viresclient

A new Python package for interacting with VirES

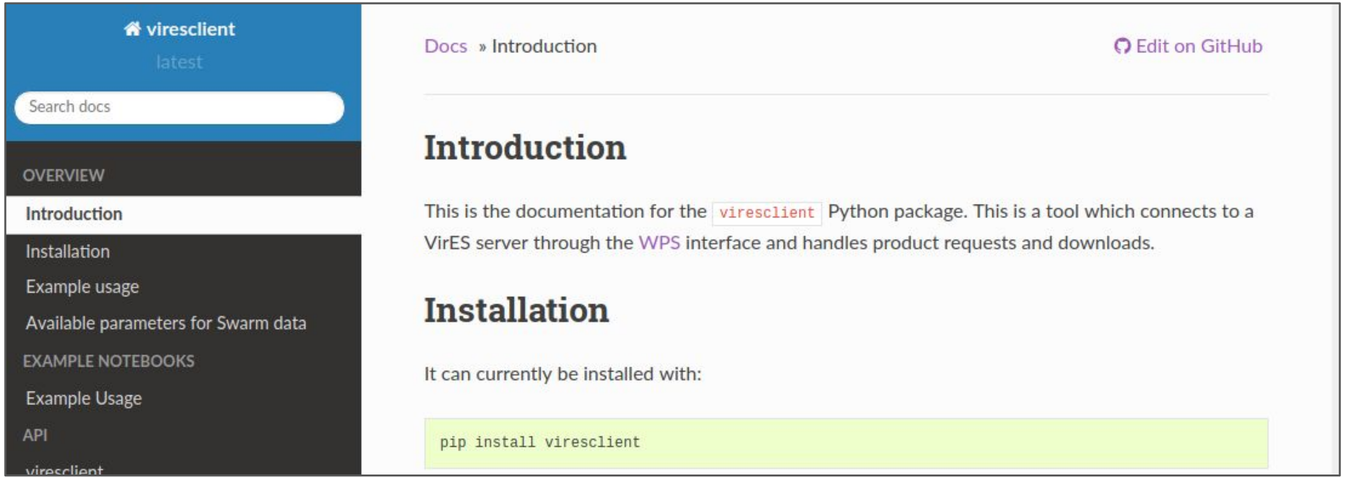
Ashley Smith

Overview

- VirES architecture & client concept
- Example usage
- Future development

Introduction

- Simple programmatic access to custom Swarm datasets
  - Output as a CDF/CSV file
  - Output as Python objects: pandas.DataFrame, xarray.Dataset
- Open source
  - https://github.com/ESA-VirES/VirES-Python-Client



Introduction

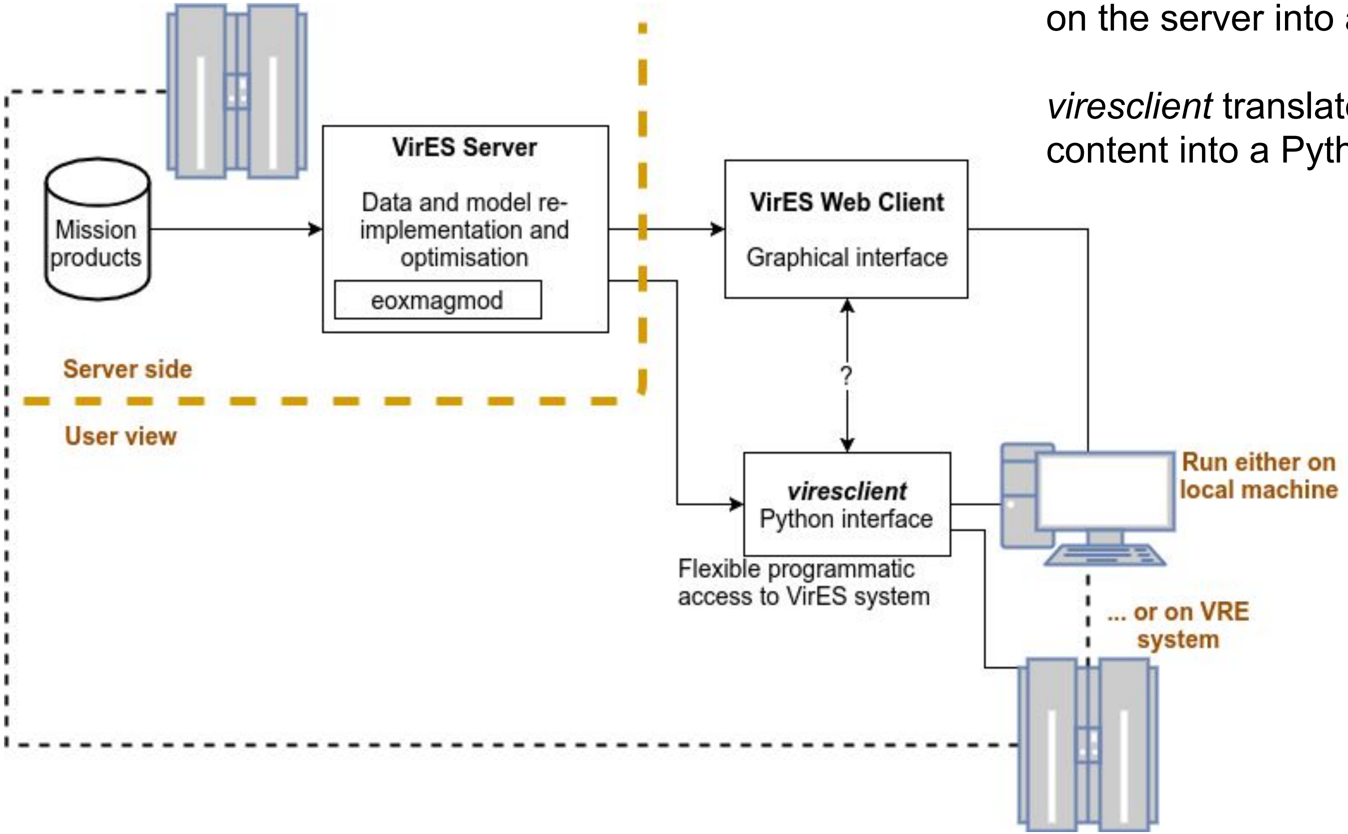
This is the documentation for the `viresclient` Python package. This is a tool which connects to a VirES server through the WPS interface and handles product requests and downloads.

Installation

It can currently be installed with:

```
pip install viresclient
```

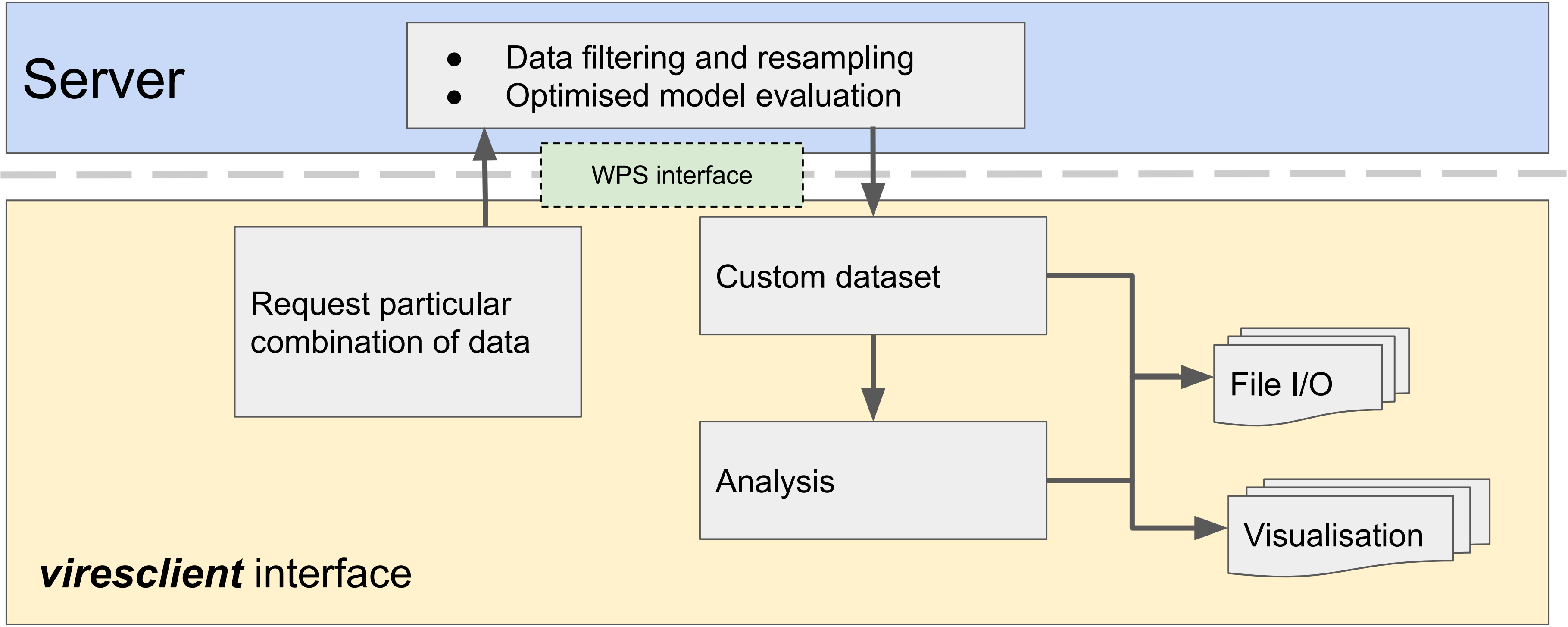
Server-Client model



The web client translates content on the server into a GUI

viresclient translates the same content into a Python interface

Server-Client model



Server

- Data filtering and resampling
- Optimised model evaluation

WPS interface

Request particular combination of data

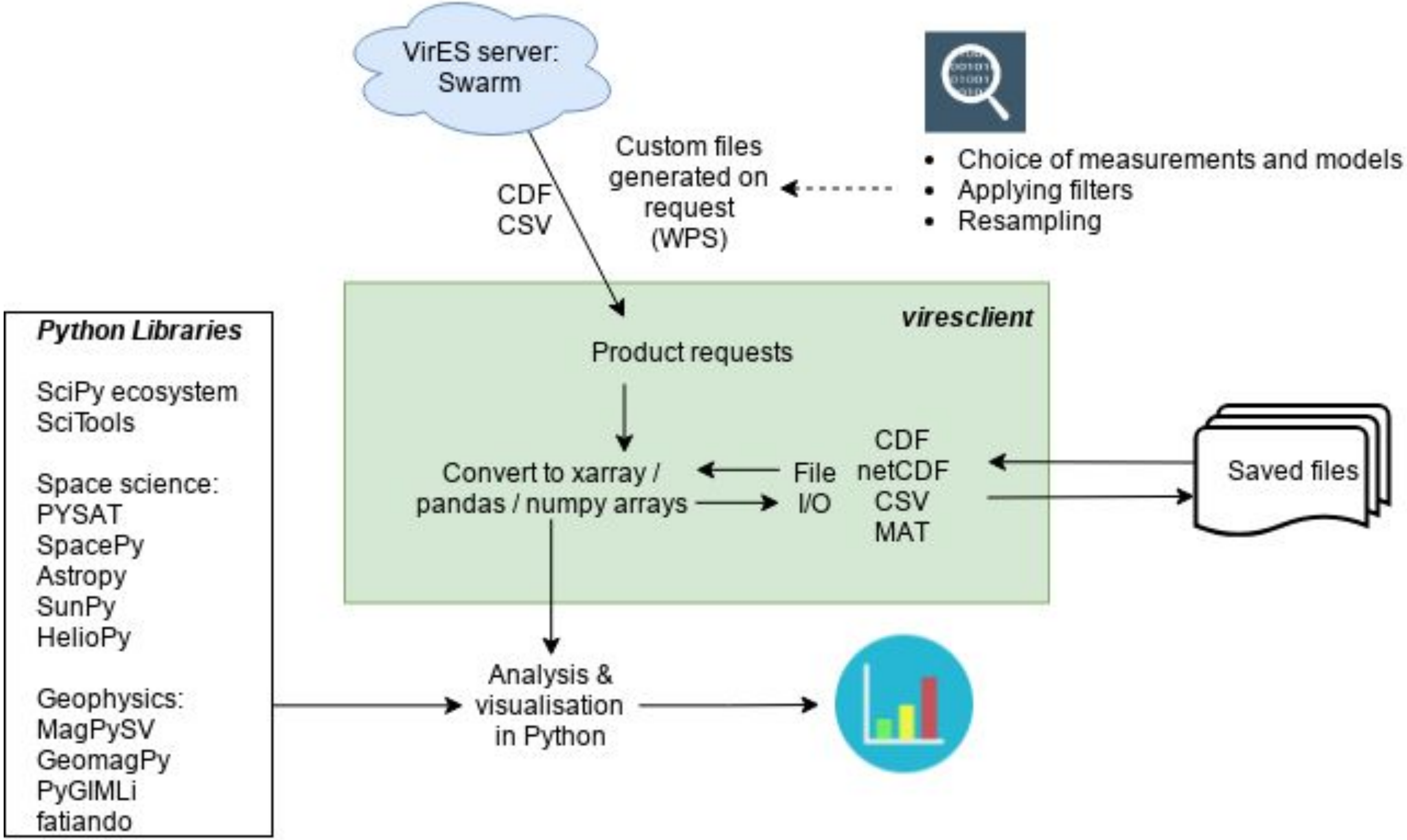
Custom dataset

Analysis

File I/O

Visualisation

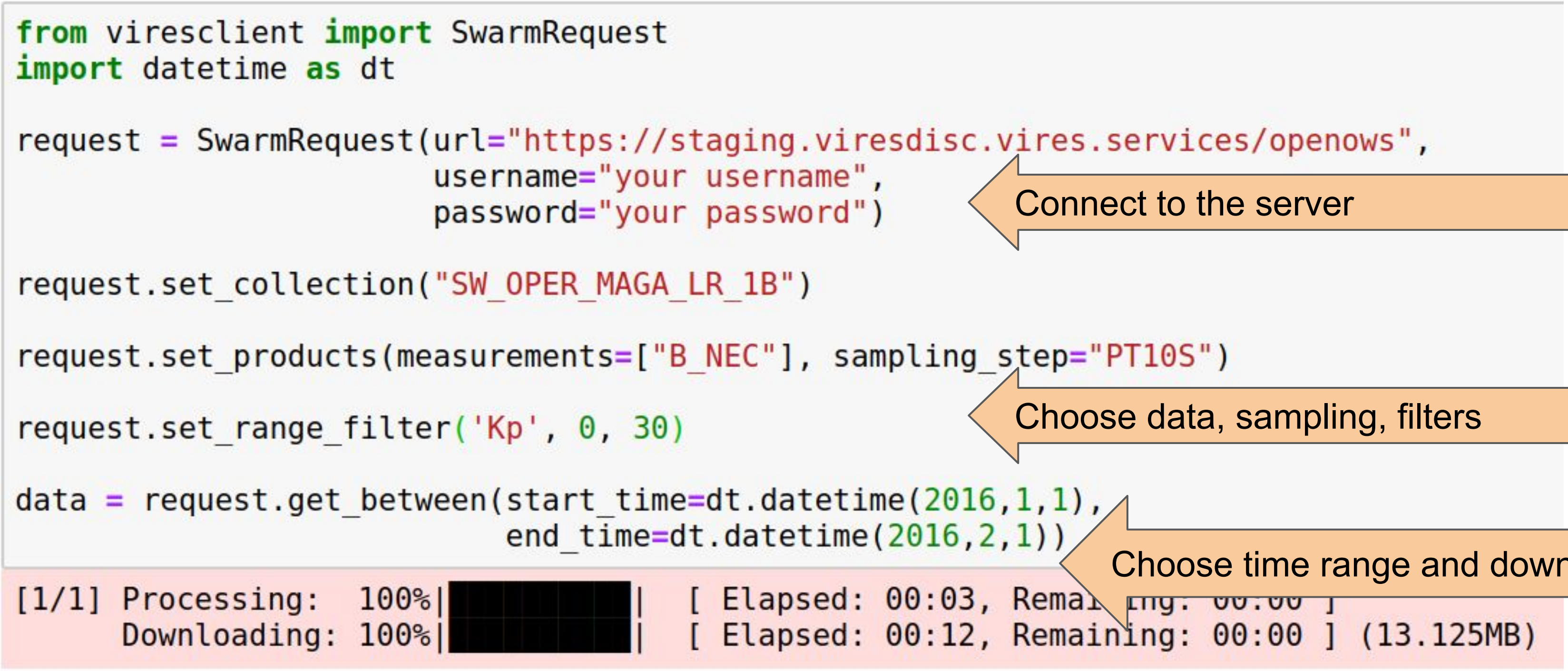
viresclient interface



VirES server:
Swarm
CDF
CSV
Custom files generated on request (WPS)
Choice of measurements and models
Applying filters
Resampling
viresclient
Product requests
Convert to xarray / pandas / numpy arrays
File I/O
CDF
netCDF
CSV
MAT
Saved files
Python Libraries
SciPy ecosystem
SciTools
Space science:
PYSAT
SpacePy
Astropy
SunPy
HelioPy
Geophysics:
MagPySV
GeomagPy
PyGIMLi
fatiando
Analysis & visualisation in Python

Example Use Cases

Example 1: Make a field model
*Use one month of **B** measurements*



```python
from viresclient import SwarmRequest
import datetime as dt

request = SwarmRequest(url="https://staging.viresdisc.vires.services/openows",
                       username="your username",
                       password="your password")

request.set_collection("SW_OPER_MAGA_LR_1B")

request.set_products(measurements=["B_NEC"], sampling_step="PT10S")

request.set_range_filter('Kp', 0, 30)

data = request.get_between(start_time=dt.datetime(2016,1,1),
                           end_time=dt.datetime(2016,2,1))
```

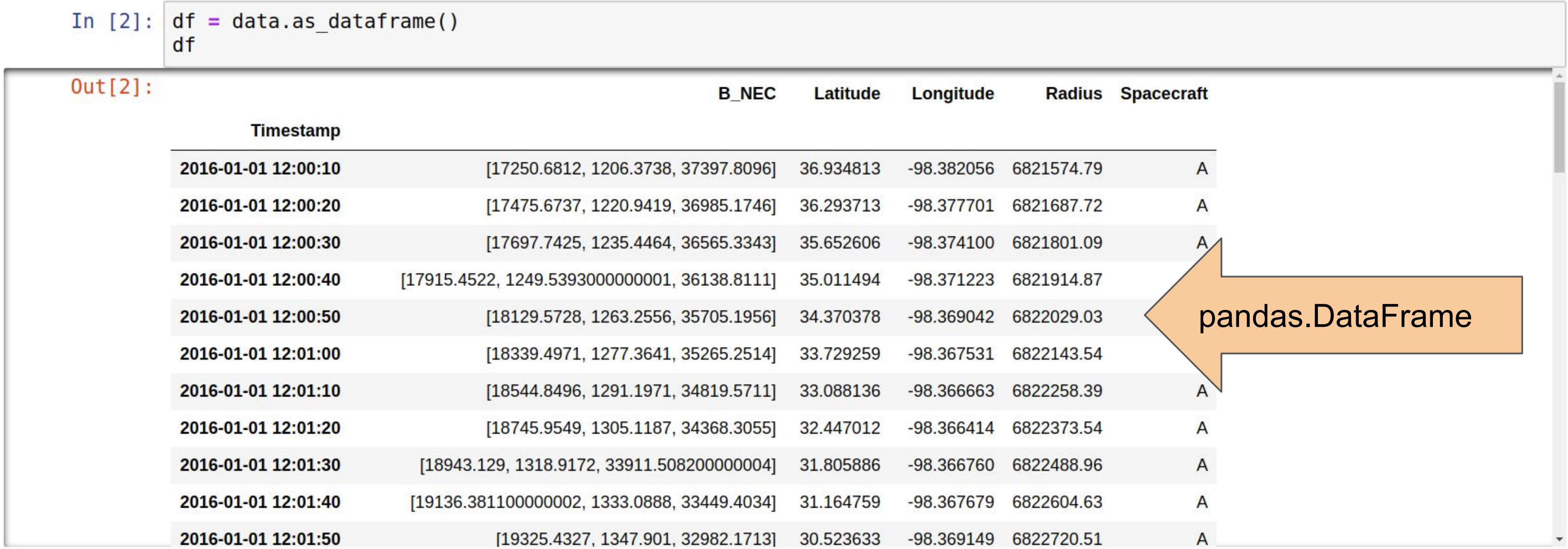
```
[1/1] Processing:  100%|██████████|  [ Elapsed: 00:03, Remaining: 00:00 ]
      Downloading: 100%|██████████|  [ Elapsed: 00:12, Remaining: 00:00 ] (13.125MB)
```

Connect to the server

Choose data, sampling, filters

Choose time range and download

Example 1: Make a field model
Transfer to a dataframe (now ready for analysis)



```
In [2]: df = data.as_dataframe()
        df
```

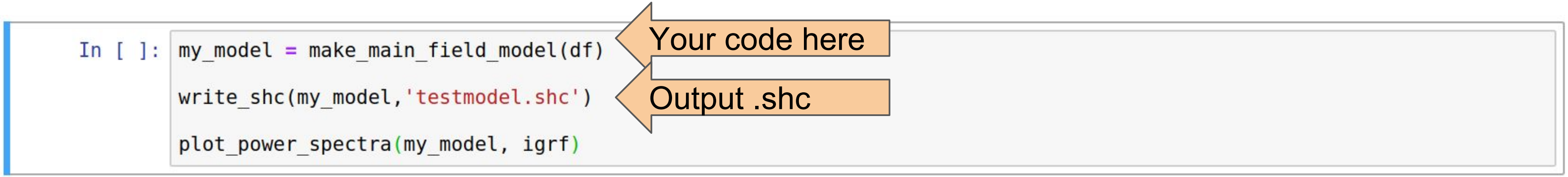
Out[2]:

Timestamp	B_NEC	Latitude	Longitude	Radius	Spacecraft
2016-01-01 12:00:10	[17250.6812, 1206.3738, 37397.8096]	36.934813	-98.382056	6821574.79	A
2016-01-01 12:00:20	[17475.6737, 1220.9419, 36985.1746]	36.293713	-98.377701	6821687.72	A
2016-01-01 12:00:30	[17697.7425, 1235.4464, 36565.3343]	35.652606	-98.374100	6821801.09	A
2016-01-01 12:00:40	[17915.4522, 1249.5393000000001, 36138.8111]	35.011494	-98.371223	6821914.87	
2016-01-01 12:00:50	[18129.5728, 1263.2556, 35705.1956]	34.370378	-98.369042	6822029.03	
2016-01-01 12:01:00	[18339.4971, 1277.3641, 35265.2514]	33.729259	-98.367531	6822143.54	
2016-01-01 12:01:10	[18544.8496, 1291.1971, 34819.5711]	33.088136	-98.366663	6822258.39	A
2016-01-01 12:01:20	[18745.9549, 1305.1187, 34368.3055]	32.447012	-98.366414	6822373.54	A
2016-01-01 12:01:30	[18943.129, 1318.9172, 33911.508200000004]	31.805886	-98.366760	6822488.96	A
2016-01-01 12:01:40	[19136.381100000002, 1333.0888, 33449.4034]	31.164759	-98.367679	6822604.63	A
2016-01-01 12:01:50	[19325.4327, 1347.901, 32982.1713]	30.523633	-98.369149	6822720.51	A

pandas.DataFrame

Example 1: Make a field model

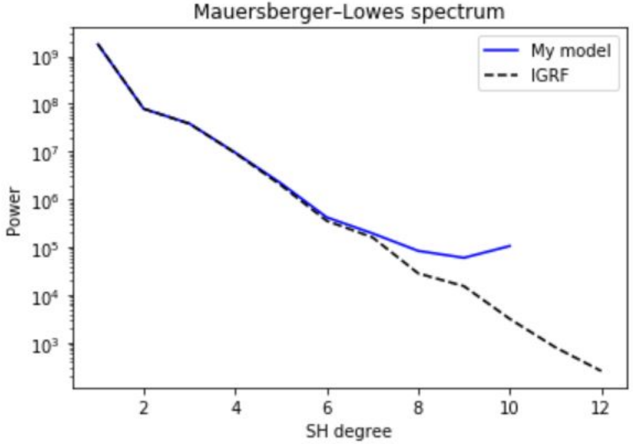
Generate Gauss coefficients



```
In [ ]: my_model = make_main_field_model(df)

        write_shc(my_model,'testmodel.shc')

        plot_power_spectra(my_model, igrf)
```

Your code here

Output .shc

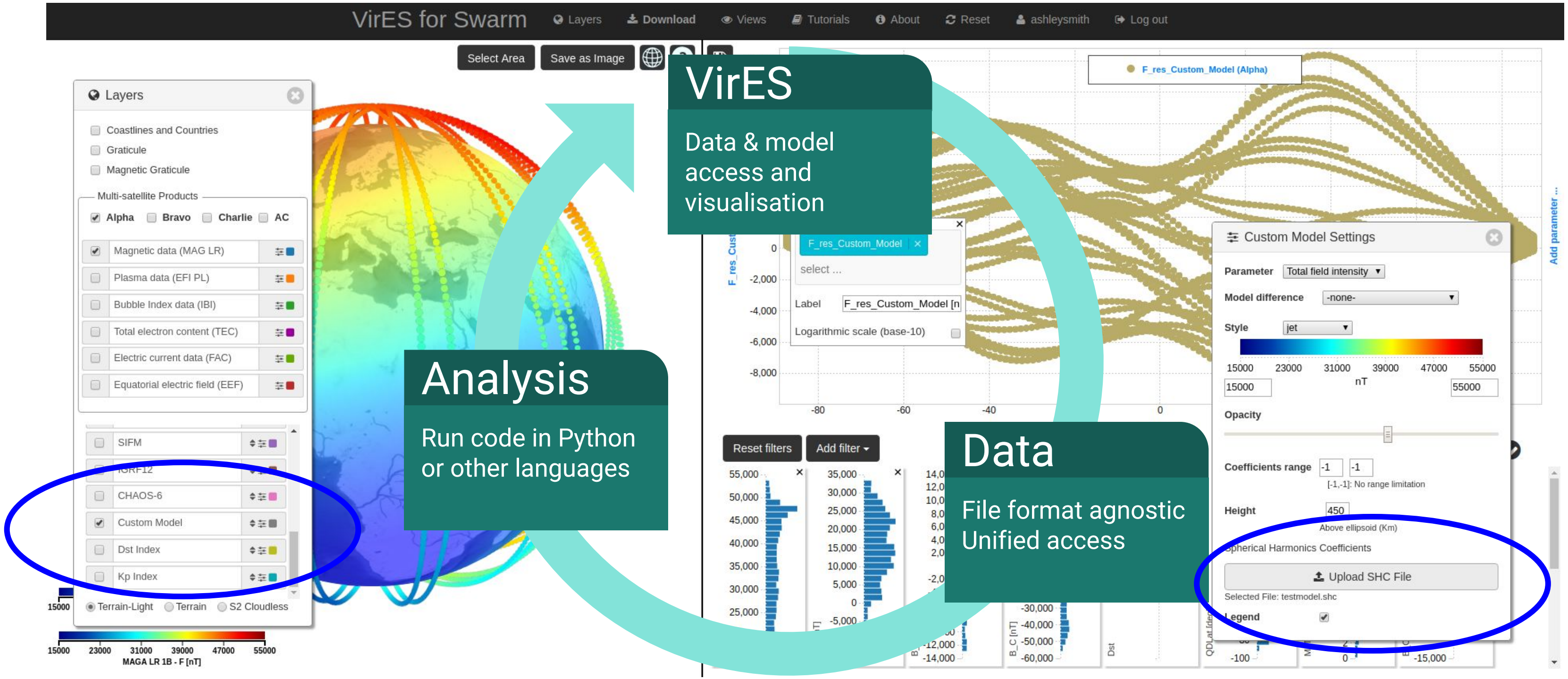


Next step… ?

An open source field modelling library

- Common preparation routines
- Different inversion options
- Analysis & validation utilities

Example 1: Make a field model
Upload as a custom model in VirES



VirES

Data & model access and visualisation

Analysis

Run code in Python or other languages

Data

File format agnostic
Unified access

Layers

- □ Coastlines and Countries
- □ Graticule
- □ Magnetic Graticule

Multi-satellite Products: ☑ Alpha □ Bravo □ Charlie □ AC

- ☑ Magnetic data (MAG LR)
- □ Plasma data (EFI PL)
- □ Bubble Index data (IBI)
- □ Total electron content (TEC)
- □ Electric current data (FAC)
- □ Equatorial electric field (EEF)
- □ SIFM
- □ IGRF12
- □ CHAOS-6
- ☑ Custom Model
- □ Dst Index
- □ Kp Index

● Terrain-Light ○ Terrain ○ S2 Cloudless

Label: F_res_Custom_Model [n

Logarithmic scale (base-10) □

Custom Model Settings

Parameter: Total field intensity

Model difference: -none-

Style: jet

15000 – 55000 nT

Opacity

Coefficients range: -1 -1

[-1,-1]: No range limitation

Height: 450

Above ellipsoid (Km)

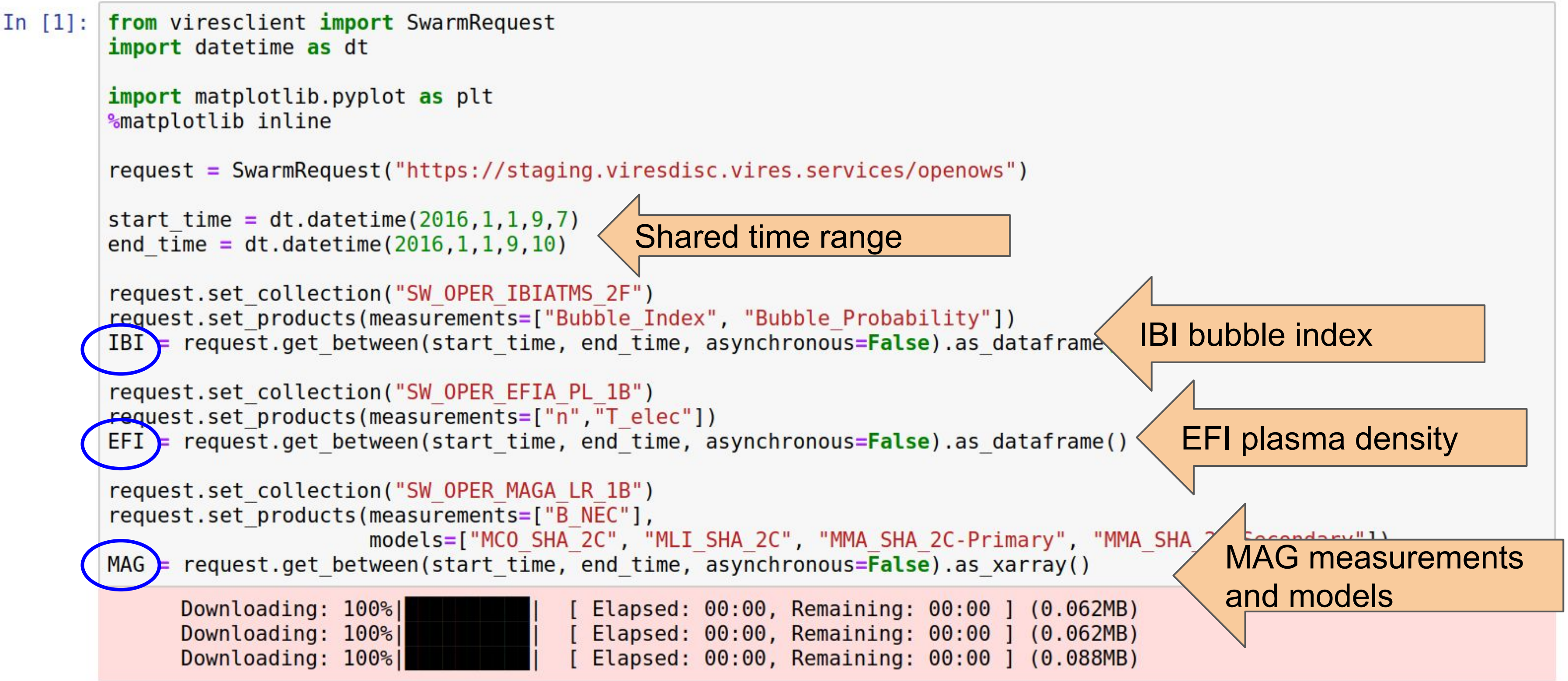
Spherical Harmonics Coefficients

Upload SHC File

Selected File: testmodel.shc

Legend ☑

Example 2: Load multiple products



```
In [1]: from viresclient import SwarmRequest
        import datetime as dt

        import matplotlib.pyplot as plt
        %matplotlib inline

        request = SwarmRequest("https://staging.viresdisc.vires.services/openows")

        start_time = dt.datetime(2016,1,1,9,7)
        end_time = dt.datetime(2016,1,1,9,10)

        request.set_collection("SW_OPER_IBIATMS_2F")
        request.set_products(measurements=["Bubble_Index", "Bubble_Probability"])
        IBI = request.get_between(start_time, end_time, asynchronous=False).as_dataframe

        request.set_collection("SW_OPER_EFIA_PL_1B")
        request.set_products(measurements=["n","T_elec"])
        EFI = request.get_between(start_time, end_time, asynchronous=False).as_dataframe()

        request.set_collection("SW_OPER_MAGA_LR_1B")
        request.set_products(measurements=["B_NEC"],
                             models=["MCO_SHA_2C", "MLI_SHA_2C", "MMA_SHA_2C-Primary", "MMA_SHA_2C-Secondary"])
        MAG = request.get_between(start_time, end_time, asynchronous=False).as_xarray()

            Downloading: 100%|██████████|  [ Elapsed: 00:00, Remaining: 00:00 ] (0.062MB)
            Downloading: 100%|██████████|  [ Elapsed: 00:00, Remaining: 00:00 ] (0.062MB)
            Downloading: 100%|██████████|  [ Elapsed: 00:00, Remaining: 00:00 ] (0.088MB)
```

Shared time range

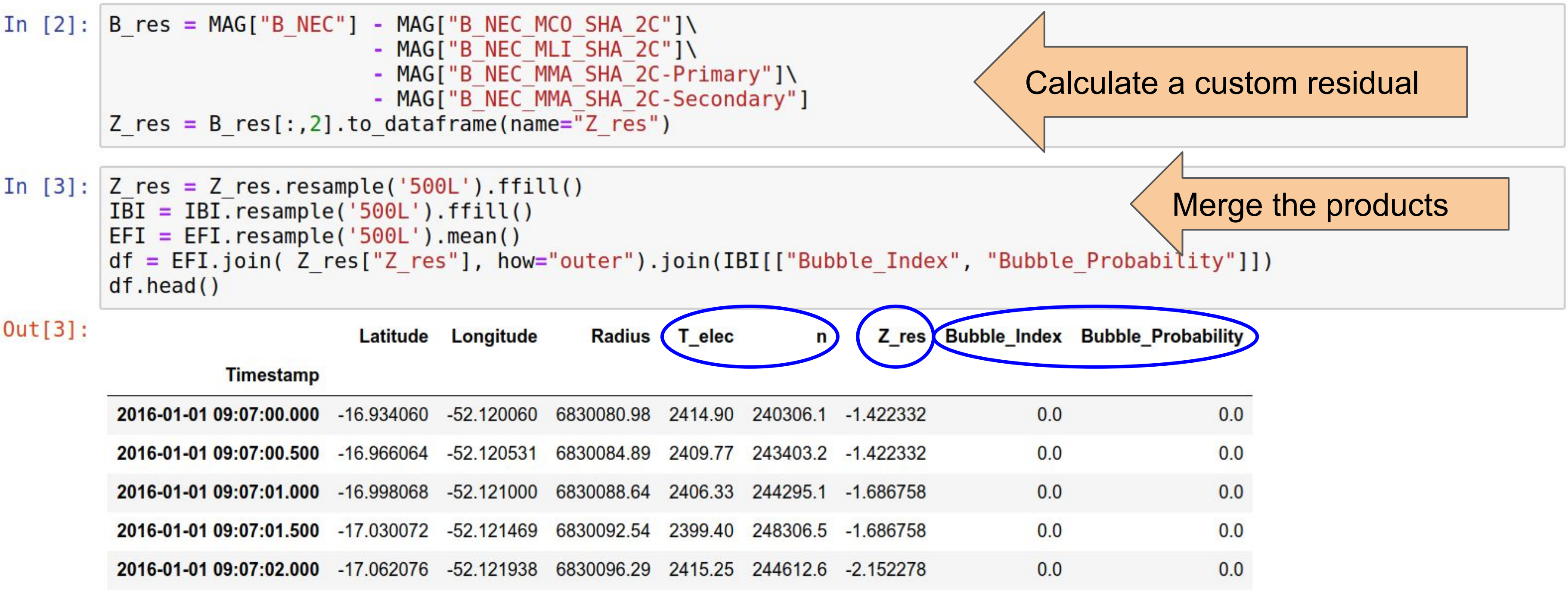
IBI bubble index

EFI plasma density

MAG measurements and models

Example 2: Load multiple products

(*merge into DataFrame*)



```
In [2]: B_res = MAG["B_NEC"] - MAG["B_NEC_MCO_SHA_2C"]\
                             - MAG["B_NEC_MLI_SHA_2C"]\
                             - MAG["B_NEC_MMA_SHA_2C-Primary"]\
                             - MAG["B_NEC_MMA_SHA_2C-Secondary"]
        Z_res = B_res[:,2].to_dataframe(name="Z_res")
```

Calculate a custom residual

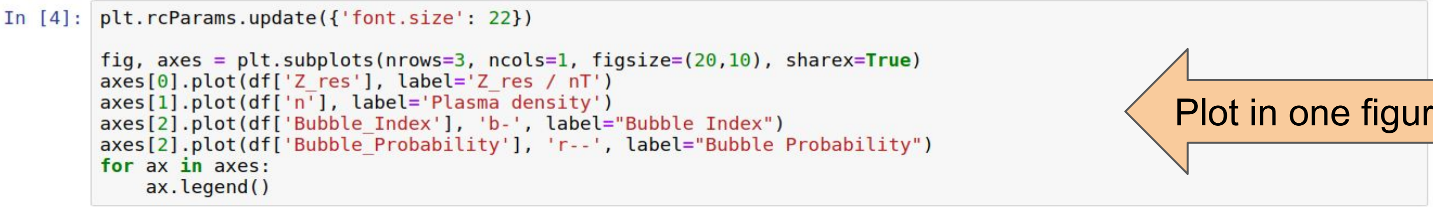
```
In [3]: Z_res = Z_res.resample('500L').ffill()
        IBI = IBI.resample('500L').ffill()
        EFI = EFI.resample('500L').mean()
        df = EFI.join( Z_res["Z_res"], how="outer").join(IBI[["Bubble_Index", "Bubble_Probability"]])
        df.head()
```

Merge the products

Out[3]:

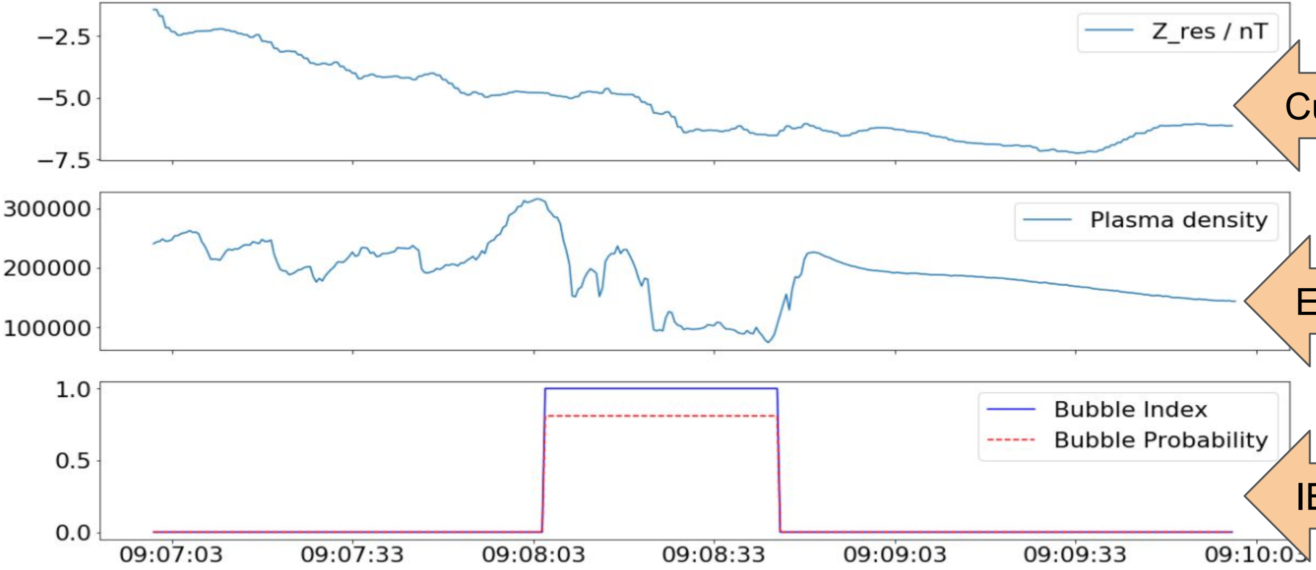
	Latitude	Longitude	Radius	T_elec	n	Z_res	Bubble_Index	Bubble_Probability
Timestamp								
2016-01-01 09:07:00.000	-16.934060	-52.120060	6830080.98	2414.90	240306.1	-1.422332	0.0	0.0
2016-01-01 09:07:00.500	-16.966064	-52.120531	6830084.89	2409.77	243403.2	-1.422332	0.0	0.0
2016-01-01 09:07:01.000	-16.998068	-52.121000	6830088.64	2406.33	244295.1	-1.686758	0.0	0.0
2016-01-01 09:07:01.500	-17.030072	-52.121469	6830092.54	2399.40	248306.5	-1.686758	0.0	0.0
2016-01-01 09:07:02.000	-17.062076	-52.121938	6830096.29	2415.25	244612.6	-2.152278	0.0	0.0

Example 2: Load multiple products *(plot with matplotlib)*



```
In [4]: plt.rcParams.update({'font.size': 22})

        fig, axes = plt.subplots(nrows=3, ncols=1, figsize=(20,10), sharex=True)
        axes[0].plot(df['Z_res'], label='Z_res / nT')
        axes[1].plot(df['n'], label='Plasma density')
        axes[2].plot(df['Bubble_Index'], 'b-', label="Bubble Index")
        axes[2].plot(df['Bubble_Probability'], 'r--', label="Bubble Probability")
        for ax in axes:
            ax.legend()
```

Plot in one figure

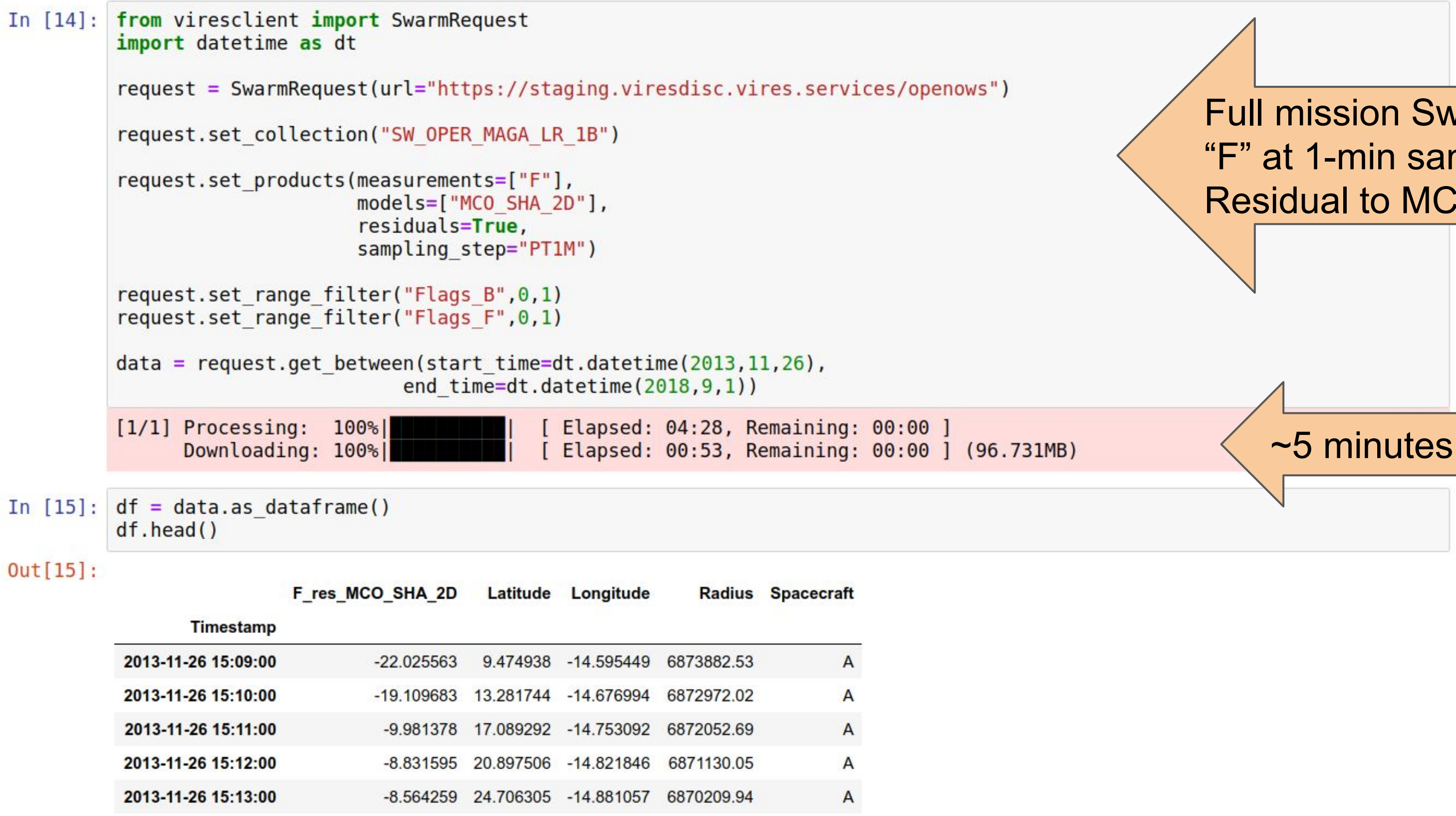


Custom MAG residual

EFI plasma density

IBI bubble index

Example 3: Accessing a large amount of data



```
In [14]: from viresclient import SwarmRequest
         import datetime as dt

         request = SwarmRequest(url="https://staging.viresdisc.vires.services/openows")

         request.set_collection("SW_OPER_MAGA_LR_1B")

         request.set_products(measurements=["F"],
                              models=["MCO_SHA_2D"],
                              residuals=True,
                              sampling_step="PT1M")

         request.set_range_filter("Flags_B",0,1)
         request.set_range_filter("Flags_F",0,1)

         data = request.get_between(start_time=dt.datetime(2013,11,26),
                                    end_time=dt.datetime(2018,9,1))
```

```
[1/1] Processing:  100%|██████████|  [ Elapsed: 04:28, Remaining: 00:00 ]
      Downloading: 100%|██████████|  [ Elapsed: 00:53, Remaining: 00:00 ] (96.731MB)
```

```
In [15]: df = data.as_dataframe()
         df.head()
```

Out[15]:

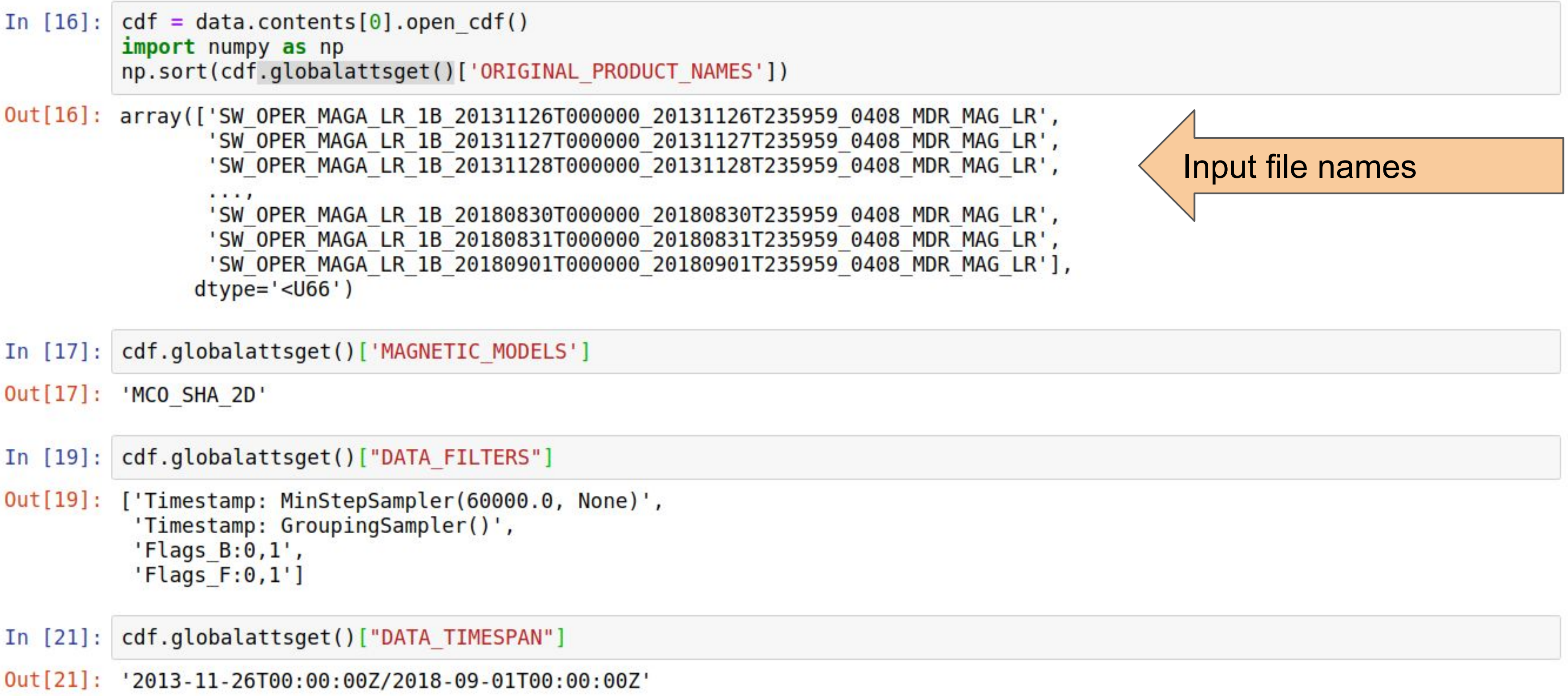
	F_res_MCO_SHA_2D	Latitude	Longitude	Radius	Spacecraft
Timestamp					
2013-11-26 15:09:00	-22.025563	9.474938	-14.595449	6873882.53	A
2013-11-26 15:10:00	-19.109683	13.281744	-14.676994	6872972.02	A
2013-11-26 15:11:00	-9.981378	17.089292	-14.753092	6872052.69	A
2013-11-26 15:12:00	-8.831595	20.897506	-14.821846	6871130.05	A
2013-11-26 15:13:00	-8.564259	24.706305	-14.881057	6870209.94	A

Full mission Swarm Alpha MAG
“F” at 1-min sampling
Residual to MCO_SHA_2D

~5 minutes to process

Example 3: Accessing a large amount of data

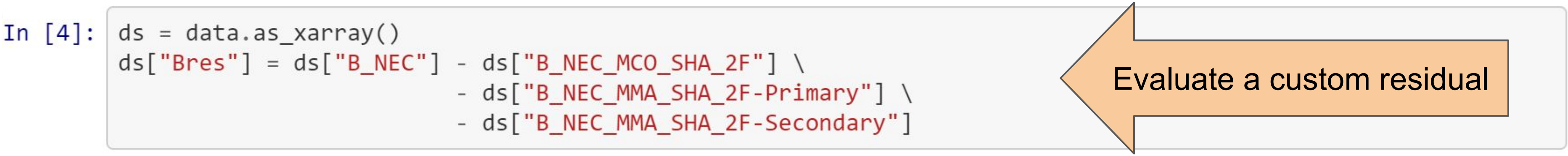
Traceability of inputs



```
In [16]: cdf = data.contents[0].open_cdf()
         import numpy as np
         np.sort(cdf.globalattsget()['ORIGINAL_PRODUCT_NAMES'])

Out[16]: array(['SW_OPER_MAGA_LR_1B_20131126T000000_20131126T235959_0408_MDR_MAG_LR',
                'SW_OPER_MAGA_LR_1B_20131127T000000_20131127T235959_0408_MDR_MAG_LR',
                'SW_OPER_MAGA_LR_1B_20131128T000000_20131128T235959_0408_MDR_MAG_LR',
                ...,
                'SW_OPER_MAGA_LR_1B_20180830T000000_20180830T235959_0408_MDR_MAG_LR',
                'SW_OPER_MAGA_LR_1B_20180831T000000_20180831T235959_0408_MDR_MAG_LR',
                'SW_OPER_MAGA_LR_1B_20180901T000000_20180901T235959_0408_MDR_MAG_LR'],
               dtype='<U66')

In [17]: cdf.globalattsget()['MAGNETIC_MODELS']

Out[17]: 'MCO_SHA_2D'

In [19]: cdf.globalattsget()["DATA_FILTERS"]

Out[19]: ['Timestamp: MinStepSampler(60000.0, None)',
          'Timestamp: GroupingSampler()',
          'Flags_B:0,1',
          'Flags_F:0,1']

In [21]: cdf.globalattsget()["DATA_TIMESPAN"]

Out[21]: '2013-11-26T00:00:00Z/2018-09-01T00:00:00Z'
```

Input file names

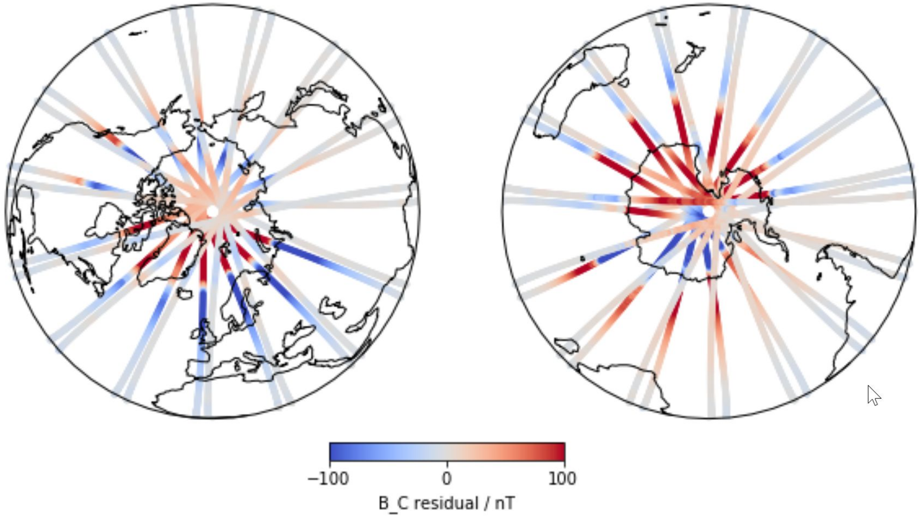
Example 4: Easy to integrate with other tools



```
In [4]: ds = data.as_xarray()
        ds["Bres"] = ds["B_NEC"] - ds["B_NEC_MCO_SHA_2F"] \
                                 - ds["B_NEC_MMA_SHA_2F-Primary"] \
                                 - ds["B_NEC_MMA_SHA_2F-Secondary"]
```

Evaluate a custom residual

$$B_{res} = B_{obs} - B_{MCO(F)} - B_{MMA(F)}$$

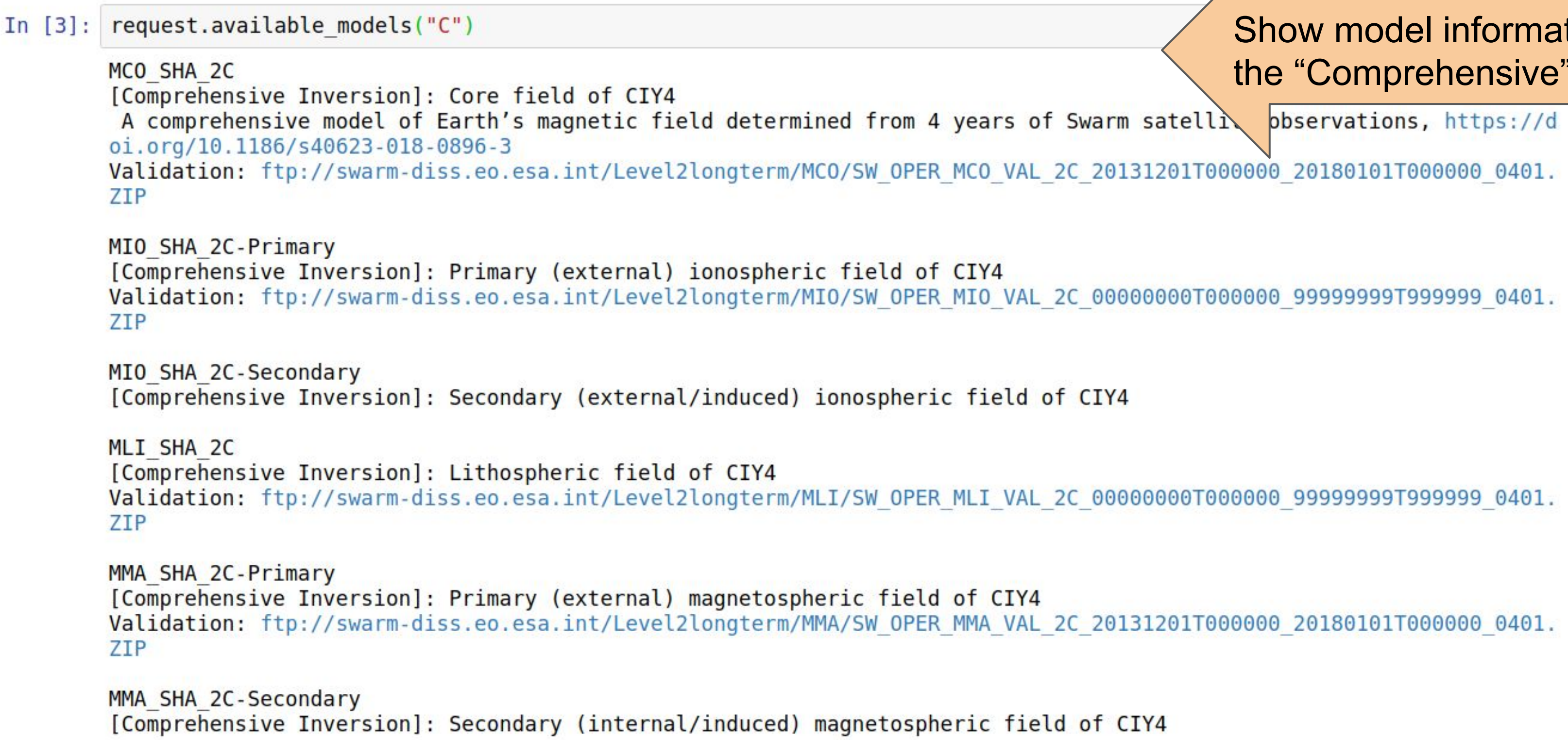


Plot a viresclient xarray using cartopy

https://scitools.org.uk/cartopy

Example 5: Querying what is available

Useful information available directly through the code



```
In [3]: request.available_models("C")

        MCO_SHA_2C
        [Comprehensive Inversion]: Core field of CIY4
         A comprehensive model of Earth’s magnetic field determined from 4 years of Swarm satelli  observations, https://d
        oi.org/10.1186/s40623-018-0896-3
        Validation: ftp://swarm-diss.eo.esa.int/Level2longterm/MCO/SW_OPER_MCO_VAL_2C_20131201T000000_20180101T000000_0401.
        ZIP

        MIO_SHA_2C-Primary
        [Comprehensive Inversion]: Primary (external) ionospheric field of CIY4
        Validation: ftp://swarm-diss.eo.esa.int/Level2longterm/MIO/SW_OPER_MIO_VAL_2C_00000000T000000_99999999T999999_0401.
        ZIP

        MIO_SHA_2C-Secondary
        [Comprehensive Inversion]: Secondary (external/induced) ionospheric field of CIY4

        MLI_SHA_2C
        [Comprehensive Inversion]: Lithospheric field of CIY4
        Validation: ftp://swarm-diss.eo.esa.int/Level2longterm/MLI/SW_OPER_MLI_VAL_2C_00000000T000000_99999999T999999_0401.
        ZIP

        MMA_SHA_2C-Primary
        [Comprehensive Inversion]: Primary (external) magnetospheric field of CIY4
        Validation: ftp://swarm-diss.eo.esa.int/Level2longterm/MMA/SW_OPER_MMA_VAL_2C_20131201T000000_20180101T000000_0401.
        ZIP

        MMA_SHA_2C-Secondary
        [Comprehensive Inversion]: Secondary (internal/induced) magnetospheric field of CIY4
```

Show model information from the “Comprehensive” series

Features that will be added

- Product metadata
  - e.g. baseline version numbers
  - Model details, coefficients, validity period
  - xarray.Dataset can carry metadata: units, dimension names, ...
- Customised model evaluation
  - Specifying choice of coefficients; combinations of models
  - At custom locations & times (e.g. for AMPS model explorer)
- User-defined model evaluation
  - Upload .shc file
- Unified file I/O
  - Managing generated files
  - Read in pre-defined types of data
    - scalar/vector time series; .shc models; …
    - Alternative versions of existing files

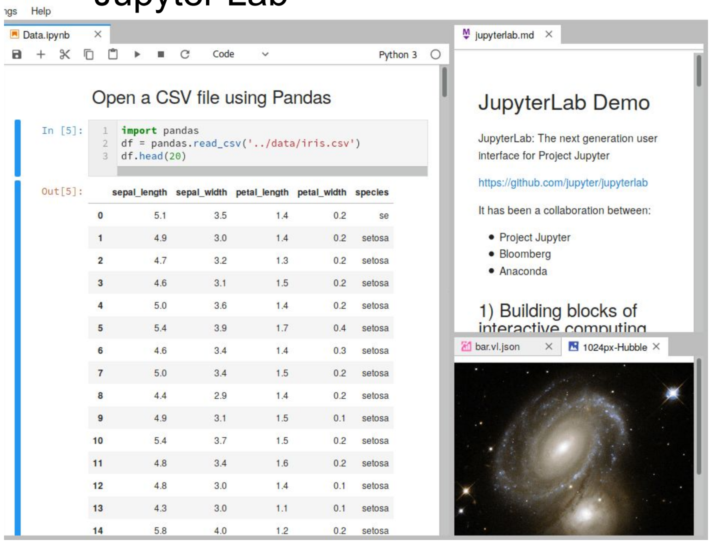
Possible future development

- Pre-defined routines
    - Generating reports, quicklook plots
    - “Cleaning” and other common procedures
- Custom data classes
    - Domain-specific tools & plots
- Integration with other Python packages
    - Spacepy, Pysat, MagPySV, GeomagPy, cartopy, ...
- Interoperability with other languages
    - Julia, Matlab, R, …
- Jupyter widgets / extensions
- Integration with VirES web visualisations

viresclient

Broader geomagnetism package?
(functionality separate but affiliated)

Jupyter Lab



Open a CSV file using Pandas

In [5]:

```
import pandas
df = pandas.read_csv('../data/iris.csv')
df.head(20)
```

Out[5]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	se
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa
10	5.4	3.7	1.5	0.2	setosa
11	4.8	3.4	1.6	0.2	setosa
12	4.8	3.0	1.4	0.1	setosa
13	4.3	3.0	1.1	0.1	setosa
14	5.8	4.0	1.2	0.2	setosa

JupyterLab Demo

JupyterLab: The next generation user interface for Project Jupyter

https://github.com/jupyter/jupyterlab

It has been a collaboration between:

- Project Jupyter
- Bloomberg
- Anaconda

1) Building blocks of interactive computing

Open source development model

- Me / EOX as the maintainer
  - Build core functionality
  - Continued development in tandem with changes to the server
  - Verify contributions & maintain high quality
- Scientists as contributors
  - Suggesting ideas to be added
  - Validation of the implementations on VirES
  - Writing domain-specific code, e.g. to generate a particular type of plot
  - Prototype features and share “recipes” in Jupyter notebooks
- Modular approach
  - Easier to manage and maintain
  - Easier to contribute
  - See astropy, sunpy as examples of how to coordinate affiliated packages

Open source development model

- Building a cookbook: common examples and best practice
  - A central location to organise common code recipes
  - Jupyter notebooks are a nice way to present these
  - https://github.com/smithara/viresclient_examples (includes examples from this presentation)
- viresclient as the “batteries” to provide data access to other packages

Access and registration

- Currently only available on VirES-DISC server
  - https://staging.viresdisc.vires.services/openows
  - Some DISC users have access but requires manual account creation

- Planned to open to all at https://vires.services/openows
- Could be public in February?
- For interested users now: we can give you access

Thank you for listening

- Blog post: https://eox.at
- Documentation: https://viresclient.readthedocs.io
- Example notebooks: https://github.com/smithara/viresclient_examples

- xarray: http://xarray.pydata.org
- cartopy: https://scitools.org.uk/cartopy
- GMT/Python: https://www.gmtpython.xyz

- MagPySV: https://magpysv.readthedocs.io 10.1029/2018GC007714
- Pysat: https://pysat.readthedocs.io 10.1029/2018JA025297

- SHTools: https://shtools.github.io/SHTOOLS 10.1029/2018GC007529

Bonus slides

Aims for viresclient

- Complementary to the web client
  - Encourage more usage of VirES
  - Leverage the power of the server-client architecture
- Access Swarm data programmatically
  - Avoid woes of file formats and data management
  - Reduce the number of steps required before performing some real analysis
- Bring data and models together
  - Easier access to model evaluations without needing to know the model details

Aims for viresclient

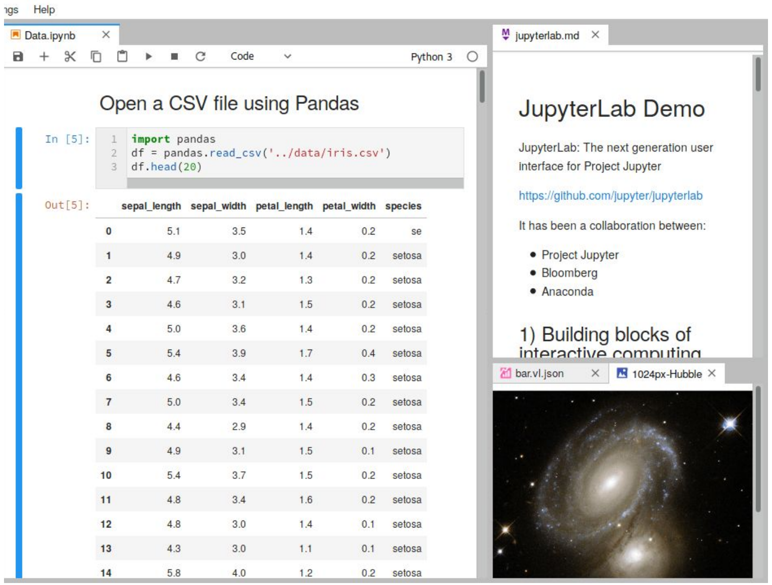
- Greater reproducibility of code
  - Develop a common framework for easier peer review and sharing of code
  - Develop shortcuts to useful reduced datasets
- Interface with the Python ecosystem
- Easy to install (but requires Python >= 3.5)
  - `pip install viresclient`
  - Current issue with Windows compatibility in v0.2.4 (will fix soon) https://github.com/ESA-VirES/VirES-Python-Client/issues/1

Field modelling library

- Community-driven open source “geomagnetic field modelling” library
  - viresclient provides the “batteries” to provide the data input
  - Wrap functionality from scipy and elsewhere in a more convenient interface
  - Tools to help build models
    - design_SHA, ...
  - Tools to analyse and validate models
    - Power spectra, correlation, …
    - Global plots
    - Comparison with reference models
  - Fast model evaluations
    - Could use VirES server
    - Could use eoxmagmod directly

Use of Jupyter

- The Jupyter project has strong support across industry
  - “ Customizable, Flexible, Scalable, Portable”
- Jupyter Notebooks (and soon Jupyter Lab)
  - Intuitive way to work with live, annotated code
  - Great for sharing and teaching
- Extensible with interactive widgets
  - Could support future VirES visualisations



Open a CSV file using Pandas

```
In [5]:
import pandas
df = pandas.read_csv('../data/iris.csv')
df.head(20)
```

Out[5]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	se
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa
10	5.4	3.7	1.5	0.2	setosa
11	4.8	3.4	1.6	0.2	setosa
12	4.8	3.0	1.4	0.1	setosa
13	4.3	3.0	1.1	0.1	setosa
14	5.8	4.0	1.2	0.2	setosa

JupyterLab Demo

JupyterLab: The next generation user interface for Project Jupyter

https://github.com/jupyter/jupyterlab

It has been a collaboration between:

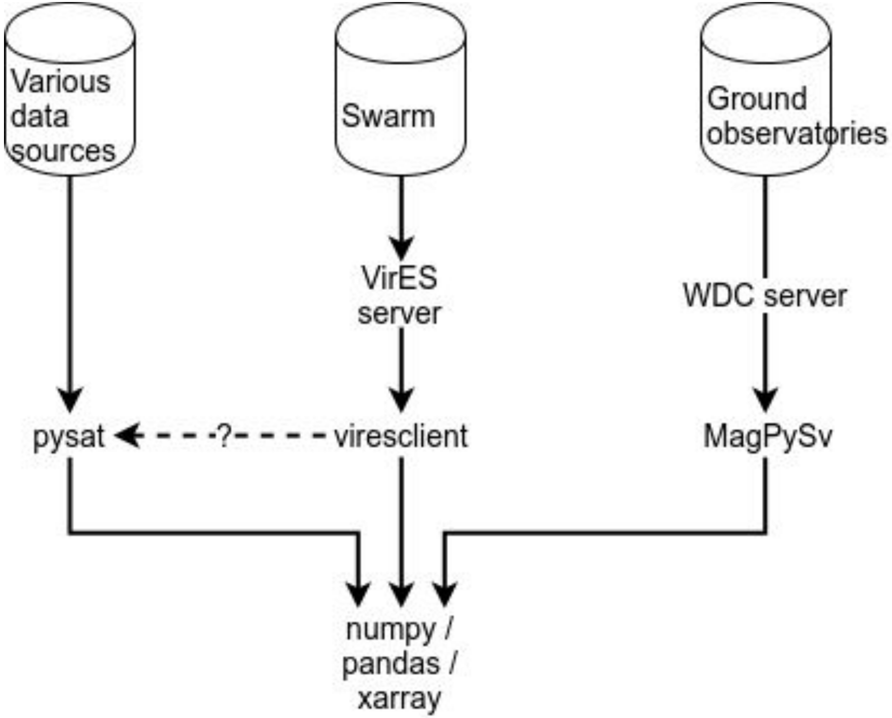
- Project Jupyter
- Bloomberg
- Anaconda

1) Building blocks of interactive computing

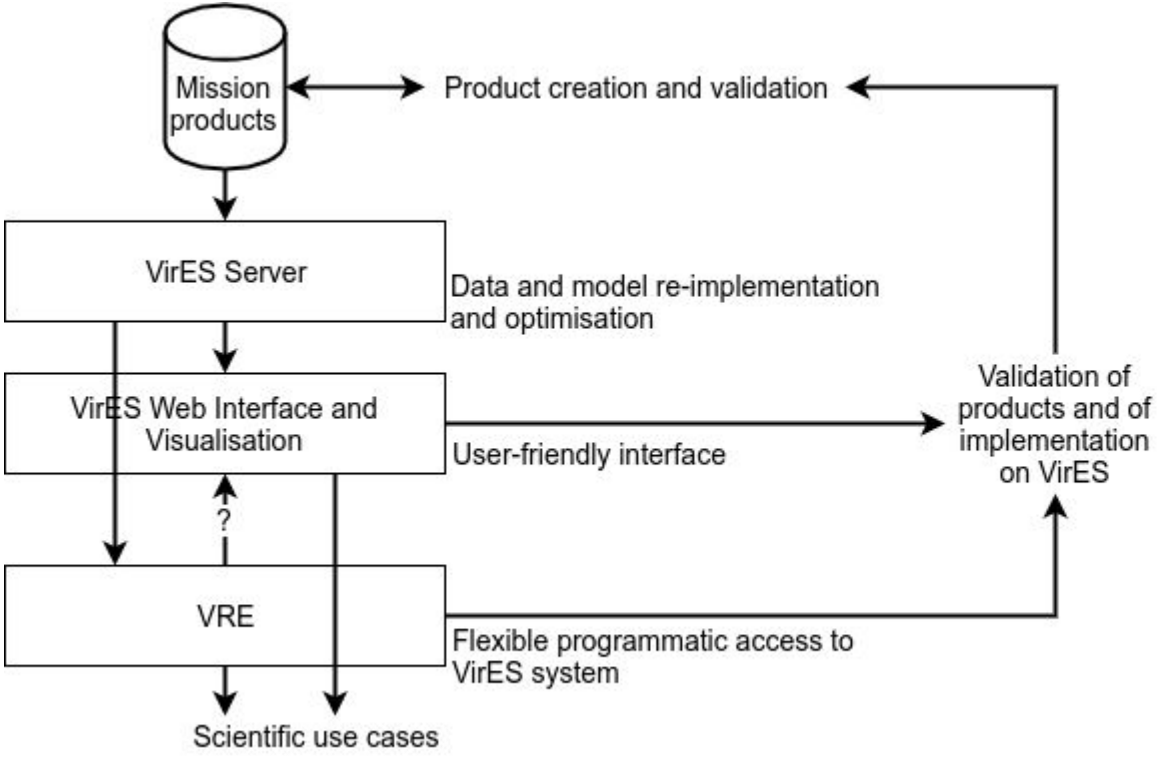
Use with existing Python tools

- pandas & xarray
- Pysat
  - https://pysat.readthedocs.io
- MagPySv & gmdata_webinterface
  - https://magpysv.readthedocs.io

- System science requires easy access to (often inhomogeneous) datasets
  - Use the appropriate tools to access each, and merge into one workflow



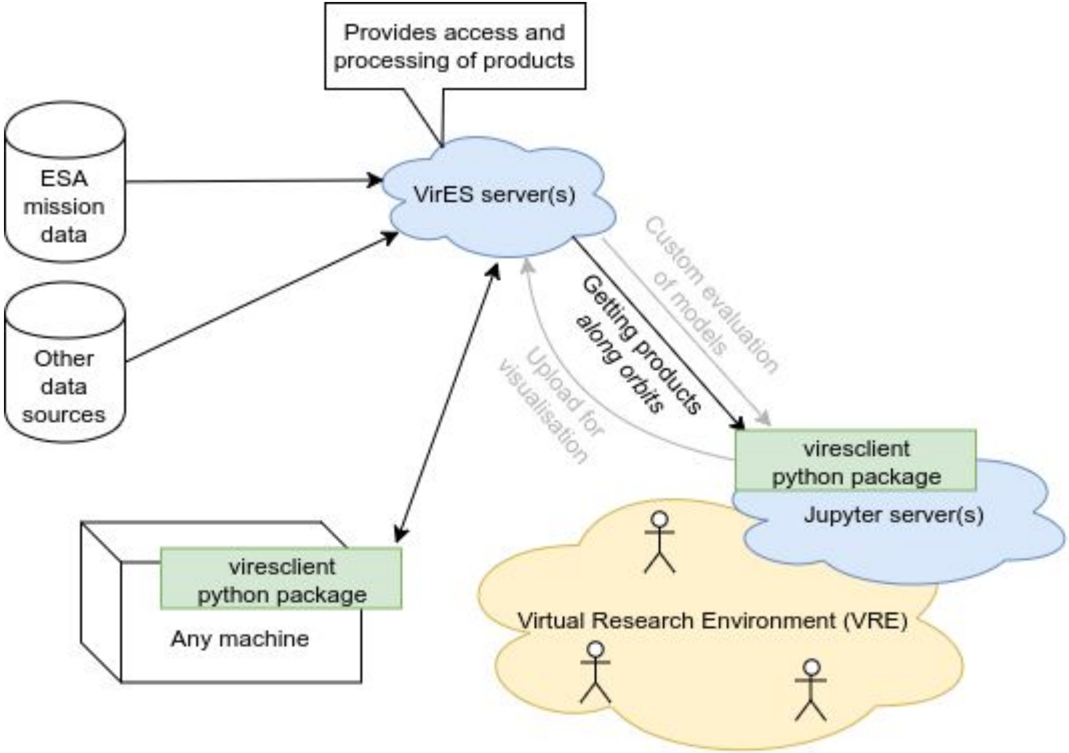
Information flow - role in product validation?


Mission products
Product creation and validation
VirES Server
Data and model re-implementation and optimisation
VirES Web Interface and Visualisation
User-friendly interface
Validation of products and of implementation on VirES
?
VRE
Flexible programmatic access to VirES system
Scientific use cases

Virtual Research Environment (VRE)

- viresclient is a core component

  “Batteries included”



Provides access and processing of products
ESA mission data
Other data sources
VirES server(s)
Custom evaluation of models
Getting products along orbits
Upload for visualisation
viresclient python package
Jupyter server(s)
viresclient python package
Any machine
Virtual Research Environment (VRE)