Track Reconstruction in KM3NeT

Brían Ó Fearraigh, Bouke Jung

November 17, 2026

1 Intro

This is - at the moment - meant to be a brief introduction to the reconstruction process in KM3NeT, and what it all means.
This document is meant to be a guide for newcomers, but also contain information useful to those familiar with the code.

2 What Is Reconstruction?

At the end of the day, reconstruction means fitting a model (specifically, model parameters) to data. The underlying, most important one for us is the Cherenkov cone hypothesis. Relativistic charged particles produce Cherenkov radiation, which propagates through our detector as a cone of light. Neutrino interactions are usually separated into 'track' events and 'shower' events. A neutrino interaction, depending on the neutrino flavour, can result in a muon coming out of the interaction point - a track - or can create a cascade of particles - hadrons or electrons - which is termed a shower. The separate event types are dealt with using separate reconstruction algorithms. The model we use assumes that a muon traverses the detector and emits Cherenkov radiation as it propagates, and the same for particles produced in a shower.

Now, what do we want to 'reconstruct'? We settle for wanting to find out the energy and direction of the muon tracks, as well as the energy and direction of showers. In muon track reconstruction, the track length is very much tied to the energy of the muon. Approximately, a 1 GeV muon travels 4 metres. For showers, this is trickier. A shower profile is not simply proportional to its energy: the number of particles varies, for example. ORCA and ARCA employ different shower reconstruction algorithms, because of the geometrical difference between the detectors. For **track reconstruction** the method is practically detector-independent.
This act of reconstructing tracks and showers is not so simple. In KM3NeT there is a 'chain' of steps taken to carry out track reconstruction. If someone refers to the '*reco chain*' or '*JMuonChain*', this is what they mean. The chain comprises several applications:

`JMuonPrefit` $\longrightarrow$ `JMuonSimplex` $\longrightarrow$ `JMuonGandalf` $\longrightarrow$ `JMuonStart` $\longrightarrow$ `JMuonEnergy`,

the first three of which fit the position and direction of track events within the detector and the last two of which estimate the length and energy of the track. As an input either a DAQ file or a simulation file can be given. Raw simulation data needs to be preprocessed by the program `JTriggerEfficiency`, before it can be handed to the reconstruction chain. This program simulates the background light (from K-40 decays), simulates the response of PMTs to photon hits, and applies a trigger to these hits. The neutrino events in these files are termed 'post-trigger'.

Now, a few brief notes on this chain. Previously, it has been concluded that `JMuonSimplex` does not contribute much to the angular resolution (see e.g. this ELOG post). Nevertheless the step is included, since it reduces the CPU time required for fit convergence in `JMuonGandalf`, as reported in this ELOG Post.

3 Track Reconstruction

The following section aims to give a brief, not overly technical description of these steps in the reconstruction chain. If more information is needed, please consult the code itself.
We want to fit to our data the hypothesis that a muon track causes hits in the KM3NeT detector. In reconstructing a muon event, we want to describe the important parameters: the position and direction of the muon at each point in time, giving 5 independent parameters. This is a non-linear problem.
In this case, performing an initial fit gives us a starting point for determining the muon direction. Neglecting light scattering and assuming a muon direction reduces the non-linear problem to a linear one. Then a full fit and maximum likelihood estimation can be carried out, followed by an estimation of the muon energy.
If you are not too familiar with `JPP`, the script name followed by -h! prints out a small description of the code and the list of default values that are used. Note that the 'default' parameters when running this chain differ between ARCA and ORCA. The text files with the reference values an be found in *JPP_DATA*[1].

3.1 JMuonPrefit

`JMuonPrefit` does what it says on the tin. It performs an initial fit in the detector, taking N track direction hypotheses in all directions. By assuming a direction, the remaining parameters to fit to - or 'reconstruct' - are the position of the muon and time at which it crosses some reference plane perpendicular to the muon direction. The parameters describing the muon are obtained by an iterative procedure to minimise the χ^2. The χ^2 is based on the time difference between the photon arrival time and its expected arrival time, and is normalised to an assumed time resolution.
`JMuonPrefit` saves a specified amount of these fits, and then hands them to `JMuonGandalf` to perform the full direction fit and reconstruct the muon trajectory.

A coordinate system is defined by taking an assumed muon direction, where the muon travels parallel to the z-axis and crosses the $z = 0$ plane at (x_0,y_0) at time t_0. The muon trajectory passes the PMT of the detector at a *distance of closest approach* R, defined by

$$R_j = \sqrt{(x_j - x_0)^2 + (y_j - y_0)^2}\,. \tag{1}$$

The expected arrival time t_j of the Cherenkov photons on the PMT j can be expressed as

$$t_j = t_0 + \frac{z_j + R_j \tan(\theta_c)}{c}\,, \tag{2}$$

with z_j the distance to the z=0 plane, c the speed of light in a vacuum, and θ_c the Cherenkov angle. A derivation can be found in A.1.

[1] These values are configurable, and investigations into changing the input parameters for `JMuonPrefit` → `JMuonGandalf` are described in the eLog here and here. Note the road width is an important parameter; `JMuonPrefit` and `JMuonGandalf` appear to need a road width of ≈ 50 metres for a clean hit selection.

Defining:

$$t'_j = \frac{t_j c}{\tan(\theta_c)} - \frac{z_j}{\tan(\theta_c)}\,, \tag{3}$$

and

$$t'_0 = \frac{t_0 c}{\tan(\theta_c)}. \tag{4}$$

This gives, for all pairs of consecutive hits i,j the relation:

$$\begin{aligned} t'^2_j - t'^2_i - 2(t'_j - t'_i)t'_0 = x_j^2 - x_i^2 - 2(x_j - x_i)x_0 \\ + y_j^2 - y_i^2 - 2(y_j - y_i)y_0\,. \end{aligned} \tag{5}$$

See A.2 for the derivation.
Note that x_0, y_0 and t_0 appear in a linear manner. Therefore, a matrix equation can be set up which contains one row for each pair of consecutive hits. This equation can be solved using a least squares minimisation. Appendix A.3 contains further details.
In `JMuonPrefit`, a cluster of causally related hits is selected from the data (in 1D), in order to minimise the linear fit being affected by the optical background. Possible outliers are removed as long as their contribution to the total χ^2 is larger than 3 standard deviations.
This procedure is repeated for a specified grid angle (the default value is currently 5° for ORCA), which is used to scan across the whole solid angle of the sky. The N best-fit test directions are stored and used in the next stage of the reconstruction as start points for a full fit. The 'best' fit is determined by the fit quality parameter Q (something that appear often), and in `JMuonPrefit` is given by

$$Q = \mathrm{NDF} - \frac{1}{4}\frac{\chi^2}{\mathrm{NDF}}\,, \tag{6}$$

where NDF is the number of degrees of freedom in the fit.
Dividing by the number of degrees of freedom is done in order to weigh the fit directions with small associated hit statistics with those fit directions with a large number of hits. The division by 4 is chosen to provide sufficient separation between the qualities of the individual fits.

3.2 JMuonGandalf

`JMuonGandalf` is our be-all and end-all of vertex and direction reconstruction for tracks. The algorithm takes the fits from `JMuonPrefit` (either all the fits or the fit with the highest quality, you decide), performs a scan *around* the corresponding track hypotheses and minimises a chi-square function in terms of the track direction and position. This chi-square estimation is based on semi-analytical arrival time distributions, which describe the number of photo-electrons observed per unit time by a PMT situated at a radial distance r_i from the track and oriented at angles θ_i and ϕ_i away from it. The arrival time distributions take into account the quantum efficiencies, transit-time spreads and angular acceptances of the PMT and contain individual contributions from direct and single scattered Cherenkov light from different sources, including the ionisation and radiative energy losses of muons, the light emission from electromagnetic and hadronic showers and the light emission by delta-rays created along the track. The wavelength-dependent absorption and dispersion of light are also factored in. A full description and derivation of the arrival time PDFs can be found in a separate document on the `Jpp doxygen` [5].

Using the arrival time distributions, it is possible to define probability density functions (PDFs) which describe the chance of observing a first photo-electron at a relative time dt with respect to the arrival

time expectation for a direct hit caused by an unscattered Cherenkov photon [6]. First, the probability to observe no hit until time dt needs to be defined. This probability is given by Poisson-statistics. If the instantaneous rate of photo-electrons observed by PMT i is given by the arrival time distribution $n(dt_i, r_i, \theta_i, \phi_i)$, the total number of photo-electrons expected up until time dt can be defined:

$$N(dt, r_i, \theta_i, \phi_i) = \int_{dt_0}^{dt} n(\tau, r_i, \theta_i, \phi_i) d\tau. \tag{7}$$

In this, dt_0 corresponds to an arbitrary starting point, typically assumed to be the start of the DAQ timeslice. The probability that no photo-electrons are registered up until relative time dt corresponds to:

$$P(N = 0; dt, r_i, \theta_i, \phi_i) = \frac{N^0}{0!} e^{-N} = e^{-N(dt, r_i, \theta_i, \phi_i)}. \tag{8}$$

To obtain the aforementioned probability density function for the registration of a first photo-electron at relative time dt, this needs to be multiplied with the instantaneous detection rate and normalised to one. Integrating over the total time window $[dt_0, dt_1]$ on which $n(dt_i, r_i, \theta_i, \phi_i)$ is defined, the normalisation factor is found to be:

$$\int_{dt_0}^{dt_1} n(dt, r_i, \theta_i, \phi_i) e^{-N(dt, r_i, \theta_i, \phi_i)} = \left[-e^{-N(dt, r_i, \theta_i, \phi_i)} \right]_{dt_0}^{dt_1} = 1 - e^{-N_{\text{tot}}}. \tag{9}$$

Therefore, the expression for the PDF of the first hit time registration becomes:

$$f(dt; r_i, \theta_i, \phi_i) = \frac{1}{1 - e^{-N_{\text{tot}}}} n(dt, r_i, \theta_i, \phi_i) e^{-N(dt, r_i, \theta_i, \phi_i)}. \tag{10}$$

Equation 10 can be used to test the hypothesis that a track particle is responsible for a given cluster of hits within the detector (H_1). In this case, the track hypothesis needs to be contrasted to the null hypothesis, which assumes the hit pattern was caused by background radiation only (H_0). The likelihood that a cluster of hits with relative detection times dt_m was caused by background only can be defined as:

$$\begin{aligned} \mathcal{L}(dt_1, dt_2, dt_3, ..., dt_{M-1}, dt_M | H_0) &= \prod_{m=1}^{M} f_B(dt_m; R_m) \\ &= \prod_{m=1}^{M} \frac{n_B(dt_m, R_m)}{1 - e^{-N_{B,\text{tot}}}} e^{-N_B(dt_m, R_m)}. \end{aligned} \tag{11}$$

In this, M corresponds to the total number of hits within the cluster and R_m corresponds to the background rate observed by the PMT which registered hit m, which is assumed to be constant over the time of the event. Furthermore, $f_B(dt_m; R_m)$ corresponds to the first hit time PDF for PMT m in the case of background hits only, which is dependent upon the arrival time distribution for background-induced photo-electrons $n_B(dt_m, R_m)$ and its cumulative distribution $N_B(dt_m, R_m)$ in an analogous way to equation 10.

The likelihood that the same cluster of hits is caused by a track signature on top of the background can be written down as:

$$\mathcal{L}(dt_1, dt_2, dt_3, ..., dt_{M-1}, dt_M|H_1) = \prod_{m=1}^{M} f_{S+B}(dt_m; r_m, \theta_m, \phi_m, R_m)$$
$$= \prod_{m=1}^{M} \frac{n_{S+B}(dt_m, r_m, \theta_m, \phi_m, R_m)}{1 - e^{-N_{S+B,\mathrm{tot}}}} e^{-N_{S+B}(dt_m, r_m, \theta_m, \phi_m, R_m)}, \quad (12)$$

where $n_{s+B} = n_B + \sum_k n_k$ constitutes the sum over all sources of light k originating from the track, in addition to the random background. Dividing equation 11 by equation 12 and taking the logarithm yields:

$$\lambda = \ln \frac{\mathcal{L}(dt_1, dt_2, dt_3, ..., dt_{M-1}, dt_M|H_0)}{\mathcal{L}(dt_1, dt_2, dt_3, ..., dt_{M-1}, dt_M|H_1)}, \quad (13)$$

which is asymptotically distributed as a chi-square distribution according to Wilk's theorem. The minimum of λ in terms of the track parameters corresponds to the track which is most compatible with the signal hypothesis and least with the background-only model. `JMuonGandalf` tries to find this minimum in an iterative way. The minimisation is performed using the Levenberg-Marquardt algorithm, which allows for continuous interpolation between the method of gradient descent and Gauss-Newton minimisation. The latter of these two sub-dependent methods relies on information on the curvature of the chi-square landscape in order to determine which step direction most efficiently and reliably approaches the minimum. Because calculations of (second) derivatives are numerically expensive, the gradient of the chi-square landscape is analytically computed within `JMuonGandalf`. The final chi-square value corresponding to the obtained minimum is saved and is used to define the quality metric $Q = -\chi^2$ of the fit. The larger the quality, the better the fit.

3.3 JMuonStart

In `JMuonStart`, the observed start position of muon trajectory is determined by back-projecting the hits onto the track under the Cherenkov angle. The first associated emission point which exceeds the random background level is selected as the start position, and the last point as the end point of the track. The vertex position of the track is then moved to the determined start point.

3.4 JMuonEnergy

`JMuonEnergy` determines the energy of the muon. This is done based on the hit/no-hit likelihood for all PMTs in a cylindrical volume surrounding the track hypothesis. The relative longitudinal position of the base of the cylinder with respect to the event vertex and the radial size of the cylinder can be modified via the options `-@ZMin_m` and `-@roadWidth_m` respectively. For each PMT in the cylinder surrounding the muon track, the number of photon hits induced by the ^{40}K-background, by delta-rays, by electromagnetic showers and by the ionisation and radiative energy losses of the muon are estimated from the tabulated `Jpp` PDFs. In this, the closest distance of approach between the muon and the PMT and the incidence angle of the incoming light are taken into account. Given an expected number of hits as function of the muon energy μ_i, the probability that the i-th PMT observes n hits in reality will be given by the Poisson distribution:

$$P(n; \mu_i) = \frac{\mu_i^n}{n!} e^{-\mu_i}. \quad (14)$$

This means that the probability that the PMT observes no hit is given by:

$$P(n=0;\mu_i) = e^{-\mu_i}, \tag{15}$$

whilst the probability that the PMT observes any number of hits above zero is:

$$P(n>0;\mu_i) = 1 - P(n=0;\mu_i) = 1 - e^{-\mu_i}. \tag{16}$$

`JMuonEnergy` maximizes the likelihood $L(E)$ of the configuration of hits in the cylinder surrounding the muon track as a function of the muon energy. In practice, this is done using an M-estimation method which minimizes a cost function $c(E)$ as function of the energy. The cost function is defined differently for energy reconstruction in ORCA and for energy reconstruction in ARCA. In ORCA, it corresponds directly to the negative log-likelihood of the hit-configuration in the cylinder surrounding the muon track:

$$c(E) \equiv -\ln\left(L(E)\right) = -\sum_{i=1}^{N} \ln\left[\left(1 - 2e^{-\mu_i(E)}\right)\mathbf{I}_{n>0} + e^{-\mu_i(E)}\right]. \tag{17}$$

In this, N corresponds to the total number of PMTs in the cylinder. The term $\mathbf{I}_{n>0}$ evaluates to 1 for all $n > 0$ and to 0 otherwise. In ARCA on the other hand, the cost function comprises a sum over Lorentzian functions $\rho(x) = 1 + \frac{1}{2}x^2$, where x corresponds to the logarithmic hit/no-hit probabilities, i.e.:

$$c(E) = \sum_{i=1}^{N}\left[1 + \frac{1}{2}\ln^2\left[\left(1 - 2e^{-\mu_i(E)}\right)\mathbf{I}_{n>0} + e^{-\mu_i(E)}\right]\right]. \tag{18}$$

This cost function was chosen based on the observation that it improves the energy resolution for tracks with high energy [7]. In contrast to this, the energy resolution for ORCA degrades when using a Lorentzian cost function [8, 9].

Once the cost functions have been set up, they are minimized over the track energy using a five-point search within a user-defined logarithmic energy range. This energy range is set via the options `-@EMin_log` and `-@EMax_log`. For each iteration of the search, five logarithmically equidistant energy points are defined. The cost function for each corresponding energy value is calculated according to equation 17 or 18. Subsequently, the limits of the energy range are adjusted such that the point which yielded the maximum likelihood falls in the middle. Five new logarithmically equidistant energy points are defined within the updated range, after which the search is repeated until the difference between the smallest and largest energy within the range falls below a user-specifiable threshold, set via the option `-@resolution`. The final energy and the corresponding cost function value are stored as part of the track fit parameters under `JENERGY_ENERGY` and `JENERGY_CHI2` respectively. Other track fit parameters that are saved include the bounds of the energy range used in the last iteration of the 5-point search (`JENERGY_MINIMAL_ENERGY` and `JENERGY_MAXIMAL_ENERGY`), the number of operational PMTs within the cylinder surrounding the track hypothesis (`JENERGY_NDF`), the corresponding total number of hits (`JENERGY_NUMBER_OF_HITS`), and the length of the muon track, computed from the final energy estimate and the average energy loss factor per unit of length (`JENERGY_MUON_RANGE_METRES`).

References

[1] ANTARES-SOFT-2007-001, M. de Jong.

[2] Multi-dimensional interpolations in C++, M. de Jong, 2019.

[3] KM3NeT/ARCA Event Reconstruction Algorithms, Melis et al. 2017.

[4] Following the Light, PhD thesis, B. Ó Fearraigh 2024.

[5] The probability density function of the arrival time of light, M. de Jong, 2021.

[6] Likelihood thoughts, A. Heijboer, 2020.

[7] Computing and Software ELOG #518, B. Jung, 2021.

[8] Computing and Software ELOG #248, L. Quinn, 2018.

[9] Computing and Software ELOG #498, B. Ó Fearraigh, 2021.

[10] The KM3NeT Detector Description Data Format, K. Graf, 2021.

[11] Detector Calibration, M. de Jong, 2021.

[12] Dynamic position calibration, M. de Jong, 2021.

[13] Dynamic orientation calibration, M. de Jong, 2021.

A Appendix

A.1 Arrival Time Derivation

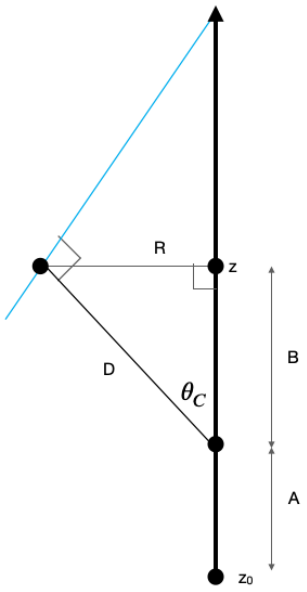


Figure 1: Diagram of muon travelling in z-direction. The blue line depicts the Cherenkov shock front.

Consider a muon, originating from a point z_0 (where it crosses the $z = 0$ plane), travelling in the z-direction. The muon travels a distance A, at which it emits a Cherenkov photon. The distance to the PMT from this point is denoted as D. The muon travels another distance, B, until it is at the point of closest approach to the PMT; this distance is denoted by R - see 1 - and this point is denoted by z. This geometry is shown in Figure 1.
So, the time for the light to reach the PMT will be

$$t_1 = t_0 + \frac{A}{c} + \frac{D}{c_{\text{water}}} \implies c \cdot t_1 = c \cdot t_0 + A + D \cdot n \,. \tag{19}$$

with n the refractive index of the medium, c_{water} the speed of light in water, and c the speed of light in a vacuum ($c_{\text{water}} = c/n$). With an emission angle of $cos\theta = \frac{1}{n \cdot \beta}$, and $\beta \approx 1$, we have $cos\theta_C = \frac{1}{n}$.
Thus,

$$c \cdot t_1 = c \cdot t_0 + A + \frac{D}{cos\theta_C} \,. \tag{20}$$

The following relations can be made:

$$sin\theta_C = \frac{R}{D} \,, \tag{21}$$

$$cos\theta_C = \frac{B}{D} \,, \tag{22}$$

and

$$A = (z - B) \,. \tag{23}$$

Now, the arrival time becomes

$$c \cdot t_1 = c \cdot t_0 + z - D \cdot cos\theta_C + \frac{R}{sin\theta_C \cdot cos\theta_C} = c \cdot t_0 + z - \frac{R}{sin\theta_C} \cdot cos\theta_C + \frac{R}{sin\theta_C \cdot cos\theta_C} \,. \tag{24}$$

Re-arranging:

$$c{\cdot}t_1 = c{\cdot}t_0{+}z{+}\big(\frac{-R \cdot cos^2\theta_C + R}{sin\theta_C \cdot cos\theta_C}\big) = c{\cdot}t_0{+}z{+}\big(\frac{R(1 - cos^2\theta_C)}{sin\theta_C \cdot cos\theta_C}\big) = c{\cdot}t_0{+}z{+}\big(\frac{R \cdot sin^2\theta_C}{sin\theta_C \cdot cos\theta_C}\big) = c{\cdot}t_0{+}z{+}\big(\frac{R \cdot sin\theta_C}{cos\theta_C}\big) \,. \tag{25}$$

This becomes

$$t_1 = t_0 + \frac{z + R \cdot tan\theta_C}{c} \,. \tag{26}$$

In reality the arrival times will not conform to the behaviour described by equation 26 exactly. There are several factors which can contribute this, such as inter-module and inter-PMT time-offsets. These time-offsets are taken into account using calibration data derived from dedicated procedures [11, 12, 13], which are fed into the reconstruction algorithms via the detector file [10]. However, even if these global time-offsets have been corrected for, fluctuations in the arrival time are still expected as a consequence of PMT-internal effects.

One important effect is time-slewing. Time-slewing refers to the dependence of the measured hit times on the hit amplitudes. Typically, the larger the amplitude of a PMT pulse, the quicker the pulse reaches the detection threshold set by the hardware discriminator and the earlier the hit is registered. Conservatively, the lower the amplitude, the longer it takes for the pulse to reach the threshold and the later it is registered. As a consequence of this dependency, hits can be observed several nanoseconds in front or behind the expected arrival time. A correction is applied to every hit which partakes in the reconstruction algorithms to account for this. The magnitudes of the corrections are based on data obtained from muon calibrations with the application `JFrodo` and are tabulated in the function object `JGetRiseTime`.

A.2 Derivation of Hit Relation

From Equation 3 and 4 we get the expressions

$$t_j = \frac{t'_j \tan(\theta_c) + z_j}{c} \,, \tag{27}$$

and

$$t_0 = \frac{t'_0 \tan(\theta_c)}{c} \,. \tag{28}$$

Substituting these into Equation 2 gives:

$$t'_j \tan(\theta_c) + z_j = t'_0 \tan(\theta_c) + z_j + R_j \cdot \tan(\theta_c) \,, \tag{29}$$

$$\implies t'_j = t'_0 + R_j \,. \tag{30}$$

$$\implies t'_j - t'_0 = \sqrt{(x_j - x_0)^2 + (y_j - y_0)^2} \,, \tag{31}$$

$$\implies \left(t'_j - t'_0\right)^2 = (x_j - x_0)^2 + (y_j - y_0)^2 \,. \tag{32}$$

This equation corresponds to a cone and relates the **fit parameters** (x_0, y_0, t'_0) to the **hit parameters** (x_j, y_j, t'_j)

$$\begin{aligned} \therefore\ t_j'^2 - 2t'_j t'_0 + t_0'^2 = x_j^2 - 2x_j x_0 + x_0^2 \\ + y_j^2 - 2y_j y_0 + y_0^2 \,. \end{aligned} \tag{33}$$

Following the same logic for a hit on the PMT i with expected arrival time t_i (with distance of closest approach R_i)

$$\begin{aligned} t_i'^2 - 2t'_i t'_0 + t_0'^2 = x_i^2 - 2x_i x_0 + x_0^2 \\ + y_i^2 - 2y_i y_0 + y_0^2 \,. \end{aligned} \tag{34}$$

Subtracting Equation 34 from Equation 33:

$$\begin{aligned} t_j'^2 - t_i'^2 - 2t'_j t'_0 + 2t'_i t'_0 = x_j^2 - x_i^2 - 2x_j x_0 + 2x_i x_0 \\ + y_j^2 - y_i^2 - 2y_j x_0 + 2y_i x_0 \,. \end{aligned} \tag{35}$$

This gives the result shown in Equation 5

$$\begin{aligned} t_j'^2 - t_i'^2 - 2(t'_j - t'_i)t'_0 = x_j^2 - x_i^2 - 2(x_j - x_i)x_0 \\ + y_j^2 - y_i^2 - 2(y_j - y_i)y_0 \,. \end{aligned} \tag{36}$$

A.3 Matrix Equation

Equation 5 can be formulated in a matrix equation considering all pairs of consecutive hits:

$$H\,\Theta = Y \tag{37}$$

where

$$H = \begin{pmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) & -2(t'_2 - t'_1) \\ 2(x_3 - x_2) & 2(y_3 - y_2) & -2(t'_3 - t'_2) \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ 2(x_1 - x_n) & 2(y_1 - y_n) & -2(t'_1 - t'_n) \end{pmatrix}, \tag{38}$$

$$Y = \begin{pmatrix} x_2^2 - x_1^2 + y_2^2 - y_1^2 - t_2'^2 + t_1'^2 \\ x_3^2 - x_2^2 + y_3^2 - y_2^2 - t_3'^2 + t_2'^2 \\ \cdot \\ \cdot \\ x_1^2 - x_n^2 + y_1^2 - y_n^2 - t_1'^2 + t_n'^2 \end{pmatrix}, \tag{39}$$

and

$$\Theta = \begin{pmatrix} x_0 \\ y_0 \\ t'_0 \end{pmatrix} . \tag{40}$$

The number of pairs of consecutive hits amounts to the number of hits n, taking into account the pair of the first and last hit. The optimal (least squares) solution to 37 is

$$\Theta = \left(H^T\, V^{-1}\, H\right)^{-1} \times H^T\, V^{-1} \times Y \tag{41}$$

with V the standard co-variance matrix, which can be expressed as

$$V = J \times \begin{pmatrix} \sigma_1^2 & . & . & . & 0 \\ . & \sigma_2^2 & . & . & . \\ . & . & \sigma_3^2 & . & \\ . & . & . & . & . \\ 0 & . & . & . & \sigma_n^2 \end{pmatrix} \times J^T . \tag{42}$$

with σ_j the resolution of the measured time t_j.

The *Jacobian matrix* J is expressed as

$$J = \begin{pmatrix} 2t'_1 & -2t'_2 & 0 & . & . & . \\ 0 & 2t'_2 & -2t'_3 & 0 & . & . \\ . & 0 & 2t'_3 & -2t'_4 & . & . \\ . & . & . & . & . & . \\ -2t'_1 & . & . & . & . & 2t'_n \end{pmatrix} , \tag{43}$$

with $J_{ij} = \delta Y_i \;/\; \delta t_j$.

Thus the co-variance matrix V is an $n \times n$ which at first order is (almost) tridiagonal:

$$V = \begin{pmatrix} (2t'_1\sigma'_1)^2 + (2t'_2\sigma'_2)^2 & -(2t'_2\sigma'_2)^2 & 0 & . & . & -(2t'_{21}\sigma_1)^2 \\ -(2t'_2\sigma'_2)^2 & (2t'_2\sigma'_2)^2 + (2t'_3\sigma'_3)^2 & -(2t'_3\sigma'_3)^2 & 0 & . & . \\ 0 & -(2t'_3\sigma'_3)^2 & (2t'_3\sigma'_3)^2 + (2t'_4\sigma'_4)^2 & . & . & . \\ . & 0 & . & . & . & . \\ . & . & . & . & . & . \\ -(2t'_1\sigma_1)^2 & . & . & . & . & (2t'_n\sigma'_n)^2 + (2t'_1\sigma'_1)^2 \end{pmatrix} , \tag{44}$$

where $\sigma'_j = \sigma_j c \,/\, \tan\theta_c$. Neglecting the off-diagonal elements of V, the number of operations needed to obtain the solution Θ is proportional to n (the number of hits).