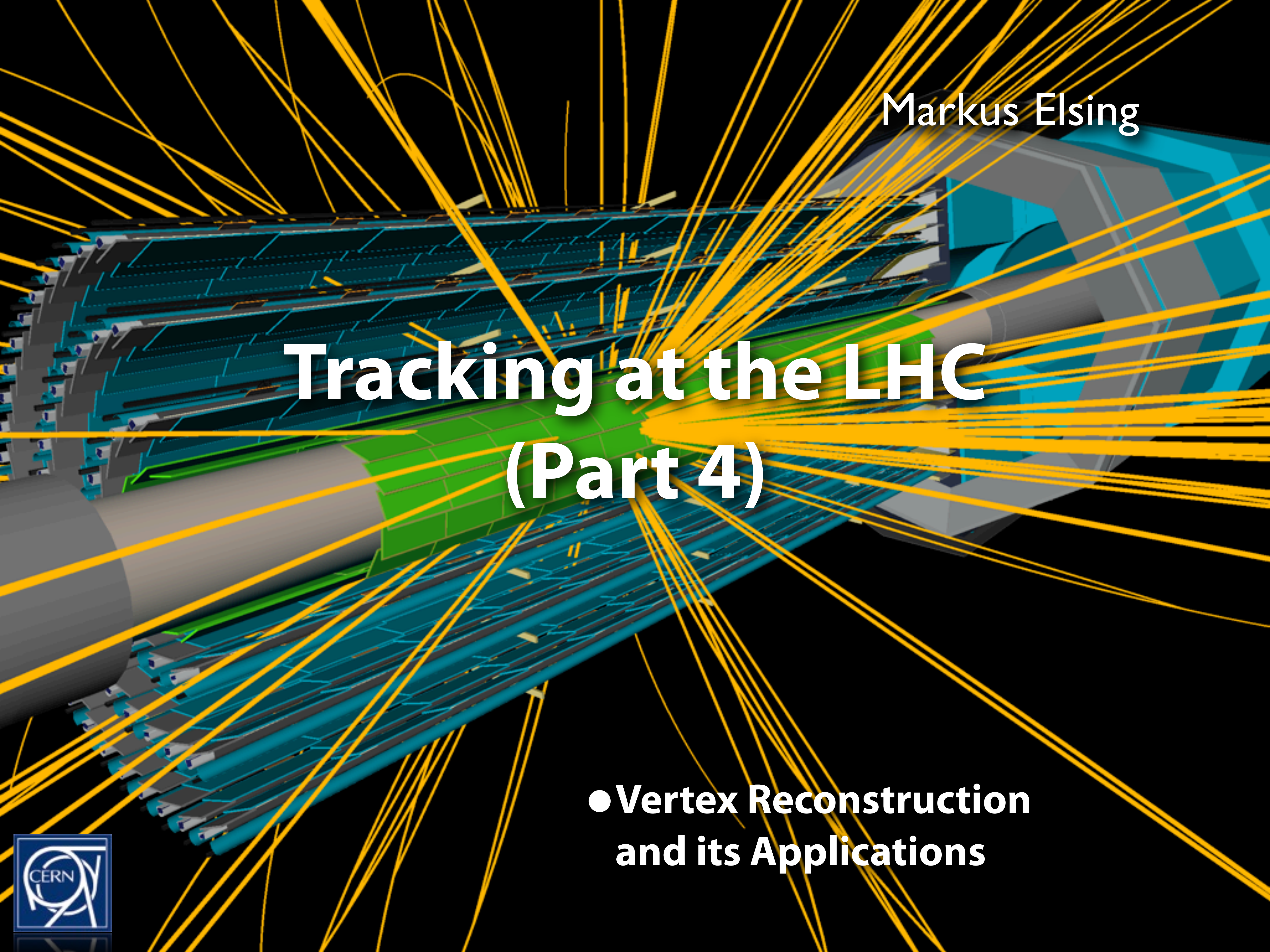


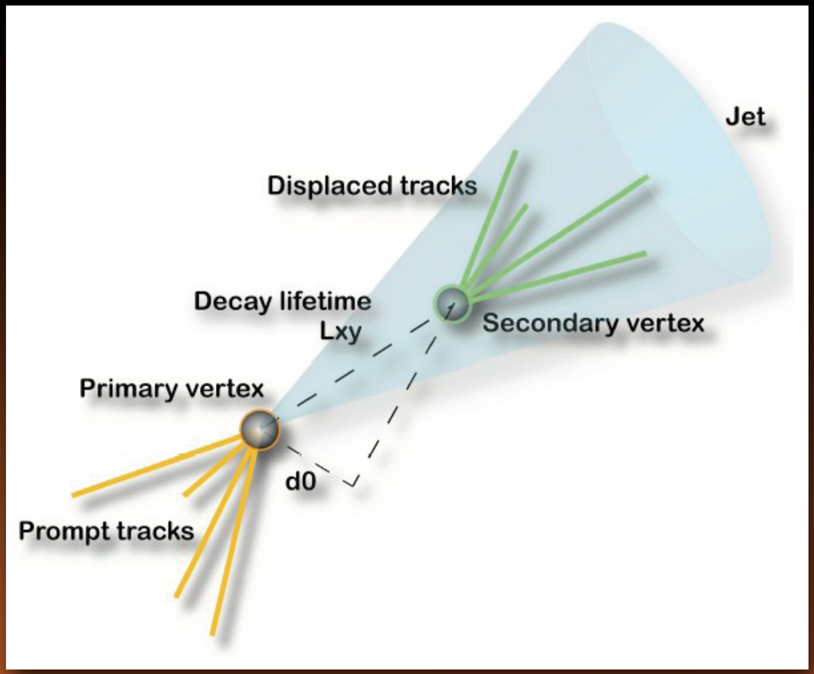
Markus Elsing

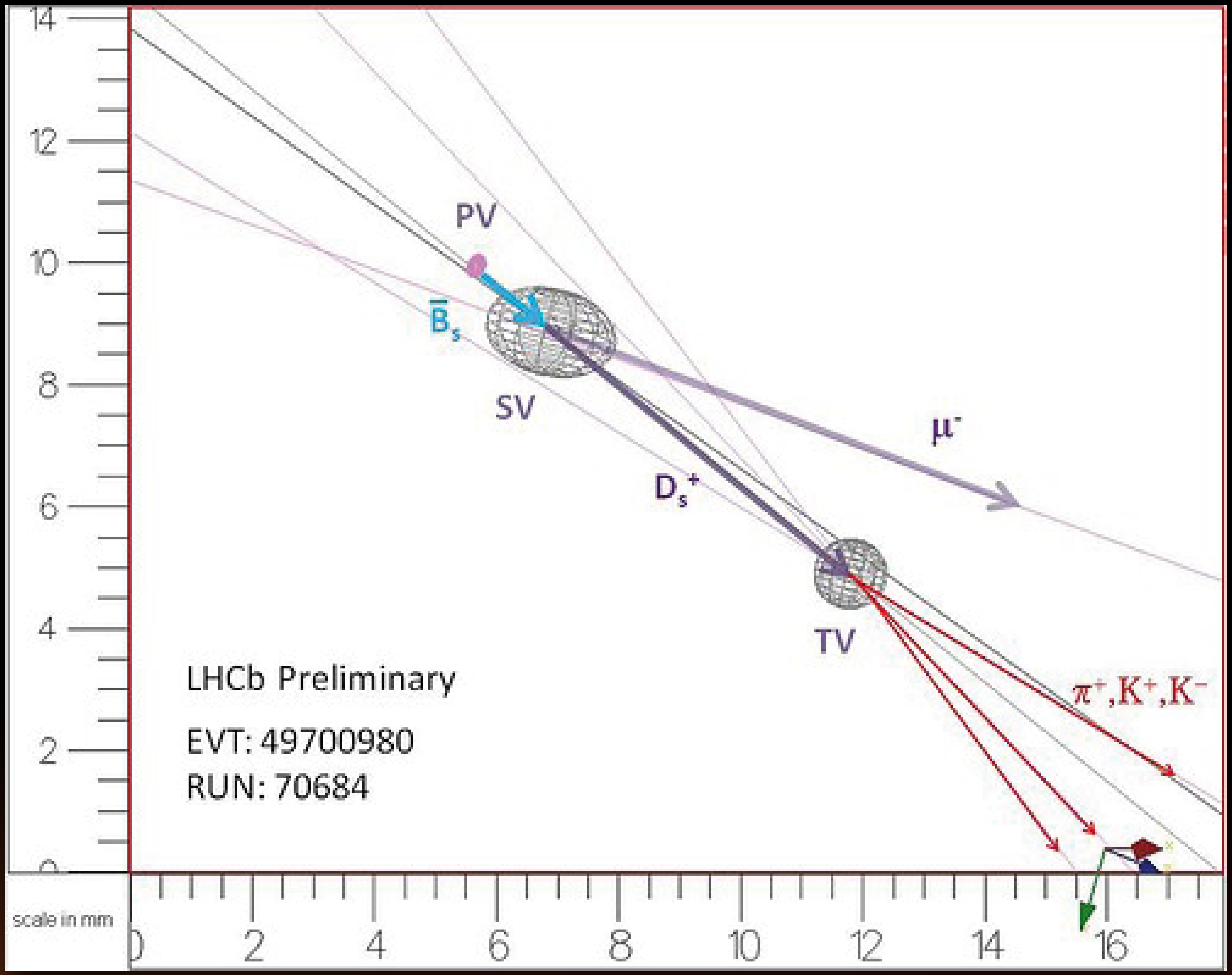
Tracking at the LHC (Part 4)

- **Vertex Reconstruction and its Applications**

Introduction: Vertexing

- b- and c-hadron lifetime
  - ➡ ≈1-1.5psec (B) and ≈0.4-1psec (D)
  - ➡ tracks have significant impact parameter, **d0** and **z0**
  - ➡ might form a reconstructed secondary vertex

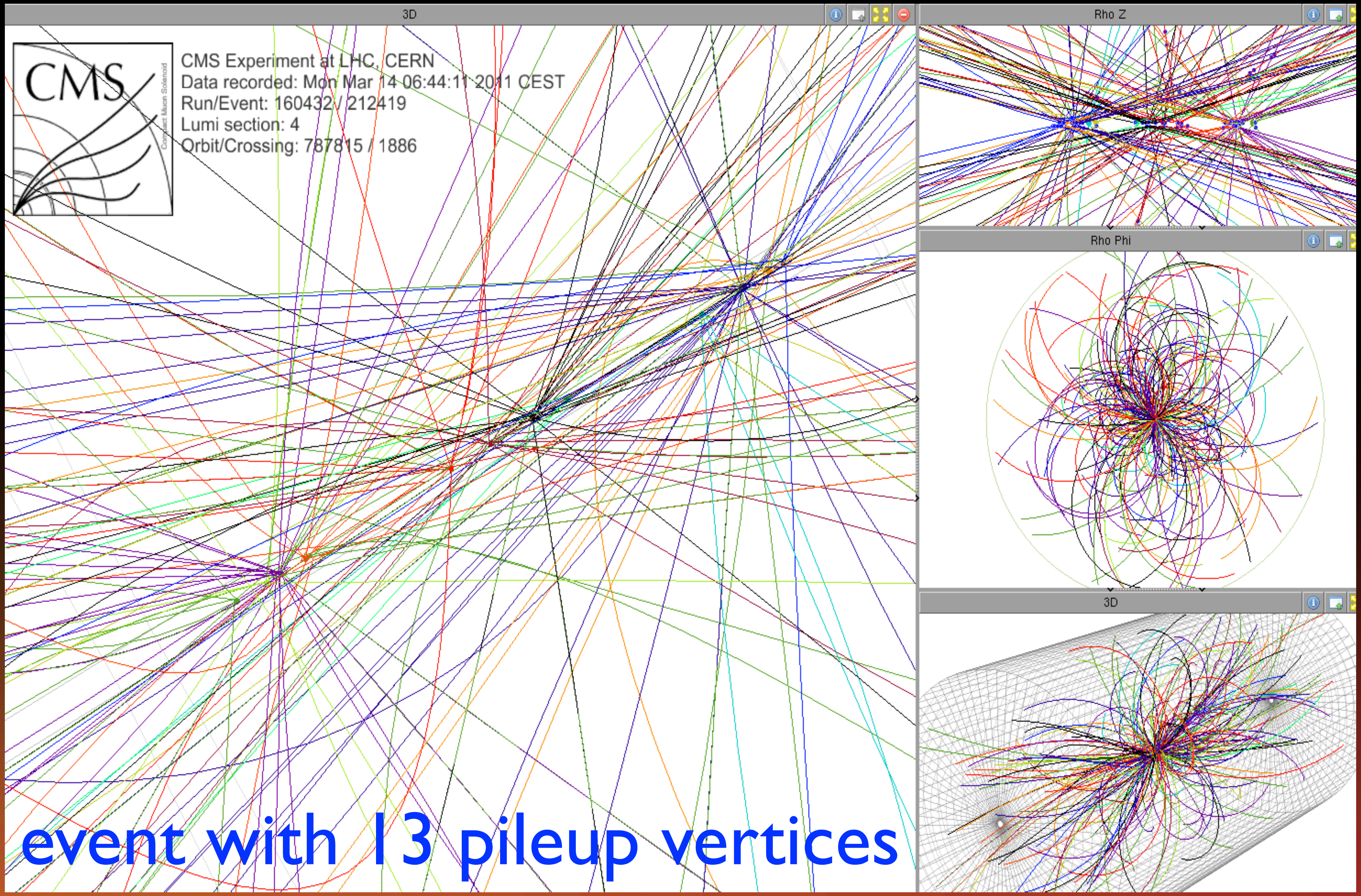




- Example:
  - ➡ a fully reconstructed $B_s \rightarrow D_s \mu\nu \rightarrow KK\pi\mu\nu$ event from LHCb
  - ➡ primary, secondary and tertiary vertex

Event Pileup

- not to forget minimum bias event pileup
  - ➡ nuisance that needs to be managed
  - ➡ affects not only tracking, but as well jet+MET reconstruction, b-tagging, ...



Outline of Part 4

- discuss vertex fitting technique
  - ➡ Least Square and Kalman Filter vertex fitter
  - ➡ adaptive vertex fitting, vertex finding, ZVTOP
- examples for vertexing applications
  - ➡ primary vertexing
  - ➡ Jet-Vertex-Fraction
  - ➡ b-tagging techniques

Vertex Fitting

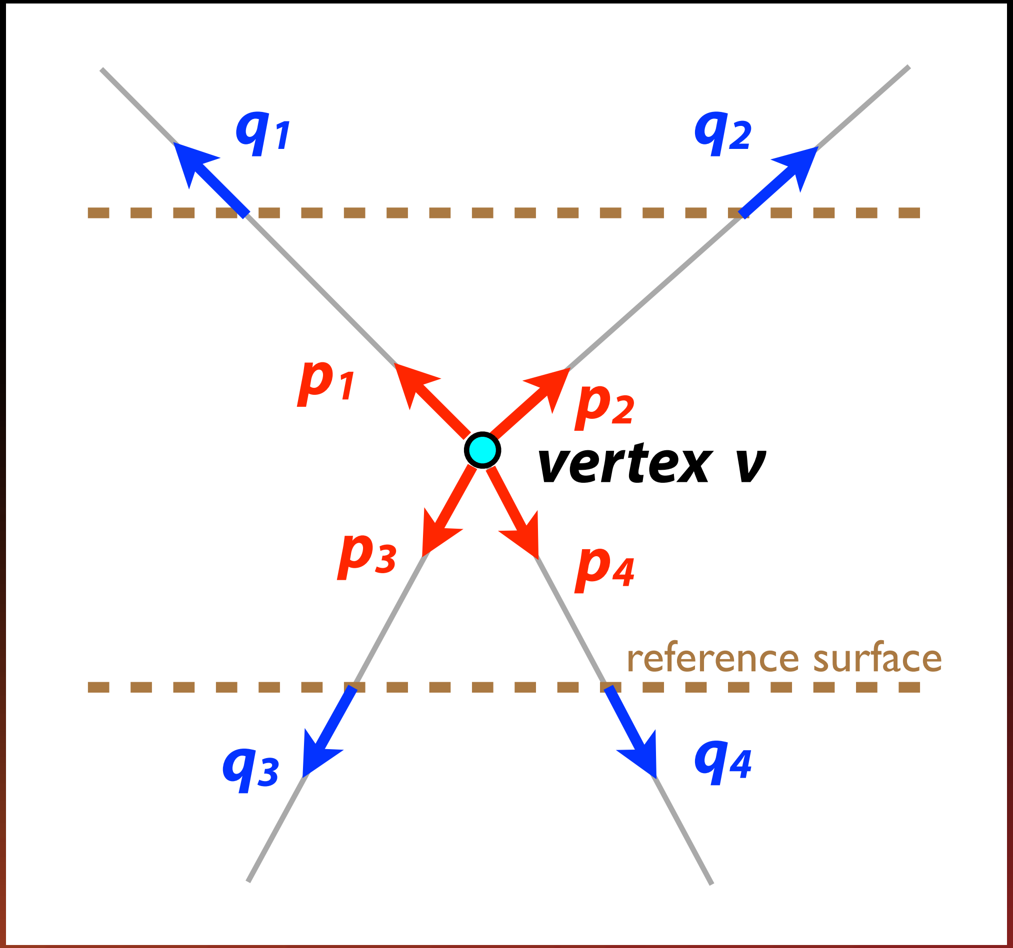
- task of a vertex fit:
  - ➡ estimate the vertex position ***v*** (and the parameters p_k at the vertex) from a set of measured track parameters q_k
- measurement model (similar to track fit)
  - ➡ in mathematical terms:

$$q_i = h_i(v, p_i) + \varepsilon_i$$

with: h_i ~ dependency of track parameters on vertex and parameters at vertex

ε_i ~ error of q_i (noise term)

Jacobians: $A_i = \dfrac{\partial h_i(v, p_i)}{\partial v} \qquad B_i = \dfrac{\partial h_i(v, p_i)}{\partial p_i}$



  - ➡ in practice: h_i is derived from trajectory model and propagator f:

$$h_i = f \circ \tilde{q}(v, p_i) \quad \text{with:} \quad \begin{aligned} v &= (v_x, v_y, v_z) \\ p_i &= (\theta_i, \phi_i, Q_i/P_i) \end{aligned}$$

  - ➡ commonly used is perigee representation for h_i

Least Square Vertex Fit

➡ let's look at the math again....

$$\chi^2 = \sum_i \Delta q_i^T G_i \Delta q_i \quad \text{with:} \quad \Delta q_i = q_i - h_i(v, p_i)$$

$$V_i = G_i^{-1} \quad \text{covariance of the measured } q_i$$

linearize the problem: $v \rightarrow v_0 + \delta v$ and $p_i \rightarrow p_{i,0} + \delta p_i$

$$h_i(v, p_i) \cong h_i(v_0, p_{i,0}) + A_i \delta v + B_i \delta p_i \quad + \text{ higher terms}$$

this yields:

$$\chi^2 = \sum_i \left(h_i(v_0, p_{i,0}) + A_i \delta v + B_i \delta p_i\right)^T G_i \left(h_i(v_0, p_{i,0}) + A_i \delta v + B_i \delta p_i\right)$$

minimizing the linearized χ^2 gives the following set of equations:

$$\frac{\partial \chi^2}{\partial v} = 0 \quad \Rightarrow \quad \left(\sum_i A_i^T G_i A_i\right) \cdot \delta v + \sum_i A_i^T G_i B_i \cdot \delta p_i = \sum_i A_i^T G_i \cdot \Delta q_{i,0}$$

$$\frac{\partial \chi^2}{\partial p_i} = 0 \quad \Rightarrow \quad B_i^T G_i A_i \cdot \delta v + B_i^T G_i B_i \cdot \delta p_i = B_i^T G_i \cdot \Delta q_{i,0}$$

with: $\Delta q_{i,0} = q_i - h_i(v_0, p_{i,0})$

➡ system of (i+1) linear matrix equations which can be solved

Least Square Vertex Fit

➡ so let's solve the system...

$$\left(\sum_i A_i^T G_i A_i\right)\cdot\delta v + \sum_i A_i^T G_i B_i \cdot \delta p_i = \sum_i A_i^T G_i \cdot \Delta q_{i,0} \qquad (1)$$

$$B_i^T G_i A_i \cdot \delta v + B_i^T G_i B_i \cdot \delta p_i = B_i^T G_i \cdot \Delta q_{i,0} \qquad (2)$$

transform (2) to replace δp_i in equation (1), gives:

$$\delta v = C \cdot \sum_i A_i^T G_i^B \cdot \Delta q_{i,0}$$

with:

$$G_i^B = G_i - G_i B_i^T W_i B_i G_i$$

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

and $C = \left(\sum_i A_i^T G_i^B A_i\right)^{-1}$ covariance of ν

➡ usually one iterates the fit to ensure convergence

Least Square Vertex Fit

- ➡ so let's solve the system...

$$\left(\sum_i A_i^T G_i A_i\right)\cdot \delta v + \cancel{\sum_i A_i^T G_i B_i \cdot \delta p_i} = \sum_i A_i^T G_i \cdot \Delta q_{i,0} \qquad (1)$$

$$\cancel{B_i^T G_i A_i \cdot \delta v + B_i^T G_i B_i \cdot \delta p_i = B_i^T G_i \cdot \Delta q_{i,0}} \qquad (2)$$

transform (2) to replace δp_i in equation (1), gives:

$$\delta v = C \cdot \sum_i A_i^T G_i^B \cdot \Delta q_{i,0} \quad \text{with:} \quad G_i^B = G_i - \cancel{G_i B_i^T W_i B_i G_i}$$

$$\cancel{W_i = \left(B_i^T G_i B_i\right)^{-1}}$$

and $C = \left(\sum_i A_i^T G_i^B A_i\right)^{-1}$ covariance of v

- ➡ usually one iterates the fit to ensure convergence
- ➡ still have to compute the corrections to p_i
- ➡ but: can obtain a **faster vertex fit**, if we neglect the δp_i terms

Least Square Vertex Fit

➡ compute the corrections to p_i

$$\left(\sum_i A_i^T G_i A_i\right) \cdot \delta v + \sum_i A_i^T G_i B_i \cdot \delta p_i = \sum_i A_i^T G_i \cdot \Delta q_{i,0} \qquad (1)$$

$$B_i^T G_i A_i \cdot \delta v + B_i^T G_i B_i \cdot \delta p_i = B_i^T G_i \cdot \Delta q_{i,0} \qquad (2)$$

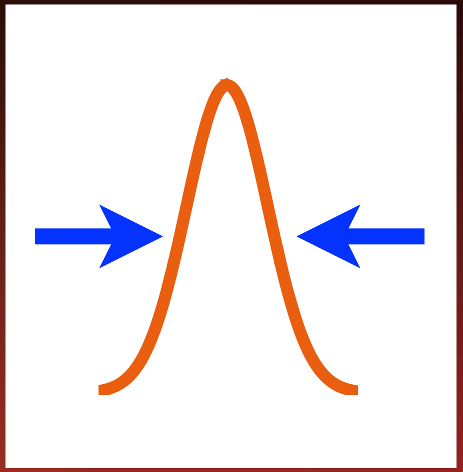
use δv in equation (2) to compute δp_i, gives:

$$\delta p_i = W_i B_i^T G_i \cdot \left(\Delta q_{i,0} - A_i \delta v\right)$$

and $D_i = W_i + W_i B_i^T G_i A_i C A_i^T G_i B_i W_i$ covariance of δp_i

➡ vertex fit is used to improve momentum measurement at vertex

➡ used to improve invariant mass resolution for reconstructed decays



Kalman Filter Notation

➡ the least square vertex fit can as well be written as a progressive fit
➡ results in an extended Kalman Filter vertex fit

1.Let's assume δv_{i-1} has been estimated using i-1 tracks. Track i is added using the update equations:

$$\delta v_i = C_i^{-1} \cdot \left[C_{i-1} \delta v_{i-1} + A_i^T G_i^B \cdot \Delta q_{i,i-1} \right]$$

and the covariance of δv_i is:

$$C_i = \left(C_{i-1}^{-1} + A_i^T G_i^B A_i \right)^{-1}$$

2.update to parameters is:

$$\delta p_{i,i} = W_i B_i^T G_i \cdot \left(\Delta q_{i,i-1} - A_i \delta v_i \right)$$

and the covariance of $\delta p_{i,i}$:

$$D_i = W_i + W_i B_i^T G_i A_i C_i A_i^T G_i B_i W_i$$

Billoir, Fruhwirth, Catlin et al.

weight matrix notation

➡ the smoother in this case is equivalent to computing the parameters $q_{i,n}$ from the final vertex estimate δv_n and $\delta p_{i,n}$

$$q_{i,n} = h_i(v_0 + \delta v_n, p_{i,0} + \delta p_{i,n})$$

with: $\mathrm{cov}(q_{i,n}) = B_i W_i B_i^T + V_i^B G_i A_i C_n A_i^T G_i V_i^B$ and $V_i^B = V_i - B_i W_i B_i^T$

Beam Spot Constraint Fit

- ➡ beam spot ***b*** and its covariance matrix ***E_b^{-1}*** determined externally
- ➡ use information in fit as external constraint
  - straight forward in Kalman Filter vertex fit, its the starting vertex

$$\delta v_0 = b \quad \text{and} \quad C_0 = E_b^{-1}$$

  - in a Least Square vertex fit an additional term is added to the χ^2

$$\chi^2 = \sum_i \Delta q_i^T G_i \Delta q_i + (b - v)^T E_b (b - v)$$

minimizing the linearize χ^2 leads to the modified set of equations:

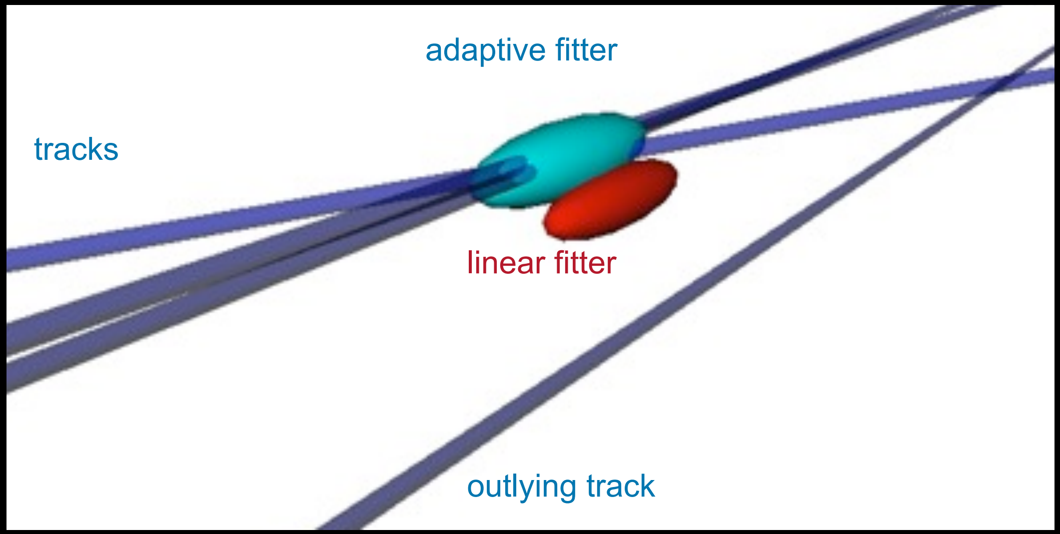
$$\left(E_b + \sum_i A_i^T G_i A_i \right) \cdot \delta v + \sum_i A_i^T G_i B_i \cdot \delta p_i = E_b (b - v_0) + \sum_i A_i^T G_i \cdot \Delta q_{i,0} \qquad (1')$$

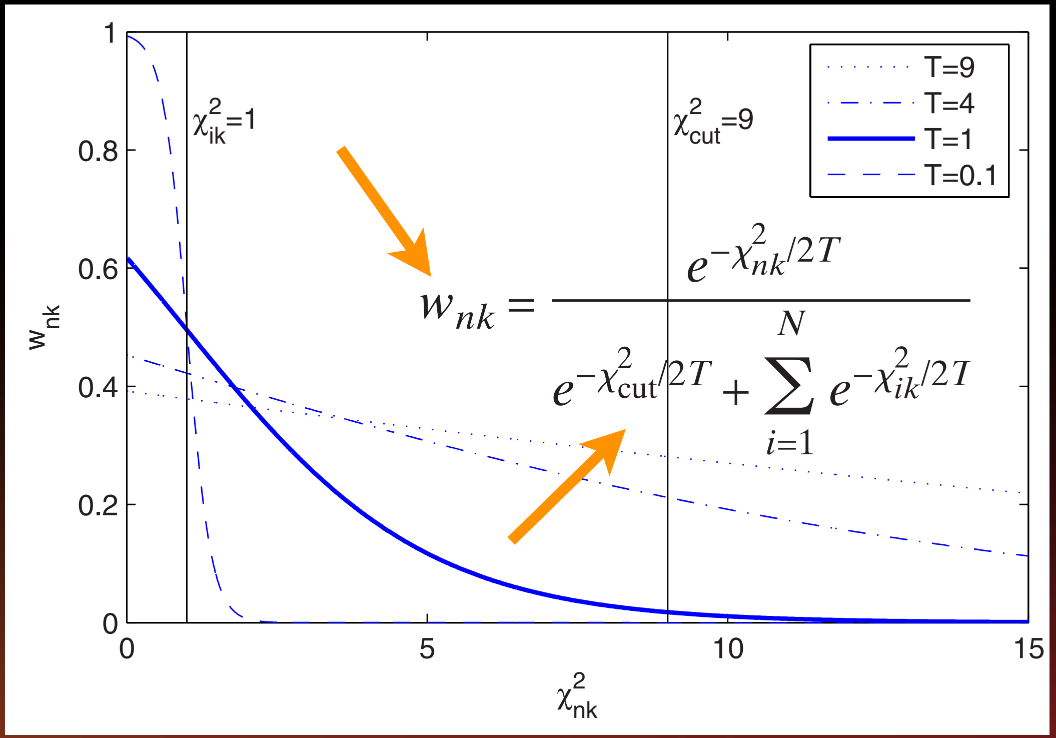
$$B_i^T G_i A_i \cdot \delta v + B_i^T G_i B_i \cdot \delta p_i = B_i^T G_i \cdot \Delta q_{i,0} \qquad (2)$$

which can be solved as before...

Adaptive Vertex Fitter

- adaptive technique
  - ➡ concept used for adaptive track fitting
  - ➡ can be applied as well on vertex fitting
- algorithm is called Adaptive Vertex Fitter
  - ➡ ATLAS and CMS vertexing packages
  - ➡ implemented as iterative, reweighted Kalman filter
    - ***w_{nk}*** is weight of track ***k*** w.r.t. vertex ***n***
    - outlying tracks are down-weighted automatically
  - ➡ robust fitter !
- extension for Multi-Vertex-Fitter
  - ➡ vertices compete for tracks





Inspecting Outliers

- common problem:
  - ➡ fit quality is bad, want to identify the χ^2 contribution of each track to overall fit (and to track with largest contribution)
  - ➡ compare χ^2 of fit to all tracks with the χ^2 of fit with 1 track less:

$$\Delta\chi_i^2 = \underbrace{\boxed{\sum_i \Delta q_i^T \cdot G_i \cdot \Delta q_i}}_{\text{track } \chi^2} + \underbrace{\boxed{\left(\Delta q_i - A_i \delta v\right)^T \cdot G_i^B A_i C^{-1} A_i^T G_i^B \cdot \left(\Delta q_i - A_i \delta v\right)}}_{\text{change to } \chi^2 \text{ from including it in } \delta v}$$

- application: **iterative vertex finder**
  - ➡ fit all tracks into 1 vertex
  - ➡ remove worst track one by one, until fit χ^2 is acceptable
  - ➡ take removed tracks and try to find next vertex
  - ➡ repeat until no further vertex with at least 2 tracks can be found

Other Vertex Finding Strategies

- vertex z-scan
  - ➡ used e.g. in primary vertex finding
  - ➡ histogram technique
  - ➡ search for peaks in z_0 of tracks extrapolated to beam line
  - ➡ seed vertex fitter with matching tracks

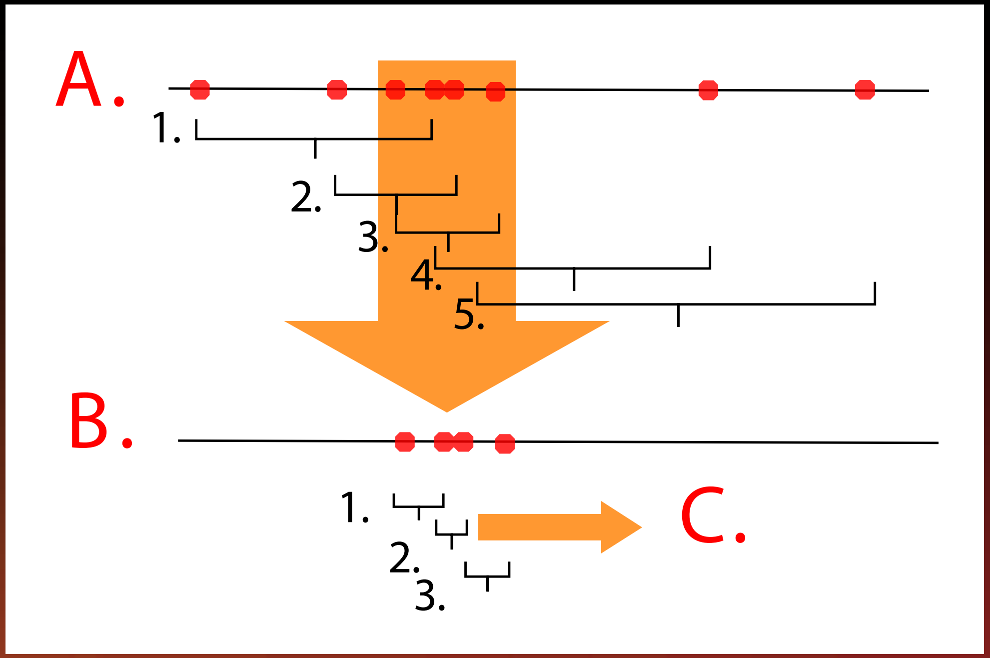
- half sample mode algorithm
  - ➡ find points of closest approach between all track pairs
  - ➡ in each of the 3 projections:

A. try all the intervals which cover 50 % of the points and take the smallest one (in this case number 3)

B. continue iterating until you have ≤ 3 points left

C. take the mean of the 2 or 3 remaining points

- ➡ defines vertex seed, find matching tracks...



Topological Vertex Finder (ZVTOP)

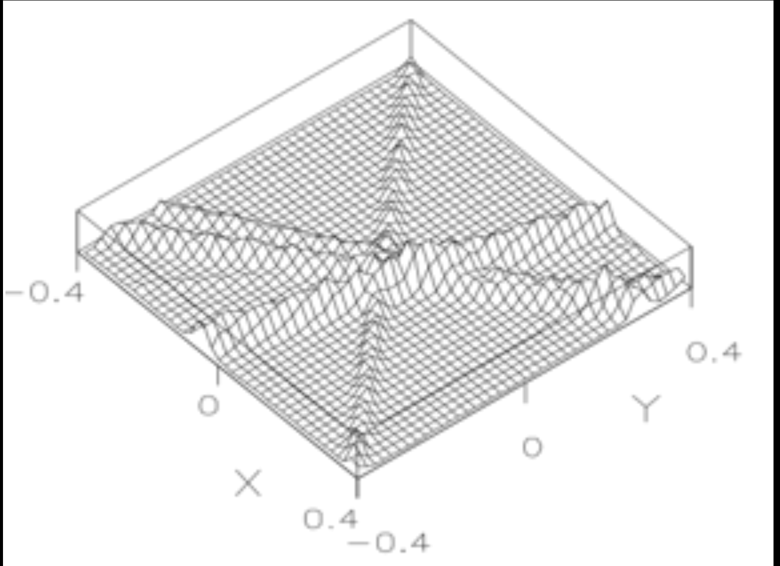
- example for an inclusive vertex finder
  - ➡ very powerful, developed by SLD
- 3 dimensional vertex search
  - ➡ construct for each track Gaussian probability tube ***$f_i(v)$***

$$f_i(\boldsymbol{v}) = \exp\left[-\frac{1}{2}(\boldsymbol{v}-\boldsymbol{r})^{\mathrm{T}}\boldsymbol{V}_i^{-1}(\boldsymbol{v}-\boldsymbol{r})\right]$$

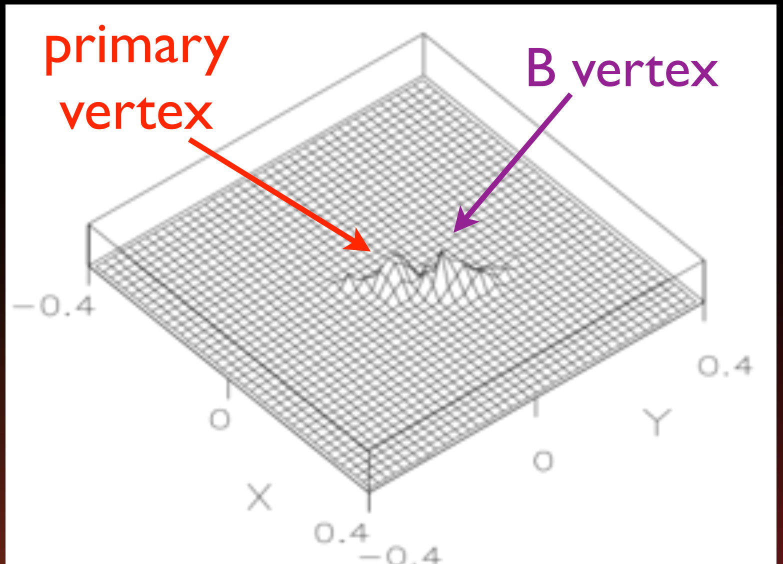
  - ***r*** is point of closest approach of track ***i*** to point ***v***
  - ➡ find all points where ***$f_i(v)$*** is significant for 2 tracks
  - ➡ define vertex probability function ***V(v)*** around those points

$$V(\mathbf{r}) = \sum_{i=0}^{N} f_i(\mathbf{r}) - \frac{\sum_{i=0}^{N} f_i^2(\mathbf{r})}{\sum_{i=0}^{N} f_i(\mathbf{r})}$$

  - search for maxima, merge nearby vertex candidates
- SLD used ZVTOP as well to construct an inclusive b-jet tagger



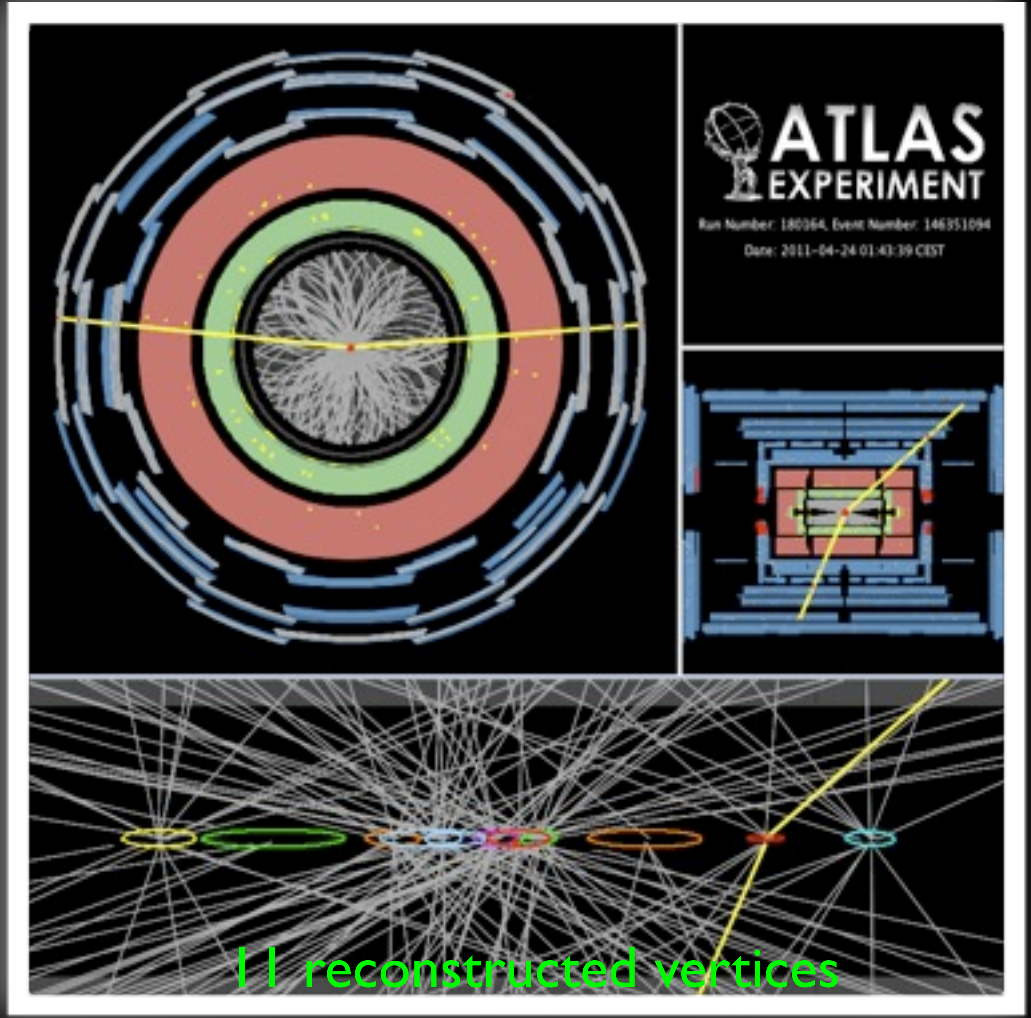
track Gaussian tube function $f_i(v)$

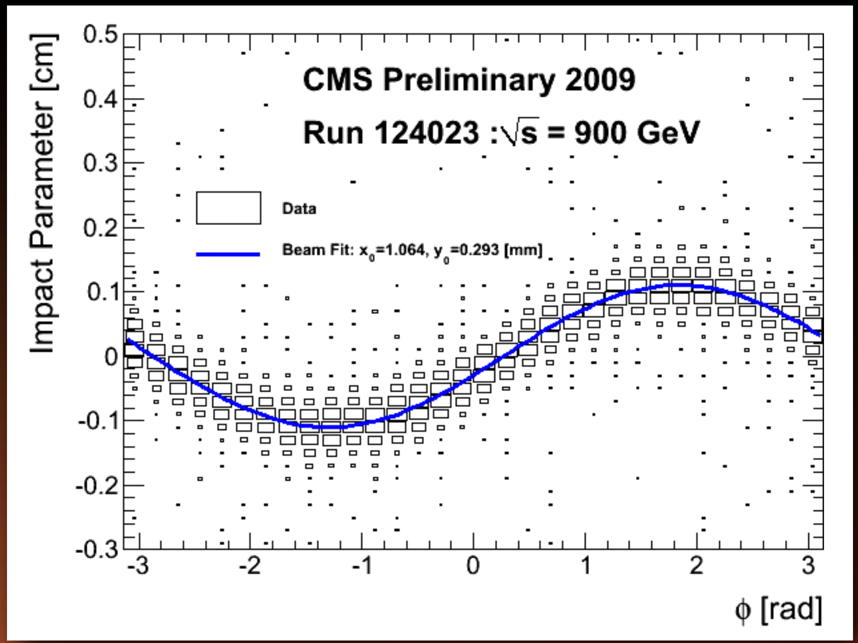


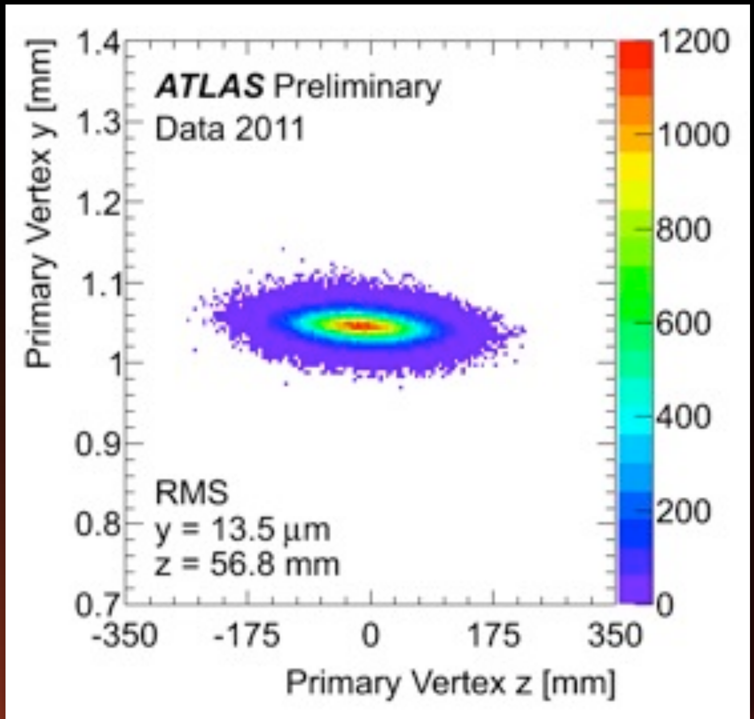
vertex probability function $V(v)$

Vertexing Applications

- primary vertex finding
  - ➡ reconstruct primary and pileup vertices
  - ➡ ATLAS (and CMS) use an iterative vertex finder and an adaptive fitter
- beam spot routinely determined
  - ➡ averaged over short periods of time
  - ➡ input to primary vertex reconstruction as a constraint
- many applications
  - ➡ primary vertex counting (luminosity)
  - ➡ jet energy scale correction for in time pileup
  - ➡ ...



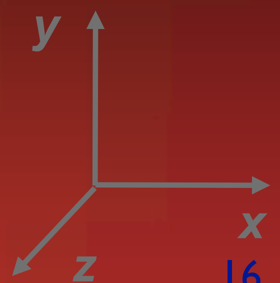




b-Jet Tagging

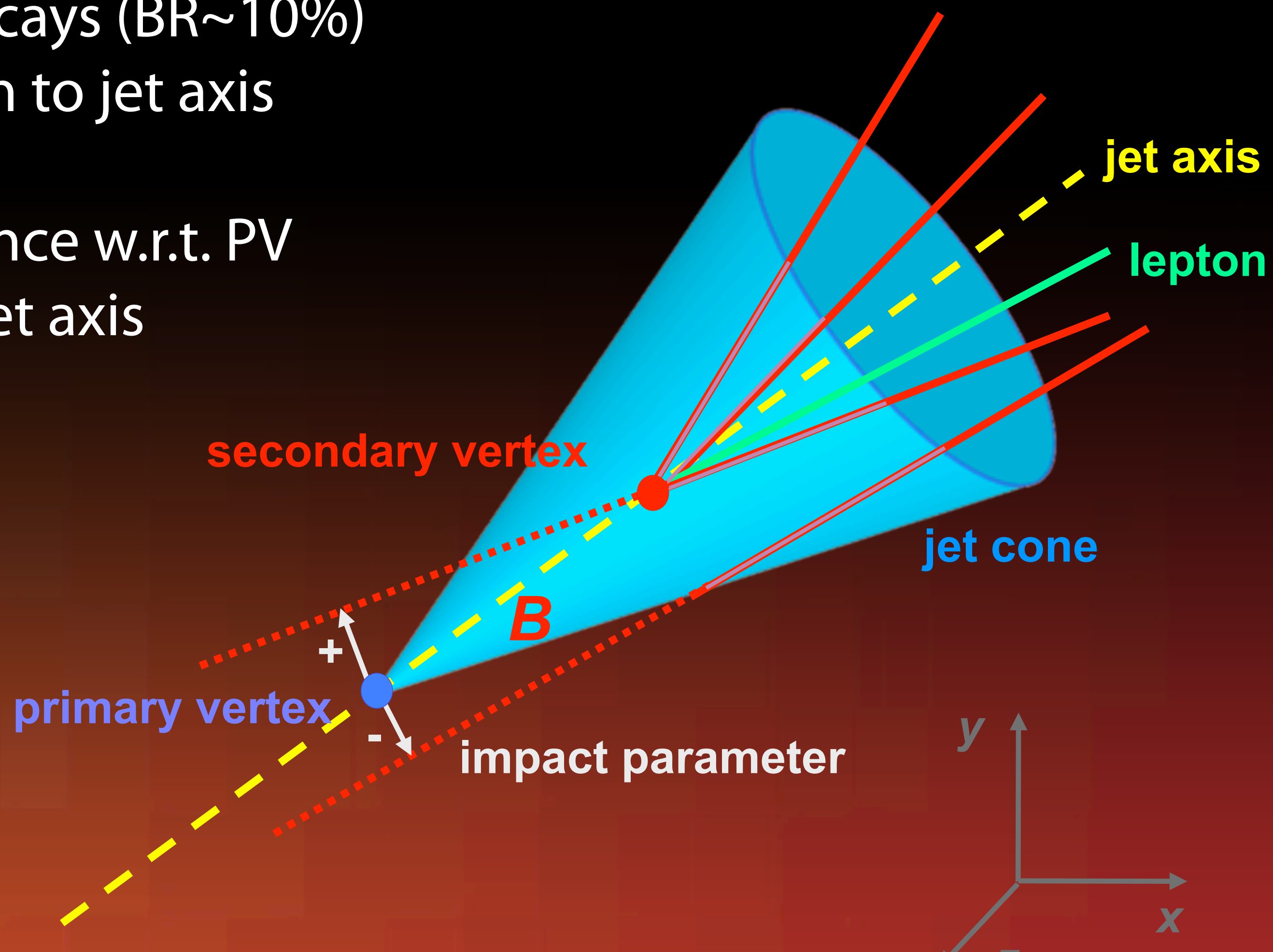
- several different techniques being explored to tag b-(and c-) jets
  - ➡ explore b-(c-) hadron fragmentation, lifetimes, mass and decay properties

- 3 categories:
  - ➡ **soft lepton** tagging
    - explore semileptonic b- and c-decays (BR~10%)
    - tagging is done using p_T of lepton to jet axis
  - ➡ **impact parameter** tagging
    - tagging is done using IP significance w.r.t. PV
    - sign impact parameter (IP) w.r.t. jet axis
    - done in Rϕ (2D) or in Rϕ+Rz (3D)
  - ➡ **secondary vertex** (SV) tagging
    - reconstruct b-(c-)decay vertex
    - use decay length significance
    - additional vertex information: mass, multiplicity, total momentum



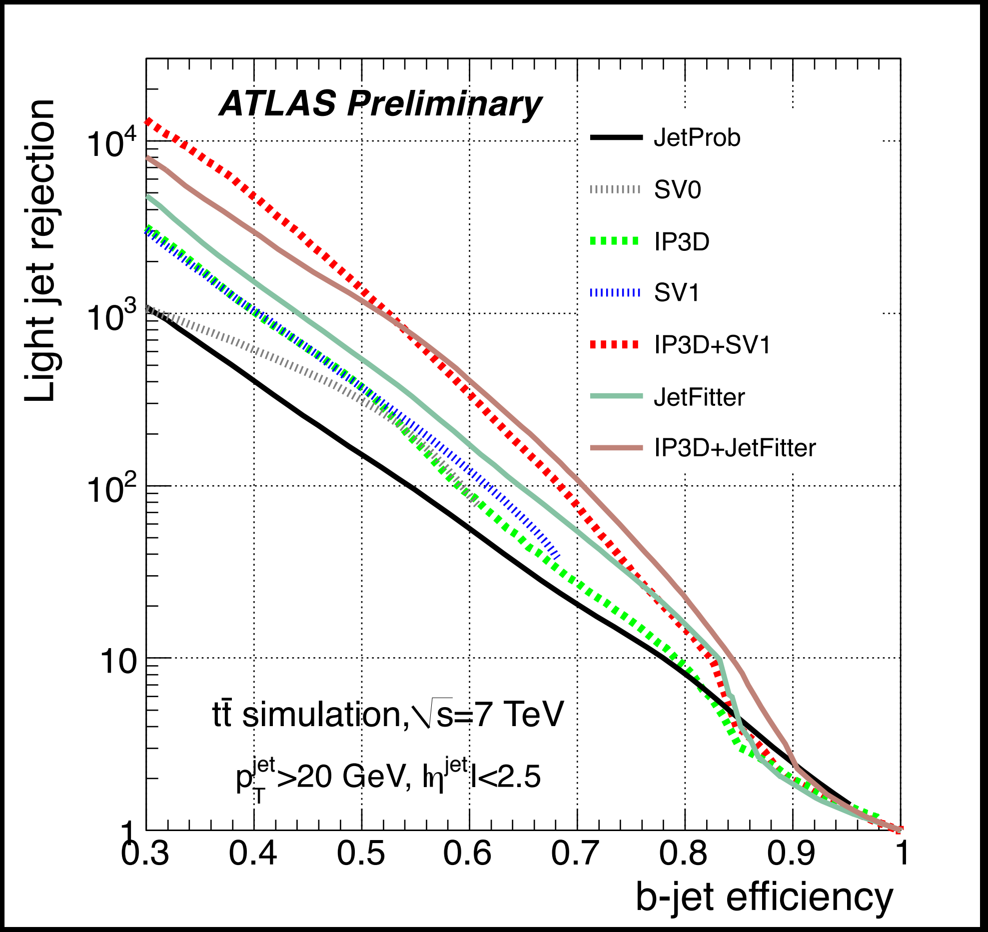
b-Jet Tagging

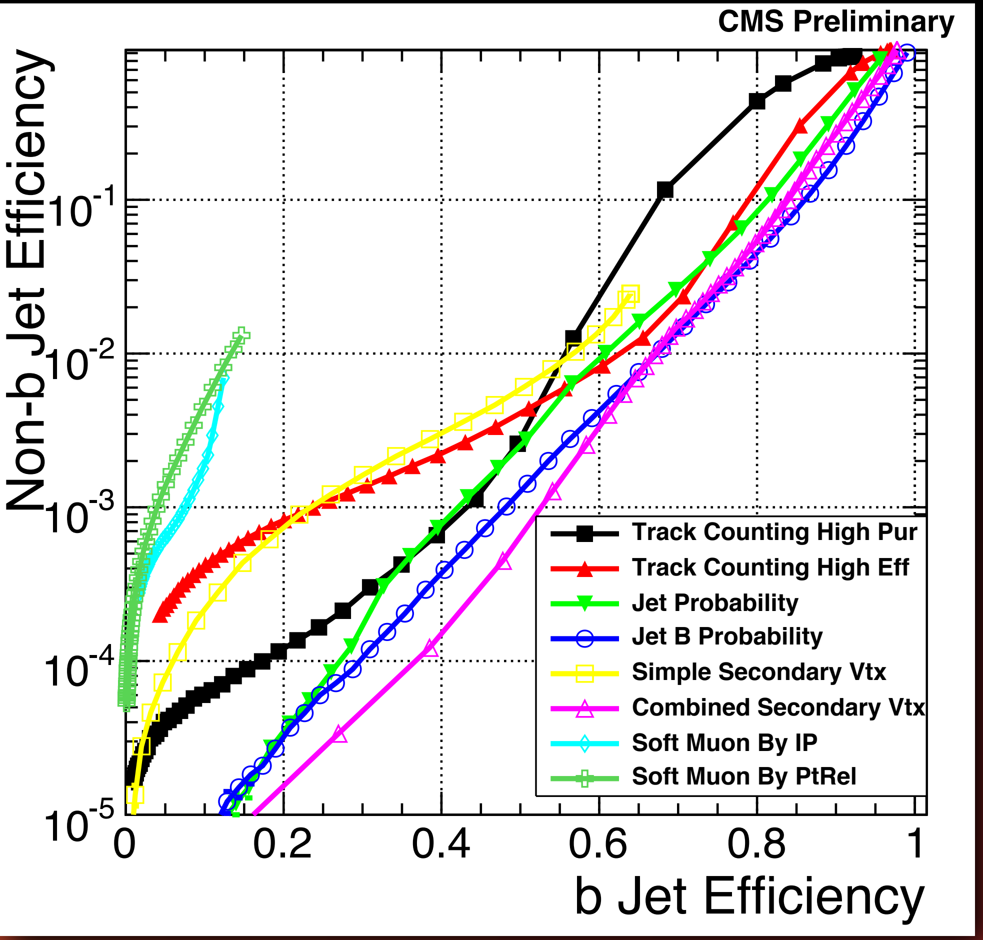
- several different techniques being explored to tag b-(and c-) jets
  - ➡ explore b-(c-) hadron fragmentation, lifetimes, mass and decay properties
- 3 categories:
  - ➡ **soft lepton** tagging
    - explore semileptonic b- and c-decays (BR~10%)
    - tagging is done using p_T of lepton to jet axis
  - ➡ **impact parameter** tagging
    - tagging is done using IP significance w.r.t. PV
    - sign impact parameter (IP) w.r.t. jet axis
    - done in Rϕ (2D) or in Rϕ+Rz (3D)
  - ➡ **secondary vertex** (SV) tagging
    - reconstruct b-(c-)decay vertex
    - use decay length significance
    - additional vertex information: mass, multiplicity, total momentum



b-Jet Tagging

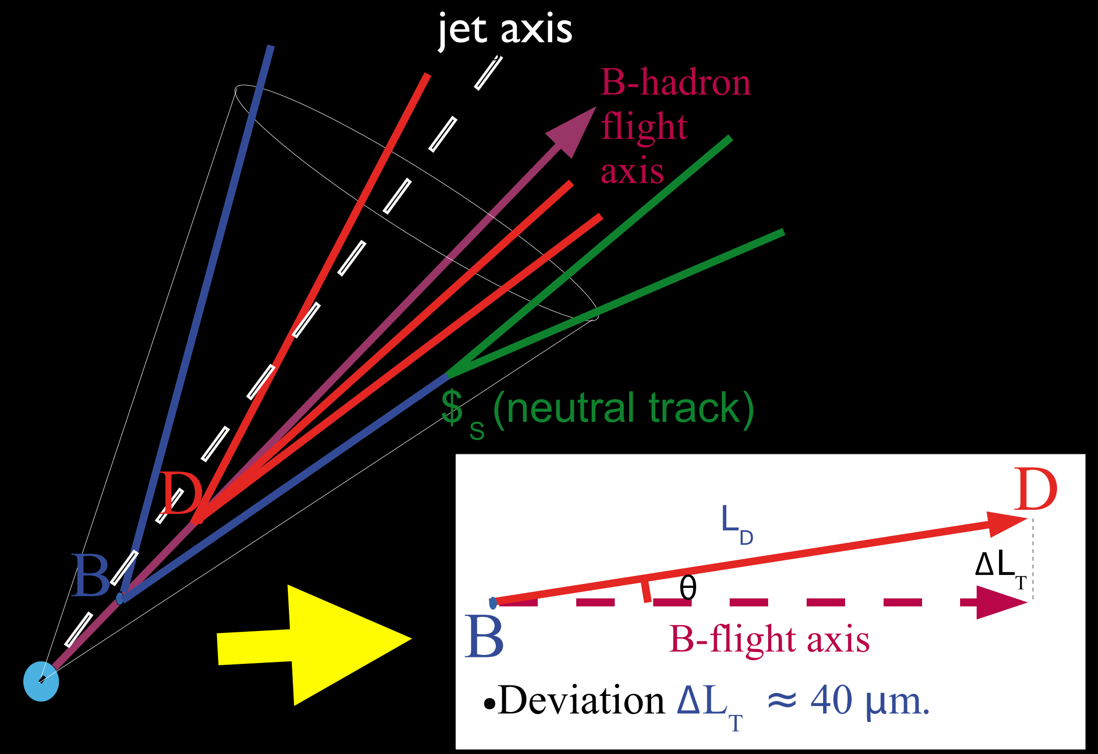
- 'simple' tagging techniques
  - ➡ **soft lepton** tagger
  - ➡ **track counting**
    - count number of tracks significant IP offsets
  - ➡ **jet probability**
    - construct probability that IP significance of all tracks in jet is compatible with PV
  - ➡ **secondary vertex** (SV) tagger
    - decay length significance
- more elaborate combined taggers
  - ➡ construct IP based likelihood using b/c/light templates (**IP2D** and **IP3D**)
  - ➡ combined likelihood taggers using IP and secondary vertex information (**IP3D+SV0**)
  - ➡ use multi-variant techniques to classify jets
- similar set of algorithms used by experiments

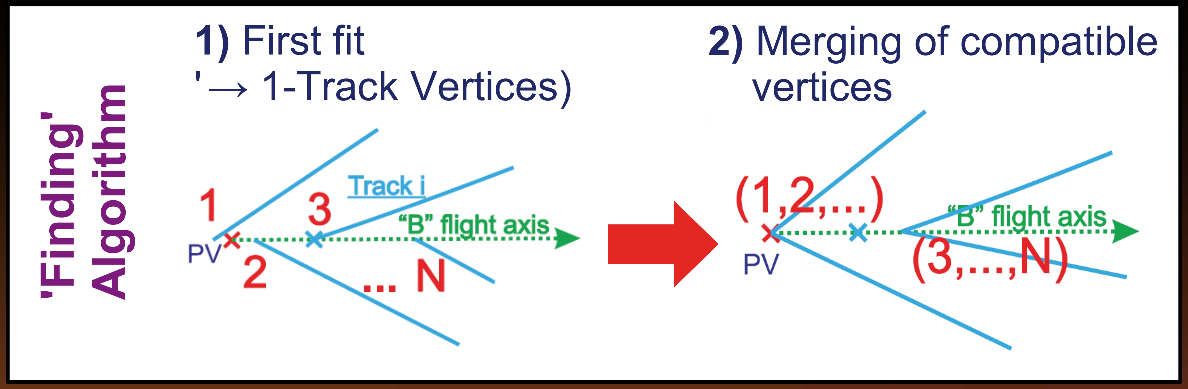


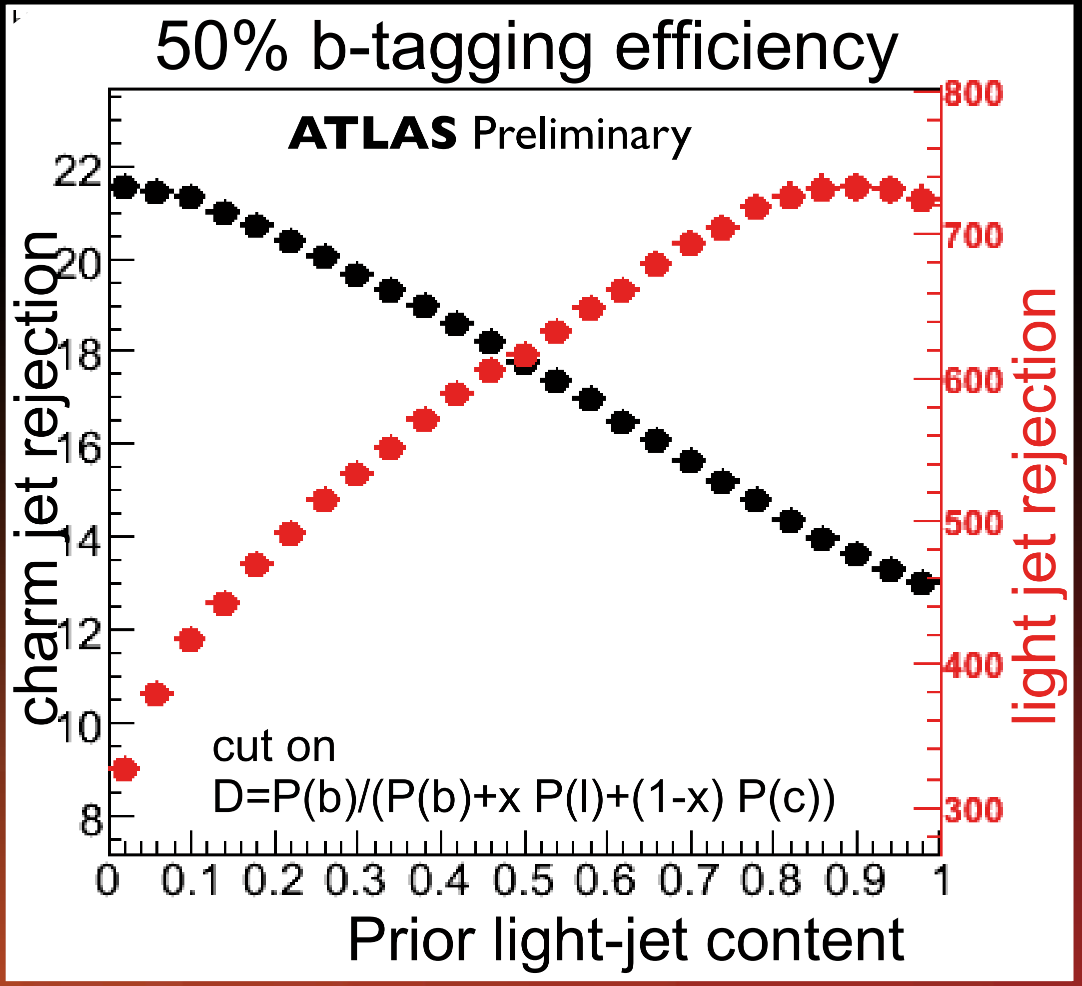


Jet-Fitter as a b-Tagger

- conventional vertex tagger
  - ➡ fits all displaced tracks into a common geometrical vertex
- Jet-Fitter
  - ➡ b-/c-hadron vertices and primary vertex approximately on the same line
  - ➡ fit of 1..N vertices along B-hadron axis
  - ➡ mathematical extension of conventional Kalman Filter vertex fitter

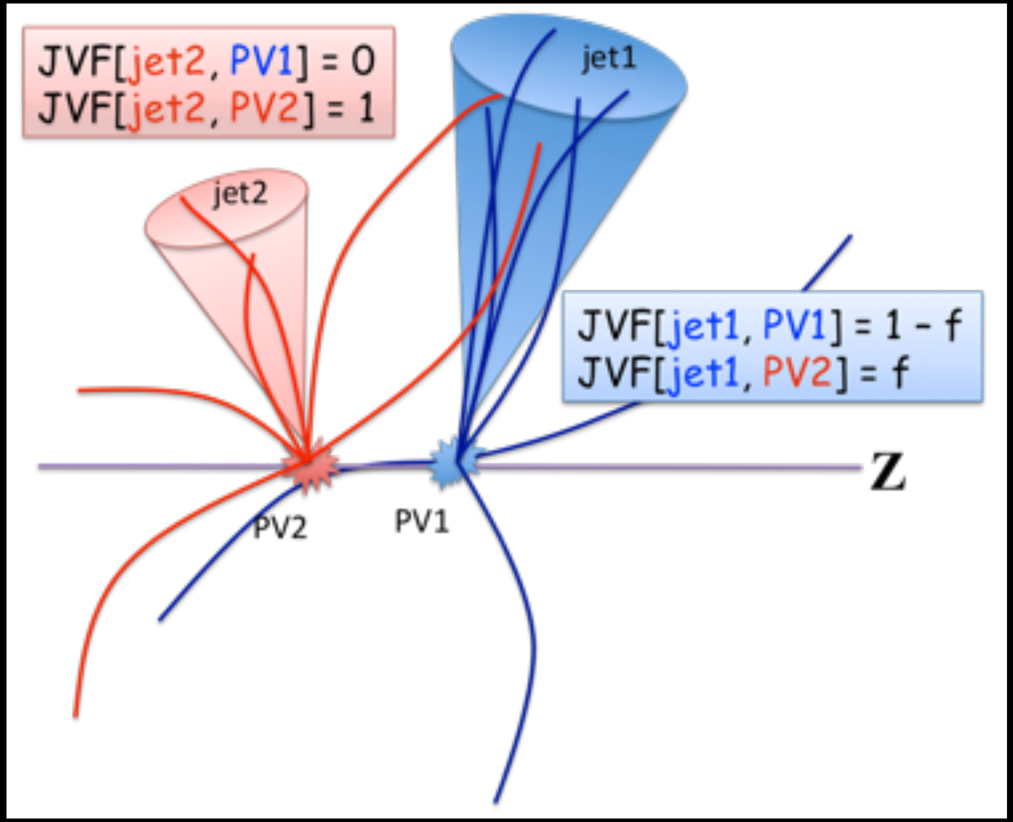






- up to 40% better light rejection
  - ➡ much improved control of charm rejection
  - ➡ best b-tagger in ATLAS

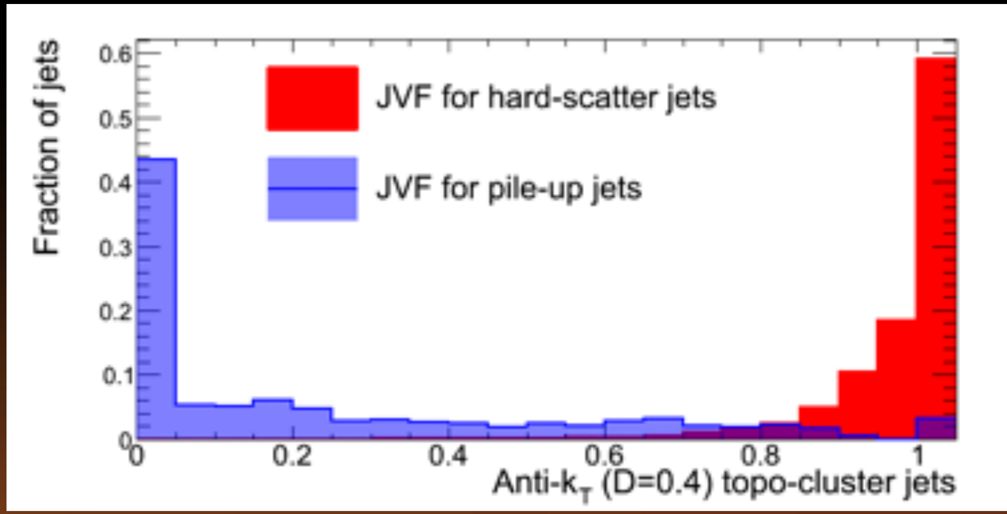
Jet-Vertex-Fraction

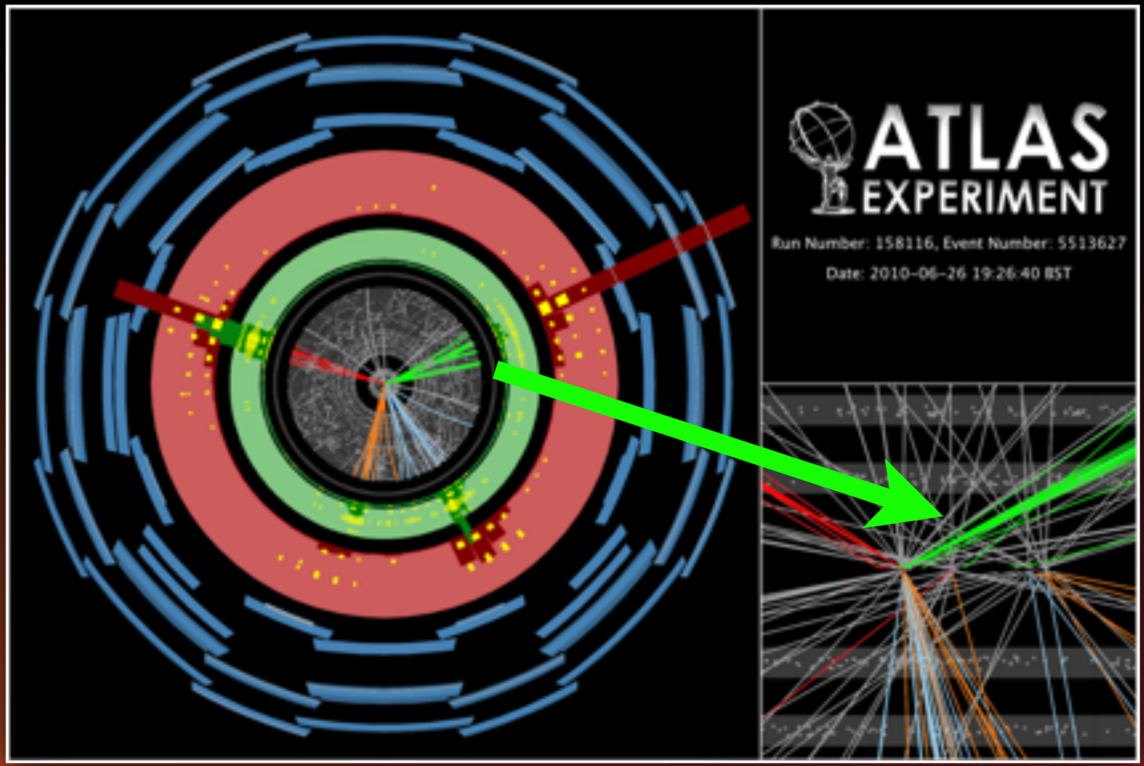


- developed at D0
  - ➡ separate jets from signal and pileup events
  - ➡ defined fraction of pT of tracks in jets associated to primary vertex:

$$JVF(\text{jet}_i, \text{vtx}_j) = \frac{\sum_k p_T(\text{trk}_k^{\text{jet}_i}, \text{vtx}_j)}{\sum_n \sum_l p_T(\text{trk}_l^{\text{jet}_i}, \text{vtx}_n)}$$

  - ➡ good separation in D0 and at LHC at low pileup





- LHC interaction region is a factor ~6 smaller than at Tevatron
  - ➡ more confusion at LHC design luminosity

Let's Summarize...

- discussed vertex fitting and finding techniques
- b-tagging and other examples for vertexing applications
- next is to discuss commissioning, alignment and performance