derpy Documentation

Release 0.1.1

Rob Scott

Sep 21, 2018

Contents:

CHAPTER 1

derpy

Financial derivatives and portfolio analysis tools for python

- Free software: MIT license
- Documentation: https://derpy.readthedocs.io.
- PyPi: https://pypi.org/project/derpy/
- Gitub: https://github.com/rjdscott/derpy

1.1 How to get up and running

to include the module in your project, you can simply use *pip install derpy* then in your python project

```
import derpy
print(derpy.__version__) # returns '0.0.1'
```

1.2 Example uses

1.2.1 Bonds

```
from derpy import bond as bd

px = 95.0428
face_val = 100.0
mat = 1.5
cpn_frq = 2
cpn_rate = 5.25
ytm = 5.5
```

(continues on next page)

(continued from previous page)

```
print('    Price: {}'.format(bd.bond_price(face_val, mat, ytm, cpn_rate, cpn_frq)))
print('    Yield: {}'.format(bd.bond_ytm(px, face_val, mat, cpn_rate, cpn_frq)))
print('   ModDur: {}'.format(bd.bond_duration(px, face_val, mat, cpn_rate, cpn_
↪frq)[0]))
print('   MacDur: {}'.format(bd.bond_duration(px, face_val, mat, cpn_rate, cpn_
↪frq)[1]))
print('Convexity: {}'.format(bd.bond_convexity(px, face_val, mat, cpn_rate, cpn_frq)))
```

1.2.2 Options

```
from derpy.options import black_scholes_merton as bsm

# usage method 1: use function wrapper
input = ['call', 20, 21, 0.20, 0.1, 0.0002, 0]
call_price = bsm.option_pricing(bsm.euro_option, input)
call_gamma = bsm.option_pricing(bsm.gamma, input)

# usage method 2: call individual functions
put_price = bsm.euro_option('put', 20, 21, 0.2, 0.1, 0.0002) # div_yield is optional
put_gamma = bsm.gamma('put', 20, 21, 0.2, 0.1, 0.0002, 0.0001)

print(call_price)  # return 0.16384395..
print(call_gamma)  # return 0.23993880..
print(put_price)  # return 1.16342..
print(put_gamma)  # return 0.2399107..
```

1.2.3 Portfolio analysis

```
from derpy import portfolio as pt

securities = ['AAA', 'BBB']
positions = [[11, 10], [12, 10], [13, 10], [13, 11], [13, 12]]
prices = [[10, 10], [11, 10], [12, 10], [12, 10], [12, 10]]
dates = ['2018-07-01', '2018-08-01', '2018-09-01', '2018-10-01', '2018-11-01']

df_positions = pd.DataFrame(data=positions, columns=securities, index=dates)
df_prices = pd.DataFrame(data=prices, columns=securities, index=dates)

p = pt.Portfolio(names=securities, positions=df_positions, prices=df_prices)

print(p.sec_values())
print(p.sec_weights())
print(p.portfolio_value())
print(p.portfolio_returns())
```

CHAPTER 2

Installation

2.1 Stable release

To install derpy, run this command in your terminal:

```
$ pip install derpy
```

This is the preferred method to install derpy, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for derpy can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/rjdscott/derpy
```

Or download the tarball:

```
$ curl  -OL https://github.com/rjdscott/derpy/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

3.1 Bonds

```
from derpy import bond as bd

px = 95.0428
face_val = 100.0
mat = 1.5
cpn_frq = 2
cpn_rate = 5.25
ytm = 5.5

print('    Price: {}'.format(bd.bond_price(face_val, mat, ytm, cpn_rate, cpn_frq)))
print('    Yield: {}'.format(bd.bond_ytm(px, face_val, mat, cpn_rate, cpn_frq)))
print('   ModDur: {}'.format(bd.bond_duration(px, face_val, mat, cpn_rate, cpn_
↪frq)[0]))
print('   MacDur: {}'.format(bd.bond_duration(px, face_val, mat, cpn_rate, cpn_
↪frq)[1]))
print('Convexity: {}'.format(bd.bond_convexity(px, face_val, mat, cpn_rate, cpn_frq)))
```

3.2 Options

```
from derpy.options import black_scholes_merton as bsm

# usage method 1: use function wrapper
input = ['call', 20, 21, 0.20, 0.1, 0.0002, 0]
call_price = bsm.option_pricing(bsm.euro_option, input)
call_gamma = bsm.option_pricing(bsm.gamma, input)

# usage method 2: call individual functions
put_price = bsm.euro_option('put', 20, 21, 0.2, 0.1, 0.0002) # div_yield is optional
```

(continues on next page)

(continued from previous page)

```
put_gamma = bsm.gamma('put', 20, 21, 0.2, 0.1, 0.0002, 0.0001)

print(call_price)  # return 0.16384395..
print(call_gamma)  # return 0.23993880..
print(put_price)  # return 1.16342..
print(put_gamma)  # return 0.2399107..
```

3.3 Portfolio analysis

```
from derpy import portfolio as pt

securities = ['AAA', 'BBB']
positions = [[11, 10], [12, 10], [13, 10], [13, 11], [13, 12]]
prices = [[10, 10], [11, 10], [12, 10], [12, 10], [12, 10]]
dates = ['2018-07-01', '2018-08-01', '2018-09-01', '2018-10-01', '2018-11-01']

df_positions = pd.DataFrame(data=positions, columns=securities, index=dates)
df_prices = pd.DataFrame(data=prices, columns=securities, index=dates)

p = pt.Portfolio(names=securities, positions=df_positions, prices=df_prices)

print(p.sec_values())
print(p.sec_weights())
print(p.portfolio_value())
print(p.portfolio_returns())
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/rjdscott/derpy/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

4.1.4 Write Documentation

derpy could always use more documentation, whether as part of the official derpy docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/rjdscott/derpy/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *derpy* for local development.

1. Fork the *derpy* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/derpy.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv derpy
$ cd derpy/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 derpy tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/rjdscott/derpy/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_derpy
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Credits

5.1 Development Lead

- Rob Scott <rob@rjdscott.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.1.0 (2018-09-19)

- Released option pricing
- Released portfolio analysis
- Added testing for Options and Portfolios

CHAPTER 7

Indices and tables

- genindex
- modindex
- search