# IRF-Uppsala Python Workshop: Snakes in Space

L. Richard, D. Andrews, V. Lanabere

## Outline

- 1st half day:
  - Introduction to Python
  - Python best practices
  - Data analysis with Python
  - CDF files in Python

- 2nd half day:
  - Time series, spectrograms & co.
  - PyRFU: An open IRF-U toolbox for space data analysis
  - geopack

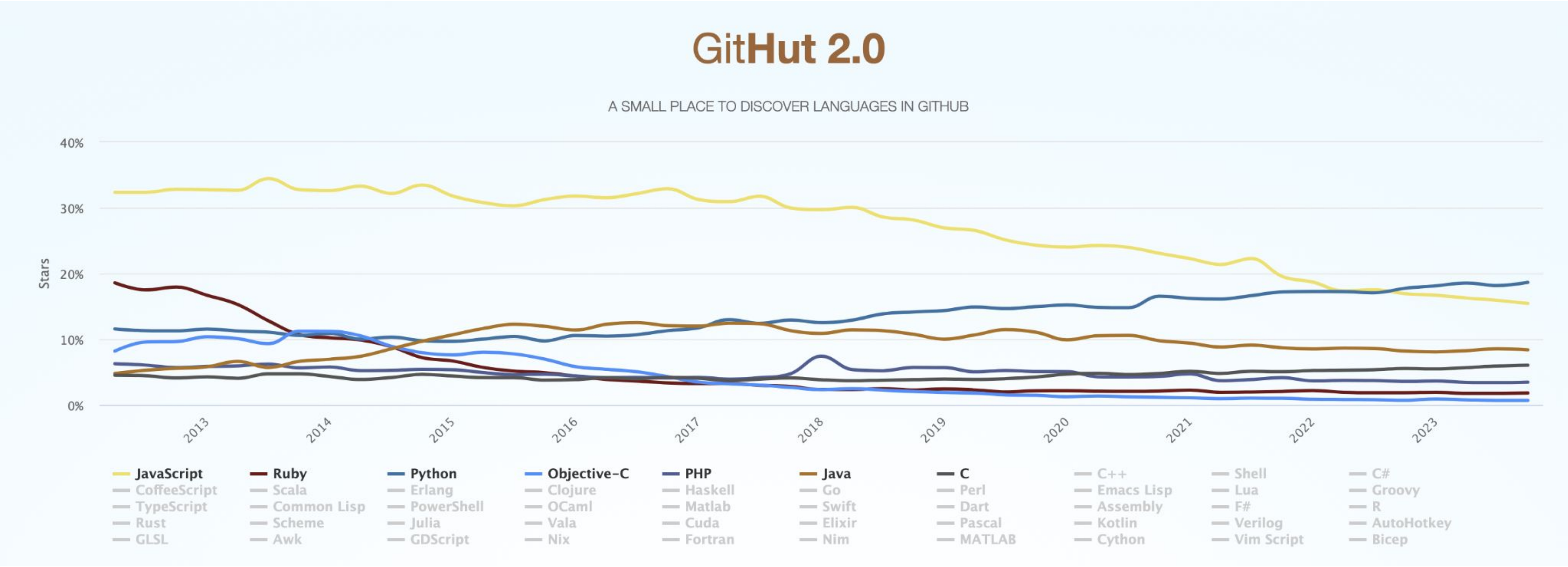
## Topics NOT covered but that we would love to discuss

- High-performance computing: Speed-up your code
    - GIL
    - Fortran, C/C++ wrapping
    - openMP, CUDA
    - numba, cupy

- Debugging:
    - Deguggers: pdb, etc.
    - Profiling: cProfile, etc.

- and many others…

# Why python?


GitHut 2.0
A SMALL PLACE TO DISCOVER LANGUAGES IN GITHUB
Stars
40%
30%
20%
10%
0%
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
JavaScript
Ruby
Python
Objective-C
PHP
Java
C
C++
Shell
C#
CoffeeScript
Scala
Erlang
Clojure
Haskell
Go
Perl
Emacs Lisp
Lua
Groovy
TypeScript
Common Lisp
PowerShell
OCaml
Matlab
Swift
Dart
Assembly
F#
R
Rust
Scheme
Julia
Vala
Cuda
Elixir
Pascal
Kotlin
Verilog
AutoHotkey
GLSL
Awk
GDScript
Nix
Fortran
Nim
MATLAB
Cython
Vim Script
Bicep

## Why python?

What is it?

- Free, open source software
- Very widely used in many situations
- Portable, mature, tested code base
- “Batteries included” - extensive standard library of tools
- Emphasis on code readability

For data analysis specifically

- Extensive, mature stack of tools; numpy, scipy, matplotlib, …
- “Glue language”
- Easy for the community to contribute well-formed, tested, portable software to the hive mind
- Facilitate “reproducible research”

# Installing (if you haven’t already)

There are many ways to install python, and this can be confusing

1) You may already have it installed anyway (and, it may already be being used by some software on your computer…)
2) https://www.python.org/downloads/ “Official” installer, for your system which will install **python**, **pip** and some other stuff.
3) Alternatives: tools like https://conda.io/projects/conda/en/latest/index.html that provide package & environment management, maybe with more GUIs that you really need/want. *[See backup slide]*
4) *Maybe viable:* the jupyter lab desktop application which includes a complete stack of python as well.

Bottom line:

***Official + pip is enough for this workshop.***

Risks to be aware of:

Different versions of python in use in different places, different versions of packages that depend on them.
Solution: **Virtual Environments**: venv or conda…

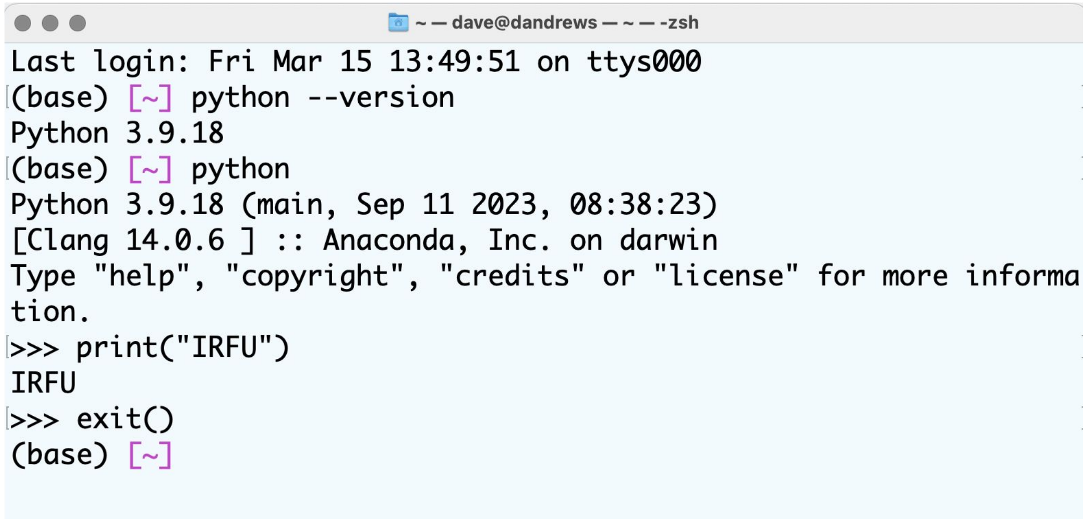
# Check python runs…

From the command line

`python --version`

Do something

exit with `exit()` or CTRL+D



```
Last login: Fri Mar 15 13:49:51 on ttys000
(base) [~] python --version
Python 3.9.18
(base) [~] python
Python 3.9.18 (main, Sep 11 2023, 08:38:23)
[Clang 14.0.6 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more informa
tion.
>>> print("IRFU")
IRFU
>>> exit()
(base) [~]
```

## Some extra tools are needed

Today, going to use “Jupyter Notebooks”: Even more interactive; inline results, images, markdown text, …

To install **jupyter**, we’ll use **pip:**

```
python3 -m pip install jupyterlab
```

For later, we’ll also need a couple other packages, let’s do that now

```
python3 -m pip install numpy scipy matplotlib ipympl
```

Loads of other packages installable this way,

pycdfpp, pyrfu, spiceypy, …

Let’s see if that worked…

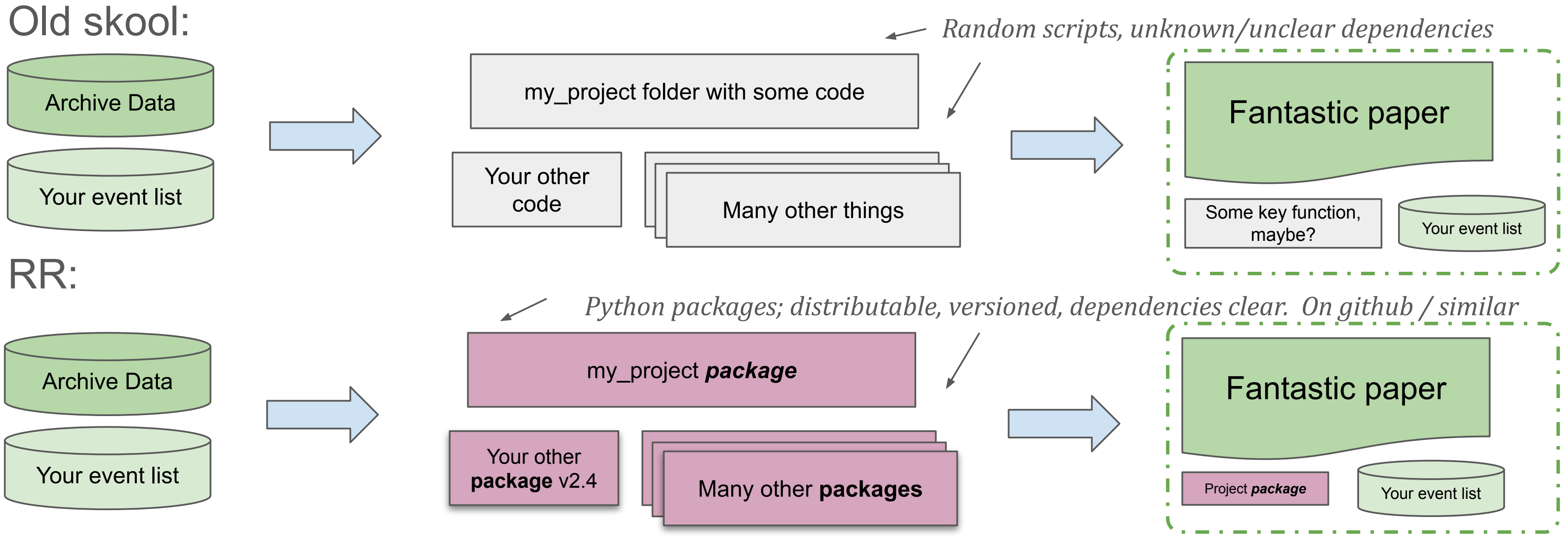
```
cd <somewhere safe>
git clone https://github.com/louis-richard/irfu-python-workshop.git
cd irfu-python-workshop
jupyter lab
```

Now we go through the first demo…

`tutorial/notebooks/1a ... .ipynb`

# Reproducible Research

A **python project (code + metadata)** is a good record of not just what your analysis did, but also **what it depends on** (such as other packages, external data)



***To reproduce:*** `pip install <my_project> && python -m my_project run_full_analysis`

# Python Best Practices: Code Style

Your code is meant to be shared, please make it readable!!!!

Python PEP8 provides guidelines in order to keep your code clean.

IDEs (e.g., Pycharm, Spyder, VSCode etc.)

- Provide autocompletion, auto-indentation, type hinting, etc.

linters (e.g., pylint, flake8, etc.):

- helps you to find where you break the PEP8 guidelines.
- tunable: you can deactivate SOME rules

Code style formatters (e.g., black, isort, etc.):

- Does the dirty job for you to make your code pretty

90% of all code comments:

## Python Best Practices: Virtual Environments

Lightweight “virtual environments” with their own site directories, optionally isolated from system site directories. Each virtual environment has its own Python binary (allowing creation of environments with various Python versions) and can have its own independent set of installed Python packages in its site directories, but shares the standard library with the base installed Python.

**Why: keeps your codes isolated from potential support breaking module updates. Good for reproducibility!!**

Create a new virtual environment:

```
>> python3 -m venv workshop
```

“Activate” your new virtual environment:

```
>> source workshop/bin/activate
```

# Python Best Practices: Packaging

Motivation: if you have bunch of functions that you would like to re-use across projects and/or share (e.g., on pypi)

A Python package typically consist of the following files:

```
.
├── LICENSE                   << where you tell how people can reuse your work
├── README.md                 << where you explain what you package is doing
├── mypkg                     << the actual package
│   ├── __init__.py           << import the directory as a regular package
│   └── example.py            << your code here
├── pyproject.toml            << tells your building tools which backend to use (+ some metadata)
└── tests                     << where you test your code (not covered* in this workshop)
```

*pun intended

## Outline (Day 2)

- 1st half day:
  - Introduction to Python
  - Python best practices

- 2nd half day:
  - Data analysis with Python
  - geopack
  - CDF files in Python
  - Time series, spectrograms & co.
  - PyRFU: An open IRF-U toolbox for space data analysis

## Data analysis in Python - Some standard tools…

"Python is a programming language that lets you work quickly and integrate systems more effectively.”

“Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.”

NumPy is the fundamental package for scientific computing in Python. It provides a multidimensional array object and more

“The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.”

“Xarray makes working with labelled multi-dimensional arrays in Python simple, efficient, and fun!”

SciPy is an open-source software for mathematics, science, and engineering.

# Common Data Format (CDF) files

## What is Common Data Format (CDF)?

- Self-describing data format for the storage of scalar and multidimensional data in a platform- and discipline-independent way
- Scientific data management package (CDF Library) allows application developers to manage these data arrays
- Transparent access to data and meta-data through Application Programming Interfaces (APIs)
- Built-in support for data compression (gZip, RLE, Huffman) and automatic data uncompression, and checksum
- Large file support (> 2G-bytes)

## International Solar-Terrestrial Physics/Space Physics Data Facility (ISTP/SPDF) standard:

“A CDF data set using ISTP guidelines, by definition forms a logically complete and self-sufficient whole (data and descriptions). The goal is to make the resulting CDF data set correctly and independently usable by the science community and accessible through the CDAWeb Display and Retrieval system”.

- Global attributes: Descriptions of the whole data set.
- Variables: data variables, support_data variables, and metadata variables including their dimensionality and what is needed for their correct display.
- Variables attributes: List and definition of attributes describing each variable.

## How to get data from your favorite spacecraft?

- Requests:
  - Get CDF file content or download CDF files from URL
- FTP:
  - Get CDF file content or download CDF files from FTP server
- Access space physics web services using:
  - speasy:
    - CDA, CSA, AMDA and SSC
  - pyspedas:
    - ERG, MAVEN, MMS, RBSP, THEMIS, etc. through URL and FTP
  - spacepy
  - PyRFU
  - …

# xarray: Timeseries, Spectrograms, VDFs & co.

Labelled multi-dimensional (a.k.a. N-dimensional, ND) arrays in Python

xarray.DataArray: A multi-dimensional array with labeled or named dimensions. DataArray objects add metadata such as dimension names, coordinates, and attributes to underlying “unlabeled” data structures such as numpy arrays. If its optional name property is set, it is a named DataArray.

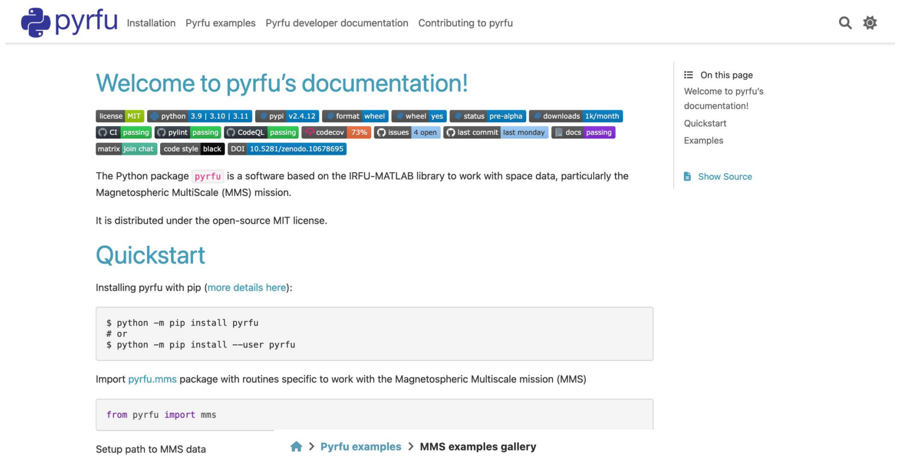
>> Timeseries, spectrograms*

xarray.Dataset: A dict-like collection of DataArray objects with aligned dimensions. Thus, most operations that can be performed on the dimensions of a single DataArray can be performed on a dataset. Datasets have data variables, dimensions, coordinates, and attributes.

>> spectrograms, VDFs

# PyRFU

- Python package to work with space plasma data
- Based on irfu-matlab + new routines
- >16.5k lines of code (75% coverage)
- Heavily documented:
  - >9k lines of comments/docstring
  - 18 examples
- pip installable
- Part of the Python in Heliophysics Community (PyHC) and the HelioCloud project.



## Welcome to pyrfu's documentation!

The Python package `pyrfu` is a software based on the IRFU-MATLAB library to work with space data, particularly the Magnetospheric MultiScale (MMS) mission.

It is distributed under the open-source MIT license.

## Quickstart

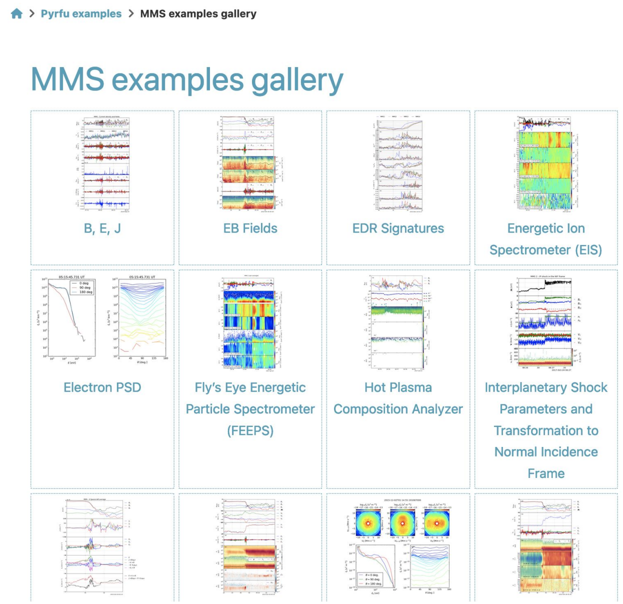
Installing pyrfu with pip (more details here):

```
$ python -m pip install pyrfu
# or
$ python -m pip install --user pyrfu
```

Import pyrfu.mms package with routines specific to work with the Magnetospheric Multiscale mission (MMS)

```
from pyrfu import mms
```

Setup path to MMS data


Pyfru examples > MMS examples gallery

## MMS examples gallery

B, E, J
EB Fields
EDR Signatures
Energetic Ion Spectrometer (EIS)
Electron PSD
Fly's Eye Energetic Particle Spectrometer (FEEPS)
Hot Plasma Composition Analyzer
Interplanetary Shock Parameters and Transformation to Normal Incidence Frame

## Backup material

# Conda

Conda is a package & environment manager together (not just for python), with a focus on data analysis tools.

1) install python 3.11 miniconda (lightweight, nothing extra included by default)
2) Run…

```
$ conda env create --name irfu_python_ws python=3.11 jupyterlab ipympl numpy scipy
matplotlib

$ conda activate irfu_python_ws

$ which jupyter # to check it is pointing to */irfu_python_ws/bin/jupyter

$ jupyter lab
```

… to create a new environment called **irfu_python_ws** with some installed packages, "**activate**" it, and start jupyter there. Packages in this environment are only available within it, and you can have multiple such environments without polluting others.

But now, you must hide from @Louis…