



Eindhoven University of Technology
&
Cerro Calán Observatorio Astronómico Nacional

Internship Report

Developing an Algorithm for the Detection of Fast Radio Bursts

S.J. Verwer
1367099

Santiago, 07 2023

Table of Contents

1	Introduction	2
2	ARTE Setup and Measurements	3
2.1	ARTE Setup	3
2.2	RFI measurements	4
2.3	Measurement Power	5
3	Detection Algorithm	6
3.1	Parameter Setup and Importing	6
3.2	Preprocessing	7
3.3	Pyramid Stacking	8
3.4	RFI Removal & Anomaly Detection	10
3.5	Dedispersion Detection	11
3.6	Hysteresis Thresholding	13
4	Algorithm Quantification	15
4.1	quadratic curve injections	15
4.2	Physical curve injections	16
4.3	Simulated FRB Injection	19
4.4	False positive detections	20
5	Conclusion	21
	Bibliography	22

1. Introduction

During a 5-month internship at the Cerro Calan radio observatory in Santiago, Chile, an algorithm has been developed for the detection of Fast Radio Bursts. Fast Radio Bursts (FRBs) as their name suggests occur for a very short time, from a few milliseconds to a couple of tens of milliseconds. Furthermore, the burst shows a quadratic dispersion behavior throughout the spectrogram. It is, however, currently unclear what the exact origin of these FRBs is, and a lot of research is conducted on the detection and understanding of these FRBs [1, 2, 3, 4]; A leading understanding is that the FRBs come from a certain class of Pulsars, magnetars [2]. An example of such a transient is shown in Fig. 1.1, where one of the first clear FRB's is shown.

The radio observatory at Cerro Calan is conducting a project called the Astronomical Radio Transient Experiment (ARTE). This project is constantly measuring the galaxy for FRB's, therefore a constant stream of data is generated and needs to be processed for any signals that are out of the ordinary, such as FRB's. The ARTE project itself will be explained in Chapter 2. However, the algorithm that will have the task of detecting these '*anomalies*', consists of preprocessing such as Radio Frequency Interference (RFI) removal. Then, two ways of detecting will be implemented, starting with hysteresis thresholding and after this detection using dedispersion is used.

The algorithm that was developed for the project works with the spectrogram and has to search through this spectrogram for any interesting phenomena. Furthermore, it will also have to filter out man-made signals to ensure a low amount of false positives while still detecting most FRBs. The working of the algorithm will be discussed in Chapter 3. In Chapter 4 the testing will be discussed to indicate how well the algorithm performs. Lastly, this report will be concluded in Chapter 5.

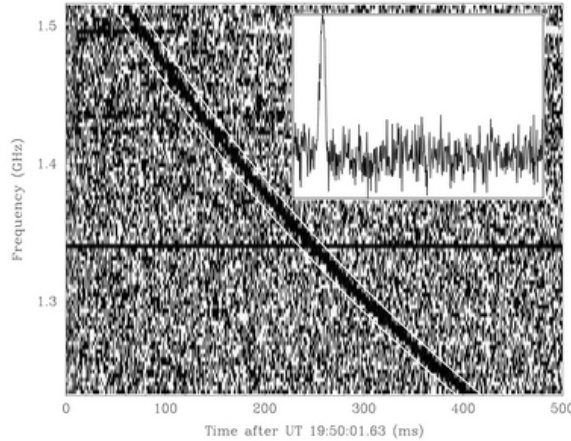


Figure 1.1: Figure showing an example of an FRB transient [5].

2. ARTE Setup and Measurements

In this chapter first, the general ARTE setup will be discussed; it will go into the reason why the setup is built as it is. Moreover, it will be entered into the measurement environment, specifically the RFI measurement at the Cerro Calan site to determine the constant RFI that will influence the measurements. This chapter will be concluded with the investigation of the temperature increase in the spectrogram that a certain power will have.

2.1 ARTE Setup

Before going into the detection algorithm, it will be necessary to discuss the setup that will generate the spectrograms and the environment in which the measurements will be made. The current theory on FRBs is that they are generated by pulsars [2]. Most of the pulsars in our galaxy are centered in the galaxy, as can be seen from Fig. 2.1. For this reason, the ARTE project consists of an antenna that will measure most of the pulsars in the center of the galaxy, which is circled in red.

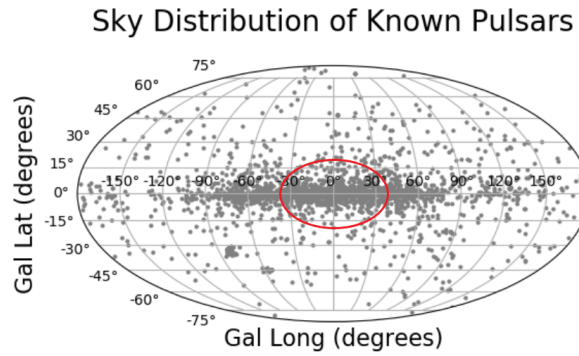


Figure 2.1: Figure showing the distribution of Pulsars in the galaxy in galactic coordinates, in red the Full Width Half Maximum (FWHM) of the ARTE antenna is shown. [6].

The antenna used is a phased array antenna that is placed on a mount that will then follow the center of the galaxy. The setup itself is shown in Fig. 2.3. The setup is divided into three subarrays, where every subarray consists of four antennas. Furthermore, the first two of these subarrays will be digitally combined to synthesize the main beam. The last one is used to determine the galactic longitude and latitude. The detection itself will consist of two parts, first a ROACH-2 FPGA will consist of a dedispersion algorithm that will perform a data drop once it detects a peak power above a certain threshold. These are data with very high resolution, and therefore too large to constantly dump. The other part is a constant log file data dump; this consists of 5 minutes of data with a time resolution of 10 ms and a frequency spacing of 292 kHz from 1200 MHz up to 1800 MHz. For the detection algorithm that will be discussed in Chapter 3 this second part, the logfile data will be analyzed on FRBs. An example of two such log files plotted as one spectrogram is shown in Fig. 2.2. The spectrogram plots the antenna temperature, not the power, so the unit for this will be K. This is also the starting point for the algorithm developed, which will be discussed in chapter 3. Important to note is that the log file has some flagged channels, these are channels that consist of a lot of RFI.

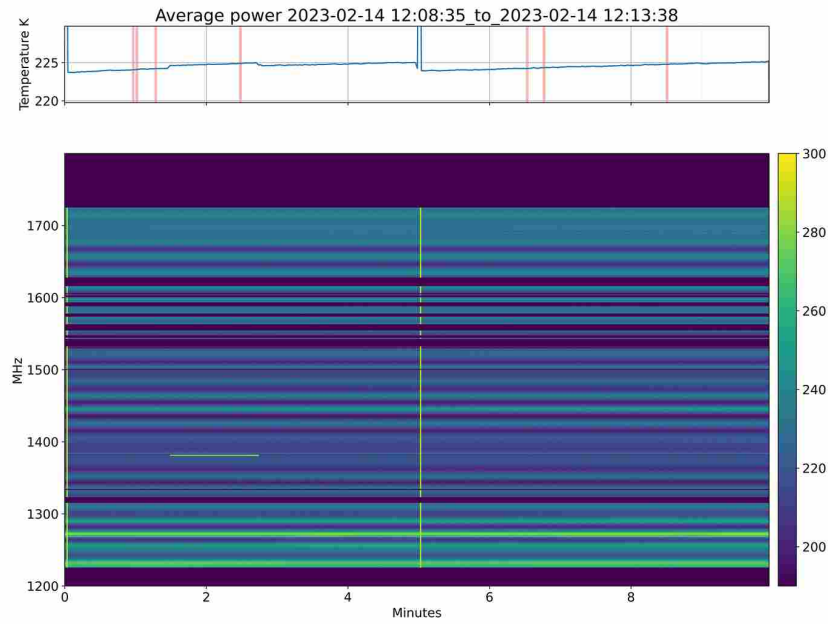


Figure 2.2: Example of a spectrogram generated from two 5-minute log files.

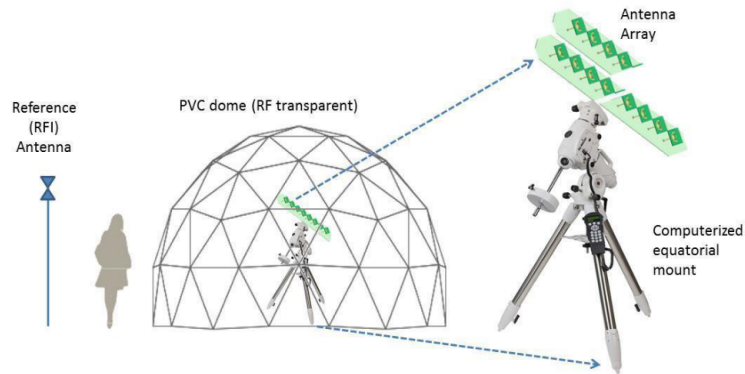


Figure 2.3: Figure showing the ARTE setup.

2.2 RFI measurements

After understanding the setup, it is necessary to understand the measurement environment. Most interesting will be the constant RFI. This is the RFI that will constantly influence the measurements and this is the RFI that is the most predictable and, therefore, can be filtered out at the beginning. Furthermore, it gives information about the measurement site itself and the electromagnetic contamination at the specific site. This measurement was performed at two sites, Cerro Calan and Santa Martina. For the measurement a large band, from 25 MHz up to 3 GHz was chosen. Moreover, the measurement was performed over a couple of minutes to see the constant RFI. The results of these measurements are shown in Fig. 2.4

The measurement is performed by splitting the band into four blocks, where one block takes an average of 28 ms, which are sequentially measured for various minutes. Because of this, only the

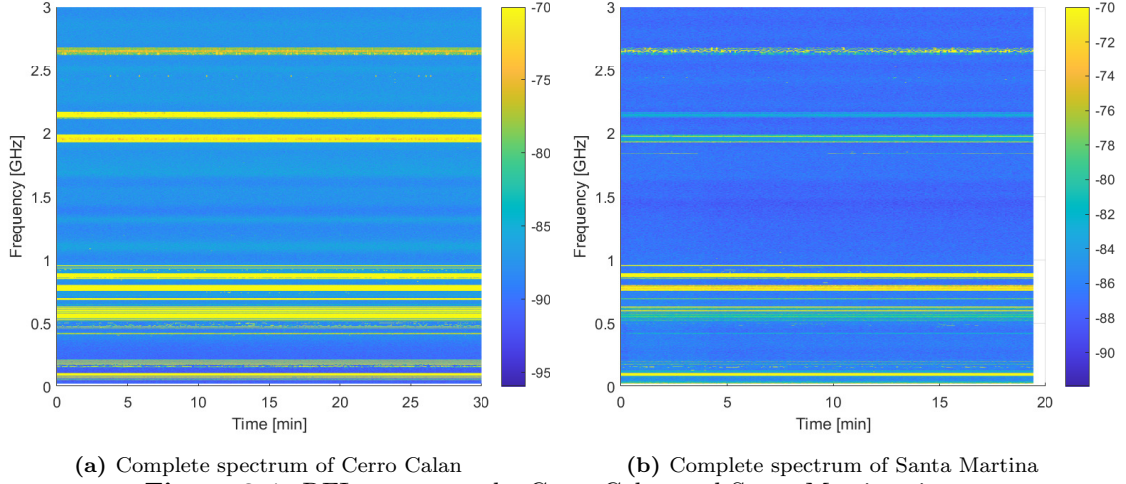


Figure 2.4: RFI spectra at the Cerro Calan and Santa Martina site.

RFI that is really constant and not pulse-like will be shown in these plots. Thus, for example in the frequency range from 1-2 GHz, it looks like there is no RFI but this is not a valid conclusion as there could be pulse-like RFI. For that reason, in Fig. 2.2 some channels are flagged although there is no RFI in the band seen from Fig. 2.4.

2.3 Measurement Power

Next, it is important to determine what effect certain powers will have on the spectrogram and to determine what will be measurable. This is something that will of course depend mostly on the noise of the spectrogram versus the signal power that the interesting phenomena will have. An indication of the standard deviation of the noise can be determined from the spectrogram shown in Fig. 2.2, where the standard deviation is approximately 5.2 K, depending of course on the specific moment at which this is calculated as there can be deviations between log files. Moreover, we are mostly interested in the relative temperature increase and not the absolute ~~increase~~, as the events will be a superposition of the noise temperature with the event temperature. The equation for this temperature increase is given as,

$$T_{source} = \frac{A_{eff}}{2k_B} S_v \quad (2.1)$$

Where, T_{source} is the temperature increase, A_{eff} is the effective aperture of the antenna, k_B is the Boltzmann constant, and S_v is the flux density of the event at earth. Calculating the flux density of an event requires rewriting this equation in terms of T_{source} . The effective aperture requires the calculation of the directivity, which can be approximated from its half-power beam width [7],

$$D_{approx} = \frac{4\pi(180/\pi)^2}{\Theta_{1d}\Theta_{2d}}. \quad (2.2)$$

Here, Θ_{1d} and Θ_{2d} are the half-power beam widths of the antenna in degrees. Then, the effective aperture is calculated as follows,

$$A_{eff} = \frac{\lambda^2}{4\pi} \eta_{eff} D_{approx}. \quad (2.3)$$

Where λ is the wavelength, η_{eff} is the antenna efficiency. Taking an efficiency of 90% for the antenna and a wavelength of 1500 MHz the power that the event would need is approximately 240 kJy. This compares to the STARE-2 setup that has a sensitivity of 200 kJy at 10 ms time resolution [2]. Thus, this would also be a good lower limit for the detection of FRBs in the ARTE system.

3. Detection Algorithm

In this chapter, the developed detection algorithm will be discussed. The algorithm can be divided into 5 main stages,

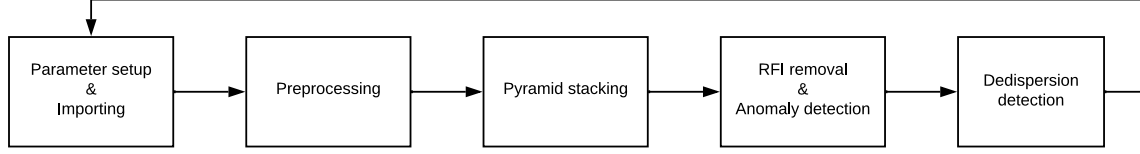


Figure 3.1: Algorithm divided into 5 general stages that will be further subdivided in flowcharts.

This section will go over all these 5 stages in the order given in Fig. 3.1. There will be a reasoning for the importance of these stages and an explanation of the functioning of the stages. The detection itself is based on hysteresis thresholding, which is used various times throughout the code such as in Pyramid stacking, the RFI removal and anomaly detection, and the Dedispersion detection. As this is a relatively thorough code, it will be explained separately in Section 3.6. It is important to note that the flowcharts shown throughout this section are meant as a guide to the code and show the general process that the code of that section follows for ease of understanding. Therefore, the more straightforward parts have been omitted to keep the flow charts short and clear.

3.1 Parameter Setup and Importing

This is the most simple part of the algorithm, starting with setting up file locations and saving locations. Then the log file will be loaded and the data will be converted to usable antenna temperature. As every log file has a moment when the data is calibrated, processing this calibration is also performed in this part of the code. The transformation from log file to antenna temperature data was already previously coded and is used in the algorithm as a starting point. Moreover, as the algorithm is meant to detect mostly FRBs and thus short events, it was decided to look at 1 log file at a time. The process of this stage can be seen in the flow chart shown in Fig. 3.2.

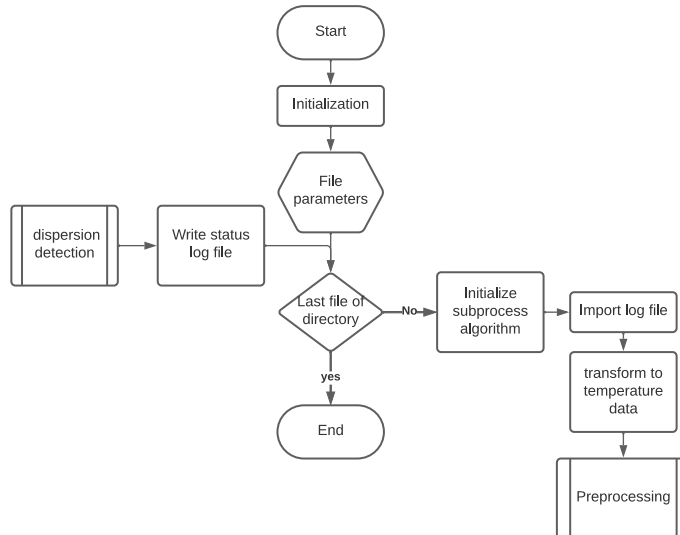


Figure 3.2: Flowchart on the parameter setup and importing part of the detection algorithm. The pre-defined process blocks indicate the other stages as shown in Fig. 3.1.

The flow chart from Fig. 3.2 is relatively self explanatory, and should be followed with the code

as it is described as the general process this part of the code follows.

3.2 Preprocessing

Once the data are loaded correctly, the pre-processing of the data will start. For preprocessing, it has been chosen to set up a Python class named '*Galactic_data*' which will then contain various functions to plot parts of the data, or a function to process the data. The processing function called '*Data_processing*' is the function that performs the preprocessing.

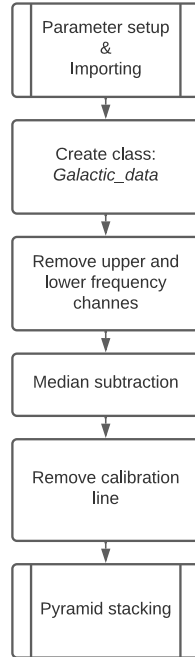


Figure 3.3: Flowchart on the parameter setup and importing part of the detection algorithm. The pre-define process blocks indicate the otehr stages as shown in Fig. 3.1.

In the flow chart shown in Fig. 3.3 there is a part that removes the upper and lower frequency channels. It has been chosen to look only in the 1300 MHz up to 1700 MHz as this is the band where the complete chain is mostly flat, while outside of this band, it falls off, as can be seen in Fig. 3.4. The following part is the median subtraction, or as the noise can be assumed as quasi-Gaussian, a mean subtraction. The reason for this, as noted above, is to only look at the differences in temperature, not the absolute temperature, which is a common approach [8, 9, 10, 11]. Finally, the calibration line will be removed; this is a line with a high temperature and would be detected as an anomaly, but as the properties are known, can be removed beforehand.

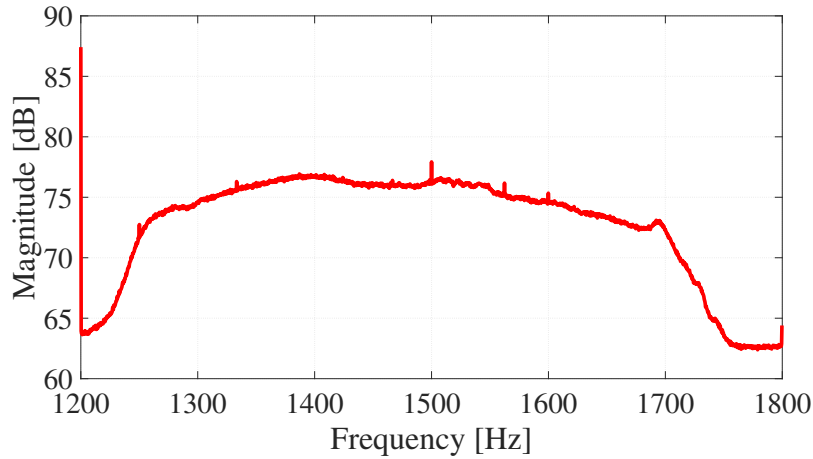


Figure 3.4: S21 of the receiver chain.

3.3 Pyramid Stacking

The pyramid stacking part is meant as a way to enhance weak signals that are visible only by averaging, as then the noise will be reduced, but the signal itself will stay more or less the same. This works by only looking at different layers and then once it detects something in that layer to give weight to that detection. However, when there is nothing there, it will not add anything. Thus, short signals will stay with a 10 ms resolution, while longer signals could have an increased time resolution depending on the signal at that specific location. The methods for creating the pyramid are based on the Laplacian pyramid method [12]. Usually, when readding all the Laplacian layers, you get the original image back; however, when changing the weights at certain locations, it is possible to enhance these parts, as can be seen in Fig. 3.6.

The way pyramid stacking is implemented is shown in the flow chart in Fig. 3.5. To generate a Laplacian pyramid you start generating a Gaussian pyramid; this is basically blurring the image using a Gaussian averaging. Following this, the Laplacian pyramid can be generated, which works by looking at the difference between the pyramids. Then the hysteresis thresholding part will look for interesting phenomena, and only those parts will be enhanced, while the rest will normally be summed back to the original picture. However, averaging the removed calibration will give some artifacts which should be properly removed. This improved spectrogram with the enhanced parts will then go on to RFI removal and anomaly detection.

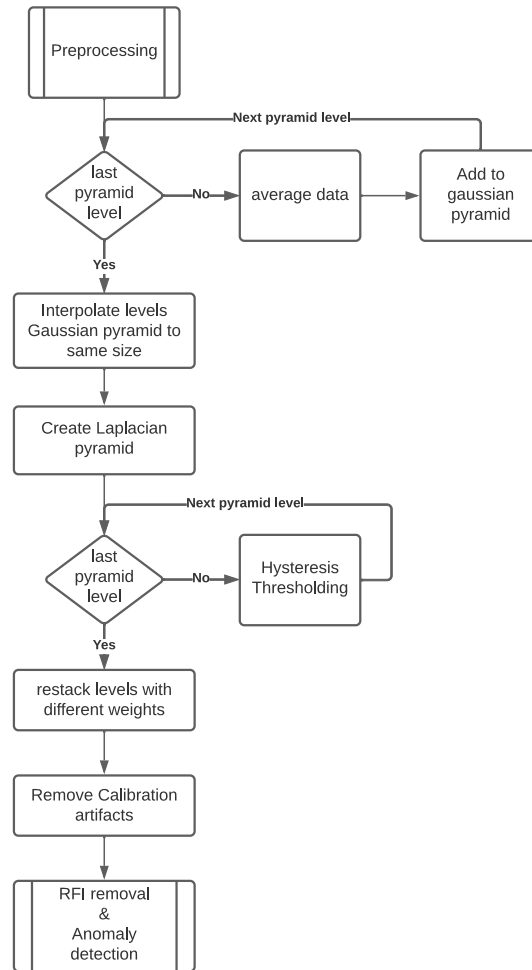


Figure 3.5: Flowchart on the pyramid stacking part of the algorithm.

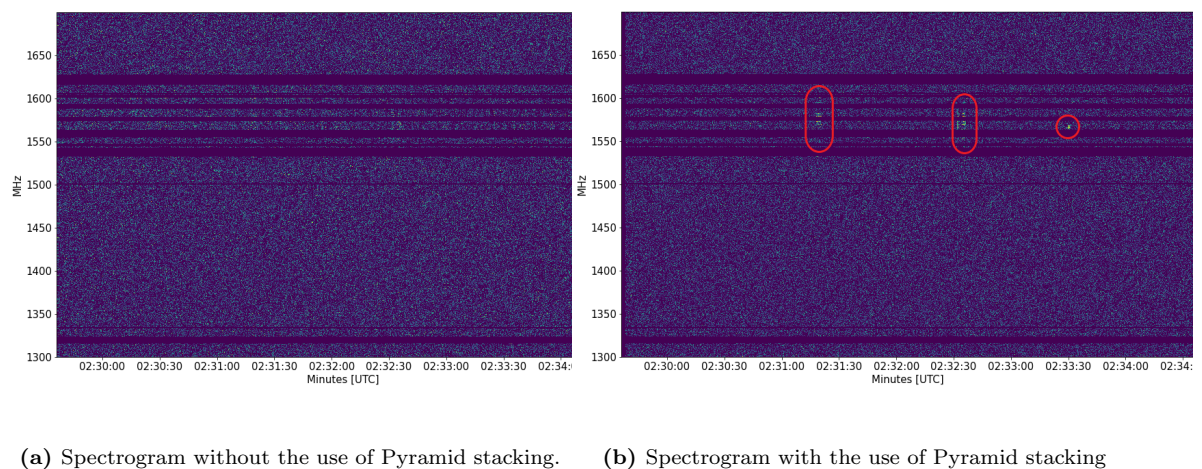


Figure 3.6: Figures showing the difference between a spectrogram with and without Pyramid stacking for the same log file.

3.4 RFI Removal & Anomaly Detection

This section will describe the removal of the RFI and anomaly detection. The flow chart for this part can be seen in Fig. 3.7. First, the hysteresis thresholding will detect signals that are out of the ordinary. This means signals that have a standard deviation of more than 4 above the noise. However, on their own, these are just points and these points will require meaning.

The way to give them meaning is to cluster them together. There are many ways of clustering but here it was chosen to use the density-based spatial clustering with noise (DBSCAN). The reason for this is that it has a good trade-off between computational time and cluster creation. In general, for clusters you want to look at how close points are together, the most simple method would be hierarchical clustering, specifically single linkage clustering [13]. However, the downside of this clustering method is the high time complexity [14]. Furthermore, there is high sensitivity to noise for these hierarchical methods, which is not the case for the used DBSCAN method [15]. Lastly, as the dataset is of a large size, there are three methods advised from the scikit library in python. These are, DBSCAN, BIRCH and K-means [16]. Where BIRCH is an hierarchical method, and K-means looks at the centre of the cluster which is not the way we want to cluster, as especially RFI will be a vertical line, so it will not capture the entire RFI.

Following the clustering, the clusters will be divided into different cluster types. These types range from Type 0 to type 5. These are arbitrary types, mostly necessary for debugging and understanding what type of signals will be placed where. The conditions can be seen in equation 3.1. It can be seen that the conditions are mostly size dependend. The FRB when detected and of strong enough power, will usually be placed in the type 1 or type 0 folder, this is due to how the detection is clustered as sometimes the FRB might be divided into multiple clusters. Next to this, there is a 5'th type, this the internal RFI type and is currently not printed, as for a usual log file this will be about 500-1000 detections. It is also classified in this type at the Time average filtering step, and not at the cluster classification step.

$$\text{Type} = \begin{cases} 4, & \text{if } f \leq 3 \text{ MHz} \\ 3, & \text{else if } 3 \text{ MHz} < f < 6 \text{ MHz} \wedge t < 200 \text{ ms} \\ 2, & \text{else if size} \leq 100 \\ 1, & \text{else if } f < 88 \text{ MHz} \\ 0, & \text{otherwise} \end{cases} \quad (3.1)$$

The time averaging step basically takes the cluster window and then averages this over all of the time, such that a frequency plot results. Then everything that is above 3 sigma standard deviation above the noise is regarded as a detection. Then it will look for peaks in this data, and if the peak is of a width smaller than 4.4 MHz then the cluster will be renamed as a type 5 cluster, and thus as RFI. The idea behind it is that, in general, the RFI is a linear effect and, relatively tightly constraint to a certain frequency band. The way the value for 4.4 MHz is determined is by testing over various values and finding when the false positives are the least, but the false negatives of FRB's are close to zero. An example of this plot can be seen in Fig. 3.8. Here various peaks are shown, and everything higher than 3 sigma from the noise is clipped. Then the width of these peaks are determined to figure out if it is RFI or not.

Now that we have distinguished between RFI and other Anomalies the removed flagged channels will be re-added to be able to go to the plotting of the spectrogram. Now that most of the detections are filtered out there will be one last check done for every cluster, this is by checking if the cluster consists of lines that are completely horizontal. This is done by sub-clustering, meaning that the

one cluster is divided up further in connecting parts. If all of these sub-clusters are considered horizontal, thus the slope is ≤ 1 . Then the cluster is regarded as RFI and will be ignored. If however, one or more sub-clusters is not horizontal than it will be regarded as something worth investigating, and in turn, this will then be saved in the folder corresponding to the previously mentioned cluster type. After all the clusters have been checked, all these detections will be set to zero and the code will continue into the dedispersion part.

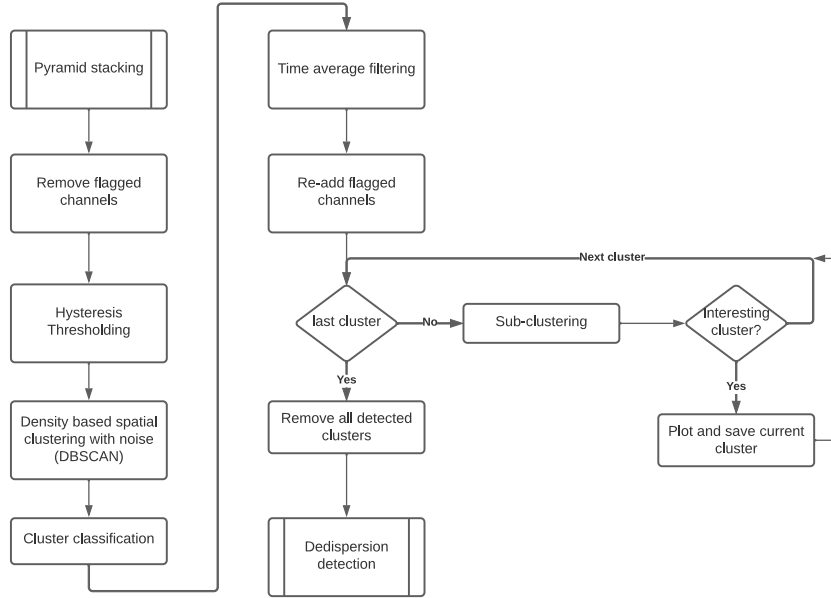


Figure 3.7: Flowchart on the removal of RFI and the detection of anomalous signals.

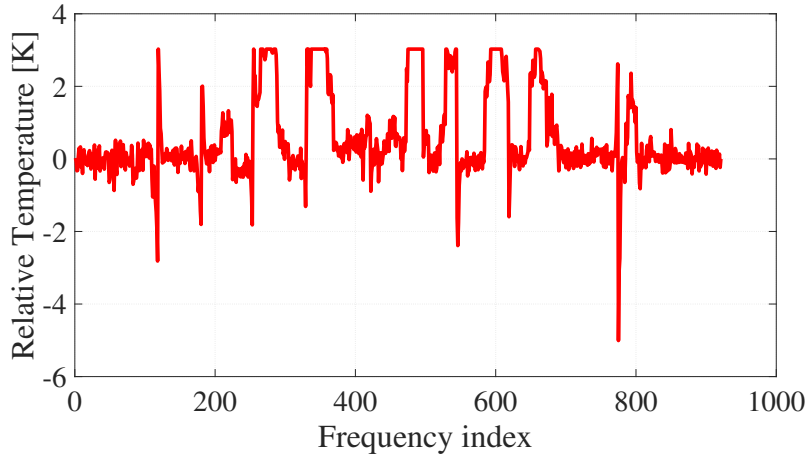


Figure 3.8: Example of the frequency averaging of the time average filtering part. It shows the detected signals as strong peaks here, where any peak with a width of less than 15 points (4.4 MHz) will be regarded as RFI.

3.5 Dedispersion Detection

The last part of the code is to see if any signals are missed, especially FRBs, by implementing another way of finding FRBs, the flow chart is shown in Fig. 3.9. In this part, the data will be

dedispersed which means time delaying the frequency channels and then summing over frequency to see if a peak will occur. In general, there are two ways to implement dedispersion, coherent dedispersion devolves the transfer function, while incoherent dedispersion shifts frequency channels in time, and averages of the result [17, 18]. In this algorithm, the incoherent dispersion is used. This method is easier to implement and computationally fast enough for the application.

It is then possible to do the dedispersion for various test dispersion measures, and then a data set can be created that shows the time versus the dedispersion measure; see Fig.3.10 for a fake FRB and its corresponding dedispersion plot. Within this dataset the hysteresis thresholding is used again to obtain interesting points. Then the points are clustered together again; as for an FRB the change in the plot will be smooth, as can also be seen in Fig.3.10b . After testing it was found that if the cluster is greater than 15 points, that the false negatives are very low. Moreover, only dispersion measures of more than 100 are taken into account and the FRB temperature must be greater than 1 K. If all these points are satisfied, then the FRB will be plotted in a specially allocated FRB folder.

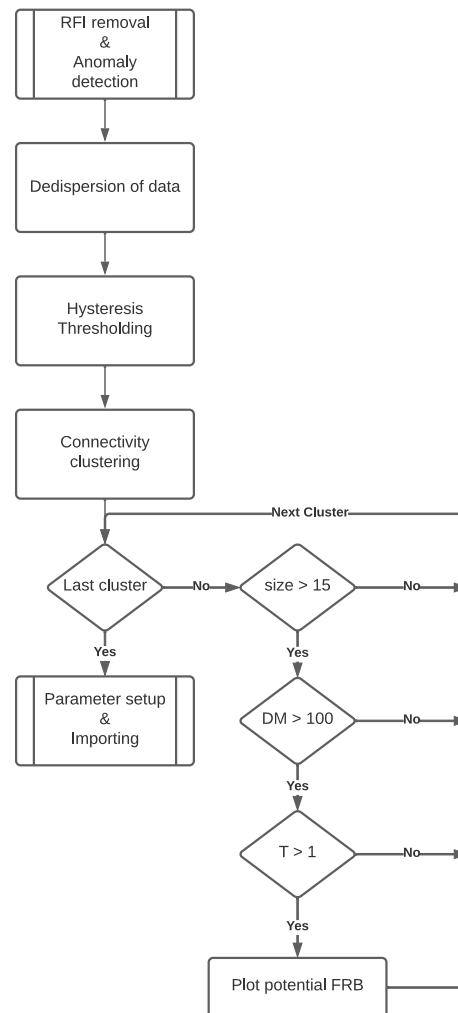
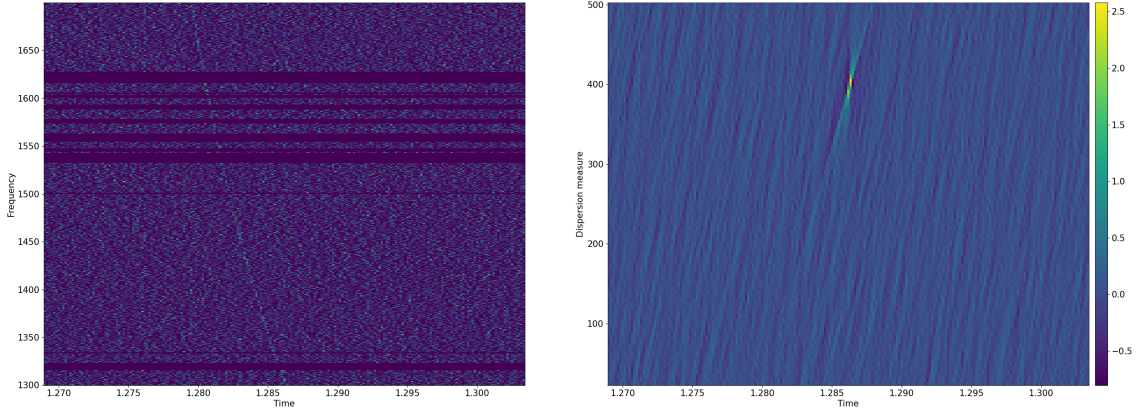


Figure 3.9: Flowchart on the implemented dedispersion detection method.



(a) Example of a fake FRB injected in the spectrogram. (b) Dedispersion plot of the injected fake FRB

Figure 3.10: Figures showing an injected fake FRB and the corresponding dedispersion plot.

3.6 Hysteresis Thresholding

As hysteresis thresholding is an important part of the code and is called various times throughout the algorithm, for understanding purposes, it is necessary to explain it in more detail. The basic concept of hysteresis thresholding is setting two threshold values. Everything below the lowest threshold is regarded as nothing. Everything above the highest threshold is considered a detection, and everything in between is only considered a detection if it is connected to a point above the highest threshold [19, 20]. The implementation can be seen in Fig. 3.11.

In the implemented version of the thresholding, the upper and lower thresholds are determined by the noise of the specific frequency channel. As an example in the RFI and anomaly detection part described in section 3.4, the lower threshold is set at 1 sigma and the upper threshold is set to 4 sigma.

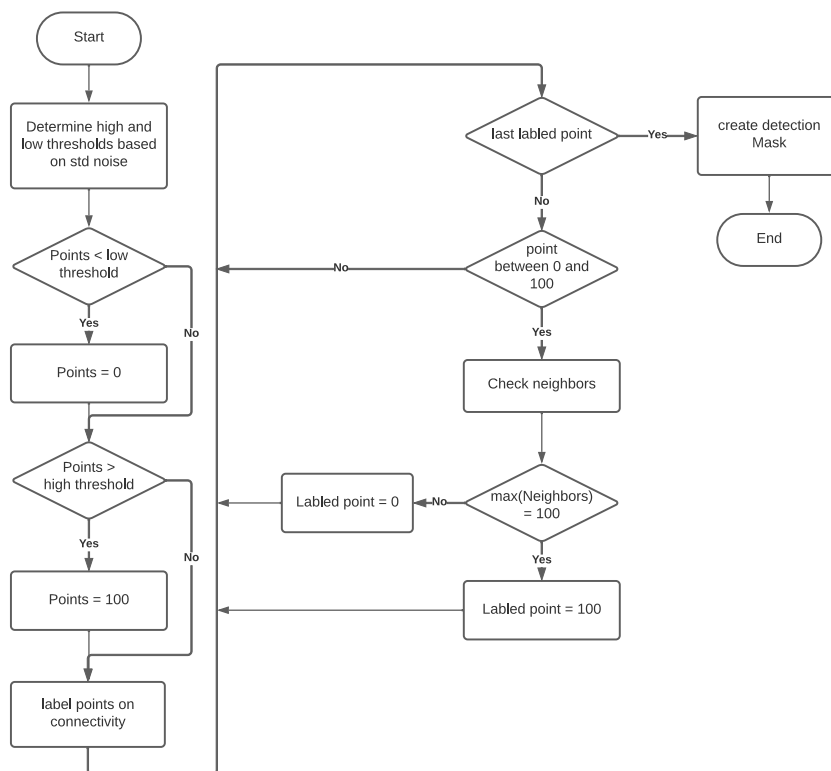


Figure 3.11: Flowchart on the implementation of hysteresis thresholding.

4. Algorithm Quantification

In this section, the performance of the described algorithm will be quantified to determine how well it works. It starts by injecting quadratic curves as fake FRBs, as this is the most simple way of describing and FRB. With this, three tests are performed. Firstly, the power of the FRB is randomized and compared with random dedispersion measures. Following this, random power is compared to the random frequency bandwidth.

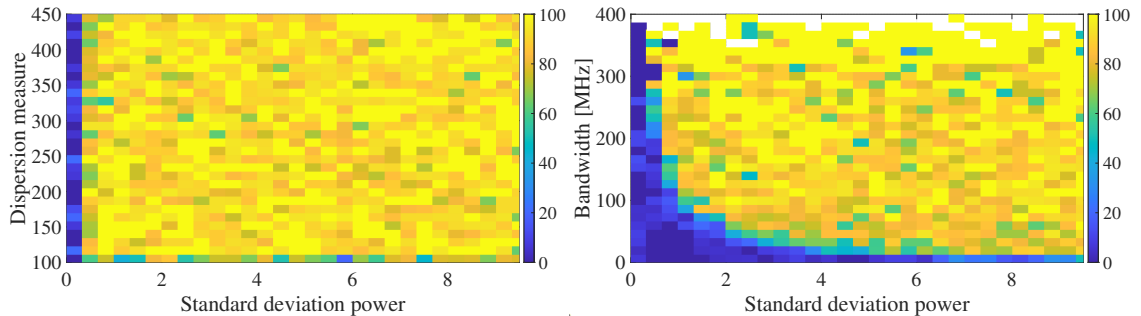
To continue the test, these quadratic curves are sent out to the system using an antenna. To see how it will be seen in the real log files without artificially adding the curve. Finally, the tests are concluded by testing the algorithm with a simulated injection of an FRB-based on this paper [21].

4.1 quadratic curve injections

The quadratic curve injection implies injecting a curve that is of the form,

$$t = k \cdot DM \cdot \left(\frac{1}{f_f^2} - \frac{1}{f_i^2} \right). \quad (4.1)$$

An example of such a curve is shown in Fig. 3.10a. For the injection a couple tests have been performed. First, a test comparing the percentage detected versus random power versus random dispersion measure. The result of this can be seen in a surface plot in Fig. 4.1a. Following this, a test has been performed to compare the percentage detected versus the random power versus the frequency bandwidth of the curve; for this test a surface plot can be seen in Fig. 4.1b. To obtain a better idea of how well the algorithm can detect FRBs with different power levels, also a figure has been generated showing the power versus percentage of curves detected, which is shown in Fig. 4.2. Important to note is that the percentages in the surface plots are lower than the actual detection. This is because the code is tested with another code, where this second code sometimes misses detected curves. The difference between a manual check of the data set and an automatic check is shown in Fig. 4.2. Furthermore, in Fig. 4.2 also the different location where the FRB ends up for different powers is shown.



(a) Standard deviation of power with respect to noise compared to the dispersion measure. (b) Standard deviation of power with respect to noise compared to the frequency bandwidth

Figure 4.1: Figures showing the percentage of fake FRBs injected according to the model of equation 4.1. The detection was performed for different frequency bandwidths and different dispersion measures.

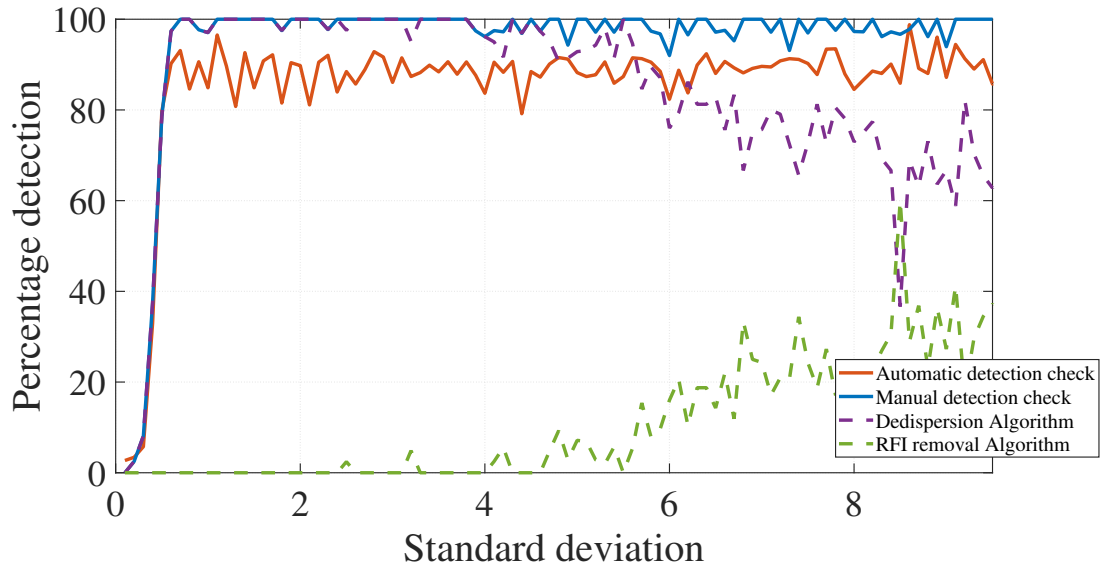


Figure 4.2: Figure showing the power versus the percentage detected for an FRB with a DM of 400. Moreover, this plot also shows the error of the automatic test for the FRB detection algorithm. In purple and green the difference is seen between where the FRB ends up being detected, the dedispersion part of the code or the RFI removal part of the code.

4.2 Physical curve injections

To test how the algorithm behaves on the curves injected with an antenna, the same quadratic curves were injected by shooting them toward the system. This can be seen in various Figures 4.3, 4.4, 4.5, 4.6. These Figures show differences between where the detections end up and in how the amplitude changes for this FRBs. Furthermore, in Fig. 4.5, 4.6 the limitations of the weak FRBs is shown. The generated RFI will of course also introduce more false positives.

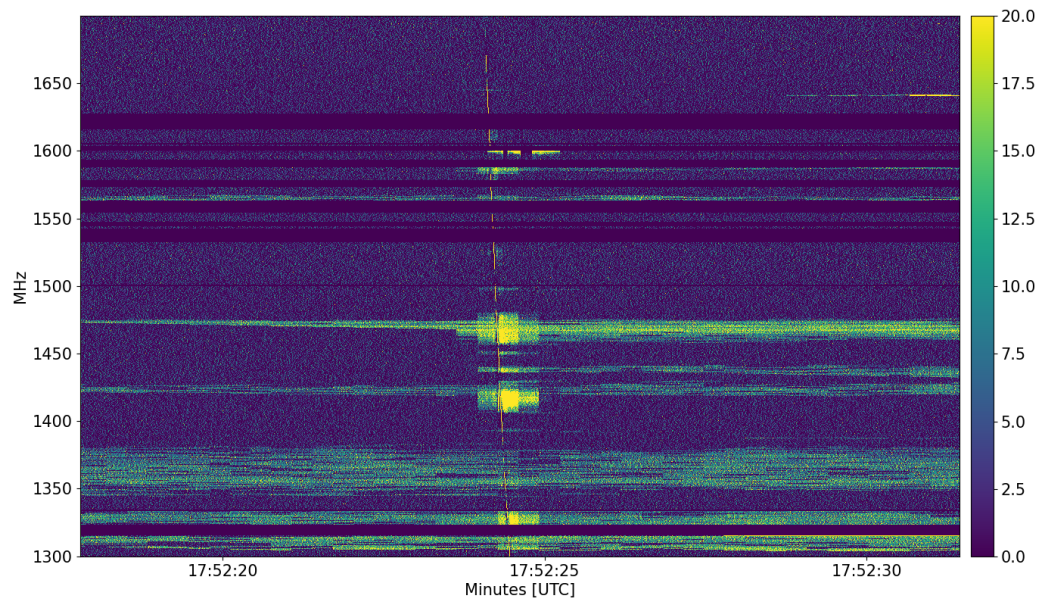


Figure 4.3: Figure showing a curve shot at the detection system, the signal is characterized by a lot of RFI generated by our generation system.

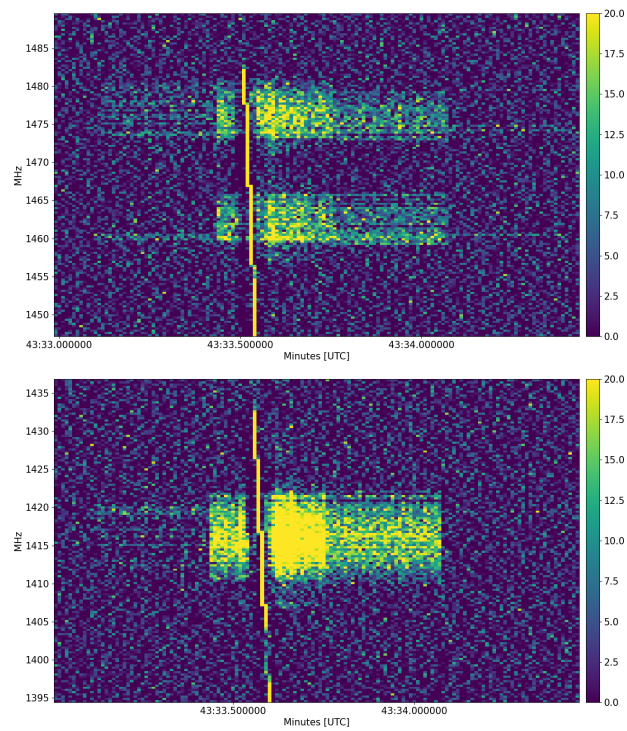


Figure 4.4: Two figures showing a shot FRB that has been split up between two different detections as the power over frequency is not uniform.

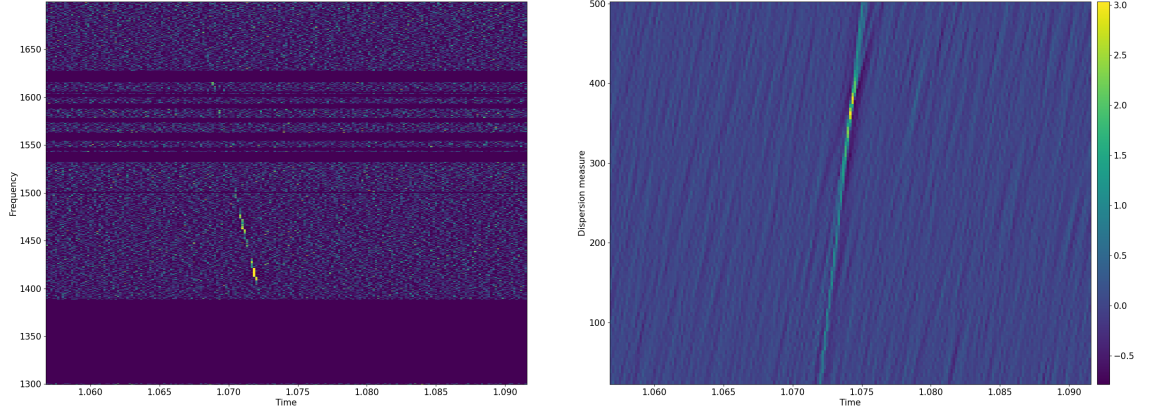


Figure 4.5: Example of a shot FRB that has been detected by the dedispersion part of the code. It can be clearly seen that it is very weak over frequency, but clearly visible in the dedispersion part.

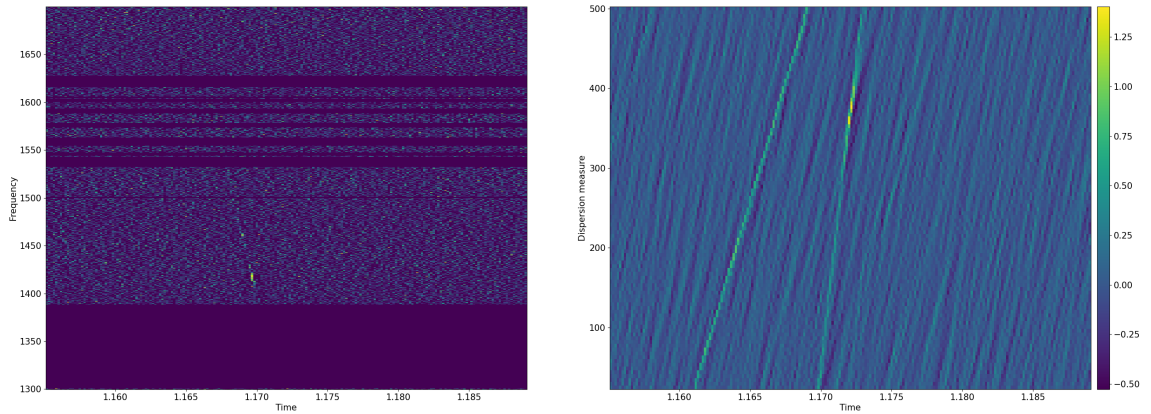


Figure 4.6: Example of a shot FRB that has been detected by the dedispersion part of the code. It can be clearly seen that it is very weak over frequency, and barely visible in the dedispersion plot.

4.3 Simulated FRB Injection

As a last FRB simulation, a better model of an FRB was also used. This model was developed by another student at the observatory and is based on the following paper [21]. For this test, due to limited time, only weak FRBs have been tested, as this will find the limit of the algorithm. Furthermore, the earlier results with the quadratic curves have shown that strong FRBs are detected very well. The result of this test is shown in Figs. 4.7 and 4.8. The idea of this method is to generate a pulse at the Pulsar, and then model the propagation through the galaxy to generate the correct dispersion measure. For this reason, the FWHM of the pulse is given and the amplitude is in arbitrary units as the original amplitude is not necessary for the algorithm. Only the received amplitude compared to the noise.

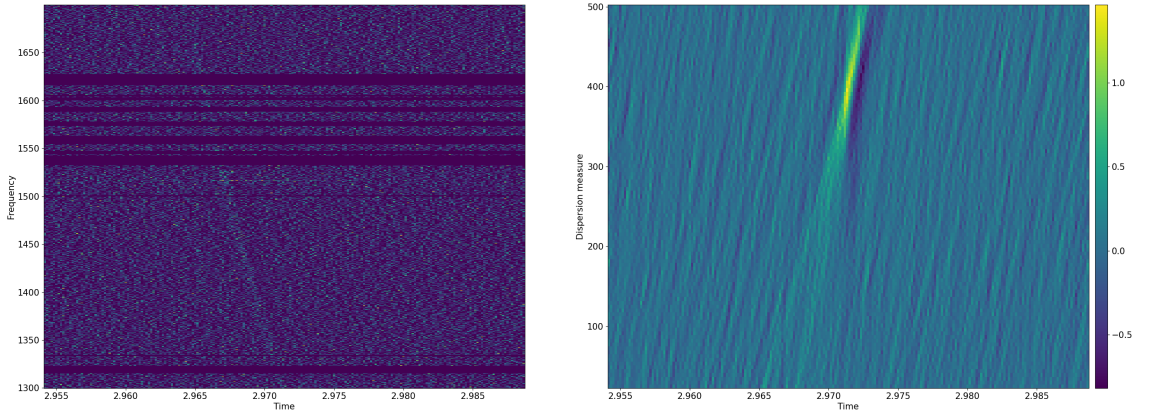


Figure 4.7: Example of a more accurate model of an FRB injected in the data. For this example, an amplitude of 5 A.U. and FWHM of 0.05 MHz has been used.

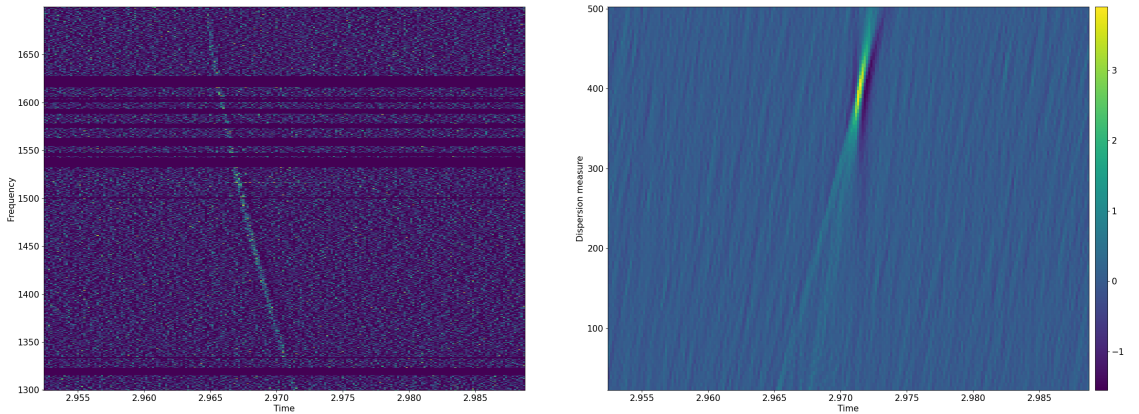


Figure 4.8: Example of a more accurate model of an FRB injected in the data. For this example, an amplitude of 10 A.U. and FWHM of 0.025 MHz has been used.

4.4 False positive detections

As a last measure, it is important to understand how many false positives are generated by the code. In general, false positives are characterized by RFI detections. To obtain an idea of how detections are generated on average per log file, the code was run on a data set from September 2022. This data set consists of 2306 log files, so in total this consists of 192 hours and 10 minutes. In general, these were 9 different days.

The results of this false positive test can be found in Table 4.1. Here, the results are split up by day to give an indication that on good days the false positives can be very low, such as on the 9th and 10th of September. However, on the 11th the false positives are relatively high. To be able to compare the values, a column with total detections is shown, following this a column with the average detections per log file is shown, as some log files have more detections than others. Lastly, the average false positives have been shown for log files that consist of one detection or more. Furthermore, these parameters have also been calculated for the complete data set.

Table 4.1: False positive detections within the September dataset

	Total detection	Mean detections per log file	Mean detections per positive log file
09-09-2022	21	0.525	3.5
10-09-2022	207	0.7289	6.9
11-09-2022	339	1.2372	19.9412
12-09-2022	314	1.0979	5.0645
13-09-2022	743	2.6162	14.86
14-09-2022	556	1.9509	6.5412
15-09-2022	665	2.3252	7.4719
16-09-2022	955	3.3509	8.7615
17-09-2022	493	1.8059	10.2708

Total detections complete set	4293
Mean detections per log file complete set	1.8617
Mean detections per positive log file complete set	4.2802

5. Conclusion

In conclusion, a first version of a detection algorithm has been developed. For FRBs with a power of up to 0.6σ , the code works well with a detection rate of 96%, and with this, the result is on par with the STARE-2 detection system. There is of course still a lot of room for improvement, mostly towards reducing the false-positive detections. Moreover, there has been a new method of calibration for the system that should be implemented into the algorithm, as the current code only works with log files that have been generated before March 2023.

Other types of improvement that could help increase the sensitivity of the algorithm would be to improve the linearity check in the sub-clustering part. This does not work perfectly and results in many false positive detections. The anomalies could be further subdivided to determine a better way to filter out the anomalies from the RFI. What is more, is that the parameters have been determined on the quadratic model, which seems to work well. However, by optimizing the parameters on the more realistic model, a better code could be reached. Lastly, the code is not perfectly efficient, and there are many code optimizations possible to make the code run more smoothly.

Bibliography

- [1] E. Petroff, J. W.T. Hessels, and D. R. Lorimer. Fast radio bursts at the dawn of the 2020s. *Astronomy and Astrophysics Review*, 30(1), 2022. 2
- [2] Christopher D. Bochenek, Daniel L. McKenna, Konstantin V. Belov, Jonathon Kocz, S. R. Kulkarni, James Lamb, Vikram Ravi, and David Woody. STARE2: Detecting fast radio bursts in the milky way. *Publications of the Astronomical Society of the Pacific*, 132(1009), 2020. 2, 3, 5
- [3] Bing Zhang. The physical mechanisms of fast radio bursts. *Nature*, 587(7832):45–53, 2020. 2
- [4] James M. Cordes and Shami Chatterjee. Fast Radio Bursts: An Extragalactic Enigma. *Annual Review of Astronomy and Astrophysics*, 57:417–465, 2019. 2
- [5] D. R. Lorimer, M. Bailes, M. A. McLaughlin, D. J. Narkevic, and F. Crawford. A bright millisecond radio burst of extragalactic origin. *Science*, 318(5851):777–780, 2007. 2
- [6] Kevin Stovall. Pulsar Overview. *Presentation IPTA 2018 Student Workshop*, 2018. 3
- [7] D. M. Pozar. *Microwave Engineering 4th Edition*. Wiley, 4 edition, 2011. 5
- [8] Yunfan Gerry Zhang, Vishal Gajjar, Griffin Foster, Andrew Siemion, James Cordes, Casey Law, and Yu Wang. Fast Radio Burst 121102 Pulse Detection and Periodicity: A Machine Learning Approach. *The Astrophysical Journal*, 866(2):149, 2018. 7
- [9] Eric Thrane and Michael Coughlin. Searching for gravitational-wave transients with a qualitative signal model: Seedless clustering strategies. *Physical Review D - Particles, Fields, Gravitation and Cosmology*, 88(8):1–8, 2013. 7
- [10] Pravir Kumar, Barak Zackay, and Casey J. Law. Detecting Fast Radio Bursts with Spectral Structure using the Continuous Forward Algorithm. 2023. 7
- [11] Kshitij Aggarwal. Searching Harder , Localizing Better , Classifying Faster : Optimizing Fast Radio Burst Detection And Analysis Searching Harder , Localizing Better , Classifying Faster : Optimizing Fast Radio Burst Detection And Analysis. 2021. 7
- [12] Peter J. Burt and Edward H. Adelson. *The Laplacian pyramid as a compact image code*, volume 0. Morgan Kaufmann Publishers, Inc., 2009. 8
- [13] Marina Blanton. Hash Functions. *Encyclopedia of Database Systems*, pages 1–2, 2014. 10
- [14] Dongkuan Xu and Yingjie Tian. A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, 2(2):165–193, 2015. 10
- [15] M. Daszykowski and B. Walczak. 2.26 - Density-Based Clustering Methods. *Comprehensive Chemometrics: Chemical and Biochemical Data Analysis, Second Edition: Four Volume Set*, 2:565–580, 2020. 10
- [16] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, and Matthieu Perrot Edouard Duchesnay. Scikit-learn: Machine Learning in Python. *Machine Learning Research*, 12:2825–2830, 2011. 10
- [17] Wei Liu, Qiao Meng, Chen Wang, Chenye Zhou, Shunyu Yao, and Irfan Tariq. A Real-time, Pipelined Incoherent Dedispersion Method and Implementation in FPGA. *Publications of the Astronomical Society of the Pacific*, 134(1031):0, 2022. 12
- [18] Girin I. A., Likhachev S. F., Andrianov A. S., Burgin M. S., Popov M. V., Rudnitskiy A. G., Soglasnov V. A., and Zuga V. A. Processing System for Coherent Dedispersion of Pulsar Radio Emission. (January 2019), 2023. 12

- [19] John Canny. A Computational Approach to Edge Detection. (6), 1986. 13
- [20] Sajid Khan and Dong-ho Lee. Efficient and Improved Edge Detection Via a Hysteresis Thresholding Method Efficient and improved edge detection via a hysteresis thresholding method. (March), 2020. 13
- [21] Marcus Merryfield, Shriharsh P. Tendulkar, Kaitlyn Shin, Bridget Andersen, Alexander Josephy, Deborah Good, Fengqiu Adam Dong, Kiyoshi W. Masui, Dustin Lang, Moritz Münchmeyer, Charanjot Brar, Tomas Cassanelli, Matt Dobbs, Emmanuel Fonseca, Victoria M. Kaspi, Juan Mena-Parra, Ziggy Pleunis, Masoud Rafiei-Ravandi, Ketan R. Sand, Paul Scholz, Kendrick Smith, and Ingrid H. Stairs. An Injection System for the CHIME/FRB Experiment. *The Astronomical Journal*, 165(4):152, 2023. 15, 19