Vertex Reconstruction And Fitting

Thomas Lohse

Humboldt University, Berlin

Contents

1 Introduction

This software package is designed as a flexible tool kit for vertex recognition and vertex fitting for HERA-B and similar experiments. The general assumption is that vertices are located close to the interaction point in a region without magnetic field where tracks are straight.

The package needs parameters of tracks and their covariance matrices as input. They have to be determined from any track fitting program. In principle, the track fits and the vertex fit have to be performed simultaneously or have to be iterated because both kinds of fits are non-linear. Since the track direction and the impact parameters are, however, dominated by the vertex detector (where tracks are straight lines), the non-linearity in the fit of track parameters is small and the χ^2-function is almost parabolic in the track parameters. Therefore the vertex fit can be performed in a fully decoupled way after the track fits without re-iteration.

This note is divided into two parts. Section 2 describes the mathematics of vertex fitting. Section 3 serves as a users guide for the software. Section 2 does not have to be studied in order to understand section 3.

2 Mathematics of Vertex Fits

2.1 The General Kalman Filter Method

The vertex fit is based on a Kalman filter technique, described in reference [1]. It allows to construct flexible and relatively fast algorithms for vertex recognition and vertex fitting. A short summary of the basic relations is given in this section.

Let $\vec{p}_k$ be the d-dimensional vector containing the parameters of track k, like slopes, offsets and momentum ($d = 5$). Let $V_k = \mathrm{cov}(\vec{p}_k)$ be the covariance matrix obtained from the track fit, and let $G_k = V_k^{-1}$. The vertex fit determines new parameters,

$\vec{x} = (x, y, z)$	the vertex position
$\vec{q}_k$	the track parameters for track k, constrained to originate in $\vec{x}$.

Because of the vertex constraint, $\dim(\vec{q}_k) = d - 2$. The fit therefore has $2n - 3$ degrees of freedom, if n is the total number of tracks participating in the vertex fit[1].

The covariance matrix of the parameters of the vertex fit can be split into sub-matrices, corresponding to

- the vertex part,

$$C = \mathrm{cov}(\vec{x}) \qquad (3 \times 3) \tag{1}$$

- the track part,

$$D_k = \mathrm{cov}(\vec{q}_k) \qquad ((d-2) \times (d-2)), \tag{2}$$

[1] If the vertex is constrained due to external conditions, like beam sizes, in j of the three spatial coordinates, the number of degrees of freedom increases to $2n - 3 + j$.

- the correlations between the vertex and the tracks,

$$E_k \; = \; \mathrm{cor}(\vec{x}, \vec{q}_k) \qquad\qquad (3 \times (d-2)), \tag{3}$$

- and the correlations between any two different tracks $k \neq l$,

$$\mathrm{cor}(\vec{q}_k, \vec{q}_l) \qquad\qquad ((d-2) \times (d-2)) \tag{4}$$

The filter step uses a linearized measurement equation, derived from the map h_k of fit parameters $(\vec{x}, \vec{q}_k)$ into the space of track parameters $(\vec{p}_k)$,

$$(\vec{x}, \vec{q}_k) \mapsto \vec{h}_k(\vec{x}, \vec{q}_k) \quad . \tag{5}$$

Expansion to first order yields

$$\vec{h}_k(\vec{x}, \vec{q}_k) \; = \; \vec{c}_k^{(0)} \; + \; A_k \cdot \vec{x} \; + \; B_k \cdot \vec{q}_k \quad , \tag{6}$$

with

$$\vec{c}_k^{(0)} \;=\; \vec{h}_k(\vec{x}^{(0)}, \vec{q}_k^{(0)}) \; - \; A_k \cdot \vec{x}^{(0)} \; - \; B_k \cdot \vec{q}_k^{(0)} \tag{7}$$

$$A_k \;=\; \left.\frac{\partial \vec{h}_k}{\partial \vec{x}}\right|_{\vec{x}^{(0)}, \vec{q}_k^{(0)}} \tag{8}$$

$$B_k \;=\; \left.\frac{\partial \vec{h}_k}{\partial \vec{q}_k}\right|_{\vec{x}^{(0)}, \vec{q}_k^{(0)}} \quad , \tag{9}$$

where $\vec{x}^{(0)}$ is the best known estimate of the vertex position, and $\vec{q}_k^{(0)}$ initially corresponds to the track parameters $\vec{p}_k$ obtained from the track fit.

The Kalman filter proceeds in the following steps:

i) Initialize the vertex position by an estimate $\vec{x}_0$ with covariance matrix C_0. This initialization might be adapted to a known constraint, like the beam position ($\vec{x}_0$) and its r.m.s. size (C_0), or $\vec{x}_0$ might be a guess, in which case C_0 must be set to infinity.

ii) Add one track after the other to the vertex and update the vertex fit: Let $\vec{x}_{k-1}$ be the vertex position with the first $k-1$ tracks participating in the fit. The track k is then included by the following recursion relations:

<u>Covariance matrices:</u>

$$\begin{aligned} C_k &= \left(C_{k-1}^{-1} \; + \; A_k^\top G_k^B A_k\right)^{-1} \\ E_k &= -F_k C_k \\ D_k &= W_k \; - \; E_k F_k^\top \quad , \end{aligned} \tag{10}$$

with

$$\begin{aligned} G_k^B &= G_k \; - \; G_k B_k W_k B_k^\top G_k \\ W_k &= \left(B_k^\top G_k B_k\right)^{-1} \\ F_k &= W_k B_k^\top G_k A_k \quad . \end{aligned} \tag{11}$$

The computation of E_k, D_k in the filter step can be omitted, if the smoother is run at the end (see below).

New fit values:

$$\vec{x}_k = C_k \cdot \left[C_{k-1}^{-1} \cdot \vec{x}_{k-1} + A_k^\top G_k^B \cdot \left(\vec{p}_k - \vec{c}_k^{(0)} \right) \right] \tag{12}$$

$$\vec{q}_k = W_k B_k^\top G_k \cdot \left[\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_k \right] \quad . \tag{13}$$

New χ^2:

$$\begin{aligned} \chi_k^2 = \ & \chi_{k-1}^2 \\ & + \left(\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_k - B_k \cdot \vec{q}_k \right)^\top G_k \left(\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_k - B_k \cdot \vec{q}_k \right) \\ & + \left(\vec{x}_k - \vec{x}_{k-1} \right)^\top C_{k-1}^{-1} \left(\vec{x}_k - \vec{x}_{k-1} \right) \quad , \end{aligned} \tag{14}$$

with

$$\chi_0^2 = 0 \quad . \tag{15}$$

iii) Once all tracks are added, the fit parameters have to be smoothed, in order to update each filter step with the information of all measured tracks:

$$\vec{x}_k^n = \vec{x}_n \tag{16}$$

$$\vec{q}_k^n = W_k B_k^\top G_k \cdot \left(\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_n \right) \quad . \tag{17}$$

iv) The procedure has to be iterated with the measurement equation expanded around these new (improved) fit parameters after smoothing, until convergence is reached.

v) After convergence, the covariance matrices of the smoothed fit parameters are computed:

$$\begin{aligned} C_k^n &= C_n \\ D_k^n &= W_k - E_k^n F_k^\top \\ E_k^n &= -F_k C_n \quad . \end{aligned} \tag{18}$$

The correlations between two tracks $k \neq l$ are given by

$$\mathrm{cor}(\vec{q}_k^n, \vec{q}_l^n) = F_k C_n F_l^\top \quad . \tag{19}$$

2.2 The Inverse Kalman Filter

For the purpose of vertex recognition it is vital to have a flexible and fast tool for vertex modification. The Kalman filter fulfills these requirements while a global vertex fit algorithm does not.

The most important vertex operations are:

i) Add a track onto a vertex which has been fit already. This is trivial in the Kalman filter language, since starting from $\vec{x}_n$ and C_n, a single additional filter step is sufficient to add the new track as track $n+1$. A single call to the smoother computes the updated track parameters. Since the vertex was already fit with n tracks, iteration might not be necessary at all, or at least the convergence will be relatively rapid.

ii) Remove track k from the vertex. This requires an inverse filter step which is again trivial in the language of Kalman filtering, if covariance matrices of all filter steps are stored. In a global method, the new vertex would have to be fit from scratch. The inverse filter step modifies the vertex position $\vec{x}_n$ and its covariance matrix C_n according to

$$C_n \mapsto C_k^{n\star} = \left(C_n^{-1} - A_k^\top G_k^B A_k\right)^{-1} \tag{20}$$

$$\vec{x}_n \mapsto \vec{x}_k^{n\star} = C_k^{n\star} \cdot \left[C_n^{-1} \cdot \vec{x}_n - A_k^\top G_k^B \left(\vec{p}_k - \vec{c}_k^{(0)}\right)\right] \quad , \tag{21}$$

and the χ^2 of the fit is reduced to

$$\chi_n^2 \mapsto \chi_n^2 - \chi_{k,S}^2 \quad , \tag{22}$$

with

$$\begin{aligned}\chi_{k,S}^2 = & \left(\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_n - B_k \cdot \vec{q}_k^n\right)^\top G_k \left(\vec{p}_k - \vec{c}_k^{(0)} - A_k \cdot \vec{x}_n - B_k \cdot \vec{q}_k^n\right) \\ & + \left(\vec{x}_n - \vec{x}_k^{n\star}\right)^\top \left(C_k^{n\star}\right)^{-1} \left(\vec{x}_n - \vec{x}_k^{n\star}\right) \quad .\end{aligned} \tag{23}$$

If, based on the reduction in χ^2, it is decided to leave the track out, the smoother has to be used in order to find the new smoothed track parameters. A re-iteration is advisable, but normally convergence will be rapid.

2.3 Implementation for HERA-B

The tracks are straight lines under small angles with respect to the z-axis. The target is assumed to be positioned near $z = 0$. The straight lines are parametrized as

$$z \mapsto \vec{a} \cdot z + \vec{b} \quad , \tag{24}$$

such that $\vec{a} = (a_x, a_y)^\top$ and $\vec{b} = (b_x, b_y)^\top$ are obviously the most suitable track parameters. A fifth parameter is the measured momentum which does not influence the fit of the vertex position and can therefore be in principle completely ignored. On the other hand, because the momentum is in general correlated with the track parameters $\vec{a}$ and $\vec{b}$, the momentum measurement is also improved by the vertex constraint so that normally all five track parameters should be passed to the vertex fit. Also, the momentum measurement has to be necessarily used, if mass constraints are to be obeyed by the vertex fit. The track parameters are thus chosen to be[2]

$$\vec{p} = (a_x, a_y, b_x, b_y, p)^\top \quad . \tag{25}$$

After the vertex constraint, the track is described by the slope and the momentum alone:

$$\vec{q} = (a_x, a_y, p) \quad . \tag{26}$$

[2] In principle, a more suitable parameter, instead of p, is Q/p, since the track model is more linear in this variable. As long as the momentum resolution is good, $\delta p/p \ll 1$, the additional non-linearity introduced by the more intuitive choice of p as track parameter is small.

The measurement equation thus reads

$$\vec{h}_k(\vec{x}, \vec{q}_k) = \begin{pmatrix} q_{k_1} \\ q_{k_2} \\ x_1 - x_3 \cdot q_{k_1} \\ x_2 - x_3 \cdot q_{k_2} \\ q_{k_3} \end{pmatrix} \quad . \tag{27}$$

This map is clearly non-linear in the third and fourth component so that iteration is unavoidable. Linearization yields:

$$A_k = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & -q_{k_1}^{(0)} \\ 0 & 1 & -q_{k_2}^{(0)} \\ 0 & 0 & 0 \end{pmatrix} \quad B_k = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_3^{(0)} & 0 & 0 \\ 0 & -x_3^{(0)} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\vec{c}_k^{(0)} = \begin{pmatrix} 0 \\ 0 \\ x_3^{(0)} \cdot q_{k_1}^{(0)} \\ x_3^{(0)} \cdot q_{k_2}^{(0)} \\ 0 \end{pmatrix} \quad , \tag{28}$$

where $\vec{x}^{(0)}$ is the best estimate of the vertex position and $\vec{q}_k^{(0)}$ are the best known track parameters (slopes and momentum) of track k (initially given by $p_{k_1}, p_{k_2}, p_{k_5}$).

The parameter arrays in the linearized measurement equation are hence quite simple (many elements are zero). The matrices W_k, G_k^B, F_k, E_k, D_k, C_k^{-1} can be directly obtained from their definition, and stay correspondingly simple so that it is practical to construct them by hand for explicit implementation in a computer program.

2.4 Mass Constraints

Often, one or several combinations of particles on a (secondary) vertex are known to originate from a narrow width mass state. In this case, the precision of the parameters obtained in the vertex fit can be improved, if all mass constraints are taken into account in the fit.

Since the constraints in general introduce complicated correlations between all fit parameters, the covariance matrix cannot any longer be easily split into sub-matrices. It is therefore best to introduce the $(3n+3)$-dimensional fit vector

$$\vec{\alpha} = (\vec{x}, \vec{q}_1, \vec{q}_2, \ldots, \vec{q}_n)^\top \quad , \tag{29}$$

with the covariance matrix

$$V_\alpha = \mathrm{cov}(\vec{\alpha}) \quad . \tag{30}$$

Before the mass constraints are introduced, V_α can obviously be constructed from C_n, D_k^n, E_k^n, and $\mathrm{cor}(\vec{q}_k^n, \vec{q}_l^n)$.

The constraints change the best estimate $\vec{\alpha}$ from the unconstrained vertex fit to a new vector $\vec{\alpha}_c$. The mass constraints are expressed in the form

$$\vec{g}(\vec{\alpha}_c) = 0 \quad , \tag{31}$$

where every component of $\vec{g} = (g_1, \ldots, g_r)$ represents one constraint equation. In linear approximation, $\vec{\alpha}_c$ can be determined analytically by minimizing

$$\chi^2 = (\vec{\alpha}_c - \vec{\alpha})^\top V_\alpha^{-1} (\vec{\alpha}_c - \vec{\alpha}) + 2\vec{\mu}^\top \vec{g}(\vec{\alpha}_c) \quad , \tag{32}$$

where $\vec{\mu}^\top = (\mu_1, \ldots, \mu_r)$ is the vector of r Lagrange multipliers introducing the constraints. The fit has r degrees of freedom.

In order to solve for $\vec{\alpha}_c$, $\vec{g}$ is expanded around an estimated solution $\vec{\alpha}^{(0)}$ (initially given by the vector $\vec{\alpha}$ from the unconstrained vertex fit):

$$\vec{g}(\vec{\alpha}) = \vec{g}(\vec{\alpha}^{(0)}) + \Gamma \cdot (\vec{\alpha} - \vec{\alpha}^{(0)}) + \mathcal{O}((\vec{\alpha} - \vec{\alpha}^{(0)})^2) \quad . \tag{33}$$

The matrix Γ is given by

$$\Gamma_{li} = \left. \frac{\partial g_l}{\partial \alpha_i} \right|_{\vec{\alpha}^{(0)}} \qquad \begin{array}{l} l = 1, \ldots, r \\ i = 1, \ldots, 3n+3 \end{array} \quad . \tag{34}$$

Minimizing χ^2 yields:

$$\begin{array}{lcl} \dfrac{\partial \chi^2}{\partial \vec{\alpha}_c} = 0 & \Longrightarrow & \Gamma \cdot \vec{\alpha}_c + \Gamma V_\alpha \Gamma^\top \cdot \vec{\mu} = \Gamma \cdot \vec{\alpha} \\ \dfrac{\partial \chi^2}{\partial \vec{\mu}} = 0 & \Longrightarrow & \Gamma \cdot \vec{\alpha}_c + \vec{g}(\vec{\alpha}^{(0)}) = \Gamma \vec{\alpha}^{(0)} \quad . \end{array} \tag{35}$$

Subtracting the two equations allows to compute the Lagrange multipliers,

$$\vec{\mu} = \left(\Gamma V_\alpha \Gamma^\top \right)^{-1} \cdot \left[\vec{g}(\vec{\alpha}^{(0)}) + \Gamma \cdot (\vec{\alpha} - \vec{\alpha}^{(0)}) \right] \quad . \tag{36}$$

Plugging this into the first equation, one finds the solution:

$$\vec{\alpha}_c = \vec{\alpha} - V_\alpha \Gamma^\top \left(\Gamma V_\alpha \Gamma^\top \right)^{-1} \cdot \left[\vec{g}(\vec{\alpha}^{(0)}) + \Gamma \cdot (\vec{\alpha} - \vec{\alpha}^{(0)}) \right] \quad . \tag{37}$$

The new vector $\vec{\alpha}_c$ can now replace $\vec{\alpha}^{(0)}$, and can be used as new expansion point for the Kalman filter, and the whole procedure can be iterated into convergence.

Finally, from the result for $\vec{\alpha}_c$, one can easily derive the covariance matrix

$$\mathrm{cov}(\vec{\alpha}_c) = V_\alpha - V_\alpha \Gamma^\top \left(\Gamma V_\alpha \Gamma^\top \right)^{-1} \Gamma V_\alpha \quad , \tag{38}$$

and the χ^2 of the constrained fit,

$$\chi^2 = \left[\vec{g}(\vec{\alpha}^{(0)}) + \Gamma(\vec{\alpha} - \vec{\alpha}^{(0)}) \right]^\top \left(\Gamma V_\alpha \Gamma^\top \right)^{-1} \left[\vec{g}(\vec{\alpha}^{(0)}) + \Gamma(\vec{\alpha} - \vec{\alpha}^{(0)}) \right] \quad . \tag{39}$$

It is not difficult to obtain closed expressions for the elements of Γ from the mass constraints. Let Z_l be the set of indices of those tracks which are required by mass constraint l to form the invariant mass M_l. From the track parameters

$$\vec{q} = (a_x, a_y, p)$$

one first computes for each track the direction

$$\vec{n} = \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \frac{1}{\sqrt{1 + a_x^2 + a_y^2}} \cdot \begin{pmatrix} a_x \\ a_y \\ 1 \end{pmatrix} \quad , \tag{40}$$

and the momentum vector $\vec{n} \cdot p$. Using the symbols

$$S_l^E = \sum_{i \epsilon Z_l} E_i \tag{41}$$

$$\vec{S}_l = \sum_{i \epsilon Z_l} p_i \cdot \vec{n}_i \quad , \tag{42}$$

where $E_i = \sqrt{p_i^2 + m_i^2}$ is the energy of particle i (m_i is the mass of particle i), the constraint reads

$$g_l(\vec{\alpha}) = S_l^{E^2} - \vec{S}_l^2 - M_l^2 \quad , \tag{43}$$

and the partial derivatives in Γ_{li} can be written in the following compact form: For $i \epsilon Z_l$

$$\frac{\partial g_l}{\partial a_{xi}} = -2 p_i n_{zi} \cdot \begin{pmatrix} 1 - n_{xi}^2 \\ -n_{xi} n_{yi} \\ -n_{xi} n_{zi} \end{pmatrix}^\top \cdot \vec{S}_l \tag{44}$$

$$\frac{\partial g_l}{\partial a_{yi}} = -2 p_i n_{zi} \cdot \begin{pmatrix} -n_{xi} n_{yi} \\ 1 - n_{yi}^2 \\ -n_{yi} n_{zi} \end{pmatrix}^\top \cdot \vec{S}_l \tag{45}$$

$$\frac{\partial g_l}{\partial p_i} = 2 \frac{p_i}{E_i} S_l^E - 2 \vec{n}_i^\top \vec{S}_l \quad , \tag{46}$$

while for $i \not\epsilon Z_l$ all partial derivatives of g_l vanish.

2.5 Decay Chain Fit with Mass Constraints

The ultimate precision of vertices and track parameters can be reached if besides possible mass constraints a topological constraint is used for the case of a fully reconstructed secondary vertex: the sum of momentum vectors of all particles on the secondary vertex has to point back to the primary vertex. This constraint correlates the secondary and primary vertices so that, in principle, the fit parameters consist of the secondary vertex position, the distance between the primary and the secondary vertex, and the track parameters $\vec{q}_i$ of all tracks on either the primary or the secondary vertex. Since multiplicities from the primary vertex are large, this leads to huge covariance matrices with the usual numerical problems in matrix operations. Since it is, however, very unlikely that the (small) effect of the topological constraint on the parameters of individual tracks is ever used in an analysis, the problem is reduced to a precise fit of the parameters of the mother particle (e.g. a B^0), decaying at the secondary vertex. These parameters are

$$\vec{\beta} = (L, \vec{x}_S, a_x, a_y, P) \quad , \tag{47}$$

where

$$\begin{aligned} L &= \text{distance between primary and secondary vertex} \\ \vec{x}_S &= \text{secondary vertex position} \\ a_x, a_y &= \text{slopes of the mother track before the decay} \\ P &= \text{total momentum of the mother particle} \quad . \end{aligned}$$

The topological constraint can again be implemented by means of a χ^2 minimization. Let $\vec{\alpha}^{(0)}$ be the $(3n+3)$-dimensional fit vector of the (possibly mass constrained) secondary vertex, and let $V_\alpha = \mathrm{cov}(\vec{\alpha}^{(0)})$. Let $\vec{x}_P$ be the primary vertex location with $C_P = \mathrm{cov}(\vec{x}_P)$ (again from a possibly mass constrained vertex fit). Let $\vec{\beta}^{(0)}$ be the vector $\vec{\beta}$, evaluated at $\vec{\alpha}^{(0)}$ and for $L = |\vec{x}_S - \vec{x}_P|$. Let $\vec{P} = P \cdot \vec{n}$ be the sum of momenta of all tracks on the secondary vertex. The χ^2 for the fit parameters $\vec{\beta}_c$ then reads

$$\chi^2(\vec{\beta}_c) = \left(\vec{\beta}_c - \vec{\beta}^{(0)}\right)^\top J^\top V_\beta^{-1} J \left(\vec{\beta}_c - \vec{\beta}^{(0)}\right) + \vec{\Delta}(\vec{\beta}_c)^\top C_P^{-1} \vec{\Delta}(\vec{\beta}_c) \quad , \tag{48}$$

with

$$J_{ij} = \delta_{i+1,j} \quad \text{for} \quad \begin{cases} i = 1, \ldots, 6 \\ j = 1, \ldots, 7 \end{cases} \tag{49}$$

$$V_\beta = J N V_\alpha N^\top J^\top \tag{50}$$

$$N = \left. \frac{\partial \vec{\beta}}{\partial \vec{\alpha}} \right|_{\vec{\alpha}^{(0)}} \quad , \tag{51}$$

and

$$\vec{\Delta}(\vec{\beta}) = \vec{x}_P + L \cdot \vec{n} - \vec{x}_S = \vec{\Delta}(\vec{\beta}^{(0)}) + M \cdot \left(\vec{\beta} - \vec{\beta}^{(0)}\right) + \mathcal{O}\left((\vec{\beta} - \vec{\beta}^{(0)})^2\right) \tag{52}$$

$$M = \left. \frac{\partial \vec{\Delta}}{\partial \vec{\beta}} \right|_{\vec{\beta}^{(0)}} \quad . \tag{53}$$

The first term in equation (48) originates from the deviation of the secondary vertex position and the momentum of the mother particle from the parameters obtained in the secondary vertex fit, the second term originates from the mismatch between the fitted vertex positions and the decay length L.

The partial derivatives in M and N read:

$$\frac{\partial L}{\partial \vec{\alpha}} = \frac{\partial \vec{x}_S}{\partial \vec{q}_i} = \frac{\partial a_x}{\partial \vec{x}_S} = \frac{\partial a_y}{\partial \vec{x}_S} = \frac{\partial P}{\partial \vec{x}_S} = 0$$

$$
\begin{aligned}
\frac{\partial x_{Si}}{\partial x_{Sj}} &= \delta_{ij} \\
\frac{\partial a_{x,y}}{\partial q_{1,2_i}} &= \frac{q_{3_i} n_{zi}}{P_z} \cdot \left(1 \,-\, n_{x,y_i}^2 \,+\, \frac{P_{x,y}}{P_z} n_{x,y_i} n_{zi} \right) \\
\frac{\partial a_{x,y}}{\partial q_{2,1_i}} &= \frac{q_{3_i} n_{zi} n_{y,x_i}}{P_z} \cdot \left(\frac{P_{x,y}}{P_z} n_{zi} \,-\, n_{x,y_i} \right) \\
\frac{\partial a_{x,y}}{\partial q_{3_i}} &= \frac{1}{P_z} \cdot \left(n_{x,y_i} \,-\, \frac{P_{x,y}}{P_z} n_{zi} \right) \\
\frac{\partial P}{\partial q_{1_i}} &= q_{3_i} n_{zi} \vec{n}^\top \cdot \begin{pmatrix} 1 \,-\, n_{xi}^2 \\ -n_{xi} n_{y_i} \\ -n_{xi} n_{zi} \end{pmatrix} \\
\frac{\partial P}{\partial q_{2_i}} &= q_{3_i} n_{zi} \vec{n}^\top \cdot \begin{pmatrix} -n_{xi} n_{y_i} \\ 1 \,-\, n_{y_i}^2 \\ -n_{xi} n_{zi} \end{pmatrix} \\
\frac{\partial P}{\partial q_{3_i}} &= \vec{n}^\top \vec{n}_i \\
\frac{\partial \vec{\Delta}}{\partial L} &= \vec{n} \\
\frac{\partial \Delta_i}{\partial x_{Sj}} &= -\delta_{ij} \\
\frac{\partial \vec{\Delta}}{\partial a_x} &= L \cdot n_z \cdot \begin{pmatrix} 1 - n_x^2 \\ -n_x n_y \\ -n_x n_z \end{pmatrix} \\
\frac{\partial \vec{\Delta}}{\partial a_y} &= L \cdot n_z \cdot \begin{pmatrix} -n_x n_y \\ 1 - n_y^2 \\ -n_y n_z \end{pmatrix} \\
\frac{\partial \vec{\Delta}}{\partial P} &= 0 \quad ,
\end{aligned}
\tag{54}
$$

where

$$
\begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} = P \cdot \vec{n} = \sum_{i \,\epsilon\, \text{sec. vert.}} q_{i_3} \cdot \vec{n}_i \quad . \tag{55}
$$

Minimization of the χ^2-function is straightforward and yields

$$
\vec{\beta}_c = W J^\top V_\beta^{-1} J \cdot \vec{\beta} + W M^\top C_P^{-1} M \cdot \vec{\beta}^{(0)} - W M^\top C_P^{-1} \cdot \vec{\Delta}^{(0)} \tag{56}
$$

$$
\operatorname{cov}(\vec{\beta}_c) = W \tag{57}
$$

with

$$
W = \left[J^\top V_\beta^{-1} J \,+\, M^\top C_P^{-1} M \right]^{-1} \quad . \tag{58}
$$

The improved position of the primary vertex which follows from $\vec{\beta}_c$, is plugged into the smoother equation for the primary vertex. Thus the updated momenta of tracks on the primary vertex are computed. The change in the secondary vertex parameters $\vec{\alpha}$ is derived directly from the change in $\vec{\beta}$:

$$\vec{\alpha} \mapsto \vec{\alpha}_c = \vec{\alpha} - V_\alpha N^\top J^\top V_\beta^{-1} J \cdot \left[N \cdot \left(\vec{\alpha} - \vec{\alpha}^{(0)} \right) - \vec{\beta}_c + \vec{\beta}^{(0)} \right] \quad . \tag{59}$$

Finally, the filter steps of the vertices are run again with these updated vertex and track parameters, and the fit for $\vec{\beta}$ is repeated with $\vec{\beta}^{(0)}$ set to $\vec{\beta}_c$. The whole procedure is reiterated until convergence is reached.

3 Documentation of the Vertex Package

3.1 Location of Software

The vertex package can be copied from the public account on x4u2.desy.de:

- Location: /home/x4u2/ftp/pub/herab/vertex
- If you copy via ftp, then type:

...	ftp x4u2.desy.de
Name (...):	anonymous
Password:	anything
ftp >	cd pub/herab/vertex
ftp >	get vt*

This note resides on the same directory under "vertex.ps".

There are two blocks of utility routines:

1. **vtfilt.f:** The general Kalman filter fitting package for vertices. It needs **vtdat.i** as include file.
2. **vtpatt.f:** The general toolbox for vertex recognition, including some first algorithms for primary and secondary vertex finding. It uses **vtfilt.f** and the include files **vtdat.i**, **vtpatt.i**, **vtwire.i**.

The routines are completely stand-alone FORTRAN programs which just require the CERN library (routine DSINV for matrix inversion).

3.2 The Data Structure

3.2.1 vtdat.i: Filter Information

The include file contains four common blocks of data structures, and several parameters, like some frequently used particle masses.

1. **Parameters:** The following parameters serve as internal buffer dimensions. The user doesn't have to care about them (normally).

ntrtot	maximum number of tracks in track buffer	(i4)
nvttrk	maximum number of vertices a single track can be assigned to	(i4)
nvtmax	maximum number of different vertices	(i4)
ntrk	maximum number of tracks per vertex	(i4)
ncomax	maximum number of mass constraints per vertex	(i4)

The following parameters are set to mass values of frequently used particles:

mPsi	J/ψ mass in GeV	(r8)
mKs	K^0_S mass in GeV	(r8)
mB	B^0 mass in GeV	(r8)
mpi	$\pi^\pm$ mass in GeV	(r8)
mmu	$\mu^\pm$ mass in GeV	(r8)
me	$e^\pm$ mass in GeV	(r8)

2. <u>Track Buffer:</u> **common /vttrbf/**

   This contains track parameters and their covariance matrices. They have to be filled by the user or an interface program to a track fit.

   (a) **trmult** actual number of tracks in buffer (i4)

   (b) **p(5,ntrtot)** track parameters (r8)

p(1,i)	slope $a_x = \mathrm{d}x/\mathrm{d}z$
p(2,i)	slope $a_y = \mathrm{d}y/\mathrm{d}z$
p(3,i)	impact parameter b_x [cm] at $z = 0$
p(4,i)	impact parameter b_y [cm] at $z = 0$
p(5,i)	track momentum [GeV]

   The fifth component can be left open if vertices are fit with straight tracks of unknown momentum.

   (c) **V(5,5,ntrtot)** covariance matrix of track parameters (r8)
   Again the fifth row and column can be ignored if vertices are fit for tracks with unknown momentum.

   (d) **G(5,5,ntrtot)** inverse of V(5,5,ntrtot) (r8)
   These matrices are computed internally. The user doesn't have to provide them.

   (e) **xmass(ntrtot)** mass of the tracks (r8)
   This array only has to be filled if mass constraints are to be applied, and if so, only for those tracks participating in the constraints.

3. <u>**Pointer buffers for track/vertex relations:**</u> **common /vtpntr/**

   These pointers administrate the track/vertex relations internally. The user should never touch them. The pointers are listed here for illustration only.

   (a) **nvt** Actual number of vertices (i4)

(b) **ktr(nvtmax,ntrk)** Pointer into main track buffer (i4)

ktr(iv,it) is the pointer into the main track buffer for the itth track on the vertex with index iv.

(c) **ntr(nvtmax)** Total number of tracks on a vertex (i4)

(d) **nv(ntrtot)** Total number of vertices assigned to a track in the main track buffer (i4)

(e) **kv(ntrtot,nvttrk)** Track/vertex pointer (i4)

kv(it,ia) is the vertex index for the iath vertex assignment to the track with index it in the main track buffer (set to zero if there is no such assignment).

(f) **jv(nvtmax,ntrtot)** Vertex/track pointer (i4)

The vertex with index iv is the jv(iv,it)th vertex assigned to the track with index it in the main track buffer (jv(iv,it) is set to zero if the track with index it is not on the vertex iv).

(g) **vtset(nvtmax)** Status of vertex (logical)

This status flag is set, if a vertex index is in use.

4. **<u>Kalman filter information:</u> common /vtkalm/**

This common contains the full information, internally needed by the Kalman filter, but also the fit results relevant to the user. Only the latter are described here.

(a) **xvs(3,nvtmax)** Fit result for vertex position (r8)

xvs(i,iv), i=1, 2, 3, are the x, y, z components of the vertex with index iv from the fit.

(b) **qvs(3,nvtmax,ntrk)** Fit result for track parameters (r8)

qvs(i,iv,itv), i=1, 2, 3, are the slopes a_x, a_y and the momentum p of the itvth track on the vertex with index iv. The tracks are constrained to originate from the vertex.

(c) **CS(3,3,nvtmax)** Covariance matrix of xvs (r8)

(d) **DS(3,3,nvtmax,ntrk)** Covariance matrix for qvs (r8)

(e) **chi2(nvtmax,ntrk)** χ^2 values for vertex fit (r8)

chi2(iv,itv) is the χ^2 from the vertex fit of the vertex with index iv, when tracks 1...itv participate in the fit. The total χ^2 of the fit is hence chi2(iv,ntr(iv)).

The meaning of the other matrices (filter information before smoothing, correlation terms between different tracks or tracks and vertices) are self-explaining when compared to the nomenclature in section 2.

5. **<u>Constraint buffer:</u> common /vtcons/**

These buffers store the mass constraints for the vertices. They are administrated internally and the user should not touch them.

(a) **xmc(nvtmax,ncomax)** Mass values for constraints (r8)
xmc(iv,ic) is the invariant mass for the icth mass constraint defined for the vertex with index iv.

(b) **nconst(nvtmax)** Number of mass constraints on a vertex (i4)

(c) **mask(nvtmax,ncomax,ntrk)** Mask for defining tracks participating in a mass constraint (logical)
mask(iv,ic,itv) is set to .true., if the itvth track on the vertex with index iv is participating in the icth mass constraint on that vertex. Otherwise, mask(iv,ic,itv) is set to .false..

3.2.2 vtwire.i: Wire Geometry Information

This information describes the relevant constraints for vertices, as provided by the knowledge of wire positions in HERA-B. This is irrelevant for general purpose vertex fits.

1. **Parameters:**

   (a) **nWmax** size of the wire buffers (i4)

2. **Common /vtwire/:**

   (a) **nw** actual number of wires (i4).

   (b) **xw(3,nwmax)** space coordinate of center of wires (r8)
   xw(i,iw), i=$1,2,3$, are the x, y, z components (in [cm]) of the center of the iwth wire, when rotated around y and z axis into the x, y-plane and into an orientation parallel to the x axis.

   (c) **phiw(nwmax)** azimuthal orientation of the iwth wire (r8)
   The angle 0 corresponds to a horizontal wire. The wire is rotated around the z-axis in the mathematically positive sense.

   (d) **thew(nwmax)** dip angle of wires along the beam direction (r8)
   The wire is rotated around the y-axis, before the azimuthal rotation is applied. The wire is assumed to be oriented along the x-axis before the rotations are applied.

   (e) **sigtw(nwmax)** width of wires (r8)
   sigtw(iw) is the r.m.s. width of the iwth wire (in [cm]).

   (f) **siglw(nwmax)** length of wires (r8)
   siglw(iw) is the r.m.s. length of the iwth wire (in [cm]).

   (g) **Cw(3,3,nwmax)** covariance matrices for vertex constraint due to wires (r8)
   This covariance matrix is filled automatically using the geometry information. The user should not bother about it.

3.2.3 vtpatt.i: Vertex Recognition Information

These parameters ond common blocks are only needed if HERA-B specific pattern recognition for vertices is attempted. It is irrelevant for general purpose vertex fits.

There are three common blocks and a number of steering parameters for primary and secondary vertex recognition. The common blocks contain vertex information relevant for the user. The steering parameters may be changed for tuning the vertex recognition quality.

1. **Assignment of tracks to wires**
   (a) **Parameters:**
      i. **ch2wtr** maximum χ^2-distance for track/wire assignment (r4)
      ii. **ch2wB** maximum χ^2-distance for B^0/wire assignment (r4)
   (b) **common /vttrwi/:**
      i. **ch2wi(nwmax,ntrtot)** buffer of track/wire χ^2-distances (r8)
      ii. **iwitr(ntrtot)** buffer of closest wires (i4)
      iwitr(it) is the index of the wire with smallest χ^2-distance to the track with index it.
      iii. **ntrwi(nwmax)** buffer of track multiplicity on wires (i4)
      ntrwi(iw) is the number of tracks assigned to the iwth wire.
      iv. **itrwi(nwmax,ntrtot)** track index for a given wire (i4)
      The track with index it is the itrwi(iw,it)th track assigned to the iwth wire (itrwi(iw,it)=0 if there is no assignment).
2. **Vertices of the $B^0 \to J/\psi K_S^0$ decay**
   (a) **Parameters:** none
   (b) **common /ivpsik/:**
      i. **ivpsik** secondary vertex index for the decay $B^0 \to J/\psi K_S^0$ (i4)
      ii. **ivb** primary vertex index for the B^0 origin (i4)
3. **Assignment of tracks to primary vertices**
   (a) **Parameters:**
      i. **xlmin, xlwid** parameters for clustering tracks which have been assigned to a common wire (r4)
      xlmin is the minimal hit separation along a wire and xlwid is the maximum width of a cluster. The clusters are used as seeds for primary vertices.
      ii. **ntmin** minimum number of tracks required on a primary vertex (i4)
      The only exception is the primary vertex of the B^0 which is kept in any case (even as intersect of the B^0 direction of flight with the wire).
      iii. **ch2vtr** maximum χ^2-distance of a track to a primary vertex seed for track/vertex merging (r4)

(b) **common /ivprim/:**

i. **nvprim** number of reconstructed primary vertices (i4)

ii. **ivprim(nvtmax)** primary vertex indices (i4)
ivprim(ivp) is the vertex index of the ivpth primary vertex.

4. Assignment of tracks to secondary vertices

(a) **Parameters:**

i. **ch2stt** maximum χ^2 for merging two tracks into a new vertex (r4)

ii. **ch2stv** maximum χ^2 for adding a track to a vertex (r4)

iii. **ch2svw** minimum χ^2-distance of a secondary vertex from a wire (r4)

iv. **Bvimpc** maximum impact parameter (in [cm]) to the B^0 vertex if the search is restricted to the vicinity of the B^0 (e.g. for charm decays) (r4)

v. **nmix** maximum number of primary vertex tracks assignable to a secondary vertex (i4)

(b) **common /ivsec/:**

i. **nvsec** number of reconstructed secondary vertices (i4)

ii. **ivsec(nvtmax)** secondary vertex indices (i4)
ivsec(ivs) is the vertex index of the ivsth secondary vertex.

5. Merging of vertices (secondary or primary)

(a) **Parameters:**

i. **ch2vv** maximum χ^2-distance of two vertices for vertex merging (r4)

(b) **Common blocks** none

3.3 vtfilt.f: The Vertex Fitting Package

The package contains many auxiliary routines for internal use. This documentation is restricted to those routines which the user may have to call directly.

3.3.1 VTZERO: General Initialization

Name: **VTZERO** (subroutine)

Calling Sequence: VTZERO

Input: None

Output: None

Action: Reset all vertex and track buffers and pointers.

Comment: To be called at the beginning of a new event.

3.3.2 VTINIT: Initialize New Vertex

Name:	**VTINIT** (subroutine)		
Calling Sequence:	VTINIT(**x**,**cov**,**iv**,**flag**)		
Input:	**x(3)**	vertex position (in [cm])	(r8)
	cov(3,3)	covariance matrix of x(3)	(r8)
Output:	**iv**	index of generated vertex	(i4)
	flag	status flag (= 0 if ok)	(i4)
Action:	Initialize a new vertex at x with errors cov.		
Comment:	If the vertex position is a guess, the covariance matrix has to be set to a diagonal matrix, where the diagonal elements are much larger than the square of the corresponding sizes of the vertex region under consideration (e.g. the target area).		

3.3.3 VTDEL: Delete a Vertex

Name:	**VTDEL** (subroutine)		
Calling Sequence:	VTDEL(**iv**,**flag**)		
Input:	**iv**	vertex index	(i4)
Output:	**flag**	status flag (= 0 if ok)	(i4)
Action:	Delete vertex iv.		
Comment:	Fails only if non-existent vertex is attemted to be deleted.		

3.3.4 VTADD: Add a Track to a Vertex

Name:	**VTADD** (subroutine)		
Calling Sequence:	VTADD(**iv**,**it**,**flag**)		
Input:	**iv**	vertex index	(i4)
	it	track index in main track buffer	(i4)
Output:	**flag**	status flag (= 0 if ok)	(i4)
Action:	Add track with index it to vertex with identifier iv.		
Comment:	If the vertex has already been fit, VTADD automatically updates the fit results without obeying possible mass constraints.		

3.3.5 VTKILL: Remove a Track from a Vertex

Name:	**VTKILL** (subroutine)		
Calling Sequence:	VTKILL(**iv**,**itv**,**flag**)		
Input:	**iv**	vertex index	(i4)
	itv	track index for vertex	(i4)
Output:	**flag**	status flag (= 0 if ok)	(i4)
Action:	Remove the itvth track from the vertex with identifier iv and (re)-fit the vertex.		
Comment:	The fit does not obey mass constraints.		

3.3.6 VTFIT: Fit a Vertex Without Mass Constraints

Name:	**VTFIT** (subroutine)		
Calling Sequence:	VTFIT(**iv**,**momflg**,**flag**)		
Input:	**iv**	vertex index	(i4)
	momflg	mode of fit	(i4)
Output:	**flag**	status flag (= 0 if ok)	(i4)
Action:	Fit the vertex with identifier iv with no constraints. For momflg=1, the track momenta are included in the fit. For momflg=0, the vertex fit is done without momentum information on the basis of straight tracks.		
Comment:	The fit results are stored in common /vtkalm/. The fit may fail because no convergence is reached. This can happen for low multiplicity vertices with almost parallel tracks.		

3.3.7 VTADDC: Add a Mass Constraint to a Vertex

Name:	**VTADDC** (subroutine)		
Calling Sequence:	VTADDC(**iv**,**xM**,**maskc**,**flag**)		
Input:	**iv**	vertex index	(i4)
	xM	invariant mass for the constraint [GeV]	(r8)
	maskc(ntrk)	track mask	(i4)
Output:	**flag**	status flag (= 0 if ok)	(i4)
Action:	Add the constraint that the tracks specified by the track mask have to have the invariant mass xM.		

Comment:	If the kth track on the vertex has to participate in the mass constraint, maskc(k) has to be set to 1, otherwise to 0.

3.3.8 VTMASS: Fit a Vertex with Mass Constraints

Name: **VTMASS** (subroutine)

Calling Sequence: VTMASS(**iv**,**par**,**cov**,**chi2m**,**flag**)

Input:

iv	vertex index	(i4)
par(3$*$ntrk+3)	fit parameter initialization (optional)	(r8)

Output:

par(3$*$ntrk+3)	fit parameters	(r8)
par(1)	x-coordinate of vertex [cm]	
par(2)	y-coordinate of vertex [cm]	
par(3)	z-coordinate of vertex [cm]	
par(4)	slope a_x of track 1	
par(5)	slope a_y of track 1	
par(6)	momentum of track 1 [GeV]	
par(7)	slope a_x of track 2	
par(8)	slope a_y of track 2	
par(9)	momentum of track 2 [GeV]	
etc.		
cov(3$*$ntrk+3,3$*$ntrk+3)	covariance matrix of fit parameters	(r8)
chi2m	χ^2 of the fit	(r8)
flag	status flag (= 0 if ok)	(i4)

Action: Fit the vertex with index iv, with all mass constraints (defined for this vertex) taken into account.

Comment: The fit may not converge in rare unfavorable cases.

3.3.9 VTCHAI: Fit a Decay Chain

Name: **VTCHAI** (subroutine)

Calling Sequence: VTCHAI(**ivp**,**ivs**,**par**,**cov**,**chi2c**,**flag**)

Input:

ivp	primary vertex index	(i4)
ivs	secondary vertex index	(i4)

Output:

par(7)	fit parameters	(r8)
par(1)	decay length [cm]	
par(2)	x-coordinate of secondary vertex [cm]	
par(3)	y-coordinate of secondary vertex [cm]	
par(4)	z-coordinate of secondary vertex [cm]	
par(5)	slope a_x of the mother particle	
par(6)	slope a_y of the mother particle	

	par(7)	momentum of mother particle [GeV]	
cov(7,7)		covariance matrix of fit parameters	(r8)
chi2c		χ^2 of the fit (2 degrees of freedom)	(r8)
flag		status flag (= 0 if ok)	(i4)

Action: Fit the chain of a primary and a secondary vertex, including all required mass constraints.

Comment: The fit may not converge in rare unfavorable cases.

3.3.10 VTSUM: Sum Momenta on a Vertex

Name: **VTSUM** (subroutine)

Calling Sequence: VTSUM(**iv**,**par**,**cov**,**flag**)

Input:

iv	vertex index	(i4)

Output:

par(5)		parameters of total momentum	(r8)
	par(1)	slope a_x	
	par(2)	slope a_y	
	par(3)	impact parameter b_x [cm]	
	par(4)	impact parameter b_y [cm]	
	par(5)	total momentum [GeV]	
cov(5,5)		covariance matrix of par	(r8)
flag		status flag (= 0 if ok)	(i4)

Action: Compute the total momentum of all particles on a vertex, taking into account all correlations between tracks.

Comment: This utility routine can be applied for any vertex which has been fit, no matter whether the fit included mass constraints or not.

3.4 vtpatt.f: The Vertex Recognition Package

Besides algorithms for the recognition of primary and secondary vertices, which are probably far from being optimized, this package contains a number of useful utility routines. The latter are of general interest and not in any way restricted to the HERA-B case.

3.4.1 VTRECB: Reconstruction of events with a decay $B^0 \to J/\psi K_S^0$

Name: **VTRECB** (subroutine)

Calling Sequence: VTRECB(**il1**,**il2**,**iKs**,**itB**,**flag**)

Input:

il1	track index of first lepton from J/ψ	(i4)
il2	track index of second lepton from J/ψ	(i4)
iKs	track index of K_S^0	(i4)

Output:	**itB**	track index of reconstructed B^0	(i4)
	flag	status flag ($= 0$ if ok)	(i4)

Action: Fit the $B^0 \to J/\psi K_S^0$ decay chain with all mass constraints, constrain the B^0 to originate on a wire, and reconstruct primary and secondary vertices. Add the B^0 at the end of the track buffer.

Comment: There is always at least one primary vertex (B^0 production vertex) and one secondary vertex (B^0 decay). The primary and secondary vertices can be found in the common blocks /ivprim/, /ivsec/, and /ivpsik/.

The reconstruction may produce error codes due to various reasons:

flag$= -1$	fit of B^0 decay vertex failed (fatal)
flag$= -2$	B^0 incompatible with all wire targets (fatal)
flag$= -3$	vertex recognition had problems (not fatal)
flag$= -4$	some fits in vertex recognition didn't converge (not fatal)

3.4.2 VTRECO: Reconstruction of Events Without a Decay $B^0 \to J/\psi K_S^0$

Name: **VTRECO** (subroutine)

Calling Sequence: VTRECO(**flag**)

Input: none

Output: **flag** status flag ($= 0$ if ok) (i4)

Action: Reconstruct primary and secondary vertices for events without a $B^0 \to J/\psi K_S^0$ decay.

Comment: If flag$\neq 0$, then some problems occured during vertex pattern recognition, probably because some fits did not converge. This is generally not fatal, since fits fail mostly for bad vertex candidates.

The primary and secondary vertices can be found in the common blocks /ivprim/, /ivsec/.

3.4.3 VTPSIK: Reconstruction of the $B^0 \to J/\psi K_S^0$ decay

Name: **VTPSIK** (subroutine)

Calling Sequence: VTPSIK(**il1**,**il2**,**iKs**,**itB**,**flag**)

Input:	**il1**	track index of first lepton from J/ψ	(i4)
	il2	track index of second lepton from J/ψ	(i4)
	iKs	track index of K_S^0	(i4)
Output:	**itB**	track index of reconstructed B^0	(i4)
	flag	status flag ($= 0$ if ok)	(i4)

Action: Reconstruct the $B^0 \rightarrow J/\psi K^0_S$ decay chain and constrain the B^0 to a target wire. Add the B^0 at the end of the track buffer.

Comment: The vertex information is stored in common /ivpsik/. There are two possible error codes, both of which are fatal:

flag$= -1$	vertex fit failed
flag$= -2$	B^0 incompatible with all wires

3.4.4 VTPRIM: Primary Vertex Recognition

Name: **VTPRIM** (subroutine)

Calling Sequence: VTPRIM(**itB**,**flag**)

Input: **itB** track index of the B^0 (i4)

Output: **flag** status flag (= 0 if ok) (i4)

Action: Reconstruct primary vertices.

Comment: If no B^0 track is present, itB has to be set to 0. If a B^0 track is present (itB>0), this track is forced to originate on a primary vertex on a wire target.

The primary vertices can be found in common /vtprim/.

Errors signalled by flag$\neq 0$ are in general not fatal.

3.4.5 VTSEC: Secondary Vertex Reconstruction

Name: **VTSEC** (subroutine)

Calling Sequence: VTRECB(**itB**,**flag**)

Input: **itB** track index of the B^0 (i4)

Output: **flag** status flag (= 0 if ok) (i4)

Action: Reconstruct secondary vertices.

Comment: VTSEC has to be called after the reconstruction of primary vertices (by whatever algorithms).

If no B^0 track is present, itB has to be set to 0. If a B^0 track is present (itB>0), then the secondary vertex search considers only tracks in the vicinity of the primary vertex of the B^0; the impact parameters of tracks with respect to the primary vertex of the B^0 have to be smaller than Bvimpc (see vtpatt.i).

The secondary vertices can be found in common /vtsec/.

Errors signalled by flag$\neq 0$ are in general not fatal.

3.4.6 VTWINI: Initialization of the Wire Target

Name:	**VTWINI** (subroutine)	
Calling Sequence:	VTWINI(**iw**)	
Input:	**iw** wire index (i4)	
Output:	none	
Action:	Initialize wire covariance matrix.	
Comment:	The routine has to be called by the user after the wire geometry is filled into common /vtwire/. If iw= 0, then all nw wires are initialized.	

3.4.7 VTWITR: Wire/Track Distances

Name:	**VTWITR** (subroutine)
Calling Sequence:	VTWITR(**it1**,**it2**)
Input:	**it1** track index of first track (i4) **it2** track index of last track (i4)
Output:	none
Action:	Compute track/wire χ^2-distances for tracks it1...it2, and fill the results into ch2wi in common /vttrwi/.
Comment:	The distance is computed in linear approximation by a single filter step. It corresponds to a χ^2 with one degree of freedom.

3.4.8 VTDVW: Vertex/Wire Distance

Name:	**VTDVW** (r8-function)
Calling Sequence:	VTDVW(**iv**,**iw**)
Input:	**iv** index of vertex (i4) **iw** index of wire (i4)
Output:	**VTDVW** χ^2-distance of vertex and wire (r8)
Action:	Compute χ^2-distance of the vertex with identifier iv to the wire with index iw.
Comment:	The distance is computed in linear approximation without iteration. It corresponds to a χ^2 with two degrees of freedom. Note that the wire has no gaussian density distribution. This distorts the χ^2 distribution.

3.4.9 VTZTR: Closest Approach of Two Tracks

Name: **VTZTR** (subroutine)

Calling Sequence: VTZTR(**it1**,**it2**,**z1**,**z2**)

Input:

it1	track index of first track	(i4)
it2	track index of second track	(i4)

Output:

z1	z-coordinate for the point on first track	(r8)
z2	z-coordinate for the point on second track	(r8)

Action: Compute the longitudinal positions z1, z2, corresponding to the two points on two tracks which have minimal distance.

Comment: The points are computed in euklidean space, without taking the track covariance matrices into account.

3.4.10 VTDV: Vertex/Point Distance

Name: **VTDV** (r8-function)

Calling Sequence: VTDV(**iv**,**xp**)

Input:

iv	vertex identifier	(i4)
xp(3)	space point [cm]	(r8)

Output: VTDV χ^2 distance of vertex iv from point xp (r8)

Action: Compute χ^2-distance of vertex iv from the space point xp(3) (three degrees of freedom).

Comment: –

3.4.11 VTDVV: Vertex/Vertex Distance

Name: **VTDVV** (r8-function)

Calling Sequence: VTDVV(**iv1**,**iv2**)

Input:

iv1	identifier of first vertex	(i4)
iv2	identifier of second vertex	(i4)

Output: **VTDVV** χ^2 distance of the vertices (r8)

Action: Compute the χ^2 distance of the vertices with identifiers iv1 and iv2 (three degrees of freedom).

Comment: The χ^2 distance is computed in linear approximation by weighted error averaging (corresponding to a Kalman filter step for vertex merging).

3.4.12 VTDVT: Vertex/Track Distance

Name:	**VTDVT**	(r8-function)		
Calling Sequence:	VTDVT(**iv**,**it**)			
Input:	**iv**	identifier of vertex	(i4)	
	it	index of track	(i4)	
Output:	**VTDVT**	χ^2 distance of vertex and track	(r8)	

Action: Compute the χ^2 distance of the vertex with identifier iv and the track with index it (two degrees of freedom).

Comment: The distance is computed by a single Kalman filter step (linear approximation).

3.4.13 VTDTT: Track/Track Distance

Name:	**VTDTT**	(r8-function)	
Calling Sequence:	VTDTT(**it1**,**it2**)		
Input:	**it1**	index of first track	(i4)
	it2	index of second track	(i4)
Output:	**VTDTT**	χ^2 distance of the tracks	(r8)

Action: Compute the χ^2 distance of the two tracks with indices it1 and it2 (one degree of freedom).

Comment: The χ^2 distance is computed in linear approximation by a single filter step.

3.4.14 VTDTP: Spatial Vertex/Track Distance

Name:	**VTDTP**	(r8-function)	
Calling Sequence:	VTDTP(**iv**,**it**)		
Input:	**iv**	identifier of vertex	(i4)
	it	index of track	(i4)
Output:	**VTDTP**	spatial distance of track and vertex [cm]	(r8)

Action: Compute the spatial distance in [cm] between the track with index it and the vertex with identifier iv.

Comment: –

References

[1] R.K. Bock et al., *Data analysis techniques for high-energy physics experiments*, Cambridge Univ. Press (1990), p.319ff.