SPD Physics & SW meeting
28 June 2023

# New algorithm of primary vertex reconstruction in SPD

V. Andreev (LPI, Moscow)

# Primary vertex reconstruction algorithm

Primary vertex reconstruction algorithm consist of two parts:

1. Initial approximation of the primary vertex.

2. Fitting procedure for the primary vertex.

3. The current primary vertex reconstruction algorithm was introduced in SPDroot in 2019 and it's performance was checked with the next procedure:

   a) comparison with the MC vertex position;

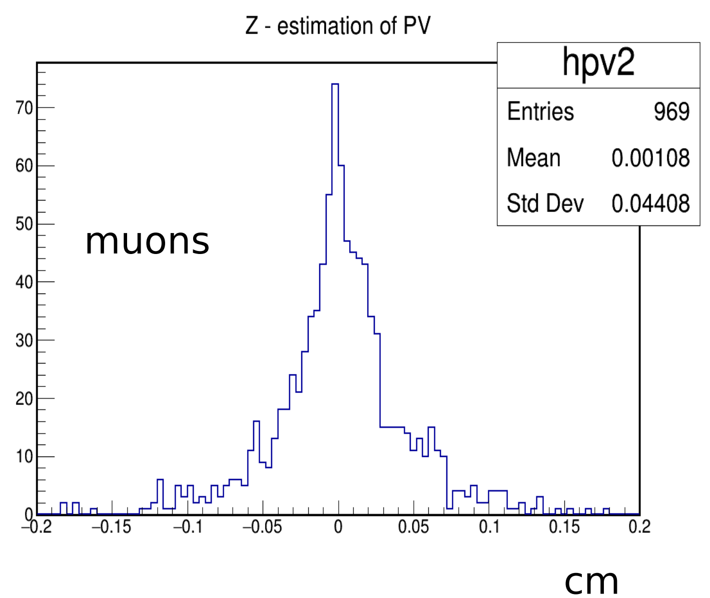
   b) comparison with the others primary vertex reconstruction algorithms (*V. Andreev, Comparison of Algorithms for Reconstructing the Primary Interaction Vertex for the SPD Experiment, Bull. Lebedev Phys. Inst. 48 (2021) 10, 301-306).*

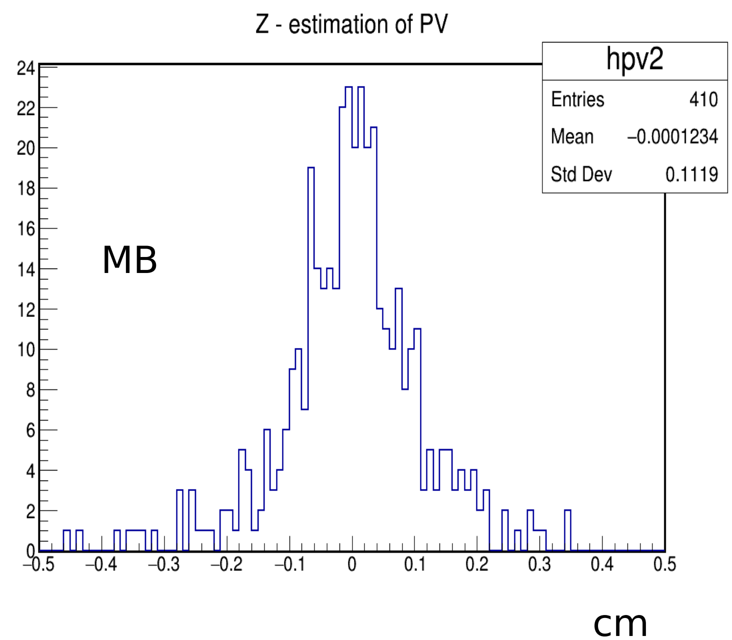
4. Current reconstruction algorithm shows the good performance

# Initial approximation of primary vertex

The important part of the vertex reconstruction is the determination of the initial approximation of the primary vertex. In SPD the next 1-D clustering algorithm is realized:

- select “good” tracks;
- extrapolate tracks to the beam axis;
- estimate z-coordinate of the point of closest approach (POCA) to the beam axis;
- apply clustering algorithm for this z-point sample for the estimation of the initial vertex position.



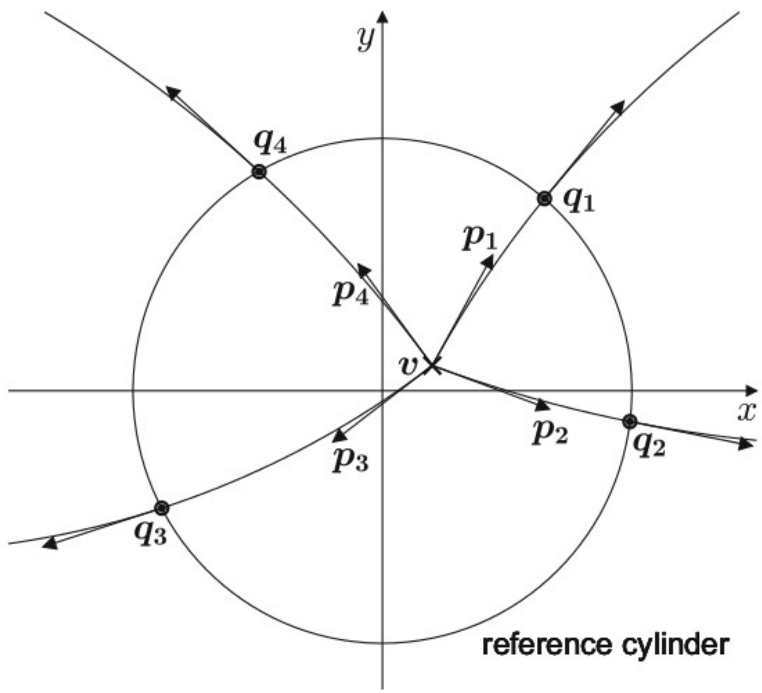




On plots above you see the difference of z-coordinate between generated primary vertex position and the initial approximation of vertex for 2 MC samples:
- 8 muons of 1 GeV/c with uniform distribution of θ and φ angles;
- Minimum Bias events.

The primary vertex position was distributed with Gaussian function $\sigma_z$ = 30 cm and $\sigma_{x,y}$ = 0.1 cm and MAPS option of vertex detector is used at this simulation.

# General approach to the vertex fit

In general, the vertex fit task can be formulated as a nonlinear regression model (P. Billoir, R. Frühwirth, M. Regler, Nucl. Instrum. Meth. Phys. Res. A 241, 115 (1985)**)**. Let us assume that there are **n** tracks which should be fitted to a common vertex $\boldsymbol{v}$. The tracks are specified by the track parameters $\boldsymbol{q_i}$ **o**n the some reference surface with corresponding covariance matrices $\boldsymbol{V_i}$ , i = 1, . . . , n. The vertex position $\boldsymbol{v}$ and momentum vectors $\boldsymbol{p_i}$ of all tracks at this vertex are the parameters which should be estimated. The track parameters $\boldsymbol{q_i = h_i(v, p_i)}$ are nonlinear functions of the parameters $\boldsymbol{v}$ and $\boldsymbol{p_i}$. If there is multiple scattering between the vertex and the reference surface, the multiple scattering has to be included as additional noise in $V_i$.




This task can be solved by the different methods. The current vertex reconstruction algorithm uses Kalman filter procedure and also the same procedure (Kalman filter) will be used in the new algorithm.

# Kalman filter for vertex fit

The nonlinear regression can be reformulated as an extended Kalman filter (*R. Fruhwirth, Application of Kalman filtering to track and vertex fitting, Nucl. Instr. Meth. A 262 (1987) 444.*).

At the first stage, the state vector consists of only the initial information about the vertex position $\mathbf{v_0}$ , and its covariance matrix $\mathbf{C_0}$. This initial information is given by the position and the size of the beam spot or the target, or from some procedure as was described above. For each track i, i = 1, . . . , n, the state vector is also presented by the three-momentum vector $\boldsymbol{p}_i$ at vertex. The Kalman filter system equations will be next: $\boldsymbol{q_i = h_i(v, p_i), i = 1, \dots, n}$

The first-order Taylor expansion of $\boldsymbol{q_i = h_i(v, p_i)}$ at a some expansion point $\boldsymbol{e_0=(v_0, p_{i,0})}$ will gives the following approximate in linear model:

$$q_i \approx A_i v + B_i p_i + c_i, \ i = 1 \dots, n,$$

$$A_i = \left.\frac{\partial h_i}{\partial v}\right|_{e_0}, \quad B_i = \left.\frac{\partial h_i}{\partial p_i}\right|_{e_0}$$

$$c_i = h_i(v_0, p_{i,0}) - A_i v_0 - B_i p_{i,0}.$$

$$r_i = q_i - h_i(v_i, p_i),$$

$$\chi_i^2 = r_i^{\mathsf{T}} G_i r_i + (v_i - v_{i-1})^{\mathsf{T}} C_{i-1}^{-1} (v_i - v_{i-1}).$$

Kalman filter equations for updating vertex and track parameters with covariance matrix will look like these:

vertex →
$$v_i = C_i \left[ C_{i-1}^{-1} v_{i-1} + A_i^{\mathsf{T}} G_i^{\mathrm{B}} (q_i - c_i) \right],$$

momentum →
$$p_i = W_i B_i^{\mathsf{T}} G_i (q_i - c_i - A_i v_i),$$

covariance →
$$\mathrm{Var}[v_i] = C_i = \left( C_{i-1}^{-1} + A_i^{\mathsf{T}} G_i^{\mathrm{B}} A_i \right)^{-1},$$

$$\mathrm{Var}[p_i] = W_i + W_i B_i^{\mathsf{T}} G_i A_i C_i A_i^{\mathsf{T}} G_i B_i W_i,$$

$$\mathrm{Cov}[v_i, p_i] = -C_i A_i^{\mathsf{T}} G_i B_i W_i.$$

$$W_i = (B_i^{\mathsf{T}} G_i B_i)^{-1}$$

$$G_i^{\mathrm{B}} = G_i - G_i B_i W_i B_i^{\mathsf{T}} G_i.$$

# Possible simplification of Kalman algorithm

The disadvantage of the basic method is that it requires too many calculations. Since the state vector ***v*** has a dimension of 3 × 3 and each measurement $\boldsymbol{q_i}$ has a dimension of 6 (or 5 for another track parametrization), complicated matrix operations must be performed at each step. In particular, at each filtration step it is necessary to invert 6 × 6 matrices. This is especially important for high multiplicity events.

For speeding up the calculations the following simplifications are usually applied:

- neglect of the magnetic field in the vertex region when tracks are considered as straight lines;

- fixation of the track directions and momenta neglecting uncertainties of these parameters;

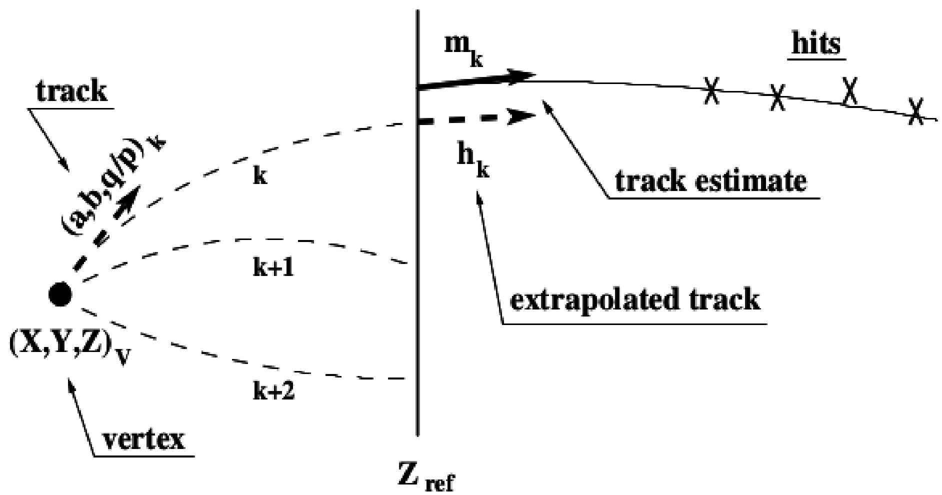
- use of initial track parameters for linearization at each iteration;

- use the helix parametrization for track with assumption of constant magnetic field.

## Current vertex reconstruction algorithm

A special feature of the current algorithm is the next:
- track is extrapolated to the some virtual plane $\boldsymbol{z_{ref}}$ ($z_{ref}$ is determined from clustering algorithm)
- then track parameters are estimated on this virtual plane;
- and finally track parameters are linearized in the vicinity of this point.

This approach makes it possible to fit the vertex without including the track parameters into the vertex state vector and to simplify the calculations. In this approach only two divisions are performed for each track, while in the standard approach inversion of a 5 × 5 matrix is required.




$\boldsymbol{r} = (x_v, y_v, z_v)^T$, $\boldsymbol{C_v}$ — the vertex position and its covariance matrix;
$\boldsymbol{t_k} = (a_k, b_k, (q/p)_k)^T$, $\boldsymbol{C^{tk}}$ — the directions and the inverse momentum of the k-th track, originating from the vertex $\boldsymbol{r}$, and covariance matrix for these parameters; measurement;
$\boldsymbol{h_k}(r, a_k, b_k, (q/p)_k)$ — parameters of the k-th track, extrapolated from $\boldsymbol{z_v}$ to $\boldsymbol{z_{ref}}$;
$\boldsymbol{m_k} = (x_k, y_k, t_{xk}, t_{yk}, (q/p)_k)^T$ — the k-th track estimation, parametrized at a certain $\boldsymbol{z_{ref}}$;
$\boldsymbol{V_k}$ — the covariance matrix of the k-th track estimate;

Each track estimation $\boldsymbol{m_k}$ is considered as measurement of the corresponding track $\boldsymbol{t_k}$.

## Current vertex reconstruction algorithm (2)

$\boldsymbol{h_k}$ ($r$, $a_k$, $b_k$, $(q/p)_k$ ) — parameters of the k-th track, extrapolated from $\boldsymbol{z_v}$ to $\boldsymbol{z_{ref}}$ .

$$\mathbf{h}_k(\mathbf{r}, a_k, b_k, (q/p)_k) = \begin{pmatrix} x_v + a_k \cdot (z_{ref} - z_v) + O((z_{ref} - z_v)^2) \\ y_v + b_k \cdot (z_{ref} - z_v) + O((z_{ref} - z_v)^2) \\ a_k \\ b_k \\ (q/p)_k \end{pmatrix}.$$

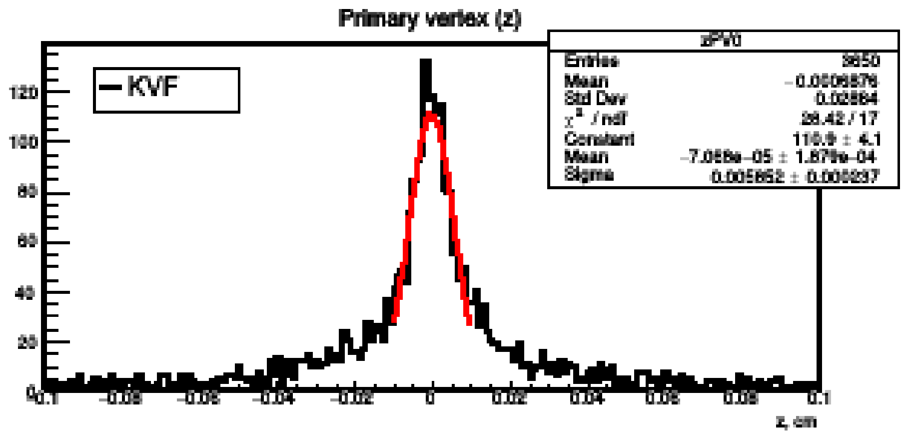
Here the term O( $(z_{ref} - z_v)^2$ ) describes the deviation of the track from a straight line in a magnetic field (see details S. Gorbunov and I. Kisel, “Analytic formula for track extrapolation in non-homogeneous magnetic field”. Nucl. Instr. and Meth. A559 (2006) ). The measurement model after linearization is:

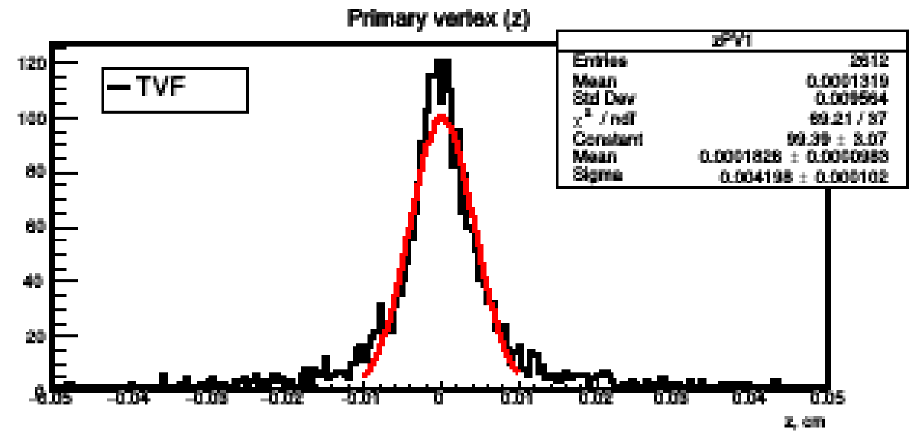
$$\mathbf{m}_k(\mathbf{r}) \approx \begin{pmatrix} a_k^0 \cdot z_v^0 \\ b_k^0 \cdot z_v^0 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & -a_k^0 & 0 & 0 & 0 \\ 0 & 1 & -b_k^0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_v \\ y_v \\ z_v \\ a_k \\ b_k \\ (q/p)_k \end{pmatrix} + \boldsymbol{\eta}_k.$$

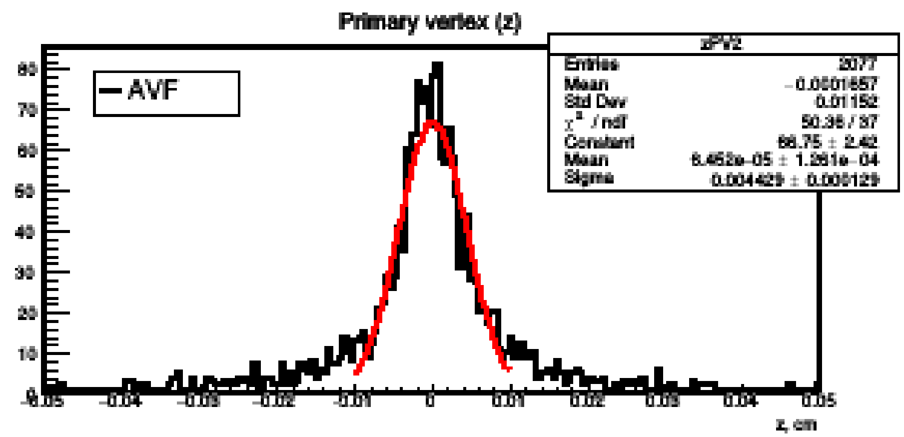
only the **x** and ***y*** components of $\boldsymbol{m_k}$ depend on the vertex parameters $\boldsymbol{r_k}$ ;
- these components do not depend on the parameters ($a_k$ , $b_k$ , $(q/p)_k$ ) of the vertex track;
- the parameters of the k-th vertex track are measured only by the k-th track estimate $\boldsymbol{m_k}$ .

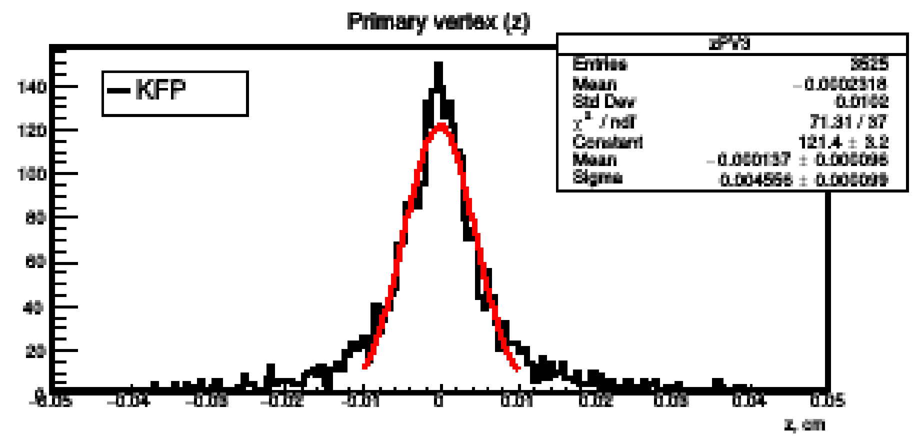
Thus, the values $a_k$ , $b_k$ , $(q/p)_k$ do not influence the measurement of the vertex position $r_k$ with the track estimate $m_k$ and, therefore, there is no need to fit these values at the k-th step of the Kalman filter.

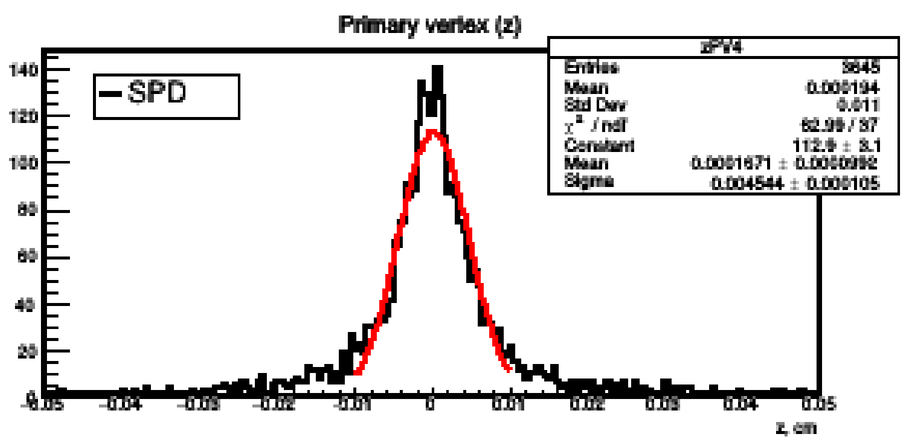
# Comparison different algorithms (MB, MAPS+DSSD, 5 layers)




**KVF** - classical Kalman vertex filter;
**TVF** - trimmed Kalman filter;
**AVF** - adaptive vertex filter;
**KFP** - vertex reconstruction from KFParticle;
**SPD –** present vertex reconstruction algorithm.

# New algorithm for vertex reconstruction

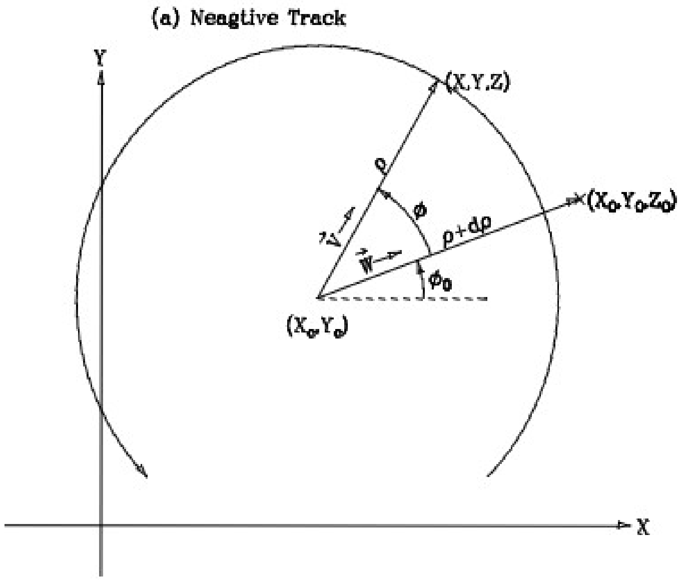
The next set parameters are used for track description in new algorithm:

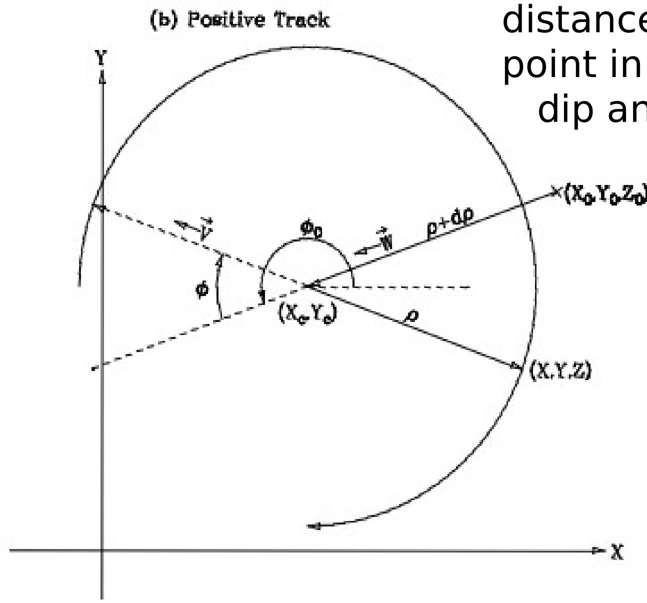
1) 6 global parameters => x,y,z – coordinates and momentum of track $p_x$ , $p_y$ , $p_z$ at this point (now in SPDroot these track parameters with covariance matrix are known at the first measured point => GetFirstState() );

2) 5 local parameters (perigee) which usually used for description of the helix in constant magnetic field. These are the same helix track parameters as used in KEK or BES3 experiments:

$$\begin{cases} x = x_0 + d_\rho \cos\phi_0 + \frac{\alpha}{\kappa}(\cos\phi_0 - \cos(\phi_0+\phi)) \\ y = y_0 + d_\rho \sin\phi_0 + \frac{\alpha}{\kappa}(\sin\phi_0 - \sin(\phi_0+\phi)) \\ z = z_0 + d_z - \frac{\alpha}{\kappa}\tan\lambda \cdot \phi, \end{cases}$$

where $\mathbf{x_0} = (x_0 , y_0 , z_0 )^T$ is an arbitrarily chosen reference point. If the reference point is fixed, the helix is determined by a 5-component parameters vector h = ($d_\rho$ , $\varphi_0$ , κ, $d_z$ , tan λ)$^T$, where $\boldsymbol{d_\rho}$ is the distance of the helix from the reference point in the xy plane, $\boldsymbol{\varphi_0}$ is the azimuthal angle to the reference point with respect to the helix center, ***k = Q/p$_T$***, $\boldsymbol{d_z}$ is the distance of the helix from the reference point in the z direction, and ***tan λ*** is the dip angle.



# New algorithm of vertex reconstruction

If we know 6 global track parameters at some x point of helix then the track helix parameters can be calculated

$$h = \begin{pmatrix} \tilde{d}_\rho \\ \tilde{\phi}_0 \\ \tilde{\kappa} \\ \tilde{d}_z \\ \tilde{\lambda} \end{pmatrix} = \begin{pmatrix} -\frac{T - p_\perp}{a} \\ \tan^{-1}\left[-\frac{p_x + ay}{p_y - ax}\right] \\ \frac{Q}{p_\perp} \\ z - \frac{p_z}{a}\sin^{-1} J \\ \frac{p_z}{p_\perp} \end{pmatrix},$$

$$p_\perp = \sqrt{p_x^2 + p_y^2},$$

$$T = \sqrt{(p_x + ay)^2 + (p_y - ax)^2},$$

$$J = \sin \rho s_\perp = \frac{p_{0x}p_y - p_{0y}p_x}{p_\perp^2} = \frac{p_y}{p_\perp} \cdot \frac{p_x + ay}{T} - \frac{p_x}{p_\perp} \cdot \frac{p_y - ax}{T} = \frac{a}{Tp_\perp}(xp_x + yp_y).$$

TMatrixD m_a(5,3); $\boldsymbol{A_i = \partial h_i / \partial v}$

```
m_a(0,0) = 0. + (py - a*x)/T;
m_a(0,1) = 0. - (px + a*y)/T;
m_a(1,0) = 0. - a*(px + a*y)/T/T;
m_a(1,1) = 0. - a*(py - a*x)/T/T;
m_a(3,0) = 0. - (pz/T)*(px + a*y)/T;
m_a(3,1) = 0. - (pz/T)*(py - a*x)/T;
m_a(3,2) = 1.;
```

TMatrixD m_b(5, 3); $\boldsymbol{B_i = \partial h_i / \partial p_i}$

```
m_b(0,0) =      (px/pxy-(px+a*y)/T)/a;
m_b(0,1) =      (py/pxy-(py-a*x)/T)/a;
m_b(1,0) = 0. - (py-a*x)/T/T;
m_b(1,1) = 0. + (px+a*y)/T/T;
m_b(2,0) = 0. - charge*px/pxy/pxy/pxy;
m_b(2,1) = 0. - charge*py/pxy/pxy/pxy;
m_b(3,0) = 0. + (pz/a)*(py/pxy/pxy-(py-a*x)/T/T);
m_b(3,1) = 0. - (pz/a)*(px/pxy/pxy-(px+a*y)/T/T);
m_b(3,2) = 0. - asin(J)/a;
m_b(4,0) = 0. - (px/pxy)*(pz/pxy)/pxy;
m_b(4,1) = 0. - (py/pxy)*(pz/pxy)/pxy;
m_b(4,2) =      1./pxy;
```

# General fitting procedure

General procedure:

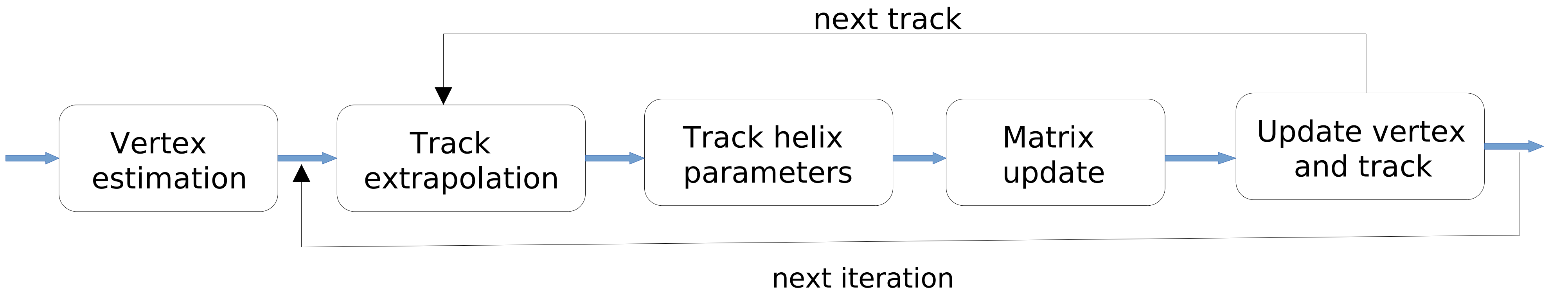
1. determine the preliminary position of the primary vertex;

2. extrapolate track to this vertex position using Runge-Kutta-Nyström method;

3. transform track to the local helix parameters in the area of this vertex;

4. apply linearization of track parameters => calculate all necessary matrix;

5. update vertex position and track parameters and corresponding covariance matrix using Kalman filter equations;



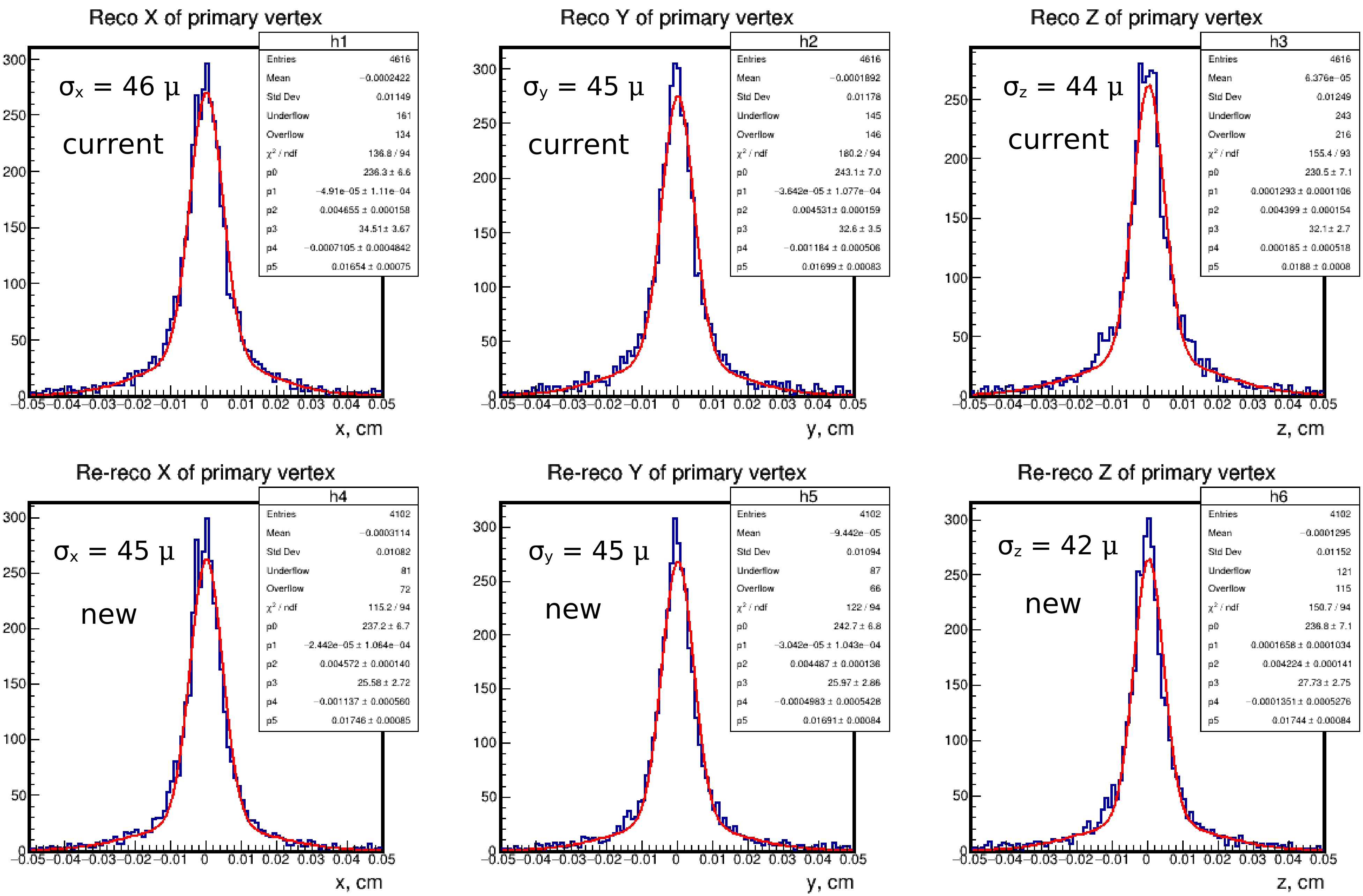

6. do loop over all selected tracks;

7. do this procedure several time ( iteration )

8. finally the primary vertex position and parameters of tracks connected with this vertex and corresponding covariance matrix are determined

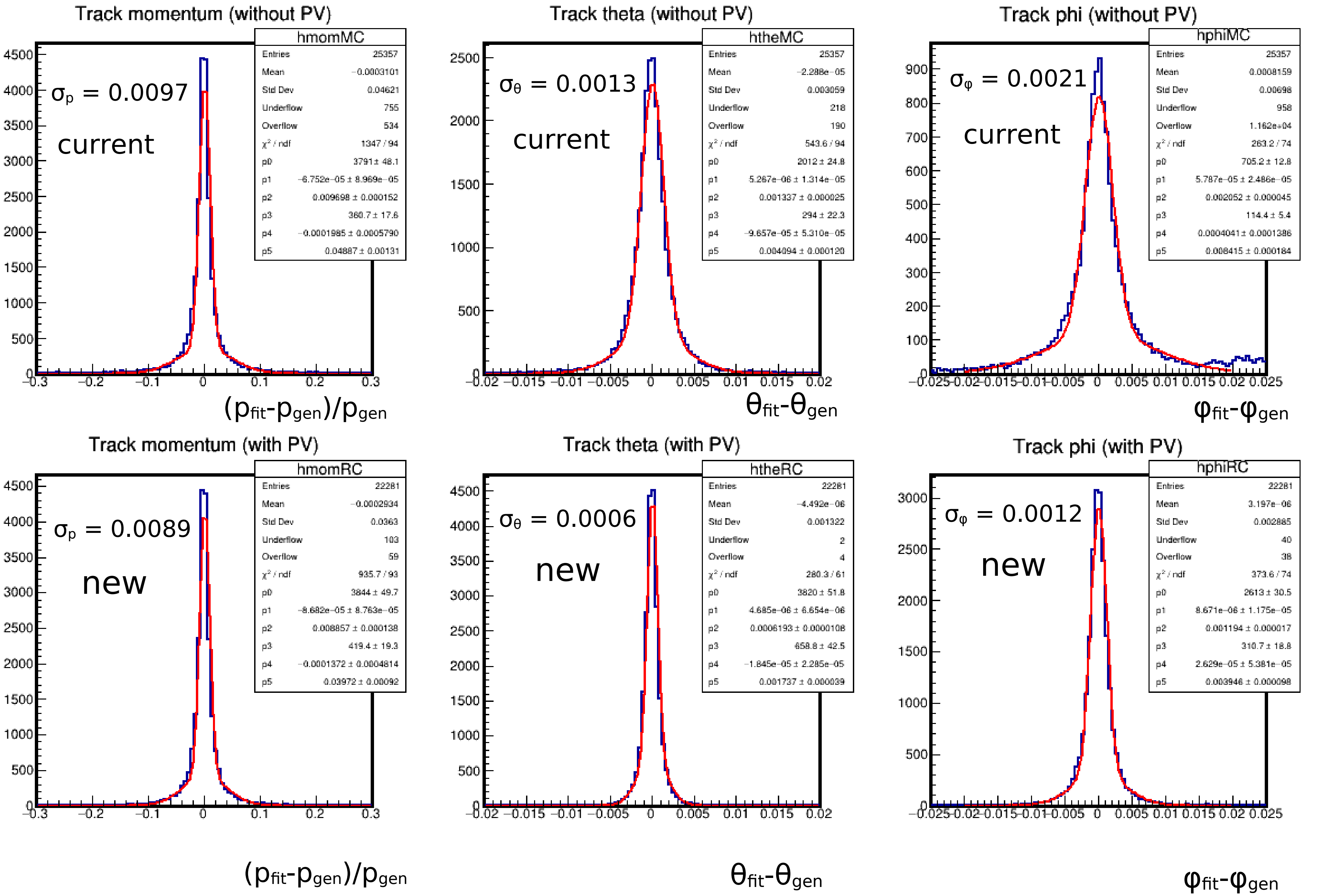
# Minimum Bias (vertex parameters)

Fit with 2 gauss



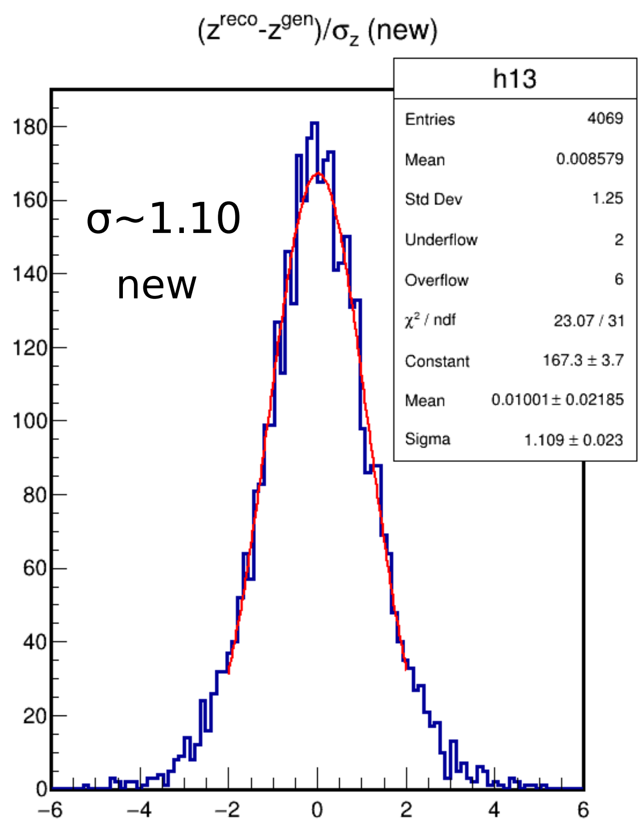
## Minimum Bias (track parameters)

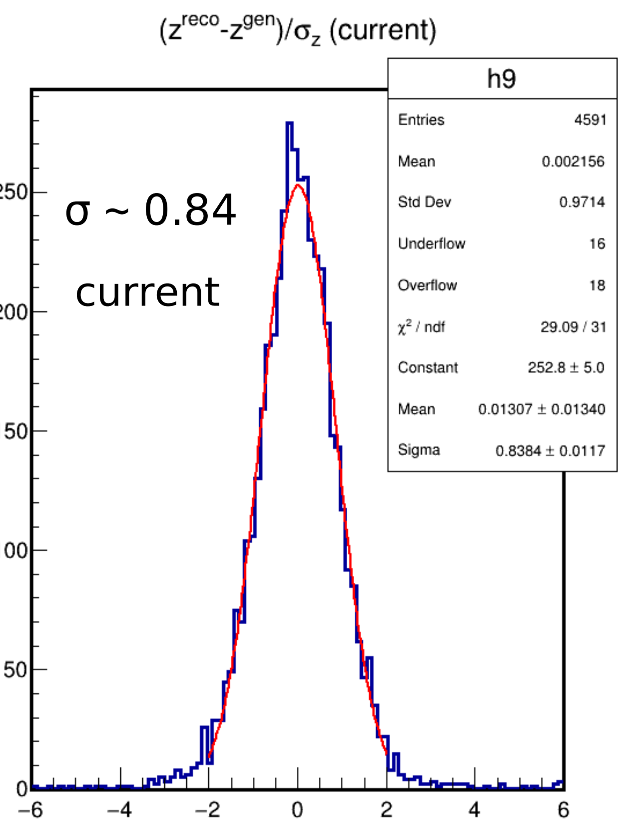
For current algorithm all tracks are extrapolated to PV, fit by 2 Gauss



# Pull distribution

Pull distribution for some variable ***x*** is determined as $\boldsymbol{(x^{reco} - x^{gen}) / \sigma_x}$ **.**
In general, for Gaussian error this distribution should have sigma $\sigma \sim 1.0$

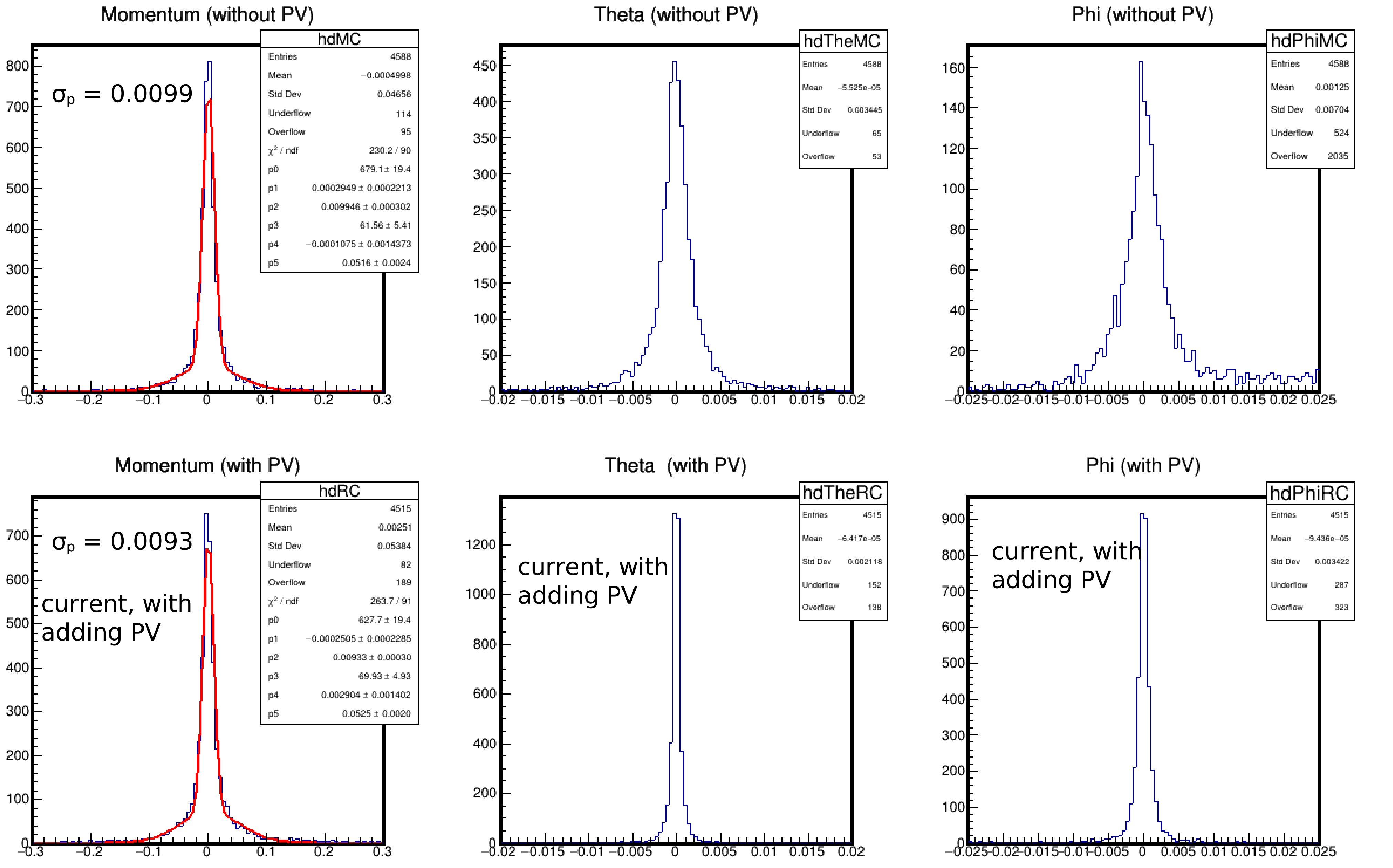



1. Error of primary vertex is overestimated for the current reconstruction algorithm as the tracks error is not fully taken into account (?).

2. For new algorithm this error is a little underestimated.

## Minimum Bias (track parameters for current algorithm with adding PV as additional point to fit)




$(p_{fit}-p_{gen})/p_{gen}$ $\theta_{fit}-\theta_{gen}$ $\varphi_{fit}-\varphi_{gen}$

## Comparison of two fitting algorithms

| | Current | New |
|---|---|---|
| 1. general selection of track | yes | yes |
| 2. additional selection $(\theta - \pi/2) < \Delta\theta$ | yes | no |
| 3. initial track parameters | 6 global | 6 global |
| 4. track extrapolation | to plane (XY) | to space point |
| 5. track extrapolation method | Runge-Kutta | Runge-Kutta |
| 6. local track parameters | 5 on plane | 5 helix (perigee) |
| 7. local track description | ~curve 2-d order | exact helix |
| 8. track linearization | yes | yes |
| 9. update vertex at each step | yes | yes |
| 10. track selection ($\chi^2$) at each step | yes | yes |
| 11. update track parameters | no | yes |
| 12. can be used in nonuniform field | yes | yes |
| 13. additional step for track parameters update | yes | no |
| 14. performance | similar | similar |

# Summary

1. Current primary vertex reconstruction program was introduced in SPDroot in 2019 on the base of CBM algorithm.

2. New primary vertex reconstructed algorithm is now proposed.

3. Both algorithms show good and compatible results.

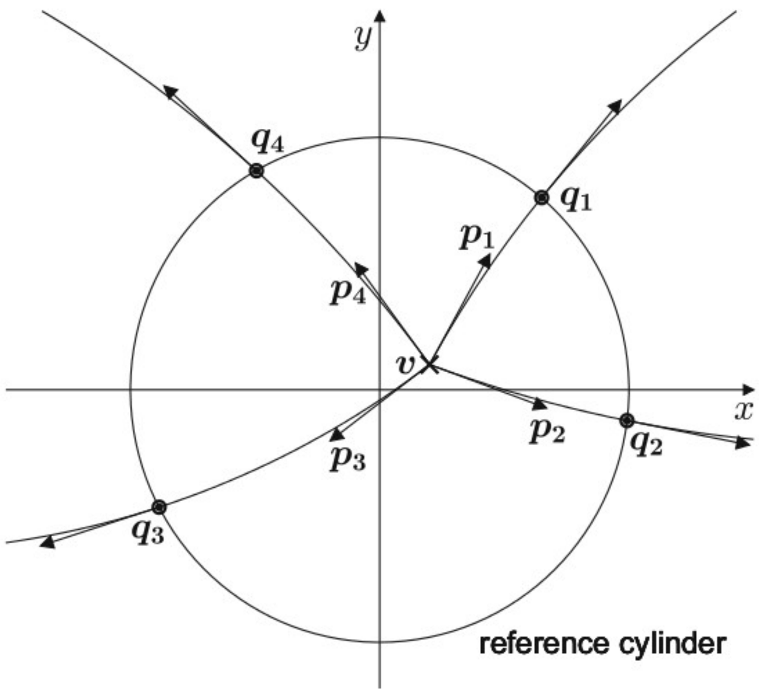
4. New algorithm works now inside standard SPDroot software (on my PC).

5. Future plan – add this algorithm in the official SPDroot.

# Backup

# General approach to the vertex fit

In general the vertex fit can be formulated as a nonlinear regression model (**P. Billoir, R. Frühwirth, M. Regler, Nucl. Instrum. Meth. Phys. Res. A 241, 115 (1985))**. Assume that there are **n** tracks to be fitted to a common vertex. The tracks are specified by the estimated track parameters $\mathbf{q}_i$ and the associated covariance matrices $\boldsymbol{V}_i$ , i = 1, . . . , n. The parameters to be estimated are the vertex position $\boldsymbol{v}$ and the momentum vectors $\mathbf{p}_i$ of all tracks at the vertex. The track parameters $\mathbf{q}_i$ are nonlinear functions of the parameters: $\boldsymbol{q_i = h_i(v, p_i) + \varepsilon_i , cov(\varepsilon_i) = V_i, I = 1\ldots,n}$.




$$\boldsymbol{q}_i = \boldsymbol{h}_i(\boldsymbol{v}, \boldsymbol{p}_i), \ i = 1\ldots, n.$$

If there is multiple scattering between the vertex and the reference surface, it has to be included as additional noise in $\boldsymbol{V}_k$. The first-order Taylor expansion of $\mathbf{h}_i$ at a some expansion point $\boldsymbol{e_0 = (v_0 , p_{i,0} )}$ gives the following approximate linear model:

$$\boldsymbol{q}_i \approx \boldsymbol{A}_i \boldsymbol{v} + \boldsymbol{B}_i \boldsymbol{p}_i + \boldsymbol{c}_i, \ i = 1\ldots, n,$$

$$\boldsymbol{A}_i = \left.\frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{v}}\right|_{\boldsymbol{e}_0}, \quad \boldsymbol{B}_i = \left.\frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{p}_i}\right|_{\boldsymbol{e}_0}$$

$$\boldsymbol{c}_i = \boldsymbol{h}_i(\boldsymbol{v}_0, \boldsymbol{p}_{i,0}) - \boldsymbol{A}_i \boldsymbol{v}_0 - \boldsymbol{B}_i \boldsymbol{p}_{i,0}.$$

$$\begin{pmatrix} \boldsymbol{q}_1 \\ \vdots \\ \boldsymbol{q}_n \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}_1 & \boldsymbol{B}_1 & \boldsymbol{O} & \ldots & \boldsymbol{O} \\ \boldsymbol{A}_2 & \boldsymbol{O} & \boldsymbol{B}_2 & \ldots & \boldsymbol{O} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_n & \boldsymbol{O} & \boldsymbol{O} & \ldots & \boldsymbol{B}_n \end{pmatrix} \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{p}_1 \\ \vdots \\ \boldsymbol{p}_n \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_1 \\ \vdots \\ \boldsymbol{c}_n \end{pmatrix}.$$

## Least square method (Billoir method)

The Least Square method estimates **v** and **p**$_i$ are obtained by:

$$\begin{pmatrix} \hat{v} \\ \hat{p}_1 \\ \vdots \\ \hat{p}_n \end{pmatrix} = M^{-1} N \begin{pmatrix} q_1 - c_1 \\ \vdots \\ q_n - c_n \end{pmatrix}, \qquad M = \begin{pmatrix} D_0 & D_1 & D_2 & \dots & D_n \\ D_1^\top & E_1 & O & \dots & O \\ D_2^\top & O & E_2 & \dots & O \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ D_n^\top & O & O & \dots & E_n \end{pmatrix}, \qquad N = \begin{pmatrix} A_1^\top G_1 & A_2^\top G_2 & \dots & A_n^\top G_n \\ B_1^\top G_1 & O & \dots & O \\ O & B_2^\top G_2 & \dots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \dots & B_n^\top G_n \end{pmatrix},$$

$$D_i = A_i^\top G_i B_i, \quad E_i = B_i^\top G_i B_i = W_i^{-1}, \quad G_i = V_i^{-1}, \quad i = 1, \dots, n,$$

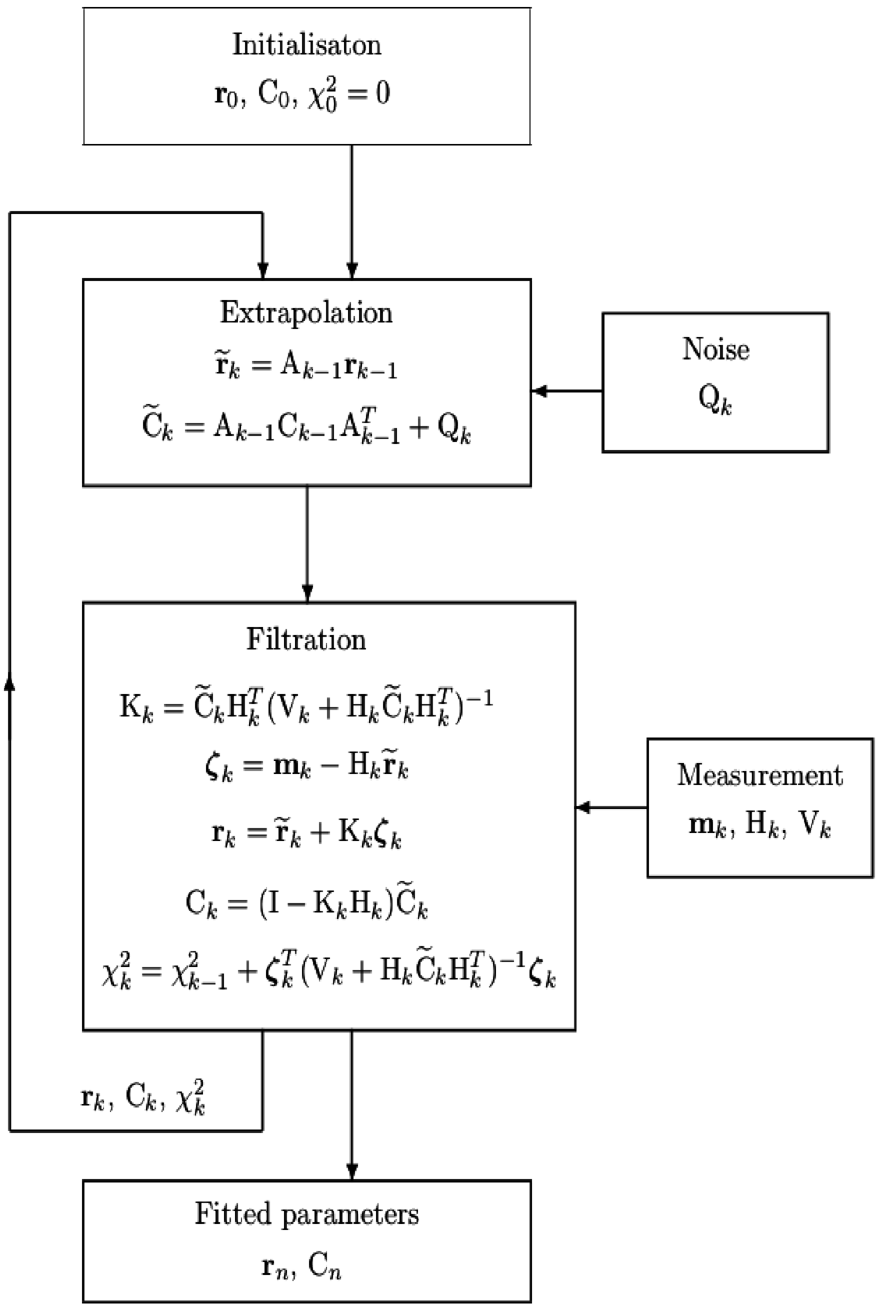
$$D_0 = \sum_{i=1}^{n} A_i^\top G_i A_i. \qquad C_{00} = \left( D_0 - \sum_{i=1}^{n} D_i W_i D_i^\top \right)^{-1},$$

$$\hat{v} = C_{00} \sum_{j=1}^{n} A_j^\top G_j (I - B_j W_j B_j^\top G_j)(q_j - c_j),$$

$$\hat{p}_i = W_i B_i^\top G_i (q_i - c_i - A_i \hat{v}), \quad i = 1, \dots, n.$$

$$\chi^2 = \sum_{i=1}^{n} (q_i - \hat{q}_i)^\top G_i (q_i - \hat{q}_i).$$

# Kalman filter algorithm




**state vector** $r^t$ – vector real numbers that represents the unknown quantities to be estimated

**Extrapolation** – changes current estimation of vector $\boldsymbol{r_k}$ upon transfer from (k-1)-th measurement to the k-th measurement

$\boldsymbol{r_k^t = A_k\, r_{k-1}^t + v_k}$, $\boldsymbol{A_k}$ – is a known linear operator, called **etrapolator;** $v_k$ – process noise between (k-1) and k - measurement

**Filtration** – the measurement information is incorporated into the estimator and its covariance matrix

**measurement** $\mathbf{m_k}$ – a known (measured) quantity with linearly depends on state vector $\mathbf{m = H \cdot r^t + \eta}$
**H** – is a (known) linearly operator represented as a matrix, called **model of measurement; η –** measurement error

Algorithm steps 2-3 sequentially repeat n times, for each measurement $m_k$, k = 1,...n. After the filtration of the last measurement $\boldsymbol{m_n}$, the obtained estimator $r_n$ is the desired best estimator with the covariance matrix $\boldsymbol{C_n}$.

In practice, the transport equation and the measurement model 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.