A Correlation Method for Low SNR Pulsar Search and Recognition

Peter East

Abstract

Pulsar detection conventionally relies on synchronously integrating or folding matched pulsar rotation periods of a data record. The final pulsar amplitude is the linear sum of the amplitudes in each data period, whereas the residual noise is the average of all the period noise patterns. Moderate level pulsar recognition techniques then use statistics or exploit known pulse train properties of the pulsar to prove its presence in the final folded result. A new approach to recognizing a true pulsar signal concentrates on eliminating the base noise by exploiting the minor differences in the noise profile of half-data folds. The data is first conditioned to enable the period data to be re-ordered in a number of ways to modify the half-file folded noise patterns whilst not affecting pulsar intercepts or timing. Cross-correlating multiple 50% sections of the re-ordered data sets, further reduces the noise base leaving the constant pulsar pulse unmodified and visible to much lower signal-to-noise levels.

Background

This is the fourth article in a series published in the Journal studying the mechanics of confidently recognizing the presence of a weak pulsar in recordings made with a minimal radio telescope. The first [1], described a back-yard, small portable radio telescope with processing software capable of collecting sufficient data over a 2-3 hour time-frame to detect the B0329+54 pulsar, albeit at low level. The second article [2], discussed the pros-and-cons of recognizing pulsar detections using analytical rather than statistical techniques; this concentrated on algorithms that exploited the pulsar pulse train properties to identify a true pulsar pulse candidate. The third article [3], looked at spectrum analysis techniques and showed that by pre-conditioning the data and using coherent folding, analyzing the harmonics added new slants on the recognition process. Indeed selecting harmonic lines and inverting the spectrum reproduced exactly the conventional folded result. This article tackles the recognition process by investigating the folded result's base noise properties and shows that cross-correlating multiple half-data fold results, enhances and confirms recognition of the true pulsar candidate by suppressing noise. Whilst all the techniques covered in these articles may not be necessary to prove detection in all cases, they do add credibility supported by a firm math grounding to prove that low-level pulsar recognition is now well within the realms of amateur capability, even with limited resources.

Introduction

With a suitable receiving system, a typical folded data result from a pulsar observation recording comprises a base noise pattern with maybe a superimposed integrated pulsar response. Professional radio astronomers use statistical methods to confidently recognize a pulsar at apparent signal-to-noise ratios (SNR) above about 10:1 or 12:1. Keith et al [4], developed a new method of scoring candidates using a series of heuristics which test for pulsar-like properties of the signal to lower this threshold. Similar techniques have been described in this article series. If the pulsar SNR exceeds about 5:1 to 7:1, an amateur observer may be confident that a pulsar is present. Natural pulsar scintillation and folded noise offsets can vary the observed SNR considerably from day to day so that not all recordings may contain a valid pulsar result. Previous articles described methods that exploit pulsar regular pulse train characteristics to improve confidence of a genuine folded/integrated pulsar

pulse. In the region, 3:1 to 5:1, with experience from daily observations, a trained operator can usually spot a weak pulsar response, but recognition confidence may be low. An alternative identity could be unfortunate interference or a period-persistent noise feature.

In this article, after explaining the difficulties of differentiating low SNR pulsar signals from the determinate folded base noise, a new method of removing noise candidates by multiple cross-correlations, which do not affect true pulsar signals, is offered. Various data folding and conditioning tools are described to facilitate this process.

The Problem of Low SNR Pulsar Data Search and Recognition

A typical real folded data result is shown in Figure 1. In this instance, a pulsar pulse is visible, albeit at a moderate SNR level of 5.3:1. In addition, a simulated scintillating pulsar with an SNR of 3.5:1 is introduced to the recorded data at equivalent bin number 600 for test and demonstration purposes. Of interest is the residual noise profile and this needs to be understood so that a low SNR pulsar pulse can be confidently differentiated from a similar noise peak. For low SNR recordings, any or none of the positive peak responses may be weak pulsar pulse. Indeed a true pulsar pulse could sit on a noise peak to enhance visibility or alternatively sit in a negative well and tend to invisibility. Various intermediate results are possible.

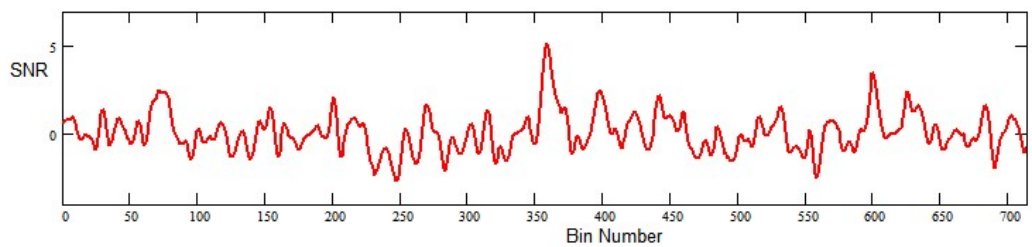

Figure 1. 5.3:1 SNR Folded Result Example

Detailed analysis of the noise background has shown that due to the intense folding of large data sets, noise peaks with apparent SNRs of between 2:1 and 4:1 can exhibit similar characteristics to true pulsars; some however can be ignored if their effective pulse width is too wide or too narrow.

As well as pulse width checking, other simple techniques for noise peak candidate rejection are sometimes used, such as double or treble period correlation or correlating two or more file data sections. These simple correlations can remove some noise peaks, but there is still the possibility that a few noise peaks can still correlate over long data integrations, either naturally or possibly from RF interference. Period search methods that monitor amplitude profile, peak drift, and pulse width variation work well at moderate SNRs, but are not so convincing at lower levels due to greater corruption from the base noise variations around the candidate pulsar.

Half-Data Noise Correlation

By simply splitting the data file into two equal parts, folding each separately and comparing them, it is observed that the noise components may be slightly different.

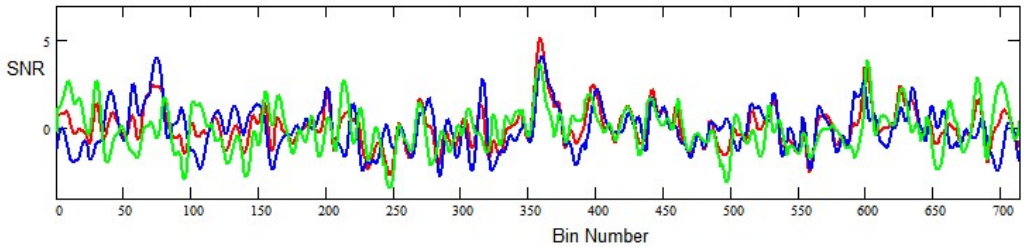

Figure 2. Half-File Fold Comparison. Red - total file; Blue - first half; Green - second half.

Figure 2 compares the base noise between the two half-file data sections. By eye, it is clear that ignoring the pulsar at bin number 358 and the simulated weaker pulsar at bin number 600; there are still significant regions of close shape correlation. The fact that these regions exist along a notional random 2+ hour noise environment demonstrates the power of the folding algorithm to find similar features that repeat on average over the 10,000 or so periods analyzed. It is noted that between the two half-file data plots, the peaks are not always precisely in-phase and this fact is exploited below.

Any pulsar signal present usually remains in place and correlates between the plots albeit with a reduced SNR and possible slight change in peak phase due to the underlying noise. Some noise peaks may still coincide in both plots.

A similar effect is noted when folding down to twin periods and now comparing the consecutive periods (see Figure 3.).

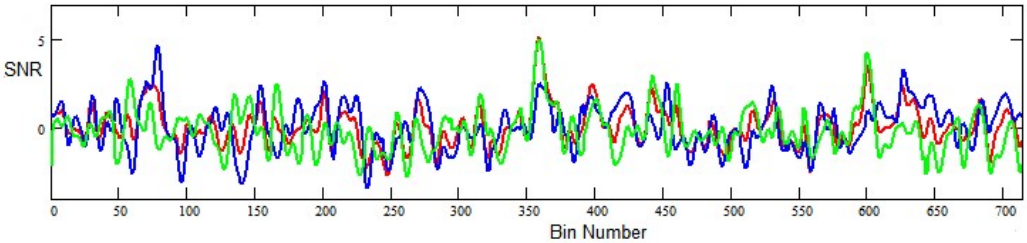

Figure 3. 2-Period Fold Comparison. Red - total file; Blue - first period; Green - second period.

Once again, it apparent that there is a high degree of similarity between the two consecutive fold periods. But also, there are some minor differences; a suggestion is that there may be some benefit in cross-correlating the four plots, especially as the true pulsar pulse and simulated pulsar remain consistent. Of course, summing these four plots reproduces exactly the same pattern as the original data folded to a single period; multiplication produces quite a different output.

Pulsar Recognition - Correlation Algorithm

Figure 4 shows the overall cross-correlation of the positive parts of these four part-folds overlaid on the standard single period fold (red curve). This process is carried out to produce the blue curve, which already has reduced the number of potential candidates due to the half-file noise not being fully identical and is described below. It is seen that some of the noise peaks have been voided but the true and simulated pulsar responses are unaffected in amplitude and shape; the real and simulated pulsar remain fully present in the cross-correlated output.

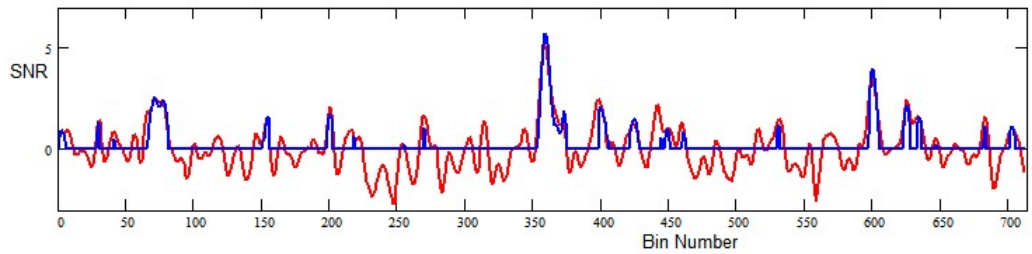

Figure 4. 5:3:1 SNR Pulsar + 3.5:1 SNR Simulation - Twin Pulse/Split-File Correlation

An analysis of the proposed correlation process follows.
Let a folded record be represented by the function, *F(n)*, where *n* is the bin number.
Let the first half of the data record when fully folded be represented by $F_{h1}(n)$ and the second half by $F_{h2}(n)$.
The new correlation algorithm, is given by,

$$F_1(n) = \sqrt{p(F_{h1}(n)).p(F_{h2}(n))} \qquad (1)$$

where, $p(x)$ represents selecting the positive part of x. This function is included for two reasons. Firstly, there is likely to be little pulsar information in the negative part of the folded file and secondly it ensures a real result when taking the square root of the correlated positive product.

Now suppose that the detected data record is folded using a double pulsar period and the two periods extracted separately, represented as $F_{p1}(n)$ and $F_{p2}(n)$. The correlation algorithm is now expressed as,

$$F_2(n) = \sqrt{p(F_{p1}(n)).p(F_{p2}(n))} \qquad (2)$$

These two measures can be cross-correlated and combined to produce the result,

$$F_O(n) = \sqrt{F_1(n).F_2(n)} \qquad (3)$$

This process on the data record example produces the blue plot in Figure 4. It is worth noting that if, $F_2(n) = F_1(n)$, the output $F_0(n) = F_1(n)$ so if a pulsar pulse is present it will mirror the single-period fold (red plot) representation as in Figure 4. If the noise is different, it either will change amplitude or be set to zero if either contributor is zero; hence the zero line in Figure 4.
There is no loss of signal-to-noise ratio within the pulse for moderate SNRs due to the multiplicative correlation approach; indeed, the pulsar SNR is the same as if the two inputs were summed. There is of course a considerable gain in the apparent SNR over the period due to the noise rejection process.
For example, within the pulse; if the SNR is *s/n* for a full fold, for each half data fold the SNR is expected to be, $s/\sqrt{2}n$. If the signal and noise in the two halves are,

$$s_1/2, n_1/\sqrt{2} \;\; and, \;\; s_2/2, n_2/\sqrt{2}, \;\; where, \left(s_1/2 + s_2/2\right) = s, \;\; and, \;\; n_1/\sqrt{2} + n_2/\sqrt{2} -> n$$

For the correlation multiplication within the pulse, we get,

$$\sqrt{\left(\frac{s_1}{2}+\frac{n_1}{\sqrt{2}}\right)\cdot\left(\frac{s_2}{2}+\frac{n_2}{\sqrt{2}}\right)} \approx \sqrt{\left(\frac{s}{2}\right)^2+\frac{\sqrt{2}ns}{2\sqrt{2}}}$$

$$=\frac{s}{2}\left(1+\frac{2n}{s}\right)^{\frac{1}{2}} \approx \frac{s}{2}\left(1+\frac{n}{s}\right)=\frac{s}{2}+\frac{n}{2}\rightarrow\frac{s}{n}$$

That is, the same SNR as for a full fold.
This is an interesting result, as it appears to mean that no loss of SNR would occur with treble, quadruple etc: data cuts or multiple period folds. A limitation does occur, however, due to the approximation made in the above derivation, which ignores the noise x noise contribution, which comes in at lower signal-to-noise ratios than presently considered. For this reason, multiplicative folding the whole data file is not recommended.

Improving Pulsar Discrimination by De-Correlating Noise

Figure 4 shows that by correlating half-file and twin-period folds on a single data record, that some base noise data can be rejected as a pulsar candidate.
The concept offered here is that if the data in each half-file can be re-ordered without corrupting the pulsar timing, then the half-file noise components will change producing a different cross-correlation noise profile. Correlating the outputs of a number of re-ordered half-data files to reduce/remove, other candidate noise peaks should improve the visibility of a true pulsar.
First, the detected file needs converting to a form that preserves the pulsar timing data. This is possible with an intermediate folding stage termed partial folding.
Two methods are considered termed Series folding and Parallel folding (described in Appendix 1). In series folding, the data is effectively compressed by folding blocks of periods down to a single period of a fixed number of bins. These block/bin-sets are then combined in series to produce a part-folded data file with pulsar periods now normalized to the bin number choice.
As an example, a 2-hour recording of pulsar B0329+54 (period approximately 714.4ms) will contain about 10,000 rotation periods. This data is split into 500 x 20-pulsar periods. Each 20-period section are down-folded to a single period of 715 bins and attached serially to earlier 20-period folded sections. The result is an output file of 500 x 715bin folded periods. The file is a compressed version of the original file with the amplitude scintillation profile also compressed but maintained along the file.
For parallel folding, this is equivalent to multi-period folding and is compatible with series-folding data if the number of periods down-folded is equal to the number of folds in the series. Continuing the example above; now the data is divided into 20 blocks, each containing 500 pulsar periods (M = 500) folded into say 500 x715bins. Summing the 20 blocks in parallel produces a file of 500x715 samples (approximately 1ms bins). Further down-folding to a single pulsar period now uses the period/bin value P = 715, so normalizing the data for ease of combining with other data records. This parallel-combining process re-orders the data so that now any pulsar scintillation is block-averaged tending to smooth out real variations. The re-ordering also of course changes significantly the half-fold/twin-period noise patterns as desired.
Figure 5 is the result of cross-correlating half-file/twin-period results from the series and parallel partial folds described above. Comparing Figures 4 and 5 shows that data reordering is a valid approach to removing noise peak candidates. Note that again the real and simulated pulsar pulses appear unaffected whereas just two major noise

candidates remain but these are slightly attenuated (see Appendix 3 for a supporting analysis).

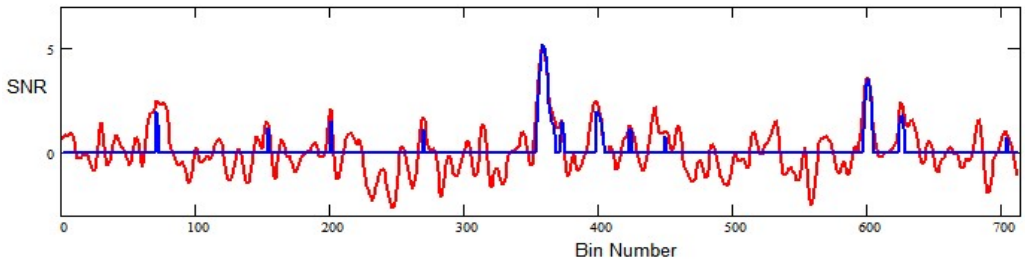

Figure 5. 5:3:1 SNR Pulsar + 3.5:1 SNR Simulation - Series/Parallel File Correlation

Finally, once series and parallel partial folded files are available, the bin data sets can be variously randomized on a bin-period basis without affecting the final single fold result. These modified data files further de-correlate the half-fold residual noise responsible for causing false peaks.

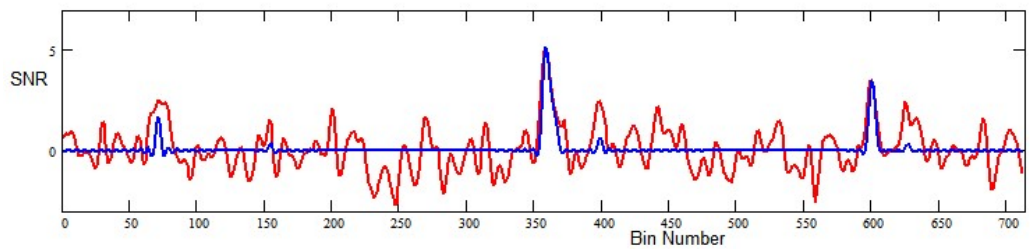

Figure 6. 5:3:1 SNR Pulsar + 3.5:1 SNR, Simulated Pulsar - Recognition Process

Figure 6 shows the results of the new technique, cross-correlating straight series/parallel files with two sets of randomized series/parallel sets. The blue responses indicating the regular components to be considered as possible pulsar intercepts and is generated using the new algorithm. The red curve is the standard folding algorithm response. The pulse at bin 358 is identified as the B0329+54 pulsar, from a data recording showing an SNR = 5.3:1; noting the correct pulse width is observed rounded to 7ms. The response at bin 600 is the simulated pulsar of 6.5ms width added to the real data recorded file at a true SNR of 3:5; the indicated pulse width is 6ms. There are four other minor spurious noise/RFI responses remaining with SNRs less than 2:1. All other fold peaks are rejected by the new procedure and it is noted that the real and simulated pulse responses are preserved and not attenuated.

Randomizing Partial Folded Data

With both series and parallel partial data folding, the resultant data is in the form of *M* periods of *B* bins (500 x 715 for the examples above), and it is possible to change the order of, or randomize the order of, the 500 bin sets. A MathCad program to randomize the bin-set order using the Fisher-Yates Shuffle algorithm is given in Appendix 2. If this is carried out the position of the notional pulsar pulse in each bin set is unchanged, but for each half or section of the file, the folded noise profile is subtly modified. Zeroing the negative half of the data in the final fold before correlation is valid if it contains no pulsar information but noise zero-crossings vary to remove residual positive noise. It is possible that this process may affect the pulsar signal if noise dominates around the pulse. The fact that in the examples of Figure 4

and 5, the pulsar pulse amplitude doesn't change means that the base noise around the pulsar and simulated pulsar is not significant.
So far, we have only discussed a 2-factor split, but greater data division factors are feasible and may be expected to remove residual noise candidates. 3, 4, and 5-file split and period multiples have been examined with some success (see Appendix 4). Residual noise peaks may also be removed by using multiple randomized data files together with multiple application of the correlation equation.

Search and Identify Procedure
On the assumptions that an observation data record with known pulsar topocentric period has been detected, down-sampled and is in the form of a text file and the available math package is a recent version of MathCad, the programs in Appendix 2 can be used. The procedure to identify a pulsar response if present is to follow the plan outlined in Figure 7 below, after first completing a standard fold and plotting the result and retaining for data comparison.

1. Produce 2 partial-fold condensed files - a series compressed version and a parallel multi-period fold version as advised in Appendix 1.
2. Carry out half-data and 2-period folds on the 4 partial-data sets so far generated
3. Separate the 2-period folds for each partial data fold, then, cross-correlate each pair in line with Equation 3; similarly for the two half-file folds, as in Figure 7.
4. Plot the result to check pulsar visibility and identify significant noise candidates.
5. Choose random seeds and random shuffle copies of the Series and Parallel partial fold files following the program in Appendix 2.
6. Cross-correlate the results for the two new files and combine via Equation 3.
7. Repeat and optimize plot by changing the random file seeds.

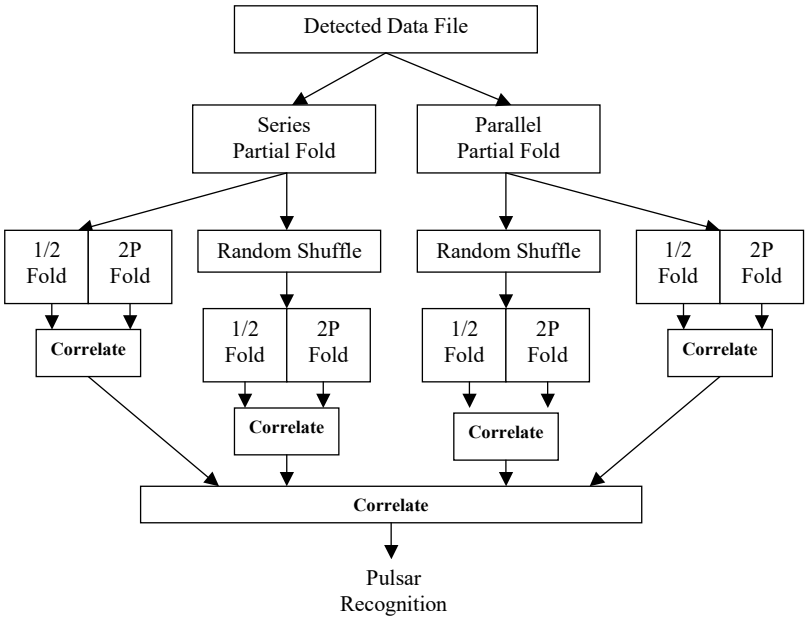

Figure 7. The Pulsar Recognition Process

This procedure may not, on the first try, suppress all the noise components and leave an unaltered pulsar pulse when compared with the standard single fold. In this case, it

may be necessary to modify the random seeds selected for the randomized files or, finally, to double-up on the number of randomized files. Modification of the random seeds is recommended anyway to assure the operator that an unchanging pulsar is present. By varying seed choices; more results improve confidence by highlighting pulsars and rejecting false noise candidates. For pulsar SNRs greater than 4:1, pulsar recognition can be rapid but down to 3:1, it may be possible for odd random seeds to blank a true pulsar due to comparability with the local base noise. In this case, frequent seed changing may soon demonstrate that a particular peak of interest remains present and mainly un-attenuated.

Apparent SNR

A continuous signal pulse train embedded in a noise of RMS amplitude *N*, having a true SNR of *S/N* is measured by estimating the mean pulse amplitude and dividing this value by the rms of the base noise outside the pulse.

The rms level in both cases may be very similar, but the observed pulse amplitude, in the fixed period, will be modified by the instantaneous noise level within the pulse. Assuming this uncertainty to be of the same magnitude as the rms noise, then the apparent SNR is likely to be within the range $S/N \pm 1$. The uncertainty can be ignored for SNRs > 10, but for a true SNR of 3:1 for example, this predicts an observed SNR is likely to be in the region of 2:1 to 4:1. The probability of apparent SNR's in this range for Gaussian/Normal noise is 68% so there is still a finite probability of this range being exceeded.

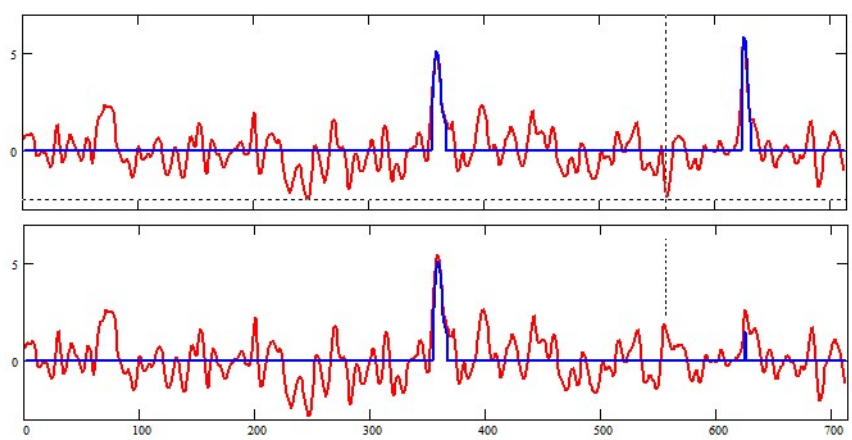

Figure 8. Real and Apparent SNR. Top plot: test to bin 625, bottom plot: test to bin 556

Figure 8, upper plot, shows the result of moving the test pulsar (true SNR = 3.5:1) from bin 600 to bin 625. The noise peak here was equivalent SNR = 2.4:1. The pulsar is still detected by the new algorithm and the apparent SNR increased to 5.8:1. In the lower plot, the test pulsar is moved to bin 556 where the equivalent noise negative peak was -1.6:1. In this case, the bin 556 SNR increased to +1.8:1 but failed algorithm detection using Figure 10 random seeds. Assuming linear addition, one would expect the positive SNR to be +1.9:1. The conclusion from this test is that it is difficult to exactly put a lower limit on minimum detectable SNR, since there is a 50% probability that the true SNR is increased or decreased, depending on the polarity of the local integrated base noise.

In fact, in this particular case, a new random seed search did find the a valid response at bin 556 meeting the amplitude and pulse width detection criteria, (Figure 9), but

without knowing its presence, it is unlikely that it would have been found with any confidence. It does however demonstrate the technique value in a confirmatory role.

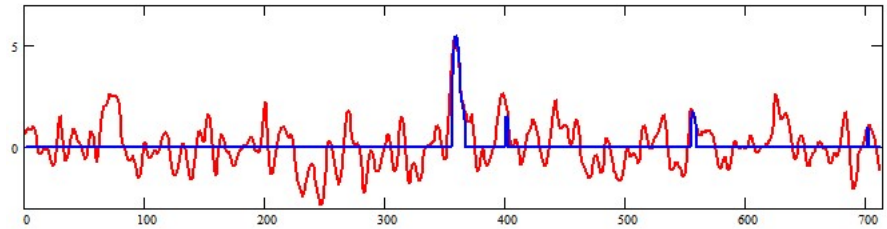

Figure 9. Random Seed Search for Test Pulsar at bin 558 (apparent SNR = 1.9, true = 3.5)

General Comments

To summarize what has been learnt so far;

1. The fully folded noise is the average noise profile derived from every period folded, as is the pulsar profile. The half-file folded results show this to be true.
2. The half-file slight noise profile differences show that there are small profile variations along the data file.
3. The correlation process between the two half-file results shows up the parts that match in both positive amplitude and time; highlighting both the pulsar pulse and significant noise peaks.
4. Changing the positions of the pulsar periods within the file does not change the full-fold results but does change the half-file fold detail.
5. In principle, by cross correlating many re-ordered period half-files all noise peaks will eventually be removed leaving only a fully correlated pulsar pulse visible.

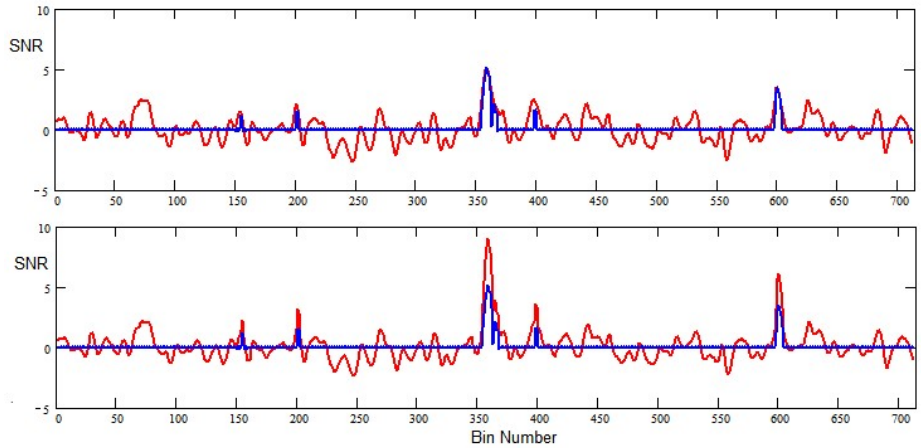

Figure 10. Recognition Process with new random seed (blue) overlaid on Standard fold (red) for upper plot and both summed (red) for lower plot.

The technique is not critical on the random seeds chosen and there appears a lot of flexibility in their choice. Another feature of the cross-correlation process is that the amplitude and shape of a true pulsar pulse is unaltered; so that it is possible to directly add the final cross-correlation result to the original folded data to effectively double the observed SNR as shown in the lower plot of Figure 10.

The upper plot shows the cross-correlation result using different random seeds to the earlier result plotted in Figure 6. This shows that some experimentation is required to

optimize the result. Otherwise, the solution is to generate more randomized files to further de-correlate the base noise.
A faster multi-split pulsar/noise discriminator solution based on correlation but not involving randomization of file data is described in Appendix 4.

Conclusions
This article describes a novel process for identifying good candidates in searching for a pulsar response in potentially low SNR recordings. The pre-requisite is that the pulsar fold period is known exactly. Experiments with a number of real data files and using simulations added to relevant recorded data indicate that even candidates below 3:1 can be detected although some may be partially nulled in the process, depending on interactions with the local noise. Not all real data from minimal systems can be expected to reveal a candidate as scintillation and day-to-day interference can vary both the integrated pulsar amplitude and the noise floor. In addition, the real data SNR can be degraded by the noise base being negative around the pulsar true position as noted earlier. Nevertheless, the technique has proved a significant addition to the range of validation tests recommended in the earlier articles. The implementation mechanics may seem complex but it works, and once the processing sequence is set up, testing new records can be very rapid and rewarding.

Postscript
To complete the circle, the new recognition algorithm has been applied to data used in the first article in this series, Reference [1]. In this document, an 8-point manual validation routine was proposed for identifying the presence of a weak pulsar. The topmost plot in Figure 11 is taken from the article showing the data fold plot for Dataset 24. Here, the central spike at bin number 357 with an indicated SNR of 3.1:1 was identified as pulsar B0329+54.

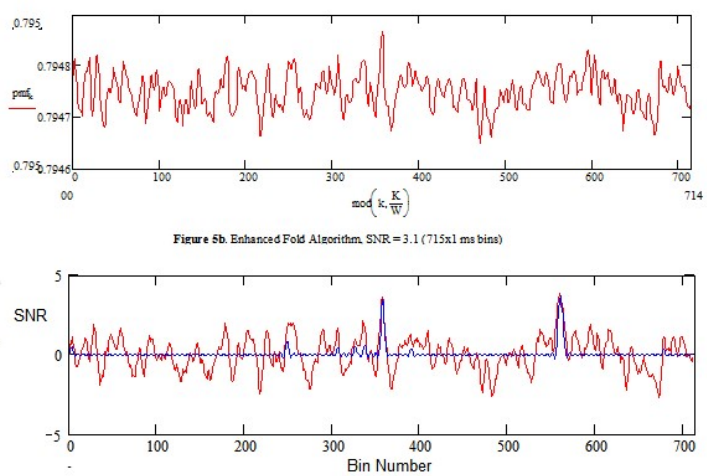

Figure 11. Human/Algorithm Pulsar Recognition Comparison

The lower plot shows the new noise-reduction algorithm result on the same raw data but with a test pulse SNR = 3.8 inserted at bin 558. With some RFI blanking, the indicated pulsar SNR at bin number 357 improved to 3.6:1.
The blue plot confirms full recognition of both real and simulated targets. Q.E.D.

References

[1] P. W. East, A Minimal Pulsar Detection System. Journal of the Society of Amateur Radio Astronomers. January-February 2018. p36.

[2] P. W. East. An Analytical Method of Recognizing Pulsars at Moderate SNR. Journal of the Society of Amateur Radio Astronomers. November-December 2018. p17.

[3] P.W. East. Spectrum Techniques for Improving Low SNR Pulsar Detection. Journal of the Society of Amateur Radio Astronomers. March-April 2019. p21.

[4] M. J. Keith et al, 'Discovery of 28 pulsars using new techniques for sorting pulsar candidates' Mon.Not. R. Astron.Soc. 395, pp837-846(2009)

PWE May 2019

Appendix 1 Partial Folding Algorithms

All the folding schemes outlined in this section, when eventually combined or down-folded to a single period will produce exactly the same result; no information is lost. The partial folding schemes having some interesting and useful properties are described later in this section.

1. Standard Fold

The standard fold method depicted in Figure A1, splits the detected data record, (RFI blanked and pre-filtered to match the pulsar pulse frequency response; see Appendix 5), into a number *R* exact, period *P* sections, divides each period data into a fixed number of bins, *B*, then sums the bin data in parallel. The resulting file is of period *P* duration as illustrated by the example in Figure 3; the pulsar pulse appearing at time *t* after the notional period start, adds linearly.

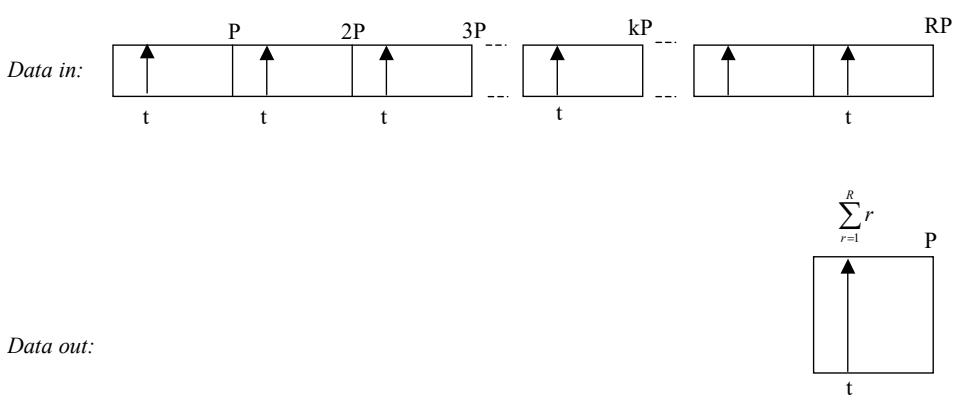

Figure A1 Standard Folding Algorithm Data Structure

2. Parallel Partial Multi-Period Fold

In this scheme, the data is split into exact multi-period blocks *MP* long as shown in Figure A2, with each period divided into *B* bins. The multi-period bin data is summed coherently in parallel as for the standard fold. The resulting data file is then of *MP* duration. The standard fold program in Figure A4 is relevant with the value *M* set for the number of periods selected for the parallel partial file.

Useful properties of this scheme are,

1. The data is compressed for ease of testing.
2. Scintillation modulation is spread out and averaged over the observation window.

3. Exact integer number of periods is included in the compressed file making data summing and spectrum investigation more predictable.

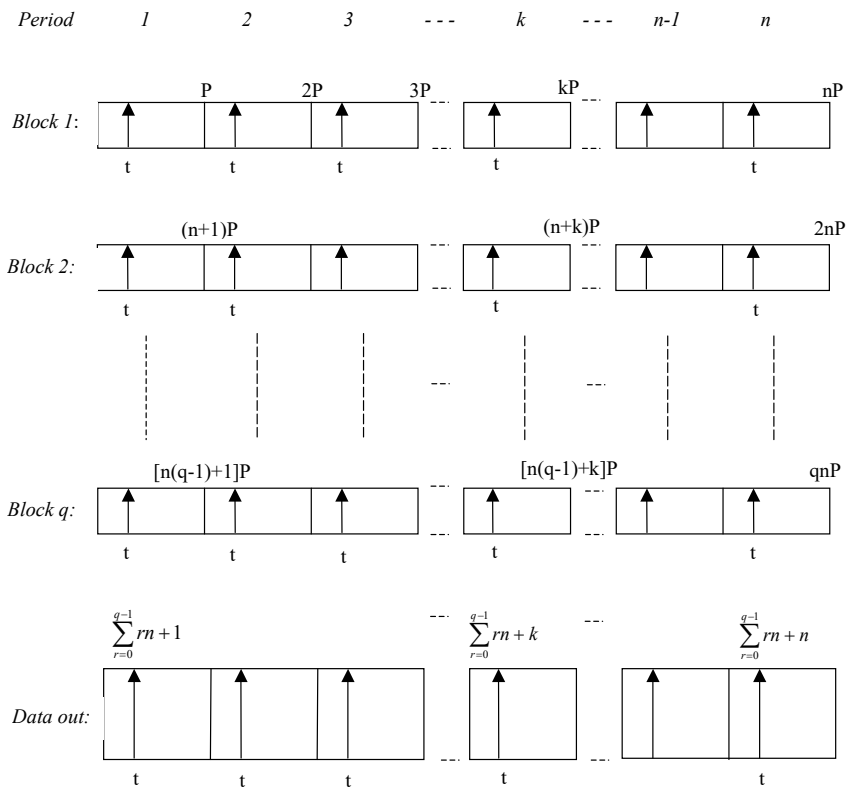

Figure A2. Multi-period Parallel Folding

A disadvantage however is that the folding data period must match the true pulsar value and because the file real-time data is period-overlapped, search routines on the data are not effective.

By maintaining the data blocks in exactly an integer number of periods, the block data can be summed coherently without any loss of pulsar data information. The resulting output file is *M*-periods long and the period is now divided into *B* bin intervals rather than the original data sample intervals. Further down-folding therefore uses this information.

3. Serial Partial Multi-Period Fold

An alternative method of data reduction is compression folding, where blocks of data are folded and attached serially. The data is now subdivided into *M* sections of '*integer part of* (*Max/M*)' periods, each section is folded as for the standard fold method, then the folded results are combined in series to produce a data file of *M* periods duration. Figure A3 illustrates the process.

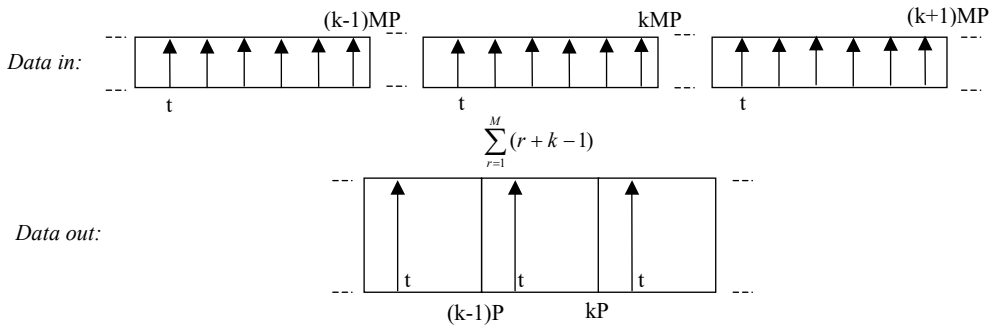

Figure A3. Multi-period Serial Folding

4. Other Folding Possibilities.

a). Harmonic Folding

Once the data has been coherently compressed using either the parallel or the serial partial folding technique into a binary power of bins, the digital Fourier Transform can be used to convert the result into its spectrum. By suitable calculation, the harmonic spacing can be deduced and the spectrum folded into a single harmonic period. If this final harmonic folded result is then converted back to time series by inverse Fourier transformation then an identical result to an all-data fold is produced. However, examination of the spectrum yields information that helps to confirm a pulse train presence and is discussed further in Reference 3.

b). Polarity Fold

In this instance positive and negative parts of the original data or the partial-folded data, first corrected to ensure a mean of zero, are folded separately. As with section folding and multi-period folding, the folded results are slightly different and can be correlated as before to remove some noise candidates. This might be expected since both pulsar signal and folded noise spectra are independent of polarity at this stage. Only after folding can negative responses be ignored.

Hard limiting data prior to folding still produces the same final fold response.

Appendix 2 Data Folding Programs

Outline MathCad programs for summing data blocks are listed below. The sequence and routines programs are fairly clear and could be translated for other coding languages as required.

In the following programs,

B is the number of fold bins
M is the number of periods in a partial fold
P is the pulsar period
Dat is the detected data file
Min/Max are the file sample start offset and end values

1. Standard/Partial Parallel Folding

Figure A4 lists the standard fold program which can also configure a parallel partial fold by setting the command variable M (M = 1, for a standard single period fold).The central body ensures synchronous pulsar integration and the final part normalizes the bin count. The second routine is the fold algorithm detail that ensures that the pulsar pulse is synchronized to the same bin/bins. The remaining lines sum periods and normalize the bin pulse counts.

```
Stfold(B,M,P,Dat,Min,Max) := | for fs ∈ 0..M·B − 1
                             |   | bdat_fs ← 0
                             |   | bcount_fs ← 0
                             | for x ∈ Min..Max − 1
                             |   | s ← floor[[(x/(M·P) − floor(x/(M·P)))·M·B] + .5]
                             |   | bdat_s ← bdat_s + Dat_x
                             |   | bcount_s ← bcount_s + 1
                             | for b ∈ 0..M·B − 1
                             |   bindat_b ← bdat_b / bcount_b
                             | bindat
```

Figure A4. Standard Folding Algorithm Code

The standard fold command applied to the example in the main text is to fold all the data file into a single period of 715 bins is,

$$Stfold(715,1,P,Dat,0,Max)$$

To produce a partial parallel fold of 500 periods, the command is,

$$Datp = Stfold(715,500,P,Dat,0,Max)$$

To further fold this data down to two half-file sections, the commands are,

$$F_{h1}(n) = Stfold(715,1,715,Datp,0,N/2)$$
$$F_{h2}(n) = Stfold(715,1,715,Datp,N/2,N)$$

where, $N = 500*715$

2. Serial Partial Fold

The fold command applied to the example in the main text is to fold the entire data file into a serial data fold of 500 periods of 715 bins is,

$$Dats = Sefold(715,715,P,Dat,0,500*715)$$

The program listing follows.

```
Sefold(B,M,P,Dat,Max) := | nmax ← floor(Max/M)
                         | for a ∈ 0..B·M − 1
                         |   outm_a ← 0
                         | for x ∈ 0..M − 1
                         |   | xs ← floor(floor(x·nmax/P)·P + .5)
                         |   | for y ∈ xs..xs + nmax − 1
                         |   |   datn_(y−xs) ← Dat_y
                         |   | out ← Stfold(B,1,P,datn,0,nmax)
                         |   | for y ∈ 0..B − 1
                         |   |   outm_(y+B·x) ← out_y
                         | outm
```

Figure A5. Multi-period Serial Folding Algorithm

3. Random Shuffle Fold

Partially folded files now have a fixed structure of, in this example, of 500 periods of 715 bins. The randomizing program below can re-order the positions of the 500 periods. Depending on the initial seed value, so that half-file and 2-period folds now have a modified noise base. The Fisher-Yates shuffle routine is coded in the third 'for' loop of Figure A6.

An example command is, *SefoldRnd*(715,500,715,*Datr*,*N*,*r*), where *r* is a random seed, and *Datr* is the partial fold file to be randomized and *N* the file length. A suitable random number generator is shown in the bottom left of Figure A6.

The program listing follows.

```
SefoldRnd(B,M,P,Dat,Max,r) := | r ← r/π − floor(r/π)
                              | nmax ← floor(Max/M)
                              | for a ∈ 0..M·B − 1
                              |   outm_a ← 0
                              | for x ∈ 0..M − 1
                              |   array_x ← x
                              | for x ∈ M − 1, M − 2..0
                              |   | r ← ran(r)
                              |   | ind ← floor(r·x)
                              |   | tmp ← array_ind
                              |   | array_ind ← array_x
                              |   | array_x ← tmp
                              | for x ∈ 0..M − 1
                              |   | xs ← floor(floor(x·nmax/P)·P)
                              |   | for y ∈ xs..xs + nmax − 1
                              |   |   datp_(y−xs) ← Dat_y
                              |   | out ← fold(B,1,P,datp,nmax)
                              |   | for y ∈ 0..B − 1
                              |   |   outm_(y+(B·array_x)) ← out_y
                              | outm

ran(r) := | n ← (π + r)^5
          | o ← n − floor(n)
          | o
```

Figure A6. Random Shuffle Algorithm + Repeatable 'Random' Number Generator

Appendix 3 Effect of Half-Fold Amplitude Differences on Correlation Plots

It has been noted in the experiments that correlation tuning has slightly different effects on true pulsar pulses than for peak noise candidates. A true pulsar response seems largely unaffected by seed changes with randomized data whereas suspected noise candidates appear to reduce in amplitude slightly for some seed values.

A simple analysis follows,

Let, S = the full-fold pulsar amplitude.

N = the local noise offset.

R = the RMS noise level outside the pulsar pulse.

Half-fold amplitudes are $\frac{S + dS}{2}$ and $\frac{S - dS}{2}$, where dS allows for an unequal signal split in the two half-folds. Similarly for the local noise offset amplitudes, $\frac{N + dN}{2}$ and $\frac{N - dN}{2}$; noise sums linearly here as they are now finite offsets.

The correlation output for a pulsar signal S and a local offset, N, is,

$$\sqrt{\left(\frac{S+dS}{2}+\frac{N+dN}{2}\right)\left(\frac{S-dS}{2}+\frac{N-dN}{2}\right)}$$
$$=\sqrt{\frac{S^2-dS^2}{4}+\frac{SN}{2}+\frac{N^2-dN^2}{4}}$$
$$=\sqrt{\frac{S^2}{4}\left(1+\frac{2N}{S}+\frac{N^2-dN^2}{S^2}\right)}\approx\frac{S+N}{2}$$

Showing that for moderate SNRs, the split differences can be ignored. The pulsar apparent peak output is therefore relatively unaffected by unequal split of power between the half-folds, but is directly sensitive to any local noise offset.

The apparent SNR is given by, $SNR=\frac{S+N}{R}$

The correlation output for a noise peak candidate, amplitude N, with unequal path noise splits dN is,

$$\sqrt{\left(\frac{N+dN}{2}\right)\left(\frac{N-dN}{2}\right)}$$
$$=\sqrt{\frac{N^2-dN^2}{4}}$$
$$\approx\sqrt{\frac{N^2}{4}\left(1+\frac{dN^2}{N^2}\right)}\approx\frac{N}{2}-\frac{dN^2}{4N}$$

The equivalent noise peak SNR is, $\frac{N-\left(dN^2/2N\right)}{R}$

Using an example with $S = 3$, $N = 2$, $dN = 1$ and $R = 1$.
The apparent pulsar SNR can range from 1:1 to 5:1 ($\pm N$) and the apparent noise peak SNR can in fact reduce, in this example, from 2:1 to 1.75:1.

Appendix 4 Quick Multi-period Correlation

The preferred approach as discussed in the main body has been on randomizing data from series and parallel partial folded files and applying the correlation equation (Equation 3) to factor 2 splits of data (half file and twin pulse folds). For a faster result it is possible to split the file into say 3 sections and coupled with folding using 3-times the pulsar period. Separating the 6 files produced far, the modified Equations, 1, 2 and 3 become,

$$F_1(n)=\left[p(F_{h1}(n)).p(F_{h2}(n)).p(F_{h3}(n))\right]^{\frac{1}{3}}$$
$$F_2(n)=\left[p(F_{p1}(n)).p(F_{p2}(n)).p(F_{p3}(n))\right]^{\frac{1}{3}}$$
$$F_O(n)=\sqrt{F_1(n).F_2(n)}$$

Where F_1 and F_2 combines the 3 file sections folds and 3 separated 3-period folds respectively.
Also, the process can be carried out on both series and parallel, partial source files
This principle can be extended to factors 4 and 5 for possibly better discrimination. With higher factors, the reduction of signal-to-noise ratio in the divided parts reduces the effectiveness of this approach as base noise can tend to reduce visibility of the pulsar signal. It is, however, useful for a quick data assessment.

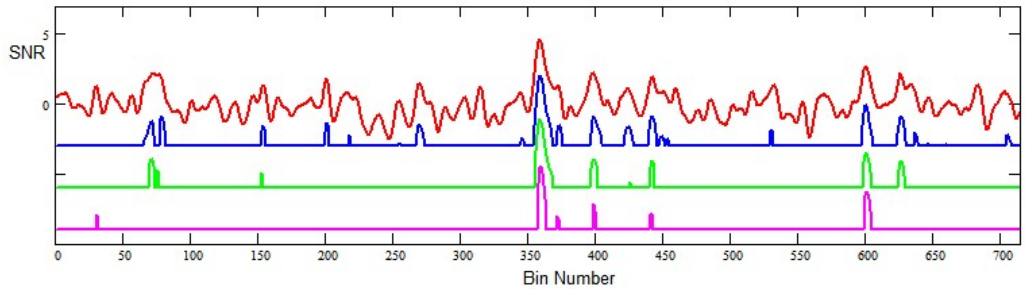

Figure A6. Comparison of Factor 2 (blue), Factor 3 (green) Factor 4 (magenta)Data Splits

Figure A6 compares the correlation algorithm on factor 2, factor 3 and factor 4 data splits (plots are offset by 3,6 and 9 units) for clarity. Note the real and simulated pulsars are still fully present. Also of note is the reduction in number of noise peaks with increasing factor but more seriously, the narrowing of the simulated pulsar pulse width due to the greater local noise significance.

Appendix 5 Data Pre-processing

Receiver-sampled data is assumed in IQ format at the receiver sampling rate that has been de-dispersed, detected ($\sqrt{I^2 + Q^2}$) and downsampled (averaged) to rate suitable for preserving the required pulse detail (~1ms for B0329+54 for example). The data file may still contain RF interference, DC offsets and noise modulations up to the sampling rate. Further data pre-processing may therefore be necessary to enhance the probability of detection.

As before,

Dat is the detected data file
Max is the number of detected file samples
R is a reasonable data block size; preferably a power of 2.
W is the pulse width in units of the downsample sample time interval.

The listing in Figure A7 removes DC and large RFI spectral lines. To set *d*, the spectrum amplitude threshold, it is sensible to test a data block spectrum first.

```
RFI(d, Dat, Max, R) := | for u ∈ 0..Max − 1
                       |   Dout_u ← 0
                       | for x ∈ 0..(Max/R) − 1
                       |   | for u ∈ 0..R − 1
                       |   |   | fout ← 0
                       |   |   | sig_u ← Dat_(u+x·R)
                       |   | fsig ← cfft(sig)
                       |   | for u ∈ 0..R − 1
                       |   |   fout_u ← if(|fsig_u| > d, 0, fsig_u)
                       |   | dout ← icfft(fout)
                       |   | for u ∈ 0..R − 1
                       |   |   Dout_(u+x·R) ← Re(dout_u)
                       | Dout
```

Figure A7. RFI Frequency Blanking

Figure A8 below, is an obvious solution to removing amplitude spikes in the data. As before a typical data block is tested to estimate a sensible threshold level *d*,

$$Dat_x = if(|Dat_x| > d, 0, Dat_x)$$

Figure A8. Amplitude Spike Blanking

Figure A9 lists the code to apply matched filtering to the data to optimize detection of a pulsar in the final data fold. The process in the bottom left-hand corner describes a Gaussian-shaped pulse of the same pulse width as the expected pulsar half-height width. The displayed signal-to-noise ratio is then no longer dependent upon the number of fold bins.

```
Fil(Dat, Max, R, W) := | for u ∈ 0..Max − 1
                       |   Dout_u ← 0
                       | for u ∈ 0..R − 1
                       |   pw_u ← PW(u, floor(R/2), W)
                       | fpw ← cfft(pw)
                       | for x ∈ 0..floor(Max/R) − 1
                       |   | for u ∈ 0..R − 1
                       |   |   | fout_u ← 0
                       |   |   | dat_u ← Dat_(u+x·rr)
                       |   | fdat ← cfft(dat)
                       |   | for u ∈ 0..R − 1
                       |   |   fout_u ← fdat_u · |fpw_u / fpw_0|
                       |   | dout ← icfft(fout)
                       |   | for u ∈ 0..R − 1
                       |   |   Dout_(u+x·rr) ← Re(dout_u)
                       | Dout

PW(x, X, W) := exp[−4·ln(2)·((x − X)/W)^2]
```

Figure A9. Matched Pulse Filtering