LCFIPlus Vertexing Module in FCCAnalyses

Freya Blekman, Florencia Canelli, Armin Ilg,
Kunal Gautam, Eduardo Ploerer

FCC Physics Performance Meeting | 16.05.22

Motivation

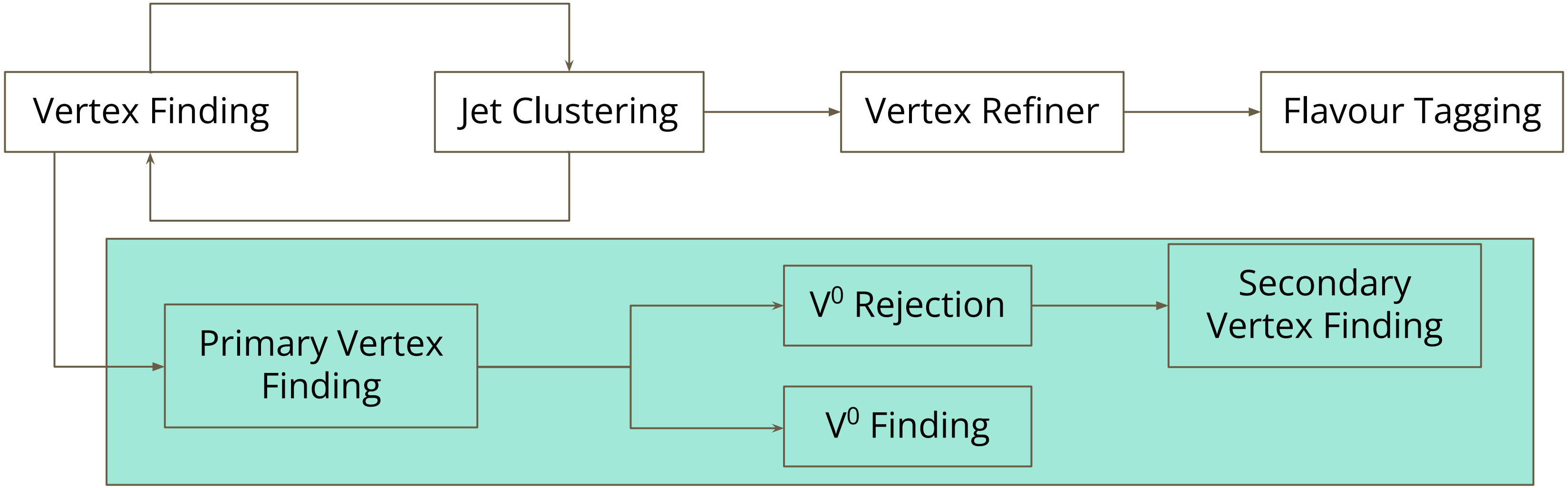
- No Secondary Vertex finding utilities in FCCAnalyses at the moment.
- Very important in a lot of studies:
  - Jet flavour tagging
  - Rare decays
  - Vertex detector performance

Available Resources

- Track information using the fast simulation of the IDEA detector concept.
- Two vertex fitters are available in FCCAnalyses.
- Primary vertex finder has recently been added.

What is LCFIPlus

- A flavour tagging framework developed for linear colliders (see 1506.08371)
- Includes a vertexing module

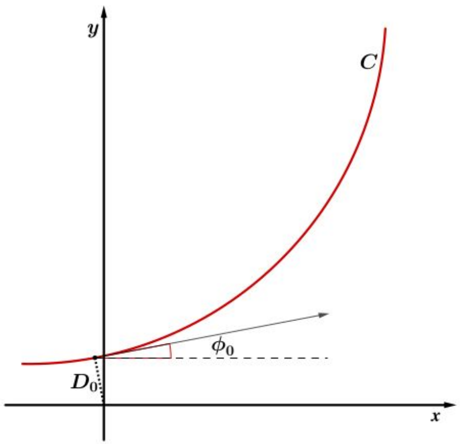


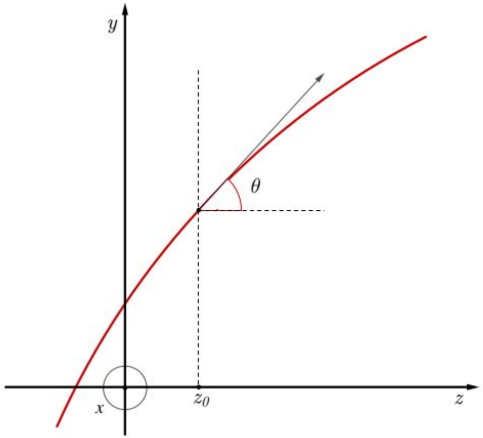
Track Representation in FCCAnalyses

Franco Bedeschi

Tracks are described by canonical track parameters (D_0, φ_0, C, z_0, $\cot(\theta)$) & full covariance matrix

- Transverse & longitudinal IP, half-curvature, azimuthal angle & cotangent of polar angle at point of closest approach
- Uses the fast-simulation of the IDEA detector concept



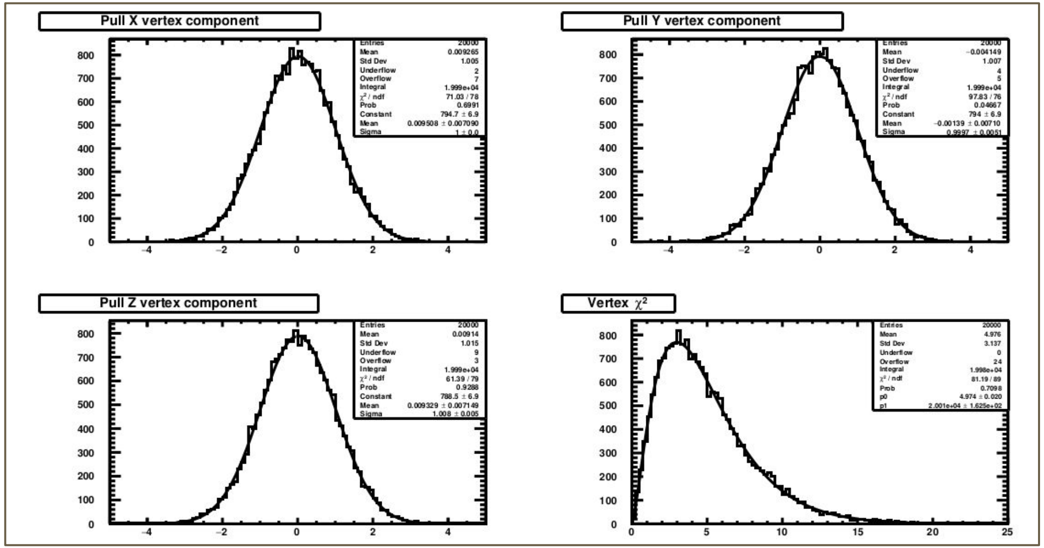


Vertex Fitter in FCCAnalyses

Franco Bedeschi

- Track parameters of all the tracks are used as the initial points for the fit
- The track parameters are updated to make all tracks pass through a common vertex while minimising the χ^2
  - Deviations in track parameters are controlled by the covariance matrix
  - Track projections are used as constraints for the χ^2 fit
  - The track helix in cartesian coordinates is used in parametric form
- BS constraint is optional.

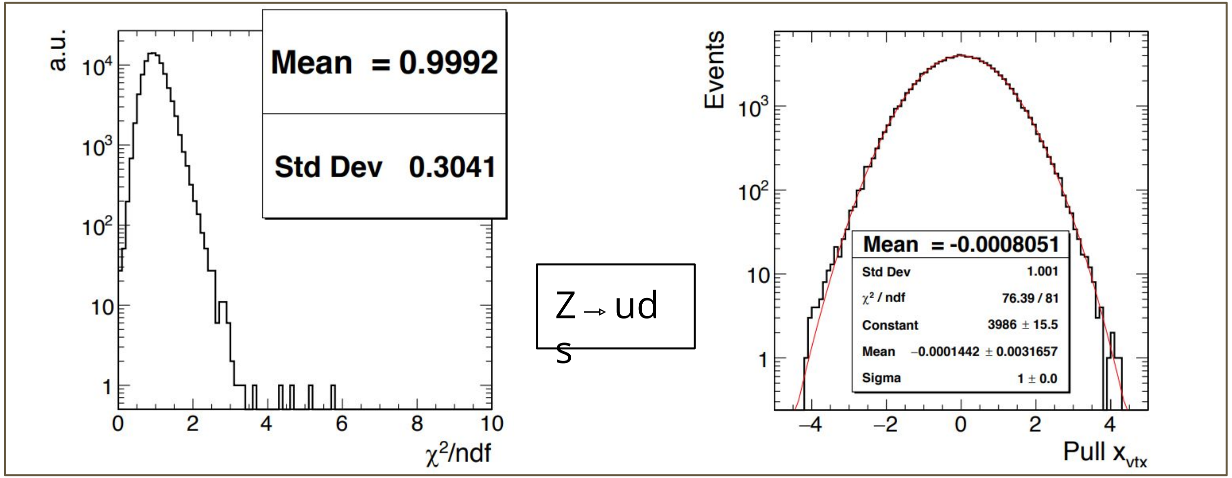
20,000 vertices:
- 4 tracks
- R_V (1-5 mm)
- |p| (1-20 GeV)



Primary Vertex Finder

Emmanuel Perez

- Tear-down approach
  - Vertex fit using all tracks - beam spot constraint ‘on’
  - Remove track with the highest contribution to χ^2
  - Repeat step 2 until no track with contribution higher than a threshold (set at 25) remains
- Beam spot size (4.5 µm, 20 nm, 0.3 mm) in (x, y, z)



From a talk by Clement & Emmanuel

Secondary Vertex Finder

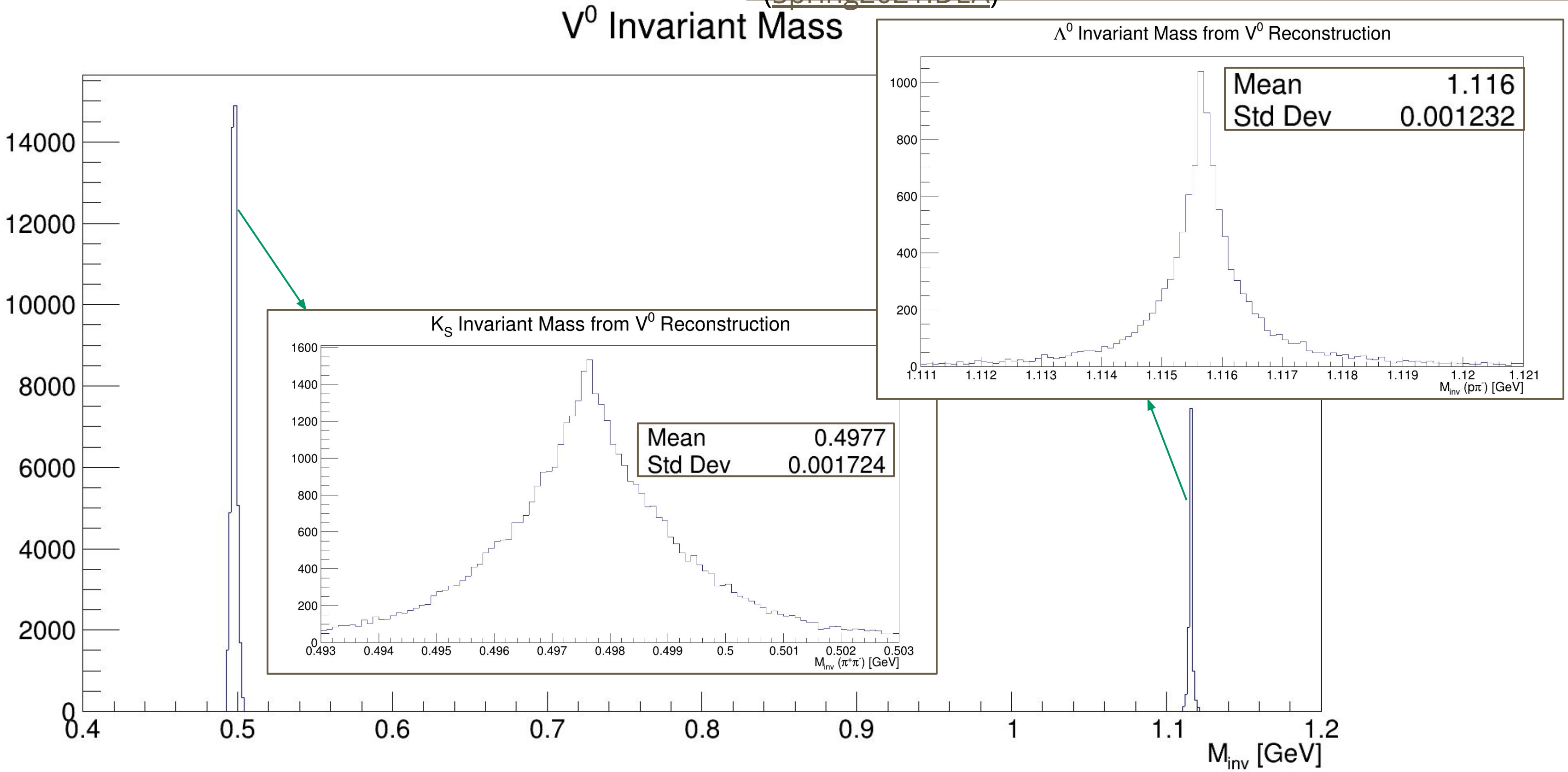
V^0 Rejection

- Especially useful for heavy jet flavour tagging - reduces contamination from light jets
- Three processes considered: $K_S \rightarrow \pi^+\pi^-$; $\Lambda^0 \rightarrow p\pi^-$; $\gamma_{conv} \rightarrow e^+e^-$
- Also slightly modified to give V^0 vertices along with PID & invariant mass

	K_S		Λ^0		γ_{conv}	
	tight	loose	tight	loose	tight	loose
Mass [GeV]	[0.493, 0.503]	[0.488, 0.508]	[1.111, 1.121]	[1.106, 1.126]	< 0.005	< 0.01
r [mm]	> 0.5	> 0.3	> 0.5	> 0.3	> 9	> 9
$\hat{p}.\hat{r}$	> 0.999	> 0.999	> 0.99995	> 0.999	> 0.99995	> 0.999

V^0 Finder Performance

100K dijet $Z \to uds$ samples [E_{CM}=91.2GeV] (Spring2021IDEA)



Finding Vertex Seed and Adding Tracks

Finding the Vertex Seed

- Every possible track pair is fitted to a vertex.
- Out of all the two-track vertices that pass a selection criteria, the one with the lowest χ^2 is chosen as the vertex seed.

Adding Tracks to the Vertex Seed

- The track that forms a vertex with the lowest χ^2 with this seed vertex and passes the selection criteria is added to the seed vertex.
- Keep adding tracks to this vertex until no more tracks pass the selection.
- Store the vertex and remove the tracks forming this vertex from the set of tracks

Continue until no more vertex seeds can be formed.

Selection Constraints

For finding the Vertex Seed

- M_{inv} < threshold (10 GeV)
- M_{inv} < sum of energy of both tracks
- V^0 rejection (with loose constraints)
- Vertex displacement and sum of track momentum in the same hemisphere
- χ^2 < threshold (9)

For adding tracks to the vertex seed

- All of the above constraints except for V^0 rejection
- Additional constraint on new track:
  - χ2 contribution of the track < threshold (5)

How to use in FCCAnalyses

Planned to be pushed to the central repo soon

Finding SVs in an event

- .Define("SV_evt1", "VertexFitterSimple::get_SV_event(ReconstructedParticles, EFlowTrack_1, PrimaryVertexObject, IsPrimary)")
  - Interface 1: Find PV before using.
- .Define("SV_evt2", "VertexFitterSimple::get_SV_event(ReconstructedParticles, EFlowTrack_1, SecondaryTracks, PrimaryVertexObject)")
  - Interface 2: Find PV and define non-primary tracks before using.

Finding SVs in jets

- .Define("SV_jet", "VertexFitterSimple::get_SV_jets(ReconstructedParticles, EFlowTrack_1, PrimaryVertexObject, IsPrimary, jets, jetconstituents)")
  - Find PV and perform jet-clustering before using.

Thresholds and whether to perform V^0 rejection can also be set by the user, otherwise set to default values.

Some functions have also been added in VertexingUtils to extract and calculate some SV properties.

FCCAnalysesSV & FCCAnalysesV0

- Vertex information is stored in **FCCAnalysesVertex** struct.
- For simplicity, defined two new structs to store SV & V^0 information.

FCCAnalysesSV

- Vector of FCCAnalysesVertex
- Vector of number of SVs per jet

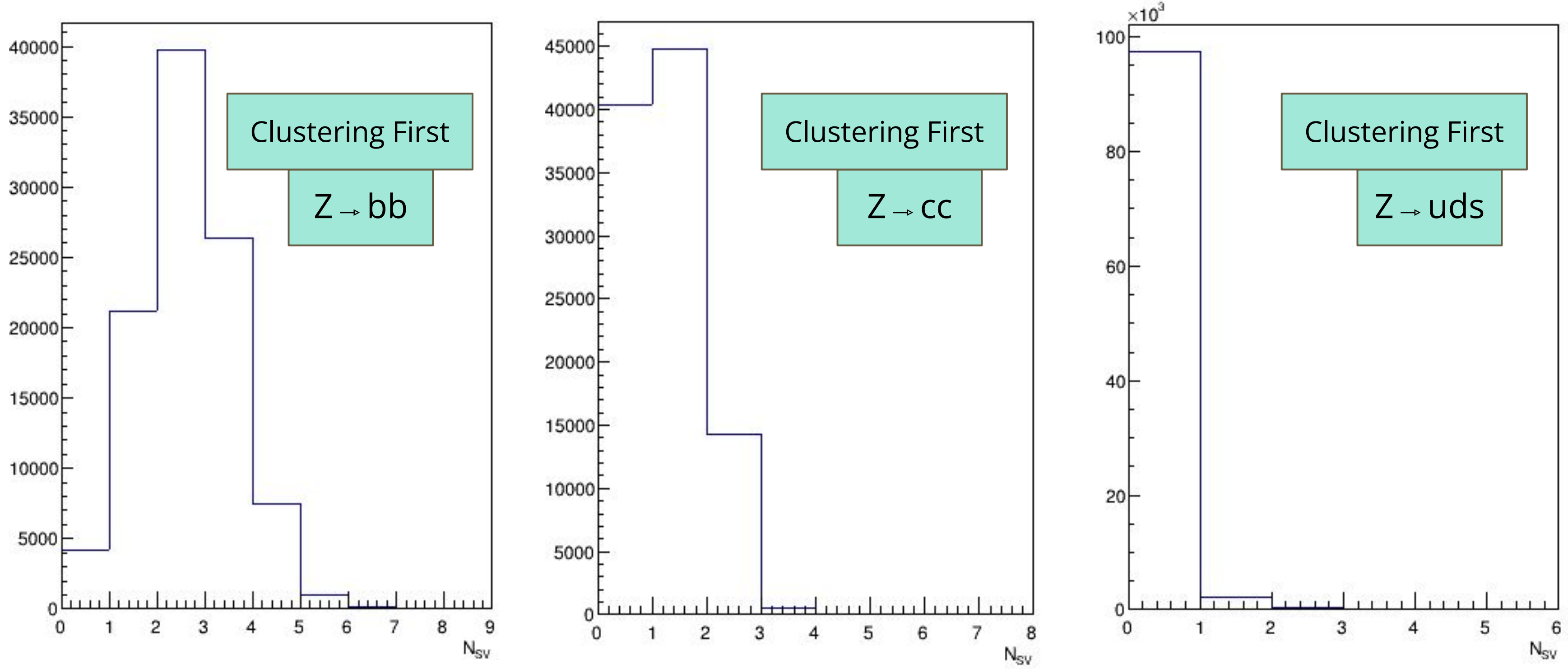
FCCAnalysesV0

- Vector of FCCAnalysesVertex
- Vector of PDG IDs of V^0s
- Vector of invariant mass of V^0s

Would like to have a discussion on a more efficient (and consistent with framework criteria) way of storing the information before making a PR

SV Finder Performance

100K each category dijet Z→qq samples [E_{CM}=91.2GeV]
q:{b, c, uds} (Spring2021IDEA)

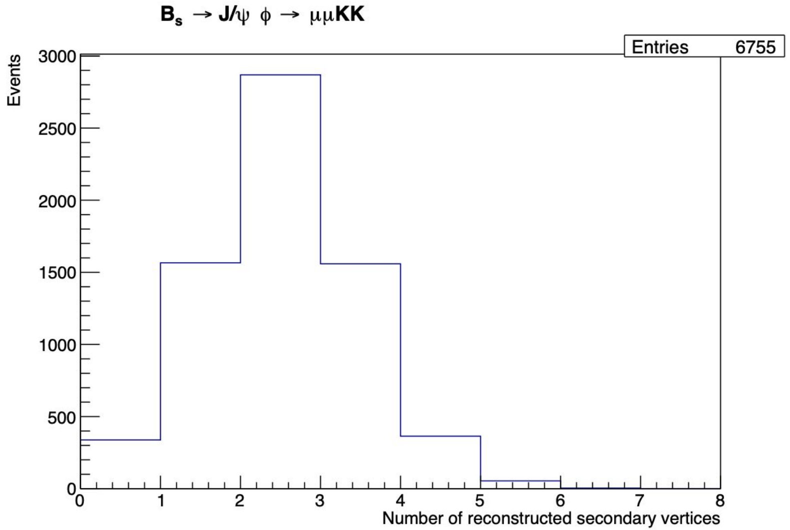


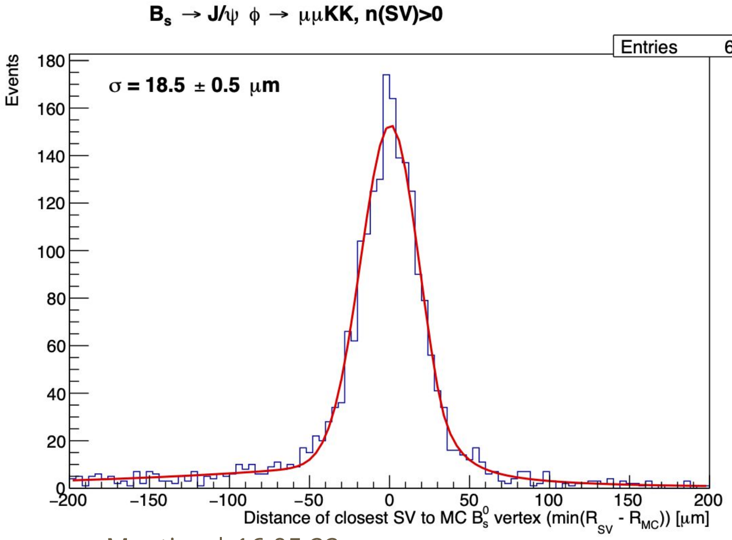
First SV Finder Performance Estimate

Adapting analysis_Bs2JpsiPhi.py:

10K $B_s \to J/\Psi + \varphi \to \mu\mu$ KK events

- 6.75k events where $B_s \to J/\Psi + \varphi \to \mu\mu$ KK decay can be found completely in MC
- 6.4k events with reconstructed SV additionally
- Check min of distance between any SV and MC B_s vertex, fit with two crystal ball functions (no further selection applied)



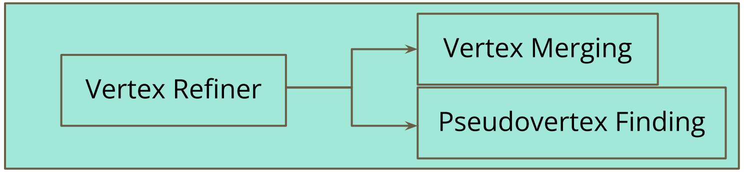


Summary

- The first implementation of an SV finder in FCCAnalyses based on the LCFIPlus framework.
- Used the available vertex fitter in FCCAnalyses.
- Most of the parameters can be chosen by the user.
- Updated implementation of V0 rejection to reconstruct and PID K_S and Λ^0.

Outlook

- Many opportunities for improvement:
  - Parameters can be tuned; currently using the values presented by the LCFIPlus authors.
  - Vertex refining can be included
  - Test using the ACTS vertex fitter
- Would like any suggestions or advice.



So long,

and **thanks** for all the fish

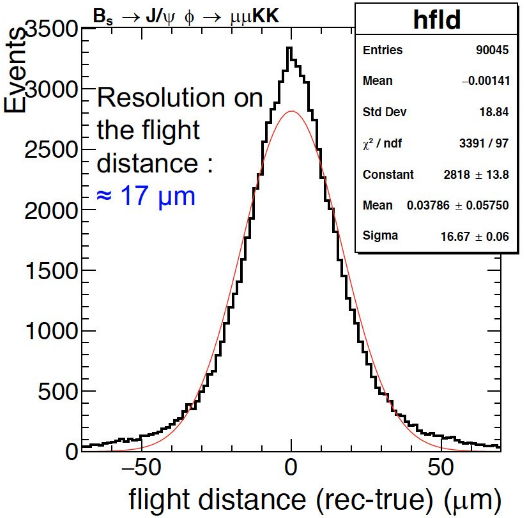
Back-up

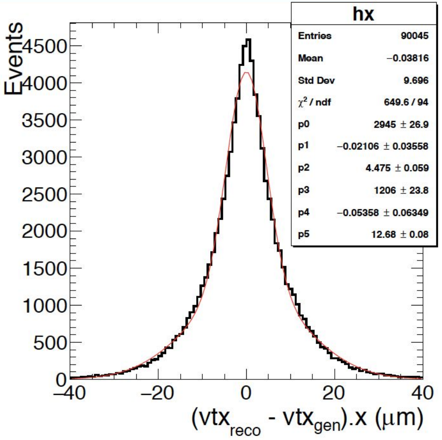
$B_s \rightarrow J/\Psi + \varphi \rightarrow \mu\mu$ KK performance with reco-match tracks

From a talk by Clement & Emmanuel

Bs → J/ψ Φ → μμKK: resolutions

Example: resolution in x, fitted with the sum of two Gaussian functions.

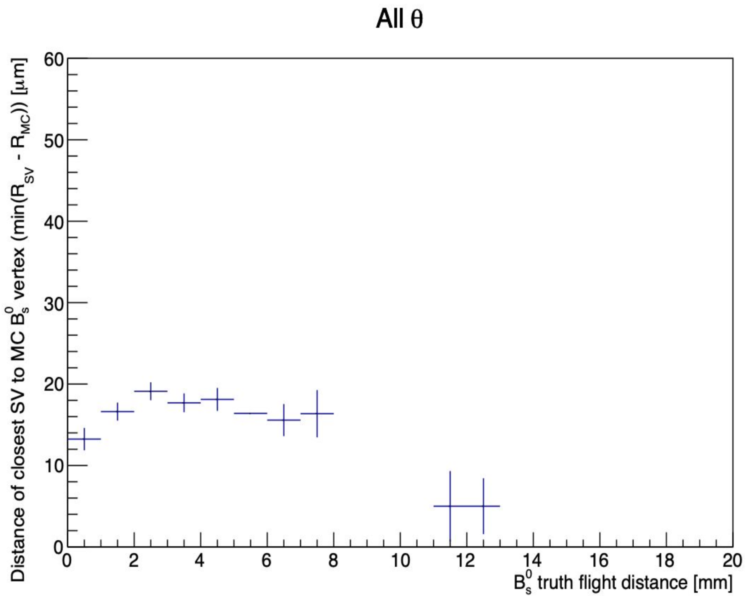


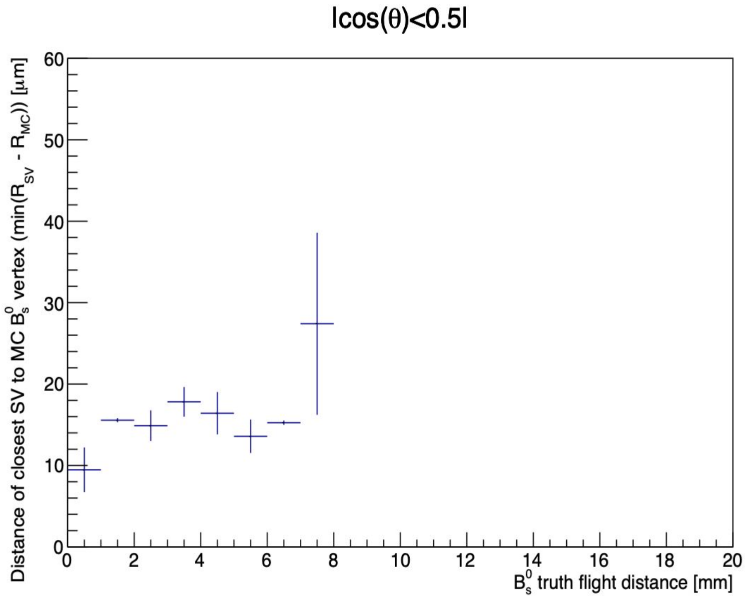


NB: Flight distance: events are generated with PV = (0,0,0), no attempt there o reconstruct the PV (a resolution of 3 μm is expected with the BS constraint, i.e. is negligible in the resolution of the flight distance).

E.Perez, C. Helsens

SV Finder Performance for different B_s flight distances





SV Finder Performance with R=1.0 cm beampipe