CSIRO PUBLISHING
www.publish.csiro.au/journals/pasa *Publications of the Astronomical Society of Australia*, 2011, **28**, 1–14

DSPSR: Digital Signal Processing Software for Pulsar Astronomy

W. van Straten[A,B] *and M. Bailes*[A]

[A] Swinburne University of Technology, PO Box 218, Hawthorn, VIC 3122, Australia
[B] Corresponding author. Email: vanstraten.willem@gmail.com

Received 2010 June 28, accepted 2010 August 24

Abstract: DSPSR is a high-performance, open-source, object-oriented, digital signal processing software library and application suite for use in radio pulsar astronomy. Written primarily in C++, the library implements an extensive range of modular algorithms that can optionally exploit both multiple-core processors and general-purpose graphics processing units. After over a decade of research and development, DSPSR is now stable and in widespread use in the community. This paper presents a detailed description of its functionality, justification of major design decisions, analysis of phase-coherent dispersion removal algorithms, and demonstration of performance on some contemporary microprocessor architectures.

Keywords: methods: data analysis — polarisation — pulsars: general — techniques: polarimetric

1 Introduction

The main technical challenges that present themselves to pulsar astronomers arise from two opposing forces: the drive for greater sensitivity and the adverse effects of propagation through the interstellar medium (ISM). For a given antenna and observing schedule, greater sensitivity may be achieved by increased instrumental bandwidth, which comes at the cost of additional signal distortion primarily due to plasma dispersion in the ISM. A wide variety of pulsar instrumentation has been developed to either mitigate or eliminate interstellar dispersion; these may be broadly classified as either post-detection or pre-detection techniques.

Post-detection methods employ a filterbank, either analog or digital, to divide the wideband signal into narrow channels. The voltage signal in each frequency channel is detected and the inter-channel dispersion delays are removed before integrating over frequency. Post-detection dispersion removal is fundamentally limited by residual intra-channel dispersion smearing and reduced temporal resolution, both of which are inversely proportional to the number of filterbank channels used. In contrast, pre-detection (or phase-coherent) dispersion removal completely eliminates dispersion smearing while retaining the original time resolution of the observed signal (Hankins & Rickett 1975). Phase-coherent dedispersion is applied directly to the voltage signal, which must be sampled at the Nyquist rate and either recorded or reduced using a high-speed digital signal processing system. Owing to the relatively high demand on both data storage and computing resources, the earliest baseband recording and processing systems were limited to relatively small bandwidths. For example, coherent dedispersion was pioneered using a baseband recorder with a modest bandwidth of 125 kHz, which could record with a 20% duty cycle for only 3 minutes before filling the magnetic tape (Hankins 1971).

Fortunately, the capacity of baseband recording and processing systems closely follows the continual growth of affordable commercial computing and data storage technologies, and by the mid-1990s a number of baseband recorders were in regular use at radio observatories around the world. The Princeton Mark IV system, capable of continuously 2-bit sampling a 2×10 MHz bandpass (Shrauner 1997; Stairs 1998; Stairs et al. 2000), was in use at the Arecibo and Jodrell Bank observatories. At Parkes, baseband recorders included the Wide Bandwidth Digital Recording (WBDR) system, capable of recording a 2×50 MHz bandpass (Jenet et al. 1997); the S2 VLBI recorder, a 2×16 MHz system (Wietfeldt et al. 1998); and the Caltech Parkes Swinburne Recorder (CPSR; van Straten et al. 2000; van Straten 2003), a 2×20 MHz version of the Caltech Baseband Recorder (CBR) at Arecibo. Each of the above systems recorded the baseband signal to magnetic tape for transport and offline analysis on high-performance computing resources. The Princeton Mark IV system could also record to a 100 GB disk array for offline processing. Also in use at Green Bank, Effelsberg, and Arecibo were versions of the coherent Berkeley Pulsar Processor (cBPP), which employed digital signal processing hardware to perform coherent dedispersion in real time. Depending on the pulsar dispersion measure, these systems were capable of correcting a bandpass of up to 2×112 MHz (Backer et al. 1997).

At Parkes, CPSR2 led the evolution away from recording on magnetic tape by using a modest cluster of workstations at the telescope to process 2×128 MHz of bandwidth in quasi-real time (Bailes 2003; Hotan 2006). Similarly, at the Westerbork Synthesis Radio Telescope,

 10.1071/AS10021 1323-3580/11/01001

the second generation of the Dutch Pulsar Machine, PuMa-II, processes 2×160 MHz of bandwidth in near real time, using 8-bit sampling to significantly reduce quantisation distortions (Karuppusamy et al. 2008). These systems use large-capacity high-speed disks to buffer the incoming baseband signal before it is processed.

The first coherent dedispersion software to work completely in memory was pioneered through the development of the Arecibo Signal Processor (ASP), Green Bank Astronomical Signal Processor (GASP), and Berkeley Orleans Nancay (BON) series of high dynamic range instruments, which use ring buffers in shared memory to process 2×128 MHz in real time (Demorest et al. 2004; Demorest 2007). Similar in design, the ATNF Parkes Swinburne Recorder (APSR) is capable of real-time processing 2×1 GHz of bandwidth with 2 bits/sample, or 2×256 MHz with 8 bits/sample. APSR uses the same digital signal processing software that was developed at Swinburne to reduce the baseband data recorded by the S2 and CPSR instruments. Later named DSPSR,[1] this software evolved through the development of PuMa-II at WSRT, and is now utilised to process data from a wide variety of instruments, including the Australian Long Baseline Array Data Recorder (LBADR) and the Giant Metrewave Radio Telescope (GMRT) software backend (Roy et al. 2010). The real-time processing requirements of APSR motivated the implementation of the shared memory and multi-threaded capabilities of DSPSR.

More recently, advances in general-purpose computing on graphics processing units (GPUs) have rapidly transformed the state of the art of digital signal processing. The Green Bank Ultimate Pulsar Processing Instrument (GUPPI) first demonstrated the use of GPUs to perform phase-coherent dispersion removal in real time (Demorest et al. 2010). Following these developments, DSPSR was extended to use the Compute Unified Device Architecture (CUDA) developed by NVIDIA®.[2] This new capability was driven by the development of a GPU-based digital signal processing system for Parkes using the Interconnect Break-out Board (IBOB) produced by the Center for Astronomy Signal Processing and Electronics Research (CASPER) at Berkeley. The CASPER Parkes Swinburne Recorder (CASPSR) performs real-time phase-coherent dispersion removal on 2×400 MHz using four server-class workstations, each equipped with two NVIDIA® Tesla C1060 GPUs.

Through these stages of evolution and refinement, DSPSR has amassed an extensive range of useful features. These are briefly summarised in Section 2, followed by a more detailed description of the algorithms developed for radio pulsar astronomy. The performance of the library is demonstrated in Section 3 using currently available technology and common observing configurations. In Section 4, the future of the software project is discussed with regard to the upcoming generation of radio telescopes.

[1] `http://dspsr.sourceforge.net`.

[2] `http://www.nvidia.com/cuda`.

2 Algorithms and Features

The DSPSR software processes continuous streams of radio pulsar astronomical data, producing integrated statistics such as the phase-resolved average polarisation of the pulsar signal. The salient features of this software are

- automatic excision of invalid data, such as those lost during transfer or corrupted by impulsive interference;
- correction of digitisation distortions via dynamic output level setting (Jenet & Anderson 1998);
- phase-coherent dispersion removal (Hankins & Rickett 1975) with optional Jones matrix convolution for high time-resolution polarimetry (van Straten 2002);
- synthetic filterbank formation with concurrent coherent dedispersion and/or Jones matrix convolution;
- full-polarimetric detection;
- computation of phase-resolved averages (folding) using either a polynomial approximation to the pulsar phase model, a constant period, or acceleration search parameters;
- formation of sub-integrations of arbitrary length, including single pulses;
- simultaneous folding of multiple pulsars, such as globular cluster or double pulsars;
- exploitation of multiple processing cores using parallel computing threads;
- accelerated computation on graphics processing units;
- transparent buffering of input data to handle the edge effects of certain algorithms, such as the overlap-save method of discrete convolution;
- real-time signal processing directly from a ring buffer in shared memory; and
- time-division multiplexing: discontiguous segments of data may be processed as a single stream to minimise initialisation time.

The following sections describe these features in greater detail while outlining a typical pulsar signal processing path.

2.1 Invalid Data Excision

Any operation may flag sections of data as invalid, such that they are ignored by subsequent components in the signal processing chain. This design feature is implemented as an array of weights that is maintained in parallel with the blocks of data that represent the astronomical signal. The weights array is typically used to flag segments of data in the time domain, such that a single weight applies to all frequency channels and polarisations over some epoch. However, the weights array can also be used to flag one or more specific frequency channels as invalid.

Many of the classes used to convert between n-bit and floating-point representations of the digitised signal monitor the data and flag sections as invalid whenever

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

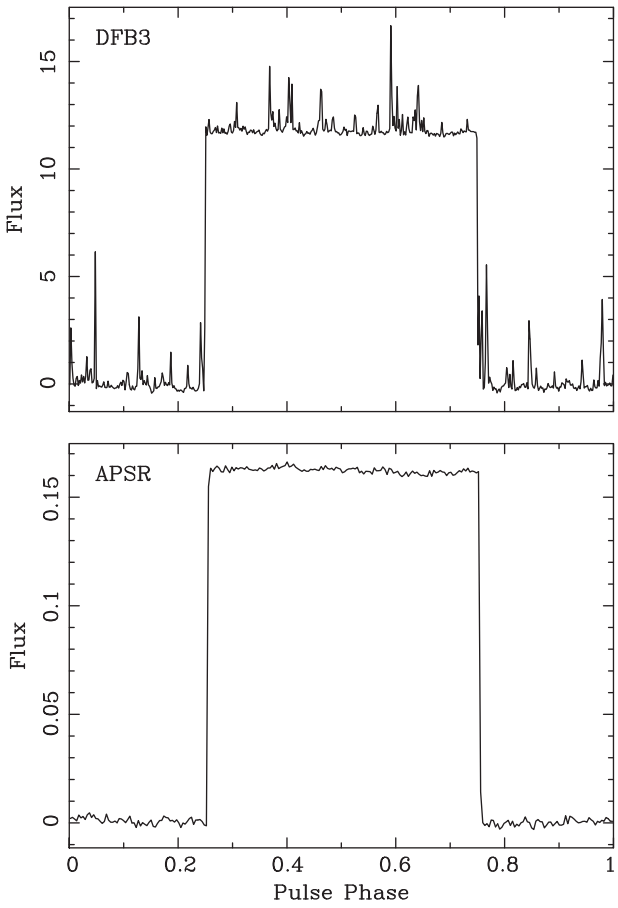

Figure 1 Automatic impulsive interference excision by DSPSR during a lightning storm. A noise diode coupled to the receptors of the Parkes 20 cm Multibeam receiver was modulated by a square wave with a 50% duty cycle. This calibrator signal was observed at a centre frequency of 1369 MHz using the Parkes Digital Filterbank (PDFB3, above) and the ATNF Parkes Swinburne Recorder (APSR, below). Both instruments observed the same 2×256 MHz band and the flux scale in each plot is uncalibrated. Each of the four 30-second integrations output by PDFB3 are irreversibly corrupted by broadband impulsive noise. However, by discarding only the contaminated segments of APSR data (less than 0.05% of the signal) the 2-minute integration produced by DSPSR is greatly improved.

certain statistics (such as the noise power) fall outside the acceptable ranges of operation. The weights array is used by operations that perform integrations (such as the pulse profile folding operation described in Section 2.6) to ensure that final results are not corrupted by invalid data. For example, invalid data may be recorded before the instrumentation is properly initialised. As illustrated in Figure 1, strong bursts of impulsive interference such as lightning are completely excised using this technique.

2.2 Dynamic Output Level Setting

Analog-to-digital conversion is a non-linear process that introduces both noise and signal distortion. Dynamic output level setting aims to correct the quantisation distortion by restoring the linear relationship between digitised and undigitised power (Jenet & Anderson 1998, hereafter JA98). In DSPSR this correction is performed during the conversion from 2-bit to floating-point representations of the digitised voltage samples. Referring to section 6 of JA98, the data are divided into consecutive segments of L digitised samples. In each segment, the number of low voltage states, M, is counted and the undigitised power, σ^2, is estimated by inverting equation 45 of JA98,

$$\Phi = \frac{M}{L} = \mathrm{erf}\left(\frac{t}{\sqrt{2}\sigma}\right), \quad (1)$$

where the error function,

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-y^2} dy, \quad (2)$$

and t is the optimal threshold between low and high voltage states. Equation (1) is solved for σ using the Newton-Raphson method, and it is computationally prohibitive to perform this iterative calculation for every L-point segment. However, as there are only $L + 1$ possible values of M, DSPSR utilises a lookup table of precomputed output levels, which are stored for only an acceptable range of input power estimates. Those segments of data for which M falls outside of the acceptable range are flagged as invalid (as described in Section 2.1). DSPSR also maintains a histogram of occurrences of M that is archived for later use as a diagnostic tool when assessing the quality of the baseband recording.

2.3 Coherent Dedispersion

Electromagnetic waves propagating through the interstellar medium (ISM) experience phase dispersion that effects a frequency-dependent group velocity. Describing the ISM as a cold, tenuous plasma, the frequency response function,

$$H(\nu + \nu_0) = \exp\left(i \frac{2\pi D \nu^2}{\nu_0^2(\nu + \nu_0)}\right), \quad (3)$$

is obtained and used to deconvolve the observed radio signal (Hankins 1971; Hankins & Rickett 1975). Here, ν_0 is the centre frequency of the observation and the dispersion D is related to the more commonly used dispersion measure DM by (Manchester & Taylor 1972)

$$DM \ (\mathrm{pc\ cm^{-3}}) = 2.41 \times 10^{-4} D \ (\mathrm{s\ MHz^2}). \quad (4)$$

The dispersion measure is equal to the integral of the free electron density along the line of sight to the pulsar. Although the constant of proportionality has been derived with greater precision (Backer et al. 1993), Equation (4) is the standard adopted by pulsar astronomers and implemented by DSPSR.

The duration of the phase dispersion impulse response function is called the sweep or smearing time, t_d. It is equivalent to the width of a band-limited delta-function

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

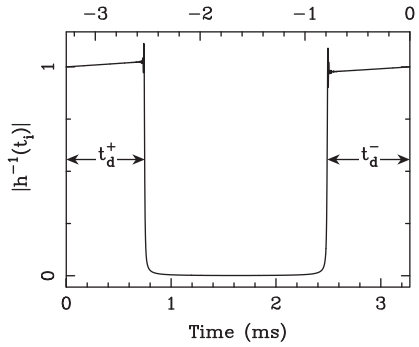

Figure 2 The magnitude of the dedispersion impulse response function, $|h^{-1}(t_i)|$, as calculated from the inverse FFT of the frequency response function, $H^{-1}(v_k)$, and normalized to unity at the origin ($t = 0$). For this example, $v_0 = 600$ MHz, $\delta v = 20$ MHz, $DM = 2$, and $N = 65\,536$ points. Reflecting the cyclicity of discrete convolution, time is labelled in both the positive (bottom axis) and negative (top axis) directions with respect to the origin, which recurs at the left and right boundaries of the plot. At the edges of the response function, where the magnitude drops to zero, the ringing is due to the cyclic discontinuity in $H^{-1}(v_k)$ at $v = 0$. The smearing in the upper half of the band, $t_d^+ = 750$ µs, and lower half of the band, $t_d^- = 788$ µs, result in asymmetry about the origin. This asymmetry also causes the observed slope in the magnitude of the impulse response; the flux at low frequency is smeared over more time than the flux at high frequency, but the integrated fluxes in the two halves of the band are equal. Note that t_d^+ and t_d^- correspond to m_+ and m_- in Figure 13.1.3 of Numerical Recipes.

after passage through the ISM, and is a function of the bandwidth, δv, and centre frequency, v_0, such that

$$t_d = D(\nu_{\rm min}^{-2} - \nu_{\rm max}^{-2}), \tag{5}$$

where $v_{\rm min} = v_0 - \delta v/2$, and $v_{\rm max} = v_0 + \delta v/2$.

Deconvolution is most efficiently performed in the frequency domain, where the observed signal is simply multiplied by the inverse of the discrete form of Equation (3). This results in cyclical convolution, as described in Chapter 18 of Bracewell (1986) and Section 13.1 of Numerical Recipes (Press et al. 1992, hereafter NR). Figure 2 (cf. Figure 13.1.3 of NR) illustrates that each time sample output by the cyclical convolution operation will depend upon the $n_d^+ = rt_d^+$ points preceding it and the $n_d^- = rt_d^-$ points following it, where r is the sampling rate and t_d^+ and t_d^- are the dispersion smearing times in the upper and lower halves of the band. Owing to the periodicity assumption of the discrete Fourier transform, the first n_d^+ points in the result of the convolution operation will be erroneously mixed with data wrapped around from the end of the data segment, and the last n_d^- points of the result will be erroneously mixed with time samples from the beginning of the input segment (cf. Figure 18.4 of Bracewell 1986). Consequently, the polluted points (called the wrap-around region) from the result of each transformation step are discarded, a step that forms part of the overlap-save method of discrete convolution. This method, depicted in Figure 3, also accounts for the fact that the duration of the impulse response function is generally much shorter than that of the signal to be convolved.

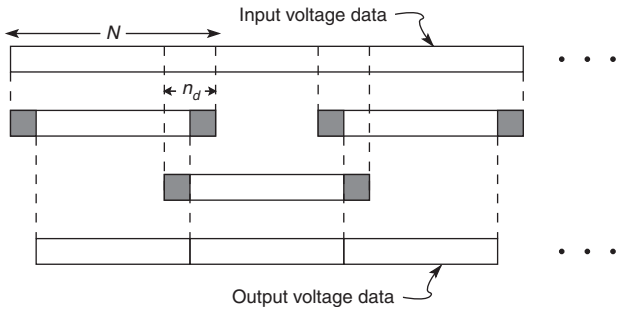

Figure 3 The overlap-save method of discrete convolution. The input data (top) are divided into N-point segments that overlap by the number of non-zero points, n_d, in the discrete phase dispersion impulse response function. Each N-point segment is separately transformed into the Fourier domain, multiplied by the frequency response function, $H^{-1}(v_k)$, and transformed back into the time domain (middle). Polluted points from the wrap-around region (shaded grey) are discarded before copying the result into the output data stream (bottom).

Noting the quadratic form of Equation (5), it can be seen that the lowest frequency must be advanced by an amount greater than the delay applied to the highest frequency. This results in asymmetry of $h(t_i)$ around $t = 0$, which must be taken into account when discarding wrap-around points from the result of each cyclical convolution operation. As $n_d = n_d^- + n_d^+$ points are discarded, the number of samples N in the result of the Fast Fourier Transform (FFT) must be greater than n_d. N is chosen by minimising the number of floating point operations per second,

$$\text{flops} = \frac{\mathcal{O}_{\rm FFT}(N)}{N - n_d} 2\delta\nu, \tag{6}$$

where $\mathcal{O}_{\rm FFT}(N)$ is the number of floating point operations required to compute the FFT of N complex input samples, which asymptotically approaches $5N\log_2 N$ when N is an integer power of two.[3] This equation is valid for both real-valued and complex-valued input data (see the Appendix for discussion). The factor of 2 arises because both forward and backward transforms are computed. Equation (6) provides an estimate of the minimum number of flops required to process the data in real time. This number can be directly compared with FFT benchmarks, such as those computed by BENCHFFT.[4] As discussed in more detail in Section 3, actual performance measurements can be used to better select the optimal FFT length.

To perform the FFT, DSPSR can utilise both open-source and proprietary FFT libraries, including the Fastest Fourier Transform in the West (FFTW; Frigo & Johnson 2005) the Intel® Math Kernel Library[5] and Intel® Integrated Performance Primitives (IPP).[6] On allocation,

[3] http://www.fftw.org/speed.

[4] http://www.fftw.org/benchfft.

[5] http://www.intel.com/software/products/mkl.

[6] http://www.intel.com/software/products/ipp.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

all arrays are aligned on 16-byte boundaries so that vector instruction sets, such as Streaming SIMD Extensions (SSE), may be exploited.

Invalid Data Excision: Referring to Figure 3, if any of the input voltage data in an N-point segment are flagged as invalid, then the all of the N–n_d samples of the output voltage data for that cyclical convolution step will be flagged.

2.4 Synthetic Filterbank

Though phase-coherent dispersion removal retains the original temporal resolution of the digitised signal, this resolution is often integrated away when forming the average pulse profile. It therefore proves useful to trade some time resolution for frequency resolution by dividing the observed band into a number of sub-bands, or channels. Both the smearing time, t_d, and the sampling rate, r, in each sub-band is reduced, resulting in smaller n_d; this reduces the transform length, N, required for subsequent coherent dedispersion, and thereby improves computational efficiency. Furthermore, with greater frequency resolution the measured polarisation may be better calibrated, or individual sub-bands corrupted by narrowband radio-frequency interference (RFI) may be deleted. Finally, when frequency resolution is retained, a better estimate of the dispersion measure may be applied at a later time to correct the inter-channel dispersion delays before further integrating in frequency. Using DSPSR, the observed signal may be divided into N_c frequency channels using either the deprecated or convolving filterbank, both of which are based on the FFT.

2.4.1 Deprecated Filterbank

In the deprecated filterbank, the data are divided into non-overlapping segments of N_c complex points (or $2N_c$ real points). Each segment is transformed into the Fourier domain, where each of the N_c complex spectral values is treated as a time sample in one of N_c independent signals, each with bandwidth, $\delta\nu' = \delta\nu/N_c$, and sampling rate, $r' = r/N_c$. Phase-coherent dispersion removal is then separately performed on each of the N_c resulting channels using unique phase dispersion frequency response functions, each tuned to the centre frequencies of the output filterbank channels.

This method of synthetic filterbank formation is equivalent to the coherent filterbank described in Section 3.1.3 of Jenet et al. (1997). Although it reduces the computational cost of coherent dedispersion (see the following section and the Appendix for further discussion), the technique suffers from the spectral leakage of the discrete Fourier transform, which mixes a significant amount of power between neighbouring frequency channels (NR, Section 13.4). When the interchannel dispersion delay is large, the artifacts of spectral leakage are readily observed as delayed images of the pulse profile, as shown in Figure 4.

Here, the dispersion delay between the 500 kHz channels of the deprecated filterbank (which ranges from ~88 to 154 μs across the band) is seen between each peak (main and inter-pulse) and the leakage artifacts on either side of it. The remaining artifacts in the APSR convolving filterbank are introduced by the Parkes Digital Filterbank (PDFB3), which is used to digitise and divide the 256 MHz band into 16 smaller 16 MHz bands. The two-stage (1024-channel analysis followed by 16×64-channel synthesis) polyphase filterbank implemented by the PDFB3 introduces spectral leakage between the 250 kHz channels produced by the analysis filterbank; the dispersion delay between these channels ranges from ~44 to 77 μs across the band. Work is currently underway to correct this artifact using a single-stage 16-channel analysis polyphase filterbank.

In contrast, CASPSR uses time-division demultiplexing to implement parallel processing on multiple GPUs; therefore, spectral leakage is limited to that introduced by the forward 512 k-point FFT in the convolving filterbank used to process these data. At this frequency resolution, the dispersion delay between neighbouring channels varies between only ~0.11 and 0.27 μs, which is less than the temporal resolution of the folded profile.

2.4.2 Convolving Filterbank

The spectral leakage function may be improved by the application of a window function in the time domain (NR, Figure 13.4.2). However, the physical bandwidth of the spectral leakage function may also be decreased by increasing the length of the Fourier transform by a factor, N'. Where $K = N' \times N_c$, each K-point segment of data is transformed into the Fourier domain, and the result is divided into N_c non-overlapping segments of N' spectral points. Each segment is inverse-transformed back to the time domain, producing N' time samples from N_c independent filterbank channels.

Coherent dedispersion may be performed simultaneously with this filterbank technique. While in the Fourier domain, each of the N_c spectral segments is multiplied by a unique dedispersion frequency-response function. In this case, N' must be chosen to match the size of the response function. As the same inverse transform size is applied to each sub-band, N' must be large enough to accommodate the maximum smearing time, $t'_d = D(\nu_{\min}^{-2} - \nu'^{-2}_{\max})$, corresponding to the sub-band with the lowest centre frequency, where $\nu_{\min} = \nu_0 - \delta\nu/2$, and $\nu'_{\max} = \nu_{\min} + \delta\nu/N_c$. Owing to the quadratic form of Equation (5), $t'_d > t_d/N_c$. Although the smearing is less severe in those sub-bands with higher centre frequencies, symmetry dictates that $n'_d = r' t'_d > n_d/N_c^2$ wrap-around points are discarded from the result of each of the N_c inverse transforms (cyclical convolution products). Each K-point segment of input data therefore overlaps by $N_c \times n'_d$ points.

As shown in the Appendix, phase-coherent dispersion removal using either the deprecated or convolving

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

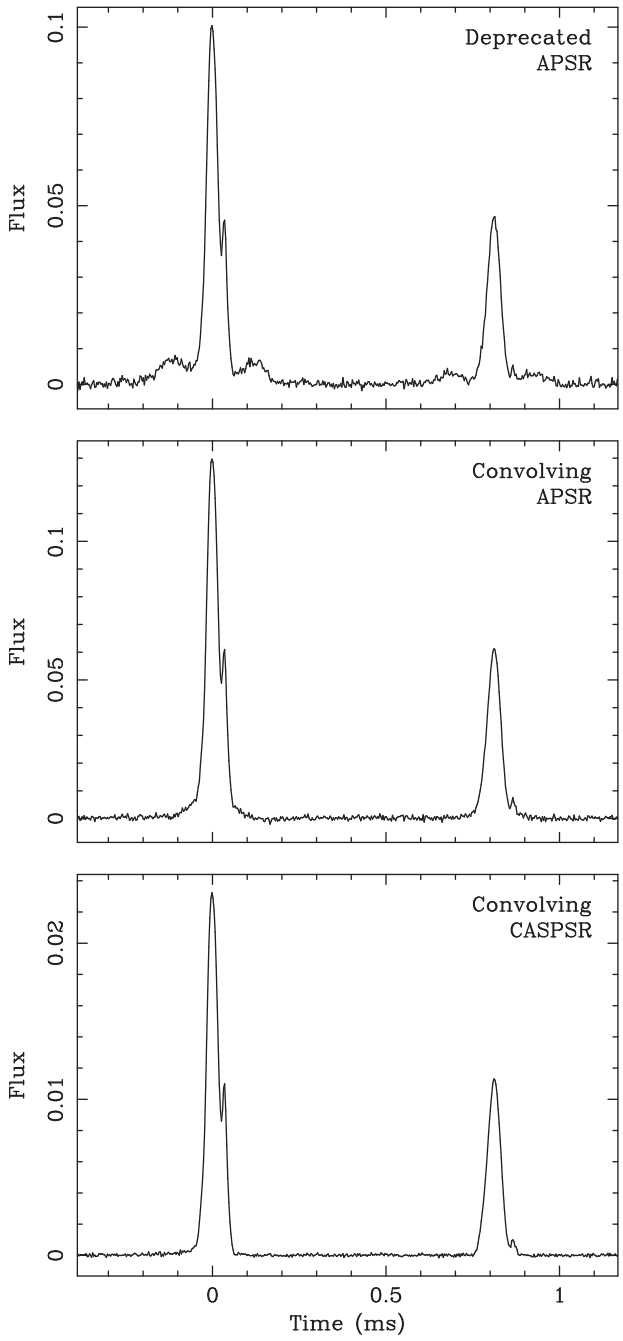

Figure 4 Correction of spectral leakage using the convolving filterbank. PSR B1937+21 was observed at a centre frequency of 1369 MHz using the 2×256 MHz band of APSR (top two panels) and at a centre frequency of 1382 MHz using the 2×400 MHz band of CASPSR (bottom panel). The APSR data were processed using a synthetic filterbank as implemented by the deprecated (top panel) and convolving (middle panel) methods. The residual distortion in the APSR data (most obvious is the triangular base of the main pulse in the middle panel) is due to the Parkes Digital Filterbank, as described in the text. The uncorrupted CASPSR observation best demonstrates the fidelity of the convolving filterbank.

filterbank methods requires the same number of floating point operations per second,

$$\text{flops}' = \frac{5N'(\log_2 N_c + 2\log_2 N')}{N' - n'_d}\delta\nu. \qquad (7)$$

Note that the size of the first FFT, $K = N_c \times N' > n_d/N_c$, is inversely proportional to the number of filterbank channels, N_c. As smaller transform lengths improve processing efficiency, N_c should be chosen to be as large as possible without sacrificing required time resolution. This remains true only up to the limit where the resulting sampling interval, $t'_s = 1/r'$, becomes larger than t'_d (i.e. where $n'_d \leq 1$).

As shown in Figure 4, the convolving filterbank method eliminates the spectral leakage artifacts produced using the deprecated method. Synthetic filterbank formation with simultaneous convolution is summarised by the following algorithm:

1. Divide the signal into K-point segments ($K = N_c \times N'$) that overlap by $N_c \times n'_d$ points (cf. top of Figure 3).
2. For each segment, perform the K-point FFT, and divide the result into N_c non-overlapping segments, or sub-bands, of N' points.
3. For each of the N_c sub-bands:
   (a) multiply the N'-point sub-band by its unique frequency response function;
   (b) perform an N'-point inverse FFT;
   (c) discard the n'_d wrap-around points; and
   (d) copy the remaining time samples into the corresponding output frequency channel.

Invalid Data Excision: In addition to the discussion of invalid data excision in Section 2.3, synthetic filterbank formation reduces the number of time samples per weight by a factor of N_c. If N_c is greater than the number of samples per weight, then the weights array will be rebinned with care to ensure that invalid data remain flagged.

2.5 Detection

If the voltage signals from both receptors of a dual-polarisation receiver are available, DSPSR computes the polarisation of the electromagnetic radiation. Radio pulsar polarimetry yields information about the physics of the emission mechanism (e.g. Edwards & Stappers 2004), the geometry of the pulsar's magnetosphere (e.g. Johnston et al. 2005), and the average magnetic field of the ISM along the line of sight (e.g. Han et al. 2006). It also provides additional constraints that may be exploited for high-precision pulsar timing (van Straten 2006).

Polarisation is described by the second-order statistics of the transverse electric field vector $\boldsymbol{e}$, as given by the complex 2×2 coherency matrix, $\rho \equiv \langle e\, e^\dagger \rangle$ (Born & Wolf 1980). Here, the angular brackets denote an ensemble average, $e^\dagger$ is the Hermitian transpose of $\boldsymbol{e}$, and an outer product is implied by the treatment of $\boldsymbol{e}$ as a column vector. A pulse phase-resolved ensemble average is performed by the folding operation described in the next section. When detecting synthetic filterbank data, the coherency matrix is computed independently in each frequency channel.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

A useful geometric relationship between the complex two-dimensional space of the coherency matrix and the real four-dimensional space of the Stokes parameters is expressed by the following pair of equations:

$$\rho = S_k \sigma_k / 2, \tag{8}$$

$$S_k = tr(\sigma_k \rho). \tag{9}$$

Here, S_k are the four Stokes parameters, Einstein notation is used to imply a sum over repeated indices, $0 \leq k \leq 3$, σ_0 is the 2×2 identity matrix, σ_{1-3} are the Pauli matrices, and *tr* is the matrix trace operator. The Stokes four-vector is composed of the total intensity S_0 and the polarisation vector, $\boldsymbol{S} = (S_1, S_2, S_3)$. Equation (8) expresses the coherency matrix as a linear combination of Hermitian basis matrices; equation (9) represents the Stokes parameters as the projections of the coherency matrix onto the basis matrices.

The fourth-order moments of the electric field, as described by the four-dimensional covariance matrix of the Stokes parameters, can optionally be formed and averaged as a function of pulse phase. The covariances of the Stokes parameters provide additional information about the pulsar radiation, such as the presence of orthogonally polarised modes of emission and the degree of correlation between mode intensities (McKinnon & Stinebring 1998; van Straten 2009).

2.6 Folding

Ensemble averages of the detected signal are computed as a function of topocentric pulse phase during a process commonly called folding. Apparent topocentric pulse phase is determined using a polynomial approximation to the pulsar timing model, $\mathcal{P}_\phi(t)$, generated using either the TEMPO (Taylor & Weisberg 1989)[7] or TEMPO2 (Hobbs et al. 2006)[8] software packages. The detected signal of interest, $Y(t_i)$, is folded by integrating each of its samples into one of n equally spaced phase bins, $Y(\phi_k)$, where bin number, k, is given by

$$k = (n\mathcal{P}_\phi(t_i)) \bmod n, \tag{10}$$

where $\mathcal{P}_\phi$ has dimensionless phase (turns) and the mod operator returns the remainder after division, such that $0 \leq k < n$.

Typically, a vector of processes, $\boldsymbol{Y}(t_i)$ is folded in parallel with one count, $N(\phi_k)$, of the number of time-samples accumulated in each phase bin. For example, $\boldsymbol{Y}(t_i)$ may represent the four Stokes parameters in each of the N_c frequency channels. The average pulse profile of each process is then given by

$$\langle Y_j(\phi_k) \rangle = Y_j(\phi_k)/N(\phi_k). \tag{11}$$

The start and end times, t_0 and t_N, of the observation are used to assign the mean time of the rising edge of phase bin zero. Under the assumption that $\mathcal{P}_\phi$ and its inverse may be calculated with sufficiently high accuracy, this is given by

$$\tau_0 = \mathcal{P}_\phi^{-1}(\mathcal{P}_\phi((t_0 + t_N)/2) \bmod 1). \tag{12}$$

During the folding operation, DSPSR can divide the signal into segments with arbitrary start and end times and produce a unique set of pulse profiles for each segment. This feature can be used to divide a long observation into regular intervals, or sub-integrations, that may be later combined using an updated version of the pulsar timing model, or following the deletion of sub-integrations that have been corrupted by instrumentation faults or radio-frequency interference.

The start and end times of each sub-integration can also be made to coincide (within the temporal resolution of the signal) with integer values of pulse phase, thereby producing single-pulse profiles. In single-pulse mode, it is usually necessary to remove the inter-channel dispersion delays, such that each single-pulse integration output by DSPSR contains the same pulse in all frequency channels. It may also be necessary to adjust the reference phase of phase bin zero so that the on-pulse region of pulse phase is not divided across multiple integrations.

For pulsars with short spin periods, such as millisecond pulsars, it may not be feasible to store every single pulse with full frequency resolution. In such cases, it becomes desirable either to integrate the data down to some manageable size or to add some conditions such that only a subset of the single pulse integrations are kept. To this end, DSPSR employs the PSRCHIVE (Hotan et al. 2004b) command language interpreter[9] to run a user-supplied script, through which a diverse range of post-processing tasks can be executed. For example, when studying the polarisation of single pulses, the PSRCHIVE script can be used to calibrate before integrating over frequency channels. When searching for giant pulses, the PSRCHIVE script can test whether the single pulse contains any signal that is over a certain threshold above the noise.

Multiple pulsars can be folded simultaneously, which is particularly useful when observing globular clusters or double-pulsar systems. When folding multiple pulsars, the end of the signal path is forked across multiple instances of the folding operation. That is, the signal is processed identically for each pulsar up to the final stage of folding. Therefore, when performing phase-coherent dispersion removal on a set of globular cluster pulsars, the mean or median dispersion measure should be used.

When parallel threads of execution are used, each thread folds its blocks of data into its own integration of the pulse profile. On completion of an integration, each thread places its result on a stack where the folded results from all threads await combination. When no threads have data to contribute to an integration, it is processed as in the single-threaded mode of operation.

[7] `http://tempo.sourceforge.net`.

[8] `http://tempo2.sourceforge.net`.

[9] `http://psrchive.sourceforge.net/manuals/psrsh`.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

Invalid Data Excision: Any data that are flagged as invalid are omitted from integration. This feature can be disabled when studying bright pulsars and/or giant pulses.

2.7 Thread-safe Input Buffering

One or more of the operations in the signal processing chain may lose samples, such that the time–bandwidth product of the output of the operation is less than that of its input. For example, for each block of input data, coherent dedispersion discards at least n_d samples and will lose more if the input block size is not an integer number of overlapping N-point segments. One possible solution is to set the raw input data block size and load overlapping blocks of raw input data accordingly. When only one of the operations imposes a constraint on the raw input data block size and/or amount of raw input data block overlap, it is trivial to set these parameters; however, it may not be possible to satisfy the block size and overlap constraints of more than one such operation in the signal chain. Therefore, each DSPSR operation manages its own overlap requirements using a thread-safe input buffering scheme. This design feature simultaneously increases both the modularity and efficiency of the code by reducing inter-dependency between signal processing components and eliminating the need to re-process data lost by operations that occur later in the signal path.

Each operation that loses samples simply buffers the appropriate amount of data from the end of its current input data block to be later prefixed to the next input data block passed to the operation. A small amount of additional book-keeping ensures the contiguity of consecutive input data blocks passed to each operation. When multiple processing threads are used, the input data blocks are distributed across multiple signal-processing paths. Consequently, in each thread, the blocks passed on consecutive calls to an operation are not necessarily contiguous. Therefore, for each component in the signal chain that requires it, the threads share a single input buffer through which the data to be prefixed are passed to the thread that receives the next contiguous input block. This method works only when the processing threads share the same physical memory. When threads are run on different computational devices, such as multiple graphics processing units (GPUs), and the cost of transmitting the buffered data between devices is high, then DSPSR will resort to the raw input data overlap strategy.

3 Performance

Digital signal processing is demanding of computational, data communications, and data storage resources. Therefore, the performance of a baseband recording and processing system depends on a number of variables, including the number of microprocessors, their architecture and clock speed, memory size and bandwidth, cache size and bandwidth, compiler version, and optimisation options. To demonstrate some of these basic considerations, a number of illustrative benchmarks are presented using currently available microprocessor technology.

3.1 Central Processing Units

For the most computationally intensive operations in DSPSR, such as coherent dedispersion and filterbank synthesis, execution time is dominated by computing the Fast Fourier Transform (FFT); therefore, the rate at which data are processed is a strong function of the performance of the FFT library. In Figure 5, the speeds of two commonly used FFT libraries are plotted as a function of transform length, $N_{\rm FFT}$, and the number of operating threads. Similar to the BENCHFFT convention, performance in Gigaflops is defined as

$$\text{Gflops} = \frac{5N_{\rm FFT}\log_2(N_{\rm FFT})}{t_{\rm ns}}, \qquad (13)$$

where $t_{\rm ns}$ is the average time required to perform a single FFT in nanoseconds. These benchmarks were performed on a workstation with dual Intel® Xeon® E5345 (Clovertown) processors, each with 4 cores running at 2.33 GHz, 4×64 kB of level 1 (L1) cache, and 2×4 MB of level 2 (L2) cache, connected via 10.66 GB/s bus to 16 GB of Random Access Memory (RAM).

Figure 5 demonstrates the impact of L1 and L2 cache sizes on FFT performance. FFT speed increases with greater FFT length until the 32 kB of L1 cache reserved for data is filled (note that $N_{\rm FFT} = 2^{12} = 4$ k corresponds to 64 kB of memory required for input and output arrays). At this point the speed drops (most notably for FFTW) and remains constant before dropping again when the L2 cache is filled ($N_{\rm FFT} = 2^{18} = 256$ k corresponds to 4 MB of memory). As each of the 8 processing cores is equipped with its own L1 cache, there are no significant differences in FFT performance as a function of the number of operating threads for $N_{\rm FFT} < 2^{17}$. However, because each L2 cache is shared by two processing cores, the effective L2 cache per core is halved when 8 processing threads are used; in this case, as seen in Figure 5, performance drops when $N_{\rm FFT} = 2^{17} = 128$ k.

Using the performance measurements plotted in Figure 5, DSPSR can select both the optimal transform length and the best FFT library to employ during phase-coherent dispersion removal and synthetic filterbank formation. The selection is made by eliminating the $\mathcal{O}_{\rm FFT}(N) \simeq 5N\log_2 N$ assumption and using the measured time required to perform an N-point FFT while minimising Equation (6) or Equation (7). Particularly for shorter transform lengths, performance is significantly improved by this optimisation.

Overall performance is evaluated by measuring the time required to completely process a block of data normalised by the real time spanned by that block of data (the real time is equal to the sampling interval times the number of samples in the block). Equation (6) plots the

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

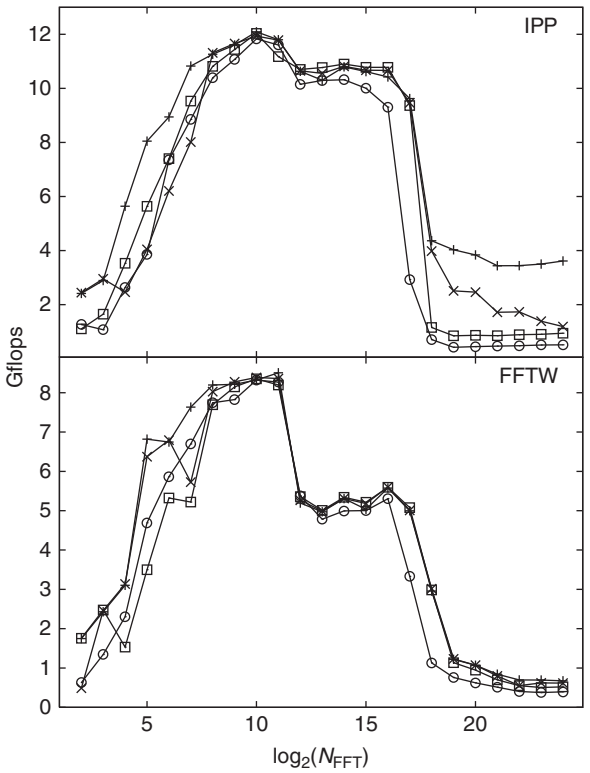

Figure 5 Fast Fourier Transform performance as a function of transform length, $N_{\rm FFT}$. Speed is measured in effective Gigaflops per processing core as described in the text. Version 3.1.2 of the Fastest Fourier Transform in the West (FFTW) and version 5.3.3 of the Intel® Integrated Performance Primitives (IPP) libraries were used. Each panel displays the results of performing a single-precision, complex-to-complex FFT on each of one (plus), two (cross), four (square), and eight (circle) processing threads. Each thread performs a separate FFT in isolation; therefore, the aggregate speed of the microprocessor is the value shown multiplied by the number of threads.

ratio of processing time to real time as a function of dispersion measure for four different instrumental configurations. For each configuration, the processing time is that required to convert 8-bit digitised data to single-precision floating-point complex numbers, form a filterbank with 500 kHz resolution while performing phase-coherent dispersion removal, detect the polarisation of the signal, and fold all frequency channels and polarisation parameters with a resolution of 1024 phase bins.

At low *DM* in Figure 6, where the FFT length is smaller than the L2 cache, DSPSR performance scales roughly linearly with the number of processing threads. At the largest *DM*, 8 threads perform slightly worse than 4 threads owing to competition for L2 cache (on the Clovertown, each L2 cache is shared by two processing cores). To further illustrate the importance of cache size and memory bandwidth, the same benchmark was repeated on a workstation with dual Intel® Xeon® E5520 (Nehalem-EP) processors, each with 4 cores running at 2.26 GHz, 4×64 kB of L1 cache, 4×256 kB of mid-level (L2) cache, and 8 MB of L3 cache, connected via 25.6 GB/s bus to 24 GB of RAM. The results are plotted in Figure 7. Performance at the lowest DM is not significantly improved, indicating that the problem is bound by the processor speed in this regime. At the highest DM, where the FFT no longer fits in L2 cache, data are processed as much as 4 times faster on the E5520, indicating that memory bandwidth must be the limiting factor on the E5345.

The relatively modest bandwidths used for the performance benchmarks presented in Figures 6 and 7 are roughly an order of magnitude smaller than those currently available with modern pulsar instrumentation. To process larger bandwidths in real time, the digitised signal must be divided between multiple workstations, the number of which depends upon the method of division. Using time-division demultiplexing, each machine must process the full bandwidth for some fraction of the time. Either the frame size must be relatively large or the frames must overlap in time to compensate for the data lost to edge effects, which may necessitate the use of one or more intermediate workstations to buffer the frames. Using frequency-division demultiplexing, each machine must process some fraction of the bandwidth for the full time. To divide the signal into frequency channels, some form of filterbank (such as a polyphase filterbank implemented on a field-programmable gate array) must be used.

When designing a digital signal processing software-based instrument, the number of workstations required to process the desired bandwidth in real time may be estimated using a benchmark similar to that presented in Figure 8. For example, considering the top panel (3 GHz), an instrument that uses time-division demultiplexing to process a 512 MHz band (circles) out to a *DM* of 1000 will require at least 11 workstations. If the signal is first divided into four sub-bands (crosses), only 8 workstations (2 per sub-band) would be required.

3.2 Graphics Processing Units

On a multiprocessor device such as a graphics processing unit (GPU), the greatest performance is achieved when all of the processors are fully utilised. To maximise utilisation, the NVIDIA® CUDA FFT library (CUFFT) enables parallel (batched) execution of multiple transforms. This feature is exploited by the GPU-based implementation of the convolving filterbank algorithm to perform the N_c inverse N'-point FFTs in parallel (refer to the algorithm at the end of Section 2.4.2). Consequently, FFT performance is a function of both N_c and $N_{\rm FFT} = N'$, as shown in Figure 9. Here, performance in Gigaflops is defined as

$$\text{Gflops} = \frac{5 N_c N_{\rm FFT} (\log_2 N_c + 2 \log_2 N_{\rm FFT})}{t_{\rm ns}}, \qquad (14)$$

where the number of floating point operations is as derived in the Appendix and $t_{\rm ns}$ is the average time in nanoseconds required to perform the forward and backward transforms in each step of the convolving filterbank

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

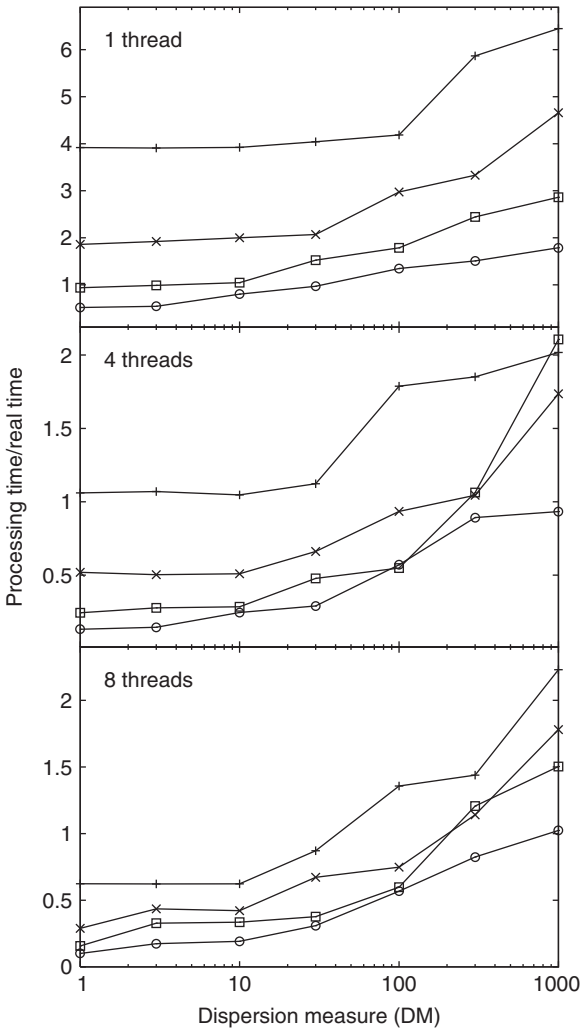

Figure 6 Processing time to real time ratio on dual Intel® Xeon® E5345 processors as a function of dispersion measure *DM*, centre frequency ν, and bandwidth $\delta\nu$. The benchmark was performed using one (top), four (middle), and eight (bottom) processing threads. Each panel plots four instrumental configurations, including $\nu = 3000$ MHz with $\delta\nu = 64$ MHz (plus), $\nu = 1500$ MHz with $\delta\nu = 32$ MHz (cross), $\nu = 750$ MHz with $\delta\nu = 16$ MHz (square), and $\nu = 375$ MHz with $\delta\nu = 8$ MHz (circle). Processing time is the total time required to perform all steps in the typical pulsar signal processing path described in the text. Fourier transforms were performed using the optimal transform length and the best of either the FFTW or IPP libraries.

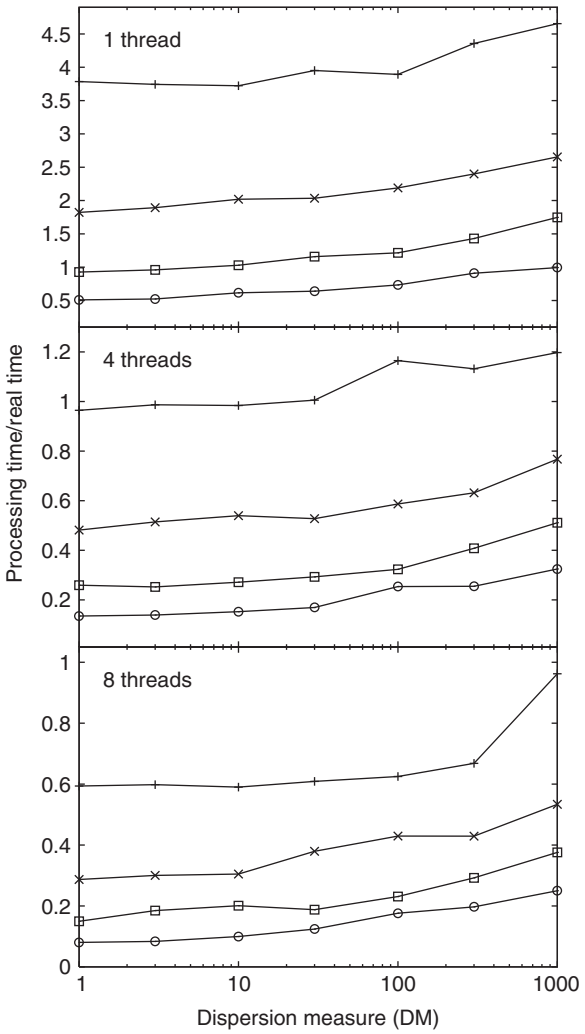

Figure 7 Processing time to real time ratio on dual Intel® Xeon® E5520 processors, as in Figure 6.

algorithm. These benchmarks were performed on a workstation equipped with an NVIDIA® Fermi architecture Tesla C2050 GPU. In contrast to the benchmarks presented in Figure 5, where memory bandwidth and cache size limits performance at large N_{FFT}, CUFFT performs best when $N_c \times N_{\mathrm{FFT}}$ is large because greater microprocessor utilisation is achieved.

Optimal GPU performance also requires minimising the amount of data transferred between the device and host memory. That is, once data are on the GPU, it is best to reduce them as much as practicable on the device. To this end, DSPSR implements GPU-based detection and folding algorithms, so that only the integrated results are transferred back to host memory. The folding algorithm exploits hardware-level multithreading to maximise utilisation by launching a processing thread for each frequency channel, polarisation, and pulse phase bin, such that the total number of independent sums to be computed is much greater than the number of available cores (448 on the Fermi architecture). By scheduling many more threads than there are processors, the latency in thread execution incurred during retrieval of data from off-chip memory is hidden by switching between active execution contexts (or *warps*: atomic groups of threads that execute simultaneously). Note that on the single-instruction, multiple-thread (SIMT) architecture employed by streaming multiprocessors, separate registers are allocated to each active thread; therefore, there is no cost associated with switching between active warps. If the maximum amount of data that fits into GPU memory corresponds to less than one pulse period and the number of frequency channels is small, then some fraction of the processors may be idle; therefore, the current folding algorithm implementation performs best on pulsars with short spin periods.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

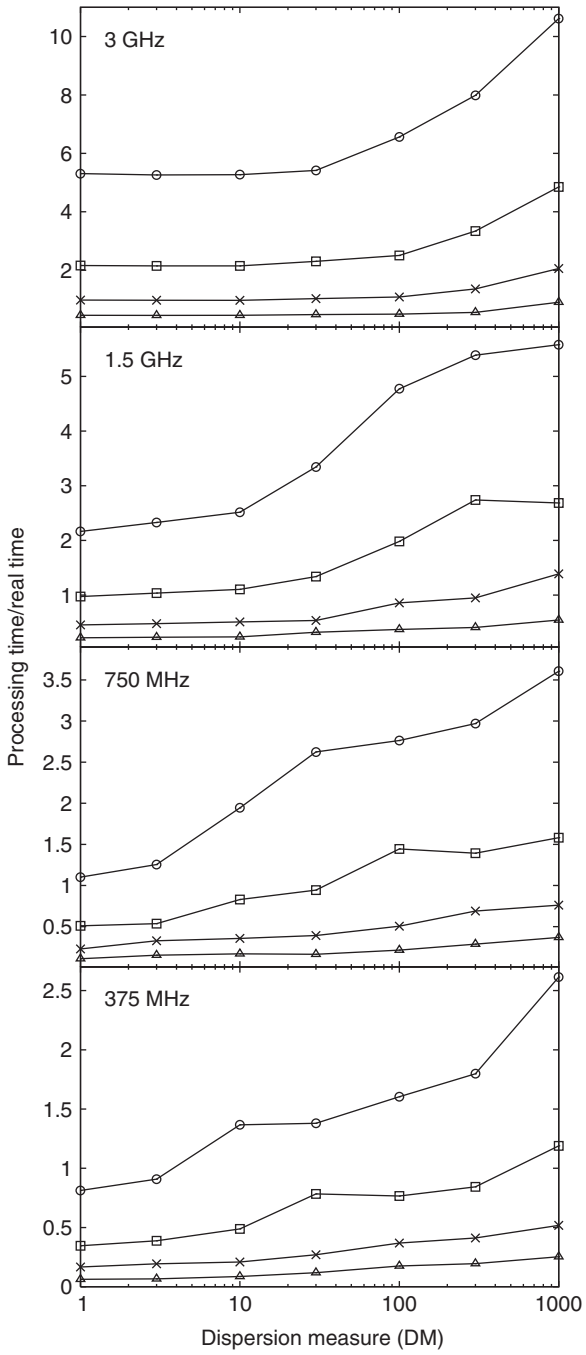

Figure 8 Processing time to real time ratio on dual Intel® Xeon® E5520 processors as a function of dispersion measure *DM*. The centre frequency ν varies from 375 MHz (bottom panel) to 3 GHz (top panel). In the bottom panel, the bandwidth $\delta\nu$ varies as $\delta\nu =$ 8 MHz (triangle), $\delta\nu = 16$ MHz (cross), $\delta\nu = 32$ MHz (square), and $\delta\nu = 64$ MHz (circle). Each symbol corresponds to the same relative bandwidth $\delta\nu/\nu$ in each panel. Processing time is defined as in Figure 6.

The overall performance of a contemporary GPU-based instrument can be estimated using the benchmarks presented in Figure 10. Focusing once again on the top panel (3 GHz), to process a 512 MHz band out to a *DM* of 1000 in real time requires only 4 GPU-equipped workstations, regardless of time- or frequency-division demultiplexing. For reference, the spin period of the pulsar used in these benchmarks is ~5.76 ms.

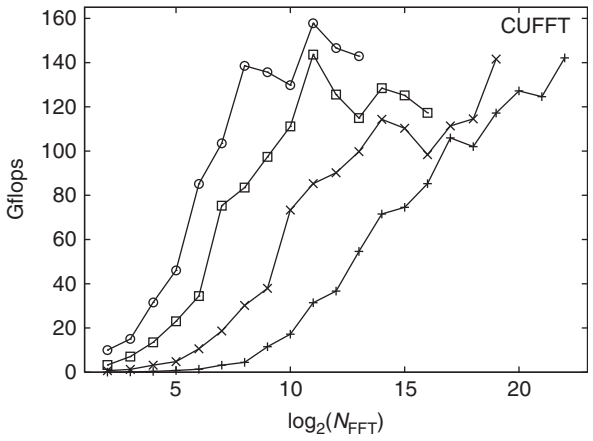

Figure 9 Filterbank performance as a function of inverse transform length $N_{\rm FFT}$ and number of frequency channels N_c. Speed is measured in effective Gigaflops for a $N_c = 4$ (plus), $N_c = 32$ (cross), $N_c = 256$ (square), and $N_c = 2048$ (circle) filterbank up to the limit of $\log_2(N_c N_{\rm FFT}) = 24$. Version 3.0 of the NVIDIA® CUDA FFT library (CUFFT) was used.

The software used to produce the benchmarks presented in this section is distributed as part of the DSPSR package. To facilitate instrumental design and planning, the latest available performance benchmarks are maintained online.[10]

4 Conclusion

DSPSR is a high-performance, general-purpose tool for radio-astronomical data reduction that has enabled a diverse range of experiments, including the discovery of the giant micropulse phenomenon (Johnston et al. 2001), a survey for sub-millisecond pulsars (Edwards et al. 2001), high-precision pulsar timing (van Straten et al. 2001; Hotan et al. 2004a; Jacoby et al. 2005; Ord et al. 2006; Verbiest et al. 2008), millisecond pulsar polarimetry (Ord et al. 2004), giant pulses (e.g. Knight et al. 2006), and dispersion measure variations (You et al. 2007), studies of giant pulses from the Crab Nebula pulsar (Bhat et al. 2008; Karuppusamy et al. 2010), the first observations of rotating radio transient polarisation (Karastergiou et al. 2009), the search for gravitational waves (Verbiest et al. 2009; Yardley et al. 2010), and an analysis of scattering by the interstellar plasma (Coles et al. 2010). DSPSR has also been used in the analysis of giant pulse data from the Mileura Widefield Array Low Frequency Demonstrator (MWA-LFD; Bhat et al. 2007), and early pulsar observations made with the Low Frequency Array Core Station 1 (LOFAR CS1; B. Stappers 2009, private communication).

Data recorded at most major radio observatories are understood by DSPSR, which can be used to process either voltage- or power-level signals (i.e. undetected or detected data). For example, DSPSR can process baseband

[10] `http://dspsr.sourceforge.net/bench`.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

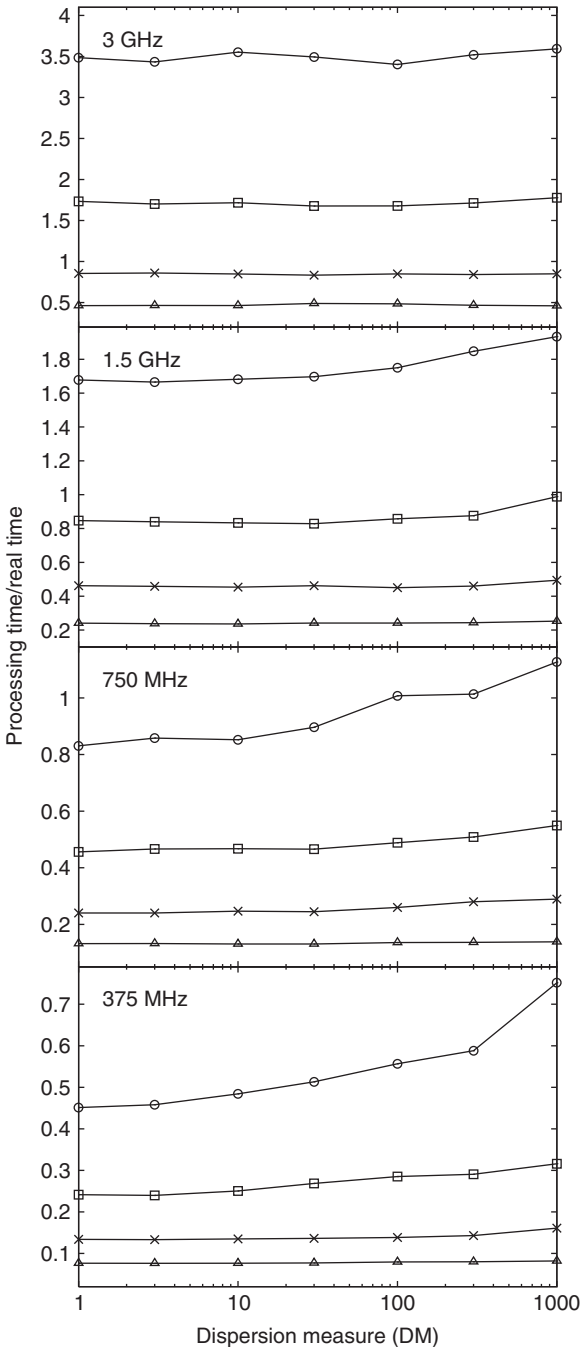

Figure 10 Processing time to real time ratio on an NVIDIA® Fermi architecture Tesla C2050 as a function of dispersion measure *DM*, as in Figure 8.

data encoded using the VLBI Data Interface Format (VDIF)[11] as well as detected data stored using the PSRFITS file format. The library also contains routines for both reading and writing files in the format used by SIGPROC, another popular signal processing package that is commonly used in the search for new pulsars, making DSPSR a useful component in survey data reduction pipelines. Although most of the design and testing of the software has been completed for the analysis of radio pulsar observations, the large majority of the algorithms may be applied to any single beam (either single-dish or phased-array) radio observation. For example, the DSPSR library could be utilised to produce high-resolution spectra for studies of large scale structure in extragalactic neutral hydrogen (e.g. Staveley-Smith et al. 2003).

Future developments of DSPSR may include the adoption of industry standards like the Open Computing Language (OpenCL)[12] for portable GPU programming. GPU performance may in turn facilitate the computation of new statistical quantities. For example, the cyclostationary statistics of the electromagnetic field promises new insight into the phenomenon of diffractive scintillation (Demorest 2010) and may enable the inversion of multipath scattering effects that arise in the interstellar medium (Walker et al. 2008). Although formation of the cyclic power spectrum is computationally prohibitive, it may be feasible to regularly observe this statistic using GPU technology.

DSPSR provides the capacity for real-time phase-coherent dispersion removal over a comprehensive range of observing configurations, including those planned for the upcoming generation of radio telescopes such as the Australian Square Kilometer Array Pathfinder (ASKAP; Johnston et al. 2008) and the Karoo Array Telescope (MeerKAT; Jonas 2009). The ASKAP Boolardy Engineering Test Array (BETA) will operate between 700 MHz and 1.8 GHz and provide a 2×300 MHz band. The MeerKAT Precursor Array (MPA — also known as KAT-7) will provide a 2×256 MHz band within the frequency range from 1.2 to 1.95 GHz. Benchmarks demonstrate that DSPSR can comfortably process these signals in real time using only 4 workstations, each equipped with a single GPU. Considering its exemplary performance, extensive functionality, and degree of maturity, both scientists and engineers are encouraged to take advantage of this well-tested open-source digital signal processing library for radio astronomy.

Acknowledgments

Throughout the development of this software, we have benefited from the advice and input of many colleagues, especially Dan Stinebring, Josh Kempner, Matthew Britton, Stuart Anderson, Russell Edwards, Haydon Knight, Ben Stappers, Ben Barsdell, and Jonathon Kocz. We are grateful to those who have implemented support for the interpretation of data from various instruments, including Paul Demorest, Aidan Hotan, Mike Keith, Jonathan Khoo, Karuppusamy Ramesh, Jayanta Roy, Kevin Stovall, and Craig West. Tim Hankins and Ingrid Stairs provided helpful comments on the text. We thank Xenon Systems and the Victorian Partnership for Advanced Computing (VPAC) for provision of the NVIDIA® Fermi GPU and assistance with its use. We acknowledge Intel® for support of our Square Kilometre Array research programme. The Parkes Observatory is part of the Australia Telescope which is funded by the

[11] `http://www.vlbi.org/vsi`.

[12] `http://www.khronos.org/opencl`.

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

Commonwealth of Australia for operation as a National Facility managed by CSIRO.

A Floating Point Operations

The number of floating point operations required to create a synthetic filterbank and perform phase-coherent dispersion removal is a function of the number of frequency channels produced by the synthetic filterbank, N_c, and the size of the dispersion response function in the output frequency channel with the lowest centre frequency, N'. This is true for both real-valued and complex-valued input data. Although twice as many real-valued input time samples are required to obtain the same resolution in the frequency domain, the $2N$-point real-to-complex Fast Fourier Transform (FFT) is readily computed with roughly the same efficiency as the N-point complex-to-complex FFT, as described in Section 12.3 of Numerical Recipes (Press et al. 1992).

In the case of the deprecated filterbank, the forward N_c-point FFT is repeated N' times, requiring

$$N' \times 5N_c \log_2 N_c$$

operations. Convolution is then performed independently on each of the N_c channels using a forward and backward N'-point FFT, costing

$$2N_c \times 5N' \log_2 N'$$

operations. In the case of the convolving filterbank, the single N_cN' forward FFT requires

$$5N_cN' \log_2 N_cN'$$

operations. For each of the N_c channels, a backward N'-point FFT is performed, costing

$$N_c \times 5N' \log_2 N'$$

operations. Although both methods require a total of

$$5N_cN'(\log_2 N_c + 2\log_2 N')$$

floating point operations, the speed with which those operations are performed will depend on the implementation details of the FFT and the architecture of the computing device. As shown in Figure 5, FFT libraries typically achieve greater efficiency as the transform size is increased until the problem no longer fits in the L1 cache. Therefore, if N_c is smaller than the L1 cache–limited transform length, then the convolving filterbank may provide better performance than the deprecated method. However, if the product of $N_c \times N'$ is large, such that the memory required to compute the FFT is larger than the L2 cache size, then the deprecated method may provide better performance. Regardless of computational performance considerations, in light of the spectral leakage artifacts introduced by the deprecated filterbank method (see Figure 4), use of the convolving filterbank is recommended.

References

Backer, D. C., Dexter, M. R., Zepka, A., Ng, D., Wertheimer, D. J., Ray, P. S. & Foster, R. S., 1997, PASP, 109, 61
Backer, D. C., Hama, S., Van Hook, S. & Foster, R. S., 1993, ApJ, 404, 636
Bailes, M., 2003, in Radio Pulsars, Eds. Bailes, M., Nice, D. J. & Thorsett, S. (San Francisco: Astronomical Society of the Pacific), 57–64
Bhat, N. D. R., Tingay, S. J. & Knight, H. S., 2008, ApJ, 676, 1200
Bhat, N. D. R., Wayth, R. B., Knight, H. S., Bowman, J. D., Oberoi, D., Barnes, D. G., Briggs, F. H., Cappallo, R. J., Herne, D., Kocz, J., Lonsdale, C. J., Lynch, M. J., Stansby, B., Stevens, J., Torr, G., Webster, R. L. & Wyithe, J. S. B., 2007, ApJ, 665, 618
Born, M. & Wolf, E., 1980, Principles of optics: electromagnetic theory of propagation, interference and diffraction of light (New York: Pergamon)
Bracewell, R., 1986, The Fourier Transform and its Applications, second, revised edition (New York: McGraw-Hill)
Coles, W. A., Rickett, B. J., Gao, J. J., Hobbs, G. & Verbiest, J. P. W., 2010, ApJ, 717, 1206
Demorest, P., Ramachandran, R., Backer, D., Ferdman, R., Stairs, I. & Nice, D., 2004, in Bulletin of the AAS, Vol. 36, 1598
Demorest, P. B., 2007, PhD thesis, University of California, Berkeley
Demorest, P., 2011, in preparation
Demorest, P., Ransom, S., DuPlain, R., Ford, J., McCullough, R., Ray, J. & Brandt, P., 2011, in preparation
Edwards, R. T. & Stappers, B. W., 2004, A&A, 421, 681
Edwards, R. T., van Straten, W. & Bailes, M., 2001, ApJ, 560, 365
Frigo, M. & Johnson, S. G., 2005, Proceedings of the IEEE, 93, 216
Han, J. L., Manchester, R. N., Lyne, A. G., Qiao, G. J. & van Straten, W., 2006, ApJ, 642, 868
Hankins, T. H., 1971, ApJ, 169, 487
Hankins, T. H. & Rickett, B. J., 1975, in Methods in Computational Physics Volume 14 — Radio Astronomy (New York: Academic Press), 55–129
Hobbs, G. B., Edwards, R. T. & Manchester, R. N., 2006, MNRAS, 369, 655
Hotan, A. W., 2006, PhD thesis, Swinburne University of Technology
Hotan, A. W., Bailes, M. & Ord, S. M., 2004a, MNRAS, 355, 941
Hotan, A. W., van Straten, W. & Manchester, R. N., 2004b, PASA, 21, 302
Jacoby, B. A., Hotan, A., Bailes, M., Ord, S. & Kuklarni, S. R., 2005, ApJ, 629, L113
Jenet, F. A. & Anderson, S. B., 1998, PASP, 110, 1467
Jenet, F. A., Cook, W. R., Prince, T. A. & Unwin, S. C., 1997, PASP, 109, 707
Johnston, S., Hobbs, G., Vigeland, S., Kramer, M., Weisberg, J. M. & Lyne, A. G., 2005, MNRAS, 364, 1397
Johnston, S., Taylor, R., Bailes, M., Bartel, N., Baugh, C., Bietenholz, M., Blake, C., Braun, R., Brown, J., Chatterjee, S., Darling, J., Deller, A., Dodson, R., Edwards, P., Ekers, R., Ellingsen, S., Feain, I., Gaensler, B., Haverkorn, M., Hobbs, G., Hopkins, A., Jackson, C., James, C., Joncas, G., Kaspi, V., Kilborn, V., Koribalski, B., Kothes, R., Landecker, T., Lenc, A., Lovell, J., Macquart, J.-P., Manchester, R., Matthews, D., McClure-Griffiths, N., Norris, R., Pen, U.-L., Phillips, C., Power, C., Protheroe, R., Sadler, E., Schmidt, B., Stairs, I., Staveley-Smith, L., Stil, J., Tingay, S., Tzioumis, A., Walker, M., Wall, J. & Wolleben, M., 2008, Experimental Astronomy, 22, 151
Johnston, S., van Straten, W., Kramer, M. & Bailes, M., 2001, ApJ, 549, L101
Jonas, J. L., 2009, IEEE Proceedings, 97, 1522

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press

Karastergiou, A., Hotan, A. W., van Straten, W., McLaughlin, M. A. & Ord, S. M., 2009, MNRAS, 396, L95
Karuppusamy, R., Stappers, B. & van Straten, W., 2008, PASP, 120, 191
Karuppusamy, R., Stappers, B. W. & van Straten, W., 2010, A&A, 515, A36+
Knight, H. S., Bailes, M., Manchester, R. N., Ord, S. M. & Jacoby, B. A., 2006, ApJ, 640, 941
Manchester, R. N. & Taylor, J. H., 1972, Astrophys. Lett., 10, 67
McKinnon, M. & Stinebring, D., 1998, ApJ, 502, 883
Ord, S. M., Jacoby, B. A., Hotan, A. W. & Bailes, M., 2006, MNRAS, 371, 337
Ord, S. M., van Straten, W., Hotan, A. W. & Bailes, M., 2004, MNRAS, 352, 804
Press, W. H., Teukolsky, S. A., Vetterling, W. T. & Flannery, B. P., 1992, Numerical Recipes: The Art of Scientific Computing, 2nd edition (Cambridge: Cambridge University Press)
Roy, J., Gupta, Y., Pen, U., Peterson, J. B., Kudale, S. & Kodilkar, J., 2010, Experimental Astronomy, 28, 25
Shrauner, J. A., 1997, PhD thesis, Princeton University
Stairs, I. H., 1998, PhD thesis, Princeton University
Stairs, I. H., Splaver, E. M., Thorsett, S. E., Nice, D. J. & Taylor, J. H., 2000, MNRAS, 314, 459
Staveley-Smith, L., Kim, S., Calabretta, M. R., Haynes, R. F. & Kesteven, M. J., 2003, MNRAS, 339, 87
Taylor, J. H. & Weisberg, J. M., 1989, ApJ, 345, 434
van Straten, W., 2002, ApJ, 568, 436
van Straten, W., 2003, PhD thesis, Swinburne University of Technology
van Straten, W., 2006, ApJ, 642, 1004
van Straten, W., 2009, ApJ, 694, 1413
van Straten, W., Bailes, M., Britton, M., Kulkarni, S. R., Anderson, S. B., Manchester, R. N. & Sarkissian, J., 2001, Nature, 412, 158
van Straten, W., Britton, M. & Bailes, M., 2000, in Pulsar Astronomy – 2000 and Beyond, IAU Colloquium 177, Ed. M. Kramer, N. Wex & R. Wielebinski (San Francisco: Astronomical Society of the Pacific), 283–284
Verbiest, J. P. W., Bailes, M., Coles, W. A., Hobbs, G. B., van Straten, W., Champion, D. J., Jenet, F. A., Manchester, R. N., Bhat, N. D. R., Sarkissian, J. M., Yardley, D., Burke-Spolaor, S., Hotan, A. W. & You, X. P., 2009, MNRAS, 400, 951
Verbiest, J. P. W., Bailes, M., van Straten, W., Hobbs, G. B., Edwards, R. T., Manchester, R. N., Bhat, N. D. R., Sarkissian, J. M., Jacoby, B. A. & Kulkarni, S. R., 2008, ApJ, 679, 675
Walker, M. A., Koopmans, L. V. E., Stinebring, D. R. & van Straten, W., 2008, MNRAS, 388, 1214
Wietfeldt, R., Straten, W. V., Rizzo, D. D., Bartel, N., Cannon, W. & Novikov, A., 1998, A&AS, 131, 549
Yardley, D. R. B., Hobbs, G. B., Jenet, F. A., Verbiest, J. P. W., Wen, Z. L., Manchester, R. N., Coles, W. A., van Straten, W., Bailes, M., Bhat, N. D. R., Burke-Spolaor, S., Champion, D. J., Hotan, A. W. & Sarkissian, J. M., 2010, MNRAS, 407, 669
You, X. P., Hobbs, G., Coles, W. A., Manchester, R. N., Edwards, R., Bailes, M., Sarkissian, J., Verbiest, J. P. W., van Straten, W., Hotan, A., Ord, S., Jenet, F., Bhat, N. D. R. & Teoh, A., 2007, MNRAS, 378, 493

https://doi.org/10.1071/AS10021 Published online by Cambridge University Press