On-line reconstruction algorithms for the CBM and ALICE experiments

Dissertation
for attaining the PhD degree
of Natural Sciences

submitted to the Faculty of Computer Science and Mathematics
of the Johann Wolfgang Goethe University
in Frankfurt am Main

by

Sergey Gorbunov

from Omsk, Russia

Frankfurt 2013

(D30)

FIAS Frankfurt Institute for Advanced Studies

accepted by the Faculty of Computer Science and Matematics of the Johann Wolfgang Goethe University as a dissertation.

Dean: Prof. Dr. Thorsten Theobald

Expert assessor: Prof. Dr. Volker Lindenstruth
Prof. Dr. Nicole Schweikardt
Prof. Dr. Dr. h.c. mult. Horst Stöcker

Date of the disputation: March 14, 2013

Abstract

This thesis presents various algorithms which have been developed for on-line event reconstruction in the CBM experiment at GSI, Darmstadt and the ALICE experiment at CERN, Geneve.

Despite the fact that the experiments are different — CBM is a fixed target experiment with forward geometry, while ALICE has a typical collider geometry — they share common aspects when reconstruction is concerned.

The thesis describes:

— general modifications to the Kalman filter method, which allows one to accelerate, to improve, and to simplify existing fit algorithms;

— developed algorithms for track fit in CBM and ALICE experiment, including a new method for track extrapolation in non-homogeneous magnetic field.

— developed algorithms for primary and secondary vertex fit in the both experiments. In particular, a new method of reconstruction of decayed particles is presented.

— developed parallel algorithm for the on-line tracking in the CBM experiment.

— developed parallel algorithm for the on-line tracking in High Level Trigger of the ALICE experiment.

— the realisation of the track finders on modern hardware, such as SIMD CPU registers and GPU accelerators.

All the presented methods have been developed by or with the direct participation of the author.

Acknowledgements

There are a lot of people who supported this work and contributed to it. Especially I would like to thank:

- My supervisor, Dr. Prof. Volker Lindenstruth, who gave me an opportunity to join the ALICE HLT project and was supporting me all the time. His expert knowledge in many areas of science and technologies combined with his enthusiasm in realisation of fresh ideas lead to creation of the most innovative High Level Trigger to ever exist. Concerning my work, he initiated the idea of using the modern GPU accelerators for the data processing in the HLT. He has pushed this project forward and he brought it to the production despite all the difficulties, starting from the software parallelisation and ending at the installation of the GPU hardware on the HLT cluster.

  Also I would like to mention that he managed to build an unique working group of people where everyone is a world-class expert in his area of science.

- Prof. Dr. Ivan Kisel, who supervised my work since I was his diploma student. He has contributed to at least a half of the the presented investigations, especially to all the CBM developments, and he has initiated most of them.

- I would like to thank the HLT group. I'm afraid there are too many names to mention and there are too many different reasons for thanks. Therefore I will not go to details and just thank them all.

  Still I would thank David Rohr for his contribution to the HLT tracker development and for revision of my thesis.

- The CBM people, with a special thanks to Dr. Youri Vassiliev.

- All the people from my group at H1 experiment at DESY. I started my scientific work at this experiment, and I learned there a lot. Not only about my work but also about a friendly working atmosphere in the scientific community.

Additionally I thank my wife and my daughter for their support.

Contents

Introduction

Reconstruction of events in High Energy Physics is a complicated task. Modern experiments have to process terabytes of input data produced in particle collisions. This requires development of fast and effective reconstruction algorithms, especially for the on-line event reconstruction.

The reconstruction of events includes many different tasks — track search, fit of particle trajectories, search for primary and secondary vertices, etc. Most of these tasks are fit problems or they involve the fit problems.

In the CBM and ALICE experiments track and vertex fit is performed by the Kalman filter method. This method is the main mathematical tool which is used in event reconstruction. It is also the most complicated and time-consuming tool. Therefore the speed and the accuracy of the fit mathematics are very important, especially for on-line event reconstruction where thousands of events per second should be processed.

To develop fast and accurate fit algorithms it was necessary to investigate the base Kalman filter mathematics and to expand upon it. In order to simplify the reading, all the mathematical investigations of the Kalman filter are contained in Chapter 1.

Chapter 2 describes principles of track fit by the Kalman filter; several important details are noticed. CBM and ALICE fit algorithms are presented separately with results of their implementation. The CBM Section includes description of a new method for track extrapolation in a non-homogeneous magnetic field. In the ALICE Section a new method for correction of nonlinear operators is presented.

Chapter 3 deals with fit of vertices and decayed particles. First, conventional approaches for vertex fit are described. Then, new methods are presented which significantly speed up the processing. A new method for fit of decayed particles is presented in a separate Section. The results of implementing all these methods in CBM and ALICE are given at the end of the Chapter.

The developed on-line tracking for CBM and ALICE experiments is presented in the last two Chapters:

- The 4th Chapter describes on-line event reconstruction in CBM. It includes speed-up of the Kalman filter using SIMD CPU instructions.

- In the last Chapter on-line event reconstruction in the ALICE High Level Trigger is presented with a description of the tracking algorithm and of its implementation on the GPU hardware.

Chapter 1

The Kalman filter method

Event reconstruction includes various tasks: track finding, fit of the particle trajectories, finding of the event vertex, alignment of detectors and others. Many of those tasks involve fit problems. Problems of this kind have an exact mathematical solution which can be found using the Kalman filter method [1, 2].

In short, the fit problem is to find the most probable value of an unknown quantity using a set of measurements of this quantity. For example, the track fit problem is to estimate the trajectory of a particle using the information obtained by the tracking detectors.

The Kalman filter method is a powerful technique which solves the fit problem in a very general way. Therefore it can be applied to any particular fit problem.

In this Chapter the Kalman filter method is described, then the developed modifications of the conventional filtration procedure are provided: filtration with an extended model of measurement, filtration with the correlated measurement, and filtration by the best estimator. These modifications allow the application of the Kalman filter method for fitting of decayed particles and optimisation of the standard Kalman filter application for the vertex fit.

1.1 Fit problem

Consider the formulation of the problem. The following terms will be used:

$\mathbf{x}^T$ — denotes the transpose of the vector $\mathbf{x}$.

brackets $<>$ — denote the mathematical expectation value.

covariance matrix cov(x) of a random vector $\mathbf{x}$ — matrix of covariances between elements of the vector:

$$\mathrm{cov}(\mathbf{x}) \quad = \quad < (\mathbf{x}- < \mathbf{x} >) \cdot (\mathbf{x}- < \mathbf{x} >)^T > \tag{1.1}$$

The covariance matrix is a symmetric non-negative definite matrix.[1] Its diagonal elements are the squared dispersions of the corresponding elements of the vector.

state vector $\mathbf{r}^t$ — vector of real numbers that represents the unknown quantities to be estimated (for example, parameters of a track).[2]

[1]The $n \times n$ squared matrix A is called a non-negative definite matrix when $\mathbf{x}\mathrm{A}\mathbf{x}^T \geq 0$ for all non-zero vectors $\mathbf{x}$ of the size n.

[2]here the suffix "^{t}" in $\mathbf{r}^t$ denotes the *true* value of the parameters, in contrast to $\mathbf{r}$ which is an estimate of the parameters.

measurement m — a known (measured) quantity which linearly depends on the state vector:

$$\mathbf{m} \;=\; \mathrm{H}\mathbf{r}^t + \boldsymbol{\eta} \tag{1.2}$$

where H is a (known) linear operator represented as a matrix, called **model of measurement**;

the variable $\boldsymbol{\eta}$ is a random (unknown) variable, called **measurement error**.

It is assumed that the measurement error $\boldsymbol{\eta}$ is unbiased (its math. expectation is **0**) and its covariance matrix V is known:

$$\begin{aligned} <\boldsymbol{\eta}> &= \mathbf{0} \\ <\boldsymbol{\eta}\cdot\boldsymbol{\eta}^T> &\equiv \mathrm{V} \end{aligned} \tag{1.3}$$

In case the state vector has several measurements $\mathbf{m}_k,\ k = 1\ldots n$, it is assumed that the errors of different measurements are uncorrelated:

$$\begin{aligned} \mathbf{m}_k &= \mathrm{H}_k\mathbf{r}^t + \boldsymbol{\eta}_k \\ <\boldsymbol{\eta}_{k_i}\boldsymbol{\eta}_{l_j}> &= <\boldsymbol{\eta}_{k_i}><\boldsymbol{\eta}_{l_j}> = \ 0 \end{aligned} \tag{1.4}$$

estimator r (of the state vector $\mathbf{r}^t$) – a vector which estimates the value of the (unknown) state vector according to a given set of measurements. The estimator is called **linear estimator** when it linearly depends on the measurements.

error of estimator $\boldsymbol{\epsilon}_\mathbf{r}$ — the difference between the estimated and the real value of the state vector:

$$\boldsymbol{\epsilon}_\mathbf{r} \;=\; \mathbf{r} - \mathbf{r}^t \tag{1.5}$$

bias of estimator $<\boldsymbol{\epsilon}_\mathbf{r}>$ — the mean value of the estimator error. The estimator is called **unbiased** when its bias is **0**.

mean squared error (MSE) of estimator $\sigma^\mathbf{2}_\mathbf{r}$:

$$\sigma^\mathbf{2}_\mathbf{r} \;=\; <{\boldsymbol{\epsilon}_\mathbf{r}}^T\cdot\boldsymbol{\epsilon}_\mathbf{r}> \tag{1.6}$$

best [linear unbiased] estimator — a linear estimator which is unbiased and has minimal MSE among all linear estimators.

With the terms introduced the fit problem is **to find the best linear unbiased estimator of a state vector according to a given set of measurements**.

There are two methods to find the best estimator (and thus solve the fit problem): the Least Squares Method (LSM) [2] and the Kalman Filter Method [2]. Both methods give the same result. In the LSM method the equations are simpler, while the Kalman filter method is able to solve a wider range of problems and therefore fits better to practical tasks.

To describe all the features of the Kalman filter it is necessary to extend the fit problem. Let the state vector $\mathbf{r}^t$ change from one measurement to the next in a random way:

$$\mathbf{r}^t_k \;=\; \mathrm{A}_k\mathbf{r}^t_{k-1} + \boldsymbol{\nu}_k \tag{1.7}$$

with A_k — a (known) linear operator, called **extrapolator**;

$\boldsymbol{\nu}_k$ — a random (unknown) variable, called **process noise** between $(k-1)$-th and k-th measurements.

It is assumed that the process noise $\boldsymbol{\nu}_k$ is unbiased and its covariance matrix Q_k is known:

$$\begin{aligned} <\boldsymbol{\nu}_k> &= \mathbf{0} \\ <\boldsymbol{\nu}_k \cdot \boldsymbol{\nu}_k^T> &\equiv \mathrm{Q}_k \end{aligned} \tag{1.8}$$

It is also assumed that the process noise $\boldsymbol{\nu}_k$ is uncorrelated with the process noise $\boldsymbol{\nu}_i$ and with all the measurement errors $\boldsymbol{\eta}_j$.

Since the state vector $\mathbf{r}^t$ now changes from one measurement to the next, it is necessary to reformulate the fit problem: now the goal is **to find the best estimator $\mathbf{r}_n$ of the state vector $\mathbf{r}_n^t$, which corresponds to the last measurement $\mathbf{m}_n$.**

1.2 The Kalman filter algorithm

The Kalman filter starts with a certain initial approximation $\mathbf{r} = \mathbf{r}_0$ and refines the estimator $\mathbf{r}$, consecutively processing one measurement after the other. The best estimator is obtained when the last measurement $\mathbf{m}_n$ is processed.

The fit algorithm consists of three steps shown in Figure 1.1:

1. **Initialisation step**. Choose an approximate value $\mathbf{r}_0$ of the state vector. Its covariance matrix is set to

$$\mathrm{C}_0 = \mathrm{I} \cdot \mathrm{inf} \tag{1.9}$$

   where "inf" denotes a large positive number.

2. **Extrapolation step**. When the state vector $\mathbf{r}^t$ changes between the $(k-1)$-th and the k-th measurement (1.7) then upon transfer to the k-th measurement its current estimation $\mathbf{r}_{k-1}$ also changes in the same manner:

$$\begin{aligned} \widetilde{\mathbf{r}}_k &= \mathrm{A}_k \mathbf{r}_{k-1} \\ \widetilde{\mathrm{C}}_k &= \mathrm{A}_k \mathrm{C}_{k-1} \mathrm{A}_k^T + \mathrm{Q}_k \end{aligned} \tag{1.10}$$

   with $\widetilde{\mathbf{r}}_k$ — an optimal estimation of the vector $\mathbf{r}_k^t$ according to the first $k-1$ measurements. Note that in contrast to the extrapolation operator A_k describing deterministic changes of the state vector $\mathbf{r}^t$ between the two measurements, the process noise Q_k describes random deviations of the state vector.

3. **Filtration step**. This step is the essence of the Kalman filter — here the measurement information is incorporated into the estimator and its covariance matrix. For each measurement $\mathbf{m}_k$ an estimator $\mathbf{r}_k$ which is the best estimator of the vector $\mathbf{r}_k^t$ according to the first k measurements is calculated:

$$\begin{aligned} \mathrm{K}_k &= \widetilde{\mathrm{C}}_k \mathrm{H}_k^T (\mathrm{V}_k + \mathrm{H}_k \widetilde{\mathrm{C}}_k \mathrm{H}_k^T)^{-1} \\ \boldsymbol{\zeta}_k &= (\mathbf{m}_k - \mathrm{H}_k \widetilde{\mathbf{r}}_k) \\ \mathbf{r}_k &= \widetilde{\mathbf{r}}_k + \mathrm{K}_k\, \boldsymbol{\zeta}_k \\ \mathrm{C}_k &= \widetilde{\mathrm{C}}_k - \mathrm{K}_k \mathrm{H}_k \widetilde{\mathrm{C}}_k \\ \chi_k^2 &= \chi_{k-1}^2 + \boldsymbol{\zeta}_k^T (\mathrm{V}_k + \mathrm{H}_k \widetilde{\mathrm{C}}_k \mathrm{H}_k^T)^{-1} \boldsymbol{\zeta}_k \end{aligned} \tag{1.11}$$

Here $\{\widetilde{\mathbf{r}}_k,\ \widetilde{\mathrm{C}}_k\}$ denote the best estimator and its covariance matrix, obtained in the previous step and extrapolated to the k-th measurement; $\{\mathbf{m}_k,\ \mathrm{V}_k\}$ are the k-th measurement and its covariance matrix; $\boldsymbol{\zeta}_k$ called residual; the matrix H_k is the model of the measurement;

the matrix K_k is so-called gain matrix; the value χ^2_k is the total χ^2-deviation of the obtained estimation $\mathbf{r}_k$ from the measurements $\mathbf{m}_1, \ldots \mathbf{m}_k$.

The algorithm steps 2.-3. sequentially repeat n times, for each measurement $\mathbf{m}_k,\ k = 1, \ldots n$. After the filtration of the last measurement $\mathbf{m}_n$, the obtained estimator $\mathbf{r}_n$ is the desired best estimator with the covariance matrix C_n.

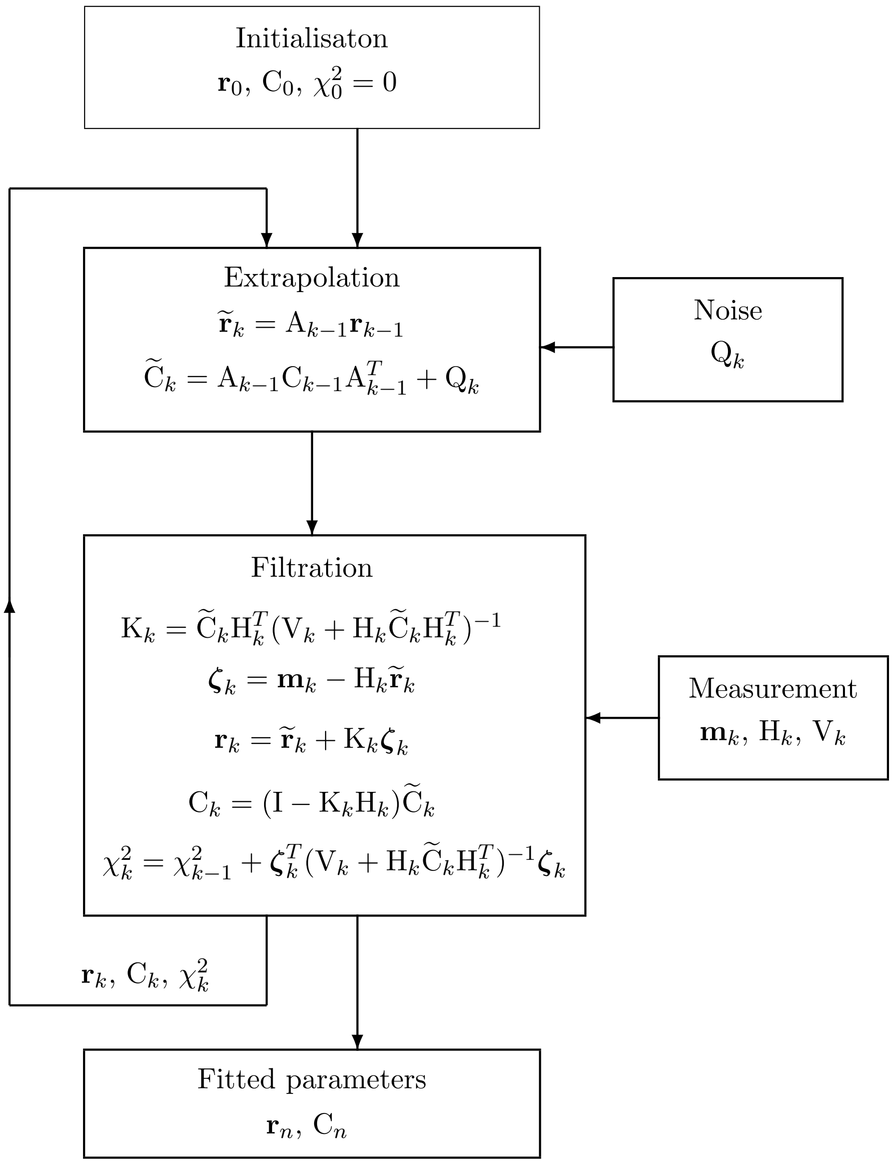

Initialisaton
$\mathbf{r}_0,\ \mathrm{C}_0,\ \chi^2_0 = 0$
Extrapolation
$\widetilde{\mathbf{r}}_k = \mathrm{A}_{k-1}\mathbf{r}_{k-1}$
$\widetilde{\mathrm{C}}_k = \mathrm{A}_{k-1}\mathrm{C}_{k-1}\mathrm{A}^T_{k-1} + \mathrm{Q}_k$
Noise
Q_k
Filtration
$\mathrm{K}_k = \widetilde{\mathrm{C}}_k\mathrm{H}^T_k(\mathrm{V}_k + \mathrm{H}_k\widetilde{\mathrm{C}}_k\mathrm{H}^T_k)^{-1}$
$\boldsymbol{\zeta}_k = \mathbf{m}_k - \mathrm{H}_k\widetilde{\mathbf{r}}_k$
$\mathbf{r}_k = \widetilde{\mathbf{r}}_k + \mathrm{K}_k\boldsymbol{\zeta}_k$
$\mathrm{C}_k = (\mathrm{I} - \mathrm{K}_k\mathrm{H}_k)\widetilde{\mathrm{C}}_k$
$\chi^2_k = \chi^2_{k-1} + \boldsymbol{\zeta}^T_k(\mathrm{V}_k + \mathrm{H}_k\widetilde{\mathrm{C}}_k\mathrm{H}^T_k)^{-1}\boldsymbol{\zeta}_k$
Measurement
$\mathbf{m}_k,\ \mathrm{H}_k,\ \mathrm{V}_k$
$\mathbf{r}_k,\ \mathrm{C}_k,\ \chi^2_k$
Fitted parameters
$\mathbf{r}_n,\ \mathrm{C}_n$

Figure 1.1: Scheme of the Kalman filter algorithm

1.3 Nonlinear Kalman filter

In practice, the transport equation (1.7) and the measurement model (1.2) are often nonlinear. To solve the nonlinear fit problem, one should linearise all the equations before applying the fitting algorithm, but the algorithm itself does not change.

When a measurement $\mathbf{m}_k$ nonlinearly depends on $\mathbf{r}^t_k$, it is necessary to linearise the model of measurement. As a point of linearisation a certain state vector $\mathbf{r}^{lin}_k$ is taken:

$$\mathbf{m}_k(\mathbf{r}^t_k) \;=\; \mathbf{h}_k(\mathbf{r}^t_k) + \boldsymbol{\eta}_k \;\approx\; \mathbf{h}_k(\mathbf{r}^{lin}_k) + \mathrm{H}_k(\mathbf{r}^t_k - \mathbf{r}^{lin}_k) + \boldsymbol{\eta}_k \tag{1.12}$$

where H_k is the Jacobian of $\mathbf{h}_k(\mathbf{r}_k)$ at $\mathbf{r}^{lin}_k$:

$$\mathrm{H}_{k\,(ij)} \;=\; \left.\frac{\partial \mathbf{h}_k(\mathbf{r}_k)_{\,(i)}}{\partial \mathbf{r}_{k\;(j)}}\right|_{\mathbf{r}_k=\mathbf{r}^{lin}_k} \tag{1.13}$$

In the same way, the nonlinear extrapolation equation 1.7 can be linearised:

$$\widetilde{\mathbf{r}}^t_k \;=\; \mathbf{a}_k(\mathbf{r}^t_{k-1}) \;\approx\; \mathbf{a}_k(\mathbf{r}^{lin}_{k-1}) + \mathrm{A}_k(\mathbf{r}^t_{k-1} - \mathbf{r}^{lin}_{k-1}) \tag{1.14}$$

$$\mathrm{A}_{k\,(ij)} \;=\; \left.\frac{\partial \mathbf{a}_k(\mathbf{r}_{k-1})_{\,(i)}}{\partial \mathbf{r}_{k-1\;(j)}}\right|_{\mathbf{r}_{k-1}=\mathbf{r}^{lin}_{k-1}} \tag{1.15}$$

The Kalman filter with the nonlinear measurement model is called the extended Kalman filter. Equations of filtration for the extended Kalman filter are the same as for the linear case (1.11) with an exception for the residual $\boldsymbol{\zeta}_k$, which is calculated according to the formula:

$$\boldsymbol{\zeta}_k \;=\; \mathbf{m}_k - \left(\mathbf{h}_k\left(\mathbf{r}^{lin}_k\right) + \mathrm{H}_k\left(\widetilde{\mathbf{r}}_k - \mathbf{r}^{lin}_k\right)\right) \tag{1.16}$$

The linearised model differs from the original one, therefore the choice of the linearisation point $\mathbf{r}^{lin}_k$ is important. The usual approach is to take the current estimator $\widetilde{\mathbf{r}}_k$ as the point of linearisation for the k-th measurement. In this case the extended Kalman filter (1.16) coincides with the linear one (1.11). This type of linearisation is not most accurate because the current estimator $\widetilde{\mathbf{r}}_k$ can be very different from the true value of the state vector $\mathbf{r}^t_k$, especially for the first measurements. In order to get more robust and precise results, the fitting procedure must be repeated several times using the obtained best estimator $\mathbf{r}_n$ as the linearisation point for all the measurements in the next iteration.

Since the preliminary linearisation of the model does not change the Kalman filter mathematics, later on all the equations will be considered linear (assuming that the linearisation has been performed).

1.4 Extensions of the Kalman filter method

In this work three modifications of the conventional filtration procedure have been developed: filtration with an extended measurement model, filtration with a correlated measurement and filtration by the best estimator.

The developed modifications allow the application of the Kalman filter method for fitting decayed particles and the optimisation of the standard Kalman filter approach for the vertex fit.

1.4.1 Filtration with an extended measurement model

The equations of filtration (1.11) can be extended for the case where the measurement $\mathbf{m}_k$ (1.2) is related to the state vector in a more general way:

$$\mathrm{G}_k(\mathbf{m}_k + \boldsymbol{\eta}_k) \;=\; \mathrm{H}_k\mathbf{r}^t_k \tag{1.17}$$

where G_k is any given matrix. In this case the measurement $\mathbf{m}_k$ should be processed by the ordinary filtration procedure (1.11) with the substitution:

$$\begin{array}{lcl} \mathbf{m}_k & \longrightarrow & \mathrm{G}_k\mathbf{m}_k \\ \mathrm{V}_k & \longrightarrow & \mathrm{G}_k\mathrm{V}_k\mathrm{G}_k^T \end{array} \tag{1.18}$$

The proof:

The proof of the formulae (1.18) is evident: the vector $\mathrm{G}_k\mathbf{m}_k$ is a measurement of the state vector, with the measurement model H_k and the measurement error $(-\mathrm{G}_k\boldsymbol{\eta}_k)$:

$$(\mathrm{G}_k\mathbf{m}_k) \;=\; \mathrm{H}_k\mathbf{r}_k^t + (-\mathrm{G}_k\boldsymbol{\eta}_k) \tag{1.19}$$

The measurement error is still unbiased, uncorellated with the estimator of the state vector, and has the covariance matrix $\mathrm{G}_k\mathrm{V}_k\mathrm{G}_k^T$:

$$\mathrm{cov}(-\mathrm{G}_k\boldsymbol{\eta}_k) \;=\; <(-\mathrm{G}_k\boldsymbol{\eta}_k)\cdot(-\mathrm{G}_k\boldsymbol{\eta}_k)^T> \;=\; \mathrm{G}_k\boldsymbol{\eta}_k\boldsymbol{\eta}_k^T\mathrm{G}_k^T \;=\; \mathrm{G}_k\mathrm{V}_k\mathrm{G}_k^T \tag{1.20}$$

Thus proving the statement.

1.4.2 Filtration with a correlated measurement

To construct a simple and fast vertex fitter described in Section 3.3 it was necessary to generalise the conditions of the conventional Kalman filter to the case where errors of different measurements are correlated.

Let us examine the k-th step of fitting a state vector $\mathbf{r}^t$ with the Kalman filter. At this stage the estimator $\widetilde{\mathbf{r}}_k$ with the covariance matrix $\widetilde{\mathrm{C}}_k$ being the best estimator of the state vector $\mathbf{r}_k^t$ according to the measurements $\mathbf{m}_1, \ldots, \mathbf{m}_{k-1}$ is already produced. Now it is necessary to improve upon the estimator by using a new measurement $\mathbf{m}_k$:

$$\begin{array}{lcl} \mathbf{m}_k & = & \mathrm{H}_k\mathbf{r}_k^t + \boldsymbol{\eta}_k \\ \mathrm{cov}(\boldsymbol{\eta}_k) & \equiv & \mathrm{V}_k \\ <\boldsymbol{\eta}_k> & = & \mathbf{0} \end{array} \tag{1.21}$$

where as before (1.4) $\boldsymbol{\eta}_k$ denotes the measurement error, V_k is the covariance matrix of the error.

In contrast to the conditions of the conventional Kalman filter, let the measurement error $\boldsymbol{\eta}_k$ be correlated with the errors of the previous measurements $\mathbf{m}_{i<k}$, and therefore correlated with the error of the estimator $\widetilde{\mathbf{r}}_k$. Assume the matrix D_k of covariances between the measurement $\mathbf{m}_k$ and the estimator $\widetilde{\mathbf{r}}_k$ is known:[3]

$$\mathrm{D}_{k\,(i,j)} \;\equiv\; \mathrm{cov}(\mathbf{m}_{k\,(i)}, \widetilde{\mathbf{r}}_{k\,(j)}) \tag{1.22}$$

In this case the standard equations of the Kalman filter (1.11) are modified as follows:[4]

$$\begin{array}{lcl} \mathrm{S}_k & = & \left(\mathrm{V}_k + \mathrm{H}_k\widetilde{\mathrm{C}}_k\mathrm{H}_k^T - \mathrm{D}_k\mathrm{H}_k^T - \mathrm{H}_k\mathrm{D}_k^T\right)^{-1} \\ \mathrm{K}_k & = & \left(\widetilde{\mathrm{C}}_k\mathrm{H}_k^T - \mathrm{D}_k^T\right)\mathrm{S}_k \\ \boldsymbol{\zeta}_k & = & \mathbf{m}_k - \mathrm{H}_k\widetilde{\mathbf{r}}_k \end{array} \tag{1.23}$$

[3]Note, that the matrix D_k can be a non square matrix.

[4]The matrix in brackets is a covariance matrix (of $\widehat{\mathrm{H}}\,\mathbf{y}_1$; see later) and therefore is invertible.

$$
\begin{aligned}
\mathbf{r}_k &= \widetilde{\mathbf{r}}_k + \mathrm{K}_k \boldsymbol{\zeta}_k \\
\mathrm{C}_k &= \widetilde{\mathrm{C}}_k - \mathrm{K}_k \left(\mathrm{H}_k \widetilde{\mathrm{C}}_k - \mathrm{D}_k \right) \\
\chi^2_k &= \chi^2_{k-1} + \boldsymbol{\zeta}_k^T \mathrm{S}_k \boldsymbol{\zeta}_k
\end{aligned}
$$

Let us note that in the case of the absence of correlations ($\mathrm{D}_k = \mathrm{O}$) the formulae (1.23) coincide with the standard Kalman filter (1.11).

In practice, when the measurement model H_k has many trivial elements (which is the case for the vertex fit), the modified Kalman filter is useful in order to reduce the calculations by splitting the multidimensional measurement into parts.

The proof:

Let us recall that the goal is to find the best estimator $\{\mathbf{r}_k,\ \mathrm{C}_k\}$ of the state vector $\mathbf{r}^t_k$ according to the measurement $\mathbf{m}_k$ and all the previous measurements.

Here the standard Kalman filter procedure can not be used directly, since its conditions are violated: there are correlations between the measurement error and the error of the current best estimator $\widetilde{\mathbf{r}}_k$.

In order to properly treat the correlations, let us substitute the measurement equation (1.21) by the equivalent set of two equations:

$$
\begin{aligned}
\mathbf{m}_k &= \mathbf{x}^t + \boldsymbol{\eta}_k && (1.24)\\
\mathbf{x}^t &= \mathrm{H}_k \mathbf{r}^t_k && (1.25)
\end{aligned}
$$

with the introduction of a temporary vector $\mathbf{x}^t$.

The first equation (1.24) is the first measurement of the temporary vector $\mathbf{x}^t$. On the other hand, the equation does not include the state vector $\mathbf{r}^t_k$. Therefore:

- the best estimator of $\mathbf{x}^t$, according to all the previous measurements and the measurement (1.24) is:

$$
\{\mathbf{x},\ \mathrm{cov}(\mathbf{x})\} = \{\mathbf{m}_k,\ \mathrm{V}_k\} \qquad (1.26)
$$

- the best estimator of $\mathbf{r}^t_k$, according to all the previous measurements and the measurement (1.24) does not change with the new measurement:

$$
\{\widetilde{\mathbf{r}}_k,\ \widetilde{\mathrm{C}}_k\} \qquad (1.27)
$$

Now let us group the state vector $\mathbf{r}^t_k$ and the temporary vector $\mathbf{x}^t$ into a combined state vector $\mathbf{y}^t$:

$$
\mathbf{y}^t = \begin{pmatrix} \mathbf{r}^t_k \\ \mathbf{x}^t \end{pmatrix} \qquad (1.28)
$$

Due to (1.26) and (1.27), the best estimator of $\mathbf{y}^t$ according to all the previous measurements $\mathbf{m}_i,\ i < k$ and the new measurement (1.24) is:

$$
\mathbf{y}_1 = \begin{pmatrix} \widetilde{\mathbf{r}}_k \\ \mathbf{x} \end{pmatrix} = \begin{pmatrix} \widetilde{\mathbf{r}}_k \\ \mathbf{m}_k \end{pmatrix} \qquad (1.29)
$$

Since by assumption the covariances D_k between $\widetilde{\mathbf{r}}_k$ and $\mathbf{m}_k$ are known, one can write the covariance matrix of the estimator $\mathbf{y}_1$:

$$
\mathrm{Y}_1 \equiv \mathrm{cov}(\mathbf{y}_1) = \begin{pmatrix} \widetilde{\mathrm{C}}_k & \mathrm{D}_k^T \\ \mathrm{D}_k & \mathrm{V}_k \end{pmatrix} \qquad (1.30)
$$

At this stage, the combined estimator $\mathbf{y}_1$ has been created and it has been proven that it is the best estimator of the combined state vector $\mathbf{y}^t$. The next step of the proof is to update the combined estimator with the second part (1.25) of the measurement.

Let us rewrite the equation (1.25):

$$
\begin{aligned}
\mathbf{0} &= \widehat{\mathrm{H}} \mathbf{y}^t \\
\widehat{\mathrm{H}} &\equiv \left(\ \mathrm{H}_k, \ \ -\mathrm{I} \ \right)
\end{aligned} \qquad (1.31)
$$

One can see that (1.31) is a measurement of the combined state vector $\mathbf{y}^t$ with the measurement value $\mathbf{0}$, the null matrix of errors (since there is no measurement error in the equation) and the measurement model $\widehat{H}$.

Since the error of the measurement (1.31) is null, it does not correlate to the error of the estimator $\mathbf{y}_1$. Thus the conditions of the standard Kalman filter are fulfilled and one can use the ordinary formula to update the combined best estimator $\mathbf{y}_1$ with this measurement.

In order to shorten the text, let us introduce two matrices A and B:

$$\begin{array}{rcl} \mathrm{A} & = & \mathrm{H}_k\widetilde{\mathrm{C}}_k - \mathrm{D}_k \\ \mathrm{B} & = & \mathrm{H}_k\mathrm{D}_k^T - \mathrm{V}_k \end{array} \tag{1.32}$$

and provide several matrices used later:

$$\begin{array}{rcl} \widehat{\mathrm{H}}^T & = & \begin{pmatrix} \mathrm{H}_k^T \\ -\mathrm{I} \end{pmatrix} \\ \widehat{\mathrm{H}}\mathrm{Y}_1 & = & \begin{pmatrix} \mathrm{A}, & \mathrm{B} \end{pmatrix} \\ \mathrm{Y}_1\widehat{\mathrm{H}}^T & = & \begin{pmatrix} \mathrm{A}^T \\ \mathrm{B}^T \end{pmatrix} \end{array} \tag{1.33}$$

According to (1.32) and (1.33) let us write down the equations of filtering (1.11) for the combined state vector $\mathbf{y}^t$:

$$\begin{array}{rclcl} \mathrm{S} & = & \left(\mathrm{O} + \widehat{\mathrm{H}}\mathrm{Y}_1\widehat{\mathrm{H}}^T\right)^{-1} & = & \left(\mathrm{A}\mathrm{H}_k^T - \mathrm{B}\right)^{-1} \\ \mathrm{K} & = & \mathrm{Y}_1\widehat{\mathrm{H}}^T\mathrm{S} & = & \begin{pmatrix} \mathrm{A}^T\mathrm{S} \\ \mathrm{B}^T\mathrm{S} \end{pmatrix} \\ \boldsymbol{\zeta} & = & \mathbf{0} - \widehat{\mathrm{H}}\mathbf{y}_1 & = & \mathbf{m}_k - \mathrm{H}_k\widetilde{\mathbf{r}}_k \\ \mathbf{y}_2 & = & \mathbf{y}_1 + \mathrm{K}\boldsymbol{\zeta} & = & \begin{pmatrix} \widetilde{\mathbf{r}}_k + \mathrm{A}^T\mathrm{S}\boldsymbol{\zeta} \\ \mathbf{m}_k + \mathrm{B}^T\mathrm{S}\boldsymbol{\zeta} \end{pmatrix} \\ \mathrm{Y}_2 & = & \mathrm{Y}_1 - \mathrm{K}\widehat{\mathrm{H}}\mathrm{Y}_1 & = & \begin{pmatrix} \widetilde{\mathrm{C}}_k & \mathrm{D}_k^T \\ \mathrm{D}_k & \mathrm{V}_k \end{pmatrix} - \begin{pmatrix} \mathrm{A}^T\mathrm{S} \\ \mathrm{B}^T\mathrm{S} \end{pmatrix} \begin{pmatrix} \mathrm{A}, & \mathrm{B} \end{pmatrix} = \\ & & & = & \begin{pmatrix} \widetilde{\mathrm{C}}_k - \mathrm{A}^T\mathrm{S}\mathrm{A} & \mathrm{D}_k^T - \mathrm{A}^T\mathrm{S}\mathrm{B} \\ \mathrm{D}_k - \mathrm{B}^T\mathrm{S}\mathrm{A} & \mathrm{V}_k - \mathrm{B}^T\mathrm{S}\mathrm{B} \end{pmatrix} \\ \chi^2_k & = & \chi^2_{k-1} + \boldsymbol{\zeta}^T\mathrm{S}\boldsymbol{\zeta} & & \end{array} \tag{1.34}$$

After taking parts which correspond to the state vector $\mathbf{r}_k^t$ from the best estimator $\{\mathbf{y}_2,\, \mathrm{Y}_2\}$, one obtains the desired best estimator $\{\mathbf{r}_k,\, \mathrm{C}_k\}$ of the state vector $\mathbf{r}_k^t$ according to all the measurements $\mathbf{m}_1, \ldots, \mathbf{m}_k$:

$$\begin{array}{rclcl} \mathbf{r}_k & = & \widetilde{\mathbf{r}}_k + \mathrm{A}^T\mathrm{S}\boldsymbol{\zeta} & = & \widetilde{\mathbf{r}}_k + \left(\widetilde{\mathrm{C}}_k\mathrm{H}_k^T - \mathrm{D}_k^T\right)\mathrm{S}\boldsymbol{\zeta} \\ \mathrm{C}_k & = & \widetilde{\mathrm{C}}_k - \mathrm{A}^T\mathrm{S}\mathrm{A} & = & \widetilde{\mathrm{C}}_k - \left(\widetilde{\mathrm{C}}_k\mathrm{H}_k^T - \mathrm{D}_k^T\right)\mathrm{S}\left(\mathrm{H}_k\widetilde{\mathrm{C}}_k - \mathrm{D}_k\right) \end{array} \tag{1.35}$$

Thus proving the statement.

1.4.3 Filtration with the best estimator

Occasionally it is necessary to fit a part of the state vector separately and then merge the fitted part with the rest of the state vector. This task appears when a particle needs to be fitted to its already reconstructed vertex.

To perform the merging, it is necessary to update the current best estimator with the fitted part, which plays the role of a measurement of the whole state vector. The error of the estimator and the error of the measurement are correlated in this situation, furthermore the correlation matrix is not known. Therefore neither the ordinary equations of filtration (1.11) nor the developed equations for the correlated measurement (1.23) can be used. This necessitates the development of a special filtering equations for this particular case.

Let the $\{\widetilde{\mathbf{r}},\, \widetilde{\mathrm{C}},\, \widetilde{\chi}^2\}$ be the best estimator of a state vector $\mathbf{r}^t$ according to a set of measurements $\widetilde{\mathrm{M}}$. Let the $\{\mathbf{m},\, \mathrm{V}\}$ be the best estimator of another state vector $\mathbf{m}^t$:

$$\mathbf{m}^t \;=\; \mathrm{H}\mathbf{r}^t \tag{1.36}$$

according to a larger set of measurements $\mathrm{M}\;:\;\widetilde{\mathrm{M}} \subset \mathrm{M}$.

Then the best estimator $\{\mathbf{r},\, \mathrm{C},\, \chi^2\}$ of the state vector $\mathbf{r}^t$, according to the measurements $\{\widetilde{\mathrm{M}}, \mathbf{m}\}$ is evaluated as follows:

$$\begin{array}{lcl} \mathrm{K} & = & \widetilde{\mathrm{C}}\mathrm{H}^T\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1} \\ \boldsymbol{\zeta} & = & \mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}} \\ \mathbf{r} & = & \widetilde{\mathbf{r}} + \mathrm{K}\,\boldsymbol{\zeta} \\ \mathrm{C} & = & \widetilde{\mathrm{C}} - \mathrm{K}\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T - \mathrm{V}\right)\mathrm{K}^T \\ \chi^2 & = & \widetilde{\chi^2} + \boldsymbol{\zeta}^T\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T - \mathrm{V}\right)^{-1}\boldsymbol{\zeta} \end{array} \tag{1.37}$$

Note that the equations (1.37) differ from the standard filtering equations (1.11) because here the errors of estimators $\widetilde{\mathbf{r}}$ and $\mathbf{m}$ are correlated.

The proof:

By the conditions of the filtration procedure (1.37), any random vector can be used as a measurement of the state vector, in particular an estimator of another state vector. The Kalman filter equations need to be reworked only when the errors of both estimators are correlated, which is the case.

To prove the equations (1.37) let us introduce a measurement $\{\mathbf{m}^0, \mathrm{V}^0\}$ — the best estimator of $\mathbf{m}^t$ according to the set of measurements $\mathrm{M}^0 = \mathrm{M} - \widetilde{\mathrm{M}}$. In the other words, $\{\mathbf{m}^0,\, \mathrm{V}^0\}$ is the estimator of $\mathbf{m}^t$, which, when updated with $\{\widetilde{\mathbf{r}},\, \widetilde{\mathrm{C}}\}$, gives the best estimator $\{\mathbf{m}, \mathrm{V}\}$.

To shorten the text, let us introduce several temporary matrices:

$$\begin{array}{lcl} \mathrm{A} & = & \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T \\ \mathrm{S} & = & \left(A + \mathrm{V}^0\right)^{-1} \end{array} \tag{1.38}$$

The desired estimator $\{\mathbf{r},\, \mathrm{C},\, \chi^2\}$ is obtained by the filtering $\widetilde{\mathbf{r}}$ with the measurement $\mathbf{m}^0$, using the ordinary filtration procedure (1.11):

$$\begin{array}{lcl} \mathrm{K}_1 & = & \widetilde{\mathrm{C}}\mathrm{H}^T\mathrm{S} \\ \boldsymbol{\zeta}_1 & = & (\mathbf{m}^0 - \mathrm{H}\widetilde{\mathbf{r}}) \\ \mathbf{r} & = & \widetilde{\mathbf{r}} + \mathrm{K}_1\;\boldsymbol{\zeta}_1 \\ \mathrm{C} & = & \widetilde{\mathrm{C}} - \mathrm{K}_1\mathrm{H}\widetilde{\mathrm{C}} \\ \chi^2 & = & \widetilde{\chi}^2 + \boldsymbol{\zeta}_1^T\mathrm{S}\boldsymbol{\zeta}_1 \end{array} \tag{1.39}$$

To exclude the unknown quantities $\{\mathbf{m}^0,\, \mathrm{V}^0\}$ from Eqs. (1.39) it is sufficient to express S and $\boldsymbol{\zeta}_1$ through the known quantities $\{\mathbf{m}, \mathrm{V}\}$.

Similar to (1.39), the estimator $\{\mathbf{m}, \mathrm{V}\}$ can be obtained by filtering $\{\mathbf{m}^0, \mathrm{V}^0\}$ with the measurement $\{\widetilde{\mathbf{r}}, \widetilde{\mathrm{C}}\}$, using the extended filtration procedure, described in Section 1.4.1:

$$\begin{array}{lcl} \mathrm{K}_2 & = & \mathrm{V}^0(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T + \mathrm{V}^0)^{-1} \;=\; \mathrm{V}^0\mathrm{S} \\ \boldsymbol{\zeta}_2 & = & (\mathrm{H}\widetilde{\mathbf{r}} - \mathbf{m}^0) \;=\; -\boldsymbol{\zeta}_1 \\ \mathbf{m} & = & \mathbf{m}^0 + \mathrm{K}_2\,\boldsymbol{\zeta}_2 \;=\; \mathbf{m}^0 - \mathrm{K}_2\,\boldsymbol{\zeta}_1 \\ \mathrm{V} & = & \mathrm{V}^0 - \mathrm{K}_2\mathrm{V}^0 \end{array} \tag{1.40}$$

Transforming the expressions for $\mathbf{m}$ and V from Eqs. (1.40):

$$\begin{array}{lcl} \mathbf{m} & = & \mathbf{m}^0 - \mathrm{V}^0\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\boldsymbol{\zeta}_1 \;=\; \mathbf{m}^0 - (\mathrm{V}^0 + \mathrm{A} - \mathrm{A})\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\boldsymbol{\zeta}_1 \\ & = & \mathbf{m}^0 - \left(\mathrm{I} - \mathrm{A}\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\right)(\mathbf{m}^0 - \mathrm{H}\widetilde{\mathbf{r}}) \\ & = & \mathrm{H}\widetilde{\mathbf{r}} + \mathrm{AS}\,\boldsymbol{\zeta}_1 \\ \\ \mathrm{V} & = & \mathrm{V}^0 - \mathrm{V}^0\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\mathrm{V}^0 \;=\; \mathrm{V}^0 - (\mathrm{V}^0 + \mathrm{A} - \mathrm{A})\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\mathrm{V}^0 \\ & = & \mathrm{V}^0 - (\mathrm{I} - \mathrm{A}\left(\mathrm{A} + \mathrm{V}^0\right)^{-1})\mathrm{V}^0 \;=\; \mathrm{A}\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}(\mathrm{V}^0 + \mathrm{A} - \mathrm{A}) \\ & = & \mathrm{A} - \mathrm{A}\left(\mathrm{A} + \mathrm{V}^0\right)^{-1}\mathrm{A} \;=\; \mathrm{A} - \mathrm{ASA} \end{array} \tag{1.41}$$

one obtains the required expressions for $\boldsymbol{\zeta}_1$ and S:

$$\begin{array}{rcl} \boldsymbol{\zeta}_1 & = & \mathrm{S}^{-1}\mathrm{A}^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ \mathrm{S} & = & \mathrm{A}^{-1}(\mathrm{A} - \mathrm{V})\,\mathrm{A}^{-1} \end{array} \tag{1.42}$$

Now one can substitute $\boldsymbol{\zeta}_1$ and S from (1.42) into (1.39):

$$\begin{array}{lcl} \mathbf{r} & = & \widetilde{\mathbf{r}} + \widetilde{\mathrm{C}}\mathrm{H}^T\mathrm{S}\boldsymbol{\zeta}_1 \;=\; \widetilde{\mathbf{r}} + \widetilde{\mathrm{C}}\mathrm{H}^T\mathrm{A}^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ \mathrm{C} & = & \widetilde{\mathrm{C}} - \widetilde{\mathrm{C}}\mathrm{H}^T\mathrm{S}\mathrm{H}\widetilde{\mathrm{C}} \;=\; \widetilde{\mathrm{C}} - \widetilde{\mathrm{C}}\mathrm{H}^T\mathrm{A}^{-1}(\mathrm{A} - \mathrm{V})\,\mathrm{A}^{-1}\mathrm{H}\widetilde{\mathrm{C}} \\ \chi^2 & = & \widetilde{\chi^2} + \boldsymbol{\zeta}_1^T\mathrm{S}\boldsymbol{\zeta}_1 \;=\; \widetilde{\chi^2} + (\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}})^T\mathrm{A}^{-1}(\mathrm{A} + \mathrm{V})\,\mathrm{A}^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ & = & \widetilde{\chi^2} + (\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}})^T(\mathrm{ASA})^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ & = & \widetilde{\chi^2} + (\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}})^T(\mathrm{A} - \mathrm{V})^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \end{array} \tag{1.43}$$

(In the last equation, the statement $\mathrm{ASA} = \mathrm{A} - \mathrm{V}$ is taken from (1.41))

After introducing a matrix K and a vector $\boldsymbol{\zeta}$ (by analogy to the notations of the ordinary filtration):

$$\begin{array}{rcl} \mathrm{K} & = & \widetilde{\mathrm{C}}\mathrm{H}^T\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1} \\ \boldsymbol{\zeta} & = & \mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}} \end{array} \tag{1.44}$$

and substituting them into Eqs. (1.43), one obtains the required equations of filtration (1.37). Thus proving the statement.

1.4.4 Subtraction of a measurement

Sometimes it is necessary to remove a wrong measurement which was previously added to a state vector. The procedure will be called *subtraction* of the measurement. This problem appears in the primary vertex reconstruction where some of the tracks are first contributed to the vertex fit, but afterwards they are recognised as non-primary tracks and have to be excluded from the already fitted vertex.

Equations for the measurement subtraction look like inverse filtering equations. Using notations of the filtering equations (1.11) but avoiding indices, let us extract pre-filtered values $\{\widetilde{\mathbf{r}}, \widetilde{\mathrm{C}}, \widetilde{\chi}^2\}$ from the measurement $\{\mathbf{m}, \mathrm{V}\}$ and the filtered values $\{\mathbf{r}, \mathrm{C}, \chi^2\}$:

$$\begin{array}{rcl} \mathrm{K} & = & \mathrm{CH}^T(\mathrm{V}-\mathrm{HCH}^T)^{-1} \\ \boldsymbol{\zeta} & = & (\mathbf{m}-\mathrm{H}\mathbf{r}) \\ \widetilde{\mathbf{r}} & = & \mathbf{r}-\mathrm{K}\,\boldsymbol{\zeta} \\ \widetilde{\mathrm{C}} & = & \mathrm{C}+\mathrm{KHC} \\ \widetilde{\chi}^2 & = & \chi^2-\boldsymbol{\zeta}^T(\mathrm{V}-\mathrm{HCH}^T)^{-1}\boldsymbol{\zeta} \end{array} \tag{1.45}$$

Remark: the equations above are only valid when the matrix of measurement errors V is invertible (which is normally the case).

The proof:

Let us recall the equation of filtration (1.11) of the covariance matrix:

$$\mathrm{C} = \widetilde{\mathrm{C}}-\widetilde{\mathrm{C}}\mathrm{H}^T(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T)^{-1}\mathrm{H}\widetilde{\mathrm{C}} \tag{1.46}$$

From (1.46) it follows that:

$$\begin{array}{rcl} \mathrm{HC} & = & \mathrm{H}\widetilde{\mathrm{C}}-\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T)^{-1}\mathrm{H}\widetilde{\mathrm{C}} \\ & = & \mathrm{H}\widetilde{\mathrm{C}}-\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T+\mathrm{V}-\mathrm{V}\right)\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}\mathrm{H}\widetilde{\mathrm{C}} \\ & = & \mathrm{V}\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}\mathrm{H}\widetilde{\mathrm{C}}\ ; \end{array} \tag{1.47}$$

$$\begin{array}{rcl} \mathrm{HCH}^T & = & \mathrm{V}\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T \\ & = & \mathrm{V}\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}\left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T+\mathrm{V}-\mathrm{V}\right) \\ & = & \mathrm{V}-\mathrm{V}\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}\mathrm{V} \end{array} \tag{1.48}$$

Equations (1.47) and (1.48) give the following expressions for the pre-filtered covariance matrix $\widetilde{\mathrm{C}}$:

$$\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1} = \mathrm{V}^{-1}\left(\mathrm{V}-\mathrm{HCH}^T\right)\mathrm{V}^{-1} \tag{1.49}$$

$$\begin{array}{rcl} \mathrm{H}\widetilde{\mathrm{C}} & = & \mathrm{V}\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{HC} \\ \widetilde{\mathrm{C}}\mathrm{H}^T & = & \mathrm{CH}^T\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{V} \end{array} \tag{1.50}$$

Substituting (1.49) and (1.50) into expression (1.46):

$$\begin{array}{rcl} \mathrm{C} & = & \widetilde{\mathrm{C}}-\left(\mathrm{CH}^T\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{V}\right)\left(\mathrm{V}^{-1}\left(\mathrm{V}-\mathrm{HCH}^T\right)\mathrm{V}^{-1}\right)\left(\mathrm{V}\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{HC}\right) \\ & = & \widetilde{\mathrm{C}}-\mathrm{CH}^T\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{HC}; \\ \widetilde{\mathrm{C}} & = & \mathrm{C}+\mathrm{CH}^T\left(\mathrm{V}-\mathrm{HCH}^T\right)^{-1}\mathrm{HC} \end{array} \tag{1.51}$$

Thus the expression for the covariance matrix $\widetilde{\mathrm{C}}$ in the statement (1.45) is proven.

Now let us recall the equation of filtration (1.11) for the state vector:

$$\mathbf{r} = \widetilde{\mathbf{r}}+\widetilde{\mathrm{C}}\mathrm{H}^T\left(\mathrm{V}+\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m}-\mathrm{H}\widetilde{\mathbf{r}}) \tag{1.52}$$

Multiplying the state vector by H:

$$\begin{aligned} \mathrm{H}\mathbf{r} &= \mathrm{H}\widetilde{\mathbf{r}} + \left(\mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T + \mathrm{V} - \mathrm{V}\right)\left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ &= \mathbf{m} - \mathrm{V}\left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \end{aligned} \tag{1.53}$$

and taking into account (1.49) one gets the expression for $(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}})$:

$$\begin{aligned} (\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) &= \left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)\mathrm{V}^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r}) \\ &= \mathrm{V}\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r}) \end{aligned} \tag{1.54}$$

Substitution of (1.49), (1.50) and (1.54) into the expression for the filtered state vector (1.52):

$$\begin{aligned} \mathbf{r} &= \widetilde{\mathbf{r}} + \widetilde{\mathrm{C}}\mathrm{H}^T\left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ &= \widetilde{\mathbf{r}} + \left(\mathrm{C}\mathrm{H}^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}\mathrm{V}\right)\left(\mathrm{V}^{-1}\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)\mathrm{V}^{-1}\right)\left(\mathrm{V}\left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r})\right) \\ &= \widetilde{\mathbf{r}} + \mathrm{C}\mathrm{H}^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r}) \ ; \\ \widetilde{\mathbf{r}} &= \mathbf{r} - \mathrm{C}\mathrm{H}^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r}) \end{aligned} \tag{1.55}$$

proves the statement (1.45) for the state vector.

Expression for the pre-filtered $\widetilde{\chi}^2$ value is proven in the same manner, by substitution of (1.49) and (1.54) into the χ^2-part of the filtering equation (1.11):

$$\begin{aligned} \chi^2 &= \widetilde{\chi}^2 + (\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}})^T\left(\mathrm{V} + \mathrm{H}\widetilde{\mathrm{C}}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\widetilde{\mathbf{r}}) \\ &= \widetilde{\chi}^2 + \left((\mathbf{m} - \mathrm{H}\mathbf{r})^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}\mathrm{V}\right)\left(\mathrm{V}^{-1}\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)\mathrm{V}^{-1}\right)\left(\mathrm{V}\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)^{-1}(\mathbf{m} - \mathrm{H}\mathbf{r})\right) \\ &= \widetilde{\chi}^2 + (\mathbf{m} - \mathrm{H}\mathbf{r})^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)(\mathbf{m} - \mathrm{H}\mathbf{r}) \ ; \\ \widetilde{\chi}^2 &= \chi^2 - (\mathbf{m} - \mathrm{H}\mathbf{r})^T\left(\mathrm{V} - \mathrm{H}\mathrm{C}\mathrm{H}^T\right)(\mathbf{m} - \mathrm{H}\mathbf{r}) \end{aligned} \tag{1.56}$$

Thus proving the statement (1.45).

1.4.5 Constrained fit with the Kalman filter

In some cases, the estimator of the state vector can be improved by taking into account several assumptions on the state vector. These assumptions are expressed in terms of constraints (or penalties) on the state vector parameters. Constraints are used, for example, in the secondary vertex fit when the fitted mother particle is required to have a certain invariant mass (so-called mass constraint) or to be pointed to the primary vertex (topological constraint) [2, 12, 7, 11].

A widely used method for applying constraints on the state vector is the Lagrange method [2, 11]. In this Section first the Lagrange method is described. Then it is proven that the constraints can be treated also by the Kalman filter as ordinary measurements of the state vector, given the same result as the Lagrange method.

Lagrange method

Let us denote the state vector before the addition of a penalty as $\mathbf{r}$ and after a penalty as $\mathbf{r}_c$, and the covariance matrices as C and C_c respectively.

A constraint is an equation on the state vector parameters that has to be satisfied:

$$\mathrm{H} \cdot \mathbf{r}_c = \mathbf{0} \tag{1.57}$$

This kind of penalty will be called **hard** constraint. For determining $\mathbf{r}_c$ it is necessary to minimise $\chi^2(\mathbf{r}_c)$ with the fulfilment of (1.57):

$$\begin{cases} \chi^2(\mathbf{r}_c) & = & (\mathbf{r}_c - \mathbf{r})^T \mathrm{C}^{-1} (\mathbf{r}_c - \mathbf{r}) \;\longrightarrow\; \min \\ \mathrm{H} \cdot \mathbf{r}_c & = & \mathbf{0} \end{cases} \tag{1.58}$$

In the Lagrange method a new function χ^2_L is constructed:

$$\chi^2_L(\mathbf{r}_c, \boldsymbol{\mu}) = \chi^2(\mathbf{r}_c) + 2\boldsymbol{\mu}^T \mathrm{H}\mathbf{r}_c \;\longrightarrow\; \min \tag{1.59}$$

where $\boldsymbol{\mu}$ is called Lagrange multiplier.[5] The function χ^2_L is then minimised with respect to $\mathbf{r}_c$ and $\boldsymbol{\mu}$.

The function χ^2_L satisfies two conditions:

- χ^2_L coincides with χ^2 when the constraint is fulfilled;
- the constraint $\mathrm{H} \cdot \mathbf{r}_c = \mathbf{0}$ is fulfilled at the point of the χ^2_L minimum.

The χ^2_L minimisation:

To minimise χ^2_L one uses partial derivatives of χ^2_L with respect to $\mathbf{r}_c$ and $\boldsymbol{\mu}$:

$$\frac{1}{2}\frac{\partial \chi^2_L}{\partial \mathbf{r}_c} = \mathrm{C}^{-1}(\mathbf{r}_c - \mathbf{r}) + \mathrm{H}^T \boldsymbol{\mu} = 0 \tag{1.60}$$

$$\frac{1}{2}\frac{\partial \chi^2_L}{\partial \boldsymbol{\mu}} = \mathrm{H}\mathbf{r}_c = 0 \tag{1.61}$$

Multiplying the first equation with HC and subtracting the second equation yields

$$-\mathrm{H}\mathbf{r} + \mathrm{HCH}^T \boldsymbol{\mu} = 0 \tag{1.62}$$

[5]The coefficient 2 is undertaken to simplify further formulae.

Assuming the matrix HCH^T is non-singular one can express $\boldsymbol{\mu}$:

$$\boldsymbol{\mu} = (\mathrm{HCH}^T)^{-1}\mathrm{H}\mathbf{r} \tag{1.63}$$

and substituting into (1.60) one obtains the desired state vector $\mathbf{r}_c$:

$$\mathbf{r}_c = \mathbf{r} - \mathrm{CH}^T(\mathrm{HCH}^T)^{-1}\mathrm{H}\mathbf{r} \tag{1.64}$$

Let us denote

$$\mathrm{K} = \mathrm{CH}^T(\mathrm{HCH}^T)^{-1} \tag{1.65}$$

Since $\mathbf{r}_c = (\mathrm{I} - \mathrm{KH})\mathbf{r}$, the covariance matrix C_c is equal

$$\mathrm{C}_c = (\mathrm{I} - \mathrm{KH})\mathrm{C}(\mathrm{I} - \mathrm{KH})^T \tag{1.66}$$

Let us rewrite and simplify it:

$$\mathrm{C}_c = (\mathrm{I} - \mathrm{KH})\mathrm{C} - (\mathrm{C}(\mathrm{KH})^T - \mathrm{KHC}(\mathrm{KH})^T) \tag{1.67}$$

Taking into account the fact that

$$\begin{aligned} \mathrm{KHC}(\mathrm{KH})^T &= \mathrm{KHCH}^T\mathrm{K}^T = \mathrm{CH}^T(\mathrm{HCH}^T)^{-1}\mathrm{HCH}^T(\mathrm{HCH}^T)^{-1}\mathrm{HC} \\ &= \mathrm{CH}^T(\mathrm{HCH}^T)^{-1}\mathrm{HC} = \mathrm{C}(\mathrm{KH})^T \end{aligned}$$

one obtains

$$\mathrm{C}_c = (\mathrm{I} - \mathrm{KH})\mathrm{C} \tag{1.68}$$

Substituting (1.64) into (1.57), one obtains the $\chi^2(\mathbf{r}_c)$ value:

$$\begin{aligned} \chi^2(\mathbf{r}_c) &= (\mathrm{CH}^T(\mathrm{HCH}^T)^{-1}\mathrm{H}\mathbf{r})^T\mathrm{C}^{-1}(\mathrm{CH}^T(\mathrm{HCH}^T)^{-1}\mathrm{H}\mathbf{r}) \\ &= (\mathrm{H}\mathbf{r})^T((\mathrm{HCH}^T)^{-1}\mathrm{HC})\mathrm{C}^{-1}(\mathrm{CH}^T(\mathrm{HCH}^T)^{-1})(H\mathbf{r}) \\ &= (\mathrm{H}\mathbf{r})^T(\mathrm{HCH}^T)^{-1}(\mathrm{H}\mathbf{r}) \end{aligned} \tag{1.69}$$

Summarising all the results:

$$\begin{aligned} \mathrm{K} &= \mathrm{CH}^T(\mathrm{HCH}^T)^{-1} \\ \mathbf{r}_c &= \mathbf{r} - \mathrm{KH}\mathbf{r} \\ \mathrm{C}_c &= \mathrm{C} - \mathrm{KHC} \\ \chi^2_c &= \chi^2 + (\mathrm{H}\mathbf{r})^T(\mathrm{HCH}^T)^{-1}(\mathrm{H}\mathbf{r}) \end{aligned} \tag{1.70}$$

Treatment of constraints with the Kalman filter

A more convenient and general way of adding penalties to the state vector is the use of the Kalman filter [12, 7, 11]. Let us first consider the case when the penalty is applied with a certain error (**soft** constraint):

$$\mathrm{H} \cdot \mathbf{r}_c + \boldsymbol{\eta} = \mathbf{0} \tag{1.71}$$

Here the random variable $\boldsymbol{\eta}$ is unbiased and has a known covariance matrix V_c:

$$\mathrm{V}_c \equiv < \boldsymbol{\eta}\boldsymbol{\eta}^T > \tag{1.72}$$

As can be seen from (1.71), the soft constraint is a measurement of the state vector with the value of measurement $\mathbf{0}$, the measurement matrix H, and the covariance matrix of error V_c. Therefore, the best estimator of the state vector after applying the soft penalty is given by the equations (1.11) of the Kalman filter:

$$\begin{aligned} \mathrm{K} &= \mathrm{CH}^T(\mathrm{V}_c + \mathrm{HCH}^T)^{-1} \\ \mathbf{r}_c &= \mathbf{r} - \mathrm{KH}\mathbf{r} \\ \mathrm{C}_c &= \mathrm{C} - \mathrm{KHC} \\ \chi^2_c &= \chi^2 + (\mathrm{H}\mathbf{r})^T(\mathrm{V}_c + \mathrm{HCH}^T)^{-1}(\mathrm{H}\mathbf{r}) \end{aligned} \tag{1.73}$$

One can see that Eqs. (1.73) of the Kalman filter coincide with Eqs. (1.70) of the Lagrange method if $\mathrm{V}_c = \mathrm{O}$. Therefore, the equations of the Kalman filter remain valid also when $\mathrm{V}_c = \mathrm{O}$.

Thus, the Kalman filter method is the general method to impose penalties, where the hard constraint is treated as a particular case of the soft constraint, being considered as a measurement with the null error.

In the case of nonlinear constraints the equation of penalty first is linearised as any other measurement, then the filtering equations (1.73) are applied.

Chapter 2

Track fit with the Kalman filter

The most typical application of the Kalman filter in high-energy physics is a fit of particle trajectories, called **tracks**. This is an interesting and non-trivial task where one can appreciate all the advantages of the method.

When a charged particle moves in a detector, its trajectory gets under the influence of several physical effects, such as the multiple scattering in the detector material, the energy losses, and a non-homogeneity of the magnetic field. These various effects make the fit of the trajectory complicated. With the LSM method it is practically impossible to take all the effects into account, especially the multiple scattering effect, as the calculations become too complicated and too slow. On the contrary, these effects can be easily treated by the Kalman filter method, thus making use of this method natural for the track fit.

2.1 The principles

The general principles of the track fit can be well understood by an examination of the simple fit of a two-dimentional straight line considered below.

Track model

To perform the fit it is necessary, first of all, to create the track model — a set of parameters that describes the particle trajectory. It is preferable[1] to choose a track model that:

- explicitly contains coordinates of some reference point on the described trajectory;
- allows any point of the trajectory to be a reference point.

Having this kind of track model, the Kalman filter has a possibility to change the reference point during the process of fitting, which makes the fit simple. The change of the reference point called **extrapolation**, **propagation** or **transport** procedure.

For the straight line track the model

$$\{x,\, y,\, t\} \tag{2.1}$$

[1]Use of this kind of model is very convenient but is not necessary — any track model can be used.

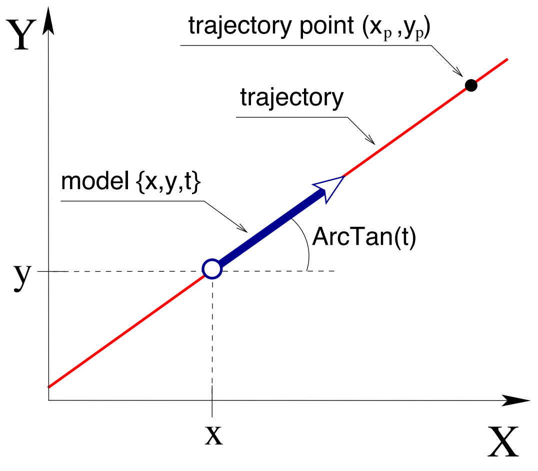


Figure 2.1: Track model.

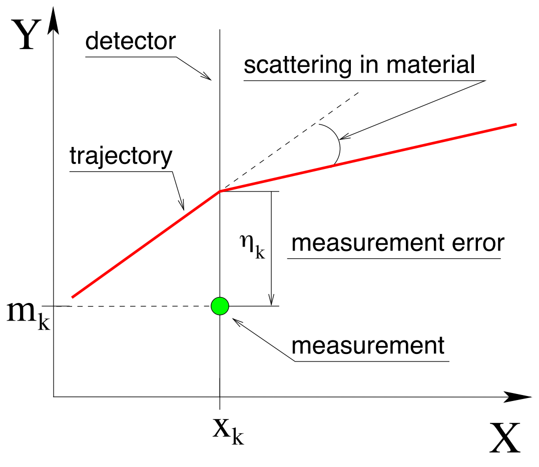


Figure 2.2: Detectors.

presented below satisfies the criteria and will be used in the considering example. In this model, the trajectory is defined as the set of points $\{x_p, \, y_p\}$ which satisfy the equation:

$$y_p = t \cdot (x_p - x) + y \tag{2.2}$$

The model is presented in Figure 2.1. The first parameter x is used as a fixed reference, while the pair $\{y, \, t\}$ defines the trajectory — y coordinate and the slope of the track at the given x reference. The state vector will be:

$$\mathbf{r} \quad = \quad \begin{pmatrix} y \\ t \end{pmatrix} \tag{2.3}$$

The parameter x is not included in the state vector since it is a reference which can be chosen arbitrarily and thus can not be fitted. In the other words: the parameter x is not a property of the trajectory but a property of the model.

Extrapolation

Once the track model is chosen it becomes necessary to describe the extrapolation procedure for the model. In the example considered the extrapolation of the state vector from the reference x to another reference $\widetilde{x}$ is defined by:

$$\begin{array}{rcl} \widetilde{x} & = & x + \Delta x \\ \widetilde{y} & = & y + t \cdot \Delta x \\ \widetilde{t} & = & t \end{array} \tag{2.4}$$

One can see that after the extrapolation the new set of parameters $\{\widetilde{x}, \, \widetilde{y}, \, \widetilde{t}\}$ describes the same trajectory as the original parameters $\{x, \, y, \, t\}$. Formulae (2.4) implies the simple extrapolation formula for the state vector (2.3):

$$\begin{array}{rcl} \widetilde{\mathbf{r}} & = & \mathrm{A}\mathbf{r} \\ \mathrm{A} & \equiv & \begin{pmatrix} 1 & \widetilde{x} - x \\ 0 & 1 \end{pmatrix} \end{array} \tag{2.5}$$

In real applications the extrapolation is usually not that trivial. When a particle has to be transported in a magnetic field, the extrapolation becomes the most complicated part of the track fit.

Measurements

Next, the measurements should be described for the chosen track model (2.3). In the example considered, the y coordinate of the track will be measured at various x positions $\{x_1, \ldots, x_n\}$, as it is shown in Figure 2.2. These measurements are one-dimensional, with the measurement values m_k and the measurement errors σ_k^2. Supposing the track is parametrised at some x, the measurement model is:

$$\begin{array}{lcl} m_k & = & \mathrm{H}_k \mathbf{r}_k^t + \eta_k \\ \mathrm{H}_k & \equiv & (1, x_k - x) \\ \sigma_k^2 & \equiv & \mathrm{cov}(\eta_k) \ = < \eta_k^2 > \end{array} \tag{2.6}$$

With the Kalman filter it is possible to simplify the measurement model (2.6) by use of the extrapolation step (1.7) of the algorithm. This is a useful advantage of the method. It allows one to separate the transport of a particle in the space from the measurement of its position, simplifying the calculations. In order to use this advantage, the track estimator will be preliminary extrapolated to the position x_k of each measurement. Under this convention the measurement model (2.6) becomes trivial:

$$\mathrm{H}_k \ \equiv \ (1, 0) \tag{2.7}$$

as the further calculations do.

Noise in the model

To imitate the multiple scattering in detectors, the track slope will be randomly changed at each plane $x = x_k$ (detector planes):

$$\begin{array}{lcl} t_k^t & = & t_{k-1}^t + \nu_k \\ q_k & \equiv & \mathrm{cov}(\nu_k) \ = < \nu_k^2 > \end{array} \tag{2.8}$$

as it is shown in Figure 2.2.

It is important to note that the presence of kink points (2.8) deforms the track. In this situation the chosen track model can not describe the real trajectory anymore.

To describe the whole trajectory — before and after the kink point — one needs to know how it changes at this point. It implies the inclusion of extra parameter ν_k to the model. As a result, the state vector will grow with the number of kink points. Moreover, the calculations will grow quadratically making the fit implementation very slow.

The problem of kink points is solved in an elegant way by use of a feature of the Kalman filter. Considering the kink point ν_k as a noise in the track model, the algorithm processes this noise at the extrapolation step; the noise covariance is simply added to the covariance matrix of the state vector. Though the value of ν_k is not fitted in this approach, the fit result (being the optimal estimator) is equivalent to the result of a complete fit with all the kink points included.

The ability to treat a noise in the model is the main advantage of the Kalman filter method. It lets the track model have as many kink points as the physical trajectory has, without the overgrow of the state vector or any complication of the algorithm.

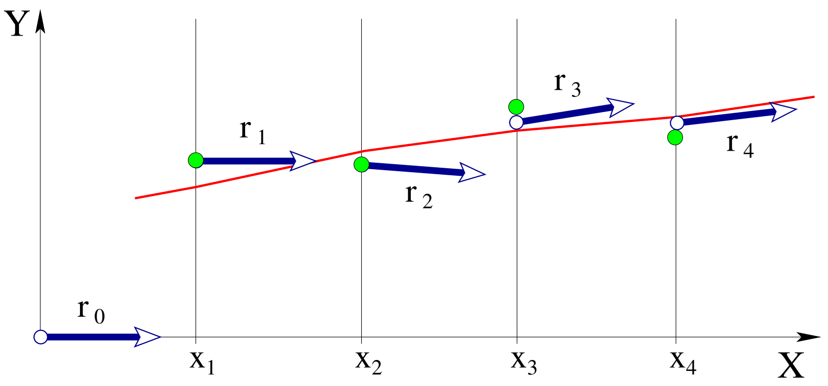


Figure 2.3: Process of the track fit. Green dots are measurements, white dots positions of the state vector.

The algorithm

When the track model, the extrapolation procedure, the measurements and the noise are described one can apply the Kalman filter algorithm (Sec. 1.2) for the track fit.

Acording to (2.4, 2.6, 2.8), for the presented example the following matrices should be substituted to the algorithms equations (1.10) and (1.11):

$$\begin{aligned} \mathrm{A}_k &= \begin{pmatrix} 1 & x_k - x_{k-1} \\ 0 & 1 \end{pmatrix} \\ \mathrm{Q}_k &= \begin{pmatrix} 0 & 0 \\ 0 & q_k \end{pmatrix} \\ \mathrm{H}_k &= \begin{pmatrix} 1 & 0 \end{pmatrix} \\ \mathrm{V}_k &= \sigma_k^2 \end{aligned} \tag{2.9}$$

In the application to the track fit, the algorithm scheme (Sec. 1.2) is the following:

First, all the measurements are ordered along the expected trajectory. In the case considered they are ordered in x direction.[2]

Second, some arbitrary initial values are set to the track estimator (**initialisation step** (1.9) of the Kalman filter). The initial errors in the covariance matrix should reflect the uncertainty of the initial parameters. Therefore it is preferable to set the initial parameters close to the expected ones in order to keep the numbers in the covariance matrix reasonably small, in order to avoid numerical problems.

After the track is initialised, the fit procedure starts. The initial estimator is extrapolated to the first measurement and the multiple scattering is added to the covariance matrix (**extrapolation step** (1.10)). Then the estimator is updated with the first measurement (**filtration step** (1.11)). After that the estimator is extrapolated to the position of the second detector and the procedure repeats for the second measurement, and so on.

The fit procedure is schematically shown in Figure 2.3. One can see that the current track estimator $\mathbf{r}_k$ is initially very far from the real track, but gradually improves while processing the measurements. The final best estimator of the track parameters is obtained after processing the last measurement.

[2]The order of the measurements is only important in the case of presence of the multiple scattering or other random contributions to the trajectory. Without these random contributions the fit result does not depend on the order of measurements.

While the actual conditions (the track model, the extrapolation formula, the measurements and the noise) differ in different experiments, the general scheme of the track fit presented above always stays the same.

Nonlinearity

When applying the above algorithm to a real experiment, two problems arise:

- First, the track extrapolation is usually nonlinear.
- Second, the noise covariance matrix and, sometimes, the measurement covariance matrix are not constant but depend on the track parameters.

To resolve these problems one estimates roughly track parameters and uses them to calculate the covariance matrices and to linearise the extrapolation formula. This estimate for track parameters is called **linearisation point** and will be marked as $\mathbf{r}^{lin}$.

The closer the linearisation point to the real track is, the better the fit result. Therefore the most precise information about the track should be put to the linearisation point.

There are two types of linearisation — implicit and explicit.

The implicit linearisation, being commonly used, sets the linearisation point to the current track estimator ($\mathbf{r}_k^{lin} = \mathbf{r}_k$). Thus the linearisation point does not explicitly appear in the equations. The disadvantage of this approach is that it can be very imprecise. Since the estimator $\mathbf{r}_k$ is defined by only k measurements, it can be far away from the real trajectory, as one can see in Figure 2.3. Moreover, the linearisation point can not be improved by using of better initial parameters, because the initial information is lost already at the second step.

The more accurate approach is the explicit linearisation, where the linearisation point $\mathbf{r}^{lin}$ is explicitly set. This approach allows one to iterate the fit, using the best estimate from the previous iteration as the linearisation point for the next iteration. The explicit linearisation makes the fit accurate and does not cost extra time or complexity within one iteration.

The equations for the Kalman filter with explicit linearisation point can be found in section 1.3.

Check of the fit quality

The quality check is performed on simulated tracks by comparing the fitted track parameters to the simulated ones. For each parameter two histograms are produced: the **residual** and the **pull**.

The residual is the difference between the fitted and the simulated value. The pull is the same difference but normalised to the expected error, taken from the covariance matrix of the fitted track. It shows how the expected error corresponds to the real error. For every parameter p:

$$\text{residual} \;\equiv\; p - p^t, \qquad \text{pull} \;\equiv\; (p - p^t)/\sigma_p \tag{2.10}$$

The residual has the dimension of the parameter, the pull is dimensionless. The dispersion of the residual is called **resolution**. Since the errors of the fitted track are defined by measurement errors and amount of material, the resolution characterises the detector rather than the fit algorithm.

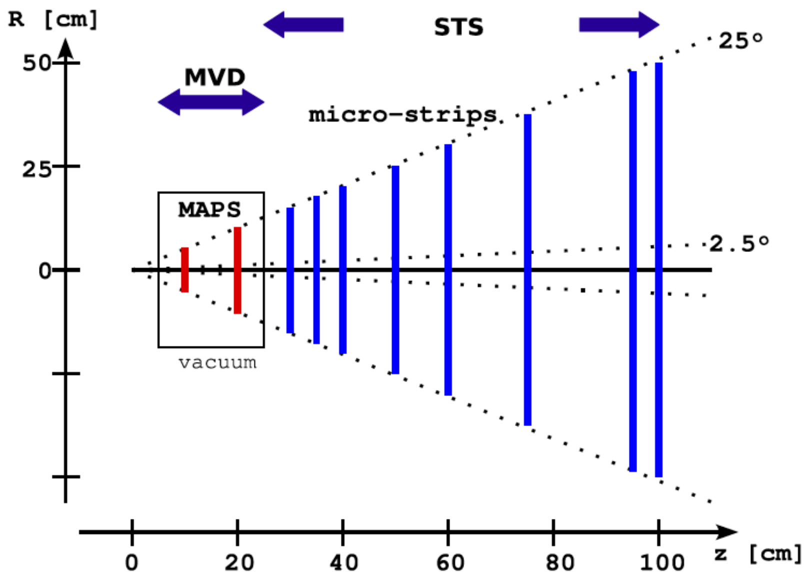


Figure 2.4: STS + MVD detector system in CBM.

When the algorithm works ideally, the mean value of the residual and the pull must be 0 while the dispersion of the pull should be 1. Deviations from these values show the quality of the fit.

2.2 Track fit in CBM

2.2.1 Overview

CBM is a fixed target experiment where charged particles are emitted from a target within a narrow cone in the beam direction. To separate the particles in the space, a strong magnetic field is introduced right after the target. The tracking system is placed inside the magnet; thus the reconstruction of events is performed in the z-region 5 cm–1 m (see Fig. 2.4). The tracking detectors are designed highly granular and to have low mass in order to track up to 1000 particles in a single event and to achieve a momentum resolution down to 1%.

The main tracking detector is the Silicon Tracking System. It consists of eight micro-strip stations. Each strip sensor is double-sided; the front and the back sensors are rotated by a stereo angle of ±7.5 and have a strip pitch of 60 μm. The silicon thickness is 300 μm.

An additional Micro-Vertex Detector (MVD) can be optionally placed in the target region. The MVD is a pixel detector based on Monolithic Active Pixel Sensors (MAPS). It consists of two stations, both have a pixel resolution $< 5\,\mu$m and a material thickness of about 50 μm.

The detector geometry is illustrated in Figure 2.4. The granularity of the detectors is so fine that the uncertainty of fitted track parameters is not defined by the measurement resolution, but is dominated by the multiple scattering in the material.

The track model used is typical for fixed-target experiments:

$$z, \ \{x, y, t_x, t_y, q/p\} \tag{2.11}$$

where x, y, z are the track position, $t_x \equiv \mathrm{d}x/\mathrm{d}z$, $t_y \equiv \mathrm{d}y/\mathrm{d}z$ are the slopes, and q/p is the particle charge divided by its momentum. The z coordinate is a reference, other parameters are to be fitted. This parametrisation is convenient for the track extrapolation, since the target and the detectors are referenced by the z coordinate.

The track fit is performed by the scheme described in Section 2. The measurement model and the description of the material are standard; details of the implementation can be found in [12].

A feature of the fit is extrapolation of tracks in a magnetic field. Due to the geometry of the magnet, the magnetic field is strongly non-homogeneous and no simple extrapolation can be applied. Therefore a special analytic formula was developed for the extrapolation, which allows one to vary the complexity of calculations, thus the CPU time, in accordance with the required precision of extrapolation. The standard Runge-Kutta extrapolation (described further) was also implemented and is used as a reference. Details of the track extrapolation are described in the next section.

Resolution					Pull				
$\delta p/p_{[\%]}$	$x_{[\mu\mathrm{m}]}$	$y_{[\mu\mathrm{m}]}$	$t_{x[\cdot 10^{-3}]}$	$t_{y[\cdot 10^{-3}]}$	q/p	x	y	t_x	t_y
0.64	27	24	1.5	1.5	1.18	1.05	1.00	1.02	1.00

Table 2.1: Resolutions and pulls (normalised residuals) of the fitted track at the event vertex.

The performance of the fit is given in Table 2.1. One can see that the momentum pull is only 18% underestimated[3] and the other pulls are close to ideal.

The developed track fit is used for the event reconstruction in all the detectors, including the muon chambers and the transition radiation detector, and for the physical analysis. It is heavily used by the on-line reconstruction, which was significantly accelerated when a special SIMDised version of the fit was developed (the SIMDised track fit is described in Section 4.2).

2.2.2 Track extrapolation in an inhomogeneous magnetic field

This Section gives an overview of the developed extrapolation method which is used in CBM. A detailed description of the method can be found in [10, 14].

The motion of a charged particle in a magnetic field obeys the differential equation, defined

[3]The underestimation of the momentum error is typical. The track momentum is the parameter which is most sensitive to all local uncertainties in the fit, because it is an invariant parameter for the whole track. During the fit, insignificant uncertainties in the momentum estimation are not neglected by multiple scattering, but are summed up along the trajectory. That is why the estimation of the momentum error is usually less accurate than the estimation of the other errors.

Specifically in the CBM fit, the reason for the underestimated momentum error is underestimation of the multiple scattering effect. For the multiple scattering the standard approximate formula [12] is used, which describes only the central part of the distribution of the scattering angle. Since the track error is a weighted sum of gaussian measurement errors and non-gaussian scattering angles, the distribution of the track error is a combination of these two different distributions. As a result, the tails from the scattering angle distribution appears in the central part of the track error distribution. This general effect becomes particularly visible in the silicon detectors, where the multiple scattering significantly contributes to the track error.

by the Lorentz force. For the CBM track model (2.11) the equation of motion becomes:

$$\frac{\mathrm{d}\mathbf{r}(z)}{\mathrm{d}z} = \begin{pmatrix} t_x \\ t_y \\ \kappa \cdot (q/p) \cdot \sqrt{1+t_x^2+t_y^2} \cdot \Big(t_x t_y \cdot B_x - \big(1+t_x^2\big) \cdot B_y + t_y \cdot B_z \Big) \\ \kappa \cdot (q/p) \cdot \sqrt{1+t_x^2+t_y^2} \cdot \Big(\big(1+t_y^2\big) \cdot B_x - t_x t_y \cdot B_y - t_x \cdot B_z \Big) \\ 0 \end{pmatrix} (z) \qquad (2.12)$$

where the dimensions are: the particle momentum $p_{[\mathrm{GeV/c}]}$, the signed charge $q_{[\mathrm{e}]}$, the magnetic field $\mathbf{B}_{[\mathrm{kG}]}$ and the coefficient $\kappa_{[(\mathrm{GeV/c})\mathrm{kG}^{-1}\mathrm{cm}^{-1}]} = 2.99792458 \cdot 10^{-4}$.

There are several methods [2, 16] of solving the equation of motion in an inhomogeneous magnetic field. The one most widely used is the Runge-Kutta extrapolation.[4] This method extrapolates a track $\mathbf{r}$ to another z position by solving the ordinary differential equation, given by (2.12):

$$\frac{\mathrm{d}\mathbf{r}(z)}{\mathrm{d}z} = \mathbf{f}(z, \mathbf{r}) \qquad (2.13)$$

with the fourth-order Runge-Kutta method.[5] The equation is solved numerically in four iterations. The derivatives of the extrapolated parameters are also calculated iteratively.

The method provides a high quality of the track extrapolation in an inhomogeneous magnetic field. It is implemented in CBM and is used as a reference.

Although the Runge-Kutta extrapolator is precise, its precision can not be increased or decreased when necessary, and its iterative structure is not optimal for the CPU computation. An additional disadvantage is that it does not provide any formula for the extrapolated track parameters, since it is a numerical method.

At the same time, the on-line tracking requires a fast and simple transport routine offering the possibility of varying the extrapolation precision in order to reach a deliberate compromise between the speed and the quality of the track fit at different stages of the reconstruction. Apart from the track fit, the track finder also needs a simple polynomial track model to group measurements into tracks.

For these purposes an exact analytic extrapolation formula was developed. In contrast to the iterative numerical methods the proposed formula performs a direct calculation of the extrapolated parameters:

$$\begin{aligned}
t_x(z_e) &= t_x(z_0) + \sum_{k=1}^{n} \sum_{i_1,\dots,i_k=x,y,z} t_{x i_1 \dots i_k}(z_0) \cdot \left(\int_{z_0}^{z_e} B_{i_1}(z_1) \dots \int_{z_0}^{z_{k-1}} B_{i_k}(z_k) \mathrm{d}z_k \dots \mathrm{d}z_1 \right) \\
t_y(z_e) &= t_y(z_0) + \sum_{k=1}^{n} \sum_{i_1,\dots,i_k=x,y,z} t_{y i_1 \dots i_k}(z_0) \cdot \left(\int_{z_0}^{z_e} B_{i_1}(z_1) \dots \int_{z_0}^{z_{k-1}} B_{i_k}(z_k) \mathrm{d}z_k \dots \mathrm{d}z_1 \right) \\
x(z_e) &= x(z_0) + \int_{z_0}^{z_e} t_x(z) \mathrm{d}z \\
y(z_e) &= y(z_0) + \int_{z_0}^{z_e} t_y(z) \mathrm{d}z
\end{aligned} \qquad (2.14)$$

whereas the coefficients $t_{x i_1 \dots i_k}(z_0)$, $t_{y i_1 \dots i_k}(z_0)$ are functions, calculated from $t_x(z_0)$, $t_y(z_0)$ at the initial position z_0, while the magnetic field is integrated along the particle trajectory.

[4]The fourth-order Runge-Kutta method is used by GEANT, for example.

[5]Fourth-order method means that the precision of the method depends on the step size to the fifth power. Runge-Kutta methods of any order exist but the fourth-order method is the optimal one with respect to CPU time consumption.

The explicit view of the coefficients is given in [10]. The extrapolation has an error in the order of $(n+1)$:

$$O\left(\frac{(\kappa B(q/p)(z_e - z_0))^{n+1}}{(n+1)!}\right) \tag{2.15}$$

The analytic formula has been derived under very general assumptions on a magnetic field.

It expands the extrapolated track parameters in a power series of the magnetic field components. Such a representation gives the possibility to evaluate only those terms which are significant with respect to a chosen precision making the transport routine fast and precise.

As the simplest case of the formula a parabolic track model for pattern recognition can be constructed [10] which takes into account local inhomogeneity of the magnetic field.

The analytic formula is also used in the final track fit procedure where high accuracy (similar or even better than provided by the fourth-order Runge-Kutta method) is crucial.

Despite the fact that the general formula is a bit cumbersome, it becomes simple in the practical implementation where a particular magnetic field makes many of the field integrals negligible.

Details of implementation

The field integrals in (2.14) are calculated along the true particle trajectory which is unknown. To solve this problem, the trajectory is first approximated using the second-order formula. Then the field values for a given z are taken at the $(x,\ y)$ position of the approximate trajectory:

$$(B_x(z),\ B_y(z),\ B_z(z)) \equiv \mathbf{B}(x_{\text{true}}(z), y_{\text{true}}(z), z) = \mathbf{B}(x_{\text{approx}}(z), y_{\text{approx}}(z), z) \tag{2.16}$$

More precise approximation is not necessary, since the CBM field is smooth enough in the XY projection at a given z.

Performance

The analytic formula has been implemented in the Kalman filter routine [12] and tested on 1000 events of central Au+Au collisions at 25 AGeV beam energy in the asymmetric inhomogeneous magnetic field. Table 2.2 contains results of tests using different extrapolators. Residuals are given as RMS of the distributions, while values of pulls are RMS of the Gaussian fits to the normalised residuals distributions.

The first line of the Table gives residuals and pulls of the track parameters obtained using the fourth-order Runge-Kutta method as the extrapolator. The fitting routine based on the Kalman filter with the fourth-order Runge-Kutta extrapolator has been well tested and provides good estimations of the track parameters. This can be seen, for instance, from the pull distributions. All the distributions have a Gaussian shape with RMS close to unity. Only the momentum estimation can still be improved and has to be further analysed in the future.

The simulation program GEANT uses the fourth-order Runge-Kutta extrapolator to transport particles through the detector volume. It is not possible to reproduce the Monte Carlo data due to measurement errors, different step size and other effects, but it is reasonable to expect that the Runge-Kutta extrapolator will provide the best (or close to the best) reconstruction results on simulated data.

Thus the existing Kalman filter fitting routine with the Runge-Kutta extrapolator can be used as a good reference for the tests.

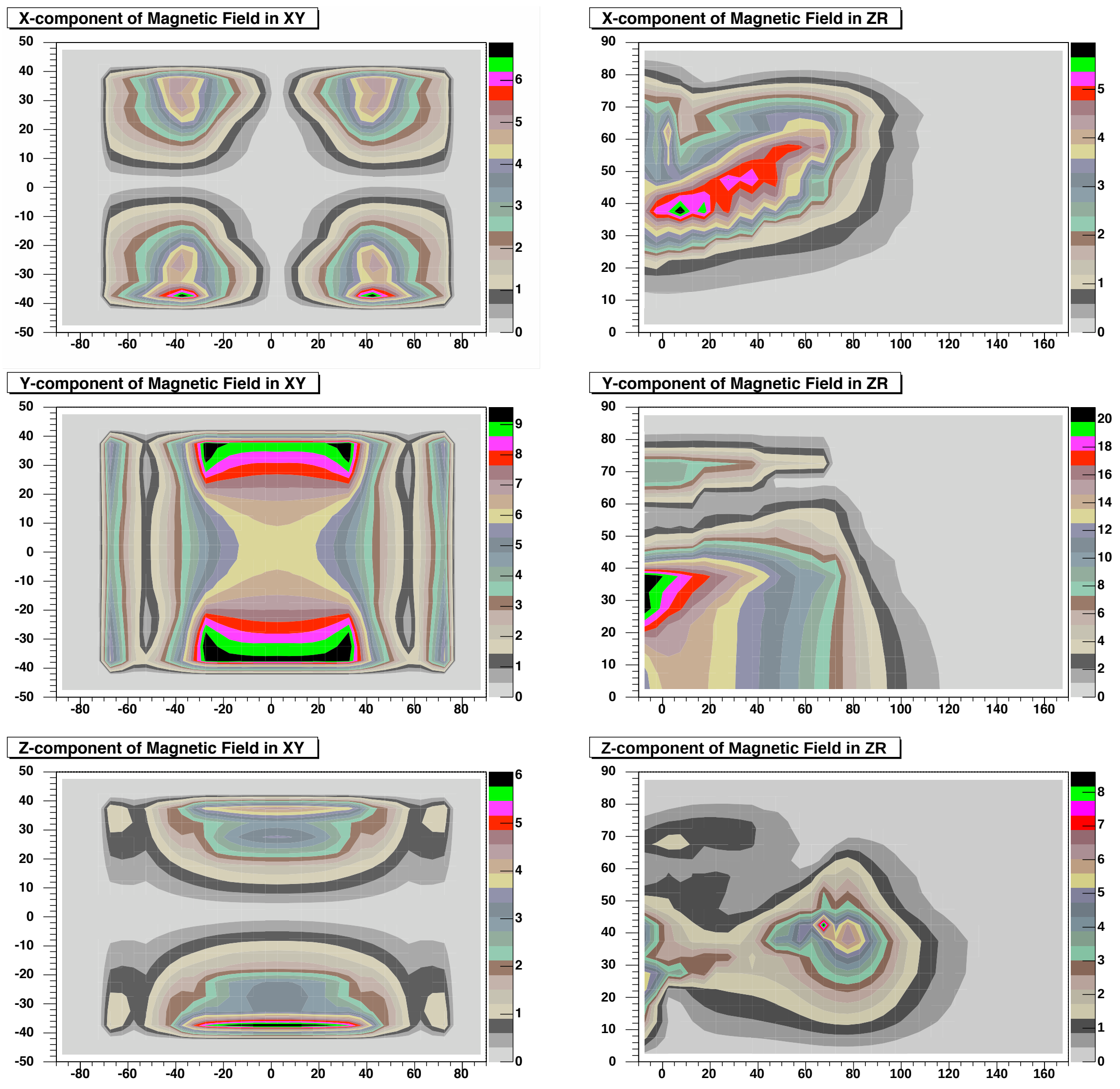


Figure 2.5: Profile of the magnetic field components.

	Residuals					Pulls				
Method	$\delta p/p$	x	y	t_x	t_y	q/p	x	y	t_x	t_y
Runge-Kutta–4	0.64	27	24	1.5	1.5	1.17	1.05	1.01	1.02	1.00
Analytic–3	0.64	27	24	1.5	1.5	1.18	1.05	1.00	1.02	1.00
Analytic–2	0.68	27	24	1.5	1.5	1.30	1.08	1.01	1.03	1.00
Analytic–1	0.94	30	25	1.5	1.5	1.90	1.37	1.03	1.10	1.02
Analytic–light	0.64	27	24	1.5	1.5	1.19	1.05	1.00	1.02	1.00
Analytic–central	2.49	38	25	1.7	1.5	3.77	2.23	1.03	1.33	1.00
LSM triplet	3.00	38	23	1.7	1.5	1.46	1.74	1.85	3.18	4.87

Table 2.2: Residuals ($\delta p/p_{[\%]}$, $(x,\ y)_{[\mu\mathrm{m}]}$, $(t_x,\ t_y)_{[\cdot 10^{-3}]}$) and normalised residuals (pulls) of the track parameters after the Kalman filter fitting routine using different extrapolators.

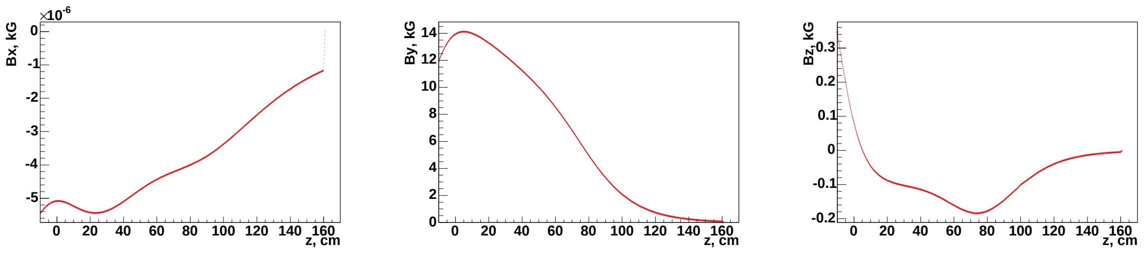


Figure 2.6: The magnetic field components at the beam line.

The analytic formula (2.14) expands the extrapolated track parameters in a power series of the magnetic field components. Having a small parameter (2.15) one can neglect the higher-order terms in the series. For instance, using terms up to the third order the extrapolator should be similar to the fourth-order Runge-Kutta method. Results (see the Table) of fitting with this Analytic–3 extrapolator demonstrate that the performance is practically identical to the Runge-Kutta method. Keeping only the terms up to the second order will give the Analytic–2 extrapolator which provides slightly worse estimations of the track parameters. Finally, the Analytic–1 extrapolator shows a further decrease of the resolution, but still provides good estimations.

c_x = 1.13e-04	**c_y = 5.64e-03**	**c_z = 3.09e-04**
c_{xx} = 3.72e-07	c_{xy} = 2.57e-06	c_{xz} = 1.25e-06
c_{yx} = 3.95e-06	**c_{yy} = 2.39e-04**	**c_{yz} = 3.04e-05**
c_{zx} = 5.21e-07	c_{zy} = 4.64e-06	c_{zz} = 3.74e-06
c_{xxx} = 5.15e-09	c_{xxy} = 2.47e-08	c_{xxz} = 8.73e-09
c_{xyx} = 3.36e-08	c_{xyy} = 3.83e-07	c_{xyz} = 6.57e-08
c_{xzx} = 5.28e-09	c_{xzy} = 3.20e-08	c_{xzz} = 1.78e-08
c_{yxx} = 4.81e-08	c_{yxy} = 5.46e-07	c_{yxz} = 9.34e-08
c_{yyx} = 7.67e-07	**c_{yyy} = 5.85e-05**	c_{yyz} = 3.75e-06
c_{yzx} = 1.30e-07	c_{yzy} = 1.35e-06	c_{yzz} = 4.03e-07
c_{zxx} = 8.33e-09	c_{zxy} = 3.35e-08	c_{zxz} = 1.47e-08
c_{zyx} = 4.76e-08	c_{zyy} = 8.63e-07	c_{zyz} = 1.33e-07
c_{zzx} = 1.59e-08	c_{zzy} = 9.91e-08	c_{zzz} = 5.49e-08

Table 2.3: Mean values of the field integrals of the Analytic–3 extrapolator.

Because in the power series terms of the same order differ significantly from each other, for an efficient realisation of the analytic formula one keeps only a few major terms within the same power. There are two possible ways to analyse the field integrals: estimating them from the formula or evaluating them numerically. Results of the numerical evaluation of the field integrals of the Analytic–3 extrapolator are given in Table 2.3. Based on the Table one can construct a light extrapolator with only 6 the most important integrals (c_x, c_y, c_z, c_{yy}, c_{yz}, c_{yyy}). Importance of these particular coefficients can be understood from relative comparison of the magnetic field components (see Figures 2.5 and 2.6). As expected, the Analytic–light extrapolator used in the Kalman filter fitting routine provides practically the same results as the full Analytic–3 extrapolator (see Table 2.2) but is much simpler.

The Analytic–central extrapolator takes the field integrals along the beam line, thus forcing the magnetic field to be constant at each detector plane and equal to the central value at z

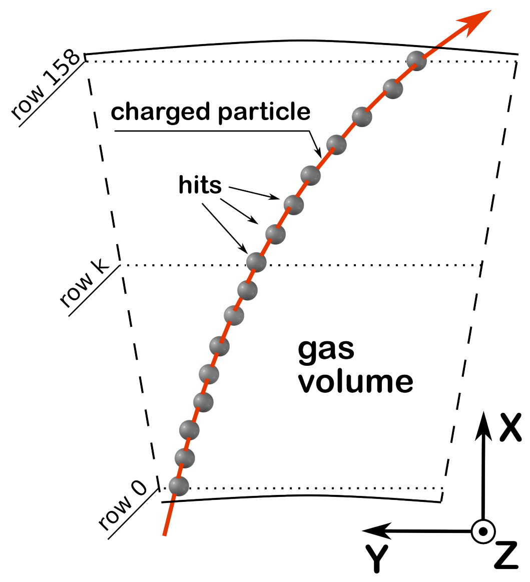


Figure 2.7: Geometry of a TPC sector.

position of a detector. With this approximation the field integrals are calculated only once for the given positions of the detectors making the algorithm extremely simple and fast. The performance of the Kalman filter fitting routine with the Analytic–central extrapolator (see Table 2.2) shows that this extrapolator is acceptable for the track finding stage. It allows reconstruction of the particle momentum with a relative error of 2.5%, while the position error is about twice as high as for the Runge-Kutta method. On the other hand, together with the decrease of the precision of the track parameter estimation the quality of the covariance matrix also degrades (see pulls part of the Table). One can see in the Table that errors of the x position and the momentum are significantly underestimated.

Note that the normalised residuals (pulls) significantly correlate with the quality of the methods as clearly seen in Table 2.2. Without pulls it is not possible to judge whether or not the maximum precision of the reconstruction has been reached.

2.3 Track fit in ALICE High Level Trigger

The main tracking detector in ALICE is the Time Projection Chamber (TPC) detector, which is a large cylindrical drift detector with two readout planes at the end-caps. The ionisation electrons drift towards the readout end-caps which are subdivided into 36 trapezoidal readout sectors. The geometry of a TPC sector is illustrated in Figure 2.7. The measurements provided by the TPC are spatial points placed at certain x positions of the TPC rows. The ALICE magnetic field is oriented along the Z axis. The field value is close to a constant in the main tracker, therefore particle trajectories are helices.

The ALICE track model is chosen with respect to the cylindrical geometry of the experiment. Tracks have local (with respect to a current TPC sector) parametrisation with 5 variable parameters at a certain x position:

$$x,\ \{y, z, \sin\phi, \lambda, q/p_t\} \tag{2.17}$$

with the current track position defined by x, y, z, the sin of the polar angle ϕ, the tan of the azimutal angle called λ, and the signed inverse transverse momentum q/p_t.

Resolution					Pull				
$\delta p_t/p_{t[\%]}$	$y_{[\mathrm{mm}]}$	$z_{[mm]}$	$\sin\phi_{[\cdot 10^{-3}]}$	$\lambda_{[\cdot 10^{-3}]}$	q/p_t	y	z	$\sin\phi$	λ
0.74(+0.21)	0.39	0.44	1.48	0.97	1.29(+0.37)	1.23	0.93	1.22	0.71

Table 2.4: Resolutions and pulls (value(±bias)) for the standard ALICE track fit.

The off-line track fit in ALICE is performed by the conventional Kalman filter. The same fit was implemented in the High Level Trigger as a first approach. Although the quality of the standard ALICE track fit is considered to be acceptable, it is not perfect. In Table 2.4 one can see that there is a significant bias in the reconstructed transverse momentum (p_t), which is the most important parameter for physics. Also the pulls shows that the standard fit procedure can be improved.

For this propose the standard fit has been investigated. As the track fit is a complicated process, it is necessary to isolate different parts of the procedure and to test each part individually. Therefore the investigation of the fit has been split into several steps. In the first step the fit task was maximally simplified:

- Use of an ideal track finder. The measurements are assigned to tracks with respect to their Monte-Carlo id's.
- Use of ideal measurements. In order to protect the fit from possible problems in the TPC cluster finder, the TPC clusters are replaced by corresponding Monte-Carlo points. The points are smeared by ideal Gaussian.
- Use of a constant 5 kG magnetic field.
- No physical effects in simulation. In particular, no multiple scattering, no energy losses and no energy loss fluctuations.
- Use of the Monte-Carlo trajectory for the initial approximation.
- Use of the Monte-Carlo trajectory as a linearisation point for the transport equation.

In the transport routine some uncertainties were found already in the first step. After the problems were resolved all the pulls became ideal, as can be seen in the first row of Table 2.5. The second step was a test of stability of the track model with respect to uncertainty of the linearisation point:

- Use of non-ideal linearisation. The fit is performed in several iterations, using the fitted trajectory as a linearisation point for the next iteration.

The second step shows biases of up to 4% in both pulls and resolutions. The problem was solved by developing a second-order correction, described in the next Subsection. The pulls before and after applying the correction are presented in Table 2.5. One can see in the Table that the correction reduces the biases significantly. The correction found is very general and may be applied for any linearised equation in any fit routine.

Having fit mathematics that works correctly, the non-constant magnetic field was introduced in the next step. Then all the physical effects were included one after another. There were

	q/p_t	y	z	$\sin\phi$	λ
Ideal linearisation	1.001(-0.000)	1.001(0.001)	1.001(-0.002)	1.001(0.000)	1.002(0.002)
Real linearisation	1.014(0.043)	1.003(0.002)	1.003(-0.006)	1.009(-0.025)	1.002(0.012)
After correction	1.005(0.002)	1.002(-0.002)	1.001(-0.008)	1.004(0.004)	1.002(0.005)

Table 2.5: Pulls (value($\pm$bias)) for non-ideal linearisation before and after applying the second-order correction to the transport equation.

some corrections applied in these steps as well. In particular, it was found that applying the energy loss before extrapolation produces a notable bias, which can be fixed by applying the energy loss in the middle of the extrapolated trajectory. For this propose the sequence "energy loss / extrapolation" was changed to "half energy loss / extrapolation / half energy loss".

	q/p_t	y	z	$\sin\phi$	λ
standard fit	1.29(+0.368)	1.23	0.93	1.22	0.71
new HLT fit	1.02(+0.001)	1.01(+0.003)	1.03(+0.013)	1.01(-0.010)	1.03(-0.019)

Table 2.6: Pulls for the new HLT track fit.

The final result is presented in Table 2.6. One can see that all the pulls are close to ideal. The resolutions are not shown in the Table because for a moment the new fit has slightly different input due to use of ideal clusters and the resolutions may not be directly compared with the standard fit.

The actual HLT fit includes:

- The explicit linearisation of equations, described in Section 2.
- A more accurate extrapolation formula.
- A more accurate energy loss formula inspired by GEANT.
- Smooth energy loss correction, described above.
- A more accurate multiple scattering formula [17].
- A second-order correction of equations, described in the next Section.

However, the investigation is not finished and the new HLT fit is at the development stage. The next steps will involve applying the real TPC clusters instead of ideal measurements (which could require some work on the calibration and on the clusterfinder) and porting the developed code to the HLT reconstruction repository (which necessarily includes speed-up and simplification of the code).

2.3.1 Second-order correction for linearised operators

The Kalman filter theory assumes that all operators in a fit are linear. But in practice the operators are nonlinear and have to be linearised before the fit mathematics can be applied. For this reason the quality of the fit always depends on the linearisation uncertainty.

Although the uncertainty of the linearisation is usually known, it is never used in the fit. The following investigation shows how fit mathematics can be improved by taking the espected linearisation uncertainty into account.

In order to shorten the text, a one-dimensional operator will be considered.

Let x be a state vector which has to be fitted (for example, a vector of track parameters).

Let $F(x)$ be a nonlinear operator which appears in the fit (for example, a track extrapolator). To perform the fit, the operator F is linearised at a certain x_0 value.

Let $L_{mc}(x)$ be an ideal linearisation of $F(x)$ at the true value x_{mc} (linearisation around the simulated Monte-Carlo track). Use of the $L_{mc}(x)$ in the fit gives an ideal result.

Let $L_0(x)$ be a non-ideal linearisation of $F(x)$ at some linearisation point x_0.

Let $\epsilon = x_{mc} - x_0$ be a linearisation error: $<\epsilon>= 0$; $<\epsilon^2>$ is known (for example, x_0 could be a fitted track with a known covariance matrix).

By definition:

$$\begin{array}{lcl} L_{mc}(x) & = & F(x_{mc}) + F'(x_{mc})(x - x_{mc}) \\ L_0(x) & = & F(x_0) + F'(x_0)(x - x_0) \end{array} \tag{2.18}$$

The expression (2.18) for the L_{mc} can be expanded as the following:

$$\begin{array}{lcl} L_{mc}(x) & = & F(x_{mc}) + F'(x_{mc})(x - x_{mc}) \\ & = & [F(x_0) + F'(x_0)\epsilon + F''(x_0)\epsilon^2/2] + [F'(x_0) + F''(x_0)\epsilon](x - x_{mc}) + O(\epsilon^3 + \epsilon^2(x - x_{mc})) \\ & = & F(x_0) + F'(x_0)\epsilon + F''(x_0)\epsilon^2/2 + F'(x_0)(x - x_{mc}) + F''(x_0)\epsilon(x - x_{mc}) + O() \\ & = & F(x_0) + F'(x_0)\epsilon + F''(x_0)\epsilon^2/2 + F'(x_0)(x - x_0 - \epsilon) + F''(x_0)\epsilon(x - x_{mc}) + O(\ldots) \\ & = & [F(x_0) + F'(x_0)(x - x_0)] + F''(x_0)\epsilon^2/2 + F''(x_0)\epsilon(x - x_{mc}) + O(\epsilon^3 + \epsilon^2(x - x_{mc})) \\ & = & L_0(x) + F''(x_0)\epsilon^2/2 + F''(x_0)\epsilon(x - x_{mc}) + O(\epsilon^3 + \epsilon^2(x - x_{mc})) \end{array} \tag{2.19}$$

From the equation (2.19) it follows:

$$< L_{mc}(x) > \ = \ < L_0(x) > + F''(x_0) < \epsilon^2 > /2 + O(\epsilon^3) \tag{2.20}$$

The equation (2.20) shows that the linearisation L_0 has the second-order statistical bias $(F''(x_0) < \epsilon^2 > /2)$. As the value of $< \epsilon^2 >$ is given, the operator L_0 can be corrected with respect to this value.

Similar calculations can be performed for the multidimensional operator L_0.

Chapter 3

Reconstruction of vertices and decayed particles

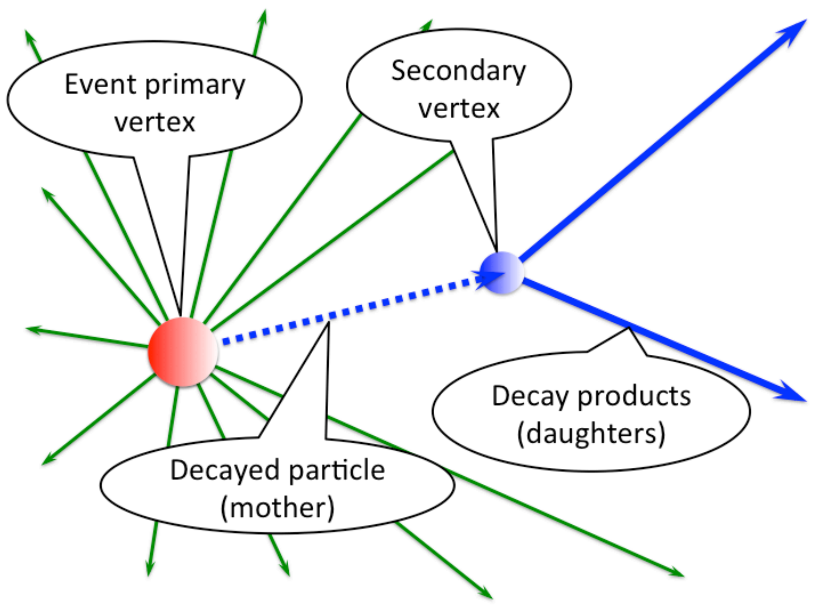


Figure 3.1: Schematic view of event vertices and a decayed particle.

3.1 Introduction

Reconstruction of a physical event does not finish with the reconstruction of the detected particle trajectories (tracks). The complete reconstruction task includes also finding of primary event vertex as well as secondary vertices and trajectories of (short-lived) decayed particles (Fig. 3.1). These tasks are performed after all the tracks are found and fitted.

The search for a common production point of the majority of particles in an event is called primary vertex search. The combinatorial part of the problem is to define the group of primary tracks. This task is usually rather simple. Then the primary vertex fit is performed with rejection of wrongly assigned tracks.

Search for secondary vertices and trajectories of decayed particles is carried out at the level of physics analysis. The combinatorial part of the task is performed by an analysis routine, where candidates for daughter particles are selected and corresponding particle hypotheses are assigned to the selected candidates. Then the secondary vertex is fitted using the daughter particles as measurements. At the same time parameters of daughter particles are corrected by fitting them to the common vertex. Parameters of the mother particle usually are not fitted directly, but are evaluated from the secondary vertex position and parameters of daughter particles after the fit.

Various algorithms have been developed for the primary and secondary vertex fit [2, 6, 7, 3, 4], the most commonly used of them are based on the Kalman filter method.

This Chapter includes a description of the traditional applications of the Kalman filter method for the primary (3.2.1) and secondary (3.2.2) vertex fit, and also the developed modifications of the standard methods (3.3, 3.4). In Section 3.5 a new algorithm for direct reconstruction of decayed particles is presented.

Results of the implementation of the developed algorithms in the CBM and the ALICE experiments can be found in Sections 3.6 and 3.7.

The algorithms use mathematical extensions of the Kalman filter method described in Section 1.4.

3.2 Standard approach for vertex reconstruction

3.2.1 Reconstruction of a primary vertex

The goal of a vertex fit is to obtain the vertex position and the associated covariance matrix using a set of track estimates and their covariance matrices. This Section is dedicated to the primary vertex fit problem which is characterised by high track multiplicity and the absence of additional physical constraints on the vertex tracks.

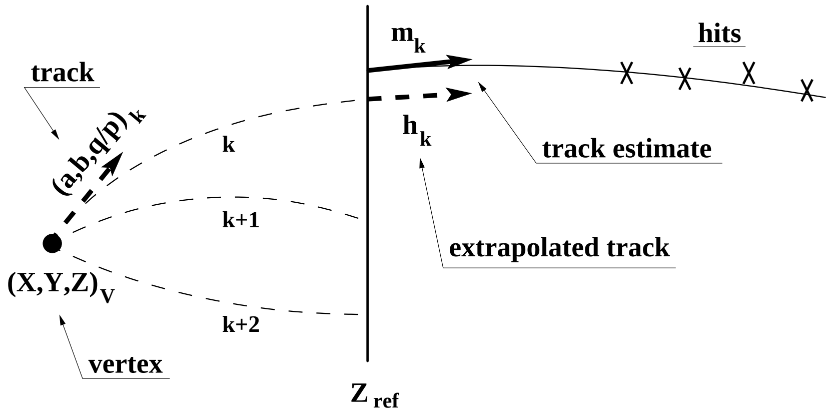


Figure 3.2: Vertex reconstruction problem.

Let us denote (see also Fig. 3.2):

$\mathbf{v} = (x_v,\ y_v,\ z_v)^T$, $C^{\mathbf{v}}$ — the vertex position and its covariance matrix;

$\mathbf{t}_k = (a_k, b_k, (q/p)_k)^T$, $C^{\mathbf{t}_k}$ — the directions and the inverse momentum of the k-th track, originating from the vertex $\mathbf{v}$, and covariance matrix for these parameters;

Measurement $\mathbf{m}_k = (x_k, y_k, t_{xk}, t_{yk}, (q/p)_k)^T$ — the k-th track estimate, parametrised at a certain z_{ref};

V_k — the covariance matrix of the k-th track estimate;

$\mathbf{h}_k(\mathbf{v}, \mathbf{t}_k)$ — parameters of the k-th track, extrapolated from z_v to z_{ref}.

Each track estimate $\mathbf{m}_k$ is considered a measurement of the corresponding track $\mathbf{t}_k$.

The measurement model which associates the track $\mathbf{t}_k$ with its estimate $\mathbf{m}_k$ is:

$$\begin{array}{rcl} \mathbf{m}_k & = & \mathbf{h}_k(\mathbf{v}^t, \mathbf{t}_k^t) + \boldsymbol{\eta}_k \\ < \boldsymbol{\eta}_k \cdot \boldsymbol{\eta}_k^T > & = & V_k \end{array} \tag{3.1}$$

The measurement model (3.1) is nonlinear in $\mathbf{v}, \mathbf{t}_k$ and, therefore, should be linearised at certain values $(\mathbf{v}^0, \mathbf{t}_k^0)$:

$$\mathbf{h}_k(\mathbf{v}, \mathbf{t}_k) \approx \mathbf{h}_k(\mathbf{v}^0, \mathbf{t}_k^0) + \left.\frac{\partial \mathbf{h}_k(\mathbf{v}, \mathbf{t}_k)}{\partial(\mathbf{v}, \mathbf{t}_k)}\right|_{\mathbf{v}^0, \mathbf{t}_k^0} (\mathbf{v} - \mathbf{v}^0, \mathbf{t}_k - \mathbf{t}_k^0) \tag{3.2}$$

The most general and accurate way to fit the vertex [2] is the direct use of the extended Kalman filter (1.16) with the state vector $\mathbf{r}$:

$$\mathbf{r}_k = (\mathbf{v}, \mathbf{t}_k) \tag{3.3}$$

and the model of measurement (3.2).

Each time the next track is added to the vertex, the parts of the state vector and the covariance matrix corresponding to the $\mathbf{t}_k$ component of the state vector are reinitialised:

$$\begin{array}{rcl} \mathbf{r}_{k-1} & \rightarrow & (\mathbf{v}_{k-1}^T, \mathbf{t}_k^0)^T \\ C_{k-1} & \rightarrow & \begin{pmatrix} C_{k-1}^v & \mathrm{O} \\ \mathrm{O} & \mathrm{I} \cdot \mathrm{inf} \end{pmatrix} \end{array} \tag{3.4}$$

The serious disadvantage of the basic method is that it requires too many calculations. Since the state vector has a dimension of $3 \times 3 = 6$ and each measurement has a dimension of 5, complicated matrix operations must be performed at each step. In particular, at each filtration step it is necessary to invert 5×5 matrices. Moreover, an additional Kalman Smoother procedure [2] is needed in order to define the track approximation $\mathbf{t}_k^0$, demanding an additional expense of time and memory.

Various fast algorithms for the vertex fit [6, 7, 3, 4] can be considered simplified versions of the basic method. However, these simplifications result in a loss of accuracy which is considered negligible.

For speeding up the calculations the following simplifications of the problem are usually applied: neglect of the magnetic field in the vertex region when tracks are considered to be straight lines [6, 7], fixation of track directions and momenta neglecting uncertainties of these parameters [3, 4], use of initial track parameters for linearisation at each iteration [3].

However, as it will be shown further (Sec. 3.3), the standard algorithm can be significantly simplified without any loss of accuracy.

3.2.2 Reconstruction of a secondary vertex

The reconstruction of secondary vertices is more complicated than the primary vertex reconstruction. Aside from the parameters of the vertex, it is necessary to obtain parameters of all vertex tracks and the complete covariance matrix. This matrix contains covariances between different tracks and covariances the between tracks and the vertex.

Keeping in mind the notations of Section 3.2.1, let a vertex (x_v, y_v, z_v) be composed of n tracks with slopes a_k, b_k and signed inverse momenta $(q/p)_k$. In contrast to the primary vertex fit (3.3), let us arrange the parameters of the vertex and the parameters of all vertex tracks in a $(3 + 3n)$-dimensional state vector $\mathbf{r}$:

$$\mathbf{r} = (\mathbf{v},\ \mathbf{t}_1,\ \dots\ \mathbf{t}_n) \tag{3.5}$$

Let us subdivide the measurement model (3.1) into coordinate and momentum parts:

$$\mathbf{h}_k(\mathbf{v}, \mathbf{t}_k) \equiv \begin{pmatrix} \mathbf{h}_k^{xy}(\mathbf{v}, \mathbf{t}_k) \\ \mathbf{h}_k^{abp}(\mathbf{v}, \mathbf{t}_k) \end{pmatrix} \tag{3.6}$$

For a new state vector (3.5) the model of the k-th measurement is:

$$\mathbf{h}_k(\mathbf{r}) = \begin{pmatrix} \mathbf{h}_k^{xy}(\mathbf{v}, \mathbf{t}_k) & \mathrm{O}\cdots\mathrm{O} & \mathrm{O} & \mathrm{O}\cdots\mathrm{O} \\ \mathrm{O} & \mathrm{O}\cdots\mathrm{O} & \mathbf{h}_k^{abp}(\mathbf{v}, \mathbf{t}_k) & \mathrm{O}\cdots\mathrm{O} \end{pmatrix} \tag{3.7}$$

Since the k-th vertex track is measured by the k-th track estimate $\mathbf{m}_k$ only; before the k-th measurement the covariance matrix C has the following form:

$$C_{k-1} = \begin{pmatrix} C_{(00)} & \cdots & C_{(0,3k)} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & 0 & \cdots & 0 \\ C_{(3k,0)} & \cdots & C_{(3k,3k)} & 0 & \cdots & 0 \\ 0 & 0 & 0 & \mathrm{inf} & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathrm{inf} \end{pmatrix} \tag{3.8}$$

Therefore, there is no need to evaluate the part of the state vector and the covariance matrix related to the remaining $k + 1, \dots, n$ vertex tracks at the k-th step. Due to this fact, the dimensionality of the state vector r_k will be considered $3+3k$ and will increase proportionally to the addition of the measurements:

$$\begin{aligned} \mathbf{r}_{k-1} &\rightarrow \left(\mathbf{r}_{k-1}^T, a_k^0, b_k^0, (q/p)_k^0\right)^T \\ C_{k-1} &\rightarrow \begin{pmatrix} C_{k-1} & \mathrm{O} \\ \mathrm{O} & \mathrm{I}\cdot\mathrm{inf} \end{pmatrix} \end{aligned} \tag{3.9}$$

In the standard approach [2] the state vector $\mathbf{r}_{k-1}$ (3.9) is filtered by the measurement $\mathbf{m}_k$ (3.7) in accordance with the formulae (1.11, 1.16) for the extended Kalman filter.

Although the standard approach for the secondary vertex fit is easy to implement, it is very time-consuming because of the large size of the state vector (3.5). On top of that, all discussed disadvantages of the primary vertex fit (Sec. 3.2.1) remain true for the secondary vertex fit.

The modified Kalman filter, constructed in Section 1.4.2, facilitates substantial simplification of the calculations for the secondary vertex fit (Sec. 3.4).

3.3 Fast reconstruction of a primary vertex

The goal of the vertex fit is to obtain the vertex position and the associated covariance matrix using a set of track estimates and their covariance matrices. This Section is dedicated to the primary vertex fit problem which is characterised by high track multiplicity and the absence of additional physical constraints on the vertex tracks.

A special feature of the presented algorithm is the extrapolation of the track estimates to the vertex linearisation point $\mathbf{r}^0$. This approach makes it possible to fit the vertex without including the track parameters into the vertex state vector and to maximally simplify the calculations. In particular, only two divisions are performed for each track, while in the standard approach two-fold inversion of a 5×5 matrix is required.

Let $\mathbf{h}_k^{xy}(\mathbf{r}, \mathbf{t}_k)$ be a coordinate part of the model of measurement $\mathbf{h}_k(\mathbf{r}, \mathbf{t}_k)$:

$$\mathbf{h}_k^{xy}(\mathbf{r}, \mathbf{t}_k) = \begin{pmatrix} x_v + a_k \cdot (z_{ref} - z_v) + O\left((z_{ref} - z_v)^2\right) \\ y_v + b_k \cdot (z_{ref} - z_v) + O\left((z_{ref} - z_v)^2\right) \end{pmatrix} \tag{3.10}$$

Here the term $O\left((z_{ref} - z_v)^2\right)$ describes the deviation of the track from a straight line in a magnetic field (see details in [14, 10]).

The linearised measurement model is:

$$\mathbf{h}_k^{xy}(\mathbf{r}, \mathbf{t}_k) \approx \begin{pmatrix} x_v + a_k^0(z_{ref} - z_v) + a_k(z_{ref} - z_v^0) + O((z_{ref} - z_v^0)^2) \\ y_v + b_k^0(z_{ref} - z_v) + b_k(z_{ref} - z_v^0) + O((z_{ref} - z_v^0)^2) \end{pmatrix} \tag{3.11}$$

Let us extrapolate all the track estimates to z_v^0 (setting $z_{ref} = z_v^0$). In this case the linearised model of measurement is:

$$\mathbf{h}_k^{xy}(\mathbf{r}, \mathbf{t}_k) \approx \begin{pmatrix} x_v + a_k^0(z_v^0 - z_v) \\ y_v + b_k^0(z_v^0 - z_v) \end{pmatrix} \tag{3.12}$$

In the linearised model (3.12) the values $a_k, b_k, (q/p)_k$ do not influence the measurement of the vertex position $\mathbf{r}_k$ with the track estimate $\mathbf{m}_k$ and, therefore, there is no need to fit these values at the k-th step of the Kalman filter.

The vertex fit is performed according to Eq. (1.11) of the Kalman filter with the state vector $\mathbf{r} = (x_v, y_v, z_v)^T$ and the measurement model H_k^{xy}:

$$\begin{aligned}
\mathrm{H}_k^{xy} &= \begin{pmatrix} 1 & 0 & -a_k^0 \\ 0 & 1 & -b_k^0 \end{pmatrix} \\
\mathrm{S}_k &= (\mathrm{V}_k^{xy} + \mathrm{H}_k^{xy}\mathrm{C}_k\mathrm{H}_k^{xyT})^{-1} \\
\mathrm{K}_k &= \mathrm{C}_k\mathrm{H}_k^{xyT}\mathrm{S}_k \\
\boldsymbol{\zeta}_k &= \mathbf{m}_k^{xy} - \mathrm{H}_k^{xy}(\mathbf{r}_{k-1} - \begin{pmatrix} 0 \\ 0 \\ z_v^0 \end{pmatrix}) \\
\mathbf{r}_k &= \mathbf{r}_{k-1} + \mathrm{K}_k\boldsymbol{\zeta}_k \\
\mathrm{C}_k &= \mathrm{C}_{k-1} - \mathrm{K}_k\mathrm{H}_k^{xy}\mathrm{C}_k \\
\chi_k^2 &= \chi_{k-1}^2 + \boldsymbol{\zeta}_k^T\mathrm{S}_k\boldsymbol{\zeta}_k
\end{aligned} \tag{3.13}$$

Here the two-dimensional measurement $\mathbf{m}_k^{xy}$ and the matrix V_k^{xy} denote the x and y components of the track estimate $\mathbf{m}_k$ and the corresponding part of its covariance matrix V_k.

The directions a^0_k, b^0_k in Eq. (3.13) are determined by fitting the track estimate $\mathbf{m}_k$ to the vertex guess $\mathbf{r}^0$ used in the linearisation. For that, one step of filtration with the vertex (x^0_v, y^0_v) as the measurement and with the null matrix of measurement errors is performed:

$$\begin{array}{rcl}
\widetilde{\mathrm{H}}_k & = & \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \\
\widetilde{K}_k & = & \mathrm{V}_k\widetilde{\mathrm{H}}_k^T(\mathrm{O} + \widetilde{\mathrm{H}}_k\mathrm{V}_k\widetilde{\mathrm{H}}_k^T)^{-1} \\
\widetilde{\boldsymbol{\zeta}}_k & = & (x^0_v, y^0_v)^T - \widetilde{\mathrm{H}}_k\mathbf{m}_k \\
\mathbf{m}^0_k & = & \mathbf{m}_k + \widetilde{\mathrm{K}}_k\widetilde{\boldsymbol{\zeta}}_k \\
a^0_k & = & \mathbf{m}^0_{k\,(2)} \\
b^0_k & = & \mathbf{m}^0_{k\,(3)}
\end{array} \tag{3.14}$$

Here $\mathbf{m}^0_k$ denotes the track estimate fitted to the $\mathbf{r}^0$ vertex from which the directions a^0_k, b^0_k are taken. Note, that there is no need to calculate the momentum $(q/p)^0_k$ of the vertex track.

Additionally, the χ^2 distance between the track estimate and the vertex guess $\mathbf{r}^0$ is calculated in order to reject wrong association of tracks to vertex:

$$\chi^2_k \;=\; \widetilde{\boldsymbol{\zeta}}_k^T(\mathrm{C}^0 + \widetilde{\mathrm{H}}_k\mathrm{V}_k\widetilde{\mathrm{H}}_k^T)^{-1}\widetilde{\boldsymbol{\zeta}}_k \tag{3.15}$$

where C^0 is the covariance matrix of the vertex guess $\mathbf{r}^0$.

The complete algorithm for the primary vertex fit is following:

1. Set first approximation of the vertex $\mathbf{r} = (x^0_v, y^0_v, z^0_v)$, C corresponding to the search area.
2. Do several iterations of the vertex fit:
   (a) Initialisation:
   - Storing of the vertex from the previous iteration: $\mathbf{r}^0 = \mathbf{r}$, $\mathrm{C}^0 = \mathrm{C}$.
   - Taking the initial vertex state vector from the previous iteration $\mathbf{r} = \mathbf{r}^0$.
   - Setting the initial covariance matrix is to $\mathrm{C} = \mathrm{I} \cdot \mathrm{inf}$.
   - Initialisation of the vertex χ^2 and number of degrees of freedom.

   (b) Filtration (repeats for every track estimate):
   - Extrapolation of the track estimate $\mathbf{m}_k$ to z^0_v.
   - Calculation of the χ^2-distance of the k-th track estimate from the vertex $\mathbf{r}^0$ (3.15).
   - Calculation of the parameters of the k-th vertex track a^0_k, b^0_k (3.14).
   - Updating of the state vector $\mathbf{r}$ with the Kalman filter formalism (3.13).

   (c) Filtration of the next track, e t.c.
3. Make the next iteration of the complete fitting routine from step 2.
4. Store $\mathbf{r}$, C.

Note that the algorithm described is also valid for the secondary vertex fit for cases where only the vertex position is to be determined.

3.4 Fast reconstruction of a secondary vertex

In this Section the secondary vertex fit with the Kalman filter is considered, where besides the parameters of the vertex it is also necessary to obtain the parameters of all vertex tracks and the complete covariance matrix, which contains the covariances between the different tracks and between the tracks and the vertex.

The conventional approach [2] to the secondary vertex fit is described in Section 3.2.2. Here a modification of the standard algorithm is presented.

To simplify the conventional algorithm, the Kalman filter mathematics have been extended for the case of correlated errors of measurements (Section 1.4.2). The use of the modified Kalman filter made it possible to exclude matrix operations completely and, as a result, to simplify the algorithm of the secondary vertex fit substantially.

3.4.1 Geometrical fit of secondary vertex

The modified Kalman filter constructed in Section 1.4.2 makes it possible to simplify the calculations.

To do so, let us split the measurement $\mathbf{m}_k$ into two parts — a momentum part $\mathbf{m}_k^{abp}$ and a coordinate part $\mathbf{m}_k^{xy}$:

$$\begin{array}{lclclcl}
\mathbf{m}_k^{xy} & = & \left(\mathbf{m}_{k\,(0)},\ \mathbf{m}_{k\,(1)}\right)^T & & \mathbf{m}_k^{abp} & = & \left(\mathbf{m}_{k\,(2)},\ \mathbf{m}_{k\,(3)},\ \mathbf{m}_{k\,(4)}\right)^T \\
\mathrm{V}_k^{xy} & = & \begin{pmatrix} \mathrm{V}_{k\,(00)} & \mathrm{V}_{k\,(01)} \\ \mathrm{V}_{k\,(10)} & \mathrm{V}_{k\,(11)} \end{pmatrix} & & \mathrm{V}_k^{abp} & = & \begin{pmatrix} \mathrm{V}_{k\,(22)} & \mathrm{V}_{k\,(23)} & \mathrm{V}_{k\,(24)} \\ \mathrm{V}_{k\,(32)} & \mathrm{V}_{k\,(33)} & \mathrm{V}_{k\,(34)} \\ \mathrm{V}_{k\,(42)} & \mathrm{V}_{k\,(43)} & \mathrm{V}_{k\,(44)} \end{pmatrix}
\end{array} \tag{3.16}$$

and consecutively add these measurements by the modified Kalman filter.

The reason for such a subdivision of $\mathbf{m}_k$ is that the momentum part $\mathbf{m}_k^{abp}$ of the track estimate measures only new parameters a_k, b_k, p_k of the state vector. These new parameters are not yet correlated with the rest of the state vector, therefore treatment of the measurement $\mathbf{m}_k^{abp}$ does not change other parameters of the state vector and can be simplified.

For this reason, the measurement $\mathbf{m}_k^{abp}$ which measures the directions and the momentum of the k-th vertex track is added first. The corresponding part of the measurement model (3.7) is:

$$\mathrm{H}_k^{abp} = \begin{pmatrix} 0\cdots 0 & 1 & 0 & 0 \\ 0\cdots 0 & 0 & 1 & 0 \\ 0\cdots 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.17}$$

Since the initial values of the parameters of the k-th track have infinite covariances (3.9), the filtration of the state vector $\mathbf{r}_{k-1}$ (3.9) by the measurement $\mathbf{m}_k^{abp}$ (3.16) is equivalent to simply copying the measurement into the state vector (the new state vector is denoted as $\tilde{\mathbf{r}}_k$):

$$\begin{array}{lcl}
\widetilde{\mathbf{r}}_k & = & \left(\mathbf{r}_{k-1}^T, \mathbf{m}_k^{abp\,T}\right)^T \\
\widetilde{\mathrm{C}}_k & = & \begin{pmatrix} \mathrm{C}_{k-1} & \mathrm{O} \\ \mathrm{O} & \mathrm{V}_k^{abp} \end{pmatrix}
\end{array} \tag{3.18}$$

The value of χ^2 does not change at this point, because the new parameters a_k, b_k, p_k do not differ from the measurement $\mathbf{m}_k^{abp}$ yet and the rest of the state vector is not changed. The number of degrees of freedom does not change either, since the introduction of three more parameters is compensated by the additional three dimensional measurement.

Now let us add the remaining coordinate part $\mathbf{m}_k^{xy}$ of the measurement $\mathbf{m}_k$ into the state vector $\widetilde{\mathbf{r}}_k$:

$$\mathrm{H}_k^{xy} = \begin{pmatrix} 1 & 0 & -a_k^0 & 0\cdots0 \\ 0 & 1 & -b_k^0 & 0\cdots0 \end{pmatrix} \tag{3.19}$$

Since the measurement $\mathbf{m}_k^{xy}$ is correlated with the measurement $\mathbf{m}_k^{abp}$, it is also correlated with the state vector $\widetilde{\mathbf{r}}_k$ (3.18). Let us denote the matrix of covariances between the measurement $\mathbf{m}_k^{xy}$ and the state vector as D_k (1.22):

$$\mathrm{D}_k = \begin{pmatrix} 0\cdots0 & \mathrm{V}_{k\,(02)} & \mathrm{V}_{k\,(03)} & \mathrm{V}_{k\,(04)} \\ 0\cdots0 & \mathrm{V}_{k\,(12)} & \mathrm{V}_{k\,(13)} & \mathrm{V}_{k\,(14)} \end{pmatrix} \tag{3.20}$$

Let us use the modified Kalman filter (1.23, 1.16). Taking into account the fact that $\mathrm{H}_k^{xy}\mathrm{D}_k^T = \mathrm{O}$ and $\mathrm{D}_k\mathrm{H}_k^{xy\,T} = \mathrm{O}$, one obtains the following equations of the geometrical fit of secondary vertex:

$$\begin{aligned}
\mathrm{S}_k &= \left(\mathrm{V}_k^{xy} + \mathrm{H}_k^{xy}\widetilde{\mathrm{C}}_k\mathrm{H}_k^{xy\,T}\right)^{-1} \\
\mathrm{K}_k &= \left(\widetilde{\mathrm{C}}_k\mathrm{H}_k^{xy\,T} - \mathrm{D}_k^T\right)\mathrm{S}_k \\
\boldsymbol{\zeta}_k &= \mathbf{m}_k^{xy} - \mathrm{H}_k^{xy}\left(\widetilde{\mathbf{r}}_k - \left(0,\ 0,\ z_v^0\right)^T\right) \\
\mathbf{r}_k &= \widetilde{\mathbf{r}}_k + \mathrm{K}_k\boldsymbol{\zeta}_k \\
\mathrm{C}_k &= \widetilde{\mathrm{C}}_k - \mathrm{K}_k\left(\mathrm{H}_k^{xy}\widetilde{\mathrm{C}}_{k-1} - \mathrm{D}_k\right) \\
\chi_k^2 &= \chi_{k-1}^2 + \boldsymbol{\zeta}_k^T\mathrm{S}_k\boldsymbol{\zeta}_k
\end{aligned} \tag{3.21}$$

3.4.2 Constrained fit of the secondary vertex

Precision of the secondary vertex parameters obtained in the geometrical vertex fit can be improved by taking into account several assumptions on the tracks associated to the vertex [2, 12, 7, 11]. These assumptions are expressed in terms of constraints on the state vector parameters.

In the secondary vertex fit every constraint is treated by the Kalman filter (Sec. 1.4.5, Eq. 1.73) as a one-dimensional measurement with null error. Since the constraints are applied after the geometrical fit, additional steps of the Kalman filter algorithm with the corresponding measurement models are implied.

Two types of constraints have been included into the vertex fit package: a topological constraint and a mass constraint. In both constraints the following values and their derivatives are used:

$$
\begin{aligned}
p_k^z &= \frac{1}{|(q/p)_k|\sqrt{1+a_k^2+b_k^2}} & \nabla p_k^z &= \left(\frac{-a_k p_k^z}{1+a_k^2+b_k^2}, \; \frac{-b_k p_k^z}{1+a_k^2+b_k^2}, \; \frac{-p_k^z}{(q/p)_k} \right) \\
p_k^x &= a_k p_k^z & \nabla p_k^x &= \left(p_k^z + a_k \frac{\partial p_k^z}{\partial a_k}, \; a_k \frac{\partial p_k^z}{\partial b_k}, \; a_k \frac{\partial p_k^z}{\partial (q/p)_k} \right) \\
p_k^y &= b_k p_k^z & \nabla p_k^y &= \left(b_k \frac{\partial p_k^z}{\partial a_k}, \; p_k^z + b_k \frac{\partial p_k^z}{\partial b_k}, \; b_k \frac{\partial p_k^z}{\partial (q/p)_k} \right) \\
E_k &= \sqrt{\frac{1}{(q/p)_k^2} + m_k^2} & \nabla E_k &= \left(0, \; 0, \; \frac{-1}{E_k (q/p)_k^3} \right)
\end{aligned}
\tag{3.22}
$$

where m_k is the mass hypothesis of the k-th particle. Here the gradient ∇ denotes the vector of the derivatives $\nabla f = (\partial f/\partial a_k, \, \partial f/\partial b_k, \, \partial f/\partial (q/p)_k)$ for a certain variable f. Parameters of the mother particle will be also used:

$$
p^x = \sum_{k=1}^{n} p_k^x \qquad p^y = \sum_{k=1}^{n} p_k^y \qquad p^z = \sum_{k=1}^{n} p_k^z \qquad E = \sum_{k=1}^{n} E_k \tag{3.23}
$$

The state vector obtained after the geometrical fit is used in both constraints as the point of linearisation $\mathbf{r}^0$.

Topological constraint

Topological constraint is used to align a mother particle with the (already) known primary vertex.[1] The mother track ends at the secondary vertex $\mathbf{v} = (x_v, y_v, z_v)$ with momentum $\mathbf{p} = (p^x, p^y, p^z)$ and has to originate from the primary vertex $\mathbf{v}_{pv} = (x_{pv}, y_{pv}, z_{pv})$.

Here only the case when the trajectory of the mother particle is a straight line (the mother particle is either not charged or its decay has occurred close to the primary vertex and the curvature of the trajectory in a magnetic field can be neglected) is considered. In this case primary and secondary vertices are connected with a straight line

$$
\mathbf{v} - \mathbf{p} \cdot t = \mathbf{v}_{pv} \tag{3.24}
$$

with an additional parameter t denoting the trajectory length of the mother particle, normalised to its momentum. The requirement (3.24) can be re-written as a set of three independent constraints:

$$
\begin{aligned}
0 &= \Delta x - p^x \cdot t \\
0 &= \Delta y - p^y \cdot t \\
0 &= \Delta z - p^z \cdot t
\end{aligned}
\tag{3.25}
$$

with notations $\Delta x = (x_v - x_{pv}), \; \Delta y = (y_v - y_{pv}), \; \Delta z = (z_v - z_{pv})$.

Since in the CBM experiment all particles have non-zero z-momentum, then also for the mother particle $p^z \neq 0$. Therefore the calculation of t can be avoided by expressing it from the third equation in (3.25):

$$
t = \Delta z / p^z \tag{3.26}
$$

[1]This is also true for the parent vertex in a decay chain.

and substituting it into the first two equations.

Finally, the topological constraint consists of two independent one-dimensional constraints:

$$\begin{aligned} 0 &= \Delta x \cdot p^z - \Delta z \cdot p^x \\ 0 &= \Delta y \cdot p^z - \Delta z \cdot p^y \end{aligned} \tag{3.27}$$

The constraint can be included directly into the Kalman filter as a set of two independent measurements with null values and null errors. As a result, the mother track will point exactly to the primary vertex.

The primary vertex errors can also be taken into account. The most elegant way to do this is to add the primary vertex parameters into the state vector [7]:

$$\mathbf{r} = (x_v, y_v, z_v,\ a_1, b_1, p_1,\ \dots\ a_n, b_n, p_n,\ x_{pv}, y_{pv}, z_{pv}) \tag{3.28}$$

with the primary vertex covariance matrix included in the extended covariance matrix of the state vector.

Two constraints (3.27) are added one by one. For the first constraint the linearised measurement matrix H^x is:

$$\mathrm{H}^x = \Bigg(\begin{array}{ccc} p^z, & 0, & -p^x, \\ \dots, \Delta x \dfrac{\partial p^z_k}{\partial a_k} - \Delta z \dfrac{\partial p^x_k}{\partial a_k}, & \Delta x \dfrac{\partial p^z_k}{\partial b_k} - \Delta z \dfrac{\partial p^x_k}{\partial b_k}, & \Delta x \dfrac{\partial p^z_k}{\partial (q/p)_k} - \Delta z \dfrac{\partial p^x_k}{\partial (q/p)_k}, \dots, \\ -p^z, & 0, & p^x \end{array} \Bigg) \tag{3.29}$$

The constructed topological constraint is used by the Kalman filter as an one-dimensional measurement with the measured value equal to 0, the null error and the linearised measurement matrix H^x:

$$0 \;=\; \Delta x \cdot p^z - \Delta z \cdot p^x + \mathrm{H}^x(\mathbf{r}^t - \mathbf{r}^0) \tag{3.30}$$

The second constraint in (3.27) is treated in the same way:

$$\mathrm{H}^y = \Bigg(\begin{array}{ccc} p^z, & 0, & -p^y, \\ \dots, \Delta y \dfrac{\partial p^z_k}{\partial a_k} - \Delta z \dfrac{\partial p^y_k}{\partial a_k}, & \Delta y \dfrac{\partial p^z_k}{\partial b_k} - \Delta z \dfrac{\partial p^y_k}{\partial b_k}, & \Delta y \dfrac{\partial p^z_k}{\partial (q/p)_k} - \Delta z \dfrac{\partial p^y_k}{\partial (q/p)_k}, \dots, \\ -p^z, & 0, & p^y \end{array} \Bigg) \tag{3.31}$$

$$0 \;=\; \Delta y \cdot p^z - \Delta z \cdot p^y + H^y(\mathbf{r}^t - \mathbf{r}^0) \tag{3.32}$$

Mass constraint

The mass constraint can be applied in the case when one or several combinations of particles in the vertex are known to originate from a narrow width mass state. Here the case of a single mass constraint is considerd, since multiple mass constraints can be treated in a similar way. Let all the tracks required by the mass constraint form the invariant mass M.

The mass constraint reads

$$M^2 = E^2 - \left(p^{x\,2} + p^{y\,2} + p^{z\,2}\right) \tag{3.33}$$

Taking the partial derivatives of M^2 one can calculate a linearised matrix H^M of the mass measurement:

$$\mathrm{H}^M = \Bigg(\begin{array}{l} 0,\ 0,\ 0,\ \cdots \\ 2E\dfrac{\partial E_k}{\partial a_k} - 2p^x\dfrac{\partial p^x_k}{\partial a_k} - 2p^y\dfrac{\partial p^y_k}{\partial a_k} - 2p^z\dfrac{\partial p^z_k}{\partial a_k}\,, \\ 2E\dfrac{\partial E_k}{\partial b_k} - 2p^x\dfrac{\partial p^x_k}{\partial b_k} - 2p^y\dfrac{\partial p^y_k}{\partial b_k} - 2p^z\dfrac{\partial p^z_k}{\partial b_k}\,, \\ 2E\dfrac{\partial E_k}{\partial (q/p)_k} - 2p^x\dfrac{\partial p^x_k}{\partial (q/p)_k} - 2p^y\dfrac{\partial p^y_k}{\partial (q/p)_k} - 2p^z\dfrac{\partial p^z_k}{\partial (q/p)_k}\,, \\ \cdots \end{array} \Bigg) \tag{3.34}$$

The mass constraint is used by Kalman filter as an ordinary one-dimensional measurement with the measured value M^2, null error and the measurement matrix H^M:

$$M^2 \;=\; E^2 - \left(p^{x\,2} + p^{y\,2} + p^{z\,2}\right) + \mathrm{H}^M(\mathbf{r}^t - \mathbf{r}^0) \tag{3.35}$$

3.4.3 Complete reconstruction scheme

The algorithm proceeds track by track and finally obtains the estimates of the vertex position and parameters of the tracks composing the vertex together with the corresponding covariance matrix. Finally, the scheme of the geometrical vertex fit algorithm is the following:

- First approximation of the state vector $\mathbf{r} = (x^0_v, y^0_v, z^0_v, \ldots, a^0_k, b^0_k, (q/p)^0_k \ldots)$.
- Several iterations of the vertex fit:
  1. Initialisation:
     - The initial vertex state vector is taken from the previous iteration $\mathbf{r} = \mathbf{r}^0$.
     - The initial covariance matrix is set to $\mathrm{C} = \mathrm{I} \cdot \inf$.
     - Initialise the vertex χ^2 and the number of degrees of freedom.
  2. Filtration (repeats for every track estimate):
     - Extrapolate the track estimate $\mathbf{m}_k$ to z^0_v.
     - Update the state vector $\mathbf{r}$ with the Kalman filter formalism:
       * Copy the track parts of the track estimate $\mathbf{m}_k$ into the state vector $\mathbf{r}$ and the covariance matrix C (3.16, 3.18).
       * Filter the state vector by the (x, y)-component of the track estimate $\mathbf{m}_k$ (3.16, 3.19, 3.20, 3.21).
  3. Filtration of the next track, etc.
- Add constraints if they are required.
- Make the next iteration of the fitting routine.

The fitting algorithm obtains the optimal state vector and its covariance matrix, from which the required output values are extracted. Besides the position of the vertex and parameters of the vertex tracks the reconstruction package also includes the procedures for the calculation of the mass and the parameters of the mother particle.

The vertex parameters are simply copied from the state vector together with the covariance matrix:

$$\begin{aligned} (x_v, y_v, z_v) &= \left(\mathbf{r}_{(0)}, \mathbf{r}_{(1)}, \mathbf{r}_{(2)}\right), \\ \mathrm{C}_v &= \begin{pmatrix} \mathrm{C}_{(00)} & \mathrm{C}_{(01)} & \mathrm{C}_{(02)} \\ \mathrm{C}_{(10)} & \mathrm{C}_{(11)} & \mathrm{C}_{(12)} \\ \mathrm{C}_{(20)} & \mathrm{C}_{(21)} & \mathrm{C}_{(22)} \end{pmatrix}. \end{aligned} \tag{3.36}$$

Parameters for the k-th vertex track are given in the conventional parametrisation (x, y, t_x, t_y, (q/p)). They are combined from the vertex position and from the momentum part of the k-th track:

$$\mathbf{T}_k = \left(\mathbf{r}_{(0)}, \mathbf{r}_{(1)}, \mathbf{r}_{(3k)}, \mathbf{r}_{(3k+1)}, \mathbf{r}_{(3k+2)}\right)^T \tag{3.37}$$

The track $\mathbf{T}_k$ is parametrised at $z_{ref} = z_v \equiv \mathbf{r}_{(2)}$.

Note that in the track parametrisation the position z_{ref} is supposed to be a fixed value while the vertex position z_v, taken as z_{ref}, is a random variable which has an error. Due to this fact the covariance matrix C_k of the track cannot simply be copied from the matrix C but needs a special correction. Let us extrapolate the track $\mathbf{T}_k$ from z_v to a certain z_{ref} and calculate the Jacobian of the extrapolation at $z_{ref} = z_v$. In accordance with (3.17, 3.19) the Jacobian J_k of the transformation $\mathbf{T}_k(\mathbf{r})$ is equal to:

$$\mathrm{J}_k = \begin{pmatrix} 1 & 0 & -\mathbf{r}_{(3k)} & 0\cdots0 & 0 & 0 & 0 & 0\cdots0 \\ 0 & 1 & -\mathbf{r}_{(3k+1)} & 0\cdots0 & 0 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 0 & 0\cdots0 & 1 & 0 & 0 & 0\cdots0 \\ 0 & 0 & 0 & 0\cdots0 & 0 & 1 & 0 & 0\cdots0 \\ 0 & 0 & 0 & 0\cdots0 & 0 & 0 & 1 & 0\cdots0 \end{pmatrix} \tag{3.38}$$

The covariance matrix C_k of the k-th track is equal to:

$$\mathrm{C}_k = \mathrm{J}_k \mathrm{C} \mathrm{J}_k^T \tag{3.39}$$

Parameters of the mother track $\mathbf{T}_m = (x, y, t_x, t_y, (q/p))^T$ are also given in the conventional CBM parametrisation. They are calculated from the momentum (p^x, p^y, p^z) of the mother particle:

$$\mathbf{T}_m = \left(\mathbf{r}_{(0)}, \mathbf{r}_{(1)}, p^x/p^z, p^y/p^z, 1/p\right)^T \tag{3.40}$$

The Jacobian J_m of the transformation $\mathbf{T}_m(\mathbf{r})$ is the following (taking into account the error of z_v):

$$J_m = \begin{pmatrix} 1 \quad 0 \quad -p^x/p^z \quad 0\cdots0 \\ 0 \quad 1 \quad -p^y/p^z \quad 0\cdots0 \\ \left(p^z\nabla p^x - p^x\nabla p^z\right)/{p^z}^2 \\ \left(p^z\nabla p^y - p^y\nabla p^z\right)/{p^z}^2 \\ -\left(p^x\nabla p^x + p^y\nabla p^y + p^z\nabla p^z\right)/p^3 \end{pmatrix} \tag{3.41}$$

The covariance matrix of the mother track is equal to:

$$\mathrm{C}_m = \mathrm{J}_m \mathrm{C} \mathrm{J}_m^T \tag{3.42}$$

In the absence of the mass constraint it is possible to calculate the mass of the mother particle and the error of the mass. Using the notations of Section 3.4.2:

$$\begin{aligned} M^2 &= E^2 - \left(p^{x\,2} + p^{y\,2} + p^{z\,2}\right) \\ \sigma^2_{M^2} &= H^M C H^{M\,T} \end{aligned} \tag{3.43}$$

It follows:

$$\begin{aligned} M &= \sqrt{M^2} \\ \sigma_M &= \frac{\sigma_{M^2}}{2M} \end{aligned} \tag{3.44}$$

Note that in the case of the secondary vertex fit with a mass constraint the fitted mass will coincide with the value of the constraint.

3.4.4 Advantages of the method

The presented algorithm of the secondary vertex fit provides the optimal estimation of the vertex position and the parameters of the vertex tracks. Its advantage in comparison with other known applications of the Kalman filter [2] is speed and simplicity of calculations.

This was possible due to:

1. Replacement of the 5-dimensional measurement (3.7) by the 2-dimensional measurement (3.16) using the modified Kalman filter. In this case the number of calculations is substantially decreased, for example, the filter executes one operation of division instead of inversion of a 5×5 matrix in the standard approach.
2. Extrapolation of the track estimates $\mathbf{m}_k$ to the point z_v^0 of the vertex linearisation. As a result the measurement model H_k^{xy} (3.19) contains only two non-trivial elements, and matrix operations with the matrix H_k^{xy} are reduced to arithmetical operations.

Avoiding matrix inversions in the implementation improves the robustness of the covariance computations against rounding errors.

In contrast to the primary vertex fit [8], in the secondary vertex fit the vertex coordinates and the track parameters are treated in common.

Both algorithms provide not only the optimal vertex position, but also optimal estimations of the track parameters, including their momenta. The algorithm of the secondary vertex fit calculates the complete covariance matrix, which includes the dependency of the vertex position on the track parameters and also between tracks. The presence of the complete covariance matrix after the geometrical fit makes it possible to refine the vertex position and the track parameters applying additional physical penalties on the tracks, such as the topological and mass constraints.

In the absence of constraints there is no need for the complete covariance matrix. In this case the primary vertex fit method described in [8] can be used to fit a secondary vertex providing the same optimal solution.

3.5 Reconstruction of decayed particles

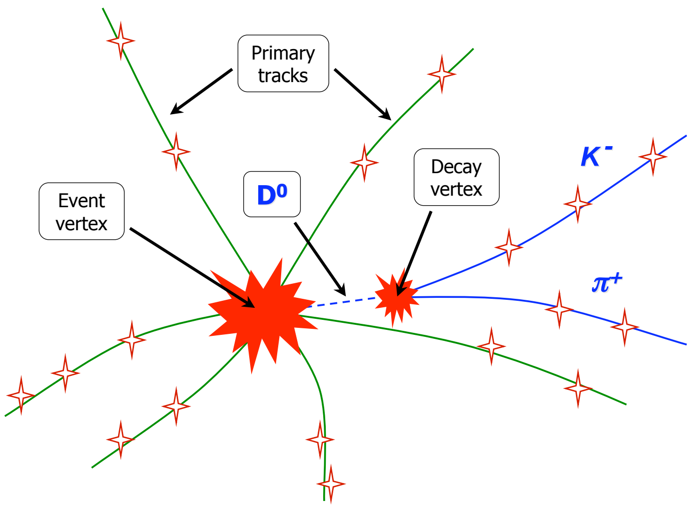


Figure 3.3: Schematic view of D^0 decay.

In modern high-energy physics experiments the most interesting physics is often extracted from the properties of (short lived) decayed particles which are not detected by the detector system and have to be reconstructed from their daughter particles (Fig. 3.3).

The existing reconstruction packages [2, 3, 5, 6, 7, 8, 9, 11] are only focused on reconstruction of production and decay vertices of decayed particles without direct estimation of particle parameters.

This Section describes a new method of reconstruction of the decayed particle parameters and associated covariance matrix using a set of daughter tracks estimates and their covariance matrices.

The developed algorithm is based on the Kalman filter method [2]. The standard Kalman filter approach has been modified in order to operate with an extended model of measurements (Sec. 1.4.1) and to filter by the best estimator (Sec. 1.4.3).

The algorithm uses a natural particle parametrisation $\mathbf{r} = (x, y, z, p_x, p_y, p_z, E, s)^T$ which makes the algorithm independent on the geometry of the detector system. After estimating the parameters of the particle, additional physical parameters which are not explicitly included into the state vector (the particle momentum P, the invariant mass M, the length of flight L in the laboratory coordinate system, and the time of life of the particle cT in its own coordinate system) are easily calculated.

The algorithm has been successfully tested on simulated data of the CBM and the ALICE experiments [12, 13].

3.5.1 Construction of the mother particle at the decay vertex

The first task in the reconstruction of the mother particle is the determination of its position, momentum and energy at the decay vertex by the modified Kalman filter (1.18), using the estimates of the daughter particles obtained after the track fit.

Let the mother particle be decayed into n daughter particles. The parameters of the mother particle reconstructed from the first k daughter particles are arranged in a 7-dimensional state vector $\mathbf{r}_k$:

$$\mathbf{r}_k \equiv (x,\, y,\, z,\, p_x,\, p_y,\, p_z,\, E)^T \tag{3.45}$$

or

$$\mathbf{r}_k \equiv \begin{pmatrix} \mathbf{v}_k \\ \underline{\mathbf{p}}_k \end{pmatrix} \tag{3.46}$$

where $\mathbf{v}_k$ — the coordinate of the particle at the decay vertex, and $\underline{\mathbf{p}}_k$ — its 4-momentum. Let us denote the covariance matrix of the state vector as C_k and the assumed position of the decay vertex, used for the linearisation of equations, as $\mathbf{v}^0$. Let us transport all daughter particles into the region of $\mathbf{v}^0$.

Let the parameters of the k-th daughter particle be denoted by $\mathbf{r}_k^d$:

$$\mathbf{r}_k^d \equiv \begin{pmatrix} \mathbf{v}_k^d \\ \underline{\mathbf{p}}_k^d \end{pmatrix} \tag{3.47}$$

and the covariance matrix be denoted by C_k^d.

For measuring the mother particle it is necessary to transport a daughter particle along its trajectory into the decay vertex. The parameters of a daughter particle at the decay vertex are denoted by $\mathbf{m}_k$:

$$\begin{aligned} \mathbf{m}_k &= \mathbf{r}_k^d + \begin{pmatrix} \mathbf{p}_k^d \\ \mathbf{p}_k^d \times B \cdot q_k \\ 0 \end{pmatrix} \cdot s_k^d + O\left(s_k^{d\,2}\right) \\ < s_k^d > &= 0 \\ \sigma^2_{s_k^d} &= \inf \end{aligned} \tag{3.48}$$

where $s_k^d = l_k^d/p_k^d$ — the unknown length of the trajectory l_k^d from the parametrisation point of the daughter particle $\mathbf{v}_k^d$ to the decay vertex $\mathbf{v}_k$, normalised by the momentum of the daughter particle; $\sigma^2_{s_k^d}$ — the error of the parameter s_k^d; B— the magnetic field value at the point $\mathbf{v}_k^d$; q_k — the charge of the daughter particle; the term $O\left(s_k^{d\,2}\right)$ describes the higher order deviations of the daughter particle's trajectory from a straight line in a magnetic field (see details in [14]).

Linearising (3.48) at $s_k^d = 0$, one obtains the measurement of the daughter particle's parameters at the decay vertex:

$$\begin{aligned} \mathbf{m}_k &= \mathbf{r}_k^d \\ \mathrm{V}_k &= C_k^d + \begin{pmatrix} \mathbf{p}^d \\ \mathbf{p}^d \times B \cdot q_k \\ 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}^d \\ \mathbf{p}^d \times B \cdot q_k \\ 0 \end{pmatrix}^T \cdot \sigma^2_{s_k^d} \end{aligned} \tag{3.49}$$

where V_k — the covariance matrix of the daughter particle parameters at the decay vertex.

The mother and daughter particles are related via the following measurement equation:

$$(\mathrm{I}, \mathrm{O})\, \mathbf{m}_k^t = (\mathrm{I}, \mathrm{O})\, \mathbf{r}_{k-1}^t \tag{3.50}$$

which is filtered by the modified Kalman filter (1.18) substituting:

$$\begin{array}{lcl} \widetilde{\mathbf{r}}_k & \equiv & \mathbf{r}_{k-1} \\ \widetilde{\mathrm{C}}_k & \equiv & \mathrm{C}_{k-1} \\ \mathbf{m}_k & \equiv & \mathbf{r}^d_k \\ \mathrm{G}_k & = & \mathrm{H}_k \;\equiv\; (\mathrm{I}, \mathrm{O}) \\ \mathbf{r}_k & \equiv & \mathbf{r}^f_{k-1} \\ \mathrm{C}_k & \equiv & \mathrm{C}^f_{k-1} \end{array} \tag{3.51}$$

Let us write the equations of filtration in detail. In order to simplify the calculations, the covariance matrix is split into the coordinate and the momentum part:

$$\mathrm{C}_{k-1} \;\equiv\; \begin{pmatrix} \mathrm{C}^v_{k-1} & \mathrm{C}^{vp\,T}_{k-1} \\ \mathrm{C}^{vp}_{k-1} & \mathrm{C}^p_{k-1} \end{pmatrix}, \qquad \mathrm{V}_k \;\equiv\; \begin{pmatrix} \mathrm{V}^v_k & \mathrm{V}^{vp\,T}_k \\ \mathrm{V}^{vp}_k & \mathrm{V}^p_k \end{pmatrix} \tag{3.52}$$

And a temporary matrix S_k is introduced:

$$\mathrm{S}_k \;=\; \left(\mathrm{C}^v_{k-1} + \mathrm{V}^v_k\right)^{-1} \tag{3.53}$$

In these notations the equations of filtration (1.18, 1.17) can be written as:

$$\begin{array}{lcl} \mathrm{K}_k & = & \begin{pmatrix} \mathrm{C}^v_{k-1} \\ \mathrm{C}^{vp}_{k-1} \end{pmatrix} \mathrm{S}_k \qquad \mathrm{K}^m_k \;=\; \begin{pmatrix} \mathrm{V}^v_k \\ \mathrm{V}^{vp}_k \end{pmatrix} \mathrm{S}_k \\ \boldsymbol{\zeta}_k & = & \mathbf{m}^v_k - \mathbf{v}_k \\ \mathbf{r}^f_{k-1} & = & \mathbf{r}_{k-1} + \mathrm{K}_k\boldsymbol{\zeta}_k \;=\; \begin{pmatrix} \mathbf{v}_{k-1} + \mathrm{C}^v_{k-1}\mathrm{S}_k\boldsymbol{\zeta}_k \\ \underline{\mathbf{p}}_{k-1} + \mathrm{C}^{vp}_{k-1}\mathrm{S}_k\boldsymbol{\zeta}_k \end{pmatrix} \\ \mathbf{m}^f_k & = & \mathbf{m}_k - \mathrm{K}^m_k\boldsymbol{\zeta}_k \;=\; \begin{pmatrix} \mathbf{v}^d_k - \mathrm{V}^v_k\mathrm{S}_k\boldsymbol{\zeta}_k \\ \underline{\mathbf{p}}^d_k - \mathrm{V}^{vp}_k\mathrm{S}_k\boldsymbol{\zeta}_k \end{pmatrix} \\ \mathrm{C}^f_{k-1} & = & \mathrm{C}_{k-1} - \mathrm{K}_k\left(\mathrm{C}^v_{k-1},\ \mathrm{C}^{vp\,T}_{k-1}\right) \;=\; \begin{pmatrix} \mathrm{C}^v_{k-1} - \mathrm{C}^v_{k-1}\mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{C}^{vp\,T}_{k-1} - \mathrm{C}^v_{k-1}\mathrm{S}_k\mathrm{C}^{vp\,T}_{k-1} \\ \mathrm{C}^{vp}_{k-1} - \mathrm{C}^{vp}_{k-1}\mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{C}^p_{k-1} - \mathrm{C}^{vp}_{k-1}\mathrm{S}_k\mathrm{C}^{vp\,T}_{k-1} \end{pmatrix} \\ \mathrm{V}^f_k & = & \mathrm{V}_k - \mathrm{K}^m_k\left(\mathrm{V}^v_k,\ \mathrm{V}^{vp\,T}_k\right) \;=\; \begin{pmatrix} \mathrm{V}^v_k - \mathrm{V}^v_k\mathrm{S}_k\mathrm{V}^v_k & \mathrm{V}^{vp\,T}_k - \mathrm{V}^v_k\mathrm{S}_k\mathrm{V}^{vp\,T}_k \\ \mathrm{V}^{vp}_k - \mathrm{V}^{vp}_k\mathrm{S}_k\mathrm{V}^v_k & \mathrm{V}^p_k - \mathrm{V}^{vp}_k\mathrm{S}_k\mathrm{V}^{vp\,T}_k \end{pmatrix} \\ \mathrm{D}^f_k & = & \mathrm{K}^m_k\left(\mathrm{C}^v_{k-1},\ \mathrm{C}^{vp\,T}_{k-1}\right) \;=\; \begin{pmatrix} \mathrm{V}^v_k\mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{V}^v_k\mathrm{S}_k\mathrm{C}^{vp\,T}_{k-1} \\ \mathrm{V}^{vp}_k\mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{V}^{vp}_k\mathrm{S}_k\mathrm{C}^{vp\,T}_{k-1} \end{pmatrix} \\ \chi^2_k & = & \chi^2_{k-1} + \boldsymbol{\zeta}^T_k\mathrm{S}_k\boldsymbol{\zeta}_k \\ \mathrm{ndf}_k & = & \mathrm{ndf}_{k-1} + 2 \end{array} \tag{3.54}$$

After the filtration the 4-momentum of the daughter particle is added to the 4-momentum of the mother particle:

$$\begin{array}{lcl} \mathbf{r}_k & = & \mathbf{r}^f_{k-1} + A_k\mathbf{m}^f_k \\ \mathrm{C}_k & = & \mathrm{C}^f_{k-1} + \mathrm{A}_k\mathrm{D}^f_k + \mathrm{D}^{f\,T}_k\mathrm{A}^T_k + \mathrm{A}_k V^f_k \mathrm{A}^T_k \end{array} \tag{3.55}$$

with the matrix A_k:

$$\mathrm{A}_k \;=\; \begin{pmatrix} \mathrm{O} & \mathrm{O} \\ \mathrm{O} & \mathrm{I} \end{pmatrix} \tag{3.56}$$

After substituting the expressions for $\mathbf{r}^f_{k-1}, \mathbf{m}^f_k, \mathrm{C}^f_{k-1}, \mathrm{V}^f_k$ and D^f_k from (3.54) into (3.55), one obtains the final equations for the update of the state vector of the mother particle by the

k-th daughter particle:

$$\begin{array}{lcl}
\mathrm{S}_k & = & \left(\mathrm{C}^v_{k-1} + \mathrm{V}^v_k\right)^{-1} \\
\mathbf{r}_k & = & \left(\begin{array}{l} \mathbf{v}_{k-1} + \mathrm{C}^v_{k-1}\mathrm{S}_k \left(\mathbf{v}^d_k - \mathbf{v}_{k-1}\right) \\ \underline{\mathbf{p}}_{k-1} + \underline{\mathbf{p}}^d_k + \left(\mathrm{C}^{vp}_{k-1} - \mathrm{V}^{vp}_k\right) \mathrm{S}_k \left(\mathbf{v}^d_k - \mathbf{v}_{k-1}\right) \end{array} \right) \\
\mathrm{C}_k & = & \left(\begin{array}{ll} \mathrm{C}^v_{k-1} - \mathrm{C}^v_{k-1}\mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{C}^{vp\,T}_{k-1} - \mathrm{C}^v_{k-1}\mathrm{S}_k \left(\mathrm{C}^{vp}_{k-1} - \mathrm{V}^{vp}_k\right)^T \\ \mathrm{C}^{vp}_{k-1} - \left(\mathrm{C}^{vp}_{k-1} - \mathrm{V}^{vp}_k\right) \mathrm{S}_k\mathrm{C}^v_{k-1} & \mathrm{C}^p_{k-1} + \mathrm{V}^p_k - \left(\mathrm{C}^{vp}_{k-1} - \mathrm{V}^{vp}_k\right) \mathrm{S}_k \left(\mathrm{C}^{vp}_{k-1} - \mathrm{V}^{vp}_k\right)^T \end{array} \right) \\
\chi^2_k & = & \chi^2_{k-1} + \left(\mathbf{v}^d_k - \mathbf{v}_{k-1}\right)^T \mathrm{S}_k \left(\mathbf{v}^d_k - \mathbf{v}_{k-1}\right) \\
\mathrm{ndf}_k & = & \mathrm{ndf}_{k-1} + 2
\end{array} \qquad (3.57)$$

For a more accurate linearisation of the measurement $\mathbf{m}_k$ (3.49), the filtration is performed twice: first, according to (3.54) an approximate momentum $\mathbf{p}^{d\,0}$ of the daughter particle is calculated:

$$\underline{\mathbf{p}}^{d\,0}_k \;=\; \underline{\mathbf{p}}^d_k - \mathrm{V}^{vp}_k \left(\mathrm{V}^v_k\right)^{-1} \left(\mathbf{v}^d_k - \mathbf{v}^0\right) \qquad (3.58)$$

then $\underline{\mathbf{p}}^{d\,0}$ is substituted into the matrix V_k (3.49), and the filtration (3.57) is carried out.

If it is necessary to select daughter tracks the χ^2 probability of the fact that the k-th particle $\mathbf{r}^d_k$ is a daughter particle is calculated according to (3.54) :

$$\chi^2_d \;=\; \left(\mathbf{v}^d_k - \mathbf{v}^0\right)^T \left(\mathrm{C}^{\mathbf{v}^0} + \mathrm{V}^v_k\right)^{-1} \left(\mathbf{v}^d_k - \mathbf{v}^0\right) \qquad (3.59)$$

with $\mathrm{C}^{\mathbf{v}^0}$ — the assumed error of the initial approximation $\mathbf{v}^0$. Then, only the particles passing the χ^2 cut are added to the mother track.

3.5.2 Measurement by a production vertex

After the particle is reconstructed at the decay vertex, a new parameter s can be added into the state vector, which is equal to the length of the particle trajectory from its production vertex to the decay vertex, normalised to the particle momentum:[2]

$$s = \frac{l}{p} \qquad (3.60)$$

with l — the length of the trajectory in the laboratory coordinate system, p — the particle momentum.

The parameter s is set initially to an approximate value s^0 (which is estimated from the distance to the given production vertex) and the corresponding element of the covariance matrix is initialised by the value $\sigma^2_s = \mathrm{inf}$:

$$\begin{array}{lcl}
\mathbf{r} & \longrightarrow & \left(\begin{array}{c} \mathbf{r} \\ s^0 \end{array} \right) \\
\mathrm{C} & \longrightarrow & \left(\begin{array}{cc} \mathrm{C} & \mathbf{0} \\ \mathbf{0} & \sigma^2_s \end{array} \right)
\end{array} \qquad (3.61)$$

Now the complete state vector is:

$$\mathbf{r} = (x, y, z, p_x, p_y, p_z, E, s)^T \qquad (3.62)$$

[2]This normalisation is convenient, because in the employed parametrisation the direction of the particle motion is assigned to the momentum vector.

After introducing the parameter s, all parameters of the particle are transported from its decay vertex to its production vertex. There the particle parameters are filtered using a given production vertex as a measurement for the Kalman filter.

Let us denote the operator for the transport of the particle parameters into the production vertex as f:

$$f(\mathbf{r}) \equiv f(\mathbf{v}, \underline{\mathbf{p}}, s) = \mathbf{r} - \begin{pmatrix} \mathbf{p} \\ \mathbf{p} \times B \cdot q \\ 0 \end{pmatrix} \cdot s + O\left(s^2\right) \tag{3.63}$$

The operator f is linearised with respect to the parameter s:

$$f(\mathbf{r}) = f(\mathbf{v}, \underline{\mathbf{p}}, s^0) - \begin{pmatrix} \mathbf{p} \\ \mathbf{p} \times B \cdot q \\ 0 \end{pmatrix} \cdot \left(s - s^0\right) \tag{3.64}$$

It is convenient to split the transport to the production vertex into two steps, corresponding to the terms in (3.64): the transport of the particle position to the fixed value $s = s^0$ and the subsequent correction of the covariance matrix taking into account the error of the parameter s.

Since the transport to the fixed value s^0 will be done only when the parameter s is either already optimal or when it is equal to s^0, the linearisation is always done at the current value $s^0 = s$.

Generally, the transport takes place in a magnetic field [14]. Here the transport in a special case will be illustrated, when the mother particle is not charged or there is no magnetic field and the transport is accomplished along a straight line. The transported particle position is:

$$\widehat{\mathbf{v}} = \mathbf{v} - s^0 \cdot \mathbf{p} \tag{3.65}$$

The other components of the state vector do not change. The Jacobian F_t of the transport along a straight line is:

$$\mathrm{F}_t = \begin{pmatrix} 1 & 0 & 0 & -s^0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -s^0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -s^0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.66}$$

and the transported particle $\widehat{\mathbf{r}}$ and its covariance matrix $\widehat{C}$ are:

$$\begin{aligned} \widehat{\mathbf{r}} &= \mathrm{F}_t \mathbf{r} \\ \widehat{C} &= \mathrm{F}_t \mathrm{C} \mathrm{F}_t^T \end{aligned} \tag{3.67}$$

Since $s^0 = s$, the state vector does not change during the correction. However, since s has an error, the covariance matrix will change. Let us consider the general case of the operator f for the charged particle in a magnetic field:

$$\begin{aligned} \widetilde{\mathbf{r}} &= \widehat{\mathbf{r}} - \begin{pmatrix} \widehat{\mathbf{p}}^0 \\ \widehat{\mathbf{p}}^0 \times B \cdot q \\ 0 \\ 0 \end{pmatrix} \cdot \left(s - s^0\right) \\ s^0 &= s \end{aligned} \tag{3.68}$$

The Jacobian F_c of the correction is:

$$\mathrm{F}_c \ = \ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & -\widehat{p}_x^0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -\widehat{p}_y^0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -\widehat{p}_z^0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -(\widehat{p}_y^0 B_z - \widehat{p}_z^0 B_y) \cdot q \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -(\widehat{p}_z^0 B_x - \widehat{p}_x^0 B_z) \cdot q \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -(\widehat{p}_x^0 B_y - \widehat{p}_y^0 B_x) \cdot q \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.69}$$

where B — the magnetic field at the production vertex, q — the particle charge.

The corrected values of the state vector and of the covariance matrix are:

$$\begin{aligned} \widetilde{\mathbf{r}} &= \widehat{\mathbf{r}} \\ \widetilde{\mathrm{C}} &= \mathrm{F}_c \widehat{C} \mathrm{F}_c^T \end{aligned} \tag{3.70}$$

After the transport of the particle into the production vertex, its position is measured by the Kalman filter using the given production vertex as a measurement. It is assumed that the optimal position $\mathbf{v}^p$ of the production vertex is already known and it does not change when fitting the particle to the vertex.[3] Since the optimal value of the vertex is given, the filtration is accomplished by the modified Kalman filter (1.37), where the production vertex $\mathbf{v}^p$ is considered as measurement with the measurement model H^p:

$$\begin{aligned} \mathbf{m}^f &\equiv \mathbf{v}^p \\ \mathrm{V}^f &\equiv \mathrm{C}^p \\ \mathrm{H}^p &= (\mathrm{I},\ \mathrm{O}) \end{aligned} \tag{3.71}$$

The measurement of the production vertex completes the reconstruction procedure.

3.5.3 The complete reconstruction scheme

Below the complete scheme of reconstruction of the particle parameters

$$\mathbf{r} = (x, y, z, p_x, p_y, p_z, E, s)^T \tag{3.72}$$

and its covariance matrix according to the daughter particles $\mathbf{r}_k^d$, $k = 1 \dots n$ is given.

Since the problem is nonlinear, the complete procedure of reconstruction is processed several times, where each iteration consists of the following steps:

1. Choice of the initial approximation $\mathbf{v}^0$, initialisation of $\chi_0^2 = 0$ and $\mathrm{ndf}_0 = -3$.
2. Transport of the k-th daughter particle $\mathbf{r}_k^d$, C_k^d into the initial vertex position $\mathbf{v}^0$, construction of the parameters $\mathbf{m}_k$ of the daughter particle at the decay vertex:

$$\begin{aligned} \mathbf{m}_k &\equiv \begin{pmatrix} \mathbf{v}_k^d \\ \underline{\mathbf{p}}_k^d \end{pmatrix} = \mathbf{r}_k^d \\ \mathrm{V}_k &\equiv \begin{pmatrix} \mathrm{V}_k^v & \mathrm{V}_k^{vp\,T} \\ \mathrm{V}_k^{vp} & \mathrm{V}_k^p \end{pmatrix} = \mathrm{C}_k^d + \begin{pmatrix} \mathbf{p}_k^d \\ \mathbf{p}_k^d \times B \cdot q_k \\ 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}_k^d \\ \mathbf{p}_k^d \times B \cdot q_k \\ 0 \end{pmatrix}^T \cdot \sigma_s^2 \end{aligned} \tag{3.73}$$

[3]Either this is the primary event vertex, or, in the case of a decay chain, the production vertex is first fitted using the particle, then the particle is fitted to the reconstructed vertex.

It is sufficient to take 10-times the distance between $\mathbf{v}^0$ and $\mathbf{v}_k^d$ divided by the momentum p_k^d as σ_s.

3. When it is necessary to select daughter tracks the χ^2 probability of the fact that the k-th particle $\mathbf{r}_k^d$ is a daughter particle is calculated:

$$\chi_d^2 = \left(\mathbf{v}_k^d - \mathbf{v}^0\right)^T \left(\mathrm{C}^{v^0} + \mathrm{V}_k^v\right)^{-1} \left(\mathbf{v}_k^d - \mathbf{v}^0\right) \tag{3.74}$$

4. Calculation of the approximated momentum $\underline{\mathbf{p}}_k^{d\,0}$ of the daughter particle:

$$\underline{\mathbf{p}}_k^{d\,0} = \underline{\mathbf{p}}_k^d - \mathrm{V}_k^{vp} \left(\mathrm{V}_k^v\right)^{-1} \left(\mathbf{v}_k^d - \mathbf{v}^0\right) \tag{3.75}$$

and refinement of the matrix V_k:

$$\mathrm{V}_k = \mathrm{C}_k^d + \begin{pmatrix} \mathbf{p}_k^{d\,0} \\ \mathbf{p}_k^{d\,0} \times B \cdot q_k \\ 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}_k^{d\,0} \\ \mathbf{p}_k^{d\,0} \times B \cdot q_k \\ 0 \end{pmatrix}^T \cdot \sigma_s^2 \tag{3.76}$$

5. Measurement of the state vector $\mathbf{r}_{k-1}$ by the daughter particle $\mathbf{m}_k$ adding the 4-momentum of the daughter particle to the 4-momentum of the mother particle:

$$\begin{aligned}
\mathrm{S}_k &= \left(\mathrm{C}_{k-1}^v + \mathrm{V}_k^v\right)^{-1} \\
\mathbf{r}_k &= \begin{pmatrix} \mathbf{v}_{k-1} + \mathrm{C}_{k-1}^v \mathrm{S}_k \left(\mathbf{v}_k^d - \mathbf{v}_{k-1}\right) \\ \underline{\mathbf{p}}_{k-1} + \underline{\mathbf{p}}_k^d + \left(\mathrm{C}_{k-1}^{vp} - \mathrm{V}_k^{vp}\right) \mathrm{S}_k \left(\mathbf{v}_k^d - \mathbf{v}_{k-1}\right) \end{pmatrix} \\
\mathrm{C}_k &= \begin{pmatrix} \mathrm{C}_{k-1}^v - \mathrm{C}_{k-1}^v \mathrm{S}_k \mathrm{C}_{k-1}^v & \mathrm{C}_{k-1}^{vp\,T} - \mathrm{C}_{k-1}^v \mathrm{S}_k \left(\mathrm{C}_{k-1}^{vp} - \mathrm{V}_k^{vp}\right)^T \\ \mathrm{C}_{k-1}^{vp} - \left(\mathrm{C}_{k-1}^{vp} - \mathrm{V}_k^{vp}\right) \mathrm{S}_k \mathrm{C}_{k-1}^v & \mathrm{C}_{k-1}^p + \mathrm{V}_k^p - \left(\mathrm{C}_{k-1}^{vp} - \mathrm{V}_k^{vp}\right) \mathrm{S}_k \left(\mathrm{C}_{k-1}^{vp} - \mathrm{V}_k^{vp}\right)^T \end{pmatrix} \\
\chi_k^2 &= \chi_{k-1}^2 + \left(\mathbf{v}_k^d - \mathbf{v}_{k-1}\right)^T \mathrm{S}_k \left(\mathbf{v}_k^d - \mathbf{v}_{k-1}\right) \\
\mathrm{ndf}_k &= \mathrm{ndf}_{k-1} + 2
\end{aligned} \tag{3.77}$$

Since at the first measurement the parameters of the mother particle have not yet been determined, the equations of filtration (3.77) are simplified and the measurement $\mathbf{m}_1$ is directly copied into the state vector $\mathbf{r}_1$:

$$\begin{aligned}
\mathbf{r}_1 &= \mathbf{m}_1 \\
\mathrm{C}_1 &= \mathrm{V}_1 \\
\chi_1^2 &= 0 \\
\mathrm{ndf}_1 &= -1
\end{aligned} \tag{3.78}$$

6. Repeating step 2 for the next daughter particle, until all the daughters are treated.

7. Improvement of the precision of the particle parameters after the fit in the case of invariant mass M of the particle is known:

$$M^2 = E^2 - \left(p_x^2 + p_y^2 + p_z^2\right) \tag{3.79}$$

In this case the parameters of the particle are measured by the conventional Kalman filter (1.11) using a one-dimensional measurement. The measurement has the value M^2, the null error, and the measurement model H_{M^2}:

$$H_{M^2} = (0, 0, 0, -p_x, -p_y, -p_z, E, 0) \tag{3.80}$$

8. Measurement of a production vertex when it is given. The constructed mother particle is transported to the production vertex and then is filtered using the production vertex as a measurement:

$$\begin{array}{lcl}
\mathbf{r} & = & \begin{pmatrix} \mathbf{v}_p \\ \widetilde{\underline{\mathbf{p}}} + \widetilde{\mathrm{C}}^{vp}\left(\widetilde{\mathrm{C}}^{v}\right)^{-1}(\mathbf{v}_p - \widetilde{\mathbf{v}}) \end{pmatrix} \\
\mathrm{C} & = & \begin{pmatrix} \mathrm{C}_p & \mathrm{C}_p\left(\widetilde{\mathrm{C}}^{v}\right)^{-1}\widetilde{\mathrm{C}}^{vpT} \\ \widetilde{\mathrm{C}}^{vp}\left(\widetilde{\mathrm{C}}^{v}\right)^{-1}\mathrm{C}_p & \widetilde{\mathrm{C}}^{p} - \widetilde{\mathrm{C}}^{vp}\left(\widetilde{\mathrm{C}}^{v}\right)^{-1}\left(\widetilde{\mathrm{C}}^{v} - \mathrm{C}_p\right)\left(\widetilde{\mathrm{C}}^{v}\right)^{-1}\widetilde{\mathrm{C}}^{vpT} \end{pmatrix} \\
\Delta\chi^2 & = & (\mathbf{v}_p - \widetilde{\mathbf{v}})^T\left(\widetilde{\mathrm{C}}^{v} - \mathrm{C}_p\right)^{-1}(\mathbf{v}_p - \widetilde{\mathbf{v}}) \\
\Delta\mathrm{ndf} & = & 2
\end{array} \tag{3.81}$$

In all the iterations, except the last one, the particle is transported back into the decay vertex by changing $-s$ to s in equations (3.69, 3.70), in order to determine the linearisation point $\mathbf{v}^0$ for the next iteration.

The reconstructed state vector and its covariance matrix contain all necessary information about the particle both at the production vertex and at the decay vertex. Therefore after the reconstruction of the parameters, the particle can be transported to the decay vertex or to the production vertex, as it is described in Section 3.5.2.

After the estimation of the parameters of the particle, additional physical parameters which are not explicitly included to the state vector can be easily calculated, such as: the particle momentum P, the invariant mass M, the length of flight L in the laboratory coordinate system, and the time of life of the particle cT in its own coordinate system:

$$\begin{array}{lcllcl}
P & = & \sqrt{p_x^2 + p_y^2 + p_z^2} & \sigma_P^2 & = & \mathrm{H}_P\mathrm{C}\mathrm{H}_P^T \\
M & = & \sqrt{E^2 - P^2} & \sigma_M^2 & = & \mathrm{H}_M\mathrm{C}\mathrm{H}_M^T \\
L & = & s \cdot P & \sigma_L^2 & = & \mathrm{H}_L\mathrm{C}\mathrm{H}_L^T \\
cT & = & s \cdot M & \sigma_{cT}^2 & = & \mathrm{H}_{cT}\mathrm{C}\mathrm{H}_{cT}^T
\end{array} \tag{3.82}$$

with

$$\begin{array}{lcllllllllll}
H_P & = & (& 0, & 0, & 0, & p_x, & p_y, & p_z, & 0, & 0 &)/P \\
H_M & = & (& 0, & 0, & 0, & -p_x, & -p_y, & -p_z, & E, & 0 &)/M \\
H_L & = & (& 0, & 0, & 0, & sp_x, & sp_y, & sp_z, & 0, & P^2 &)/P \\
H_{cT} & = & (& 0, & 0, & 0, & -sp_x, & -sp_y, & -sp_z, & sE, & M^2 &)/M
\end{array} \tag{3.83}$$

3.5.4 Advantages of the method

The chosen parametrisation of the decayed particle contains all necessary information about the particle both at the point of its production and at the point of its decay. Therefore, the developed method is suitable for both the complete reconstruction of decayed particles and the reconstruction of vertices only. In the second case, the state vector can be reduced to $\mathbf{v}$ and all operations with $\underline{\mathbf{p}}$ and s are removed. Then the algorithm is similar to the approach for the primary vertex fit (Section 3.3).

The choosen parametrisation is also physically natural and, therefore, is convenient for further physical analysis. Table 3.4 shows resolutions and pulls of D^0 physical parameters reconstructed by the algorithm. The algorithm provides, for instance, estimations of the time of life of the particle and the decay length together with the corresponding errors. Here the time of life cT is reconstructed with an accuracy of 9.8 μm, showing that the reconstructed D^0 particles are well separated from the event primary vertex.

The developed algorithm has significantly reduced the amount of calculations compared to the standard approach of vertex fitting. The state vector has a fixed size and does not grow when the number of daughter particles increases. There were no inversions of 5×5 matrices in the modified equations of filtration, thus improving the robustness of the covariance computations with respect to rounding errors.

The algorithm extrapolates the track estimates $\mathbf{r}_k^d$ to the point $\mathbf{v}^0$ of the vertex linearisation. As a result, the measurement model H_k (3.51) is trivial and does not require matrix operations. The linearisation of all measurements remains correct even when a magnetic field is present.

There is no filtration of the first daughter track. This feature reduces twice the amount of calculations in the case of two-prong decays . Furthermore, this avoids large initial values in the covariance matrix making the algorithm numerically stable.

The reconstructed mother particle can be treated as an ordinary track. For instance, the algorithm is able to transport the charged mother particle in a magnetic field. It is also possible to add measurements to the reconstructed mother particle, which is important when the decay has occurred at a considerable distance from the production vertex, and the mother particle itself has been registered by the detector system.

3.6 Implementation in the CBM experiment

The developed algorithms have been successfully implemented in the CBM experiment [12]. The algorithms are highly accurate and is suitable for further implementation at the trigger level.

3.6.1 Reconstruction of the primary vertex

Parameter	Resolution (μm)	Pull
x_v	0.67	1.08
y_v	0.64	1.11
z_v	3.62	1.10

Table 3.1: Residuals and normalised residuals (pulls) of the primary vertex parameters obtained from 10,000 simulated central Au+Au collisions at 25 AGeV.

Table 3.1 gives the precision of the primary vertex reconstruction obtained from 10,000 simulated central Au+Au collisions at 25 AGeV. The algorithm proves to be highly accurate: the residuals of the x_v and y_v positions of the primary vertex are less than 1 μm, and the z_v position is reconstructed with an accuracy better than 4 μm. The normalised residuals (pulls) are close to unity. A little increase of the pulls is probably due to the inclusion of some secondary tracks into the primary vertex fit.

The total number of reconstructed tracks used in the primary vertex fitting routine (Fig. 3.4) is quite large. In order to investigate the dependence of the vertex resolution on this number, the primary vertex fitting routine has been applied to smaller subsets of tracks (Fig. 3.5).

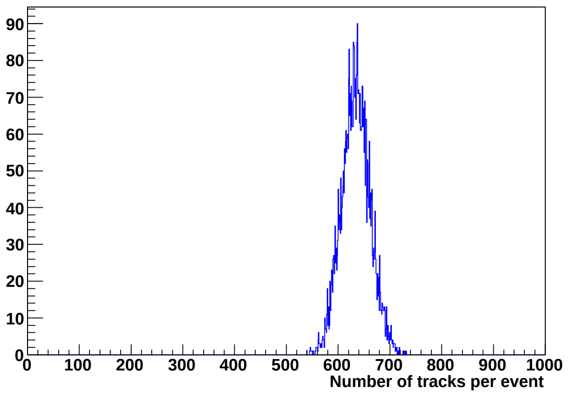


Figure 3.4: Number of reconstructed tracks per event used by the primary vertex fit algorithm for central Au+Au collisions at 25 AGeV.

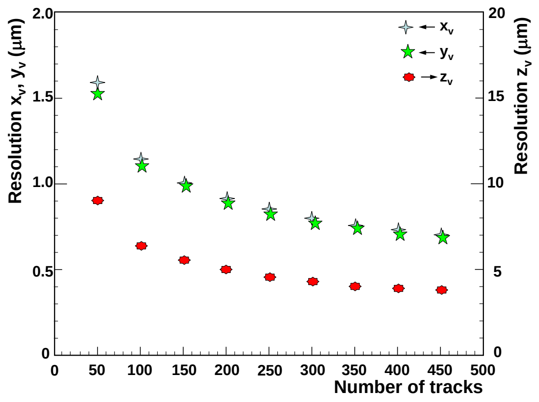
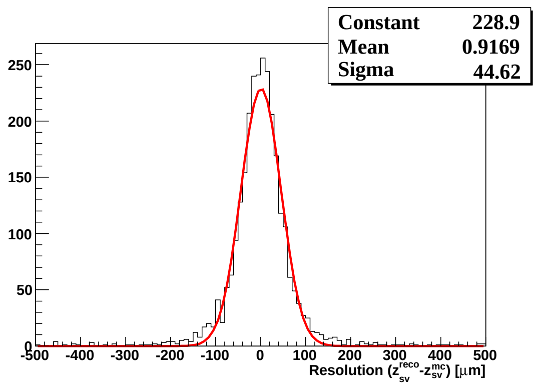


Figure 3.5: Primary vertex position resolutions versus number of tracks used in the primary vertex fit (the scale for z_v is shown on the right side).

It shows a $1/\sqrt{N}$ behaviour which allows a significant speed-up of the fitting routine in case the maximal precision of the primary vertex is not necessary. This will be especially important for on-line event selection where time consumption is crucial.

3.6.2 Reconstruction of secondary vertices



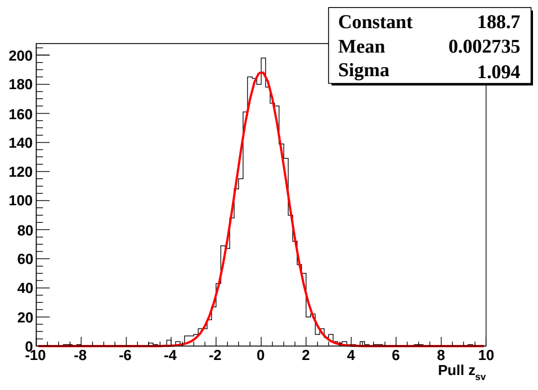


Figure 3.6: Residuals and normalised residuals (pulls) of the secondary vertex z-position obtained from 10^4 D^0 decays in the non-homogeneous magnetic field by applying the full sequence of the vertex fitting routines: the geometrical fit and the fits with mass and topological constraints.

Table 3.2 and Figure 3.6 show residuals and normalised residuals of the D^0 decay vertex parameters obtained at different stages of the vertex fitting procedure. One can see that the mass constraint mainly improves the z-position of the vertex while the topological constraint increases the resolution of the transversal parameters of the vertex.

The best resolution is reached by applying both mass and topological constraints. The longitudinal resolution is improved compared to the geometrical fit and now equals 44.6 μm. The pull of the secondary vertex z-position shows that the vertex parameters are well estimated.

Particle hypothesis have been used during the track fits to account for multiple scattering effects properly. Additionally they were used in the vertex fit procedure to apply the con-

Parameter	G	G+M	G+T	G+M+T
x_v	8.1	7.8	2.1	2.0
y_v	8.1	8.0	2.0	2.0
z_v	49.5	47.5	45.8	44.6

Table 3.2: Accuracy (in μm) of the secondary vertex parameters obtained from 10^4 D^0 decays in the inhomogeneous magnetic field by applying different sequences of the vertex fitting routines: the geometrical (G) fit and the fits with mass (M) and topological (T) constraints.

straints.

Parameter	S	S+G	S+G+M	S+G+T	S+G+M+T
p_{π^+}	0.68	0.68	0.44	0.65	0.42
p_{K^-}	0.69	0.68	0.54	0.66	0.52

Table 3.3: Relative momentum resolution $\delta p/p$ (in %) of the secondary tracks obtained from 10^4 D^0 decays in the inhomogeneous magnetic field by applying different sequences of the track and vertex fitting routines: the standalone (S) track fit, the geometrical (G) vertex fit and the vertex fits with mass (M) and topological (T) constraints.

Table 3.3 shows the relative momentum resolutions of the secondary tracks at different stages of the event reconstruction. The mass constrained secondary vertex fit gives the most significant improvement of the momentum resolution for both particles. The difference in the momentum resolution behavior of π^+ and K^- is probably due to their masses.

3.6.3 Reconstruction of decayed particles

For these studies central Au+Au collisions at 25 AGeV have been simulated. In the simulations the main tracking detector of 7 silicon pixel stations positioned at 10, 20, 30, 40, 60, 80 and 100 cm from the target was used. The first two stations had a thickness of 150 μm, while the other stations had a thickness of 400 μm. All detectors have idealised response (no fake hits, efficiency losses, pile-up etc.). The non-homogeneous active magnetic field has been used to trace particles through the detector.

For tests of the developed algorithm D^0 mesons have been reconstructed. They are generated at the event primary vertex and then decay into π^+ and K^- particles. Since a D^0 meson has a very short lifetime, it is not detected by the detector system, while its daughter particles are well within the detector acceptance.

The ideal[4] track finder has been used to group hits into tracks. The track fitting routine realises the Kalman filter in its conventional approach. The default π particle hypothesis has been used for all tracks. For the π^+ and K^- daughter particles the correct particle hypothesis have been used during the track fit in order to properly account for multiple scattering effects, and in the reconstruction procedure to calculate the π^+ and K^- energy.

In the tests the algorithm first reconstructs the event production vertex using all reconstructed tracks, then D^0 mesons are reconstructed from their two daughter particles π^+ and K^- using the event primary vertex as the production vertex.

[4]It uses Monte-Carlo information.

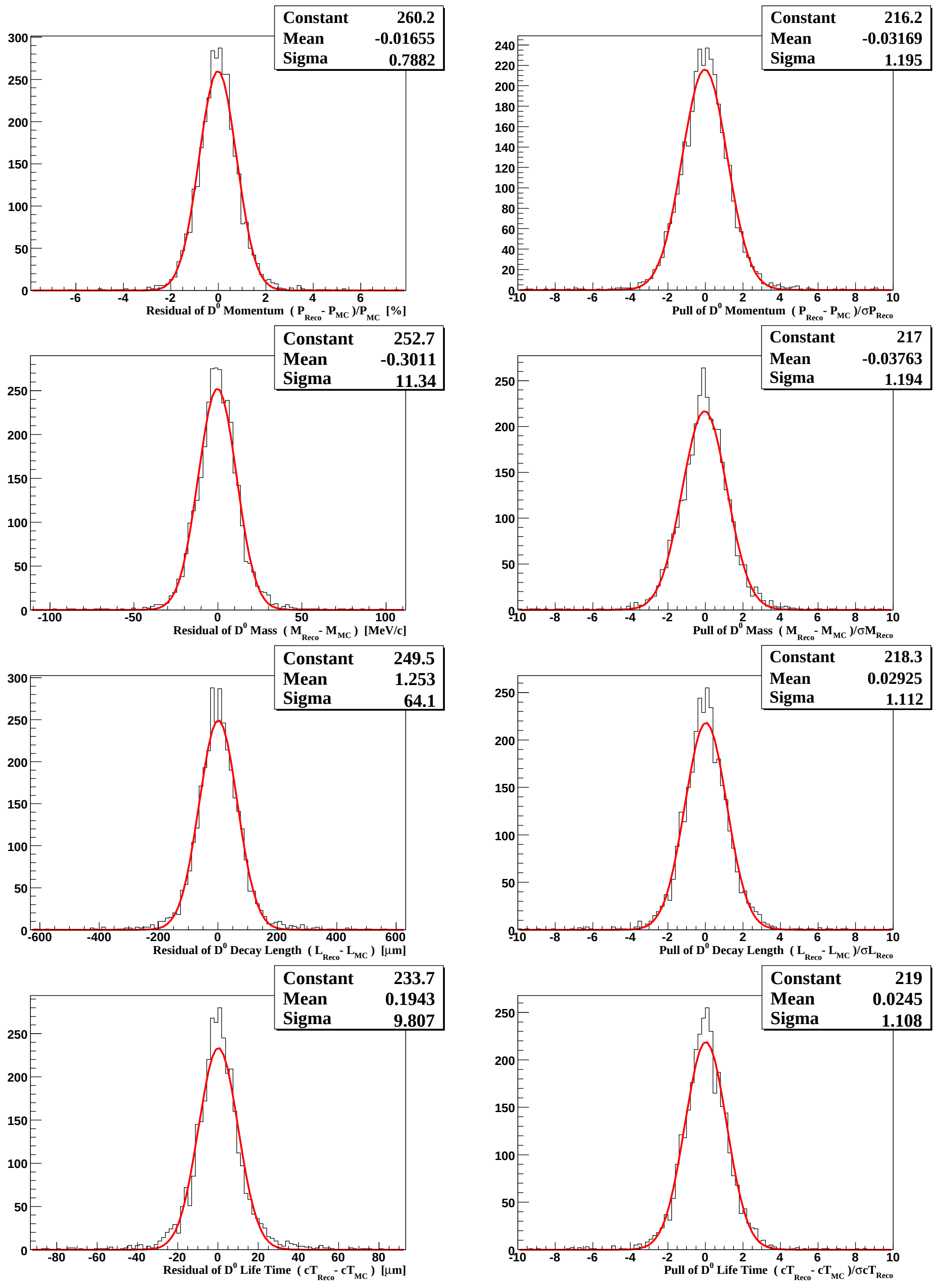


Figure 3.7: Residuals and normalised residuals (pulls) of the D^0 physical parameters P, M, L and cT.

	Production Vertex$_{[\mu m]}$			Decay Vertex$_{[\mu m]}$			Physical Parameters			
	x	y	z	x	y	z	$P_{[\%]}$	$M_{[MeV/c]}$	$L_{[\mu m]}$	$cT_{[\mu m]}$
Accuracy	0.81	0.73	5.50	2.64	2.64	63.88	0.79	11.34	64.10	9.81
Pull	1.14	1.10	1.11	1.13	1.13	1.10	1.20	1.19	1.11	1.11

Table 3.4: Resolutions and pulls of the decayed particle parameters obtained from 10^4 D^0 decays in central Au+Au collisions at 25 AGeV.

Table 3.4 shows, that the algorithm provides a very high accuracy for the event vertex: the resolutions of the x and y positions of the D^0 production vertex are less than 1 μm, and the z position is reconstructed with an accuracy 5.5 μm. The resolution of the D^0 decay vertex is 2.64 μm for x and y, and 63.88 μm for z. The normalised residuals (pulls) are close to unity, thus showing that all parameters are well estimated.

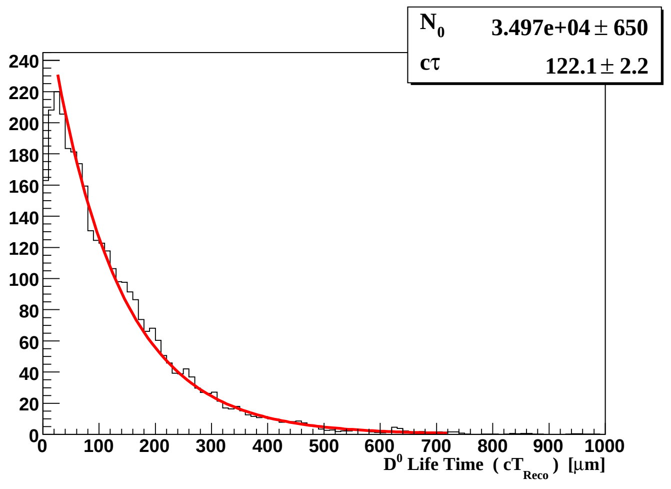


Figure 3.8: Distribution of the D^0 life time.

In addition, Figure 3.7 gives distributions of residuals and normalised residuals (pulls) of the D^0 physical parameters. The RMS of the Gaussian fits to the residual and normalised residual distributions are also given. A measure of the reliability of the fit is the pull distribution of the fitted parameters. All pulls are centered at zero indicating that there is no systematic shift in the reconstructed track parameter values. The distributions are well fitted using Gaussian functions with small tails caused by the various non-Gaussian contributions to the fit.

Figure 3.8 gives the distribution of the D^0 life time with the fitted mean life (122.1 ± 2.2) μm, which is close to the D^0 mean life $c\tau_{D^0} = 122.9$ μm [15] used in the simulations.

3.7 Implementation in the ALICE experiment

The algorithm for reconstruction of decayed particles, described in Section 3.5, has been implemented in the ALICE experiment. Here the method is used for off-line and on-line physics analysis and for the primary vertex reconstruction in the High Level Trigger.

Since the method uses physical particle parametrisation, there were no special mathematics developed for ALICE: the exactly same mathematical core is implemented in both the CBM

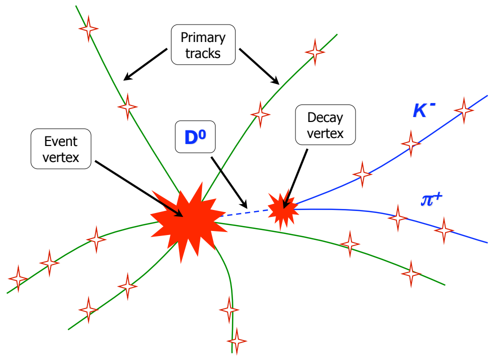


Figure 3.9: Schematic view of D^0 decay in ALICE.

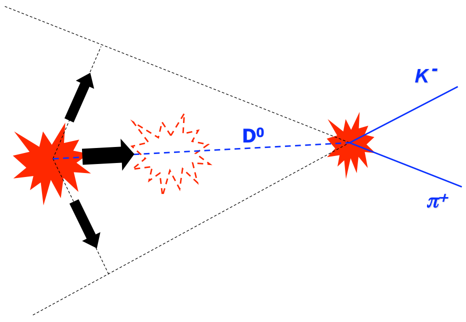


Figure 3.10: Systematic bias of the event primary vertex due to inclusion of D^0 daughters to the primary vertex fit.

and the ALICE reconstructions. Specifically, the original CBM code was simply copied to the ALICE software repository with renaming classes with respect to the ALICE naming convention. A simplified extrapolation method which is specific for the ALICE magnetic field has been implemented afterwards. In addition, some interfaces to ALICE classes have been developed.

10^6 events	Production Vertex$_{[\mu m]}$			Decay Vertex$_{[\mu m]}$			Physical Parameters			
	x	y	z	x	y	z	$P_{[\%]}$	$M_{[MeV/c]}$	$L_{[\mu m]}$	$cT_{[\mu m]}$
Accuracy	49.1	48.8	67.4	75.4	75.0	88.6	0.75	9.9	165.5	100.4
Pull	0.95	0.95	0.98	0.92	0.92	0.97	0.92	0.94	0.93	0.93

Table 3.5: Resolutions and pulls of the decayed particle parameters obtained from D^0 decays in proton-proton collisions at 14 TeV.

The fit quality was tested on D^0 decay using simulated ALICE proton-proton events. Initially, all the residuals and pulls were accurate except of the reconstructed decay length which appeared to be systematically smaller than the generated value.

An investigation showed that the event primary vertex is systematically biased in the direction of D^0 decay vertex. This bias was caused by inclusion of D^0 daughters to the primary vertex fit, as it is shown in Figure 3.10. Thus the problem is not specific for the D^0 analysis but is more general.[5]

To solve the problem it is necessary to exclude daughter tracks from the primary vertex when creating a decayed particle. It can be easily done without vertex re-fit by use of the measurement subtraction procedure described in Section 1.4.4.

The final pulls and resolutions for the D^0 decay are summarised in Table 3.5. The residual and pull distributions for the most important parameter – the mass – are presented in Figure 3.11. One can see that the pull is close to unity showing no bias.

Additionally, simple analysis macro was provided to show the package functionality. In this

[5]Interesting that the primary vertex bias was never seen in the CBM D^0 analysis because of a much higher number of tracks contributing to the CBM vertex.

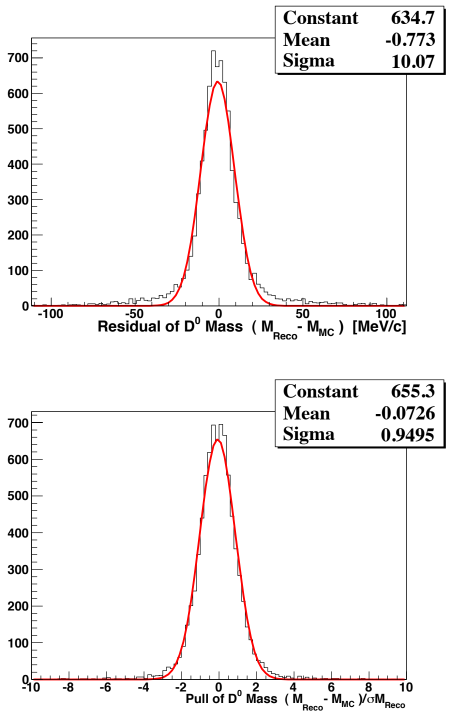


Figure 3.11: Residual and pull of the D^0 mass.

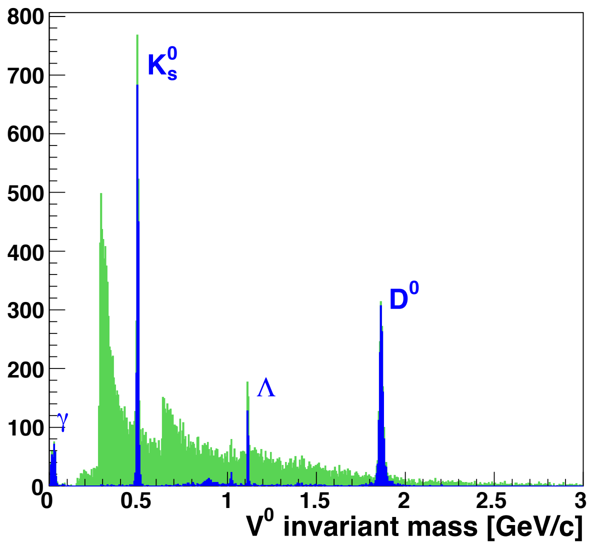


Figure 3.12: Test of the decayed particle reconstruction on simulated ALICE p-p events.

test for each pair of positive and negative particles a mother particle is created and constrained to the primary vertex. True Monte-Carlo PID is set to the daughter particles, but no other Monte-Carlo information is used. In the case when the created mother particle passes a χ^2 cut, its mass is put into a histogram. The resulting invariant mass distribution is shown in Figure 3.12. One can see narrow mass peaks which correspond to various V^0 decays and low background from false combinations of daughter particles.

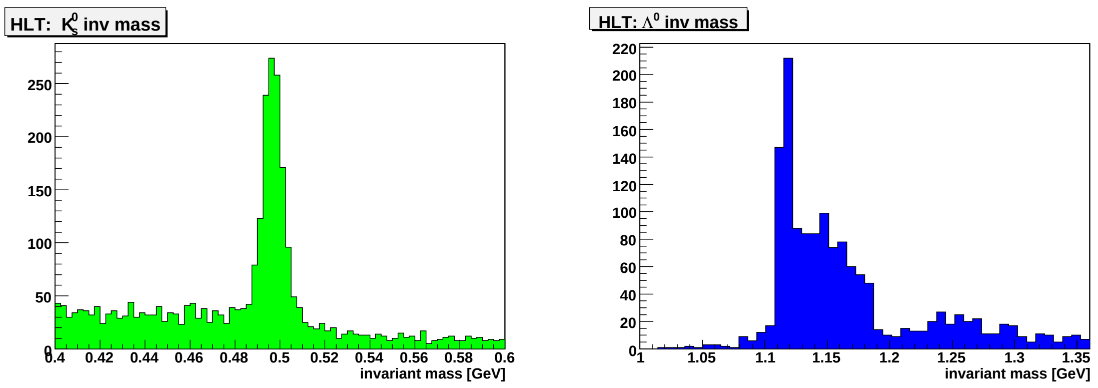


Figure 3.13: HLT K^0_s and Λ^0 finders. Real data, run 00010480 (2009).

A real analysis of the decayed particles is implemented in the High Level Trigger. Some of the V^0 decays, such as K^0_s-, Λ^0-, and γ-decay are monitored on-line in order to check the quality of the data and the consistency of the HLT reconstruction chain (see Fig. 3.13).

In addition, the Armenteros-Podolanski plot [19] is displayed (see Fig. 3.14). The plot is a

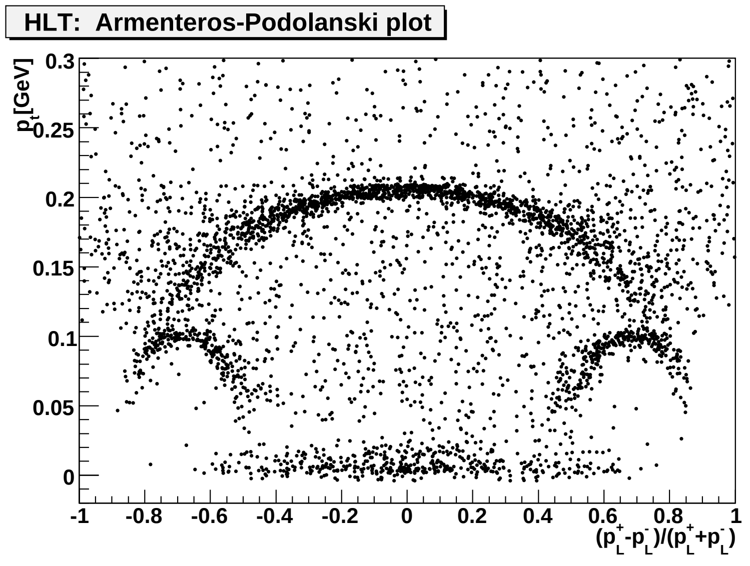


Figure 3.14: HLT Armenteros-Podolanski plot. Real data, run 00010480 (2009).

convenient way of identifying V^0 decays without taking any assumption on the masses of the decay products. It is a two dimensional plot of transverse momentum p_t of the oppositely charged decay products with respect to the V_0 momentum versus the longitudinal momentum asymmetry $\alpha = (p_l^+ - p_l^-)/(p_l^+ + p_l^-)$. The largest ellipse in Figure 3.14 is K_S^0 decay. It is centred about $\alpha = 0.0$ as the decay particles are two pions which have the same mass and therefore carry similar momenta. The centres of the Λ and $\bar{\Lambda}$ bands are shifted to $\alpha = +0.7$ and $\alpha = -0.7$ respectively due to the asymmetry between the masses of the decay products.

The both Figures 3.13 and 3.14 show the first ALICE physics obtained by the HLT in the first collision runs in late 2009.

3.7.1 Reconstruction of primary vertex in the High Level Trigger

It has been pointed out in Section 3.5 that the method of reconstruction of decayed particles can be applied for the vertex reconstruction as well. In this approach, a phantom mother particle for all the primary tracks is reconstructed. The position of the mother particle at its decay point gives the desired vertex position while other parameters of the reconstructed particle are meaningless.

Despite mathematical correctness, the method is not optimal in the sense of computing time, as it evaluates eight particle parameters while only three of them are needed. Therefore an extra code for the vertex reconstruction has been developed, where the calculations are reduced to the reconstruction of the particle position only.

Reconstruction of the primary event vertex is more than the fit of the vertex position. It is also necessary to determine which of the tracks belong to the vertex (primary tracks) and which are not (secondary tracks). Therefore the primary vertex reconstruction is usually performed iteratively:

1. It starts with preliminary selection of primary tracks.
2. The vertex is fitted with the selected tracks.

3. Those primary tracks which are too far (in the terms of χ^2) deviated from the vertex are removed.

4. The vertex is re-fitted, and so on from step 3.

The developed decayed particle mathematics allows one to skip the iterations. A track can be excluded from the fitted vertex by use of the formula for the measurement subtraction, given in Section (1.4.4). Having both filtering and subtraction operations implemented, one can add or remove tracks from the vertex at any time in an arbitrary order, making the fit iterations unnecessary.

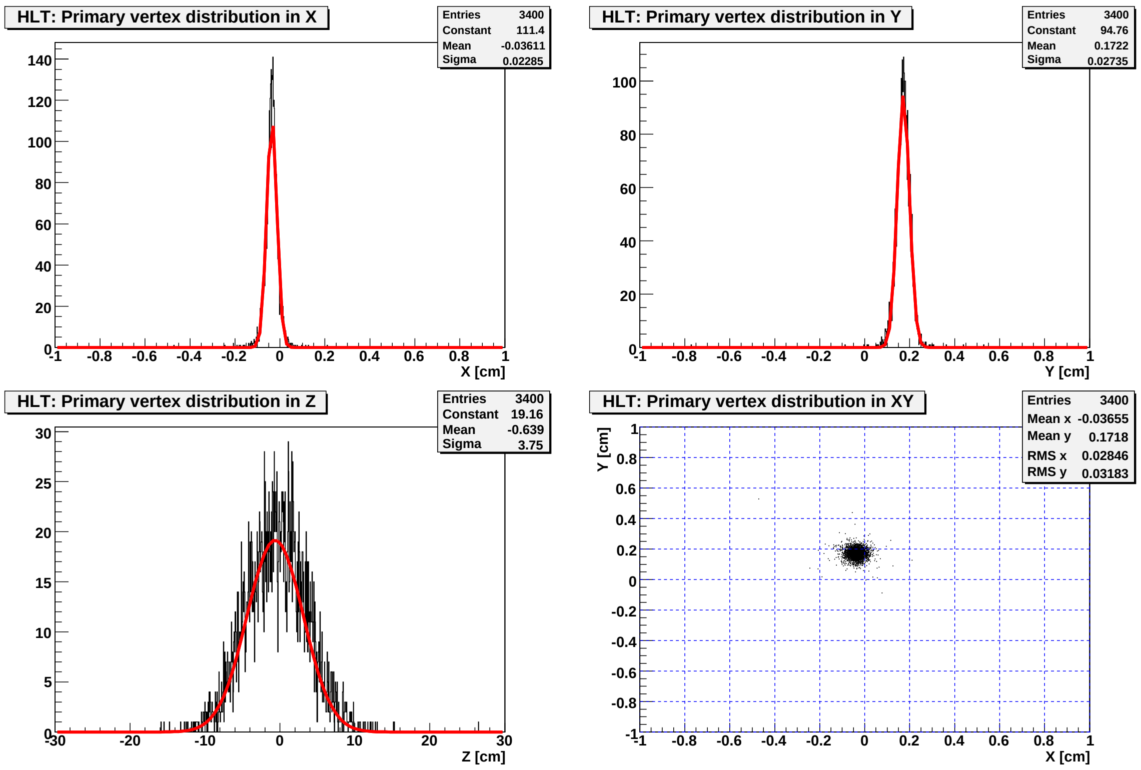


Figure 3.15: HLT primary vertex distribution. Real data, run 00010485 (2009).

Parameter	Resolution (μm)	Pull
x_v	100	1.07
y_v	102	1.12
z_v	138	1.11

Table 3.6: Residuals and normalised residuals (pulls) of the primary vertex parameters obtained from 1, 000 simulated proton-proton collisions at 14 TeV.

Performance of the primary vertex fit in the High Level Trigger for 1, 000 simulated proton-proton events is given in Table 3.6. The obtained vertex resolutions are 100 μm for x and y coordinates and 138 μm for the z coordinate, the pulls are close to unity. The speed of the algorithm is 1, 000 events per second, thus only one CPU is needed to reconstruct 14 TeV proton-proton data on-line.

The algorithm is used for the on-line reconstruction of the real data. Figure 3.15 shows an example of the primary vertex distribution, obtained by the HLT during first collision runs in the late 2009.

Chapter 4

On-line event reconstruction in the CBM experiment

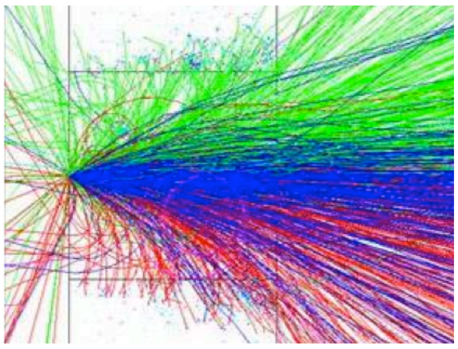


Figure 4.1: Simulated Au+Au event in CBM.

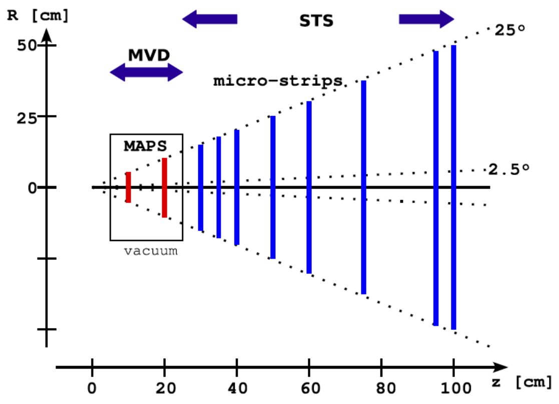


Figure 4.2: Geometry of the STS+MVD detectors in CBM.

4.1 Overview

The interaction rates in the CBM experiment are up to 10 MHz (minimum bias events) which corresponds to a beam intensity of 10^9 beam particles per second with 1% interaction target. Particle trajectories are detected by the Silicon Tracking System (STS) which is placed inside the magnet (Fig. 4.2). Large track densities (on average 500 tracks in the inner tracker in a typical central Au+Au collision, see Figure 4.1 as an example) together with the presence of a non-homogeneous magnetic field make the event reconstruction in STS complicated.

The reconstruction of tracks is based on the Cellular Automaton method. This method creates short three-hit track segments (tracklets) in neighbouring detector planes and links them into long tracks (see an illustration in Figure 4.3). The algorithm scheme is the following:

- First, the algorithm generates tracklets for each group of three consecutive STS stations. Cuts are applied to create only reasonable tracklets. To each tracklet a counter is assigned, which marks tracklet position on a track (initially the counter is set to 1— the first tracklet of a track).

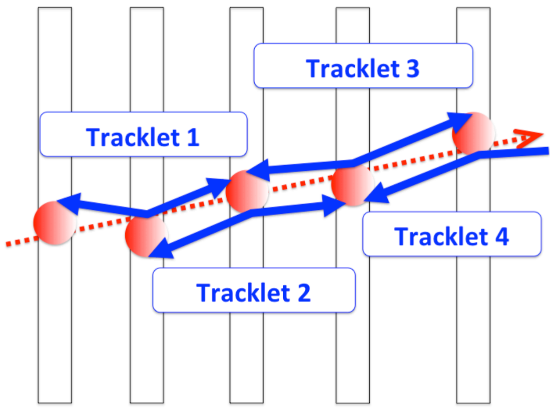


Figure 4.3: Construction of tracklets.

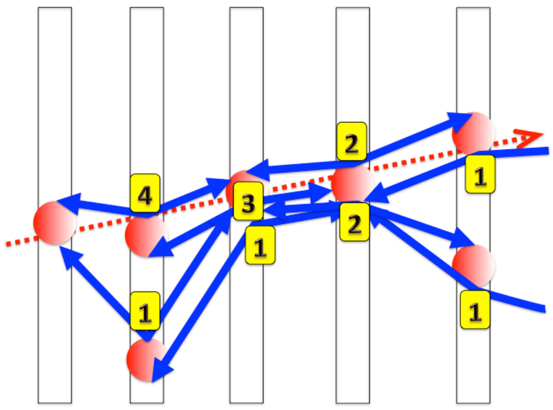


Figure 4.4: Illustration of the cellular automaton algorithm.

- In the second step, all the tracklets are extrapolated to the next STS station in the target direction.
- Then, each tracklet finds its neighbours among the tracklets, extrapolated from the previous station. The neighbours are possible tracklet continuations according to the track model. If there are neighbours found, the counter of the tracklet is incremented with respect to the largest neighbour's counter: $counter = counter^{max}_{neighb} + 1$, as it is illustrated in Figure 4.4.

After proceeding the above steps for all the STS stations, the algorithm builds track candidates out of the tracklets. It starts with the tracklets having the largest counter (in Figure 4.4 $counter^{max} = 4$). For each of these tracklets it takes a neighbour which has a $(counter = counter^{max} - 1)$. In case there are several neighbours which satisfy this condition, corresponding combinatorial branches are created. Then the algorithm follows the counters (e.g. goes from 4 to 3, but not 2) further, and finally keeps the best (χ^2) track for each initial tracklet with the largest counter.

Then, a selection of the created track candidates is performed. The algorithm starts with the best (in terms of χ^2) track and flags all hits of the track as “used”. It continues with the next track candidate (with higher χ^2), checks if the number of its ”used” hits (the hits which are already used by the other tracks) is acceptable and flags its hits (or deletes the candidate). Then it proceeds with the next track candidate, etc.

The algorithm repeats collecting tracks decrementing the maximal counter until the shortest tracks are collected.

Then, in case of a significant detector inefficiency the algorithm merges short tracks (clones) into long tracks using the track model.

Finally, the algorithm applies extra cuts to kill ghost tracks. Ghost tracks are mostly short tracks which are not pointed to the interaction point.

Being essentially local and parallel, the cellular automaton method avoids exhaustive combinatorial searches, even when implemented on conventional computers. Since the algorithm operates with highly structured information, the amount of data to be processed in the course of the track search is significantly reduced.

For evaluation purposes all simulated and reconstructed tracks are subdivided into several

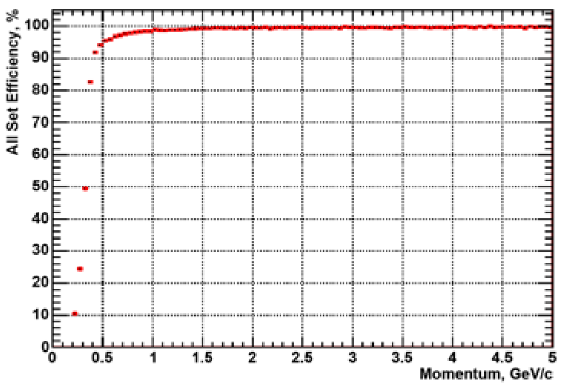


Figure 4.5: Track reconstruction efficiency as a function of momentum.

categories: one set for *reference* tracks, one for *clone* tracks, one for *ghost* tracks, and one *extra* set for non-reference tracks, and also *clone* and *ghost* tracks [12]. The *reference* and the *extra* tracks compose the set of *all* reconstructible tracks.

By definition, a track from the *all* set should intersect the sensitive regions of at least four stations. In addition, a *reference* track should have a momentum greater than 1 GeV/c . The *reference* set of tracks also includes tracks of particular interest to physics: secondary tracks from interesting decays; primary tracks coming from the target region. In addition to these tracks, an *extra* set of tracks is considered, containing tracks in the *all* set which are not *reference* tracks.

Track category	Efficiency, %
Reference set	99.45
All set	96.98
Extra set	89.46
Clones	0.01
Ghost	0.61

Table 4.1: Tracking efficiency for different sets of tracks.

A reconstructed track is assigned to a generated particle, if at least 70% of its hits have been caused by this particle. A generated particle is regarded as found, if it has been assigned to at least one reconstructed track. If the particle is found more than once, all additionally reconstructed tracks are regarded as *clones*. A reconstructed track is called *ghost* when it is not assigned to any generated particle (70% criteria).

The efficiency of track reconstruction for particles detected in at least four stations is presented in Fig. 4.5 and Table 4.1. Tracks of high momentum particles are reconstructed very well with efficiencies of 99.45%, while multiple scattering in detector material leads to a lower reconstruction efficiency of 89.46% for slow particles from the extra set of tracks.

The total efficiency for all tracks with a large fraction of soft secondary tracks is 96.98%. The clone rate of the algorithm is 0.01% and the ghost rate 0.61%.

The track and the vertex fit in CBM is performed by the Kalman filter, described in Section 2.2. As the fit is applied already at the level of tracklet creation, the speed of the fitter is very important for the on-line data processing.

Starting from the conventional Kalman filter with a Runge-Kutta extrapolator, the fit was improved by developing a special analytic extrapolator, described in Section 2.2.2. Next, the fit was sped up by a factor of $10,000$ by memory optimisation (in particular, by introducing a polynomial approximation of the magnetic field) and by use of SIMD CPU instructions. The optimisation of the track fit is described in the next Section.

4.2 Fast SIMDised Kalman filter

To achieve high track-finding efficiency, the Kalman filter fitting algorithm is extensively used withing the track finder. Therefore the speed of the Kalman filter is of crucial importance in on-line data reconstruction.

Having SIMD instructions implemented in Pentium 4 processors allows one to increase the speed of the Kalman filter by rewriting the algorithm in terms of vectors instead of scalars. The vectorisation will become even more important in the near future when Cell processor based PCs become widely used in data processing.

In order to speed up the on-line CBM reconstruction, the track fitter was first modified to use the SIMD unit of the Pentium 4 processor [20] and then ported to the Cell processor [21, 22] which is considered a candidate for the L1 hardware [26].

4.2.1 SIMD architecture

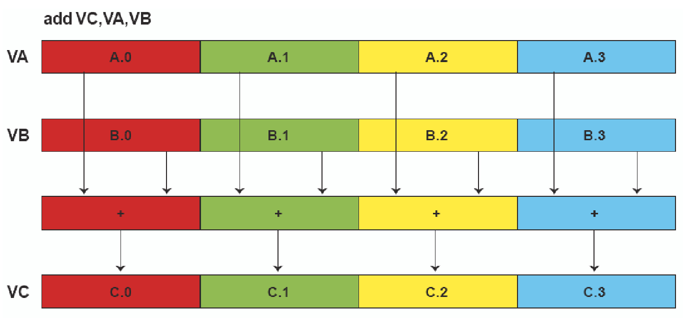


Figure 4.6: Four concurrent add operations [22].

There are three important classes of computer architectures based upon the number of concurrent instruction and data streams:

- Single instruction, single data stream (SISD) — a single instruction stream on scalar data.
- Single instruction, multiple data streams (SIMD) — multiple data streams against a single instruction stream to perform operations which may be naturally parallelised.

- Multiple instruction, multiple data (MIMD) — many functional units perform different operations on di erent data.

The basic data unit of SIMD is the vector, which is why SIMD computing is also known as vector processing. A vector is a row of individual numbers or scalars. A regular CPU operates on scalars, one at a time. A vector processor, on the other hand, lines up a whole row of these scalars, all of the same type, and operates on them as an unit.

These vectors are represented in packed data format. Data is grouped into bytes (8 bits) or words (16 bits) and packed into a vector to be operated on.

The vector size defining the number of scalars processed in parallel is one of the most critical design aspects of SIMD implementations. For instance, using a 4-element, 128-bit vector one can do four-way single-precision (32-bit) foating-point calculations in parallel (see Fig. 4.6).

Today, SIMD instructions can be found on most CPUs, including the PowerPC's AltiVec, Intel's MMX, SSE, SSE2, SSE3 and SSE4 as well as AMD's 3DNow!

A flexible C++ interface to the SIMD instructions has been developed for the CBM on-line framework. The interface allows one to run the same source code on different SIMD architectures as well as on scalar CPUs.

4.2.2 Cell Broadband Engine

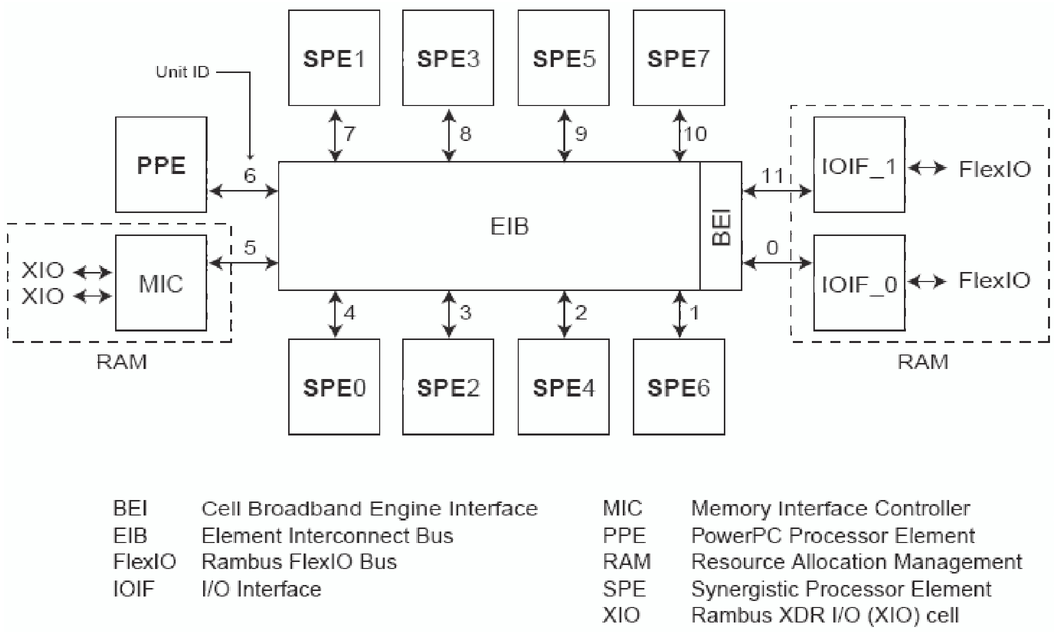

SPE1
SPE3
SPE5
SPE7
Unit ID
PPE
6
7
8
9
10
11
EIB
BEI
5
0
XIO
XIO
MIC
IOIF_1
IOIF_0
FlexIO
FlexIO
4
3
2
1
RAM
RAM
SPE0
SPE2
SPE4
SPE6
BEI Cell Broadband Engine Interface
EIB Element Interconnect Bus
FlexIO Rambus FlexIO Bus
IOIF I/O Interface
MIC Memory Interface Controller
PPE PowerPC Processor Element
RAM Resource Allocation Management
SPE Synergistic Processor Element
XIO Rambus XDR I/O (XIO) cell

Figure 4.7: Cell Broadband Engine overview [22].

Cell[1] is a microprocessor architecture developed jointly by a Sony, Toshiba, and IBM alliance known as STI. Cell combines a general-purpose Power-architecture core of modest

[1]Cell is a shorthand for Cell Broadband Engine Architecture, commonly abbreviated CBEA in full or Cell BE in part.

performance with multiple streamlined co-processing elements which greatly accelerate multimedia and vector processing applications, as well as many other forms of dedicated computation. [21, 22]

The resulting architecture emphasizes efficiency/watt, prioritises bandwidth over latency, and favors peak computational throughput over simplicity of program code. For these reasons, Cell is widely regarded as a challenging environment for software development.

The major commercial application of Cell is in Sony's PlayStation 3 game console. Although the Cell Broadband Engine was initially intended for applications in game consoles and media-rich consumer-electronics devices, such as high-definition televisions, the architecture and the Cell Broadband Engine implementation have been designed to enable fundamental advances in processor performance.

The Cell Broadband Engine (Fig. 4.7) is a single-chip multiprocessor with nine processors operating on a shared, coherent memory. The Cell processor can be split into four components:

- the main processor called the Power Processing Element (PPE) (a two-way SMT multithreaded Power 970 architecture compliant core),
- eight fully-functional co-processors called the Synergystic Processing Elements (SPEs),
- a specialised high-bandwidth circular data bus connecting the PPE, input/output elements and the SPEs, called the Element Interconnect Bus (EIB),
- external input and ouput structures.

The first type of processor — the PPE — is not intended to perform all primary processing for the system, but rather to act as a controller for the other eight SPEs, which handle most of the computational workload. The PPE is fully compliant with the 64-bit PowerPC Architecture and can run 32-bit and 64-bit operating systems and applications.

The second type of processor — the SPE — has RISC architecture with a fixed-width 32-bit instruction format. It is optimised for running compute-intensive applications, and it is not optimised for running an operating system. The SPEs are designed for vectorised floating point code execution.

In one typical usage scenario, the system loads the SPEs with small programs, chaining the SPEs together to handle each step in a complex operation. Another possibility is to partition the input data set and have several SPEs performing the same kind of operation in parallel.

4.2.3 Speed-up of the Kalman filter algorithm

Being the core part of the reconstruction software of the CBM experiment, the Kalman filter track fit procedure has been chosen for investigation of its vectorisation ability and further implementation in the Cell processor.

The track fitting routine in the CBM experiment realises the Kalman Filter in its conventional approach. All variables in the routine are scalars and most of them have floating point representation in double precision.

In the first stage, the memory access in the algorithm has been optimised, because operating data in the main memory is significantly slower compared to working within the cache. This is especially true for the Cell processor with the size of the local storage of SPEs comparable with the cache size, where unpredictable access to the main memory of the PPE is a blocking process which stalls the algorithm.

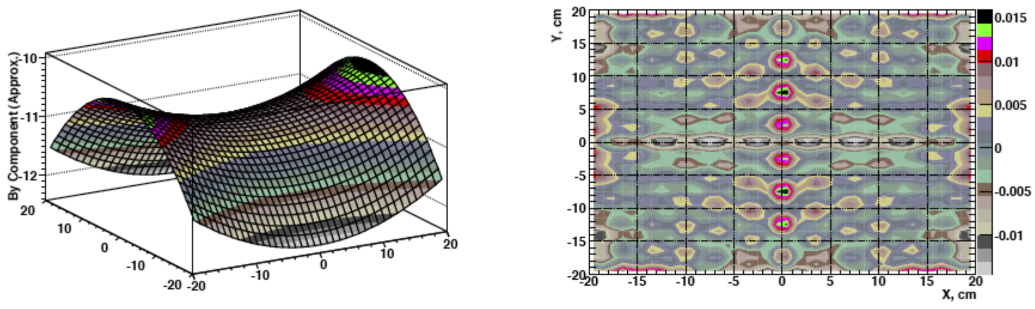


Figure 4.8: The most significant (By) component of the magnetic field in the middle of the detector system (z = 50 cm) calculated using the polynomial approximation (left) and the difference between two alternative field representations (right).

The original fitter permanently accesses the main memory, as it needs to read the non-homogeneous magnetic field which is stored in a 70-MByte large map. The access to the field map was avoided by the use of a local polynomial approximation of the field due to the fact that the field is relatively smooth in XY-slices.

For the propagation step of the Kalman filter it was found to be sufficient to use a polynomial of the 4-th order to approximate the field in STS stations (see Fig. 4.8 for comparison of two alternative field representations and Fig. 4.2 to recall the STS geometry).

The field behaviour in the space between the STS stations is approximated for every tracklet individually as a parabola, calculating parabola coefficients from the field taken at the track hits, since only the field value along the track is needed [10]. Track parameters taken with the polynomial approximation of the magnetic field are as good as those calculated using the magnetic field map showing no degradation at all.

At the second optimisation stage the fit algorithm has been signifcantly modified in two directions: changing variables in the algorithm from double to single precision and using a computationally optimised implementation of the Kalman filter method.

Operating with data in single precision has several advantages. First, memory requirements are reduced by 50%. This results in faster memory access, since twice as much data can be read into the cache of a conventional CPU or into local storage of the SPE. Second, twice the amount of data can be packed into a vector, thus automatically doubling the speed of the SIMDised algorithm. Third, the current implementation of the Cell processor is optimised for SIMD operations in single precision. Therefore, performing double precision operations is an order of magnitude slower [21].

Changing the precision of all floating point variables from double to single precision, it has been realised that 32 bits of single precision is not enough for the conventional Kalman filter to be numerically stable. The outcome is unacceptable due to poorer quality of the track parameters and bad numerical properties of the covariance matrix. In particular, the matrix has negative diagonal elements, which theoretically cannot be and makes the results useless.

It is possible to keep some variables (such as the covariance matrix) in double precision and also process critical calculations in double precision. However, operations in double precision on SPE have significant extra charge. Therefore it was decided to find a numerically stable and accurate single precision approach of the Kalman filter.

Step	Description	Timing, %
1	Initialisation	11
2	Prediction	45
3	Process noise	8
4	Filtration	36

Table 4.2: Timing (in %) for different steps of the Kalman filter.

There are several methods to improve the numerical quality of the conventional Kalman filter [24]. One of the best numerically stable single precision approaches is the square root implementation of the Kalman filter [24, 25], where calculations are performed using the square root of the covariance matrix ($\mathrm{C}_k = \mathrm{S}_k \mathrm{S}_k^T$). Although algebraically equivalent to the conventional approach, the square root filter exhibits improved numerical characteristics, providing the same accuracy in single precision as the conventional Kalman filter in double precision. The square root filter includes extra transformations and therefore requires about 30% more processing time. Such an overhead is usually considered acceptable for the benefit of improved numerical stability.

The square root filter was implemented first. Then a comprehensive analysis showed that the only source of the Kalman filter instability is the filtration of the first measurements. At this sage errors of the initial track parameters are several orders of magnitude larger than errors of the measurements. It causes significant rounding errors (see eqs. 1.11) and loss of precision in the calculations. A reduction of the initial track errors makes the calculations stable, but degrades the quality of the track parameters, as they become biased to the initial values.

The dilemma was resolved by splitting the filtration mathematics into two branches.

The covariance matrix is updated in the conventional way, while the track parameters are evaluated differently for the case when the track errors are much bigger than the measurement error. In this case the filtering equations for the state vector are simplified by eliminating the measurement error. This approach keeps the algorithm stable and accurate in single precision. In contrast to the square root filter, this method requires only negligible amount of extra calculations. Therefore it has the same speed as the original Kalman filter algorithm.

When the numerically stable approach was found, the Kalman filter was mathematically optimised. Originally it started with arbitrary initial track parameters and a large covariance matrix, and then iterated the whole fit several times in order to converge to the optimal solution. A preliminary estimation of the track parameters from the input data reduced the number of necessary iterations to one. Furthermore, as a result of the polynomial approximation of the magnetic field, the propagation step of the Kalman filter can be performed directly from measurement to measurement without the necessity of additional intermediate steps. Other optimisations have also been implemented, like the replacement of matrix multiplications by direct operations on non-trivial matrix elements only.

The algorithm has also been extensively analysed with respect to its numerical optimisation, for instance: most of the loops have been unrolled in order to provide additional instructions for interleaving; most branches have been eliminated from the algorithm to avoid branch misprediction penalties; calculations have been reordered for a better use of the processor pipeline.

Table 4.2 gives the relative timing for different steps (see Section 1) of the Kalman filter routine. One can see that even without reading the magnetic field map the propagation of the track parameters (the prediction step) is still the most time consuming part of the fit procedure because of the complexity of the propagation in a non-homogeneous magnetic field.

At the third optimisation stage the algorithm has been vectorised. The parallelisation scheme of the track fit is simple: corresponding parameters of four tracks are packed into a vector. Denoting the parameters of the four original tracks as

$$\begin{aligned} \mathbf{T}_1 &= \{p_1^1,\, p_1^2,\, p_1^3,\, p_1^4,\, p_1^5\} \\ \mathbf{T}_2 &= \{p_2^1,\, p_2^2,\, p_2^3,\, p_2^4,\, p_2^5\} \\ \mathbf{T}_3 &= \{p_3^1,\, p_3^2,\, p_3^3,\, p_3^4,\, p_3^5\} \\ \mathbf{T}_4 &= \{p_4^1,\, p_4^2,\, p_4^3,\, p_4^4,\, p_4^5\} \end{aligned} \tag{4.1}$$

the SIMDised track is

$$\begin{aligned} \mathbf{T} = \{ \ & (p_1^1, p_2^1, p_3^1, p_4^1)\,, \\ & (p_1^2, p_2^2, p_3^2, p_4^2)\,, \\ & (p_1^3, p_2^3, p_3^3, p_4^3)\,, \\ & (p_1^4, p_2^4, p_3^4, p_4^4)\,, \\ & (p_1^5, p_2^5, p_3^5, p_4^5) \ \} \end{aligned} \tag{4.2}$$

Even such a simple vectorisation scheme requires preliminary sorting of the tracks according to their length in order to avoid branches in the vectorised algorithm. The problem of having branches can be very non-trivial for more complicated algorithms, like a track finder with a lot of combinatorial analysis. In some of the cases one should not be restricted to using the vertical vector operations, but should also use horizontal operations to exchange the content of different vectors, avoiding the algorithm branches within a vector.

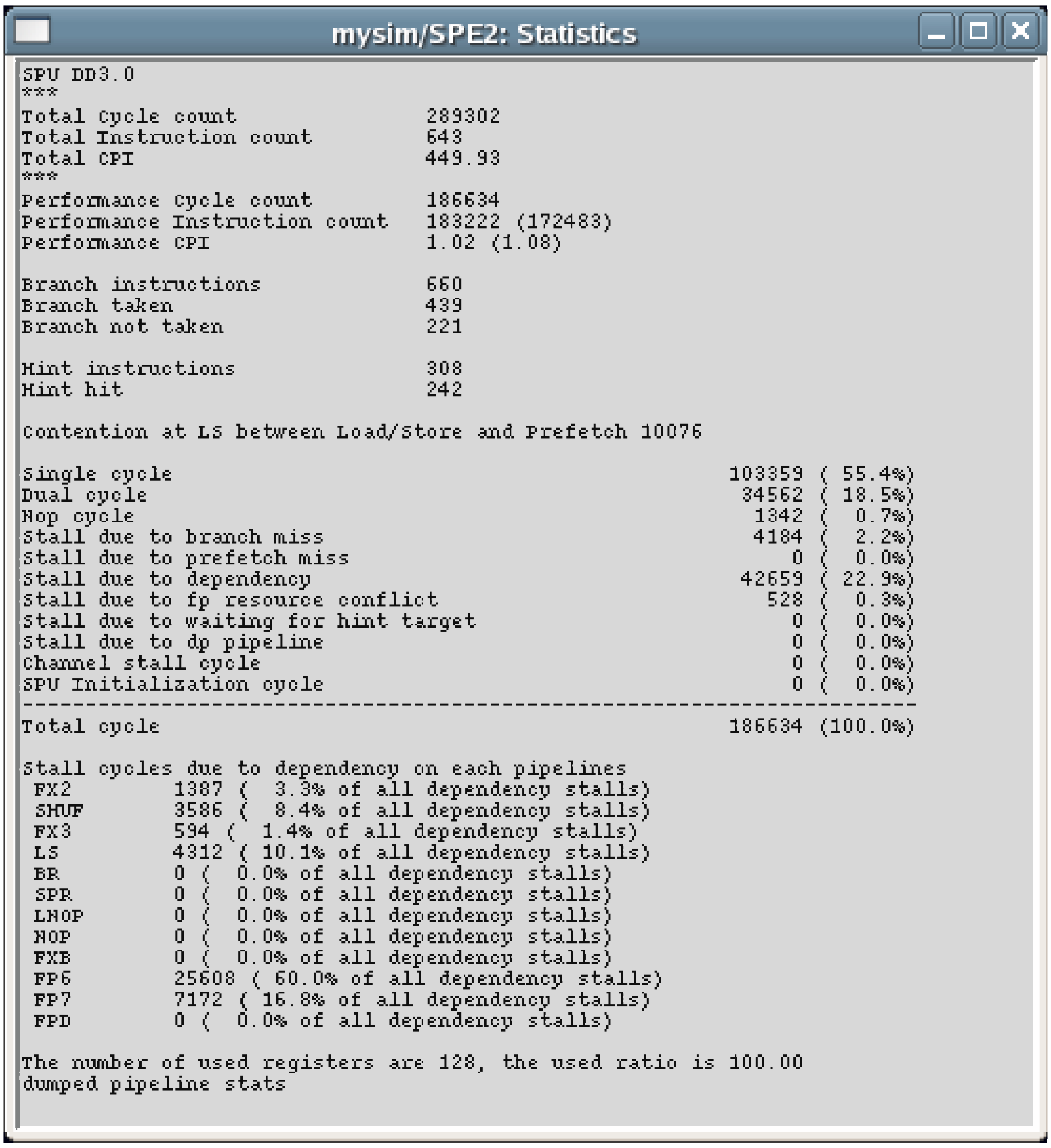
After the parallelisation scheme was developed and tested, the track fit algorithm was adopted to use the SSE2 instruction set. The problem is that the vector instructions look completely different from the corresponding scalar instructions: for instance, the scalar operation $c = a+b$ becomes $c = vec\_add(a, b)$. Re-writing the code using vector instructions would require support for both – scalar and vector – versions, in particular, duplicating modifications in both versions and initiating an extra loop of debugging and testing. Therefore, it was decided to implement the SSE2 vector instruction set in a header file, overloading all operands and inlining several functions.[2] In this way the same source code is used in scalar and vector implementations and possible changes in the code are valid for both versions. The quality of the track parameters and the covariance matrix of the SIMDised version and of the scalar version are identical.

At the fourth stage of the optimisation the algorithm has been ported into the Cell simulator and run on the PPE processor. For that the AltiVec instruction set of the PPE was implemented in a header file. It was still possible to run both – scalar and vector – versions on the PPE examining the consistency of the results.

In the last step the code was ported to the SPE processor. Again, this was accomplished by writing another header, which implements the specialised SIMD instruction set of the SPE. In addition, the code was slightly modified in order to provide communication between the PPE and the SPE and to exchange data between the main memory and the local store of the SPE. Because the total size of the SPE code is only 50 kB, the code fits very well in the local store of the SPE leaving the remaining 200 kB for data.

The SPU statistics of the Cell simulator is given in Fig 4.9. It shows that the algorithm achieves a very good overall cycles per instruction (CPI) performance of 1.03. It has 15.5%

[2]In case no SIMD instruction set is supported by a computer, the vector type is substituted by the pseudovector array of four scalars.



mysim/SPE2: Statistics

```
SPU DD3.0
***
Total Cycle count              289302
Total Instruction count        643
Total CPI                      449.93
***
Performance Cycle count        186634
Performance Instruction count  183222 (172483)
Performance CPI                1.02 (1.08)

Branch instructions            660
Branch taken                   439
Branch not taken               221

Hint instructions              308
Hint hit                       242

Contention at LS between Load/Store and Prefetch 10076

Single cycle                                             103359 ( 55.4%)
Dual cycle                                                34562 ( 18.5%)
Nop cycle                                                  1342 (  0.7%)
Stall due to branch miss                                   4184 (  2.2%)
Stall due to prefetch miss                                    0 (  0.0%)
Stall due to dependency                                   42659 ( 22.9%)
Stall due to fp resource conflict                           528 (  0.3%)
Stall due to waiting for hint target                          0 (  0.0%)
Stall due to dp pipeline                                      0 (  0.0%)
Channel stall cycle                                           0 (  0.0%)
SPU Initialization cycle                                      0 (  0.0%)
-----------------------------------------------------------------------
Total cycle                                              186634 (100.0%)

Stall cycles due to dependency on each pipelines
 FX2        1387 (  3.3% of all dependency stalls)
 SHUF       3586 (  8.4% of all dependency stalls)
 FX3        594 (  1.4% of all dependency stalls)
 LS         4312 ( 10.1% of all dependency stalls)
 BR         0 (  0.0% of all dependency stalls)
 SPR        0 (  0.0% of all dependency stalls)
 LNOP       0 (  0.0% of all dependency stalls)
 NOP        0 (  0.0% of all dependency stalls)
 FXB        0 (  0.0% of all dependency stalls)
 FP6        25608 ( 60.0% of all dependency stalls)
 FP7        7172 ( 16.8% of all dependency stalls)
 FPD        0 (  0.0% of all dependency stalls)

The number of used registers are 128, the used ratio is 100.00
dumped pipeline stats
```

Figure 4.9: A dynamic timing analysis of the SPE using the IBM Full System Simulator for the Cell Broadband Engine.

dual-issue (odd and even pipeline use) rates, almost no stall due to branch misses (1.9%) and low dependency stalls (19.3%). This is an excellent result for such a complicated algorithm. It is also important that all of the 128 registers have been used.

After extensive tests on the simulator, the algorithm ran on a Dual Cell-Based Blade computer running at 2.4 GHz. There were no significant problems observed at this stage.

At the last stage of testing all 16 SPEs of the two Cell processors available on the Cell Blade where running in parallel to process different data samples.

4.2.4 Performance of the SIMDised track fit

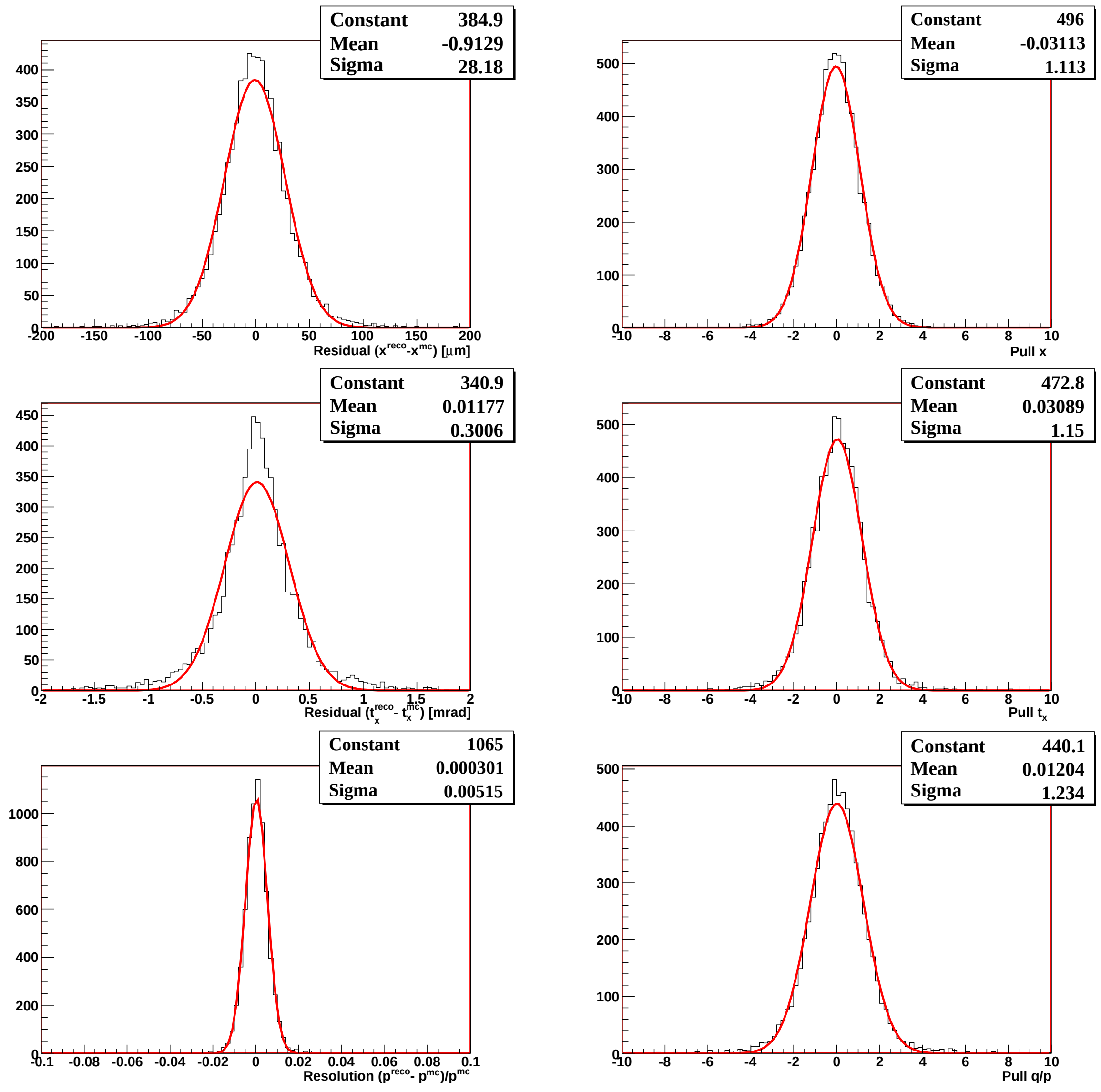


Figure 4.10: Residuals and normalised residuals (pulls) of the estimated track parameters at the production vertex for central Au+Au collisions at 35 AGeV in the approximated magnetic field of the CBM experiment obtained on the Cell Blade computer.

The Kalman filter based track fit has been tested on simulated data of the CBM experiment.

Stage	Description	Time/track	Speed-up factor
	Initial scalar version	12 ms	
1	Approximation of the magnetic field	240 μs	50
2	Optimisation of the algorithm	7.2 μs	35
3	Vectorisation	1.6 μs	4.5
4	Porting to SPE	1.1 μs	1.5
5	Parallelisation on 16 SPEs (2 Cells)	0.1 μs	10
	Final SIMDised version	0.1 μs	120000

Table 4.3: Summarised stages of the optimisation procedure.

In the CBM experiment with forward geometry the natural choice of the state vector[3] is:

$$\mathbf{r} = \{x, y, t_x, t_y, q/p\}$$

where x and y are track coordinates at the reference z-plane, $t_x = \tan\theta_x$ is the track slope in the xz plane, $t_y = \tan\theta_y$ is the track slope in the yz plane, and q/p is the inverse particle momentum signed according to its charge.

Quality of the track parameters was monitored at all stages of the optimisation of the algorithm. Figure 4.10 shows residuals and normalised residuals (pulls) of the track parameters at the production vertex obtained on a Cell Blade computer.

The residuals of fitted track parameters, for instance, of the x-coordinate, are determined as:

$$\rho_x = x^{reco} - x^{mc}$$

where x^{reco} — reconstructed and x^{mc} — true Monte-Carlo values of the x-coordinate.

The normalised residual (pull) distributions of the fitted track parameters are a measure of the reliability of the fit. Normalised residuals are determined according to the formula:

$$P(x) = \frac{\rho_x}{\sqrt{\mathrm{C}_{xx}}}$$

where C_{xx} — the corresponding diagonal element of the covariance matrix, obtained in the track fit. In the ideal case the normalised error distributions of the coordinates and slopes of the track should be unbiased and Gaussian distributed with width of 1.0.

To get the parameters and the covariance matrix of a track at a vertex where the track originates, the fitted track was propagated to its Monte-Carlo vertex, taking into account the remaining traversed material.

Figure 4.10 gives also the RMS of the Gaussian fit of the residuals and pulls. All the pulls are centered at zero indicating that there is no systematic bias in the reconstructed values. The pull distributions are well fitted by a Gaussian with small tails caused by the various non-Gaussian contributions to the fit. The q/p pull shows slightly underestimated errors. This could be the result of several approximations made in the fitting procedure, mainly in the part of the material treatment.

Table 4.3 summarises all stages and gives a timing analysis and the speed-up factors after each development stage. One can see that the elimination of the magnetic field map (thus elimination of the access to main memory during the fit) boosts the speed of the Kalman

[3]The z-coordinate points downstream the spectrometer axis, and x and y are transverese coordinates.

filter 50-fold. A further 35-fold increase in speed was achieved by the code optimisations. The vectorisation of the code at the next stage of the development improved the speed by a factor of 4.5.[4] In contrast to pure software improvements at the first two stages, the vectorisation required both software and hardware changes. Porting to the SPE at the next stage resulted in a 1.5-fold boost of the speed with respect to the Pentium 4 processor used at the previous stages, probably due to a higher number of registers in the SPE. At the last stage, the implementation was multi-threaded using all 16 SPEs of the Cell Blade computer. This boost performance 10-fold over a single SPE. In total, the Kalman filter was sped up by a factor of $120,000$.

An extra benchmark of the SIMDised version of the Kalman has been done on three different computers based on:

- Two Intel Xeon Processors with Hyper-Threading enabled and 512 kB cache at 2.66 GHz; [5]
- Two Dual Core AMD Opteron Processors 265 with 1024 kB cache at 1.8 GHz; [6]
- Two Cell Broadband Engines with 256 kB local store at 2.4 GHz. [7]

Both Intel and AMD based personal computers are treated by the operating system as having four processors each.

Processing Unit	Clock	Time/track	kCycle/track
Intel Xenon	2.66 GHz	1.47 μs	3.91
AMD Opteron	1.80 GHz	1.86 μs	3.35
Cell SPE	2.4 GHz	0.87μs	2.09

Table 4.4: Real-time performance of the SIMDised version of the Kalman filter fit for a single track, fitted on three different CPU families.

Table 4.4 gives real-time performance for the fit of a single track on different computers. Only one processing unit (CPU or SPU) is active, while others are in the idle state or running the operating system. Since only one track of about 0.5 kB size is fitted, all computations are located within the cache or the local memory. The Cell Blade computer has the fastest performance requiring half clocks per track with respect to the Intel Xeon based computer.

Figure 4.11 gives real time per track for the SIMDised version of the Kalman filter for the Intel Xeon, AMD Opteron and Cell based computers running a different number of processes in parallel. A very large sample of tracks exceeding many times the size of the cache or the local store has been processed in order to include the effect of communication to the main memory. Compared to Table 4.4 one can see that for all computers there is a little overhead of about 10% for reading data from the main memory. It is also clear that the hyper-threading of the Intel Xeon processor does not contribute in this particular case of the fit procedure, in contrast the dual core technology of the AMD Opteron processor which shows the stability of the timing performance due to its architecture (four full cores). In the Cell Blade computer all 16 SPEs work in parallel completely independently. They have the constant speed of the

[4]The SIMDised code uses special CPU instructions, like fast inversion, which are not available for scalar registers.

[5]lxg1411 at the Gesellschaft fur Schwerionenforschung mbH, 64291 Darmstadt, Germany.

[6]eh102 at the Kirchho Institute for Physics, University of Heidelberg, 69120 Heidelberg, Germany.

[7]blade11bc4 at the IBM Laboratory Boblingen, Schonaicher Str. 220, 71032 Boblingen, Germany.

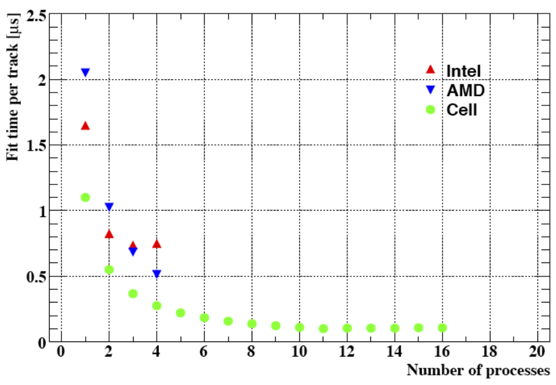


Figure 4.11: Fit time per track of the SIMDised version of the Kalman filter measured in real time for the Intel Xeon, AMD Opteron and Cell based computers running different number of processes in parallel.

algorithm per processing unit up to 11 processes, then slightly reducing the speed probably due to large data flow through the element interconnect bus.

Having significant differences in the architectures and clock rate all computers have shown similar speed[8] of the algorithm per processing unit. The reason may be that the final algorithm does not require large memory and most of the calculations are done within the cache or the local store. Second, the algorithm implements the Kalman filter technique in the same source code, which after compilation by a gcc compiler produces executables with similar performances.

The local store of the SPE requires special consideration, but can give more freedom to the developer compared to the cache in Intel or AMD processors. The vector instruction set SSE2 has relatively limited capabilities to operate with the cache. In contrast, the instruction set of the SPE has a considerable number of operands for (non-blocking) operation with the local store. In our case, the track fit algorithm does not require a large exchange of data between the local store and the main memory, therefore this difference between the processors has not been observed.

[8]It should be noted, that there are processors with higher clock rates available.

Chapter 5

On-line event reconstruction in the ALICE High Level Trigger

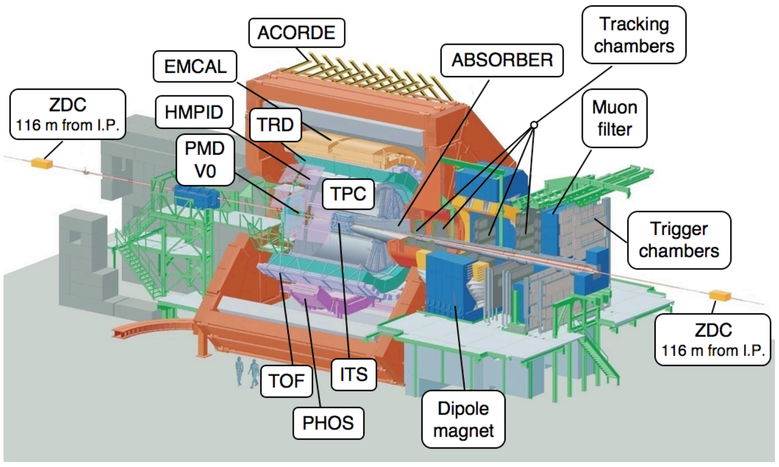


Figure 5.1: The ALICE spectrometer at LHC.

5.1 Overview

The ALICE High-Level Trigger [27, 28] processes proton-proton collisions at 2 kHz and heavy ion collisions at 300 Hz; with an average of 25 tracks in each proton-proton event and up to 25, 000 tracks in the heavy ion events corresponding to an input data stream of 30 GB/s. Figures 5.2 and 5.3 show complexity of both types of events to be reconstructed in real-time.

In recent years, the increase in processor clock speed has stagnated. Instead a new trend to multi- and many-core chips has developed. It is evident that, for raw computation power, the best approach is a big set of small and simple cores as it has been realized within graphics cards for many years now.

While at first they could only be used for very special problems using algorithms that had to be developed with a particular architecture in mind, there are frameworks available today to run general purpose code written in high level languages on GPUs with little changes.

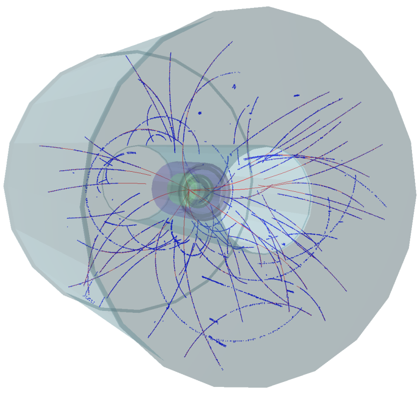


Figure 5.2: Proton-proton event in the ALICE TPC detector, reconstructed by HLT.
Real data, run 00010480 (2009).

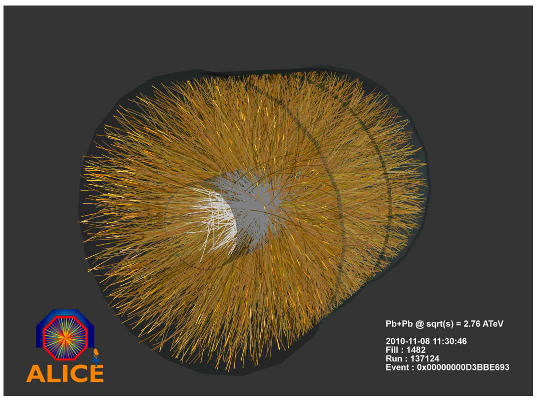


Figure 5.3: Heavy ion event in the ALICE TPC detector, reconstructed by HLT (TPC clusters are not shown).
Real data, run 00137124 (2010).

The Cellular Automaton tracking algorithm [29, 30] used in the ALICE HLT for online reconstruction has been developed with multi-core support in mind. All steps of the algorithm can easily be spread over many independent processors. Primarily targeted at processing Pb-Pb events, with up to $25,000$ tracks and several million clusters (Fig. 5.3), the tracker was adjusted to run on GPUs. A framework being able to run the same source code on a CPU as well as a GPU was developed, where the same source files are included in wrappers for both processor types. This assures that code maintainability does not suffer.

5.2 HLT Reconstruction Scheme

The Time Projection Chamber (TPC) detector, which is shown on Figures 5.1, 5.2, 5.3, is the main tracking detector of the ALICE experiment. It consists of 18 sectors on either Z-side. The detector measures the track positions in 159 rows as shown in Figure 5.5.

The overall reconstruction scheme is presented in Figure 5.4. It starts with the TPC cluster finder, which finds the hits by identifying localised clusters and computing their centre of gravity. These reconstructed hits are sent to the sector tracker which reconstructs the tracks in each TPC sector individually. Then the sector tracks are merged by the track merger algorithm and later updated with the measurements from the ITS detector. The reconstruction of the event's vertex and the physical triggers are running at the end of the reconstruction tree structure. Typically every processing stage reduces the size of the event data.

This scheme processes data as early as possible avoiding any unnecessary copy steps and uses all available data locality and parallelisation.

The core of the event reconstruction happens in the TPC sector tracker, which creates the tracks from the measurements. It is the only component which processes the TPC hits, the higher level components operate on the reconstructed sector tracks.

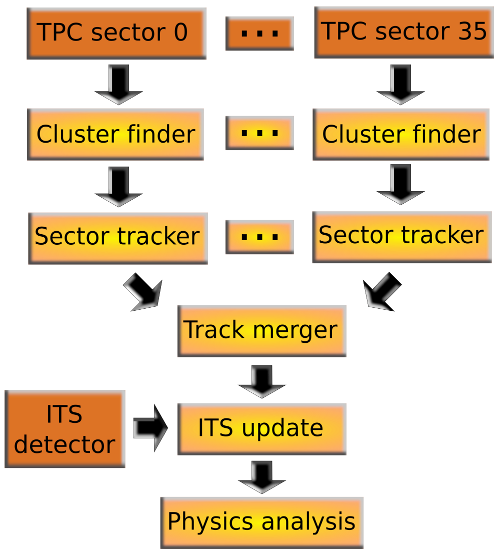


Figure 5.4: HLT reconstruction scheme.

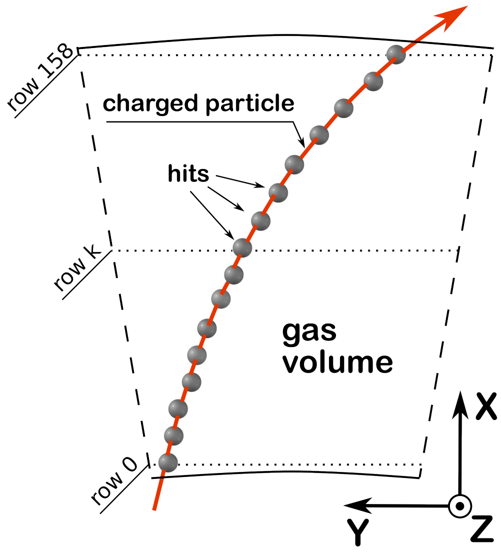


Figure 5.5: Geometry of a TPC sector.

5.3 HLT Tracker Algorithm

An event coming from the detector only contains information about the spatial position of hits, but no information about the particles which caused the hits. The task of the track finder is to group the hits in such a way that they form the particle trajectories.

This is a combinatorial pattern recognition problem. Since the potential number of hit combinations is enormous,[1] there is no exact solution to the problem, therefore heuristic methods are applied. Due to the large combinatorial background the key issue is the dependence of the reconstruction time on the number of tracks to be reconstructed. Figure 5.6 shows that the presented algorithm requires 130 μs per track independently from detector occupancy, thus the combinatorial part of the algorithm is built optimally.

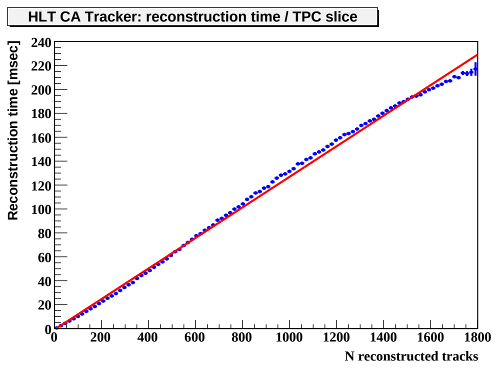


Figure 5.6: Reconstruction time on CPU.

The track reconstruction algorithm starts with a combinatorial search for track candidates

[1]For example, given n tracks producing hits in each of 159 TPC rows, the number of possible hit combinations to create a single track is equal to n^{159}.

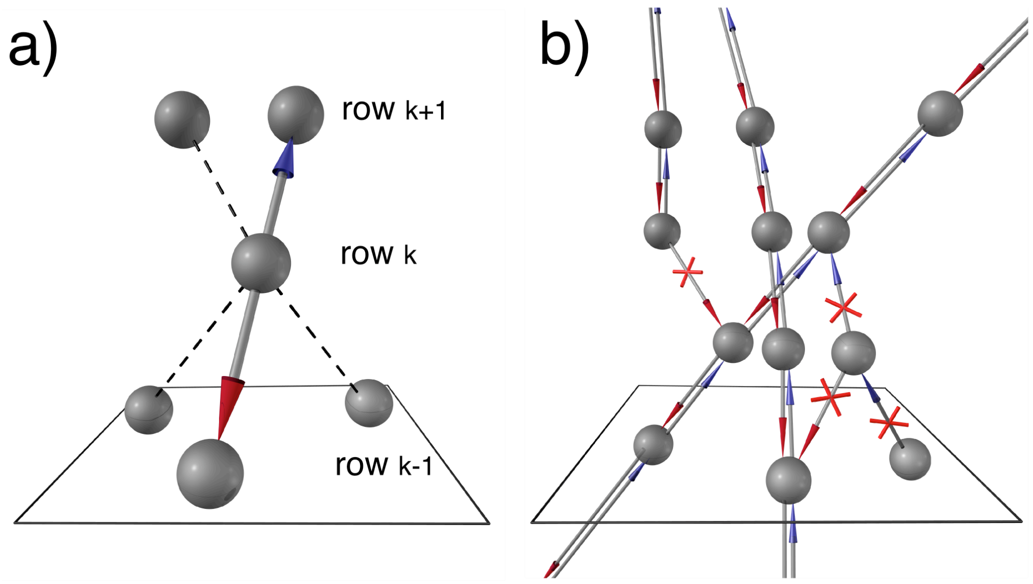


Figure 5.7: a) Neighbours finder. b) Evolution step of the Cellular Automaton.

(tracklets), which is based on the Cellular Automaton method [18]. Local pieces of trajectories are created from hits which are located nearby to each other, thus eliminating absurd hit combinations at the local level. The combinatorial processing composes the following two steps:

- 1. Neighbour finder: For each hit at k-th row the best pair of neighbouring hits from rows k+1 and k-1 is found, as it is shown in Fig. 5.7 a). The neighbour selection criteria requires the hit and its two best neighbours to form a straight line. The links to the best two neighbours are stored. Once the best pair of neighbours is found for each hit, the step is completed.

- 2. Evolution step: Reciprocal links are determined and saved, all the other links are removed (see Fig. 5.7 b)).

Every saved one-to-one link defines a part of the trajectory between the two neighbouring hits. Chains of consecutive one-to-one links define the tracklets. One can see from Fig. 5.7 b) that each hit can belong to only one tracklet because of the strong evolution criteria.

This uncommon approach is possible due to the abundance of hits on every TPC track.

Such a strong selection of tracklets results in a linear dependence of the processing time on the number of track candidates. When the tracklets are created, the sequential part of the reconstruction starts, implementing the following two steps:

- 3. Tracklet construction: The tracklets are created by following the hit-to-hit links as it is described above. The geometrical trajectories are fit using a Kalman Filter, described in the Section 2.3, with a χ^2 quality check. Each tracklet is extended in order to collect hits close to its trajectory.

- 4. Tracklet selection: Some of the track candidates can have intersected parts. In this case the longest track is saved, the shortest removed. A final quality check is applied to the reconstructed tracks, including a cut on the minimal number of hits and a cut for low momentum.

5.4 HLT Tracker Efficiency

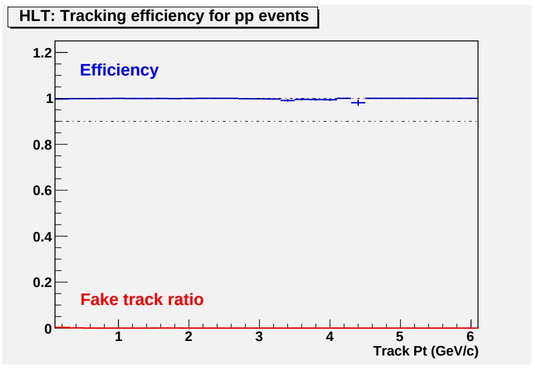


Figure 5.8: Reconstruction performance for proton-proton collisions at 14TeV.

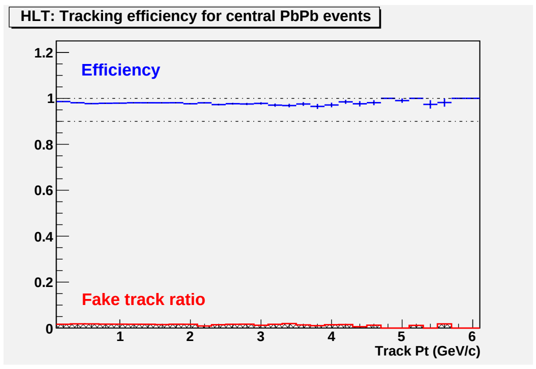


Figure 5.9: Reconstruction performance for central heavy ion collisions at 5TeV.

The performance of the HLT track finder of 99.9 % for proton-proton events and 98.5 % for central Pb-Pb collisions has been verified on simulated events. Corresponding efficiency plots are shown on Fig. 5.8 and 5.9. In addition to the high efficiency, the on-line reconstruction is two orders of magnitude faster than the off-line algorithm used as reference.

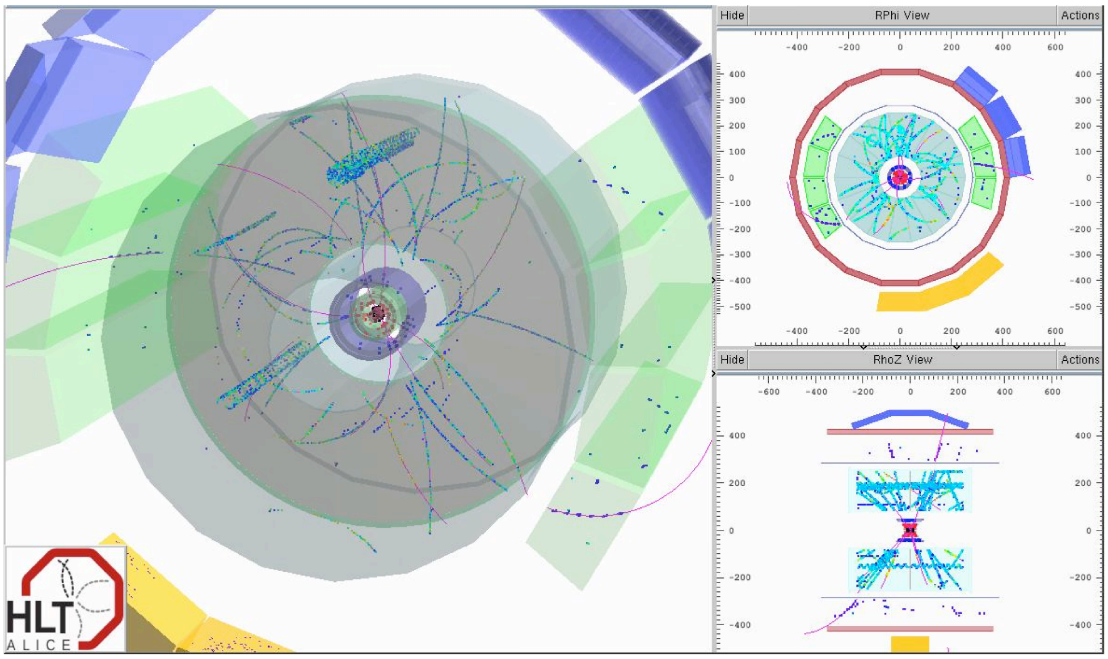


Figure 5.10: The first proton-proton event, obtained by the ALICE High Level Trigger.

The algorithm described has the advantage of a high degree of locality and parallelism. Step one only searches for local neighbors to each hit. It can be done in parallel for all the hits as the result does not depend on the order of processing. Step three follows each tracklets hit by hit, which can also be done in parallel. Only the last selection step of the algorithm is sequential. These locality and parallelism allow for massively parallel implementation as outlined in the following section.

There are many parts of the event reconstruction which are running after the tracker, in particular the primary vertex finder and the V0 finder, which are described in Section 3.7. As it was already noticed among the text the HLT reconstruction was not only tested on simulated data, but it has been running on the real data since 2009. A historical snapshot of one of the first ALICE proton-proton events, obtained by the HLT is shown in Figure 5.10.

5.5 Tracking on GPU hardware

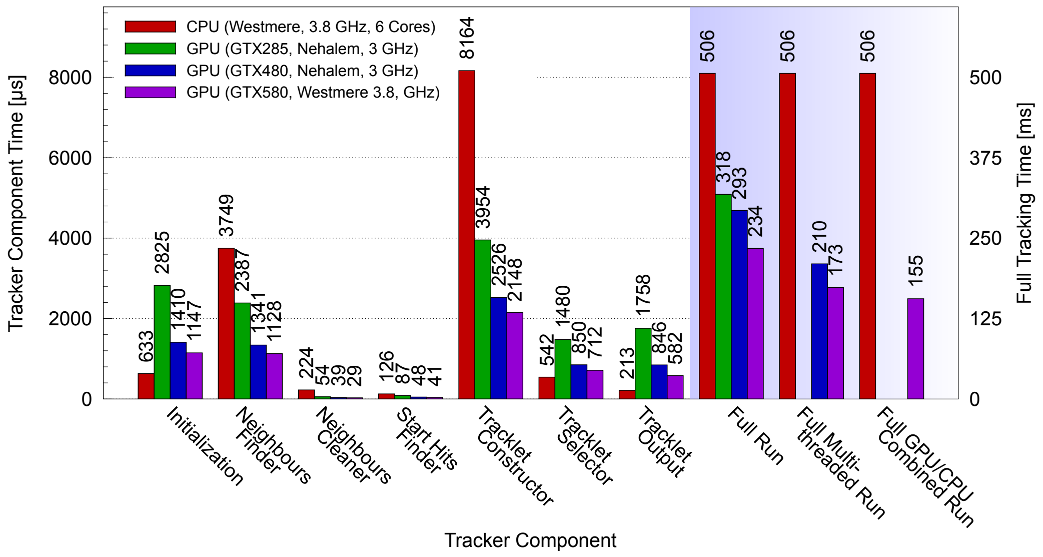


Figure 5.11: CPU and GPU Tracker Performance of the named processing steps for central heavy ion event.

The tracker has been adjusted to run on graphics processing units (GPUs), implementing hundreds ALUs running in parallel.

During the tracking there are 5 steps with a non-negligible requirement of computation time: Initialization, Neighbour Finding, Tracklet Construction, Tracklet Selection and Tracklet Output. The required processing time for the these stages of the tracking algorithm are shown in Figure 5.11.

Of these, the Tracklet Construction contains all the mathematics and most non trivial calculations, while consuming 70 % of the CPU time. It is therefore both the part best suited for running on a GPU and the part with the best opportunity for optimisation. Currently the hardware deployed in the HLT are graphic cards from NVIDIA.

All above-mentioned steps have been ported to CUDA language [31] with the most effort put into the Tracklet Construction. The GPU tracker is implemented in a way, that the main tracking algorithm is contained in common source code for the CPU and GPU versions. Only two specialised wrappers are used for each architecture respectively. The common source file is included in both wrappers and processed by the CPU and GPU compiler. This ensures that changes to the algorithm have to be applied only once.

Since the employed GPU chips show good performance only for single precision calculations, the whole tracker code uses single precision only. An adaptation of the Kalman filter, described in Sec. 4.2 ensures numerical stability to the algorithm in single precision.

To efficiently run the Tracklet Construction on the GPU, a basic understanding of the GPU's architecture is needed. The first GPU hardware used in the HLT was the NVIDIA GTX285 card with a GT200b chip. The chip consists of 30 independent multiprocessors with 8 Arithmetic-Logic Units (ALU) each. Each multiprocessor can handle a vast number of threads in parallel. One should have about 256 concurrent threads running on each multiprocessor for fully exploiting the GPU. The threads running on a multiprocessor are organized in warps of 32 threads each. All threads in one warp can only execute one particular common instruction. If different threads are to execute different instructions, for example due to branching in the code, these operations have to be serialized.

The GPU implementation of the Tracklet Construction has each tracklet processed by a different thread. The problem arising here is caused by different lengths of the tracklets. As all threads within one warp must execute a common instruction, they have to wait for the one thread processing the longest tracklet, even if their current task is already finished. This resulted in the GPU Utilization staying below 20 % for the first implementation (see Fig. 5.12).

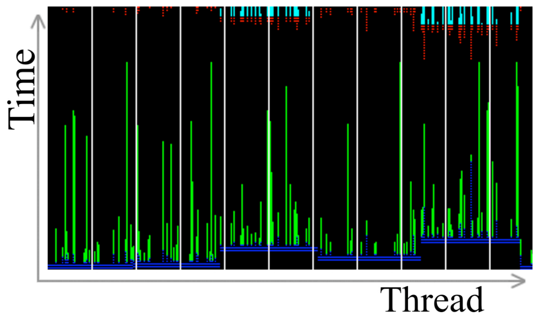


Figure 5.12: GPU utilization without scheduling during Tracklet Construction.[2]

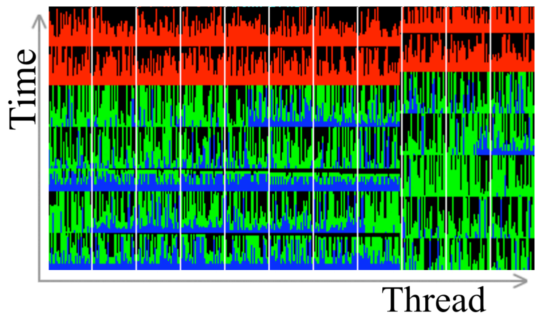


Figure 5.13: GPU utilization with scheduling during Tracklet Construction.[2] [3]

The inefficiency was solved by D. Rohr [29] by introducing a dynamic scheduler. The Tracklet Construction task is split into several subtasks. Within one subtask threads only extrapolate tracklets for a constant amount of rows. Afterward all unfinished tracklets are redistributed among threads and even multiprocessors. To suppress short tracklets in the first way, during the evolution phase it is already checked whether at least three hits will be found for a given start hit. If not, the start hit is ignored. The scheduler works more efficiently when the tracker processes multiple slices in parallel. This ensures that there are always enough threads available for scheduling. By applying these changes the GPU utilization raised to almost 70 % (Fig. 5.13).

Additionally, the memory layout was changed in such a way, that threads which are executed in parallel access consecutive memory addresses. This is done by interleaving the data structures for different threads. The GPU memory controller can coalesce accesses from different threads into one single memory transaction.

Apart from the Tracklet Construction also the Neighbour Finding and Tracklet Selection was ported onto the GPU. The performance of the Neighbours Finder could be significantly

[2]White borders seperate threads of one warp. Colors stand for: black — idling, colored — different states during Tracklet Construction.
[3]The three rightmost threads belong to different multiprocessors and are scheduled separately.

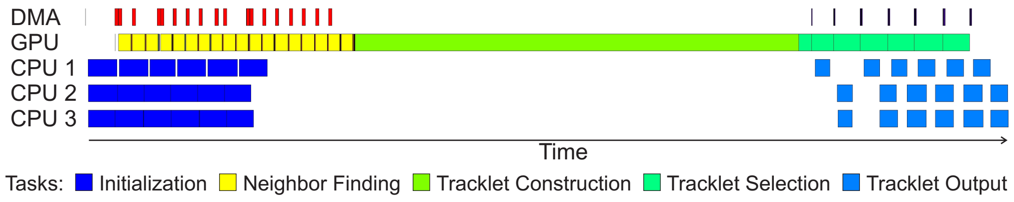

DMA
GPU
CPU 1
CPU 2
CPU 3
Time
Tasks: Initialization Neighbor Finding Tracklet Construction Tracklet Selection Tracklet Output

Figure 5.14: Asynchronous event processing.

improved by caching intermediate data in the fast shared memory of the NVIDIA GPU.

Running the Tracklet Selection on the GPU is necessary since even though the Tracklet Selection is slower as compared to the CPU version (see Fig. 5.11), it greatly decreases the output of the GPU and thus the amount of data that is transferred back to the host. Contrary to all these tasks, the Initialization and Output steps do not involve computation, but instead have lots of random memory reads requiring most data only once. This is not well suited for a GPU, especially considering the additional data transfer required, but can benefit from big and advanced caches of state of the art CPUs and therefore should stay on the CPU.

Keeping the GPU cores operating at full capacity is the main objective. Since 36 TPC slices are handled simultaneously anyway, to allow efficient scheduling the steps are pipelined asynchronously using both, CPU and GPU, while data is transferred via DMA. This way, after having initialized the tracker data structures for the first slice, the CPU can immediately preprocess the next slice while the GPU starts tracking the first one, as can be seen in Figure 5.14. In general three tasks run concurrently: the GPU tracks slice n, slice $n-1$ is transferred to the GPU via DMA, the CPU preprocesses slice $n-2$. In order to get higher performance, multiple CPUs are used for the pipeline.

5.6 Efficiency and Performance of the GPU Tracker

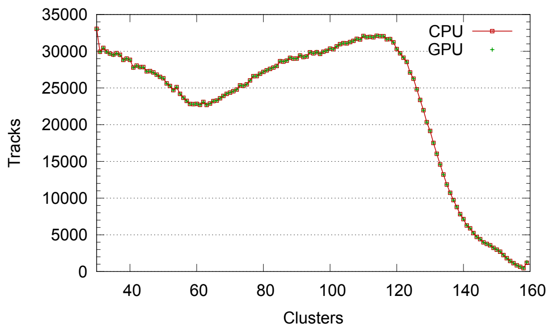


Figure 5.15: Clusters per track distribution for CPU and GPU trackers.

To test the GPU tracker real data and data from Monte-Carlo simulations is used. A comparison shows that the output of the GPU tracker is equal to the output of the CPU tracker. As an example, comparison of the number of TPC clusters per track distributions is presented in Figure 5.15. On the simulated data, the GPU tracker shows the same efficiency as the CPU tracker.

Additionally, a bitwise comparison of the GPU/CPU outputs has been done by D. Rohr. It shows that in heavy ion events only 0,012% of the reconstructed tracks differ. An analysis shows that the difference is caused by rounding errors in floating point arithmetic and therefore is fully acceptable. It can be concluded that the GPU tracker is in no inferior to the CPU version. Use of the graphic cards gives factor of 3 speed-up with respect to 10-core CPU (Fig. 6.14).

The GPU tracker is incorporated to the HLT framework and running on-line since 2011.

The first working version of the GPU tracker was written by the author, then the development has been continued by David Rohr. The author thanks David for providing figures for this and for the previous section.

Conclusions

This thesis presents various algorithms which have been developed for event reconstruction in the CBM and ALICE experiments.

All the developed algorithms are aimed for on-line data processing, where the challenge is to create not only efficient but very fast reconstruction which would match high data rates of the modern High Energy Physics experiments.

The effort to achieve the goal has been exerted in three directions:

- Development of fast mathematics for track and vertex fit (Chapters 2 and 3). For that propose the existing fit algorithms has been improved and simplified using modifications of the Kalman filter method, described in Chapter 1.
- Development of fast track finders to treat the most combinatorial part of the event reconstruction (Chapters 4, 5). Here the Cellular Automaton-based algorithms have been developed.
- Implementation of the developed algorithms on the modern hardware, such as SIMD for CBM and GPU for ALICE, which allows one to perform massive parallel calculations (Chapters 4, 5).

All the developed algorithms have proven their quality and speed and are used (in ALICE experiment) or planned to be used (in CBM experiment) for the on-line data processing.

Bibliography

[1] R. E. Kalman, A new approach to linear filtering and prediction problems, Trans. ASME, Series D, J. Basic Eng. 82 (1960) 3545.

[2] R. Frühwirth et al., Data analysis techniques for high-energy physics. Second edition, Cambridge Univ. Press (2000).

[3] P. Billoir and S. Qian, Fast vertex fitting with a local parametrization of tracks. Nucl. Instr. and Meth. A311 (1992) 139-150.

[4] G. Lutz Billoir, Topological vertex search in collider experiments. Nucl. Instr. and Meth. A337 (1993) 66-94.

[5] R. Frühwirth, P. Kubinec, W. Mitaroff and M. Regler, Vertex reconstruction and track bundling at the LEP collider using robust algorithms. Comp. Phys. Commun. 96 (1996) 189-208.

[6] D. Emeliyanov, I. Kisel, S. Masciocchi, M. Sang and Yu. Vassiliev, Primary vertex reconstruction by Rover. HERA-B note 00-139, 2000.

[7] V. Eiges, D. Emeliyanov, V. Kekelidze, I. Kisel and Yu. Vassiliev, Test of vertex reconstruction and fitting algorithms on $J/\psi \to \mu^+\mu^-$ data. HERA-B note 00-182, 2000.

[8] S. Gorbunov and I. Kisel, Primary vertex fit based on the Kalman filter. CBM-SOFT-note-2006-001, GSI, Darmstadt, 9 January 2006.

[9] S. Gorbunov and I. Kisel, Secondary vertex fit based on the Kalman filter. CBM-SOFT-note-2006-002, GSI, Darmstadt, 14 September 2006.

[10] S. Gorbunov and I. Kisel, An analytic formula for track extrapolation in an inhomogeneous magnetic field. CBM-SOFT-note-2005-001, GSI, Darmstadt, 18 March 2005.

[11] W. Hulsbergen, Decay chain fitting with a Kalman filter. Nucl. Instr. and Meth. A 552 (2005) 566-575.

[12] CBM Collaboration, Compressed Baryonic Matter Experiment. Technical Status Report. GSI, Darmstadt, 2005; 2006 Update (`http://www.gsi.de/documents/DOC-2006-Feb-108-1.pdf`).

[13] I. Kisel, Event reconstruction in the CBM experiment. Nucl. Instr. and Meth. A566 (2006) 85-88.

[14] S. Gorbunov and I. Kisel, Analytic formula for track extrapolation in non-homogeneous magnetic field. Nucl. Instr. and Meth. A559 (2006) 148-152.

[15] W.-M. Yao et al. (Particle Data Group), Review of Particle Physics, J. Phys. G: Nucl. Part. Phys. 33 (2006) 1-1232.

[16] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, “Numerical Recipes in C”, Cambridge University Press, Second edition, 1995.

[17] G. R. Lynch and O. I. Dahl, Approximations to multiple Coulomb scattering. Nucl. Instr. and Meth. B58 (1991) 6-10.

[18] I. Kisel, Event reconstruction in the CBM experiment, Nucl. Instr. and Meth A 566 (2006), pp. 8588.

[19] J. Podolanski and R. Armenteros, Phil. Mag. 45, (1954), 13.

[20] IA-32 Intel Architecture Optimization Reference Manual, Intel, June 2005.

[21] Cell Broadband Engine. Programming Tutorial. IBM, ver. 1.0, October 21, 2005.

[22] Cell Broadband Engine. Programming Handbook. IBM, ver. 1.0, April 19, 2006.

[23] I. Ollmann, AltiVec Tutorial, ver. 1.2, 2002 (`http://www.simdtech.org`).

[24] M.S. Grewal and A.P. Andrews, Kalman Filtering: Theory and Practice using MATLAB. New York, NY.: John Wiley & Sons, 2nd Ed., 2001.

[25] P.G. Kaminski, A.E. Bryson and S.F. Schmidt, Discrete square root filtering: A survey of current techniques, IEEE Trans. Auto. Control, vol. AC-16 (1971) 727-736.

[26] S. Gorbunov, U. Kebschull, I. Kisel, V. Lindenstruth, and W. F. J. Müller, “Fast SIMDized Kalman Filter based track Fit,” *Comp. Phys. Comm.*, vol. 178, no. 5, pp. 374 - 383, Mar. 2008.

[27] The ALICE collaboration, “ALICE - Technical Proposal for A Large Ion Collider Experiment at the CERN LHC,” CERN, Geneve, Rep. CERN-LHCC-95-71; LHCC-P-3, 1995. Available: http://cdsweb.cern.ch/record/293391http://cdsweb.cern.ch/record/293391

[28] The ALICE collaboration, “The ALICE Experiment at the CERN LHC,” *JINST,* vol. 3, no. 08, Aug. 2008. Available: http://dx.doi.org/10.1088/1748-0221/3/08/S08002http://dx.doi.org/10.1088/1748-0221/3/08/S08002

[29] S. Gorbunov, D. Rohr et al. ALICE HLT high speed tracking and vertexing. *Proc.17th Real Time Conference, Lisbon, May 2010*

[30] S. Gorbunov, D. Rohr et al. ALICE HLT High Speed Tracking on GPU . *IEEE Transactions on Nuclear Science*, vol. 58, no. 4, Aug. 2011

[31] NVIDIA Corporation. CUDA C Programming Guide 4.0 (June 2011).

Chapter 6

Zusammenfassung (in German)

Diese Dissertation präsentiert verschiedenen Algorithmen, die für die Echtzeit-Ereignisrekonstruktion im CBM-Experiment der GSI (in Darmstadt) und im ALICE-Experiment am CERN (in Genf) entwickelt wurden.

Obwohl diese Experimente unterschiedlich sind — CBM ist ein Fixed-Target Experiment mit Forward-Geometrie, während ALICE eine typische Collider-Geometrie hat — gibt es bei der Rekonstruktion gemeinsame Aspekte.

Diese Arbeit beschreibt:

— allgemeine Änderungen an der Kalman-Filter-Methode, die bestehende Fit-Algorithmen (auch Anpassungsalgorithmen genannt) beschleunigen, vereinfachen sowie deren numerische Stabilität verbessern.

— Fit-Algorithmen, die für die CBM und ALICE Experimente entwickelt wurden, inklusive einer neuen Methode für die Spurextrapolation in nicht-homogenen Magnetfeldern.

— die entwickelten Algorithmen für die Bestimmung der primären und sekundären Vertices in beiden Experimenten. Insbesondere wird eine Methode zur Rekonstruktion der zerfallenen Teilchen vorgestellt.

— parallelisierte Methoden für die Echtzeit-Spursuche im CBM Experiment.

— parallelisierte Methoden zur Echtzeit-Spursuche im High Level Trigger des ALICE-Experiments.

— die Realisierung der Spurrekonsturtion auf moderner Hardware, insbesondere Vektorprozessoren und GPUs.

Alle vorgestellten Methoden sind vom oder mit direkter Beteiligung des Autors entwickelt worden.

6.1 Die Kalman-Filter-Methode

Die Ereignisrekonstruktion in der Hochenergiephysik umfasst verschiedenen Anpassungsprobleme. Um es kurz zu fassen: es müssen die wahrscheinlichsten Werte von Größen anhand von Messungen dieser Größen errechnet werden. Ein Beispiel ist die Suche nach den Parametern, die die Trajektorie eines Teilchens beschreiben, unter Verwendung der von den Detektoren gelieferten Informationen. Der Kalman-Filter [1, 2] ist eine leistungsfähige Methode, die das

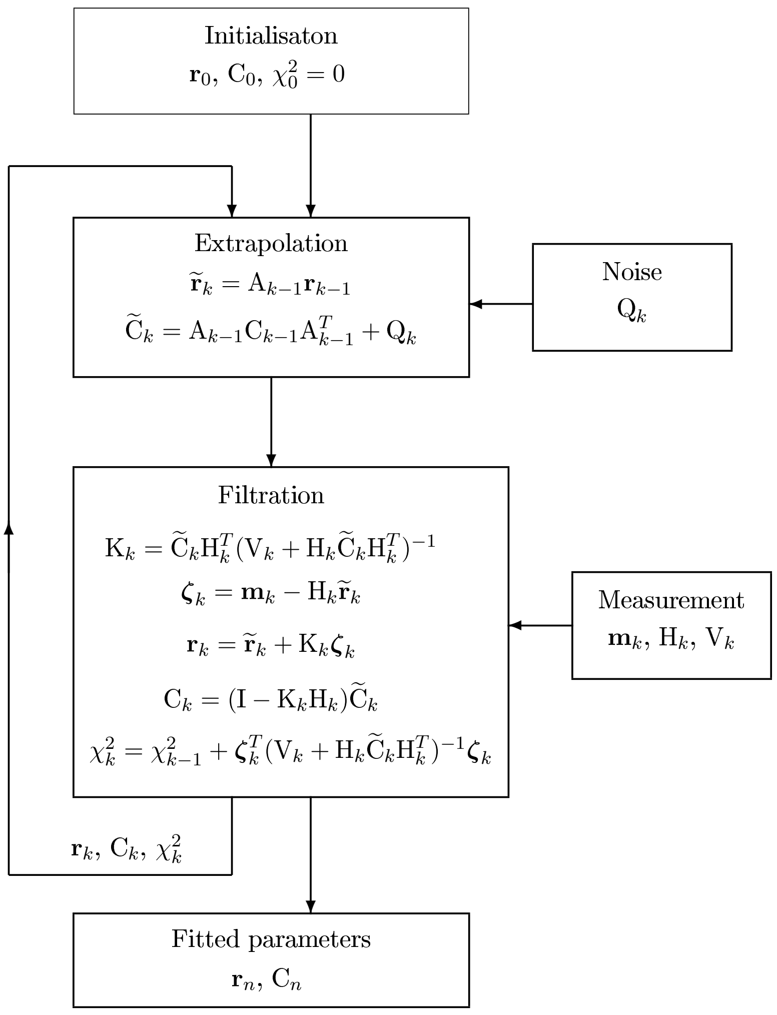

Initialisaton
$\mathbf{r}_0$, C_0, $\chi^2_0 = 0$
Extrapolation
$\widetilde{\mathbf{r}}_k = \mathrm{A}_{k-1}\mathbf{r}_{k-1}$
$\widetilde{\mathrm{C}}_k = \mathrm{A}_{k-1}\mathrm{C}_{k-1}\mathrm{A}^T_{k-1} + \mathrm{Q}_k$
Noise
Q_k
Filtration
$\mathrm{K}_k = \widetilde{\mathrm{C}}_k\mathrm{H}^T_k(\mathrm{V}_k + \mathrm{H}_k\widetilde{\mathrm{C}}_k\mathrm{H}^T_k)^{-1}$
$\boldsymbol{\zeta}_k = \mathbf{m}_k - \mathrm{H}_k\widetilde{\mathbf{r}}_k$
$\mathbf{r}_k = \widetilde{\mathbf{r}}_k + \mathrm{K}_k\boldsymbol{\zeta}_k$
$\mathrm{C}_k = (\mathrm{I} - \mathrm{K}_k\mathrm{H}_k)\widetilde{\mathrm{C}}_k$
$\chi^2_k = \chi^2_{k-1} + \boldsymbol{\zeta}^T_k(\mathrm{V}_k + \mathrm{H}_k\widetilde{\mathrm{C}}_k\mathrm{H}^T_k)^{-1}\boldsymbol{\zeta}_k$
Measurement
$\mathbf{m}_k$, H_k, V_k
$\mathbf{r}_k$, C_k, χ^2_k
Fitted parameters
$\mathbf{r}_n$, C_n

Figure 6.1: Schema der Kalman-Filter-Methode.

Anpassungsproblem in einer sehr allgemeinen Weise löst.

Anpassungsproblem

Zur Formulierung des Anpassungsproblems werden Folgende Begriffe verwendet:

Zustandsvektor $\mathbf{r}^t$ — Dies ist ein reellwertiger Vektor, der die unbekannten Größen des Anpassungsproblems (z.b. die Parameter einer Spur) enthällt. Allgemein kann sich der Zustandsvektor mit der Zeit ($\mathbf{r}^t_0 \to \mathbf{r}^t_1 \to \ldots \to \mathbf{r}^t_n$) auf zufällige Weise verändern:

$$\mathbf{r}^t_k = \mathrm{A}_k\mathbf{r}^t_{k-1} + \boldsymbol{\nu}_k \quad (k = 1\ldots n) \tag{6.1}$$

wobei A_k ein (bekannter) linearer Operator ist, der **Extrapolierer** genannt wird; $\boldsymbol{\nu}_k$ ist eine Zufallsvariable, die **Prozessrauschen** heißt. Das Prozessrauschen hat Erwartungswerk 0 und seine Kovarianzmatrix Q_k ist bekannt.

Messung $\mathbf{m}_k$ — Die Messung ist die gemessene (bekannte) Größe, die vom Zustandsvektor linear abhängig ist:

$$\mathbf{m}_k = \mathrm{H}_k\mathbf{r}^t_k + \boldsymbol{\eta}_k \tag{6.2}$$

wobei H_k ein (bekannter) linearer Operator ist, der **Messungsmodell** heißt. Die Variable $\boldsymbol{\eta}_k$ ist eine Zufallsvariable, die **Messfehler** genannt wird. Der Messfehler hat ebenfalls Erwartungswert 0 und seine Kovarianzmatrix V_k ist bekannt.

Es wird davon ausgegangen, dass alle Zufallsvariablen $\boldsymbol{\nu}_i$, $\boldsymbol{\eta}_j$ unkorreliert sind.

Schätzer $\mathbf{r}_k$ — Dies ist ein Vektor, der den Wert des (unbekannten) Zustandsvektors $\mathbf{r}_k^t$ anhand von gegebenen Messungen einschätzt. Der Schätzer wird **linearer Schätzer** genannt, wenn er linear von den Messungen abhängt.

Das Anpassungsproblem besteht nun darin, **den besten linearen Schätzer $\mathbf{r}_n$ des Zustandsvektors $\mathbf{r}_n^t$ zu finden, der der letzten Messung $\mathbf{m}_n$ entspricht**.

Die Kalman-Filter-Methode

Der Kalman-Filter startet mit einer gewissen Approximation $\mathbf{r} = \mathbf{r}_0$, verfeinert den Schätzer $\mathbf{r}$, indem er die Messungen nacheinander bearbeitet, und liefert nach der Bearbeitung der letzten Messung den besten Schätzer.

Der Anpassungsalgorithmus besteht aus drei Schritten: Initialisierung, Extrapolation und Filtrierung, sehe Abbildung 6.1. Die Schritte der Extrapolation und der Filtrierung werden n–mal nacheinander für jede Messung $\mathbf{m}_k$, $k = 1, \dots n$ wiederholt. Nach der Filtrierung der letzten Messung $\mathbf{m}_n$ entspricht der Schätzer $\mathbf{r}_n$ dem gesuchten besten Schätzer mit der Kovarianzmatrix C_n.

Erweiterungen der Kalman-Filter-Methode

In dieser Arbeit wurden mehrere Änderungen des konventionellen Filtrierungsverfahrens entwickelt. Sie ermöglichen es, die Kalman-Filter-Methode für die Rekonstruktion von zerfallenen Teilchen anzuwenden und den Standardansatz für den Vertex-Fit zu optimieren.

- **Filtrierung mit erweitertem Messmodell** (Sec. 1.4.1). Eine Erweiterung der Kalman-Filter-Methode erlaubt es die Gleichung in Abbildung 6.1 durch eine allgemeinere zu ersetzen.
- **Filtrierung mit einer korrelierten Messung** (Sec. 1.4.2). Um einen einfachen und schnellen Vertexanpasser, der in Abschnitt 3.3 beschrieben wird, zu konstruieren, war es notwendig, die Bedingungen des konventionellen Kalman-Filters für den Fall, dass die Fehler der verschiedenen Messungen untereinander korreliert sind, zu verallgemeinern.
- **Filtrierung mit bestem Schätzer** (Sec. 1.4.3). Gelegentlich ist es notwendig, einen Teil des Zustandsvektors separat anzupassen und dann den angepassten Teil mit dem Rest des Vektors zusammenzufügen. Dieses Problem tritt auf, wenn ein Teilchen in einen bereits rekonstruierten Vertex eingefügt werden soll.
- **Subtraktion einer Messung** (Sec. 1.4.4). Manchmal ist es notwendig, eine falsche Messung, die zuvor in einen Zustandsvektor integriert wurde, zu entfernen. Dieses Problem besteht bei der primären Vertex-Rekonstruktion.
- **Anpassung mit Nebenbedingungen** (Sec. 1.4.5). In einigen Fällen kann das Anpassungsergebnis durch Einführen von Einschränkungen für den Zustandsvektor verbessert werden. In Abschnitt 1.4.5 wird nachgewiesen, dass die Einschränkungen durch den

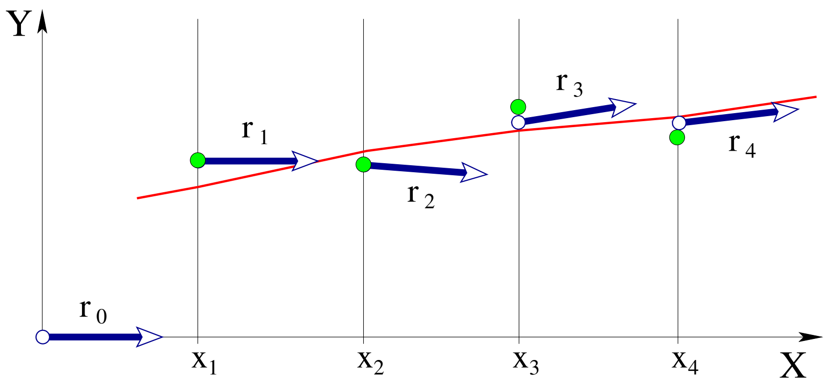


Figure 6.2: Prozess der Track-Fit. Grüne Punkte sind Messungen, weiße Punkte sind Positionen des Zustandsvektors.

Kalman-Filter als gewöhnliche Messungen des Zustandsvektors betrachtet werden können. Diese Nebenbedingungen werden hauptsächlich in der sekundären Vertexrekonstruktion verwendet.

6.2 Track-Fit mit dem Kalman-Filter

Die typische Anwendung des Kalman-Filters in der Hochenergiephysik ist die Bestimmung der Teilchentrajektorien, die **Spuren** oder **Tracks** genannt werden. Das ist eine interessante und nicht-triviale Aufgabe, bei der man alle Vorteile der Methode zu schätzen lernt.

Wenn ein geladenes Teilchen sich in einem Detektor bewegt, wird seine Trajektorie von mehreren physikalischen Effekten beeinflusst, wie z. B. Streuung am Detektormaterial, Energieverlusten und einem nicht-homogenen Magnetfeld. Alle diese Einflüsse können durch das Kalman-Filter-Verfahren berücksichtigt werden, was diese Methode für die Track-Fit unverzichtbar macht.

Die Anwendung des Track-Fits verläuft wie folgt:

Zuerst werden alle Messungen entlang der erwarteten Trajektorie geordnet.

Danach wird der Spurschätzer beliebig initialisiert (**Initialisierungsschritt** des Kalman-Filters).

Nach der Initialisierung der Spur wird das Fit-Verfahren gestartet. Der Anfangsschätzer wird zur ersten Messung extrapoliert und die Mehrfachsteuerung wird zur Kovarianzmatrix (**Extrapolation**) hinzugefügt. Dann wird der Schätzer mit der ersten Messung aktualisiert (**Filtrierung**). Es folgt die Extrapolation auf die Position des zweiten Detektors und der Vorgang wird für die zweite Messung wiederholt, usw.

Das Fit-Verfahren ist schematisch in Abbildung 6.2 gezeigt. Man kann sehen, dass der Anfangsschätzer $\mathbf{r}_0$ weit weg von der tatsächlichen Spur liegt, er wird aber während der Messungsverarbeitung schrittweise verbessert. Der endgültige beste Schätzer $\mathbf{r}_n$ der Spurparameter entsteht nach der Verarbeitung der letzten Messung.

Obwohl sich die tatsächlichen Verhältnisse (das Spurmodell, die Extrapolationsformel, die Messungen und das Rauschen) von verschiedenen Experimenten unterscheiden, bleibt das allgemeine oben beschriebene Schema des Track-Fit immer gleich.

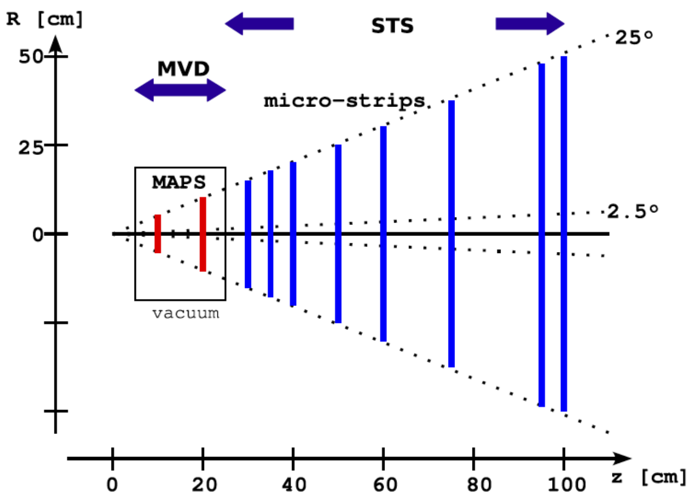


Figure 6.3: STS + MVD Detektorsystem in CBM.

Track-Fit in CBM

CBM ist ein Experiment mit fixem Target, bei dem die geladenen Teilchen aus dem Target innerhalb eines engen Kegels in Strahlrichtung emittiert werden. Um die Teilchen im Raum zu trennen, wird ein starkes Magnetfeld direkt nach dem Target benutzt. Der Detektor ist innerhalb des Magneten platziert, deswegen wird die Rekonstruktion der Ereignisse in der z-Region zwishcen 5 cm und 1 m durchgeführt (siehe Abb. 6.3).

Das verwendete Spurmodell ist typisch für Experimente mit fixem Target:

$$z, \ \{x, y, t_x, t_y, q/p\} \tag{6.3}$$

wobei x, y, z die Koordinaten sind; $t_x \equiv \mathrm{d}x/\mathrm{d}z$, $t_y \equiv \mathrm{d}y/\mathrm{d}z$ die Steigungen und q/p der Quotient von Ladung und Impuls. Die Spur wird entland der z-Koordinate parametrisiert. Diese Parametrisierung ist für die Spurextrapolation sehr günstig, da Target und die Detektoren entlang der z-Achse platziert sind.

Der Track-Fit wird nach dem im Abschnitt 6.2 beschriebenen Schema durchgeführt. Aufgrund der Geometrie des Magneten ist das Magnetfeld stark inhomogen, was eine einfache Extrapolation ausschließt. Daher wurde eine spezielle analytische Extrapolationsformel entwickelt, die das Variieren der Rechnungskomplexität und damit auch die CPU-Zeit in Übereinstimmung mit der erforderlichen Genauigkeit ermöglicht. Die Standard-Runge-Kutta-Extrapolation wurde ebenfalls implementiert und wird als Referenz verwendet. In Tabelle 6.1 ist zu sehen, dass die entwickelte Formel, Analytic-* genannt, die gleiche Qualität wie das Runge-Kutta-Verfahren aufweist.

Ein großer Vorteil der entwickelten Formel ist, dass die Berechnungen für mehrere Spuren parallel mit SIMD-CPU-Registern durchgeführt werden können, was mit dem iterativen Runge-Kutta-Extrapolator nicht möglich ist. Der beschriebene Track-Fit stellt den Kern der Online-Rekonstruktion im CBM Experiment dar (Absch. 4.2).

Track-Fit im ALICE High Level Trigger

Der wichtigste Spur-Detektor in ALICE ist der Time Projection Chamber (TPC) Detektor, der in 36 trapezförmige Auslesungssektoren unterteilt ist. Die Geometrie eines TPC-Sektors

	Residuals					Pulls				
Method	$\delta p/p$	x	y	t_x	t_y	q/p	x	y	t_x	t_y
Runge-Kutta–4	0.64	27	24	1.5	1.5	1.17	1.05	1.01	1.02	1.00
Analytic–3	0.64	27	24	1.5	1.5	1.18	1.05	1.00	1.02	1.00
Analytic–2	0.68	27	24	1.5	1.5	1.30	1.08	1.01	1.03	1.00
Analytic–1	0.94	30	25	1.5	1.5	1.90	1.37	1.03	1.10	1.02
Analytic–light	0.64	27	24	1.5	1.5	1.19	1.05	1.00	1.02	1.00
Analytic–central	2.49	38	25	1.7	1.5	3.77	2.23	1.03	1.33	1.00

Table 6.1: Residuen ($\delta p/p_{[\%]}$, $(x,\ y)_{[\mu m]}$, $(t_x,\ t_y)_{[\cdot 10^{-3}]}$) und normalisierte Residuen (pulls) der Spurparameter nach dem Kalman-Filter mit verschiedenen Extrapolations-Methoden.

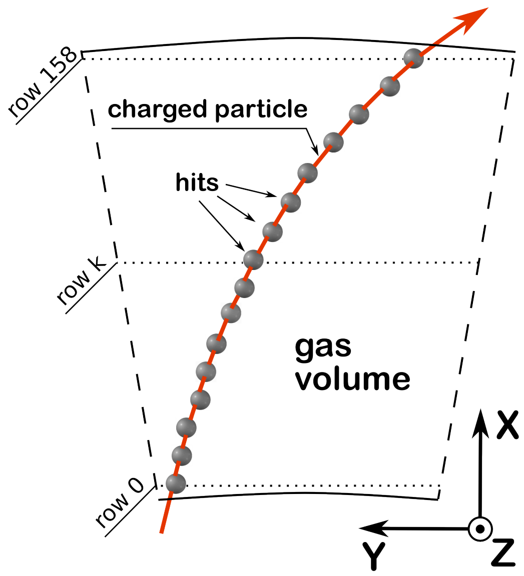


Figure 6.4: Geometrie eines TPC-Sektor.

ist in Abbildung 6.4 dargestellt. Die TPC-Messungen sind Raumpunkte. Das ALICE-Magnetfeld ist entlang der z-Achse orientiert. Da das Magnetfeld innerhalb der TPC fast konstant ist, sind die Teilchentrajektorien Helices.

Das ALICE-Spurmodell wird unter Berücksichtigung der zylindrischen Geometrie des Experiments gewählt. Für jeden TPC-Sektor haben die Spuren eine lokale Parametrisierung an bestimmter x-Position mit 5 variablen Parametern:

$$x,\ \{y, z, \sin\phi, \lambda, q/p_t\} \tag{6.4}$$

x, y, z sind die Koordinaten; ϕ is der Polarwinkel; λ is der Tangenz des azimutalen Winkels; q/p_t ist der vorzeichenbehaftete inverse Transversalimpuls.

Der Offline-Track-Fit in ALICE wird mit dem konventionellen Kalman-Filtern durchgeführt. In Tabelle 6.2 kann man sehen, dass es einen signifikanten systematischen Fehler im rekonstruierten Transversalimpuls (p_t) gibt, der der wichtigste physikalische Parameter ist. Für den ALICE High Level Trigger ist ein verbesserter Track-Fit implementiert.

Die Anpassung wurde verbessert, wie im Abschnitt 2.3 beschrieben. Insbesondere wurde

	q/p_t	y	z	$\sin\phi$	λ
standard fit	1.29(+0.368)	1.23	0.93	1.22	0.71
new HLT fit	1.02(+0.001)	1.01(+0.003)	1.03(+0.013)	1.01(-0.010)	1.03(-0.019)

Table 6.2: Pulls der neuen HLT Track-Fit: Wert (±systematische Fehler).

eine Korrektur zweiten Grades entwickelt, die sehr allgemein ist und für alle linearisierten Gleichung in jedem Track-Fit angewendet werden kann. In Tabelle 6.2 ist zu sehen, dass die systematischen Fehler der gefundenen Parameter nach der Verbesserung nahezu ideal sind.

6.3 Rekonstruktionen von Vertices und zerfallenen Teilchen

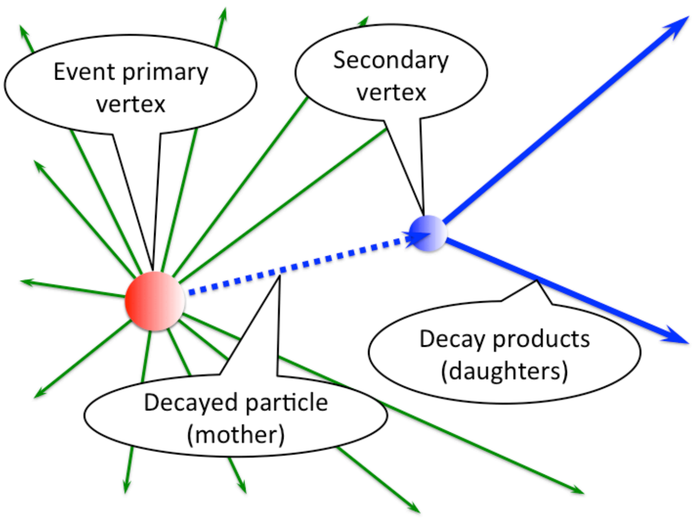


Figure 6.5: Schematische Darstellung von Ereignisvertices und zerfallenden Teilchen.

Die Rekonstruktion eines physikalischen Ereignisses besteht nicht nur aus der Spurrekonstruktion sondern umfasst auch die vollständige Suche und Rekonstruktion des primären Vertex sowie der sekundären Vertices und der Trajektorien der zerfallenen Teilchen (Abb. 6.5).

Schnelle Rekonstruktion der primären und sekundären Vertices

Das Ziel des Vertex-Fits ist die Bestimmung der Vertexposition und der zugehörigen Kovarianzmatrix unter Verwendung der vorhandenen Teilchenspuren. Beim sekundären Vertex-Fit müssen außer den Vertex-Parametern auch alle Parameter der zum Vertex gehörende Spuren sowie die Kovarianzmatrix berechnet werden.

Der modifizierte Kalman-Filter (Absch. 3.2.1-3.4) zeigt eine Verbesserung gegenüber dem konventionellem Ansatz des Kalman-Filters. Die Menge der Berechnungen wurde erheblich reduziert. Insbesondere benutzen die entwickelten Algorithmen eine einzelne Division anstatt der Inversion einer 5×5-Matrix im Standardansatz. Das Vermeiden von Matrixinversionen verbessert auch die Robustheit gegenüber Rundungsfehlern.

Rekonstruktion der zerfallenen Teilchen

Abschnitt 3.5 beschreibt eine neue Methode zur Rekonstruktion der Parameter von zerfallenen Teilchen und der zugehörigen Kovarianzmatrix. Der Algorithmus verwendet eine kanonische Teilchenparametrisierung $(x, y, z, p_x, p_y, p_z, E, s)^1$, wodurch er unabhängig von der Geometrie des Detektorsystems ist.

Die gewählte Parametrisierung ist eine natürliche Wahl und sehr gut für die physikalische Analyse geeignet. Sie enthält alle notwendigen Informationen über die Teilchen sowohl am Erzeugungsvertex als auch am Zerfallsvertex. Daher ist die entwickelte Methode auch geeignet, um Vertices zu bestimmen.

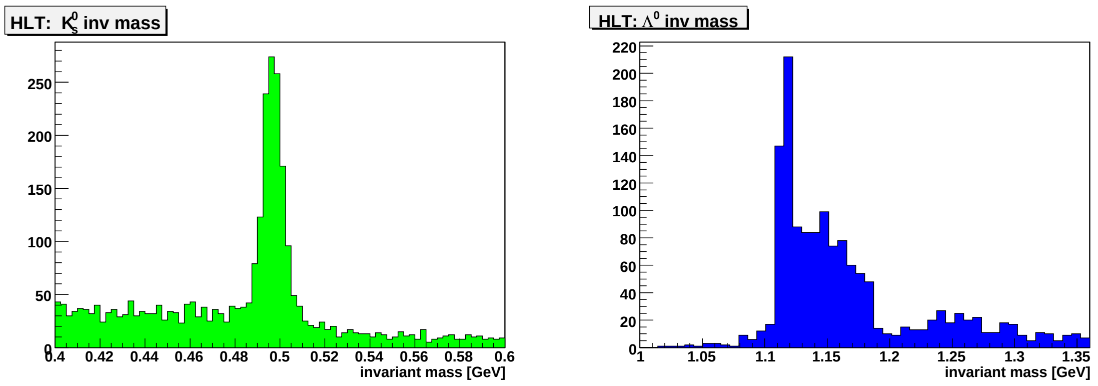
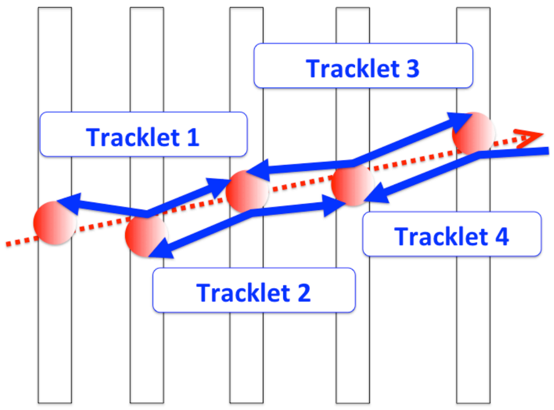


Figure 6.6: ALICE HLT K^0_s und Λ^0 Finder. Reale Daten, run 00010480 (2009).

Die entwickelten Algorithmen zur Suche nach Vertices und zerfallenen Teilchen wurden erfolgreich in den CBM und ALICE Experimenten implementiert (Absch. 3.6, 3.7). Ein Beispiel für reale Datenanalyse ist in Abbildung 6.6 dargestellt.

6.4 Online-Ereignisrekonstruktion im CBM-Experiment



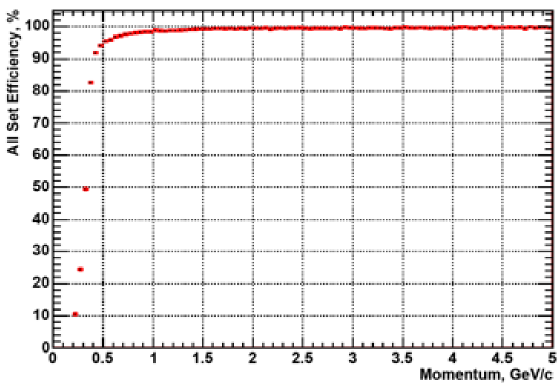


Figure 6.7: Darstellung des zellulären Automatenalgorithmus.

Figure 6.8: Effizienz der Spurrekonstruktion als Funktion des Impulses.

[1]Der Parameter $s = \frac{l}{p}$ ist die Länge der Teilchentrajektorie normiert auf den Teilchenimpuls.

Die hohe Spurdichte (im Durchschnitt 500 Spuren im inneren Detektor in einer typischen zentralen Au + Au Kollision) zusammen mit dem nicht-homogenen Magnetfeld macht die Rekonstruktion der Ereignisse im CBM-Experiment sehr kompliziert.

Der CBM-Algorithmus zur Spursuche basiert auf der Methode des zellulären Automaten. Er erstellt kurze Drei-Hit Spurabschnitte (Tracklets) in benachbarten Detektorebenen und verknüpft diese dann zu vollständigen Spuren (Abb. 6.7). Der Track-Fit (Absch. 2.2) wird bereits auf Ebene der Erstellung des Tracklets angewendet. Dadurch ist die Geschwindigkeit des Fits sehr wichtig für die Online-Datenverarbeitung.

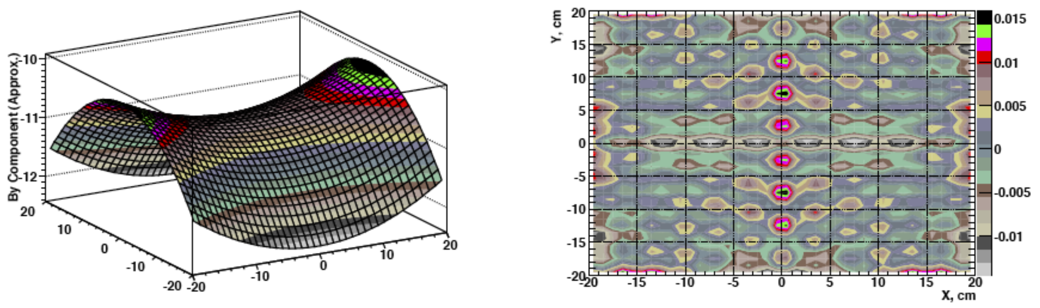


Figure 6.9: Die bedeutendste (By) Komponente des magnetischen Feldes in der Mitte des Detektorsystems ($z =\ 50\,\mathrm{cm}$) unter Verwendung der Polynomapproximation (links) und der Unterschied zwischen zwei alternativen Felddarstellungen (rechts).

Kapitel 4 beschriebt die am Algorithmus vorgenommenen Optimierungen:

1. Der Zugriff auf eine große Magnetfeldkarte konnte durch die Anwendung einer polynomialen Approximation des Feldes (Abb. 6.9) vermieden werden.

2. Der Algorithmus wurde von doppelter auf einfache Rechengenauigkeit umgestellt. Verbesserungen am Kalman-Filter garantieren die numerische Stabilität und konnten die Geschwindigkeit beträchtlich steigern.

3. Der Algorithmus wurde vektorisiert, um die SIMD-Einheit des Pentium 4 Prozessors zu verwenden.

4.-5. Der Algorithmus wurde auf den Cell-Prozessor [21, 22] portiert, der als ein Kandidat für die Online-Hardware betrachtet wird.

Stage	Description	Time/track	Speed-up factor
	Initial scalar version	12 ms	
1	Approximation of the magnetic field	240 μs	50
2	Optimisation of the algorithm	7.2 μs	35
3	Vectorisation	1.6 μs	4.5
4	Porting to SPE	1.1 μs	1.5
5	Parallelisation on 16 SPEs (2 Cells)	0.1 μs	10
	Final SIMDised version	0.1 μs	120000

Table 6.3: Zusammengefasste Stadien des Optimierungsverfahrens.

Tabelle 6.3 summiert alle Ergebnisse und listet eine Zeit-Analyse und die Beschleunigungsfaktoren nach jedem Entwicklungsstadium auf. Man kann sehen, dass der vektorisierte Fit um das 10.000–fache schneller als das Original ist. Insgesamt wurde der Kalman-Filter um bis zu einem Faktor 120.000 beschleunigt.

6.5 Online-Ereignisrekonstruktion in ALICE High Level Trigger

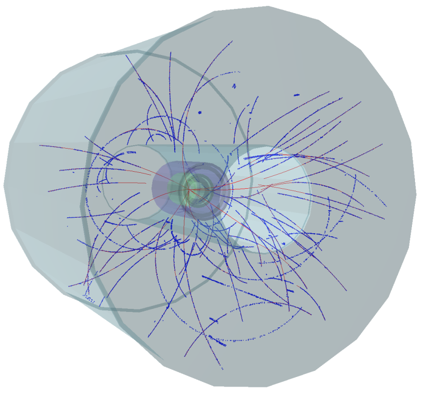


Figure 6.10: Reales Proton-Proton-Ereignis, rekonstruiert vom HLT (2009).

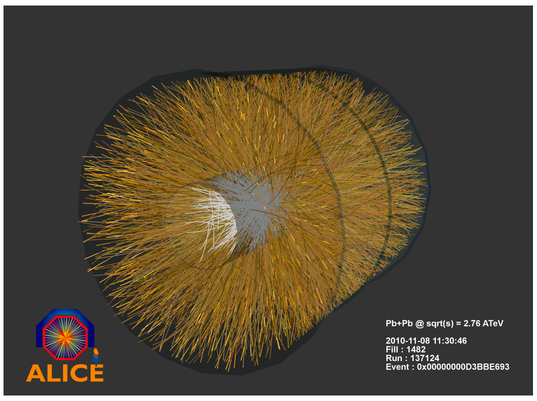


Figure 6.11: Reales Schwerionen-Ereignis, rekonstruiert vom HLT (2010).

Der ALICE High Level Trigger bearbeitet Proton-Proton-Kollisionen bei 2 kHz und Schwerionen-Kollisionen bei 300 Hz; ein Durchschnitt von 25 Spuren in Proton-Proton-Ereignissen und bis zu 25.000 Spuren in Schwerionen-Ereignissen entspricht einem eingehenden Datenstrom von bis zu 30 GB/s. Abbildungen 6.10 und 6.11 zeigen die Komplexität der beiden Typen von Ereignissen. Die Rekonstruktion erfolgt in Echtzeit.

Der HLT-Tracker basiert auf der Methode des zellulären Automaten [18]. Neben der hohen Effizienz (Abb. 6.12), ist die Online-Rekonstruktion um zwei Grössenordnungen schneller als der Offline-Algorithmus, der als Referenz verwendet wurde. Abbildung 6.13 zeigt, dass der Algorithmus 130 μs pro Spur unabhängig von der Spuranzahl im Detektor erfordert, also der kombinatorische Teil des Algorithmus optimal gebaut ist.

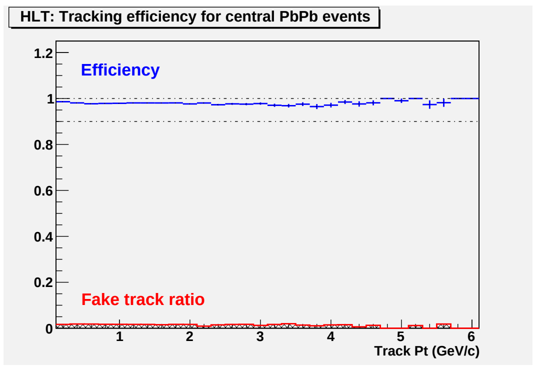


Figure 6.12: Rekonstruktionsleistung für zentrale Schwerionen-Kollisionen bei 5 TeV.

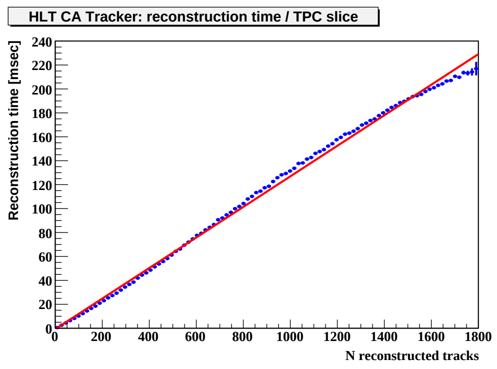


Figure 6.13: Rekonstruktionszeit auf CPU.

Der Tracker ist für die Verarbeitung von Schwerionen-Ereignissen ausgelegt, deswegen wurde er für die Anwendung auf Grafikkarten (GPUs) angepasst, in denen mehr als hundert ALUs parallel laufen. Die Nutzung der Grafikkarten gibt eine weitere 3-fache Beschleunigung

(Abb. 6.14). Der GPU-Tracker ist in das HLT-Framework integriert und läuft dort seit 2011.

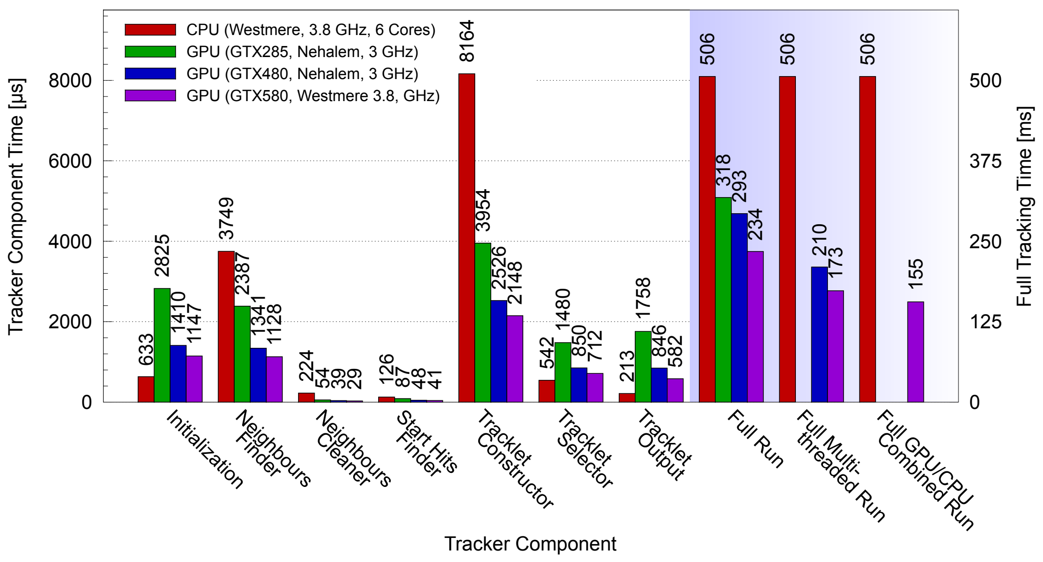


Figure 6.14: CPU-und GPU Leistungen der genannten Verarbeitungsschritten für zentrale Schwerionen-Ereignisse.

Curriculum Vitae

Sergey Gorbunov

Date of birth: January 23, 1976
Place of birth: Omsk, Russia
E-mail: sergey.gorbunov@fias.uni-frankfurt.de

Education: MSc. in Mathematics from the Omsk State University, Russia, 1998.
MSc. Thesis under Prof. R.Faizullin "Ring recognition method based on the elastic neural net".

Employment:

1998-2006	: Scientist at JINR (Dubna, Russia).
2002-2006	: Guest scientist at DESY (Zeuthen, Germany).
2006-2007	: Scientist at GSI (Darmstadt, Germany).
2007-2010	: Ph.D. student at KIP (Heidelberg, Germany).
2010-present	: Scientist at FIAS (Frankfurt, Germany).

Fields of Research and Professional Experience:

Data reconstruction and analysis in high energy physics: event reconstruction; track and vertex fit based on the Kalman filter; fast on-line algorithms; detector alignment.

Participation in Experiments and Projects:

1. **COMPASS experiment**: Study of hadron structure and hadron spectroscopy with high intensity muon and hadron beams (CERN, Geneva, Switzerland, invited by Prof. S. Paul).
2. **H1 experiment**: Measure the structure of the proton, study fundamental interactions between particles, search for physics beyond the Standard Model of the elementary particles (DESY Zeuthen, Germany, invited by Prof. M. Klein).
3. **CBM experiment**: Investigation of the properties of highly compressed baryonic matter as it is produced in nucleus-nucleus collisions (GSI, Darmstadt, Germany, invited by Prof. P. Senger).
4. **ALICE experiment**: Dedicated heavy-ion detector to exploit the unique physics potential of nucleus-nucleus interactions at LHC energies.(CERN, Geneva, Switzerland, invited by Prof. V. Lindenstruth).

Selected publications:

1. S. Gorbunov, I. Kisel and V. Tretyak, **Ring recognition method based on the elastic neural net**, ***Computing in High Energy and Nuclear Physics***, CHEP'01, Beijing, China, September 3-7, 2001, "Beijing 2001, CHEP", pp. 162-163.
2. Antoniou, S.V. Gorbunov, V.V. Ivanov, I.V. Kisel and E.V. Konotopskaya, **Elastic neural nets for the traveling salesman problem**, ***Journal of Computational Methods in Sciences and Engineering***, 2 (2002) 111-115.
3. S. Gorbunov and I. Kisel, **Elastic neural net for stand-alone RICH ring finding**, ***Nucl. Instr. and Meth. A*** 559, Proceedings of the X International Workshop on Advanced Computing and Analysis Techniques in Physics Research - ACAT'05, (2006), 139-142, 05/05.
4. S. Gorbunov and I. Kisel, **Analytic formula for track extrapolation in non-homogeneous magnetic field**, ***Nucl. Instr. and Meth. A*** 559, Proceedings of the X International Workshop on Advanced Computing and Analysis Techniques in Physics Research - ACAT'05, (2006), 148-152, 05/05.
5. CBM Collaboration, **Compressed Baryonic Matter Experiment. Technical Status Report**, GSI, Darmstadt, 2005
6. S. Gorbunov, I. Kisel, **Reconstruction of decayed particles based on the Kalman filter**, ***CBM-SOFT-note-2007-003***, GSI, Darmstadt, 7 May 2007
7. S. Gorbunov, U. Kebschull, I. Kisel, V. Lindenstruth, and W.F.J. Mueller, **Fast SIMDized Kalman Filter based track Fit**, ***Comp. Phys. Comm.***, vol. 178, no. 5, pp. 374 - 383, Mar. 2008.
8. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **First proton–proton collisions at the LHC as observed with the ALICE detector: measurement of the charged-particle pseudorapidity density at $\sqrt{s}$=900**, ***Eur Phys J. C*** (2010) 65: 111-125
9. S. Gorbunov, D.Rohr et al., **ALICE HLT High Speed Tracking on GPU**, ***IEEE Transactions on Nuclear Science***, vol. 58, no. 4, Aug. 2011.

List of publications:

1. S. Gorbunov, I. Kisel, E. Konotopskaya and R. Faizullin, **Comparison of the guaranteed smoothing method and the elastic net method on the travelling salesman problem in plane**, ***JINR Communication*** P5-97-258 (in Russian), Dubna, 1997.
2. S. Gorbunov, I. Kisel and V. Tretyak, **Ring recognition method based on the elastic neural net**, ***COMPASS note*** 1999-3.
3. S. Gorbunov, I. Kisel and V. Tretyak, **Progress report on ring reconstruction with the elastic neural net**, ***COMPASS note*** 1999-18.
4. S. Gorbunov and I. Kisel, **Slow track reconstruction using space points**, ***COMPASS note*** 2000-7.
5. I. Antoniou, S.V. Gorbunov, V.V. Ivanov, I.V. Kisel and E.V. Konotopskaya, **Elastic neural nets for the traveling salesman problem**, ***MTCP'00 conference***, Dubna, 2000.
6. S. Gorbunov, I. Kisel and V. Tretyak, **Ring recognition method based on the elastic neural net**, ***Computing in High Energy and Nuclear Physics***, CHEP'01, Beijing, China, September 3-7, 2001, "Beijing 2001, CHEP", pp. 162-163.
7. Antoniou, S.V. Gorbunov, V.V. Ivanov, I.V. Kisel and E.V. Konotopskaya, **Elastic neural nets for the traveling salesman problem**, ***Journal of Computational Methods in Sciences and Engineering***, 2 (2002) 111-115.
8. S. Gorbunov and I. Kisel, **Elastic neural net for stand-alone RICH ring finding**, ***Nucl. Instr. and Meth. A*** 559, Proceedings of the X International Workshop on Advanced Computing and Analysis Techniques in Physics Research - ACAT'05, (2006), 139-142, 05/05.
9. S. Gorbunov and I. Kisel, **Analytic formula for track extrapolation in non-homogeneous magnetic field**, ***Nucl. Instr. and Meth. A*** 559, Proceedings of the X International Workshop on Advanced Computing and Analysis Techniques in Physics Research - ACAT'05, (2006), 148-152, 05/05.
10. CBM Collaboration, **Compressed Baryonic Matter Experiment. Technical Status Report**, GSI, Darmstadt, 2005
11. S. Gorbunov, I. Kisel, **An analytic formula for track extrapolation in an inhomogeneous magnetic field**, ***CBM-SOFT-note-2005-001***, GSI, Darmstadt, 18 March 2005.
12. S. Gorbunov, I. Kisel, I. Vassiliev, **Analysis of D0 meson detection in Au+Au collisions at 25 AGeV**, ***CBM-PHYS-note-2005-001***, GSI, Darmstadt, 23 June 2005.
13. S. Gorbunov, I. Kisel, **Elastic net for standalone RICH ring finding**, ***CBM-SOFT-note-2005-002***, GSI, Darmstadt, 22 September 2005.

14. S. Gorbunov, I. Kisel, **Primary vertex fit based on the Kalman filter**, ***CBM-SOFT-note-2006-001***, GSI, Darmstadt, 9 January 2006.
15. S. Gorbunov, I. Kisel, **Secondary vertex fit based on the Kalman filter**, ***CBM-SOFT-note-2006-002***, GSI, Darmstadt, 14 September 2006.
16. S. Gorbunov, U. Kebschull, I. Kisel, V. Lindenstruth and W.F.J. Müller, **Fast SIMDized Kalman filter based track fit**, ***CBM-SOFT-note-2007-001***, GSI, Darmstadt, 22 January 2007.
17. S. Gorbunov, I. Kisel, **Reconstruction of decayed particles based on the Kalman filter**, ***CBM-SOFT-note-2007-003***, GSI, Darmstadt, 7 May 2007
18. S. Gorbunov, U. Kebschull, I. Kisel, V. Lindenstruth, and W.F.J. Mueller, **Fast SIMDized Kalman Filter based track Fit**, ***Comp. Phys. Comm.***, vol. 178, no. 5, pp. 374 - 383, Mar. 2008.
19. S. Gorbunov, D. Rohr et al., **ALICE HLT High Speed Tracking on GPU**, ***IEEE Transactions on Nuclear Science***, vol. 58, no. 4, Aug. 2011.

As a collaborator of the H1 experiment:

20. V. Andreev, ..., S. Gorbounov et al. (H1 Collaboration), **Isolated electrons and muons in events with missing transverse momentum at HERA**, ***Phys. Lett. B*** 561 (2003) 241 , 01/03
21. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Diffractive Photoproduction of J/psi Mesons with Large Momentum Transfer at HERA**, ***Phys Lett B*** 568 (2003) 205-218 , 06/03
22. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Multi-Electron Production at High Transverse Momenta in ep Collisions at HERA**, ***Eur Phys J C*** 31 (2003) 17 , 07/03
23. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Single Top Quark Production in ep Collisions at HERA**, ***Eur. Phys. J. C*** 33 (2004) 9 , 10/03
24. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Muon Pair Production in ep Collisions at HERA**, ***Phys. Lett. B*** 583 (2004) 28 , 11/03
25. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Inclusive Dijet Production at Low Bjorken-x in Deep Inelastic Scattering**, ***Eur. Phys. J. C*** 33 (2004) 477 , 10/03
26. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Dijet Production at Low Q^2 at HERA**, ***Eur. Phys. J. C*** 37 (2004) 141-159 , 01/04
27. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Squark Production in R-Parity Violating Supersymmetry at HERA**, ***Eur. Phys. J. C*** 36 (2004) 425-440 , 03/04

28. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Anti-Deuteron Production and a Search for Heavy Stable Charged Particles at HERA**, ***Eur. Phys. J. C*** 36 (2004) 413-423 ,   03/04
29. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Evidence for a Narrow Anti-Charmed Baryon State**, ***Phys. Lett. B*** 588 (2004) 17-28 ,   03/04
30. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Forward pi^0 Production and Associated Transverse Energy Flow in Deep-Inelastic Scattering at HERA**, ***Eur. Phys. J. C*** 36 (2004) 441-452 , 04/04
31. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of the Proton Structure Function F2 at Low Q2 in QED Compton Scattering at HERA.**, ***Phys Lett B*** 598 (2004) 159-171, 06/04
32. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for bosonic stop decays in R-parity violating supersymmetry in e^+ p collisions at HERA**, ***Phys Lett B*** 599 (2004) 159-172 ,   05/04
33. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Prompt Photon Cross Sections in Photoproduction at HERA**, ***Eur. Phys. J. C*** 38 (2005) 437-445 ,   07/04
34. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **A General Search for New Phenomena in ep Scattering at HERA**, ***Phys Lett B*** 602 (2004) 14-30 ,   08/04
35. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Inclusive Production of D^+, D^0, D_s^+ and D^*+ Mesons in Deep Inelastic Scattering at HERA**, ***Eur. Phys. J. C*** 38 (2005) 447-459 ,   08/04
36. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of F_2^{c\bar{c}} and F_2^{b\bar{b}} at High Q^2 using the H1 Vertex Detector at HERA**, ***Eur. Phys. J. C*** 40 (2005) 349-359 ,   11/04
37. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Light Gravitinos in Events with Photons and Missing Transverse Momentum at HERA**, ***Phys. Lett. B*** 616 (2005) 31-42 ,   01/05
38. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **A Direct Search for Stable Magnetic Monopoles Produced in Positron-Proton Collisions at HERA**, ***Eur. Phys. J. C*** 41 (2005) 133-141 ,   01/05
39. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Dijet Cross Sections in ep Interactions with a Leading Neutron at HERA**, ***Eur. Phys. J. C*** 41 (2005) 273-286 ,   01/05
40. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Beauty Production at HERA Using Events with Muons and Jets**, ***Eur. Phys. J. C*** 41 (2005) 453-467 ,   02/05
41. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Charm and Beauty Photoproduction at HERA using D* mu Correlations**, ***Phys. Lett. B*** 621 (2005) 56-71 ,   03/05

42. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Deeply Virtual Compton Scattering at HERA**, ***Eur. Phys. J. C*** 44 (2005) 1-11 , 05/05
43. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Leptoquark Bosons in ep Collisions at HERA**, ***Phys. Lett. B*** 629 (2005) 9-19 , 06/05
44. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **A Determination of Electroweak Parameters at HERA**, ***Phys. Lett. B*** 632 (2006) 35-42 , 07/05
45. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of F_2^{c\bar{c}} and F_2^{b\bar{b}} at Low Q^2 and x using the H1 Vertex Detector at HERA**, ***Eur. Phys. J. C*** 45 (2006) 23-33 , 07/05
46. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Forward Jet Production in Deep Inelastic Scattering at HERA**, ***Eur. Phys. J. C*** 46 (2006) 27-42 , 08/05
47. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Elastic J/Psi Production at HERA**, ***Eur. Phys. J. C*** 46 (2006) 585-603 , 10/05
48. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Event Shape Variables in Deep-Inelastic Scattering at HERA**, ***Eur. Phys. J. C*** 46 (2006) 343-356 , 12/05
49. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **First Measurement of Charged Current Cross Sections at HERA with Longitudinally Polarised Positrons**, ***Phys. Lett. B*** 634 (2006) 173, 12/05
50. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Photoproduction of Dijets with High Transverse Momenta at HERA**, ***Phys. Lett. B*** 639 (2006) 21, 03/06
51. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Diffractive Photoproduction of Rho Mesons with Large Momentum Transfer at HERA**, ***Phys. Lett. B*** 638 (2006) 422 , 03/06
52. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Tau Lepton Production in ep Collisions at HERA**, Submitted to ***Eur. Phys. J. C*** 48, (2006) 699-714, 04/06
53. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Doubly-Charged Higgs Boson Production at HERA**, ***Phys. Lett. B***, 638 (2006) 432 , 04/06
54. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of Charm and Beauty Dijet Cross Sections in Photoproduction at HERA using the H1 Vertex Detector**, ***Eur. Phys. J. C*** 47, (2006) 597-610, 05/06
55. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for a Narrow Baryonic Resonance Decaying to $K^0_s p$ or $K^0_s \bar{p}$ in Deep Inelastic Scattering at HERA**, ***Phys. Lett. B*** 639 (2006) 202, 04/06

56. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Diffractive Deep-Inelastic Scattering with a Leading Proton at HERA**, ***Eur. Phys. J. C*** 48, (2006) 749-766, 06/06
57. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement and QCD Analysis of the Diffractive Deep-Inelastic Scattering Cross Section at HERA**, ***Eur. Phys. J. C*** 48, (2006) 715-748, 06/06
58. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Inclusive D*-Meson Cross Sections and D*-Jet Correlations in Photoproduction at HERA**, ***Eur. Phys. J. C*** 50 (2007) 251 , 08/06
59. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Diffractive Open Charm Production in Deep-Inelastic Scattering and Photoproduction at HERA**, ***Eur. Phys. J. C*** 50 (2007) 1 , 10/06
60. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Production of D*-Mesons with Dijets in Deep-Inelastic Scattering at HERA**, ***Eur. Phys J. C*** 51 (2007) 271 , 01/07
61. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Lepton Flavour Violation in ep collisions at HERA**, ***Eur Phys J. C*** 52 (2007) 833 , 03/07
62. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Tests of QCD Factorisation in the Diffractive Production of Dijets in Deep-Inelastic Scattering and Photoproduction at HERA**, ***Eur Phys J. C*** 51 (2007) 549 , 03/07
63. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Search for Baryonic Resonances Decaying to Xi pi in Deep-Inelastic Scattering at HERA**, Submitted to ***Eur Phys J. C*** 52 (2007) 507, 04/07
64. F. D. Aaron, ..., S. Gorbounov et al. (H1 Collaboration), **Charged Particle Production in High Q**2 Deep-Inelastic Scattering at HERA**, ***Phys. Lett. B*** 654:148-159,2007, 06/07
65. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of inclusive jet production in deep-inelastic scattering at high Q**2 and determination of the strong coupling**, ***Phys. Lett. B*** 653:134-144,2007, 06/07
66. A. Aktas, ..., S. Gorbounov et al. (H1 Collaboration), **Dijet Cross Sections and Parton Densities in Diffractive DIS at HERA**, ***JHEP*** 0710:042,2007 , 08/07
67. F. D. Aaron, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of deeply virtual Compton scattering and its t-dependence at HERA**, ***Phys. Lett. B*** 659:796-806,2008, 09/07
68. F. D. Aaron, ..., S. Gorbounov et al. (H1 Collaboration), **Measurement of isolated photon production in deep-inelastic scattering at HERA**, ***Eur. Phys. J. C*** 54:371-387,2008 , 11/07

69. F. D. Aaron, ..., S. Gorbounov et al. (H1 Collaboration), **Three- and Four-jet Production at Low x at HERA**, ***Eur. Phys. J. C*** 54:389-409,2008 , 11/07

As a collaborator of the ALICE experiment:

70. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **The ALICE Experiment at the CERN LHC**, ***J. Instrum.*** 3, S08002 (2008)
71. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **First proton–proton collisions at the LHC as observed with the ALICE detector: measurement of the charged-particle pseudorapidity density at √s=900**, ***Eur Phys J. C*** (2010) 65: 111-125
72. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Alignment of the ALICE Inner Tracking System with cosmic-ray tracks**, ***J. Instrum.*** 5, P03003
73. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Charged-particle multiplicity measurement in proton–proton collisions at √s=0.9 and 2.36 TeV with ALICE at LHC,** ***Eur. Phys. J. C*** (2010) 68: 89–108
74. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Charged-particle multiplicity measurement in proton–proton collisions at √s=7 TeV with ALICE at LHC,** ***Eur. Phys. J. C*** (2010) 68: 345–354
75. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Midrapidity Antiproton-to-Proton Ratio in pp Collisons at √s=0.9 and 7 TeV Measured by the ALICE Experiment,** ***Phys Rev Lett*** Vol.105, No.7, (2010)
76. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Two-pion Bose-Einstein correlations in pp collisions at √s=900 GeV,** ***Phys. Rev. D*** 82, 052001 (2010)
77. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Transverse momentum spectra of charged particles in proton–proton collisions at √s=900 GeV with ALICE at the LHC,** ***Physics Letters B*** 693 (2010) 53–68
78. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Charged-particle multiplicity density at mid-rapidity in central Pb-Pb collisions at sqrt(sNN) = 2.76 TeV,** ***Phys. Rev. Lett.*** 105, 252301 (2010)
79. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Suppression of Charged Particle Production at Large Transverse Momentum in Central Pb-Pb Collisions at √sNN = 2.76 TeV,** ***Phys. Lett. B*** 696 (2011) 30-39
80. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Elliptic flow of charged particles in Pb-Pb collisions at 2.76 TeV,** ***Phys. Rev. Lett.*** 105, 252302 (2010)
81. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Centrality dependence of the charged-particle multiplicity density at mid-rapidity**

in Pb-Pb collisions at sqrt(sNN) = 2.76 TeV, ***Phys. Rev. Lett.*** 106, 032301 (2011)

82. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Two-pion Bose-Einstein correlations in central PbPb collisions at sqrt(s_NN) = 2.76 TeV,** ***Phys. Lett. B*** 696 (4): 328-337, 2011
83. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Strange particle production in proton-proton collisions at √s = 0.9 TeV with ALICE at the LHC,** ***Eur. Phys. J. C*** 71 (3), 1594 (2011)
84. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Rapidity and transverse momentum dependence of inclusive J/psi production in pp collisions at sqrt(s)=7 TeV,** ***Phys. Lett. B*** 704 (2011) 442-455
85. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Higher harmonic anisotropic flow measurements of charged particles in Pb-Pb collisions at 2.76 TeV,** ***Phys. Rev. Lett.*** 107, 032301 (2011)
86. K. Aamodt, ..., S. Gorbunov et al. (ALICE Collaboration), **Production of pions, kaons and protons in pp collisions at sqrt(s)= 900 GeV with ALICE at the LHC,** ***Eur. Phys. J. C*** 71(6): 1655, 2011