what is it?

- “Decay Tree Fitter” (a.k.a. “DTF”)
  - least squares algorithm
  - extracts four-momenta, decay times, vertex positions from a decay chain

- developed for BaBar: now in use in LHCb, Panda, Belle-2, …

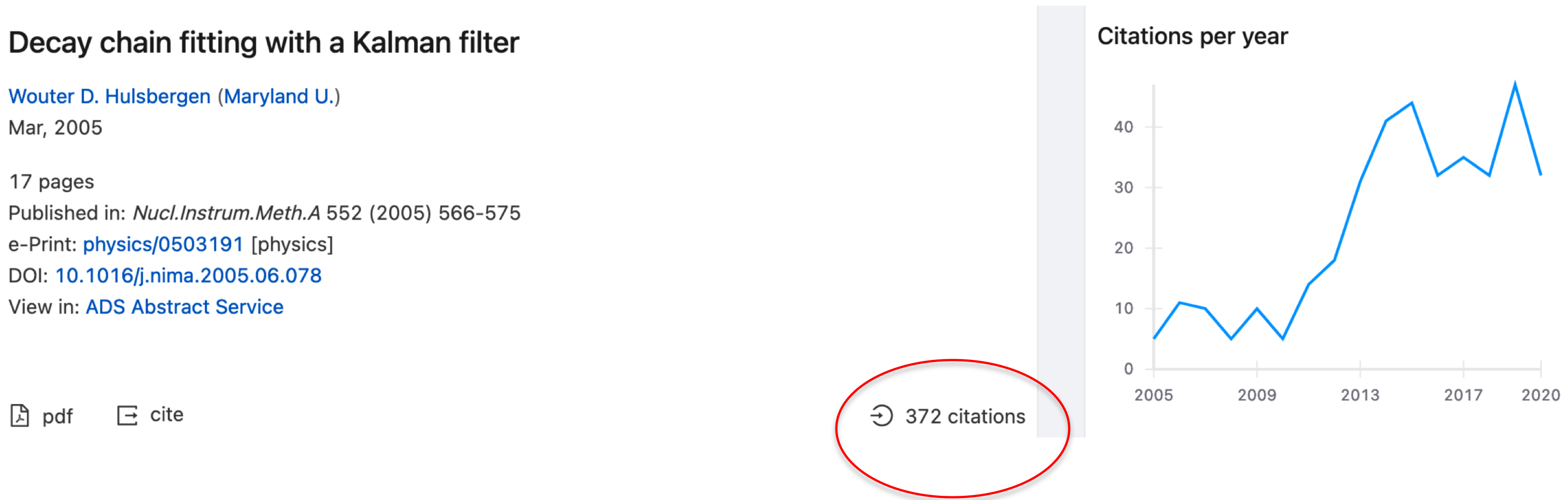
- original implementation is in C++
  - the LHCb code is basically just a fork of the BaBar code
  - Belle-2 code is independent (but inspired by the LHCb/BaBar code)

- code is not in a fantastic state:
  - >16 years old, lot’s of dynamic allocation, still uses CLHEP!
  - happy to share it, but one could also start from scratch

write-ups (by no means a bibliography!)

- decay tree fitter paper: https://inspirehep.net/literature/679286
- recent pedagogical talk on vertex fitting in LHCb: https://surfdrive.surf.nl/files/index.php/s/oCOqV59WOXpByob
- lectures on track and vertex fitting: https://www.nikhef.nl/~wouterh/topicallectures/TrackingAndVertexing
- vertex fit in alignment, including LHCb-specific write-up of Billoir-Frühwirth-Regler algorithm: https://inspirehep.net/literature/1123118
- some incomplete notes on fast vertex fitting for LHCb: https://gitlab.cern.ch/wouter/efficient-vertexing-for-lhcb/

- it is worth ‘publishing’ a vertex algorithm:



Decay chain fitting with a Kalman filter

Wouter D. Hulsbergen (Maryland U.)
Mar, 2005

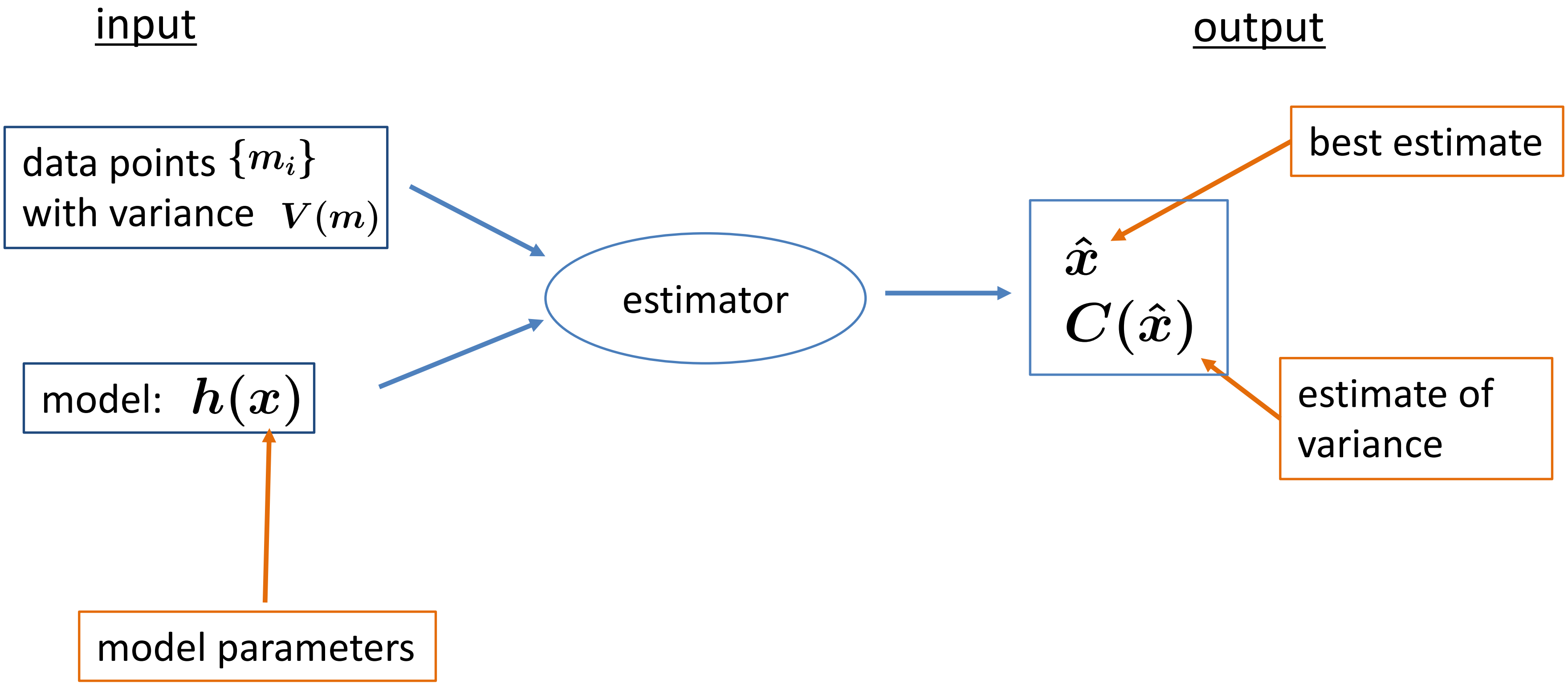
17 pages
Published in: *Nucl.Instrum.Meth.A* 552 (2005) 566-575
e-Print: physics/0503191 [physics]
DOI: 10.1016/j.nima.2005.06.078
View in: ADS Abstract Service

- but don’t be impressed: I think that 371 out of 372 could count as “self-citations” ☺

outline

- description of least squares fitting, to introduce the concepts
- motivation for a ‘decay chain fit’
- how to parametrize a decay chain (two options)
- how to minimize the chi2
- very brief: what it would take to implement this in FCC-ee (or a more generic software framework)

Notation



input

data points $\{m_i\}$ with variance $V(m)$

model: $h(x)$

model parameters

estimator

output

$\hat{x}$

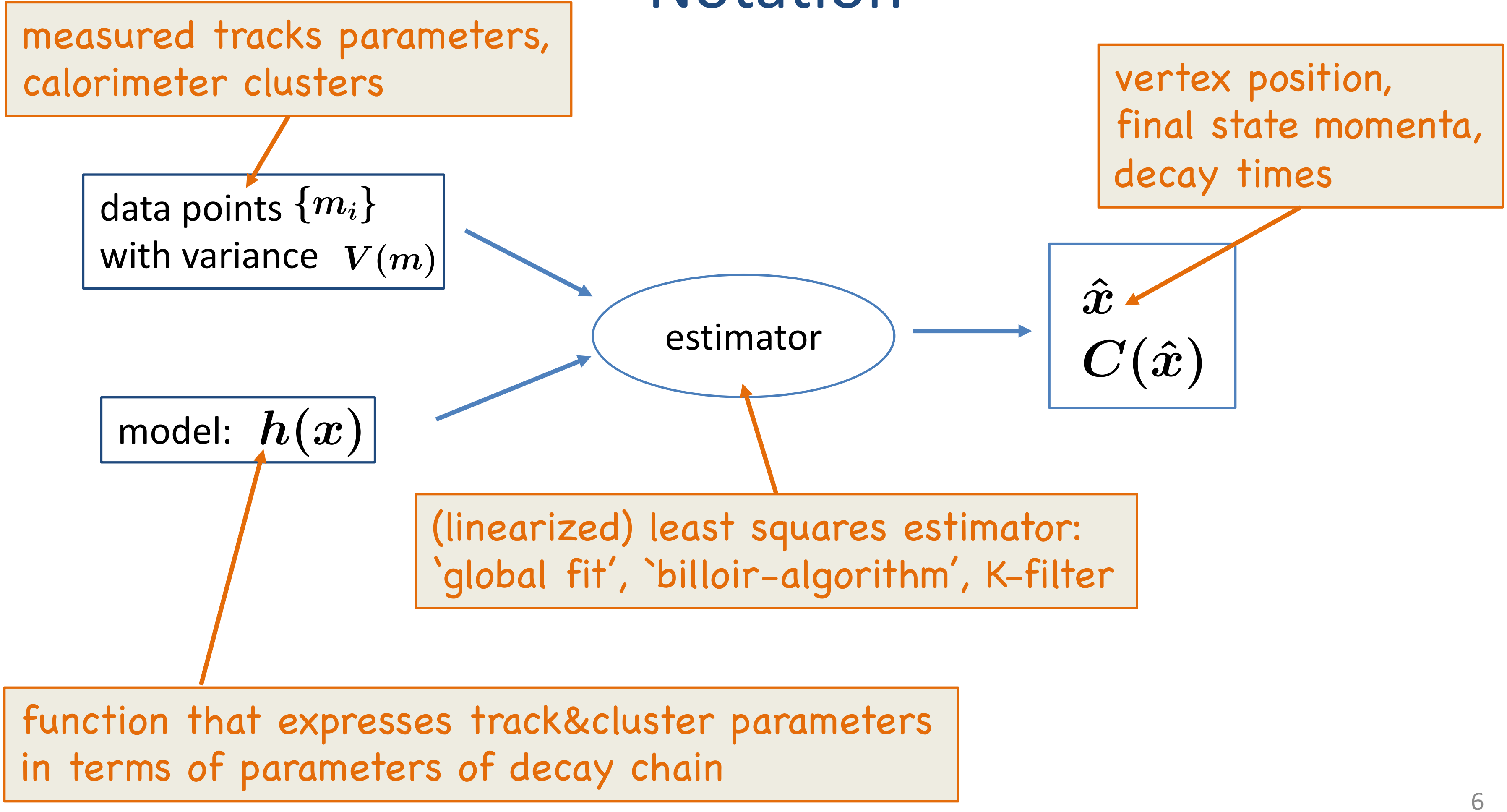
$C(\hat{x})$

best estimate

estimate of variance

(Notation close to Frühwirth ’87, https://inspirehep.net/literature/259509)

Notation



measured tracks parameters, calorimeter clusters

data points $\{m_i\}$
with variance $V(m)$

model: $h(x)$

estimator

$\hat{x}$
$C(\hat{x})$

vertex position,
final state momenta,
decay times

(linearized) least squares estimator:
`global fit', `billoir-algorithm', K-filter

function that expresses track&cluster parameters in terms of parameters of decay chain

Method of least squares

input measurements

model: ‘prediction’

$$\boxed{\chi^2 = \sum_i \left(\frac{m_i - h_i(x)}{\sigma_i} \right)^2}$$

“estimated RMS of parent distribution of error of m_i”

Least-squares-estimator: value of x for which χ^2 is minimal $\boxed{\left. \frac{\mathrm{d}\chi^2}{\mathrm{d}x} \right|_{\hat{x}} = 0}$

Matrix notation

vector of residuals

$$\chi^2 = (\boldsymbol{m} - \boldsymbol{h}(\boldsymbol{x}))^T \; \boldsymbol{V}^{-1} \; (\boldsymbol{m} - \boldsymbol{h}(\boldsymbol{x}))$$

measurement covariance matrix.
often block-diagonal

Linear least squares estimator

- consider a **linear model**

$$\boxed{\boldsymbol{h(x) = h_0 + H\,x}}$$

- least squares condition:

$$\boxed{\frac{\mathrm{d}\chi^2}{\mathrm{d}\boldsymbol{x}} = -2\,\boldsymbol{H}^T\boldsymbol{V}^{-1}\,(\boldsymbol{m} - \boldsymbol{h}_0 - \boldsymbol{H}\boldsymbol{x}) = 0}$$

- solution:

$$\boxed{\hat{\boldsymbol{x}} = \left(\boldsymbol{H}^T\boldsymbol{V}^{-1}\boldsymbol{H}\right)^{-1}\,\boldsymbol{H}^T\boldsymbol{V}^{-1}\,(\boldsymbol{m} - \boldsymbol{h}_0)}$$

$$\boxed{\mathrm{var}\,(\hat{\boldsymbol{x}}) \equiv \boldsymbol{C} = \left(\boldsymbol{H}^T\boldsymbol{V}^{-1}\boldsymbol{H}\right)^{-1}}$$

"Linear Least squares Estimator (LSE)"

Linear least squares estimator

- closer inspection:

$$\boxed{\underset{\text{M-dim vector}}{\hat{x}} = \underbrace{\left(H^T V^{-1} H\right)^{-1}}_{\substack{\frac{1}{2}\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2}\Big|_{x=0} \\ \text{MxM symmetric matrix}}} \; \underbrace{H^T V^{-1}\left(m - h_0\right)}_{\substack{\frac{1}{2}\frac{\mathrm{d}\chi^2}{\mathrm{d}x}\Big|_{x=0} \\ \text{M-dim vector}}}}$$

- inversion expensive for problems with many parameters → alternatives
  - *Kalman filter*: useful when input measurements uncorrelated
  - problem-specific solutions that *exploit emptiness* of MxM matrix

Non-linear models: Newton-Raphson

1. expand around initial solution

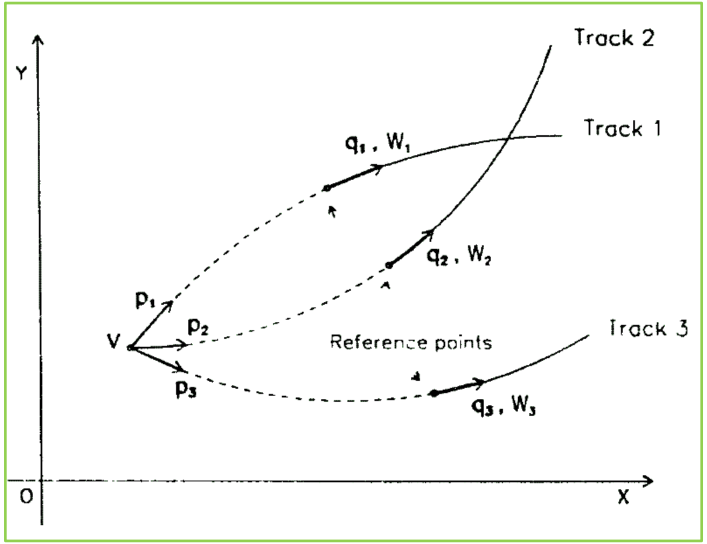
$$\boxed{h(x) \;=\; h(x_0) + H(x - x_0)} \qquad\qquad \boxed{H \;=\; \left.\frac{\mathrm{d}h(x)}{\mathrm{d}x}\right|_{x_0}}$$

2. compute a new value for x

$$\boxed{\hat{x} \;=\; x_0 + \underbrace{\left(H^T V^{-1} H\right)}_{\boxed{\left.\frac{1}{2}\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2}\right|_{x_0}}}{}^{-1} \; \underbrace{H^T V^{-1}\left(m - h(x_0)\right)}_{\boxed{\left.\frac{1}{2}\frac{\mathrm{d}\chi^2}{\mathrm{d}x}\right|_{x_0}}}}$$

3. use x-hat as new expansion point and iterate until $\Delta\chi^2$ is small

the ‘ordinary’ vertex fit



input: **N track parameter vectors** with covariance matrix

model: **1 vertex + N momentum vectors**

Vertex fit minimizes the total chi2:

$$\chi^2 = \sum_{\text{tracks } i} (q_i - h(x, p_i))^T V_i^{-1} (q_i - h(x, p_i))$$

measured track parameters

vertex position

momentum vector

track cov matrix

minimizing chi2 in vertex fits

- N-prong vertex has **M = 3 + 3N** parameters
  - naïve LSE requires inversion of MxM symmetric matrix
  - expensive if number of outgoing tracks large, e.g. PV fit

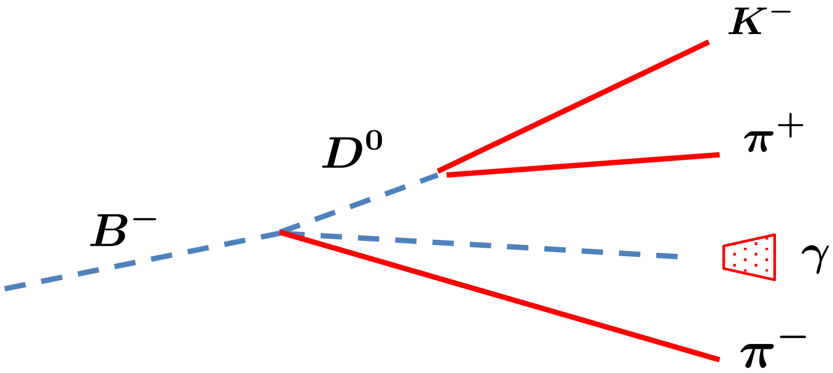
- two 'fast' methods (*very closely related*)
  - Billoir-Frühwirth-Regler '85: exploit (empty) structure of $H^T V^{-1} H$
  - Kalman filter (e.g. Fruhwirth '87)
- final state particle momentum vectors (and cov. matrix) *can* be calculated, but can also be omitted (which saves time, e.g. in PV fit)

- since ***measurement-model not linear***, need **iterations**
- expressions not very illuminating, so we'll skip them: see references

reconstructing a decay chain

- consider a multi-level decay chain

$$\begin{aligned} B^- &\to D^{*0}\pi^- \\ &\quad\hookrightarrow D^0\gamma \\ &\qquad\quad\hookrightarrow K^-\pi^+ \end{aligned}$$

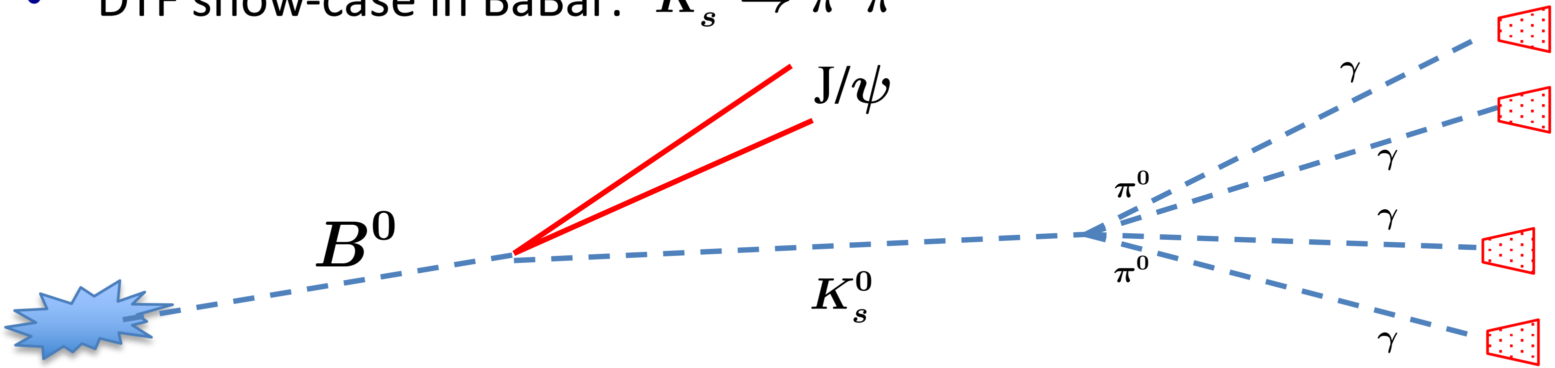


- traditional method: “leaf-by-leaf fitting”
  - fit most downstream vertices first
  - use composites as input to next upstream level
  - very natural way to reconstruct and select cascade decays

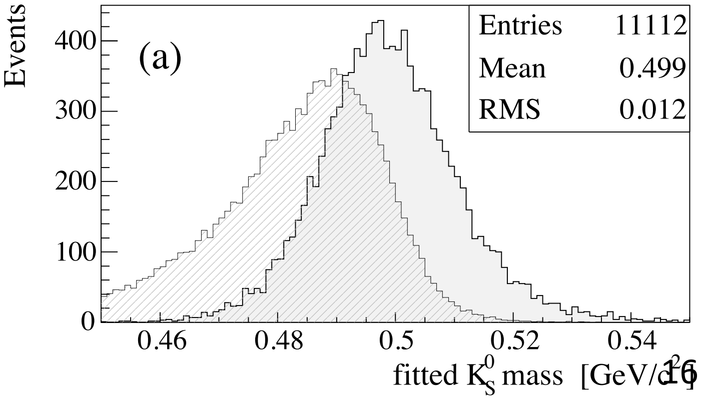
- to implement this, need to extend track-based vertex fit with constraints for
  - photons, merged pi0 (calorimeter clusters)
  - short-lived composites (e.g. D*, J/psi)
  - long-lived composites (e.g. Ks, D0, B+)

motivation for ‘global decay chain fit’

- sometimes the ‘mother’ is needed to constrain the downstream vertex
- DTF show-case in BaBar: $K_s^0 \to \pi^0\pi^0$

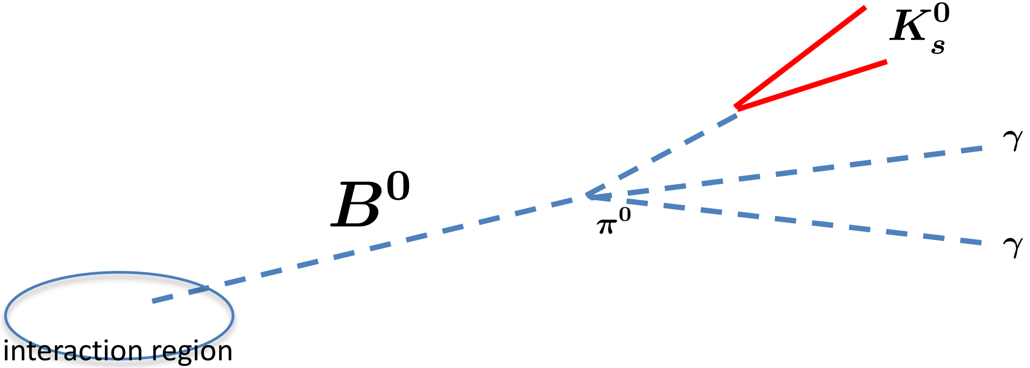


- cannot compute π^0 vertex position without constraint from mother
- to measure K_s decay length or invariant mass, also need mass constraint for at least one π^0



motivation for ‘global decay chain fit’

- another Babar show-case: time-dependent analysis of $B^0 \to K_s^0\pi^0$



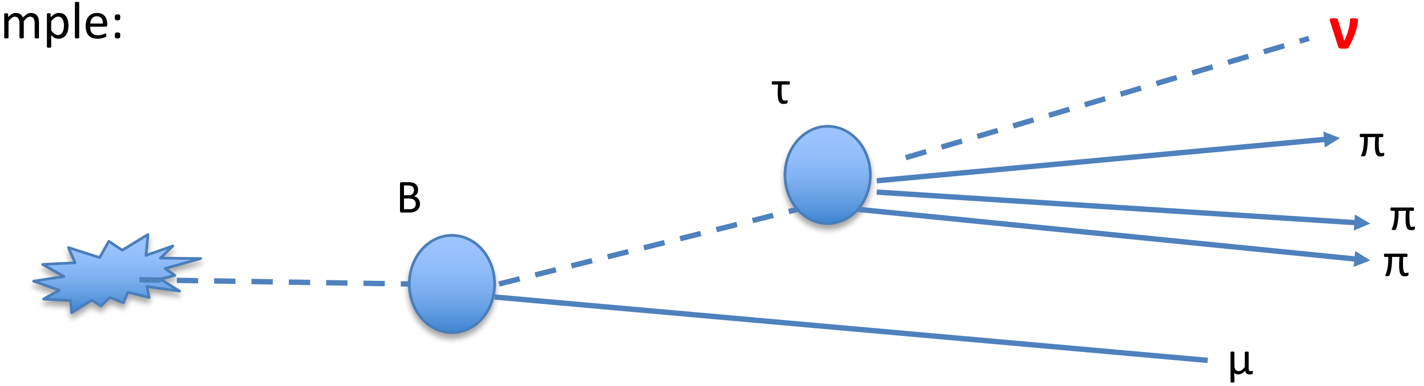
Jawahery,
Farbin,
Hulsbergen 2003

- only one ‘trajectory’ from B vertex (the K-short)
- use known average beam-spot location to constrain B origin

- this decay was main physics motivation to develop DTF, though admittedly it was not used for the first analysis, or in Belle

motivation for `global decay chain fit': missing particles

- in theory, DTF can also be used to fit decays with a missing particle
  - exploit mass or vertex constraints to over-constrain the problem
- example:



- in practice, for such problems you'd like a bit of flexibility in the choice of parametrization for the neutrino
- myself not fully convinced that the decay tree fits is more useful than 'quasi-analytical' formulas, but it does give access to a chi2 and cov matrix

Decay Tree Fitter

- Decay Tree Fitter is a ‘global decay chain fit’
- input
  - a hypothesis for the decay chain
  - tracks for charged particles, and ECAL clusters for photons, K-long
  - eventually: ‘origin’ constraint (PV), mass constraints, …
- output
  - four momenta of all particles in the tree
  - vertex positions
  - decay times
  - full covariance matrix
  - total chi-square

Some use-cases in LHCb

- ***vertex-constrained and mass-constrained final state momenta*** to compute decay angles (J/psi phi, $K^*\ell\ell$)

- ***improved mass resolution using PV constraint***: D*->D π_{soft}

- ***improved mother mass resolution using daughter mass constraint***, e.g. in B->J/ψX, or decays with pi0

- ***improve daughter mass resolution using B mass constraint***, e.g. q^2 in $K^*\ell\ell$

- estimate particle momentum in calibration channels with one final track without momentum estimate

- …

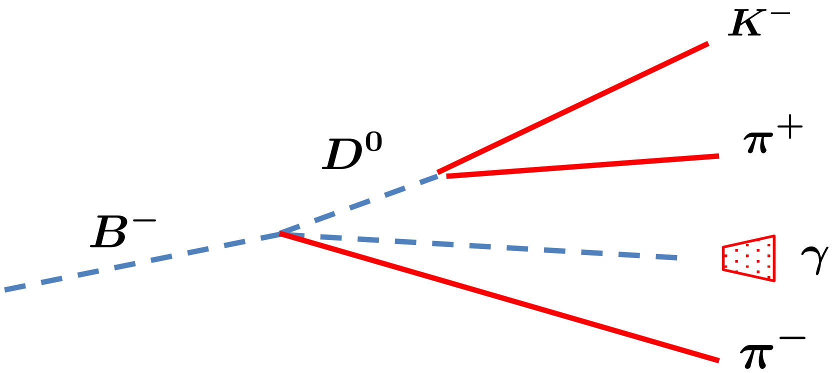
Decay chain fitting challenges

- two main challenges
  1. how to parametrize the decay chain
  2. how to minimize chi2 efficiently

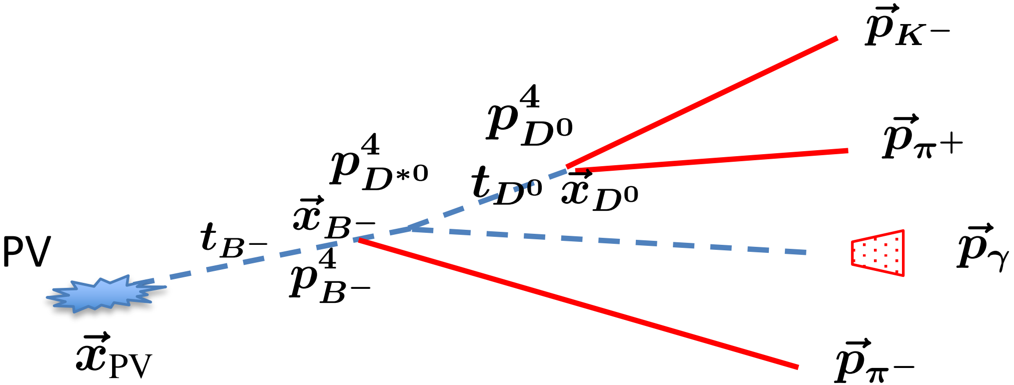
- the original paper discusses these issues,
  plus algebraic expressions for the measurement model,
  plus a computationally little more stable expression for the Kalman filter

- it also discusses how to ‘order’ constraints in the filter:
  this is *obsolete*:
  the LHCb implementation uses an extended K-filter *with ‘**reference**’*

parametrizing a decay chain



parametrizing a decay tree



- lot’s of parameters (momenta, vertex positions, decay times): I count 35!
- but … many are redundant due to ‘physical constraints’
  - four-vector conservation at each vertex: $\boxed{p^4_{\mathrm{mother}} = \sum_{\mathrm{daughters}\ i} p^4_i}$
  - geometrical ‘vertex’ constraints: $\boxed{\vec{x}_{\mathrm{decay}} = \vec{x}_{\mathrm{production}} + \vec{p}\,t\,/\,m}$
- if you use these, there are 17 parameters left

parametrizing a decay tree

1. approach 1: minimal number of parameters (17 in the example)
    - position of head of tree
    - decay times (or decay lengths) for long-lived composites
    - final state momentum vectors

2. approach 2: all parameters end-user may be interested in (35 in the example)
    - minimal + all intermediate vertex positions and four-momenta
    - remove ‘redundant’ parameters by adding physical constraints as Lagrange constraints in chi2 minimization

- DTF uses approach 2
    - pro: simplifies implementation considerably
    - con: expensive, especially for `global minimum chi2 fit’

minimization the chi-square: Kalman filter

- many parameters → large covariance matrix → expensive inversion

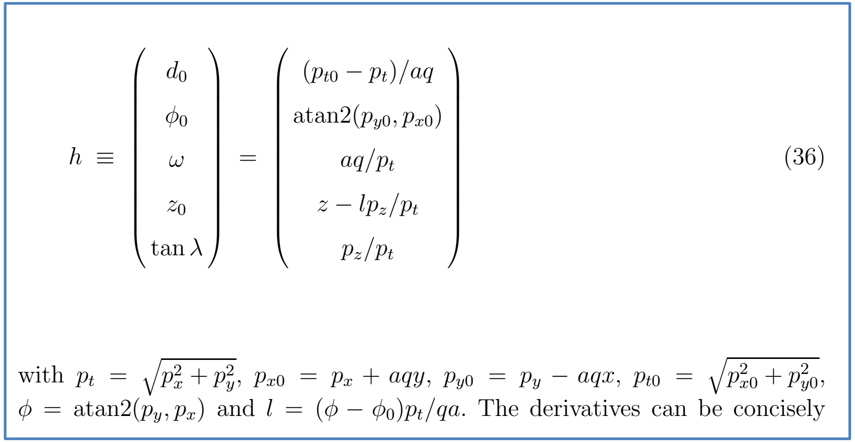
- with chosen parametrization, many redundant parameters
  - these are removed with ‘physical constraints’
  - in a global LSE, this is implemented with Lagrange multiplier
    → one extra parameter for every constraint
    → that makes it even more expensive

- in 2003 chose an (extended) **Kalman filter** because I expected it would win
  - no inversion of large matrix
  - no extra parameters for Lagrange constraints

- that said, now have working (but not entirely complete) version of DTF that uses the ‘minimal’ parametrization and the global LSE, and is significantly faster: https://gitlab.cern.ch/wouter/DecayTreeFitterTwo)

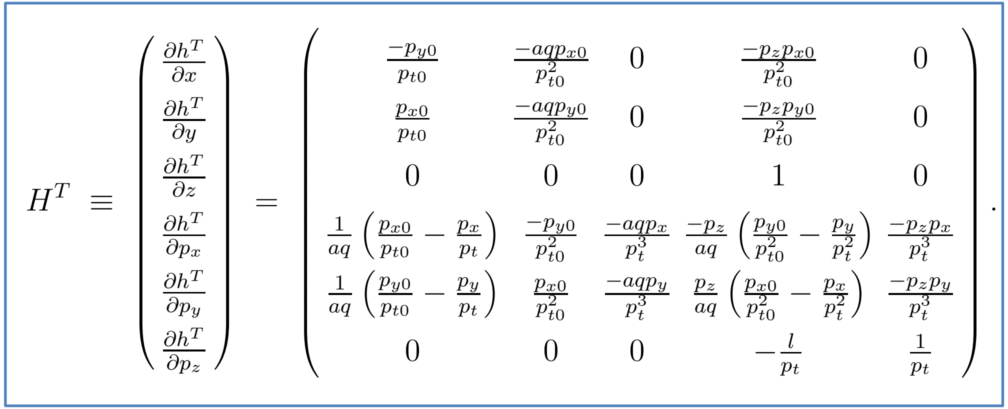
measurement model: not rocket science, but not trivial either

- minimal implementation needs ‘measurement’ models for
  - tracks
  - photon cluster
- example: track in helix parametrization, in most compact notation



$$h \equiv \begin{pmatrix} d_0 \\ \phi_0 \\ \omega \\ z_0 \\ \tan\lambda \end{pmatrix} = \begin{pmatrix} (p_{t0} - p_t)/aq \\ \mathrm{atan2}(p_{y0}, p_{x0}) \\ aq/p_t \\ z - l p_z/p_t \\ p_z/p_t \end{pmatrix} \tag{36}$$

with $p_t = \sqrt{p_x^2 + p_y^2}$, $p_{x0} = p_x + aqy$, $p_{y0} = p_y - aqx$, $p_{t0} = \sqrt{p_{x0}^2 + p_{y0}^2}$, $\phi = \mathrm{atan2}(p_y, p_x)$ and $l = (\phi - \phi_0)p_t/qa$. The derivatives can be concisely



$$H^T \equiv \begin{pmatrix} \frac{\partial h^T}{\partial x} \\ \frac{\partial h^T}{\partial y} \\ \frac{\partial h^T}{\partial z} \\ \frac{\partial h^T}{\partial p_x} \\ \frac{\partial h^T}{\partial p_y} \\ \frac{\partial h^T}{\partial p_z} \end{pmatrix} = \begin{pmatrix} \frac{-p_{y0}}{p_{t0}} & \frac{-aqp_{x0}}{p_{t0}^2} & 0 & \frac{-p_z p_{x0}}{p_{t0}^2} & 0 \\ \frac{p_{x0}}{p_{t0}} & \frac{-aqp_{y0}}{p_{t0}^2} & 0 & \frac{-p_z p_{y0}}{p_{t0}^2} & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \frac{1}{aq}\left(\frac{p_{x0}}{p_{t0}} - \frac{p_x}{p_t}\right) & \frac{-p_{y0}}{p_{t0}^2} & \frac{-aqp_x}{p_t^3} & \frac{-p_z}{aq}\left(\frac{p_{y0}}{p_{t0}^2} - \frac{p_y}{p_t^2}\right) & \frac{-p_z p_x}{p_t^3} \\ \frac{1}{aq}\left(\frac{p_{y0}}{p_{t0}} - \frac{p_y}{p_t}\right) & \frac{p_{x0}}{p_{t0}^2} & \frac{-aqp_y}{p_t^3} & \frac{p_z}{aq}\left(\frac{p_{x0}}{p_{t0}^2} - \frac{p_x}{p_t^2}\right) & \frac{-p_z p_y}{p_t^3} \\ 0 & 0 & 0 & -\frac{l}{p_t} & \frac{1}{p_t} \end{pmatrix}.$$

- for charged particles may also need (external) tools to propagate through non-homogenous field, including jacobian

decay tree fitter for LHC-ee?

- it seems useful for the flavour physics program
  - LHCb/B-factory physicists will certainly appreciate it

- migration from BaBar to LHCb was very straightforward
  - adapt to different implementation of ‘particle’
  - adapt to different track/cluster models

- these things are well isolated in the code, so, it should be reasonably easy to do this for FCC-ee

- that said … the core needs real work too, for instance:
  - CLHEP → Eigen?
  - virtual inheritance & dynamic allocation → templates, variants, …
  - remove historical parts, like obsolete ordering of constraints

why ***not*** use a global decay chain fit?

- good old leaf-by-leaf fit is good enough for 99% of vertex problems!
- in LHCb DTF is mostly popular because Vanya Belyaev wrote great interface
- high price: because the algorithm computes a large covariance matrix, including momenta of final state particles, it is extremely slow
  - example: in LHCb even something as simple as a 2-track vertex fit is >20x slower with DTF than with the traditional Billoir fit
- message: use global decay chain fits sparsely
  - make sure that you also have a traditional 'single' vertex fit e.g. for use in selections with a lot of combinatorics
  - use it at final stage of selection/analysis

Conclusion

- Decay Tree Fitter is an implementation of a global decay chain fit

- used in several flavour physics experiments

- code base is C++
  - not experiment independent, but perhaps reasonably easy to adapt
  - may be a good student/postdoc project: a few months should be more than sufficient to (re)implement it

BACKUP

what is a mass constraint?

- fixes the mass of a ‘composite’ particle in the decay tree by adding additional term to chi-square for “$m - m_{pdg}$”

- will affect both momenta and vertex positions

- effectively ‘decorrelates’ invariant mass measurements in the tree
  - compute M1
  - compute M2 with mass constraint in M1
  - M1 and M2 will now be uncorrelated
  - M2 resolution is better than without M1 constraint

what is a Primary Vertex (PV) constraint?

- adds a term to the chi-square that fixes the **production vertex of the head** of the decay tree to the primary vertex
- needed when computing a decay time (though not necessarily as part of the ‘vertex fit’)
- sometimes helps to improve mass resolution, e.g. soft pion in D*->Dpi
- sometimes helps to improve decay angle resolution