GArSoft ML Update

Thomas Campbell

University of Colorado

May 1, 2019

Overview

How low can we push GArSoft's reconstruction thresholds?

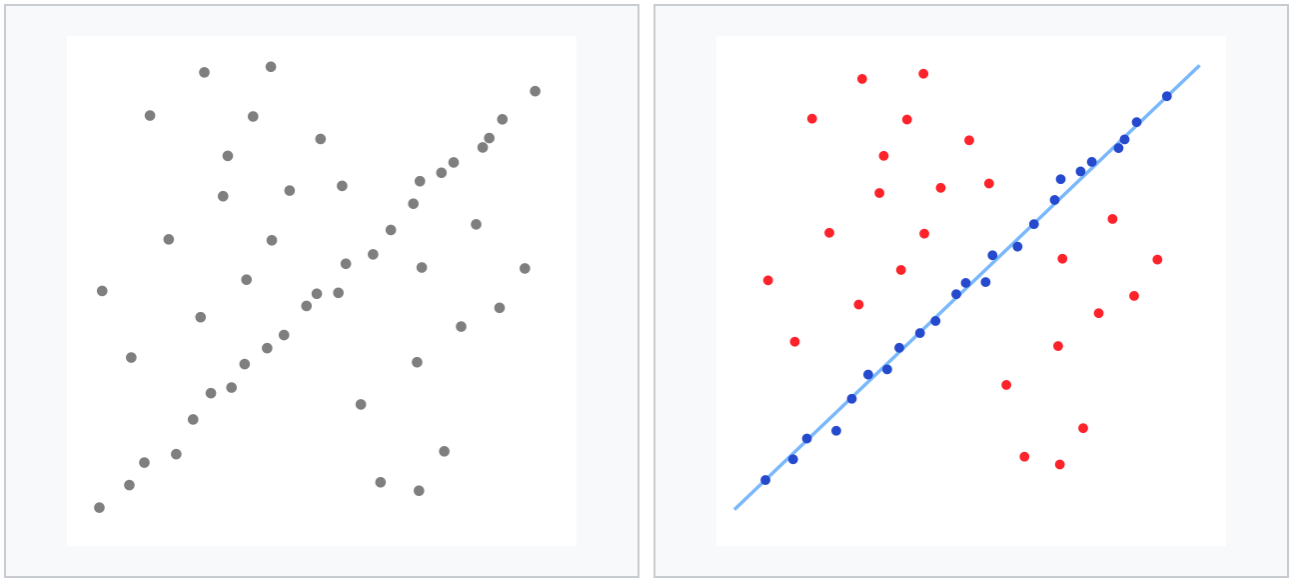
- Existing Reco does great for many particles across large portion of phase space
  - Complimentary ML approach to attempt to recover some of what might be missed

Automated ML based reconstruction on GArSoft leftovers

- Rely on existing GArSoft to catch all long tracks and identify vertex
- Take all unused TPC hits near vertex as input and:
  - RANSAC based clustering algorithm to assign hits to short track like groups
  - Feed these tracks into a neural net for PID, observables

RANSAC

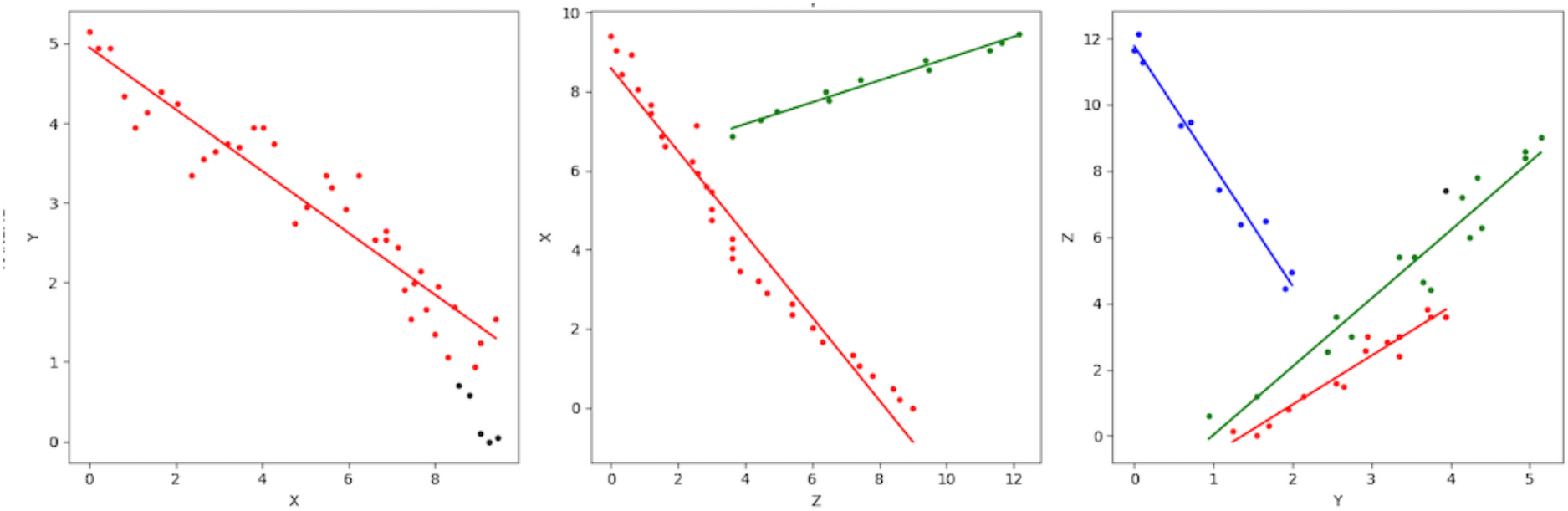
Random Sample Consensus



https://en.wikipedia.org/wiki/Random_sample_consensus

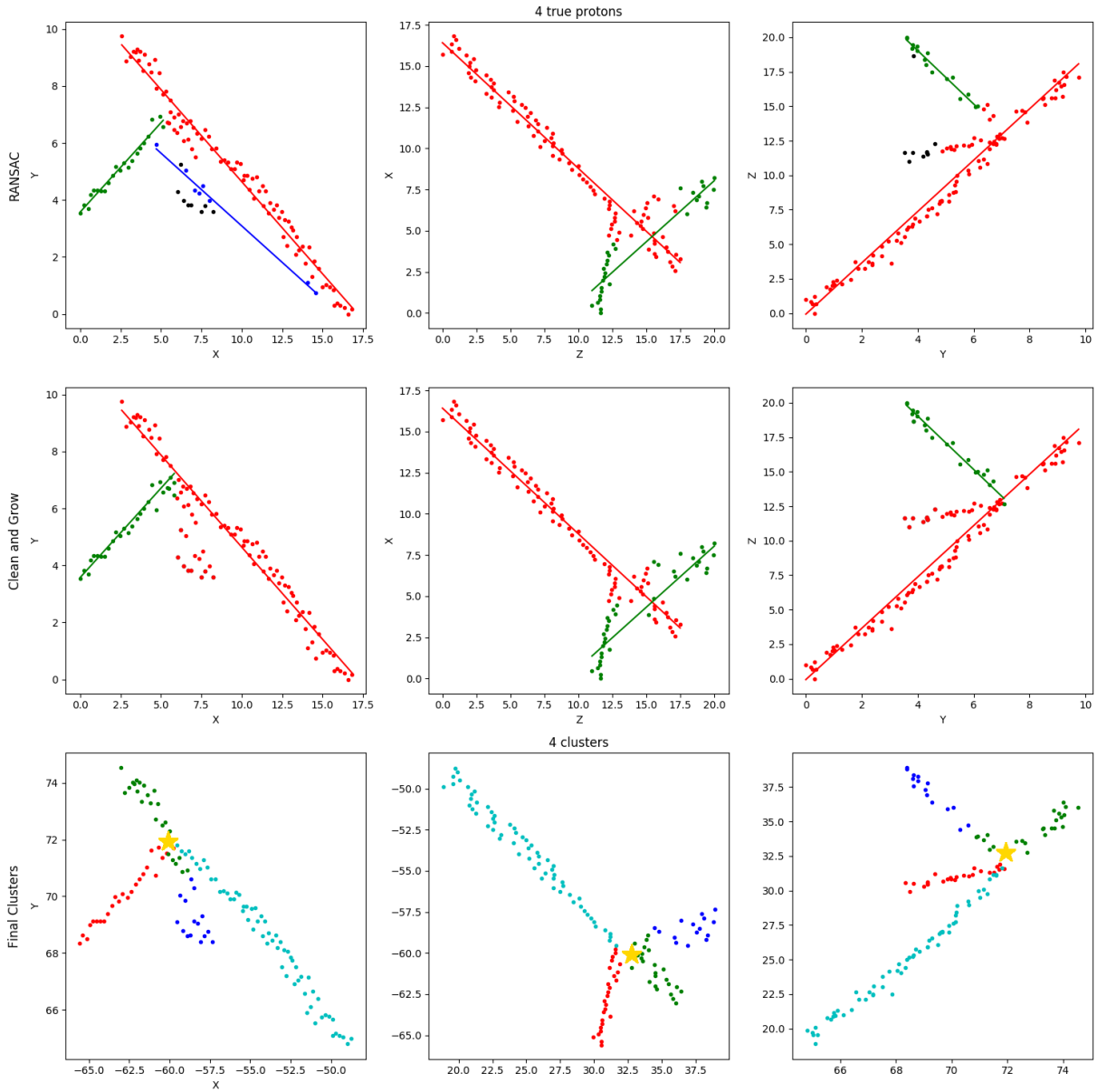
- iterative, voting based method for robust parameter estimation
  - randomly select minimal subset of data, fit
  - apply model to whole data set, consensus set are those within tolerance of model prediction
  - best model according some some score found over a fixed number of iterations is returned along with corresponding consensus set

Iteratively RANSACing

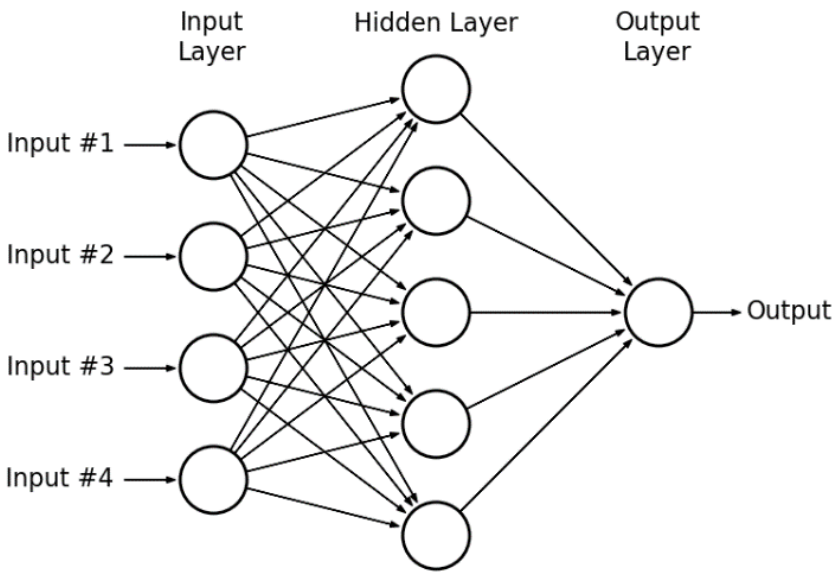


- Perform iterative RANSACs in each X-Y, Y-Z, Z-X plane (and in reciprocal planes) to get a collection of track cluster hypotheses, also do some cleaning
- Take all possible intersections across all planes, keep the highest quality ones and then grow clusters point-by-point with any remaining
- Test on set of events containing 1-4 protons with common vertex, momentum 100-150 MeV/c

RANSAC Example



Neural Nets



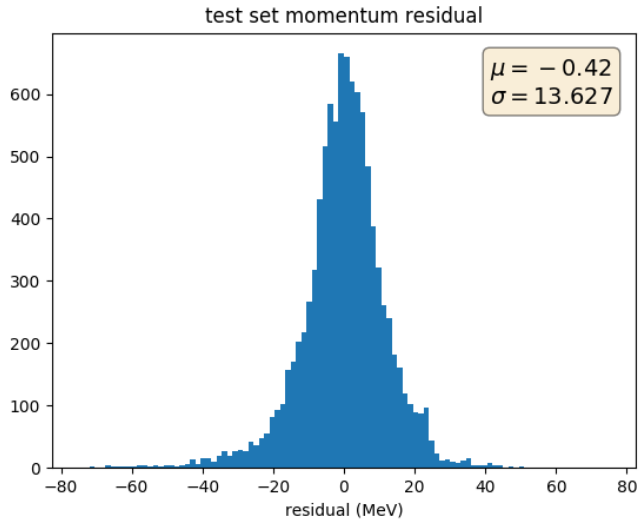
Fully Connected Multi-Layer Preceptron (MLP)

- Input features are relative hit positions and integrated charge collected

- Trained and validated on individual particle gun samples

Success so far in:

- Proton/Pion classification
- Kinetic energy regression



Test Set

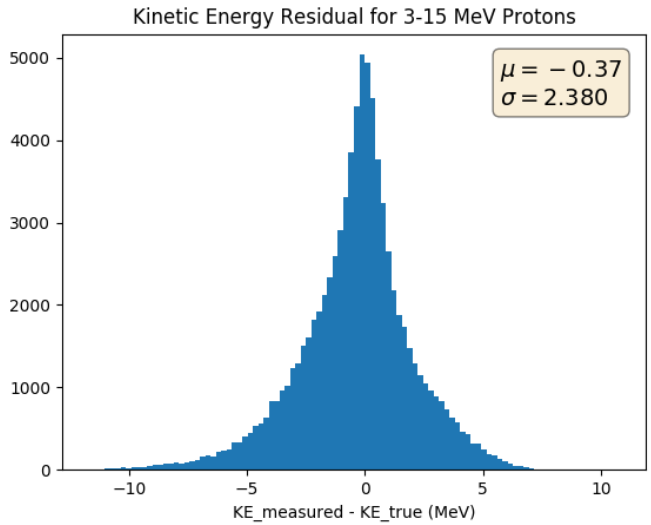
Sample of multi-proton events. Each event has:

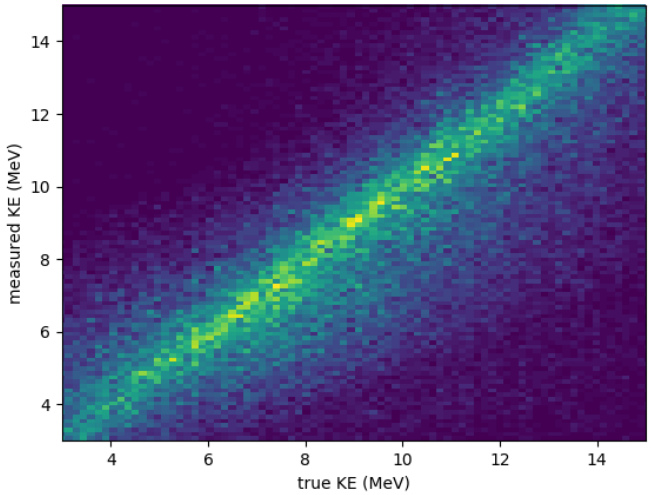
- 0-4 protons, number determined randomly with equal probabilities
- all protons share a common vertex at a random location in the TPC
- each proton is assigned randomly and independently:
  - a direction in space (isotropically distributed)
  - a scalar momentum between 0 and 200 MeV/c (flatly distributed)

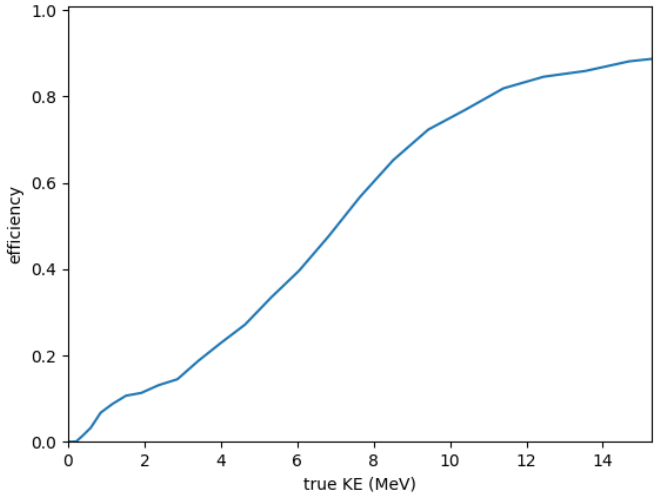
Process:

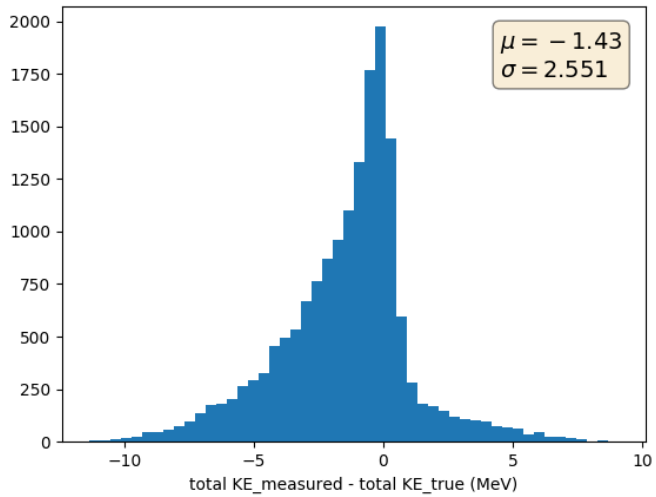
- GArSoft reconstructs event, outputs TPC hits
- TPC hits are assigned to proton candidate sets using RANSAC based algorithm
- each proton candidate set is passed to a neural net trained on single proton events to predict KE

Results


Kinetic Energy Residual for 3-15 MeV Protons
$\mu = -0.37$
$\sigma = 2.380$
KE_measured - KE_true (MeV)


measured KE (MeV)
true KE (MeV)


efficiency
true KE (MeV)


$\mu = -1.43$
$\sigma = 2.551$
total KE_measured - total KE_true (MeV)

Summary

- Machine learning effort in GArSoft underway
- Promise for several use cases identified
- Aside from models shown today, framework (scikit-learn based) enables rapid prototyping, evaluation, and comparison of a vast array of ML methods
- Fully automated reconstruction of KE for multi-proton events has good preliminary performance

Next Steps:

- Reconstruct vector momentum
- Implement PID NN
- Clean interface with GArSoft
- Evaluate existing RANSAC+NN performance on GENIE events

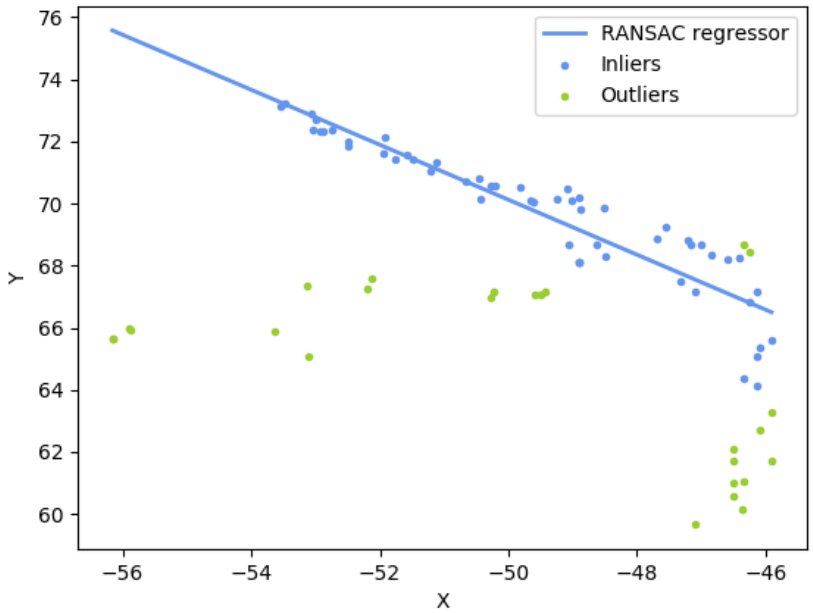
GArSoft ML Update

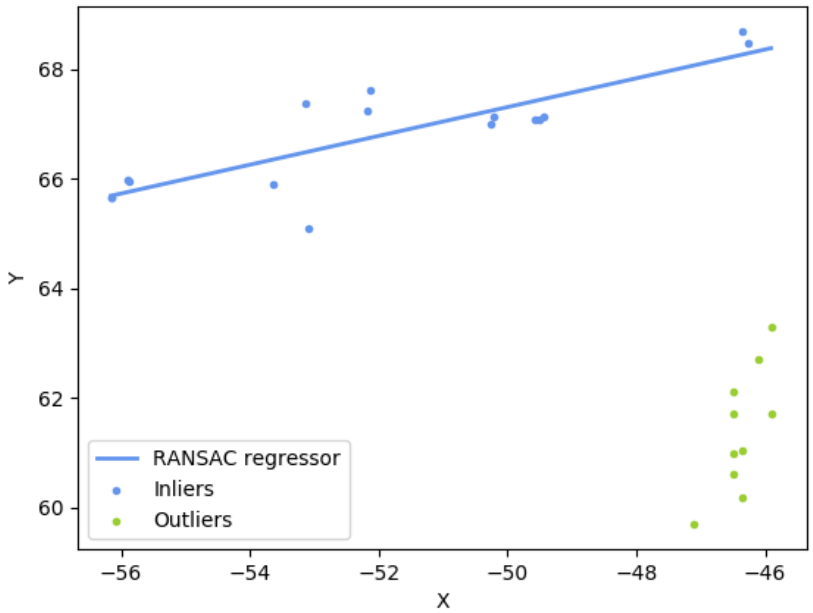
Thomas Campbell

University of Colorado

May 1, 2019

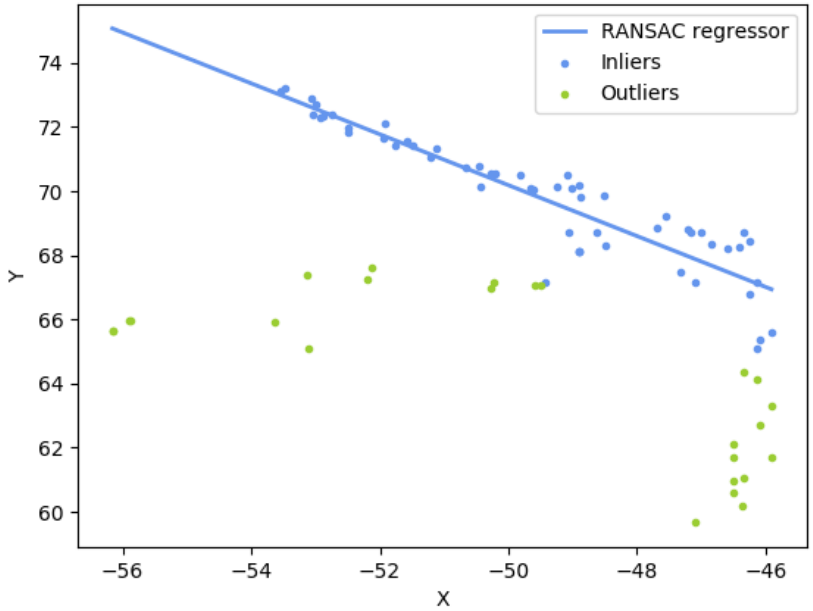
Iteratively RANSACing

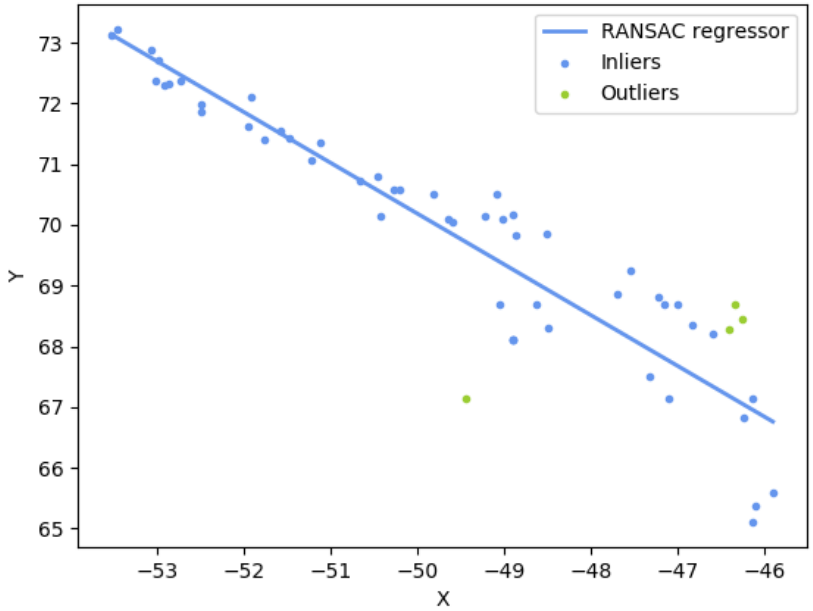




- each application of RANSAC returns a line, inliers, outliers
- apply RANSAC again to
  - **outliers: look for more lines**
  - inliers: look for lines inside lines, refine inlier decision boundary

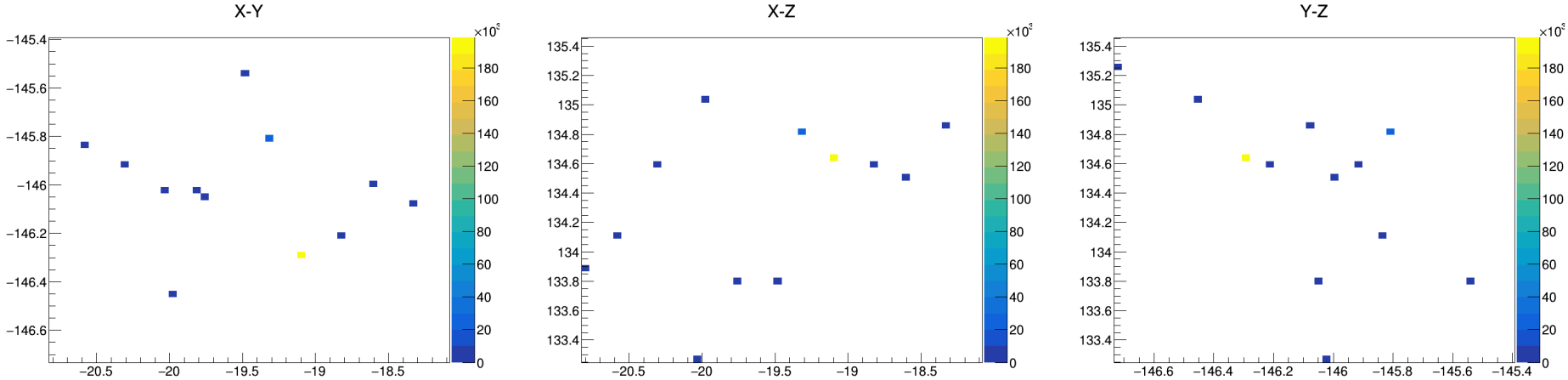
Iteratively RANSACing





- each application of RANSAC returns a line, inliers, outliers
- apply RANSAC again to
  - outliers: look for more lines
  - **inliers: look for lines inside lines, refine inlier decision boundary**

Data



Feature Engineering:

- consider only events with 250 or less hit-clusters
  - effectively all protons in this data set captured
- feature vector is hit positions and integrated signal for all hits relative (position only) to the hit with the minimum x position, sorted by x position
- $[0, 0, 0, s_0, (x_1 - x_0), (y_1 - y_0), (z_1 - z_0), s_1, ...]$
- seems a sensible first choice to me, *lots* of work could be done here experimenting with alternatives