SLAC-99
UC-34
(EXPI)

CIRCE — A GENERAL MULTI-PRONG FITTING COMPUTER PROGRAM FOR SPATIAL COORDINATES IN A NON-UNIFORM MAGNETIC FIELD

DIETRICH E. FRIES
STANFORD LINEAR ACCELERATOR CENTER
STANFORD UNIVERSITY
Stanford, California

PREPARED FOR THE U.S. ATOMIC ENERGY
COMMISSION UNDER CONTRACT NO. AT(04-3)-515

March 1969

ACKNOWLEDGEMENTS

I am grateful to Dr. J. Butcher for illuminating discussions and to Mr. R. Ching for assistance in programming.

TABLE OF CONTENTS

LIST OF FIGURES

A. INTRODUCTION

A standard problem connected with the detection and analysis of high energy particles is the optimization of particle trajectories with respect to some measured points along the trajectory.

Point coordinates of this kind are measured, for instance, by photographic or "on-line" techniques, in spark and wire chambers, set up with analyzing magnets, hodoscope systems with magnetic analysis as well as in bubble or streamer chambers.

The problem becomes considerably more complex when the magnetic field is not uniform but defined as some three-component function of the space coordinates, or when the interaction vertex is not a measured point and its position correlates several trajectories originating from it. A situation such as this will be encountered in experiments whenever the target is not directly viewed and the coordinates of the interaction vertex can not be measured or when trajectories pass through areas of nonuniform magnetic fields such as extended fringe fields.

One expects as a result of the optimization procedure track parameters such as spatial angles and the momentum at the vertex which fit best the measured data. Simplifying assumptions about the analytic form of the trajectories and the ignoring of the correlations between trajectories (especially in the case of an unmeasured vertex) may result in incorrect information on the mean and on the errors of the optimized parameters. The testing of the conservation of energy and momentum in an observed interaction, in particular the testing of kinematical hypotheses as to the masses of the observed particles or the parameters of an unmeasured particle requires an accurate knowledge of the covariance matrix of the particle angles and momenta.

In an earlier paper[1] a method was described for a simultaneous least square fit of the unmeasured vertex and all originating trajectories, allowing for a general form of a three-component magnetic field. In this report we want to document in some detail the computer procedure CIRCE which was developed to perform the nonlinear least square fit of the problem in question. We are describing a version of this computer program which requires as an input the measurements x_i, y_i, z_i of the points along a trajectory. The point coordinates are to be measured in the same coordinate system in which the magnetic field is defined. The distribution of the magnetic field can be arbitrary; thus points inside and outside the magnetic field region can enter the fit. Generally all trajectories of a multiprong event are

fitted simultaneously by the same fit. The vertex coordinates are regarded as three additional parameters which are being optimized together with the momenta and angles of all tracks emerging from this vertex. The computer routine allows to introduce correlations between measured points and renders the full covariance matrix of the fitted parameters of all tracks.

B. MATHEMATICAL FORMULATION

B.1 Definition of the Chisquare Sum

The optimization of the parameters is achieved by minimizing a chisquare which is defined as follows:

$$\chi^2 = \sum_{JI} \frac{(\vec{m}_{JI} - \vec{m}^*_{JI})}{\sigma^2_{JI}} \tag{1}$$

The vector $m = (x_{JI}, y_{JI}, z_{JI})$ represents the measured points of all tracks of the same event. I refers to the count of points of the track J. $\vec{m}^*$ represents the corresponding computed points on the trajectory which are defined as having a minimum distance to the measured points $\vec{m}$. σ_{JI} denotes the estimated measuring error for the measurement of the point $\vec{m}_{JI}$.

In a somewhat more general way one can write chisquare in a matrix form[3]

$$\chi^2 = (m - m^*)^T W (m - m^*) \tag{1a}$$

('T' means transposed)

Now m and m* represent rectangular matrices, containing all measured points of all tracks. For example (we will denote by x, y, z points on the computed trajectory)

$$m^* = \begin{bmatrix} x_{11} & x_{12} & x_{13} \cdots & x_{21} & x_{22} & x_{23} \cdots \\ y_{11} & y_{12} & y_{13} \cdots & y_{21} & y_{22} & y_{23} \cdots \\ z_{11} & z_{12} & z_{13} \cdots & z_{21} & z_{22} & z_{23} \cdots \end{bmatrix}$$

W is the weight matrix which is calculated by taking the inverse of the covariance matrix of the measurements of the point coordinates. For uncorrelated measurements W is simply a diagonal matrix containing the inverse of the squared measurement error. However the matrix formulation (1a) permits one to assign different errors to measurements of x, y and z or to introduce corresponding correlations if those are present.

B. 2 Representation of Trajectories

The computed track coordinates m* are points along the theoretical trajectory of a charged particle through a given magnetic field. The trajectory depends of course on parameters such as particle momentum, starting point and angles. The most general form of a trajectory is the solution of the set of differential equations

$$\frac{d\vec{u}}{ds} = \frac{c}{P}\ \vec{u} \times \vec{B}$$

$$\vec{u} = \left(\frac{dx}{ds}\ ,\ \frac{dy}{ds}\ ,\ \frac{dz}{ds}\right)$$

$$\vec{B} = \left(B_x\,(x,\ y,\ z),\ B_y\,(x,\ y,\ z),\ B_z\,(x,\ y,\ z)\right) \qquad (3)$$

magnetic field in $\left[\text{kG}\right]$

$c = 2.99791$

$P = \frac{1}{K}$ the particle momentum in $\left[\frac{\text{GeV}}{c}\right]$

s = arc length along the trajectory in [meter]

Equation (3) is derived from the Lorentz invariant form of the electromagnetic force equation. All quantities are defined in the laboratory system and electric fields are absent. Equation (3) is therefore valid for particles with arbitrary velocities. Energy dissipative forces which may obey a scalar equation of the form

$$\frac{d|P|}{ds} \sim \phi(s) \qquad (4)$$

do not enter the equation (3) explicitly. They can be taken in account by inserting an appropriate solution of Eq. (4) $P = P(s)$ into Eq. (3). (The numerical procedure to solve (3), which is described in section B. 6, will also work with a variable $P = P(s)$.)

The general set of initial values

$$\vec{X}_0 = (x_0,\ y_0,\ z_0) \qquad \text{at } s = s_0$$
$$\vec{u}_0 = \left(\frac{dx}{ds},\ \frac{dy}{ds},\ \frac{dz}{ds}\right) \qquad \text{at } \vec{x} = \vec{X}_0 \tag{5}$$

determines uniquely a solution of (3), which is obtained by a numerical method.

With our choice of the coordinate system (see Fig. 1) the derivatives are related to the dip angle λ and the azimuthal angle ϕ of the track at the vertex:

$$\left(\frac{dx}{ds}\right)_{\vec{X}_0} = \cos\lambda \cos\phi$$
$$\left(\frac{dy}{ds}\right)_{\vec{X}_0} = \cos\lambda \sin\phi \tag{6}$$
$$\left(\frac{dz}{ds}\right)_{\vec{X}_0} = \sin\lambda$$

The particular parametrization of Eq. (3) renders $\vec{u}$ as a unit vector. whose components are the directional cosines at any given point s of the trajectory. Thus we have at $s = s_0$ only two independent parameters represented by $\vec{u}$.

One can use a vector notation for the set of all parameters to be fitted. We count for an event

$$3\,N_R + 3$$

parameters which are to be fitted. The parameter vector then is defined as

$$\vec{\alpha} = (K_1,\ \lambda_1,\ \phi_1,\ K_2,\ \lambda_2,\ \phi_2,\ \ldots K_{N_R},\ \lambda_{N_R},\ \phi_{N_R},\ X_0,\ Y_0,\ Z_0)$$

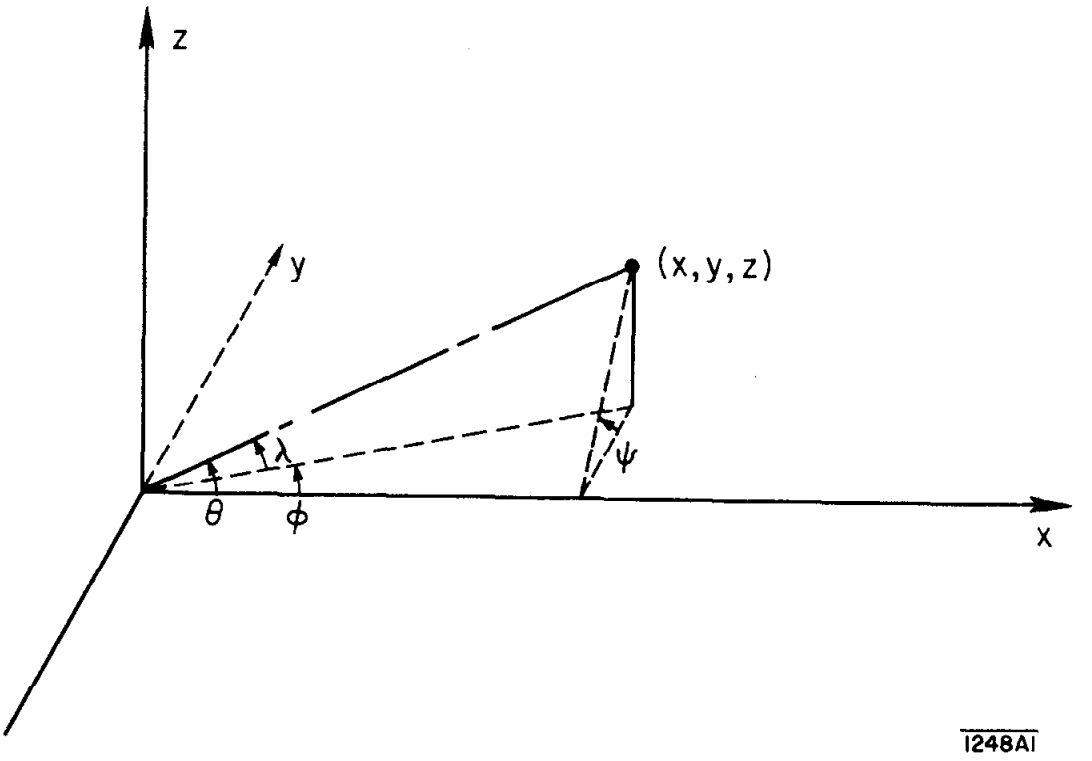

FIG. 1

N_R is the number of measured tracks emerging from the same interaction vertex. One notices that the common vertex is treated on the same basis as any other parameter and appears, regarding the initial value solution of (3), as a natural choice.

B. 3 Fitting of Single Trajectories or Multivertices

The method of fitting an entire event, including the vertex, poses however a problem for the fitting of a single trajectory. In this case one can optimize only two out of three vertex coordinates, since a single trajectory is defined, relative to measured points, only up to movements along the trajectory itself. Therefore, fitting of a single trajectory renders the origin of a track only at a point in a plane, arbitrarily defined, which intersects the track. The final parameters K, λ, ϕ are being fitted at this point.

It is therefore necessary to define one of the vertex coordinates as being fixed and not subject to variation due to the fit. (Alternatively one could also define a linear relation of the vertex coordinates; however one would have to modify slightly the input into the matrix A.)

For the problem of fitting more than one vertex of the same event, we suggest first to fit only trajectories originating from the same vertex and then to combine the results in a common output record by restoring the original numbering of the tracks. This way one does not obtain the geometrical correlations between trajectories from different vertices, which however could only be of interest when a charged track connects the different vertices.

In order to activate the different modes of fitting BOOLEAN switches have been provided.

B. 4 Linearization of the Chisquare Sum and the Corrections for the Parameters

The least square procedure to minimize chisquare employs the usual linearization by a first order Taylor expansion of the function. Starting from the definition of χ^2 (1a) one can expand

$$m^*(\vec{\alpha}_0 + \Delta\vec{\alpha}) = m_0^* + A\,\Delta\vec{\alpha} \tag{7}$$

with

$$m_0^* = m^*(\vec{X}_0)$$

and

$$A = \left(\frac{\partial m^*}{\partial \vec{\alpha}}\right)_{\vec{\alpha}=\vec{\alpha}_0} \tag{8}$$

inserting (7) into (1a) one obtains

$$\chi^2 = (\Delta C - A\Delta\vec{\alpha})^T W (\Delta C - A\Delta\vec{\alpha})$$

with

$$\Delta C = m - m_0^*$$

the condition

$$\frac{\partial \chi^2}{\partial \Delta\vec{\alpha}} = 0$$

leads to

$$(A^T W A)\ \Delta\vec{\alpha} = A^T W \Delta C \tag{9}$$

By solving the linear system (9) one can compute correction $\Delta\vec{\alpha}$ to $\vec{\alpha}$. An iterative procedure is provided to approach the minimum of χ^2 as defined in (1a). The decision whether the iteration has reached the χ^2 minimum is controlled by criteria which are discussed in the following section.

B.5 Convergence Criteria

After linearization of the nonlinear functions in (9) involving the parameters and after optimization of the parameters one expects that near the chisquare minimum the optimized parameters $\vec{\alpha}$ have a frequency distribution

$$\sim \exp -\frac{1}{2}\left\{(\vec{\alpha} - \vec{\alpha}')^T G^{-1} (\vec{\alpha} - \vec{\alpha}')\right\} \tag{10}$$

provided the measurements $\vec{m}$ are Gaussian distributed and their covariance matrix W is known. ($\vec{\alpha}'$ being the parameter vector, when χ^2 = minimum.)

The matrix G is related to W by a congruence transformation

$$G = A^T W^{-1} A \tag{11}$$

G is the positive definite covariance matrix under the assumption that W^{-1} is the covariance matrix of the measured quantities. (More detailed discussion of this can be found in Ref. 2.) Criteria for the convergence of the iterative fitting procedure can be established in the following ways:

(1) The difference of the chisquare of two consecutive iterations $\Delta\chi^2$ can be computed from (9). If $\Delta\chi^2$ is smaller than a positive number, say 0.3 one assumes to be close enough to the chisquare minimum. Actually we must perform this test using the standard deviation $\overline{\chi}$ rather than χ^2 which we defined as

$$\overline{\chi} = \sqrt{\frac{\chi^2}{n_f}} \tag{12}$$

where n_f is the number of degrees of freedom, since n_f varies from event to event as the number of tracks and measured points varies.

(2) A weak criterion for convergence would be to terminate the iteration when $\overline{\chi}$ is smaller than some number such as 0.8. It can be used to save computer time when the accuracy of a fit which reached a standard deviation of 0.8 is considered to be sufficient.

(3) A more exact criterion is obtained when one computes, by using the correction vector $\Delta\vec{\alpha}$, the bilinear form

$$\Delta\vec{\alpha}^T G\, \Delta\vec{\alpha} = \Delta\vec{\alpha}^T A^T WA\, \Delta\vec{\alpha}$$

which represents a multidimensional ellipsoid in the space $\Delta\vec{\alpha}$. In particular if one obtains near the chisquare minimum

$$\Delta\vec{\alpha}^T A^T WA \Delta\vec{\alpha} \leq 1 \tag{12a}$$

one expects that further iterations will give only corrections within the boundary of the error matrix. This can be seen if one assumes the covariance matrix G is diagonal and $\Delta\vec{\alpha}$ represents a vector whose components are equal to the orthogonal errors $\delta\alpha_i$ (eigenvectors to G); in this case (12a) is trivially equal to 1.

In order to calculate the standard deviation $\overline{\chi}$ one has to know the number of degrees of freedom which is defined as the number of measured points minus the number of parameters within the number of constraint equations. Since we

have for each measured point x, y, z one constraint equation namely the function putting these points on a trajectory we count $2/3\ N_p - (3N_R + 3)$ degrees of freedom per event where N_p is the number of measured points and N_R the number of tracks.

B.6 Numerical Computation of Trajectories

The numerical computation of trajectories is achieved by a step by step integration of the system of differential equations (3). The numerical method which was employed is a third order RUNGE-KUTTA method, which is one of the standard methods for a numerical solution of a differential equation. It does not require additional differentiations or additional initial values beyond the mathematically required ones. Thus the RUNGE-KUTTA is well suited for our problem, where the standard initial values contain the parameters to be optimized. We computed trajectories for a realistic magnetic field configuration to test the convergence and the computation speed and compared in particular the RUNGE-KUTTA results with the results obtained from a PREDICTOR CORRECTOR method. In general the integration step in a RUNGE-KUTTA method can be chosen considerably larger than in the PREDICTOR CORRECTOR method in order to obtain the same numerical accuracy of the integration (about a factor 5). Therefore, although the magnetic field

$$\vec{B}\ (x,\ y,\ z)$$

has to be computed three times per integration step in a RUNGE-KUTTA method (instead of two times in our PREDICTOR CORRECTOR procedure) the gain in computation speed due to the larger step size was still sizable.

We have formulated the RUNGE-KUTTA approximation for the system of differential equations (3) as follows: Let us write the system of first order differential equations in matrix form

$$\frac{dX}{ds} = F(\vec{u},\ \vec{x})$$

with the definitions

$$X = \begin{pmatrix} \vec{u} \\ \vec{x} \end{pmatrix}$$

$$F(\vec{u},\ \vec{x}) = \begin{pmatrix} c\vec{u} \times \vec{B}\ (\vec{x}) \\ \vec{u} \end{pmatrix}$$

Given a set of initial conditions at $s = s_0$

$$\vec{u}(s_0) = \vec{u}_0$$

$$\vec{x}(s_0) = \vec{X}_0$$

A third order RUNGE-KUTTA procedures requires the computation of essentially three intermediate quantities for the calculation of the next integration step Δs. They take for the equations (3) the following form:

$$\begin{pmatrix} \vec{k}'_1 \\ \vec{k}_1 \end{pmatrix} = \begin{pmatrix} \vec{u}_0 \times \vec{B}_0 \\ \vec{u}_0 \end{pmatrix} \Delta s$$

$$\begin{pmatrix} \vec{k}'_2 \\ \vec{k}_2 \end{pmatrix} = \begin{pmatrix} [\vec{u}_0 + 1/2\, \vec{u}_0 \times \vec{B}_0] \times \vec{B}_1 \\ u_0 + 1/2\, \vec{u}_0 \times \vec{B}_0 \end{pmatrix} \Delta s$$

$$\begin{pmatrix} \vec{k}'_3 \\ k_3 \end{pmatrix} = \begin{pmatrix} [\vec{u}_0 + 2\vec{k}'_2 - \vec{k}'_1] \times \vec{B}_2 \\ u_0 + 2\vec{k}_2 - \vec{k}_1 \end{pmatrix} \Delta s$$

with

$$\begin{aligned} \vec{B}_0 &= \vec{B}\,(s_0) \\ \vec{B}_1 &= \vec{B}\,(s_0 + 1/2\,\Delta s) \\ \vec{B}_2 &= \vec{B}\,(s_0 + \Delta s) \end{aligned} \qquad (14)$$

the solution of (3) at a point $s_0 + \Delta s$ is then obtained from

$$X(s_0 + \Delta s) = \begin{pmatrix} \vec{u}_0 + 1/6\,[\vec{k}'_1 + 4\vec{k}'_2 + \vec{k}'_3] \\ \vec{X}_0 + 1/6\,[\vec{k}_1 + 4\vec{k}_2 + \vec{k}_3] \end{pmatrix}$$

our parametrization of the initial values (5) is consistant with the constraint condition

$$|\vec{u}| = 1 \qquad (15)$$

as it was mentioned earlier and introduces in fact only five parameters, which are sufficient to determine uniquely the initial value solution of (13). Thus we make no explicit use of (15), which is taken into account in the actual computation in the SUBROUTINE TRACK automatically.

C. DISCUSSION OF THE COMPUTATION FLOW AND THE FUNCTION OF VARIOUS SUBROUTINES

C.1 MAIN, SUBROUTINES ESTIM and FIT

An effort was made to keep the computer procedure CIRCE transparent in its logical structure and simultaneously versatile so that it can be adapted to a large class of applications. The provision of suitable INPUT-OUTPUT routines (packing of data, etc.) for the analysis of a large number of data has been left to the user. No extended failure messages (IFAIL) have been provided either.

The first SUBROUTINE called in MAIN is SETLOG which initializes and sets the necessary constants. Then we proceed in MAIN to read data (measured point coordinates). FORMAT and other details are listed in section D. 2.

Due to a logic which is essential for our fitting routine, we have to provide for each fit the measured points, sequenced, starting from the vertex, of all tracks originating at the same vertex. They are stored in two-dimensional arrays. XSTORE (J, I), YSTORE (J, I), ZSTORE (J, I) . J refers to the track number, I refers to the point number. The estimated errors of the measured points SGX, SGY, SGZ have been set in SETLOG. They are assumed to be constant for all points, and they are assigned to the weight matrix W. In general the errors can be different for each point. Next the SUBROUTINE ESTIM is called.

The nonlinear least square optimization requires one to provide starting values for the parameters (first estimates). It is difficult to define in general in what proximity of the final fit the starting values have to lie, since it depends critically how well a function can be represented by a first order Taylor linearization. In practical cases first estimates of the momentum which were 10% or 20% off the final value converge without difficulties, in some cases even estimates which were wrong by a factor 5 could be fitted.

Using simulated events we tested that for first estimates as computed in the SUBROUTINE ESTIM, the final result is independent of the starting values,

to the extent that the fluctuations of the final results are small compared to the error of the parameters. In order to achieve convergence in as few iterations as possible, for reasons of computer economics, it is profitable to provide starting values reasonably close to the final values of the parameters.

In ESTIM we compute in a simple way starting values for track momenta, angles and the common vertex. By selecting two trajectories one finds from their intersection vertex coordinates in the horizontal plane. Fitting a circle to the track points in the magnetic field renders projected momenta and angles. The z coordinates of the measured track points yield first estimates for the dip angle λ. The FORTRAN names under which the parameters appear in CIRCE are P(J), K(J), LAMDA(J), PHI(J) and XIN, YIN, ZIN. In many cases one can provide certain starting values as being constant for all events.

The SUBROUTINE FIT is a bookkeeping routine which communicates mainly with the SUBROUTINE FONC and FUNCTION SOLVE by COMMON-blocks. By looping through all points and track indices for each point all residuals and partial derivatives are assembled and filled into the matrix A, which establishes the link to the FUNCTION SOLVE. Thus in SOLVE, which is called for each point, the system of linear equations (9) can be formed. By matrix inversion SOLVE proceeds to compute corrections $\Delta\vec{\alpha}$ for the parameter $\vec{\alpha}$. The corrections are transmitted from SOLVE back into FIT by the same matrix A. The corrections are added to the parameters in FIT. SOLVE = SI provides a number, which signals the status of convergence of the iterative fitting procedure. If SI is equal -1, a further iteration is initiated. If SI is equal zero the fit is considered a failure (either chisquare increases or more than 8 iterations are required) and is discontinued. If SI is bigger than zero the fit has converged and SI is the standard deviation of the fit. These criteria can be overridden if one accesses the chisquare information in SOLVE after each iteration and uses one's own criteria. (For information about SOLVE see Appendix I and Ref. 2.)

C.2 SUBROUTINE FONC

FONC being called in a loop over all points and tracks in the SUBROUTINE FIT, activates the basic blocks of computation necessary for the fitting procedure. The first job carried out in FONC is the computation of a numerical solution of the equation (3) by calling the track integration routine TRACK. FONC activates

TRACK for each trajectory J and current set of parameters K(J), LAMDA(J), PHI(J), XIN, YIN, ZIN. As it will be described in section C. 3, the solution of (3) is rendered as a sequence of numbers

$$x(s, \vec{\alpha}), \quad y(s, \vec{\alpha}), \quad z(s, \vec{\alpha})$$

defined as track points having the closest distance to the measured points $\vec{m}$. They are stored in arrays XCAL, YCAL, ZCAL.

Next the partial derivatives

$$\frac{\partial x}{\partial \vec{\alpha}}, \quad \frac{\partial y}{\partial \vec{\alpha}}, \quad \frac{\partial z}{\partial \vec{\alpha}}$$

required by the least square method have to be computed numerically. This is being done in a corresponding way by using the same integration routine TRACK, applied for slightly varied parameters

$$\alpha_i' = \alpha_i + \delta\alpha_i$$

forming a difference for each track point $(x - x')/\delta\alpha_i$.

The derivatives with respect to the vertex, however, are of a particularly simple nature. In fact, for the case of a helix one has

$$\left(\frac{\partial m}{\partial \vec{X}_0}\right)_{s=\text{const.}} = I \tag{16}$$

where I is a diagonal unit matrix.

However, as we have discussed in Ref. 1, the partial derivatives used in our procedure are not computed at s = constant, but at a track point where the trajectory has the closest distance from the corresponding measured point. This necessitates for the case of a helix replacing (16) by

$$\frac{\partial m}{\partial \vec{X}_0} = I - \vec{u}^T \cdot \vec{u} \tag{16a}$$

These simplified partial derivatives (with respect to the vertex coordinates) have been used successfully in a magnetic field which varied from maximum values to zero field and where a second (radial) field component reached 30% of the maximum value of the main field component within the region where track-coordinates had to be fitted. We have provided a BOOLEAN switch, PRECIS,

in order to use optionally for $\frac{\partial m}{\partial \vec{X}_0}$ either the numerically computed partials or the approximation (16a).

$$\frac{\partial x}{\partial \vec{\alpha}} = \frac{x(s, \vec{\alpha}') - x(s, \vec{\alpha})}{\delta \vec{\alpha}}$$

$$\frac{\partial y}{\partial \vec{\alpha}} = \frac{y(s, \vec{\alpha}') - y(s, \vec{\alpha})}{\delta \vec{\alpha}}$$

$$\frac{\partial z}{\partial \vec{\alpha}} = \frac{z(s, \vec{\alpha}') - z(s, \vec{\alpha})}{\delta \vec{\alpha}}$$

one can approximate for most practical cases $\delta\alpha_i$ by

$$\delta\alpha_i \leq 10^{-3} \times \alpha_i$$

Then the result of the fit appears to be independent from the choice of $\delta\alpha$.

The partials are stored in the arrays

XDU,	YDU,	ZDU,
XDV,	YDV,	ZDV,
XDW,	YDW,	ZDW.

This method to obtain partial derivatives on a digital computer is relatively time consuming.

For our least square fit of a multiprong event about 80% of the computer time is used to compute partials. One can think of using partial derivatives of an approximate solution of (3) which may have a simple analytic form such as a helix; on the other hand, high speed digital computers usually make the numerical computation of partials feasible even for large samples of events.

C.3 SUBROUTINE TRACK

The numerical method described in section B.6 is carried out in the SUBROUTINE TRACK. After each step Δs, the stepwise integration provides coordinates $\vec{x}$ and directional cosinus $\vec{u}$ of the trajectory. Starting from the initial values,[5] the numerical integration proceeds by testing after each step

the distance vector $\vec{d}_i$, which connects the last point on the trajectory with the nearest measured spatial point $\vec{m}_i$. A very simple test indicates when the trajectory passes through the closest distance to $\vec{m}_i$, namely when the condition

$$\vec{u} \cdot \vec{d} \leq 0$$

is satisfied. Then by a linear geometrical approximation we calculate the particular step size Δs by which the exact track point of closest distance $|\vec{d}_i|$ = Min. can be reached. By updating this point in the list of measured points we proceed to integrate and to test the next of the sequence of measured points until the last of the measured points has been updated. As an immediate consequence of this, TRACK can compute trajectories which bend more than 180° or even cross over in a loop, provided track points are properly sequenced. The step size Δs has to be chosen with some care, for reasons of accuracy on one hand and for computation economies on the other hand.

An empirical scaling rule for the step size Δs has been adopted:

If $P < 1$ BeV then $\Delta s = 0.1$ meter

If $P \leq 1$ BeV then $\Delta s = 0.1 \times P$ meter

In general one should scale the step size Δs not only according to the momentum but also in relation to the magnetic field $|\vec{B}|$, especially in order to adjust Δs for the field-free regions where Δs could of course be large.

The parameters K(J), LAMDA(J), PHI(J) are transmitted through the calling sequence of the SUBROUTINE TRACK. XIN, YIN, ZIN and the computed track coordinates XCAL(I), YCAL(I), ZCAL(I) are communicated by COMMON-blocks.

Here, as before, I refers to the count of points and J to the count of tracks. Information about the tracks length or the directional cosines at any point is also available.

C.4 Representation of the Magnetic Field

A two component magnetic field in the standard version of CIRCE is represented by functions FLDBZ and FLDBR of (x, y, z), which are called from the SUBROUTINE TRACK. (For a computer where the access time to a FUNCTION is appreciable compared to the time which is actually used for computation, one can program $\vec{B}$ (x, y, z) directly into the SUBROUTINE TRACK.) In order to access the magnetic field information one has several possibilities.

One can store the actual measurements of the field components and retrieve them from a 3-dimensional table by interpolation. Or one can instead store data, obtained by fitting a smooth function to the measurements and interpolate. We preferred to not use any table, but to compute the field for each point, where it is required, from functions which were obtained by fitting the measured field data. Since the field components can vary as much as 100% a polynomial fit to the entire magnetic field would have an excessive number of coefficients. It is, however, straight forward to subdivide the region in which the field has to be known into several parts, and to represent it in the various regions by relatively simple functions. For the functions we used polynomials or – especially for regions with large field gradients – inverse polynomials.

The numerical stepping procedure permits without any difficulty to decide in which region the point lies for which the field has to be computed. No effort was made to match the function at the boundaries of the various areas, since wherever they are defined, they represent the field within known uncertainties. Due to the fact that the field is represented by asymptotic functions no effort was made to find functions exactly satisfying the div $\vec{B} = 0$ condition from Maxwell's equations.

The field representation used in the standard version of CIRCE represents a fit to the measurements of a large magnet with circular pole pieces (diameter 2 meters). The functions used require 78 coefficients which are stored in SETLOG and communicated by a COMMON-block.

D. SUMMARY OF THE USE OF THE PROCEDURE CIRCE

D.1 Units, Identifiers and Coordinates

The units used throughout CIRCE are

length in	[METER]
angles in	[RADIANS]
magnetic field in	[K GAUSS]

The main variables and identifiers are in the list that follows.

I	index referring to measured points
J	index referring to trajectories originating at the same vertex
NO(J)	number of points per track
NR	number of tracks from the same vertex
XSTORE(J, I), YSTORE(J, I), ZSTORE(J, I)	measured spatial coordinates sequenced
SGX, SGY, SGZ	measurement errors of the coordinates XSTORE, YSTORE, ZSTORE
XIN, YIN, ZIN	unmeasured vertex coordinates to be fitted for
K(J) = 1/P(J), LAMDA(J), PHI(J)	parameters to be fitted for. J referring to the different trajectories
CHQ(J)	charge of the particle of track J
BKG	approximate average magnetic field strength
A	matrix containing residuals and partial derivatives. Communicates with SOLVE
W	weight matrix

The coordinate system in which the magnetic field and the measured coordinates of the track are defined in our standard version of CIRCE is a right-handed Cartesian system where the x axis points along the incident beam, the x y plane is horizontal and z points vertically upwards.

The point (x = 1.00, y = 0, z = 0) lies in the geometrical center on the vertical symmetry axis of the gap of the (circular) magnet. The angles are defined as shown in Fig. 1 with

$$-\pi \leq \lambda \leq \pi$$

$$-\pi \leq \phi \leq \pi$$

D. 2 Input Requirements

The main sequence of computations is indicated in the flow diagram Fig. 2. Subsidiary and utility routines are left out of this diagram.

In the standard version of CIRCE, the control parameters are read from data cards in the form of single digit integer numbers which set the corresponding

FLOW DIAGRAM

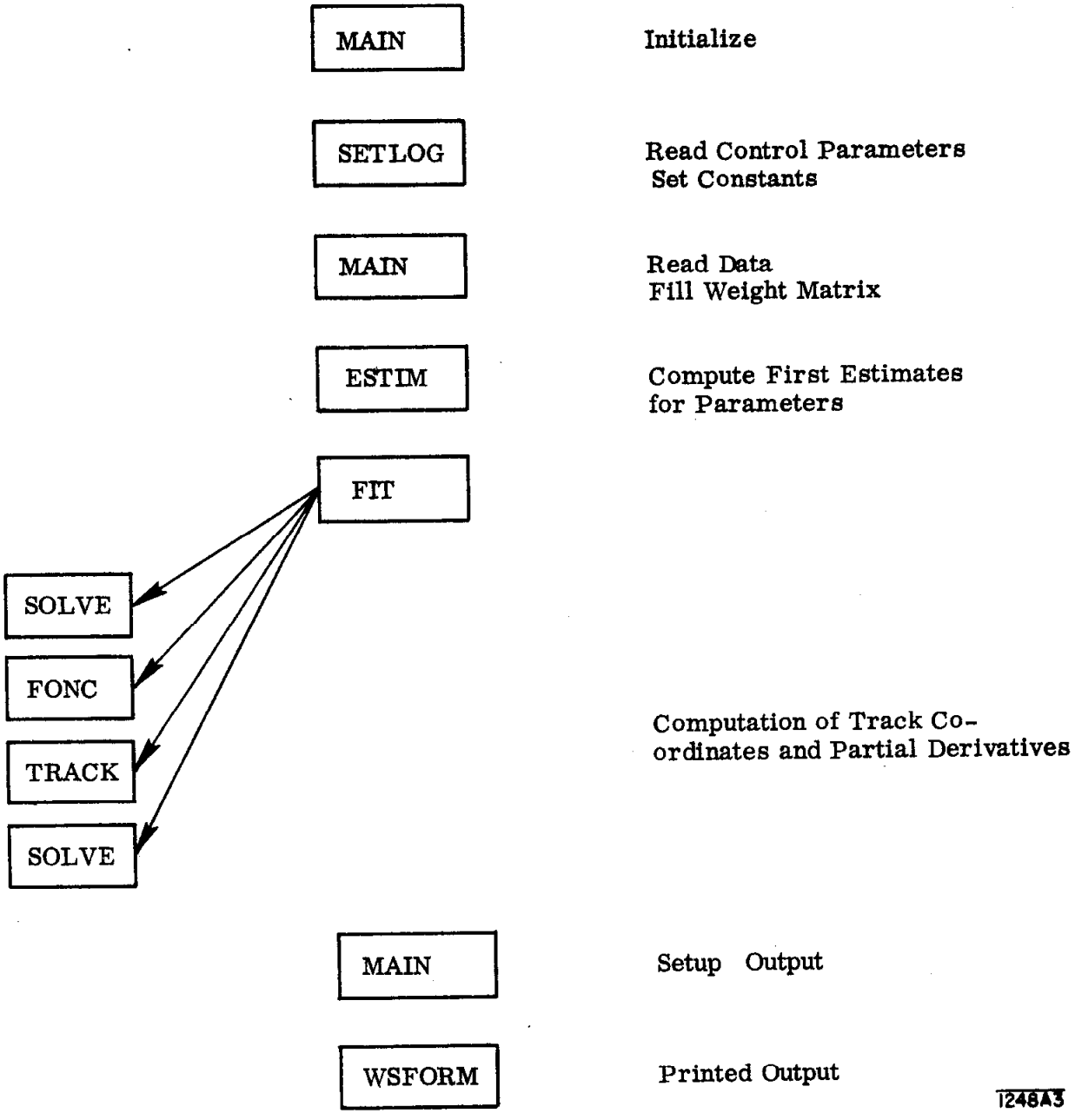

FIG. 2

BOOLEAN variables in SETLOG. The following 5 integers J1, J2, J3, J4, J5 are read from the

FIRST DATA CARD	FORMAT (5 I2)
J1 = 0	all partials are computed numerically
J1 = 1	partials with respect to the vertex are approximated
J2 = 1	a single track is to be fitted
J2 = 0	otherwise
J3 = 0	a multi-prong event (NR $\geq$ 2) is to be fitted
J3 = 1	otherwise
J4 = 0	origin of a single track defined in a plane (x_0 = const., y, z)
J4 = 1	origin of a single track defined in a plane (x_0 = XSTORE (1, 1), y, z) = first measured point on the track. J4 is achieved only if J2 = 1.
J5 = 0	printed output only
J5 = 1	a tape record will be written

Starting with the second Data Card always an entire event is being read before the fitting procedure starts:

SECOND DATA CARD FORMAT (I10, F10.5, I10)

reads the event sequence number, the charge of the track J and the number of measured points for the track J.

THIRD DATA CARD FORMAT (3E 16.8)

reads the x, y, z coordinates.

The input requirements can be summarized as follows:

(a) point coordinates XSTORE, YSTORE, ZSTORE have to be sequenced, starting from the (unmeasured) vertex, as indicated in Fig. 3.

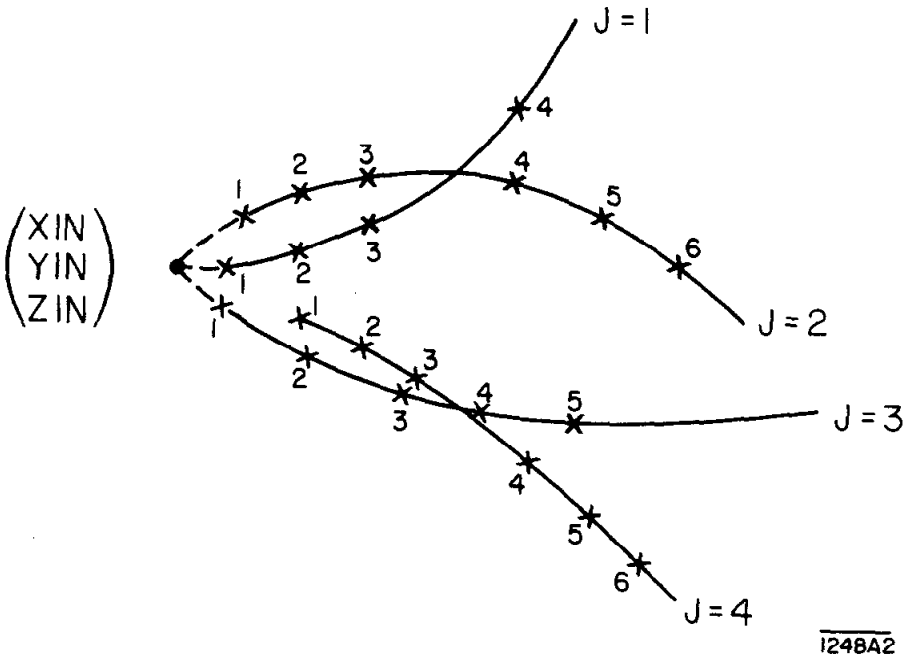

FIG. 3

(b) if the measurements of the point coordinates are uncorrelated, the weight matrix is

$$W = \begin{pmatrix} \frac{1}{SGX**2} & 0 & 0 \\ 0 & \frac{1}{SGY**2} & 0 \\ 0 & 0 & \frac{1}{SGZ**2} \end{pmatrix}$$

(c) starting values must be supplied for the parameters

$$K(J) = 1/P(J)$$

$$LAMDA(J),\ PHI(J),\ ZIN,\ YIN,\ XIN,\ CHQ(J)$$

$$J = 1,\ 2,\ 3\ .\ .\ .\ N_R$$

They are computed in ESTIM if a circle represents a track approximately ESTIM can be bypassed if starting values are supplied externally.

(d) The magnetic field is supplied through SUBROUTINES, FLDBZ and FLDBR. The coefficients for the functions represented by FLDBZ and FLDBR are stored in SETLOG.

If the magnetic field has components B_x, B_y, B_z, then appropriate functions entering TRACK have to be defined.

D.3 Output Facilities

Printed Output

An example of printed output for a fitted event is given in Fig. 4.

First the track coordinates XSTORE, YSTORE, ZSTORE are printed in 3 parallel columns. There we distinguish 3 tracks. The numbers preceeding the track coordinates denote (from left to right) event number (= 4), charge (= ±1.0) and number of points per track (= 12, 14 and 9).

```
         4       1.00000000              12
 1.13836956      -0.02890500       0.03832400
 1.23844051      -0.02913400       0.04294200
 1.33765753      -0.03215200       0.04150000
 1.43824768      -0.03306900       0.03558200
 1.53816223      -0.03465800       0.03716600
 1.63844109      -0.03732600       0.03601500
 1.73969173      -0.03828800       0.04042600
 1.83918190      -0.04155800       0.04461900
 1.93867683      -0.04482300       0.03037800
 3.93741226      -0.11276096       0.03643500
 5.93638611      -0.18648100       0.03799000
 8.00079536      -0.26067799       0.03474200

         4      -1.00000000              14
 0.93938595      -0.01888800       0.03689900
 1.03912449      -0.01702300       0.03837200
 1.13891411      -0.01376000       0.03861800
 1.23868465      -0.01164400       0.04451400
 1.33822632      -0.00955600       0.03797200
 1.43840122      -0.00671500       0.04187200
 1.53797531      -0.00402000       0.04697800
 1.63798428      -0.00011100       0.04193600
 1.73744678       0.00297700       0.03277300
 1.83799458       0.00597100       0.03419500
 1.93847752       0.00993900       0.03987700
 3.93580818       0.09742695       0.04307600
 5.93478354       0.18841499       0.04563700
 8.00015640       0.28171897       0.04408400

         4       1.00000000               9
 0.65146697      -0.11124796       0.03956700
 0.65360695      -0.19894195       0.03822600
 0.64661294      -0.28770798       0.04704000
 0.62952900      -0.37232095       0.04834600
 0.60181999      -0.45744896       0.05549300
 0.56443298      -0.53628695       0.05394500
 0.52034497      -0.61246598       0.05975100
 0.46794397      -0.68196797       0.04491400
 0.40551096      -0.75049698       0.05189800
```

```
NUMBER OF POINTS USED FOR FIT =   12   14    9

FIRST ESTIMATE  P(1) =  25.00000 LAMDA(1) =   0.01593 PHI(1) =   0.08269

FIRST ESTIMATE  P(2) =  25.00000 LAMDA(2) =   0.00861 PHI(2) =  -0.00724

FIRST ESTIMATE  P(3) =   0.30058 LAMDA(3) =   0.04231 PHI(3) =  -1.41362

       0.644760        0.0             0.040000

 CORR



   260.6  105   0.00068   0.00447  -0.08965   0.00176  -0.00390   0.02402   0.08925   0.00917   0.04707  -0.00769
-0.02500  -0.01458



    6.9  105   0.00758  -0.01985   0.00373   0.00149  -0.00366  -0.00111  -0.01197  -0.02489  -0.00223   0.00180
 0.00057   0.01279



    1.1  105   0.00009  -0.00107  -0.00000   0.00013  -0.00022  -0.00002   0.00574  -0.00216  -0.00003   0.00001
 0.00001   0.00089



    1.0  105   0.00000  -0.00000  -0.00000   0.00000  -0.00000   0.00000   0.00058  -0.00001   0.00001  -0.00000
-0.00000   0.00000

   STDEF      CHI**2         DF
     1.0      63.90         58.

    COV
 0.00109
 0.00046  0.00
 0.00069  1.00 -0.00
 0.00115 -0.83  0.00 -0.85
 0.00047  0.00  0.27  0.00  0.00
 0.00073  0.84 -0.00  0.87 -1.00  0.00
 0.01888  0.02  0.02  0.03  0.03  0.02 -0.03
 0.00318  0.01  0.28  0.01 -0.01  0.29  0.01  0.05
 0.00340 -0.12  0.01 -0.12  0.18  0.01 -0.19  0.96  0.02
 0.00064  0.09 -0.01  0.09 -0.15 -0.00  0.16 -0.82 -0.01 -0.92
 0.00043 -0.90  0.00 -0.93  0.92 -0.00 -0.94 -0.00 -0.01  0.16 -0.13
 0.00099 -0.00 -0.51  0.00 -0.00 -0.52  0.00 -0.03 -0.55 -0.02  0.01 -0.00


ROLL     0      FRAME      0      ETYPE     300     MEAS. TABLE**     MEASUR.  0      DATE      0      NSFO     4
IFAIL    0      STDV   1.0      CHISQ    63.9      DF  58.0          PROBL  0.00      NO. ITR  4      TIME    1.85     IREC3 -915619724

TRK NPT     XIN       DXIN       YIN       DYIN       ZIN       DZIN       P         DP        LAMDA     DLAMDA     PHI       DPHI
 1  12   63.8873    0.0642   -2.4430    0.0435    3.9098    0.0993  20.6802    0.4653   -0.0005    0.0005   -0.0032    0.0007
 2  14   63.8873    0.0642   -2.4430    0.0435    3.9098    0.0993  23.0469    0.6106    0.0008    0.0005    0.0157    0.0007
 3   9   63.8873    0.0642   -2.4430    0.0435    3.9098    0.0993   0.2932    0.0016    0.0244    0.0032   -1.3688    0.0036

000000000 000000000 000000000 000000000
```

FIG. 4

Next, number of points and the starting values of the parameters are printed. The vertex coordinates are in the last row before CORR. After each iteration one obtains from SOLVE the following information (see example):

260.6	105	.00068	.00447	-.08965	.00176	-.00390
standard deviation	degrees of freedom	ΔK_1	$\Delta\lambda_1$	$\Delta\phi_1$	ΔK_2	$\Delta\lambda_2$

The last corrections are ΔXIN, ΔYIN, ΔZIN.

After the final iteration follows the result of the fit. Standard deviation (= STDEF), chisquare and degrees of freedom are preceeding the errors and correlations.

The errors which are the square roots of the elements on the main diagonal of the variance covariance matrix CA are printed in the right hand column. The triangular matrix to the left of the errors contains the normalized correlations of the parameters in the form

$$
\begin{array}{llll}
\left[\lambda_1, K_1\right] & & & \\
\left[\phi_1, K_1\right] & \left[\phi_1, \lambda_1\right] & & \\
\left[K_2, K_1\right] & \left[K_2, \lambda_1\right] & \left[K_2, \phi_1\right] & \\
\left[\lambda_2, K_1\right] & \left[\lambda_2, \lambda_1\right] & \left[\lambda_2, \phi_1\right] & \left[\lambda_2, K_2\right]
\end{array}
$$

Normally one would suppress this output and print only the short form output which follows next.

The heading provides printed information, most of which is left to the user to fill. The information on the tracks is self explanatory. The last line contains for a N_R-prong (N = 3 here) $9 + N_R \times 9$ integer words INOTE for special information. They are not used in the present version of CIRCE. The print is only useful in this form if in INOTE only single digit words are used. A simple list of IFAIL is supplied which sets a flag when a track has less than 3 points, when one exceeds 8 iterations or when the standard deviation of a fit is less than 3.0. They can be easily changed by the user to his own specifications.

Output on Tape

The tape record is written by a utility routine BWRITE (for which a complementary routine BREAD is available).

The SUBROUTINE BWRITE (IOUTAP, 4HGEOM, 2, VNF, CAW) generates a binary record on the tape unit IOUTAP. The data are organized in this record as follows:

First, 20 integer words IREC which carry general information about the event and the length of the logical blocks. In particular we assign:

IREC(3)	to the event identifier (event number)
IREC(5)	to the number of tracks
IREC(6)	to a label for the data set
IREC(8)	to the standard deviation of the fit (floating point word)
IREC(10)	to the length of block VNF
IREC(11)	to the length of block CAW

Second, the data follow subdivided into two blocks, VNF and CAW. VNF contains the fitted parameters. The allocations are the following:

VNF(1)	λ of incident track
VNF(2)	ϕ of incident track
VNF(3)	K of track 1
VNF(4)	λ of track 1
VNF(5)	ϕ of track 1
VNF(6)	K of track 2
VNF(7)	λ of track 2
VNF(8)	ϕ of track 2
VNF(9)	K of track 3
.	
.	
.	
VNF (2 + 3 × (NR-1))	K of track NR
.	
.	
.	
VNF (3 + 3 × NR)	XIN
VNF (4 + 3 × NR)	YIN
VNF (5 + 3 × NR)	ZIN

The block CAW contains the variances and covariances of the parameters. The allocation is self explanatory from the way we sequence the elements of the triangular matrix CAW. Using the notation $(Q, R)_I$ where $(R, R)_I$ means the

variance of the parameter R and $(Q, R)_I$ means the covariance of the parameters Q and R, and I gives the sequence number of the element in the one-dimensional array CAW.

$(K_1, K_1)_1$

$(\lambda_1, K_1)_2$ $(\lambda_1, \lambda_1)_3$

$(\phi_1, K_1)_4$ $(\phi_1, \lambda_1)_5$ $(\phi_1, \phi_1)_6$

$(K_2, K_1)_7$ $(K_2, \lambda_1)_8$ $(K_2, \phi_1)_9$ $(K_2, K_2)_{10}$

$(\lambda_2, K_1)_{11}$ $(\lambda_2, \lambda_1)_{12}$ $(\lambda_2, \phi_1)_{13}$ $(\lambda_2, K_2)_{14}$

D. 4 Analysis of Simulated Events

We supplied for the standard version of CIRCE a test fit by analyzing simulated data of the type

$$\gamma p \rightarrow p\, K^+\, K^- \qquad \text{(3 prong)}$$

where the invariant mass of the $K^+ K^-$ system was assumed to be 1080 MeV with a width of 12 MeV. The distribution of the four momentum transfer to the proton was sampled from an exponential distribution. The photon energy has a uniform spectrum

$$40 \leq E_\gamma \leq 50 \text{ GeV}$$

We generated track coordinates with random errors ($\Delta x = \Delta y = \pm .5$ mm), and generated trajectories which extended into the zero field region and used simulated track coordinates from the region where the magnetic field is high (13 kG) as well as from the zero field region. The magnetic field represents a fit to the actual measurements of the SLAC 2 meter streamer chamber magnet. Measurements of the fringe field are included in our fit, where we used 100 coefficients to represent $\vec{B}(x, y, z)$ which is subdivided into three radial regions. The field is axial symmetrical and represented in 2 components $B_z(R, z)$ and $B_R(R, z)$. The number of degrees of freedom is about 90 per fit; the number of parameters to be fitted per event is 12. The computation time on an IBM 360/91 computer was about 1 sec per event. (The computation time could be reduced

by about a factor 5 if it is possible to use approximate partial derivatives in FONC, which could be calculated assuming an approximate explicit representation of the trajectories, e.g., a helix.) Furthermore, the computation time scales roughly with the number of degrees of freedom and the number of coefficients for the representation of the magnetic field.

An example of a fitted event of the type described above is given in Fig. 4 (see also section D. 3). Here we used a fixed starting value for the momenta of track one and two and also for YIN and ZIN. The fitted values are very close to the one generated by the fake program:

Track	P(FIT)	ΔP	P(FAKE)	[GeV]
1	20.6802	.4653	20.970	"
2	23.0469	.6106	23.100	"
3	.2932	.0016	.2932	"

Vertex	FIT			FAKE		
x_0	y_0	z_0	x_0	y_0	z_0	[cm]
63.887	-2.443	3.9098	63.865	-2.449	4.046	

E. POSSIBLE MODIFICATIONS

In the standard versions the measurement errors SGX, SGY, SGZ are assumed to be constant for all points and all tracks, but in general it is straightforward to assign different errors or general weights to each point, or even to introduce correlations related to SGX, SGY and SGZ. It is necessary then to fill the matrix within the general loop over all measured points, which is executed in FIT, where the residuals are also set up.

A continuous or discontinuous energy loss of a particle along its path of flight can be taken into account by modifying K(J) in TRACK. For constant energy loss $\frac{d|P|}{ds} = \epsilon$ the modification would be $\frac{1}{K} = P \rightarrow P(1 - \epsilon \cdot s)$.

There can be constraint equations $F(\vec{x}, \vec{\alpha}) = 0$ of geometrical or kinematical nature which one wishes to be satisfied simultaneously with the fit. Instead of using Lagrange multipliers, one can treat the constraint equation the

same way as a measurement; that is to say the corresponding residual takes the form

$$\Delta C = F(\vec{x}, \vec{\alpha}) - 0$$

with an appropriate error

$$\sim \frac{\partial F}{\partial \vec{x}} \Delta x$$

Conveniently one can use the fourth column in A for this purpose (since column 1 to 3 are occupied by residuals and partials due to x, y, z).

A kinematical fit using energy and momentum conservation can be performed using the fitted geometry parameters. A kinematical fitting routine, matched to the output of CIRCE was developed at SLAC by I. Derado and R. Leedy.[4]

REFERENCES

1. D.C.E. Fries, Nucl. Inst. Methods 44, 376 (1966).
2. F. T. Solmitz, Ann. Rev. Nucl. Sci. 14, 375 (1964).
3. R. K. Böck, "Application of a generalized method of least squares for kinematical analysis of tracks in bubble chambers," CERN-60-30, (1960).
4. I. Derado and R. Leedy, "TEUTA - Kinematical analysis program for colliding beam events," Report No. SLAC-72, Stanford Linear Accelerator Center, Stanford University, Stanford, California (1967).

APPENDIX I

SUBROUTINE SOLVE (Q, M)

We copy here some of the most important instructions for the use of the SLAC computer library routine SOLVE as they were provided by Ch. Moore.

SOLVE has a number of parameters, three of which are explicit in the call statement SOLVE (Q, M):

Q (integer) specifies which entry point is intended: 1, 2, 3 or 4.

M (integer) has a meaning depending upon the value of Q.

SOLVE (real) advises the user as to the state of convergence.

However, the user must also employ certain implicit parameters by including in his program the data block:

COMMON /SOL / N, A (16, 4), W(4, 4)

where:

N (integer) is the number of variable parameters (maximum 15).

A (real) contains up to 4 equations (residuals and partial derivatives).

W (real) contains the weight matrix for the equations.

S = SOLVE(1, M): This call initializes certain parameters within SOLVE. M is used to set switches; it is coded as follows:

The units digit indicates whether the corrections obtained each iteration are to be printed (1) or not (0).

The tens digit specifies whether the first set of equations are to be used to obtain corrections (1) or only used to obtain scale factors (0).

The hundreds digit indicates how many iterations are to be permitted before admitting failure.

S = SOLVE(2, 1): This call will be employed within a loop to transmit a single equation. The arrays must be set as follows:

A(1, 1): the observed data minus the computed value of the function (residual).

A(2, 1) to A(N+1, 1): The partial derivatives of the function with respect to each of the variables.

W(1, 1): The weight of the equation - 1. /variance of data (inverse square of standard deviation of measurement).

M=1: Means the vector function has one component.

S = SOLVE(3, 0): When all data have been transmitted, this call solves for the corrections. A(2, 1) to A(N+1, 1) contain the corrections to be added to the variables. S indicates the state of convergence:

-1: Re-examine data for another iteration.

0: Solution has failed to converge.

greater than 0: Solution has been obtained and S is the standard deviation of the fit.

S = SOLVE(4, M): The units digit of M specifies whether the covariance matrix is to be printed (1) or not (0). The following are printed:

The standard deviation of the fit.

Chisquared (the sum of the (weighted) squares of the residuals, which was the quantity to be minimized).

Number of degrees of freedom (number of equations minus number of variables).

The standard deviations of the variables; in a vertical array to two significant figures (as defined by the normal equations).

The correlations between the variables; a lower triangular matrix, to two decimals.

If a variable was not fitted the last iteration, its standard deviation and correlations are not defined and are replaced by zero.

The tens digit of M specifies whether the user wants access to the covariance matrix (1) or not (0). If he does, it is rescaled and placed in the (real) array CA(N, N). To reference CA, the user must declare the data block:

```
COMMON /SOL2 / CA(16, 16), D(15)
```

Note that in the array A, the first variable is referenced by A(2, 1); but in the array CA it is CA(1, 1).