Supporting Open Science with Data Access Standards like HAPI

Jeremy Faden[1], Jon Vandegriff[2], Alex Antunes[2], Robert S. Weigel[3], Douglas Lindholm[4], Robert Candey[5]
1. Cottage Systems, 2. Applied Physics Laboratory, 3. George Mason University, 4. Laboratory for Atmospheric and Space Physics, 5. NASA/Goddard

Abstract

One of the assumptions of open science is that data is freely and easily accessible to everyone. A key aspect of democratizing access is reducing the data wrangling load required to access data from a new source or a slightly different community outside your area of expertise. The HAPI (Heliophysics Application Programmer Interface) specification is an access standard that simplifies low-level data ingest, and has also been adopted across multiple domains (Planetary, Heliophysics, including Space Weather). HAPI offers a path towards FAIR access by baking FAIR principles into the standard, such as open access to data with no login required. We will present the key features of HAPI and how they relate to open science and the FAIR principles, and will showcase the suite of open source tools that exist or are being developed to support the Heliophysics and Planetary science communities.

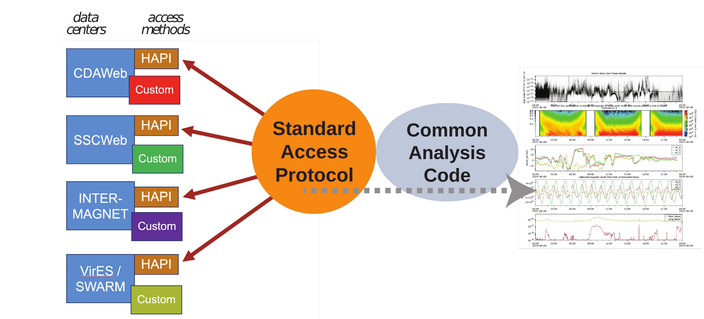
The problem HAPI tries to solve

When the HAPI project was begun, data was found on various websites in various file formats. Different fields of study use different file formats, and usually in Heliophysics, CDF and ASCII files are used to store time series data. Other fields use NetCDF. Often, comma-separated-value (CSV) files are sufficient to represent the structure of the data, and it seems reasonable to expect consumers of data to write a small program to ingest the data into their analyses.

Other groups would use web interfaces to provide access to their data, setting up a website with a custom API (Application Programmer Interface) they've defined and maybe writing a library that knows how to use it. Many different APIs exist, each needing new client software to access the data for analysis. This software is written in just one or two languages and often only just enough to access the data for analysis, leaving some of the server capabilities unused. The next person who uses this new client finds the client code under-implemented and is forced to complete the implementation for themself.

This is an inevitable condition, but when new software is needed to access data, development slows and software programmers are needed in the science analysis process.

We decided we should define an interface which we would call "HAPI," which tries to accomplish the a common functionality with one interface. This would be added to existing servers:



One HAPI team

HAPI is a group of people with a common interest in transmitting data from a server to clients. We've all come up with our own ways of doing this in the past, and we all realize that to come up with a simple but generally agreeable interface is productive. We have been willing to meet weekly and carefully work out ideas into a well-documented and widely implemented protocol. `https://github.com/hapi-server` contains specifications for each HAPI version released.

What we've found is that by doing this, our protocol is better supported and more complete than any of our other server implementations. It may not be optimal for the data any one of us serves, but it's good enough that it can be supported. From the client perspective, there might be things they would want, but they have to prove to themselves that this is really needed, and be ready to make the argument that the protocol needs extension.

Many servers and clients

The problem with setting up a server to provide your data is that you are assuming that people will write code to read data from your server. HAPI provides easily-implemented interfaces, and many languages have clients written for them already. All of these are open-source and may be used freely:

Servers:
- Java
- NodeJS
- Python

Clients:
- Python
- IDL
- Matlab
- Java
- JavaScript

Applications:
- Autoplot
- SPEDAS and PySPEDAS
- hapi-server.org/servers
- wget/curl

Introduction to HAPI

HAPI is a protocol for providing time-series data to a community, short for the Heliophysics Application Programmer's Interface. We often have long time series of the same type of data collected over years and wish to be able to freely browse any portion of the data. Using HAPI, clients can discover what servers are available, what datasets are available on a server, and what parameters are available within each dataset. Once the dataset is identified, any time interval of the data can be loaded.

What HAPI servers are available?

```
https://github.com/hapi-server/
   servers/blob/master/all.txt
```

What datasets are available at this server?

```
https://cdaweb.gsfc.nasa.gov/hapi/catalog
```

What parameters are available within a dataset?

```
https://cdaweb.gsfc.nasa.gov/hapi/info
   ?id=OMNI_HRO_5MIN
```

Each of those responses is a well-documented, JSON-formatted object containing information about the labels and units of the data, what time span for when the data is available, and the type of each data parameter. These might be scalars, vector components, or spectra.

The data is then requested, and the response will be a comma-separated values (CSV) file containing the requested columns. Retrieving the URL

```
https://cdaweb.gsfc.nasa.gov/hapi/data?id=OMNI_HRO_5MIN
    &start=2023-10-01T00:00Z&stop=2023-10-31T00:00Z
    &parameters=T,E,Beta
```

will return a CSV where each line contains a time tag in ISO8601 format, then the parameters T, E, and Beta. Many servers support an optional binary response, but CSV responses are always available.

Validation

- A validator tool scans through a HAPI server performing hundreds of tests, ensuring correctness.
- JSON schemas are a standard way to ensure JSON responses are conforming, speeding up server development

Metadata Handling

HAPI has metadata which leverages off of existing work:

- `unitsSchema` declares the schema to use. For example, the ESA Cluster mission has a standard set of units notation used in data files.
- `coordinateSystemSchema` element works the same way. SPASE defines a set we can use.
- `additionalMetadata` object can contain lots of metadata, and has a schemaURL which describes it. Software like Autoplot can then recognize the schema and harvest more information.
- Info responses may also include ORCIDs, SPASE IDs, and DOIs.
- `resourceURI` and `resourceID` point to additional areas which describe the resource more fully.

HAPI is FAIR

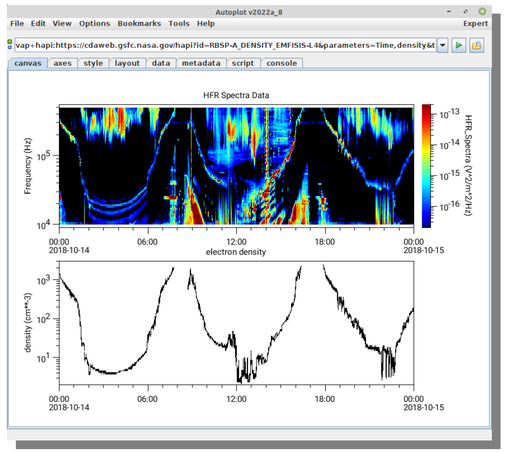
Findable - HAPI allows a uniform way to identify the datasets at multiple institutions

Accessible - HAPI has a very simple access interface

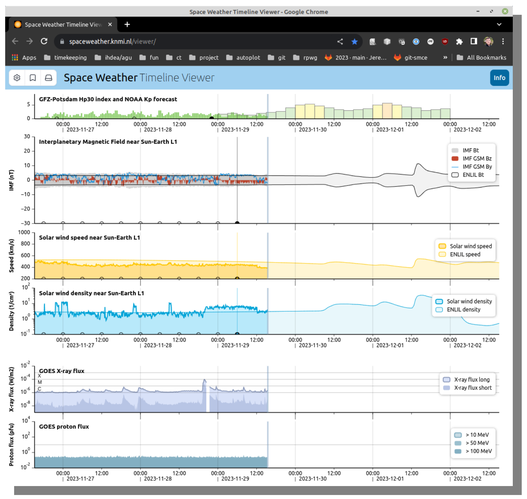
Interoperable - as a standard, HAPI offers uniform access across many data centers covering different science domains

Reusable - HAPI supports the sharing of durable REST-ful links to data; these links can be used to communicate a specific selection of data in a very precise way

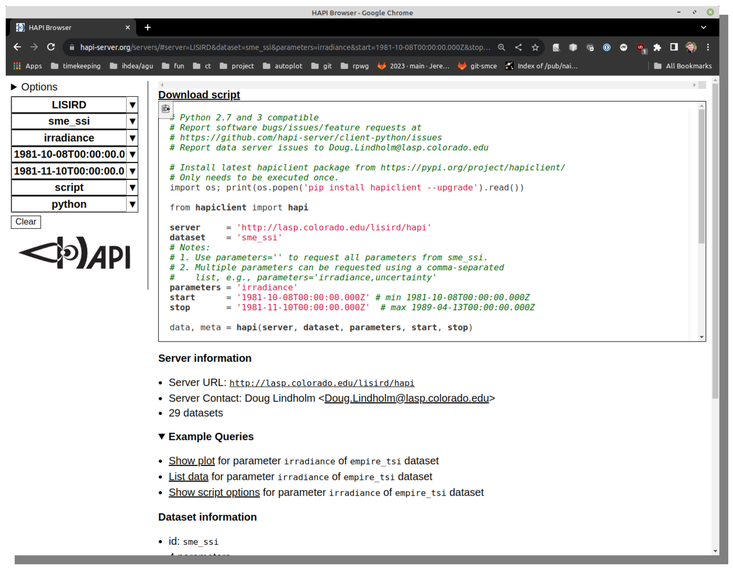
Autoplot is a desktop application maintained by Jeremy Faden. It works directly with HAPI servers. Groups setting up a HAPI server can use it to test the server before it's made visible to the public. The location of the HAPI server is entered into Autoplot's address bar, and it will show what data the HAPI server provides and the parameters with each data set.



Eelco Doornbos' Space Weather Timeline Viewer is at https://spaceweather.knmi.nl/viewer/. This web site was developed independently of our group. It uses a JavaScript graphics library to provide interactive plots of data served by HAPI servers.



https://hapi-server.org/servers, created by Bob Weigel, displays data available from each of the discoverable HAPI servers listed in all.txt and dev.txt on the github site. JavaScript is used to browse the selected server's datasets, and server-side Python is used to provide displays of parameters, or data can be downloaded from the site. The site also provides code generation showing how the data can be loaded into the scientist's Python script.



Conclusions

The HAPI server protocol is a simple interface providing access to time series data sets. Existing servers have been adapted to HAPI, making them accessible in more use cases. Standard clients are available in many languages, so scientists can start working with data from servers immediately using proven and efficient code.

HAPI has always been designed with openness and standards in mind, using JSON responses with schemas describing them. Data responses are simple CSV that is easily parsed by most programming languages and spreadsheet programs, and many open-source clients are already available. Standards like SPASE, NetCDF CF, ORCID IDs, and DOIs are encouraged. HAPI tries to make it easy and convenient to use these standards, and we hope that more people will do so.