# Breakthrough Listen Internship 2020

Charles Giese,
Trinity College Dublin, BSRC Breakthrough Listen, I-LOFAR

September 15, 2020

**Abstract**

...

## 1 Radio Astronomy with I-LOFAR

One of the main challenges of this internship was getting to grips with the data formats of radio astronomy data as well as the software used for analysing radio data. The main tool used during this internship was Sigproc, which is detailed below. Furthermore, Radio emission undergoes a process called dispersion, wherein the Interstellar medium (ISM) causes the speed at which electromagnetic waves propagate to vary with frequency. The dispersion measure $DM$ is a constant of proportionality relating the frequency of radiation to the amount of time by which that radiations is delayed, relative to vacuum. The dispersion measure depends on the distance $d$ from source to observer, as well as the electron column density $n_e$. An expression for the dispersion measure is derived below.

### 1.1 Dispersion

The ISM can be treated as a plasma. At the centre of this plasma all particles oscillate naturally like harmonic oscillators as a result of coulombic interactions. For two particles;

$$\frac{q_1 q_2}{4\pi\epsilon_0 r^2} = -m\omega^2 x \tag{1}$$

Exchanging one of the particles for a charge distribution gives

$$\frac{q_1 \rho dV}{4\pi\epsilon_0 r^2} = -m\omega^2 x \tag{2}$$

$$\implies \frac{e^2 n_e 4\pi r^2 dx}{4\pi\epsilon_0 r^2} = -m\omega^2 dx \tag{3}$$

$$\implies \omega_p = \sqrt{\frac{n_e e^2}{m_e \epsilon_0}} \tag{4}$$

$$\implies f_p = \frac{\omega_p}{2\pi} = \sqrt{\frac{n_e e^2}{4\pi^2 m_e \epsilon_0}} \tag{5}$$

which is the plasma oscillation frequency.Light waves, travelling through a plasma, like the ISM, have a group velocity that is not c, due to the refractive index of the plasma;

$$v_g = cn, \quad n^2 = 1 - \left(\frac{f_p}{f}\right)^2 \tag{6}$$

$$\tag{7}$$

The time it then takes to travel from a distance $d$ to earth is then

$$t = \int_0^d \frac{dl}{v_g} \tag{8}$$

$$= \frac{1}{c}\int_0^d (1 - \left(\frac{f_p}{f}\right)^2) dl \tag{9}$$

which can be expanded using a Taylor expansion to

$$t - \frac{d}{c} = \frac{1}{c}\int_0^d \frac{1}{2}\left(\frac{f_p}{f}\right)^2 dl + \ \ldots\ldots \tag{10}$$

where $\frac{d}{c}$ is the vacuum travel time. Thus

$$t_{dm} = \left(\frac{e^2}{8\pi^2 m_e c \epsilon_0}\right)\frac{1}{f^2}\int_0^d n_e dl \tag{11}$$

$$= 4148 \ \times \ \frac{DM}{f_{MHz}^2} \ seconds \tag{12}$$

thus giving the dispersion measure as well as the time delay due to dispersion for a given frequency. The effect of dispersion is illustrated below in Figure 1, a Gaussian pulse arrives at the observer first at the highest frequency (channel 0) and all subsequent channels are delayed. Note the characteristic quadratic shape the pulses form.

In order to negate the effects of dispersion, radio data is de-dispersed before being analysed. In practise, this was done using either Sigproc (incoherent de-dispersion) or David McKenna's Coherent De-dispersion Measure Trials (CDMT). For details of the difference between coherent and incoherent de-dispersion, see D. R. Lorimer and Michael Kramer's Handbook of Pulsar Astronomy [1].

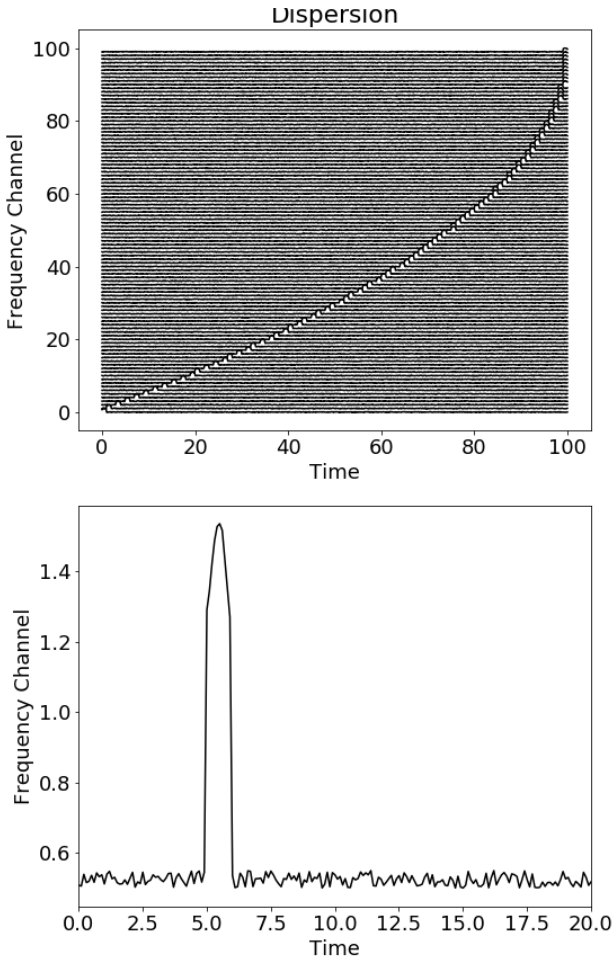


Figure 1: Illustration of how radio signals are dispersed, pulse arrives first at the highest frequency (channel 0) and subsequent channels are delayed

When analysing data of a known source, the data is simply de-dispersed at the correct dispersion measure. Otherwise, data is de-dispersed for a range of dispersion measure trials, and the results of each de-dispersion are searched.

## 1.2 Sigproc

One of the primary tools used during this internship was Sigproc. Sigproc ((Pulsar) Signal Processing Programs) is a suite of software developed by Dunc Lorimer [2], used for processing radio pulsar data. The suite can be used to form filter-bank files, which can be easily processed and searched. In general a filter-bank file consists of *nifs* polarisation channels, *nchans* frequency channels of *nbit* numbers. Thus, the data can be thought of as a 1-D array of N-1 elements where

$$N = nifs \times nchans \times nsamples$$

where *nsamples* is the observation time divided by *tsamples*. Sigproc includes a number of useful functions allowing filterbank files to be de-dispersed, decimated in time and/or frequency, folded, or searched for single pulses and periodicity. If a future intern wishes to use Sigproc at REALTA I recommend using docker to do so. This is relatively simple, a sample command for starting up docker is illustrated below;

```
docker run -it
-v <direc>:<direc>
pulsar-gpu-dsp2020
```

where <direc> is the directory you want to mount to the docker container. You'll want to mount directories containing any data and scripts you want to use in the docker container. If using a GPU, include the flag --gpus all. The docker container also includes a number of other pulsar/FRB processing software including PSRCHIVE and DSPSR. Once the docker container has been activated, any and all Sigproc functions can be called by simply entering the function in the command line. Documentation for basic use of Sigproc can be found on the Sigproc SourceForge page [2] and M.J. Keith has written a helpful guide to using Pulsar search software. [3]

## 1.3 Data Recording and Processing

Observing a source of your choice can be straightforward at I-LOFAR. First, one has to be allocated a time slot when the station is in local mode. This is generally done on a first come first served basis. An observing script should then be generated and sent to whomever is Observer for this local mode. An example observing script, created by David McKenna, is outlined in Appendix A.

This produces a set of 4 .log files, detailing a log of the recording process as well as 4 .zst files, which contain compressed rawUDP data packets generated during the observation. These 4 files correspond to 4 ports and can be processed using the udpPacketManager [4] or coherently de-dispersed using Coherent Dispersion Measure Trials CDMT, all of which are included in the aforementioned docker container. The udpPackerManager is useful for performing ordering/re-ordering operations or for processing the data and forming Stokes. The outputs of data ordering/reordering produces .rawudp files which can then be further processed in python if needed, for example see my Polyphase filterbank in my Git repository. Use of the CDMT is straightforward, here is an example command;

```
cdmt_udp
-d 61.252,0,1
-o '202007'
-m psr_ref_hdr
-f 8
udp_%d.ucc1.2020-07-01T05:31:00.000.zst'
```

where the flags signify DM (start, step, num), output name, header file, FFTs per operation and file location respectively. %d iterates over the 4 ports 16130 to 16133. Through use of these tools Sigproc style filterbank files can be produced from raw data. It should be noted, that for large data files at high DMs, the time delay between top and bottom frequency channels is too large for Sigproc to parse when de-dispersing. In these situations, if the data is in .zst format, it is best to de-disperse using CDMT, or if the data is already in filterbank format, to de-disperse using Python. When using Python, the numpy function numpy.memmap() can be very useful for not loading entire data files at once, see code in my GitHub repository for examples. Sigpyproc is also useful for loading filterbank files into python.

## 2 Lorimer Burst

A Fast Radio Burst (FRB) is a transient radio pulse lasting from a fraction of a millisecond to a few milliseconds. They are known to be caused by some high energy astrophysical phenomenon not yet understood. The first ever detected FRB was discovered in 2007 by Duncan Lorimer who assigned his student, David Narkevic, [5] archival Parkes data from 2001 to analyze again. In order to become familiar with radio data and Sigproc, this data was searched at I-LOFAR in order to find the so called Lorimer Burst.

In order to search the data for signs of an FRB, the data had to first be de-dispersed using the correct dispersion measure (DM) value. As this value was unknown, the data was de-dispersed for a range of DM values. The upper limit of this range was determined by first determining the DM contribution of the Milky way using the Cordes-Lazio NE2001 Galactic Free Electron Density Model [6], which was found to be 44.33 pc/cc. Since most FRB's are extragalactic the max DM value would have to be far greater than this, and a value of 500 pc/cc was chosen. Each of the 13 pointings was de-dispersed at integer values in the range $(0, 500)$, forming timeseries. A single pulse search was then carried out on each timeseries and the results saved. *Seek*, Sigproc's search utility, can produce .pls files, which contain details of every single pulse the algorithm found. In order to analyse the results, a plot was produced with time on the x-axis, DM on the y-axis and S/N of the hit on the z-axis, which was implemented through a colormap.

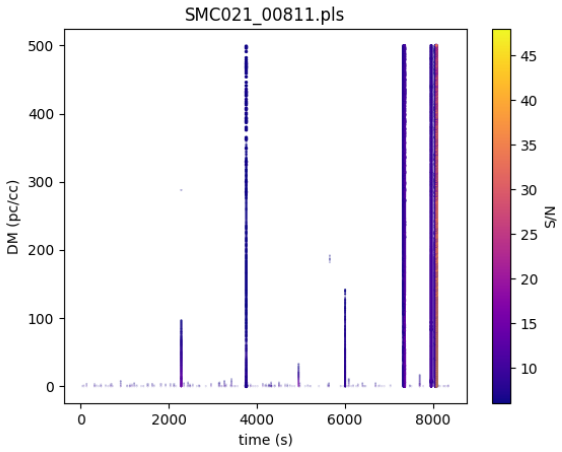


Figure 2: Results of Sigproc single pulse search for Beam 1

Above is an example of such a plot, for Beam 1. A lot of RFI can be easily identified from these plots. Any signals which appear at all DM's can be disregarded, as well as signals which appear in all 13 pointings. Thus, a candidate signal was determined and found to appear in Beams 6, 7, and D.

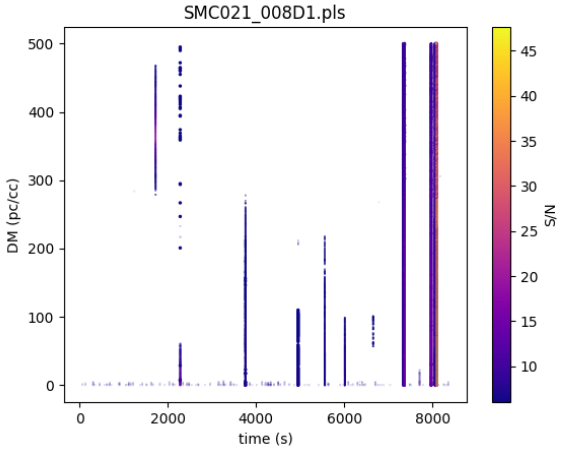


Figure 3: Single pulse search results for Beam D

The signal can be seen in the results of the pulse search for beam D, the leftmost series of hits seen from around 300-450 pc/cc at approximately 1700 s. Zooming in on that part of the timeseries allows one to see why this signal has appeared at multiple DM values.

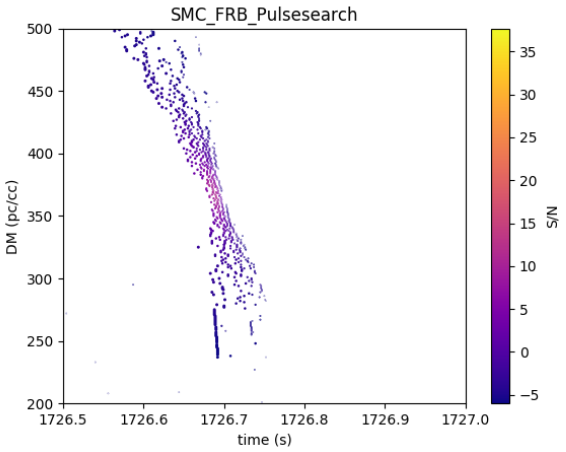


Figure 4: Close up of signal found in single pulse search, see characteristic smearing of signal, both in time and DM.

The S/N is highest at the centre of the hit distribution, here the data is correctly de-dispersed. When the data is incorrectly de-dispersed, the pulse is still bright enough to be detected, but it's detection is smeared in the time direction as a result of the wrong DM value being chosen. Once this signal is seen graphically it is possible to look at the requisite part of the time series in the .pls file, and find the hits with the highest signal to noise. The .pls provides the DM, pulse duration, exact time sample, and SNR. This then allows the pulse itself to be plotted, which was done by decimating and chopping the original filterbank file to create an 8bit filterbank centred on the pulse. Scripts for doing this can be found in my GitHub repository. This was plotted in Python and is illustrated below;

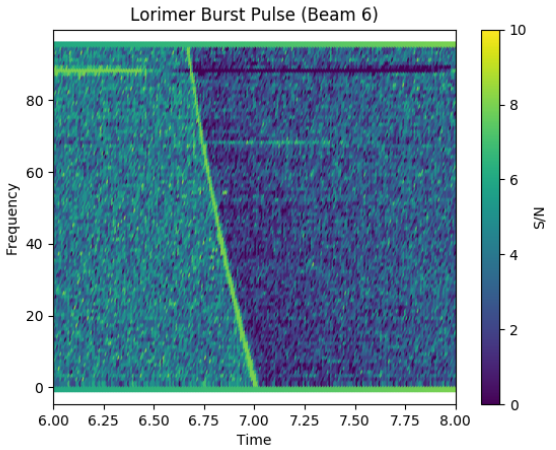


Figure 5: 8 bit representation of Lorimer Burst in Waterfall plot. Note the quadratic drift as a result of dispersion.

This particular plot is from Beam 6, in which the pulse was brightest. In order to determine the exact position of the FRB, the S/N of the pulse in each of the three beams was found in order to triangulate the position based on the layout of the beams as below.

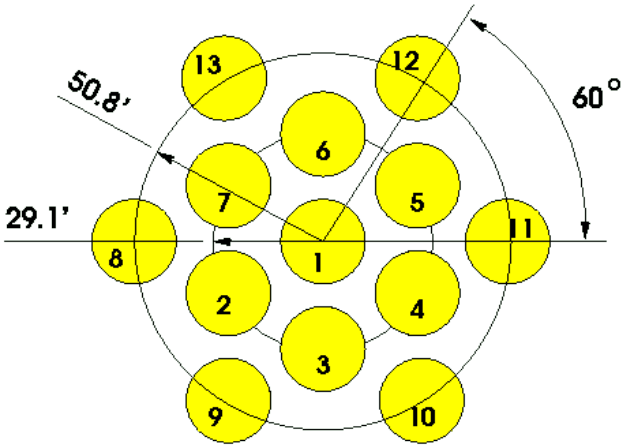


Figure 6: Parkes Observatory beam configuration

Using the radiometer equation, the rms noise level of an observation can be calculated in Jankies;

$$\sigma_\nu = \frac{\beta G T_{sys}}{\sqrt{n_{pol} * BW * t_{obs}}}$$

where $\beta$ is the digitisation loss factor. In the case of this Parkes data, the data is 1 bit digitised which means that approximately $80\%$ of the information is retained. Thus the digitisation loss factor is $1/0.8 = 1.25$. G is the gain of the telescope, in this case the gain and FWHM of each beam are detailed below;

- Beam 1: 0.735 K/Jy, 14.0 arcmin
- Beams 2-7: 0.690 K/Jy, 14.1 arcmin
- Beams 8-13: 0.581 K/Jy, 14.5 arcmin

Note, the gain is sometimes also stated in units of $Jy/K$. $T_sys$ is the system temperature which on average is about 30 K on average [7], $n_{pol}$ is the number of polarisations, in this case 2, BW is the bandwidth and $t_{obs}$ is the length of the observation, in this case the duration of the pulse. One needs only to multiply the RMS noise level by the S/N of a pulse then to obtain the peak flux density in Janksies. For the three beams mentioned above these are;

$$S_6 = 0.4438\ Jy$$
$$S_7 = 0.1902\ Jy$$
$$S_D = 0.3170\ Jy$$

The fluence is the integral of $S\ dt$

$$S_6\ dt = 5.3256\ Jy\ ms$$
$$S_7\ dt = 2.2824\ Jy\ ms$$
$$S_8\ dt = 3.8040\ Jy\ ms$$

As 1Jy= $10^{-26}\ W\ m^{-2}\ Hz^{-1}$, this can be integrated over time and frequency to obtain the radiant exposure $H_e$

$$H_{e,6} = 1.533 \times 10^{-20}\ J\ m^{-2}$$
$$H_{e,7} = 0.657 \times 10^{-20}\ J\ m^{-2}$$
$$H_{e,D} = 1.095 \times 10^{-20}\ J\ m^{-2}$$

According to the original paper on the Lorimer Burst [5], the burst originated from no more than 1Gpc distant, we take that as our maximum distance here for calculating a lower limit on the energy of the burst. WORK HERE

## 2.1 Triangulation

In order to triangulate the exact position of the burst, a python script was written, iterate over best guesses of the true on-axis S/N of the burst and make a plot of what distance the burst must be from each beam to fulfil this, until those distances match up.

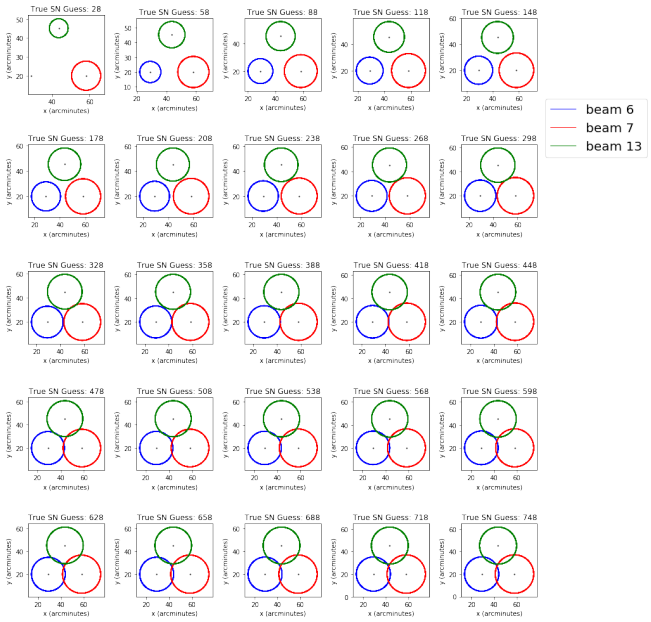


Figure 7: Results of python script for triangulating

This script requires as input simply the S/N of the signal in the beams in question, their FWHM (assuming the beam response is gaussian) and the position of the mean, in this case 0. The script then models the beam response as a gaussian, and calculates the requisite value of x to satisfy;

$$S/N = S/N_{max} \exp \frac{-(x - x_0)^2}{2sigma^2}$$

where $x_0$ is the mean and $\sigma$ is the standard deviation, calculated from the FWHM. It does this for each beam and then plots the beams with a circle of radius $x$ centred on the beam's axis. $S/N_{max}$ is at first taken to be the highest $S/N$ of the 3 beams. It then increases the guess of the true on-axis S/N and produces the same plot. This is repeated until the circles around each beam meet at a common point, this is then the position of the signals origin. If we analyse the plots we can see that for a guess of 748 the area between the 3 circles is becoming very small. Pushing this a little further gives the outcome in Figure 8, from which we can then estimate a value for the true S/N, which can be used to manually compute the position of the source. I'm hoping to add this functionality into the script at some point in the future.

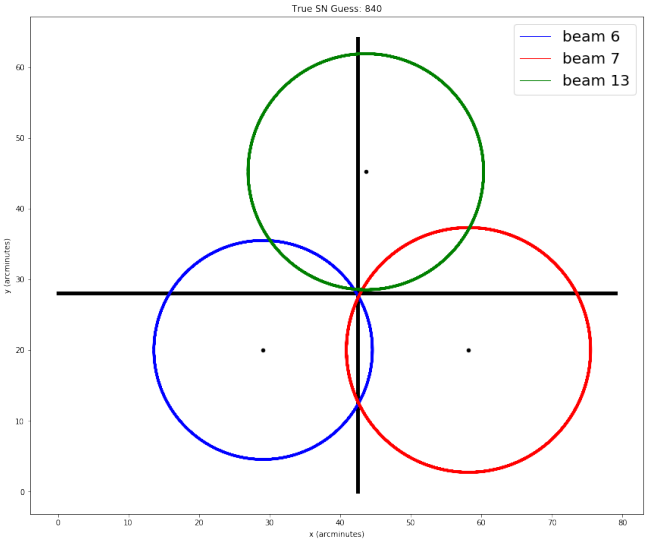


Figure 8: Results of python script for triangulating

Trying to push beyond this point seemed arbitrary to me, as finite line widths will limit the accuracy anyway. So using a value of $S/N_{max} = 840$ the position of the source of the burst is calculated to be approximately 28 arcminutes above beam 6 and 42.5 arcminutes to the right of it.

# 3 Pulsar Observations

Data from the first pulsar observation carried out with IE613 in Birr was now processed in order to identify the pulsar in question, measure it's associated parameters and look at the pulse profile. This data came in the form of a single filterbank file, which was de-dispersed for a much smaller range of dispersion measure trials, 0-30, with a step size of $\Delta d = 0.003$ pc/cc., as pulsars are usually galactic in origin. A single pulse search was then carried out as before for each dispersion measure. However, as pulsars are of course by their very nature periodic, a periodicity search was also carried out. Both are plotted below;

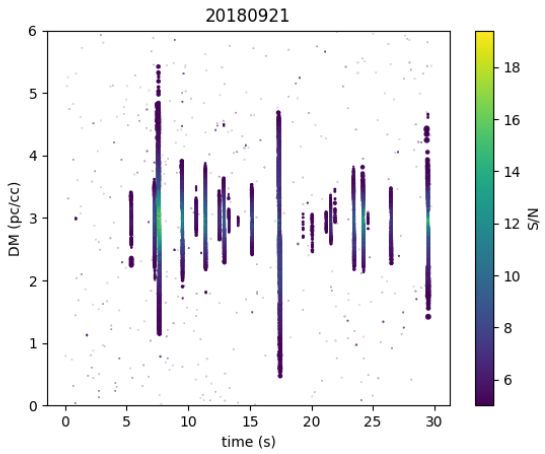


Figure 9: Results of sigproc's single pulse search

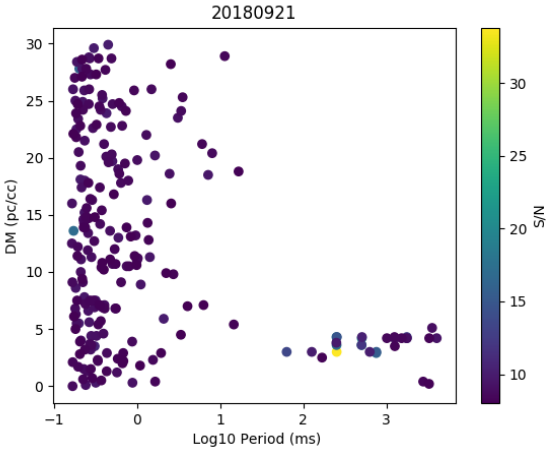


Figure 10: Results of sigproc's periodicity search

The single pulse search is the less informative of the two, it does reveal that the pulsar has a DM of around 3.0 pc/cc. The periodicity search produces a .prd file which contains hits detailing a S/N and period, for 5 different harmonic folds. Sigproc also ships with a function called *Best* which searches a .prd file and selects the most promising results, eliminating low S/N hits and hits across harmonic folds that are probably the same signal. This then produces a .lis file which contains the top hits. This is the file that was plotted for analysis. As is clear from the graph, the highest signal to noise was observed around $10^{2.4} = 251.19$ ms. Manual analysis reveals a signal of period 253 ms and $DM = 3.0$. Using the ATNF Pulsar Catalogue v1.63 the pulsar was thus identified as B0950+08.

Knowing the period of the pulsar allowed for the data to be folded at this period in order to produce a folded pulse profile.

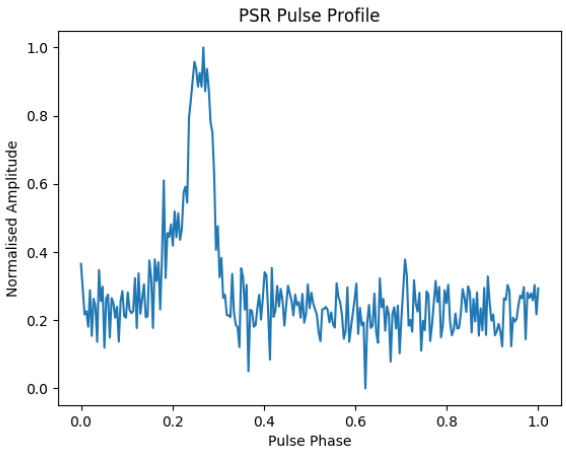


Figure 11: Folded pulse profile of B0950+08

# 4 Crab Pulsar Observations

Data taken in 2017 from a pointing of the Crab Nebula was searched for pulses from the Crab Pulsar and the 30 brightest pulses were kept on file. This data was stored in a raw format, with each pulse observation consisting of 4 .rawudp files; the real and imaginary components of the X and Y polarisations. These were formed into Stokes data using python and saved as filterbank files for use with Sigproc. I tried to incoherently de-disperse the data first in python before saving to file, this was done by calculating the time delay relative to the highest frequency channel of every other channel, using the formula;

$$t_{delay} = \frac{4148 \times DM}{{f_{MHz}}^2} s$$

and then scaling the delays so that $t_{delay}$ of the highest frequency channel was equal to zero. The value used for the *DM* above was the generally accepted value for the Dispersion measure of the Crab; $DM = 56.77$ pc/cc. However, trying to de-disperse the data in such a fashion proved extremely difficult as it was difficult to rearrange the arrays properly to facilitate this. Instead, I decided to just save the file in filterbank format and de-disperse using Sigproc instead. Sigproc requires a header which was formed and attached to the .fil file using mockHeader [8]. Sigproc's *Dedisperse* utility was also unsuccessful, it failed to de-disperse the file properly, which I think is because it couldn't handle a time delay of this size. (16s in this case). Instead, the file had to first be decimated in the time axis, by a factor of 4, thus keeping enough time resolution to see the pulses in detail, but allowing Sigproc to de-disperse successfully.

Sigproc's *dedisperse* utility then formed timeseries which were plotted and are illustrated below.

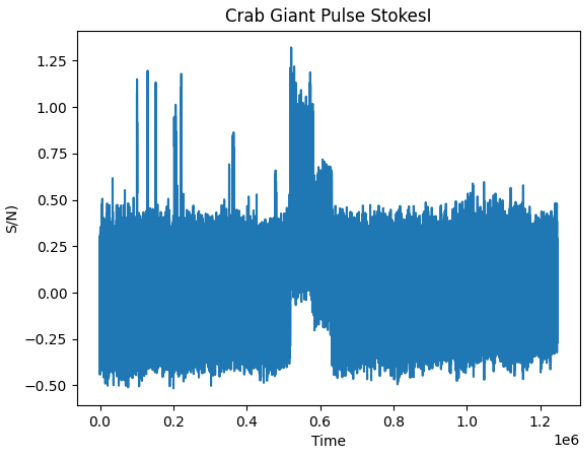


Figure 12: Raw Timeseries

As there was too much noise in this timeseries to spot any pulses, a bandpass of the data was formed, which involves collapsing the data in the time direction and plotting Intensity vs frequency where intensity has arbitrary units. This allowed channels which were over-saturated with RFI to be determined and removed. The bandpass before and after removal of saturated channels is illustrated below.

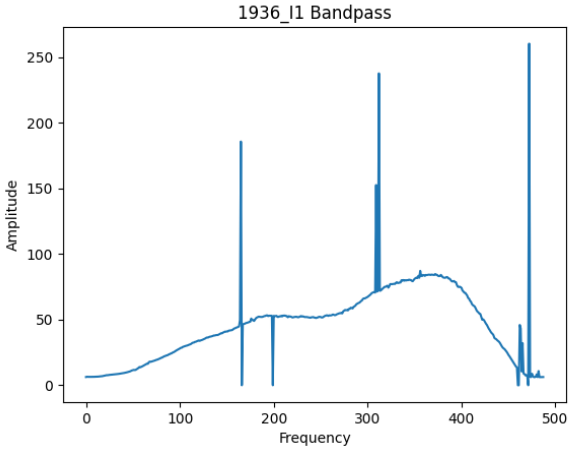


Figure 13: Original Bandpass

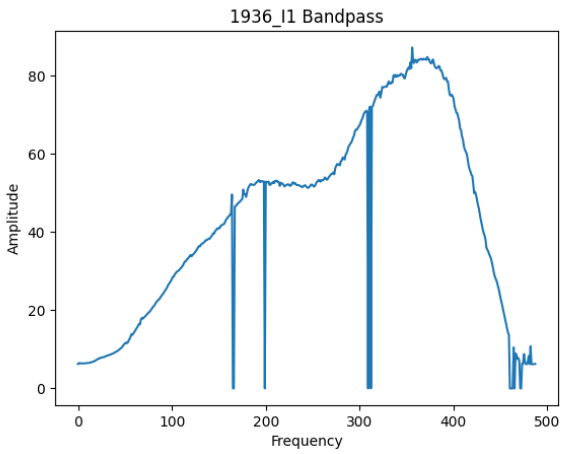


Figure 14: Bandpass once rfi channels have been blitzed

Once the most noisy RFI channels had been removed, the timeseries could be searched for giant crab pulses. By decimating this timeseries and increasing the dynamic range, it was possible to identify 3 giant crab pulses, as shown below. These time series have been decimated by a factor of 32.

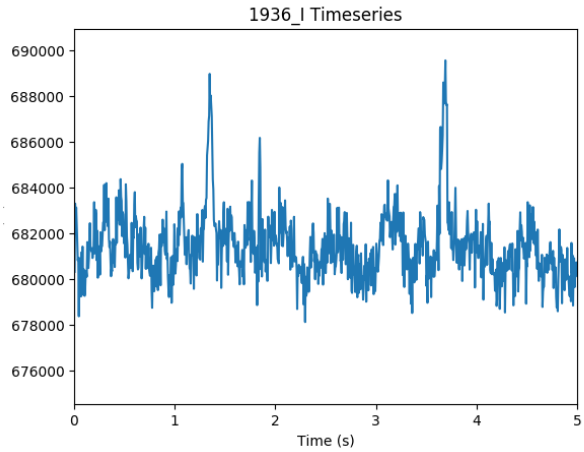


Figure 15: Two Giant crab pulses, note the y-axis is in arbitraty units of power.

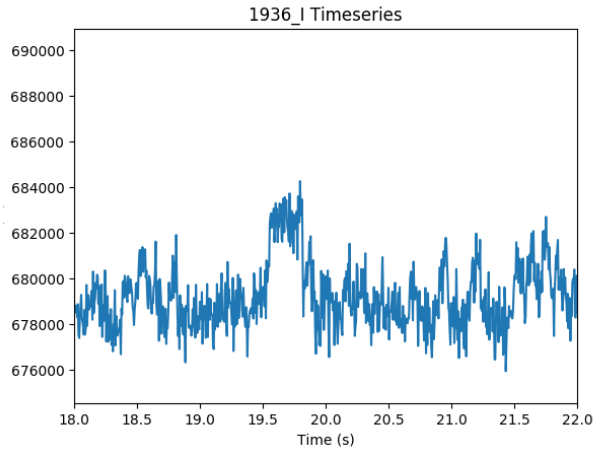


Figure 16: A third giant crab pulse

Unfortunately, it was not possible to calculate the brightness of these pulses in Janskies as for the Lorimer Burst as the the data here contains the background emission of the Crab Nebula.

### 4.1 Coherent Dedispersion

As some pulses could not be detected using incoherent dedispersion methods, it was then attempted to coherently de-disperse the data. This was implemented utilising a script published by Griffin Foster in a notebook containing the code developed for the analysis of the FRB180301 detection [9]. Coherent de-dispersion involves recovering the intrinsic voltage signal as it originated from the source $v_{int})(t)$ from the observed voltage signal $v(t)$. This is possible as the modification of the propagating signal through the ISM/IGM can be modelled as a 'phase-only' filter, or *transfer function, H*. [1]. The idea is then to determine this transfer funtion H for a given source and apply it's inverse to the observed signal. (See Handbook of Pulsar Astronomy (Duncan Lorimer and Michael Kramer) for more detailed theory.)

$$V_{int}(f) = V(f)H^{-1}(f0 + f)$$

Here, $V_{int}(f)$ and $V(f)$ are the Fourier transforms of the raw voltages. This operation is performed on both X and Y polarisations and the results are added quadratically to form StokesI by;

$$I = |X^2| + |Y^2|$$

This was implemented in python by

- Computing a discretely sampled chirp function for n samples,
- Fourier transform the n samples and then multiplying the result by the chirp function, element-wise
- Perform an inverse fourier transform to return to the time domain
- Ignore the 'padded' values at the start and beginning of the time series [1]
- Repeat for the next n samples

If $n_{dm}$ time samples corresponds to the time delay across the entire bandwidth, one needs to pad the data with at least $n_{dm}/2$ samples of 'padding' either end of the data stream to de-disperse correctly [1]. These points, the 'wings' at either end of the data, can then be ignored after the convolution. Once this was performed on every subband, stokesI was formed as above. Unfortunately, this did not quite work, and more work is needed on that script, which can be found in my GitHib repository.

## 5 turboSETI at I-LOFAR

Breakthrough Listen is an initiative by the Breakthrough Foundation aimed at searching for ETI, extraterrestrial intelligence. The project uses observations in the radio spectrum from Greenbank Observatory [10] and Parkes Observatory [11] in order to search for narrow-band drifting signals.

The premise behind these searches is the idea that if ETI exists it may be using radio technology to communicate, the leakage of which could be detectable, or that ETI is knowingly blasting radio signals into space with the hope that someone sees them. Radio emission in nature tends to be broadband and artificial signals are generally narrow-band and so BL's search focuses on narrow-band signals. Since the 1960 we have been able to produce signals with bandwidth's on the order of a single Hz, and so any searches would need to look for very narrow signals. We are searching for signals which 'drift' in frequency as a radio transmitter which is accelerating with a non-zero radial component with respect to a receiver will produce a signal that appears to change its frequency over time. [12] This is usually not an issue with radio data as the frequency resolutions usually utilised are too large for these drifts to be seen. Thus, turboSETI is an algorithm which searches radio data for drifting narrow-band signals which could resemble a signal from ETI.

RFI is a monumental issue when it comes for narrow-band signals, and turboSETI is designed to rule out as much RFI as possible. Primarily, terrestrial signals are unlikely to have a drift rate, and so 0 drift signals can be ruled out, this can be done automatically using the algorithm. Furthermore, observations are made in cadences, where the telescopes toggles between being 'on source' and 'off source', usually in a ABACAD or ABABAB pattern, where A is 'on source' and B, C and D are 'off

source'. If turboSETI detects the same signal in on and off cadences, this can then also be ruled out.

In order to begin utilising turboSETI to search data taken at I-LOFAR it was first installed on the BL compute node at Birr, blc00. Using this install data downloaded from the BL Open Data archive was searched for drifting narrow-band signals. The first track of data which was searched was taken from a diagnostic observation of Oumuamua, the first interstellar object known to have passed through the Solar system [13]. There are generally three different data products produced for Breakthrough Listen from raw data as detailed in Lebofsky et al. (2019). These are a high spectral resolution data product which has frequency resolution of approximately $3Hz$, a high time resolution data for pulsar searches which has a time resolution of approximately $348\mu s$, as well as an intermediate product. For SETI, the high spectral resolution product is most useful. Example scripts for searching this data can again be found in my GitHub repository.

A turboSETI search essentially involves;

- Loading the data into Python using Blimpy's Waterfall() class.
- Searching the data for drifting narrow-band using the FindDoppler() class, which searches within a drift-rate range for signals above a defined S/N threshold.
- The output is .dat file detailing which hits were found.
- In order to filter these hits into 'events', the find_events () class is used, which compares observations in a cadence and rejects them based on a S/N threshold, as well as on which cadences a hit appears in.
- The resulting CSV file contains any events found and can be plotted using the plot_event_pipeline (). These are then processed manually.

This was done for high resolution data from Oumuamua, which formed a 6 part cadence. However, using a S/N threshold of 10 and searching in the range $\pm 4\ Hz\ s^{-1}$, there were no non-zero drift rate signals detected. Some interesting RFI is illustrated below, the source probably being US geo-location of some sort. (The data is from the Green Bank Observatory).

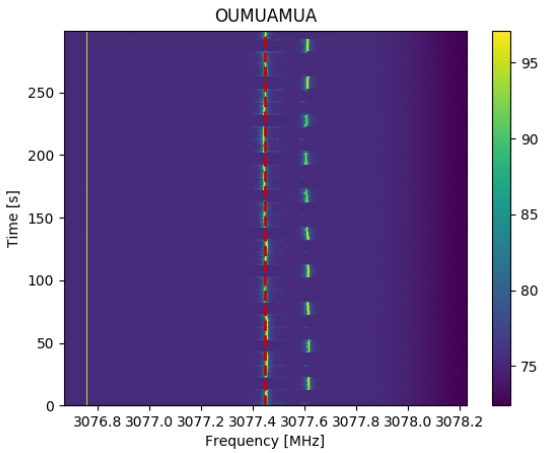


Figure 17: RFI from Doppler Drift search of Oumuamua data

Next, in order to test the find_event () class, data from observations of Trappist1 was searched for narrow-band drifting signals. This time, non-zero drift signals were found and filtered using the find_event () class and plotted for manual inspection using the plot_event_pipeline (). The find_event () class has 3 separate filter thresholds;

- Filter Threshold 1: Returns hits above the SNR cut, taking into account the check_zero_drift parameter, but without an on-source/off-source check.
- Filter Threshold 2: Returns hits that passed level 1 AND that are in at least one on-source but no off-sources.
- Filter Threshold 3: Returns events that passed level 2 AND that are present in ALL on-sources.

The hits found in Trappist1 data were filtered using this class with filter threshold 3; SN cutoff of 10 and not checking zero-drift hits. Various events were returned and plotted. One such example is plotted below.

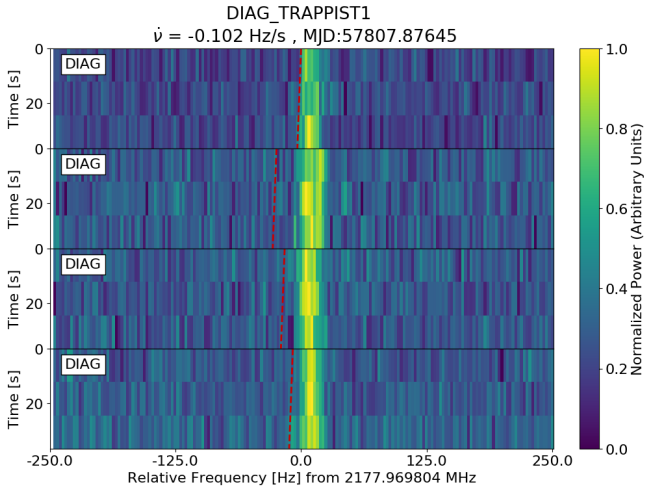


Figure 18: RFI from Doppler Drift search of Trappist1 data

Above is a clear example of how RFI can be ruled out, a signal appears in all 4 cadences and is thus classed as RFI.

Having gotten turboSETI working on the Breakthrough Listen compute node at I-LOFAR, the next step was to make the Breakthrough Listen data products from existing data in order to figure out what parameters are needed. See Lebofsky et al. (2019) [14] for more information on the BL data products. Three data products are produced which vary in their frequency and time resolutions. Of course, the frequencies used at LOFAR are far lower than those utilised at Green Bank and Parkes, and so the drift rates searched would need to be different too, which leads to the need for different data products. The reason for this, is that drift rate is proportional to the rest frequency of the signal. This for LOFAR, where the frequencies are approximately 13 times lower, the expected drift rates for the same targets would be 13 times smaller.

$$\dot{f} = \frac{dv_r}{dt}\frac{f_{rest}}{c}$$

According to Sheikh et al. [12], drift searches up to 200 Hz/s at 1 GHz are physically motivated, which amount to a drift rate of around 15 Hz/s at LOFAR frequencies. For practical/computing reasons, SETI searches like Price et al. (2019) [15] have used drift rate ranges of $\pm 4 Hz/s$.

## 5.1 I-LOFAR SETI Pipeline

So in order to carry out SETI with I-LOFAR, some observations of targets suggested by a Breakthrough Listen - TESS collaboration were carried out. The raw voltages were extracted and re-ordered using the lofar_udp_extractor, which is installed in the aforementioned docker container. This data was saved in .dada format with headers supplied as according to the standard DSPSR header format. [16]

DSPSR's Digifil was then utilised to perform channelisation of the data and form Stokes in the form of a standard filterbank. Data was channelised to obtain a spectral resolution of approximately 3 Hz. Frequency resolution was chosen to obtain the requisite drift rate resoltution which is simply the spectral resolution divided by the observation time, in this case 5 minutes. Once these filterbank files were created, they were searched using turboSETI on the blc00 compute node. As we were only able to test the pipeline on a single observation, we did not have a cadence and so were unable to filter hits into events. Illustrated below is a histogram of the S/N of hits.

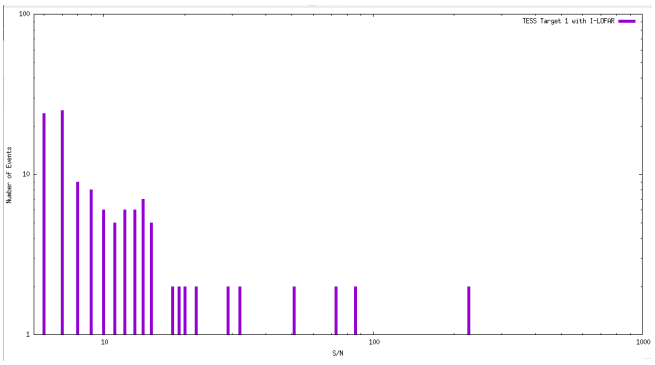


Figure 19: Histogram of S/N of hits found during testing of I-LOFAR SETI pipeline

For I-LOFAR, the noise floor was calculated to be approximately 6. In reality however, it appears to be around 10, presumably due to our noise not being strictly gaussian. In the future of course, it would be helpful to streamline the pipeline by installing the I-LOFAR docker container on blc00, as well has performing observations in cadences. In the process of testing this pipeline, some issues and bugs were found in both turboSETI and Digifil and will be reported in the near future. It would also be helpful to perform SETI searches for higher drift rates in the future, the pipeline was tested for drift values in the range $\pm 4$ Hz/s.

## References


[1] D. R. Lorimer and M. Kramer, *Handbook of pulsar astronomy*. University Press, 2012.

[2] D. Lorimer, "Sigproc–v3.7 : (pulsar) signal processing programs," march 2006.

[3] M. J. Keith, "Installation and use of pulsar search software," 2012.

[4] D. McKenna, "David-mckenna/udppacketmanager."

[5] D. R. Lorimer, M. Bailes, M. A. McLaughlin, D. J. Narkevic, and F. Crawford, "A bright millisecond radio burst of extragalactic origin," *Science*, vol. 318, p. 777–780, Nov 2007.

[6] J. M. Cordes and T. J. W. Lazio, "Ne2001.i. a new model for the galactic distribution of free electrons and its fluctuations," 2002.

[7] L. Staveley-Smith, W. E. Wilson, T. S. Bird, M. J. Disney, R. D. Ekers, K. C. Freeman, R. F. Haynes, M. W. Sinclair, R. A. Vaile, R. L. Webster, and A. E. Wright, "The Parkes 21 CM multibeam receiver," , vol. 13, pp. 243–248, Nov. 1996.

[8] E. O'Cathain, *mockHeader*, (accessed August 28, 2020).

[9] G. Foster, "dust: Calculate the intensity of dust scattering halos in the X-ray," Mar. 2015.

[10] "Green bank observatory home," Oct 2019.

[11] "Welcome to the csiro parkes observatory."

[12] S. Z. Sheikh, J. T. Wright, A. Siemion, and J. E. Enriquez, "Choosing a maximum drift rate in a seti search: Astrophysical considerations," *The Astrophysical Journal*, vol. 884, p. 14, Oct 2019.

[13] . Paul VoosenNov. 20, . Leslie RobertsAug. 17, E. N. Niina H. Farah, . Kelly ServickAug. 14, . David GrimmAug. 14, . Erik StokstadAug. 13, . Rasha AridiAug. 17, . Lucy HicksAug. 12, . Lucy HicksJul. 31, . Charlotte HartleyJul. 30, and et al., "Updated: For the first time, astronomers are tracking a distant visitor streaking through our solar system," Dec 2017.

[14] M. Lebofsky, S. Croft, A. P. V. Siemion, D. C. Price, J. E. Enriquez, H. Isaacson, D. H. E. MacMahon, D. Anderson, B. Brzycki, J. Cobb, and et al., "The breakthrough listen search for intelligent life: Public data, formats, reduction, and archiving," *Publications of the Astronomical Society of the Pacific*, vol. 131, p. 124505, Nov 2019.

[15] D. C. Price, J. E. Enriquez, B. Brzycki, S. Croft, D. Czech, D. DeBoer, J. DeMarines, G. Foster, V. Gajjar, N. Gizani, and et al., "The breakthrough listen search for intelligent life: Observations of 1327 nearby stars over 1.10–3.45 ghz," *The Astronomical Journal*, vol. 159, p. 86, Feb 2020.

[16] "Dspsr documentation."

# 6 Appendix: Example Recording Script

```
#! bin/bash

#generic startup sequence
echo 'Initialising: SWLEVEL 2'
eval swlevel 2
rspctl --wg=0
sleep 1
rspctl --rcuprsg=0
sleep 1
rspctl --bitmode=8
rspctl --bitmode
sleep 1
killall beamctl
sleep 3
echo 'Initialising: SWLEVEL 3'
eval swlevel 3
sleep 2
rspctl --splitter=0
sleep 3
rcus='0:83,86:191'

#generic pointing block
#Insert the RA and DEC of your source
pointing='0.602601133,0.74243089,J2000'
beamctl --antennaset=HBA_JOINED
--rcus=$rcus --band=110_190 #this is the requisite band
--beamlets=0:487 --subbands=12:499
--anadir=$pointing --digdir=$pointing &
bash sleepuntil.sh 20200722 035500
killall -9 beamctl

#generic reporting block
obs_start='2020-07-22T04:00:00.0'
obs_end='2020-07-22T04:55:00.0'
source='J0218+4232' # No Spaces
bash sleepuntil.sh 20200722 035800
# Try to set it 1-2 minutes before the observation starts
echo $source $obs_start $obs_end
logdate=`date +'%Y%m%d%H%M%S'`
bash generic_ucc1_timestamps.sh $obs_start $obs_end 'dump_udp_ow_12' $source 2>&1
```