Analysis of Summer20 data: AmBe on LIME

Cavoto, Di Marco, Pinci

CYGNO reco and analysis meeting,
1 April 2021

Recap of reco changes

Implemented the step 0 of directional clustering to gather long tracks

- useful wrt GAC because it handles better the track overlap, which in LIME/500ms exposure is likely

-tuned a lot on the Summer20 data (bkg-only), tuning depends on the running conditions

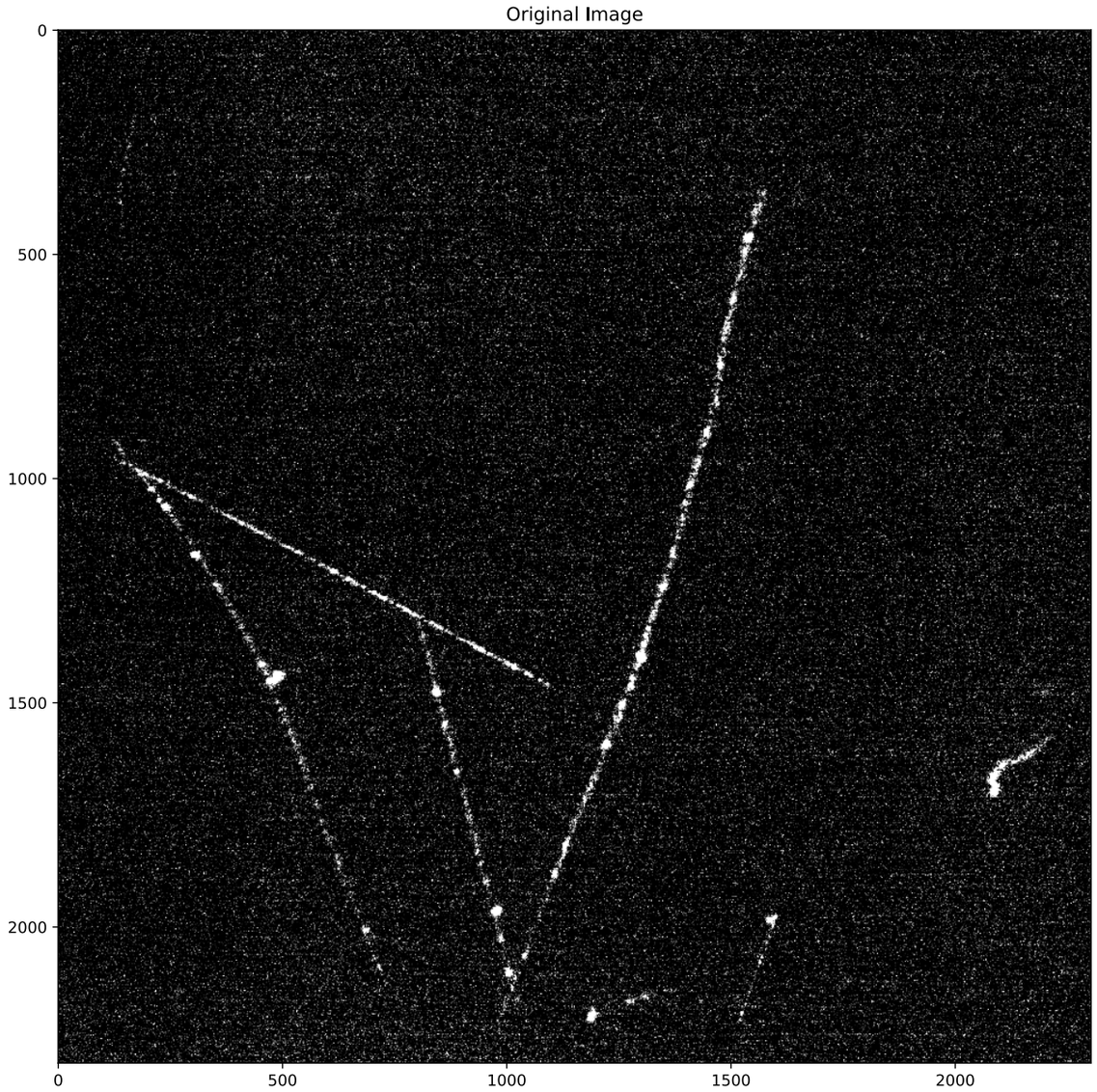
-speed up of factor >10 achieved by moving to 2D for the seeding (because each hit was replicated by N times -N=pixel intensity)

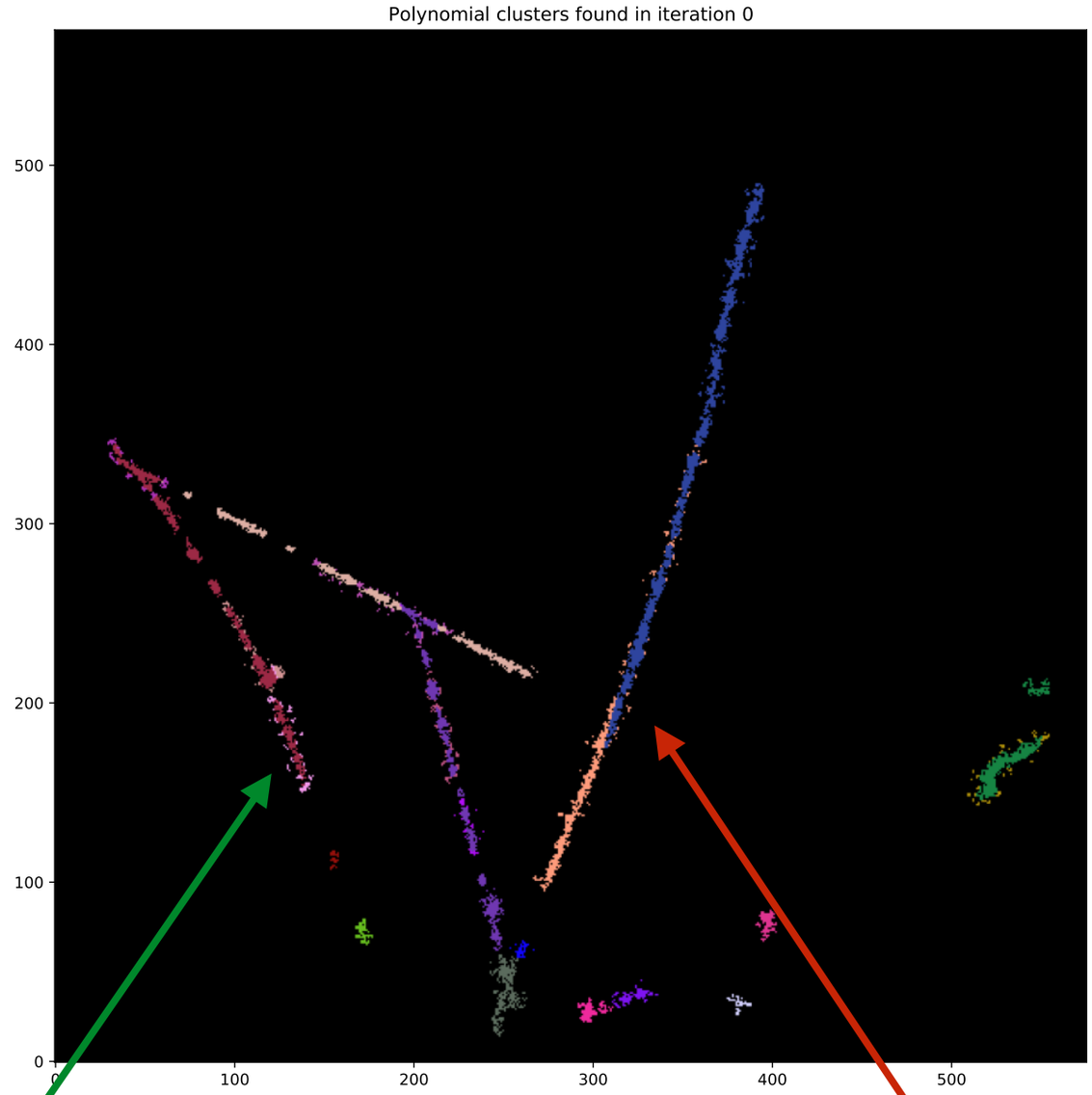
-the remaining pixels after step 0 are clustered only if they are isolated from long (clustered) tracks

- noticed issue: still halos leftover from trakcs, with low-intensity remain. Easily rejected offline (e.g. with density)

Re-reconstructed almost ALL the AmBe Summer20 data with this.

Example 1: AmBe

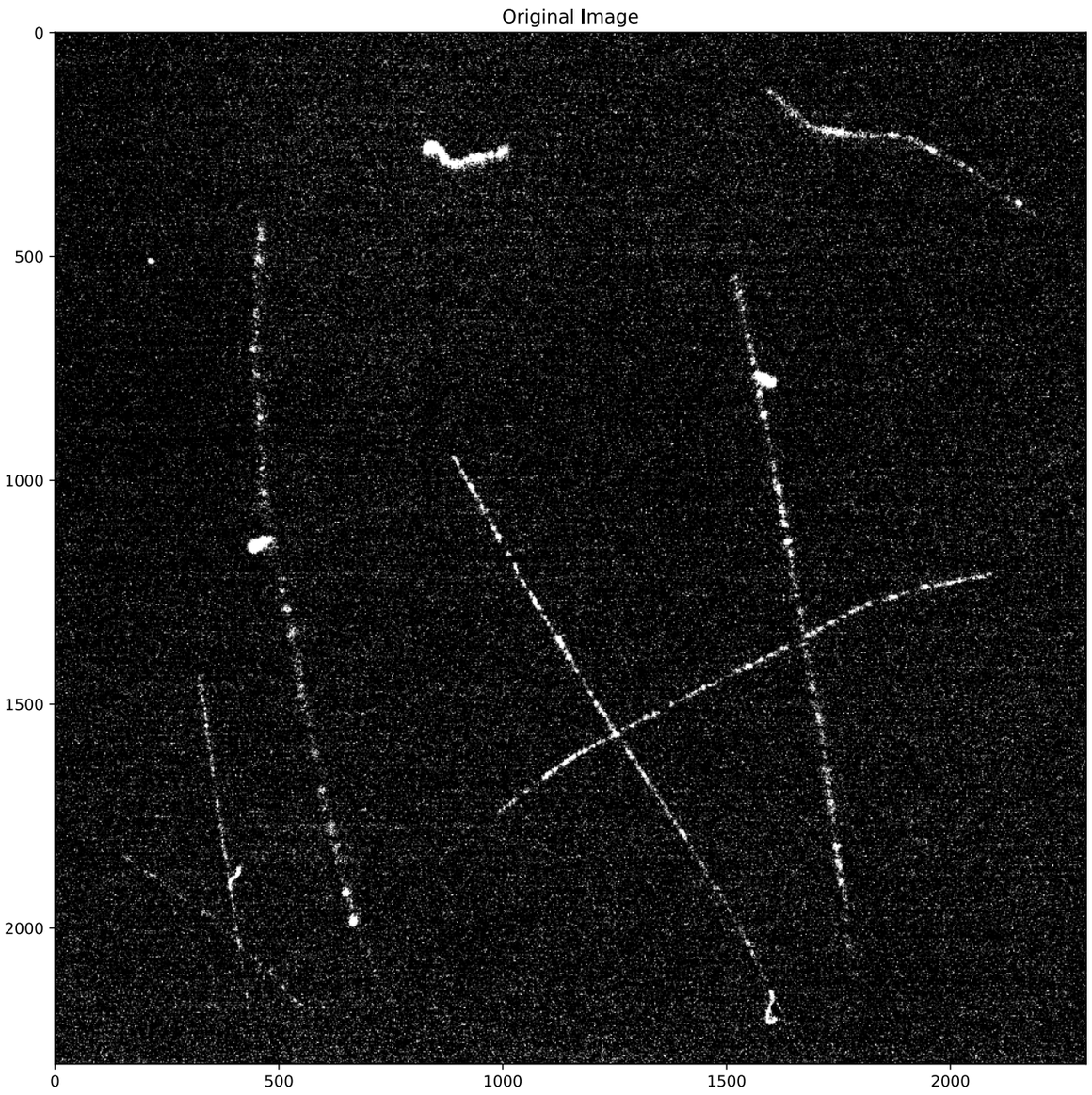


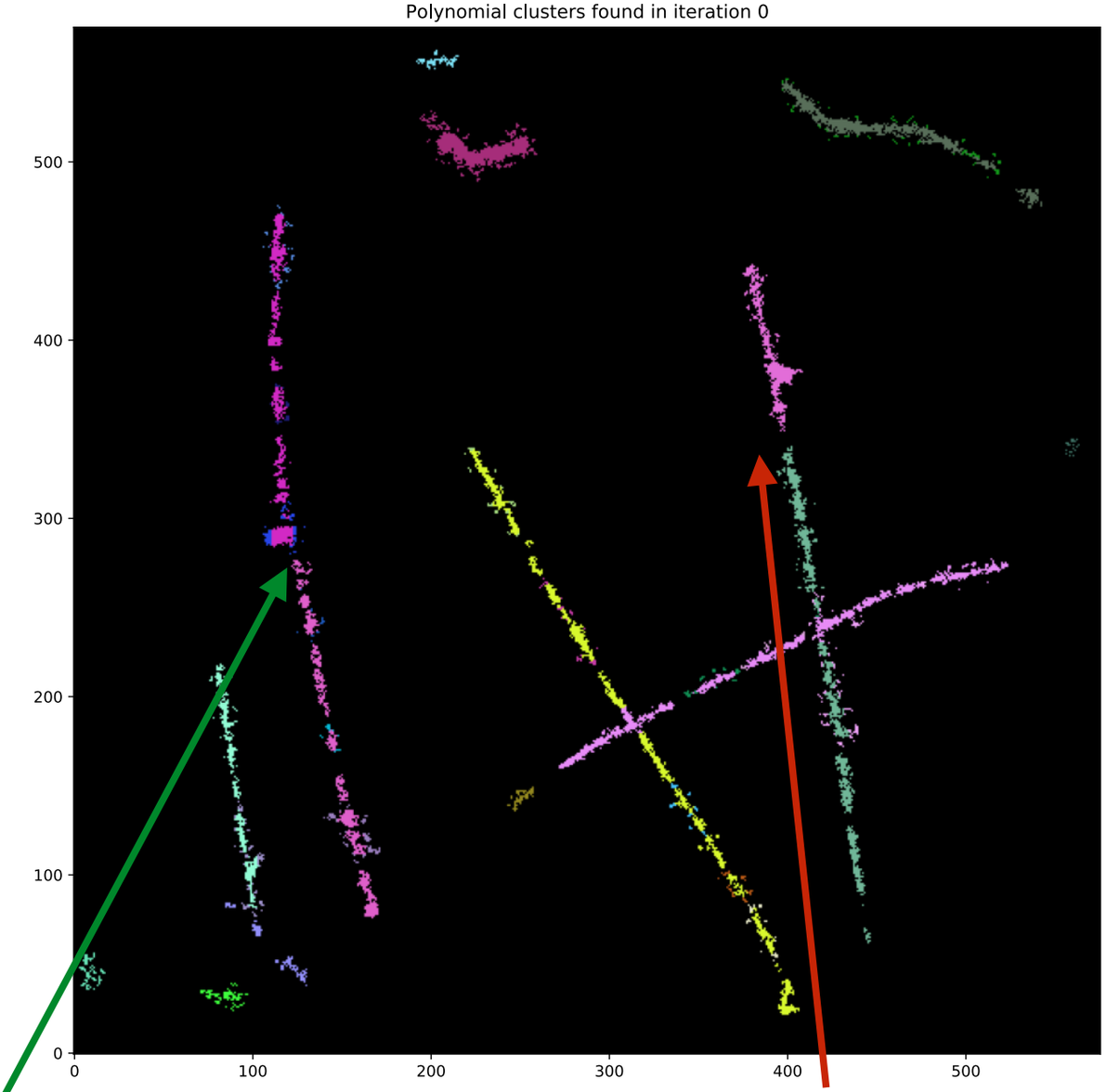


Long track split in two

Halo of low density hits

Example 2: AmBe

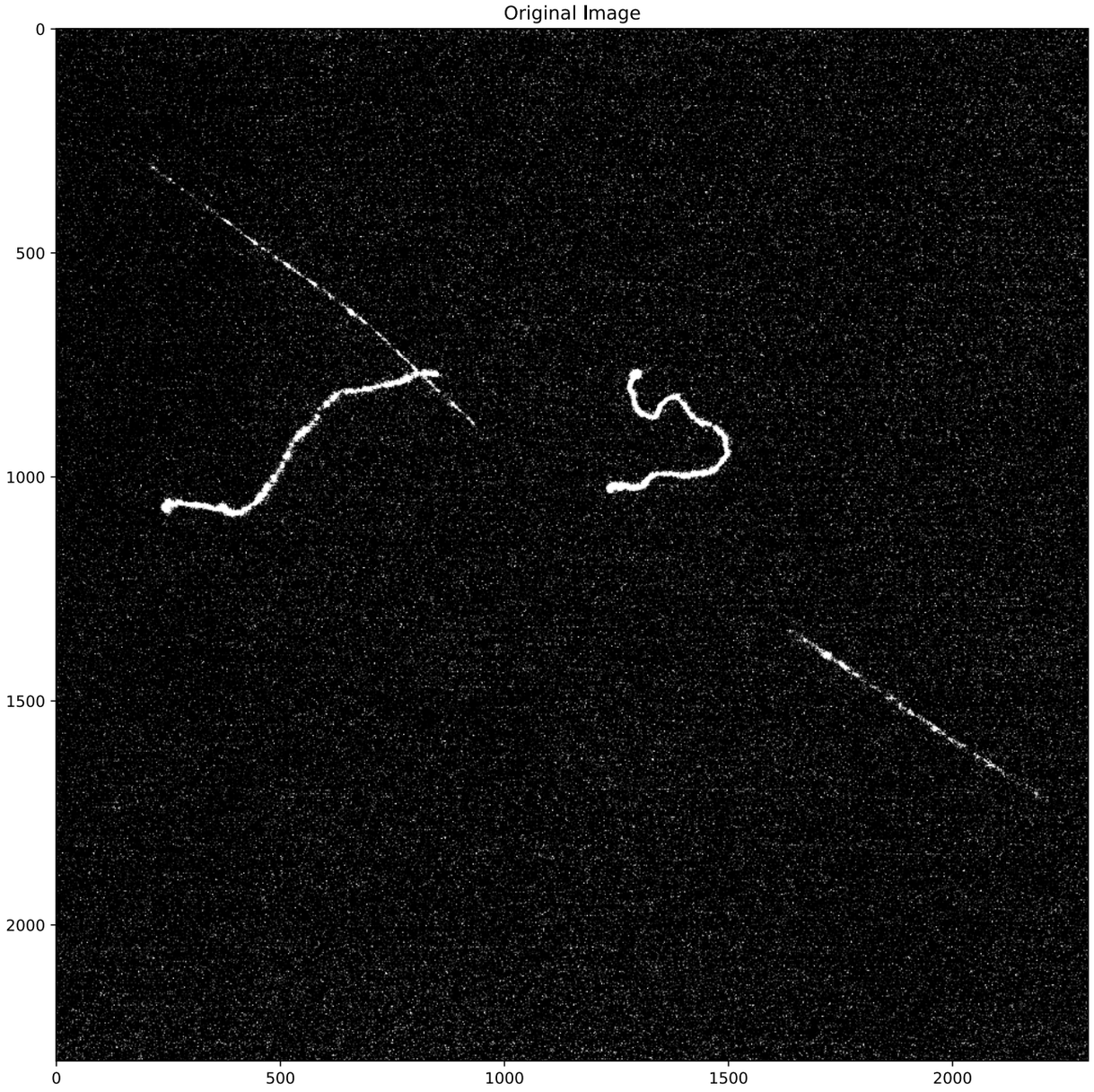


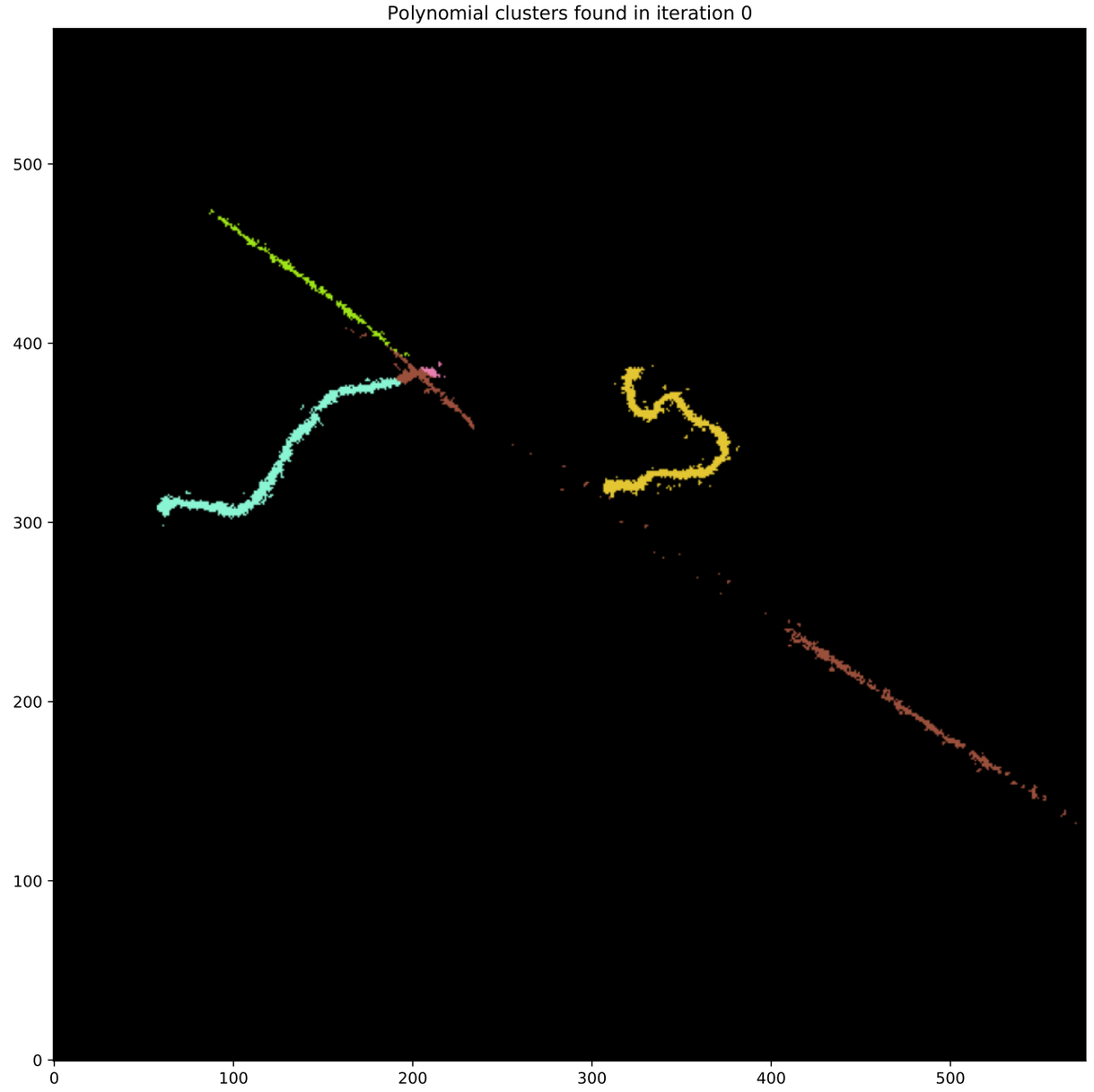


Long track split in two

Halo of low density hits

Example 3: bkg-only





manages to split correctly overlapping tracks.

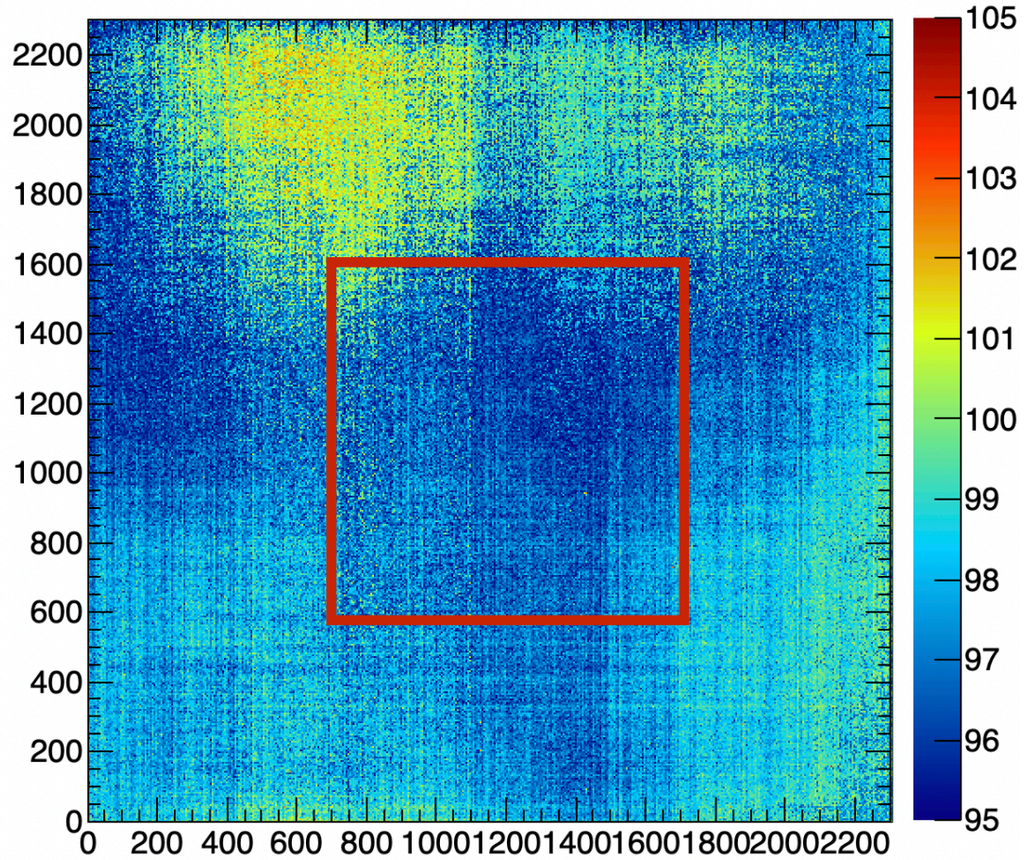
N.B. The overlap is not shared: first clustered gets the hits...

Rereco of AmBe data

The speed up of the code of factor >10 allowed to reconstruct:

- ~half of AmBe runs taken with LIME during the Summer: [3737-3791]: 26k events

- 3 runs of bkg-only [3792-3794] (the later ones have sizeable energy scale shift)



- N.B. This set of data is affected by the light entering the camera corners.

➡ select a small "dark" region in the center (~1/4 acceptance)

➡ cannot use de/dx of muons (long straight tracks) for E calibration (need to use only the part in the dark region)

Selections

Despite the small acceptance, the statistics is very large wrt LEMON (~10 x larger)

Cosmic Control Region (CCR)

- length > 10cm, |1-pathlength/length|<10% (straightness), slimness < 0.15, σ_T^{Gauss} <= 1.5 pixel

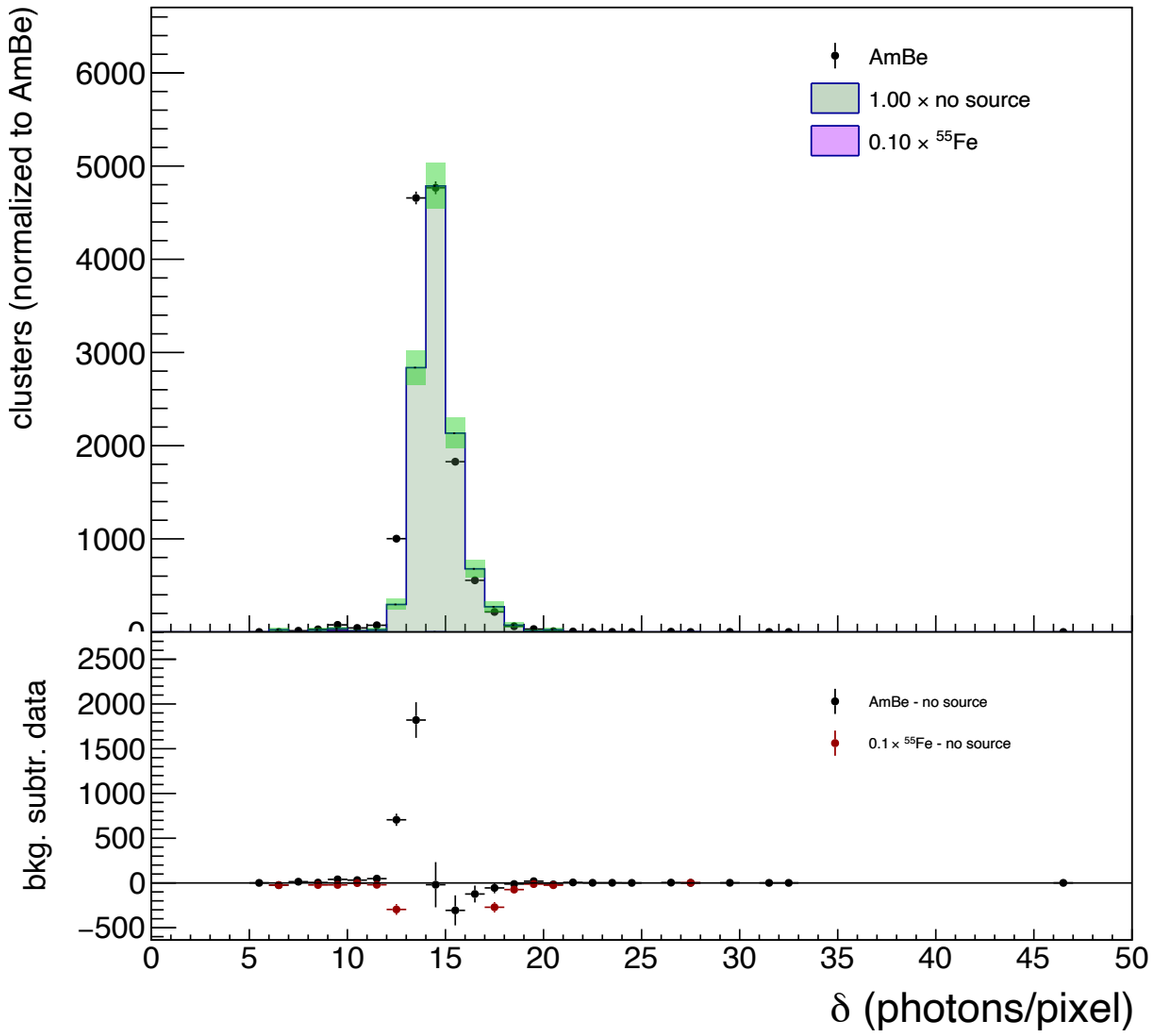
Signal Preselection:

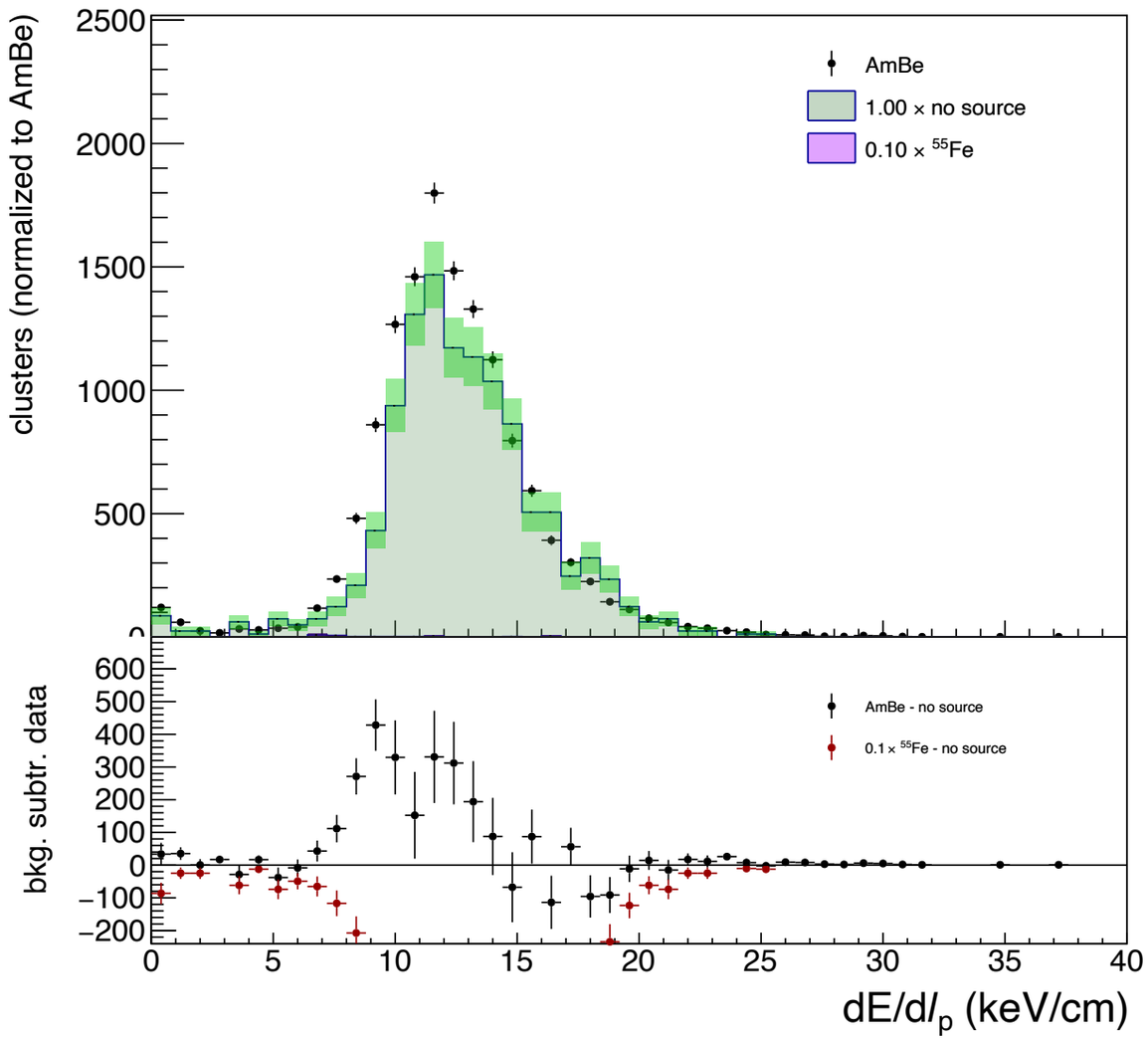
- length < 5cm, slimness>0.4, width<6.5 mm, σ_T^{Gauss} < 0.3mm

Signal Tight Selection:

- In the LEMON AmBe paper we used δ = Integral/pixels > XXX

Cosmics CR



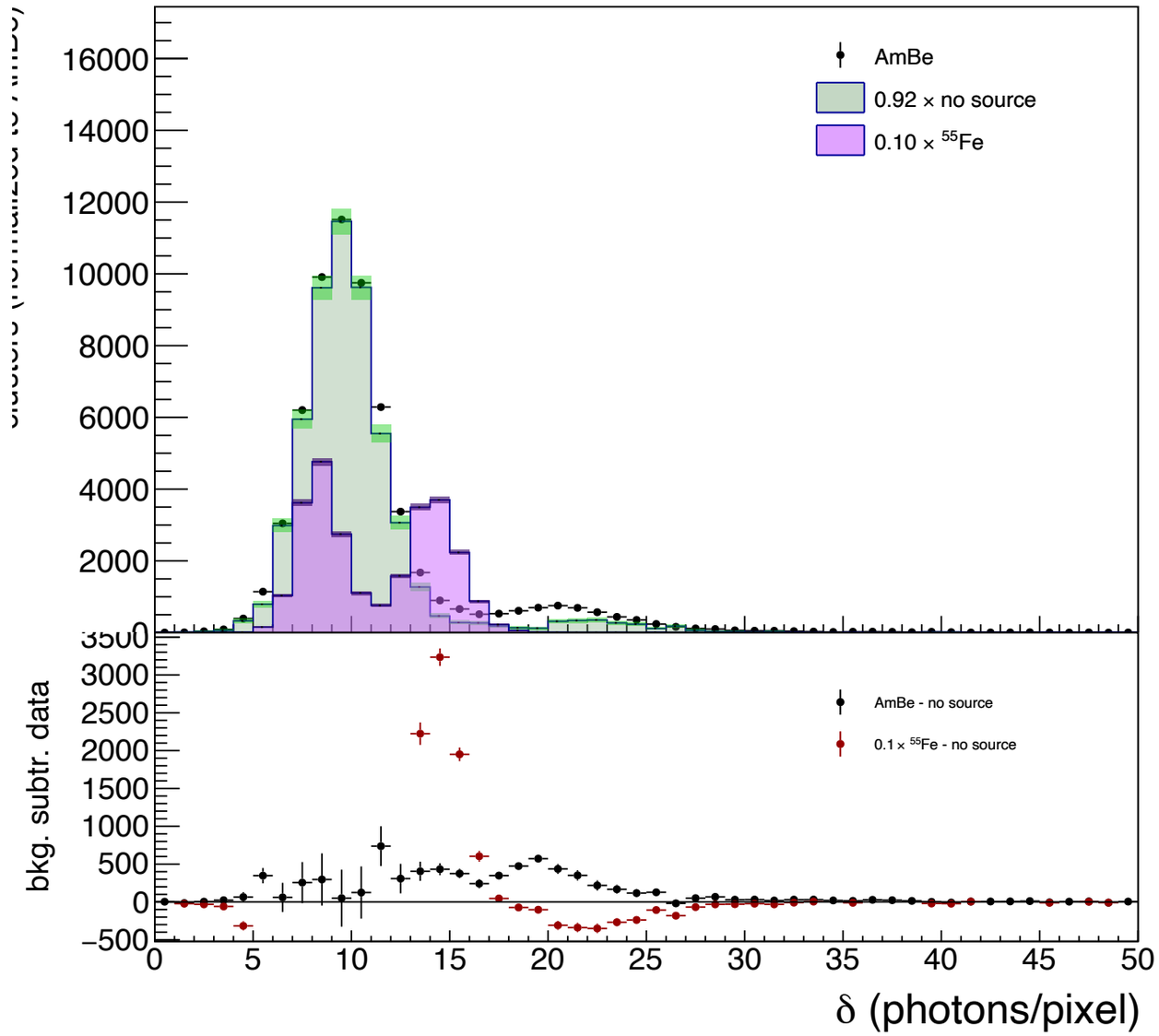


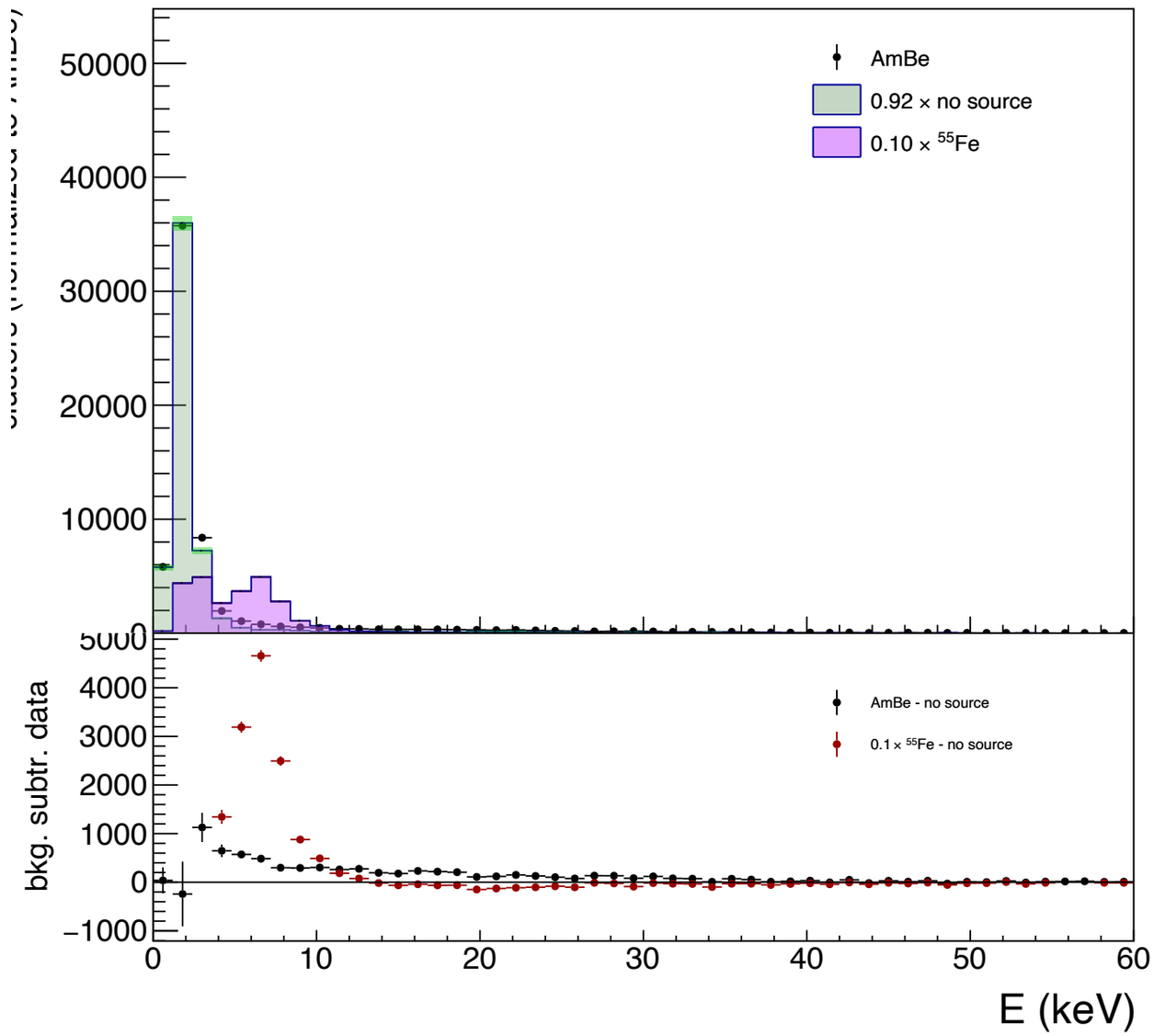
* Density: long tracks have δ~14.
* Slight shift wrt 2 sets of runs.
* Normalization scale factor

AmBe/bkg = 0.92

dE/dx on full track unusable (bias due to light in the corners)

Signal preselection





Density shows a clear peak around 10. These are leftover of long tracks (also in Fe - with a small shift)

The events with \delta>17 are high-energy NR candidates, [10-17] could be NRs as well as bkg

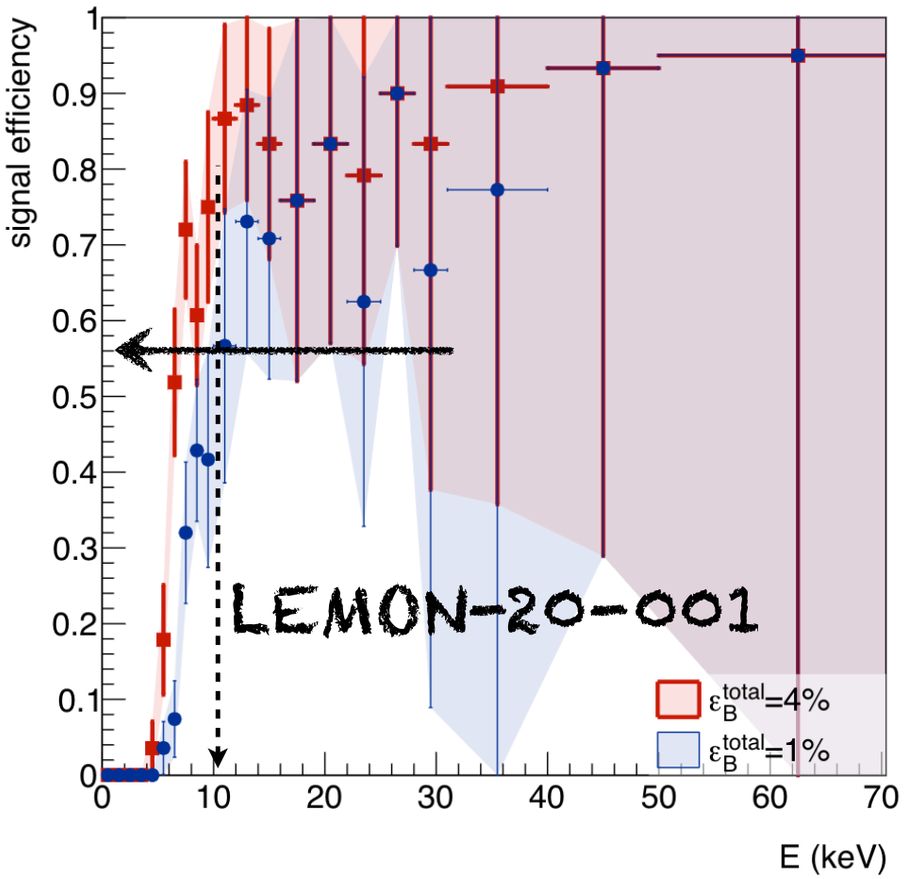
Energy spectrum seems exponential, no efficiency for E<3 keV

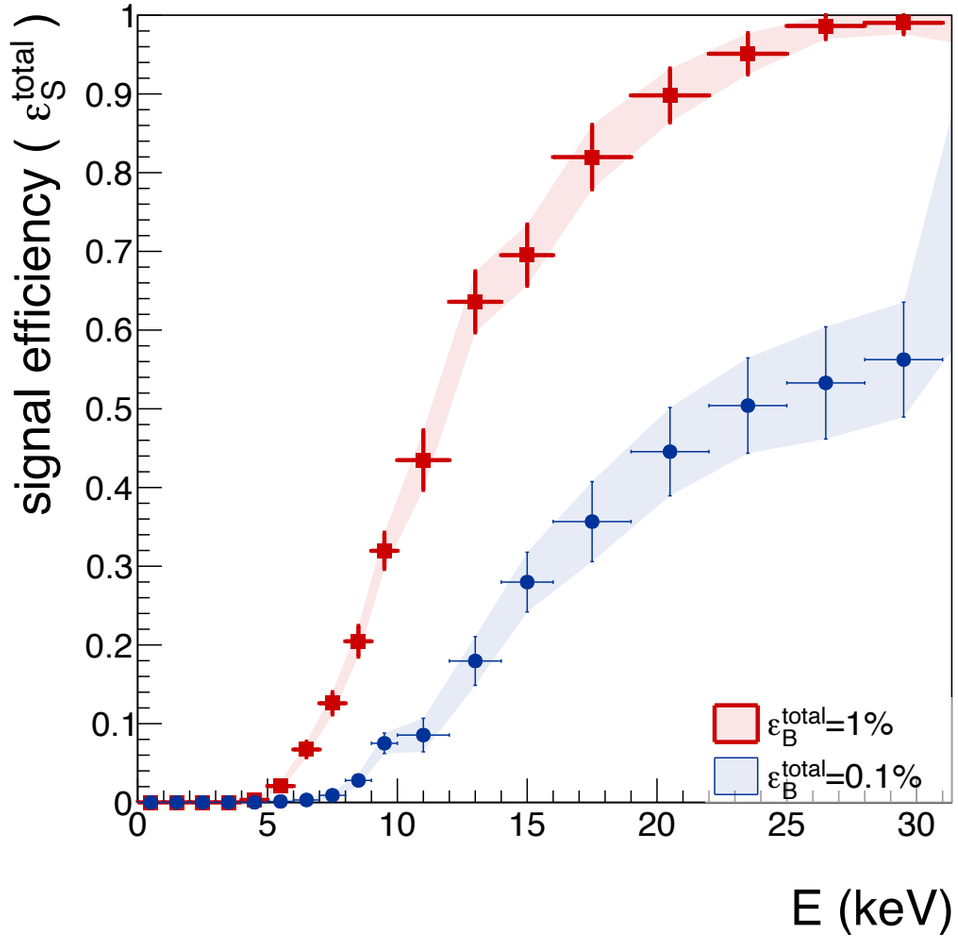
Efficiency for fixed ER rejection

Define two selections on δ at a given Fe rejection (i.e. rejection of ERs of E=6 keV).

In LEMON tested 99% and 96% rejection. Try 99% and 99.9% rejection. Compute the efficiency of this cut as a function of E.

Baracchini et al, *Measur.Sci.Tech*. 32 (2021) 2, 025902





For 10^{-2} ERs efficiency the NR at ~[6-8]keV improves from 5-8% to [8-15]%. For 10^{-3} ER efficiency, NR inefficient for E < 8 keV.

Beyond simple 1D cut

Until now, used only the cut on 1 variable (δ), while more cluster shapes are available.

Try an intermediate step to go beyond it and use more the shape of (few) cluster shapes, and their correlations (which are also informative).

Training a multiclass Deep Neural Network (DNN) with 3 output classes:

1. NRs 2. ERs (6keV) 3. other backgrounds (cosmics, natural radioactivity

The output in each class is normalized such that for a given cluster it represents the probability to be in that class (i.e.

$$\sum_{c=1}^{N=3} p_i^{clu} = 1$$

Training samples

Apart ML technicalities, this is the bottleneck. We need a pure sample of each class to train the model.

Class 3. can be taken from data (bkg-only runs)

Class 2. can be taken from Fe runs, with a loose selection to remove pieces of cosmic tracks (e.g. loose cut on δ)

Class 1 (NRs signal). Is the most problematic. We don't have it. A perfect candidate would be SIM. Tried with a sample that Giulia quickly produced, but verified that there is some large data/MC discrepancy.

- Used the AmBe sample in data
  - bkg-subtracted to get the 1D distributions
  - generate toy MC to get events according 1D shapes
  - assume the same correlation matrix of the sig+bkg sample in data

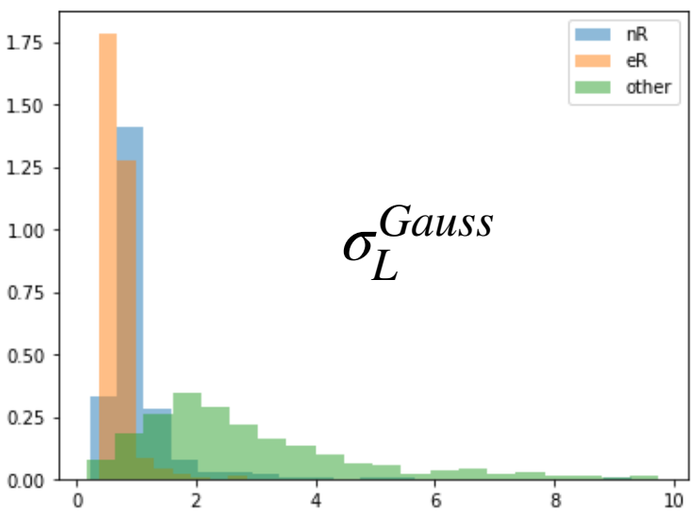
Variables used

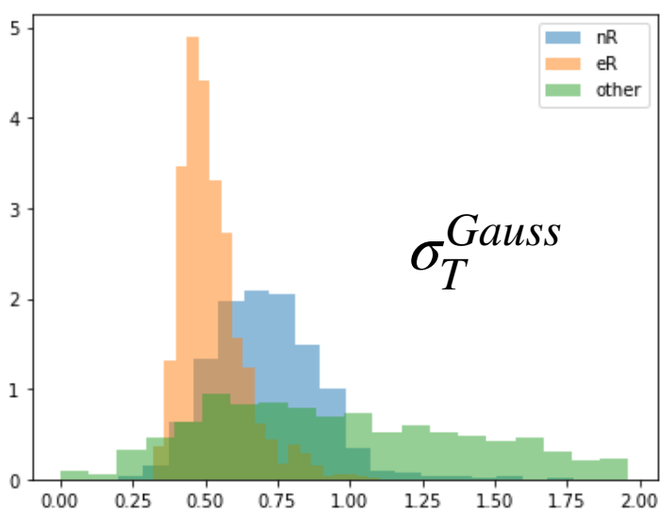
9 variables: δ, length, slimness, σ_L^{Gauss}, σ_T^{Gauss}, RMS_L, RMS_T, size (n pixels), N_{hits} (n pixels/threshold)

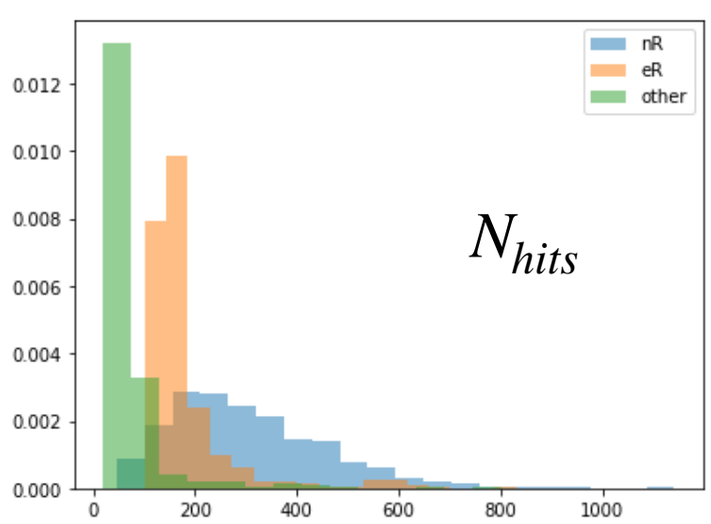
N.B. 1 Avoid to put energy (apart normalized to N_{hits}) not to make the DNN to learn the spectrum of Fe/NRs in the input

N.B. 2: train after the NR preselection. Can be made more effective/aggressive training on all the reconstructed clusters

N.B. 3: 9 variables is a joke for a DNN. Can be made more effective e.g. putting the energy release pattern in the slices along the cluster







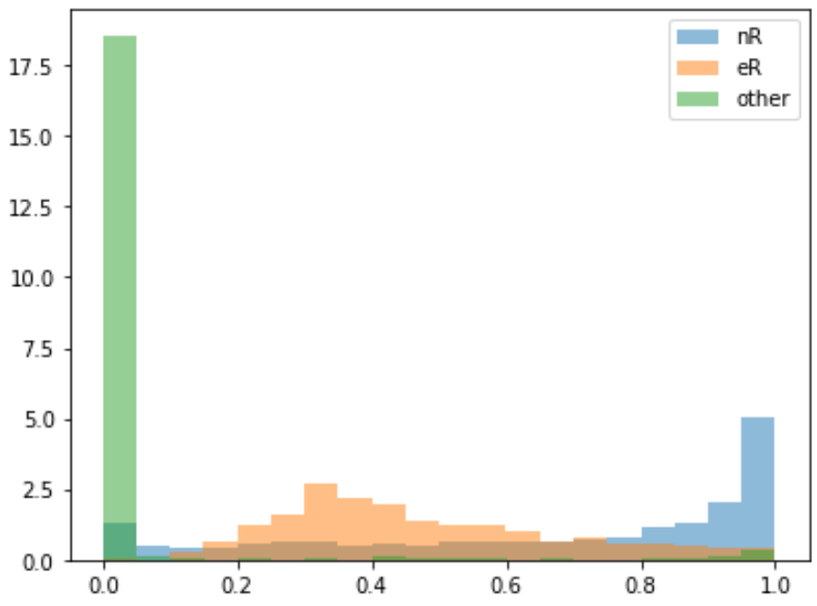
Training details

Trained with Tensorflow with 9 (input dimensions) -> 10 -> 7 -> 3 (output dimensions) nodes

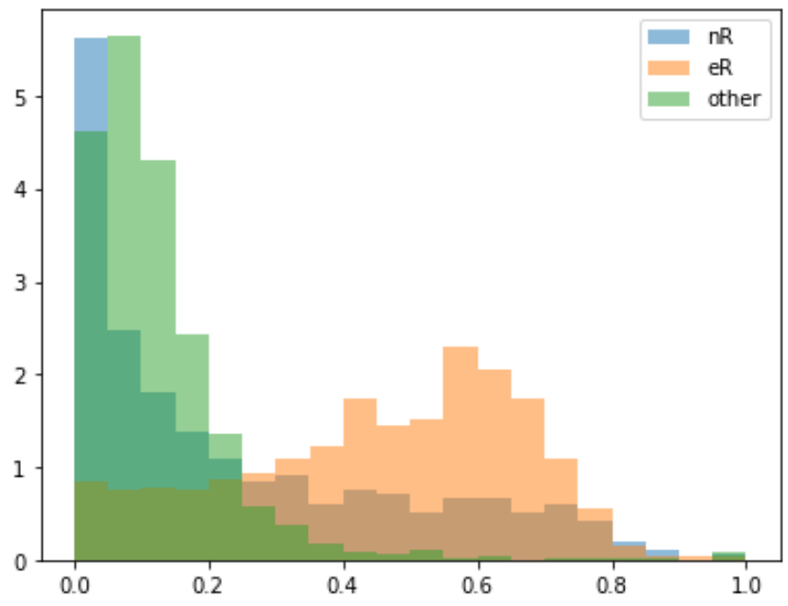
"Softmax" function used in the last step so that the 3 output values are in (0,1) with sum 1 (probability interpretation)

"Categorical cross-entropy" loss used to evolve the network

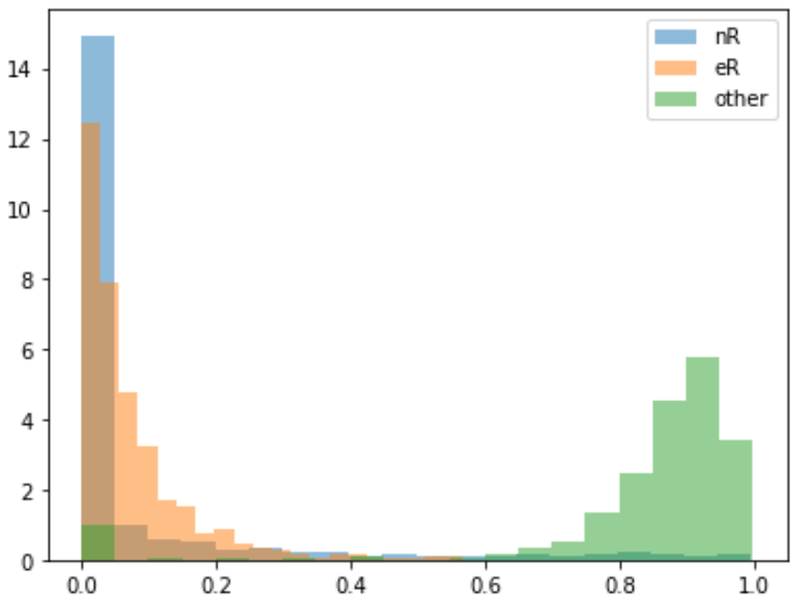
DNN scores in the 3 classes:



NRs DNN node



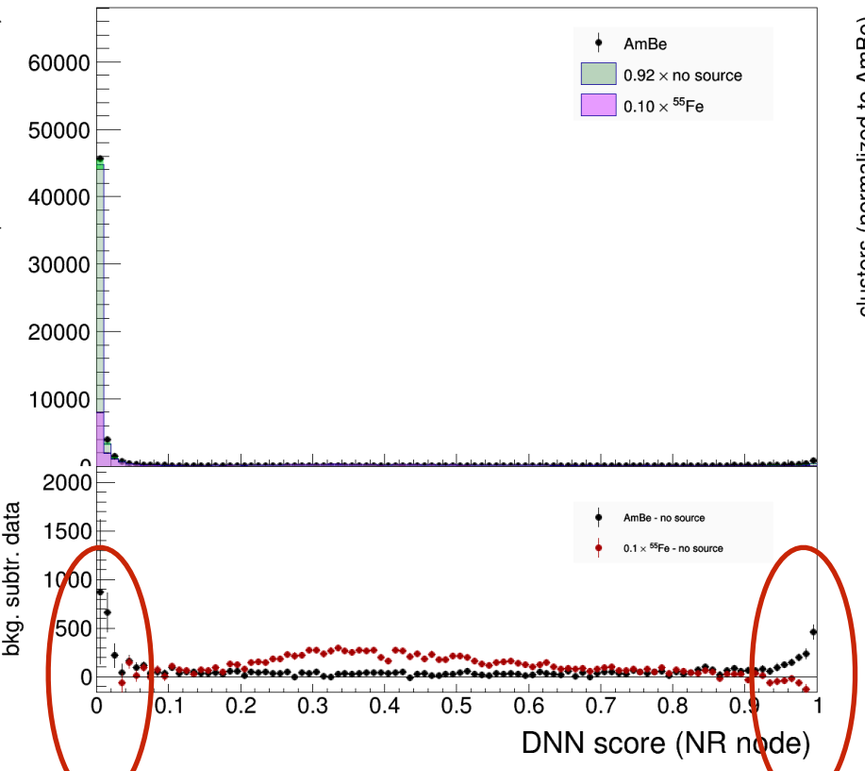
ERs DNN node



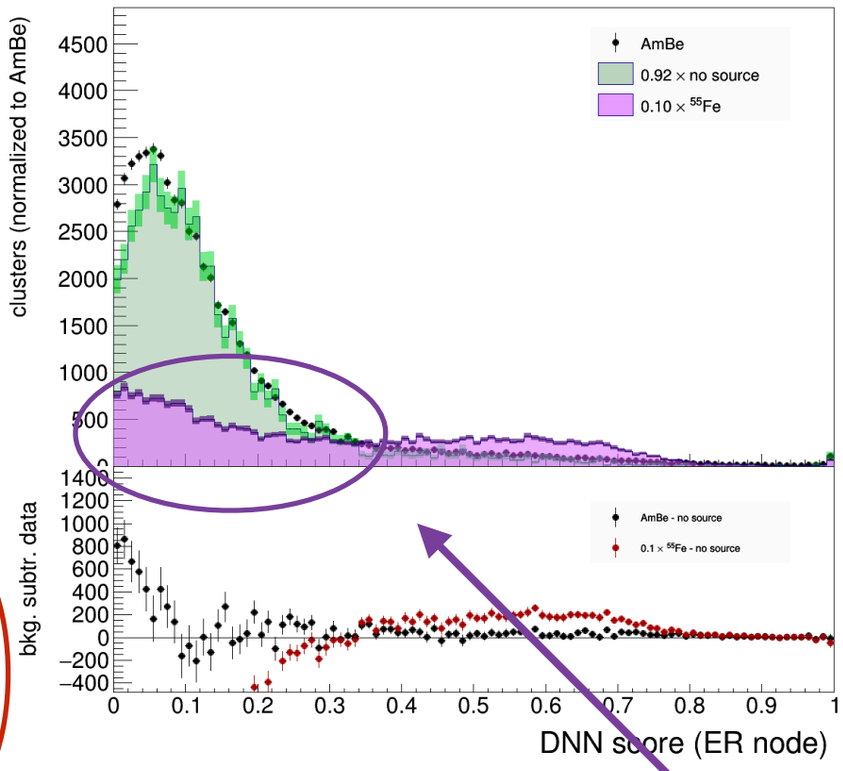
"other" DNN node

DNNs in data

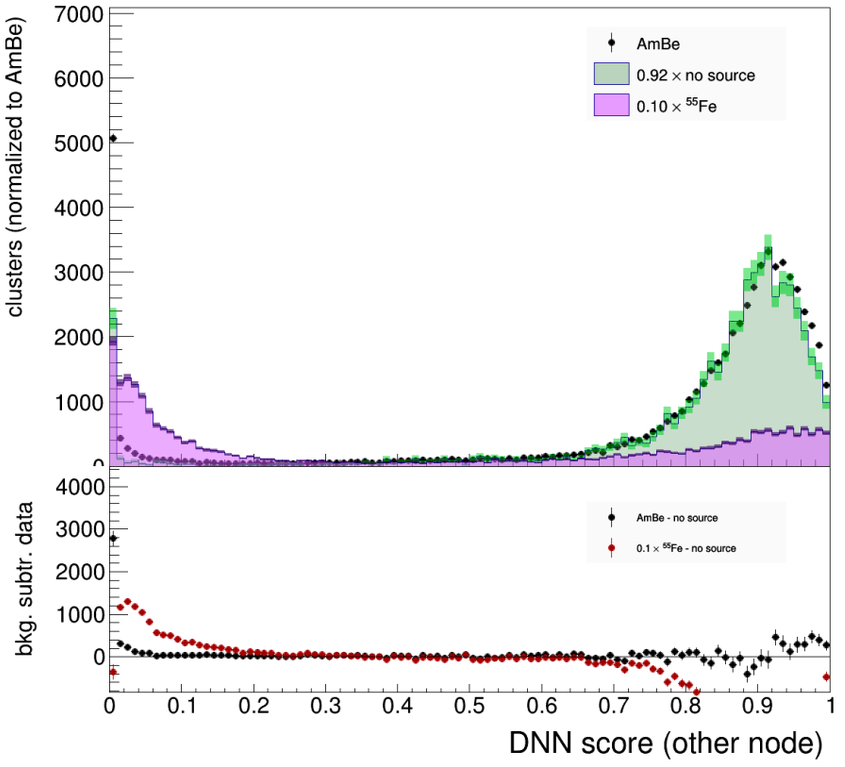
Compute the DNN scores for the 3 classes for each cluster



NRs DNN node



ERs DNN node



"other" DNN node

high-density NRs

these are the cosmic-pieces in Fe data

low-density NRs: this is a problem, comes from the sample definition?

Simple usage: 1D cut

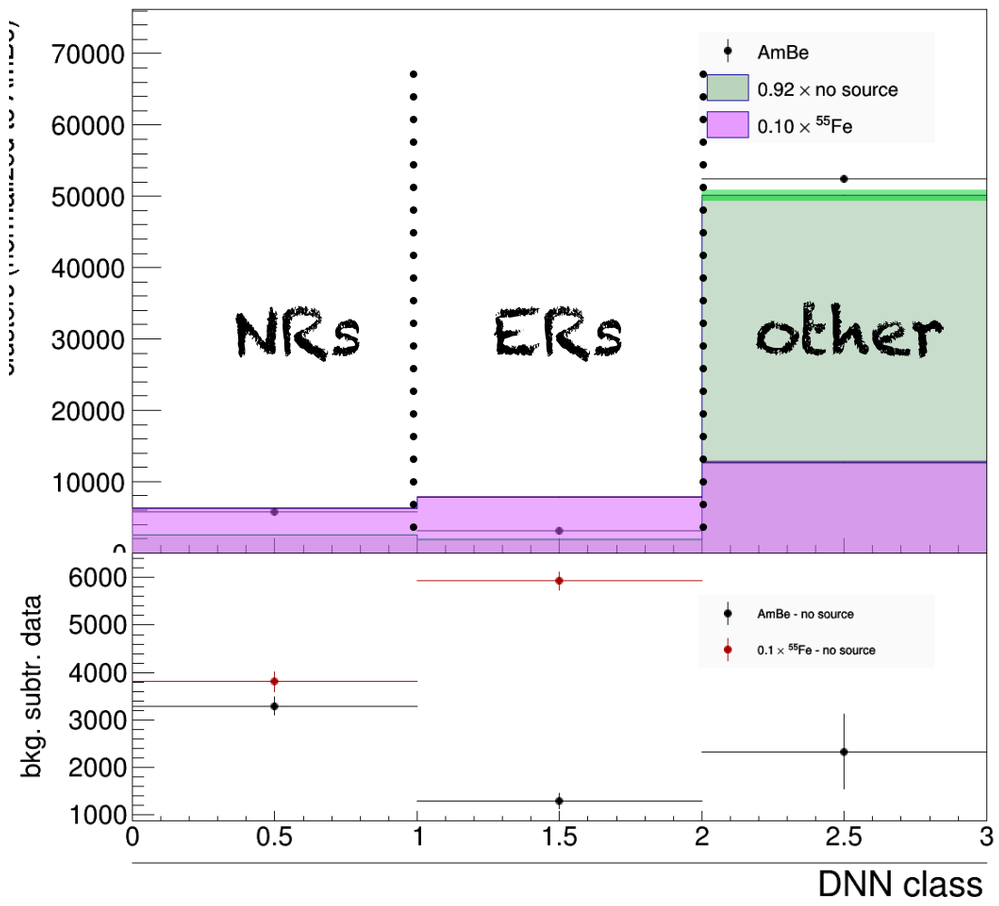
One can simply do the simplest usage of it. Substitute \delta with DNN NRs node output.

Fix the problem of the low-density clusters in DNN before evaluating the performance... stay tuned.

Extended shape usage

For each cluster, can use the max(DNN_{NR},DNN_{ER},DNN_{other}) to define the "category" of cluster.

This defines 3 samples, each one enriched of that class of clusters



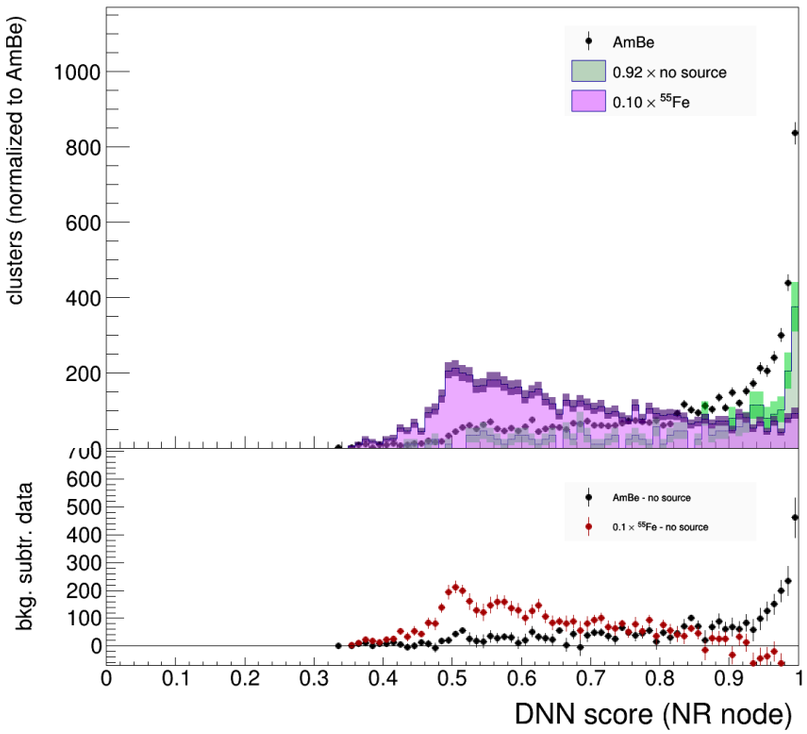
Each cluster enters the distribution "finalDNN"
only once, using the output for specific cluster category

A simultaneous fit of the finalDNN shapes in the 3 categories allows to normalize the ERs / other background in data.

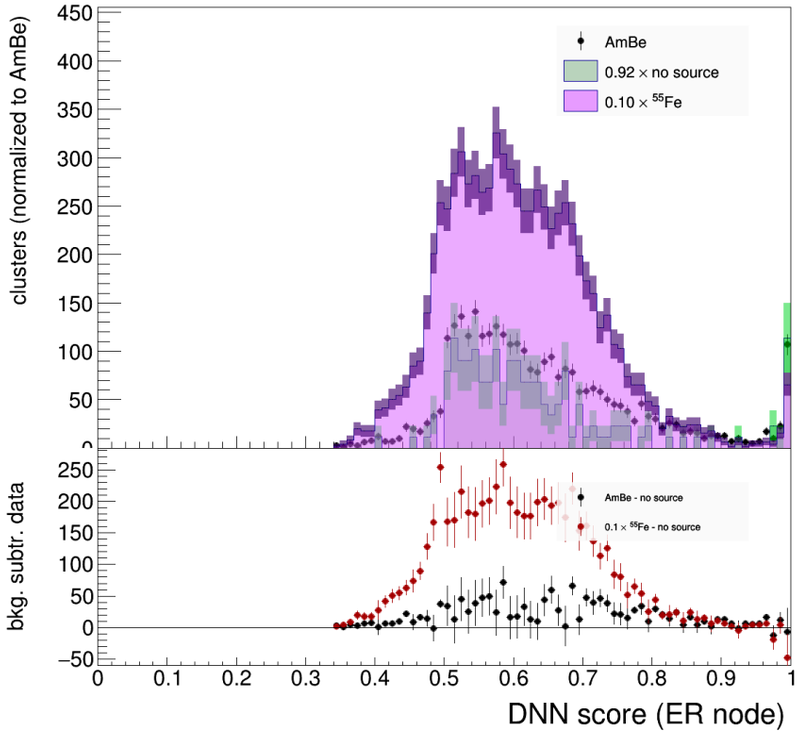
All bkg uncertainties correctly propagated with the fit covariance matrix

Fit variable "finalDNN"

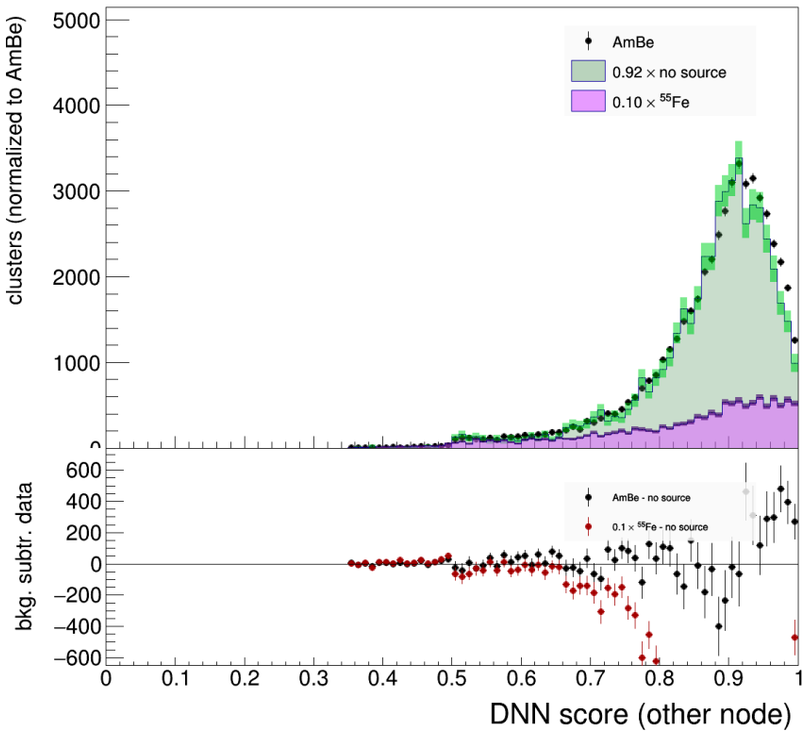
E.g. The fit to the "other node" of DNN in the 'others' category will constrain the small "other" background in the NRs category



NRs (signal) category



ERs category



others category

A way to test this approach is take data close in time with LIME and: 1. no-source 2. Fe^{55} only 3. AmBe 4. AmBe + Fe^{55}.
Compare NR/ER/other yields in (4) one with 1., 2. and 3.

Conclusions

Speeded up / tuned for LIME code of cluster reconstruction (V6 tag of the "lime21" GIT branch)

First tests on SIM not crazy (thanks Giulia). Need to look at the E resolution.

Re-reconstructed all the AmBe / cosmics / Fe data with V6

Re-analysis a la LEMON-20-001 seems ok (up to x2 efficiency in the region 6-8 keV for 10^{-2} ERs efficiency)

Tried a multivariate approach to use better cluster shapes instead of only δ. Two approaches started:

1. cut based

2. simultaneous fit of MVA output in 3 categories

Both to be finalized...