9th Patatrack Hackathon

TUE Edition

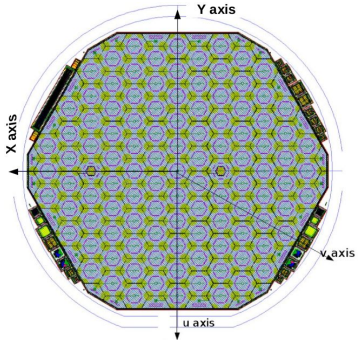
The datasets and score functions

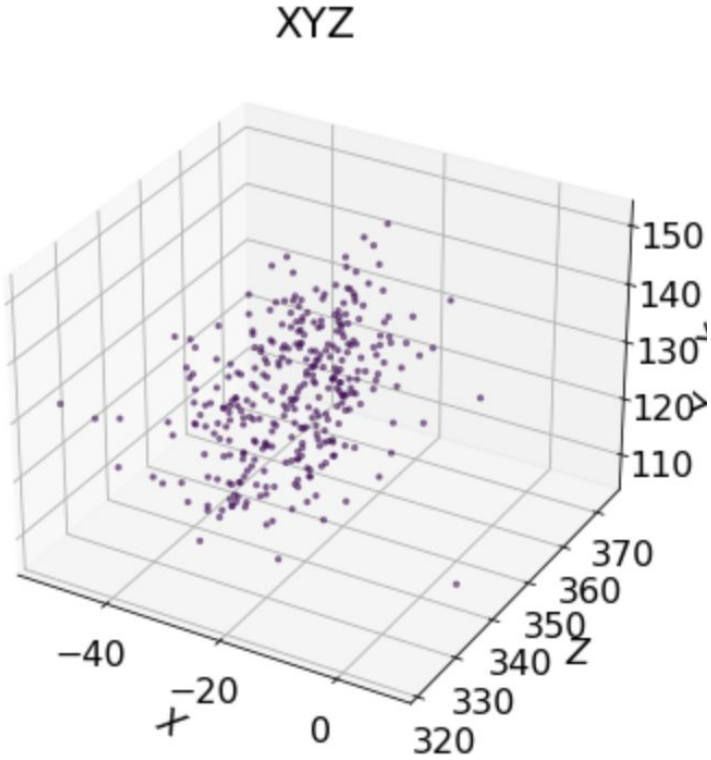
Datasets

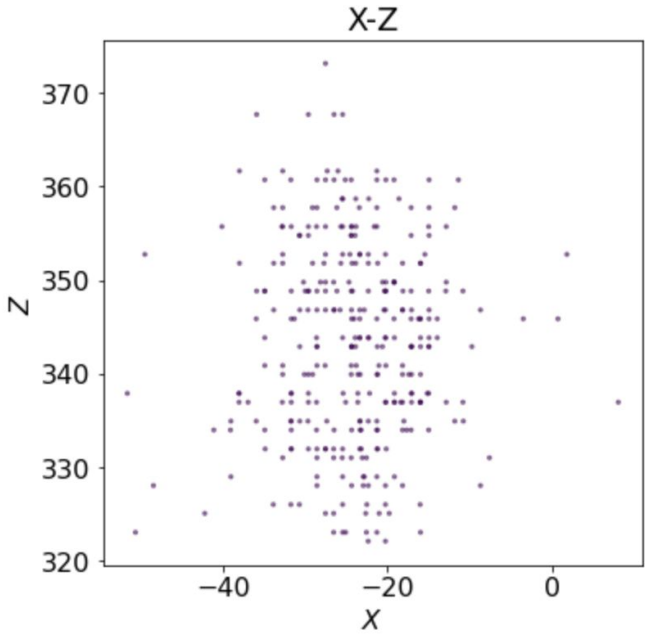
- In the ipython notebook [1], you will find the paths so several different datasets that will help you develop a pattern recognition algorithm
- They correspond to different types of particles being “shot” into HGCal
- Each dataset contains roughly 10,000 “events”
  - An event is a “snapshot” of the detector corresponding to one read-out cycle
  - You don’t have to read all events into memory at the same time for development. Use the nEvents variable and set it to 50 or so to get a first feeling
- For each event, information is stored about the “points” in that event
  - A point is a detected signal in a certain read-out channel. Sometimes we call a “point” also a “hit”.
- Your task is to group the points into reasonable physics objects with an algorithm that also performs well in a multi-particle environment
- Let’s look at some “event displays” that are also part of the notebook

[1] https://github.com/cms-patatrack/tue-hackathon-2021

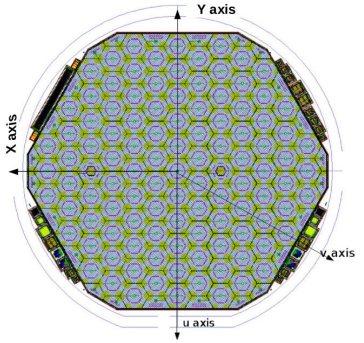
Datasets: shooting single photons

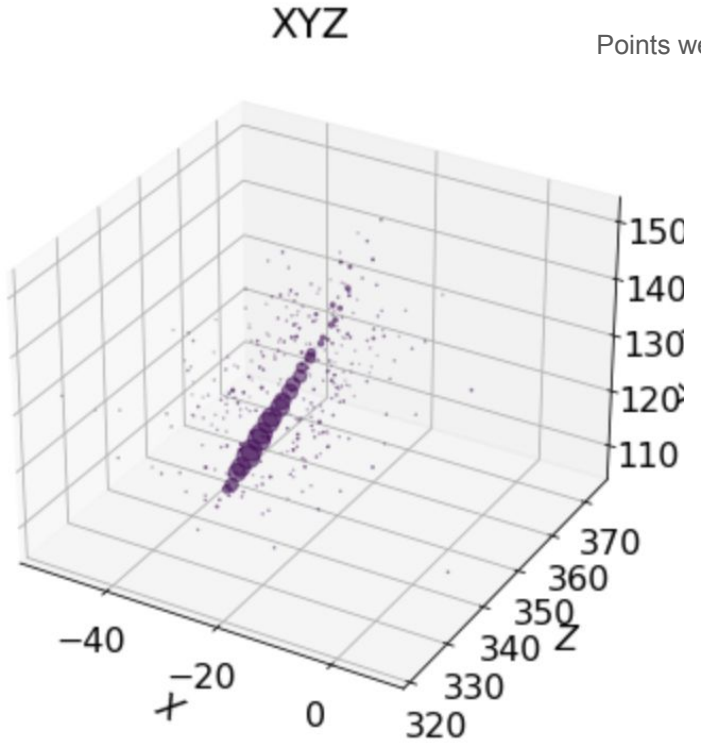




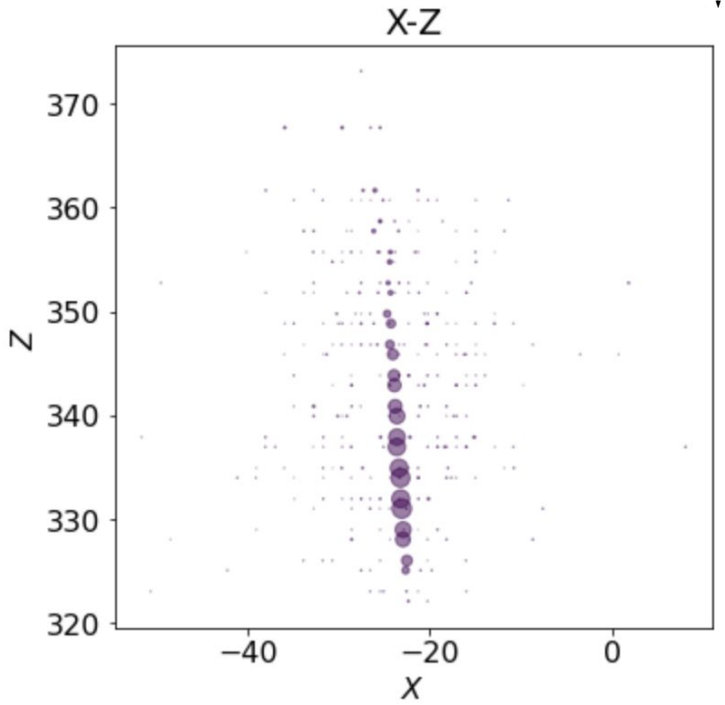


Datasets: shooting single photons

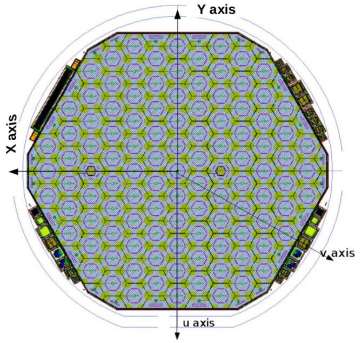


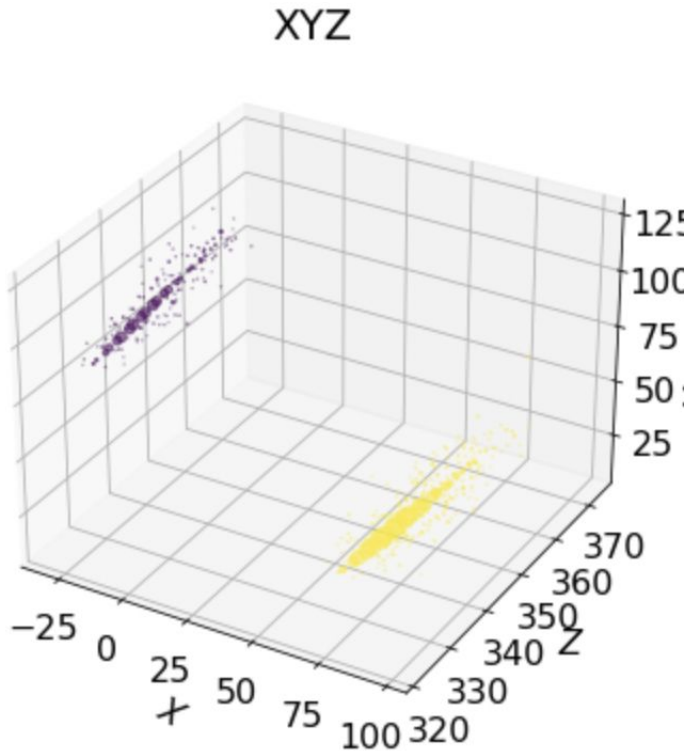


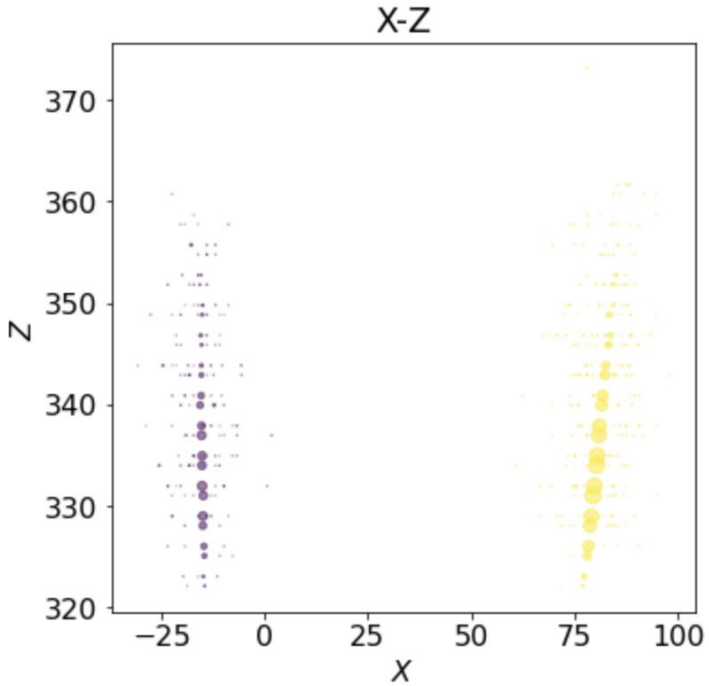
Points weighted by deposited energy



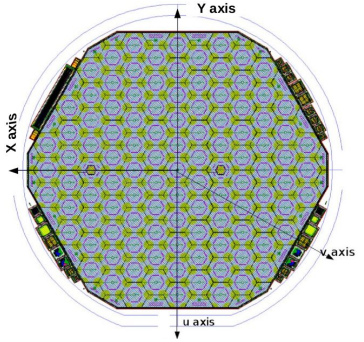
Datasets: shooting two photons

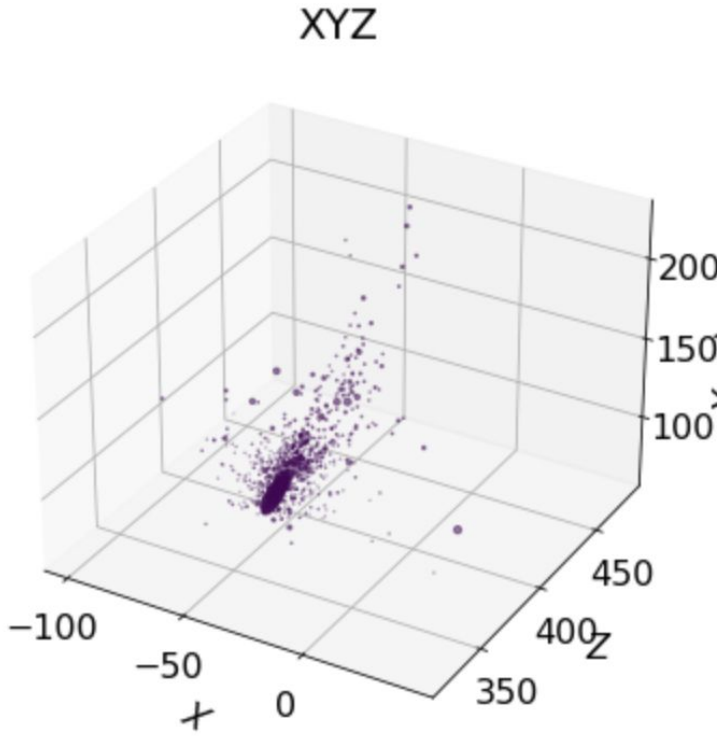


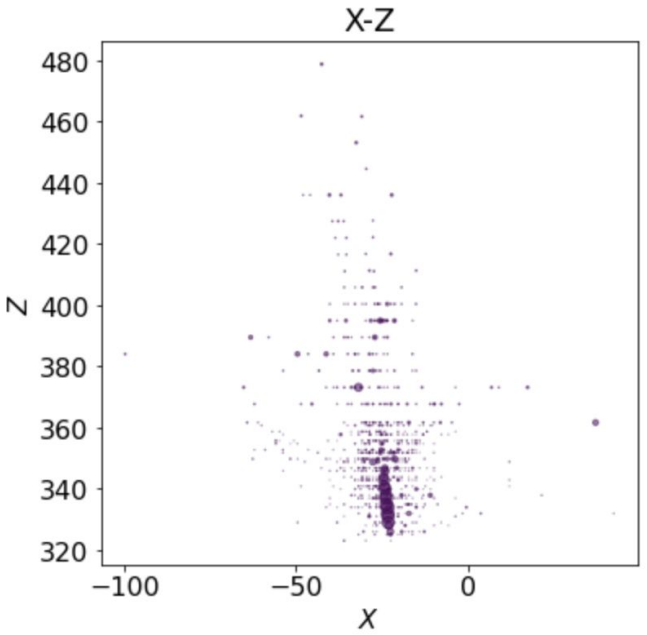




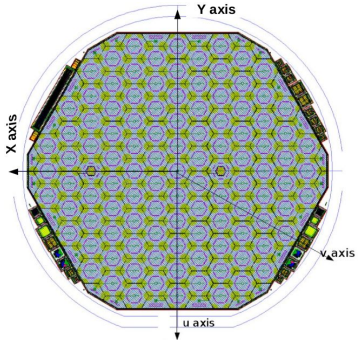
Datasets: shooting single pions

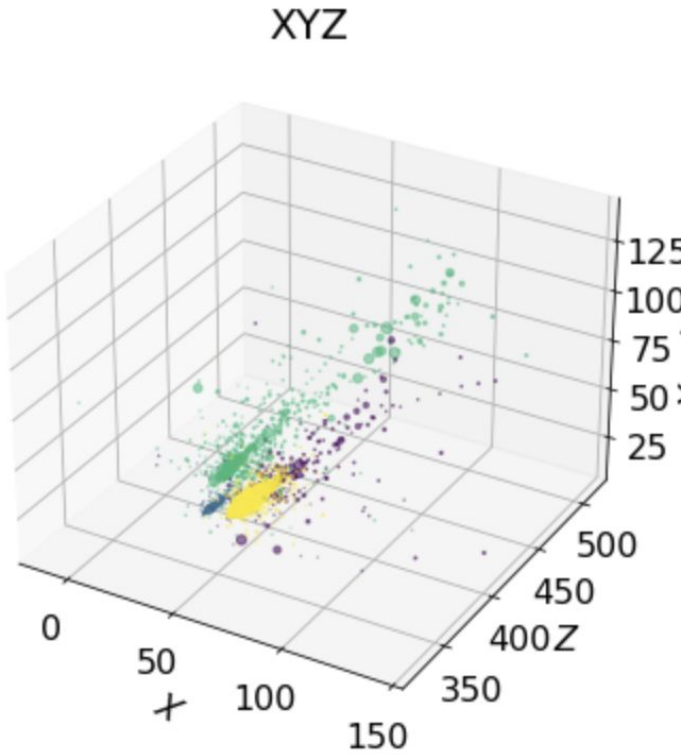


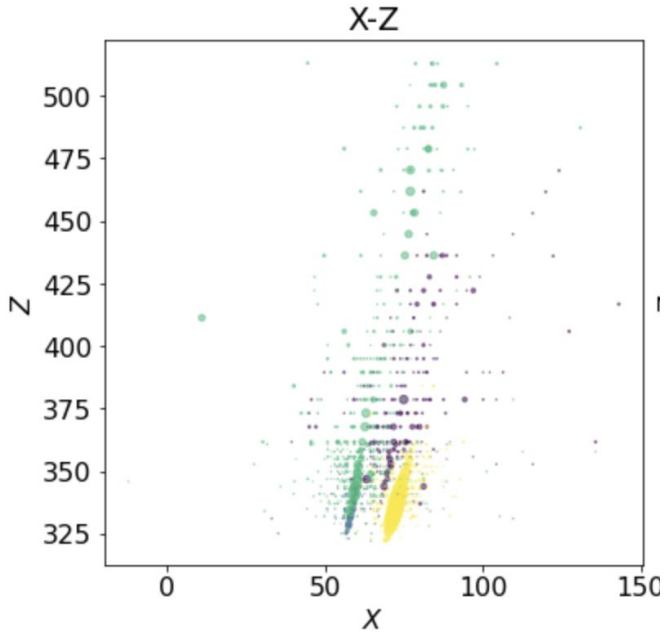




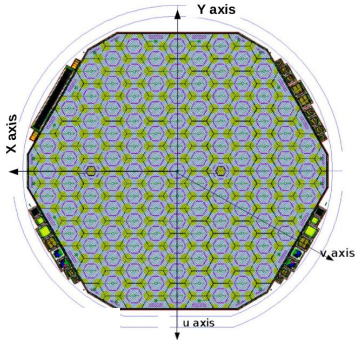
Datasets: shooting a mixture of few particles

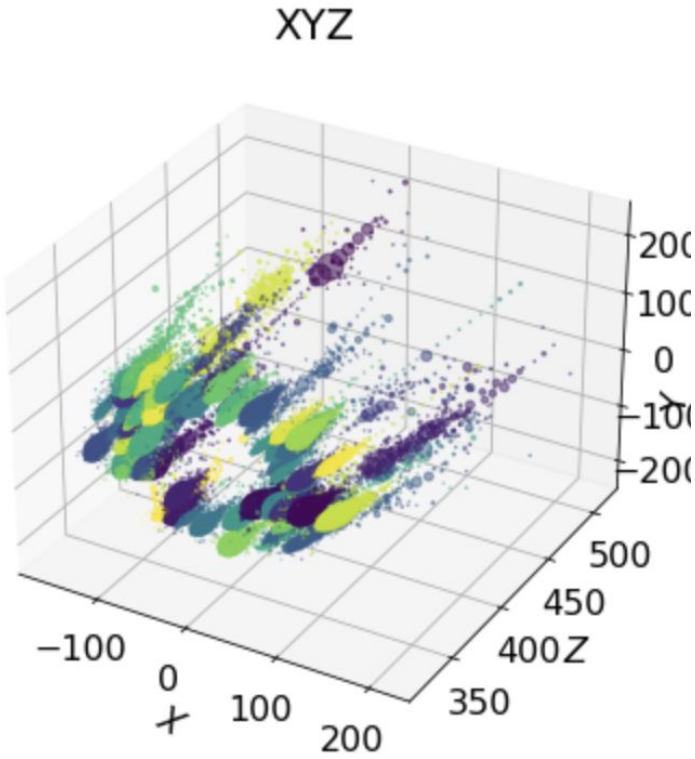


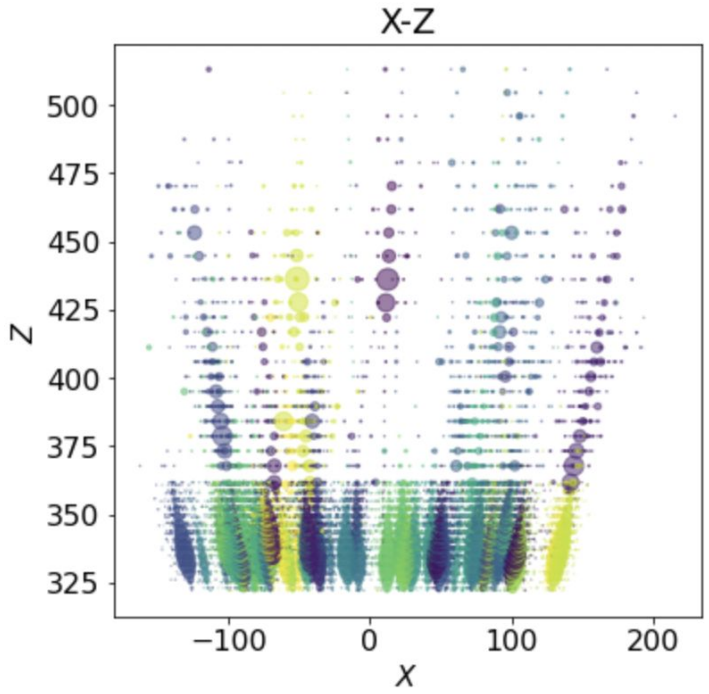




Datasets: shooting a mixture of many particles







Datasets

- You are supposed to assign each point to up to 3 particle showers
- Each event comes with a so-called “truth” definition:

Truth definition

The target is to assign each point to up to 3 particle showers - aka Tracksters - sorted by the higher fraction of the point's energy contained. The truth information is as follows:

- `trueTrackster_idx` : index of the (up to 3) showers associated with the point at hand; if the point is associated to less than 3 showers, the remaining array elements are -1;
- `enFraction` : the fraction of the point's energy that should be assigned to each of the (up to 3) showers.

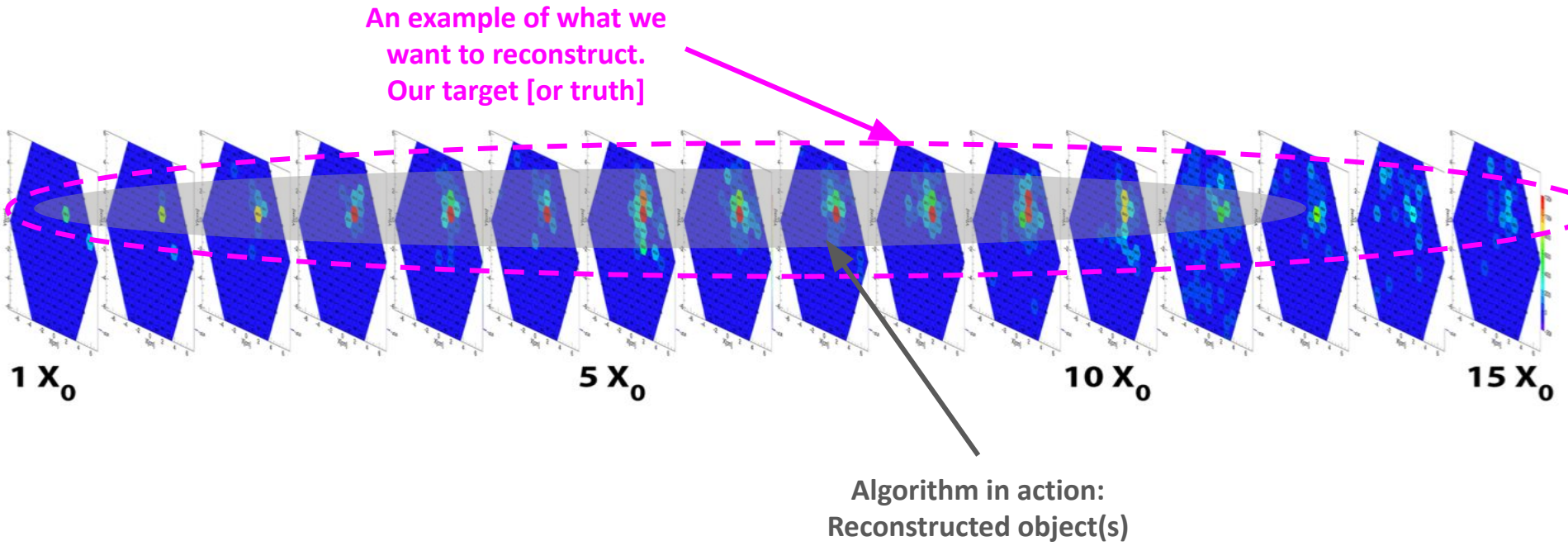
```
In [80]: trueTrackster_idx = events['simTst_idx'].array(entry_stop=nEvents)
         trueTrackster_idx[0]

Out[80]: <Array [[93, -1, -1], [39, ... [19, -1, -1]] type='21516 * var * int64'>
```

- You have test and evaluation datasets - simply two statistically independent sets of events

How good is our algorithm doing ?

- We define three metrics to gauge the performance of the algorithm
  - Effectively by comparing the some “key” reconstructed quantities against “truth” information



How good is our algorithm doing ? (II)

- We define three metrics to gauge the performance of the algorithm
  - Effectively by comparing the some “key” reconstructed quantities against “truth” information

- Metric 1: Reconstruction efficiency [0 -> 100% (best)]
  - # of matched reconstructed objects (RecoTracksters) / # of generated objects (SimTracksters)
  - As “matched” RecoTrackster, those that have > 50% of the energy of the Simtrakcster
- Metric 2: RecoTrackster purity [0 -> 100% (best)]
  - The amount of energy in the RecoTrackster associated to multiple SimTracksters (Shared energy)
    - Ideally all energy coming from a single SimTrackster (100% purity)
- Metric 3: Fake rate or better “over-shooting” rate [1 (best) -> inf]
  - # of RecoTracksters / # of SimTracksters
    - Ideally we want to reconstruct as many RecoTracksters as the SimTracksters