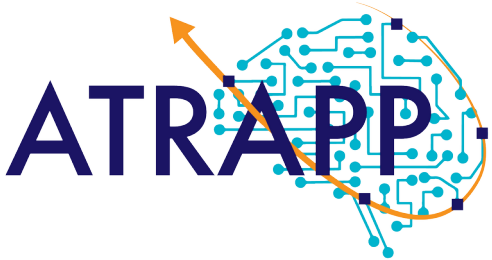


Tracking with Hashing

First year PhD Seminar

23th June 2023

ATRAPP team:
Jessica Levêque
Sabine Elles
Florencia Castillo

Jeremy Couthures

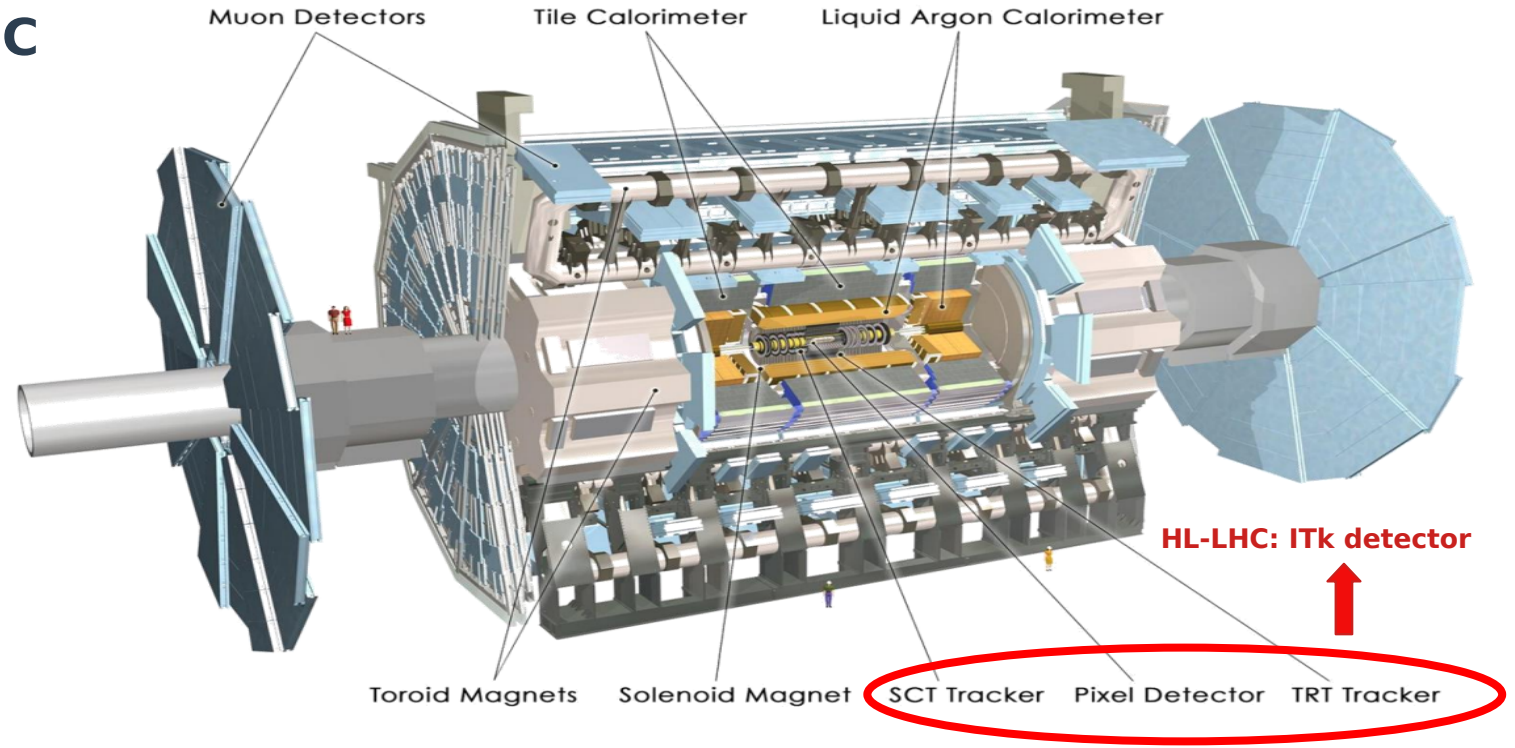
Outline

1. **Thesis context: ITk building**
2. **ATLAS Qualification Task**
3. **Thesis work:**
   - Baseline
   - Method
   - Results
   - Where are we going?

1. Thesis context: ATLAS detector for HL-LHC

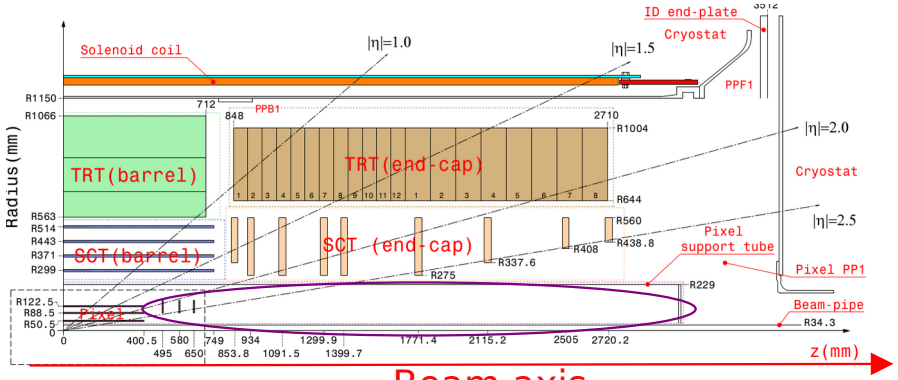
High Luminosity-LHC (HL-LHC):

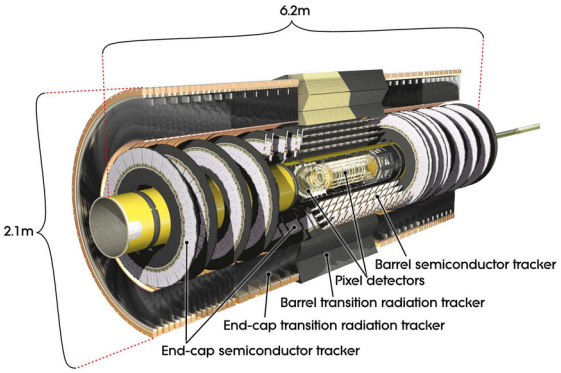
- **Expected in 2029**
- **Increase of luminosity**
  - Luminosity: ~ number of collisions per seconds



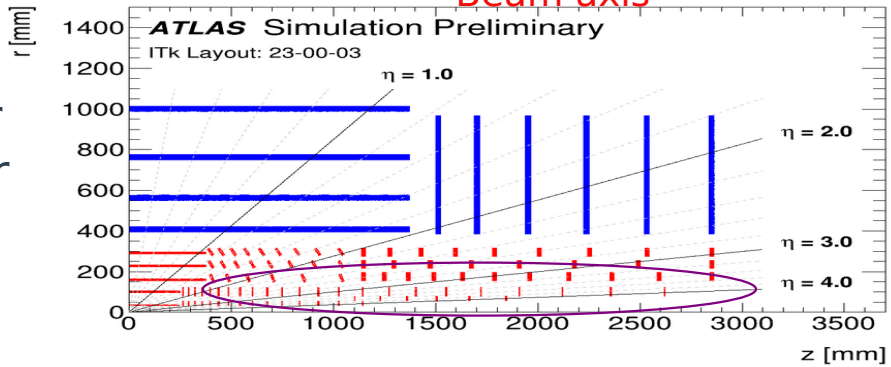
Inner Detector Overview

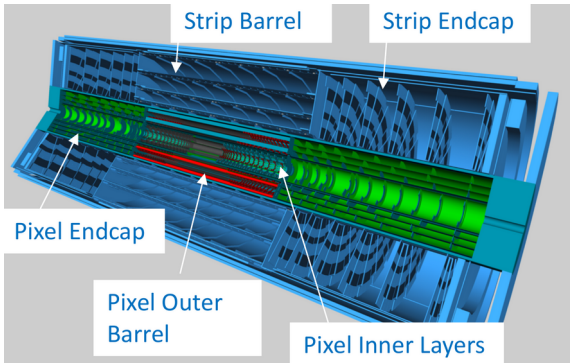
Current
Inner detector





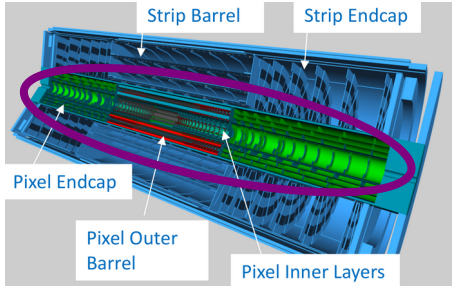
Inner Tracker
(ITk) detector

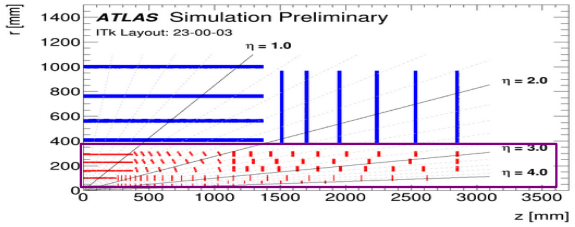


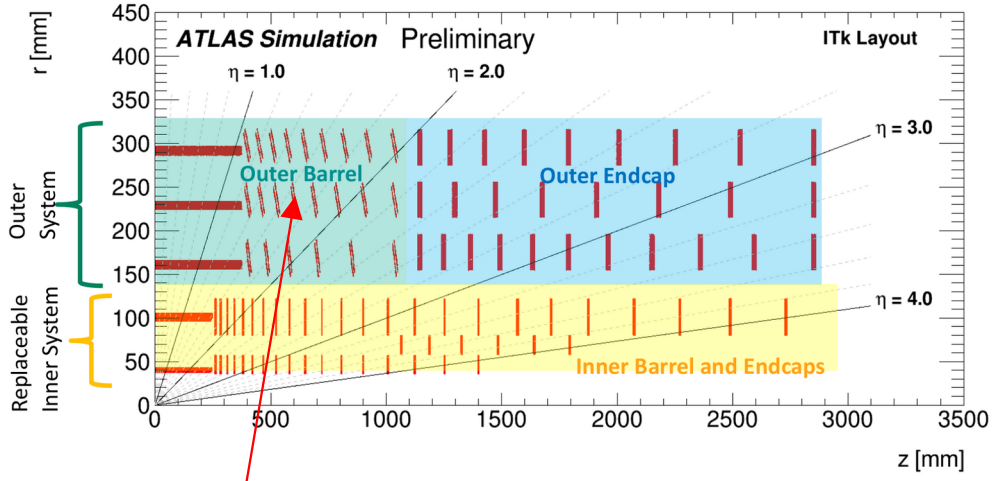


ITk: Wider coverage: |η| < 4
Higher granularity

ITk Pixel Detector Overview







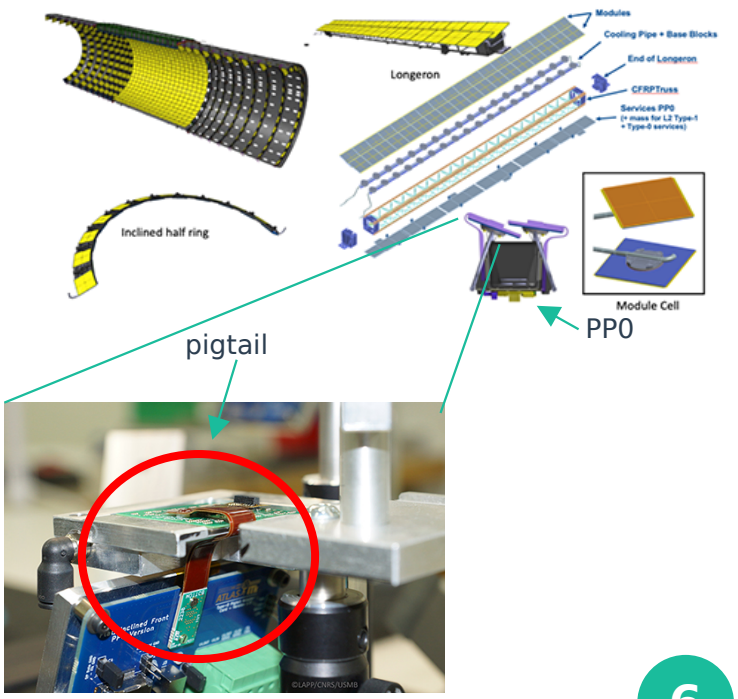
LAPP
Laboratoire d'Annecy de Physique des Particules

LAPP is building components of the Outer Barrel (OB)

LAPP ITk Pixel Detector Overview

- **LAPP is producing 75% of the OB Types 0 (5000 pigtails, 400 PP0 boards) and will be integrating 25% of the local supports (with LPSC and CPPM)**
  - → Qualification Task on Production database for Types 0 (pigtails and PP0)
  - → In 2024, participation to the electrical testing of modules at CERN or LAPP

- **Types 0: Components directly on the detector**
- **Pigtails: Power supply, monitoring of the cell and transmit data from the module cell**
- **Patch Panel 0 (PP0): Distribute power supply and aggregate data**

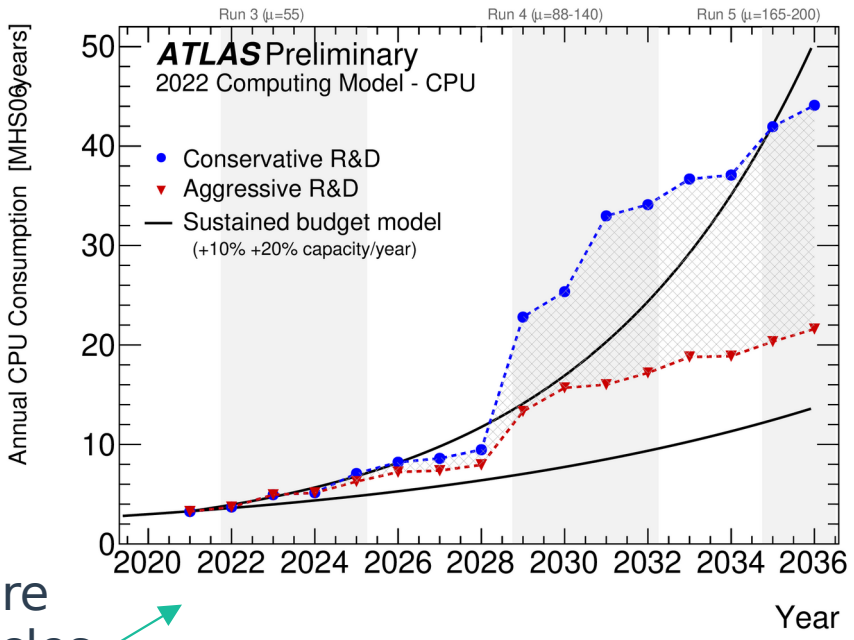


Pictures taken from: https://lapp.in2p3.fr/spip.php?article3307

Inner Tracker (ITk) for HL-LHC

- **ITk:**
  - Wider coverage: |η| < 4
  - Higher granularity

More particles detected

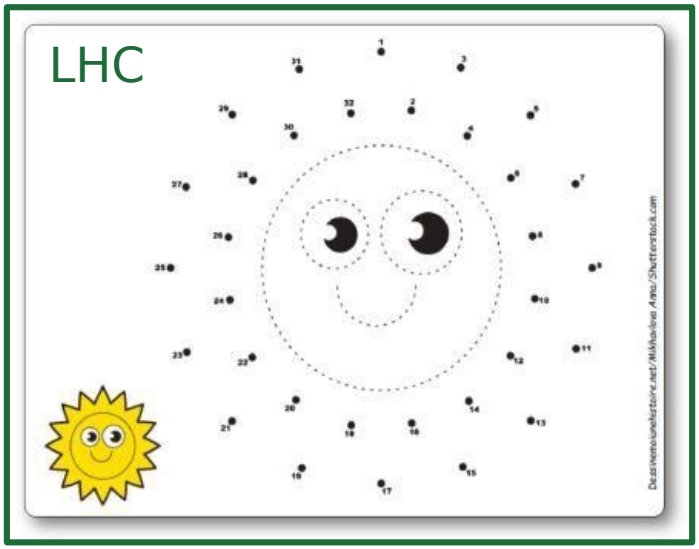


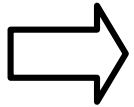
- **High Luminosity-LHC (HL-LHC):**
  - Between now and 2029: Luminosity x2.5
    - Collisions piles up in an event
    - Pile-up (<μ>): average number of collisions in an event

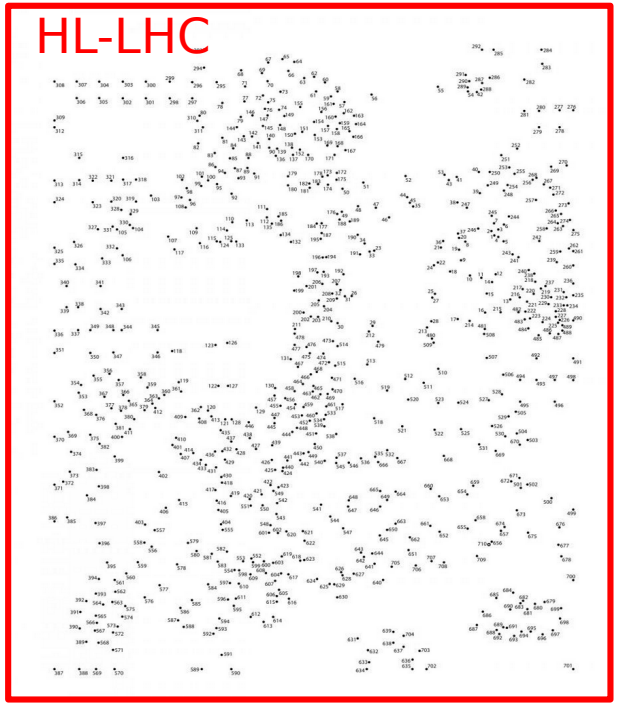
More particles created

ATLAS CPU previsions: need to improve ***tracking*** performance significantly

Inner Tracker (ITk) for HL-LHC: in short







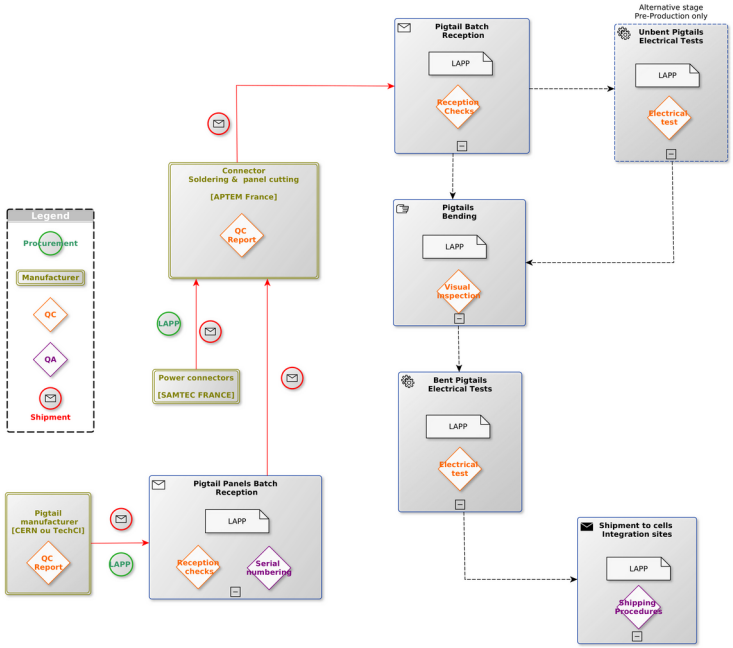
Tracking:

2. ATLAS Qualification Task: Production Database

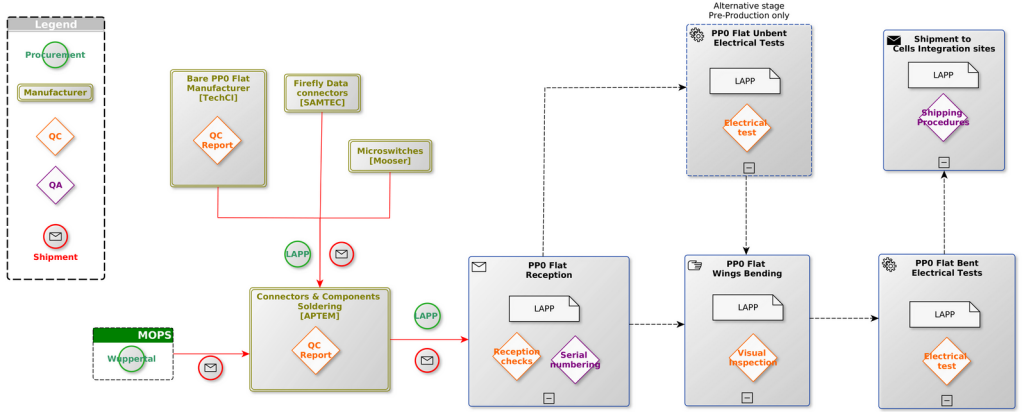
- **ATLAS Qualification Task:**
  - 1 year, 6 months of work
  - Allows to publish as ATLAS member
- **ATLAS Production Database**
  - Create components, store quality control data, track shipping, API

- **Qualification Task:**
  - Creation of a dedicated “LAPP Types 0 Web app” to improve data registration in the database, robustness and scalability (***5000 pigtails***)
- **Help other laboratories: make the tool available for all the laboratories involved in Type 0 qualification**

ATLAS Qualification Task: Types 0 web app



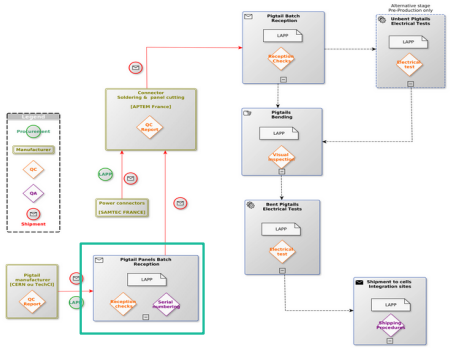
Pigtails production flow



PP0 production flow

Web app link: https://itk-web-apps-pigtails.app.cern.ch/

ATLAS Qualification Task: Types 0 web app example

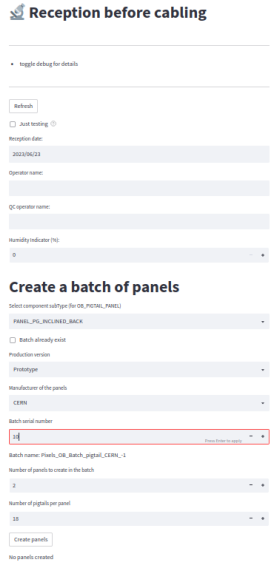


Pigtails production flow

Example:
Registering 20 pigtails

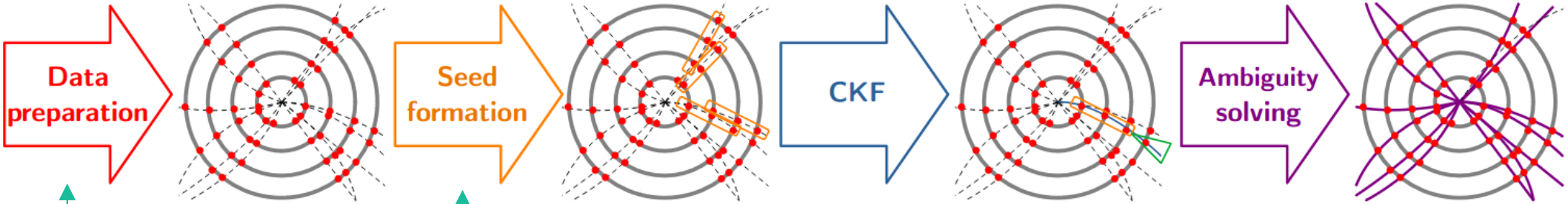
ATLAS PDB interface
x20

Web app
x1



Web app link: https://itk-web-apps-pigtails.app.cern.ch/

4. ATLAS Tracking



Transform detector hits into clusters and space points

Group compatible triplets of space points

Seed: 3 space points

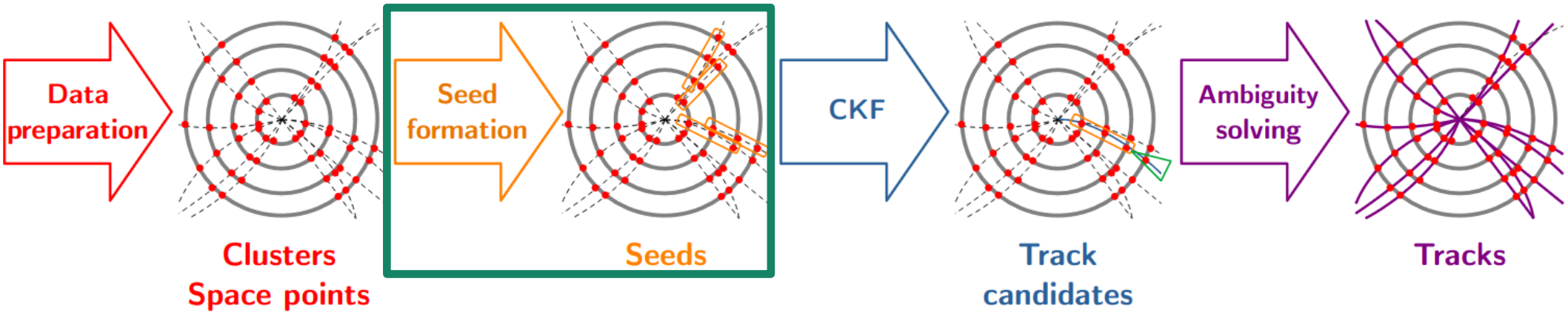
Expand the seeds with remaining space points

Track: arbitrary number of space points

Remove duplicated tracks and bad ones

Image made by Noemi Calace

ATLAS Tracking Focus on Seeding



- **What do we hope to improve?**
  - Seeds' efficiency: reconstruct at least one seed per track
  - Seeds' purity (fake rate): reconstruct only tracks' seeds
  - Seeds' redundancy (duplication rate): reconstruct just enough seeds per track
- **How? Group similar Space Points? Might also improve timing?**

Default Seeding Algorithm description

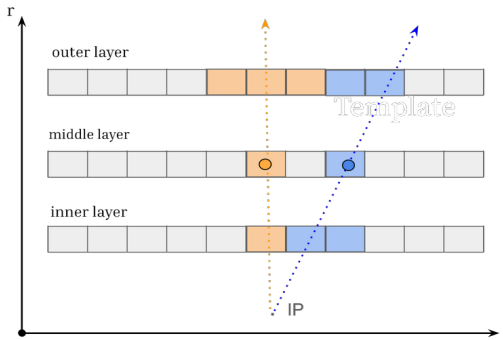
1. Seed Finder

- Check if the triplet forms a nearly straight line in the (r,z) plane

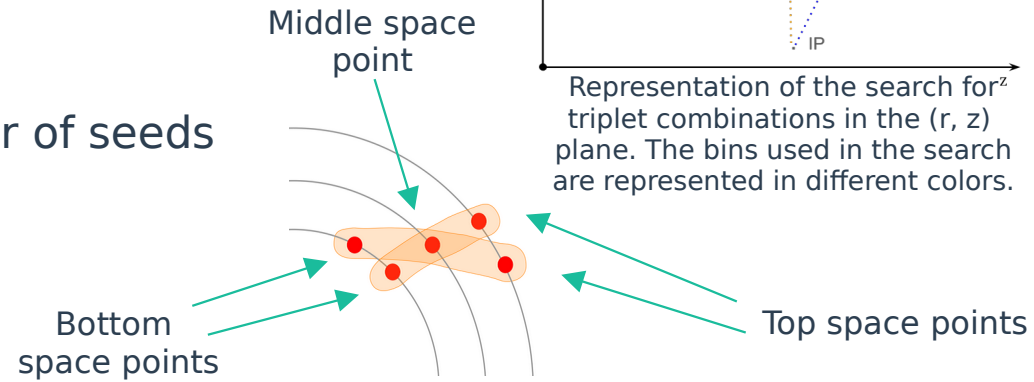
2. Seed Filter

- MaxSeedPerSpM cut limits the number of seeds to speed up the tracking

- **Possible improvement:**
  - MaxSeedPerSpM: Non physical cut
- **Can we remove it?**



Representation of the search for triplet combinations in the (r, z) plane. The bins used in the search are represented in different colors.



Seeds sharing the same middle space point

Baseline: Setup

ACTS: A Common Tracking Software
(generic tool) (experiment independent)

acts github

Run 4:
<μ> = 140

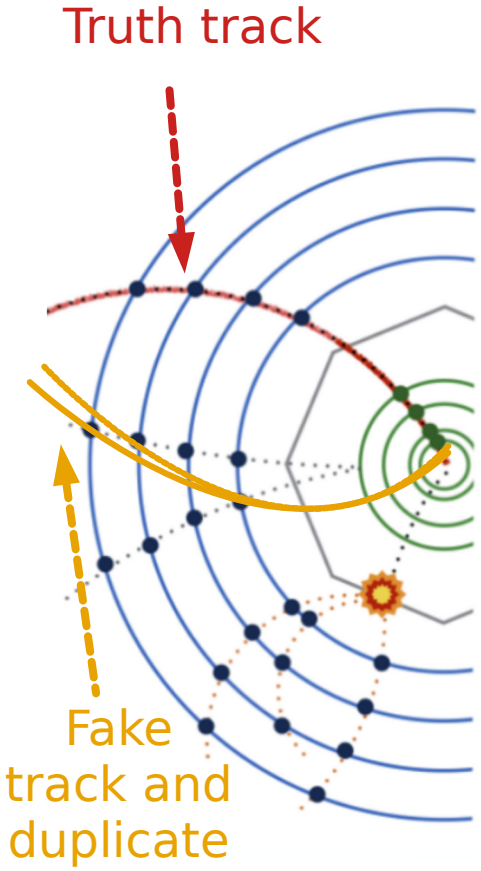
Sample: 100 tt events; |η| ≤ 4; pT > 1GeV; μ = 50, 100, 150

Generic detector (default detector, ideal detector without material)

Softwares: Pythia8 (simulation), FATRAS (propagation)

But not using Geant4: No secondary particles (particles created from the interaction with the detector)

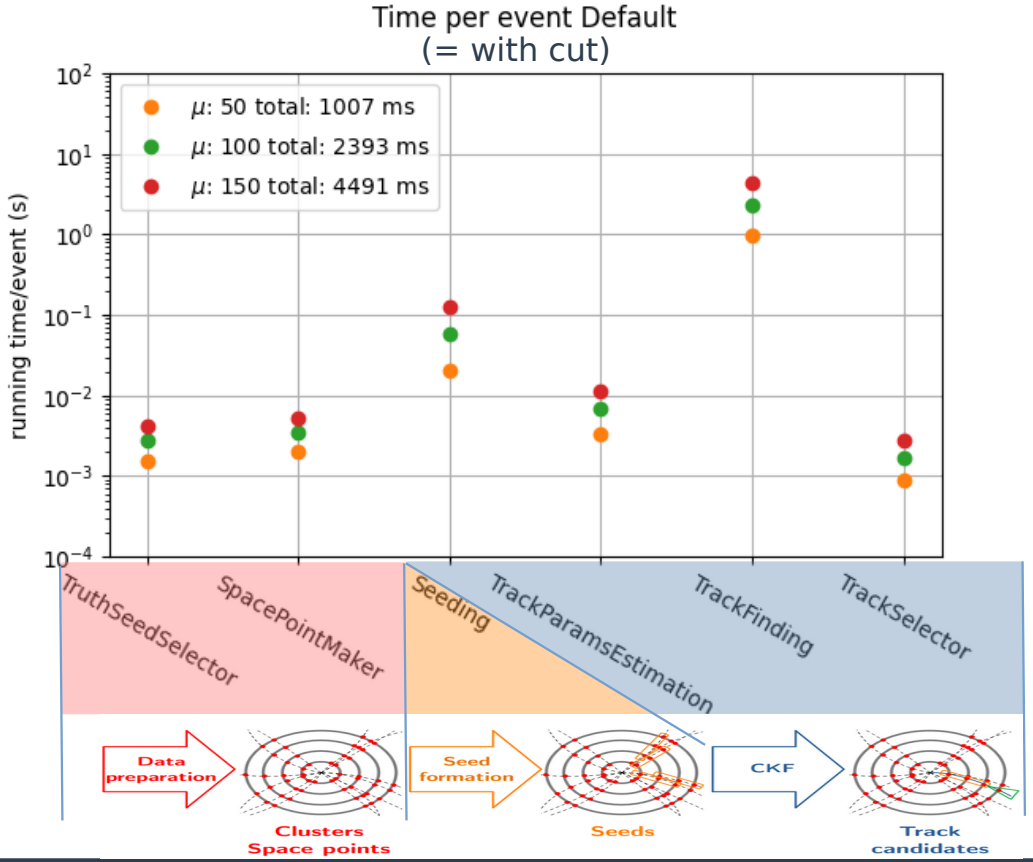
Evaluation

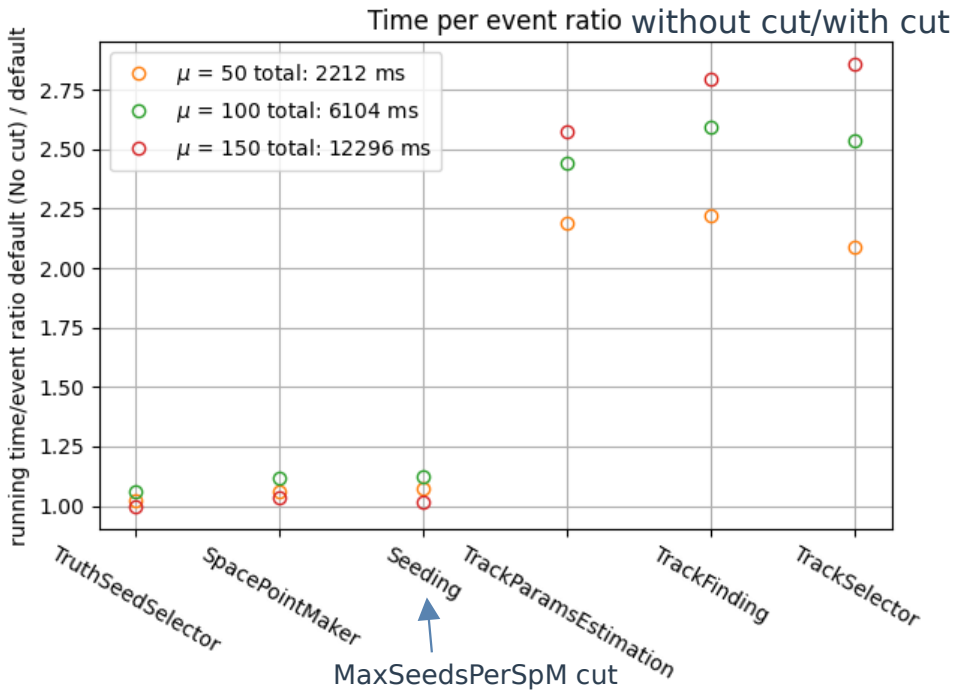


Evaluate on tracks:

1. Efficiency: Reconstruct as much “truth” tracks as possible
2. Fake rate: Reconstruct as low “fake” tracks as possible
3. Duplication rate: Avoid to duplicate tracks
4. Running time: Going as fast as possible

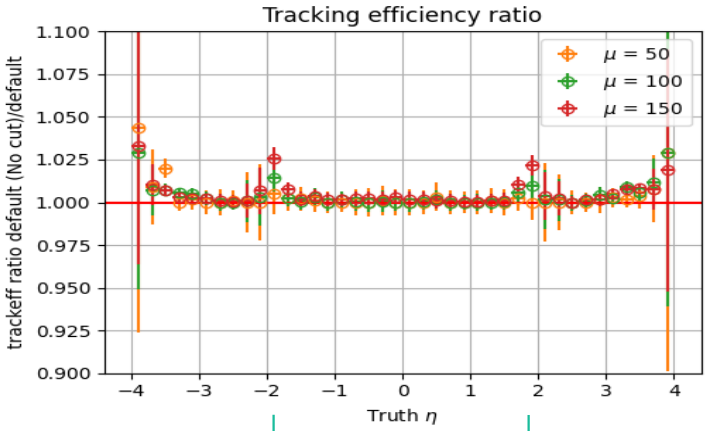
ACTS performance: Timing/event





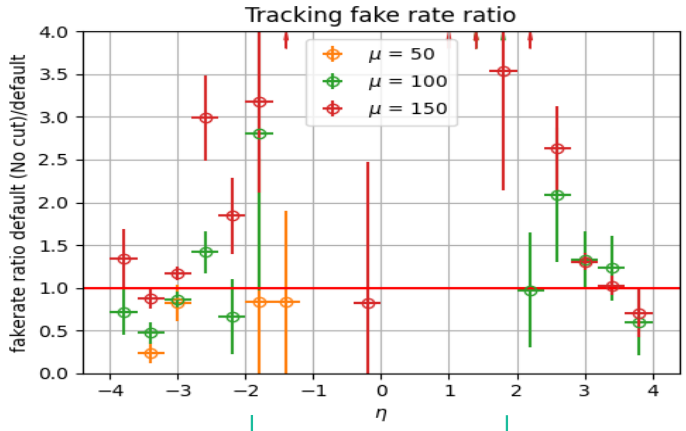
Timing multiplied by ~2-3 without the cut

ACTS performance: Physics



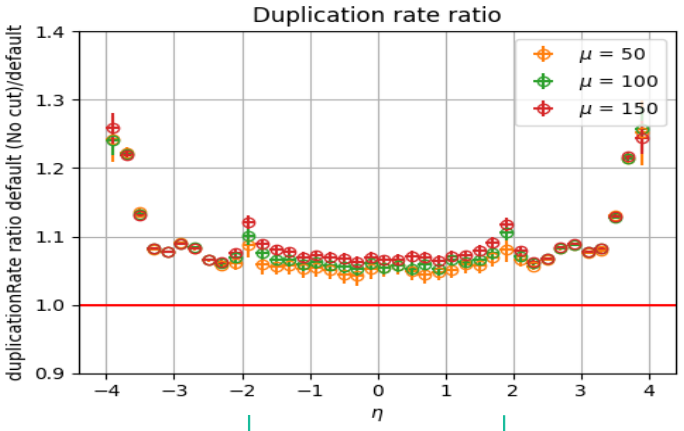
Forward region | Central region | Forward region

Same in central region,
Better in forward region



Forward region | Central region | Forward region

Worst in central region,
Same in forward region



Forward region | Central region | Forward region

Worst everywhere,
Even worse in forward region

MaxSeedPerSpM cut decreases the performance in forward region
But improves in central region

Target Goal

- **Without the cut: improve performance**
  <u>but</u> timing is crucial

- **Goal: Improve performance with same timing**
  - Keep the cut but try to bypass it

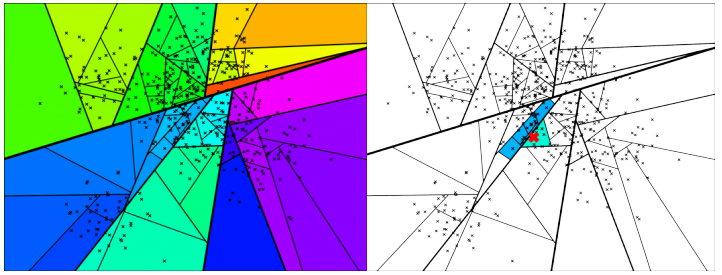
A new method:
Machine Learning/Hashing in the Seeding

Hashing:
1. Group similar space points into buckets
2. Do the seeding on each bucket

Algorithm used:
Approximate Nearest Neighbors Oh Yeah (**Annoy**)
→ Used by Spotify

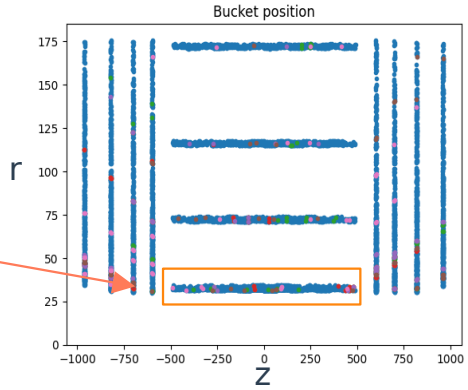
- Machine Learning algorithm type:
  - k Nearest Neighbors (unsupervised)
  - Random based
- Constraint: Starting from layer 0
- Parameter: Number of Neighbors (bucket size)
- Use the distance between the points
  → **need to define a (relevant) metric**

Annoy:



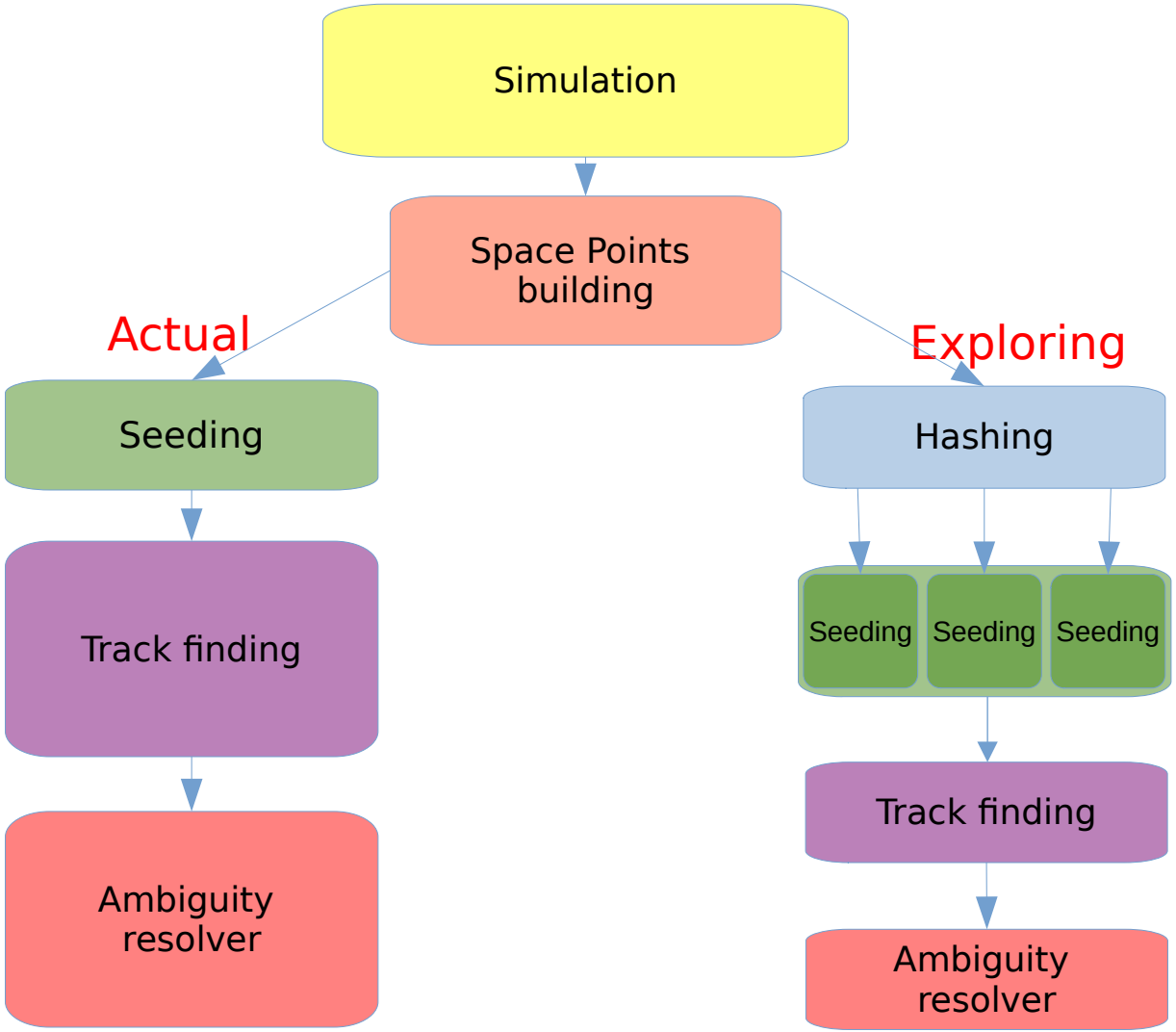
Space separation

Look for neighbors in the closest regions



1 space point in layer 0 → 1 bucket

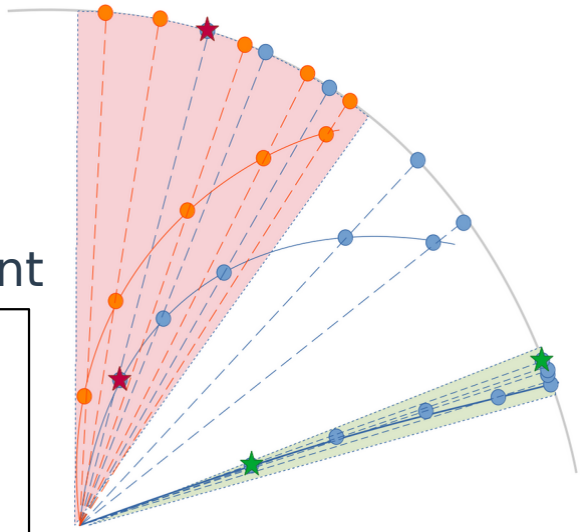
Approaches



- Seeding parallelization
- Hashing groups space points into buckets
- Hashing reduces the number of space points at a time (focus on relevant space points)
  → less seeds per bucket

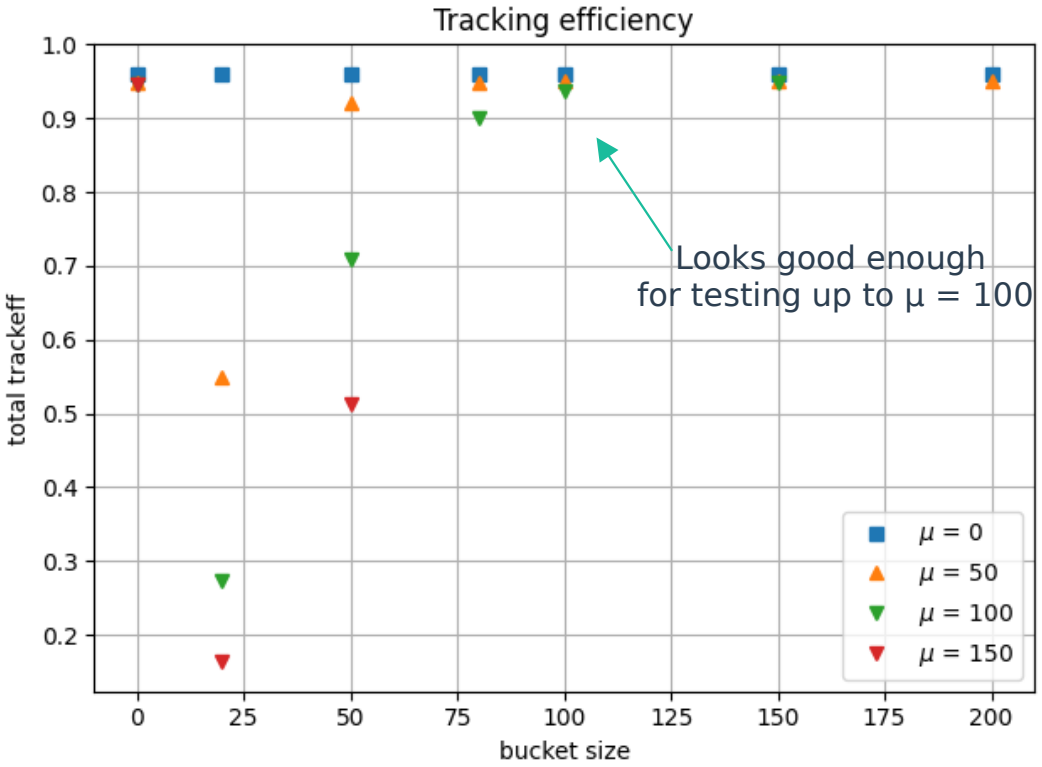
Metric and bucket size

Metric: Δφ



SP: Space Point

- Layer 0 SP
- SP
- Bucket
- Projection
- Track



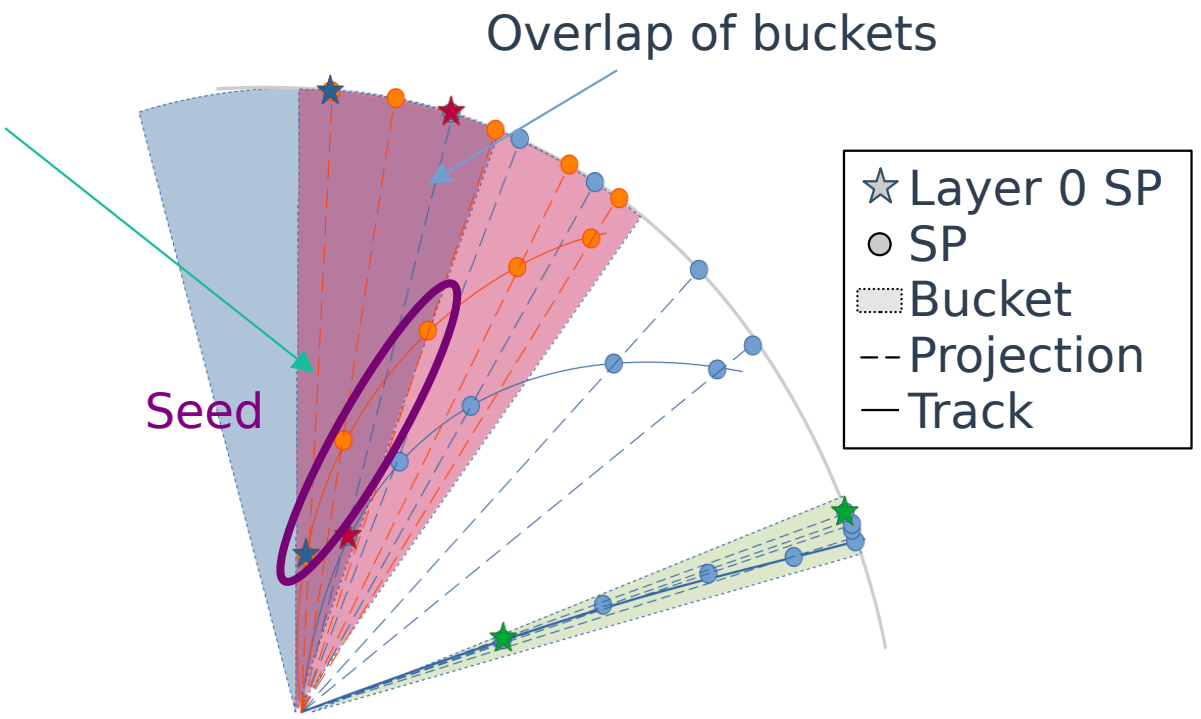
Fix the bucket size to 100

Hashing and overlap

- **Hashing introduces overlaps:**
  - The same seed can be reconstructed in several buckets (14 times in average)
  - Need some overlap to bypass the cut

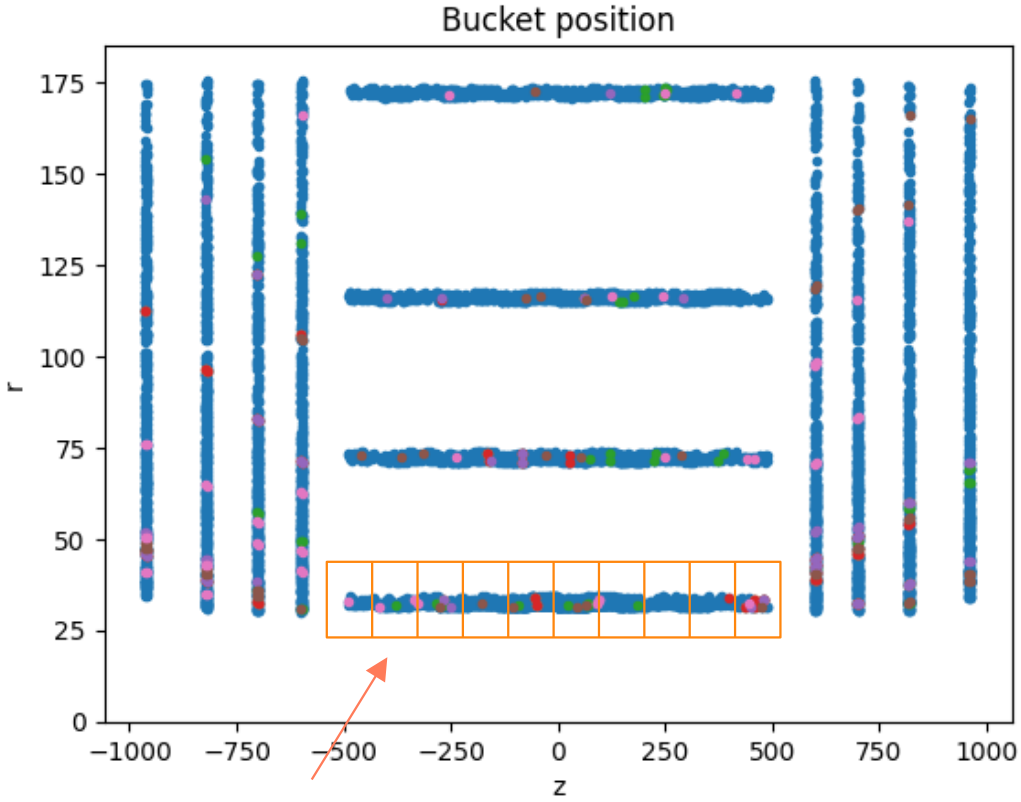
μ = 150	Timing/ event (ms)
Without Hashing	4491
With Hashing	7909

Hashing made timing x2



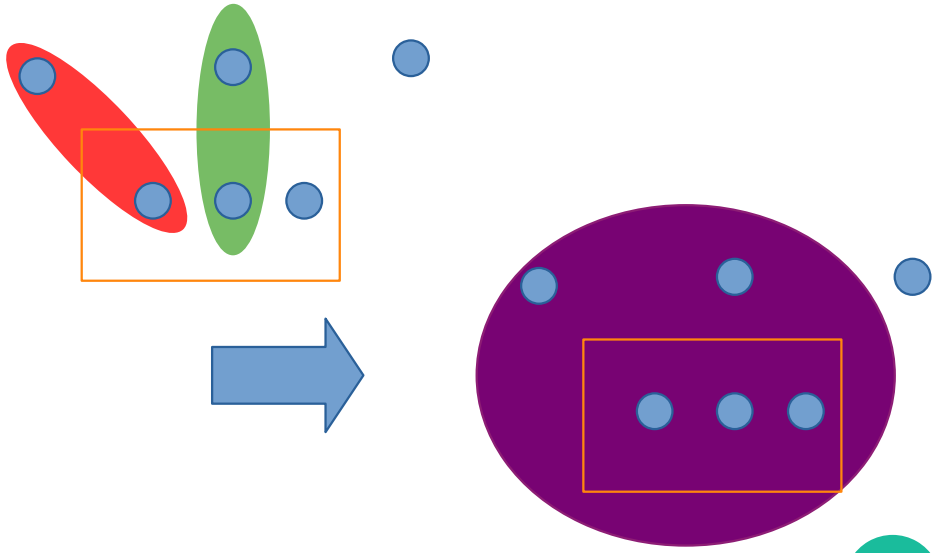
New idea: Group buckets
→ less overlap

Super buckets and binning

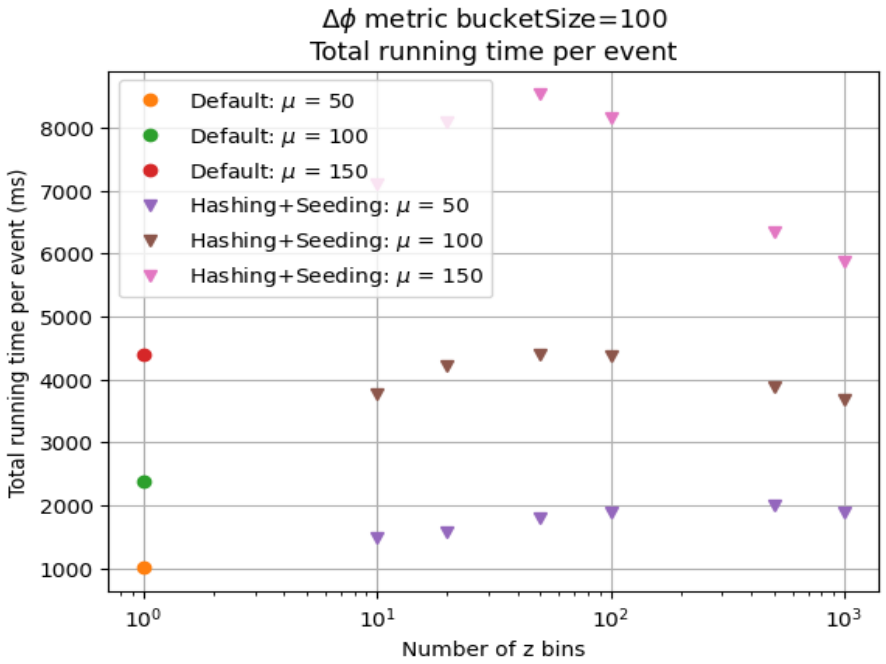


1 bin → 1 super bucket

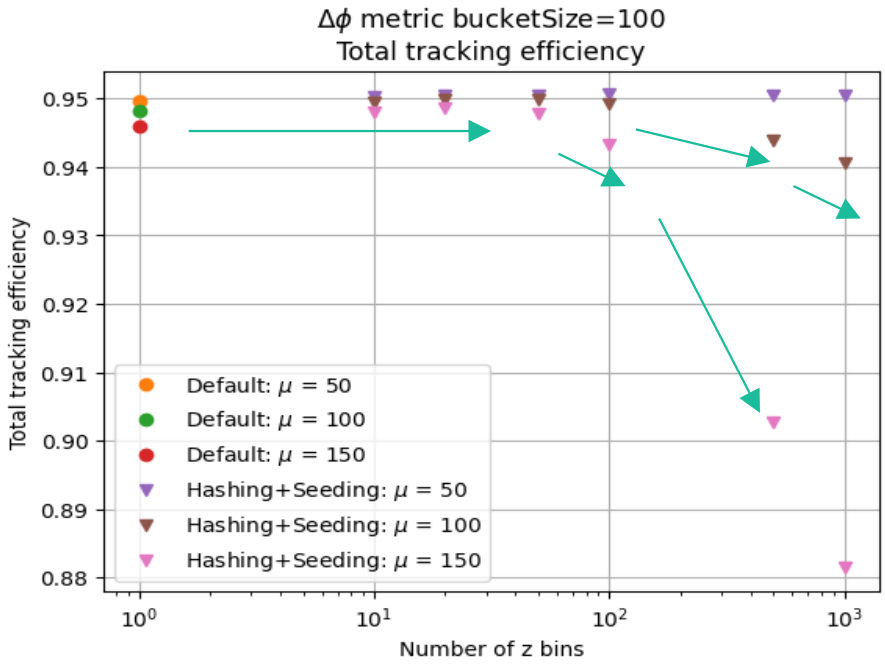
Super bucket:
Merging of the buckets created from the space points inside the z bin



Hashing performance: Timing and efficiency



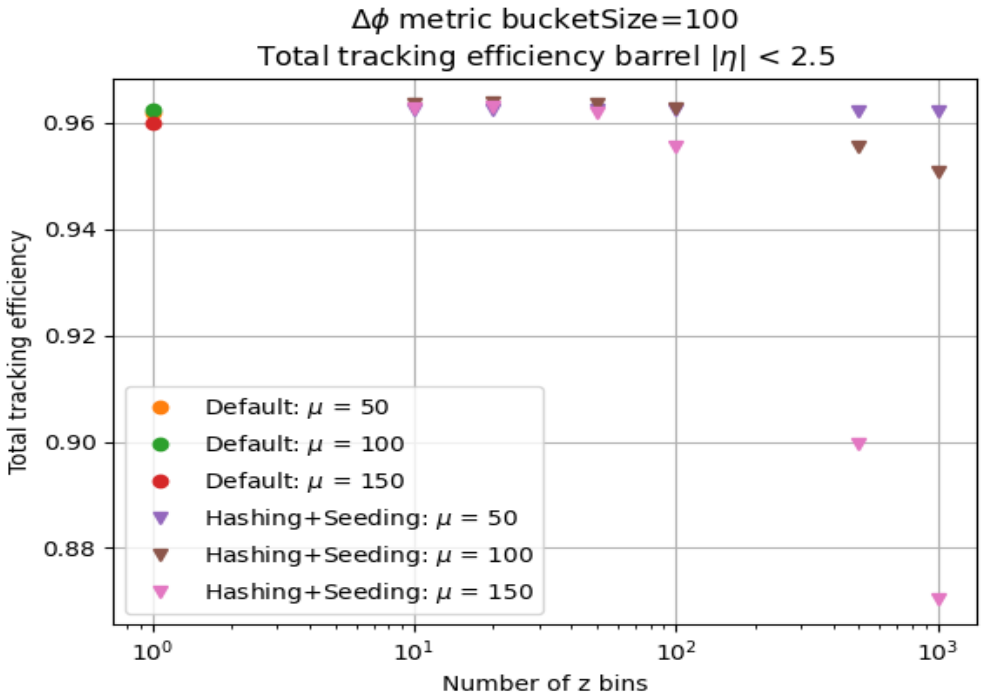
Running time ~ x2



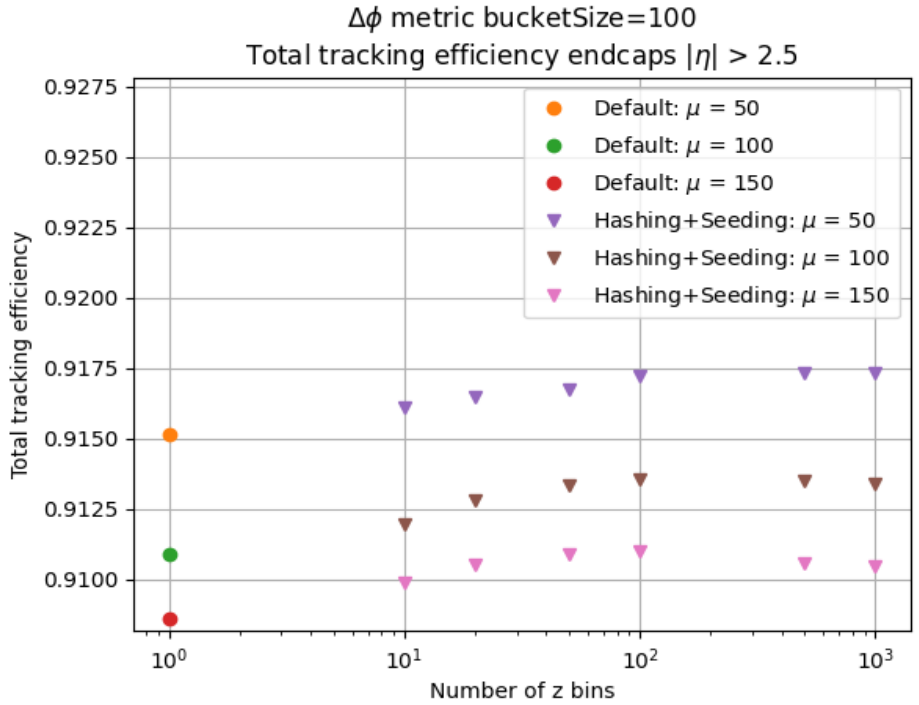
Improvement for small number of bins

Slower but better

Hashing performance: Efficiency (detailed)



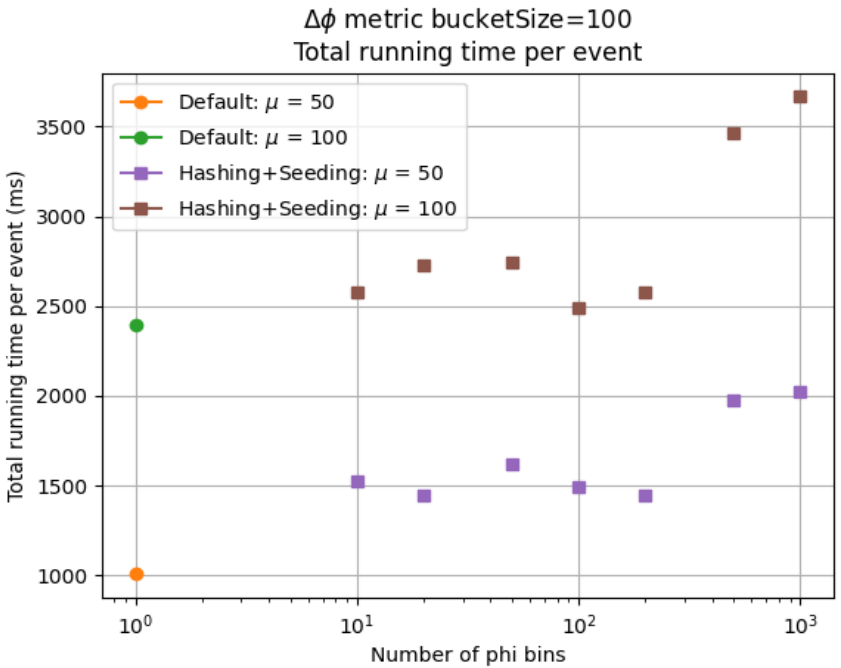
Improves then drops



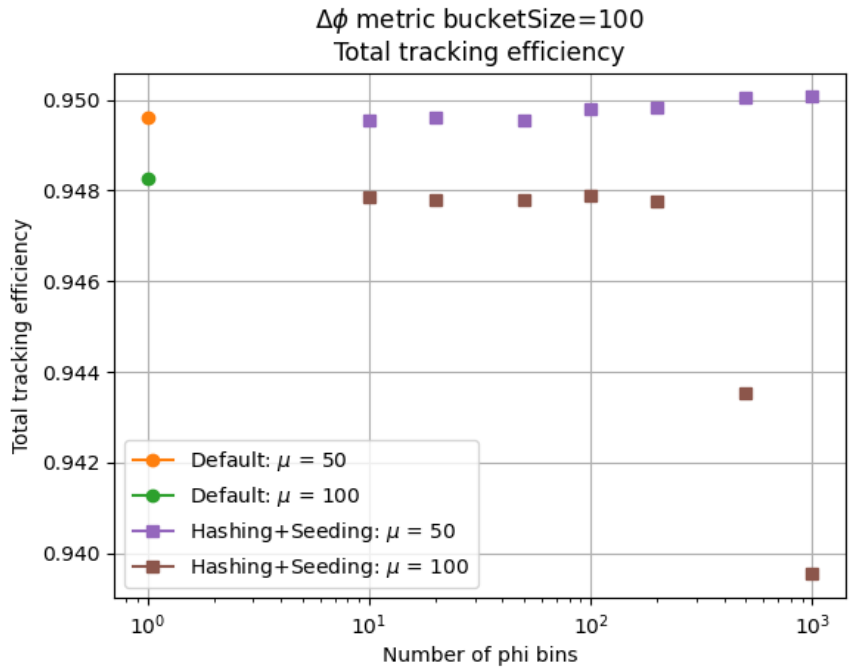
Always improve

Can be better in every regions

Hashing φ bins: Timing and efficiency



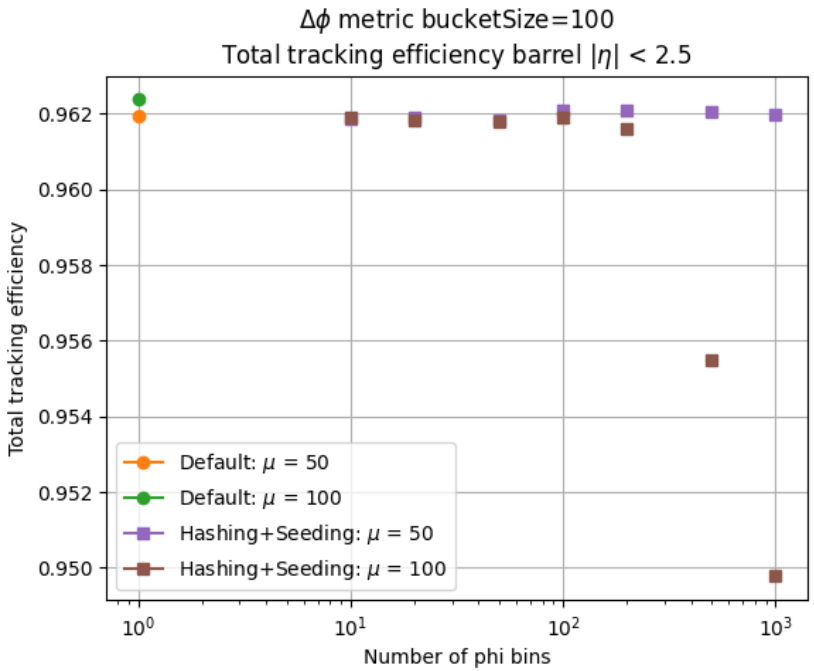
Similar running times



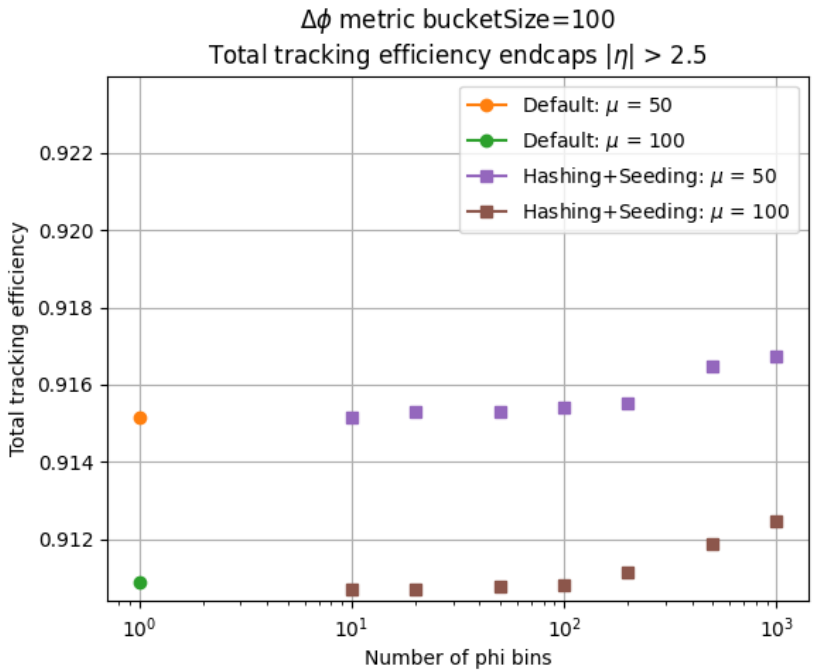
Small loss of efficiency

Not as good but as fast

Hashing φ bins: Efficiency (detailed)



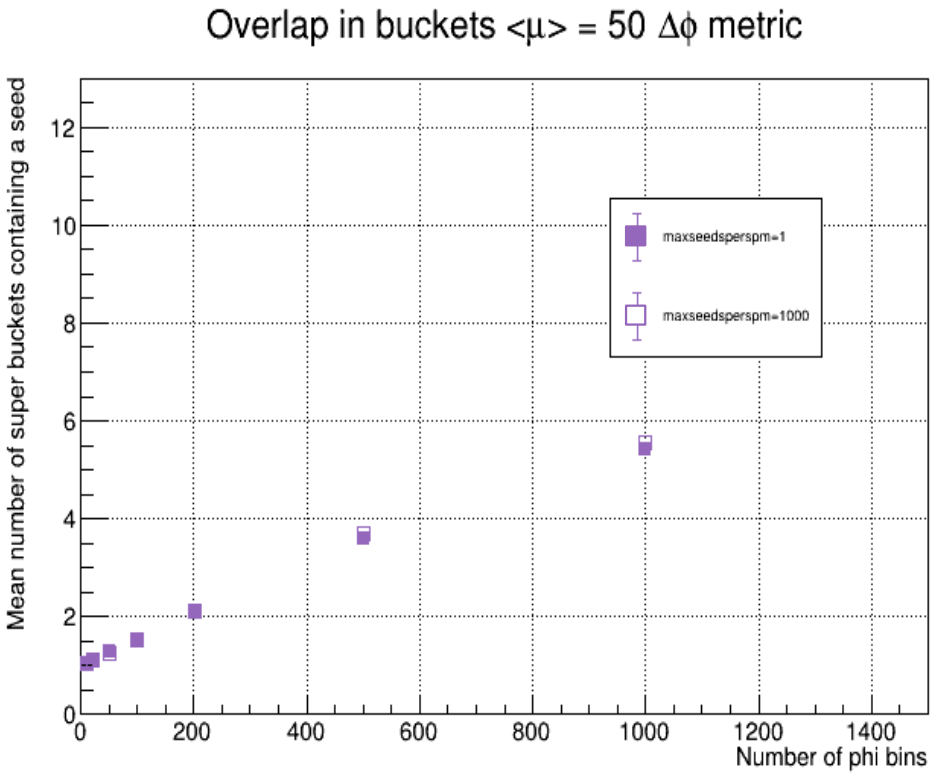
Drop of efficiency in the barrel

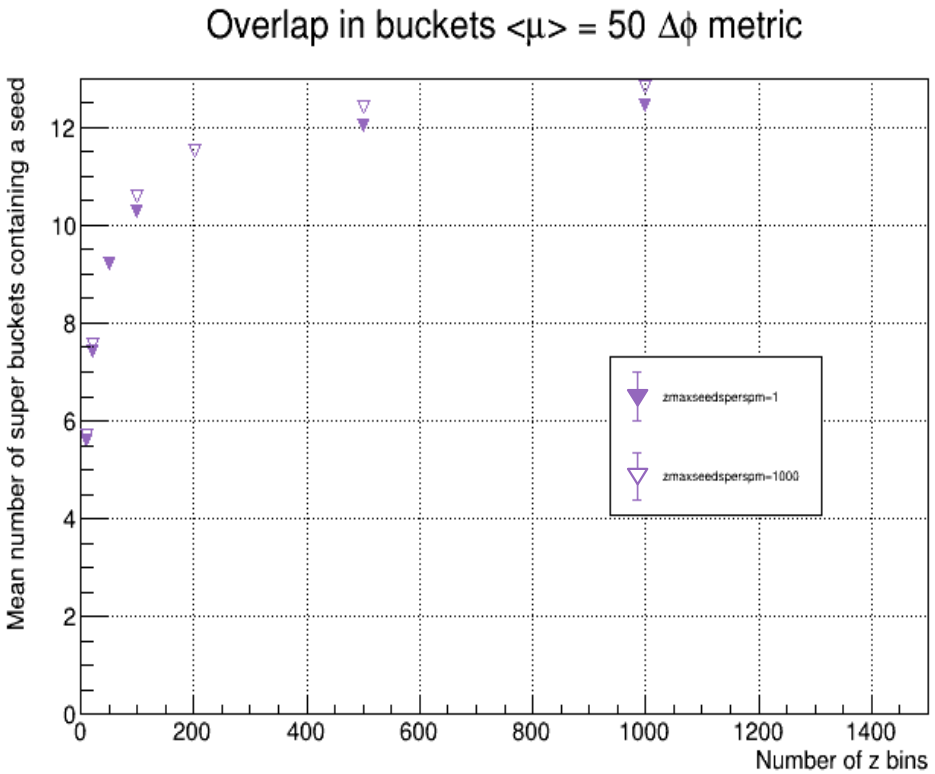


Better efficiency in the endcaps

Better in forward region

Overlap in buckets





Less overlaps between buckets with φ binning

Where are we going?

- **Current state:**
  - \+ Comparable performance with the baseline (slight improvement in the forward region!)
  - \+ Trade off between timing and performance (Z binning vs φ binning)
  - − Not better timing than baseline for now

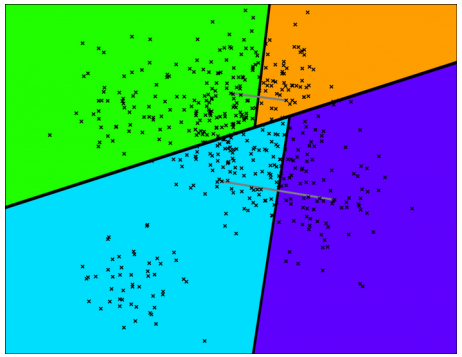
Next:

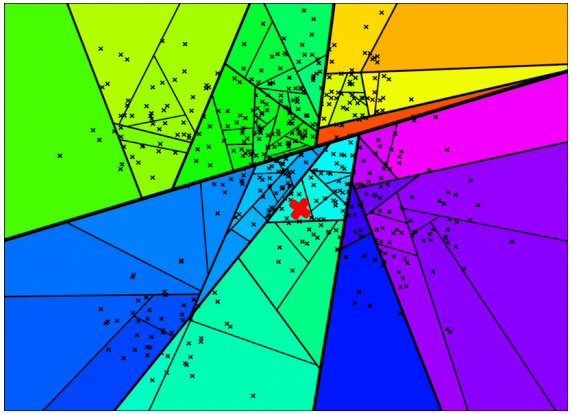
- **Full sim with Geant4 (secondaries), ITk, metric learning, binning in (z, φ), changing bucket size with detector region, Cluster shape?**
- **Summer School: HEP and ML. Link**
- **September: Quark/Gluon Tagging with ML**

Backup

Annoy training

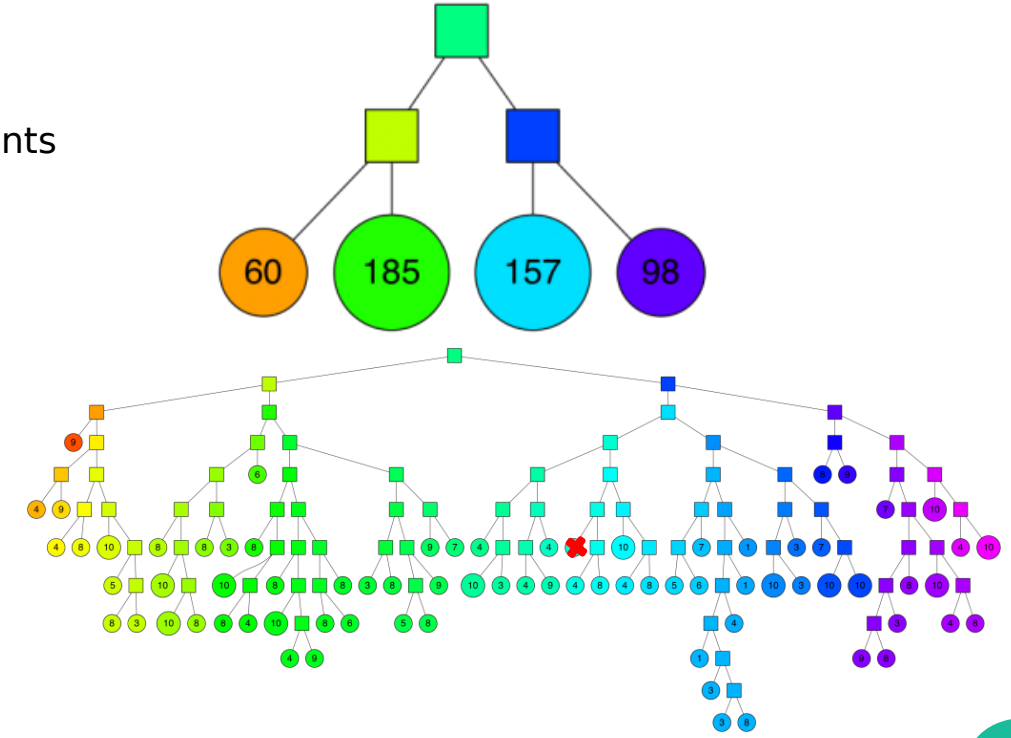
Space separation



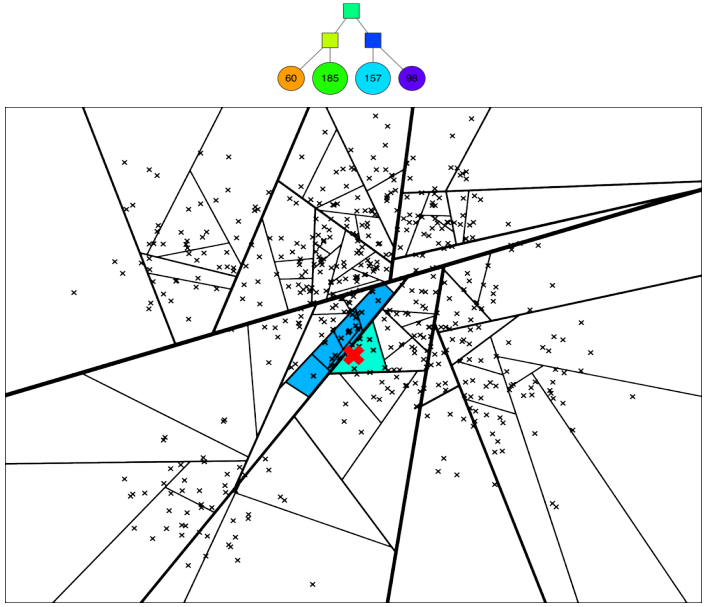


Takes two random points iteratively

Corresponding binary tree

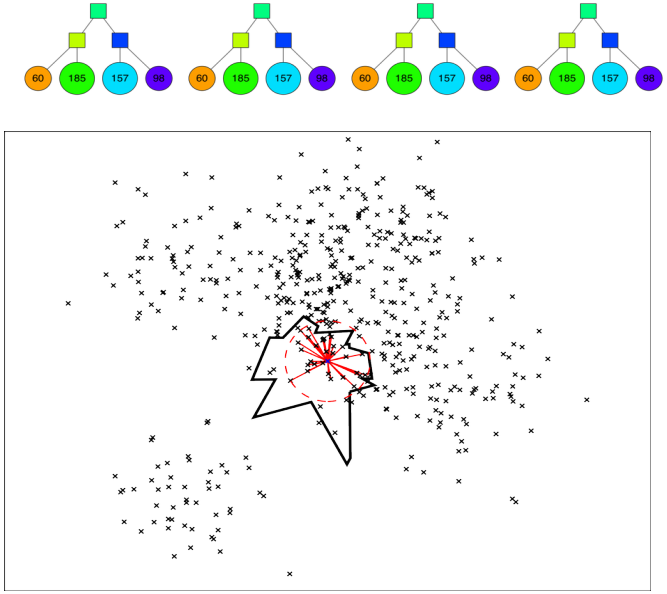


Annoy query



Merge neighbor subspaces

Approximation

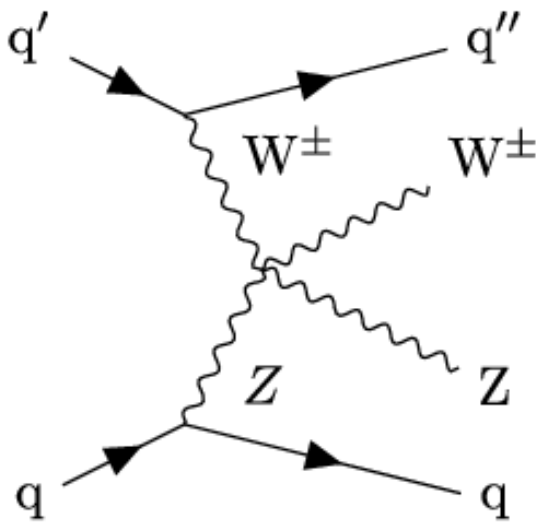


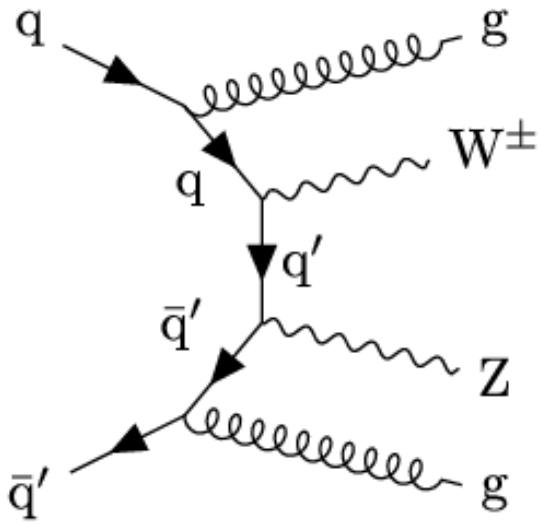
Union of trees' subspace

- Annoy tuning parameters: number of neighbors, number of trees, metric used, features used, number of subspace to look at

Quark/Gluon tagging: Physics motivation 1

Vector Boson Scattering: Quartic Gauge coupling measurement



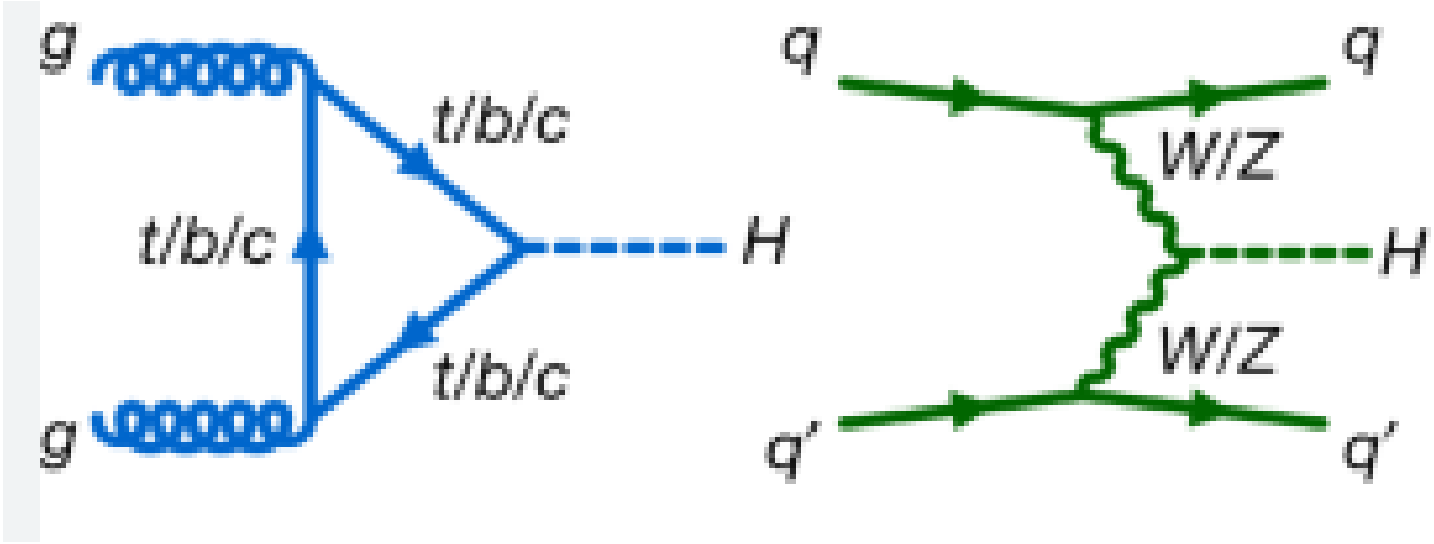


SIGNAL: Electroweak process
Forward quark jets

Background: QCD Process
randomly distributed gluon jets

Quark/Gluon tagging: Physics motivation 2

Higgs Boson production: separate cross-section measurements in various modes

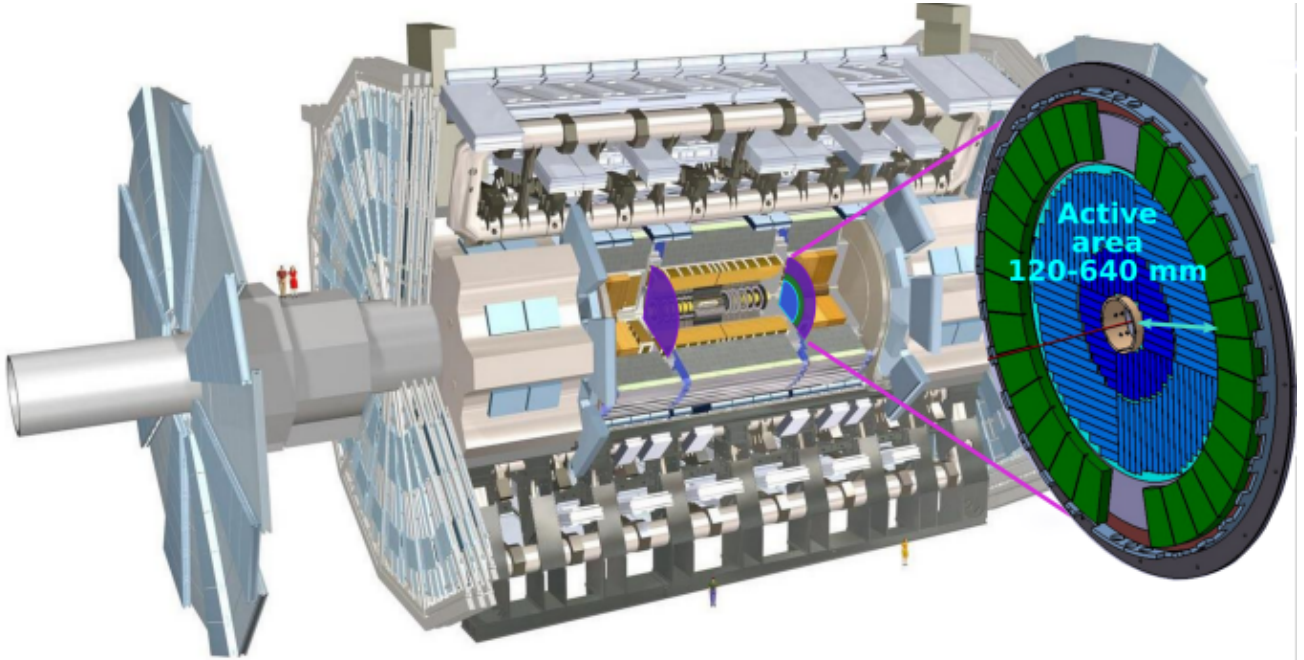


QCD Process: randomly distributed gluons jets

VBF process: forward quark jets

Quark/Gluon tagging: ATLAS forward region@HL-LHC

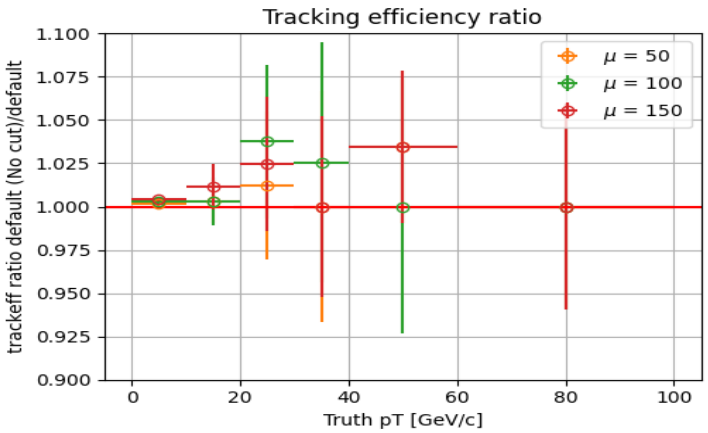
Nouveau détecteur ITk + HGTD:

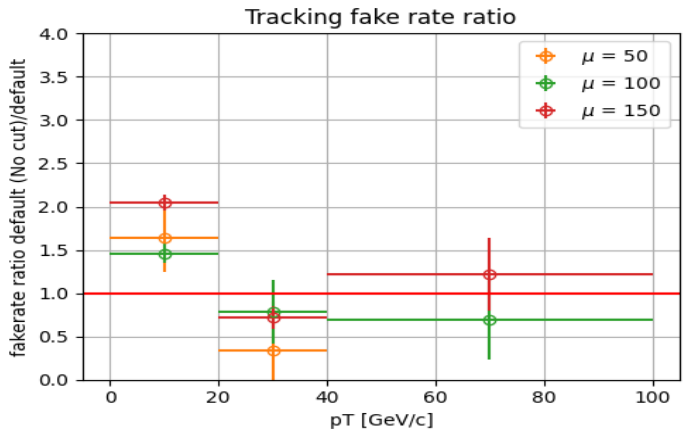


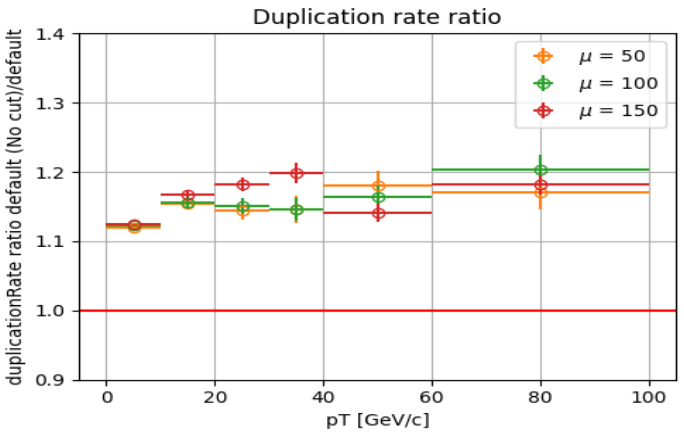
Formations

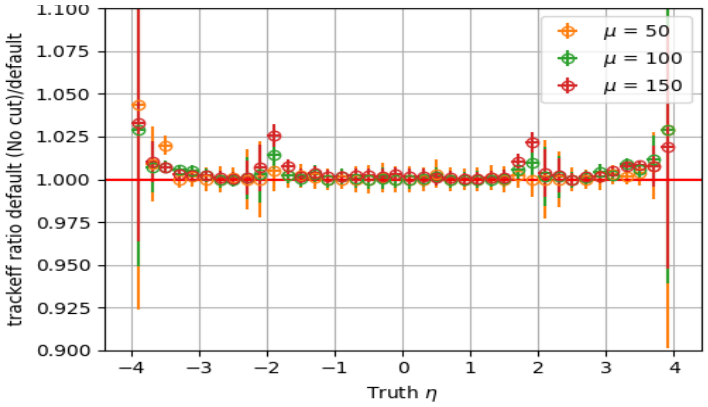
- **Doctoral School (UGA):**
  - Requires 120 hours: 1/3 Scientific, 1/3 Professional, 1/3 Transversal
- **Professional:**
  - “S'ADAPTER A SON ENVIRONNEMENT DE TRAVAIL” (10 hours)
- **Scientific:**
  - Workshops: ATLAS ML, ITk Tracking, ATLAS Induction Day and Software Tutorial (44 hours)
- **Transversal:**
  - Opened Science and HAL (4 hours)
  - “JOURNEE DE RENTREE DES DOCTORANTS 2022” (10 hours)
  - Planned: Mooc on ethics (15 hours)
- **BONUS: Summer School@ MITP (HEP and ML). Link**

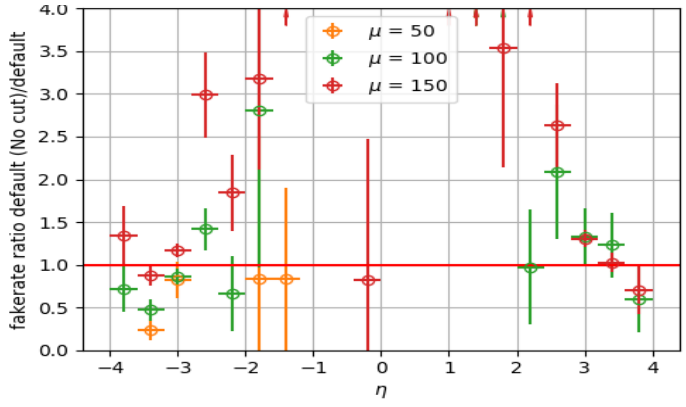
ACTS performance: Physics

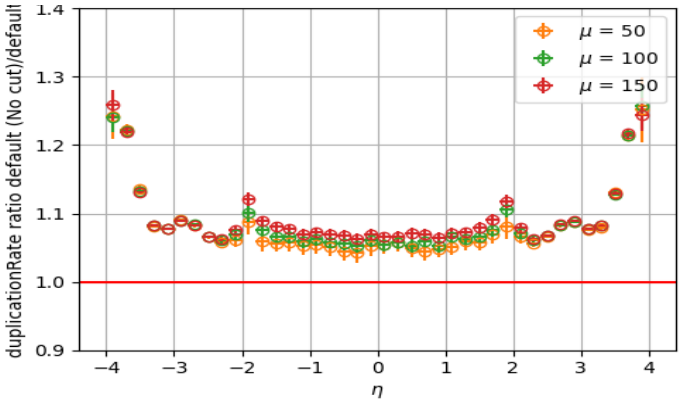












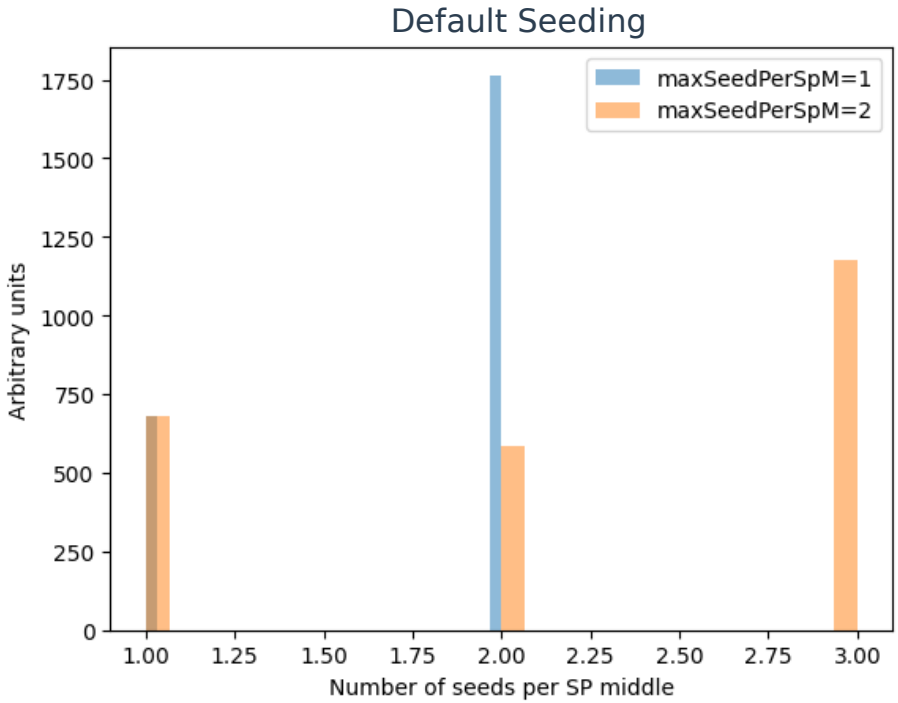
Seed finder configuration

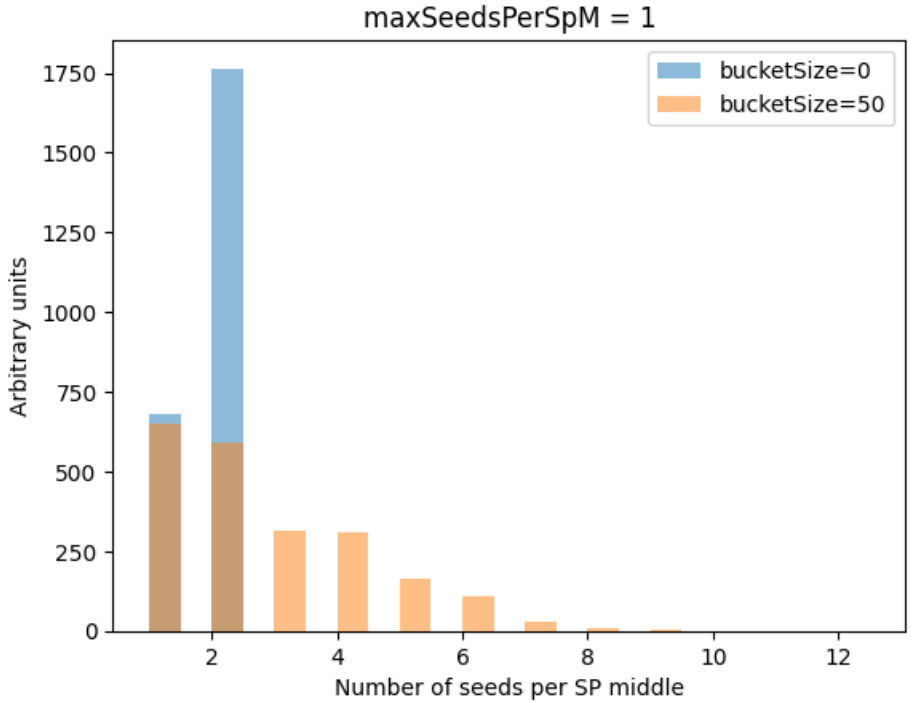
```
SeedfinderConfigArg = SeedfinderConfigArg(
        r=(None, 200 * u.mm),  # rMin=default, 33mm
        deltaR=(1 * u.mm, 60 * u.mm),
        collisionRegion=(-250 * u.mm, 250 * u.mm),
        z=(-2000 * u.mm, 2000 * u.mm),
        maxSeedsPerSpM=1,
        sigmaScattering=5,
        radLengthPerSeed=0.1,
        minPt=500 * u.MeV,
        bFieldInZ=1.99724 * u.T,
        impactMax=3 * u.mm,
        cotThetaMax=cotThetaMax # =1/tan(2×atan(e^(-eta)))
    )
```

MaxSeedsPerSpM cut

- **Purpose:**
  - Reduce the number of seeds to expand to speedup the track finding
- **Idea:**
  - Only keep at most MaxSeedsPerSpM+1 seeds sharing the same middle space point
- **Implementation:**
  - Uses a score to compare the seeds
  - The score is related to how close the impact parameter is to 0
- **Benefit:**
  - speedup and less memory used
- **Consequence:**
  - Loss of efficiency

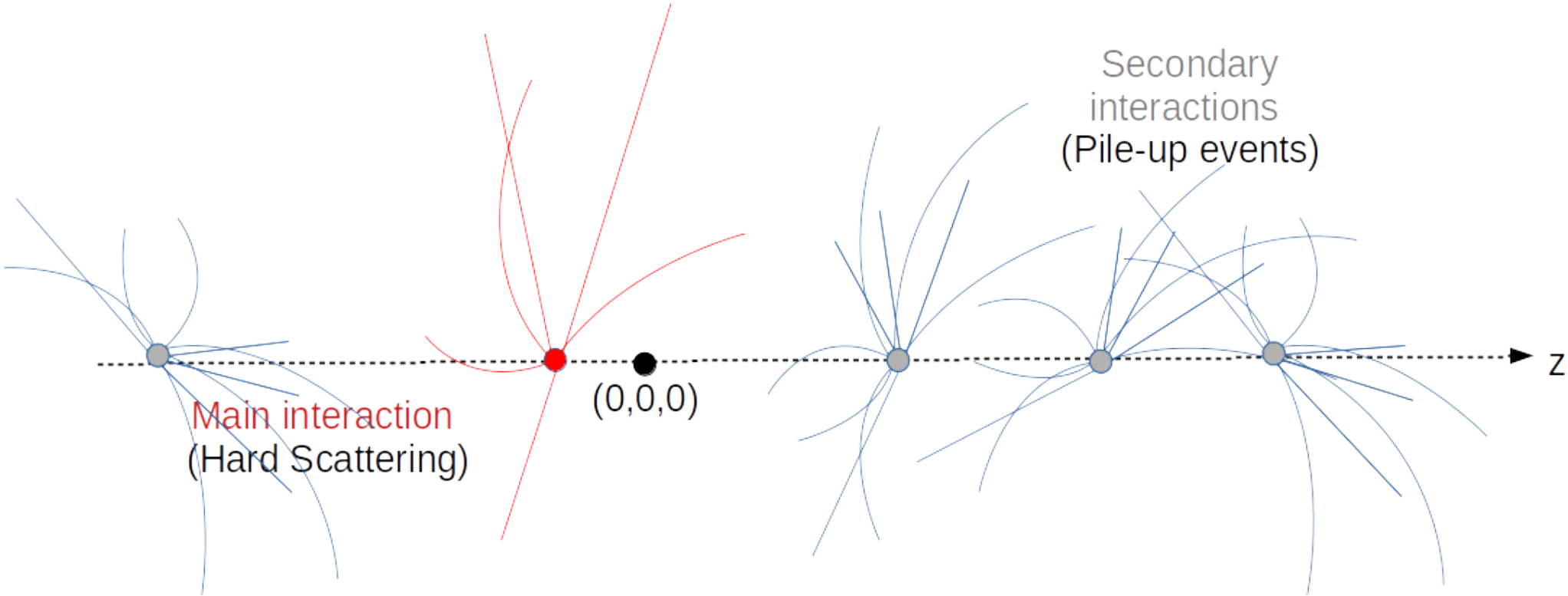
MaxSeedsPerSpM cut vs Hashing



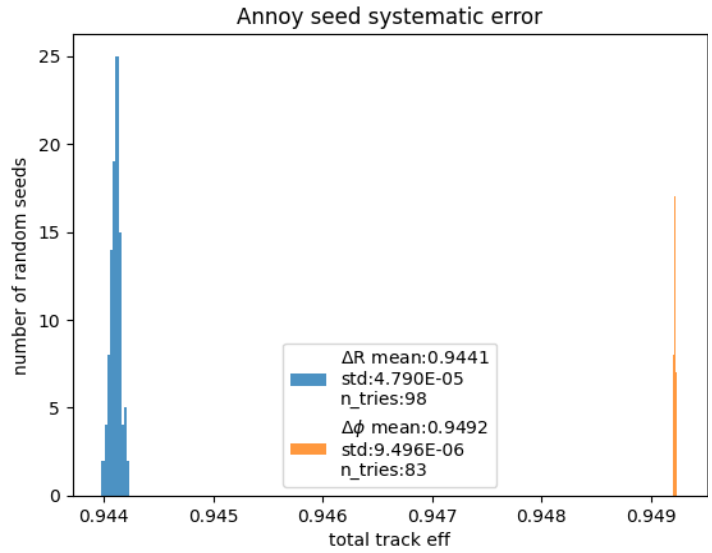


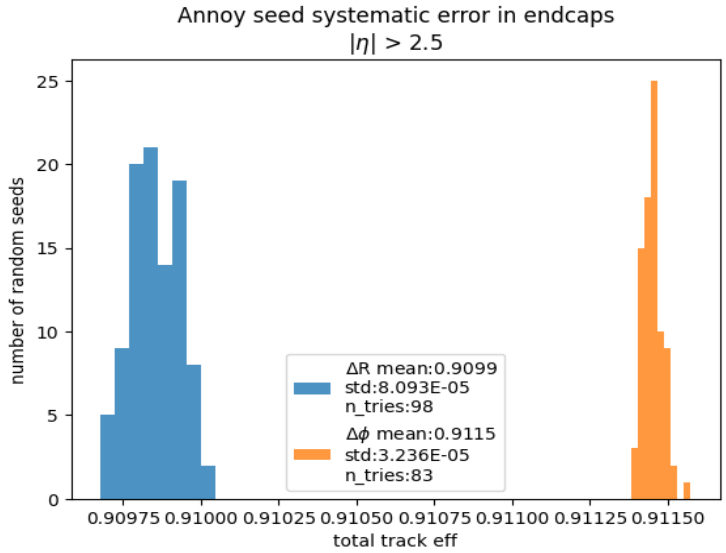
➡ Hashing get through the cut

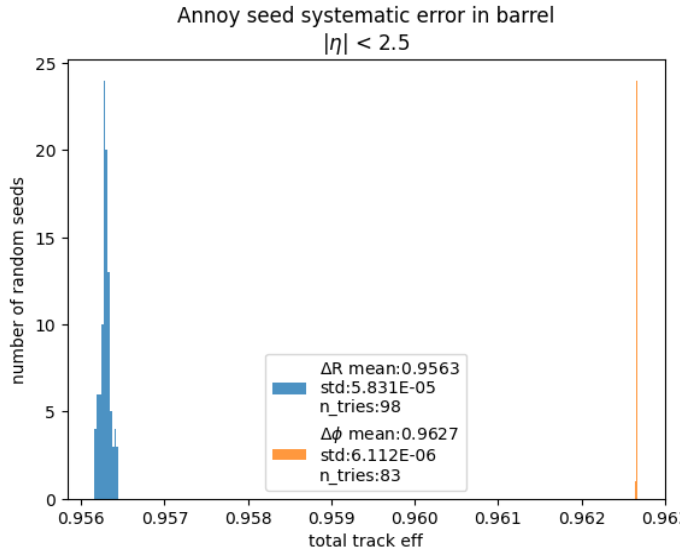
Pile-up events



Annoy random seed systematic error







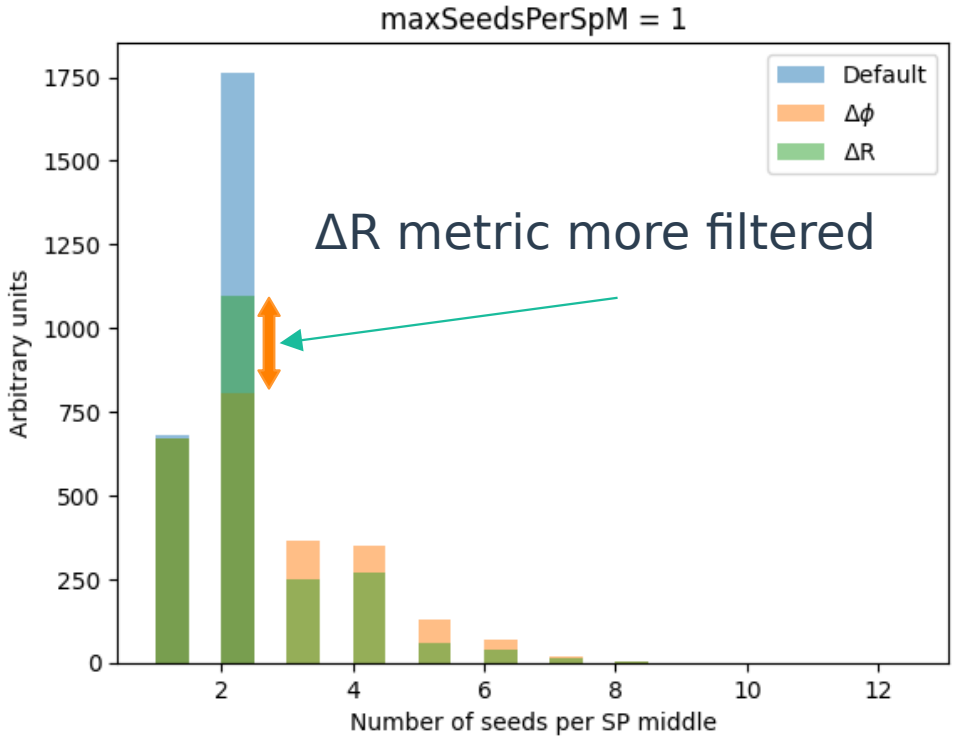
1000 events
in each try

BucketSize: 100
Mu: 50

Δφ is better

MaxSeedsPerSpM and ΔR metric

On 1 event:



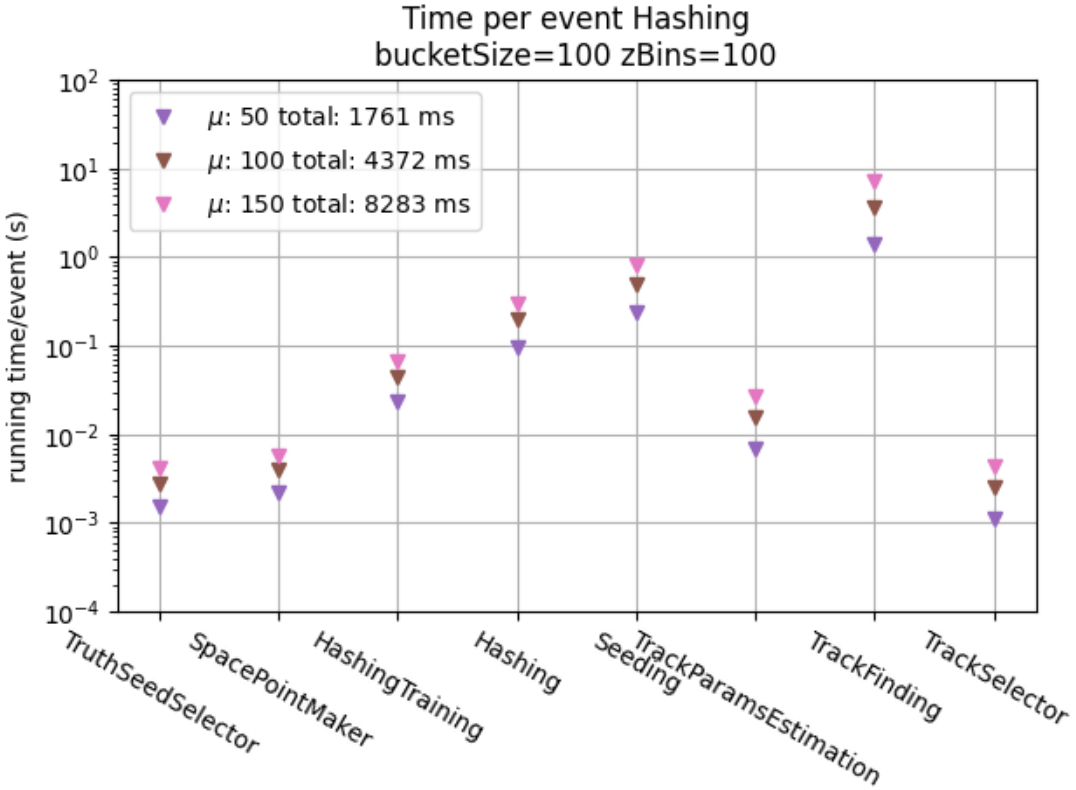
Filtered Middle Space points are on the maxSeedsPerSpM bin

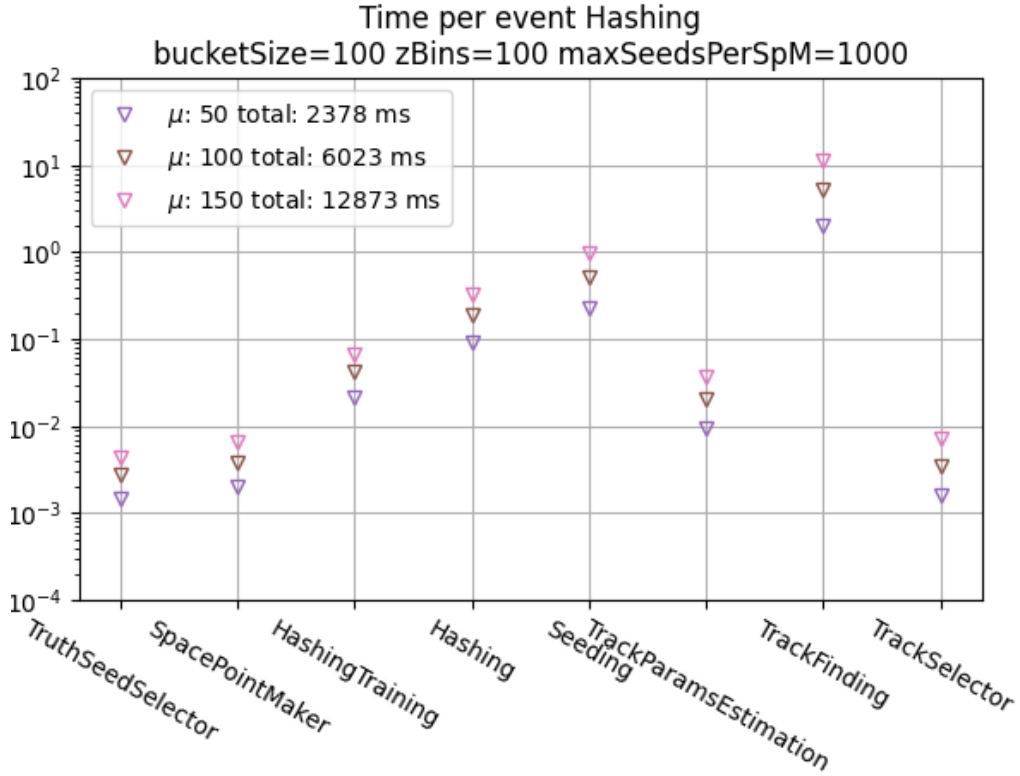
Some of the “Buckets shared Middle Space points” are on the bins after the maxSeedsPerSpM bin

Differences in the bins before maxSeedsPerSpM correspond to lost seeds

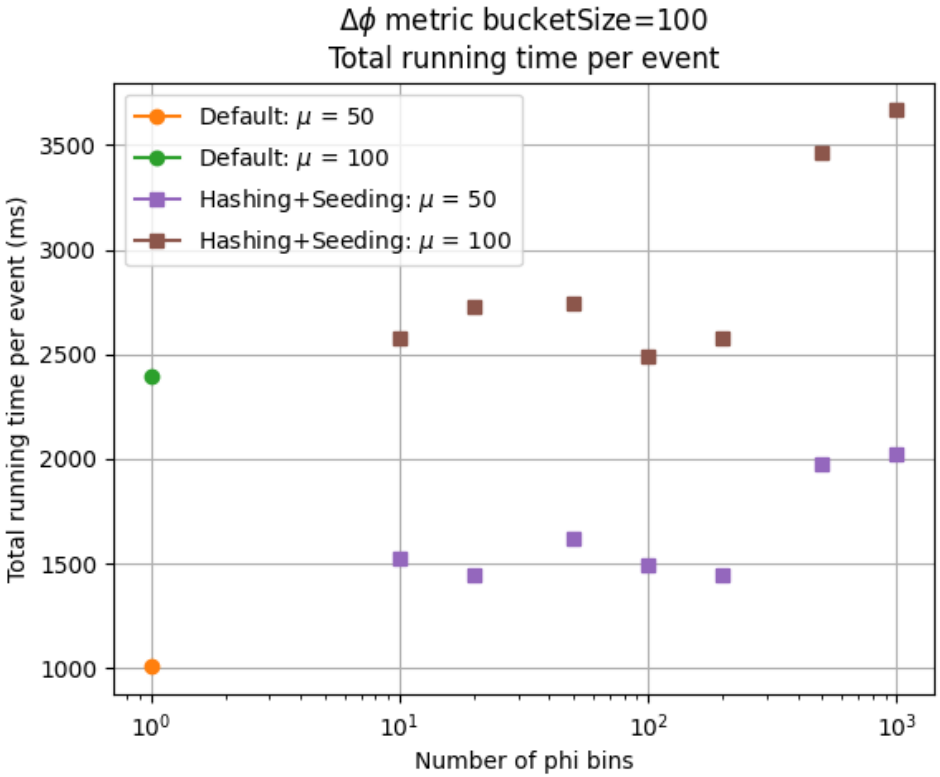
Default nSeeds: 4208
Δφ nSeeds: 6053
ΔR nSeeds: 5300

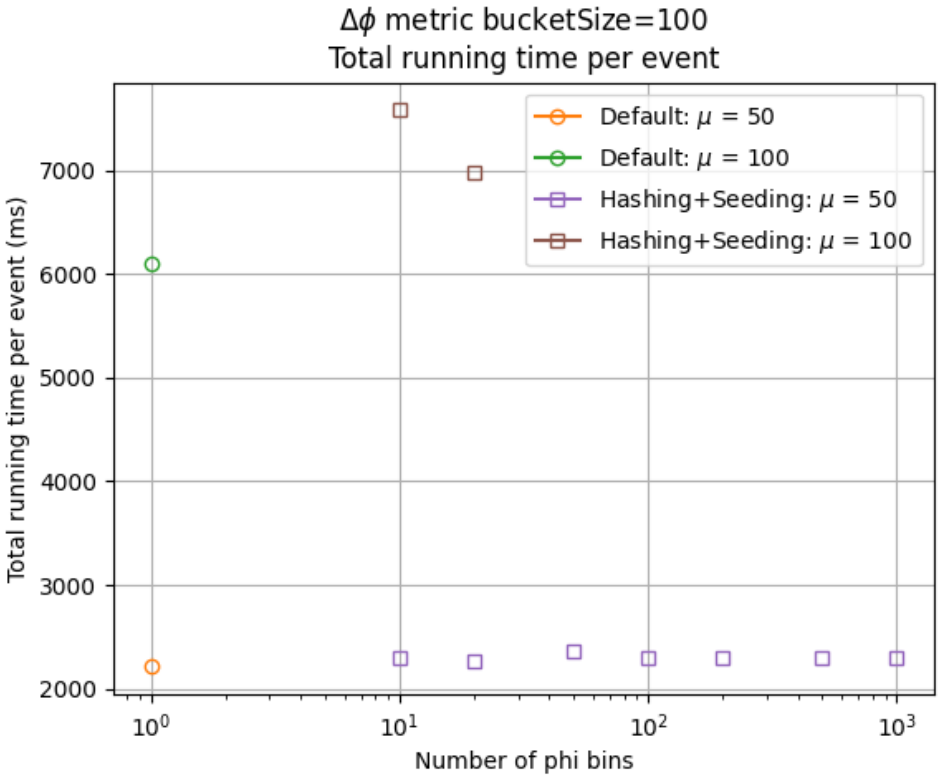
Running time no cut



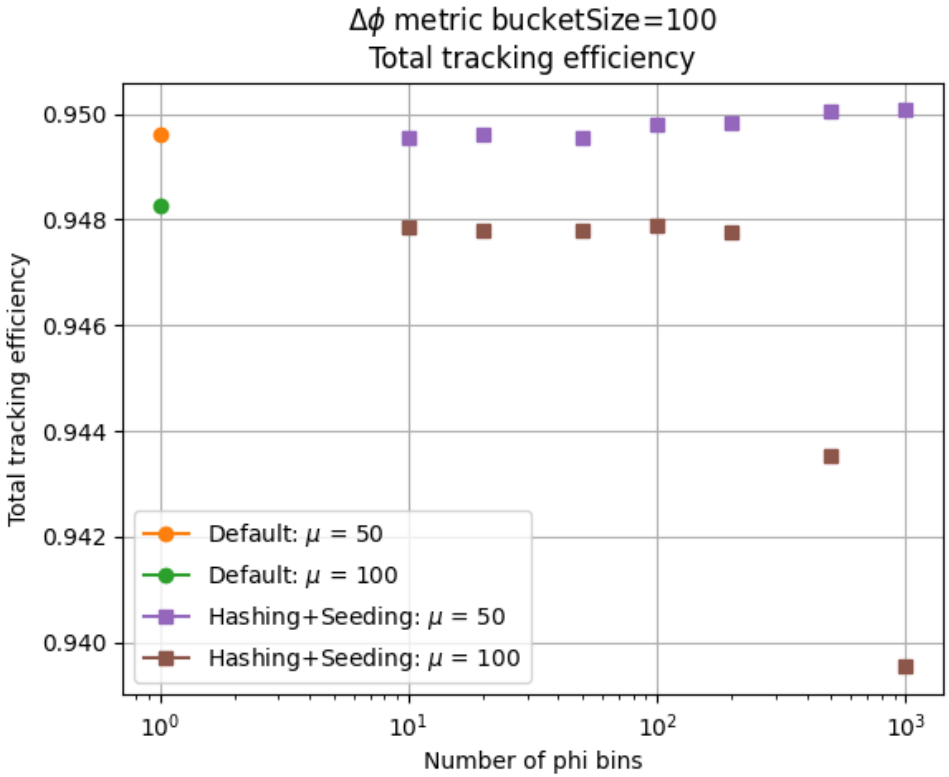


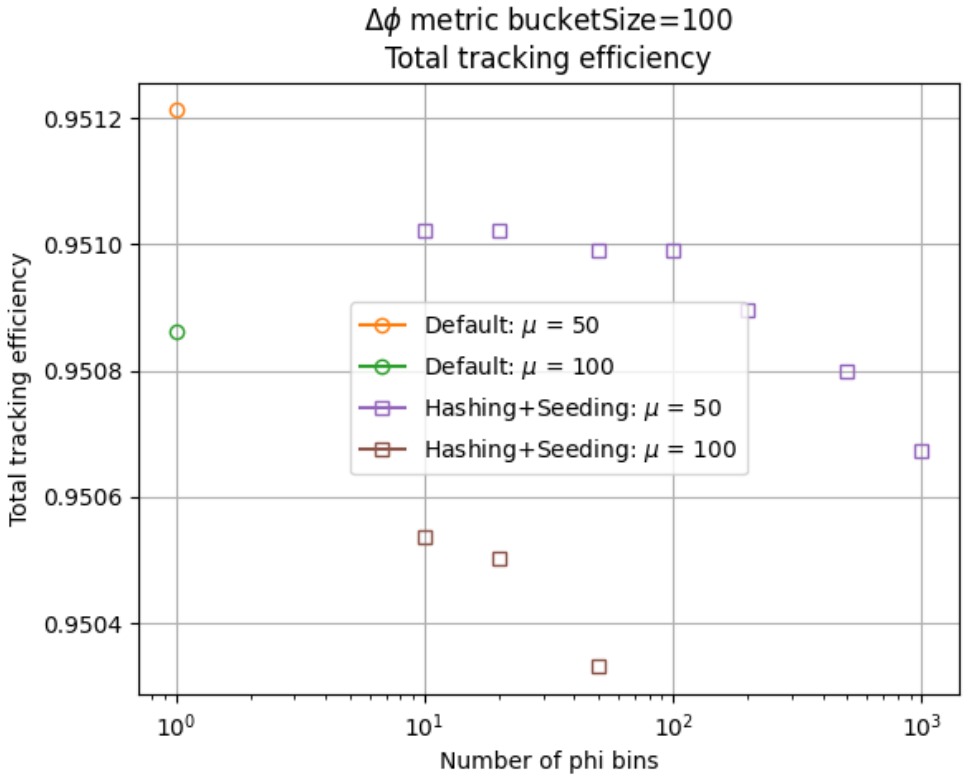
Phi bins: Timing



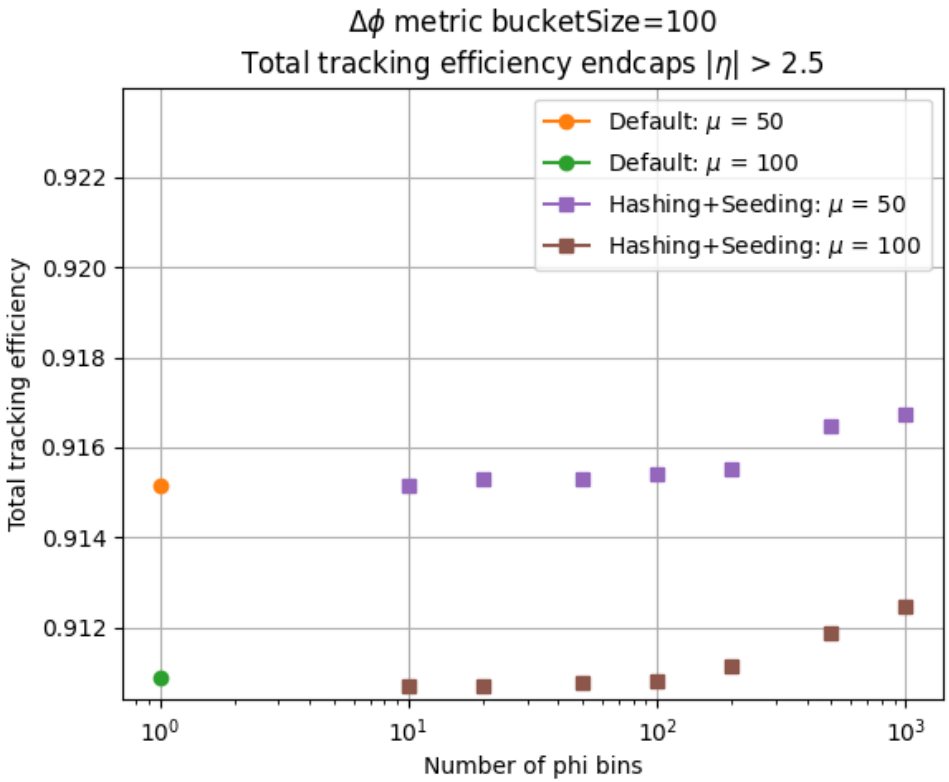


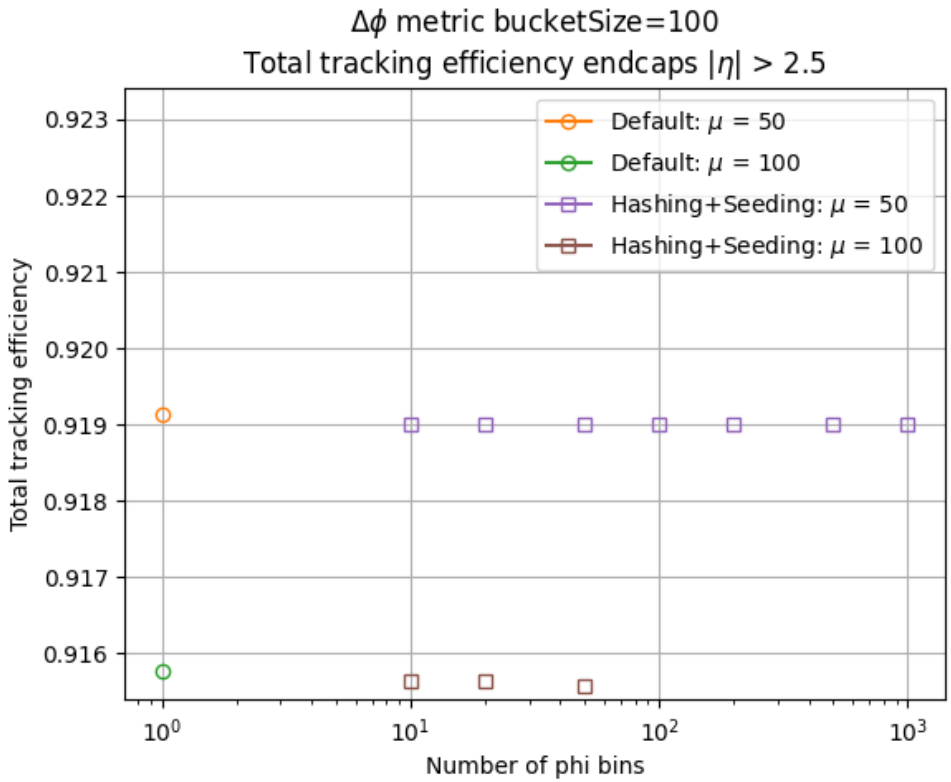
Phi bins: Tracking efficiency



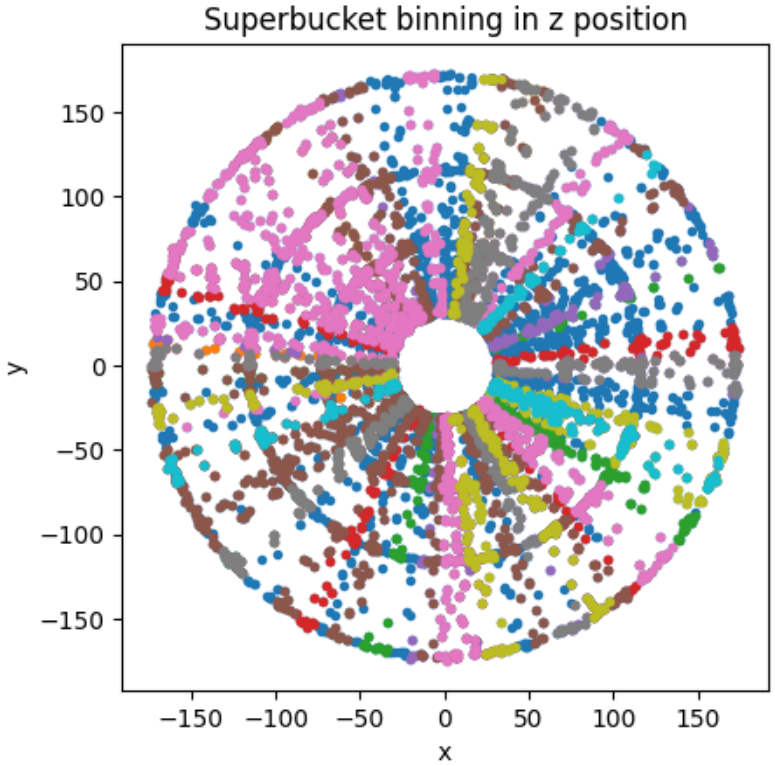


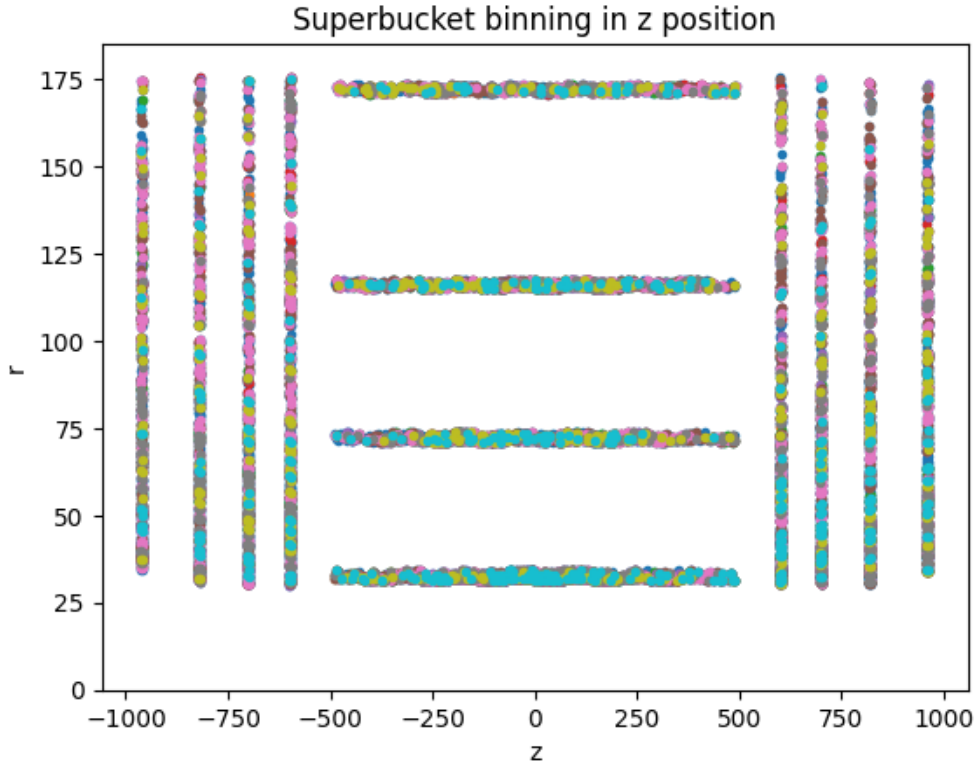
Phi bins: Tracking efficiency



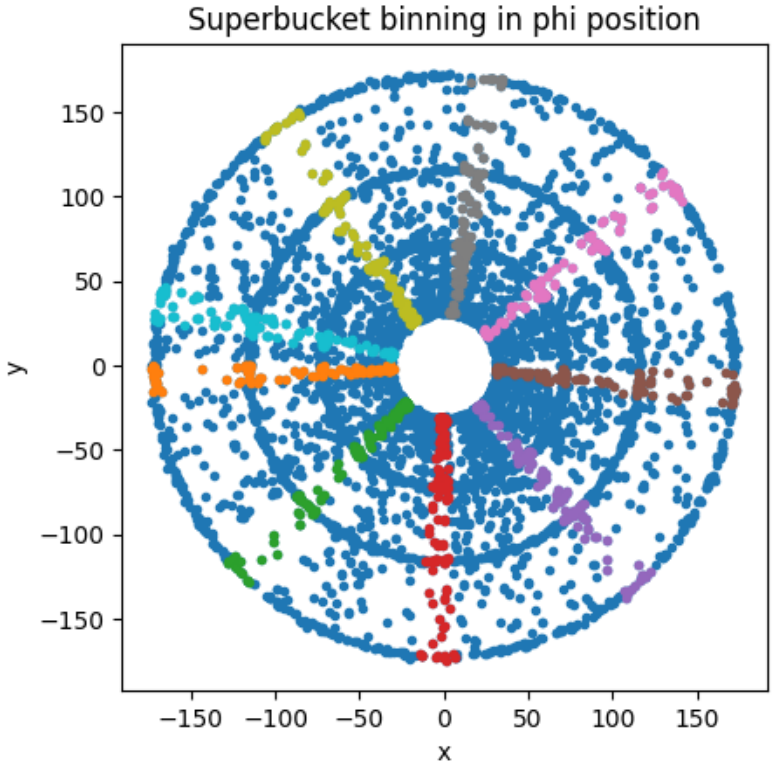


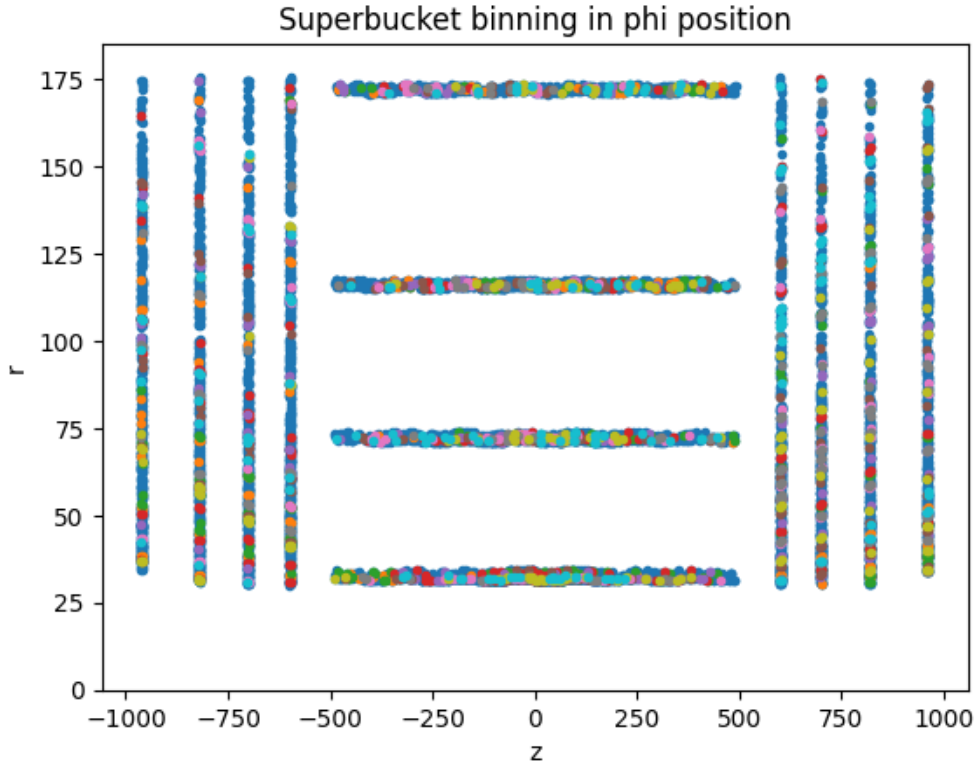
Superbucket binning in Z position





Superbucket binning in Phi position

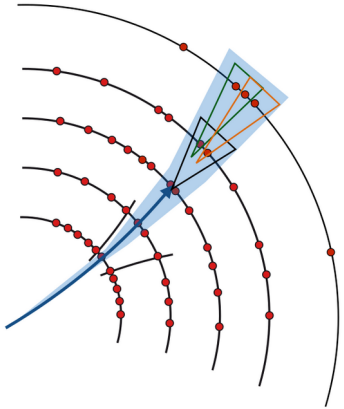


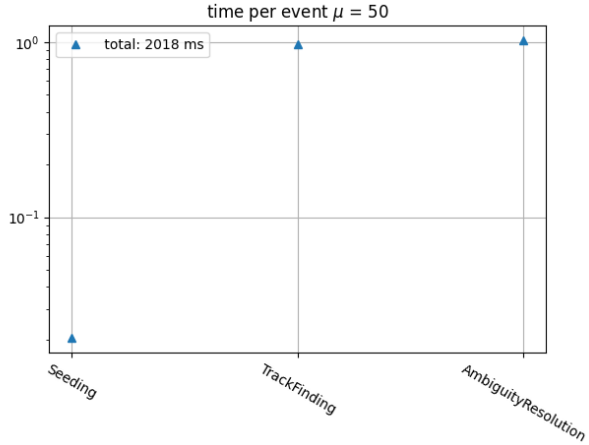


Combinatorial problem

Combinatorial Kalman Filter:

- Several possibilities of expanding the seeds at each layer → need to test them all
- Number of combinations increases exponentially with the number of layers





- **Every seed is expanded:**
  - Less seeds → less tracks → less bad quality and duplicated tracks

ACTS Poor man's Ambiguity resolver

How to get less seeds?

→ Remove the bad ones!

- How?
  - Current: Filter the seeds + detailed optimisation
  - My work: Build the seeds differently