Python Ephemeris Module for Radio Astronomy

Thomas B. Kuiper*

ABSTRACT. — An extension of the Python `pyephem` module was developed for Deep Space Network (DSN) radio astronomy. The class `DSS()` provides the geodetic coordinates of the DSN stations as well as other properties such as antenna diameter. The class `Quasar()` provides positional data for the sources in the National Radio Astronomy Observatory Very Large Array (NRAO VLA) Calibrator Handbook and flux estimates based the University of Michigan Radio Astronomy Observatory (UMRAO) Database or the VLA Calibrator Handbook. Flux calibration data are also available for the bright planets. Class `Pulsar()` provides the data from the Australia Telescope National Facility (ATNF) Pulsar Catalogue in Python format.

I. Background

Deep Space Network (DSN) stations observe radio astronomical sources for research, calibration, navigation, clock synchronization and other purposes. Therefore, engineers and scientists need convenient tools to predict the positions of radio sources and estimate their fluxes and other properties. For many years, `xephem` [1] has been a popular tool with amateur and professional astronomers. The underlying astronomical calculations were adapted to Python as module `pyephem` [2]. However, the tool and module are still strongly oriented towards optical astronomy.

This article reports an extension of ephem, called `Ephem`, which provides all the original features of `ephem` and adds DSN stations and catalogs of radio sources and their properties.

II. Superclasses

Whereas Python is flexible about the programming paradigm — it is easy to switch to Python from procedural languages like FORTRAN and C — `ephem` is object-oriented. The source and the observer are objects derived from classes and much of the astronomical calculation connecting the two is hidden from the programmer. It can make programming quite succinct:

* Astrophysics and Space Sciences Section.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

```
In [1]: from Ephem import *
In [2]: source = Quasar ("3C84")
In [3]: station = DSS(14)
In [4]: station.date = now()
In [5]: source.compute(station)
In [6]: print(station.next_rising(source),
              station.next_setting(source))
(2012/8/19 04:35:20, 2012/8/18 21:58:41)
```

This example also introduces two of the new classes.

A. Class DSS()

The class DSS() is based on Observer(), which describes the properties of the observer's location, such as latitude, longitude, and elevation. DSS() inherits all the methods (functions) and attributes (parameters) of the Observer() class. The main purpose of the new class is to enter the known parameters of the DSN station from a stored table. It also adds additional attributes. In its current implementation, the station data are, for ease of code maintenance, taken from DSN radio astronomy module Astronomy, but the table could easily be incorporated into the Ephem source code.

The attributes inherited from Observer() and supplied from the table are elevation, lat, long, and name according to the station number, which is provided as a required argument when an instance is created. (See ipython input line [3] above for an example.)

Additional attributes unique to DSS() are timezone (difference between local standard time and Universal Time) and diam (antenna diameter in meters).

```
In [7]: station.name
Out[7]: 'Goldstone Mars'
In [8]: station.diam
Out[8]: 70
```

B. Class Quasar()

The class Quasar() is based on the class FixedBody(), which is used for sources that are more or less fixed on the celestial sphere. A source name is required when an instance Quasar() is created. The name can be a name from the Third Cambridge Catalogue, without a space, such as "3C454.1," or the Julian International Astronomical Union (IAU) convention[1] ("J2250+714") [3] or the Besselian IAU convention ('B2248+712'). If a "3C" includes a space, the space is removed by the module. If the IAU prefix is left off, it is assumed to be a Julian IAU name.

The name cross-reference depends on the National Radio Astronomy Observatory (NRAO) list of Very Large Array (VLA) calibrators. The sources known to the Ephem module can be listed.

---

[1] http://cdsweb.u-strasbg.fr/Dic/iau-spec.html

```
In [9]: cat_3C_dict.keys()
Out[9]:
['3C57',
 '3C293',
 ....
 '3C368',
 '3C207']
In [10]: Bname_dict.keys()
Out[10]:
['0620+389',
 '1145+268',
 ...
 '2149-306',
 '1514-241']
In [11]: Bnames.keys()
Out[11]:
['1510-057',
 '1748+085',
 ...
 '1051-316',
 '0428-379']
```

An unknown name will raise an exception (generate an error halt). The inherited attribute name is the IAU J-name without the leading J.

The inherited attributes a_ra and a_dec are provided from the VLA calibrator catalogue. These are the astrometric geocentric position for the epoch, which is J2000 for this module. If compute() is invoked with a date, the apparent geocentric position, g_ra and g_dec, is calculated correcting for precession, relativistic deflection, nutation, and aberration. If the compute() method is invoked with an Observer() instance and the instance's date attribute has been set, then the attributes ra and dec give the apparent topocentric position, after correction for parallax and refraction. The private _class attribute is set to 'Q'. Quasar() is instantiated initially with a date of "2000/1/1 00:00:00".

In ephem, refraction correction is done for visible light. To disable the refraction correction or to apply your own, the Observer() attribute pressure must be set to zero. The module Ephem does this as the default since the Observer() default is 1010 mBar.

New attributes added for the class Quasar() are Bname, Jname, freq, flux, and flux_ref. The latter indicates which catalog was used for the flux calculation.

The source flux is computed using the method interpolate_flux(freq, date=None).[2]
The first argument resets the instance's freq attribute. If the argument date is not given, the current date and time are used.

In computing the flux, the module first checks to see if the required data are available in the University of Michigan Radio Astronomy Observatory Database. The University of Michigan Radio Astronomy Observatory (UMRAO) Database consists of long-term light curves showing two-week averages of the total flux density for selected sources. These observations

[2] The first argument self, a Python convention, has been left out so as not to confuse readers unfamiliar with object-oriented Python.

are provided as a service to the astronomical community to be used for calibrating data obtained with other radio telescopes.[3] The flux data are first interpolated or extrapolated in time and then a linear fit is done for the three UMRAO Database frequencies of 4.8, 8.0, and 14.5 GHz.

If UMRAO data are not available, then the fluxes in the NRAO VLA Calibrator Manual[4] are used. If the database entry contains data at two or more wavelengths, a linear fit to frequency is done and interpolated or extrapolated. The fluxes in the table were taken at different, unspecified epochs so the resulting flux can only be considered a rough indication. Flux history is available for VLA calibrators through a Java interface,[5] but getting those data automatically is a project for the future.

C. Class `Pulsar()`

The class `Pulsar()` is also based on `FixedBody()`. The data for this class are taken from the Australia Telescope National Facility Pulsar Catalogue.[6] As for `Quasar()`, the pulsar name may be given in IAU Julian or Besselian notation. However, the Besselian notation is abbreviated in the manner common in pulsar radio astronomy, that is,

```
In [15]: x=Pulsar('B0329+54'); y=Pulsar('J0332+5434')
In [16]: x.name,y.name
Out[16]: ('J0332+5434', 'J0332+5434')
```

This also shows that, unlike for `Quasar()`, the instance name includes the J prefix. If a prefix letter is not given when the instance is created, J is assumed.

A `Pulsar()` instance inherits all the `FixedBody()` methods and attributes. In addition, all the properties given in the Australia Telescope National Facility (ATNF) Pulsar Catalogue are included in a `properties` attribute, except for those properties for which inherited or new attributes exist.

```
In [1]: from Ephem import *
In [2]: p = Pulsar('B0329+54')
In [3]: print(p.a_ra,p.a_dec)
(3:32:59.37, 54:34:43.6)
In [4]: print(p._pmra, p._pmdec)
(16.999999870815579, -9.5000004262777598)
In [5]: print(p.period,p.dpdt)
(714.51969972580105, 2.0482647498485117e-15)
In [6]: print(p.properties.keys())
['F0', 'POSEPOCH', 'F2', 'DM', 'DIST_DM', 'TAU_SC', 'S925',
 'S1400', 'PX', 'W50', 'DIST_AMN', 'W10', 'F1', 'DIST_DM1',
 'S400', 'PEPOCH', 'SURVEY', 'RM', 'SPINDX', 'DIST_AMX',
 'S600']
In [7]: print(p.properties['DM'])
26.833
```

---

[3] This research has made use of data from the University of Michigan Radio Astronomy Observatory, which has been supported by the University of Michigan and by a series of grants from the National Science Foundation and the Fermi program through NASA Fermi grants NNX09AU16G, NNX10AP16G, and NNX11AO13G.

[4] http://www.aoc.nrao.edu/~gtaylor/csource.html

[5] http://www.vla.nrao.edu/astro/calib/flux/

[6] http://www.atnf.csiro.au/people/pulsar/psrcat/

The Pulsar() private attribute _class is set to "L". Pulsar() is instantiated initially with a date of "2000/1/1 00:00:00". Also illustrated above are the new attributes period and dpdt.

D. Function calibrator

Module Ephem has a global method (function) calibrator that will return either a Planet() or a Quasar() instance according to the name passed to it. Each planet is its own class.

```
In [8]: pl = calibrator('Venus')
In [9]: type(pl)
Out[9]: <class 'ephem.Venus'>
In [10]: qs = calibrator("3C273")
In [11]: type(qs)
Out[11]: <class 'Ephem.Quasar'>
In [12]: pl.name
Out[12]: 'Venus'
In [13]: qs.name
Out[13]: '1229+020'
```

DSN radio astronomy module Radio_Astronomy.radio_flux has functions planet_brightness(planet, freq) and get_planet_flux(planet, freq, date) to provide the necessary calibration data. planet in this case is a string, not an ephem Planet() instance.

III. Supporting Modules

The Ephem module depends on other modules developed for DSN radio astronomy.

A. Module Radio_Astronomy.michigan

This module uses a Python dictionary (a look-up table) to get the URL[7] of the source's data table in the UMRAO database. get_flux_data(url) then reads that table.

Function polate_flux(Jname, datenum, freq) uses the above function to get the data. For each frequency, it finds the two nearest data points in the table and does a linear interpolation or extrapolation to the given date, provided as a module matplotlib.dates date number. It then does a linear fit to the three frequencies in the database — 4.8, 8.0, and 14.5 GHz — and interpolates or extrapolates to the specified frequency.

B. Module Radio_Astronomy.vla_cal

This module has functions to get and parse the VLA Calibrator Manual and then saves the data on local disk as a Python dictionary keyed to the IAU Julian names. This is done only rarely, however.

The functions in the module that are relevant to module Ephem are:

[7] Uniform Resource Locator, an address for Web-accessible data.

get_3C_coords(3Cname)
: Gets the coordinates and IAU name for a given 3C source.

get_cal_dict()
: Gets the VLA calibrator dictionary from local disk.

get_cal_data(source)
: Gets the data for a given source identified by IAU J-name.

VLA_name_xref(cal_data)
: Gets Python dictionaries cross-referencing B and 3C names to J names.

fix_name(name)
: Fixes an IAU designator that is too long or too short.

IAU_name_parts(name)
: Splits the IAU name into an right ascension part and an unsigned declination part.

match_IAU_name(name, name_list)
: Tries to find if name that more or less matches something in name_list.

C. Module **Radio_Astronomy.radio_flux**

This module provides the following functions:

radio_flux(source, freq)
: Computes the flux density in Jy of standard calibrators 'Virgo', 'Omega', or 'Orion' based on work by Ekelman [4].

planet_brightness(planet, freq)
: Computes the brightness temperature of a planet. For Venus, the data are taken from observations by Baars et al. [5] for 14.5 GHz, Butler et al. [6] for 4.86–22.46 GHz, Ulich et al. for 86.1 GHz [7], and Yefanov et al. [8] for 37.5 and 138.9 GHz. The Jupiter data are from Baars et al. [5] and Ulich et al.[7]. The Saturn data are for the one frequency in Ulich et al. [7].

get_planet_flux(planet, freq, date)
: Planet fluxes based on the brightness observations and modeling of Baars et al. [5] for Jupiter, Welch et al. [9] for Saturn, Dent et al. [10] for Mars, and Morrison and Klein for Mercury [11].

D. Module **Astrophysics.Pulsars.pulsar_data**

This module provides data from the Australia Telescope National Facility Pulsar Database [12] in a Python format. The data are stored on local disk in the module's directory and should be refreshed from time to time. The data are in the form of a Python dictionary pulsar_data.data keyed to the pulsar IAU Julian names. Each entry is itself a dictionary keyed to the pulsar property names used in the ATNF catalogue:

```
In [1]: from Astrophysics.Pulsars.pulsar_data import data
In [2]: data['J0332+5434']
Out[2]:
{'DECJ': '54:34:43.57',
 'DIST_AMN': '1.7',
 'DIST_AMX': '2.0',
 'DIST_DM': '1.44',
 'DIST_DM1': '0.98',
 'DM': '26.833',
 'F0': '1.399541538720',
 'F1': '-4.011970E-15',
 'F2': '5.3E-28',
 'PEPOCH': '46473.00',
 'PMDEC': '-9.5',
 'PMRA': '17.0',
 'POSEPOCH': '46473.00',
 'PSRB': 'B0329+54',
 'PX': '0.94',
 'RAJ': '03:32:59.368',
 'RM': '-63.7',
 'S1400': '203',
 'S400': '1500',
 'S600': '1300',
 'S925': '386',
 'SPINDX': '-1.6',
 'SURVEY': 'misc,jb1,gb1,gb2,gb3,gb4',
 'TAU_SC': '6.31e-08',
 'W10': '31.4',
 'W50': '6.6'}
```

Note that all data are strings may need to be converted using `float()` or other function. Explanations of the parameter keys are available through `key_help(key)`:

```
In [4]: from Astrophysics.Pulsars.pulsar_data import key_help
In [5]: key_help('POSEPOCH')
Out[5]: 'Epoch at which the position is measured (MJD)'
```

Sometimes data may be given in one form and sometimes another, like `P0` and `F0`, or a parameter not given may be computable from others. For this reason, the following functions are provided:

`equatorial(data)`
: Returns the J2000 right ascension and declination in hours and degrees from keyed data or from ecliptic or galactic coordinates, whatever is given.

`period(data)`
: Returns the pulsar period in milliseconds.

`period_change_rate(data)`
: Returns the time rate of change of the period in s/s.

The first of these depends on yet another module, `Astronomy`, for the functions `formats.parse_colon_delimited_angles(ra, decl)` and `ecliptic_to_J2000(elong, elat, mjd)`, but to limit the depth of recursion these are left as exercises for the reader.

IV. Recommendations for Future Work

It would be more elegant to extend certain planet classes, namely `Mercury`, `Venus`, `Mars`, and `Jupiter` with methods to calculate their radio frequency fluxes.

Much more flux calibration data would be available by automating getting flux history data from the VLA database.

Extending `Ephem` to deal with Doppler shifts and pulsar timing, even crudely, would enhance its usefulness to radio astronomy. The required software exists in other modules so this would be a straightforward programming task.

V. Conclusion

`pyephem` is a sophisticated tool for planning and reducing observations. By extending it with additional classes and methods derived from radio astronomical catalogs, it can serve a larger number of astronomers. Hopefully, this feature can be incorporated into future versions of `pyephem`, but in the meantime module `Ephem` can provide these features.

`Ephem` is quite accurate, suitable for optical positional astronomy and therefore better than required for single dish radio astronomy. However, it is not a precision tool. For such calculations, software based on the JPL's ephemeris [13] such as SPICE [14] are required.

References

[1] E. C. Downey, "XEphem: Interactive Astronomical Ephemeris," *Astrophysics Source Code Library*, p. 12013, December 2011.

[2] Brandon C. Rhodes, "PyEphem: Astronomical Ephemeris for Python," *Astrophysics Source Code Library*, p. 12014, December 2011.

[3] M.-C. Lortet, S. Borde, and F. Ochsenbein, "The Second Reference Dictionary of the Nomenclature of Celestial Objects," *Astronomy and Astrophysics Supplement Series*, vol. 107, pp. 193–218, October 1994.

[4] E. P. Ekelman, "Radio Star Flux Density Expressions for Accurate Antenna Gain Measurements," *IEEE Antennas and Propagation Symposium*, vol. 2, pp. 1048– 1051, August 1999.

[5] J. W. M. Baars, P. G. Mezger, and H. Wendker, "The Flux Density of the Strongest Thermal Radio Sources at the Frequency 14.5 GHz," *Zeitschrift für Astrophysik*, vol. 61, pp. 134–143, 1965.

[6] Bryan J. Butler, Paul G. Steffes, Shady H. Suleiman, Marc A. Kolodner, and Jon M. Jenkins, "Accurate and Consistent Microwave Observations of Venus and Their Implications," *Icarus*, vol. 154, no. 2, pp. 226–238, December 2001.

[7] B. L. Ulich, J. H. Davis, P. J. Rhodes, and J. M. Hollis, "Absolute Brightness Temperature Measurements at 3.5-mm Wavelength," *IEEE Transactions on Antennas and Propagation*, vol. 28, no. 3, pp. 367–377, May 1980.

[8] V. A. Yefanov, A. G. Kislyakov, I. G. Moiseyev, and A. I. Naumov, "Observations of Jupiter, Venus, and Source 3C293 at 2 and 8 mm Wavelengths," *Radiofizika*, vol. 13, pp. 219–224, 1970.

[9] W. J. Welch, D. D. Thornton, and R. Lohman, "Observations of Jupiter, Saturn, and Mercury at 1.53 Centimeters," *Astrophysical Journal*, vol. 146, pp. 799–809, May 1966.

[10] W. A. Dent, M. J. Klein, and H. D. Aller, "Measurements of Mars at λ 3.75 cm from February to June, 1965," *Astrophysical Journal*, vol. 142, no. 4, p. 1685, November 1965.

[11] David Morrison and Michael J. Klein, "The Microwave Spectrum of Mercury," *The Astrophysical Journal*, vol. 160, p. 325, April 1970.

[12] R. N. Manchester, G. B. Hobbs, A. Teoh, and M. Hobbs, "The Australia Telescope National Facility Pulsar Catalogue," *The Astronomical Journal*, vol. 129, no. 4, pp. 1993–2006, April 2005.

[13] X X Newhall, E. M. Standish, and J. G. Williams, "DE 102 — A Numerically Integrated Ephemeris of the Moon and Planets Spanning Forty-Four Centuries," *Astronomy and Astrophysics*, vol. 125, no. 1, pp. 150–167, August 1983.

[14] C. H. Acton, "Ancillary Data Services of NASA's Navigation and Ancillary Information Facility," *Planetary and Space Science*, vol. 44, no. 1, pp. 65–70, January 1996.