Software Citation Tools, Technologies, and Best Practices

Matthew Feickert
(University of Wisconsin-Madison)
matthew.feickert@cern.ch

Software Citation and Recognition in HEP Workshop 2022
November 23rd, 2022

American Family Insurance
Data Science Institute
UNIVERSITY OF WISCONSIN–MADISON

iris hep

ATLAS EXPERIMENT

Introduction and Overview

- In Tuesday's session, Daniel Katz already gave very nice high level overview of software citation **principles** and **tools**
- This is an opinionated summary of the tooling landscape and examples of workflows
  - Full disclosure: Opinions formed from pyhf development and from Scikit-HEP community discussions (c.f. Eduardo's talk).
- Meant to be recommendations to software developers on making your work as **easy to cite as possible**
  - These recommendations can transfer to experiment software as well

Image courtesy of DataCite

Daniel Katz's talk

Make clear how to cite in documentation

- The easiest, but least robust way: If you have a particular citation that you want people to use, put it **everywhere**
  - Version control repository README
  - Online software documentation (landing page, how to cite page)
  - Package distribution websites (e.g. PyPI)
- Having single source of truth for citations: version control repository that all other sources derive from.
- Make your citation preferences clear to the world and SEO. Do not rely on people emailing to ask (they shouldn't have to).

pyhf's "Use and Citations" page in documentation

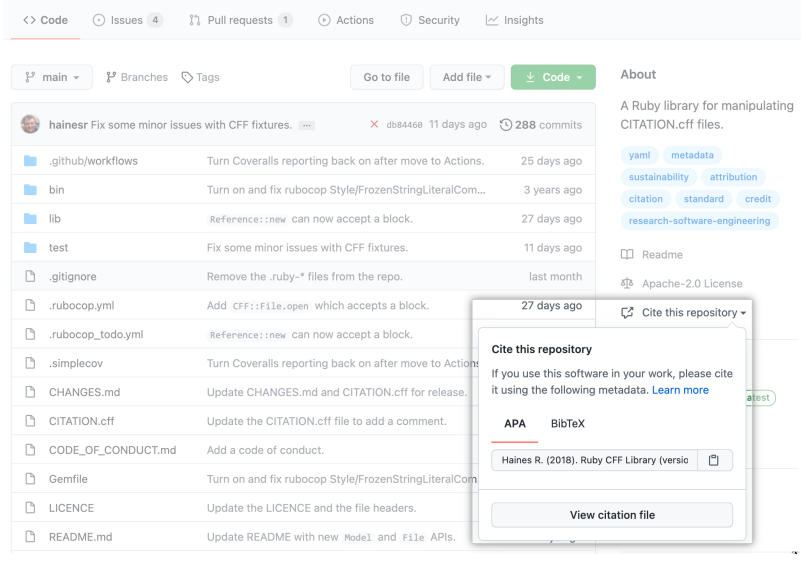
CITATION.cff

- Adopt the Citation File Format as a common standard and add a `CITATION.cff` to project repository
  - Human- and machine-readable file format in YAML
  - Has well defined, versioned schema
  - Convertible to other citation formats (BibTeX, CodeMeta, EndNote, RIS, schema.org, Zenodo, APA)
- Supported by GitHub, Zenodo, and Zotero!
- Web tool initializer for easily creating first `CITATION.cff`
- Tooling for validation

```
$ python -m pip install cffconvert
$ cffconvert --validate
Citation metadata are valid according to schema version 1.2.0.
```

```
cff-version: 1.2.0
message: "If you use this software, please cite it as below."
authors:
  - family-names: Druskat
    given-names: Stephan
    orcid: https://orcid.org/0000-0003-4925-7248
title: "My Research Software"
version: 2.0.4
doi: 10.5281/zenodo.1234
date-released: 2021-08-11
```

Example of minimal `CITATION.cff`

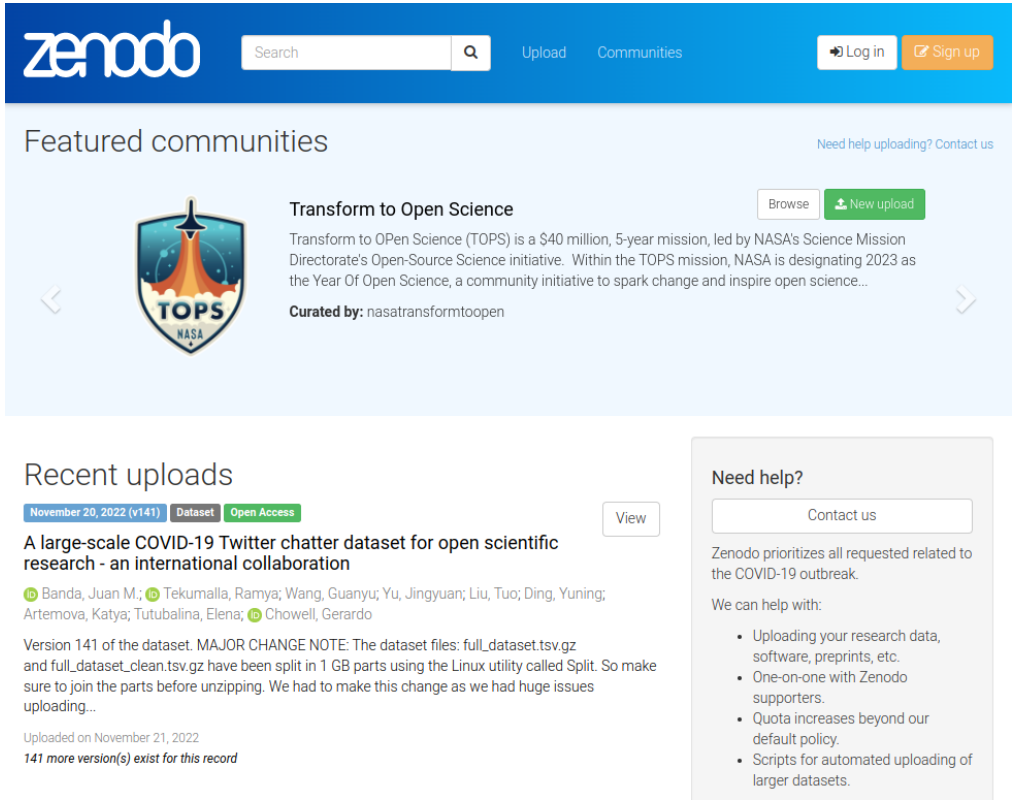


CITATION.cff: How to keep up to date?

- As plain text, very easy to update version information when cutting a release
- Can use tool control of version update to make it easier
  - Example: `tbump`
  - `$ tbump <version target>`
- Also possible to have automated version bump workflows using continuous integration
- (Jumping ahead a slide) What about the Zenodo DOI?
  - For simplicity, use the project level DOI and not the version level DOI

```
cff-version: 1.2.0
message: "Please cite the following works when using this software."
type: software
...
title: "mylibrary: v1.2.3"
version: 1.2.3
doi: 10.5281/zenodo.1123456
repository-code: "https://github.com/myorg/mylibrary/releases/tag/v1.2.3"
url: "https://mylibrary.readthedocs.io/en/v1.2.3/"
```

Zenodo



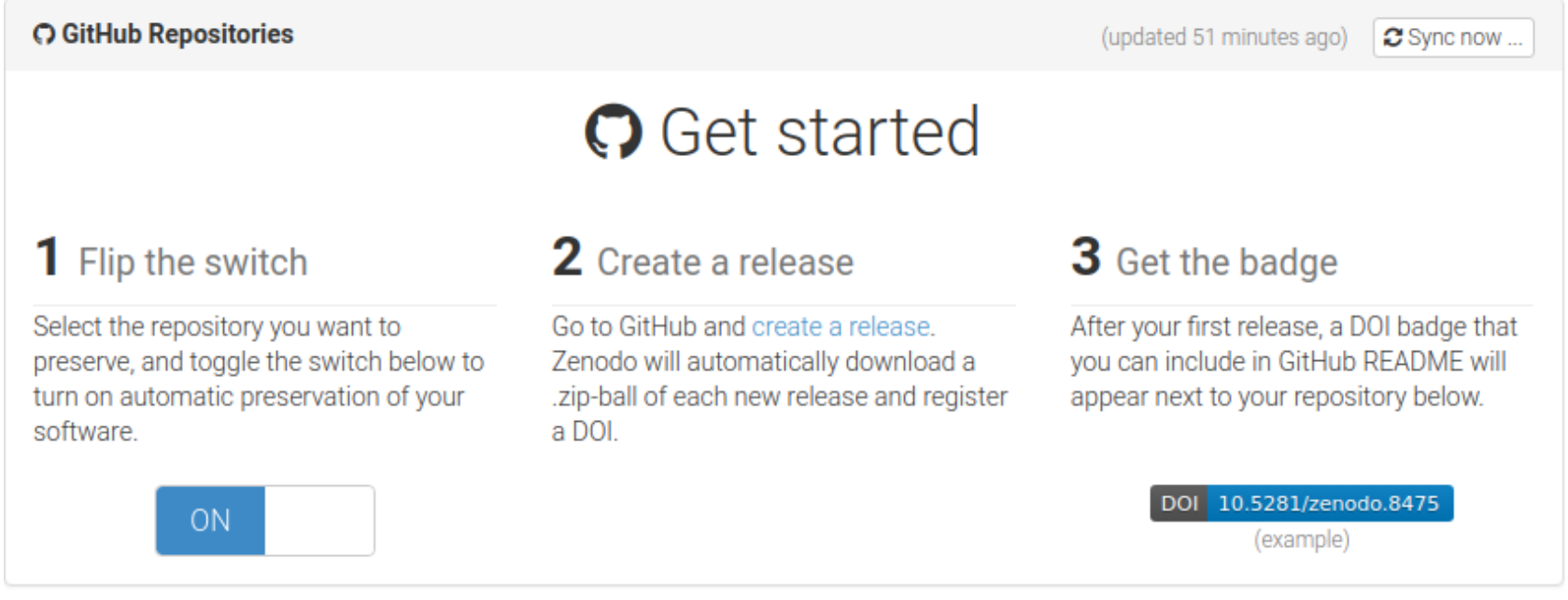
- Open source (but your files can be closed access)
- Versioned archival of everything: code, documents, data products, data sets

Why use Zenodo?

- **Safe** — your research is stored safely for the future in CERN's Data Centre for as long as CERN exists.
- **Trusted** — built and operated by CERN and OpenAIRE to ensure that everyone can join in Open Science.
- **Citeable** — every upload is assigned a Digital Object Identifier (DOI), to make them citable and trackable.
- **No waiting time** — Uploads are made available online as soon as you hit publish, and your DOI is registered within seconds.
- **Open or closed** — Share e.g. anonymized clinical trial data with only medical professionals via our restricted access mode.
- **Versioning** — Easily update your dataset with our versioning feature.
- **GitHub integration** — Easily preserve your GitHub repository in Zenodo.
- **Usage statisics** — All uploads display standards compliant usage statistics

Zenodo: DOI minting made easy

- Everything on Zenodo has a DOI
  - Provides both a **project** DOI (resolves to latest) and **version specific** DOI
- Enable it to automatically preserve work from GitHub (can also directly upload, but lose out on automation)
  - Benefit from having a DOI for **every version** regardless of software paper landscape state
- Once you have a DOI, put it **everywhere** (again)
  - Recommend sharing the project DOI and letting users select a specific version if they want it

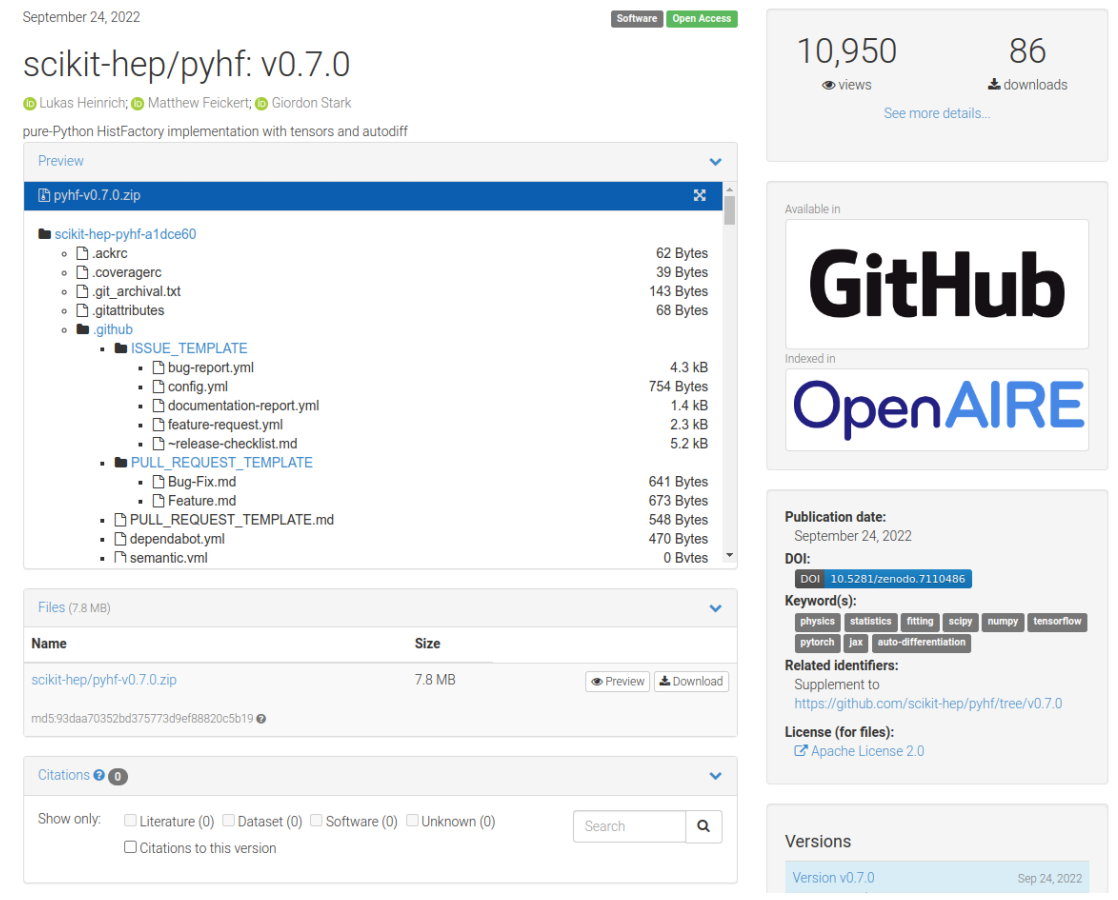


Zenodo + CITATION.cff

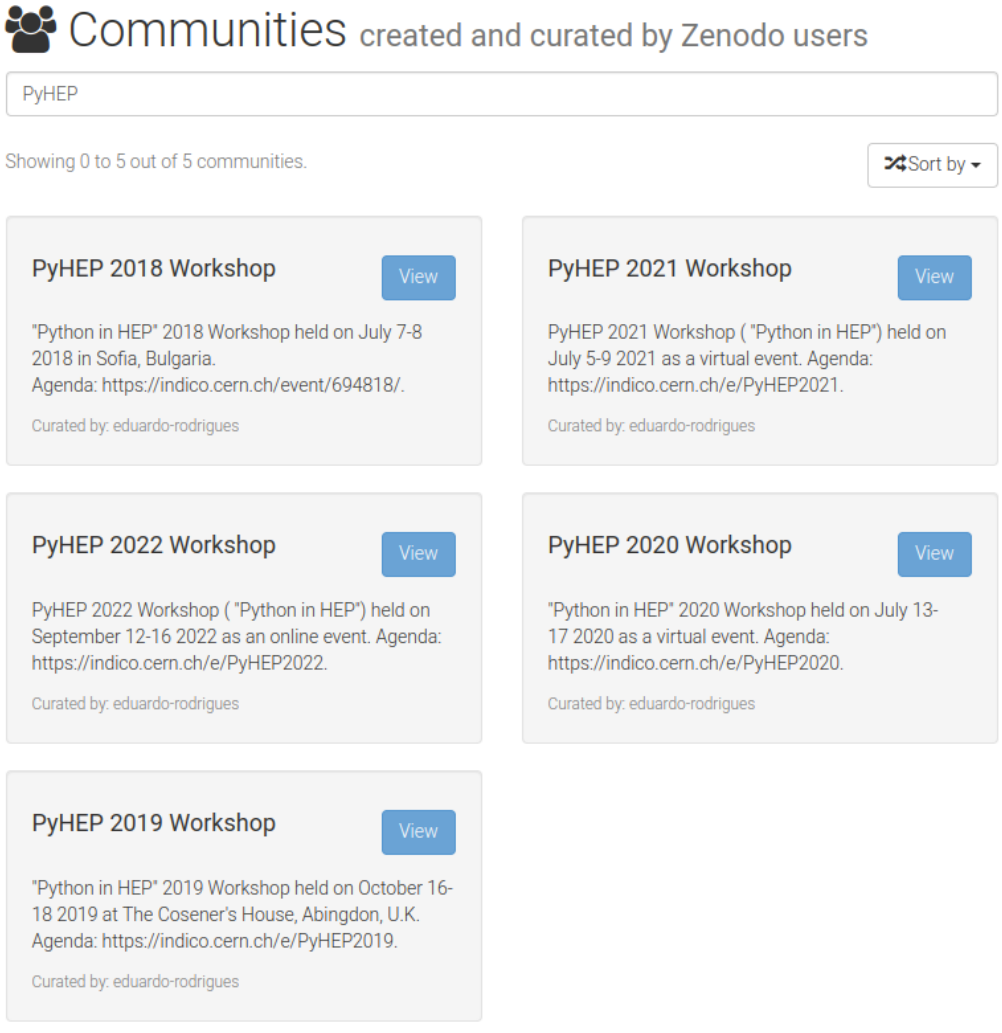
CITATION.cff used by Zenodo importer to fully define Zenodo archive metadata

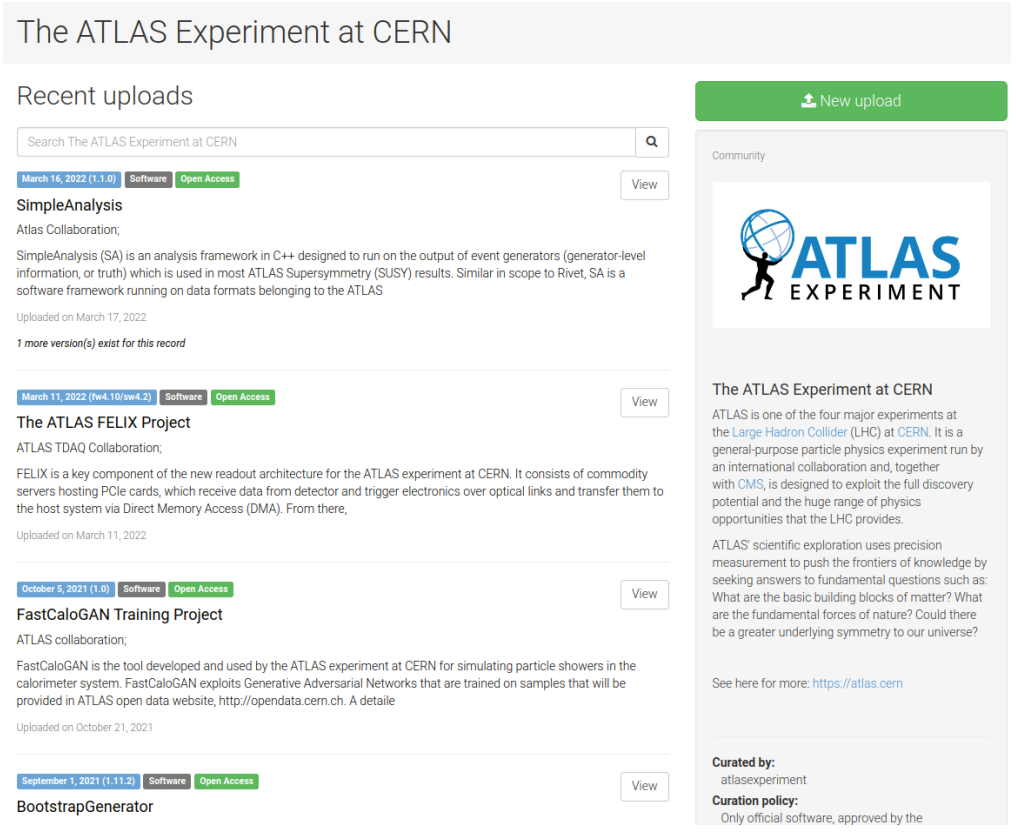
71 lines (71 loc) · 2.47 KB

```yaml
cff-version: 1.2.0
message: "Please cite the following works when using this software."
type: software
authors:
- family-names: "Heinrich"
  given-names: "Lukas"
  orcid: "https://orcid.org/0000-0002-4048-7584"
  affiliation: "Technical University of Munich"
- family-names: "Feickert"
  given-names: "Matthew"
  orcid: "https://orcid.org/0000-0003-4124-7862"
  affiliation: "University of Wisconsin-Madison"
- family-names: "Stark"
  given-names: "Giordon"
  orcid: "https://orcid.org/0000-0001-6616-3433"
  affiliation: "SCIPP, University of California, Santa Cruz"
title: "pyhf: v0.7.0"
version: 0.7.0
doi: 10.5281/zenodo.1169739
repository-code: "https://github.com/scikit-hep/pyhf/releases/tag/v0.7.0"
url: "https://pyhf.readthedocs.io/en/v0.7.0/"
keywords:
  - python
  - physics
  - statistics
  - fitting
  - scipy
  - numpy
  - tensorflow
  - pytorch
  - jax
  - auto-differentiation
license: "Apache-2.0"
```



Zenodo: Communities allow archival collections





Proving a citation information from APIs

- In addition to providing standard formats, providing users a language API or CLI API to get the citation information for the version of the tool is helpful
  - User doesn't have to check if the information they find online matches their version.
- Historically, this was done by printing a banner with citation or copyright information when the library is used
  - This should **not** be done now. This creates noise for users and if multiple tools did this your terminal would get filled.
  - Most libraries that used to do this have now abandoned this approach.
- Opinion: There are tools in broader scientific ecosystem that provide citation information for their dependencies as well. While very conscientious, I think this is **unnecessary** and can be confusing to users.

```
# CLI API
$ mytool --citation
$ mytool --cite


# Python API
import mytool
mytool.utils.citation()
```

Example APIs

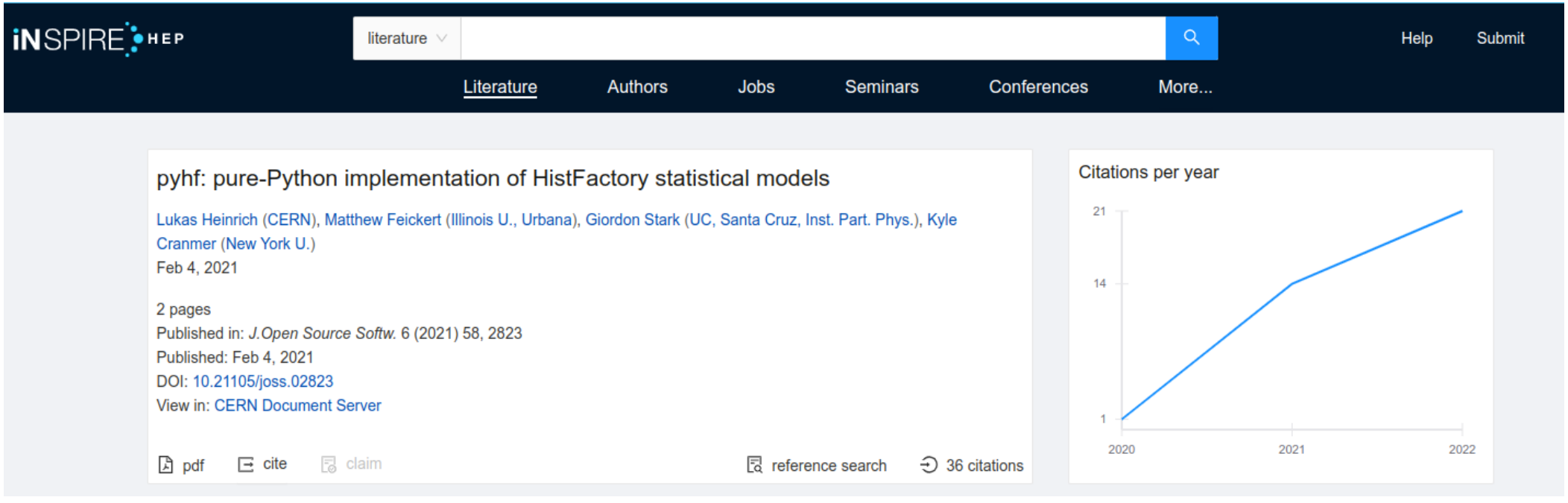
Summary

- Build community practices on top of **established standards**
- If citation of your software is important to you, **make it easy** for a user to find your citation information
- Modern standards like `CITATION.cff` allow for **single source of citation information** that can be exported as needed
- Long term archives + FAIR practices
  - Zenodo provides automatically release information each release

Backup

Does any of this actually work?

As mentioned, these opinions have been formed from developing pyhf, and the citation count for the JOSS paper has increased each year.



The end.