Chap 2 - Element-wise Matrix Operations in NumPy

March 24, 2018

In Python, what do we do when we want to add a number to every item in the list?

```
In [1]: number = 10
        v = [1,2,3,4,5] # len(v) = 5
        for i in range(len(v)): #range(len(v)) = range(0,5)
            v[i] += number
        print(v)
```

```
[11, 12, 13, 14, 15]
```

In NumPy, the code above can be done by

```
In [2]: # NumPy's array

        import numpy as np

        number = 10
        v = np.array([1,2,3,4,5]) + number
        print(v)
```

```
[11 12 13 14 15]
```

```
In [3]: # NB v1 != v2
        v1 = np.array([1,2,3,4,5])
        v2 = np.ndarray([1,2,3,4,5])
        #print('v1:', v1)
        #print('v2:', v2)
        print('Shape of v1:', v1.shape)
        print('Shape of v2:', v2.shape)
```

```
Shape of v1: (5,)
Shape of v2: (1, 2, 3, 4, 5)
```

```
In [4]: # NB the type of NumPy array is ndarray
        print('type(v1):', type(v1))
        print('type(v2):', type(v2))
        print('v1 is instance of ndarray:', isinstance(v1, np.ndarray))
```

```
type(v1): <class 'numpy.ndarray'>
type(v2): <class 'numpy.ndarray'>
v1 is instance of ndarray: True
```

NumPy has functions such as add, multiply etc but also supports using the standard math operators.

```
In [5]: v = np.array([1,2,3,4,5]) # shape: (5,)
        v = v.reshape(1,5) # 1x5
        w = np.array([[1,0],[0,1],[0,0],[1,1],[1,1]]) # 5x2

        # Multiply matrix with a scalar
        x = np.multiply(v, number)
        y = v * number
        print('x:', x)
        print('y:', y)

        # Matrix multiplication
        print('v.shape: ', v.shape)
        print('w.shape: ', w.shape)
        # multiply
        z = np.matmul(v,w)
        print('z.shape: ', z.shape)
```

```
x: [[10 20 30 40 50]]
y: [[10 20 30 40 50]]
v.shape:  (1, 5)
w.shape:  (5, 2)
z.shape:  (1, 2)
```

To reset a matrix to zero, we simply multiply it with zero and assign the result back to the matrix.

```
In [6]: print('x:', x)

        # Reset the matrix
        x *= 0
        print('x:', x)
```

```
x: [[10 20 30 40 50]]
x: [[0 0 0 0 0]]
```

0.0.1 Element-wise Matrix Operations

When we do `v *= v`, it works beause it's an element-wise multiplication!

```
In [7]: print('v:', v)
        v *= v # square
        print('v:', v)

v: [[1 2 3 4 5]]
v: [[ 1  4  9 16 25]]
```

However, when we do matrix operation, we have to consider whether the **shapes** of the matrices are **compatible**.

```
In [8]: # Another example
        a = np.array([[1,2],[3,4]])
        b = np.array([[1,0],[0,1]])
        print('a + b =\n', (a+b))
        print('a * b =\n', (a*b))

a + b =
 [[2 2]
 [3 5]]
a * b =
 [[1 0]
 [0 4]]
```