Ingredients for successful domain specific research software

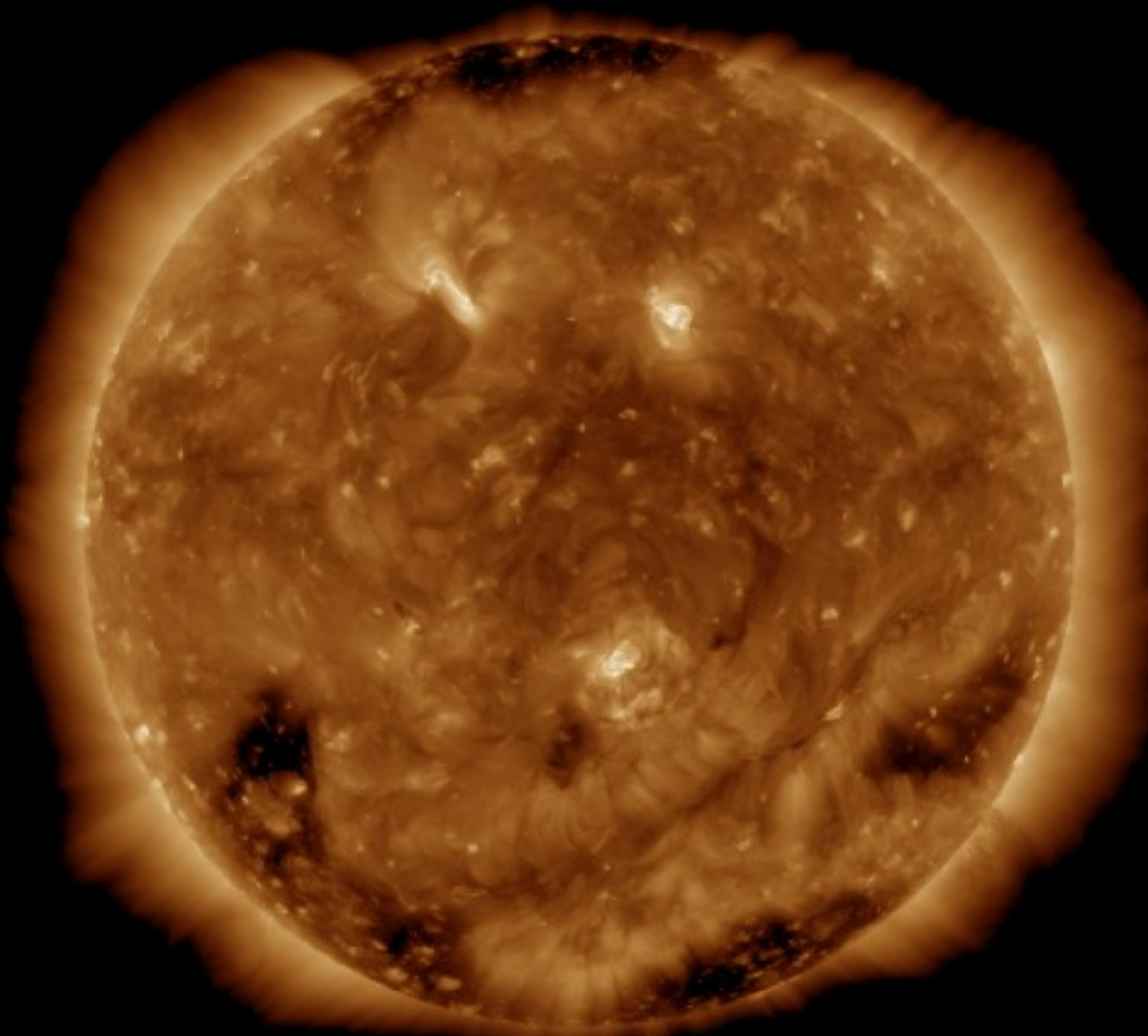
David Stansby — d.stansby@ucl.ac.uk

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?

Question

Is heating correlated with magnetic field strength?

Hypothesis

Model the magnetic field

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

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

...

As a scientist, I'm paid to do these

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

…

They 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

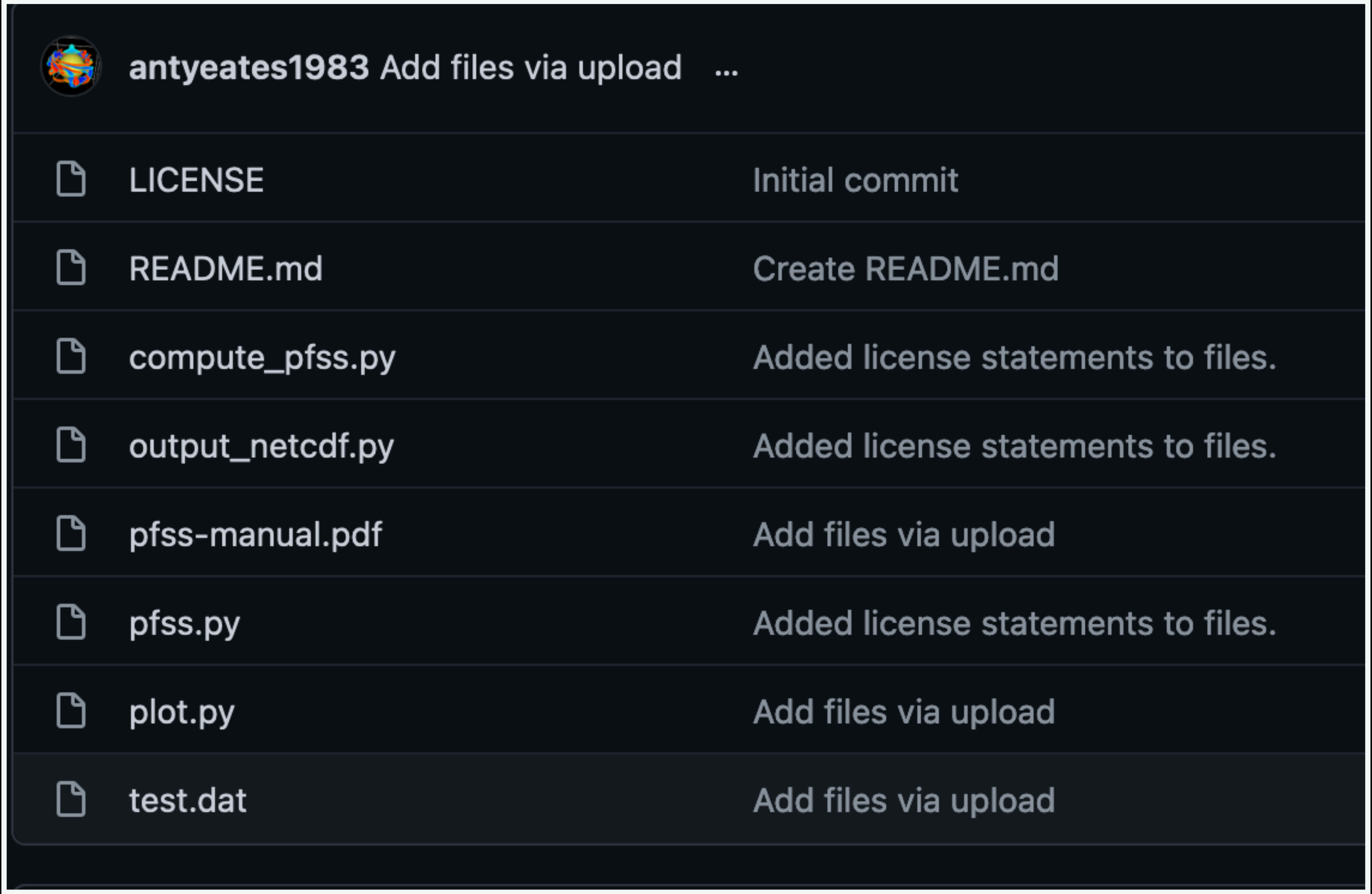
using Python

- 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

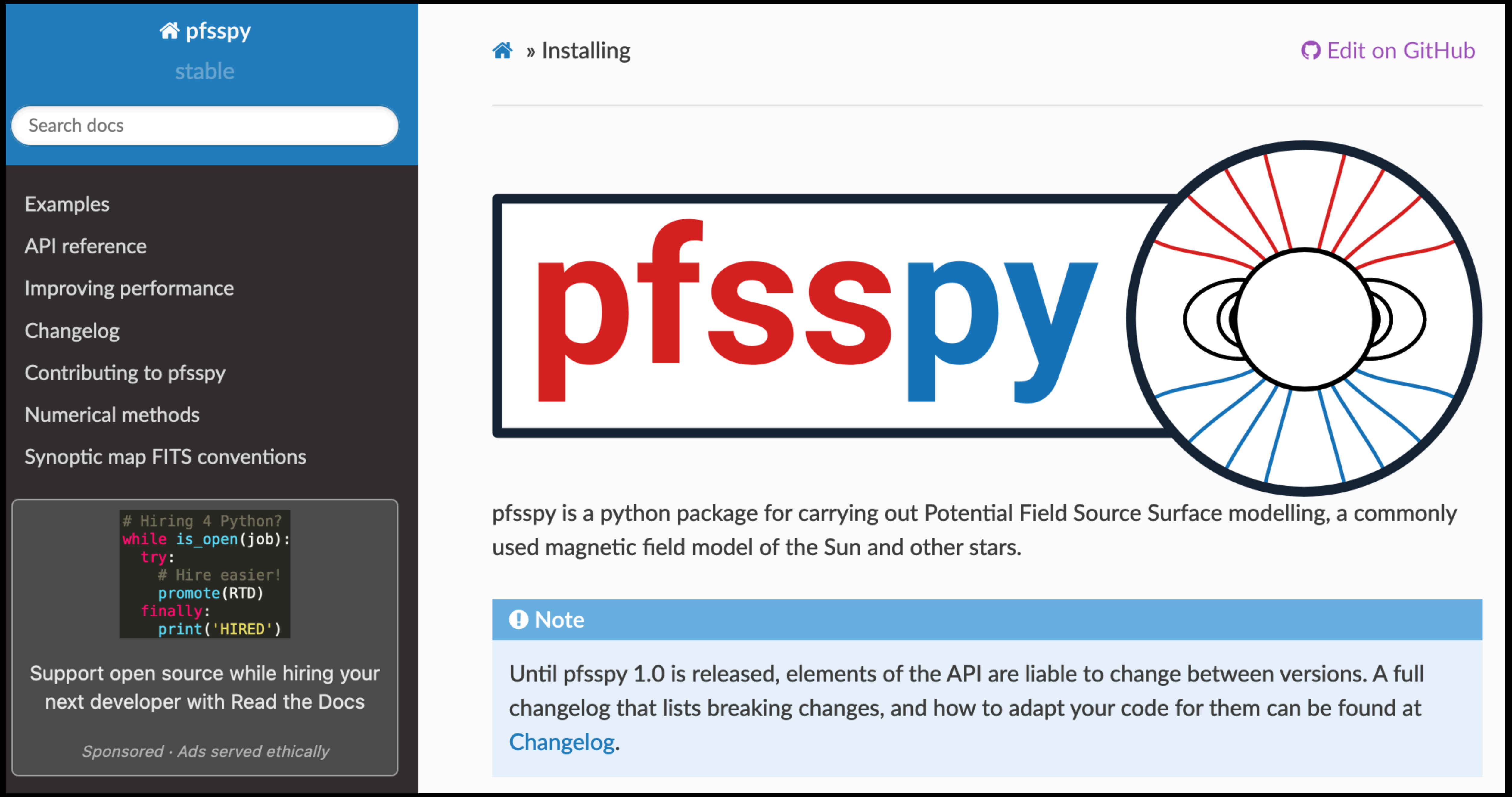
The state in 2017



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



pfsspy.readthedocs.io

- 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

So, what did I add, and why?

Versioning + changelog

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:

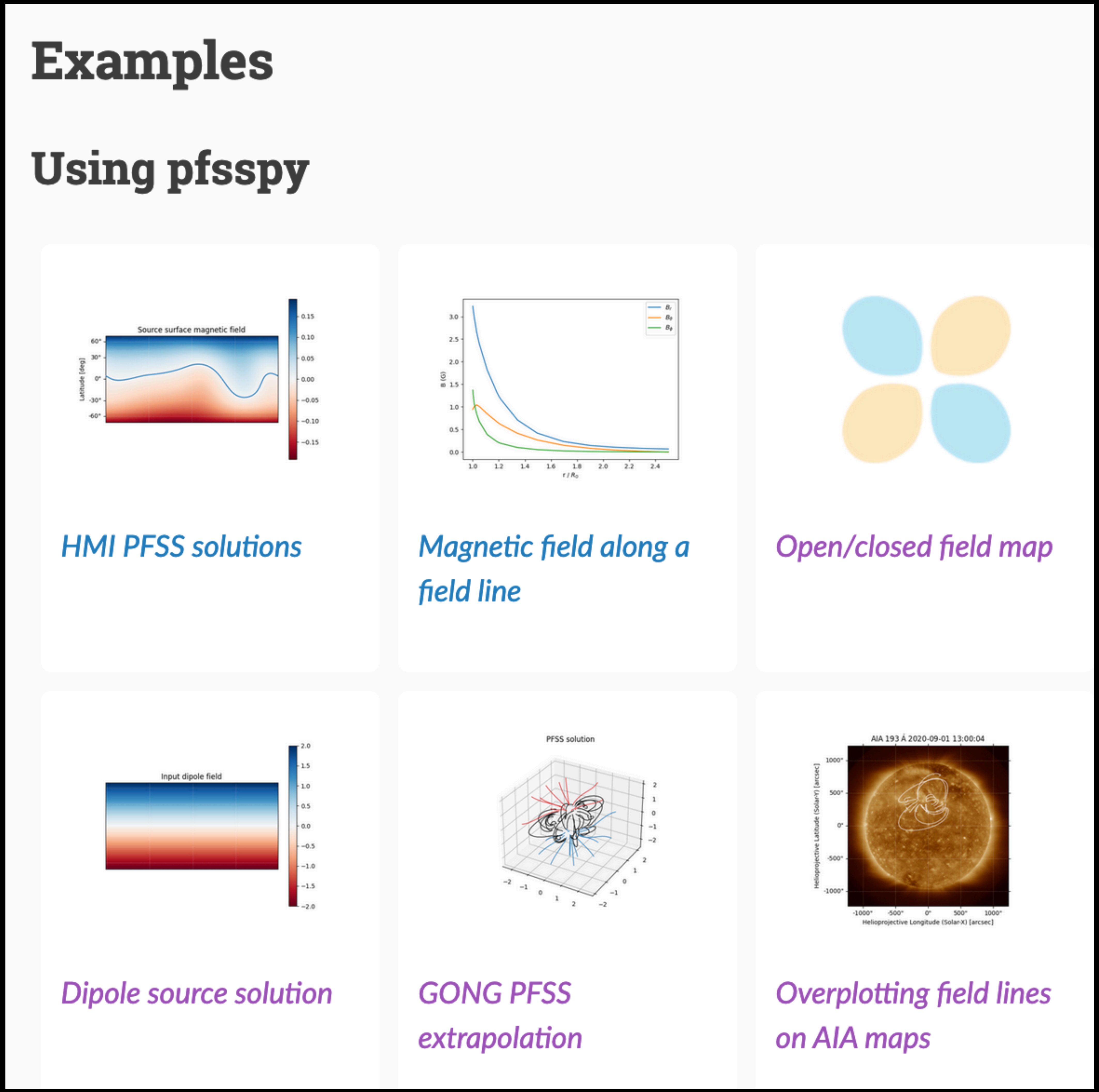
- `pfsspy.Input.plot_input`. Instead `Input` has a new `map` property, which returns a SunPy map, which can easily be plotted using `sunpy.map.GenericMap.plot`.
- `pfsspy.Output.plot_source_surface`. A map of B_r on the source surface can now be obtained using `pfsspy.Output.source_surface_br`, which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

https://pfsspy.readthedocs.io/en/stable/changes.html

Allows users to stick to one version for reproducibility

Tell users exactly **why** to update, **how** to update

Examples



https://pfsspy.readthedocs.io/en/stable/auto_examples/index.html

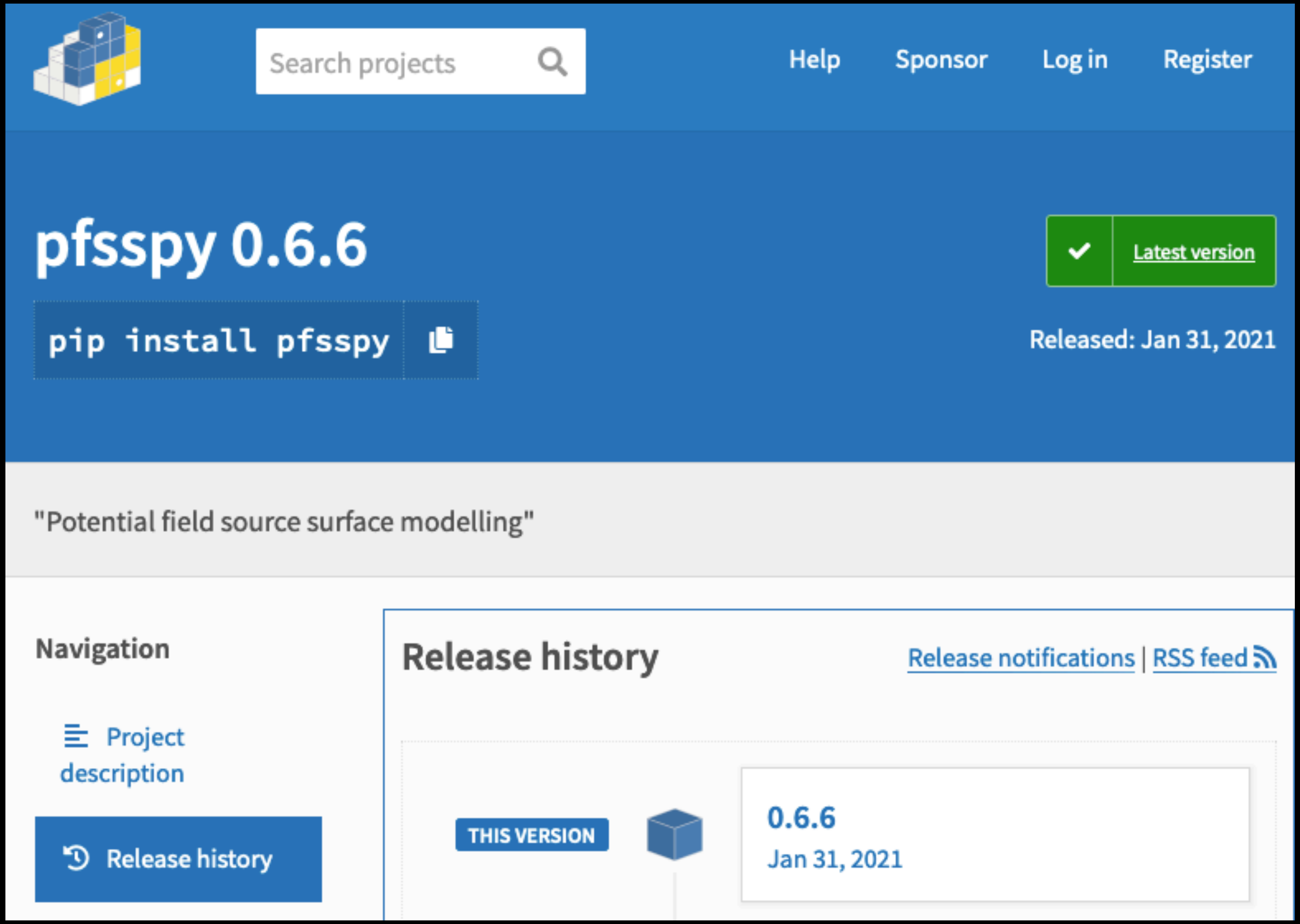
Reduces barrier to use

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



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

```
▸ Run pytest --cov-report=xml
============================= test session starts ==============================
platform linux -- Python 3.6.12, pytest-6.2.2, py-1.10.0, pluggy-0.13.1
rootdir: /home/runner/work/pfsspy/pfsspy, configfile: setup.cfg
plugins: cov-2.11.1
collected 29 items

pfsspy/tests/test_coords.py .                                            [  3%]
pfsspy/tests/test_fieldline.py .....                                     [ 20%]
pfsspy/tests/test_map.py ..                                              [ 27%]
pfsspy/tests/test_pfss.py ............                                   [ 68%]
pfsspy/tests/test_tracers.py ...                                         [ 79%]

pfsspy/tests/test_utils.py ......                                        [100%]

---------- coverage: platform linux, python 3.6.12-final-0 -----------
Coverage XML written to file coverage.xml
```

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

Answer science question Motivation

is paid to

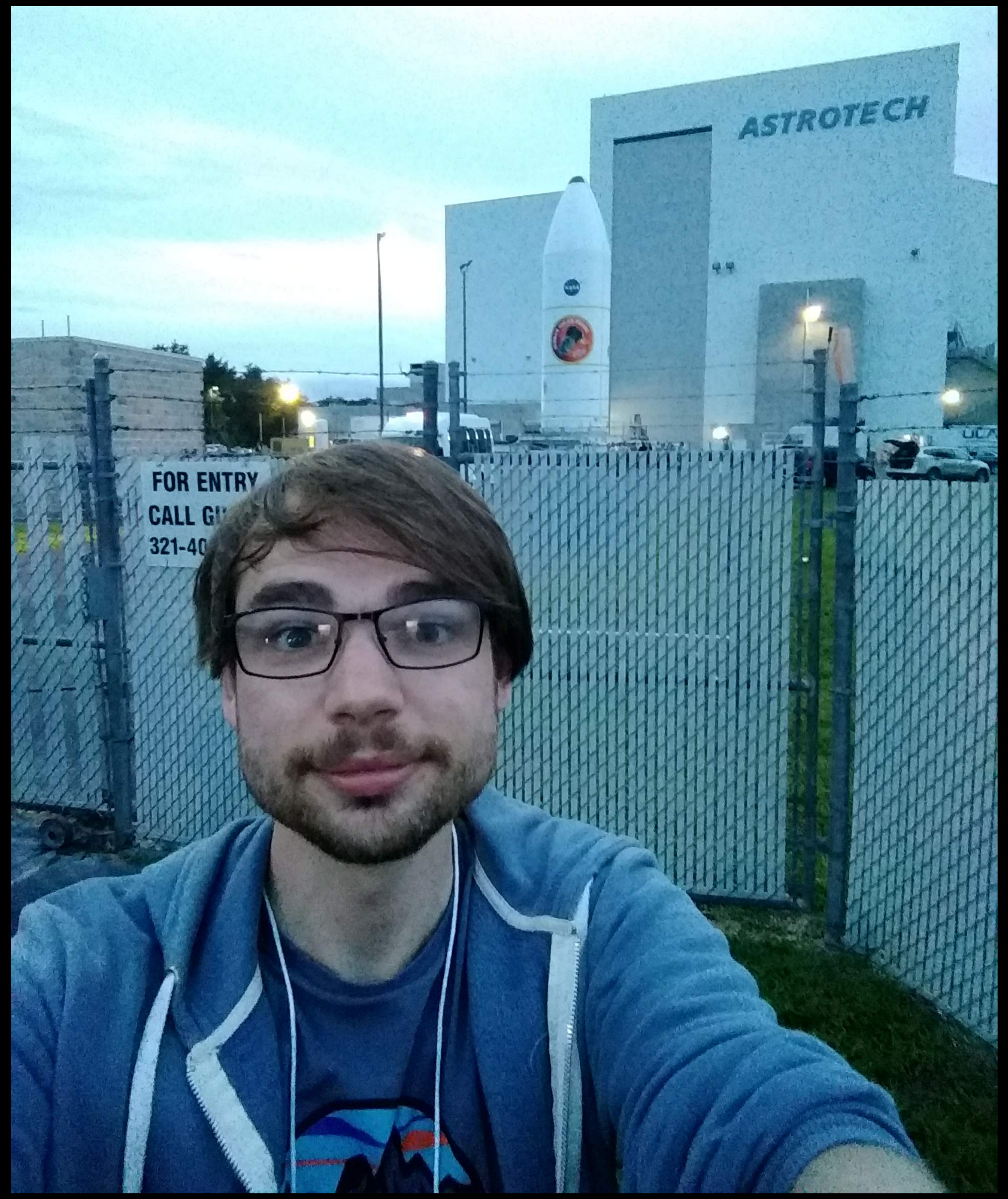
Scientist → **Test a hypothesis**

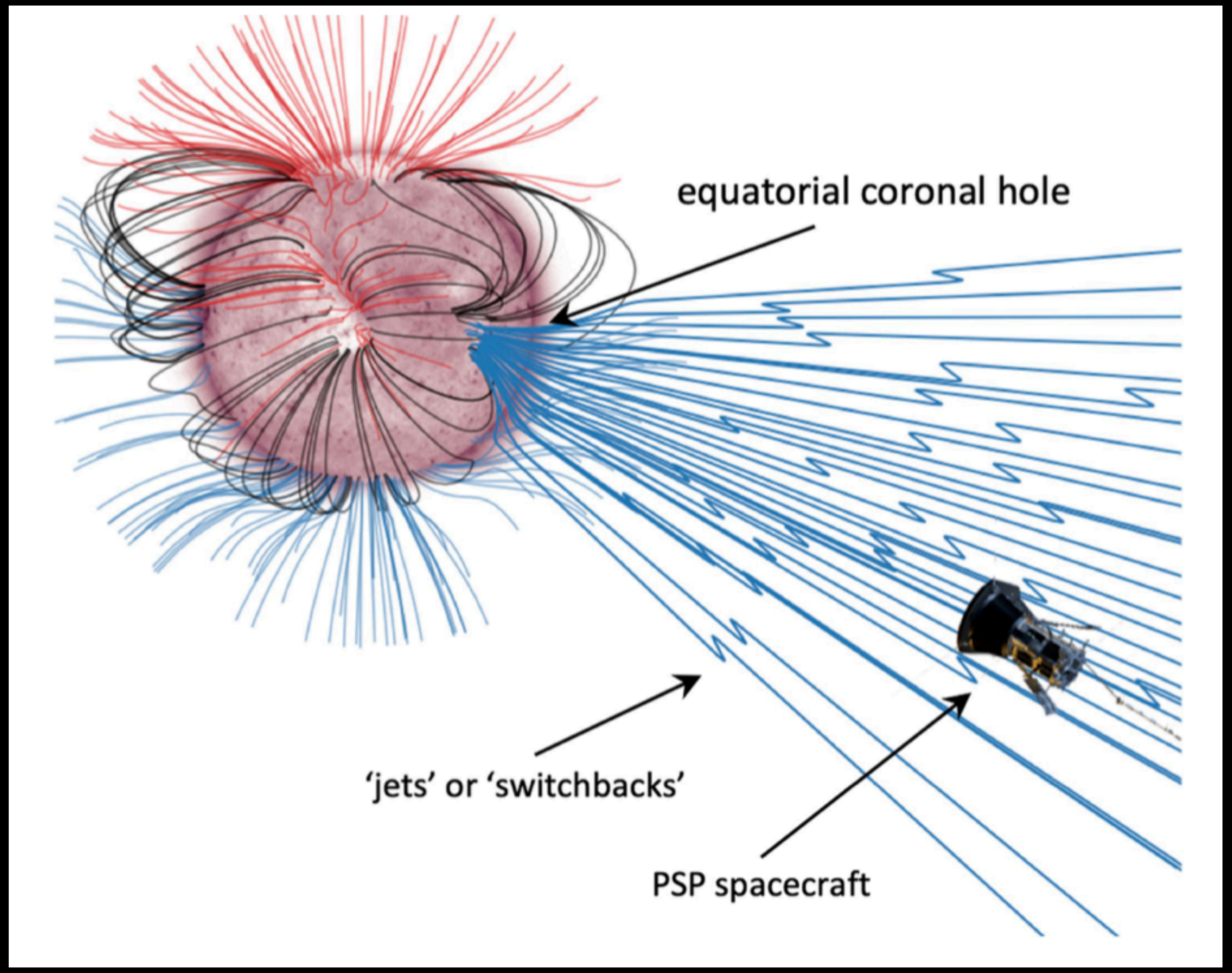
Using domain specific software Tool

- 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?

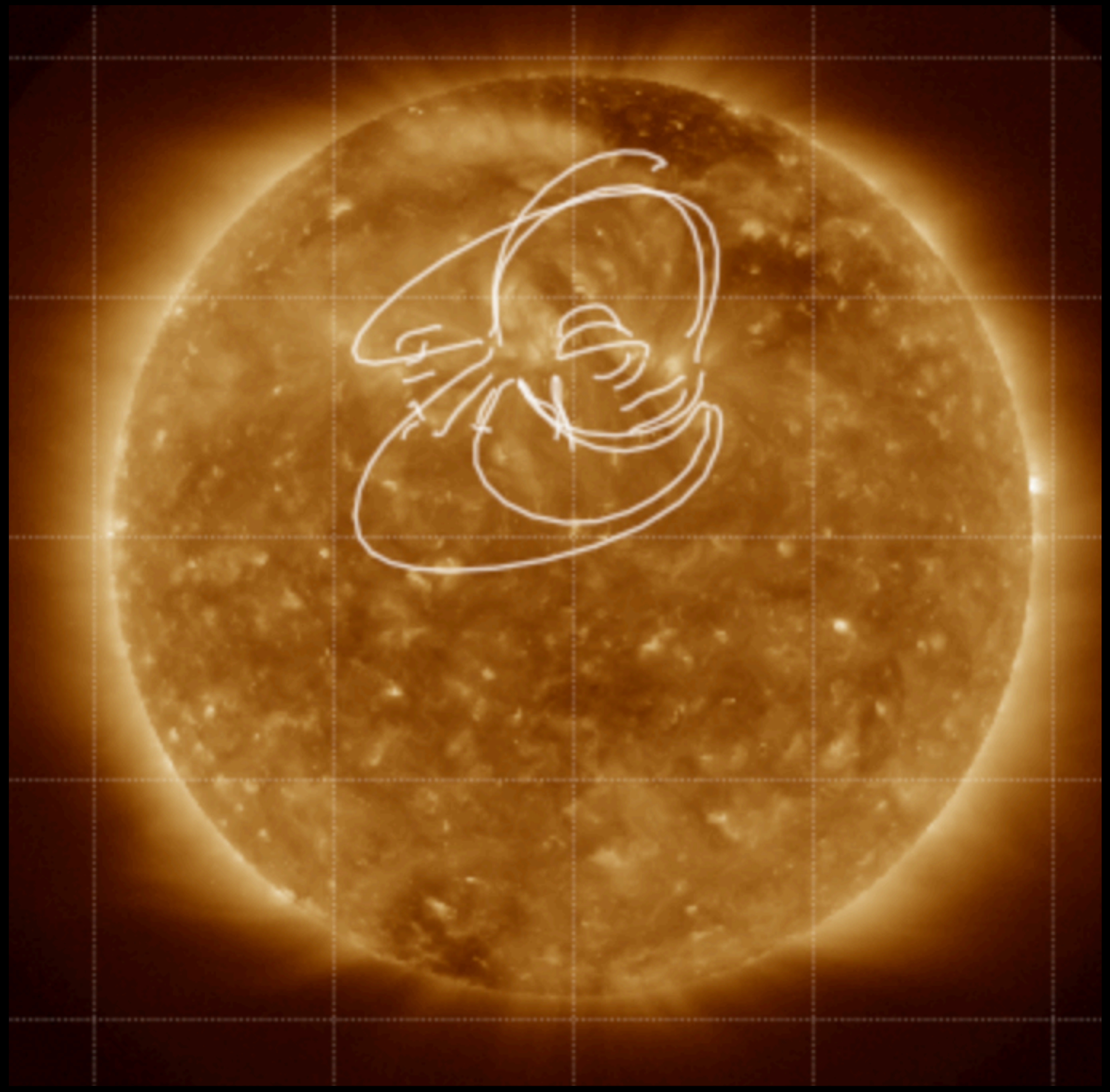
- Used in 18 papers (and counting!)
- Critical for interpreting results from Parker Solar Probe, NASA's $2bn mission to "Touch the Sun"
- It took a lot of unfunded work to get here; I was lucky to have time and flexibility
- I think unique position as both scientist **and** software engineer helped make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