ATLAS Particle Reconstruction discussion

Rodrigo Estevam de Paula

February 2nd, 2024

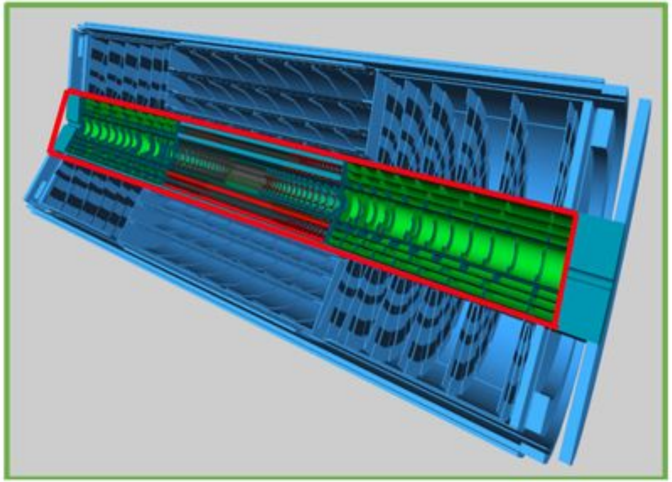
Outline

1. **ATLAS tracking particle process**
   a. Clustering
   b. Seeding
   c. Track finding and fitting
2. **Kalman Filters applied for track fitting**
3. **Kalman Filters applied for track finding**
4. **Possible improvements discussion**

- **More detailed overview (Personal notes)**

1. - Detector layout and working principle

- The track reconstruction is realized by a highly segmented detector formed by semiconductor sensors
- Electromagnetic particles ionizes the sensors and the charges are guided through an electrode and read by a readout chain



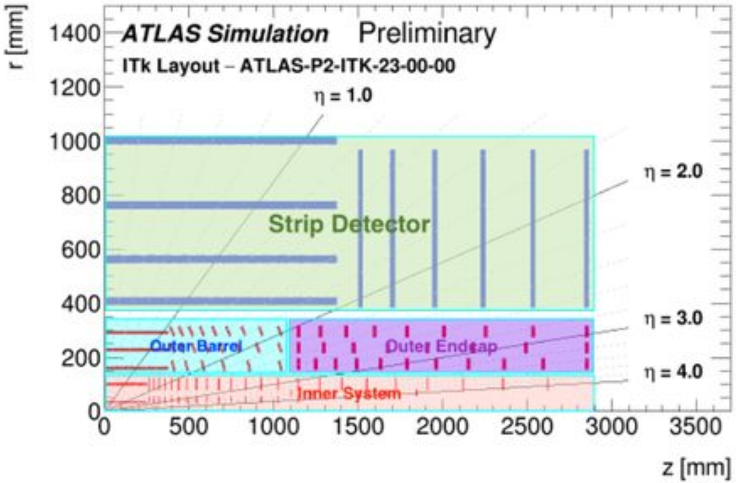


Fig 1.1: ATLAS Inner Tracker (ITk) layout

1.a - Clustering

- Rather than only activating one sensor per layer, one particle ionizes multiple sensors at once. In a manner that is necessary to cluster deposits originated from the same detection
- Given a central pixel, recursively check if neighbours are also activated (Fig 1.1)
  - An sensor is considered activated if the energy deposited there is above a certain threshold
- The center of the cluster can be estimated as:

$$\vec{r} = \frac{1}{N}\sum_{i=1}^{N} \vec{l}_i \qquad \vec{r} = \frac{1}{\sum_{i=1}^{N} q_i}\sum_{i=1}^{N} q_i \vec{l}_i .$$

$\vec{l}_i \quad q_i$: location and charge of i-th sensor

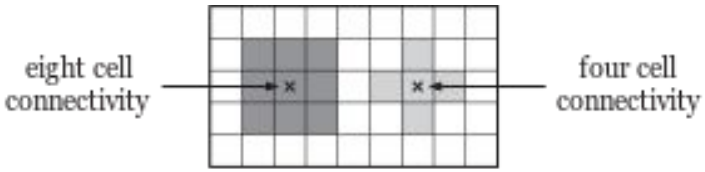


Figure 1.2: Criteria used to check neighbour connectivity [X]

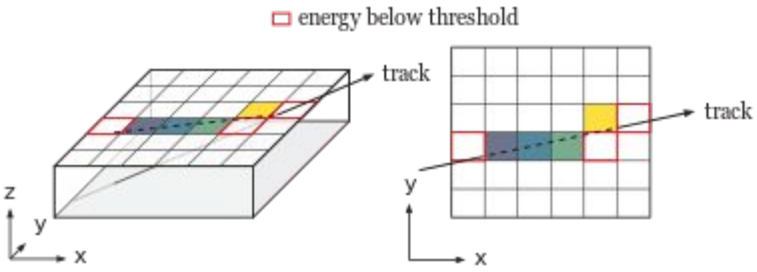
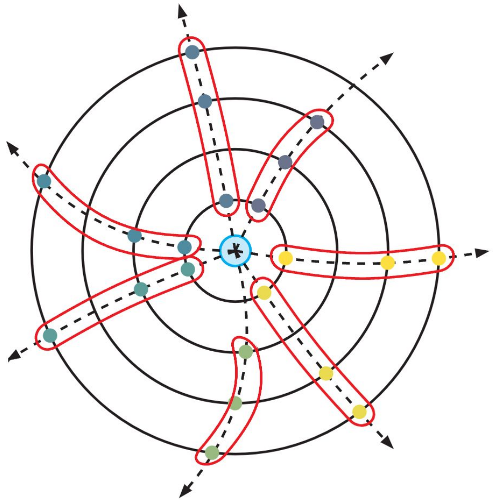


Figure 1.3: Example of clusterization [X]

1.b - Seeding

- Given the clusters, the seeding evaluates triplets of depositions aiming to find track candidates
  - Corse segmentation (high-level filter)
  - Evaluates if it follows an ellipsoidal trajectory
  - Many thresholds and quality factors are used in order to filter out bad candidates
- This is an important step that is way more complex than can be described in one slide

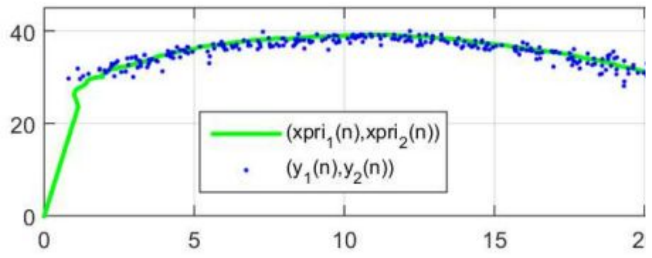


- Further reading: ACTS seeding implementation

1.c - Track Finding & Fitting

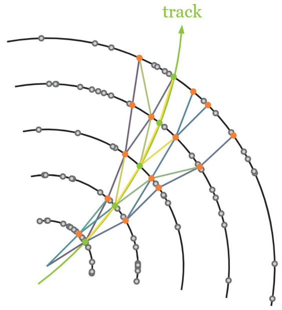
Track Fitting:

- Starting from an initial seed (closest to the interaction point) fits a trajectory to the depositions



Track Finding:

- As there can be multiple deposition on the same layer is important to choose the "right" track
- Find the most probable trajectory over all possibilities



How to do both? Combinational Kalman Filter!

Kalman Filters applied for track fitting

2.a - Defining a state space

- We can define the measures we observe as a function of the true position and measurement errors
- A simple measurement equation would be:

$$\vec{m}(n) = \mathbf{H}(n)\vec{x}(n) + \vec{\epsilon}(n)$$

n	layer (surface) index
$\vec{m}(n)$	measure vector (x,y,z)
$\mathbf{H}(n)$	projection matrix (x to m)
$\vec{x}(n)$	state vector ⇦ **Value to be estimated**
$\vec{\epsilon}(n)$	measurement error, white gaussian noise

- As we know the system dynamics, we can also define a system equation

$$\vec{x}(n) = \mathbf{F}(n-1)\vec{x}(n-1) + \vec{\omega}(n-1)$$

$\mathbf{F}(n-1)$	transport state vector from (n-1) to (n) *extrapolation achievable by numeric integration
$\vec{\omega}(n)$	system error, white gaussian noise is independent of the measurement error

- These two equations define our state space

2.b - Innovation process and estimative update

- If we have a prior estimative of the state vector (before observing the actual measurement) is possible to define a metric that measures the information gain that the new measurement offers
- The innovation is achievable with the following equations:

$$\vec{\alpha}(n) = \vec{m}(n) - \hat{m}(n)$$

$$\vec{\alpha}(n) = \vec{m}(n) - \mathbf{H}(n)\hat{x}(n|\mathfrak{m}_{n-1})$$

$$\hat{m}(n) = \mathbb{E}[\vec{m}(n)] = \mathbf{H}(n)\vec{x}(n)$$

$\hat{x}(n|\mathfrak{m}_{n-1})$ estimative of state vector given prior measures

- The innovation can be used to adjust the prior estimative:

$$\hat{x}(n|\mathfrak{m}_n) = \hat{x}(n|\mathfrak{m}_{n-1})) + \mathbf{K}(n)\vec{\alpha}(n)$$

- Where $\mathbf{K}(n)$ is the Kalman gain, which is chosen to minimize the mean-square value of the estimation error

$$\varepsilon(n|n) = \vec{x}(n) - \hat{x}(n|\mathfrak{m}_n)$$

$$\mathbb{J} = \mathbb{E}\{||\varepsilon(n|n)||^2\}$$

Finding the Kalman gain

- Defining the following correlation matrixes

$$\mathbf{P}(n|n) = \mathbb{E}\{\varepsilon(n|n)\varepsilon^T(n|n)\} \qquad \mathbf{S}(n) = \mathbb{E}\{\vec{\alpha}(n)\vec{\alpha}^T(n)\}$$

- We can express our error metric in function of P

$$\mathbb{J} = \mathbb{E}\{||\varepsilon(n|n)||^2\} = \mathrm{tr}[\mathbf{P}(n|n)]$$

- Then we just need to find the argument K that minimizes the metric

$$\mathbb{J}(\mathbf{K}(n)) = \mathrm{tr}[\mathbf{P}(n|n-1)] - 2\mathrm{tr}[\mathbf{K}(n)\mathbf{H}(n)\mathbf{P}(n|n-1)] + \mathrm{tr}[\mathbf{K}(n)\mathbf{S}(n)\mathbf{K}^T(n)]$$

$$\mathbf{K}^o(n) = \mathbf{P}(n|n-1)\mathbf{H}^T(n)\mathbf{S}^{-1}(n)$$

2.c Filtering Estimatives

- Iteration between **prior estimative** and **filtered estimative (posteriori)**

$$\vec{x}(n|\mathfrak{m}_n) = \vec{x}(n|\mathfrak{m}_{n-1}) + \mathbf{K}(n)\vec{\alpha}(n)$$
$$\vec{x}(n+1|\mathfrak{m}_n) = \mathbf{F}(n)\vec{x}(n|\mathfrak{m}_n)$$

- It's also necessary to iterate over error estimative matrixes in order to calculate the Kalman gain

$$\boldsymbol{P}(n|n) = [\boldsymbol{I} - \boldsymbol{K}(n)\boldsymbol{H}(n)]\boldsymbol{P}(n|n-1)$$
$$\boldsymbol{P}(n+1|n) = \boldsymbol{F}(n)\boldsymbol{P}(n|n)\boldsymbol{F}^T(n) + \mathbf{V}_x$$
$$\mathbf{K}^o(n) = \mathbf{P}(n|n-1)\mathbf{H}^T(n)\mathbf{S}^{-1}(n)$$

- After all measures are available, it is also possible to smooth the estimates.

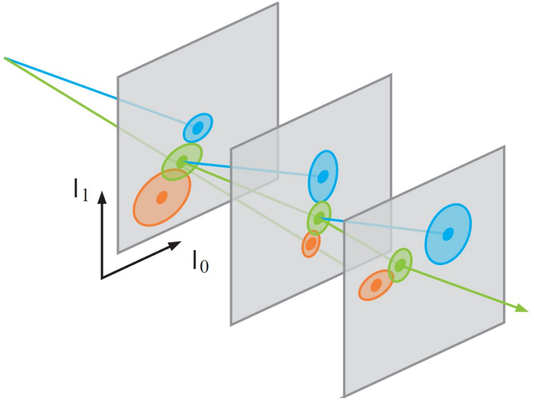


Figure 2.1: Illustration of KF estimative iteration. Measurement represented in orange, (prior) estimative in blue and filtered (posteriori) estimative in green.

2.d Kalman filter algorithm for fitting

```python
import numpy as np

def Kalman(m, Delta, var_u, var_f):
    #Incializacao
    N = len(m)
    M = 3
    H = np.eye(M)
    x_post = np.zeros(N,M)
    x_prior = np.zeros(N,M)
    x_prior[0] = m[0]
        cov_x = var_u*np.eye(M)
    cov_m = var_f*np.eye(M)
    #estimative corr error matrix
    P_prior = Delta*np.eye(M)

    for i in range(N):
        # update corr matrix of information gain
        alpha_corr = H*P_prior*H.T + cov_y
        #Kalman gain calculation
        K = P_prior*H.T*np.inv(alpha_corr)
        #information gain update
        alpha = m[i] - H*x_prior
        #Posteriori estimatives
        x_post[i] = x_prior[i] + K*alpha
        P_post = (np.eye(M) - K*H)*P_prior
        #Update matrix F to use extrapolation
        # from layer i to i+1 and calcualte
        # prior estimatives for i+1
        F = transport(i)
        x_prior[i+1] = F*x_post[i]
        P_prior = F*P_post*F.T + cov_x
    return x_post
```

Kalman Filters applied for track finding

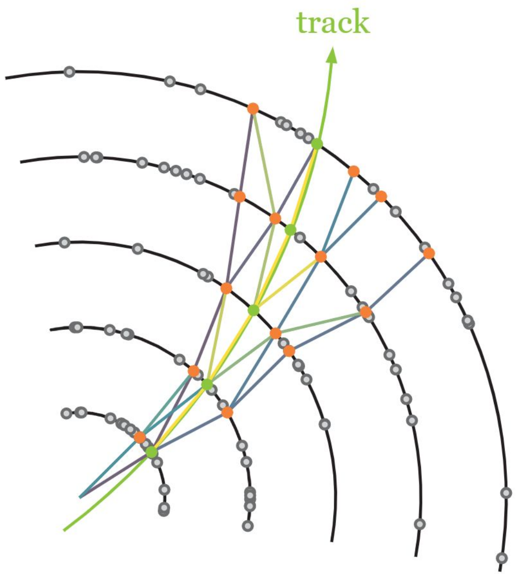
Track scoring

- We can define a residual between the posteriori estimate and the measure

$$\vec{r}(n) = \vec{m}(n) - \mathbf{H}(n)\hat{x}(n|\mathfrak{m}_n)$$

- This value contributes to a quality factor of the reconstructed track

$$\chi^2_+ = \vec{r}^T(n)[(\mathbf{1}-\mathbf{H}(n)\mathbf{K}(n))\mathbf{V}(n)]^{-1}\vec{r}(n)$$



- The algorithm iterates over all possible tracks and uses the global quality parameter χ^2 (also depends on other track attributes*) to filter the best estimate tracks

Discussion

Possible improvements

Seedling

- Need to understand better how it is structured

Track fitting

- Substitute Kalman for RLS algorithms (less accurate but faster)
  - Suggested on Haykin's book [1]
  - Raffaello research*
    - https://ieeexplore.ieee.org/abstract/document/9436012
    - https://ieeexplore.ieee.org/abstract/document/8645174
- Increase parallelism

Track finding

- Include HGTD hits in the cumulative score
  - Use additional time information in the decision process
- Increase parallelism

Machine learning base strategies

Next steps

Explore the ACTS track reconstruction framework (on going)

- Understand how to simulate complex events with high luminosity
- Extract fitting performance metrics
- Next meeting:
  - Simulation of simple events and extraction of performance metrics

Follow HGTD ACTS integration campaign

Study more about Machine Learning based reconstruction

- Physics Informed Machine Learning
- GNNs

References

[1] Simon Haykin. *Adaptive filter theory*. Prentice Hall, Upper Saddle River, NJ, 4th edition, 2002.

[2] Paul Gessinger-Befurt. *Development and improvement of track reconstruction software and search for disappearing tracks with the ATLAS experiment*, 2021. Presented 30 Apr 2021.

Backup