Clustering Analysis of Fast Ion Driven Instabilities

John Gresl[1]
William Heidbrink[1]
Shaun Haskey[2]
Boyd Blackwell[3]

[1]*Department of Physics and Astronomy, University of California Irvine, California 92612, USA*
[2]*Princeton Plasma Physics Lab, Princeton University, New Jersey 08540, USA*
[3]*Plasma Research Laboratory, Research School of Physics and Engineering, The Australian National University, Canberra, ACT 0200, Australia*

Submitted: June 16, 2017

Abstract

Beam ions often drive Alfvén eigenmodes and other instabilities unstable in DIII-D. Many of these modes have been unambiguously identified but some frequently occurring features have been neglected. In this work, datamining and analysis techniques[1] that successfully analyzed magnetics data from the H-1NF Heliac are applied to arrays of magnetic and electron cyclotron emission (ECE) data from DIII-D. The clustering techniques group toroidal instabilities with similar toroidal magnetic features into clusters of identical mode numbers. Similar analysis is performed on DIII-D's poloidal magnetic array however the varying distance between the poloidal probes and the plasma led to inconsistent signals and no appreciable clusters were found.

Introduction

Plasma is one of the four fundamental states of matter and makes up an estimated 99% of the matter in the observable universe. A plasma can be thought of as a quasineutral medium of unbound positively and negatively charged particles. These moving charged particles generate local electric and magnetic fields which affect the motion of other nearby particles leading to *collective behavior*, or motion that depends on the physical state of the plasma in local regions.

Harnessing nuclear fusion is one of the prime motivators of studying plasma physics. Nuclear fusion produces more energy per amount of fuel than any other available fuel source. For instance, one gallon of heavy water (water with all of the hydrogen atoms replaced with deuterium atoms) provides 100,000 times more energy when fused than a gallon of gasoline. Deuterium-tritium (D-T) plasmas are known as the most efficient plasma for energy production due to their high mass-to-charge ratio, making it easier to overcome coulomb repulsion and fuse together. There are four main reactions that occur in D-T plasmas.

$$D + D \to (0.82\,MeV)\ {}^3He + (2.45\,MeV)\ n$$

$$D + D \to (1.01\,MeV)\ T + (3.02\,MeV)\ p$$

$$D + {}^3He \to (3.71\,MeV)\ \alpha + (14.64\,MeV)\ p$$

$$D + T \to (3.56\,MeV)\ \alpha + (14.03\,MeV)\ n$$

Above, D is a deuterium ion, T is a tritium ion, p is a proton, n is a neutron, and α is a ^{2}He nucleus.

Many research plasmas require temperatures on the order of 10^8 K, making the plasma hot enough to destroy anything it comes into contact with. For this reason, strong magnetic fields are used to shape and contain the plasma. The introduction of these magnetic fields, while necessary, can lead to some undesirable effects such as unwanted resonances, instabilities, and particles escaping from the plasma and colliding with the inner walls. The study of these interactions is crucial for creating higher quality magnetically confined plasmas.

Background Physics

Cyclotron Motion

Charged particles trapped in magnetic fields exhibit circular orbits and is known as cyclotron motion. For a particle with charge, q, in a magnetic field, $\boldsymbol{B}$, and velocity $\boldsymbol{v}$, the magnetic force, $\boldsymbol{F_B}$ is equal to:

$$\boldsymbol{F_B} = q(\boldsymbol{v} \times \boldsymbol{B})$$

It is easy to show that there is no work done by this force.

$$\frac{dW}{dt} = \boldsymbol{F} \cdot \boldsymbol{v} = q(\boldsymbol{v} \times \boldsymbol{B}) \cdot \boldsymbol{v} = 0$$

Since $(\boldsymbol{v} \times \boldsymbol{B})$ is perpendicular to both $\boldsymbol{v}$ and $\boldsymbol{B}$, where W is work. The total energy of the particle does not change. We can rewrite velocity in terms of a new basis

$$\boldsymbol{v} = \boldsymbol{v}_\perp + \boldsymbol{v}_\parallel$$

where $v_\perp$ and $v_\parallel$ represent the components of velocity perpendicular and parallel to the magnetic field, respectively. We can then rewrite $\boldsymbol{F_B}$.

$$\boldsymbol{F_B} = q[(\boldsymbol{v}_\perp + \boldsymbol{v}_\parallel) \times \boldsymbol{B}]$$

$$\boldsymbol{F_B} = q[\boldsymbol{v}_\perp \times \boldsymbol{B} + \boldsymbol{v}_\parallel \times \boldsymbol{B}]$$

$$\boldsymbol{F_B} = q[\boldsymbol{v}_\perp \times \boldsymbol{B}]$$

We can say $\boldsymbol{v}_\parallel \times \boldsymbol{B}$ goes to zero since its magnitude, $|\boldsymbol{v}_\parallel \times \boldsymbol{B}|$, is equal to 0 since $\boldsymbol{v}_\parallel$ is parallel to $\boldsymbol{B}$ by definition.

This force causes charged particles to rotate in circles as they travel along magnetic field lines, all while keeping their parallel velocity constant (see figure 1). Equating the magnetic force to the centripetal force and solving for the radius of curvature yields an expression for the radius of curvature of this gyrating charge known as the Larmor radius, r_L.

$$q\boldsymbol{v}_\perp \boldsymbol{B} = m\frac{\boldsymbol{v}_\perp^2}{r_L} \rightarrow r_L = \frac{m\, v_\perp}{|q|B}$$

From the Larmor radius, one can also determine a parameter known as the cyclotron frequency, Ω_c:

$$\Omega_c = 2\pi f = \frac{\boldsymbol{v}_\perp}{r_L} = \frac{|q|\boldsymbol{B}}{m}$$

When electrons undergo cyclotron motion, they emit radiation in the form of photons. This is known as electron cyclotron emission (ECE) and is an important diagnostic when studying plasmas.

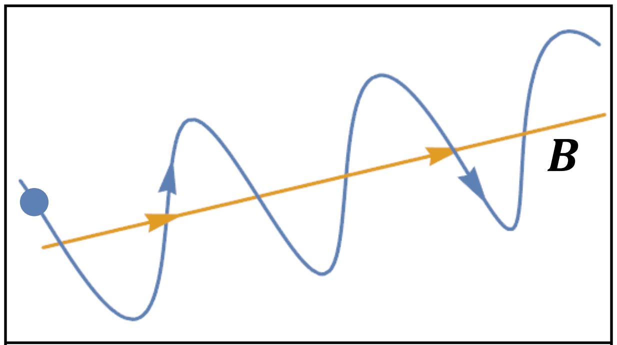

Figure 1. Graphical representation of the path of a charged particle (blue) travelling along a magnetic field line (yellow).

Alfvén Waves and Eigenmodes

Alfvén waves are low-frequency (relative to Ω_c) oscillations along a magnetic field line[2]. The motion of an Alfvén wave is analogous to a wave travelling along a stretched string. In this analogy, the tension in the magnetic field lines is the same as the tension in the string, and the Alfvén wave is the result of the “plucking” of the string.

From Maxwell’s equations and fluid equations for the plasma, we can derive the velocity at which these Alfvén waves travel to be $v_a \equiv B_0/\sqrt{\mu_0\ \rho}$. A derivation of the Alfvén velocity can be seen in Appendix A. The Alfvén velocity refers to the characteristic speed in which perturbations of the lines of the force travel.

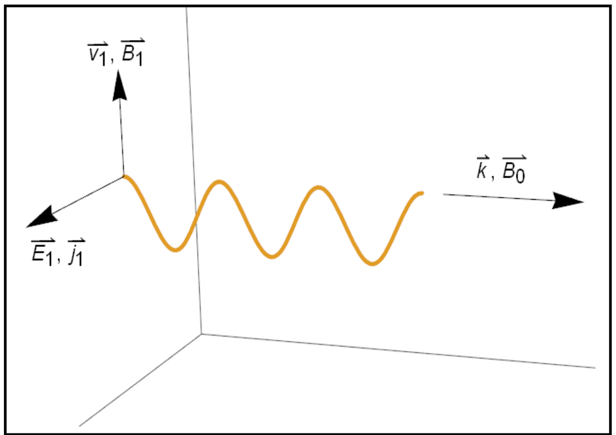

Figure 2. Geometry of an Alfvén wave. The wave has $\boldsymbol{k}$ along $\boldsymbol{B_0}$, $\boldsymbol{E_1}$ and $\boldsymbol{j_1}$ perpendicular to $\boldsymbol{B_0}$ and $\boldsymbol{B_1}$ and $\boldsymbol{v_1}$ perpendicular to both $\boldsymbol{B_0}$ and $\boldsymbol{E_1}$.

Particles within the plasma may interact undesirably with the external magnetic fields creating modes or local instabilities. Many of these instabilities are detrimental to the plasma’s health and can sometimes cause energetic particles to escape the plasma and damage expensive equipment on the inner walls. For our experiment, we are interested in the particles that resonate in the Alfvén waves and refer to them as Alfvén eigenmodes (AE’s).

The Device and Diagnostics

This experiment is done exclusively with data gathered from the DIII-D tokamak and diagnostics which have already been installed. DIII-D is a torus-shaped plasma device which confines plasma by using strong magnetic fields. The tokamak has a major radius of 1.67 m and a minor radius of 0.67 m. A coil of large electromagnets (B-coils) are positioned like rings around the tokamak to

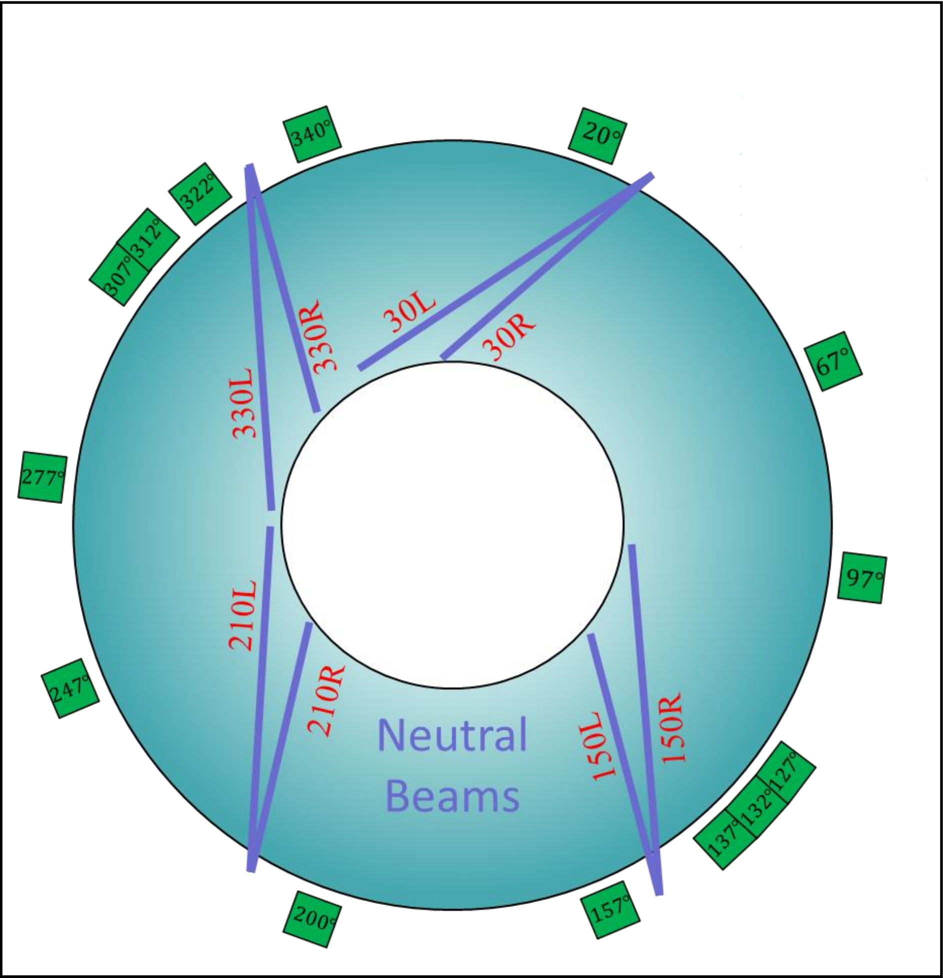

Figure 3. Cartoon of top down view of DIII-D. The 4-pairs of neutral beam injectors are shown in purple and the various toroidal magnetic Mirnov probes are shown in green with their angular position indicated.

produce toroidal magnetic fields on the order of 2 Tesla. Field shaping coils are also used to provide additional stability and confinement to the plasma.

DIII-D also utilizes 4 pairs of neutral beam injectors which function as an auxiliary energy source for the plasma. These neutral beams supply high energy neutral particles to the plasma where they can interact with other particles by collisions. One capability which sets the neutral beams at DIII-D apart from others is its ability to run their beams at a very high confidence over the range 40 kV to 75 kV *without* the need for special calorimetry or test shots[3].

Magnetic Mirnov Probes

Among the many diagnostics at DIII-D, the magnetic Mirnov probes are one of the most important. The probes consist of coils of wire which measure the changing magnetic field dB/dt caused by moving charges in the plasma. We are interested in the main toroidal magnetic Mirnov array and the 322-degree poloidal magnetic Mirnov array. We will refer to these arrays as the *toroidal array* and the *poloidal array*, respectively. The toroidal array consists of 14 probes and lies on the toroidal midplane of the tokamak. The arrangement of the 14 probes in the toroidal array can be seen in figure 3, along with the orientation of the neutral beam injectors. The poloidal array consists of 31 probes and encircles the tokamak poloidally at the 322-degree mark.

Major Radius, R_0	1.67 m
Minor Radius, a	0.67 m
Magnetic Field Strength, B_T	2.0 T
Plasma Current, I_P	1.0 MA
Electron Density, n_e	10^{20} m^{-3}
Core Electron Temperature, T_e	2 keV
Electron Debye Length, λ_{De}	10^{-4} m

Table 1. Typical plasma parameters at DIII-D.

Electron Cyclotron Emission (ECE)

The ECE array at DIII-D consists of a 40-channel heterodyne radiometer that provides the electron temperature of the plasma at a given radius and time. Fluctuations within the plasma can cause perturbations in the amplitude of the ECE which is proportional to T_e. Alfvén eigenmodes frequency affect the mode structure and can cause variations in T_e, which makes the ECE array a helpful tool for studying Alfvén eigenmodes.

Instabilities and Alfvén Eigenmodes in the DIII-D Tokamak

Instabilities can manifest within DIII-D's plasma in the form of resonances or Alfvén eigenmodes. These instabilities are studied using magnetohydrodynamics (MHD) and treating the plasma as a fluid. Alfvén eigenmodes can be very detrimental to confinement[4] and it has even been found that particles resonating in these Alfvén eigenmodes reduce the beam power[5], making it harder to achieve ignition. The study of these Alfvén eigenmodes and how to suppress their detrimental effects is of great importance for larger tokamaks in the future, such as ITER.

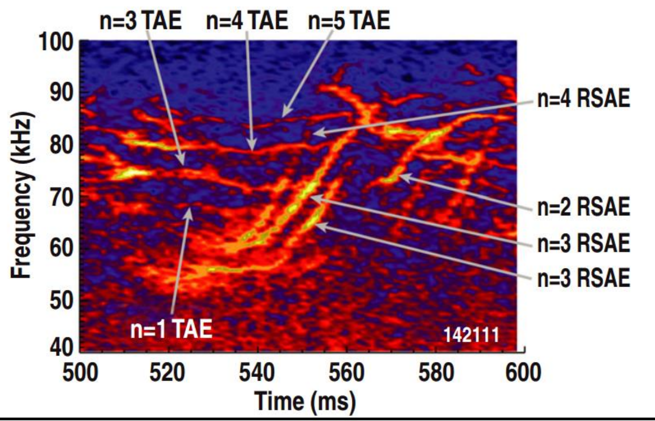

Figure 4. Frequency vs. time spectrogram for shot 142111 with different Alfvén eigenmodes identified. Frequency peaks in the fourier transform are shown in bright yellow. Figure taken from *Fast ion profile stiffness due to the resonance overlap of multiple Alfven Eigenmodes*[8].

The three Alfvén eigenmodes we are interested in are toroidal Alfvén eigenmodes (TAE's), reversed shear Alfvén eigenmodes (RSAE's), and Beta Induced Alfvén eigenmodes (BAE's). These different Alfvén eigenmodes are identified by analyzing a frequency vs. time spectrogram.

Identification of Toroidal Mode Number (n)

The toroidal mode number for a point in time-frequency space can be determined by tracking the phase difference of the signal between probes. For instance, let's say we are interested in the toroidal mode numbers for instabilities around f=75 kHz for shot 159243. First, we take the fourier transform of the raw probe signal and then identify frequency peaks by hand (figure 5).

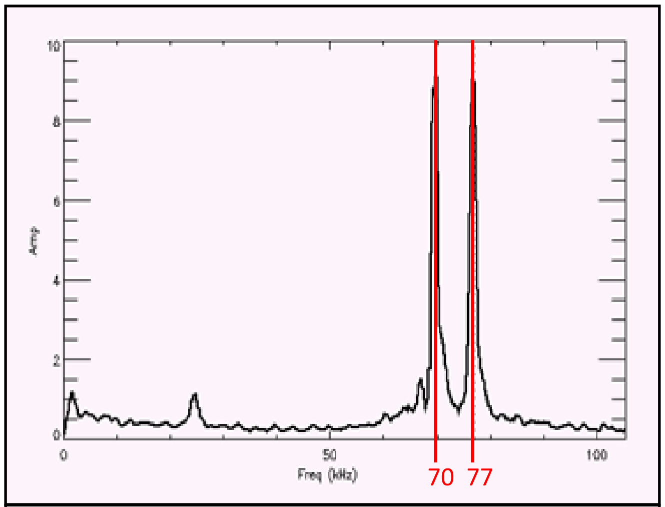

Figure 5. Fourier transform of shot 159243 at t = 450 ms. Frequency peaks shown in red.

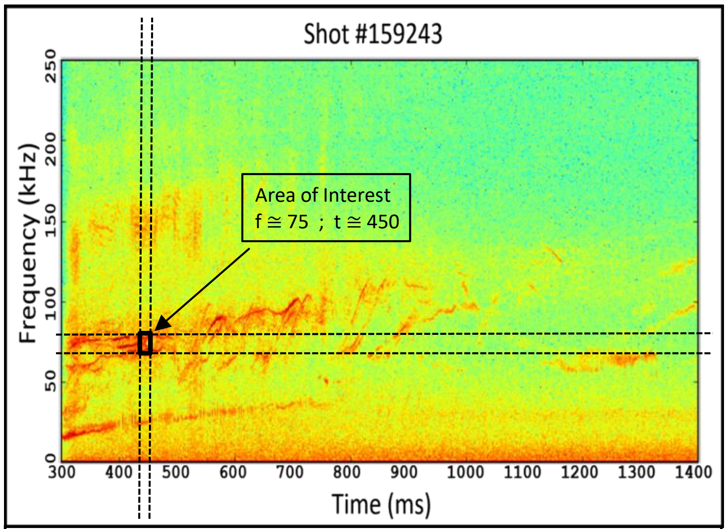

Figure 6. Spectrogram of shot 159243.

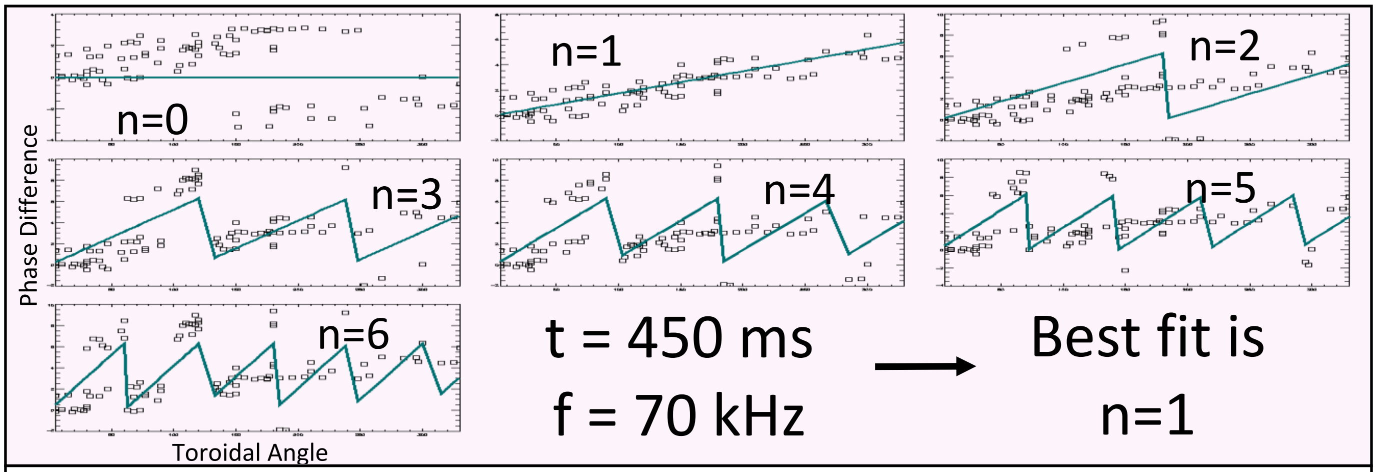

Figure 7. Each graph is an attempt to fit the toroidal mode number. Data points are phases and phase differences between probe pairs. Solid line represents the theoretical fit to that toroidal mode number.

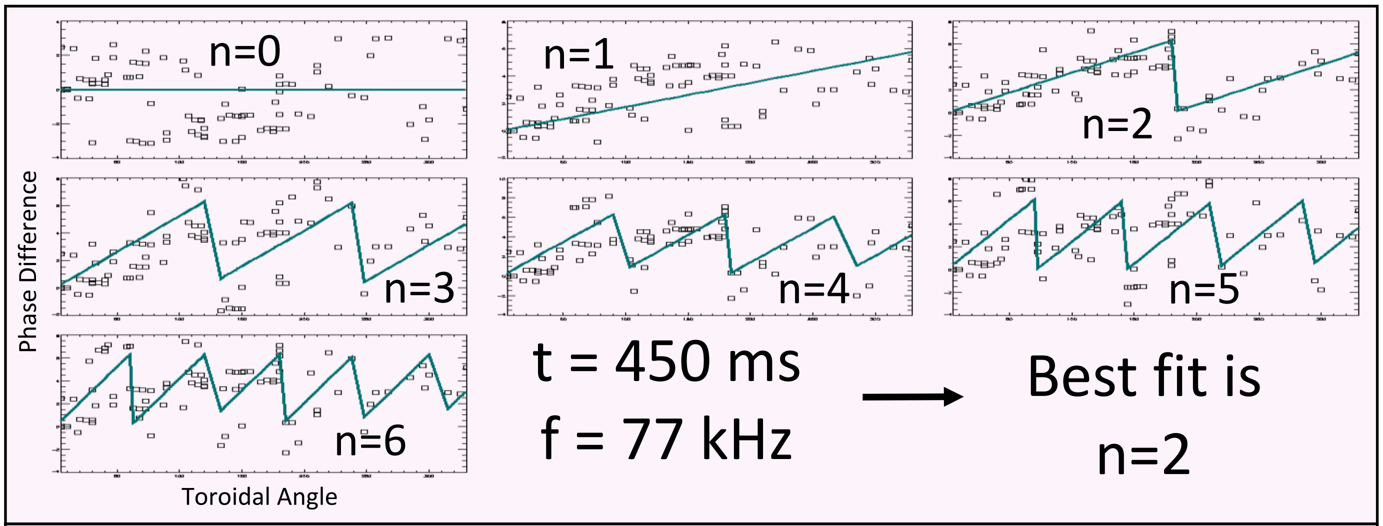

Figure 8. Each graph is an attempt to fit the toroidal mode number. Data points are phases and phase differences between probe pairs. Solid line represents the theoretical fit to that toroidal mode number.

For each frequency peak found, we can use the phase differences between probes to fit the most likely toroidal mode number (figure 7 and figure 8). For our example, we have one frequency peak at f = 70 kHz and another frequency peak at f = 77 kHz which leads to a toroidal mode number of 1 and 2, respectively.

Data

Clustering Analysis

Clustering is the act of sorting objects into groups based on user determined clustering parameters. Performing clustering analysis on bulky data sets is an effective method of uncovering an underlying structure. One common clustering algorithm, known as *k-means* uses distance between two points as the clustering parameter and groups objects together based on how far apart they are. A general k-means algorithm behaves as follows: let's say you have n-data points in a 2-dimensional space that you would like to separate into k-clusters. The algorithm begins by selecting k-cluster centers (or *means*) and then grouping each data point to the mean that it is closest too. Once each data point has been classified, the mean of each group is calculated and all the data points are regrouped based on this new set of means. This process of grouping, calculating the mean, and regrouping will iterate a set amount of times, or until a convergence criteria is reached. A more in-depth example of this process is given in Appendix B using the python programming language.

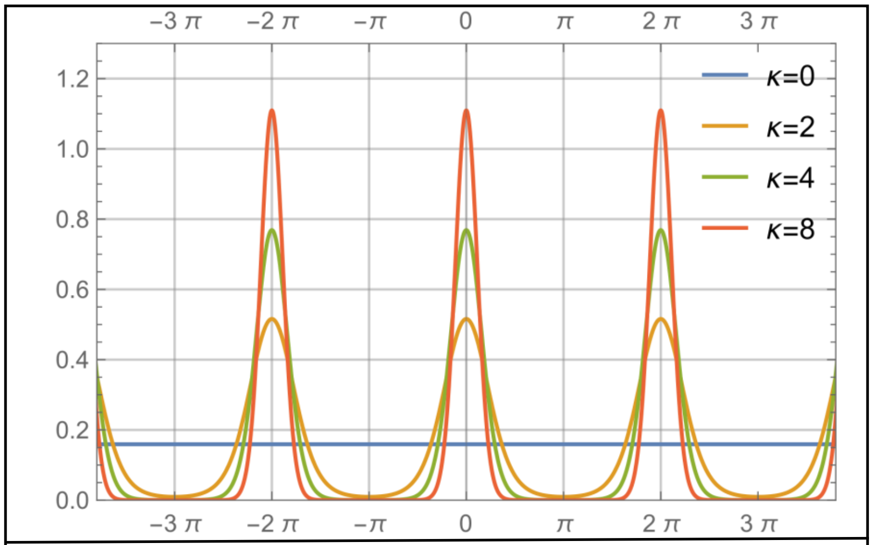

Figure 9. Example of a von-mises distribution function with the mean, μ, centered at 0 with several different κ values. κ set equal to 0 represents a uniform distribution function.

The three clustering algorithms used in this experiment were k-means, expectation maximization with gaussian mixture models (EM-GMM), and expectation maximization with von-mises mixtures (EM-VMM). EM-GMM is a clustering algorithm which uses a gaussian probability distribution function to determine the likelihood that a data point belongs to a cluster. K-means and EM-GMM are fantastic clustering algorithms due to their simplicity and robust design, however, they will both fail to adequately form clusters within periodic data. When attempting to cluster periodic data such as the phases of certain waves, we must consider that a phase of $-\pi$ is equal to a phase of $+\pi$ and form our clusters accordingly. This can be done with a few simple modifications to the EM-GMM probability distribution function, transforming it into a von-mises probability distribution function[6]:

$$f(x:\mu,\kappa) = \frac{1}{2\pi I_0(\kappa)} e^{\kappa \operatorname{Cos}(x-\mu)}$$

where x is position of the data point, μ is the mean or cluster center, κ is a parameter related to the density of the cluster, and $I_0(\kappa)$ is a modified Bessel function of the first kind with order 0. Similar clustering analysis using a program called PyFusion has been conducted on magnetics data from the H1-NF Heliac at Australia National University[1].

Toroidal Analysis and Verification

For this project, we decided to examine DIII-D shots numbered 159243 through 159257 each with a time window from 300ms-1400ms. These shots scanned across different values of beam power to vary the strength of Alfvén eigenmodes[7]. Timeseries probe data is collected and then segmented into very small (~5 ms) overlapping time chunks where we then perform short time fourier transform (STFT) analysis. The STFT spectra is averaged for each time chunk and the highest peaks are selected from the spectra. High peaks within the spectra signify a prominent frequency at that time chunk.

The phase difference between each probe and a fixed reference probe is calculated and is saved in a phase difference array (PDA). PyFusion is then run on the PDA. We begin with a k-means algorithm due to its ability to quickly locate approximate cluster centers. Once the approximate cluster centers are located, we switch over to an EM-VMM algorithm until we reach reasonable convergence. The low-density clusters identified by the EM-VMM algorithm are filtered out based on their κ value from the von-mises probability distribution function and analysis continues to the next shot. Because the analysis of each shot is independent of each other, we can utilize multiple processing threads simultaneously to dramatically decrease the amount of time required for each analysis run. After each shot is analyzed, clustered, and filtered, we run PyFusion again on *all* the shots at once. A typical run of PyFusion that clusters every shot at once (without first clustering and filtering individual shots) can take upwards of *80* GB of memory! By going through the trouble of clustering, un-clustering, and then re-clustering, we can bring down the memory usage to a much more modest 20-40 GB.

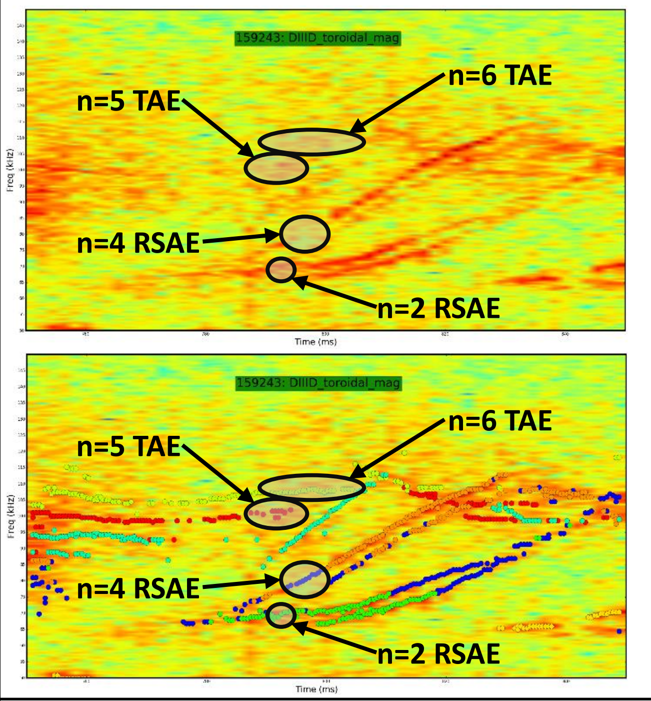

Figure 10. Top: Raw spectrogram. Bottom: Spectrogram with different clusters plotted in different colors. Identification of different Alfvén eigenmodes from *Critical gradient behavior of fast-ion transport from AE's*[7].

When developing a new method, it is in good practice to routinely test the method on a sample set of data where the outcome is already known. This becomes increasingly more important as the complexity of your problem increases. For this reason, we check to see that different toroidal mode numbers are correctly being separated into different toroidal clusters. This is done by running PyFusion on a set of known Alfvén eigenmodes[7] with different toroidal mode numbers and ensuring that they are separated into different clusters. An example of this analysis can be seen in figure 10.

Poloidal Analysis

With all this evidence that clustering analysis on the toroidal array could group different instabilities together, we thought it would be a good idea to attempt clustering analysis on the poloidal array as well. The idea of clustering on the poloidal array immediately brought up several concerns, the first of which being the fact that the poloidal probes are all at different distances to the plasma (while the toroidal probes were equidistant from the plasma). We were afraid that this would lead to erroneous spikes in the probe signal simply because some probes are closer to the plasma than others. We were also afraid of the non-uniformity of the signal since there is not nearly as much symmetry to the poloidal array than the toroidal array.

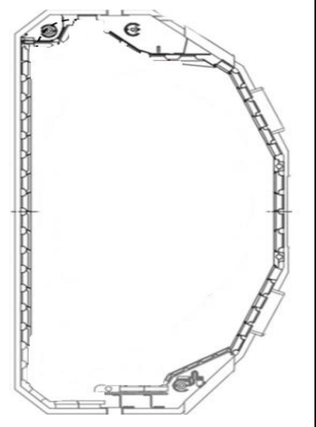
Figure 11. Poloidal cross section of DIII-D. Probes located outside of the perimeter will have varying distances to the plasma.

Fears aside, we performed the same clustering analysis that we did on the toroidal array on the poloidal array. Unfortunately, our fears were justified and we were unable to see any significant differences between poloidal clusters. We plotted the amplitude vs. probe position for 3 different clusters in figure 13, figure 14, and figure 15. We saw large peaks at the probes closest to the plasma, and almost no signal at the probes furthest away. We were unable to determine any conclusive results regarding the poloidal array.

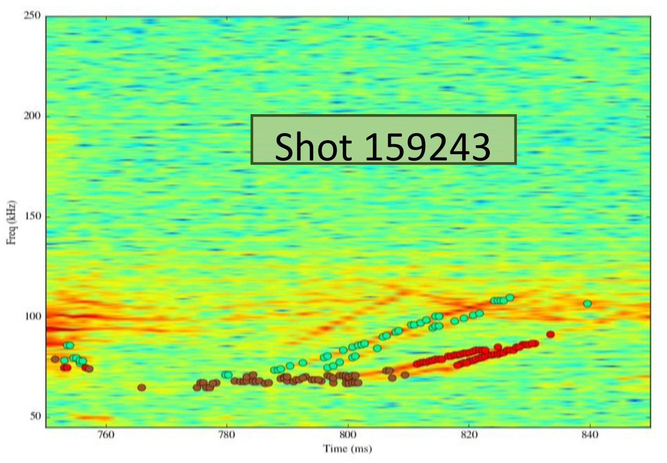

Figure 12. Poloidal Spectrogram of shot 159243 with 3 different clusters shown. In the following figures, the brown points will be referred to as *cluster 1*, the red points will be referred to as *cluster 2*, and the light green points will be referred to as *cluster 3*.

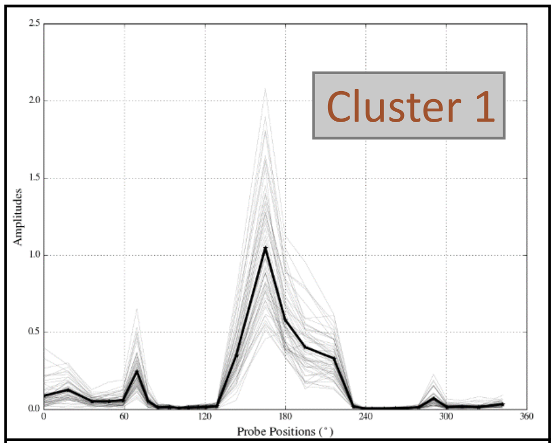

Figure 13. Poloidal amplitude vs. probe position for cluster 1. Faint lines correspond to individual signals and the dark line corresponds to the average.

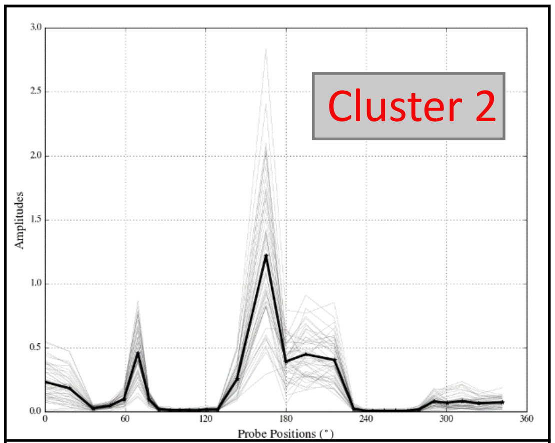

Figure 14. Poloidal amplitude vs. probe position for cluster 2. Faint lines correspond to individual signals and the dark line corresponds to the average.

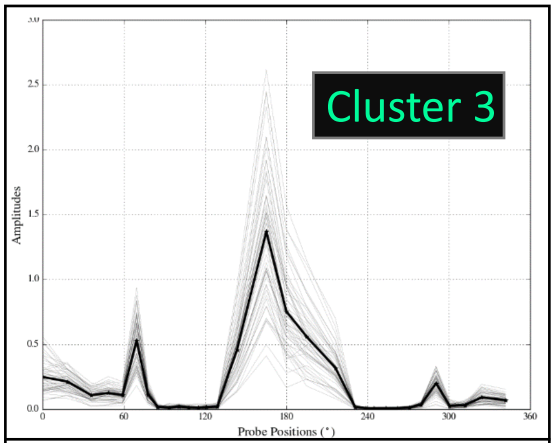

Figure 15. Poloidal amplitude vs. probe position for cluster 1. Faint lines correspond to individual signals and the dark line corresponds to the average.

Summary

A clustering algorithm, PyFusion, used to analyze magnetics data on the H1-NF Heliac at Australia National University was adapted to perform clustering analysis on the toroidal, poloidal and ECE arrays at the DIII-D tokamak in San Diego. The results from our clustering analysis showed that clustering is an effective tool for studying toroidal instabilities. When run on the toroidal array, PyFusion grouped events with similar toroidal mode numbers into distinct clusters based on the phase differences between the magnetic probes. However, when the analysis was run on the poloidal array, we were unable to discern any appreciable differences in the clusters. The varying distance between the poloidal probes and the plasma seem to have the most detrimental effect on this analysis.

Continuation of this project can go towards several different directions. Now that we are confident our clustering algorithm is capable of grouping known toroidal instabilities, we can perform our clustering analysis on a set of unknown toroidal instabilities and use metadata to determine which plasma parameters contribute to the instabilities. As it stands, this "metadata analysis" would have to be done by hand which opens up the possibility of automating the entire process. i.e. a program pipeline that performs clustering analysis and examines metadata for identification of unknown clusters. The latter is undeveloped and would require a more theoretical background on plasma behavior. On the other hand, analyzing metadata by hand could lead to discoveries and the development of new theoretical frameworks.

Computationally, PyFusion is a bit obtuse. There is a steep learning curve involved for anyone that wants to start using it, and it uses an unreasonable amount of memory (~30GB per run). Contributions towards the user-friendliness of the program could open it up to many more users and researchers alike, potentially leading to a plethora of new discoveries. Reducing the memory footprint of the program would also make it accessible to researchers who don't have access to generous worker nodes on industrial sized servers.

Acknowledgements

Special thanks go to Dr. William Heidbrink for the support and guidance throughout this project and to Dr. Shaun Haskey for the technical assistance in getting the PyFusion up and running.

All code for this project is publicly available at https://github.com/SyntaxVoid/PyFusionDIIID and is being updated with new improvements constantly.

Bibliography

1. Haskey, S. R., Blackwell, B. D. & Pretty, D. G. Clustering of periodic multichannel timeseries data with application to plasma fluctuations. *Comput. Phys. Commun.* **185,** 1669–1680 (2014).

2. Chen, F. F. *Introduction to Plasma Physics and Controlled Fusion*. (Springer, 1984).

3. DIII-D Beams. (2016). Available at: https://diii-d.gat.com/diii-d/Beams_feat. (Accessed: 16th May 2017)

4. Ogawa, K. *et al.* A study on the TAE-induced fast-ion loss process in LHD. *Nucl. Fusion* **53,** 53012 (2013).

5. Heidbrink, W. W., Strait, E. J., Doyle, E., Sager, G. & Snider, R. T. An investigation of beam driven Alfvén instabilities in the DIII-D tokamak. *Nucl. Fusion* **31,** 1635–1648 (2011).

6. Gopal, S. & Yang, Y. Von Mises-Fisher Clustering Models. in *Proceedings of the 31st International Conference on Machine Learning* (eds. Xing, E. P. & Jebara, T.) **32,** 154–162 (PMLR, 2014).

7. Collins, C. S. *et al.* Critical Gradient Behavior of Alfvén Eigenmode Induced Fast-Ion Transport in Phase Space. in *APS Meeting Abstracts* (2016).

8. Todo, Y., Van Zeeland, M. A. & Heidbrink, W. W. Fast ion profile stiffness due to the resonance overlap of multiple Alfvén eigenmodes. *Nucl. Fusion* **56,** 112008 (2016).

Appendix

A. Derivation of Alfvén Velocity

Refer to figure 2 for definitions of variables and terms. From Maxwell's equations, we have

$$\boldsymbol{\nabla}\times\boldsymbol{\nabla}\times\boldsymbol{E_1} = -\boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{E_1}) + k^2\boldsymbol{E_1} = \frac{\omega^2}{c^2}\boldsymbol{E_1} + \frac{i\,\omega}{\epsilon_0\,c^2}\boldsymbol{j_1} \quad [1]$$

Since $\boldsymbol{k}$ is only oriented in the $\hat{\boldsymbol{z}}$ direction and $\boldsymbol{E_1}$ is only oriented in the $\hat{\boldsymbol{x}}$ direction, only the x component if nontrivial. Our equation becomes

$$\epsilon_0(\omega^2 - c^2k^2)E_1 = -i\omega n_0 e(v_{ix} - v_{ex}) \quad [2]$$

The thermal motion of the particles is not important, so we can use the solution to the ion equation of motion, [3], with T_i equal to zero.

$$M\frac{\partial \boldsymbol{v_{i1}}}{\partial t} = -e\boldsymbol{\nabla}\phi_1 + e\boldsymbol{v_{i1}}\times\boldsymbol{B_0} \quad [3]$$

Which separates into

$$-i\omega M v_{ix} = -eE_1 + ev_{iy}B_0 \quad [4]$$
$$-i\omega M v_{iy} = -ev_{ix}B_0 \quad [5]$$

Solving for v_{iy} explicitly in [5] yields

$$v_{iy} = \frac{eB_0}{i\omega M}v_{ix} = \frac{\Omega_c}{i\omega}v_{ix} \quad [6]$$

We can plug this into [4] and solve for v_{ix}

$$v_{ix} = -\frac{1}{i\omega M}\left(-eE_1 + \frac{e\Omega_c}{i\omega}v_{ix}\right) = \frac{ei}{M\omega}\left(1-\frac{\Omega_c^2}{\omega^2}\right)^{-1}E_1 \quad [7]$$

Which is the velocity of the ions in the x direction. To find the electron velocity, we consider the transformation where $M \to m_e,\ e \to -e,\ \Omega_c \to \omega_c$ and the limit where $\omega_c^2 \gg \omega^2$

$$v_{ex} = \frac{ei}{m_e\omega}\frac{\omega^2}{\omega_c^2}E_1 \to 0 \quad [8]$$

This term goes to zero since $\omega_c^2 \gg \omega^2$; the Larmor gyrations are negligible so the electrons only have an $\boldsymbol{E}\times\boldsymbol{B}$ drift in the y direction. Substituting [7] and [8] into [2] yields

$$\epsilon_0(\omega^2 - c^2k^2)E_1 = -i\omega n_0 e\left[\frac{ei}{M\omega}\left(1-\frac{\Omega_c^2}{\omega^2}\right)^{-1}E_1\right] \qquad [9]$$

$$\omega^2 - c^2k^2 = \Omega_p^2\left(1-\frac{\Omega_c^2}{\omega^2}\right)^{-1} \qquad [10]$$

where $\Omega_p^2 = n_0e^2/\epsilon_0 M$ is the square of the ion plasma frequency.

Now, we take into consideration the fact that Alfvén wave frequencies are much below the ion cyclotron frequency, $\omega^2 \ll \Omega_c^2$. In this limit, [10] becomes

$$\omega^2 - c^2k^2 = -\Omega_p^2\frac{\omega^2}{\Omega_c^2} = -\left(\frac{n_0e^2}{\epsilon_0 M}\right)\omega^2\left(\frac{M^2}{e^2B_0^2}\right) = -\omega^2\frac{\rho}{\epsilon_0 B_0^2} \qquad [11]$$

$$\frac{\omega^2}{k^2} = \frac{c^2}{1+\frac{\rho}{\epsilon_0 B_0^2}} \qquad [12]$$

where $\rho = n_0 M$ is the mass density. Recall that $\epsilon_0 = (\mu_0 c^2)^{-1}$ and [12] becomes

$$\frac{\omega^2}{k^2} = \frac{c^2}{1+\frac{\rho\mu_0}{B_0^2}c^2} \qquad [13]$$

The denominator of [13] can be recognized as the relative dielectric constant for low frequency perpendicular motions, ϵ_R.

$$\frac{\omega}{k} = \frac{c}{\sqrt{\epsilon_R}} \qquad [14]$$

ϵ_R is much larger than one for most laboratory plasmas[2], so [13] can be approximated as

$$\frac{\omega^2}{k^2} = \frac{c^2B_0^2\,\epsilon_0}{\rho} = \frac{B_0^2}{\mu_0\rho} \qquad [15]$$

And we have arrived at our result for the Alfvén velocity.

$$v_a \equiv \frac{\omega}{k} = \frac{B_0}{\sqrt{\mu_0\rho}} \qquad [16]$$

B. K-Means Clustering Example in Python

This code example may be found in my GitHub repository at:
https://github.com/SyntaxVoid/PyFusionDIIID/blob/master/Writing/ClusteringExample.py

```
import matplotlib.pyplot as plt
import numpy as np
import random
```

```
## 0. Create a Point class and some usefull functions
class Point:
    ## A point is created by calling Point(x,y)
    def __init__(self, x, y):
        self.x = x
        self.y = y
        return

    def plot_self(self, axes, marker, color):
        ## Plots itself on a given matplotlib axes with a certain color
        axes.plot(self.x, self.y, marker=marker, color=color)

def distance(point1, point2):
    ## Returns the distance between point1 and point2
    return np.sqrt((point2.x - point1.x) ** 2 + (point2.y - point1.y) ** 2)

def random_point(a, b):
    ## Returns a point object with x,y values between a and b.
    return Point(random.uniform(a, b), random.uniform(a, b))

def cluster_mean(cluster):
    ## Returns the mean value of a cluster of points
    x_sum = 0
    y_sum = 0
    for point in cluster:
        x_sum += point.x  ## a += b represents a = a + b
        y_sum += point.y
    x_avg = x_sum / len(cluster)
    y_avg = y_sum / len(cluster)
    return Point(x_avg, y_avg)

## 1. Create a random set of 1000 data points
n_points = 1000
points = []
for i in range(n_points):
    points.append(
        random_point(2, 8))  # This will add a random point with x,y values
between 2 and 8 to our array of points

## 2. Generate our first set of 7 random cluster centers
n_clusters = 7
means = []
for cluster_id in range(n_clusters):
    means.append(random_point(2, 8))

## 3. Perform clustering
n_iterations = 50
for i in range(n_iterations):  ## Iterate the process 50 times
    clusters = []  ## Clear our clusters before each iteration
    for cluster_id in range(n_clusters):
        clusters.append([])  ## Create an empty cluster for as many clusters
as we have
    for point in points:  ## Iterate over every data point
        dists = []
        for mean in means:
            dists.append(distance(point, mean))  ## Calculates the distance
from the current point to each mean
```

```
        closest_mean_index = dists.index(min(dists))   ## Returns the index of
the smallest distance
        clusters[closest_mean_index].append(point)   ## Add the point to the
appropriate cluster
    for cluster_id in range(n_clusters):
        means[cluster_id] = cluster_mean(clusters[cluster_id])   ## Calculate
new mean for each cluster

## 4. Plot your clusters
colors = ["blue", "orange", "red", "green", "black", "yellow", "cyan"] ## Add
more colors here for each cluster you have
figure, axes = plt.subplots(1,1) ## Create our figure and axes objects
for cluster_id in range(n_clusters): ## Iterate over each cluster
    for point in clusters[cluster_id]: ## Iterate over each point in the
cluster
        point.plot_self(axes=axes, marker="o", color=colors[cluster_id])
    mean = means[cluster_id]   ## Next few lines plot circles around each
cluster
    mean_radius = max([distance(point, mean) for point in
clusters[cluster_id]])
    axes.add_artist(plt.Circle((mean.x, mean.y), mean_radius,
color=colors[cluster_id], alpha=0.25))

plt.xlabel("X"); plt.ylabel("Y")
plt.title("Clustering Example", fontsize=20); plt.grid(); plt.show()
```

Once the above code has been run, it should produce a something like figure 16. Notice that the clusters intersect each other and some points appear to lay in more than one cluster. This is simply an artifact of our choice to draw the cluster circles based on the most distant point from the cluster mean. Also, our input data was inherently random so one should not expect to see dense clusters.

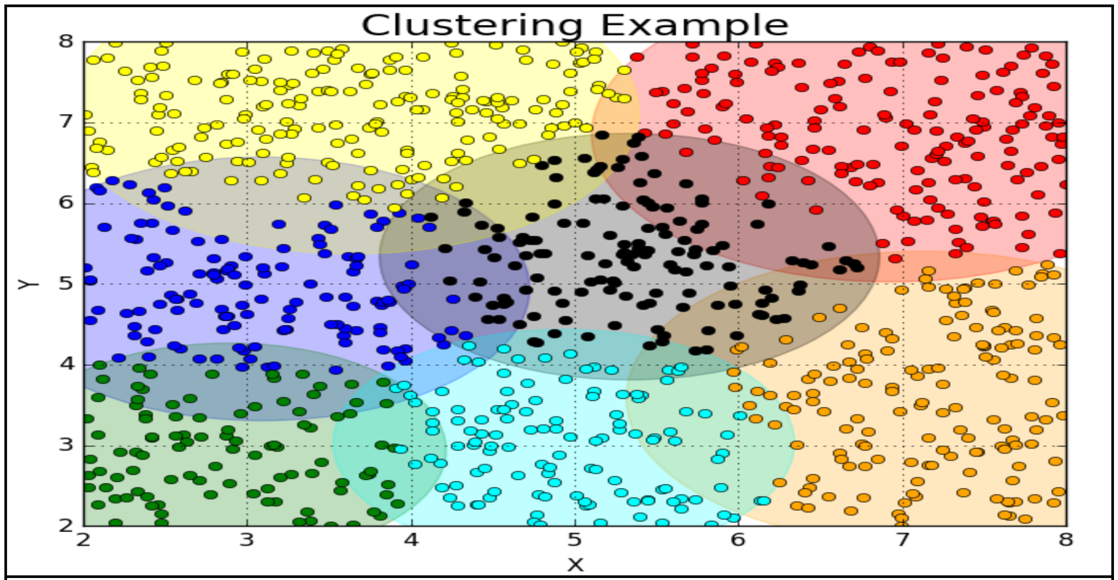

Figure 16. Sample output from ClusteringExample.py using 1000 random data points and a k-means algorithm with 7 total clusters. Clusters plotted in different clusters with a large circle indicating the radius of the most-distant point.

C. Datamining Example Using PyFusion

This code example may be found in my GitHub repository at:
https://github.com/SyntaxVoid/PyFusionDIIID/blob/master/pyfusion/DataminingExample.py

Note: You must have access to the "Analysis" module which is found in the project directory PyFusionDIIID/pyfusion/Analysis.py.

```
import Analysis

## Some settings
shots = range(159243,159246+1)
time_window = [300,1400]
probes = "DIIID_toroidal_mag"
datamining_settings = {"n_clusters": 8, "n_iterations": 20, "start": "k_means",
                       "verbose": 0, "method": "EM_VMM", "seeds": [832]}
## First, we create the Analysis object which conveniently stores all our information
in one place.
A1 = Analysis.Analysis(shots=shots, time_windows=time_window,
                       probes=probes,datamining_settings=datamining_settings,
                       markersize=7, n_cpus=1)
## Then we run the analysis, which will perform all the datamining and clustering
A1.run_analysis()
## Then, we can plot the clusters for reference
A1.plot_clusters()
## We can plot some diagnostics for a particular time and frequency
A1.plot_diagnostics(time_window=time_window, t0=801.711, f0=85.9375, idx="tor")
```

Results from this example are shown below.

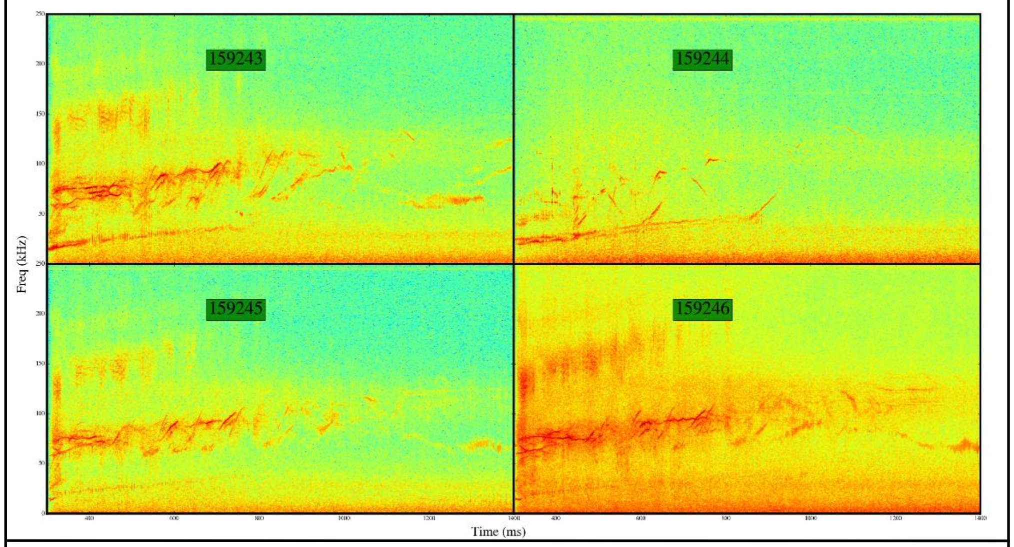

Figure 17. Raw spectrogram from toroidal magnetic Mirnov probes of shots 159243 (top left), 159244 (top right), 159245 (bottom left), and 159246 (bottom right) over a time window from 300 ms to 1400 ms.

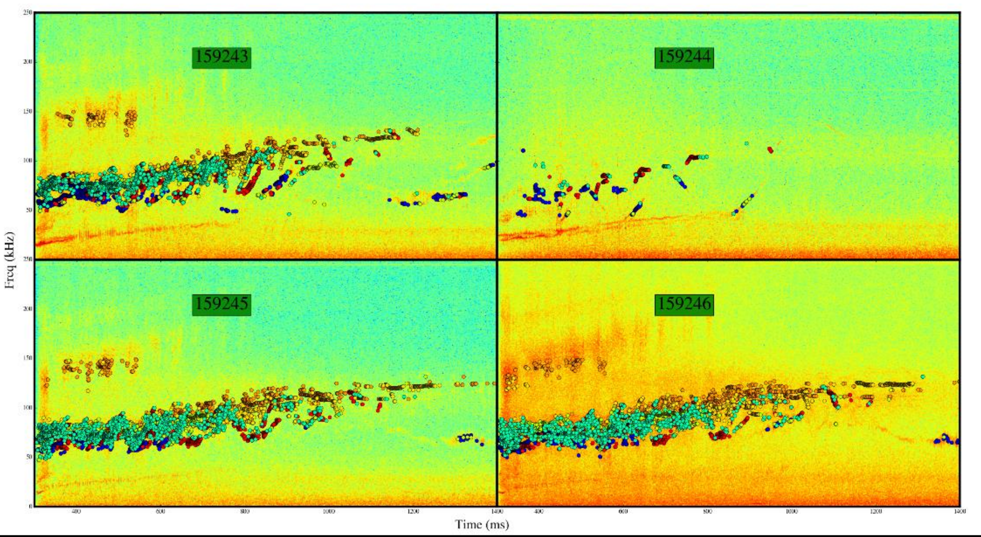

Figure 18. Spectrogram of shots 159243 (top left), 159244 (top right), 159245 (bottom left), and 159246 (bottom right) over a time window from 300 ms to 1400 ms. Different clusters are shown in different colors.

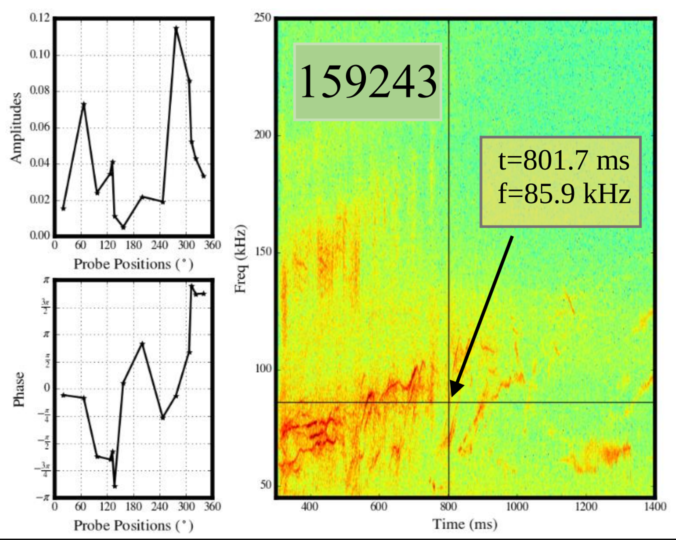

Figure 19. Top left: Amplitude vs. probe position. Bottom left: Phase vs. probe position. Right: Raw spectrogram of shot 159243. Output of *Analysis.plot_diagnostics* routine.