How to use the new ID calibration

JADE Computer note 94

J. SPITZER

March 26, 1987

ABSTRACT. Implementation of the new $r-\varphi$ calibration for the inner detector and the programs that can use it are described.

Implementation of the new ID calibration.

The new $r-\varphi$ calibration covers the entire JADE data taking period and provides a substantially better resolution than the standard calibration. The calibration data together with new run vertices are stored on the standard calibration files (F11LHO.AUPDAT1 e.t.c). KLREAD reads them into the common /CALIBR/ starting at the location associated with the name 'IDJS'. Subroutine KALIBR, which is standardly called now for each event, immediately after the call to KLREAD calls the new routine CNEWID(INDEX) which transforms the calibration data into the form used later. The input parameter INDEX is zero for these "normal" calls.

The selection of new or standard calibration can be changed with a single call to CNEWID in the following way: INDEX=-1 selects standard calibration, -2 new calibration with old run vertex overwritten with the new one (on each event) in the standard locations of /CALIBR/. (This latter feature can be supressed with -3 instead of -2 but since the vertex chamber software takes the vertex from a different place, -3 should not be used.) The new run vertex is in agreement in general with the old one but more detailed, each one corresponds to (typically) 200 single Bhabha tracks. The new selection will be in effect starting at the next event until an other call with these negative arguments. Normally, you will have to make the choice *before the first event is read in.* Probably, by the time of reading, the new calibration will be default and you have to make a call only if you need the standard calibration instead.

The new calibration has currently effect only for those programs which use JFETCH for reconstructing the hit coordinates. (It is not available in the graphics, in particular.) JFETCH has been modified to call the new reconstruction routine JFTNEW, having the same output conventions, in case a flag in a common set by CNEWID indicates that the new calibration was requested, otherwise the code in JFETCH will be executed.

Refitting tracks in $r-\varphi$.

JFETCH was written by P. Steffen for a final reconstruction of all hits assigned to a given track and used by his refitting package initiated by the subroutine FITEVR. For tracks originating from the run vertex (within 25 mm in $r-\varphi$), FITEVR calls for an $r-\varphi$ refit and subsequently for a common z vertex fit. The subroutine FRFITV which actually performs the $r-\varphi$ fit is basically the same as the routine REFITV but the latter can be used alone, independently from FITEVR.

These routines however were written for special applications and although the majority of physically interesting tracks gets refitted, for some tracks the old fit remains. Only parabola fit is provided and in case of a bad fit or/and large curvature, the third ring or even the second ring is omitted. This may be reasonable for tracks coming indeed from the centre of the detector (for which these routines were written), but may fail for V^0's originating in the second ring, for instance. The residual cut in these routines has not been optimized for the FADC resolution and in case a vertex constraint is requested it may happen that the residual cut removes so many hits that the number of degrees of freedom becomes zero and a divide check occours.

For the purpose of refitting essentially all tracks in $r-\varphi$, I've written a fitting routine that provides circle fit, a better, unified for DL8 and FADC hit cleaning procedure and a more appropriate handling of the vertex constraint. The vertex chamber hits have not been included but it is forseen to develope, together with the vertex chamber group, a routine using all the experience we have. (They are already using the new calibration provided in JFTNEW together with the vertex chamber hits in their private fitting routine.) The common r-z fit which is very important indeed, especially for the FADC data, has to be done with FITEVR. I'm planning to incorporate it into the new routine sometime in the future. In what follows, I describe first how the new routine(s) are called and then give certain details concerning the fit itself.

Subroutine XYRFTV(MODE).

This routine contains a loop over the tracks and selects the 'PATR' bank to be updated by the single track fitting routine. The input argument MODE has a similar meaning as the one for FITEVR:

(= 0) Overwrite old 'PATR' bank with new results

(= 1) Create new 'PATR' bank with new results

(+ 2) Not used (for FITEVR: also common z fit)

(+ 4) Vertex weakly constrained (ERRFAC = 100.0)

(+ 8) Not used (for FITEVR: rerun patrec in case of bad 'JHTL' bank)

(+16) No vertex constraint (ERRFAC = 1000.0)

In the most likely case, you don't need vertex contraint and you need to call with MODE = 16 or 17. Even if you ask for a vertex constraint (MODE = 0 or 1) in which case ERRFAC is set to 1.0, the vertex will not be used if it is incompatible with the track fitted without vertex constraint first (see later).

Subroutine XYRFT1(IPTR,IPJHTL,ERRFAC).

This routine performs the $r-\varphi$ fit for a single track (without the vertex chamber hits). IPTR is the pointer for the track in the pattern bank to be updated, IPJHTL is the pointer to the 'JHTL' bank (passed to JFETCH) and the meaning of the real parameter ERRFAC determining the strength of the vertex constraint has been indicated above. The routine is called from XYRFTV for each track in a loop but you may as well call it yourself for selected tracks only, with possibly different requests concerning the vertex constraint. Like any rou

tine calling JFETCH, it can be used with the standard as well as with the new calibration. The routine will update all parameters and only those which are determined in it (see the code for details), in particular it supplies a curvature error in the way it is done in FRFITV (the (more) standard routines set it to zero).

Bits set in the program identifier word of the 'PATR' bank.

The routine JFTNEW sets on the bit corresponding to 256 in the word IDATA(ITPO+2) to indicate that the new ID calibration was used. XYRFT1 sets on the bit 512 or 1024 in case the vertex constraint was actually used.

Some details concerning the $r-\varphi$ fit in XYRFT1.

The hits assigned to the given track are reconstructed by JFETCH (or JFTNEW) in the coordinate system with x-axis going through the first and last points of the track as given by the earlier fit, the origin being halfway in between. If the curvature from the earlier fit multiplied by the half distance between the first and last points does not exceed .04, parabola fit, otherwise circle fit is planned to be performed. The parabola fit needs a smaller amount of cpu time and hence is useful in case the data sample has mostly large momentum tracks. The algorithm for the circle fit is safe againt division by zero curvature and machine precision problems like subtraction of large numbers and could be used for small curvatures as well. The fit range is extended beyond the first and last points to recover hits frequently lost (mostly if the curvature is large) by the earlier fit.

Shift and rotation with curvature kept fixed to the original value is fitted first. Except for a very small fraction of tracks, the changes in the fit parameters are within certain specified limits and one may start the iterative fitting procedure and "hit cleaning". For those tracks which did not survive these cuts and for those for which an adjustment of the original curvature had previously been necessary in order to become compatible with the specified (by the original fit itself) first and last points (this occours for large curvature tracks fitted with parabola), a search for appropriate starting values is necessary or else the circle fit based on the linearized equations would later not converge. In this search, a parabola with three parameters is fitted to the residuals to the circle and the circle parameters are updated using three far points. The procedure is repeated 10 times altogether without checking any stop condition but reducing in each step the original very large 400 mm residual cut by a factor of 2 (not going below 10 mm). It is necessary to start with this very lose cut otherwise tracks will be lost. Very rarely, the circle fit will not converge even after this procedure, but the result obtained here will still be accepted.

In the iterative fit and hit cleaning part only a lose residual cut of 8 mm is used. At each iteration, the largest residual hits are exluded in such a way that they may be included later if their residual happens to decrease. The number of hits allowed to be thrown away, N/8 is proportional to the number of hits N available for the fit. The iteration continues until a stopping condition based on two parameters, the χ^2/d.o.f. and its rate of decrease for the next fit. (At larger χ^2/d.o.f. the iteration stops below a smaller decrease then at a smaller χ^2/d.o.f.) According to my experience, cutting on a single variable like residual, χ^2/d.o.f., its rate of decrease or χ^2-probability give very similar results but not as good as the one based on the two variables. Note that for the circle fit the linearized equations are used (two iterations each time) and instead of the distance between the points and the circle in radial

direction the distance along the y-axis is minimized.

The vertex is not included in the above procedure (it could only disturbe it). Instead, at the end, if vertex constraint was requested, it is checked whether the track is consistent with the run vertex by comparing the contribution of the vertex to the χ^2 with the final fit parameters and the χ^2/d.o.f. If a certain (safe) cut condition is satisfied, the fit is repeated (once, no more hits thrown away) with the vertex constraint otherwise the fit without it will be kept.

It has been checked using μ-pairs that this procedure gives slightly better results for the DL8 and definitely better results for the FADC than FRFITV in its current, not optimized for the FADC, form. Concerning low momentum tracks, they are in general longer when fitted with this routine, apparently the picture is better recognized. An example is included in the following figures. I have not seen a failure of the routine (but I have not tried very hard). I have looked at the cases in which a problem indicated by loss of too many hits occurred, the result was always reasonable and better than the earlier fit. If you detect a problem or have some suggestion, please, tell me. Good luck!