PROCEEDINGS OF SCIENCE

Pattern recognition and estimation methods for track and vertex reconstruction

Rudolf Frühwirth*
Institute of High Energy Physics of the Austrian Academy of Sciences, Vienna, Austria
E-mail: fru@hephy.oeaw.ac.at

Are Strandlie
Gjøvik University College and University of Oslo, Norway
E-mail: are.strandlie@hig.no

The reconstruction of charged tracks and interaction vertices is an important step in the data analysis chain of particle physics experiments. We give a survey of the most popular methods that have been employed in the past and are currently employed by the LHC experiments. Whereas pattern recognition methods are very diverse and rather detector dependent, fitting algorithms offer less variety and can be applied to both track and vertex estimation with minimal changes. In particular, we trace the development from standard least-squares estimators to robust and adaptive estimators in both contexts.

13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research, ACAT2010
February 22-27, 2010
Jaipur, India

*Speaker.

PoS(ACAT2010)003

1. Introduction

Track and vertex reconstruction are essential steps in the data analysis chain of high-energy physics experiments. The performance of the algorithms is a crucial factor in quality of physics analysis. Track and vertex reconstruction are also of growing importance for the trigger, both in the first and in higher levels.

The task of track reconstruction is to determine the location, the direction and the (inverse) momentum of charged tracks. The task of vertex reconstruction is to determine the location of an interaction point and the momenta of the participating tracks. Track and vertex reconstruction have many basic features in common.

Both track and vertex reconstruction usually proceed in three steps: track/vertex finding (pattern recognition), track/vertex fitting (estimation), and track/vertex quality check (testing).

1. **Pattern recognition.** The assignment of the detector hits to the tracks is unknown a-priori and has to be determined by a pattern recognition algorithm. The latter is usually highly dependent on the detector and the shape of the magnetic field. In the case of vertex finding, tracks are assigned to vertex candidates. This problem is nearly detector independent. The performance of the pattern recognition is strongly influenced by the amount of background. In the case of track finding, the background consists mainly of hits from low-momentum tracks, secondary tracks, and pile-up tracks. In the case of vertex finding, the main source of background is low-momentum tracks and pile-up tracks.

2. **Estimation.** The track fit estimates the track parameters at one or several points along the track. This requires the following ingredients:

   - The track model, i.e., the solution of the equation of motion of the charged particle in the magnetic field. The solution can be analytical or numerical.
   - The amount of material crossed by the particle. It can be obtained from the pattern recognition stage.
   - The covariance matrix of the observation errors. It has to be determined in the course of the detector calibration.
   - The effect of the material on the trajectory, mainly multiple scattering, bremsstrahlung and energy loss by ionization. It can be computed to a good approximation from theory [1].

   The vertex fit estimates the position of the interaction vertex and updates the momenta of the participating tracks. The necessary ingredients are:

   - The track model. It is the same as in the track fit.
   - The track parameters and their covariance matrices. They are provided by the track fit.

   If the track parameters are sufficiently close to the vertex location, no material effects have to be taken into account.

From the mathematical point of view, the two estimation problems have a nearly identical structure. Consequently, both estimation procedures can be formulated as extended Kalman filters [2].

3. **Testing.** After the track fit, the quality of the track candidate is checked. Besides a test on the χ^2 statistic of the track, track length and the overlap with other tracks can be used to assess the quality. If the quality is insufficient, one may try to improve it by identifying and removing outliers. The quality check of vertex candidates proceeds along the same lines.

The approach outlined above, with its strict separation between pattern recognition and estimation, can be called the *classical* approach. Recently, the boundary has become more fuzzy. In the *adaptive* approach, the pattern recognition is reduced to a preliminary selection of hits or tracks. The final decision, which hits (tracks) are to be included in a track (vertex), is deferred to the estimation stage. In some cases, the pattern recognition step can be dispensed with entirely. Typically, an adaptive method gives soft associations of hits to tracks and of tracks to vertices, in the form of posterior weights or probabilities. In the testing phase the final decision about which hits (tracks) to retain can be based on these posterior weights.

A welcome side-effect of adaptive estimators is their *robustness*, i.e., the fact that they are less sensitive to outliers than the classical least-squares estimators. As outliers are automatically down-weighted, an explicit removal of outliers is often unnecessary.

2. Track finding

We can make a rough distinction between *local* or *sequential* methods on the one hand and *global* or *parallel* methods on the other hand. A local method first generates small track segments, called *seeds*, and then completes the seeds to full track candidates. In contrast, a global method finds the track candidates simultaneously, by some sort of clustering algorithm.

2.1 Local methods

1. **Track road.** The algorithm starts with selecting two or three hits in the detector. From these, it computes an approximative track plus a tolerance band around the track. Finally, it picks up hits inside the tolerance band.

2. **Track following.** The algorithm starts with the construction of an initial track segment (seed) from two or three hits. The seed is extrapolated, and matching hits are attached to the track candidate. This is repeated until last detector layer.

   The seed can be constructed at the outer edge of the tracking detector, where track density is usually lower, or at the inner edge, where in many cases detector resolution and two-track resolution is higher.

3. **Progressive track finding.** This is a statistically refined version of track following, i.e., an extended Kalman filter [3, 4]. The algorithm starts with the construction of an initial track segment (seed) from two or three hits. The seed is extrapolated, and the best matching hit inside the tolerance is selected. The track parameters are updated with the information of

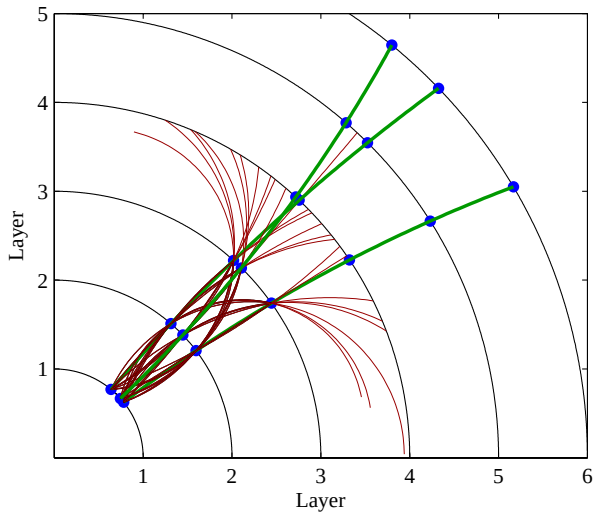

Figure 1: An example of progressive track finding with three tracks. Out of the 27 seeds that are generated in the innermost three layers, only three survive at the outermost detector layer. The blue dots are the detector hits, the red lines are the seeds, and the green line are the found tracks.

this hit. This is repeated until last detector layer. An illustration of the procedure is shown in Figure 1.

2.2 Global methods

1. **Hough transform.** The Hough transform [5] is a transformation from the image space to a parameter space. A point (x_0, y_0) in the image space (x/y) is transformed into the straight line $d = y_0 - kx_0$ in parameter space (k/d). Points on the straight line $y = k_0 x + d_0$ in image space are transformed into lines intersecting in the point (k_0/d_0) in parameter space. If the parameter space is discretized, intersections of lines can be found by histogramming and subsequent peak finding. For circle finding, circles through the origin can be transformed to lines by a conformal transformation [6].

   Alternatively, a point (x_0, y_0) on a circle through the origin with center u/v is transformed into the straight line

   $$v = -\frac{x_0}{y_0} u + \frac{x_0^2 + y_0^2}{2y_0}$$

   in parameter space (u/v). Points on the circle with center u_0/v_0 in image space are transformed into lines intersecting in the point (u_0/v_0) in parameter space. An example is shown in Figure 2.

2. **Legendre transform.** The Legendre transform is a generalization of the Hough transform and can be used for track finding in drift tubes [7]. The drift circles are transformed to sine curves in polar coordinates. The intersections of several sine curves in the parameter space correspond to common tangents to the drift circles. For an illustration, see Figure 3.

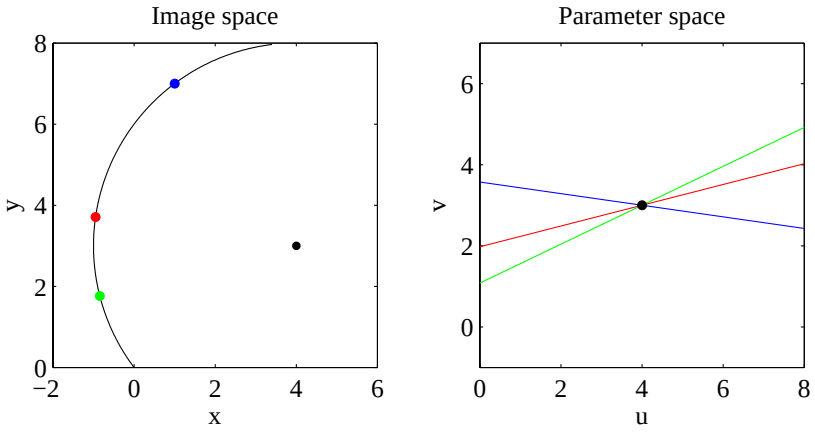

Figure 2: Finding circles with the Hough transform. The colored points in image space (left) are transformed into the corresponding colored lines in parameter space (right). The intersection of the lines in parameter space gives the circle center in image space.

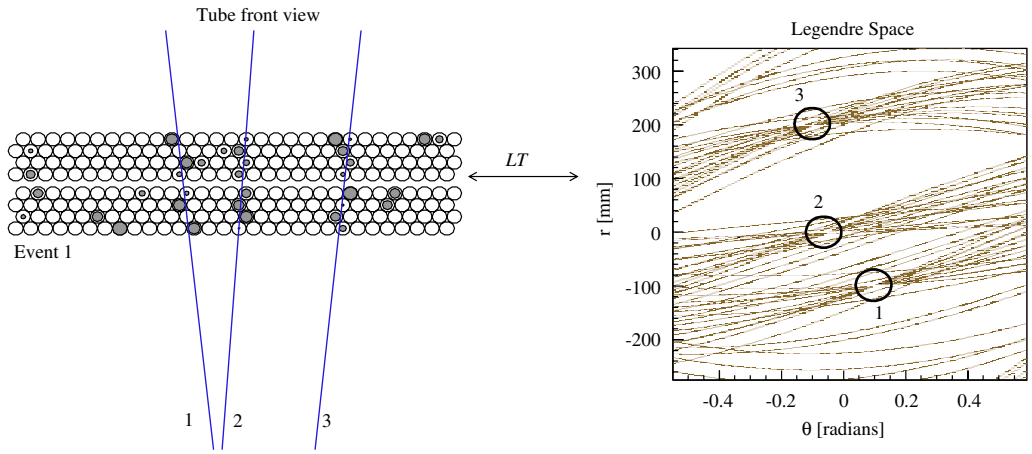

Figure 3: Track finding in drift tubes with the Legendre transform. From [7].

3. **Hopfield network.** Track finding with a Hopfield network can be considered as the first adaptive approach to track reconstruction [8, 9, 10]. Short track segments are the neurons of a recursive artificial neural network (ANN). The network weights are constructed such that smooth tracks without bifurcations are favored. The energy function of the ANN is minimized by gradient descent, and deterministic annealing helps to find global optimum. Note that the weights do not use a physical track model. The initial and final state of a network is shown in Figure 4

4. **Elastic net.** The elastic net is a special type of neural network, related to Traveling Salesman Problem. The neurons are attracted to the detector hits and to each other [11]. An example

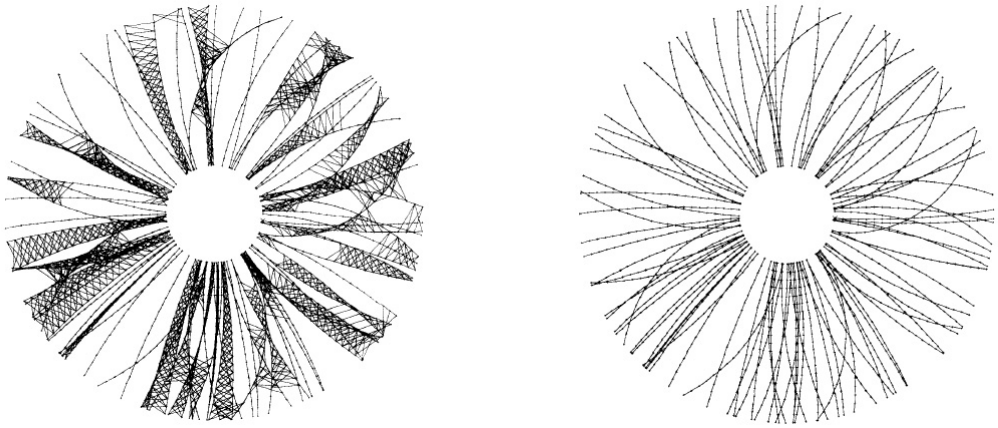

Figure 4: Initial state (left) and final state (right) of a Hopfield network, using simulated tracks. From [10].

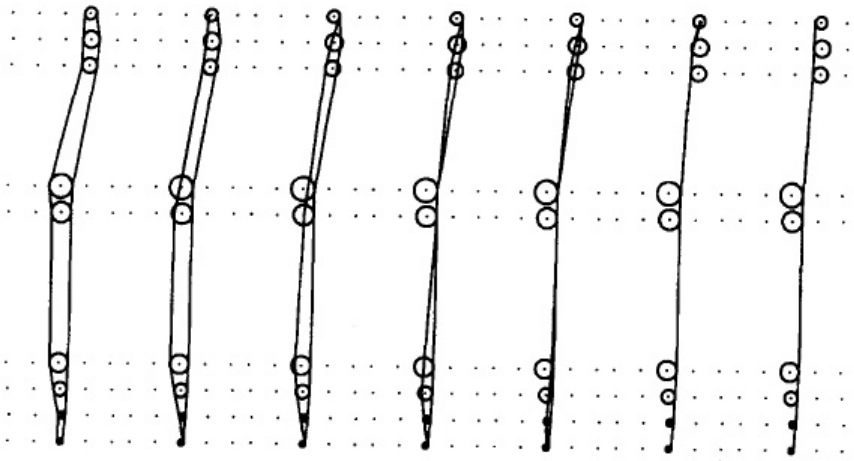

Figure 5: Example of the elastic net for a track with multiple scattering. From [11].

is shown in Figure 5. Elastic nets can also be used for ring finding in RICH detectors [12].

5. **Cellular automaton.** Track finding with a cellular automaton is described in [13, 14]. The cells of the automaton are space points or track segments. The update rules of the automaton are defined such that a sequence of cells is found that maximizes track length and track smoothness.

3. Track fitting

3.1 Classical approach

Traditionally, after pattern recognition each track candidate is passed to a least-squares esti-

mator. The estimator can be implemented in three different ways [15]: as a non-linear regression model, as a breakpoint fit, and as a recursive estimator (extended Kalman filter). All three are *optimal* in the linear case with normal noise.

1. **Regression.** The regression model is in general non-linear:

$$\boldsymbol{m} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\varepsilon}, \quad \mathrm{Cov}[\boldsymbol{\varepsilon}] = \boldsymbol{V} = \boldsymbol{G}^{-1},$$

   where $\boldsymbol{m}$ is the vector of all observations, $\boldsymbol{x}$ is the vector of initial track parameters, $\boldsymbol{h}$ is the track model, $\boldsymbol{\varepsilon}$ is the stochastic noise (measurement errors plus multiple scattering), and $\boldsymbol{V}$ is its covariance matrix. The following objective function is minimized to obtain the estimate $\tilde{\boldsymbol{x}}$:

$$M(\boldsymbol{x}) = (\boldsymbol{m} - \boldsymbol{h}(\boldsymbol{x}))^{\mathrm{T}} \boldsymbol{G} (\boldsymbol{m} - \boldsymbol{h}(\boldsymbol{x})), \quad \tilde{\boldsymbol{x}} = \arg\min M(\boldsymbol{x})$$

   There is a large choice of minimization methods: Gauss-Newton, Newton-Raphson, conjugate gradients, and many others. The usual test statistics are the total $\chi^2 = M(\tilde{\boldsymbol{x}})$, and the standardized residuals (pulls).

2. **Breakpoint fit.** In the breakpoint fit the multiple scattering angles are estimated explicitly. Prior information about multiple scattering angles is used:

$$\mathrm{E}[\vartheta_{\mathrm{p}}] = 0, \quad \mathrm{var}[\vartheta_{\mathrm{p}}] = \sigma^2(m, p, d)$$

   where ϑ_{p} is the projected scattering angle, m is the mass of the particle, p is the momentum of the particle, and d is the thickness of the material at the breakpoint.

3. **Kalman filter.** The Kalman filter is a recursive least-squares estimator, so no large matrices need to be inverted. The estimated state vectors stay *close* to the actual track. The track is described as *discrete dynamic system* [2]. The evolution of the track is described by the *system equation*:

$$\boldsymbol{x}_k = \boldsymbol{f}_k(\boldsymbol{x}_{k-1}) + \boldsymbol{\delta}_k, \quad \mathrm{Cov}[\boldsymbol{\delta}_k] = \boldsymbol{Q}_k,$$

   where $\boldsymbol{x}_k$ is the state vector (local track parameters) in layer k, $\boldsymbol{f}_k$ is the local track model, and $\boldsymbol{\delta}_k$ is the local process noise (multiple scattering). The dependence of the observations on the local state is described by the *measurement equation*:

$$\boldsymbol{m}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{\varepsilon}_k, \quad \mathrm{Cov}[\boldsymbol{\varepsilon}_k] = \boldsymbol{V}_k,$$

   where $\boldsymbol{m}_k$ is the measurement in layer k, $\boldsymbol{h}_k$ is the measurement model, and $\boldsymbol{\varepsilon}_k$ is the measurement error.

   The Kalman filter proceeds *recursively* by alternating two steps: *prediction* and *update*. In the prediction step the state vector and its covariance matrix is propagated to the next layer, and the covariance matrix is incremented by contributions from multiple scattering and energy loss. In the update step, *weighted mean* of the extrapolation and the observation is computed. There is a local χ^2 statistic, its number of degrees being equal to the dimension of $\boldsymbol{m}_k$. The sum of the local χ^2 statistics gives the total χ^2 of the track. The prediction and update (filter) step is illustrated in Figure 6.

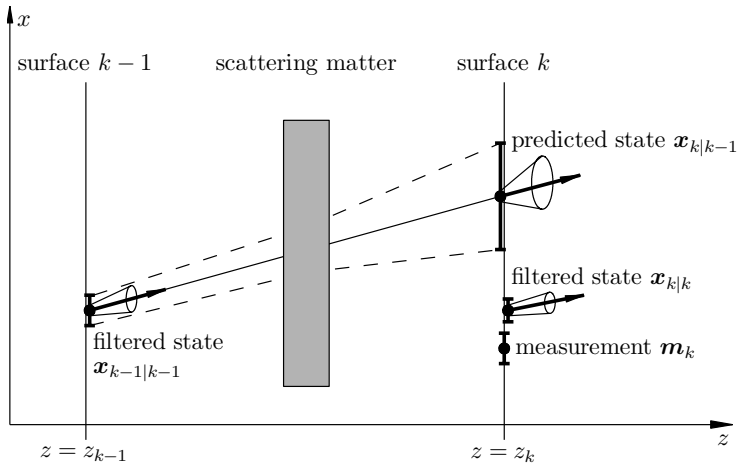

Figure 6: Prediction and filter step

The filter can be complemented by a *smoother*. The smoother computes optimal estimation of the track state in *each layer* of the tracking detector. The smoother is best implemented by combining two filters running forward and backward by a weighted mean. The smoother gives an additional test statistic, the local χ^2 of the smoother.

3.2 Adaptive approach

As a standard least-squares estimator the Kalman filter is sensitive to outliers, i.e., *not robust*. As a consequence, it is difficult to identify multiple outliers, because each outlier biases the entire track. A solution to this problem is the application of *adaptive estimators*. They have the following advantages:

- Pattern recognition and estimation are performed *concurrently*. The final decision is deferred to the fitting stage, where the complete information about the track is available.
- Adaptive estimators feature *competition* of observations. If several observations in a layer are compatible with the track candidate, the estimator automatically selects the globally best match.
- Adaptive estimators feature *automatic suppression of background*. This reduces the bias due to outliers, and there is no need to remove or add hits during the fits.
- The assignment of hits to the track is "soft" during the entire fit. If required, the assignment can be made "hard" after the optimal solution to the assignment problem has been found. Finding the global optimum can be assisted by introducing *deterministic annealing*.

Adaptive estimators can be implemented in various ways. Among them are:

- Elastic arms, deformable templates: based on neural network paradigm [16];
- Elastic tracking: inspired by Radon transform [17];

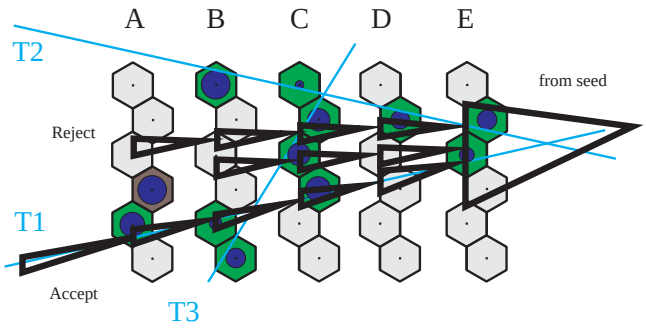

Figure 7: The combinatorial Kalman filter. From [18].

- Combinatorial Kalman filter: full discrete combinatorial exploration [18];
- Gaussian-sum filter: based on mixture models of measurement or process noise [19];
- Deterministic annealing filter: inspired by EM algorithm [20].

For reasons of space, we will not discuss elastic tracking and the Gaussian-sum filter; the reader is referred to the literature.

The *combinatorial Kalman filter* is an extension of progressive track finding. Its defining feature (and limitation) is a full combinatorial exploration. At each stage of the filter, several candidates are propagated in parallel. For each candidate, a branch is generated for each compatible hit, and optionally a branch with a missing hit. The growth of the candidates is limited by dropping branches with bad total chi-square, branches with too many missing hits, and branches that are subsets of other candidates. Upon reaching the last layer, the "best" branch is selected as the final track. An example is shown in Figure 7.

The *elastic arms* or *deformable templates* algorithm can be considered as the first truly adaptive estimator. The arms or templates are parameterized tracks. The method computes the concurrent solution of two optimization problems. The first one is continuous: minimize a least-squares objective function; the second one is discrete: decide which hit belongs to which template. The discrete problem is transformed into a continuous one by *deterministic annealing*. The resulting non-quadratic energy function is minimized at each temperature in the annealing schedule.

The deterministic annealing filter (DAF) is based on the same principle as the elastic arms algorithm. Minimization is done, however, by the EM algorithm, implemented as an *iterated reweighted Kalman filter*. It is therefore easy to deal with the process noise (multiple scattering). A *weight* is assigned to each observation. The DAF iterates two principal steps:

1. Full Kalman filter+smoother, using the current weights, and
2. calculation of weights, using the current estimates of the track parameters.

The iteration ends when the weights are stable. The weight of observation i in layer k is defined by:

$$p_{ik} = \frac{\exp(-\chi^2_{ik}/2T)}{\exp(-\chi^2_{\text{cut}}/2T) + \sum_j \exp(-\chi^2_{jk}/2T)},$$

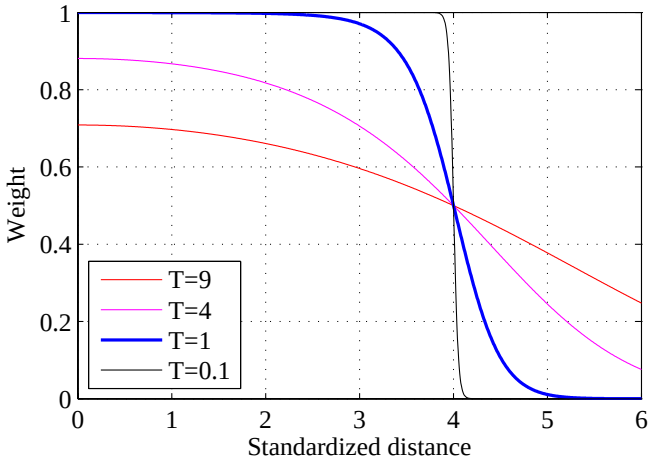

Figure 8: Weight function of an observation without competition.

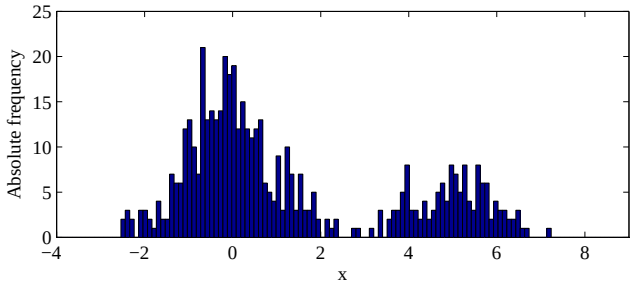

Figure 9: An artificial data set with outliers. The distance between the inlier mean and the outlier mean is equal to $m = 5$.

where χ^2_{ik} measures the *distance* of observation i in layer k from the smoothed track state in layer k, χ^2_{cut} is the *cut-off* parameter, and T is the *annealing* factor (temperature). For a single observation $p_{ik} = 0.5$ if $\chi^2_{ik} = \chi^2_{\text{cut}}$. If an observation has no competitor, its weight function has the form shown in Figure 8.

The robustness of the adaptive estimator is illustrated by the following example. Figure 9 shows an artificial data set with outliers. The aim is to estimate the location of the inliers. The evolution of the objective function is shown in Figure 10. At the starting temperature ($T = 5$) the objective function is nearly quadratic, and the minimum is close to the sample mean including the outliers. As the temperature is lowered, the objective function starts to reflect the structure of the data. At the final temperature ($T = 0.1$), the estimate is very close to zero, the mean of the inliers.

If there are several observations in a detector layer, they may *compete* with each other, so that a matching observation *suppresses* the other ones. Deterministic Annealing helps to reach the globally optimal solution. At positive temperature the association of hits to the track is "soft". Cooling down to $T = 0$ yields "hard" association, but this is not necessarily optimal. The weight function of an observation with competition is shown in Figure 11.

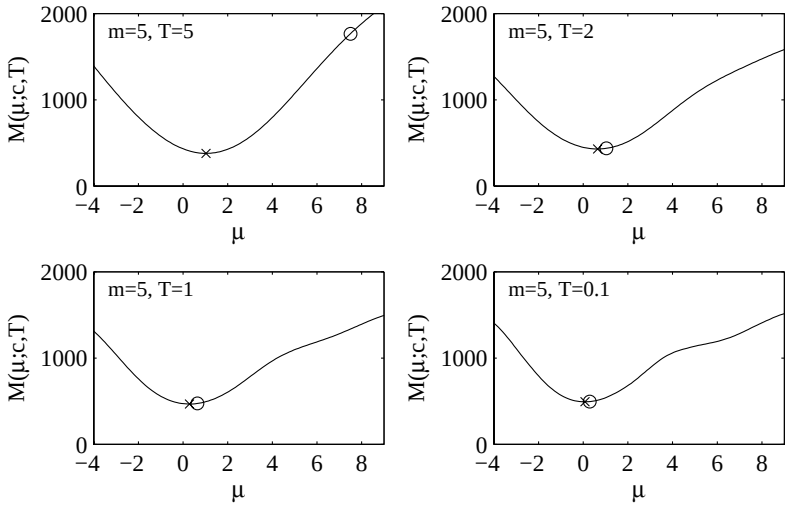

Figure 10: Evolution of the objective function of the adaptive estimator. At each temperature, the circle (○) is the starting value of the estimate, and the cross (×) is the minimum of the objective function.

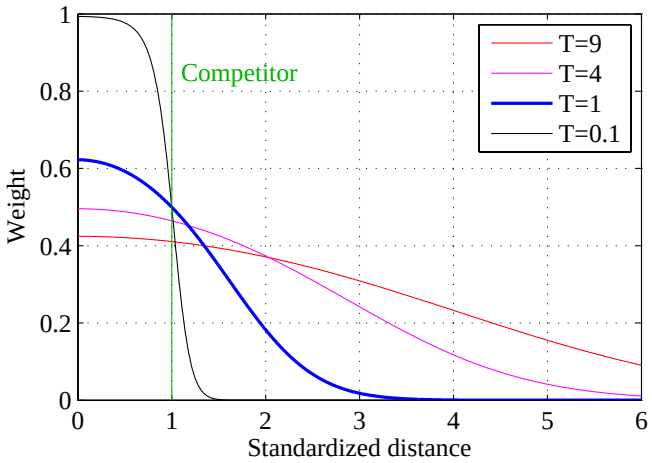

Figure 11: Weight function of an observation with competition. The competitor is at one standard deviation from the track.

The DAF was evaluated in the CMS experiment in different physics contexts [21] and implemented in the reconstruction software of CMS and ATLAS. A study dedicated to the track finding capabilities of the DAF can be found in [22].

4. Vertex reconstruction

4.1 Vertex finding

Vertex finding can be accomplished by many different types of algorithms. A few of them are:

- Hierarchical clustering of the agglomerative or divisive type;

- Topological vertex finding, similar to a Radon transform [23];
- Minimum spanning tree [24];
- Multi-layer perceptron [25];
- Adaptive vertex reconstructor (see below).

4.2 Vertex estimation

Least-square estimation of the vertex position can be performed via a non-linear regression model or, alternatively, by an extended Kalman filter. As the basics of track and vertex estimation are very similar, the concept of adaptive estimation can be transferred almost one-to-one from track to vertex fitting. The resulting algorithm is called the *Adaptive Vertex Fitter* (AVF, [26]). In analogy to the DAF, it is implemented as *iterated re-weighted Kalman filter*. As outlying tracks are automatically down-weighted, the resulting estimator is highly robust, but much easier to compute than other robust estimators such as least median of squares (LMS) or least trimmed squares (LTS). The adaptive vertex fitter can be extended to a *Multi-Vertex Fitter (MVF)*, in which several vertices compete for the available tracks.

The AVF can also be used for concurrent vertex finding and vertex fitting. The algorithm is called the *Adaptive Vertex Reconstructor* (AVR, [27]). Vertices are found by iterating the AVF:

- Fit all tracks to a common vertex, using the AVF
- Remove all tracks with weight above threshold
- Fit all remaining tracks to a common vertex, using the AVF
- Repeat until no valid vertex can be fitted

The AVF has been implemented and successfully validated in the CMS offline software. All CMS algorithms (KF, AVF, AVR, ...) have been packed into a detector-independent vertex reconstruction toolkit called RAVE [28]. It has been used in the new Belle II framework as well as for ILC studies (ILD, SiLC).

5. Conclusions and Outlook

Adaptive estimators are useful tools for track and vertex reconstruction. For a detailed exposition and experimental results the reader is referred to a recent review [29]. Adaptive estimators have several advantages over conventional least-squares estimators:

- Background is automatically down-weighted. There is no need for iterative rejection of outliers.
- Competition between hits or vertices is possible.
- Deterministic annealing helps to reach the globally optimal solution of the hits-to-track or track-to-vertices assignment problem.

- Implementations can be built on existing methods such as the Kalman filter.
- Adaptive estimators are resistant to high levels of noise. This is important for future experiments at the Super-LHC and at the upgraded B-factory at KEK.

Because of their iterative nature, adaptive estimators are intrinsically slower than least-squares estimators. On the other hand, their deployment can be restricted to particularly difficult cases such as narrow jets or detector regions with high background. In addition, there is important work on *parallelization* of track reconstruction going on [30, 31]. It should be investigated how far adaptive estimators can profit from these developments.

Acknowledgments

We thank our colleagues Wolfgang Adam, Thomas Speer, Wolfgang Waltenberger and Matthias Winkler, who have made important contributions to the development and implementation of adaptive methods in track and vertex reconstruction.

References

[1] C. Amsler et al., *The Review of Particle Physics*, *Phys. Lett. B* **667** (2008) 1.

[2] R. Frühwirth, *Application of Kalman filtering to track and vertex fitting*, *Nucl. Instrum. Meth. Phys. Res. A* **262** (1987) 444.

[3] P. Billoir and S. Qian, *Simultaneous pattern recognition and track fitting by the Kalman filtering method*, *Nucl. Instrum. Meth. Phys. Res. A* **294** (1990) 219.

[4] P. Billoir and S. Qian, *Further test for the simultaneous pattern recognition and track fitting by the Kalman filtering method*, *Nucl. Instrum. Meth. Phys. Res. A* **295** (1990) 492.

[5] P. V. C. Hough, *Machine analysis of bubble chamber pictures*. In *Proceedings of the International Conference on High Energy Accelerators and Instrumentation*, CERN, Geneva 1959.

[6] M. Hansroul, H. Jeremie, and D. Savard, *Fast circle fit with the conformal mapping method*, *Nucl. Instrum. Meth. Phys. Res. A* **270** (1988) 498.

[7] T. Alexopoulos et al., *Implementation of the legendre transform for track segment reconstruction in drift tube chambers*, *Nucl. Instrum. Meth. Phys. Res. A* **592** (2008) 456.

[8] B. Denby, *Neural networks and cellular automata in experimental high energy physics*, *Comput. Phys. Commun.* **49** (1988) 429.

[9] C. Peterson, *Track finding with neural networks*, *Nucl. Instrum. Meth. Phys. Res. A* **279** (1989) 537.

[10] G. Stimpfl-Abele and L. Garrido, *Fast track finding with neural networks*, *Comput. Phys. Commun.* **64** (1991) 46.

[11] I. Kisel and V. Kovalenko, *Elastic net for broken multiple scattered tracks*, *Comput. Phys. Commun.* **98** (1996) 45.

[12] S. Lebedev, *Fast Parallel Ring Recognition Algorithm in the RICH Detector of the CBM Experiment at FAIR*, In *Proceedings of the 13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, PoS(ACAT2010)060.

[13] I. Kisel et al., *Cellular automaton and elastic net for event reconstruction in the NEMO-2 experiment*, *Nucl. Instrum. Meth. Phys. Res. A* **387** (1997) 433.

[14] I. Abt et al., *Cellular automaton and Kalman filter based track search in the HERA-B pattern tracker*, *Nucl. Instrum. Meth. Phys. Res. A* **490** (2002) 546.

[15] R. Frühwirth et al., *Data Analysis Techniques for High-Energy Physics, 2nd edition*, Cambridge University Press, Cambridge 2000.

[16] M. Ohlsson, C. Peterson, and A. L. Yuille, *Track finding with deformable templates — the elastic arms approach*, *Comput. Phys. Commun.* **71** (1992) 77.

[17] M. Gyulassy and M. Harlander, *Elastic tracking and neural network algorithms for complex pattern recognition*, *Comput. Phys. Commun.* **66** (1991) 31.

[18] R. Mankel, *A concurrent track evolution algorithm for pattern recognition in the HERA-B main tracking system*, *Nucl. Instrum. Meth. Phys. Res. A* **395** (1997) 169.

[19] R. Frühwirth, *Track fitting with non-gaussian noise*, *Comput. Phys. Commun.* **100** (1997) 1.

[20] R. Frühwirth and A. Strandlie, *Track fitting with ambiguities and noise: a study of elastic tracking and nonlinear filters*, *Comput. Phys. Commun.* **120** (1999) 197.

[21] M. Winkler, *A comparative study of track reconstruction methods in the context of CMS physics*, PhD thesis, University of Technology, Vienna 2002.

[22] A. Strandlie and R. Frühwirth, *Reconstruction of charged tracks in the presence of large amounts of background and noise*, *Nucl. Instrum. Meth. Phys. Res. A* **566** (2006) 157.

[23] D. J. Jackson, *Topological vertex reconstruction algorithm for hadronic jets*, *Nucl. Instrum. Meth. Phys. Res. A* **388** (1997) 247.

[24] S. Hillert, *ZVMST: a minimum spanning tree-based vertex finder*, Technical Report LC-DET-2008-004, DESY, Hamburg 2008.

[25] C. S. Lindsey and B. Denby, *Primary vertex finding in proton–antiproton events with a neural network simulation*, *Nucl. Instrum. Meth. Phys. Res. A* **302** (1991) 217.

[26] W. Waltenberger, R. Frühwirth, and P. Vanlaer, *Adaptive vertex fitting*, *J. Phys. G: Nucl. Part. Phys.* **34** (2007) N343.

[27] W. Waltenberger, *Adaptive vertex reconstruction*, Technical Report CMS NOTE-2008/033, CERN, Geneva 2008.

[28] W. Waltenberger, W. Mitaroff, and F. Moser, *RAVE — a detector-independent vertex reconstruction toolkit*, *Nucl. Instrum. Meth. Phys. Res. A* **581** (2007) 549.
Download from http://projects.hepforge.org/rave.

[29] A. Strandlie and R. Frühwirth, *Track and vertex reconstruction: From classical to adaptive methods*, *Rev. Mod. Phys.*, in press.

[30] I. Kisel, *Parallel approach to online event reconstruction in the CBM experiment*, In *Proceedings of the 13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, PoS(ACAT2010)062.

[31] A. Lebedev, *Fast parallelized tracking algorithm for the muon detector of the CBM experiment at FAIR*, In *Proceedings of the 13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, PoS(ACAT2010)056.