MetPy 1.1.0 Milestones: Code Fixes and Verification

Lydia Bunting
Ryan May, Drew Camron, Unidata, UCAR

What is MetPy?

A collection of tools in Python for reading, visualizing, and performing calculations with weather data.

Development is supported by the National Science Foundation.

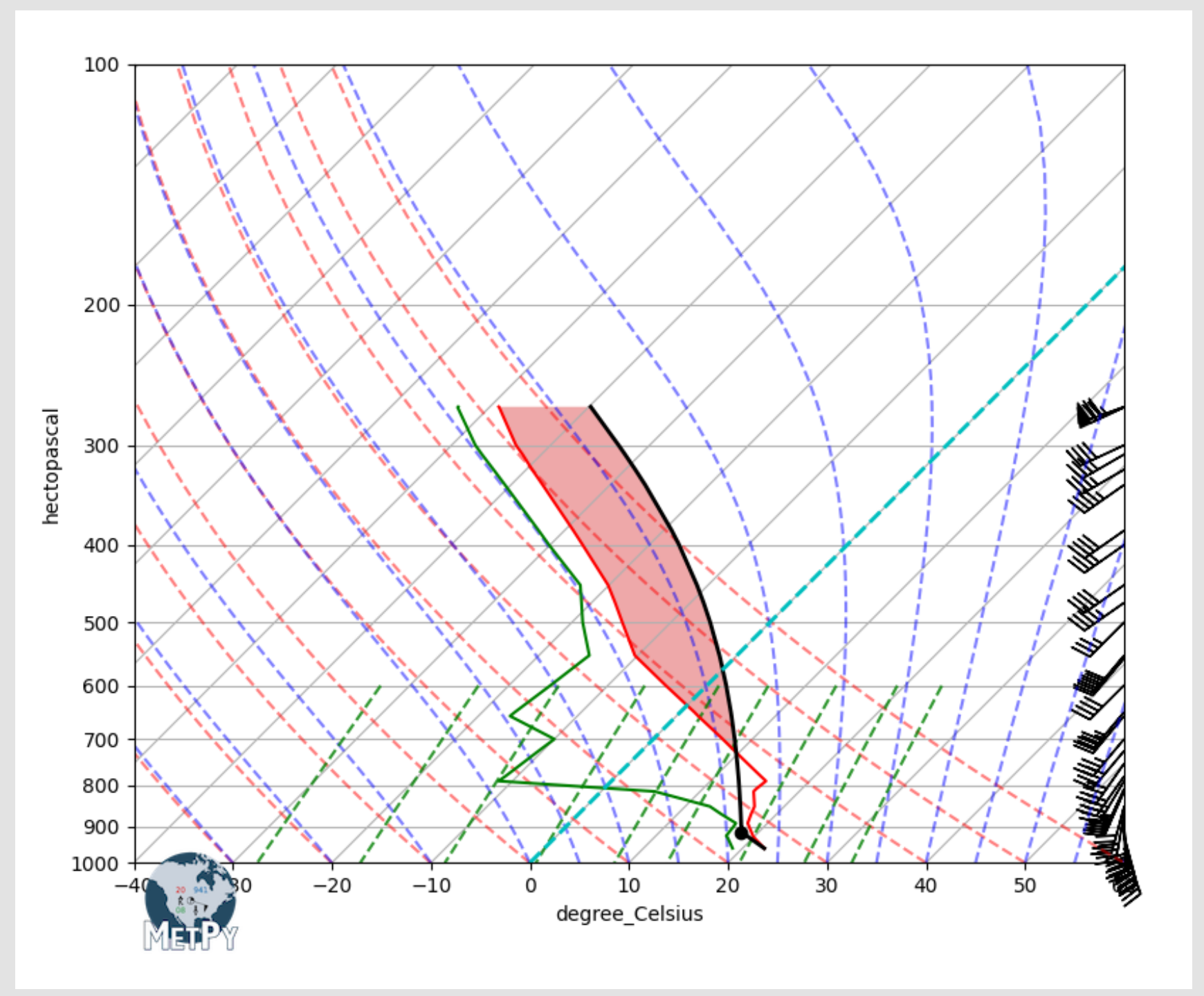
Primary Uses:

Meteorological research, including performing calculations, reading data, and plotting.

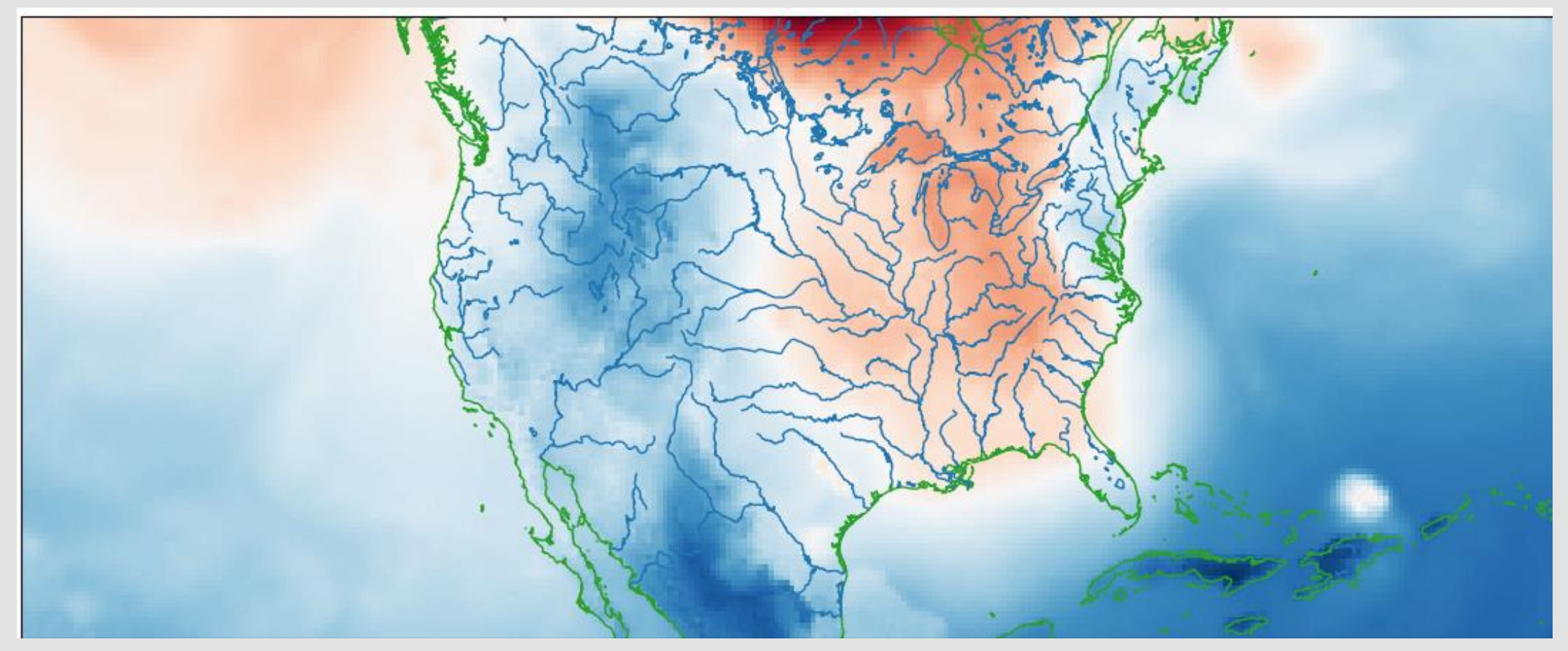
What can MetPy be used for?

Some examples:

Plotting sounding data and performing calculations:



Plotting data on a map using XArray and CartoPy:



MetPy 1.1.0 Milestones

Code enhancements or bug fixes to be addressed for the 1.1.0 update.

Presented as "issues" in GitHub to be addressed before the update is implemented.

Issue 1844

Initial problem:

pyproj CF (climate and forecasting) output not accepted by *metpy.assign_crs().*

➢ The function *Metpy.assign_crs()* assigns a coordinate reference system to the MetPy data array based on CF projection attributes.

Initial fix:
Adding **earth_radius** to the input directory.

New problem:
Latitude of projection center missing in CF listing.

Cause:
Conversion from PyProj to CF results in a value 0 for the attribute *inverse_flattening*.

New fix:
Interpret the 0 inverse_flattening as a spherical datum and do not pass the value on.

Code Verification

Before fixes are merged with MetPy, we need to verify it works as expected.

This is done through **unit testing**.

Unit Testing
A piece of code that "activates" a piece of a system to ensure it behaves as expected by developers.

Starts with the smallest components first:

➢ Ensures they work properly before integrating them with larger portions of code.

Goal

- Isolate each part of the program and show it is correct.

Importance

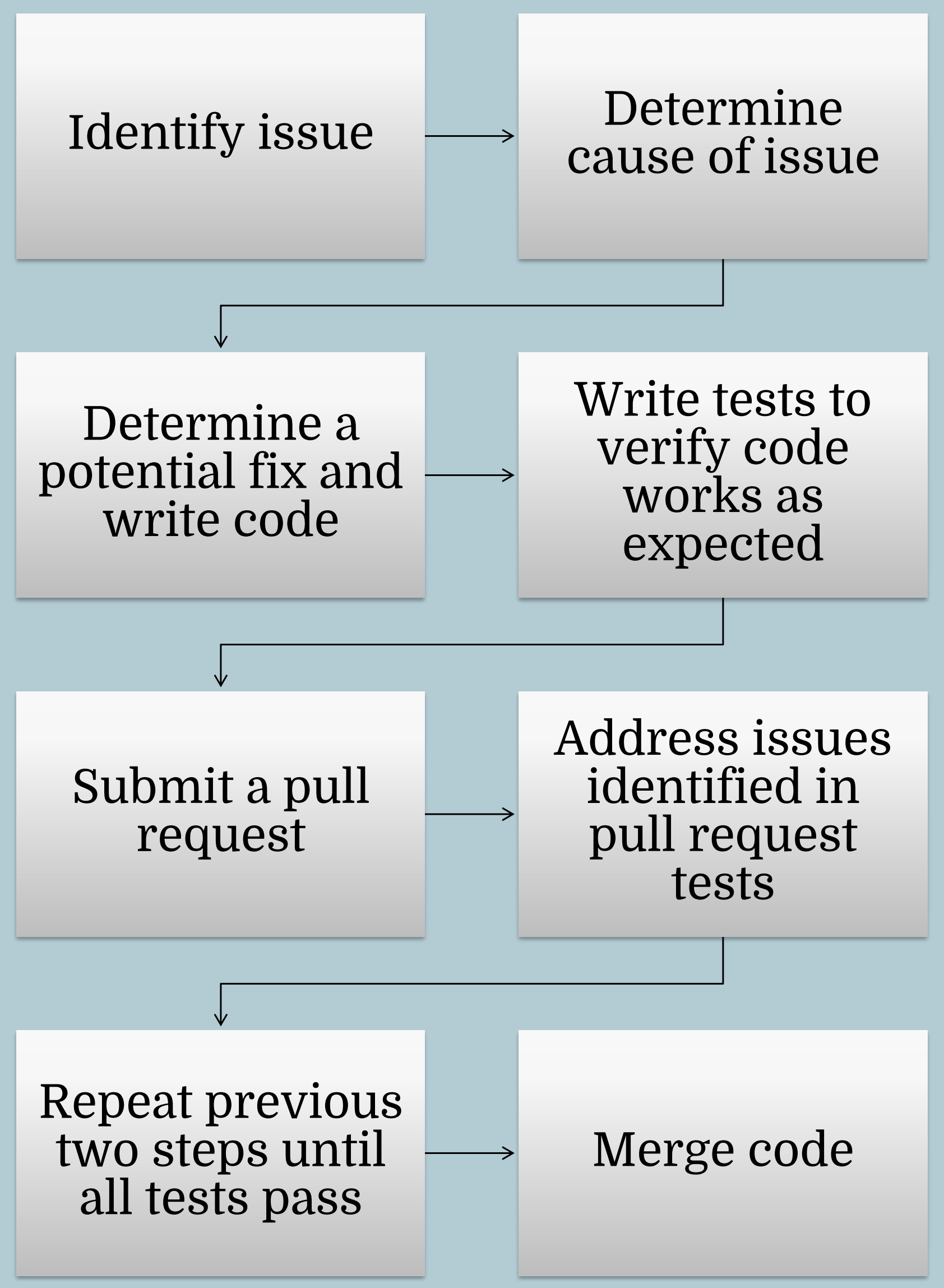
- Finds problems early as code is developed.
- Forces developers to think through code thoroughly.
- Neglecting tests can lead to broken code and problems for users.

Test for Issue 1844

Introduce the case where inverse_flattening = 0 to "activate" new code where this is the case.

➢ Want to make sure the value is not being passed onto the rest of the program.

Complete Process


Identify issue
Determine cause of issue
Determine a potential fix and write code
Write tests to verify code works as expected
Submit a pull request
Address issues identified in pull request tests
Repeat previous two steps until all tests pass
Merge code

Summary

➢ MetPy 1.1.0 Milestones are bugs or additions to be completed before its implementation.
➢ Performed by identifying what is causing bugs or how to add new functions.
➢ New code is written and submitted for review via pull request.
➢ Code functionality is ensured through unit tests that check if all code works as expected.
➢ When review is completed, changes are merged with existing MetPy code.

Acknowledgements

Thank you for Unidata and UCAR for this incredible opportunity! Thank you also to Ryan May, Drew Camron, and the rest of the tech team, for all their help and support this summer.