PROF2FITS

Suryarao Bethapudi

Introduction

PROFIO is a program which takes in PROF file, as outputted by SIGPROC , and spits out a FITS file conforming to the PSRFITS standards as specified at this HTTP URL. It is primarily written to standardize observed data at GMRT and ORT observatories. Another strong motivation for this is to make observed data compatible with PSRCHIVE software suite.

Installation

PROFIO written in C++ using BOOST libraries for parsing and providing program options. Hence, it is depended on boost for successful compilation. PROFIO heavily uses CFITSIO to perform FITS IO operations. There is a Makefile shipped with the code which can be used to build the suite.

To setup BOOST, run

```
sudo apt-get install libboost-all-dev
```

To build CFITSIO, run the following commands in the cfitsio directory:

```
./configure --prefix=XXXX CFLAGS=-fPIC FFLAGS=-fPIC
make shared
make install
make clean
```

Please replace XXXX with root directory (i.e., where you have downloaded the prf2fits source files).

```
make all
```

CFITSIO is built in place, meaning the include file and the library files are build in cfitsio/lib cfitsio/include . This guarantees to not mess up with anything outside the directory. If you have CFITSIO installed and don't want to build it again(for whatever reason), you will have to edit the Makefile manually so that the compiler can find the CFITSIO library and header files. Note that, it is expected that CFITSIO library you had compiled before is a shared object (which means it should end with .so).

Tell me what I should run?

```
sudo apt-get install libboost-all-dev
git clone https://github.com/shiningsurya/prf2fits.git
cd cfitsio
./configure --prefix=XXXX CFLAGS=-fPIC FFLAGS=-fPIC
make shared
make install
make clean
cd prf2fits
make all
```

You will also have to add the path to PROFIO and CFGEN to your PATH variable so that you can use it anywhere in the SHELL.

Usage

PROFIO

If we run PROFIO with no arguments or with -h argument, it prints the following:

```
Options:
  -h [ --help ]                         Prints help
  --bug-in-code                         Prints contact info
  -o [ --observatory ] arg (=observatory.cfg)
                                        Observatory cfg file
  -n [ --pulsar ] arg (=pulsar.cfg)     Pulsar cfg file
  -p [ --project ] arg (=project.cfg)   Project cfg file
  -s [ --scan ] arg (=scan.cfg)         Scan cfg file
  --input arg                           Input PROF file
                                        You can directly give the input prof
                                        file as the one and only positional
                                        argument here.
                                        You dont have to type --input or -i.
  -f [ --out ] arg                      FITS file
                                        Make sure that is no FITS with the same
                                        name there.
                                        CFITSIO routines cause error when we
                                        create a FITS file
                                        with the same name as one already
                                        there.
```

The minimalistic illustration of PROFIO is

```
prf2fits -f out.fits in.prof
```

This takes in in.prof prof file and then creates a out.fits FITS file on successful execution of the code. As shown, input file can be given as the only positional argument as well. Alternatively, one can also achieve the same by doing

```
prf2fits -f out.fits -i in.prof
```

In the above run, observatory, pulsar, project and scan take in the default arguments, we can specify them manually as well. The following illustration makes it more obvious.

```
prf2fits -f out.fits -o obs.cfg -n pul.cfg -p proj.cfg -s scan.cfg in.prof
```

Note that one could also use the long option, for example, the above command is equivalent to

```
prf2fits --out out.fits --observatory obs.cfg --pulsar pul.cfg --project proj.cfg --scan scan.cfg in.prof
```

These cfg files have special structure. They are of the form *key* = *value*;. CFGEN can be used to generate those cfg files.

CFGEN

The help screen is as follows:

```
Usage : <program> arg
+-------+---------------------------------------------+
|  ARG  |               Description                   |
+-------+---------------------------------------------+
|   1   |  Ask everything. Takes some time.           |
|   2   |  Minimal IO. Assumes you're in GMRT.        |
|   3   |  Dump the default values in cfg files.      |
|       |  You will have to edit them manually        |
+-------+---------------------------------------------+
```

Send in 1 as the argument if you want to manually input all the required parameters for successful FITS conversion. Send in 2 if you want to keep most of the default parameters and only change the following:

- RA
- DEC
- Source Name
- Number of channels

Send in 3 if you want to dump with the default values. You will have to manually edit them later.

Queries/Questions/Doubts/Bugs?

If you come across any errors or have any queries on how to make this program suite your problem, please feel free to drop me an email at

ep14btech11008 [at] iith [dot] ac [dot] in