New Features in AST - a WCS Management and Manipulation Library

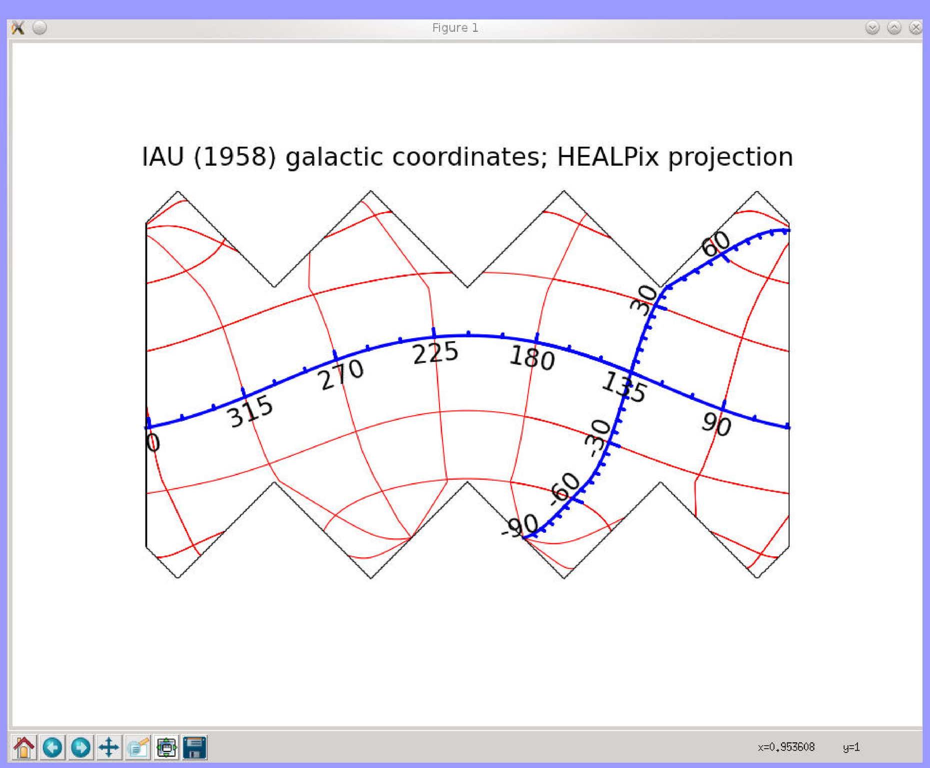
Dr David Berry
d.berry@jach.hawaii.edu

David Berry, Tim Jenness

Joint Astronomy Centre, Hilo, Hawaii

PyAST—a Python wrapper for AST:

- PyAST is a Python library that provides wrappers for the majority of AST functions.
- PyAST requires Python 2.7 or later (version 3 is supported).
- PyAST depends only on the numpy library. A copy of AST is bundled with PyAST, so no other parts of the Starlink Software Collection are required.
- PyAST is publicly available and can be downloaded from github. See the "Links" section below.
- An optional interface is provided for use with the AST Plot class that allows annotated axes to be drawn using the popular matplotlib graphics library. This has many advantages over direct use of the axis annotation facilities provided by matplotlib itself. For instance, axis labels can be placed within the body of the plot, rather than round the edges—beneficial for many projections, and essential for all-sky projections. Also, AST can draw coordinate grids for projections that have peculiarities such as singularities and discontinuities.



PyAST displaying a HEALPix grid in a matplotlib window.

- An optional interface is provided for use with the AST FitsChan class that allows AST to read and write FITS headers stored in a PyFITS header.
- PyAST provides a few high level functions that wrap up other PyAST calls to perform commonly required operations more easily.

```
>>> import pyfits
>>> import starlink.Atl as Atl
>>> import matplotlib.pyplot
>>>
>>> hdulist = pyfits.open( 'hpx.fit' )
>>> Atl.plotfitswcs( matplotlib.pyplot.figure().add_subplot(111),
>>>                  [ 0.1, 0.1, 0.9, 0.9 ], hdulist )
>>> matplotlib.pyplot.show()
```

The code needed to produce a plot such as the HEALPix grid shown above.

Support for FITS-WCS "TAB" algorithm:

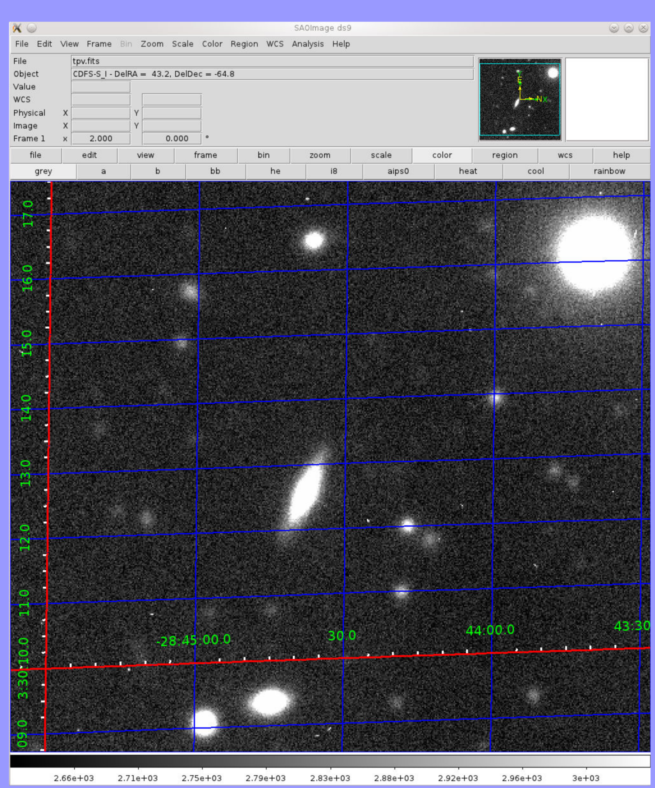
- AST now includes support for reading and writing tabular WCS information in the form of FITS headers using the "-TAB" algorithm described in FITS-WCS Paper III (*Representation of Spectral Coordinates in FITS,* Greisen et al, 2006).
- Currently, no support is included for the multi-dimensional tables needed to describe non-separable axes.

Support for distorted projections:

- When published, FITS-WCS Paper IV will address the issue of the representation of distorted projections. But in the meantime, AST supports several of the interim schemes that are in common use, as listed below.
- IRAF "**-TNX**": AST can now reads TNX projections described by Chebyshev or simple polynomial with half-cross terms.
- IRAF "**-ZPX**": AST can now reads ZPX projections described by Chebyshev or simple polynomial with half-cross terms.
- Spitzer "**-SIP**": AST has been able to read SIP projections for some time, but SIP support has been improved recently. Within a SIP header, the forward and inverse transformations between world and pixel coordinates are defined by separate polynomials, but some SIP headers do not define an inverse transformation (from world to pixel coordinates). For such a header, AST can now implement an iterative inverse transformation.
- NOAO "**–TPV":** AST now supports this renaming of the distorted TAN projection included in an early draft of FITS-WCS Paper II (*Representation of celestial coordinates in FITS,* Calbretta & Greisen).
- SCAMP "**–TAN":** This is exactly the same as the TPV projection, except that it uses a CTYPE code of "-TAN" instead of "-TPV". AST differentiates between SCAMP TAN headers and standard TAN headers by looking for PV keywords attached to the latitude axis (a standard TAN projection should have no such latitude PV keywords).
- AUTOASTROM "**–TAN":** This is another representation of the TPV projection, again using a CTYPE code of "-TAN", but using QV keywords instead of PV keywords to store the polynomial coefficients.

DS9 and AST:

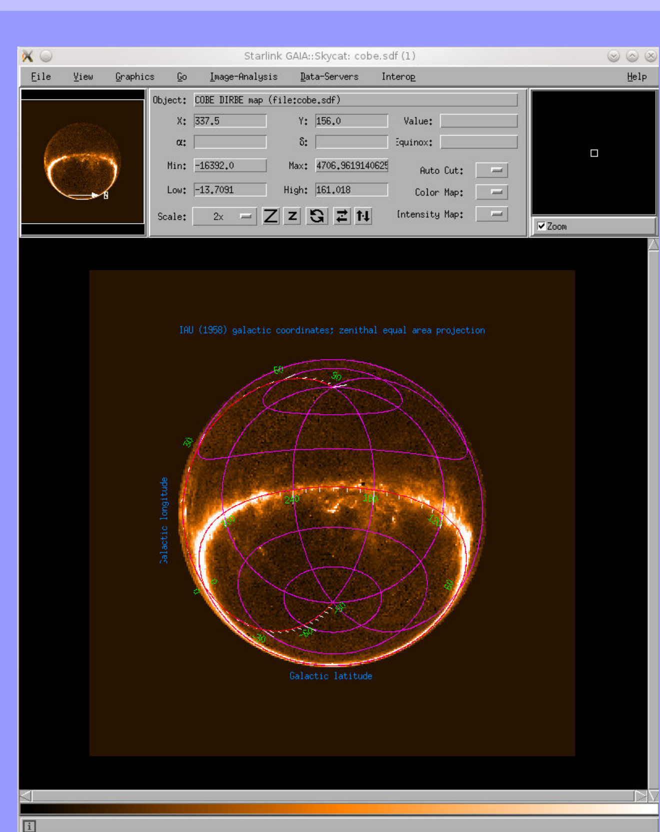
- The DS9 image browser has for many years used AST to draw its annotated coordinate grids. As of DS9 version 7.0 (currently in beta testing), it will use AST additionally for all its WCS transformations, thus benefitting from the improvements to support for distorted projections listed above.



DS9 using AST to display a coordinate grid for a TPV (a distorted TAN) projection.

What is AST?

- AST is a library of functions that implement an object oriented model for describing physical coordinate systems, and the transformations that exist between them.
- It provides a comprehensive range of facilities for attaching world coordinate systems (WCS) to astronomical data, for retrieving and manipulating that information and for generating graphical output such as coordinate grids based upon it.
- It can read and write WCS information in several different forms, including FITS-WCS headers of various flavours.
- It is written in pure ANSI C but also has interfaces for Python, Java, Perl and FORTRAN.
- It has built-in intelligence for identifying types of celestial, spectral, time and other coordinate systems (including compound systems that combine axes of different types) and determining how to transform between them. This allows general purpose code to be written that makes no assumptions about the nature of the coordinate systems.
- It includes a flexible and versatile "tool-kit" for creating and modifying collections of coordinate frames interconnected by arbitrarily complex transformations.
- It includes easy-to-use graphical facilities that allow the production of annotated 2D or 3D grids. Graphics are draw via a simple "driver" module which AST calls to draw lines, strings, markers, *etc*. AST includes drivers for PGPLOT; drivers for other graphics systems (Tcl/Tk, Java/Swing, *etc.*) can easily be (and have been) written.
- It is actively supported and developed by the Joint Astronomy Centre, Hawaii.
- It forms the basis of the coordinate handling facilities in the Starlink Software Collection, including GAIA, SPLAT and KAPPA. It is also used in other non-Starlink software such as DS9 and XIMAGE.



GAIA using AST to display a coordinate grid for a ZEA (zenithal equal area) projection.

Links:

Starlink Software Collection: **www.starlink.ac.uk**

AST: **www.starlink.ac.uk/ast**

PyAST download**: github.com/timj/starlink-pyast/downloads**

PyAST docs: **dsberry.github.com/starlink/pyast.html**

DS9: **hea-www.harvard.edu/RD/ds9/**

Joint Astronomy Centre, Hawaii: **www.jach.hawaii.edu**

Numpy: **numpy.scipy.org**

Matplotlib**: matplotlib.sourceforge.net**

PyFITS: **www.stsci.edu/resources/software_hardware/pyfits**