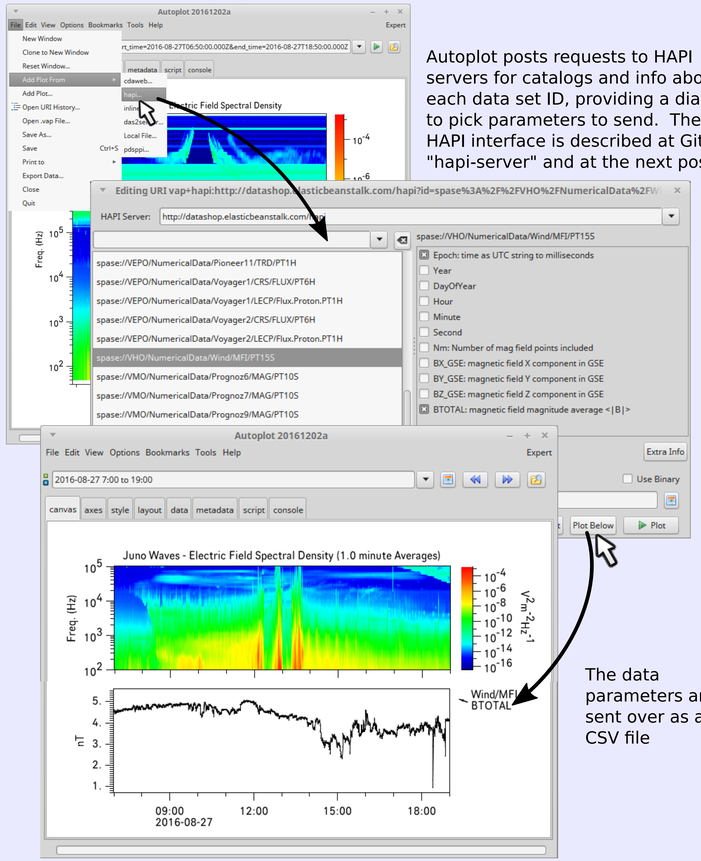
Improvements to Autoplot's HAPI Support

Jeremy Faden (Cottage Systems), Jon Vandegriff (JHU/APL), Robert Weigel (George Mason University)

What is Autoplot and What is a HAPI Server?

Autoplot is an application developed as part of the NASA Virtual Observatories, which provides graphics from data in ASCII, CDF and HDF files, and many other forms. Servers are accessible as well, providing a client which can access data from:

- OpenDAP, though this support is not regularly used
- CDAWeb, the heliophysics data base at NASA/Goddard
- PDS/PPI Node, the planetary data base at UCLA
- Das2Servers, used at the University of Iowa (and soon VESPA)
- HAPI Servers, an emerging standard API used in Heliophysics



Autoplot posts requests to HAPI servers for catalogs and info about each data set ID, providing a dialog to pick parameters to send. The HAPI interface is described at GitHub "hapi-server" and at the next poster.

The data parameters are sent over as a CSV file

Here, a HAPI server is used to provide B-Field data from WIND/MFI, plotting the data below Juno Waves data from a Das2Server at the University of Iowa.

The scientist can interact with the data, for example slicing spectrograms and adjusting axes, and then save the configuration to a .vap file and send it to a colleague, so they can look at the same graphic. The PNGWalk tool runs batch plots, and Jython scripting is used to provide a rich analysis environment.

Autoplot uses URIs to identify data sets, where a URI is a single string similar to a URL: http://autoplot.org/data/autoplot.cdf?BGSM Autoplot will download the .cdf file and plot the BGSM parameter found within.

Running Autoplot

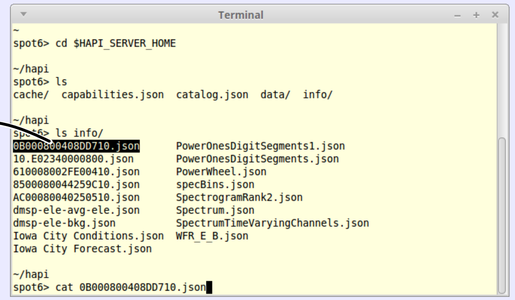
autoplot.org

Autoplot is open-source and freely available at autoplot.org. It requires Java 7 or 8 be installed, and is launched via web-start JNLP or as a single-jar file.

Autoplot Use to Implement Servers

Autoplot reads in data specified by URIs into its standard data model (QDataSet). This makes it well-suited for integration into HAPI servers. For example, one such server is a set of JSON files to configure the server, with JSON files for capabilities and catalog responses, and each info response also has a URI to load the data:

```
{
  "HAPI": "2.0",
  "createdAt": "now",
  "modificationDate": "lastday",
  "parameters": [
    {
      "length": 24,
      "name": "Time",
      "units": "UTC", ...
    },
    {
      "name": "temperature",
      "type": "double",
      "units": "deg F", ...
    }
  ],
  "startDate": "2012-01-09T00:00Z",
  "stopDate": "lasthour",
  "sampleStartDate": "lastday-P1D",
  "sampleStopDate": "lastday",
  "uri": "file:/home/jbf/1wire/data/$Y/$m/$d/0B000800408DD710.$Y$m$d.d2s"
}
```



This URI can be plotted in Autoplot

This Autoplot-based server loads the URI to fulfill data requests, converting it to CSV or binary. Many URIs support streaming, which allows the server to send data while it is read from disk, allowing it to handle long data requests. This includes "aggregations" like ...Ym$d.d2s, where the file granules are combined to form a long time series. Keywords like "now" and "lastday" (last day boundary) are resolved to ISO8601 times and the uri tag is removed to form a proper HAPI info response.

Slow reads are cached on the server side, so that responses are always interactive. A NodeJS-based HAPI-server validator is available on the GitHub site, and this HAPI server should be able to satisfy its checks. (There are a few minor problems right now, where JavaScript clients may have problems with security and large data requests.)

This server runs as a web application on a Apache/Tomcat server. A .war file containing Autoplot code and additional server code is added to Tomcat. See https://sourceforge.net/p/autoplot/code/HEAD/tree/autoplot/trunk/HapiServerDemo/

Future Work:

- HAPI Server data upload capability. Sometimes you are creating an ad-hoc dataset where you need to collaborate with others and don't care much how the data is stored, but you need team members to easily access the data. This would allow authorized clients to push data to the server so that others can download it.
- Validation of JSON files, so errors are seen before launching. Presently the server just sends out the files from the disk without checking that they are valid JSON with data that can be read by Autoplot.

Autoplot's Export-to-HAPI Feature

Autoplot's Export Data function formats data so that you can see what the responses should be from a HAPI server, for different data types like spectrograms and line plots. This can be useful to those creating HAPI servers who have data which can already be read by Autoplot. Use [menubar]->File->Export Data, select then select the name <dataset>.json, and the info response will be created along with catalog json and CSV formatted data. (This has not yet been released, but will be available immediately after AGU.)

The HAPI specification is well-documented at: https://github.com/hapi-server/data-specification

Client-Side Data Caching

The great thing about using APIs is that the scientist doesn't think about file granules, the data just magically appears. However, our experience is that this introduces a few new problems:

- Server load can be overwhelming. We use inexpensive machines to handle our modest needs (typically one to five simultaneous requests), but this is not always the case.
- Client performance. The HAPI server sends over data at full resolution, so it is essential that we preserve this work.
- Offline use. Occasionally we don't have network access (many flights and meetings).
- Data provenance. It would be nice to access data as of a particular date. This is supported by some servers, but it can also be solved on the client side with caching.

How Autoplot's Caching Works, Presently

First, the data is made into granules. A HAPI data set is partitioned into daily intervals and as the stream is loaded it is broken up into data granules by time and parameter. Cache reads are performed by combining these granules. The cache looks like this with daily data granules in monthly folders:

```
.../autoplot_data/fscache/hapi/http/jfaden.net/HapiServerDemo/hapi/data/WFR_E_B/2017/02/
        20170209.Spect.csv      20170210.Spect.csv      20170211.Spect.csv
        20170209.Time.csv       20170210.Time.csv       20170211.Time.csv
```

So the trick is to make sure the records of each .csv correspond to one another. Alternatively, the timetags could be stored along with each variable, but this roughly doubles the size of the cache and its not that hard to get it right. Also, each file will be compressed using gzip.

When can the cache be used? When caching is enabled, cache granules will have a lifespan of one hour. The HAPI "info" requests can provide caching information as well, namely the modificatonDate. When this is found, granules newer than the modificationDate will be used. Last, when Autoplot is in its "offline" mode and no network resources are available, then the cached resources are used.

Future Work:

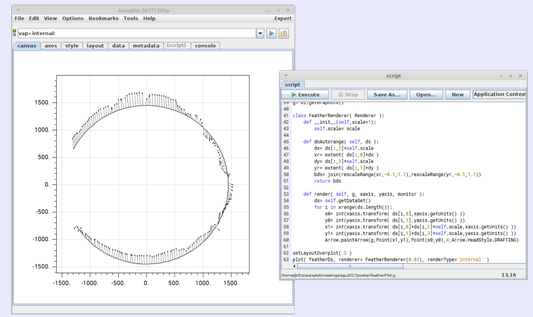
- Provenance. A GUI droplist of downloaded versions of the data would provide access to old datasets.
- Active Data Sets, which are constantly growing as data is collected. These could be cached, but the modificationDate keeps changing preventing effective caching.
- Cache unloading. Autoplot has never had a mechanism to remove data from its cache of downloaded data, and with the short lifespan (1 hour) of these granules, this should be explored.

So, in a few words:

Autoplot provides a useful and flexible client for HAPI servers.

Autoplot can be used within servers to read data and to demonstrate HAPI responses for plotted data.

Client-side data caching should recover the features of file-based data dissemination, while providing the ease of use and interoperability of servers.



Autoplot's scripting loads data from a B-field data from a HAPI server, a position from a Das2Server, and adds a custom code to render the data on to the plot.

This work supported by NASA Grant NNG16PX40C