IPTA student workshop 2017: Day 1, preparing timing data with PSRCHIVE

June 26, 2017

Kuo Liu[1], Stefan Osłowski[2]

[1]Max-Planck-Institut für Radioastronomie, Auf dem Hügel 69, D-53121 Bonn, Germany
[2]Centre for Astrophysics & Supercomputing, Swinburne University of Technology, PO Box 218, Hawthorn VIC 3122, Australia

1 Introduction

Welcome to the IPTA 2017 student workshop!

In the first tutorial of the week, we will learn how to pre-process pulsar data for timing analysis. The data we are going to tackle come directly from pulsar backend during the pulsar observations, and are commonly referred to as **archive data (*.ar files)**. Archive files contain astronomical data recorded during the observation, and information of the observation and data recording saved as metadata.

Data provided for this tutorial are stored under the route:

'/home/ipta2017/Day01/PulsarTiming_I/'.

There are three sub-directories under this route. The **'ar/'** directory contains archive files produced during pulsar observation, each of which consists of **integrated pulse profile** data formed by folding/integrating a given length of observation, and their corresponding metadata. These data need to be properly reduced to obtain the equivalent **pulse time-of-arrival (TOA)** of the integrated pulse profile, which will be used in the timing analysis afterwards. In the **'cal/'** directory you will find calibration archive data which are recorded during observation of an artificial signal called **noise diode**. These data are used to calibrate the polarisations of the pulsar data. The **'std/'** directory contains an archive file of a high signal-to-noise (S/N) integrated pulse profile obtained after appropriate data reduction. This pulse profile shall later be used as the template profile to calculate TOAs. In the route directory there is ephemeris file (the J1022+1001.par file) which will be later used to quickly check the TOAs you have.

We will be using the **PSRCHIVE** software package to reduce the archive data. PSRCHIVE is a suit of software tools commonly used within the pulsar community to reduce pulsar data.

It has been installed on the virtual machine and is ready to use. The **usage** of routines that are going to be used for this tutorial is described in the **Appendix**. Please refer to the Appendix when you go through the following exercises. For more information of PSRCHIVE, see `http://psrchive.sourceforge.net/` and `https://arxiv.org/abs/1205.6276`.

Intermediate-stage data will be generated during the following exercises, but will be no more than a few hundred mega-bytes. Please organise yourself on where to place the generated data and the results!

2 Exercise 0: Familiarising yourself with the data

Before data reduction, it is necessary to first get familiar with the data available. The archive files contain metadata of relevant information of the observation and data recording, such as the source name, source position in right ascension and declination, MJD, central frequency, bandwidth, etc. This information can be viewed and edited (if necessary) with the PSREDIT routine.

Activities

1. Use PSREDIT to browse the metadata of a few selected (as you wish) pulsar archive files and get information of a). Source name; b). Number of phase bins; c). Number of frequency channels; d). Number of sub-integrations; e). Number of polarisations; f). Integration length of each archive file; g). Length of the entire observation.

2. Browse the metadata of the calibration archive file. Compare with the metadata from those in the pulsar archive files and note down the differences;

Answer

1. a). Number of phase bins: 1024; b). Number of frequency channels: 128; c). Number of sub-integrations: 1; d). Number of polarisations: 4; e). Length of integration: 10.0 s; f). Length of the entire observation: 1280 s.

2. A few notable differences: Observation type, Observation mode, dispersion measure, start time, time per bin or sample, etc.

3 Exercise 1: Visualising the data

The archive data can be visualized with the PSRPLOT routine. An initial check of the data quality is always helpful for designing the procedure of data reduction.

Activities

1. Use PSRADD to combine all pulsar archive files together into a single archive file.
2. Use PSRPLOT to plot a polarisation profile of both the combined archive file and the given template profile in 'std/', and describe the difference between them.
3. Use PSRPLOT to make a pulse phase vs frequency image of the combined archive file. What can be seen?
4. Use PSRPLOT to make a pulse phase vs time image of the combined archive file. What can be seen?
5. The calibration signal is a square wave signal intrinsically with 100% linear polarisation. Use PSRPLOT to make a polarisation profile of the calibration archive file. What do you see?

Answer

1. Simply run e.g.,:

   ```
   psradd 2*.ar -o yours_total.ar
   ```

2. The total intensity (I) profile of the combined data is in general consistent with the post-reduction profile, but the linear and circular components are largely different.
3. In the pulse phase vs frequency image of the combined data, there are channels with significantly stronger and different signals from the pulse profile. These are narrow-band radio interference (RFI). If you have not included de-dispersion in preprocessing commands, the pulsar signals from different frequency channels are not aligned due to dispersion delay. In this case, adding 'D' to the preprocessing commands will align the signals.
4. In the pulse phase vs time image of the combined data, there are sub-integrations with significantly stronger and different signals from the pulse profile. These are impulse time-domain RFI.
5. The calibration signal has total intensity as a square wave, but the linear component is not 100% and there is circular component. Comparison of the intrinsic and observed signal of the calibrator gives the measurement of polarisation calibration.

4 Exercise 2: Polarisation calibration

The calibration data can be used to measure the receiver response and correct the polarisations of the pulsar data.

Activities

1. Use PAC to create a database file of the available calibration data.
2. Use PAC to calibrate all the individual pulsar archive files.
3. Use PSRADD to combined the calibrated pulsar data into a single archive file, and compare the polarisations profiles with what you see from the template profile.

Answer

1. You can use options '-w', '-p', '-u', and '-k' (some others may work as well).
2. You can use options '-d', '-O' (some others may work as well).
3. The linear and circular polarisation should now be consistent with seen in the given template profile.

5 Exercise 3: RFI mitigation

As we have noticed from the previous exercise, there is significant contamination by RFI in the pulsar data which has to be removed to ensure the quality of the data. This can be done in both a manual fashion and an automatic fashion. You can choose either of the following two options, or try out both and compare the results. Normally, manual RFI removal / zapping (option 1) tends to be more straightforward.

Activities

Option 1: Manual cleaning with pazi.

1. Use PAZI to load the single archive file you have formed after polarisation calibration. In the pulse phase vs time panel, zap sub-integrations with significant RFI.
2. Switch to pulse phase vs frequency image and zap channels with significant RFI.
3. Repeat the above for a few iterations until no significant RFI remains in either domain, and save the result.

Option 2: Automatic cleaning with paz.

1. Create a preprocessing command file to include usage of both the MSD and PLM method for the cleaning.
2. Use PAZ to load the command file and deploy the cleaning on the single archive file you have formed after polarisation calibration, and save the cleaned version in a new file.
3. Use PSRPLOT to make a pulse phase vs frequency image and a pulse phase vs time image of the cleaned data, and compare with those before the cleaning.

Note: The default memory of the virtual machine is only 1 GB, and it would be a bit slow to run pazi. So increase the memory as you can do, to e.g., 2 GB or even 4 GB.

Answer

Option 1:

1. The sub-integrations with power largely exceeding the others or with signals not in the phase where pulsar signal should be, have RFIs needed to be cleaned.
2. The frequency channels with power largely exceeding the others or with signals not in the phase where pulsar signal should be, have RFIs needed to be cleaned.
3. All RFIs should be cleanable after zapping some sub-integrations and frequency channels.

Option 2:

1. Create a preprocessing command file having the following content:

```
#! /usr/bin/env psrsh
zap median window=24

zap median cutoff=3

zap median exp={$all:max-$all:min}

zap median

zap mow robust

zap mow window=0.1

zap mow cutoff=4.0

zap mow
```

2. Use PAZ as e.g. the following:

```
paz -J yours_pre_cmd_file -e your_ext yours.ar
```

3. In both pulse phase vs frequency and pulse phase vs time image, the cleaned version should have most RFI removed.

6 Exercise 4: Calculating TOAs

As the data have now been calibrated for polarisation and been cleaned up for RFI, we can move on to the calculation of TOAs. The TOA is calculated by cross-correlating the observed integrated pulse profile with an independent template profile and finding the phase offset. The template profile needs to have high S/N in order to be a good reference for the TOA.

Activities

1. Use PSRSMOOTH to remove the noise from the template profile in 'std/'.
2. Use PAT to calculate TOAs from the single archive file that you have already calibrated and cleaned for RFI. Integrate the data in frequency and use the smoothed version of the template profile. Save the TOAs in a new file (e.g., named yours.tim).
3. Display the timing residuals of the TOAs with TEMPO2, by running:

   ```
   tempo2 -f J1022+1001.par yours.tim -gr plk -nofit
   ```

   This allows us to compare our derived TOAs to the prediction from the timing model. Have a look to see if the data points are in general random-distributed within the tolerance of their error bars. It will also produce output in the terminal window. Find the reduced chi-square value to see if it is close to 1.

Answer

1. Run PSRSMOOTH as

   ```
   psrsmooth -W std/J1022+1001.std
   ```

2. Run PAT as e.g.:

   ```
   pat -F -s std/J1022+1001.std.sm -f tempo2 yours.ar > yours.tim
   ```

3. The timing residuals in the plot should look in general randomly distributed within the scale of their error bars, i.e., no structure or trend. The reduced chi-square should be close to 1 (e.g., 1.1-1.2).

7 Exercise 5 (optional): Automating the above steps

Once each stage of data reduction has been familiarised and validated, we can then gather the command lines used to form a shell script that can go through the entire procedure automatically. The pipeline can be later used to process new data in a straightforward and consistent way.

Activities

1. Use the routines in the previous exercises to build a pipeline including polarisation calibration and RFI mitigation, to automatically reduce the pulsar data and calculate one TOA.

Answer

1. An example skeleton of the pipeline is given below:

```
#! /bin/bash

# Create database for polarisation calibration
pac -wp cal/ -u it -k database.txt

# Calibrate pulsar data
pac -d database.txt -O . ar/*.ar

# RFI mitigation
paz -J pre.psh -e zz *.calibP

# Integration
psradd -T *.zz -o all.ar

# Make smoothed template
psrsmooth -W std/J1022+1001.std

# Calculate TOA
pat -F -s std/J1022+1001.std.sm all.ar -f tempo2
```

8 Exercise 6 (optional): Automating processing with psrsh

The purpose of this exercise is to introduce the basic concept of using PSRSH, if you want to learn some advanced skills in PSRCHIVE.

PSRSH is a command language interpreter that provides access to data processing algorithms in PSRCHIVE. It can be used either as an interactive shell or as a shell script command processor. For a detailed description of its usage, see: `http://psrchive.sourceforge.net/manuals/psrsh/`.

It is possible to reproduce part of the pipeline from Exercise 5 using only psrsh. The main advantage is that the implementation above will unload and reload data multiple times. We can take advantage of the data being already present in the memory and apply multiple processing steps in one go which will speed the pipeline up.

Activities

1. Attempt to rewrite parts of the pipeline you have from Exercise 5 in PSRSH. As the first step, implement polarisation calibration and RFI removal in PSRSH.

2. Can you think of a way to also reproduce the PSRADD step with PSRSH?

Answer

1. Below is what could go into your PSRSH script:

```
# load the calibration database. To save time, only do it once:
init cal load database.txt
init cal type ovhb04
# calibrate the archive
Cal
# unload calibrated data
unload ext=calibP
# applying zapping
# paste contents of pre.psh except the shebang line (the first line)
# unload
unload ext=zz
```

2. (Try to think about it yourself......or ask one of the tutors if you want to embarrass him!)

A Guideline of PSRChive

The usage of routines included in this tutorial is explained as below. You can obtain more details of PSRCHIVEroutines from van Straten, Paul Demorest & Stefan Osłowski (2012) (HTTPS://ARXIV.ORG/ABS/1205.6276), and HTTP://PSRCHIVE.SOURCEFORGE.NET/MANUALS/.

A.1 psredit

This routine is used to display information of the pulse profile data saved in the metadata. For a full list of options, run:

```
psredit -h
```

Simply run it with an archive file gives a full list of metadata information:

```
psredit yours.ar
```

The first, second, third column give the keyword, its physical meaning and its value. To display a certain keyword, simply do:

```
psredit -c keyword yours.ar
e.g.
psredit -c file yours.ar
psredit -c nbin,nchan yours.ar
```

You can use it edit the value of a certain keyword. To rewrite the same file, do:

```
psredit -m -c "keyword=new_value" yours.ar
e.g.
psredit -m -c "freq=1400.0" yours.ar
psredit -m -c "rcvr:basis=cir" yours.ar
```

Or you can write to a new file with a different extension, as:

```
psredit -e diff_ext -c "keyword=new_value" yours.ar
```

A.2 psrplot

This routine is used to plot the pulse profile data in a variety of ways. For a full list of options, run:

```
psrplot -h
```

Use '-D' option to choose device to display the plot. E.g., to display in another terminal window, use '-D /xs'. If not specified in the command line, the routine will ask you to specify one before making the plot.

The data may need preprocessing before making the plot. For this, use the '-j' option. Append any preprocessing commands afterwards, such as

```
psrplot -jFTp
```

for integration in frequency (F), time (T) and polarisation (p). For a complete list of supported preprocessing commands, type

```
psrsh -H.
```

To see a full list of available plot types, run:

```
psrplot -P
```

For example:

```
psrplot -jFT -pS yours.ar
psrplot -jTp -pG yours.ar
psrplot -jFp -pY yours.ar
```

The above commands plot polarisation profile (white I, red L, blue V), pulse phase vs frequency image of flux, and pulse phase vs time image of flux.

A.3 pazi

This routine allows visualization of the pulse profile, and removal of RFI at the same time. Run as:

```
pazi yours.ar
```

It generates a plot of the pulse profile, and a pulse phase vs time image in an interactive panel which you can operate on.

Press ’h’ on the panel for a full list of options.

Press ’f’ to switch to pulse phase vs frequency image.

Press ’t’ to switch back to pulse phase vs time image.

Left click twice to choose a range to zoom in.

Right click once to zap a selected sub-integration / frequency channel.

Left click + right click to zap a range of sub-integrations / frequency channels.

Press ’u’ to undo the last operation done.

Press ’s’ to save the zapped file. It will save the zapped version in a new file with an extra extension ’.pazi’, i.e., ’yours.ar.pazi’.

A.4 paz

This routine is used to zero weight given frequency channels & sub-integrations so as to mitigate RFI. For a full list of options, run:

```
paz -h
```

For example, to zero weight channels with indices ’a’, ’b’, ’c’, do:

```
paz -z’a b c’
```

To zero weight channels with indices from ’a’ to ’b’ (inclusive), do:

```
paz -Z’a b’
```

Use ’-m’ to rewrite the file or use ’-e’ to give the extension for the created new file.

There are a few automatic RFI detection and excursion methods installed. The Median S-moothed Difference (MSD) method builds a running-median smoothed version of the spectrum, and zap channels based on the tolerance to differences between the observed bandpass and the smoothed version. It is designed mainly to remove narrow-band RFI. The Profile Lawn Mowing (PLM) method builds a running-median smoothed version of the profile, and replaces phase bins with noise based on tolerance to differences between the observed pulse profile and the smoothed version. It is designed to remove RFI appearing as broad-band bursts of impulsive emission.

To deploy the methods, create a preprocessing command file (equivalently a psrsh script) and make use of the job preprocessor (’-J’ option), as

```
paz -J pre.psh -e your_ext yours.ar
```

For the MSD method, have the following in the command file:

```
#! /usr/bin/env psrsh

# set the median smoothing boxcar width to 24 channels
zap median window=24

# set the cutoff to 3.5 times the residual standard deviation
zap median cutoff=3.5

# set the expression evaluated in each frequency channel
zap median exp={$all:max-$all:min}

# apply the median smoothing filter
zap median
```

For the PLM method, use the following content:

```
#! /usr/bin/env psrsh

# use robust statistics (median absolute deviation)
zap mow robust

# set the median-smoothing window to 20\% of the pulse profile
zap mow window=0.2

# set the cutoff to 4.0 times the residual standard deviation
zap mow cutoff=4.0

# execute the zap mow algorithm
zap mow
```

The values of the parameters can be tuned to optimize the effect of RFI removal.

A.5 pac

This routine is used to calibrate the polarisations of the pulse profile data. For a full list of options, run:

```
pac -h
```

It normally runs in two tiers: create a database of calibrators and perform the calibration.

For database, use '-w' option to enable database generation, use '-p' option to specify the routine where the calibration data are, use '-u' option to specify the extension of calibration files, and use '-k' option to give the name of the output database file. E.g.:

```
pac -w -p cal_path -u cal_ext -k name_database
```

For calibration, use '-d' option to give the name of the database file, use '-e' option to specify the extension of the calibrated files (by default calibP), and use '-O' option to give the route for the output. E.g.:

```
pac -d name_database -O route_out yours.ar
```

A.6 psradd

This routine appends integrated pulse profiles in individual files together to a single file. For a full list of options, run:

```
psradd -h
```

For example, to simply append individual files together, run it as

```
psradd yours1.ar yours2.ar ... -o yours_total.ar
psradd *.ar ... -o yours_total.ar
```

The option '-o' specify the name of the added file. The resulting file will have number of sub-integrations the same as the number of files given. To integrate them all (so that the number of sub-integration is 1), add '-T' option.

A.7 psrstat

This routine calculates a wide variety of derived quantities of the pulse profile data. For a full list of options, run:

```
psrstat -h
```

Use '-c' option to print the value of given quantity, e.g.:

```
psrstat -c snr yours.ar
```

Run with a file without any command-line arguments to print a list of all available quantities:

```
psrstat yours.ar
```

A.8 psrsmooth

This routine is used to remove white noise in the profile data. For a full list of options, run:

```
psrsmooth -h
```

It is recommended to use the '-W' option to enable wavelet smoothing. E.g., run:

```
psrsmooth -W yours.ar
```

It will generate a smoothed version of the given archive file named 'yours.ar.sm'.

A.9 pat

This routine calculates TOA of the given pulse profile. For a full list of options, run:

```
pat -h
```

E.g., use option '-s' to give the template as the reference of the TOA, option '-F' to integrate the profile in frequency, option '-T' to integrate the profile in time, option '-A' to choose fitting algorithm and option '-f tempo2' to specify tempo2 format output:

```
pat -FT -A FDM -s yours.std -f 'tempo2' yours.ar
```

The tempo2 format output is as below:

```
<filename> <Frequency> <TOA> <TOA error> <Telescope ID>
```