V2.0 June 2, 2024

CCERA Data Format

1 Introduction

In this note, we propose a data format that will accomodate all or nearly all modes of data acquistion currently anticipated for the 12.8-m Carp dish. The adoption of single format eliminates the need to specify the format for each observation and allows for the standardization of analysis code. By archiving crucial file metadata, it makes it possible to reanalyze old datasets.

2 Requirements

1. The format should be suitable for all presently envisaged modes of data acquistion including: transit scans, acquisition of spectral data, and pulsar observations.
2. The data itself should be stored in binary format, readily readable by standard Python (numpy) code.
3. File metadata, stored in plain JSON format, should be available for each data file.
   (a) The metadata should consist of a minimum set of required items plus an arbitrary number of additional optional items.
   (b) Reading the metadata should not require advanced knowledge of the nature of the metadata.

3 Use Cases

As a supplement to the requirements listed above, we describe typical use cases.

3.1 Transit Scan

Here the telescope is set to a fixed azimuth and altitude, and data is taken for a prolonged period. Since objects drift through the beam of the telescope relatively slowly, the sampling rate need not be high. Moreover, spectral data may or may not be required. A typical dataset might involve total run time of 1.5 hours, with a sample rate of 0.1 Hz, with each sample comprising a single total power value. The sample size is quite small (of order 1 kByte).

3.2 24-Hour Drift Scan

Again the telescope is set to a fixed azimuth and altitude and data is taken for a full day. The objective is to see how the 21-cm signal spectrum evolves over the course of the day. Detailed spectral information is of interest, so we assume that the spectrum has 2048 bins. Since the evolution of the spectrum is relatively slow, one spectrum every five minutes suffices, resulting in 288 spectra for a 24-hour scan. Two approaches to data logging are possible: i) a single file comprising 288 spectra of 2048 bins each is recorded; or ii) 288 files, each comprising a single 2048 bin spectrum, are recorded. The total sample size is about 2.4 Mbyte for either approach.

3.3 Pulsar Observation

The dish is set to track a pulsar and data is taken over an extended period (here we assume one hour) so as to average away noise. Assuming the pulsar has a period of one second and we want to have 100 phase bins, a sample rate of at least 200 Hz is needed. Spectral data are needed for de-dispersing the data. Thirty-two spectral bins suffice. Assuming a sample rate of 200 Hz and 32 spectral bins per sample, a one-hour run yields a file size of 100 Mbyte.

4 Implementation

4.1 File Names

Each dataset comprises two files: a binary data file and an JSON metadata file. Although the metadata could be included as a header to the binary data, that would involve files with mixed data types, which can be difficult to read. Separating the two is ultimately simpler.

All files consist of a basename of the form `YYYY-MM-DD_HH:MM:SS_X` and an extension, which is either `.raw` for data or `.json` for metadata. The time specified in the basename is the UTC time at which the run begins. The value of "X" is the USRP channel and can be omitted if only a single channel is recorded.

4.2 Data

All data sets can be represented as a time series. Each element of the series is either a single value or a power spectral density array. The length of the time series (N_{samples}) can range from 1 to a very large number, limited only by the maximum file size (4GB?). The spectral data take the form of a power spectral density (PSD) array (also called a "filter bank" array) having a variable number of bins (N_{bins}) lying in the range $1 < N_{\text{bins}} < 2^N$. In principle, N is unlimited but in practice will generally lie in the range $5 < N < 16$. Each PSD element takes the format of a 32-bit floating point number.

4.3 Metadata

The metadata file should consist of a set of entries, each consisting of a key, followed by a corresponding parameter value. The parameter can be a string, a float, or an integer. Strings must be enclosed by quotes, floats must contain a decimal place, and integers must consist of only digits. Internally, the metadata take the form of a Python dictionary, which can be conveniently represented in an ASCII file using the JSON format.

Table 1 is a list of mandatory parameters that every metadata file must contain. The order does not matter, as long as all parameters are given. From this minimal set of parameters, it is possible to reconstruct other values such as the number of FFTs that are averaged to construct the PSD arrays that are saved in the file, the RA and dec in which the dish is pointing when the run starts, etc. The run_type parameter is crucial in that it tells the system how to determine the telescope's position as time evolves: "Transit" means that the telescope remains at a constant azimuth and altitude, whereas "Track" means that the telescope moves so as to maintain a constant right acension and declination.

The selection of mandatory parameters has been done to avoid redundancy. For example, if one knows the time and the telescope's azimuth and altitude, there is no need to specify the right ascension and declination, since those values can be calculated. A set of helper functions can be used to compute dependent parameters—e.g., right ascension and declination—from the mandatory parameters.

Table 1: List of mandatory metadata parameters.

Name	Type	Meaning
t_start	float	time of first sample in seconds past the epoch
freq	float	center frequency of receiver
srate	float	frequency of USRP clock in Hz
t_sample	float	time between time series samples (or spectra) in seconds
n_chans	int	Number of USRP channels recorded
fft_size	int	number channels in power spectrum
run_type	string	'Transit' or 'Track'
az	float	azimuth of telescope at beginning of run in degrees
alt	float	azimuth of telescope at beginning of run in degrees

Table 2 lists optional metadata parameters. New parameters can be added at any time. The comment parameter is special in that multiple comments can be used.

Table 2: List of optional metadata parameters.

Name	Type	Meaning
target	string	Astronomical object—e.g. "Sun" or "M31"
comment	string	text comment
observatory	string	Observatory name—e.g., "Carp"