CSSI Framework: An open source software ecosystem for plasma physics

N. A. Murphy[1] (PI), D. Stańczak,[2] (Collaborator), E. T. Everson[3] (Project Personnel), S. T. Vincena[3] (Co-I), D. Schaffner[4] (Co-I), T. N. Parashar[5,6] (Co-I emeritus), B. Maruca[5] (Co-I elect), A. J. Leonard[7] (Collaborator), & T. Carter[3] (Collaborator) on behalf of the PlasmaPy Community

[1]Center for Astrophysics | Harvard & Smithsonian, [2]University of Warsaw, [3]UCLA, [4]Bryn Mawr College, [5]University of Delaware, [6]University of Wellington, [7]Aperio Software

2020 NSF CSSI Principal Investigator Meeting

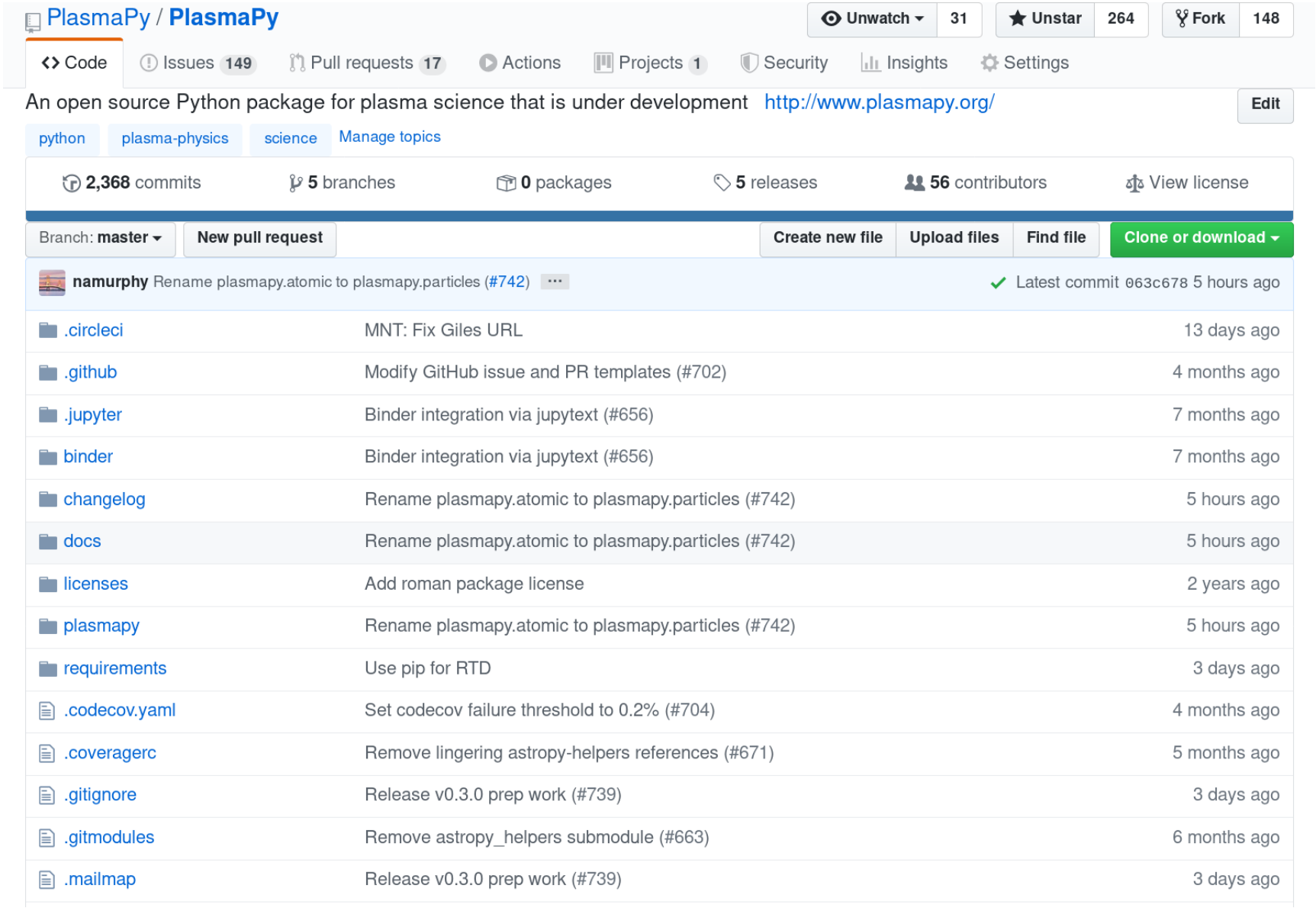
plasmapy

Introduction

- In recent years, researchers in several different subfields of physics and astronomy have collaboratively developed core Python packages such as Astropy[1] and SunPy[2]
- These packages provide core functionality and common frameworks for data analysis and visualization
- A similar open source package would greatly benefit plasma science
- **We are developing PlasmaPy: a community-developed and community-driven open source core Python package for plasma physics**

[1] Astropy Collaboration (2018)
[2] SunPy Community (2015)

The goal of PlasmaPy is to facilitate a fully open source software ecosystem for plasma physics



PlasmaPy / **PlasmaPy**

Unwatch 31 | Unstar 264 | Fork 148

Code | Issues 149 | Pull requests 17 | Actions | Projects 1 | Security | Insights | Settings

An open source Python package for plasma science that is under development http://www.plasmapy.org/ Edit

python | plasma-physics | science | Manage topics

2,368 commits | **5** branches | **0** packages | **5** releases | **56** contributors | View license

Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

namurphy Rename plasmapy.atomic to plasmapy.particles (#742) ··· Latest commit 063c678 5 hours ago

.circleci	MNT: Fix Giles URL	13 days ago
.github	Modify GitHub issue and PR templates (#702)	4 months ago
.jupyter	Binder integration via jupytext (#656)	7 months ago
binder	Binder integration via jupytext (#656)	7 months ago
changelog	Rename plasmapy.atomic to plasmapy.particles (#742)	5 hours ago
docs	Rename plasmapy.atomic to plasmapy.particles (#742)	5 hours ago
licenses	Add roman package license	2 years ago
plasmapy	Rename plasmapy.atomic to plasmapy.particles (#742)	5 hours ago
requirements	Use pip for RTD	3 days ago
.codecov.yaml	Set codecov failure threshold to 0.2% (#704)	4 months ago
.coveragerc	Remove lingering astropy-helpers references (#671)	5 months ago
.gitignore	Release v0.3.0 prep work (#739)	3 days ago
.gitmodules	Remove astropy_helpers submodule (#663)	6 months ago
.mailmap	Release v0.3.0 prep work (#739)	3 days ago

Plasma physics has a "roll your own" culture for code development

- Scientists tend to be self-taught as programmers
- Time pressure prevents us from improving programming skills
- Software is often written "in-house" as needed
- Code is often written in a rush to get a paper out
- Code is often written for a specific purpose, which makes it hard to generalize
- Documentation is often insufficient
- Codes often lack a testing framework
- Frequent duplication of functionality between groups
- Packages lack interoperability

Consequences of “roll your own” culture

- **Beginning research is difficult** due to software overhead
- **Collaboration is difficult** due to lack of interoperability
- Plasma research is much **less reproducible**
- Research can be **frustrating**

Plasma science can learn from what other fields are doing to change “roll your own” culture.

PlasmaPy uses Astropy's open development model

- Release code under an **open source license**
- **Develop openly** on GitHub
- **Anyone may contribute**
- New contributors are **actively welcomed**
- Adopt a **code of conduct**

Planned subpackages for PlasmaPy version 0.4.0

- `particles` allows access to basic atomic and particle data
- `formulary` contains plasma parameters, transport coefficients, and mathematical functions
- `plasma` will contain base classes for plasma configurations
- `simulation` contains a particle pusher and will contain broader simulation capabilities
- `diagnostics` will provide tools to access experimental data
- `analysis` will contain tools to analyze simulations and experimental data
- `utils` provides utilities used throughout the package
- `tests` will include test helper functionality
- `addons` will allow user extensions to PlasmaPy

PlasmaPy uses the `astropy.units` package for units

This package creates `Quantity` objects with attached units.

```
>>> from astropy import units

>>> distance = 44 * units.imperial.mile
>>> time = 30 * units.minute
>>> distance / time
<Quantity 88.0 mi / h>

>>> (1.21 * units.GW).cgs
<Quantity 1.21e+16 erg / s>

>>> 2 * units.m + 4 * units.m / units.s
UnitConversionError: Can only apply 'add' function
to quantities with compatible dimensions
```

PlasmaPy's particles subpackage provides functional and object-oriented interfaces to particle data

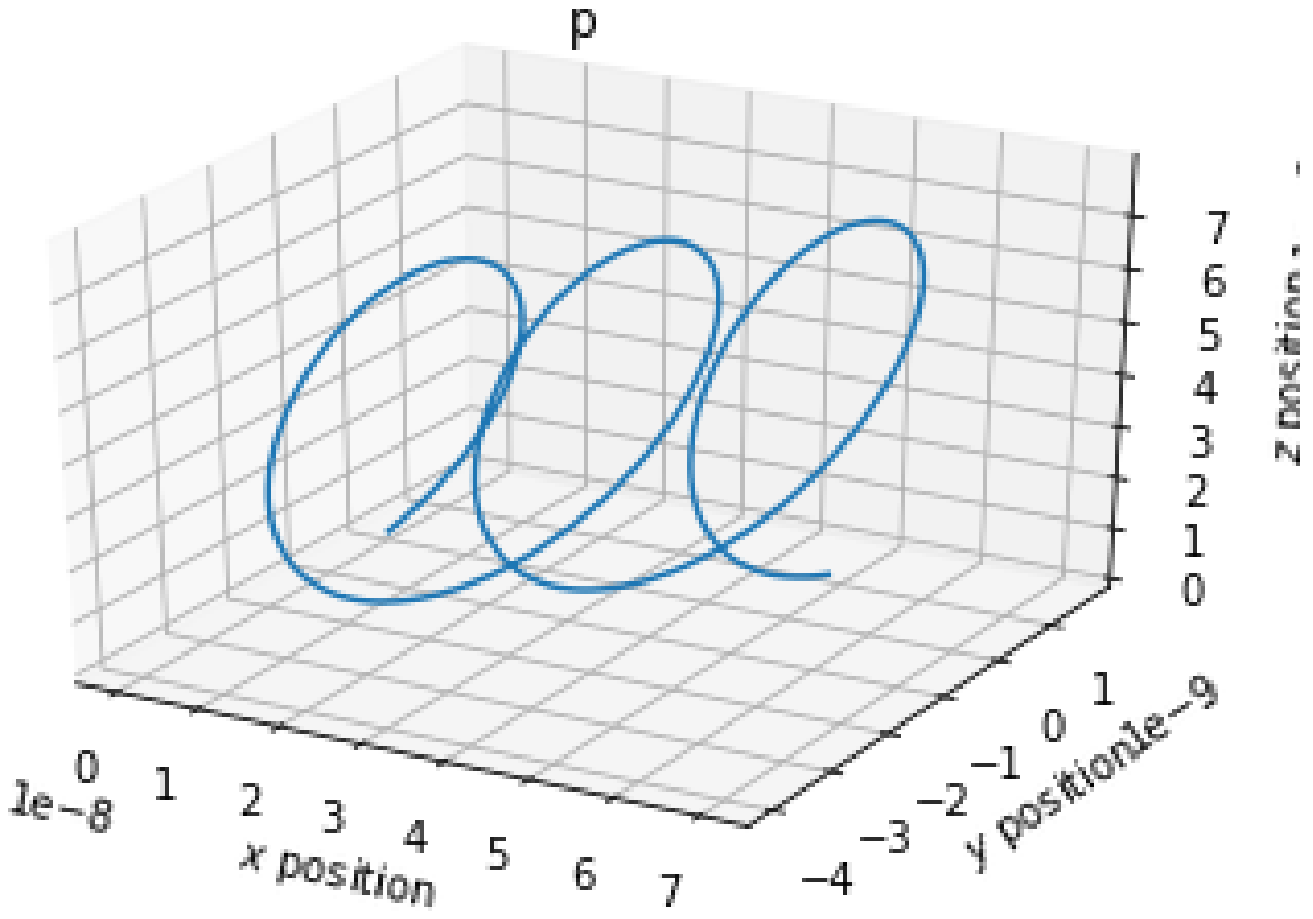
Instances of the Particle class may be used to represent individual atoms, ions, or elementary particles.

```
>>> from plasmapy.atomic import Particle

>>> alpha = Particle("He-4++")
>>> alpha.mass
<Quantity 6.64465709e-27 kg>

>>> electron = Particle("e-")
>>> electron.charge
<Quantity -1.60217662e-19 C>
>>> electron.is_category(require={"lepton", "fermion"})
True
>>> ~electron  # find antiparticle with invert operator
Particle("e+")
```

plasmapy.simulation contains a particle pusher[3]



[3]More examples: http://docs.plasmapy.org/en/latest/auto_examples

Code development priorities for 2020

- **Refactor existing code and tests**
  - Strengthen foundation for future development
- **Interfaces for plasma simulations**
  - Classes to represent problem setup independent of numerics
  - Metadata schemas (akin to openPMD)
  - Interchangeable simulation modules
- **Plasma diagnostic analysis tools**
  - Create diagnostic classes to be instantiated for specific probes
  - Provide tools for the broader plasma community to develop analysis packages for different experiments
- **Educational Jupyter notebooks**
  - Use PlasmaPy to introduce fundamental plasma concepts

Organizational infrastructure is needed for long-term software sustainability

- The **Coordinating Committee** oversees the PlasmaPy project and code base
  - Ideally have representation across plasma subdisciplines
  - Currently strong representation by heliophysicists
- In practice, **most coordination is done informally**
  - GitHub issues and pull requests
  - Matrix/Gitter channel for text-based chat
  - Weekly community meetings
- **PlasmaPy Enhancement Proposals (PLEPs)** allow the community to influence the direction of PlasmaPy
- The **PlasmaPy Community on Zenodo**[4] contains presentations, PLEPs, white papers, and proposals

[4] https://zenodo.org/communities/plasmapy

PlasmaPy has documentation![5]

- Each function and class has a docstring
- Subpackages have narrative documentation
- Docstrings and narrative documentation are transformed into online documentation
- Test builds of documentation are run for every pull request
- Code examples are tested to make sure output is correct

[5] https://docs.plasmapy.org

PlasmaPy has tests!

- All pull requests undergo **continuous integration testing**
  - We know right away when we break something
  - Useful error messages help narrow down causes
- Automated **test coverage checks** show which lines of code are not covered by tests
  - We know what tests we still need to write
  - We can find and delete unused portions of code
- Helpful practices
  - Write tests before production code
  - Turn bugs into tests cases

Anticipated benefits of PlasmaPy

- More reproducible, open, and efficient research
- Reduce duplication of functionality
- Reduce barriers to entry for plasma research
- Let students hit the ground running on first research projects
- Help us learn collaborative code development practices
  - Helpful for students entering industry upon graduation
- Provide well-documented and well-tested software
- Improve interoperability between different packages
- Enable cross-disciplinary and cross-device studies
- Reduce software development overhead costs for experiments
- Create tools for plasma pedagogy

Summary

- We are developing PlasmaPy to contain core functionality for an **open source software ecosystem** for plasma science
  - Version 0.4.0 is expected to be released this summer
- PlasmaPy is building bridges among laboratory, heliospheric, and astrophysical plasma physicists
  - Active in Python in Heliophysics Community
  - Participating in community planning process for fusion energy sciences
- If there is functionality that you would like in PlasmaPy, please raise an issue in our GitHub repository!