Tracking with Hashing Overview

Comité de Suivi Individuel

6 juin 2023

Jeremy Couthures

Outline

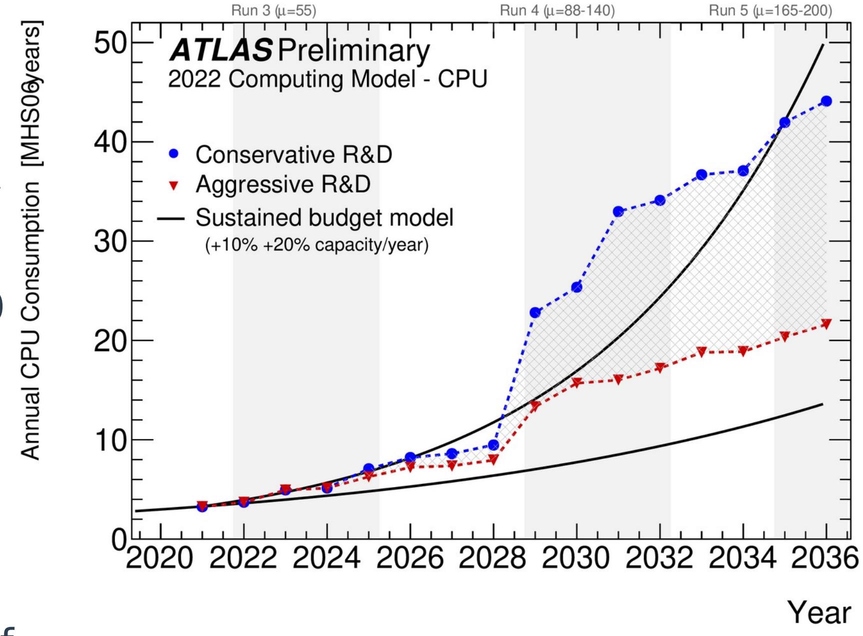
1. **Thesis context: ITk building**
2. **ATLAS Qualification Task**
3. **Formations**
4. **Thesis work:**
   - Baseline
   - Method
   - Results
   - Where are we going?

1. Thesis context: ITk building

- **New Inner Tracker (ITk) for ATLAS HL-LHC:**
  - Wider coverage: |η| < 4
  - Higher granularity
  - ATLAS CPU previsions: need to improve ***tracking*** performance significantly
- **LAPP is producing 75% of the OB Types 0 (5000 pigtails, 400 PP0 boards) and will be integrating 25% of the local supports(*)**

  → Qualification Task on Production database for Types 0

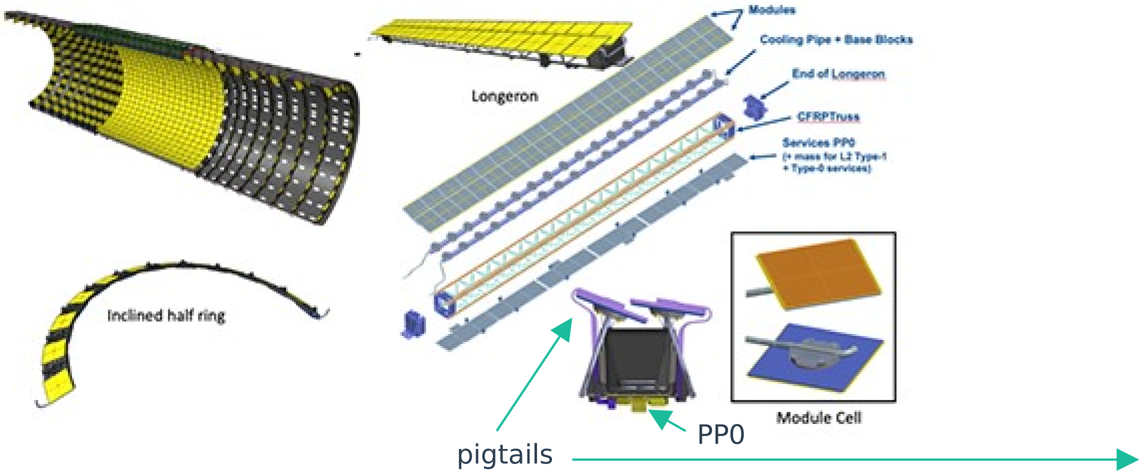
  → In 2024, participation to the electrical testing of modules at CERN or LAPP



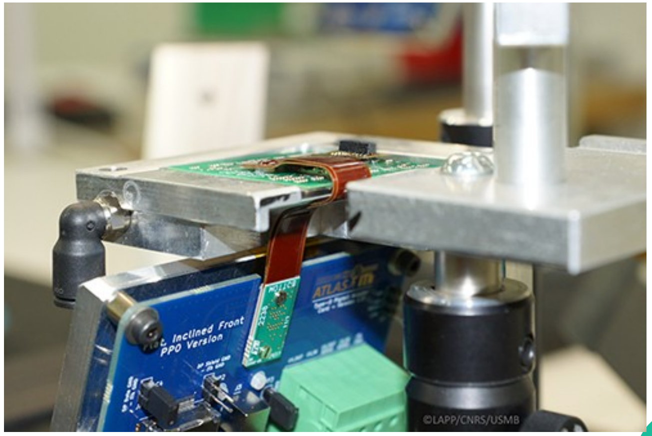
(*)With LPSC and CPPM

ITk Pixel Detector Overview

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



https://lapp.in2p3.fr/spip.php?article3307

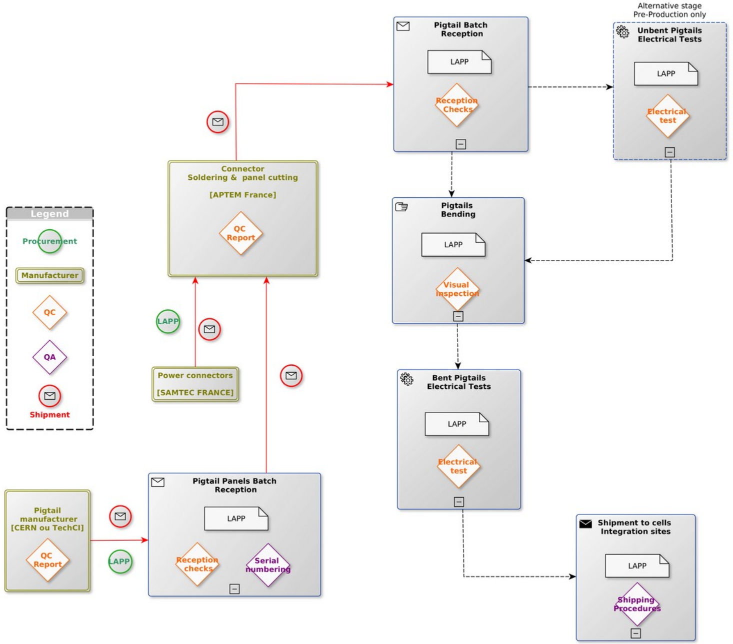


2. ATLAS Qualification Task: PDB

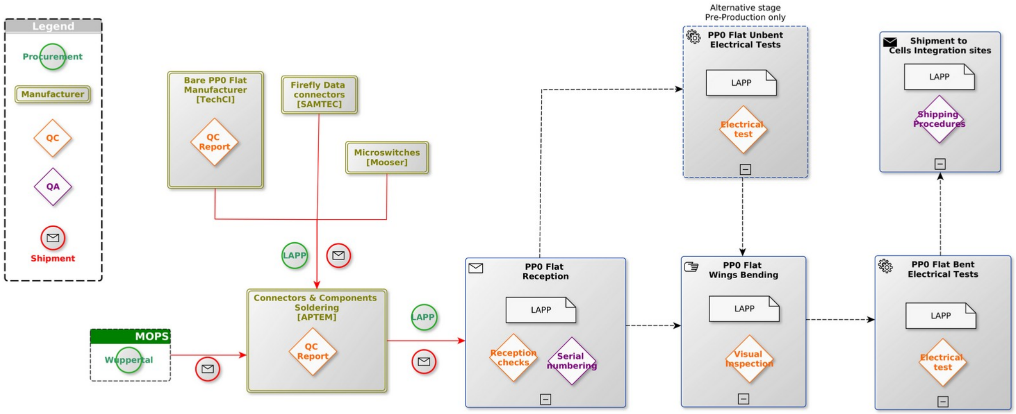
- **ATLAS Qualification Task:**
  - 1 year, 6 months of work
  - Allows to publish as ATLAS member
- **ATLAS Production Database**
  - Create components, store quality control data, track shipping, API
  - Interface not easy to use, slow, error prone and poor scalability

- **Qualification Task:**
  - Creation of a dedicated “LAPP Types 0 Web app”
    to ease the registration in the database
- **Help other laboratories to build their own**

ATLAS Qualification Task: Types 0 web app



Pigtails production flow



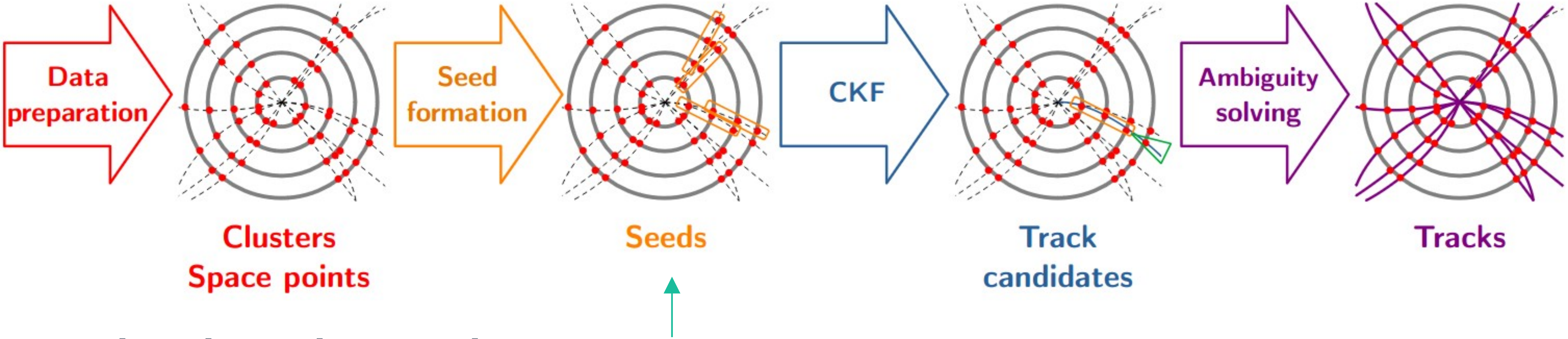
PP0 production flow

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

3. Formations

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

4. ATLAS Tracking



- **What do we hope to improve?**
  - Seeds' efficiency: reconstruct at least one seed per track
  - Seeds' purity: reconstruct only tracks' seeds
  - Seeds' redundancy: reconstruct just enough seeds per track
- **How? Improving timing and group similar Space Points**

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

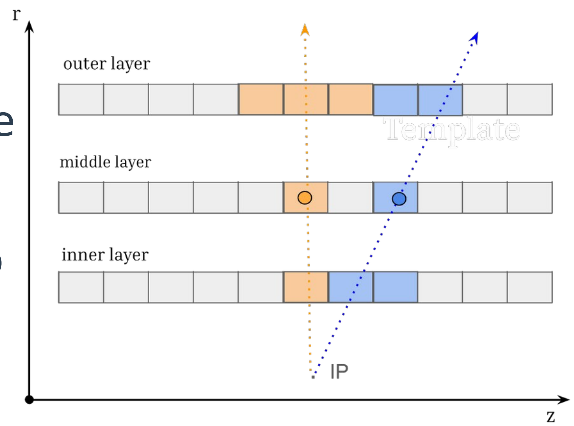


Fig. 26 Representation of the search for triplet combinations in the (r, z) plane. The bins used in the search are represented in different colors.

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

Baseline: Setup

acts github

Run 4:
$\langle\mu\rangle = 140$

100 tt events

$|\eta| \leq 4$

pT > 1GeV

Pythia8
FATRAS

Not using Geant4:
→ no secondaries

Generic detector: (toy detector)

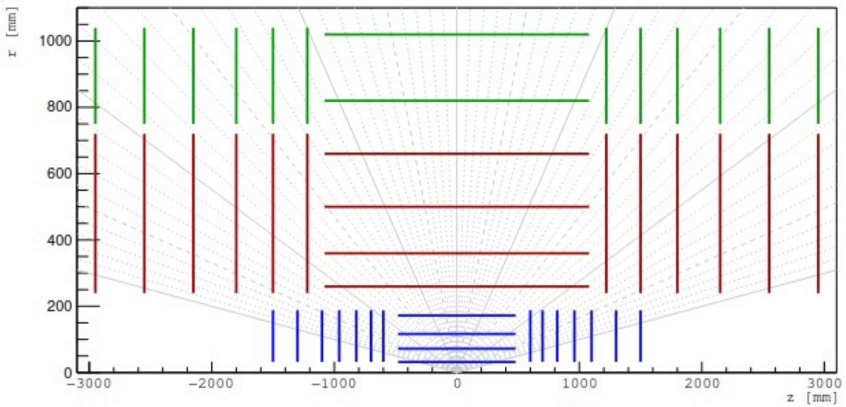
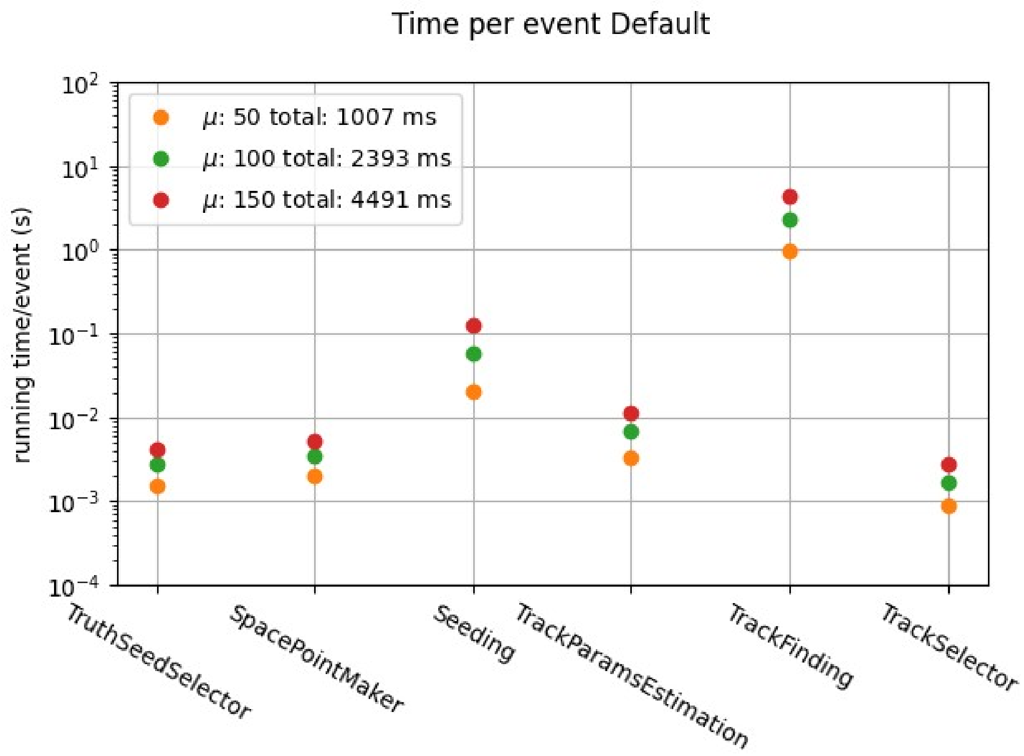
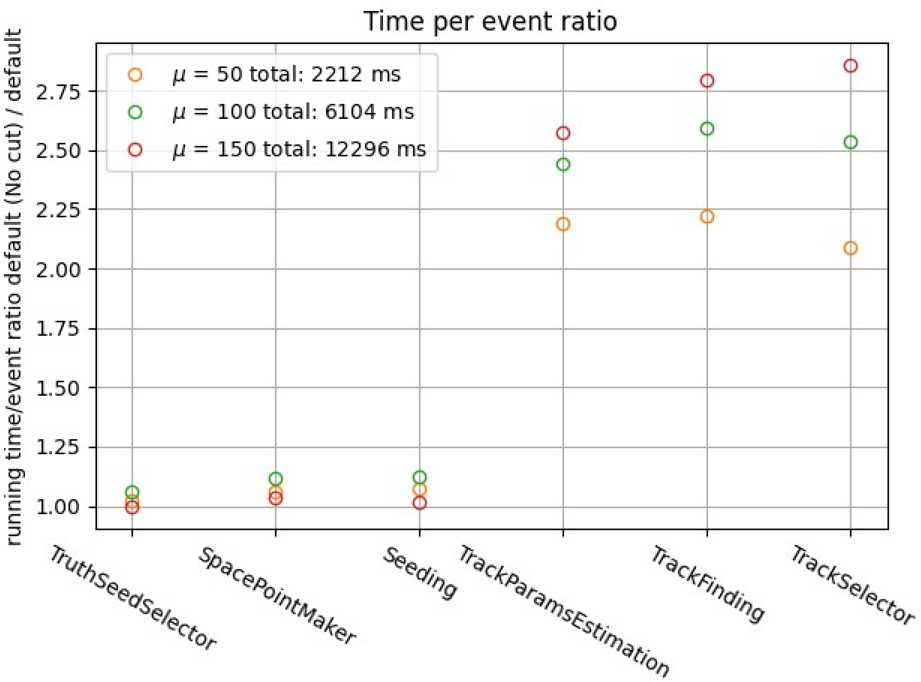


Fig. 1 Sketch of the TrackML detector as used in both the "Accuracy" and "Throughput" phase. Vertical lines indicate disks while horizontal lines indicate cylinders, all with the *z* axis as axis of revolution. Three different sub detectors build the overall detector setup: a central pixel system (blue), enclosed by first a short strip (red) and then a long strip detector (green).

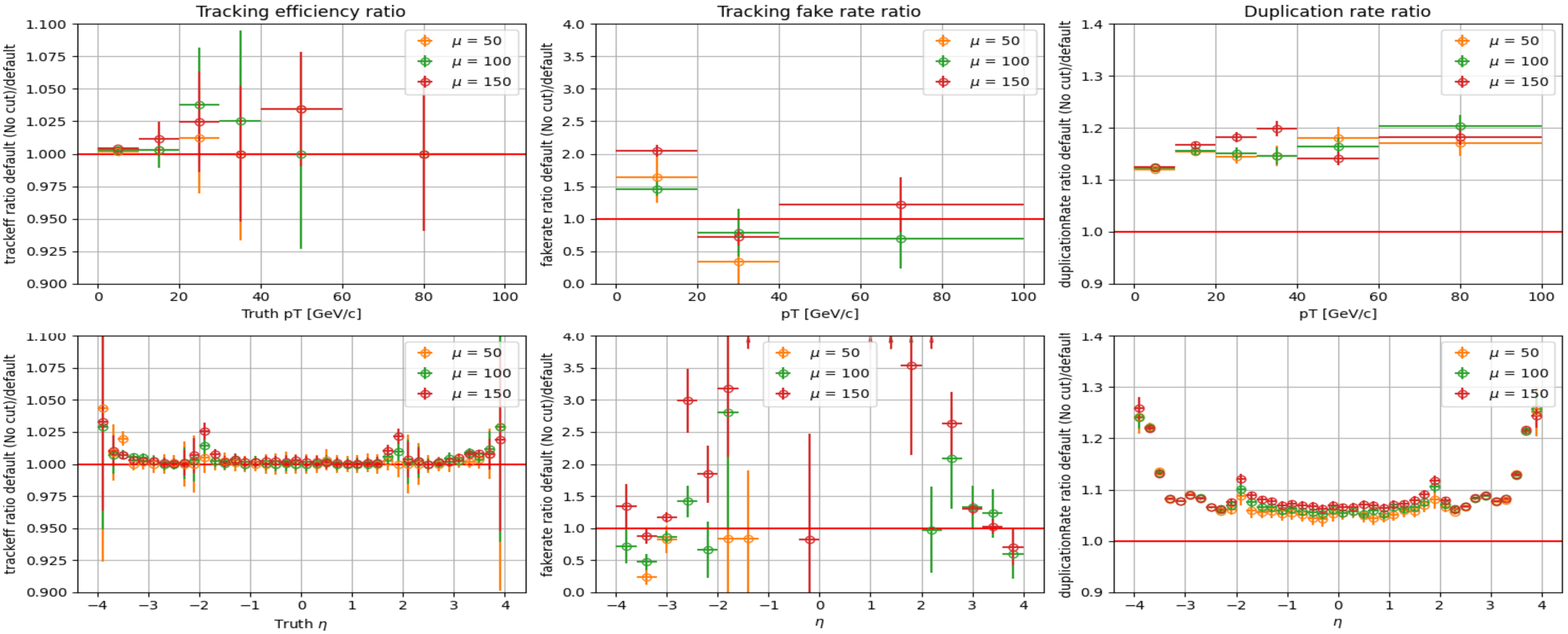
https://arxiv.org/pdf/2105.01160.pdf

ACTS performance: Timing/event





ACTS performance: Physics



Method

Hashing:

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

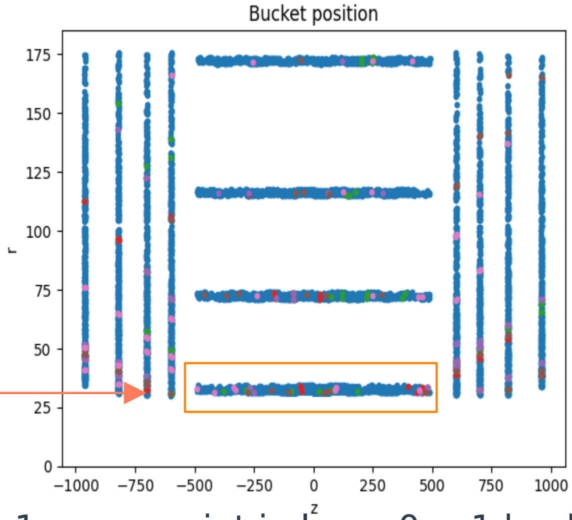
Algorithm used:

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

- Machine Learning algorithm type:
  - k Nearest Neighbors (unsupervised)
  - Random based

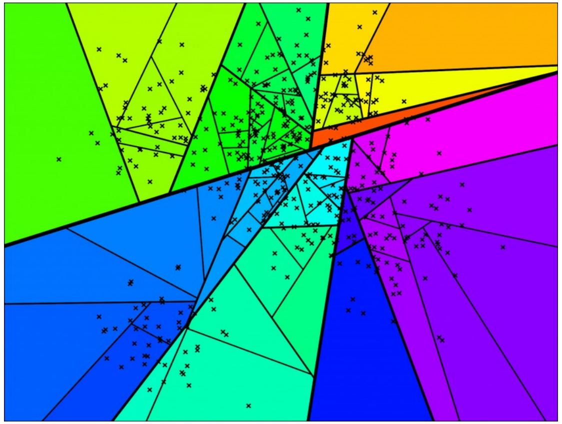
- Number of Neighbors (bucket size)
- Starting from layer 0

- Use the distance between the points
  → need to define a (relevant) metric



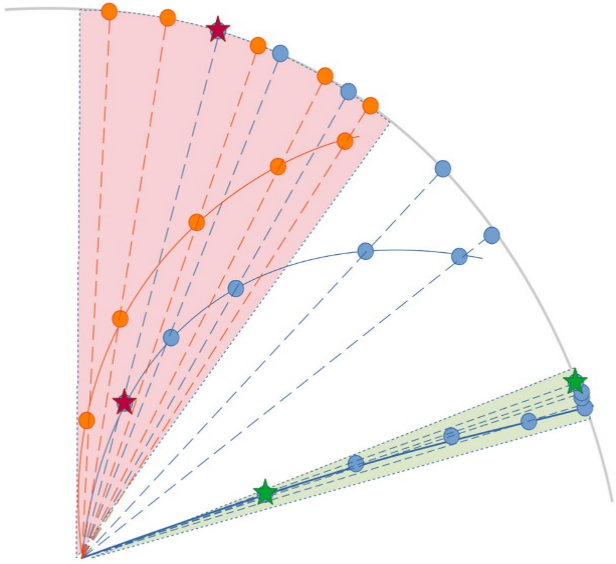
1 space point in layer 0 → 1 bucket

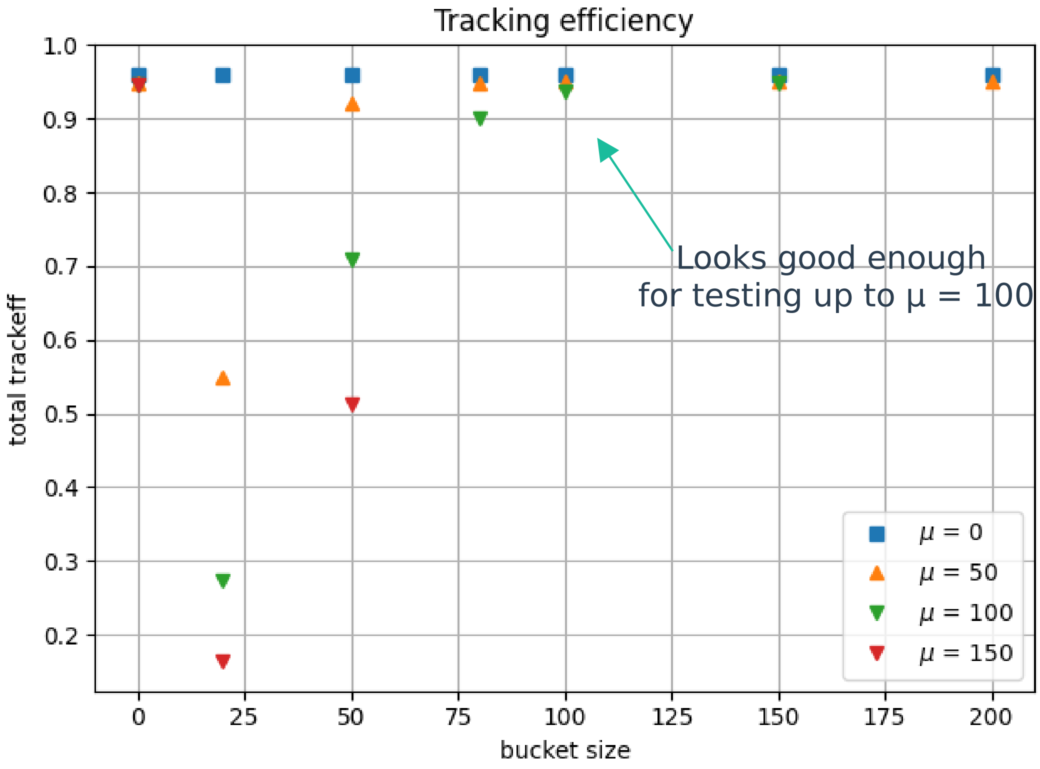
2D Space separation by Annoy



Metric and bucket size

Metric: Δφ



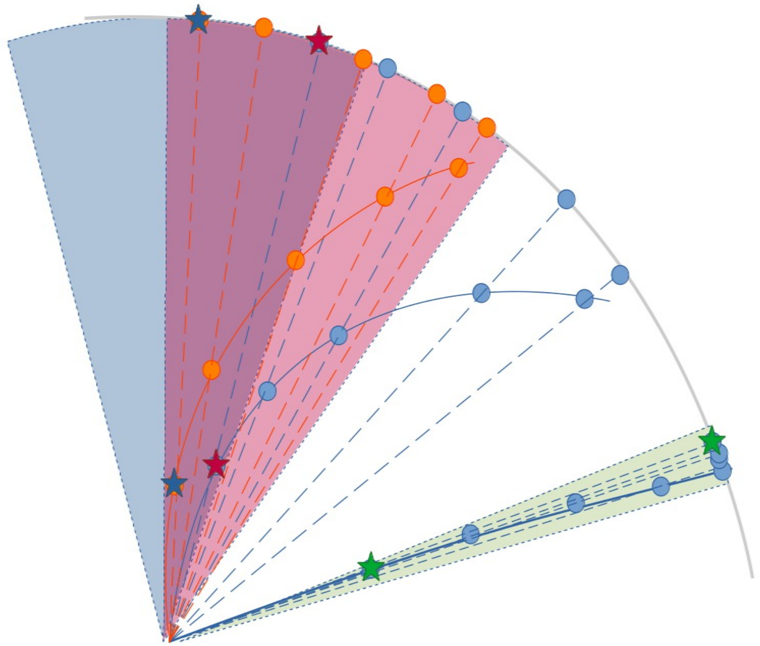


Looks good enough
for testing up to μ = 100

Hashing and overlap

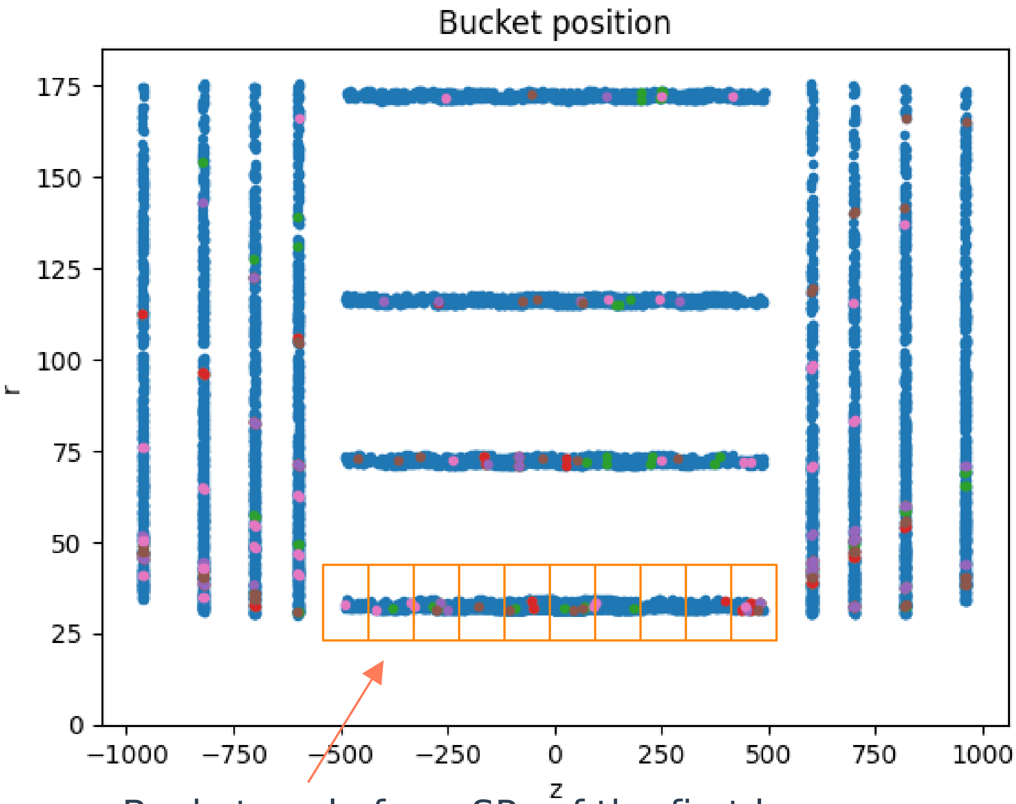
- **Hashing introduces overlaps:**
  - The same seed can be reconstructed in several buckets (14 times in average)

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



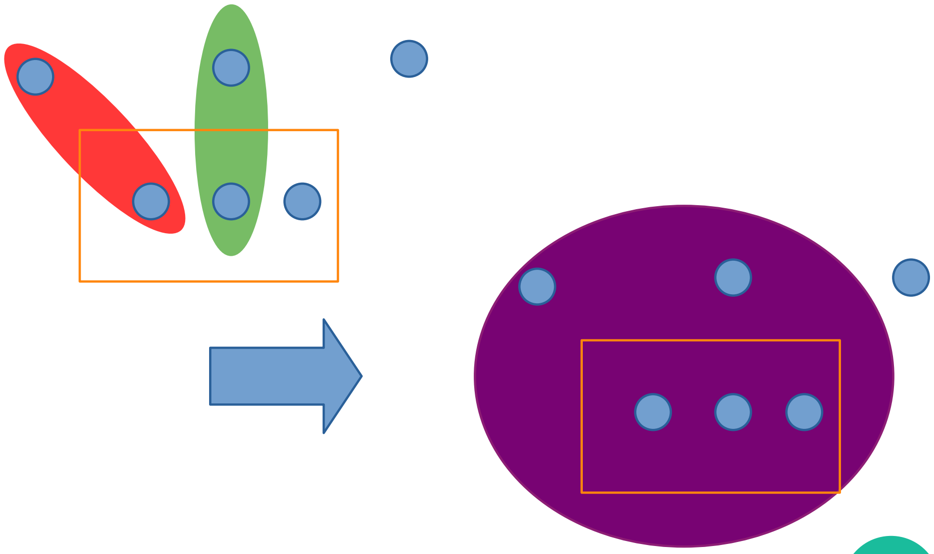
Group buckets → less overlap

Super buckets and binning

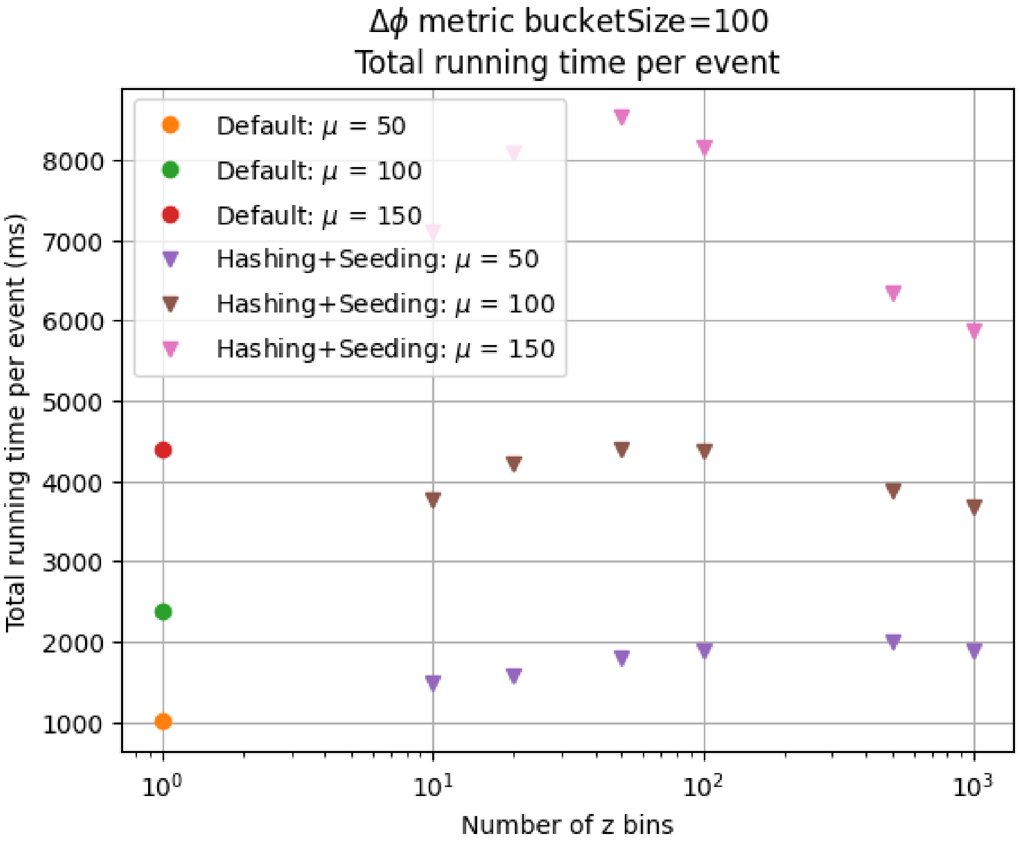


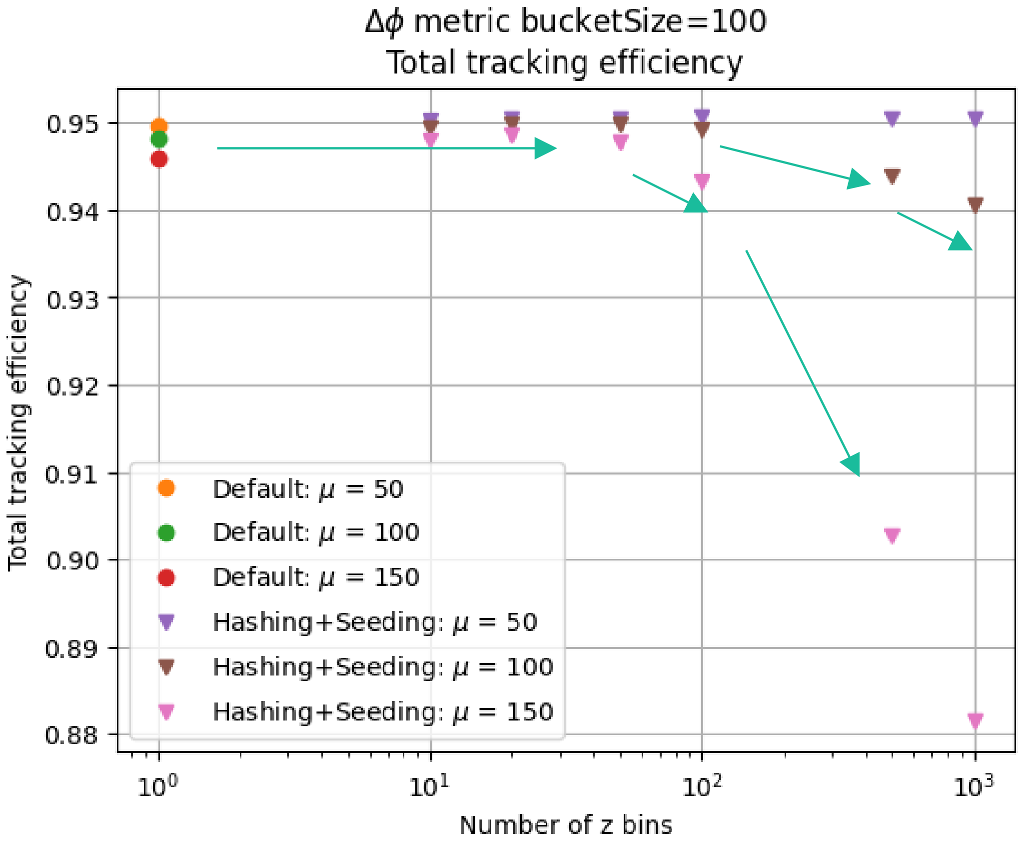
Buckets only from SPs of the first layer

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

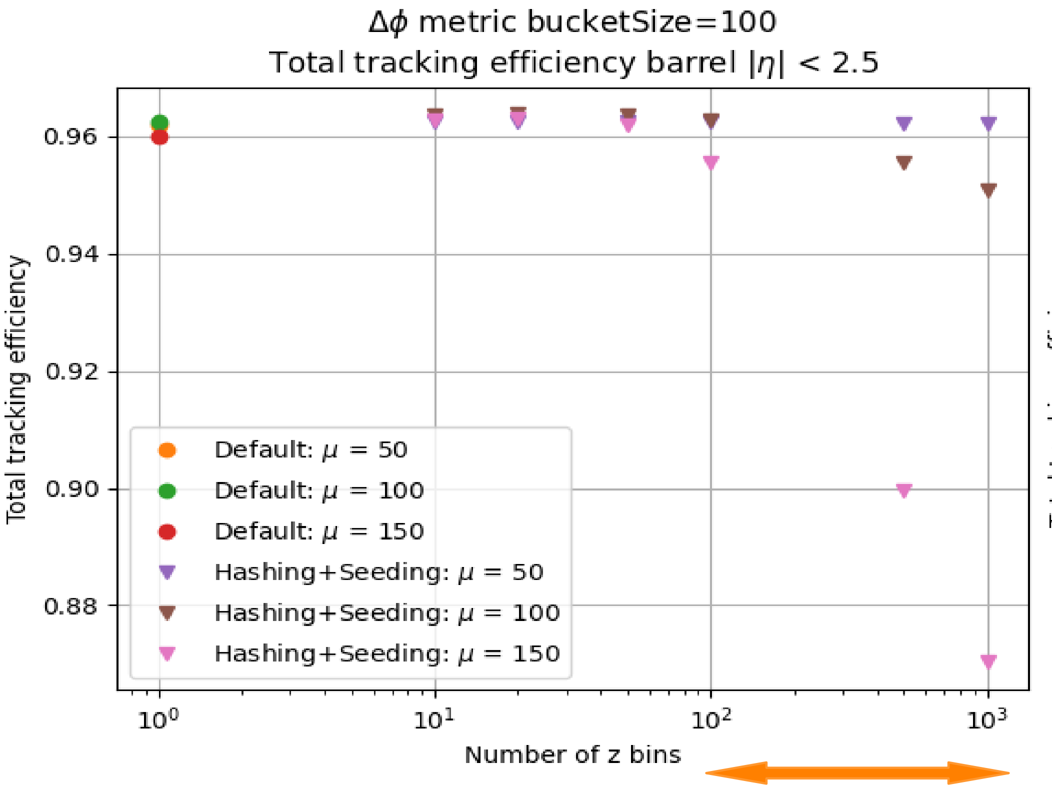


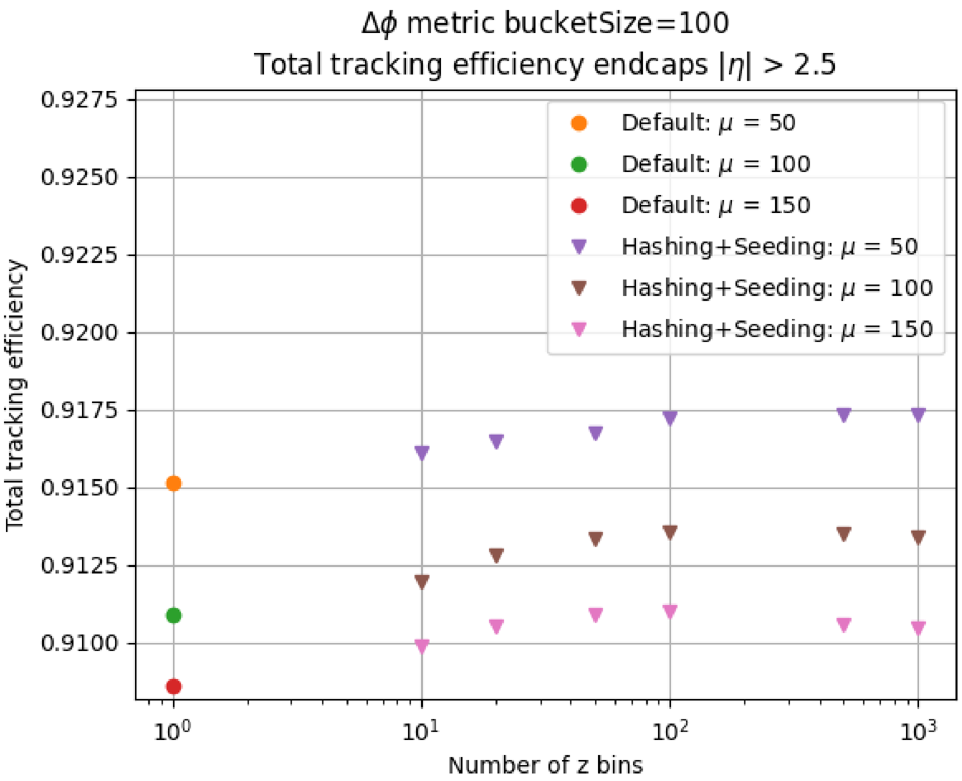
Hashing performance: Timing and efficiency





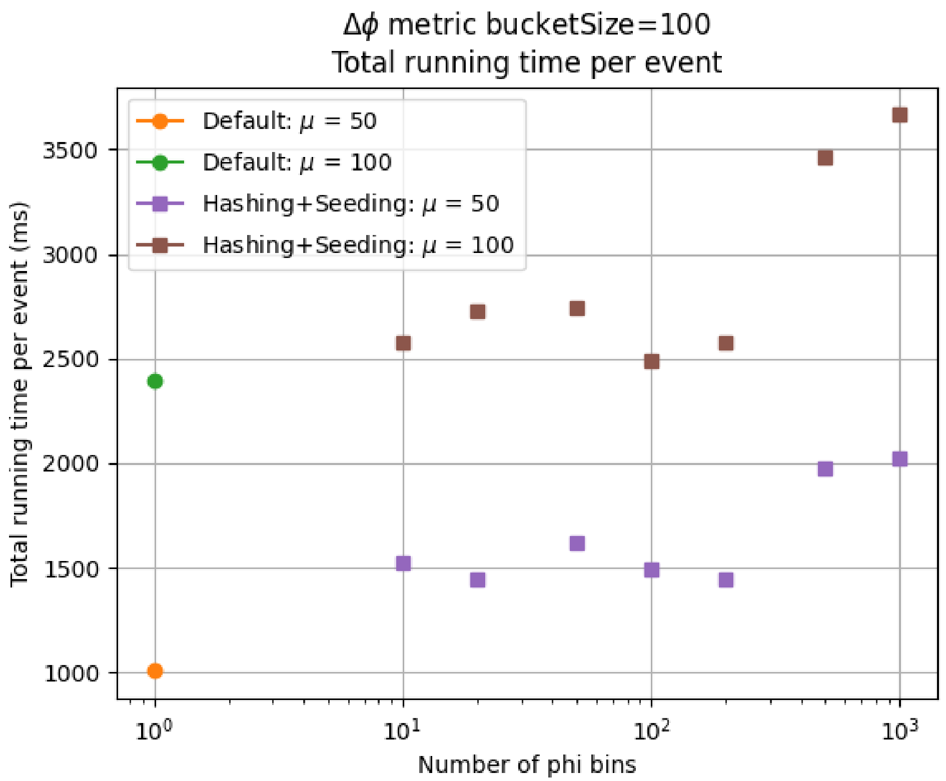
Hashing performance: Efficiency (detailed)

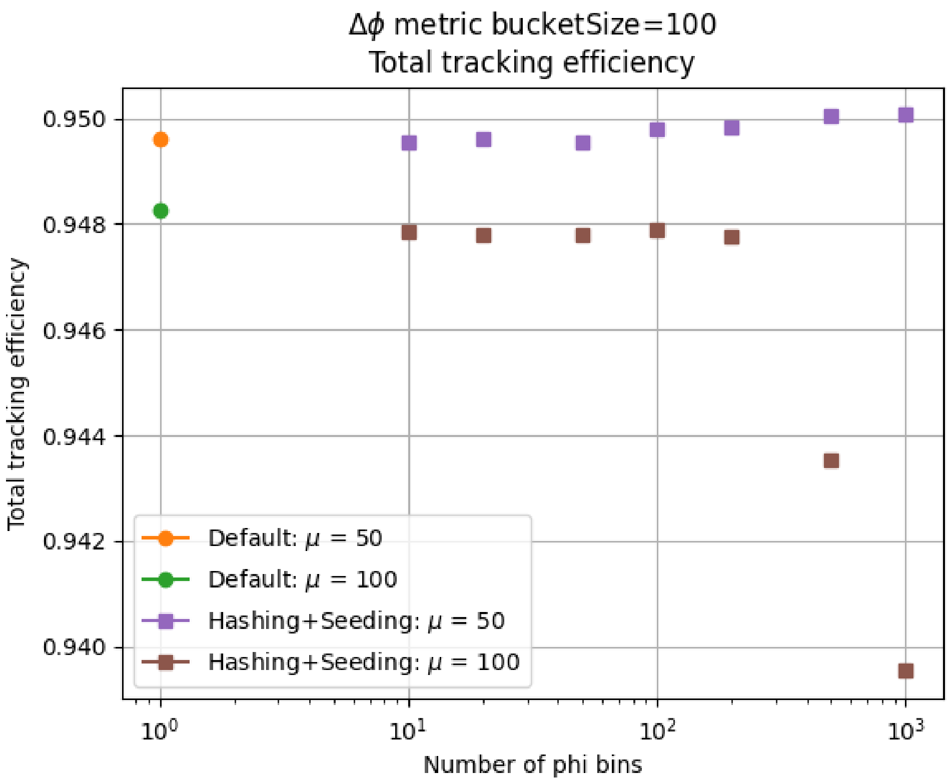




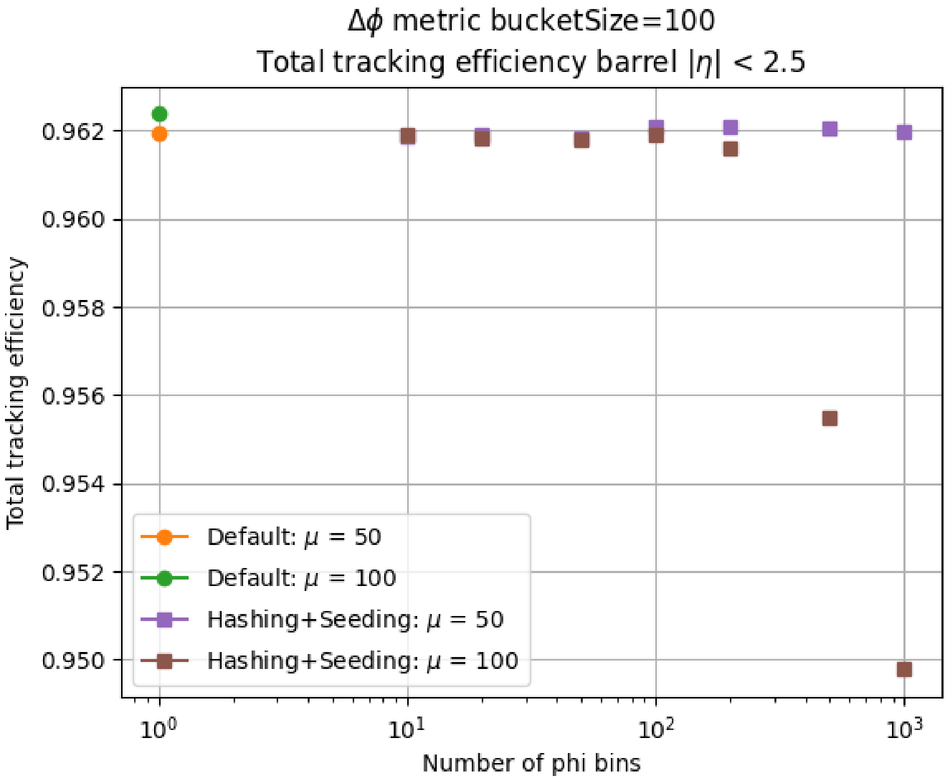
MaxSeedsPerSpM cut dominates

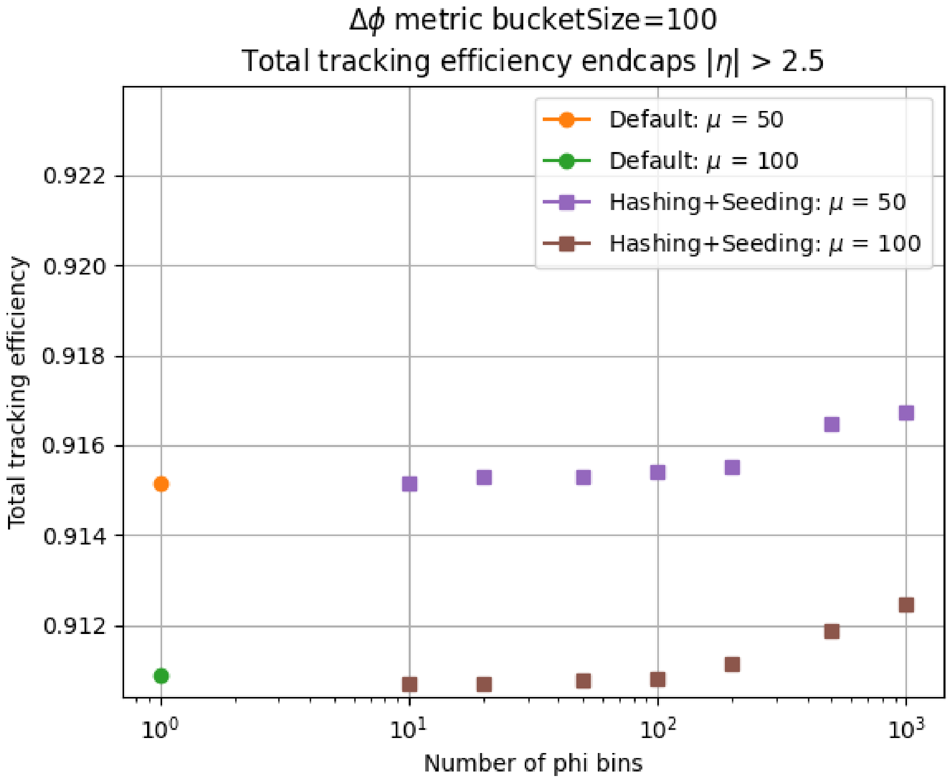
Hashing φ bins: Timing and efficiency



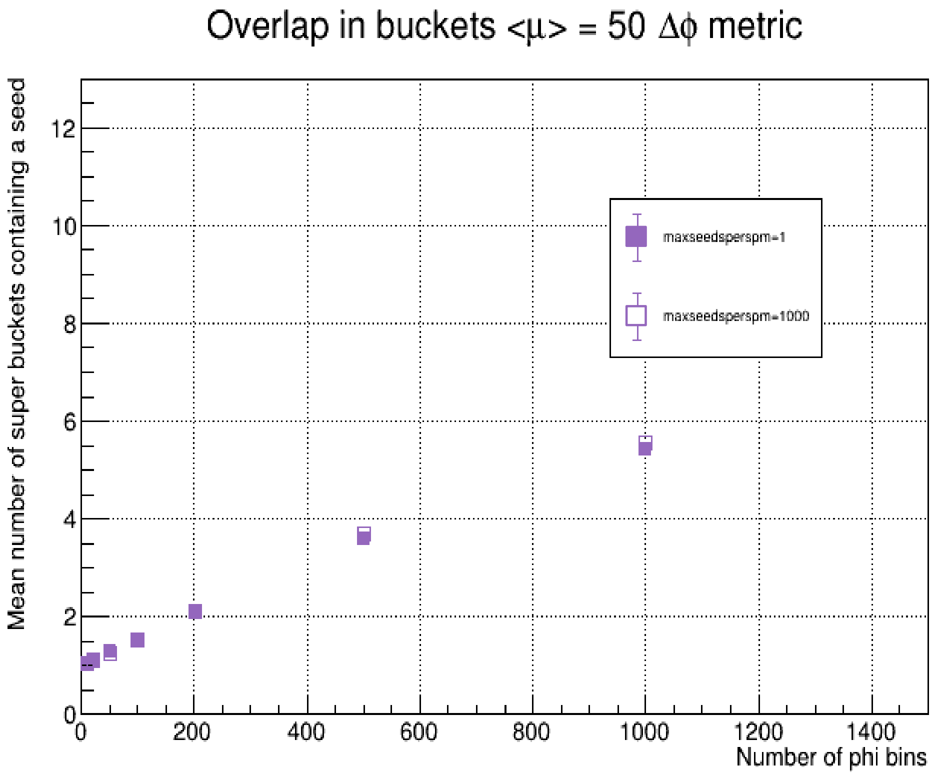


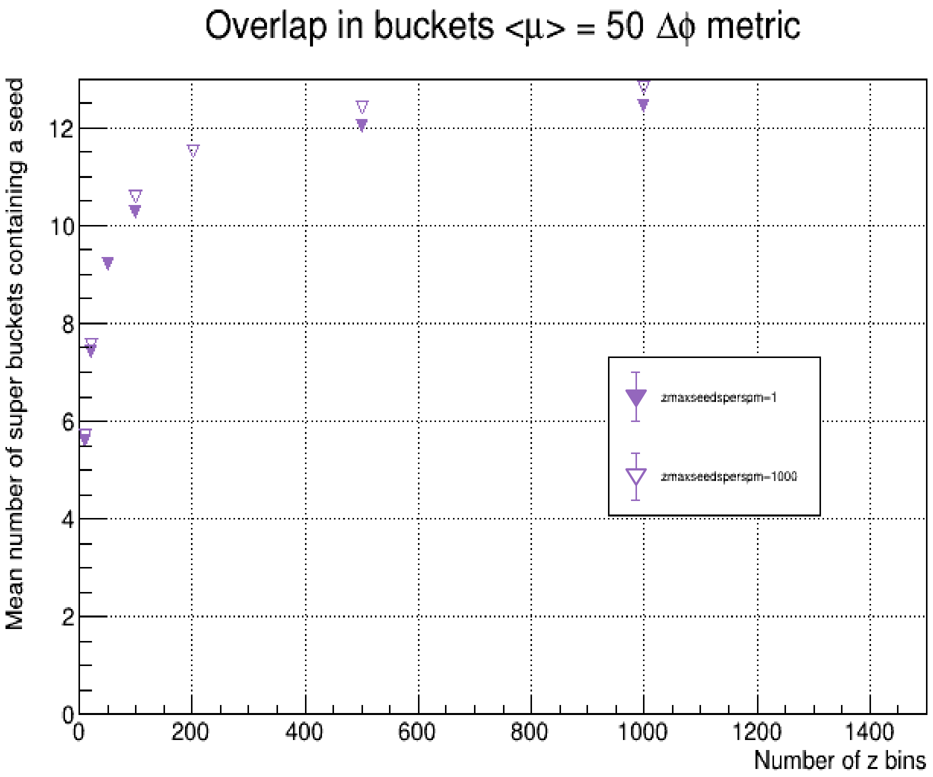
Hashing φ bins: Efficiency (detailed)





Overlap in buckets





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

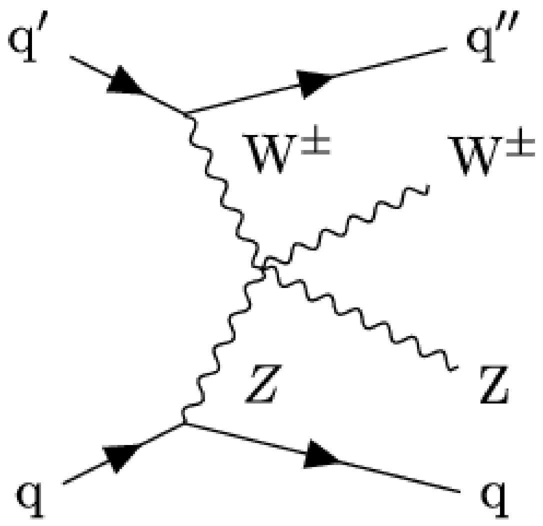
Next:

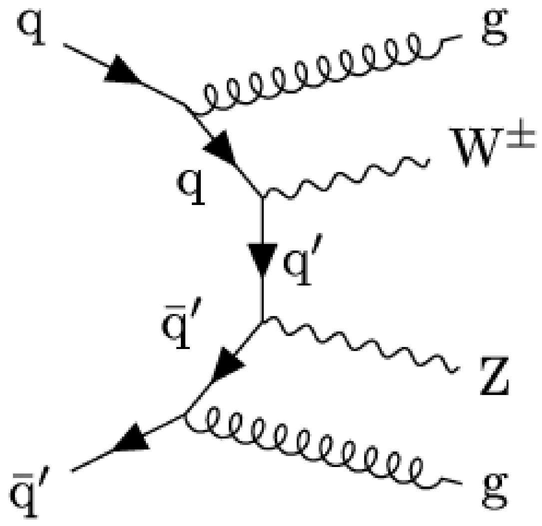
- **ODD, full sim with Geant4 (secondaries), ITk, metric, binning in (z, φ), changing bucket size with detector region, Cluster shape?**
- **Summer School**
- **September: Quark/Gluon Tagging with ML**

Backup

Quark/Gluon tagging: Physics motivation 1

Vector Boson Scattering: Quartic Gauge coupling measurement



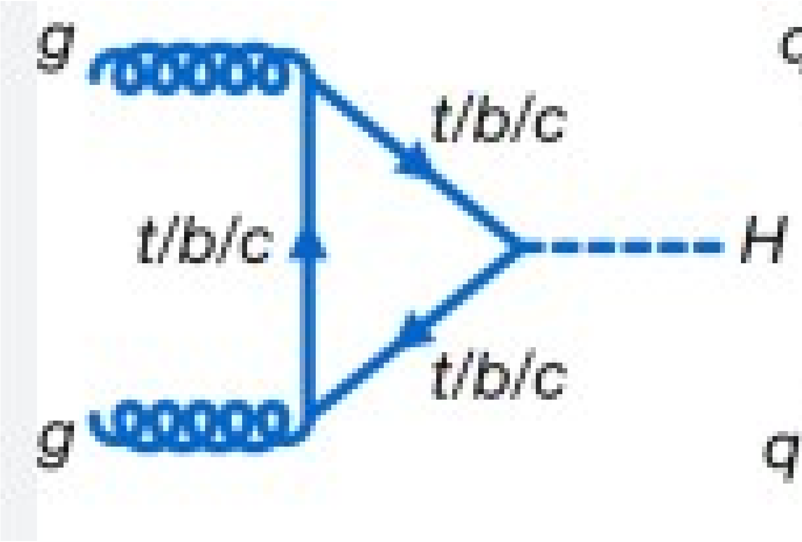


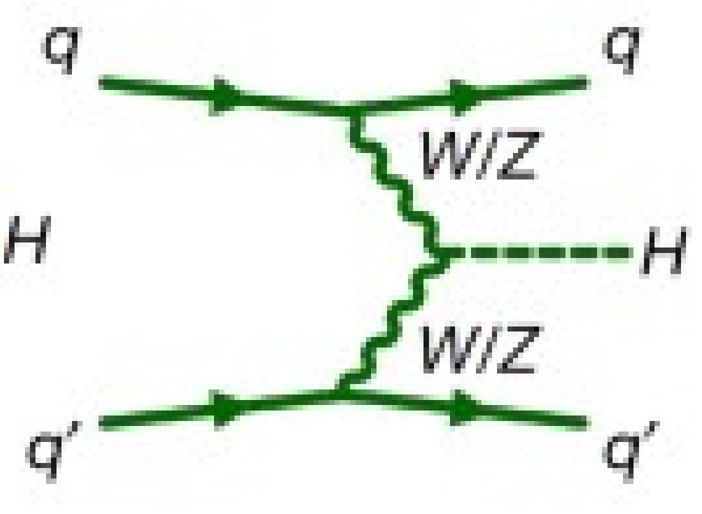
SIGNAL: Electroweak process
Forward quark jets

Background: QCD Process
randomly distributed gluon jets

Quark/Gluon tagging: Physics motivation 2

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



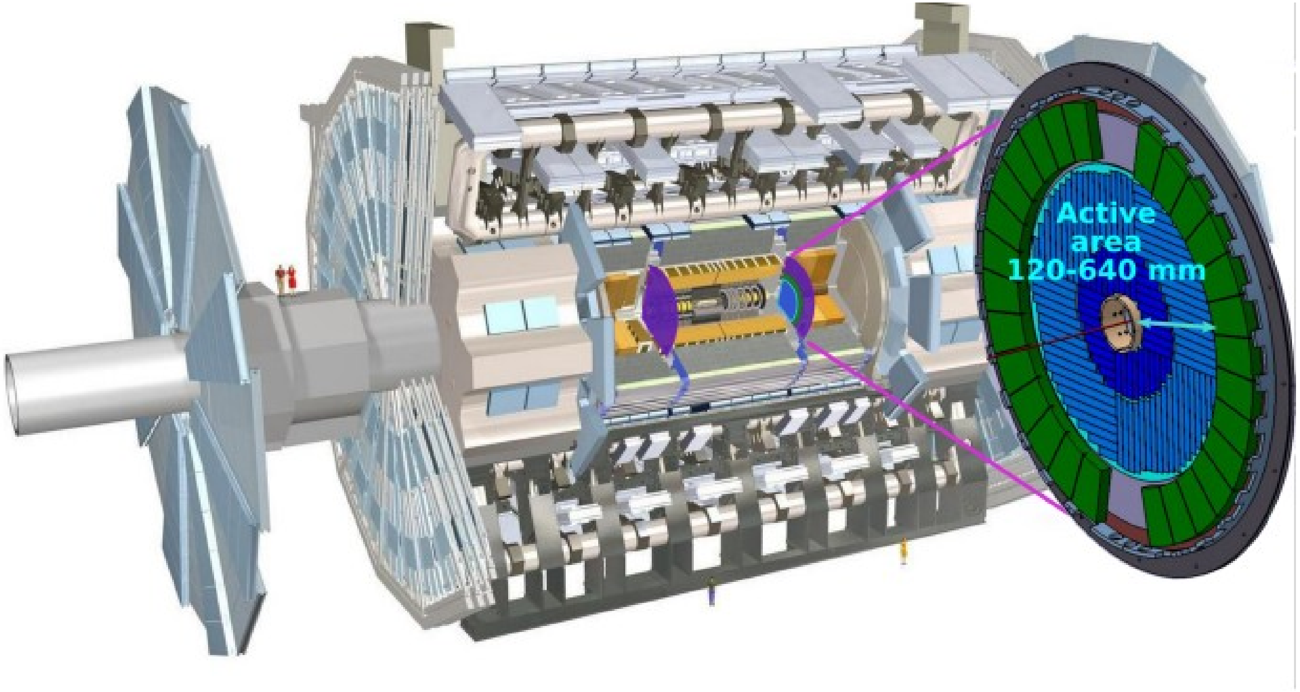


QCD Process: randomly distributed gluons jets

VBF process: forward quark jets

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

Nouveau détecteur ITk + HGTD:



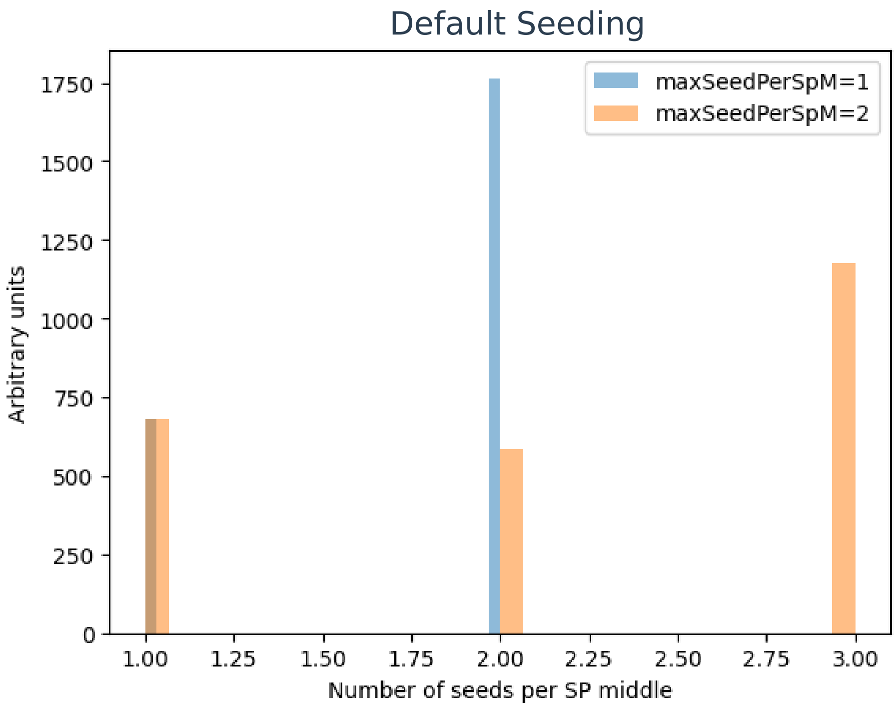
Seed finder configuration

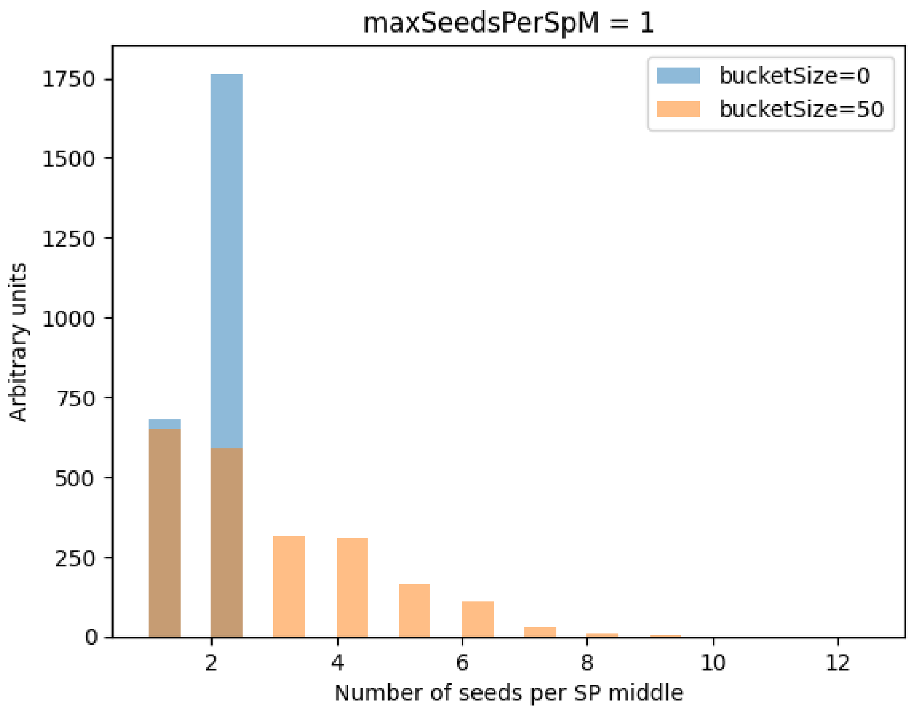
```
SeedfinderConfigArg = SeedfinderConfigArg(
        r=(None, 200 * u.mm),  # rMin=default, 33mm
        deltaR=(1 * u.mm, 60 * u.mm),
        collisionRegion=(-250 * u.mm, 250 * u.mm),
        z=(-2000 * u.mm, 2000 * u.mm),
        maxSeedsPerSpM=1,
        sigmaScattering=5,
        radLengthPerSeed=0.1,
        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

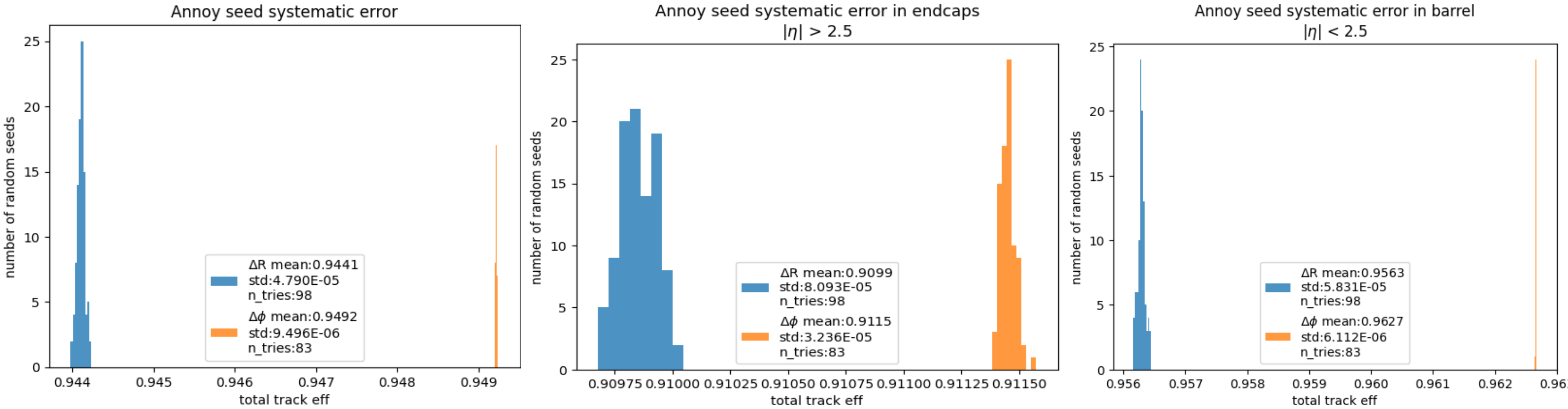
MaxSeedsPerSpM cut vs Hashing





→ Hashing get through the cut

Annoy random seed systematic error



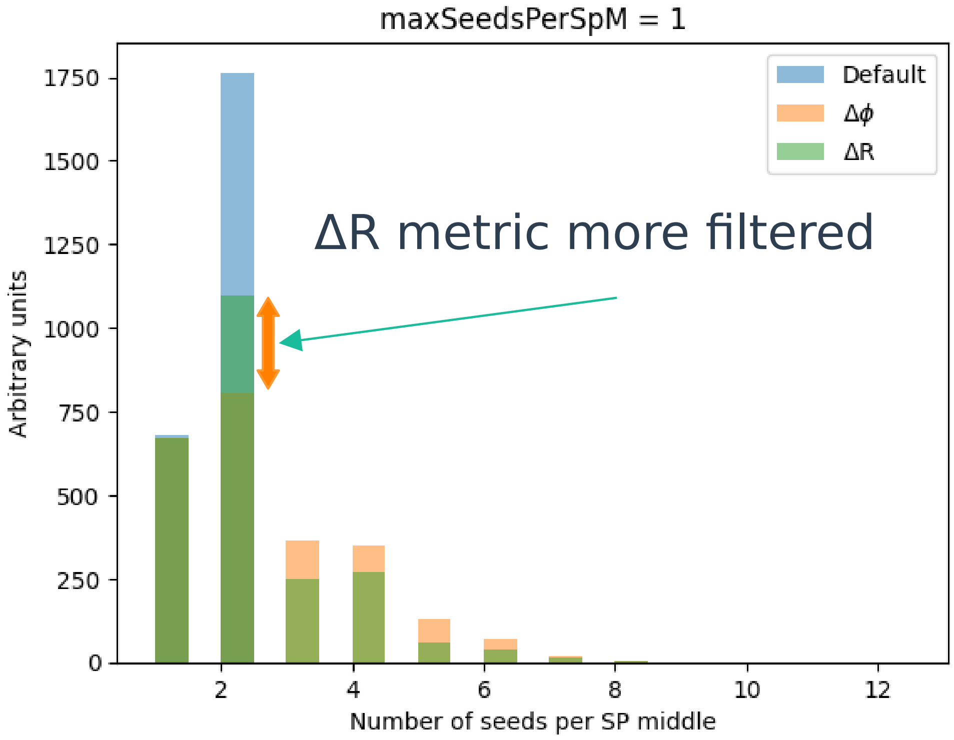
1000 events
in each try

BucketSize: 100
Mu: 50

Δφ is better

MaxSeedsPerSpM and ΔR metric

On 1 event:



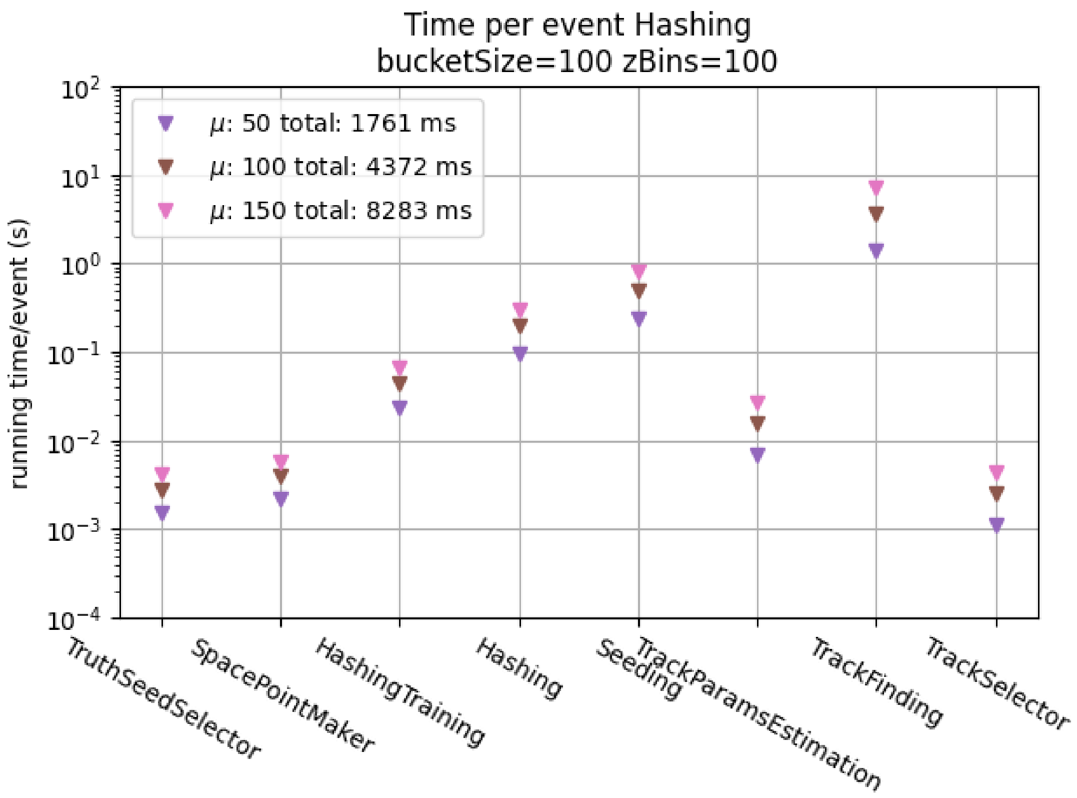
Filtered Middle Space points are on the maxSeedsPerSpM bin

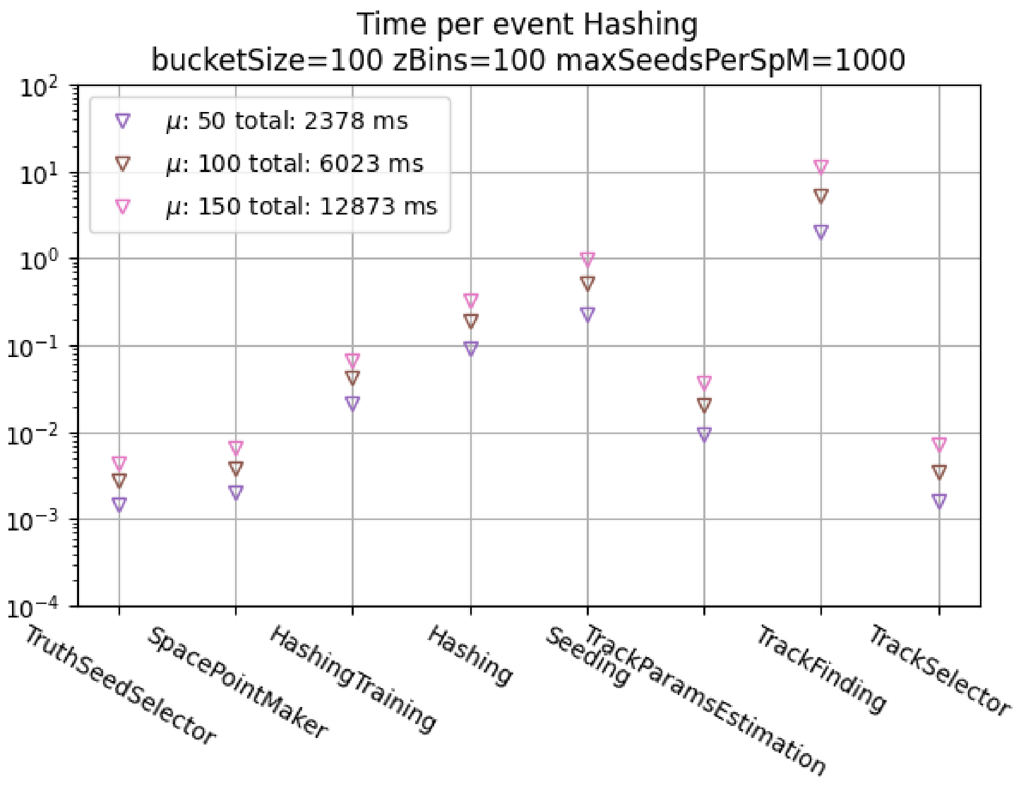
Some of the “Buckets shared Middle Space points” are on the bins after the maxSeedsPerSpM bin

Differences in the bins before maxSeedsPerSpM correspond to lost seeds

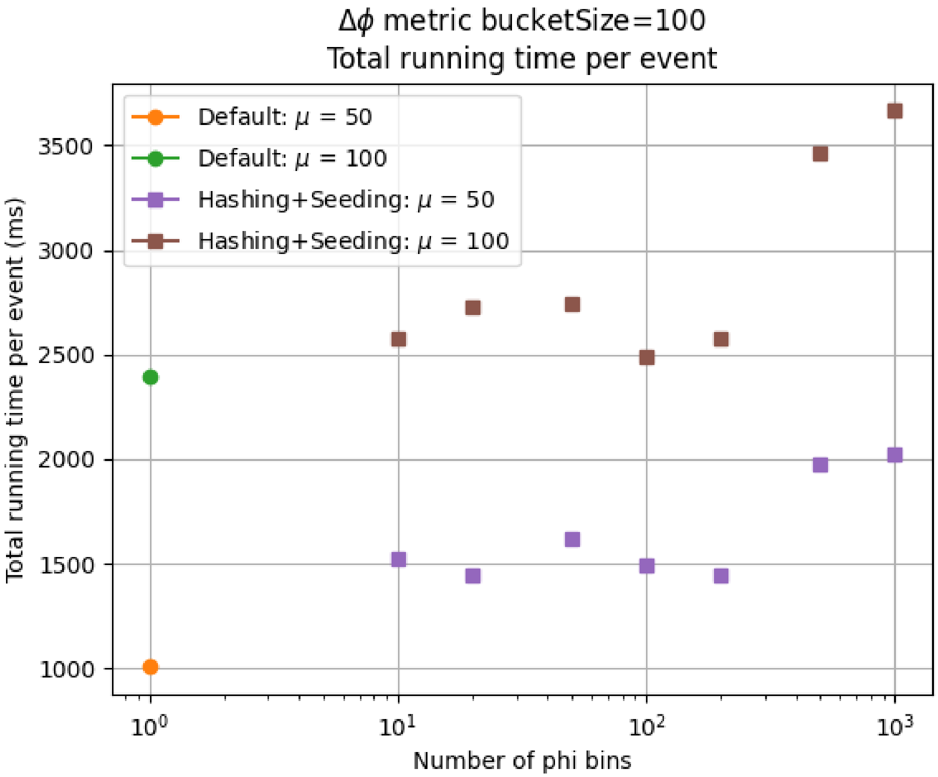
Default nSeeds: 4208
Δφ nSeeds: 6053
ΔR nSeeds: 5300

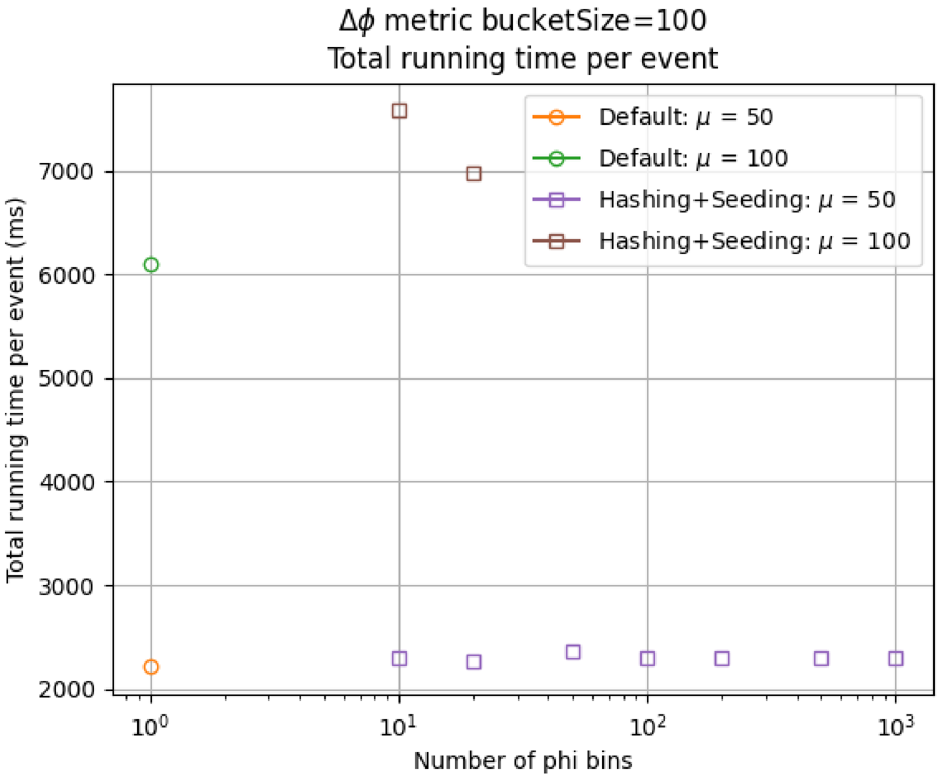
Running time no cut



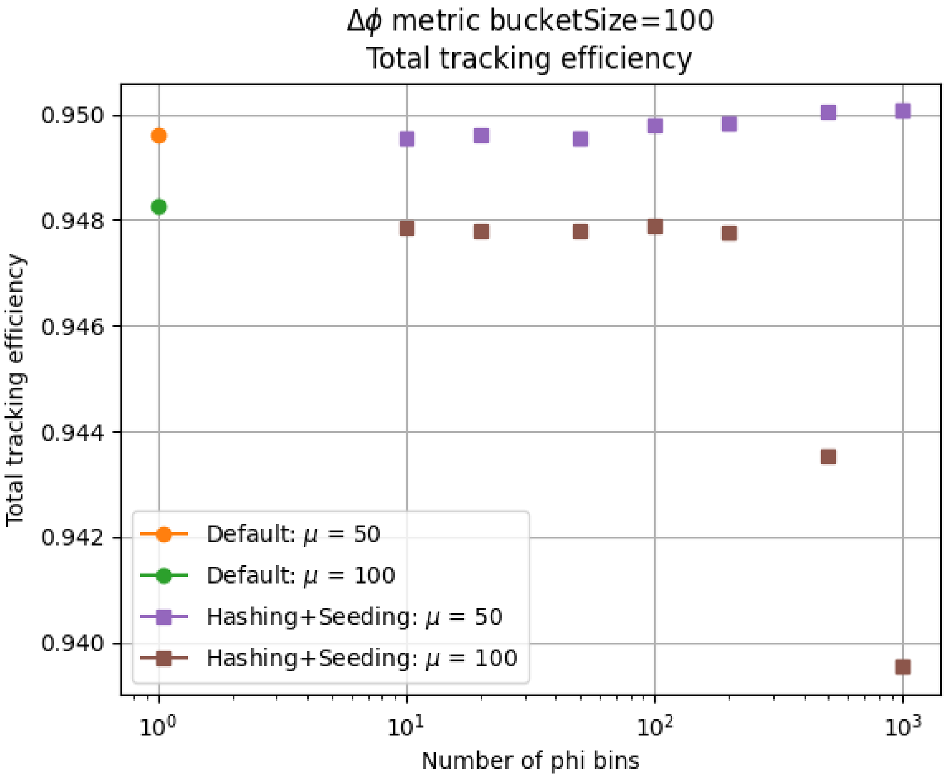


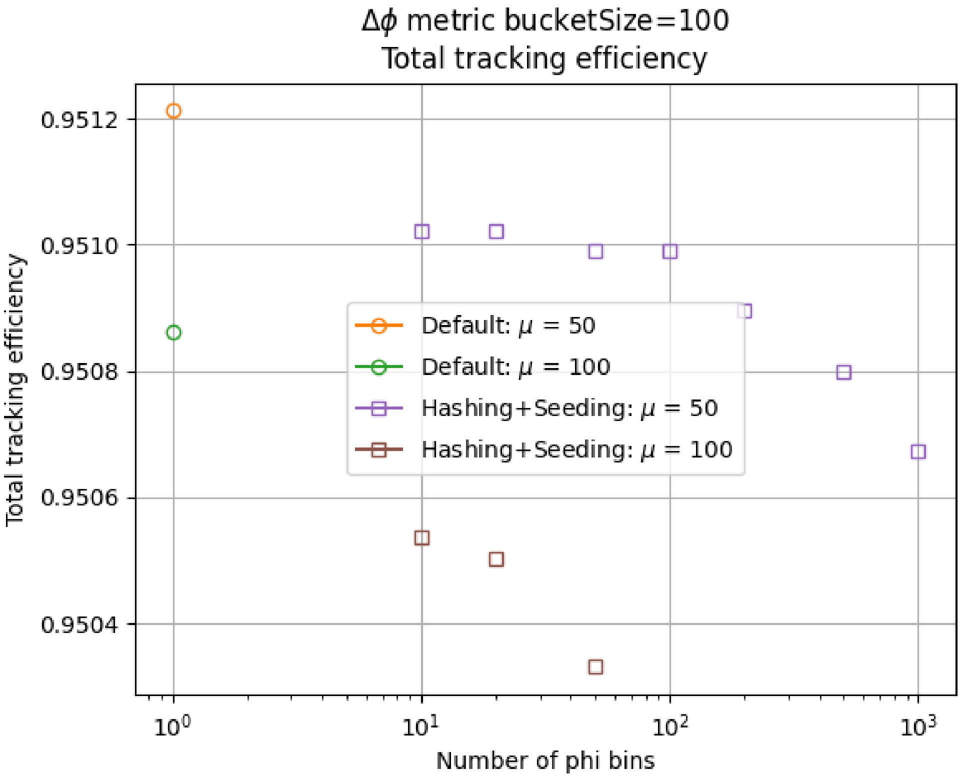
Phi bins: Timing



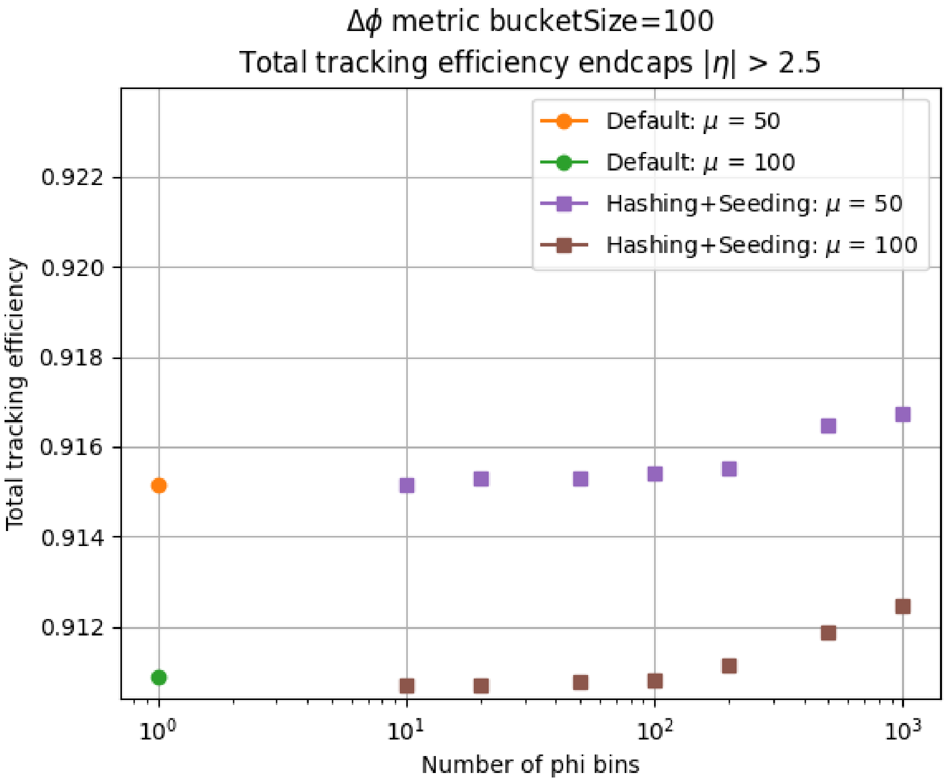


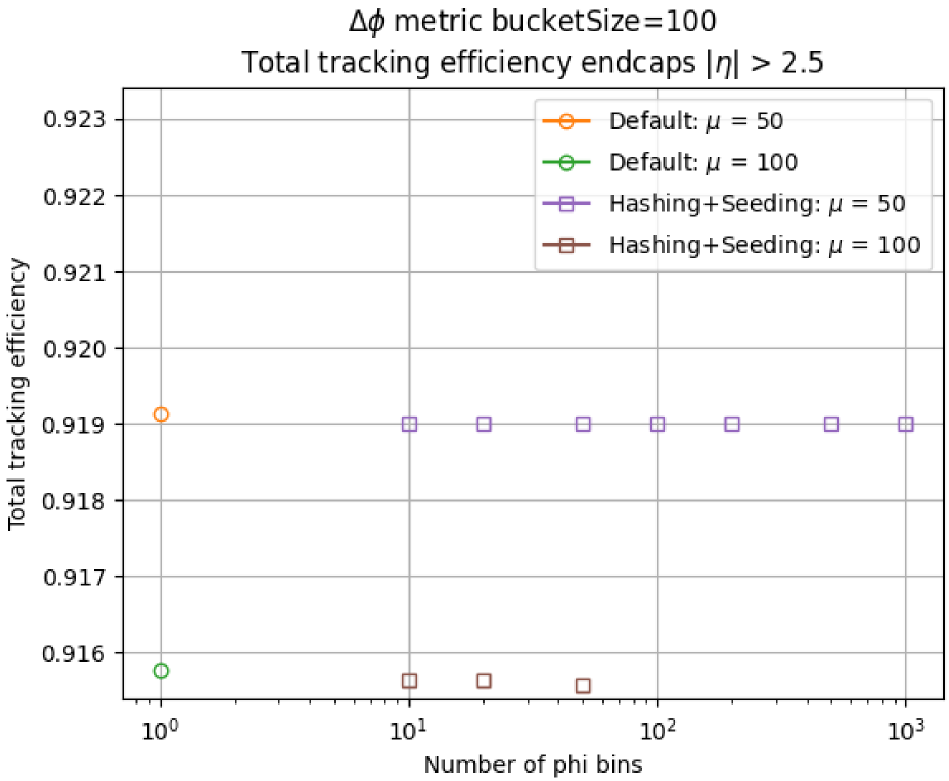
Phi bins: Tracking efficiency



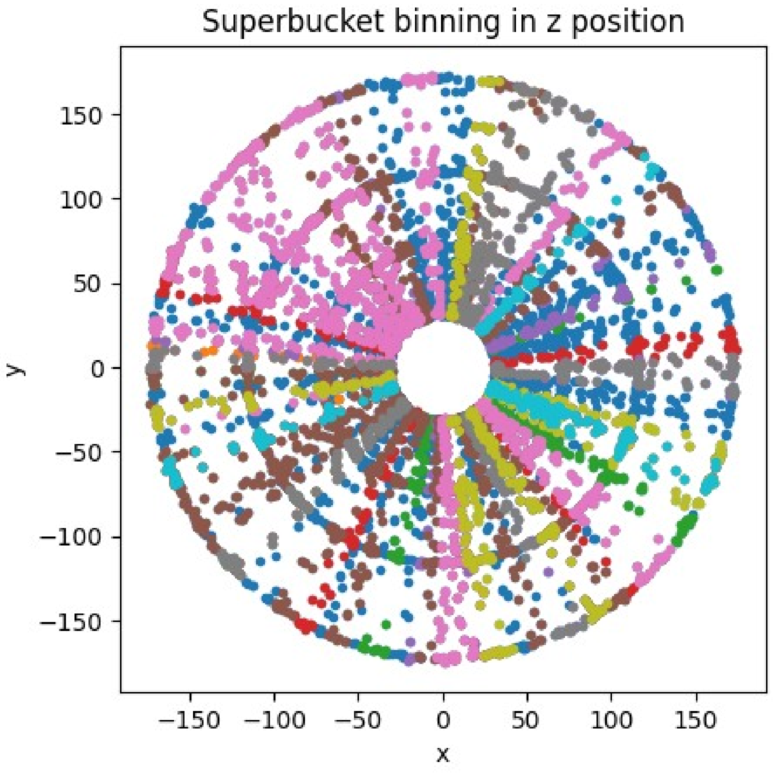


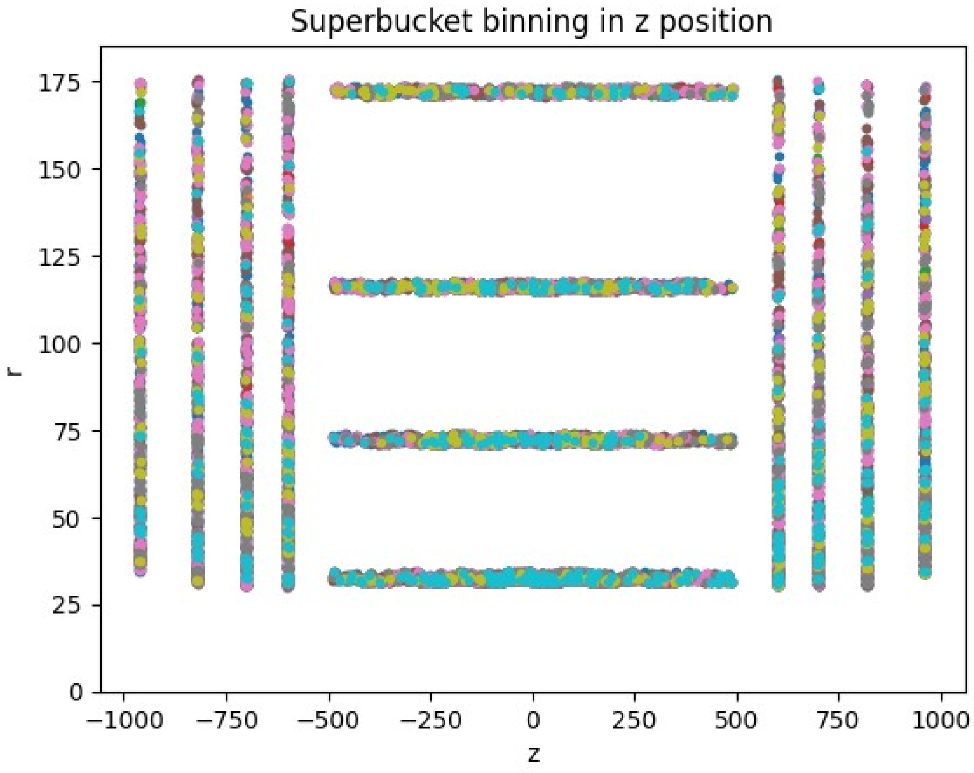
Phi bins: Tracking efficiency



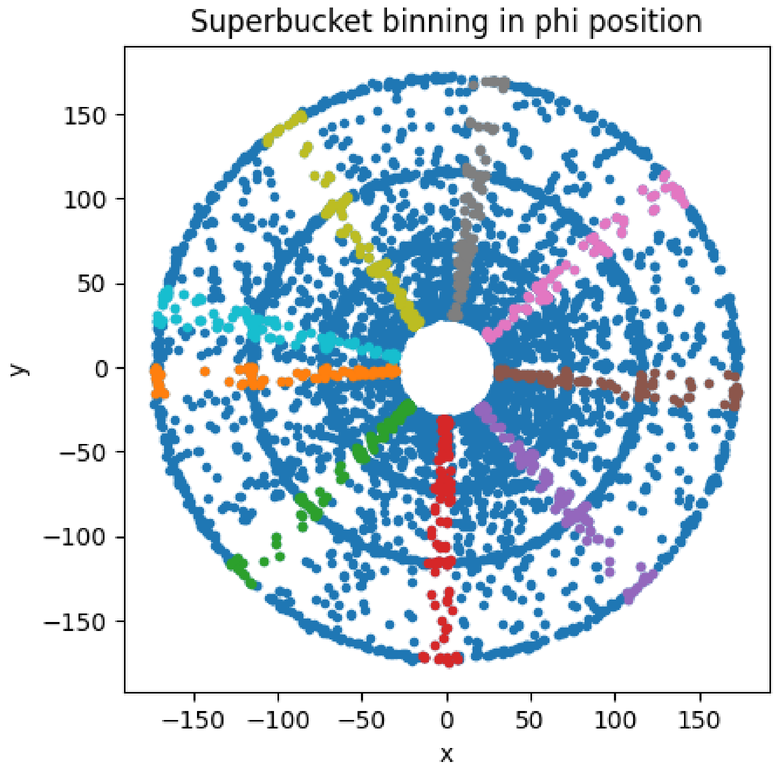


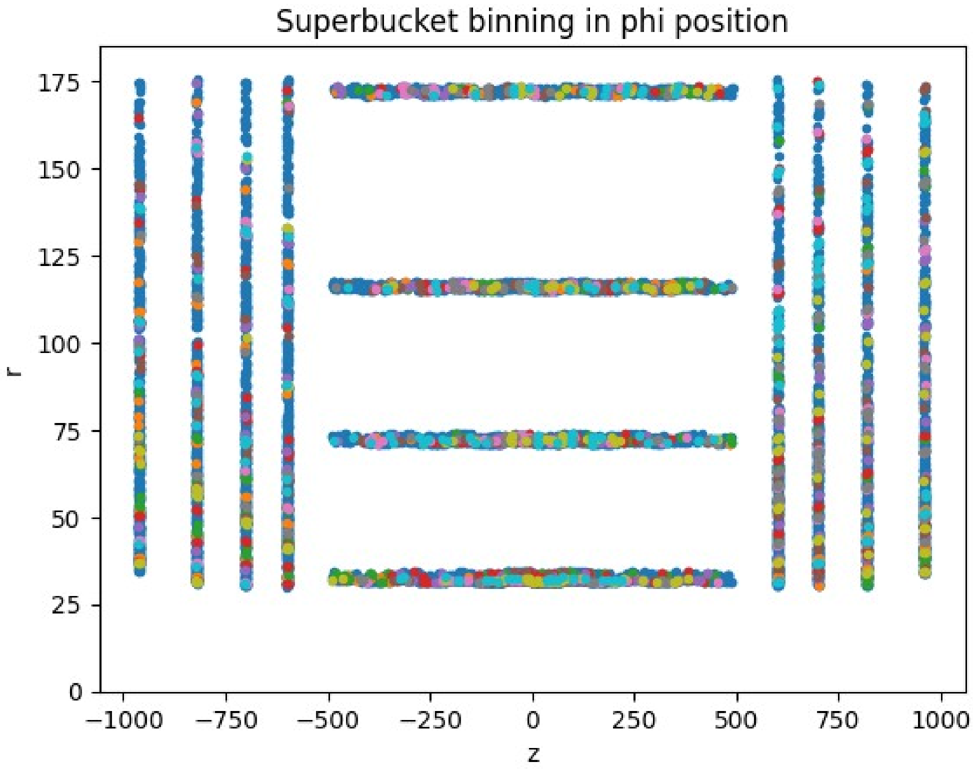
Superbucket binning in Z position



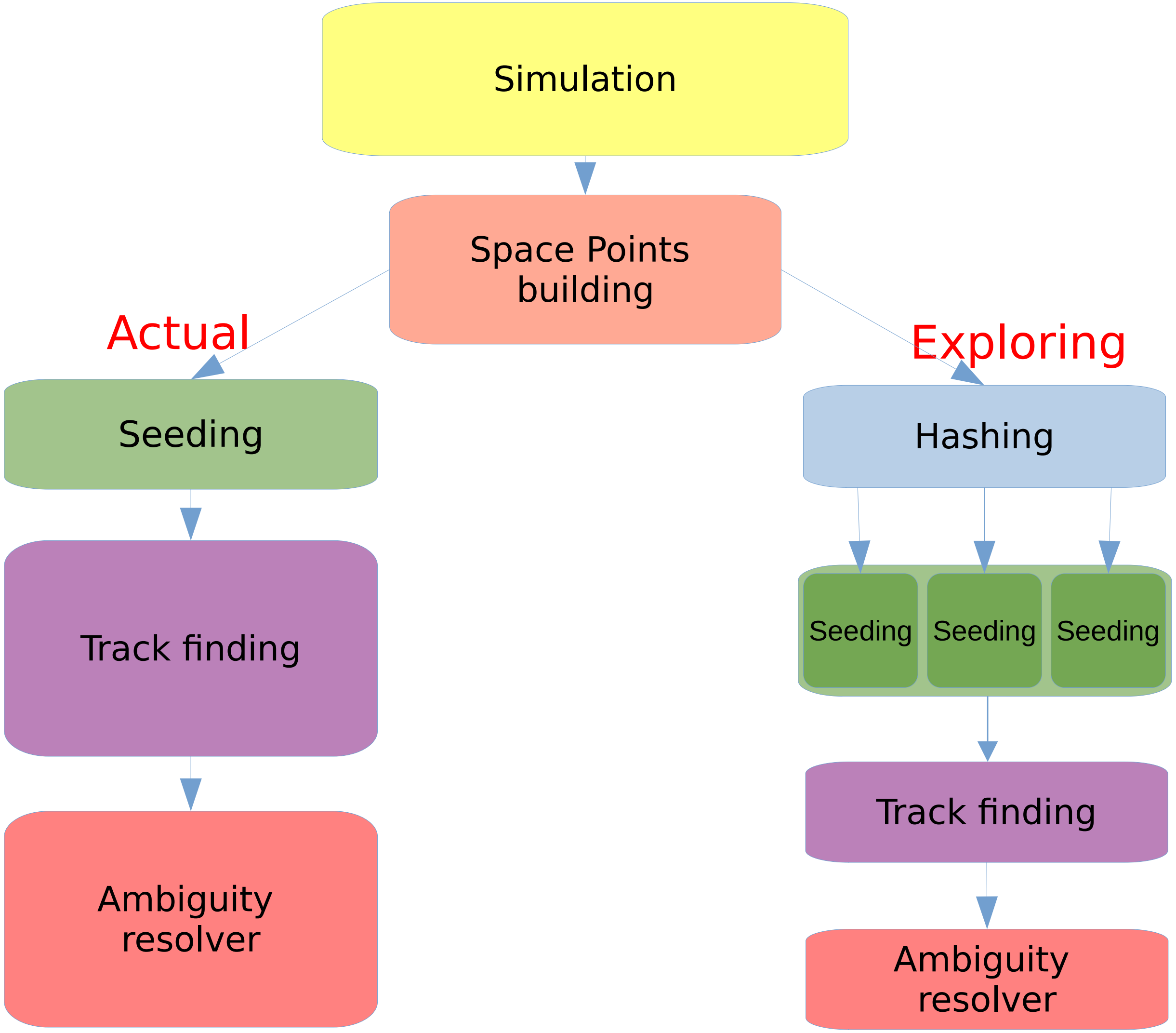


Superbucket binning in Phi position





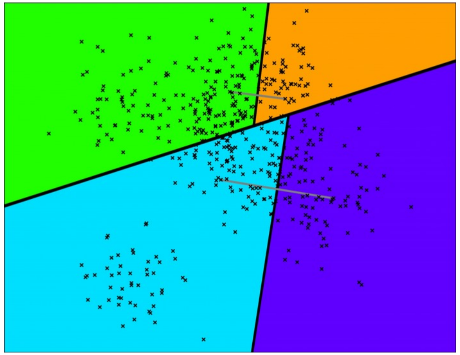
Approaches

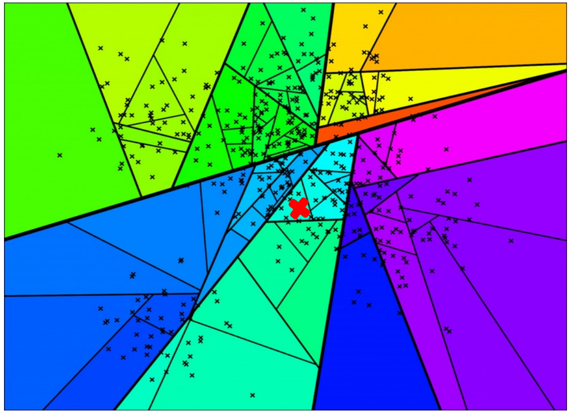


- Seeding parallelization
- Hashing groups space points into buckets
- Hashing reduces the number of space points at a time (focus on relevant space points) → less seeds per bucket

Annoy training

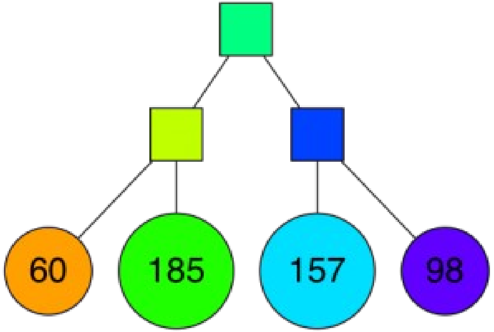
Space separation

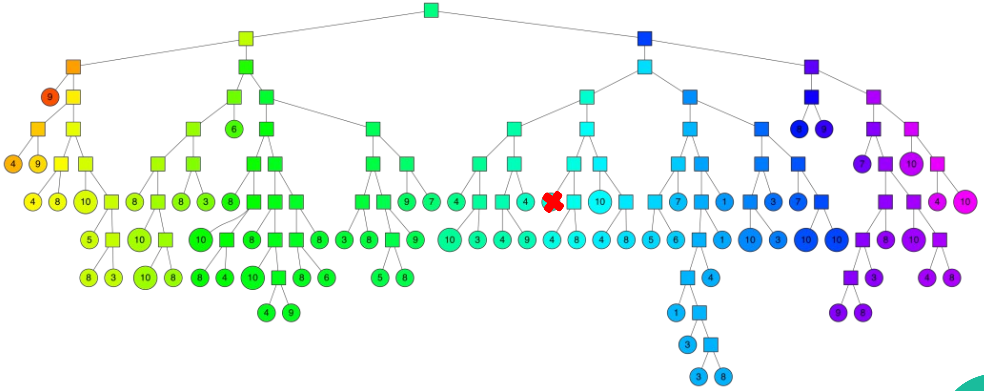




Takes two
random points
iteratively

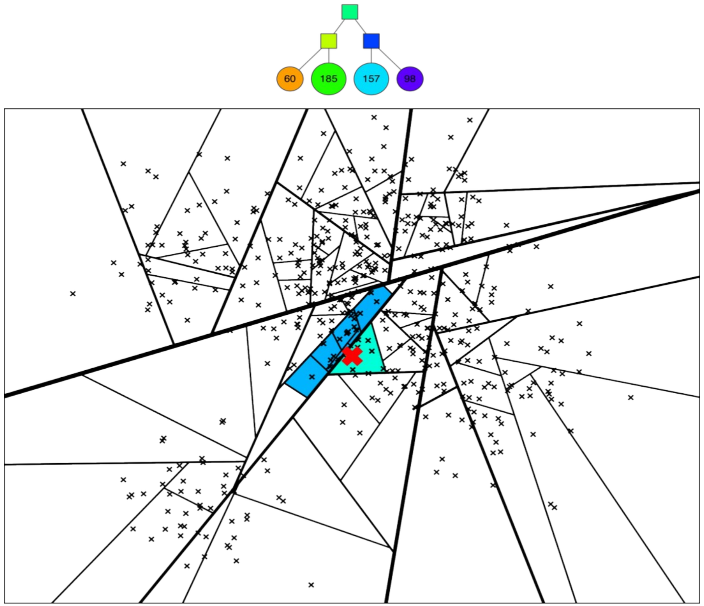
Corresponding binary tree



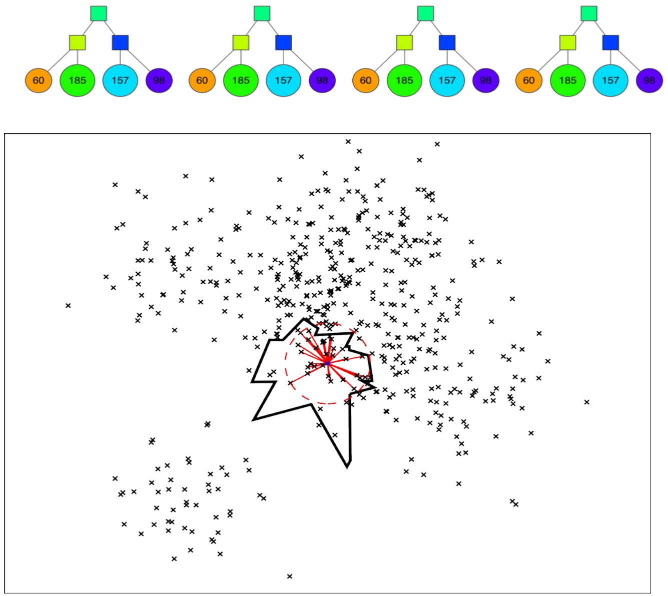


Annoy query

Approximation



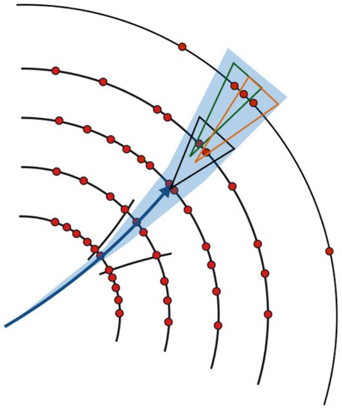
Merge neighbor subspaces

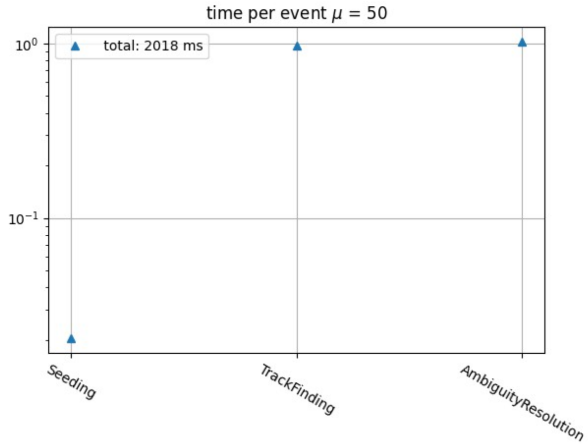


Union of trees’ subspace

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

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