Noise filtering

Alexander Khanov

PHYS6260: Experimental Methods is HEP
Oklahoma State University

October 30, 2023

Track finding/fitting

- We want to find and fit a particle trajectory measured at N planes
- A straightforward approach: global least squares fit
  - try all combinations of points (one point per plane)
  - combine them in a least square estimator
  - pick the combination with least χ^2
  - exclude found points, repeat
- This approach is difficult for several reasons:
  - there are too many combinations – want to add one plane at a time
  - if there are n measurements, calculation of χ^2 implies inversion of an $n \times n$ covariance matrix – very computationally intensive
  - as the particle propagates from plane to plane, it undergoes multiple scattering – how to take it into account?

Kalman filter

- Kalman filter is a recursive least squares estimator, mathematically equivalent to global least squares fit
  - the computation time is proportional to the number of measuring planes and depends very little on the number of measurements per detector
  - due to recursive nature, it is well suited for combined track finding and fitting
- A particle trajectory is described by the state vector $\mathbf{x}$ and its covariance matrix of errors C
  - the typical choice of the state vector is q/p_{T}, θ, ϕ, x_{T}, y_{T}
- The model consists of particle propagation and measurement
  - as the particle propagates from plane $k-1$ to plane k, the state vector changes as $\mathbf{x}_k = F_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1}$, where $\mathbf{w}$ is propagation noise (due to multiple scattering) with mean $\langle \mathbf{w} \rangle = 0$ and covariance matrix $cov(\mathbf{w}) = Q$
  - at each plane k, we measure the position of the particle $\mathbf{m}$ that is related to the state vector as $\mathbf{m}_k = H_k\mathbf{x}_k + \epsilon_k$, where ϵ is measurement noise (detector measurement error) with mean $\langle \epsilon \rangle = 0$ and covariance matrix $cov(\epsilon) = V$
- In general, the model doesn't have to be linear, then $\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1}$, $\mathbf{m}_k = h_k(\mathbf{x}_k) + \epsilon_k$, and f_k, h_k must be expanded into Taylor series

The algorithm description

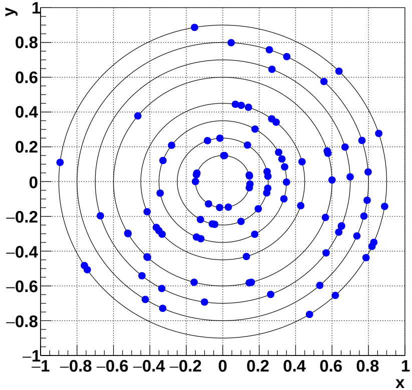
- The method begins with approximate values of the state vector components and "large" covariance matrix
  - theoretically the initial state vector can be arbitrary, but in practice it's better to pick some "good" initial approximation to make the procedure more stable
- The procedure consists of alternating two types of steps: prediction (extrapolation of the state vector from plane to plane), and filtering (update of the state vector in accordance with measurements at a given plane)
- Prediction
  - extrapolation of the state vector from plane $k-1$ to plane k: $\mathbf{x}_k^{k-1} = F_{k-1}\mathbf{x}_{k-1}$
  - extrapolation of the covariance matrix: $C_k^{k-1} = F_{k-1}C_{k-1}F_{k-1}^T + Q_{k-1}$
  - residuals of predictions: $\mathbf{r}_k^{k-1} = \mathbf{m}_k - H_k\mathbf{x}_k^{k-1}$
  - convariance matrix of predictions: $R_k^{k-1} = V_k + H_k C_k^{k-1} H_k^T$
- Filtering (weighted means formalism):
  - update of the state vector: $\mathbf{x}_k = C_k\left[(C_k^{k-1})^{-1}\mathbf{x}_k^{k-1} + H_k^T V_k^{-1}\mathbf{m}_k\right]$
  - update of the covariance matrix: $C_k = \left[(C_k^{k-1})^{-1} + H_k^T V_k^{-1} H_k\right]^{-1}$
  - χ^2 update: $\chi_k^2 = \chi_{k-1}^2 + \mathbf{r}_k^T V_k^{-1}\mathbf{r}_k + (\mathbf{x}_k - \mathbf{x}_k^{k-1})^T(C_k^{k-1})^{-1}(\mathbf{x}_k - \mathbf{x}_k^{k-1})$

Properties of the algorithm

- An additional step ("smoothing" or back propagation) allows one to evaluate the state vector at all previous planes taking into account all measurements
- Since χ^2 is evaluated at each step, it is easy to compare the candidates and pick (a few) best continuation(s) and detect outliers
- The method can be also used to fit vertex positions and separate secondary vertices from primary ones
- Generalizations:
  - deterministic annealing filter
  - Gaussian-sum filter
- Credits:
  - R. E. Kalman, J. Basic Eng. Mar 1960, 82(1) 35
  - P. Billoir, Nucl. Instr. and Meth. 225 (1984) 352: proposed the method for track finding without recognizing it as Kalman filter
  - R. Frühwirth, Nucl. Instr. and Meth. A262 (1987) 444: all you need to know about track and vertex fitting with Kalman filtering

Histogramming

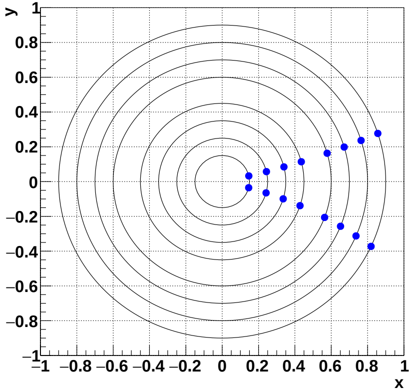
- When we are searching for new resonances, we are looking for signal peaks over flat background
- Sometimes the information about objects is spread out over the data, like in a hologram
- Can we bring it together to form a peak?



where are the arcs?

Histogramming

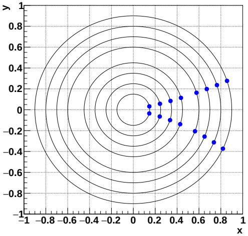
- When we are searching for new resonances, we are looking for signal peaks over flat background
- Sometimes the information about objects is spread out over the data, like in a hologram
- Can we bring it together to form a peak?

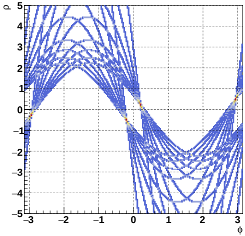


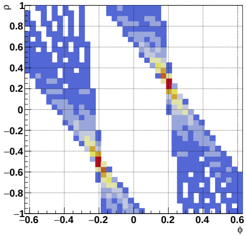
easy to see if there were no noise

Hough transform

- Trajectories originated from (0,0) can be described as $\rho r = 2\sin(\phi - \phi_0)$, where ϕ_0 is the trajectory slope at $r = 0$, and ρ is the curvature (signed inverse radius)
- This can be thought as a transformation from (x, y) space to (ϕ_0, ρ) space
  - in (x, y) space, each trajectory is a line, and measurements are points
  - in (ϕ_0, ρ) space, each trajectory is a point, and measurements are lines
  - lines from measurements of the same trajectory intersect at the same point which is the (ϕ_0, ρ) parameters of this trajectory

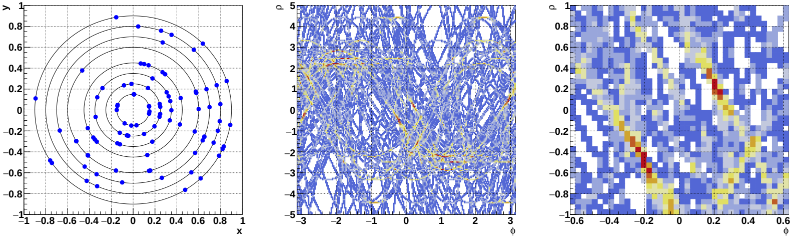






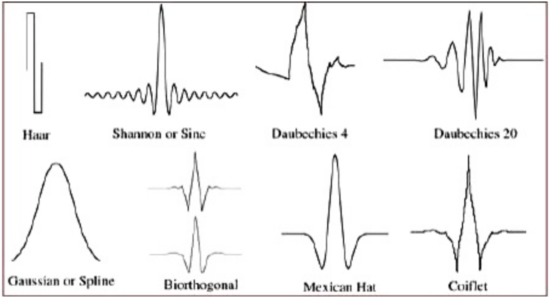
Hough transform

- Trajectories originated from (0,0) can be described as $\rho r = 2\sin(\phi - \phi_0)$, where ϕ_0 is the trajectory slope at $r = 0$, and ρ is the curvature (signed inverse radius)
- This can be thought as a transformation from (x, y) space to (ϕ_0, ρ) space
  - in (x, y) space, each trajectory is a line, and measurements are points
  - in (ϕ_0, ρ) space, each trajectory is a point, and measurements are lines
  - lines from measurements of the same trajectory intersect at the same point which is the (ϕ_0, ρ) parameters of this trajectory



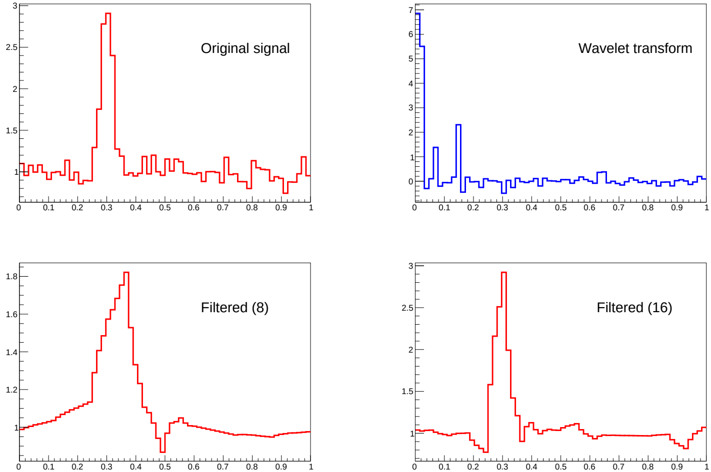
Wavelets

- What is wavelet?
  - it's kind of Fourier transform, but instead of sine/cosine basis it uses special kind of functions called wavelets: $X_{a,b} = \int_{-\infty}^{+\infty} x(t)\psi_{a,b}(t)\,dt$ (a=scale, b=translation)
- Regular Fourier transform doesn't catch local bursts in frequency (basically, it gets spikes in frequency but it doesn't know where they happen)
  - wavelets try to take care of that
- There are many various types of wavelets
  - for my example, I picked Daubechies 4-tap wavelets



Wavelets: strong signal

- Original signal: $1+2\times\text{Gaus}(0.3,0.02)$
- Filter: Daubechies-4 wavelets



Wavelets: weak signal

- Original signal: $1+0.2\times$Gaus(0.3,0.02)
- Filter: Daubechies-4 wavelets