Python Crash Course Review (Part III: Numpy, Scipy, Matplotlib)

1.a Numpy provides the array structure ...

```
from numpy import array, arange, matrix
#import numpy as np
```

```
a = np.array([0., 7, 2])                        # np.array([0,7,2])
A = np.array( [ [0, 1, 2], [3, 4, 5] ] )        # use nd-arrays for numerical mathemati
cs, not matrices
B = np.matrix( [ [0, 1, 2], [3, 4, 5] ] )
```

```
print(a,"\n", A, type(A), "\n", "B:\n", B, type(B))
```

> Make the code work.

... and some functions

> https://docs.scipy.org/doc/numpy/reference/index.html
> https://docs.scipy.org/doc/numpy/reference/generated/numpy.array.html

```
b = np.arange(3)
c = np.arange(3, dtype=float)
```

```
print( b,"\n", c)
```

```
from numpy import zeros, ones, identity, eye
#import numpy as np
```

```
a = zeros(3, dtype=int)
print(a)
```

```
A = ones((3,3))
print(A)
```

```
B = eye(4)
C = identity(4)
are_they_equal = np.all( (B-C) == 0 )
print(are_they_equal)
```

1.a) Create arrays (vectors)

```
import numpy as np
a = np.linspace(0,1,6)              # start, end, how many values
print('linspace:', type(a), a)
```

1.b) Numpy input and output (IO) from text files

```
np.savetxt("a.dat",a)
b = np.loadtxt("a.dat")
print(b)
```

> https://docs.scipy.org/doc/numpy/reference/routines.io.html

2. SciPy (pronounced “Sigh Pie”) is open-source software for mathematics, science, and engineering.

```
import numpy as np
import scipy as sp
from scipy import linalg
A = np.array([[2, -1],
              [-1, 1]])
b = np.array([0,1])

x = linalg.solve(A,b)
print(x)
```

> Check if the solution to the system of linear equations is correct.
> Find the scipy reference page for linalg.solve.

```
# print(linalg.cholesky(np.ones((3,3))))
```

> Make the code work and interpret the error message.
>
> Use nd-arrays and numpy/scipy functions and not matrices and matlib functions

> https://docs.scipy.org/doc/numpy/reference/routines.matlib.html
>
> What is the difference between numpy and scipy?
> https://www.scipy.org/scipylib/faq.html

3. Matplotlib and Pylab

```
import scipy as sp
from math import pi #, cos
import pylab as pl

t = np.array(np.linspace(0, 6*pi, 100) )
y = np.cos(t)

pl.plot(t, y)
pl.show()
```

```
# ploting 2d functions on regular meshes
x,y = np.meshgrid(np.linspace(-1,1,100), np.linspace(-1,1,100))
f = x**2 + y**2

pl.figure()
pl.pcolormesh(x, y, f)
pl.show()
```

Make sure that you feel comfortable with numpy, scipy, matplotlib by the end of week two [3h].

> Find information about certain functions in the numpy reference, the numpy user guide, and the scipy reference.
>
> Work through Chapter 1 of the scipy-lectures.
>
> Use the matplotlob gallery as resource for images and source code.
>
> Ask stackoverflow if you have a question.