Name: _____

Date: _____

Phys 194–FYRE Assignment #7
Generating TOAs & Using Timing Software (TEMPO2)

Assignment Policy: You can consult class notes, books, and online resources. You can work in small groups (2 or 3), but you must turn in your own work. Make sure you are clear about the process you use to solve the problems: partial credit will be awarded.

Getting Started

To access the raw data files today and in the future, we will log in remotely to *bowser*. To do so, we use a command like the following:

```
ssh -XY uwmresearchNN@bowser.phys.wvu.edu
```

where `NN` is a placeholder for your two-digit number (e.g. 01, 02, 03, etc.). Initially, the password will be identical to the username, but you will be prompted to change it as soon as you log in. Remember your password!

In your home directory, you can list both normal and hidden files with `ls -a`. If you do this, you'll notice a file called `.bashrc`, which needs to be edited slightly using `vim`, `emacs`, `nano`, etc. – your choice. At the end of the file, you'll notice a line that starts with `module load`, followed by the names of different pieces of software for pulsar processing (separated by spaces). Go to the end of that line and add one more: `psrfits_utils_demorest`. Save the file and type `source .bashrc` on the command line. Now you're ready to go!

Generate TOAs

As you did last week, navigate (`cd`) to your user directory `/lakitu/data/swiggum/FYRE/[user]` and pick up where you left off with the same data file. At this point, you have all successfully run `prepfold` to confirm that you've detected the pulsar. Now, fill in all necessary information in a par file, using the P_{bary} value you found for `P0` and $Epoch_{bary}$ for `PEPOCH`.

1. Run `fold_psrfits` using your par file and set the *subint* length to 10 s. If you can't remember how to do this, use the `-h` flag, look for examples on Slack, and if all else fails, ask for help!

2. The previous step will result in a *folded* file (`GUPPI*.fits`) being written to your working directory. Run `ls -alh` and compare this file's size with that of the raw data file. What are their respective sizes and why do you think they're different?

3. Run `pazi` on the folded file and look at phase vs. time and phase vs. frequency plots – you can toggle back and forth by selecting the plot window and pressing `t` or `f` respectively. Do you notice any obvious RFI? If so, zap it! To save your work, press `s`, then exit `pazi` with `q`. Jot down a few notes here on what you zapped and how you did it.

4. Your folded file with RFI masked will have a `.pazi` extension (`GUPPI*.pazi`). Before making TOAs, run `pam` on this file to scrunch in time/frequency. Scrunch down to five subints and two frequency channels and write a file with an informative extension (e.g. `.t5f2`). Again, you can consult `pam -h` and/or Slack to figure out how to do this.

5. Copy `0038.std` from the parent directory and generate TOAs with `pat -s 0038.std *.t5f2`. You can copy/paste or redirect (`>`) the results into a file called `0038-25_[MJD].toa`. Once you get here, pause and help other groups finish so that we can discuss everyone's results together.

Running Tempo2

TEMPO2 is one version of pulsar timing software that is used to compare collected data (TOAs; `.tim`/`.toa` files) with models based on measured parameters (`.par` files), apply appropriate delays/conversions, compute residuals, fit for new/existing parameters and solve pulsars! After looking at the TOAs you just generated, try your hand and solving a few pulsars in the examples provided using TEMPO2, e.g.:

```
tempo2 -gr plk -f example1.par example1.tim
```

For each one you solve, use the final parameters to find the pulsar in the ATNF catalog and compare your parameters with the ones listed there. To do each example you'll want to make a new directory, `cd` into it and copy over all the example files, e.g.:

```
cp ../../tempo2_examples/* .
```

1. Example 1: Pulsar name? How do your parameters compare?

2. Example 2: Pulsar name? How do your parameters compare?

3. Example 3: Pulsar name? How do your parameters compare?

4. Example 4: Pulsar name? How do your parameters compare?

5. Example 5: Pulsar name? How do your parameters compare?