Everything Everywhere “Parallel” at once

SACHIN GUPTA

PHYSIKALISCHES INSTITUT

TRIFELS ANNUAL RETREAT

Overview

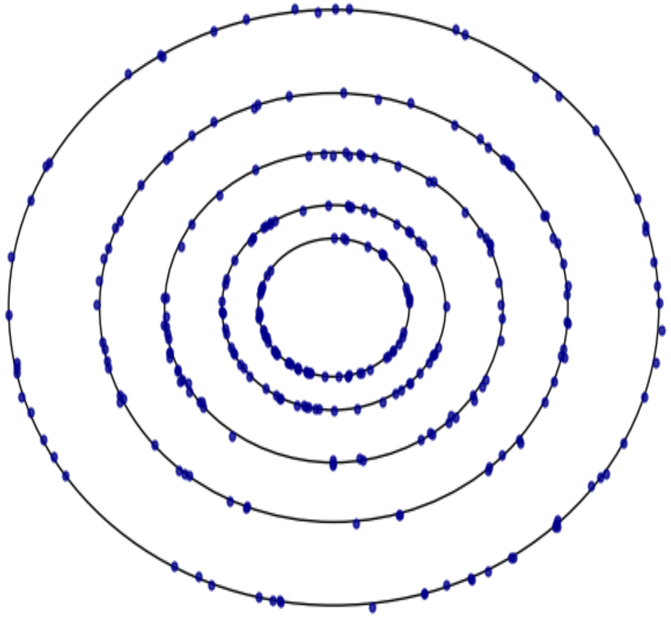
- Part 1 : Why “everything everywhere” ?
- Part 2 : Why “parallel” at once ?
- Part 3 : Result

Everything, Everywhere

Track Reconstruction

- Goal : Getting the properties of charged particle from the raw detector measurement.

- Done in two stages :
  - Track finding : (Grouping of hits)
  - Track fitting : (Fitting track parameters)



Detector layers with hits

What is Track Finding ?

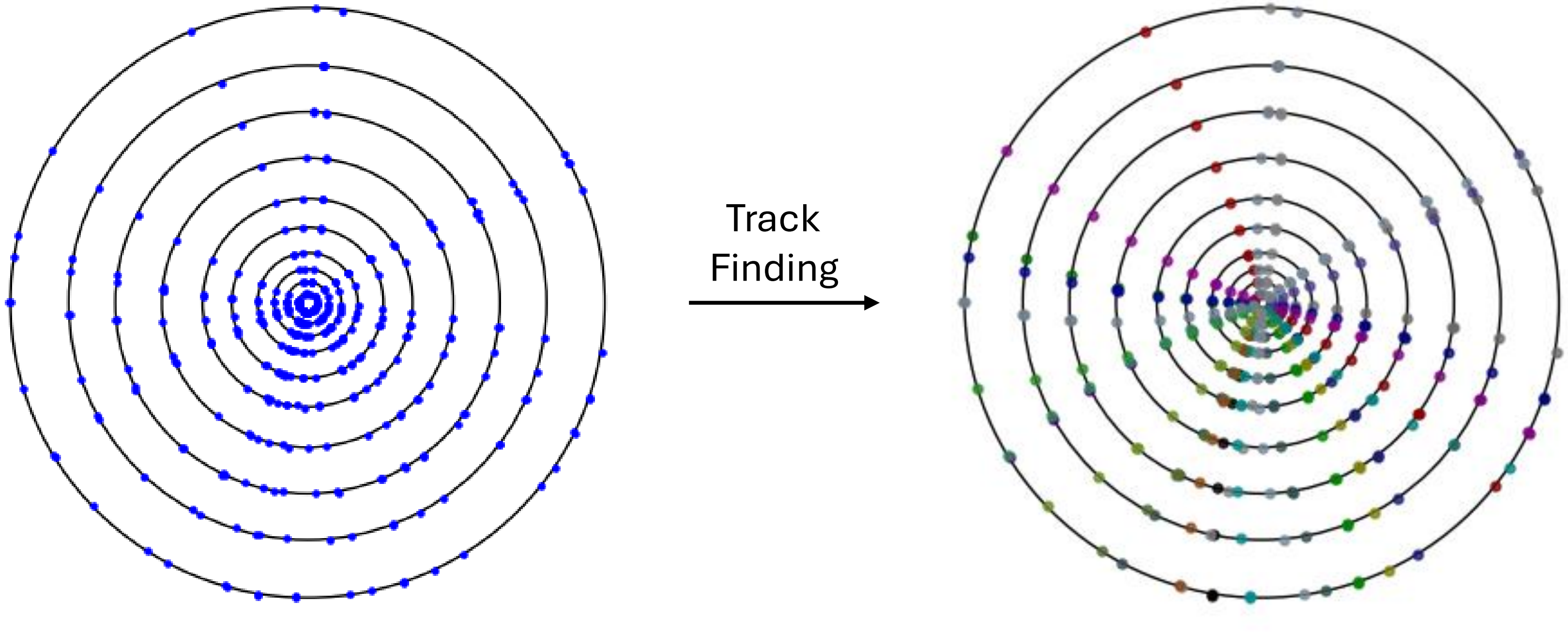
September 2022

February 2023

Track finding : Grouping leaves

- From the pile of leaves, form sets.
- Each set can be associated with a branch symbolizing a particle trajectory.

Track finding



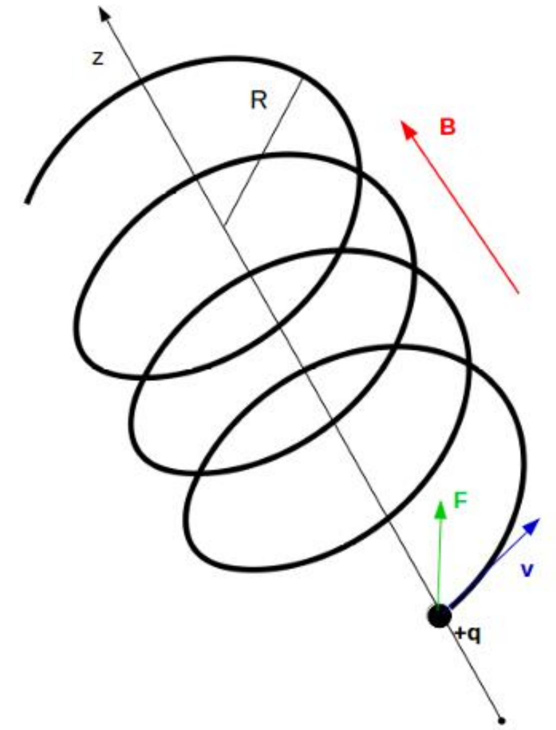
- Track Finding is the pattern recognition problem.
- Forming set of hits such that each set corresponds to track candidate.

Track fitting

- From measurement to parameters

- Kinematic properties of a particle tracks are obtained after fitting.

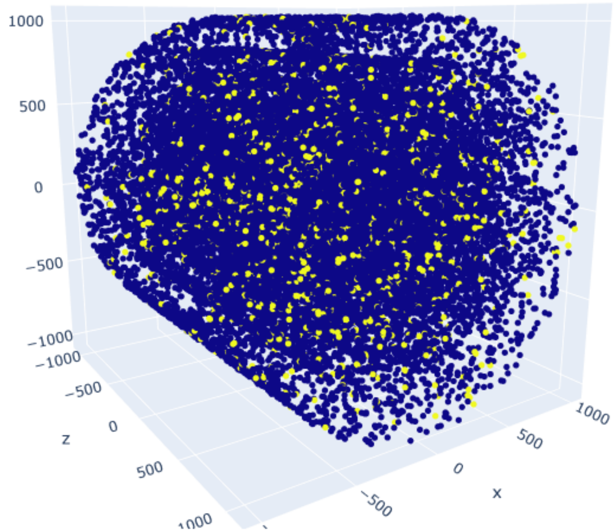
- Eg: Describing trajectory with helix.



Why Everything, Everywhere ?

- Both steps consume substantial amount of computing resources.

- Computationally intensive task in high multiplicity environment.



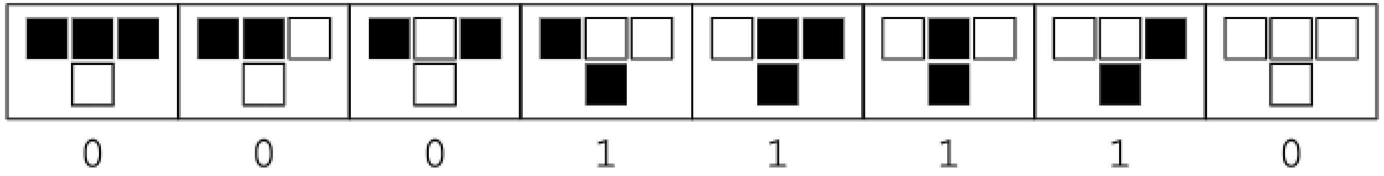
“Parallel”

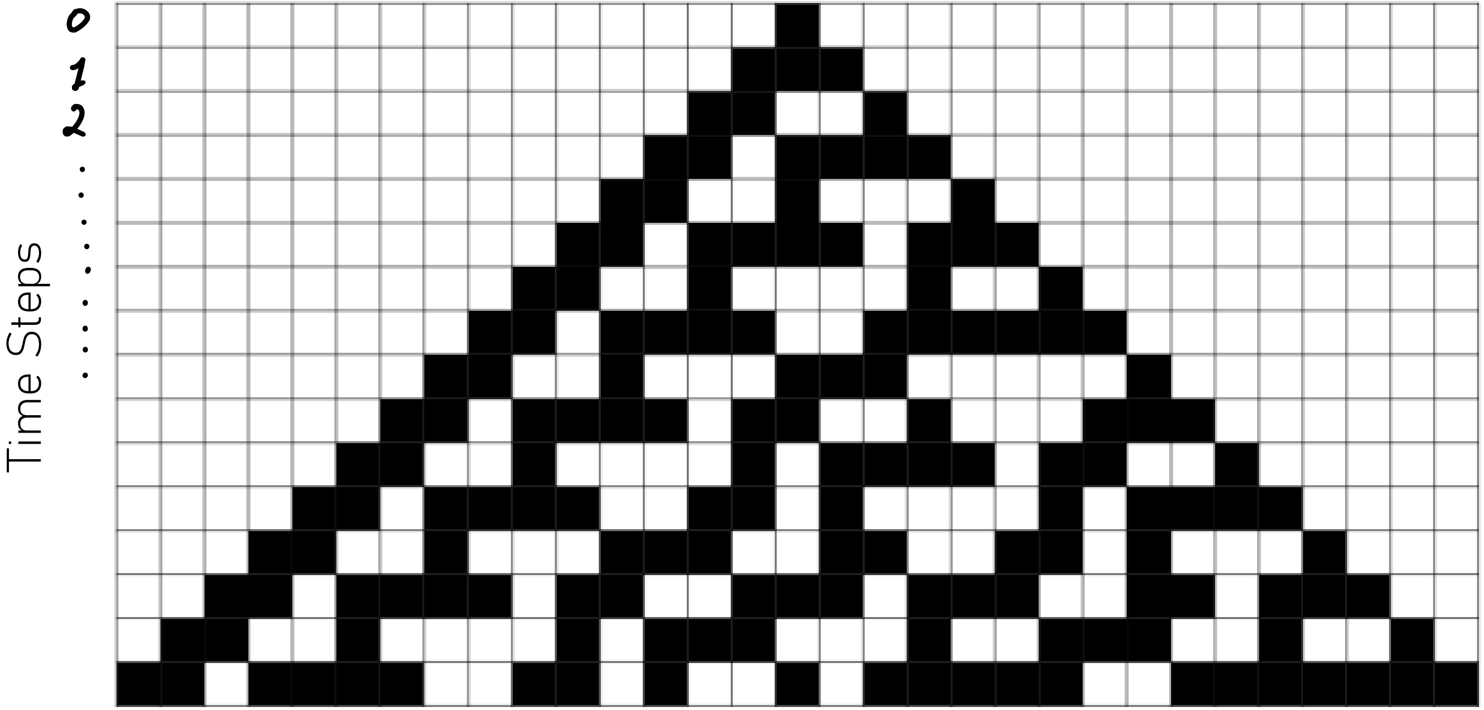
At once

Cellular Automata

- Dynamical system where space and time is both discretized.
- Each space site is called “Cell” that can either take integer or binary values.
- The evolution of each cell depends on its local neighbour cell values. The rule is universal.
- In one time step, all cells are updated simultaneously.

Example Rule 30 (1 D Grid)





https://mathworld.wolfram.com/Rule30.html

Takeaway

- CA are local systems and gives full freedom for deciding
  - Information to be embedded on a cell (universal)
  - Neighbour formation
  - Local law for cell evolution

- Since all cells in a grid update their values simultaneously, they can be implemented on GPUs (CAs are parallelizable)

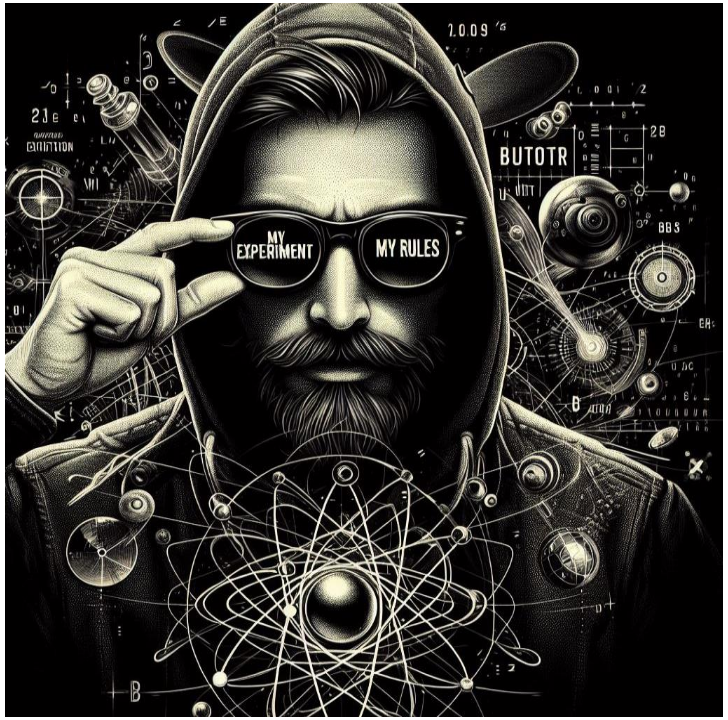
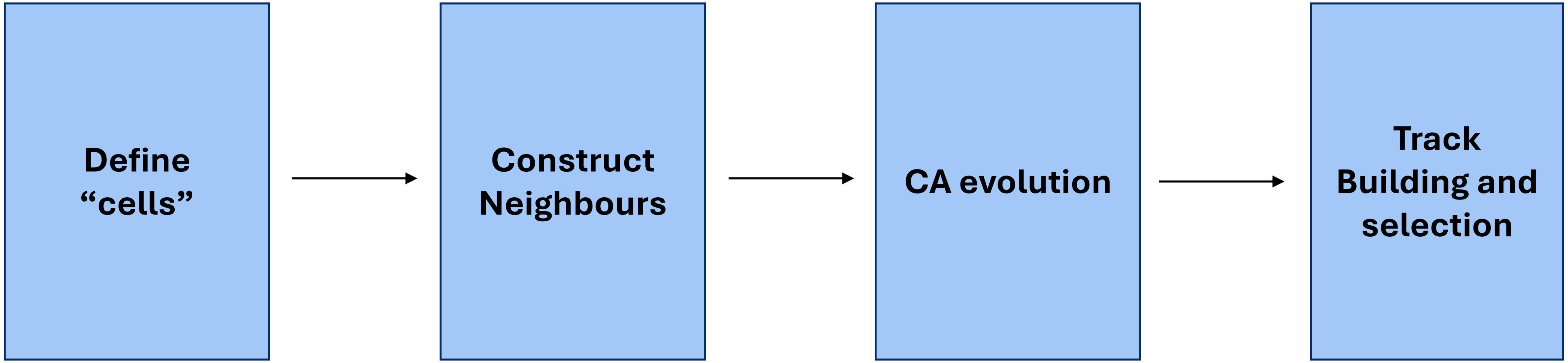


Image generated by MS copilot

Implementation of CA for track finding



Define “cells”

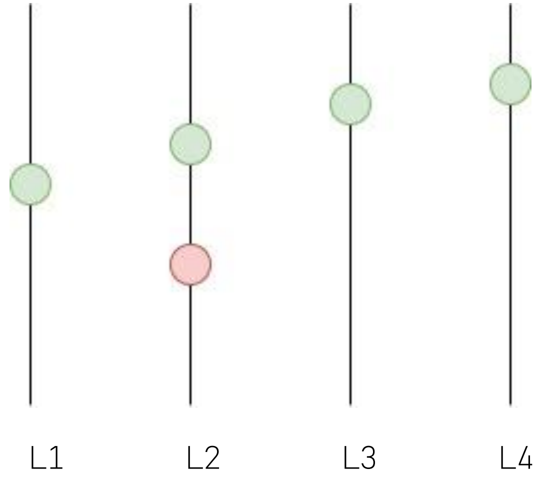
Construct Neighbours

CA evolution

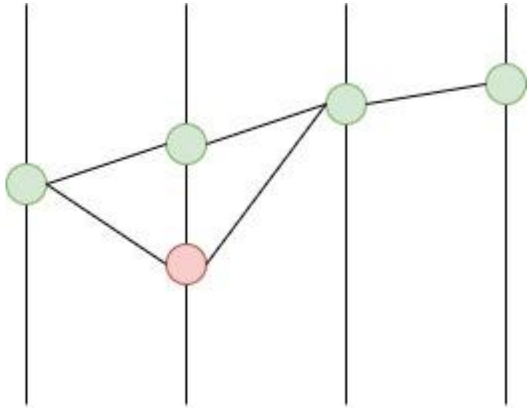
Track Building and selection

Single Hit as a Cell

- Example – 4 layer detector , one particle track with one noisy hit (B =0)
- Cells – Hits
- Neighbour – all Hits in previous layer – (fully connected graph)
- No neighbour within the layer



Defining cells

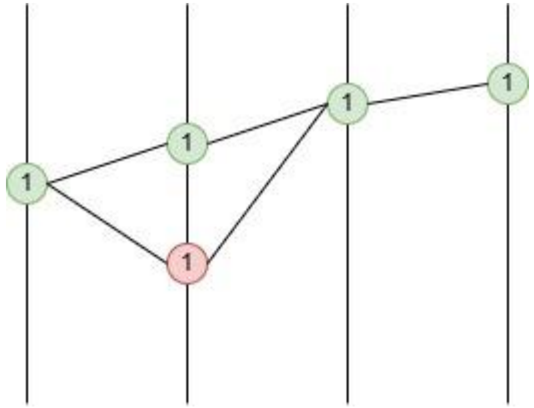


Neighbour formation

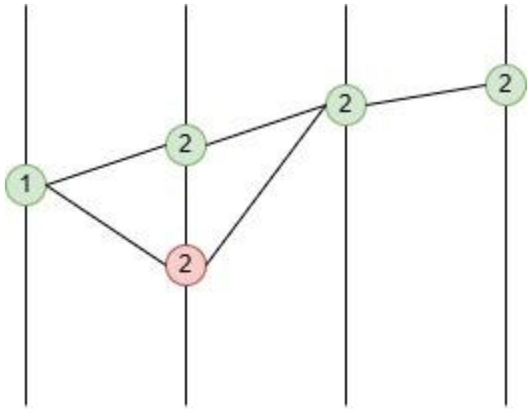
CA evolution Rules

- Initialize each cell with value = 1
- At each iteration –
  - each cell looks at its left neighbour (if any)
  - Find the max(neighbour values)
  - New value = max(neighbour values) +1

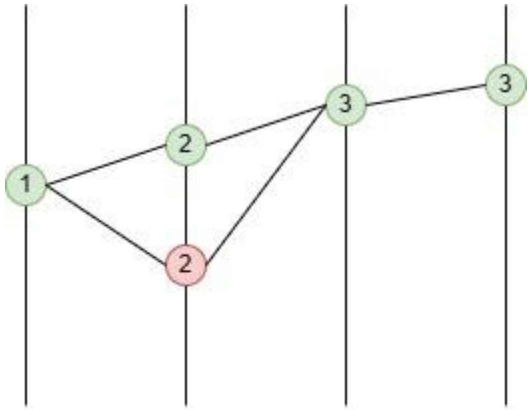
t = 0



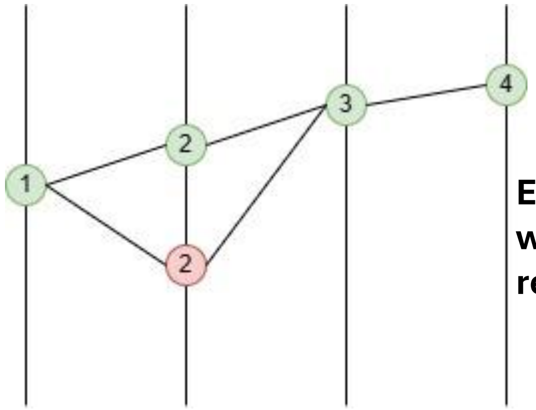
This ensures the longer tracks are preferred



t = 1



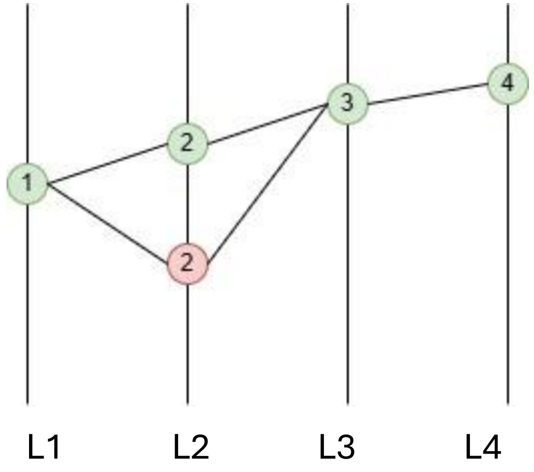
t = 2



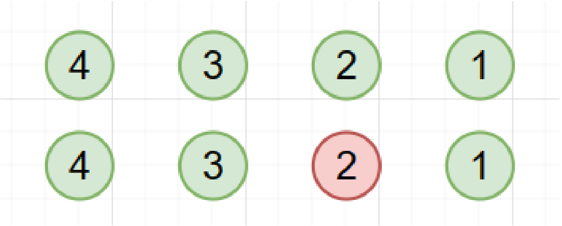
t = 3

Evolution is stopped when CA grid values remains unchanged

Track Building (Depth First search)



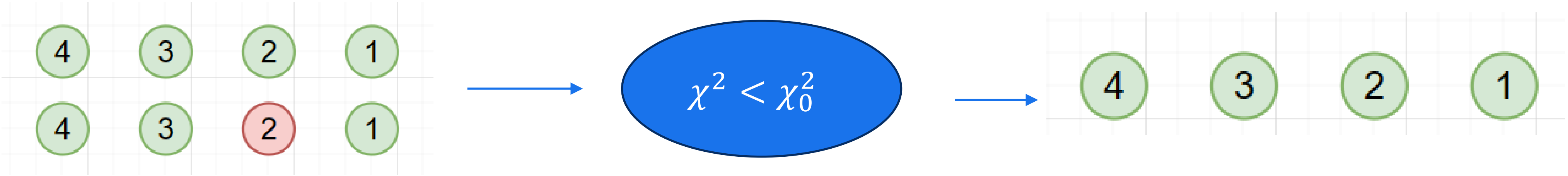
- Track building starts with the final configuration of CA.
- Our aim was to construct track with four hits so we start with the layer 4 cells.
- In this case two tracks are possible –



- Not only that, but cell’s value also represents the position of each cell in a track

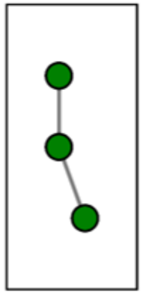
Track Selection

- χ^2 cut can be used to further select the track
- χ^2 is calculated for each track and the best value is selected.



Triplet as a Cell

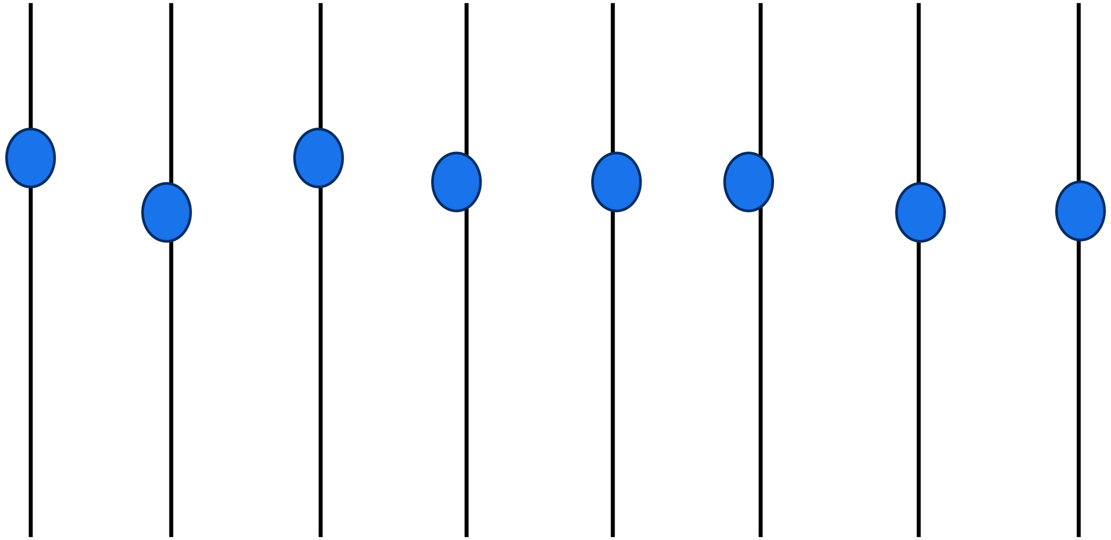
- In high track multiplicity environment ($\approx 10^5$ hits), using single hit as a cell will become highly computationally and memory intensive.
- Instead higher level information in a cell should be embedded on “cell” i.e. Triplet



- Many fake combinations can be neglected before CA evolution.
- Since minimum three points are required to calculated the p_T of a track, thus triplets are well suited for our approach.

Visualization

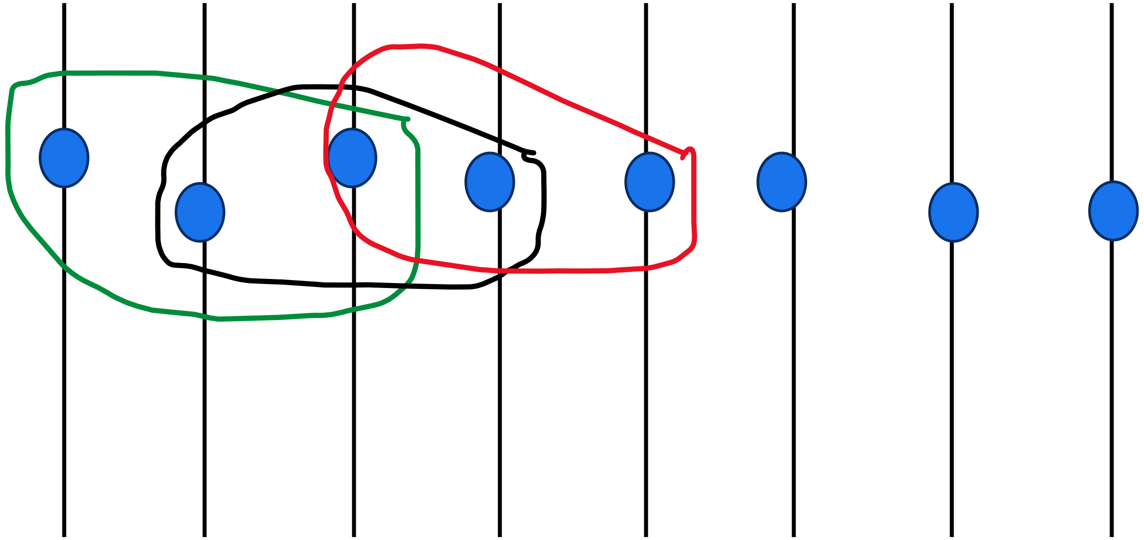
Forming track by connecting Hit



8 detector layers

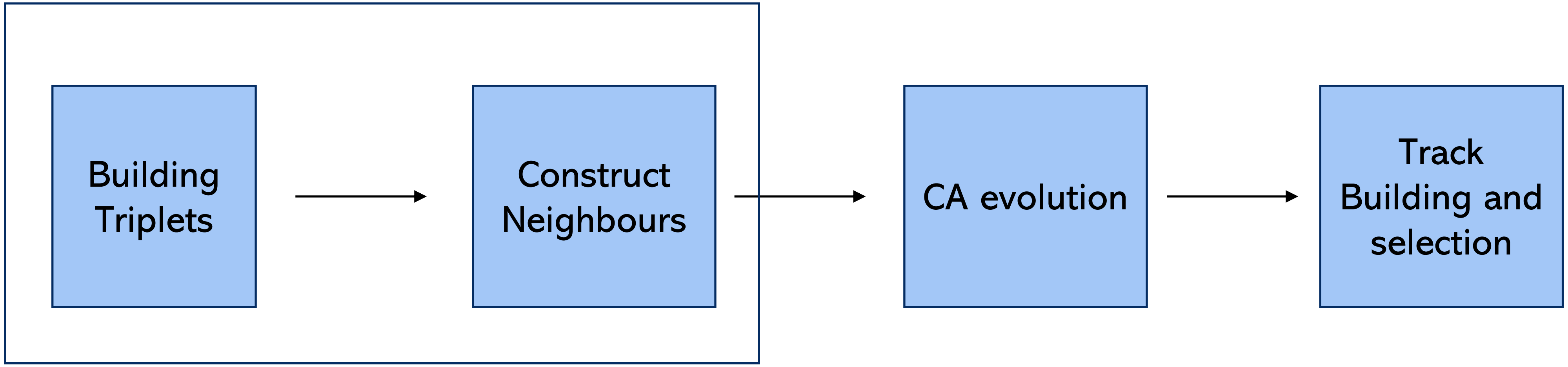
Visualization

Forming track by connecting Triplet



8 detector layers

Implementation of CA for track finding



Building Triplets

Construct Neighbours

CA evolution

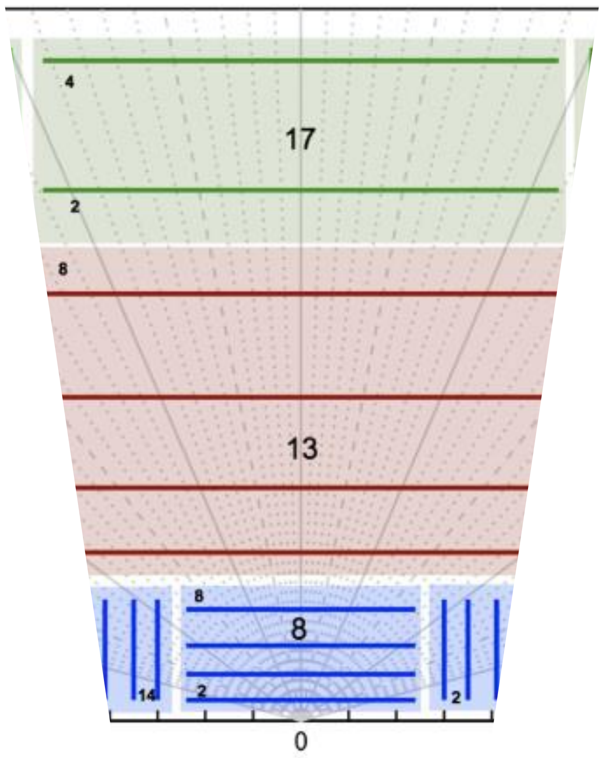
Track Building and selection

Two consecutive
triplets having
two common hits

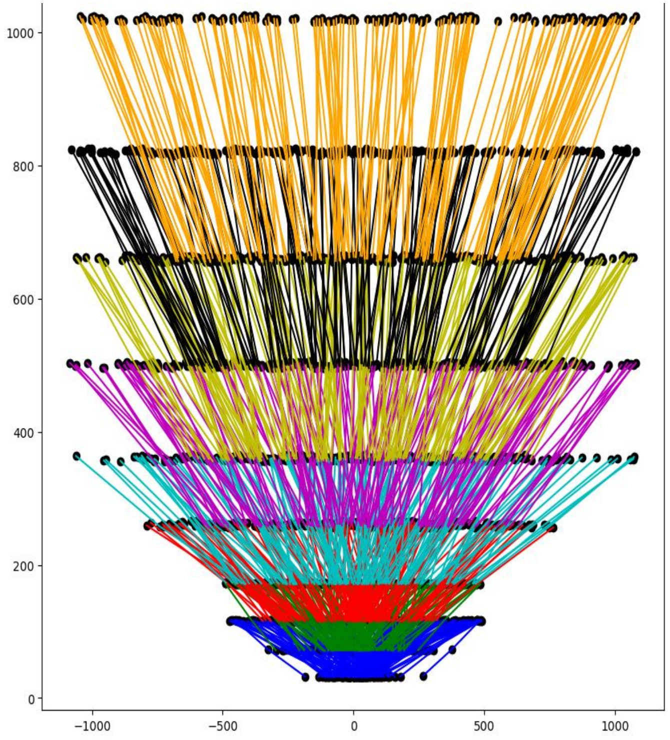
Building Triplets with TrackML Dataset

- Simplified detector geometry, adapted from early ATLAS ITk designs
- Pile – up 200 conditions like @ HL-LHC

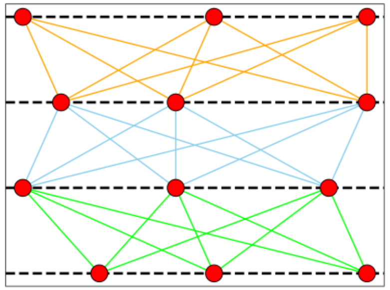
Aim :
Finding
longest track
in barrel
region



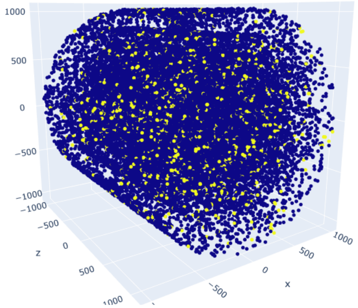
Barrel region of the TrackML Dataset



Roadmap for Building Triplet



Dataset of barrel region including noise

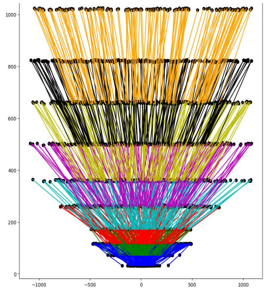


↓

Doublet Formation (Pair of two hits in consecutive layers)

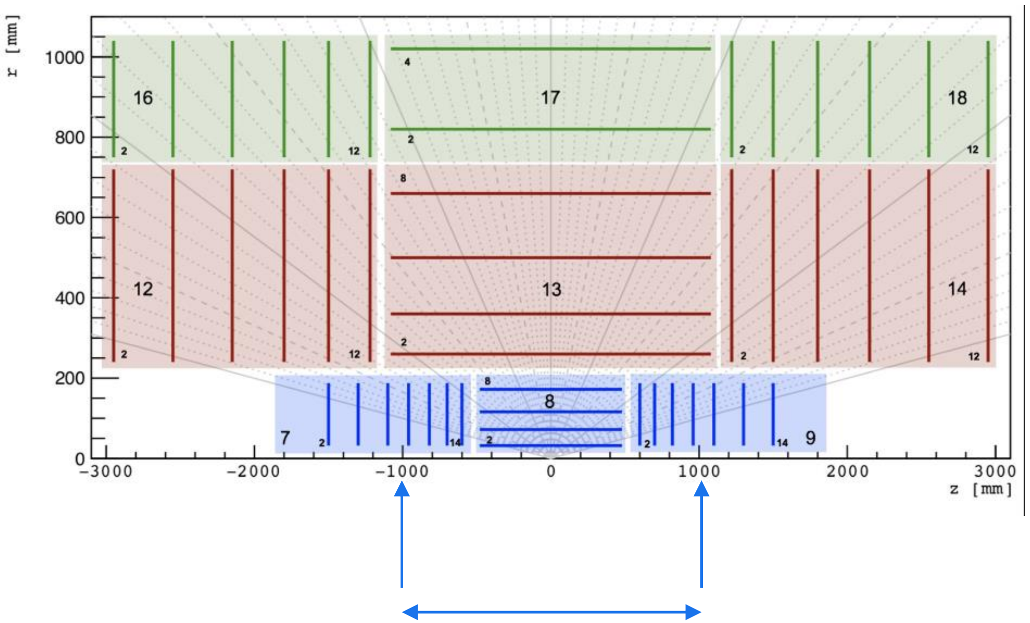
↓

Triplet Formation (formed from two doublets sharing one hit)



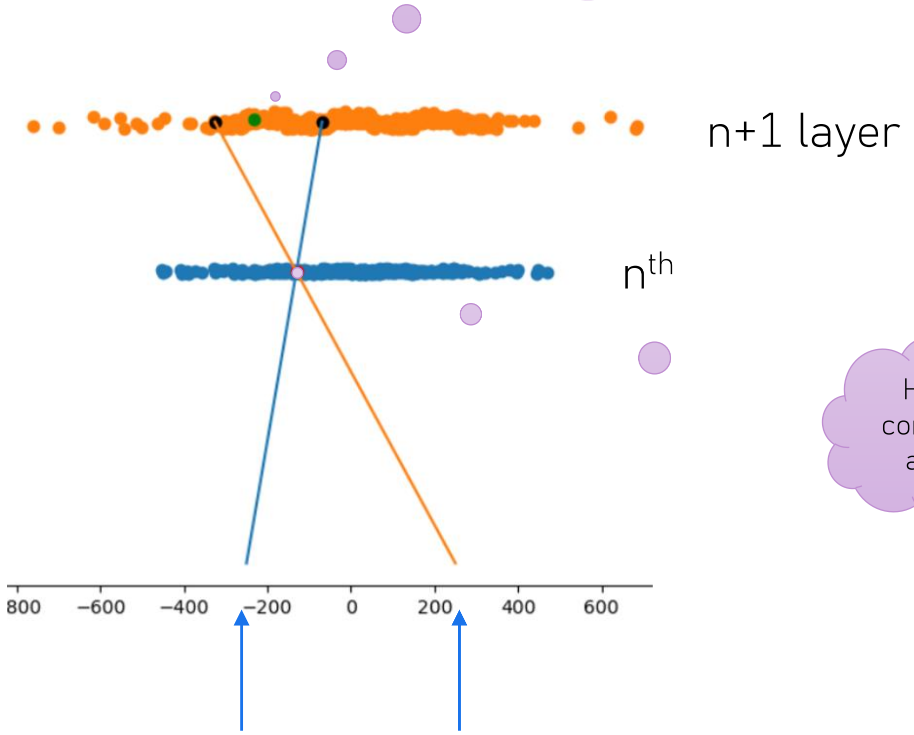
Doublet cut : z_0 search window

Longitudinal plane



$2z_0$

Form all doublets within this window



Hit in consideration

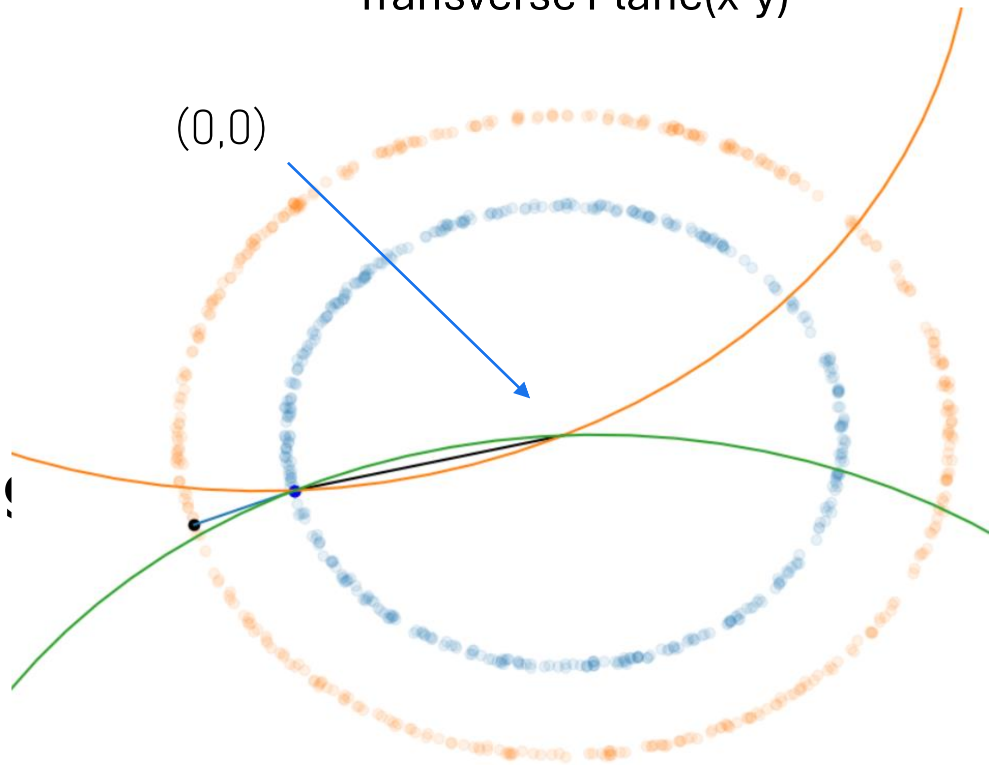
$-z_0$ z_0

$d\phi$ search window

- Beamline constrained in transverse plane.
- Target Particles $\mathrm{p}_T > 1$ GeV.
- Two circle can be formed that passes throu hit in consideration and (0,0).

$p_{T_{cut}}$(GeV) = 0.3 B r (m)

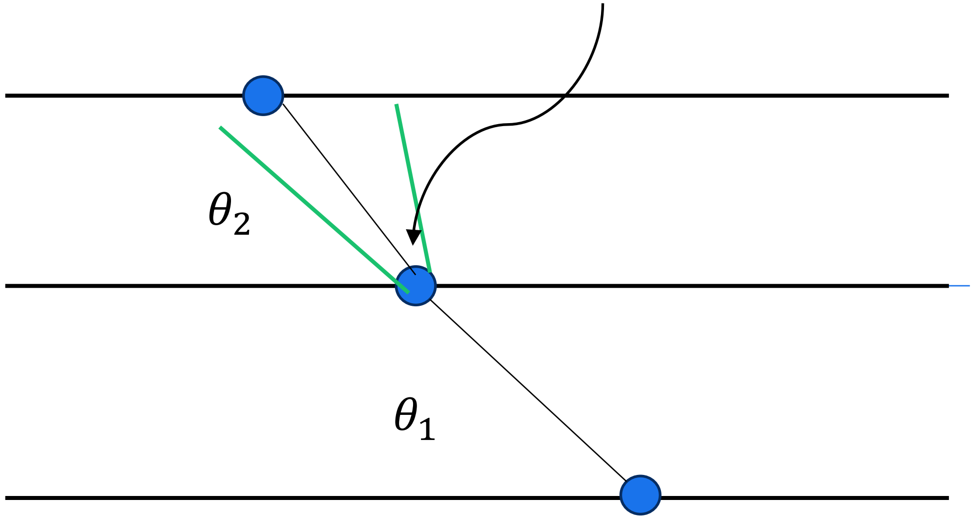
Transverse Plane(x-y)



- ϕ : angle made by hit (blue) with the x axis
- $\phi + d\phi$: Green circle
- $\phi - d\phi$: Orange circle

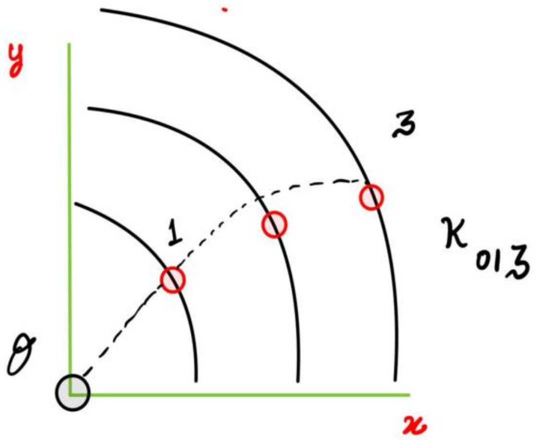
Triplets Cuts

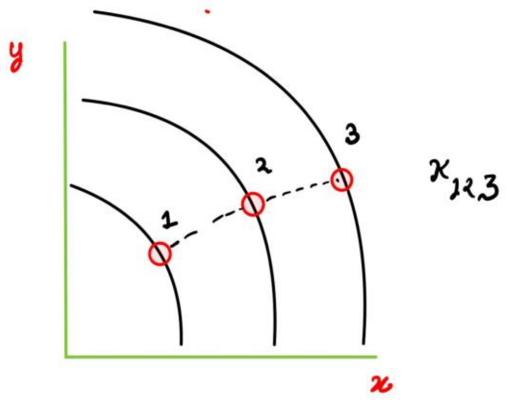
Polar angle difference $d\theta$



- **No bending in Longitudinal plane**

Curvature Difference $d\kappa$

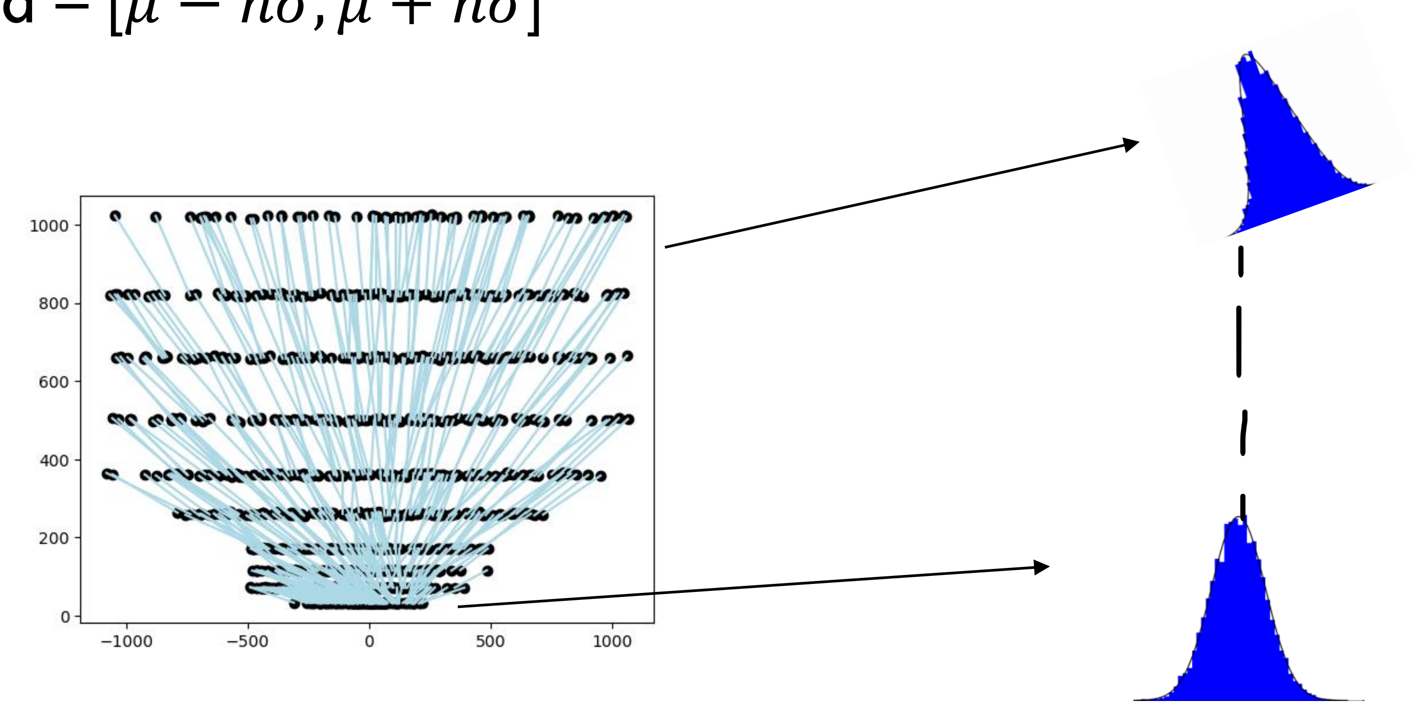




Transverse plane

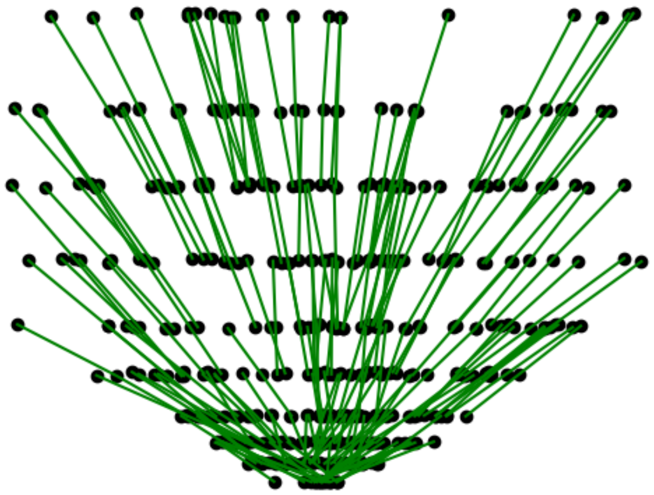
Layer wise cuts

- Three quantities($z_0, d\theta, d\kappa$) are calculated for truth segments for each layer.
- In all three cases gaussian is fitted for each layer data and the selection window is calculated – $[\mu - n\sigma, \mu + n\sigma]$

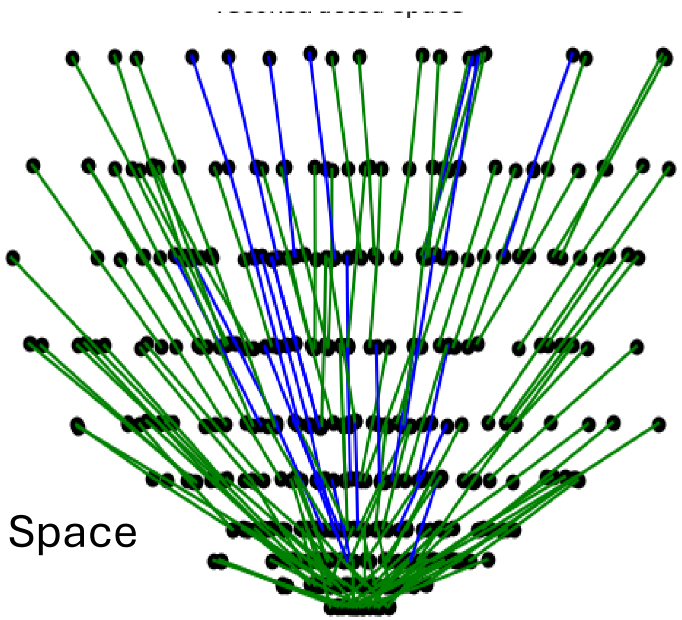


Performance metric

- Efficiency $= \frac{\#\ reconstructed\ signal\ segments}{\#\ total\ signal\ segments}$
  - $\frac{\#\ Green\ (RS)}{\#Green\ (TS)}$
- Purity $= \frac{\#\ reconstructed\ signal\ segments}{\#\ total\ reconstructed\ \ segments}$
  - $\frac{\#\ Green\ (RS)}{\#Green\ (RS) + \#Blue(RS)}$

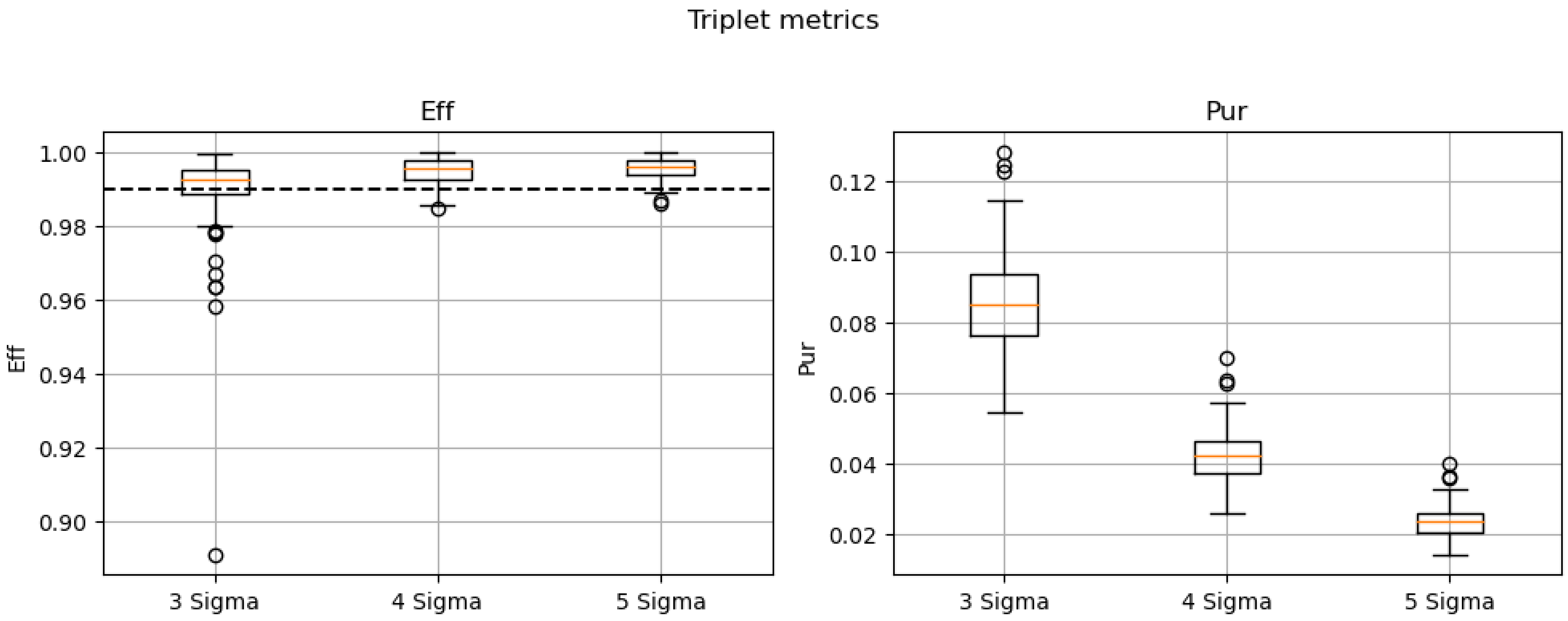


Truth Space



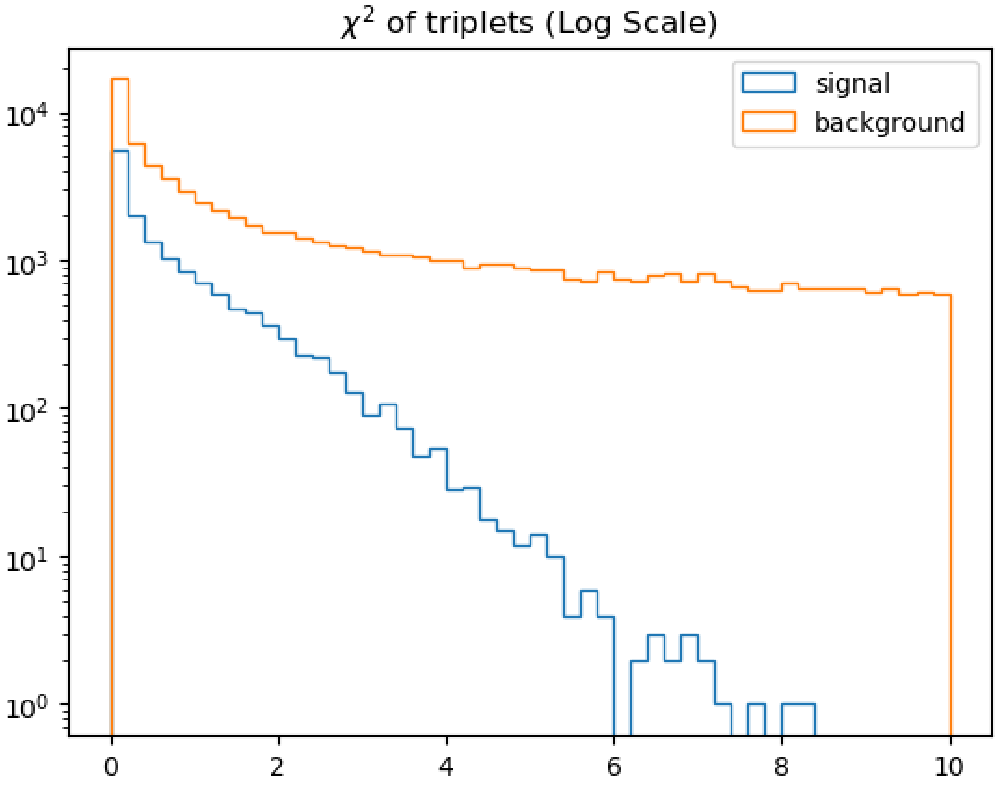
Reconstructed Space

Triplet Metrics

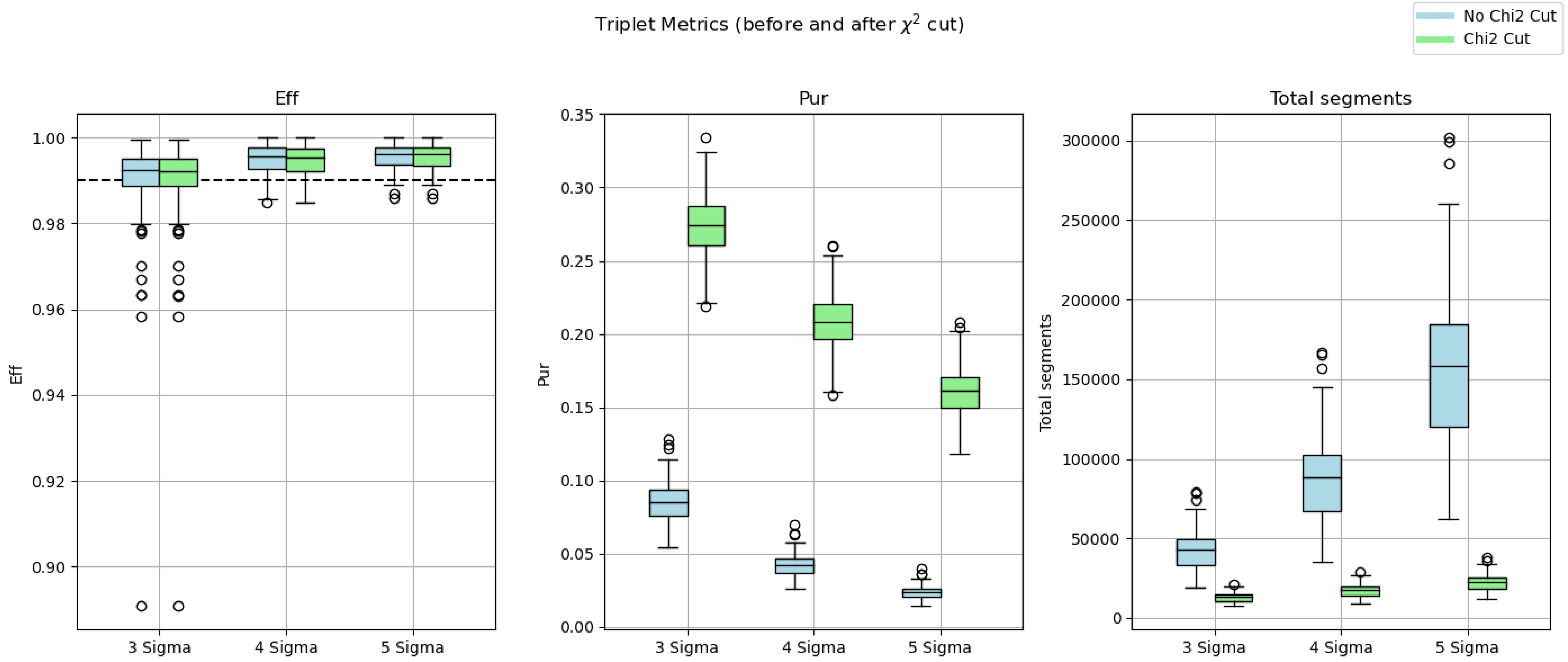


Fitting Triplets

- Local Triplet fit : Implemented and χ^2 is calculated for each triplet
- Chi square cut is applied before CA evolution

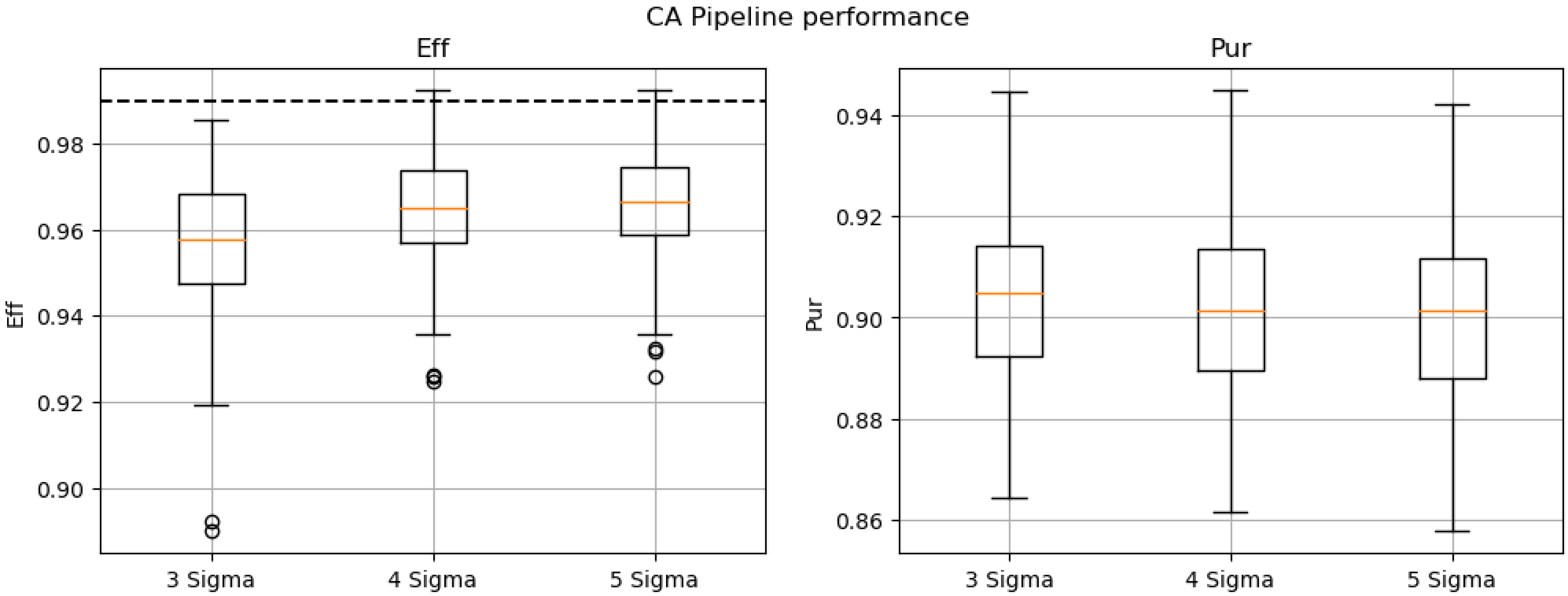


The magic of Triplet Fit



Final Track finding performance

- Reconstructed signal if 50 % or more hits belong to a signal track.



Summary and outlook

- Physics performance of Track finding and track fitting algorithm that is fully parallelizable for High multiplicity environment.
- The final efficiency above 3 sigma stays above 94 % and purity stays around 90 % for all sigmas
- Currently optimizing algorithm for missing tracks.
- Future aspects : GPU implementation of the entire pipeline

Thank you for your attention

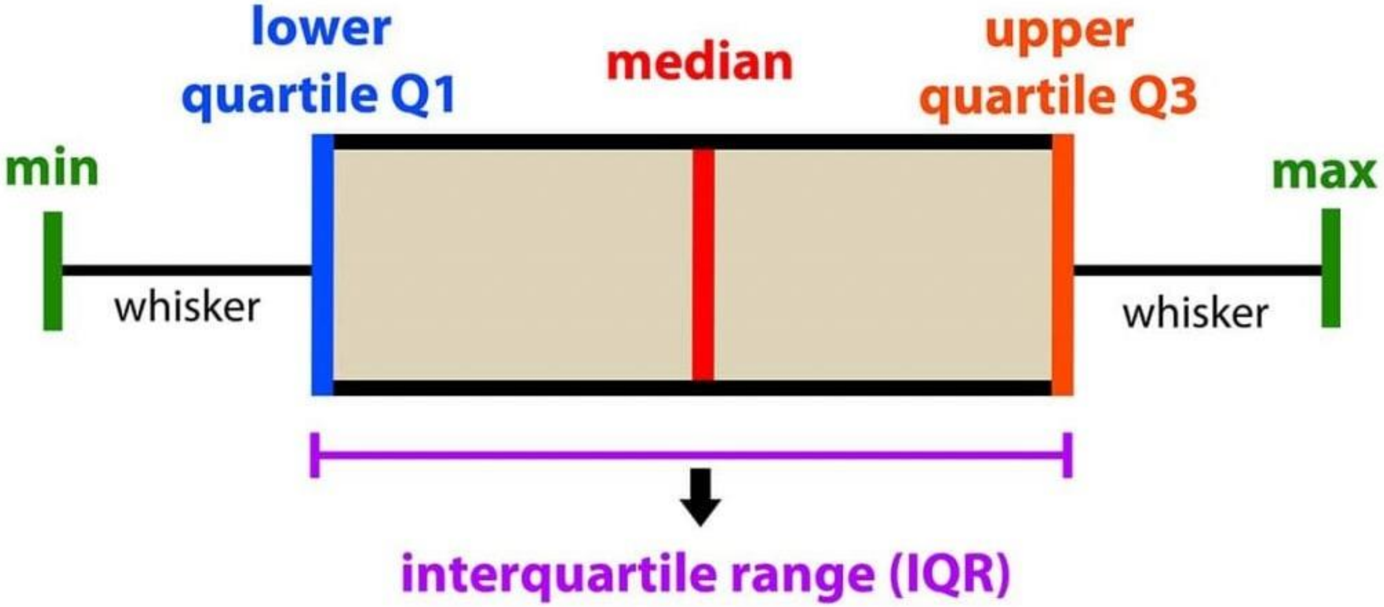
The Five Orange Pips, The Adventure of Sherlock Holmes

The ideal reasoner would, when he had once been shown a single fact in all its bearings, deduce from it not only all the chain of events which led up to it but also all the results which would follow from it.

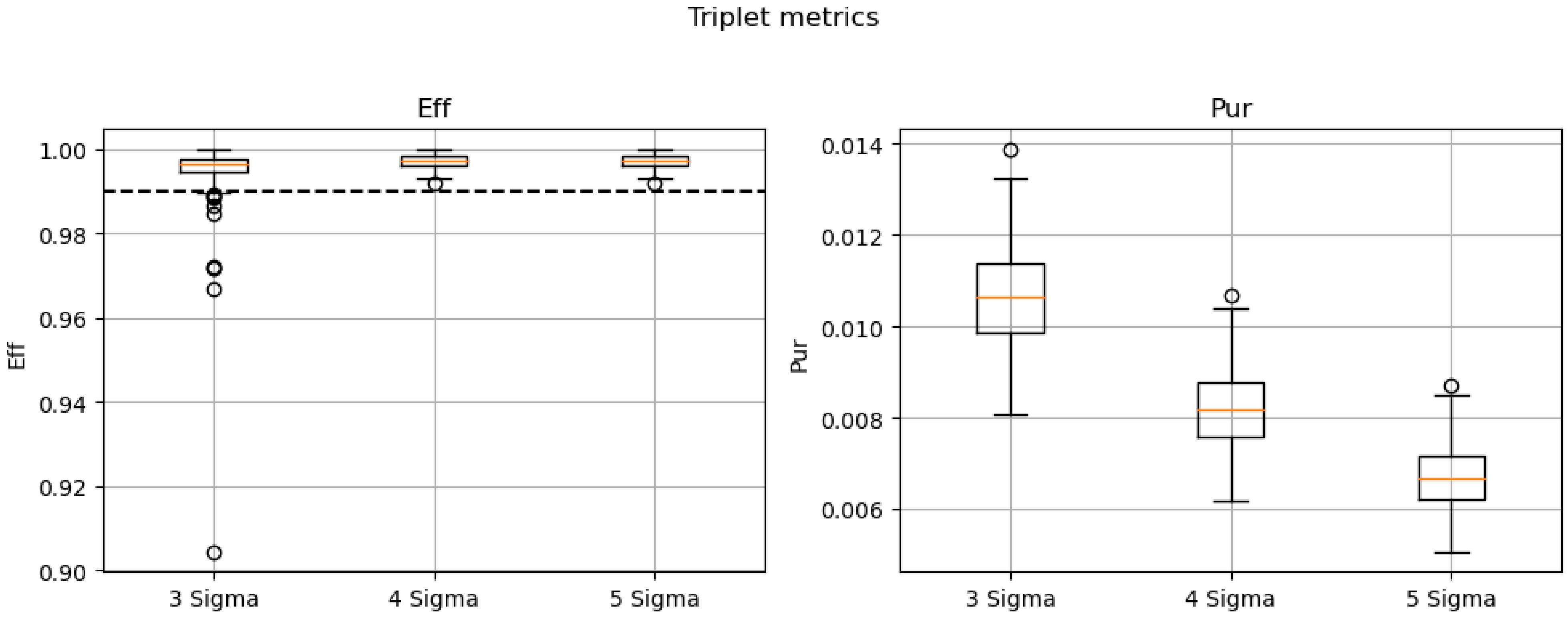
- Backup

Box and Whiskers Plot

- Represents where the most data is situated at .
- Spread of the data without looking at actual distribution.
- Helps spot outliers in the data.
- Box contains 50 % of the data, outliers lie outside of the whiskers .



Doublet eff & purity



Track Building : Depth First search