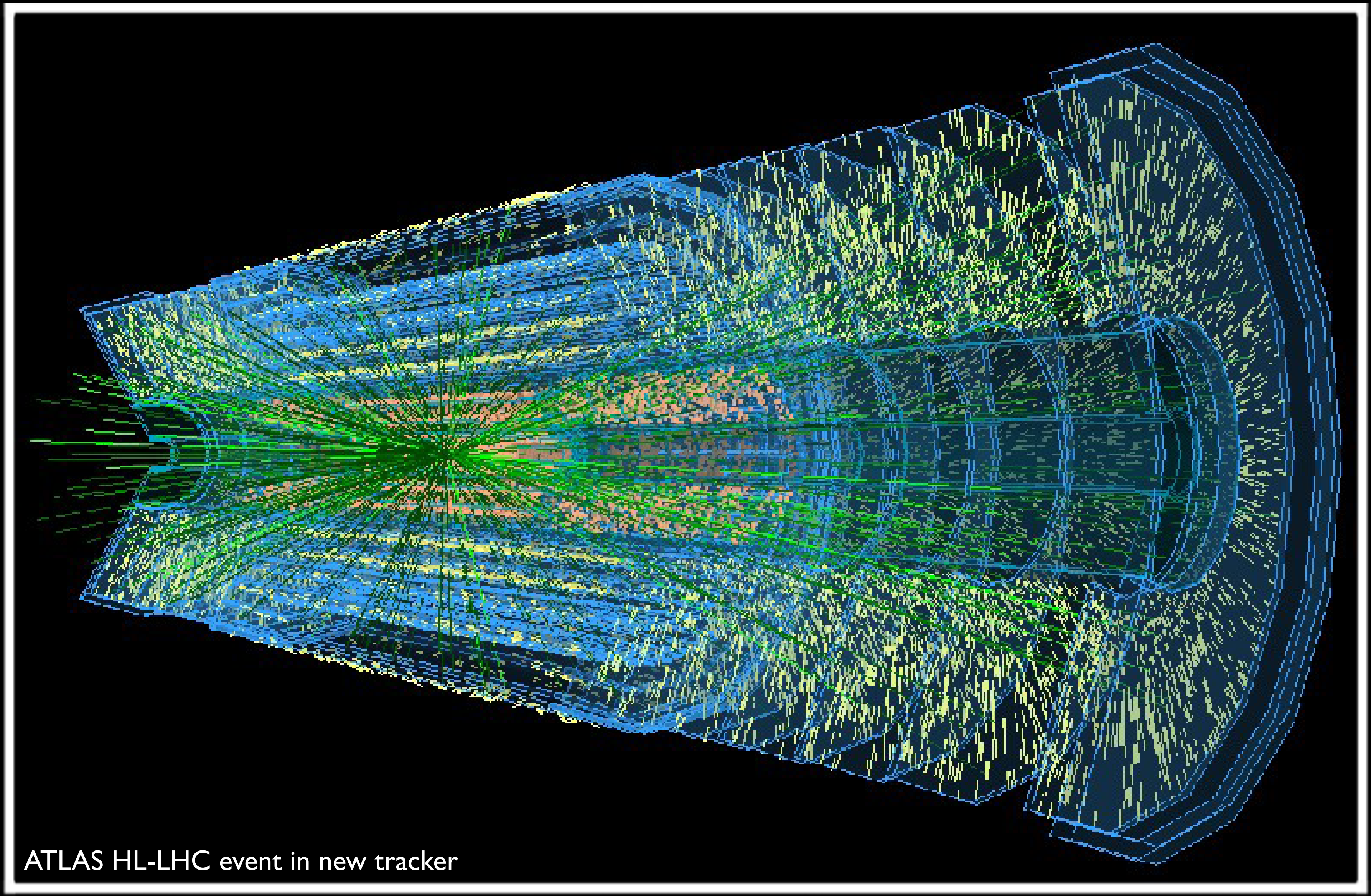
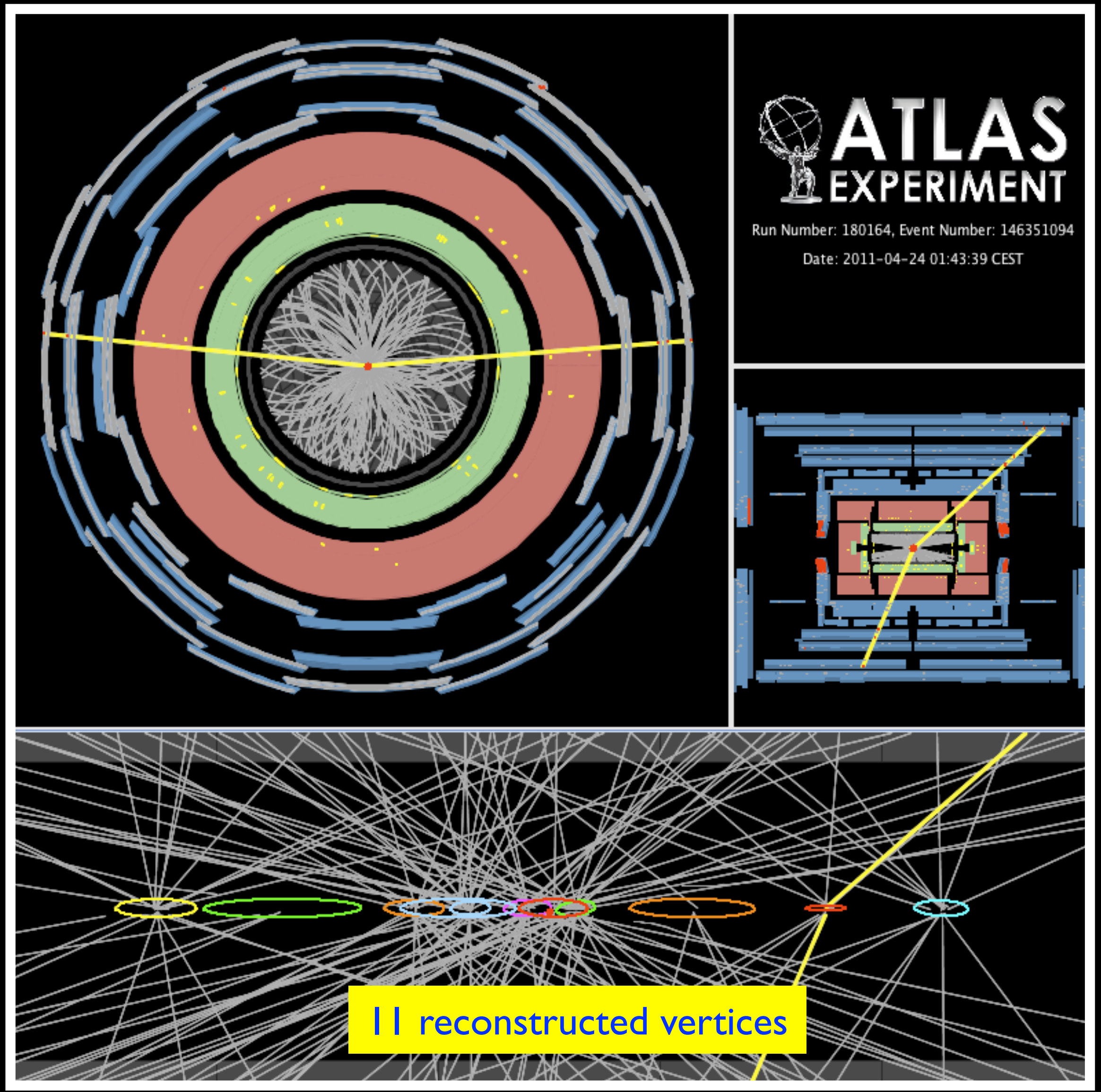
Tracking at the LHC (Part 4):
Vertex Reconstruction and its Applications

Lectures given at the University of Freiburg
Markus Elsing, 12-13.April 2016



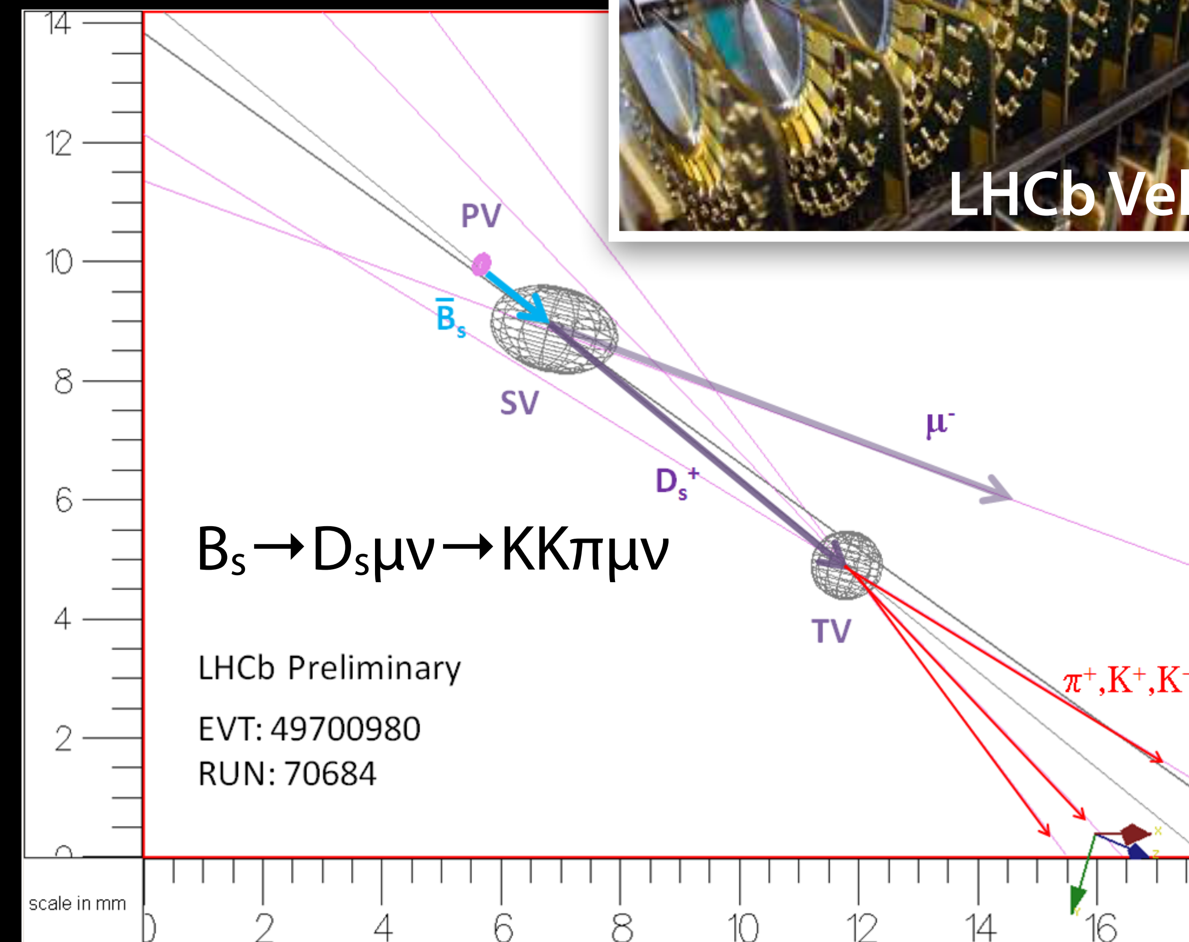
Introduction to Vertex Reconstruction

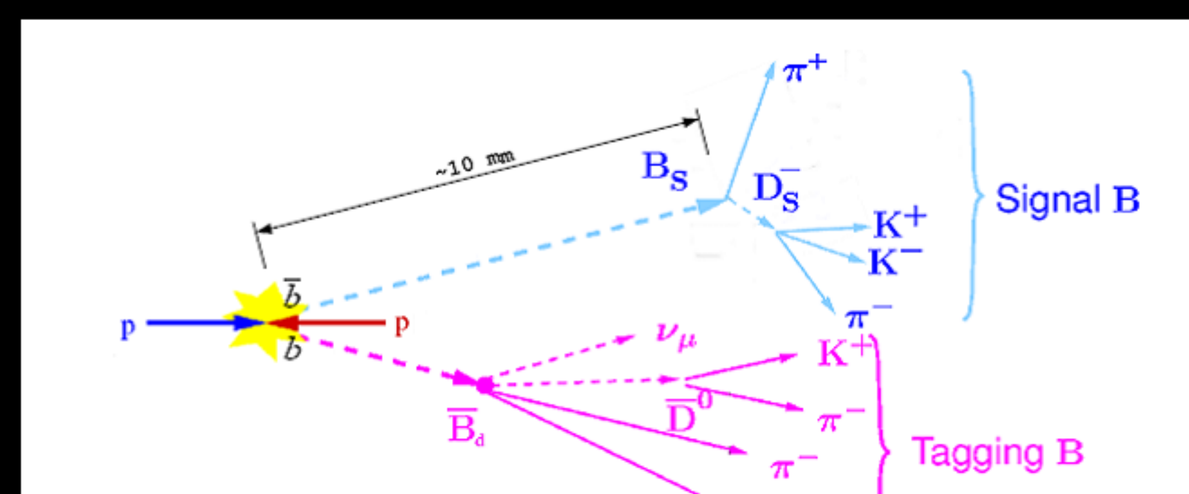
- vertex fitting techniques play an important role
  - ➡ in reconstruction chain following track reconstruction
    - primary interaction vertex reconstruction and identification
    - in time pileup estimation and pileup mitigation in particle flow reconstruction
    - secondary vertex finding for b-/c-jet identification, τ-reconstruction, photon conversions finding
  - ➡ in physics analysis
    - primary interaction vertex selections for leptons, jets, ...
    - pileup corrections to jets and missing energy
    - full reconstruction of hadronic decays like heavy flavours (B/D/...) or strange hadrons (K^0_s , Λ, ...)
    - displaced secondary vertex finding for R-hadron searches (RPV-SUSY)
    - material studies in tracker using photon conversions and hadronic showers
    - ...



Introduction to Vertex Reconstruction

- large parts of LHC physics program depends on vertex reconstruction
  - ➡ precision heavy flavour physics (LHCb)
  - ➡ b-jet tagging for SM/top/SUSY physics
  - ➡ ...
- explores b- and c-hadron lifetime
  - ➡ 1-1.5 psec (B) and 0.4-1psec (D)
  - ➡ allows to reconstruct secondary vertices
  - ➡ tracks get significant impact parameters
- silicon detectors allow for precise impact parameter reconstruction
  - ➡ stereo strips in current LHCb Velo detector
  - ➡ Pixels in ALICE, ATLAS and CMS

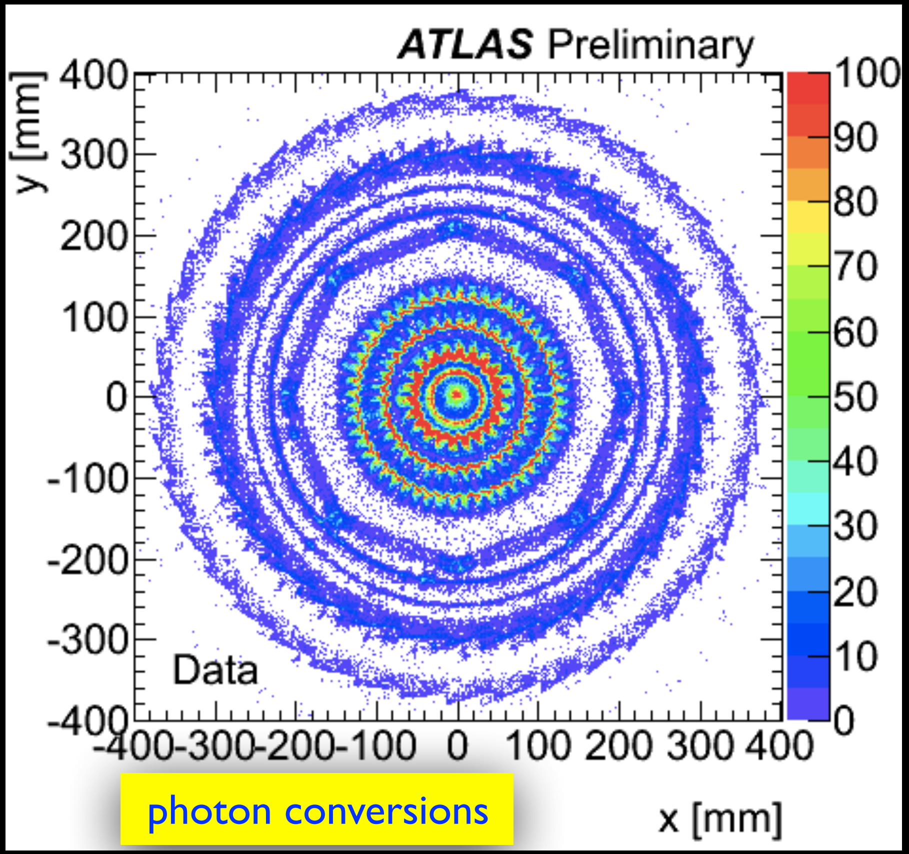


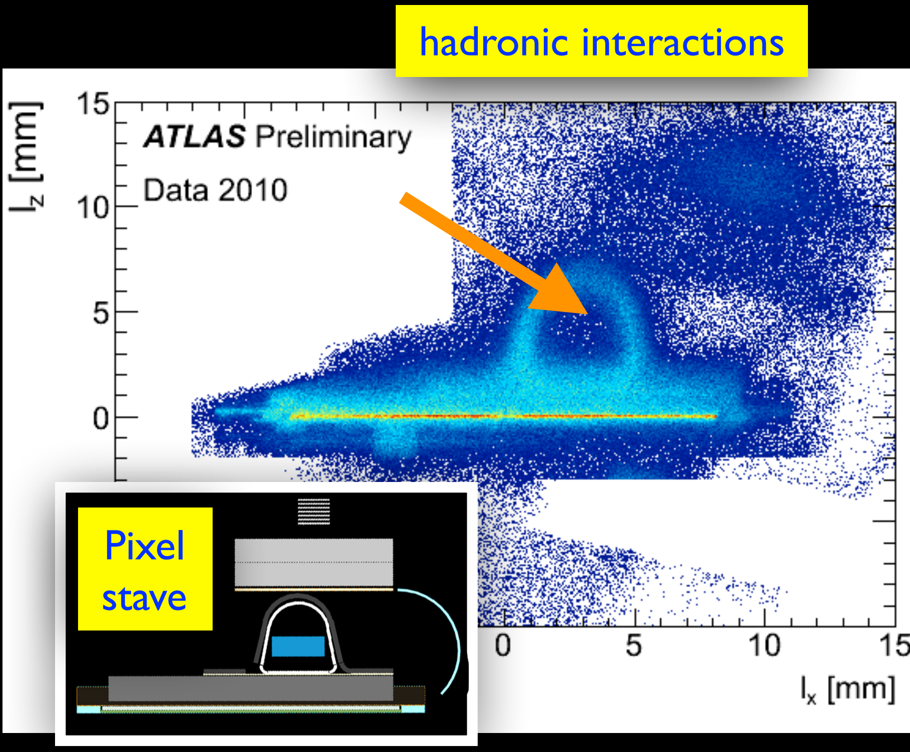


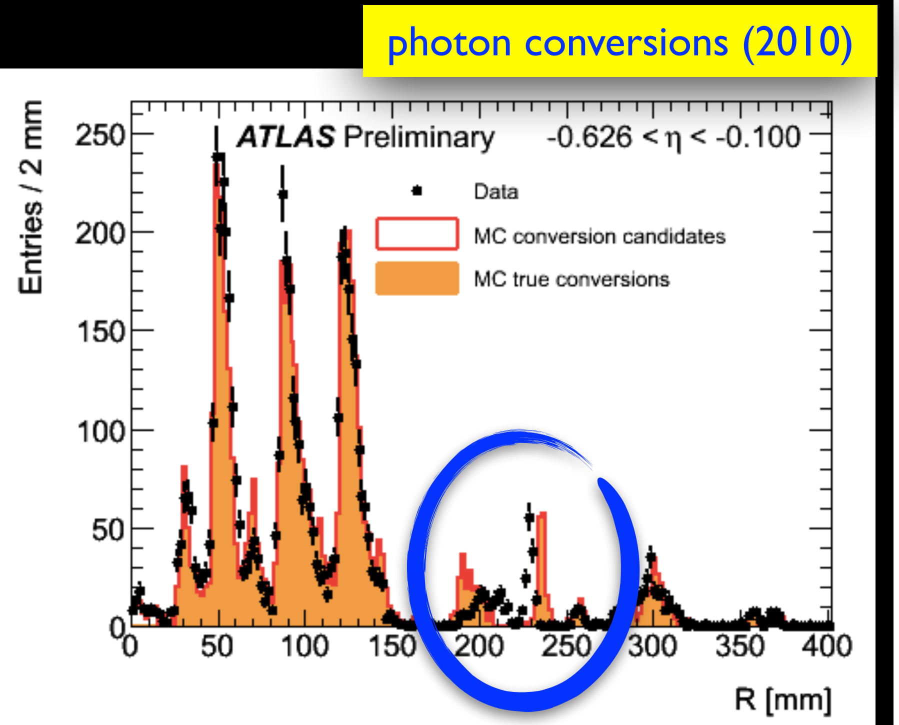
event topologies

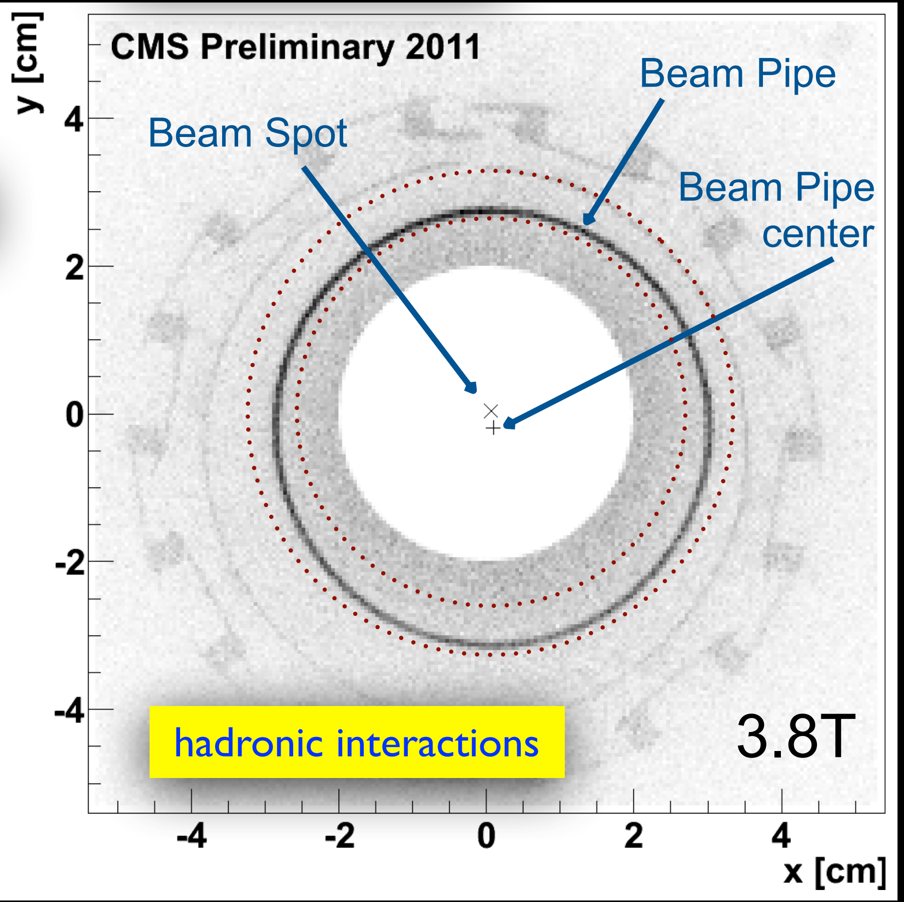
Introduction to Vertex Reconstruction

- vertexing as well tool for material studies
  - ➡ remember tracker performance limited by material !
- photon conversions
  - ➡ used to study MC vs data
  - ➡ can normalise acceptance e.g. on "known" beam pipe
- hadronic interactions
  - ➡ larger multiplicity and opening angles allows for better positions resolution
  - ➡ e.g. ATLAS corrected in Monte Carlo the amount of liquid in Pixel cooling pipes







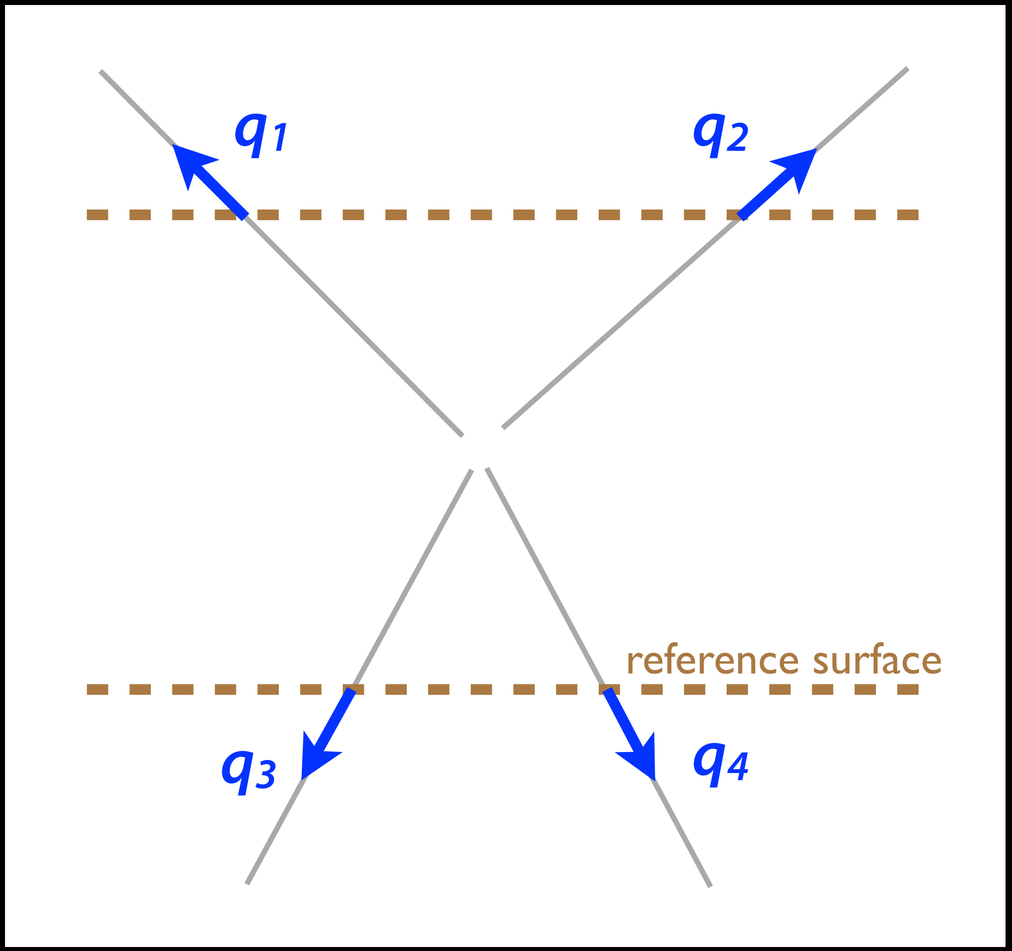


- discuss vertex fitting and finding technique
  - ➡ Least Square and Kalman Filter vertex fitter
  - ➡ adaptive vertex fitting, vertex finding and related
- examples for vertexing applications
  - ➡ beam spot, primary vertex reconstruction and jet-vertex-fraction
  - ➡ b-jet tagging techniques

Vertex Fitting and Finding

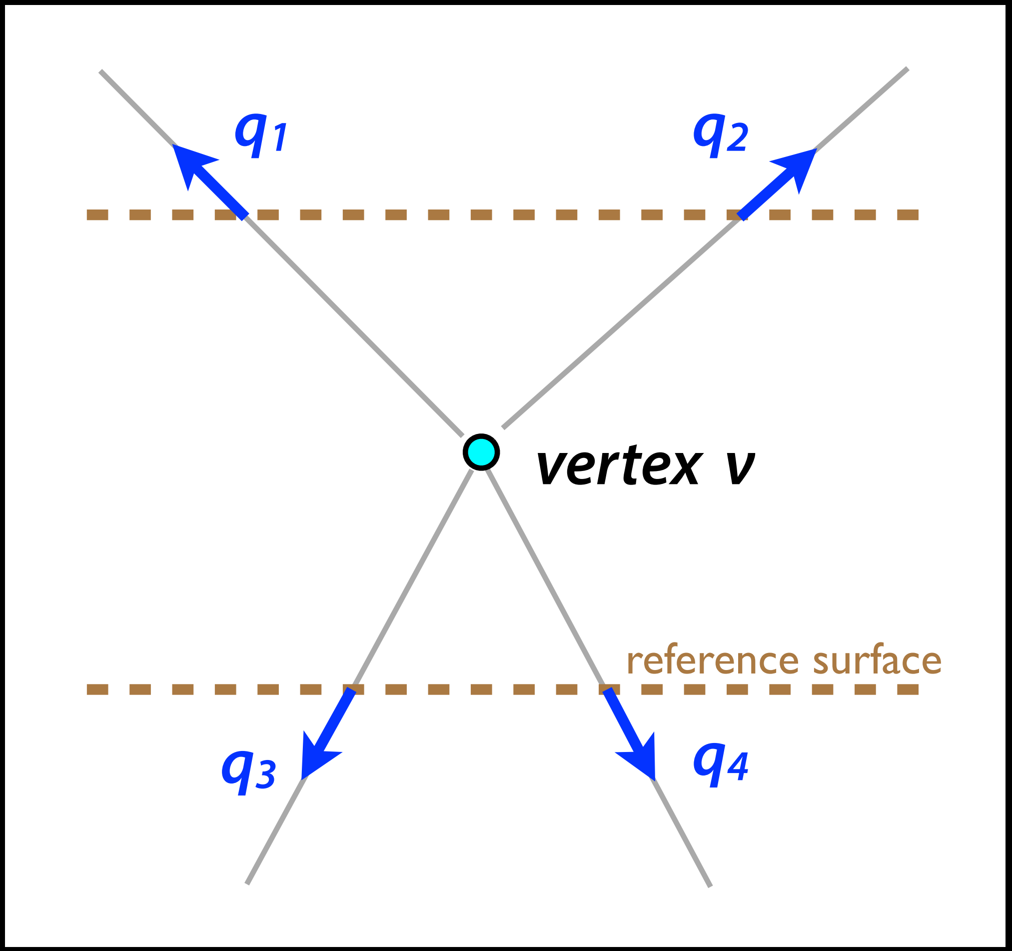
Vertex Fitting Formalism

- task of a vertex fit:
  - ➡ start from a set of measured track parameters q_i



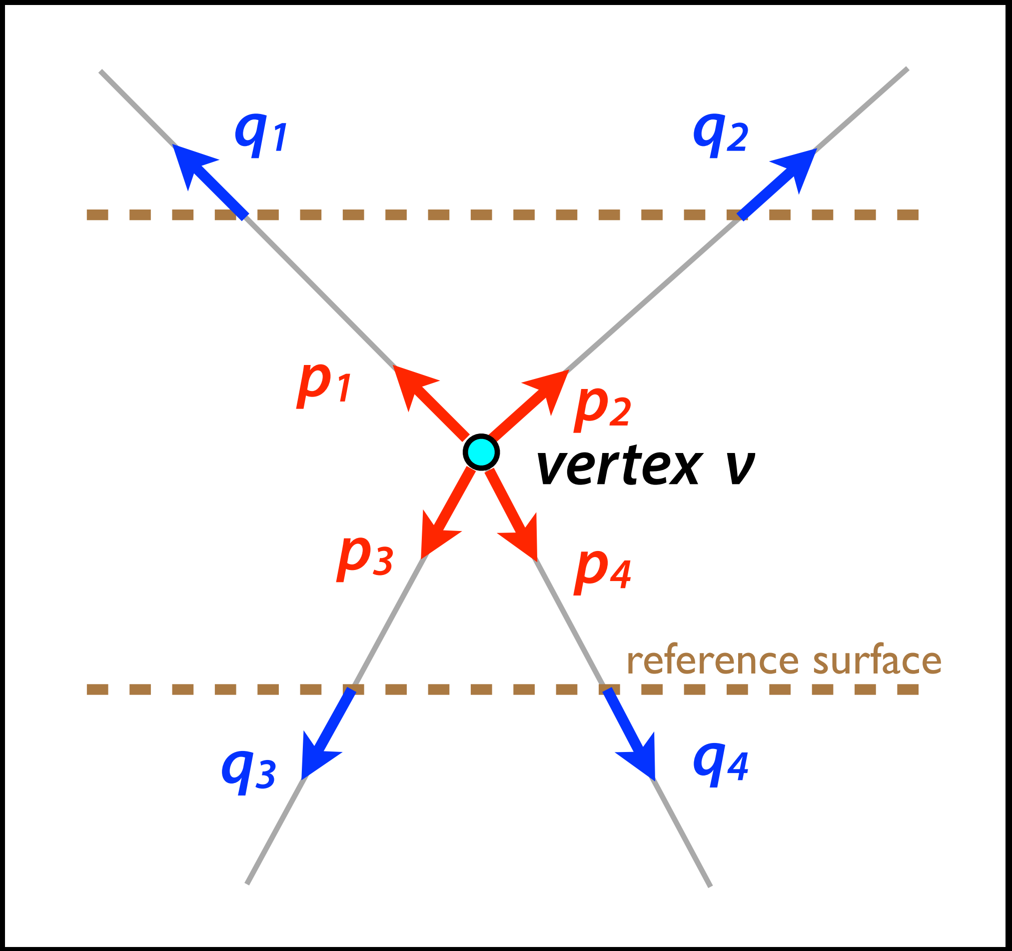
Vertex Fitting Formalism

- task of a vertex fit:
  - ➡ start from a set of measured track parameters q_i
  - ➡ estimate the vertex position v

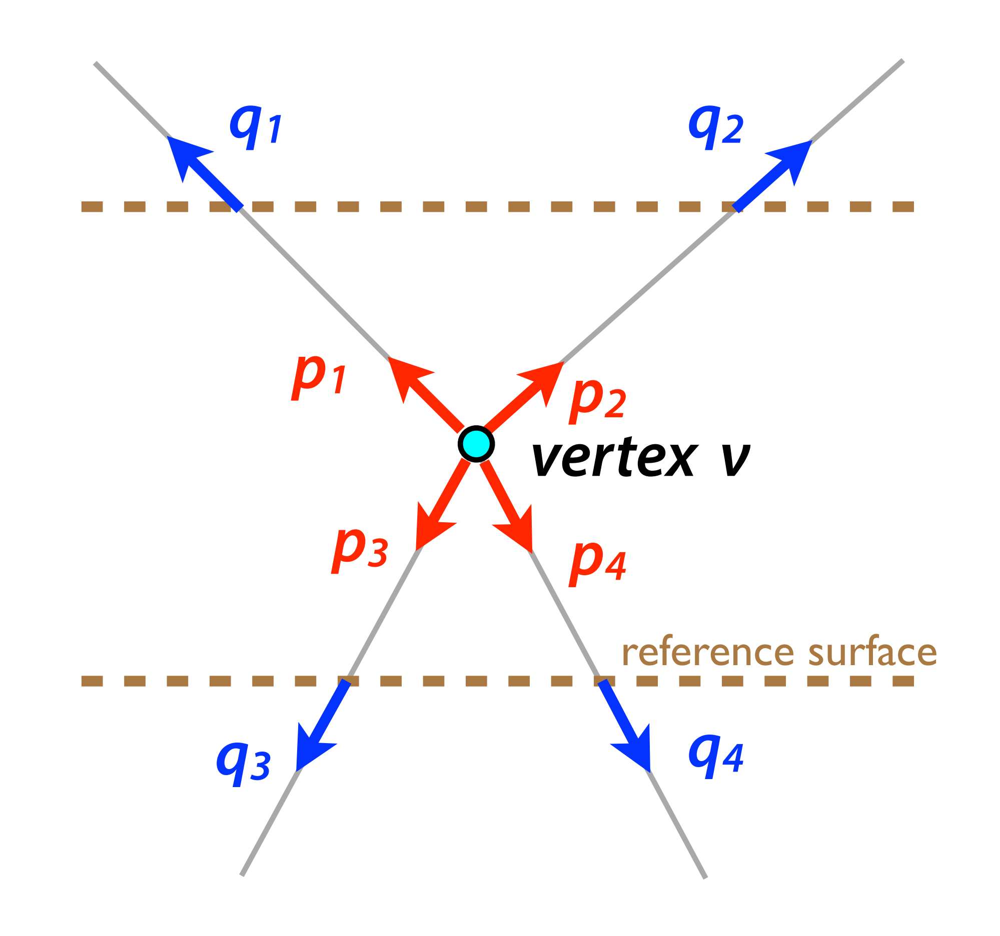


Vertex Fitting Formalism

- task of a vertex fit:
  - ➡ start from a set of measured track parameters q_i
  - ➡ estimate the vertex position v
  - ➡ and the parameters pi at the vertex



Vertex Fitting Formalism



- **task** of a vertex fit:
  - ➡ start from a set of measured track parameters q_i
  - ➡ estimate the vertex position v
  - ➡ and the parameters pi at the vertex

- 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 v and parameters q_i at vertex

ε_i ~ error of q_i (noise term)

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

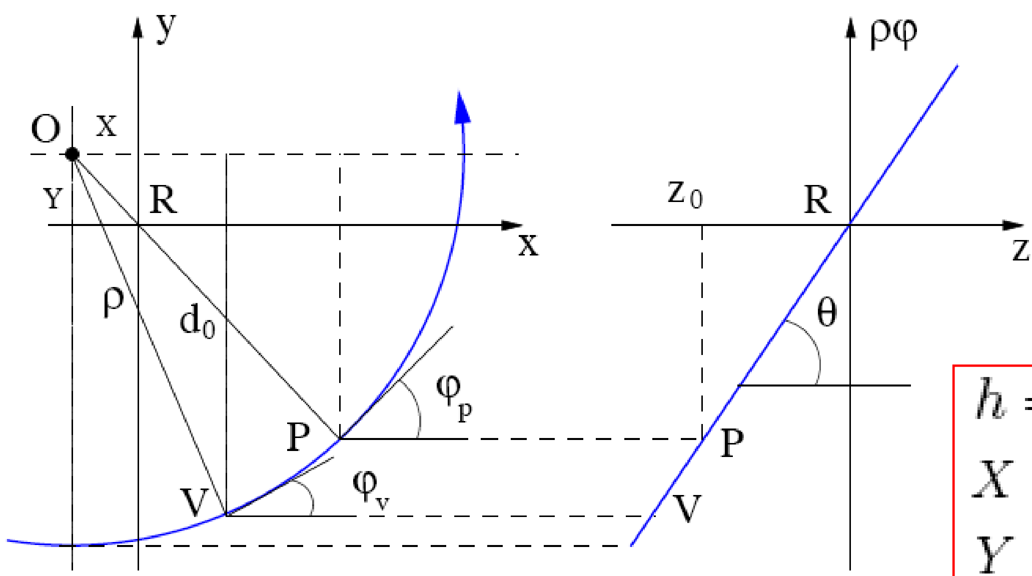
  - ➡ in practice: h_i is derived from parameter representation and propagator f :

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

commonly used is perigee representation for h_i

Helix Propagation for Perigee Parameters

- most commonly used by vertexing codes
  - ➡ summary of equations for propagation of perigee parameters from reference point ***P*** to vertex ***V***, and the corresponding Jacobian matrices ***A*** and ***B*** for fitting



$$
\begin{aligned}
x(\phi) &= x_R - d_{0P}\sin\phi_P + \rho(\sin\phi_P - \sin\phi),\\
y(\phi) &= y_R + d_{0P}\cos\phi_P + \rho(\cos\phi - \cos\phi_P),\\
z(\phi) &= z_R + z_P + \rho\frac{(\phi_P - \phi)}{\tan\theta_P},
\end{aligned}
$$

$$h = sign(\rho)$$

$$
\begin{aligned}
X &= x_V - x_R + \rho\sin\phi_V & R &= X\cos\phi_V + Y\sin\phi_V\\
Y &= y_V - y_R - \rho\cos\phi_V & Q &= X\sin\phi_V - Y\cos\phi_V\\
S &= \sqrt{X^2 + Y^2} & \Delta\phi &= \phi_P - \phi_V.
\end{aligned}
$$

(definitions)

Results for position and momentum jacobian:

$$
A = \frac{\partial(d_{0P}, z_P, \phi_P, \theta_P, q/p)}{\partial(x_V, y_V, z_V)} = \begin{pmatrix} -h\frac{X}{S} & -h\frac{Y}{S} & 0\\ \frac{\rho}{\tan\theta}\frac{Y}{S^2} & -\frac{\rho}{\tan\theta}\frac{X}{S^2} & 1\\ -\frac{Y}{S^2} & \frac{X}{S^2} & 0\\ 0 & 0 & 0\\ 0 & 0 & 0 \end{pmatrix}
$$

$$
B = \frac{\partial(d_{0P}, z_P, \phi_P, \theta_P, q/p)}{\partial(\phi_V, \theta, q/p)} =
$$

$$
\begin{pmatrix} -\frac{h\rho}{S}R & \frac{\rho}{\tan\theta}\left[1 - \frac{h}{S}Q\right] & -\frac{\rho}{q/p}\left[1 - \frac{h}{S}Q\right]\\ \frac{\rho}{\tan\theta}\left[1 - \frac{\rho}{S^2}Q\right] & \rho\left[\Delta\phi + \frac{\rho}{S^2\tan^2\theta}R\right] & \frac{\rho}{q/p\tan\theta}\left[\Delta\phi - \frac{\rho}{S^2}R\right]\\ \frac{\rho}{S^2}Q & -\frac{\rho}{S^2\tan\theta}R & \frac{\rho}{S^2 q/p}R\\ 0 & 1 & 0\\ 0 & 0 & 1 \end{pmatrix}
$$

G.Piacquadio

Formulating a Least Square Vertex Fit

- same approach as for Least Square track fit:

Least Square function to minimise for vertex fit:

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

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

Formulating a Least Square Vertex Fit

- same approach as for Least Square track fit:

Least Square function to minimise for vertex fit:

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

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

linearise the problem around starting values v_0 and $q_{0,i}$:

$$v \longrightarrow v_0 + \delta v$$

$$p_i \longrightarrow p_{i,0} + \delta p_i$$

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

yields:

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

Formulating a Least Square Vertex Fit

- same approach as for Least Square track fit:

Least Square function to minimise for vertex fit:

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

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

linearise the problem around starting values v_0 and $q_{0,i}$:

$$v \rightarrow v_0 + \delta v$$
$$p_i \rightarrow p_{i,0} + \delta p_i$$

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

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

Solution to Least Square Vertex Fit

➡ let's solve the system of linear equations:

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

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

Solution to Least Square Vertex Fit

➡ let's solve the system of linear equations:

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

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

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 v

➡ usually one iterates the fit to ensure convergence

Solution to Least Square Vertex Fit

➡ let's solve the system of linear equations:

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

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

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

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

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

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

➡ usually one iterates the fit to ensure convergence

➡ still have to compute the vertex correction to track parameters p_i

➡ but: can obtain a faster vertex fit, if we neglect the δp_i terms as an approximation

Solution to Least Square Vertex Fit

➡ compute the vertex correction to track parameters p_i :

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

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

Solution to Least Square Vertex Fit

➡ compute the vertex correction to track parameters ***p**$_i$* :

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

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

use δν 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

Solution to Least Square Vertex Fit

➡ compute the vertex correction to track parameters p_i :

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

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

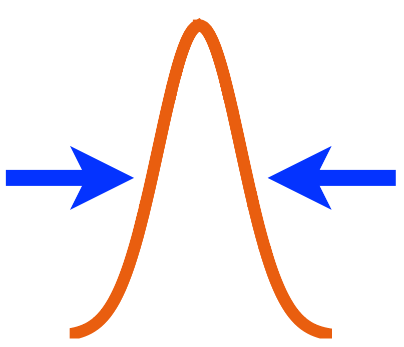
use δv in equation (2) to compute δp_i, gives:

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

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 can be used to improve track momentum measurement at vertex

- improve e.g. invariant mass resolution for reconstructed decays



e.g. J/Ψ signal

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

weight matrix notation

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

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

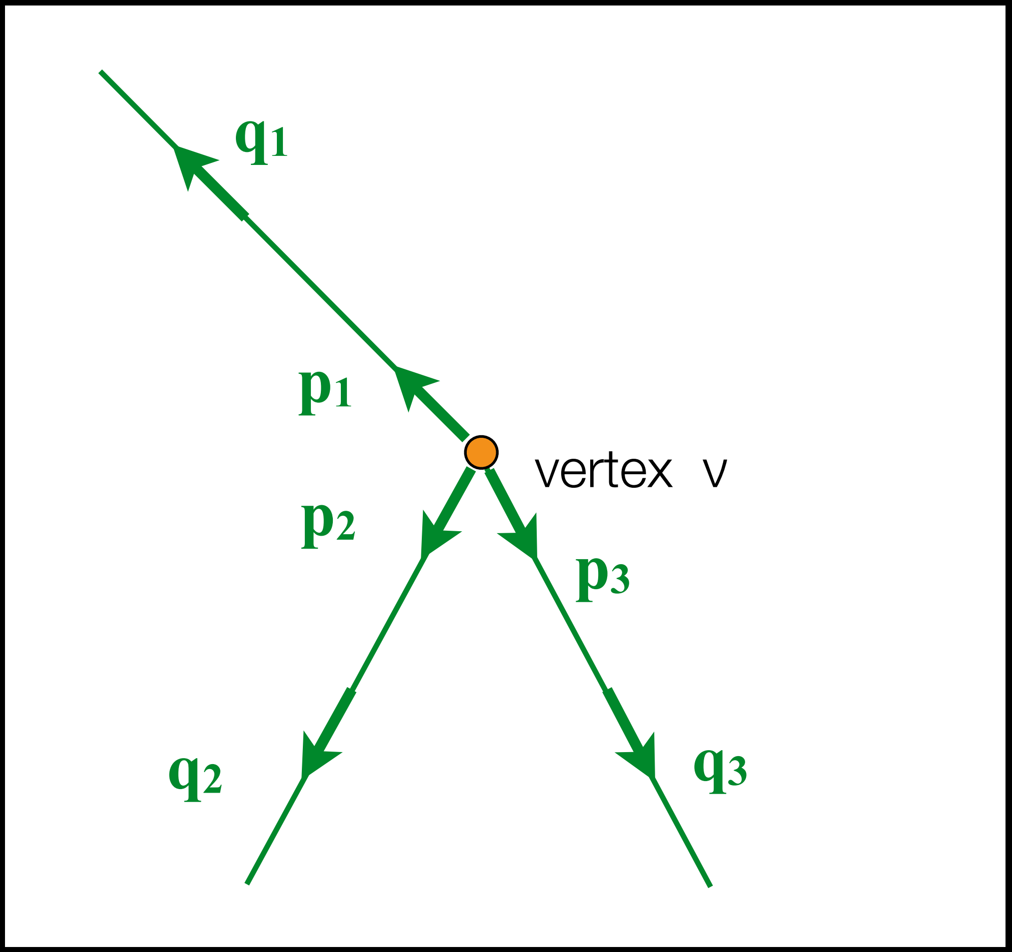
covariance: $$C_i = \left(C_{i-1}^{-1} + A_i^T G_i^B A_i \right)^{-1}$$

2.update to parameters is:

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

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

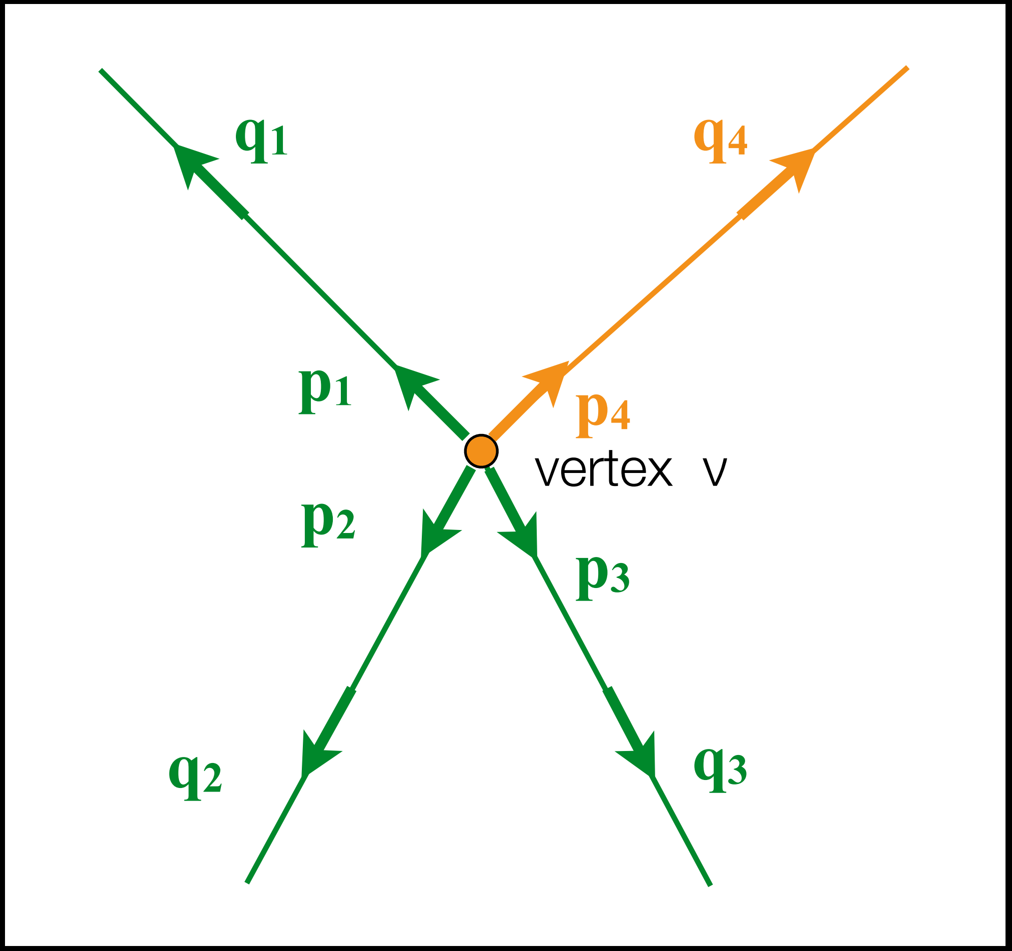
Billoir, Fruhwirth, Catlin et al.



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



weight matrix notation

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

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

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

2. update to parameters is:

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

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

Billoir, Fruhwirth, Catlin et al.

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

weight matrix notation

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

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

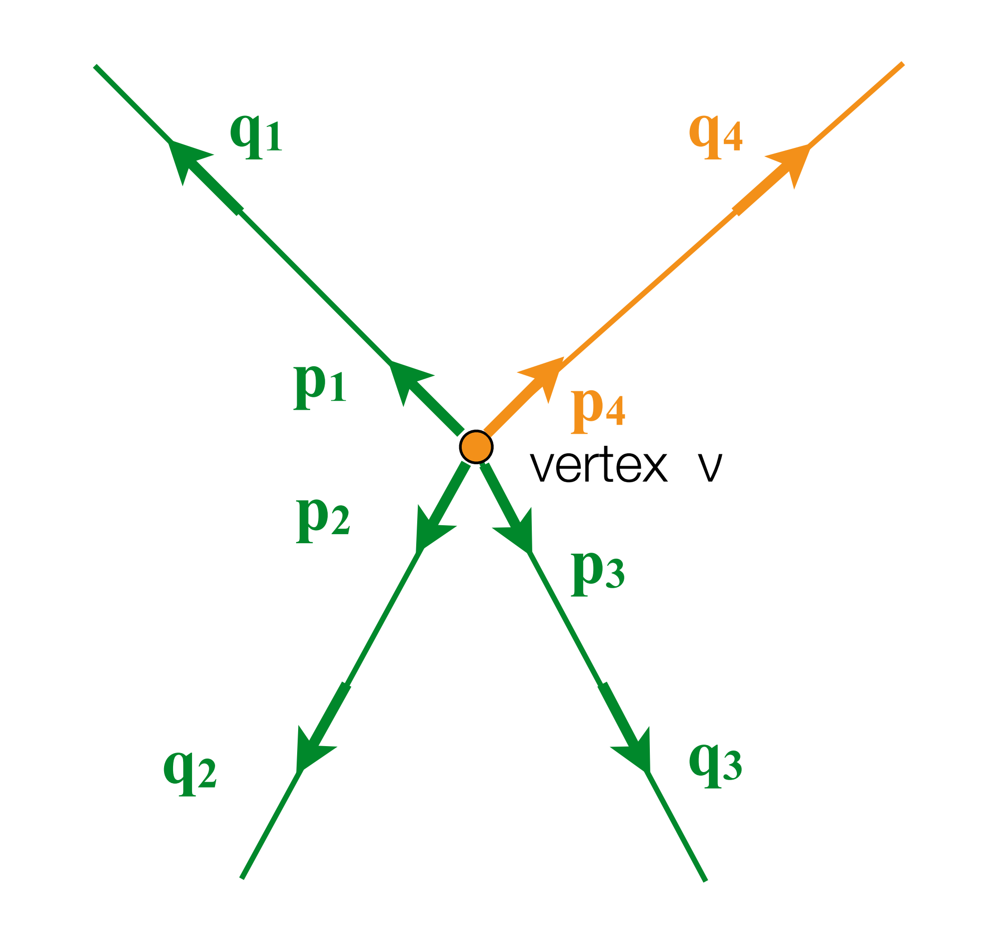
covariance: $C_i = \left(C_{i-1}^{-1} + A_i^T G_i^B A_i \right)^{-1}$

2. update to parameters is:

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

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

Billoir, Fruhwirth, Catlin et al.



- ➡ 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

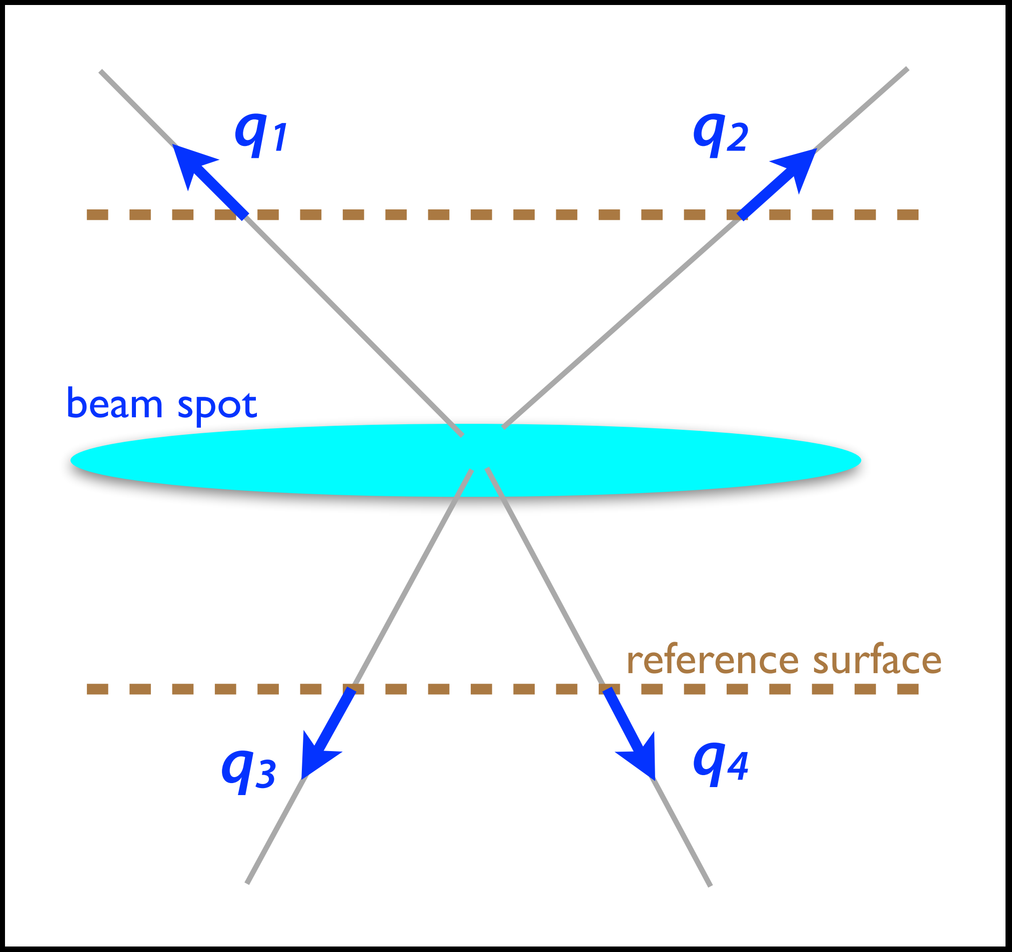
➡ important for primary vertex reconstruction

- beam spot b and its covariance matrix E_b^{-1} determined externally

➡ use beam spot in fit as external constraint

- straight forward in Kalman Filter vertex fit, its the starting vertex:

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

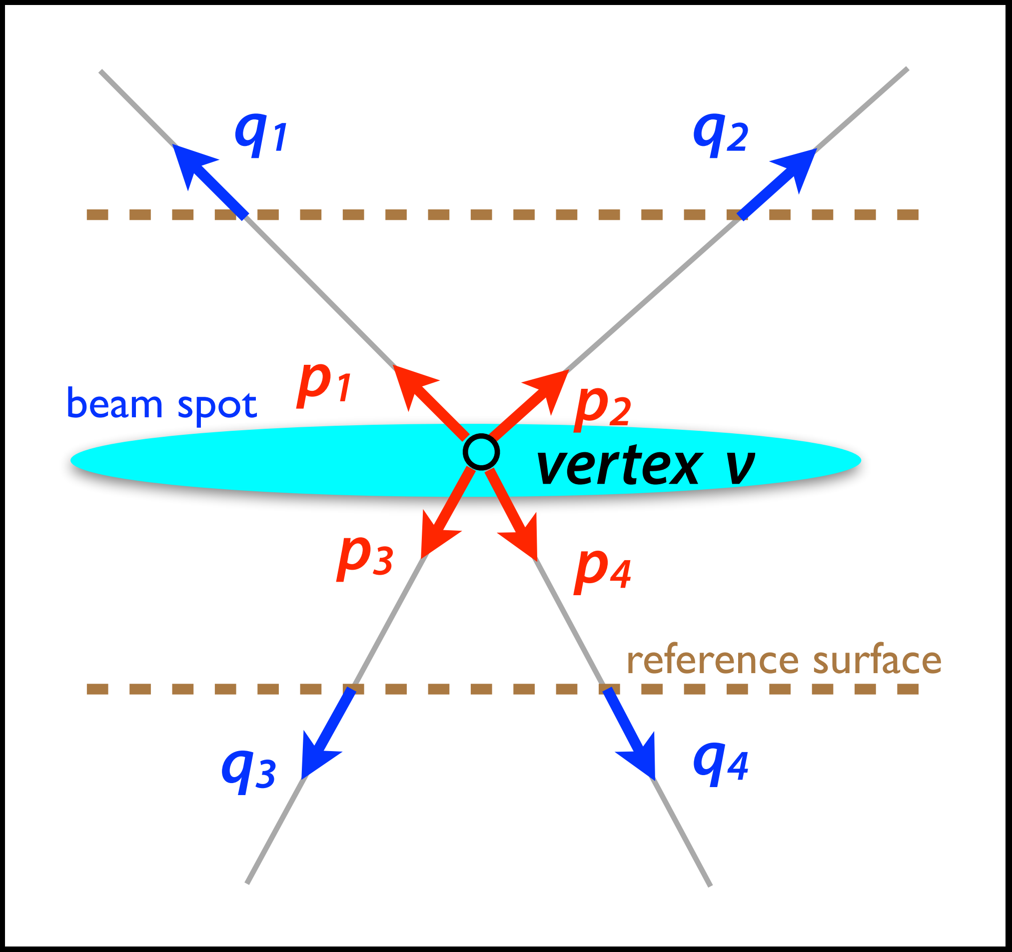


Beam Spot Constraint Fit

➡ important for primary vertex reconstruction
- beam spot b and its covariance matrix E_b^{-1} determined externally

➡ use beam spot in fit as external constraint
- straight forward in Kalman Filter vertex fit, its the starting vertex:

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



Beam Spot Constraint Fit

➡ important for primary vertex reconstruction

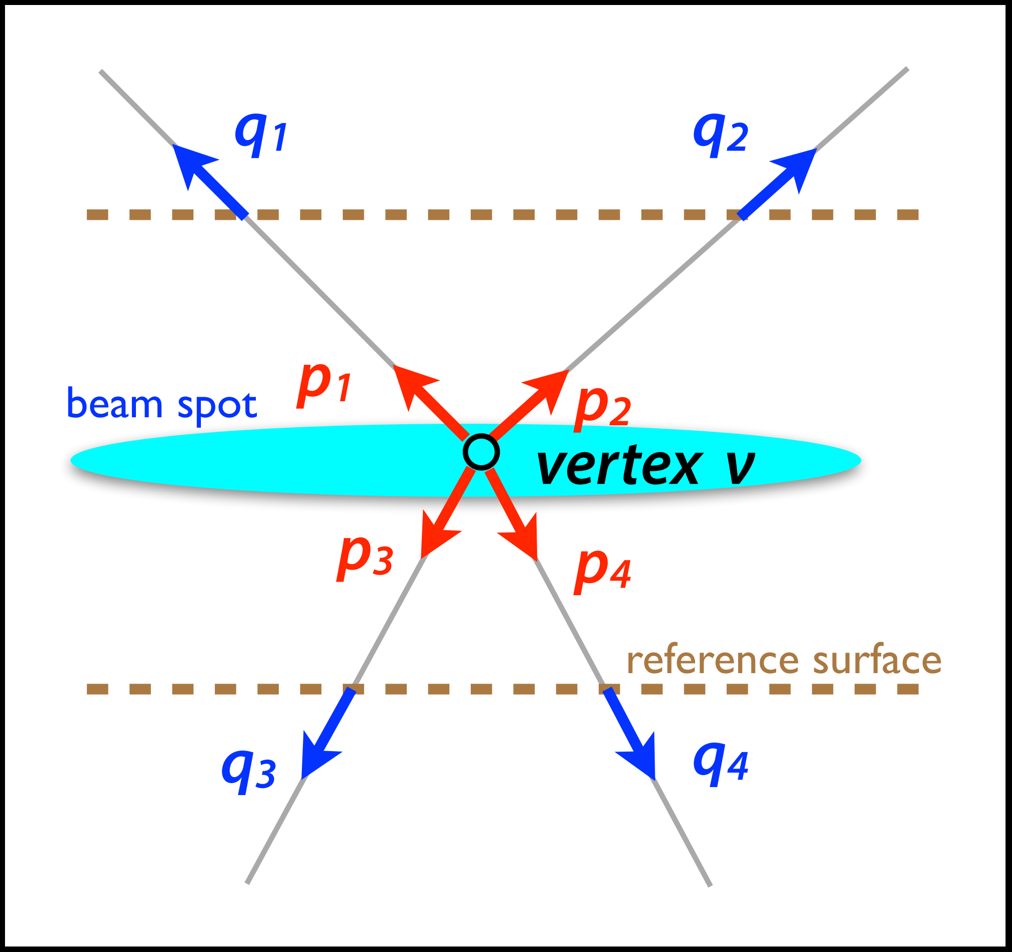
- beam spot b and its covariance matrix E_b^{-1} determined externally

➡ use beam spot 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:

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

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

which can be solved as before...

Inspecting Outliers

- common problem:
  - ➡ fit quality is bad, need to calculate the *χ2* contribution of each track to overall fit to identify outliers
  - ➡ need to compare *χ2* of fit to all tracks to the *χ2* of fit with 1 track less:

$$\Delta\chi_i^2 = \Delta q_i^T \cdot G_i \cdot \Delta q_i + \left(\Delta q_i - A_i \delta v\right)^T \cdot G_i^B A_i C^{-1} A_i^T G_i^B \cdot \left(\Delta q_i - A_i \delta v\right)$$

track *χ2* — change to *χ2* from including this track in *δv*

  - ➡ used to iteratively remove outliers

Inspecting Outliers

- common problem:
  - ➡ fit quality is bad, need to calculate the ***χ2*** contribution of each track to overall fit to identify outliers
  - ➡ need to compare ***χ2*** of fit to all tracks to the ***χ2*** of fit with 1 track less:

$$\Delta\chi_i^2 = \Delta q_i^T \cdot G_i \cdot \Delta q_i + \left(\Delta q_i - A_i\delta v\right)^T \cdot G_i^B A_i C^{-1} A_i^T G_i^B \cdot \left(\Delta q_i - A_i\delta v\right)$$

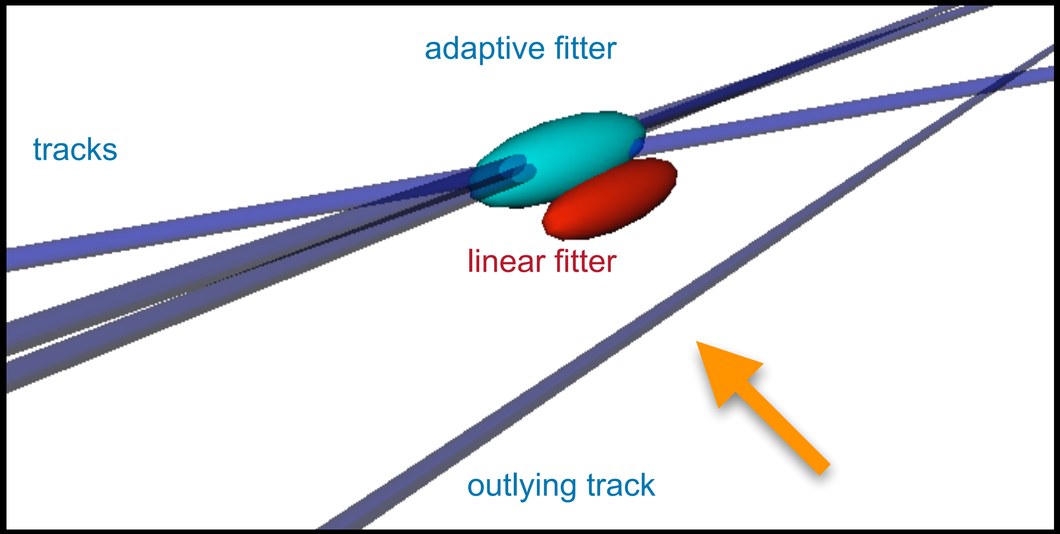
track ***χ2*** change to ***χ2*** from including this track in ***δv***

  - ➡ used to iteratively remove outliers

- application: Iterative Vertex Finder for multiple vertices
  - ➡ fit all tracks into 1 vertex
  - ➡ remove worst track one by one, until fit ***χ2*** is acceptable
  - ➡ take removed tracks and try to find next vertex
  - ➡ repeat until no further vertex with at least 2 tracks can be found

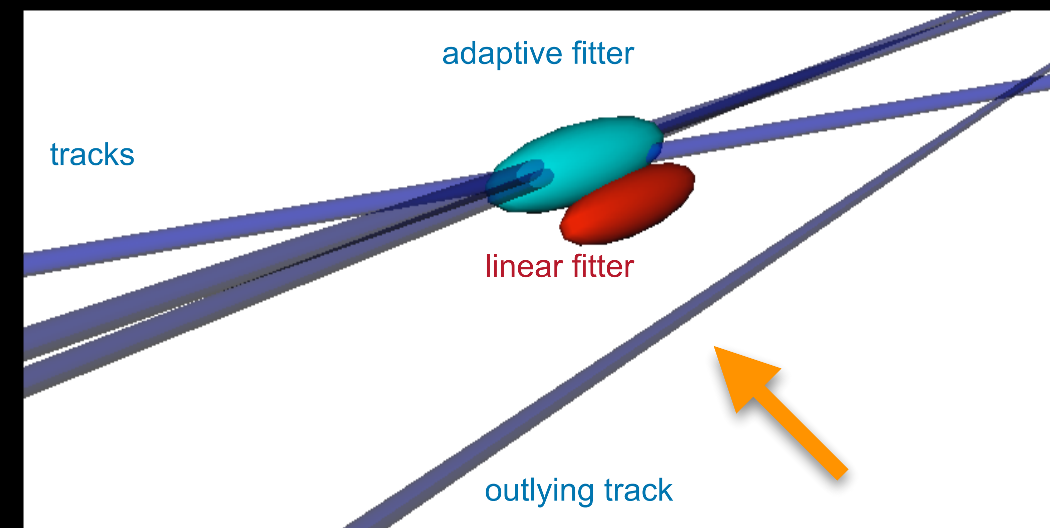
Adaptive Vertex Fit

- robust fitting can suppress effects of outliers on fit result
  - ➡ concept used for adaptive track fitting in Deterministic Annealing Filter (DAF)
  - ➡ can be applied as well on vertex fitting

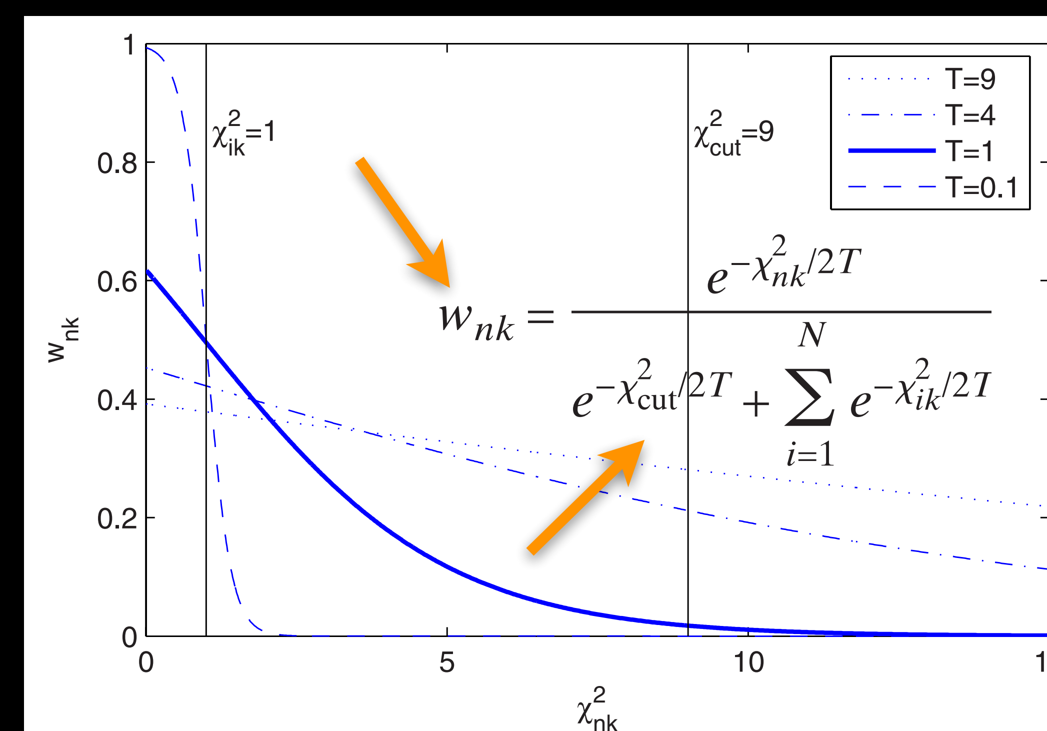


Adaptive Vertex Fit

- robust fitting can suppress effects of outliers on fit result
  - ➡ concept used for adaptive track fitting in Deterministic Annealing Filter (DAF)
  - ➡ can be applied as well on vertex fitting



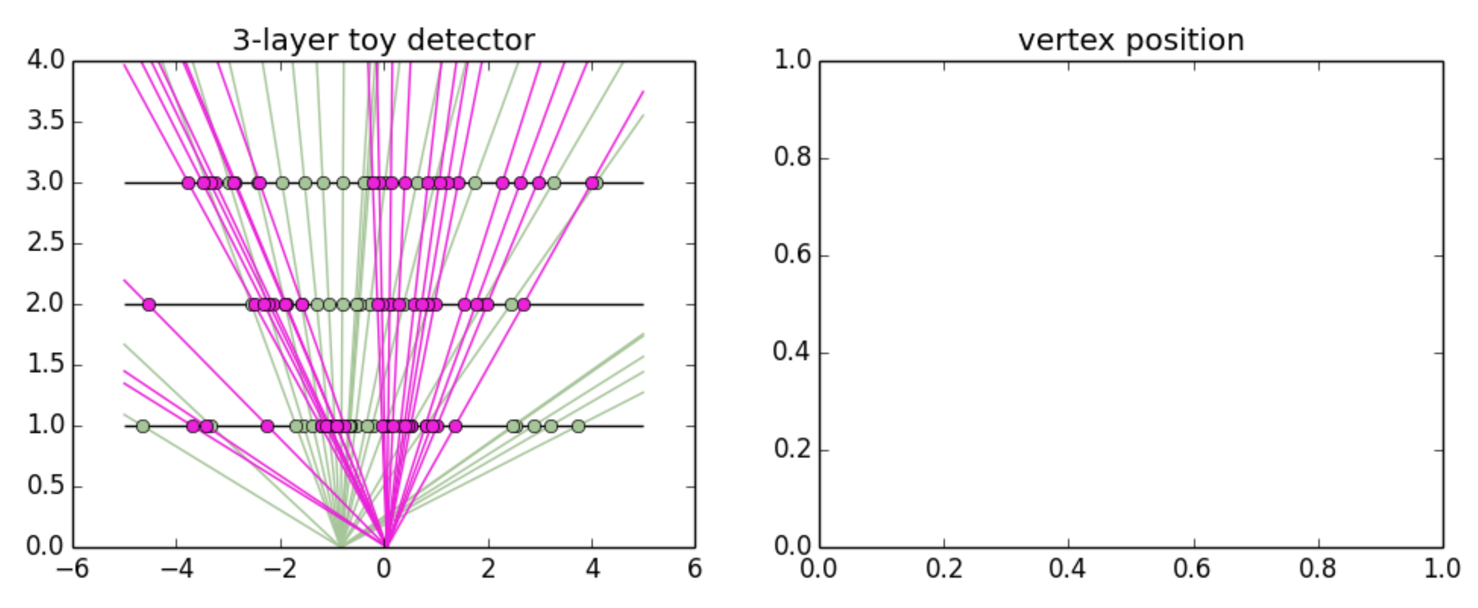
- technique called Adaptive Vertex Fit
  - ➡ can be implemented as iterative, re-weighted Kalman Filter
    - w_{nk} is weight of track k w.r.t. vertex n
    - like for DAF, uses Boltzman factors with "temperature" T (χ^2_{cut} is tuning parameter)
    - reducing T results in automatically down-weighted outlying tracks
  - ➡ technique commonly used in ATLAS and CMS



- extension for Multi-Vertex-Fitter
  - ➡ adaptive fit for n vertices in one go
  - ➡ compute track weights w.r.t. each vertex, such that vertices compete for tracks

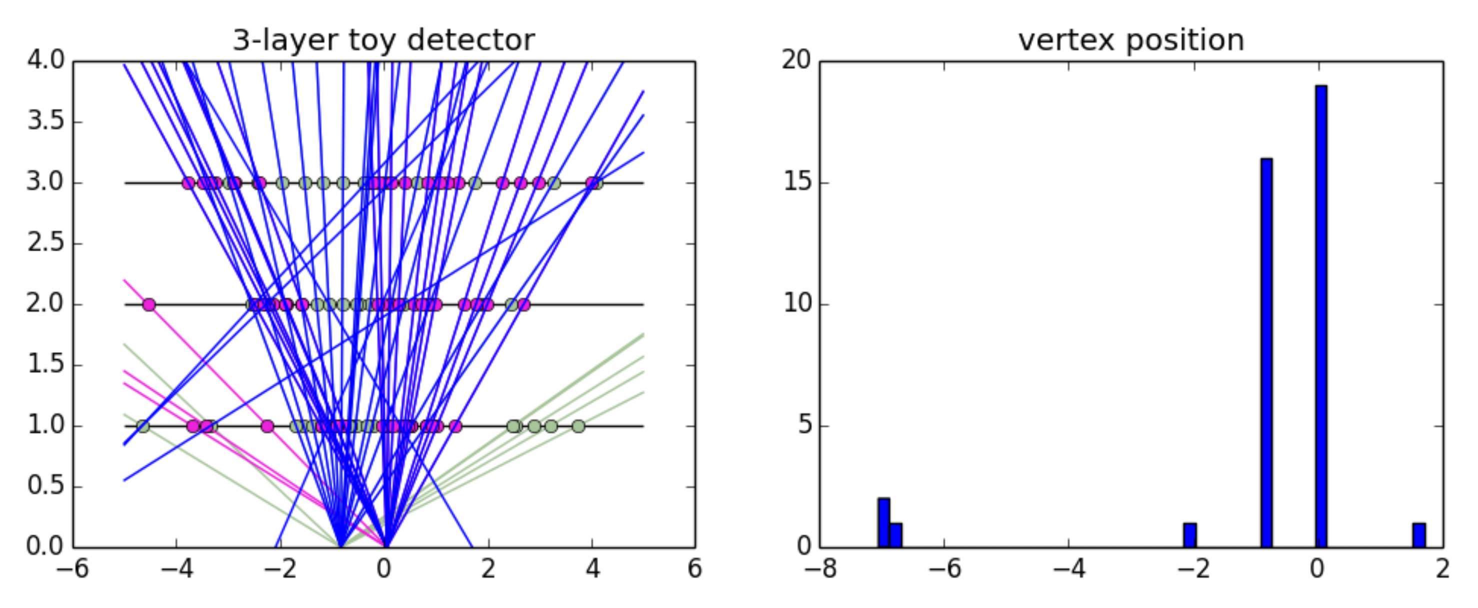
Strategies to Seed the Vertex Finding

- vertex z-scan on beam line
  - ➡ histogram technique that searches for peaks in ***z0*** of hit combinations extrapolated to beam line
  - ➡ used e.g. to seed primary vertex finding or to constrain HLT tracking to point to primary vertex



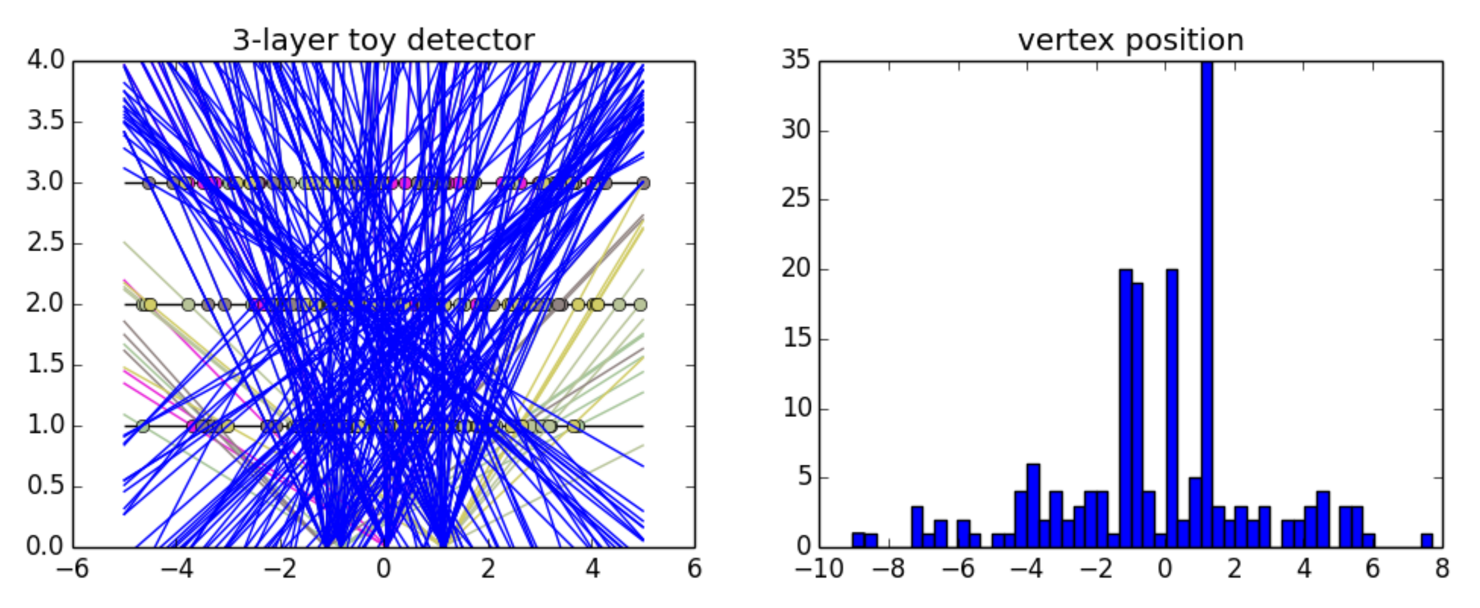
Strategies to Seed the Vertex Finding

- vertex z-scan on beam line
  - ➡ histogram technique that searches for peaks in ***z0*** of hit combinations extrapolated to beam line
  - ➡ used e.g. to seed primary vertex finding or to constrain HLT tracking to point to primary vertex



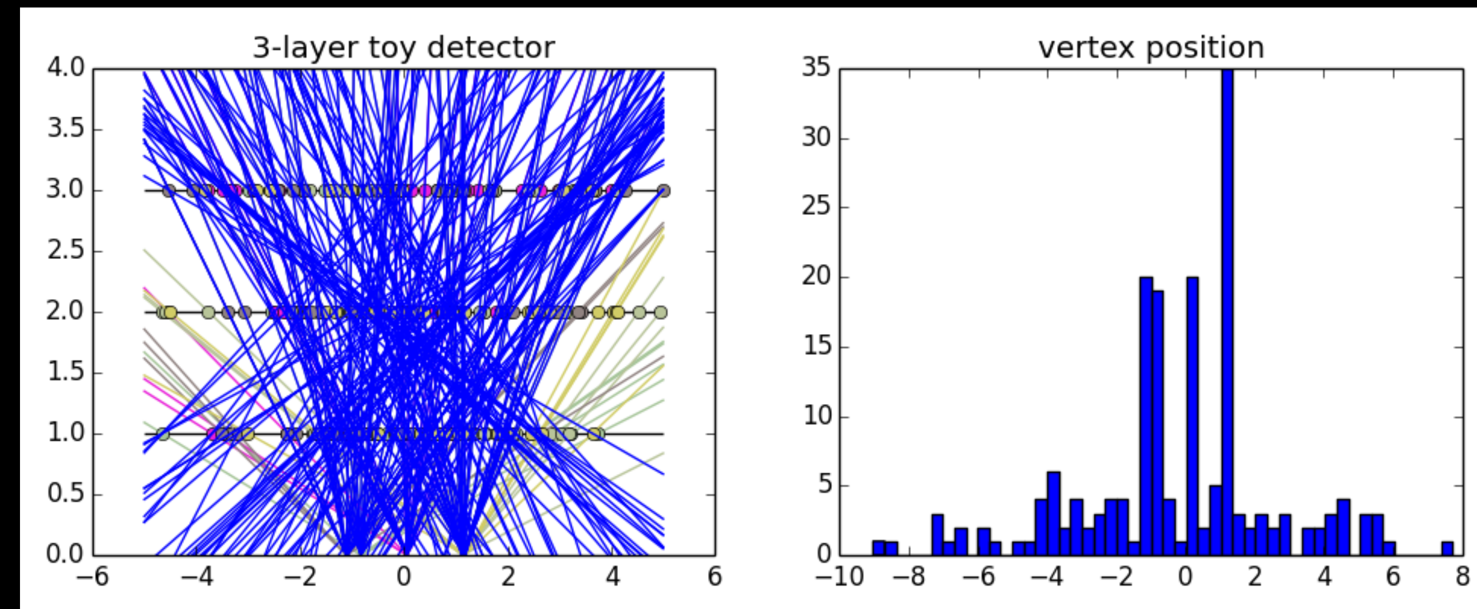
Strategies to Seed the Vertex Finding

- **vertex z-scan on beam line**
  - ➡ histogram technique that searches for peaks in ***z0*** of hit combinations extrapolated to beam line
  - ➡ used e.g. to seed primary vertex finding or to constrain HLT tracking to point to primary vertex

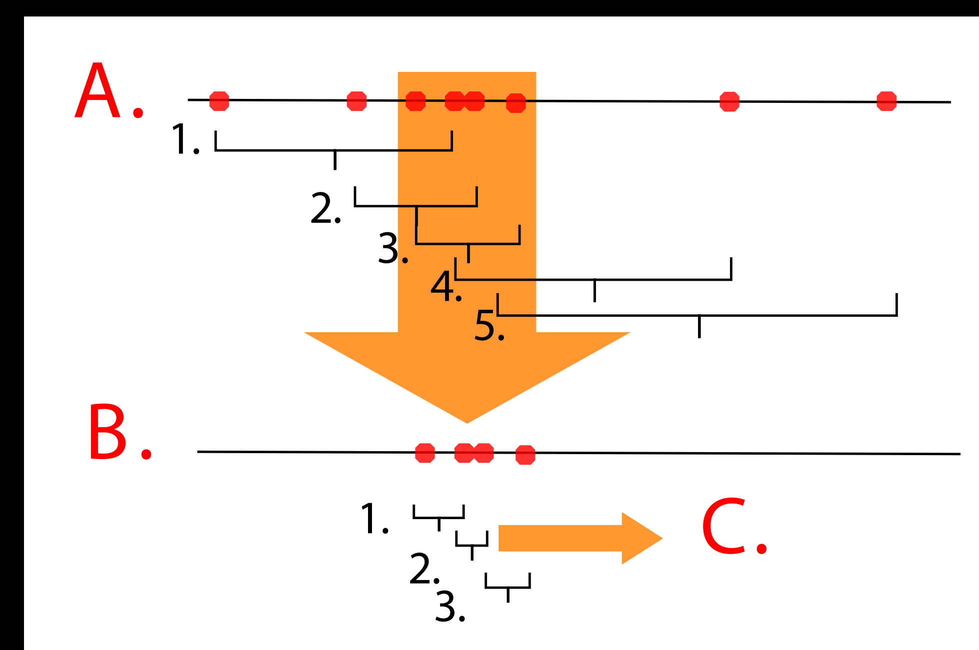


Strategies to Seed the Vertex Finding

- **vertex z-scan on beam line**
  - ➡ histogram technique that searches for peaks in ***z0*** of hit combinations extrapolated to beam line
  - ➡ used e.g. to seed primary vertex finding or to constrain HLT tracking to point to primary vertex

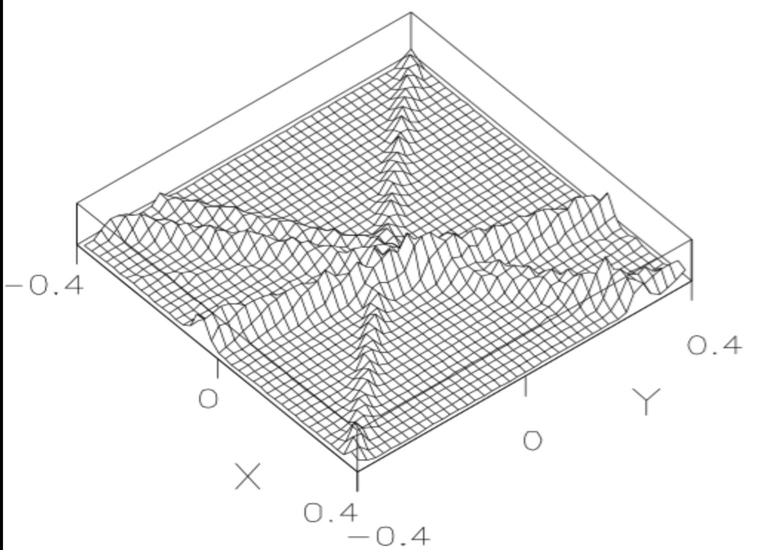


- **half sample mode algorithm**
  - ➡ find points of closest approach between all track pairs
  - ➡ in each of the 3 projections:
    - A. try all the intervals which cover 50 % of the points and take the smallest one
      (in this case number 3.)
    - B. iterate again until you have ≤ 3 points left
      (in this case number 2.)
    - C. take the mean of the 2 or 3 remaining
  - ➡ defines vertex seed, find matching tracks...



Topological Vertex Finder (ZVTOP)

- example for an inclusive vertex finder
  - ➡ very powerful, developed by SLD experiment in the 1990th



track Gaussian tube function $f_i(v)$

- 3 dimensional vertex search
  - ➡ construct for each track Gaussian probability tube $f_i(v)$

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

  here r is point of closest approach of track i to point v
  - ➡ find all points v where $f_i(v)$ is significant for 2 tracks

Topological Vertex Finder (ZVTOP)

- example for an inclusive vertex finder
  - ➡ very powerful, developed by SLD experiment in the 1990th
- 3 dimensional vertex search
  - ➡ construct for each track Gaussian probability tube $f_i(v)$

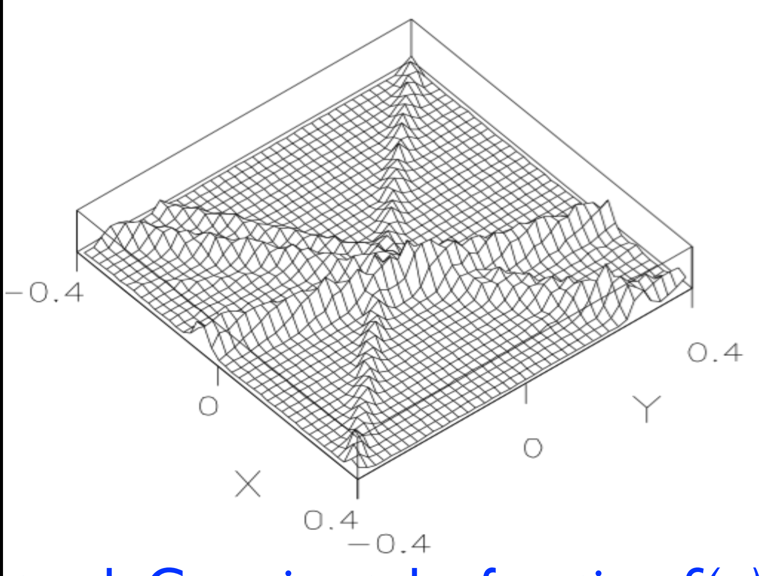
$$f_i(\boldsymbol{v}) = \exp\left[-\frac{1}{2}(\boldsymbol{v}-\boldsymbol{r})^{\mathrm{T}} \boldsymbol{V}_i^{-1}(\boldsymbol{v}-\boldsymbol{r}) \right]$$

here r is point of closest approach of track i to point v

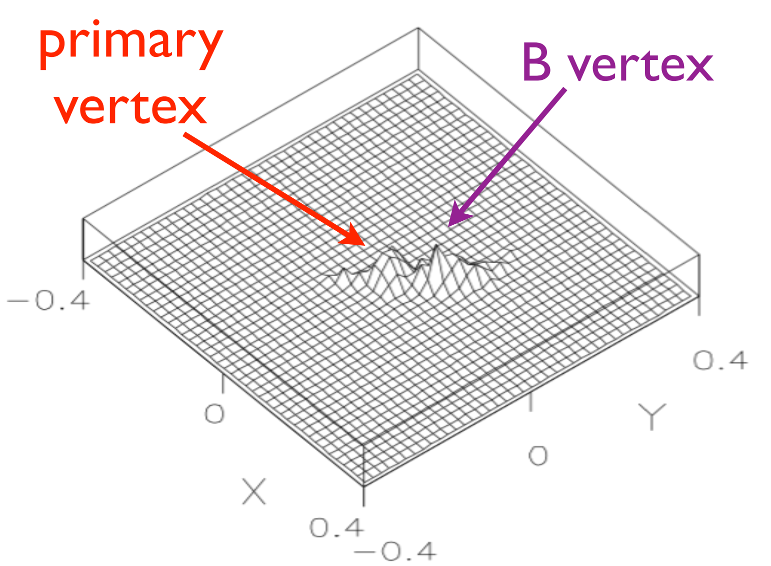
  - ➡ find all points v where $f_i(v)$ is significant for 2 tracks
  - ➡ define vertex probability function $V(r)$ using all tracks around those points v

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

- search for maxima, merge nearby vertex candidates
- run a vertex fit on the set of tracks

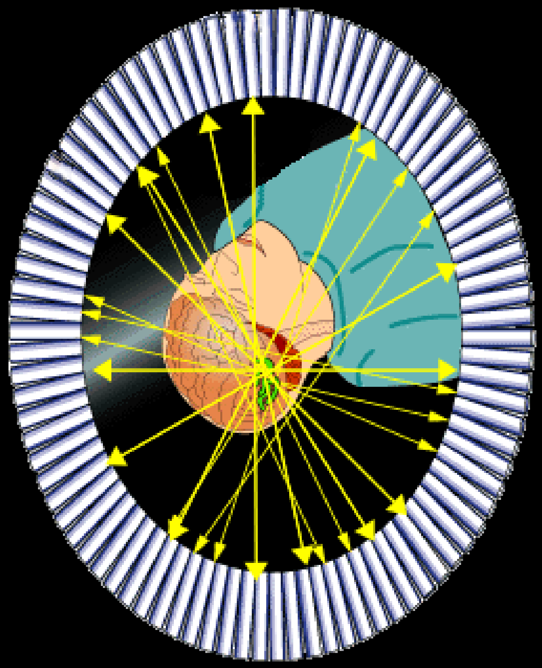


track Gaussian tube function $f_i(v)$



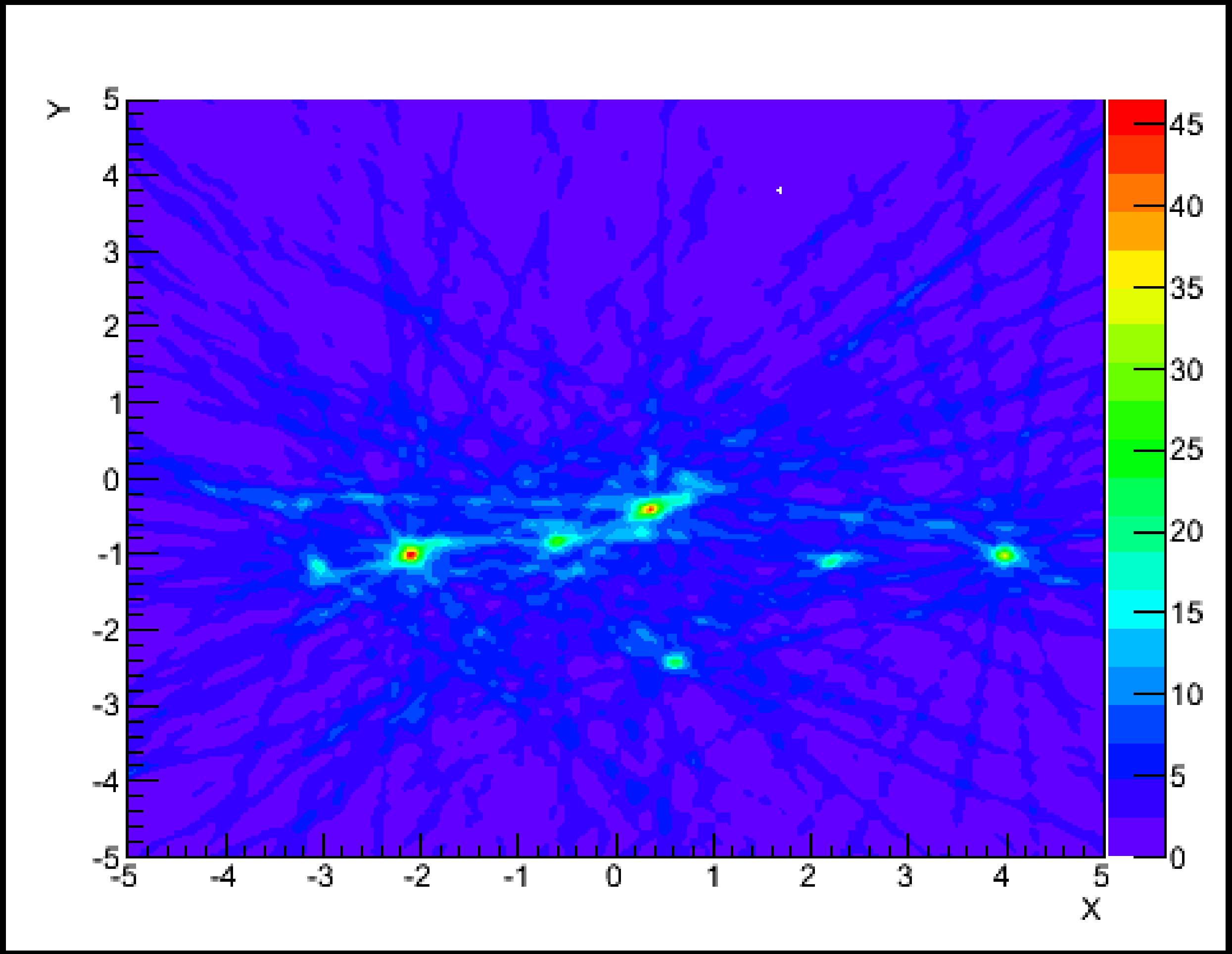
vertex probability function $V(r)$

Medical Imaging inspired Vertexing

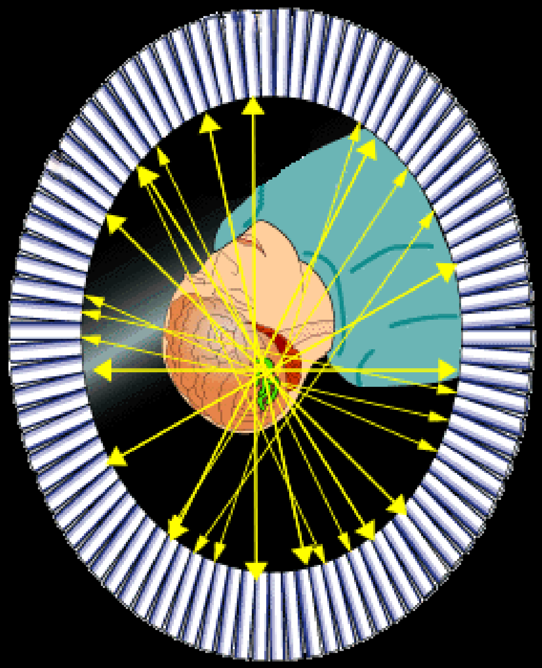


- based on inverse Radon transformation
  - ➡ inspired by imaging techniques used for PET scans
  - ➡ vertex finder is essentially a variant of ZVTOP
    - see: 2012 J. Phys.: Conf. Ser. 396 022021
  - ➡ potentially faster with high pileup
    - evaluated e.g. in ATLAS for primary vertex finding

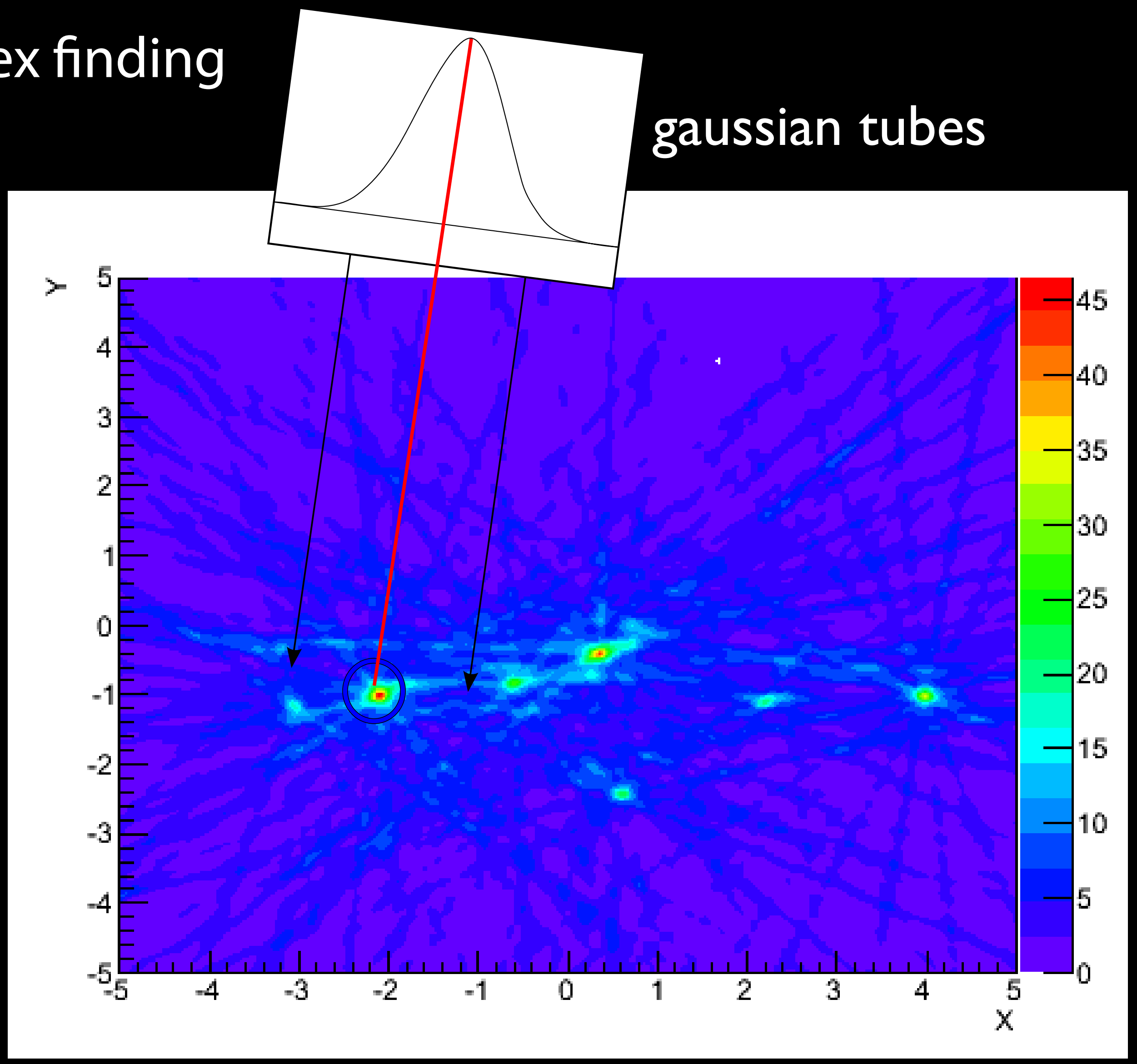
- steps for vertex finder:
  - ➡ create 3D track density maps



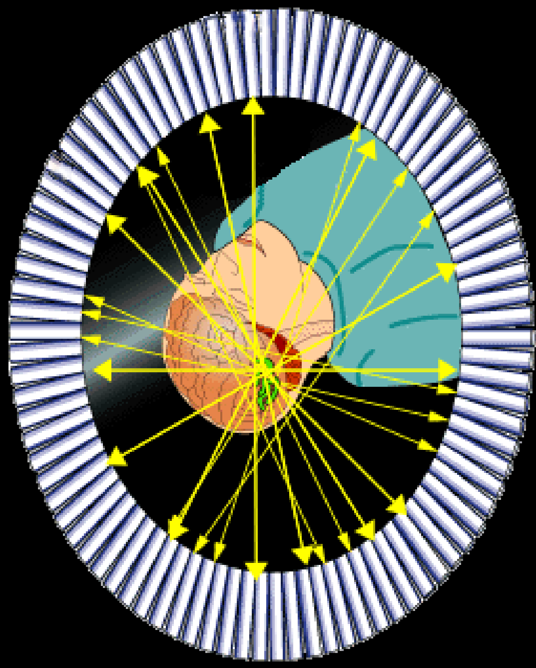
Medical Imaging inspired Vertexing



- based on inverse Radon transformation
  - ➡ inspired by imaging techniques used for PET scans
  - ➡ vertex finder is essentially a variant of ZVTOP
    - see: 2012 J. Phys.: Conf. Ser. 396 022021
  - ➡ potentially faster with high pileup
    - evaluated e.g. in ATLAS for primary vertex finding
- steps for vertex finder:
  - ➡ create 3D track density maps
  - ➡ Fourier transform the Gaussian tubes



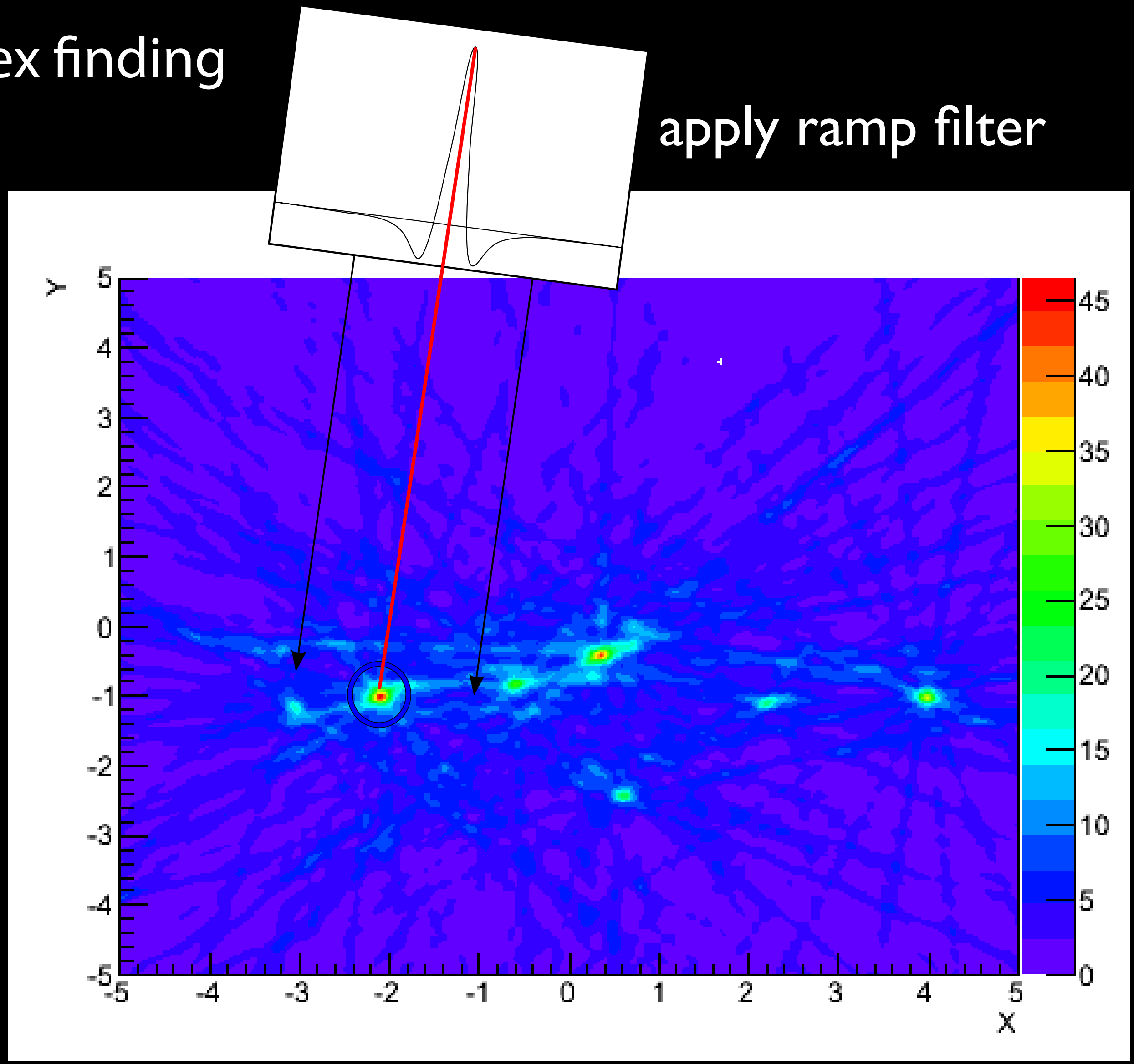
Medical Imaging inspired Vertexing



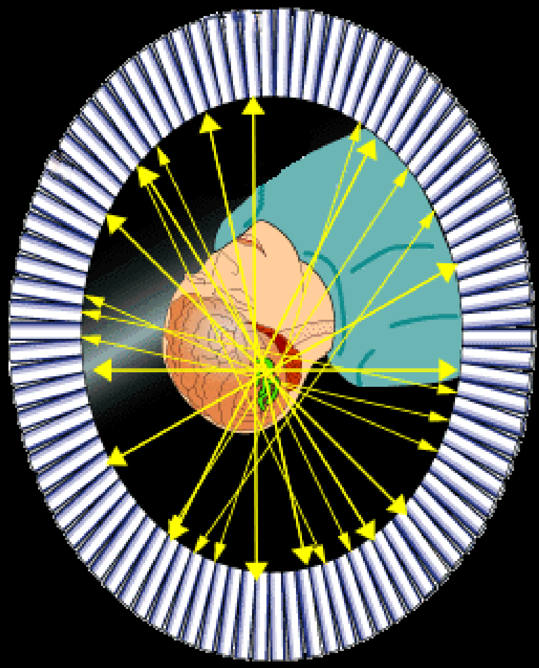
- based on inverse Radon transformation
  - ➡ inspired by imaging techniques used for PET scans
  - ➡ vertex finder is essentially a variant of ZVTOP
    - see: 2012 J. Phys.: Conf. Ser. 396 022021
  - ➡ potentially faster with high pileup
    - evaluated e.g. in ATLAS for primary vertex finding

- steps for vertex finder:
  - ➡ create 3D track density maps
  - ➡ Fourier transform the Gaussian tubes
  - ➡ apply (ramp) k-filter

apply ramp filter



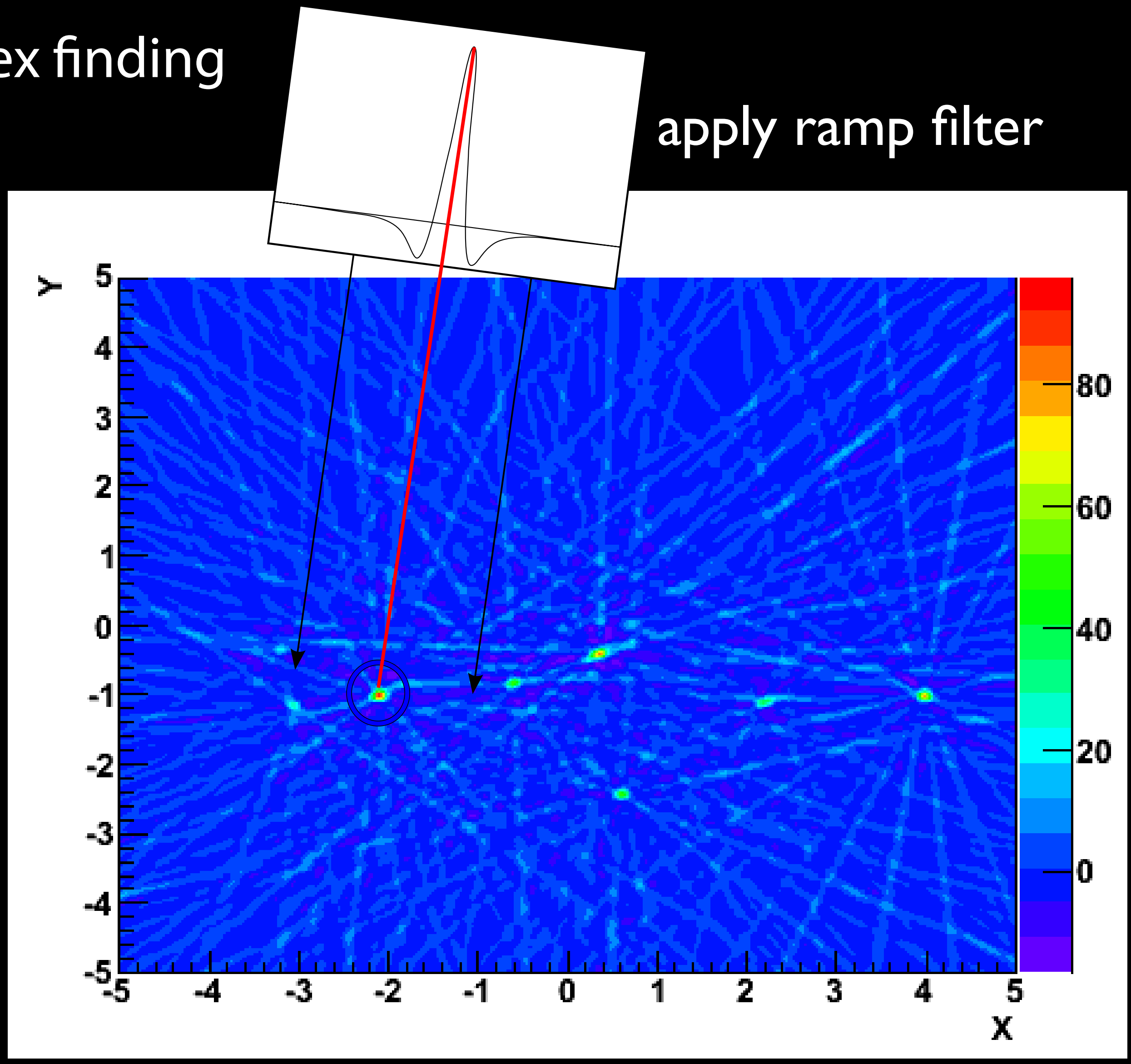
Medical Imaging inspired Vertexing



- based on inverse Radon transformation
  - ➡ inspired by imaging techniques used for PET scans
  - ➡ vertex finder is essentially a variant of ZVTOP
    - see: 2012 J. Phys.: Conf. Ser. 396 022021
  - ➡ potentially faster with high pileup
    - evaluated e.g. in ATLAS for primary vertex finding

- steps for vertex finder:
  - ➡ create 3D track density maps
  - ➡ Fourier transform the Gaussian tubes
  - ➡ apply (ramp) k-filter
  - ➡ back transform image
  - ➡ find vertex candidates as local maxima
  - ➡ run vertex fits on candidates (like ZVTOP)

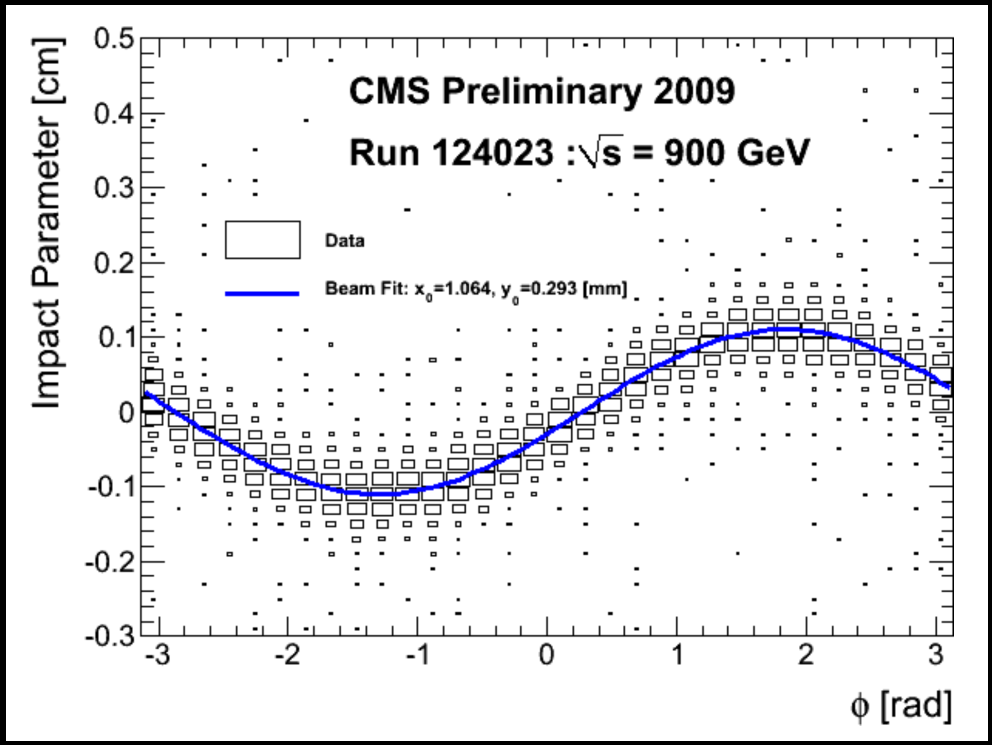
- sharper image, but more noise



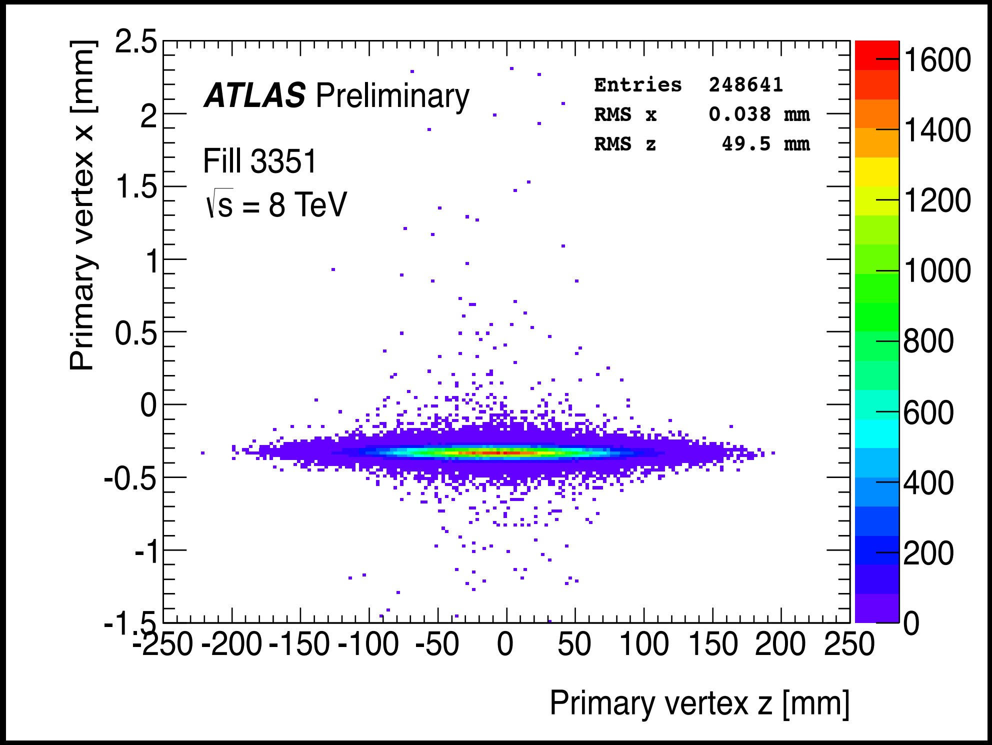
Vertexing Applications

Beam Spot + Primary Vertex

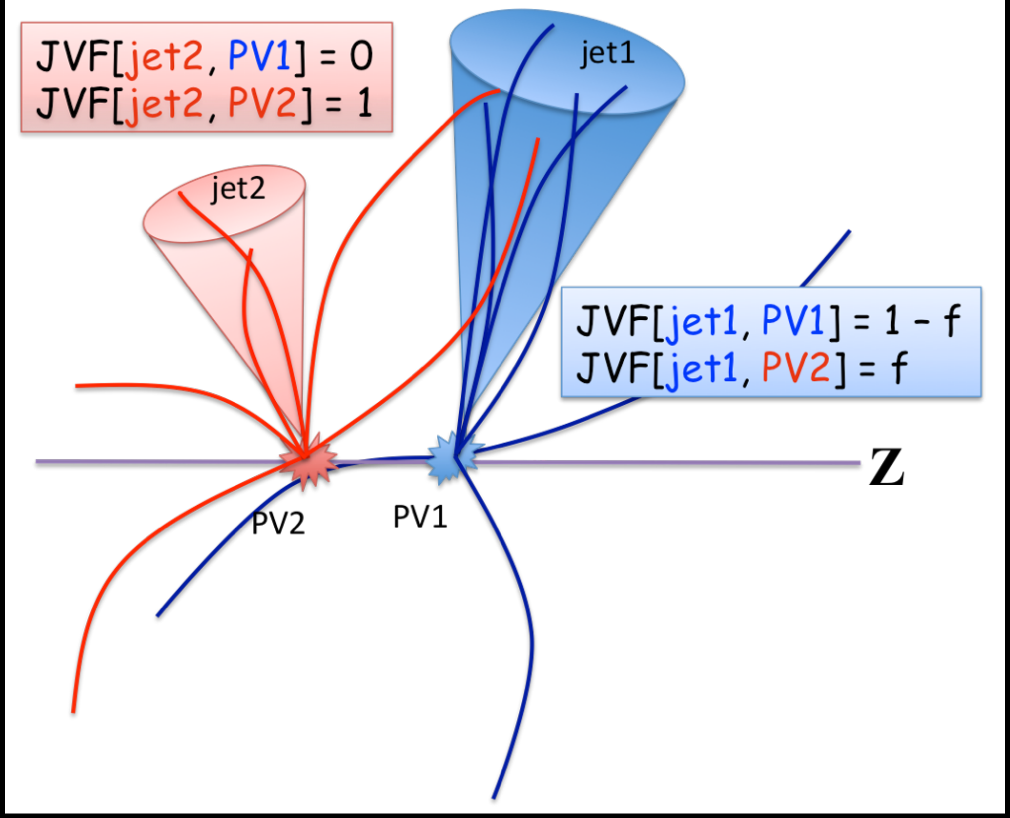
- beam spot routinely determined
  - ➡ either average unbiased primary vertices, or estimate from impact parameter vs ϕ
  - ➡ averaged over short periods of time to follow eventual beam (or detector) movements
  - ➡ input to primary vertex fitting as a constraint



- primary vertex (PV) finding
  - ➡ reconstruct primary and pileup vertices
    - identify primary (hard) interaction vertex, e.g. highest Σp_T^2 of associated tracks
  - ➡ ATLAS (and CMS) use an iterative vertex finder and an adaptive fitter
  - ➡ input to:
    - object selection, e.g. IP of leptons w.r.t. PV
    - pileup corrections to jets and missing E_T
    - luminosity monitoring with PV counting
    - ...



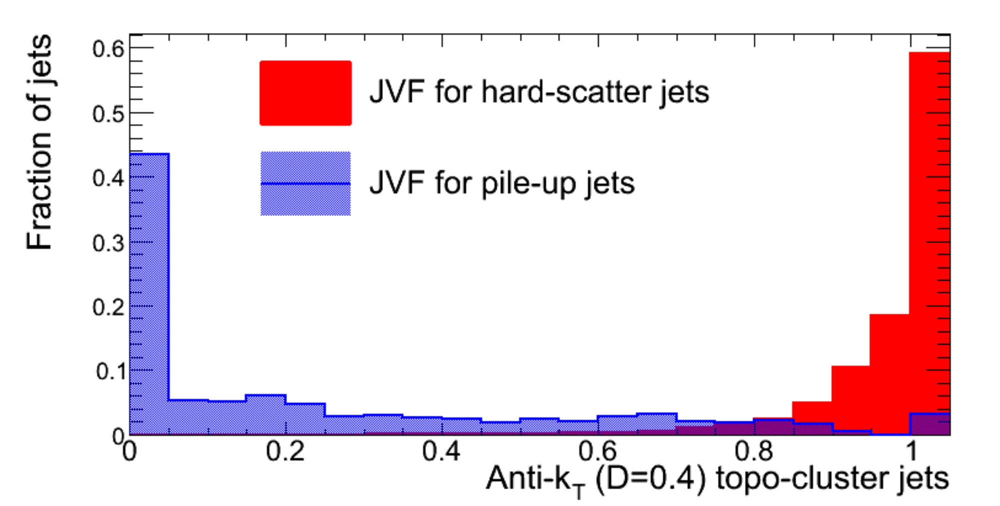
Jet-Vertex-Fraction

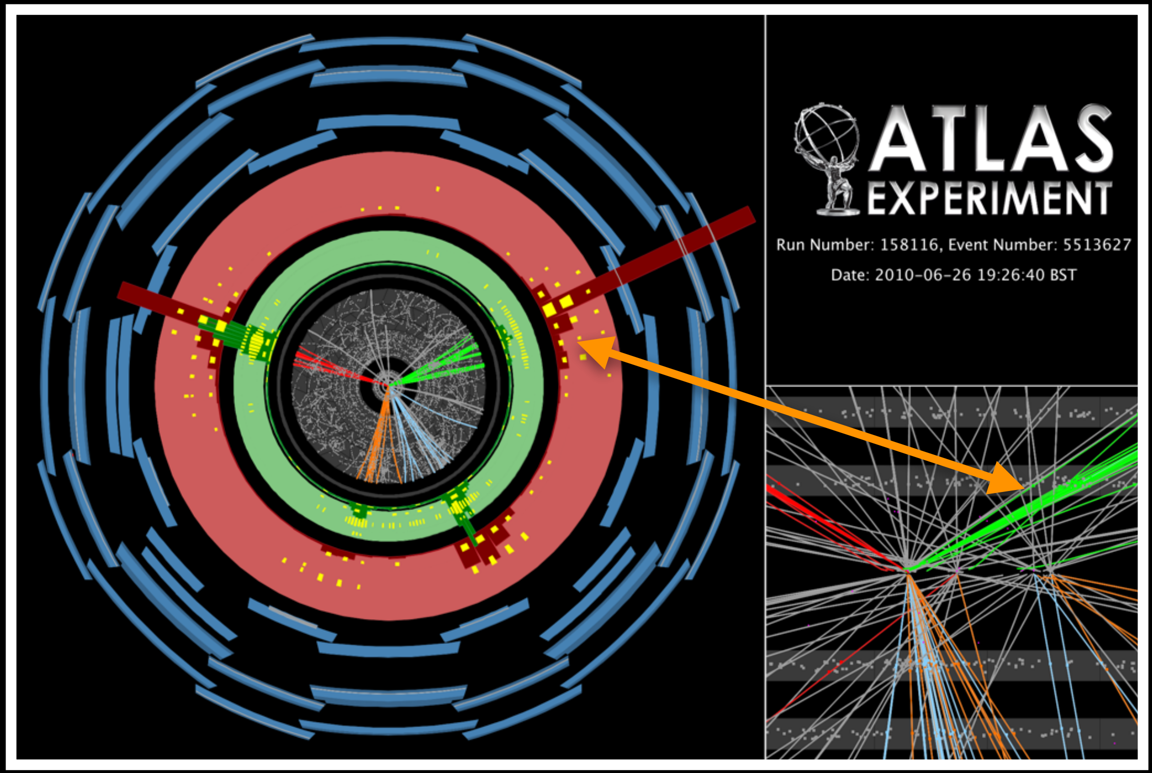


- developed at D0 experiment
  - ➡ separate jets from primary and pileup events
  - ➡ defined fraction of p_T of tracks in jets associated to primary vertex:

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

  - ➡ good separation in D0 for different types of jets



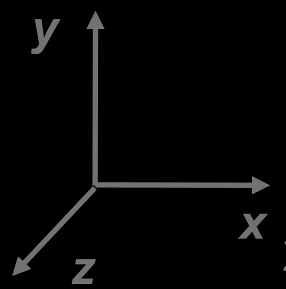


- more complex at LHC
  - ➡ interaction region is a factor ~6 smaller and LHC reached higher levels of pileup
  - ➡ ATLAS replaced JVF with multi-variant techniques, CMS uses combined particle flow

b-Jet Tagging

- several different reconstruction techniques being explored to tag b-(and c-) jets
  - ➡ explore b-(c-) hadron fragmentation, lifetimes, mass and decay properties
  - ➡ a "large industry" to combine the different techniques with multi-variant methods

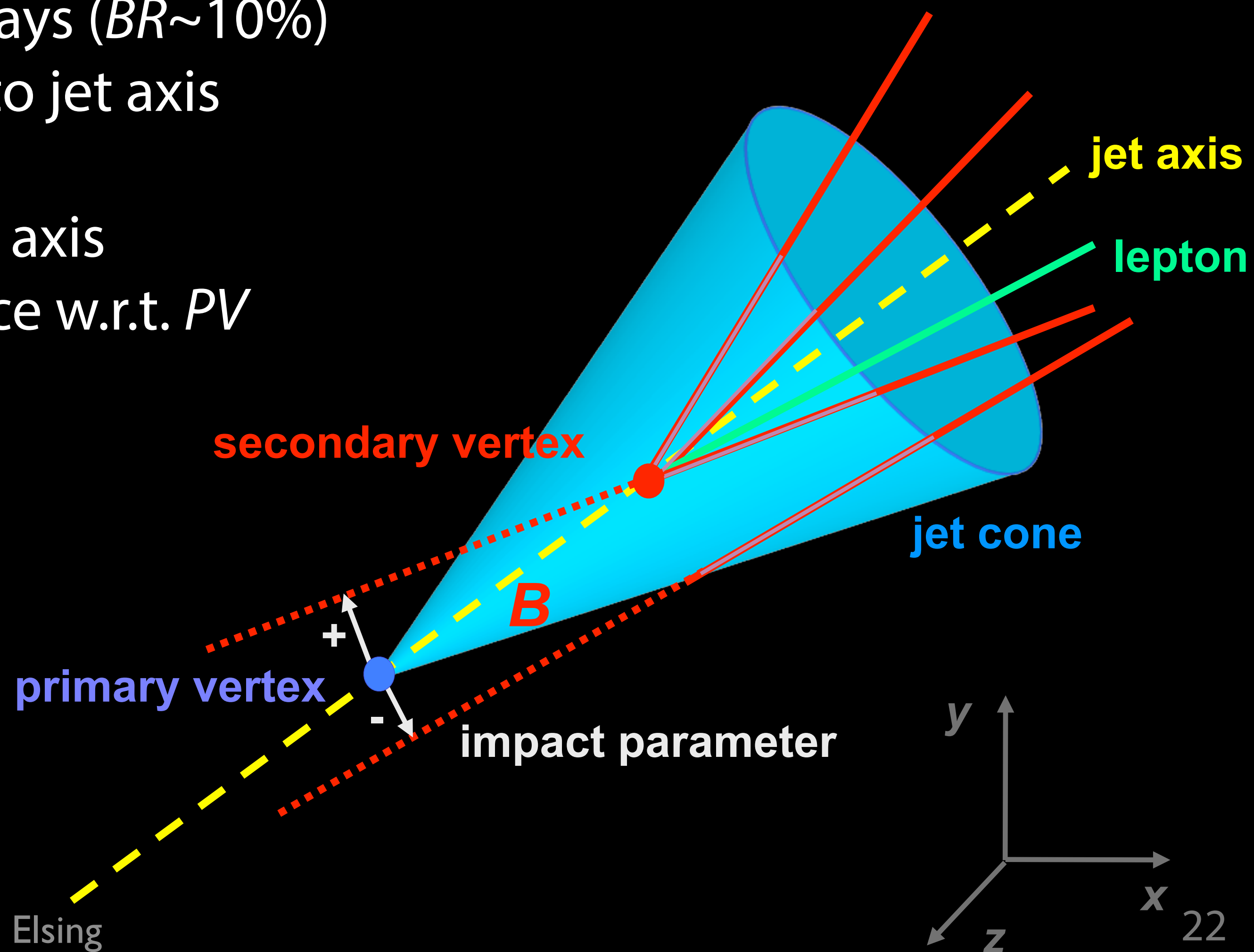
- 3 categories
  - ➡ **soft lepton** tagging
    - explore semi-leptonic b- and c-decays (*BR*~10%)
    - tagging is done using p_T of lepton to jet axis
  - ➡ **impact parameter** tagging
    - sign impact parameter (*IP*) w.r.t. jet axis
    - tagging is done using *IP* significance w.r.t. *PV*
  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



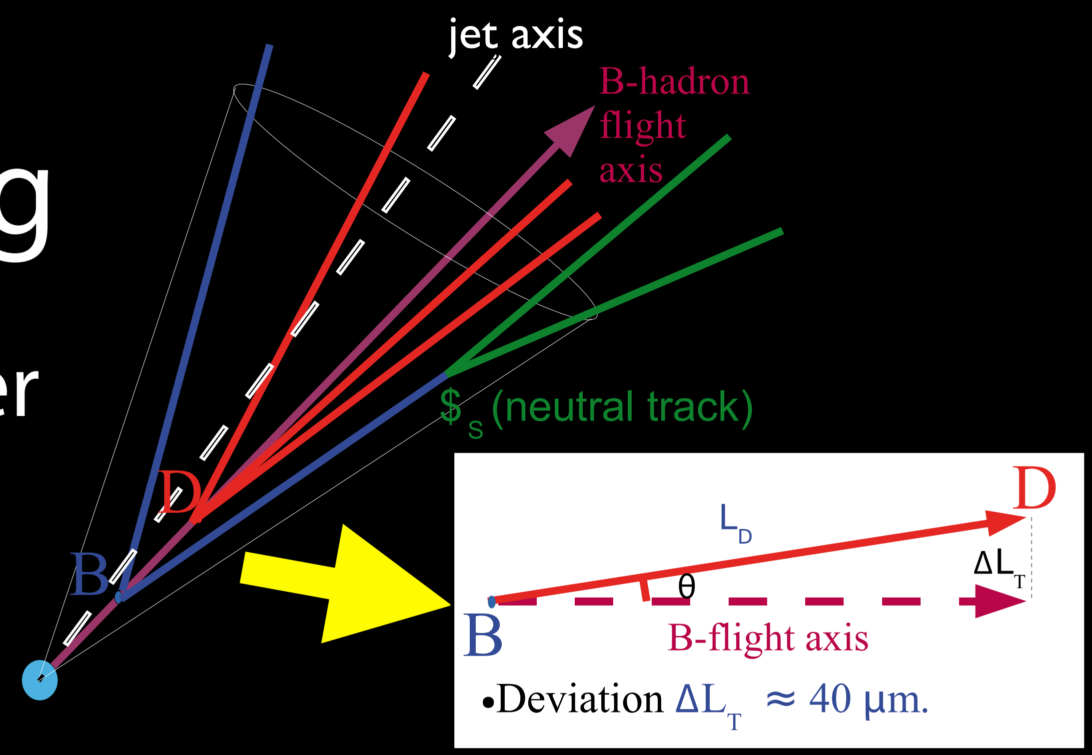
b-Jet Tagging

- several different reconstruction techniques being explored to tag b-(and c-) jets
  - ➡ explore b-(c-) hadron fragmentation, lifetimes, mass and decay properties
  - ➡ a "large industry" to combine the different techniques with multi-variant methods

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

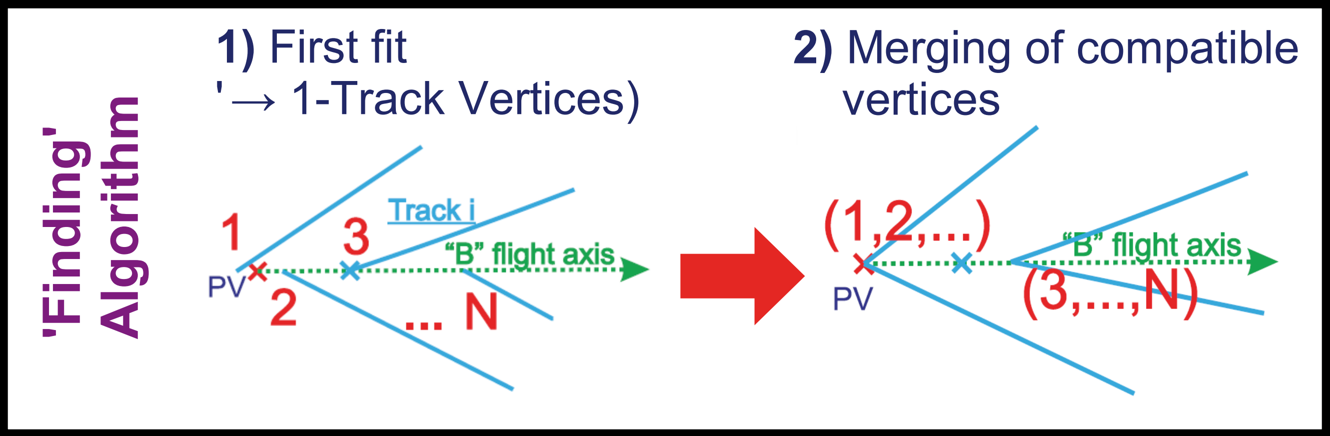


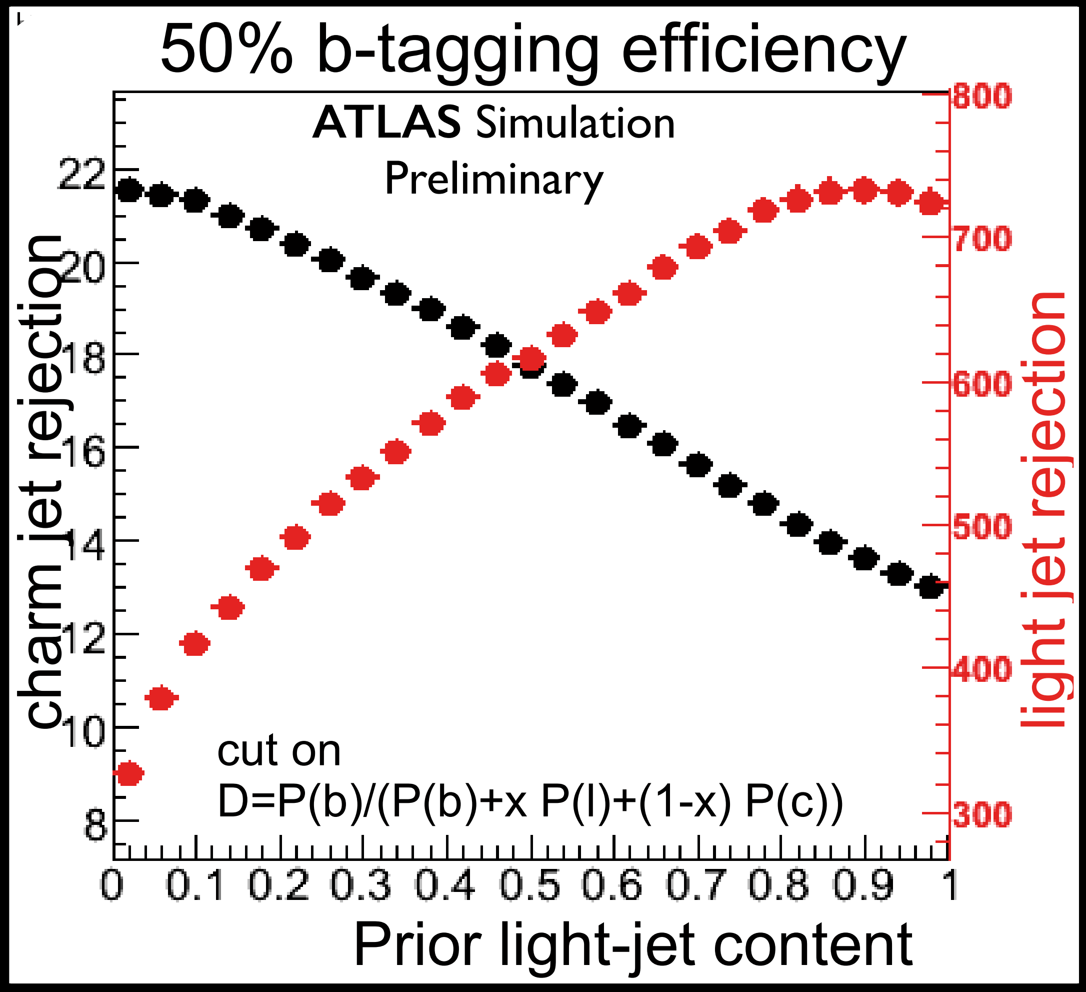
Jet-Fitter and b-Jet Tagging



- conventional secondary vertex tagger
  - ➡ fits all displaced tracks into one common vertex
- Jet-Fitter developed here in Freiburg
  - ➡ fit of 1 to N vertices near the b-jet axis
    - *b*- and *c*-hadron vertices are approximately on the jet axis from primary vertex
  - ➡ mathematical extension of conventional Kalman Filter vertex fitter

G.Piacquadio

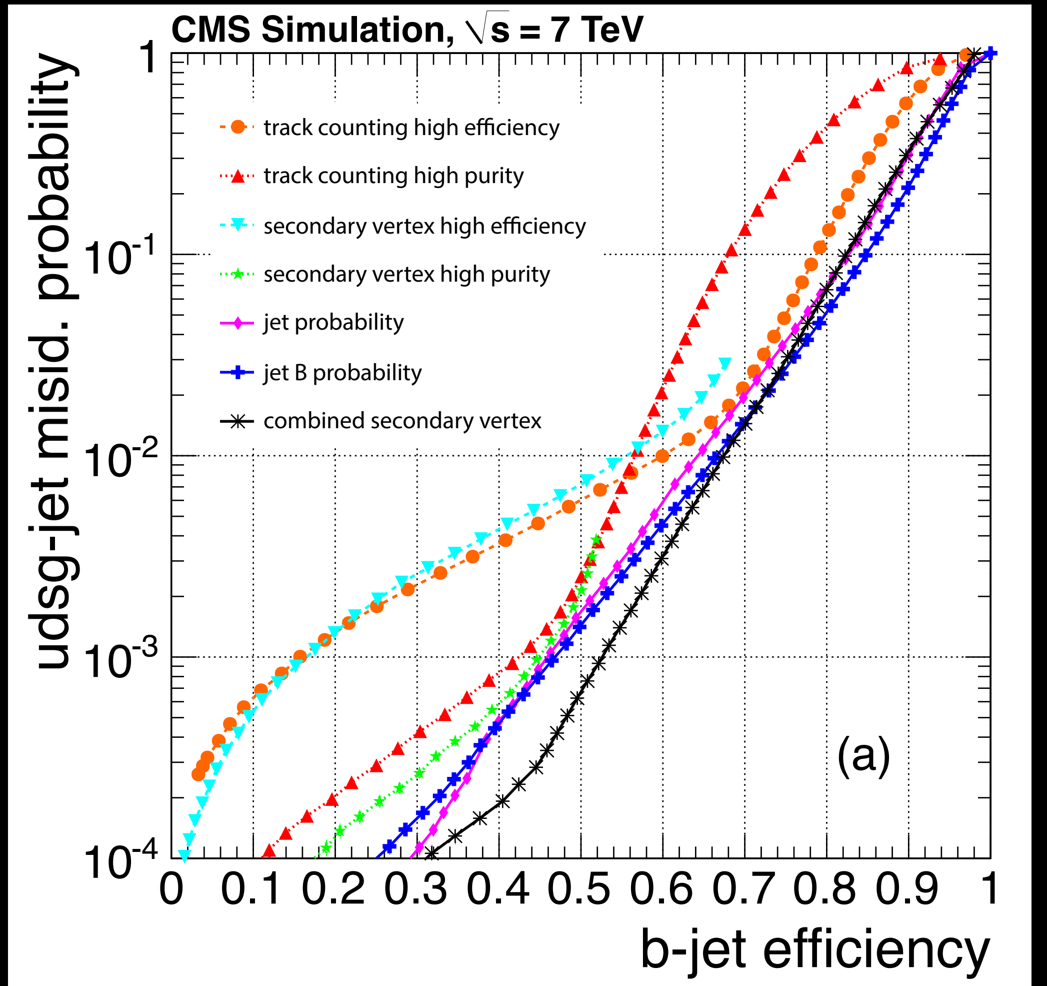


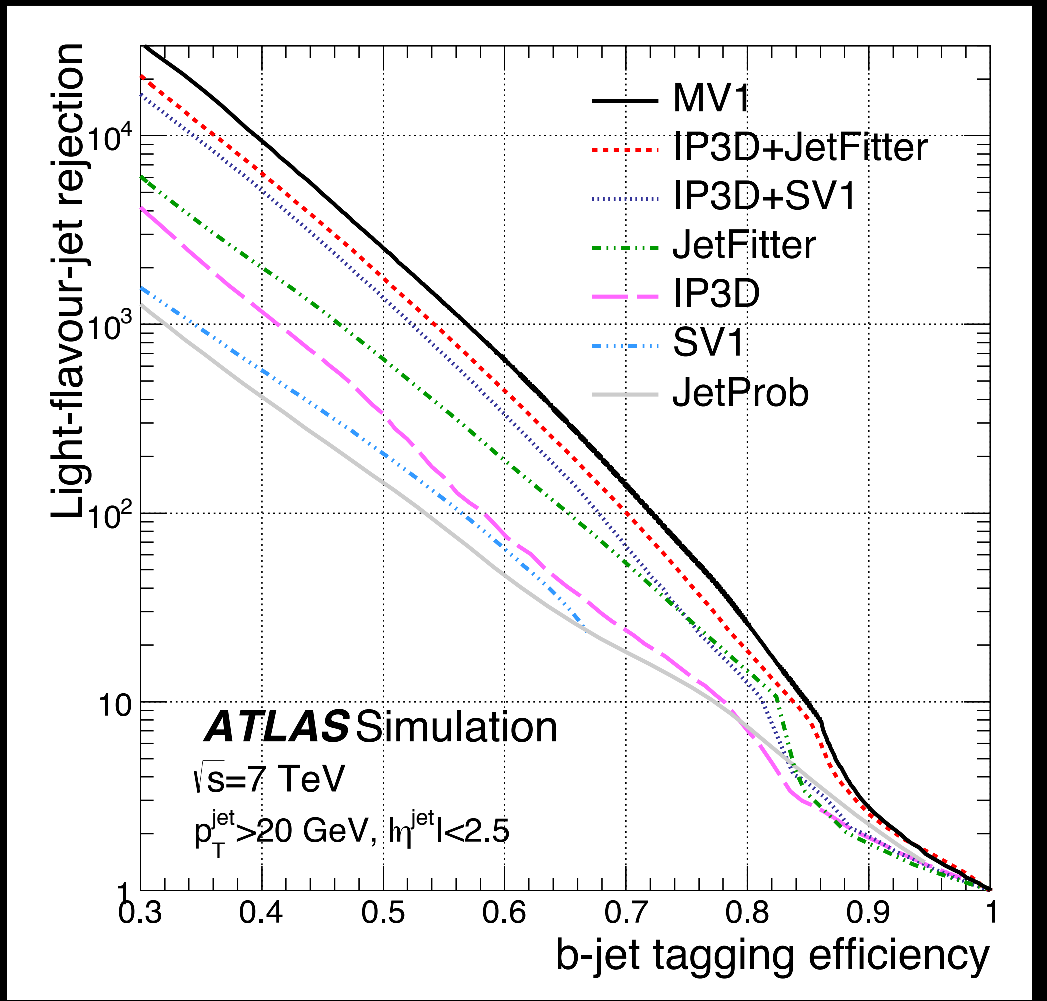


- up to 40% better light rejection
  - ➡ much improved control of charm rejection
  - ➡ IP3D+JetFitter best b-tagging technique in ATLAS

b-Jet Tagging Performance

- ATLAS and CMS use similar techniques
  - ➡ soft lepton tagger
    - explore p_T of leptons to jet axis
    - limited by B/D semi-leptonic branching ratio
  - ➡ track counting of tracks with significant IP offsets
    - robust, but not optimal usage of information
  - ➡ jet probability (JetProb)
    - construct probability that IP significance of all tracks in jet is compatible with PV
  - ➡ variant of JetProb is IP3D
    - likelihood ratio using b/c/light templates
  - ➡ secondary vertex (SV1) tagger
    - high purity, but limited by vertexing efficiency
  - ➡ combined secondary vertex (IP3D+SV1) tagger
    - combined IP significance and secondary vertexing techniques using e.g. likelihood ratios
  - ➡ variant of a combined tagger is IP3D+JetFitter
    - best vertex tagger combined with IP significance
  - ➡ multi-variant techniques to classify jets (e.g. MV1)
    - baseline today, aims at optimal combination of tagging techniques





Markus Elsing

Let’s Summarise...

- discussed vertex fitting and finding techniques
- b-tagging and other examples for vertexing applications

... that's it for this set of lectures !

Thanks !