BUILDING COMMUNITIES AROUND OPEN-SOURCE GEOSCIENCE PROJECTS

or how to win contributors and influence a dicipline

JOHN LEEMAN, RYAN MAY

unidata

CREATING A COMMUNITY

Training

Workshops - We travel and train our users, as well as provide online self-paced training material.

Example Gallery - Providing examples is a great way to get users engaged. Our examples are relevant real-life problems many meteorologists face.

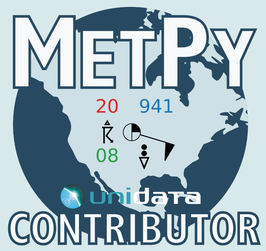
MetPy Mondays - Every Monday we release a short (~10 minute) video describing how to do something with Python/MetPy.

Scope

One of the most difficult things in maintaining a domain specific library is defining and maintaining the scope of the project. MetPy is defined as tools useful specifically to a meteorological audience. Contributions that are more general are declined and directed towards the more appropriate general package (CartoPy, matplotlib, numpy, etc.).

Contributors

- Fast and friendly code review
- Curate “Good First Issues”
- “Zen and the Art” Guide
- Contributor Stickers
- Contributor’s Guide
- Code of Conduct



Meteorological Software Timeline

Late 80’s
GEMPAK Developed

Mid-Late 90’s
NCL (NCAR Command Language) begins development

2008
MetPy begins with a base of graduate student code

2014
GEMPAK goes to “no new development”

2015
Unidata begins support of Python projects

2017
MetPy receives NSF SSI grant

Second full time developer hired at Unidata

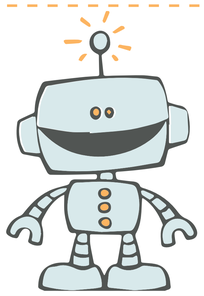
CORNERSTONES

Testing

With scientific calculations, it is crucial that we know that the library of calculations and plotting utilities remains reliable, consistent, and correct.

Documentation

Our documentation explains how calculations are carried out and references literature for both the calculation and verification cases.



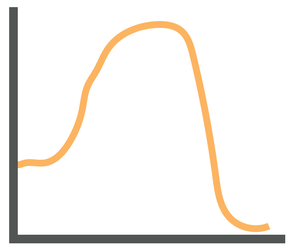
Automation

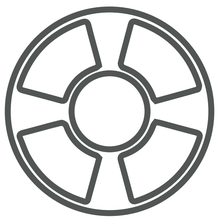
We run as many things (testing, flake8, examples, docs, etc.) as possible in an automated way. It helps the project scale with increased contributions.

CHALLENGES

Activation Energy

The geosciences have a history of developing open software. It often results in a cargo-cult like culture of use without understanding due to cumbersome API design.





Legacy Software

There are masses of scripts using legacy tools. The resistance to change is strong, but breaking down as legacy software is increasingly difficult to utilize.

Contribution Hurdle

Making the leap from user to contributor is still too high and there is git, the code base, style guides, how to view build output, run tests, and more to learn.

Unidata and this work are supported through grants NSF-1344155 and OAC-1740315. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.