Ranking-based machine learning for track seed selection in ACTS

Corentin Allaire
Hadrien Grasland
David Rousseau
Françoise Bouvet

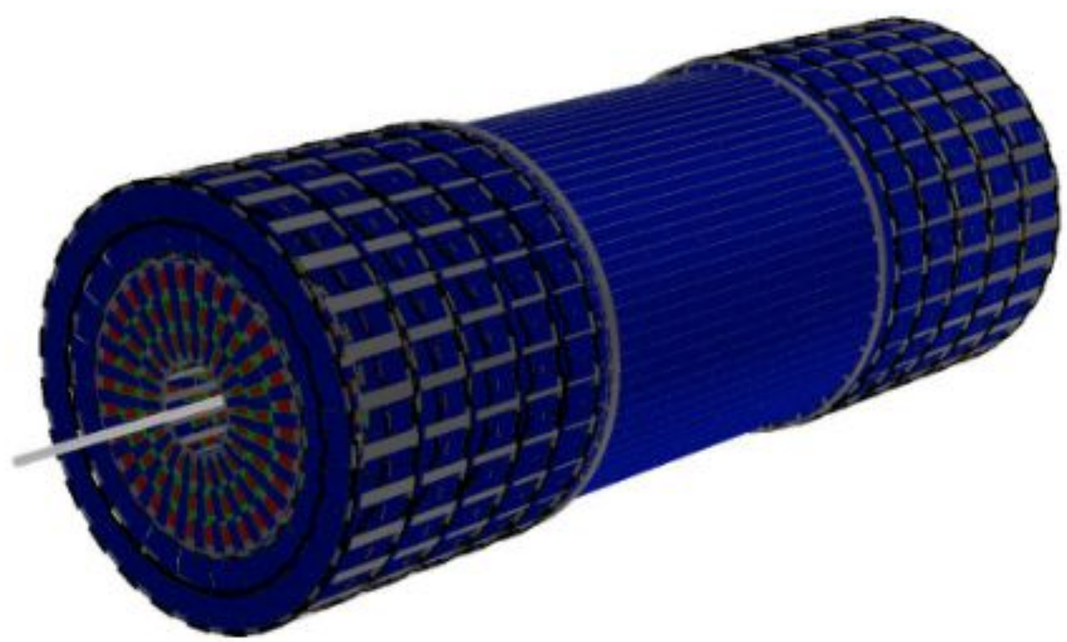
With support of ATRAPP ANR-21-CE31-0022

Acts and Open Data Detector

Open source tracking software:
https://github.com/acts-project/acts

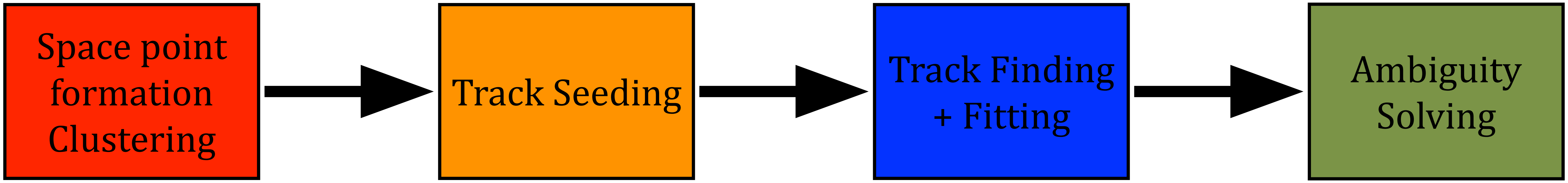
Testing environment for new tracking algorithms:

- Open Data Detector (ODD) :
  - Virtual detector: benchmarking
  - Based on the **Track ML challenge**
  - Full silicon design (similar to ATLAS ITk)
  - Realistic detector material



- Performance benchmarks:
  - Full tracking chain
  - Performance writer
  - Useful for **machine learning** developing/testing

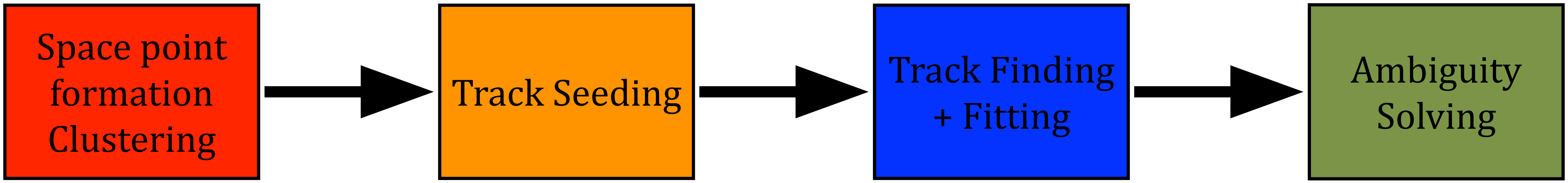
The Classical Tracking Chain



Four main steps:

- **Space point formation**: Create measurement points (hits) from detector data
- **Track Seeding**: Find triplets of hits compatible with particle hypothesis to serve as starting points for trajectories
- **Track Finding**: Starting from the seed, find the particle trajectory in the detector (and reconstruct their associated parameters)
- **Ambiguity Solving**: Cleaning step, remove extra duplicated and fake tracks

The Classical Tracking Chain



When considering the ODD with ttbar events, pille up 200, we have (per events):

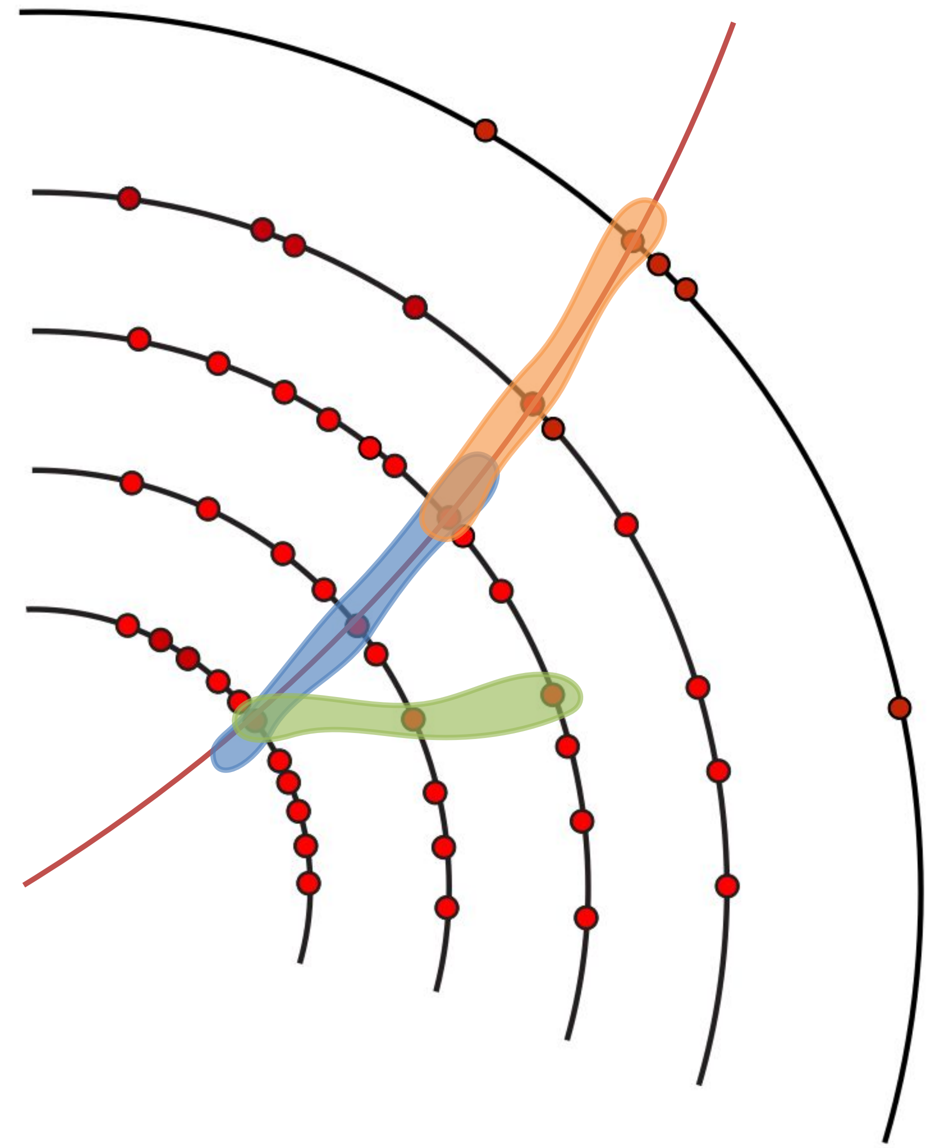
- **Seed**: ~100k seed per events
- **Tracks (after finding)**: ~ 10k Tracks
- **Tracks (after solving)**: ~ 800
- **Total truth particle**: ~ 800

Where are all these extra seeds coming from?

Three types of seed

The seed can be sorted into 3 categories:

- **Good seed**: Seed corresponding to **truth particles**; their 3 hits are all associated with the same truth particle **(~ 1k)**
- **Duplicated seed**: Same as the good hits but leads to worst quality tracks (fewer hits, larger Chi2 …), can be ranked based on track quality **(~24k)**
- **Fake seed**: Seed with hits coming from more than 1 truth particles will lead to a fake trajectory **(~77.5k)**



Each fake and duplicated seed will be reconstructed by the CKF (track finding) afterwards ➔ Huge time loss

Removing seed early can help us speed up the tracking chain

Machine learning based Seed filtering

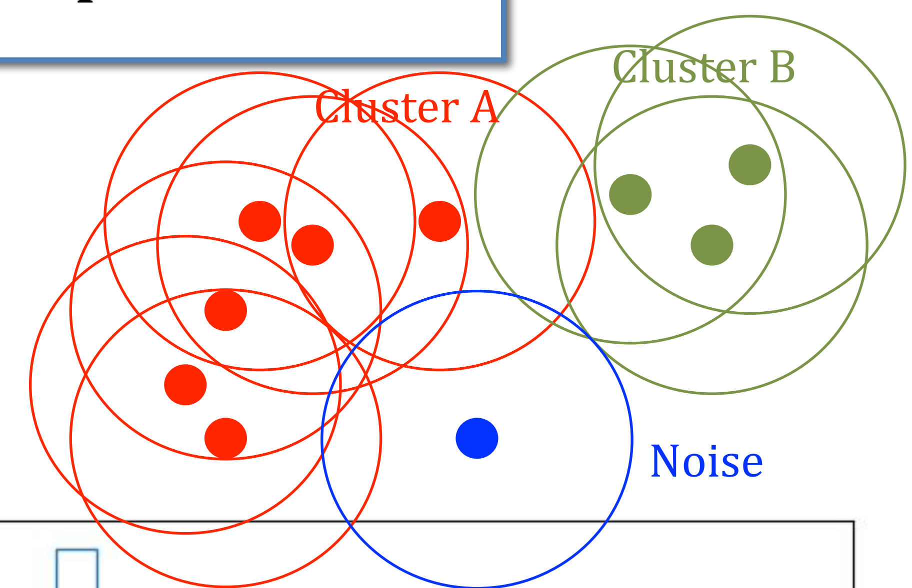
Cluster seed together → Network to determine the best seed per cluster

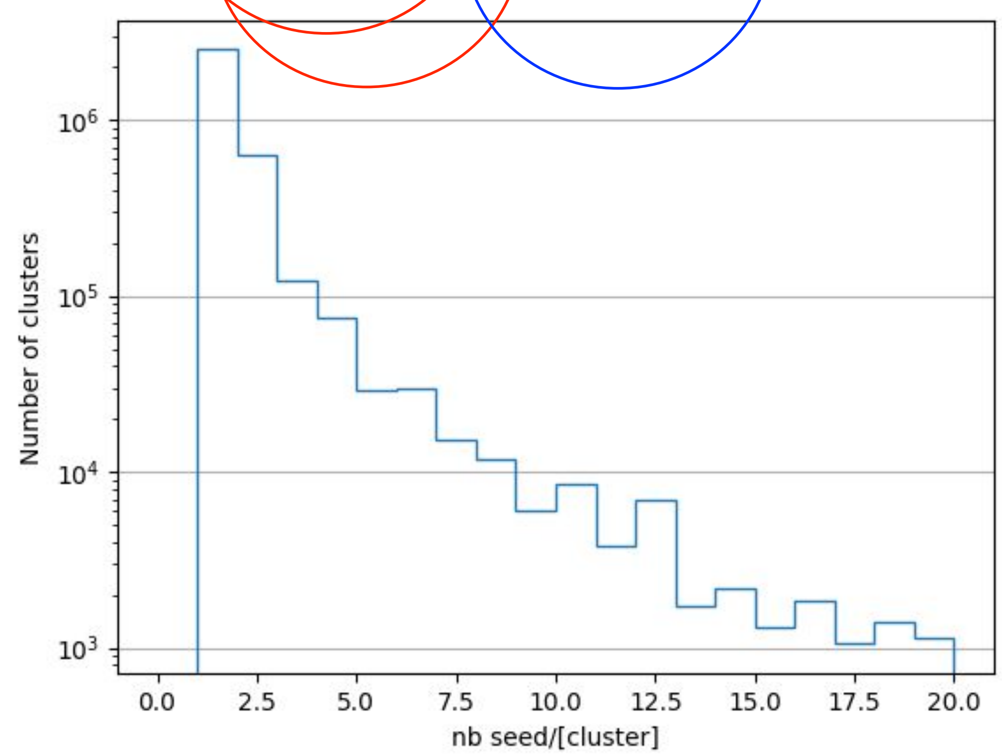
Filtering inspired by the ranking-based ambiguity solver shown at CHEP23:

- Remove the **fake** and **duplicate** at the same time
- Uses a **DBScan clustering** algorithm to aggregate seeds that appear to come from the same **particle**
- 4D DBScan using 4 dimensions:
  - ϕ, η, Z_0/50 and pT
- Two DBScan parameters
  - ε=0.07 ; $\mathrm{Min}_{\mathrm{sample}}$=2

Max distance seeds

Min number of seeds per cluster



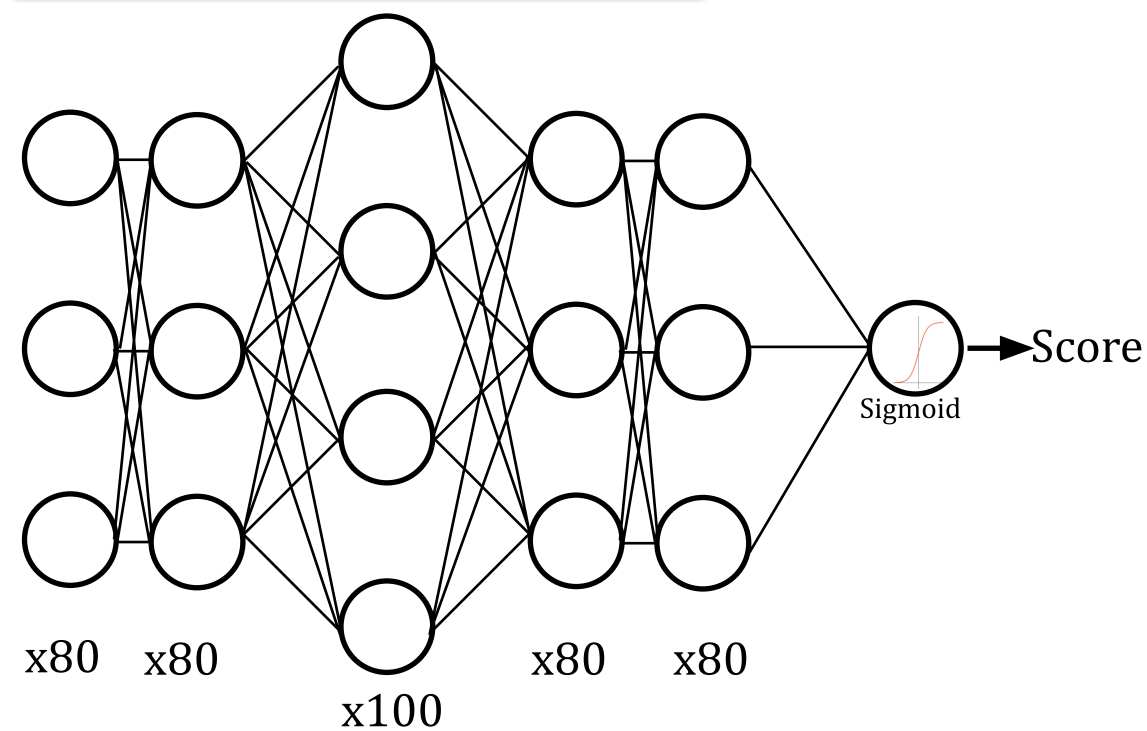


Machine learning based Seed filtering

Cluster seed together → Network to determine the best seed per cluster

Neural Network: Score each seed, keep the **highest score** per cluster, remove the **lowest scores**:

- 5 hidden layers MLP (80, 80, 100, 80, 80)
- Use **14 seed variables** as input
- **One score per seed ➡ Select the best one in each cluster**



Input parameters :

- Pt
- Eta
- Phi
- Z0
- Seed Quality
- Space point 1 (x,y,z)
- Space point 2 (x,y,z)
- Space point 3 (x,y,z)

- Available in Acts via **Onnxruntime**
- Hyper-parameters of the networks are not fully tuned; our lab got flooded 2 weeks ago

Ranking Neural Network

Goal: one score per seed, highest for the **good** one

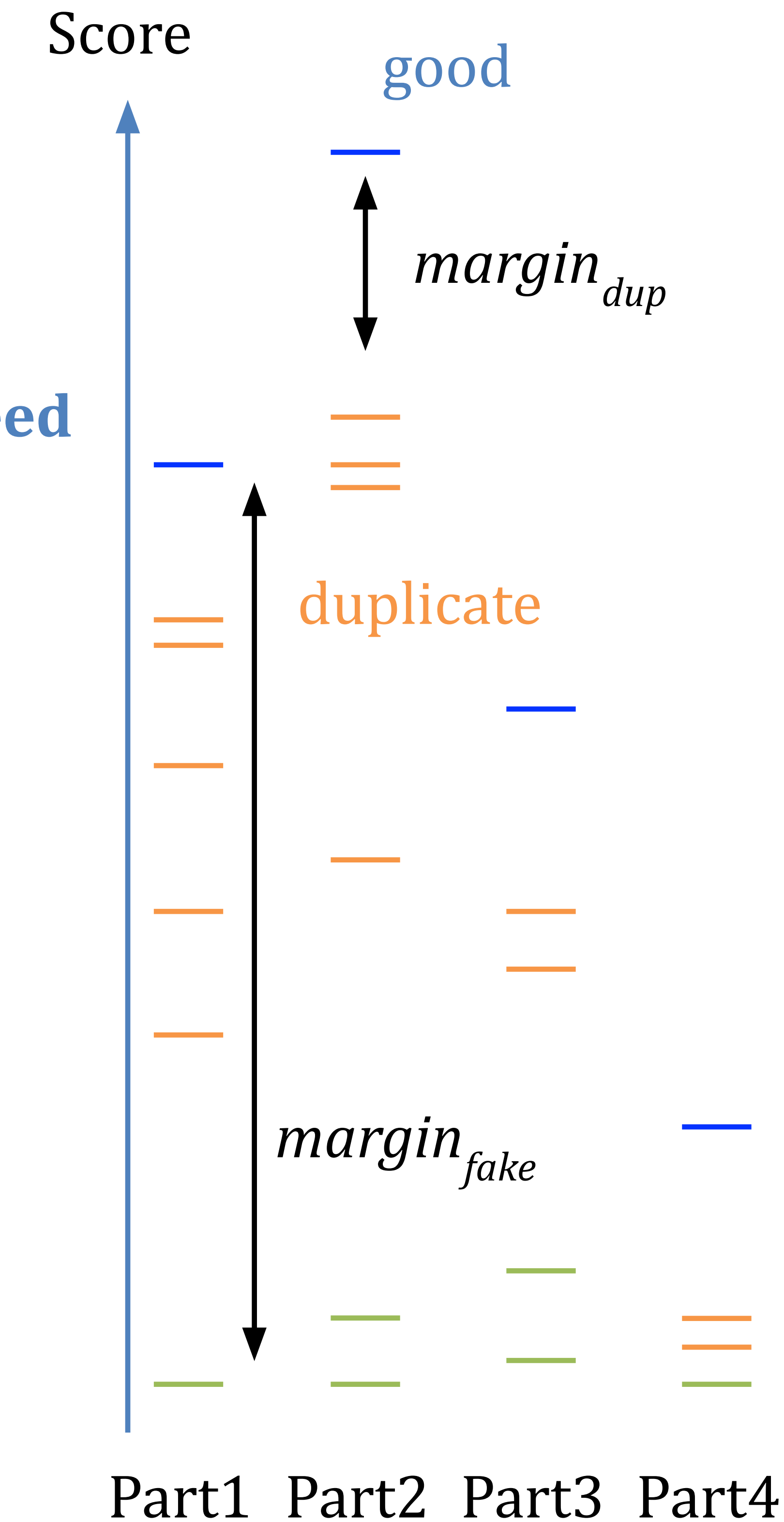
- Training without clustering ➡ use **truth** matching instead
- Compute one loss per **truth particle**
- Use a **Margin Ranking Loss :**

Good seed

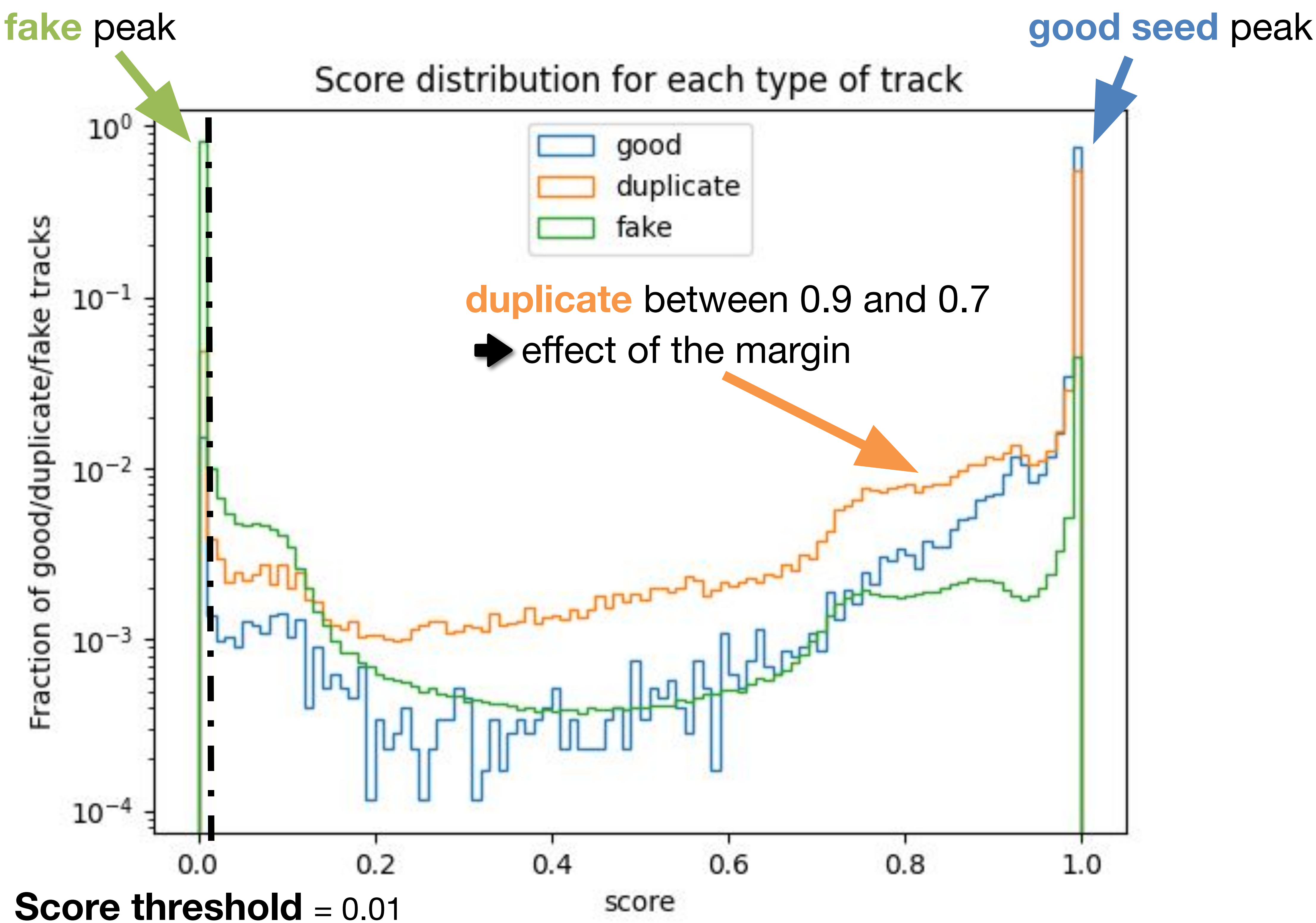
$$loss_{part} = \frac{1}{N_{tracks}} \sum^{tracks} max(0, x - y + margin)$$

duplicate/ **fake**

- x: seed score; y: **good** seed score;
- Uses 2 different margins :
  - $margin_{dup}$: between **good** and **duplicated** = 0.2 + 0.01✖ rank
  - $margin_{fake}$: between **good** and **fake** = 0.9



Score distribution



Score threshold = 0.01

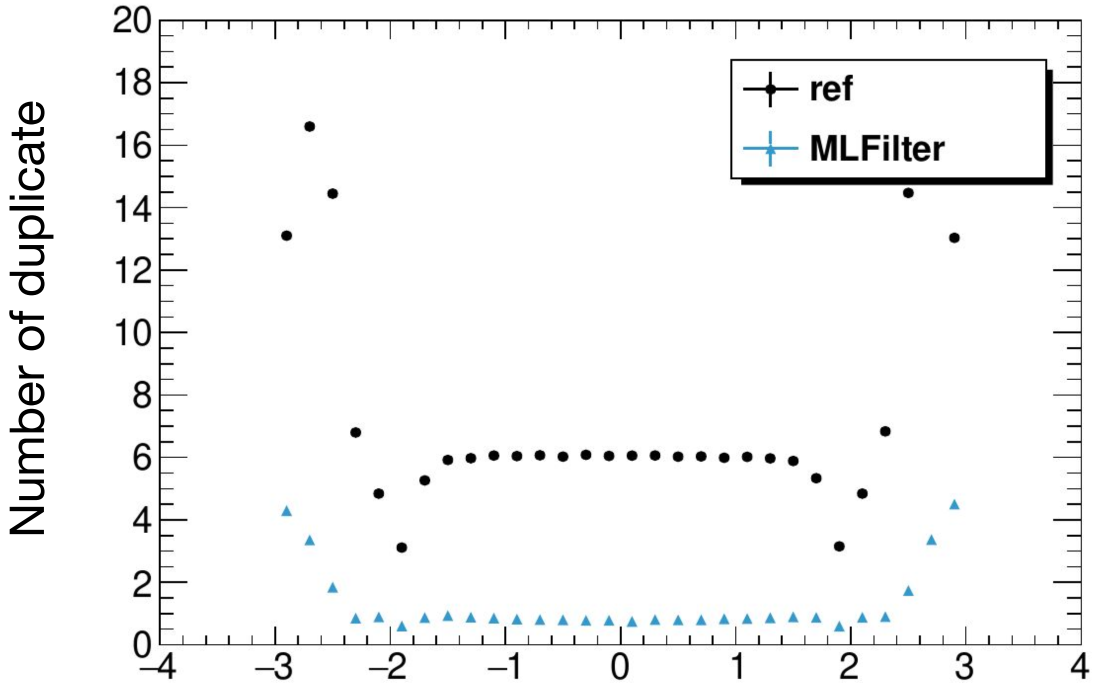
Seed Efficiency

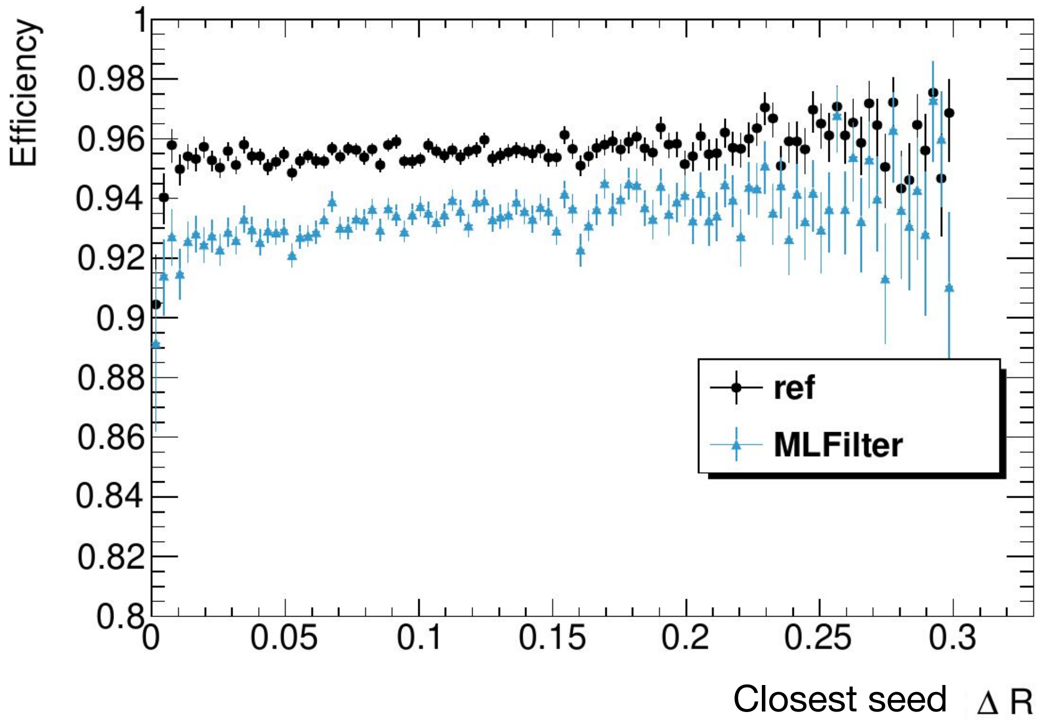
- Performances studied at the level of the **seeds**
- **Efficiency (good seed)**: Fraction of the original good seed still present
- **Efficiency (truth matched)**: Fraction of the original truth particles still matched to at least 1 seed (good or duplicate)
- Reduction of the number of seeds by a **factor of ~10** with a minor drop in efficiency

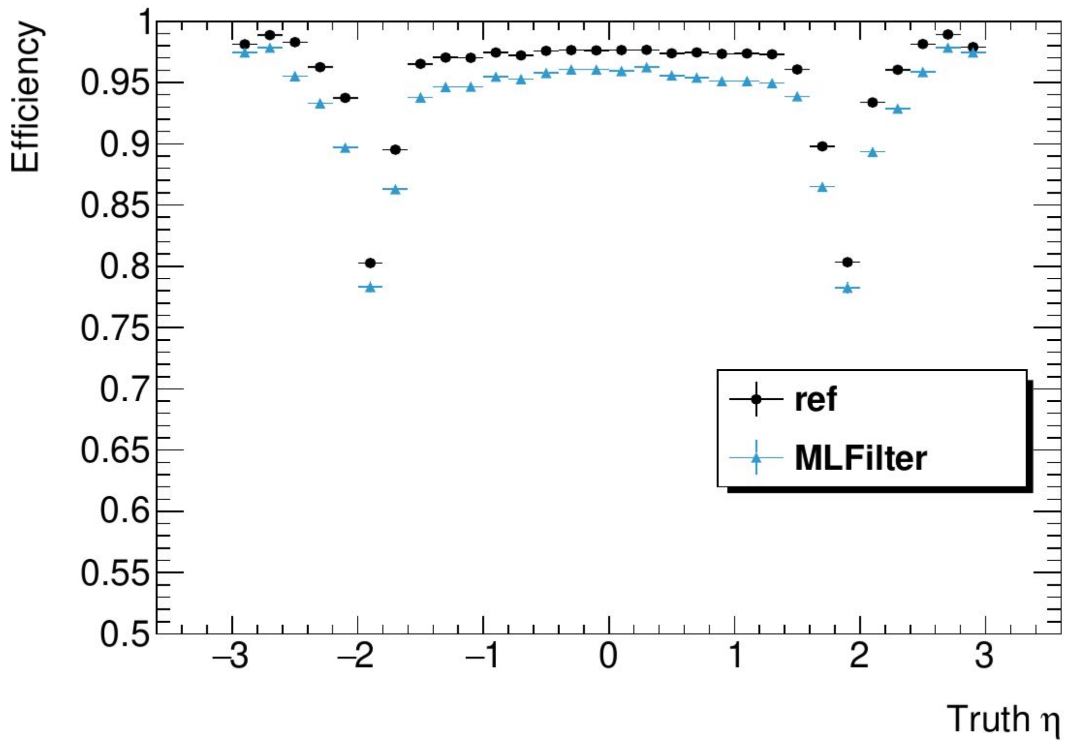
	Number of seed	Efficiency (good seed)	Efficiency (truth matched)	Duplicate Seed	Fake Seed
Default Seeding	$109{\times}10^3$		100 %	$5.5{\times}10^3$	$105{\times}10^3$
Default + Clustering	$54{\times}10^3$	44.0 %	99.2 %	$1.1{\times}10^3$	$52{\times}10^3$
Default + Clustering + Threshold	$12{\times}10^3$	43.4 %	98.7 %	$1.1{\times}10^3$	$10{\times}10^3$

Seed Efficiency

- Reduction of the **duplicate rate** by a **factor of ~5**
- Effect on the efficiency **uniform** through the detector
- Impact on the efficiency **independent** of the local **seed density** (the clustering works properly)





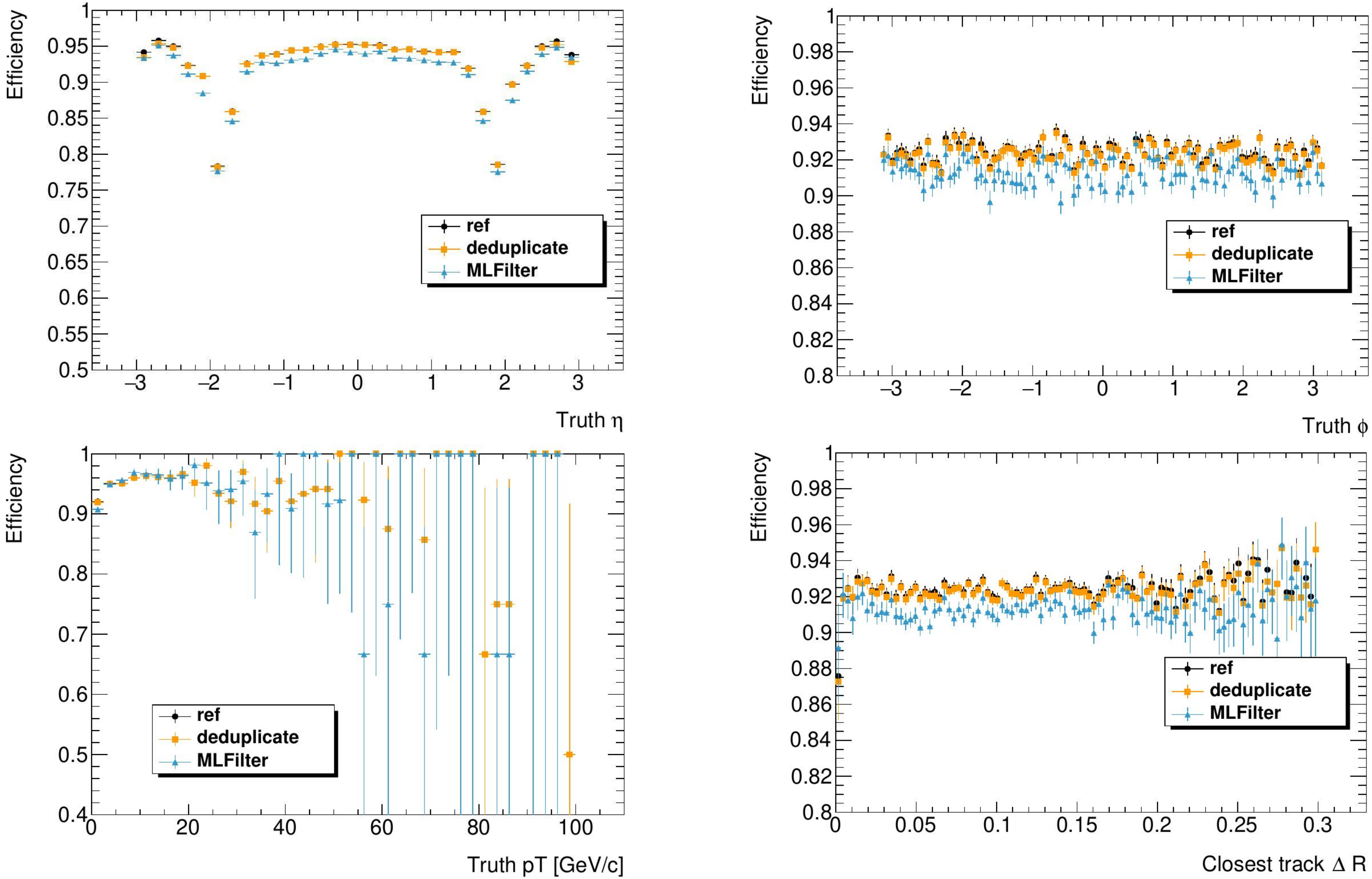


Track Efficiency

- Effect on the **ML Seed filtering** tested on the **full tracking chain** (track remaining after the ambiguity solver)
- Acts implement a **seed deduplication** as part of the CKF to remove duplicates
- Efficiency computed with respect to the number of **truth particle**
- Seed include Seed Filtering + CKF + Ambiguity Solver
- Minor decrease in performance, **speed up by a factor of ~2**

	Efficiency	Duplicate Rate	Fake Rate	Speed [s/event]
Default	92.6 %	2.5×10^{-3} %	0.22 %	7.2
Default + Seed deduplucation	92.5 %	1.5×10^{-3} %	0.22 %	3.4
Default + ML Seed Filter	91.5 %	2.1×10^{-3} %	0.17 %	1.5

Track Efficiency



Summary

- **ML Seed Filter**: Combine **clustering** and a **ranking based** neural network
- **~2 times faster** than the classical one and similar performances
- **Available right now in Acts** with an example to run it with the ODD, can be tested by any experiment using Acts

Outlook

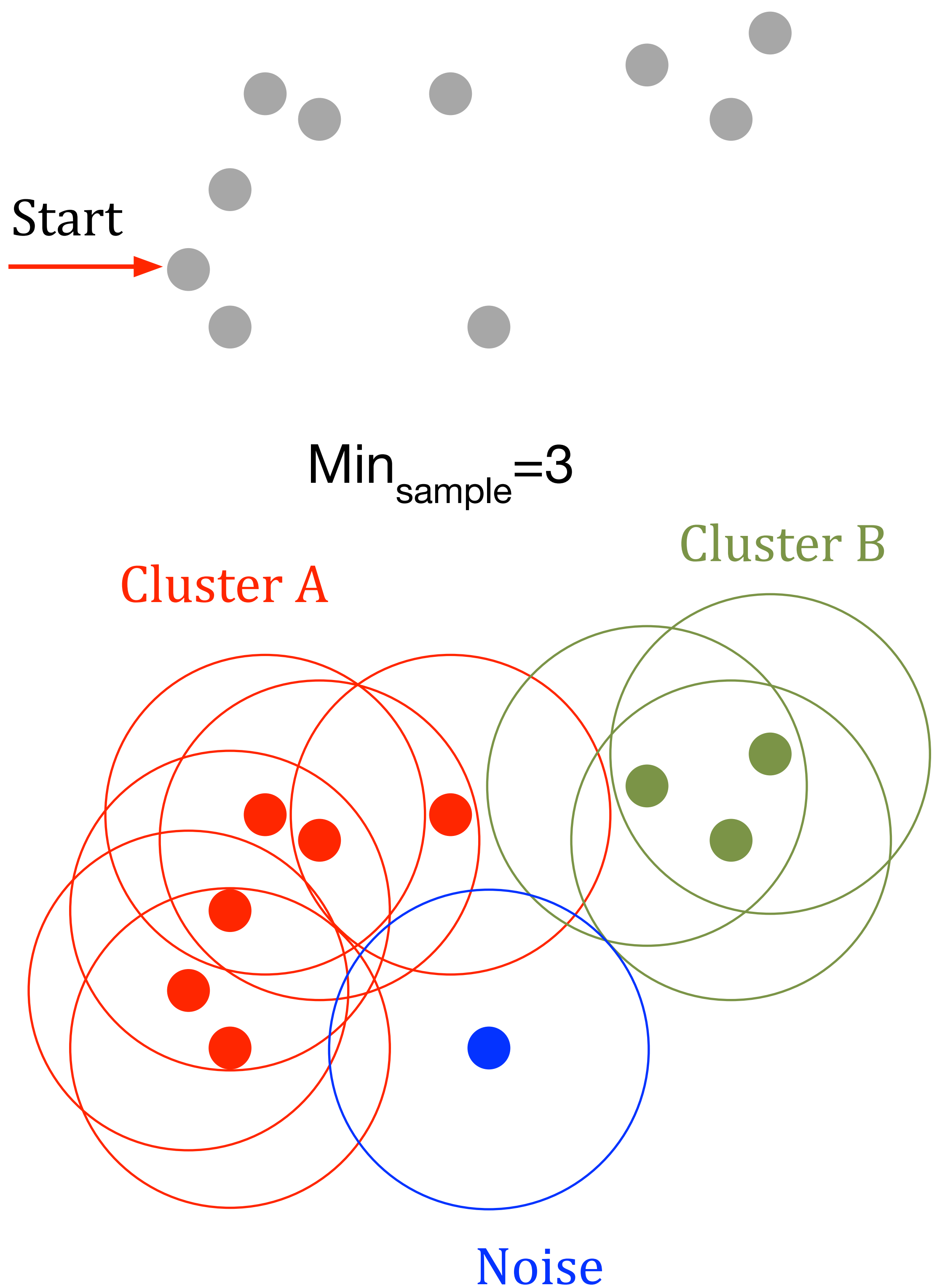
- Fine-tuning needed (I am waiting for the river to leave our lab alone)
- Only removes 80% of the duplicates, could use **metric learning** to project the seed in a space where clustering is easier
- Testing planned on real detectors


David Rousseau

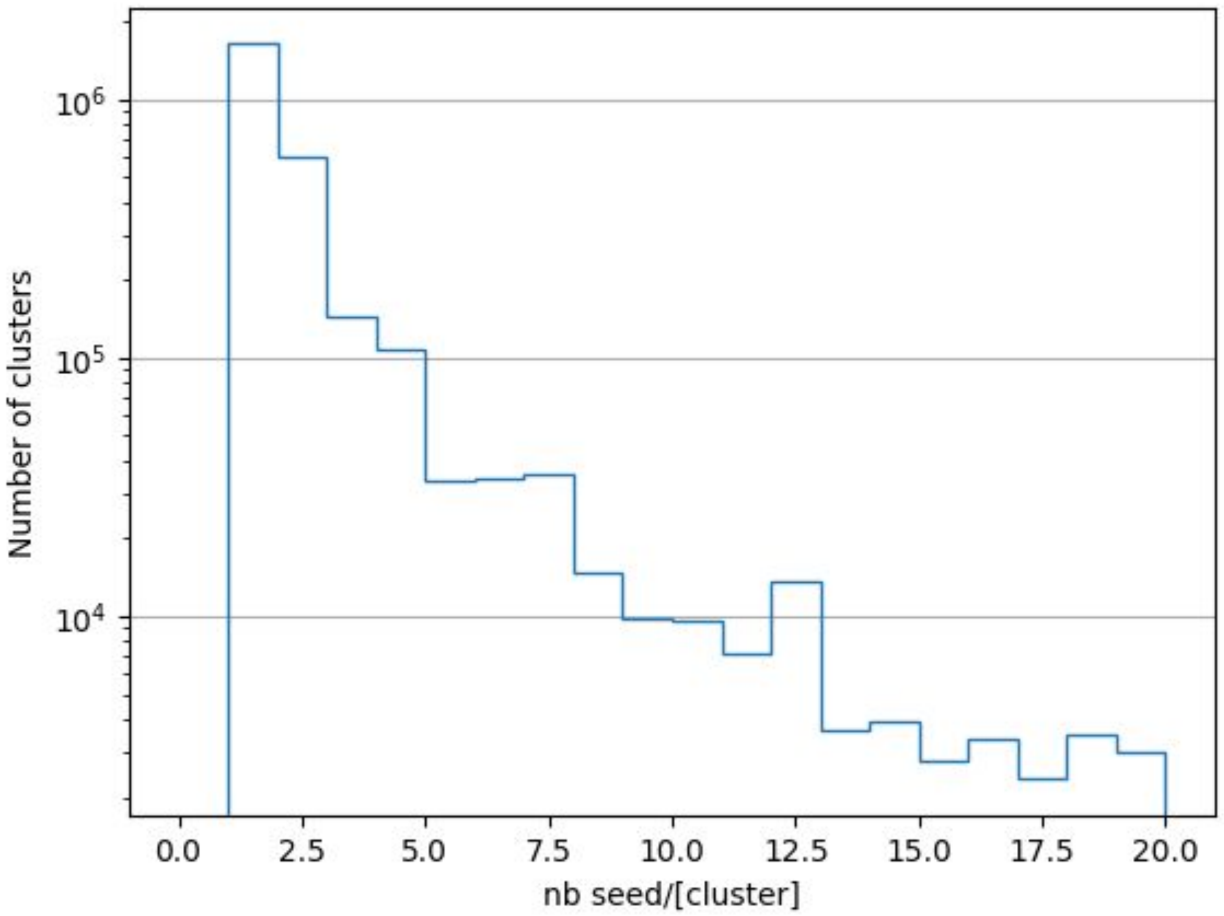
BACKUP

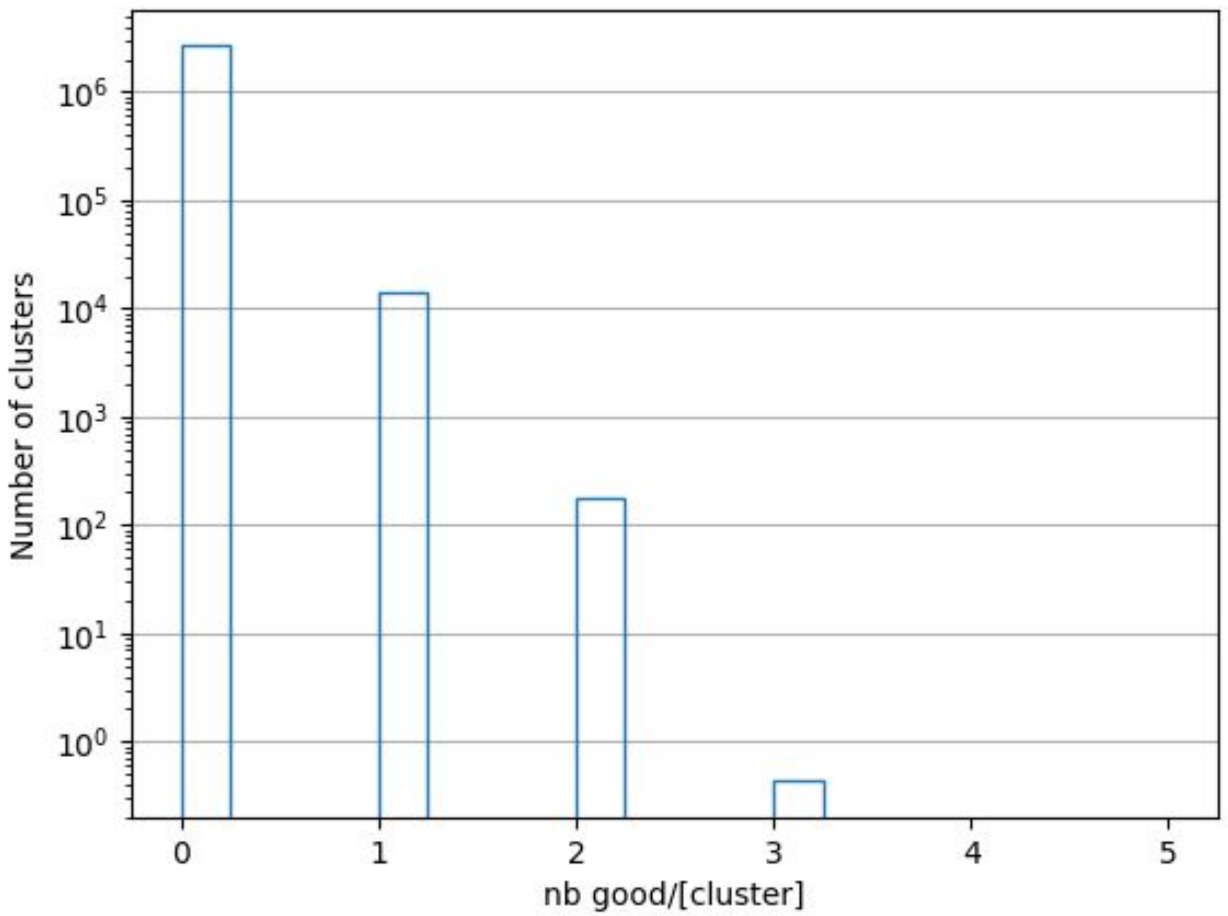
DBScan clustering

- Idea : 1 cluster = 1 truth particle
- Reimplemented in Acts
- Clustering based on **data density**
- Use 2 parameters :
  - ε: Max distance between neighbour
  - Min_{sample}: Min number of elements per cluster
- More than Min_{sample} neighbour ➡ Create a cluster
- For each element of the cluster, do the same ➡ extend the cluster
- In the Ambiguity Solver :
  - distance in (η,ϕ); ε=0.07 ; Min_{sample}=2



Cluster components





Cluster components