PySolo: A Python package to run Solo II, a radar editing software.

Author: Ammar Siddiqi

NCAR UCAR

Introduction

- **Solo II** is a display and hand-picked editing software program for radar observations, initially developed in the late 1980s in C
- A radar data file is broken up into many rays, with each ray representing a 1 dimensional array of values. Solo II's functions perform data manipulation on a single ray at a time.
- Solo II is used by many students, businesses and researchers. However, Solo II is a dated software and does not work out of the box for current computers.
- There is a newer radar editing program in development called Hawkeye. It seeks to incorporate and build upon current Solo II functionality.
- This brings the need to transform Solo II into a more dynamic, accessible, and extensible environment.

Objectives

- Run Solo II directly from a Python script by calling its C-functions directly.
- Hide the details of calling C-functions from Python by building a high-level importable package.
- Make package importable from a repository via pip.
- Test package with unit tests and real radar data from NetCDF files.
- Make Solo II platform independent, as originally, Solo II was limited to Mac/Linux computers.

Tools Used:

Py-Art

Results

- Introducing **PySolo:** A python interface for Solo II.
- Allows for users to run Solo II functions in a more simplified, pythonic way, with python built-in types.
- This package is uploaded to the PyPI public repository.
- Can perform operations over a 2D region, as compared to original Solo II functions, which were refined in a 1D space.
- Can be used alongside PyArt, a Python Radar toolkit to visualize radar data, to quickly display Solo II functions.
- Below are examples of two functions: ring zap and despeckle. Over 20 other functions are included.
- PySolo is adaptable to future changes in the Solo II source code.

Next Steps

- Solve the mystery of missed unfolds: the developers of PyArt have run into issues while on their own de-aliasing algorithm. PySolo has its own de-aliasing algorithm, allowing us to compare and contrast the algorithms and further investigate the issue.
- Integrate PySolo into PyArt.
- Implement "boundary masks" into PySolo. Boundary masks allows for users to create a shape for a region in which they desire to modify the enclosed data.
- Utilize the PySolo package to further develop the Solo II library for future changes.

Ring Zap

This function removes a ring in the data, centered around the radar. PyArt is used to read a radar file. Next, **from_km** and **to_km** are assigned to 30 and 35, respectively. The **km_between_gates** is obtained from the radar data file. Finally, the PySolo function **ring_zapped_mask** is executed, which takes field data "reflectivity" from the file, and variables assigned earlier. This returns an array which can then be graphed into a plot.

Code

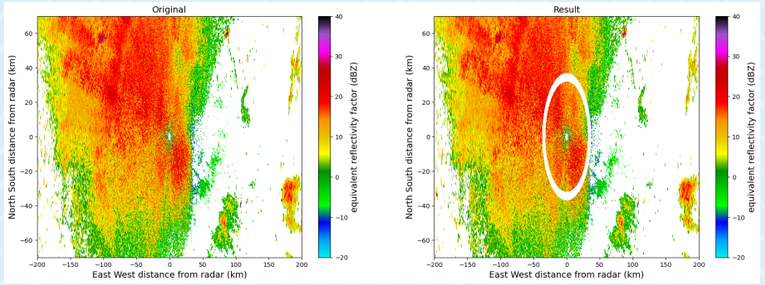
```python
import pyart
import pysolo_package as solo
import matplotlib.pyplot as plt

radar = pyart.io.read(path_to_data)

from_km = 25
to_km = 30
km_between_gates = radar.range["meters_between_gates"] / 1000
ring_zapped_mask = solo.ring_zap_masked(
    radar.fields["reflectivity"]["data"],
    from_km, to_km, km_between_gates
)
```

Graph

Ring Zap from 30 to 35 km



Despeckle

This function removes noise from a radar file. Using this function follows the same procedure as ring zap. Import the packages, and then read from the radar file. Then, define a number to **a_speckle**. The larger a_speckle is, the more noise will be removed. However, this comes at the cost of removing potentially valid data.

From there, **despeckle_mask** is executed, taking in data from 'reflectivity' and a_speckle

Code

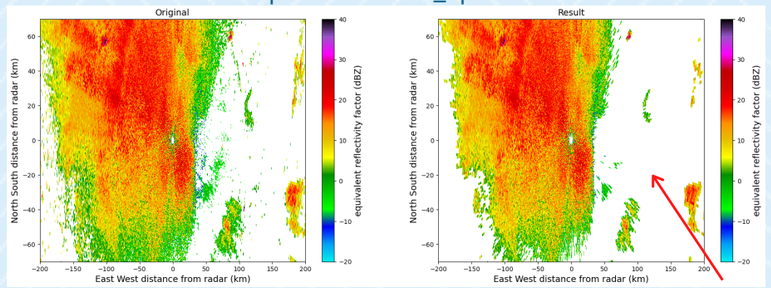
```python
import pyart
import pysolo_package as solo
import matplotlib.pyplot as plt

radar = pyart.io.read(path_to_data)

a_speckle = 7
despeckle_mask = solo.despeckle_mask(
    radar.fields["reflectivity"]["data"],
    a_speckle
)
```

Graph

Despeckle with a_speckle=10



Install

```
pip install pysolo
```

Acknowledgements

I would like to thank:

- Brenda Javornik, my mentor,
- Alex, Brad, Mike, and Wen-Chau from the HawkEdit users group,
- the PyArt team,
- our SUPER coordinators.
- NSF, NCAR & EOL

Install the package from **PyPI**, test it on **Google Colab**, or contribute to the project from **GitHub**

GitHub