Methods for Data Cleaning

Jonathan Edelen, Auralee Edelen, & Dean Edstrom

radiasoft SLAC NATIONAL ACCELERATOR LABORATORY Fermilab

presented at the 2nd ICFA Beam Dynamics Mini-Workshop: Machine Learning Applications for Particle Accelerators, PSI

(Simulations and Modeling)

28 February 2019

Motivation

- What is data-cleaning?
  - *Removing un-wanted or erroneous data from large training datasets*
  - *Identifying inconstancies in across large datasets that span different runs*

Motivation

- What is data-cleaning?
  - *Removing un-wanted or erroneous data from large training datasets*
  - *Identifying inconstancies in across large datasets that span different runs*
- Why do we want to clean our data? Errors in training datasets can
  - *propagate into models*
  - *increase model complexity*
  - *slow down the learning process*

Motivation

- What is data-cleaning?
  - *Removing un-wanted or erroneous data from large training datasets*
  - *Identifying inconstancies in across large datasets that span different runs*
- Why do we want to clean our data? Errors in training datasets can
  - *propagate into models*
  - *increase model complexity*
  - *slow down the learning process*
- Where do these errors come from?
  - *Simulation runs terminate unexpectedly*
  - *Machine calibration errors*
  - *Uncharacterized drift*

Motivation

- What is data-cleaning?
  - *Removing un-wanted or erroneous data from large training datasets*
  - *Identifying inconstancies in across large datasets that span different runs*
- Why do we want to clean our data? Errors in training datasets can
  - *propagate into models*
  - *increase model complexity*
  - *slow down the learning process*
- Where do these errors come from?
  - *Simulation runs terminate unexpectedly*
  - *Machine calibration errors*
  - *Uncharacterized drift*
- Why automate this process?
  - *Manual data cleaning is time consuming!*

Objective

- Explore the use of unsupervised learning for automatic data cleaning using case studies:
  - *Start simple*
    - Batch simulation scans of the FAST LINAC
  - *Increase complexity:*
    - Classification of machine drift at FAST
    - Multi-slit emittance measurements
    - Temperature-frequency data from a high power RFQ

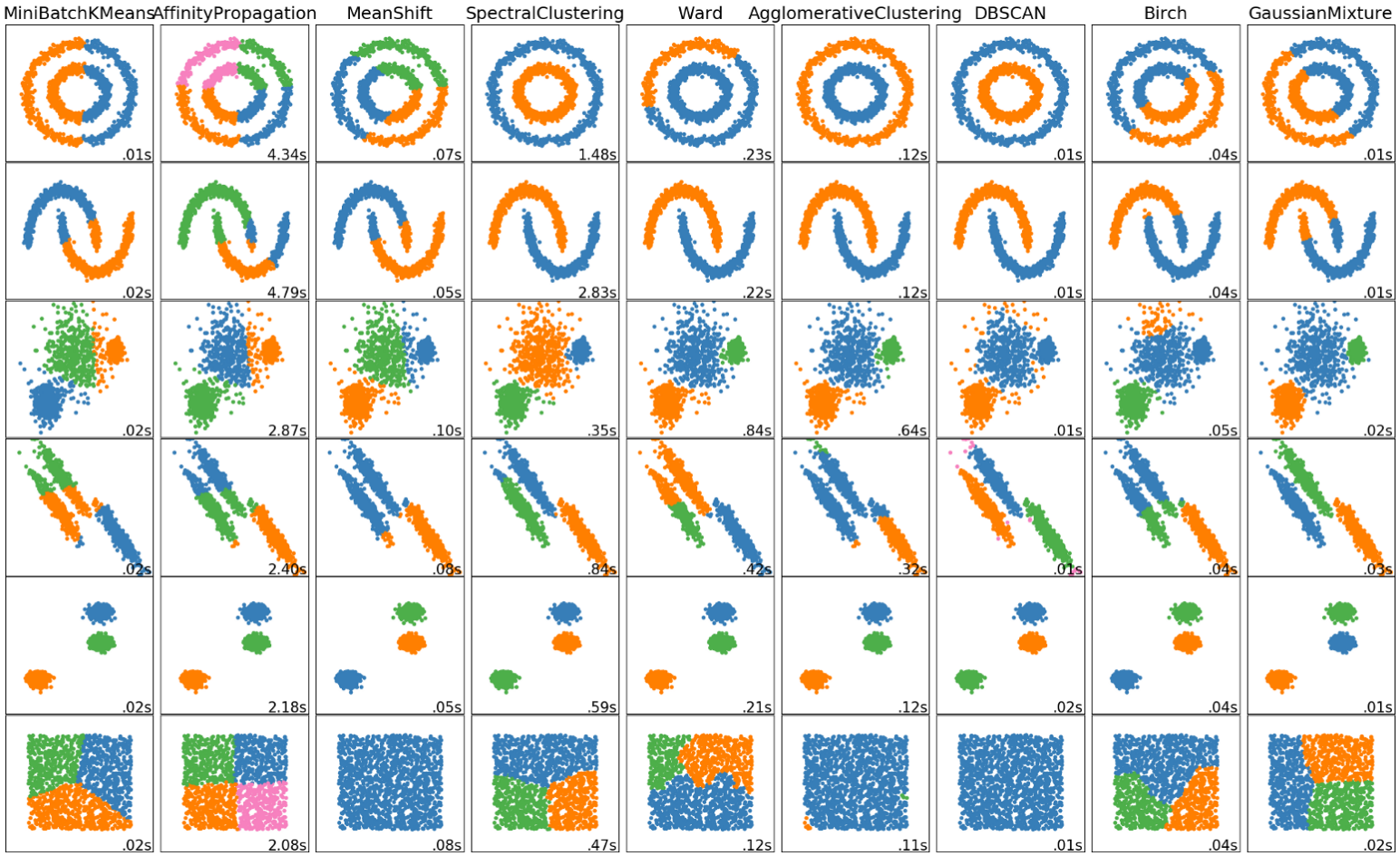
Overview of methods used

- Unsupervised learning
  - *DB-Scan*
    - Well suited for clusters of uniform density and odd shape
  - *Gaussian Mixture Modeling*
    - Well suited for clusters with a Gaussian distribution
  - *K-means*
    - Well suited for clusters that are uniformly distributed from a center
  - *Agglomerative Clustering*
    - Aggregating tiny clusters rather than dividing large clusters

Overview of methods used

- Unsupervised learning
  - *DB-Scan*
    - Well suited for clusters of uniform density and odd shape
  - *Gaussian Mixture Modeling*
    - Well suited for clusters with a Gaussian distribution
  - *K-means*
    - Well suited for clusters that are uniformly distributed from a center
  - *Agglomerative Clustering*
    - Aggregating tiny clusters rather than dividing large clusters
- Physics based clustering
  - *Smoothness*
    - Continuity
    - First order smoothness
    - Etc.

Clustering resource aside



https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68

Smoothness tests

- Principle:
  - *Parameter scans in simulation should produce smooth functions for bulk parameters (for the FAST LINAC this is the case)*

Smoothness tests

- Principle:
  - *Parameter scans in simulation should produce smooth functions for bulk parameters (for the FAST LINAC this is the case)*

- How do we determine if a discrete dataset is continuous?
  - *Define a metric for the data: There exists some i and j such that this condition is satisfied* $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < m$
  - *Violations are discontinuities*

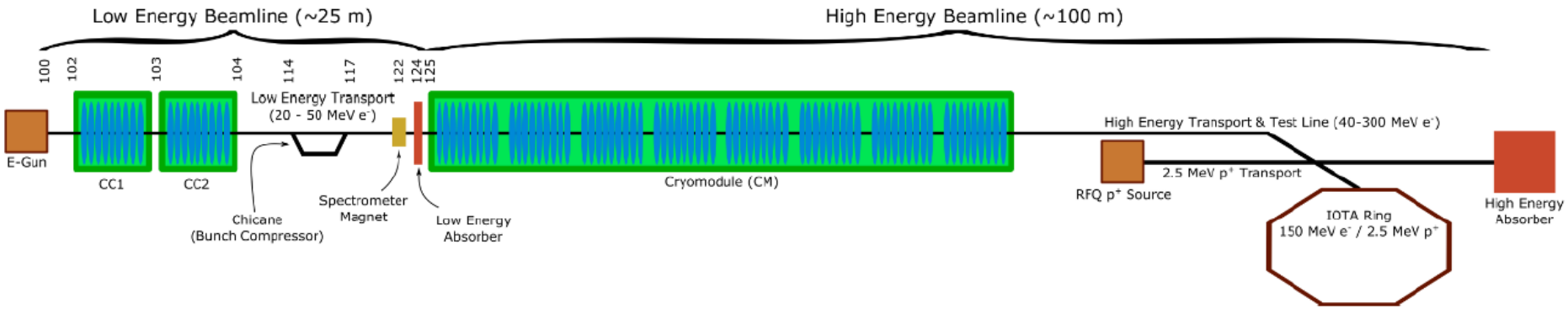
Smoothness tests

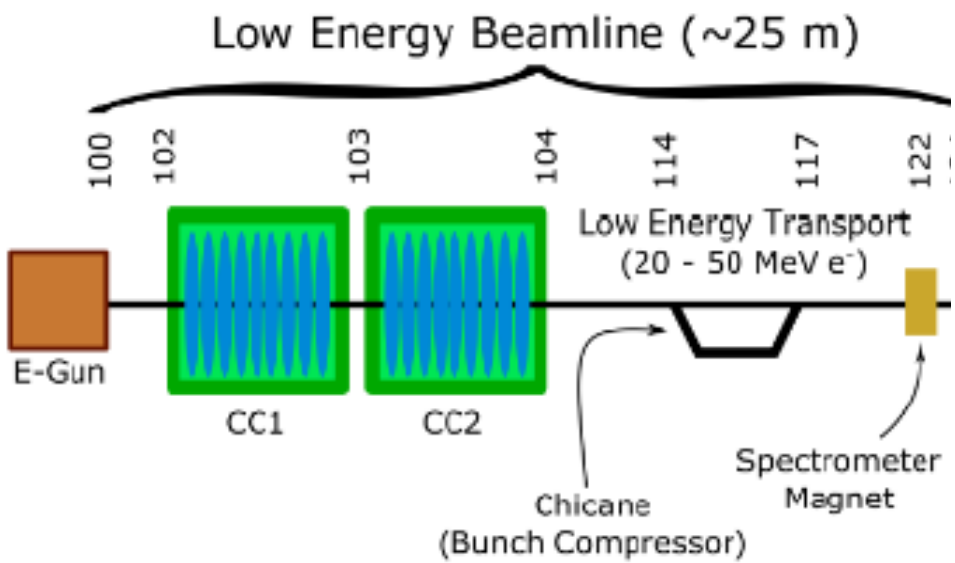
- Principle:
  - *Parameter scans in simulation should produce smooth functions for bulk parameters (for the FAST LINAC this is the case)*

- How do we determine if a discrete dataset is continuous?
  - *Define a metric for the data: There exists some i and j such that this condition is satisfied* $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < m$
  - *Violations are discontinuities*

- What about smoothness?
  - *Compute derivatives and use above criteria to evaluate if derivatives are continuous*

Overview of the FAST Linac

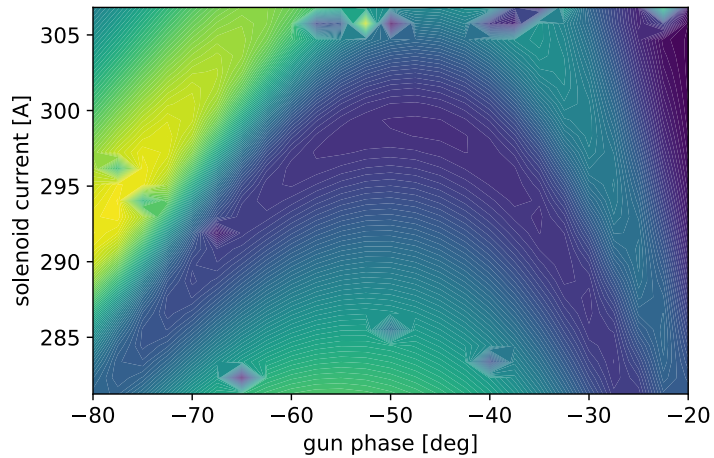


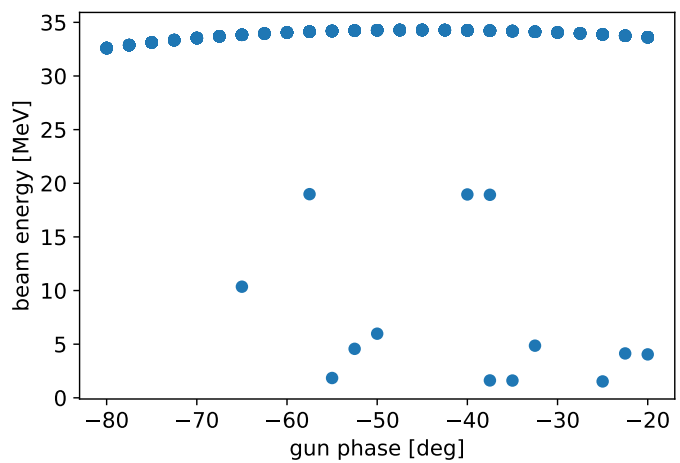


- 1.3 GHz photocathode RF gun
  - *PITZ style gun with solenoid and bucking coil*
  - *Beam accelerated to ~4 MeV*
- 1.3 GHz 9-cell Tesla type cavities
  - *Beam accelerated to ~35 MeV*

Simulation data from the FAST LINAC

- 2-D scan of gun-phase and solenoid strength
  - *Run on high performance computing, (Linux cluster with 100 cores)*
  - *Some simulations terminated unexpectedly*
  - *Remove unwanted data from dataset*
- Energy is the cleanest indicator of good vs. bad
  - *Use this to label dataset but exclude from clustering analysis*

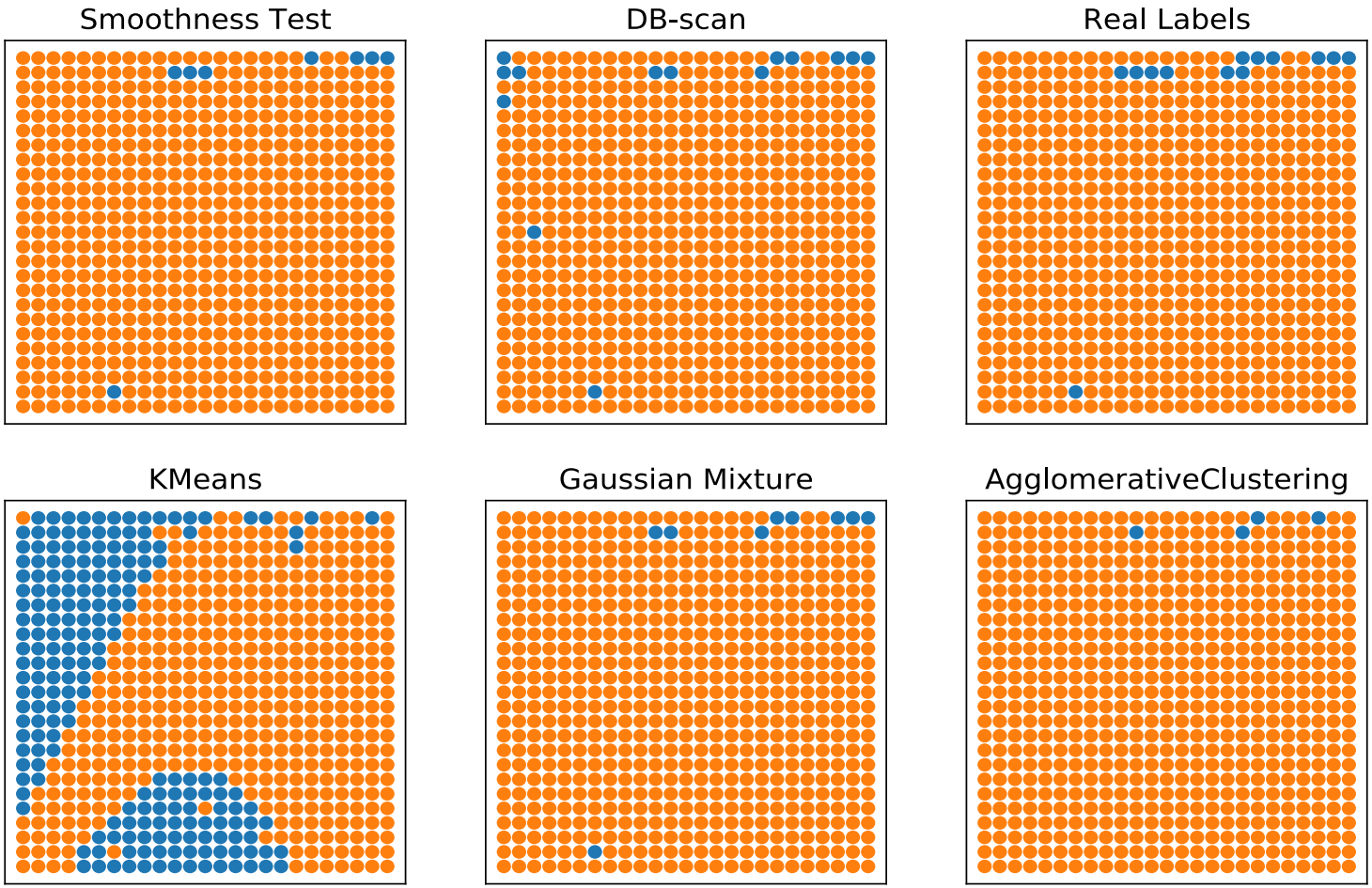




Choosing hyper-parameters

- Training procedures:
  - *DB-scan*
    - minimize the distance and number of points while keeping only two clusters
  - *Gaussian Mixture Modeling*
    - choose 2 clusters to start
  - *K-means*
    - choose 2 clusters to start
  - *Agglomerative Clustering*
    - choose 2 clusters to start
  - *Continuity clustering*
    - scan the metric and choose knee in curve of number of discontinuities vs metric size

Initial results



Orange: Identified good run
Blue: Identified bad run

Initial results

	K-means	DB-Scan	Gaussian Mix	Agglo	Smoothness
Percentage Correct	67.2%	98.6%	99.4%	98.6%	99.2%
Correctly Identified Bad Runs	3/13	9/13	9/13	4/13	8/13
False Positive	10/13	4/13	4/13	9/13	5/13
False Negative	195/612	5/612	0/612	0/612	0/612

False Positive: Predicted to be good but are actually bad. False Negative: Predicted to be bad but are actually good

- DB-Scan/Gaussian Mixture/Smoothness have similar performance
- K-means and Agglomerative are both poor performers
- Gaussian mixture is a very good option for this dataset as specification of hyper-parameters is easiest and zero false positives

Work in progress: Identifying machine drift

- Can we identify drift in RF calibrations?
  - *Using Energy Measurements*
  - *Using BPM Data*

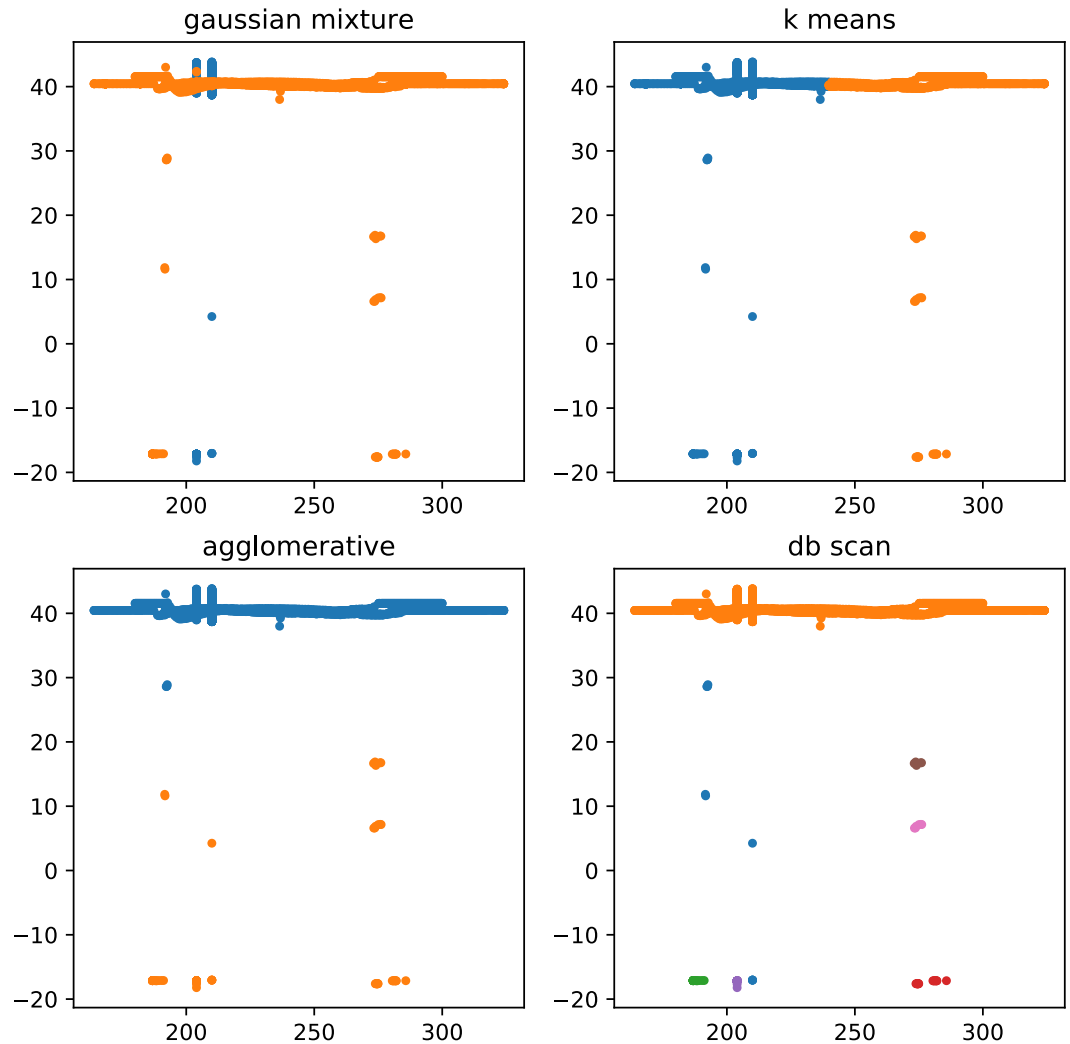
- Case study:
  - *FAST emittance measurement studies*
  - *Data collected during 3 separate studies spanning a 4 month period*
    - November 2018, Dec 2018, and Feb 2019

- Using different clustering algorithms
  - *Apply clustering to remove bad data*
  - *Apply clustering to identify calibration drifts*

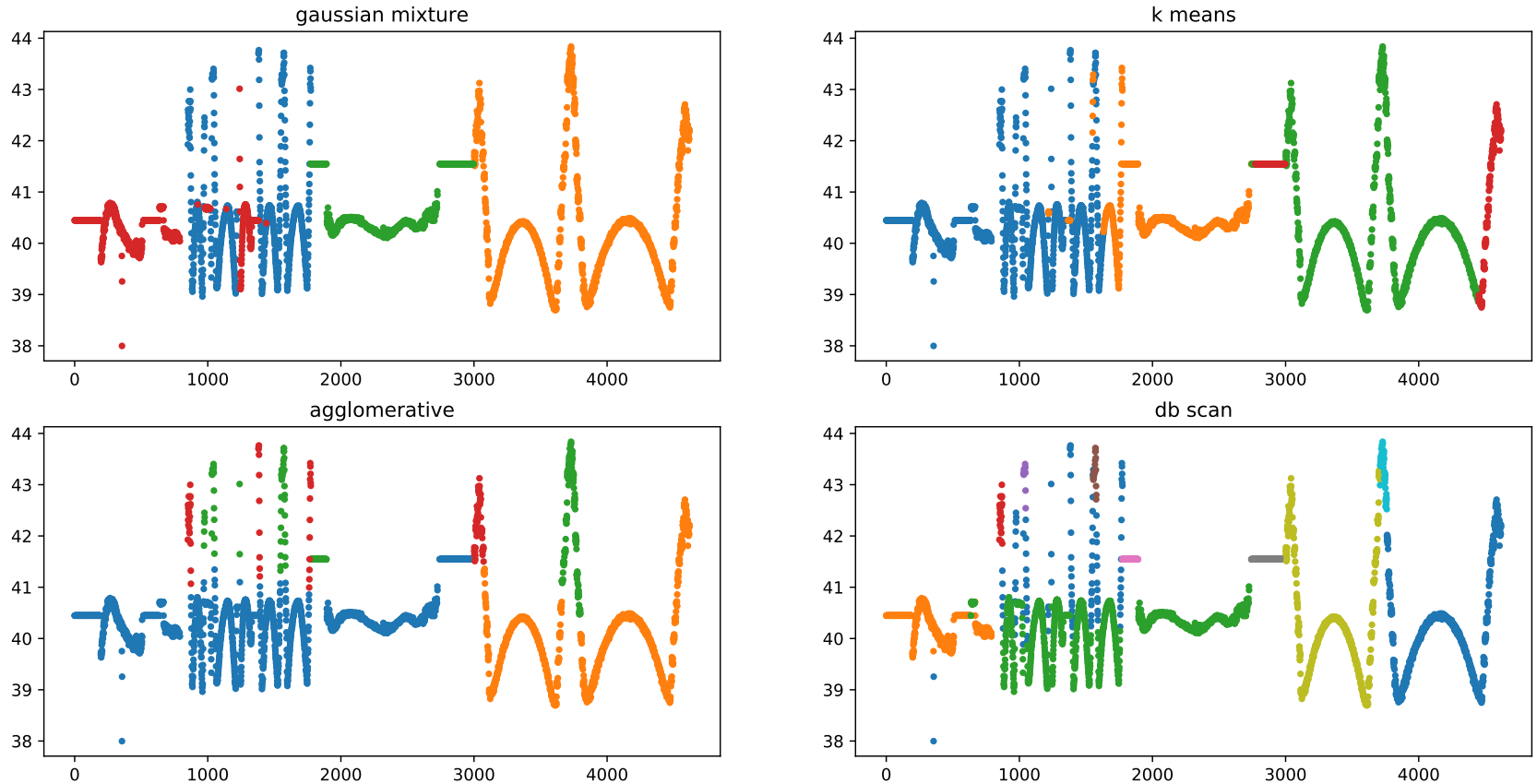
Initial data cleaning

- Four different clustering methods applied
  - *K-means was the only one that really failed*
  - *Both Gaussian mixture and agglomerative have some sub-optimal behavior*
  - *DB Scan is the most "correct"*

- Horizontal axis: gun phase
- Vertical axis: beam energy



Identifying RF Calibration Drift



- We know there was drift from early in the run to the end of the run. The question we want to answer is, is it possible for a clustering algorithm to detect this drift automatically.

Conclusions and Next Steps

- Conclusions:
  - *DB-Scan is relatively effective for cleaning data and has good hyper parameter tuning heuristics*
  - *Agglomerative methods are effective for large outliers in machine data without hyper parameter tuning*

- Future efforts:
  - *Continue to work on more complicated machine and simulation data*
  - *Attempt to generalize procedure for automated data cleaning*
  - *Explore other clustering algorithms or anomaly detection methods*