reasonable Documentation

Release 1

Fastats Contributors

Sep 16, 2017

Contents:

1 Quickstart 3

2 Floats 5

3 The x object 7

4 Gaussian 9

reasonable makes it easy to compare floats using the generative testing package *hypothesis*.

All the documentation is on this one page, so it's easy to search from your browser.

CHAPTER 1

Quickstart

To use *reasonable*, replace the *floats* import from *hypothesis* with

```
from reasonable import floats, x
```

floats is a replacement for the *hypothesis floats* function.

x is a helper object which makes it trivial to define constraints on the generated floats.

For example, if you were minimising a function and wanted to ensure that all input values above 1.5 will converge on the correct answer, you could write a test like

```
from hypothesis import given
from reasonable import floats, x
from pytest import approx


@given(floats(x > 1.5))
def test_my_minimisation(n):
    result = minimise(my_func, x0=n)
    assert result == approx(0.23456789)
```

Alternatively, you could embrace *property testing* as opposed to the more traditional *unit-testing* shown above, by validating that the result of the minimisation is less than the result of the objective function for every input

```
from hypothesis import given
from reasonable import floats, x


@given(floats(x > 1.5))
def test_actually_minimises(n):
    result minimise(my_func, x0=n)
    assert result < my_func(n)
```

Both of these are useful tests to perform, however neither is easy using just *hypothesis* because of the float values generated - with the *reasonable* package, we generate more reasonable float values, allowing simpler testing :)

CHAPTER 2

Floats

CHAPTER 3

The *x* object

The *x* object is an instantiated object, not a class or function definition.

It can be imported from the top-level *reasonable* module:

```
from reasonable import x
```

It is a helper object - by using it in a comparison with an int or float, it will set the constraints on the generated values.

This will generate uniformly-distributed random floats between 1.0 and *sys.maxsize*:

```
float(1.0 < x)
```

CHAPTER 4

Gaussian