MetPy 1.1.0 Milestones: Code Fixes and Verification

LYDIA BUNTING

SUMMER 2021
Acknowledgements

• Unidata and UCAR
• Ryan May and Drew Camron
• Connor Cozad and Izzy Pfander
What it is:

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.
MetPy Usage Examples

1. Plotting time series data as a meteogram (right)
2. Plotting data on a map using XArray and CartoPy (below)
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.
  ➢ The value of lat_0 is lost.

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 that value on.
Addressing error

- To address the issue, added an ‘if’ statement to address the case where inverse_flattening = 0

```python
# interpret the 0 inverse_flattening as a spherical datum
# and don’t pass the value on.
if kwargs.get('inverse_flattening', None) == 0:
    kwargs['ellipse'] = 'sphere'
    kwargs.pop('inverse_flattening', None)
```
Code Verification

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

• This is done through unit testing.

• Starts with the smallest components first:
  • Ensures they work properly before integrating them with larger portions of code.

**DEFINITION: Unit Testing**

A piece of code that “activates” a piece of a system to ensure it behaves as expected by developers.
Code Verification

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

• Need to test new code by writing a test to “activate” it.

• For Issue 1844, introduce the case where inverse_flattening = 0.

```python
def test_inverse_flattening_0():
    """Test new code for dealing the case where inverse_flattening = 0."""
    attrs = {'grid_mapping_name': 'lambert_conformal_conic', 'semi_major_axis': 6367000,
             'semi_minor_axis': 6367000, 'inverse_flattening': 0}
    proj = CFProjection(attrs)

    crs = proj.to_cartopy()
    globe_params = crs.globe.to_proj4_params()

    assert globe_params['ellps'] == 'sphere'
    assert globe_params['a'] == 6367000
    assert globe_params['b'] == 6367000
```
Pull request process

1. Submit pull request
   • Submits the changed code for testing and review

2. Automated tests
   • Identify code that may have been missed by manual testing process.
   • Check for drops in code coverage and style variations.

3. Code review by Unidata staff

4. Merging
   • Performed once all tests and details of the pull request are addressed.
Complete Process

1. Identify issue
2. Determine cause of issue
3. Determine a potential fix and write code
4. Write tests to verify code works as expected
5. Submit a pull request
6. Address issues identified in pull request tests
7. Repeat previous two steps until all tests pass
8. Merge code
Summary

• Code verification is an essential component to code development.
  ➢ Unit testing is the primary way this is achieved.

• Failing to perform code verification can lead to broken code and lack of functionality.

• MetPy is a program used for a variety of applications and by a variety of users:
  ➢ This makes adequate testing even more important.
  ➢ Broken code can have a lasting impact on research & user experience.
Thank you!