A New Multi-Vertex Fitter
and Updated Vertex-Locator Information

Mark Lakata
October 23, 1995 - Version 1.00

Abstract
A new vertex fitting package was written to replace the current fitting routines. The new routines do multi-vertex fitting, suited for displaced $K_S$ vertices, by fitting a new helix track to the SVX/PWC/JDC hits, with the constraint that the tracks pass through a common point. The old routines, by contrast, fit only one vertex (if at all) and did not fit the helices to the hits, but rather fitted a vertex to the helices. The new routines offer improved resolution and efficiency, especially in events with shorter tracks and 4-prong vertices.

Contents
1 Introduction 2
2 1-2-3 Instructions 2
3 TCVER3, Multivertex Configurator 3
4 TCVRHX, Full Helix & Vertex Fitting 7
5 TCVRHY, User Aide Routine 1 8
6 TCVRHZ, User Aide Routine 2 8
7 TCVRTX, Modified Old Software 10
8 Comparisons 10
9 Appendix A: Zebra Bank Details 15
10 Appendix B: Details of TCVRHX 17
11 Bibliography 20
1 Introduction

The first generation of vertex fitting was done by two routines: tcver3 and tcvertex. tcvertex decided which tracks to use in the fit and then called tcvertex to do the actual fitting. A new version of tcvertex was written, called tcver2 for use with the kinematic fitter. However, tcver2 does not make any attempt at finding the best combination of tracks to pair together; it simply makes all $2^N - N - 1$ combinations of $N$ tracks, and passes each combination to tcvertex for fitting. If the fit converges and passes $\chi^2$ cuts, then the vertex is kept. In an analysis of displaced vertices, such as for $K_S \rightarrow \pi^+ \pi^-$, one would like to have just one good configuration of pairings.

Thus tcver3 was born. tcver3 is best suited for looking for displaced vertices, and returns only those vertices of the best configuration. This allows one to make a cut on the number of prongs at each vertex of an event. For instance, for $p\bar{p} \rightarrow K^+ \pi^- K_S, K_S \rightarrow \pi^+ \pi^-$ typical has two separated vertices, the primary vertex near $(0,0,0)$ and the secondary $K_S$ decay vertex located a few cm away from $(0,0,0)$. In this case, a good event would be called $(2,2)$ since there are two vertices, each with two tracks. This is distinguished from $(4)$ which means one vertex with four tracks. The notation is that of the crystal barrel event display (cndisp V1.10/07 or newer).

Another serious problem with the old vertex fitter was that the positions of the vertices were only indirectly constrained by the actual raw data hits. Helix tracks were fitted to the raw data hits, resulting in 5-parameter helices. These 5-parameter helices were then used as the raw data input to a 3-parameter vertex fit. The error matrix does not constrain the vertex fit sufficiently enough, and the result is that a parameter such as the initial track direction can be modified by the fit much more than it should be. tcvertex addresses this problem by fitting the raw data hits directly, with 3-parameter helices that pass through a 3-parameter common point.

The event display (cndisp) was modified to automatically display multiple vertices. Use JD:VERT to display the vertices and their associated tracks. Also, the number of prongs of each vertex is listed, thus “0,2,2” would mean that there was one neutral vertex and two 2-prong vertices in the event.

2 1-2-3 Instructions

1. Configure choff to redo the vertex tracking. This involves putting the 'VERT' word after the CHAN card in the choff steering file, along with 'RTRK'. Thus the minimal line in the choff steering file for produced data would be:

CHAN 'TRAK' 'RAWS' 'PATT' 'CIRC' 'HELX' 'VERT' 'RTRK' 'GPWC'

You can substitute GVXT for GPWC if necessary.
2. Add these cards to your locater steering cards (or create new file if you
don’t already have a locater steering file). Note that there are other things
you can put in this file too, see sections 3, 4.

VERT 3
VFIT 2

3. Set Fortran Unit 81 to point to the locater steering file, and Fortran Unit
99 to point to the cfoff steering file. For example, on UNIX:

```bash
setenv FORT81 /user/analysis/locater.steer
setenv FORT99 /user/analysis/cfoff.steer
```

4. (optional) From your user routine, call a routine such as TCVRHY (see
section 5) to make cuts on the events based on the types of vertices found,
and a routine such as TCVRHZ to decode some of the vertex information
(see section 6).

5. (optional) If you concerned about CPU time limitations, and your data
still contains all the locater banks (such as TCHT, TCTK, TCHX etc.) (This
is not true for production data) then you can save some CPU time and
leave out the intermediate cards, for example.

```
CHAM 'TRAK' 'VERT' 'RTRK'
```

However, the official crystal barrel produced data only contains the TCTR
bank, so full retracking must be redone. Another idea is to first skim
the production data (without retracking) and looking for events with the
desired prong and PED multiplicities, and then to fully retrack (with
vertex fitting) the skimmed data on a second pass.

### 3 TCOVER3, Vertex Configuration Generator

**Call Arguments:** none
**Output:** bank structure at TCVX. See fig 5 and section 9
**Called from:** TCTRAX

TCOVER3 attempts to build up a vertex configuration starting by pairing
individual tracks, and then merging vertices if necessary. Here is a description
of the algorithm.

1. Make two tables of positively charged tracks and negatively charged tracks.
   Use only tracks that have no helix error, and which are at least V3MH
   SVX/PWC/JDC hits long. T3MH can be changed, and the default value is
   3.
Figure 1: Example of 2 double vertex fit, in the channel $K_s K^+ \pi^-$
2. Pair each + track to a - track, and attempt to fit a common vertex, with a call to \texttt{tcvertx}. This results in a list of vertices and the tracks connected to them.

3. Make all allowable configurations of vertices, using the above list of vertices, such that no track is used more than once in a configuration.

4. Pick the configuration that has the lowest reduced $\chi^2$, that is the minimum value of

$$\frac{\chi^2}{n} = \frac{\sum_{\text{vertices}} \chi^2}{\sum_{\text{vertices}} N_{df}}$$

Now we have a list of the best 2-prong vertices.

optional step: Let $\vec{\tilde{r}}_{ij}$ = difference of positions of any two vertices $i$ and $j$ and $\vec{p}_i$ = the total momentum of all tracks at vertex $i$, which assumes that the vertex is the result of a neutral particle decay into charged particles. If $\vec{\tilde{r}}_{ij} \cdot \vec{p}_i > 0$, that can mean that the neutral particle that decayed at vertex $i$ is moving towards (rather than away from) vertex $j$. Since this is an unlikely event and likely to be a wrong combination of tracks, a $\chi^2$ penalty of \texttt{V3BA} is added to the configuration. \texttt{V3BA} can be changed, and the default value is 100.

5. For the remaining unmatched tracks, fit a pseudo-vertex to each single track (best guess of the vertex).

6. At this point, we have a list of 2-prong and 1-prong vertices ($v_i$) which includes every track.

7. Now attempt to combine these vertices together. For each vertex ($v_i$), try to attach every other vertex ($v_j, j \neq i$) to it.

(a) Find the distance between the vertices $v_i$ and $v_j$. Reject those that are greater than \texttt{V3DI} cm apart. \texttt{V3DI} can be changed, and the default is 3.

$$\vec{\tilde{r}}_{ij} = \vec{v}_i - \vec{v}_j$$

Reject if $|\vec{\tilde{r}}_{ij}| > \texttt{V3DI} \text{ cm}$

(b) Calculate the $\chi^2$ for the hypothesis that $v_i \equiv v_j$. Reject those pairings with $< \texttt{V3CL} \% \text{ CL. } \texttt{V3CL}$ can be changed, and the default is 0.01.

$$CL = \text{PROB}(\chi^2, N_{df} = 3)$$
\[
\chi^2 = \hat{r}_{ij}(C_i + C_j)^{-1}\hat{r}_{ij}
\]

\[
C_i = \begin{pmatrix} 
\sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\
\sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\
\sigma_{zx} & \sigma_{zy} & \sigma_{zz}
\end{pmatrix}
\]

(c) Repeat the above two steps for all the vertices \( v_j \).

(d) Once we have a list of vertices \( (v_i, v_j, v_{j2}, \ldots) \) that could possibly go together, attempt to fit all the tracks to a common vertex.

8. Drop all unused vertices, and keep the good ones in TCXX, update TCTR banks.

The end result is that every track will be paired to one and only one vertex. These vertices will have 1 or more tracks attached to them. Exception: Global tracking adds one “neutral” vertex, with error code 1000, which is by default \( (0,0,0) \), and used as the vertex for all PEDs. Thus every event has 1 “neutral” vertex (regardless if there are PEDs or not) and zero or more “charged” vertices. See figure 1 for a good example of a two vertex fit.

### Controlling TCVER3

To use vertex fitting at all, in the cboff steering card file (Fortran Unit 99) use the 'VERT' option with the CHAN card. Then use VERT 3 to use TCVER3.

These are the steering cards for locater and cboff that involve TCVER3.

The one card that is probably most important is V3MH, the minimum number of hits on a track. The default value of 3 is the minimum from the helix fit, but in reality a value of 5-7 is more reasonable. I have used 7 in the comparison tests, as it seems to cleanup noisy events better, but keep in mind that it also cuts down on the solid angle acceptance. With 2-prong, long-track triggered data, there is no reason that this value be smaller than 7.

<table>
<thead>
<tr>
<th>Steers</th>
<th>Card</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>locater</td>
<td>cboff</td>
<td>CHAN 'VERT'</td>
<td>off</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VERT ( n )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = TCVERT One vertex fit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 = TCVER2 CBKFIT vertex fit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 = TCVER3 Multivertex fit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>V3CL ( CL_{min} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>V3DI ( d_{max} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>V3BA ( \chi^2_{pen} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>V3MH ( n_{hit} )</td>
</tr>
</tbody>
</table>
4 TCVRHX, Full Helix & Vertex Fitting

Call Arguments: CALL TCVRHX(NTRKS,NTRK,ITCVT,IERR,XGUES)

- INTEGER NTRKS - number of tracks in NTRK()
- INTEGER NTRK() - array of TCTK track numbers.
- INTEGER ITCVT - bank pointer to empty vertex bank.
- REAL XGUES(3) - initial guess of vertex.

Output: bank structure at TCVT. See section 9

Called from: TCVER3

The current vertex fitter TCVRTEX uses the results of the helix fit as the raw input to a fit. Thus the vertex is fitted against the constraints of the helix parameters (5 values per track) rather than against the JDC/PWC/SVX hits. In general, this leads to a systematic error caused by the particular parameterization method of the helices. For example, one of the helix parameters, $\psi_0$, gives the initial angle of the center of circle, which is equal to the initial angle of the momentum up to a constant of $\pi/2$. This quantity is highly constrained by the hits, especially the hits in the outer layers, since they have a big “lever-arm” on the track.

However, without the hits as inputs to the fit, values like $\phi_0$ begin to wander, more than they should. For short tracks, this value can change by several degrees, thus moving the fitted track far from the original hits. This is clearly unsatisfactory.

The new vertex fitting routine is based on the helix fitting routine. Instead of fitting several tracks independently, each with 5 helix parameters, the new fit fits all tracks simultaneously, each with 3 helix parameters, plus 3 more parameters for the location of the vertex. Thus the number of fit parameters goes from $5N$ to $3 + 3N$, which is smaller for $N \geq 2$. The fit is done by iterating the method of Lagrange multipliers.\(^1\)

If only one track is attempted in the fit, then assign a vertex position that is either

- the closest approach to (0, 0, 0) IF the hit information is missing or if the first hit is at JDC layer 3 or before.
- the position of the first hit of the track otherwise.

Controlling TCVRHX

To use vertex fitting at all, in the choff steering card file (Fortran Unit 99) use the 'VERT' option with the CHAM card. Then use VERT 3 and VFIT 2 to use TCVRHX. TCVRHX only works with TCVER3!\(^1\)

\(^1\)Brandt, Siegmund, Statistical and Computational Methods in Data Analysis
These are the steering cards for LOCATER that involve TCVRHX. Some of the parameters from TCVRHX are also used... see the source code if you really care.

<table>
<thead>
<tr>
<th>steer</th>
<th>Card</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
</table>
| locater | VFIT n | 1 | Selects which vertex fit to do  
1 = TCVRHX Fits to helix parameters  
2 = TCVRHX Fits to wire hits |
| VHER &verbar; n | VHCT \((\chi^2/n)_{\text{thresh}}\) | 0.001 | Maximum change in \(\chi^2/n\) for convergence |
| VHGC \((\chi^2/n)_{\text{greatest}}\) | VHER \(f\) | 1.0 | Hit error estimate scaling factor |
| VHGC \((\chi^2/n)_{\text{greatest}}\) | VHER \(f\) | 3.0 | Greatest \(\chi^2/n\) at end of fit |
| VHGC \((\chi^2/n)_{\text{greatest}}\) | VHER \(f\) | 50.0 | Greatest \(\chi^2/n\) during fit. |

5 TCVRH Y, User Aide Routine 1

Call Arguments: CALL TCVRH Y(NVERT, NPRONG, IERR)  
INTEGER NVERT - number of vertices (not counting neutral)  
INTEGER NPRONG(10) - array of prongness of vertices.  
INTEGER IERR - number of vertices with error code greater than 100.  

This routine returns an array, where each element gives the number of vertices with that prongness. For example, a two vertex event, each with two tracks, would have NPRONG(2)=2 and the rest of NPRONG(x)=0. A one vertex event with four tracks would have NPRONG(4)=1. Note NVERT = \(\sum_{i=1}^{10} i \times NPRONG(i)\).

For example, in USER:

SUBROUTINE USER

INTEGER NVERT,NPRONG(10),IERR

CALL TCVRH Y(NVERT,NPRONG, IERR)  
IF (NVERT.EQ.2 .AND. NPRONG(2).EQ.2 .AND. IERR.EQ.0) THEN
  *  
  * do something interesting with the event
  *  
  ENDF
  RETURN
END

6 TCVRHZ, User Aide Routine 2

Call Arguments: CALL TCVZH (NPRONG, IVERT, E, PX, PY, PZ, CHRG, MASS, X, T, RCHI, ITCVT, IERR)
input:

NPRONG - how many prongs desired
IVERT - Ith vertex with that many prongs

output:

E(10) - energies of outgoing tracks (1..NPRONG)
PX(10) - momentum x (1..NPRONG)
PY(10) - momentum y (1..NPRONG)
PZ(10) - momentum z (1..NPRONG)
CHRG(10) - charges (1..NPRONG)
MASS - invariant mass of vertex
X(3) - (x, y, z) position of vertex
T(4) - \( \left( p_x, p_y, p_z, \epsilon \right) \) momentum of vertex
RCHI - Reduced \( \chi^2/n \) of vertex
ITCVT - pointer to vertex zebra bank
IERR - return code

This routine returns various pieces of information (given above) for a particular vertex. The user should call TVUVNRY first to determine how many of a certain vertex there are, e.g., how many 2-prong vertices there are. Then the user can call TVVHZ with that prong value (e.g., 2), and ask for the first, second, third, etc. vertex with that prong value. For example, to extract the information for a two vertex event, each vertex having two prongs, use this example code:

```
INTEGER NVERT,NPRONG(10),IERR
INTEGER ITCVT
REAL E(10),PX(10),PY(10),PZ(10),CHRG(10),MASS,X(3),CHI,T(4)

CALL TVVHY(NVERT,NPRONG,IERR)
IF (NVERT.EQ.2.AND.NPRONG(2).EQ.2.AND.IERR.EQ.0) THEN
  DO I=1,2
  * the '2' argument is to indicate that we want a 2 prong vertex
  * the 'I' argument is to indicate which one (first or second)
    CALL TVVHZ(2,I,E,PX,PY,PZ,CHRG,MASS,X,T,CHI,
               & ITCVT,IERR)
  *
  * do something here ...
  *
  ...
ENDDO
ENDIF
```
7 TCVRTX, Modified Old Software

There was one modification to this routine. The definition of the parameter, TVRTX, was extended. If the value is positive, then it represents the number of the vertex bank to use, e.g. 1 to N. If the value is negative, then the absolute value is a ZEBRA pointer to a TCVT bank. This modification is totally backward compatible with TCVERT and TCVER2, which use the first case, and is also compatible with TCVER3 which uses the second case.

8 Comparisons

$K_S$ Monte Carlo, Mass Resolution

10000 $k_s k^+ \pi^-$ Monte Carlo events were analyzed, by plotting the invariant masses of found 2-prong vertices. A gaussian plus linear polynomial was fitted to the $\pi^+ \pi^-$ invariant mass. TCVRTX and TCVRHX were each separately used along with TCVER3, and the results given in the table below. Note that the width of the peak is significantly reduced from 11.7 MeV to 10.6 MeV, an improvement of 9%. See figure 2. The values for background and SNR are given assuming that one makes a 1 sigma cut on the $K_S$ mass.

Also, the reconstructed vertex position was compared to the MC generated vertex position, by measuring the distance ($\delta r$) between the two in the $xy$ plane, in the $z$ direction and also in $xyz$ space. Since the distance distributions have long tails, a better comparison is made with the log of the distance. See figure 3 and the table below. The best improvement is in the $xy$ plane, where the mean of the log($\delta r_{xy}$) dropped from -1.39 to -1.66, which corresponds to the $\delta r_{xy}$ distances of .25 cm and .19 cm respectively. See figure 3.

<table>
<thead>
<tr>
<th></th>
<th>TCVRTX (old)</th>
<th>TCVRHX (new)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Events in $K_S$ peak</td>
<td>2900 ± 80</td>
<td>3270 ± 90</td>
</tr>
<tr>
<td>Width (MeV)</td>
<td>11.7 ± 0.3</td>
<td>10.6 ± 0.3</td>
</tr>
<tr>
<td>Mean (MeV)</td>
<td>496.4 ± 0.3</td>
<td>496.3 ± 0.2</td>
</tr>
<tr>
<td>Background (1 sigma cut)</td>
<td>270 ± 40</td>
<td>250 ± 70</td>
</tr>
<tr>
<td>SNR (1 sigma cut)</td>
<td>7.3 ± 1.2</td>
<td>8.9 ± 2.3</td>
</tr>
<tr>
<td>log($\delta r_{xy}$)</td>
<td>-1.39</td>
<td>-1.66</td>
</tr>
<tr>
<td>$\sim \delta r_{xy}$</td>
<td>0.25 cm</td>
<td>0.19 cm</td>
</tr>
<tr>
<td>log($\delta r_z$)</td>
<td>-1.12</td>
<td>-1.26</td>
</tr>
<tr>
<td>$\sim \delta r_z$</td>
<td>0.33 cm</td>
<td>0.29 cm</td>
</tr>
<tr>
<td>log($\delta r_{xyz}$)</td>
<td>-0.59</td>
<td>-0.77</td>
</tr>
<tr>
<td>$\sim \delta r_{xyz}$</td>
<td>0.55 cm</td>
<td>0.46 cm</td>
</tr>
</tbody>
</table>
Figure 2: $K_S$ peak in $\pi^+\pi^-$ invariant mass. (a) TCVRTX (old fitter) (b) TCVRHX (new fitter), fitted with a gaussian plus linear function.
Figure 3: Vertex reconstruction error, log scale. (a) in $xy$ plane (b) in $z$ dimension. Solid line is $\text{tcvrmx}$, dashed line is $\text{tcvrtx}$. 
Figure 4: Missing mass squared histograms for (a) helix fit results, (b) TCVRTX fit results and (c) TCVRHX fit results. The dashed line represents $m_{\pi^0}^2$, while the solid line is the mean.
$\pi^+\pi^-\pi^0$ Test

June 1994 2-prong data (164517 events) were scanned for “Gold-plated $\pi^+\pi^-\pi^0$” events, by using these cuts:

1. Default production
2. Events must have 2 Prongs (88.5 %)
3. Events must have no type 13 PEDs (76.3 %)
4. Use only SMART type 0 PEDs, ignore the rest
5. Use only PEDs above 10 MeV, ignore the rest
6. Events must have 2 remaining unmatched PEDs (11.7 %)
7. • (case A) Kinematic Fit of 10% CL or greater to the hypothesis $\pi^+\pi^-\pi^0$ (37.8 %)
   • (case B) PI0FND finds exactly one $\pi^0$ (79.0 %)

This selects 4896 events for case A and 7650 events for case B. The events are then processed again, with full reconstruction, including both types of vertex fitting, TCVRTX and TCVRHX. The momenta from the vertex bank are used to calculate the “missing mass” of the event, using just the charged tracks (or in other words the recoil mass of the two tracks). This should give a peak at the $\pi^0$ mass. Since the resolution is so poor in both cases, the mass-squared peak is smeared out into forbidden negative regions, yielding unphysical values of mass. However, the distributions peak near $m_{\pi^0}^2$ (see figure 4). The values of the square root of the mean and the RMS width are tabulated, and show that the old TCVRTX and new TCVRHX fits give similar results for both cases A and B (see table below). TCVRHX gives a small improvement in the RMS width at a slight efficiency cost.

The errors quoted for the means are

$$\sigma_{\sqrt{\pi}} = \frac{\sigma_{RMS}}{2\sqrt{N_{\text{entries}}}}$$

and are not necessarily one sigma gaussian errors. Errors in RMS width are roughly $\text{RMS}/\sqrt{N} \sim 0.001$.

<table>
<thead>
<tr>
<th></th>
<th>TCHELX</th>
<th>TCVRTX</th>
<th>TCVRHX</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI0FND</td>
<td>4896</td>
<td>4889</td>
<td>4870</td>
</tr>
<tr>
<td>$\sqrt{\text{Mean}}$ (MeV)</td>
<td>268 ± 2</td>
<td>159 ± 4</td>
<td>162 ± 4</td>
</tr>
<tr>
<td>RMS (GeV^2)</td>
<td>0.088</td>
<td>0.081</td>
<td>0.081</td>
</tr>
<tr>
<td>CBFKTI</td>
<td>7650</td>
<td>7605</td>
<td>7522</td>
</tr>
<tr>
<td>$\sqrt{\text{Mean}}$ (MeV)</td>
<td>273 ± 2</td>
<td>179 ± 3</td>
<td>179 ± 3</td>
</tr>
<tr>
<td>RMS (GeV^2)</td>
<td>0.110</td>
<td>0.103</td>
<td>0.099</td>
</tr>
</tbody>
</table>
9 Appendix A: Zebra Bank Details

The bank format structure has not been changed, except that now the TCVT bank may be filled with more than one vertex (TCVERT only outputted one vertex at most). As a summary, all the bank structures that are given in the locater manual are also given herein, plus the previously undocumented bank of TCVH is also presented. The bank structure is given in the following tables, and the layout is given graphically in fig 5.

![Bank Structure Diagram]

Figure 5: Layout of the bank structure containing all found vertices.

**TCVX: The master vertex bank**

<table>
<thead>
<tr>
<th>Link</th>
<th>Variable</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITCTVN</td>
<td>LQ(LTCVX-N)</td>
<td>link to N\textsuperscript{th} vertex</td>
</tr>
<tr>
<td>ITCTVT</td>
<td>LQ(LTCVX-1)</td>
<td>link to first vertex</td>
</tr>
<tr>
<td></td>
<td>LQ(LTCVX+1)</td>
<td>number of vertices</td>
</tr>
</tbody>
</table>
TCVT: The parameters of the vertex

<table>
<thead>
<tr>
<th>Link</th>
<th>Variable</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITCVT$_N$ = LQ(ITCVT-#)</td>
<td></td>
<td>link to N$^{th}$ vertex</td>
</tr>
<tr>
<td>ITCVP = LQ(ITCVT-1)</td>
<td></td>
<td>link to ITCVP</td>
</tr>
<tr>
<td>IQ(ITCVT+1)</td>
<td></td>
<td>number of tracks at vertex</td>
</tr>
<tr>
<td>IQ(ITCVT+2)</td>
<td></td>
<td>number of neutrals at vertex</td>
</tr>
<tr>
<td>IQ(ITCVT+3)</td>
<td></td>
<td>Error code</td>
</tr>
<tr>
<td>IQ(ITCVT+4)</td>
<td></td>
<td>Number of degrees of freedom in fit</td>
</tr>
<tr>
<td>Q(ITCVT+5)</td>
<td></td>
<td>$x$ [cm] of vertex</td>
</tr>
<tr>
<td>Q(ITCVT+6)</td>
<td></td>
<td>$y$ [cm] of vertex</td>
</tr>
<tr>
<td>Q(ITCVT+7)</td>
<td></td>
<td>$z$ [cm] of vertex</td>
</tr>
<tr>
<td>Q(ITCVT+8)</td>
<td></td>
<td>$\sigma_{xx}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVT+9)</td>
<td></td>
<td>$\sigma_{xy}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVT+10)</td>
<td></td>
<td>$\sigma_{yy}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVT+11)</td>
<td></td>
<td>$\sigma_{xz}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVT+12)</td>
<td></td>
<td>$\sigma_{yz}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVT+13)</td>
<td></td>
<td>$\chi^2$</td>
</tr>
<tr>
<td>Q(ITCVT+14)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TCVP: Improved parameters of tracks at the vertex

<table>
<thead>
<tr>
<th>Link</th>
<th>Variable</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITCVP = LQ(ITCVT-1)</td>
<td></td>
<td>link to ITCVP</td>
</tr>
<tr>
<td>ITCVH = LQ(ITCVP)</td>
<td></td>
<td>link to ITCVH</td>
</tr>
<tr>
<td>IQ(ITCVP+1)</td>
<td></td>
<td>Track number in TCTR</td>
</tr>
<tr>
<td>IQ(ITCVP+2)</td>
<td></td>
<td>Quality Word</td>
</tr>
<tr>
<td>Q(ITCVP+3)</td>
<td></td>
<td>Charge</td>
</tr>
<tr>
<td>Q(ITCVP+4)</td>
<td></td>
<td>$P_x$ [MeV/c]</td>
</tr>
<tr>
<td>Q(ITCVP+5)</td>
<td></td>
<td>$P_y$ [MeV/c]</td>
</tr>
<tr>
<td>Q(ITCVP+6)</td>
<td></td>
<td>$P_z$ [MeV/c]</td>
</tr>
<tr>
<td>Q(ITCVP+7)</td>
<td></td>
<td>$E$ [MeV]</td>
</tr>
<tr>
<td>Q(ITCVP+8)</td>
<td></td>
<td>$\sigma_{xx}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVP+9)</td>
<td></td>
<td>$\sigma_{xy}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVP+10)</td>
<td></td>
<td>$\sigma_{yy}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVP+11)</td>
<td></td>
<td>$\sigma_{xz}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVP+12)</td>
<td></td>
<td>$\sigma_{yz}$ [cm$^2$]</td>
</tr>
<tr>
<td>Q(ITCVP+13)</td>
<td></td>
<td>$\sigma_{xx}$ [cm$^2$]</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>1 to 13 repeated for other tracks</td>
</tr>
</tbody>
</table>

To access the Nth track, use ITCVH = LQ(ITCVT-1)+LENVP*(N-1). LENVP is defined in trkprm.inc or with +SEQ, TRKPRM.
TCVH: improved helix parameters for the tracks at the vertex

See locater manual (CB Note 93,123) for explanation of helix parameters.

<table>
<thead>
<tr>
<th>Link</th>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITCVH</td>
<td>LQ(ITCVP)</td>
<td>link to ITCVH</td>
</tr>
<tr>
<td>Q(ITCVH+1)</td>
<td>r₀</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+2)</td>
<td>z₀</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+3)</td>
<td>α</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+4)</td>
<td>tan λ</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+5)</td>
<td>ψ₀</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+6)</td>
<td>s</td>
<td></td>
</tr>
<tr>
<td>Q(ITCVH+7)</td>
<td>λ²</td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>1 to 7 repeated for other tracks</td>
<td></td>
</tr>
</tbody>
</table>

To access the Nth track, use ITCVH = LQ(ITCVP)+LENVH*(N-1). LENVH is defined in trkprm.inc or with *SEQ,TRKPRM.

10 Appendix B: Details of TCVRHX

The parameters (see fig. 6) are:

- (xᵥ, yᵥ, zᵥ) The position of the vertex.
- α The curvature of the track. The momentum is BνGTC/α.
- φ₀ The position of the vertex w.r.t. the center of the circle.
- tan λ The z dependence.
- s The “sign” of the track. The true charge is s×BνGNC (a function of the sign of the magnetic field). The direction of motion around the circle is defined by s. If s = -1, the particle circulates counter-clockwise, while if s = +1, the particle circulates clockwise in the xy plane.
- β is the free parameter, that varies from 0 at the vertex to +∞. In the fit, these are quantized as βᵢ for each hit i.

The helix parameterization used in tcvrhx is as follows:

\[
x(\beta) = x_v - \frac{\cos \phi_0}{\alpha} + \frac{\cos(\phi_0 - s\beta)}{\alpha} \quad (1)
\]

\[
y(\beta) = y_v - \frac{\sin \phi_0}{\alpha} + \frac{\sin(\phi_0 - s\beta)}{\alpha} \quad (2)
\]

\[
z(\beta) = z_v + \frac{\tan \lambda}{\alpha} \beta \quad (3)
\]
Figure 6: New vs. old helix parameterizations. Note, $\beta = 0$ at vertex, otherwise $\beta > 0$ always.
Here are some relationships between this parameterization and the standard TCHELX helix parameterization.

\[
\Psi_0 = \psi_0 - s \frac{\pi}{2} = \text{ATAN2}(y_b - \frac{\sin \phi_0}{\alpha}, x_v - \frac{\cos \phi_0}{\alpha})
\]

\[
r_0 = s \left[ \sqrt{\left( x_v - \frac{\cos \phi_0}{\alpha} \right)^2 + \left( y_b - \frac{\sin \phi_0}{\alpha} \right)^2} - 1/\alpha \right]
\]

\[
z_0 = z_v + \frac{s \tan \lambda}{\alpha} (\Psi_0 - \phi_0 + \pi)_{\alpha \leq \pi, \pi < \alpha}
\]

The routine TCVRHIX uses the same fitting algorithm of TCHELX, except that the number of fitted parameters and input variables is different.

<table>
<thead>
<tr>
<th>Input variables</th>
<th>TCHELX</th>
<th>TCVRHIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fitted parameters</td>
<td>3N Values</td>
<td>3 + 3M Values</td>
</tr>
<tr>
<td>... for 1 track</td>
<td>5 Values</td>
<td>-</td>
</tr>
<tr>
<td>... for 2 tracks</td>
<td>-</td>
<td>9</td>
</tr>
<tr>
<td>... for 4 tracks</td>
<td>-</td>
<td>15</td>
</tr>
<tr>
<td>(Y Vector)</td>
<td>(r_0, z_0, \alpha, \tan \lambda, \psi_0)</td>
<td>(x_v, y_b, z_v, \alpha^1, \tan \lambda^1, \phi_0^1, \alpha^2, \tan \lambda^2, \phi_0^2, \ldots)</td>
</tr>
<tr>
<td>Degrees of Freedom</td>
<td>3N - 5</td>
<td>3 (\sum_{i \in \text{tracks}} N_i - 1 - M)</td>
</tr>
<tr>
<td>... for 1 track</td>
<td>\sim 64</td>
<td>-</td>
</tr>
<tr>
<td>... for 2 tracks</td>
<td>-</td>
<td>\sim 129</td>
</tr>
<tr>
<td>... for 4 tracks</td>
<td>-</td>
<td>\sim 261</td>
</tr>
</tbody>
</table>

Where \( N \) = number of hits on a track and \( M \) = number of tracks at the vertex.

The equations of constraint and derivatives of them with respect to the various input and output variables are given below.

\[
C_1 = \frac{1}{\alpha} (\cos^2 \beta_i + \sin^2 \beta_i - 1) 
\]

\[
C_2 = x_v - z_i + \frac{\tan \lambda \beta_i}{\alpha} 
\]

where \( \beta_i = s (\phi_0 - \text{ATAN2}(\alpha(y_b - y_h) + \sin \phi_0, \alpha)(x_i - x_v) + \cos \phi_0) \)

The derivatives with respect to the hits (\( dx_v, dy_v, dz_v \)) and to the fitted helices (\( dx_v, dy_v, dz_v, d\alpha, d\tan \lambda, d\phi_0 \)) are:

\[
\frac{dC_1}{dx_v} = 2 \cos \beta_i 
\]

\[
\frac{dC_1}{dy_v} = 2 \sin \beta_i 
\]
\[ \frac{dC_1}{dz_i} = 0 \] (9)
\[ \frac{dC_2}{dx_i} = \frac{s \tan \lambda \sin \beta_i}{\cos^2 \beta_i + \sin^2 \beta_i} \] (10)
\[ \frac{dC_2}{dy_i} = -\frac{s \tan \lambda \cos \beta_i}{\cos^2 \beta_i + \sin^2 \beta_i} \] (11)
\[ \frac{dC_2}{dz_i} = -1 \] (12)
\[ \frac{dC_1}{dx_v} = -\frac{dC_1}{dx_i} \] (13)
\[ \frac{dC_1}{dy_v} = -\frac{dC_1}{dy_i} \] (14)
\[ \frac{dC_1}{dz_v} = -\frac{dC_1}{dz_i} \] (15)
\[ \frac{dC_1}{d\alpha} = \frac{2}{\alpha} \left[ \cos \beta_i (x_i - x_v) + \sin \beta_i (y_i - y_v) \right] - \frac{1}{\alpha} C_1 \] (16)
\[ \frac{dC_1}{d\tan \lambda} = 0 \] (17)
\[ \frac{dC_1}{d\phi_0} = \frac{2}{\alpha} \left( \sin \beta_i \cos \phi_0 - \cos \beta_i \sin \phi_0 \right) \] (18)
\[ \frac{dC_2}{dx_v} = -\frac{dC_2}{dx_i} \] (19)
\[ \frac{dC_2}{dy_v} = -\frac{dC_2}{dy_i} \] (20)
\[ \frac{dC_2}{dz_v} = -\frac{dC_2}{dz_i} \] (21)
\[ \frac{dC_2}{d\alpha} = \tan \lambda \left( \frac{\frac{2}{\alpha} \left[ \cos \beta_i (y_i - y_v) - \sin \beta_i (x_i - x_v) \right]}{\cos^2 \beta_i + \sin^2 \beta_i} \right) \] (22)
\[ \frac{dC_2}{d\tan \lambda} = \frac{\beta_i}{\alpha} \] (23)
\[ \frac{dC_2}{d\phi_0} = -\frac{s \tan \lambda}{\alpha} \left( \frac{\cos \beta_i \cos \phi_0 + \sin \beta_i \sin \phi}{\cos^2 \beta_i + \sin^2 \beta_i} - 1 \right) \] (24)

11 Bibliography