Using Datamining Techniques for Single Burst Pulsar Recognition

Steijn Kistemaker, Arjan Nusselder, Nadya Peek
{skistema, anussel, npeek}@science.uva.nl
Universiteit van Amsterdam

February 4, 2008

Abstract

Pulsars are very dense rotating radiating stars, which have only recently been discovered in our galaxy. Automatic data processing techniques for detecting pulsars are still in development. This paper describes an algorithm for detecting irregular pulsars which do not easily show up in detection techniques relying on the periodicity of a pulsar. It does this by analyzing single bursts based on various features. We have tested it with the GBT350 data, and have found some promising results.

Contents

1. Introduction

A new distributed astronomical data collecting system is being implemented through the Netherlands, Germany, Great Britain and France which will collect up to 1 square kilometer of radio-frequency (wavelengths of around 1 meter) data. The project is called LOFAR, for Low Frequency Array, and is a interferometric array of radio telescopes which will be particularly suited for finding the radio signals transmitted by pulsars, or pulsating stars [5].

Pulsars are very interesting to astrophysicists because of the extreme conditions they create. They are highly magnetized collapsed massive stars (neutro stars) which emit electromagnetic radiation while rotating, sometimes as quickly as once every couple of milliseconds. When the axis of rotation is orthogonal to our line of sight, we can observe the electromagnetic radiation in the form of periodic radio waves, similarly to how we see a blinking light from a lighthouse. Pulsars have such strong gravitational forces that gravitational radiation, which was predicted but not observed during the conception of general relativity, can be observed and studied [3].

So far, pulsars have been detected using very large radio telescopes such as the Green Bank Telescope in West Virginia. The amounts of data produced by readings from the GBT were small enough that astronomers would plot the data they gathered and detect pulsars by eye. So far, some 1800 pulsars have been detected, but due to the current positioning of radio telescopes, most of those pulsars

have been detected in one particular quadrant of the galaxy. Once the LOFAR is fully operational however, there will be a lot of data from other quadrants, too much to be able to have a human look at it all. To be able to aid the detection of pulsars, it has become very interesting to develop some data-processing methods which will filter data from the radio telescopes and preliminarily detect possible pulsars.

In this paper, we will present results from a pilot project in automated single-burst pulsar detection. We used data from the GBT and developed algorithms to separate astronomical signals from noise, and used the algorithms to present the user with plots of data which contain possible pulsars. In section 2 we will explain some of the attributes of pulsars. In section 3 we will explain features of the radio data we are aiming to analyze. In sections 4 and 5 we will explain some techniques for finding pulsars, and in section 6 we will outline our approach to detection. The technical implementation is detailed in section 7, and finally you can see the results of our algorithm and a conclusion in sections 8 and 9.

2. Aspects of Pulsars

There are several different kinds of pulsars: rotation-powered pulsars, accretion-powered pulsars and magnetars. Radiation is powered through the loss of rotational energy in the first, through gravitational potential energy in the second and through decay of magnetic field in the third.

The distance from the earth to the pulsars is measured by *dispersion measure* or DM. Dispersion measure is not an absolute distance, but uses the dispersion of the signal to calculate how much lays between the source of the signal and the observer. It can be seen as a quantification of the amount of electrons between the observer and the source. We are of course most interested in finding pulsars very close to earth, or with a very low DM (<100).

However, the closer we are to the earth, the more interference we will find in our radio frequency data. This is commonly referred to as RFI, or radio frequency interference, and is not only created by radio signals on earth, but also by satellites, Aurora Borealis, the Sun, and many other sources.

The signals that pulsars emit can be very weak, and therefore can be easily lost in RFI. Their periodicity is also not necessarily predictable: while some pulsars may pulse every couple of milliseconds, other pulsars may only emit single bursts every 100 seconds, or even emit bursts very irregularly. Collecting enough data to find the periodicity of the irregularly or infrequently pulsating pulsars is difficult, and the subsequent detection of such single-burst pulsars is hard to automate using currently popular techniques such as Fourier analysis.

3. The GBT350 Data

For the our automated pulsar detection methods, we analyzed radio data collected in the GBT350 survey [2]. The data covers approximately 1000 square degrees of sky visible from the Northern Hemisphere, and is separated into single pointings, each approximately 0.6 degrees in diameter. The observing frequency of the data is between 325 and 375 MHz, and each pointing was observed for 120 seconds. The observations have been made between 0 and 1500 DM, but we will only be analyzing up to 300 DM.

When observing a pointing with a beam of 0.6 degrees, it is still possible to observe pulsars which are some distance away (up to 3 degrees, depending on the strength and distance of the pulsar). Because of this we could calculate which pulsars we could possibly see in which pointings, and locate known pulsars in the data. Using the characteristics of the signals made by the known pulsars, we will attempt to find new pulsars.

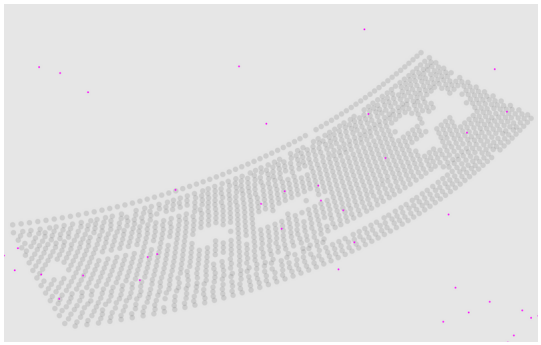

Figure 1: The GBT350 pointings (grey) and the known pulsars in the area (pink). There are some gaps in the data, but there is no space between the pointings.

4. Known Pulsar Detection Methods

Because many pulsars have periodic signals, they are easily detected using a discrete Fourier transform, which separates a discrete signal into various harmonic functions [1]. Even with a lot of RFI, discrete Fourier transforms are very good at detecting periodic signals. There has been a reasonable

amount of work done in the automated detection of periodic radio signals, for instance in the PRESTO suite [4], and many of the pulsars which have already been detected in the GBT350 data have been done using these techniques.

5. Detecting Irregular or Slow Pulsars

In this project, we will focus on finding *single-burst pulsars*. This name is slightly misleading, because we are aiming to detect pulsars which only show one burst during the time we observed that area of the sky, which is only for 2 minutes. As pulsars, they of course will be pulsing more than once. To be able to do this, we will examine bursts found in our data, and attempt to find bursts which we think might come from a pulsar.

There are typical characteristics of bursts which make them more likely to originate in a pulsar. We will be looking for the following attributes for each burst:

- The signal to noise ratio should be higher than the mean of signal to noise ratios taken over a DM range.
- The burst should show strongly in one DM and have a diminishing effect on the DMs surrounding it.

There are also typical characteristics of RFI signals, which we could use to filter out bursts likely to be interference. Especially characteristics pertaining to the strength of neighboring bursts in surrounding dispersion measures can be telling for RFI. To be able to adequately filter out the RFI, we would need to have a more precise description of these features, especially in how they can be distinguished from pulsar bursts.

6. Approach

We propose to do a multiple pass analysis on the data. Firstly, probable RFI and other noise should be removed from the data (per single pointing). Then the remaining data should be analyzed per candidate burst, where the burst context is taken into account when scoring. Each burst then receives a score based on the criteria specified in section 7, a combination of which will define its final 'probable pulse' score. This score can easily be thresholded to present candidate pulsars.

The criteria on which to base the scoring however are not uniform. Ideally, the best net influence of each criterium (or combination of criteria) could be learned automatically. One possibility is to bootstrap the variables based on known and clearly visible pulsars, iteratively revising them to approach a good estimate. The problem with this is that 'single pulse' pulsars are explicitly hard to distuingish, and might not be described best by 'clear pulsars' at all. Therefore we will attempt a more algorithmic approach for single burst recognition here. However, it would be very interesting to automatically learn features of pulsars, and some ideas regarding that are outlined in section 9.

6.1. Outline

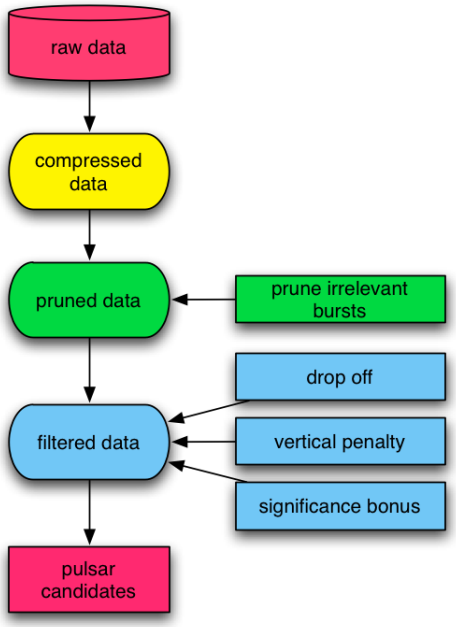

Figure 2: System overview

For the time being, we propose 3 different filters (figure 2) over the pruned data which we will combine to produce a score per dispersion measure (DM). The first, *dropoff* should analyze the shape of the burst and determine whether it looks like astronomical data or noise. The second, *vertical penalty* should determine how likely a burst is really just noise based on its neighboring bursts. Finally, the significance bonus should reward statistically very unlikely bursts in their scores.

6.2. Main algorithm

```
scorePulsar(data)
1  cleandata = removeRFI(data)
2  candidates = findCandidates(cleandata)
3  for all DM
4        do score =
             scoreDM(dm,candidates,cleandata)
5  return score-vector
```

We propose to confine the filters to a function called scoreDM, which will be called individually for each dispersion measure. The dispersion measure ranking highest in a pointing will be returned as the most likely DM to find a pulsar. The processes before the scoring of the DMs will simply attempt to remove as much noise as possible. Section 7 will give more details on how each DM score is built up from various filters.

---

7. Multipass filter, technical implementation

This section is divided in three parts. The removal of noise, the explanation of specific filters and the final score.

7.1. RFI

As a way of cleaning our datasets we intended to remove noise from it. Removing this noise should make it easier to detect pulsars, especialy weak ones, and make it easier to see them by eye when plotted. Noise in the datasets can be subdivided into two types, being random noise and RFI (or radio-frequency interference). Because random noise can not be modelled we decided to focus on RFI. RFI can show up in all DMs and can be characterized by very strong signals of about the same signal-to-noise ratio showing up over a range of DMs on small time-range. In the plots of time vs DM (vs signal-to-noise) RFI shows up as vertical collumns.

Removing RFI from the datasets proved to be quite difficult. Although RFI can be quite easily distinguished by eye it is somewhat harder to do this in an algorithmic approach. We noticed that RFI shows similar characteristics as pulsar bursts. The diamond-shape characteristic to pulsar bursts also shows up in RFI, though RFI has a tendency to spread out over a much larger DM-range and to have a more consistent signal-to-noise ratio (instead of decreasing). Our pulsar burst finding algorithm had a hard time to distinguish between RFI and pulsar bursts, so we decided not to remove this noise (which would cause a lot of pulsar bursts also to be removed). We have left the RFI-removal however as an interface in our algorithm to be implemented later when an expert can make a better characteristic of RFI vs pulsar bursts. We have developed a counter measure in our scoring mechanism however against RFI showing up as good pulsar candidates, detailed in section7.

7.2. Filters

In our algorithm we have used a multi-pass filter system, where likely candidates of pulsar bursts are pruned and eventually a score per DM is calculated. We used a 2d-array with DM and time (in the form of timebins of size 0.1s) as its dimensions. These bins contain bursts with all their characteristics as delivered in the data files. For an overview see codebox 7.3.

Our first-pass filter is a very simple filter used to filter out possible diamond-shape candidates, and looks for local maxima within a time range. It searches for bursts with bursts in the DM above and below it with a smaller signal-to-noise ratio. Bursts with a burst on only one of the adjacent DMs are also approved. Only of these candidates the drop-off is calculated, which is the number of drop-off steps, that can be made from the local-maximum-burst. For this we use a drop-factor which is used to control the decline-rate. The lower this is set, the lower the average drop-off score gets. We use a value of .95 at the moment.

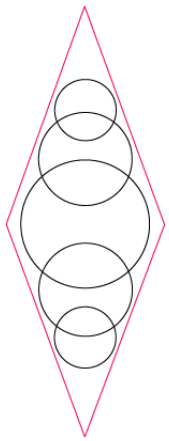

Figure 3: The diamond shape we are looking for which is characteristic to pulsars. First pruning would find the big circle (high sigma) as a candidate, and next assign it a drop-off score of 4.0 for the surround bursts.

In our scoring mechanism a second filter is used,

only assigning scores to bursts with a drop-off-score above a certain threshold. This is used to distinguish between very small diamonds (with a drop-off of 1 or 2), which can be assigned to coincidence, and more probable pulsar bursts with a higher drop-off.

7.3. Scoring

Our scoring mechanism is a combination of scores of certain characteristics determined by experiment (and intuition) that are distinctive for pulsars. The scores are combined as to assign a score to a single burst. These scores are then summed per DM which leads to a score per DM. In post-processing the maximum score of these scores per DM is used as the score for the pointing, and its DM as the most probable DM containing the pulsar. Of course the vector containing the various score of each DM is kept, so any score above a threshold can be examined in a pointing, and it is of course also possible to detect two separate pulsars in one pointing.

For our scoring of a single burst we use three characteristics; drop-off score, signal-to-noise significance and RFI-penalty. Drop-off score and signal-to-noise significance are positive characteristics which increase the score of a burst. *RFI-penalty* (codebox 7.3) is a negative characteristic which lowers the score of a single burst. Again only the scores for the pruned bursts are calculated.

RFIpenalty(*pulse*,*candidates*)

```
1  timebin = pulse.timebin
2  lnum = len(candidates.timebin.left)
3  cnum = len(candidates.timebin)
4  rnum = len(candidates.timebin.right)
5  return (lnum + cnum + rnum)
```

The *drop-off score* is the number of steps that can be made to adjacent DMs following the diamond shape (codebox 7.3). This score is used to score a burst's resemblance to the diamond shape characteristic of pulsars or astronomical signals in general.

getDropOff(*pulse*,*cleandata*)

```
1  dropfactor = 0.95
2  startsig = pulse.sigma
3  sig = startsig
4  apulse = pulse
5  upscore = downscore = 0
6  while (apulse.dmUp.sig < sig · dropfactor)
7        do apulse = apulse.dmUp
8           sig = apulse.sig
9           upscore++
10 sig = startsig
11 apulse = pulse
12 while (apulse.dmDown.sig < sig · dropfactor)
13       do apulse = apulse.dmDown
14          sig = apulse.sig
15          downscore++
16 return upscore + downscore
```

The *signal-to-noise significance* score is used to determine the significance of the burst in the pointing, based on the average and standard deviation of the signal-to-noise in all bursts in a pointing. This targets random noise and RFI at the same time. Random noise will not be significant in a pointing, because its sigma will not deviate significantly from the average. RFI is targeted because it causes a multitude of high signal-to-noise bursts, which increases the average, and causes most of the RFI bursts to be considered non-significant (although they are still assigned a score). A burst is considered significant when it has a probability of <0.001 using the following probability density function:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

When a burst is considered significant it is given a bonus of 10.0 in the score. This score was chosen because drop-off scores range on average from 1 to 10, and significance of a burst should give the score a boost equal to a good diamond shape. A gradient bonus score could also be used, but we thought that this would affect the scores of weak pulsars in a negative way, which is not what we want.

The negative characteristic *RFI-penalty* is used to penalize RFI-noise. Its intuitive interpretation is that a certain burst is less significant (and thus acquires a lower score) when it is found in RFI-noise. The RFI-penalty is the number of candidate bursts in the timebin of a burst (over all DMs) and its two adjacent timebins. This penalty targets RFI-noise directly, because datasets with RFI-noise in it tend to have a much higher number (factor 10 to 100) of candidates. Because of this, bursts in RFI-noise get very low scores.

The formula to calculate the score of a single burst

is given by:

$$score = \frac{(2 \cdot dropoff + sigbonus)}{RFI\text{-}penalty}$$

where 2 is a scaling factor that rates dropoff-score higher than significance bonus.

The pseudocode for the DM scoring is given in codebox 7.3.

scoreDM(*dm*,*candidates*,*cleandata*)

```
1  score = 0
2  for all TimeBins
   where candidates-DM = dm
3        do for all Pulses in TimeBin
4           sigbonus = 0.0
5           if isSignificant(pulse)
6              then sigbonus = 10.0
7           vpenalty =
                 RFIpenalty(pulse,candidates)
8           dropoff =
                 getDropOff(pulse,cleandata)
9           score += (2 · dropoff + sigbonus) / vpenalty
```

As said previously, these scores are summed per DM, and given as an output vector. The maximum score with its DM are used as the score for the pointing. When all datasets are analyzed these pointings are sorted on their score, resulting in likely pulsar candidates showing up high in the list, and less likely candidates showing up lower. A threshold can easily be used on these scores to discard of very unlikely pulsar candidates.

8. Results

For our results we evaluated our algorithm on a list of known pulsars. We wrote a program that would output a list of pointings in our dataset that would contain known pulsars (by calculating which dataset-pointings were close to pointings of known pulsars). Later we noticed that some pulsars show up in multiple pointings, and that pulsars do not show up in all pointings outputted (because the pulsar's DM could be outside our range). We then hand checked the list we had (which had about 40 entries) with the plots to see if a pulsar showed up, and created a list of actual pulsars in our dataset.

When our algorithm started giving likely pulsar candidates we soon constructed a large list of falsely 'new' pulsars, we found out that the angle at which pulsars could show up in our pointings was bigger than we originaly expected, and so we adjusted our angles of expectation from 0.3 deg (which is the radius of a pointing) to 0.6 deg and 1.0 deg. Not all of the pointings had to contain pulsars, so we decided to use this list as a check if our found pulsars already existed or not. We then started working through our list of likely pulsars, and generated a list of pulsars in our dataset. We hand checked all these candidates by looking at the plots, and only counted them as pulsars if they were evident. It is possible we made some true negatives, but we only counted evident pulsars as pulsars so we are quite sure we haven't listed any false positives.

When evaluating our results we did notice a certain sort of ground noise (see figure 4 below) that was rated very highly in our scoring. These scores however were all found at a DM <10. Because we didn't have the knowledge to distinguish this from pulsars we decided to make a separation between the results above 10 DM and below 10 DM. Because of the amount of datasets with this same kind of ground noise, we didn't evaluate them all by hand, and decided to ignore them for the time being. However eliminating these ground noise pointings from our list of good candidates would be a big improvement to the system (see future research in section 9).

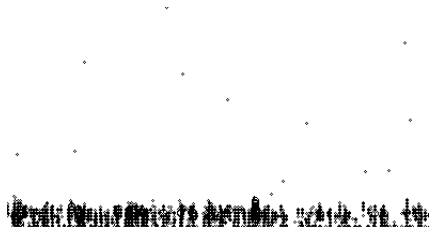

Figure 4: Small part of a plot of Time (X-axis) vs DM (Y-axis) containing characteristic type of ground noise

8.1. Detecting Known Periodic Pulsars

Although our algorithm is intended to find irregular or single-burst pulsars which will not show up using a Fourier transform analysis, it is also very efficient at finding periodic pulsars. Because our list of known pulsars only gives pointings where a pulsar could possibly show up, we cannot use this directly to check if we have found all previously know pulsars. We do know however that there are about 60 known pulsars in the region of our dataset and that we have generated a list of 90 pulsars. The surplus could be explained by the fact that strong pulsars show up in multiple pointings.

Below we can see that the results above 10 DM are very reasonable. We have classified the candidates into three types, being pulsar samples, RFI-noise samples which contain a lot of RFI, and random-noise samples that are reasonably empty, but con-

tain a few higher pulses. We only classified pointings as containing pulsars if they were clear examples, so it is possible that some of the random-noise samples contain (weak) single-burst pulsars. We can see that when the scores drop, the percentage of noise increases. The 154 pointings with a score above 20 (and above 10 DM) however contain 90 pulsars and 64 noise samples of which 47 are in the 20-25 score interval. We think this is a good result.

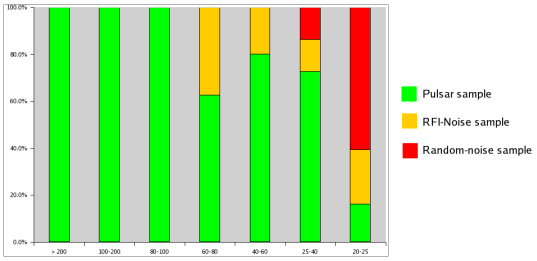

Figure 5: Plot of Noise vs Pulsar ratio (from the results above 10 DM)

8.2. Detecting Known Single-Burst Pulsars

It was hard to test if our program was good at finding single-burst pulsars since we only had a list of 3 known single-burst pulsars (for an example see figure 6). Our program however found all these 3 single-burst pulsars (though one of them was below 10 DM). When evaluating our list of potential pulsars we also found a set of pulsars which pulsed weakly and only 2 to 5 times in 120 seconds which would be hard to find with Fourier analysis. These pulsars aren't single-burst pulsars by definition, but show that our technique of finding pulsars is also successful for weak and irregularly pulsing pulsars.

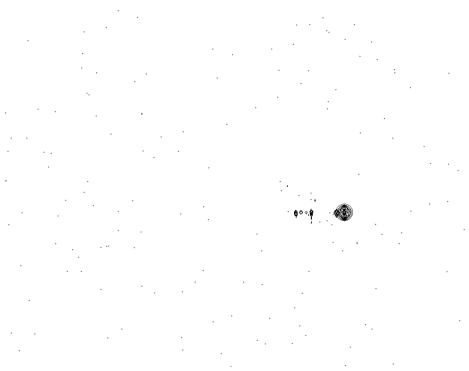

Figure 6: Plot of Time (X-axis) vs DM (Y-axis) containing a single-burst pulsar (in G12380-168)

8.3. Detecting New Pulsars

Since our system is to be used for the detection of new pulsars it must be efficient at finding pulsars. Although our results indicate that our system is indeed efficient at finding pulsars it does not contain an automated method to check if the found pulsars already exist. If this would be done automaticaly our system could be used very efficiently to find new pulsars in a dataset. We think researchers would be more interested in scores between 20 and 30, because the most weak or irregular pulsars (that are otherwise hard to find) tend to show up in this score-range. Also our ordering of scores makes it very easy to work through the list from probable pulsar pointings to less probable ones, saving the trouble of working through all the pointings.

When evaluating our results, we also found a pointing which looked like it might be a (weak) pulsar, and that wasn't yet known. We asked an expert (Dr. J.W.T. Hessels) to evaluate this pulsar for us, but it is yet to be decided if this pointing indeed contains a pulsar. To truly know whether or not there is a pulsar located in that particular pointing, we will have to collect new data from the GBT to see if pulses at the same DM show up again. In figure 7 a plot of this possible pulsar can be seen, with two small bursts at DM 112.

9. Conclusion and Future Research

At the moment we are doing very little preprocessing on the data to remove noise. Especially under 10 DM, this is problematic for our algorithm. At the moment we have no adequate solution for dealing with the noise under 10 DM– when we know beforehand we should be seeing a pulsar below 10 DM, we can run the algorithm on this particular pointing and the pulsar under 10 DM will pop up. If we do this for all pointings however, too many false positives will show up under 10 DM, which will make the subsequent processing of the pulsar candidates tedious.

Because this ground noise that is bothering us below 10 DM seems consistent, it might be possible to learn the various attributes of this noise and filter it out. It would be very interesting to see how many of the various types of noise could be filtered out with learning techniques in general.

The combination of the scoring filters to the final score remains a bit arbitrary, as the relations were only found with trial and error. It would be interesting to introduce perhaps some more filters and test different methods of interaction.

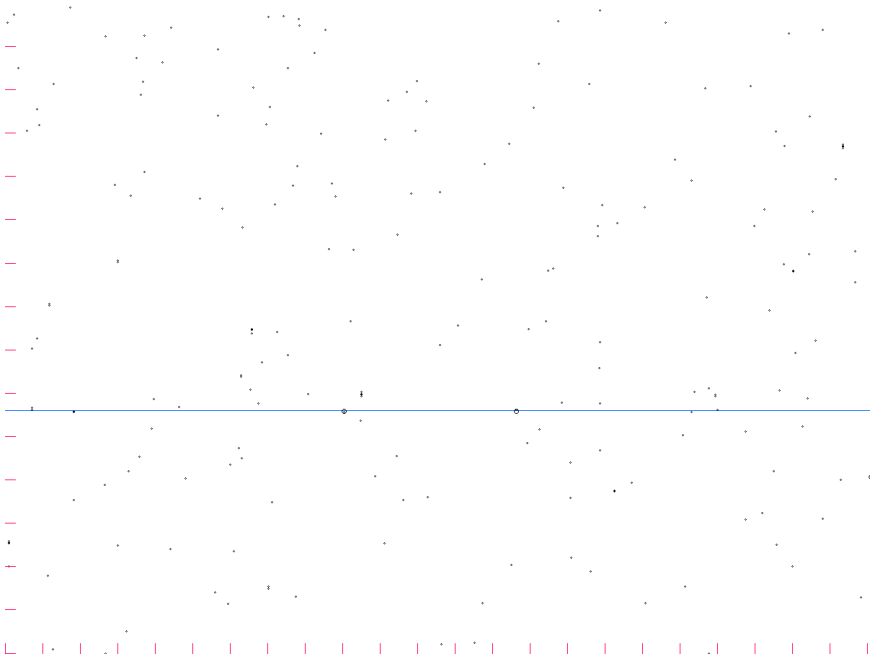

Figure 7: Plot of Time (X-axis) vs DM (Y-axis) with 2 bursts at 112 DM

Finally, it would of course be very useful to provide an interface which would immediately compare the pulsars it would find in pointings to a database of known pulsars, and thus automatically label the pulsars which it thinks it already recognizes. At the moment, the comparison with existing pulsars is done by looking up the coordinates of the pointing and comparing it to the coordinates and DMs of pulsars in the neighborhood. While feasible when dealing with smaller datasets, it would of course be preferable to have the system distinguish between 'new' pulsars and 'old' pulsars.

All in all, we think that these results show some promising possibilities for the automatic detection of single and multi burst pulsars. With some more work on the interface, we think that some of this work could become part of a very useful pulsar finding tool. We also sincerely hope that the plot in figure 7 is indeed a pulsar!

Usage of the algorithm which was implemented in python and C is detailed in appendix A. It takes tar.gzed data from the GBT as input.

References

[1] J. M. Cordes and M. A. McLaughlin. Searches for fast radio transients. *The Astrophysical Journal*, 2003.

[2] J. W. T. Hessels, S. M. Ransom, V. M. Kaspi, M. S. E. Roberts, D. J. Champion, and B. W. Stappers. The GBT350 survey of the Northern Galactic Plane for Radio Pulsars and Transients. In *Proceedings of Forty Years of Pulsars: Millisecond Pulsars, Magnetars and More Conference; Montreal, Canada*. American Institute of Physics, 2007.

[3] M. A. Mclaughlin, A. G. Lyne, D. R. Lorimer, M. Kramer, A. J. Faulkner, R. N. Manchester, J. M. Cordes, F. Camilo, A. Possenti, I. H. Stairs, G. Hobbs, N. D'Amico, M. Burgay, and J. T. O'Brien. Transient radio bursts from rotating neutron stars. *Nature*, 439(7078):817–820.

[4] S. Ransom. *New Search Techniques for Binary Pulsars*. PhD thesis, Harvard University, September 2001.

[5] H. Röttgering, M. van Haarlem, and G. Miley. Lofar - a new low-frequency radio telescope. In P. Williams, C. . Shu, and B. Menard, editors, *IAU Colloq. 199: Probing Galaxies through Quasar Absorption Lines*, pages 381–387, mar 2005.

A. Usage

```
Usage guide for analyzeData.py
N.B. python2.5 is required.

This program serves mostly as a wrapper for the analysis functions. If
you want to view or edit the code of the scoring algorithm, please see:
'../plot/analyze.c'

Assumed as data input are one or more tarfiles ('G*.tar.gz') containing
the folder plus file:
'SINGLE/*.singlepulse'
Files not adhering this naming scheme will not disrupt the analysis of
other files, but will print an error like:
'G13462-084_SINGLE: ERROR: File "ls:": File not found'

There are two basic sequences of operations you might want to follow.
First (1) is a test of all the data with the current analysis, and
plotting/viewing good candidates. Second (2) is a repeated test on some
data-sample with changes in the scoring algorithm.

(1) Commands:
    `python analyzeData.py -nsA <datadir>`
    `python analyzeData.py -nlt <threshold>`
    `python analyzeData.py -nlt <threshold> -PA <datadir>`

    First run `python analyzeData.py -nsA <datadir>` where <datadir> is
    a directory containing the .tar.gz files (' -A <dir> ' can be called
    multiple times to use additional directories). This operation
    analyzes all the data and saves the result for future use. It should
    take approximately 15 to 30 minutes.

    Next, run `python analyzeData.py -nlt <threshold>` to create output
    files with the results over the given score <threshold>. At the time
    of writing, a threshold of 25 looks reasonable for decent
    candidates.
    The output is ordered by score. Five files are created:

    - 'outputOver<threshold>.txt'
      # Contains all data of pointings over the <threshold> where the
      # highest score is found above 10DM.

    - 'outputOverview<threshold>.txt'
      # Contains an overview of the data of pointings over the
      # <threshold> where the highest score is found above 10DM.

    - 'outputOver<threshold>Under10DM.txt'
      # Contains all data of pointings over the <threshold> where the
      # highest score is found below 10DM.

    - 'outputOverview<threshold>Under10DM.txt'
      # Contains an overview of the data of pointings over the
      # <threshold> where the highest score is found below 10DM.

    - 'output.txt'
```

```
  # Contains all analyzed data, including those under the threshold.
  # This serves mainly as reference and should be ignored.

If you want to plot the pointings with a graphical indication of the
DM where the highest score was found, the program needs the location
of the data, but will not reanalyze the data unless told to do so.
Currently only high scores found above 10DM will be plotted.
Run ‘python analyzeData.py -nlt <threshold> -PA <datadir>‘ to this
end. This will create two .bmp files for each pointing:

- ’DMSigma_tmp<tarfile>.list.bmp’
  # This shows the DM (horizontal) versus Sigma (vertical) plot,
  # just for reference.

- ’TimeDM_tmp<tarfile>.list.bmp’
  # This shows the Time (horizontal) versus DM (vertical) plot.
  # Higher sigma are drawn with larger circles. Two blue bars
  # indicate the DM where the highest score was found.

(2) Commands:
    ‘python analyzeData.py -nkKPf <tarfile>‘

    If you want to review the effect of changes in the scoring
    algorithm, you can run the command above on a single file (although
    ’-A <datadir>’ can still be used as well). This will create output
    in a similar fashion as described above (regardless of score, but
    still with plots only over 10DM), and keep the tarfile unpacked for
    reuse. Note however that the output[...].txt files and plots WILL be
    overwritten with every run.
```