RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

METEO FRANCE

Developing Pyrad

Jordi Figueras i Ventura
Pyrad course

Contents

- Introduction
- Step by step approach to contributing

1. Introduction

Principles

- Collaboration to Pyrad development is a wide field, not only code contribution:
  - Use reports
  - Bug reports
  - Feature requests
  - Contributing to code discussions
- Approving new features to the open repository is the responsibility of the PIs from Météo-France and MeteoSwiss
- There are two **permanent branches** *dev* and *master*. New developments should target the *dev* branch. The master branch is only used for new stable releases
- The Pyrad approach to CI is to make sure that overall functionality is maintained. That means that CI tests automatically run a set of minimal config files and tests whether the output data is as is expected (concept currently under development)

- **2. Step by step approach**

Step by step approach

1. Make sure that the feature you need is not yet available in Pyrad, Py-ART or wradlib:
    - Wradlib and Py-ART functions can be called by Py-ART with ease and are already basic dependences

Step by step approach

2. Open an issue in the pyrad github issues page describing the desired new feature :

- Input fields (specify if the input field does not yet exist)
- Output fields and/or data (if the output field does not yet exist in Pyrad specify it)
- Auxiliary input data (does it need extra configuration files, auxiliary data, etc. ?)
- User-defined parameters
- Input data object required (radar object, grid object, other ...)
- Output data object (radar object, grid object, a csv file ...)
- Dataset family to which it belongs (e.g. VOL, GRID, a new family, ...)
- Py-ART module and file where to write the processing function (if it belongs to a separate family of functionalities the functions should be stored in a separate file)
- Pyrad proc file where to write the call to Py-ART

Step by step approach : Py-ART development

3. Once the new development has been agreed with the Pyrad PI, fork the dev branches of Py-ART and Pyrad

4. Write the required main processing function in the suitable MeteoSwiss Py-ART file (typically within the **correct** or **retrieve** modules). Auxiliary general purpose functions may be written in other modules (e.g. **filters, util**)

5. Make available the Py-ART functions by declaring it in the *__init__.py* file of the Py-ART module where they have been written. Make sure the function is mentioned in the docstring

6. If necessary, declare new fields in the pyart config file (as for example the Météo-France Py-ART config file)

Step by step approach: Pyrad development

7. Write the call to the Py-ART function in the suitable file of the Pyrad proc module

8. Make available the Pyrad proc function by declaring it in the __init__.py file of the Pyrad proc module where it has been written. Make sure that the function is mentioned in the docstring

8. Assign a keyword to the pyrad function in the process_aux.py file. Make sure the keyword is described in the docstring

9. If necessary, assign a Pyrad field keyword to the new fields in *get_fieldname_pyart* function in io_aux.py

Step by step approach : pull request

10. Make sure your new code complies with the Python PEP 8 by running pylint and pycodestyle

11. Make sure all new functions have docstrings that follow the current template

12. Write a minimal config file that uses the new feature and make sure it produces the desired output

13. Create a pull request to the dev branches of Py-ART and Pyrad. In the description of your PR upload the config files you have used and if possible the results and the data used

14. If the code passes all CI tests the Pyrad PIs will review your contribution and eventually integrate it into the dev branch

Step by step approach: new release

15. The Pyrad PI will merge the dev branches into master
16. New PyPI packages pyart-mch and pyrad-mch will be created
17. New conda packages pyrad-mch and pyrad-mch will be available automatically in conda-forge out of the PyPI packages after some hours

Another conda package called arm_pyart is identical to pyrad-mch but using the Original Py-ART from arm-doe and therefore with diminished functionality


Thank you!
Grazie mille!
Moltes Gràcies!
Merci!