Testing the smalldata_tools Package

Hannah Zhang, Vincent Esposito
LCLS, SLAC National Acceleration Laboratory

Introduction

The *smalldata_tools* package is used in data analysis at LCLS. It provides a suite of functions for processing and analyzing data obtained from various detectors. Ensuring the reliability and accuracy of the code throughout the development and maintenance cycles of the package is thus crucial, just like in any software development process. We thus implemented a series of tests based on the PyTest framework to help the continuous development of the package.

PyTest Overview

PyTest is a testing framework that simplifies and automate test execution and implementation. It generates concise and readable test code, making it easier to understand test cases and their results. PyTest's assertion capabilities enable effortless validation of expected output properties.

Methods and results

- We structured our tests by creating individual test functions, each dedicated to a specific combination of detector and analysis function.
- **Fixtures** provide a way to set up and share resources needed by multiple tests. They allow us to define reusable setup and teardown code, making our tests more concise and maintainable. For example, we used them to instantiate the various detector classes.

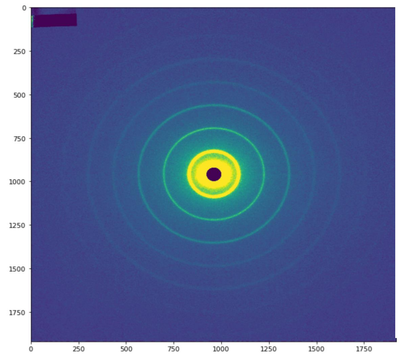
```
@pytest.fixture(scope='module')
def detector(request, datasource):
    ds, run = datasource
    detector_name = request.param.get('name')
    detector = DetObject(detector_name, ds.env(), run)
    yield detector
```

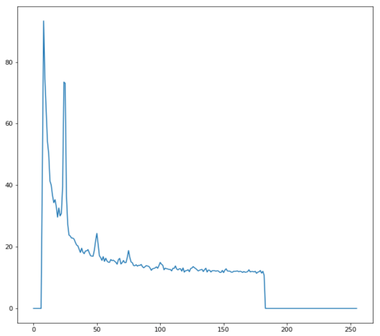
The **yield** statement marks the separation between the setup and teardown phases of the fixture. After the test is executed, the teardown steps, if any, are performed automatically.

Within each test function, we leveraged fixtures to utilize the predefined environment and data for testing purposes. Following a convention, we organized the tests into separate test files, with each file corresponding to a specific detector.

```
@pytest.mark.parametrize('datasource', [{'exp': 'xpptut15', 'run': 650}], indirect=True)
@pytest.mark.parametrize('detector', [{'name': 'jungfrau1M'}], indirect=True)
def test_detector_type(datasource, detector):
    logger.info('Running detector type test')
    det = detector
    assert(isinstance(det, smalldata_tools.DetObject.JungfrauObject))
    logger.info('Pass the Detector_type test')
```

The tests executed successfully, validating the correct functioning of ROI, Projection, Spectrum, Azimuthal Integration, Droplet, and Photon function. The test confirmed that the package produces the expected outputs for various input scenarios.





Conclusions

The tests conducted ensured the correctness and robustness of the software functionality. Running this series of tests at the end of a development of maintenance cycle will ensure that the package functions as expected after each iteration, allowing for safe deployment to new experiments.

Acknowledgments

Use of the Linac Coherent Light Source(LCLS), SLAC National Accelerator Laboratory, is supported by the US Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515