Pulsar data processing

Pulsar software, methods and algorithms

George Hobbs

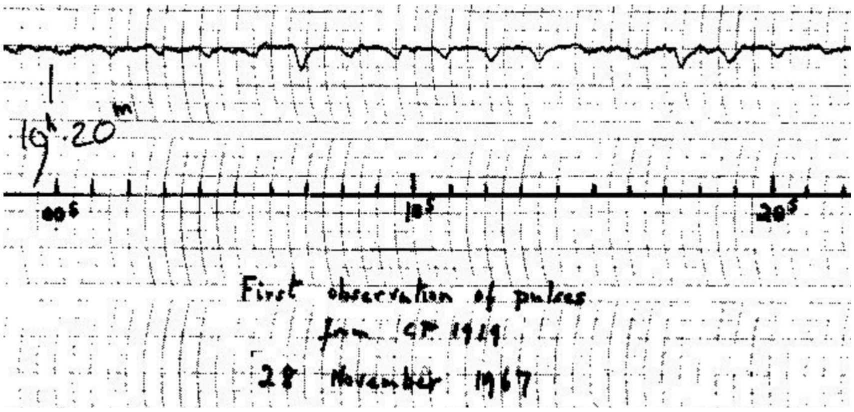
Introduction

Pulsars are highly magnetized, rapidly rotating neutron stars. They are also astrophysical laboratories for the study of extreme physics. There's no doubt that pulsars and neutron stars are intrinsically interesting. Representing an end-state of stellar evolution, they are incredibly dense and have huge magnetic fields. Neutron stars can be observed in order to study electrodynamics and plasma processes within highly magnetized media or to study quantum and particle physics in the superconducting superfluid of their interiors. Pulsars are (usually) detected by the regular sequence of pulses they emit. Pulsars creating very stable pulsed signals are observed in the hope of making a direct detection of gravitational waves and also to search for irregularities in terrestrial time standards. Pulsars in binary systems are a unique means of studying theories of gravitation in the strong-field limit. They have also been used to probe the electron density distribution in the interstellar medium and globular clusters, to study the size and direction of the Galactic magnetic field and to determine the gravitational potential of the Galaxy and globular clusters. Pulsar observations have also led to measurements of the mass of solar system objects and to search for unknown objects in our solar system. The first extrasolar planets to be discovered were orbiting a pulsar.

A new generation of telescopes and instrumentation will soon revolutionize pulsar astronomy. With luck, within a few years, pulsar astronomers will be studying gravitational waves in detail, probing gravity by studying pulsars orbiting black holes and analyzing the population of pulsars in distant galaxies. In the not too distant future it is also possible that spacecraft will be navigating their way through our solar system by means of pulsar observations.

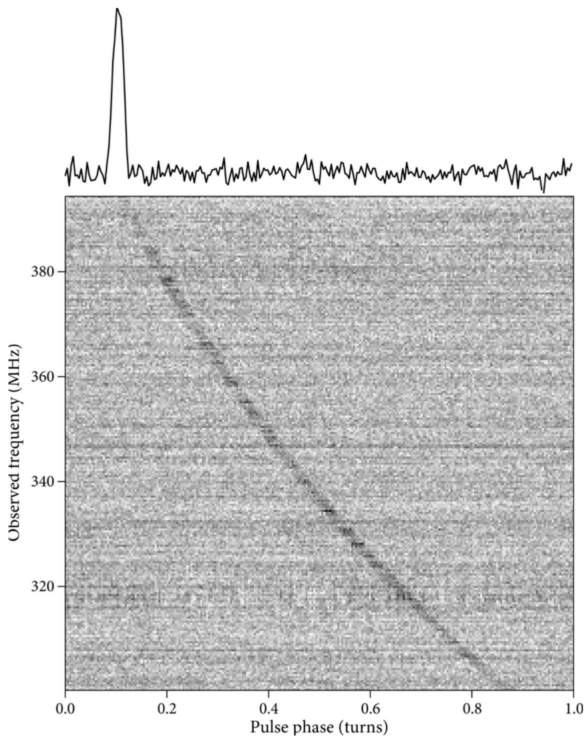
The first **pulsar** was discovered during the year 1967 by Jocelyn Bell Burnell and published by Hewish et al. (1967). There are two intriguing twists to this first discovery. In the late 1950s, a visitor (with no professional astronomy training) viewed the Crab Nebula using an optical telescope at the University of Chicago. She noted to a local astronomer that she could see a source flashing, but her observation was disregarded simply as ``scintillation''. Perhaps she actually was observing optical pulses from the Crab pulsar. The second twist was brought up during a conference in Montreal in 2007. During that meeting Charles Schisler described how he had been a US Air Force sergeant at a remote Alaskan outpost[1]. At the conference he produced his notes from 1967 which showed that his military radar equipment had detected the pulsar in the Crab Nebula. However, it was not until much later that he was to learn what it was that he had seen.

The discovery observation (reproduced below) shows the basic properties of pulsar observations:



The top track in this figure shows the output of the telescope used to make the first pulsar observation. The pulsar signal is seen as the sequence of pulses (with their recording system the pulses go downwards on the plot, but they refer to an increase in the detected radio signal). The time between adjacent pulses in known as a **pulse period**. Currently known pulsars have periods between ~1ms and ~10 seconds. The first-discovered pulsar (PSR B1919+21) has a period of 1.3 seconds.

Pulsars are named after their sky positions. The first character in a pulsar name (a B or J) represents the Besselian or Julian dates (B1950 or J2000). The remaining characters give the right ascension and declination. All pulsars have a J-name. Pulsars discovered before the year ~1995 also have a B-name. The first-discovered pulsar is therefore known as both PSR B1919+21 and PSR J1921+2153.



If a pulsar signal is observed over a wide-observing-bandwidth then it is clear that the signal is dispersed in the interstellar medium: the pulse arrives earlier at a higher observing frequency than at a lower observing frequency. The time delay between pulses arriving at two frequencies can be determined from:

$$\Delta T \approx \frac{e^2}{2c\pi m_e}\left(\frac{1}{\nu_1^2} - \frac{1}{\nu_2^2}\right)\int_0^d n_e(l)dl$$

where e is the electronic charge, me is the electron rest mass, $n_e(l)$ is the charged particle density at a distance l along the line-of-sight between the pulsar and the Earth. The integral is known as the **dispersion measure, DM:**

$$\mathrm{DM} = \int_0^d n_e(l)dl$$

The discovery observation of a pulsar provides an initial estimate of the pulsar's pulse period, P, its dispersion measure, DM and its position. As repeated observations of a pulsar are made it becomes possible to measure more properties such as the changes in the pulse period (either apparent changes from orbital effects, or from an actual spin-down of the pulsar), its proper motion and many other effects.

In this book we will describe the methods and software packages used to observe pulsars and to process the resulting data files.

An overview of pulsar software is as follows:

- Observations taken in order to search for new pulsars produce pulsar **search-mode** data files. These data files can be viewed using packages like **pfits_fv** and processed using **sigproc** or **presto**. If a pulsar is observed then the pulses can be studied using **dspsr**.

- When a pulsar has been found subsequent observations can "fold" the data at the known period of the pulsar. Such data files are known as **fold-mode** data files. These files can be processed using the **PSRCHIVE** suite of software.

- The properties of the pulsar can be determined using programs like **pdmp** that improves estimates of the pulsar's pulse period and dispersion measure, **fitorbit** that provides initial estimates of binary parameters and using timing packages such as **tempo2**.

- The best available, published parameters for each pulsar can be obtained from the **psrcat** catalogue.

- Timing packages such as **tempo2** can also be used to search for gravitational wave signals and other physical effects within the data set.

- Various tools exist for simulating pulsar data sets. Simulations of pulse arrival times (for pulsar timing) can be made using **ptaSimulate**. Simulations of pulsar search data sets can be made using **sigproc** or with **simulatePsr**.

The pulsar catalogue

The Australia Telescope National Facility (ATNF) pulsar catalogue was launched in 2005. The catalogue can be accessed via a web interface (http://www.atnf.csiro.au/research/pulsar/psrcat) or through a command-line interface[1]. The public catalogue only provides access to pulsar parameters that have been published. Note that the choice of parameters to include for each pulsar is slightly subjective - often a new timing model becomes available that is more precise in some parameters and less precise in some other parameters. The collators of the catalogue (currently Richard Manchester and Lawrence Toomey) decide which parameters to update. Most parameters contain a value, an uncertainty and a bibliographic reference. The bibliographic reference to each pulsar's name is the paper that described the discovery of that pulsar.

Basic use of the web interface to PSRCAT

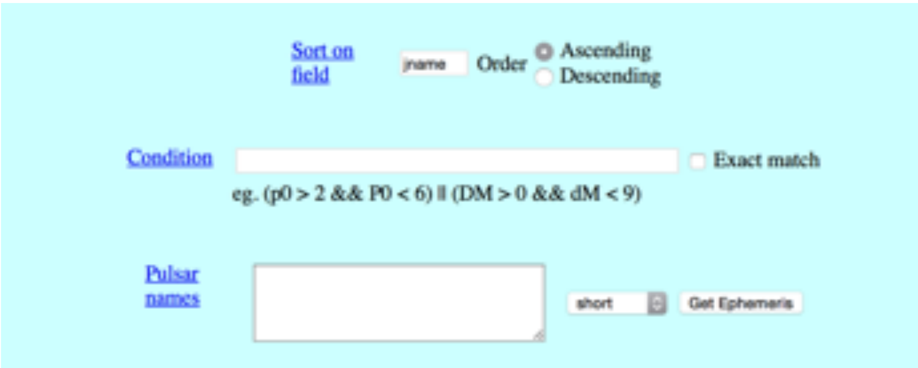
The web interface provides a list of parameters that can be selected. Each parameter is described in the documentation. This can be accessed by clicking on the parameter name or by clicking on "documentation" at the top of the screen. In order to get a list of the names, positions and period of all pulsars then select these parameters (Name, RaJ, DecJ, P0) and then select "TABLE". This returns a table like:

[1] The software can be downloaded from the web-interface.

#	NAME		RAJ (hms)			DECJ (dms)			P0 (s)		
1	J0006+1834	cnt96	00:06:04.8	2	cn95	+18:34:59	4	cn95	0.69374767047	14	cn95
2	J0007+7303	aaa+09c	00:07:01.7	2	awd+12	+73:03:07.4	8	awd+12	0.3158731909	3	awd+12
3	B0011+47	dth78	00:14:17.75	4	hlk+04	+47:46:33.4	3	hlk+04	1.240699038946	11	hlk+04
4	J0023+0923	hrm+11	00:23:16.87910	3	mnf+15	+09:23:23.871	1	mnf+15	0.00305	0	hrm+11
5	B0021-72C	mld+90	00:23:50.35311	9	fck+03	-72:04:31.4926	4	fck+03	0.005756779999551320	17	fck+03
6	B0021-72D	mlr+91	00:24:13.87934	7	fck+03	-72:04:43.8405	3	fck+03	0.005357573208686266	18	fck+03
7	B0021-72E	mlr+91	00:24:11.1036	1	fck+03	-72:05:20.1377	4	fck+03	0.003536329915276031	13	fck+03

The codes like cnt96, fck+03 are bibliographic references. Click on these links to obtain the full reference. The uncertainty on the parameters are, by default, listed as the uncertainty in the last decimal place of the parameter value.

On the main screen, it is possible to sort the data on a particular parameter, to provide a conditional statement or to provide a list of pulsar names:



The conditional statements are similar to statements in "C" or similar programming languages. The "&&" represents an "AND" clause. A "||" represents "OR". For instance:

(p0 > 2 && p0 < 6)

will list all pulsars with periods between 2 and 6 seconds.

The output format can be changed using the options under "Output Style". A simple output can be obtained using "Short without errors":

#	NAME	RAJ (hms)	DECJ (dms)	P0 (s)
1	J0006+1834	00:06:04.8	+18:34:59	0.693748
2	J0007+7303	00:07:01.7	+73:03:07.4	0.315873
3	B0011+47	00:14:17.7	+47:46:33.4	1.240699
4	J0023+0923	00:23:16.8	+09:23:23.8	0.003050
5	B0021-72C	00:23:50.3	-72:04:31.4	0.005757

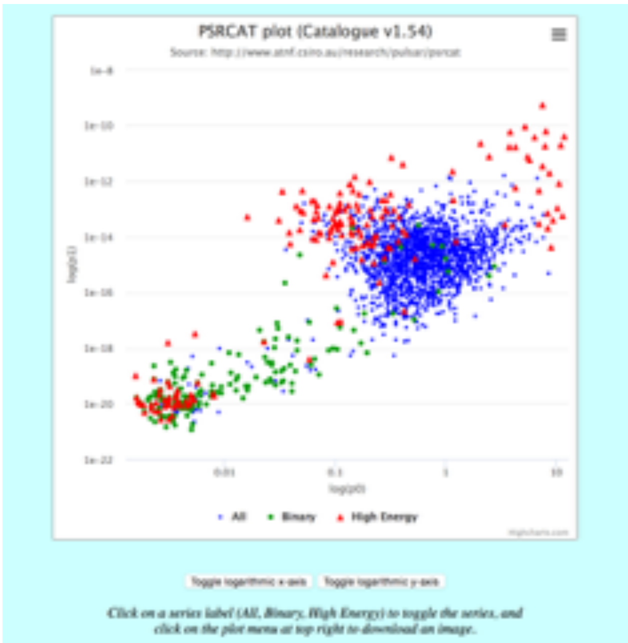
It is also possible to obtain output in CSV format for loading into packages such as EXCEL.

PSRJ	J1713+0747	
RAJ	17:13:49.5331505	5.000e-07
DECJ	+07:47:37.49284	2.000e-05
DM	15.9700	
PEPOCH	53729	
F0	218.8118438547255	5.000e-13
F1	-4.08386E-16	5.000e-21
PMRA	4.918	2.000e-03
PMDEC	-3.914	5.000e-03
POSEPOCH	54971	
DMEPOCH	56466	
BINARY	DD	
PB	67.8251383185	1.700e-09
ECC	0.0000749402	6.000e-10
A1	32.34242187	1.300e-07
T0	53761.0328	3.000e-04
OM	176.1966	1.400e-03
EPHEM	DE421	
PBDOT	0.36E-12	1.700e-13
RM	+8.4	6.000e-01
PX	0.85	3.000e-02
SINI	0.9505	3.800e-03
KOM	88	2.000e+00
M2	0.286	1.200e-02
UNITS	TDB	

You can access all the parameters for an individual pulsar by typing its name into “Pulsar names” and then selecting “Get Ephemeris”. This gives an output (for J1713+0747) as given to the left.

This set of parameters for the pulsar can be used by timing software packages such as tempo and tempo2.

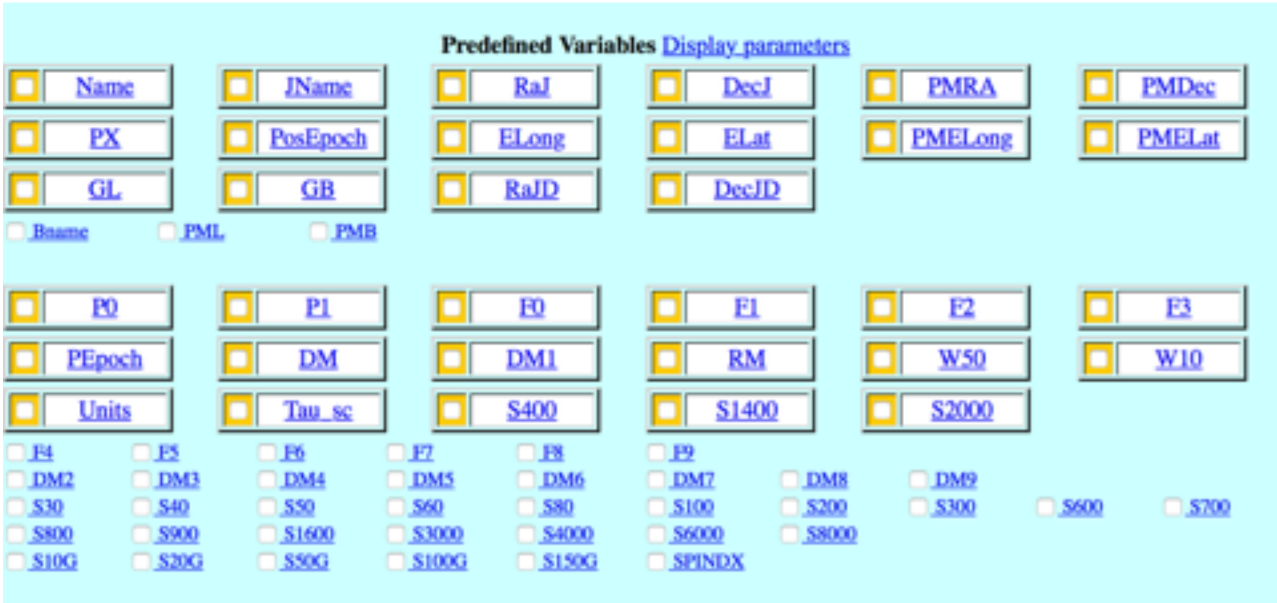
For compatible web browsers it is also possible to make simple plots of the data. For instance to make a P-Pdot diagram then use the “Plotted Output” options:



and then click on “PLOT” instead of clicking on “TABLE”. The plot is interactive. The mouse cursor identifies the closest pulsar and clicking the mouse and dragging zooms-in on a region of the figure. The different colours represent different types of pulsar (i.e., those in binary systems or pulsars with high energy detections).

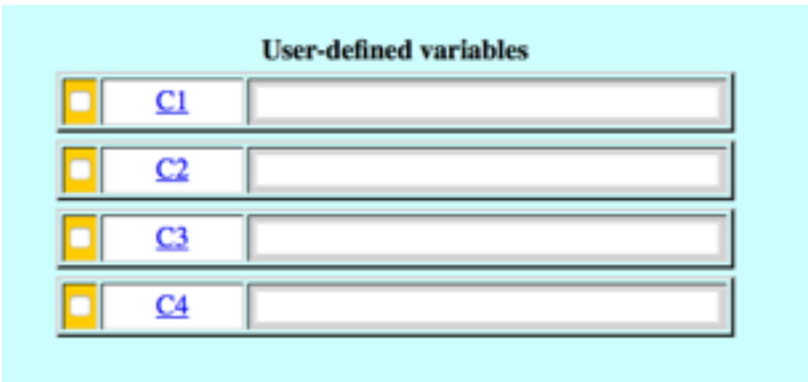
Expert use of the web interface to PSRCAT

The top menu bar allows the user to select “Expert”. This provides a larger selection of parameters to select.



In general these are parameters that are not commonly used and often only measured for a few pulsars.

It is also possible (with the expert and non-expert interface) to create user-defined parameters:

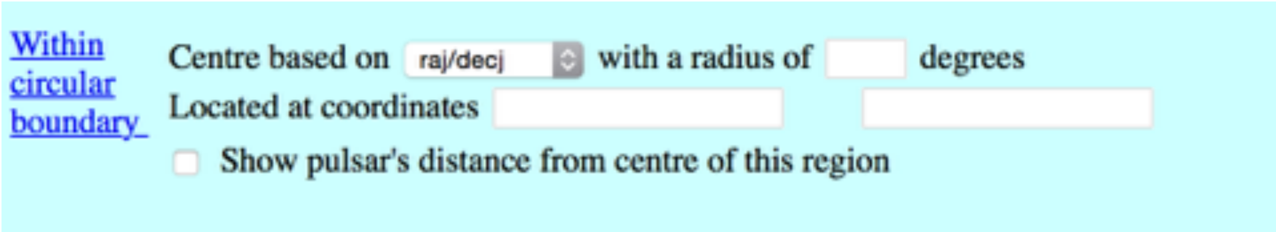


These can be analytic combinations of other parameters. For instance C1 could equal:

2*P0/W50

(i.e., twice the period divided by the pulse width at 50% of the pulse height). The parameter C1 can then be used just like any other parameter.

Sometimes the user needs to identify all pulsars within a specific region on the sky. This can be achieved using the "Within circular boundary" options in which the user presents the central coordinates and a radius. All pulsars within that radius are selected.



Use of the command-line interface

The PSRCAT software is written in C. The software can be downloaded (with installation instructions) from the web interface. The code generally compiles easily on linux or Mac systems. It is important to set $PSRCAT_RUNDIR and $PSRCAT_FILE as environment variables that point to the directory containing the catalogue file and the actual catalogue file respectively.

To test you PSRCAT installation type:

psrcat -h

This should provide help information about the use of the software. To get a tempo/tempo2 parameter file for a specific pulsar (this example is for PSR J0437-4715)

psrcat -E J0437-4715

This returns:

```
PSRJ            J0437-4715
RAJ             04:37:15.883250          3.000e-06
DECJ            -47:15:09.031863         3.700e-05
DM              2.64476                  7.000e-05
PEPOCH          52005
F0              173.687946184768         3.000e-12
F1              -1.728408E-15            3.000e-21
PMRA            121.679                  5.200e-02
PMDEC           -71.820                  8.600e-02
POSEPOCH        54100.0
BINARY          DD
PB              5.74104646               2.000e-06
ECC             1.9180E-5                7.000e-09
A1              3.36669708               1.400e-07
T0              52009.852429             7.800e-04
OM              1.2224                   4.900e-02
OMDOT           0.01600                  8.000e-03
START           50191.0
FINISH          53819.2
CLK             TT(TAI)
EPHEM           DE405
```

```
PBDOT          3.73E-12            6.000e-14
RM             0.0                 4.000e-01
PX             6.396               5.400e-02
KOM            207.8               6.900e+00
KIN            137.58              2.100e-01
M2             0.254               1.800e-02
UNITS          TCB
```

The -c command line option enables different parameters to be selected. For instance:

psrcat -c “jname p0 dm s1400”

lists each pulsars’ Jname, period, dispersion measure and flux density at 1400 MHz:

```
-------------------------------------------------------------------------------------------------------------------------------
#     PSRJ                    P0                                     DM                           S1400
                              (s)                                    (cm^-3 pc)                   (mJy)
-------------------------------------------------------------------------------------------------------------------------------
1     J0006+1834   cnt96      0.69374767047           14  cn95       12.0           6  cn95       *           0  *
2     J0007+7303   aaa+09c    0.3158731909             3  awd+12     *              0  *          *           0  *
3     J0014+4746   dth78      1.240699038946          11  hlk+04     30.85          7  hlk+04     3           1  lylg95
4     J0023+0923   hrm+11     0.00305                  0  hrm+11     14.3           0  hrm+11     *           0  *
5     J0024-7204C  mld+90     0.00575677999551320     17  fck+03     24.599         2  fkl+01     0.6         0  mld+90

6     J0024-7204D  mlr+91     0.00535757328486266     18  fck+03     24.729         2  fkl+01     *           0  *
7     J0024-7204E  mlr+91     0.00353632915276031     13  fck+03     24.230         2  fkl+01     *           0  *
8     J0024-7204F  mlr+91     0.00262357935251098     14  fck+03     24.379         5  fkl+01     *           0  *
```

If a parameter has not been published for a given pulsar then the default output is a “*” symbol. This can be changed using the -null command line option. The output format can be modified. If the user does not wish to have the header, numbers, references or uncertainties then:

psrcat -c “jname p0 dm s1400” -o short -nonumber -nohead

This gives:

```
J0006+1834     0.693748       12.00          *
J0007+7303     0.315873           *          *
J0014+4746     1.240699       30.85       3.00
J0023+0923     0.003050       14.30          *
J0024-7204C    0.005757       24.60       0.60
J0024-7204D    0.005358       24.73          *
```

Examples of commonly used commands are given in the table below:

Command	Description
psrcat -c “name p0 dm”	Tabulates all pulsars with their names, periods and dispersion measures
psrcat -c "name p0 dm” -o short	As above, but only provide the parameters to a few decimal places and do not give uncertainties or references. Useful for obtaining text output for plotting in another package.
psrcat -c “name p0 dm” J1744-1134 J0437-4715	Only display the parameters for the two listed parameters.
psrcat -c “name p0 dm” -s p0	Sort based on the pulse period (the default is sorting on the pulsar names)
psrcat -c "name p0 dm" -l “p0 > 3 && dm > 40”	List parameters for pulsars in which the pulse period is greater than 3 sec and the dispersion measure greater than 40 cm^{-3}pc.
psrcat -c "name assoc” -l “assoc(GC)”	List all pulsars associated with a globular cluster
psrcat -c “name assoc" -l “assoc(SNR)”	List all pulsars associated with a supernova remnant.
psrcat -c "name type” -l “type(RRAT)”	List all pulsars known to be RRATs
psrcat -c "name type" -l “type(NRAD)”	List all pulsars that are not radio emitters (i.e., non-radio pulsars)
psrcat -c "name p0 s1400 c1” -c1 “p0*2/w50” -s c1	Define a new parameter (c1) that is the period*2/w50. Then display each pulsars’ name, period, flux density and the c1 parameter. Sort the table by the c1 parameter.
psrcat -c “jname” -boundary “1 17:13:00 07:40:00 10”	List all pulsars within 10 degrees of the sky position (17:13:00, 07:40:00). The initial “1” in the boundary command states that these positions are in equatorial coordinates.
psrcat -c "name elong elat gl gb”	Tabulate the names, ecliptic longitudes and latitudes and Galactic longitude and latitudes for each pulsar.
psrcat -p	Lists all the possible parameters that can be tabulated.

Derived parameters

The pulsar catalogue is based on published values (such as positions, pulse frequencies, dispersion measures, etc.). The catalogue software can calculate commonly used parameters that are derived from these values (such as characteristic ages, surface magnetic fields etc.). The derived parameters are listed in the following table:

Label	Parameter	Equation
P0, F0	Pulse period (P0) and frequency (F0)	
P1, F1	Time derivative of the pulse period or frequency	
age	Characteristic age (yr)	$\tau = P/(2\dot{P})$
bsurf	Surface magnetic flux density (G)	$B = 3.2 \times 10^{19}(P\dot{P})^{1/2}$
EPS1, EPS2	ELL1 binary model parameters	EPS1 = ECC x sin(OM) EPS2 = EXX x cos(OM)
MinMass, MedMass	MinMass is the Minimum companion mass assuming i=90 degrees and neutron star mass is 1.35 Mo. Medmass is the median companion mass assuming i=60 degrees.	
Dist	Best available estimate of the pulsar distance as follows: • If DIST_A is set then use that parameter • If PX exists then use 1/PX if the parallax has been measured with >3 sigma significance. • If DIST_AMN and DIST_AMX exist and DIST_DM lies within the boundary then use DIST_DM else use the closest limit to DIST_DM • If DIST_DM is not defined then use the mean of DIST_AMN and DIST_AMX • Otherwise use DIST_DM	
Dist_DM	Distance based on the Taylor & Cordes (1993) electron density model	
DMsinb	Dispersion measure multiplied by sine(galactic latitude)	DM x sin(b)
XX, YY, ZZ	Distance from the Galactic plane (based on DIST) in kpc	
R_lum	Radio luminosity at 400 MHz (mJy kpc^2)	
R_lum14	Radio luminosity at 1400 MHz (mJy kpc^2)	
Edot	Spin down energy loss rate (ergs/s)	
Edotd2	Energy flux at the Sun (ergs/kpc^2/s)	
PMTot	Total proper motion (mas/yr)	
VTrans	Transverse velocity (based on DIST; km/s)	
Shklovskii corrections	P1_i, Age_i, Bsurf_i are the corresponding parameters after correction for the Shklovskii (proper motion) effect.	

Label	Parameter	Equation
B_lc	Magnetic field at the light cylinder (G)	

Creating your own catalogue

You may wish to make your own catalogue of particular pulsars. You can also append new discoveries or possible new candidates to the existing catalogue. The catalogue entries have the following form:

```
PSRJ     J0007+7303                      aaa+09c
RAJ      00:07:01.7                 2    awd+12
DECJ     +73:03:07.4                8    awd+12
F0       3.165827392                3    awd+12
F1       -3.6120E-12                5    awd+12
F2       4.1E-23                    7    awd+12
F3       5.4E-30                    9    awd+12
PEPOCH   54952                           awd+12
NGLT     1
DIST_AMN 1.1                             plmg93
DIST_AMX 1.7                             plmg93
TYPE     NRAD
ASSOC    GRS:2FGL_J0007.0+7303[naa+12],XRS:RX_J0007.0+7303[cdm+10],SNR:CTA1[hgc+04]
SURVEY   FermiBlind
@-----------------------------------------------------------------
PSRB     B0011+47                        dth78
PSRJ     J0014+4746
RAJ      00:14:17.75                4    hlk+04
DECJ     47:46:33.4                 3    hlk+04
POSEPOCH 49664.00                        hlk+04
PMRA     19.3                       18   bfg+03
PMDEC    -19.7                      15   bfg+03
F0       0.805997239145             7    hlk+04
F1       -3.6669E-16                9    hlk+04
F2       7.3E-28                    143  hlk+04
PEPOCH   49664.00                        hlk+04
DM       30.85                      7    hlk+04
S400     14                         2    lylg95
S600     9                          2    lylg95
S1400    3                          1    lylg95
SPINDX   -1.3                       4    lylg95
W50      88.7                            lylg95
W10      142.5                           lylg95
DIST_DM  1.82                            tc93
DIST_DM1 1.56                            cl02
SURVEY   gb1,gb2,gb3
```

Each entry is written in plain-text. Every entry is delimited using "@——-" characters. The first column is the parameter name. Every pulsar must have a "PSRJ" entry giving the pulsar's J-name. The second column is the parameter value. The third column is the uncertainty as the error on the last given decimal place. This column is optional. The fourth column (or the third column, if the uncertainty is not provided) is the bibliographic reference.

Note: it is common for users to use the wrong format for the uncertainties. This can lead to PSRCAT crashing. Ensure that the uncertainties are a single digit as defined above.

If a user has created a completely new catalogue file then that can be used instead of the published catalogue file as:

psrcat -c “name p0 dm” -db_file mypath/mycatalgoue.db

If the user wishes to append the new catalogue file to the published catalogue then use:

psrcat -c “name p0 dm” -merge “mycat*.db”

(The above line will merge all the catalogue files with filenames starting with mycat in the current directory). If multiple catalogue files are merged then they are combined in alphabetical order. For instance, if the parameter P0 is set in “aaa.db” then that will overwrite the P0 parameter in the published catalogue. However, if P0 is also set in a file called “bbb.db” then that parameter will overwrite the value in both the published catalogue and in aaa.db.

Extra catalogue files can be included in $PSRCAT_RUNDIR. Instead of specifying these individually using the -merge option it is possible to use -all:

psrcat -all -c “name p0 dm”

and this will use all the available catalogue files in $PSRCAT_RUNDIR. This is used at the Parkes observatory to enable observers to include new candidates or improved parameters easily for the observing system to access.

By default the parameters are simply appended. However, in a few cases it is necessary to remove all current parameters for a given pulsar before giving a new set of parameters. For instance, it may be that the original catalogue has a P0 value for the pulse period, but the new set of parameters include F0 for the pulse frequency. To do this add:

PSRCAT_CLEAR

into the catalogue file before entering new parameters. In all cases it is worth checking that the parameters are as expected by running

psrcat -all -E <pulsar name>

before continuing with the data processing.

Publishing results from the catalogue

In your paper please provide a citation to the original publication describing the parameters that you are using. The bibliographic references can be accessed using both the web-interface and the command-line interface. We also appreciate a citation to Manchester, Hobbs, Teoh & Hobbs (2005; http://adsabs.harvard.edu/abs/2005AJ....129.1993M).

Acknowledgments for the catalogue

The pulsar catalogue (PSRCAT) format and initial parameters were based on an earlier catalogue system known as PSRINFO. The software for PSRCAT was mainly developed by George Hobbs, Albert Teoh and Jonathan Khoo. The parameters have been kept up-to-date by Richard Manchester, Maryam Hobbs, Lucyna Chudczer, Lawrence Toomey and Ryan Shannon.

How telescopes record pulsar data

digitisation

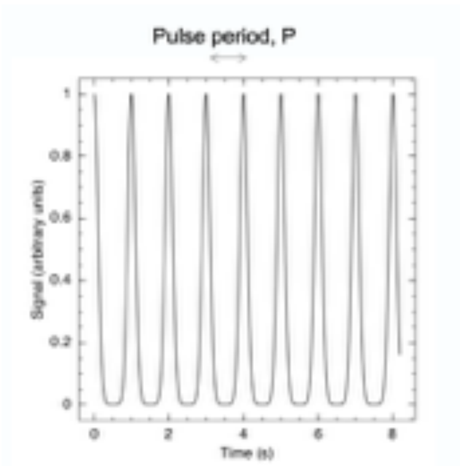
Spectral analysis

Baseband data

Pulsar searching

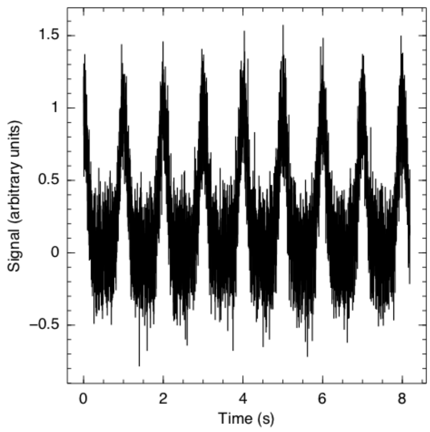
Pulsars are found by recording the telescope output with a large number of frequency channels and with high time resolution. Modern surveys may have hundreds or thousands of frequency channels and time resolution around 64us. For typical observing durations of minutes to hours the amount of data is enormous. A survey covering a large area of sky can now create TBs or even PBs of data.

An idealised pulsed signal looks like the figure below:



This figure defines the pulse period as the time between adjacent pulses. For known pulsars this time is between 1ms and ~10 seconds.

Of course, an actual pulsar observation would include noise. This amount of noise will depend upon the sky temperature in the direction of the pulsar and also properties of the telescope and observation. A slightly more realistic pulse train is therefore shown as the second figure.

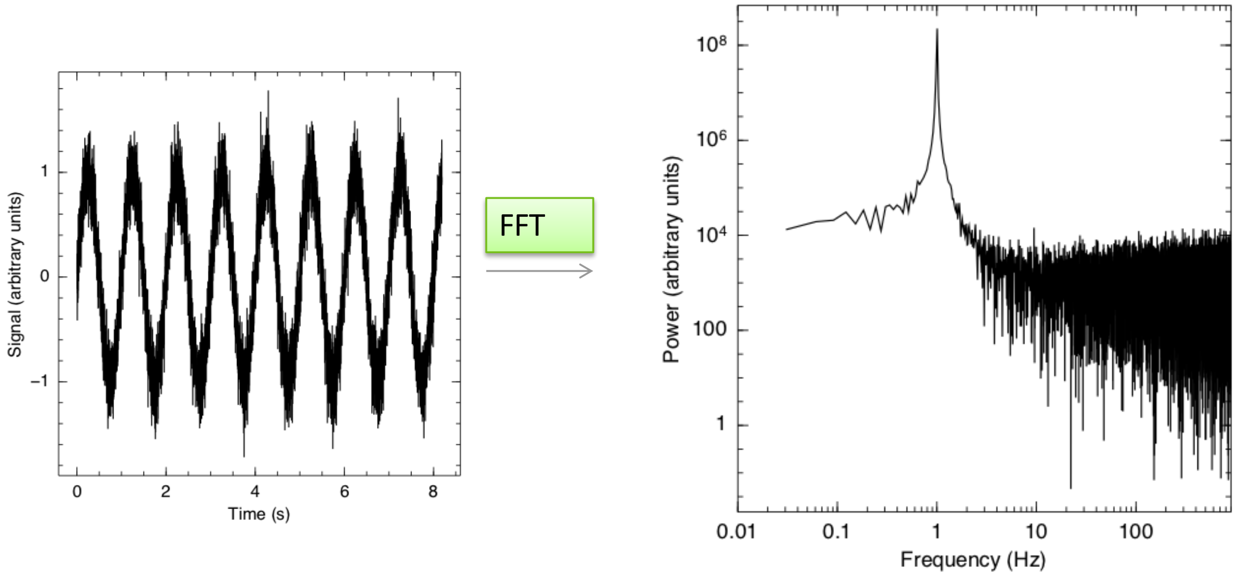


In order to discover a pulsar it is therefore necessary to:

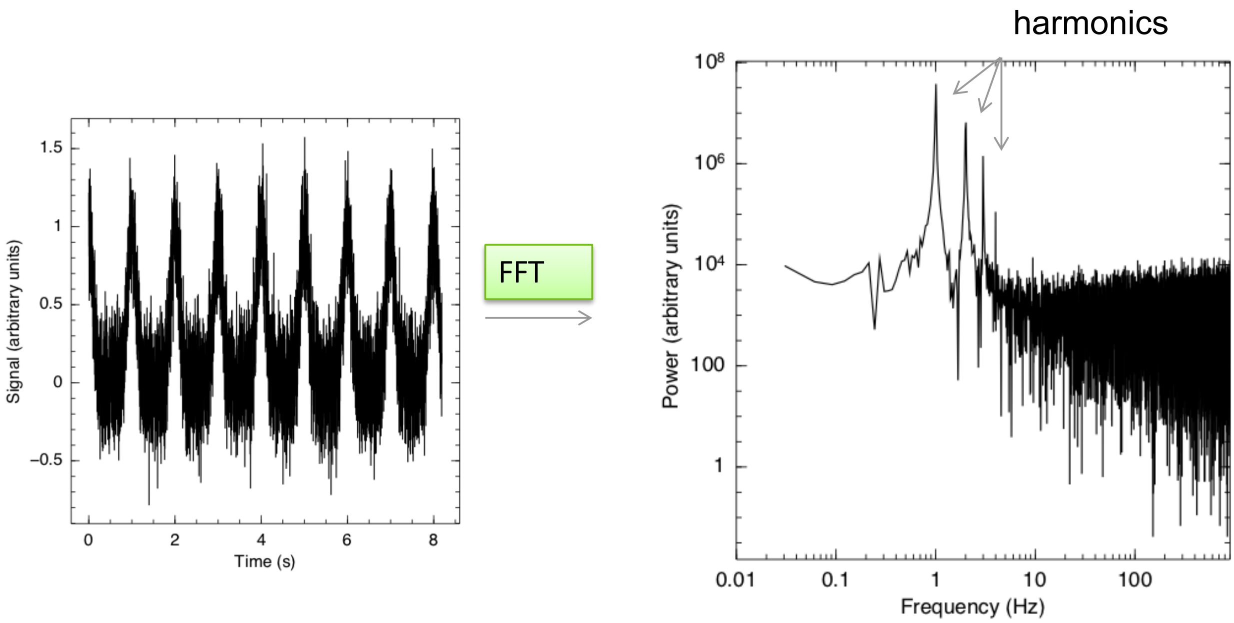
- detect the periodic pulses, or
- detect at least one individual pulse.

both methods are now in common use and will be discussed below.

The standard method to detect a periodic signal is to calculate a Fourier transform of the time series.



In the right-hand figure above we should the power spectrum of the Fourier transform (on logarithmic axes) of a pure sinusoid. As expected this leads to a single peak in the power spectrum at the frequency of the sinusoid. Pulsar signals are not pure sinusoids and therefore we observe harmonics in the power spectrum of a pulse sequence:



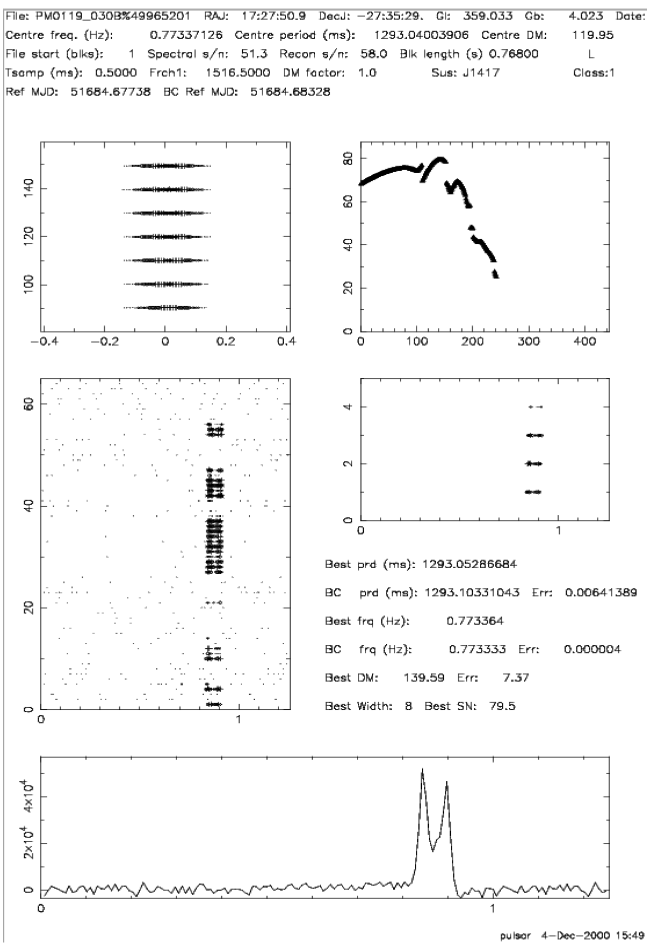
The number of harmonics (that occur at $2f_0$, $3f_0$, $4f_0$ etc., where f_0 is the fundamental frequency) depends upon the pulse shape and width.

As we have multiple frequency channels we need to remove the expected dispersion and then sum the channels before applying the FFT. For a known pulsar the amount of dispersion is known. It is therefore trivial to shift each channel by the known time delay and then sum the channels to produce a single, de-dispersed time series. If the dispersion measure is not known (which is the usual case in pulsar searching[2]) then a range of dispersion measures must be searched.

The basic search method (for periodicity searches) is to:

- Record some data
- De-disperse at a range of possible dispersion measures
- Sum across frequency channels to form a single time series for each dispersion measure
- Fourier transform each time series
- Relate harmonic features in the power spectrum (this is usually known as harmonic summing)
- Look for periodic signals

These periodic signals are known as pulsar candidates and are often plotted to be inspected by eye or using a Machine Learning algorithm. An example pulsar candidate is shown below:



This pulsar candidate was obtained from the Parkes multibeam pulsar survey. The parameters for the observation (date, position etc.) are listed at the top of the figure. The bottom figure gives the pulse profile that has been folded at the candidate pulse frequency. In this case a clear, bright pulsar is observed.

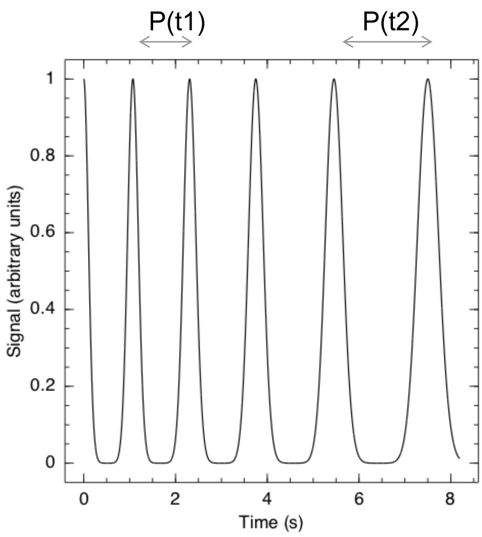
The other parameters show the pulse amplitude as a function of time (this is a nulling pulsar - it switches on and off in time) and as a function of frequency.

The top right figure shows the signal-to-noise of the pulse as a function of dispersion measure. Radio-frequency-interference would be expected to have a peak S/N at DM = 0 cm^{-3}pc, whereas a pulsar have a non-zero DM.

[2] The dispersion measure may be known if a globular cluster is being searched for new pulsars. Existing pulsars within the cluster will provide a reasonable estimate of the dispersion measure for any new pulsar.

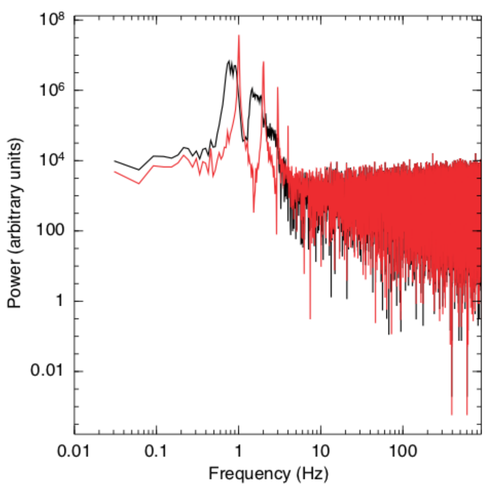
In order to confirm that a pulsar candidate is real, it is necessary to re-detect the same candidate using another observation of the same sky position. It is also necessary to confirm that the observed candidate is not an already known pulsar (checking for the exact period is straight-forward, but the candidate period may be a harmonic of the actual pulsar period).

If the candidate is confirmed then it is common to "time" the pulsar over a year in order to determine the pulsar's position, spin period, slow down rate etc. and get an initial determination of the timing precision achievable and the type of noise present in the pulsar timing residuals.



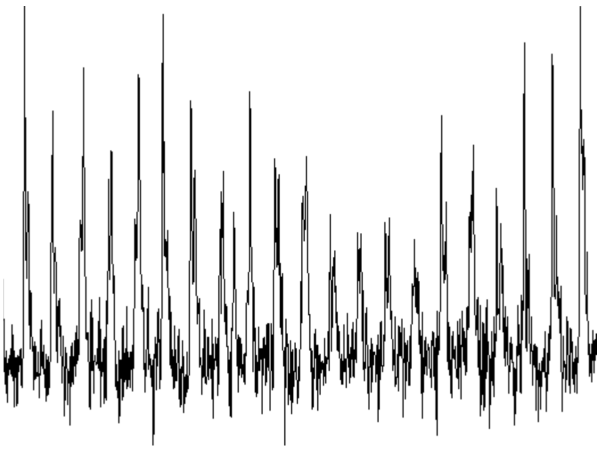
For a pulsar in orbit with a companion then the pulse sequence will not have a simple pulse period - the observed pulse train will be Doppler shifted by the orbital motion.

The power spectrum for a pulsar in a binary system will be smeared and finding binary pulsars gets harder and harder as the orbital period gets shorter. Various acceleration search methods exist that de-Doppler the time series, but these methods are very computationally expensive.



Red = solitary pulsar, black = binary pulsar

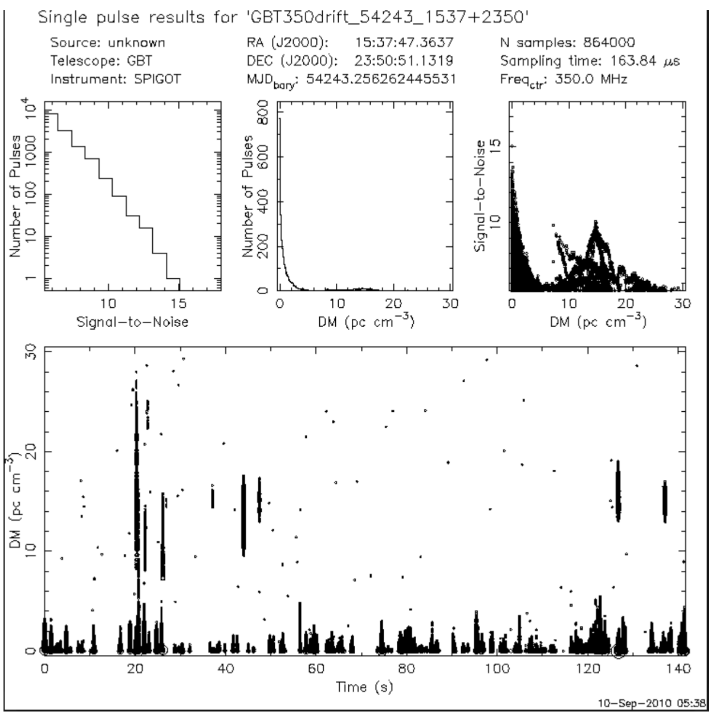
In the idealised pulse sequences drawn in the figures above each pulse has the same shape and amplitude. However, as shown in the figure below the single pulses from a real pulsar can vary significantly.



Some of the individual pulses can be much brighter than the average. This provides the possibility of searching for individual, bright pulses instead of searching for the pulse periodicity.

This was first attempted on a major survey by McLaughlin et al. (2006) who carried out a single pulse search of the Parkes Pulsar Multibeam Survey. She discovered single pulses without a clear periodicity, but at the same dispersion measure in some sky directions. These sources were termed Rotating Radio Transients (RRATs). Initially it was not clear exactly what RRATs were, but they are now understood to be bright single pulses from a pulsar. The working definition of an RRAT is "a pulsar which is more easily discoverable in a search for bright single pulses."

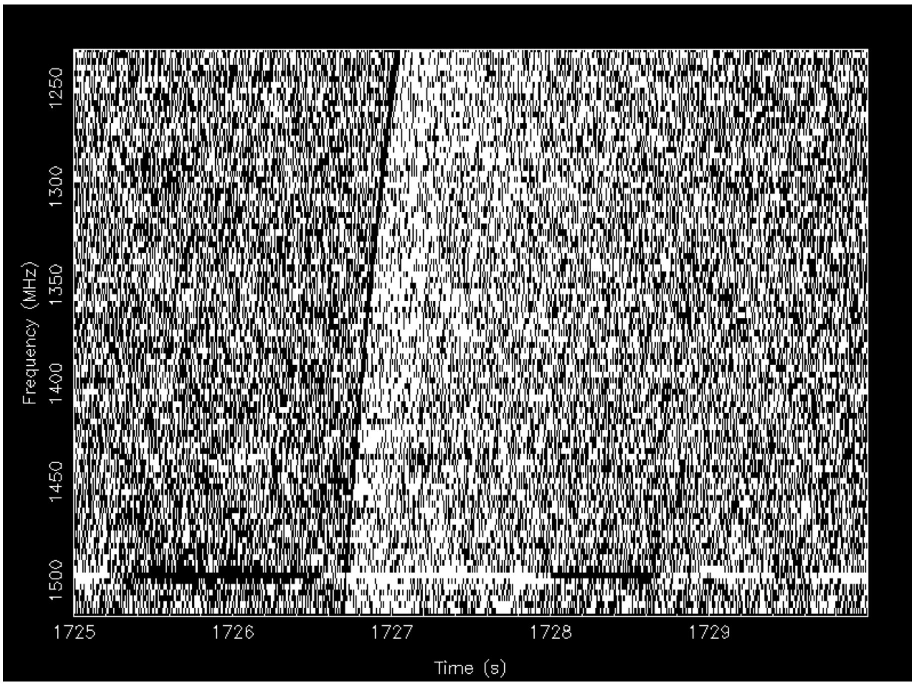
An example RRAT from the GreenBank telescope is shown below:



The bottom panel shows the individual pulses that were detected as a function of time (x-axis) and dispersion measure (y-axis). A large number of pulses are seen at zero-DM. These correspond to local radio-frequency-interference. Another group of pulses are detected at DM ~ 15cm^{-3}pc. These are coming from the RRAT.

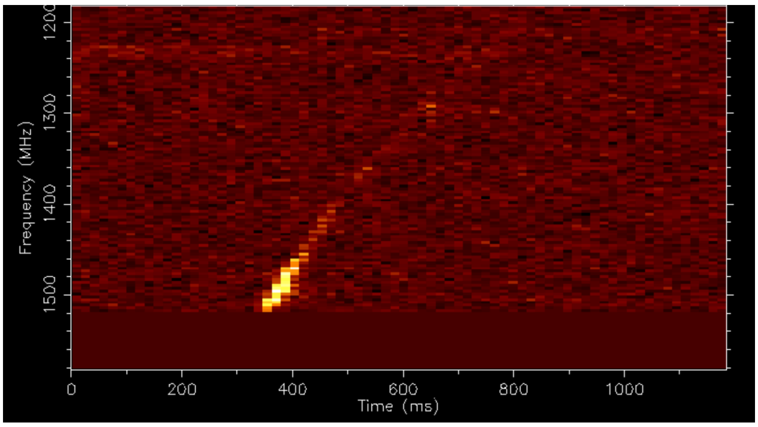
RRATs have dispersion measures that are similar to known pulsars and can be explained as bright pulses coming from objects in our Galaxy.

However, Lorimer et al. discovered a bright pulse in a survey of the Magellanic Clouds that had such a high dispersion measure that it could not be coming from our Galaxy. The pulse was extremely bright and can be seen in the raw data file:



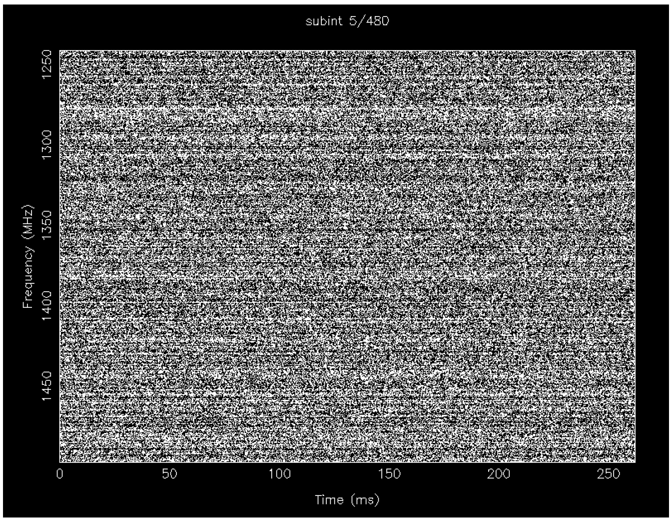
In this figure, time is shown in the x-axis and frequency in the y-axis. The data is 1-bit digitised and therefore the signal in each time and frequency segment is either a 0 or a 1 (black or white in the figure). Some radio-frequency-interference is seen around 1500 MHz. The detected burst event is seen as the black line that goes across the plot. The curvature is caused by the dispersion (note that high frequency is shown at the bottom of the plot and low frequency is at the top). This burst is known as a Fast Radio Burst (FRB) and understanding such bursts is an active area of current pulsar research.

Observers must be very careful when studying individual burst events as local interference at (or around) the observatory can also produce similar events. For instance, the following figure has a similar (but not identical) form to the FRBs, but is known to come from microwaves that used to be on the Parkes observatory site. Such signals were termed "perytons". Even though perytons are now understood, identifying and mitigating radio frequency interference is a major problem for current pulsar surveys.



Search mode data files

Pulsar search-mode data files store the total intensity pulsar signal as a function of time and frequency channel (in a few cases, polarisation is also recorded). As the observations can be long it is common to only record the data using a small number of bits (typically 1 or 2). The figure below shows this graphically:



In this figure the data have been 1-bit digitised and so each value is either a 0 or a 1 (black or white). This figure only shows ~250ms of data whereas typical search mode observations can last minutes or hours.

The data are usually recorded in a binary data file in which all the values are given for all the frequency channels for the first time interval and then the same for the second time interval etc. i.e.,

f1(t1) f2(t1) f3(t1) f4(t1) … fN(t1) f1(t2) f2(t2) f3(t2) f4(t2) … fN(t2) … f1(tM) f2(tM) … fN(tM)

where fN(tM) represents the signal in frequency channel N and time M. Some file formats (such as the SIGPROC format) write all the data usually without any breaks. The PSRFITS format writes NSBLK time intervals (usually a value like 2048) before writing further header information. This means that the PSRFITS format files are slightly larger than SIGPROC files, but contain more information.

The data files are usually written using bytes (8-bit values) and so, for 1-bit data, 8-frequency channels are packaged together to form a single byte. Note that the number of frequency channels must be divisible by 8.

Data file sizes can be easily estimated. For 64us time sampling and a 30 minute observation we require 30x60/64e-6 samples = 28125000. If we have 512 frequency channels then we require 28125000x512 values = 14400000000 values. For 1 bit data this requires 14400000000 bits = 1.6 GB. The data file will be ~10% larger because of the necessary header information.

Sigproc data format

The SIGPROC pulsar search software was developed by D. Lorimer and is available from http://sigproc.sourceforge.net. The software defines a straight-forward template for storing pulsar-search mode data that is in common use. Details are available in http://sigproc.sourceforge.net/sigproc.pdf. The data are stored as follows:

HEADER_START <header parameters> HEADER_END <data values>

The header parameters are as follows (please check with the SIGPROC documentation in case these have been updated):

- telescope_id (int): character code representing the telescope used to collect the data
- machine_id (int): code representing the backend system used
- data_type (int): 1 = filterbank, 2 = time series
- rawdatafile (char []): the name of the original data file
- source_name (char []) the name of the source being observed
- barycentric (int): 1 = data have been barcentred, 0 = they have not
- pulsarcentric (int): 1 = data are pulsarcentric, 0 otherwise
- az_start (double): telescope azimuth at the scan start (degrees)
- za_start (double): telescope zenith angle at scan start (degrees)
- src_raj (double) right ascension (J2000) of source (hhmmss.s)
- src_dej (double): declination (J2000) of source (ddmmss.s)
- tstart (double) time stamp (MJD) of first sample
- tsamp (double): time interval between samples (s)
- nbits (int): number of bits per time sample
- nsamples (int): number of time samples in the data file (often ignored)
- fch1 (double): centre frequency (MHz) of the first filterbank channel
- foff (double): filterbank channel bandwidth (MHz)
- FREQUENCY_START (character): start of frequency table
- fchannel (double): frequency channel value (MHz)
- FREQUENCY_END (character): end of frequency table
- nchans (int): number of frequency channels
- nifs (int): number of separate IF channels
- refdm (double): reference dispersion measure ($cm^{-3}pc$)
- period (double): folding period (s)

Extracting a particular IF, frequency channel and time sample from the data stream can be obtained from element number:

s x nIFs x nChans + i x nChans + c

where s is the time sample required, i is the IF channel and c is the frequency channel. The frequency corresponding to channel c is:

fch1 + c x foff.

PSRFITS search mode data format

The PSRFITS data format (http://www.atnf.csiro.au/research/pulsar/index.html?n=Main.Psrfits) can be used to store both pulsar search- and fold-mode data. The PSRFITS documentation (http://www.atnf.csiro.au/research/pulsar/index.html?n=PsrfitsDocumentation.Txt) provides indepth details of the data format.

GOT TO HERE
GIVE EXAMPLE cfitsio code.

Processing pulsar search-mode data

Getting initial pulsar parameters

fitorbit, gridding, parameters known from the search

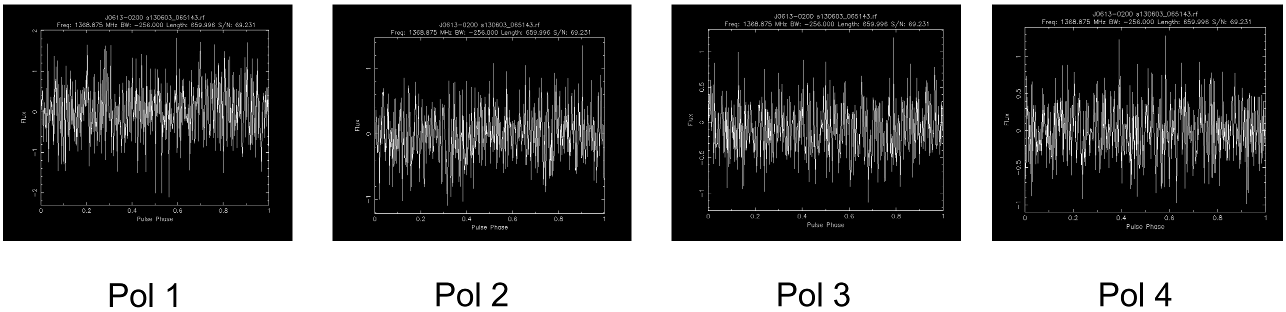
Pulse folding

dspsr

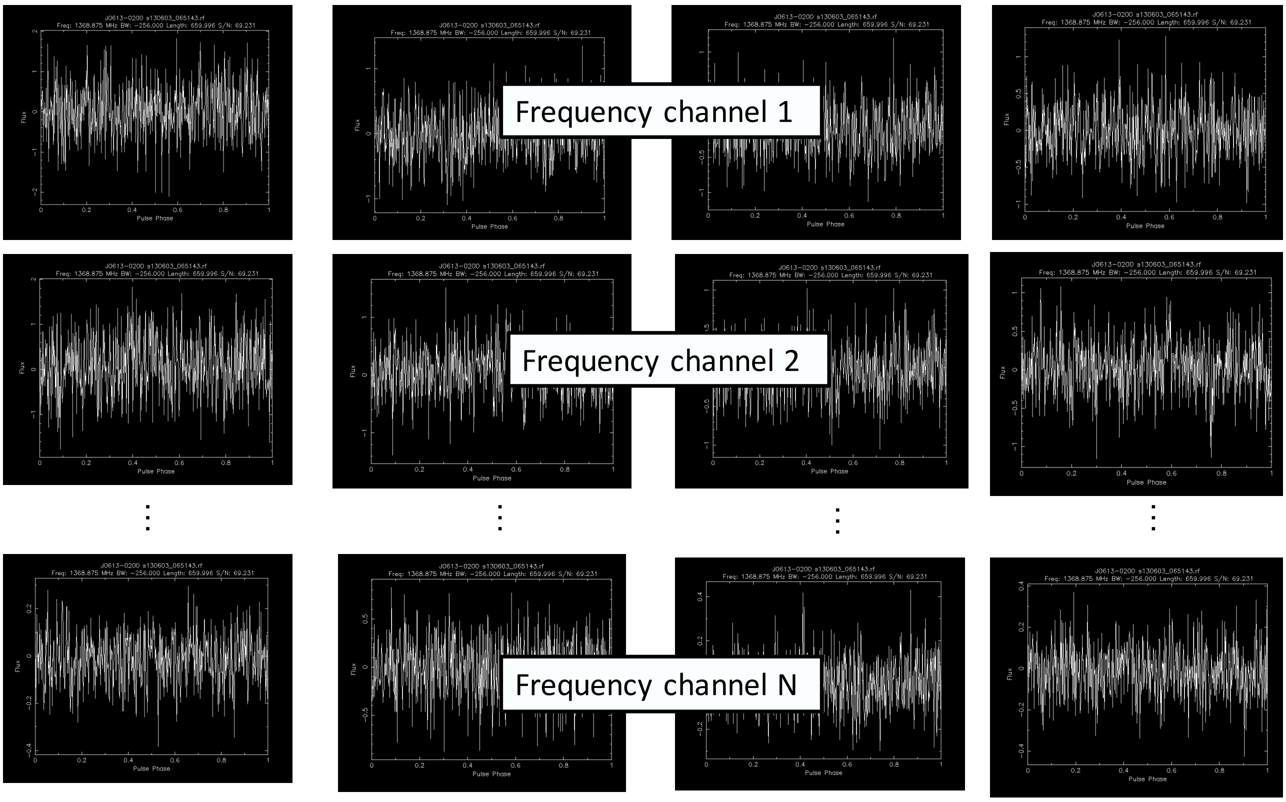
PSRFITS fold-mode files

We record the pulse profile as function of observing frequency. The pulse profile is recorded with a certain number of phase bins. In order to describe the pulse profile entirely it is necessary to record polarisation information. Pulse profiles are usually uncalibrated and post-processing methods are required to form the four Stokes parameters. However, it is necessary to record four profiles for each frequency channel in order to enable this post-processing.

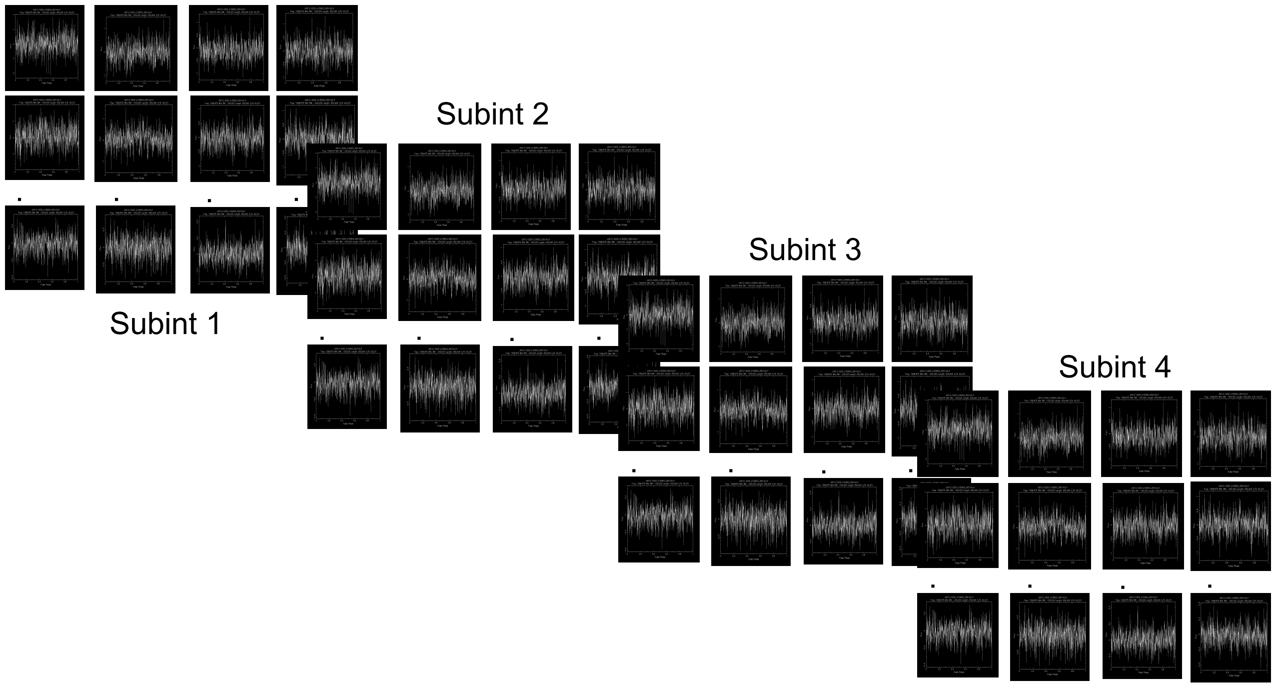
A data file is therefore split into four polarisation profiles:



Such profiles are recorded for each frequency channel:



It is possible for fold the data throughout the entire observation. However, it is more common only to fold relatively small segments of data. These are known as sub-integrations. If part of the data are corrupted by radio frequency interference then it is often only necessary to delete a few subintegrations instead of corrupting the entire observation.



How large is a PSRFITS fold-mode data file?

Each profile has N_b bins (let's assume that $N_b = 512$ which is typical for an observation). In the PSRFITS format each value is stored as a 2-byte value. Each profile is therefore 512x2 = 1024 bytes = 1kB.

We have N_p polarisation states (almost always $N_p = 4$). The data size to store each polarisation profile is therefore N_p x N_b = 1024x4 = 4kB.

We have N_c channels (a typical value is $N_c = 1024$). The data size is therefore N_p x N_b x N_c = 1024x4x1024 = 4MB.

This is the size of the file for a single subintegration. We have N_s subintegrations (for our example let's assume $N_s = 11$). The total data size is therefore N_p x N_b x N_c x N_s = 44MB.

The actual data size (this example was for the file s130603_061543.rf) is 45 MB. The extra MB corresponds to meta-data information in the PSRFITS headers.

The PSRFITS fold-mode format

PSRFITs provides a definition to produce virtual-observatory-compliant FITS files suitable for pulsar searching and folding. The description and definition can be viewed and a template format downloaded from http://www.atnf.csiro.au/research/pulsar/index.html?n=Main.Psrfits. The format is described by Hotan, van Straten & Manchester (2004; PASA, 21, 302).

The PSRFITS format is compliant with standard FITS and can be processed using standard FITS software tools. For instance:

- cfitsio is a library for C programs: http://heasarc.gsfc.nasa.gov/fitsio/fitsio.html
- pyfits/astropy are FITS reading routines for python: http://www.stsci.edu/institute/software_hardware/pyfits/

The PSRFITS format defines specific tables that can have header information and store the data:

PSRFITS - A summary of the current definition (V5.4)

HDU Name	Description
Main header	Observer, telescope and receiver information, source name and observation date and time
HISTORY	Date, program and details of data acquisition and each subsequent processing step
OBSDESCR	Free-format ascii description of the observation or signal processing
PSRPARAM	Pulsar ephemeris used to create or modify pulse profile data
POLYCO	History of the TEMPO polyco files used to predict the apparent pulsar period
T2PREDICT	The TEMPO2 predictor file used to predict the apparent pulsar period
COHDDISP	Parameters used for coherent dedisperion of baseband data
BANDPASS	Observed bandpass in each polarisation averaged over observation
FLUX_CAL	System temperature and injected noise calibration data as a function of frequency across the bandpass
CAL_POLN	Apparent polarisation of injected noise calibration signal as a function of frequency
FEEDPAR	Parameters of feed cross-coupling as a function of frequency
SPECKURT	Statistics for spectral kurtosis RFI excision
SUBINT	Pulse profiles or streamed data as a function of time, frequency and polarisation
DIG_STAT	Digitiser mode, attenuator settings and count statistics
DIG_CNTS	Digitiser mode and count rate distribution

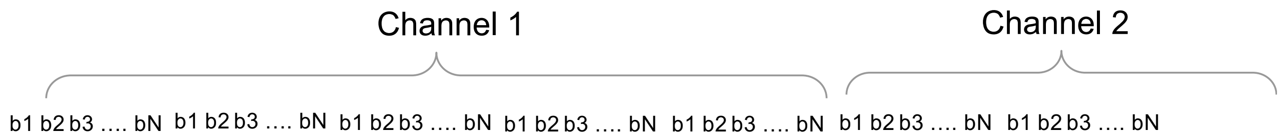
The pulse profiles are stores in the SUBINT table., This table includes a row for each sub-integration. Each row contains separate tables containing frequency-channel information and profile information.

In each sub-integration row the data are stored by first recording the first polarisation profile (with N_b bins):
b1 b2 b3 … b_{Nb}

The second, third and fourth polarisations are then stored:

$$\overbrace{\text{b1 b2 b3 …. bN}}^{\text{Pol 1}}\ \overbrace{\text{b1 b2 b3 …. bN}}^{\text{Pol 2}}\ \overbrace{\text{b1 b2 b3 …. bN}}^{\text{Pol 3}}\ \overbrace{\text{b1 b2 b3 …. bN}}^{\text{Pol 4}}$$

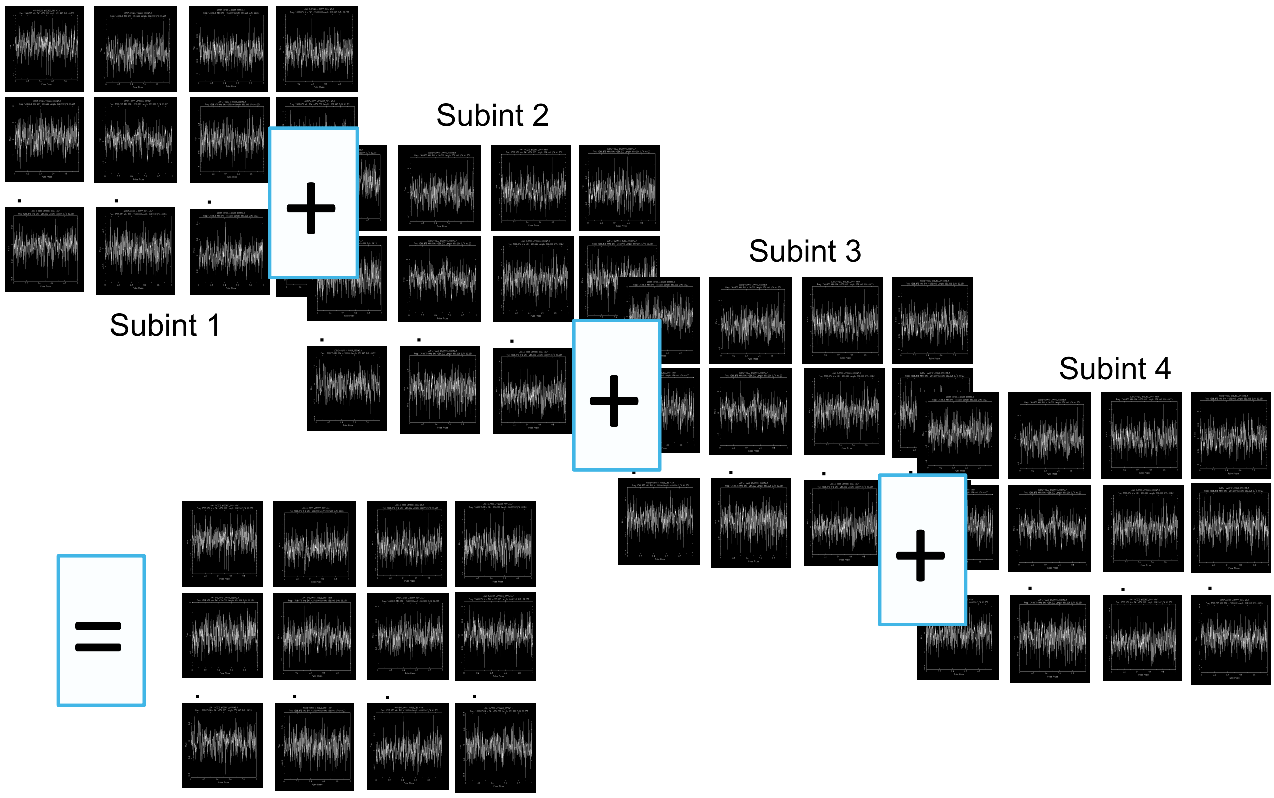
Each channel is then stored to give the data stream as follows:



Clearly these data files can be large. Often the science is not affected if the signal is summed in frequency (after de-dispersing the pulse signal), time (after removing impulsive RFI) or after forming Stokes I (the total intensity). Such summing is known as "scrunching".

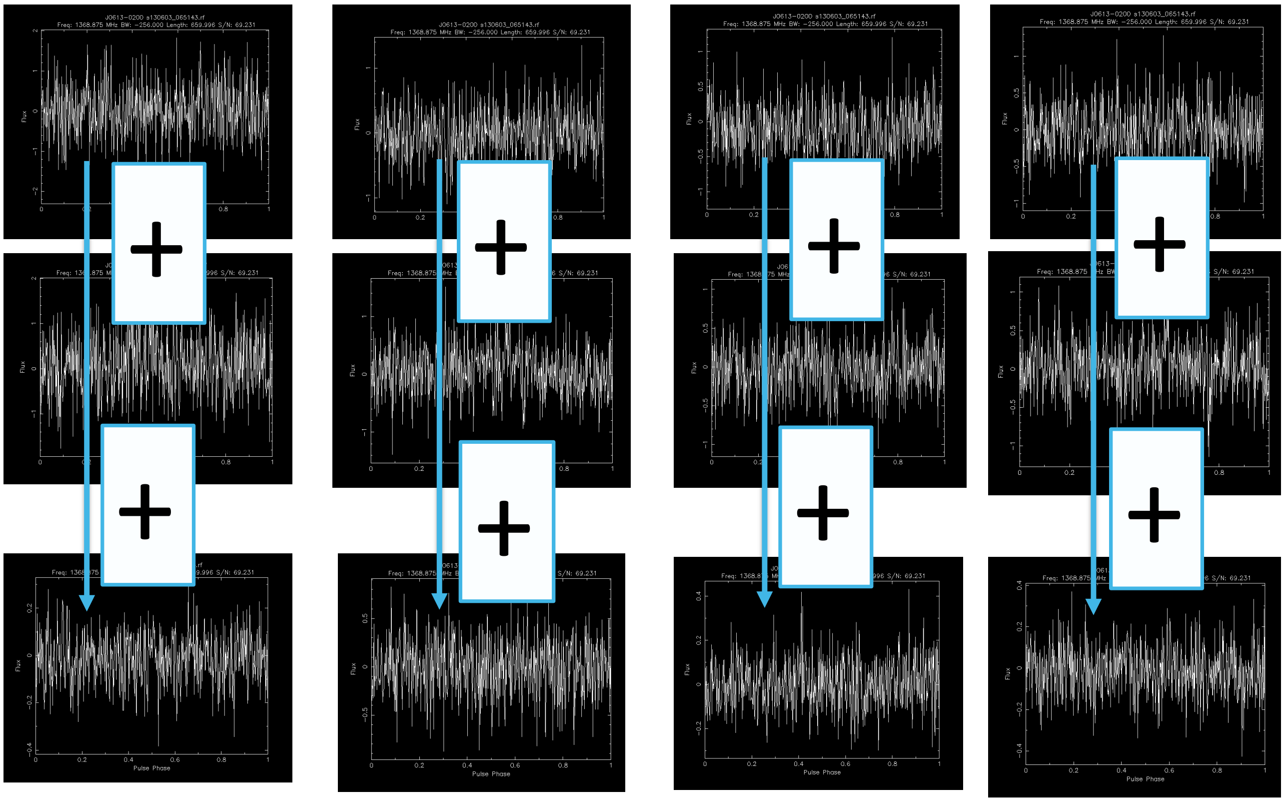
Frequency, time and polarisation scrunching

Scrunching in time creates a new profile with only a single subintegration.



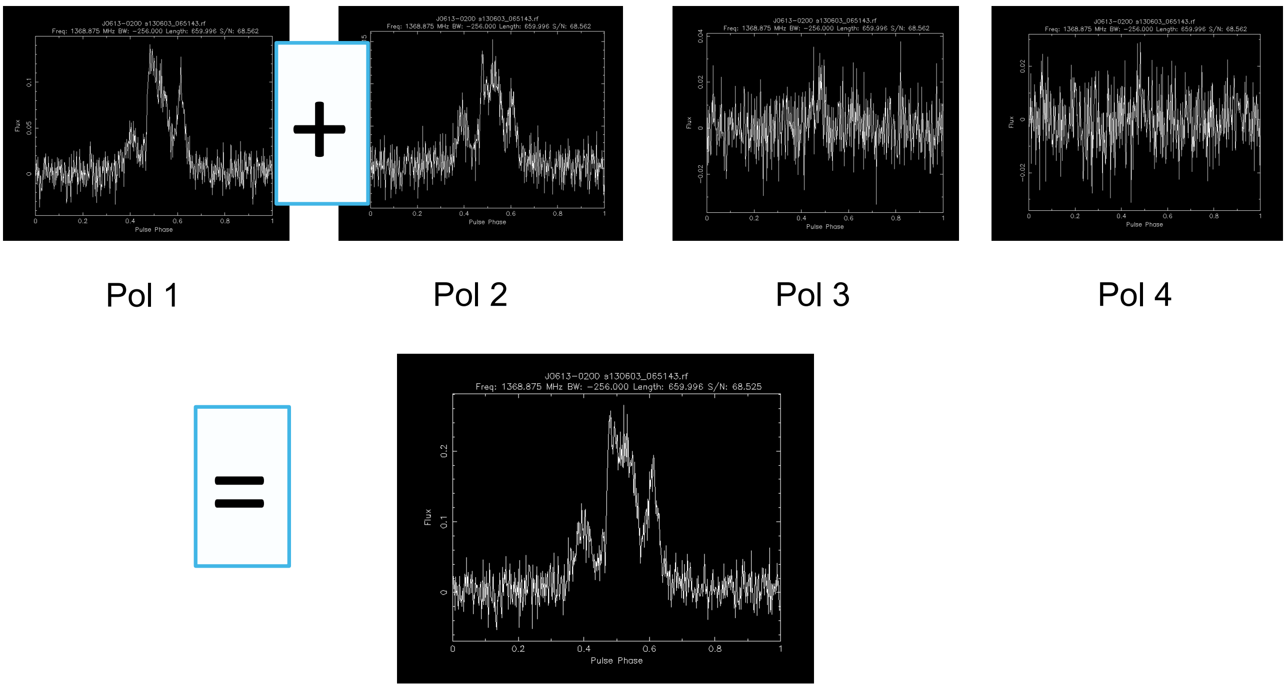
Usually time scrunching is carried out after removing interference and calibrating any time-dependent effects.

Scrunching in frequency reduces the number of frequency channels to a single channel. This should only be carried out after narrow-band interference has been removed and the pulse signal has been de-dispersed:



After time and frequency scrunching the remaining profile simply has 4 polarisation states.

After polarisation calibration methods have been applied the total intensity signal can be formed usually by summing the first two polarisation profiles:

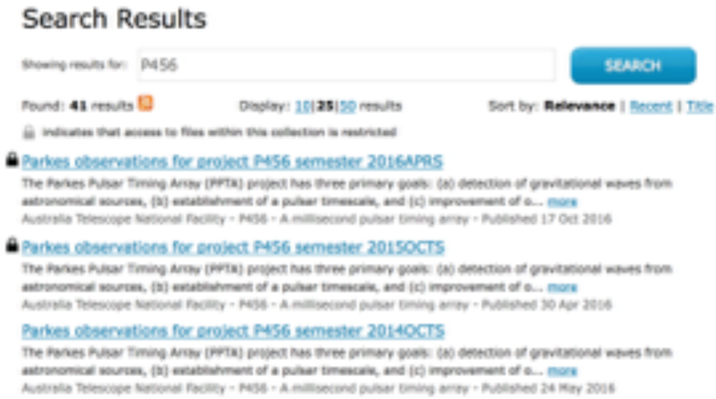


Such "scrunching" can be carried out using the PSRCHIVE software package. The file extension usually states whether any scrunching has been applied. For instance, the raw data file usually has the extension ".rf". A file with extension ".FTp" will have been frequency (F), time (T) and polarisation (p) scrunched.

Accessing pulsar data files from the CSIRO archive

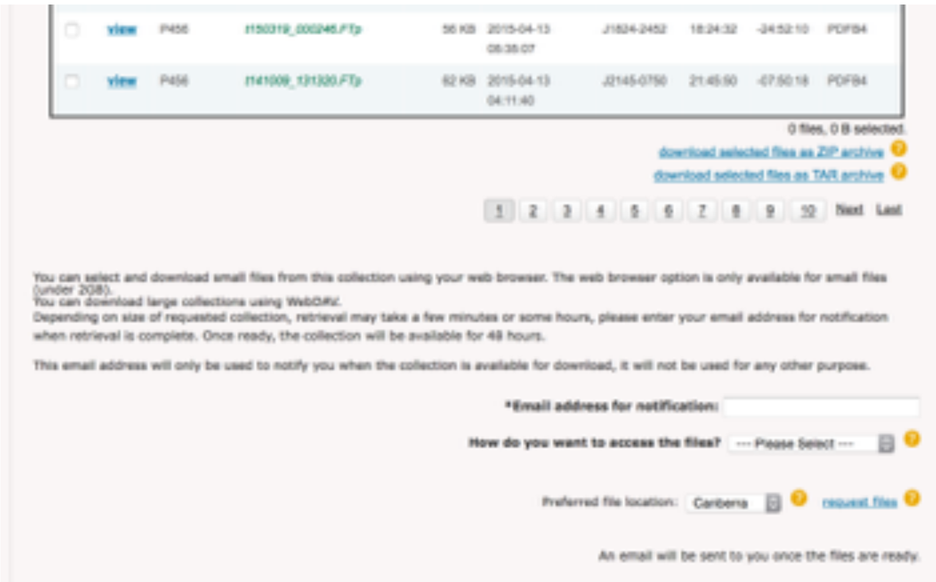
Pulsar observations produce large data files. Such data files are difficult to store and to access. The CSIRO has a policy to make all observations publically available from the Parkes radio telescope after an embargo period of 18 months (in rare instances this embargo period can be extended by the director).

The CSIRO data archive can be accessed through the Data Access Portal (DAP) using a web-based interface or via virtual observatory services. There are currently two calls each year for observing proposals with the Parkes telescope. Successful proposals are allocated a project code. The observations are subsequently carried out and all the observations that occurred during a particular semester are stored in a **data collection.** For instance, the Parkes Pulsar Timing Array project has project code P456. All data collections for this collection can be identified from https://data.csiro.au:



Collections that are still within their embargo period are identified using padlock symbols (see right-hand figure above). Some basic information about each data collection (even embargoed collections) can be viewed by clicking on the link representing that collection:

The data for publically available data can be downloaded by clicking on the "Files" tab at the top of the webpage (see figure above)[3]. You can download a few individual files (as a ZIP or TAR archive) using your web-browser. You can also download the entire collection. To do this you must enter you email address and then wait until you receive an email stating that the data collection is ready for download. For non-CSIRO users, the only access method currently available is to use WebDAV.



The project codes used at Parkes are simply in time order. In the following table we list the project codes for data collections currently available in the data archive:

[3] If you were on the original observing proposal then you can log-in to the system and access data that are embargoed for other people.

Project code	Description
P000	Parkes pulsar observations with undefined project IDs for 2013
P050	70cm pulsar survey and timing
P116	The 2004 periastron passage of PSR B1259-63
P138	Renewed Observations of PSR J0045-7319
P140	Precision Pulsar Timing
P236	Polarization and Rotation Measures of Recently Discovered Pulsars
P262	Timing of Young Pulsars
P268	Parkes multibeam survey
P269	A deep pulsar survey of the Small Magellanic Cloud
P276	Timing of multibeam pulsar discoveries
P282	Timing and Searching for Pulsars in 47 Tucanae
P309	An intermediate-latitude millisecond pulsar survey
P360	A high-latitude millisecond pulsar survey
P361	Studies of Relativistic Binary Pulsars
P362	Bumps and giants in the Vela pulsar
P393	Polarisation of the Vela and Mouse pulsars
P400	Geodetic precession in PSR J1141-6545
P417	A new class of pulsar-like neutron stars
P425	Intrinsic Rotation Measure Variations in Pulsars
P427	Timing and searching millisecond pulsars in globular clusters
P428	Subbeam-carousel rotation times in radio pulsars
P439	Properties of neutron star kicks
P450	X-ray and radio observations of giant pulses in PSR B0540-69
P452	High frequency polarimetry of radio pulsars
P453	Monitoring the interstellar medium with pulsars
P455 (P455_NOSK)	Timing & geodetic precession in the double pulsar and two relativistic binaries
P456 (P456_NOSK)	A millisecond pulsar timing array
P463	Geodetic precession in two new double neutron star systems
P468	Interstellar turbulence: strongly scattered pulsar radiation
P478	Monitoring of an eclipsing pulsar with a massive companion

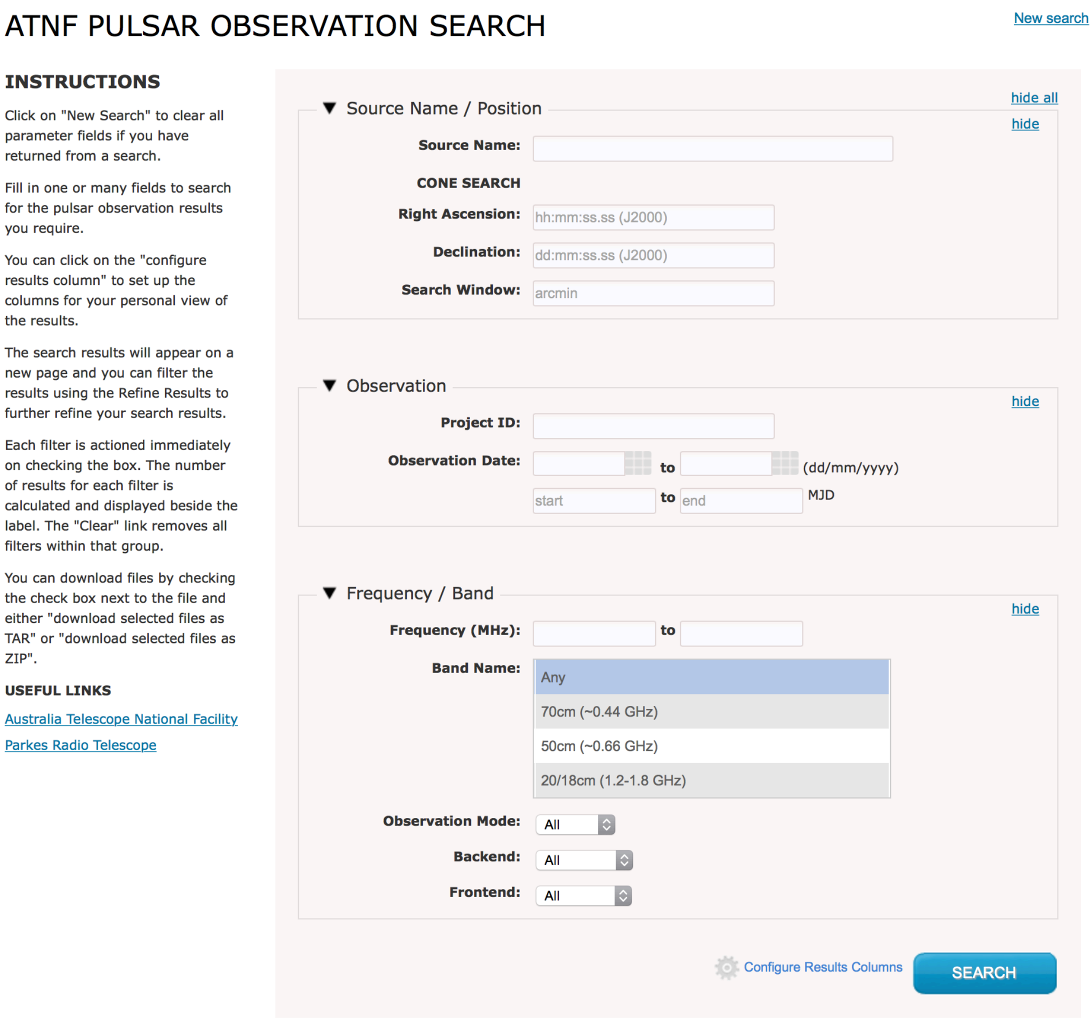
Project code	Description
P491	Pulsar searches of the Galactic Centre
P494	Integrated and individual radio pulsar properties
P499	Single pulse properties of millisecond pulsars
P501	Timing of binary and millisecond PKSMB/PH pulsars
P509	PSR J1833-1034, the Very Young Pulsar in SNR G21.5-0.9
P510	Timing and polarimetry of the relativistic binary pulsar J1906+0746
P512	A Methanol Multibeam Pulsar Survey
P535	A Census of Pulsar Emission
P554	Simultaneous Parkes/XMM observations of the AXP source XTEJ1810-197
P564	Studying the magnetar XTE J1810-197
P574 (P574_NOSK)	Pulsar timing and the GLAST mission
P575	Calibrating the Digital Filterbank System
P580	Parkes pulsar observations with undefined project IDs for 2007ik
P584	The polarization of Rotating Radio Transients
P594	A search for main- interpulse correlations in pulsars
P595 (P595_NOSK)	PULSE@Parkes (Pulsar Student Exploration at Parkes)
P602	An unusual radio pulsar
P617	Parkes 300 to 900 MHz Rotation Measure Survey: Pilot Study
P622	A stirring pulsar
P628	Testing of focal plane array technologies for ASKAP
P630	The High Time Resolution Universe
P661	The Search for and Confirmation of Nearby RRAT Candidates
P675	Radio search for gamma-ray pulsar counterparts
P682	A high time resolution survey of the Small Magellanic Cloud
P699	The Enigmatic Binary PSR J1723-28: A Baby Millisecond Pulsar?
P737 (P737_NOSK)	Commissioning the pulsar backends at Parkes
P741	High time resolution observations of CU Virginis
P743	Timing of New Magellanic Cloud Pulsars
P778	A new deep search for millisecond pulsars in globular clusters
P781	Observations of the magnetar J1622–4950 and other pulsars at 13-mm

Project code	Description
P786	Transient Radio Neutron Stars
P789 (P789_NOSK)	Timing of Binary & Millisecond Pulsars Discovered at Parkes
P791	Monitoring known X-ray magnetars for intermittent radio emission
P793	A Second Extragalactic Radio Burst: The Beginnings of a Population
P794	The Mass of PSR J1933-6211
P809	Parkes search for radio pulsars from unidentified Fermi Sources at 50 cm
P813	New Pulsars from Einstein@Home
P814	Millisecond pulsar searches in unidentified Fermi sources at high Galactic latitudes
P816	High-sensivity baseband observations of the Vela Pulsar: A rich dataset for rich science
P824	Confirmation and timing of new millisecond pulsars
P832	Bridging the Gap in Pulsar Modulation Timescales with PSR J1107-5907
P834	Searching for the pulsar in SN1987A
P835	Parkes pulsar observations with undefined project IDs for 2013
P842	High time resolution studies of giant pulses
P850	Studying intermittency in PSR J1717-4054
P852	Detecting high-energy gamma-ray emission from a newly-discovered magnetar
P855	A Parkes transit survey for pulsed radio emission during windstows and maintenance
P858	SUPERB - A SUrvey for Pulsars & Extragalactic Radio Bursts
P859	Searching towards the Galactic Centre region for pulsed radio emission
P860	Initial Follow-up of Pulsar Discoveries from the HTRU Galactic Plane Survey
P863	Analysis of state switching pulsars
P864	A Search for the Intergalactic Magnetic Field
P865	Commensal searches for microhertz gravitational waves and fast radio bursts: A pilot study
P866	Candy from the 47 Tuc shop
P867	Searching for Radio Millisecond Pulsars in a New Set of Fermi Sources
P868	Simultaneous X-ray/radio observations of the mode-switching PSR B0943+10
P869	Ultra-Deep searches for pulsars around Sgr A*

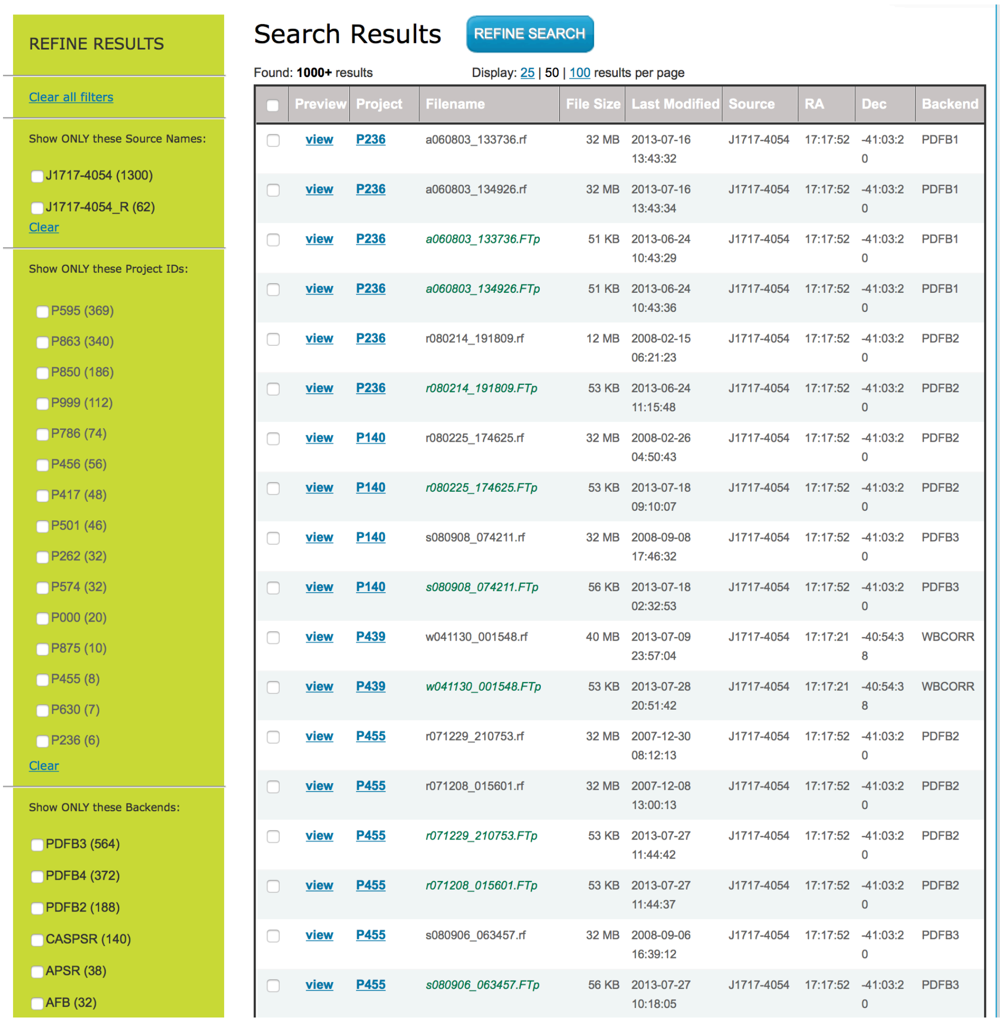
Project code	Description
P871	A follow-up campaign for fast radio bursts
P873	Correlated torque changes and emission variability in pulsars
P874	Detecting and timing a pulsed radio counterpart to the recently discovered high magnetic field X-ray pulsar PSR J1640-4631
P875	A Snapshot Survey of 500 Pulsar Spectral Indices
P877	Measuring the bandwidth of the pulsar emission mechanism
P878	Detection the dispersion and rotation measure caused by the solar wind using millisecond pulsars
P879	Exploring the Progenitors of Fast Radio Bursts
P880 (P880_NOSK)	Investigating the "transitional" binary pulsar XSS J12270-4859
P883	The Parkes All Radio Transients in the skY (PARTY) survey
P885	Understanding the Remarkable Behaviour of Radio Magnetars
P887	Probing the Gamma-ray Pulsar Population with Parkes
P888	Expanding the rotating radio transient parameter space
P892	SUPERBx - The SUrvey for Pulsars & Extragalactic Radio Bursts Extension
P895 (P895_NOSK)	Where are the gravitational waves?
P896 (P896_NOSK)	Wide-band pulsar emission studies using simultaneous observations with Parkes and the MWA
P897 (P897_NOSK)	Planets, Plasma, or Precession? Searching for Pulse Profile Variation in Modulated Pulsars
P898	Timing Observations of a Massive Neutron Star with a "Giant" Companion
P901	Testing the neutron-star model of Fast Radio Bursts
P903	Multifrequency Orbital Studies and Extended Timing of a Black Widow and a Redback
P914	Searching for pulsars from steep-spectrum MWA candidates
P916	Finding the smallest stable structures in the Cold Neutral Medium
P917	Mode changing of PSR J0738-4042
P918	Do FRBs repeat? A case study of FRB 150807
P920	Chasing pulsar emission mechanism via sub-pulse drifting
P922	The polarisation of drifting subpulses of neutron stars
P926	A search of radio pulsation from the candidate Red Back 3FGLJ2039.6?5618
P928	Revisiting the Lorimer burst

Project code	Description
P929	Timing the Highly Eccentric Binary Millisecond Pulsar in NGC 6652
P959	Parkes pulsar observations with undefined project IDs for 2008
P999 (P999_NOSK)	Undefined project codes

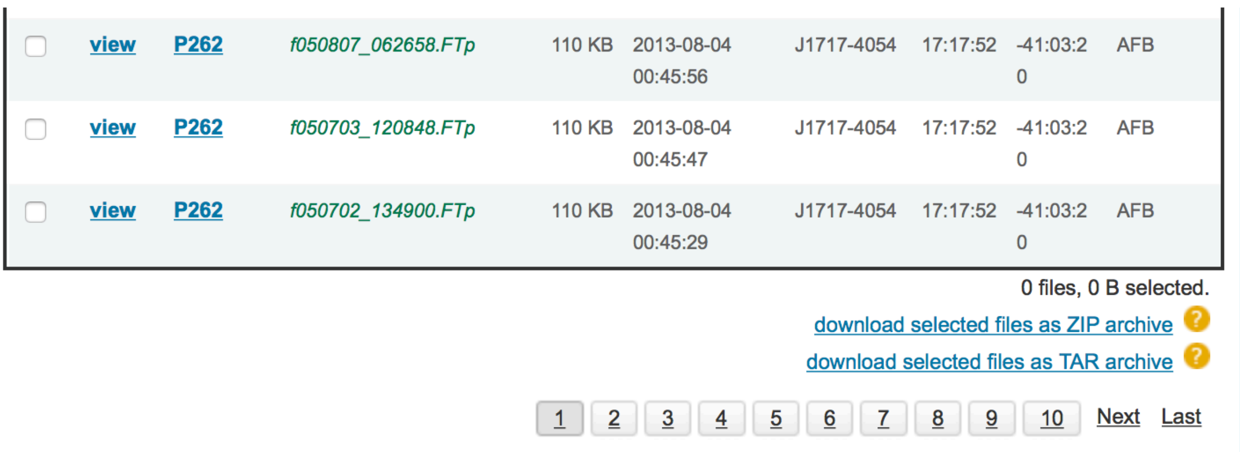
It is uncommon for anyone (apart from the original proposers) to download all the data in a particular collection. More commonly, a user will want all the available data on a particular pulsar, or in a particular direction on the sky. Selecting such data can be carried out using the Pulsar domain search (http://data.csiro.au/psrsearch). This leads to a website like:



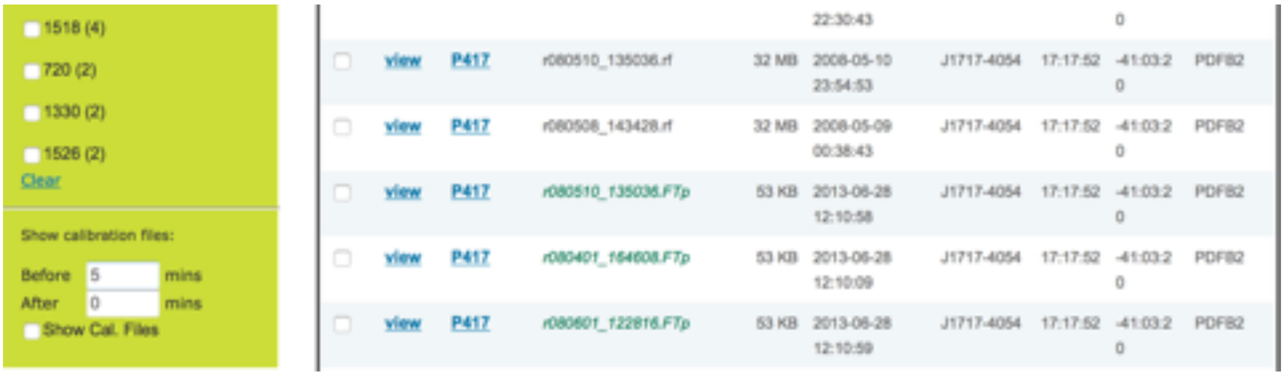
The user can enter source names (such as J0437-4715), sky-positions, project codes and select different observing dates, frequency bands and modes. After clicking on "search" the user obtains a screen like:



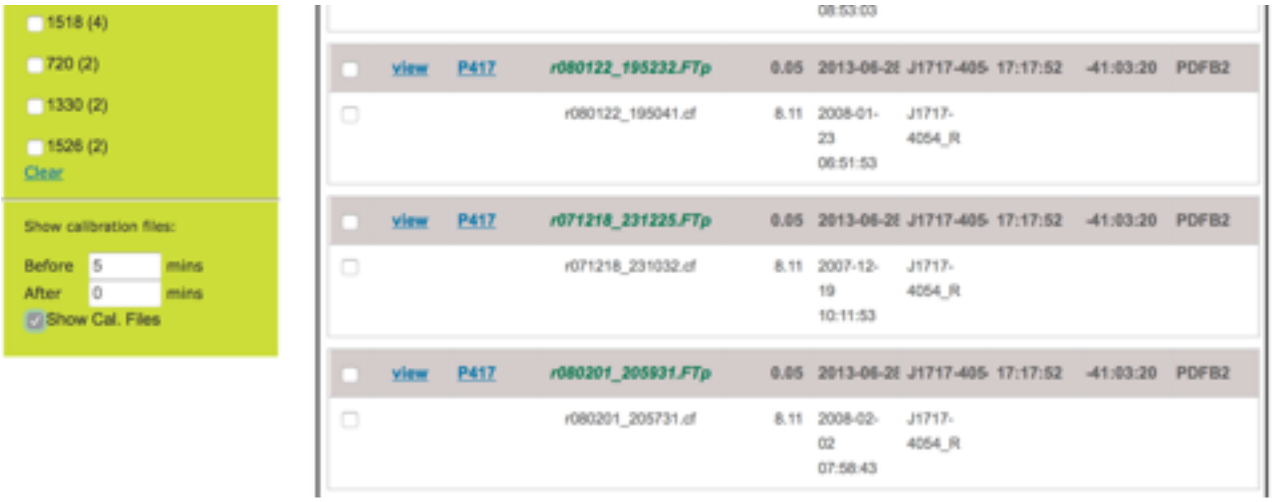
The user can view a thumbnail image of the observation (for fold-mode data) by clicking on "view" or can make different data selections based on the panel on the left of the screen (i.e., the user can select specific project codes, backend instruments, etc.). After selecting one or more observations they can be downloaded as a ZIP or TAR archive through the web browser.



Most fold-mode observations in the Parkes data archive have a corresponding calibration observation[4]. These calibration observations are standard PSRFITS files that were recorded with the same instrumental setup as the pulsar observation. During the calibration observation a noise source is pulsed. This leads to a square signal in the calibration data files with a pulse frequency usually of 11.123 Hz. Such files usually have the file extension .cf. If an entire data collection is downloaded then the calibration files will be included within the download. If the pulsar observation search has been used then the calibration files are not, by default, downloaded. The calibration files can be selected for download by using the menu bar on the left-hand side of the screen. The user can choose to display calibration files before and/or after the pulsar observation:



This results in:



and now the calibration files can be selected and downloaded. These pulsed calibration observations allow a simple polarisation calibration (using the PSRCHIVE program: pac).

[4] For the Parkes Pulsar Timing Array project (P456) all observations have a calibration observation before the pulsar observation. A few pulsars also have a calibration observation at the completion of the pulsar observation. Other observing projects, such as the Fermi Timing project (P574), record a calibration observation and then observe a small number of pulsars before re-recording the calibration source.

A complete calibration, including flux calibration and accounting for cross-coupling in the feed requires a further set of data files. The raw data files required for this calibration can be downloaded from the data archive, but we have also provided a pre-processed set of files for such calibration. The collection is known as the Parkes pulsar calibration data release and is available from http://doi.org/10.4225/08/57FC7776E2758.

Acknowledging the data sets

All the data that can be downloaded from the archive may be processed and published by any researcher. However, we recommend that the authors attempt to provide a citation or acknowledgement to people who carried out the observations. The NASA Astrophysics Data System (ADS) provides citable references for most of the recent data collections in the archive. For instance a bibliographic reference to the You et al. (2014) data collection can be accessed on ADS from http://adsabs.harvard.edu/abs/2014atnf.prop.6397Y. Any publication making use of Parkes observatory data should include an acknowledgement as listed at http://www.atnf.csiro.au/research/publications/Acknowledgements.html.

CSIRO data archiving acknowledgements

In early 2010, the ANDS-CSIRO-ATNF Pulsar Data Management Project was funded through the Australian National Data Service (ANDS) to establish a data archive for pulsar radio astronomy data. The successful project was completed in March 2011. In particular we acknowledge the enormous efforts of James Dempsey, Jessica Chapman and Lawrence Toomey for their work on the archive.

Looking at profiles with the PSRCHIVE software package

The PSRCHIVE software package was initially developed by W. van Straten. It processes and provides visualisation tools for pulsar fold-mode data sets[5]. The software is written in C++ and various packages based on the PSRCHIVE library are now available:

Program	Description
paas	Form standard templates for timing
pac	Polarisation and flux calibration
pam	Process data files and produce new files
pat	Form pulse arrival times for timing
pav	View a pulsar data file
paz	Remove radio-frequency-interference
pazi	Interactively remove interference
pcm	Form polarisation calibration files
pdmp	Improve knowledge of a pulsar's period and DM
pdv	Get information about the profile in text format
fluxcal	Form flux calibration files
psrflux	Use a template to calculate the flux density of a profile
psrplot	Detailed plotting package
psrsh	Shell scripting language for PSRCHIVE
psrwt	Weight the frequency channels in an archive
rmfit	Measure a pulsar's rotation measure

Information about most of these software packages can be obtained using the -h or -H command line options, e.g.,

```
pav -h
```

[5] Some of the routines work with PSRFITS search mode files. This is dangerous as the software was not developed for search mode files and such data files may get deleted or corrupted.

Looking at header information

PSRCHIVE provides two standard packages for viewing metadata information about each file: vap and psredit.

Vap is used to obtain specific, commonly-accessed information from a data file. A complete listing of the possible metadata is available by typing:

vap -H

The -c option allows the user to list common header information. For instance, if we want to know the pulsar name, observing frequency, receiver and bandwidth of a particular data file we can use:

vap -c name,freq,rcvr,bw <filename>

This provides an output similar to:

```
filename   name   freq   rcvr   bw
t090812_011609.rf   J1349-6130   1369.000   MULTI   -256.000
```

If you wish to find information for all files in the local directory then simply use:

vap -c name,freq,rcvr,bw *.rf

to get an output like:

```
filename   name   freq   rcvr   bw
t090812_011609.rf   J1349-6130   1369.000   MULTI   -256.000
t090812_022250.rf   J1349-6130   1369.000   MULTI   -256.000
t091028_005533.rf   J1349-6130   1369.000   MULTI   -256.000
t091207_195349.rf   J1349-6130   1369.000   MULTI   -256.000
t140724_021837.rf   J1349-6130   1369.000   MULTI   -256.000
t140729_022849.rf   J1349-6130   1369.000   MULTI   -256.000
t141014_013500.rf   J1349-6130   1368.875   MULTI   -256.000
t150121_233209.rf   J1349-6130   1368.875   MULTI   -256.000
t160122_013353.rf   J1349-6130   1368.875   MULTI   -256.000
t160330_094507.rf   J1349-6130   1526.250   H-OH   512.000
t160330_094922.rf   J1349-6130   1526.250   H-OH   512.000
t160330_101307.rf   J1349-6130   1526.250   H-OH   512.000
t160330_101855.rf   J1349-6130   1526.250   H-OH   512.000
t160330_111959.rf   J1349-6130   1526.250   H-OH   512.000
```

(The formatting can be improved using the -f option.) The following table provides a list of commonly-used parameters:

Parameter	Description
backend	Name of the backend instrument
beconfig	Backend configuration file
period	Folding period of the pulsar (sec)
bw	Observing bandwidth (MHz)
dm	Dispersion measure ($cm^{-3}pc$)
length	Observation length (sec)
name	Name of the pulsar
rm	Rotation measure (rad/m^2)
tsub	The length of the first subintegration (sec)
freq	Central frequency (MHz)
nchan	Number of frequency channels
nbin	Number of pulse phase bins
npol	Number of polarisations
tbin	Time per pulse phase bin
observer	Observer name
projid	Project code identifier
ra	Right ascension of source (hh:mm:ss.ssss)
dec	Declination of source (dd:mm:ss.ssss)
mjd	MJD of observation
rcvr	Name of receiver

PSRCHIVE processes fold-mode data files. In order to determine the parameters used in the folding then use:

vap -E <filename>

to get an output such as:

```
PSRJ                 J1349-6130
RAJ                  13:49:36.6499999999967
DECJ                -61:30:17.1000000000559
PEPOCH               51138.0000000000000000
F                    3.8556021464199999D+00
F1                  -7.6190000000000001D-14
DM                   2.8460000000000002D+02
RM                  -3.8000000000000000D+02
START                50940.6779999999998836
FINISH               51335.4020000000018626
```

```
CLK                 UNCORR
EPHEM               DE200
EPHVER              2
```

Full meta-data information can be obtained using psredit. This program also allows the user to change meta-data that may have been recorded incorrectly.

Simply running psredit gives all the available meta-data information available for that observation. For instance:

```
psredit t090812_011609.rf
```

gives:

```
file                      Name of the file                                  t090812_011609.rf
nbin                      Number of pulse phase bins                        1024
nchan                     Number of frequency channels                      1024
npol                      Number of polarizations                           4
nsubint                   Number of sub-integrations                        4
type                      Observation type                                  1
site                      Telescope name                                    PARKES
name                      Source name                                       J1349-6130
coord                     Source coordinates                                13:49:36.650-61:30:17.10
…
sub:nbits                 Nr of bits/datum (SEARCH mode 'X' data, else 1)   1
sub:nch_strt              Start channel/sub-band number (0 to NCHAN-1)      -1
sub:nsblk                 Samples/row (SEARCH mode, else 1)                 1
sub:nrows                 Nr of rows in subint table (search mode)          4
sub:zero_off              Zero offset for SEARCH-mode data                  0
sub:signint               1 for signed ints in SEARCH-mode data, else 0     0
```

psredit can also be used to obtain a few parameters of interest:

```
psredit -c freq,bw t090812_011609.rf
```

returns:

```
t090812_011609.rf freq=1369 bw=-256
```

There are usually multiple subintegrations in a file. To output the MJD for each subintegration use:

```
psredit -c int:mjd t090812_011609.rf
```

This returns:

```
t090812_011609.rf
int:mjd=55055.0531816670,55055.0535269093,55055.0538751538,55055.0542233983
```

(i.e., the values are given as a list delimited using commas.) Header parameters can be changed as follows:

```
psredit -c name="Test" -e change t090812_011609.rf
```

will change the header parameter for the source name ("name") and replace it with "Test". A new file will be written which will have the same filename as the original, but the file extension will be changed to "change" (as defined using the -e option).

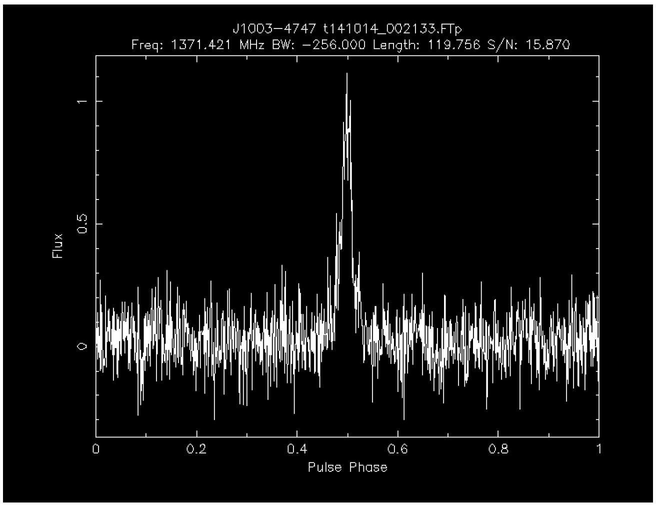
Viewing the profiles

PSRCHIVE provides tools to view pulse profiles, bandpasses, calibration solutions etc. The two main packages are "pav" and "psrplot". pav provides a quick, but simple interface. However, the resulting plots are unlikely to be publishable. psrplot provides a much more powerful set of plotting tools and can be used to make publishable plots.

As always:

pav -h

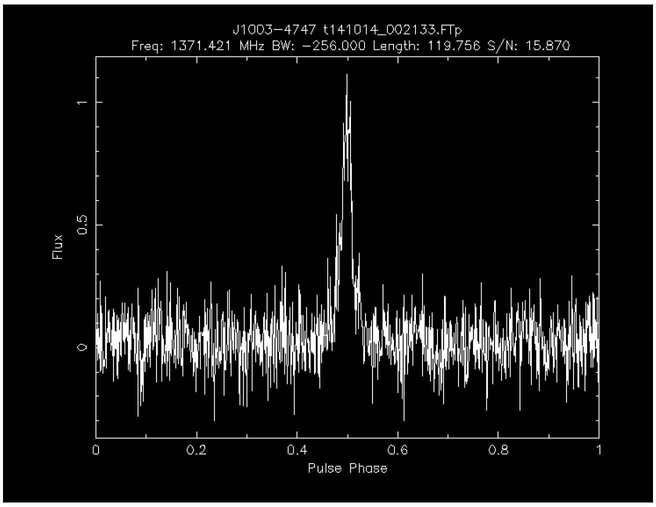
provides help information. A simple plot of the profile can be obtained using the -D option:



pav -CD t14014_002133.FTp

The -C option centres the profile. Note that the command line options can be -C -D as two separate characters or -CD where they are joined.

Note that this example was for a data file that had already been scrunched in time, frequency and polarisation.



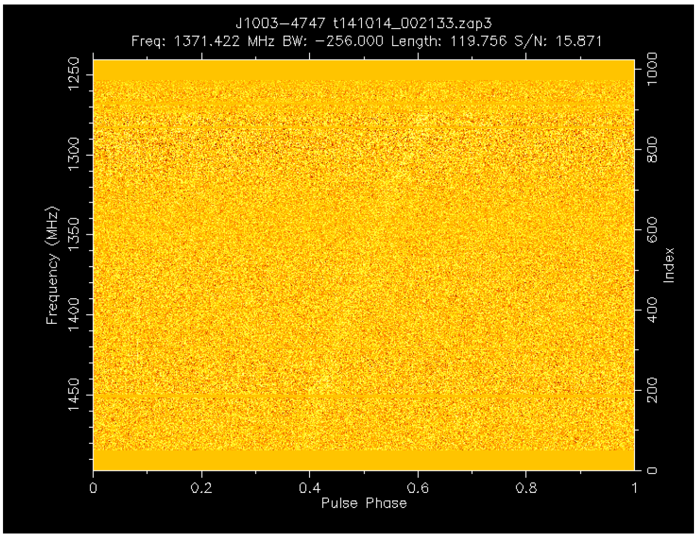
pav can also carry out some pre-processing steps if the profile has not been scrunched. For instance:

pav -CDFTp t14014_002133.rf

will frequency (-F), time (-T) and polarisation (-p) scrunch the profile before plotting and therefore the resulting profile is the same as above.

It is also possible to plot the pulse profile as a colour scale as a function of pulse phase and observing frequency using the -G option:

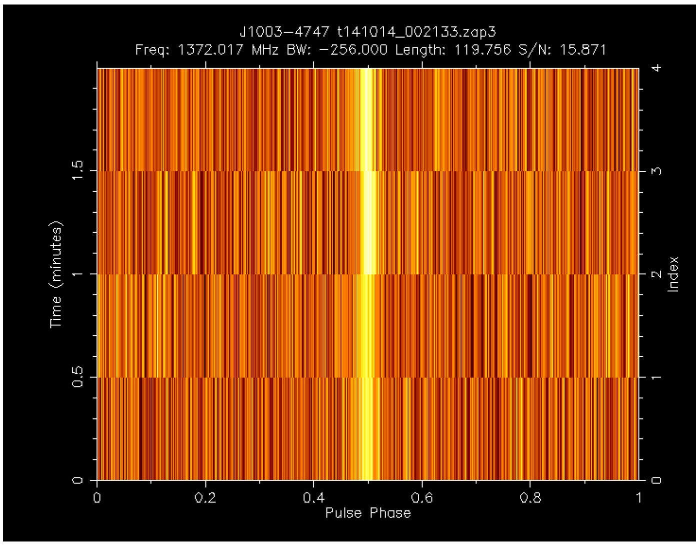
pav -CGTp t14014_002133.rf



In this plot the pulse is seen as the dispersed curve. The -d option can be used to remove the dispersion.

Similarly, pav can plot the pulse profile as a function of subintegration number using the -Y option:

pav -CYFp t14014_002133.rf

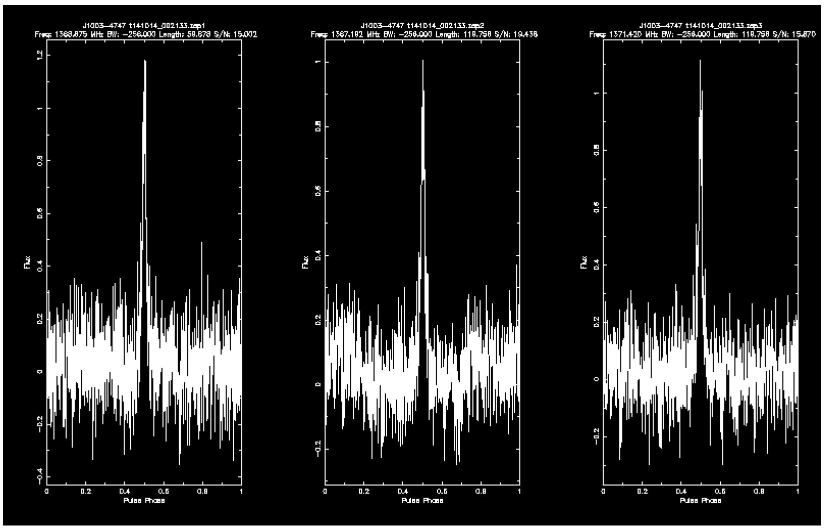


If multiple profiles are listed then, by default, pav will display each one in turn and request that the user presses a key between each one.

It is possible to plot multiple figures simultaneously using the -N <x,y> option. This divides the screen into x columns and y row. For instance:

pav -CDFTp -N 3,1 *.rf

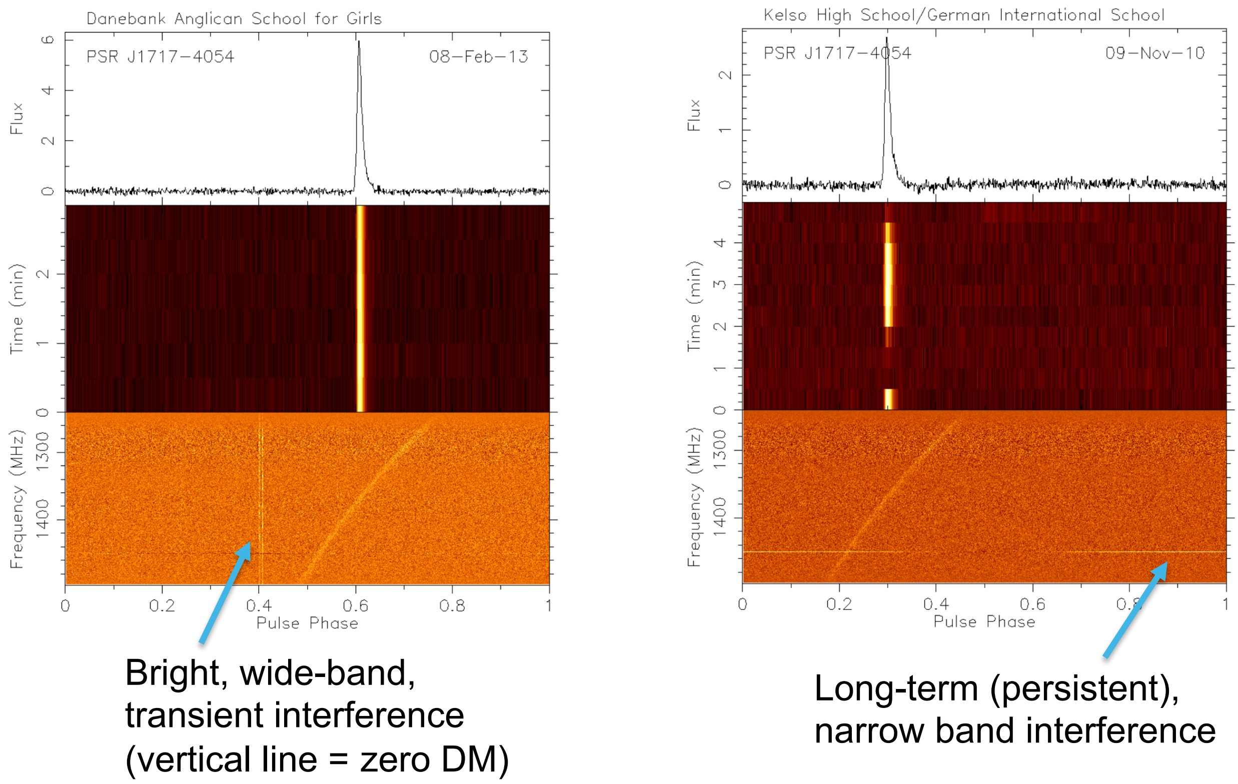
will divide the screen in three columns and one row:



Other common options include plotting the Stokes parameters (-S), displaying the bandpass (-J) and various zoom commands (-z, -k and -y allow the user to zoom in to a pulse phase range, a frequency range or a subintegration range respectively). A particular frequency channel can be identified using the -H option (starting from 0). Similarly a single polarisation or subintegration can be selected using -P and -I respectively.

Removing radio frequency interference (RFI)

Consider the following two observations of PSR J1717-4054.



In both cases the pulse profile is clear and seen as a dispersed signal in the frequency-phase plot. However, in the left-hand figure, two vertical lines are also seen in the frequency-phase plot. In the right-hand figure, a horizontal line is seen. The vertical lines represent wide-band (they occur across all frequency channels), impulsive (they only last a short time) interference. The horizontal line represents narrow-band (only in a few frequency channels), persistent (lasts a long time) interference. Both types of radio-frequency-interference (RFI) are common in pulsar data.

PSRCHIVE provides two packages for removing RFI:

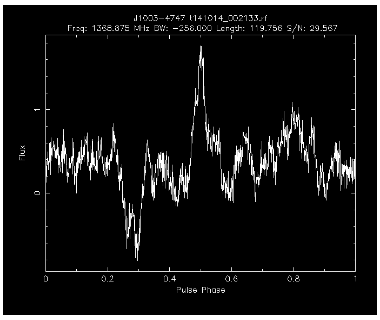
- paz: command-line based package for removing RFI based on known time or frequency ranges or using automatic methods
- pazi: interactive package which provides a graphical-interface to identify and remove RFI.

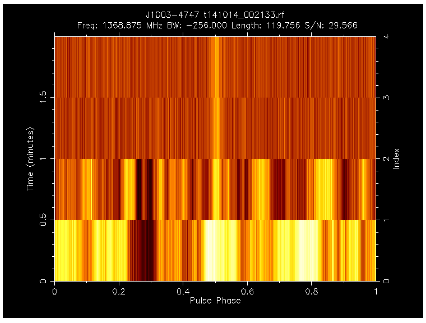
These RFI routines provide the following methods for removing the interference:

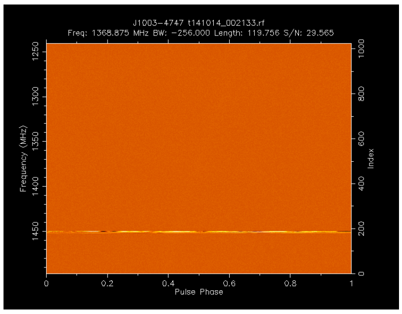
- Remove one or more frequency channels
- Remove one of more subintegrations
- Replace one or more bins in a profile (for a specific frequency channel and subintegration)

To demonstrate these methods let's consider the observation t141014_002133.rf for PSR J1003-4744.

The pulse profile is shown in the left-hand figure below. The profile as a function of pulse phase and subintegration is shown in the middle figure. The right-hand figure shows the pulse profile as a function of phase and frequency:





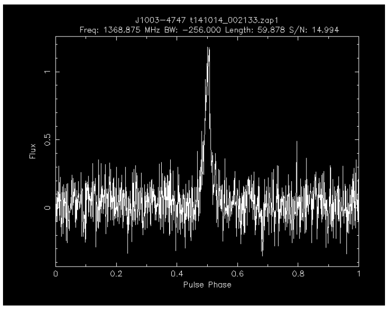


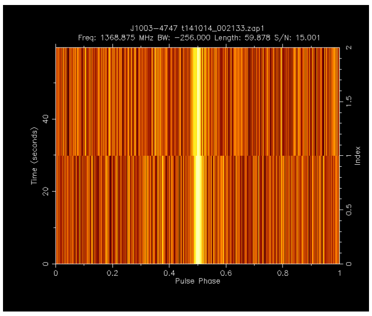
The profile is clearly corrupted by RFI (the baseline is not flat and the pulse is difficult to see). It is clear from the second figure that the RFI is only present during the first two sub-integrations (shown as the bottom two subintegrations in the figure). The right-hand panel shows that the RFI is only present in a small frequency range. We therefore have the options of removing the first two subintegrations (but note that this is equivalent to deleting half of our data as we only have four subintegrations in total) or removing the frequency channels around the RFI (this means that we remove less of the total data).

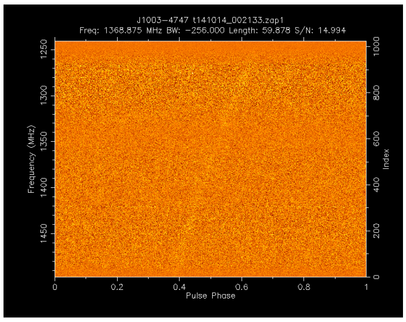
Let's first remove the first two subintegrations using paz:

paz -s "0 1" -e zap1 t141014_002133.rf

This produces a new data file with the file extension .zap1. If we now view that new profile we get:





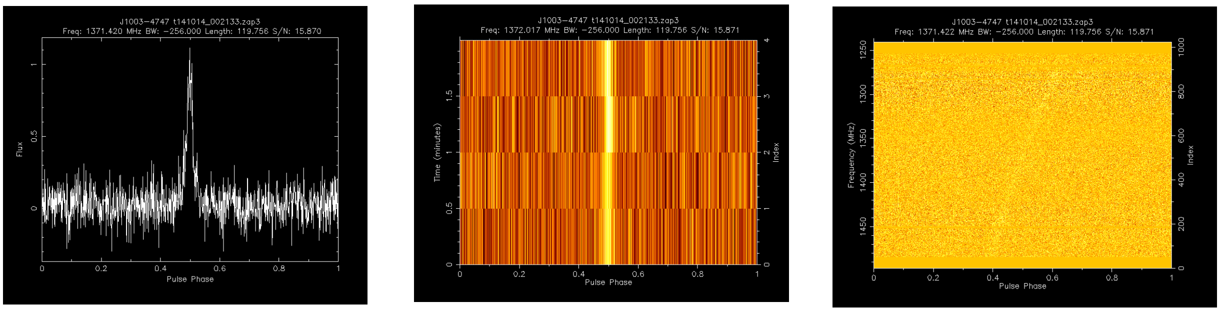


Clearly we have removed the worst of the RFI. The pulse profile (left panel) is now clear, we only have two subintegrations remaining (central panel) and the dispersed pulse is clear without being affected by RFI in the right panel.

We can also remove a small range of frequency channels:

paz -F “1445 1455” -e zap2 t141014_002133.rf

This leads to profiles like:



Again we have removed the RFI, but now we have kept all four subintegrations (central panel) and as we have not removed so much data the S/N of the final profile is larger than in the previous example.

paz also enables automatic RFI removal. A similar plot to above can be obtained using:

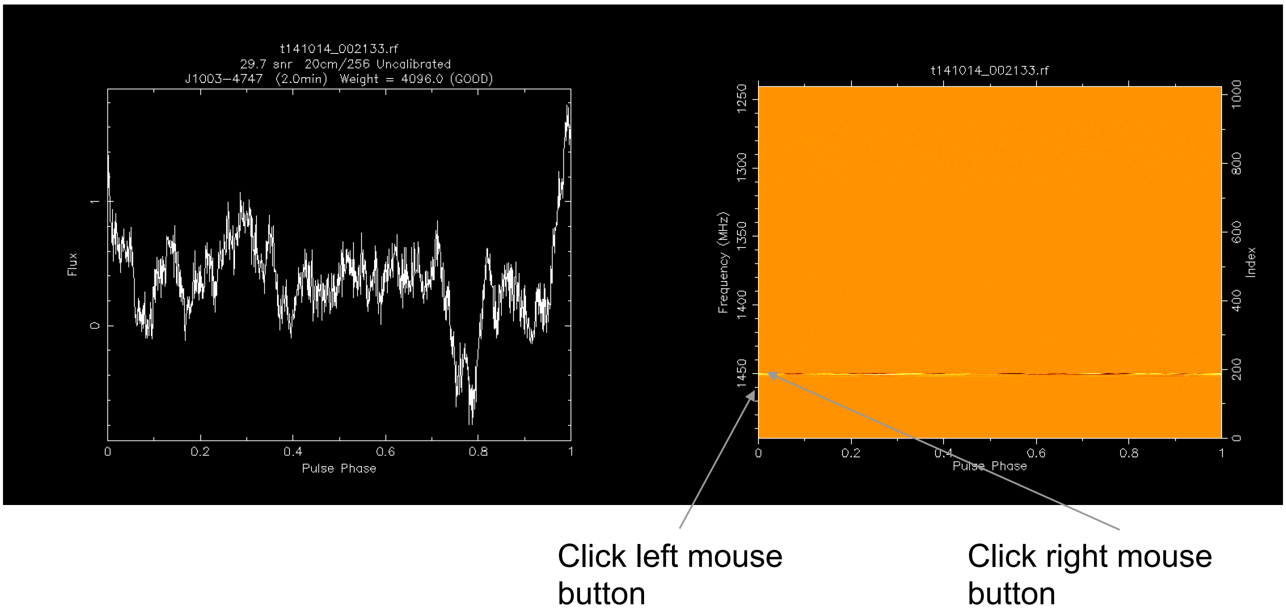
paz -E 5 -r -e zap3 t141014_002133.rf

The command line argument “-E 5” automatically removes 5% of the band-edges. These edges are often affected by the bandpass that can drop off rapidly near the edges. The “-r” option ensures that paz using a median smoothed difference algorithm to identify bad frequency channels.

For a large number of data files it is necessary to use automated RFI-rejection methods. However, for only a small number of files it is preferable to use a manual rejection method. The pazi package provides this functionality. It is run simply by running pazi in interactive mode:

pazi -i t141014_002133.rf

This produces two figures. One is of the profile and the other is a plot that is either the pulse as a function of phase and subintegration number or phase and frequency channel. This figures can be toggled by pressing ’t’ (for subintegration) and ‘f’ (for frequency). Press ‘h’ to get up-to-date help information, but RFI can be selected using mouse clicks (generally a region is defined by first clicking the left-mouse button and then the right-mouse button):



Once the data file has been cleaned then a new file (with file extension .pazi) can be written to disk by pressing 's' to save.

Scrunching with pam

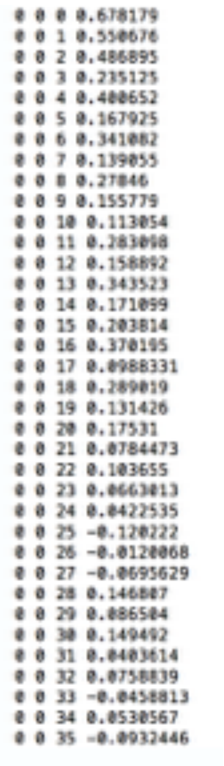
The pam package can be used to scrunch data files in time, frequency and/or polarisation. The command:

pam -FTp -e FTp t141014_002133.zap3

uses the package to scrunch in frequency (-F), time (-T) and polarisation (-p). It produces a new data file with extension FTp (given by -e FTp). Use the -h option to find all the possible command-line arguments for pam. It can be used to install new pulsar timing models (-E), change the number of frequency channels or subintegrations (-f, -t), transform to Stokes parameters (-S) or rotate the profiles (-r).

Even though PSRCHIVE provides various tools for analysing and plotting pulse profiles, sometimes it is necessary simply to extract the profile as a text file for plotting/analysing in e.g., GNUPLOT, MatLAB, IDL etc. This can be carried out using pdv:

pdv -t t141014_002133.FTp



```
0 0 0 0.678179
0 0 1 0.550676
0 0 2 0.486895
0 0 3 0.235125
0 0 4 0.400652
0 0 5 0.167925
0 0 6 0.341082
0 0 7 0.139055
0 0 8 0.27846
0 0 9 0.155779
0 0 10 0.113054
0 0 11 0.283098
0 0 12 0.158892
0 0 13 0.343523
0 0 14 0.171099
0 0 15 0.203814
0 0 16 0.370195
0 0 17 0.0988331
0 0 18 0.289019
0 0 19 0.131426
0 0 20 0.17531
0 0 21 0.0784473
0 0 22 0.103655
0 0 23 0.0663013
0 0 24 0.0422535
0 0 25 -0.120222
0 0 26 -0.0120068
0 0 27 -0.0695629
0 0 28 0.146807
0 0 29 0.086504
0 0 30 0.149492
0 0 31 0.0403614
0 0 32 0.0758839
0 0 33 -0.0458813
0 0 34 0.0530567
0 0 35 -0.0932446
```

This produces an output similar to that shown to the left. The first column is the subintegration number (here always 0 as the data file has been time scrunched). The second column is the frequency channel (again always 0 here as the file has been frequency scrunched). The third column is the bin number and the fourth column is the profile value.

pdv can also be used to extract information such as the pulse flux density (for calibrated profiles), pulse widths and signal-to-noise ratios. This information is provided using the -f option, e.g:

pdv -f t141014_002133.FTp

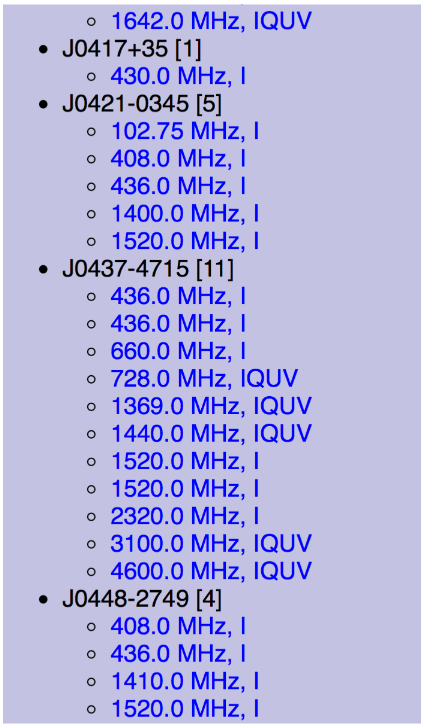
Using python

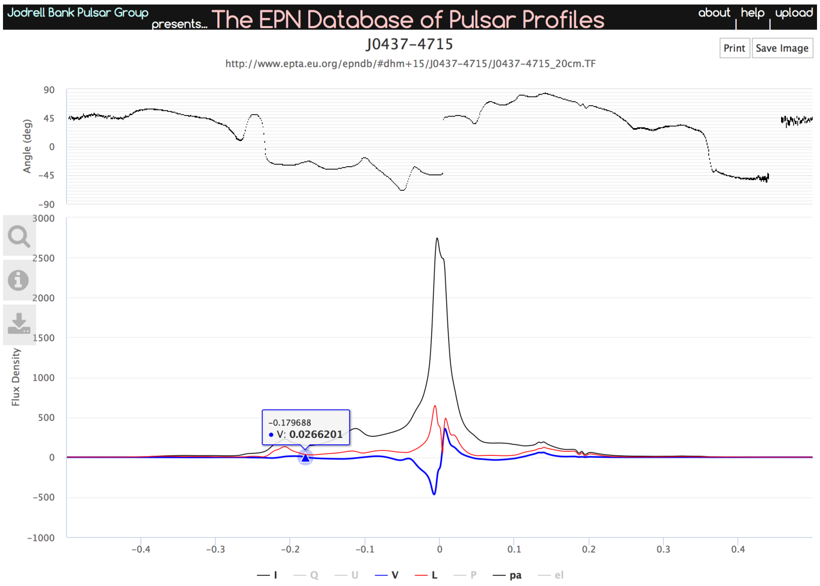
Dynamic spectra

Getting processed pulsar data files

Observation files as recorded from the Parkes telescope are available from the CSIRO data archive. Such files require careful processing before they can be used for publication. Various pulsar groups also make processed data files available.

Pulse profiles can be viewed and downloaded from the EPN archive (http://www.epta.eu.org/epndb/). This is an interactive webpage in which profiles at different observing frequencies can be accessed:





The CSIRO data archive also provides the following collection of processed data:

Description	Access
Processing of the Parkes Pulsar Timing Array data set from Reardon et al. (2015). This includes parameters, arrival times and model files for the PPTA pulsars.	http://doi.org/10.4225/08/561EFD72D0409
Pulse profiles for 24 millisecond pulsars from Dai et al. (2015)	http://doi.org/10.4225/08/54F3990BDF3F1
Data sets produced for the Madison et al. paper that describes directional searches for gravitational wave signals.	http://doi.org/10.4225/08/560A00E2036F6
The Parkes pulsar timing array (PPTA) data release 1 as described by Manchester et al.	http://doi.org/10.4225/08/534CC21379C12
The 23 year PSR B1259-63 dataset	http://doi.org/10.4225/08/5318FF909B6DD

The North American pulsar timing array data sets are available from https://data.nanograv.org.

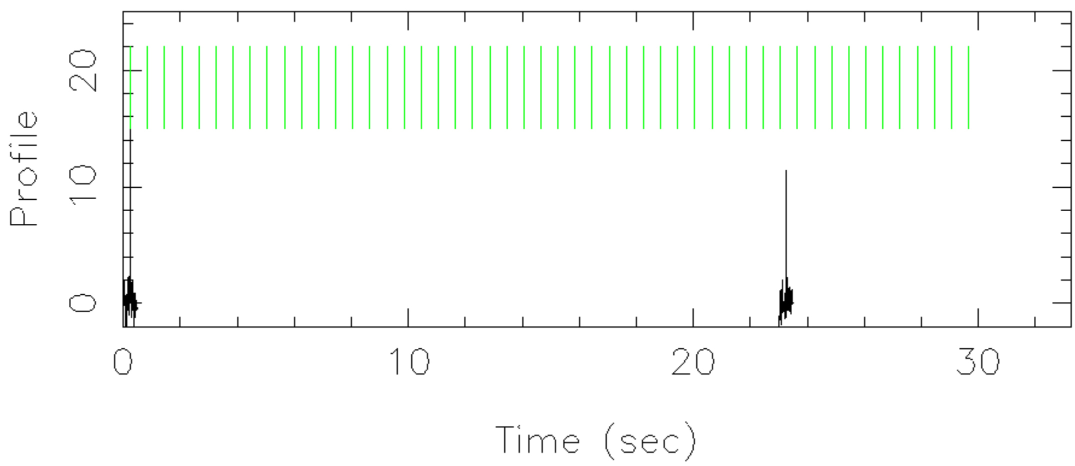
Pulsar timing

Within a year of the discovery of pulsars Counselman & Shapiro (1968) wrote:

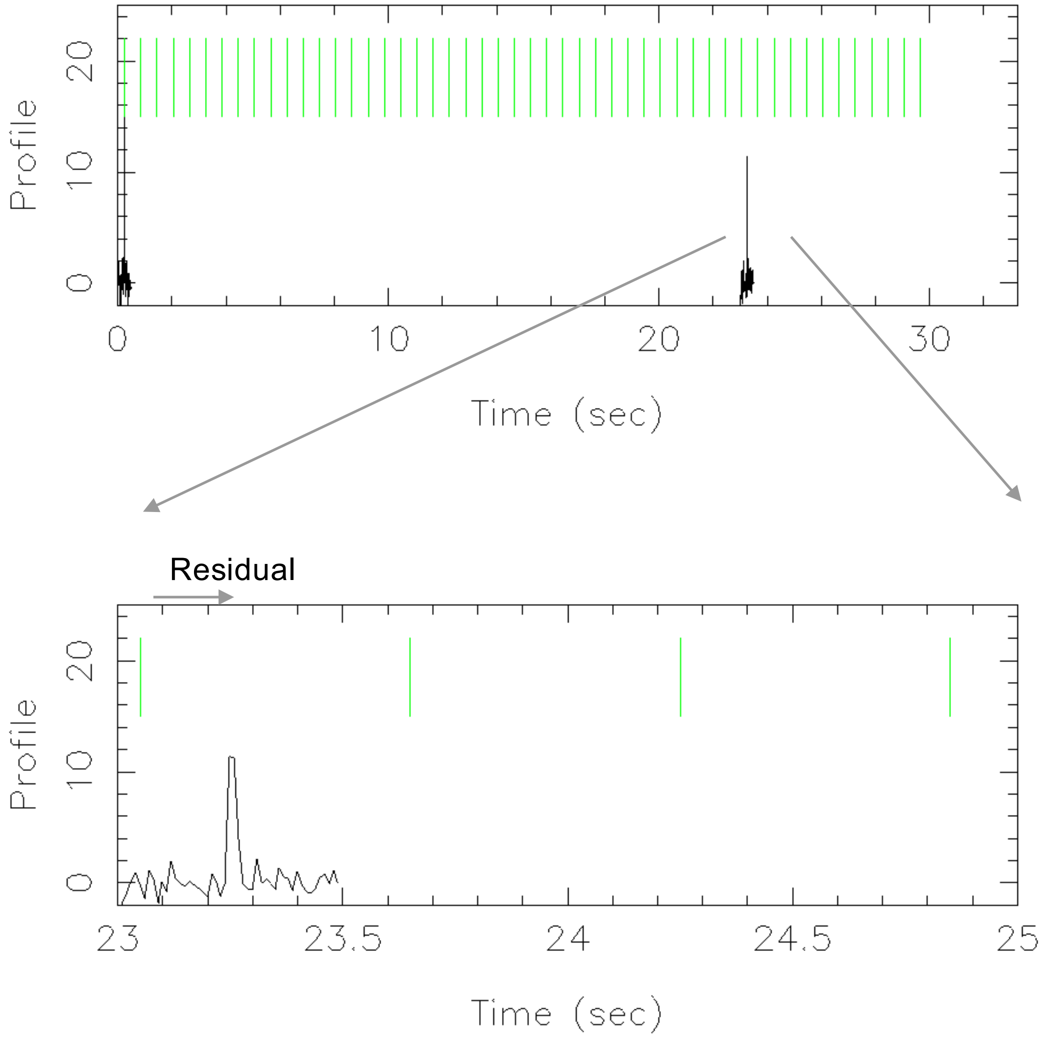
"pulsars … can be used to test general relativity, to study the solar corona, and to determine the earth's orbit and ephemeris time … and the average interstellar electron density."

This is a remarkably complete description of the applications of pulsar timing observations. Along with those applications described above, we now use pulsars also to search for dark matter clumps, in the hunt for ultra-low-frequency gravitational waves, to study extra-solar planets and to probe the interior of neutron stars.

The pulsar timing method is based on folded pulse profiles. In the simple (and not very realistic) example below we assume that we have obtained two folded pulse profiles separated by many pulse periods (for simplicity we assume that our observations are only separated by about 20 seconds, in reality they will be separated by hours, days or even weeks):

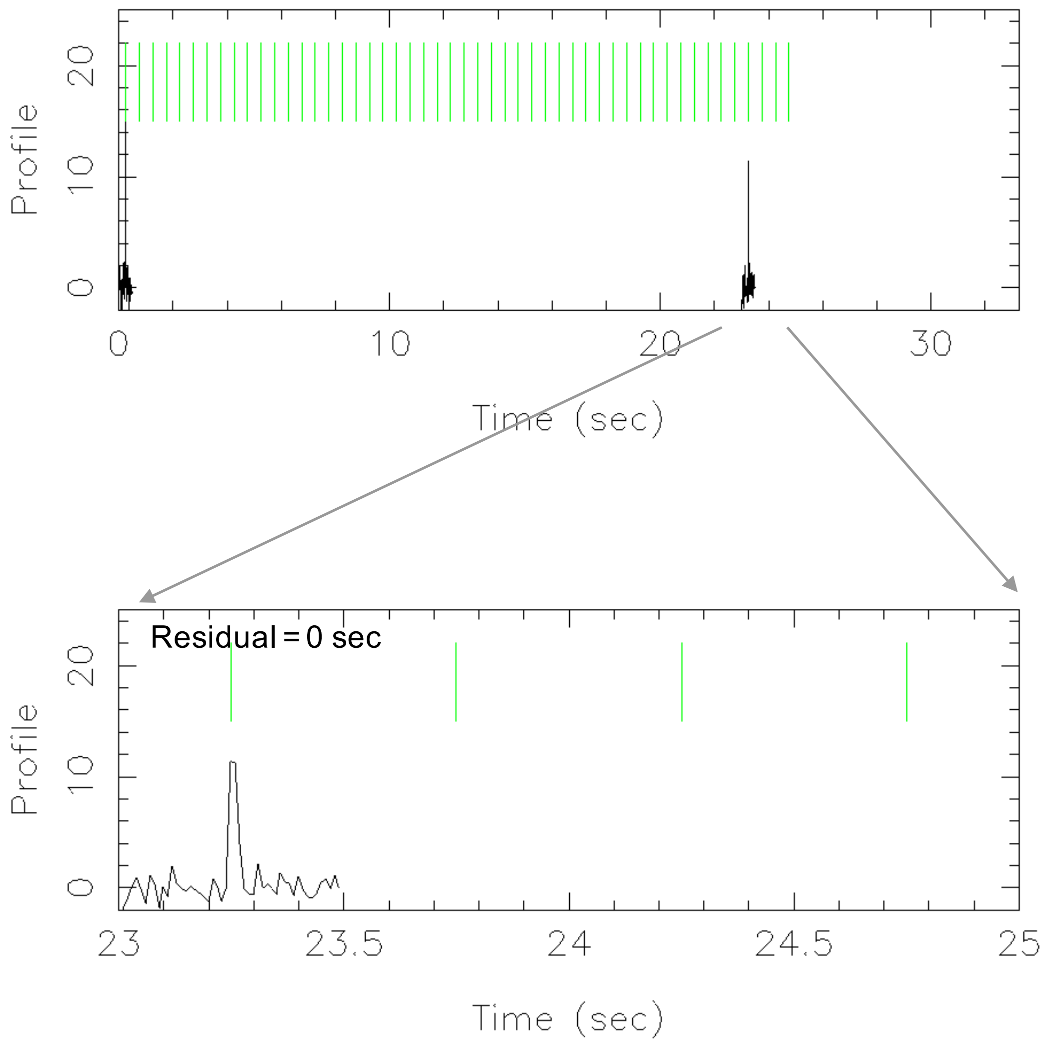


The black lines indicate the folded profiles. The main goal of the pulsar timing method is to determine a model of the pulsar that accurately predicts when the pulses will arrive at our telescope. For the simple example above, we can predict that the pulsar has a pulse period of 0.6 seconds. This allows us to predict when the pulses will be arriving - shown using the green, vertical bars. We now have some observations and a prediction and so let's see how well we have predicted time time of the second pulse. If we zoom-in around the time of the second pulse we get the next figure:



It is clear that our model does not accurately predict the time of the second profile. The time between the prediction and the actual measurement is known as the timing residual. If our prediction was perfect then the residual would be statistically consistent with zero. If the residuals are not zero then we can attempt to improve our model. For instance, in the above example we could try a different period guess. For instance, if we guess that the period is 0.5 s then we can successfully predict the pulse arrival times:

Tempo2 mailing list



General use of tempo2

Input parameter files

Input arrival time files

plk plugin

Selecting data

The splk and plotMany plugins

Clock chains

Fitting for glitch events

Compiling pulsar software

Virtual machines and docker

Angle between two stars

Simulating search mode data files

Introduction

Search-mode data files can be simulated using the simulateSearch package. This package is being developed by George Hobbs (CSIRO), Yuyun Xu (Guizhou normal university) and Di Li (NAOC). The user separately simulates:

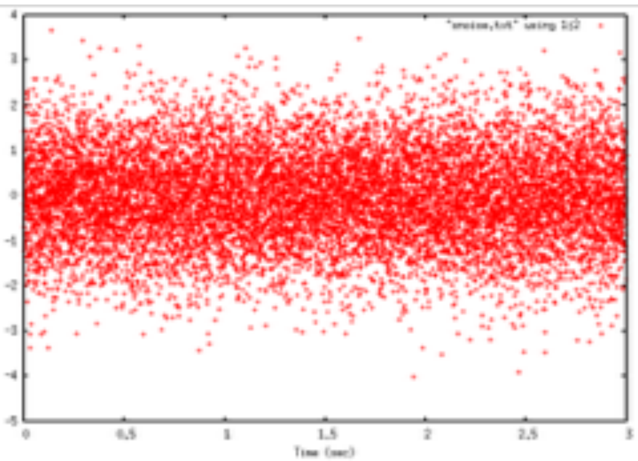
- pulse sequences from various pulsars
- calibration sources
- receiver and sky noise
- a survey strategy (i.e., pointed observations or drift-scans)
- receivers (single pixel, wide-band, multibeam …)
- backends (specified time sampling, number of frequency channels, digitisation, …)

Simulating system noise

Let's start by simply simulating system noise:

```
./simulateSystemNoise -o snoise.dat -to snoise.txt 0 3 0 3
```

The output, binary data file containing the result will be in snoise.dat (defined using the -o command-line argument). However, we want to check that the noise looks sensible. To do this we have also requested some "text output" (-to) to be written to snoise.txt. This file would be enormous if we wrote out the entire data set in text format and so we are only requesting a time range between 0 and 3 seconds (the first two values) and frequency channels between 0 and 3 (the second two values). The first column in the output text file is the time. The other columns are the signal for each of the requested frequency channels. We can simply plot the simulated noise in gnuplot to get a figure like that shown on the left. Note that the noise level has a Gaussian distribution with a mean of 0 and a standard deviation of 1.



It is not necessary to produce the text output, but it is necessary to produce the binary output file (snoise.dat). If you want to inspect the parameters for the binary file then use:

```
./inspectBinaryFile -f snoise.dat
```

This will give an output like:

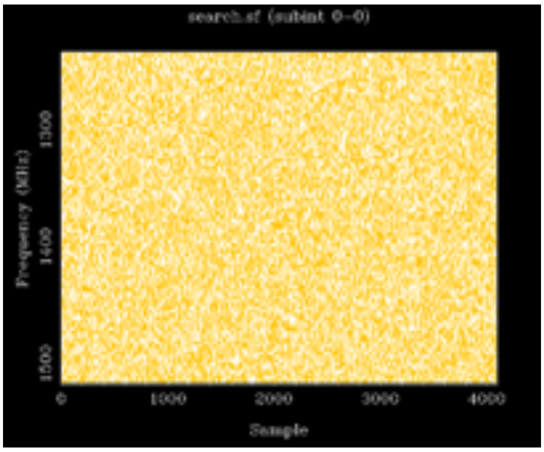
```
Format:        FORMAT 1
Name:          Parkes system
t0 (sec):      0.000000
t1 (sec):      30.000000
tsamp (sec):   0.00025
f1 (MHz):      1518.000000
f2 (MHz):      1230.000000
nchan:         96
RAJ (rad):     0
DECJ (rad):    0
Use angle:     0
Random seed:   -1478753983
```

In this example default parameters were used that are similar to the Parkes multibeam pulsar survey (although the default observation length is only 30 seconds). As this simulation is of radiometer noise the angles provided (RAJ, DECJ) are not relevant (the system noise here is assumed not to depend on the telescope pointing direction).

The binary data file cannot be directly loaded into any of the standard search-mode visualisation or processing packages. We therefore must convert the data file into a standard PSRFITS search mode file. This is carried out using the createSearchFile program:

```
./createSearchFile -o search.sf -f snoise.dat -to out.dat 0 3 0 3
```

This command will load in the binary data file that we have just produced (snoise.dat) and produce a PSRFITS search mode file with filename “search.sf”. The text output (out.dat) should be identical to tnoise.txt that we produced earlier). The resulting PSRFITS search mode file (search.sf) can be viewed using pfits_plot or loaded into the PRESTO package. For instance:




```
pfits_plot -f search.sf -s 0
```

plots the first subintegration as shown in the left-hand plot.

So far nothing in the simulation is particularly realistic and we are only using default input parameters. We can use an input file that contains the simulation parameters as follows:

```
./simulateSystemNoise -o snoise.dat -to snoise.txt 0 3 0 3 -p
params1.dat
```

The file “params1.dat” is a text parameter file containing parameters and values. An example parameter file is given below:

name: Parkes system
f1: 1518
f2: 1230
nchan: 128
t0: 0
t1: 30
raj: 0
decj: 0
useAngle: 0

Inspecting the output file (using inspectBinaryFile) would now indicate that this data file has 128 channels as specified in the parameter file.

The "createSearchFile" software ensures that the same parameters are used when forming the PSRFITS file compared to when the original binary files were created.

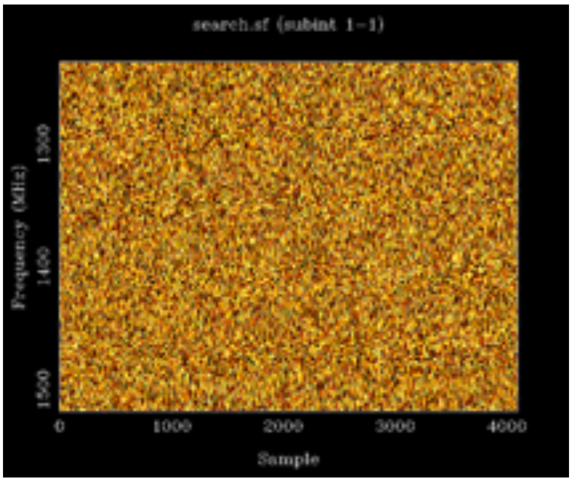
Our simulation does not, so far, include any knowledge of the telescope gain or the system temperature. These parameters can be set in the parameter file using:

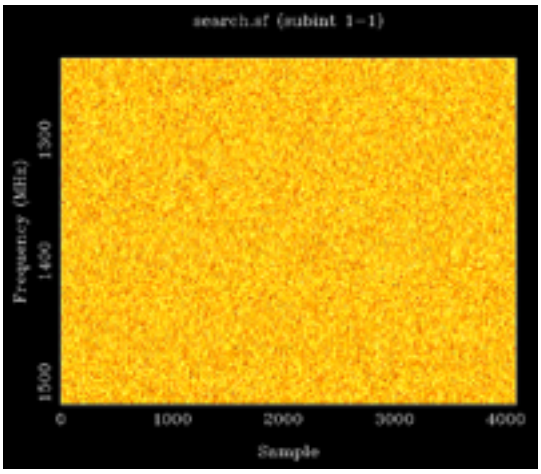
gain: <value in K/Jy>
tsys: <system temperature in K>

With these parameters set "simulateSystemNoise" will produce noise based on the radiometer equation.

By default "simulateSystemNoise" will simulate 1-bit digitised data. This can be set in the parameter file as:

nbits: <1, 2 or 8>





(Note that the levels are not being sensibly set yet). The same data set, but recorded with 2-bit digitisation produces the pfits_plot output shown on the left above and with 8-bit digitisation on the right. Of course the output data file size scales with the number of bits.

Adding in a simple pulsar

We now wish to add in a simulated pulsar into our data set. The software provides two methods for simulating pulsars: a quick and easy way and a more complex (and slower) method.

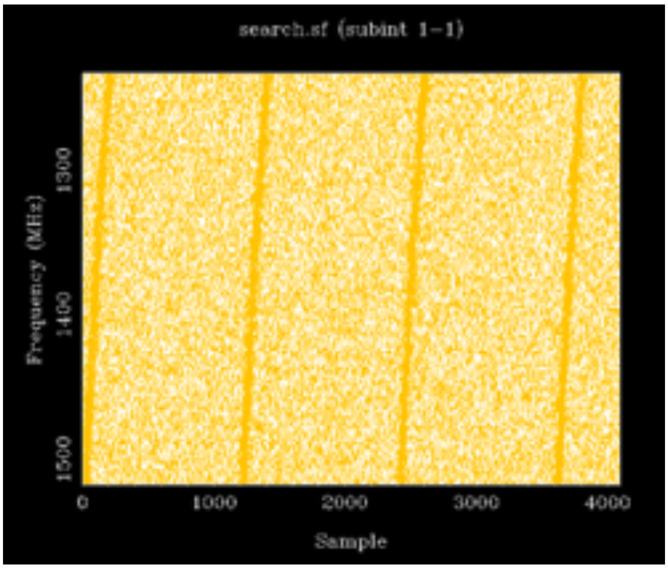
To create a default pulsar use:

```
./simulateSimplePsr -p params1.dat -o pulsar.dat
```

We can then add the pulsar signal to the radiometer noise signal when forming the output PSRFITS file:

```
./createSearchFile -o search.sf -f snoise.dat -f pulsar.dat
```

(Note that we simply add more binary files using multiple '-f' options)



This gives the figure on the left. Single pulses from default pulsar are clearly visible. This default pulsar has a period of 0.3 seconds, a dispersion measure of $50\mathrm{cm}^{-3}\mathrm{pc}$ and a peak amplitude of 3.

The periodicity is assumed to be exact - i.e., no slow down, Earth's motion or orbital motion is modelled.

The parameter file can include parameters to describe the pulsar:

p0: <pulse period in seconds>
dm: <dispersion measure>
width: <pulse width in seconds>

It is often useful to have one parameter file for the system setup and a second one for a given pulsar. Different parameter files can be placed on the command line (they supercede each other - i.e., any parameter in a later file will overwrite the same parameter in an earlier file):

```
./simulateSimplePsr -p params_setup.dat -p params_pulsar1.dat -o
pulsar.dat
```

We now have params_pulsar1.dat as:

name: Pulsar 1
p0: 0.1
dm: 20
raj: 4.510914803
decj: 0.13602659
width: 0.03
useAngle: 0

and params_setup.dat as:

name: Parkes system
f1: 1518
f2: 1230
nchan: 128
t0: 0
t1: 30
raj: 0
decj: 0
useAngle: 0
nbits: 1

Note that the simple pulsar generation code (simulateSimplePsr) only works if the dispersion across the band is less than 1 pulse period. This assumption is made to make the code run quickly.

The pulsar's flux density can be set in the parameter file using the:

flux: <flux density in Jy>

option. If this parameter is set then the area under the simulated pulse will equal this value. Of course, this should only be set if the system noise is being correctly determined using the radiometer equation.

Using sky positions

Clearly the pulsar will only be detected if a receiver beam is actually pointing at the pulsar. The telescope may be observing in drift mode or the pulsar may be being detected in a side-lobe of the main beam. To make use of beam shapes, drifts etc. it is necessary to tell the software to make use of sky positions. To do this use:

useAngle: 1
raj: <pulsar right ascension in radians>
decj: <pulsar declination in radians>

in the parameter file used when simulating the pulsar, but ensure that

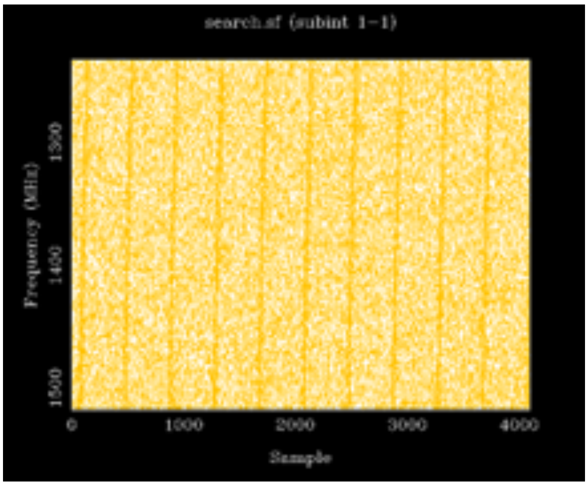
useAngle: 0

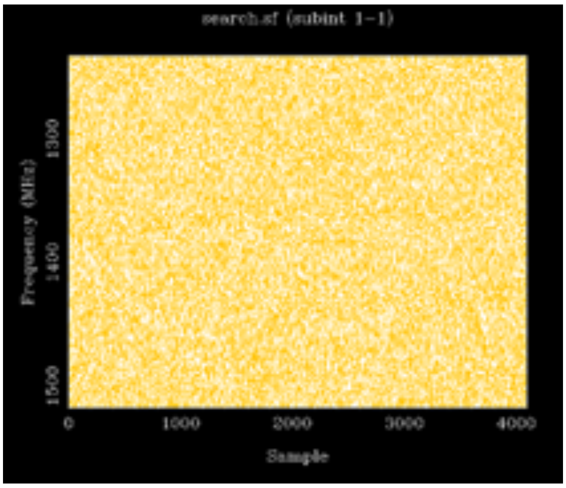
is used when simulating e.g., radiometer noise (as the radiometer noise does not depend on the telescope position).

It is now necessary to define the telescope pointing position and whether it changes as a function of time. The parameter file used when running createSearchFile can have:

beamRA0: <right ascension in radians>
beamDEC0: <declination in radians>

These correspond to the beam pointing position at the start of the observation. By default the telescope is assumed to track the source and so the beam position does not change. As expected when the beam position is the same as the pulsar position (left panel below) then the individual bright pulses are seen. When the beam position is offset by a large amount then no pulses are detected (right panel):





The telescope beam shape is modelled as a sinc-squared function. This provides a reasonable model that includes side-lobes etc. The beam shape is defined by the observing frequency and the telescope diameter. The default telescope diameter used in the simulation is 300m, but this can be changed using:

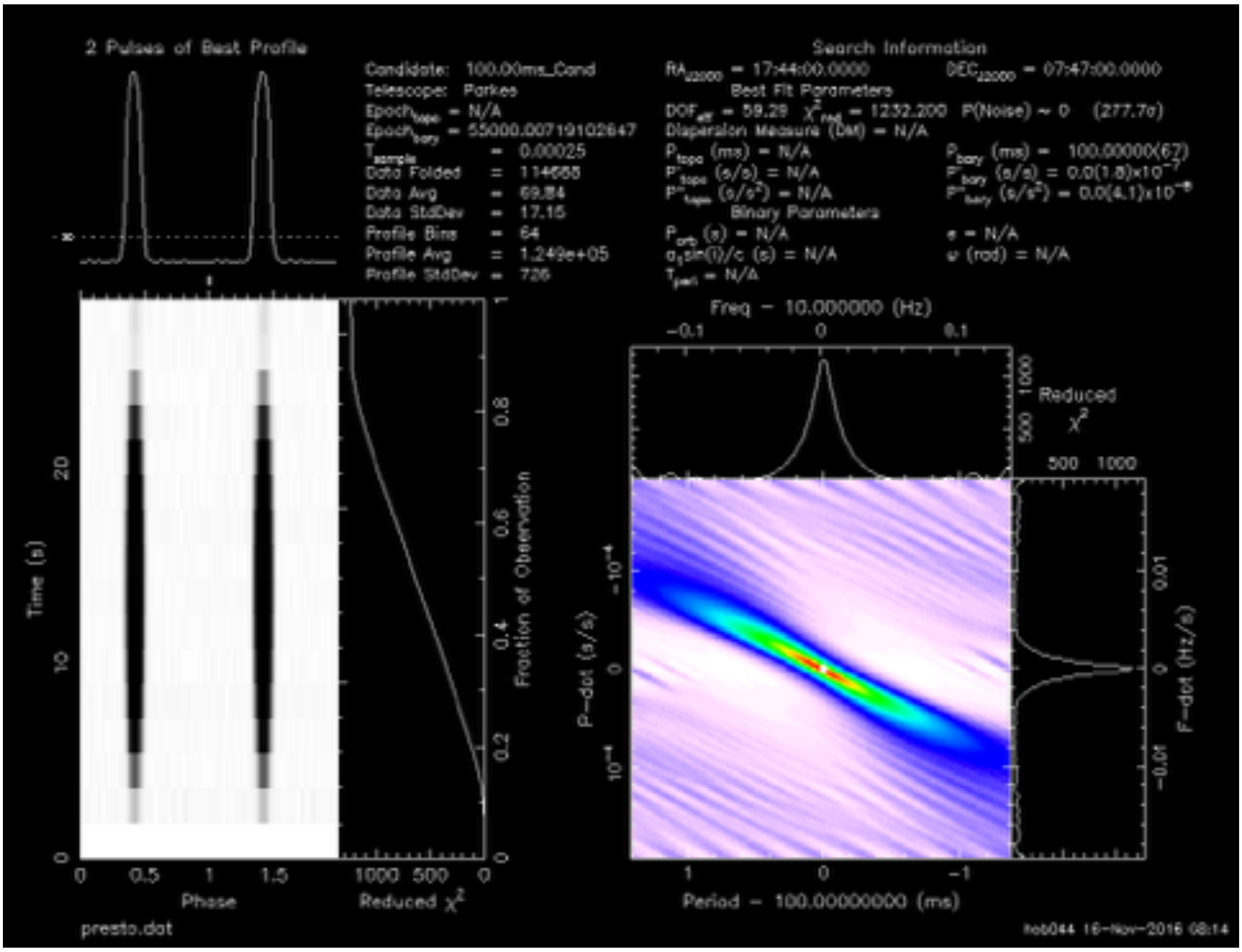
diameter: <telescope diameter in m>

in the parameter file. It is also possible simulate a drift survey in which the sky drifts overhead the telescope. In this mode:

surveyType: 2

(the default, 1, is for pointed observations) the beamRA will be updated as a function of time. We can process the data using PRESTO:

$PRESTO/bin/prepdata -dm 20 -o presto -psrfits search.sf
$PRESTO/bin/prepfold -p 0.1 -npart 16 presto.dat



the resulting figure clearly shows the pulse and the time-phase plot demonstrates the drift. The pulsar moves into the beam, remains for about 15 seconds and then drifts out of the beam.

Realistic pulse sequences

The simulations shown above assume that the pulse sequence is defined by a non-changing pulse period. In reality the observed pulse period will be changing because of the Earth's motion, the pulsar slow-down and the pulsar's orbital (for a binary system). The pulse phase and pulse period for a given time and observing frequency can be determined using tempo2 predictors based on real (or simulated) parameters. For instance we can use the parameters for PSR J1744-1134 from PSRCAT:

psrcat -E J1744-1134 > 1744.par

We then need to make a tempo2 predictor that provides a simple model for the phase of the pulsar as a function of time. The command is:

```
tempo2 -pred  "sitename mjd1 mjd2 freq1 freq2 ntimecoeff nfreqcoeff
seg_length (s)" -f <parameter file>
```

At the moment it is the responsibility of the user to ensure that the parameters here (telescope, MJD range, frequencies etc.) are consistent with the parameters being simulated. For our example we use:

```
tempo2 -pred  "PKS 55999.8 56000.1 1200 1600 16 2 1000" -f 1744.par
```

The tempo2 output is:

```
RMS error = 1.49e-09 s MAV= 4.96e-09 s
```

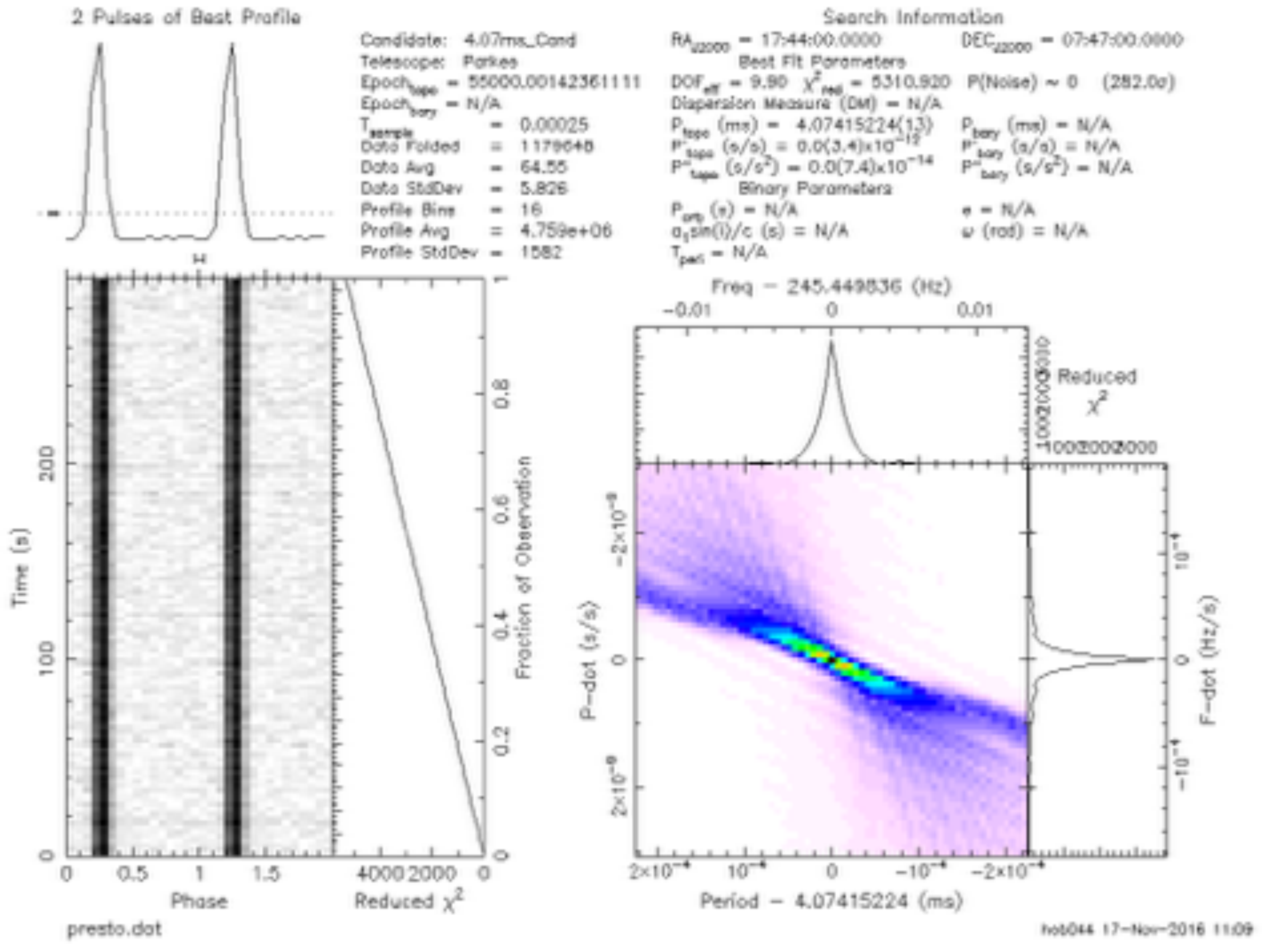
which indicates that the solution is sufficient at the few nanosecond level. This is fine for our purposes. For fast binary pulsars it will be necessary to increase the number of time coefficients or decrease the segment length.

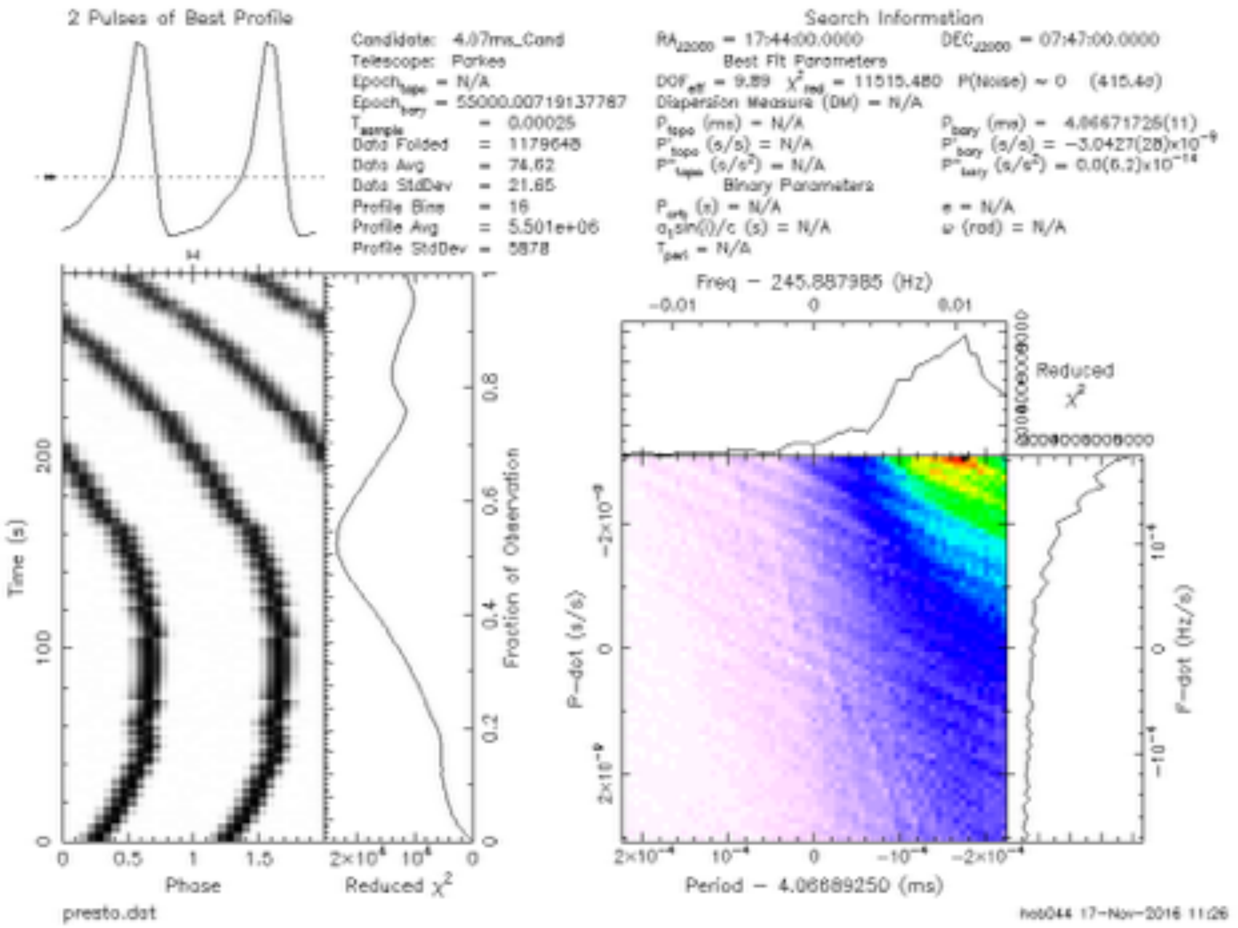
Tempo2 produces a file on disk called t2pred.dat (note that it is currently not possible to change this filename and so this is a problem for running the simulation code simultaneously on multiple nodes - we should update tempo2 to ensure that the file name can be set on the command line).

We then can use this t2pred.dat file to simulate a realistic pulse train:

```
./simulateComplexPsr -o complexPsr.dat -p params_setup.dat -p
params_pulsar1.dat
./createSearchFile -o search.sf -f snoise.dat -f complexPsr.dat -p
params_setup.dat
$PRESTO/bin/prepdata -dm 3.139 -o presto -psrfits search.sf -nobary
$PRESTO/bin/prepfold -p 0.00407415223690579 -npart 128 presto.dat
```

(I'm confused about the -nobary option in prepdata. That shouldn't be needed??)





In order to fake a binary pulsar we add the following into the parameter file:

```
BINARY DD
PB              0.1
ECC             0.0000749402
A1              10.34242187
T0              53761.0328
OM              176.1966
```

We have to run tempo2 again in the prediction mode to produce t2par.dat. Then simulateComplexPsr. As seen in the PRESTO figure below, our simulated pulsar is now in a very fast binary system.

Wide-bandwidth receivers

Let us try to make a "realistic" prediction for single pulses from PSR J1713+0747 using the FAST ultra-wide-band receiver during a drift scan observation. To do this we first obtain the timing model for this pulsar:

psrcat -E J1713+0747 > 1713.par

we then produce a tempo2 predictor

tempo2 -pred "PKS 55999.8 56000.1 200 1600 16 2 3600" -f 1713.par
(NOTE: Should update PKS to FAST)

We then setup the parameter file for FAST uncooled UWB system (params_setup_fast.dat):

name: FAST uncooled UWB
f1: 1600
f2: 270
nchan: 4096
tsamp: 64e-6
t0: 0
t1: 30
raj: 0
decj: 0
useAngle: 0
gain: 16
tsys: 70
nbits: 1
beamRA0: 4.509914803
beamDEC0: 0.1534798910

diameter: 300
surveyType: 2

Now we need to simulate the radiometer noise:

```
./simulateSystemNoise -o snoise_fast.dat -to snoise_fast.txt 0 3 0 3 -p
params_setup_fast.dat
```

This takes 3.5 minutes on a laptop. It takes a long time because 4096 channels are being simulated with samples every 64us. The resulting data file is 7G.

Now we need to add in our pulse sequence from PSR J1713+0747:

name: J1713+0747
raj: 4.510914803
decj: 0.1534798910
width: 0.03
useAngle: 1
flux: 10.2e-3

The flux density and position (in radians) is obtained from the PSRCAT catalogue. This takes 6 minutes to run on a laptop. Now we can produce a search mode data file accounting for the drift:

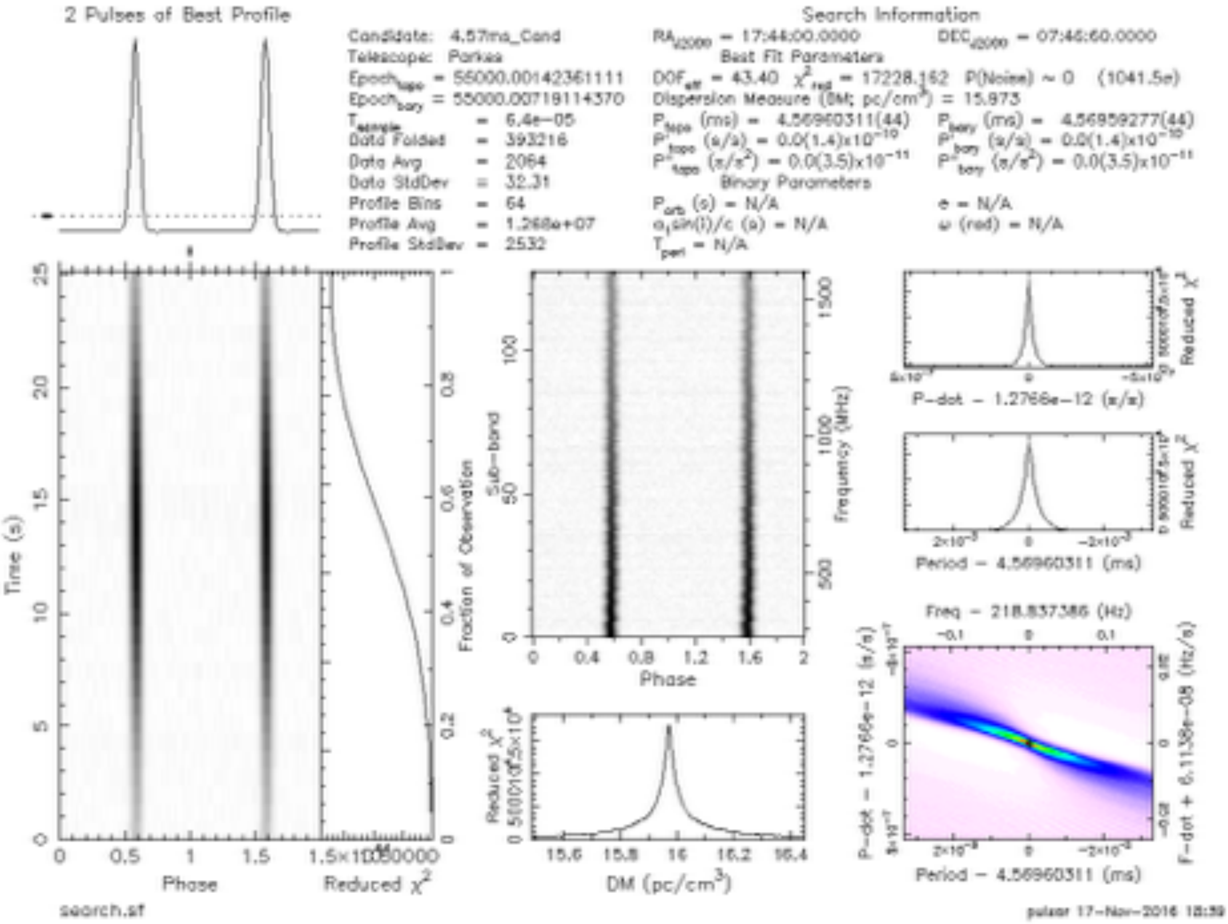
```
./simulateComplexPsr -o J1713+0747.dat -p params_setup_fast.dat -p params_1713.dat
./createSearchFile -o search.sf -f snoise-fast.dat -f J1713+0747.dat -p params_setup_fast.dat
```

using default parameters this takes 1 hour to run on a laptop.

```
 prepfold -p 0.00456959276781935 -npart 32  -dm 15.97 search.sf
```

The resulting figure clearly shows the high S/N achievable and the drift:

The simulation shown above does account for the differing beam sizes as a function of observing frequency. However, there are many other wide-band phenomena that should be simulated including:



- scattering
- dispersion smearing
- varying system temperatures across the band
- profile evolution (both shape and flux density)

In order to add scattering effects then include:

scatter: 1

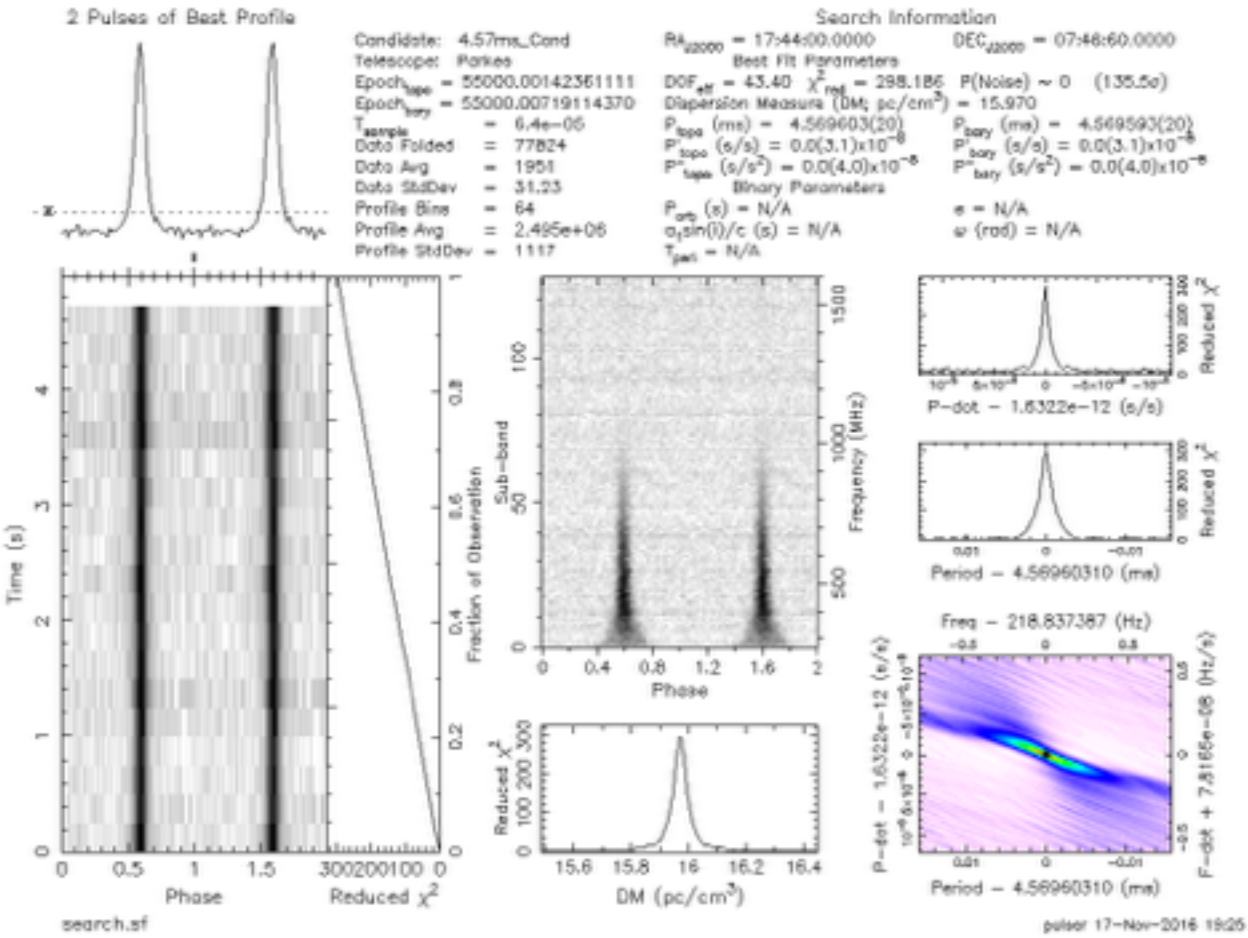
into the parameter file when simulating the pulsar. Note that the DM must be set in the parameter file.

In order to add dispersion smearing use:

dmSmear: 1

in the parameter file.

The figure below shows the result for the first 5 seconds of data:

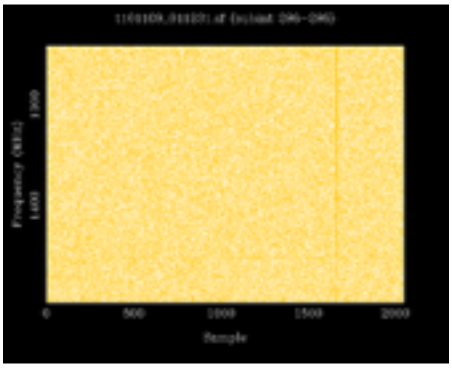
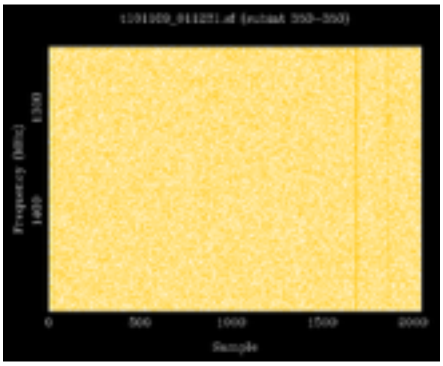


Burst events

Radio frequency interference

RFI can occur from stationary objects on the Earth (i.e., mobile transmissions, electric fences, etc.). The detected signal strength will therefore depend upon the telescope azimuth and elevation (i.e., where the telescope is pointing). RFI can also occur from moving objects such as aircraft and satellites. Such RFI will therefore depend on both the telescope pointing direction and the position of the moving object. Some RFI will come from the telescope itself (such as RFI caused from the mains electricity) and therefore will not depend on the telescope pointing direction.

Real transient RFI events are shown in the figure.

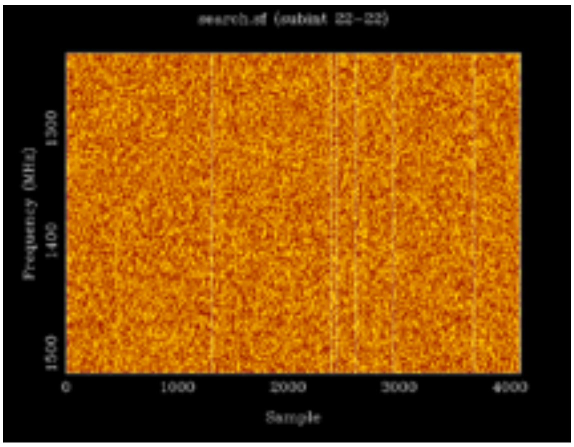
 

For transient RFI we can:

- choose transient burst events from a text file containing the pulse amplitude and width. The pulse shape will be a rectangular pulse or a fast rise and exponential decay. It is assumed to cover the entire frequency band. The number of events occurring during the simulation can be defined. The exact event time will be randomly chosen.
- make a sequence of burst events that occur over a short time interval. The user defines the properties of a given burst event and then provides the number of repeats and the number of events in the data set.

To demonstrate this we can set up a parameter file (params_rfi.dat) containing:

name: RFI impulsive
rfi_file: rfiEvents.dat
nrfi: 1000

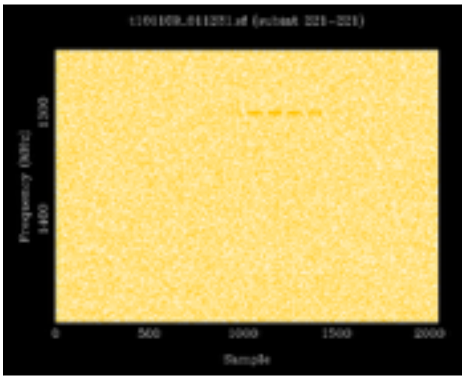


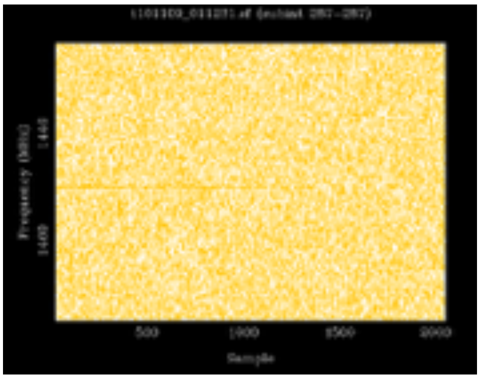
This leads to an output data file (for 2-bit data) like the picture on the left. The RFI is seen as the vertical stripes.

For persistent, narrow band RFI the user can define:

- the frequency range affected
- the time the RFI starts and ends (t0 and t1)
- the typical duration of the RFI when it is on (deltaT1)
- the variation in the on time scale (sigma_T1)
- the typical off duration of the RFI (delta T2)
- the variation in the off time scale (sigma_T2)
- amplitude variation (satellites moving in and out of the beam or track away from source)

Real RFI, narrow-band signals are shown below:





Simulating RFI signals such as these would occur by adding into the parameter file:

rfi_narrowBand: 1300 1310 10

which gives the frequency range (1300 to 1310 MHz) and a peak amplitude (Jy). Without any other options this RFI would persist throughout the observation. It is also possible to give a start time (seconds from the start) and a duration (sec):

rfi_narrowBand: 1300 1310 10 0.3 0.1

If the user wants a random start time then use:

rfi_narrowBand: 1300 1310 10 r 10

It is possible to simulate RFI that drifts into and out of the beam (e.g. from satellites):

rfi_narrowBand: 1300 1310 10 5 10 0.1

(the 0.1 here gives the time scale for the drift. Use 0 for no drift.)

Finally the RFI can repeat on a specified time scale (in seconds):

rfi_narrowBand: 1300 1310 10 5 10 0.1 20

Multiple RFI signals can be specified on different lines in the parameter file:

rfi_narrowBand: 1300 1310 10 5 10 0.1 20
rfi_narrowBand: 1330 1340 10 10 10 0.1 20
…

Satellite interference (use model of satellite sky position - what about geostationary satellites?)

For periodic, locally generated RFI the user can define:

- frequency (e.g., 50 or 60Hz mains electricity), deltaF giving the variability of the pulse frequency, the amplitude and whether the signal is a sinusoid or it has been rectified. (SHOULD SEE IF THERE IS A TEXT FILE SHOWING THE VARIABILITY - see https://en.wikipedia.org/wiki/Electric_power_quality)

Such signals are defined by:

rfi_frequency: 49.99 50.11 1 1

These signals are not perfectly stable in frequency. Therefore the user can define an upper and low range of the actual frequency (49.99 and 50.11 Hz in the above example and a time scale in which the frequency changes - 1 second). The final value (1) states whether the signal is rectified or not.

Counters … 1024, 1 … Square pulse trains

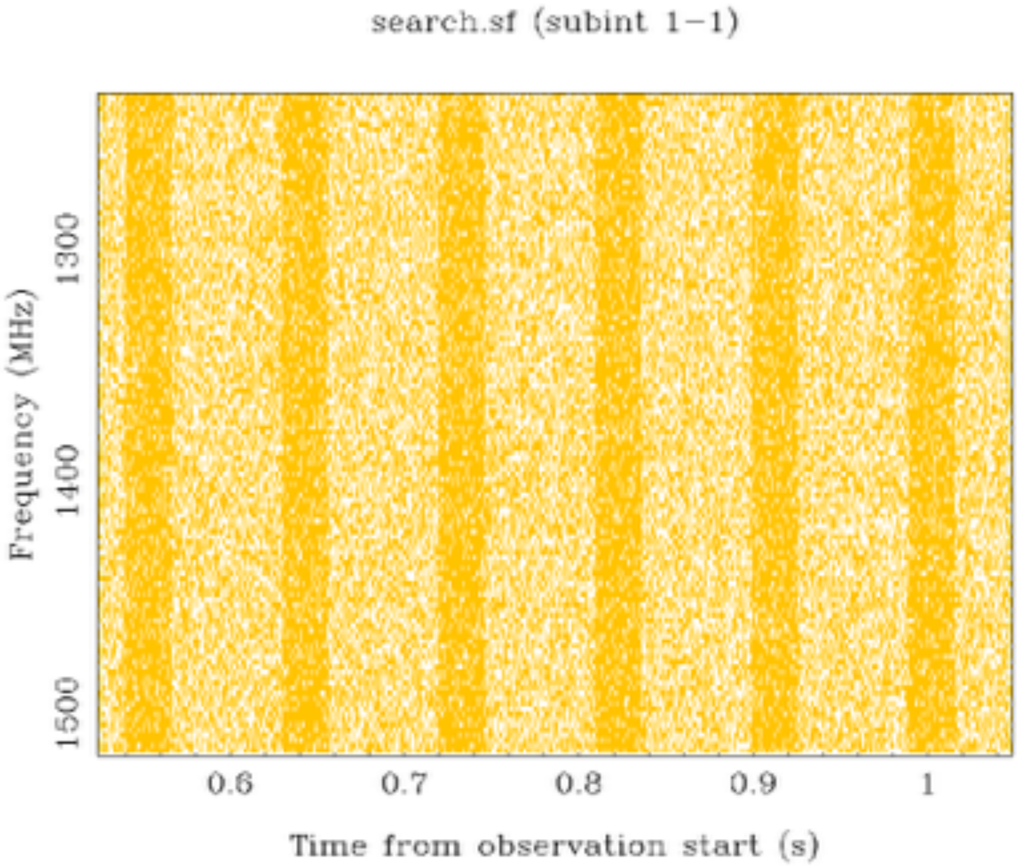
Calibration sources

Most telescopes and receiver systems provide the means to calibrate the data sets. A common calibration signal is a pulsed, broadband signal. For pulsar calibration the signal is pulsed at periods similar to a pulsar period (e.g., 11.123 Hz is used at Parkes). For HI observations the calibration signal can be pulsed at a slower rate (e.g., 1Hz or lower).

calAmp: 1
calFreq: 11.123
calDuty: 0.3

```
./simulateCal -o calNoise.dat  -p params.dat
```

will produce a calibration file that can be combined as usual with system noise, pulsar files etc. For this particular example we get:



Single pulse variability

Scintillation

Sky temperature and continuum sources

Using real data files

For the examples above all the data are simulated. However, the simulations will never be completely realistic. Your particular telescope will be affected by noise sources specific to that telescope. Therefore you may wish to inject simulated singles into actual observations. This is non-trivial as your actual observation data file will already have been digitised. For instance, if your data is 1-bit digitised then adding any signal to your data will produce the value "1" which is not realistic.

Output data files

The default output format for the software is the PSRFITS format. It is also possible to select SIGPROC output using:

outputFormat: sigproc

For long observations (either long pointed observations or drift-scan observations) it is common to split the observation into multiple files. These files can subsequently be combined back together again as needed. The simulation software enables this using:
fileLen: 600

The total file length will be close to 600 seconds (the exact time will be automatically chosen based on the sampling time).

5% of Tsys for 1 minute on and then 1 minute off
2x Tsys for 1 second every 60 seconds
5% of Tsys for 5 minutes on and then 5 minutes off

5 minute files

Making your own modules

Simulating a calibration source

```
./simulateCal -o scal.dat -to scal.txt 0 50 0 3
```

Creating a PSRFITS file

```
./createSearchFile -o search.sf -f snoise.dat -f scal.dat
```

Each binary data file (the output files given above) are stored as floats. Each file also has a header containing:

- Format (64 characters: currently "FORMAT 1"; this may be changed in later versions)
- Description (128 characters - this could be the pulsar name, or the system configuration etc.)
- t1 (float; start time)
- t2 (float; end time)
- tsamp (float; sample time)
- f1 (float; frequency 1)
- f2 (float; frequency 2)
- nchan (number of channels)
- Right ascension in degrees (this should be set for e.g., a pulsar, FRB and set to 0 for data that does not depend on position; float)
- Declination in degrees (this should be set for e.g., a pulsar, FRB and set to 0 for data that does not depend on position; float)
- Scale with angle (1 = yes, 0 = no)
- Random number seed (long value)

Working on:

- simulate pulsars
  - realistic arrival times
  - scattering
  - DM smearing
  - Single pulse statistics
    - amplitude variations
    - nulling
    - intermittency
    - drifting
    - pulse shape variations
- RFI
  - broad-band, impulsive
  - narrow-band
  - 50 Hz
- Burst events
  - FRBs
  - Aliens
- 2/4bit digitisation
- Realistic cal
- Realistic data file creation
- Tsky
- Level setting

Simulating baseband data

Simulating fold-mode data files

Simulating timing data sets