# Py-ART Science lead summary of Roadmap reviews

**Scott Collis, Argonne National Laboratory**

## Background

The Python-ARM Radar Toolkit, Py-ART, is a community based open source package used to interact with weather radar data. Py-ART evolved from a desire to make algorithm work funded by ARM publicly available and to provide users a default framework for working with ARM radar data that accelerated adoption as a Value Added Product. Additionally, making algorithms available allows user customization as default parameters do not suit all needs.

Py-ART was released publicly in 2013 and there was a pleasantly surprising level of community engagement. As the desire of the community to contribute increased and the review of contributions and management workload of the ARM funded team grew the need for a roadmap which prioritized and focused Py-ART development became apparent. The role of the roadmap is to:

- Provide a vision statement for Py-ART consistent with ARM stakeholder needs.
- List key capabilities the team will prioritize for inclusion, via a GitHub "Pull Request" into the package.
- Where needed enable ARM funded development for key modules.
- Define what Py-ART should be able to do in five years. Define success for the project.

In preparing the roadmap collecting stakeholder information is essential. This began with a survey of the community (results available on GitHib [1]). Following the drafting of the roadmap specific stakeholder input was collected from the Chair of the Radar Science and Engineering Team. Finally a set of questions was crafted [2] and sent to four ASR funded scientists from Universities and other institutions. These scientists were chosen as to represent a range of users from technical radar experts to downstream data users. This document is the summary, written by the Science Lead for Py-ART, for ARM leadership.

## Key points of agreement between reviewers

Reviewers agreed to make their raw feedback anonymously available on GitHub [3]. We are very grateful to this team of scientists for their feedback. While there was a range of responses there were some key areas of consensus:

- Py-ART is time saving and a worthwhile activity by ARM.
- The Py-ART roadmap is generally on-point. With a few modifications.
- The ability to ingest WRF and other data into Py-ART's grid data model is desired.
- There should be a focus on basic corrections and QC. For example improving Doppler velocity Dealiasing and Phase Processing.
- Updating Py-ART to work with the new python mapping backend, Cartopy is both needed and a worthwhile effort.
- Multi-Doppler retrievals are desired by the community however Py-ART's strength of making the use of such retrievals so easy could lead to misuse of the technique. Such an endeavour would need to be done carefully.
- Both reviewers one and two believe Py-ART is not the place for simple grid statistics (CFADs) and retrievals (rainfall rate, particle ID). They should either be done by PIs or be in other open source toolkits such as CSU_radartools [4]. Reviewer four did not discuss this point much while reviewer

three was in disagreement with reviewers one and two by stating it would be good to have Py-ART as a "one stop shop".

In terms of the specific items from the draft roadmap:

- **Support for Multi-Doppler efforts:** Support for this is mixed. There is a clear concern that it would be difficult to implement a system that would be robust and not open to misinterpretation. *Support: Mixed*
- **Volume summary statistics:** There is concern that this really is the role of the end user. However there is disagreement as summary statistics can aid in data discovery. *Support: Mixed*
- **Cell Tracking:** There was a general consensus that this capability would be a nice addition to Py-ART or in a dependant toolkit. Reviewer one went further to suggest a study would be needed to determine the efficacy of various techniques. *Support: Strong*
- **Cartopy Backend:** Strong consensus that this is required upgrade to Py-ART. The ability to plot radar data on maps should not be lost. *Support: Very strong*
- **Support for radar spectra**: It is hard to gauge the level of support from the language in the reviews. Reviewer one supports radar spectra processing from point of view of QC. Reviewer three discusses it in terms of model comparison which is more in-line with the thinking of the Py-ART Science Lead although Py-ART could be used to develop new QC techniques that were then implemented on radar hardware. Reviewer four supports ARM developing the data model for spectra and then other teams (such as Brookhaven) helping by contributing code. *Support: Mixed to strong*
- **Fixing existing processing code:** It is clear Py-ART's efforts in dealiasing and phase processing are appreciated by the reviewers. The take home from the reviews is they want us to go even further and expand the capabilities of Py-ART. In reading the reviews it can be concluded that this is the area Py-ART can have the highest value to the community in reducing the person-hours required to do good science. *Support: Very Strong*

## General comments beyond the roadmap

There were several themes that emerged from the reviews:

- The fact that Py-ART works in the Cloud and can work with multiple data formats, especially NEXRAD, is appreciated. Reviewers see this as ARM leading the way.
- At least one reviewer acknowledges the value of Py-ART as a "base toolkit" on which other code can use.
- More examples, cookbooks, and outreach is needed and is a key part in maximizing Py-ART's impact.

## Recommendations in modifying the roadmap

To react to feedback from the reviewers the Py-ART Science Lead proposes the following action items when modifying the roadmap:

- Work in QVPs, CFADs, et al be de-emphasised. Something that is nice to have but will not be a priority.
- Work in Multi-Doppler be restricted to ensuring Py-ART's compatibility with the NASA lead Multi-Dop package. Py-ART's value-adding is firmly in the pre-calculation QC.
- Add an item to the roadmap enabling ingest from WRF files to Py-ART's grid data model. The fastest and most sustainable way to do this may be to enable support within Py-ART for Xarray [5] grids.
- Expand on the section on QC, move it to the top of the roadmap as a key priority.
- Note the support of Cartopy as a back end is a priority (2nd behind QC).
- Restrict development on handling radar spectra to the development of a data model.

[1] https://github.com/ARM-DOE/pyart-roadmap/tree/master/survey_data

[2] https://github.com/ARM-DOE/pyart-roadmap/blob/master/roadmap/targeted_request_for_review.txt

[3] https://github.com/ARM-DOE/pyart-roadmap/blob/master/reviews/concatinated_reviews.md

[4] https://github.com/CSU-Radarmet/CSU_RadarTools

[5] http://xarray.pydata.org/en/stable/