Where will PyRAF lead us?: The future of data analysis software at STScI

Perry Greenfield
Science Analysis Tools Project

Space Telescope Science Institute
Baltimore, MD

Original Goals

- A more robust CL for IRAF
  - Primarily to address the difficulties of debugging and error handling
  - But also to allow a more standard scripting language that has wide support
  - It's hard to compete these days with one's own custom language
- A way to integrate non-IRAF software with IRAF capabilities without resorting to many separate processes.
- Provided the possibility that IDL-like capabilities could be added.

Why Python?

- First: Why not IDL?
  - Writing a CL for IRAF based on IDL not feasible.
  - That's what we would have done if it were.
- There are other drawbacks for IDL but they were secondary. E.g.,
  - Commercial (and expensive), closed source
  - Not general purpose (driving users to use other solutions for other scripting needs)
  - Doesn't support compiled code subroutines well

Python pros

- Free, Open Source
- Very strong (deep and broad) user and developer community
  - Still growing
  - Long lifetime expected
- General purpose
- Very portable
- Very extensible (in Python and with compiled code)
- Comparatively easy to learn
- Scales well
- *Still the clear choice among scripting languages*

IRAF CL features not currently supported

CL is not completely emulated:

- Not all CL language behaviors supported if considered problematic
- GOTOs not completely (but now partly) supported
- Background tasks*
- Package unloading*
- Text-based epar*
- Graphics redirection*
- New IRAF CL error handling*

Python and Astronomical Data Analysis

The conventional view:

- It's a good scripting language
  - Becoming the standard scripting language for astronomy:
    - PyMIDAS
    - Parseltongue (AIPS)
    - ALMA
    - PySL (CIAO/s-lang)
- But that's all it's good for

Python and Astronomical Data Analysis (cont.)

We take a broader view:

- Eminently suitable for many astronomical applications with the appropriate libraries.
  - Thus do as much in Python as possible
- And when it isn't, it is possible to use compiled languages to handle the exceptions.
- It's also more accessible to development by astronomers.

Thus most of our Python infrastructure efforts have been towards enabling this.

Our efforts

To expand the tools within Python to make it more productive as a development environment. E.g.,

- PyRAF (interface to IRAF tasks)
- PyFITS (FITS I/O)
- Numarray (array manipulations)
- Scipy (numerical libraries)
- Matplotlib (2D plotting)
- Astronomical utility libraries (astrolib)
  - WCS
  - Coordinates
  - Photometry
  - VO tools

Our efforts (cont.)

To begin promoting its use by astronomers:

- Introductory tutorial written showing how to use Python for *interactive* data analysis:
  - http://www.scipy.org/wikis/topical_software/Tutorial
  - Tutorials given to STScI staff and at ADASS
- Astrolib wiki set up

Future plans for PyRAF

- Add capabilities marked with ‘*’ in near future
- Improve graphics features
- Support Gemini use of PyRAF in their pipelines
- PyRAF work near the top of the priority queue now
- In strategic value to STScI development, PyRAF is dropping relative to our other Python infrastructure projects
  - No new software being written for IRAF VOS
  - More of IRAF functionality slowly appearing in other Python-based libraries
  - PyRAF was our mechanism to transition to Python and has been very successful in allowing us to do so.
- Its strategic value is to our user community

Numarray vs Numeric/numeric3

- Effort underway to replace both Numeric and numarray with “scipy_core” (aka Numeric3) to end current split in user community
- Much progress made
- STScI currently testing scipy_core
- Plan to start porting our software (e.g, pyfits, Multidrizzle, etc.) very soon
- Will support both for some transition period