qef Documentation

Release 0.1.0

Jose Borreguero

Dec 27, 2017

Contents

Contents:

CHAPTER 1

qef

quasielastic fitting

- Free software: MIT license
- Documentation: https://qef.readthedocs.io.

1.1 Features

- TODO

1.2 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER 2

Installation

2.1 Stable release

To install qef, run this command in your terminal:

```
$ pip install qef
```

This is the preferred method to install qef, 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 qef can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/jmborr/qef
```

Or download the tarball:

```
$ curl  -OL https://github.com/jmborr/qef/tarball/master
```

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

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

CHAPTER 3

Usage

To use qef in a project:

```
import qef
```

CHAPTER 4

Modules

4.1 Models

4.1.1 DeltaDiracModel

class qef.models.deltadirac.**DeltaDiracModel**(*independent_vars=['x'], prefix='', missing=None, name=None, **kwargs*)

Bases: lmfit.model.Model

A function that is zero everywhere except for the x-value closest to the center parameter.

At value-closest-to-center, the model evaluates to the amplitude parameter divided by the x-spacing.

Fitting parameters:

- integrated intensity `amplitude` A
- position of the peak `center` E_0

guess(*y*, *x=None*, ***kwargs*)

Guess starting values for the parameters of a model.

Parameters

- **y** (ndarray) – Intensities
- **x** (ndarray) – energy values
- **kwargs** (*dict*) – additional optional arguments, passed to model function.

Returns parameters with guessed values

Return type Parameters

qef.models.deltadirac.**delta_dirac**(*x*, *amplitude=1.0*, *center=0.0*)

function is zero except for the x-value closest to center.

At value-closest-to-center, the function evaluates to the amplitude divided by the x-spacing.

Parameters

- **x :class:‘~numpy:numpy.ndarray‘** – domain of the function, energy
- **amplitude** (*float*) – Integrated intensity of the curve
- **center** (*float*) – position of the peak

Returns **values** – function values over the domain

Return type ndarray

4.1.2 StretchedExponentialFTModel : Fourier transform of the stretched exponential

class qef.models.strexpft.**StretchedExponentialFTModel**(*independent_vars=['x'], prefix='', missing=None, name=None, **kwargs*)

Bases: lmfit.model.Model

Fourier transform of the symmetrized stretched exponential

$$S(E) = A \int_{-\infty}^{\infty} dt/h e^{-i2\pi(E-E_0)t/h} e^{|\frac{x}{\tau}|^\beta}$$

Normalization and maximum at $E = E_0$:

$$\int_{-\infty}^{\infty} dE S(E) = Amax(S) = A\frac{\tau}{\beta}\Gamma(\beta^{-1})$$

Uses scipy.fftpack.fft for the Fourier transform

Fitting parameters:

- integrated intensity `amplitude` A
- position of the peak `center` E_0
- nominal relaxation time `tau` ` τ
- stretching exponent `beta` β

If the time unit is picoseconds, then the reciprocal energy unit is mili-eV

guess(*y*, *x=None*, ***kwargs*)

Guess starting values for the parameters of a model.

Parameters

- **y** (ndarray) – Intensities
- **x** (ndarray) – energy values
- **kwargs** (*dict*) – additional optional arguments, passed to model function.

Returns parameters with guessed values

Return type Parameters

qef.models.strexpft.**strexpft**(*x*, *amplitude=1.0*, *center=0.0*, *tau=10.0*, *beta=1.0*)

Fourier transform of the symmetrized stretched exponential

$$S(E) = A \int_{-\infty}^{\infty} dt/h e^{-i2\pi(E-E_0)t/h} e^{|\frac{x}{\tau}|^\beta}$$

Normalization and maximum at $E = E_0$:

$$\int_{-\infty}^{\infty} dES(E) = A$$

$$max(S) = A\frac{\tau}{\beta}\Gamma(\beta^{-1})$$

Uses `fft()` for the Fourier transform

Parameters

- **x** (`ndarray`) – domain of the function, energy
- **amplitude** (*float*) – Integrated intensity of the curve
- **center** (*float*) – position of the peak
- **tau** (*float*) – relaxation time.
- **beta** (*float*) – stretching exponent
- **If the time units are picoseconds, then the energy units are mili-eV.**

Returns **values** – function over the domain

Return type `ndarray`

4.2 Operators

4.2.1 Convolution operator

class qef.operators.convolve.**Convolve**(*resolution*, *model*, ***kws*)

Bases: `lmfit.model.CompositeModel`

Convolution between model and resolution.

It is assumed that the resolution FWHM is energy independent. Non-symmetric energy ranges are allowed (when the range of negative values is different than that of positive values).

eval(*params=None*, ***kwargs*)

qef.operators.convolve.**convolve**(*model*, *resolution*)

Convolution of resolution with model data.

It is assumed that the resolution FWHM is energy independent. We multiply by spacing dx of independent variable x.

$$(model \otimes resolution)[n] = dx * \sum_m model[m] * resolution[m - n]$$

Parameters

- **model** (*numpy.ndarray*) – model data
- **resolution** (*numpy.ndarray*) – resolution data

Returns

Return type numpy.ndarray

CHAPTER 5

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:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/jmborr/qef/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.

5.1.2 Fix Bugs

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

5.1.3 Implement Features

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

5.1.4 Write Documentation

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

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/jmborr/qef/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 :)

5.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/qef.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 qef
$ cd qef/
$ 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 qef 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.

5.3 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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/jmborr/qef/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests.test_strexpft
```

CHAPTER 6

Credits

6.1 Development Lead

- Jose Borreguero <borreguero@gmail.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

History

7.1 0.1.0 (2017-12-13)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Donwload the documentation

Python Module Index

q

Index