameters which are ***variable at the time that the matrix is read in.***

END Command
This command is the last command in a data block, but it implies that there is at least one more data block to follow. The end of the *last* data block should be signalled by the command EXIT, without a command END. When the END card is encountered, before going to the next data block, the program asks whether a command CALL FCN 3 3 has yet been made in this data block. If not, an automatic call to FCN will be suppressed if the first argument of the END command is $\geqslant 1$.

END-RETURN Command
This command causes control to return to the main program, and should be used only when the main program has been written by the user. This option allows MINUIT to be used as a subroutine called by the user's main program (see fig. 1). Example:

```
CALL PRIVAT
CALL MINNEW
CALL MYCALC
CALL MINNEW
CALL MYCALC
STOP
END
```

Each call to MINNEW causes the processing of one full data block which should be terminated by the END RETURN command. The other calls in this example perform private calculations.

EXIT Command
This is exactly the same as the END Command, except that it signals the end of the last data block.

ERROR DEF Command
This card serves to define what the user means by the

errors on the parameters. The first argument of this command specifies the quantity UP, which is the change in the function value produced by changing a parameter by one "error". Example:

ERROR DEF 0.5

The above card would be used for example, if the FCN was the negative of a log likelihood function in one parameter and the user wanted the 68% confidence interval. For a chisquared function, the value of UP would be twice as great.

If no ERROR DEF command appears, UP is taken to be equal to one.

Since UP is used by SIMPLEX, MIGRAD, MINOS, CONTOUR, IMPROVE, and PUNCH, the ERROR DEF command should precede these commands if UP is different from one.

The error definition may of course be changed during a run to get MINOS errors or contours corresponding to different numbers of standard deviations. For example:

```
ERROR DEF    1.0
MIGRAD
MINOS
ERROR DEF    4.0
MINOS
EXIT
```

FIX Command

This command causes a parameter to be removed from the variable list and its value to be fixed at the value it has when the command is executed. The number of variable parameters (NPAR) is reduced by one. If a covariance matrix exists at the time of the execution of the FIX command, it is properly reduced (in general all elements are modified), but the new matrix is not automatically printed (the MATOUT command may of course be used). Although the parameter fixed may later be restored to variable status by a RELEASE or a RESTORE command, it is important to note that the information in the covariance matrix concerning this parameter is irretrievably lost.

The first argument of the FIX command is the (external) parameter number of the parameter to be fixed. For example:

```
|FIX     |     4|
```

The above command causes parameter No. 4 to be fixed. If this parameter is already constant or not defined, the command is ignored and a short error message is printed.

If, at the time of execution of a FIX command, a covariance matrix exists which is not positive definite, the matrix will be destroyed by the FIX command, and this condition will be signalled to the rest of the program through ISW(2) = 0.

Not more than 15 parameters may be FIXed at any one time.

GRADIENT Command

This command, which has one argument, indicates that FCN is prepared to calculate its own derivatives. The main program then assumes that when FCN is called with IFLAG = 2, all the first partial derivatives of FCN with respect to all variable parameters will be calculated by FCN and returned in the vector G which is the second formal argument of the Subroutine FCN. In addition, of course, the function value must be calculated as for *all* FCN calls. When FCN is called with IFLAG $\neq$ 2, the gradient need not be calculated. Only Subroutine MIGRAD uses the derivatives of FCN, and the GRADIENT command therefore has no effect if no MIGRAD or MINOS commands are present.

If the argument of the GRADIENT command is zero or blank, the derivative calculation in FCN is compared with a numerical calculation by finite differences, and the results of the comparison are printed. If the values of the derivatives do not agree within the errors on the numerical estimation, MINUIT does not accept the derivatives from FCN.

In order to suppress the above test (when derivatives are known to be calculated correctly), use the command

```
GRADIENT   1.0
```

HESSE Command

The command causes the covariance matrix to be calculated and printed. If a covariance matrix already exists from previous calculations, it is destroyed and replaced by the newly calculated matrix.

The calculation requires about $\frac{1}{2}$ NPAR**2 calls to FCN (it does not use derivatives from FCN) to determine all the mixed partial second derivatives, and then inverts the resulting matrix. If a diagonal second deriva-

tive is negative, the calculation is stopped with an error message, since there is then no meaningful way to force the matrix to be positive-definite. If the square of an off-diagonal second derivative is bigger than the product of the corresponding diagonal elements, the off-diagonal element is set to zero and a message is printed. Some other crude checks for positive definiteness are made, but there is no absolute guarantee that the resulting covariance matrix is positive-definite. See also MATOUT command.

IMPROVE Command
This command causes the program to try to jump out of a good local minimum in order to find a better (lower) minimum. It should be used only after convergence to a local minimum with the covariance matrix already determined (i.e., usually the IMPROVE command will follow a MIGRAD command).

The method used is due to Goldstein and Price [5]. It consists essentially of transforming the function by dividing it by its quadratic part (i.e. "removing" the local minimum) and seeking a minimum of the transformed function.

If IMPROVE fails to find a new minimum or finds a local minimum not as good as the original, it tries again starting with a new (random) search direction. If it does not find an improved minimum, it continues searching until it exceeds either the maximum no. of FCN calls or the maximum no. of searches as specified by the user on the command card. Example:

```
IMPROVE      1500.      8.
```

This causes IMPROVE to stop after 1500. FCN calls or 8. searches. If these arguments are blank, standard values of 1000 and NPAR+2 are assumed.

The subroutine considers itself successful only if it finds an improvement on the original minimum where the new minimum is at least 0.1 standard deviations away from the previous minimum in at least one variable. In this case it prints the message "Improve has found a truly new minimum".

The message "Improve has returned to region of original minimum" means that either (1) no minimum was found, or (2) a new minimum was not as deep as the original one, or (3) a new minimum was very close to the original so is in fact just a better point in the same hole. In case (2) and sometimes in case (1), a printout will give the point where the program stopped searching. If this point is far away from the original minimum, it usually indicates the existence of a secondary minimum in the neighbourhood of the stopping point.

MATOUT Command
This command causes the printing of the parameter value and all the individual correlation coefficients normalized off-diagonal elements of the covariance matrix V_{ij}/SQRT (V_{ii}*V_{jj}) and the global correlation coefficients

$$\rho_k^2 = 1 - [V_{kk} * (V^{-1})_{kk}]^{-1}.$$

In addition, if the printout level is greater than one, the full (internal) covariance matrix is printed.

When the covariance matrix does not exist, or is not meaningful, it is calculated by subroutine HESSE. (See HESSE command.)

MIGRAD Command
This command causes the program to perform a minimization using the MIGRAD technique, which is described under subroutine MIGRAD. The command may have three arguments controlling the convergence. They are:

Arg 1: NFCNMX – Maximum number of calls to FCN. The minimization will be stopped after this number of calls, even if convergence is not attained. If this argument is blank, zero, or negative, NFCNMX is set equal to 1000. The number of FCN calls is checked against NFCMX only once per iteration; therefore NFCNMX may be somewhat exceeded.

Arg. 2: EPSI – Tolerance on the minimum function value. The program has the ability to predict how far it is (vertically) from the true minimum, based on the covariance matrix and the first derivatives of FCN. When this predicted distance ρ becomes smaller than EPSI for two consecutive iterations, this convergence criterion is satisfied. If argument is blank or zero, EPSI is set equal to 0.1*UP. (See ERROR DEF command.)

Arg. 3: VTEST – Tolerance on the stability of the error matrix. The program calculates the average fractional change in the diagonal elements of the covariance matrix from one iteration to the next. When this quantity is smaller than

VTEST for two consecutive iterations, this convergence criterion is satisfied. If argument is blank or zero, VTEST is set equal to 0.01.

Convergence is attained when both the EPSI and VTEST criteria are satisfied simultaneously, or when ρ (defined above) becomes smaller than 10^{-5}*EPSI.

MINIMIZE Command

This command causes the program to attempt to minimize FCN by calling first SIMPLX and then MIGRAD. The arguments are the same as for SIMPLEX, namely:

First argument: maximum number of FCN calls (default value 1000).

Second argument: tolerance on FCN value (default value 0.1).

The limit on the number of FCN calls is applied globally to SIMPLX and MIGRAD together.

MINOS Command

This command causes a MINOS error analysis to be performed on all or certain indicated parameters. For details on the meaning of MINOS errors, see subroutine MINOS. MINOS in turn calls either MIGRAD or SIMPLX, depending on how good the initial error estimate is. (MINOS cannot operate unless at least an approximate covariance matrix exists. When not, MINOS writes a message. It is then advisable to precede the MINOS command by a MIGRAD command.)

The first argument of the MINOS command gives NFCNMX, the limit of the number of FCN calls which may be made during the MINOS analysis of *each* parameter. When this limit is exceeded, MINOS goes on to the next parameter even if it has not finished the analysis for the current parameter. If this first argument is blank or zero, it is taken as 1000.

Starting in column 21 of the MINOS command card, a *special format* is used, for it is here that are punched the parameter numbers of those parameters for which a MINOS analysis is desired. If nothing is punched after column 20, MINOS is performed for all parameters. The special format is 3012, that is, a one-digit parameter number should be punched in an even-numbered column, and a two-digit number should start in an odd-numbered column. All blanks and zeros are *ignored*. Example:

```
|MINOS       |       300. |blb3bb22b4|
```

The above card would result in MINOS analysis for parameters numbered 1, 3, 22 and 4 (in that order), with a limit of 300 calls for the analysis of each parameter.

An ERROR DEF Command must appear *before* the MINOS command if it is desired that UP$\neq$1.

If a new minimum is found during MINOS error analysis, a new MIGRAD minimization (in the full variable space) is attempted, with

NFCNMX = 4 * NFCNMX
VTEST = 0.5 * VTEST

After this step the MINOS error analysis recommences from the beginning (with the original NFCNMX). If a new minimum is found a second time, a final MIGRAD minimization is attempted (without changes in NFCNMX or VTEST).

PRINTOUT Command

This command by itself does not cause any printing, but it sets the value of ISW(5) whicy is used internally in other subroutines to indicate the amount of printout desired. This value is the first argument. Example:

```
|PRINTOUT    |     3|
```

ISW(5) can take on values between zero and five inclusive, and in general the higher the value, the more printout is generated during the minimization.

Before this card appears, the printout level is taken as = 1.

General guidelines on printout levels are:

0 = absolute minimum, but usually enough
1 = normal printout
2 = extra printout, often useful for difficult cases
>2 = very voluminous, usually used only for debugging MINUIT itself.

PUNCH Command

This command causes the current parameter names, values, errors and limits to be punched on cards in the same format as for input data. These cards can then be used in future runs for restarting, continuing or modifying the minimization procedure.

In addition, if a covariance matrix exists at the time the PUNCH command is executed, this covariance matrix is also punched in suitable format to be used in continuing or modifying the minimization procedure in future runs.

RELEASE Command
This command causes a fixed parameter to become variable again. It can only operate on parameters which were originally variable and which have been fixed by a FIX command. On the command card the user only has to specify the external numbers of the fixed parameters to be released. If the parameter requested is not fixed, the command is ignored and an error message is printed. If a covariance matrix exists at the time of execution of a RELEASE command it is destroyed. (See also COMMAND RESTORE.)

RESTORE Command
This command is similar to RELEASE, except that the user does not specify which parameter is to be released. According to whether the argument of the RESTORE command is zero or one, either all fixed parameters, or the last one fixed, are restored to variable status. Example:

```
FIX       3.
FIX       7.
FIX      11.
MIGRAD
RESTORE   1.   (restores parameter 11)
MIGRAD
RESTORE   0.   (restores parameters 7 and 3)
```

SEEK Command
This command causes a Monte Carlo search of FCN. (For details on this procedure, see subroutine SEEK.)

The first argument is the number of FCN calls to be made during the search. For each FCN call from SEEK, all variable parameters are chosen randomly according to uniform distributions centered at the best parameter values with widths equal to the previous step sizes (or equal to the approximate errors given on the data cards if no previous minimizations were done). At the end of SEEK, the parameters are set equal to the best values found.

SIMPLEX Command
This command causes the program to perform a minimization using the technique described under subroutine SIMPLX. The command may have two arguments controlling the convergence. They are:

Arg. 1: NFCNMX – Maximum number of calls to FCN. The minimization will be stopped after this number of calls, even if convergence is not attained. If this argument is blank, zero or negative, NFCNMX is set equal to 1000.

Arg. 2: EPSI – Tolerance on the minimum function value. If blank or zero, it is set to 0.1*UP. (See ERROR DEF command.)

Convergence is attained when the function values at the NPAR+1 simplex points differ by less than EPSI.

STANDARD Command
This command causes MINUIT to call Subroutine STAND. This subroutine must be supplied by the user, and it may be used to perform any desired actions.

A limited amount of information may be transmitted to STAND using the arguments of the STANDARD command. These arguments are automatically stored in COMMON block /CARD/, which is accessible also to STAND.

3.3. Using MINUIT – Some sample command sequences

In this section we give some idea of the possibilities available with MINUIT by suggesting some command sequences suitable for certain types of problems.

Example 1: A simple problem
The user has a chisquared function which is a fit to a mass histogram. The variable parameters are masses, widths and percentages of resonances, all of which can roughly be estimated by eye, so that reasonable starting values are known for the parameters. An estimate of the parameter errors is desired, but it is suspected that more fits will be done later, so that no time should be wasted calculating exact errors with MINOS. The command cards required are:

```
MINIMIZE
PUNCH
EXIT
```

With the above command, printout level one is assumed (short printout) and errors are evaluated using UP = 1.0 (function increment defining errors) since no ERROR DEF command is present. The function will be minimized using first SIMPLEX and then MIGRAD. Since the call limit is not specified, a limit of 1000 FCN calls will be placed on the minimization. The PUNCH command will punch the last parameter values found, even if the minimizing failed to converge, so that it may be restarted at a better point. When the EXIT card is en-

countered, FCN is called with IFLAG = 3 since no CALL FCN 3 command has appeared.

Example 2: A simple but more difficult problem
The user has a log likelihood function which is a fit over all of phase space. The variable parameters are complex reaction amplitudes for which no reasonable starting values can be estimated. As in example 1, only minimization is desired, but in the present example, this is expected to be more difficult. Suggested commands are:

```
ERROR DEF     0.5
PRINTOUT      2.
SEEK        100.
SIMPLEX     300.      1.
PUNCH
MIGRAD
PUNCH
EXIT
```

The first command sets UP = 0.5 since this corresponds to one standard deviation for likelihood. Since we wish to follow the progress of the fit, but do not wish too much printout, we set the printout level to two, which reports every tenth new minimum in SIMPLEX and MIGRAD and also gives some extra information in SEEK. Then we ask for 100 Monte Carlo calls to get a reasonable starting point. Then SIMPLEX is called with weak convergence and a limit of 300 FCN calls, to get us to the region of the minimum. At this point, we punch the current parameter values just in case the program is cut for time limit later on. Then MIGRAD is called for full normal convergence and error matrix, which are punched for further use, and FCN is automatically called with IFLAG = 3 by EXIT.

Alternatively one may wish to have the IFLAG = 3 printout already after SIMPLEX in case the program is cut in MIGRAD for time limit. The command card sequence then is

```
ERROR DEF     0.5
PRINTOUT      2.
SEEK        100.
SIMPLEX     300.    1.
CALL FCN      3
PUNCH
MIGRAD
CALL FCN      3
PUNCH
EXIT
```

The second call to FCN with IFLAG = 3 is now necessary, because the EXIT command no longer calls FCN automatically, since it already has been called once.

Example 3: A continuing investigation
The user's FCN has already been minimized in a previous run, but it gave a surprizingly high value for parameter 3. It is desired to see if there is another minimum with parameter 3 closer to its starting value, and also to investigate the behaviour of parameter 3 by a MINOS error analysis and by tracing FCN contours as a function of parameters 3 and 5. A reasonable command sequence is:

```
FIX           3.
MINIMIZE
RELEASE       3.
MIGRAD
IMPROVE     200.
MINOS       500.          03
CONTOUR       3.      5.          2.
EXIT
```

4. User subroutines

The principal subroutine which must be supplied by the user is FCN, since it is in FCN that the function to be minimized (F) must be calculated.

In addition the user may supply (optional) subroutine STAND which is meant to take the place of a sequence of COMMAND cards or to perform any operations which are not performed automatically by COMMAND cards. That is, it allows the possibility of directing MINUIT from a FORTRAN subroutine instead of using COMMAND cards. Subroutine STAND is called whenever the COMMAND card STANDARD appears.

4.1. Subroutine FCN(NPAR, G, F, X, IFLAG)

Subroutine FCN calculates the value of the function to be minimized or studied. The arguments have the following meaning (the actual names are of course dummies and may be changed):

NPAR*	The number of variable parameters (at most 15).
G	A vector into which the derivatives are to be put.
F	The function value calculated in FCN.
X*	A vector containing the external parameter values.
IFLAG*	A marker whose meaning is described below.

The arguments marked with * above are input to FCN and must not be changed by FCN (that is, they must never appear on the left side of an equals sign). The marker IFLAG instructs FCN on what it should do at the particular call in question, according to the definitions given in table 2.

Table 2

Value of IFLAG	Calculation or operation to be performed by FCN
1.	Initializing entry. Read in all necessary special data to FCN, calculate constants, print input tables if desired, etc.
2.	Normal entry with gradient. Calculate the derivatives in vector G and the function value in F at the point X.
3.	Terminating entry. Write out any special summaries, output tables, etc. for the minimum point.
4.	Normal entry without gradient. Calculate only the function value F at point X.
5.	Not used (reserved).
Other	Follow any special procedure according to the user's own private convention. (The parameter values may be changed by the user whenever IFLAG > 5 according to the convention given under the CALL FCN command.)

Since most of the computing time is spent inside FCN, it should be carefully optimized for calles when IFLAG = 4 (or = 2 if derivatives are calculated).

For an example of the structure of a typical FCN, see below.

4.2. FCN structure

For the example below we suppose that F is a function of

i) adjustable parameters contained in the vector X
ii) constants C
iii) constant vectors E(I), the use of which imply loops over I.

In order to make FCN as fast as possible, no constants C or vectors E(I) must be computed more than once. In particular one should avoid unnecessary loops over I.

Note that the user may (but need not) calculate the gradient G of the function F.

```
      SUBROUTINE FCN (NPAR, G, F, X, FLAG)
      DIMENSION G(15), X(15)
      COMMON/DUMMY (see note below)
      GO TO (10, 20, 30, 40, 50, 60)IFLAG
10    CONTINUE
      [ Read data cards.
        Set the constants C.
        Compute function of C.
        Compute function of C and E(I).]
20    CONTINUE
      [ Compute function of X and C needed for G.
        Compute functions of X, C and E(I) needed for G.
        Compute G.]
40    CONTINUE
      [ Compute functions of X and C needed for F.
        Compute functions of X, C and E(I) needed for F.
        Compute F.]
      RETURN
30    CONTINUE
      [ Execute all operations which require the parameter
      values X at the minimum of F. For instance, print
      out final tables, plot curves punch cards, etc.]
      GO TO 40
50    [ Do not use]
60    [ Follow any special procedure according to own con-
        ventions]
      RETURN
      END
```

Note that the constants C, E(I) and variables X must not be changed under 20 or 40. The variables X are changed in other parts of MINUIT, and any other changes will hinder MINUIT to find the minimum.

4.3. Unphysical regions

The variable parameters may be assigned physical limits outside of which they will not be allowed to

vary. (It is advised to allow for rounding errors of $\sim 10^{-8}$ in setting physical limits.)

If the problems involve "non-rectangular" constraints (i.e. constraints which cannot be expressed by imposing simple independent limits of the form $A < X(I) < B$), the user must devise a way to "fool" MINUIT into thinking it has an unconstrained problem. The recommended way to do this is by the *penalty function* method as follows:

Suppose we wish to minimize the function f(X) subject to the condition that $g(X) > 0$. Then define the FCN value F to be:

$$F = f(X) \text{ if } g(X) > 0,$$

$$F = f(X) + ag^2(X) \text{ if } g(X) < 0,$$

where a is a constant, large compared with f. Note that this method requires $f(X)$ to be defined everywhere, and continuous at $g(X) = 0$, which may sometimes be difficult to arrange, but otherwise the method is usually found to work well and is perfectly general (i.e. independent of the method used for minimization).

5. Output formats

The printed output should be self-explanatory, but a few comments may be in order.

5.1. Data card printout (subroutine MIDATA)

The starting values and step sizes for the parameters are printed in FORMAT F15.6, so that in some cases, perfectly good data may overflow or underflow this format, but values between 10^{-3} and 10^{9} will be printed in an easy-to-read format. In any case, the values *used* by the program are those punched on the card, while the values printed by the program may be different due to truncation by the format.

Self-explanatory diagnostic messages are printed whenever fatal or non-fatal errors are detected. If one or more fatal errors occur, the program stops, but first reads all the data cards if possible in order to find further errors.

5.2. Command card printout (subroutine COMMAND)

All COMMAND cards are printed as soon as they are read, before being acted upon. This printout can always be distinguished by three short rows of asterisks at the extreme left-hand side of the page, followed at the right by the Command and its arguments. The arguments may be punched as floating-point numbers or as right-adjusted integers, but they are immediately transformed always to floating-point numbers and are printed out in FORMAT F15.4. All arguments which are not printed are blank or zero. If a COMMAND is not recognized by MINUIT (i.e. if it is a comment or if it is misspelled) it is ignored and the words COMMAND IGNORED are printed under the COMMAND.

If mistakes on the MINOS command card are found (i.e. some specified parameter number is outside the allowed range or corresponds to a fixed parameter) the rest of the card is treated normally and a warning message is printed indicating simply that there are some mistakes.

Note that the printed Command arguments are taken directly from the Command card *before* any default values have been assumed for these arguments.

5.3. Normal parameter printout (subroutine MPRINT)

For each printout, the current value of the function is printed, as well as the total number of FCN calls since the beginning of the data black, the total time in minutes since the beginning of the job, and the current estimate of the vertical distance to the minimum (EDM) if such an estimate exists.

For each parameter, the following quantities are printed:

1. internal parameter number – only variable parameters have internal numbers.
2. external parameter number – as referenced by FCN.
3. parameter name appearing on parameter data card.
4. external parameter value – this is the real value of the parameter as used inside FCN.
5. external parameter error – this is the current best estimate of the real parameter error, taking account of UP (see ERROR DEF command). If a covariance matrix exists, the error is calculated inside MPRINT from the appropriate diagonal element.
6. internal parameter value – this is the value of the parameter transformed for internal use (see section 4C). If the parameter is unbounded, it is the same as the external value.
7. internal step size – at the beginning of the program,

this is the starting error read from data cards, transformed into internal coordinates. During or after a minimization, it is equal to the last step taken. Internal values are always used since these are the values used directly by the minimizing routines. This quantity is not defined in SEEK.

At the end of a MINOS error analysis, a block of printout is produced in Subroutine MINOS, which assembles all the principal results from the minimization and error analysis in handy form. If a MINOS error is zero, it means either that the MINOS error was not requested, or that MINOS has failed to find the true error (usually because it lies outside the allowed region or because too many FCN calls are required).

5.4. Covariance matrix and error correlation matrix printout (subroutine MATOUT)

When the covariance matrix is printed, it is always in *internal* coordinates (see 4C), which are the same as the real external parameter values only if there are no limits on the parameters. The matrix is always symmetric, so the elements above the diagonal are not printed.

Unfortunately, the internal covariance matrix used by MINUIT is not usually of direct interest to the user. For this reason, the following interesting numbers which can be derived from the covariance matrix are printed separately:

1. The square roots of the diagonal elements, transformed to external coordinates if the parameters are bounded, and scaled by the error definition UP, are the parameter errors appearing in the parameter printout.
2. The off-diagonal elements give the *correlation coefficients* ρ between parameters from

   $$\rho_{ij} = V_{ij}/\sqrt{V_{ii}V_{jj}},$$

   where V is the covariance matrix in internal *or* external coordinates. If V is positive-definite, $|\rho| < 1$ for all elements. If $\rho = 0$, the two parameters are uncorrelated, and if $|\rho| = 1$, the two parameters are completely correlated.
3. *The global correlation coefficient* [4] for a given parameter is the correlation between it and that linear combination of the other parameters most highly correlated with it. All such coefficients should be between zero and one for a positive-definite covariance matrix.

5.5. CONTOUR printout

Before each contour is calculated, the specifications of the contour are printed. After the contour is finished, a message is printed stating that the calculation was successful (and giving the number of points determined on the contour) or giving the reason for failure if unseccessful.

If the printout level is $\geqslant 2$ (see PRINTOUT command), one line additional information is printed for each contour point determined (usually between $\frac{1}{2}$ and 1 page per contour). This line gives the actual coordinates of the point (both absolute and relative to the minimum), an indication of the possible error (difference between FCN value on contour and FCN value at closest point where FCN was actually evaluated to determine contour point), the separation between the last two points, the slope of the line joining the last two points (theta in degrees) and the total number of FCN calls at that point.

After all contours of a given set are calculated, they are automatically plotted on one page. The scaling of the plot is chosen so that intervals are round numbers (subroutine BINSIZ), which usually results in different scaling for X- and Y-axes.

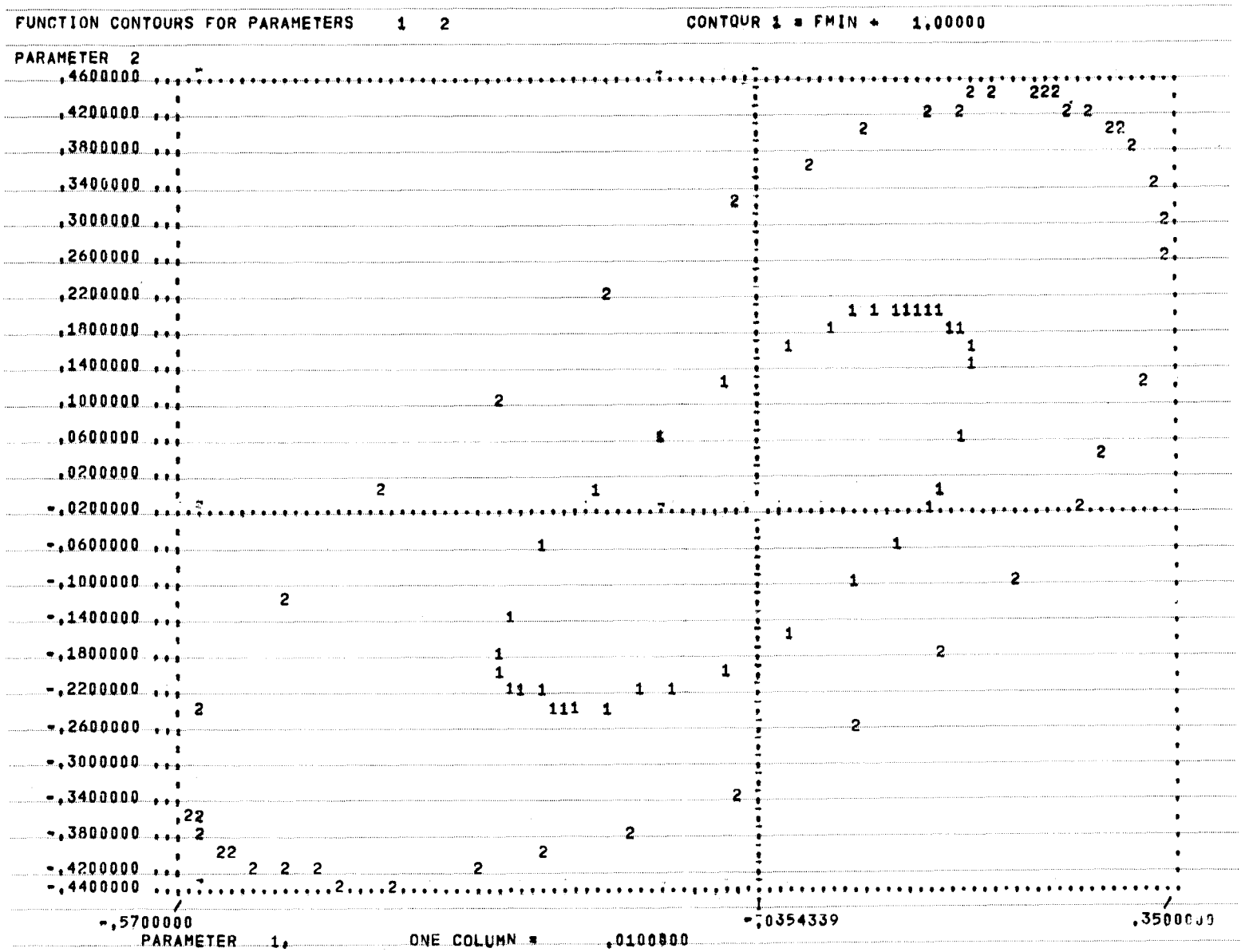
References

[1] F. James, Function Minimization in Proc. of the 1972 CERN Computing and Data Processing School, CERN 72–21 (1972). (Separate reprints of the minimization lectures alone are also available from the author).
[2] J.A. Nelder and R. Mead, Comput. J. 7 (1965) 308.
[3] R. Fletcher, Comput. J. 13 (1970) 317.
[4] W.T. Eadie, D. Drijard, F.E. James, M. Roos, and B. Sadoulet, Statistical Methods in Experimental Physics (North-Holland Publ. Co., Amsterdam, 1971).
[5] A.A. Goldstein and J.F. Price, Math. Comp. 25 (1971) 569.
[6] H.H. Rosenbrock, Comput. J. 3 (1960) 175.
[7] W.C. Davidon, Comput. J. 10 (1968) 406.
[8] R. Fletcher and M.J.D. Powell, Comput. J. 6 (1963) 163.
[9] F. James, Monte Carlo for Particle Physicists (Section 6.1) in: Methods in Subnuclear Physics, Ed. M. Nikolic, (Gordon and Breach Publ.) Vol. IV, Part 3.

TEST RUN OUTPUT

```
                                        **********
                                        *  D506  MINUIT *
                                        *  VERSION  5.75  *
                                        * DATA BLOCK NO,  1 *
 ********************************************************************************
 FIT TO TIME-DISTR OF KZERO LEPTONIC DECAYS                          TIME   .113
 ********************************************************************************
 ********************************************************************************
      1     REAL ETA      0.000000        .100000        +0.          -0.
      2     IMAG ETA      0.000000        .100000        +0.          -0.
      3     NORMFACT      1.000000        .100000        +0.          -0.
      4     DELTA M        .460000
 ********************************************************************************

 FIRST ENTRY TO FCN

  FCN VALUE      CALLS    TIME     EDM       INT,EXT,  PARAMETER     VALUE          ERROR        INTERN,VALUE  INT,STEP SIZE
 .7648158E+01      1      .113    0.            1   1   REAL ETA    0.             .10000E+00    0.             .10000E+00
                                                2   2   IMAG ETA    0.             .10000E+00    0.             .10000E+00
                                                3   3   NORMFACT     .10000E+01    .10000E+00     .10000E+01    .10000E+00
                                                    4   DELTA M      .46000E+00

 **********
 ***  1****
 **********
 **********
 ***  2****FIX              3.00000
 **********
 **********
 ***  3****MIGRAD
 **********
 START MIGRAD MINIMIZATION,          CONVERGENCE CRITERIA  --  ESTIMATED DISTANCE TO MINIMUM (EDM) .LT.  .10E-05
                                     OR EDM .LT.  .10E+00 AND FRACTIONAL CHANGE IN VARIANCE MATRIX .LT.  .10E-01

  FCN VALUE      CALLS    TIME     EDM       INT,EXT,  PARAMETER     VALUE          ERROR        INTERN,VALUE  INT,STEP SIZE
 .7648158E+01      7      .113    .82E+00       1   1   REAL ETA    0.             .14909E+00    0.             .20000E-02
                                                2   2   IMAG ETA    0.             .15102E+00    0.             .20000E-02
 MIGRAD MINIMIZATION HAS CONVERGED

  FCN VALUE      CALLS    TIME     EDM       INT,EXT,  PARAMETER     VALUE          ERROR        INTERN,VALUE  INT,STEP SIZE
 .7342787E+01     27      .113    .29E+06       1   1   REAL ETA     .48068E+02    .20126E+00     .48068E+02    .30305E-04
                                                2   2   IMAG ETA     .90441E+01    .21496E+00     .90441E+01    .79734E-04
                                                    3   NORMFACT     .10000E+01
                                                    4   DELTA M      .46000E+00

                                     ERRORS CORRESPOND TO FUNCTION CHANGE OF     1.0000

       INTERNAL COVARIANCE MATRIX    LAST FRACTIONAL CHANGE WAS     .014430
       CORRELATION COEFFICIENTS
 INT.    1
   2  .660
```

```
                                                    GLOBAL CORRELATION
                                                PARAMETER      COEFFICIENT
                                            1   REAL ETA         .43539
                                            2   IMAG ETA         .43539

**********
***  4****RESTORE
**********
                PARAMETER   3,  NORMFACT  RESTORED TO VARIABLE,
**********
***  5****MIGRAD
**********
START MIGRAD MINIMIZATION,                  CONVERGENCE CRITERIA  =  ESTIMATED DISTANCE TO MINIMUM (EDM) .LT.  .10E-05
                                            OR EDM .LT. .10E-00 AND FRACTIONAL CHANGE IN VARIANCE MATRIX .LT.  .10E-01

  FCN VALUE      CALLS     TIME      EDM      INT.EXT. PARAMETER      VALUE          ERROR       INTERN.VALUE  INT.STEP SIZE
 .7342787E+01      33      .113    .19E-00      1   1  REAL ETA     .48068E-02    .15247E+00     .48068E-02    .60611E-06
                                                2   2  IMAG ETA     .90441E-01    .16163E+00     .90441E-01    .15947E-05
                                                3   3  NORMFACT     .10000E+01    .49497E-01     .10000E+01    .20000E-02
MIGRAD MINIMIZATION HAS CONVERGED

  FCN VALUE      CALLS     TIME      EDM      INT.EXT. PARAMETER      VALUE          ERROR       INTERN.VALUE  INT.STEP SIZE
 .7137547E+01      75      .113    .21E-09      1   1  REAL ETA    -.35434E-01    .24439E+00    -.35434E-01    .10453E-05
                                                2   2  IMAG ETA    -.12031E-01    .32232E+00    -.12031E-01   -.72647E-06
                                                3   3  NORMFACT     .96669E+00    .74693E-01     .96669E+00   -.70402E-06
                                                    4  DELTA M      .46000E+00

                                            ERRORS CORRESPOND TO FUNCTION CHANGE OF     1.0000

        INTERNAL COVARIANCE MATRIX          LAST FRACTIONAL CHANGE WAS     .002629
        CORRELATION COEFFICIENTS
      INT.    1      2
        2   .734
        3   .419   .232

                                                    GLOBAL CORRELATION
                                                PARAMETER      COEFFICIENT
                                            1   REAL ETA         .57004
                                            2   IMAG ETA         .75827
                                            3   NORMFACT         .56716
**********
***  6****CONTOUR          1.00000        2.00000
**********

PARAMETERS NO,   1 AND   2          CONTOUR 1    F = FMIN +   1.000000

CONTOUR 1   IS COMPLETED WITH   45 POINTS


PARAMETERS NO,   1 AND   2          CONTOUR 2    F = FMIN +   2.000000

CONTOUR 2   IS COMPLETED WITH   48 POINTS
```


FUNCTION CONTOURS FOR PARAMETERS 1 2
CONTOUR 1 = FMIN + 1.00000
PARAMETER 2
.4600000
.4200000
.3800000
.3400000
.3000000
.2600000
.2200000
.1800000
.1400000
.1000000
.0600000
.0200000
-.0200000
-.0600000
-.1000000
-.1400000
-.1800000
-.2200000
-.2600000
-.3000000
-.3400000
-.3800000
-.4200000
-.4400000
-.5700000
-.0354339
.3500000
PARAMETER 1.
ONE COLUMN = .0100800

```
 FCN VALUE      CALLS     TIME      EDM       INT,EXT, PARAMETER      VALUE          ERROR       INTERN,VALUE  INT,STEP SIZE
 ,7137547E+01     239     ,115     ,21E-09      1   1  REAL ETA     -,35434E-01    ,24439E+00    -,35434E-01    ,10453E-05
                                                2   2  IMAG ETA     -,12031E+01    ,32232E+00    -,12031E-01    -,72647E-06
                                                3   3  NORMFACT     ,96669E+00    ,74693E-01     ,96669E+00    -,70402E-06
                                                    4  DELTA M      ,46000E+00

                                                ERRORS CORRESPOND TO FUNCTION CHANGE OF      1,0000

**********
***  7****PRINTOUT
**********
**********
***  8****MINOS            =0,00000
**********
MINOS ERRORS REQUESTED FOR PARAMETERS  1  2  3

DETERMINATION OF POSITIVE MINOS ERROR FOR PARAMETER  1   REAL ETA

PARAMETER   1 SET TO  -,354E-01 +   ,244E+00 =     ,209E+00
MIGRAD MINIMIZATION HAS CONVERGED

PARAMETER   1 SET TO  -,354E-01 +   ,208E+00 =     ,173E+00
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE POSITIVE MINOS ERROR OF PARAMETER  1,  REAL ETA , IS   ,2128E+00
                                                **************************************************************************

DETERMINATION OF NEGATIVE MINOS ERROR FOR PARAMETER  1   REAL ETA

PARAMETER   1 SET TO  -,354E-01 +  -,244E+00 =    -,280E+00
MIGRAD MINIMIZATION HAS CONVERGED

PARAMETER   1 SET TO  -,354E-01 +  -,290E+00 =    -,325E+00
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE NEGATIVE MINOS ERROR OF PARAMETER  1,  REAL ETA , IS  -,2991E+00
                                                **************************************************************************

DETERMINATION OF POSITIVE MINOS ERROR FOR PARAMETER  2   IMAG ETA

PARAMETER   2 SET TO  -,120E+01 +   ,322E+00 =     ,310E+00
MIGRAD MINIMIZATION HAS CONVERGED

PARAMETER   2 SET TO  -,120E-01 +   ,307E+00 =     ,295E+00
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE POSITIVE MINOS ERROR OF PARAMETER  2,  IMAG ETA , IS   ,3076E+00
                                                **************************************************************************

DETERMINATION OF NEGATIVE MINOS ERROR FOR PARAMETER  2   IMAG ETA

PARAMETER   2 SET TO  -,120E-01 +  -,322E+00 =    -,334E+00
MIGRAD MINIMIZATION HAS CONVERGED

PARAMETER   2 SET TO  -,120E-01 +  -,340E+00 =    -,352E+00
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE NEGATIVE MINOS ERROR OF PARAMETER  2,  IMAG ETA , IS  -,3413E+00
                                                **************************************************************************

DETERMINATION OF POSITIVE MINOS ERROR FOR PARAMETER  3   NORMFACT

PARAMETER   3 SET TO   ,967E+00 +   ,747E+01 =     ,104E+01
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE POSITIVE MINOS ERROR OF PARAMETER  3,  NORMFACT , IS   ,7283E-01
                                                **************************************************************************

DETERMINATION OF NEGATIVE MINOS ERROR FOR PARAMETER  3   NORMFACT

PARAMETER   3 SET TO   ,967E+00 +  -,747E+01 =    -,892E+00
MIGRAD MINIMIZATION HAS CONVERGED

                                                THE NEGATIVE MINOS ERROR OF PARAMETER  3,  NORMFACT , IS  -,7591E-01
                                                **************************************************************************
```

RESULTS OF FULL MINOS ERROR ANALYSIS
****

FIT TO TIME-DISTR OF KZERO LEPTONIC DECAYS

FCN VALUE	CALLS	TIME	EDM	INT.	EXT.	PARAMETER	VALUE	PARABOLIC ERROR	MINOS ERRORS POSITIVE	MINOS ERRORS NEGATIVE
.7137547E+01	386	.116	.21E-09	1	1	REAL ETA	-.35434E-01	.24439E+00	.21283E+00	-.29907E+00
				2	2	IMAG ETA	-.12091E-01	.32232E+00	.30764E+00	-.34128E+00
				3	3	NORMFACT	.96669E+00	.74693E-01	.72827E-01	-.75912E-01
					4	DELTA M	.46000E+00			

ERRORS CORRESPOND TO FUNCTION CHANGE OF 1.0000

INTERNAL COVARIANCE MATRIX
CORRELATION COEFFICIENTS

INT.	1	2
2	.734	
3	.419	.732

	PARAMETER	GLOBAL CORRELATION COEFFICIENT
1	REAL ETA	.57004
2	IMAG ETA	.75827
3	NORMFACT	.56716

****
*** 9****EXIT
****

CALL TO FCN WITH IFLAG = 3