Server and Data Models in **das2py**

SH41C-3316 **C.W. Piker, L.J. Granroth, Z. Girazian** *The University of Iowa*

github.com/das-developers/das2py

Das2py is a python wrapper for the portable *das2C* library which handles much of the implementation. Both of these open-source projects are hosted at https://github.com/das-developers. These server and dataset abstractions were originally created to support daily analysis for Voyager PWS, Cassini RPWS, Mars Express MARSIS and Juno Waves, though the generalized query system was first utilized to support investigations of the Martian ionosphere by Z. Girazian *et. al.* 10.1029/2019GL083643.

das2.org

Mars Express

Root URL

EXTERNAL CATALOG

The catalog is the seed point for standard data access. If no seed URL is specified, a default root node is loaded. Instead of using a special server, the catalog is simply a directed acyclic graph of static JSON or XML documents, pointing to child documents via redundant *fully qualified* URLs. Das2py handles network node walking and dead end recovery so that catalogs appear as nested dictionaries.

```
>>> uiowa = das2.get_catalog('tag:das2.org,2012:site:/uiowa')
>>> src = uiowa['mex']['marsis']['ne-density']['das2']
>>> print(src.info())
Key                Type   Units    Default     Values
-----------------  -----  -------  ----------  -----------------
coord.alt.maximum  float  km       10000       0 to 10000
coord.alt.minimum  float  km       0           0 to 10000
coord.sza.maximum  float  degrees  180         0 to 180
coord.sza.minimum  float  degrees  0           0 to 180
coord.time.maximum float  UTC      2016-01-01  2004-01-01 to now
coord.time.minimum float  UTC      2015-01-01  2004-01-01 to now
data.dens.enabled  bool   cm**-3   True        False
>>> datasets = src.get( {'sza':(0,90), 'alt':(0,500)} )
```

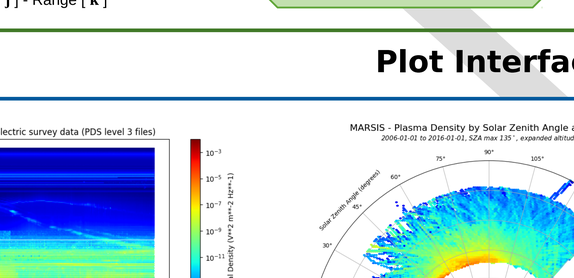
Catalog Nodes
provide redundant absolute URLs for child nodes.

Collection Nodes
identify data collections, list equivalent data sources

Source Nodes
provide call templates for backends such as CGI stream servers or files

Client-Side URI Resolution

Each node names its children. Since top nodes start with a `tag:` authority, the concatenation of all the node names along a path defines a *URI*. These path URIs are sufficient to find data sources without an explicit URL or published DOI.



Catalog Browser

https://das2.org/browse

The `interface` definitions in the Source Nodes contain enough information to automatically generate GUIs. The `protocol` definitions contain enough information to formulate HTTP GET requests.

Putting these two together, the das2 catalog browser is a single file python CGI script that can be run under Apache. It generates HTML forms that can be used to query any source in the catalog. The output varies by the backend server type.

Query Interface

QUERY RESULTS

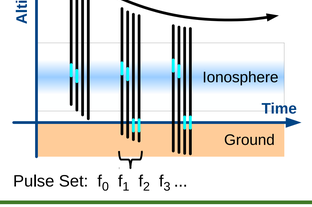
A query produces 0-N datasets. A dataset is a set of arrays that always appear homogeneous in index space. Arrays are organized by physical dimension, and the role they play in that dimension. The most common role is 'center' which defines the center point of a coordinate or measurement. Others roles include, but are not limited to: 'min', 'max', 'reference', 'offset', and standard deviation.

```
>>> print(dataset)

Dataset: 'dens_01' from group 'dens' | i:0..92410
  Property: title | MARSIS Plasma Parameter
  Property: renderer | polar

  Data Dimension : dens
    Spans:    alt, sza
    Property: label | N!De!N (cm**-3)
    Variable: dens['center'][i] (float64) cm**-3

  Coordinate Dimension : alt
    Property: label | Altitude (km)
    Variable: alt['center'][i] (float64) km

  Coordinate Dimension : sza
    Property: label | Solar Zenith Angle (degrees)
    Variable: sza['center'][i] (float64) degrees
```

Shape in index space · *Named Attributes* · *Spanning Coordinates* · *Units* · *Role* · *Storage Type*

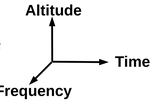
MARSIS Coordinates and Array Indexes

A sounding radar typically emits a set of pulses with increasing radio frequency and samples the return energy of each pulse as a function of time. Low frequency pulses are reflected within the ionosphere while higher frequency pulses are able to propagate to the surface and below. After a rest period, the cycle repeats. The natural storage arrays for this data set are determined by the instrument cycle.



Array Index Space
i = Radar Pulse Set Index
j = Radar Pulse Frequency Index
k = Echo Sample Index

Desired Coordinate Space



Coordinate Determination
Time = First Pulse Time [**i**] + Repeat Time [**j**]
Frequency = Pulse Frequency [**j**]
Altitude = Spacecraft Altitude [**i**, **j**] - Range [**k**]

Datasets
define high-level properties such as the overall shape in index space

Physical Dimensions
group related arrays representing aspects of physical properties

Variables
provide an array with a purpose in a dimension, such as 'center'

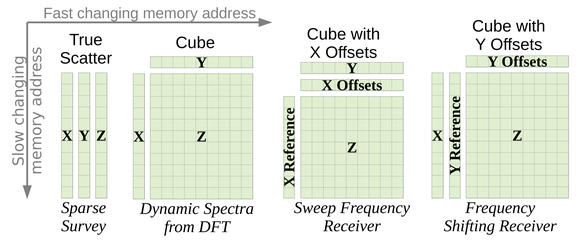
Quantities
Slicing variables produces these **astropy** objects

Why broadcast all arrays to a homogeneous index space?

Query output from das2py really is homogeneous. Meaning that if the main data array is rank 2, then the supporting data arrays also appear as rank 2, with the same index ranges. So why not use DEPEND_0, DEPEND_1, etc. in the fashion of the CDF ISTP guidelines?

- NumPy array broadcasts use no extra memory
- We are trying to support fully automatic plot generation

A smart human writing custom plotting code knows what to do with most datasets. ***Automated*** plotting software isn't as bright. Das2py makes its job easier by not conflating the spanning physical coordinates (time, frequency) with array directions (i, j) and by automatically generating 'center' arrays as needed.



Plot Interface

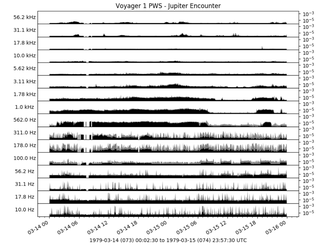
OUTPUT

The primary output is presumed to be graphics produced by matplotlib. Most scientific python users are familiar with matplotlib and know what images they wish to produce, thus automatic plotting facilities are not included in das2py, though the **`das2.mpl`** sub-module provides assistance with plot labels.

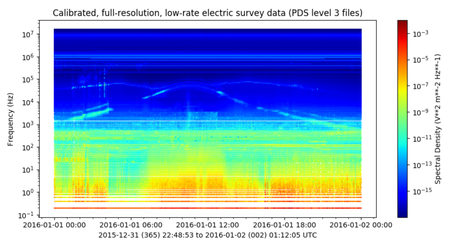
```
>>> qAlt   = dataset['alt']['center'][:]
>>> qTheta = dataset['sza']['center'][:]
>>> qDens  = dataset['dens']['center'][:]

>>> aX = (qAlt.value + radius_mars)*numpy.cos(qTheta.value)
>>> aY = (qAlt.value + radius_mars)*numpy.sin(qTheta.value)

>>> hexbin(aX, aY, qDens.value)
>>> pyplot.show()
```

In addition to generating plots, if **`spacepy.pycdf`** is on the python path, datasets can be written directly to CDF for use in outside tools such as Autoplot.

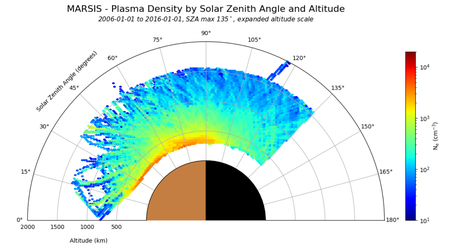
```
>>> das2.cdf.write(dataset, 'das2_query_output.cdf')
```



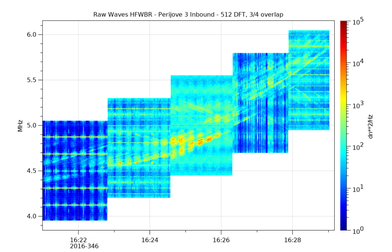
Voyager PWS
Data source with noise filter options
`tag:das2.org,2012:site:/uiowa/voyager/1/pws/specanalyzer-4s-efield`



Cassini RPWS
Data source with highly variable multi-mode output
`tag:das2.org,2012:site:/uiowa/cassini/rpws/survey`



Mars Express MARSIS
Custom CGI service, does not require time range in queries
`tag:das2.org,2012:test:/uiowa/mars_express/marsis/ne-density-planetographic`



Juno Waves
Data source with offset and reference variables, exported to CDF, rendered in Autoplot
`tag:das2.org,2012:site:/uiowa/juno/wav/survey`