REVOLUTION IN ASTRONOMY WITH ALMA: THE THIRD YEAR
ASP Conference Series, Vol. 499
Daisuke Iono, Ken'ichi Tatematsu, Al Wootten, Leonardo Testi, eds.

Radio Astronomy Tools in Python: Spectral-cube, pvextractor, and more

Adam Ginsburg,[1] Thomas Robitaille,[2] Chris Beaumont, Erik Rosolowsky,[3] Adam Leroy,[4] Crystal Brogan,[5] Todd Hunter,[5] Peter Teuben,[6] and Drew Brisbin[5]

[1]*European Southern Observatory, Garching bei Muenchen, Bavaria, Germany;* `adam.ginsburg@eso.org`

[2]*MPIA Heidelberg*

[3]*Alberta Canada*

[4]*OSU*

[5]*National Radio Astronomy Observatory, Charlottesville, Virginia, USA*

[6]*Maryland*

Abstract. The radio-astro-tools organization has been established to facilitate development of radio and millimeter analysis tools by the scientific community. The first packages developed under its umbrella are:

- The `spectral-cube` package, for reading, writing, and analyzing spectral data cubes
- The `pvextractor` package for extracting position-velocity slices from position-position-velocity cubes along aribitrary paths
- The `radio-beam` package to handle gaussian beams in the context of the `astropy` quantity and unit framework
- `casa-python` to enable installation of these packages - and any other - into users' `CASA` environments without conflicting with the underlying CASA package.

Community input in the form of code contributions, suggestions, questions and comments is welcome on all of these tools. They can all be found at `http://radio-astro-tools.github.io`.

Many tools exist for analysis of imaging data, spectra, and spectral cubes, but most of these tools are standalone packages developed for a specific single purpose without extensibility or interoperability in mind. We saw the need for a community-developed package for analysis of ALMA and other radio cube data and have begun to implement some of the most general features needed by all users. This work was begun as part of a North American ALMA development study "Community Science Tool Development" with the goal of sparking further community development for ALMA data analysis.

1. `spectral-cube`

The `spectral-cube` package is a robust tool for reading cubes from FITS, CASA .image, and CLASS .lmv files and writing them to FITS or sending them to other analysis packages. It is intended to be the main hub for spectral cube analysis around which other tools can be constructed. When reading from FITS files, it can very quickly access subsets of data without reading the whole cube by using memory mapping.

Key features include:

- Moments along any axis, preserving units
- Efficient masking using lazy evaluation
- A complete implementation of the Greisen & Calabretta (2006) spectral world coordinate system, allowing easy transformation between velocity, wavelength, and frequency *without* resampling the data
- Easy exporting to `yt`, `glue`, `ds9`, and the `pvextractor` GUI

Example:
Load data:
> from spectral_cube import SpectralCube
> cube = SpectralCube.read('SgrB2_a_03_7M.H2CS303-202.image')

Change units:
> from astropy import units as u
> vcube = cube.with_spectral_unit(u.km/u.s, velocity_convention='radio',
> rest_value=103.04045*u.GHz)

Measure moments of sub-cubes:
>m0 = vcube.spectral_slab(-10*u.km/u.s,10*u.km/u.s).moment0(axis=0)

2. `pvextractor`

The position-velocity extractor package allows users to specify an arbitrary path in position-position space to extract a position velocity diagram. The result is readily output in FITS form with `CTYPE1='OFFSET'`. The path can be extracted by averaging over a specified width or by interpolating between pixels. Paths can be drawn using an internal GUI, using `glue`, or using `ds9` line or segment regions.

3. `radio-beam`

The radio-beam package provides a wrapper around a `beam` unit in the astropy units framework. It includes tools to read the beam parameters from FITS headers using the `BMAJ`, `BMIN`, and `BPA` keywords, or to recover them from an `AIPS`-reduced file that has the information stored in a `HISTORY` keyword. It includes the ability to convolve or deconvolve beams.

4. `casa-python`

The `casa-python` tool gives users access to the python environment installed with CASA without starting up CASA itself. It allows users to install packages into their own user directory, providing access to external packages without risk of conflict with CASA's internal packages.

Example - Install astropy into your CASA environment:
> casa-pip install astropy