fits2hdf Documentation

Release 1.0

Danny Price

August 16, 2016

Contents

CHAPTER 1

About fits2hdf

`fits2hdf` is a conversion to utility to port FITS files to Hierarchical Data Format (HDF5) files in the HDFITS format. In addition, there is a utility to port MeasurementSets (MS) to HDF5 files. This work was first presented at the ADASS XXIV conference in Calgary, 2014. A more complete overview is given in Astronomy & Computing.

The `fits2hdf` utility works by first mapping data from FITS/MS/HDF into an in-memory interchange format (IDI). `fits2hdf` is written in python and uses h5py, pyFits, and pyrap for file I/O.

`fits2hdf` is still under development, so should be considered an 'alpha' release that is likely to change. Community feedback is encouraged, and if you are interested in development please get in touch. This work is intended as a pathfinder toward getting astronomical data into a standardized HDF5 format, so that the advantages of HDF5 can be leveraged in the future.

1.1 Motivation: why HDF5, and why not FITS?

The Flexible Image Transport System (FITS) file format has enjoyed **several decades** of widespread usage within astronomy. Its ubiquity has been attributed to the guiding maxim "once FITS, always FITS": that changes to the FITS standard must be incremental so as to never break backward compatibility. This maxim limits what modifications can be made, and FITS is showing its age; the guiding principle that made FITS so successful can now be seen as its Achilles heel.

The limitations of FITS are succinctly summarized in Thomas et al. (2014) and Thomas et al. (2015). Some of the limitations are quite frankly archaic: 8-character maximum keywords in the FITS header, lack of Unicode support, and incomplete support of basic unsigned integer types. Other limitations become apparent when compared against newer formats: FITS has no support for chunking, parallel I/O, or hierarchical data models.

Motivated by data volumes, the Hierarchical Data Format (HDF5) is becoming increasingly common in astronomy and science in general. HDF5 has several advantages over FITS, with I/O access speed being a compelling reason to make the switch. For example, HDF5 allows efficient reading of portions of a dataset, such as reading along slowly-varying axes of a dataset, which incurs significant overhead when reading from FITS files. A switch to HDF5 may save you lots of processing time.

For a more complete discussion, see HDFITS: porting the FITS data model to HDF5.

1.2 The HDFITS specification

The HDF5 format has an abstract data model, capable of storing myriad data structures. While there are many possible mappings from FITS into HDF5, we've implemented one that we are calling **HDFITS v1.0**. We encourage feedback, and invisage that comments from the broader community will lead to a HDFITS v2.0.

The goal of HDFITS/fits2hdf is to provide a HDF5-based equivalent of the FITS file format, and to provide utilities for converting between the two formats. The motivation of this approach, as opposed to creating an HDF5-based format from scratch, is that decades of widespread FITS usage has left a legacy that would otherwise be discarded. By preserving the familiar underlying data model of FITS, software packages designed to read and interpret FITS can be readily updated to read HDF5 data. Maintaining backwards-compatibility with FITS, so that data stored in HDF5 files can be converted into FITS for use in legacy software packages is another persuasive reason to pursue a FITS-like data model within HDF5.

Of course, a port of the FITS data model to HDF5 does not address issues with the FITS data model itself. Nevertheless, as the HDF5 data model is abstracted from its file format, an HDF5-based version of the FITS data model can be extended without requiring changes to the storage model. HDFITS can be used as a starting point and as a testbed for enhancing the FITS data model.

1.3 Alternatives

There are a bunch of alternative data formats, and if you're a free spirit you can always roll your own higher-level data model inside the HDF5 abstract data model. Otherwise, look into:

- NDF is the file format used by Starlink. Recently, Starlink added HDF5 support, meaning that you can use the Starlink utilities to convert from FITS into the NDF-HDF5 format.
- hickle provides a HDF5-based drop-in replacement to the Python *pickle* package, allowing nice & lazy dumping of common python objects to file.
- ASDF is a file format being developed for interchange, particularly for JWST.
- MeasurementSets are how the CASA software package stores its data.
- VOTables is an XML-based format designed for the Virtual Observatory.

1.4 Copyright and referencing

This software is licensed under the MIT license. If you use this in published research, it sure would be swell if you could cite the fits2hdf Astronomy & Computing paper:

```
D.C. Price, B.R. Barsdell, L.J. Greenhill, HDFITS: Porting the FITS data model to HDF5,
Astronomy and Computing, Available online 22 May 2015, ISSN 2213-1337,
http://dx.doi.org/10.1016/j.ascom.2015.05.001.
```

`fits2hdf` makes use of a few excellent packages:

- Astropy, a community-developed core Python package for Astronomy.
- h5py, a Pythonic interface to the HDF5 binary data format.

CHAPTER 2

Getting started

2.1 Installation

To install, you first need to clone the directory from github:

```
git clone https://github.com/telegraphic/fits2hdf
```

and then run:

```
python setup.py install
```

from the command line. You'll need astropy and h5py to be installed. If you want to use bitshuffle compression (good for radio astronomy data), you'll need to install that too.

Installation through `pip` will likely be added in the future.

2.2 Command line usage

To use `fits2hdf` to convert FITS files to HDF5, use the *fits2hdf* command line tool:

```
fits2hdf input_dir output_dir <options>
```

Optional arguments are:

-h, --help show this help message and exit

-c COMP, --compression=COMP Data compression algorithm: None, lzf, gzip and bitshuffle (if installed).

-x EXT, --extension=EXT File extension of FITS files. Defaults to .fits

-v VERBOSITY, --verbosity=VERBOSITY verbosity level (default 0, up to 5)

-s SCALE_OFFSET, --scaleoffset=SCALE_OFFSET Add scale offset (HDF5 compression option). NB: this can be lossy!

-S, --shuffle Apply byte shuffle filter (HDF5 compression option)

-C, --checksum Compute fletcher32 checksum on datasets.

To convert back into FITS, run `hdf2fits`, which uses similar options:

```
hdf2fits input_dir output_dir <options>
```

As many HDF5 features don't have equivalents in FITS, this will (probably) only work for HDFITS files.

2.3 Quickly adding HDF5 support in Python

If you have an existing program and want to quickly be able to read HDFITS files, just change your:

```
from astropy.io import fits
```

line to:

```
from fits2hdf import pyhdfits as fits
```

Or, if you're using pyfits, change your `import pyfits as pf` line to `from fits2hdf import pyhdfits as pf`.

By doing this, any data in HDFITS will be converted to the PyFITS/ Astropy HDUList object on the fly.

2.4 Loading HDFITS files in python

Functions to read / write HDFITS files into an in-memory data object in python are located in `fits2hdf.io.hdfio`. There are equivalent functions to read FITS files into an interchange format in `fits2hdf.io.fitsio`.

For example, to read a FITS file in, then export it to HDFITS you would do:

```
from fits2hdf.io.fitsio import read_fits
from fits2hdf.io.hdfio import export_hdf
a = read_fits('my_file.fits')
export_hdf(a, 'my_file.hdf')
```

and to convert the other way:

```
from fits2hdf.io.fitsio import export_fits
from fits2hdf.io.hdfio import read_hdf
a = read_hdf('my_file.hdf')
export_hdf(a, 'my_file.fits')
```

For more exciting API musings you can read the API.

CHAPTER 3

API

- idi: Abstract classes for python Header-Data unit object
- io: File input / output functions.
- pyhdfits: A drop-in replacement for `pyfits` or `astropy.io.fits`.
- unit_conversion: Unit parsing, checking and sanitizing.
- printlog: Functions for pretty printing and logging.
- check_file_type: File type checking routines.

CHAPTER 4

About fits2hdf

`fits2hdf` is a conversion to utility to port FITS files to and from Hierarchical Data Format (HDF5) files in the HDFITS format. In addition, there is a utility to port MeasurementSets (MS) to HDF5 files. This work was first presented at the ADASS XXIV conference in Calgary, 2014. A more complete overview is given in Astronomy & Computing.

The `fits2hdf` utility works by first mapping data from FITS/MS/HDF into an in-memory interchange format (IDI). `fits2hdf` is written in python and uses h5py, pyFits, and pyrap for file I/O.

`fits2hdf` is still under development, so should be considered an 'alpha' release that is likely to change. Community feedback is encouraged, and if you are interested in development please get in touch. This work is intended as a pathfinder toward getting astronomical data into a standardized HDF5 format, so that the advantages of HDF5 can be leveraged in the future.