Clustering in SPD Range System

SPD SW&C group meeting,

2021/12/21

Georgy Golovanov

on behalf of SPD Muon Group

Outline

Two steps of particle identification in SPD Range System

- Clustering
- Particle tagging

Coordinate systems for clustering

Reference clustering in SpdRoot

Clustering algorithms

- k-Means algorithm
- DBSCAN-based algorithm

Model performance

- Purity and V-measure
- Muon/Pion efficiencies

Summary

Particle reconstruction in Range System

Main goal: to develop algorithms able to reconstruct muons and hadrons based on the information from Range System *standalone* by using machine learning techniques.

This is aimed to speed up the reconstruction process compared to traditional Kalman Filter technique.

Information available from Range System:

- two coordinate electronic readout: wires and strips — provides 3-dimension hits (thanks Alexander for strip emulation in SpdRoot);
- hits in **Barrel**: (x, y) of wires at layers and z of strips (30mm pitch);
- hits in **EndCaps**: (y, z) of wires and x of strips;

Reconstruction task is split on *two steps*:

Step 1: Clustering — forms group of hits (clusters) in Range System

- K-Means;
- DBSCAN;
- etc.

Step 2: Particle identification (muon/pion cluster tagging) — separates muon clusters from the clusters associated with hadrons

- Boosted Decision Tree algorithms (XGBoost, CatBoost);
- Convolutional Neural Network;
- etc.

Coordinate system for clustering

There is a choice of coordinate systems for clustering in RS:

1. (x,y) in Barrel + (z,y) in Endcaps *Wire readout*

+ natural for MDT wires orientation;
- needs cluster matching between Barrel and EndCaps.

2. (r,φ) in Barrel + (z,θ) in EndCaps

+ takes into account projection geometry;
- needs cluster matching between Barrel and EndCaps.

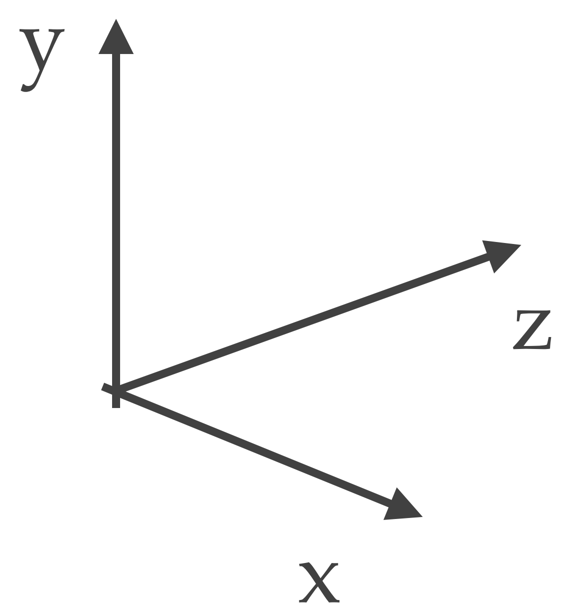
3. (θ,φ) or (η,φ) {hit map} *Wire/Strip readout*

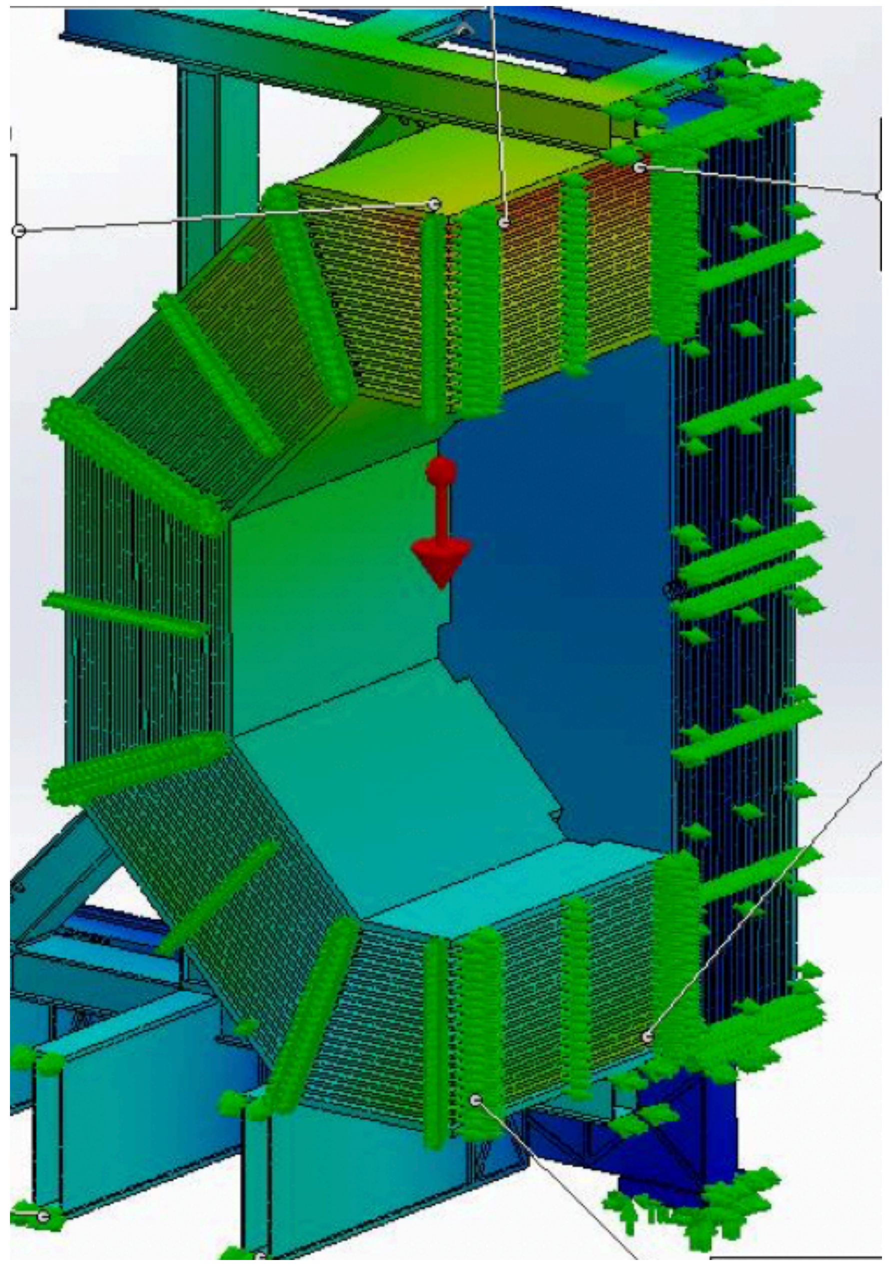
+ no cluster split between Barrel/EndCaps;
- clusters can be very dense since all hits are summed over 20 RS layers;

4. (x,y,z) {hit space}

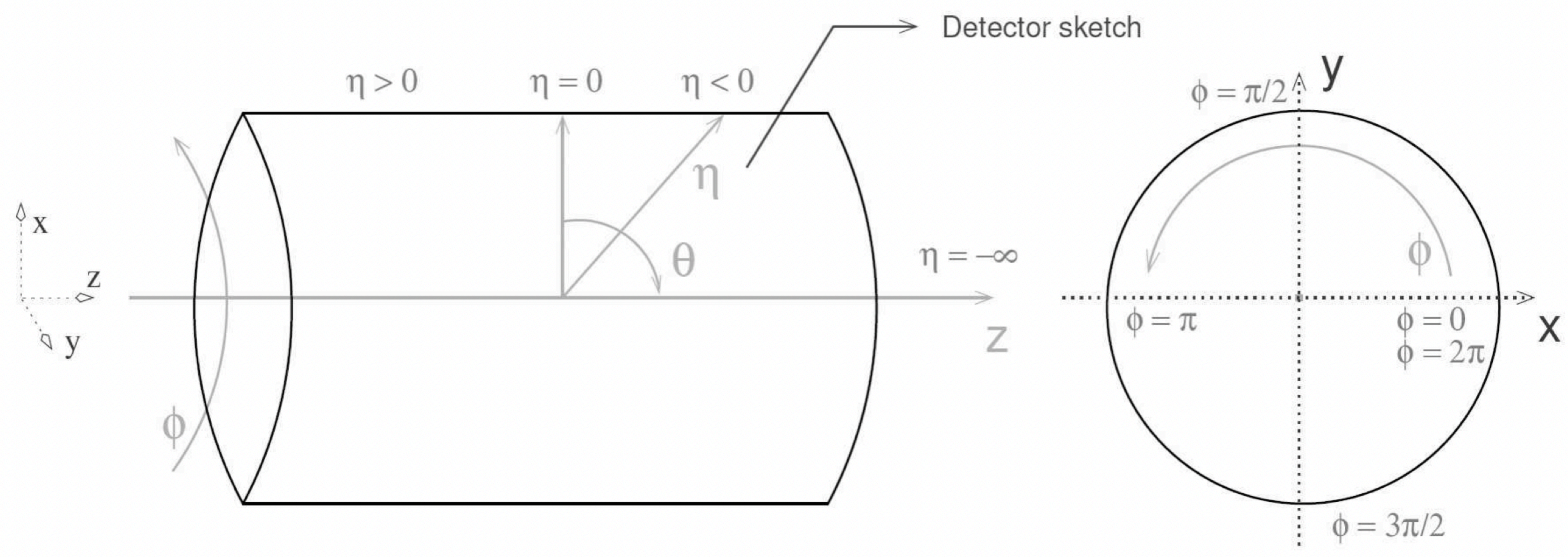
+ natural for MDT wires and strips orientation;
+ no cluster split between Barrel/EndCaps;
+ takes into account 3D shape of the cluster, no overlapping;
- algorithms may be slower in 3D-space compared to 2D;

The chosen coordinate system should be in agreement with the later pion/muon separation algorithms, since they will use found clusters as an input.





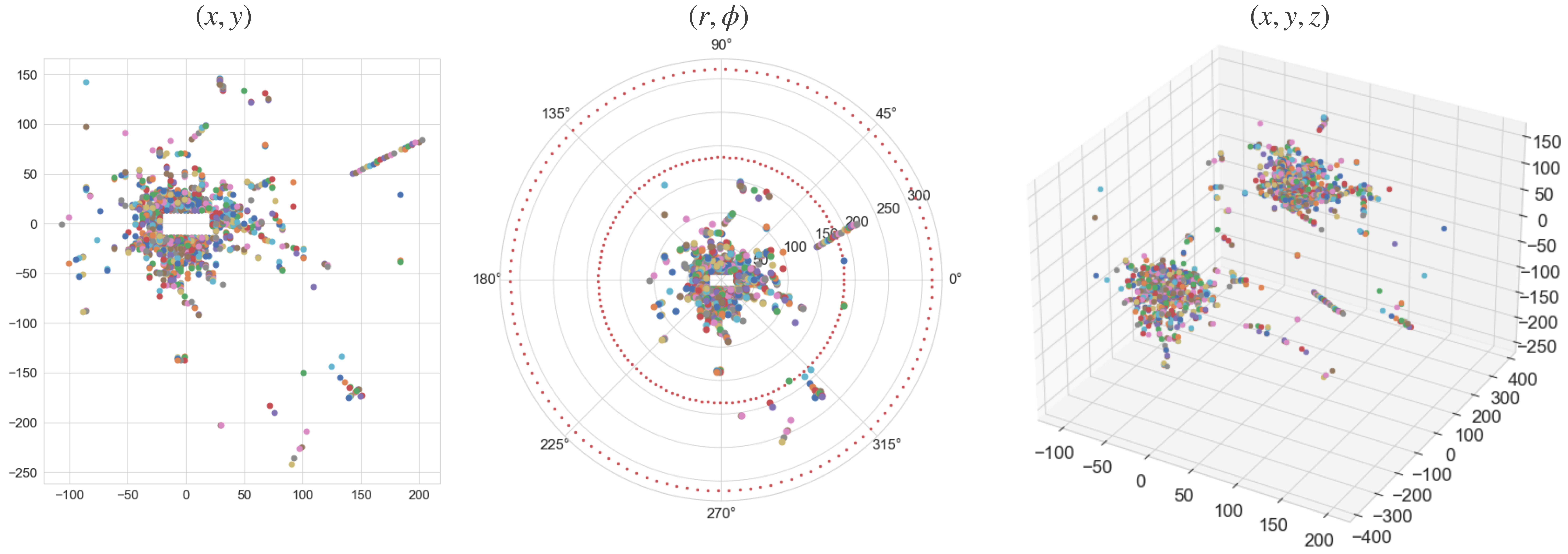
Cross section of the SPD RS



PoS (ACAT) 055

Coordinate system for clustering

A single event represented in various coordinate systems



(x, y)

(r, ϕ)

(x, y, z)

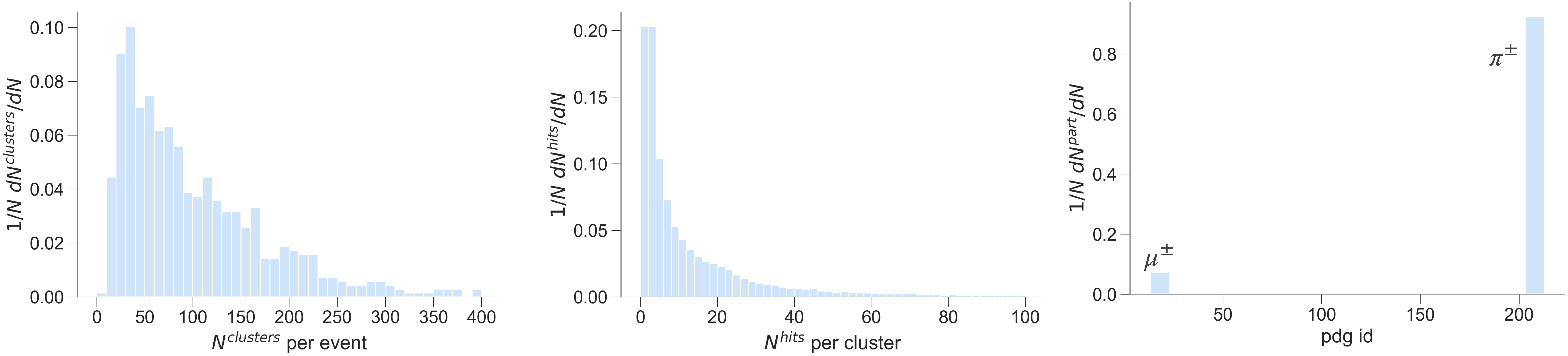
Increased density in 2-dimenstion systems due to hit overlapping

Reference clustering in SpdRoot

Reference clusters are defined as group of hits in Range System produced by a particle in front the RS (thanks Artur for SpdRoot realization).

Should provide ground truth for clustering algorithm evaluation metrics.

- can be up to 400 clusters/event $\Longrightarrow$ high hit density events (mostly in forward region);
- single hits clusters: ~40% clusters have $\leq$4 hits $\Longrightarrow$ might be a problem for clustering, considered as noise;
- most pion (hadron) clusters are originated in the ECal;



Based on 5k $J/\psi \to \mu\mu$ Pythia8 MC sample from SpdRoot reference example by Igor

RS Clustering: k-Means

Clustering is one of the *unsupervised machine learning* techniques.

Hard clustering (each hit belongs to a cluster or not) algorithms are considered.

Only geometrical informations is available in Range System
(unlike ECAL no energy is provided $\Longrightarrow$ unable to use *SimpleCone, kT, anti-kT, etc.* algorithms based on the energy deposition in the tower/cell).

Centroid-based algorithm k-Means:

- Select a number of clusters to use and randomly initialize their respective center points;
- Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it;
- Recompute the group center by taking the mean of all the vectors in the group;
- Repeat these steps for a set number of iterations or until the group centers don't change much between iterations.

Advantage: Fast, linear complexity $O(n)$
Downside: Need to know number of clusters in advance

Approximation: a number of hits in the 1st layer of RS is taken as number of clusters;

This parameter should not be explicitly defined beforehand!

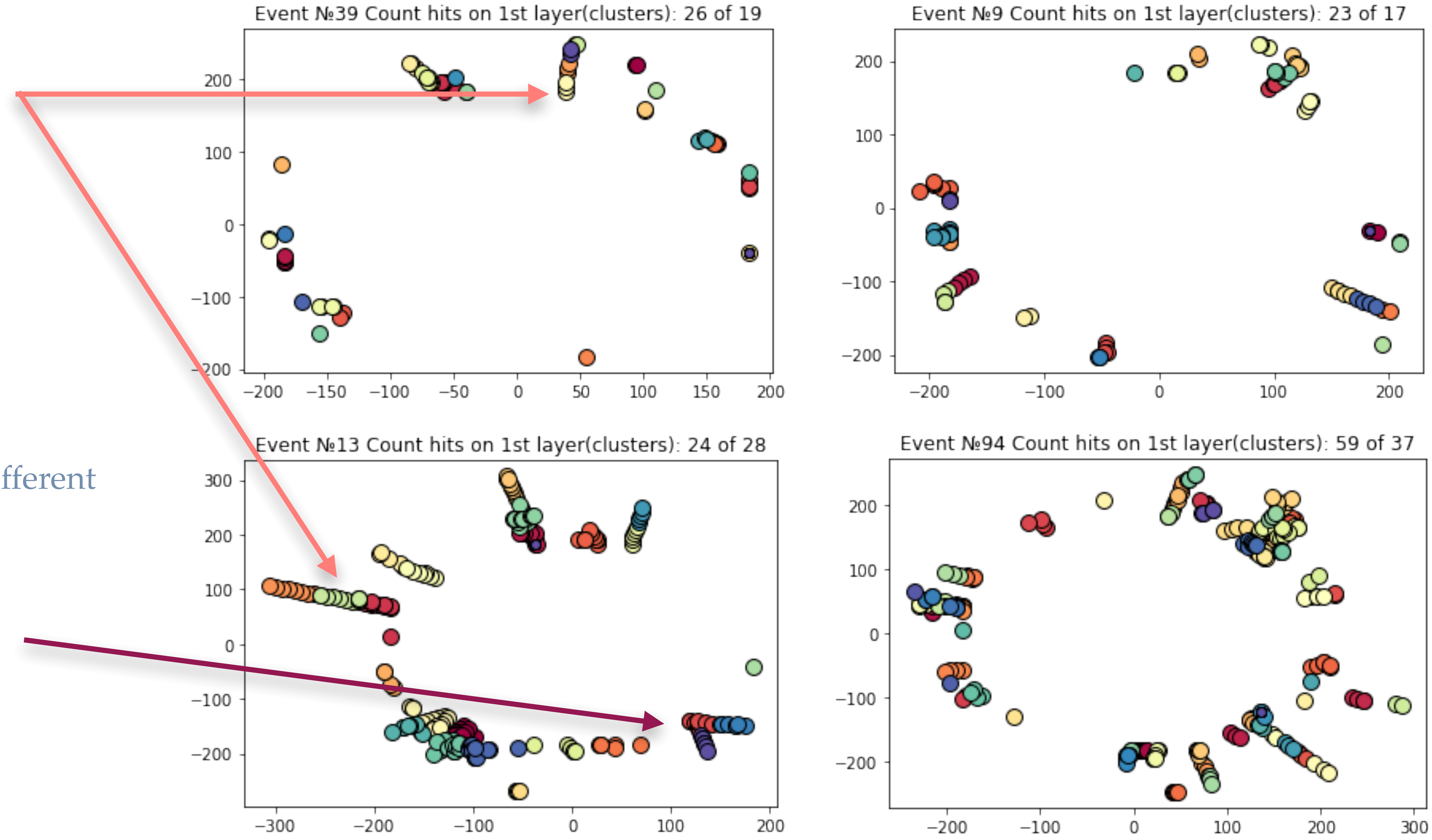
RS Clustering: k-Means (cont'd)

Splits muon cluster into segments, otherwise includes a lot of single hits

⟹ as a result, predicted Nclust is overestimated

Various colours show different clusters

Splits hadron showers



(x,y) view in the Barrel

RS Clustering: DBSCAN

DBSCAN (Density-based Spatial Clustering Application with Noise)

- views clusters as areas of high density separated by areas of low density;
- clusters found by DBSCAN can be any shape, as opposed to k-Means which assumes that clusters are convex shaped;
- two hyper-parameters: *min_samples* and *epsilon;*
- defines a **core sample** as being a sample in the dataset such that there exist *min_samples* other samples within a distance of *epsilon*, which are defined as neighbors of the core sample.
- A cluster also has a set of **non-core samples**, which are samples that are neighbors of a core sample in the cluster but are not themselves core samples. Hits that cannot be reached are defined as **outliers** and considered as noise;
- Parameter *min_samples* primarily controls how tolerant the algorithm is towards noise (on noisy and large data sets it may be desirable to increase this parameter), usually set to *min_samples>Ndim+1*
- Parameter *epsilon* controls the local neighborhood of the points: optimal *epsilon* is about 2x÷3x distance between layers to take into account missing hits on some layers;

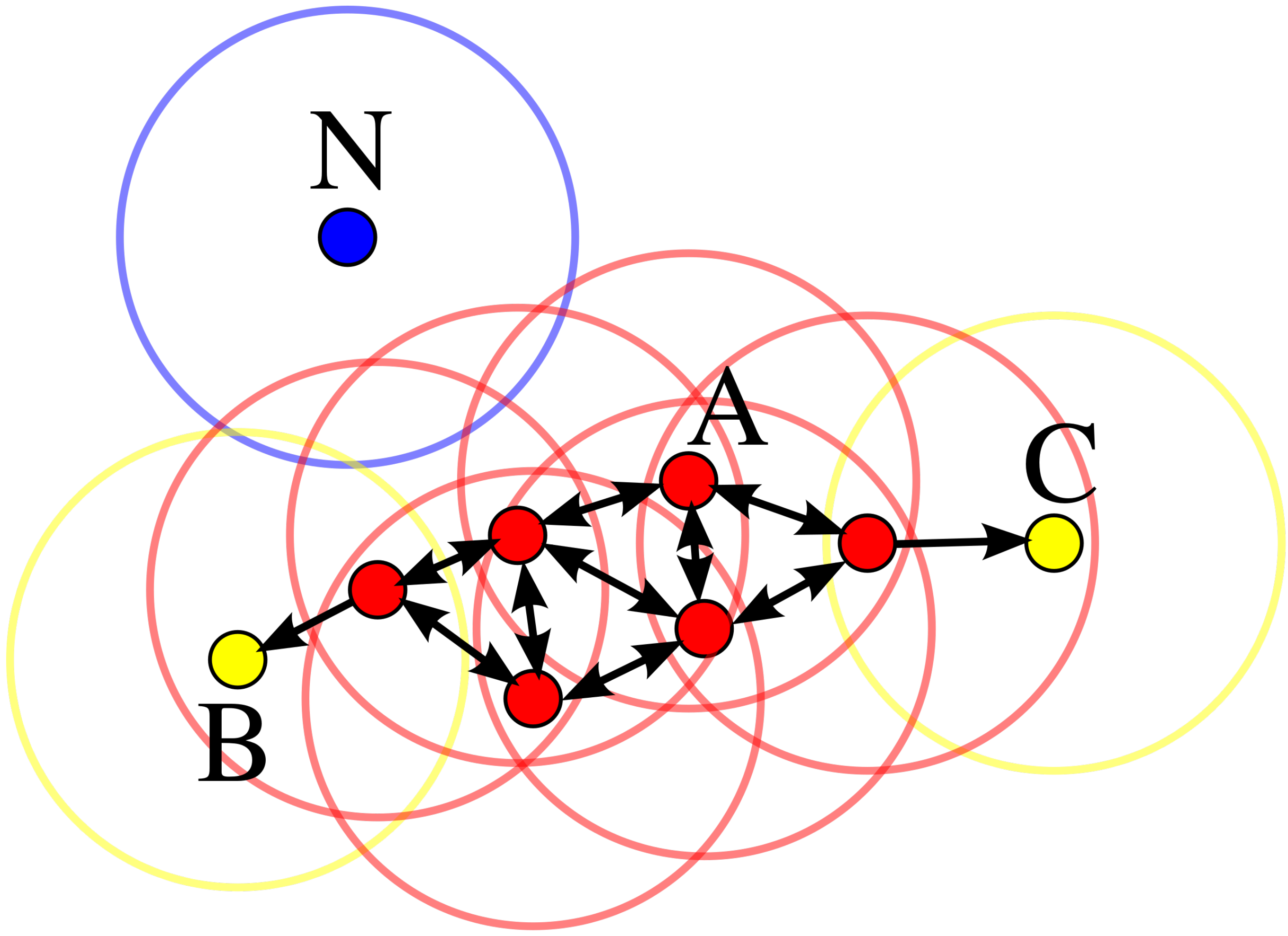
Advantages:

- it does not require a pre-set number of clusters at all;
- it also identifies outliers as noise;
- associates hits to clusters of arbitrary shapes;
- naturally exploits 3D coordinates of hits (wire/strips);

Downsides:

- it doesn't perform as well as others when the clusters are of varying density
- has two parameters to be optimized.

"A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise" Ester, M., H. P. Kriegel, J. Sander, and X. Xu, In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, AAAI Press, pp. 226–231. 1996



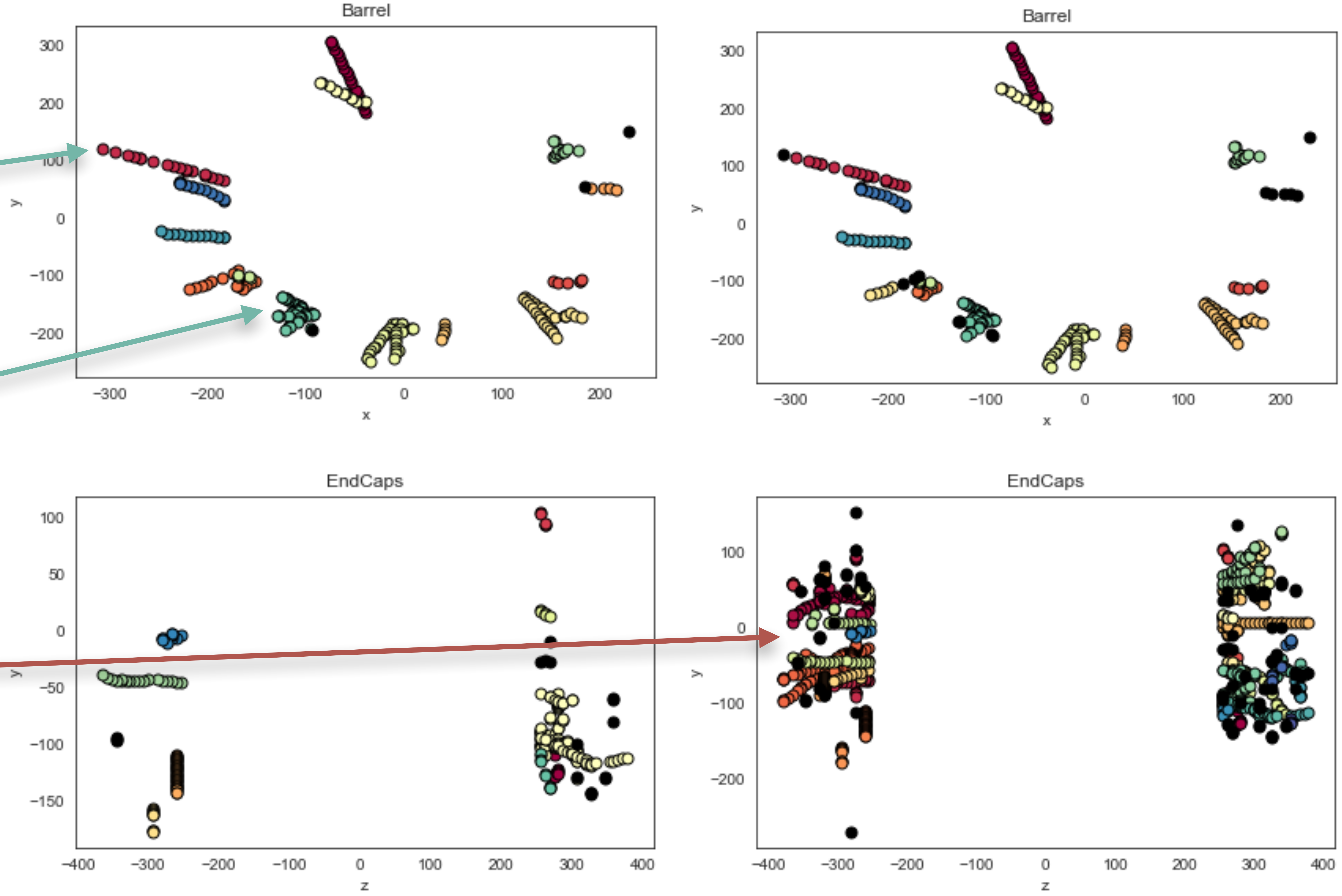
A and red points - core hits;
B,C - not core hits, but reachable from A via red hits;
N - noise.

RS Clustering: DBSCAN (cont’d)

Detects muons even with missing hits on some layers

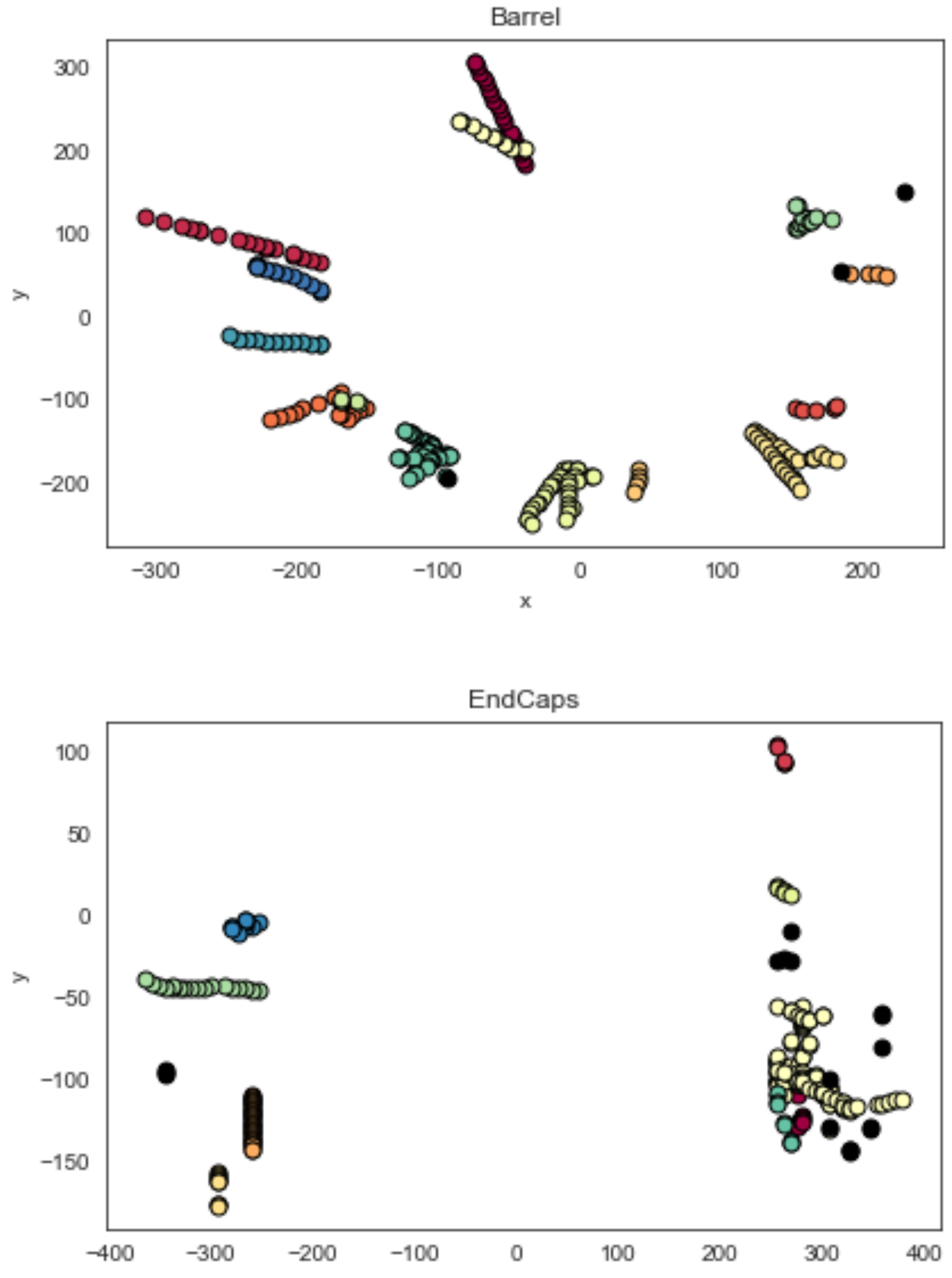
Detects hadron showers

Needs proper hyper-parameter tuning in high density regions



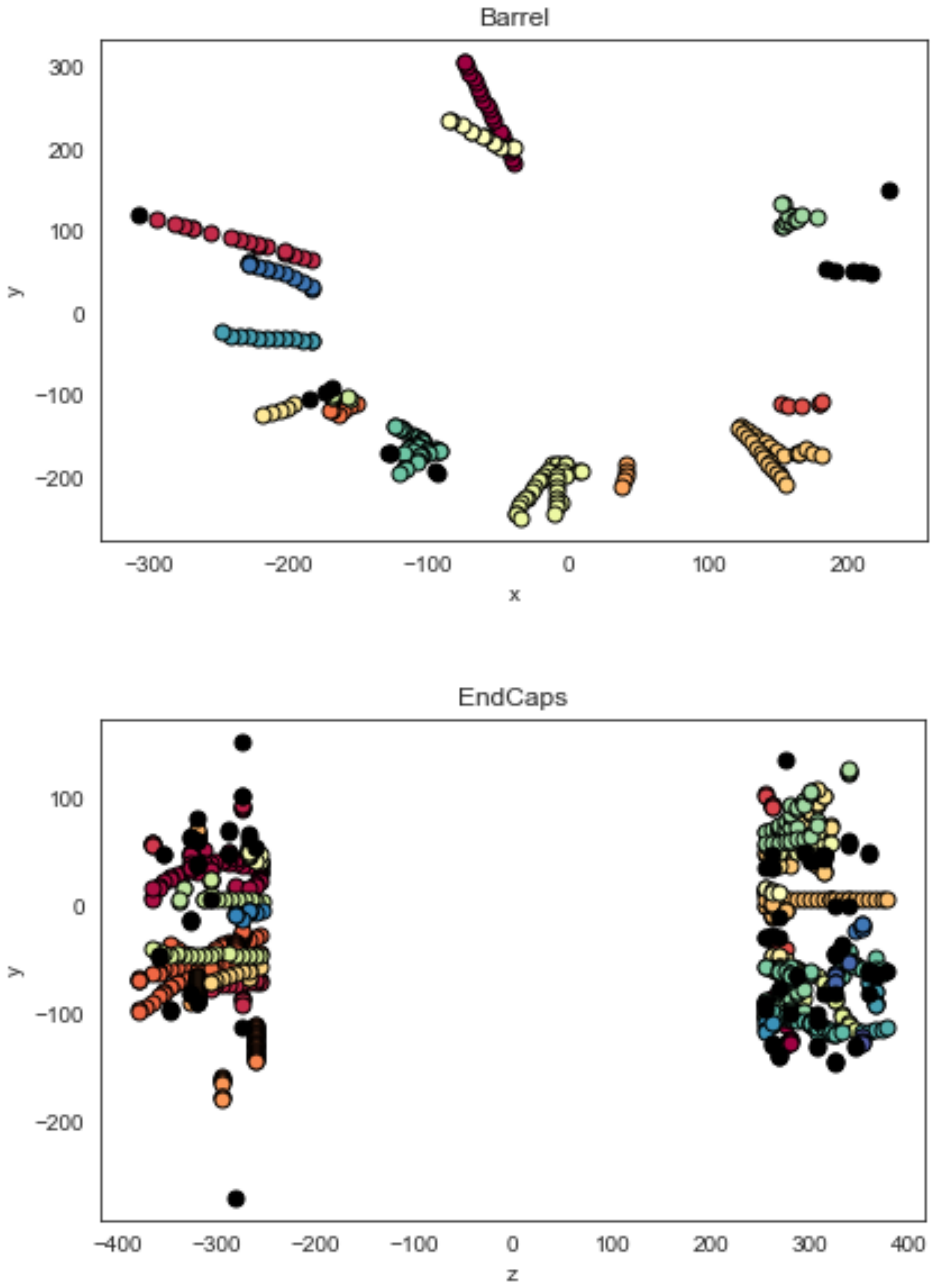
DBSCAN in Barrel and EndCaps

$\pi/8 < \theta < 7\pi/8$



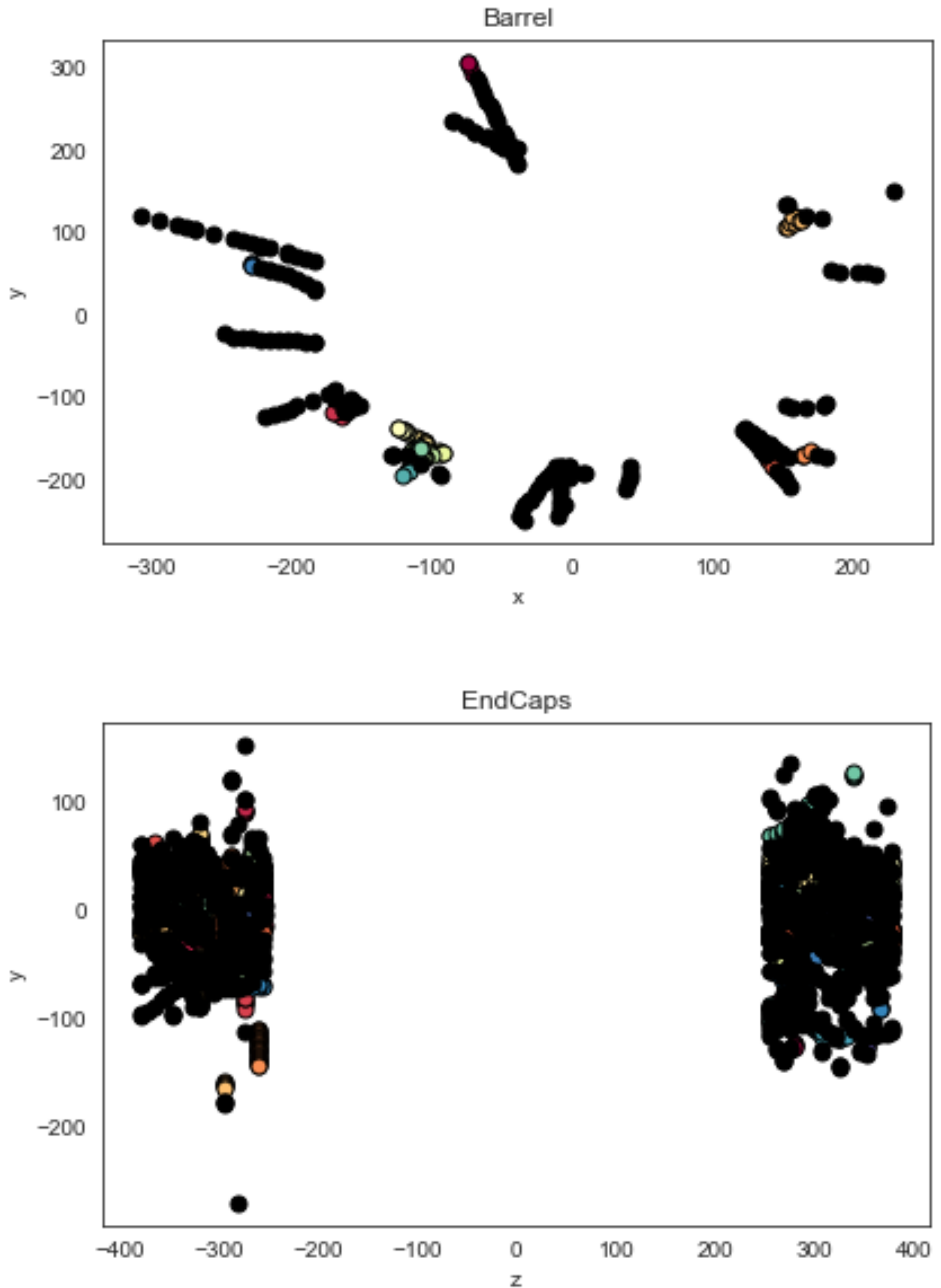
Igor used $|Cos\theta| < 0.9$ cut in his analysis, similar to this threshold

$\pi/16 < \theta < 15\pi/16$



Approx. corresponds to $|\eta| < 2.4$ regularly used by J/psi analysis

all

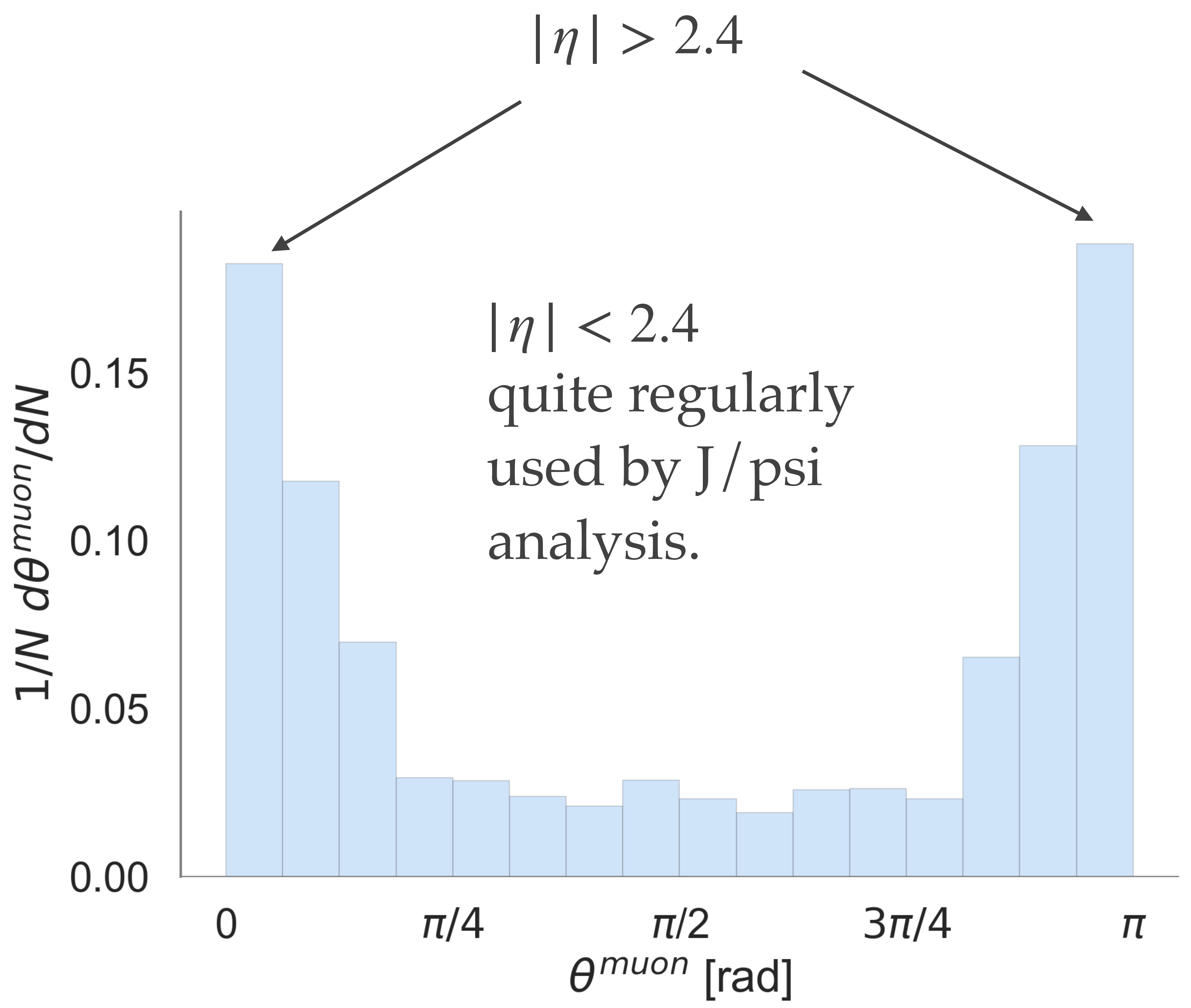


Large amount of hits concentrated around the beam pipe leads the algorithm to fail

θ-Threshold

By applying the threshold on θ (η) polar angle to get rid of dense very forward region (around beam pipe): how much muon candidates (true muon clusters) are we going to lose?

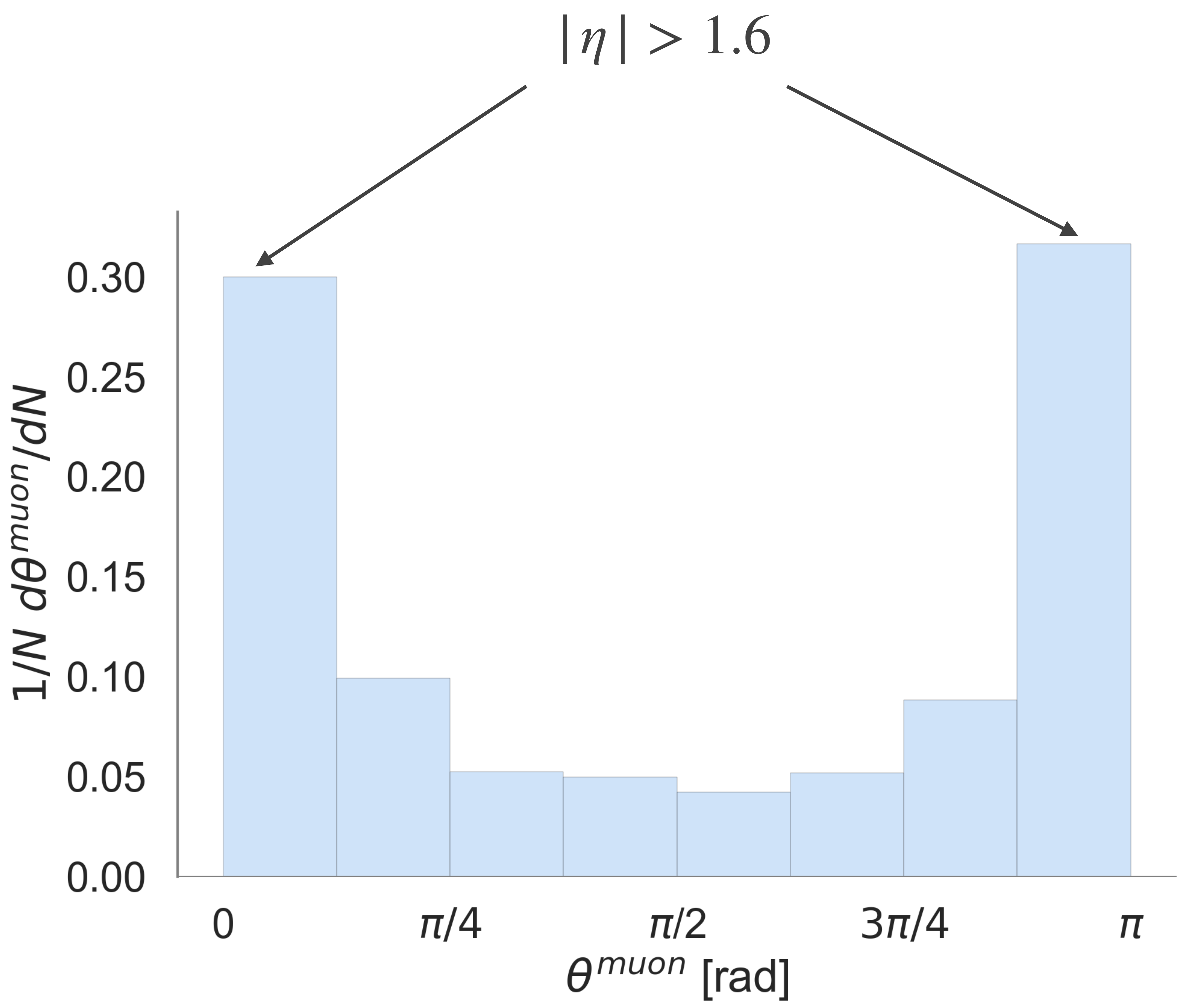
Pseudorapidity: $\eta = -\ln[\tan(\theta/2)]$



$|\eta| > 2.4$

$|\eta| < 2.4$
quite regularly
used by J/psi
analysis.

About 30% muons loss



$|\eta| > 1.6$

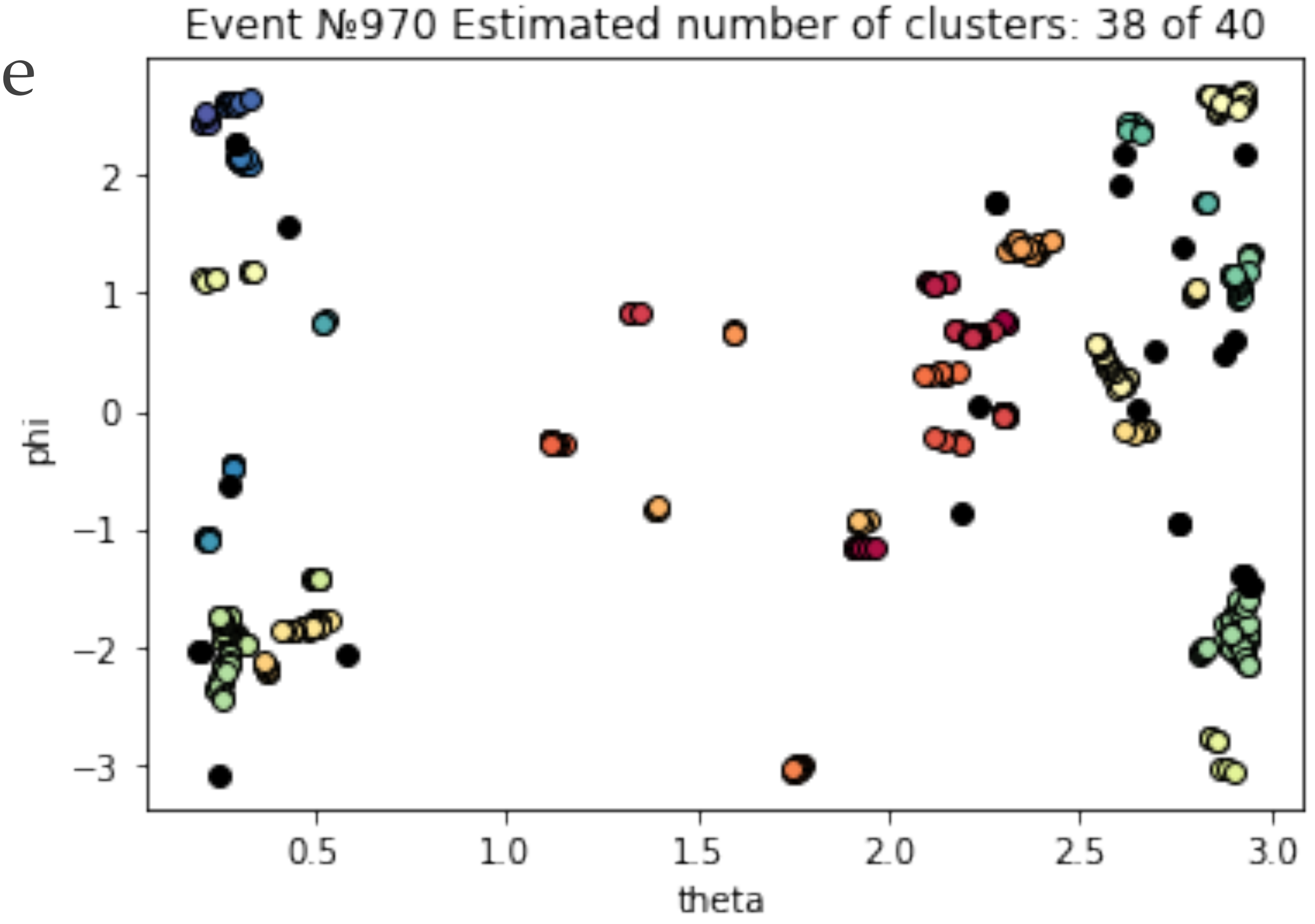
About 60% muons loss

Soft, low transverse momenta muons are expected around the beam pipe at the very forward region.

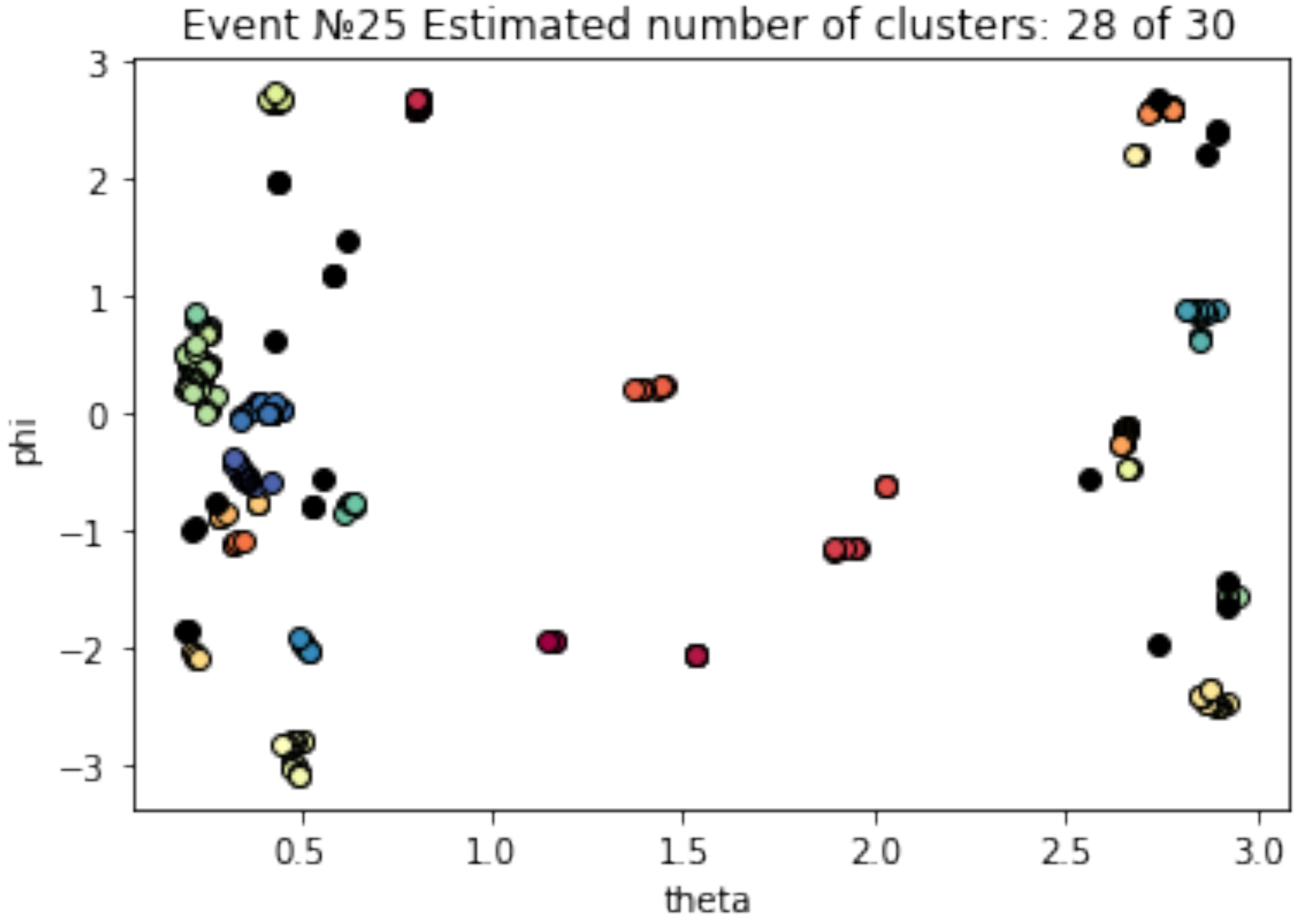
DBSCAN-based algorithm

DBSCAN application in the (θ,φ) space

$\pi/16 < \theta < 15\pi/16$

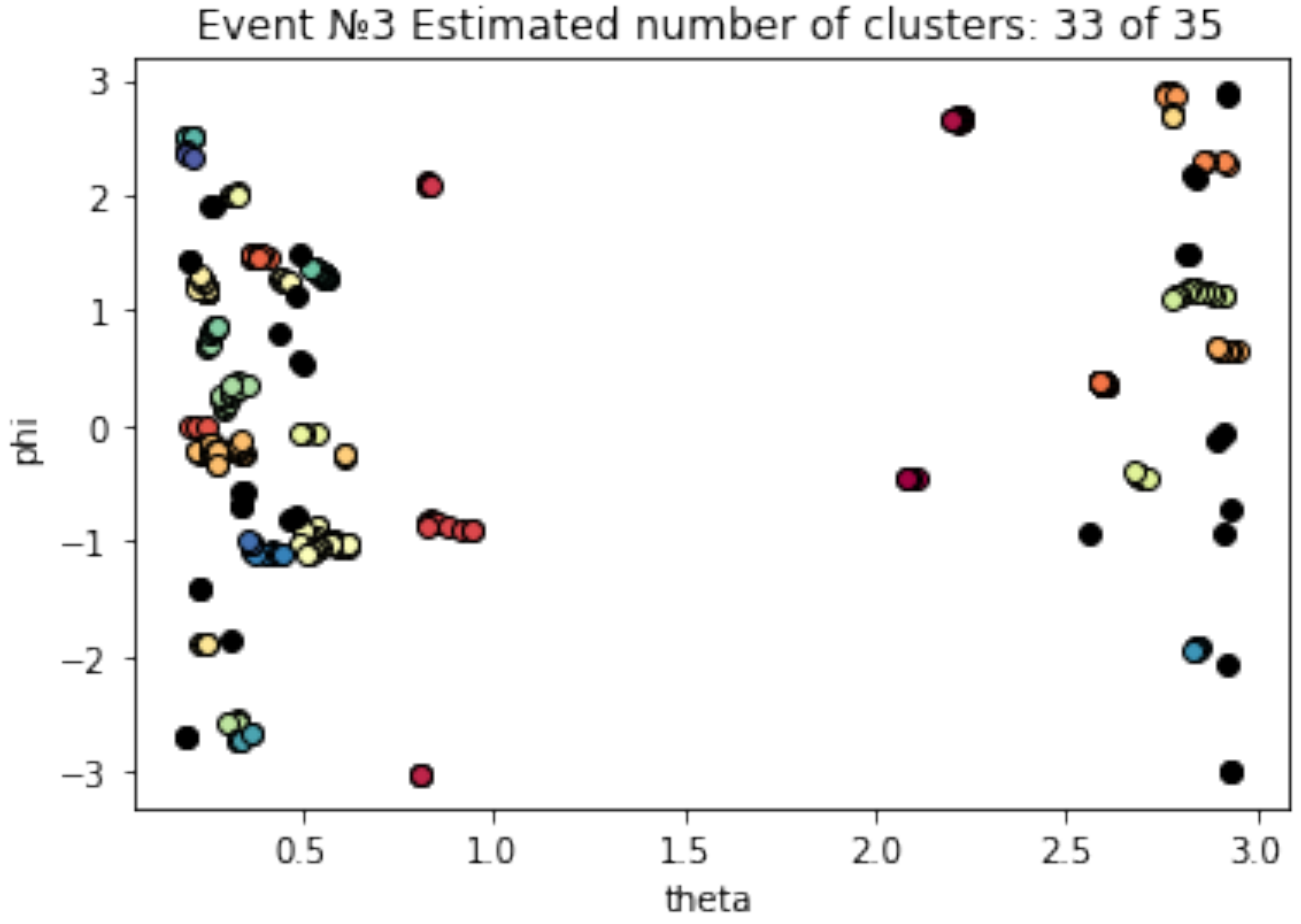


Event №970 Estimated number of clusters: 38 of 40

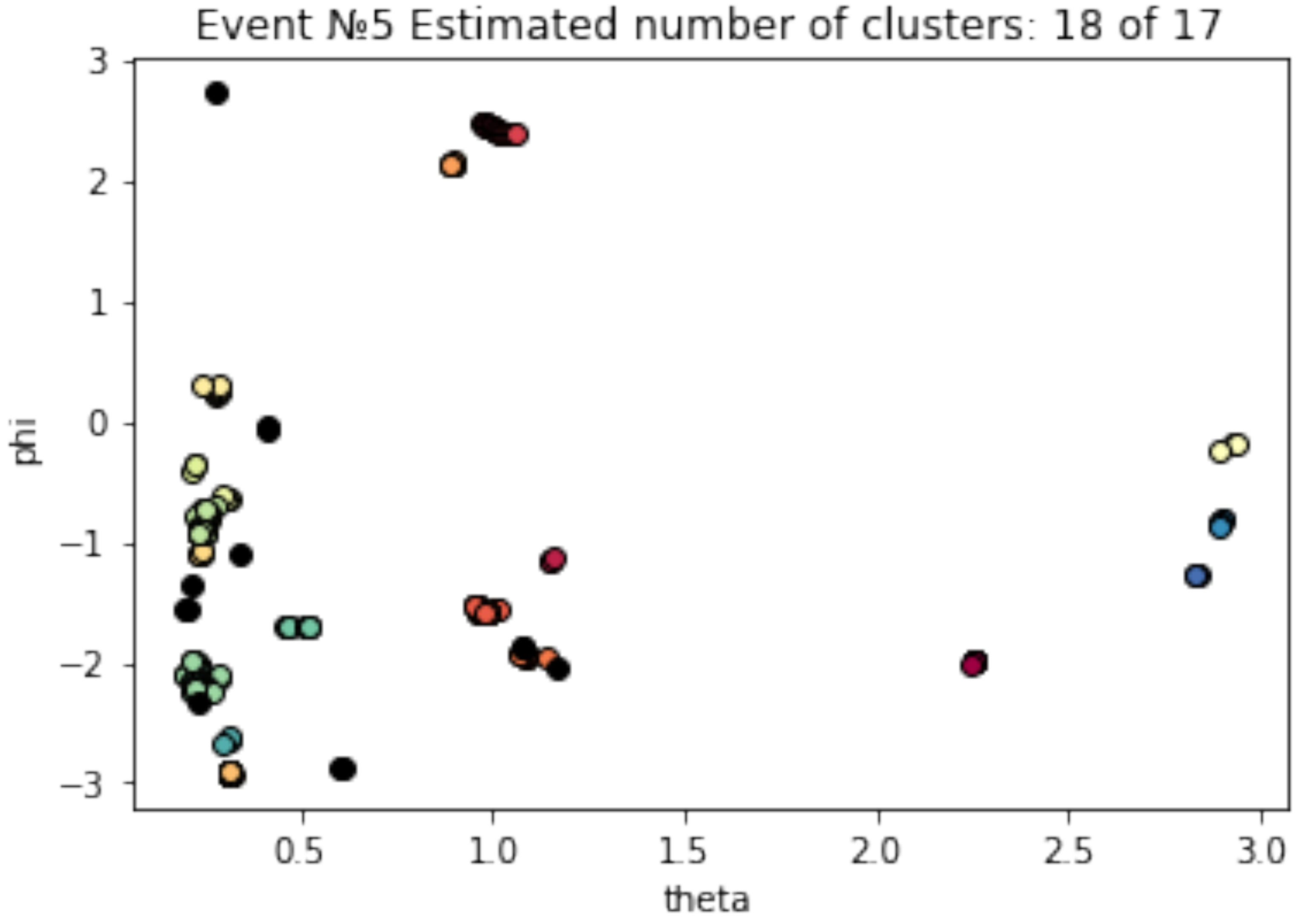


Event №25 Estimated number of clusters: 28 of 30

Muon and pion clusters look more similar, especially in Barrel



Event №3 Estimated number of clusters: 33 of 35



Event №5 Estimated number of clusters: 18 of 17

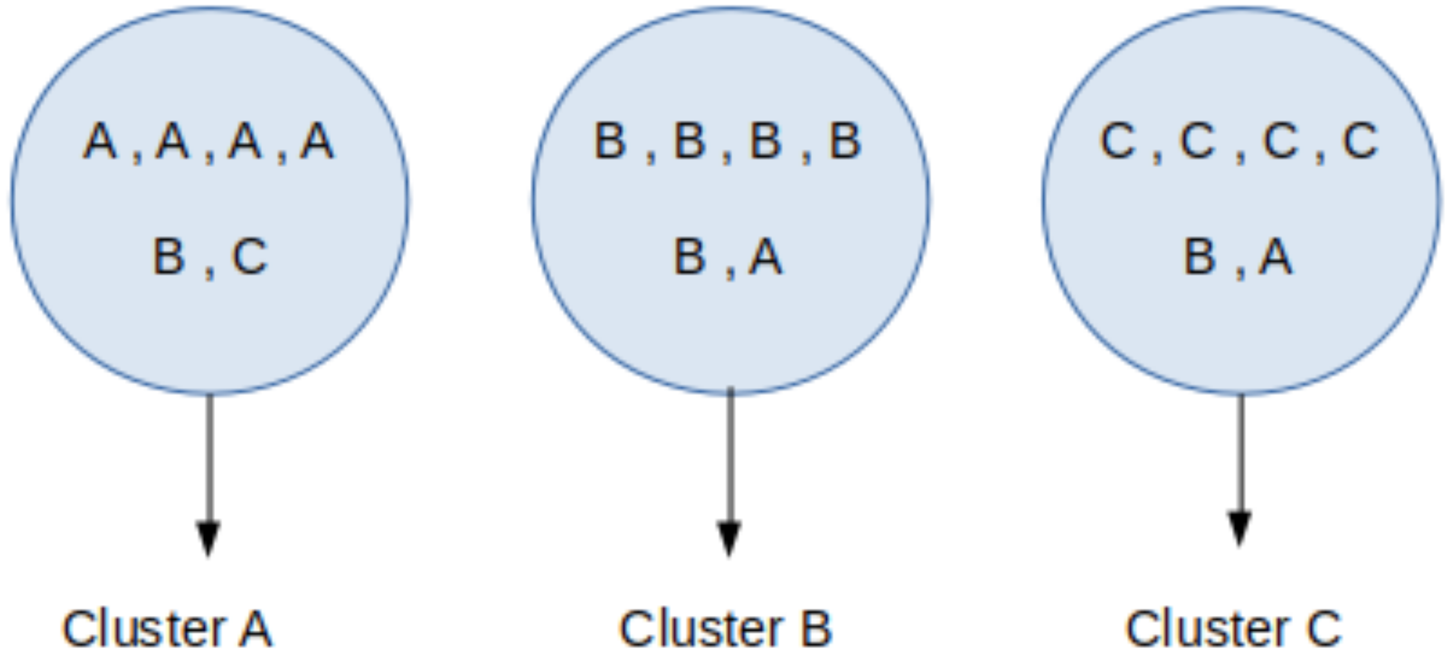
Muons can have some length in the EndCaps

Clustering evaluation metrics

1. Purity:

$$P = \frac{\sum_i N_{i,\ hits}^{correct}}{N_{hits}^{total}} = \frac{ClusterA + ClusterB + ClusterC}{Total}$$

Advantages: straightforward and transparent metrics, easy to calculate;
Downsides: increases as the number of cluster increases.



2. V-measure:

- **homogeneity**: each cluster contains only members of a single class.
- **completeness**: all members of a given class are assigned to the same cluster.

Harmonic mean between the homogeneity and completeness:

$$v = \frac{(1+\beta) * homogeneity * completeness}{(\beta * homogeneity + completeness)},$$

where by default $\beta = 1$.

Advantages: normalized [0, 1]; can be used to compare different clustering models that have different number of clusters;
Downsides: random labelling won't yield zero scores especially when the number of clusters is large

There are a lot of other, more sophisticated metrics.
Will be considered later.

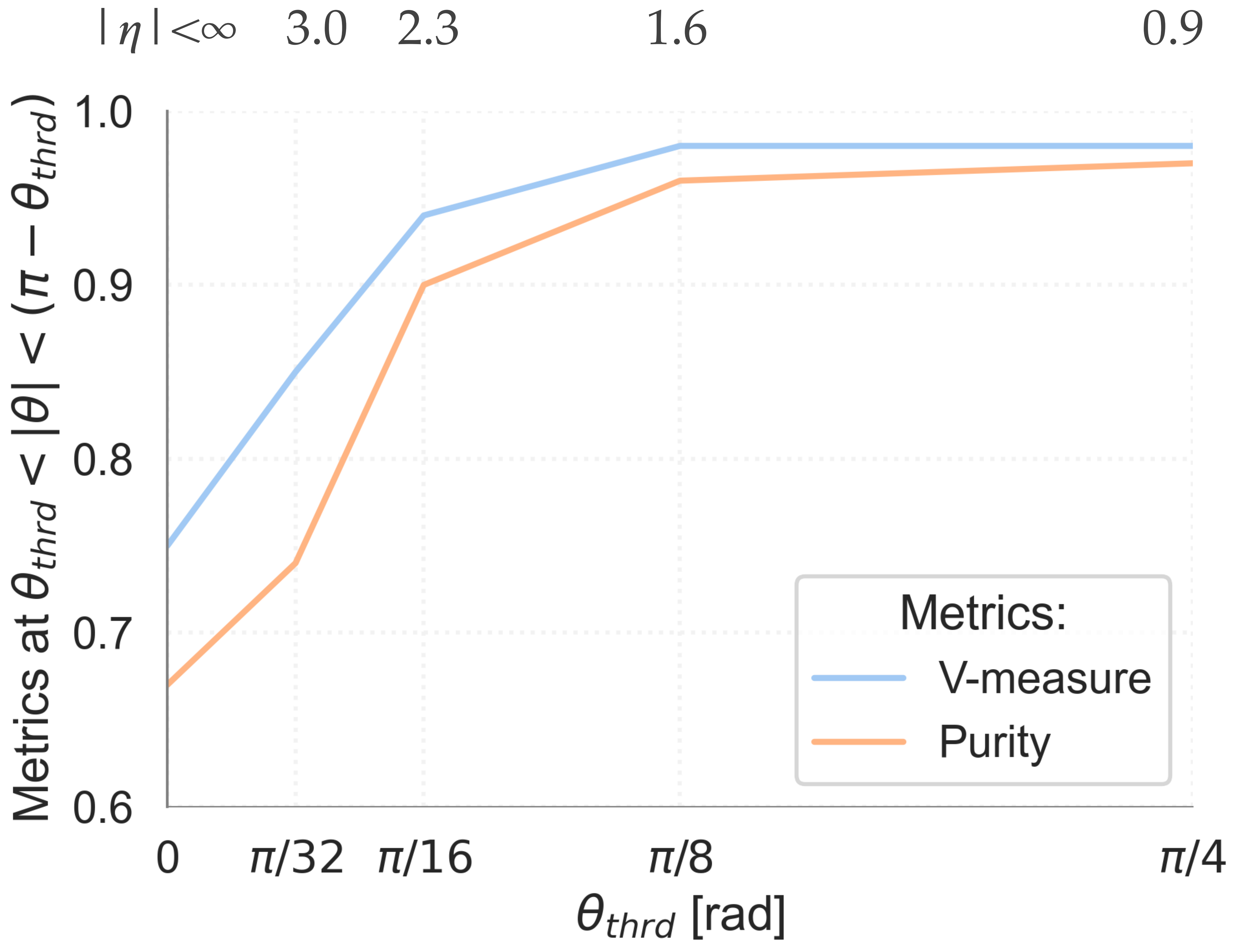
DBSCAN evaluation metrics

Using the ground truth from reference clustering one can evaluate the performance of the algorithm

Purity of the clusters is at the level of 90% for $\theta_{thrd} = \pi/16$ (approximately corresponds to $|\eta| < 2.4$ requirement) and up to 96% for $|\eta| < 1.6$;

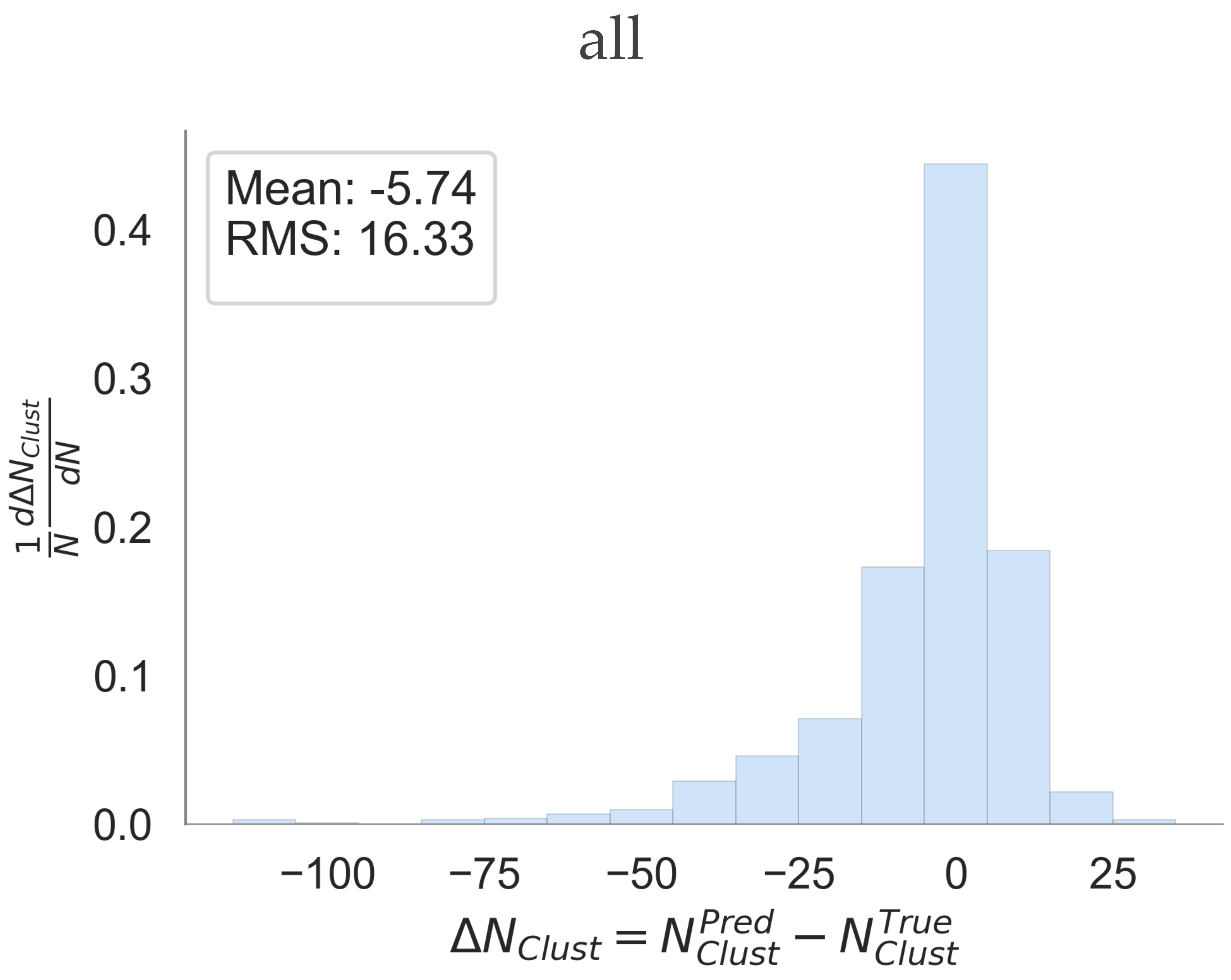
Quite ordinary performance with no threshold requirement due to high hit density in very forward region;

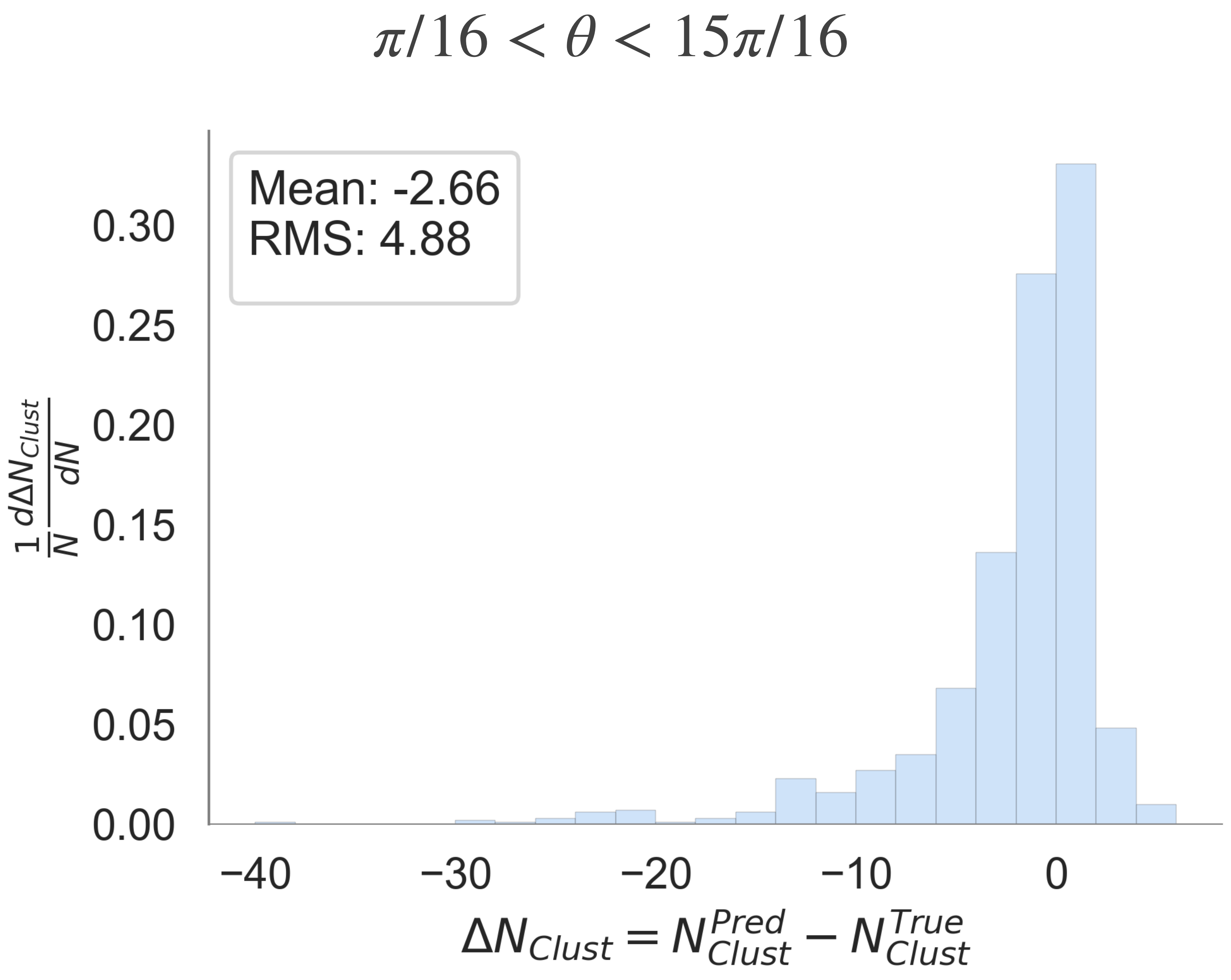
Similar values for V-measure metrics;



DBSCAN evaluation metrics applied in *(x,y,z)* coordinates, as a function of polar angle threshold.

Clustering efficiency





Relative error: $\dfrac{\Delta N_{Clust}}{N_{Clust}} \sim 18\,\%$

Relative error: $\dfrac{\Delta N_{Clust}}{N_{Clust}} \sim 15\,\%$

Clustering algorithms performance

Algorithm	Threshold	Purity	V-measure	N_{clust} relative error (%)
K-Means	—	0.76	0.76	0.41
DBSCAN (θ,φ)	—	0.58	0.71	0.37
DBSCAN (θ,φ)	$\vert\eta\vert < 2.4$	0.89	0.93	0.13
DBSCAN (θ,φ)	$\vert\eta\vert < 1.6$	0.96	0.97	0.08
DBSCAN (x,y,z)	—	0.67	0.75	0.18
DBSCAN (x,y,z)	$\vert\eta\vert < 2.4$	0.94	0.94	0.15
DBSCAN (x,y,z)	$\vert\eta\vert < 1.6$	0.96	0.98	0.09

Conclusion

First look made at the centroid- and density-based clustering algorithms application to the SPD RS:

- k-Means clustering algorithm was tested and showed relatively poor performance for RS hit clustering.
- Density-based algorithm (DBSCAN) is found to be more promising to that end;
- One shows relatively good performance in (η, ϕ) and 3-dimensional (x,y,z) coordinate frames;
- Additional hyper-parameter optimization and fine tuning is needed to improve performance.

The DBSCAN-based algorithm might be useful for Online Filter tasks;

As a product we need to organize a pipeline {clustering + π/μ separation algos} and evaluate combined metrics focused on muons.

DBSCAN implementation in C++ is ~200 lines of code; no complex external libraries dependences are needed; can be quite easily incorporated within SpdRoot.

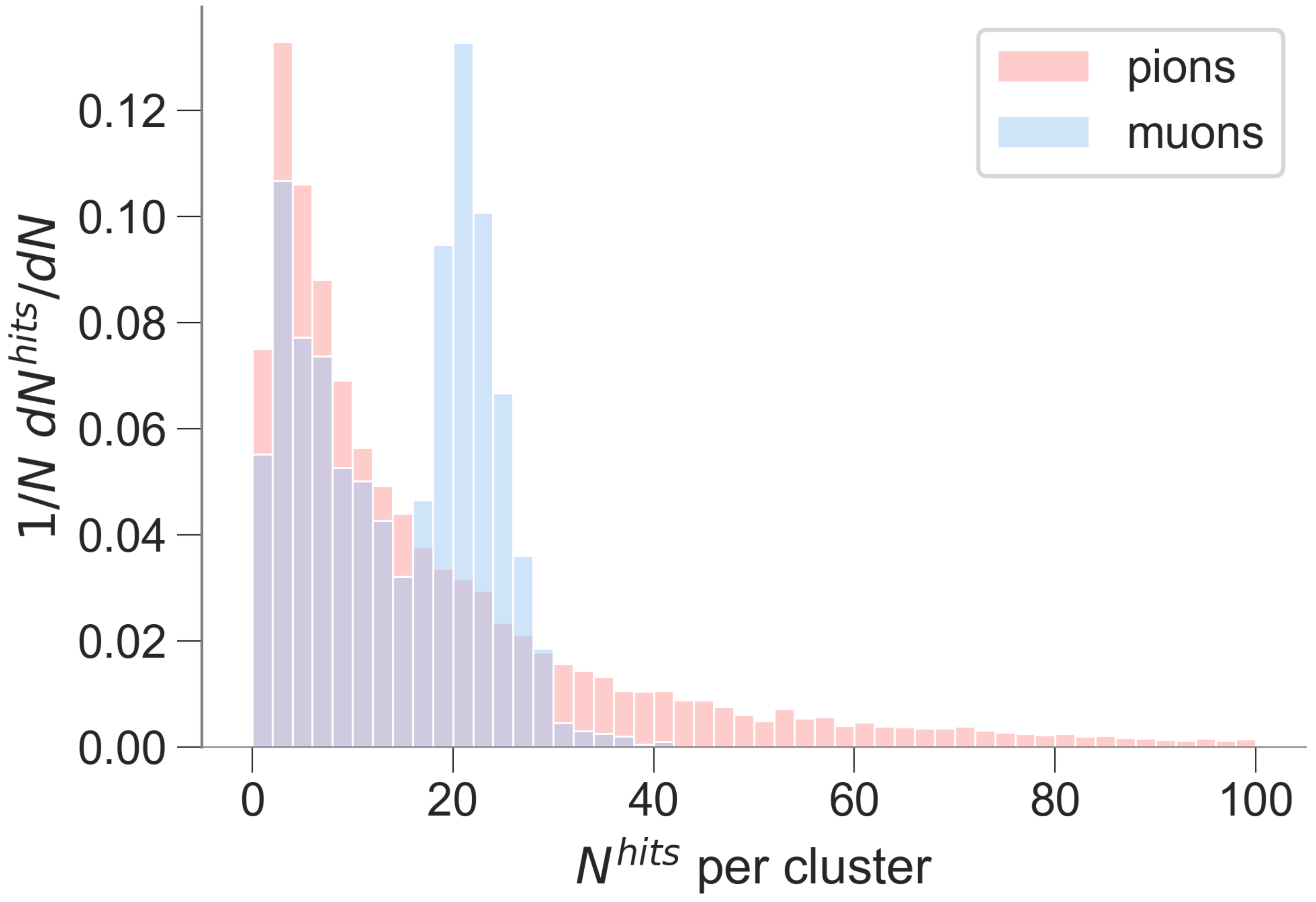
There is a faster version DBSCAN++ (additional research is needed);

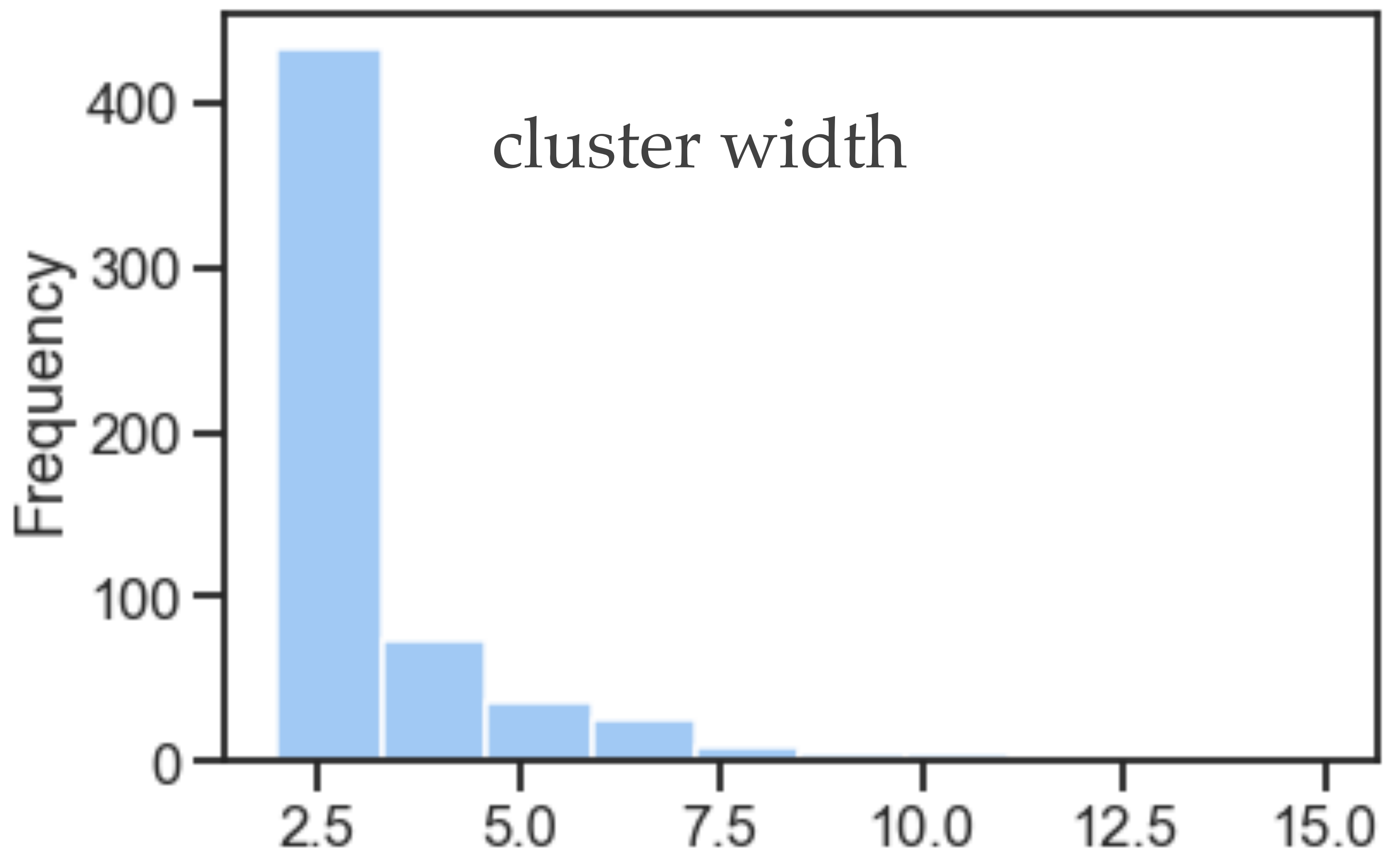
Viktoriya Zel (Balandina) from Muon Group has joined this task. This might be her Master thesis topic at Dubna Univ.

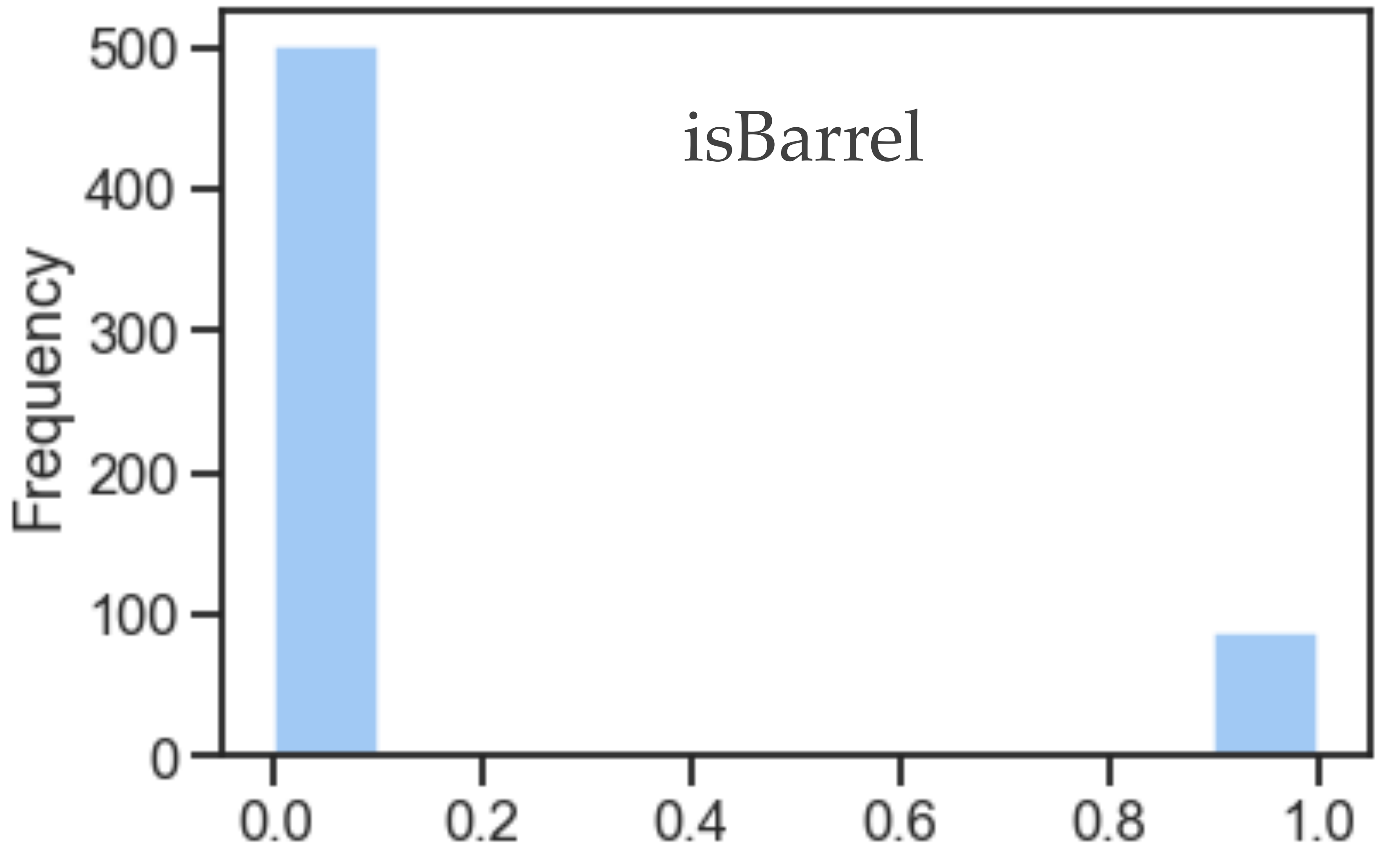
The OPTICS algorithm - a generalization of the DBSCAN that relaxes *epsilon* from a single value to a range - can be another option for improvement; HDBSCAN and hierarchal clustering algorithms are other candidates.

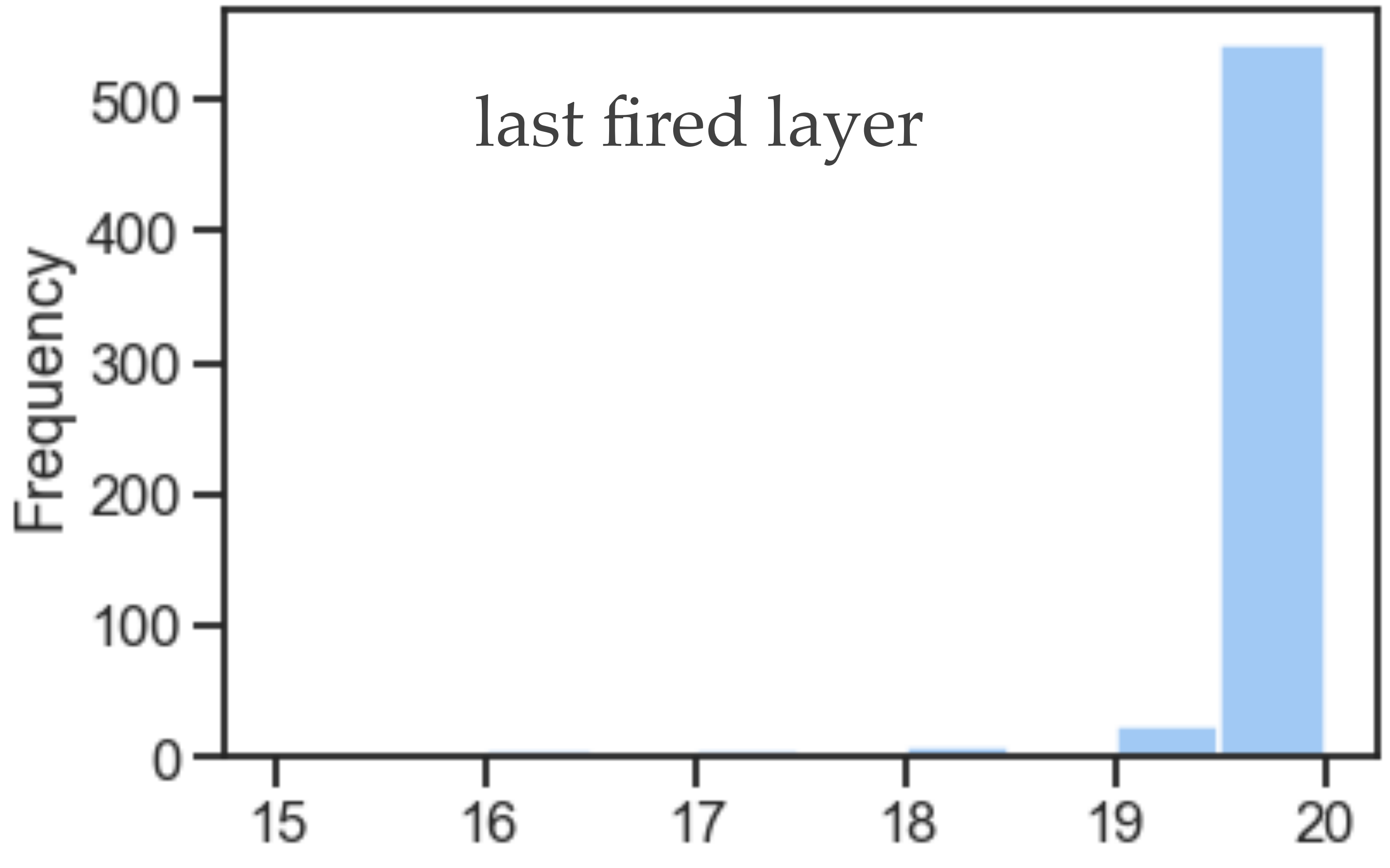
Thank you for attention!

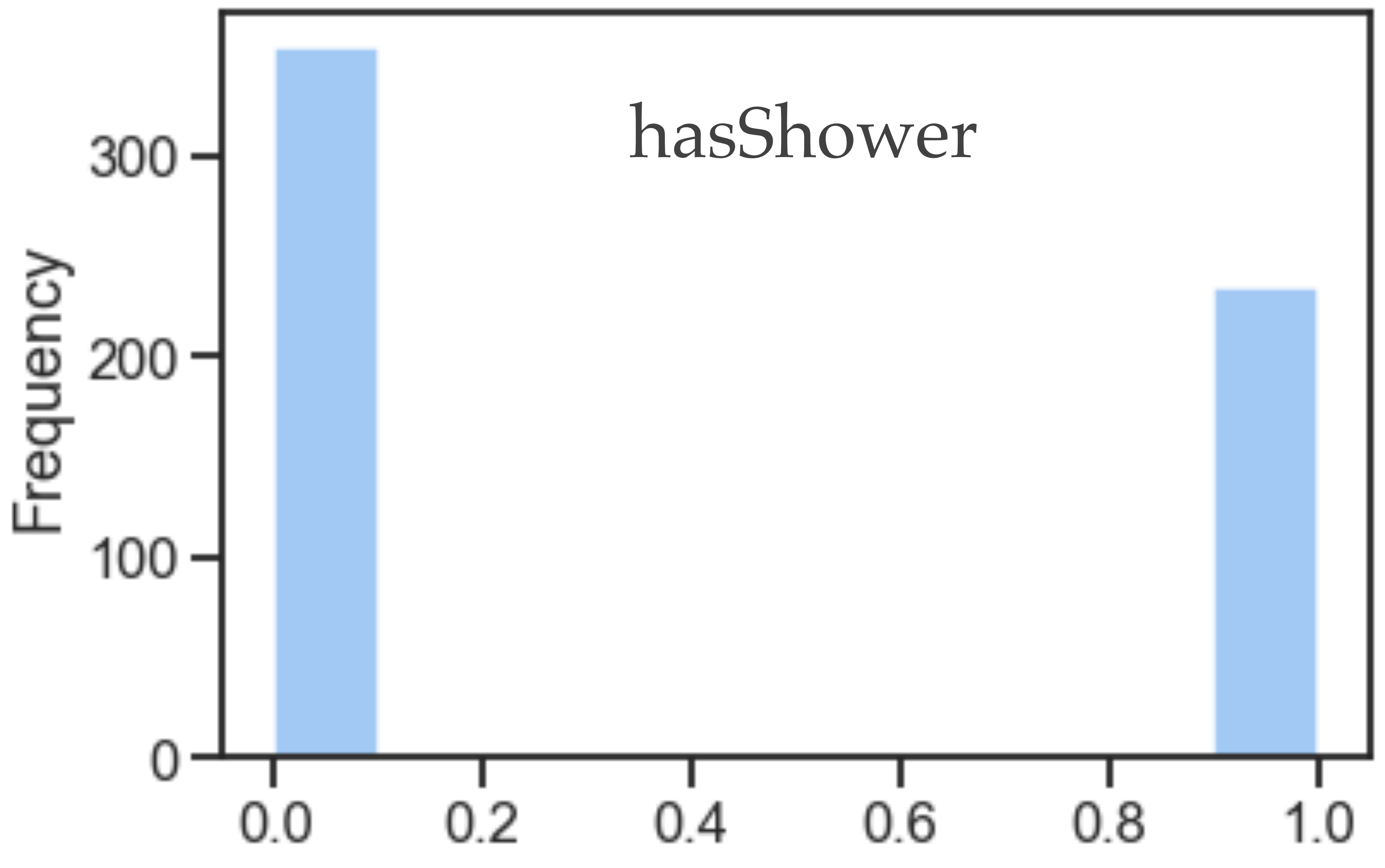
Backup


pions
muons
$1/N\ dN^{hits}/dN$
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0
20
40
60
80
100
N^{hits} per cluster


cluster width
Frequency
0
100
200
300
400
2.5
5.0
7.5
10.0
12.5
15.0


isBarrel
Frequency
0
100
200
300
400
500
0.0
0.2
0.4
0.6
0.8
1.0

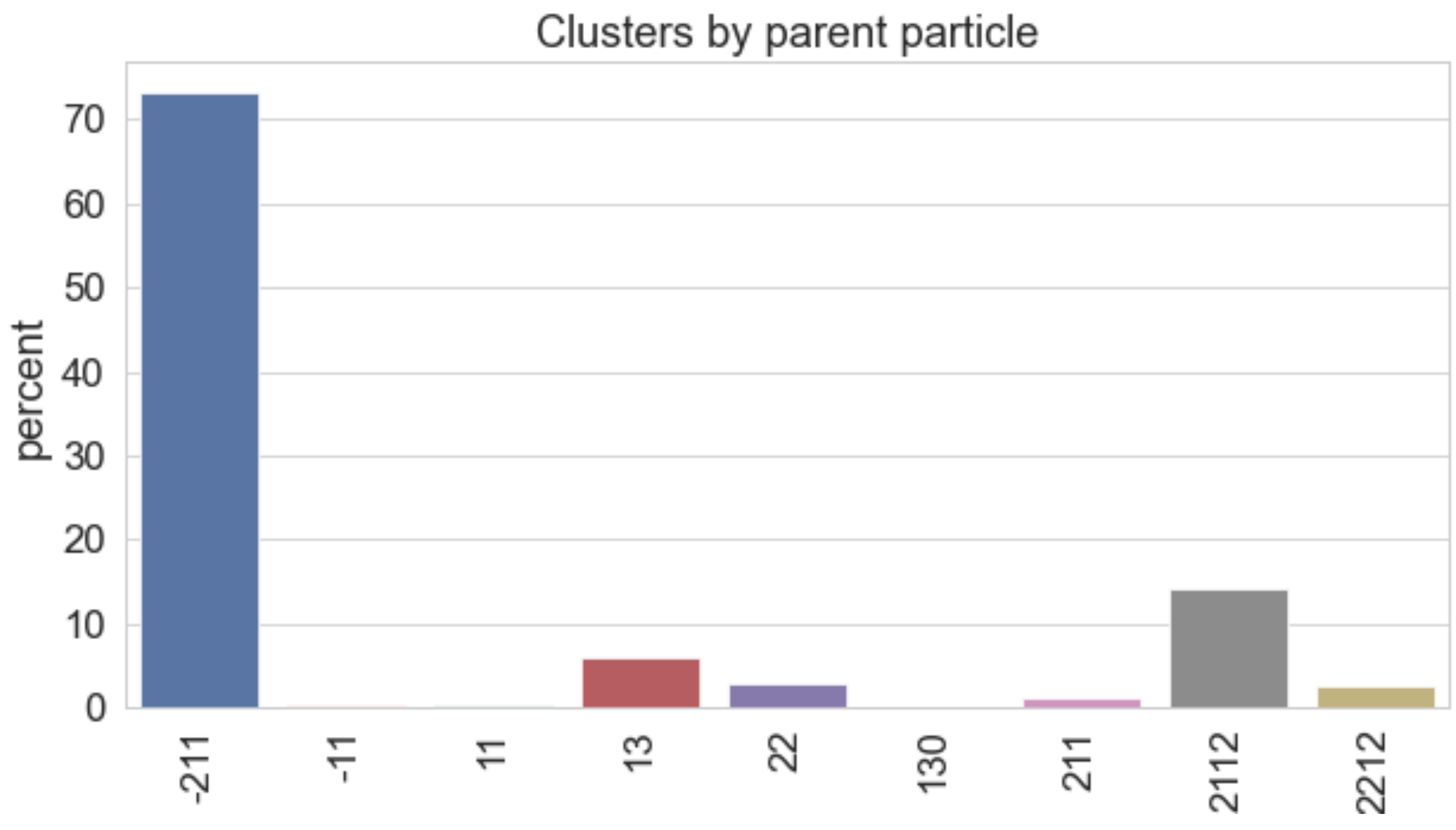

last fired layer
Frequency
0
100
200
300
400
500
15
16
17
18
19
20


hasShower
Frequency
0
100
200
300
0.0
0.2
0.4
0.6
0.8
1.0

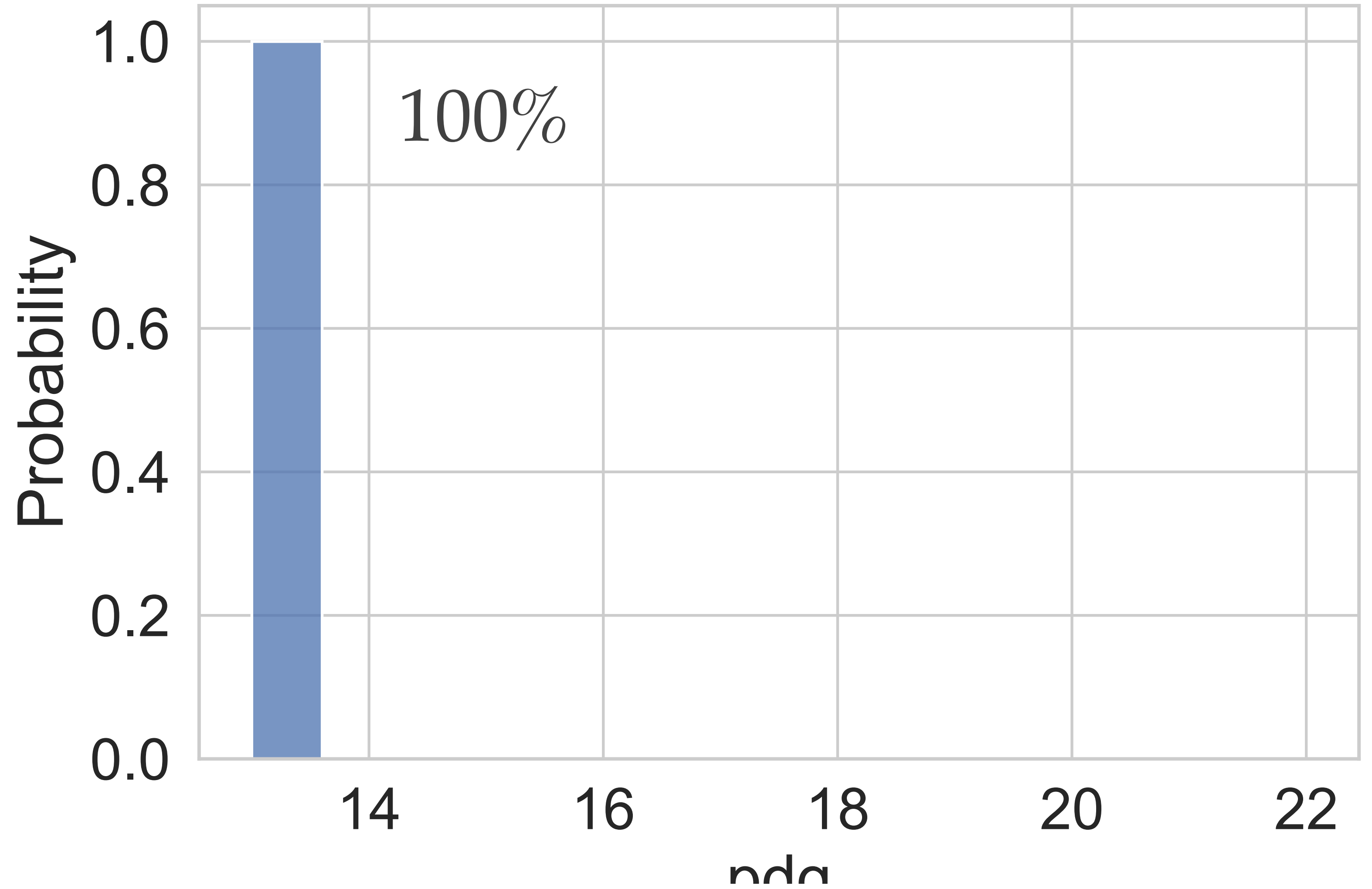
Backup

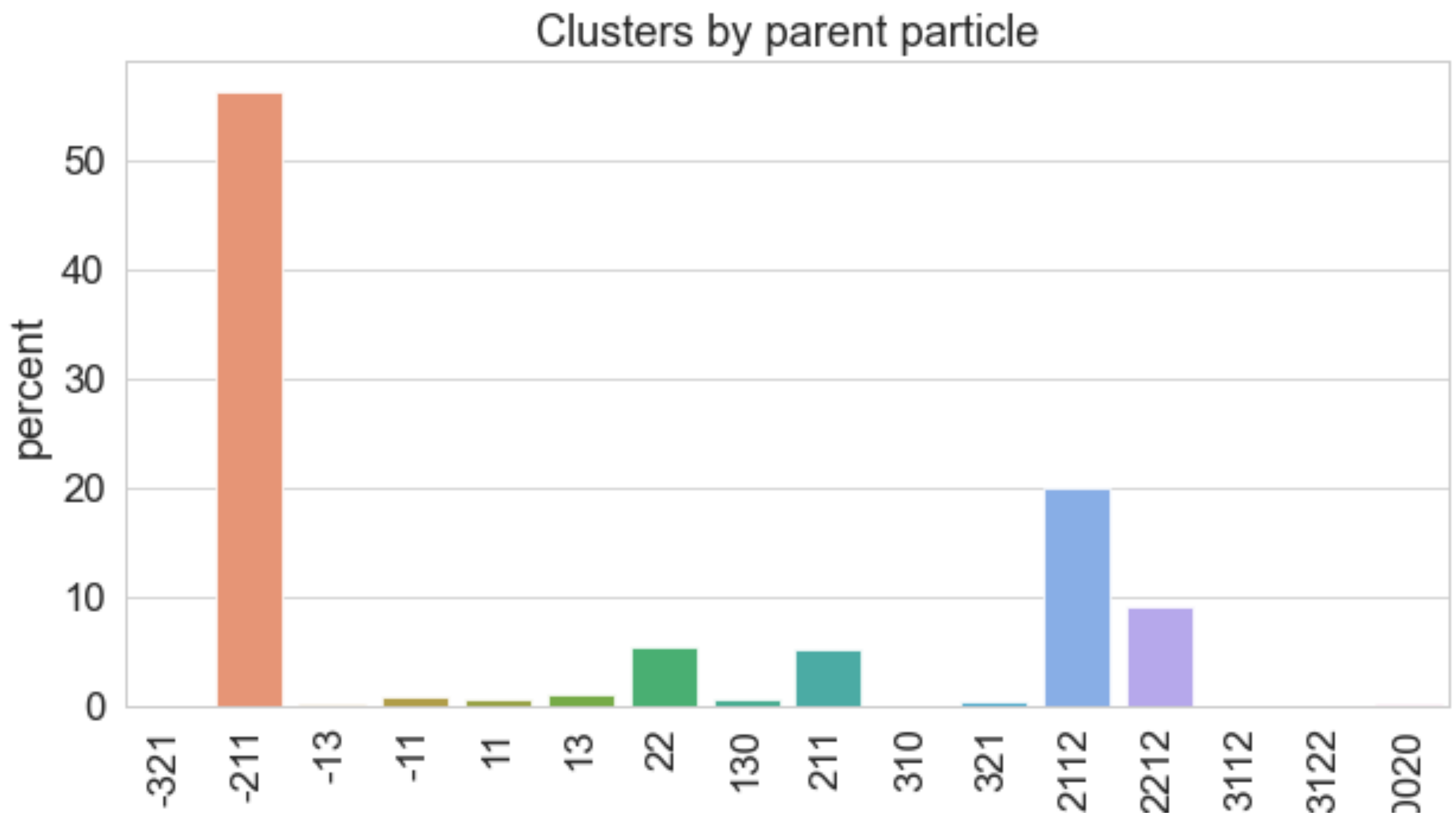
Single Pions

Single Muons

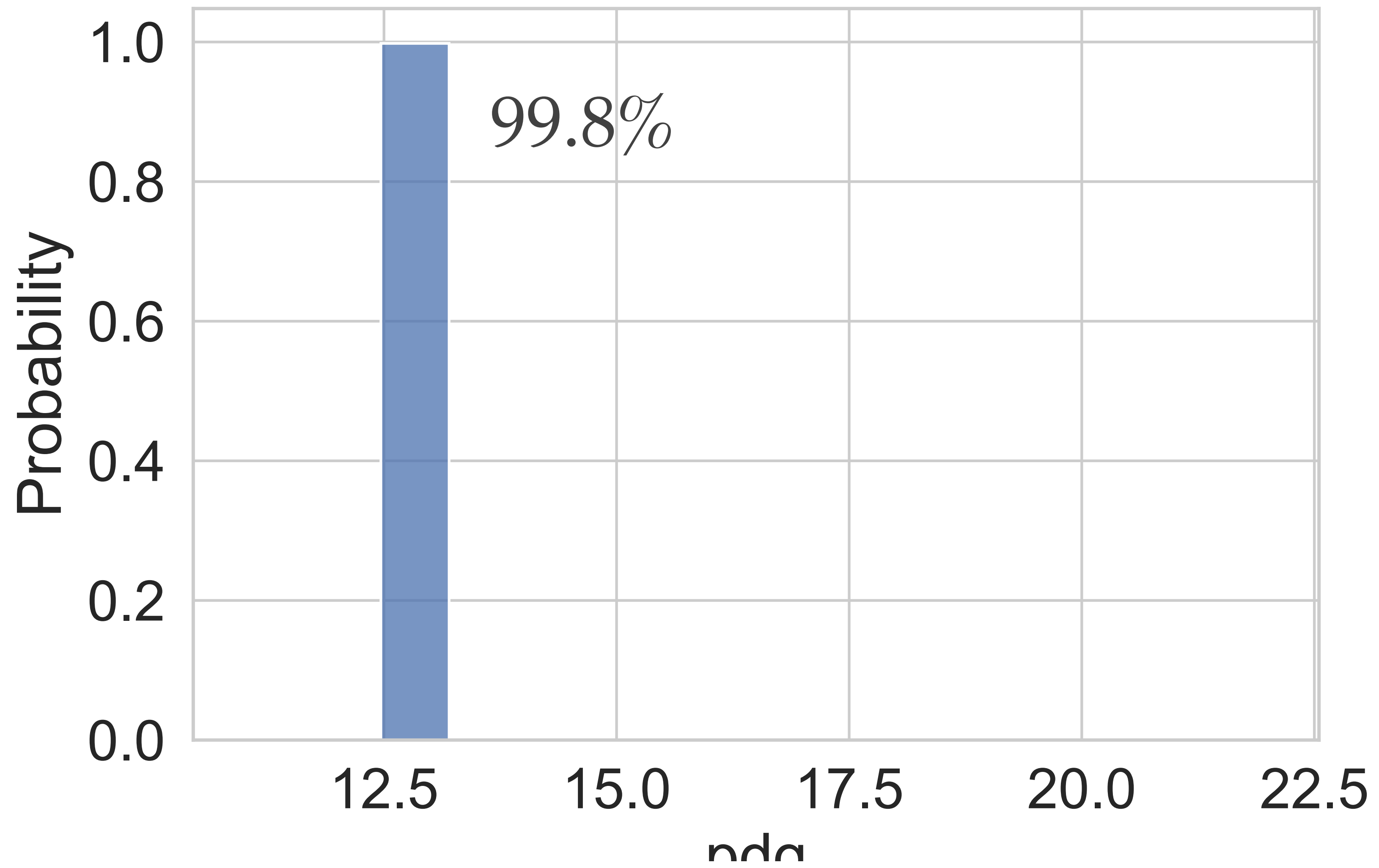


1 GeV





3 GeV


Probability
pdg