MultiDop: An open-source, Python-powered, multi-Doppler radar analysis suite

Timothy J. Lang, Christopher J. Schultz

Corey K. Potvin

Robert Jackson, Scott Collis

Brenda Dolan
The Context

• NASA Weather program (under Tsengdar Lee) seeks to improve NASA severe weather observational and modeling capabilities - NASA STORM project, FY 2016

• Independent but parallel effort to VORTEX-Southeast

• Three Main Goals

  1. Expansion of North Alabama Lightning Mapping Array (NALMA)

  2. Advanced ensemble model severe weather forecasting

  3. Expand open-source tools for severe weather analysis
The Dream

Wouldn’t it be nice to have an open-source, Python-based toolkit for multi-Doppler wind syntheses?

• Three-dimensional winds from arbitrary radar networks

• Enable community-supported severe weather analyses

• Significantly lower barrier to entry for new users
Realizing the Dream, Part I - Python ARM Radar Toolkit (Py-ART)

• Developed at Argonne National Lab (Helmus and Collis 2016)

• Simplified File I/O

• Facilitates filtering via GateFilter object

• Automated Doppler velocity dealiasing

• Interpolation to a Cartesian grid

• Display of spherical and gridded data

• Advection correction under development

https://arm-doe.github.io/pyart/dev/auto_examples/index.html
Realizing the Dream, Part II - DDA C Application

• “Dual-Doppler Analysis” Developed at OU/CIMMS

• Based on 3D Variational Analysis (3DVAR)

• Mass conservation constraint becomes a tunable parameter

• Also tunable: Vorticity, Smoothness, Sounding weights

• 3DVAR often superior to traditional multi-Doppler methods (e.g., Gao et al. 1999, Potvin et al. 2012, North et al. 2017)

![Graphs showing True, 3DVAR, and Traditional Vertical Velocity](Image)

Total Cost Function $J = J_O + J_M + J_V + J_S$, where:
- $J_O$: Obs
- $J_M$: Mass
- $J_V$: Vort.
- $J_S$: Smooth Cont.
The Culmination of the Dream - **MultiDop**

- Developed at NASA Marshall Space Flight Center
- Python wrapper for DDA C-based application
- Python classes to bridge Py-ART and DDA
- DDA updated to accept Py-ART grid format
- Python install script for compiling both C and Python components
How Does It All Work, Then?

- MultiDop makes Py-ART and DDA work together
- A sample workflow is available as a Jupyter notebook

Use Py-ART to ingest, filter, correct, and grid radar data

Correct Grid variables for advection

Define parameters in dictionary and pass Grid files to MultiDop

MultiDop calls modified DDA C application using subprocess

Use MultiDop to convert DDA output to CF-compliant, Py-ART Grid file

Analysis Success! (hopefully)
Py-ART Advection Correction

- For radars that are non-synchronized, we need to determine and correct for advection of radial velocity patterns.

- We have implemented a image shift detection technique to get X/Y advection between volumes using cross correlation (same as in image stabilization)

- We also have implemented an image shifter using NDImage


- **To Do**: Combine forward and backward projected images, “Advective interpolation”
Define Parameters Step

- Tunable and user-defined parameters are handled via a dictionary

- ParamFile and CalcParamFile objects use this dictionary to create input scripts used by the DDA application

- Default values are used to fill in what end user does not provide
DDA Application Step

- MultiDop calls the DDA C application via the subprocess module

- Text output from the application is captured, but not displayed until after application completes

- Entire process usually takes a few minutes
MultiDop Checkout

North Alabama convection
- Supercell
- Multicell
- QLCS

Lessons Learned
- CEDRIC/MultiDop updraft locations and magnitudes qualitatively match
- MultiDop tunable parameters can greatly modify results
- Pay special attention to horizontal and vertical smoothing
MultiDop Checkout (cont.)

Northern Colorado DC3 Cases
• CSU-CHILL and CSU-Pawnee
• Volumes from 5 & 6 June 2012
• Multicellular convection

Lessons Learned
• MultiDop w/in ~1 m s⁻¹ of CEDRIC
• Good spatial correspondence
• MultiDop ~10x slower than CEDRIC, but many times easier to use!
• Pay special attention to Py-ART gridding
MultiDop Checkout (cont.)

Northern Australia Convection
- CPOL (Darwin) & Berrima S-band
- ~40,000 volumes!
- Cluster: 1 instance MultiDop/core

Lessons Learned
- Needed strong mass continuity constraint (e.g., C2b = 1500) to suppress high-altitude noise in W
- Used Leise filter and strong horizontal smoothing to remove artifacts near edge of lobes
- Took advantage of 4 day⁻¹ soundings to help the retrieval
MultiDop is available at https://github.com/nasa/MultiDop

Current version = 0.3, tested and working under Python 2.7 and 3.6

Also requires numpy, Py-ART, xarray, C compilers, and netCDF libraries

If you use MultiDop, you MUST cite the following papers:


Possible Future Directions

• Refactor C application to C library, so that it can be imported directly to Python via ctypes module

• Identify or develop efficient cost minimization modules in Python, then port C code to Python

• Better web-based documentation

• Testing module

• Your help is appreciated!
Also part of NASA STORM – Equip Py-ART to output GeoTIFFs for easier GIS integration

Gridded merged radar reflectivity from NEXRADs and ARMOR, for N. Alabama case in March 2016, as viewed in GIS application
If you like MultiDop, you may also be interested in:

**CSU_RadarTools** – Diverse toolkit for radar analysis and processing

**DualPol** – Polarimetric radar hydrometeor ID, DSD, rainfall, etc.

**SingleDop** – 2D low-level wind retrievals from Doppler radar

**PyBlock** – Beam blockage correction for polarimetric radar

**PyTDA** – Turbulence retrievals from Doppler radar

**MMM-Py** – MRMS 3D radar reflectivity mosaic ingest and analysis

**PyAMPR** – Work with NASA AMPR airborne microwave radiometer data