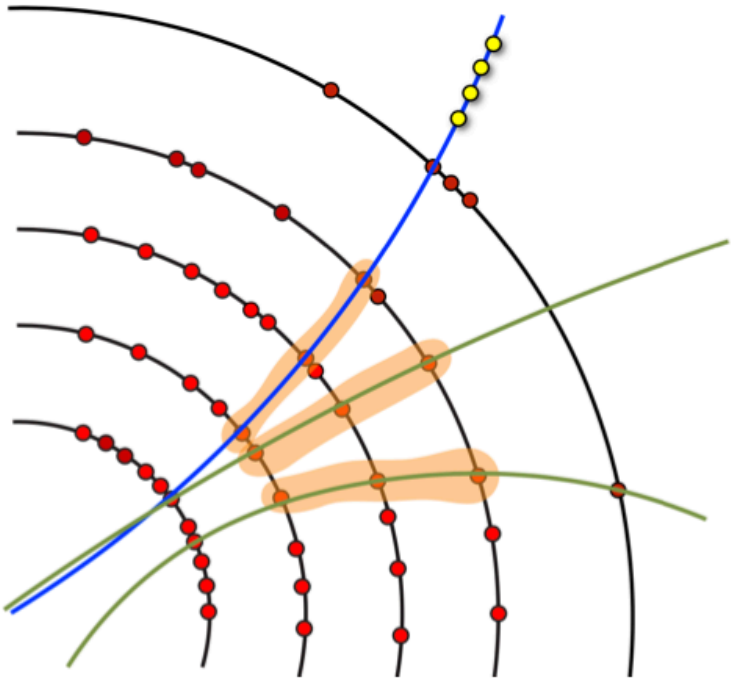
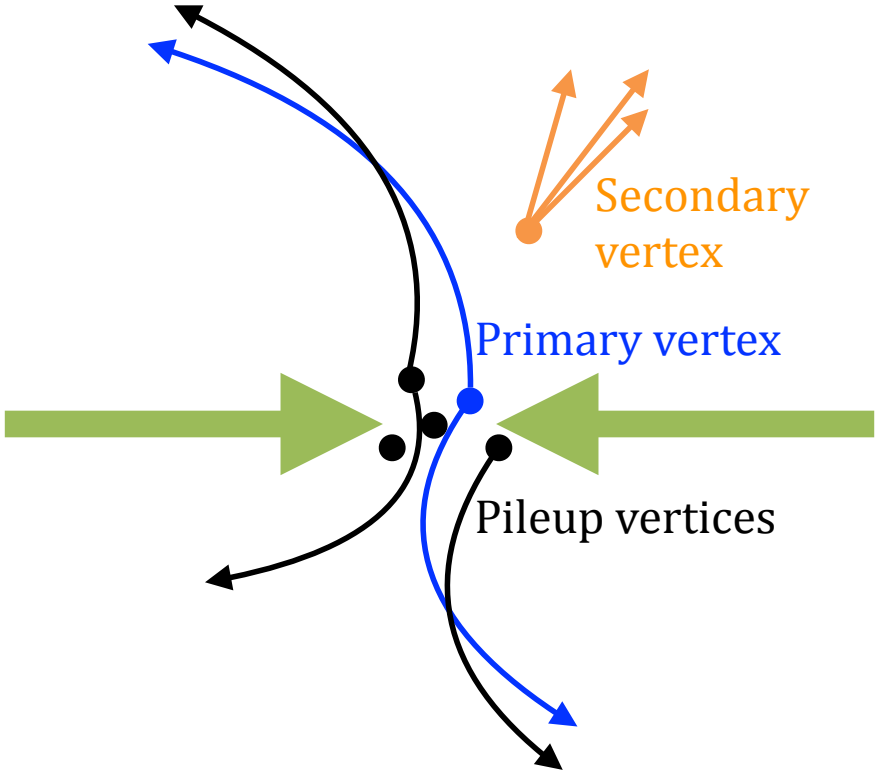
# Machine learning for track reconstruction using ACTS

Corentin Allaire
on behalf of
the Acts ML community

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004761.

# Track Reconstruction

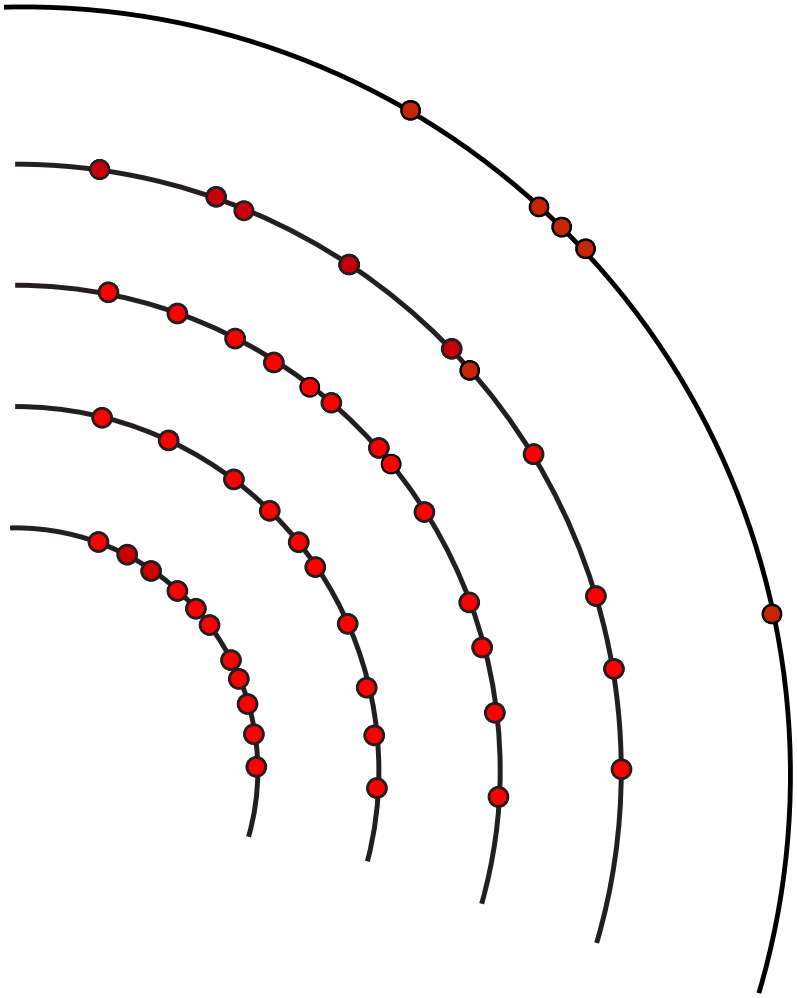





- Reconstruct the trajectory of (charged) particles in detectors based on the hits in the sensors
- Extract track parameters from their reconstructed trajectory
- Reconstruct the interaction vertices

# Track Reconstruction

Space point formation
Clustering



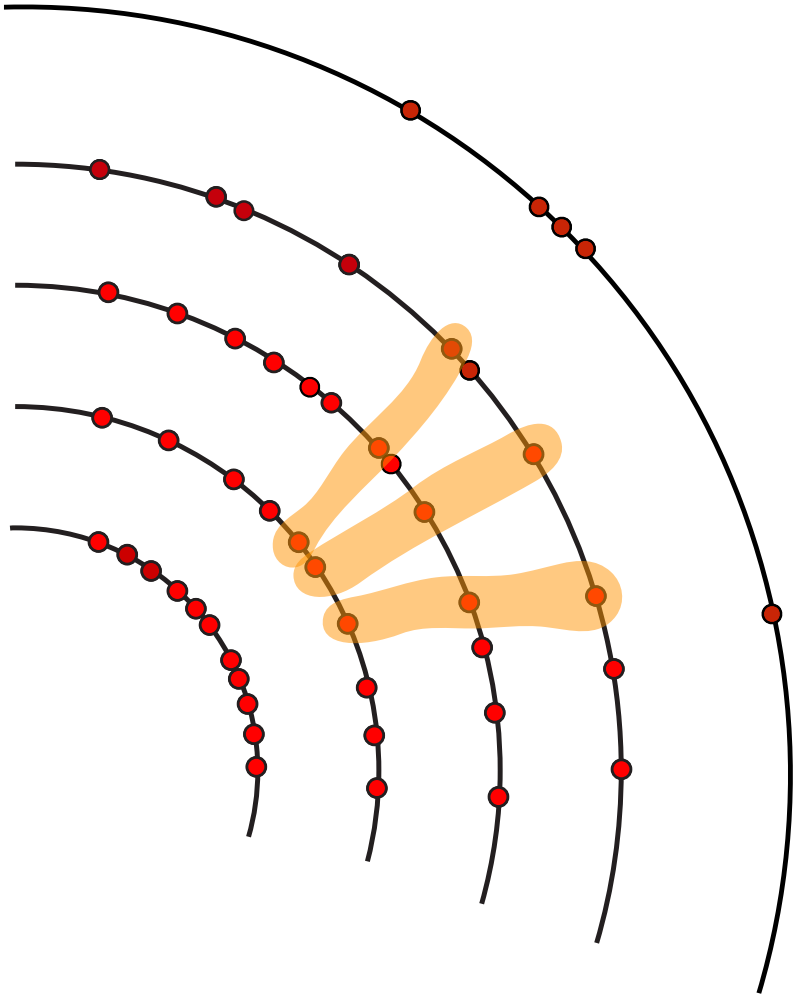
- Hits in the detector are collected to create measurement (space) points

# Track Reconstruction

Space point formation
Clustering

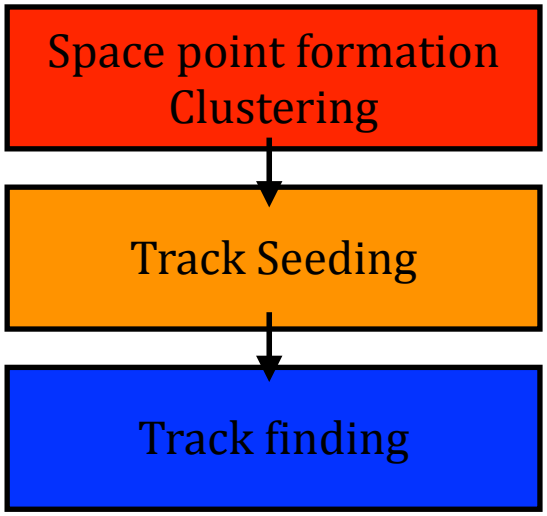
↓

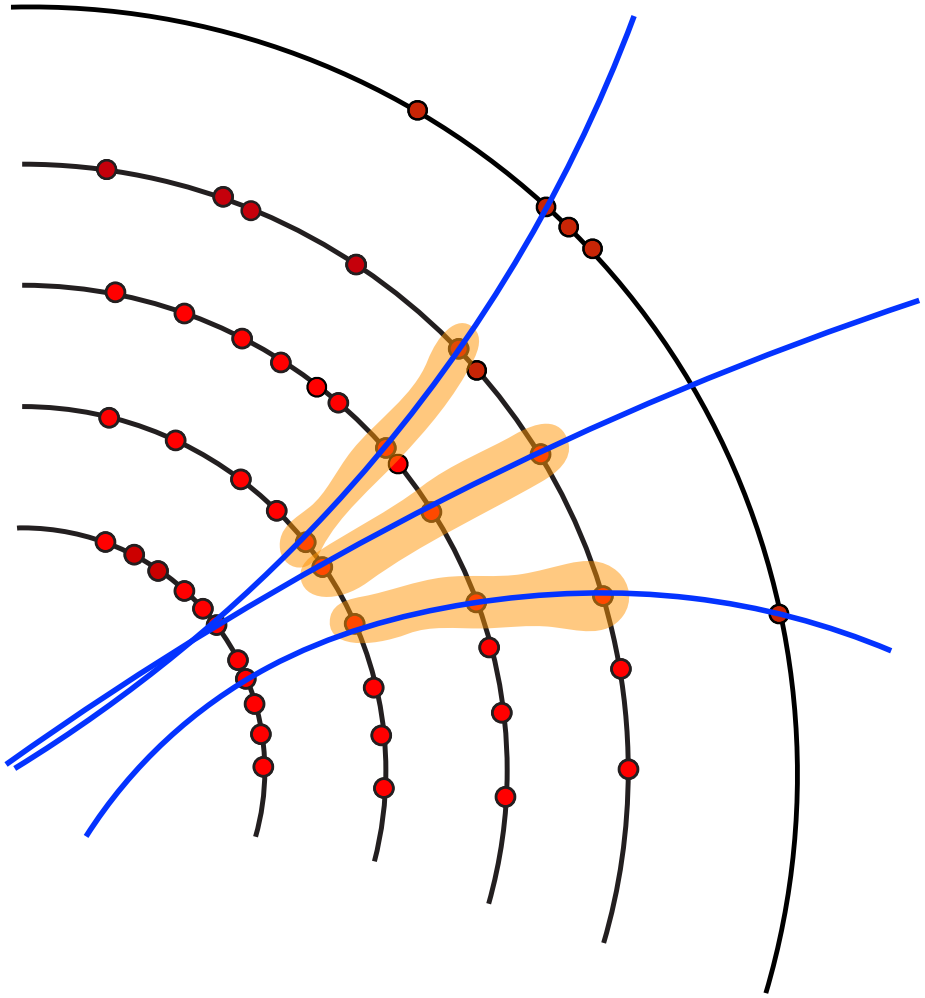
Track Seeding



- **Seeds** are groups of **three measurements** compatible with basic track hypothesis
- Will be extended to create the tracks candidate

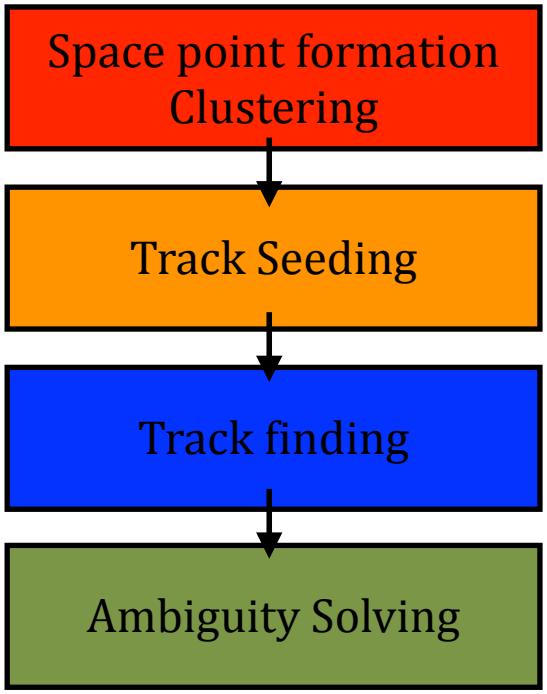
# Track Reconstruction

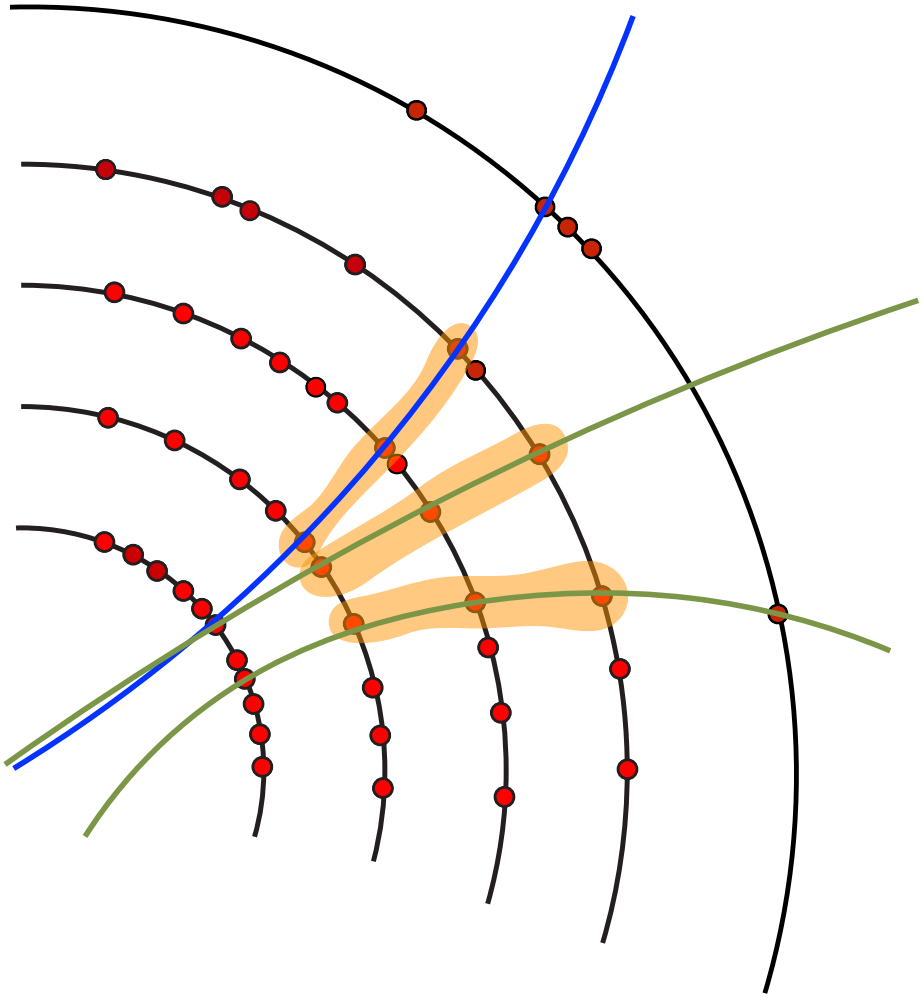


- A combinatorial Kalman filter is then used to build track candidates from the seed
- Compatible measurement are added to the track seed
- More than one **track candidate** can be built from a seed if multiple paths are possible

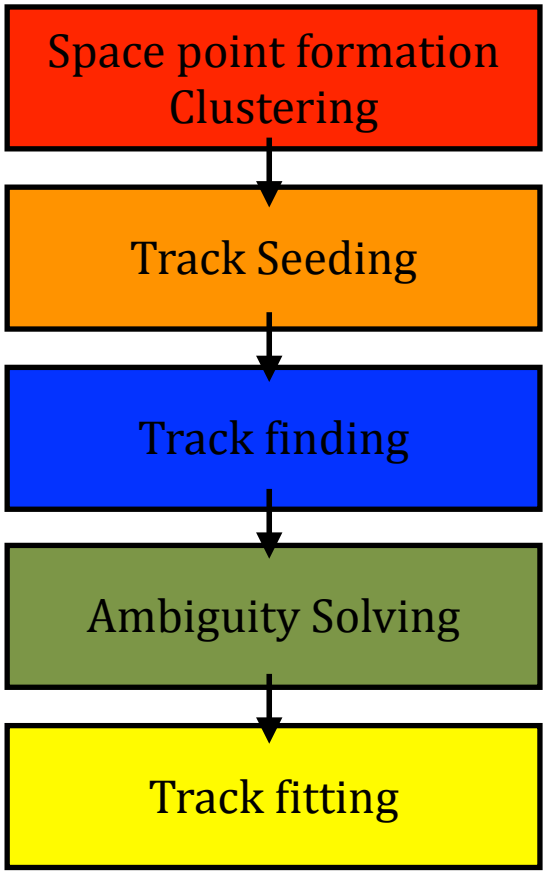
# Track Reconstruction

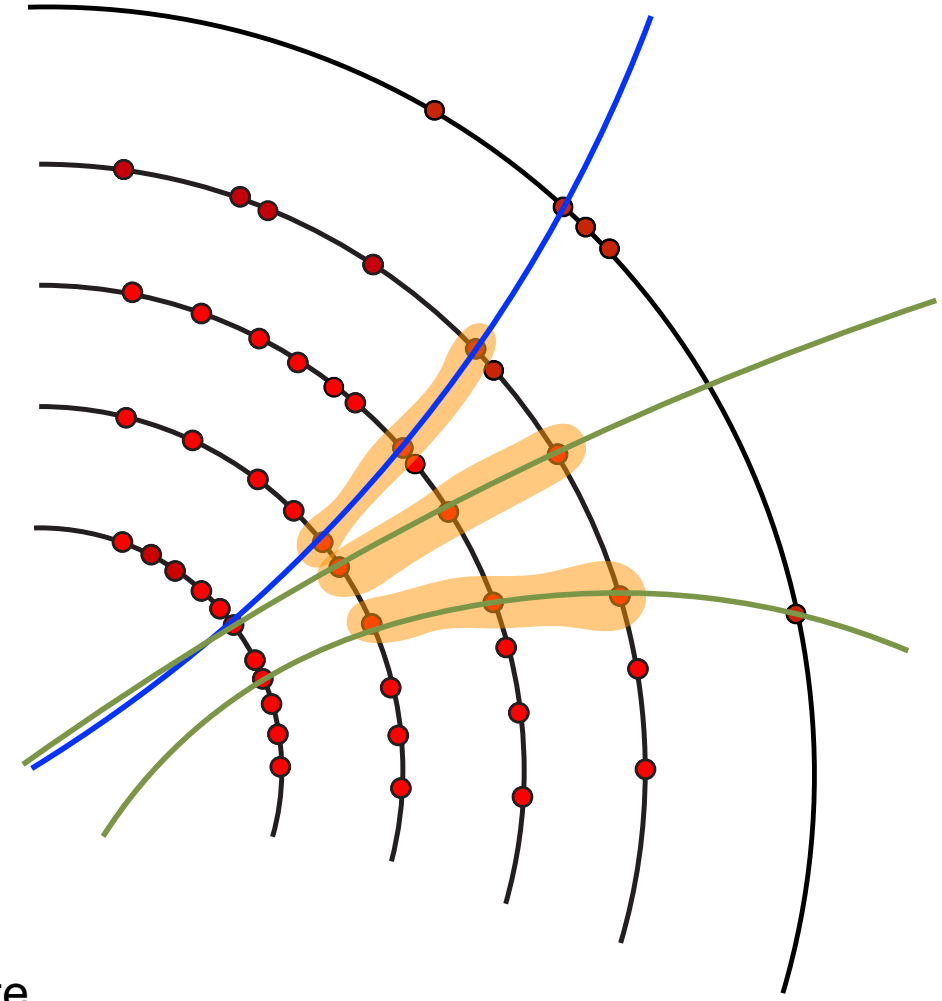


- A score is then associated to each track candidate
- To resolve ambiguity the reconstructed track are kept in descending order of a track score

# Track Reconstruction



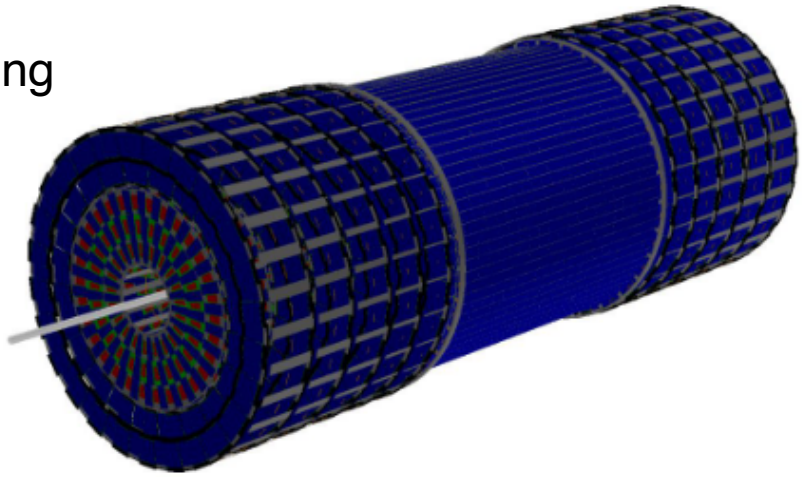
- The tracks are **fitted** using the full detector geometry and the appropriate material -> track parameters

# Acts : A Common Tracking Software

- Open source tracking software : https://github.com/acts-project/acts
- Experiment independent toolkit :
  - ATLAS
  - ALICE
  - BVG
  - FASER
  - LDMX
  - sPHENIX
  - ePIC (EIC)
  - …
- Developed with modern C++
- Features :
  - Tracking geometry description
  - Simple event data model
  - Most track reconstruction algorithms
  - Example framework with python bindings
  - Performance evaluation algorithms

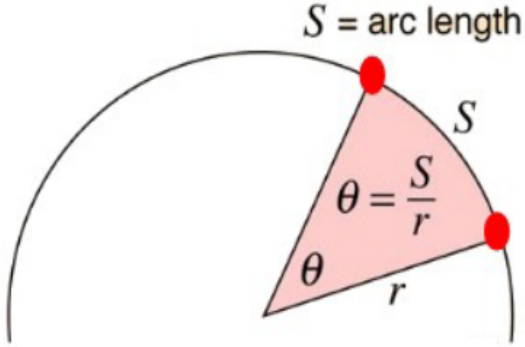
# Acts : A Common Tracking Software

- Provides a testing environment for new tracking algorithms
- Open Data Detector (ODD) :
  - Virtual detector implemented for testing purposes
  - Based on the Track ML challenge
  - Full silicon design (similar to ATLAS ITk)
- Offers a full tracking chain -> Used to evaluate the performances of our algorithms
- Great environment for developing and testing new machine learning based tracking algorithm



# Hashing for hits selection

- Track finding : Combinatorial problem -> scales quadratically with the pileup (PU)
- Hashing : perform the reconstruction on subsets of hits (buckets) -> reduce the PU dependency
- Buckets created using Approximate nearest neighbours search (Annoy)
- For a given metric, return the nearest neighbours
- Metric : angular distances
- Other metric possible : Metric learning ?

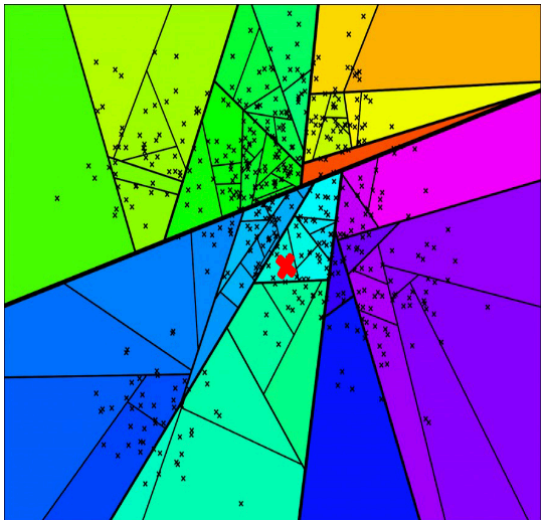



$$\theta = \frac{S}{R}$$

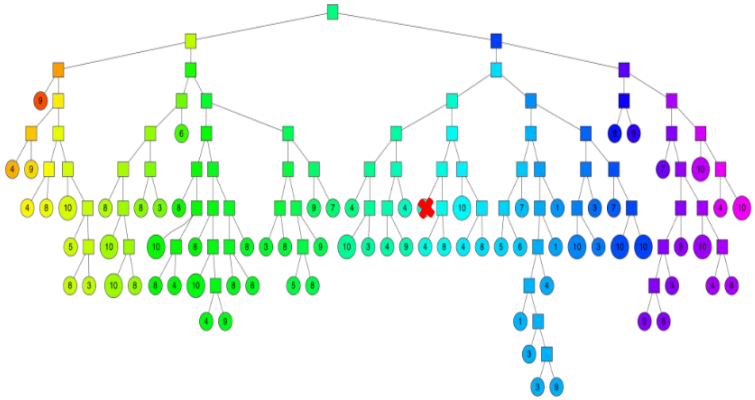
S = Distance travelled (on the arc)

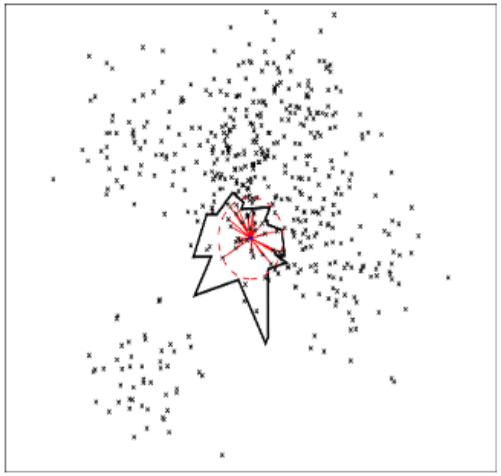
R = Radius of the circle

# Hashing for hits selection



- Annoy : Build trees by dividing the space
- Collect hits in the same leaf and check leaf from close split
- Perform in parallel the search in many trees and get the union
- Finds nearest neighbours

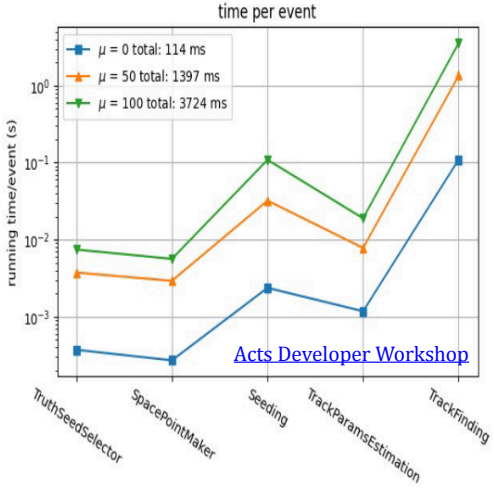


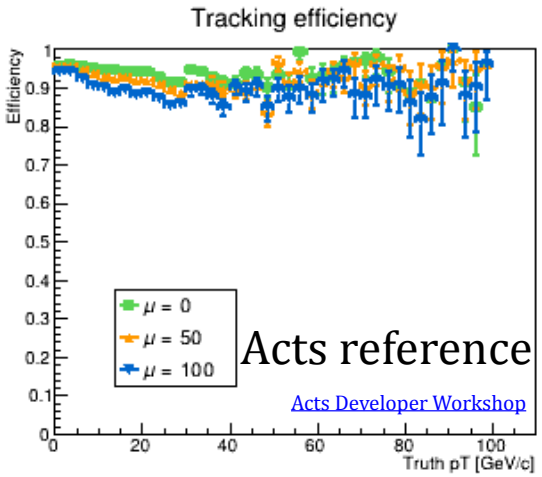


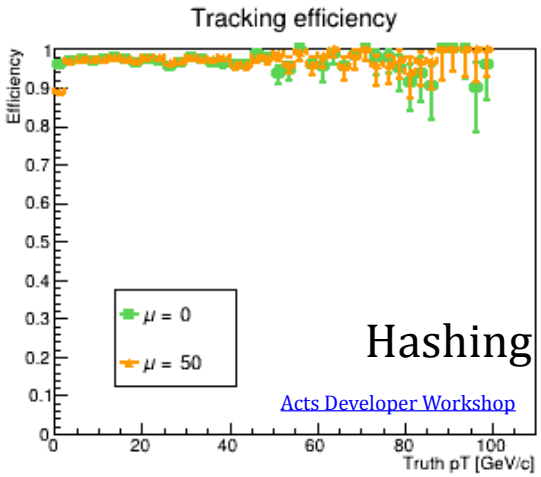
# Hashing for hits selection

- Similar performances to the CKF can be achieved with hashing (some issue at low pt -> larger bucket?)
- Creates a lot of seeds -> slow down the seeding
- More work needed to improve the chain

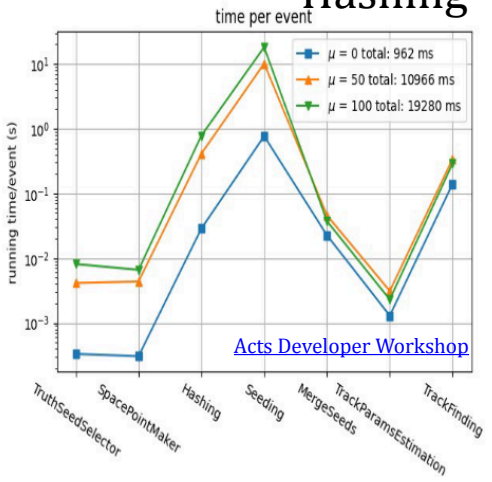
Acts reference




Hashing



# Parameter auto-tuning

- Most track reconstruction algorithm use multiple parameters to account for experimental condition :
  - Detector geometry
  - Material Configuration
  - Pile-up
  - Center-of-mass energy
  - Many other factors

- Currently -> Hand-tuned :
  - Time-consuming (need expert)
  - Need to be retuned when condition change

- Idea -> Auto-tuning :
  - Less time-consuming (trade CPU time for human time)
  - Easy retuning
  - Allow more granular tuning

- Tried different optimisation :
  - Oríon: Black box optimisation framework with different optimisation techniques
  - Optuna: Open source software for automatic hyper-parameter search
  - Many others …

Oríon

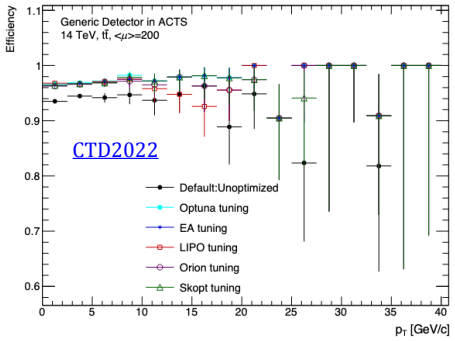
OPTUNA

# Parameter auto-tuning

- Seeding goals : efficiency and the track duplication rate (slow the reconstruction)
- Performance evaluated after the track finding

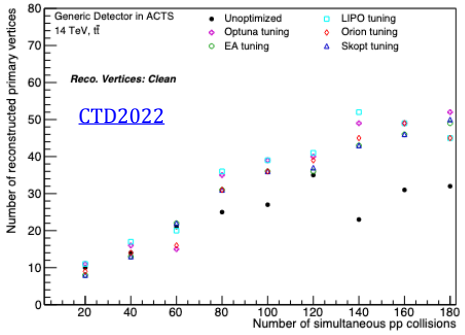
$$\text{Score Function} = \textit{Efficiency} - \left(\textit{FakeRate} + \frac{\textit{DuplicateRate}}{K} + \frac{\textit{RunTime}}{K}\right),$$

(K = 7 for all algorithms)

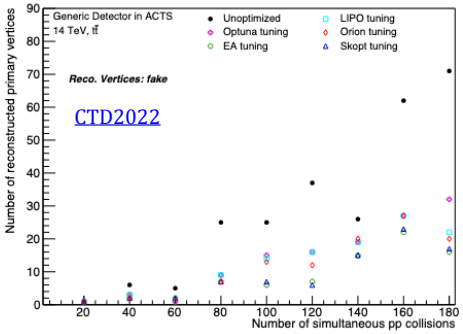



- Vertex reconstruction goals : efficiency and cleanness of the vertex

$$\text{Score Function} = (\textit{Eff}_{Total} + 2\textit{Eff}_{Clean}) - (\textit{Merged} + \textit{Split} + \textit{Fake} + \textit{Resolution})$$

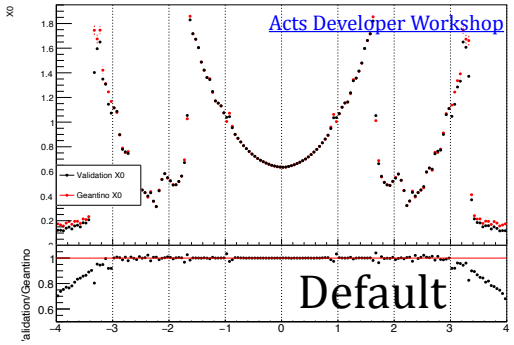


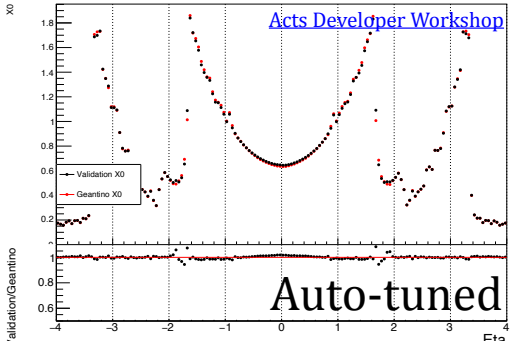

- Both show clear improvements with respect to the unoptimised solution
- Scoring function needs more work
- Available in ACTS, can be used on any detector



# Parameter auto-tuning

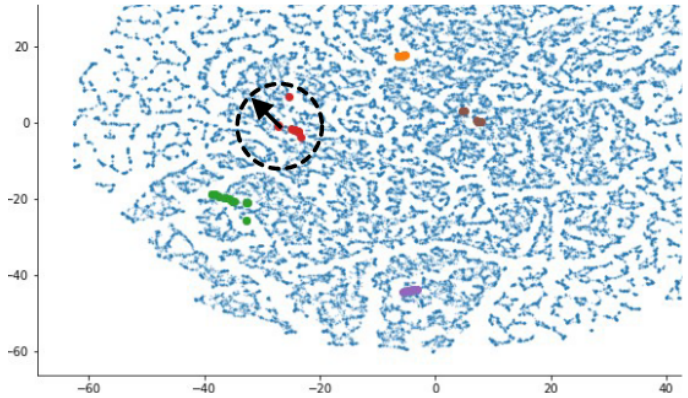
- Material Mapping goals : create a simplified material representation for the navigation
- Approximate effects of detector materials by projecting them onto binned surfaces
- Use Orion to optimise the binning on each surface
- Score : $\frac{1}{Bins}\sum_{bin} variance_{mat} \times (1 + \frac{1}{hits_{bin}})$
- Works great on the ODD, some small issue in the barrel (change the score ?)
- Also available in Acts

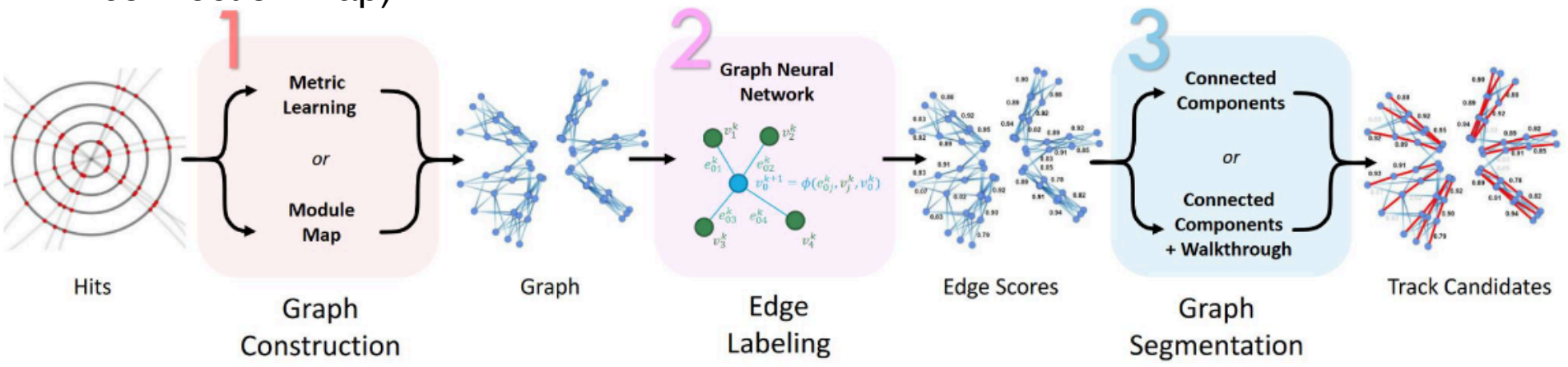



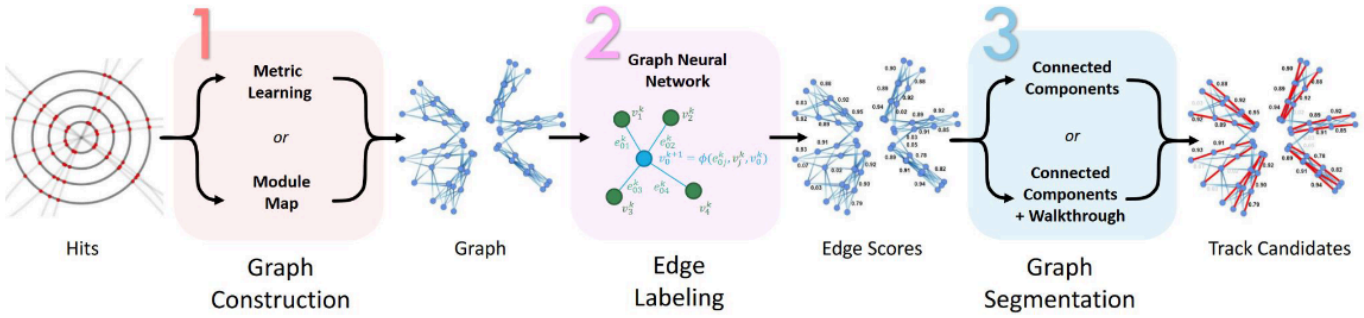
# GNN for track finding

- Track finding scales quadratically with the PU
- Interpret the hits in the detector as a graph -> Use Graph Neutral Network (GNN) to reconstruct tracks
- Embedding : Use all the hits in the detector to build a graph
- Filtering : Neural-Network predicts if nodes should be connected (can also use a connection map)



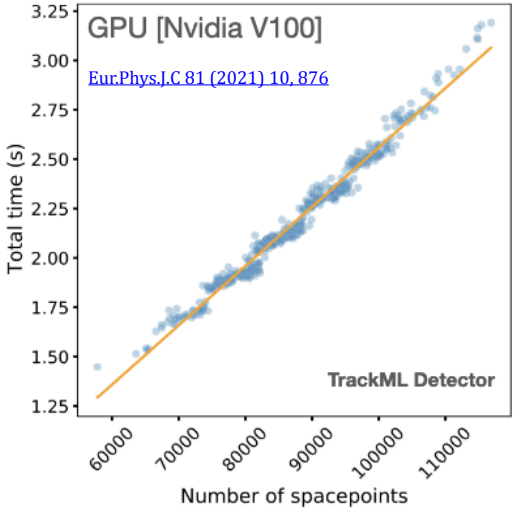
# GNN for track finding

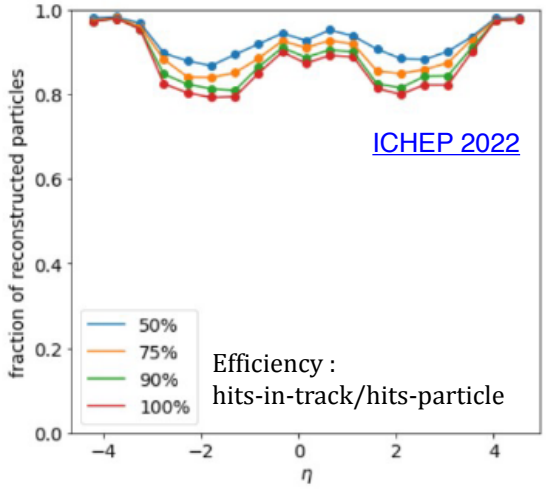


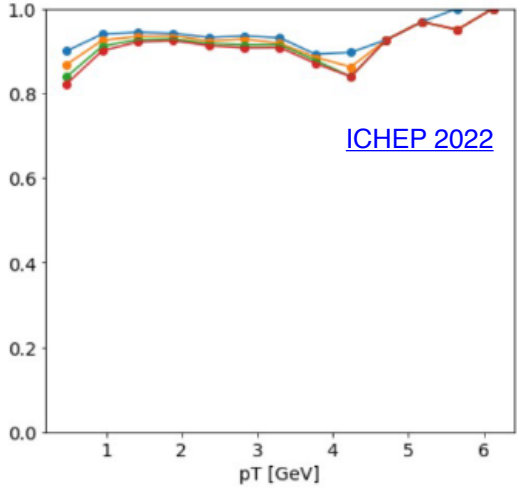

- EDGE labelling : compute a score for each edge
- Message-passing graph neural network : pass variables from nodes to edge then from edges to nodes
- After multiple iteration, compute scores for the edges and use them to remove the bad edges
- Graph Segmentation : track reconstruction from the connected nodes

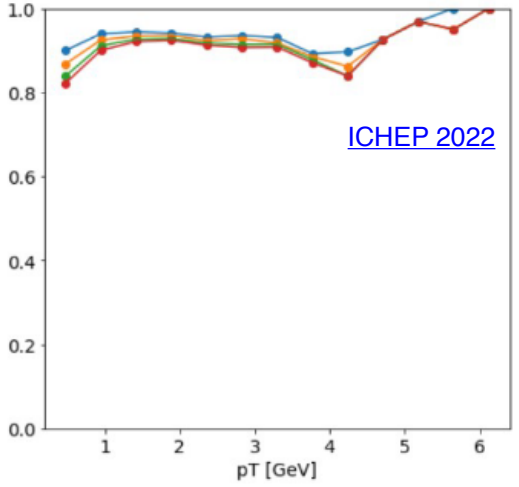
# GNN for track finding

- A good efficiency can be achieved using this method
- Tested and fully implemented in Acts and usable by any experiment
- Computing performance scales linearly with number of hits (and not quadratically)
- Real comparison of computing performance and reconstruction efficiency with standard CKF planned soon



# Conclusion

- ACTS is an Open Source tracking toolkit
- Provides virtual ODD detector to test different algorithm
- Great environment to develop new Machine learning based solution for track reconstruction :
  - Hashing for track reconstruction
  - Parameter auto-tuning for tracking algorithm (available in Acts)
  - GNN for track finding (also available)

# BACKUP