UNIVERSITY OF COPENHAGEN

September 28, 2015

Track reconstruction
Lecture notes

Nordic detector course, November 2015

Peter Hansen

Contents

Chapter 1

Track reconstruction

The data recorded by hadron collider experiments are very complex due to the high track density, the fact that several hadron collisions take place at each bunch crossing (*pile-up*), the need to cover basically all particle momenta and directions and the unavoidable amount of passive material in the tracking volume. Therefore, highly sophisticated and error-tolerant track finders and track fitters are needed. This chapter covers the various steps in the tracking process:

- Translating the raw digits from the detector elements to *space points*.
- Associating a subset of space-points to a reconstructed track (*pattern recognition*).
- Momentum measurement.
- Track fitting.
- Vertex finding.
- Constrained fits.
- Alignment.

1.1 Space-point formation

1.1.1 The barycenter

Detector elements register *hits* from signals on pick-up electrodes induced by an *electron cloud*. If the cloud induces a signal on only one electrode, the *space-point* precision is $\Delta/\sqrt{12}$, where Δ is the electrode size. It is better if the hit signal is distributed over two electrodes as illustrated in Figure 1.1. In that case the space-point can be determined very accurately - *if* the pulse-heights, P_i, on the electrodes are measured and *if* the width of the electron cloud is well-known.

If the signal is distributed over several electrodes, the *barycenter*, $x = \sum x_i P_i / \sum P_i$, is a popular estimate of the space-point. Due to the finite size electrodes, this estimate has a bias for purely geometric reasons (the bias is largest for true impacts half way between the edge and the middle of the electrode), but this bias can be corrected for.

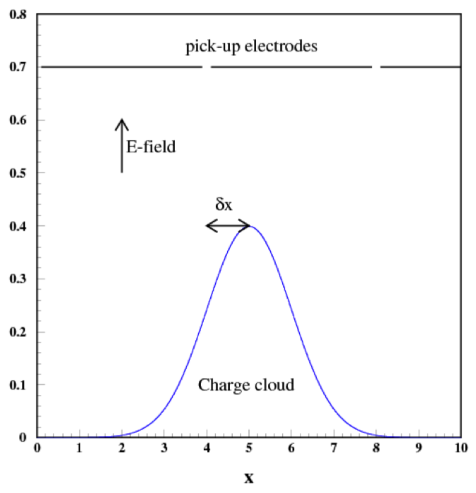


Figure 1.1: The geometry of a detector "hit"

1.1.2 Stereo angles

Another geometric problem is that often the hit information is provided by strips or wires where the coordinate along the wire (the *second coordinate*) is unknown. A solution could be to have every second layer of wires or strips oriented perpendicularly as illustrated in Figure 1.2 (left). This leads, however, to many "ghost hits" in a dense track environment. An often used

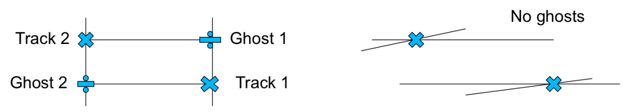


Figure 1.2: Ghost reduction using a small stereo angle

compromise is the choice of a small *stereo angle* of 20-80 mrad (Figure 1.2 right). This removes most of the ghosts while allowing some moderate precision in the determination of the second coordinate. Of course, the high precision coordinate is chosen in the bending plane so as to optimize the momentum measurement.

1.1.3 Calibrations

In order to accurately reconstruct a space-point, we must know the *response function*, the probability distribution of induced pulse-heights for a given track impact on the surface of a detector module. This response function can be measured with test beam probes and with the real data themselves. However, it may vary from channel to channel and may also vary over time for a given channel. Continuous *calibrations* of all the individual response functions must therefore be carried out during the data taking.

A few examples of response functions, or parameters of those, are:

- The relationship between the measured drift time and the track distance to the wire in a drift tube

- The correction due to the magnetic field to the barycenter of a silicon pixel cluster
- The electronic noise level (called *the pedestal*) in each channel.
- The list of dead or too noisy channels.

In order to establish the *R-t relation* in a drift tube, the average distance, R between a track hitting the tube and the central wire of the tube is plotted against the drift time, t, and an appropiate function R(t) is fitted to these data.

In order to determine the *Lorentz angle*, the angle between the electric field and the direction of drift of electrons and holes in a silicon strip or pixel detector submerged in a magnetic field, the cluster size is plotted against the incident track angle and the angle at minimum size is found.

The *pedestal* is determined by reading out the detector without beam. Beam-induced noise is determined by recording unpaired bunch crossings in colliders.

The list of *dead or noisy channels* is determined by finding holes or peaks in a histogram of hit channels filled during a run. Such channels should be ignored, but their presence still taken into account in clustering algorithms.

In the following, we will assume that a set of well-calibrated space-points have been measured in the detector, and we will concentrate on interpreting them in terms of *charged tracks*, each with a certain *momentum* and a certain *production point* or *vertex*.

1.2 Momentum measurement

The Lorentz force from a *homogeneous* magnetic field, B, on a moving charge, q, will bend the charge into *circular motion* with a radius of curvature, ρ, in the plane perpendicular to the field. The momentum component transverse to the field is given by:

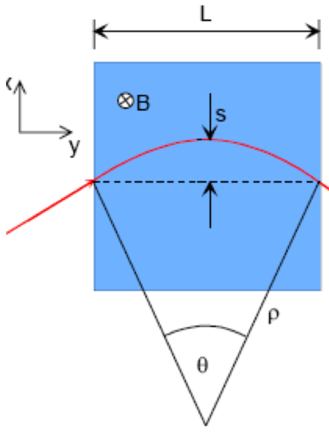
$$p_T = 0.3B\rho$$

if the momentum is measured in *GeV/c*, the magnetic field in *Tesla* and ρ in *meters*. This is the *basic equation* of a spectrometer.

In 3D, the particle trajectory will be a *helix*. If the field is confined to a stretch, L, the momentum projected on the bending plane can be found by a measurement of the bending angle, θ, or alternatively of the *sagitta*, s, of the trajectory piece of arc inside the field region. This is argued in Figure 1.3.

Let us assume that we have three measurement stations inside the magnet: One at the entrance, one in the middle and one at the exit. They measure the coordinate x with a resolution of $\sigma(x)$. Thus the estimate of the sagitta is $s = x_2 - \frac{x_1+x_3}{2}$ and the resulting relative uncertainty on the measured momentum *due to measurement resolution* is:

$$\frac{\sigma(p_T)}{p_T} = \frac{\sigma(s)}{s} = \frac{\sqrt{\frac{3}{2}}\sigma(x)8p_T}{0.3 \cdot BL^2}$$



$$p_T = qB\rho$$
$$p_T\ (\mathrm{GeV/c}) = 0.3B\rho \quad (\mathrm{T\cdot m})$$
$$\frac{L}{2\rho} = \sin\theta/2 \approx \theta/2 \quad \rightarrow \quad \theta \approx \frac{0.3L\cdot B}{p_T}$$
$$\Delta p_T = p_T \sin\theta \approx 0.3L\cdot B$$
$$s = \rho(1-\cos\theta/2) \approx \rho\frac{\theta^2}{8} \approx \frac{0.3}{8}\frac{L^2B}{p_T}$$

Figure 1.3: Principle of a magnetic spectrometer. (From Christian Jorams summer student lectures at CERN)

with p_T in GeV/c, B in Tesla and L in meters. So the most important parameter is the *lever arm*, L, and hereafter follows the position resolution, $\sigma(x)$ and the field strength, B. Note that the relative momentum uncertainty due to measurement resolution *increases linearly with momentum*. So that is actually $1/p$ that is measured with a gaussian error. Note also that *the number of space-point measurements* enters only with a square root in the momentum resolution.

Another reason for not adding too many measurement stations is that they add material in the path of the charged particle. In a spectrometer of length L, made of a material with radiation length X_0, *multiple scattering* (MS) will smear the particle direction and thus contribute an uncertainty in the transverse momentum of

$$\Delta p^{MS} = p\sin\theta_{MS} \approx p\cdot 0.0136\frac{1}{p}\sqrt{\frac{L}{X_0}}$$

with p in GeV/c. Since the p_T-kick that the particle receives from the magnetic field is $\Delta p_T = 0.3\cdot BL$, the relative momentum uncertainty due to multiple scattering is [1]:

$$\frac{\sigma^{MS}(p_T)}{p_T} = \frac{\Delta p^{MS}}{\Delta p_T} = \frac{0.0136\sqrt{\frac{L}{X_0}}}{0.3\cdot BL} = 0.045\frac{1}{B\sqrt{LX_0}}$$

Note that this contribution is independent of p.

Thus we have the situation illustrated in Figure 1.4 where the uncertainty is dominated by multiple scattering at low momentum and by measurement errors at high momentum. In the ATLAS Inner Detector (which has an average of about 1 radiation length of material), the transition happens at a transverse momentum of around 30 GeV/c.

1.3 Track parameters

The task of track reconstruction is to find the collection of hits created by the passage of each track and to estimate its *track parameters*. As an example, consider a charged particle in a

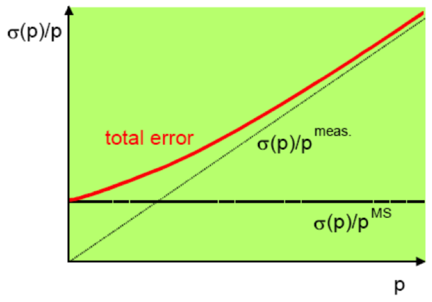


Figure 1.4: Relative momentum uncertainty. (From Christian Jorams summer student lectures at CERN)

homogeneous magnetic field. Here the trajectory that satisfies the equations of motion is a *helix*, characterized by the radius, R, the position at an arbitrary point on the helix (x_0, y_0, z_0), the tangential direction at the same point (given by the angles λ and α_0) and a sign h indicating which way the particle moves. Moving along the helix away from the arbitrary point by the path length, s, we trace out the positions:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 + R\cdot(\cos(\alpha_0 + \frac{hs}{R}\cdot\cos\lambda) - \cos\alpha_0) \\ y_0 + R\cdot(\sin(\alpha_0 + \frac{hs}{R}\cdot\cos\lambda) - \sin\alpha_0) \\ z_0 + s\cdot\sin\lambda \end{pmatrix}$$

The helix parameters could, however, be chosen differently and a particular point could be chosen to start from. A popular choice is the *perigee* parameters, where the *perigee* is the point on the helix of closest approach to the beam-axis (z-axis) as illustrated in Figure 1.5. The

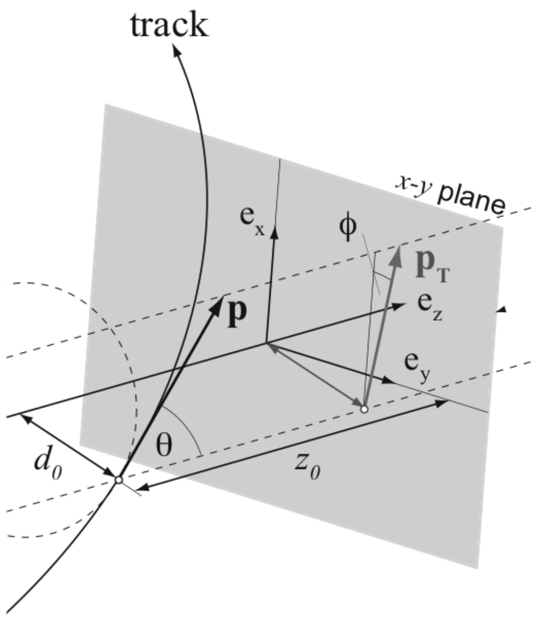


Figure 1.5: Perigee parameters. Figure by A. Salzburger.

chosen parameters are:

- The signed distance of closest approach, $d0$, where the sign gives the sense of rotation.
- The coordinate along the z-axis at this point.
- The azimuthal angle, ϕ_0, of the tangent at this point.
- The polar angle, θ, of the track to the z-axis.
- The charge over momentum, q/p, since this is in principle measured with a universal Gaussian uncertainty, as we saw in the previous section.

The vector

$$\bar{x} = (d_0, z_0, \phi_0, \theta, \frac{q}{p})$$

is called the *state vector* of the track at the perigee.

The beam axis is a particular choice. We may replace it by an axis of some detector element and quote the state vector with respect to this axis. In fact, one can say that a "track" *is* a collection of state vectors, supplemented with their covariance matrices, one for each of the detector surfaces where the track has an associated hit. In the end, however, it is the track state at its production vertex which are the parameters of interest.

1.4 Covariance matrix - quick recap

Let us recall that for any two random variables x_i and x_j, the *covariance matrix* is

$$V_{ij} = E\left((x_i - E(x_i)) \times (x_j - E(x_j))\right)$$

where $E(x)$ is the *expectation value* of x. By construction, V_{ij} is symmetric and the diagonal elements are the *variances* of the x's. The off-diagonal elements reflect the *degree of correlation* between the x's.

For the primary measurements, $\bar{m}$ in the different detector planes, we expect no correlations. But for the derived track parameters, which depend on $\bar{m}$, we do expect correlations.

Any set of functions f_i of the x's has (to first order in a Taylor expansion) the covariance matrix:

$$C^f_{ij} = \sum_{kl} \frac{\partial f_i}{\partial x_k} \frac{\partial f_j}{\partial x_l} V_{kl} = (FVF^T)_{ij} \tag{1.1}$$

This is called the *chain rule*.

1.5 The projection matrix

So what are the measurements, $\bar{m}$, expected for some track state, $\bar{x}$? This is given by a function $\bar{h}(\bar{x})$, or, assuming linearity, by the *projection matrix* $H = \delta\bar{m}/\delta\bar{x}$. As an example, the measurement could be the hit strip number in Figure 1.2 for the case of a small stereo angle α. This strip number is counted along the y'-axis which is tilted by the angle α with respect to the y-axis. So, to arrive at the y' in the "measurement frame" for a given track state (x, y), we do

$$y' = H\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -\sin\alpha & \cos\alpha \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

Let us consider another example that we shall meet again in the exercises. It is a simple spectrometer consisting of four silicon pixel planes with a dipole magnet at the center as shown in Figure 1.6. The momentum of a traversing particle is determined from the up/down angular

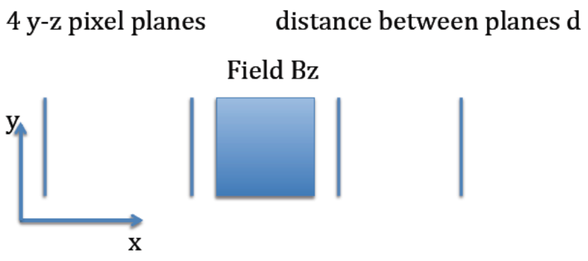


Figure 1.6: A simple spectrometer

deflection, $\Delta\phi$, caused by the field, B_z :

$$\tan\Delta\phi = 0.3\int B_z dx \frac{q}{p} = bq$$

We choose the track state at the first plane to be $\bar{x} = (z, z', y, y', q/p)^T$, where $z' = dz/dx$, and the measurement vector to be $\bar{m} = (z_1, z_2, z_3, z_4, y_1, y_2, y_3, y_4)^T$. The predicted $\bar{m}$ is then $H\bar{x}$, where

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & d & 0 & 0 & 0 \\ 1 & 2d & 0 & 0 & 0 \\ 1 & 3d & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d & 0 \\ 0 & 0 & 1 & 2d & bqd/2 \\ 0 & 0 & 1 & 3d & 3bqd/2 \end{bmatrix}$$

1.6 The Kalman filter

The idea of the *Kalman filter* [5] is to determine the track state dynamically moving from one detector surface to the next as shown in Figure 1.7. The track state at A is propagated to B using the equations of motion. The predicted measurement at surface B from the propagated track state is compared with the actual measurement. Based on this comparison, the measurement can be discarded or accepted. In the latter case, the track state at B can be updated and extrapolated further to detector surface C.

This has several advantages. Since only one measurement is considered at a time, only small covariance matrices need to be inverted (for "dividing by the error squared"), which makes the

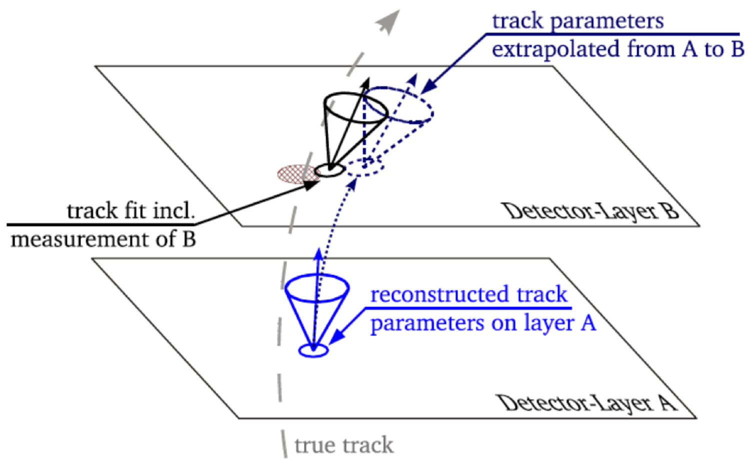


Figure 1.7: A Kalman filter step. The vector represents the track state and the cone its covariance matrix. Figure from S. Fleischmann's thesis.

algorithm *fast*. Second, *pattern recognition* (the sorting out which hits belongs to which tracks) is integrated in the track fit. Finally, the algorithm can easily take into account extra effects such as multiple scattering, energy-loss and noise, which makes the algorithm *efficient*.

Several extensions of the algorithm exist to cope with special situations. For example to deal with very densely populated track environments. Another important limitation of the simple algorithm is that the errors must be gaussian distributed around zero for the algorithm to work. This calls for special treatment of electron tracks suffering bremsstrahlung, which will be covered later.

Let us now step through the Kalman filter track finding and fitting algorithm in some detail. Clearly, we have to start with a track *seed*. Various strategies have been used to select a track seed. In experiments with precision pixel layers close to the interaction region, the seed is often a combination of two or three pixel hits, approximately pointing back to the interaction region and starting with those seeds with the highest density of extrapolations to the z-axis.

Starting with a track seed at the $k-1$'th detector plane, we then need to propagate it to the k'th plane and, again, we linearize the problem with a *propagator matrix*, F:

$$\tilde{x_k} = F_k x_{k-1}$$

where $\tilde{x_k}$ means the state extrapolated to plane k. In fact, the *propagation matrix*, F, in a detector, is exactly the same thing as the *transfer matrix* in accelerator physics.

Also the covariance matrix of the x's needs to be propagated. Here we use the chain rule, eq. 1.1, and at the same time we take the opportunity to add multiple scattering and other random fluctuations involved in the step:

$$C_k^{k-1} = F_k C_{k-1} F_k^T + Q_k$$

A very simple example is the propagation of z (unaffected by the field) over the distance d from the second to the third plane in the example spectrometer of Figure 1.6:

$$\begin{aligned}
\tilde{z}_3 &= \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_2 \\ z_2' \end{bmatrix} \\
C_3^2 &= \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix} C_2 \begin{bmatrix} 1 & 0 \\ d & 1 \end{bmatrix} + Q \\
C_2 &= \begin{bmatrix} \sigma^2 & \sigma^2/d \\ \sigma^2/d & 2\sigma^2/d^2 \end{bmatrix} \\
Q &= \begin{bmatrix} \theta_{MS}^2 d^2 & \theta_{MS}^2 d \\ \theta_{MS}^2 d & \theta_{MS}^2 \end{bmatrix}
\end{aligned}$$

where σ is the coordinate measurement resolution, and θ_{MS} is the *rms* multiple scattering angle in a single detector plane.

In general, propagation through an inhomogeneous field is needed. The preferred method is a numerical technique called *Runge-Kutta integration*. When propagating the trajectory from surface A to surface B, the most naive approach would be to move along the tangent sampled at surface A. In the Runge-Kutta step, the derivatives (directions) are instead sampled at a number of intermediate positions and a weighted average is taken. Consider a coordinate y with value $y_n = y(t_n)$ and with a time derivative $y' = f(t, y)$ given by the equations of motions. We now take a short step h in time the following way:

$$\begin{aligned}
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2} k_1) \\
k_3 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2} k_2) \\
k_4 &= f(t_n + h, y_n + h k_3) \\
y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}$$

This holds true up to a h^5 term. You can check it, for example, on the very nonlinear case $y = e^t$. In the spectrometer application, however, y'' is given by the Lorentz force from a parametrized B-field and we thus need two Runge-Kutta integrations to arrive at F. All of this can be parametrized before data taking.

After propagating the trajectory to a new surface, k, the track state is $\tilde{x}_k$. The predicted measurement, $H_k \tilde{x}_k$, is now compared with the nearest actual measurement, m_k, yielding a *residual*, r, together with its covariance matrix, R:

$$r_k^{k-1} = m_k - H_k \tilde{x}_k \qquad (1.2)$$

$$R_k^{k-1} = V_k + H_k C_k^{k-1} H_k^T \qquad (1.3)$$

where V_k is the covariance matrix of the measurement, which may have more than one component but is normally diagonal. At this point, the measurement m_k may be rejected if the *chi-squared*, $\chi_k^2 = r_k^2 / R_{kk}$, exceeds at certain value.

If m_k is an acceptable hit, it is used to *update the track state*. The updated state is the weighted average of the propagated state and the state suggested by the new hit:

$$\begin{aligned} x_k &= C_k\left((C_k^{k-1})^{-1}\tilde{x_k} + H_k^T V_k^{-1} H_k m_k\right) \\ C_k^{-1} &= (C_k^{k-1})^{-1} + H_k^T V_k^{-1} H_k \end{aligned}$$

The updated state is shown as a solid black cone at plane B in Figure 1.7.

Once the last hit is added to the track, a procedure called *smoothing* starts. Here, the Kalman filter is repeated *backwards*, updating again the track states at each surface, but this time using information from all the n hits. The step from surface $k+1$ to k can be written:

$$\begin{aligned} C_{k|n}^{-1} &= C_{k|k}^{-1} + C_{k|k+1}^{-1} \\ x_{k|n} &= C_{k|n}\left(C_{k|k}^{-1}x_{k|k} + C_{k|k+1}^{-1}x_{k|k+1}\right) \end{aligned}$$

where $x_{k|k}$ is the current state, $x_{k,k+1}$ is the state propagated from surface $k+1$ and $x_{k|n}$ is the updated, *smoothed* state. It is at this point possible to exclude *outliers* from the fit, hits contributing a large χ^2, while still having them associated with the track.

If the track at the end of this stage fulfills certain quality criteria (based on the number of hits used in the track fit, the χ^2 and the fitted parameters), a new entry is added to the list of reconstructed tracks. The entry contains the track state extrapolated to the perigee and also contains each hit with its associated track state at the hit surface. After all seeds have been tested, the yet unused hits may then be available for a second pass trying to reconstruct more tracks with more relaxed quality criteria, such as a lower p_T cut. Also, while the first pass is usually done *inside-out*, starting from the innermost planes, a subsequent *outside-in* pass is often added, starting from a seed in the outer layers. This is especially useful to pick up tracks from secondary vertices such as K^0 decays or γ-conversions.

This procedure is the preferred one in most applications.

1.7 The Newton-Raphson fit

If the list of hits associated with a track is known in advance, an alternative, but mathematically equivalent, track fitting algorithm is the well-known *Newton-Raphson* or *global least-squares* fit. The procedure is to minimize $\chi^2 = \sum(m_i - h_i(\bar{x}))^2/\sigma(m_i)^2$, varying the track state $\bar{x}$, where $h_i(\bar{x})$ is the physics model predicting the ith measurement.

In the linear approximation, $h_i(\bar{x}) \approx H\bar{x}$, we can write

$$\chi^2 = (\bar{m} - H\bar{x})^T V^{-1} (\bar{m} - H\bar{x})$$

which is readily minimized by the track state

$$\bar{x} = (H^T V^{-1} H)^{-1} H^T V^{-1} \bar{m} \tag{1.4}$$

having the covariance matrix (using the chain rule):

$$Cov(x) = (H^T V^{-1} H)^{-1} = \frac{1}{2}\left(\frac{\partial^2 \chi^2}{\partial \bar{x}^2}\right)^{-1} \tag{1.5}$$

If the projection $h(x)$ is *not* linear, it can be forced linear by Taylor expansion around an initial value x_0 and the derivatives of χ^2 can be found:

$$\begin{aligned} \frac{d\chi^2}{dx}(x_0) &= -2H^T V^{-1}(m - h(x_0)) \\ \frac{d^2\chi^2}{dx^2}(x_0) &= 2H^T V^{-1} H \end{aligned}$$

In the next iteration, if $\frac{d\chi^2}{dx}(x_0)$ is not already zero, x_0 is replaced by the better value:

$$x_1 = x_0 - (\frac{d^2\chi^2}{dx^2}(x_0))^{-1}\frac{d\chi^2}{dx}(x_0)$$

and this continues until convergence of the x's.

In the global least-squares fit, a multiple scattering angle can be introduced as an extra track parameter at each scattering surface, giving an extra contribution of $(\theta_{MS}/\theta_0)^2$ to χ^2. You can avoid these extra parameters at the cost of introducing correlations, non-diagonal elements, in the measurement covariance matrix by making the substitution:

$$V \to V + S^T \Theta_0 S$$

where $S = \delta\bar{r}/\delta\bar{\theta}$ (for a straight track S_{ij} would be equal to the distance between planes i and j) and Θ_0 is θ_0^2 times the diagonal unit matrix.

1.8 Vertex finding

1.8.1 Kalman filter

During LHC runs there will in the future be up to 200 beam particle collisions happening during a single bunch crossing. The location in space of each collision is called a *vertex*. The collision that triggered the recording of the bunch crossing event (normally the most violent collision) is called the *primary vertex* and the rest are *pile-up vertices*. We would like to reconstruct all of them.

The reconstruction of the vertices can be accomplished by a Kalman filter as illustrated in Figure 1.8. The interaction region, also called the *beam spot*, is known in advance from the distribution of vertices in previous bunch crossings and from beam diagnostic equipment. Already a *seed* of two tracks crossing each other within the beam spot locates the vertex much better than the beam spot (the vertex labeled $n = 2$ in the Figure). The algorithm then proceeds exactly as for track finding, adding one track at a time, if more tracks are found consistent with this vertex, and updating the vertex to higher and higher precision. The vertex labeled $n = 3$ shows both an accepted an a rejected third track.

Obviously, the first choice of two seed tracks is a crucial step. Typically one first looks at the spot with the highest density of tracks crossing the beam line and from here selects two seed tracks (for example the two with the closest perigees). More elaborate extensions to this algorithm exist that use adaptive and iterative techniques aiming to minimize the global chi-squared for *all* possible associations of N tracks to an arbitrary number of vertices [5].

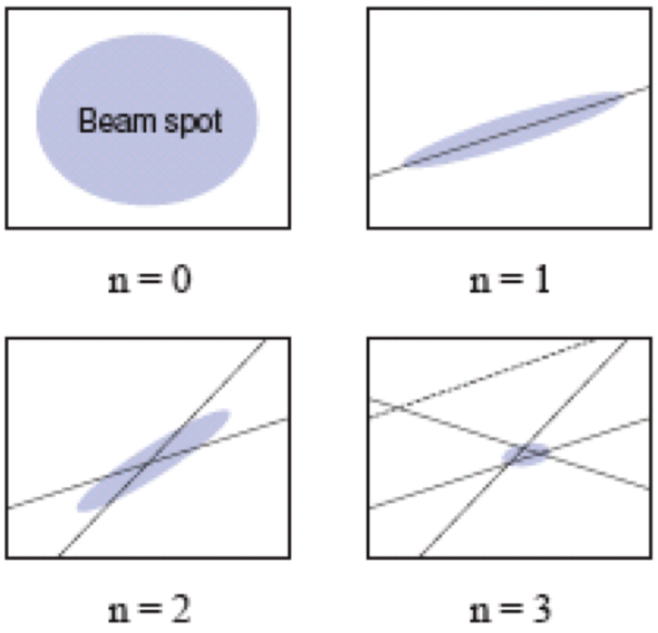


Figure 1.8: Reconstruction of a vertex

1.8.2 Billoir fit

Just as for track fitting, the Newton-Raphson method (called the Billoir fit in the vertex fitting application) is an alternative, if the tracks associated with a given vertex are known in advance. This could be performed after the Kalman Filter and is equivalent to the Kalman Smoother step. Not only the vertex position is refitted here, but also the track momenta, this time using the constraint that they must all origin from a common vertex point.

Let $\bar{v}$ be the vertex position, $\bar{p}_i$ the 3-momentum of the i'th associated track at the vertex and $\bar{x}_i$ be the state vector of the i'th track at some reference surface. Evidently, this track state can be predicted from the vertex and the momentum, so we have $\bar{x}_i = F(\bar{v},\bar{p}_i)$. Let now $\bar{v_0}$ and $\bar{p_{0i}}$ be the first rough estimates of the vertex and the i'th momentum. A Taylor expansion of the predicted track state around these parameters gives

$$F = F(\bar{v_0},\bar{p_{0i}}) + D\delta\bar{v} + E_i\delta\bar{p}_i$$

Let then $\delta\bar{x}_i{}^{meas} = \bar{x}_i{}^{meas} - F(\bar{v_0},\bar{p_{0i}})$ be the *residual* and V_i be its covariance matrix. We can then write down the chi-squared:

$$\chi^2 = \sum(\delta\bar{x}_i{}^{meas} - D\delta\bar{v} - E_i\delta\bar{p}_i)^T V_i^{-1}(\delta\bar{x}_i{}^{meas} - D\delta\bar{v} - E_i\delta\bar{p}_i)$$

Since we have now linearized the track model, we can analytically find the parameters minimizing χ^2 by solving the linear equations $d\chi^2/d(\delta\bar{v}) = 0$ and $d\chi^2/d(\delta\bar{p}_i) = 0$ [7]:

$$\begin{aligned}\bar{v} &= \bar{v_0} + \left(A - \sum B_i C_i^{-1} B_i^T\right)^{-1}\left(\bar{t} - \sum(B_i C_i^{-1})^T\bar{u}_i\right)\\ \bar{p}_i &= \bar{p_{0i}} + C_i^{-1}(\bar{u}_i - B_i^T\delta\bar{v})\end{aligned}$$

where

$$\begin{aligned} A &= \sum D^T V_i^{-1} D \\ B &= D^T V_i^{-1} E_i \\ C_i &= E_i^T V_i^{-1} E_i \\ \bar{t} &= \sum D^T V_i^{-1} \delta \bar{x}_i^{meas} \\ \bar{u}_i &= E_i^T V_i^{-1} \delta \bar{x}_i^{meas} \\ \delta \bar{v} &= \bar{v} - \bar{v_0} \end{aligned}$$

Now we can replace $\bar{v_0}$ with $\bar{v}$ and $\bar{p_{0i}}$ with $\bar{p}_i$ and repeat the exercise until convergence. The 3×3 covariance matrices of the fitted parameters are given by:

$$\begin{aligned} Cov(\bar{v}) &= \left(A - \sum B_i C_i^{-1} B_i^T\right) \\ Cov(\bar{p}_i) &= C_i^{-1} + \left(B_i C_i^{-1}\right)^T Cov(\bar{v}) B_i C_i^{-1} \\ Cov(\bar{v}, \bar{p}_i) &= -Cov(\bar{v}) D E_i^{-1} \\ Cov(\bar{p}_i, \bar{p}_j) &= \delta_{ij} E_j^{-1} - E_i^{-1} D^T Cov(\bar{v}, \bar{p}_i) \end{aligned}$$

While the machinery above is reasonably straight forward to code, the hard work for the "fit-designer" is to find the track model, or projection matrix, F, and a good initial guess for $\bar{v_0}$. These depend on the detailed geometry and magnetic field map of the setup.

1.8.3 Constrained fits

How can *a priori* knowledge be exploited in the fits to increase precision? For example, the beam spot could be known in advance to be $\bar{b}$ with some variance V_b. In that case one could add an extra piece to the χ^2:

$$\delta\chi^2 = (\bar{v} - \bar{b})^T V_b^{-1} (\bar{v} - \bar{b})$$

which would modify the χ^2 derivatives with respect to v.

In another example, a neutral particle created at the primary vertex propagates to a *secondary vertex* where it decays into, say, the 5 tracks in Figure 1.9 at "SV". We want to fit the secondary

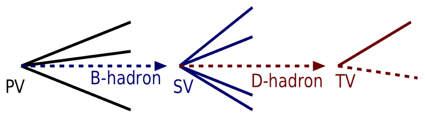


Figure 1.9: Event with a *secondary* and a *tertiary* vertex

vertex and at the same time exploit that the sum of the momenta should point back to the primary vertex. Note that in this case we know the constraint to be *exactly true*. In this case, one could use *Lagrange multipliers* and add a piece:

$$\delta\chi^2 = -\bar{\lambda} \cdot \bar{d}(v_{\bar{PV}}, v_{\bar{SV}}, \sum \bar{p}_i)$$

where $\bar{\lambda}$ are three new arbitrary fit parameters and $\bar{d}$ is the directed distance between the primary vertex and the extrapolation from the secondary vertex using minus the summed momentum at the secondary vertex. This $\bar{d}$ should be zero if all particles are well measured.

In general, the Lagrange multiplier method expresses the constraints in terms of some functions of the fit parameters that are, in principle, exactly zero, $H(x) = 0$. As a third example, consider a photon conversion in the detector. In addition to the constraint that the e^+e^- pair should point back to the primary vertex, there is also the constraint that they should emerge parallel from a common secondary vertex (since the photon has zero mass). This can be expressed as

$$\bar{H}(\bar{p_1},\bar{p_2},\bar{r_1},\bar{r_2}) = (\frac{\bar{p_1}}{E_1} - \frac{\bar{p_2}}{E_2}, \bar{r_1} - \bar{r_2}) = 0$$

where the p's and r's of the two tracks refer to their starting point. Again we make a good guess about the momenta and the starting point, $\bar{x_A}$, and Taylor expand H around that:

$$\bar{H}(\bar{x}) \approx D\Delta\bar{x} + \bar{H}(\bar{x_A}) = 0$$

In this case, with $\bar{x} = (\frac{\bar{p_1}}{E_1}, ...)^T$, the derivative, D, would be just a diagonal matrix with ± 1's. Then a piece is added to the χ^2:

$$\chi^2 = (\bar{x} - \bar{x_0})^T V_0^{-1} (\bar{x} - \bar{x_0}) + 2\bar{\lambda}^T (D(\bar{x} - \bar{x_A}) + H(\bar{x_A})) \tag{1.6}$$

Here "0" refers to the unconstrained solution, in this case the two independently fitted track parameters with their block-diagonal covariance matrix. The requirement that χ^2 should be at minimum is again a set of linear equations and thus exactly solvable for the track parameters x and the *Lagrange multipliers* λ. The multipliers are not used for anything other than a calculational tool. If the constraints were not linear functions from the start, iterations with x_A replaced by x are necessary. At each iteration, the explicit solutions to the χ^2 minimization are

$$\begin{aligned}
\lambda &= V_D(D(x_0 - x_A) + H(x_A)) \\
x &= x_0 - V_0 D^T \lambda \\
V_D &= (D V_0 D^T)^{-1} \\
V_x &= V_0 - V_0 D^T V_D V_0
\end{aligned}$$

1.9 Global optimization

Track finding and fitting has up to now been done by considering one track candidate at a time. In very dense track environments, however, we are faced with the problem where several track candidates compete for the same hit. What is the *globally* optimal assignment of hits to tracks (or maybe to noise)? The nearest track candidate is not necessarily always the optimal choice. This is a variant of the *travelling salesman's problem.* Several approaches have been established to solve this problem, see e.g. [6] for a review. They typically minimize some global "energy function" using methods borrowed from statistical physics. For the traveling salesman the energy function results in two competing forces, one attracting the route to the cities on his list and one trying to minimize the length of travel. In track reconstruction we want as many hits as possible with as small residuals as possible associated with the tracks.

1.9.1 Combinatorial Kalman Filter

In the combinatorial Kalman filter [5] a track seed is as usual propagated to the first layer. If this contains several hits compatible with the track, several branches of the filter is created and propagated in parallel to the next layer where they each may multiply again. A branch is allowed to have no hit in a detector layer, to have a *hole*, but if that happen consecutive times (and the planes are not known to be dead), the branch is dropped. In the end, only the branch with the highest quality, in terms of total number of hits, number of holes and total χ^2, is included in the final list of tracks.

1.9.2 Gaussian Sum Filter

The *Gaussian Sum Filter* is designed to cope with material effects that sometimes can create large changes in the track state. The prime example is *bremsstrahlung* of electrons that occasionally causes an energy loss by a fraction $1-z$. The probability distribution is given by the distinctly non-gaussian Bethe-Heitler law [2]:

$$f(z) = (-lnz)^{c-1}/\Gamma(c)$$

where $c = X/ln2$ and X is the thickness in radiation lengths of the current detector layer. In the Gaussian Sum Filter this distribution is modelled by a sum of n Gaussians, each carrying a certain weight determined beforehand by simulation. At each detector layer, the Kalman filter branches out in n branches that are each propagated to the next layer. Here they are updated by the hits, whereby also an assignment weight is multiplied on the total weight of the branch, according to the distance from the hit to the branch. Then the branches multiply again and eventually a “component reduction” is neccesary to limit CPU consumption, removing the branches with the smallest total weights. In the end, the surviving branches are collapsed into one weighted mean. This procedure has been proven to significantly improve electron reconstruction in the ATLAS and CMS experiments, but it is also very computational demanding.

1.9.3 Elastic Arms

A truly global approach is called the *elastic arms* algorithm that works with a fixed number of “deformable track templates”. The energy function would be of the form:

$$E = \sum_{a}^{Tracks} \sum_{i}^{Hits} S_{ia} M_{ia} + \lambda \sum_{i} (\sum_{a} S_{ia} - 1)^2$$

where M_{ia} is typically the χ^2 of the hit and S_{ia} is an *assignment strength*, to be understood as the *probability* for the hit to belong to track a. The second term imposes a penalty for the case where the hit is not assigned to any track. We can get rid of that term if the a’s also include a “noise template” to which a hit can be associated if it is too far from any track.

If S is chosen to be either zero or one (either the hit belongs to the track or not), the energy landscape can become very spiky with lots of local minima. This may be amended by a *fuzzy assignment probability*:

$$S_{ia} = \frac{e^{-\beta M_{ia}}}{e^{-\beta\lambda} + \sum_{a=1}^{Tracks} e^{-\beta M_{ia}}}$$

where track number 0 is the "noise template". The parameter $\lambda = M_{i0}$ acts as a χ^2 cut-off so that for $M_{ia} > \lambda$, the assignment strength to the "noise template", S_{i0}, dominates.

The β is analogous to the inverse temperature appearing in the Boltzmann distribution. A process called "annealing" starts at a high temperature (small β) where S_{ia} is relative large, even for distant hits. This irons out the local minima. The energy function is then iterated to its global minimum using its derivatives with respect to the track parameters, similar to the Newton-Raphson fit. Note, that this requires a parametric track model. In a next iteration, the temperature is lowered by 5-10% and the procedure repeated with a new initial track states for the fixed number of tracks, and these iterations continue until the assignment strengths all take approximately the discrete values 0 or 1. This algorithm has proven useful, for example in the HERA-B experiment, which was a fixed target experiment resulting in a very dense track environment in the forward region [5].

1.9.4 Deterministic Annealing Filter

A weakness of the elastic arms method is that it needs a parametric track model and does not cope with effects of the detector material. It also needs a fixed number of tracks. The Kalman filter has a convenient way of incorporating detector effects, including inhomogeneous magnetic fields, and to dynamically determine the number of tracks. A way to recover the advantages of the Kalman filter in the elastic arms is provided by the *Deterministic Annealing Filter* [6]. Here, the Kalman filter is used with an assignment strength for each hit in a given detector plane, instead of just choosing the closest hit. The assignment strength is simply the Gaussian likelihood for the current extrapolated track candidate to produce the hit, but using an error on the hit measurement inflated by a temperature factor T. In order to allow for the possibility that no hit was produced at this detector surface, the assignment strength for hit m_i in plane k may be normalised as follows:

$$S_i^k = \frac{\phi_k^i}{\sum_j^{n_j}(\Lambda_k^j + \phi_k^j)}$$

where, as explained above,

$$\phi_k^i = Gauss(m_i, H_k x_k, TV_i + H_k C_k H_k^T)$$

and Λ is proportional to $e^{-\frac{\lambda}{2T}}$ so that λ again serves as a χ^2 cut-off, exiling too far away hits to the noise hypothesis.

This means that when the track state is updated at plane k, it is not done by adding one particular hit, but a weighted average of all the hits in the neighbourhood of the track in that plane. Again the procedure is repeated with a lower temperature for the tracks that have survived the quality criteria, now replacing their seeds, and this continues until again the assignment strengths have stabilized around 0 or 1 (it may, however, always be discussed whether or not to allow hits being shared between two tracks).

The Deterministic Annealing Filter (DAF) has proven especially good at determining the right hit association for a given track, for example in the case of drift tubes where the measurement itself does not tell whether the track passed the anode wire left or right. Therefore the possiblity has been studied of using the DAF for track fitting, instead of the conventional Kalman

Smoother, while keeping the Kalman Filter, possibly the Combinatorial Kalman Filter, for track finding. These studies have been carried out for different setups and different track density and noise environments, and they consistently show improvements in tracking efficiency and precision when using the DAF [6]. The DAF has also been extended to an efficient algorithm for finding and fitting multiple pile-up vertices [6].

1.10 Alignment

In order to achieve unbiased, high-resolution tracking, *alignment* of the detector elements is of great importance. The challenge of controlling the positions to the micron level of hundreds of millions of channels over a volume of thousands of cubic meters must be addressed in the experiment design at all stages. First, it is of utmost importance to maintain very tight mechanical tolerances in the manufacturing of detector elements and their support structures and to keep the temperature constant throughout operations. Second, all the detector elements are surveyed optically, both before and after assembly. Often laser based systems are used to trace changes in relative positions after assembly. The ultimate alignment precision, however, is obtained from the fitted track data themselves and this is the main subject of this section which closely follows Reference [8] where many more details can be found.

Consider our example spectrometer from Figure 1.6. The measured vertical coordinates y_i are predicted by the track model $y = h(\bar{x}, \bar{\alpha})$, where $\bar{x}$ are the track parameters and α are the alignment corrections to the position and orientation of each detector plane. Let us just consider the y-position of the planes. The most straight forward estimate of $\bar{\alpha}$ is then a vector of vertical displacements given by the *average residuals*, $< r_i = y_i - h(x, \alpha) >$ over a large sample of reconstructed tracks.

This approach is called *local alignment*, because each alignment correction is regarded as independent of the other corrections. Therefore, this method in general requires many iterations. Actually, there is no guarantee of convergence, because of the correlations introduced by the track fit between the planes, but the method might still work well.

In the *global* alignment approach, a *total* chi-squared is defined over a large sample of reconstructed tracks:

$$\begin{aligned} \chi^2 &= \sum_{tracks} rR^{-1}r^T \\ r(x,\alpha,m) &= m - h(x,\alpha) \end{aligned}$$

where r are the residuals, m the measurements, and α the alignment parameters. Assume, for example, that we have many strip modules each forming a piece of plane in space. Then the α parameters would be corrections to the coordinates of the midpoint of each module plus corrections to its three Euler angles. The ambitious idea is to minimize simultaneously the total χ^2 with respect to all the track states, $\bar{x}$, in the large sample *and* with respect to the 6 alignment corrections, $\bar{\alpha}$, for each little module in the detector.

If the χ^2 is at minimum wrt. $\bar{x}$ for the current alignment parameters, we have for the *total* *derivative* of χ^2 wrt. $\bar{\alpha}$:

$$\frac{d\chi^2}{d\bar{\alpha}} = 2\sum_{tracks}(\frac{\partial\bar{r}}{\partial\bar{\alpha}})R^{-1}\bar{r}^T$$

The desired condition, $d\chi^2/d\bar{\alpha} = 0$, leads to M equations for the M alignment parameters, $\bar{\alpha}$. These equations are in general nonlinear in $\bar{\alpha}$. Therefore we again force them linear by Taylor expanding χ^2 around the current estimate of α and express the desired condition as:

$$-\frac{d\chi^2}{d\bar{\alpha}}(\alpha_0) \approx \frac{d^2\chi^2}{d\bar{\alpha}^2}(\alpha_0)\Delta\bar{\alpha} \tag{1.7}$$

The idea is now to solve these equations, then to correct α_0 by $\Delta\alpha$ and then iterate until convergence. If the residuals are linear in α, there is no reason to iterate, since the solution is exact. The solution to the equations at each iteration step is:

$$\Delta\alpha_i = -\left[\sum_{tracks}\frac{\partial\bar{r}}{\partial\alpha_i}R^{-1}\left(\frac{\partial\bar{r}}{\partial\alpha_j}\right)^T\right]^{-1}\sum_{tracks}\left[\frac{\partial\bar{r}}{\partial\alpha_j}R^{-1}\bar{r}^T\right] \tag{1.8}$$

where summation over j is implicit and $R = V - HCH^T$ is the covariance matrix of the residuals, r (here there is a minus instead of the plus seen in Eq. 1.3, because the particular measurement has *participated* in the track fit).

Let us define

$$\bar{b} = -2\frac{d\chi^2}{d\bar{\alpha}} \quad A = 2\frac{d^2\chi^2}{d\bar{\alpha}^2}$$

The inverse of A is the covariance matrix, C, of the fitted alignment parameters according to Eq. 1.5. If indeed A has an inverse, we can diagonalize A and find the M eigenvectors $u^{(j)}$ with eigenvalues d_j. The inverse has the same eigenvectors with eigenvalues $1/d_j$:

$$C_{kl}^{(\alpha)}(\Delta\bar{\alpha}) = \sum_{j=1}^{M}\frac{1}{d_j}u_k^{(j)}u_l^{(j)} \tag{1.9}$$

In the case at hand, the fitted parameters are corrections to the alignment parameters and the eigenvectors describe orthogonal collective distortions of the modules making up the tracking detector. If the eigenvalues are sorted after decreasing absolute value, the eigenvector decomposition has the name *principal component analysis*.

The alignment corrections minimizing χ^2 can be expanded in eigenvectors:

$$\Delta\bar{\alpha} = \sum_j \frac{u^{(j)T}b}{d_j}u^{(j)}$$

Clearly, Eq. 1.9 predicts exploding uncertainties when some of the eigenvalues get close to zero. Such eigenvalues belong to collective distortions that more or less preserve the helical shape of the tracks and go under the name *weak modes*. The χ^2 track fit has no chance of spotting those distortions because the derivative of χ^2 with respect to the weak modes becomes zero. Therefore they creep under the radar. The most trivial example is a translation of the entire detector. That can of course not be seen in the χ^2 of the track fits. But there are many other collective distortions (twists, compressions etc of the detector) that are invisible to the χ^2.

The only way to identify misalignment contributions from *weak modes* is to use *external constraints*. Let us express the constraints as functions $\bar{g}(\bar{\alpha})$ that ought to be zero within the precision given by the covariance G. This will contribute an extra term in the χ^2:

$$\chi_g^2 = g^T G^{-1} g$$

which will change the first and second derivative in Eq. 1.7.

The trivial example of an overall translation or rotation of the entire detector could be resolved by an external optical survey, the result of which is entered in the χ^2 as in the equation above. This particular mode could also be handled with a so-called *exact constraint*. We may simply *define* the average position of all modules to be the origo of the coordinate system. This exact constraint could be implemented by parameter substitution or by the Lagrange multiplier method in Eq. 1.6.

More sophisticated weak modes can be handled by exploiting other external knowledge. In collider detectors you can have rotations of the individual detector layers (or of the field) that would leave the χ^2 invariant, but change the fitted momentum. Such weak modes could be targeted by studying the E/p ratio of the calorimeter energy to the fitted momentum of identified high-energy electrons. Other weak mode distortions can be cornered by studying the deviations from the known masses of reconstructed Z bosons, J/Ψ mesons and K_S^0 mesons. Any such study must be binned in ϕ and θ in order to identify a particular weak mode. Cosmic ray runs are especially useful because cosmic rays penetrate the entire detector, while tracks from collision events origin from the middle of the detector and thus only probe half of it. In general, the more detector parts that participate in the track fit and the global alignment, the better.

Finally, information from laser based systems and piezo-electro feelers may be used to trace fast changes in module positions, changes that are too fast to be followed by alignment based on a large sample of reconstructed tracks. These corrections are applied directly on a run-by-run basis with the constraint that they average to zero over the event sample used for track-based alignment.

1.11 Exercises

1. **Design a spectrometer**

   - Copy all the files in

     `http://www.nbi.dk/~phansen/nordforsk/`

     This is C++ code using ROOT libraries that simulates a compact spectrometer with very thin (0.05mm) CMOS pixel layers (the MIMOSA chip) and a weak spectrometer magnet in the middle. You are assumed to be running some unix on your computer. Figure 1.6 illustrates the setup, except that we here use three planes on each side of the magnet. The goal is to measure positron production in the 50-1000 MeV range from electrons impinging on a diamond target upstream of the first layer. Such a spectrometer is used by an Aarhus group in connection with studies of a positron factory for the future CLIC accelerator.

The program simulates mono-energetic single positrons emitted along the beam (x) axis, at a small cost of generality. The simulated positrons are traced and digitized through the spectrometer, including effects from multiple scattering, noise and detection inefficiencies.

In this excise you implement a Kalman Filter, in a way suggested below, for the pattern recognition of the simulated data, resulting in a number of track candidates After that, a global chi-squared fit finds the best among the candidates. The pattern recognition requires hits in all layers, except for the last two where only one is required.

- Implement the method Spectrometer::KalmanFilter in Spectrometer.C using the matrix formalism of these notes with help of the ROOT TMatrixD class. The method must propagate a track candidate from the last detector layer with a hit to layer $p1$ in the non-bending x-z plane. It must then update the track parameters in x-z using the hit *ihit* and return the chisquared contributed by the hit as well as the updated RecoTrack, tout. The first lines could be:

```
float Spectrometer::KalmanFilter
(int p1, int ihit, RecoTrack t, RecoTrack& tout) {

  if(debug) cout << `` in Kalman Filter `` << endl;
  double s2   = resolution*resolution;
  double pinv = 1./beamMomentum;        //use here a fixed momentum
  double t0   = multScattAngle*pinv;    //average multiple scattering angle
  TMatrixD z  = t.GetPar();             //track parameters at current plane
  TMatrixD C  = t.GetCov();             //their covariance matrix
  //we are not sure which plane is the current, so check
  vector<int> hits=t.GetHits();
  int plane0  = hitdata.at(hits.back()).GetPlane(); //so this is it
  double d=distBetweenPlanes*(p1-plane0);  //propagation distance
  if(debug) cout << ``  propagate from plane `` << plane0
                << `` to plane `` << p1 << endl;

  Then make the propagator matrix (TMatrixD) F and it's transposed.
  Then make the multiple scattering smearing matrix.
  Then make the total covariance matrix of the propagated track.
  Then propagate the track parameters, z, to plane p1.
  Then make the covariance matrix of the updated track.
  Then make the parameters of the updated track.
  Then, if chi-squared<Cut1, construct the updated RecoTrack, tout.
  Finally return the chi-squared.

  Warning: If you transpose or invert a TMatrixD into a new matrix
  (by doing HT=H.T() or Cinv=C.Invert()) then H or C is also left in the
```

```
transformed state, so you need to transform them back ( just do H.T()
or C.Invert() again, or else copy the matrices before transforming).

When you have edited Spectrometer.C to your liking, then type make.
Then type ./SMgr.exe | tee spectrometer.out
Look then at the output, type root and in ROOT type
TBrowser b and click at the histogram file and the histograms
you want to see, fit, zoom or whatever.
```

- Is the spectrometer configuration optimal? Try vary the length by varying distBetweenPlanes. Can we do with fewer planes? Try 4 instead of 6. What happens if the magnet is twice as strong? What should be chosen for the adjustable noise occupancy? What is a reasonable chi-squared cut? Use the 1/p accuracy, the track reconstruction efficiency and the computing time as deciding quality factors. Try out both 0.05 GeV and 1 GeV in each case.
- The simulated efficiency is in fact a pure guess. If the simulated data had been real, how would you measure the plane efficiency (the probability for a track to produce a hit in a plane). How would you then estimate the track reconstruction efficency?
- Since we deal with positrons, we need to worry about bremsstrahlung, but this is not taken into account in the program. How much trouble do you expect from this? (Hint: make a hit-and-miss simulation of the fractional energy loss in each of the three planes before the magnet and in the line calculating the angular bend in the magnet replace beamMomentum with a reduced momentum).

2. **Reconstruct a vertex**. Let us consider the case where our spectrometer is traversed by two pions from a K0 decay somewhere upstream. Imagine that our program is now made able to return *two* fitted tracks, each with parameters (z0,z',y0,y',q/p) and their covariance matrix.

   - Write down an initial estimate of the decay vertex v0 and the derivative matrices D and E from the Billoir vertex fit.
   - How could you take advantage of external information? For example that the beam, including the K0's, propagates along the x-axis with a gaussian transverse profile of size *b*, or that you are sure that the two pions really come from a K0 decay.

3. **Make an alignment algorithm**. Among the downloaded files are AlignSpectrometer.C and AMgr.C. Edit the Makefile and, if needed, the AlignSpectrometor constructor. Then type make and then

   ```
   ./AMgr.exe | tee alignspectrometer.out
   ```

   Here, each plane is shifted randomly in each direction by a sigma of 0.1mm. Both the local and global method is implemented and can be chosen in the constructor. Look at the output (both the "out" file and the root file).

- What accuracy is obtained and does that make sense? Does it help to make iterations? Is global better than local? Are there “ weak modes “ giving trouble? Anything you could do about those? Then assume that the true beam axis in the $x-z$ plane has been measured with beam monitoring devices to be $z_0 = 0.002 \pm 0.001 cm$ at the first plane and have a slope $dz/dx = 0.1 \pm 0.01 mrad$. Try to implement that knowledge in the simulation and take advantage of it in the alignment fit.
- The program has only considered unwanted translations of the planes. How would you formally implement the possibility of a rotation, for example around the local z-axis of the planes?

Bibliography

[1] J. Beringer et al., (Particle Data Group), “Review of Particle Properties”, Phys.Rev. **D86**, 010001 (2012), on line: pdg.lbl.gov.

[2] H. Bethe and J. Ashkin, “Passage of radiation through matter”, in E. Segre (.ed), “Experimental Nuclear Physics”, Vol 1,Part 2, Wiley, (1959)

[3] M.Cacciari et al.,”The anti-kt clustering algorithm”, JHEP 0804:063 (2008) arXiv.0802.1189

[4] S. Tavernier,” Experimental Techniques in Nuclear and particle Physics”, Springer

[5] R.Mankel,”Pattern Recognition and Event Reconstruction in Particle Physics Experiments”, arXiv:0402039 (2004)

[6] A. Strandlie and R. Fruhwirth, “ Track and vertex reconstruction: from classical to adaptive methods”,Reviews of Modern Physics, 82 (2010) 1419

[7] T. Lenz, “Vertex Fitting in the ATLAS Inner Detector”, Diplomarbeit, WUD-0606 (2006)

[8] ”TRT alignment for SR1 cosmics and beyond”, A. Bocci and W. Hulsbergen, ATLAS-INDET-PUB-2007-009

Physical and material constants

Quantity	symbol	value
Speed of light	c	299792458 m/s
Planck's const	h	$6.6260696 \times 10^{-34}$ J s
Planck's const, reduced	$\hbar = h/2\pi$	$6.5821193 \times 10^{-22}$ MeV s
Conversion const	$\hbar c$	197.326963 MeV fm
Electron charge	e	$1.602176490\ 10^{-19}$ C
Electron mass	m_e	0.51099891 MeV/c^2
Proton mass	m_p	938.27201 MeV/c^2
Atomic mass unit	u	931.494 MeV/c^2
Atomic mass unit	u	$1.6605388 \times 10^{-27}$ kg
Vacuum permittivity	ε_0	8.854188×10^{-12} F m^{-1}
Vacuum permeability	$\mu_0 = 1/\varepsilon_0 c^2$	$4\pi \times 10^{-7}$ N A^{-2}
Fine structure const	$\alpha = e^2/4\pi\varepsilon_0\hbar c$	1/137.036
Classical electron radius	$r_e = e^2/4\pi\varepsilon_0 m_e c^2$	$2.81794029 \times 10^{-15}$ m
Bohr radius	$a_\infty = r_e\alpha^{-2}$	$0.529177209 \times 10^{-10}$ m
Rydberg energy	$hcR = m_e c^2\alpha^2/2$	13.6056919 eV
Thomson cross section	$8\pi r_e^2/3$	0.665246 barn
Bohr magneton	$\mu_B = e\hbar/2m_e$	$5.78838176 \times 10^{-11}$ MeV T^{-1}
Avogadro const	N_A	6.0221418×10^{23} mol^{-1}
Boltzmann const	k	8.617343×10^{-5} eV K^{-1}

Table 1.1:

Material	Z	A	Nucl. int. length g cm^{-2}	Rad. Length g cm^{-2}	$\frac{dE}{dx}$(min) MeV g^{-1} cm^2	density g cm^{-3} (g l^{-1})	Refr. index
H_2	1	1.008	52	63.04	4.104	0.071(0.084)	1.11
He	2	4.003	71.0	94.32	1.937	0.125(0.166)	1.02
Be	4	9.012	77.8	65.19	1.595	0.534	1.02
Graphite	6	12.011	85.8	42.70	1.742	2.210	
dry air	7	14	90.1	36.6	1.815	1.205	
water			83.3	36.08	1.992	1.	1.33
CO_2			88.9	36.20	1.819	1.563(1.842)	
Al	13	26.98	107.2	24.01	1.615	2.699	
Si	14	28.086	108.4	21.82	1.664	2.329	3.95
Ar	18	39.95	119.7	19.55	1.519	1.574(2.98)	1.32
Fe	26	55.85	132.1	13.84	1.451	7.874	
Cu	29	63.55	137.3	12.86	1.403	8.96	
W	74	183.84	191.9	6.76	1.145	19.3	
Pb	82	207.2	199.6	6.37	1.122	11.35	

Table 1.2: Gasses refer to NTP (20° 1 atm). When there are two numbers, the first is for liquid and the second for gas.