Tracking with Hashing Overview

CSI

20th March 2025

Jeremy Couthures

Outline

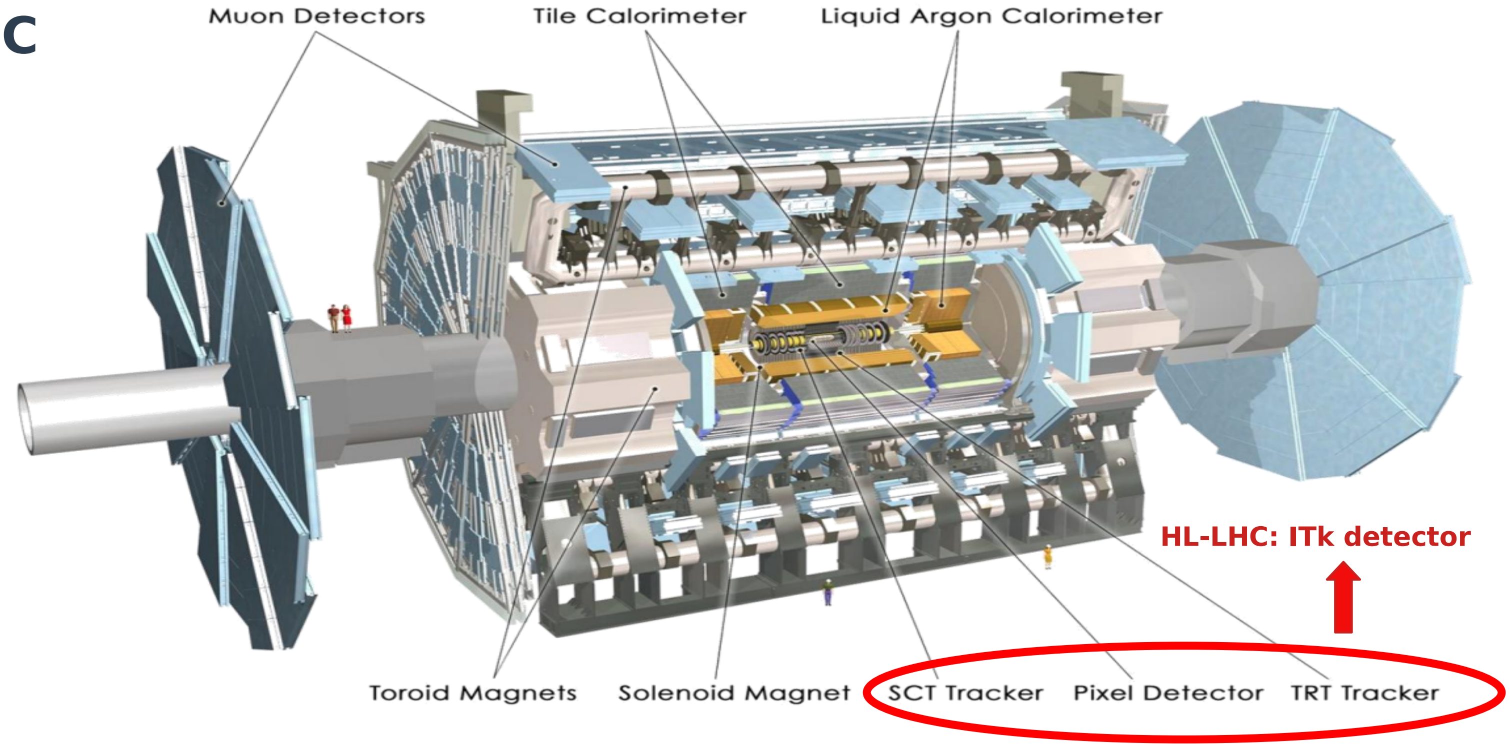
1. **Inner Tracker building**
2. **ATLAS Tracking**
3. **Hashing**
   - ACTS
   - Athena
4. **Interpretability**
5. **Track parameter regression**
6. **Doctoral training**

INNER TRACKER BUILDING

ATLAS detector for HL-LHC

High Luminosity-LHC (HL-LHC):

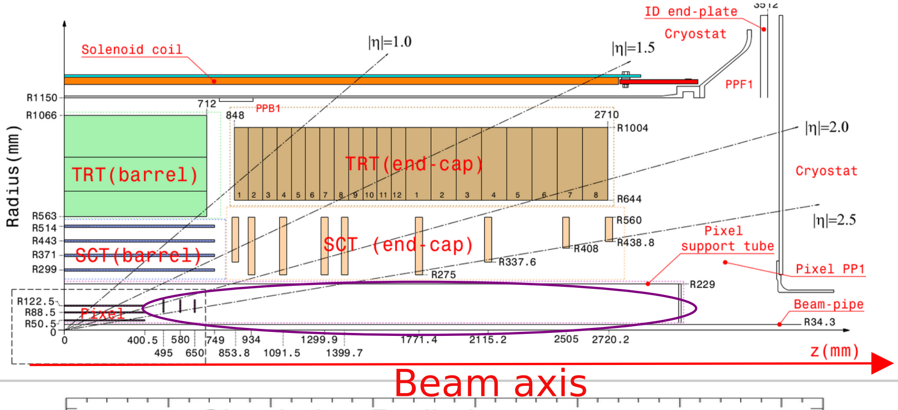
- **Expected in 2029**
- **Increase of luminosity**
  - Luminosity: ~ number of collisions per seconds

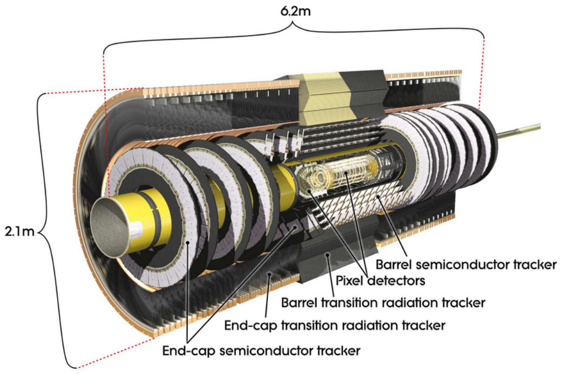


Inner Detector Upgrade

Current Inner detector

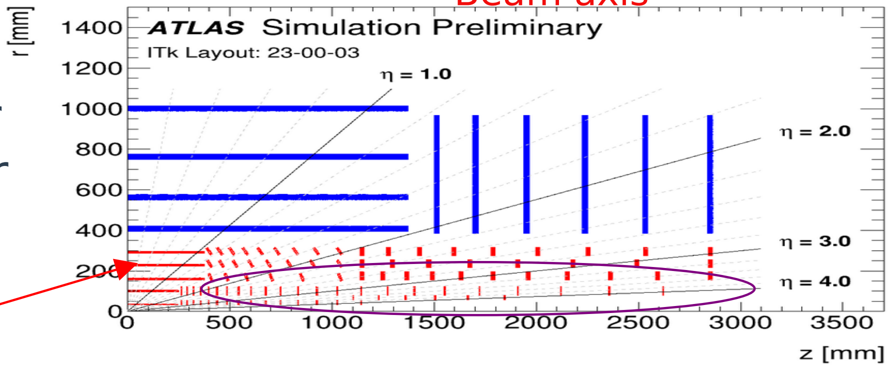
Smallest Pixel size:
50 x 250 µm²

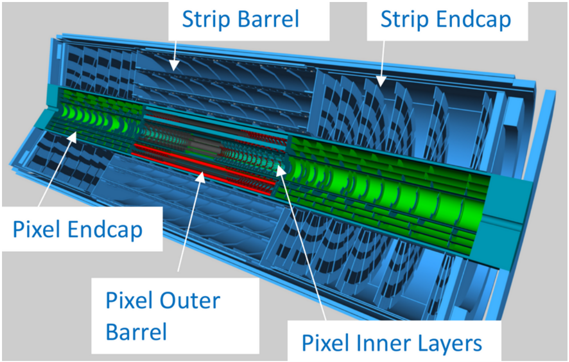




Inner Tracker (ITk) detector

Pixel size:
50 x 50 µm²





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

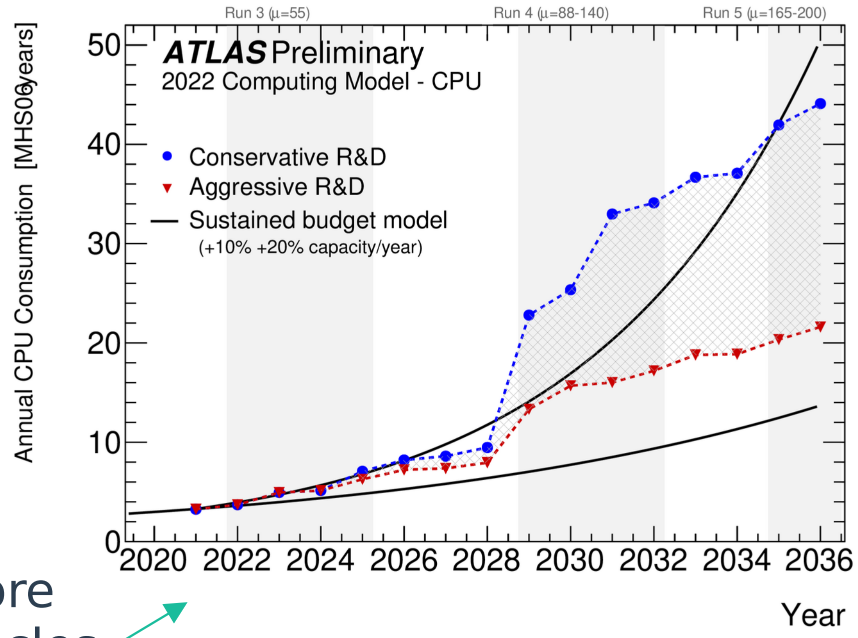
Inner Tracker (ITk) for HL-LHC

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

More particles detected

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

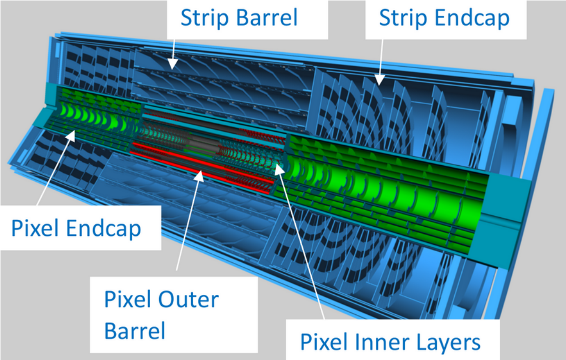
More particles created



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

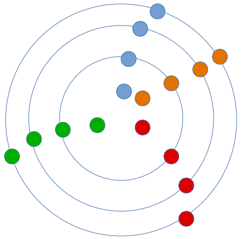
ATLAS TRACKING

ATLAS Tracking simplified

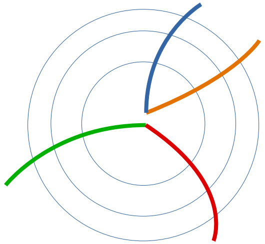


InnerTracker (ITk)

ATLAS Detector at
High Luminosity LHC



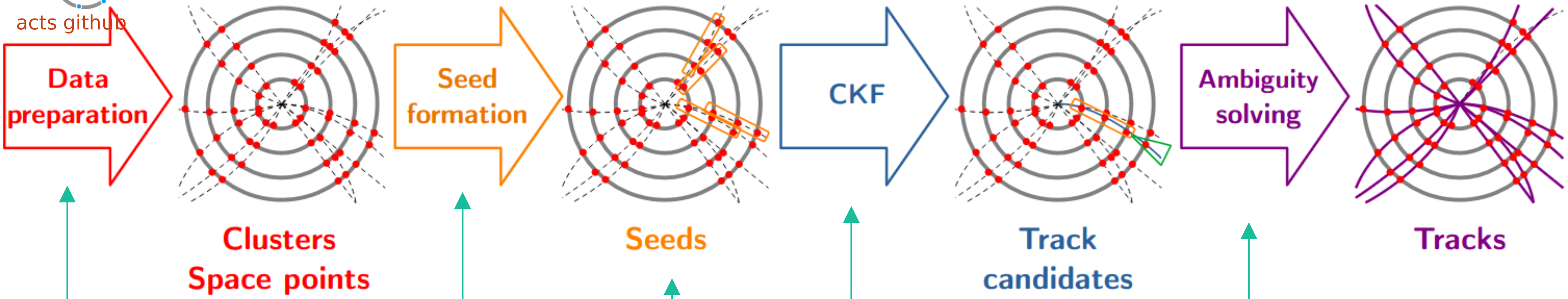
Hits / Space points



Tracks

ATLAS Tracking less simplified

acts github



Transform detector hits into clusters and space points

Group compatible triplets of space points

Seed:
3 space points

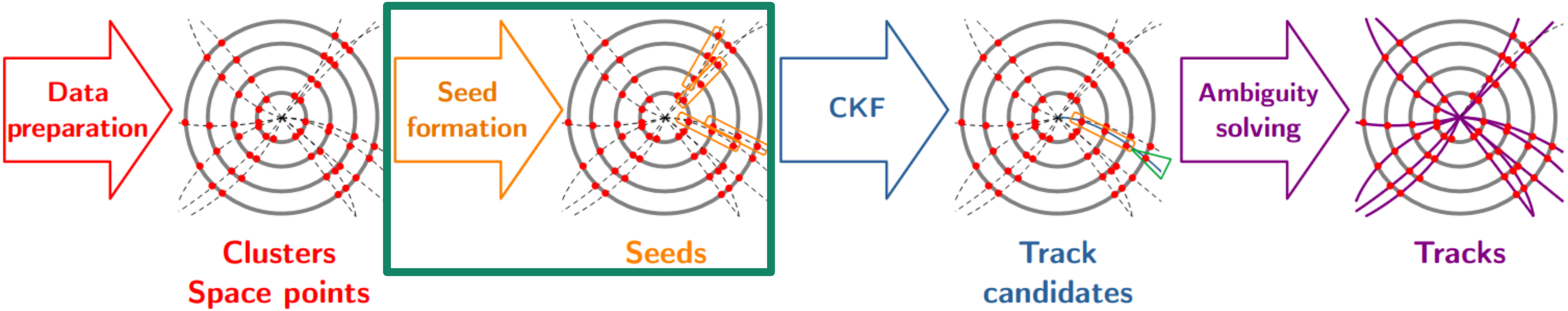
Expand the seeds with remaining space points

Track:
arbitrary number of space points

Remove duplicated tracks and bad ones

Image made by Noemi Calace

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

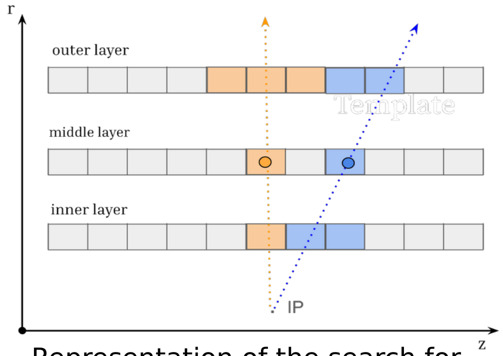
Seeding Algorithm steps

1. Seed Finder

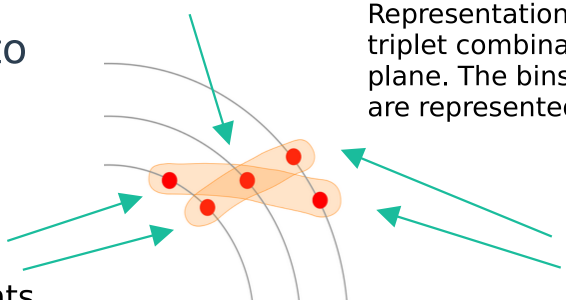
- Check if the triplet forms a nearly straight line in the (r,z) plane

2. Seed Filter

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



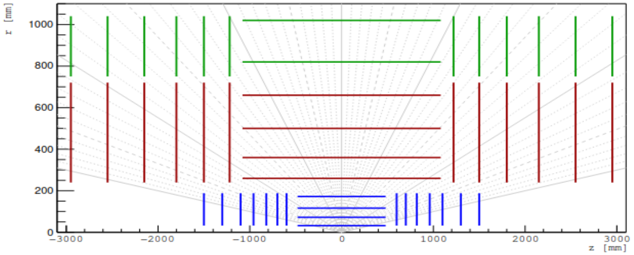
Representation of the search for triplet combinations in the (r, z) plane. The bins used in the search are represented in different colors.



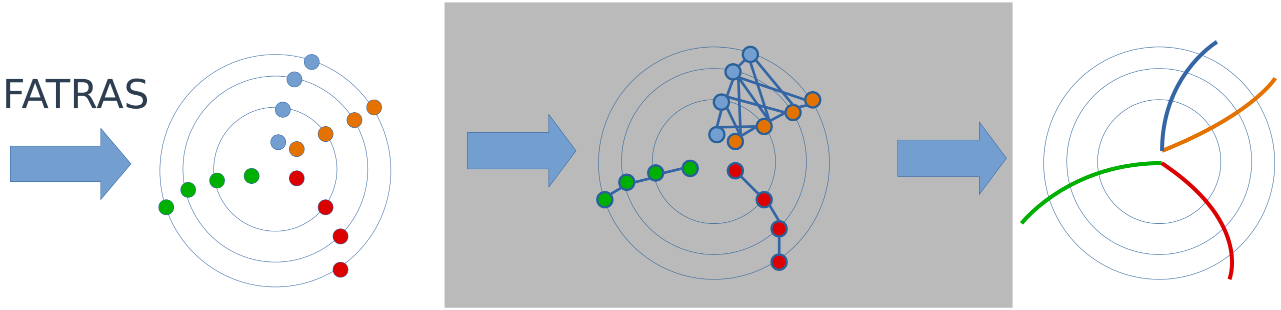
- **Possible improvement:**
  - maxSeedPerSpM: Non physical cut
    → can remove good seeds
- **Can we remove it?**

Initial study

Generic detector
(virtual toy detector)



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



Combinatorics
→ maxSeedsPerSpM=1

Run 4:
<μ> = 140

Pythia8: 100 $t\bar{t}$ events
μ = 50, 100, 150

Not using Geant4:
→ no secondaries

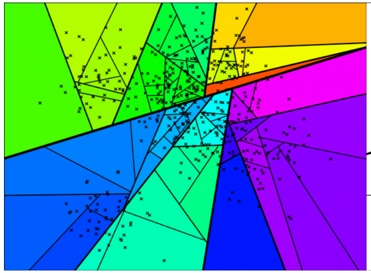
|η| ≤ 4
pT > 1GeV

HASHING IN ACTS

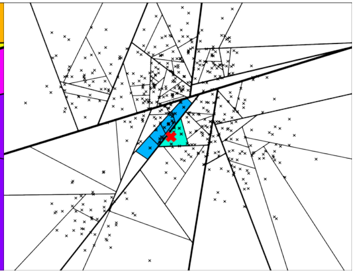
A new method:
Machine Learning/Hashing in the Seeding

Hashing:

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



Space separation



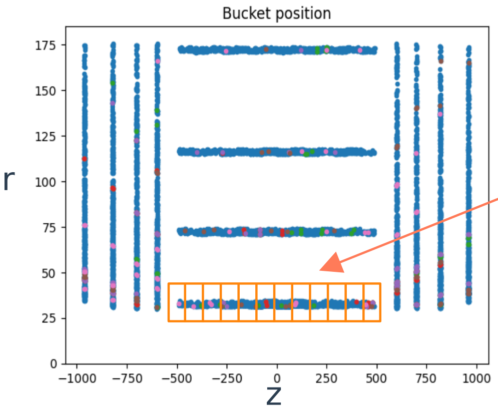
Look for neighbors in the closest regions

Algorithm used:

Approximate Nearest Neighbors Oh Yeah (**Annoy**)

→ Used by Spotify

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

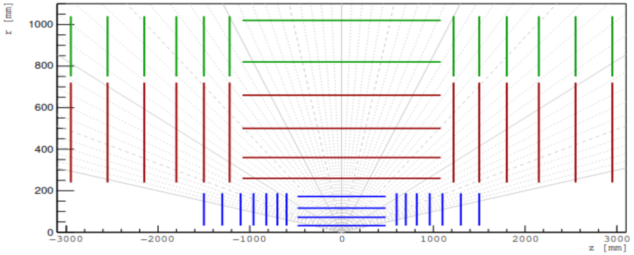


Application:

1) Make bins in layer 0
2) Find Neighbors of the points inside a bin and group them
   1 bin → 1 bucket
3) Do the seeding on the bucket

Overview

Generic detector
(virtual toy detector)

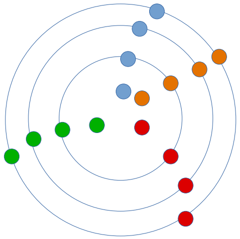


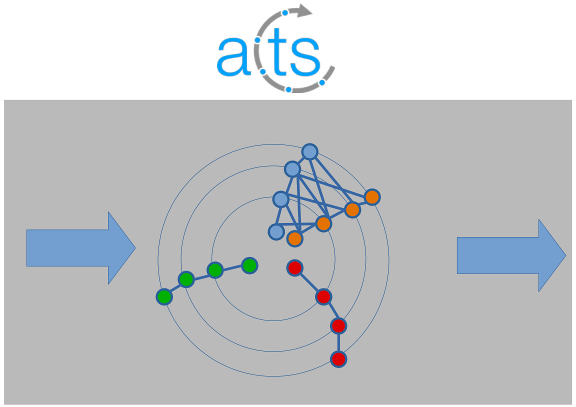
https://arxiv.org/pdf/2105.01160.pdf

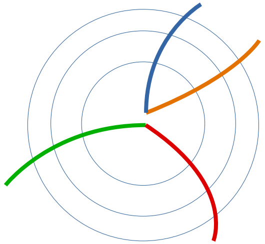
Pythia8: 100 $t\bar{t}$ events
μ = 50, 100, 150

$|\eta| \leq 4$
pT > 1GeV

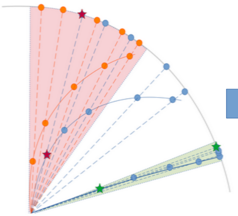
FATRAS



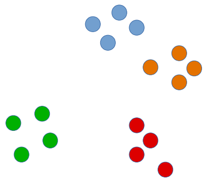




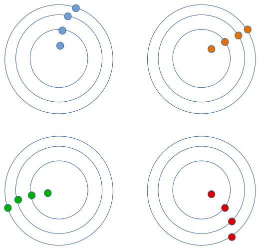
Combinatorics
→ maxSeedsPerSpM=1



Metric: $\Delta\varphi$
Suitable for high pT tracks



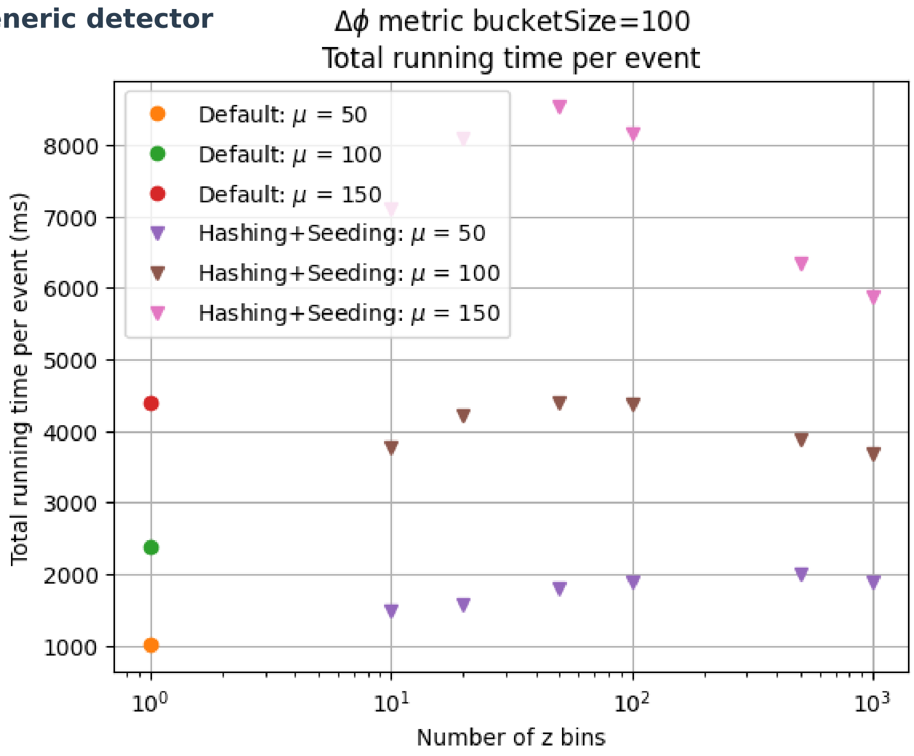
Clustering:
Annoy



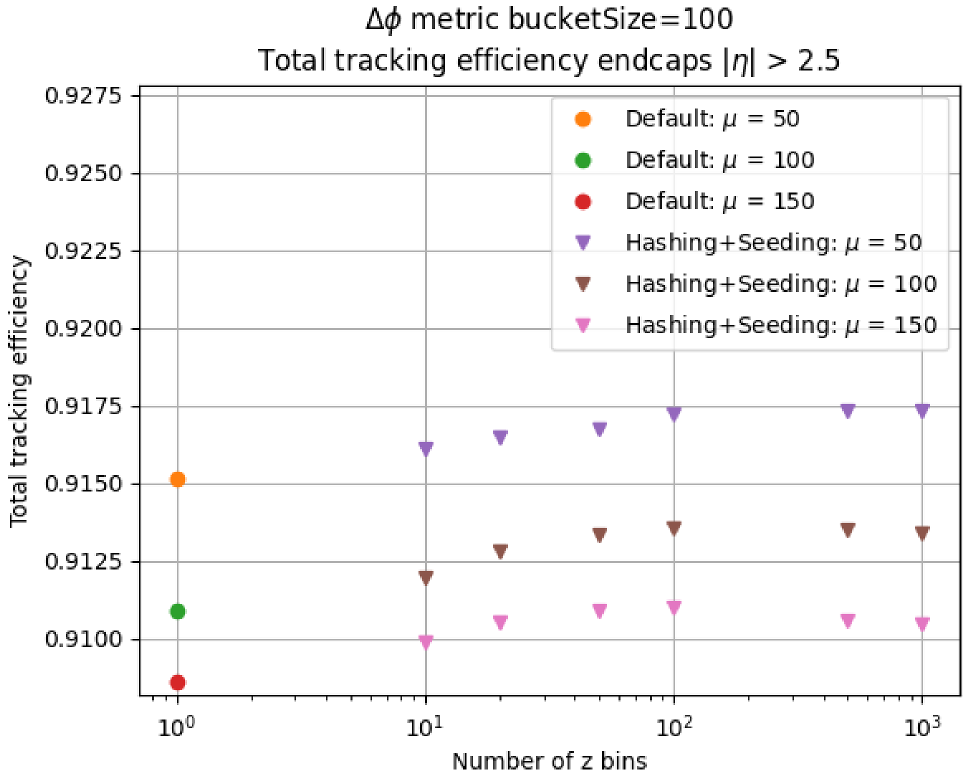
Parallelization

Timing and endcaps efficiency

Generic detector



Running time ~ x2

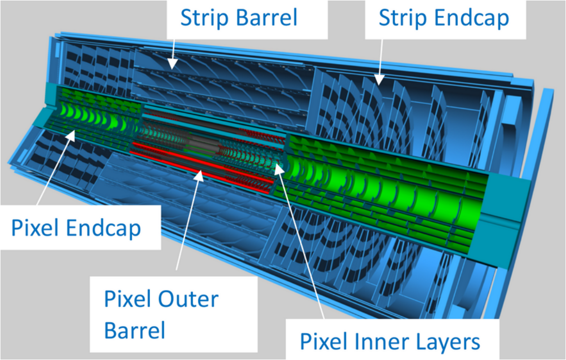


Always improve

HASHING IN ATHENA

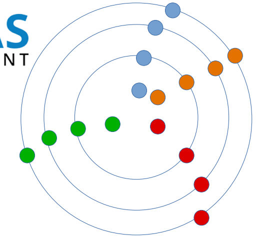
Realistic case

Inner Tracker (ITk)
(beeing built)

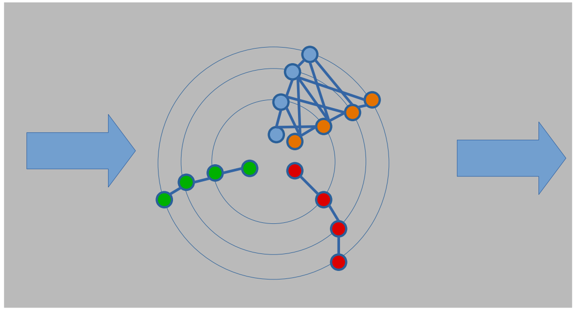


HL-LHC:
$\langle\mu\rangle = 200$

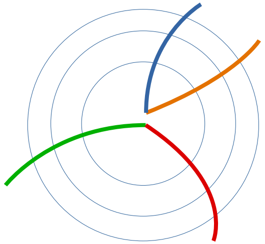
Official simulations
1000 $t\bar{t}$ events
μ = 0, 60, 140, 200



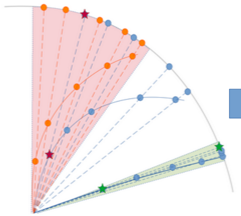
ATLAS EXPERIMENT + acts



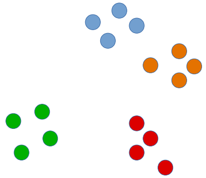
Combinatorics
→ maxSeedsPerSpM=4



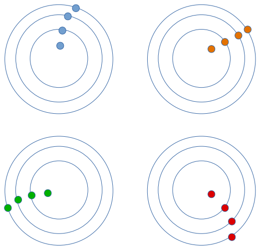
acts + ATLAS EXPERIMENT



Metric: $\Delta\varphi$
Suitable for high pT tracks



Clustering:
Annoy

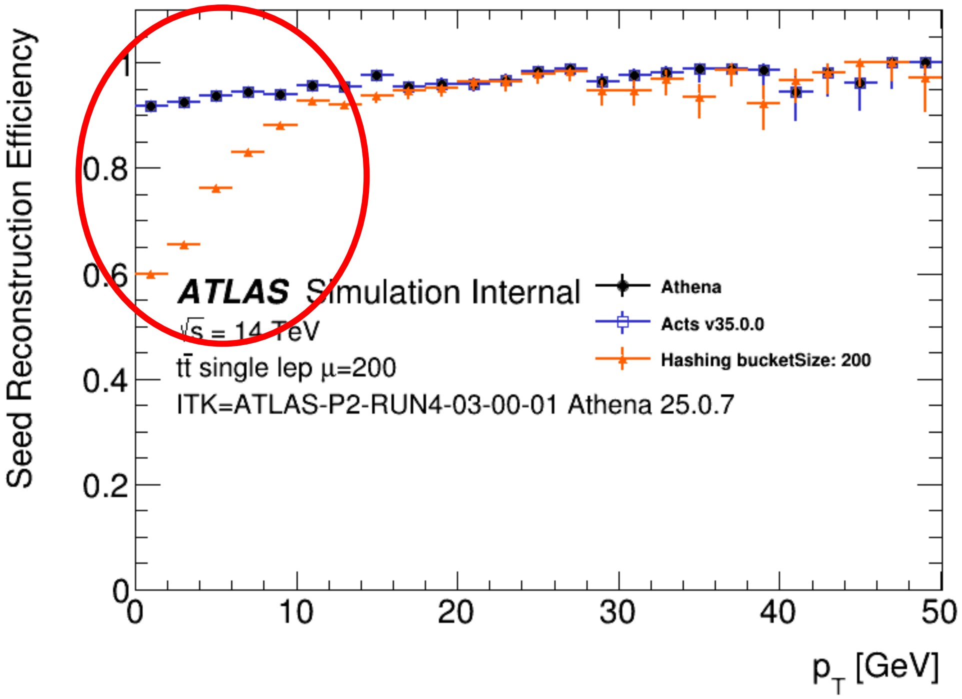


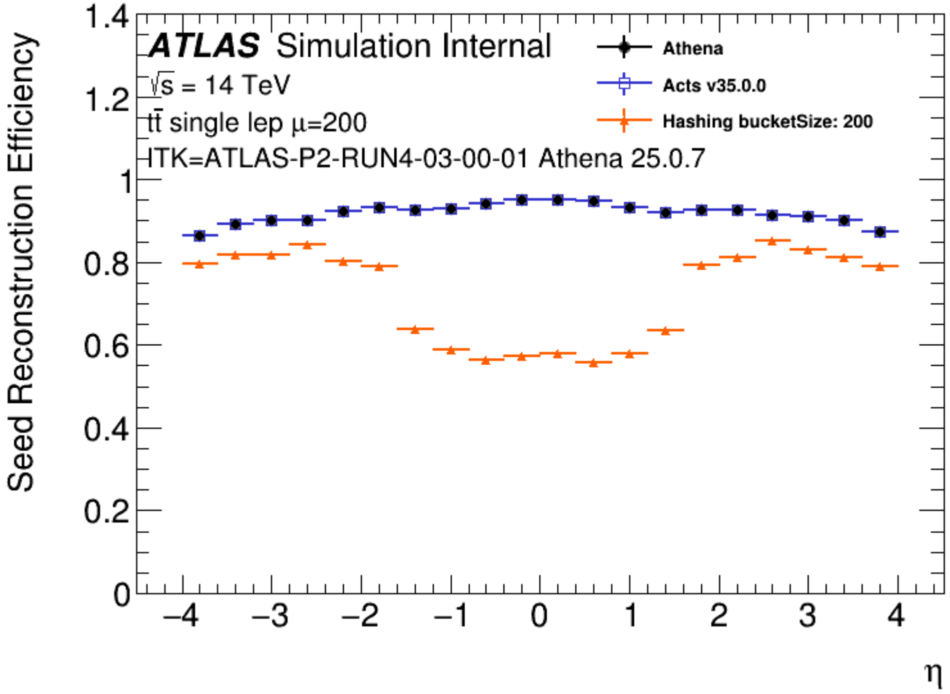
Parallelization

Δφ: Seed Efficiency μ=200

InnerTracker

WARNING: not only first layer selected

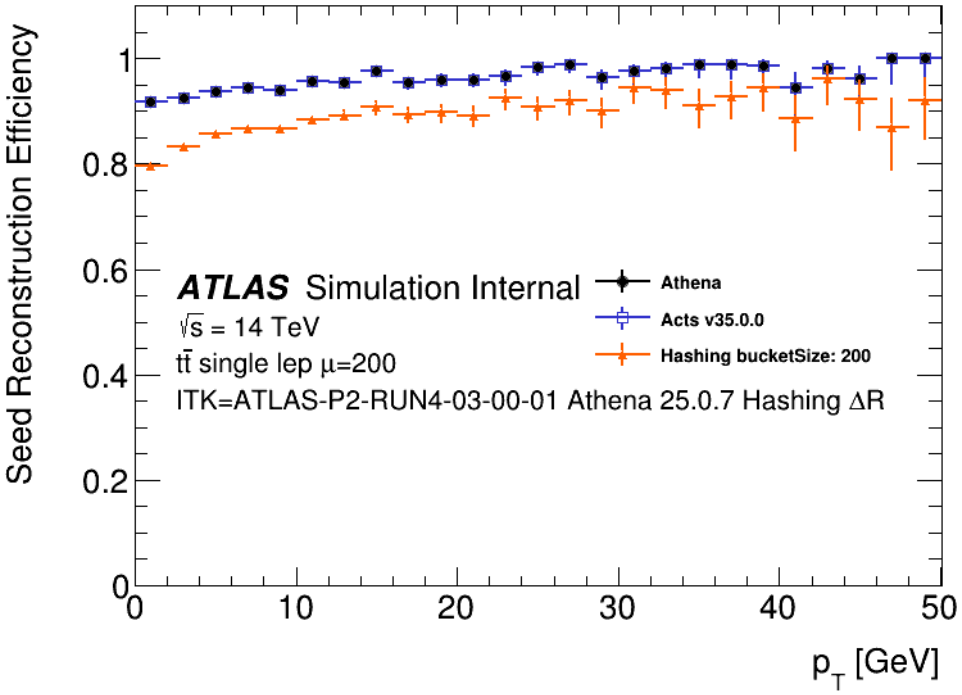


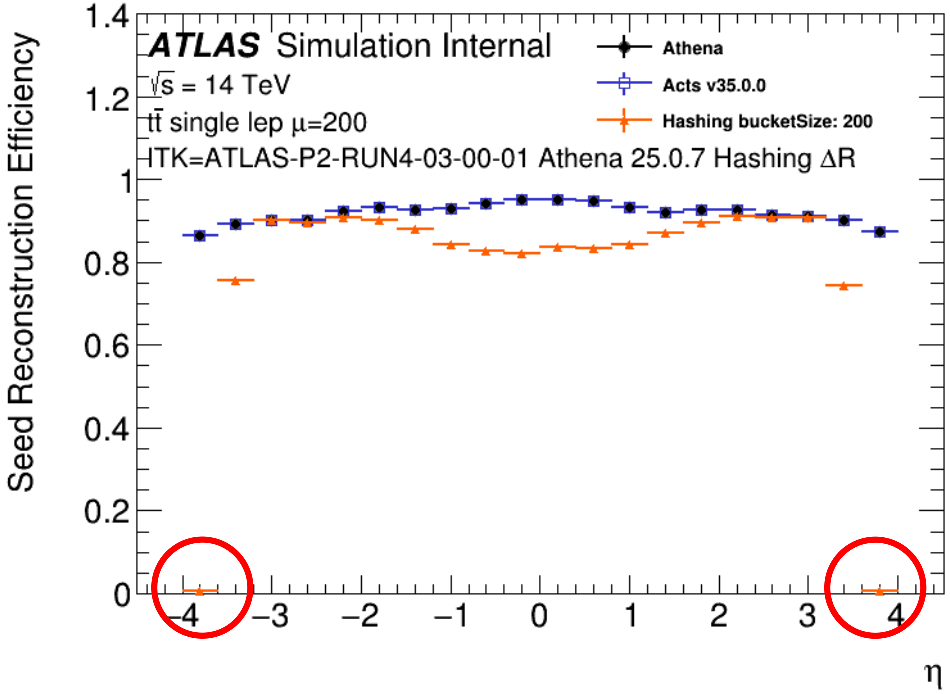


ΔR: Seed Efficiency μ=200

InnerTracker

WARNING: not only first layer selected





Hashing study summary

- **Current state:**
  - Efficiency depends on the region and the metric
  - Timing does not match standard algorithms

Next:

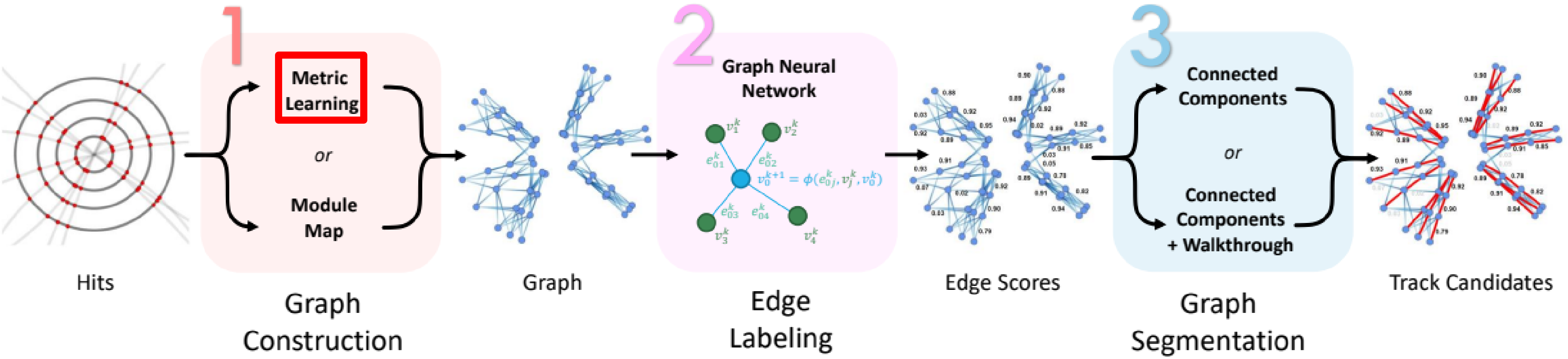
  - Metric learning could help work on different regions

METRIC LEARNING

Acorn

acorn

gitlab



Schematic overview of the GNN-based track finding pipeline

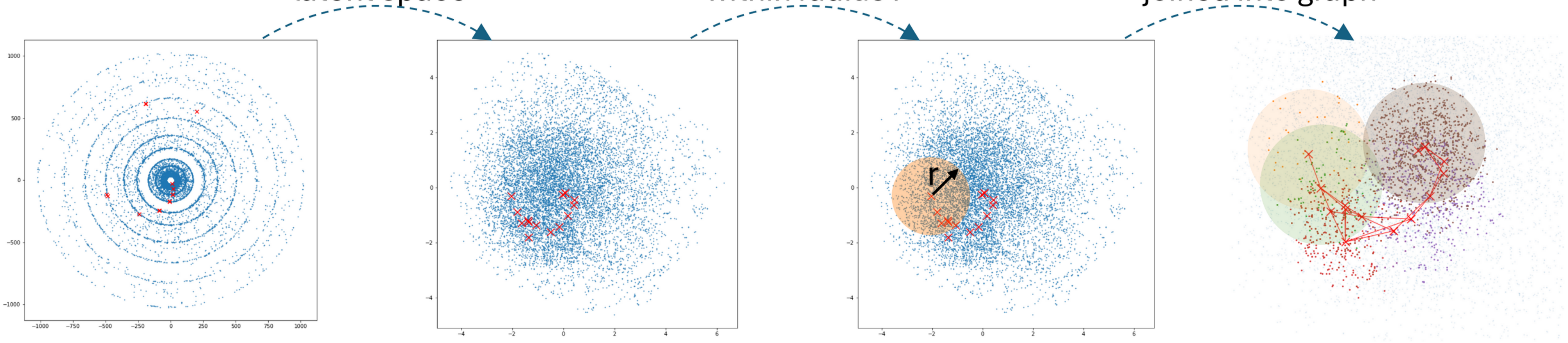
https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/IDTR-2022-01/

GNN Metric Learning

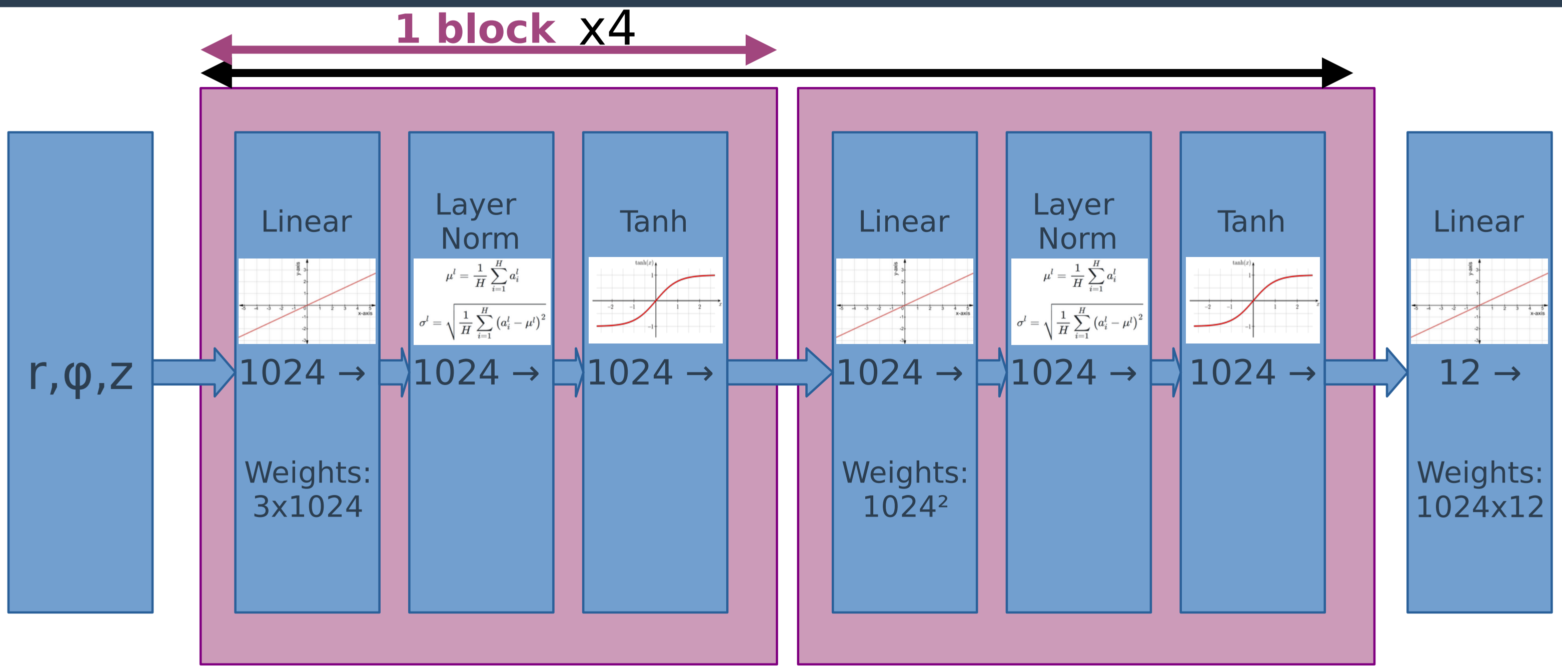
Embed into learned latent space

Connect all space points within radius r

All space point pairs joined into graph

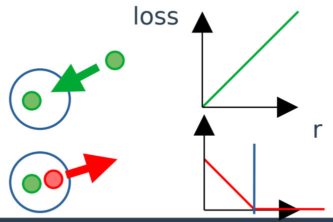


Architecture

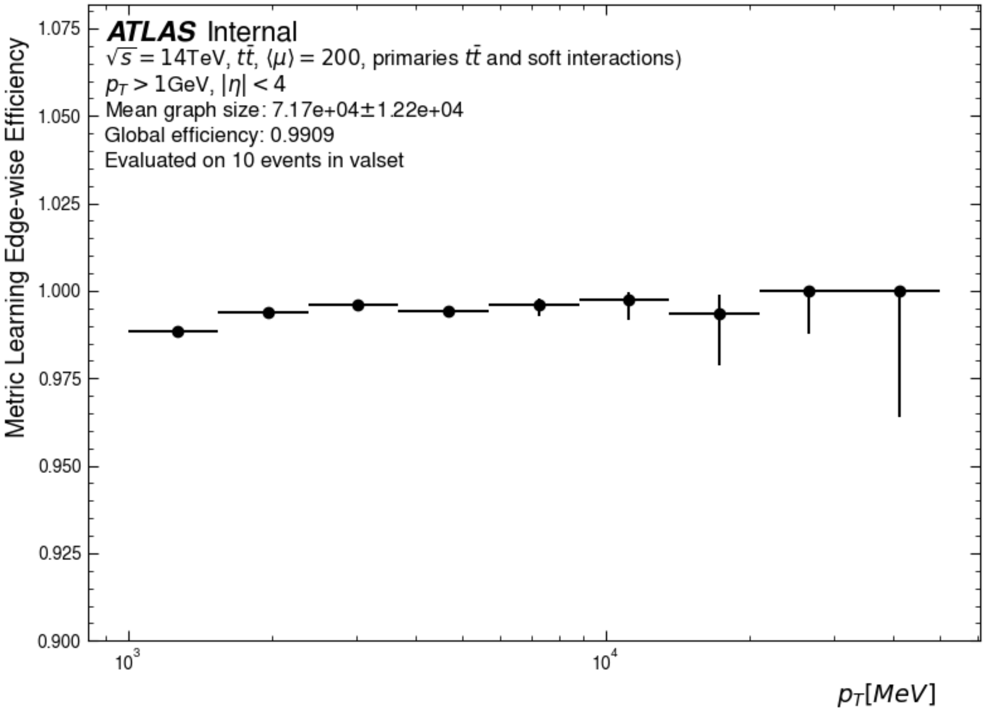


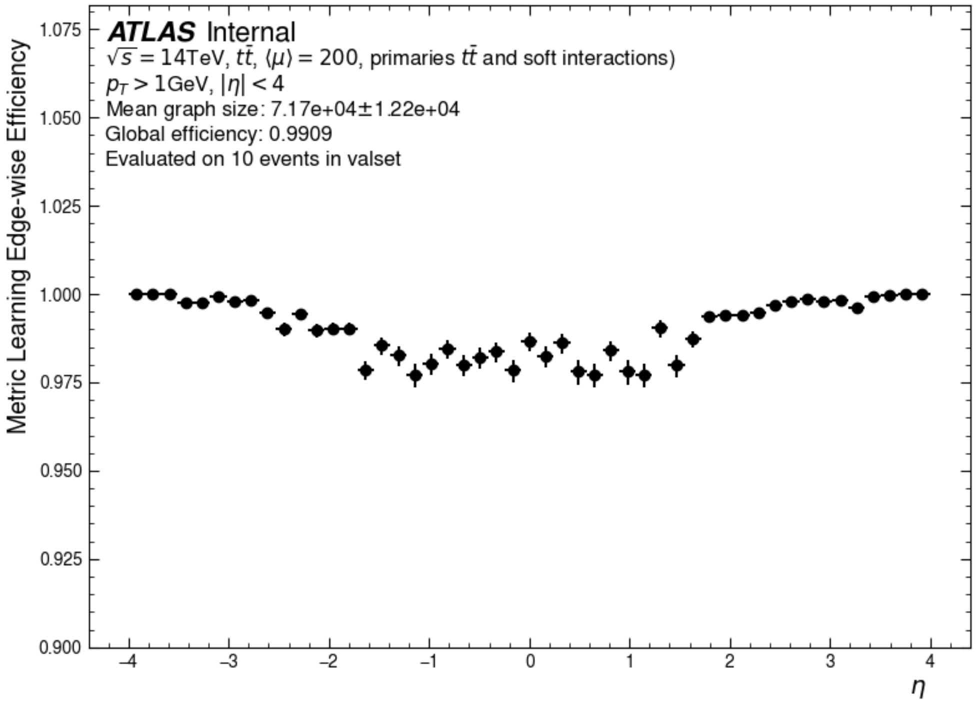
Hinge Loss

$$l_n = \begin{cases} x_n, & \text{if } y_n = 1, \\ \max\{0, margin - x_n\}, & \text{if } y_n = -1, \end{cases}$$



Performance

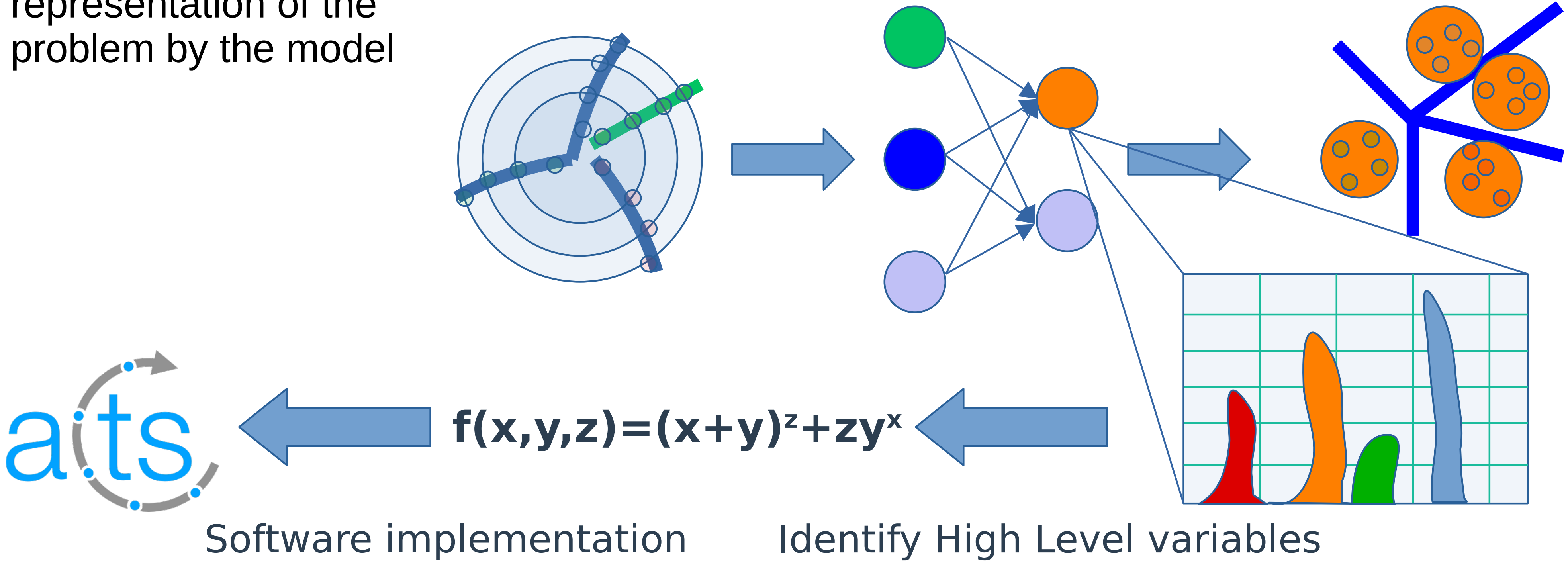




INTERPRETABILITY

Interpretability

- study the internal representation of the problem by the model



Interpretability: The *How*, but of *What* ?

- **Goal: Understand the model with physics**
  - Ideal: from black box (ML) to algorithm (physics)
- ***How* is the prediction done?:**
  - What are the steps taken?
- **Need to understand *What* it predicts:**
  - Objective (loss function): group **consecutive** hits of **same particle**
    - But not necessarily what is done (poorly trained / untrained vs trained)
  - Performance plots: How good are the predictions with respect to the objective
  - Constraints: Hit by hit application → no curvature (q, pT) information

Interpretability approaches

- **Extracting information:**
  - Assume the model is building an **algorithm** internally: ***mechanistic interpretability***
- **Approaches:**
  - identify parts of this algorithm (relevant pieces)
  - identify known high-level features built internally

Identifying parts of the algorithm

- **Approach:**
  - Interpret relevant neurons as formulas
- **Steps:**
  1) Identify relevant neurons
  2) Symbolic regression to obtain a formula of the quantity approximated
  3) Identify relevant parts of the equation
  4) Compare with known physics high-level variables

Permutation importance

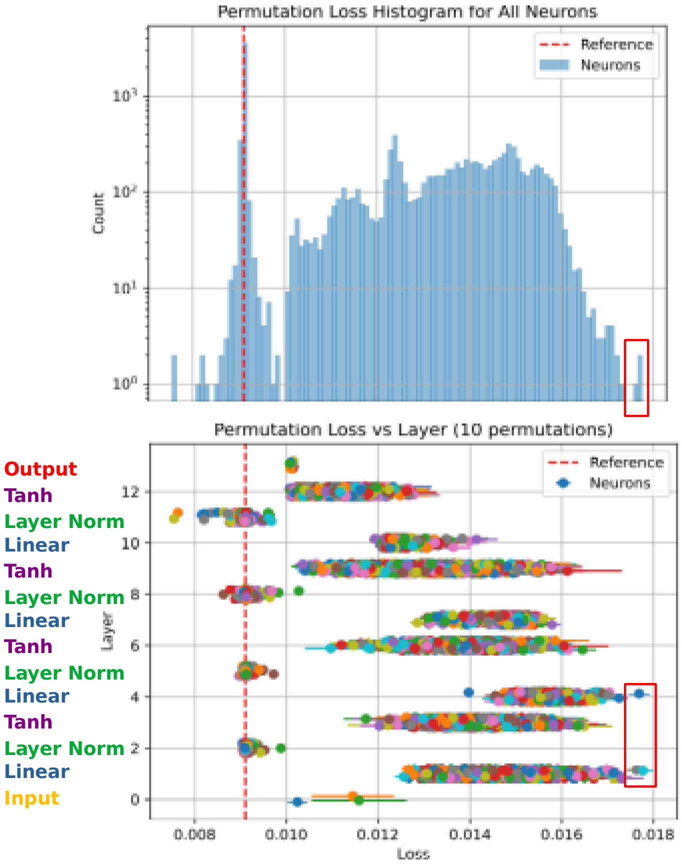
- **Idea:**
  - Evaluate performance on the samples
  - Swap values of a target variable between samples in the dataset
  - Evaluate performance on the samples with the target variable shuffled (and only this one)
  - Compare performance with and without shuffling
  - Variables with the most drop in performance are the most important
- **Objective dependent metric**

Dataset

- **Used training set to avoid out-of-distribution issues**
  - Not about how well it reconstructs a high-level variable in general (= test set) but which one it tries to reconstruct (= training set; might overfit)
  - Performance plots show good global generalization

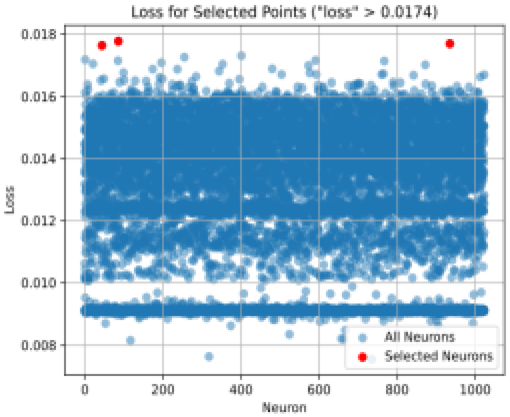
Neuron identification: Permutation loss

- **3 promising neurons:**
  - 2 on layer 1 (***Linear*** with input layer)
  - 1 on layer 4 (More complex)
- **Normalization Layers not perturbed by permutation → Information is shared among neurons**



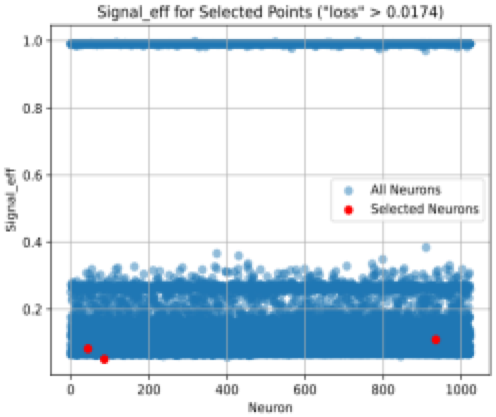
Neuron specificities: Permutation metrics

Mean value of the metrics after permutations:

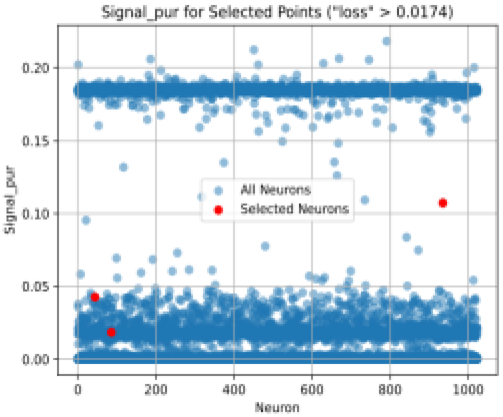


Loss: lower is better

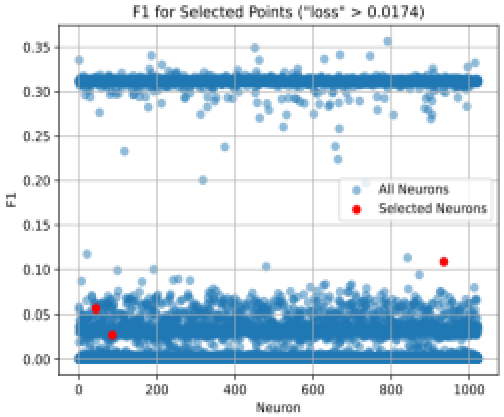
Signal eff: higher is better



Signal purity: higher is better



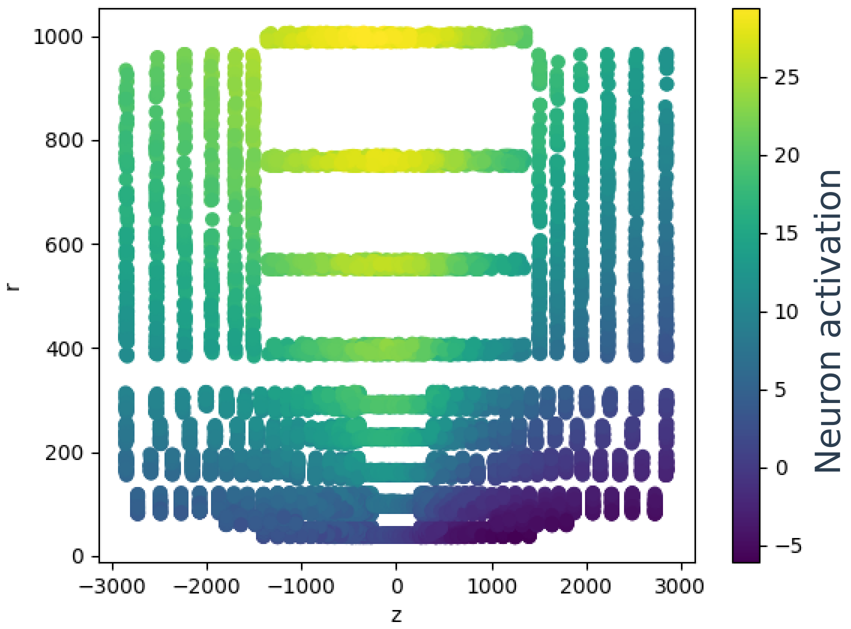
F1 score: higher is better



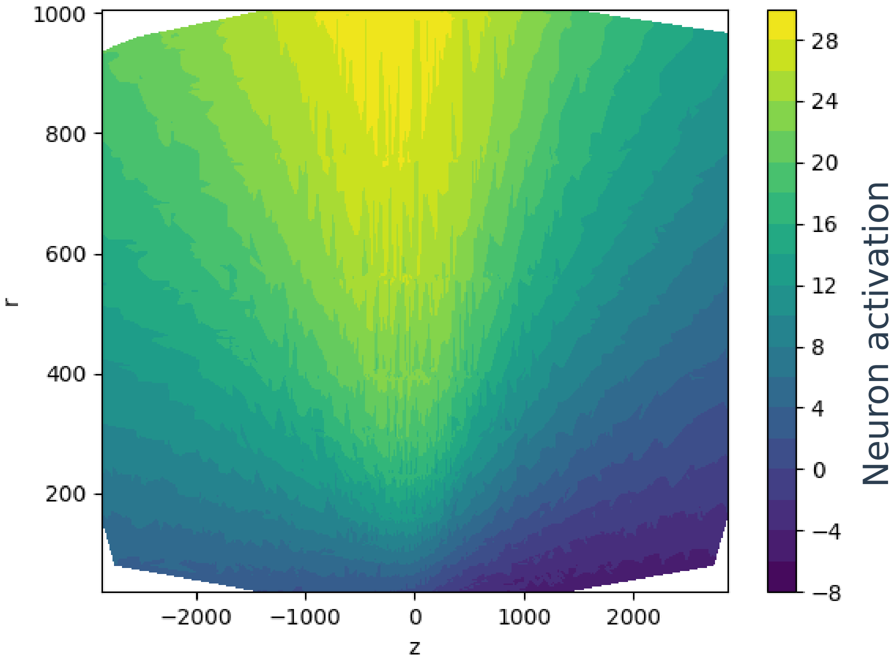
Model performance heavily rely on those neurons

Non-Linear Layer: Activations r-z neuron 935

Real



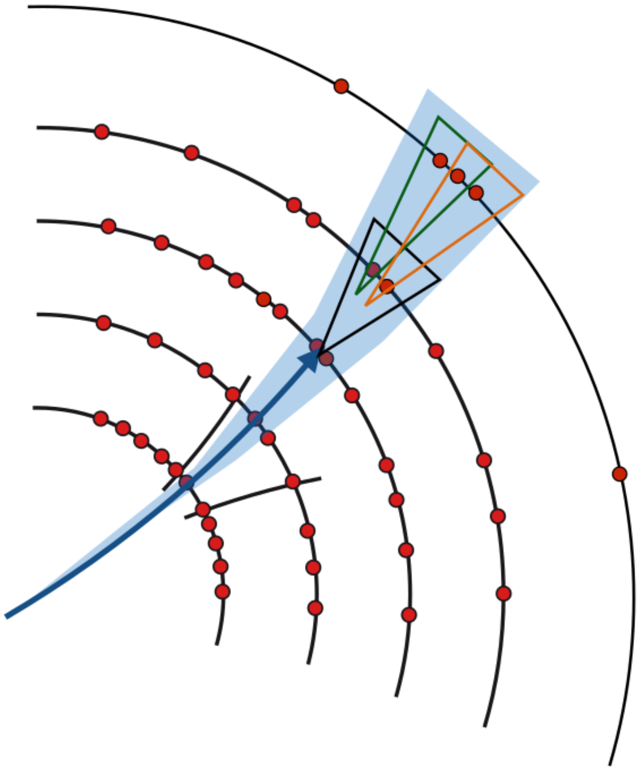
Isocurves



Combinatorial problem

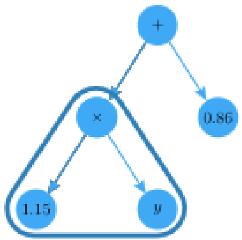
Combinatorial Kalman Filter:

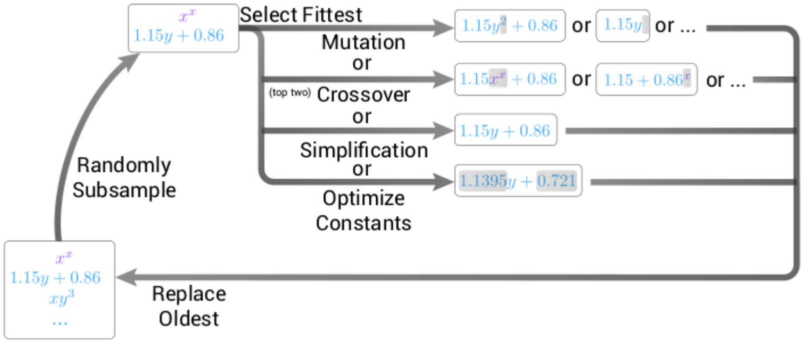
- Several possibilities of expanding the seeds at each layer → need to test them all
- Number of combinations increases exponentially with the number of layers



Symbolic regression

$1.15y + 0.86$





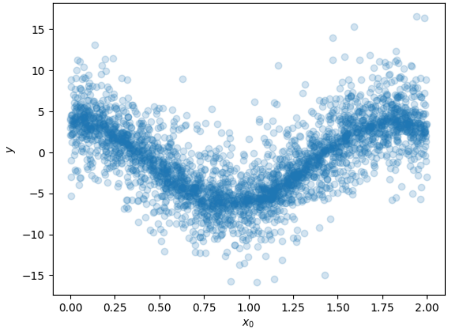
Genetic Algorithm

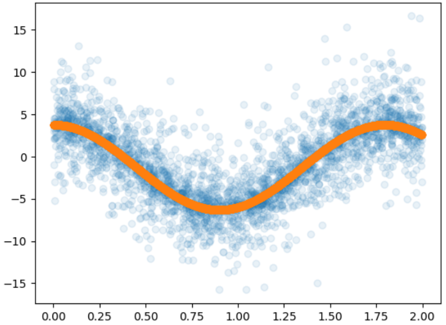
$$\sigma \sim U(0.1, 5.0)$$
$$\epsilon \sim N(0, \sigma^2)$$
$$y = 5\ \cos(3.5x_0) - 1.3 + \epsilon.$$

Truth

$$5.0337477 \cos\left(3.496164 x_0\right) - 1.29099218487498$$

Learned

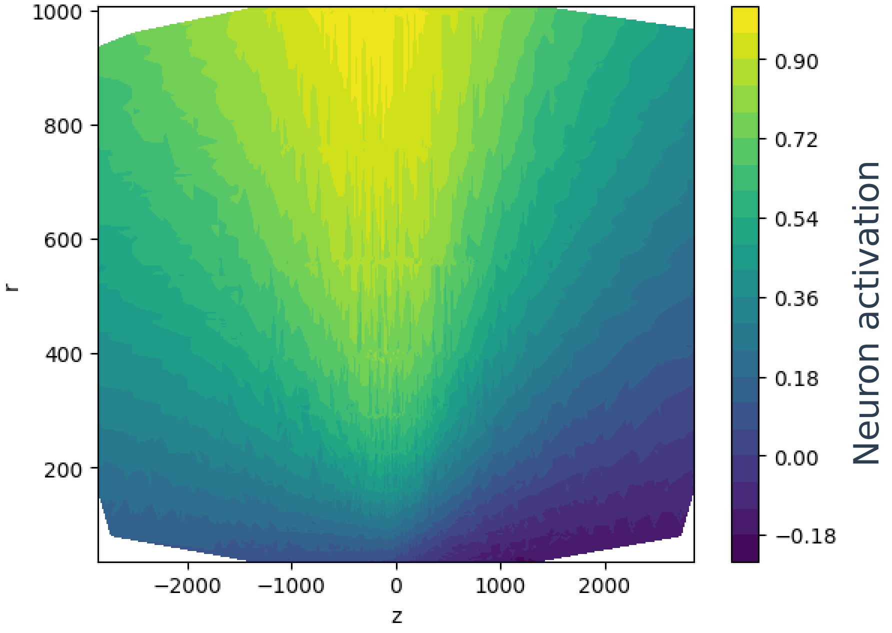


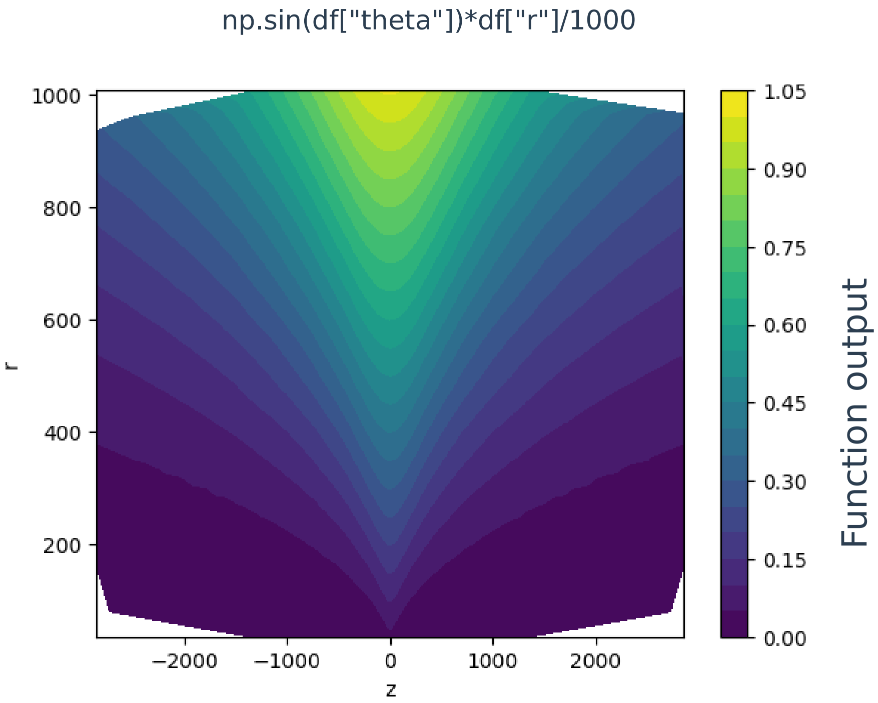


Neural Nets + Symbolic Regression

https://github.com/MilesCranmer/PySR

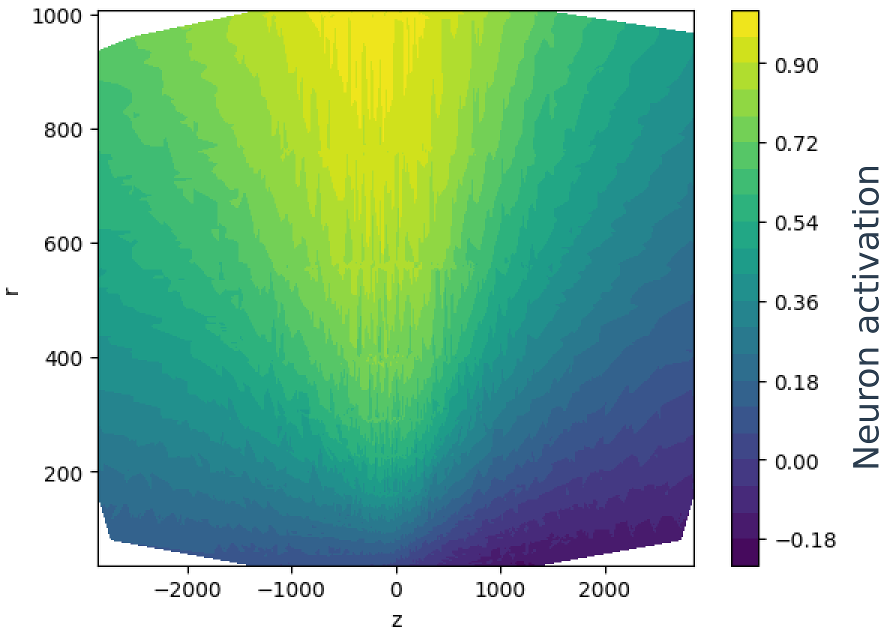
Symbolic regression

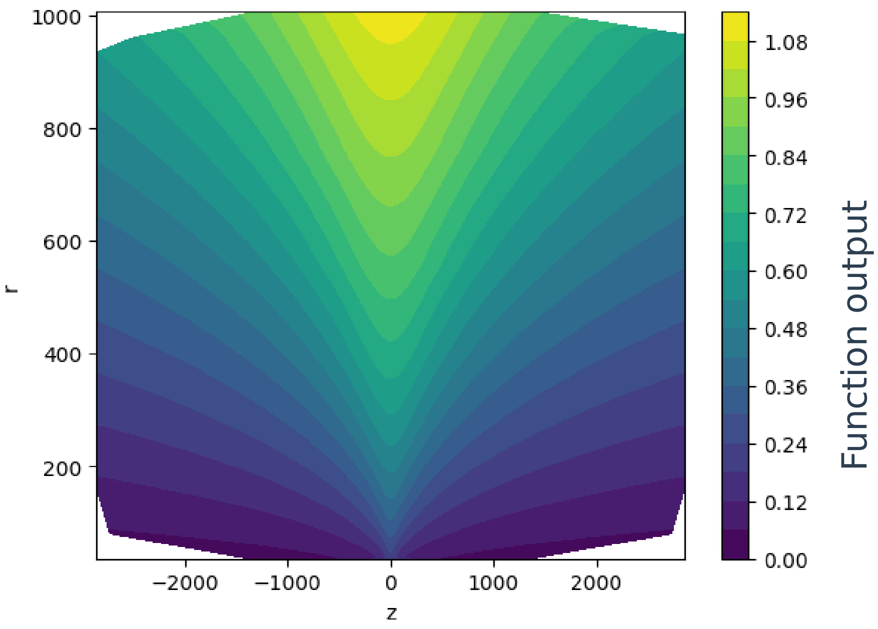




Symbolic regression

np.sin(df["theta"]) * 0.03505834 * np.sqrt(df["rho"])

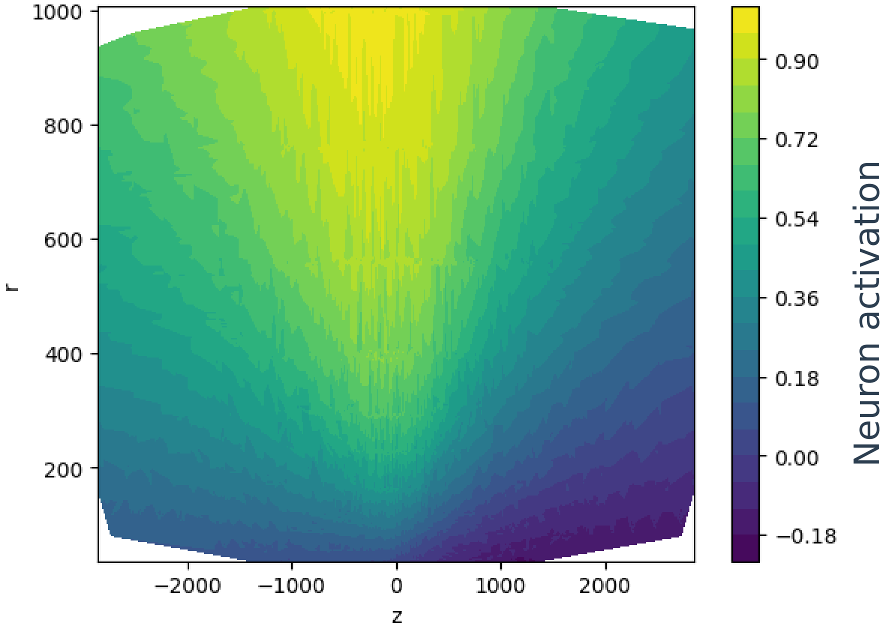


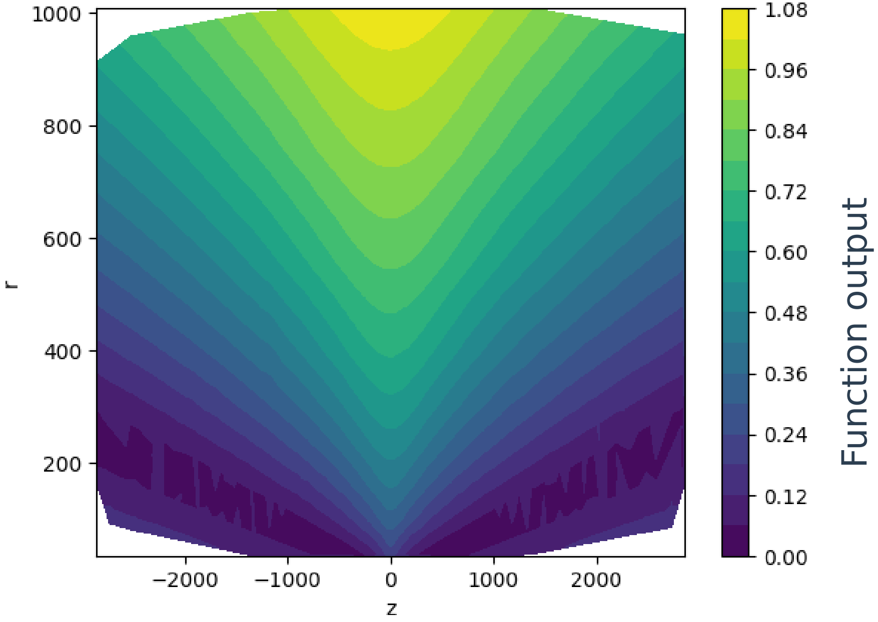


PySR

Symbolic regression

np.sqrt(0.0007077842 * (1 + np.cos(df["eta"])) * df["r"])

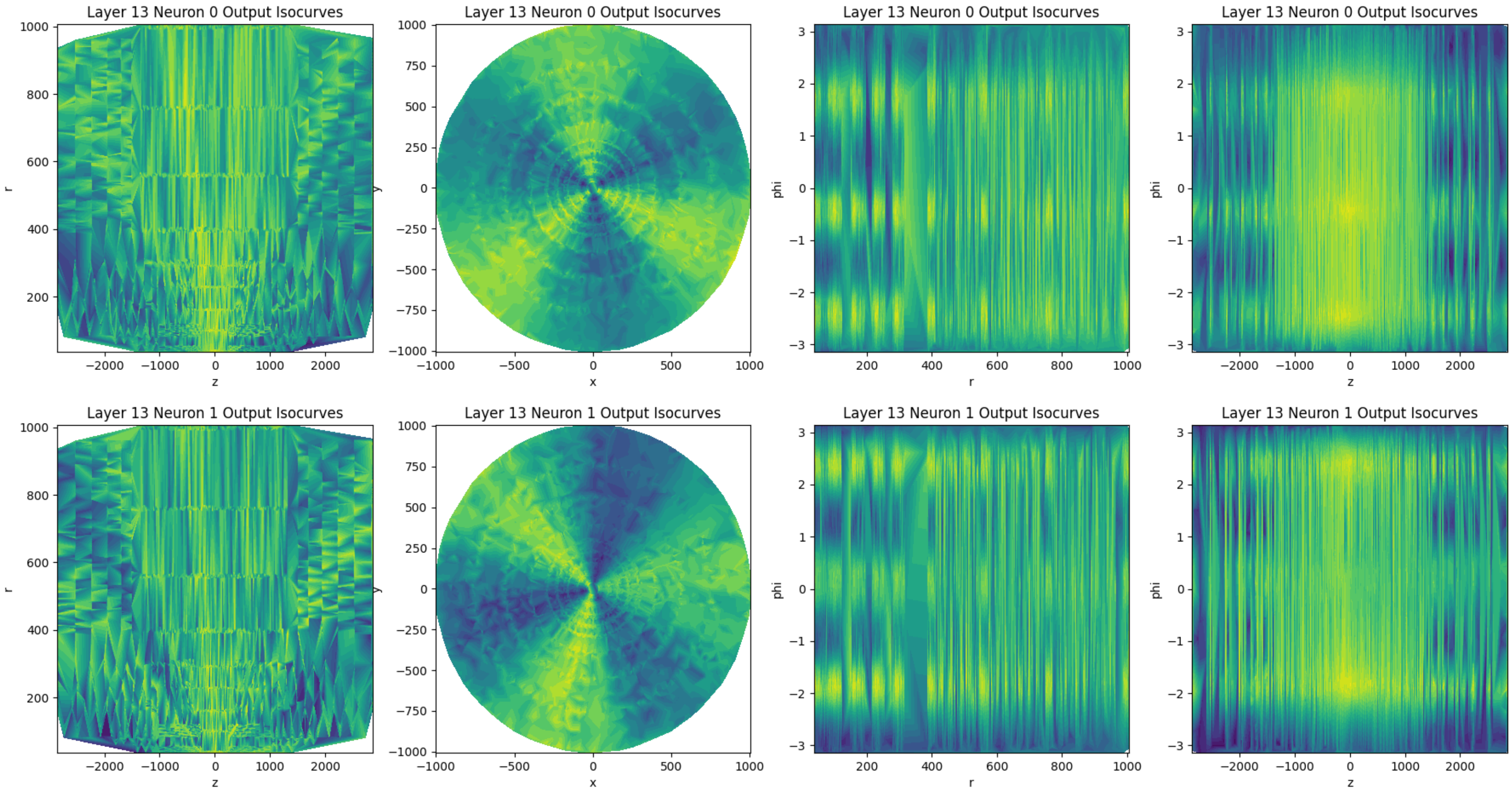




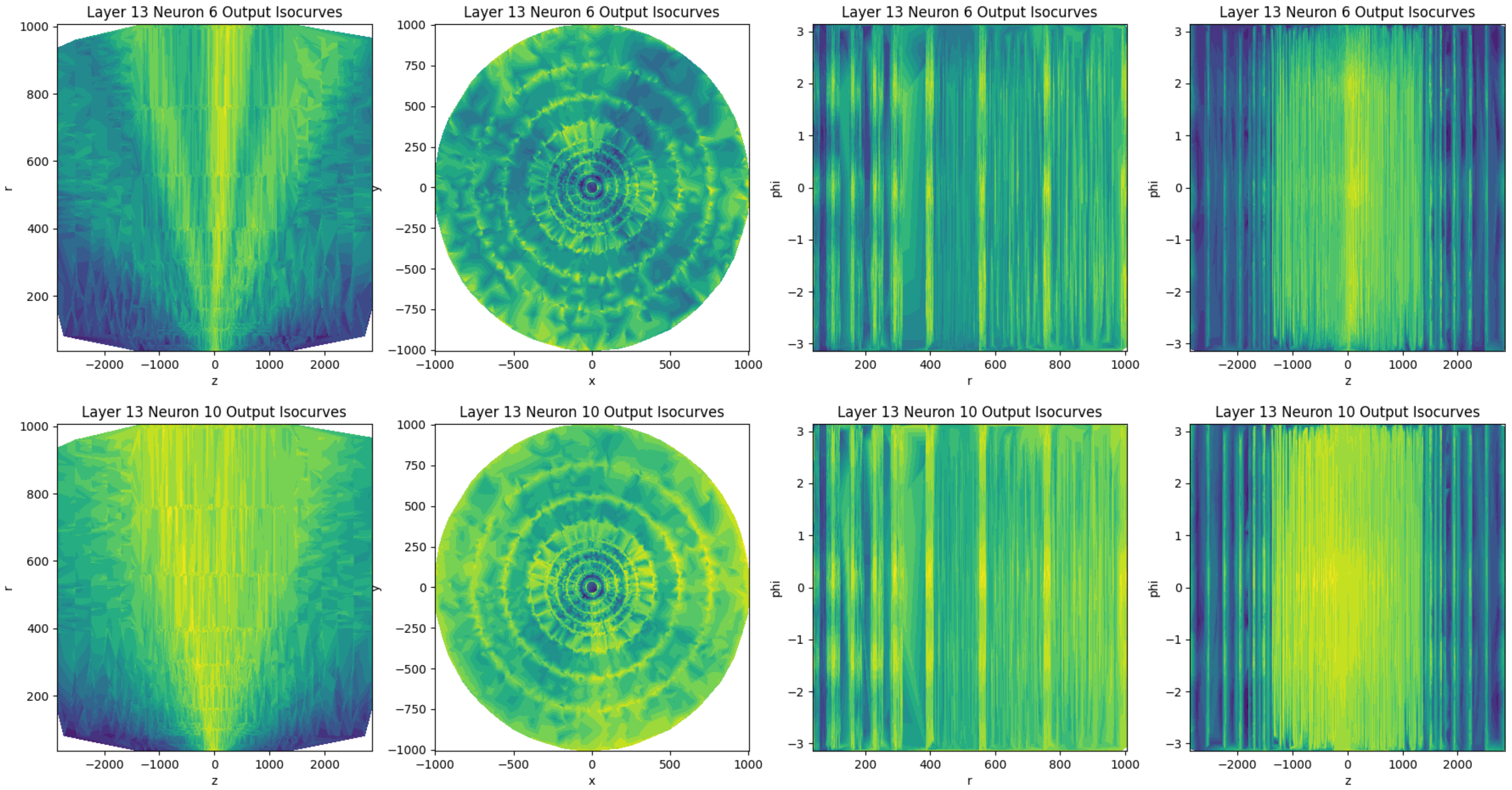
Interpretability

- ***How* is the prediction done?:**
  - What are the steps taken?
  - Does it predict track features (q/pT, eta, phi, d0, z0)?

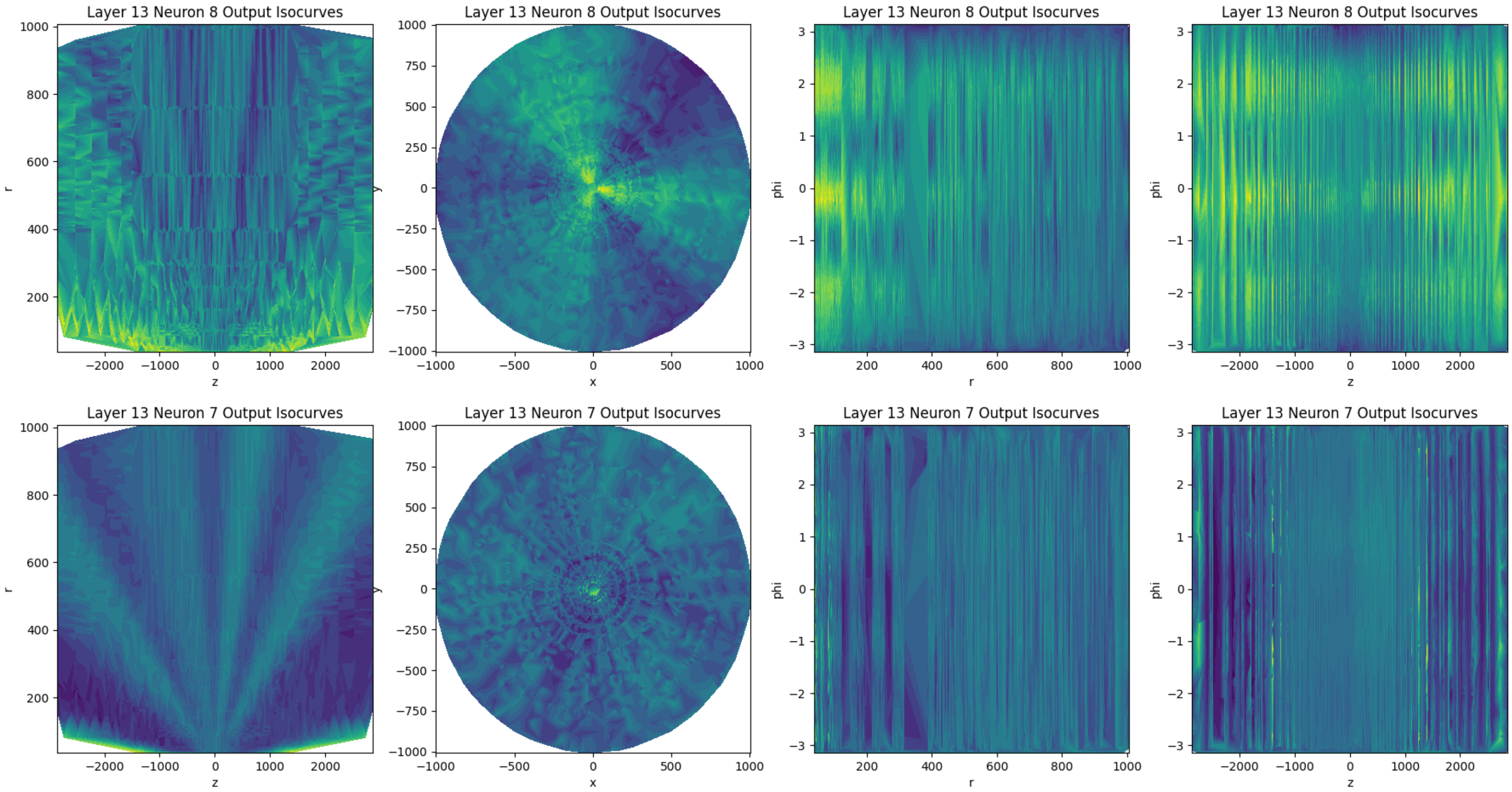
Latent space



Latent space



Latent space



Reconstructed high-level features

- **Assumption:**

  the model is using high-level features in the output latent space (12 neurons)

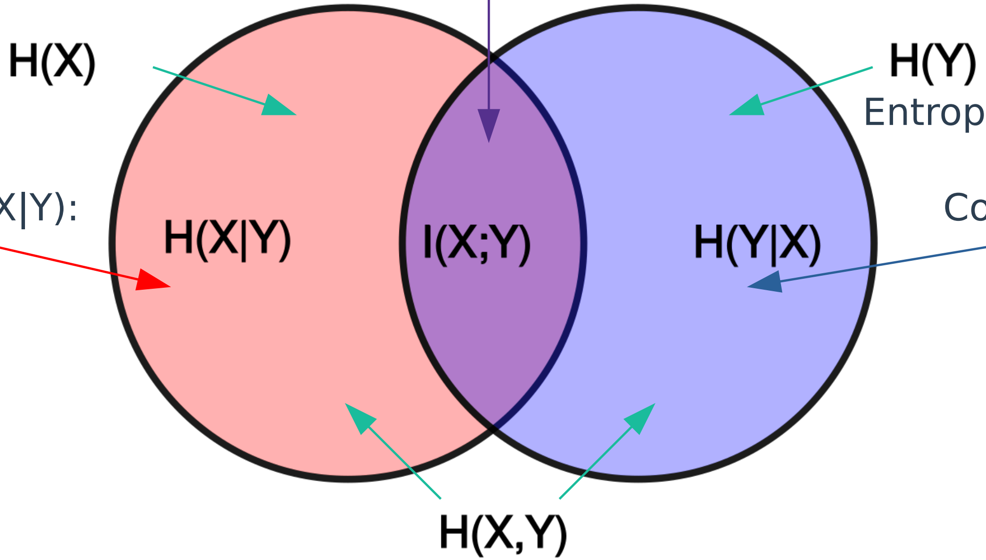
- **Approach:**
  - Information theory: conditional entropy of high-level features conditioned on the joined output latent space distributions → gives how much of the high-level feature can be predicted from the latent space alone

Entropy

Mutual Information:
$H(X) \cap H(Y)$

$\approx$ $Y = f(X)$
$X = g(Y)$

Conditional entropy $H(X|Y)$:
$H(X) \setminus H(Y)$



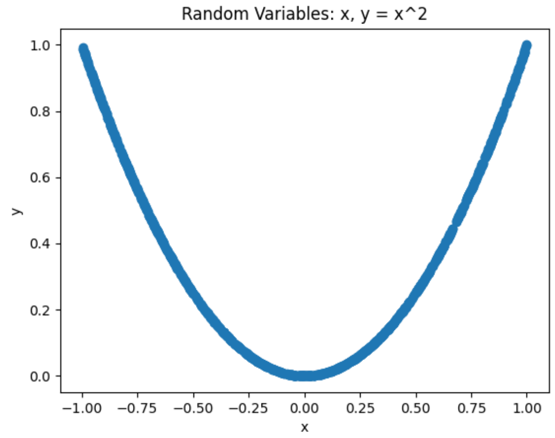
Entropy

Conditional entropy $H(Y|X)$:
$H(Y) \setminus H(X)$

Joint entropy:
$H(X) \cup H(Y)$

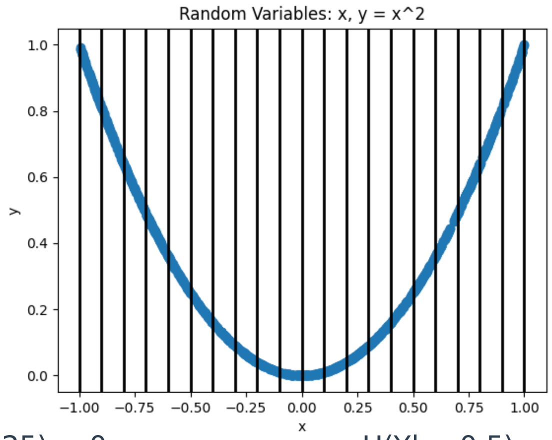
Conditional entropy

$$\mathrm{H}(Y|X) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)}$$

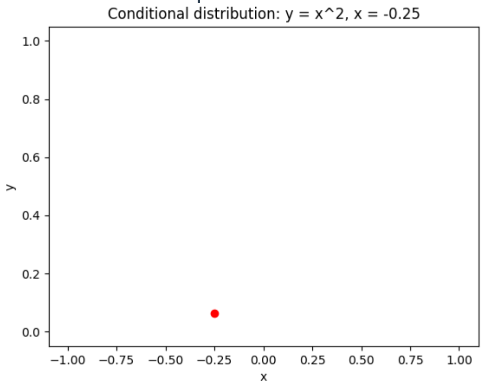


Conditional entropy

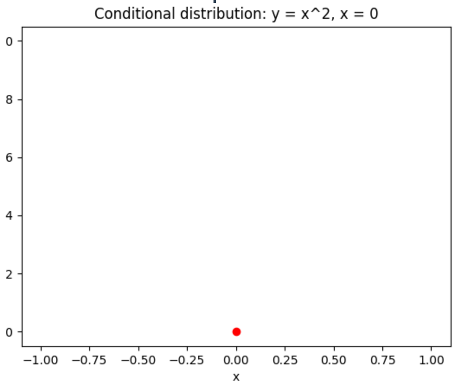
$$\mathrm{H}(Y|X) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)}$$



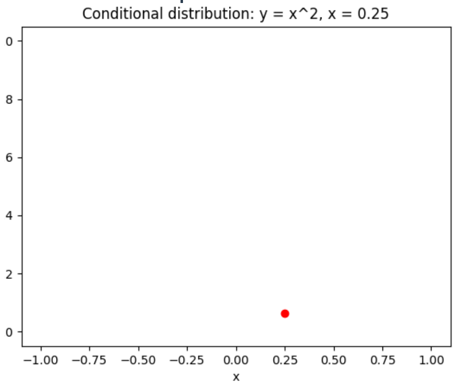
H(Y|x=-0.25) = 0



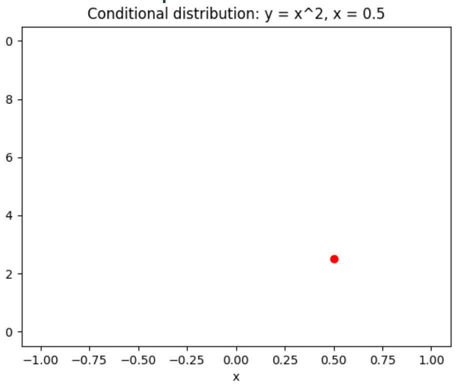
H(Y|x=0) = 0



H(Y|x=0.25) = 0

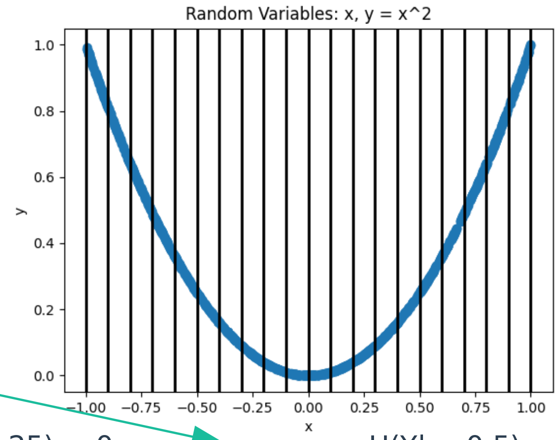


H(Y|x=0.5) = 0



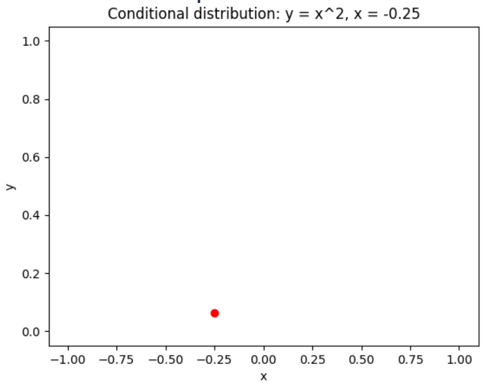
Conditional entropy

$$H(Y|X) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)}$$

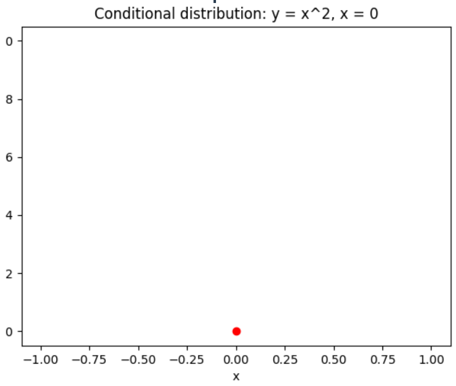


H(Y|X) = 0

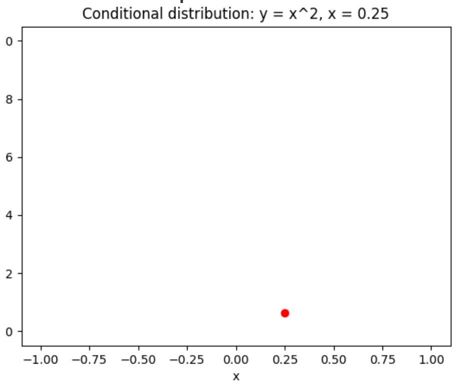
H(Y|x=-0.25) = 0



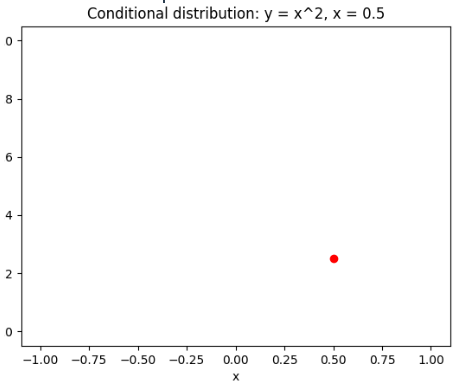
H(Y|x=0) = 0



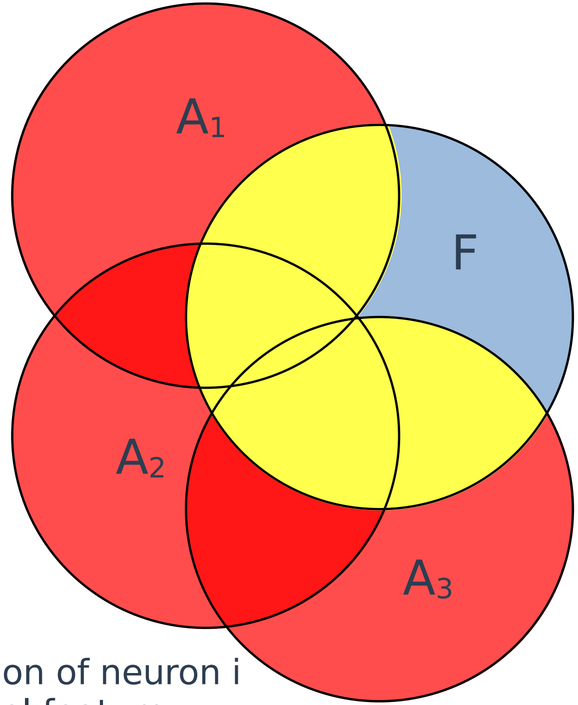
H(Y|x=0.25) = 0



H(Y|x=0.5) = 0

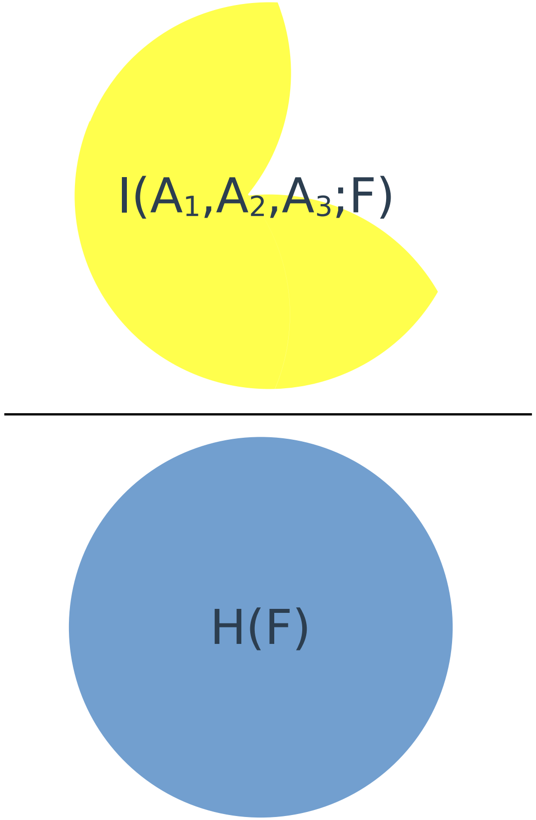


Proficiency

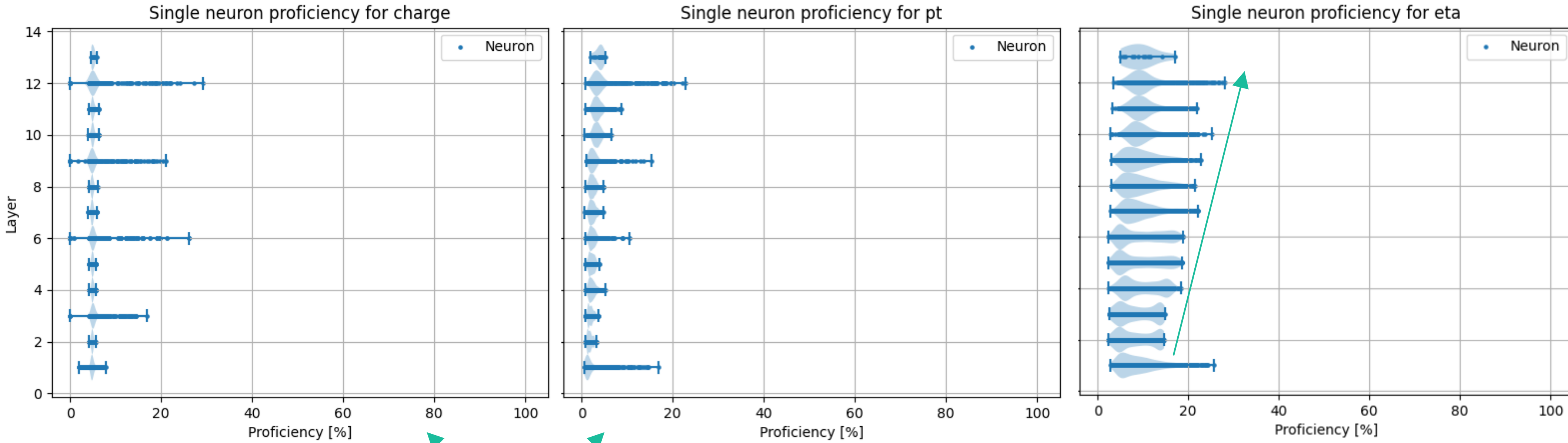


A_i: activation of neuron i
F: high-level feature

Proficiency =

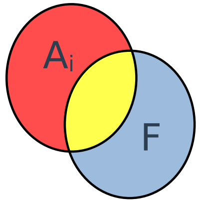


High-level variables from single neurons



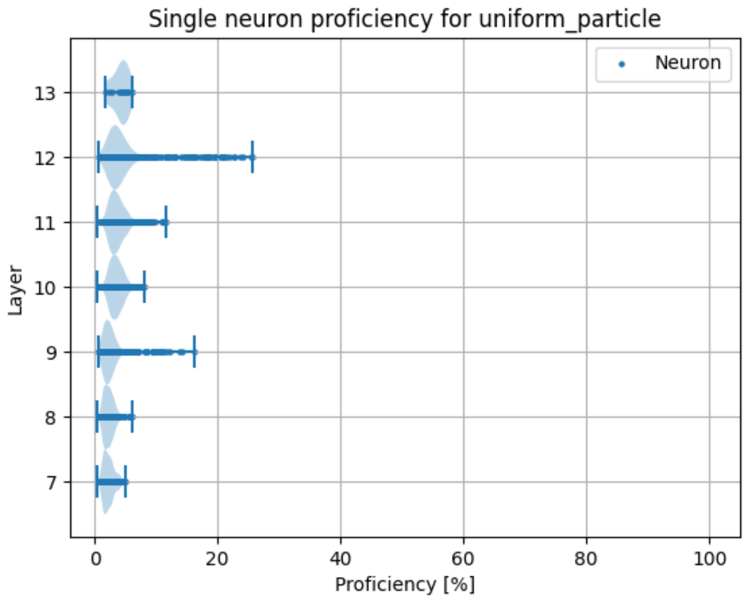
Cannot physically estimate from a single hit

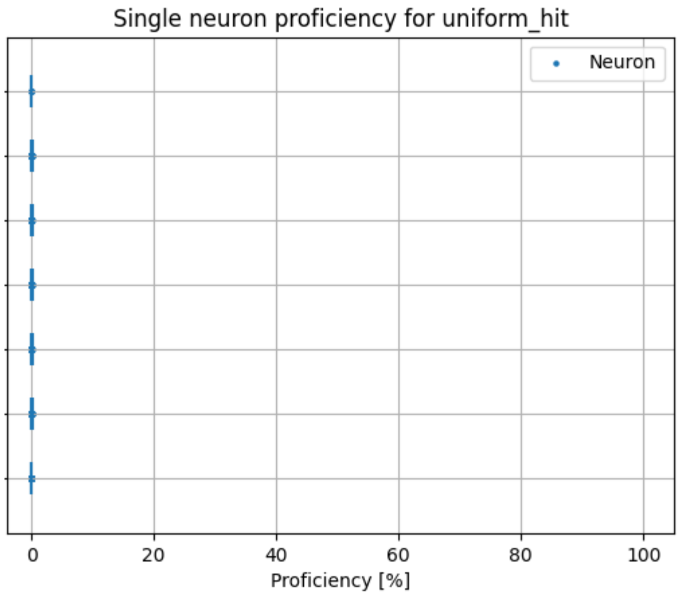
7 events: 118 115 hits



From Scikit-learn Mutual information calculation

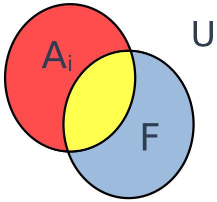
Random variables from single neurons: uniform





Should both be close to 0: Any particle can hit any cell

Indicates not enough data



Unique particles can be found

Unique hit cannot be found (= no overfit)

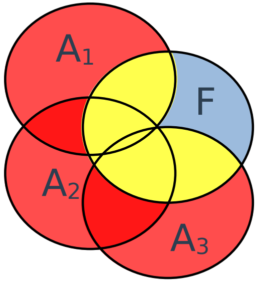
7 events: 118 115 hits

From Scikit-learn Mutual information calculation

Layer proficiency

- **Scikit-learn do only single dimensional X and Y**

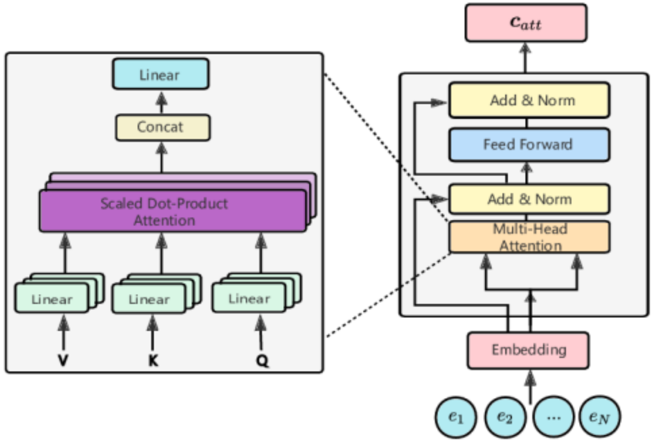
Theory: To estimate the joint MI between $\{X_1, X_2, \ldots, X_m\}$ and Y, the high-dimensional variables $\{X_1, X_2, \ldots, X_m\}$ should be treated as a whole and n_x would be defined as the number of points in the m-dimensional space.



Parameter regression

TrackFormer

- **Transformer for track parameter regression**
- **Tested on several dataset: ToyTracks, Acts, TrackML**
- **Regression of pt and pz**
- **Shown promising results**



Sequences were padded to a fixed length

Dataset selection

Selections:

- nhits >= 3
- 0.5 <= pT <= 10   [GeV]
- |v_x| < 1 && |v_y|< 1   [mm]
- |eta| <= 1

Before:

Training: 11 222 273 particles
Validation: 1 334 273 particles
Testing: 1 404 273 particles

$\xrightarrow{/ 10}$

After:

Training: 1 232 896 particles
Validation: 154 082 particles
Testing: 153 788 particles

Training

Architecture:
- input_dim: 3
- model_dim: 128
- num_classes: 2
- num_heads: 4
- num_layers: 2

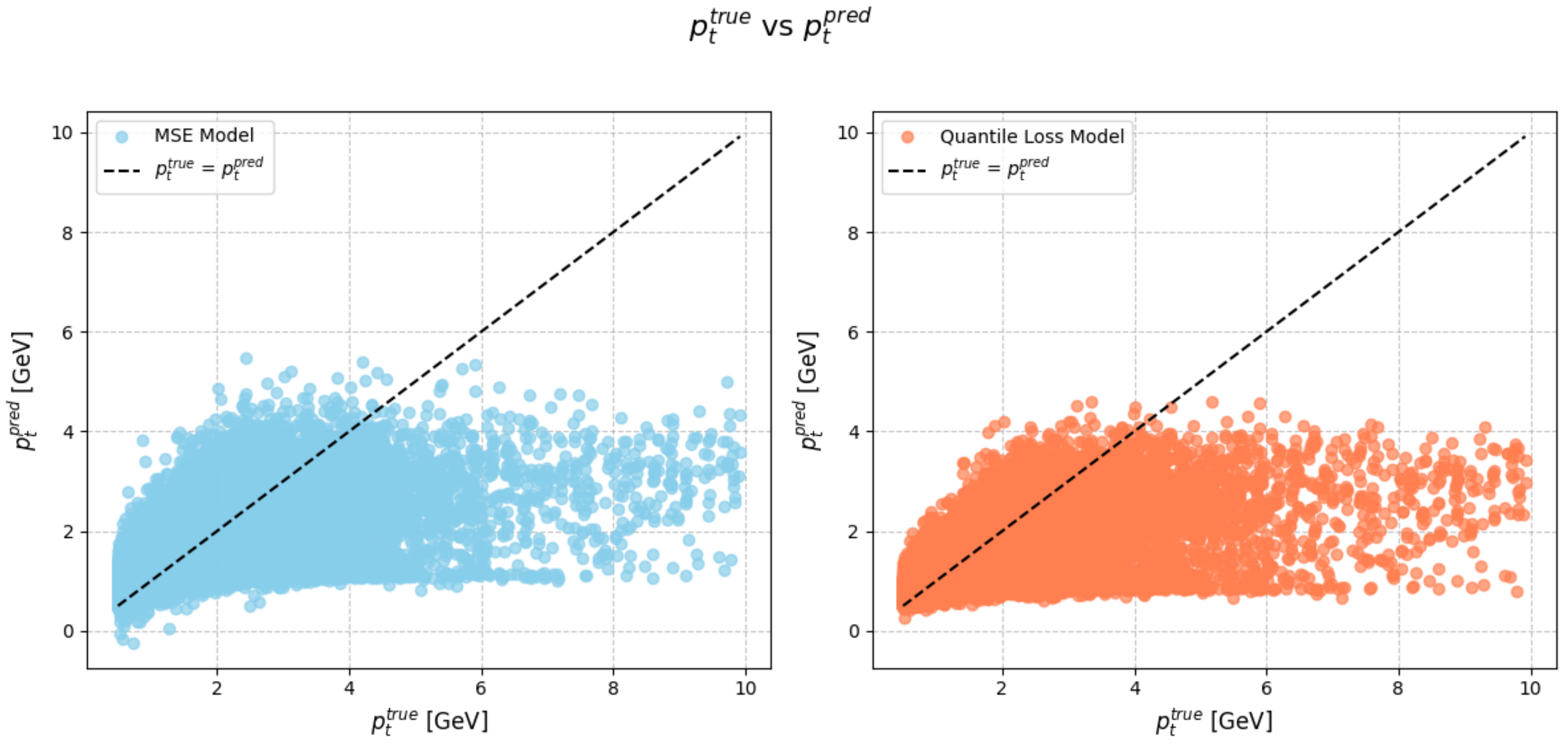
Training:
- warmup: 100
- lr: 0.0005
- dropout: 0.1
- input_dropout: 0.1
- batch_size: 1024
- max_epochs: 100

Saving:
- monitor: val_loss
- mode: min

Variables:
- input:
  - tx, ty, tz
- input:
  - tr, tphi, tz
- target:
  - pt, pz

More results

r phi z



Dataset update

Selections:
- nhits >= 3
- 0.5 <= pT <= 10   [GeV]
- |v_x| < 1 && |v_y|< 1   [mm]
- |eta| <= 1

TrackML Kaggle:

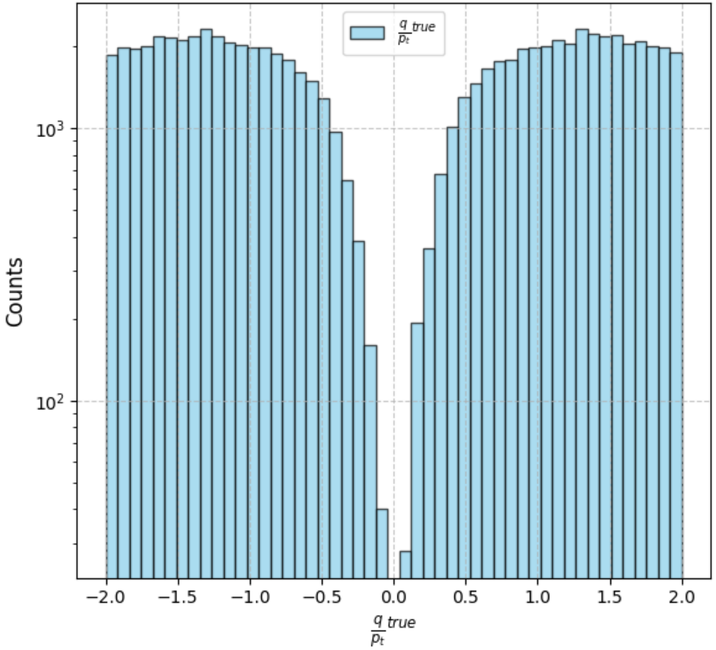
Training: 1 232 896 particles
Validation: 154 082 particles
Testing: 153 788 particles

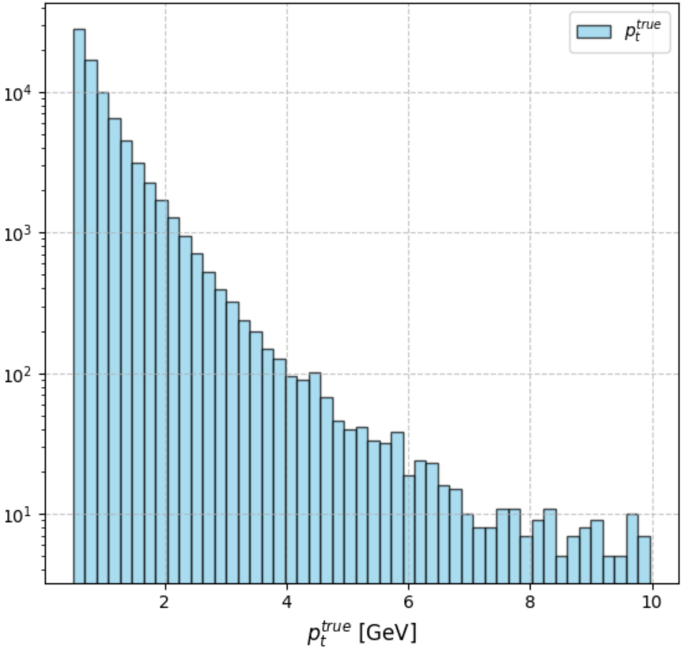
→

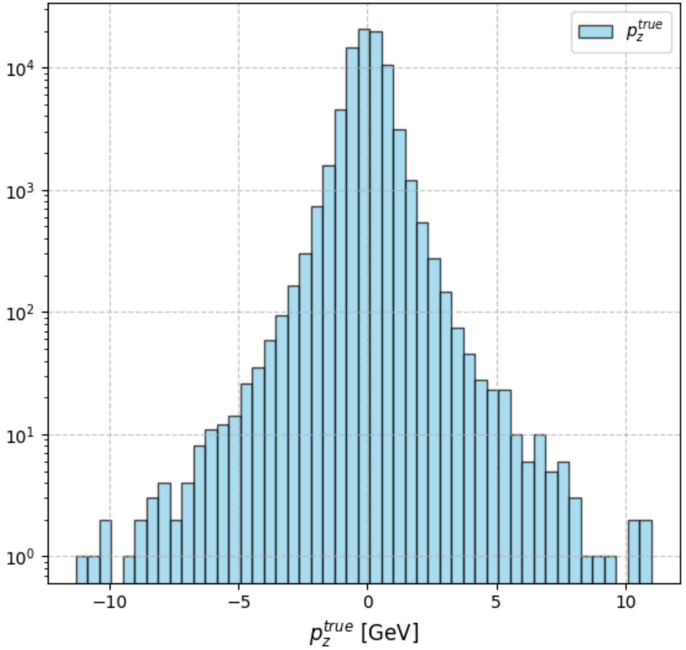
TrackML Zenodo:
(first file)

Training: 629 265 particles
Validation: 78 107 particles
Testing: ~78 656 particles

Target variables







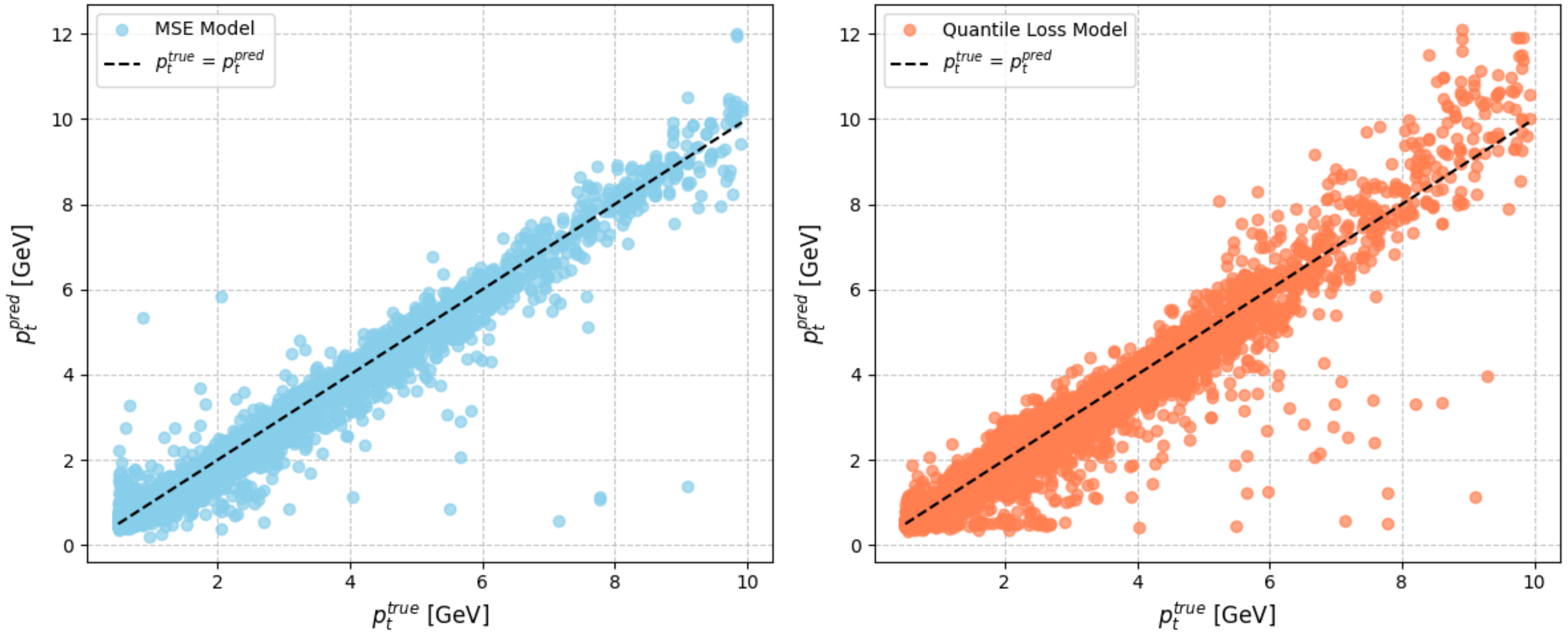
More results

x y z

masked

TrackML Kaggle

p_t^{true} vs p_t^{pred}

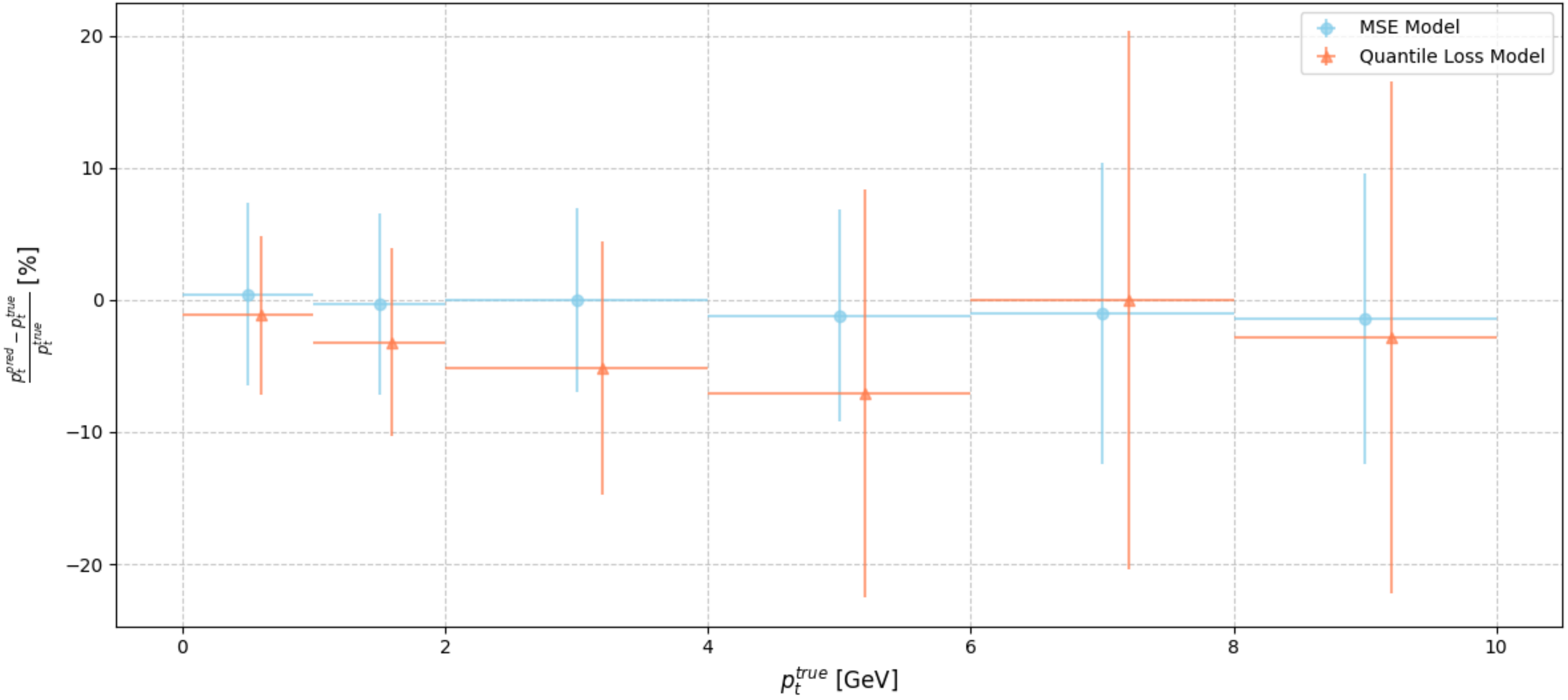


Resolution

x y z

masked

TrackML Zenodo



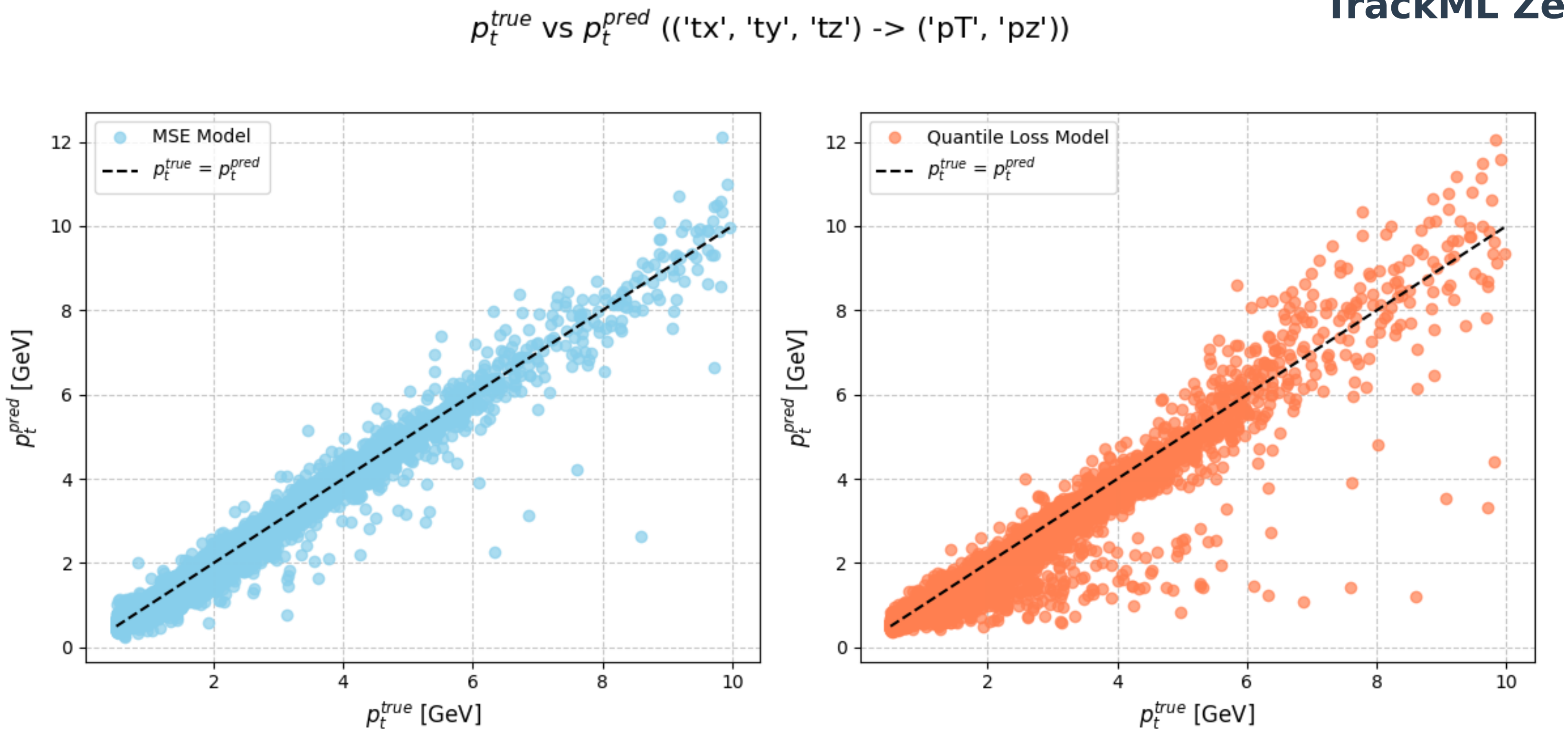
Quantile loss drops a bit with dataset change

More results

x y z

masked

TrackML Zenodo

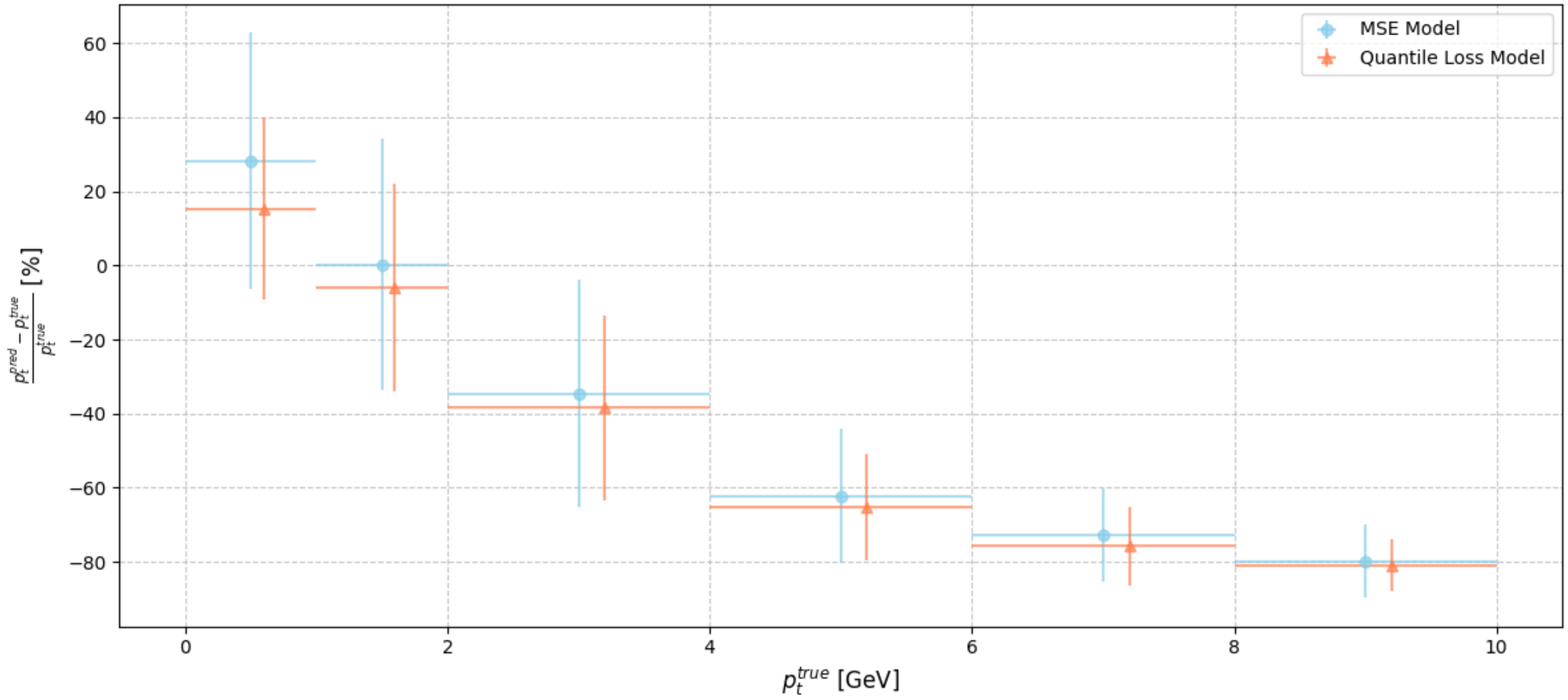


Resolution

r phi z

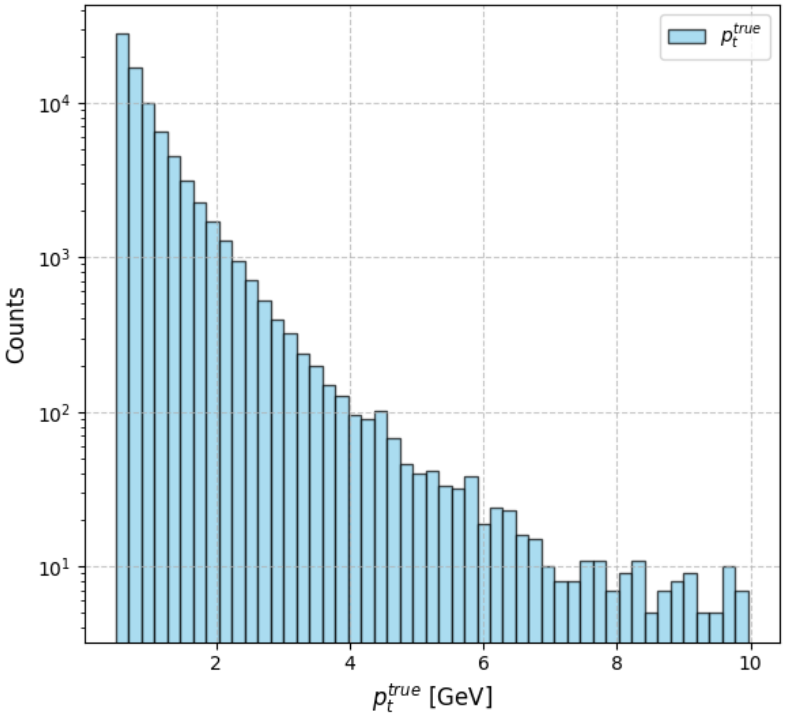
TrackML Zenodo

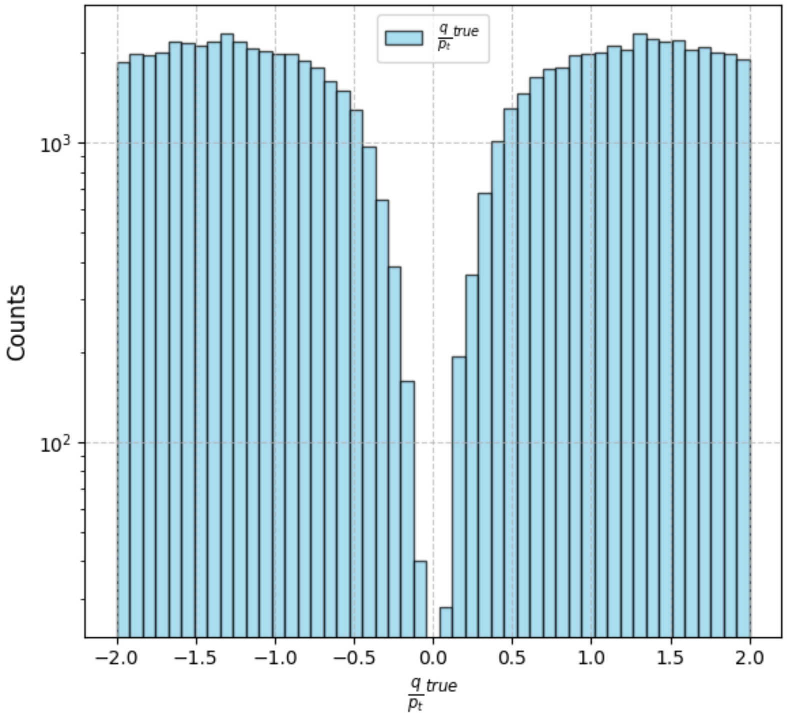
masked



q/pT

Target variable:



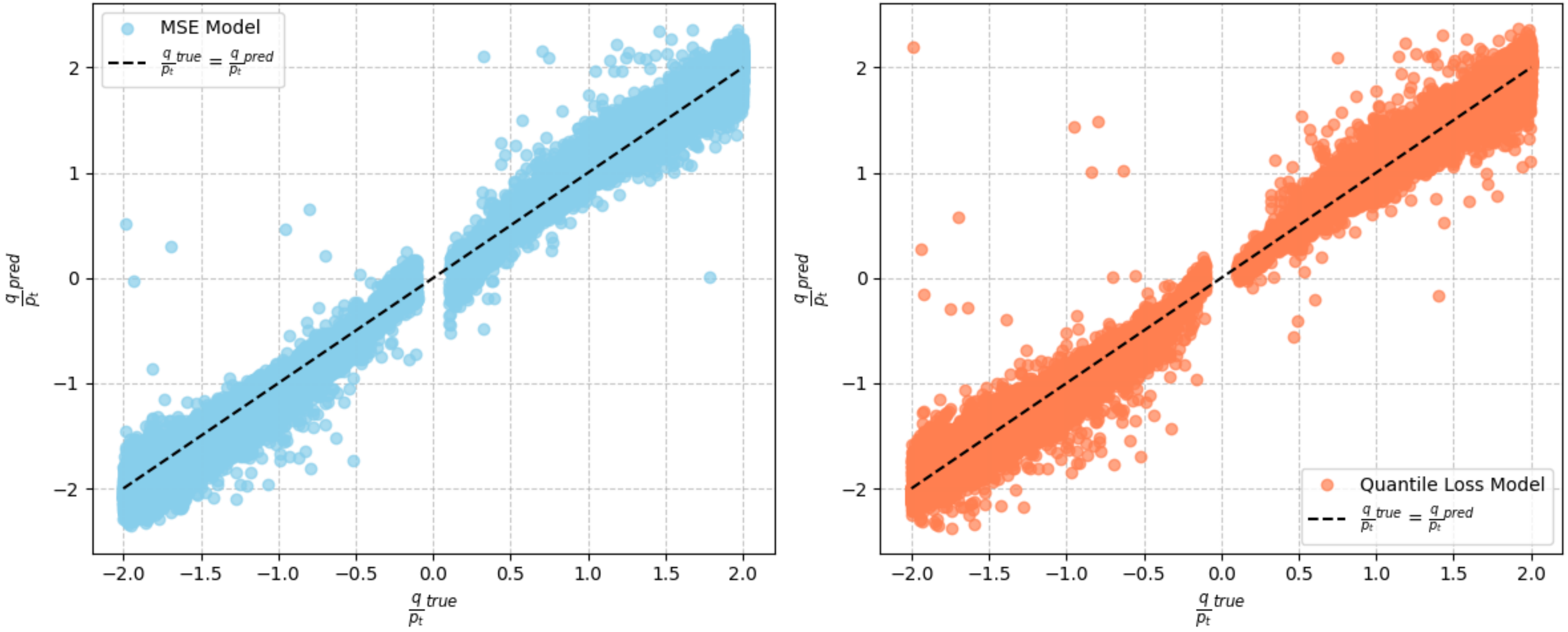


More results

x y z

TrackML Zenodo

$\frac{q}{p_t}$ true vs $\frac{q}{p_t}$ pred (('tx', 'ty', 'tz') -> ('qopT', 'pz'))

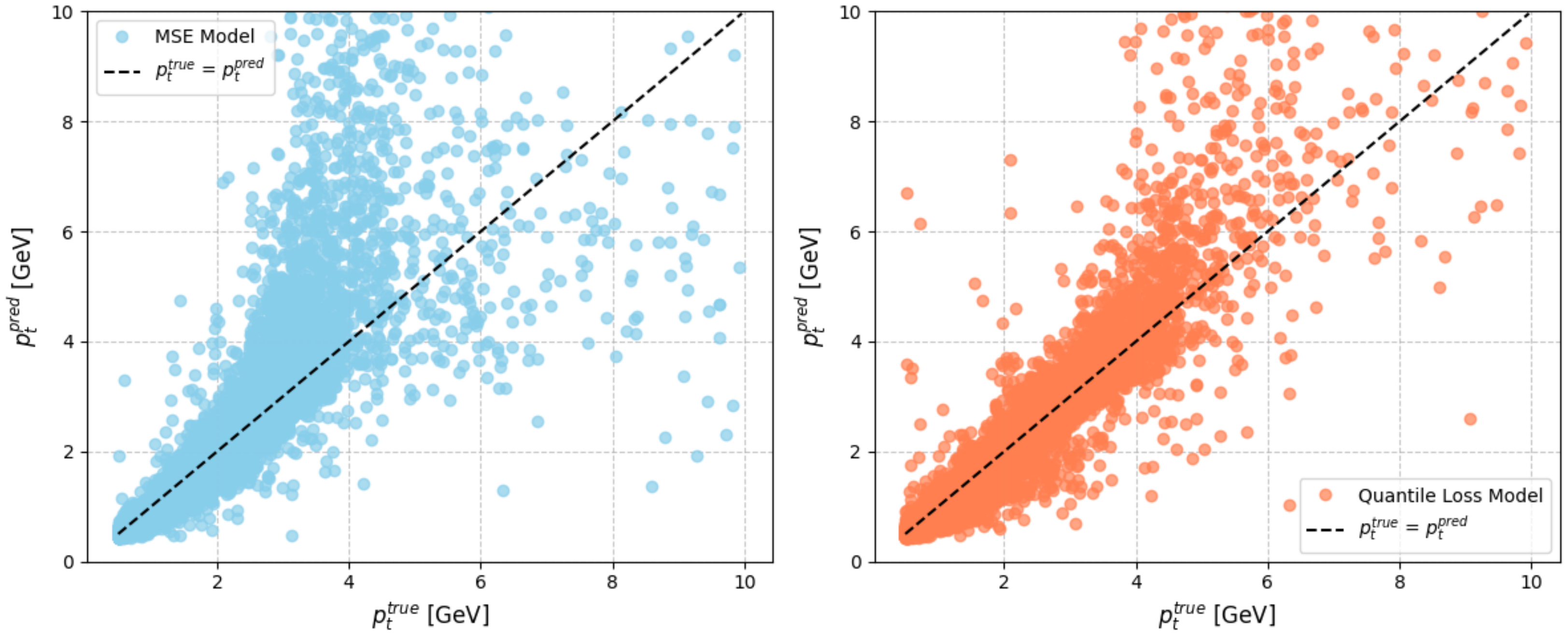


More results

x y z

TrackML Zenodo

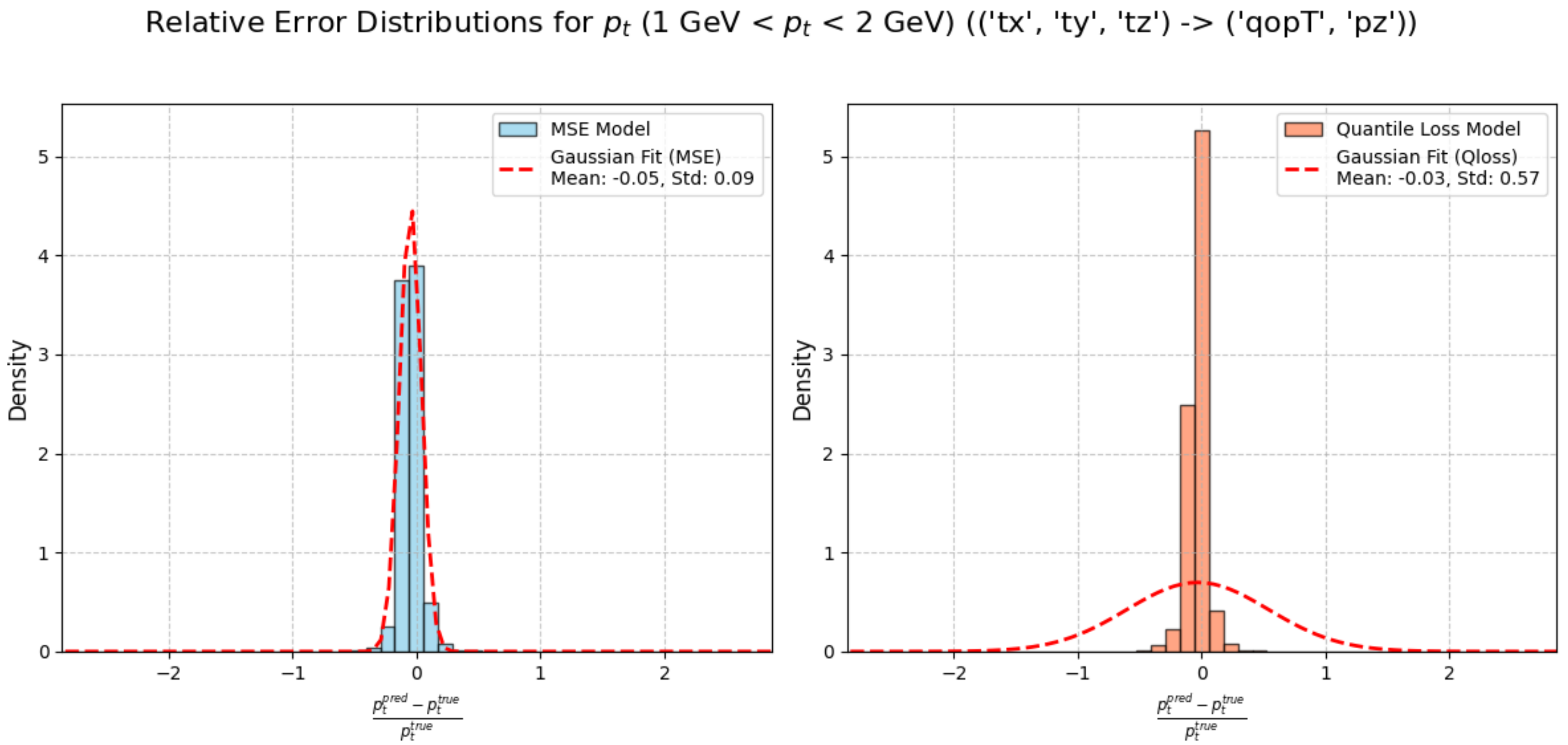
p_t^{true} vs p_t^{pred} (('tx', 'ty', 'tz') -> ('qopT', 'pz'))



Resolution

x y z

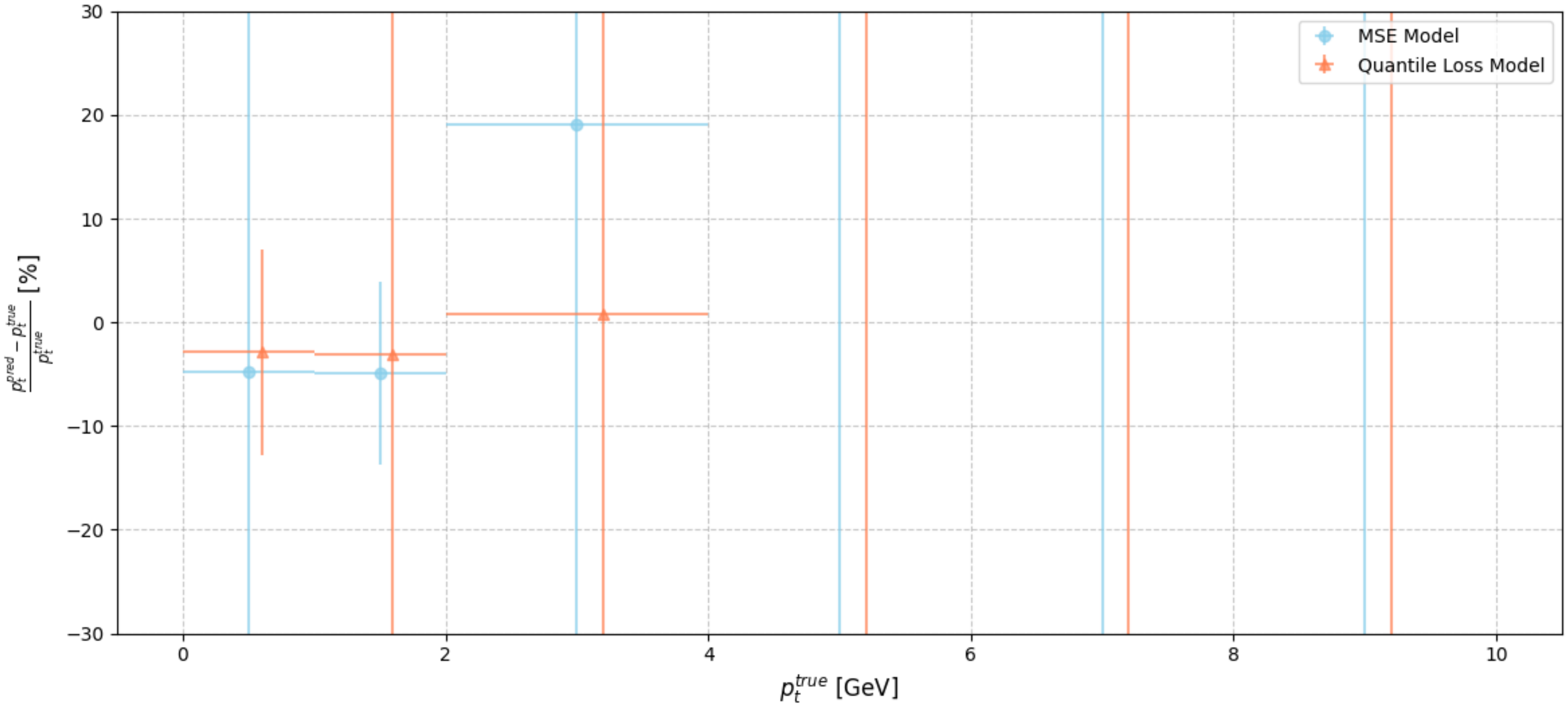
TrackML Zenodo



Resolution

x y z

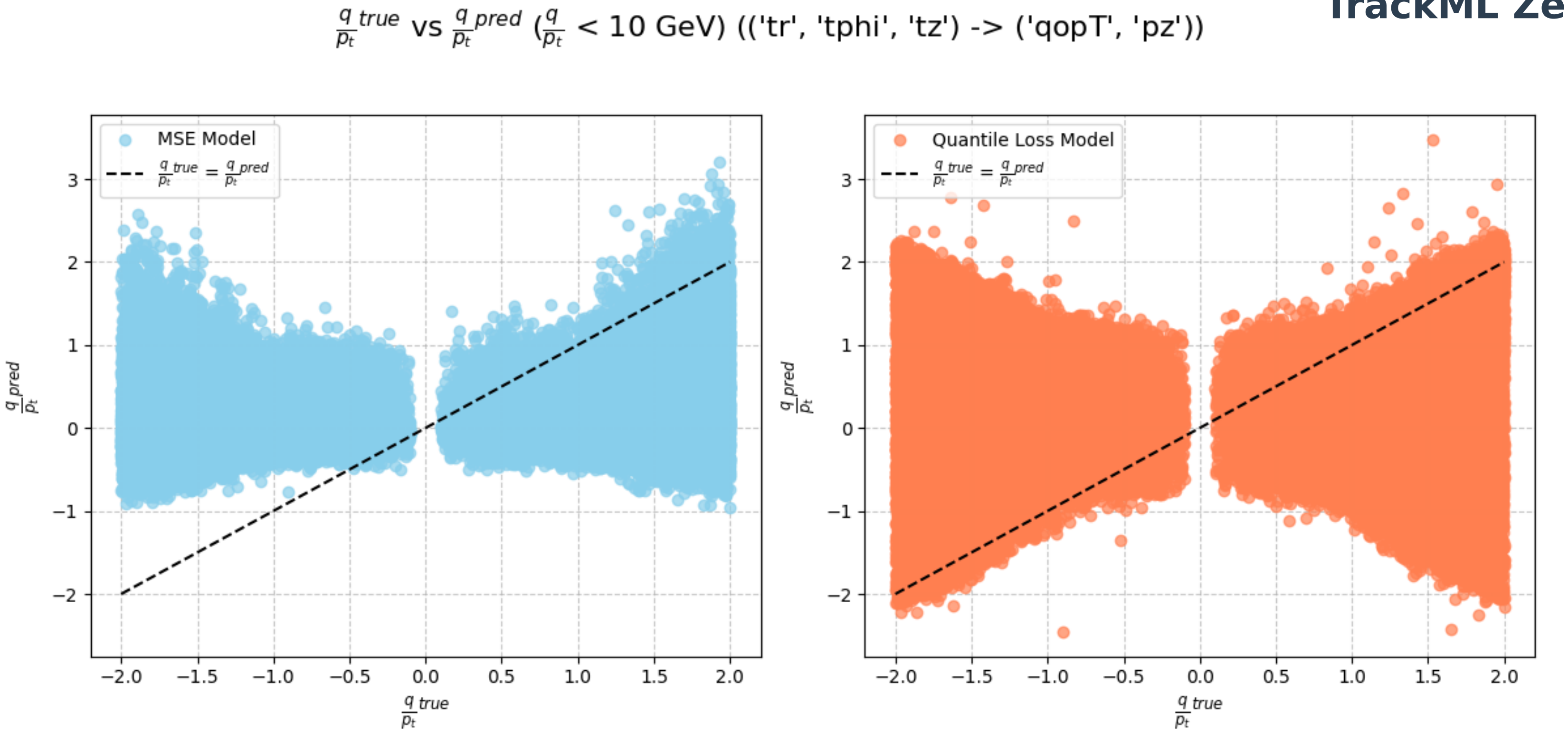
TrackML Zenodo



More results

r phi z

TrackML Zenodo

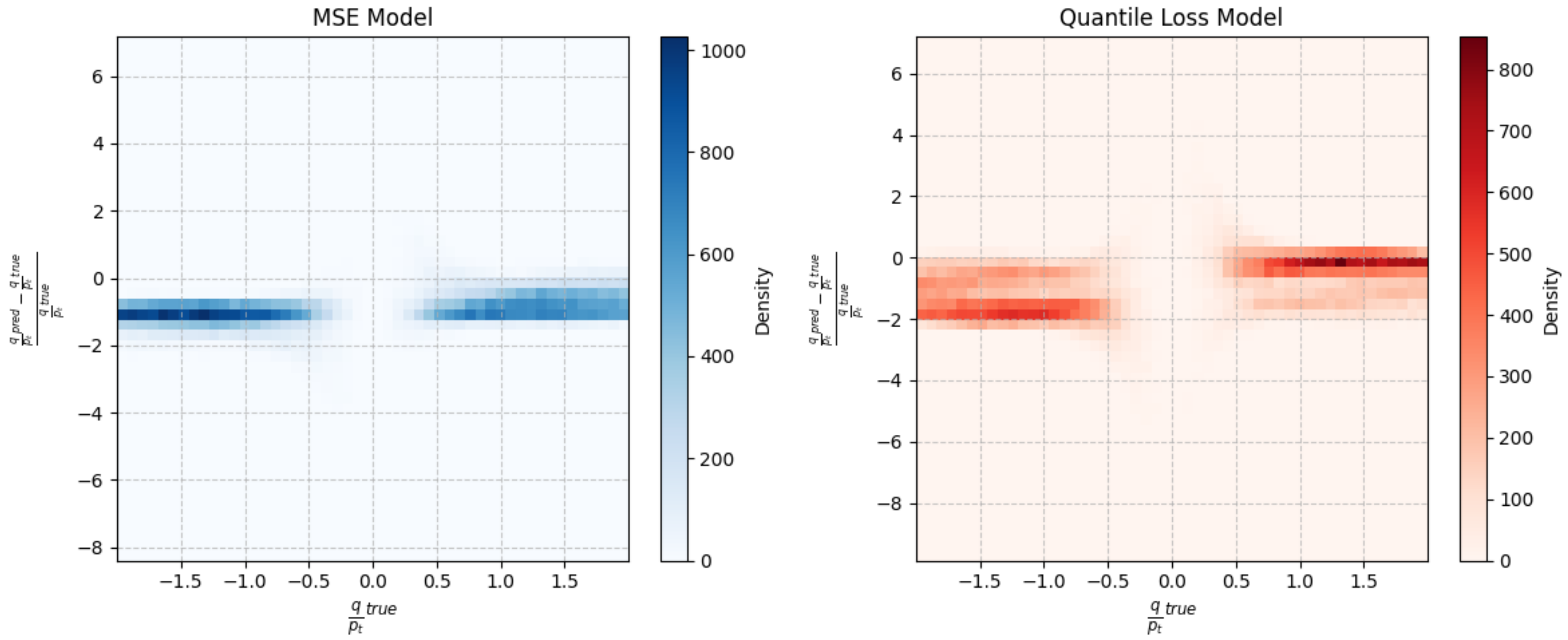


Resolution

r phi z

TrackML Zenodo

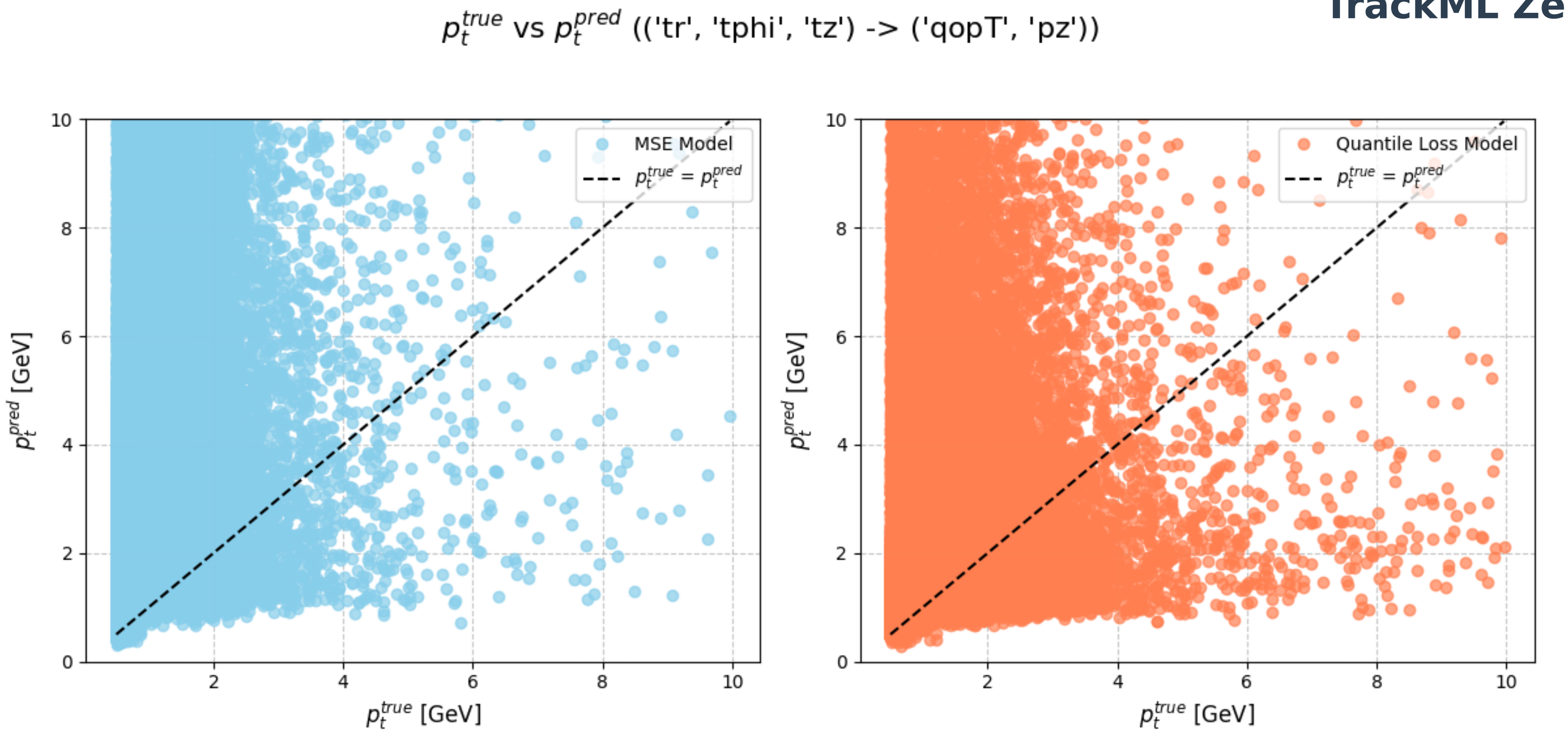
Relative error resolution for $\frac{q}{p_t}$ (('tr', 'tphi', 'tz') -> ('qopT', 'pz'))



More results

r phi z

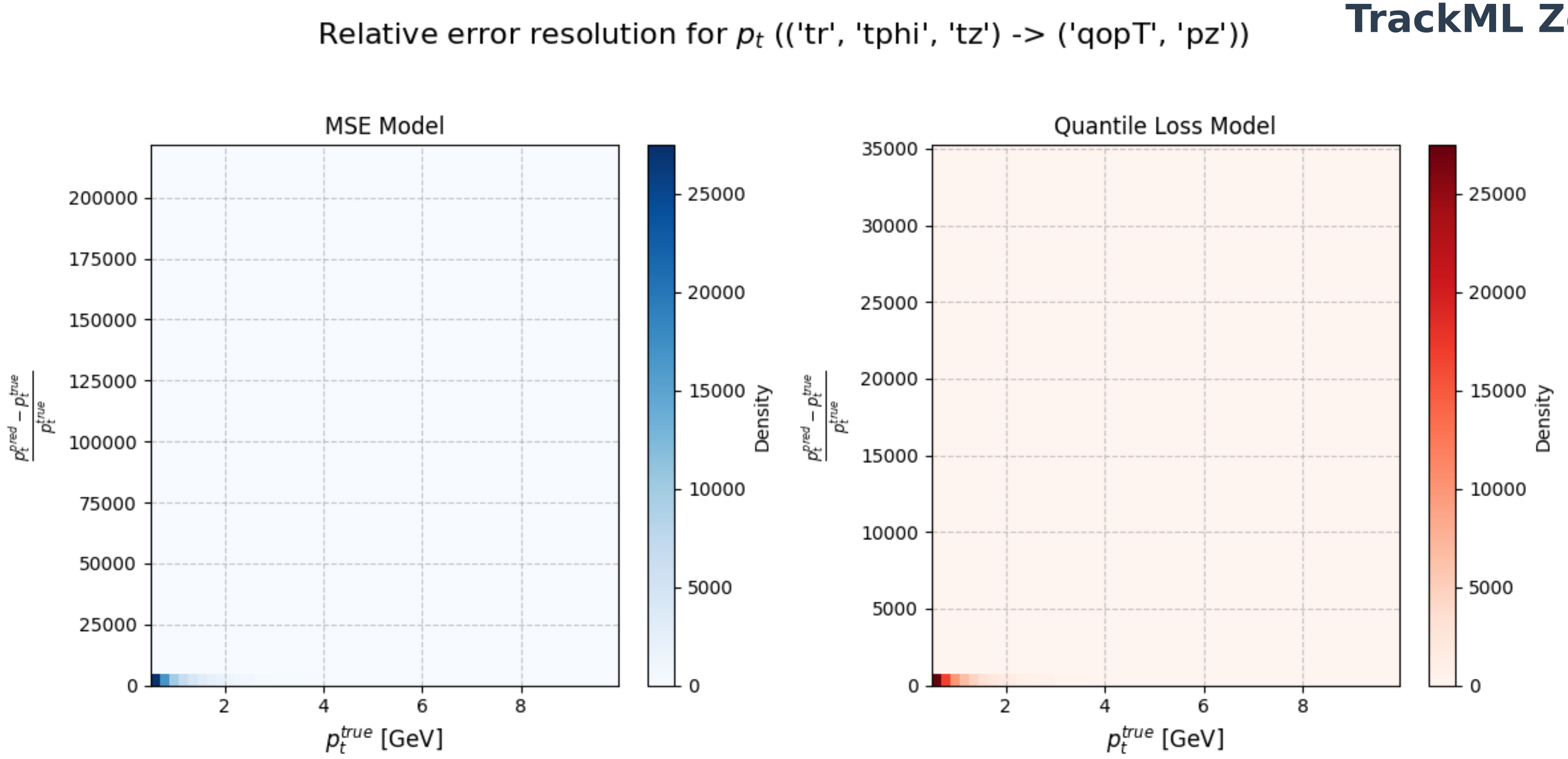
TrackML Zenodo



Resolution

r phi z

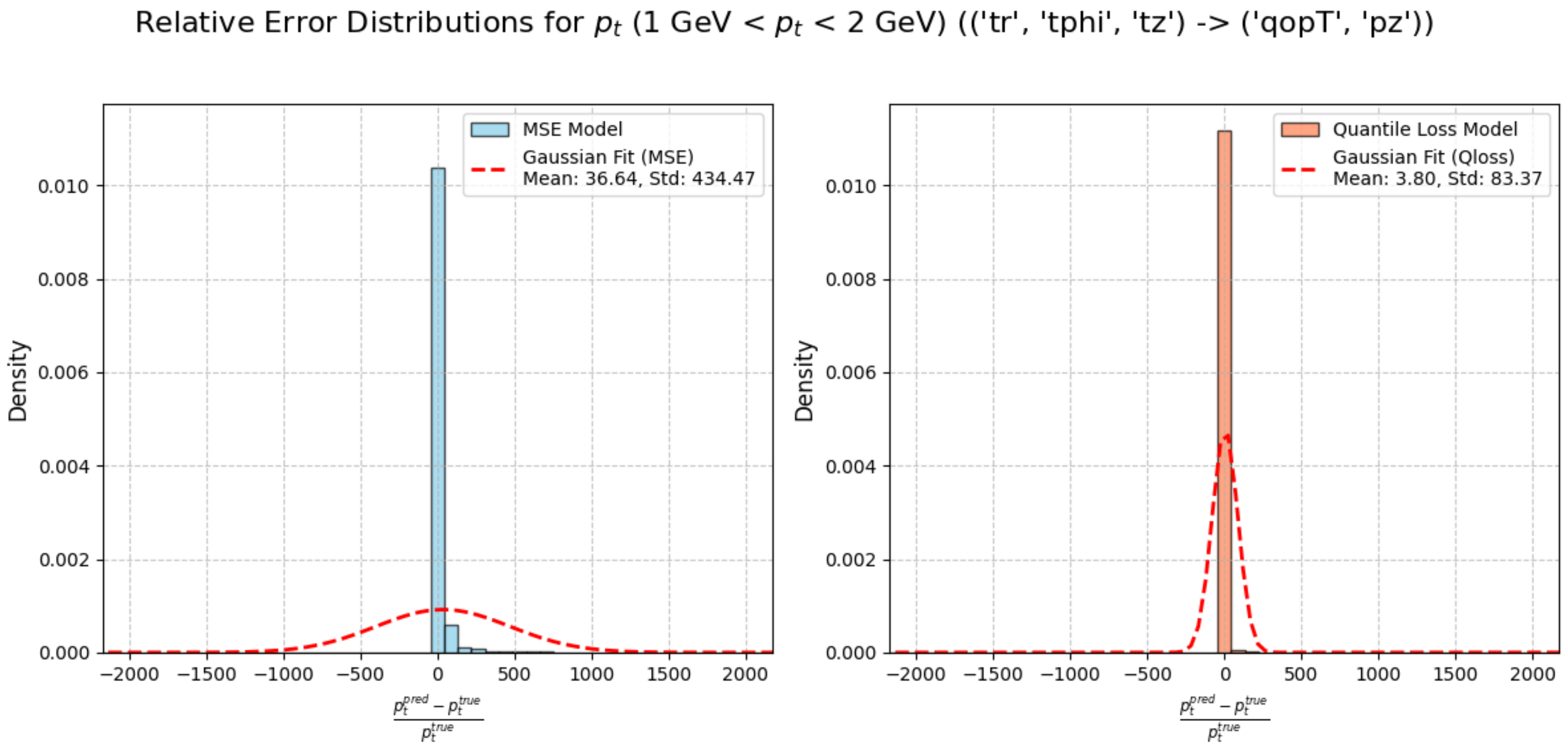
TrackML Zenodo



Resolution

r phi z

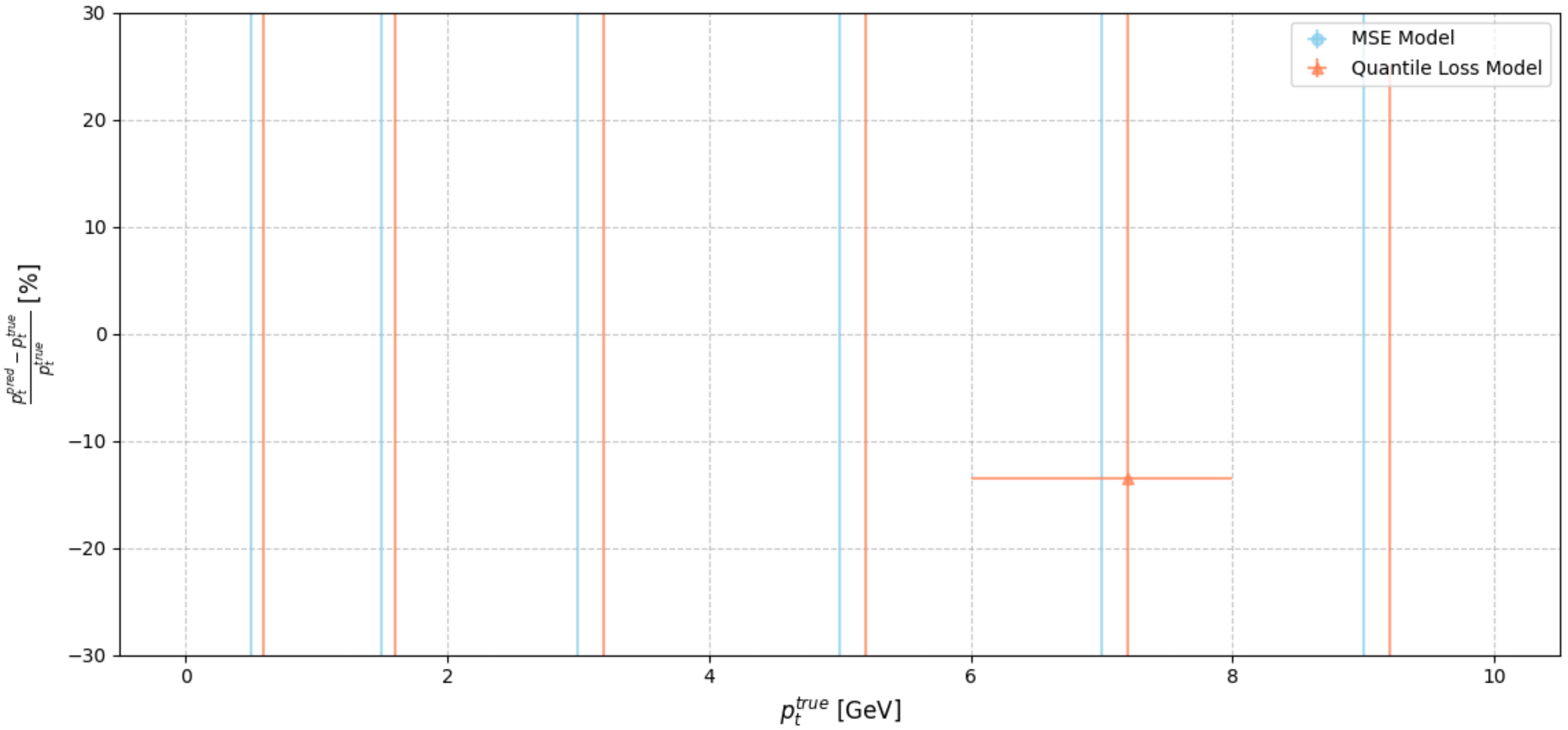
TrackML Zenodo



Resolution

r phi z

TrackML Zenodo



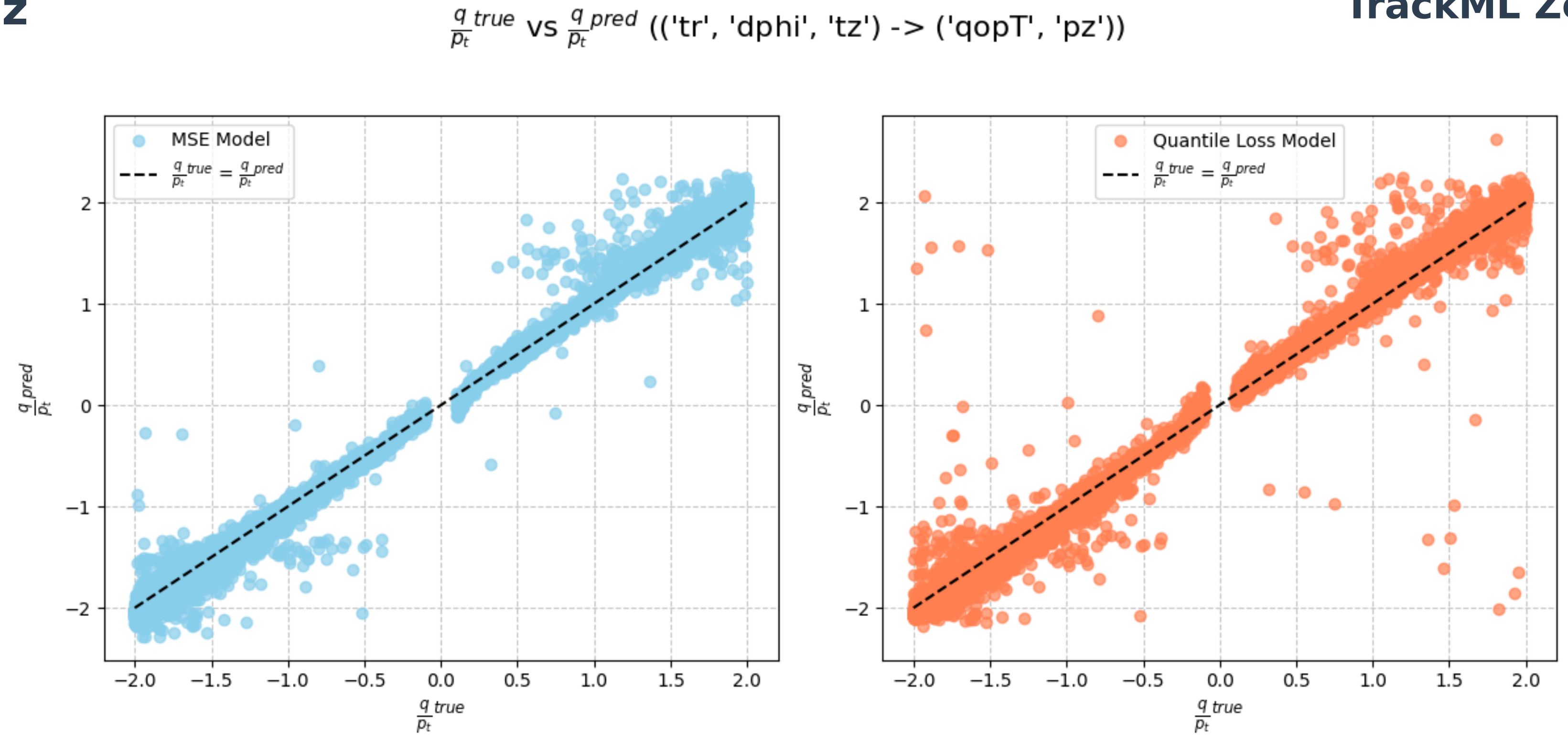
dPhi

- Use dphi instead of phi as input
- dphi = Phi – phi_0 (phi_0 = phi of first hit)
- Introduces circular symmetry

More results

r dphi z

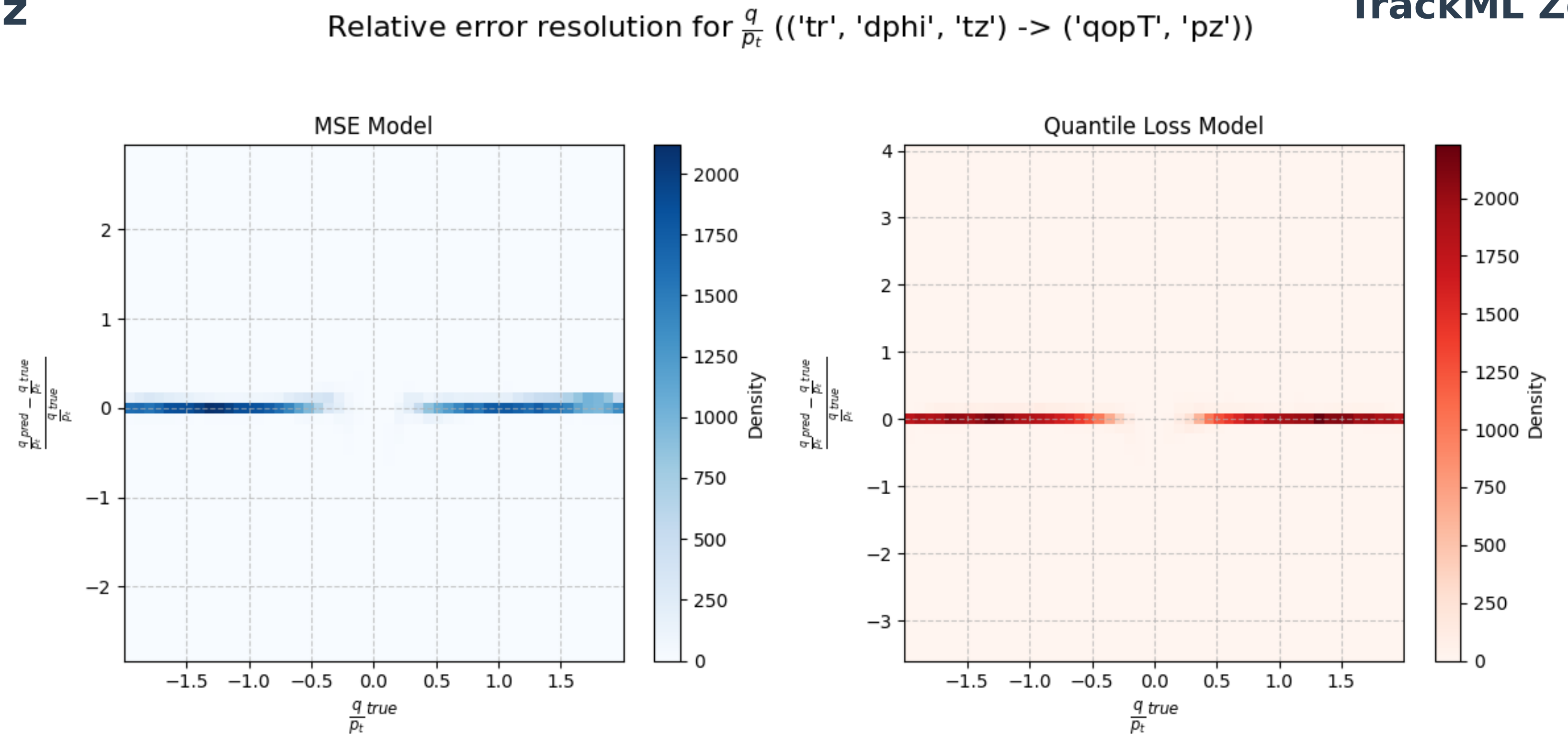
TrackML Zenodo



Resolution

r dphi z

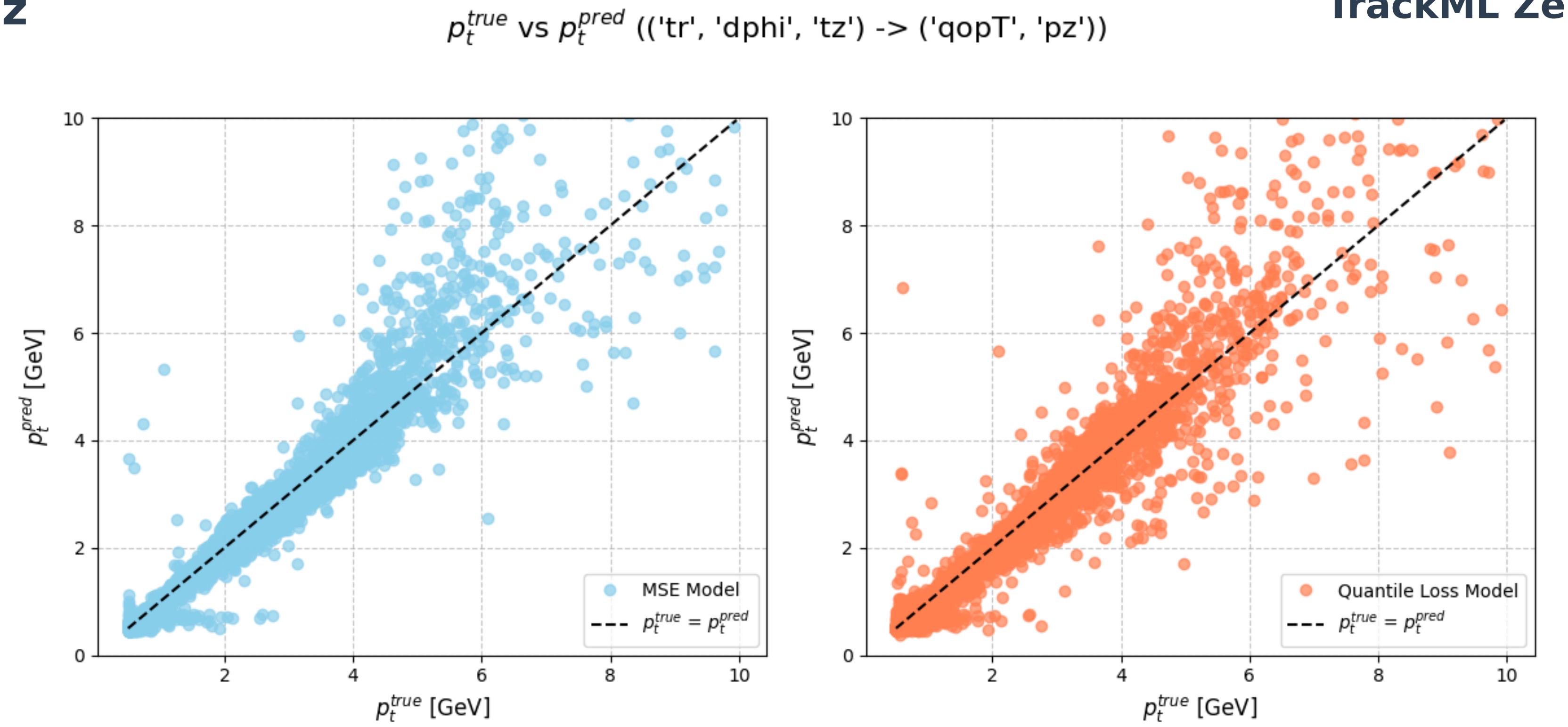
TrackML Zenodo



More results

r dphi z

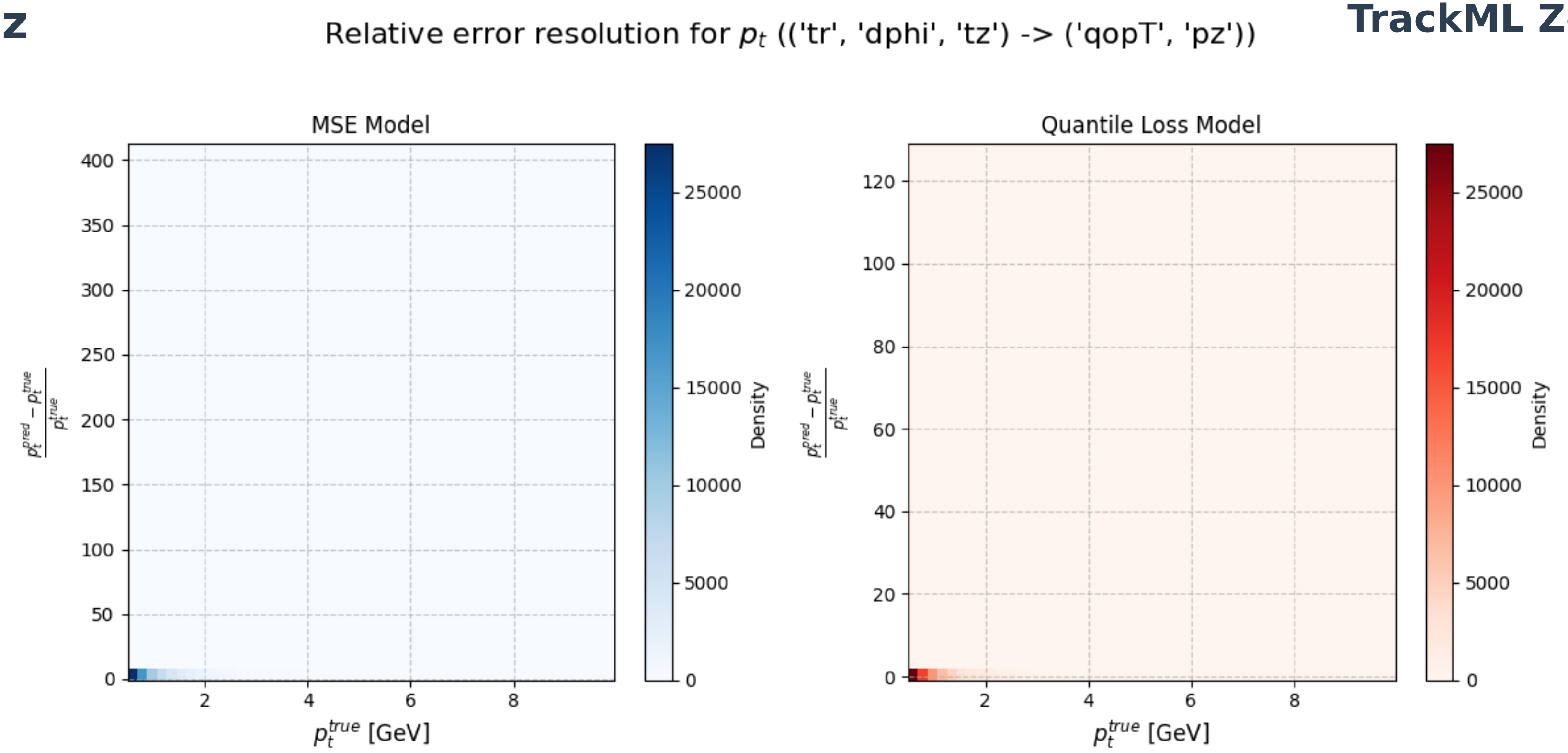
TrackML Zenodo



Resolution

r dphi z

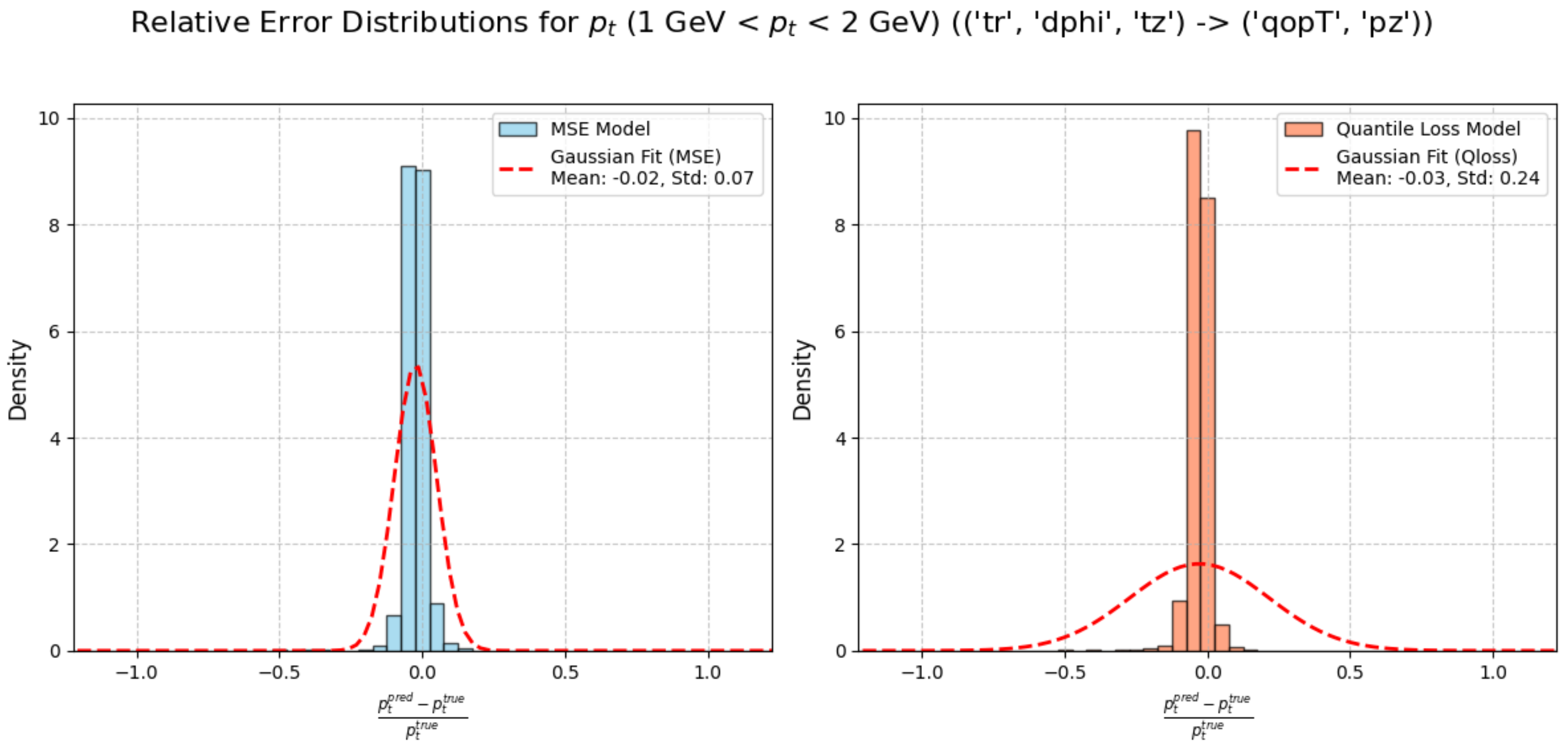
TrackML Zenodo



Resolution

r dphi z

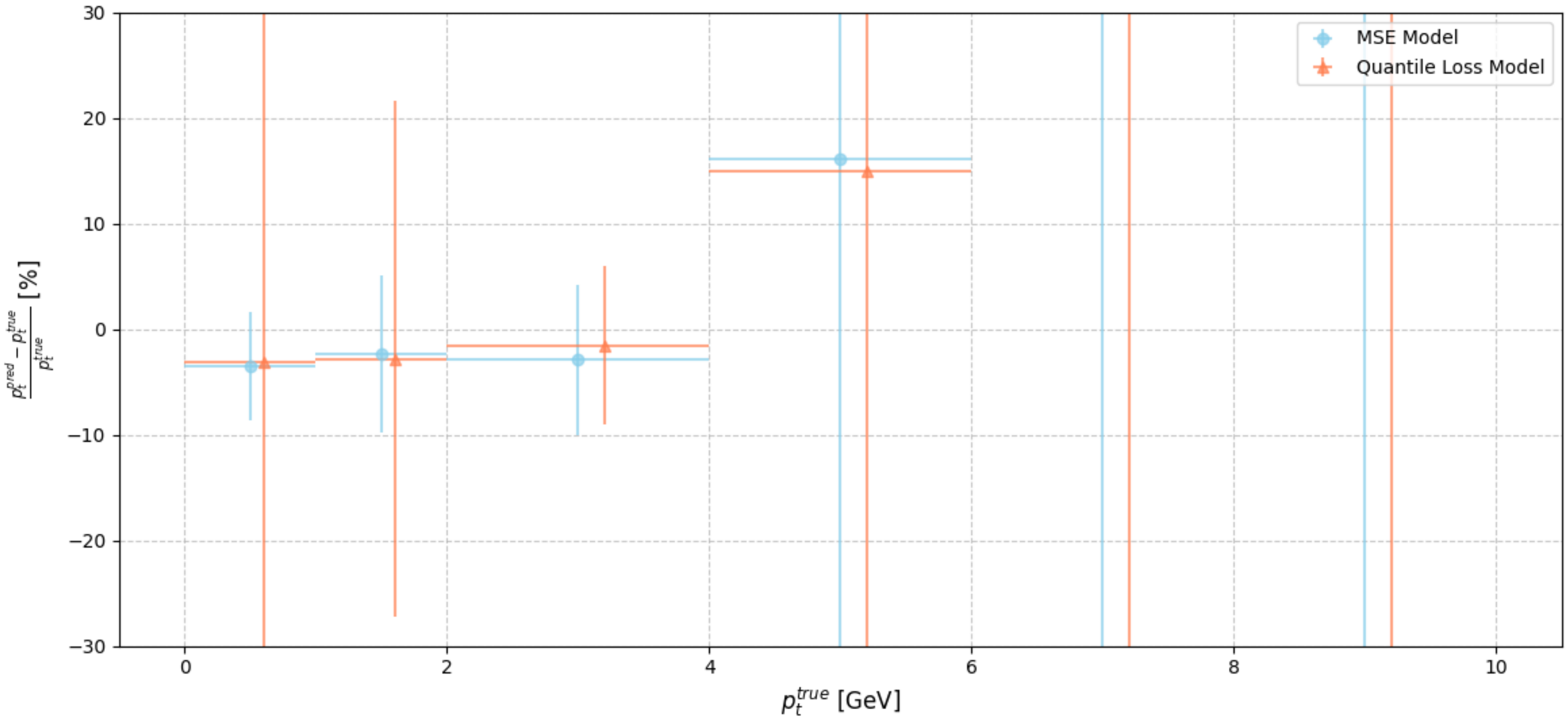
TrackML Zenodo



Resolution

r dphi z

TrackML Zenodo

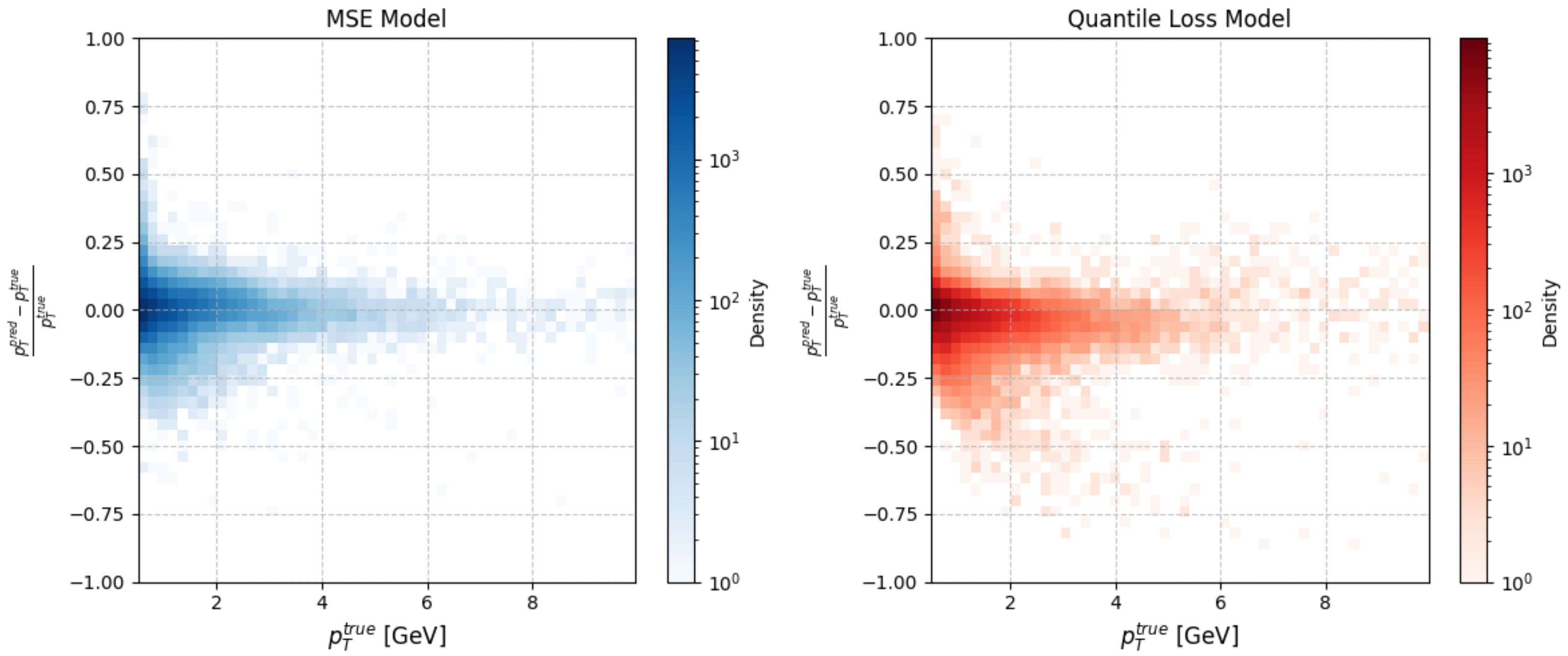


More results

x y z

TrackML Zenodo

Relative error resolution for p_T $((tx, ty, tz) \rightarrow (p_T, p_z))$



Computing resolution

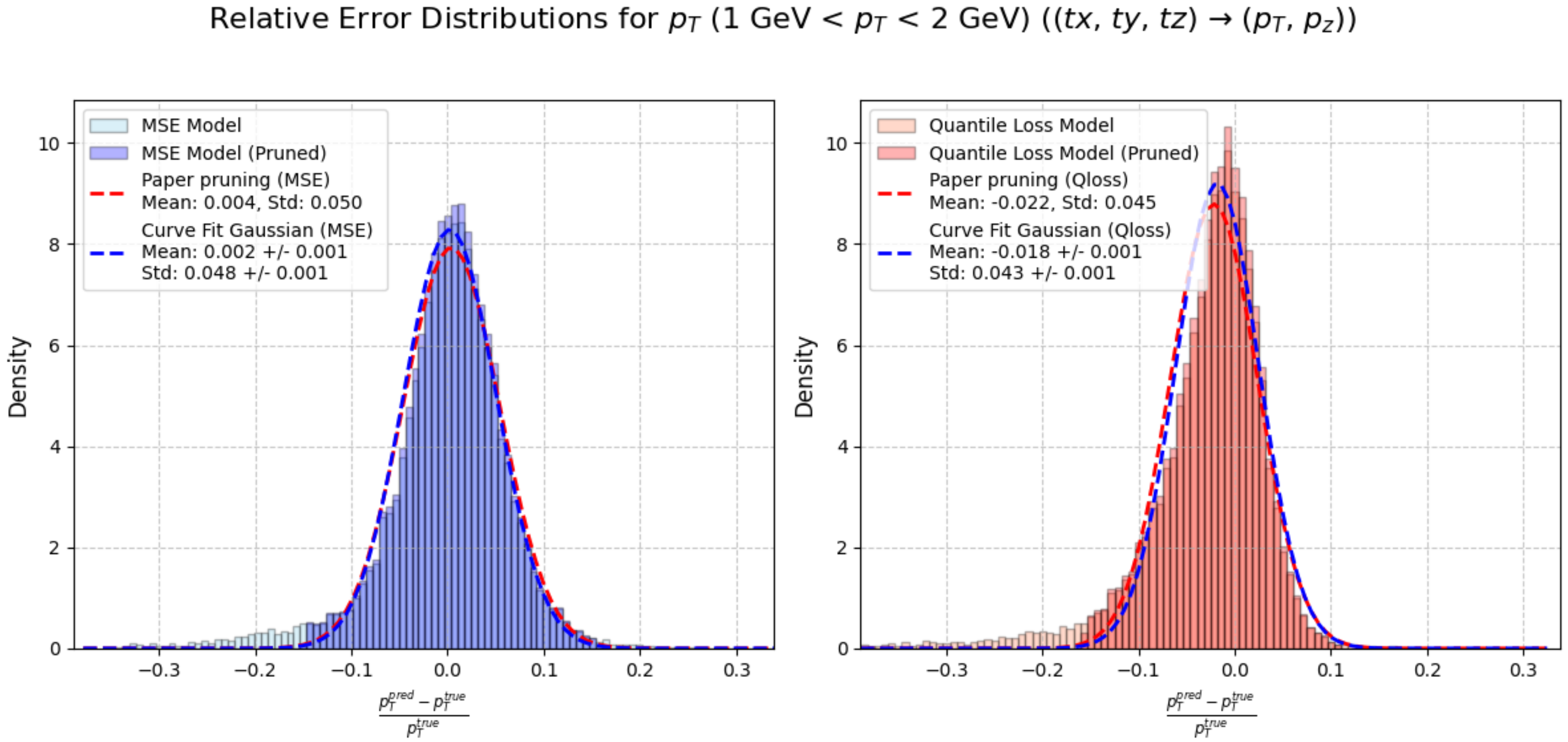
3 approaches:

- **Paper approach:**
  - Iterative pruning of distribution pred-truth from points away from the mean by more than 3 rms
- **Quantile approaches:**
  - Take quantiles equivalent to 5 sigma (of a normal distribution) from the median:
    99.99994266968912% quantile
    5.733031088084317e-05% quantile
  - Estimate mean and std with MLE (scipy norm.fit)
  - Use curve_fit or ROOT to fit a gaussian

Resolution

x y z

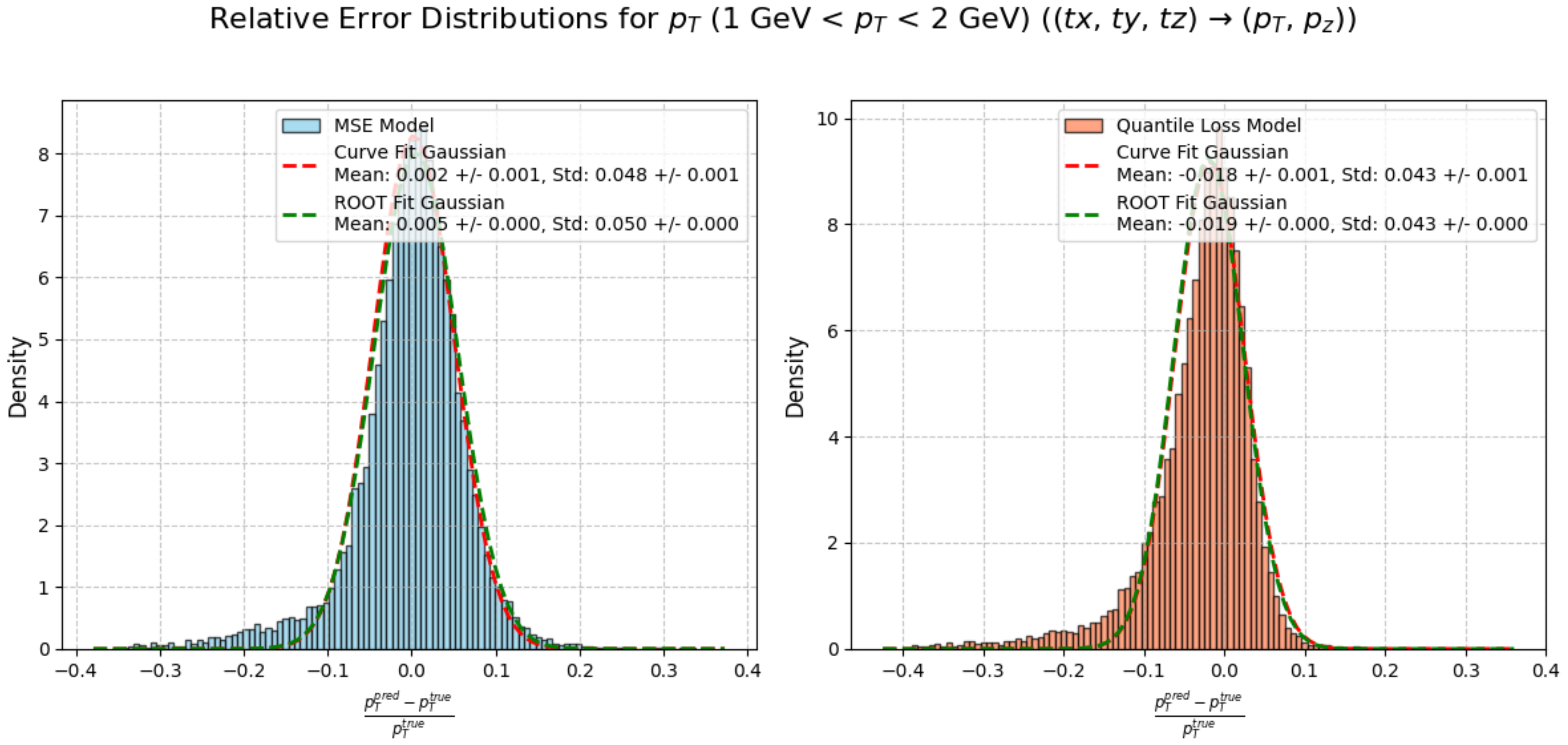
TrackML Zenodo



Resolution

x y z

TrackML Zenodo

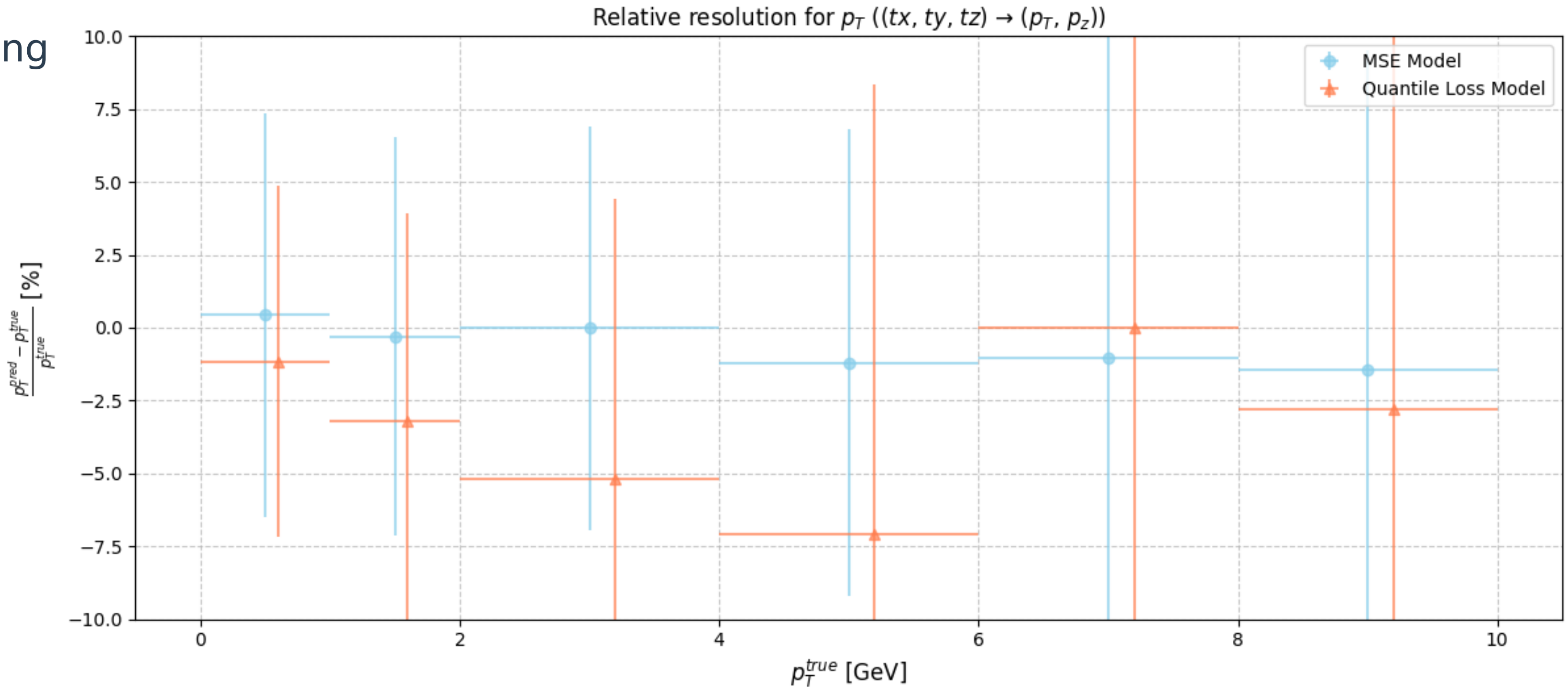


Resolution

x y z

TrackML Zenodo

No pruning



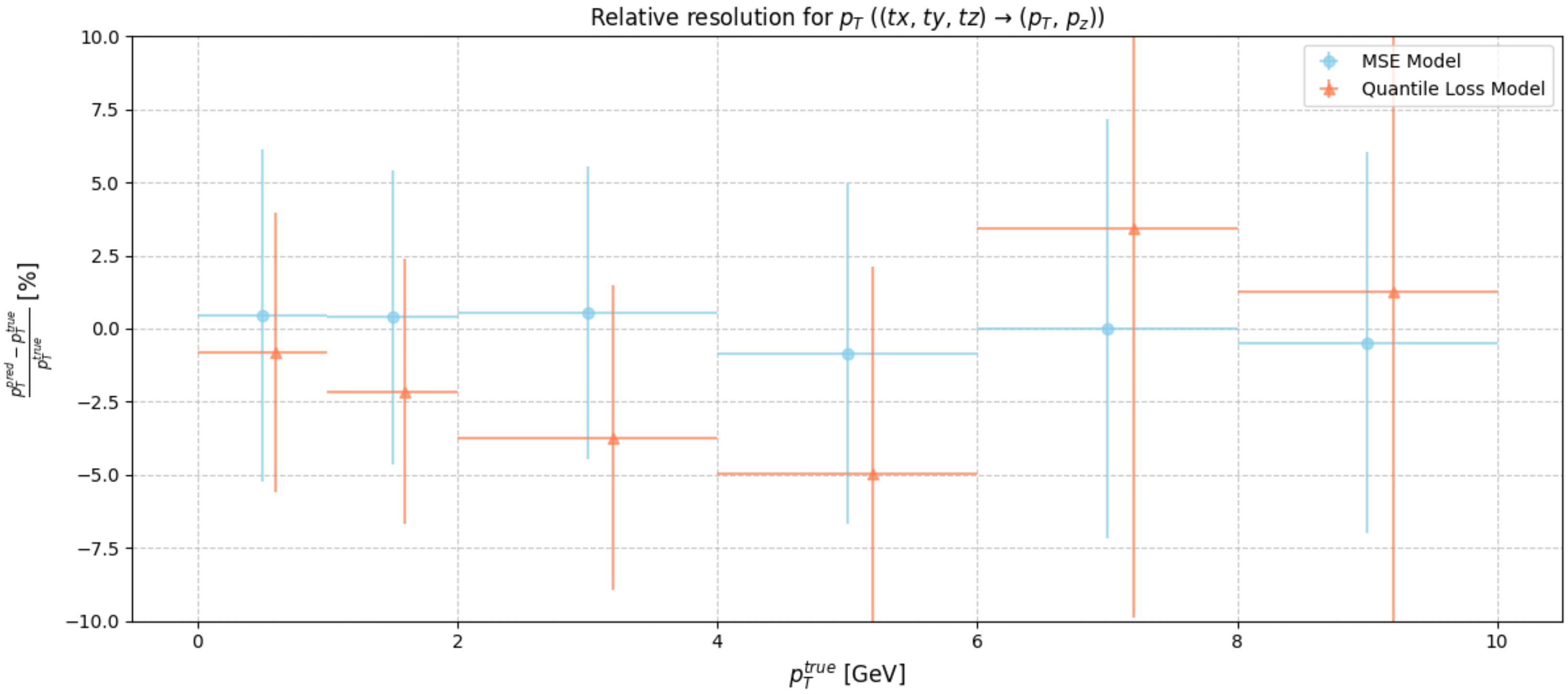
Quantile loss is worst

Resolution

x y z

TrackML Zenodo

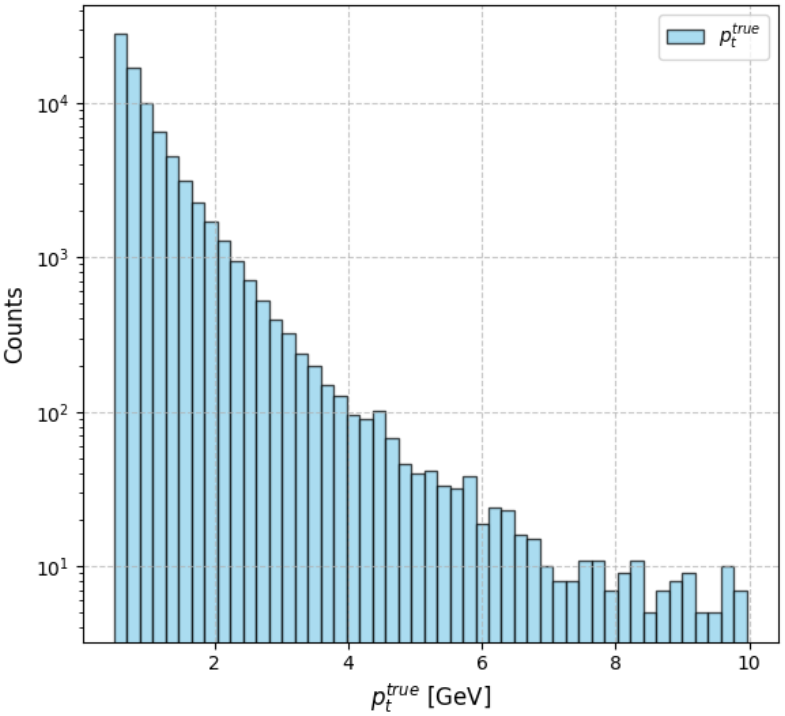
Pruning

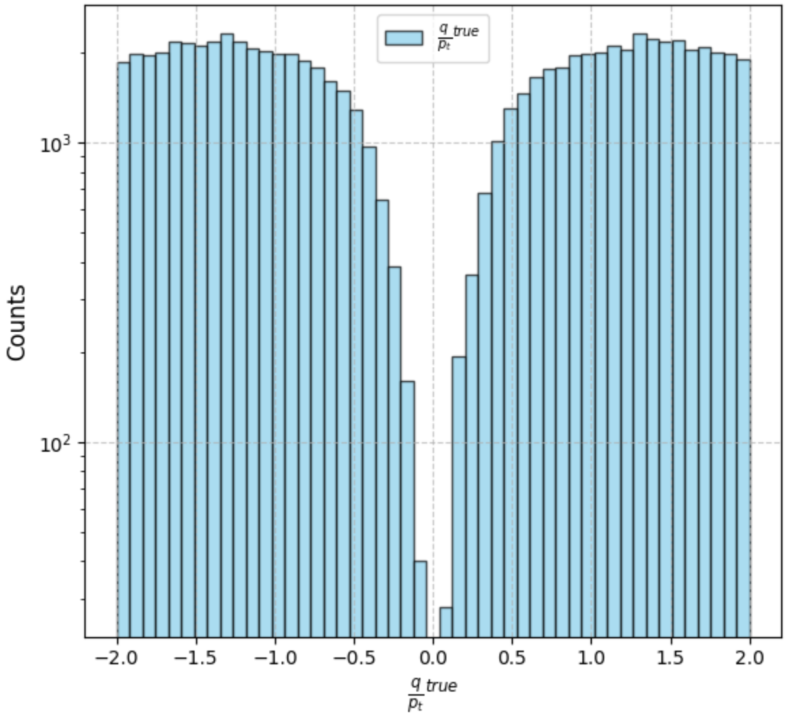


Pruning improves results

q/pT

Target variable:



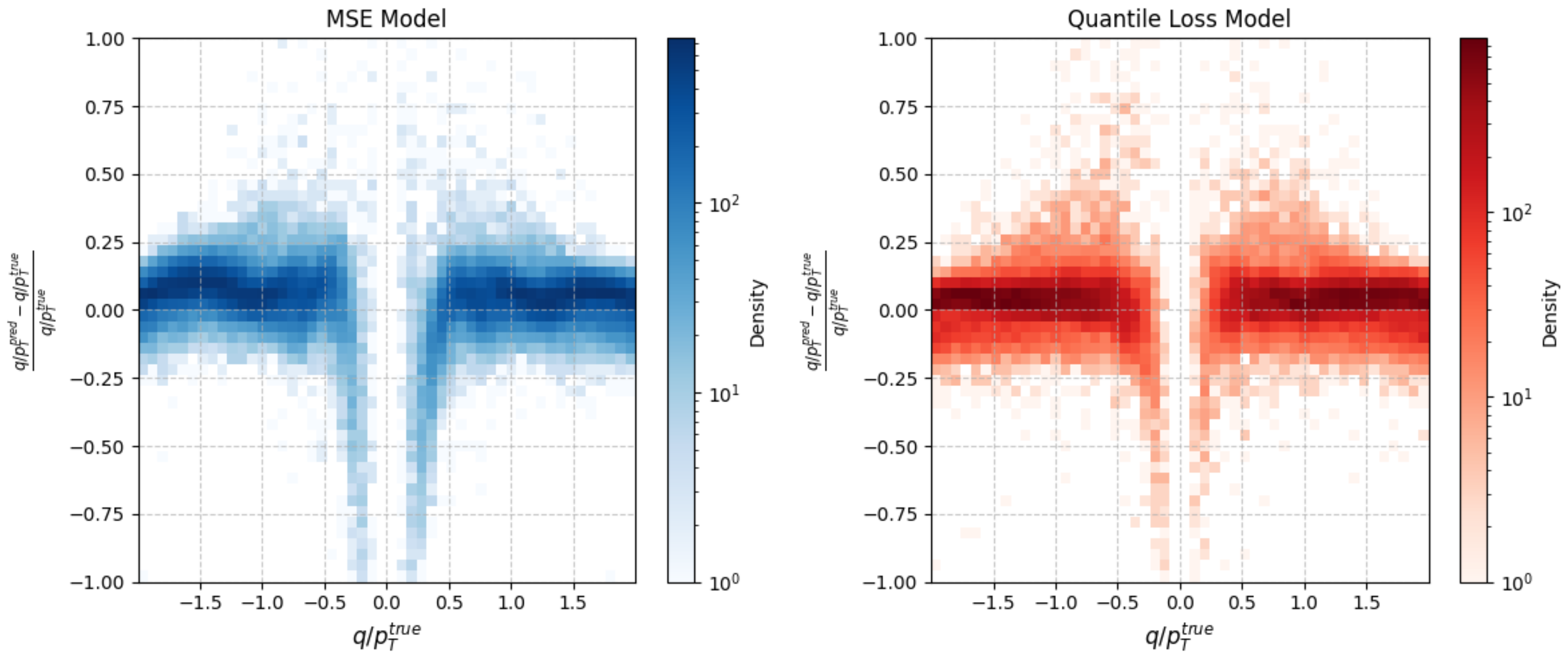


Resolution

x y z

TrackML Zenodo

Relative error resolution for q/p_T $((tx, ty, tz) \rightarrow (q/p_T, p_z))$

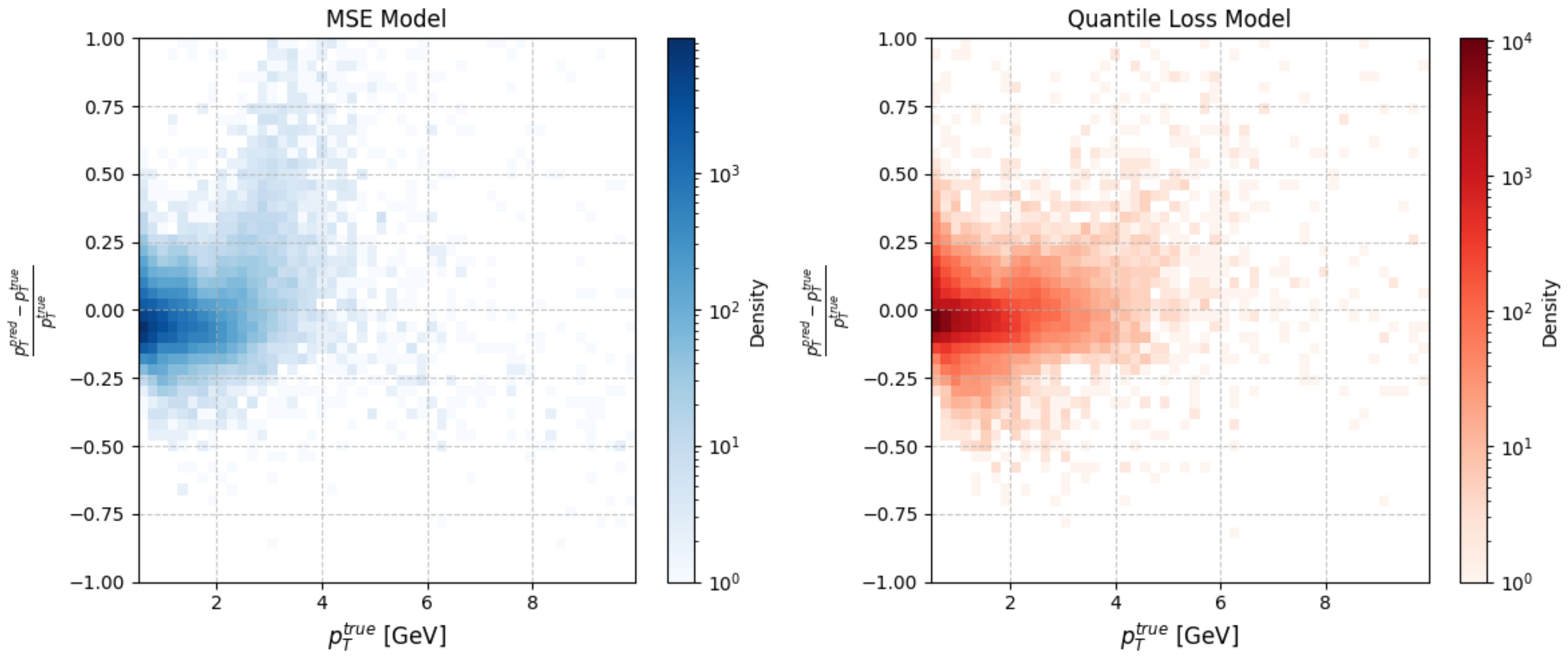


Resolution

x y z

TrackML Zenodo

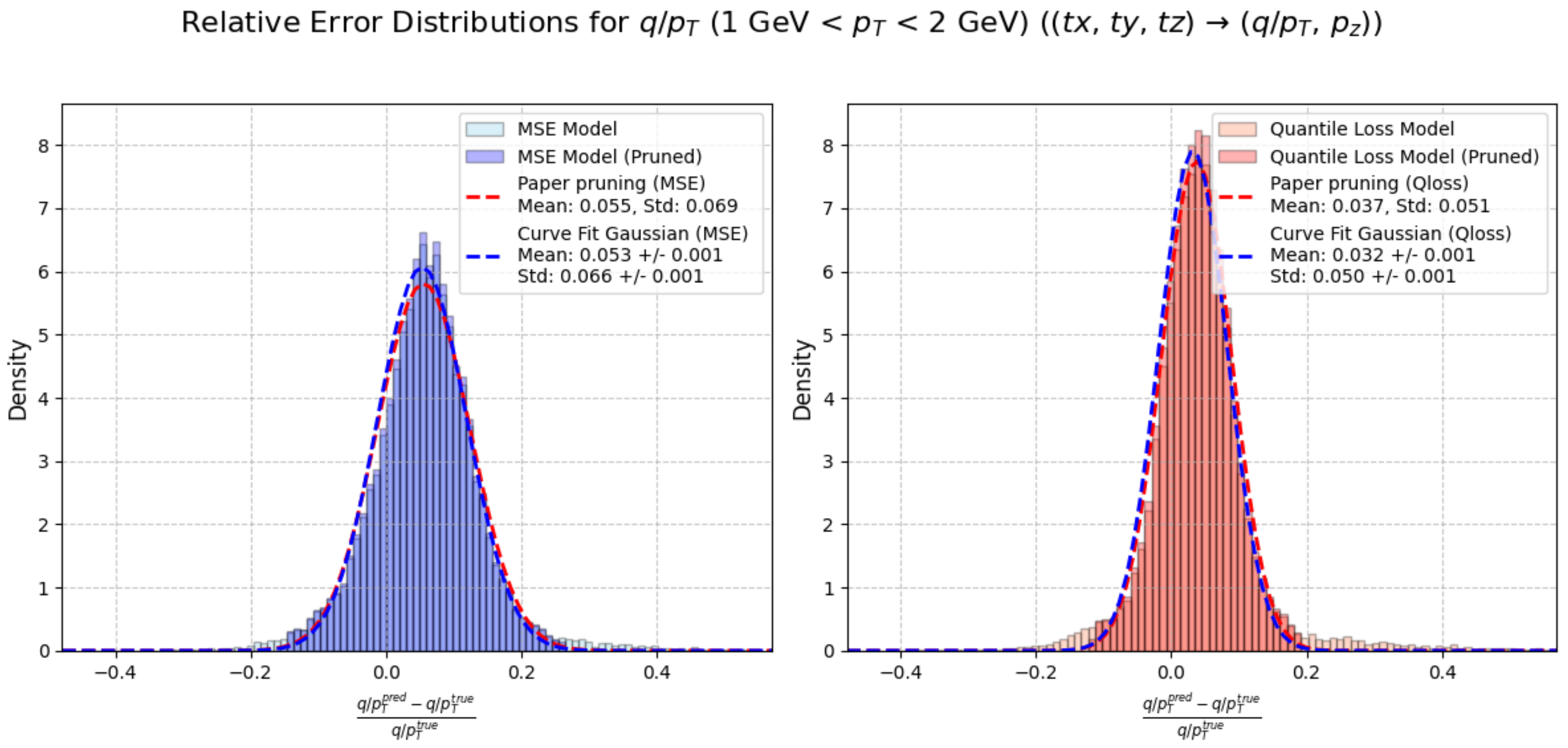
Relative error resolution for p_T $((tx, ty, tz) \rightarrow (q/p_T, p_z))$



Resolution

x y z

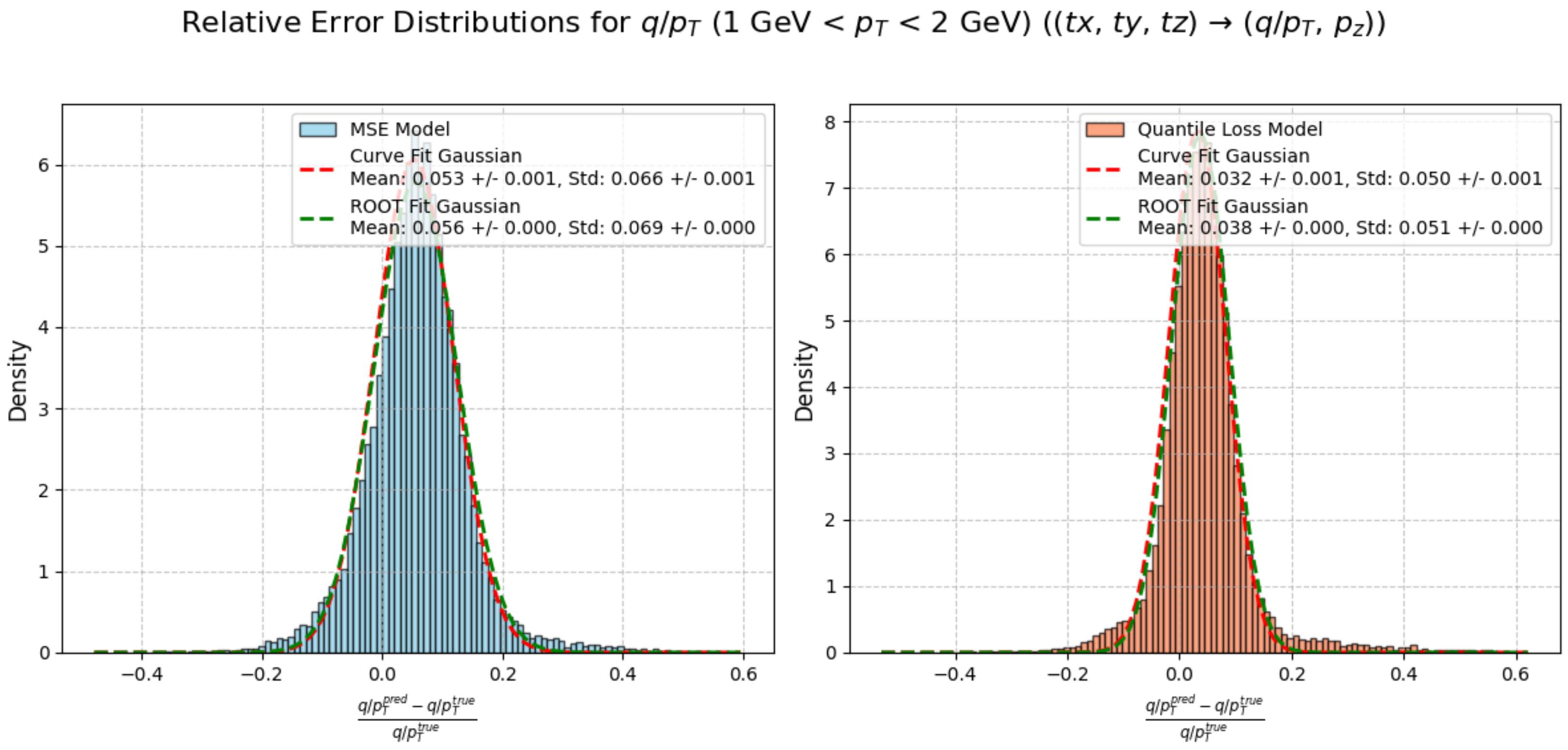
TrackML Zenodo



Resolution

x y z

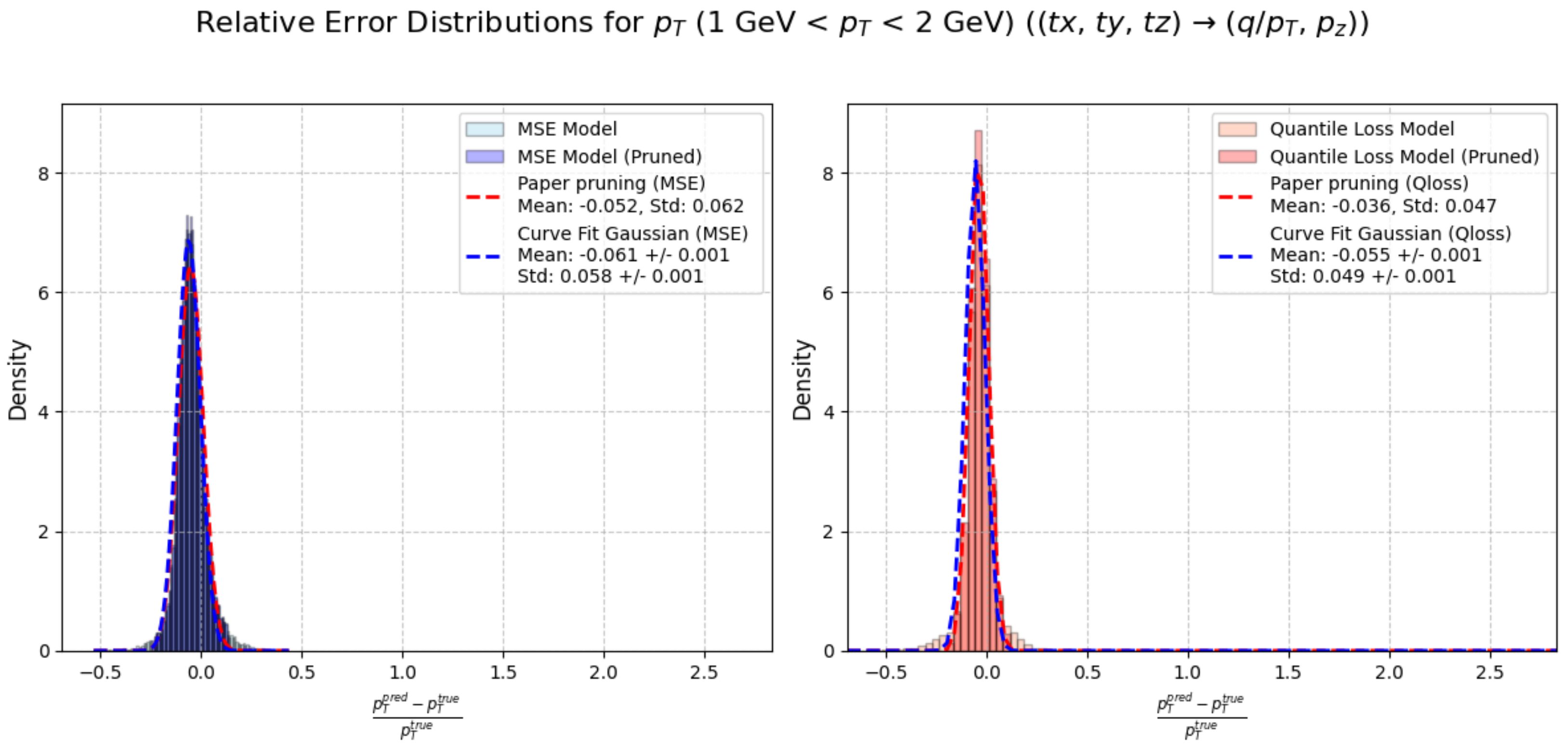
TrackML Zenodo



Resolution

x y z

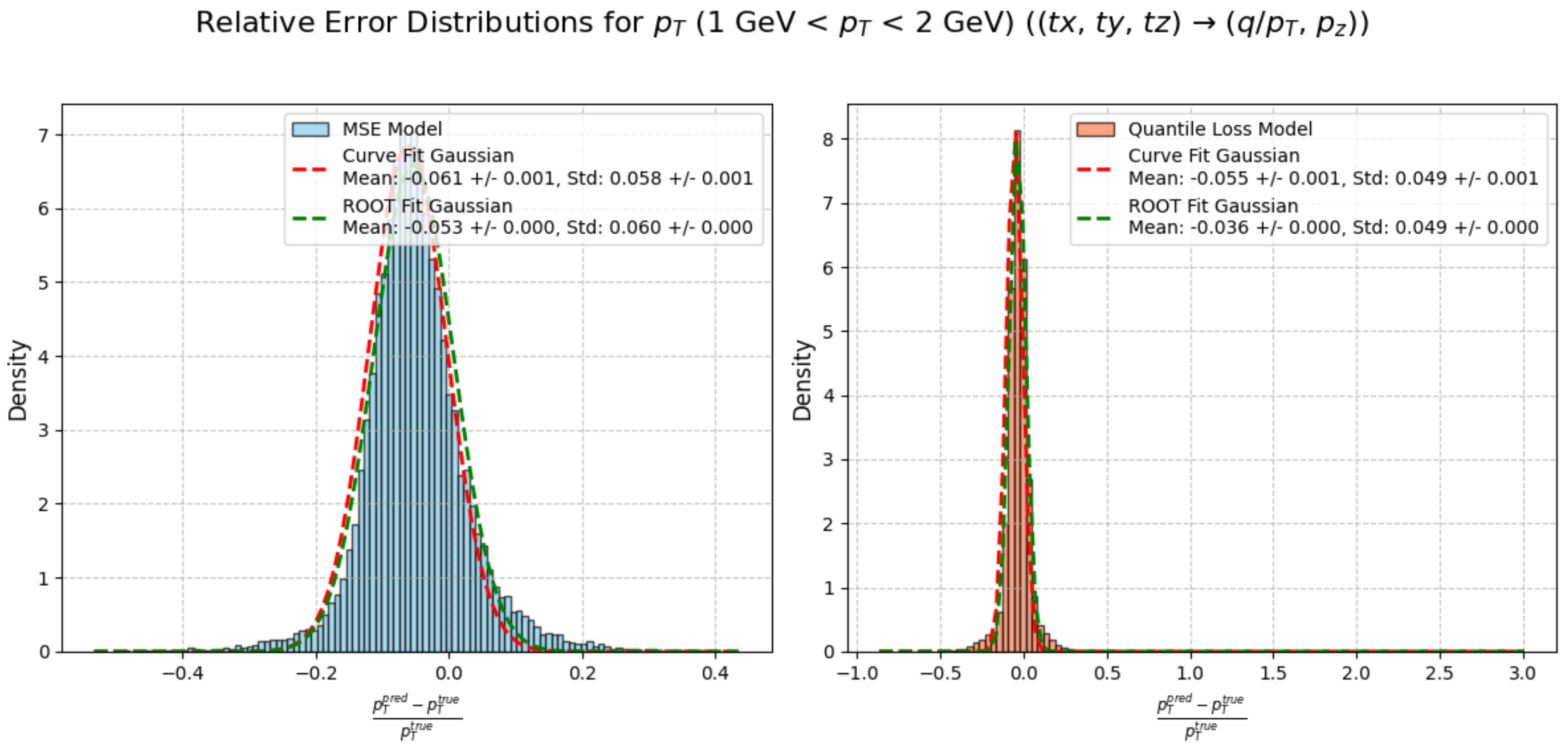
TrackML Zenodo



Resolution

x y z

TrackML Zenodo

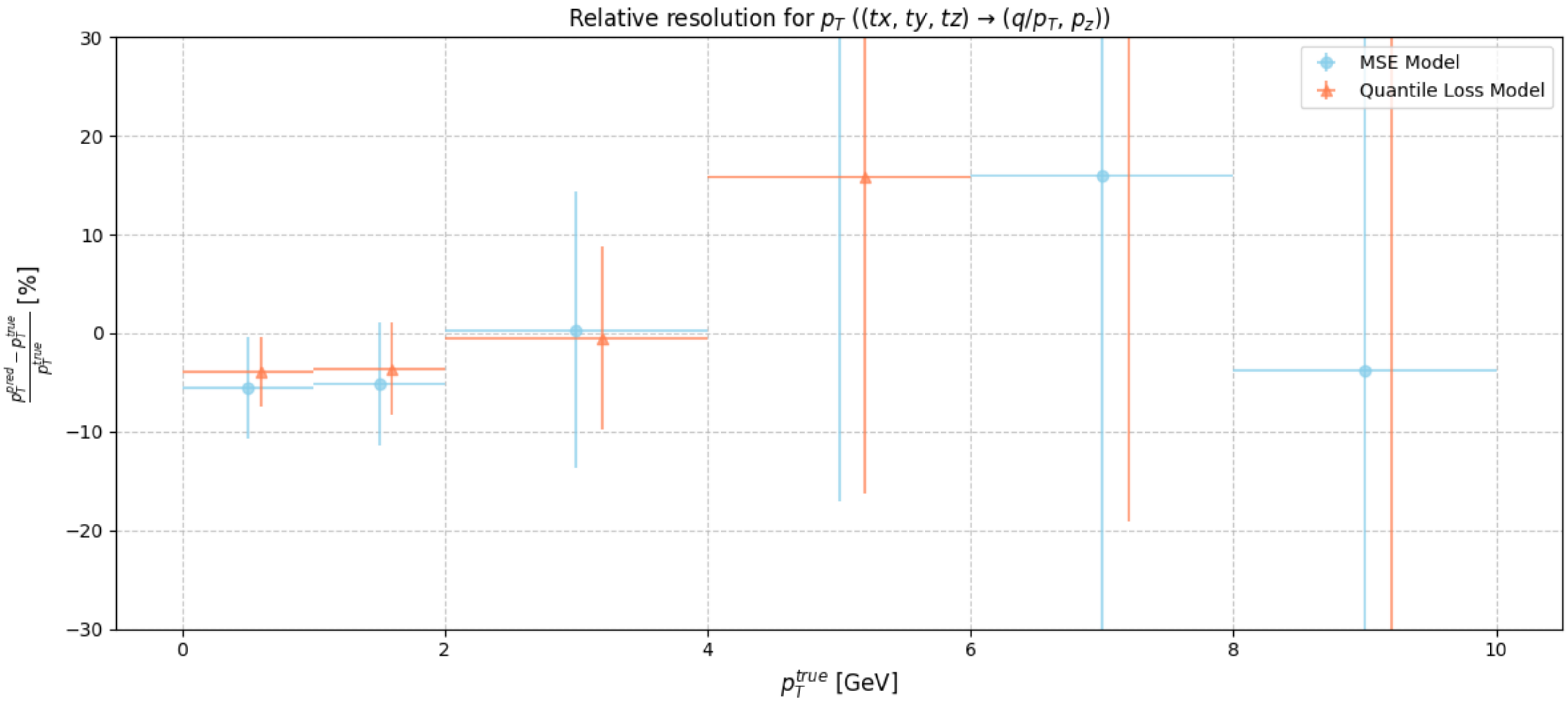


Resolution

x y z

TrackML Zenodo

Pruning



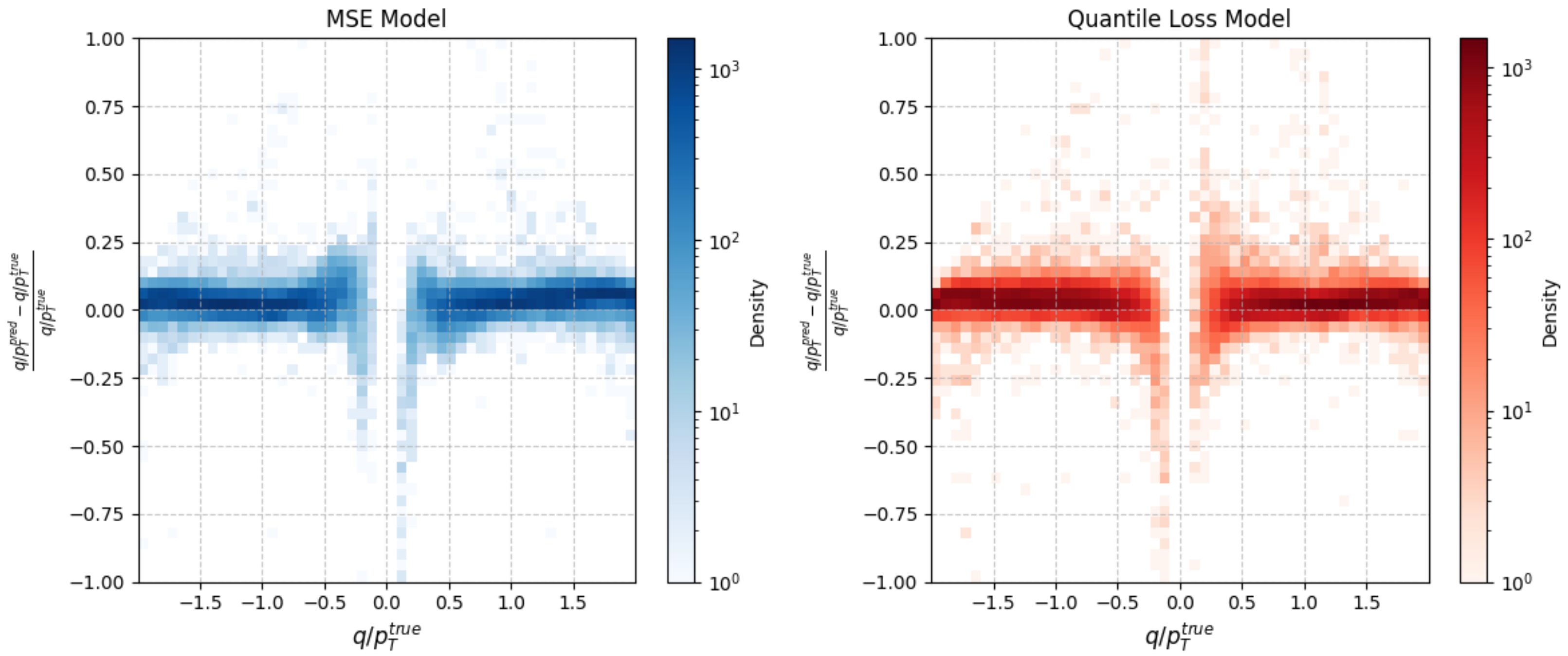
r, dphi, z → q/pT, pz

Resolution

r dphi z

TrackML Zenodo

Relative error resolution for q/p_T ((tr, $d\varphi$, tz) → (q/p_T, p_z))

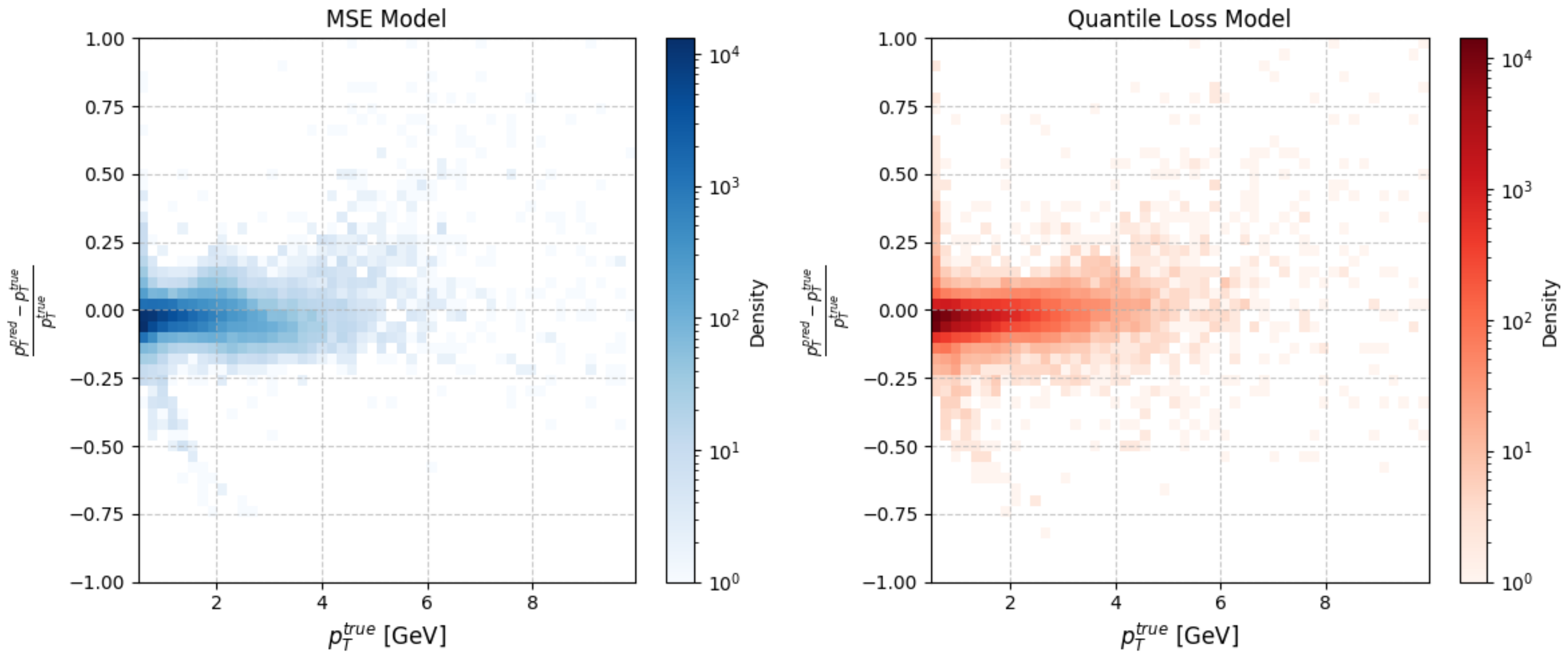


Resolution

r dphi z

TrackML Zenodo

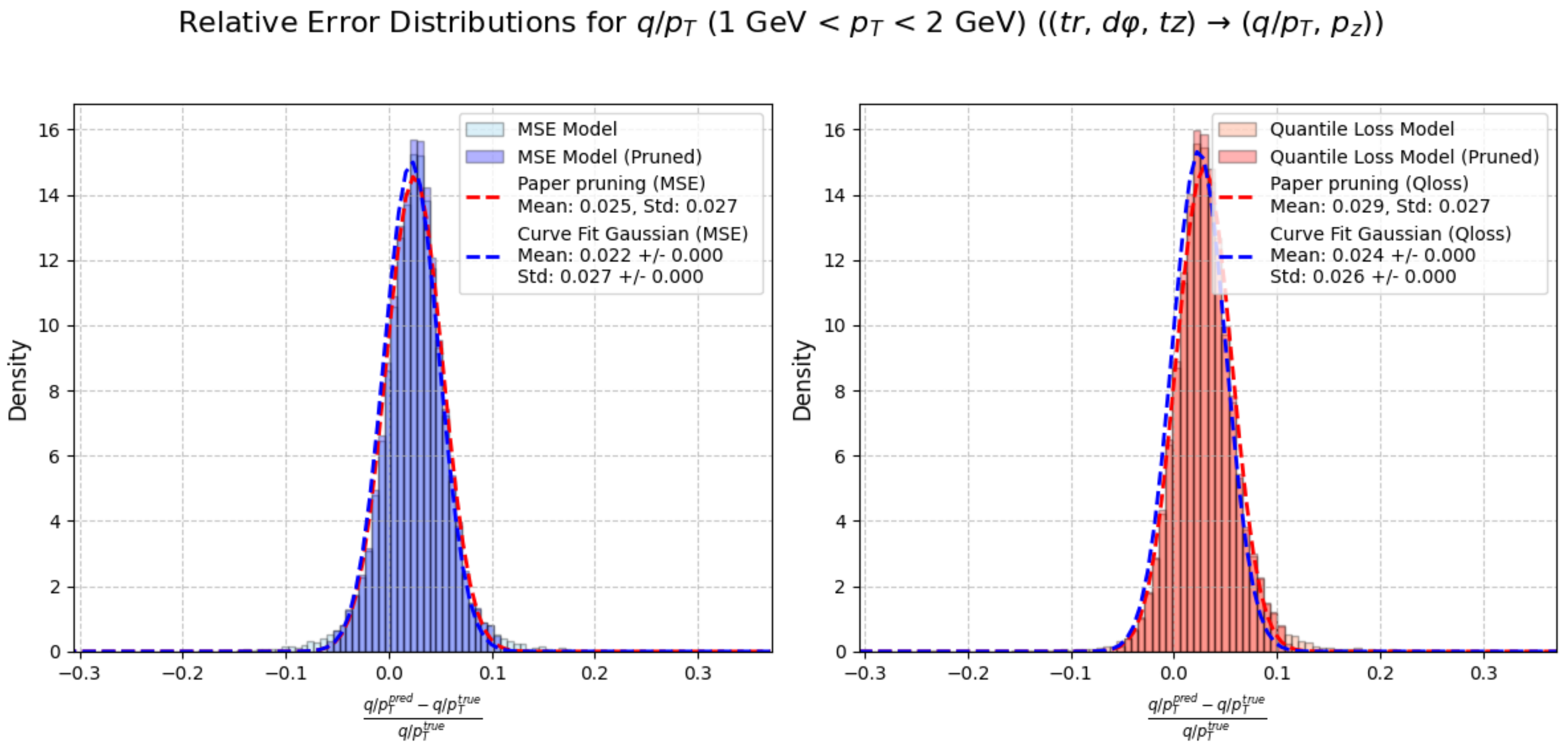
Relative error resolution for p_T ($(tr, d\varphi, tz) \rightarrow (q/p_T, p_z)$)



Resolution

r dphi z

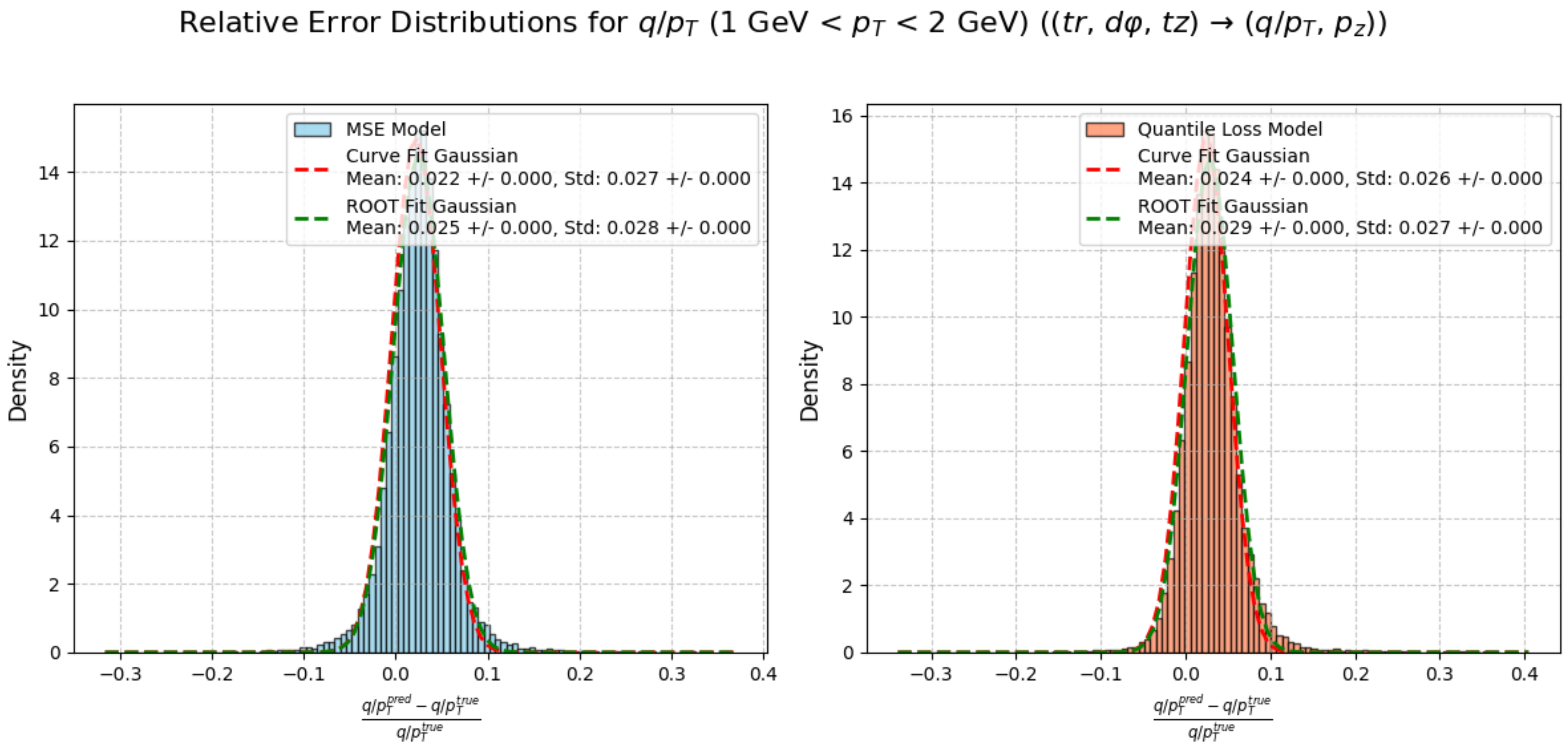
TrackML Zenodo



Resolution

r dphi z

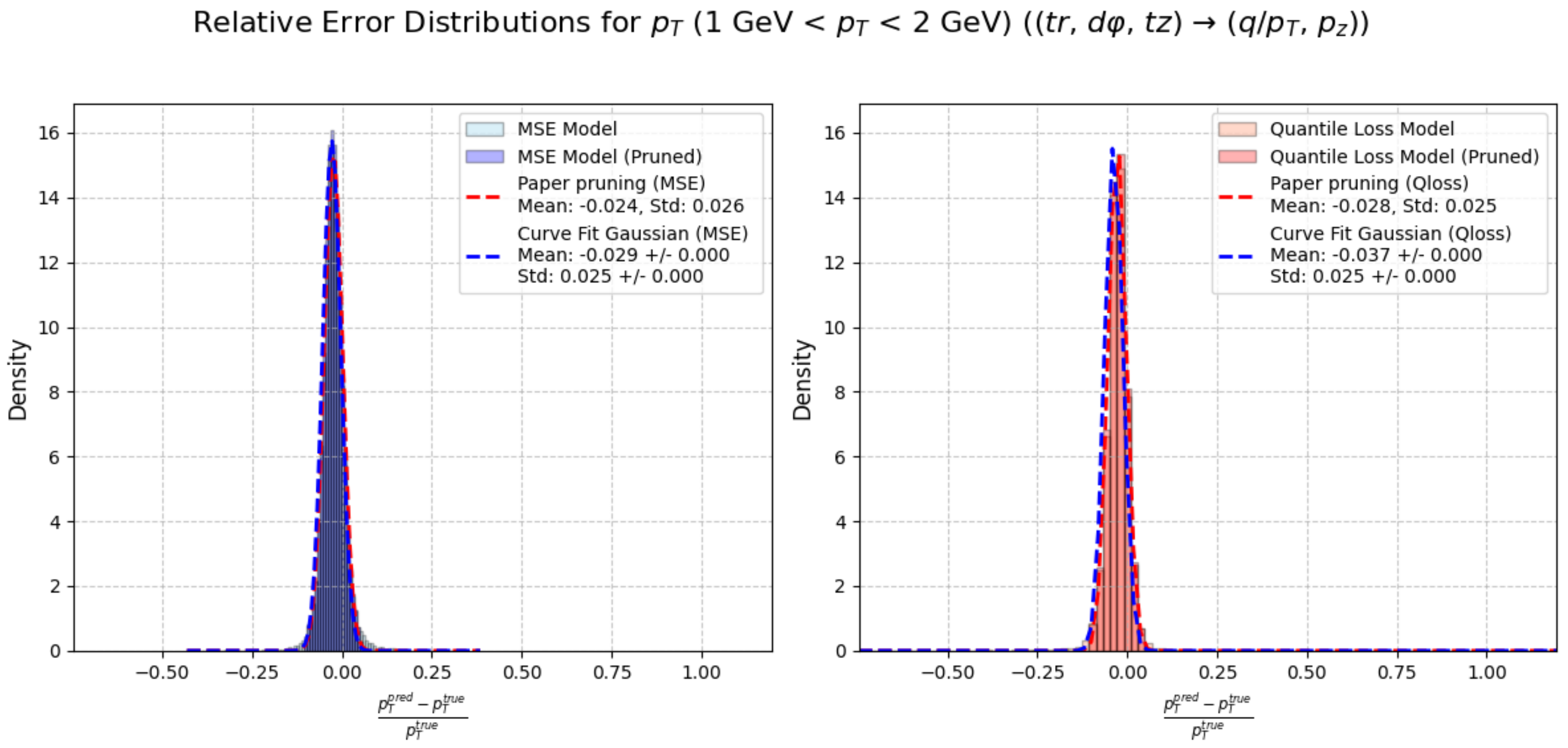
TrackML Zenodo



Resolution

r dphi z

TrackML Zenodo

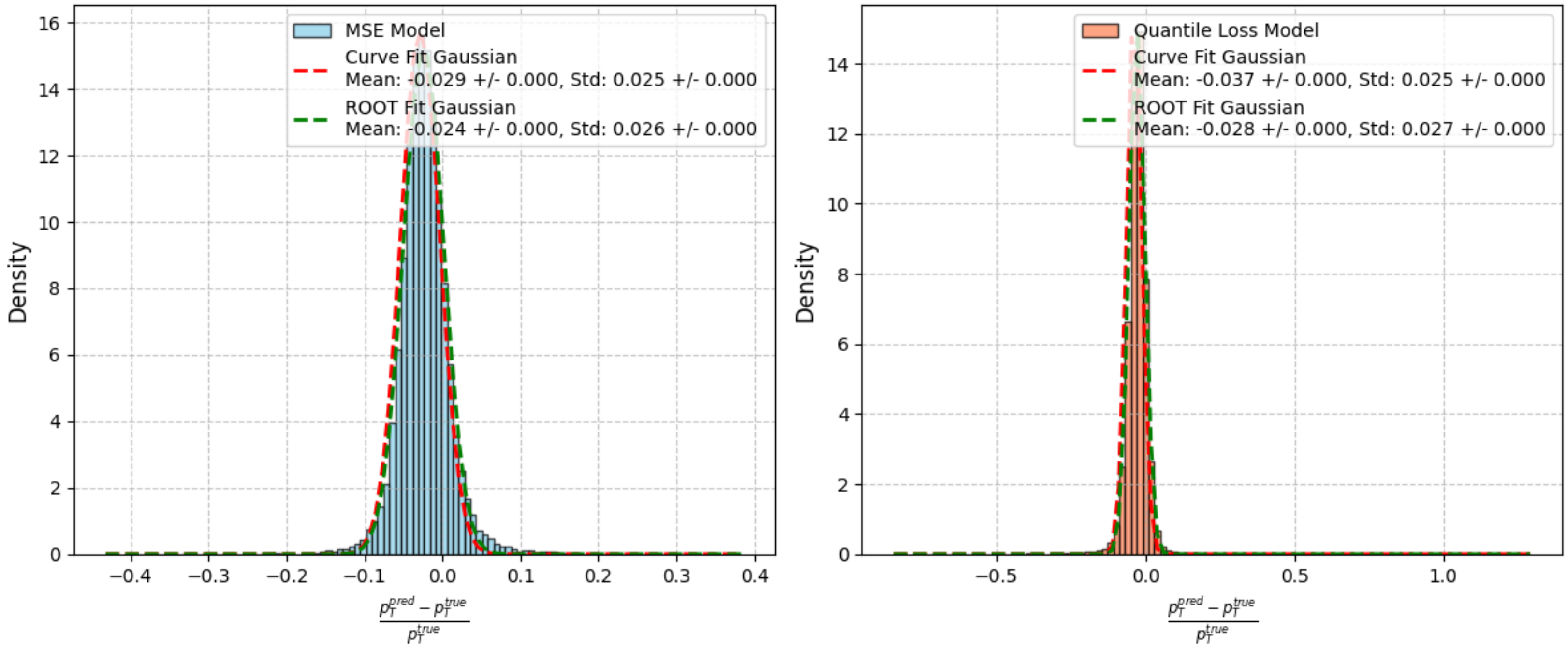


Resolution

r dphi z

TrackML Zenodo

Relative Error Distributions for p_T (1 GeV < p_T < 2 GeV) ((tr, $d\varphi$, tz) → (q/p_T, p_z))

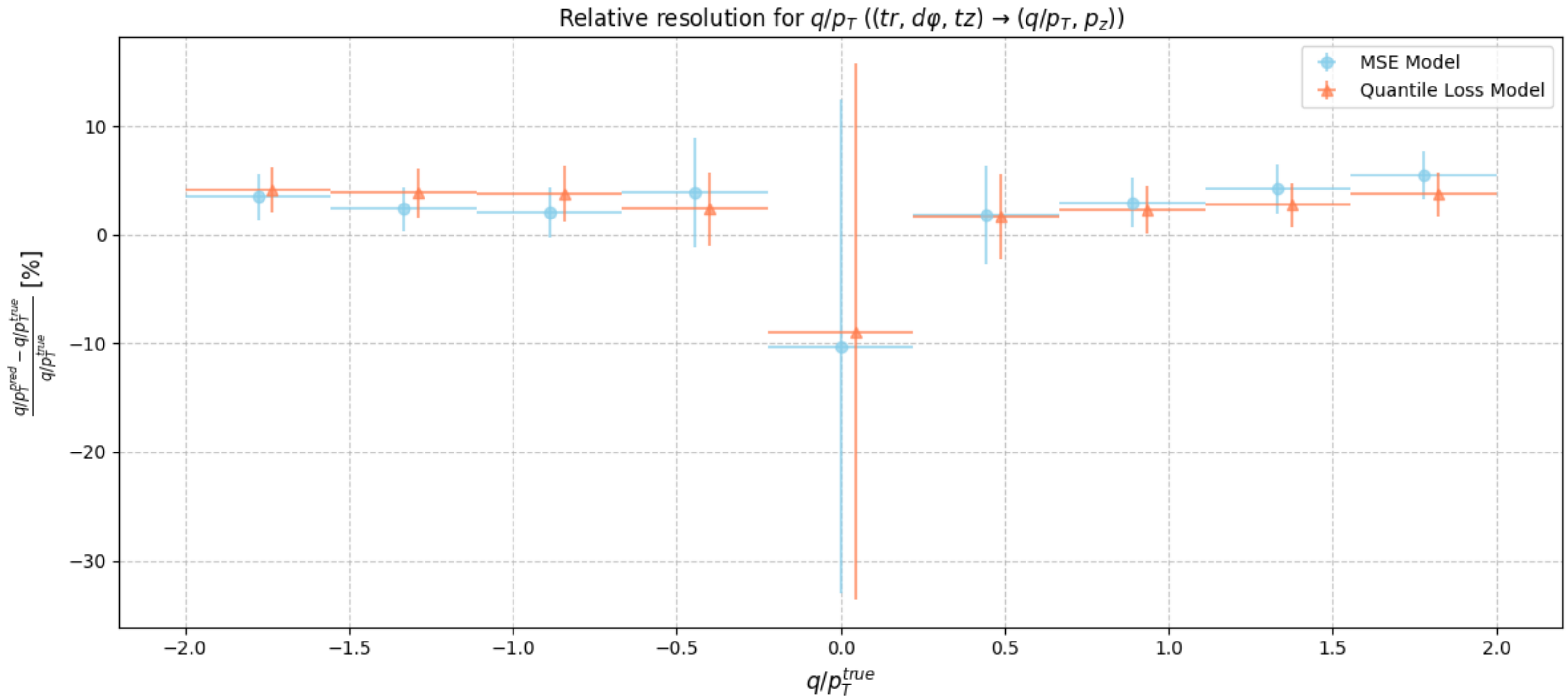


Resolution

r dphi z

TrackML Zenodo

Pruning

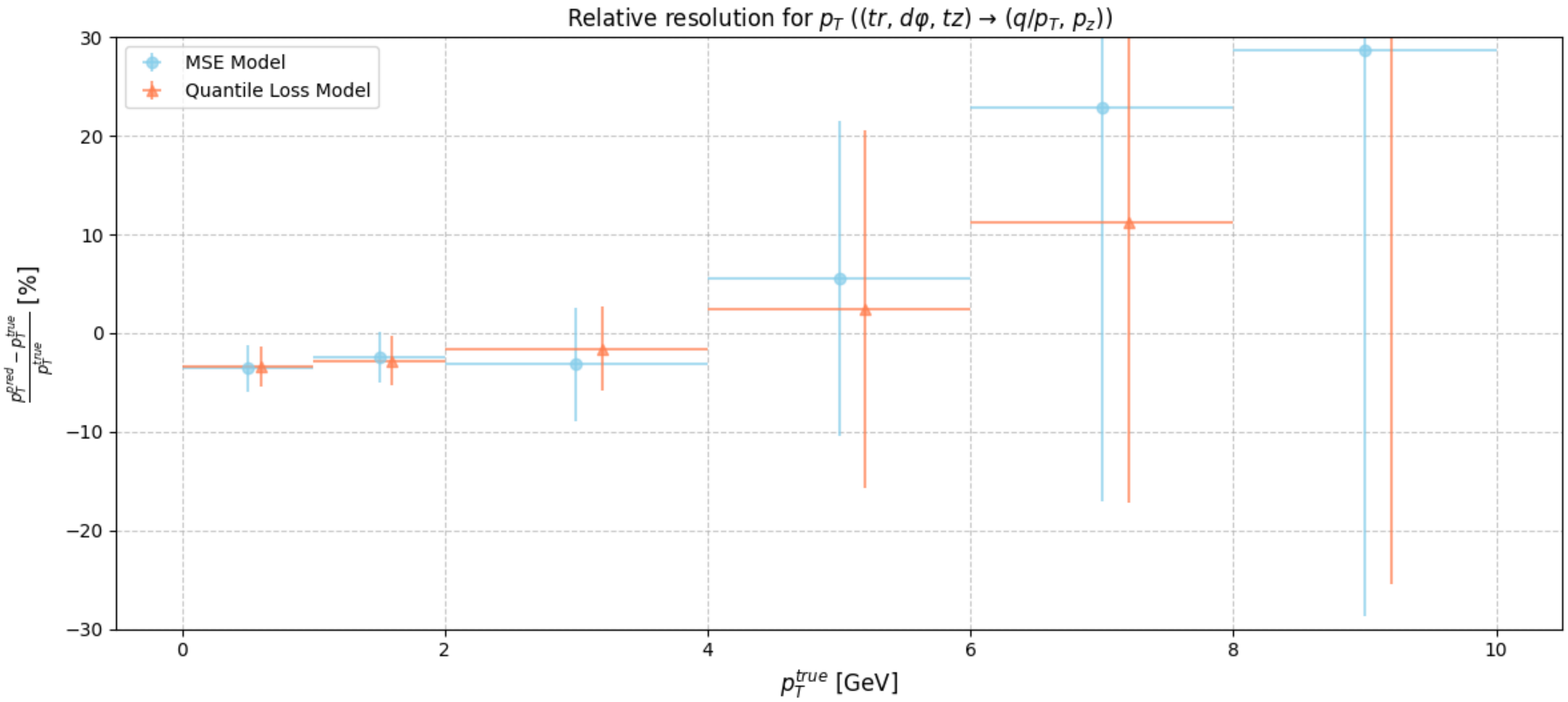


Resolution

r dphi z

TrackML Zenodo

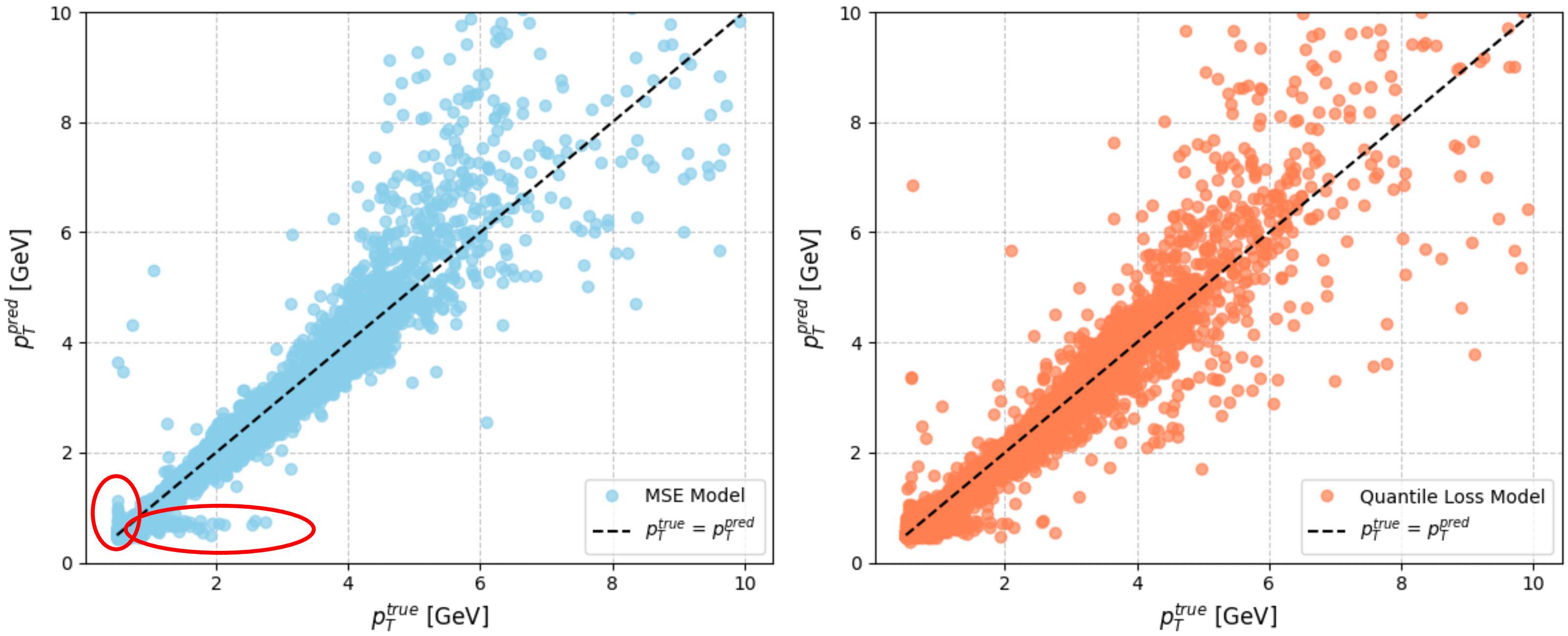
Pruning



Impact on predictions

p_T^{true} vs p_T^{pred} $((tr, d\varphi, tz) \rightarrow (q/p_T, p_z))$

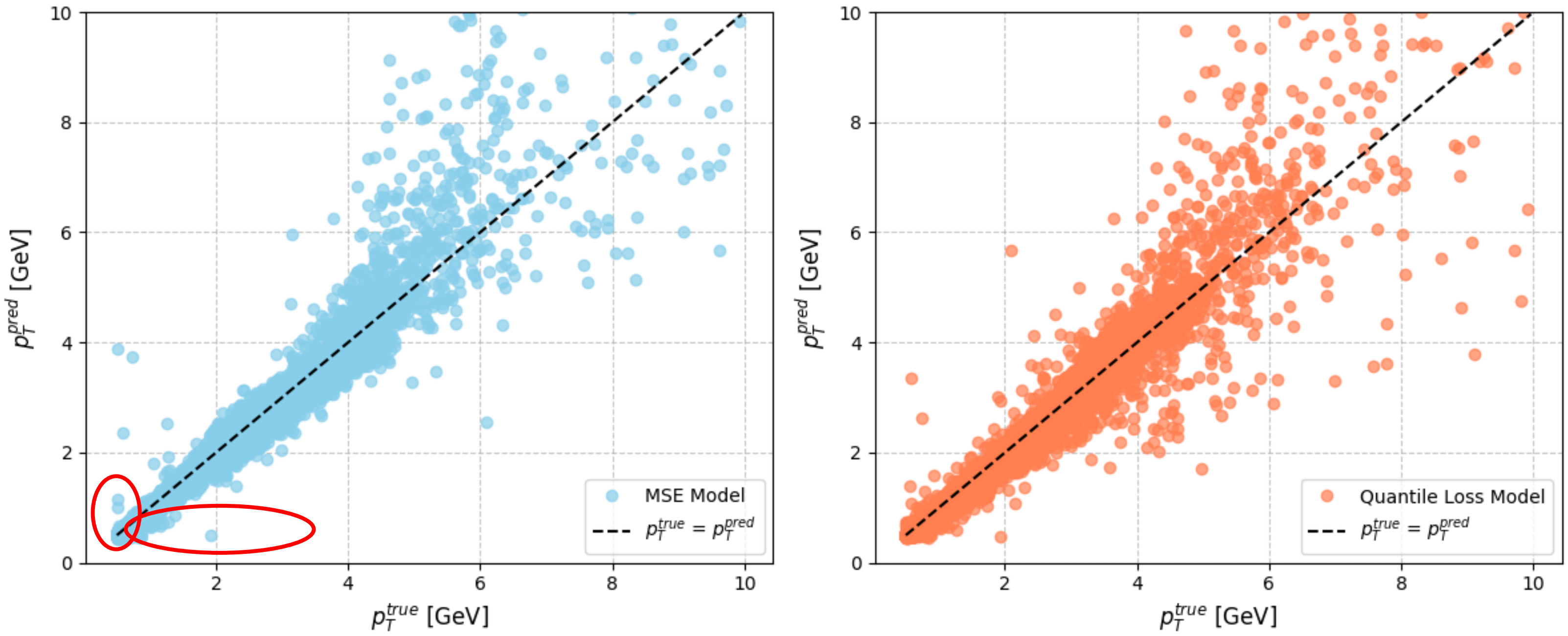
No scattering cut



Impact on predictions

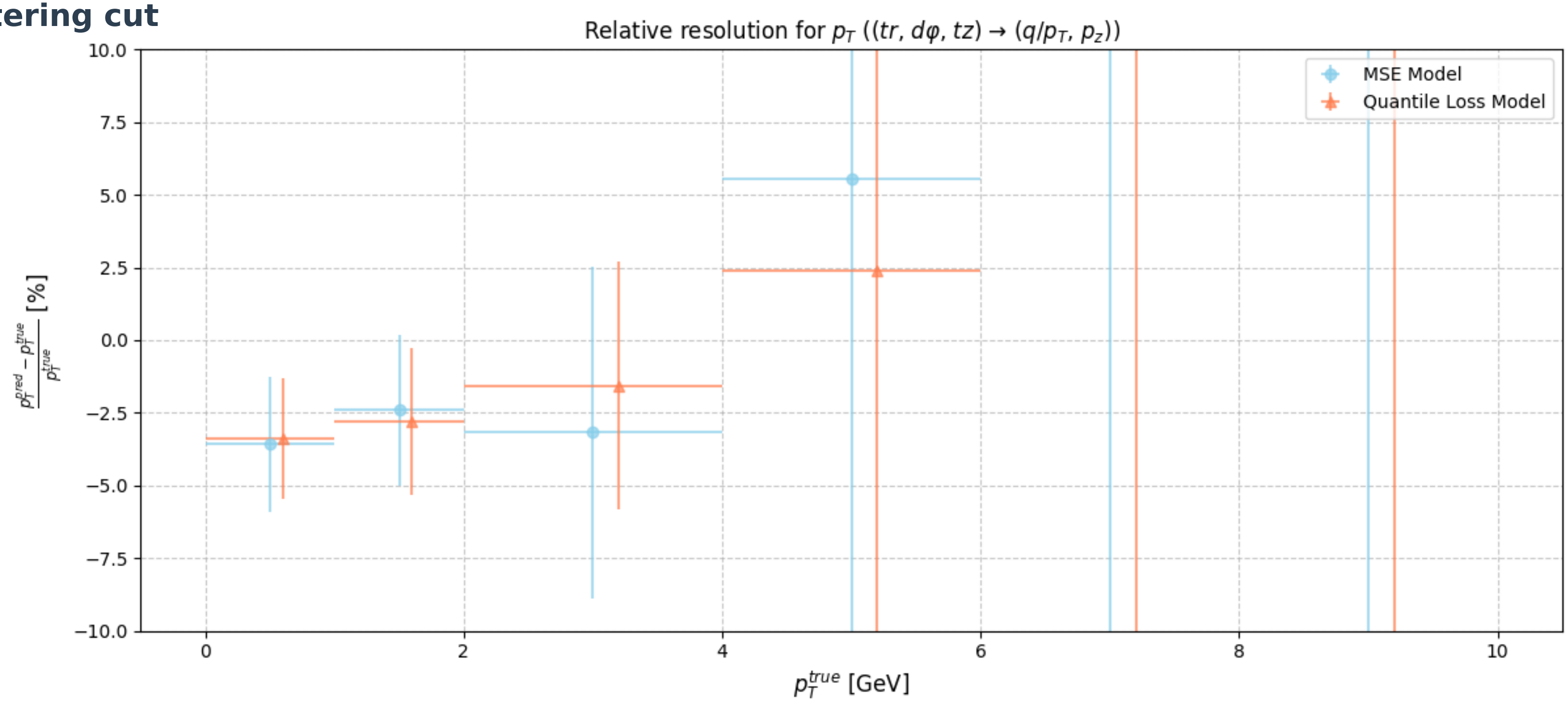
p_T^{true} vs p_T^{pred} $((tr, d\varphi, tz) \rightarrow (q/p_T, p_z))$

Scattering cut (test only)



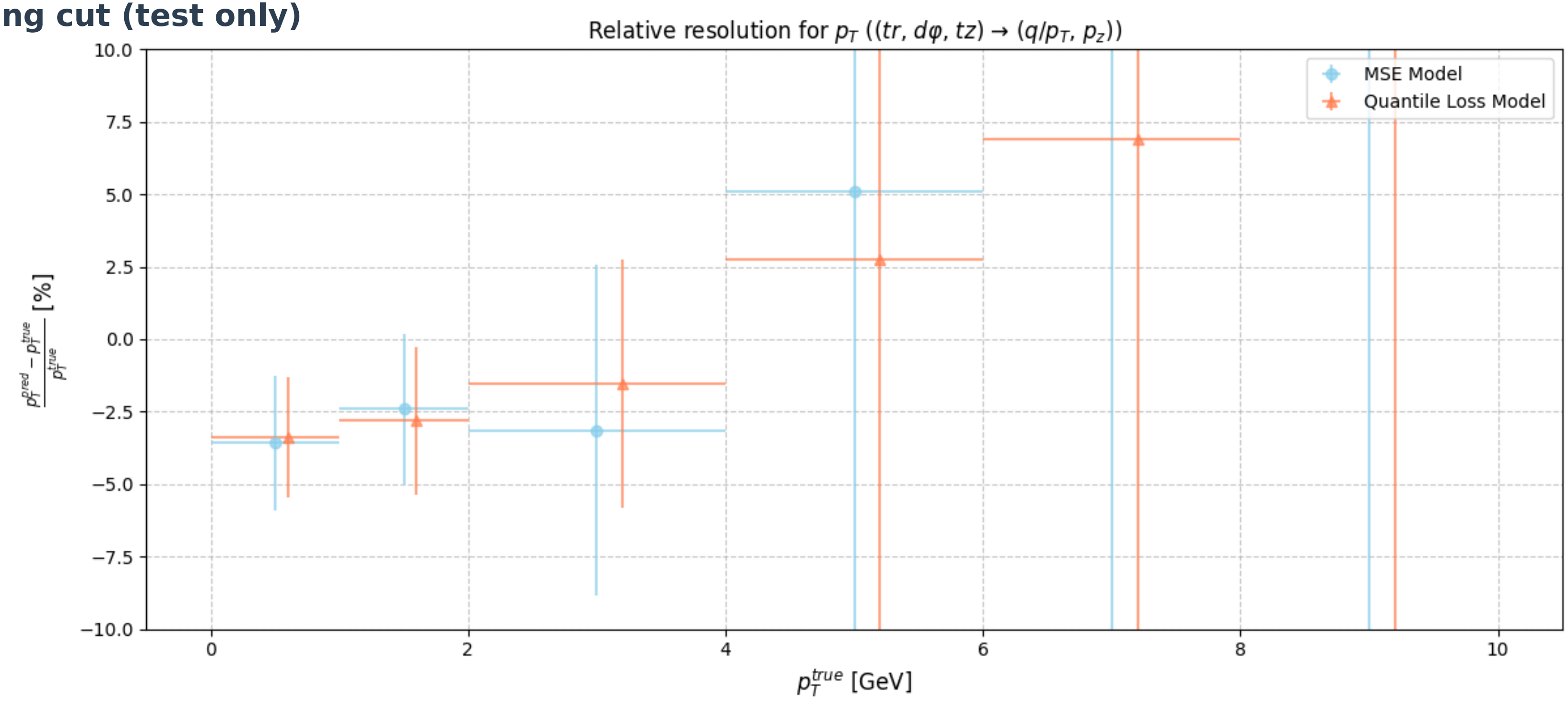
Impact on resolutions

No scattering cut



Impact on resolutions

Scattering cut (test only)

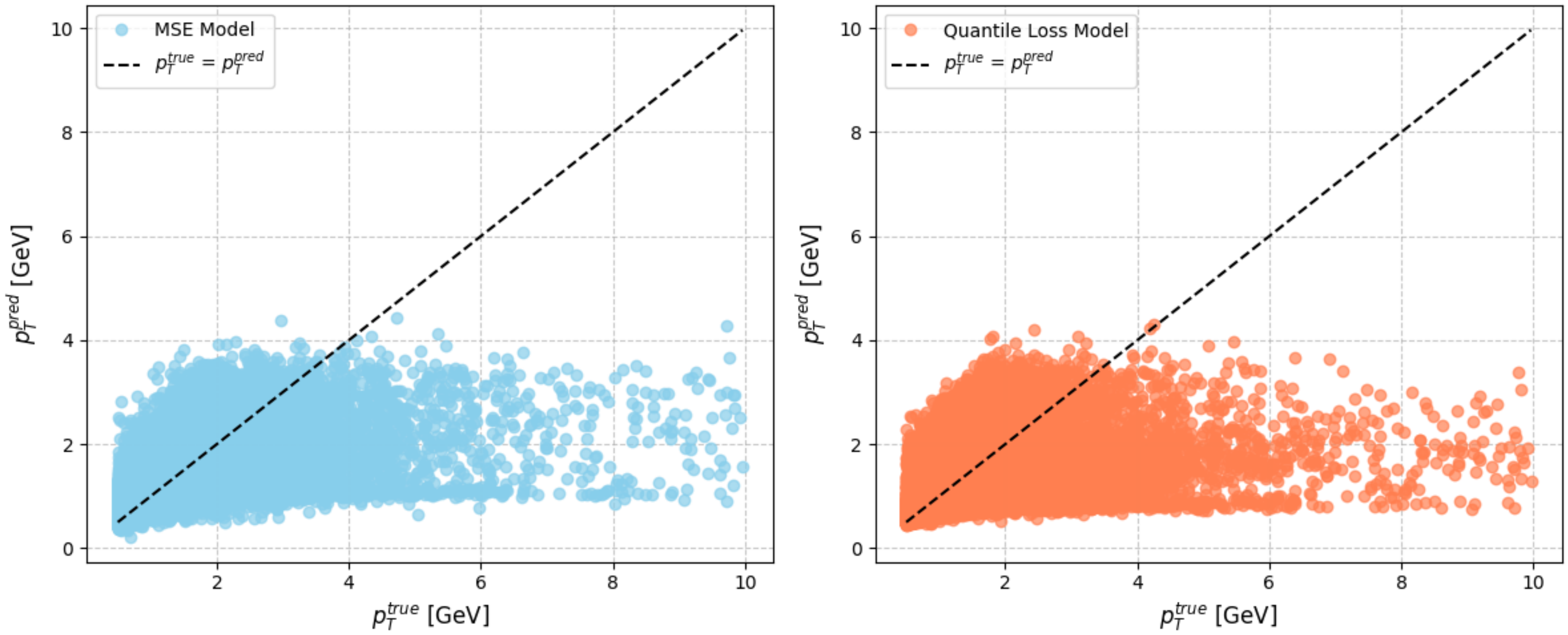


Impact of p_T sign

Signed p_T vs unsigned

Unsigned
No scattering cut

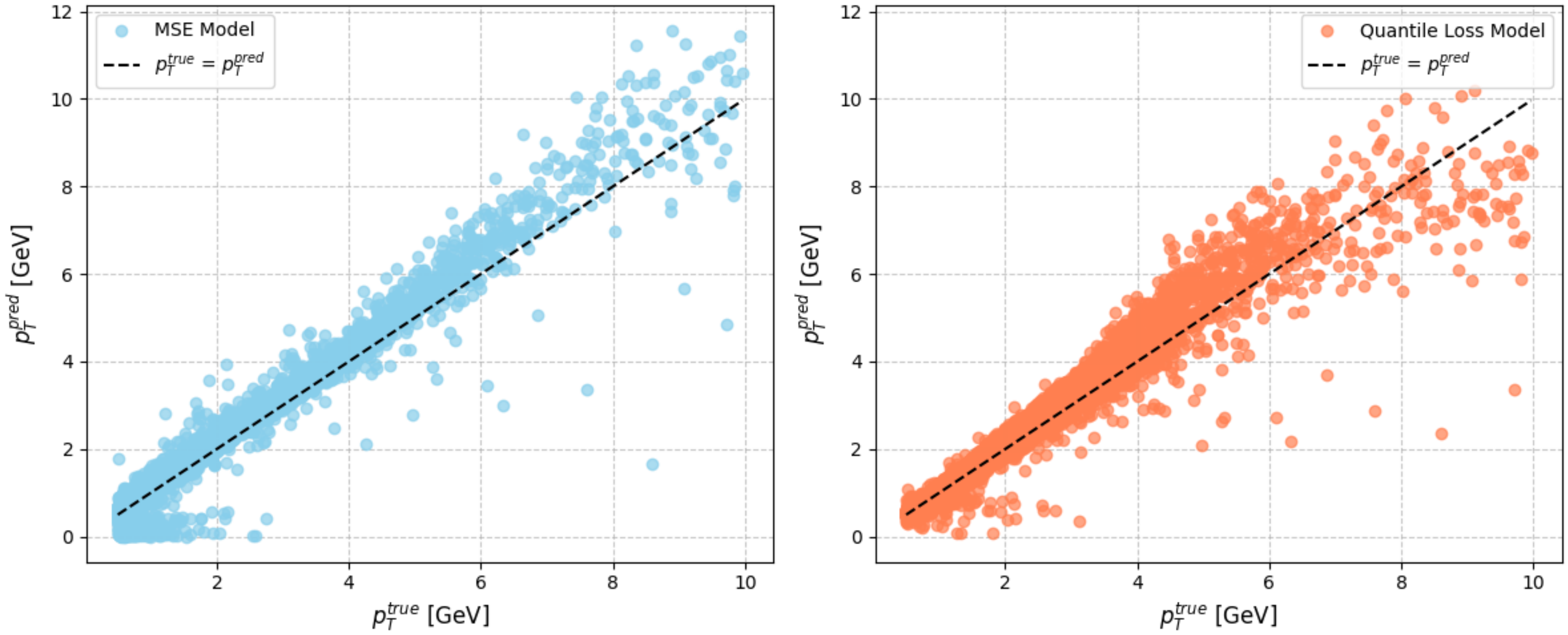
p_T^{true} vs p_T^{pred} $((tr, d\varphi, tz) \rightarrow (p_T, p_z))$



Signed p_T vs unsigned

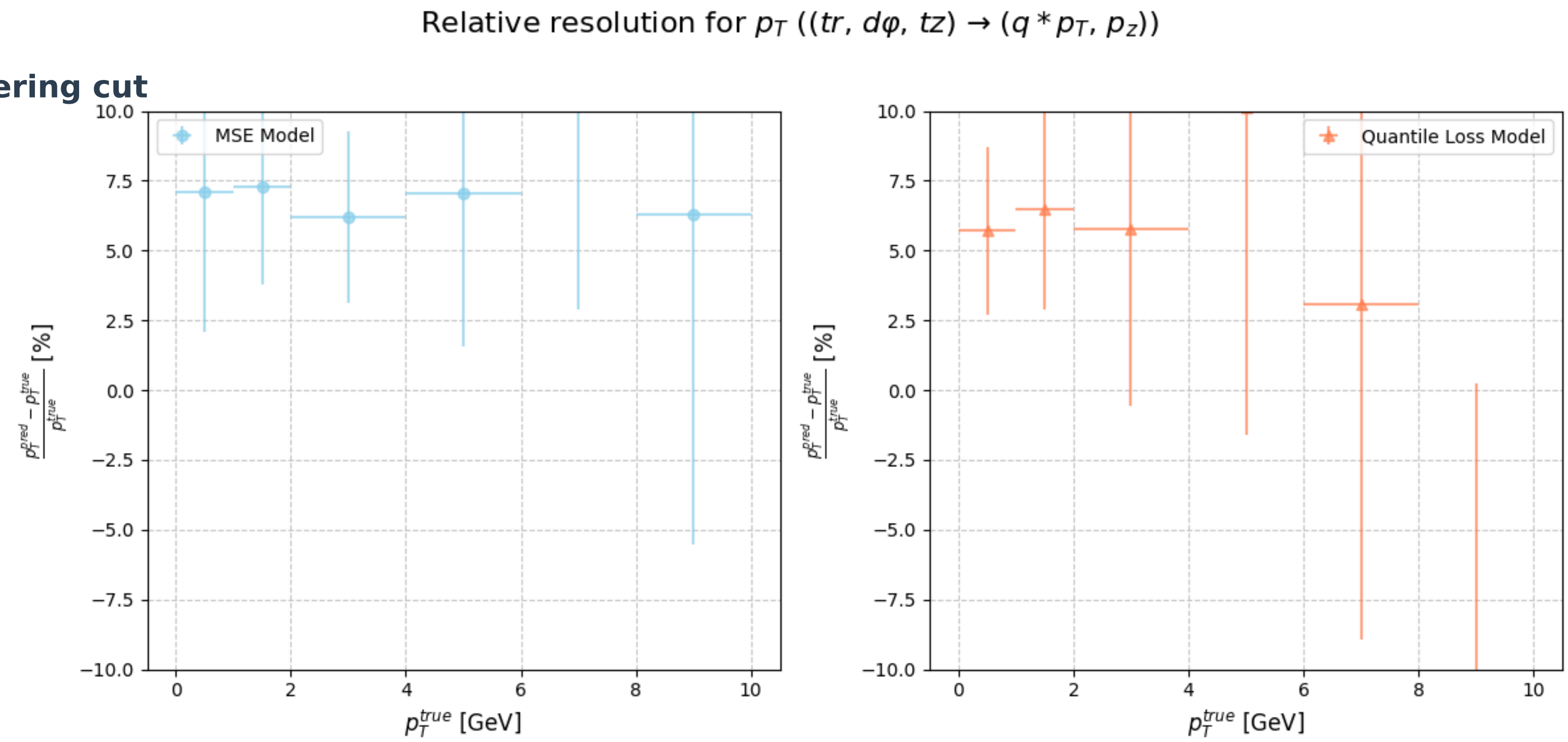
Signed
No scattering cut

p_T^{true} vs p_T^{pred} $((tr, d\varphi, tz) \rightarrow (q * p_T, p_z))$



Signed p_T vs unsigned

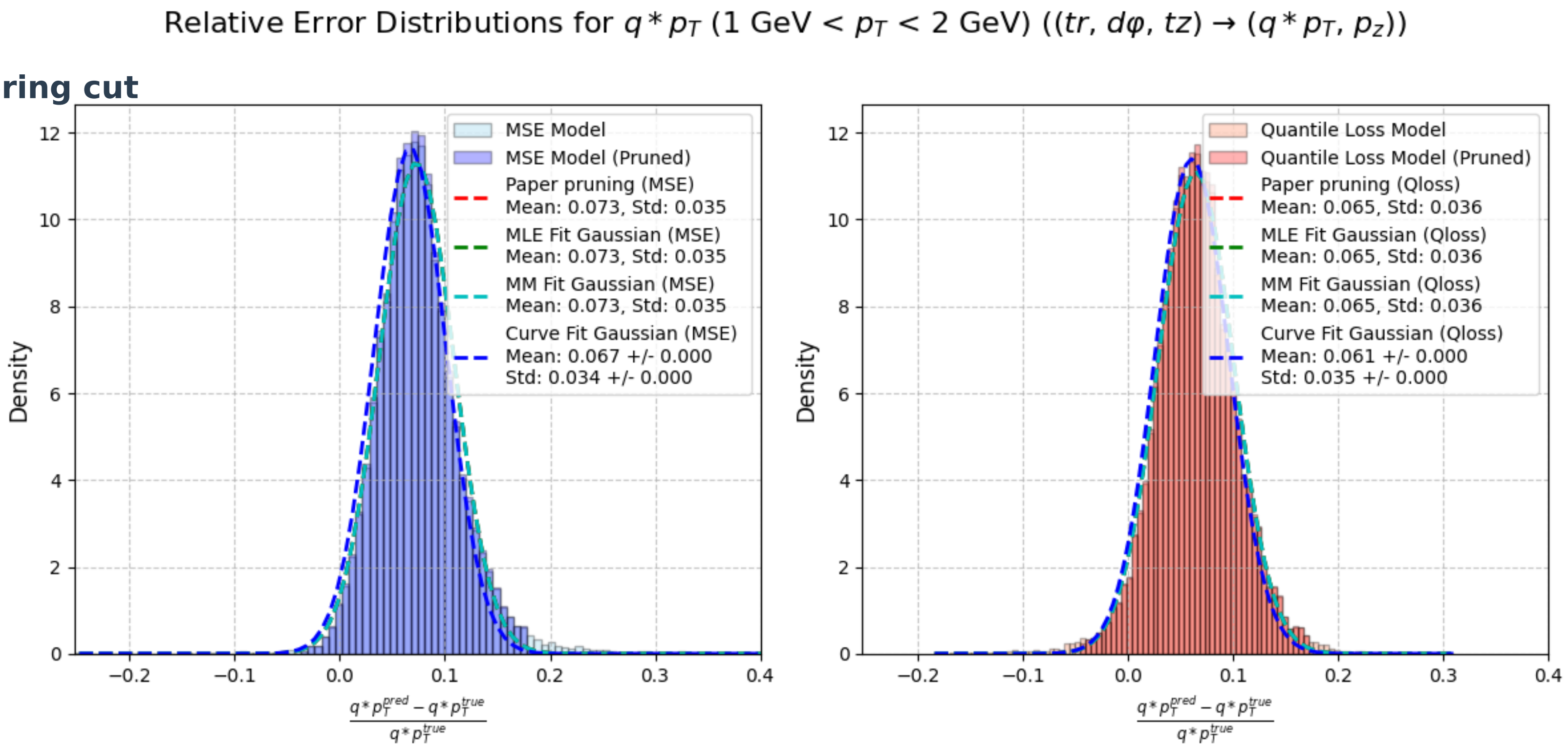
Signed
No scattering cut



Signed p_T vs unsigned

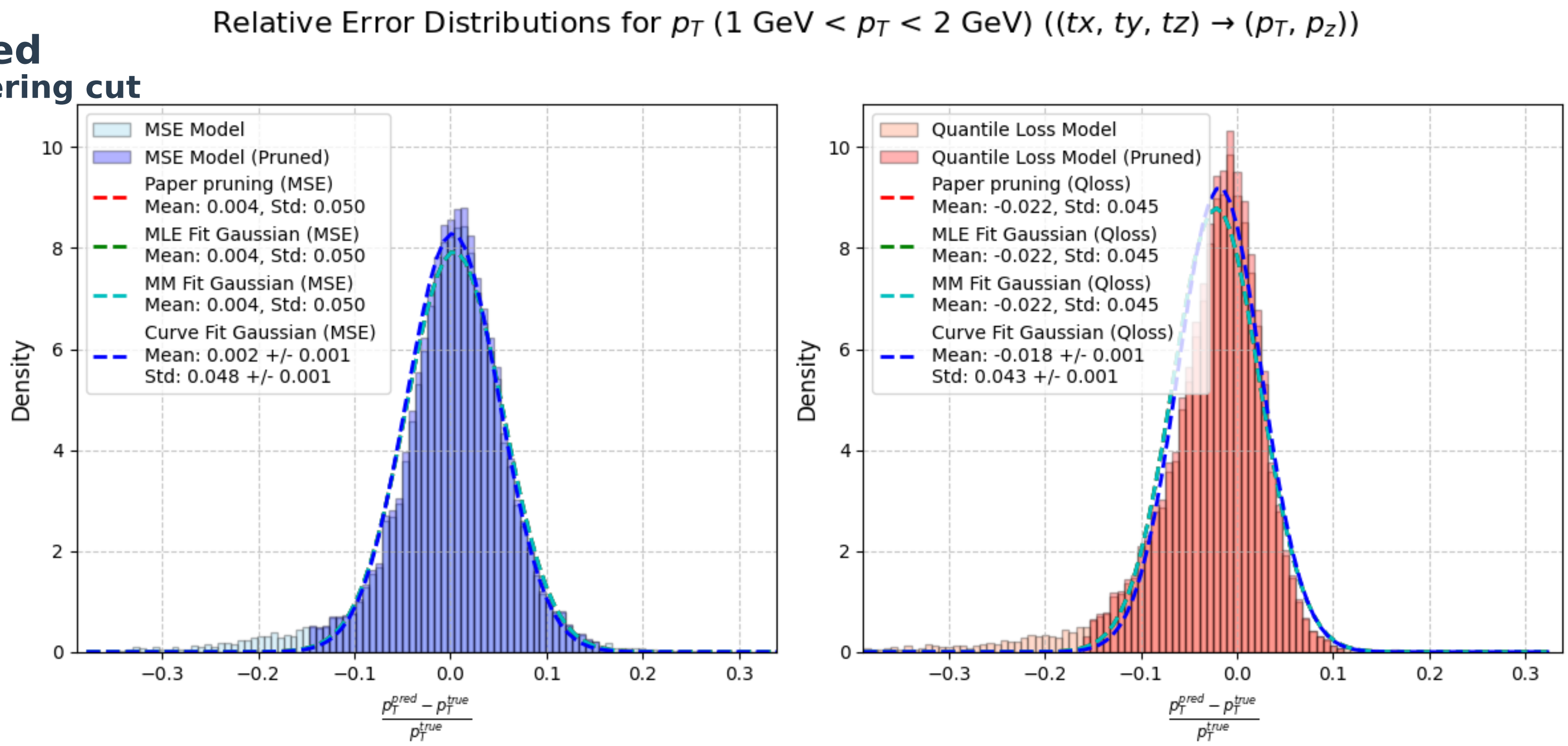
Signed

No scattering cut



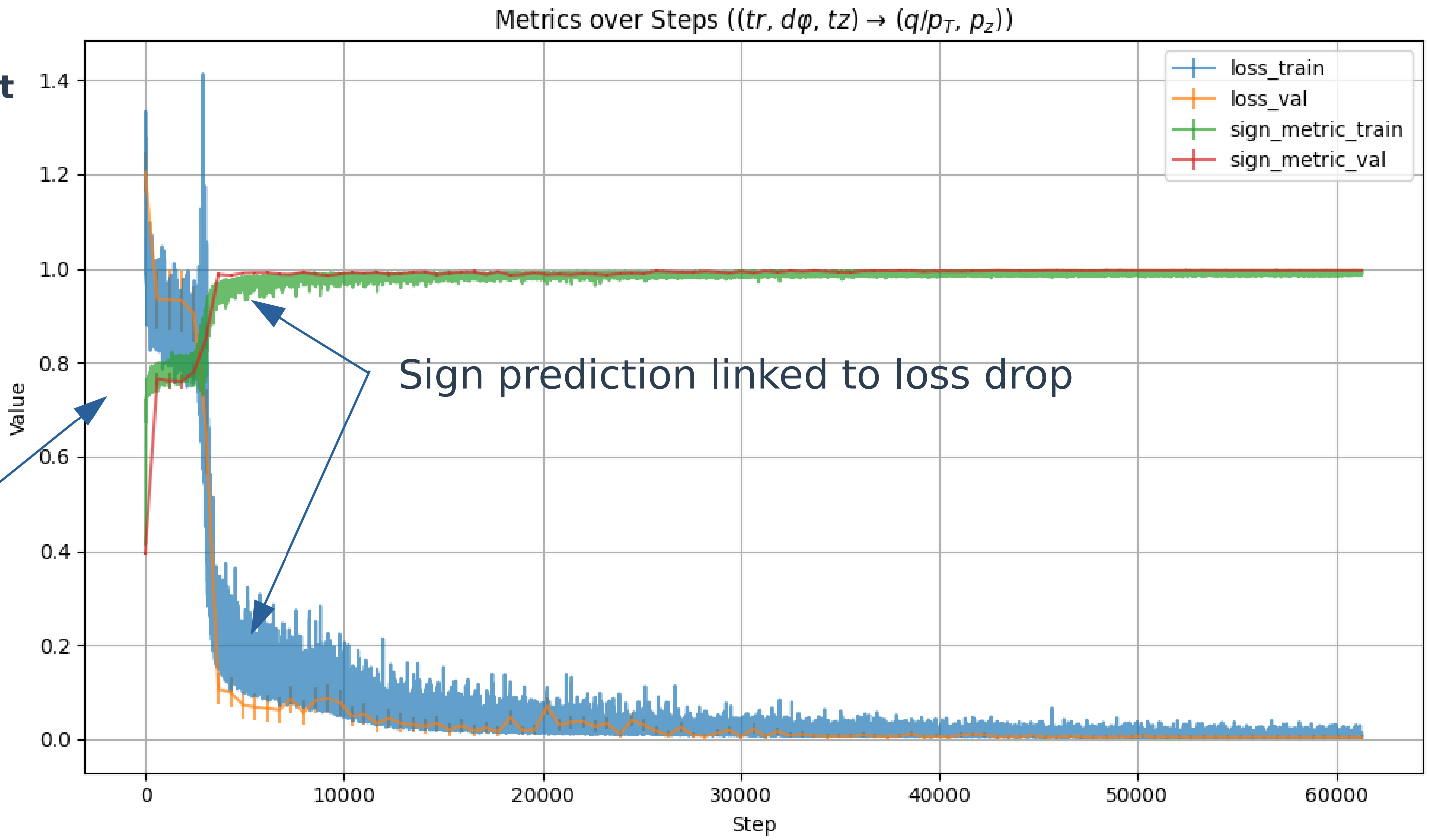
Signed p_T vs unsigned

Unsigned
No scattering cut



Loss and charge sign

Scattering cut



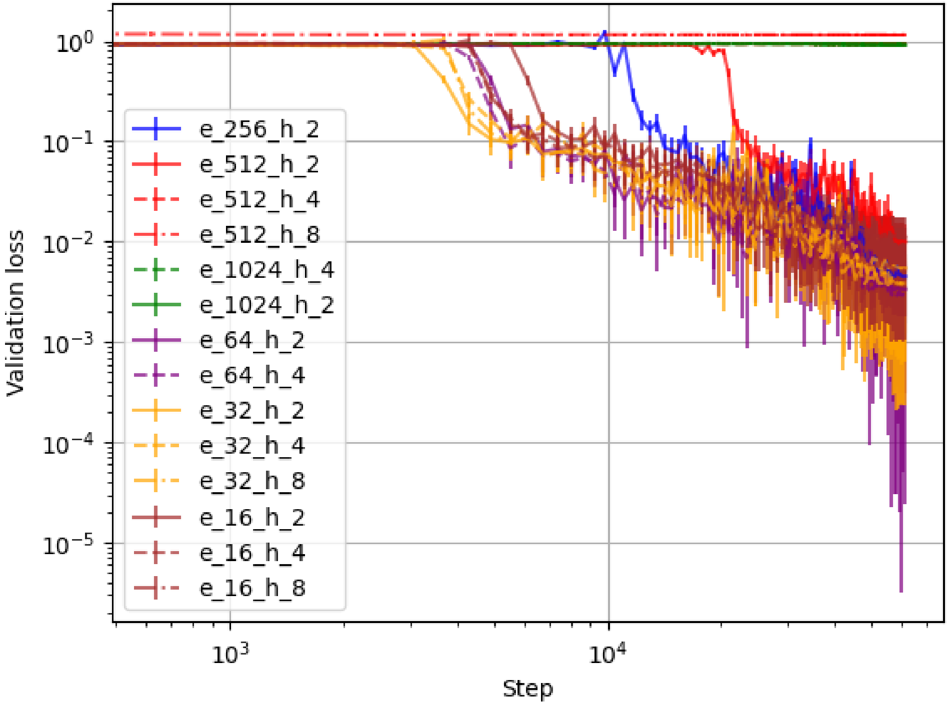
Why 80%?

Sign prediction linked to loss drop

Hyperparameters

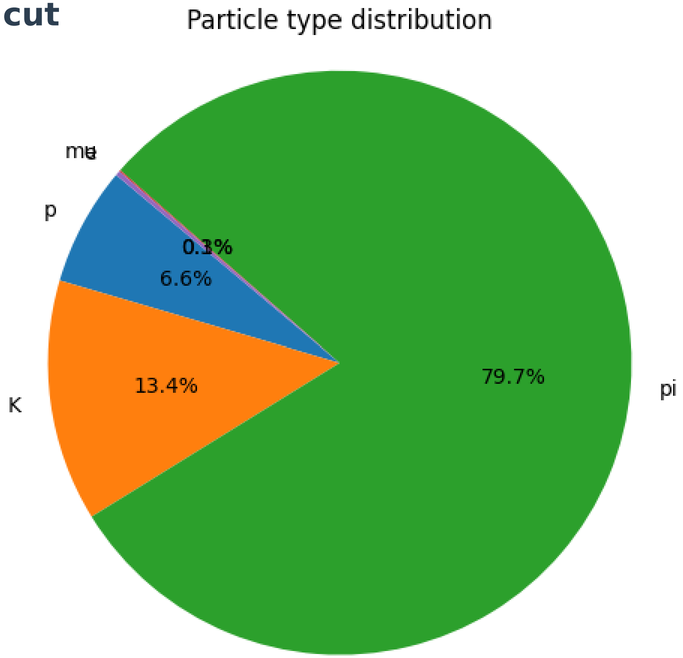
Architecture optimization

No scattering cut
100 epochs



Signed p_T vs unsigned

1st file (test dataset)
After scattering cut

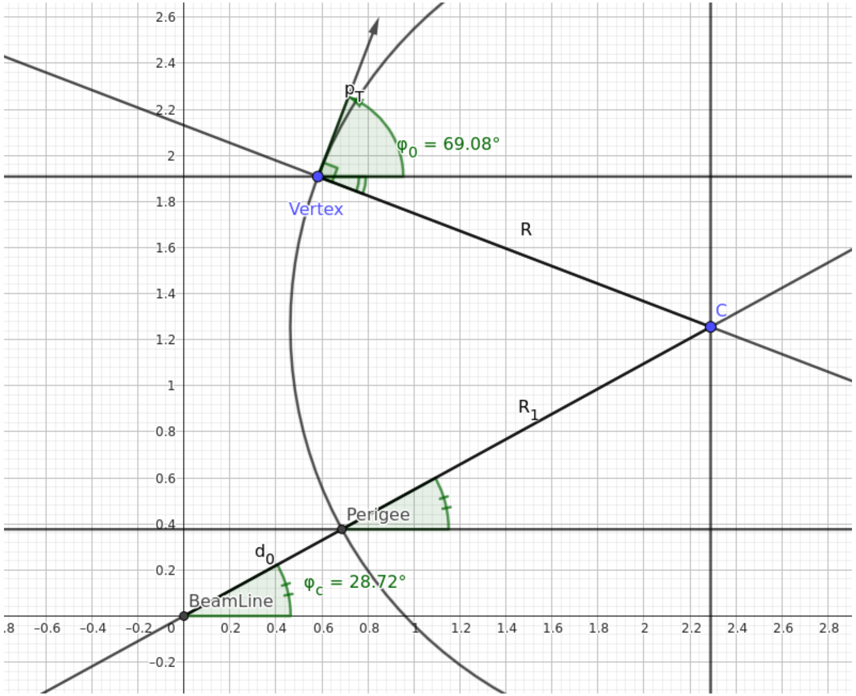


pi: 79.72% (62457) / 78345
K: 13.37% (10471) / 78345
p: 6.57% (5149) / 78345
e: 0.26% (205) / 78345
mu: 0.08% (63) / 78345

Computation of d0

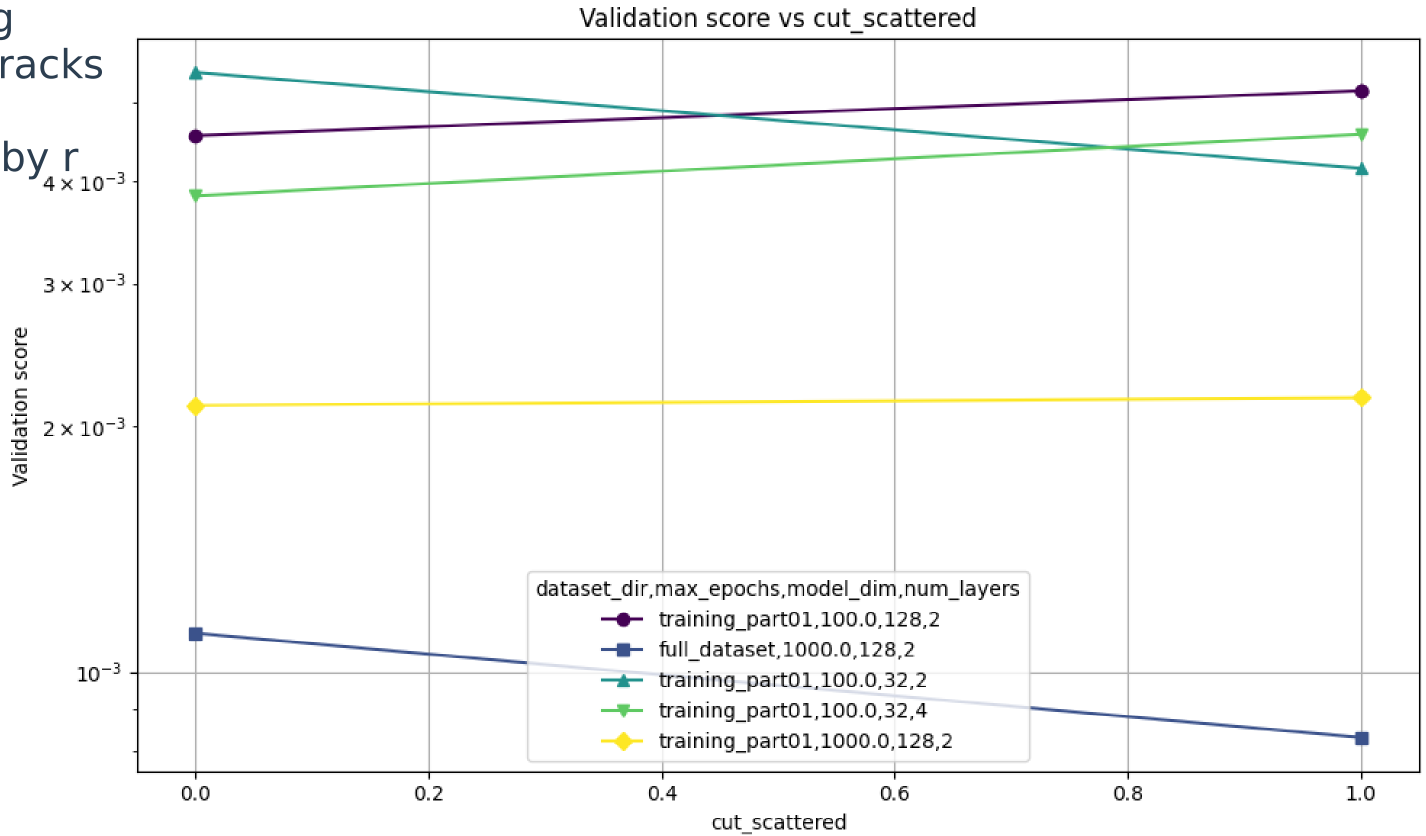
Lorentz force

$$R\,[m] = \frac{p_T\,[GeV/c]}{q\,[C] \times B\,[T]}$$

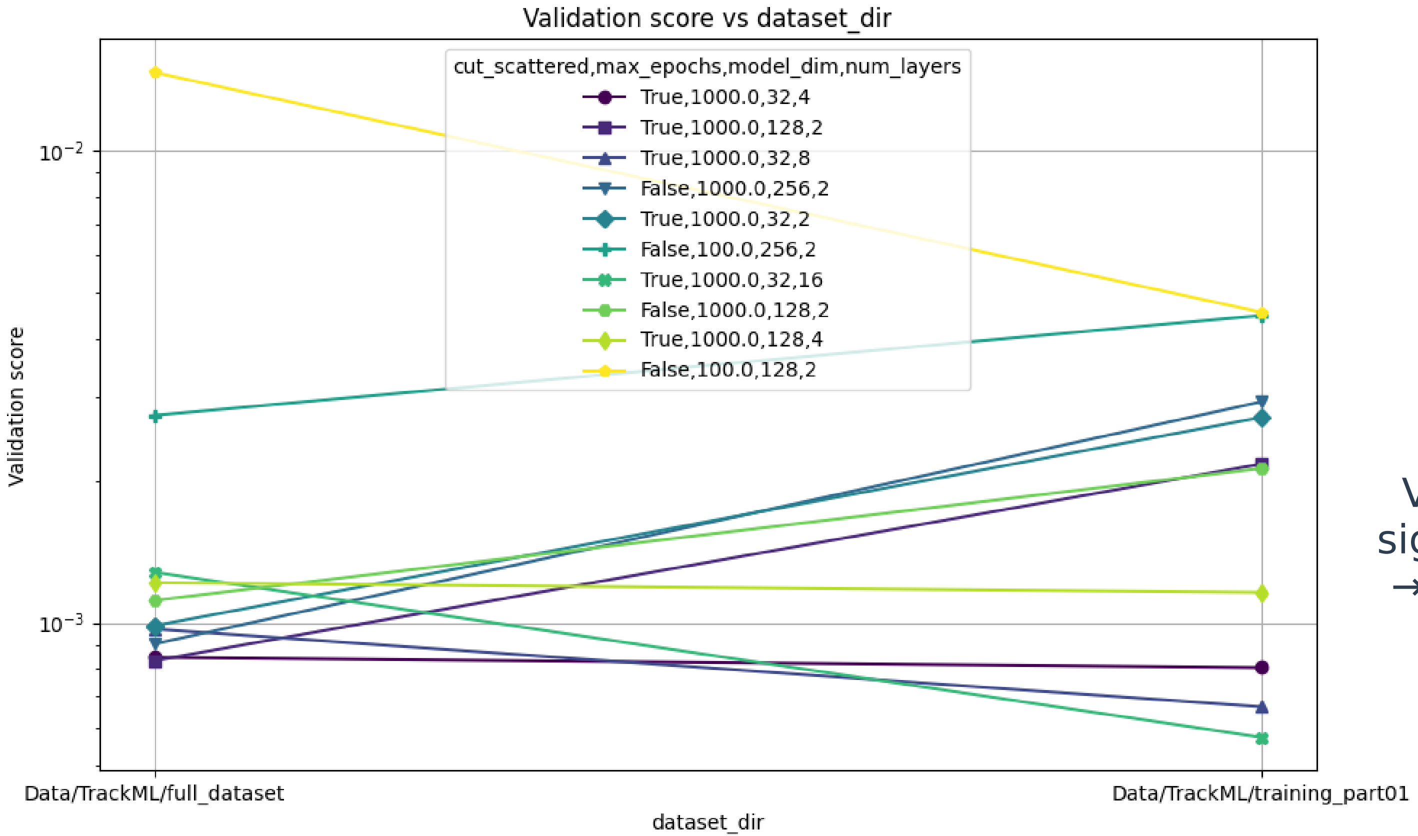


Hyperparameter optimization

Removing
0.3% of the tracks
and
sorting hits by r

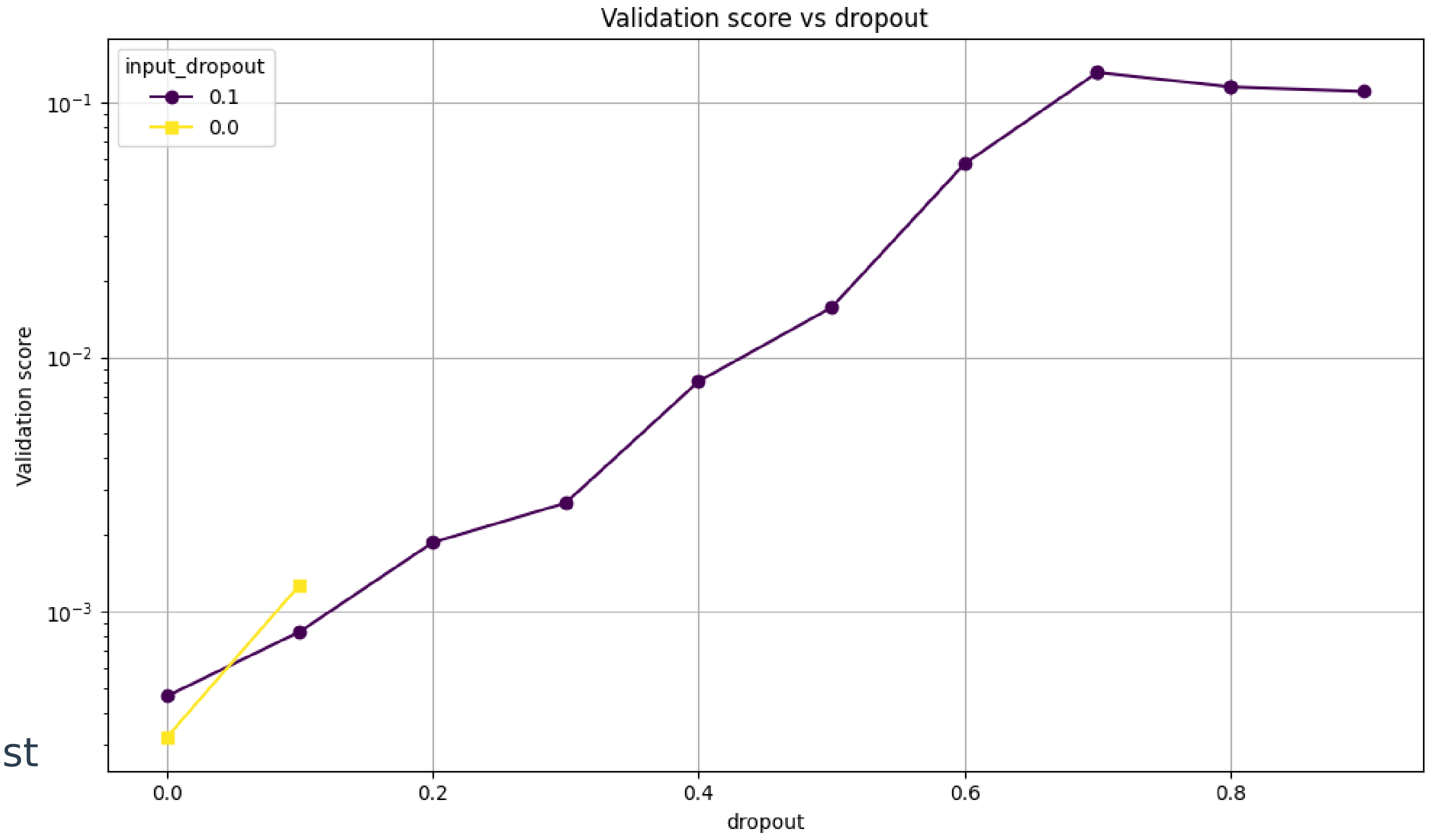


Hyperparameter optimization



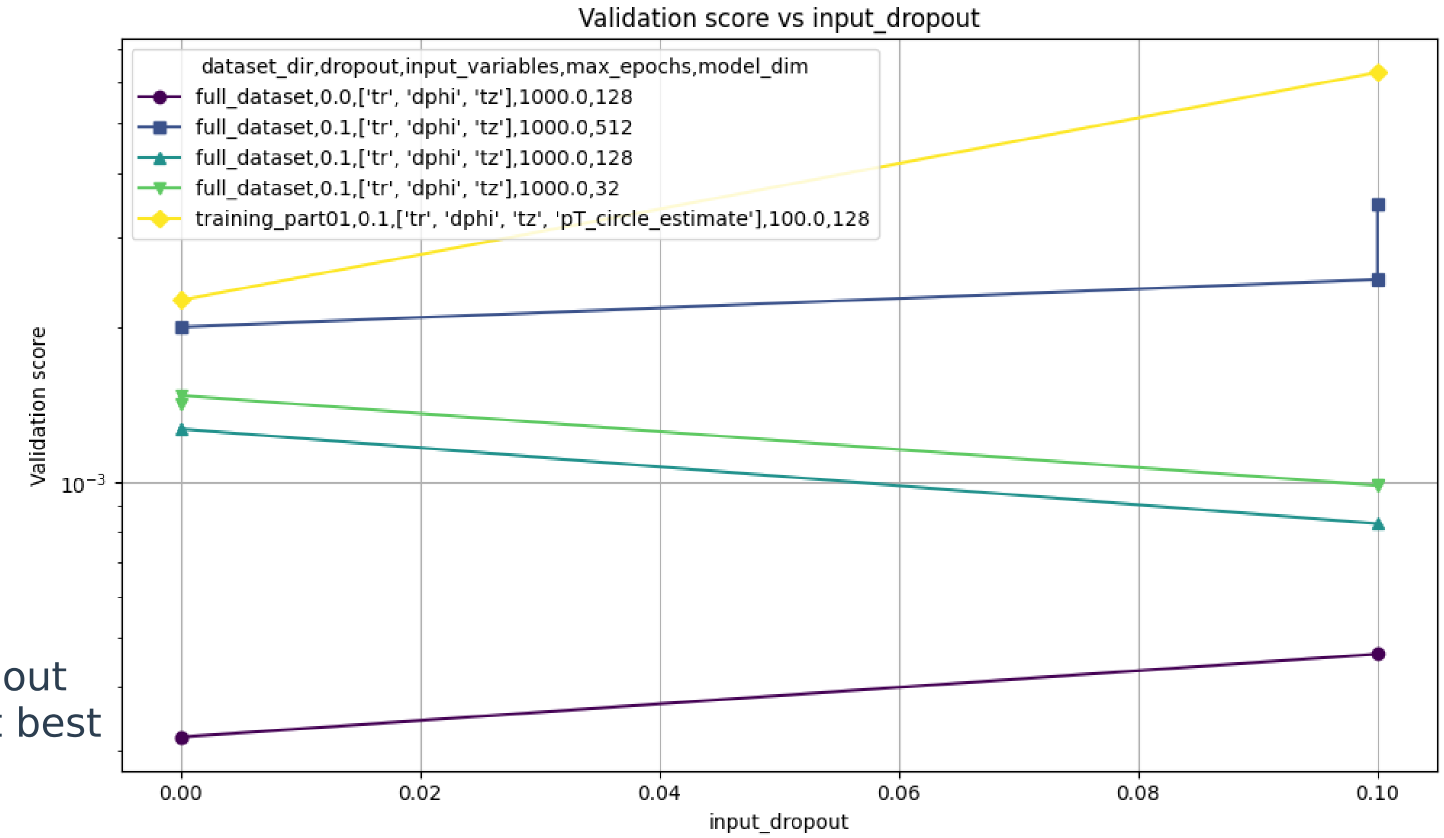
Validation dataset significantly different → hard to conclude

Hyperparameter optimization



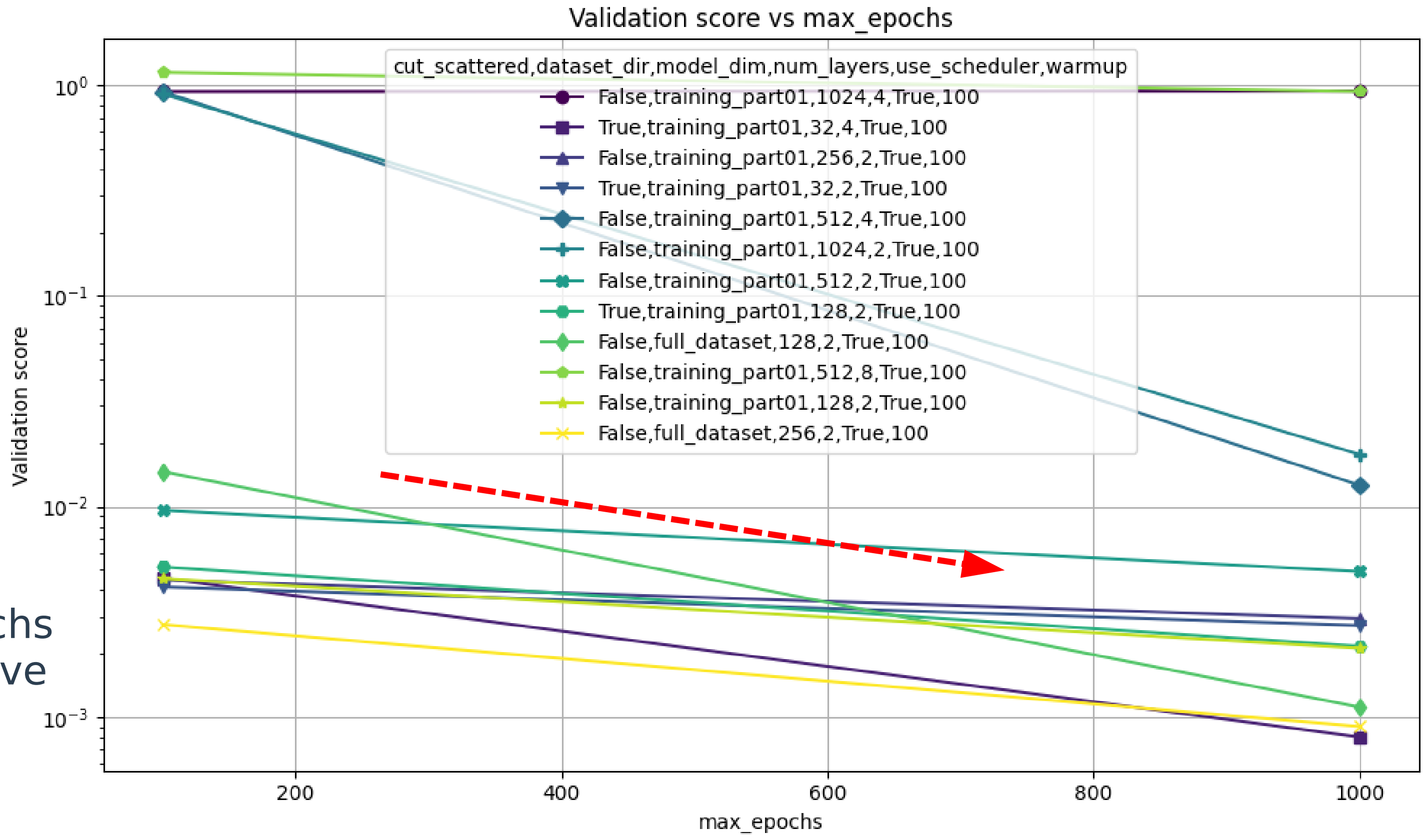
No dropout works the best

Hyperparameter optimization



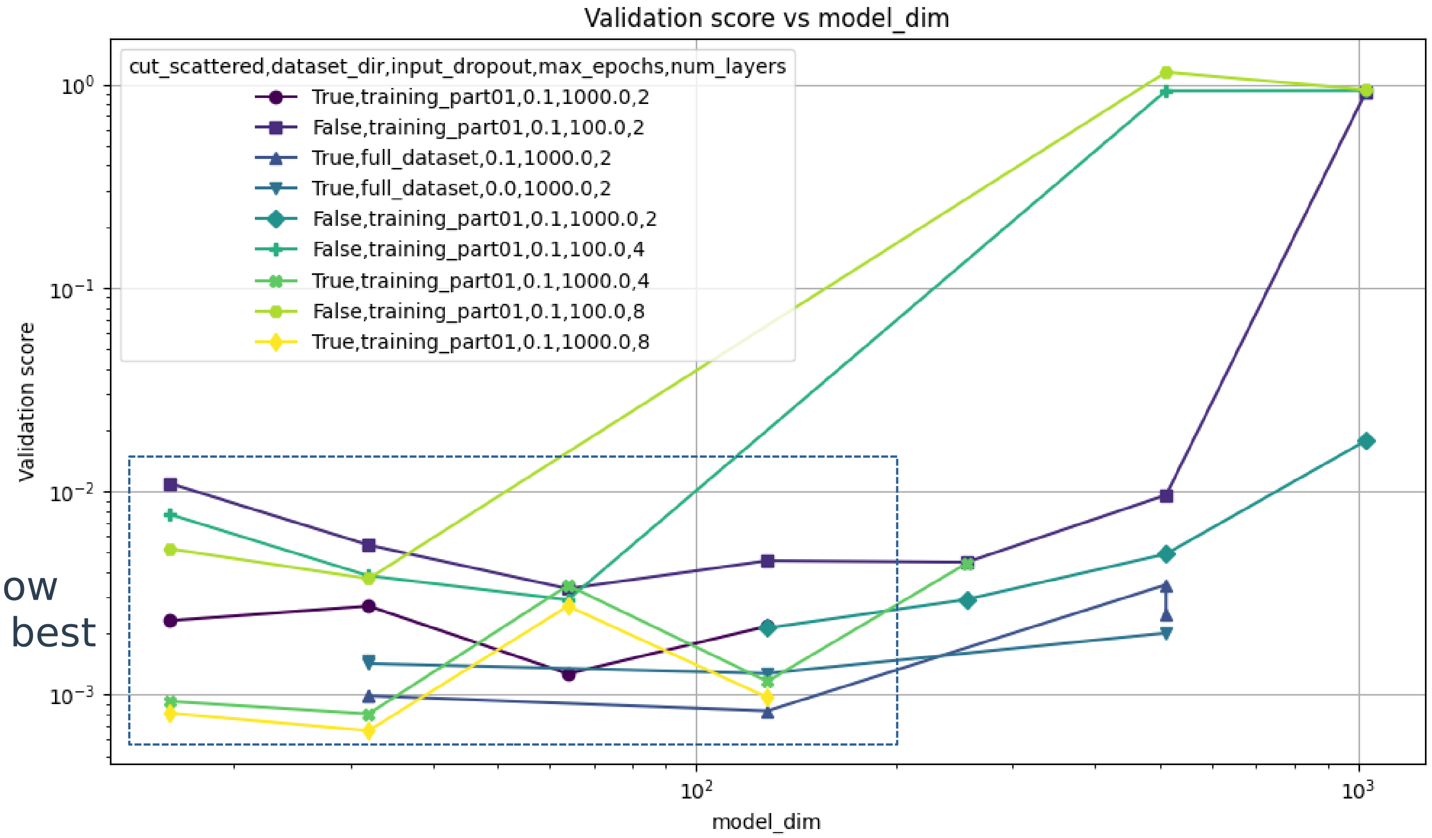
Unclear
but no dropout
is the current best

Hyperparameter optimization



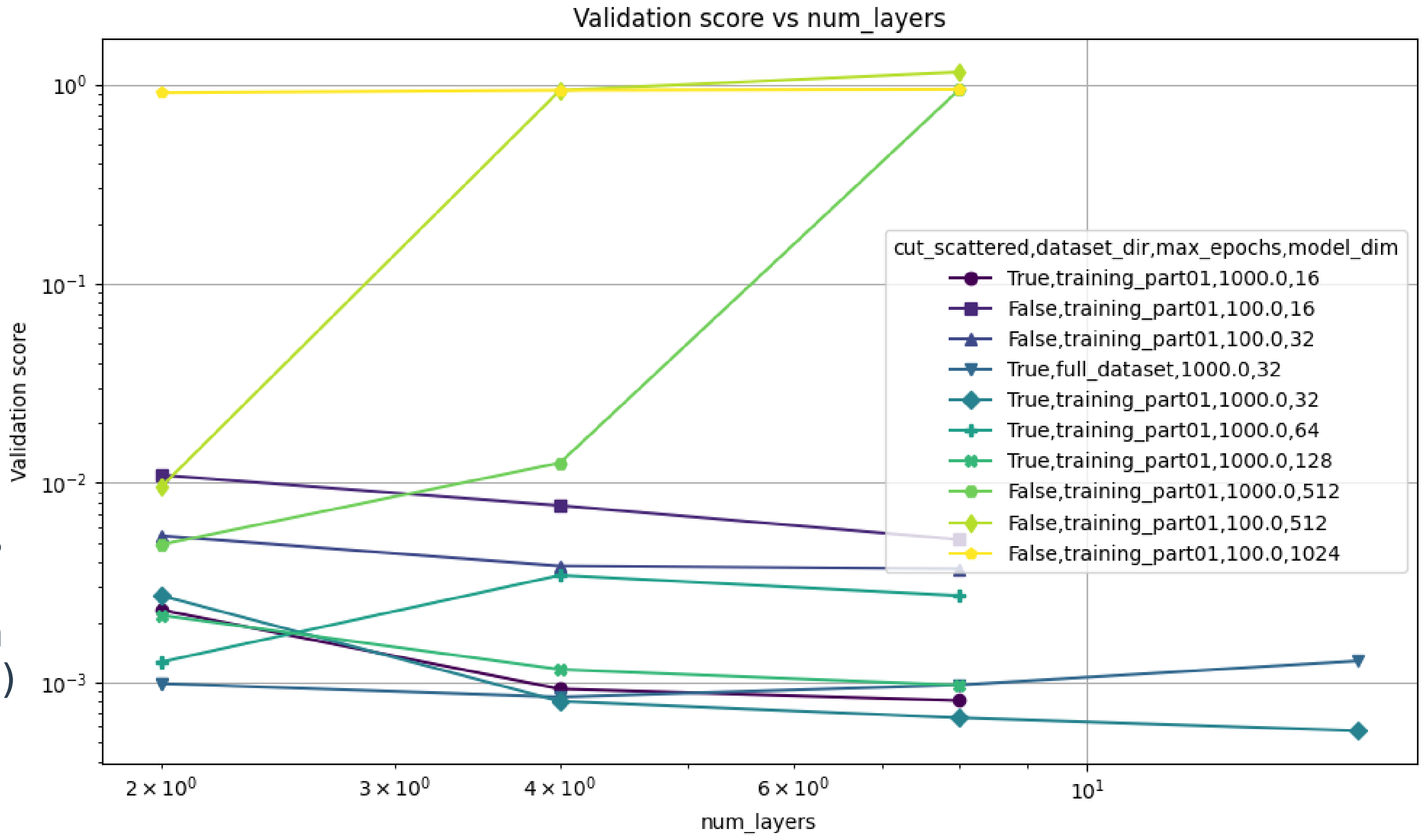
Training
for more epochs
always improve

Hyperparameter optimization



128 and below seem to work best

Hyperparameter optimization



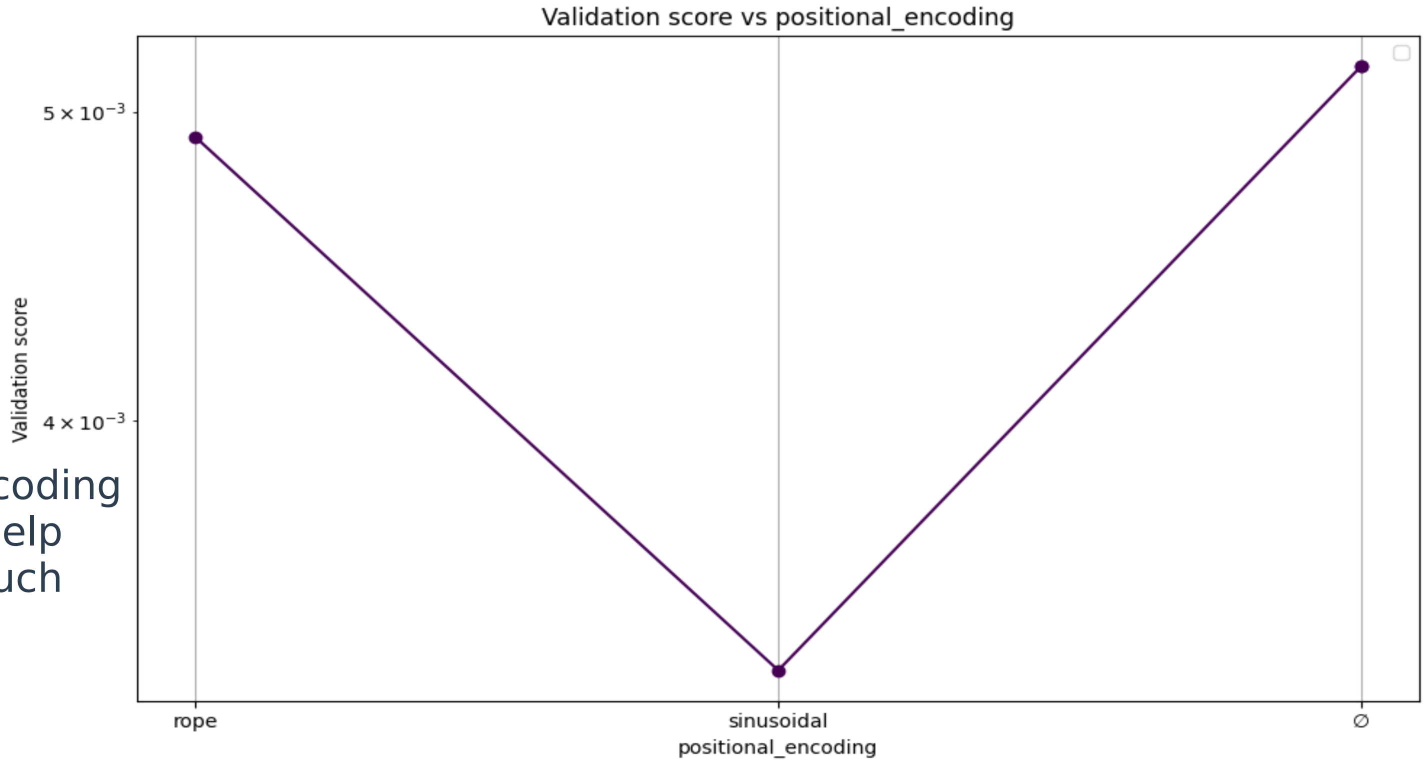
More layers is better (except with dim >= 512)

Hyperparameter optimization

Tested giving track hit index position to the model

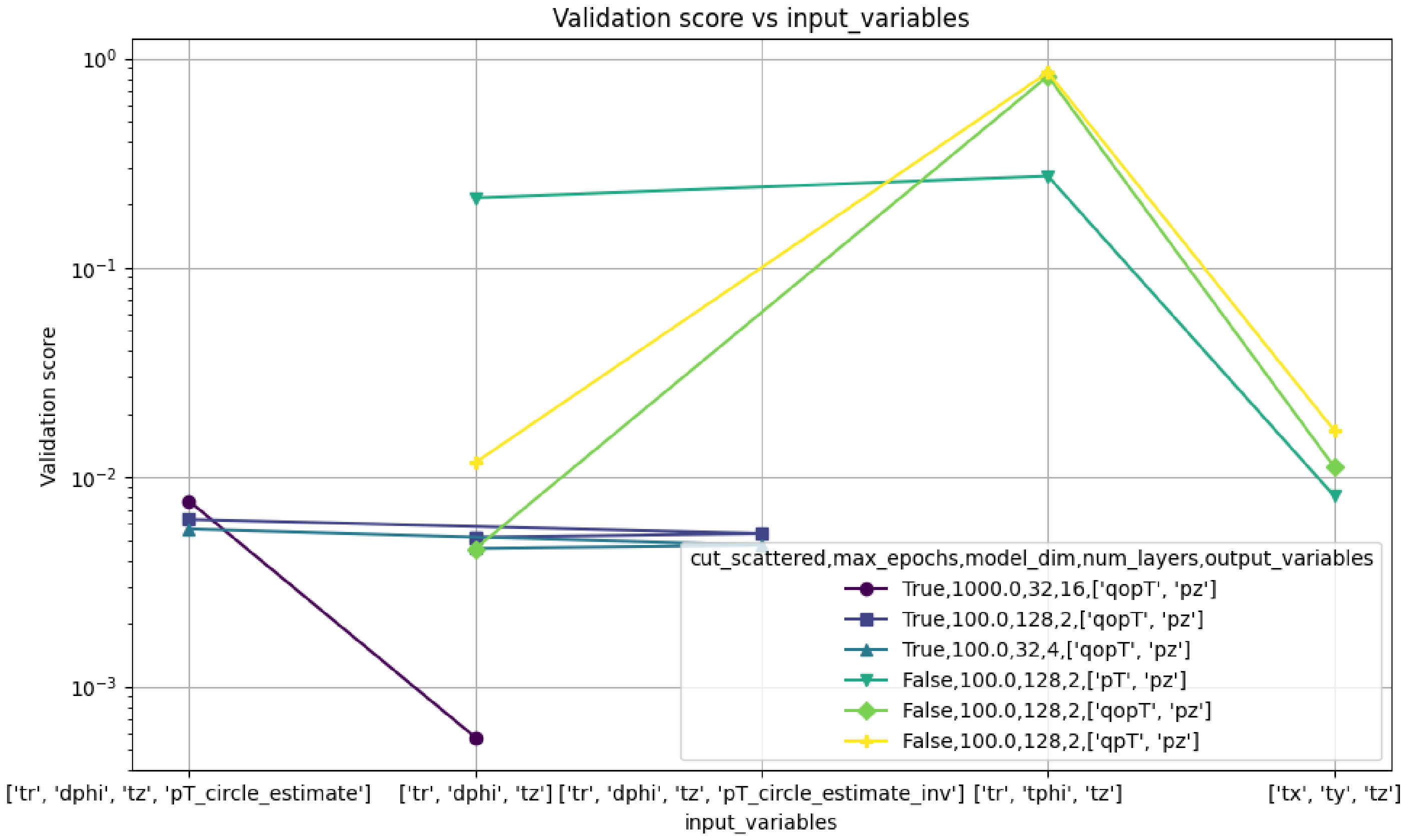
100 epochs
Sorted by r

Still testing



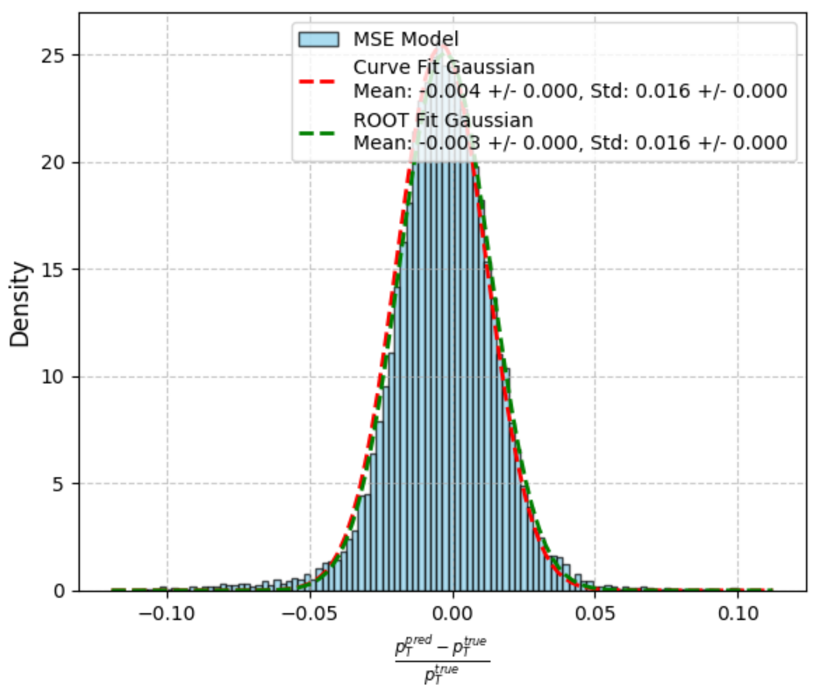
Positional encoding seems to help but not much

Hyperparameter optimization

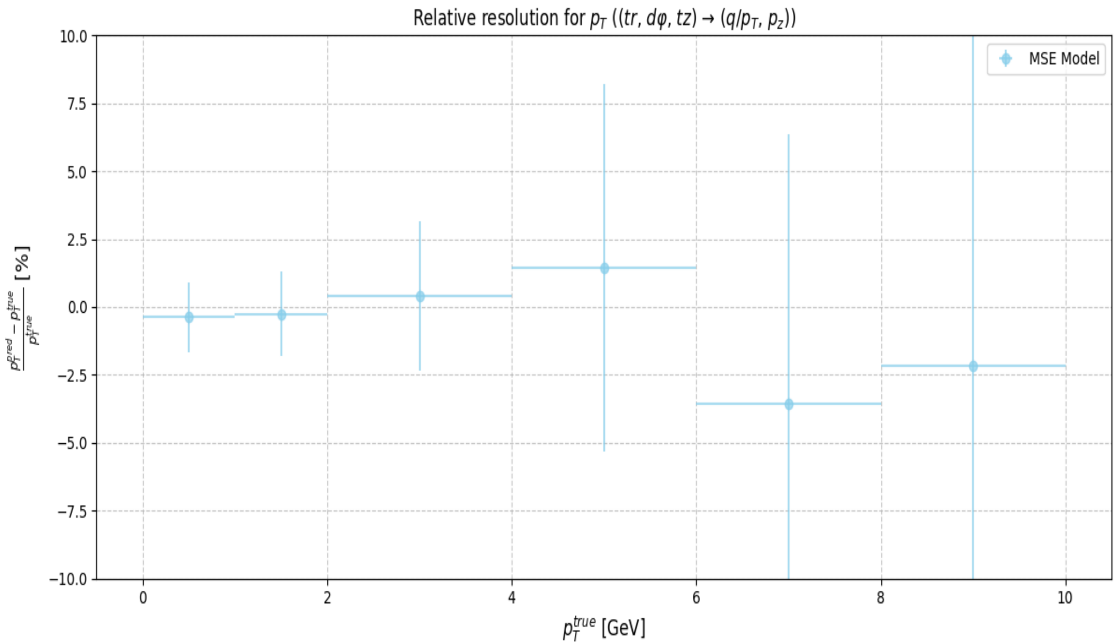


Current best models

r Distributions for p_T (1 GeV < p_T < 2 GeV) ((tr, $d\varphi$, tz)

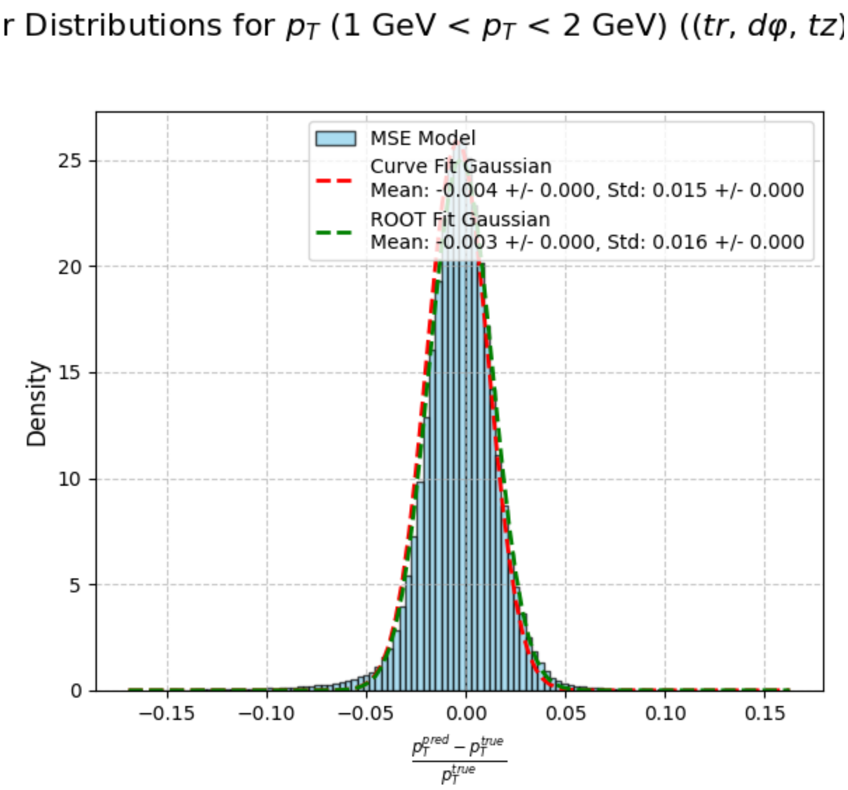


Embedding: 32; 16 layers



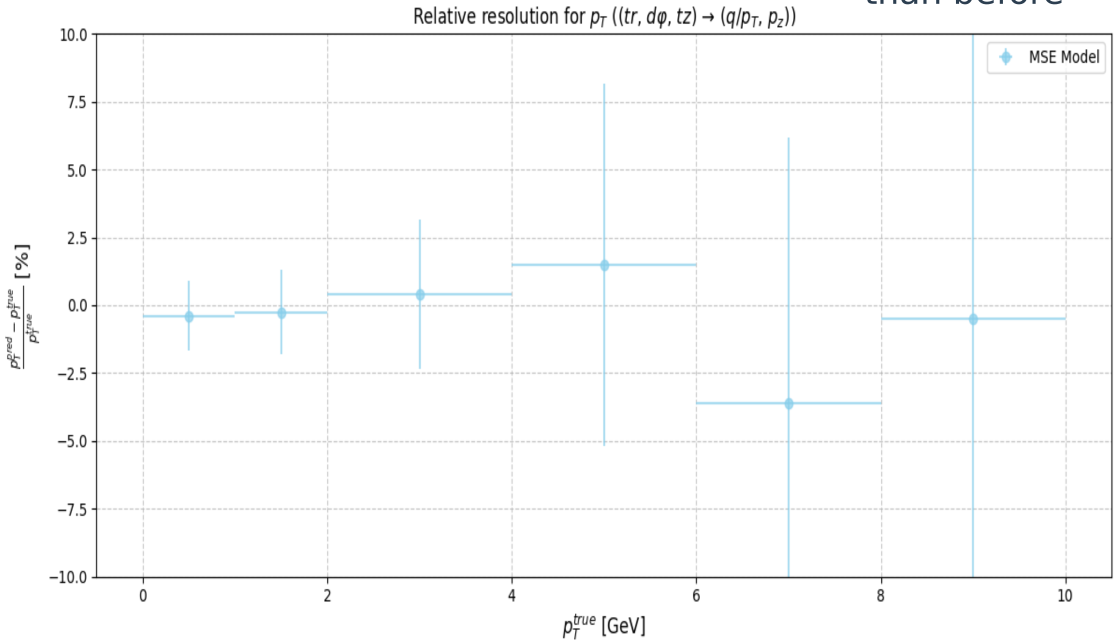
1st file (test dataset)
After scattering cut

Current best models



Embedding: 32; 16 layers

Same model than before

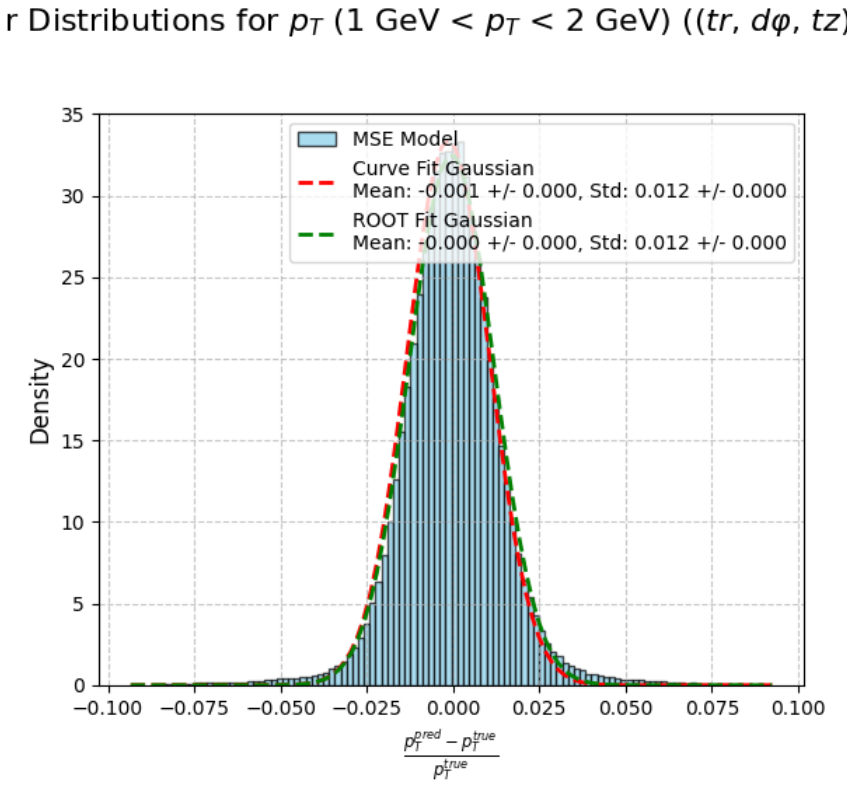


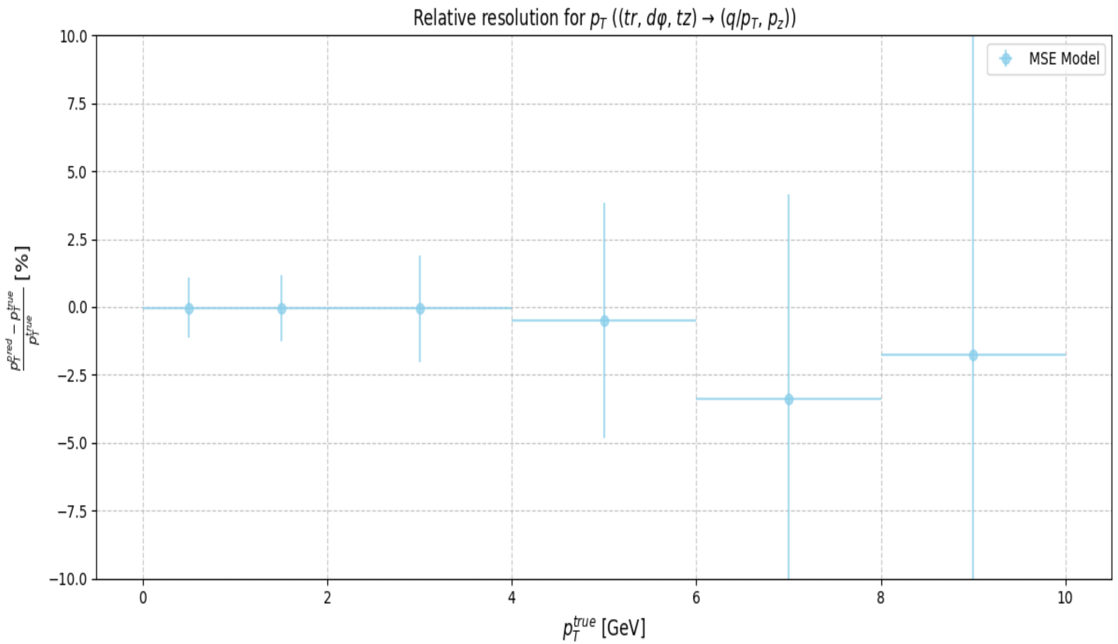
Full dataset (test dataset)

After scattering cut

Current best models

Embedding: 128; 2 layers; no dropout





Full dataset (test dataset)

After scattering cut

DOCTORAL TRAINING

Formations

- **Ecole Doctorale (UGA):**
  - Requires 120 hours: 1/3 Scientific, 1/3 Professional, 1/3 Transversal
  - Current: 113/120 (+15 waiting for validation)
- **Professional:**
  - "S'ADAPTER A SON ENVIRONNEMENT DE TRAVAIL" (10 hours)
  - "Formation Entreprenariat PhDiscovery 2024" (30 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)
  - Mooc on ethics (15 hours)
  - MOOC "Intégrité scientifique dans les métiers de la recherche" (15 hours)

Poster and publications

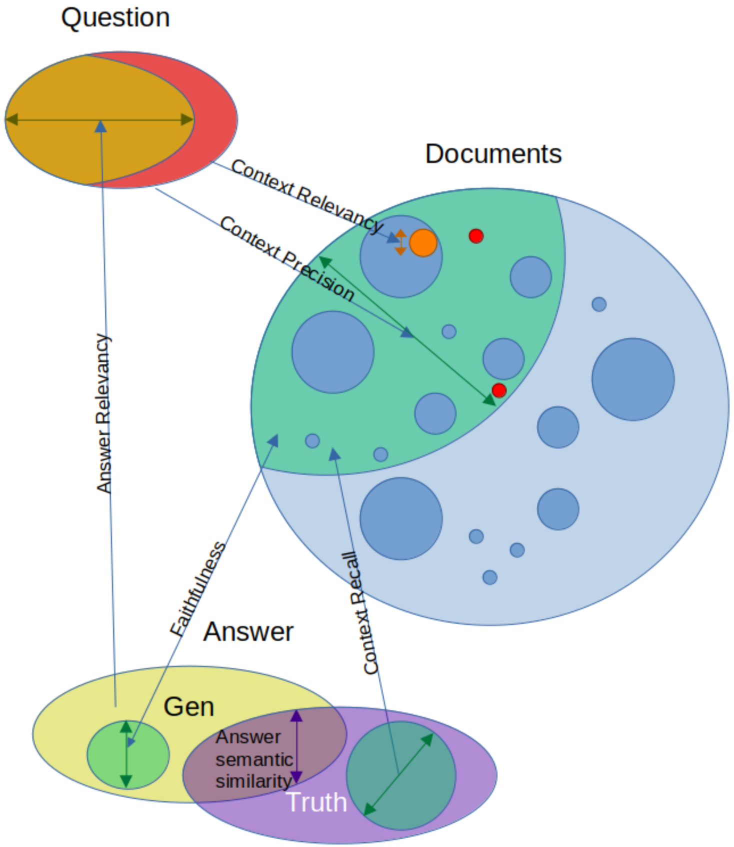
- Poster Connecting The Dots 2023
- Proceeding Poster Connecting The Dots 2023
- Proceeding Journée Rencontre Jeunes Chercheurs 2023
- Tutoriel ATLAS Machine Learning Workshop - chATLAS

chATLAS

- **ATLAS chatbot with ATLAS protected documents**

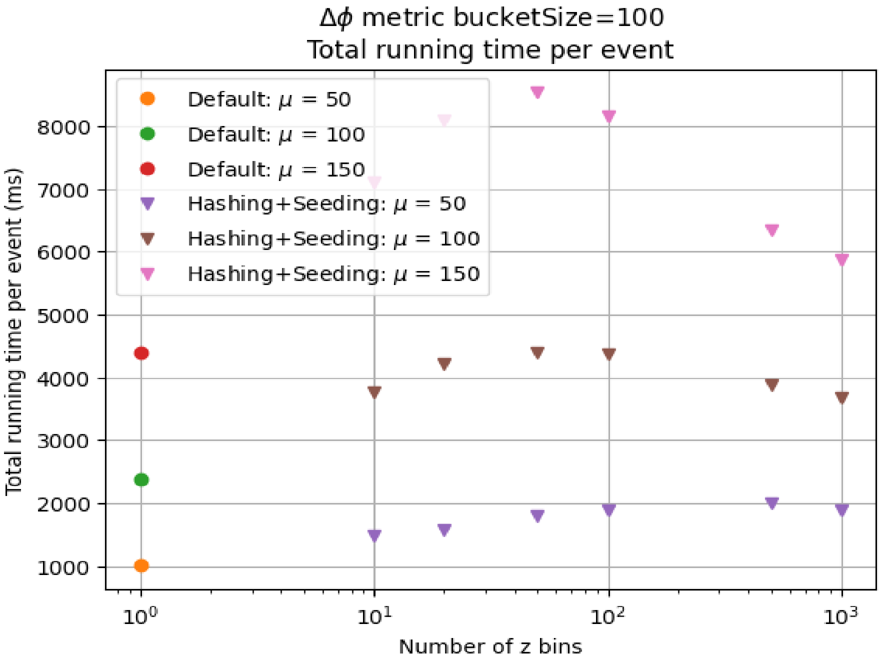
  (Retrieval Augmented Generation)

- **Worked on the evaluation**
- **Quitted team in september 2024**

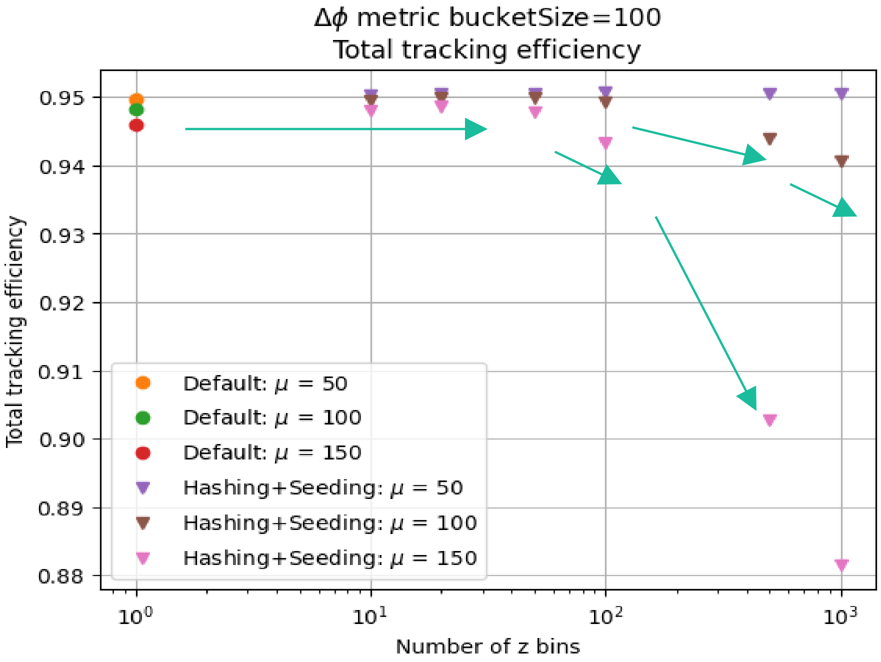


BACKUP

Hashing performance: Timing and efficiency

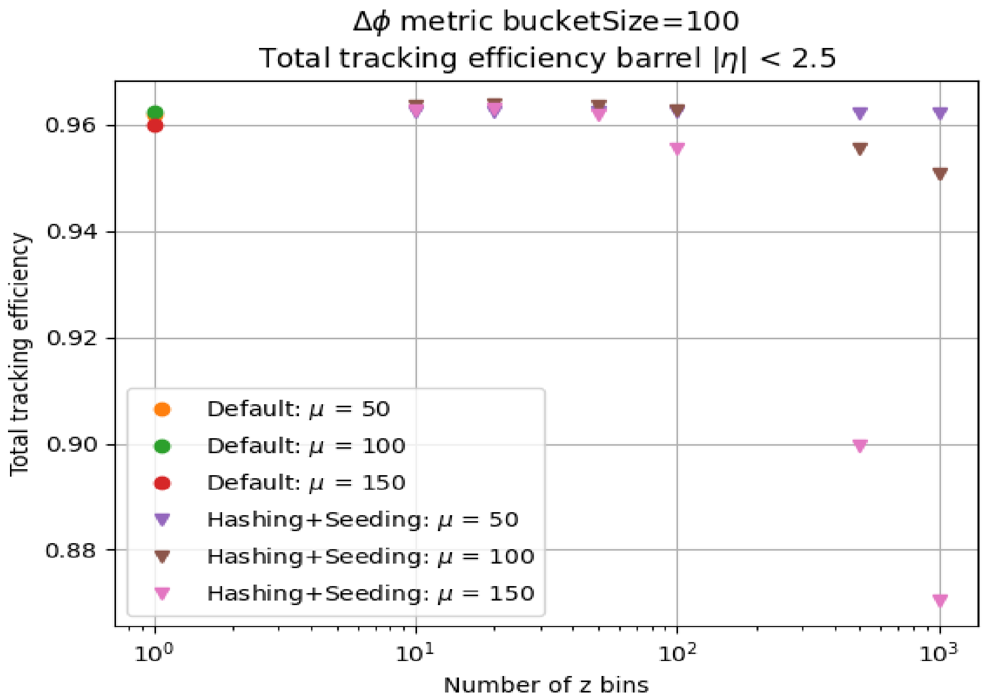


Running time ~ x2

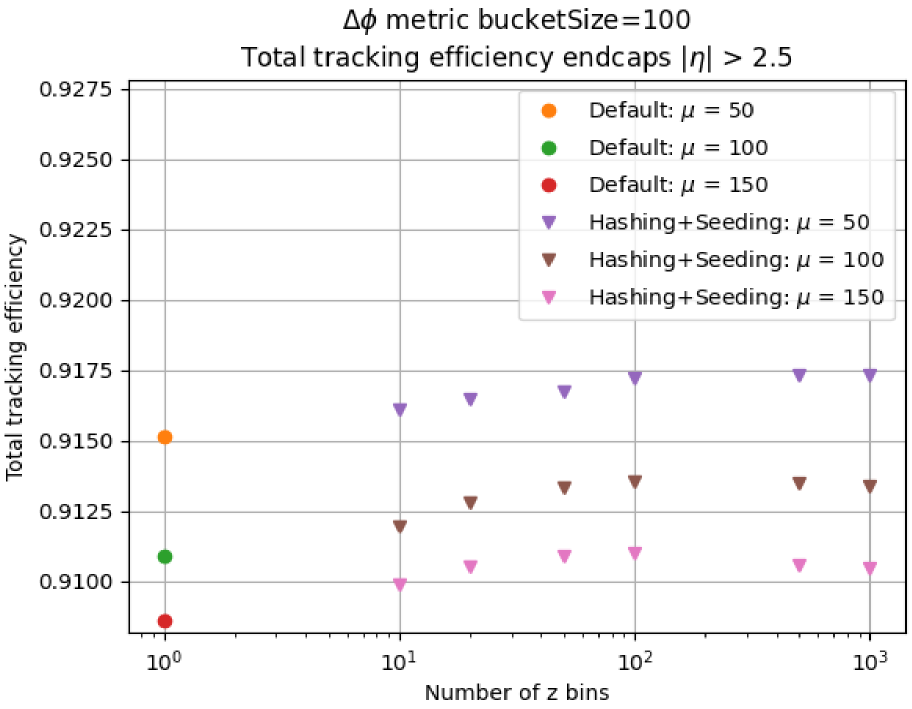


Improvement for small number of bins

Hashing performance: Efficiency (detailed)

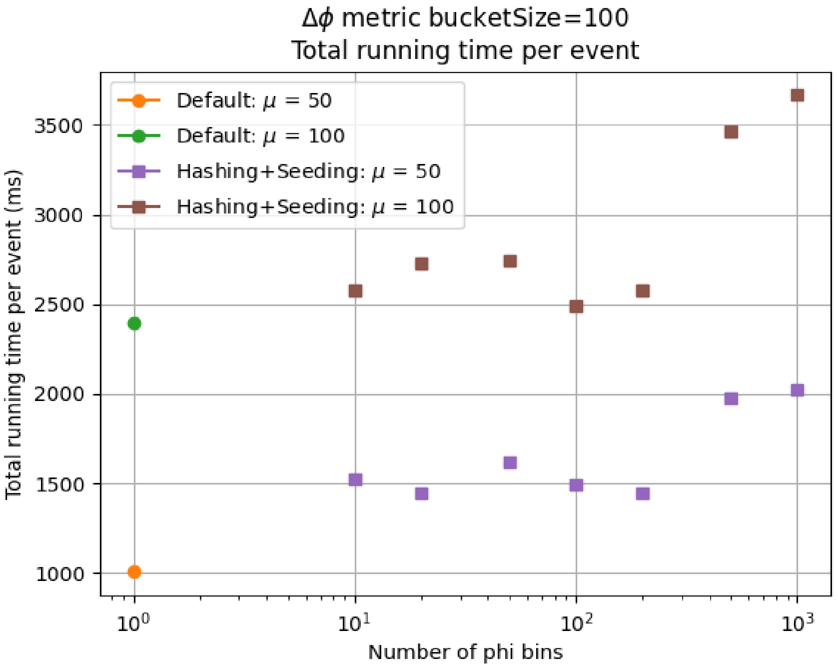


Improves then drops

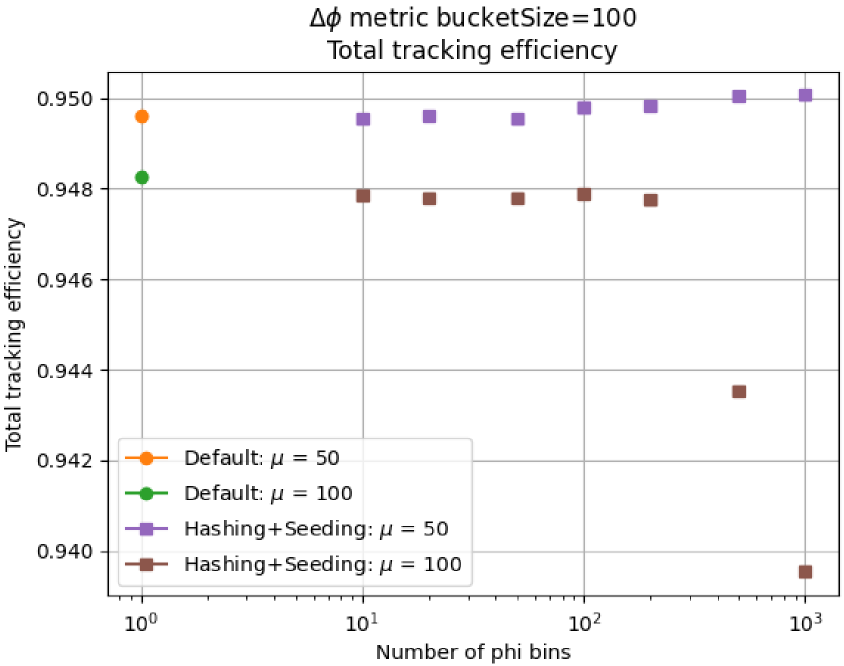


Always improve

Hashing φ bins: Timing and efficiency

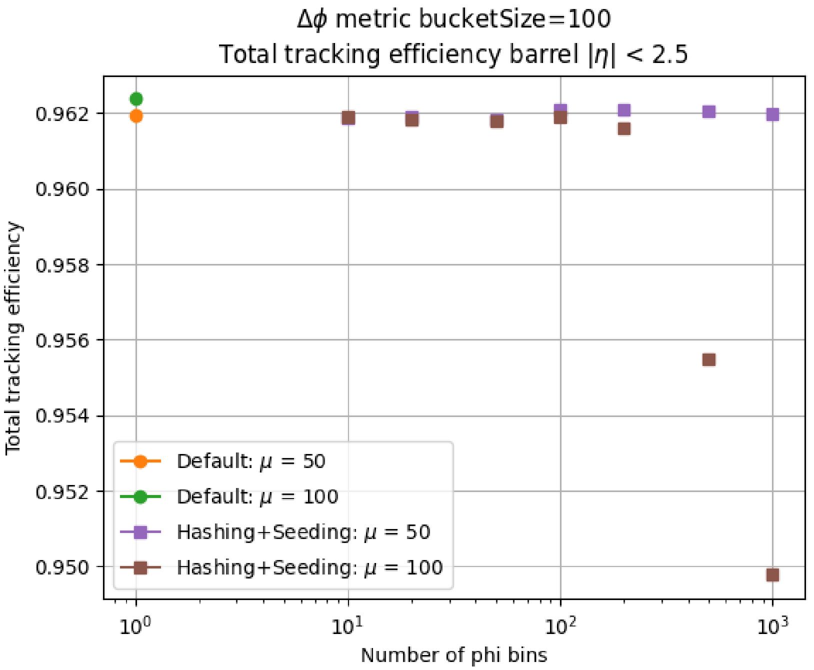


Similar running times

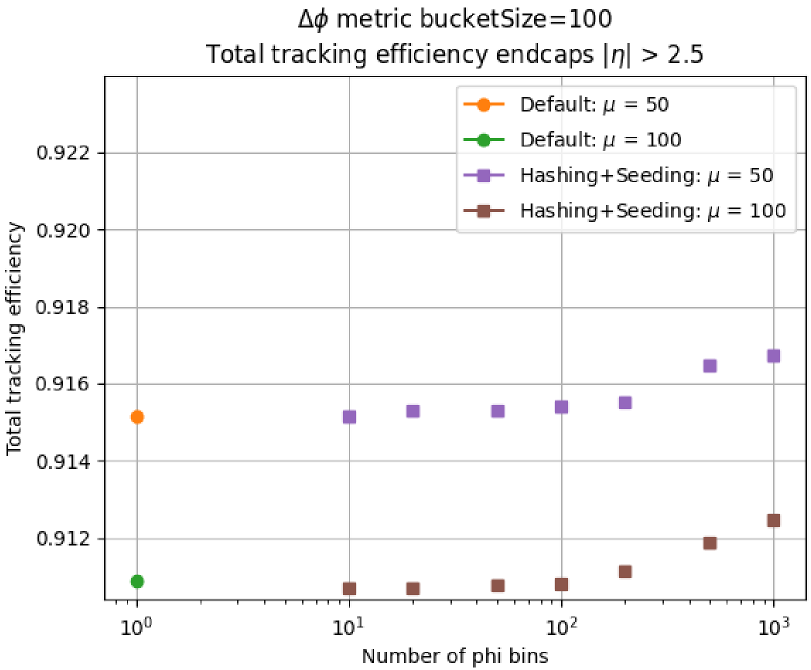


Small loss of efficiency

Hashing φ bins: Efficiency (detailed)

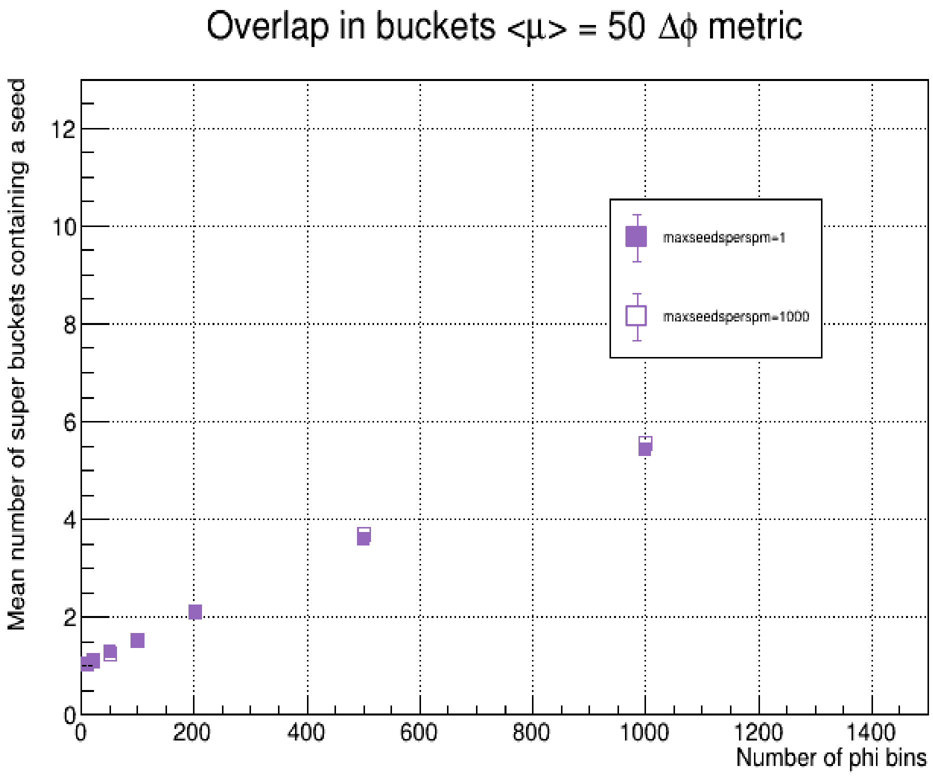


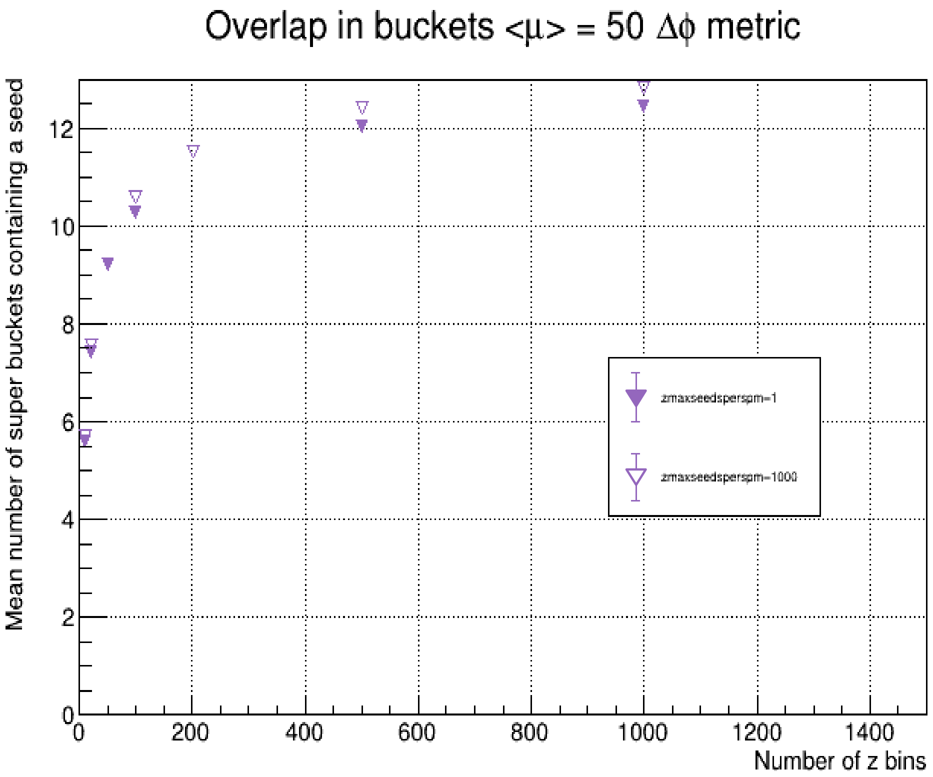
Drop of efficiency in the barrel



Better efficiency in the endcaps

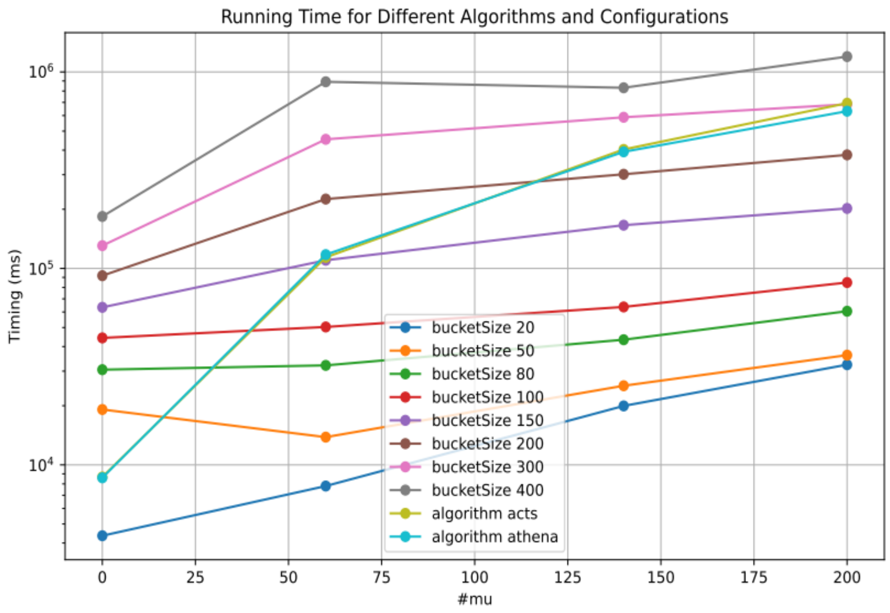
Overlap in buckets

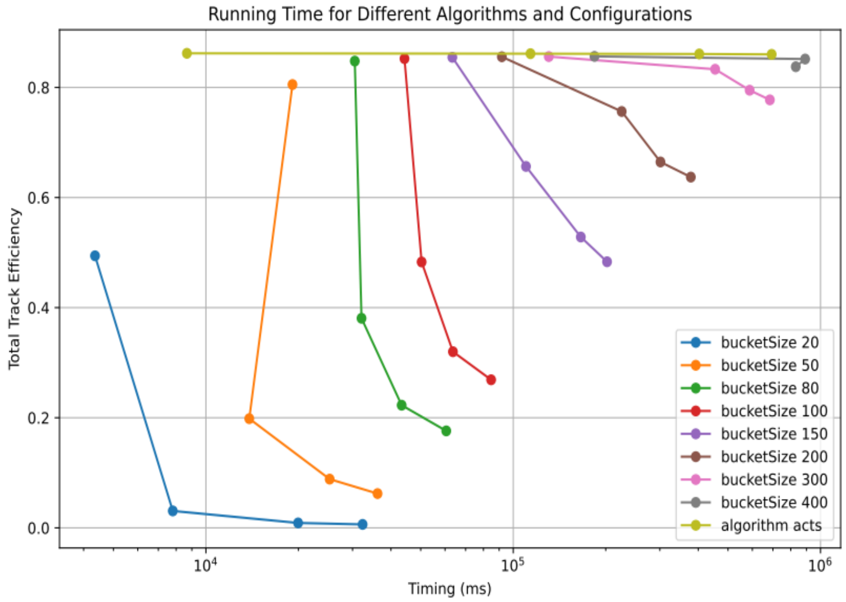




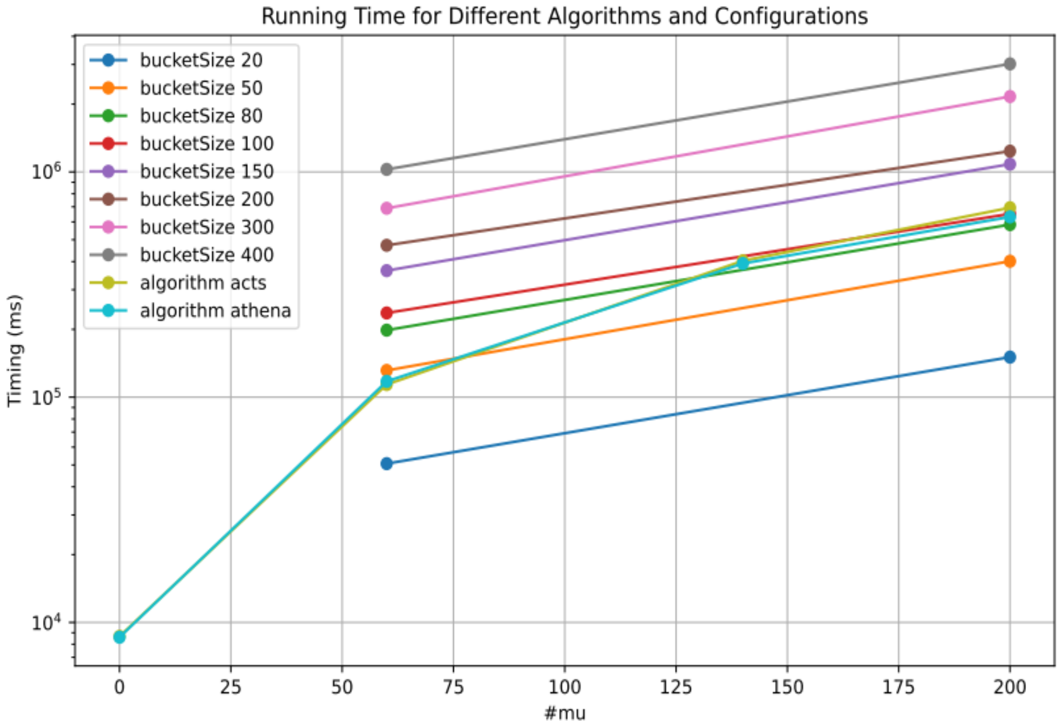
Less overlaps between buckets with φ binning

Some timing plots: Δφ





Some timing plots: ΔR



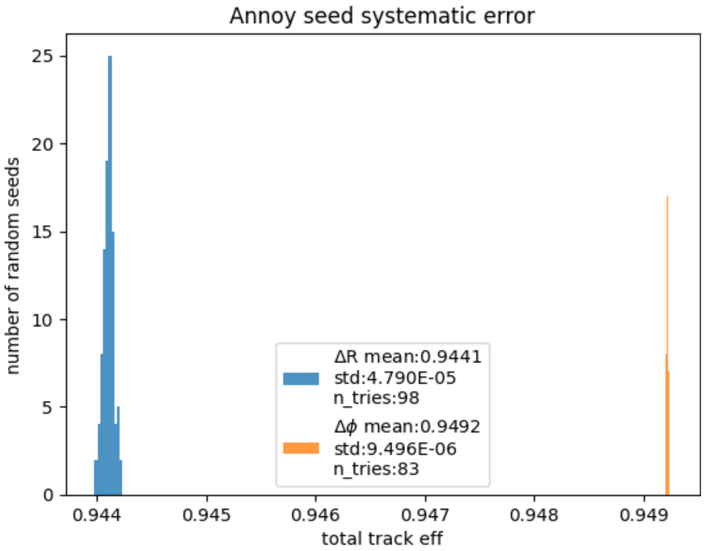
Seed finder configuration

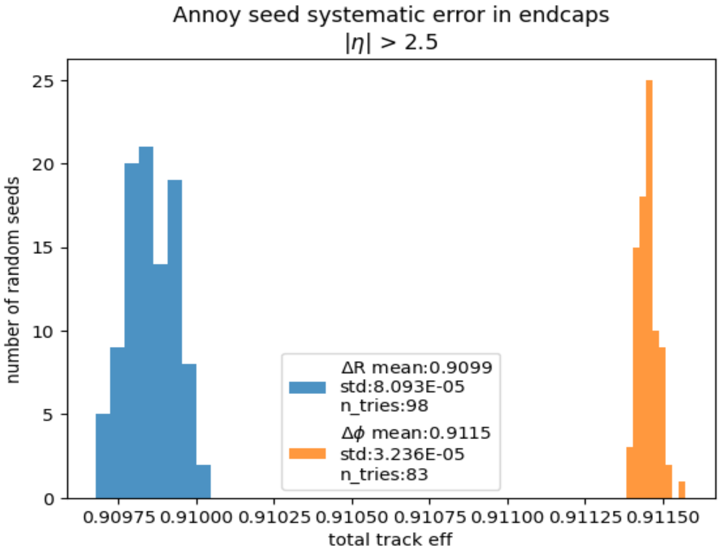
```
SeedfinderConfigArg = SeedfinderConfigArg(
        r=(None, 200 * u.mm),  # rMin=default, 33mm
        deltaR=(1 * u.mm, 60 * u.mm),
        collisionRegion=(-250 * u.mm, 250 * u.mm),
        z=(-2000 * u.mm, 2000 * u.mm),
        maxSeedsPerSpM=1,
        sigmaScattering=5,
        radLengthPerSeed=0.1,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

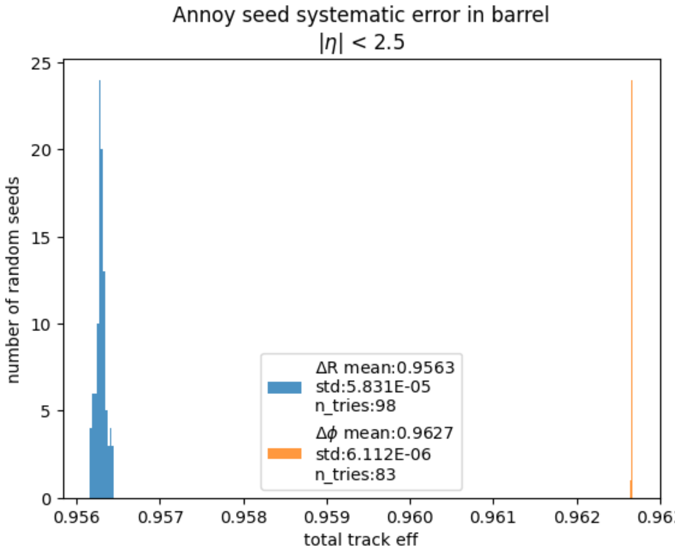
MaxSeedsPerSpM cut

- **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

Annoy random seed systematic error





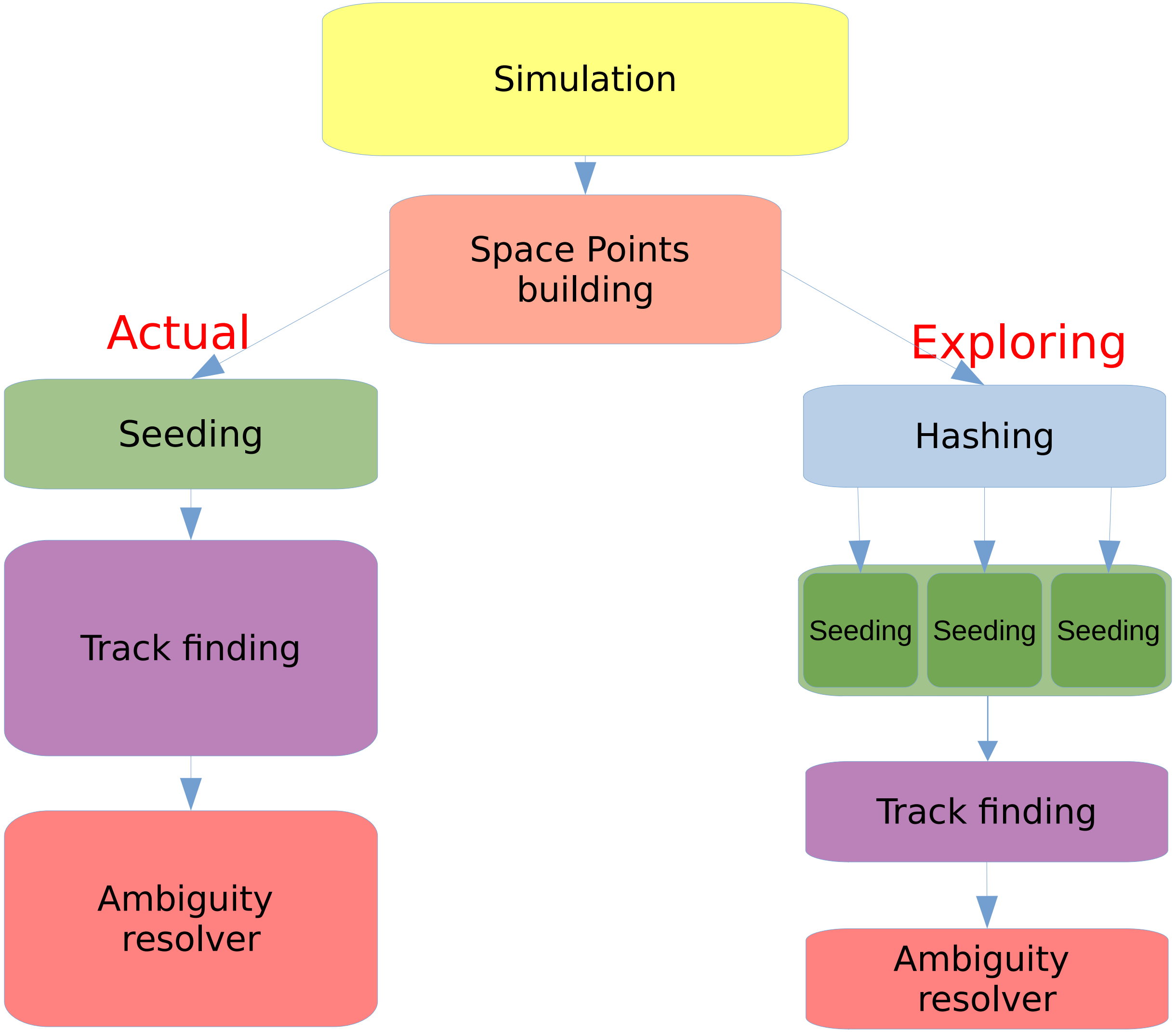


1000 events
in each try

BucketSize: 100
Mu: 50

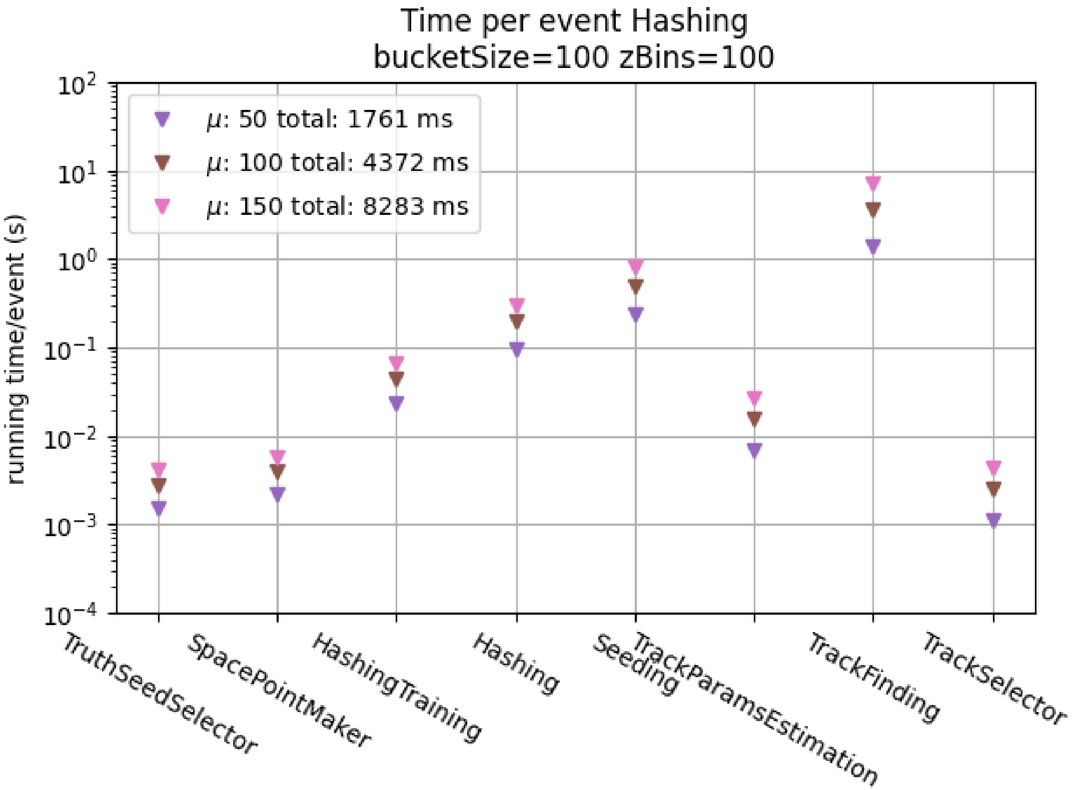
Δφ is better

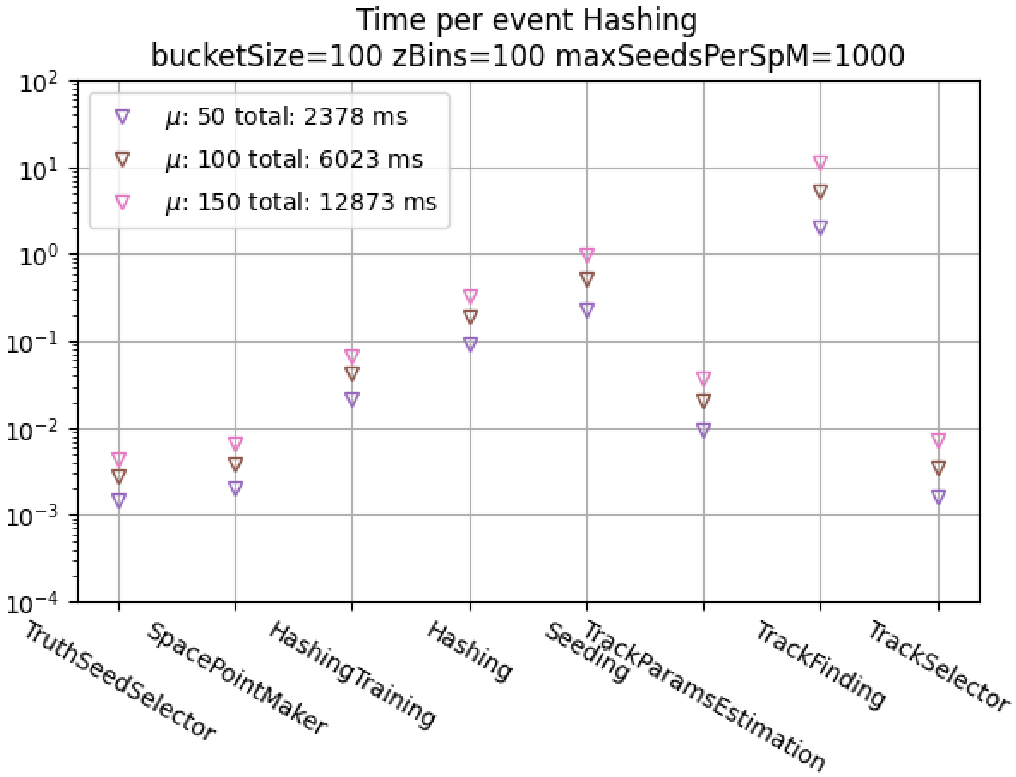
Approaches



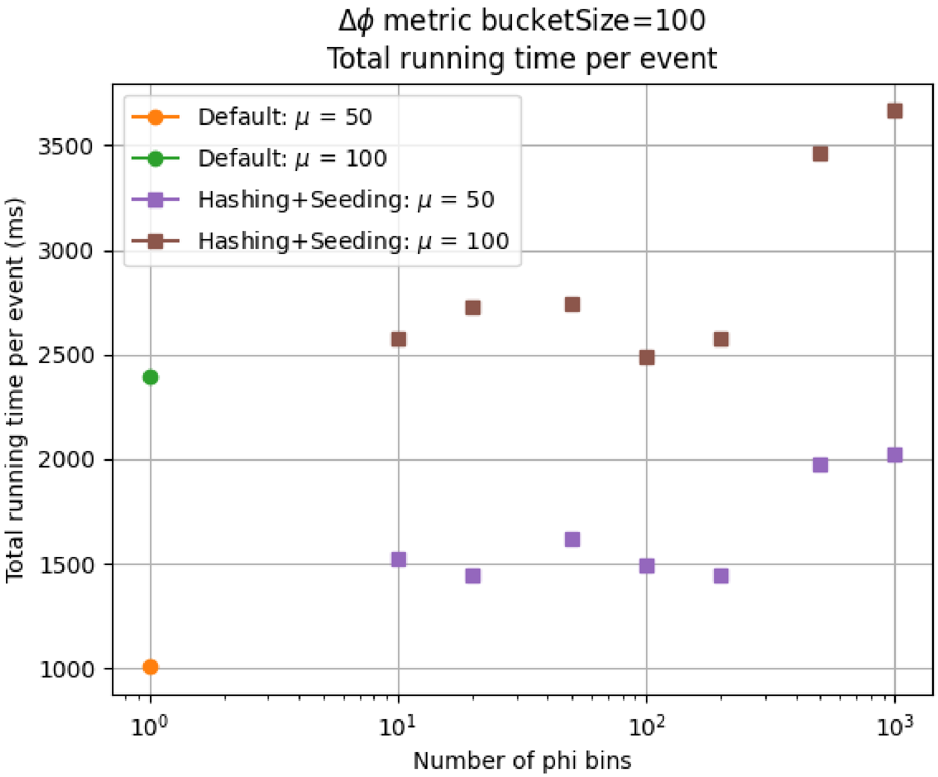
- Seeding parallelization
- Hashing groups space points into buckets
- Hashing reduces the number of space points at a time (focus on relevant space points) → less seeds per bucket

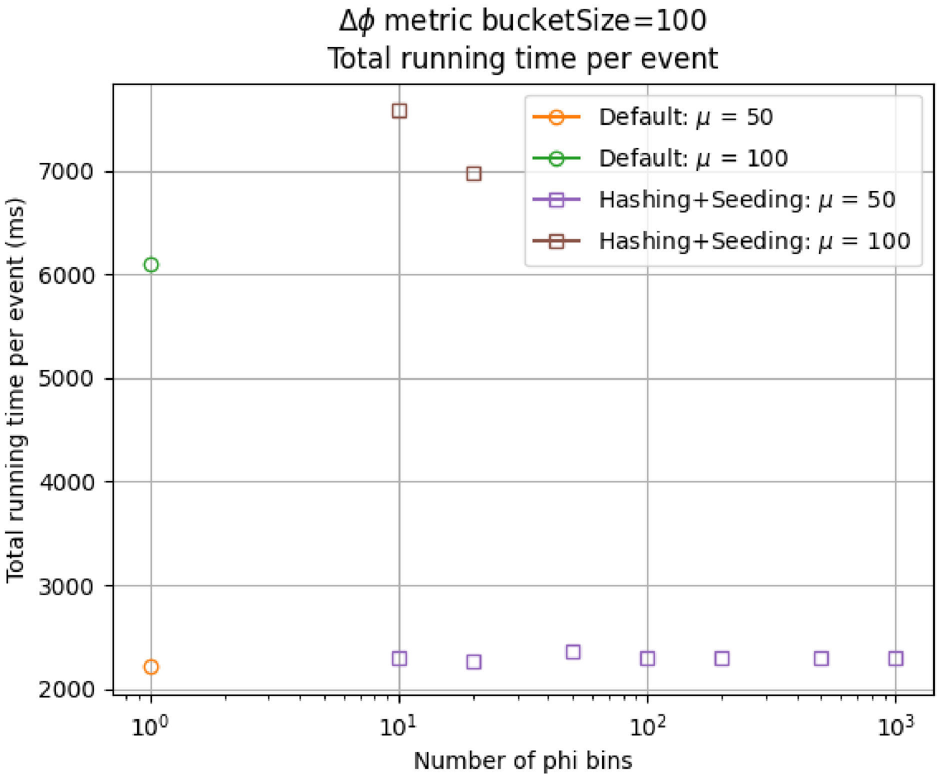
Running time no cut



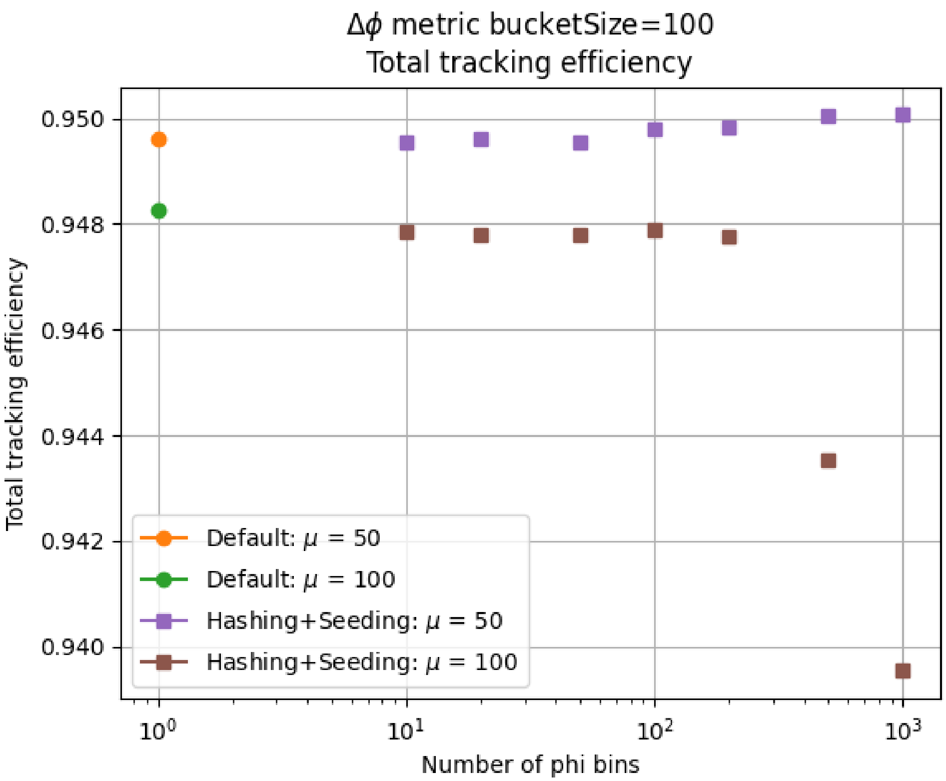


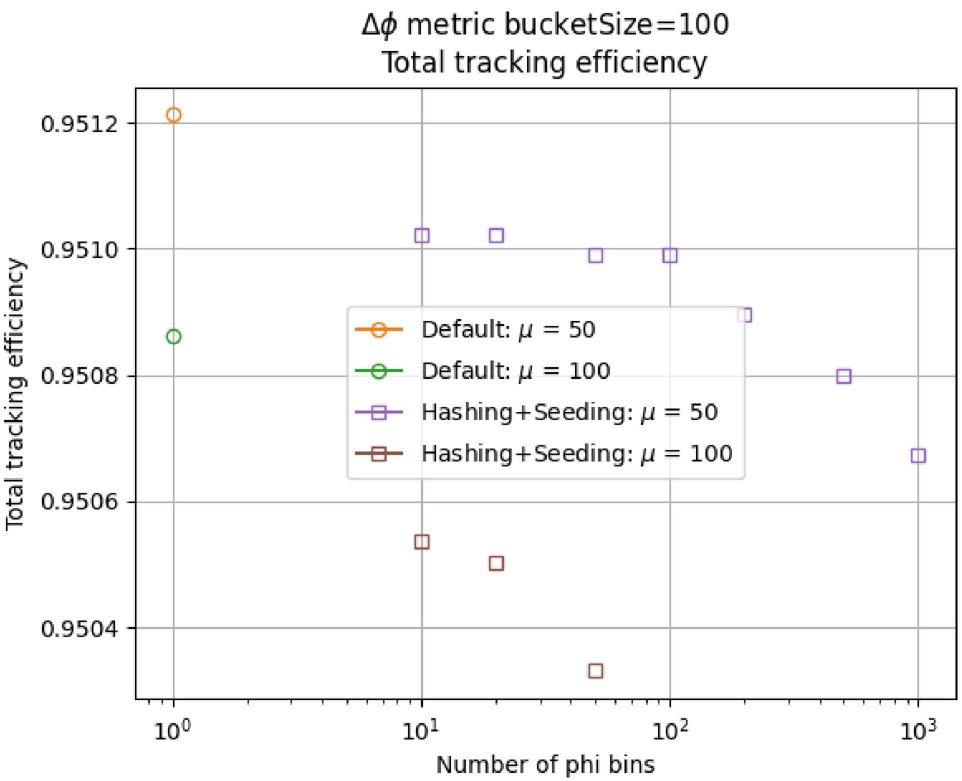
Phi bins: Timing



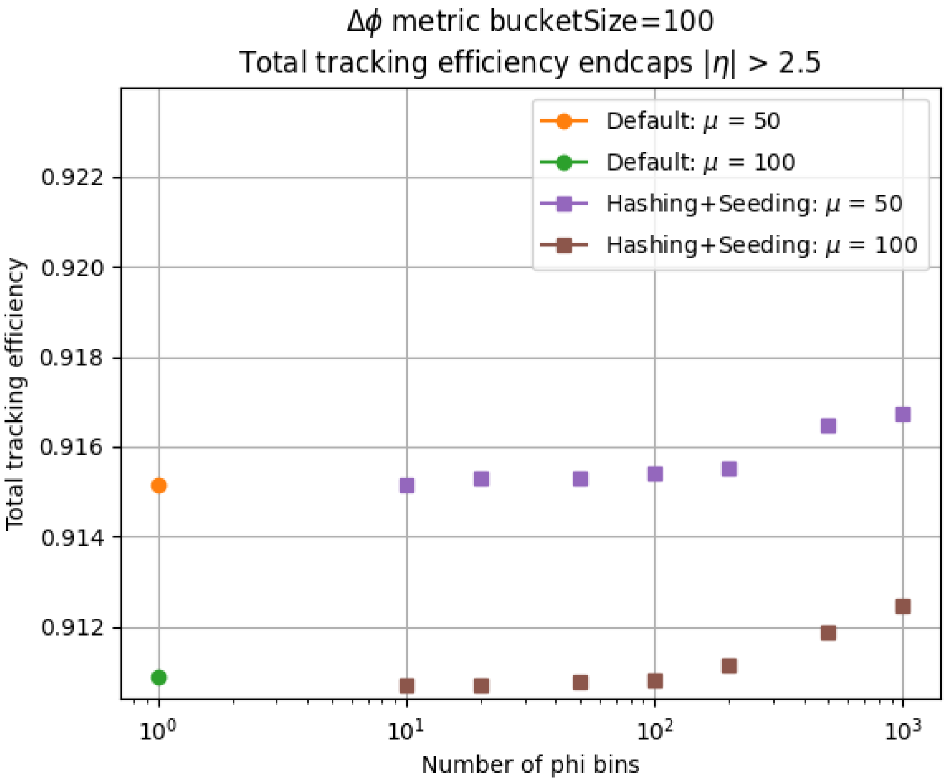


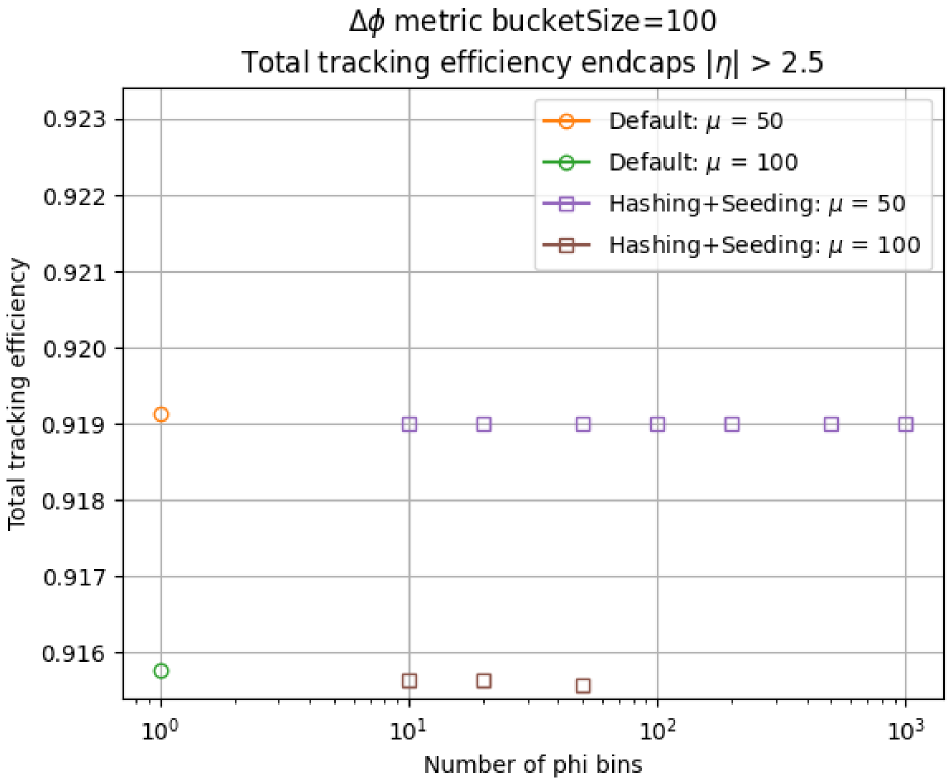
Phi bins: Tracking efficiency



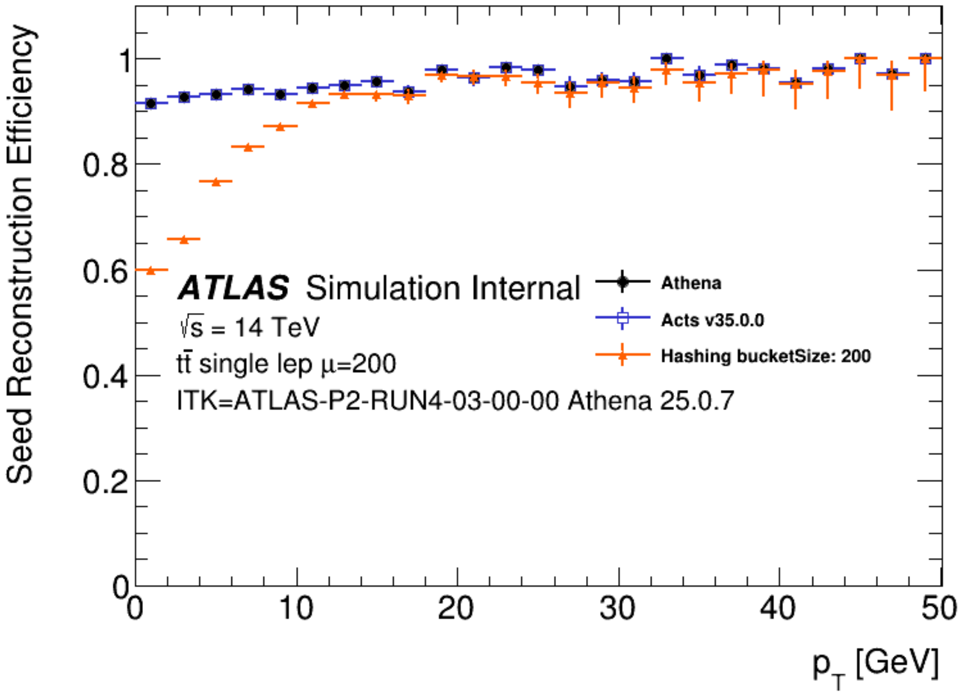


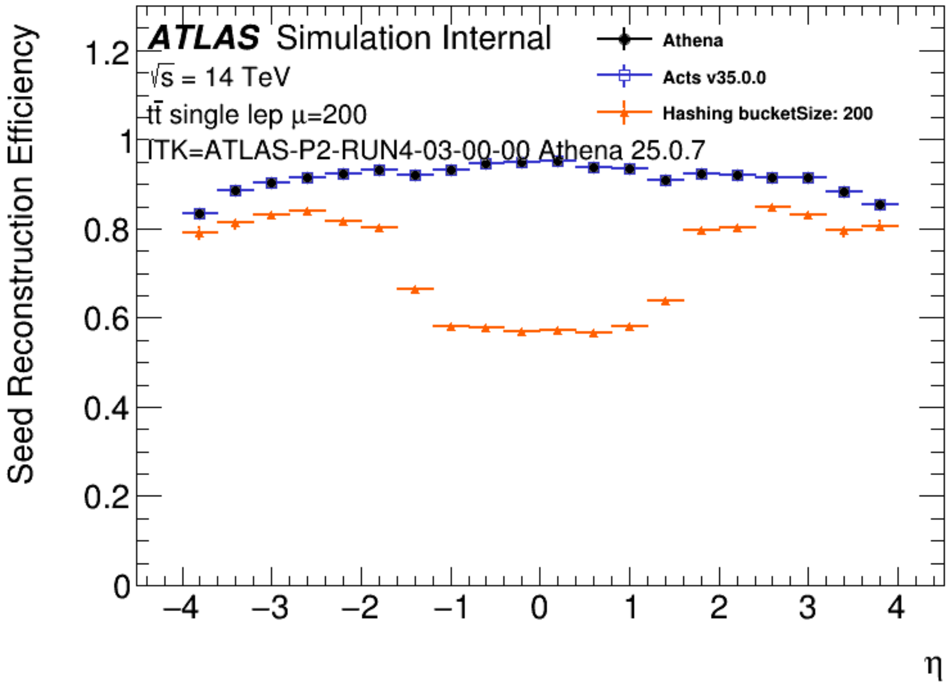
Phi bins: Tracking efficiency





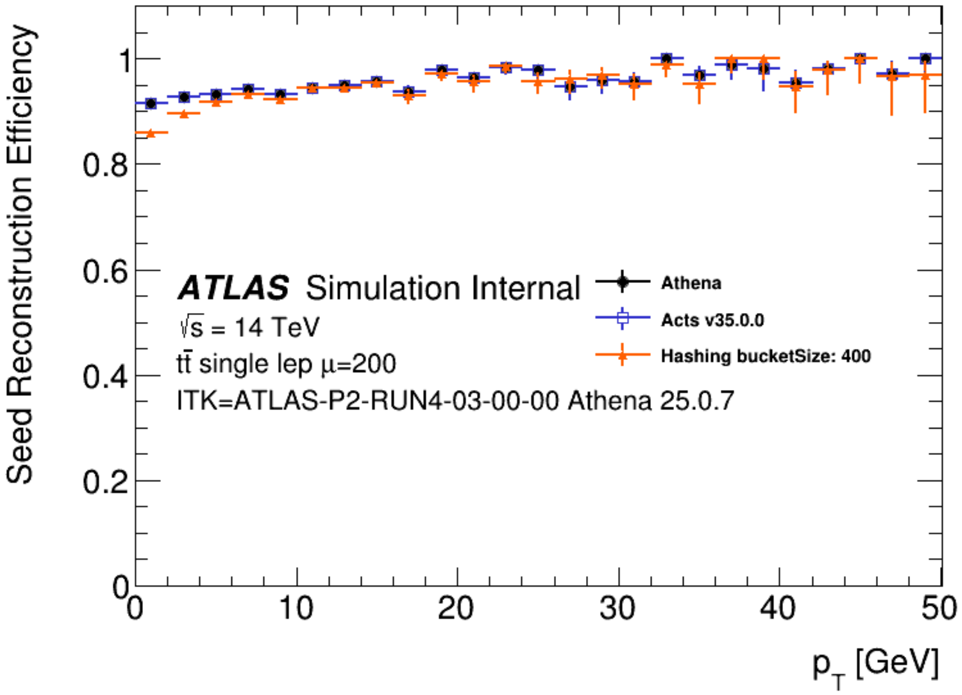
Hashing μ = 200

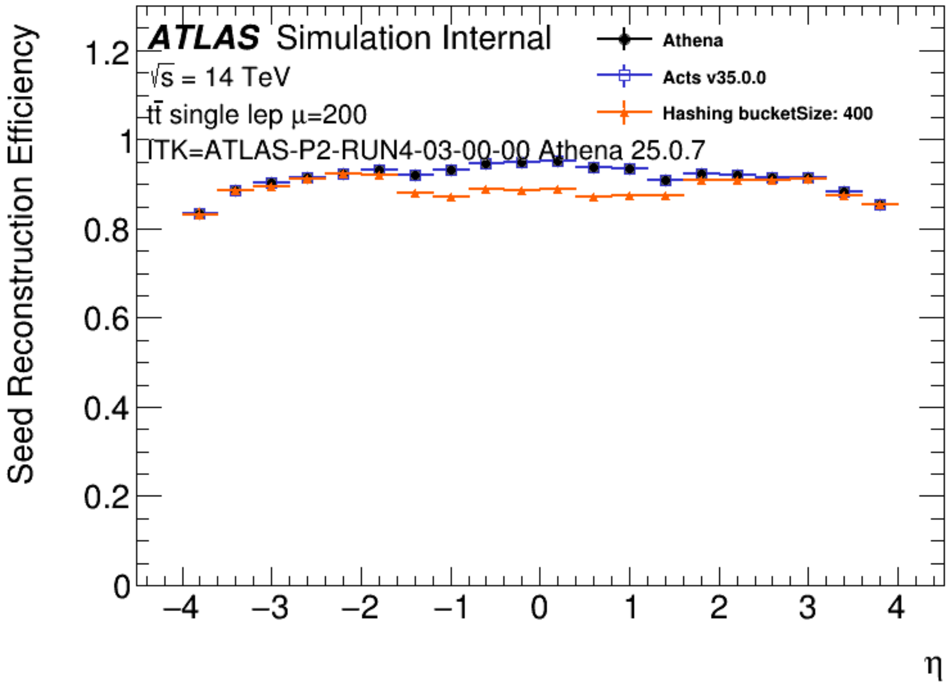




1000 events

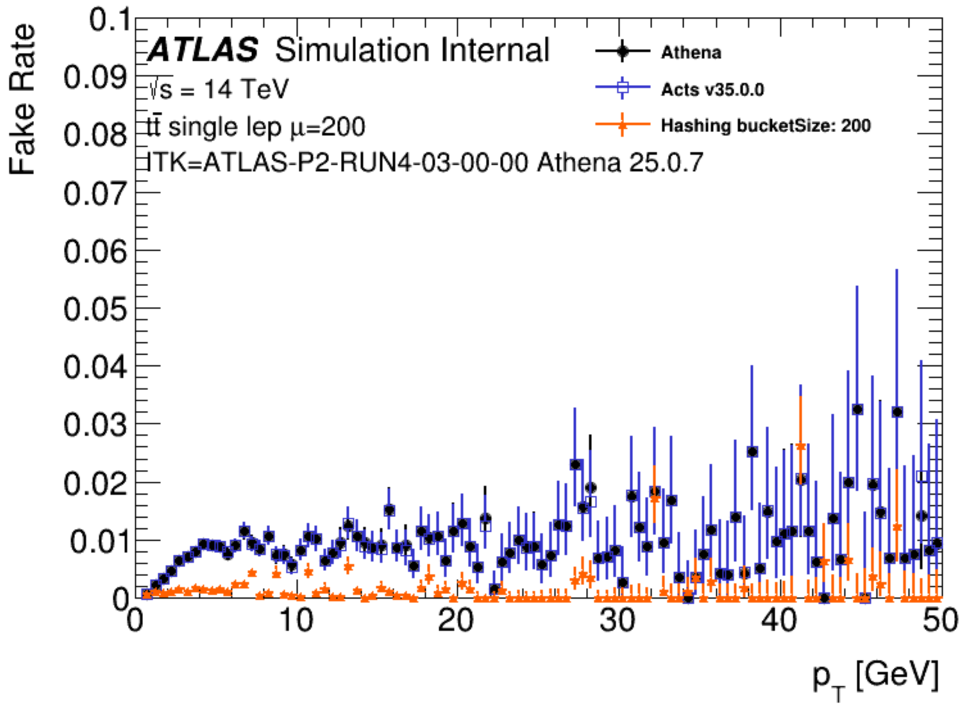
Hashing μ = 200

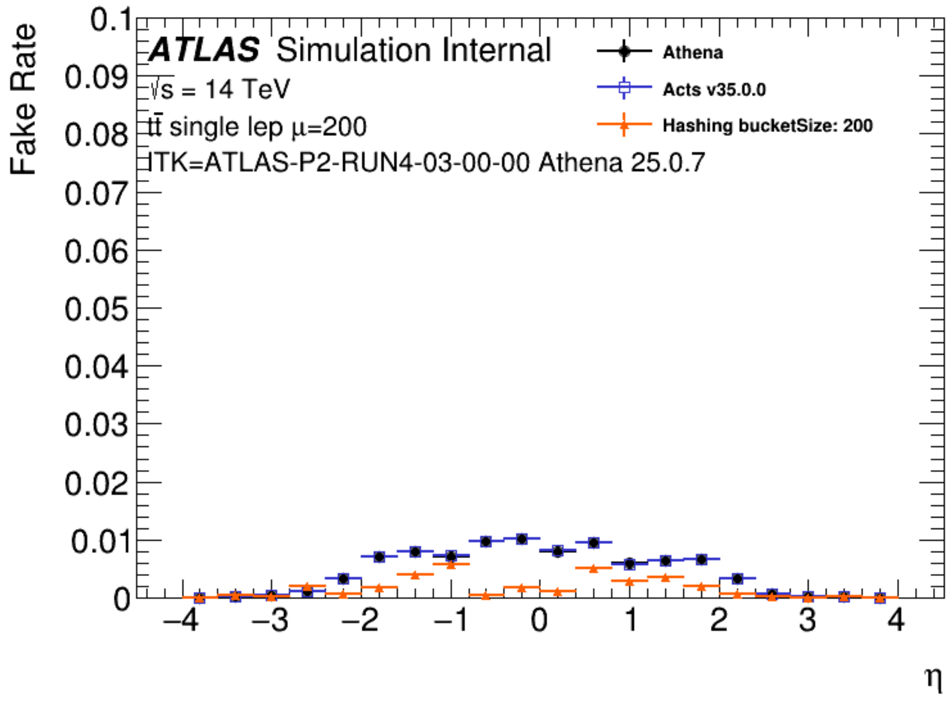




1000 events

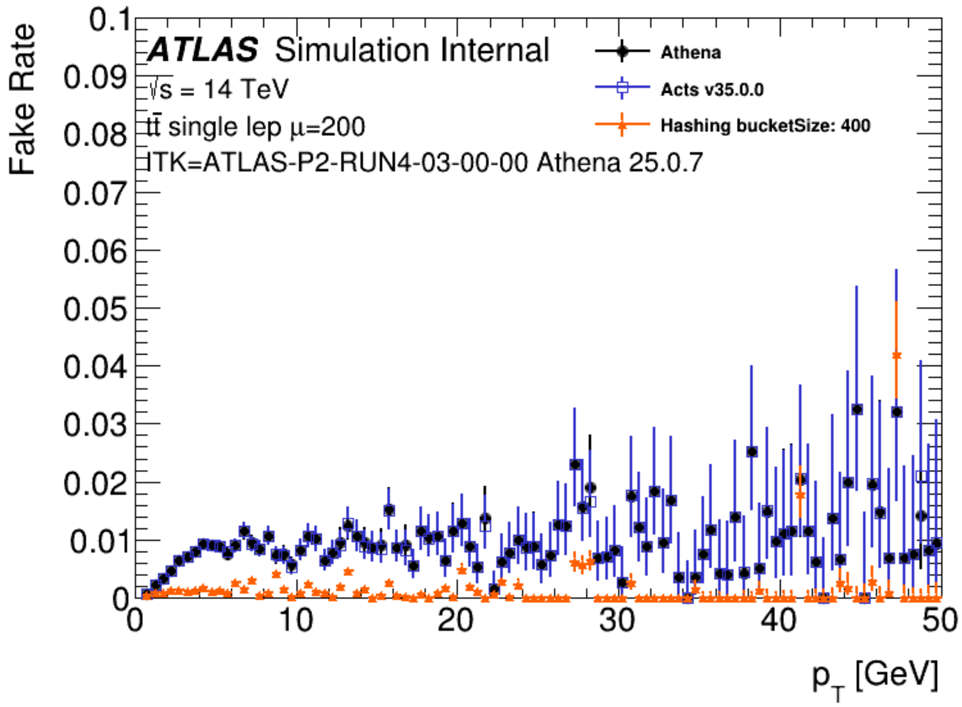
Hashing μ = 200

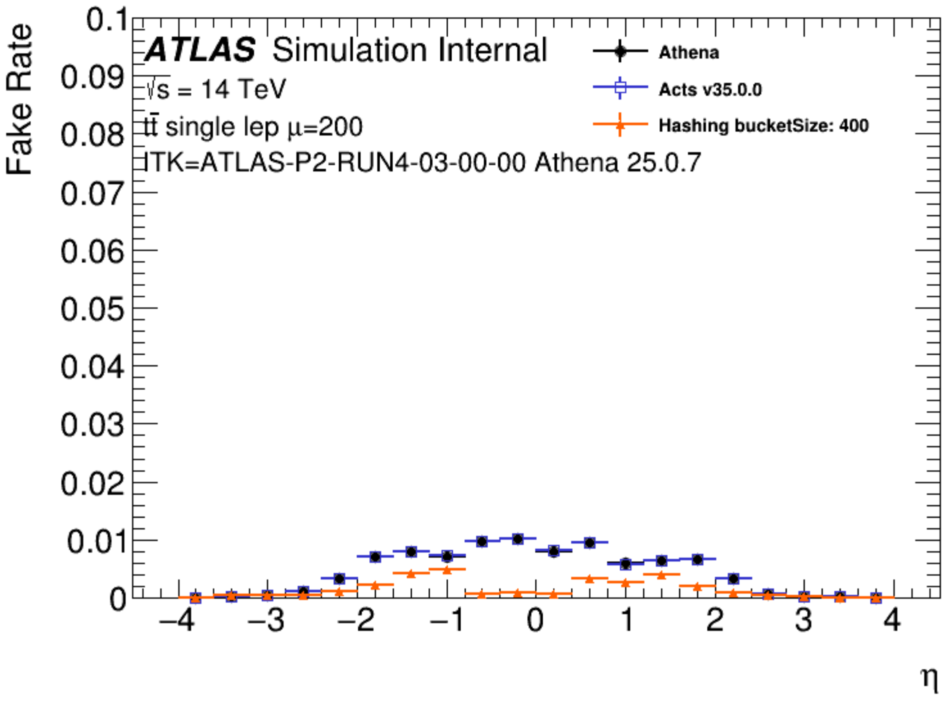




1000 events

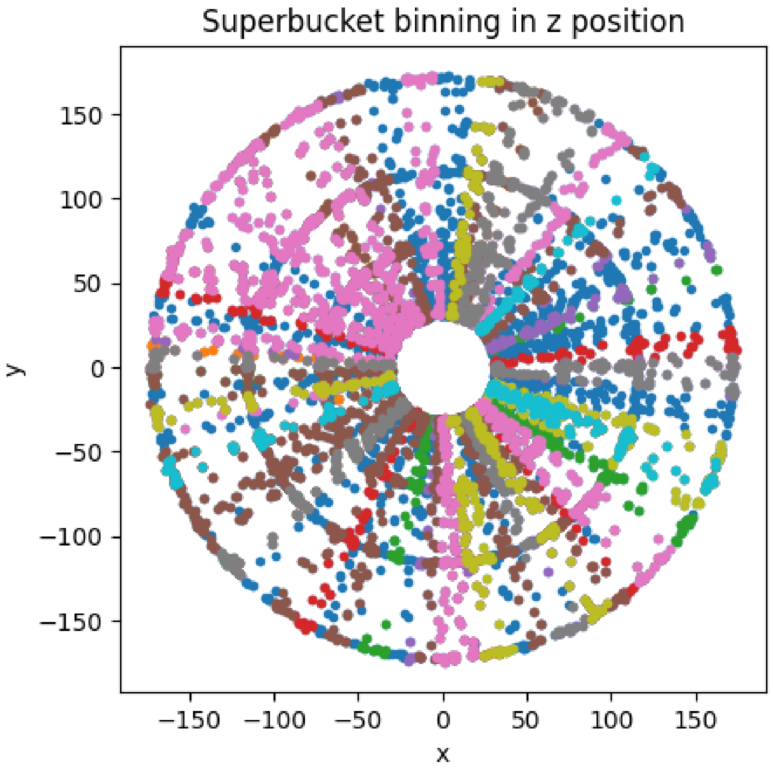
Hashing μ = 200

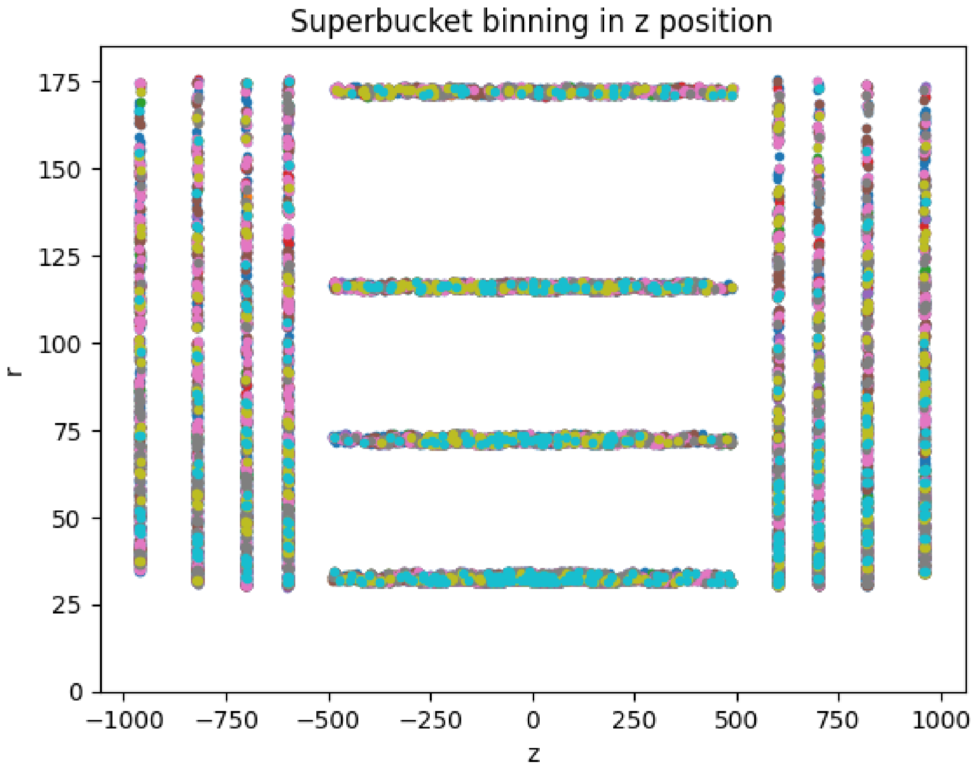




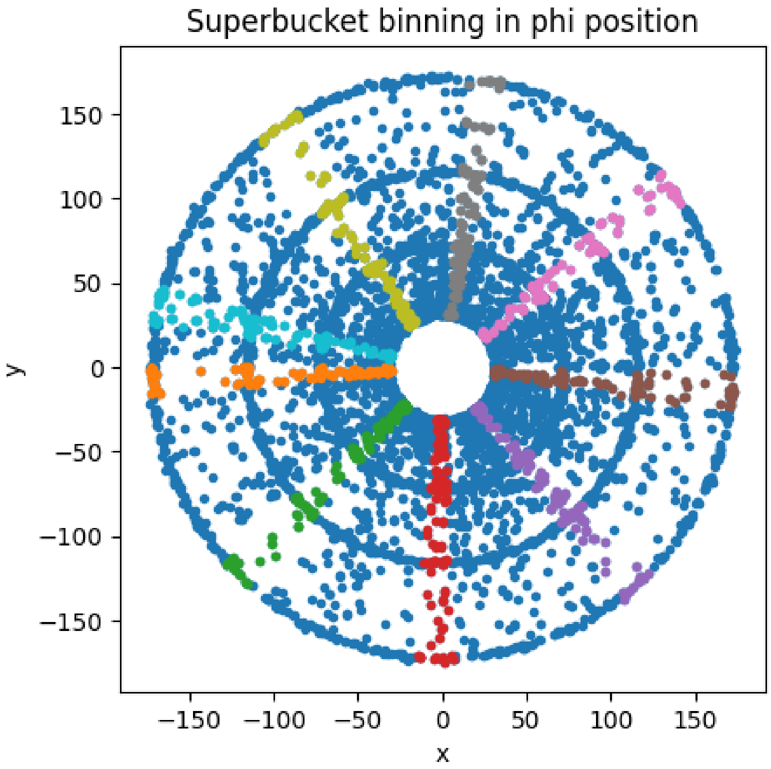
1000 events

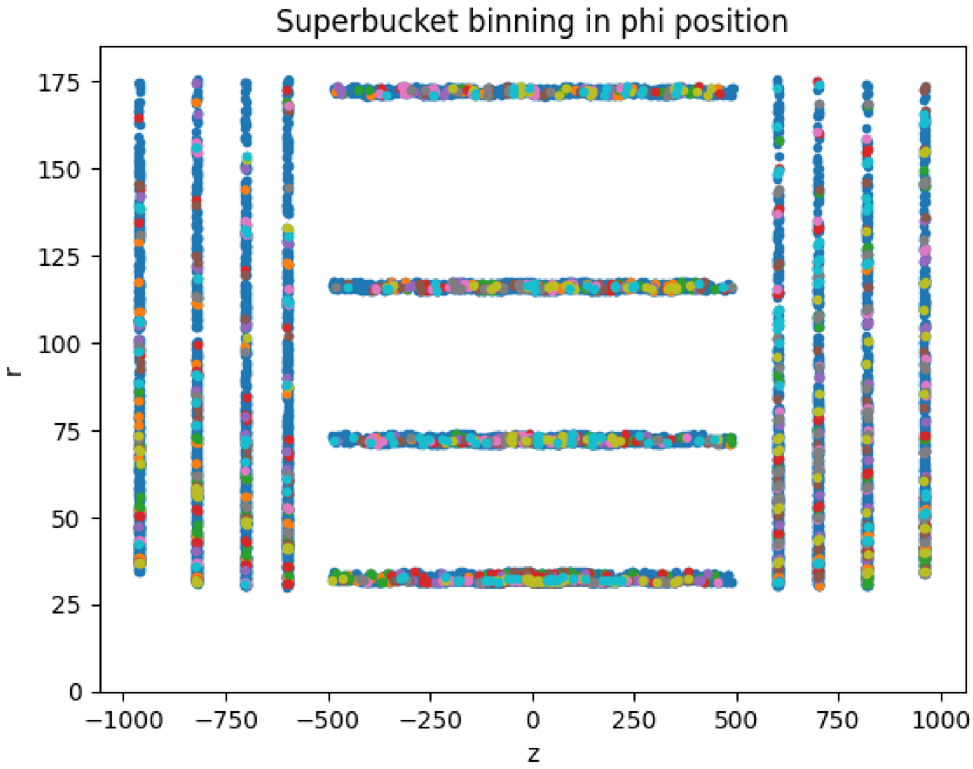
Superbucket binning in Z position





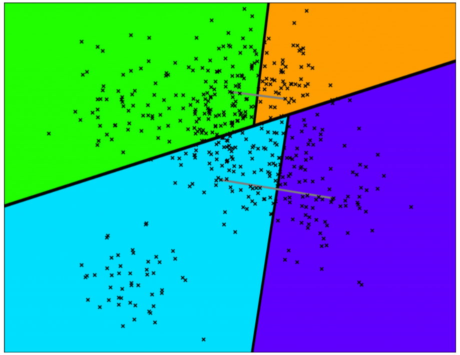
Superbucket binning in Phi position

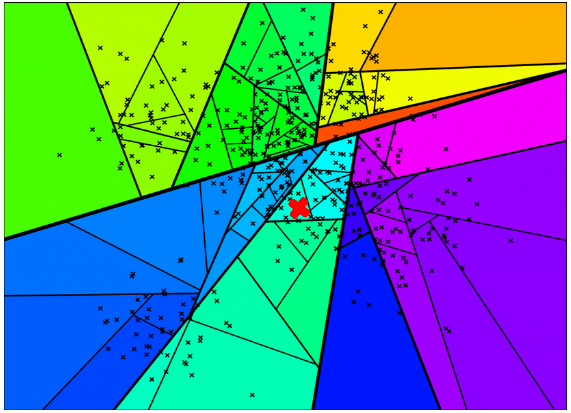




Annoy training

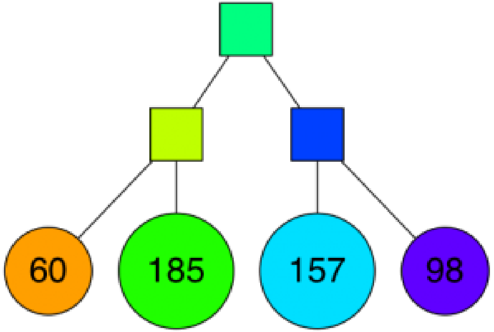
Space separation

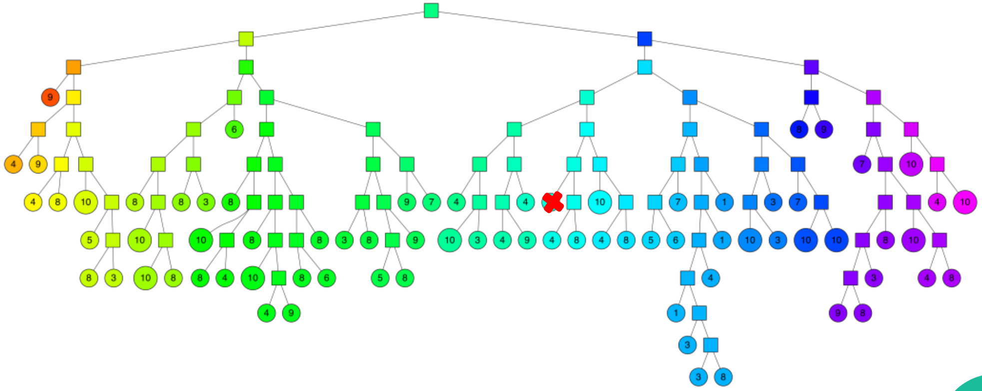




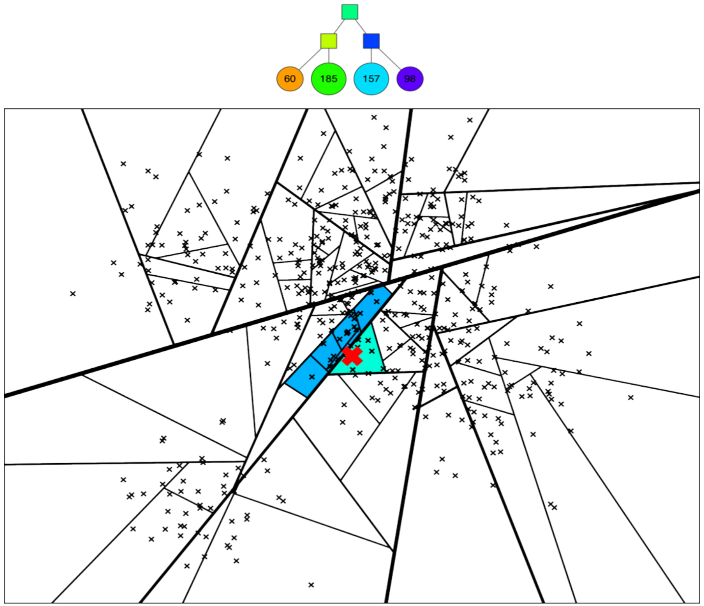
Takes two random points iteratively

Corresponding binary tree

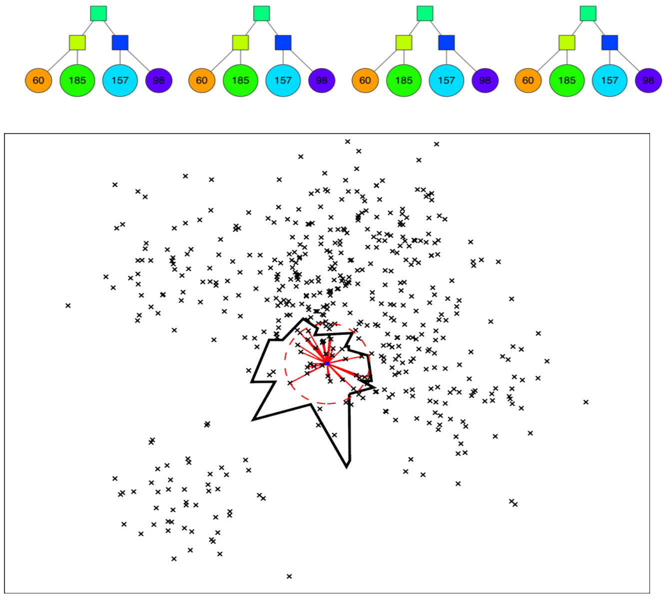




Annoy query



Approximation



Merge neighbor subspaces

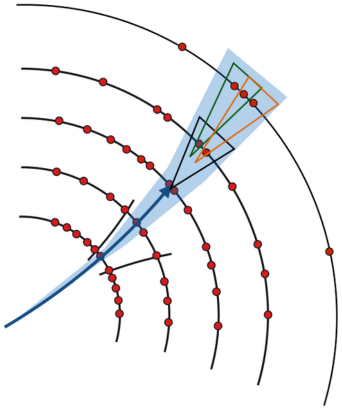
Union of trees' subspace

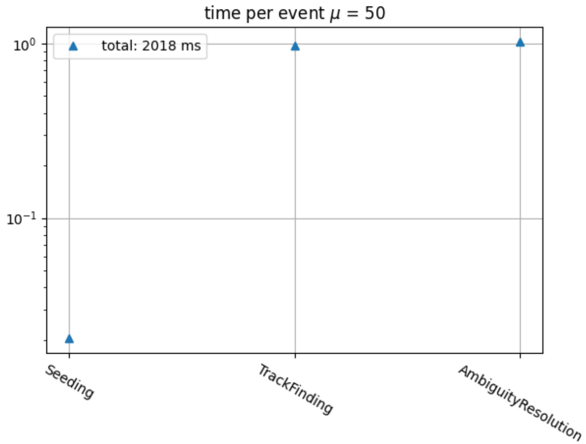
- Annoy tuning parameters: number of neighbors, number of trees, metric used, features used, number of subspace to look at

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

Seeding: Skipping triplets check with sets

Overlap indicator

- **Event 98: Hashing mu=50 bucketSize=100**
- **9860 Space Points → ~100.000.000 possible doublets**

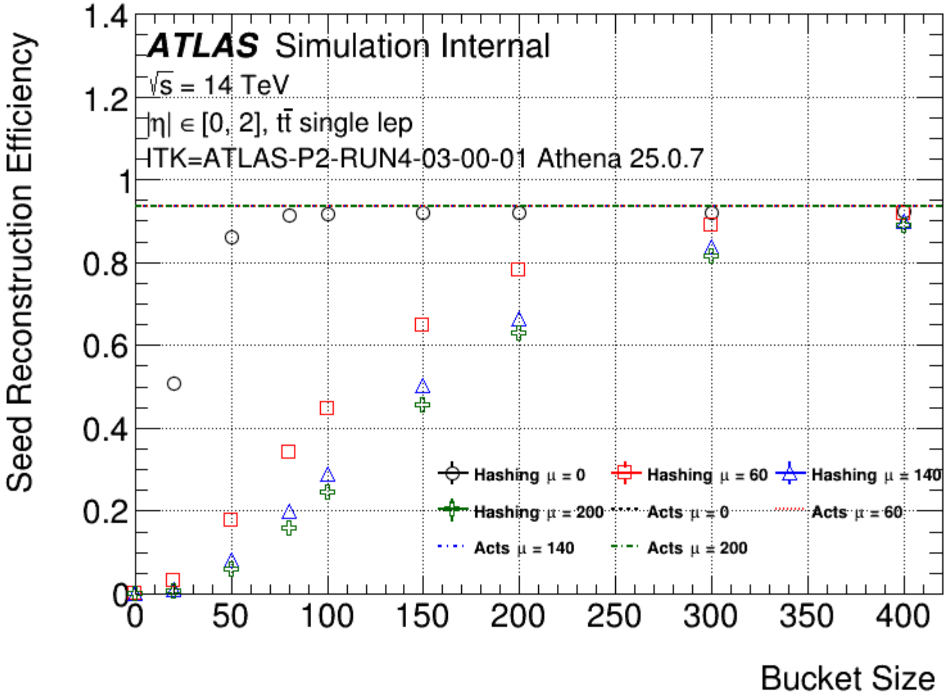
Set name	Set size	nSkipped	Ratio
Bad bottom	**24.433.199**	322.132.498	13,18
Good bottom	**3.592.664**	63.294.324	17,62
Bad top	**30.363.102**	392.248.454	12,92
Good top	**4.973.975**	91.166.619	18,33
Triplets	**18.204.058**	269.635.750	14,81
Seeds	**5.623**	x	x

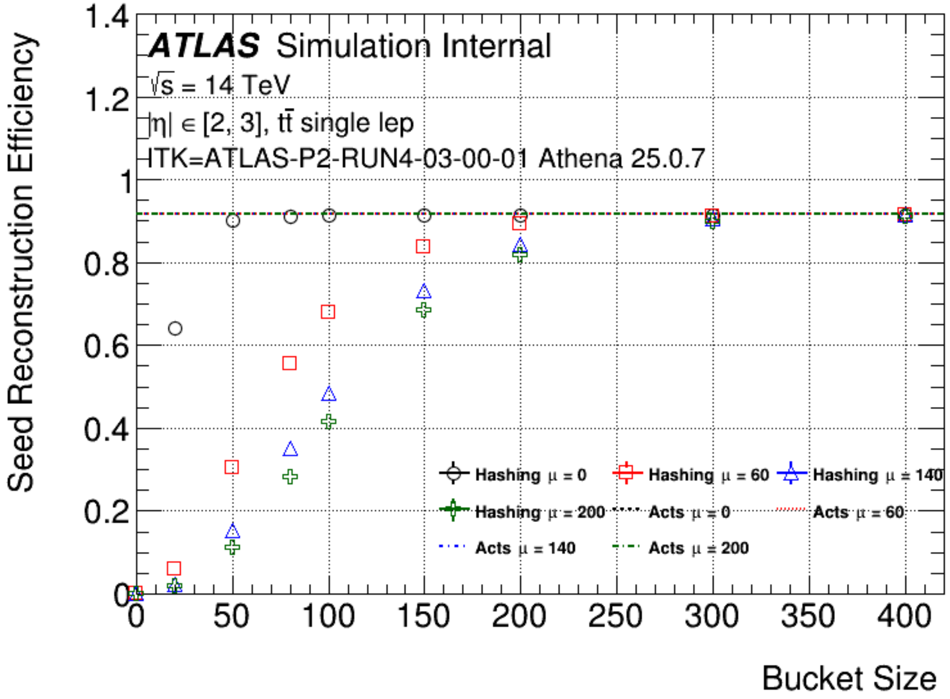
Total running time x1.5

Bucket Size Δφ: η

InnerTracker

WARNING: not only first layer selected



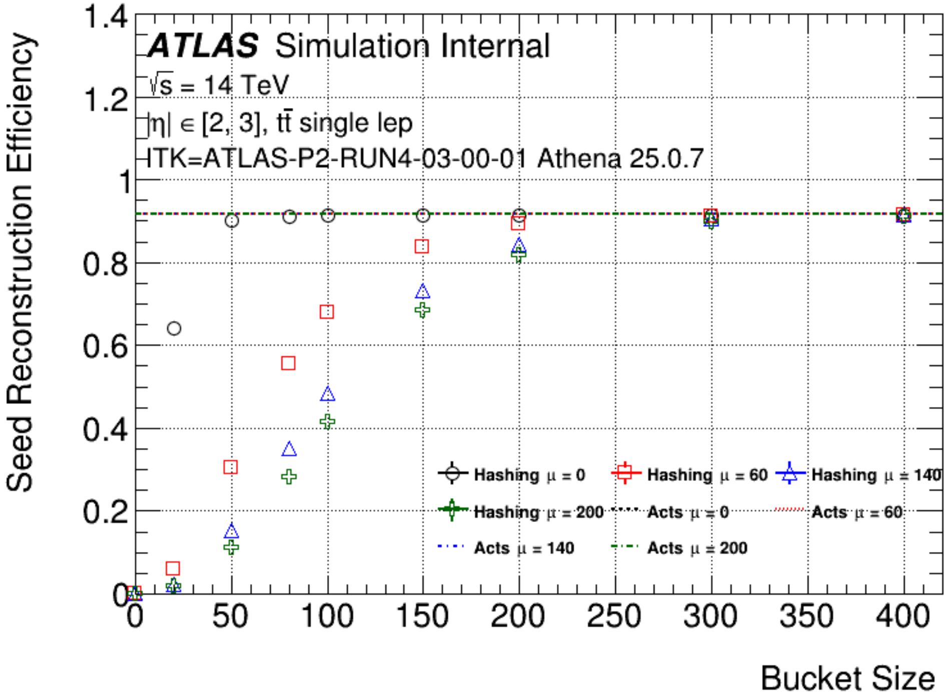


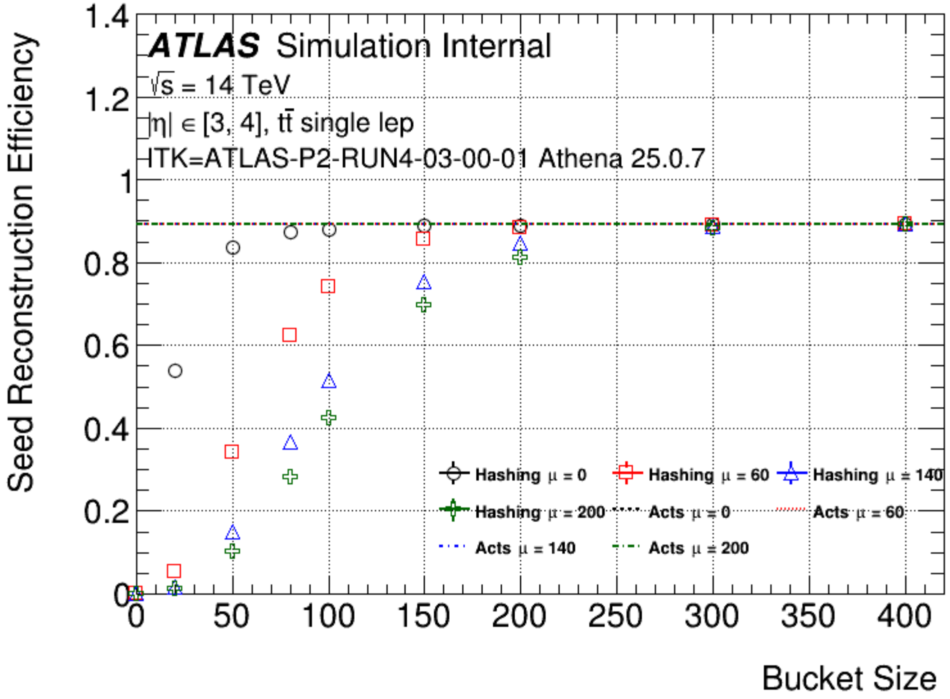
1000 events

Bucket Size Δφ: η

InnerTracker

WARNING: not only first layer selected



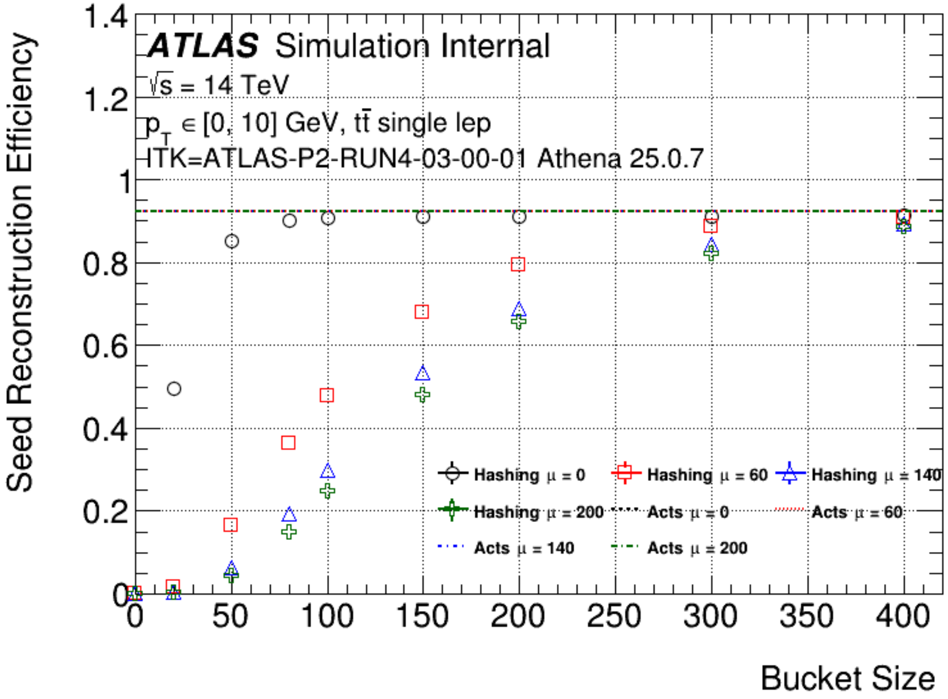


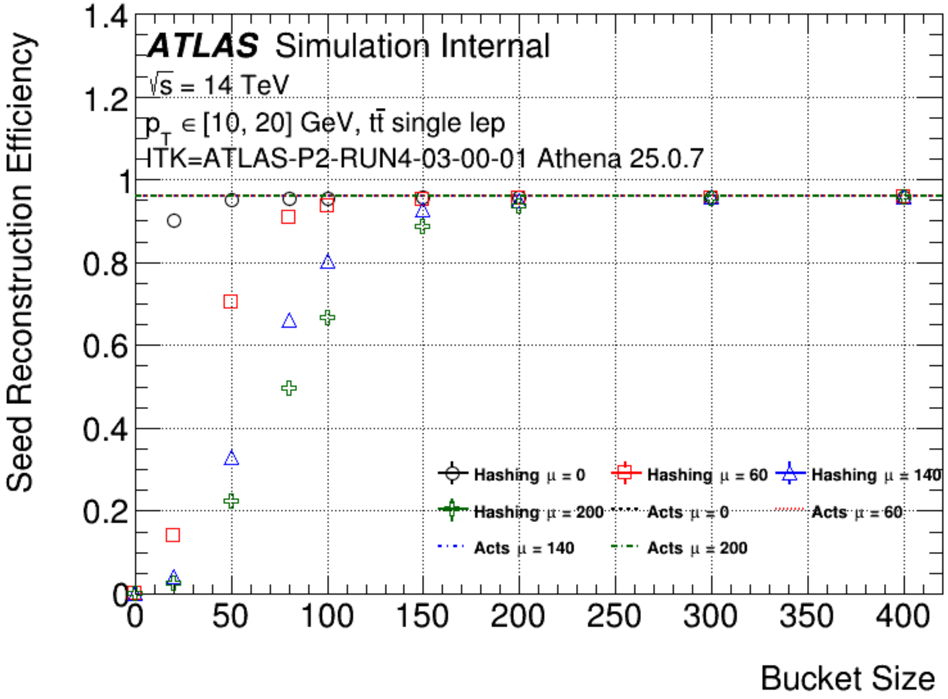
1000 events

Bucket Size Δφ: pT

InnerTracker

WARNING: not only first layer selected



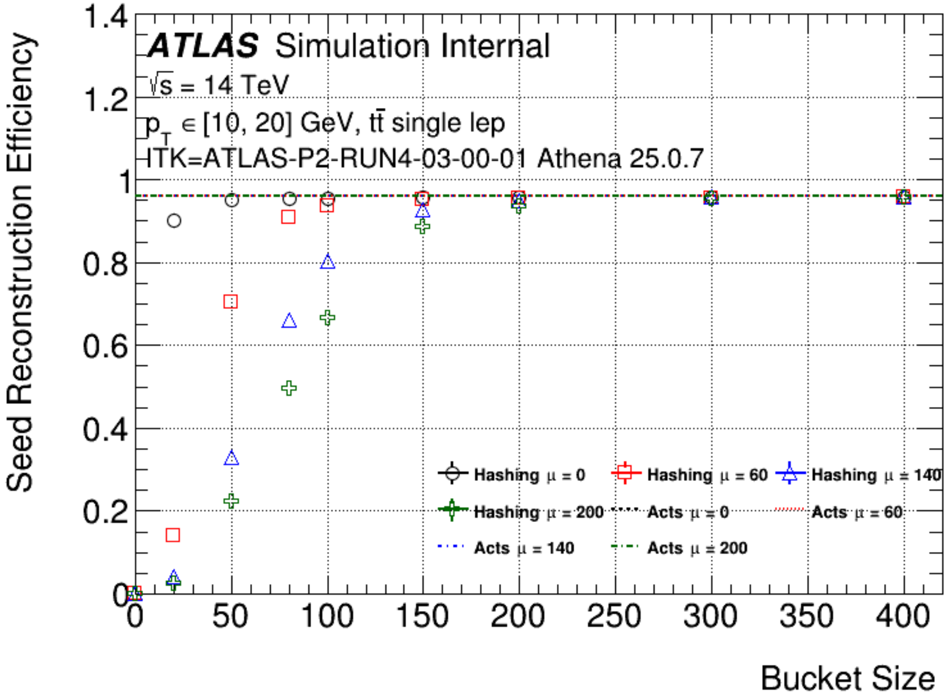


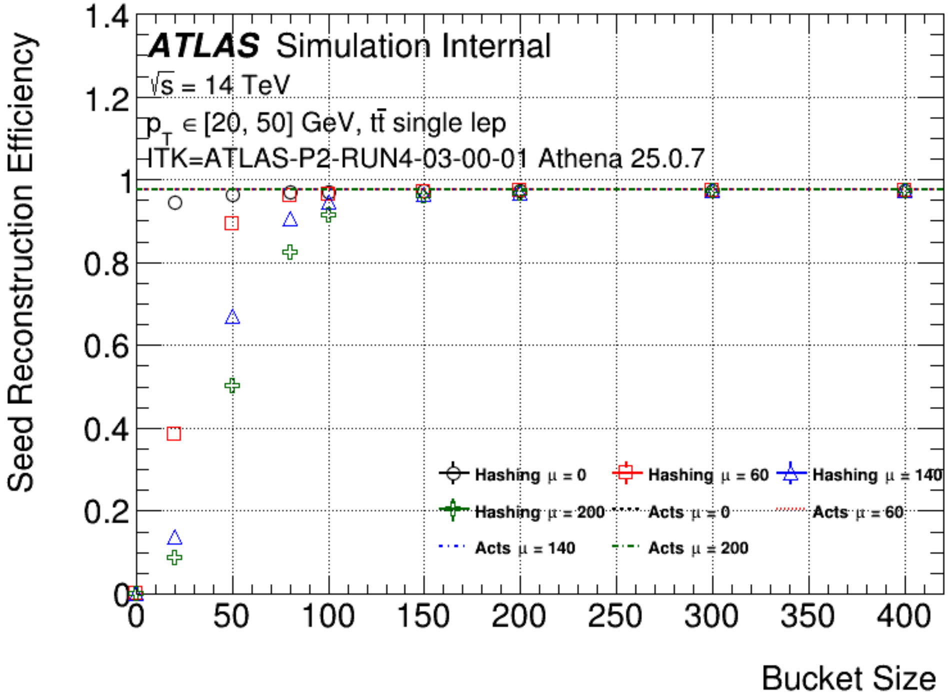
1000 events

Bucket Size Δφ: pT

InnerTracker

WARNING: not only first layer selected



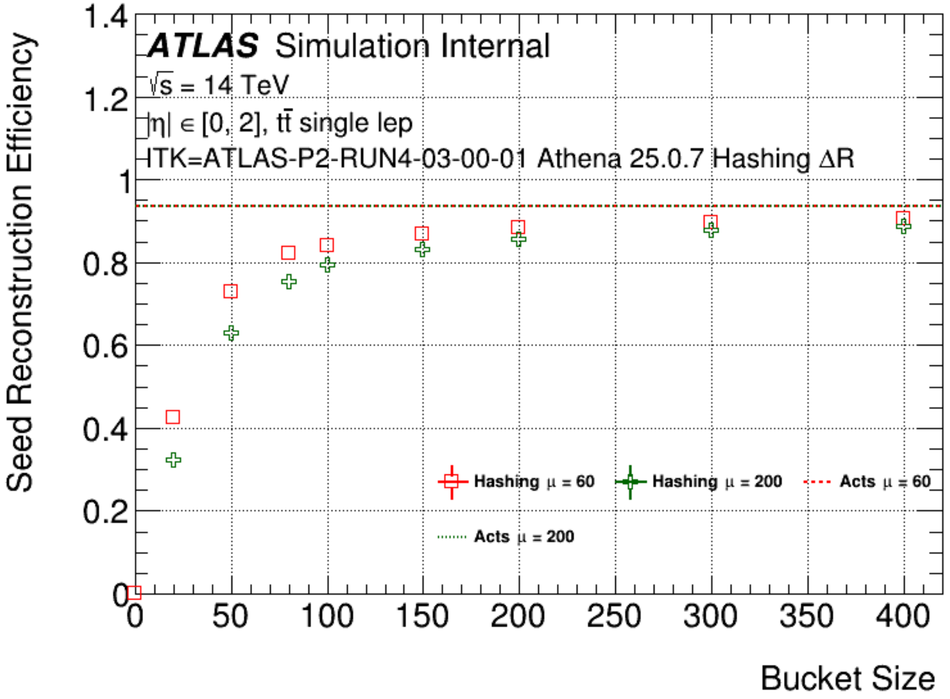


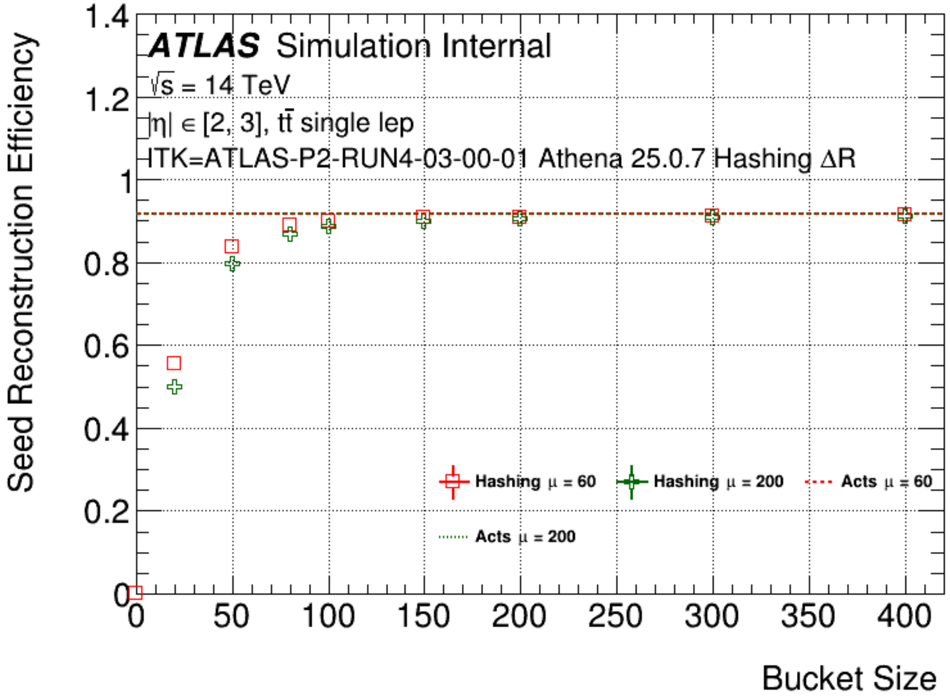
1000 events

Bucket Size ΔR: η

InnerTracker

WARNING: not only first layer selected



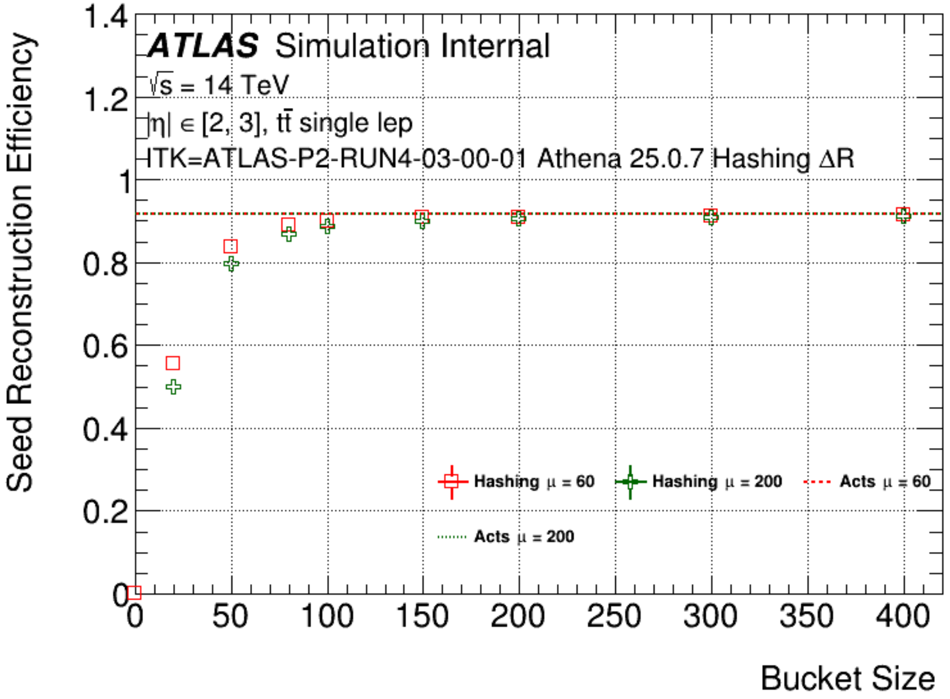


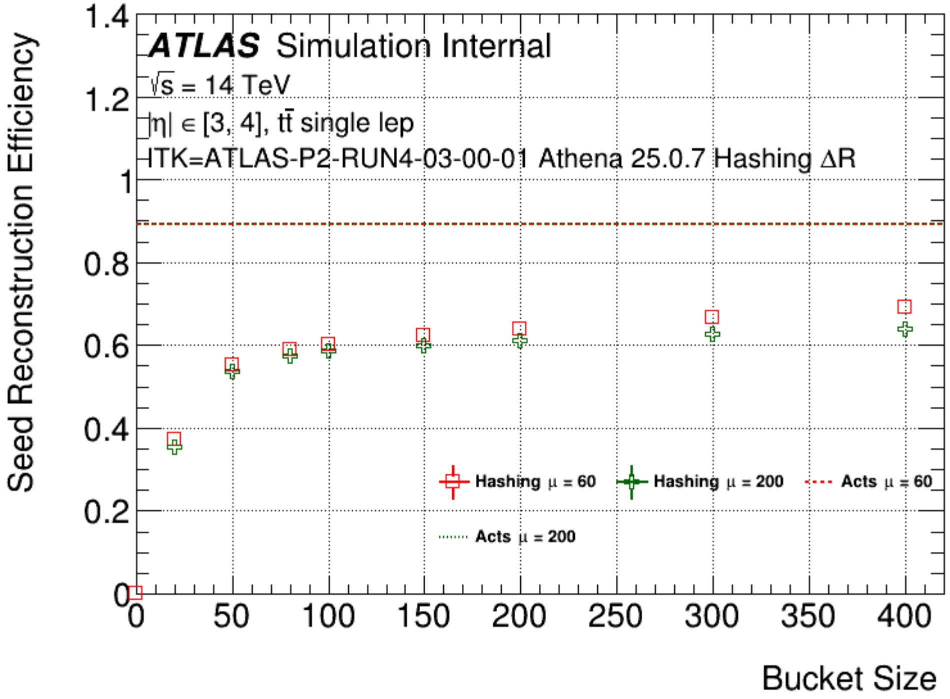
1000 events

Bucket Size ΔR: η

InnerTracker

WARNING: not only first layer selected



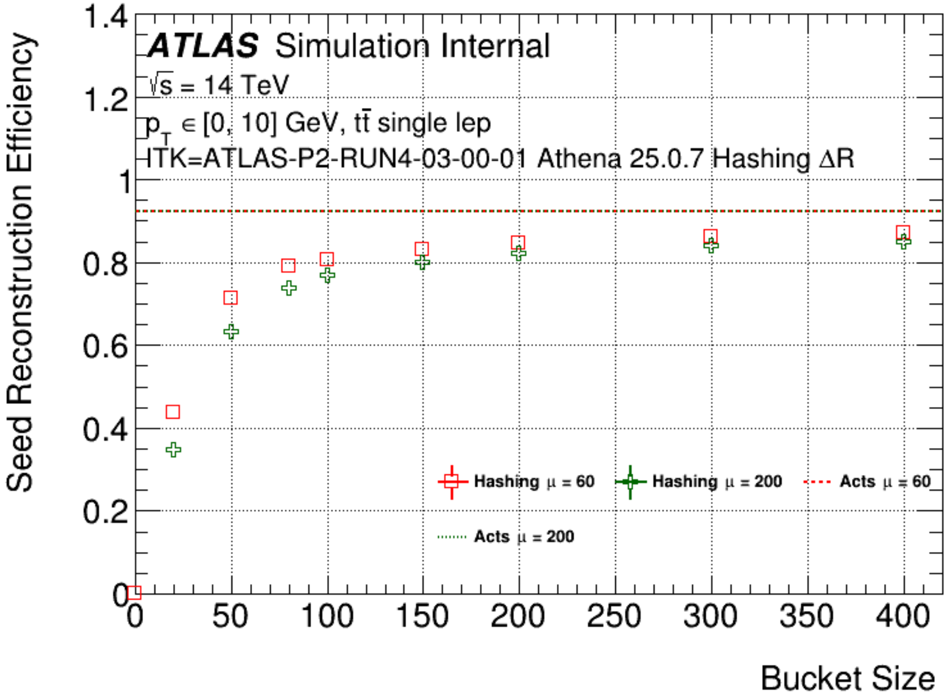


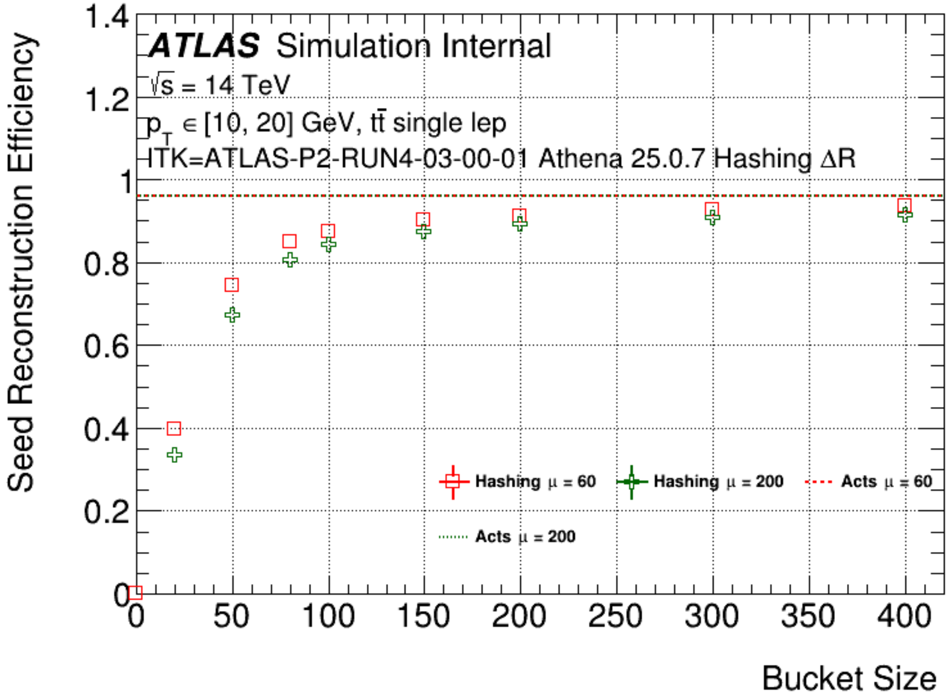
1000 events

Bucket Size ΔR: pT

InnerTracker

WARNING: not only first layer selected



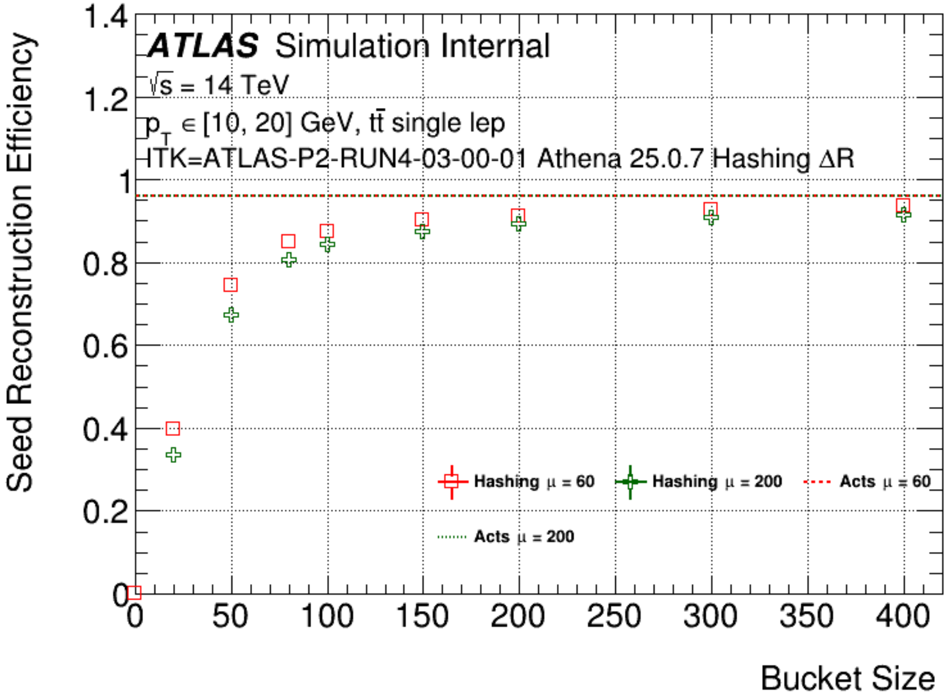


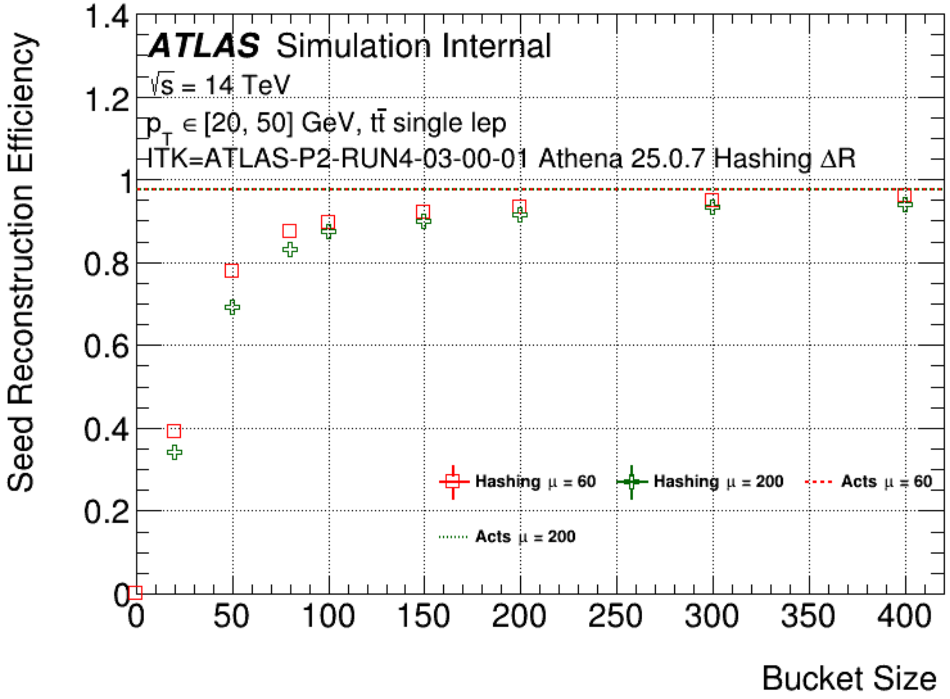
1000 events

Bucket Size ΔR: pT

InnerTracker

WARNING: not only first layer selected

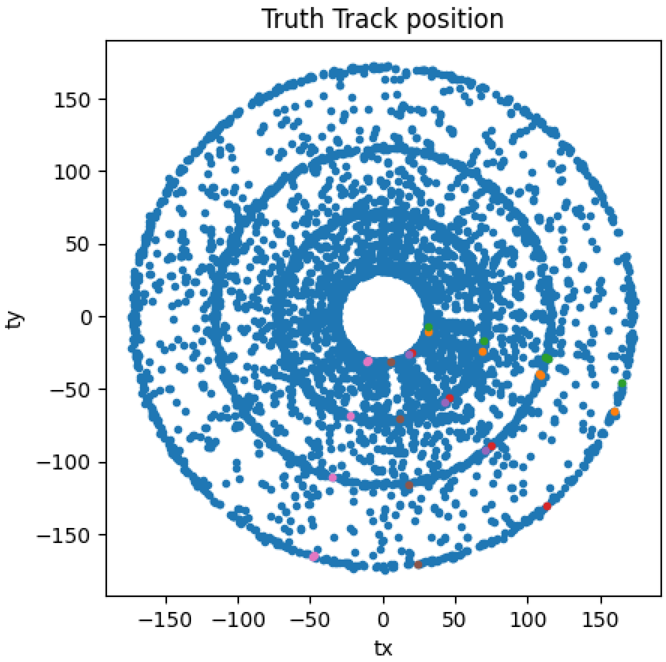




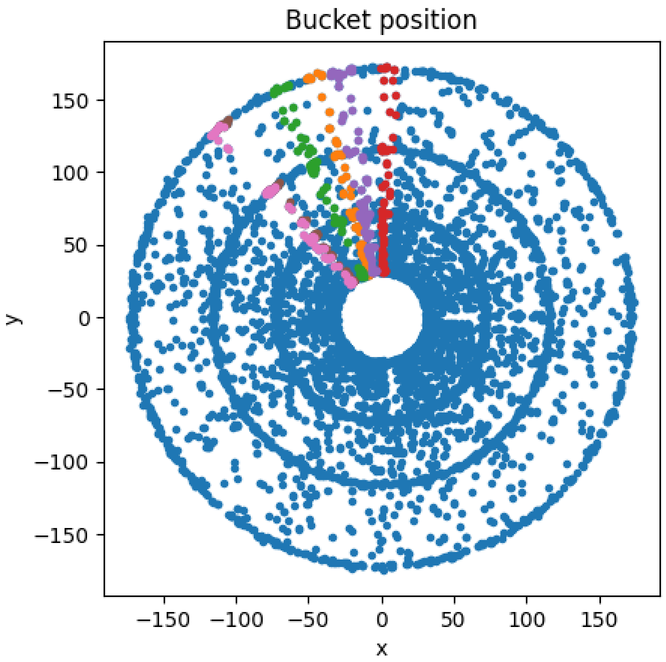
1000 events

Comparison (x,y) plan

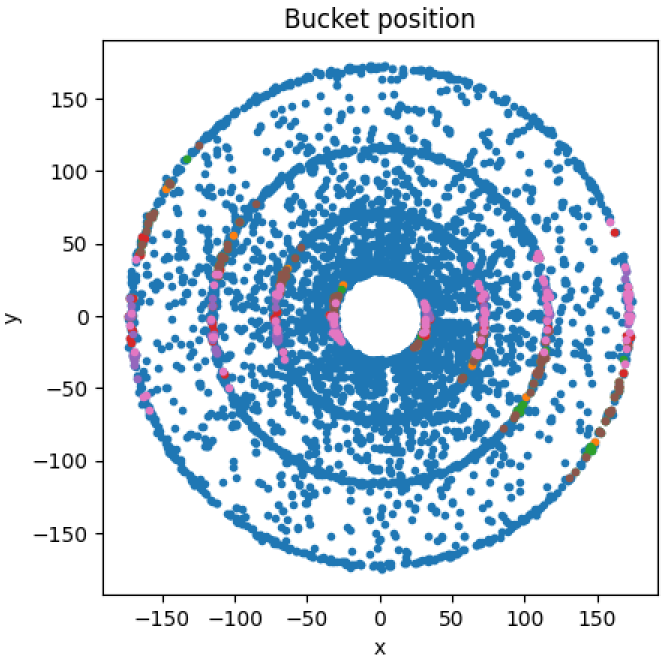
Truth Tracks (hits)



Angular: Δφ

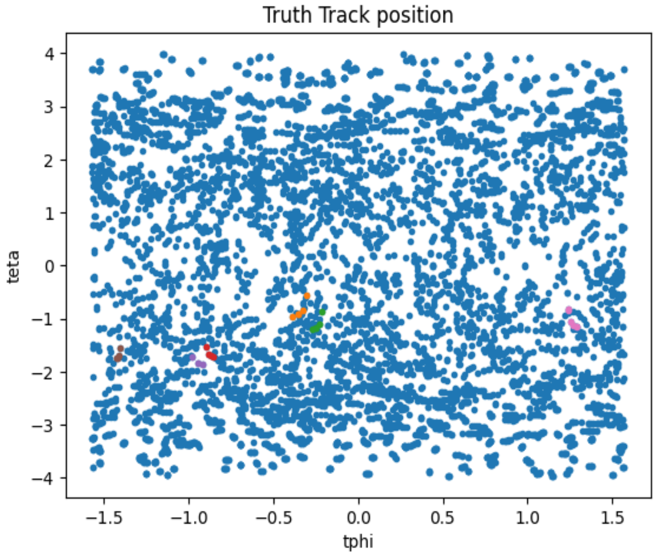


ΔR

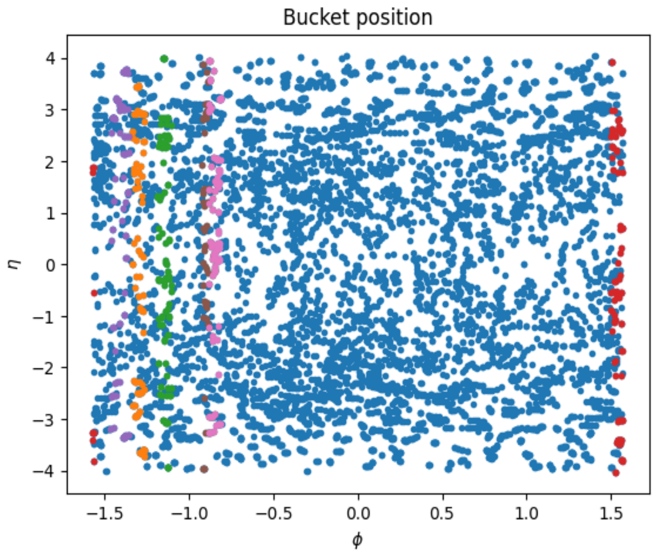


Comparison (φ,η) plan

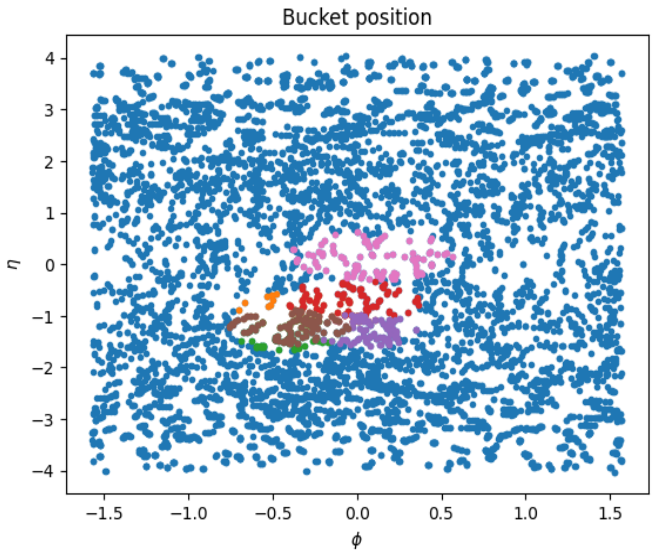
Truth Tracks (hits)



Angular: Δφ



ΔR



Layer information coverage

- **Information coverage varies with h**

h=0.5

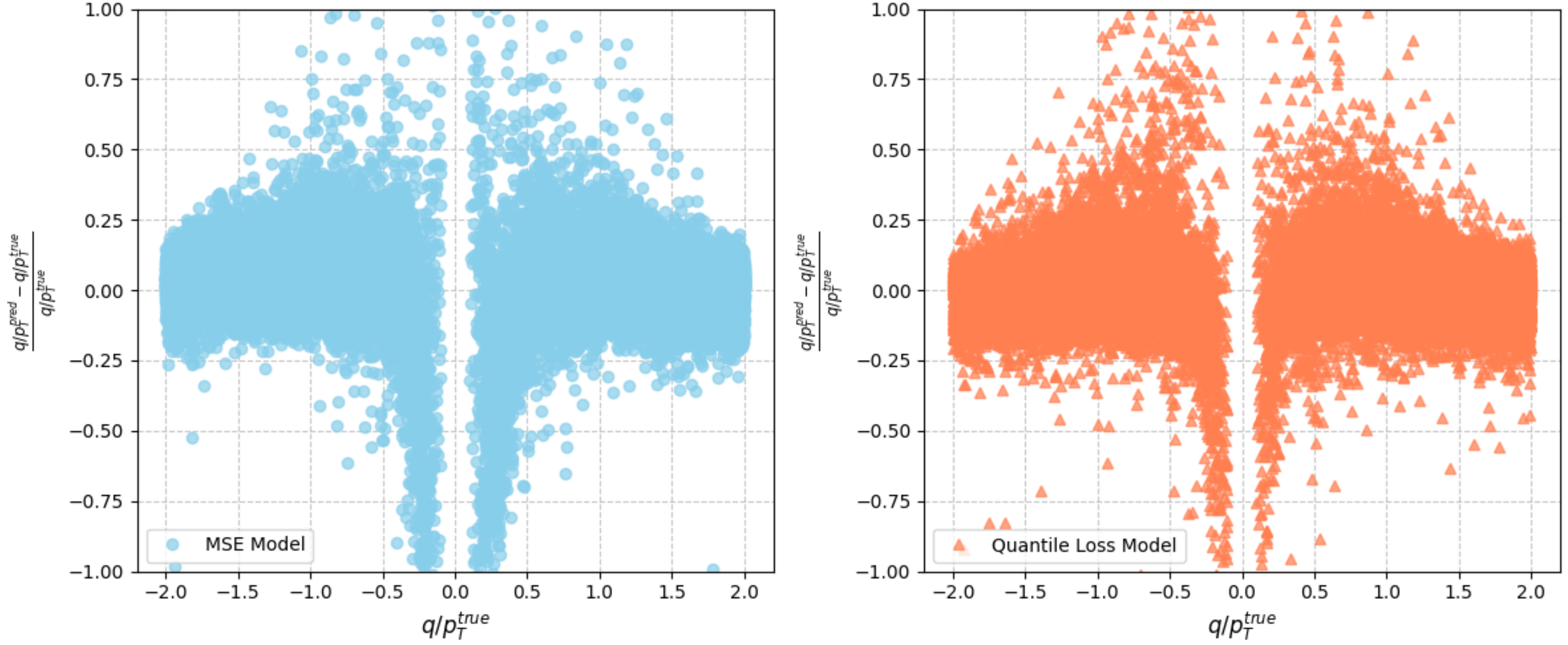
layer	particle_id	subevent	barcode	px	py	pz	pt	eta	vx	vy	vz	radius	status	charge	pdgId	vProdNIn	vProdNOut
12	0.57	0.38	0.55	0.54	0.54	0.54	0.51	0.56	0.38	0.38	0.38	0.35		0.09	0.17	0.31	0.34

More results

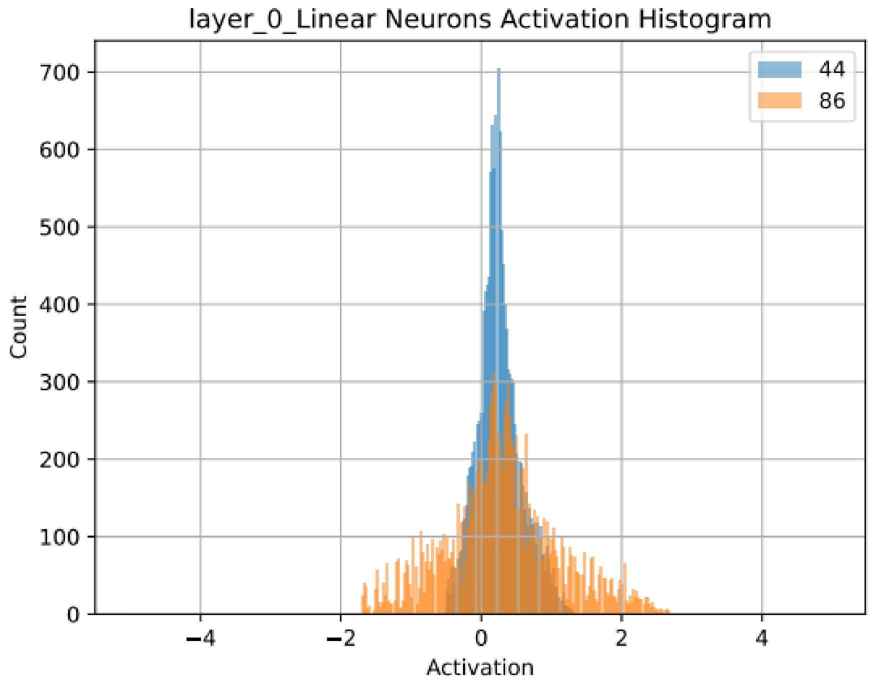
x y z

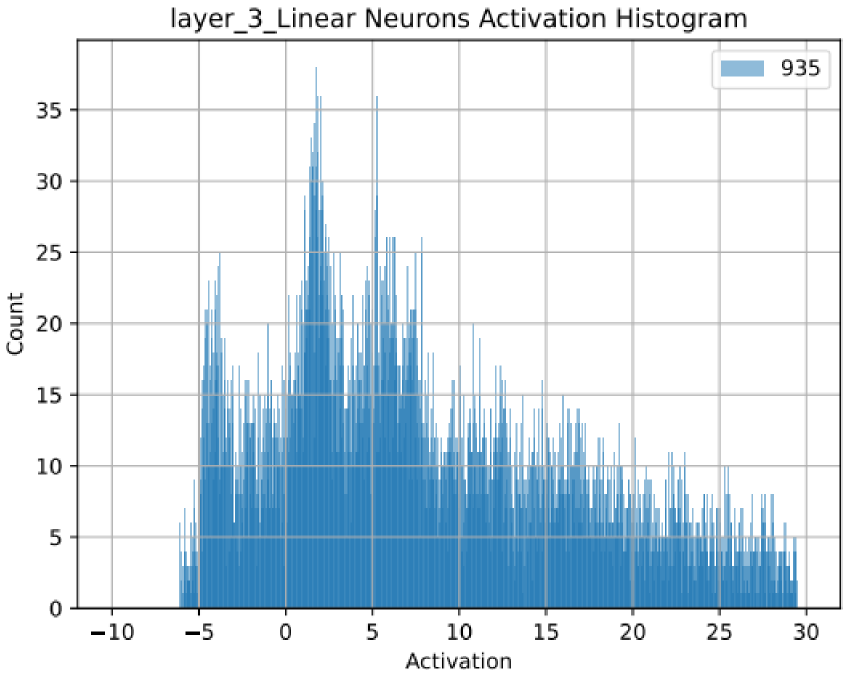
TrackML Zenodo

Relative error resolution for q/p_T ($(tx, ty, tz) \rightarrow (q/p_T, p_z)$)



Activations





KNN estimation

- **Estimate mutual information**

$$||z - z'|| = \max\{||x - x'||, ||y - y'||\}$$

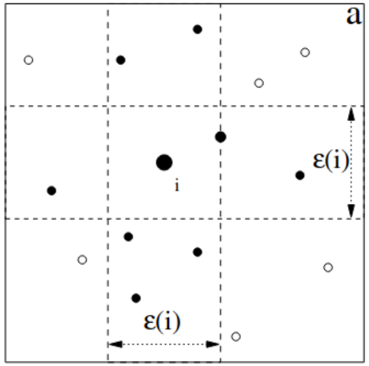
$$\langle \dots \rangle = N^{-1} \sum_{i=1}^{N} \mathsf{E}[\dots(i)]$$

$$I^{(1)}(X,Y) = \psi(k) - \langle \psi(n_x+1) + \psi(n_y+1) \rangle + \psi(N)$$

Here, $\psi(x)$ is the digamma function, $\psi(x) = \Gamma(x)^{-1} d\Gamma(x)/dx$. It satisfies the recursion $\psi(x+1) = \psi(x) + 1/x$ and $\psi(1) = -C$ where $C = 0.5772156\dots$

Let us denote by $\epsilon(i)/2$ the distance from z_i to its k-th neighbour, and by $\epsilon_x(i)/2$ and $\epsilon_y(i)/2$ the distances between the same points projected into the X and Y subspaces. Obviously, $\epsilon(i) = \max\{\epsilon_x(i), \epsilon_y(i)\}$.

In the first algorithm, we count the number $n_x(i)$ of points x_j whose distance from x_i is strictly less than $\epsilon(i)/2$, and similarly for y instead of x.

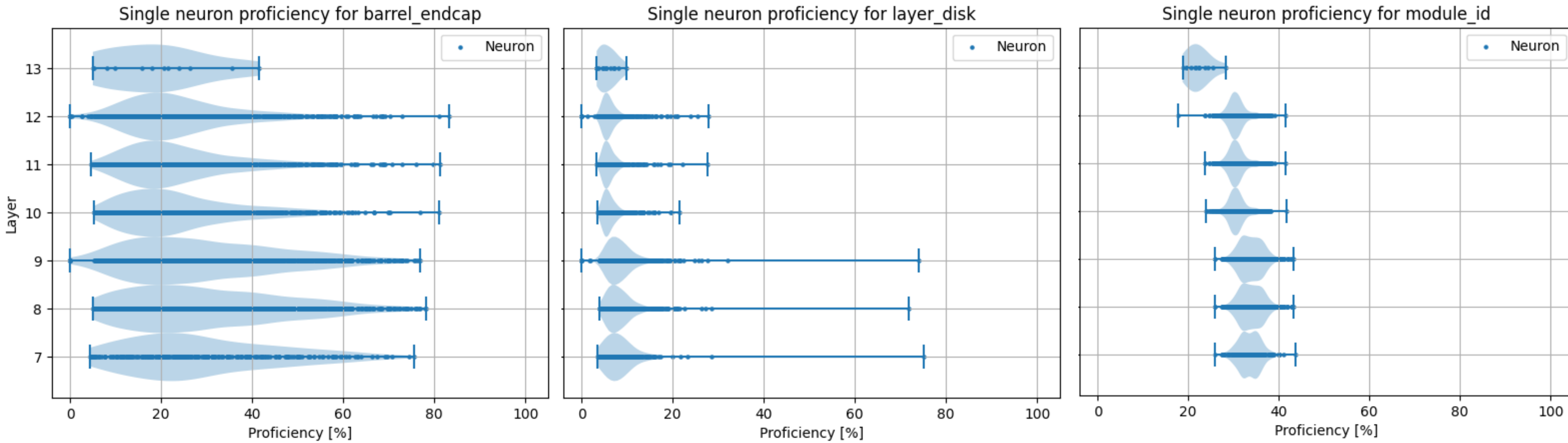


Used by scikit-learn

Outline

1. Inner Tracker building
2. ATLAS Qualification Task
3. ATLAS Tracking
4. Hashing in ACTS
5. Hashing in Athena
6. Interpretability
7. Track parameter regression
8. Doctoral training

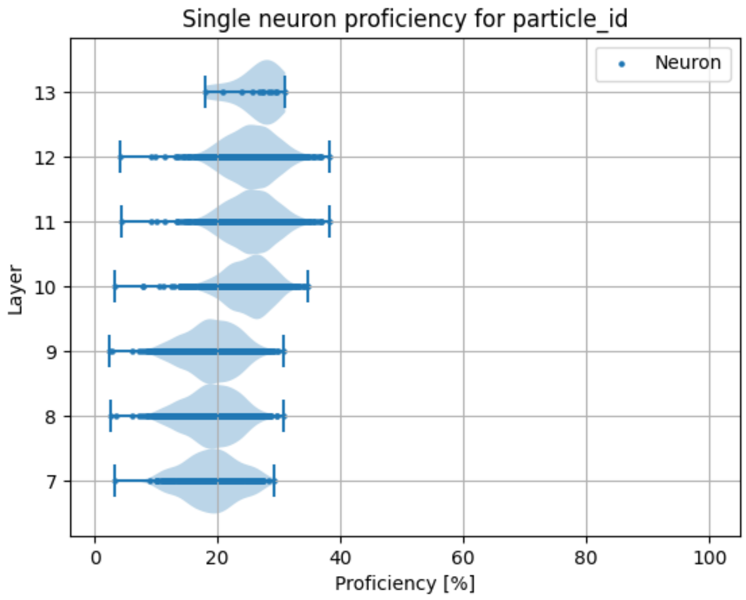
Highest variables single neuron

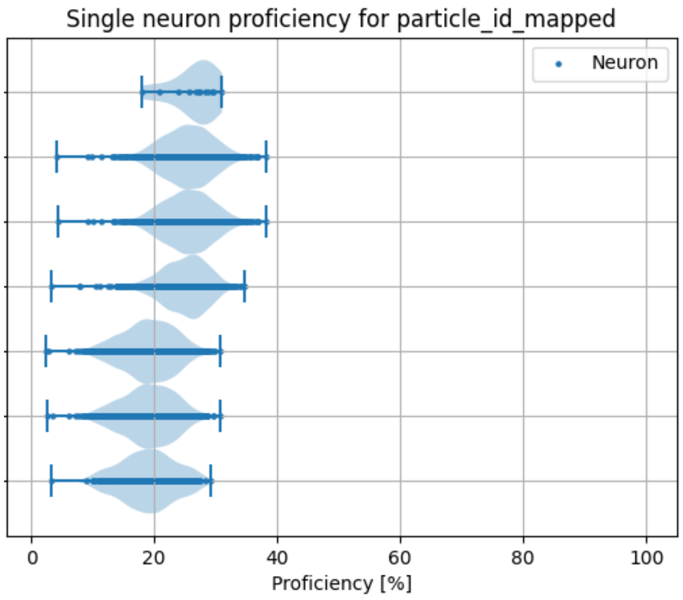


7 event: 118115 hits

From Scikit-learn Mutual information calculation

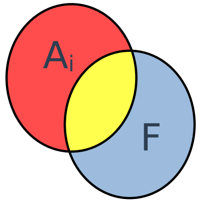
Change of variable impact





Should be close to 0:
Any particle can
hit any cell

Indicates not enough data

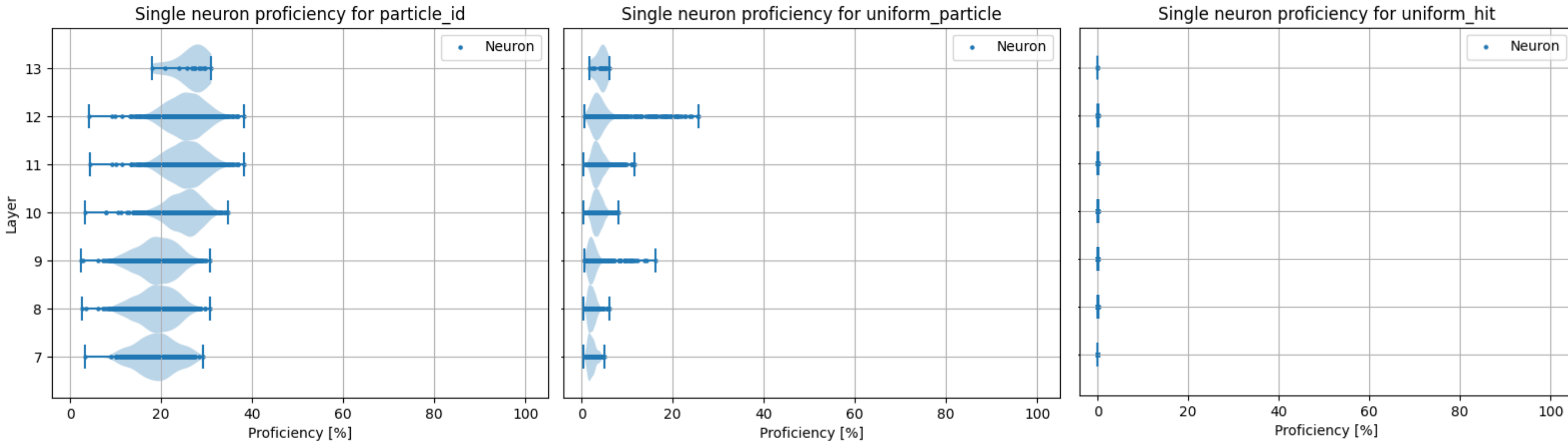


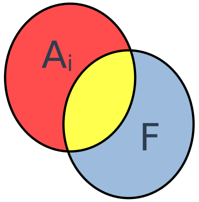
Same

From Scikit-learn Mutual information calculation

7 event: 118115 hits

Random variables single neuron: uniform

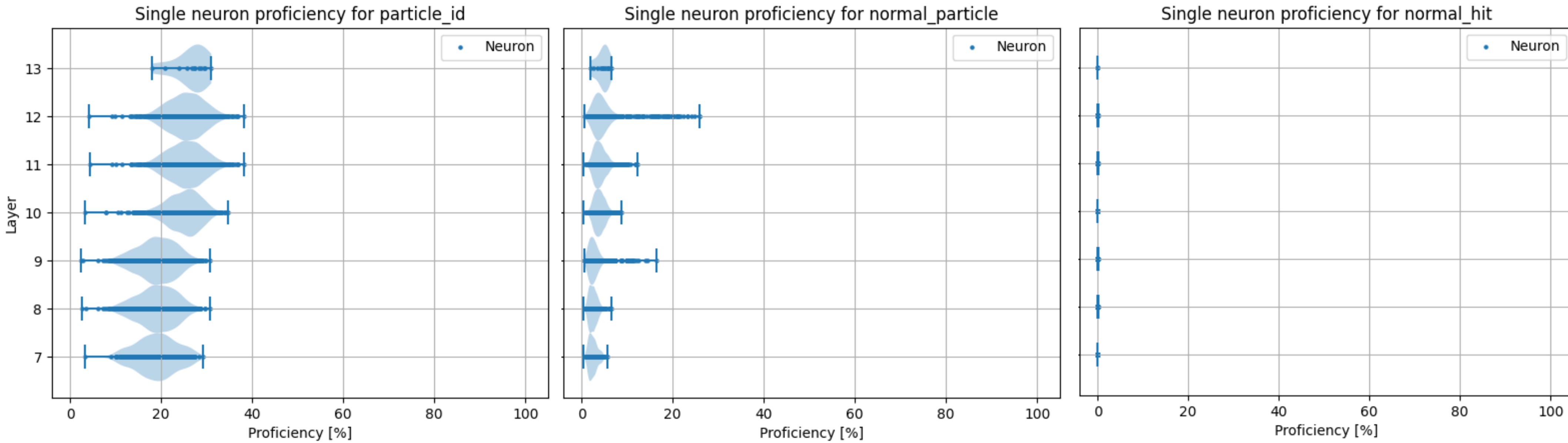


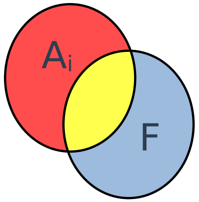


From Scikit-learn Mutual information calculation

7 event: 118115 hits

Random variables single neuron: normal

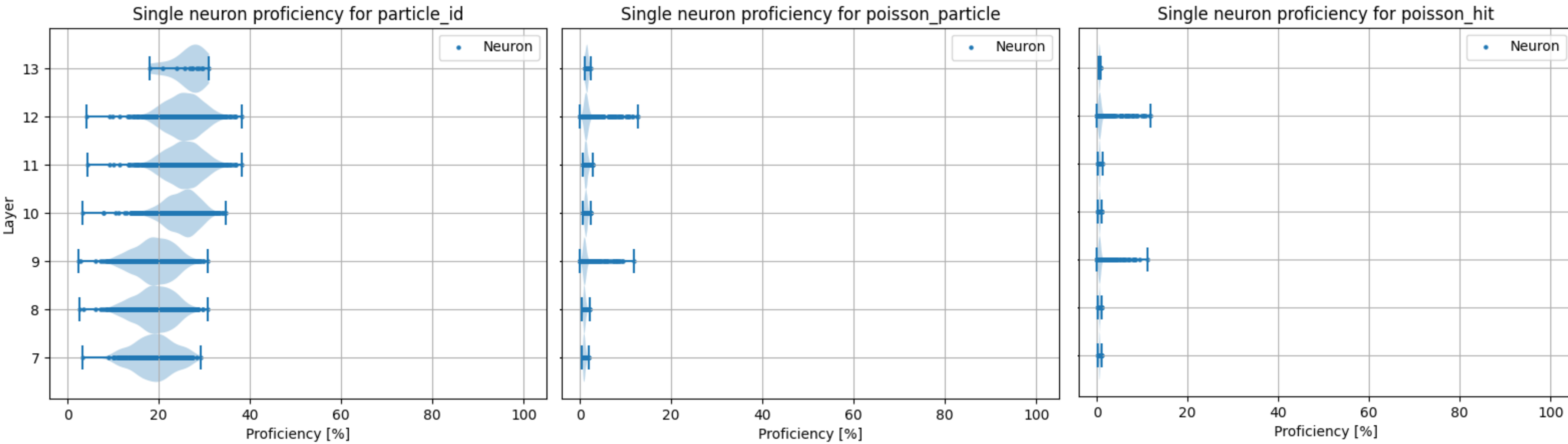


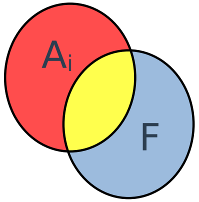


From Scikit-learn Mutual information calculation

7 event: 118115 hits

Random variables single neuron: poisson

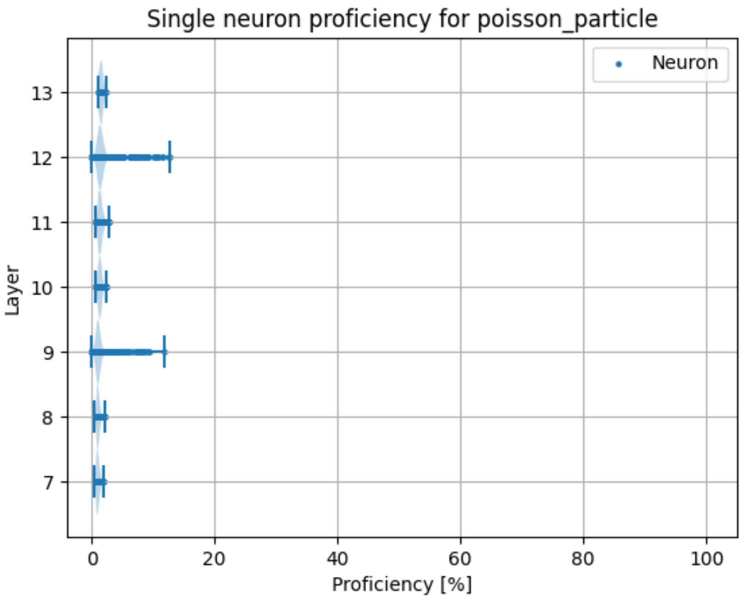


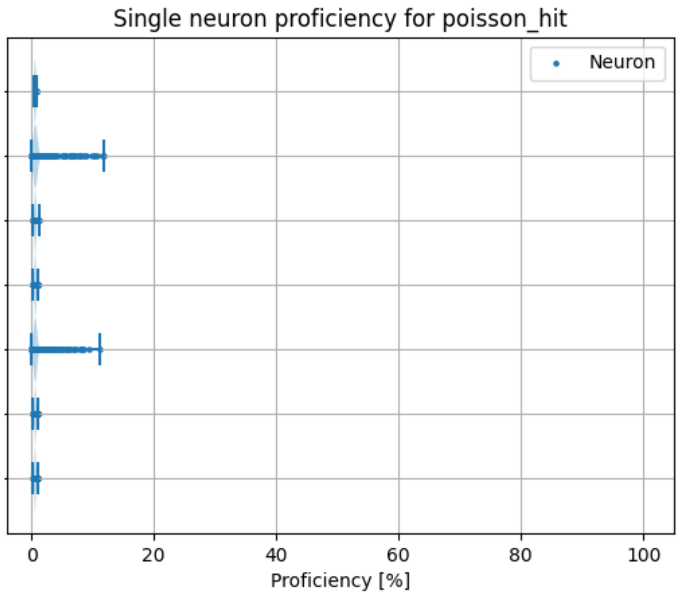


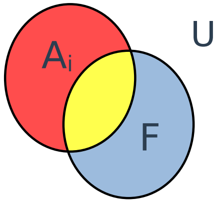
From Scikit-learn Mutual information calculation

7 event: 118115 hits

Random variables from single neurons: poisson







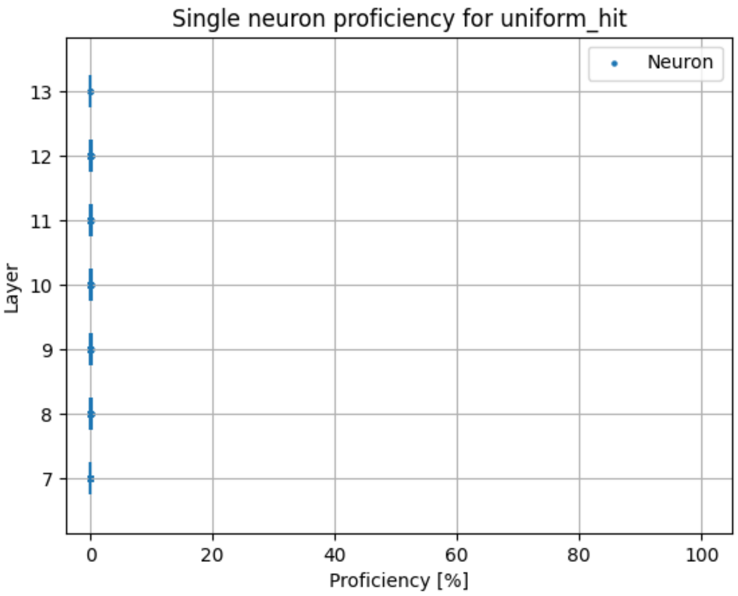
Unique particles can be found

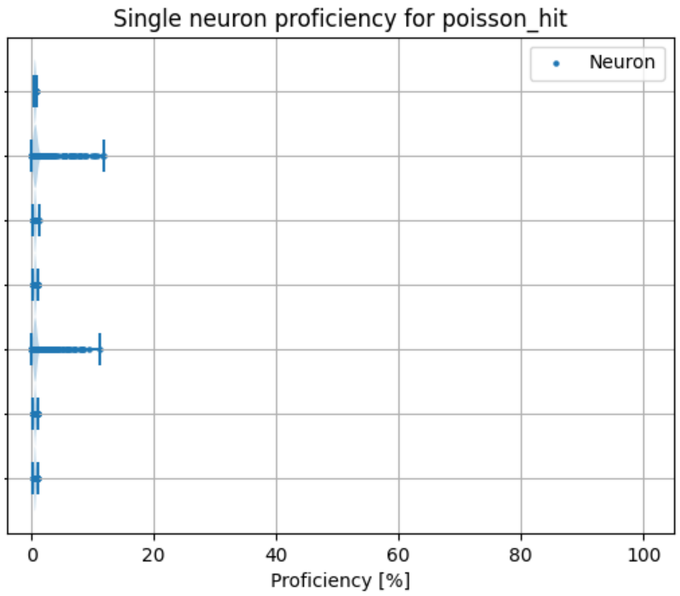
Unique hit can be found

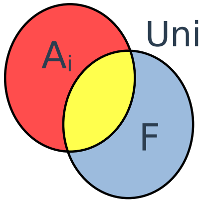
7 event: 118115 hits

From Scikit-learn Mutual information calculation

Discrete vs continuous variables







Unique hit value cannot be found

Random but repeated hit values can be found

7 events: 118 115 hits

From Scikit-learn Mutual information calculation

QT Supervision

Repository

GitLab

Interface

Streamlit

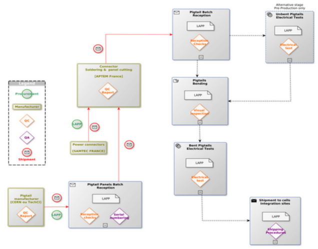
Hosting

okd

API

Database

CERN
ITk Production Database



Production flows

Monika Wieler

Kenneth Wraight

Jessica Levêque

Renaud Gaglione

Users

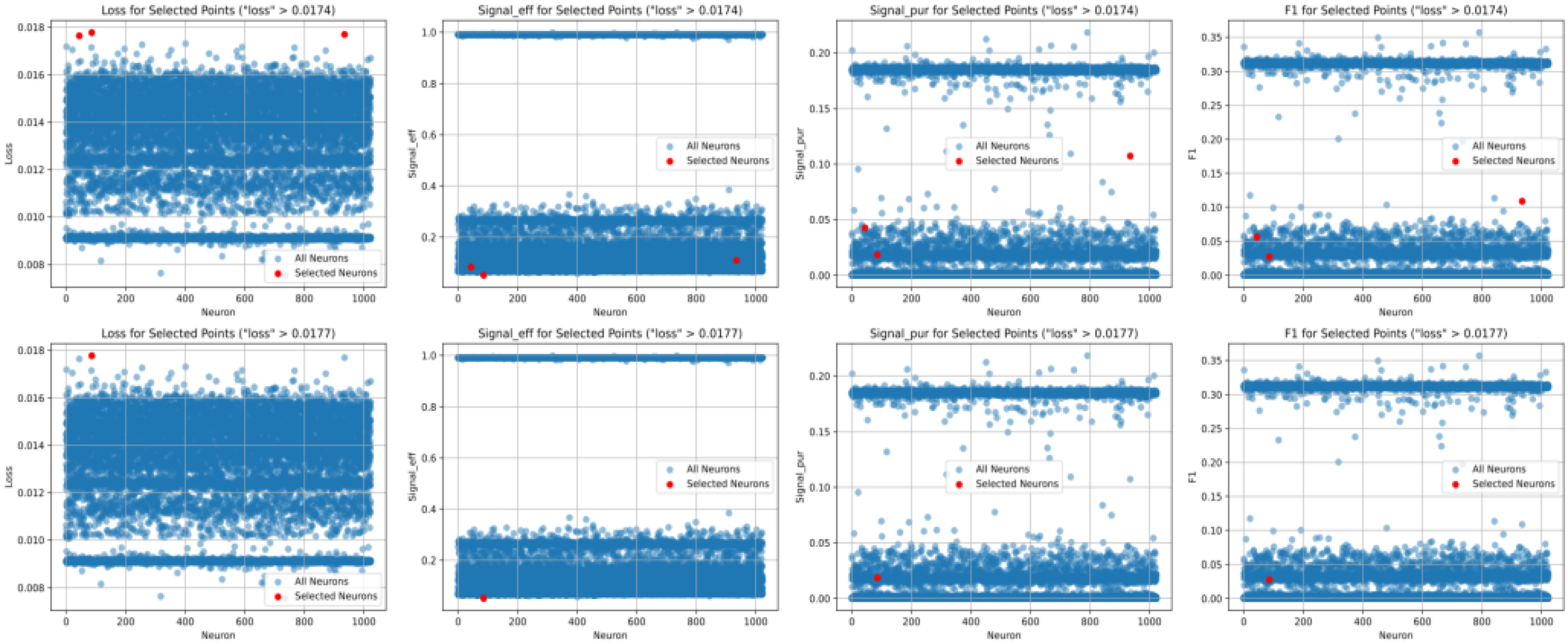
LAPP

Andrea Jeremie
Francesco Costanza

Dedicated web app:

Forked from itk-web-apps

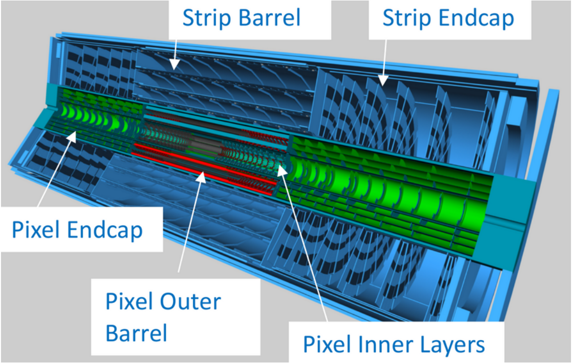
Neuron specificities: Permutation metrics

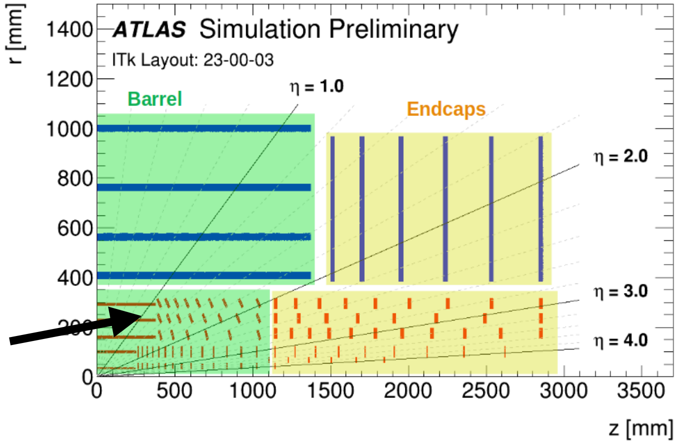


ATLAS QUALIFICATION TASK

Inner Tracker building at LAPP

LAPP is producing 75% of the OB Types 0 (5000 pigtails, 400 PP0 boards) and will be integrating 25% of the local supports(*)





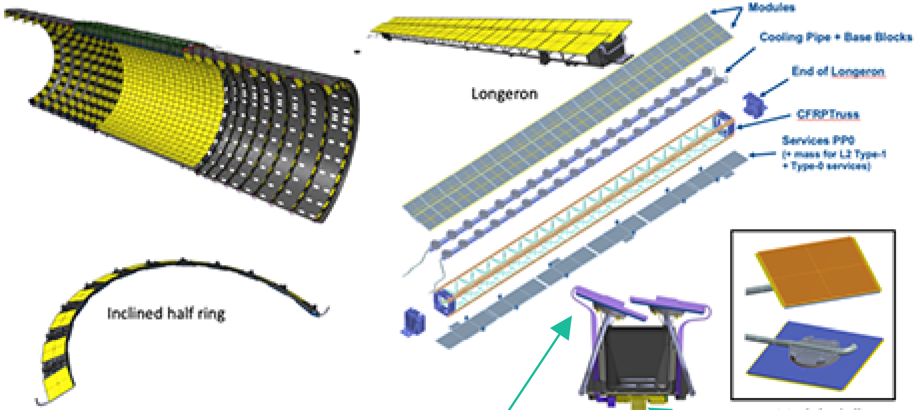
Types 0:
Components directly on the detector

LAPP
Laboratoire d'Annecy de Physique des Particules

(*)With LPSC and CPPM

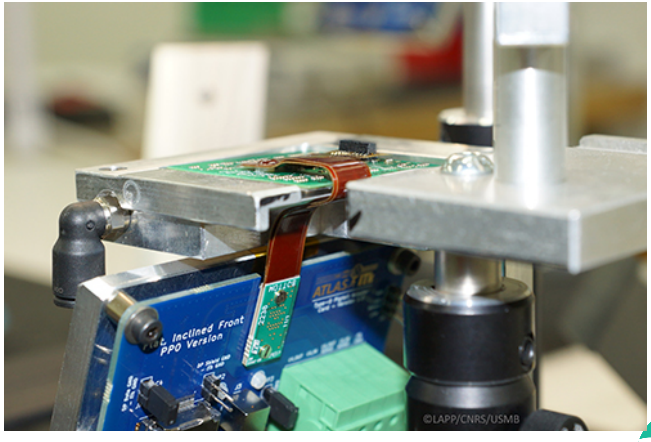
Inner Tracker Pixel Detector Overview

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



PP0

pigtails



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

ATLAS Qualification Task: Production Database

- **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

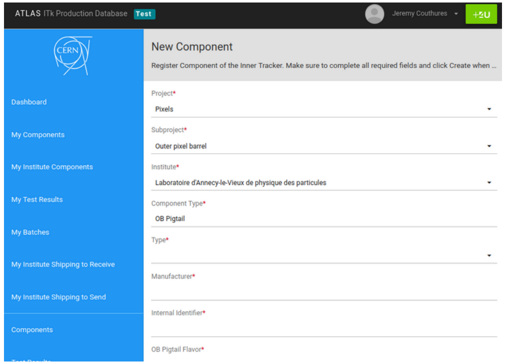
Registration in the database

Registration speed

1 to 1

Low level UI

Fields and buttons



1 to many

Web app

Fields and buttons

Streamlit

No operator

Web app

JSON files from LabVIEW

Streamlit

LabVIEW

Pigtails production flow

Similar production flow for PP0

BAD_HV_TEST
BAD_FROM_ELECTRICAL_TEST_DATA_TRANSMISSION
BAD_FROM_ELECTRICAL_TEST_RESISTANCE

Electrical Tests

Pass/Fail

Unbent Electrical Tests

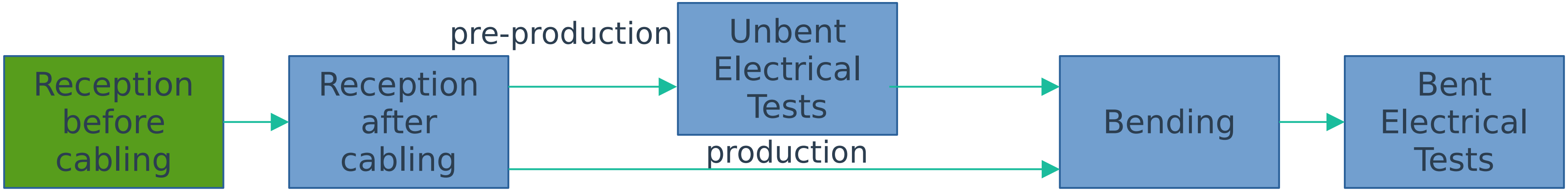
pre-production

BAD_HV_TEST
BAD_FROM_ELECTRICAL_TEST_DATA_TRANSMISSION
BAD_FROM_ELECTRICAL_TEST_RESISTANCE

Pass/Fail

Reception before cabling → Reception after cabling → (production) → Bending → Bent Electrical Tests

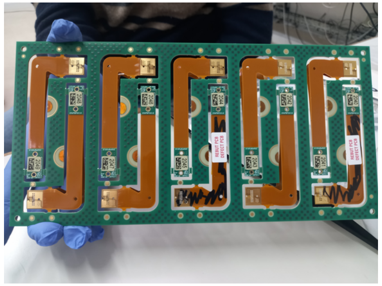
Reception before cabling	Reception after cabling	Bending	Bent Electrical Tests
Visual Inspection	Visual Inspection	Visual Inspection	Electrical Tests
Pass/Fail+properties set	Pass/Fail+info	Pass/Fail+properties set	
BAD_FROM_MANUFACTURER	BAD_FROM_CABLING_VI	BAD_BENDING	



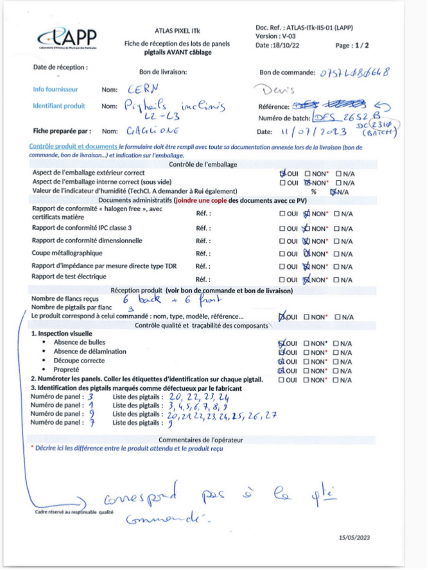
Creation of the pigtails in the database:

- Panel level comment? Pigtail marked as bad? From which panel?

Object



Form



LAPP

ATLAS PIXEL ITk
Fiche de réception des lots de panels
pigtails AVANT câblage

Doc. Ref. : ATLAS-ITk-IIS-01 (LAPP)
Version : V-03
Date :18/10/22
Page : **1 / 2**

Date de réception :
Bon de livraison:
Bon de commande: 0757L080648

Info fournisseur — Nom: CERN — Denis
Identifiant produit — Nom: Pigtails inclinés L2 - L3
Référence: ~~DFS~~ ~~[illegible]~~
Numéro de batch: DFS_2652_B DC 2310 (BATCH)
Fiche preparée par : Nom: GAGLIONE
Date: 11/07/2023

Contrôle produit et documents *le formulaire doit être rempli avec toute sa documentation annexée lors de la livraison (bon de commande, bon de livraison...) et indication sur l'emballage.*

Contrôle de l'emballage

Aspect de l'emballage extérieur correct		☒ OUI	☐ NON*	☐ N/A
Aspect de l'emballage interne correct (sous vide)		☐ OUI	☒ NON*	☐ N/A
Valeur de l'indicateur d'humidité (TechCI. A demander à Rui également)		%		☒ N/A

Documents administratifs (joindre une copie des documents avec ce PV)

Rapport de conformité « halogen free », avec certificats matière	Réf. :	☐ OUI	☒ NON*	☐ N/A
Rapport de conformité IPC classe 3	Réf. :	☐ OUI	☒ NON*	☐ N/A
Rapport de conformité dimensionnelle	Réf. :	☐ OUI	☒ NON*	☐ N/A
Coupe métallographique	Réf. :	☐ OUI	☒ NON*	☐ N/A
Rapport d'impédance par mesure directe type TDR	Réf. :	☐ OUI	☒ NON*	☐ N/A
Rapport de test électrique	Réf. :	☐ OUI	☒ NON*	☐ N/A

Réception produit (voir bon de commande et bon de livraison)

Nombre de flancs reçus: 6 back + 6 front
Nombre de pigtails par flanc: 3
Le produit correspond à celui commandé : nom, type, modèle, référence... ☒ OUI ☐ NON* ☐ N/A

Contrôle qualité et traçabilité des composants

1. Inspection visuelle

- Absence de bulles ☒ OUI ☐ NON* ☐ N/A
- Absence de délamination ☒ OUI ☐ NON* ☐ N/A
- Découpe correcte ☒ OUI ☐ NON* ☐ N/A
- Propreté ☒ OUI ☐ NON* ☐ N/A

2. Numéroter les panels. Coller les étiquettes d'identification sur chaque pigtail. ☐ OUI ☐ NON* ☐ N/A

3. Identification des pigtails marqués comme défectueux par le fabricant

Numéro de panel : 3 Liste des pigtails : 20, 22, 23, 24
Numéro de panel : 1 Liste des pigtails : 3, 4, 5, 6, 7, 8, 9
Numéro de panel : 9 Liste des pigtails : 20, 21, 22, 23, 24, 25, 26, 27
Numéro de panel : 7 Liste des pigtails : 9

Commentaires de l'opérateur

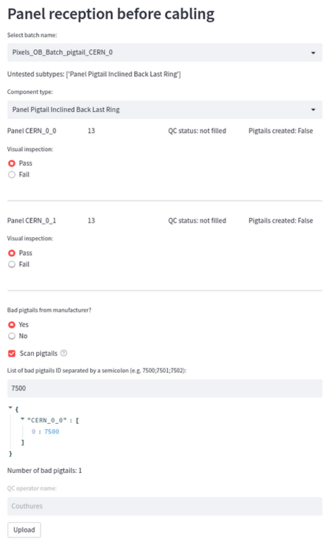
* *Décrire ici les différence entre le produit attendu et le produit reçu*

correspond pas à la qté commande.

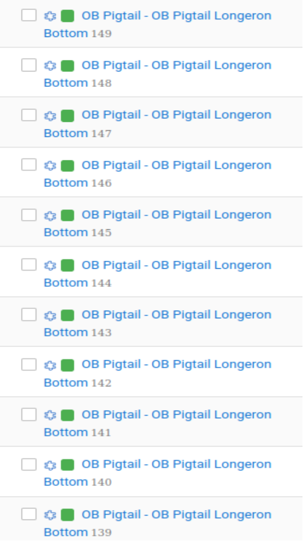
Cadre réservé au responsable qualité

15/05/2023

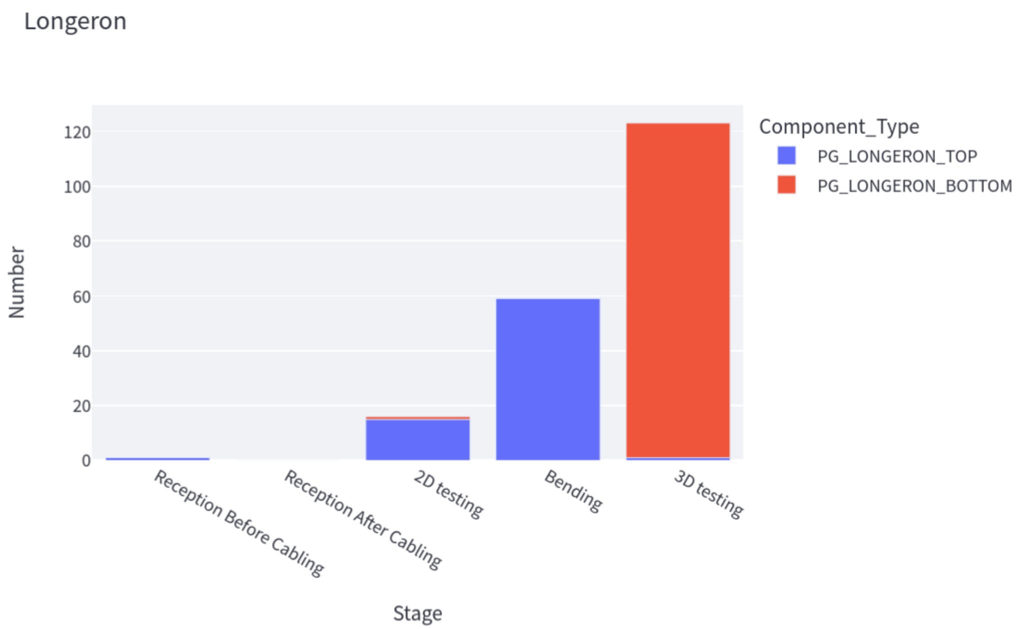
Web app

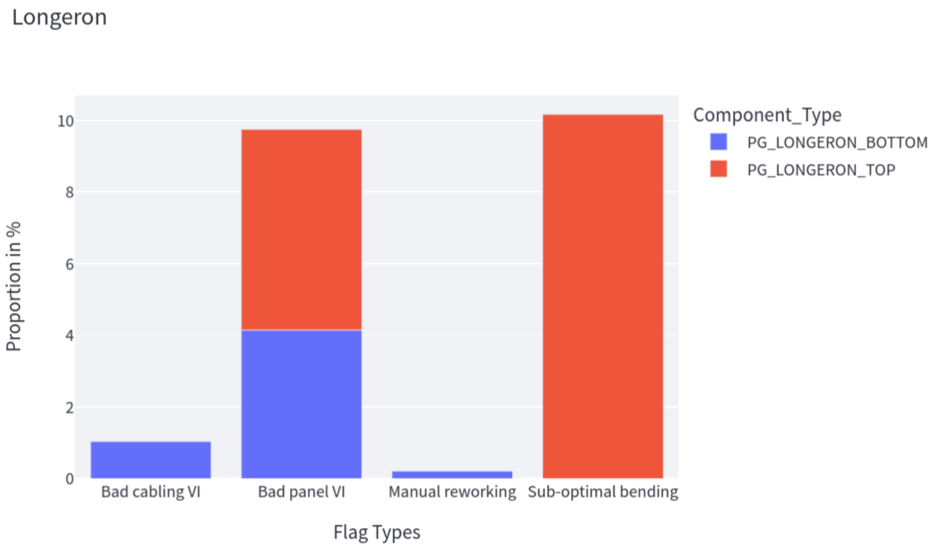


Database



Reporting





Plots not possible without the web app

ATLAS Qualification Task: Types 0 web app

Qualified since January 2024

Code on gitlab:
gitlab repository

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

QT presentation link:
indico link

Select component type:

OB_PIGTAIL

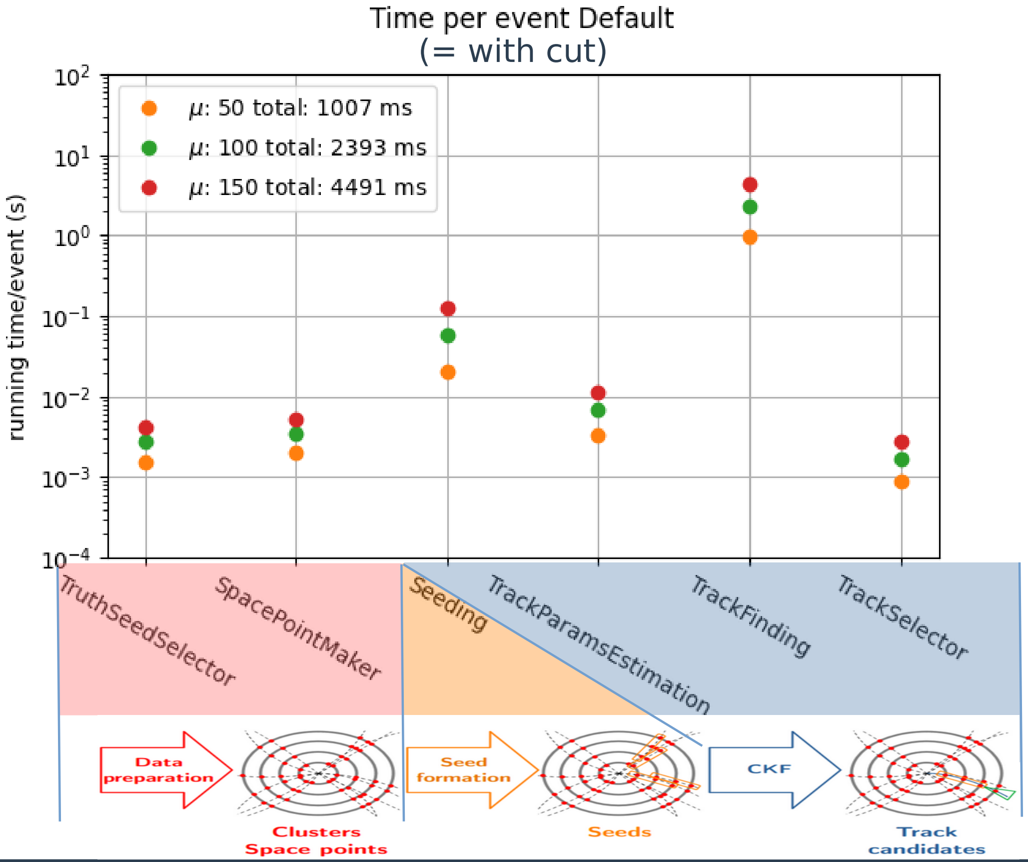
Select stage:

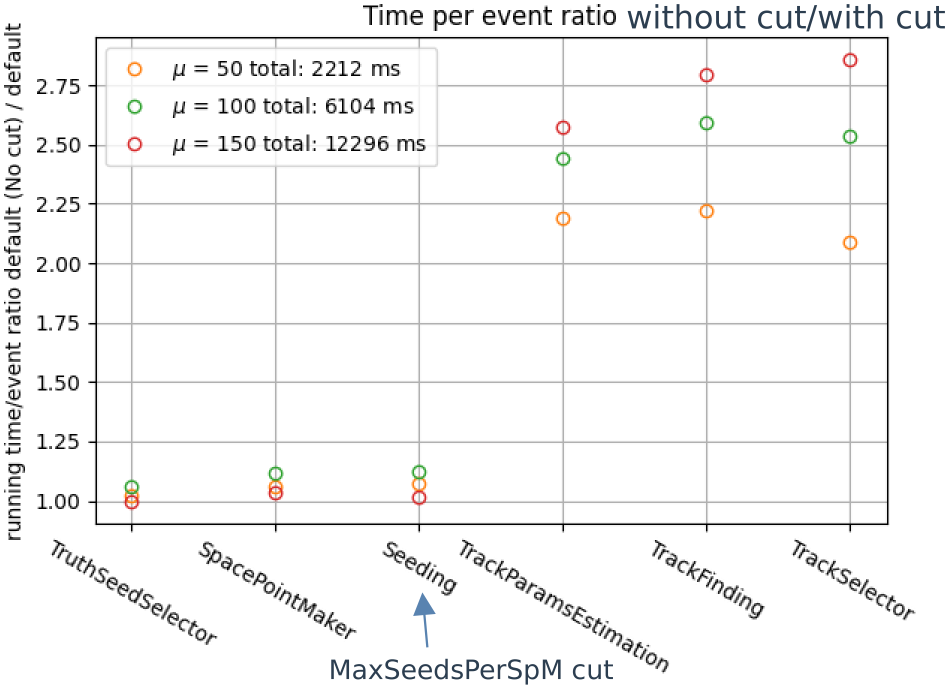
Reception Before Cabling

☐ Remove flagged components

Internal ID	Type	Stage	Link to PDB
lapp_2152	OB Pigtail Longeron Top	Reception Before Cabling	e90a6f1259e399ca44a7f078b5732d76
lapp_2148	OB Pigtail Longeron Top	Reception Before Cabling	86945b16501f77d00332c2244c852355
lapp_2142	OB Pigtail Longeron Top	Reception Before Cabling	f4c2488adcb76d3ad05b05b7fea5f632
lapp_2135	OB Pigtail Longeron Top	Reception Before Cabling	0d56d4136c8b65f739d2d0ad5fcd983b
lapp_2133	OB Pigtail Longeron Top	Reception Before Cabling	2cad78db9727bc97d0f19d78ec8a6f43
lapp_2123	OB Pigtail Longeron Top	Reception Before Cabling	bb4494d8125de87f1ba424f25f0073e8
lapp_2121	OB Pigtail Longeron Top	Reception Before Cabling	6b426487f703254c0a2ad6e9a00e7b8d

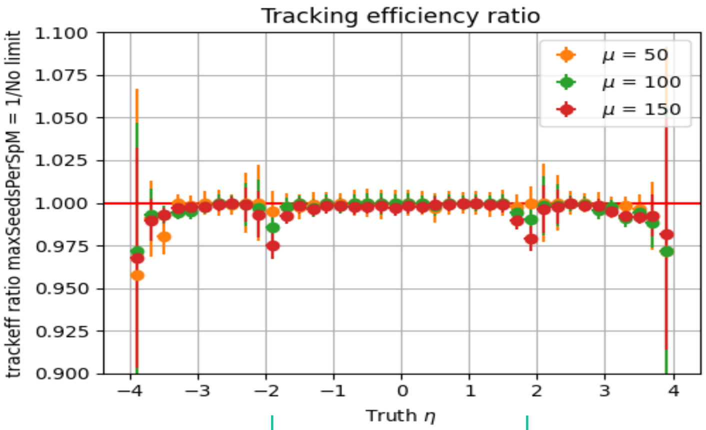
ACTS performance: Timing/event

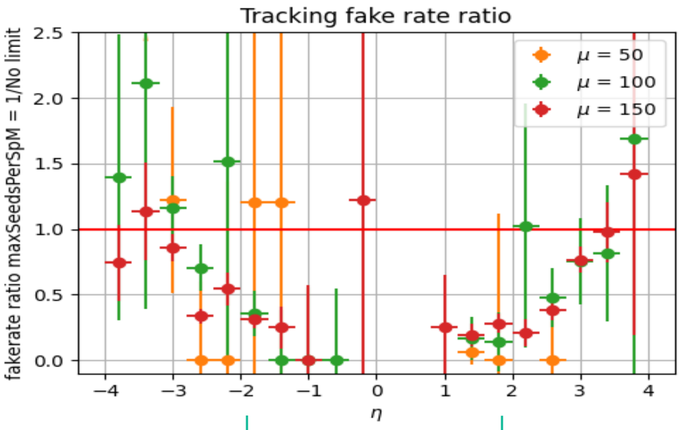


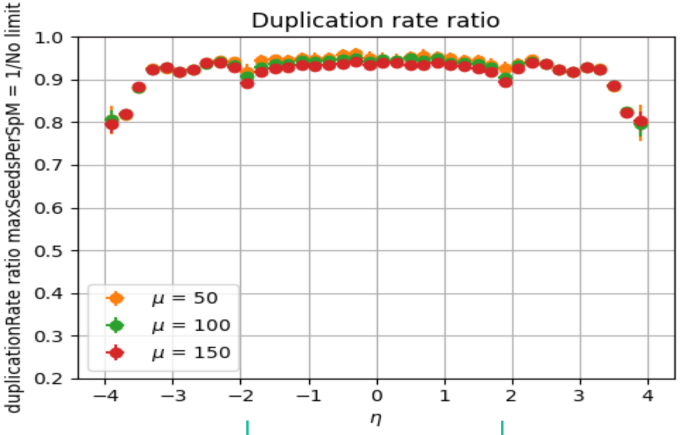


Timing multiplied by ~2-3 without the cut

ACTS performance: Physics







Forward region	Central region	Forward region

Same in central region,
Lower efficiency in forward region

Forward region	Central region	Forward region

Less fake tracks in central region,
Same in forward region

Forward region	Central region	Forward region

Less duplicated tracks,
Even less in forward region

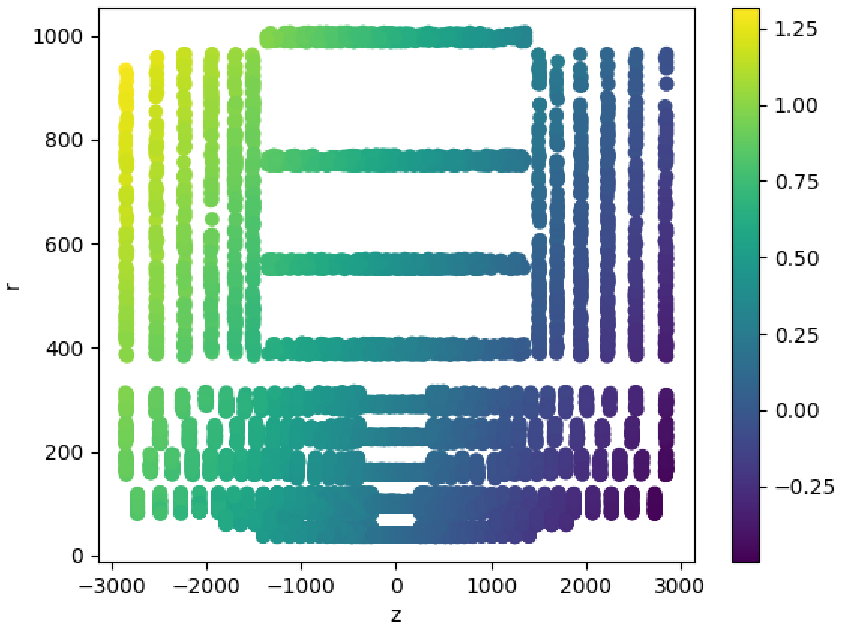
MaxSeedPerSpM cut decreases the performance in forward region
But improves in central region

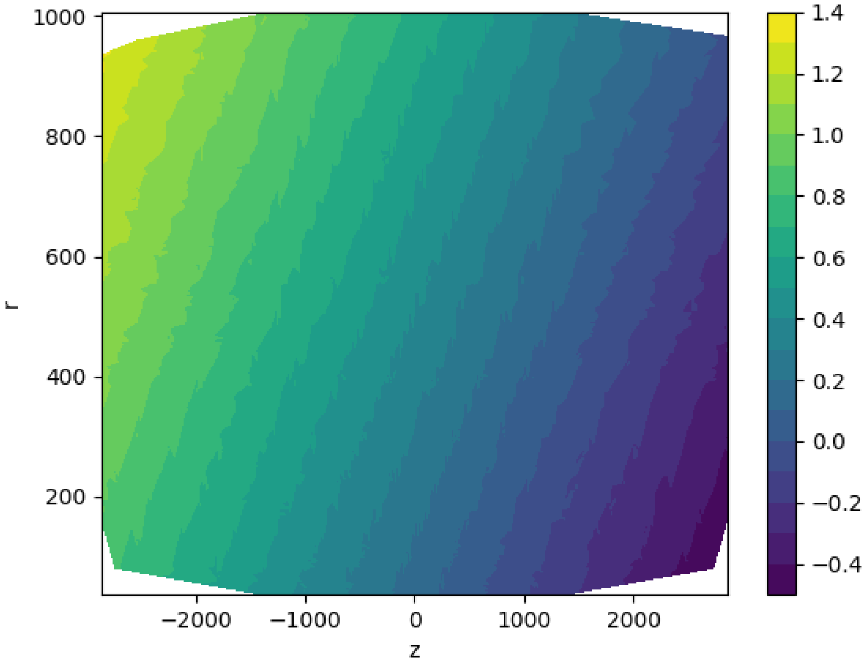
Target Goal

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

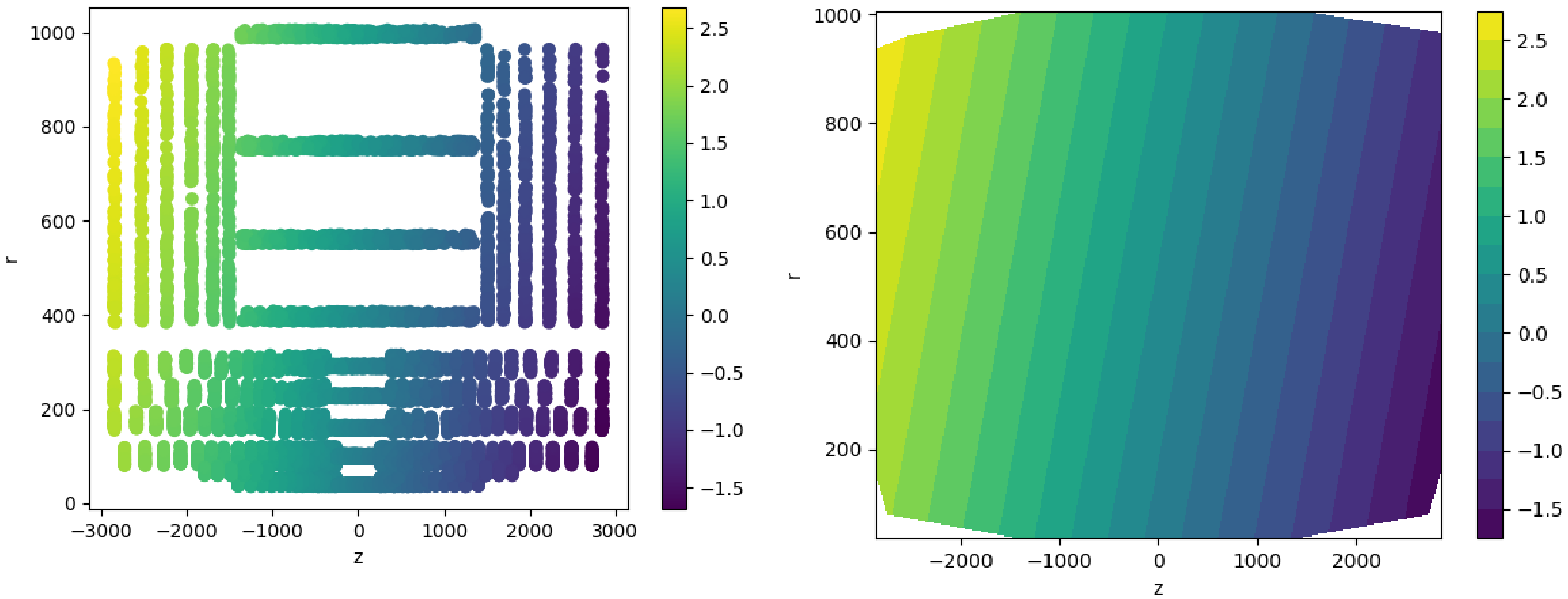
- **Goal: Improve performance with same timing**
  - Keep the cut but try to bypass it

Linear Layer: Activations r-z neuron 44



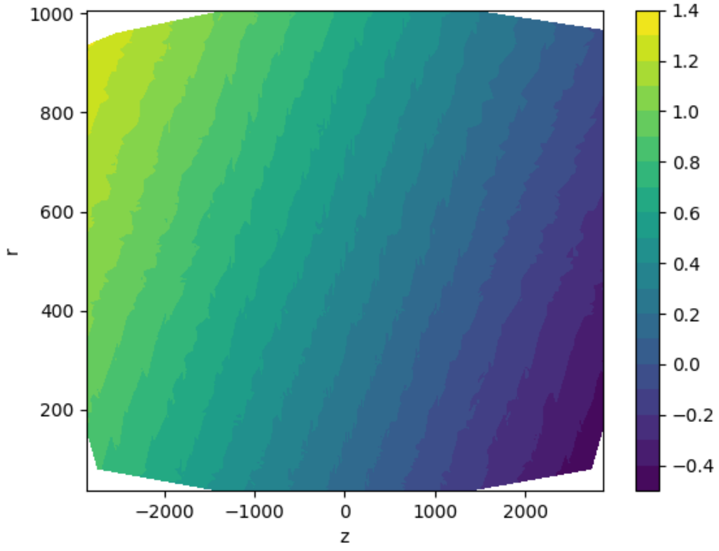


Linear Layer: Activations r-z neuron 86

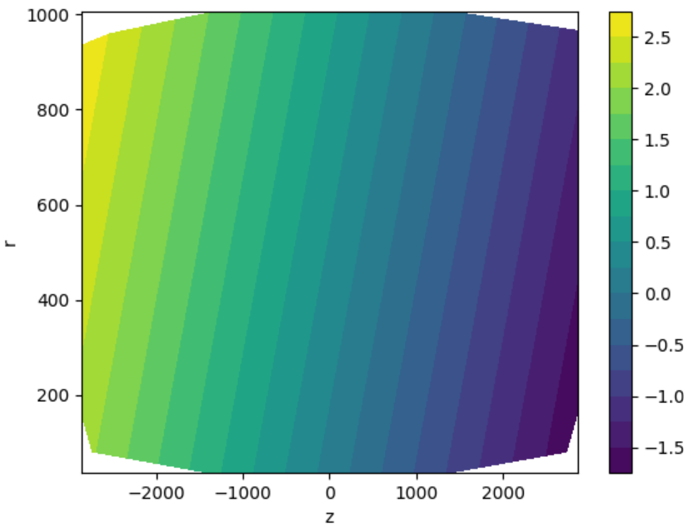


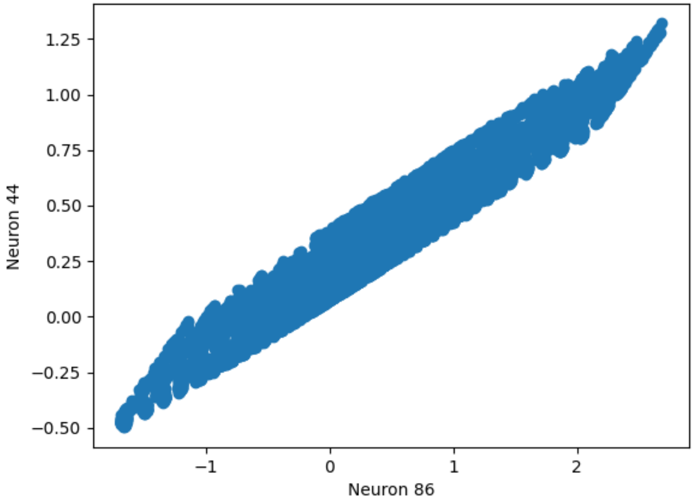
Linear Layer: Neuron 44 vs neuron 86

Neuron 44



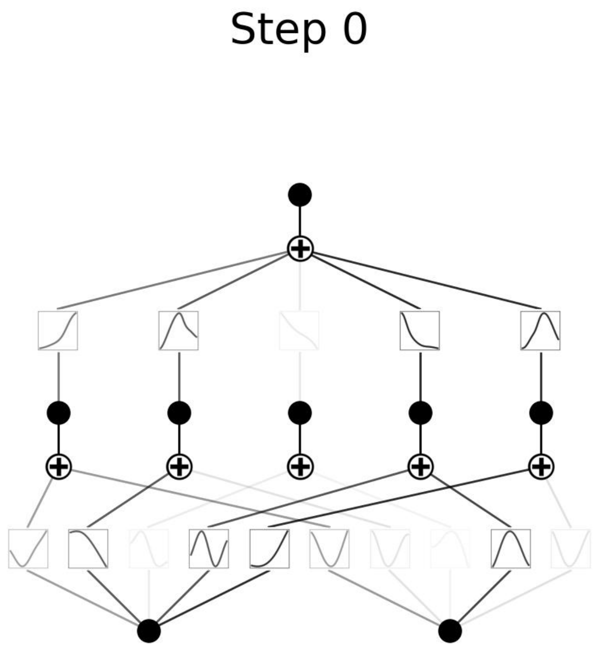
Neuron 86





Kolmogorov-Arnold Networks (KAN)

- **Training did not converge**
  - Did not improved after first batch
- **Playing with hyper-parameters did not help**



A new method:
Machine Learning/Hashing in the Seeding

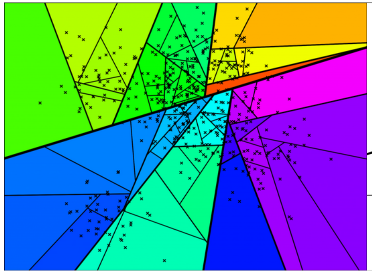
Hashing:
1. Group similar space points into buckets
2. Do the seeding on each bucket

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

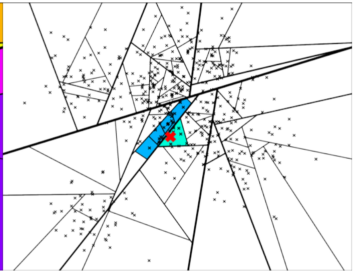
- Machine Learning algorithm type:
  - k Nearest Neighbors (unsupervised)
  - Random based

- Find Neighbors of the points in layer 0

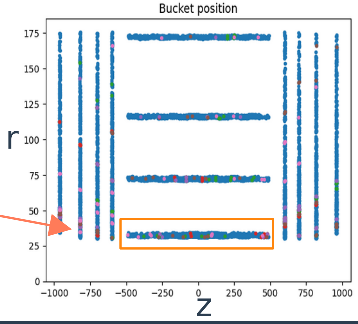
1 space point in layer 0 → 1 bucket

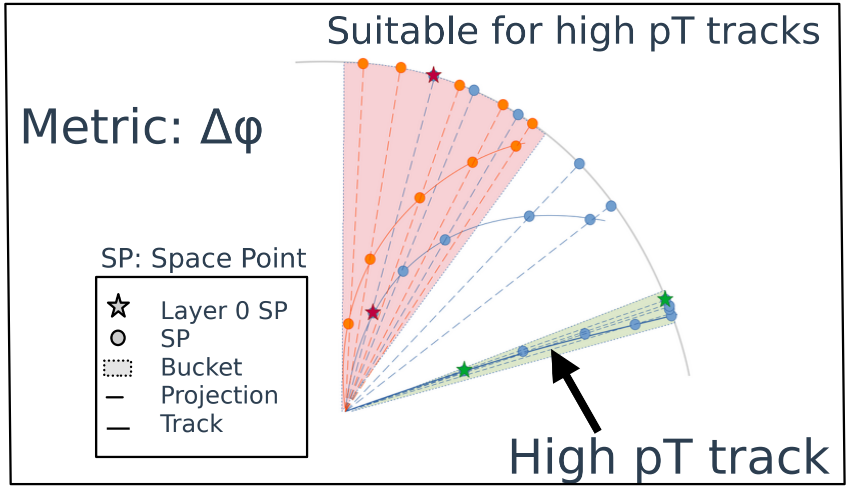


Space separation



Look for neighbors in the closest regions



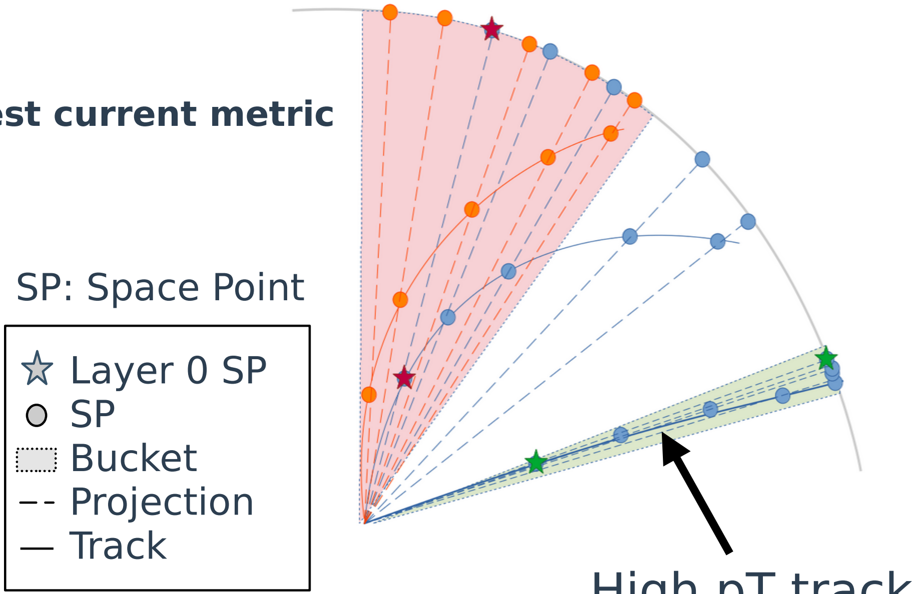


Metric and bucket size

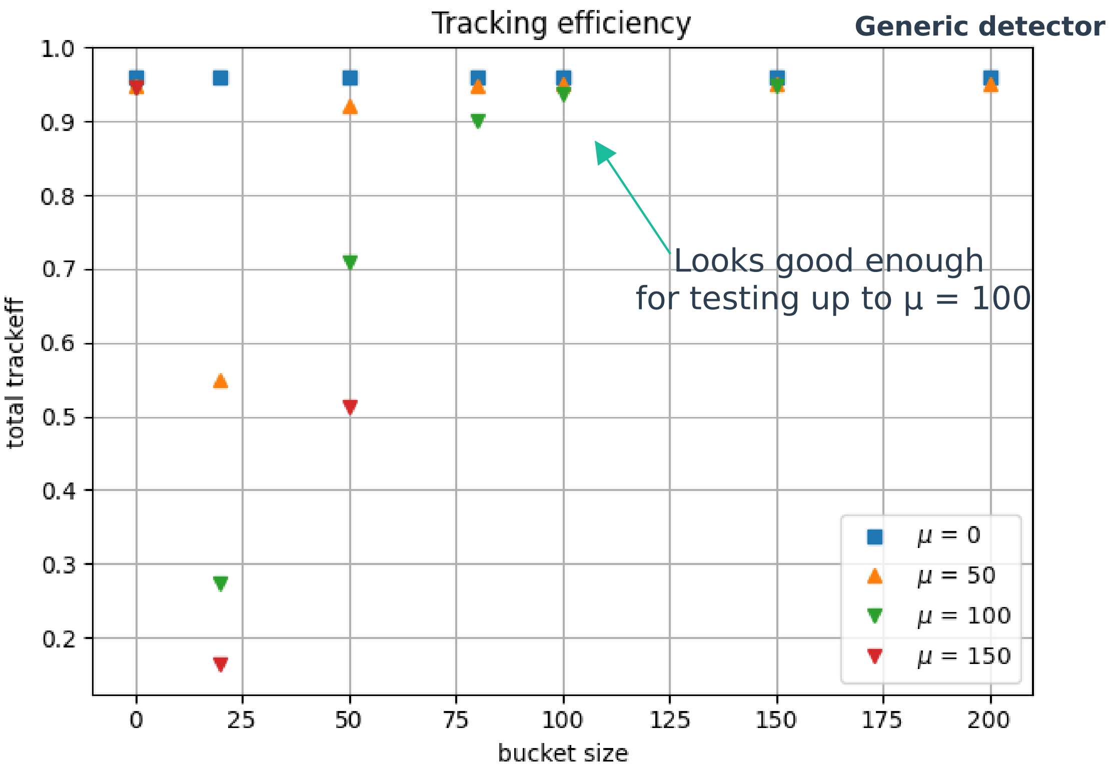
Metric: Δφ

Suitable for high pT tracks

Best current metric

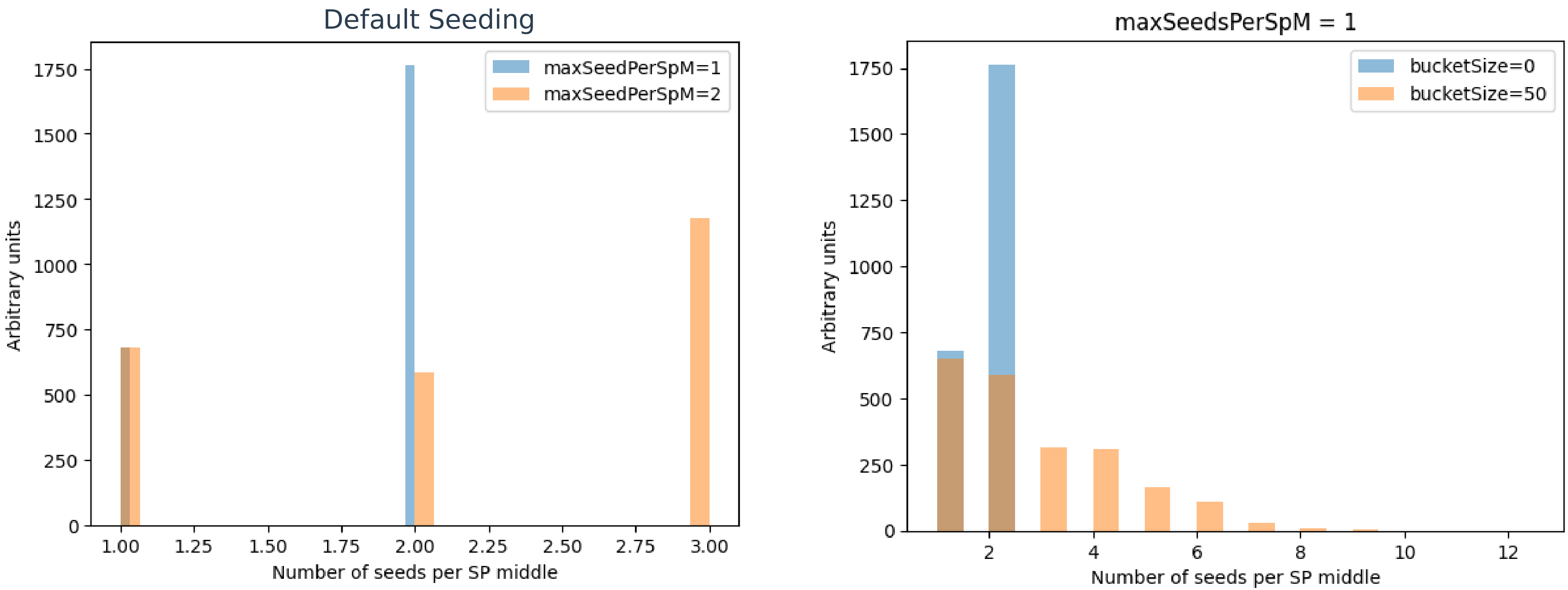


SP: Space Point



Fix the number of neighbors (bucket size) to 100

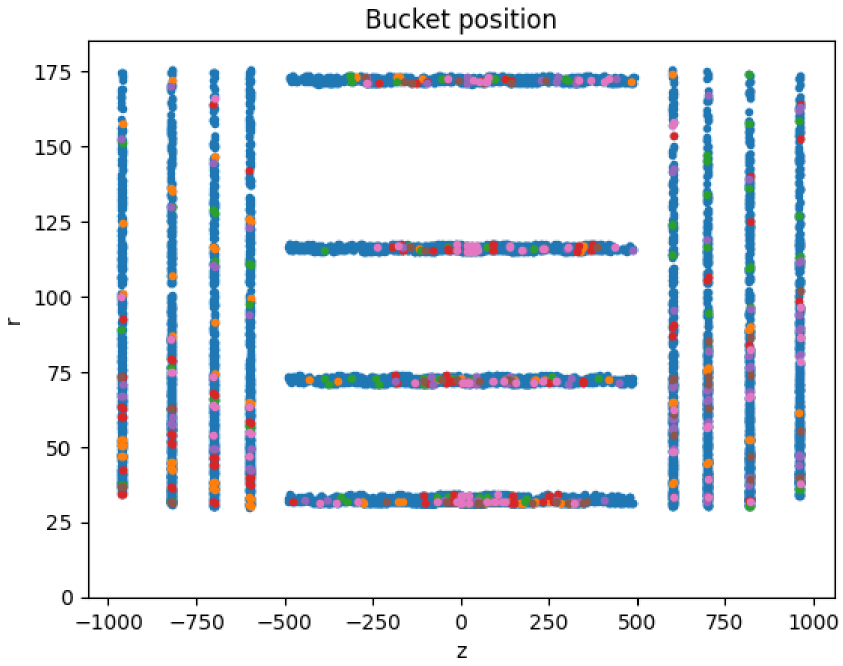
MaxSeedsPerSpM cut vs Hashing



Hashing get through the cut

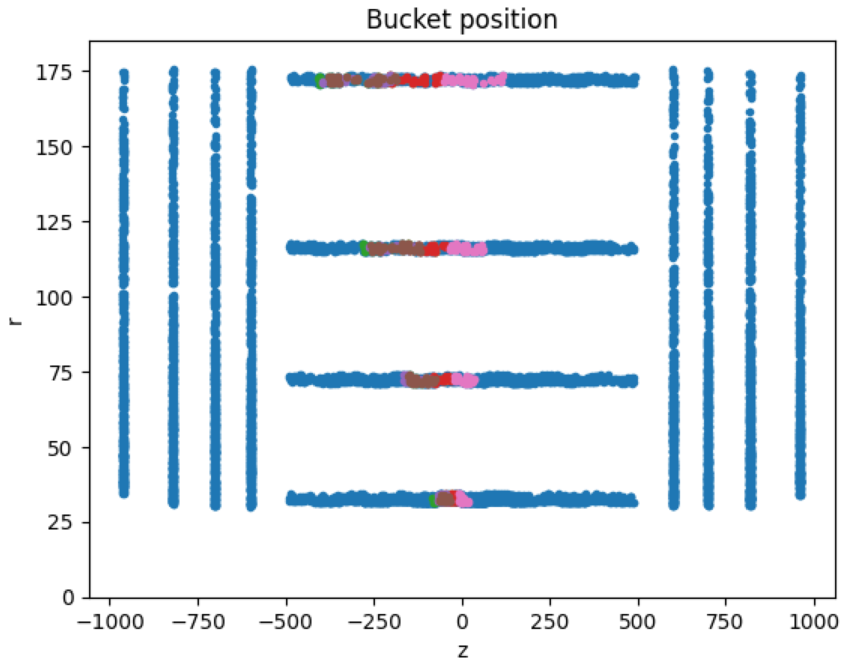
Other metric: ΔR

Angular: $\Delta\varphi$



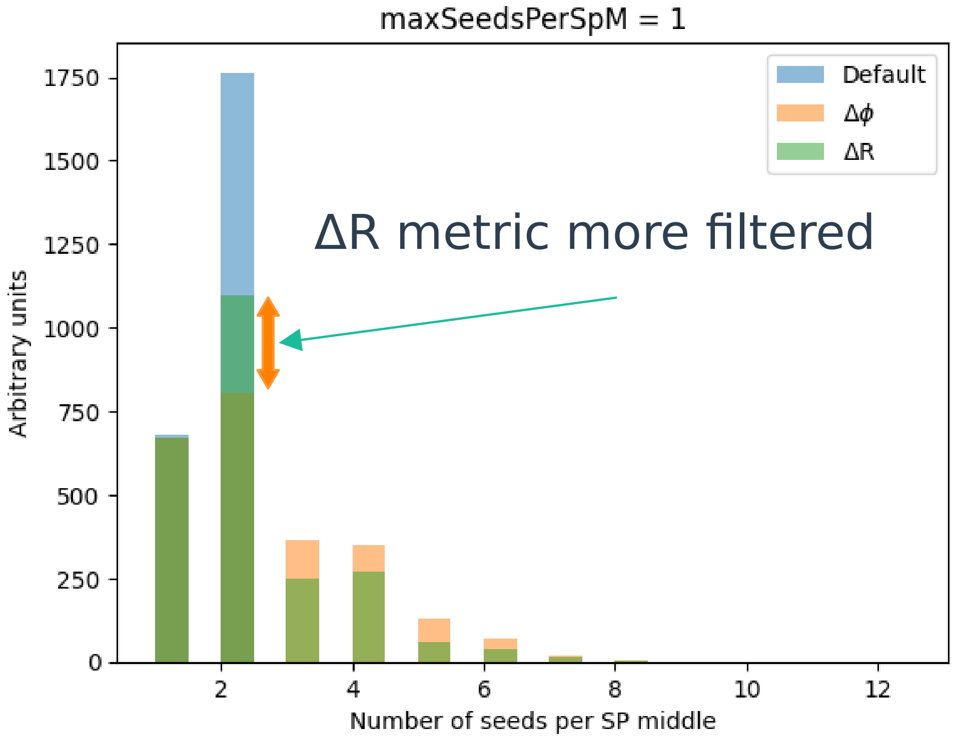
$\Delta R = \sqrt{(\Delta\varphi^2 + \Delta\eta^2)}$

If $\Delta\varphi > \pi$:
$\Delta\varphi = 2*\pi - \Delta\varphi$



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

Hashing and overlap

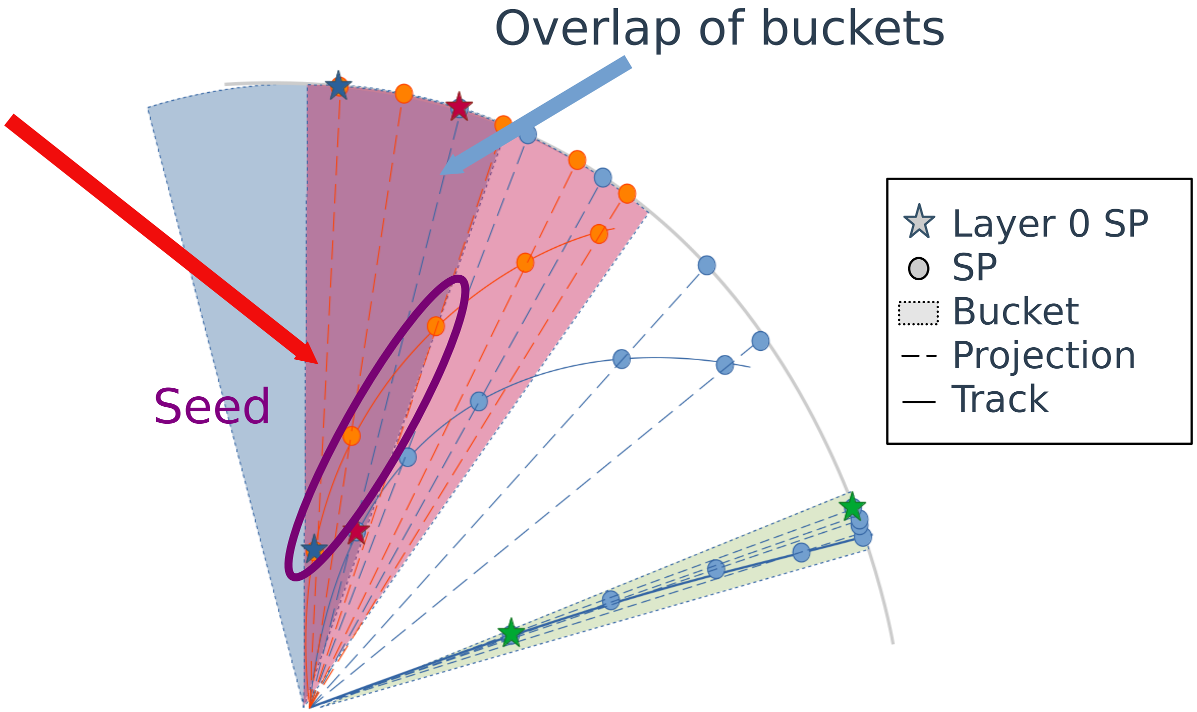
Hashing introduces overlaps:

- The same seed can be reconstructed in several buckets (14 times in average)

Generic detector

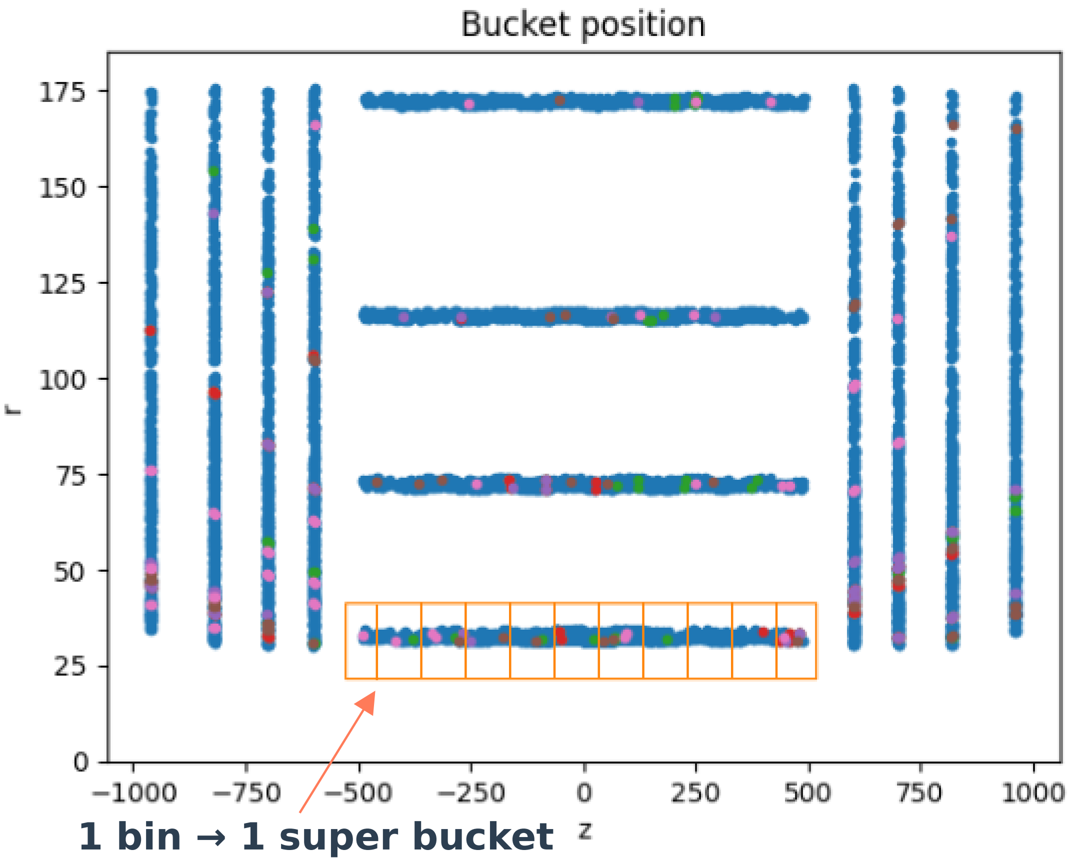
μ = 150	Timing/ event (ms)
Without Hashing	4491
With Hashing	7909

Hashing made timing x2



New idea: Group buckets → less overlap

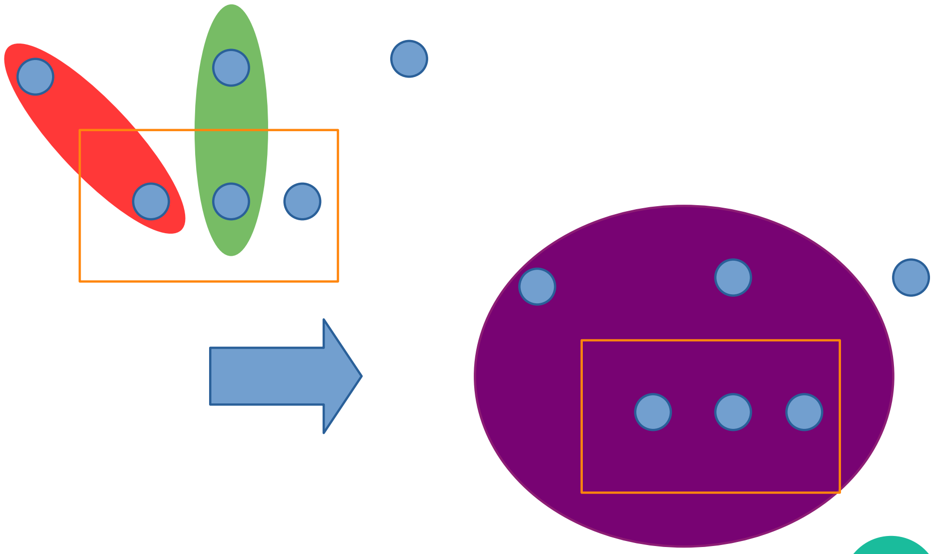
Super buckets and binning



1 bin → 1 super bucket

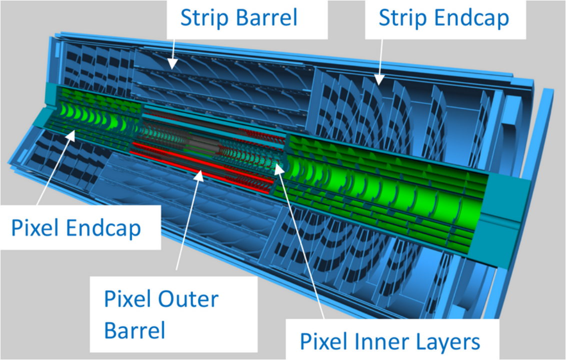
Super bucket:
Merging of the buckets created from the space points inside the z bin

→ **less overlap between buckets**



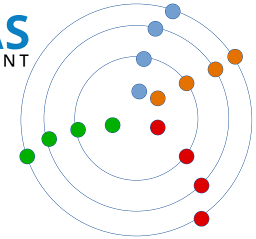
Realistic case

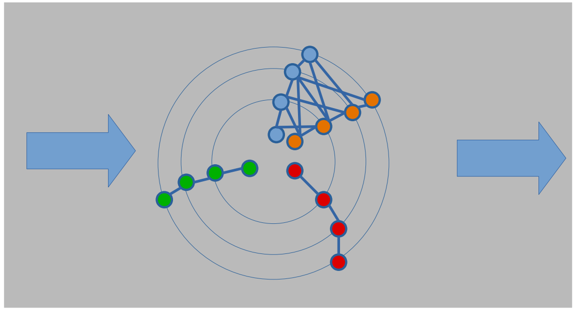
Inner Tracker (ITk)
(beeing built)

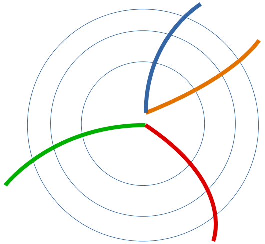


HL-LHC:
<μ> = 200

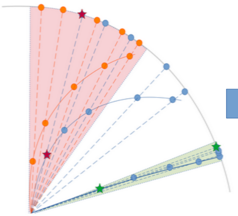
Official simulations
1000 $t\bar{t}$ events
μ = 60, 140, 200



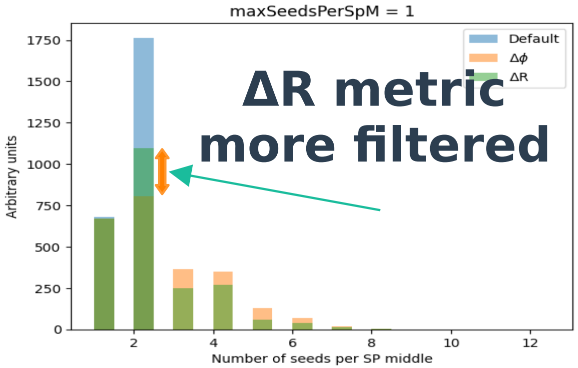




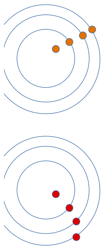
Combinatorics
→ **maxSeedsPerSpM=4**



Metric: Δφ



ΔR metric more filtered

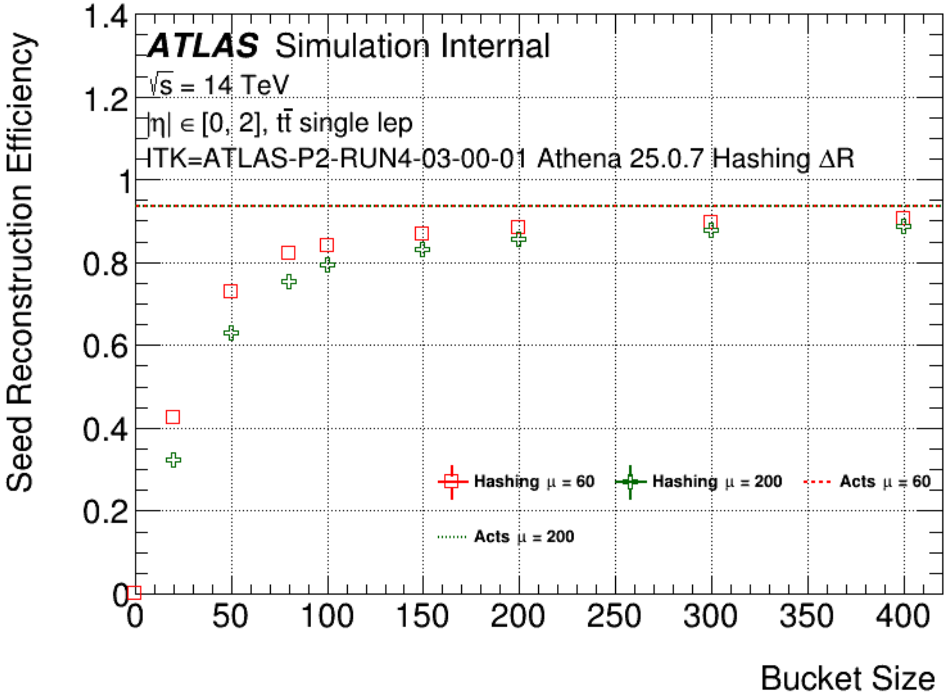


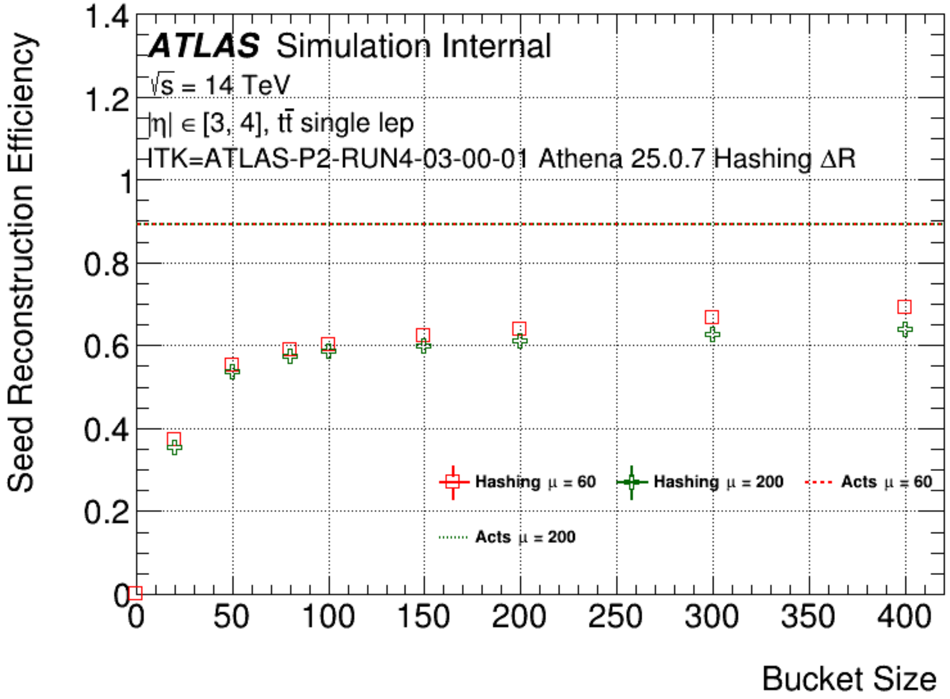
zation

Bucket Size ΔR: η

InnerTracker

WARNING: not only first layer selected

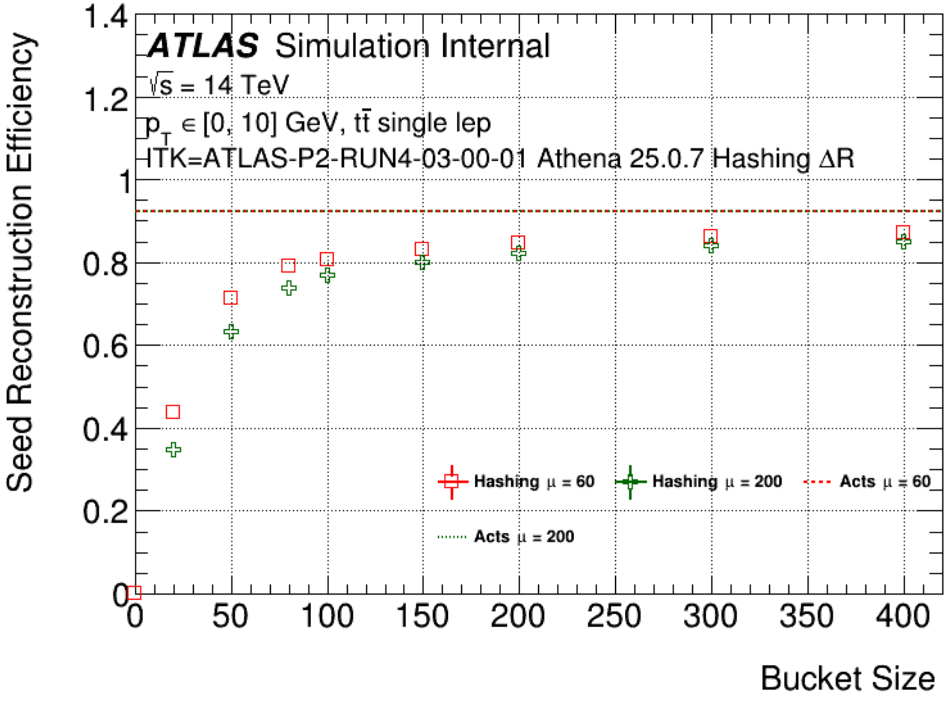


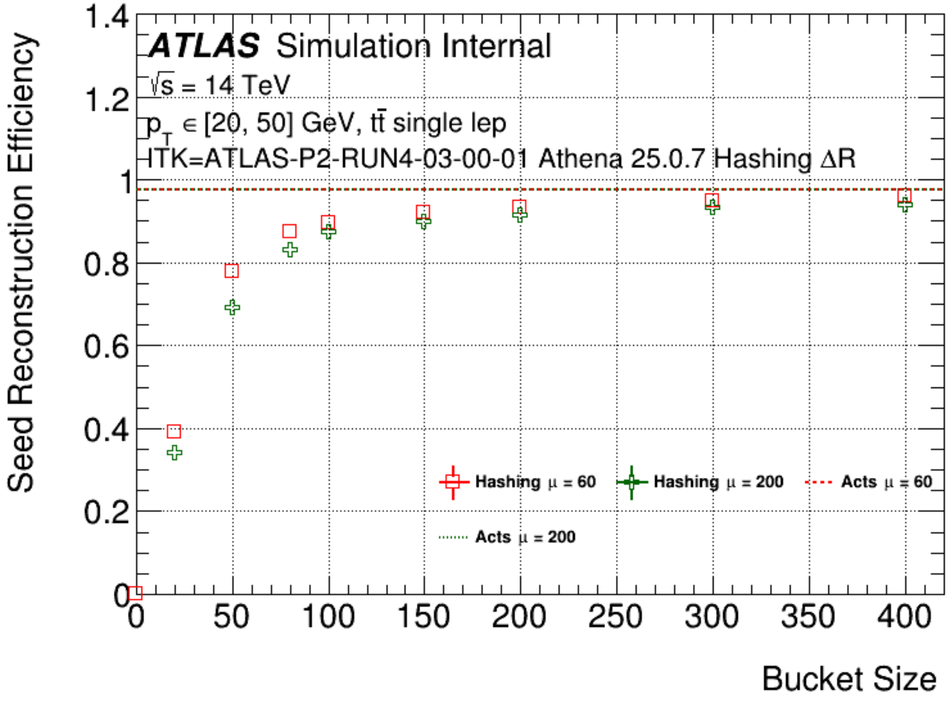


Bucket Size ΔR: pT

InnerTracker

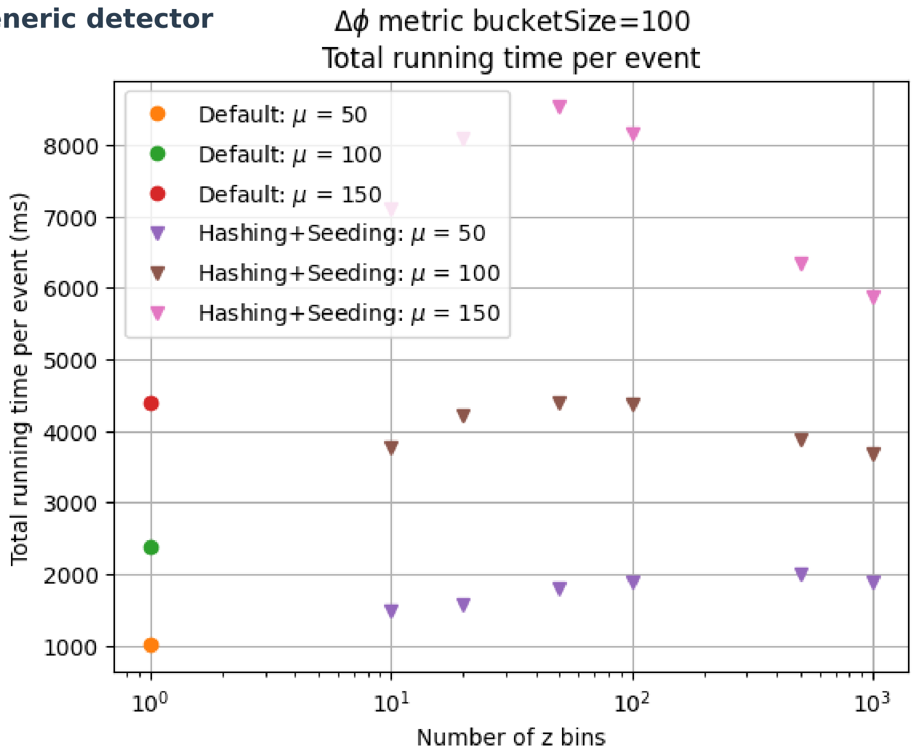
WARNING: not only first layer selected



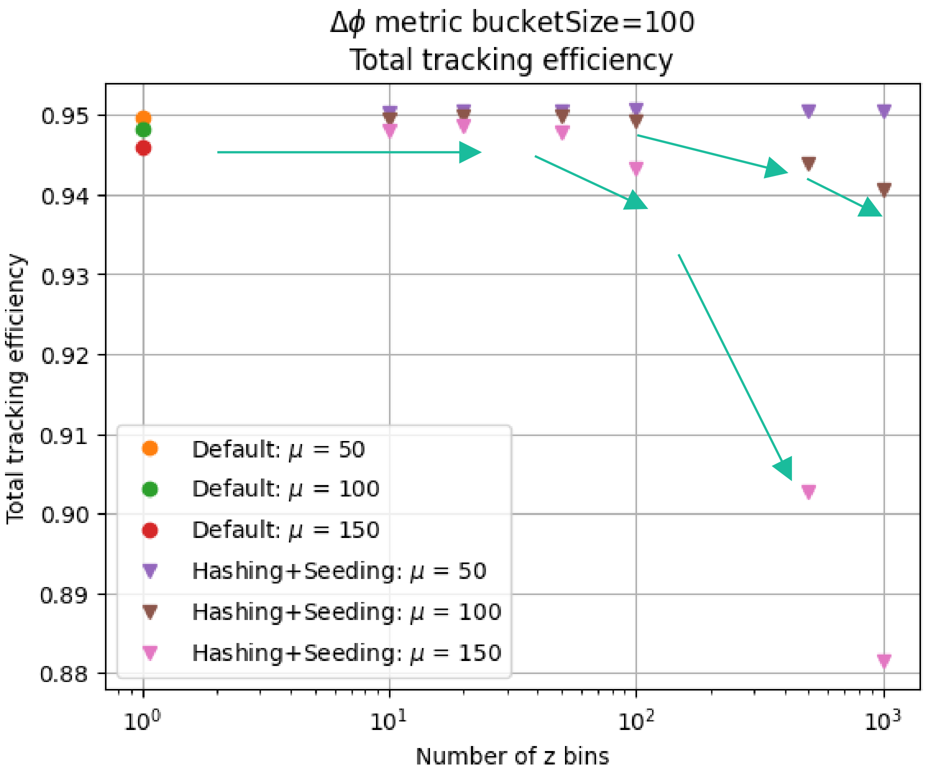


Hashing performance: Timing and efficiency

Generic detector



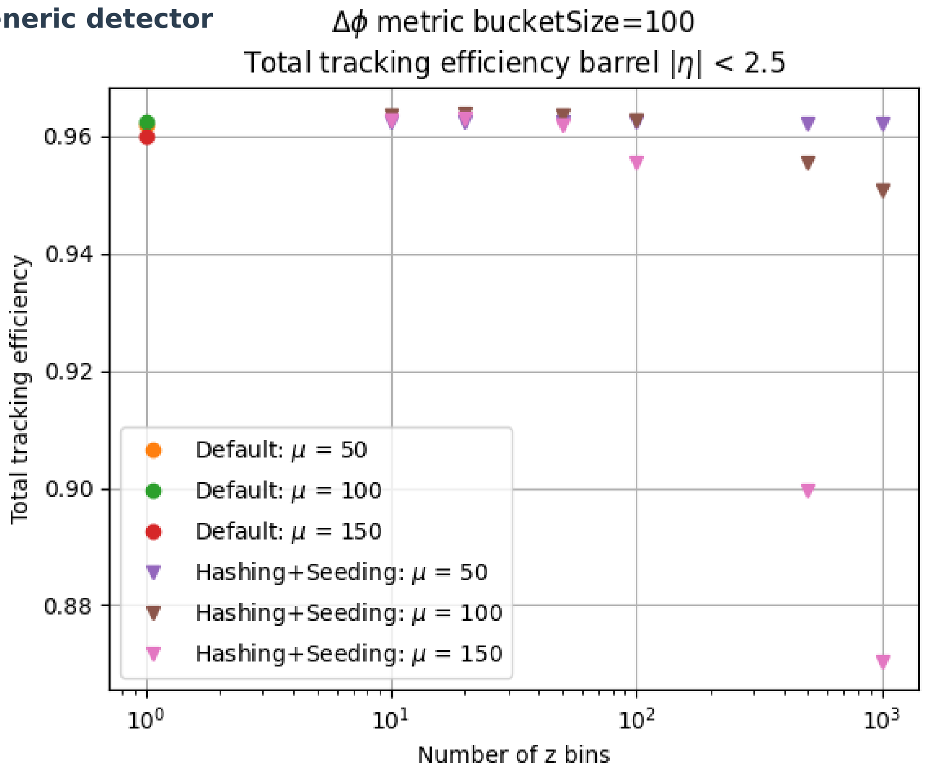
Running time ~ x2



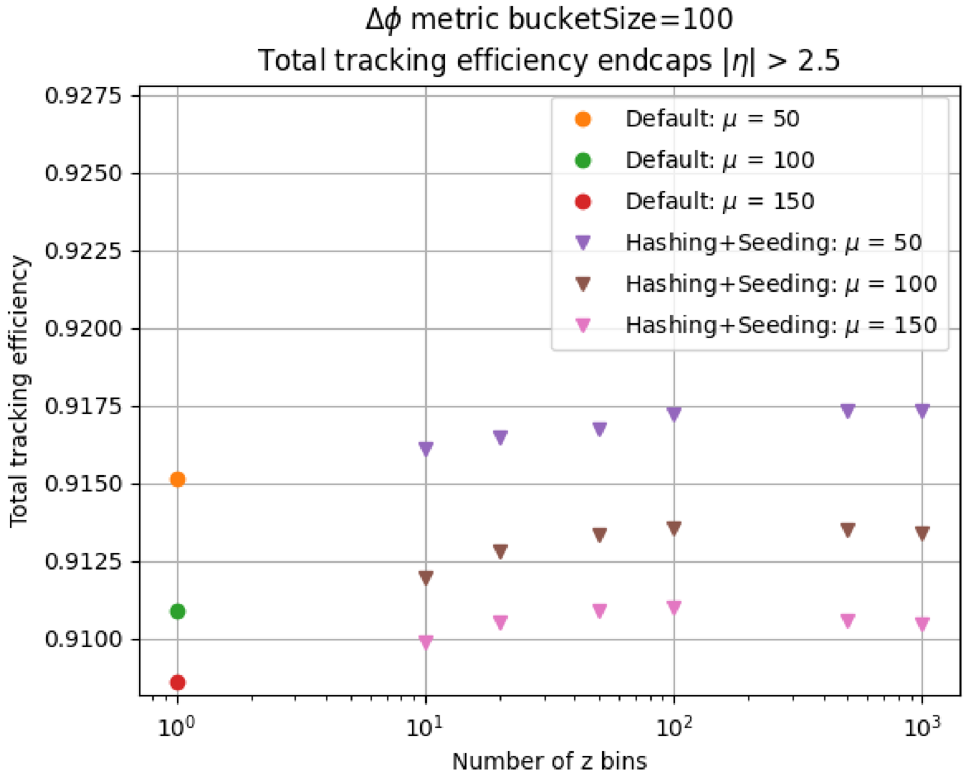
Improvement for small number of bins

Hashing performance: Efficiency (detailed)

Generic detector



Improves then drops

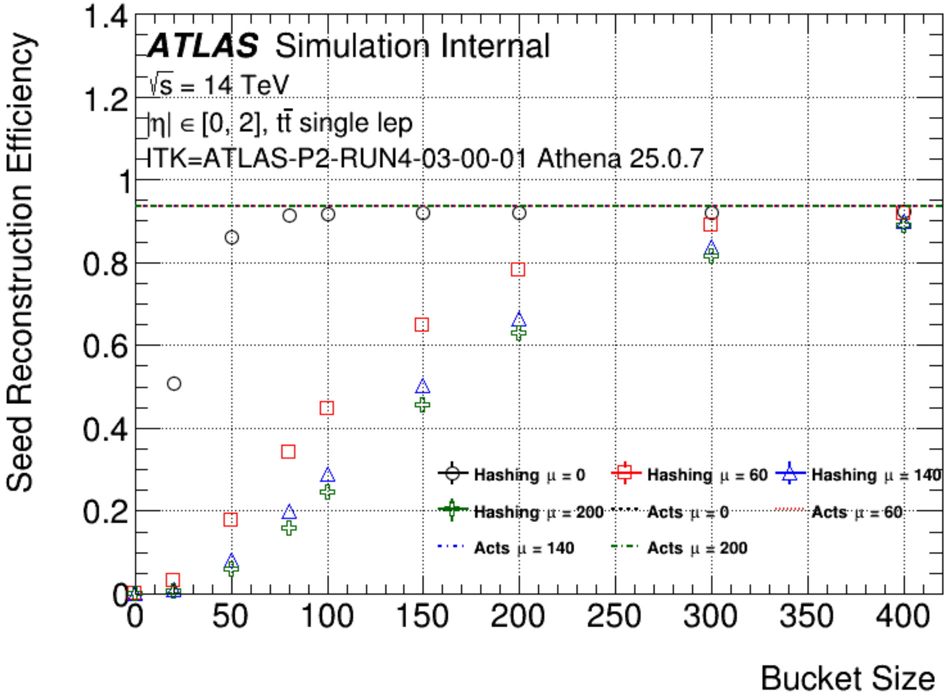


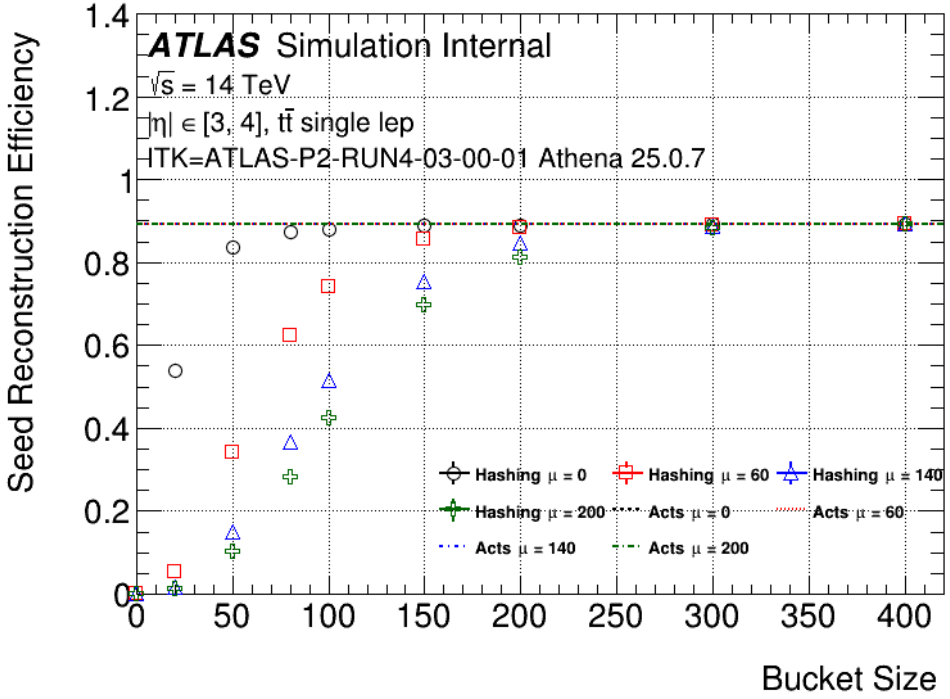
Always improve

Bucket Size Δφ: η

InnerTracker

WARNING: not only first layer selected

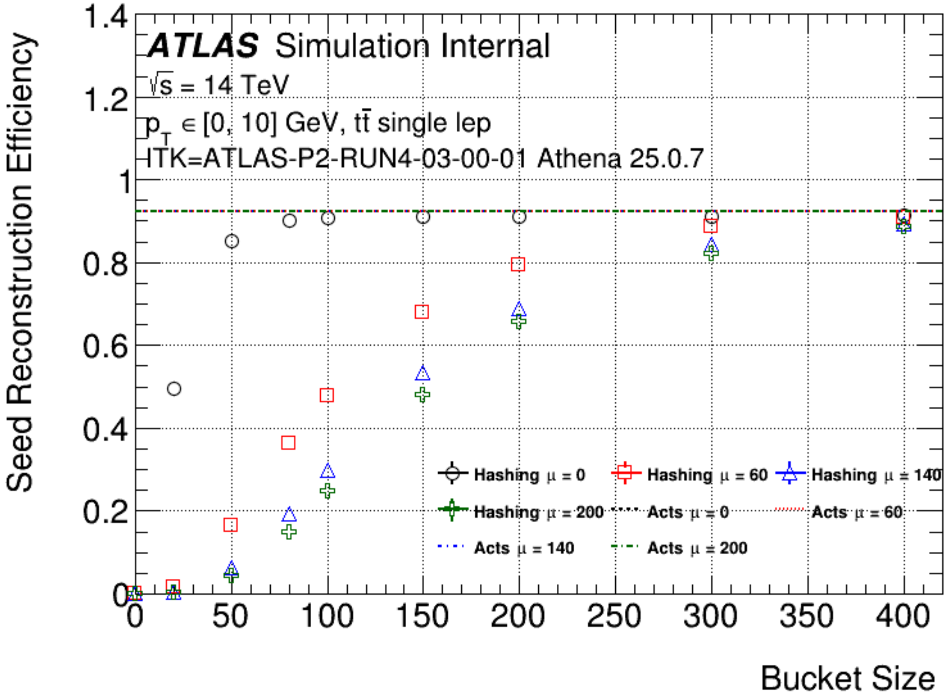


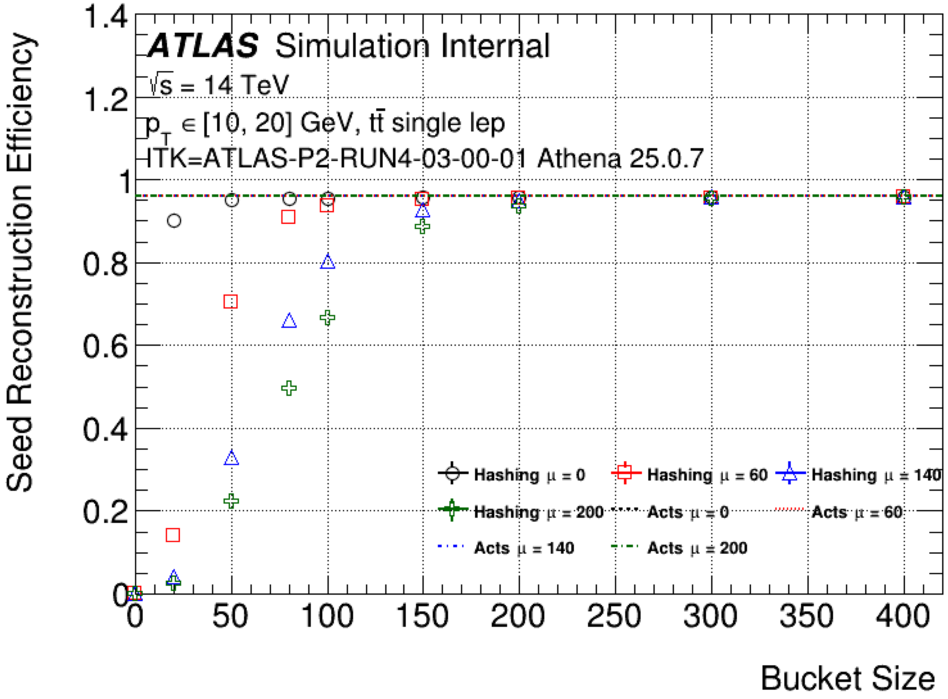


Bucket Size Δφ: pT

InnerTracker

WARNING: not only first layer selected

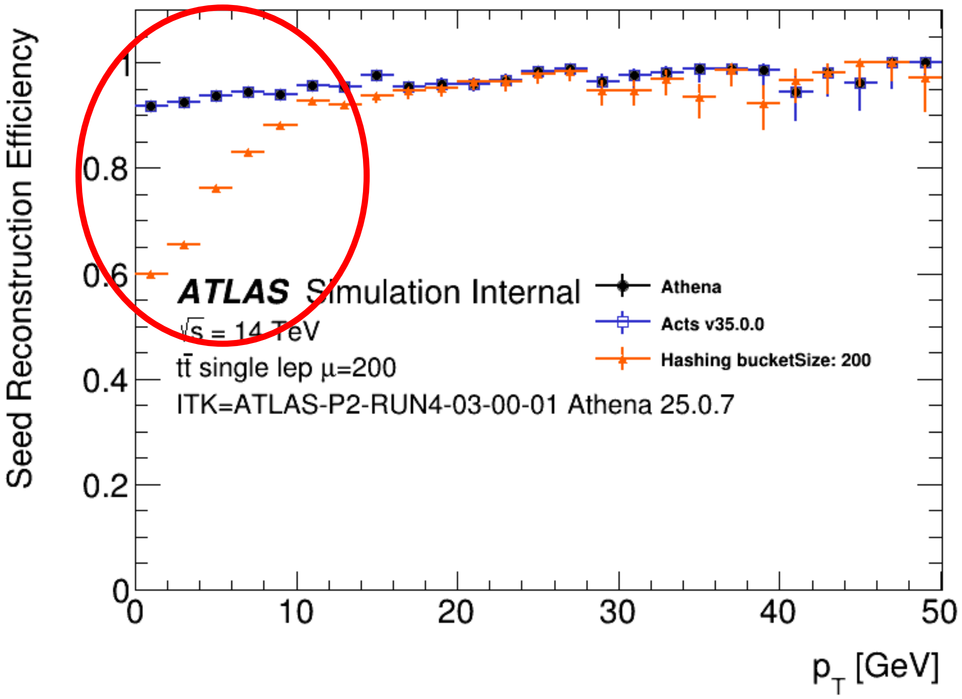


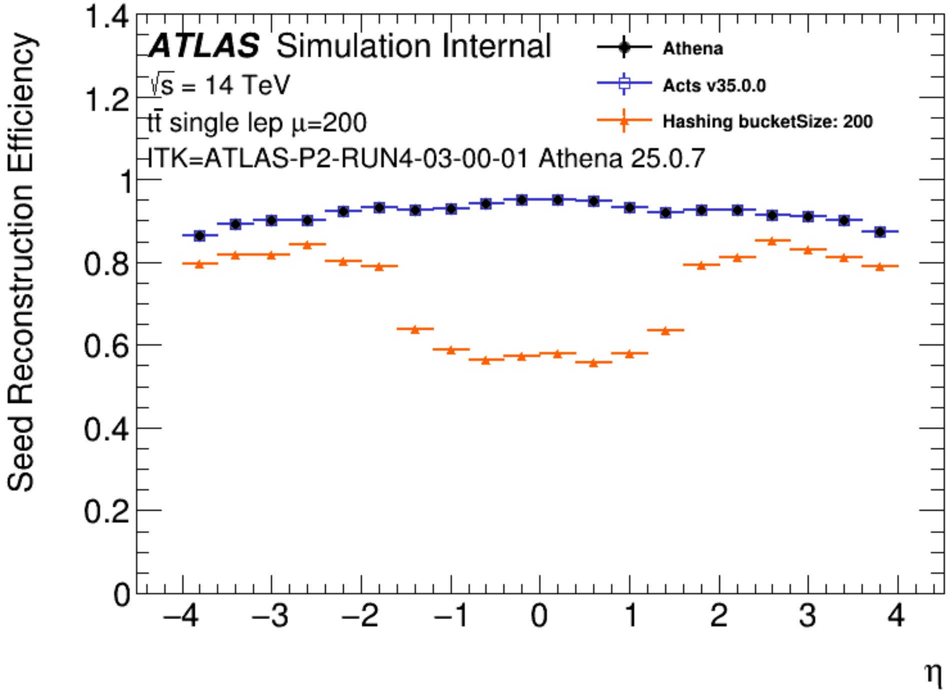


Δφ: Seed Efficiency μ=200

InnerTracker

WARNING: not only first layer selected

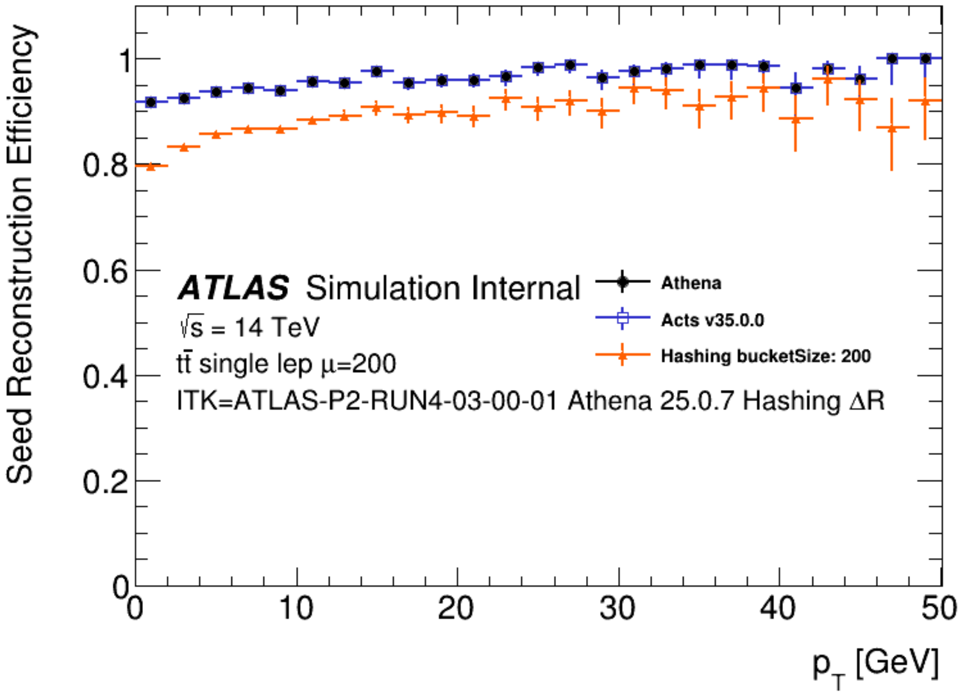


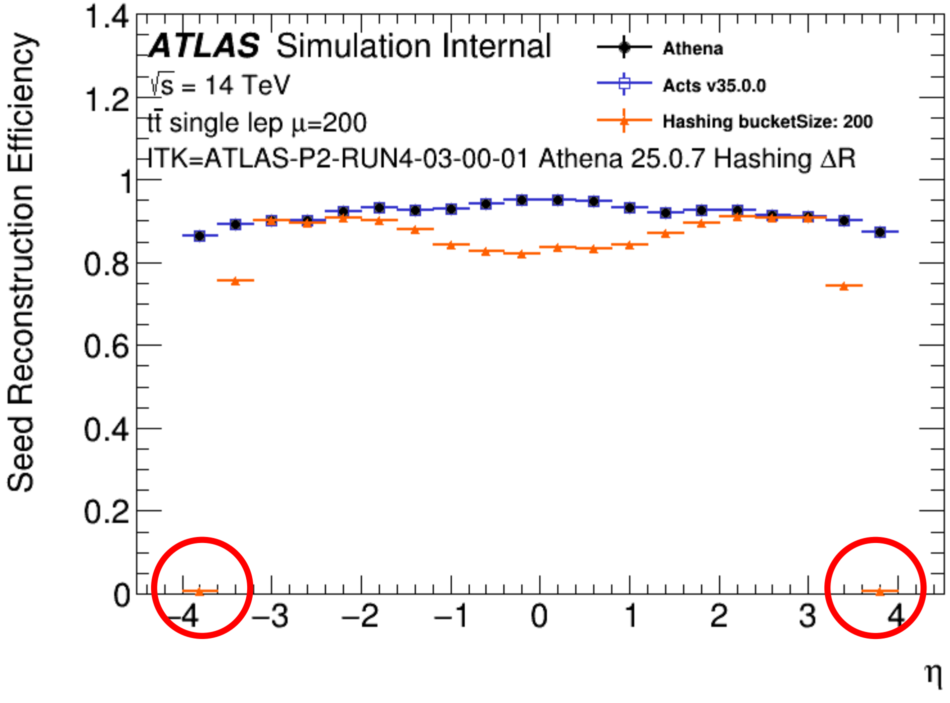


ΔR: Seed Efficiency μ=200

InnerTracker

WARNING: not only first layer selected

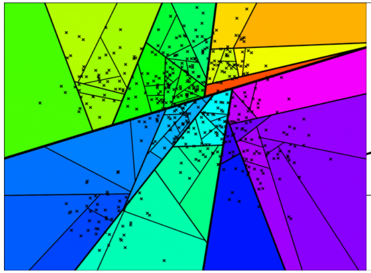




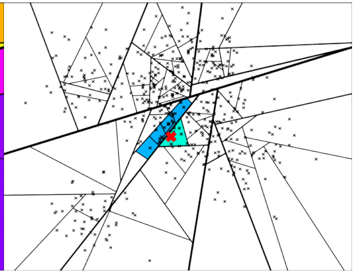
A new method:
Machine Learning/Hashing in the Seeding

Hashing:

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



Space separation



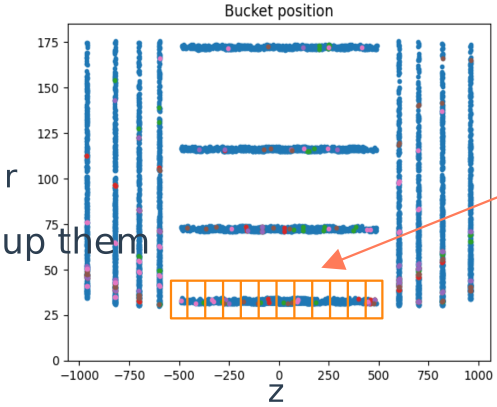
Look for neighbors in the closest regions

Algorithm used:

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

- Machine Learning algorithm type:
  - k Nearest Neighbors (unsupervised)
  - Random based
- Make bins in layer 0
- Find Neighbors of the points inside a bin and group them
  1 bin → 1 bucket

Application:



1) Make bins in layer 0
2) Find Neighbors of the points inside a bin and group them
   1 bin → 1 bucket
3) Do the seeding on the bucket