L01s01

September 13, 2018

0.1 Python Fundamentals

Ref - [DWAP] Derivatives Analytics with Python by Yves Hilpisch - Appendix of [DWAP]

0.1.1 First Steps

```
In [1]: 3 + 4
```

```
Out[1]: 7
```

```
In [2]: 3 / 4
```

```
Out[2]: 0.75
```

```
In [3]: 3 / 4.
```

```
Out[3]: 0.75
```

```
In [4]: a = 3
```

```
In [5]: # sin(a)
```

```
In [6]: from math import sin
```

```
In [7]: sin(a)
```

```
Out[7]: 0.1411200080598672
```

```
In [8]: b = 4
```

```
In [9]: import math
```

```
In [10]: math.sin(b)
```

```
Out[10]: -0.7568024953079282
```

```
In [11]: def f(x):
             return x ** 3 + x ** 2 - 2 + math.sin(x)
```

```
In [12]: f(2)
```

```
Out[12]: 10.909297426825681

In [13]: f(a)

Out[13]: 34.141120008059865

In [14]: %run A_pyt/a_first_program.py

f(a) = 34.141
f(b) = 77.243
```

0.1.2 Array Operations

```
In [15]: import numpy as np

In [16]: a = np.arange(0.0, 20.0, 1.0)  # (start, end, step)

In [17]: a

Out[17]: array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12.,
                13., 14., 15., 16., 17., 18., 19.])

In [18]: a.resize((4, 5))

In [19]: a

Out[19]: array([[ 0.,  1.,  2.,  3.,  4.],
                [ 5.,  6.,  7.,  8.,  9.],
                [10., 11., 12., 13., 14.],
                [15., 16., 17., 18., 19.]])

In [20]: a[0]  # first row

Out[20]: array([0., 1., 2., 3., 4.])

In [21]: a[3]  # fourth (=last) row

Out[21]: array([15., 16., 17., 18., 19.])

In [22]: a[1, 4]  # second row, 5th (=last) element

Out[22]: 9.0

In [23]: a[1, 2:4]  # second row, third & forth element

Out[23]: array([7., 8.])

In [24]: a * 0.5
```

```
Out[24]: array([[0. , 0.5, 1. , 1.5, 2. ],
                [2.5, 3. , 3.5, 4. , 4.5],
                [5. , 5.5, 6. , 6.5, 7. ],
                [7.5, 8. , 8.5, 9. , 9.5]])
```

```
In [25]: a ** 2
```

```
Out[25]: array([[  0.,   1.,   4.,   9.,  16.],
                [ 25.,  36.,  49.,  64.,  81.],
                [100., 121., 144., 169., 196.],
                [225., 256., 289., 324., 361.]])
```

```
In [26]: a + a
```

```
Out[26]: array([[ 0.,  2.,  4.,  6.,  8.],
                [10., 12., 14., 16., 18.],
                [20., 22., 24., 26., 28.],
                [30., 32., 34., 36., 38.]])
```

```
In [27]: def f(x):
             return x ** 3 + x ** 2 - 2 + np.sin(x)
```

```
In [28]: f(a)
```

```
Out[28]: array([[-2.00000000e+00,  8.41470985e-01,  1.09092974e+01,
                  3.41411200e+01,  7.72431975e+01],
                [ 1.47041076e+02,  2.49720585e+02,  3.90656987e+02,
                  5.74989358e+02,  8.08412118e+02],
                [ 1.09745598e+03,  1.44900001e+03,  1.86946343e+03,
                  2.36442017e+03,  2.93899061e+03],
                [ 3.59865029e+03,  4.34971210e+03,  5.19903860e+03,
                  6.15324901e+03,  7.21814988e+03]])
```

```
In [29]: for i in range(5):
             print(i)
```

```
0
1
2
3
4
```

```
In [30]: b = np.arange(0.0, 100.0, 1.0)
```

```
In [31]: for i in range(100):
             if b[i] == 50.0:
                 print("50.0 at index no. %d" % i)
```

```
50.0 at index no. 50
```

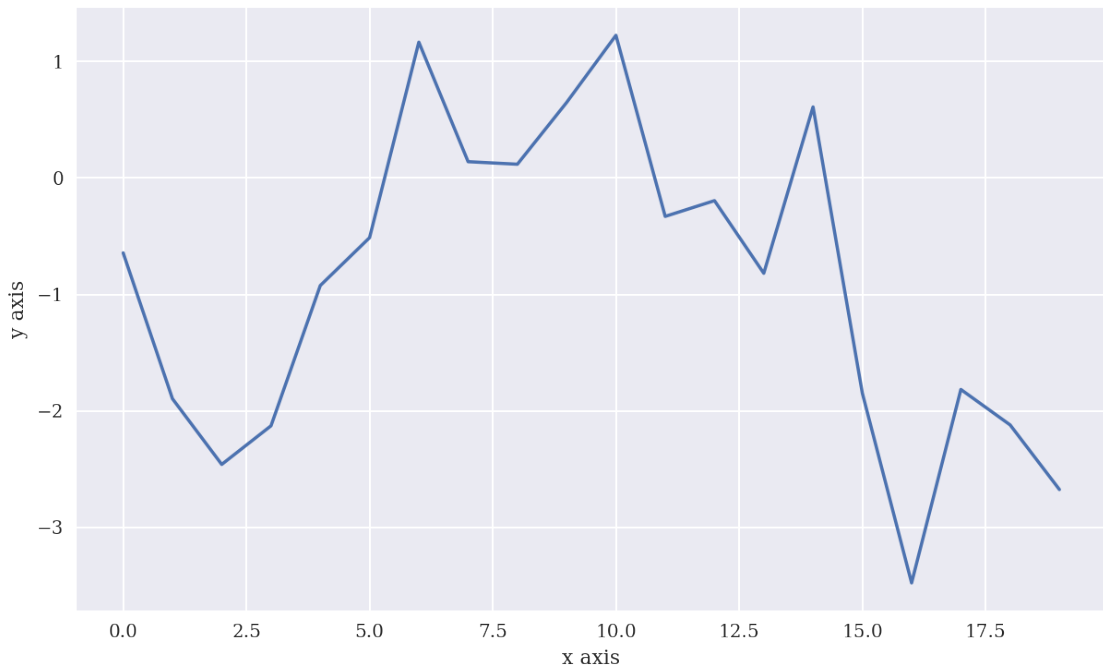
```
In [32]: print("%d divided by %d gives %6.3f" % (1000, 17, 1000./17))

1000 divided by 17 gives 58.824
```

0.1.3 Random Numbers

```
In [33]: import numpy as np
         np.random.seed(5000)

In [34]: b = np.random.standard_normal((4, 5))

In [35]: b

Out[35]: array([[-0.64371681, -1.25182043, -0.56391455,  0.3314386 ,  1.20390744],
                [ 0.41091404,  1.67824248, -1.02596417, -0.02176213,  0.53048021],
                [ 0.57600497, -1.55430075,  0.13509601, -0.6231574 ,  1.42761494],
                [-2.45615932, -1.62936212,  1.66033378, -0.30442536, -0.55443482]])

In [36]: np.sum(b)

Out[36]: -2.6749854009795335

In [37]: np.mean(b)

Out[37]: -0.1337492700489767

In [38]: np.std(b)

Out[38]: 1.1148394461082733
```

0.1.4 Plotting

```
In [39]: from pylab import plt
         plt.style.use('seaborn')
         import matplotlib as mpl
         mpl.rcParams['font.family'] = 'serif'
         %matplotlib inline

In [40]: plt.figure(figsize=(10, 6))
         plt.plot(np.cumsum(b))
         plt.xlabel('x axis')
         plt.ylabel('y axis')
         plt.savefig('../images/A_pyt/line_plot.pdf')

  Out[40]:
```


1
0
−1
−2
−3
y axis
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
x axis