Pulsar data analysis with PSRCHIVE

IPTA Student Week 2024

This is largely based off the 2018 tutorial by Ryan Shannon and Aditya Parthasarathy, and George Hobbs' 2023 tutorial.

In this tutorial, we will work through the steps for processing a raw pulsar profile into high-level data products: calibrated profile products, and times-of-arrival. We will be using PSRCHIVE, a ubiquitous pulsar data analysis software toolkit. We will learn how to view, edit, and pre-process pulsar data for timing and profile analysis.

PSRCHIVE is an open-source, object oriented data analysis software that implements a wide range of data analysis algorithms for use in data calibration, statistical analysis and visualization. Please refer to van Straten et al. (2012) or visit http://psrchive.sourceforge.net/ for a comprehensive description of the PSRCHIVE software suite.

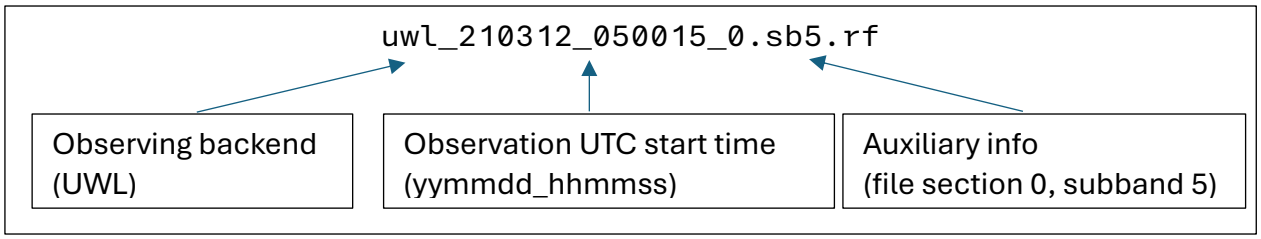
Note: all code/filenames/commands will be formatted with `monospace` font.

Introduction

The data we will be working with are referred to as **archive** data. These files contain astronomical data recorded during a pulsar observation and are typical stored as a three-dimensional array of pulse profiles, the axes being time (sub-integration), frequency (channel) and polarization. The data have attributes called metadata that describe the various physical attributes of the observation.
A useful point to note here is that, many of the PSRCHIVE programs print out a brief help message when the -h command line option is used, for e.g. `psrstat -h`. Descriptive usage instructions are also available online.

Viewing and evaluating pulsar metadata

We will be working with some recent observations of the PPTA's favourite pulsar, PSR J0437-4715. These will be contained in data/. To start with, the file we will be working with is `uwl_210312_050015_0.sb5.rf`
The naming convention is simple:



This already tells us quite a bit of useful information! These are Parkes observations with the Ultra-Wideband Low receiver, which spans 704-4032 MHz. However, to keep things easy to work with in this tutorial, we have cut out a small sub-band (instrumental sub-band 5) and averaged the data down to keep it light-weight. Let's continue to dig deeper to investigate the data.

There are a number of ways to quickly investigate file metadata – these include `vap` and `psredit`. `psredit` can also be used to edit archive metadata, which can be useful and/or dangerous!
Firstly, let's investigate what each of these tasks do using `vap -h` and `psredit -h`

We can also check out the online documentation
http://psrchive.sourceforge.net/manuals/psredit/
http://psrchive.sourceforge.net/manuals/vap/

1. Run `psredit uwl_210312_050015_0.sb5.rf`.

   What are the dimensions of the data?
   How many polarizations, frequency channels, and sub-integrations are there?
   What is the central frequency, bandwidth, and observation length?
   What telescope and receiver was used for this observation?
   Are these the same for all files in the folder?

2. Use `psredit -c` or `vap -c` to grab the metadata in the previous question above. Is there any difference between `psredit` and `vap` in this use case?

3. Use wildcards (e.g. `*.rf`) with parameter selections to investigate the metadata across several files. Which parameters change, and which stay the same?

Now let's use `psrstat` to investigate the data further.
http://psrchive.sourceforge.net/manuals/psrstat/

4. Try running `psrstat uwl_210312_050015_0.sb5.rf`
   What difference is there between the output of `psrstat` and `psredit`?

We'll now look at this signal-to-noise distribution of the observations – in a similar way to `psredit/vap`, we can select `psrstat` parameters using the -c flag

5. In the data directory, run
   `psrstat -c snr *.rf`

This will print the S/N value for each archive file.
Note, however, that by default, many psrchive routines (such as psrstat) will only operate on the first sub-integration, first channel, and first polarisation. Beware - this is a common pitfall for new users!

There are multiple ways we can learn details about the entirety of the data in one archive file as a whole. The first is to use *looping* functionality (`-l`), which will execute the desired operation in a loop over a specified dimension. In this case, we can use the loop functionality to print the S/N of an archive file as a function of frequency channel.

6. In the data directory, run:
   ```
   psrstat -Q -l chan=-0-nchan -c snr
   uwl_210312_050015_0.sb5.rf
   ```

   a. What is the purpose of the -Q option?

   b. Redirect the psrstat output to a file by adding `> snr_out.dat` to the end of the command above. Using python, gnuplot, or your favourite plotting library, what does the S/N distribution look like across channels?

   c. Can you also print the S/N for polarisation 4 across all channels? (hint: can you use multiple loops in one psrstat command?)

      How about the S/N for channel 10 across all 4 polarisations?

Using `pam` and `psrsh` commands to pre-process data

In the last section, we ran into the feature/pitfall that psrstat only operates on the first channel, subint, and polarisation of the archive file. This is useful if we want to investigate properties across dimensions in the file, but sometimes we just want to know about the integrated data product.
We can use pre-processing commands to integrate over various dimensions. This can be done using `pam`, or `psrsh`.

`pam` is a program for manipulating archive files in various ways, writing out new files or modifying existing files.
https://psrchive.sourceforge.net/manuals/pam/
Use `pam -h` to see a detailed list of options.

`psrsh` is a powerful, flexible command language interface that provides access to psrchive data processing algorithms.
https://psrchive.sourceforge.net/manuals/psrsh/
See `psrsh -h` for basis usage and `psrsh -H` to see a list of commands – the first column gives the full command name, the second gives a shorthand, and the third gives a short description

Entering the interactive `psrsh` shell, you can also obtain further information on each of the commands e.g.
`psrsh> help delete`
and even further layers of help are possible with options for each command

Along with much greater functionality available with `psrsh`, it also allows you to perform tasks *on the fly* before executing a certain command, whereas pam will simply modify the archive file as requested and save the result to disk.

7. Use the `psrsh` pre-processing flag in `psrstat` (-j) to "frequency-scrunch" the `uwl_210312_050015_0.sb5.rf` archive data before computing the S/N, for all four polarisations. What are the S/N

8. Do the same, but first using `pam -F` to write out a new file, before running `psrstat`:
```
pam -F -e rf.F uwl_210312_050015_0.sb5.rf
psrstat -Q -l pol=0- -c snr uwl_210312_050015_0.sb5.rf.F
```

Note that we can now use pam or psrsh on-the-fly processing to determine the frequency, time, and polarization-averaged S/N for all files:

9. Run
   `psrstat -Q -j FTp -c snr *.rf > snr.dat`
   This will calculate the average S/N for each archive file and redirect the output to a file snr.dat

   Inspect the S/N values manually and by plotting their distribution.
   How does it compare to the previous distribution?
   Is the S/N generally higher before or after integration?

Visualising archive data with `pav` and `psrplot`

The utility of visualising your data in different ways cannot be understated. This can help to diagnose problems, and quickly gain insights into your observations.
PSRCHIVE offers multiple ways to view data, which include `pav` and `psrplot`.
https://psrchive.sourceforge.net/manuals/pav/
https://psrchive.sourceforge.net/manuals/psrplot/
As always, we can use
`pav -h`
and
`psrplot -h`
to find usage and descriptive help on these commands.

Also, note that further documentation is available through `psrplot -P` (which lists the available plot types)

`psrplot` offers more flexibility than pav, while `pav` may be useful for generating "standard" plots quickly. Like `psrstat`, `pav` and `psrplot` will only work on the first channel, sub-integration, and polarisation, unless told to do something different or preprocessing has been applied.

We can see this if we plot a single profile using `pav -D`:

10. Run `pav -D uwl_210312_050015_0.sb5.rf`

    Does the S/N look high? Consider that this is a reasonably long observation of the brightest MSP in the sky.

11. Now run `pav -DFp uwl_210312_050015_0.sb5.rf`

    How does this compare with the previous plot? What have the additional option flags "Fp" done to the data?

Next, let's plot the pulse intensity against pulse phase and frequency:

12. Run `pav -G uwl_210312_050015_0.sb5.rf`

    What do you notice about the pulse over frequency?

Which pre-processing flag should we use to correct for this effect? (look into the help menu)

Is there anything unusual-looking at the top and bottom of the frequency range? What is going on? How could we fix this?

13. Use `pav -SF uwl_210312_050015_0.sb5.rf` to produce a frequency-integrated plot of the pulse profiles for Stokes I (total intensity, red), linearly polarised intensity (red), and circularly polarized intensity (blue) with the linear polarisation position angle above. What do you notice about the relative intensities of the different polarisations?

14. Now reproduce the previous pav plots using psrplot .
    Hint: `psrplot -p flux uwl_210312_050015_0.sb5.rf` will produce a single plot of flux density for this archive.

15. Use `psrplot` to plot the frequency-averaged profile for all Stokes parameters. Hint: use `psrplot -p s` with appropriate `psrsh` pre-processing

16. Use `psrplot` to plot the pulse intensity vs frequency and phase for all four Stokes parameters. Hint: use the loop functionality to loop over polarisations (`-l pol=0-3`), and use the pre-processing command `-j "state Stokes"` to ensure the archive is converted from an instrumental polarisation basis (AA, BB, Re(AB), Im(AB)) (state "Coherence") to the Stokes basis (I, Q, U, V)
    How do the different polarisations look?

    Do you notice any difference if the instrumental polarisation basis is used (use `-j "state Coherence"`)

We've now explored various plotting routines with `pav` and `psrplot`. There is much more functionality to explore than what we have covered here, so feel free experiment!

Calibrating archives using `pac`

Let's now calibrate our data. The archives we have been investigating so far have been raw, uncalibrated files. You may have noticed, for example, the shape of the instrumental bandpass is visible when plotting pulse intensity vs phase and frequency.
The purpose of calibration is to remove all effects imparted by the telescope signal chain, and to obtain measurements as close as possible to the real physical signal from the source.

The primary task to perform calibration with PSRCHIVE is `pac` .
https://psrchive.sourceforge.net/manuals/pac/

A simple calibration is usually performed in two stages:

- Create a database of calibrators
- Perform the calibration

The calibrators directory contains a set of clean observations of a switched, polarised noise diode injected into the telescope signal chain while pointing slightly away from the target pulsar. These are used to calibrate the gain and polarisation response of the telescope. It also contains a flux density calibrator file "`newwifi.fluxcal`" which is derived from an observation of a known bright quasar (PKS B1934-638) to correct the absolute flux density of the archives.

17. Create a calibrator database - inside the "calibrators" directory, issue:

```
pac -w -u cf -u fluxcal -k calib_db
```

Inspect the output using `more calib_db`

The above command produces a database file named `calib_db`, specified using the `-k` option. The `-w` option enables creation a new database. The `-u` option specifies the extension of the calibration archive files.

Now that we have created the calibration database, we can calibrate our files:

18. From inside the "calibrators" directory, issue:

```
mkdir ../data/calibrated
pac -d calib_db -T -O ../data/calibrated ../data/*.rf
```

This uses the database (`calib_db`) to calibrate profiles (with `.calib` extension) that are stored in the calibrated directory.

19. Using `psrplot` or `pav`, what differences do you notice between calibrated and uncalibrated files? How do the profiles for each Stokes parameter differ? Hint: You can plot 2 files in the same PGPLOT window using the -N option in `psrplot` or `pav`

Generating arrival time estimates using `psradd`, `psrsmooth`, `pat`

Now that we have calibrated our files, we will proceed with generating an initial set of times of arrival.

The first step in this quest will be to prepare our data for timing, to construct a standard template, and then generate the times of arrival themselves.

Typically, timing is performed on polarisation-integrated, time- and frequency-integrated archives (although in the real world you may like to experiment with varying degrees of time- frequency-averaging - why might this be useful?)

Let's first prepare our calibrated data using pam to frequency and time-scrunch the data. We will also "install" a timing ephemeris using the -E option and the par-file in the `ephemerides` folder.

20. From the "`calibrated`" directory, run
    `pam -E ../../ephemerides/J0437-4715.par -FTp -e FTp *.calib`

the -E option specifies the ephemeris to install, and the -FTp will frequency, time, and polarisation-scrunch the data. This new ephemeris will allow us to refine the profile alignment across epochs using an updated timing model, improving signal-to-noise in the inter-epoch average.

Standard templates are typically formed using long-term average profiles to maximise the signal-to-noise and to ensure any epoch-to-epoch variations (e.g. induced by scintillation) are "smoothed out").

We can now use `psradd` to concatenate all archives.
https://psrchive.sourceforge.net/manuals/psradd/

21. In the "`calibrated`" directory, run:
    `psradd -o J0437-4715.add uwl_210*.FTp`
    then
    `pam -T -e add.T J0437-4715.add`

    Check the un-averaged added file by running
    `psrplot -p Y -D /xs J0437-4715.add`
    Double-check that the profiles reasonably phase-aligned.

Then, plot the resulting averaged, added file using
`psrplot -p D -j FT -j p -D /xs J0437-4715.add.T.`
What is the resulting S/N? How does the average profile look compared to single observations?

We will now generate a smooth, noise-free template using `psrsmooth` –
Check `psrsmooth -h` for documentation

22. Run `psrsmooth -W J0437-4715.add.T`

This generates a smooth, noise-free template which can be used to optimally derive times of arrival.

23. Inspect the noise-free template using `psrplot -p flux J0437-4715.add.T`
    How does it compare to the averaged, added profile?

We have now set up all ingredients we need to generate times of arrival. This task can be done using `pat`
https://psrchive.sourceforge.net/manuals/pat/
`pat -h`

This task provides a number of different methods for time of arrival generation, which you are free to explore. Today we will be using the Fourier-domain Monte Carlo method (`-A FDM`), which is based off the Fourier Phase Gradient algorithm (based off the Fourier Shift theorem – see Taylor 1992) with a Monte-Carlo approach to estimating time of arrival uncertainties.

24. Run
    `pat -A FDM -s J0437-4715.add.T.sm -f 'tempo2 IPTA' *.FTp > J0437-4715.tim`
    This will generate a tim-file containing times of arrival for all observations using the smooth standard template we just created, and the Fourier domain Monte Carlo algorithm. The flag `-f 'tempo2 IPTA'` outputs many useful ToA flags.
    Inspect the tim-file using `more` or another text editor

An abridged introduction to TEMPO2

Now we have a parfile and timfile, we can do timing! This will be covered in much more detail in this workshop, but for now we can have a quick play-around.

We'll use TEMPO2, a mature pulsar timing software package.

25. Run
    `tempo2 -gr plk -f ../../ephemerides/J0437-4715.par J0437-4715.tim`

    Do the residuals look any good (i.e. tightly bound around 0, scatter on same scale as errorbars?)
    Try fitting for DM, then F0 and F1. Do things improve? What sort of things could be causing the scatter?
    Output a new parfile.

Let's now investigate the effect that calibration had on our timing.

26. From the `data` directory (containing the raw .sb5.rf archive files), run:
    ```
    cp calibrated/J0437-4715.tim .
    pat -A FDM -s calibrated/J0437-4715.add.T.sm -f 'tempo2' -j FTp *.rf | sed 's/$/ -f uncal/' | grep -v FORMAT >> ./J0437-4715.tim
    ```

    This should produce a tim file with ToAs both from the uncalibrated data (with a flag "uncal") and calibrated data. Now run tempo2 again with the new parfile from earlier.
    `tempo2 -gr plk -f calibrated/new.par J0437-4715.tim`
    Now highlight the uncalibrated points by hitting `ctrl-i` in the `plk` window then typing `-f` in the terminal command interface.

    What do you notice about the uncalibrated ToAs compared with the calibrated ones?

Go forth and produce beautiful archives!