

# Fuzzy clustering

Ben Carls

*Fermilab*

(Dated: January 23, 2013)

## Abstract

An article usually includes an abstract, a concise summary of the work covered at length in the main body of the article.

## I. INTRODUCTION

This clustering algorithm attempts to separate physics objects such as electrons and muons within a single 2-D plane. It does so using a multistep approach. The first step involves application of the fuzzy c-means clustering algorithm. The second step identifies lines using a Hough transform. The third step merges the fuzzy clusters into the identified Hough lines to create physics object driven clusters.

To demonstrate the clustering algorithm, the event displays for two GENIE events will be shown as they step through the algorithm. The unclustered event display for a CCQE muon event appears in Fig. 1. The unclustered event display for a CCQE electron event appears in Fig. 2.

For the muon event, the longer track appearing in the lower portion of each view of the event display come from the muon. The shorter track appearing in the upper portion of each view of the event display is created from a proton in the event. The goal of this clustering algorithm is to place the hits generated from the proton into a separate cluster than those of the muon hits.

For the electron event, the shower appearing in each view of the event display results from the electron. The track appearing is then the result of a proton. For this event, the goal of clustering is to separate the hits of the proton from those of the electron.

## II. FUZZY C-MEANS CLUSTERING

The first step runs the fuzzy c-means clustering (FCM) algorithm. Unlike the DBSCAN algorithm, hits in the FCM algorithm have varying degrees of belonging to a given cluster. The FCM algorithm is described in detail elsewhere [1], a brief description of the algorithm follows.

For a given set of  $n$  elements (hits in this case)  $X = x_1, \dots, x_n$ , the FCM algorithm places them into  $c$  number of clusters with centers  $C = c_1, \dots, c_c$ . The amount of belonging of each hit is described by a partition matrix  $W = w_{i,j} \in [0, 1], i = 1, \dots, n, j = 1, \dots, c$ . The FCM algorithm attempts to minimize an objective function of the form

$$w_k(x) = \frac{1}{\sum_j \left( \frac{d(\text{center}_k, x)}{d(\text{center}_j, x)} \right)^{2/(m-1)}},$$

with  $m$  being the fuzzifier, defining the amount of fuzziness between clusters.

The cluster centers are then given by

$$c_k = \frac{\sum_x w_k(x)x}{\sum_x w_k(x)}.$$

The FCM algorithm is implemented in the following manner:

1. A number of clusters is determined
2. Each hit is randomly assigned an amount of belonging to each cluster

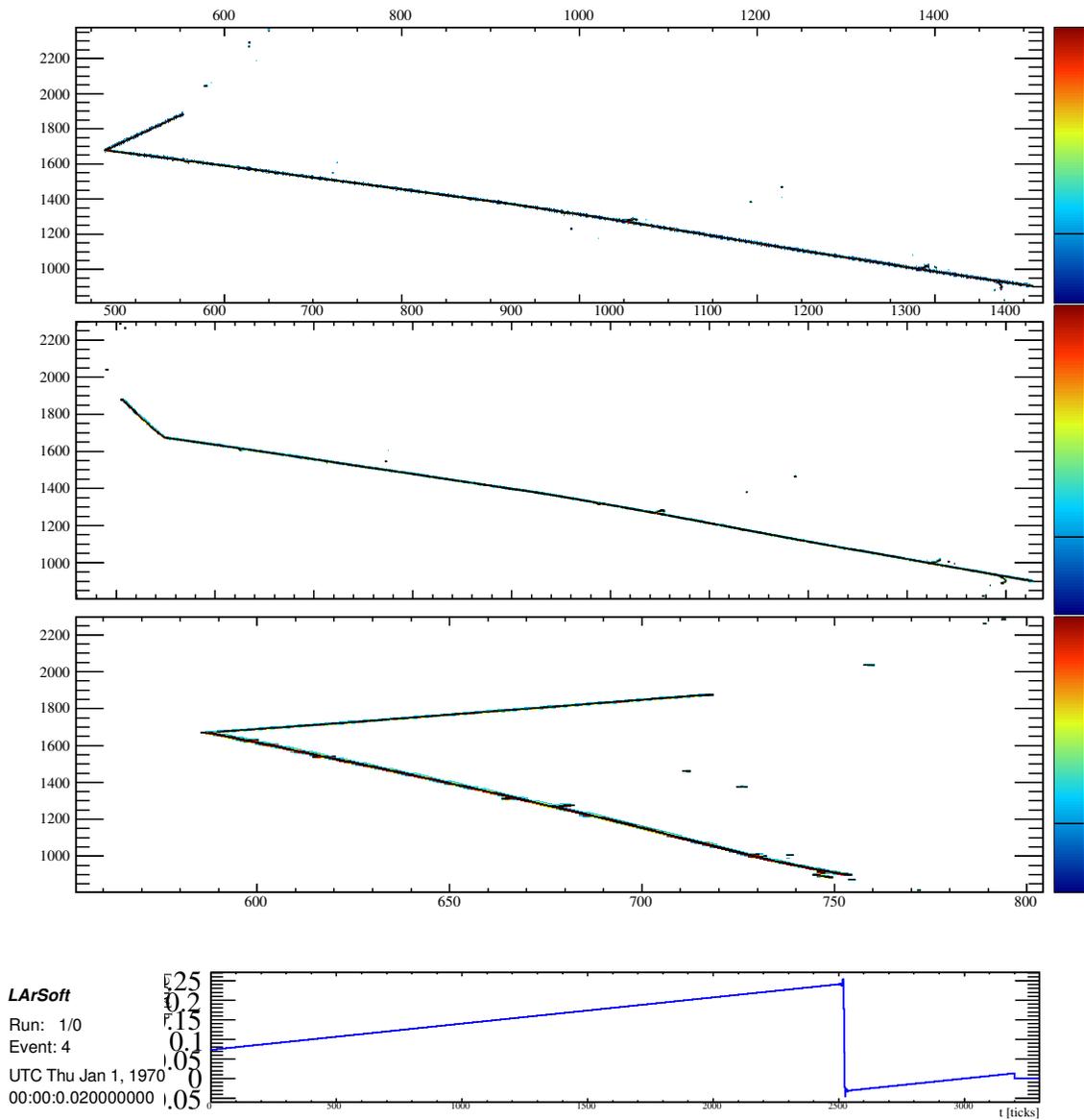


FIG. 1. The CCQE muon event which will be stepped through the fuzzy clustering algorithm.

3. Repeat until a convergence has been found for the objective function
  - (a) Compute the centroid of each cluster
  - (b) Compute the new belongingness of each hit for each cluster

An issue with the FCM algorithm is the need to specify the number of clusters as an input. The algorithm as it is currently implemented in LArSoft, attempts fuzzy clustering with 9 different numbers of clusters,  $c = 2, \dots, 10$ . To determine the optimal number of clusters, two steps are taken. Firstly, a validity measure is examined for each number of

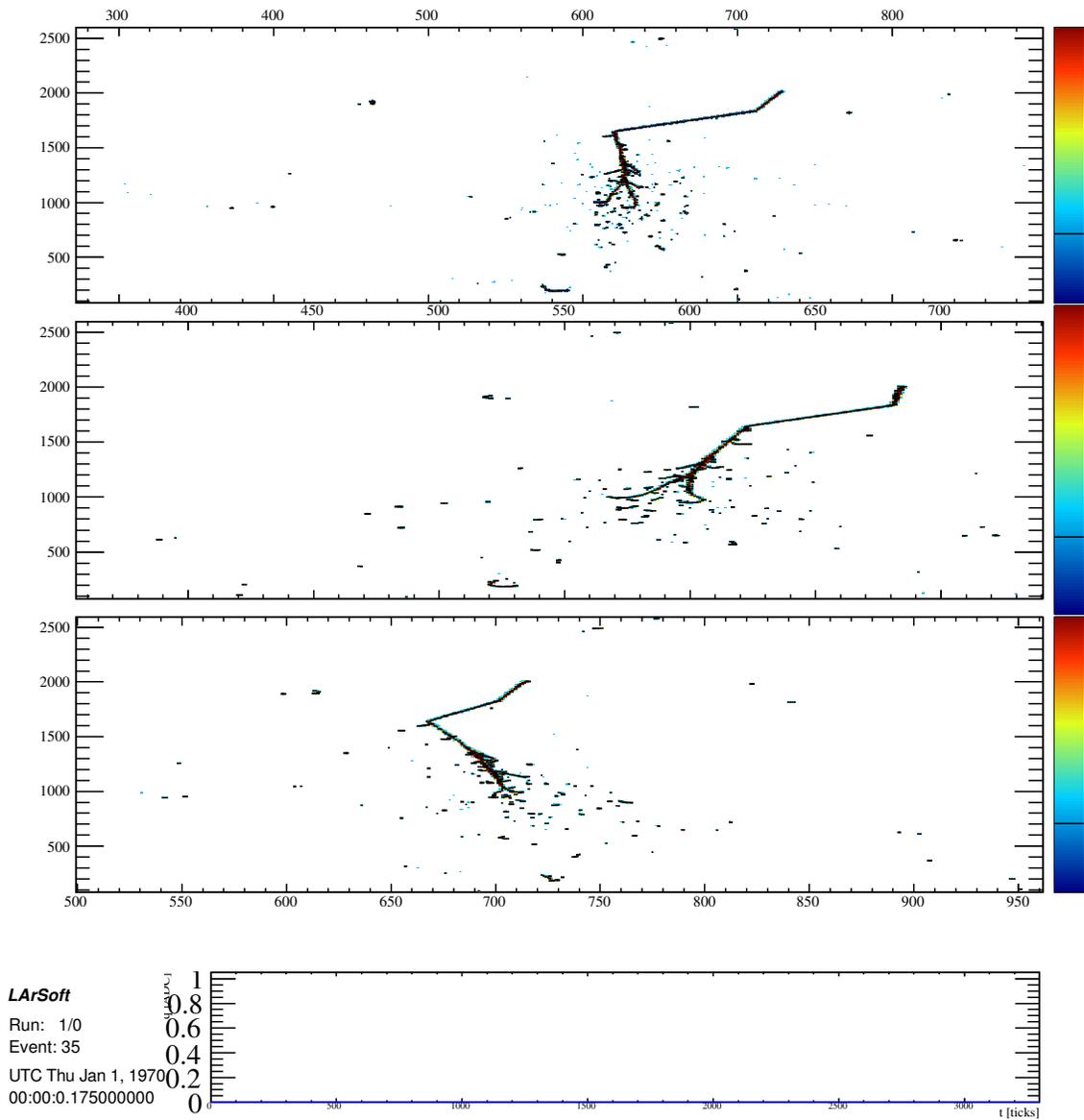


FIG. 2. The CCQE electron event which will be stepped through the fuzzy clustering algorithm.

clusters. Secondly, the fuzzy clusters are merged.

The validity measure used is the Xie-Beni index [2]. The index is defined as

$$XB = \frac{\frac{1}{n} \sum^n i = 1 \sum^c j = 1 w_{i,j} \|x_i - c_k\|^2}{\min_{k,l} \|c_k - c_l\|^2},$$

where  $\|x_i - c_k\|$  is the Euclidean distance between the  $i$ -th hit and the  $k$ -th cluster centroid and  $\|c_k - c_l\|$  is the Euclidean distance between the centroids of the  $k$ -th and  $l$ -th clusters.

The number of clusters is chosen based on which minimizes the index.

The results of running the FCM algorithm and choosing the number of clusters via the Xie-Beni index appears for the muon and electron events in Fig. 3 and 4 respectively.

The second step toward determining the number of clusters involves merging them based on Euclidean distance. This step searches for the point in cluster  $i$  closes to the centroid in cluster  $j$ . The distance is then determined between that point and the point in cluster  $j$  closest to the centroid of cluster  $i$ . If the distance is under a threshold, the clusters are merged.

The results of merging the fuzzy clusters through the Euclidean distance appears for the muon and electron events in Fig. 5 and 6 respectively.

### III. HOUGH LINE FINDING

After running FCM clustering, the algorithm searches for Hough lines in the already identified fuzzy clusters. The Hough line algorithm employed is the Progressive Probabilistic Hough Transform (PPHT) [3]. The primary advantage of the PPHT is it's speed. The algorithm is fairly simple, using the following steps:

1. Randomly select one hit, removing it from the collection of hits
2. Check if the highest peak in the accumulator modified by the hit is hight than a threshold cut; if not, go to step 1
3. If highest peak modified by the hit is above threshold, add hits along the line, removing them from the collection of hits
4. Remove hits from the accumulator from the line that was found
5. Repeat step 1 if the collection of hits is not yet empty

The results of the PPHT line finder appears for the muon and electron events in Fig. 7 and 8 respectively.

#### IV. MERGING THE PIECES TOGETHER

After Hough lines have been identified, an attempt is then made to merge them into physics objects. A muon will typically have a long track. The PPHT will typically identify

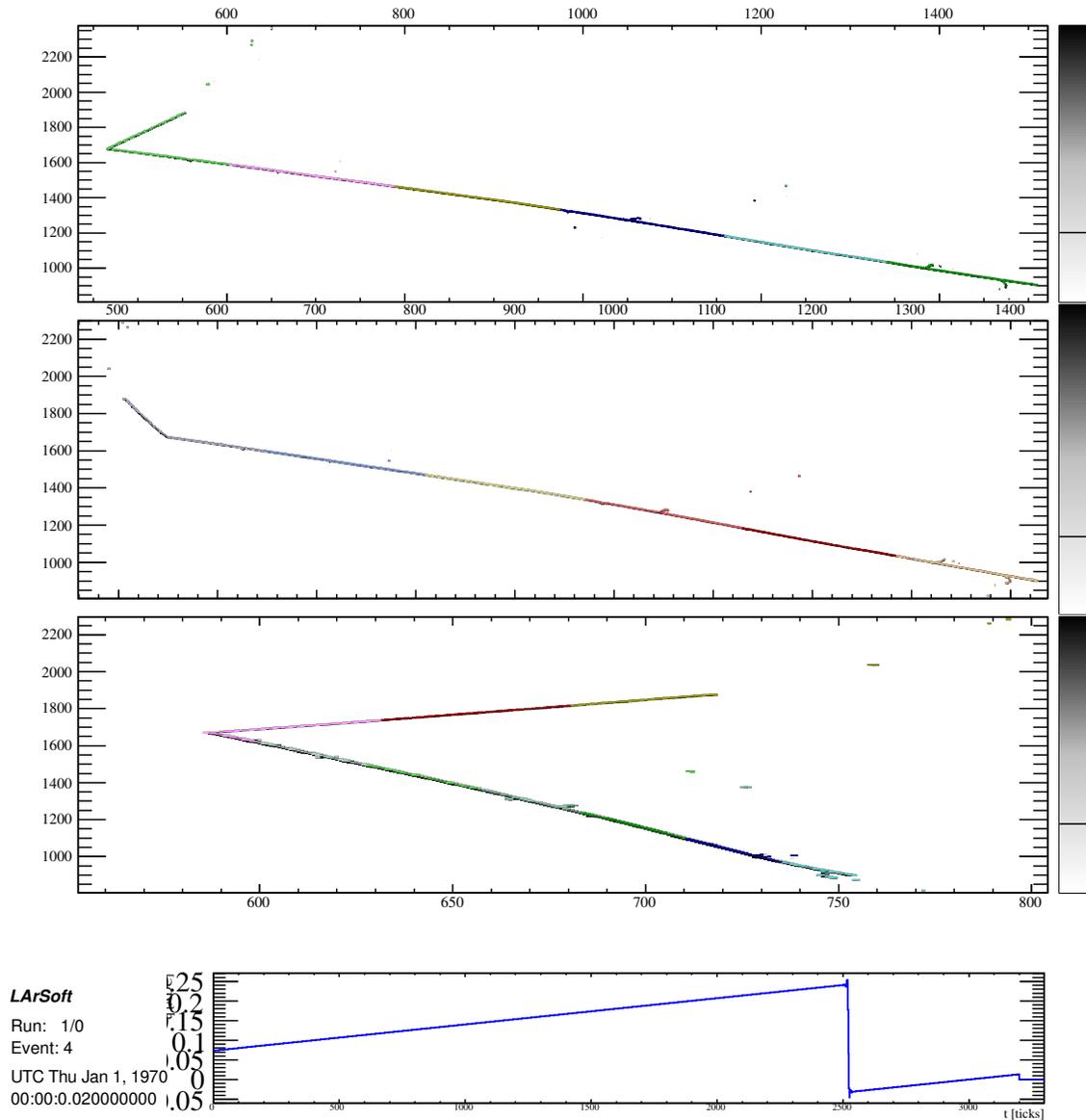


FIG. 3. The CCQE muon event run through the FCM algorithm and the number of clusters chosen via the Xie-Beni index.

multiple lines along the track. The merging is performed to link them together. For an electron shower, the PPHT typically identifies several lines within the shower. The merging's goal is to group lines in the shower together.

The merging criteria consists of the angle between the lines and the distance between them. If both criteria are met, the lines are merged. The results of the PPHT line merging appears for the muon and electron events in Fig. 9 and 10 respectively.

After completing the Hough line merging, the remnants of the fuzzy clusters remain. The

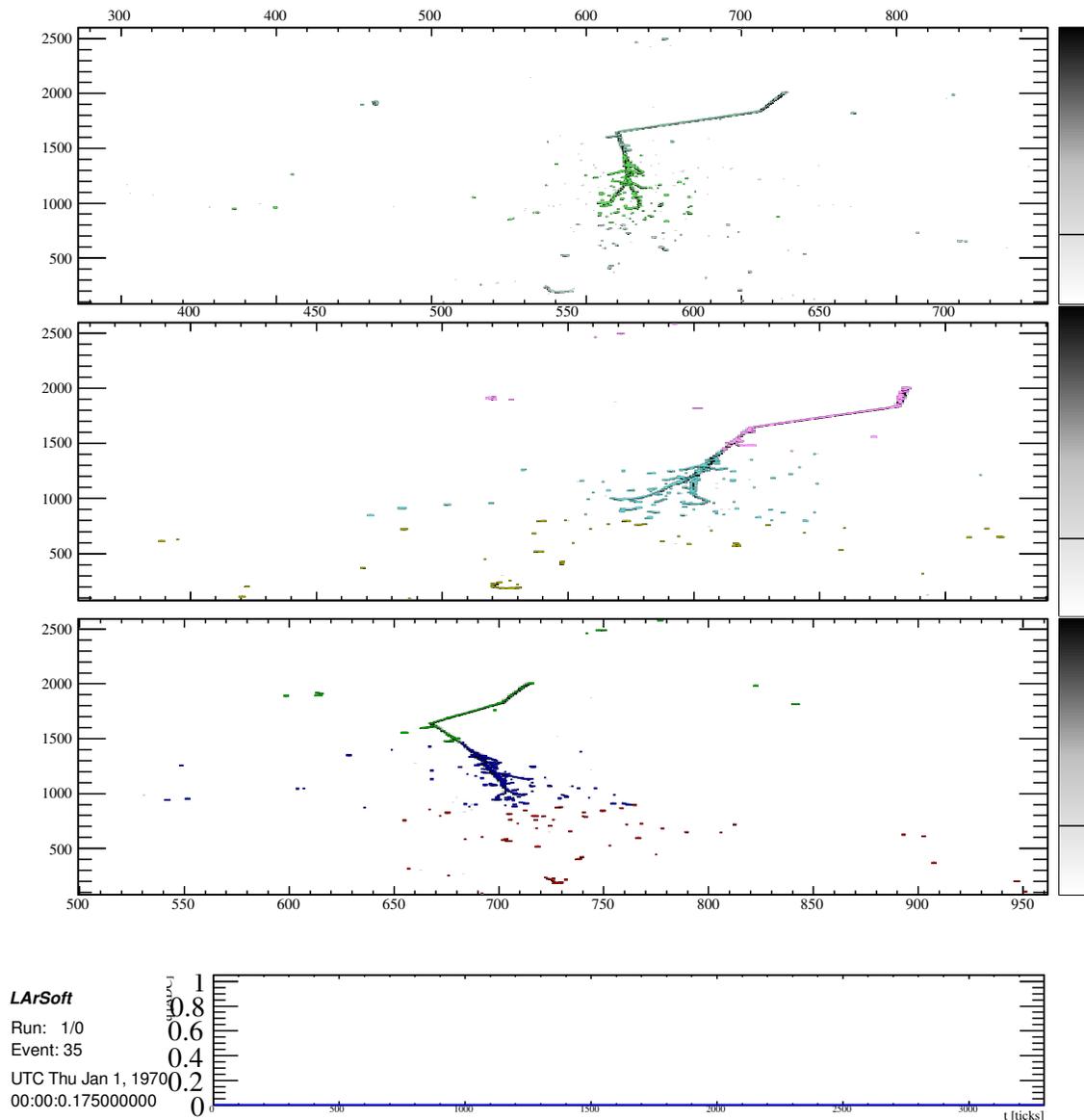


FIG. 4. The CCQE electron event run through the FCM algorithm and the number of clusters chosen via the Xie-Beni index.

final step in the algorithm is to merge the fuzzy cluster remnants into the recently merged Hough lines. The remnants are merged based on the metric of the distance to a collection of merged lines divided by the total number of hits in the collection of merged lines. The result of this merge appears for the muon and electron events in Fig. 11 and 12 respectively.

- [1] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms* (Kluwer Academic Publishers, Norwell, Massachusetts, 1981).

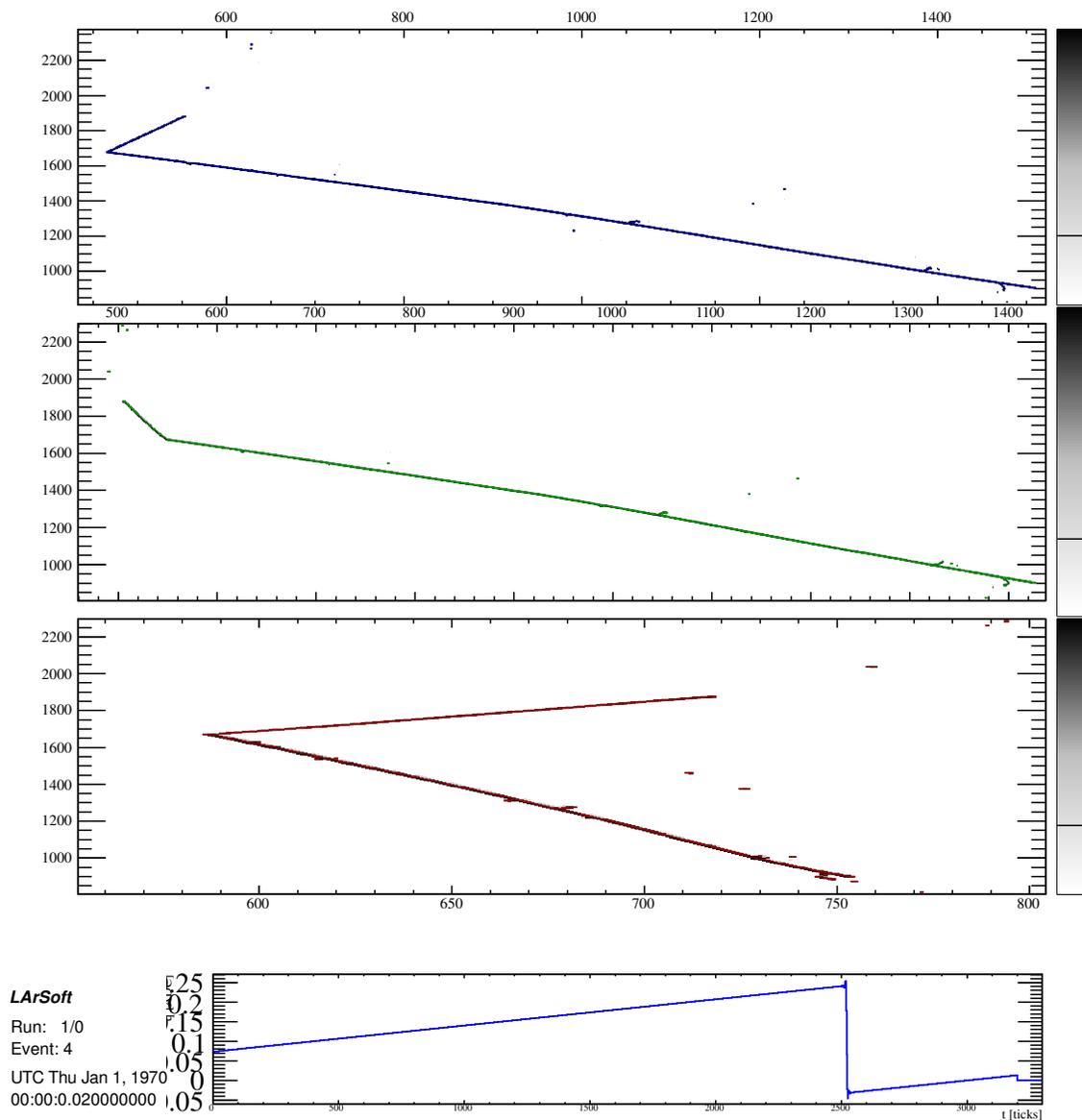


FIG. 5. The CCQE muon event with its fuzzy clusters merged by Euclidean distance.

- [2] X. Xie and G. Beni, IEEE Trans. Pattern Analysis and Machine Intelligence **13**, 841 (1991).
- [3] J. M. et al., Computer Vision and Image Understanding **78**, 119 (2000).

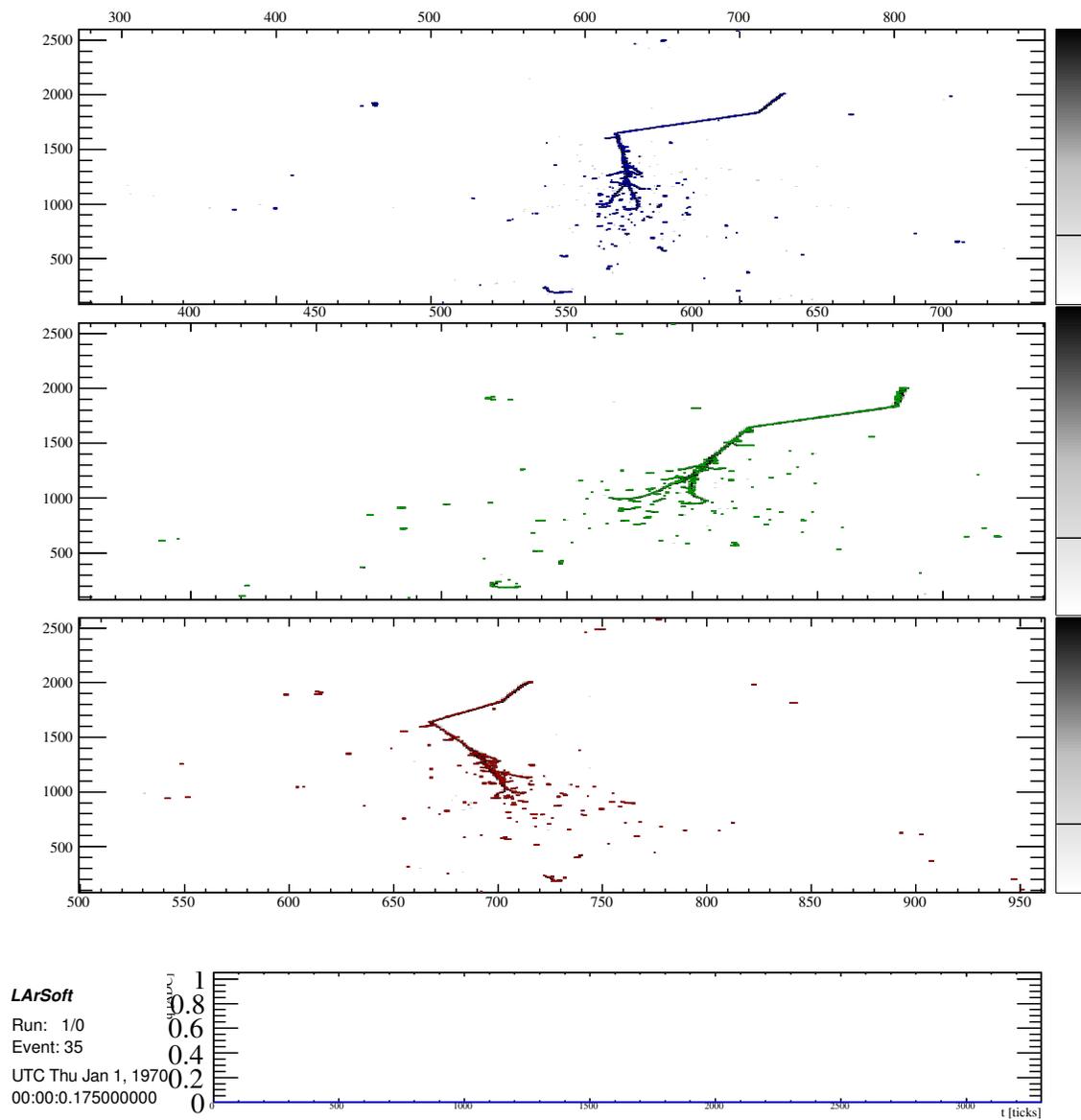


FIG. 6. The CCQE electron event with its fuzzy clusters merged by Euclidean distance.

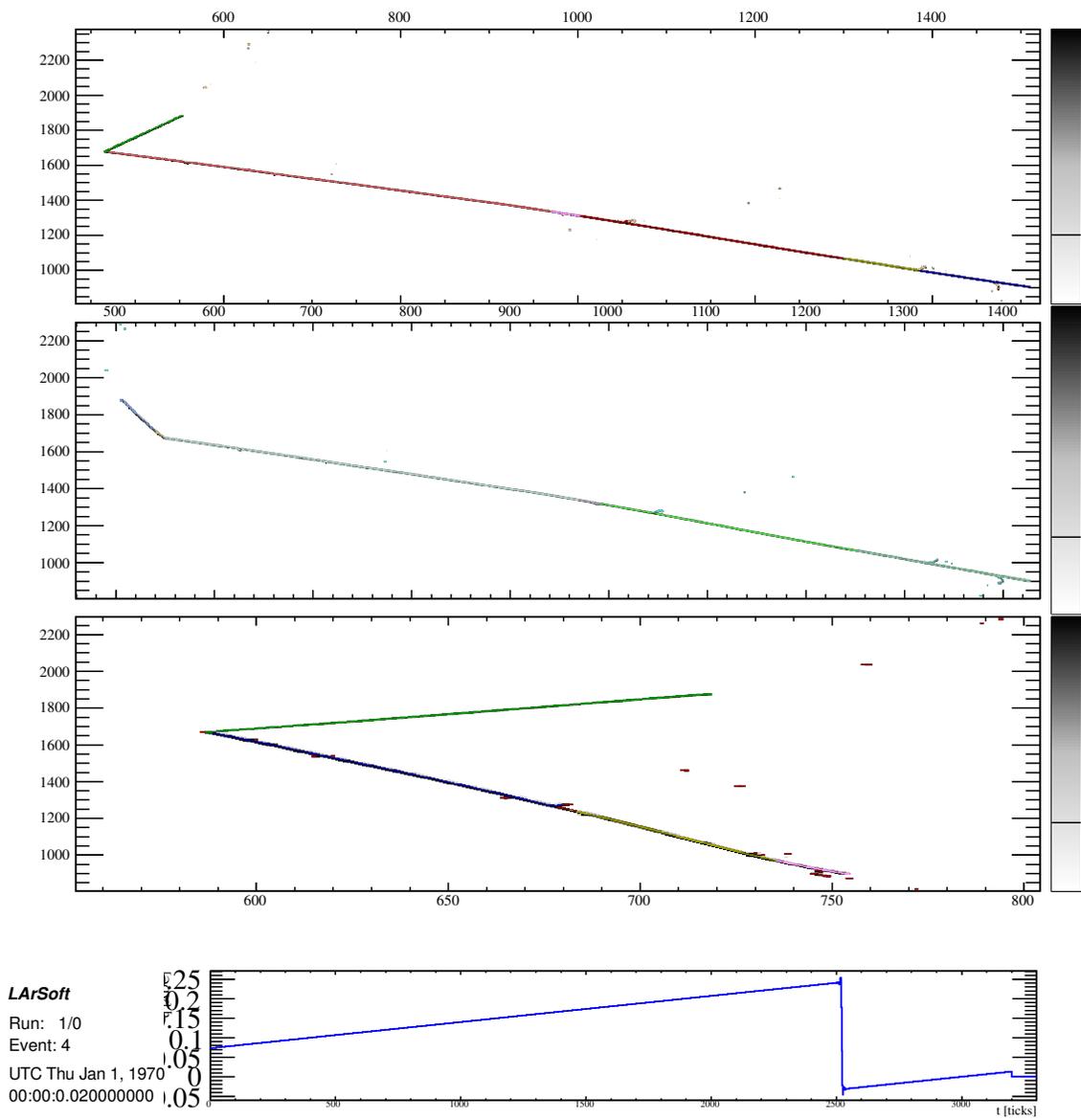


FIG. 7. The CCQE muon event after the PPHT line finder has been run.

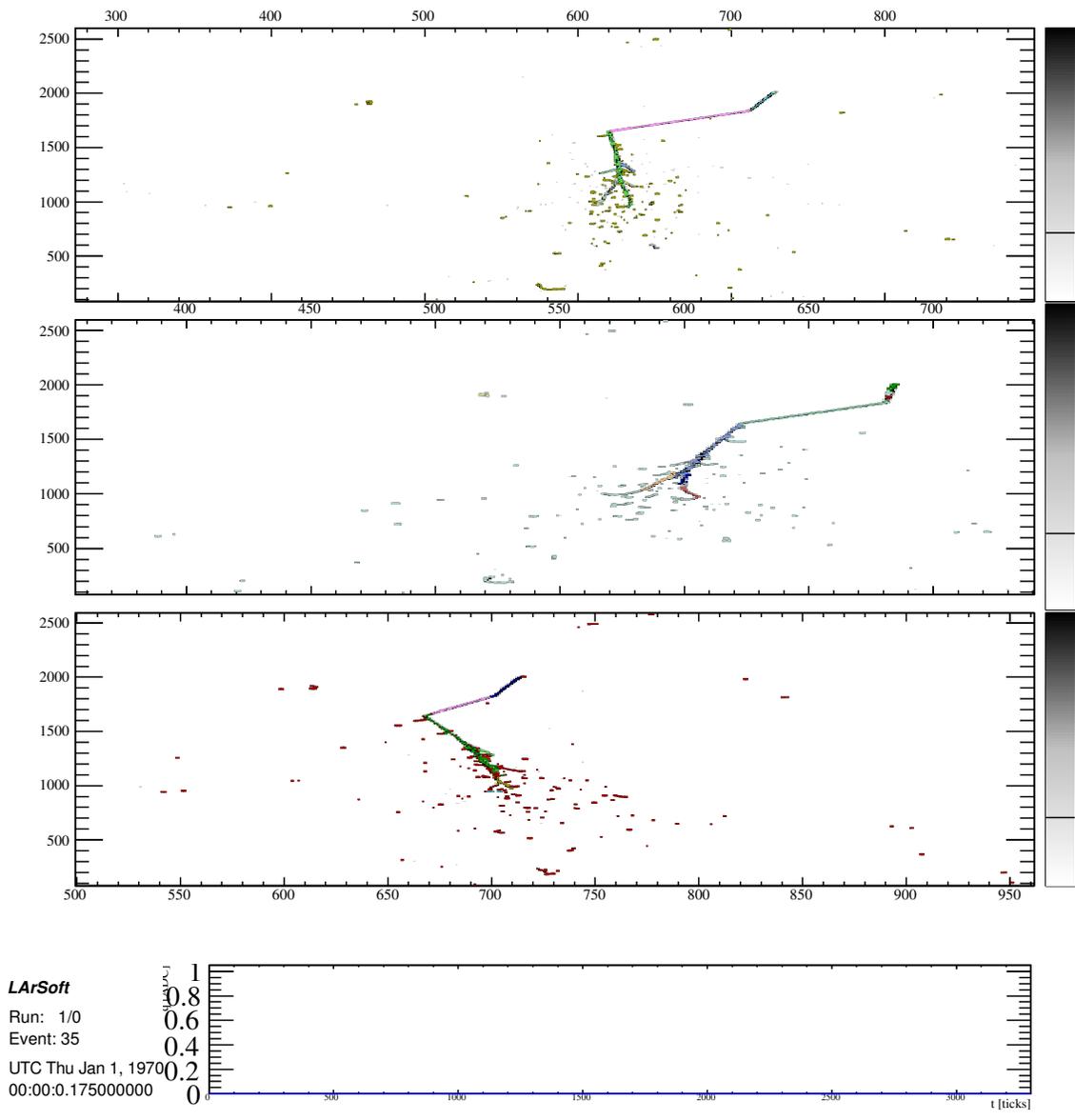


FIG. 8. The CCQE electron event after the PPHT line finder has been run.

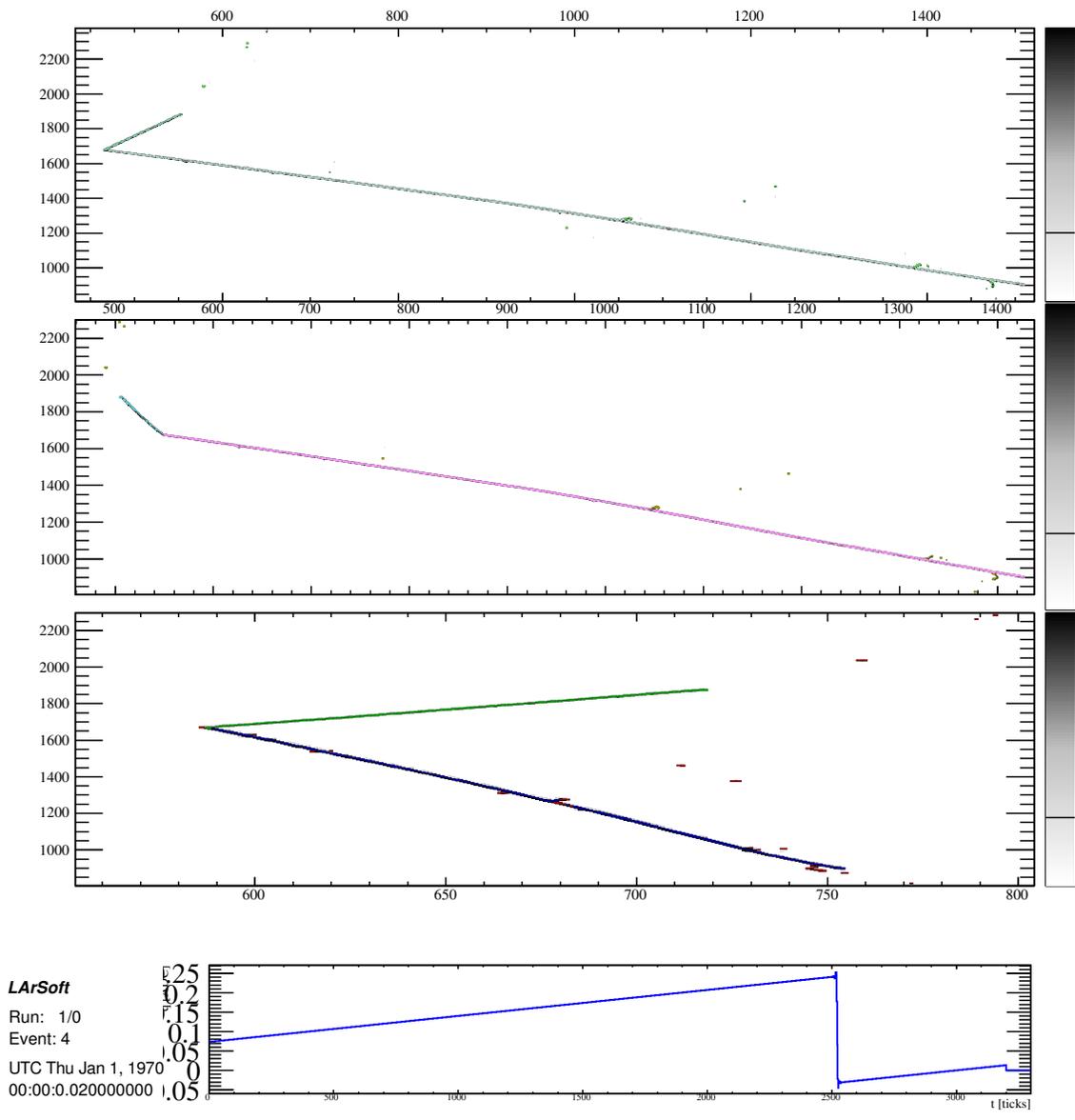


FIG. 9. The CCQE muon event after the PPHT lines have been merged.

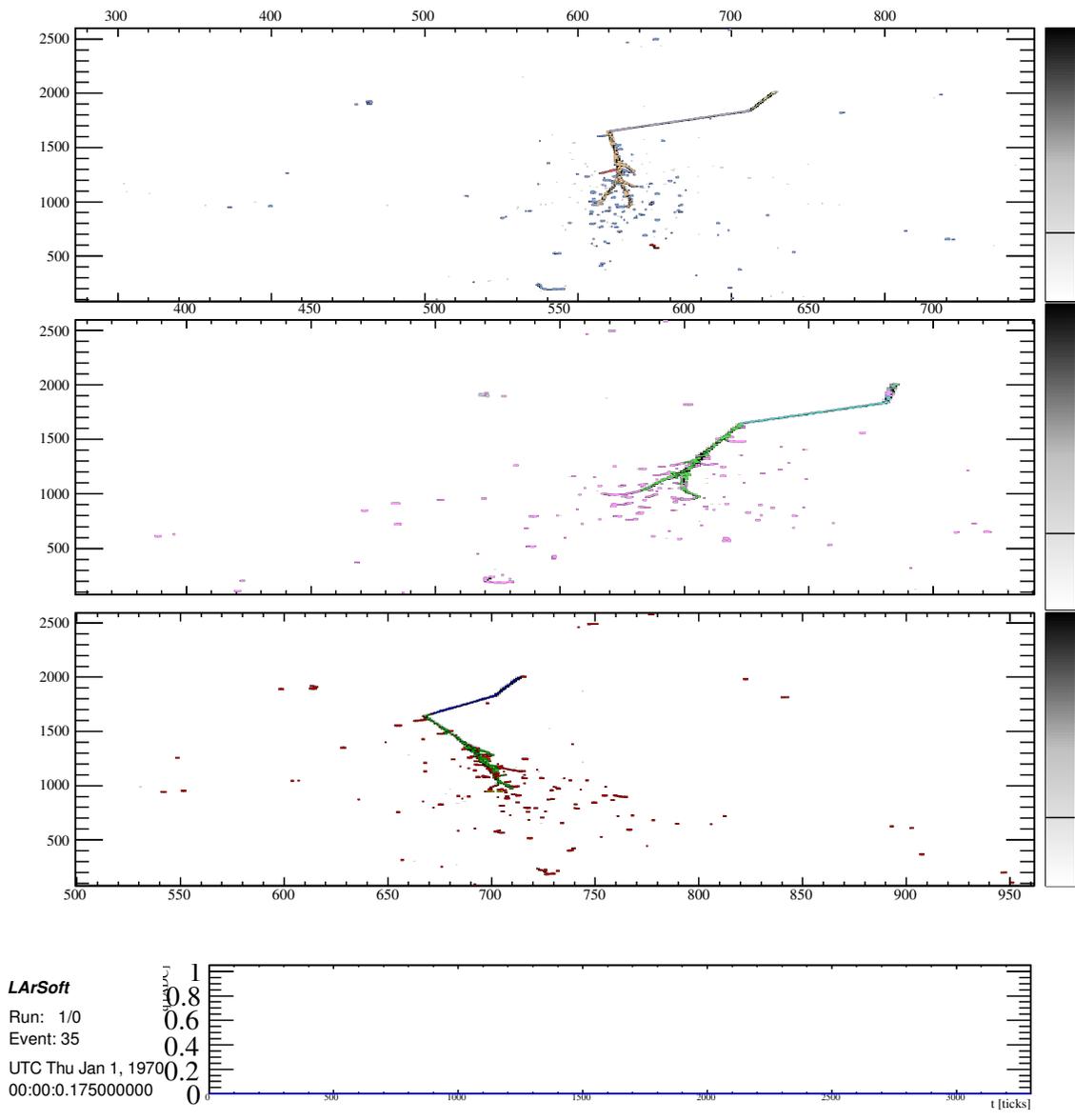


FIG. 10. The CCQE electron event after the PPHT lines have been merged.

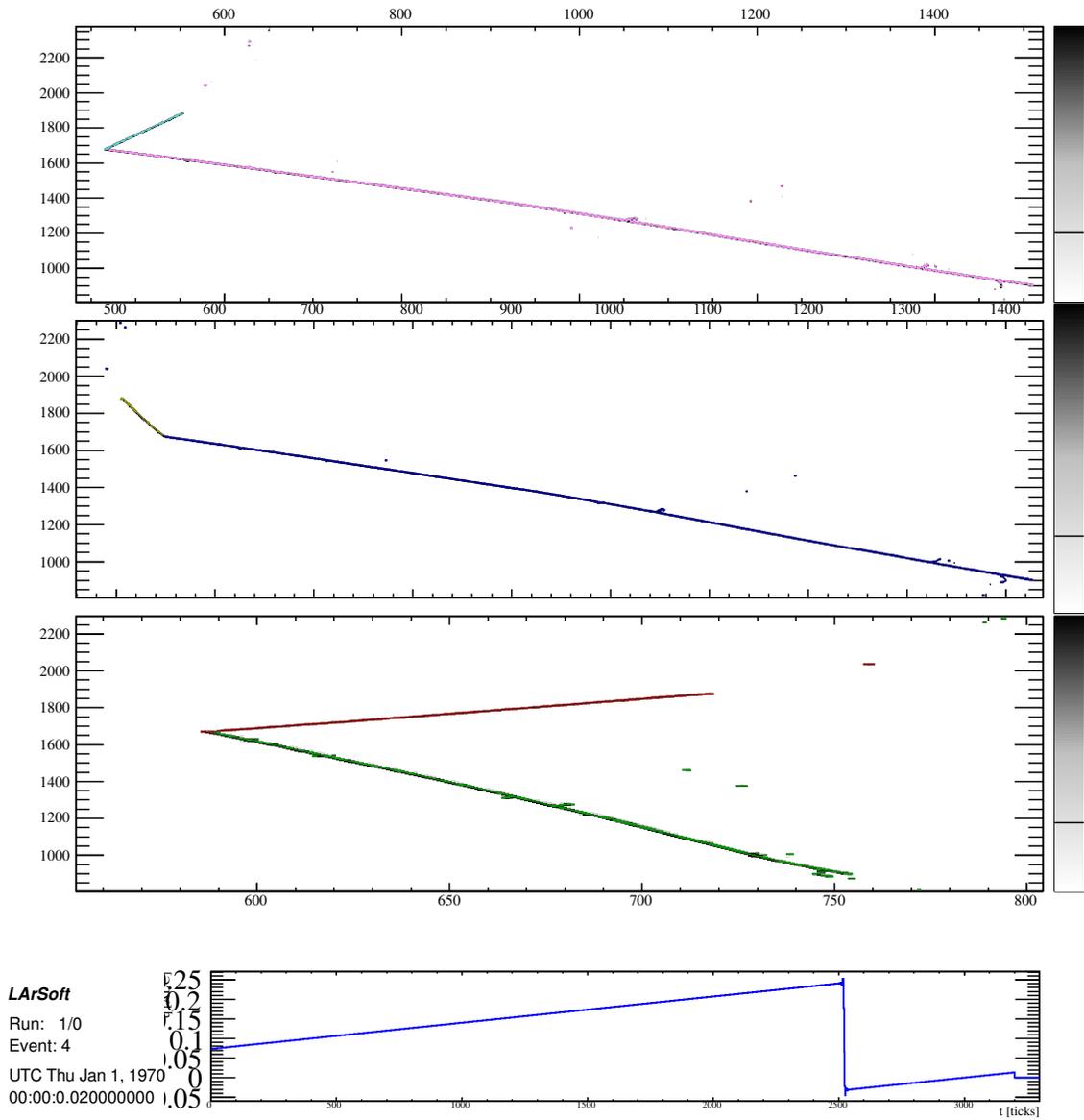


FIG. 11. The CCQE muon event after the fuzzy cluster remnants have been merged into a collection of merged lines.

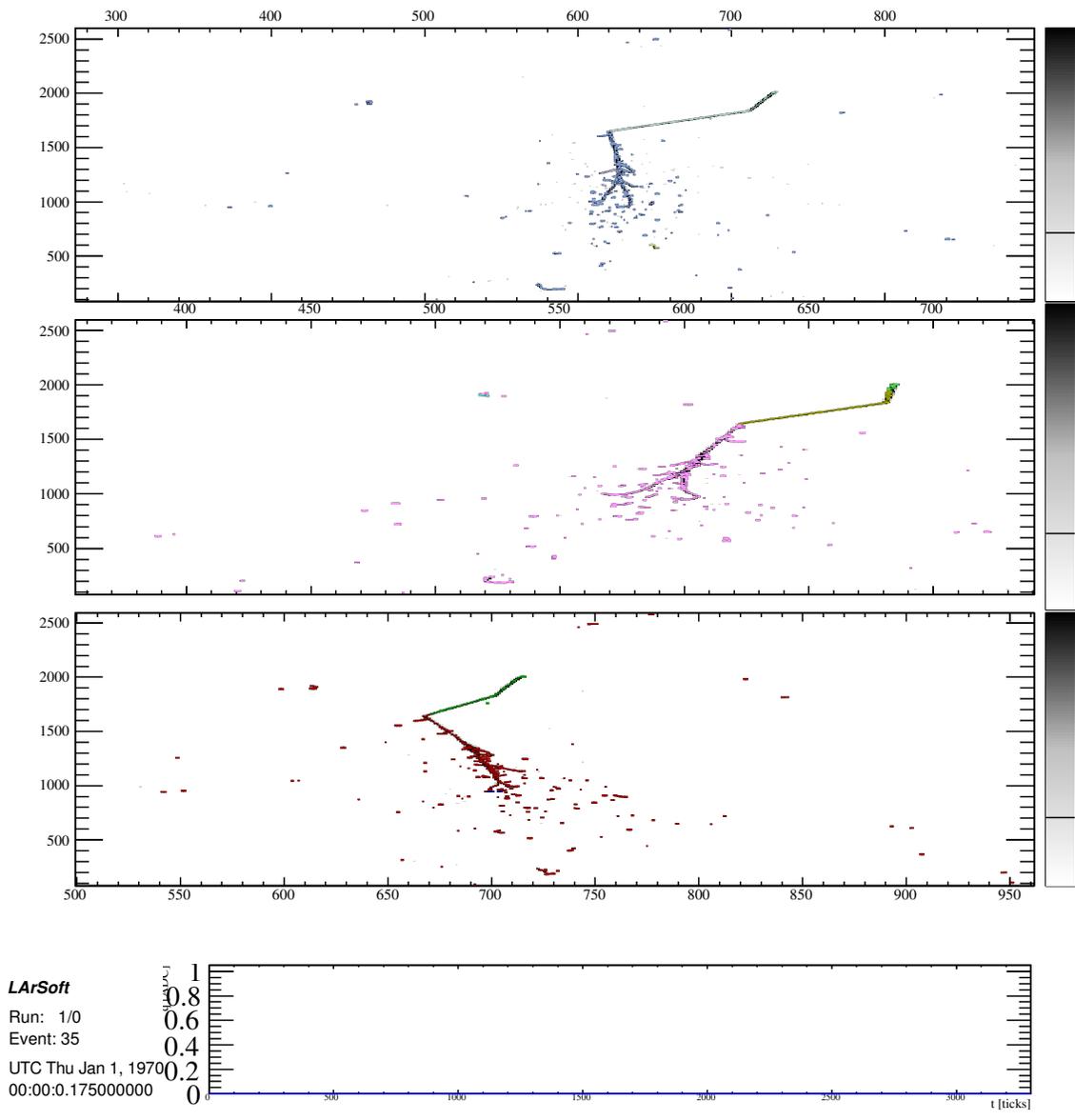


FIG. 12. The CCQE electron event after the fuzzy cluster remnants have been merged into a collection of merged lines.