Enabling Declarative Syntax while using Matplotlib’s pcolormesh in MetPy

By: Nathaniel Martinez
About Me!

• Rising 4th year at UChicago
• Computer Science & Environmental Science
• Interning since the end of May
  – Software development process
  – Adding declarative syntax for pcolormesh()
  – Documentation and example improvements
Software Development Process

• Planning
  • Identify the issue to be fixed or new functionality to be added
  • Identify stakeholders’ software needs

• Design
  • Compile a design resolving the issues identified while planning
Software Development Process

• Implementation
  • Draw from the design to write the code to fulfill the identified criteria

• Testing
  • Create sufficient tests to ensure all new or edited code is verified to work properly
Software Development Process

• Integration
  • Create a pull request for the new code into the code repository, resolving any conflicts
Declarative Syntax

• Simplifies plotting process
  • No need to call Matplotlib functions directly
  • Add support for pcolormesh

```python
data = xr.open_dataset(get_test_data('narr_example.nc', as_file_obj=False))

raster = RasterPlot()
raster.data = data
raster.field = 'Temperature'
raster.level = 700 * units.hPa

panel = MapPanel()
panel.area = 'us'
panel.projection = 'lcc'
panel.layers = [{'coastline'}]
panel.plots = [raster]

pc = PanelContainer()
pc.size = (8, 8)
pc.panels = {panel}
pc.draw()
```
**Raster Plots**

- Plots a grid of values
- Helpful in plotting key meteorological data
- Temperature, wind speed, humidity, etc.
- Potential for radar reflectivity plots

```python
# Full data out of the file
swepn = a

# First item in ray is header, which has azimuth angle
az = np.array(ray[0][az_angle for ray in swepn])
diff = np.diff(az)
diff[diff > 360] = 0
 diff[diff < -360] = 360
 avg_spacing = diff.mean()
 az = (az[-1] + az[0]) / 2
 az = np.concatenate((az[0] - avg_spacing, az, (az[-1] + az[0])))

# 5th item is a dict mapping a var name (byte string) to a tuple
# of (header, data array)
ref_hrd = f.swepn[swepn[0][4][b'REF']][0]
ref_range = np.arange(ref_hrd['num-gates'] + 1) - 0.5
 ref_hrd['gate_width'] + ref_hrd['first-gate]
 ref = np.array(ref[4][b'REF'][1] for ray in f.swepn)

rho_hrd = f.swepn[swepn[0][4][b'HRD']][0]
rho_range = np.arange(rho_hrd['num-gates'] + 1) - 0.5
 rho_hrd['gate_width'] + rho_hrd['first-gate']
rho = np.array(r[f][b'HRD'][1] for ray in f.swepn)

fig, axes = plt.subplots(2, 2, figsize=(8, 8))
add_entry_image(fig, 100, 85, size='large')
for var_data, var_range, ax in zip((ref, rho), (ref_range, rho_range), axes):
    # Turn into an array, then mask
    data = np.ma.array(var_data)
data[~np.isfinite(data)] = np.ma.masked

data[ref, ax]
xlocs = var_range + np.cos(np.deg2rad(az))
ylocs = var_range + np.cos(np.deg2rad(az))

# Plot the data
ax.plot(np.log10(data), ylocs, data, cmap='viridis')
ax.set_xlim(-30, 30)
ax.set_ylim(-30, 30)
add_timestamp(ax, f'dt', y=0.82, high_contrast=True)

plt.show()
```
Documentation Improvements

• Updating Documentation
  • Updated examples to reflect added functionality
  • Resolved example issues and demonstrated best practices for users to follow

```python
# Grab azimuths and calculate a range based on number of gates
az = np.array(datadict['start_az'] + [datadict['end_az'][-1]])
rng = np.linspace(0, f.max_range, data.shape[-1] + 1)

# Grab azimuths and calculate a range based on number of gates,
# both with their respective units
az = units.Quantity(np.array(datadict['start_az'] + [datadict['end_az'][-1]]), 'degrees')
rng = units.Quantity(np.linspace(0, f.max_range, data.shape[-1] + 1), 'kilometers')

# Extract central latitude and longitude from the file
cent_lon = f.lon
cent_lat = f.lat

# Convert az, range to x,y
xlocs = rng * np.sin(np.deg2rad(az[:, np.newaxis]))
ylocs = rng * np.cos(np.deg2rad(az[:, np.newaxis]))
xlocs, ylocs = azimuth_range_to_lat_lon(az, rng, cent_lon, cent_lat)
```
Acknowledgments

• Thank you to Unidata, UCAR/NCAR/UCP for having me!
• Thank you to Drew Camron and Ryan May for their guidance and mentorship this summer!
• Thanks to Rhoen, Hassan, and the other interns for having a great summer together!