Python and gradients

Python

- python
  - define function, input and output

    ```
    def f(input):
        output = input**3
        return output
    ```
- Numpy
  - Arrays
  - Element-wise operations
  - Sizes/shapes of arrays
  - `x+y`, `np.dot`, and `x*y` “does the loop for you”

Python

- define two vectors of 3 elements

```
a = np.array([5, 4, 2])
b = np.array([10, 2, 4])
```

- add two vectors (element-wise addition)

```
a + b
```

- apply the inner product between a and b

```
np.dot(a, b)
```

We can check that the gradient implementation is correct

- Finite-difference approximation of the gradient
  - $\epsilon = 1e - 4$
  - Gradient definion:

$$\frac{\partial f(x)}{\partial x_i} = \lim_{\epsilon \to 0} \frac{f(x_i + \epsilon) - f(x_i)}{\epsilon}$$

  - Therefore,

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x_i + \epsilon) - f(x_i)}{\epsilon}$$

Code to approximate the gradient

- Formula
$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x_i + \epsilon) - f(x_i)}{\epsilon}$$
- Create a simple gradient check for single-variable function

```
def forward_diff(x, f):
    # Approximates gradient
    eps = 1e-4
    g_approx = f(x + eps) - f(x)
    g_approx /= eps
```

Code to approximate the gradient

- Using scipy:

```
import numpy as np
from scipy.optimize import approx_fprime

approx = approx_fprime(x0, foo, 1e-4)
exact = foo_grad(x0)

print("My gradient      : %s" % exact)
print("Scipy's gradient: %s" % approx)

# Assert that the two are almost equal
np.testing.assert_almost_equal(approx, exact, 3)
```