

Ingredients for successful domain specific research software

David Stansby — d.stansby@ucl.ac.uk



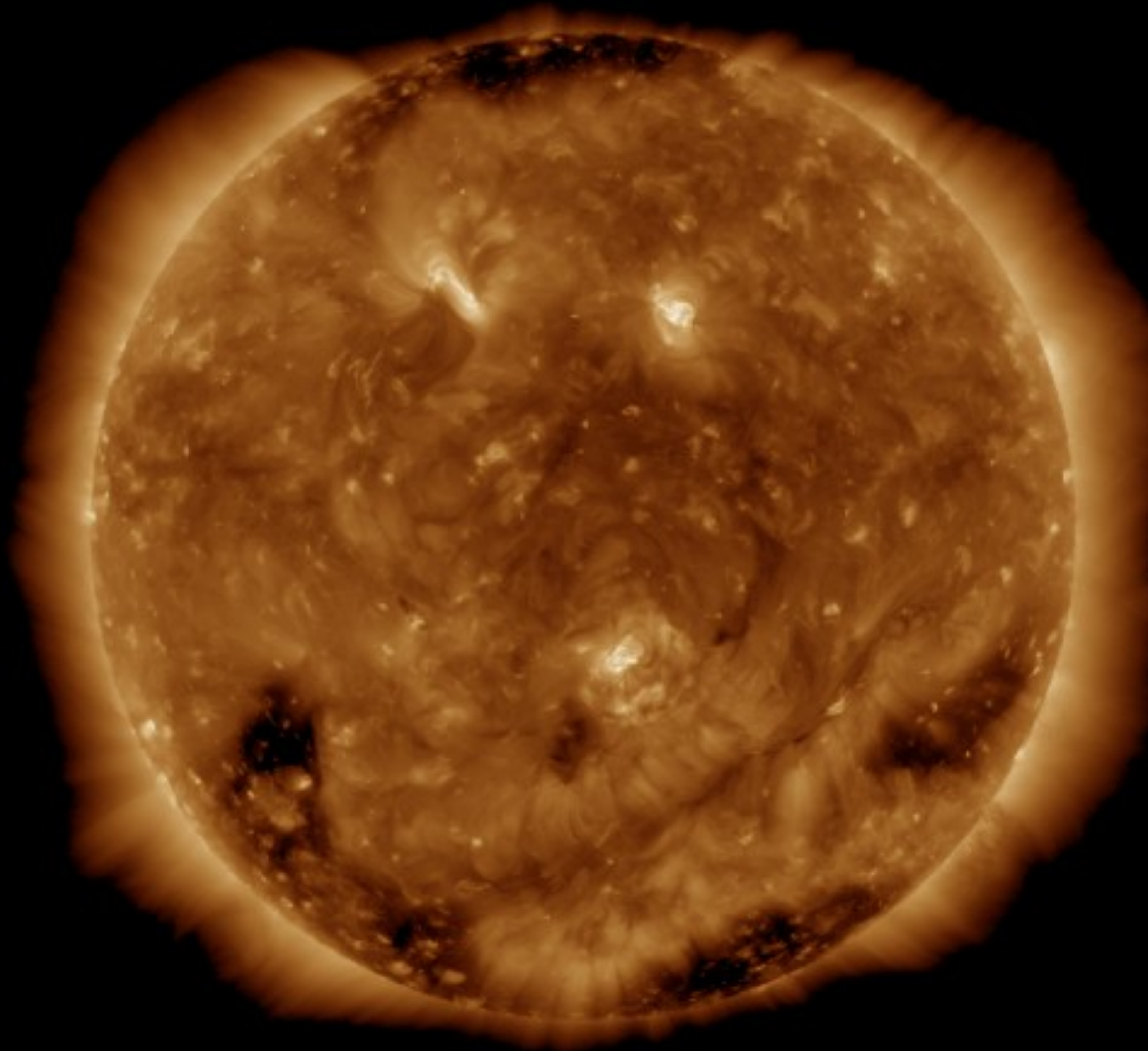
The surface of the Sun is hot!
(~ 6000 K)



The surface of the Sun is hot!
(~ 6000 K)

**The atmosphere of the Sun
is even hotter!**
(~ 1,000,000 K)

Why?



**Why is the Sun's atmosphere
so hot?**

**Is heating correlated with
magnetic field strength?**

Model the magnetic field

Question

Hypothesis

Task

**As a scientist, I'm paid
to do these**

**Why is the Sun's atmosphere
so hot?**

Question

**Is heating correlated with
magnetic field strength?**

Hypothesis

**As a scientist, I'm paid
to do these**

Model the magnetic field

Task

using Python

converted to byte code

run on a C virtual machine

compiled to machine code

that runs on transistors

made from atoms

that obey quantum mechanics

...

Layers

**Why is the Sun's atmosphere
so hot?**

**Is heating correlated with
magnetic field strength?**

Model the magnetic field

using Python

converted to byte code

run on a C virtual machine

compiled to machine code

that runs on transistors

made from atoms

that obey quantum mechanics

...

Layers

Why is the Sun's atmosphere
so hot?

Is heating correlated with
magnetic field strength?

Model the magnetic field
using Python

converted to byte code

run on a C virtual machine

compiled to machine code

that runs on transistors

made from atoms

that obey quantum mechanics

...

The layer above what
we're doing is motivation

The layer below what
we're doing are tools

Layers

Is heating correlated with magnetic field strength?

Model the magnetic field



Research software goes here

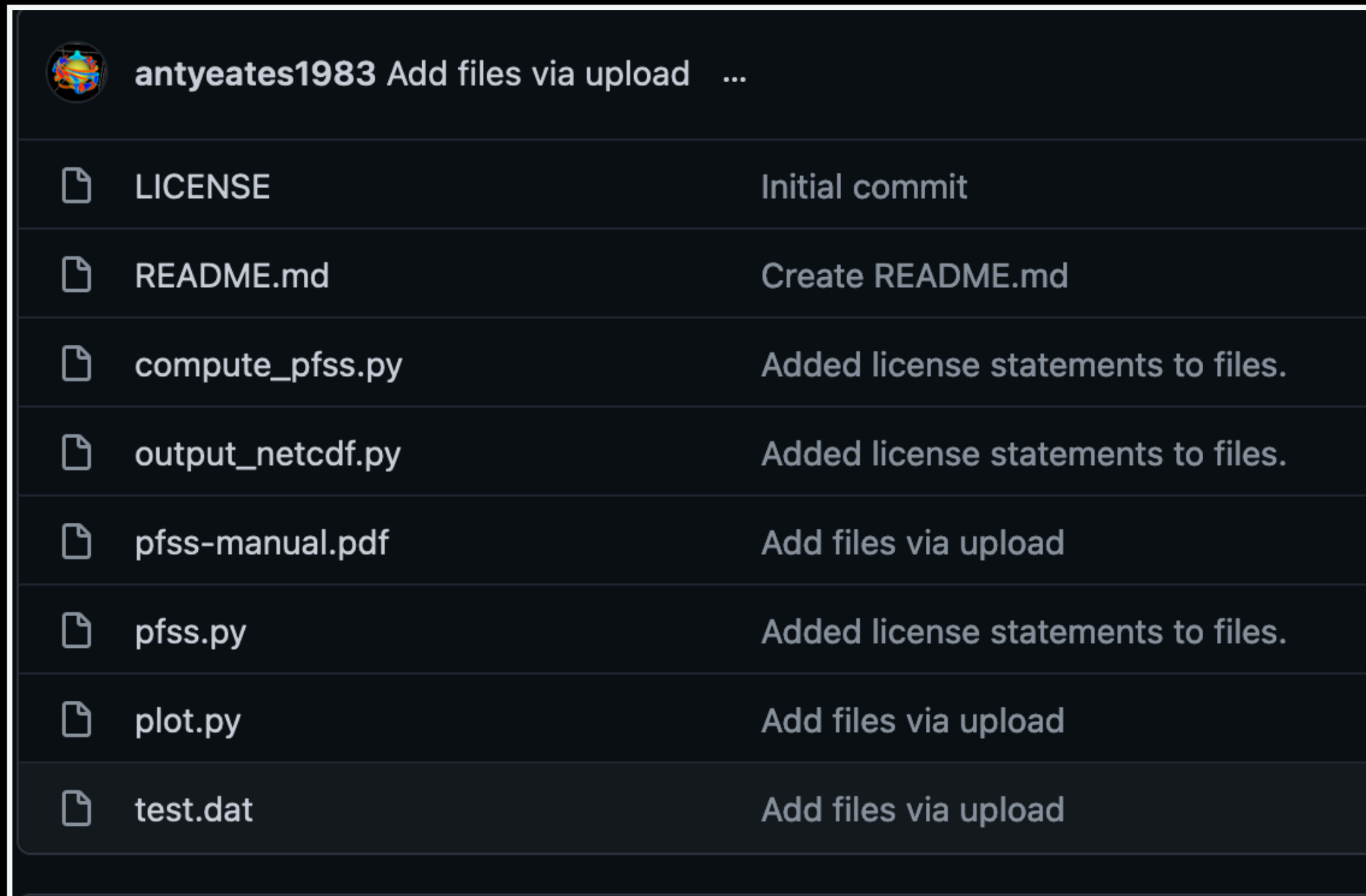
using Python

- When is it worth adding another layer?
- It takes time/effort to make software
- Key question:

Will adding a layer be a net time saving in the future?

- Take into account time taken to write, maintain
- Take into account how widely it will be used

The state in 2017

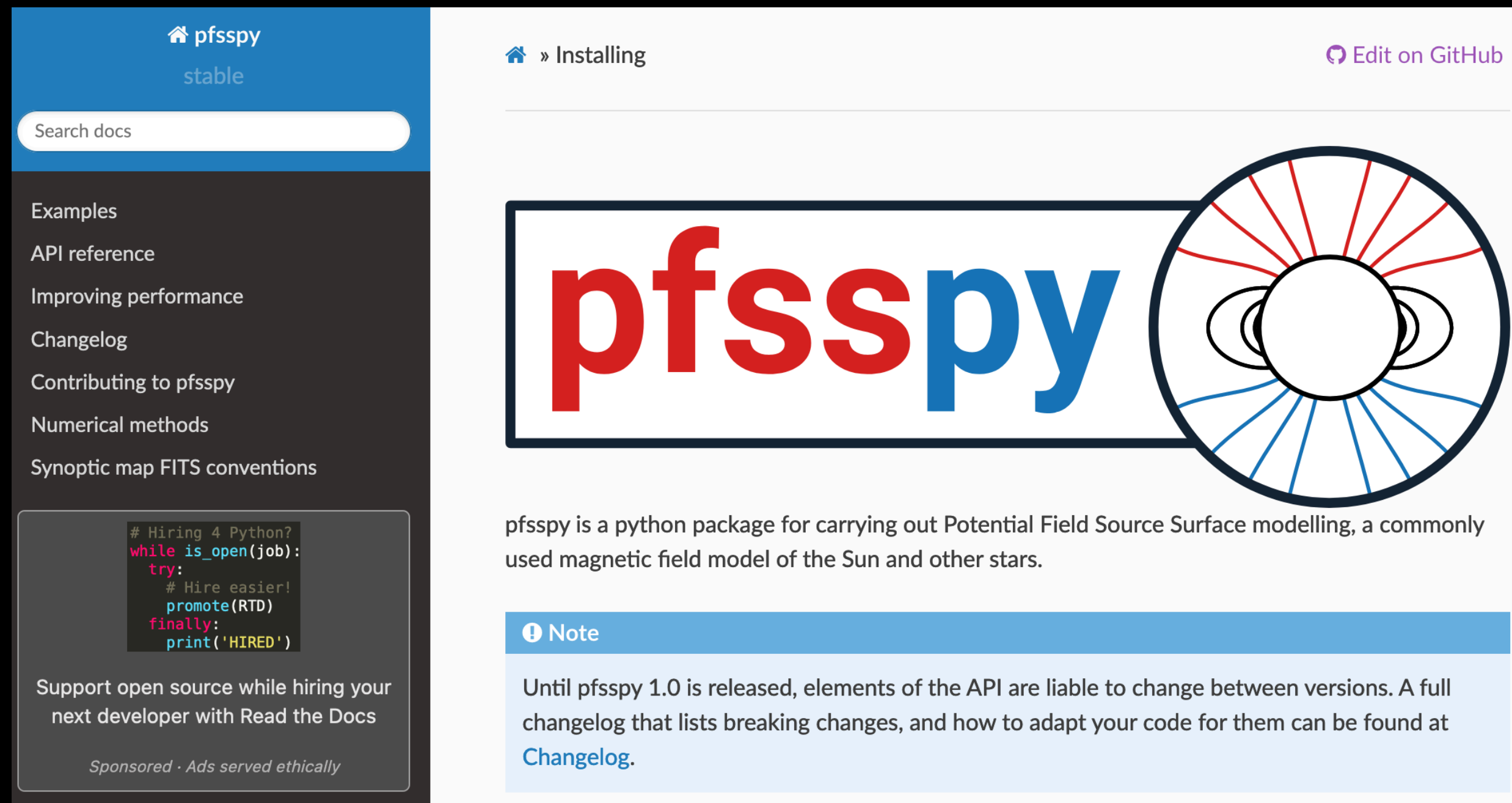


File	Commit Message
LICENSE	Initial commit
README.md	Create README.md
compute_pfss.py	Added license statements to files.
output_netcdf.py	Added license statements to files.
pfss-manual.pdf	Add files via upload
pfss.py	Added license statements to files.
plot.py	Add files via upload
test.dat	Add files via upload

<https://github.com/antyeates1983/pfss>

- 816 citations for original method paper (published 1969)
- A Python implementation is released in 2017 🎉
- 1 function, ~250 line script
- Well documented
- But, no
 - Examples, tests, input data cleaning, integration w/ other packages

The state in 2021



The screenshot shows the pfsspy documentation website. The left sidebar contains navigation links: Examples, API reference, Improving performance, Changelog, Contributing to pfsspy, Numerical methods, and Synoptic map FITS conventions. Below these is a code snippet for hiring a developer with Read the Docs. The main content area is titled 'Installing' and features the pfsspy logo, which consists of the text 'pfsspy' in red and blue, followed by a circular diagram with red and blue lines radiating from a central point. Below the logo is a description: 'pfsspy is a python package for carrying out Potential Field Source Surface modelling, a commonly used magnetic field model of the Sun and other stars.' A 'Note' box states: 'Until pfsspy 1.0 is released, elements of the API are liable to change between versions. A full changelog that lists breaking changes, and how to adapt your code for them can be found at [Changelog](#).'

- A full blown python package
- 11 files, 3039 lines of code
- 11 examples
- 1 paper in Journal of Open Source Software
- Full integration with astropy, sunpy
- An excellent distraction from my thesis

pfsspy.readthedocs.io

So, what did I add, and why?

Versioning + changelog

API reference
Improving performance

Changelog

- 0.6.5
- 0.6.4
- 0.6.3
- 0.6.2
- 0.6.1
- 0.6.0
- 0.5.3
- 0.5.2
- 0.5.1
- 0.5.0
- 0.4.3
- 0.4.2
- 0.4.1
- 0.4.0
- 0.3.2
- 0.3.1
- 0.3.0
- 0.2.0
- 0.1.5
- 0.1.4
- 0.1.3

Read the Docs v: stable

0.5.0

Changes to outputted maps

This release largely sees a transition to leveraging Sunpy Map objects. As such, the following changes have been made:

`pfsspy.Input` now *must* take a `sunpy.map.GenericMap` as an input boundary condition (as opposed to a numpy array). To convert a numpy array to a `GenericMap`, the helper function `pfsspy.carr_cea_wcs_header` can be used:

```
map_date = datetime(...)
br = np.array(...)
header = pfsspy.carr_cea_wcs_header(map_date, br.shape)

m = sunpy.map.Map((br, header))
pfss_input = pfsspy.Input(m, ...)
```

`pfsspy.Output.source_surface_br` now returns a `GenericMap` instead of an array. To get the data array use `source_surface_br.data`.

The new `pfsspy.Output.source_surface_pils` returns the coordinates of the polarity inversion lines on the source surface.

In favour of directly using the plotting functionality built into SunPy, the following plotting functionality has been removed:

- `pfsspy.Input.plot_input`. Instead `Input` has a new `map` property, which returns a SunPy map, which can easily be plotted using `sunpy.map.GenericMap.plot`.
- `pfsspy.Output.plot_source_surface`. A map of B_r on the source surface can now be obtained using `pfsspy.Output.source_surface_br`, which again returns a SunPy map.

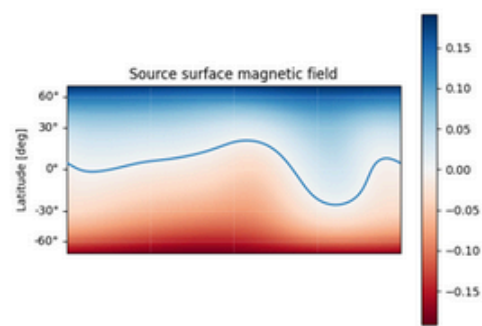
Allows users to stick to one version for reproducibility

Tell users exactly why to update, how to update

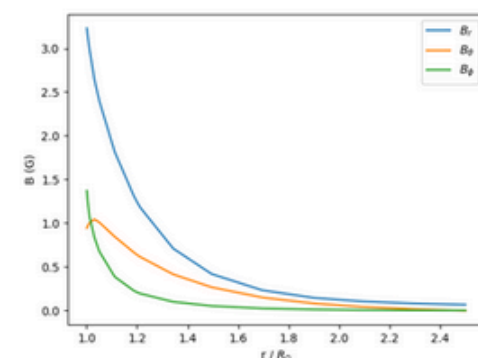
Examples

Examples

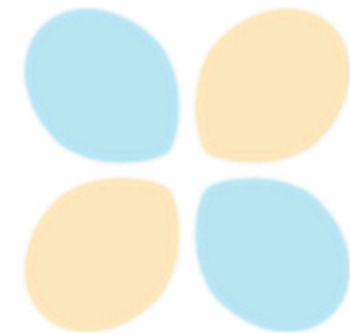
Using pfsspy



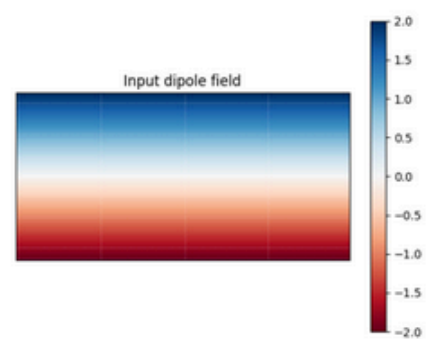
HMI PFSS solutions



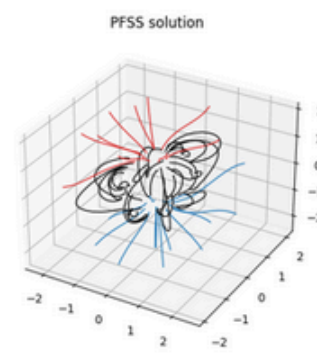
Magnetic field along a field line



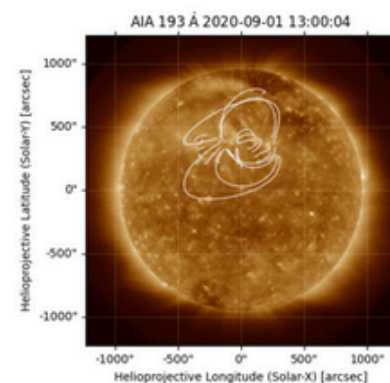
Open/closed field map



Dipole source solution



GONG PFSS extrapolation



Overplotting field lines on AIA maps

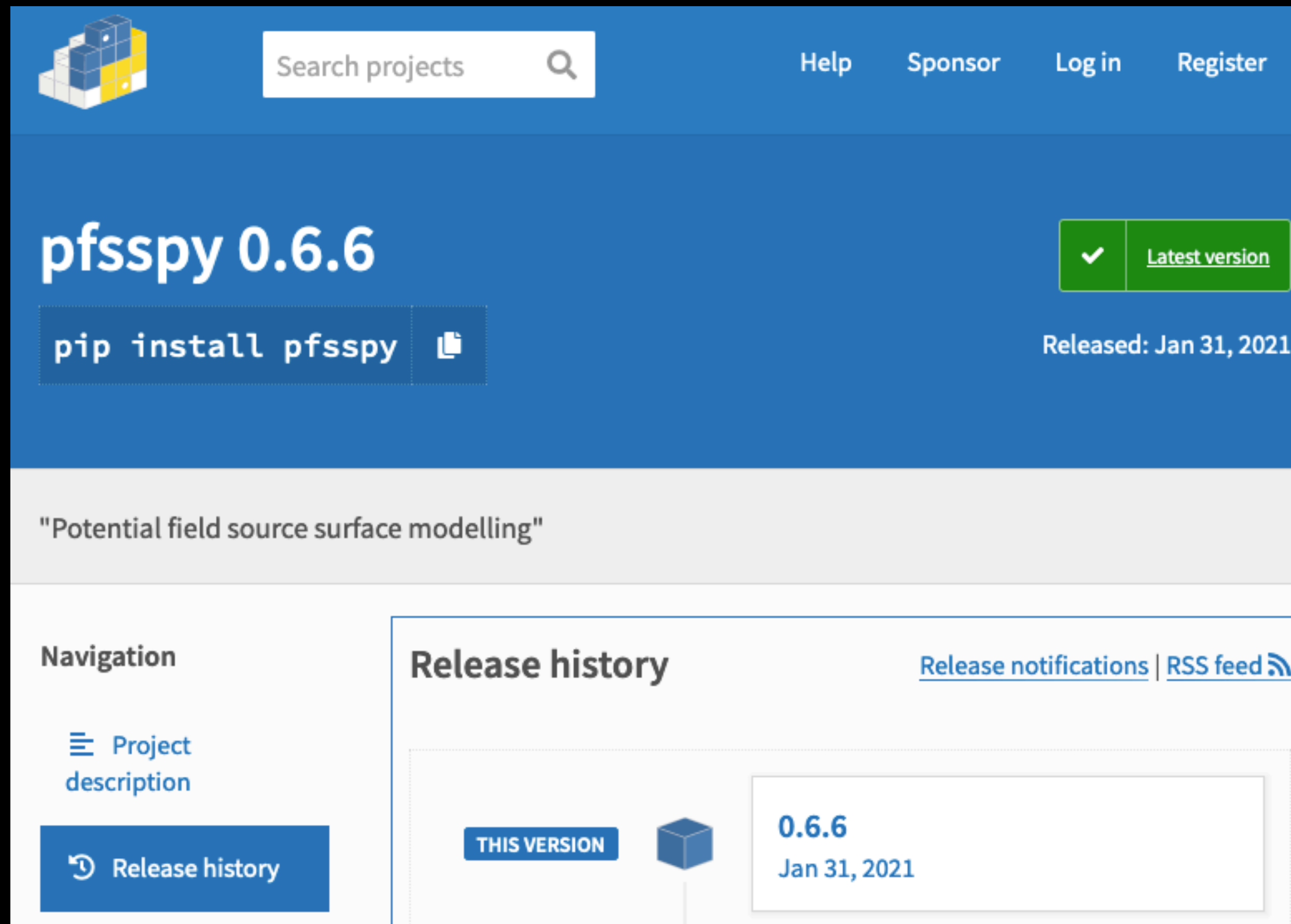
Reduces barrier to use

Gives a recipe that users can adapt for their situation

The best introduction to a package

Gives you confidence that your package is doing what you expect!

Package



The screenshot shows the PyPI package page for 'pfsspy 0.6.6'. At the top, there is a search bar and navigation links for 'Help', 'Sponsor', 'Log in', and 'Register'. The package name 'pfsspy 0.6.6' is prominently displayed, along with a green checkmark and 'Latest version' badge. Below this, the installation command 'pip install pfsspy' is shown with a copy icon, and the release date 'Released: Jan 31, 2021' is noted. The package description is '"Potential field source surface modelling"'. A navigation sidebar on the left includes 'Project description' and 'Release history'. The 'Release history' section shows a table with one entry: '0.6.6' released on 'Jan 31, 2021', marked as 'THIS VERSION'.

Version	Released
0.6.6	Jan 31, 2021

Gives users a common method to install and use code

Aids reproducibility

Documented API

Input

`class pfsspy.Input(br, nr, rss)`

Bases: `object`

Input to PFSS modelling.

Warning

The input must be on a regularly spaced grid in ϕ and $s = \cos(\theta)$. See `pfsspy.grid` for more information on the coordinate system.

- Parameters:
- `br` (`sunpy.map.GenericMap`) – Boundary condition of radial magnetic field at the inner surface. Note that the data *must* have a cylindrical equal area projection.
 - `nr` (`int`) – Number of cells in the radial direction to calculate the PFSS solution on.
 - `rss` (`float`) – Radius of the source surface, as a fraction of the solar radius.

Attributes Summary

`map`

`sunpy.map.GenericMap` representation of the input.

Attributes Documentation

`map`

`sunpy.map.GenericMap` representation of the input.

Gives advanced users an overview of what they can change and how

Tests

```
1  ▶ Run pytest --cov-report=xml
7  ===== test session starts =====
8  platform linux -- Python 3.6.12, pytest-6.2.2, py-1.10.0, pluggy-0.13.1
9  rootdir: /home/runner/work/pfsspy/pfsspy, configfile: setup.cfg
10 plugins: cov-2.11.1
11 collected 29 items
12
13 pfsspy/tests/test_coords.py . [ 3%]
14 pfsspy/tests/test_fieldline.py ..... [ 20%]
15 pfsspy/tests/test_map.py .. [ 27%]
16 pfsspy/tests/test_pfss.py ..... [ 68%]
17 pfsspy/tests/test_tracers.py ... [ 79%]
18
19 pfsspy/tests/test_utils.py ..... [100%]
20
21 ----- coverage: platform linux, python 3.6.12-final-0 -----
22 Coverage XML written to file coverage.xml
23
```

Ensures package is working as intended

Makes sure you don't break your API...

...or if you do, it is intended and understood

Components of a research package

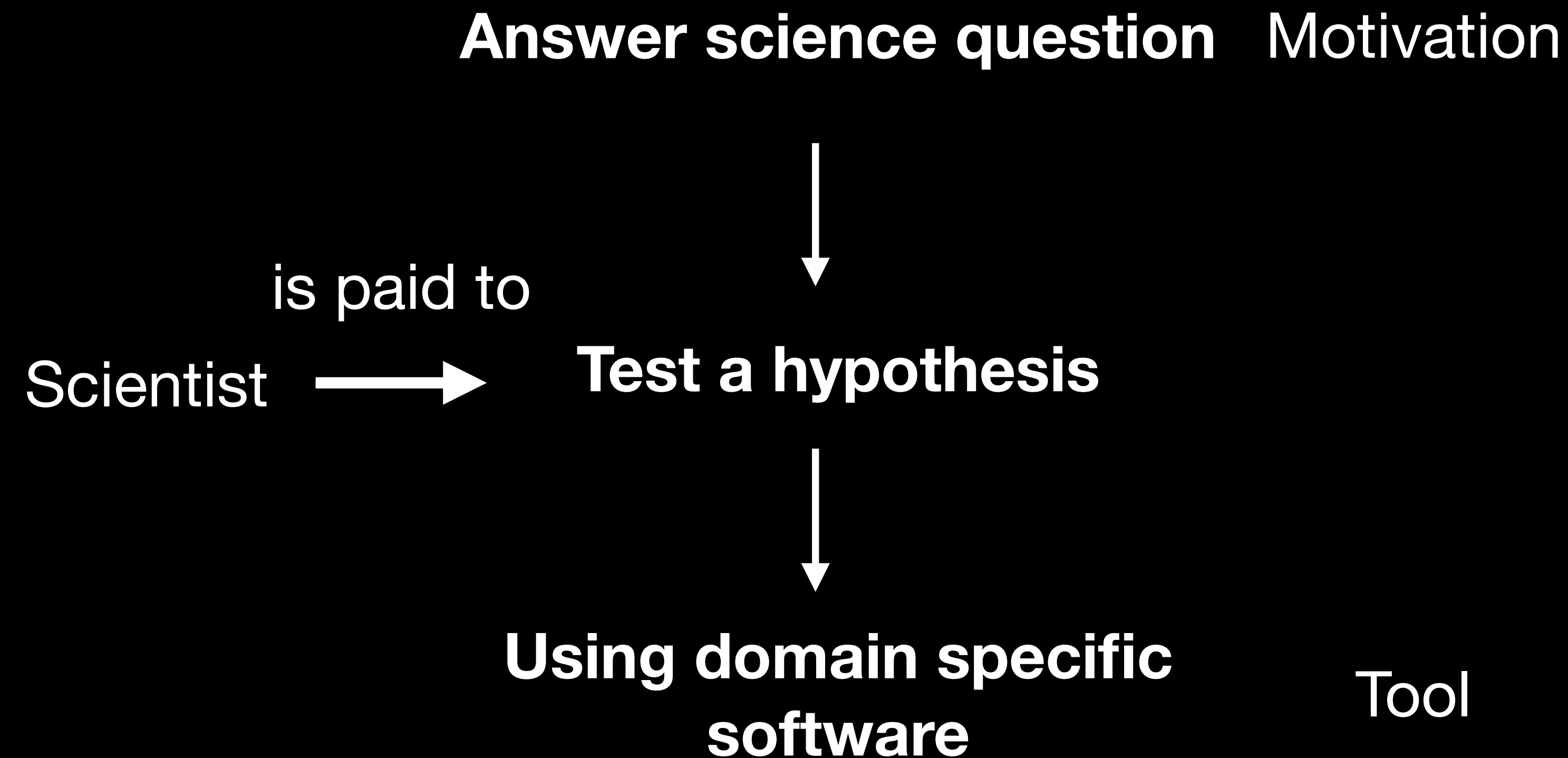
In order of importance:

- Exists
- Versioned
- Changelog
- Examples
- Package
- API docs
- Tests

These aid both **usability**
and **reproducibility**

What skills do you
need to do this,
and how to develop
them?

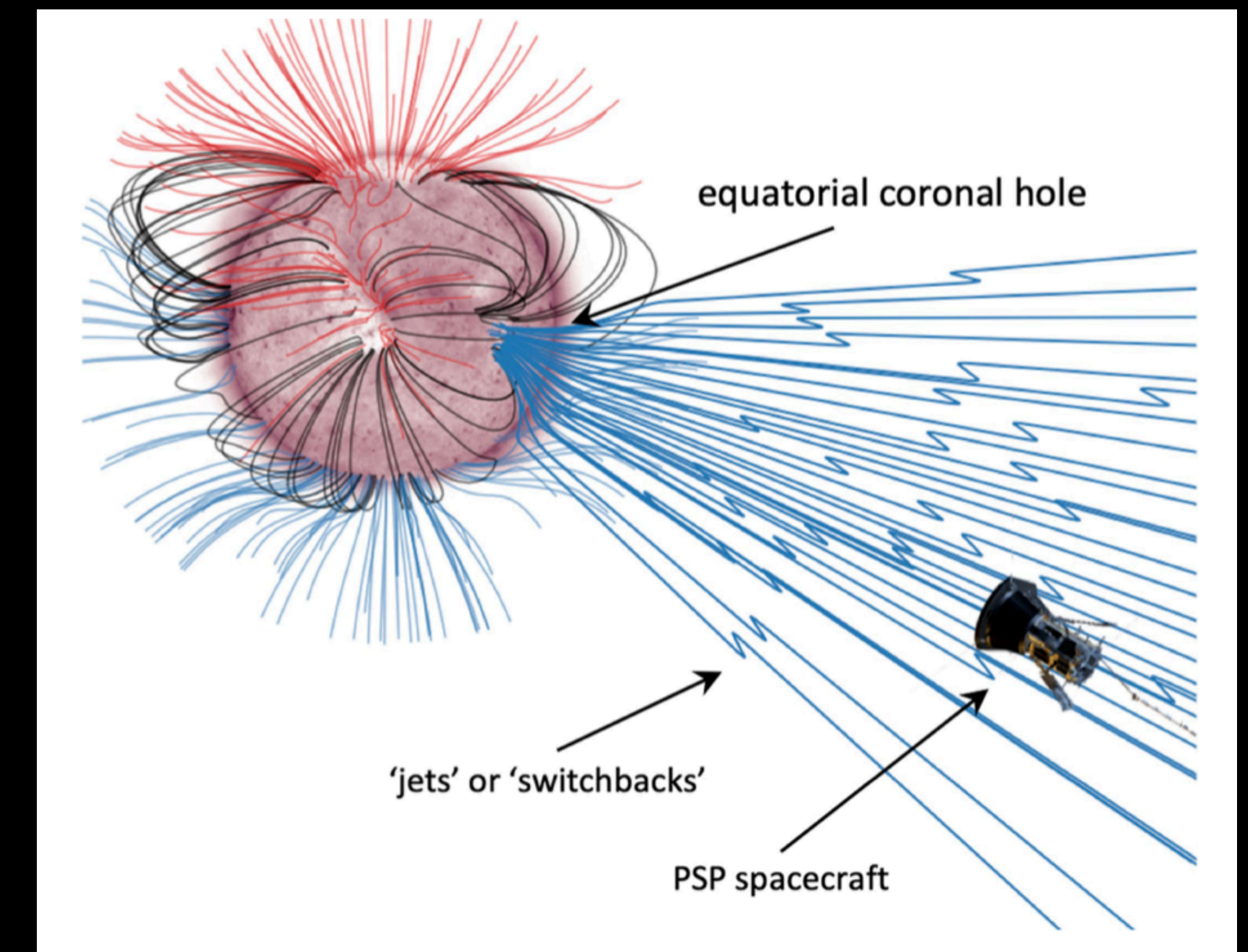
Back to layers



- Too add a layer, you have to know about layer above (motivation) and layer below (tools)
- Move towards Research Software Engineers (RSEs), who are experts in tools
- ...but scientists are experts in motivation
 - ▶ Teach scientists about tools
 - ▶ Teach RSEs about motivation

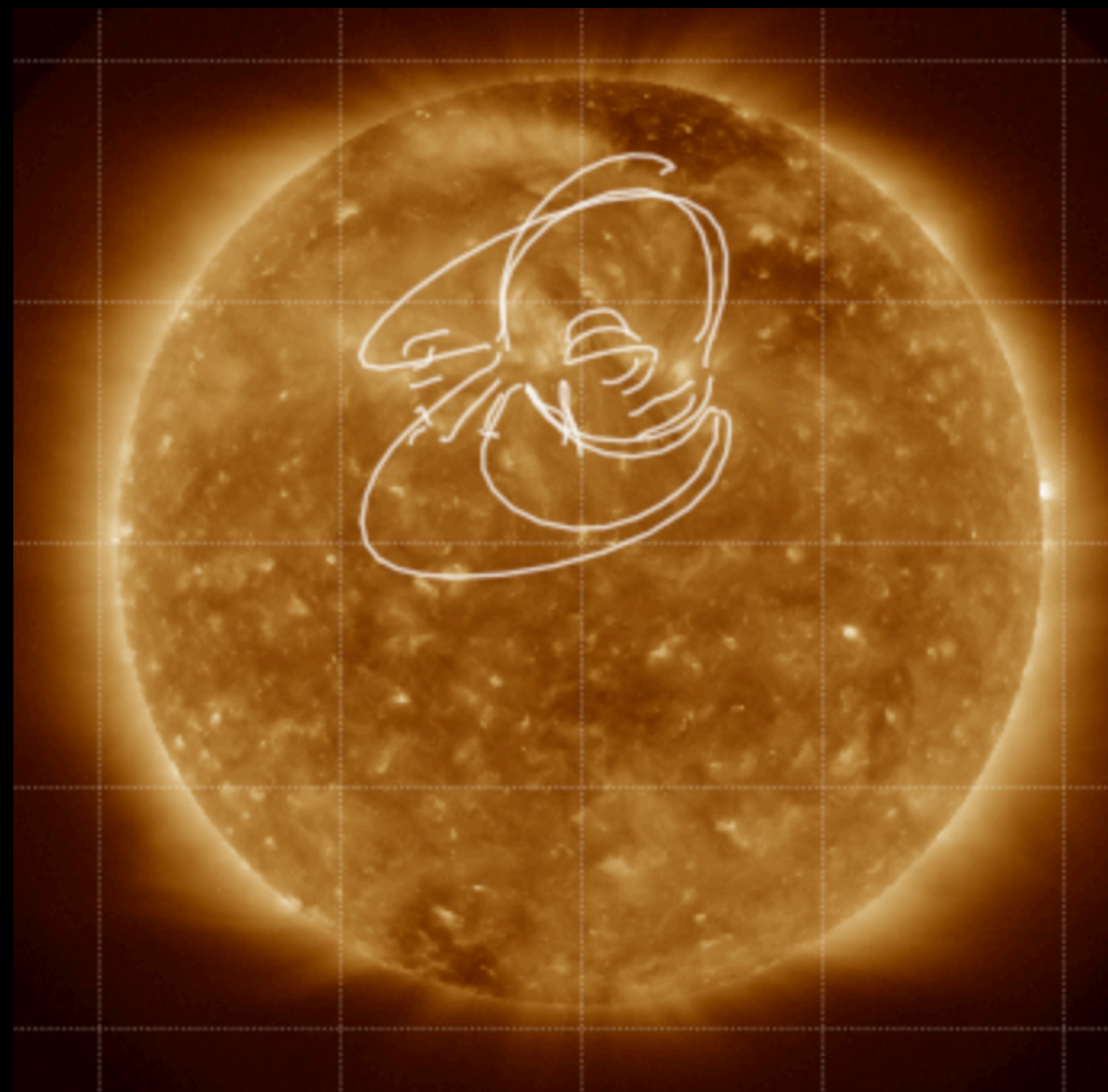
So was pfsspy successful?

- Used in 18 papers (and counting!)
- Critical for interpreting results from Parker Solar Probe, NASA's \$2bn mission to "Touch the Sun"
- It took a lot of unfunded work to get here; I was lucky to have time and flexibility
- I think unique position as both scientist **and** software engineer helped make package useable for a wide community



Successful research software

- Performs a specific task (not one hypothesis or question)
- Performs a widely used task



In order of importance:

- Exists
 - Versioned
 - Changelog
 - Examples
 - Package
 - API docs
 - Tests
- Need to nurture (and ideally teach) these practices at PhD level
 - Recognise that s/w development improves research efficiency
 - Good steps being taken by NASA in US and UKRI/EPSCRC in UK