SPD Physics & MC meeting
20 March 2024

Primary vertex reconstruction algorithms in SpdRoot

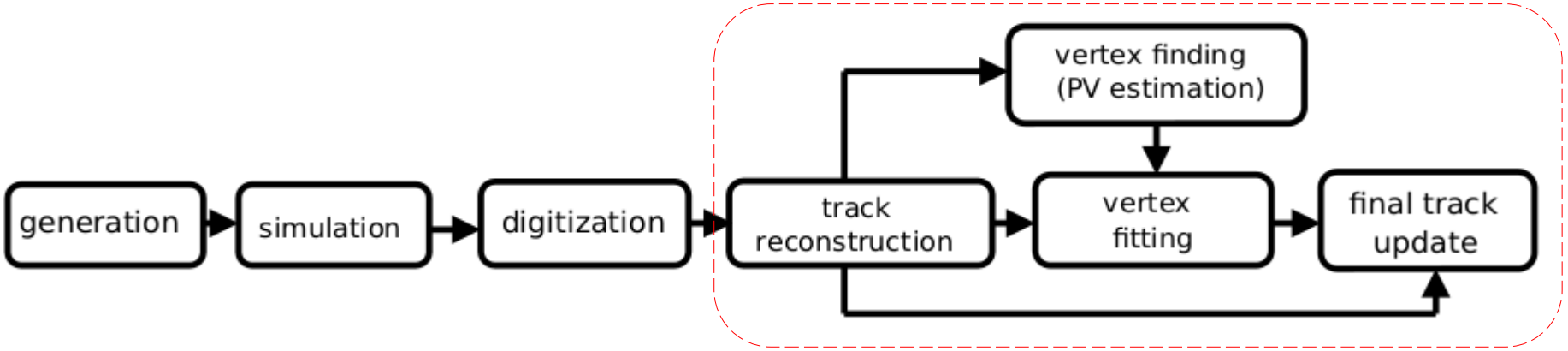
V. Andreev (LPI, Moscow)

General scheme of primary vertex reconstruction algorithm

Primary vertex reconstruction algorithm consists of two parts:

1. Initial approximation of primary vertex (vertex finding).

2. Fitting procedure for primary vertex (vertex fitting).



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 MC vertex position;

   b) comparison with the others primary vertex reconstruction algorithms used in High Energy Physics.

4. Current reconstruction algorithm (on the base of CBM algorithm) shows the good performance.

5. The two new additional algorithms will be presented in this talk which can be also used for primary vertex reconstruction procedure.

Cross check with others reconstruction algorithms

The cross check for the current SPD primary vertex reconstruction algorithm was done (in 2020) on the base of comparison with the others vertex reconstruction algorithms which used in High Energy Physics.

The next algorithms were considered:

- classical Kalman vertex filter (KVF) which uses all tracks in event;

- trimmed Kalman filter (TVF) which removes some “bad” tracks in event;

- adaptive vertex filter (AVF) which assigns some weight to each track;

- primary vertex reconstruction from KFParticle package (KFP);

- current SPD vertex reconstruction algorithm.

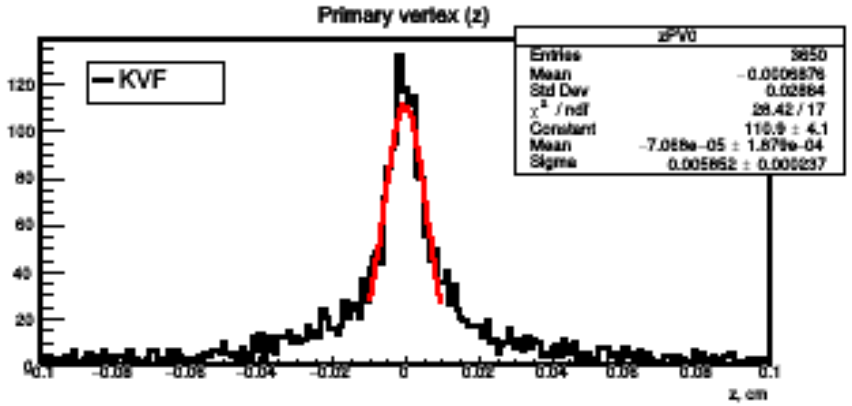
The KVF, TVF and AVF primary vertex reconstruction algorithms are realized in the special RAVE package (Reconstruction in an Abstract, Versatile Environment) which was developed for vertex reconstruction in CMS experiment.

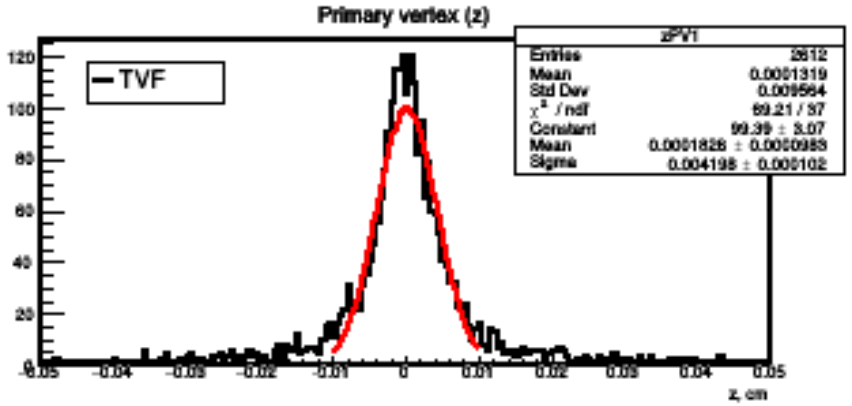
It is necessary to emphasize that the considered algorithms in RAVE use the different parameters for track representation and also various algorithms for finding initial approximation of the primary vertex with comparison of SPD reconstruction.

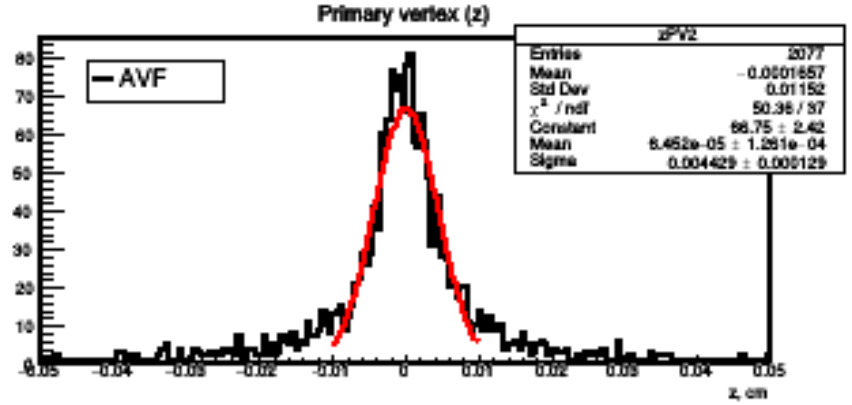
Also these algorithms (KVF, TVF, AVF, KFP) are realized in assumption of the constant magnetic field and for this reason the comparison was done for the constant magnetic field also for SPD algorithm.
For the initial approximation of the vertex position (vertex finding) the value from generator level was used.

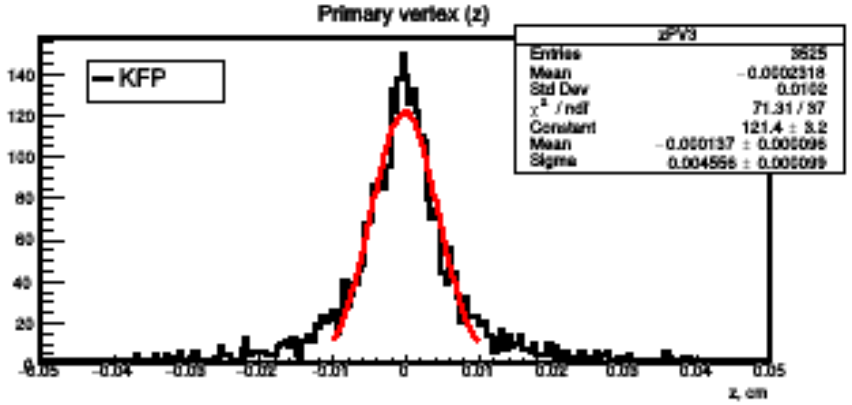
The special interface was written in order to use data from SPD in RAVE package.

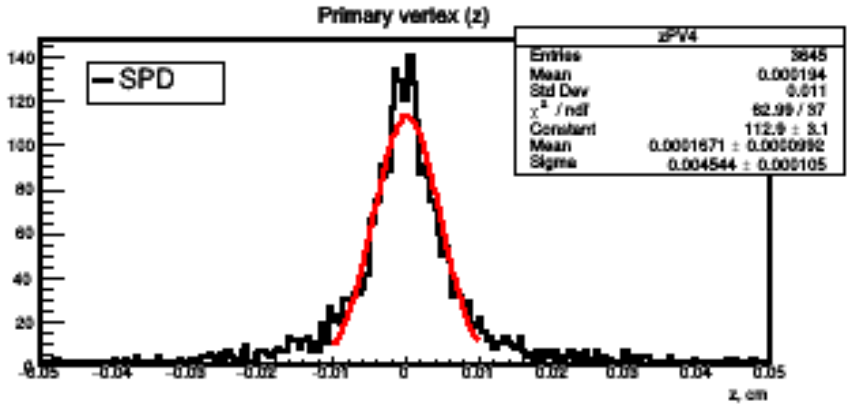
Comparison of different algorithms (MB, MAPS+DSSD)











Comparison was done in 2020 for the old SPD geometry of vertex and tracking detector and for constant magnetic field. All algorithms use Kalman filter procedure:

Current primary vertex reconstruction algorithm shows the good agreement comparison with TVF and KFP algorithms.

Primary vertex finding algorithms (method 1)

Primary vertex finding procedure is very important part of the vertex reconstruction algorithm.

The main specific feature of SPD experiment is the very broad primary vertex distribution in z-direction (σ_z = 30. cm). In SPD the vertex finding procedure is realized in assumption, that the beam distribution in transverse XY-plane is relatively small and has Gaussian shape with $\sigma_{x,y}$ = 0.1 cm.

In present moment there are 3 different algorithms for initial approximation of the primary vertex:

A) 1-st method => 1-D clustering procedure:

- select good fitted tracks;

- extrapolate tracks to the beam axis (x=y=0) and assign to each track a weight which is proportional to inverse track error at the point of closest approach to the beam axis;

- remove tracks which are faraway from the beam axis;

- construct 1-st estimation of z-vertex position from the selected tracks with taking into account the track weight;

- find the “best” and “worst” tracks:
  a) “worst” track – track with maximum distance from z-vertex position approximation;
  b) “best” track – track with maximum weight (or minimum error) inside some range around z-vertex position;

- produce 2-nd estimation of z-vertex position without “worst” track;

- do next iteration;

- final vertex estimation => construct cluster around “best” track using only tracks inside some range around z-position of the “best” track with taking into track weight.

Primary vertex finding algorithms (method 2)

B) 2-d method => 1-D (simplified) clustering procedure:

- select good fitted tracks;
- extrapolate tracks to the beam axis (x=y=0.0);
- remove tracks which are faraway from the beam axis;
- estimate z-coordinate for point of the closest approach to the beam axis and form array of z-points;
- calculate all possible distances between z-points;
- find point with minimum distance (or two tracks with minimum distance between, along z coordinate);
- construct 1-st estimation of z-vertex position using these two tracks;
- construct z-vertex position using tracks inside some range around 1-st estimation of z-vertex position;
- remove all used tracks inside this range;
- do next iteration;
- use for primary vertex estimation from cluster position with maximum number of used tracks.

Primary vertex finding algorithms (method 3)

C) 3-d method => 3-D clustering procedure on the base of KFParticle package.
But need to point out that KFParticle package are valid only for constant magnetic field (there is some specific option for ALICE configuration of magnetic field) and we must to remember this.

Vertex construction procedure in KFParticle package from n-daughter particles:

- determine point of closest approach of two first daughter particles;
- extrapolate daughter particle to this point;
- update position of this point using standard Kalman filter equations;
- track parameters are remained unchanging during this procedure.

Procedure for primary vertex finding algorithm:

- select good fitted tracks;

- extrapolate tracks to the beam axis (using Runge-Kutta algorithm) and construct KFParticle tracks with the new first “measured” point as the point of closest approach to the z-axis;

- remove tracks which are faraway from the beam axis;

- construct vertex from tracks using KFParticle procedure (put Bz value at x=y=z=0.0);

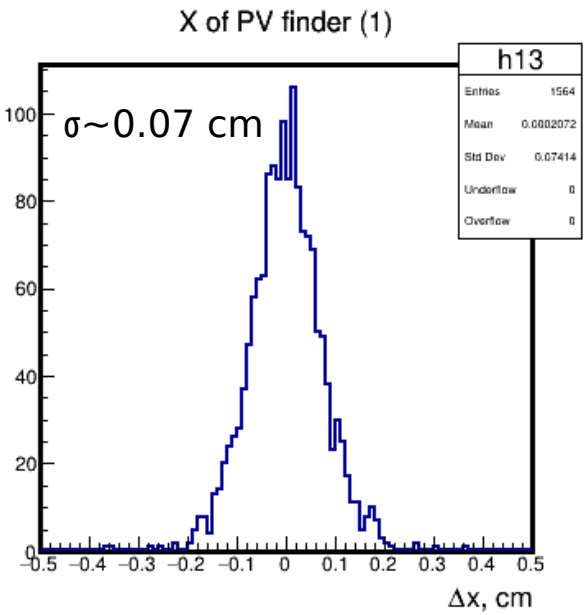
- find “ worst” track with maximum chi2 to the constructed vertex and larger than some threshold value;

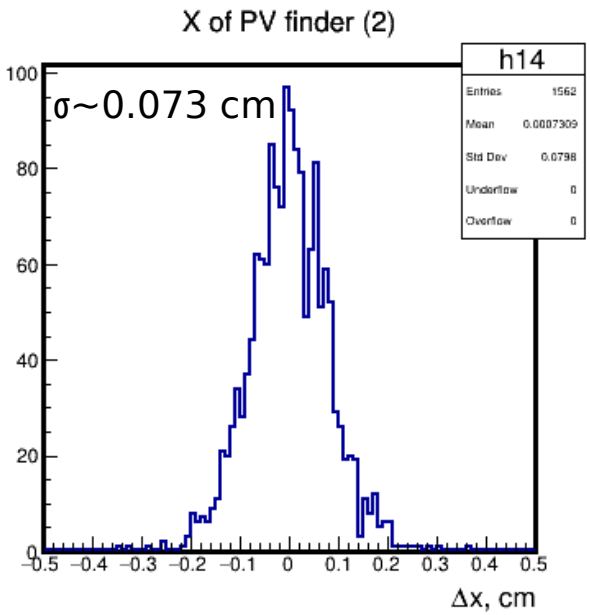
- remove “ worst” track;

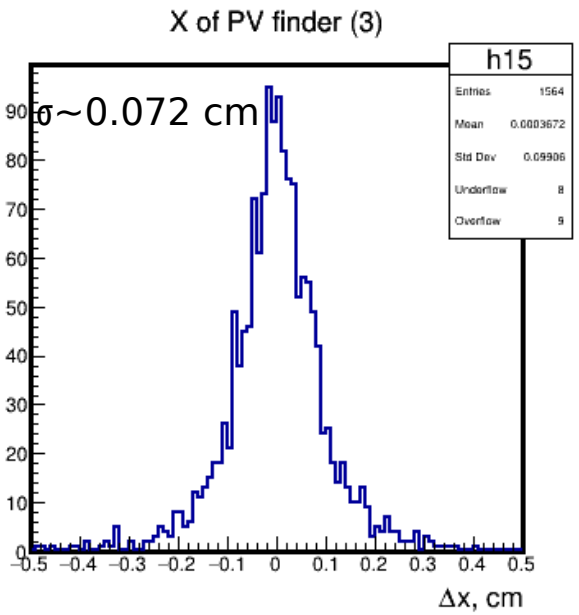
- do next iteration until all “worst” tracks will be removed or only 2 tracks are remained;

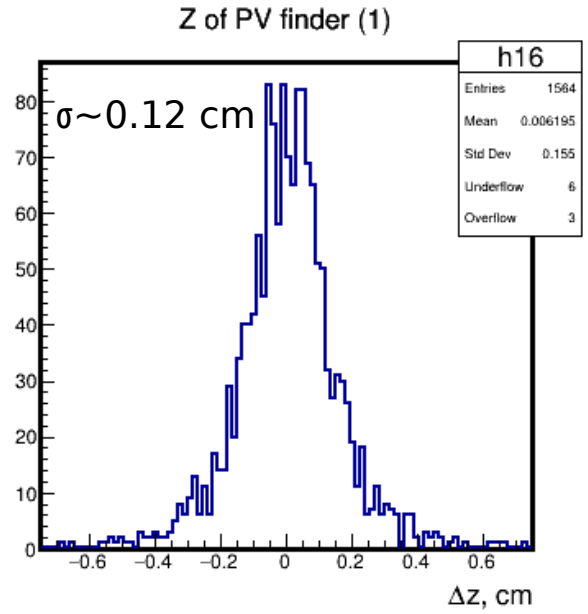
- construct the final primary vertex estimation using remaining KFParticle tracks.

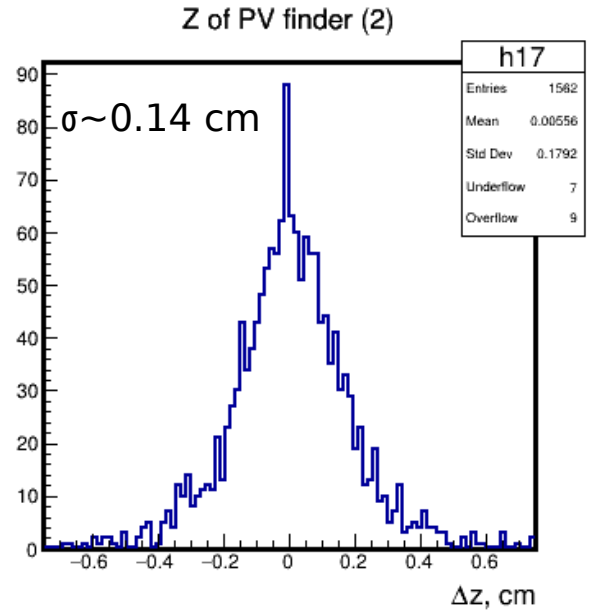
Primary vertex finding (results, 1)

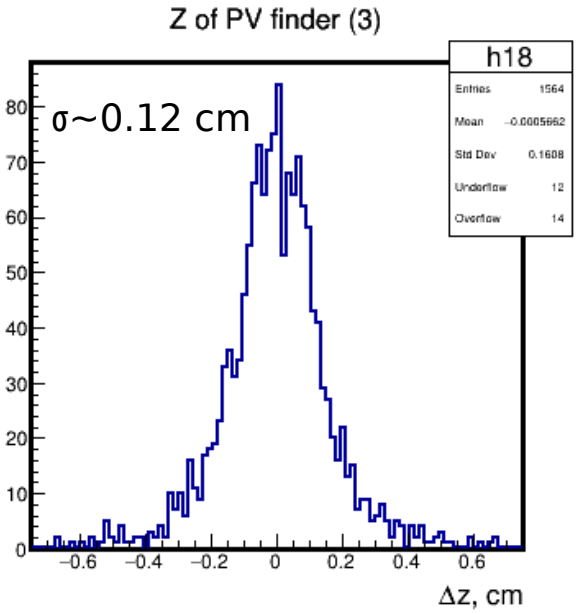






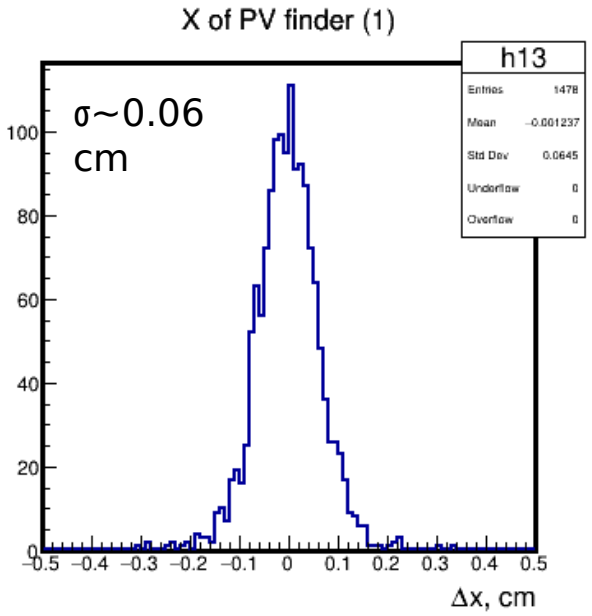


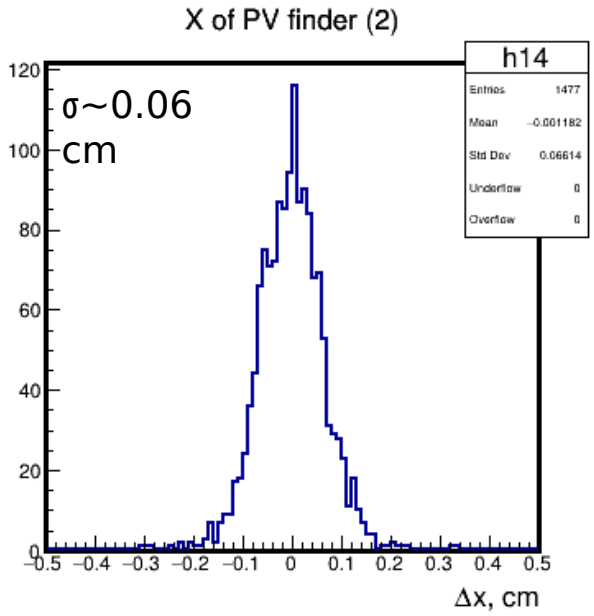


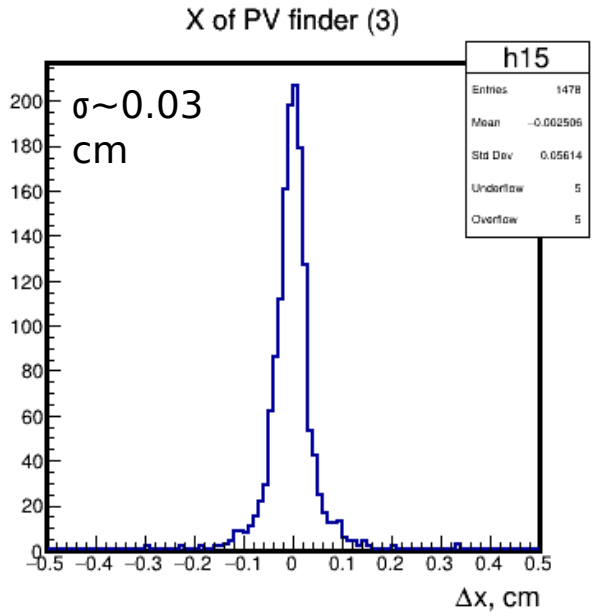


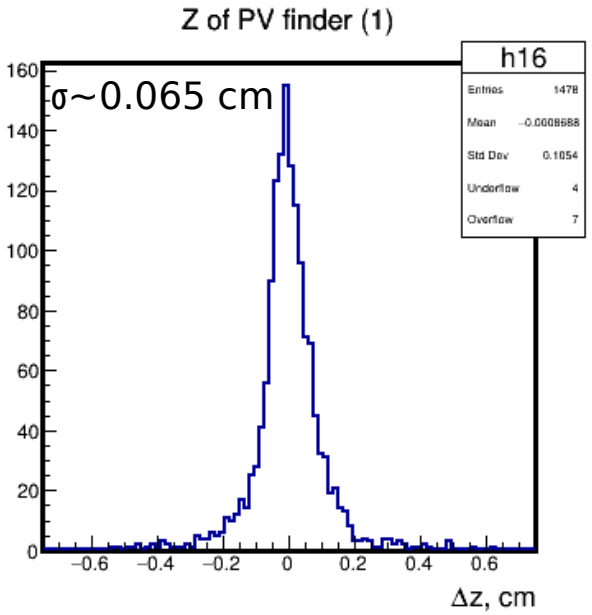
Plots show the difference in x and z-coordinates between the generated primary vertex position and vertex position which is provided by the finding methods for Minimum Bias events and without vertex detector.

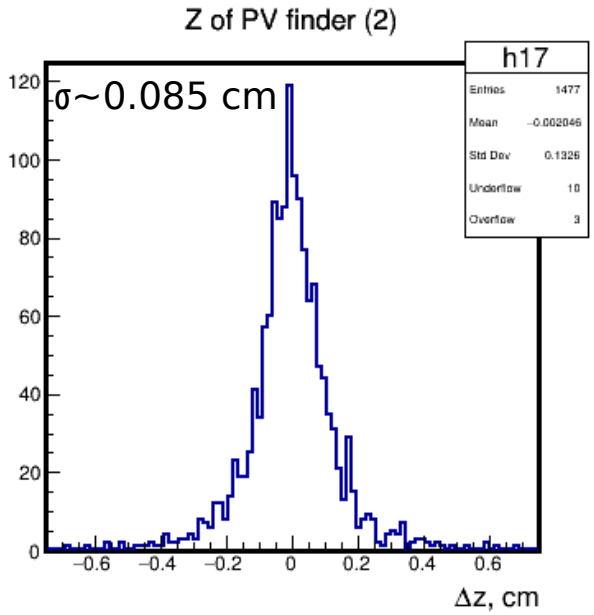
Primary vertex finding (results, 2)

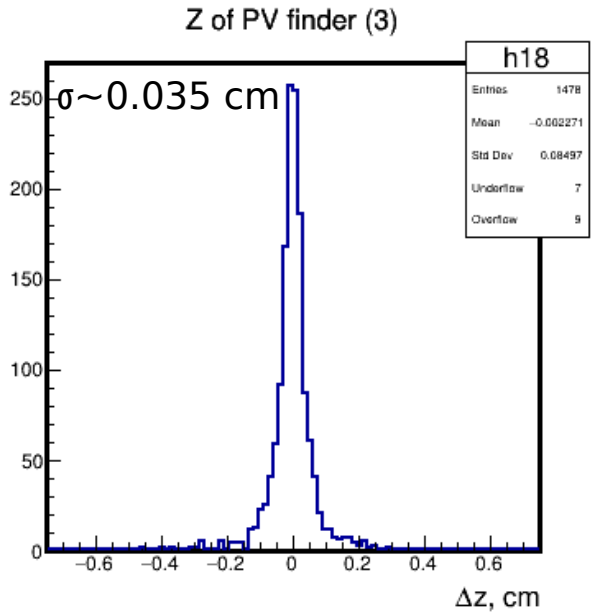












These plots show the difference in x and z-coordinates between the generated primary vertex position and vertex position which is provided by the different vertex finding methods for Minimum Bias events and with 1 layer of Micro Megas vertex detector. Best result is for KFParticle based algorithm (~2 times).

Primary vertex fitting algorithm (method 1)

Next step of the primary vertex reconstruction algorithm is the vertex fitting procedure in which the estimated vertex position on the previous step (vertex finding) is used as the 1-st approximation.

3 different fitting algorithms were studied and will be presented below. All presented algorithms use the Kalman filter procedure.

A) Algorithm base on the CBM experiment method. A special feature of this algorithm is the next:

- track is extrapolated to the some virtual plane z_{ref} (z_{ref} is determined from vertex finding algorithm);
- then track parameters are calculated on this virtual plane;
- remove “bad” tracks which are faraway from vertex position;
- then use the second order curve for description of the track trajectory and after linearize track parameters in the vicinity of this point;
- update vertex position using standard Kalman filter equations;
- repeat this procedure several times.

This approach gives possibility to fit the primary vertex without including the track parameters into the vertex state and so to simplify the calculations.

The current CBM primary vertex fitting algorithm has the some “weak” points:

- does not include track with theta angle +-3^0 around the vertical plane in fitting procedure;
- use the second order curve for track trajectory description;
- track parameters are considered unchanged during fitting procedure (?);

Primary vertex fitting algorithms (method 2)

B) The next primary vertex fitting algorithm is based on the exact description of track trajectory and full set of Kalman filter equations.

For the track description the next two set of track parameters are used:

1) 6 global parameters => ***x,y,z*** – coordinates and momentum of track $\boldsymbol{p_x}$, $\boldsymbol{p_y}$, $\boldsymbol{p_z}$ at some point;
2) 5 local parameters (helix) which usually used for description of the helix in constant magnetic field.

Algorithm procedure is the next:

- extrapolate track to the 1-st estimation of primary vertex (from vertex finding) using Runge-Kutta method;

- in the small vicinity of this vertex the magnetic field can be considered as constant;

- transform 6 global track parameters to the 5 local helix parameters in the vicinity of this vertex;

- linearize of track parameters => calculate all necessary matrix using exact formulas;

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

- remove “bad” tracks with large chi2;

- repeat this procedure several times.

Primary vertex fitting algorithms (method 3)

C) Primary vertex fitting algorithm which uses the KFParticle package:

- extrapolate track to the 1-st estimation of primary vertex (from vertex finding) using Runge-Kutta method;

- construct KFParticle tracks with the new first “measured” point as the point of closest approach to the 1-st estimation of primary vertex;

- in the small vicinity of this vertex the magnetic field can be considered as constant and KFParticle can be used;

- remove tracks which are faraway from the vertex;

- construct vertex using KFParticle procedure (put Bz value at xv, yv, zv);

- find “ worst” track with maximum chi2 from the constructed vertex and larger than some maximum value;

- remove “ worst” track;

- do iteration until all “worst” tracks will be removed or only 2 tracks are remained;

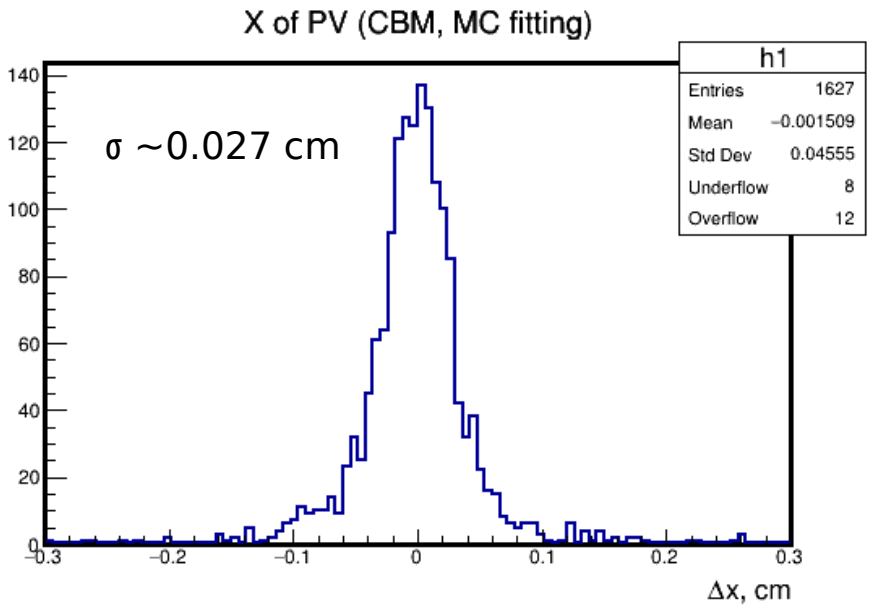
- construct final primary vertex using remaining KFParticle tracks;

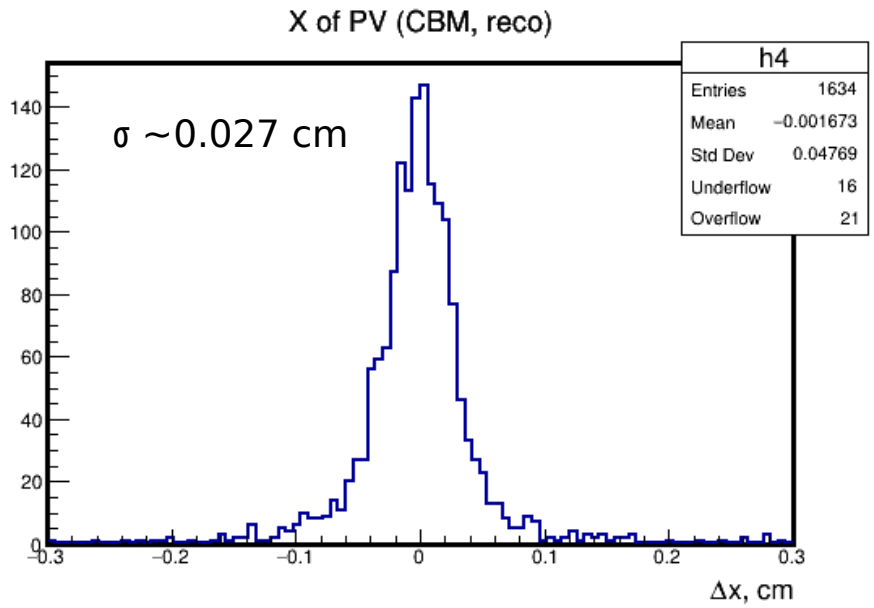
- track parameters are considered unchanged during the fitting procedure;

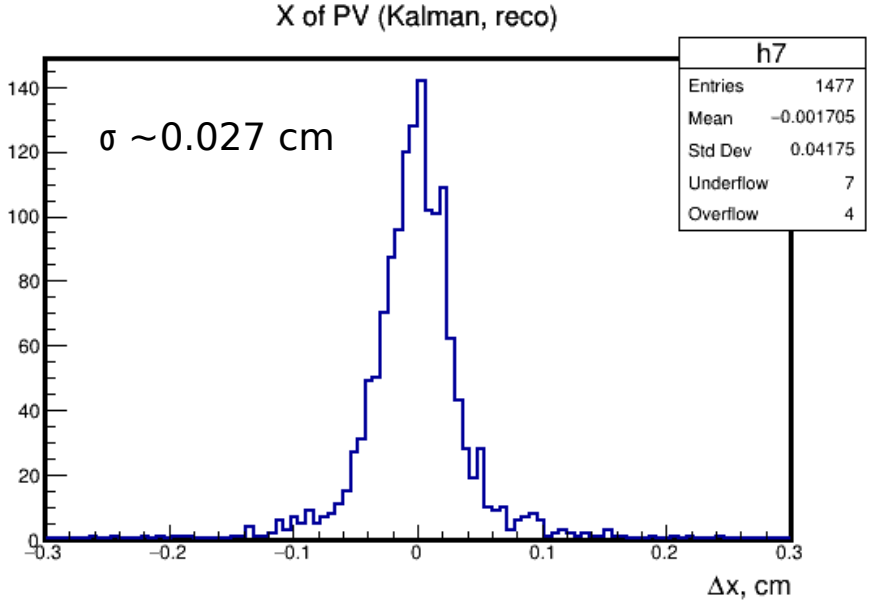
For checking of these primary vertex reconstruction algorithm 2000 Minimum Bias events was simulated.
Vertex detector had 1 layer of Micro Megas detector and standard configuration of the straw tracker.
Primary vertex was smeared with Gaussian shape ($\sigma_{x,y}$ = 0.1 cm and σ_z = 30 cm).

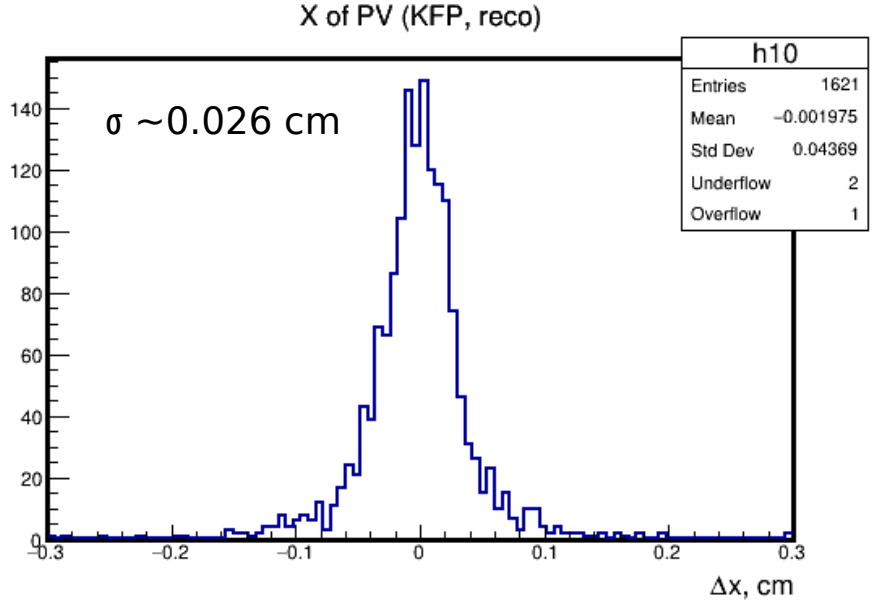
Reconstruction precision Is determined as the difference between reconstructed and generated value.

Primary vertex reconstruction precision (1)



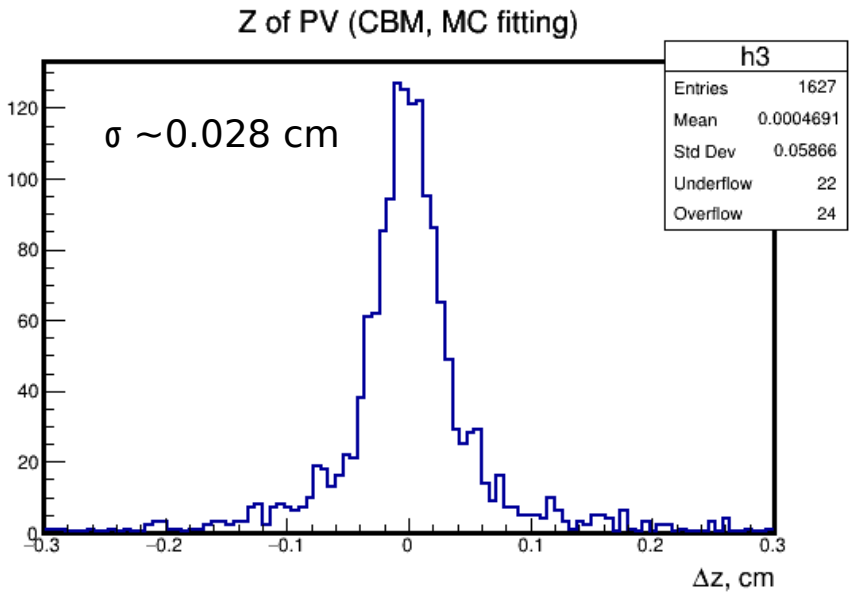


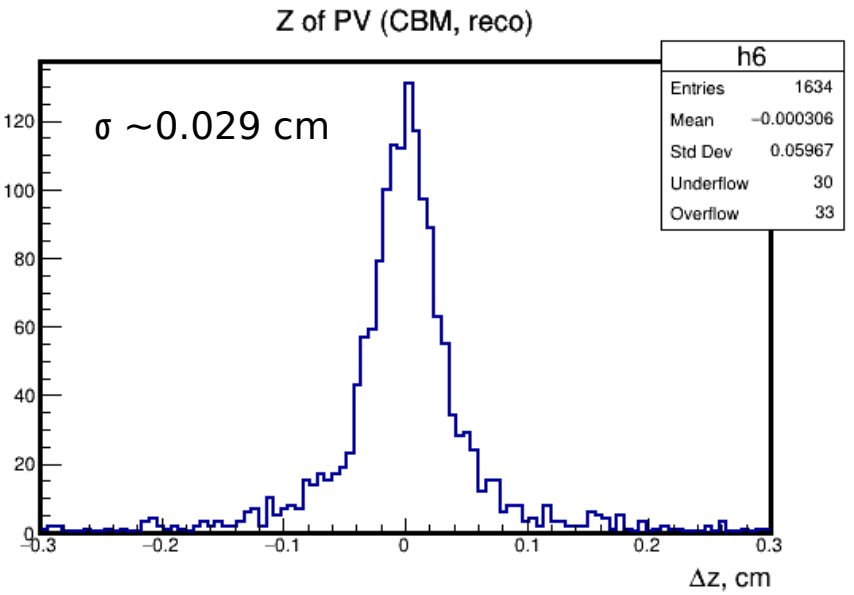


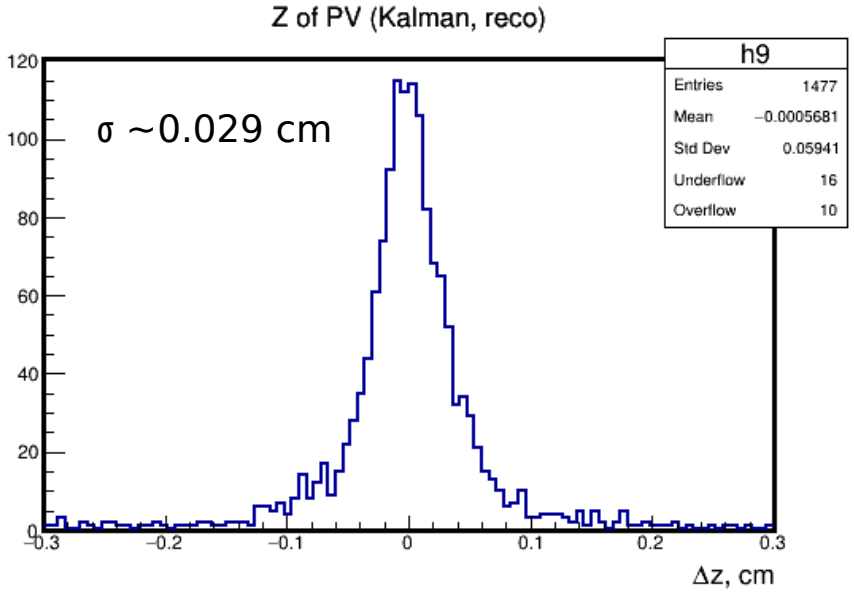


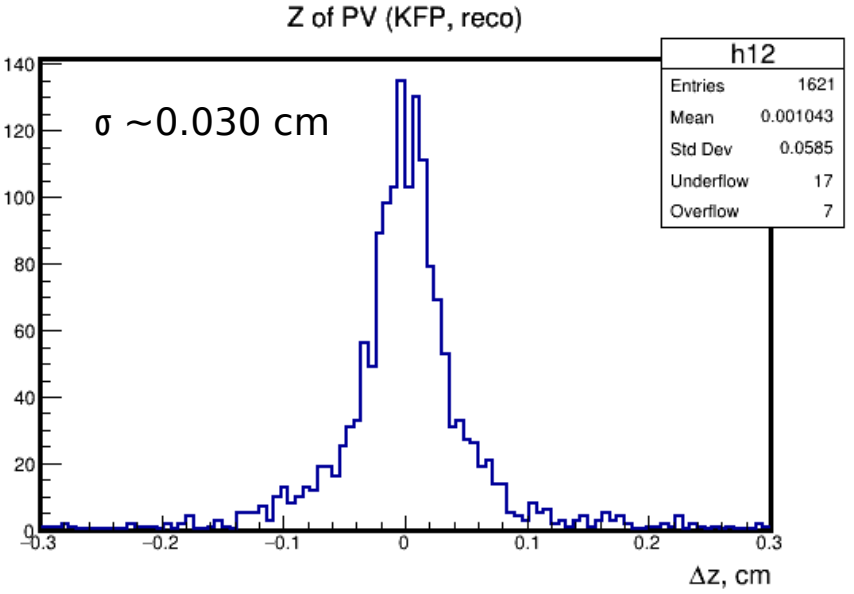
All algorithms show the similar reconstruction precision for x-coordinate of primary vertex

Primary vertex reconstruction precision (2)



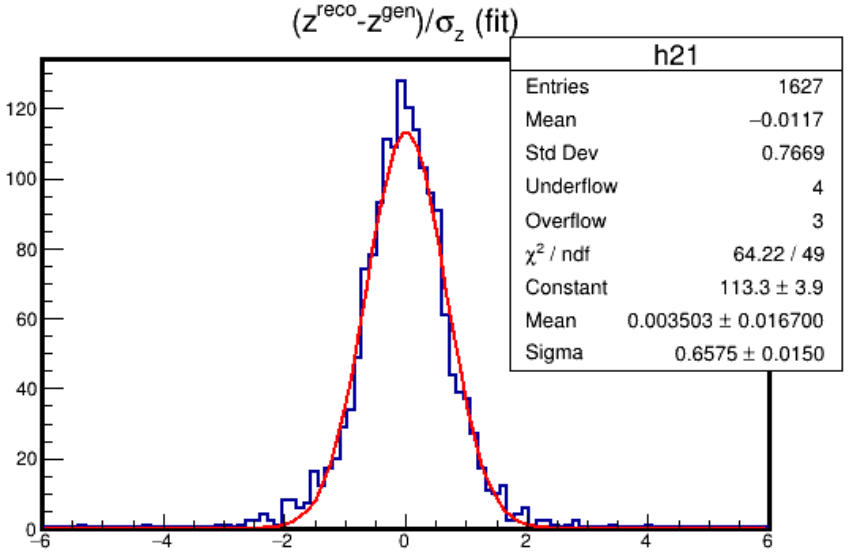


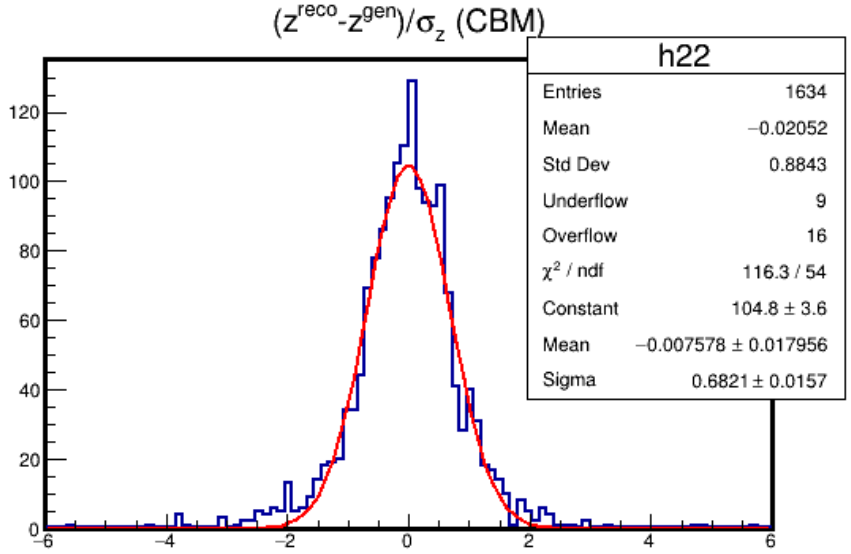


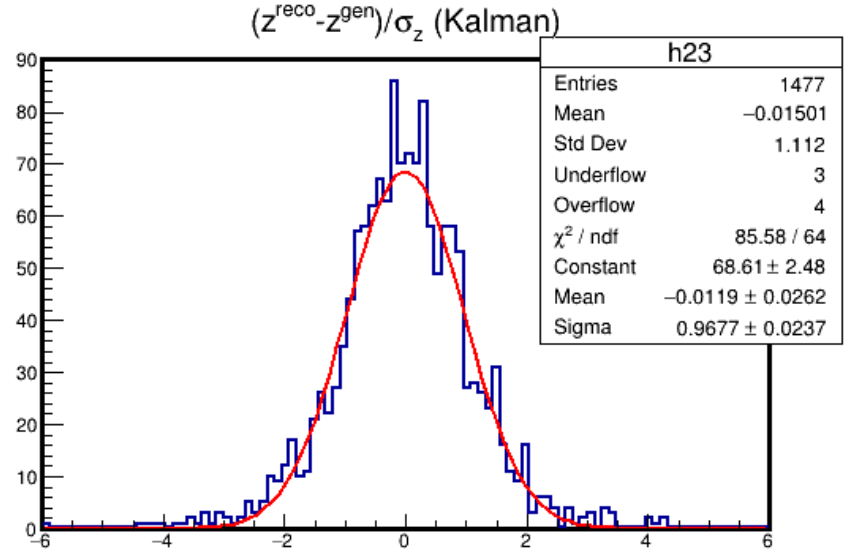


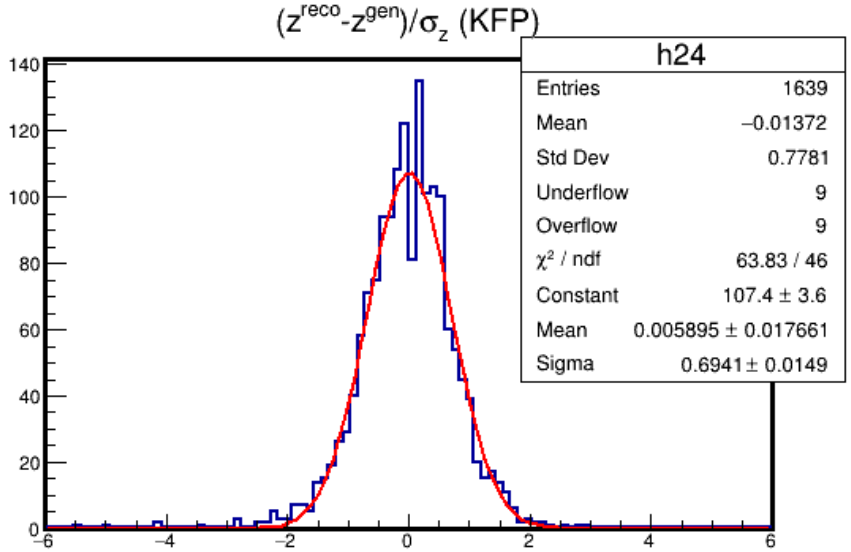
All algorithms also show the similar reconstruction precision for z-coordinate of primary vertex

Pull of z-cordinate with 1 layer of MM detector



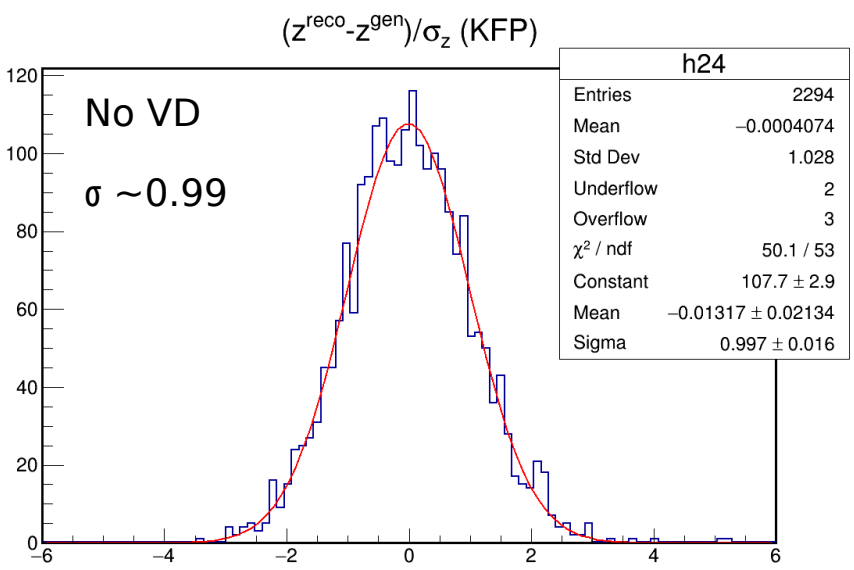


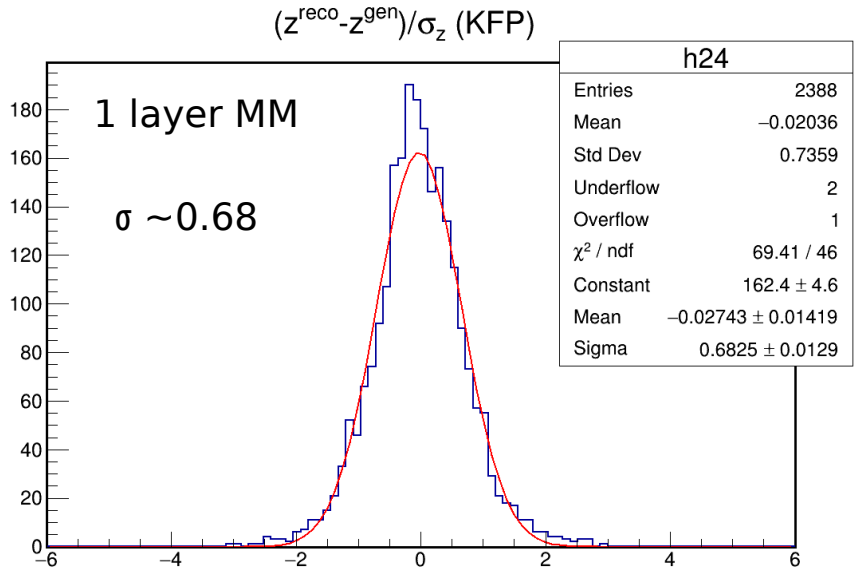


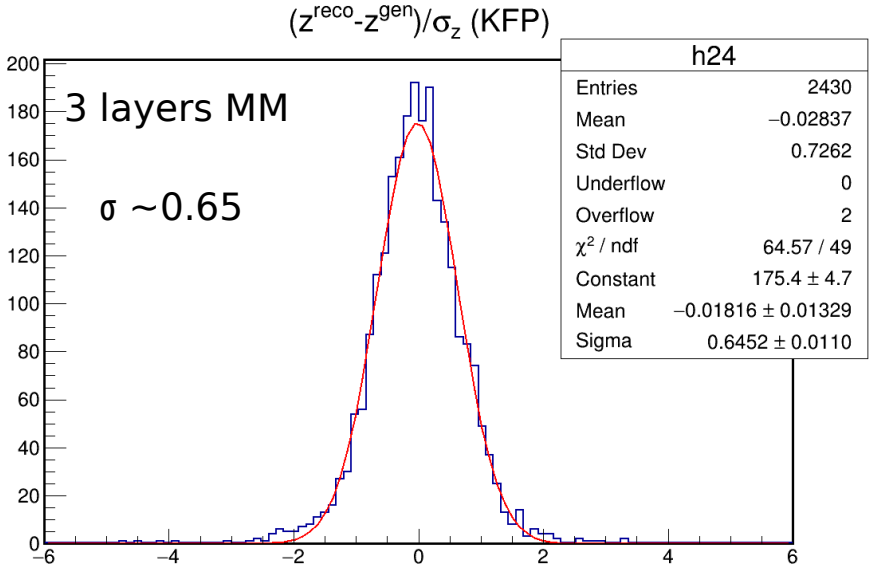


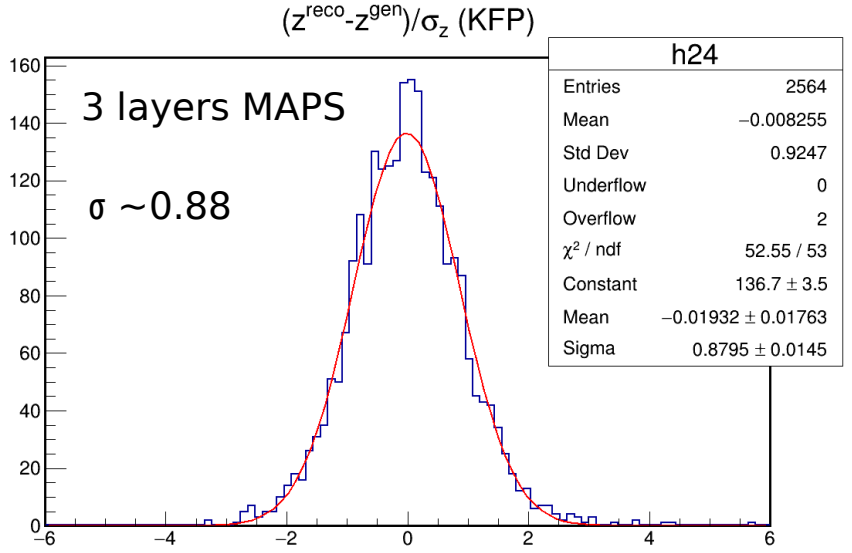
All algorithms, except Kalman, show overestimated (or larger error) of primary vertex coordinates ($\sigma < 1$ of pull distribution). It can be connected with the overestimated error of track parameters in track fitting procedure when the Micro Megas vertex detector is used.

Pull of z-cordinate (different option of vertex detector)









May be it connected with not correct description of the multiple scattering in vertex detector in track fitter program or not correct description of the detector response.
Additional study is necessary for this point.

Summary

1. Three different primary vertex finding algorithms are considered.

2. Vertex finding algorithm on the base of KFParticle package shows the better results (~2 times) with Micro Megas vertex detector and will be include as default finding algorithm in the primary vertex reconstruction procedure.

3. Three different primary vertex fitting algorithms have been also presented:
   base on CBM experiment, so called exact Kalman fitter and from KFParticle package.

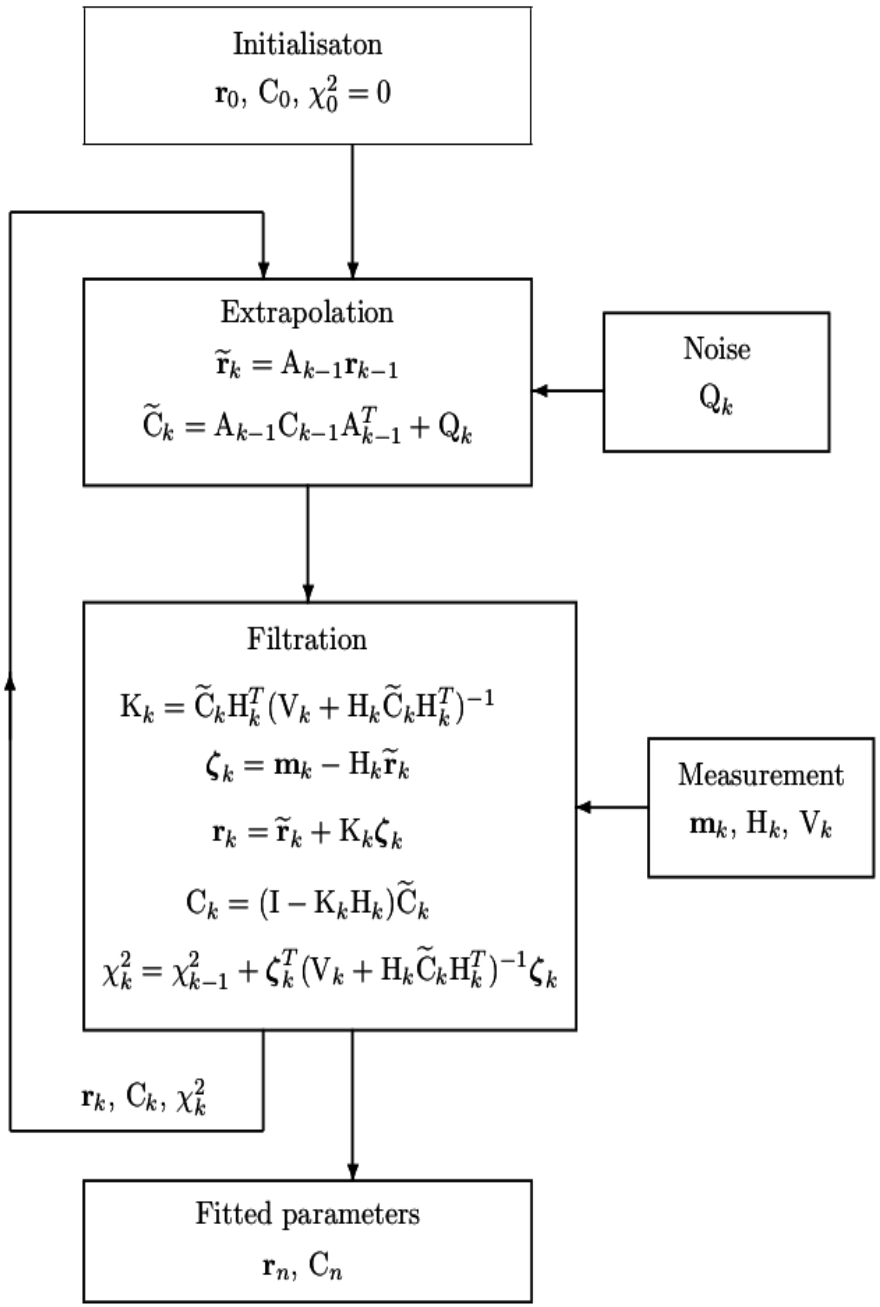
4. All considered fitting algorithms shows good and compatible precision in reconstruction of primary vertex.

5. Next plan is to include all considered vertex finding and fitting algorithms in some unified method:

   – user can select necessary algorithm just set one parameter in the reconstruction macro or
   - user can also repeat desirable reconstruction procedure in your analyzing macro.

Backup

Kalman filter algorithm



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

Extrapolation – changes current estimation of vector r_k upon transfer from (k-1)-th measurement to the k-th measurement

$r_k^t = A_k\, r_{k-1}^t + v_k$, 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

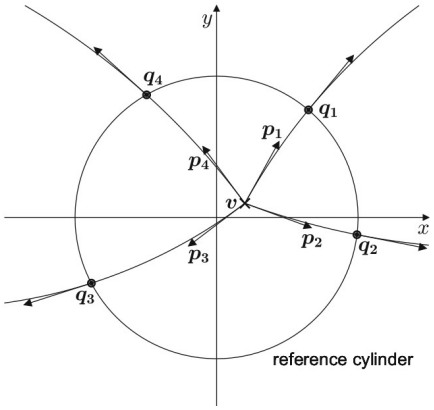
measurement $\mathbf{m}_k$ – a known (measured) quantity with linearly depends on state vector $\mathbf{m} = \mathbf{H} \bullet \mathbf{r}^t + \boldsymbol{\eta}$
H – is a (known) linearly operator represented as a matrix, called **model of measurement; η –** measurement error

Steps extrapolation and filtration 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.

Vertex fitting algorithm

The Kalman filter algorithm is used for primary vertex fitting procedure. This algorithm was formulated as an extended Kalman filter (R. Fruhwirth, Application of Kalman filtering to track and vertex fitting, Nucl. Instr. Meth. A 262 (1987) 444.).



Usually the track parameters $\mathbf{q_i}$ are known on the some reference surface with corresponding covariance matrices $\mathbf{V_i}$, 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 vertex and track parameters $\boldsymbol{v}$ and $\boldsymbol{p_i}$.

The initial information about the vertex position $\boldsymbol{v_0}$, and its covariance matrix $\boldsymbol{C_0}$ are given by some procedure as was described above.

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:

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

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

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

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

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

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. 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;

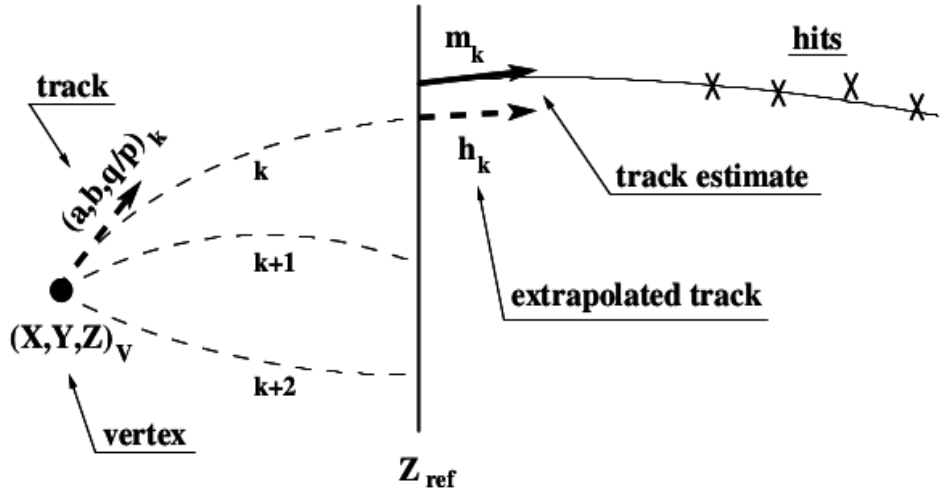
- fixation of the track directions and momenta neglecting uncertainties of these parameters;

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

Current vertex fitting algorithm

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

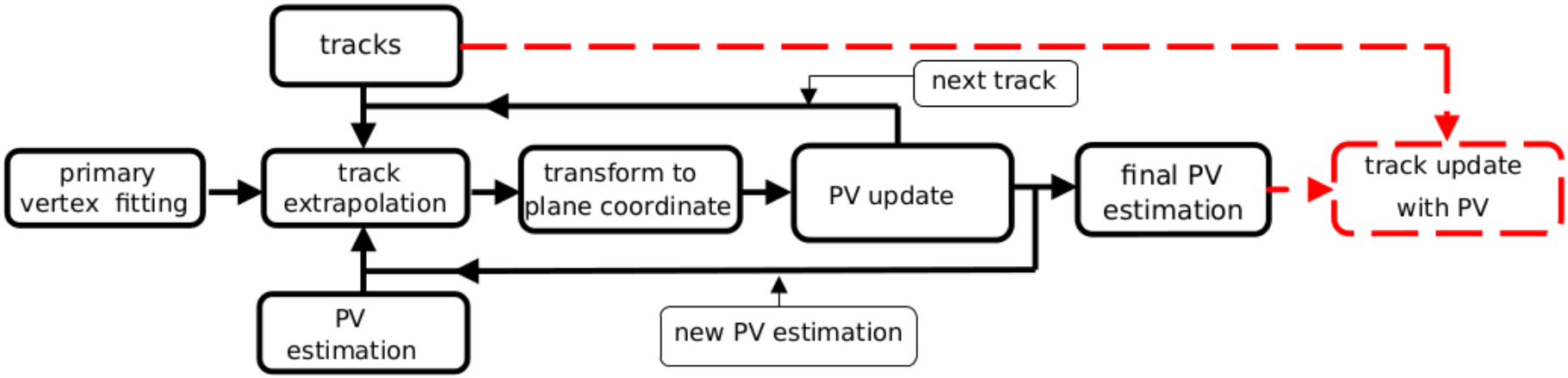
This approach makes it possible to fit the vertex without including the track parameters into the vertex state vector and to simplify the calculations.



$\boldsymbol{r} = (x_v , y_v , z_v)^T$, $\boldsymbol{C_v}$ — 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 fitting algorithm



General schema for current primary vertex fitting algorithm

The current primary vertex fitting algorithm has the some “weak” points:

- does not include track with theta angle +-3^0 around the vertical plane in fitting procedure;
- use the second order curve for track trajectory description;
- track parameters are considered constant during fitting procedure;
- need additional step for final update track parameters at primary vertex (red box).

New algorithm for vertex fitting

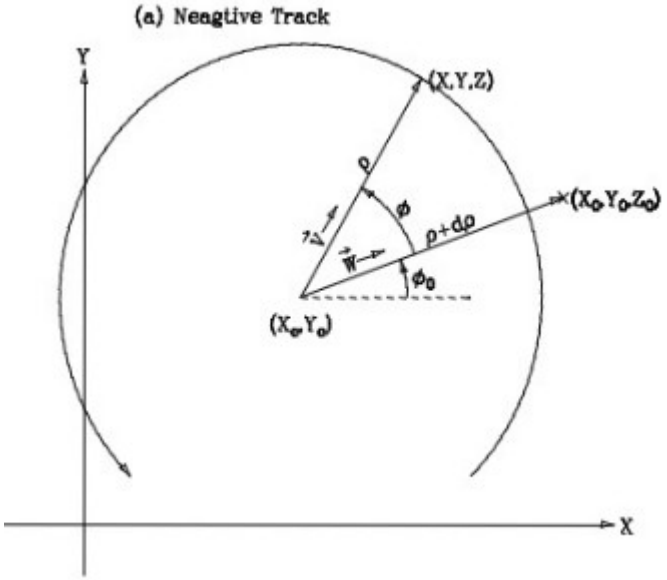
New primary vertex reconstruction algorithm will try to remove the “weak” points of the current algorithm.

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

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

2) 5 local parameters (helix) 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 $x_0 = (x_0 , y_0 , z_0)^T$ is an arbitrarily chosen reference point. If the reference point is fixed, the helix is determined by a 5-component parameters vector $h = (d\rho , \varphi_0 , \kappa, d_z , \tan\lambda)^T$, where $d\rho$ is the distance of the helix from the reference point in the xy plane, φ_0 is the azimuthal angle to the reference point with respect to the helix center, $k = Q/p_T$, d_z is the distance of the helix from the reference point in the z direction, and $\tan\lambda$ is the dip angle.

New algorithm for vertex fitting

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

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

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

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

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

`TMatrixD m_a(5,3);` **$A_i = \partial h_i/\partial v$**

```
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);` **$B_i = \partial h_i/\partial p_i$**

```
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;
```

New vertex fitting procedure

General fitting procedure:

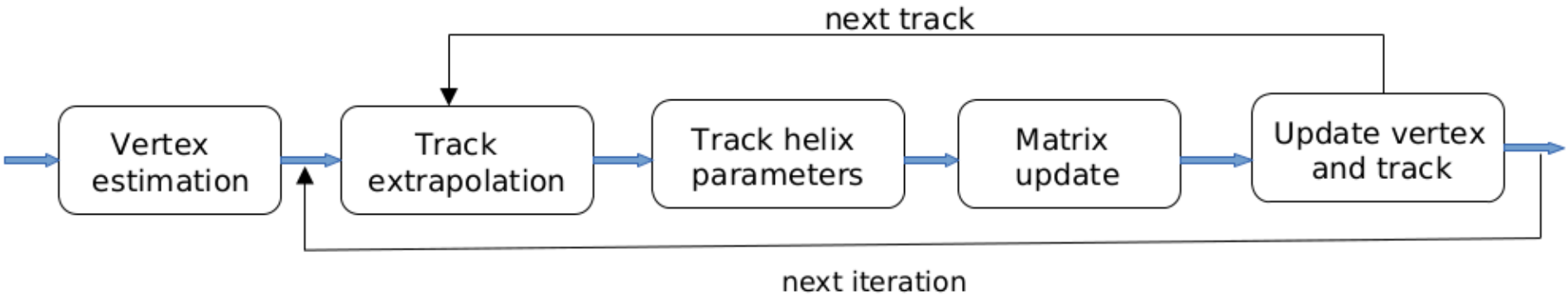
1. determine the preliminary position of the primary vertex (vertex finding);

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;

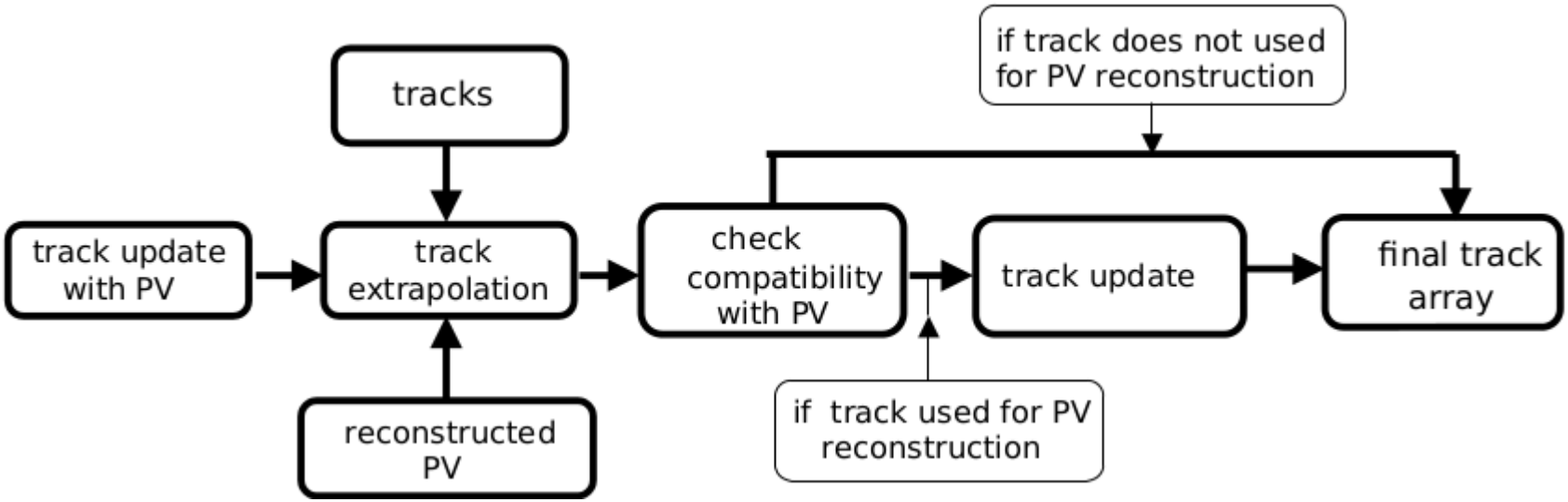


6. do loop over all selected tracks;

7. do this procedure several time (iteration)

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

Update track parameters at primary vertex for current algorithm



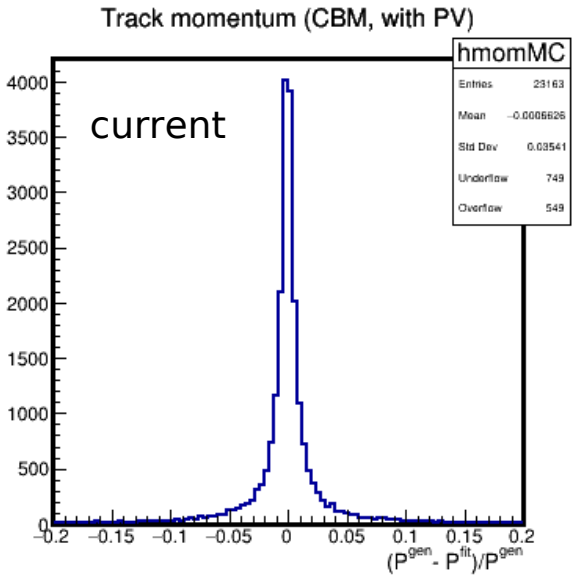
1. track is extrapolated to the reconstructed primary vertex and the track parameters (position, momentum and covariance) are defined at point of closest approach;

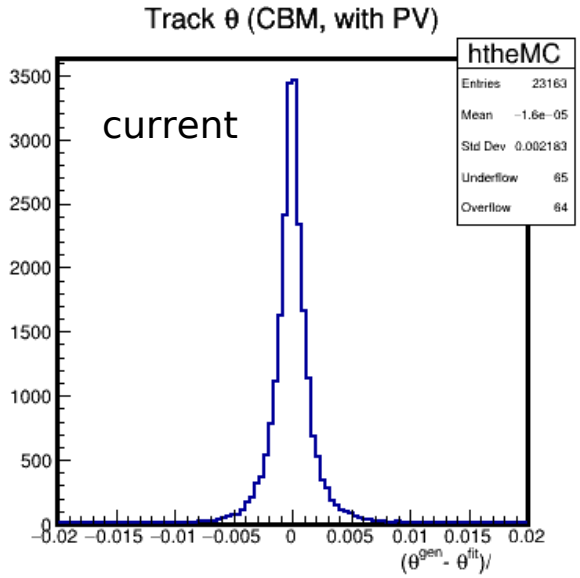
2. if track was used for the primary vertex fitting procedure
   then PV is considered as additional hit point for this track and
   track parameters are updated with Kalman filter procedure;

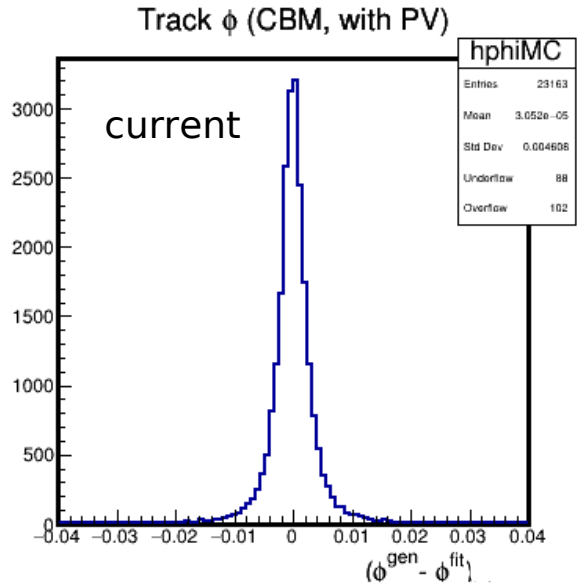
3. put these new track parameters in FinalState – function (GetFinalState());

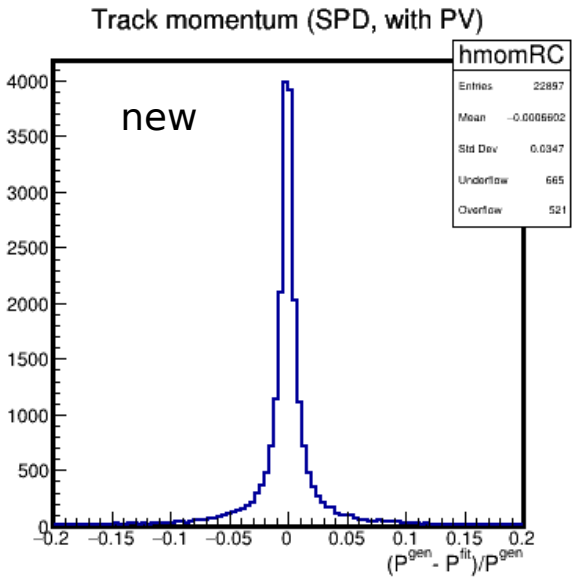
4. if track was not used for the fitting procedure – nothing to do with this track.

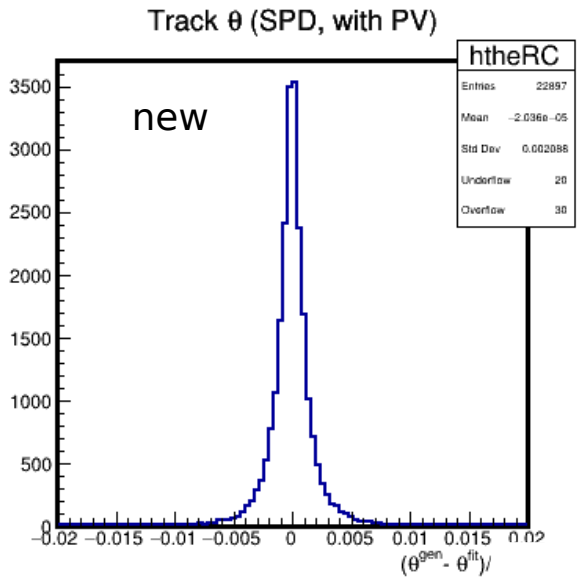
Track parameters precision

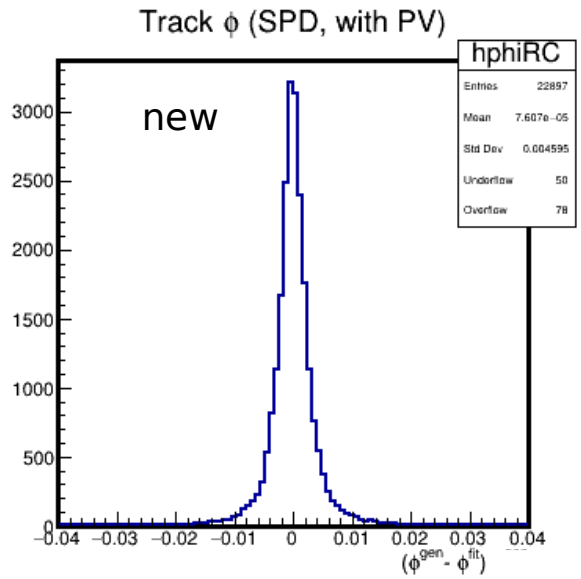












Current primary reconstruction algorithm with additional step for updating track parameters and new primary vertex algorithm show the very similar results

Comparison of two primary vertex 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
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