Pytest and Travis

Why test?

- Testing is the foundation of solid software development.
- Gives you confidence that you can rely on individual function.

How to test

- Prepare the environment.
- Prepare expected result.
- Call the code under test.
- Assert that the actual result matches the expected result.

Guidelines

- Commitment
  - Takes time but often makes the process faster and more reliable in the long run
  - Always make changes to tests with changes to your code
  - Set up EARLY!
- Discipline - make sure tests always pass
- Automate
  - Run with every commit - TravisCI
- Make them FAST!
- Untested code is broken

Tools

- Pytest
- IPytest
- Travis

Get started

- Clone our github repo: git clone https://github.com/ContextLab/CDL-tutorials.git or pull latest
- Follow the Docker tutorial if you haven't already completed it
- Launch docker

Launch docker

```
$ docker start CDL && docker attach CDL
```

Install pytest

```
$ pip install pytest
```

Test

```
# Create a really simple function in fun.py

def func(x):
    return x + 1


# Create a test_sample.py with tests for func

from fun import func

def test_correct_func():
    assert func(4) == 5

def test_incorrect_func():
    assert func(3) == 5
```

Run test

```
$ cd /test/folder
$ pytest
```

Pytest will run all files of the form test_*.py or *_test.py in the current directory and its subdirectories.

Raise error

```
# Create a test_sysexist.py and`raises` helper
to assert that some code raises an exception

import pytest

def f():
    raise SystemExit(1)

def test_mytest():
    with pytest.raises(SystemExit):
        f()
```

```
$ pytest test_sysexit.py
```

Pytests can also run invidivudally. In this example we will run the module by passing its filename.

Multiple tests in a class

```
# Create a test_class.py with multiple tests

class TestClass(object):
    def test_one(self):
        x = "this"
        assert 'h' in x

    def test_two(self):
        x = "hello"
        assert hasattr(x, 'check')
```

```
$ pytest -q test_class.py
```

Function with 'quiet' reporting mode (-q)

Temporary directory

```python
# Create a test_tmpdir.py to request a unique
temporary directory for functional tests

def test_needsfiles(tmpdir):
    print(tmpdir)
    assert 0
```

```
$ pytest -q test_tmpdir.py
```

IPytest

- You can follow the same examples but in IPython form in Testing.ipynb
  - Uses ipytest.magics

Travis CI

- Free for public projects, but $ for private
- Continuous integration service used to build and test software projects hosted at GitHub
  - Activate Travis CI for a repository
  - GitHub will notify it whenever new commits are pushed to that repository or a pull request is submitted.
  - Travis CI will check out the relevant branch and run the commands specified in .travis.yml, which runs automated tests.

Summary

- Set up EARLY!
- Keep them fast
- Make sure tests always pass
- TravisCI
- Untested code is broken