

Autoplot Update for IHDEA 2021

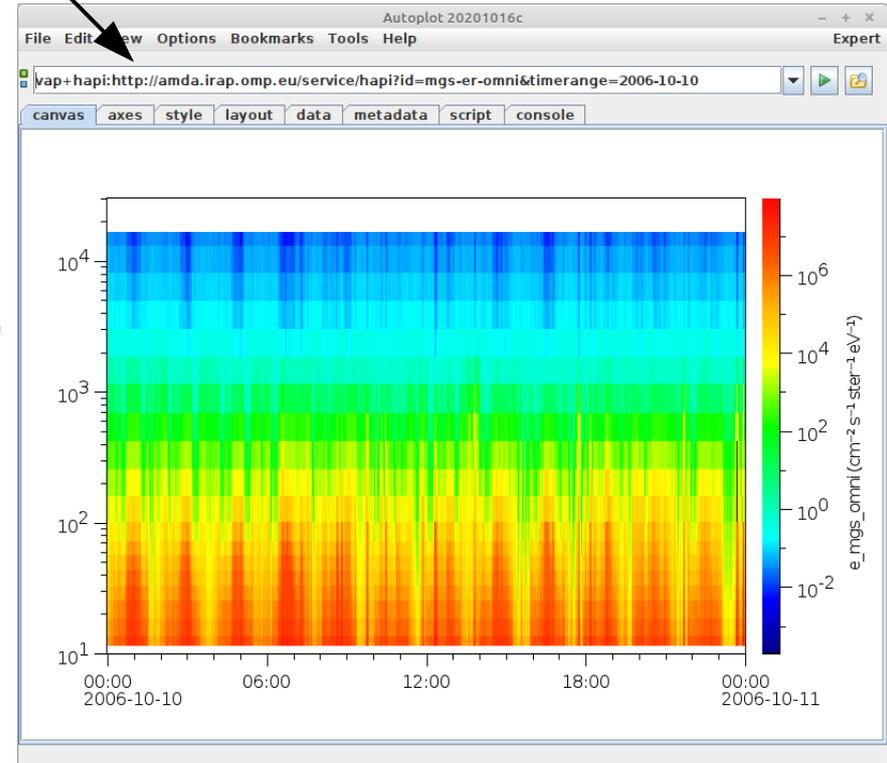
Jeremy Faden
faden@cottagesystems.com

Autoplot Overview

- Autoplot is a Java application for display of data, reading in data from many different file types and data servers.
- Data sources are identified by “Autoplot URIs” which are entered in the address bar at the top.
- Data is then displayed as spectrograms, lineplots, etc, depending on its structure.
- GUI editors allow scientists to create URIs.
- Lots of interactive analysis is be done, such as zooming in, scanning through a data set, and slicing data.

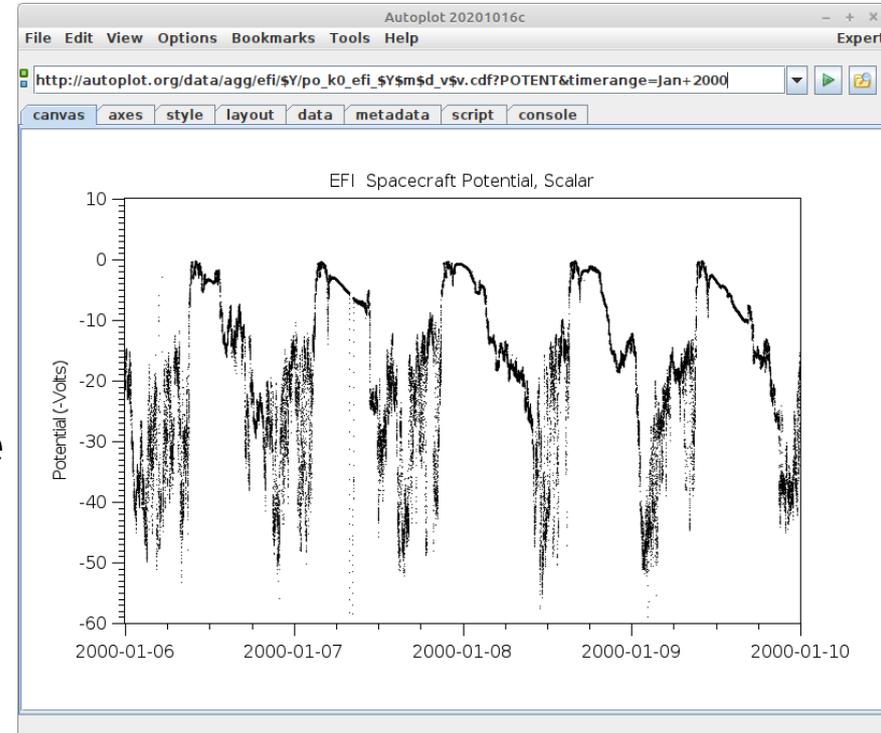
Autoplot URI

vap+hapi:http://amda.irap.omp.eu/service/hapi?id=mgs-er-omni&timerange=2006-10-10



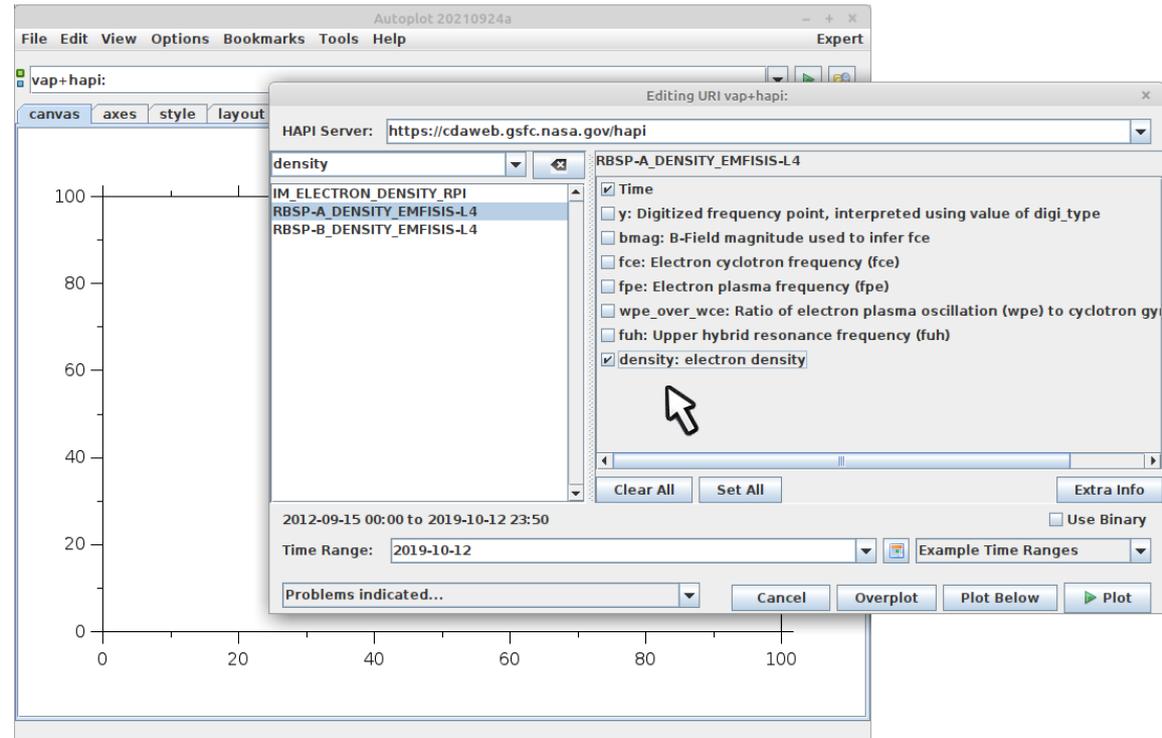
Autoplot Overview – Aggregation

- The URI can be a file on a remote server, and it will be downloaded and cached automatically.
- Autoplot can also take an “aggregation” template specification for filenames, like `http://autoplot.org/data/agg/efi/$Y/po_k0_efi_$Ymd_v$v.cdf?timerange=Jan+2000`
- The website is listed, and each file’s time range is interpreted.
- The files within the time range are read in and aggregated.



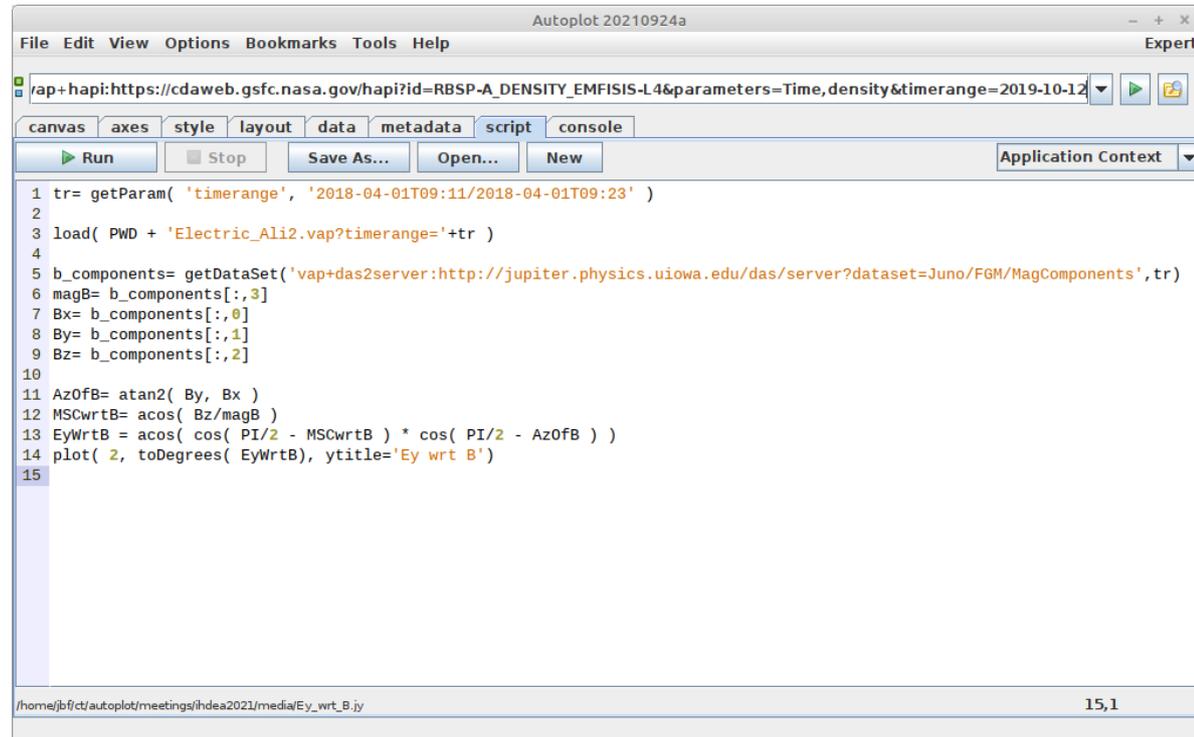
Autoplot Overview – HAPI Server Support

- Autoplot uses custom client software to talk to servers like CDAWeb, PDS-PPI, and Das2Servers
- Support for HAPI servers
 - Autoplot is part of HAPI development
 - Supports ‘bleeding edge’ features
 - Works around mistakes in implementations to assist teams setting up servers, provides feedback on log console.
 - Anyone can set up a HAPI server, but there is a registry of HAPI servers Autoplot will propose.



Autoplot Overview – Scripting

- Autoplot includes Jython scripting (where Jython is an old version of Python implemented in Java)
- Jython scripts are used extensively
 - Create custom applications (interactive digitizer)
 - Perform data handling (bulk CSV to CDF conversion)
 - Read new file formats (RadioJove SPS file)



The screenshot shows the Autoplot 20210924a application window. The 'script' tab is active, displaying a Jython script. The script defines a time range, loads data from a NASA HAPI endpoint, retrieves magnetic field components (Bx, By, Bz) from a Juno dataset, calculates the angle of the magnetic field (AzOfB), and plots the angle (EyWrtB) against time. The script is as follows:

```
1 tr= getParam( 'timerange', '2018-04-01T09:11/2018-04-01T09:23' )
2
3 load( PWD + 'Electric_Ali2.vap?timerange='+tr )
4
5 b_components= getDataSet( 'vap+das2server:http://jupiter.physics.uiowa.edu/das/server?dataset=Juno/F6M/MagComponents', tr )
6 magB= b_components[:,3]
7 Bx= b_components[:,0]
8 By= b_components[:,1]
9 Bz= b_components[:,2]
10
11 AzOfB= atan2( By, Bx )
12 MSCwrtB= acos( Bz/magB )
13 EyWrtB = acos( cos( PI/2 - MSCwrtB ) * cos( PI/2 - AzOfB ) )
14 plot( 2, toDegrees( EyWrtB ), ytitle='Ey wrt B' )
15
```

The status bar at the bottom of the window shows the file path: /home/jb/ct/autoplot/meetings/hdea2021/media/Ey_wrt_B.jy and the page number 15,1.

Useful Libraries coming out of Autoplot

- Autoplot is 750,000-lines of Java code delivered in 30MB jar file.
- I've been extracting libraries from there to make functionality useful to other Java codes
- TimeUtil library for managing time
 - ISO8601 parsing and formatting of times and time ranges
 - Previous midnight, next day, day-of-year, etc.
 - <https://github.com/hapi-server/uri-templates/blob/master/UriTemplatesJava/src/org/hapiserver/TimeUtil.java>
- Stand-alone HAPI client <https://github.com/hapi-server/client-java>
 - Up-to-date with the HAPI feature set
 - Supports streamed responses, so codes can process data as it's received.
 - Client-side caching, where repeat reads are fast and cache data is kept up-to-date. Python client will use the same cache.
- URI_Templates
 - Autoplot's aggregation feature based on this, format and parse strings like `Ym$d_$v.dat`
 - <https://github.com/hapi-server/uri-templates>

Pure-Java CDF Library

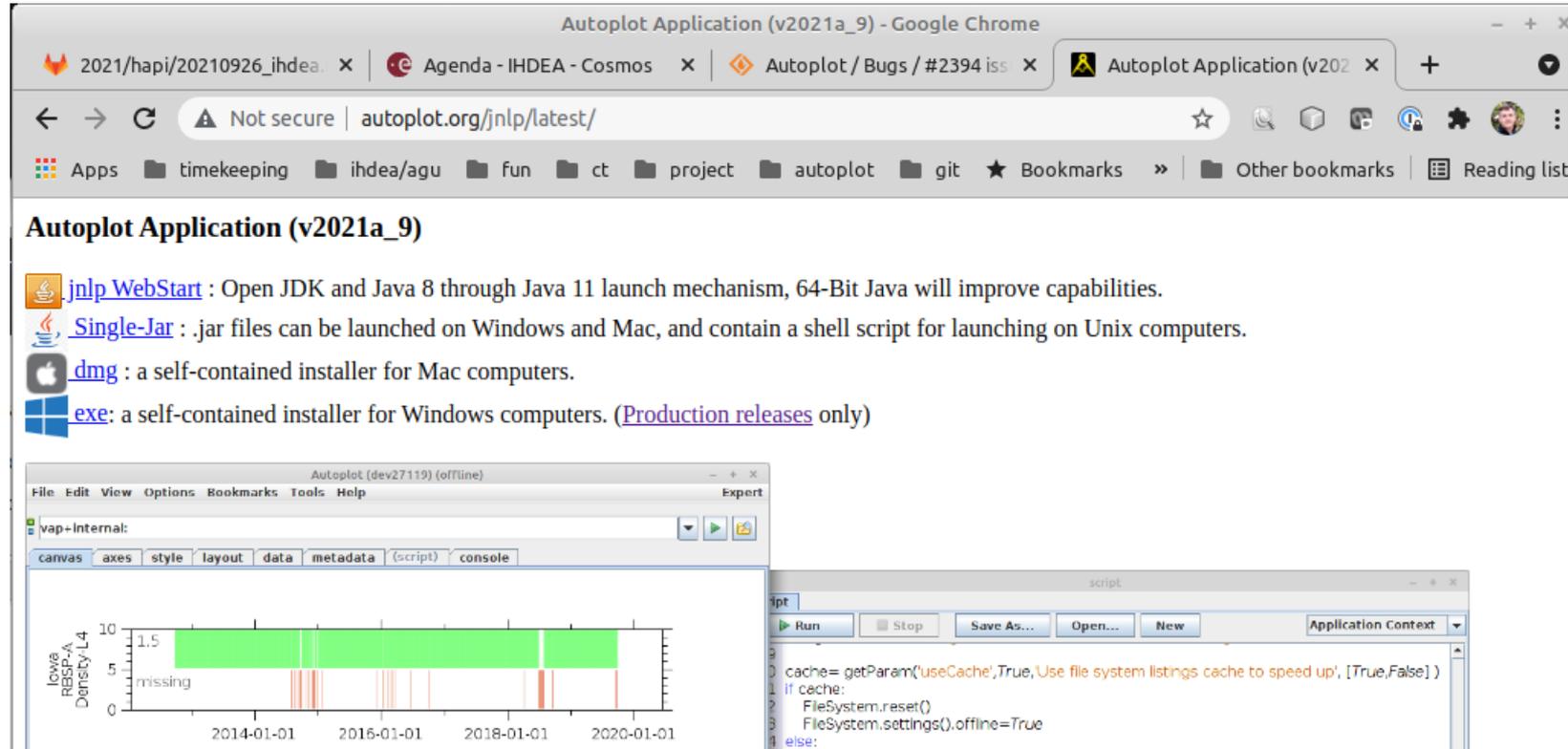
- Nand Lal's CDF library was moved to Github to be maintained under Autoplot's area (for now).
- Jeff Maxwell at Uptake (a private US company) is using it for applications.
- He modernized the project, adding unit testing and modern build systems.
- With Nand's passing this last year, Jeff and I will continue to curate it, in coordination with the CDAWeb team.
- <https://github.com/autoplot/cdfj>

Build and Distribution moved to U. Iowa Site

- autoplot.org was an Amazon AWS node
- To simplify the site and allow it to be moved to a new host, the software distribution was moved to U. Iowa servers
- This has been almost transparent to the scientists who use Autoplot
- The mechanism for launching Autoplot with CDF file CDAWeb with WebStart uses has been problematic. (I think this still has issues?) With WebStart going away, I wonder if we should look at using the SAMP Hub for this.

Build and Distribution moved to U. Iowa Site

- New launch page encourages use of non-WebStart launch methods (.dmg for Macs, .exe for Windows) that work more reliably.
- 64-bit Java is assumed, 32-bit still supported.



The image shows a screenshot of a web browser displaying the Autoplot Application (v2021a_9) website. The browser's address bar shows the URL `autoplot.org/jnlp/latest/`. The website content includes the following text:

Autoplot Application (v2021a_9)

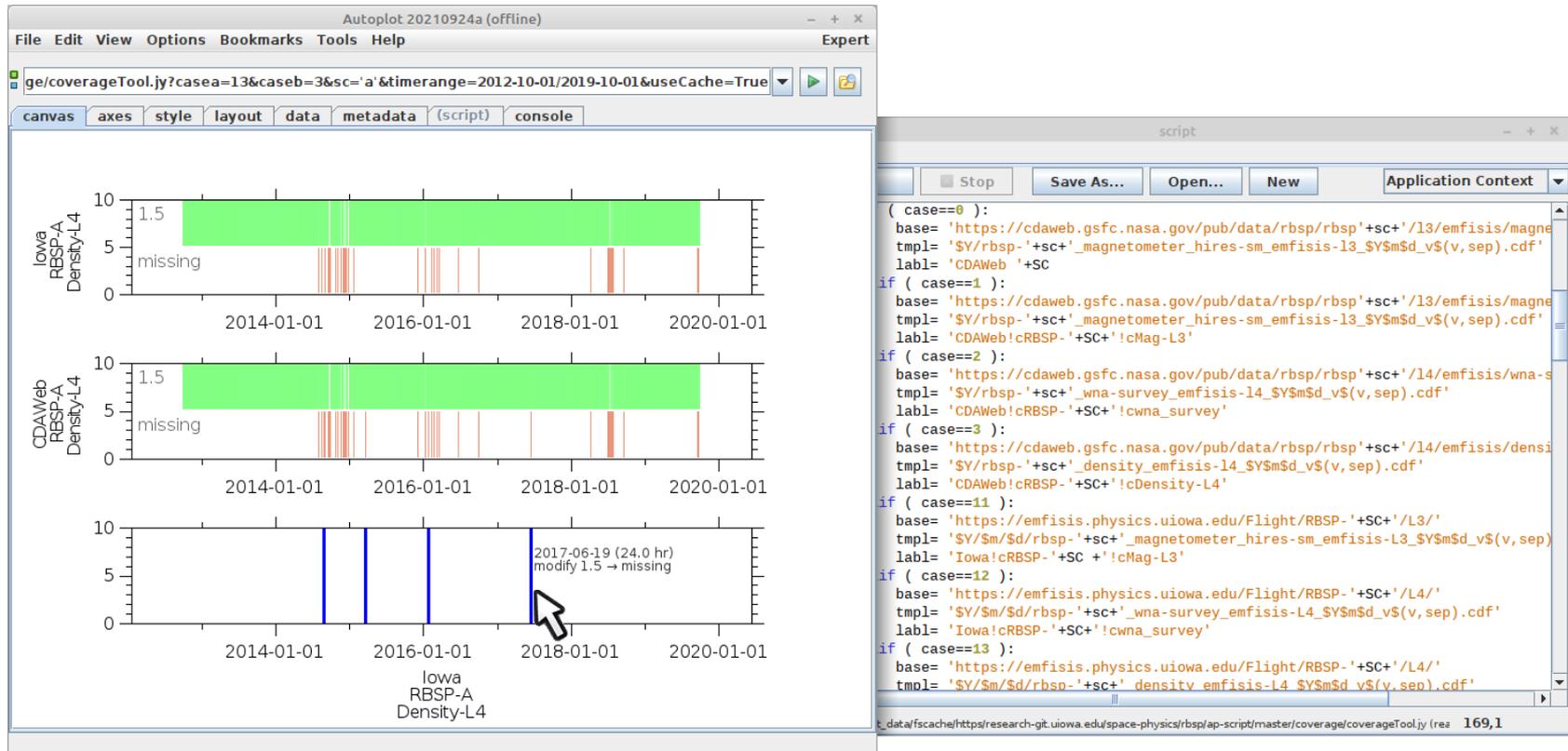
- [jnlp WebStart](#) : Open JDK and Java 8 through Java 11 launch mechanism, 64-Bit Java will improve capabilities.
- [Single-Jar](#) : .jar files can be launched on Windows and Mac, and contain a shell script for launching on Unix computers.
- [dmg](#) : a self-contained installer for Mac computers.
- [exe](#): a self-contained installer for Windows computers. ([Production releases](#) only)

Below the text, there are two overlapping windows from the Autoplot application. The top window, titled "Autoplot (dev27119) (offline)", shows a plot of "Lowest RBSP-A Density L4" over time from 2014 to 2020. The plot features a green shaded area representing a range of values and vertical red lines indicating specific data points. The bottom window, titled "script", shows a code editor with the following code:

```
1pt
2 Run Stop Save As... Open... New Application Context
3 cache= getParam('useCache',True,'Use file system listings cache to speed up', [True,False] )
4 if cache:
5     FileSystem.reset()
6     FileSystem.settings().offline=True
7 else:
```

Example Scripts – File Coverage

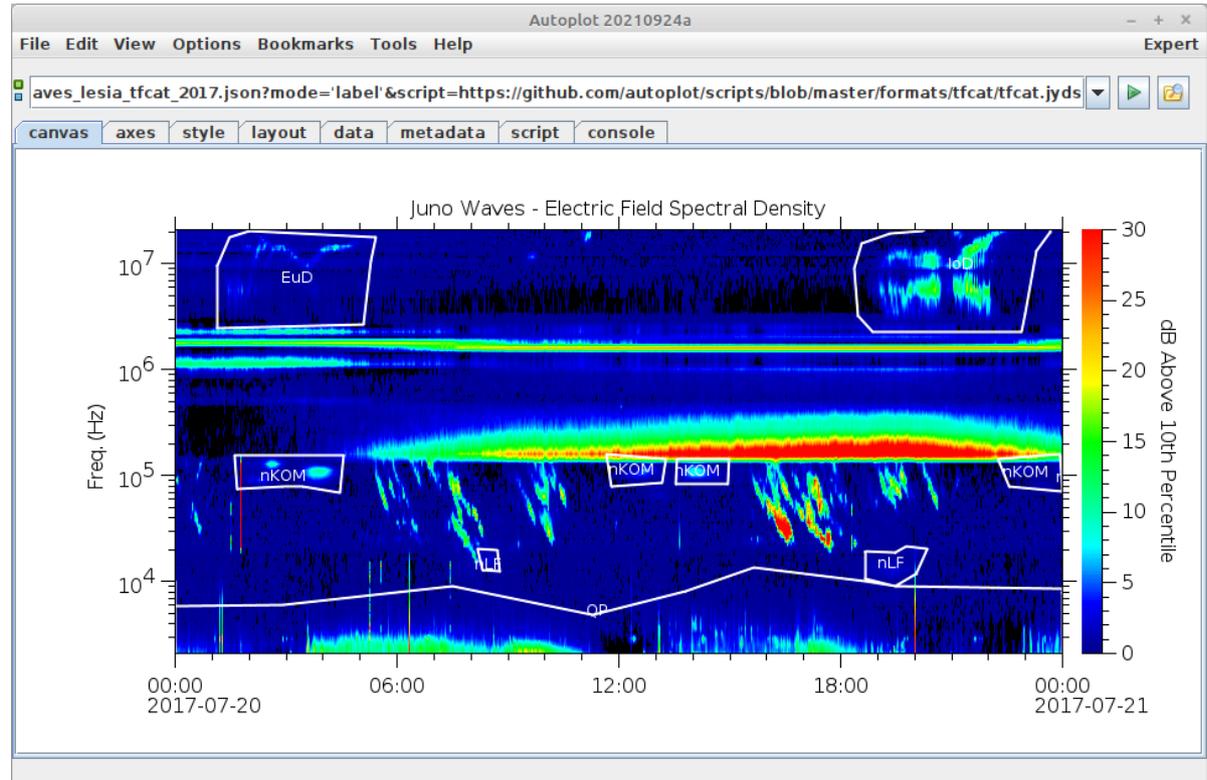
Bobby at CDAWeb may use a script I made for the RBSP-EMFISIS team, which uses aggregation ($\$Y\$m\$d\dots$) to compare data products at two sites.



Example Scripts – TFCat Reader

Baptiste and Corentin are using script to read new TFCat formatted JSON files.

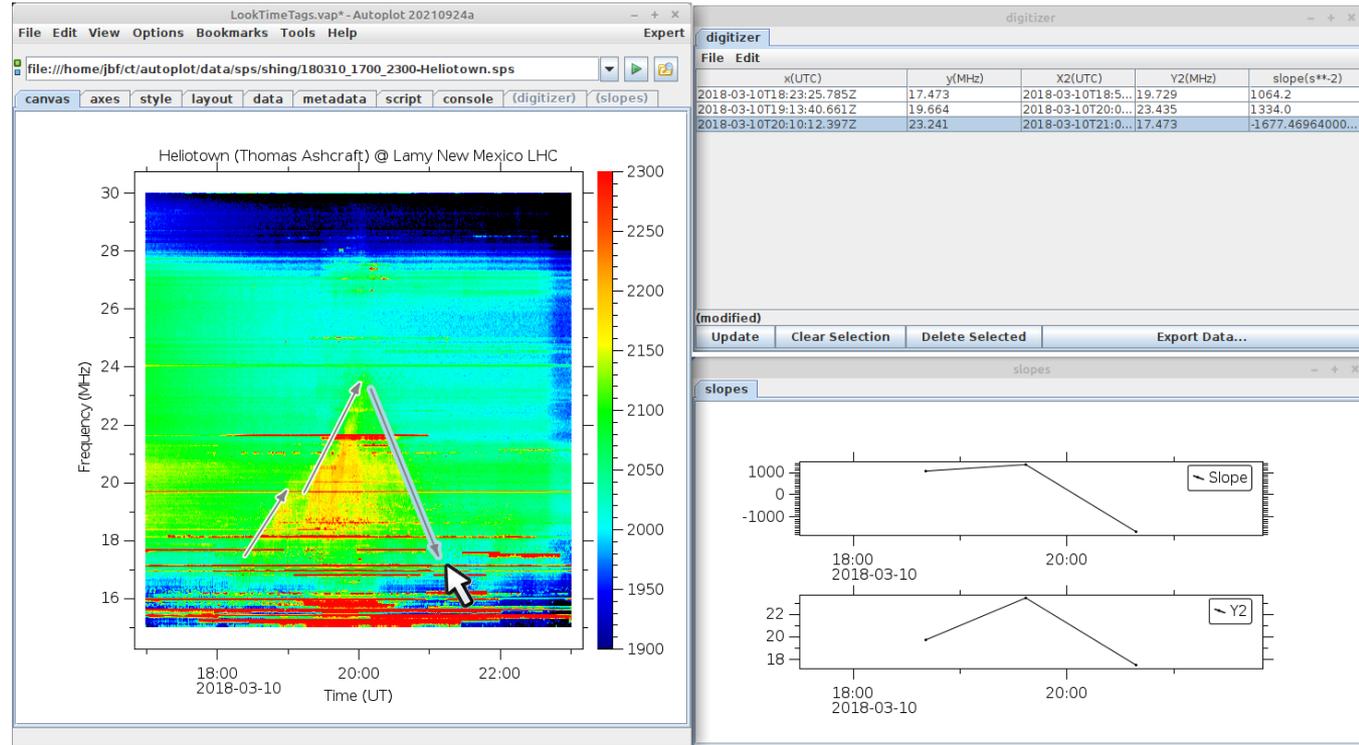
This script could be built-in to Autoplot so that the files are recognized on any running Autoplot.



Example Scripts – Digitizing

Shing is using a script for digitizing RadioJove data, which implements a workflow so the human operator can quickly process features.

The RadioJove reader is itself a script which is built-in to Autoplot.



GitHub/GitLab Support

Note that scripts are often maintained at GitHub or at a GitLab instance. Das2Java's FileSystems API allows these server types to be "mounted" and files retrieved from them.

This allows the scientists share a script without having to worry about moving it back and forth, and GitHub/GitLab provide editors so the script can be modified directly, right on the websites.

The screenshot displays a web browser window with a GitHub repository page for 'autoplot/scripts'. The repository is public and has one contributor, 'jbfaden', who updated the file 'readTypeSps.jyds'. The file size is 6.92 KB and contains 200 lines (169 sloc). The code is shown in a light blue editor with line numbers 1 through 14. The code includes imports for 'LoggerManager' and 'setScriptTitle', and uses 'getParam' to retrieve 'resourceURI' and 'antenna' values. A 'RadioJove SPS Reader' dialog box is overlaid on the code, showing the 'resourceURI' and 'antenna' fields. The 'resourceURI' field is set to 'http://www-pw.physics.uiowa.edu/~jbf/autoplot/jyds/formats/radiojove/1522660086180402060...' and the 'antenna' field is set to '0'. The dialog box also has a 'Select variable' dropdown and buttons for 'Cancel', 'Overplot', 'Plot Below', and 'Plot'. The 'Plot' button is highlighted in green. The background shows a plot of 'Time (UT)' with data points at 06:10, 06:20, and 06:30 on 2018-04-02.

GitHub/GitLab Support

The RadioJove data is read into Autoplot using a script that Masafumi Imai wrote and maintains. Autoplot sees the .sps file extension and uses the script, grabbing a current version from the website.

Currently the GitHub/GitLab support doesn't recognize instances automatically, so people must provide the server URL (e.g. <https://github.umn.edu>) and I have to hard-code it in. Also, private projects are not accessible to Autoplot, so only public projects are supported. I hope to fix both of these problems this coming year.

The screenshot displays the Autoplot software interface. On the left, a web browser window shows the GitHub repository page for `autoplot / scripts`. The repository is public and has one contributor, `jbfaden`, who updated `readTypeSps.jyds`. The file size is 6.92 KB and it contains 200 lines of code. The code snippet shown is:

```
1 from org.das2.util import LogManager
2 setScriptTitle( 'RadioJove SPS Reader' )
3
4 logger= LogManager.getLogger( 'apdss.radiojove.sps' )
5
6 resourceURI= getParam( 'resourceURI', 'http://www-pw.physics.uiowa.edu/~jbf/autoplot/jyds/formats/radiojove/1522660086180402060' )
7
8 antenna= getParam( 'antenna', 0, 'the antenna to load when multiple dual antenna file is used.', { 'values':[0,1], 'labels':['0'
9
10 logger.fine( 'resourceURI= %s' % resourceURI )
11 logger.fine( 'antenna= %d' %antenna )
12
13 oldSlow= 'F'
14
```

On the right, the 'RadioJove SPS Reader' dialog box is open. It has a title bar 'RadioJove SPS Reader' and a menu bar 'File Edit View Options Bookmarks Tools Help'. The dialog contains the following text:

resourceURI, example file to load:

antenna, the antenna to load when multiple dual antenna file is used.:

Below this, there is a section: 'Select from the variables calculated by the script, 'data' or 'result' is used by default:'. The 'Select variable:' field is empty. At the bottom of the dialog, there are buttons: 'Cancel', 'Overplot', 'Plot Below', and 'Plot'. A plot area is visible at the bottom of the dialog, showing a time axis from 06:10 to 06:30 on 2018-04-02, with a single data point at 06:30 labeled 'Time (UT)'. The status bar at the bottom of the dialog says 'ready'.

New Formats

It's fairly simple for me to add support for new formats (like RINEX mentioned on Wednesday, for example) and people who would like to use Autoplot with their data should feel welcome to ask. Baptiste's team asked me to add support for TFCat, and I was able to provide the function to them in about an hour.

Once the format is added, all of Autoplot's functionality is available with that data.

Coming this next year...

- Autoplot has a built-in HAPI server I use for testing, and I plan to make a free-standing Java version of the HAPI server. This software might be useful to the Madrid ESAC team, for example.
- Office hours, inviting people to Zoom in to ask questions. Of course people can email me, but I think this might bring in some new people.
- Monthly show-and-tell Zoom meetings, where people can ask questions, share things that they've been working on, and I can show new features.
- Hold Zoom classes for groups requesting a live introduction to the Autoplot. I wrote a new class outline and did this with U. Minnesota Space Physics people earlier this week.

Thanks!

Jeremy Faden

faden@cottagesystems.com