Vertex and Kinematic Fitting for Physics Analysis

Vishwajeet Jha

Nuclear Physics Division, BARC, Mumbai

Outline:

- Introduction
- Vertex and kinematic Fitting with constraints
- Tests of Vertex & kinematic fitters
- New and ongoing Developments
- Summary & Outlook

Introduction:

- Vertex Fitting: To find the vertex position compatible with reconstructed tracks, Also its errors with fit probability
- Kinematic Fitting : Kinematic relation between particles imposed, fit probability to make cuts
- Improved track parameters for the daughter particles and new covariance matrix
- Track parameter of virtual particles and its covariance matrix

Vertex and Kinematic Fitting Methods:

The constraint equation $H(\alpha) = 0$ is linearized around suitable point (α_a , x_a)

Solution can be obtained by using the least square minimization

$$\chi^2 = (\alpha - \alpha_0)^T V_{\alpha 0}^{-1} (\alpha - \alpha_0) + \lambda (D(\alpha - \alpha_a) + E(x - x_a) + d)$$

Minimize χ^2 with respect to α , x and λ

D is derivative w.r.t α,

E is derivative w.r.t x,

d is the value, $H(\alpha_a , x_a)$

Constraint eqn. $p_{xi}\Delta y_i - p_{yi}\Delta x_i - (a_i/2)(\Delta x_i^2 + \Delta y_i^2) = 0$

$$\Delta z_i - (p_{zi}/a_i) \sin^{-1}[a_i (p_{xi}\Delta x_i + p_{yi}\Delta y_i)/p_{Ti}^2] = 0$$

Kinematic Fitting Outputs :

New Track parameters : $\alpha = \alpha_0 - V_{\alpha 0} D^T \lambda$

New Covariance Matrix : $V_\alpha = V_{\alpha 0} - V_{\alpha 0} D^T V_D V_{\alpha 0} + V_{\alpha 0} D^T V_D E Vx E^T V_D D V_{\alpha 0}$

New Vertex Position : $x = x_0 - V_{x0} E^T \lambda$

Vertex Covariance matrix : Vx & $cov(\alpha, x) = - V_{\alpha 0} D^T V_D E Vx$

Track parameters for vertexed particle : x_V , p_V

Robust Vertex Fitting :

- Track parameters & covariance matrix (Rho TCandidate)
- Find Good start vertex (POCA Finder)
- Track propagation to reference point
- Compute kinematic matrices
- Iterative minimization
- Output daughter candidates and vertices

Implemented in PndKinVtxfitter

Kinematic Fitting with Constraints:

One or more constraints can be used in combination (PndKinFitter) :

Kinematic constraints:

i) 4 vector constraint : (Add4MomConstraint (TLorentzvector lv)

ii) momentum constraint (AddMomConstraint (Tvector3 v)

iii) Total energy /Momentum (AddTotEConstraint (double E)

iv) Mass constraint (AddMassConstraint double mass)

Implemented in PndKinFitter

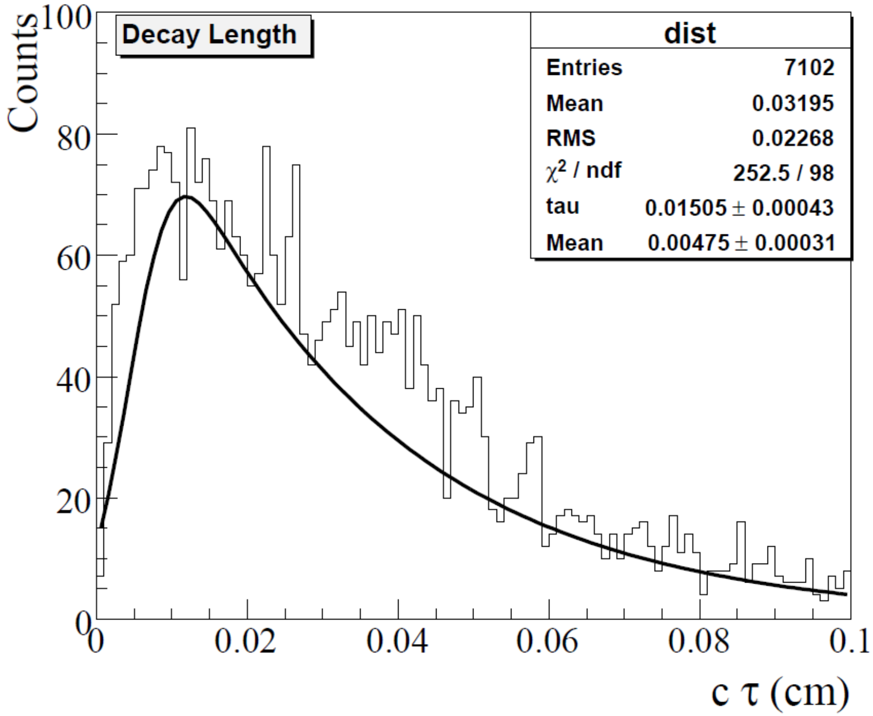
Any other Constraint by user

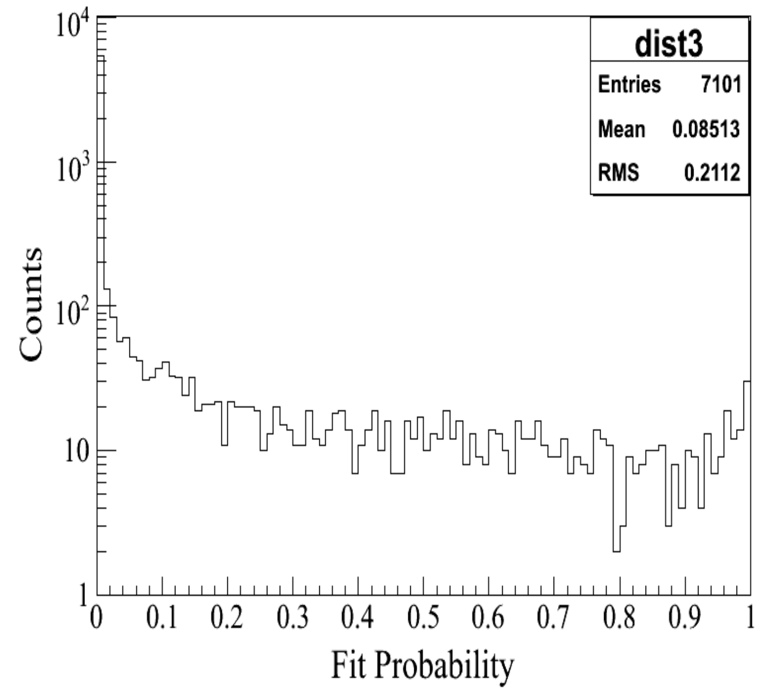
User Analysis Code:

```
for (j=0;j<dm.GetLength();++j) {
PndKinVtxFitter vtxfitter(dpm[j]); // *** instantiate the vertex fitter; input is the object
to be fitted
vtxfitter.Fit(); // *** perform fit
TCandidate *dmfit = vtxfitter.FittedCand(dm[j]); // *** get the fitted candidate
TVector3 dmvtx = dmfit->Pos(); // *** and the decay vertex position
double chi2_vtx = vtxfitter.GlobalChi2(); // *** and the chi^2 of the fit
int dgf =vtxfitter.GetDof(): // Degree of freedom
hdmvtx_chi2->Fill(chi2_vtx);
if ( chi2_vtx<2 ) // *** if chi2 is good enough, fill some histos
{hdm_vf->Fill(dmfit->M());hdmpos->Fill(dmvtx.X(),dmvtx.Y());}}
```

```
for (j=0;j<jpsi.GetLength();++j) {
PndKinFitter mfitter(jpsi[j]); // *** instantiate the vertex fitter; input is the
object to be fitted
mfitter.AddMassConstraint(3.097); // *** set the fixed mass for the
constraint
mfitter.Fit();
double chi2m = mfitter.GlobalChi2(); // *** get the chi2 of the fit
if (chi2m<2) hjpsim_mcfs->Fill(jpsi[j].M()); // *** if chi2 is sufficiently
good fill histogram with _unfitted_ mass
}
```

Vertex Fitter Tests (I):





Fitted Ds proper Lifetime = 150 μm, PDG Value =147 μm

Vertex Fitter Tests (II):

$\sigma_v/\mu m$	Poca	PRG	VtxKin
x	47.3	57.9	44.3
y	45.6	51.6	42.9
z	88.4	94.9	90.2

Table: D^0

$\sigma_v/\mu m$	Poca	PRG	VtxKin
x	56.9	86.1	46.9
y	56.3	84.8	46.1
z	113	125	93.2

Table: D^+

$\sigma_v/\mu m$	Poca	PRG	VtxKin
x	47.5	58.3	44.6
y	46.3	51.9	43.5
z	88.4	94.1	89.3

Table: $\overline{D^0}$

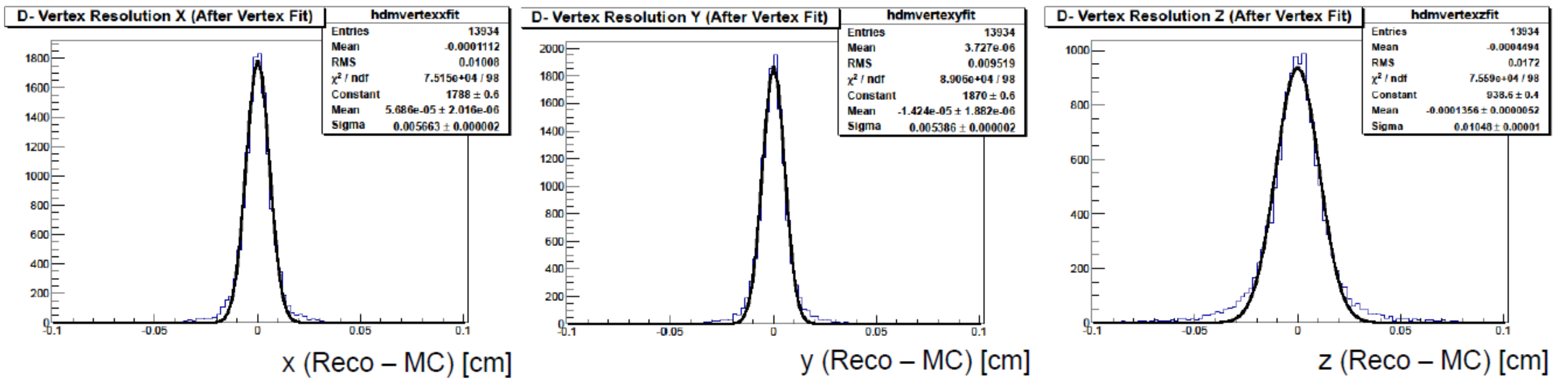
$\sigma_v/\mu m$	Poca	PRG	VtxKin
x	57.4	85.3	46.3
y	56.0	84.4	45.7
z	110	123	94.1

Table: D^-

Tests by (R. Kliemt):

Vertex Fitter Tests (III)

$\bar{p}p \rightarrow \Psi(3770) \rightarrow D^+D^- \rightarrow K^-\pi^+\pi^+ K^+\pi^-\pi^-$



σ_x = (56.63±0.02) μm **σ_y = (53.86±0.02) μm** **σ_z = (104.8±0.1) μm**

Questions and comments :

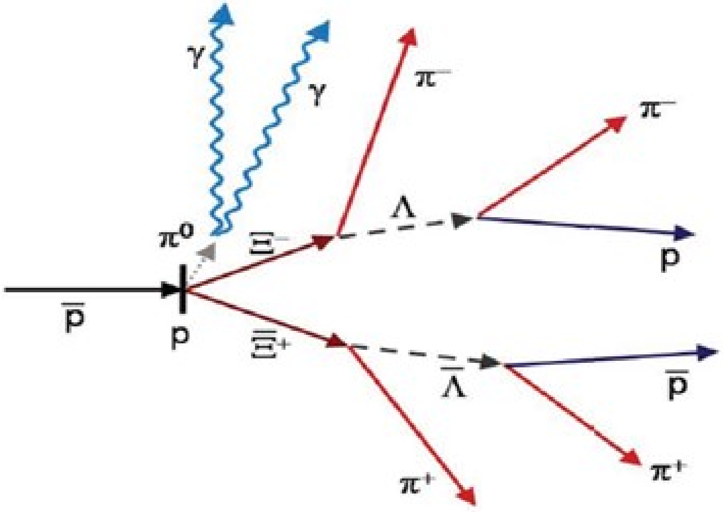
What happens when multiple fitters are applied?

What happens when fitting decay trees with several levels?

A quick way to fit a whole candidate list and retrieve the best fit or a fitted candidate list

M. Mertens

Decay Tree Fitting:



Four class of particles (objects) :

i) Reconstructed Track

ii) Photons reconstructed as cluster

iii) Composites or virtual particles :

a) prompt decay (resonances)

b) Macroscopic decay length (composites)

iv) Missing particles

Fitting the Decay Tree I:

Sequential (Leaf by leaf based approach):

Constraints applied sequentially to build the decay chain. In the bottom-up approach we generate new composite particles /resonances along the way

Composite has all the information of daughter tracks in linear approximation.

Efforts to optimize the Tree and node navigation (by Ralf K.)

Fitting the Decay Tree II:

Global Approach:

All constraints are applied simultaneously for complete decay tree.

Better treatment of non-linearities and track-track correlation

Large Matrices need to be inverted

Progressive fit based on Kalman filter can be used.

(Some cases absolutely essential

Decay tree with $K_s \to \pi^0 \pi^0$)

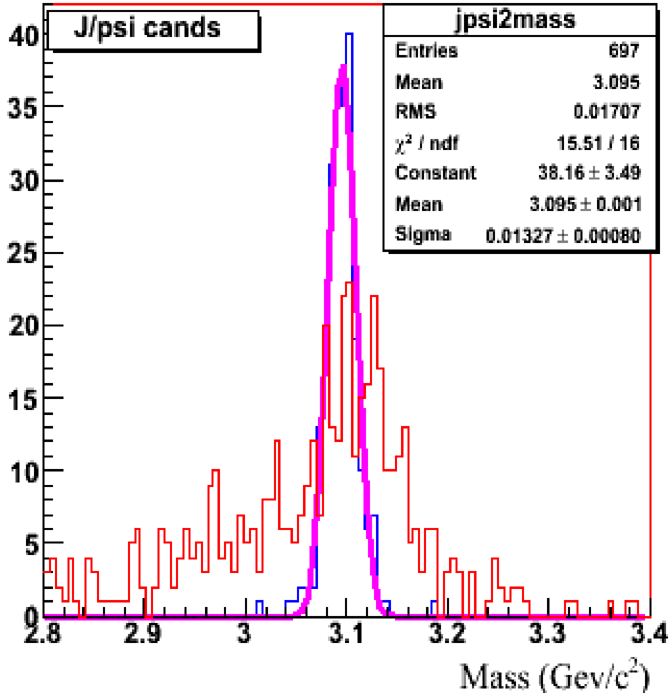
Sequential Fitting :

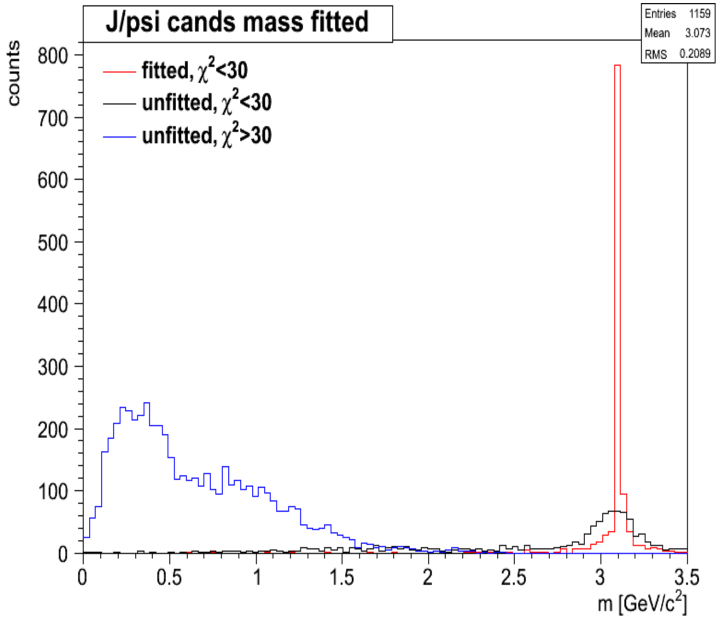
Constraints applied sequentially to build the decay chain

1st step ppbar -> J/ψ π+π-

4 Momentum fit for the ppbar system :

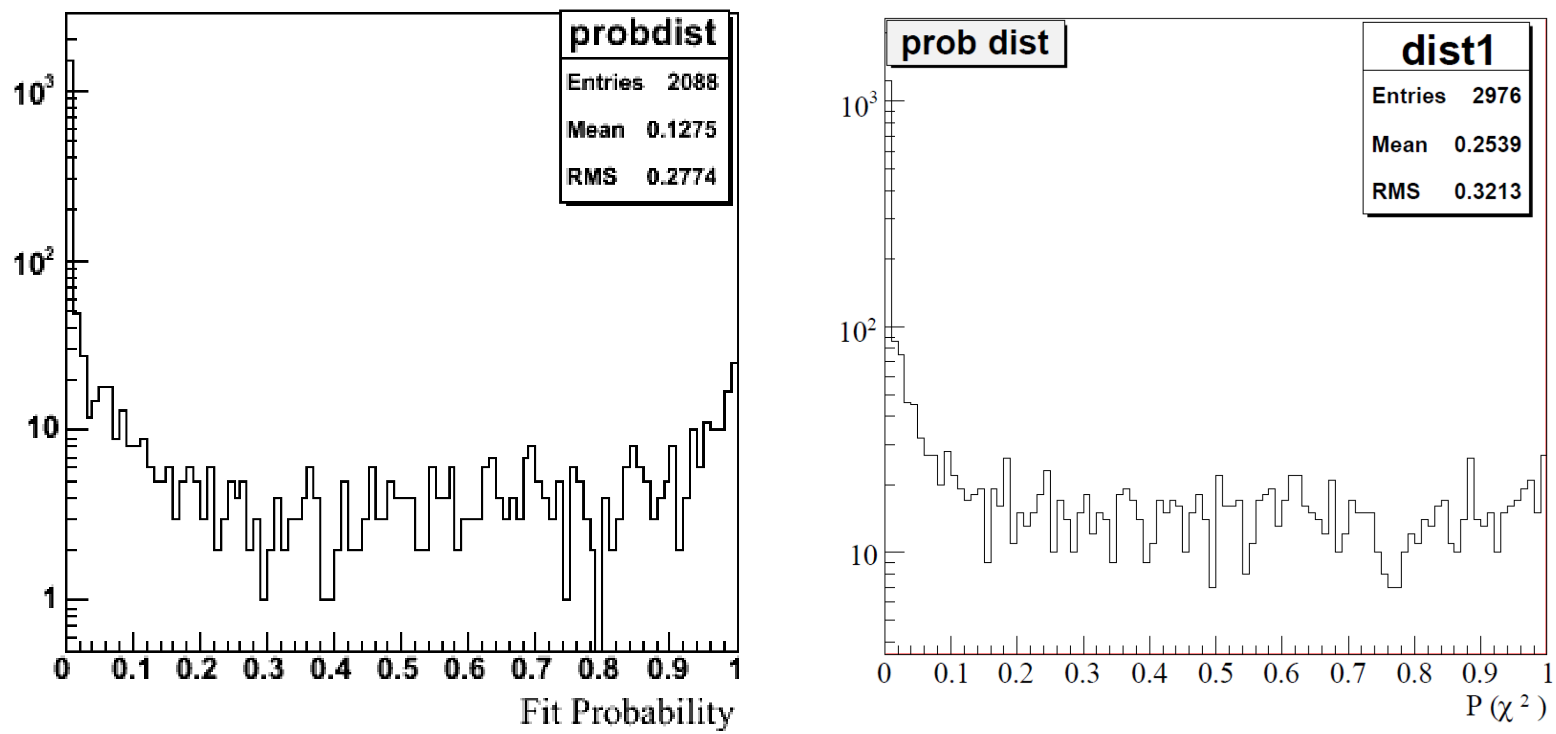
Channel ψ(2S) → J/ψ $\pi^+\pi^-$,
mass constraint J/ψ





Sequential Fitting :

2nd Step : Probability of the Vertex fit for the ppbar vertex



Fit Probability before and after Vertex fit

New Developments I:

1. Building virtual particles (From vertex Fit) :

Virtual particles with new track parameters are built (x_V, p_V):

$x_V = x$, , $p_V = A\,\alpha + B\,x$

Covariance matrix :

$$V_{\alpha_V} = \begin{pmatrix} V_{p_V} & \mathrm{cov}(p_V, x_V) \\ \mathrm{cov}(x_V, p_V) & V_{x_V} \end{pmatrix}$$

$$V_{p_V} = AV_\alpha A^T + A\mathrm{cov}(\alpha, x)B^T + B\mathrm{cov}(x, \alpha)A^T + BV_x B^T$$

$$\mathrm{cov}(p_V, x_V) = A\mathrm{cov}(\alpha, x) + BV_x$$

$$V_{x_V} = V_x$$

New Developments II:

Efficient Start vertex finder in case of multiple tracks used in vertex fit

Neutrals and missing particles (Ongoing):

Neutrals and missing particles (π^0) are not used for the vertex fit

They need to be however used for building virtual particles.

In principle, same formalism as before is used (without E matrix)

Secondary Vertex:

Pointing constraint :

Secondary Vertex resolution can be improved by imposing a pointing constraint

PndKinVtxFitter :: AddPointingConstraint (const TCandidate& head, const VAbsVertex& pVtx)

Decay Proper Time Fitter :

Proper decay time of any particle can be determined by using all the track parameter information (at secondary vertex point) and the beam information at the primary vertex point

(New Implementation for pandaroot : PndProperTimeFitter)

Progressive Decay Tree Fitter:

The whole decay tree is considered at once

The constraints are applied progressively

Least Square minimization and casting in terms of K_k (Gain Matrix)

$$\chi^2 = (\alpha - \alpha_{k-1})^T \; C_{k-1}^{-1} (\alpha - \alpha_{k-1}) \; + \; r(\alpha_k)^T V_k^{-1} r(\alpha_k)$$

$$\alpha_k = \alpha_{k-1} + K_k^{-1} r(\alpha_k)^{k-1}$$

Gain Matrix K_k

$$C_k = (1- K_k H_k) \; C_{k-1} \; (1- K_k H_k)^T + K_k V_k^{-1} K_k^T$$

Implementation just started

Summary :

Vertex Fitters have been implemented

Kinematic fitters with many constraints have been included.

Tests of their performance have been made.

Full Decay tree fitting is an ongoing activity (requires

better synergy with Rho package)

Outlook :

Progressive method for full decay tree fitting is being started.

Developing the other fitter functionalities for physics analysis

धन्यवाद

dhanyawad