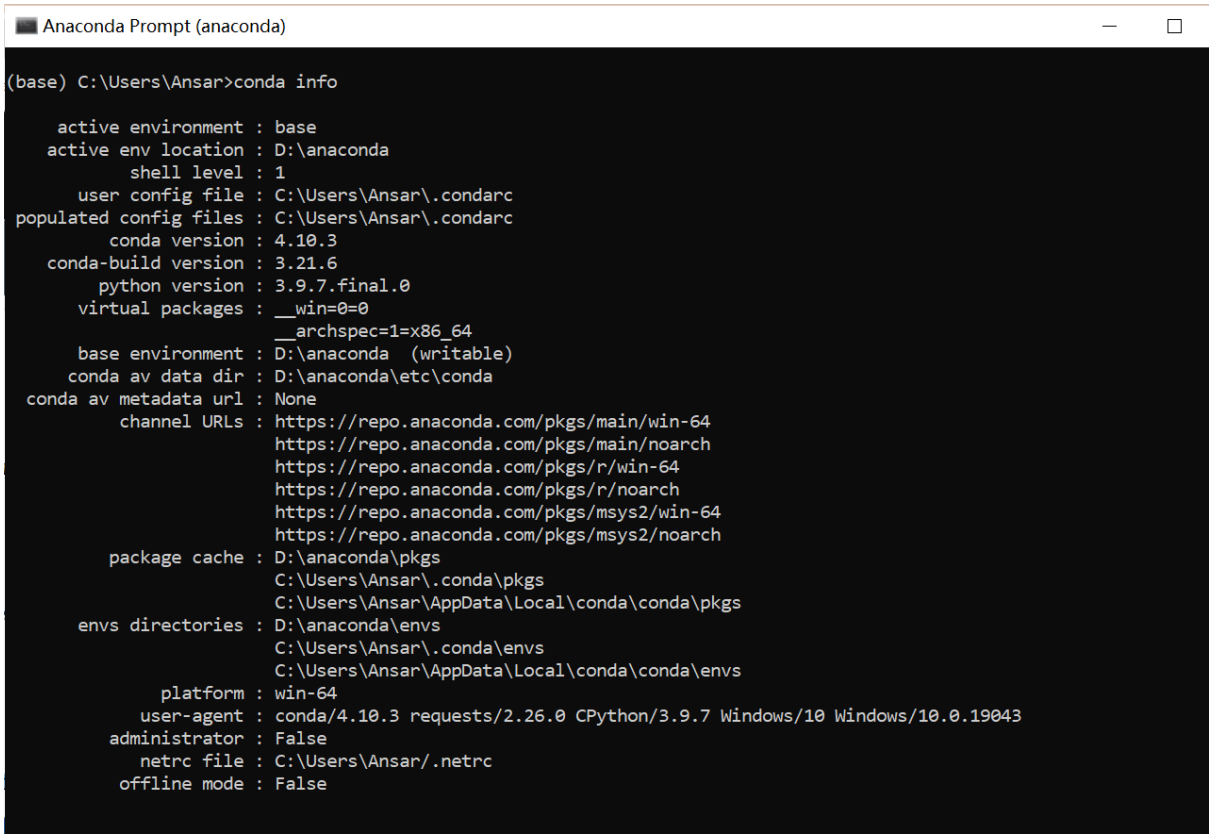
Assignment 0

1. Python Machine Learning Stack (Anaconda)



2. Transition from MATLAB to Python

TASK2

In [3]:

```
import numpy as np
import scipy.linalg as lina

a = np.array([1,2,3,4,5])
b = np.array([[1.,2.,3.],[4.,5.,6.]])
```

In [2]:

```
np.ndim(a)
```

Out[2]:

```
1
```

In [3]:

```
a.size
```

Out[3]:

```
5
```

In [4]:

```
a.shape
```

Out[4]:

```
(5,)
```

In [5]:

```
a.shape[0]
```

Out[5]:

```
5
```

In [7]:

```
b.shape
```

Out[7]:

```
(2, 3)
```

In [9]:

```
b.shape[0]
```

Out[9]:

```
2
```

In [10]:

```
b.shape[1]
```

Out[10]:

```
3
```

In [11]:

```
c = np.array([1.,2.,3.])
d = np.array([2.,3.,4.])
e = np.array([3.,4.,5.])
f = np.array([4.,5.,6.])
np.block([[c,d],[e,f]])
```

Out[11]:

```
array([[1., 2., 3., 2., 3., 4.],
       [3., 4., 5., 4., 5., 6.]])
```

In [12]:

```
c[-1]
```

Out[12]:

```
3.0
```

In [13]:

```
c[1:2]
```

Out[13]:

```
array([2.])
```

In [16]:

```
g = np.array([[1.,2.,3.],[4.,5.,6.]])
g[1,:]
```

Out[16]:

```
array([4., 5., 6.])
```

In [17]:

```
g[np.ix_([0],[0])]
```

Out[17]:

```
array([[1.]])
```

In [18]:

```
g[::2,:]
```

Out[18]:

```
array([[1., 2., 3.]])
```

In [19]:

```
g[::-1,:]
```

Out[19]:

```
array([[4., 5., 6.],
       [1., 2., 3.]])
```

In [20]:

```
g[np.r_[:len(g),0]]
```

Out[20]:

```
array([[1., 2., 3.],
       [4., 5., 6.],
       [1., 2., 3.]])
```

In [21]:

```
g.transpose()
```

Out[21]:

```
array([[1., 4.],
       [2., 5.],
       [3., 6.]])
```

In [23]:

```
g.conj().transpose()
```

Out[23]:

```
array([[1., 4.],
       [2., 5.],
       [3., 6.]])
```

In [27]:

```
g @ c
```

Out[27]:

```
array([14., 32.])
```

In [30]:

```
g * c
```

Out[30]:

```
array([[ 1.,  4.,  9.],
       [ 4., 10., 18.]])
```

In [31]:

```
g / c
```

Out[31]:

```
array([[1. , 1. , 1. ],
       [4. , 2.5, 2. ]])
```

In [32]:

```
g ** 3
```

Out[32]:

```
array([[  1.,   8.,  27.],
       [ 64., 125., 216.]])
```

In [33]:

```
(g > 4)
```

Out[33]:

```
array([[False, False, False],
       [False,  True,  True]])
```

In [34]:

```
np.nonzero(g > 4)
```

Out[34]:

```
(array([1, 1], dtype=int64), array([1, 2], dtype=int64))
```

In [35]:

```
g[g<0.5] = 0
```

In [38]:

```
g*(g>0.5)
```

Out[38]:

```
array([[1., 2., 3.],
       [4., 5., 6.]])
```

In [39]:

```
g[:] =3
```

In [40]:

```
g
```

Out[40]:

```
array([[3., 3., 3.],
```

```
       [3., 3., 3.]])
```

In [41]:

```
h = g.copy()
```

In [42]:

```
g,h
```

Out[42]:

```
(array([[3., 3., 3.],
        [3., 3., 3.]]),
 array([[3., 3., 3.],
        [3., 3., 3.]]))
```

In [45]:

```
h = g[1,:].copy()
h
```

Out[45]:

```
array([3., 3., 3.])
```

In [46]:

```
h = g.flatten()
h
```

Out[46]:

```
array([3., 3., 3., 3., 3., 3.])
```

In [47]:

```
np.arange(1.,11.)
```

Out[47]:

```
array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
```

In [48]:

```
np.r_[1.,11.]
```

Out[48]:

```
array([ 1., 11.])
```

In [49]:

```
np.r_[:9:10j]
```

Out[49]:

```
array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
```

In [50]:

```
np.arange(1.,11.)[:, np.newaxis]
```

Out[50]:

```
array([[ 1.],
       [ 2.],
       [ 3.],
       [ 4.],
       [ 5.],
       [ 6.],
       [ 7.],
       [ 8.],
       [ 9.],
```

```
       [10.]])
```

In [51]:

```
np.zeros((3,4))
```

Out[51]:

```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [52]:

```
np.zeros((3,4,5))
```

Out[52]:

```
array([[[0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.]],

       [[0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.]],

       [[0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.]]])
```

In [53]:

```
np.ones((3,4))
```

Out[53]:

```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [54]:

```
np.eye(3)
```

Out[54]:

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [55]:

```
np.diag(g)
```

Out[55]:

```
array([3., 3.])
```

In [56]:

```
np.diag(g,0)
```

Out[56]:

```
array([3., 3.])
```

In [57]:

```
np.random.rand(3,4)
```

Out[57]:

```
array([[0.36450828, 0.60869459, 0.49515357, 0.16384437],
       [0.21221788, 0.54337415, 0.99152795, 0.96108266],
       [0.3095499 , 0.12696432, 0.67535449, 0.2819018 ]])
```

In [58]:

```
np.random.random_sample((3,4,5))
```

Out[58]:

```
array([[[0.78059178, 0.21039292, 0.99396567, 0.58697086, 0.52794981],
        [0.95631458, 0.68105572, 0.48236609, 0.77398689, 0.82111461],
        [0.19272934, 0.16170707, 0.96697485, 0.04319777, 0.31501013],
        [0.80087342, 0.15822023, 0.02032751, 0.74752711, 0.64170189]],

       [[0.34640186, 0.58738873, 0.74743927, 0.14844871, 0.4652412 ],
        [0.54508788, 0.73585115, 0.80236618, 0.42746068, 0.80897041],
        [0.73890606, 0.21374191, 0.07917872, 0.47070606, 0.06925808],
        [0.53411049, 0.26542537, 0.05777422, 0.00314655, 0.81873183]],

       [[0.81916831, 0.44656944, 0.3526462 , 0.98071913, 0.98209887],
        [0.87957153, 0.34570035, 0.25052421, 0.54070308, 0.65888455],
        [0.97187326, 0.11542015, 0.3902913 , 0.2271047 , 0.42686728],
        [0.11687921, 0.77072906, 0.93356241, 0.70466193, 0.35765139]]])
```

In [59]:

```
np.linspace(1,3,4)
```

Out[59]:

```
array([1.        , 1.66666667, 2.33333333, 3.        ])
```

In [60]:

```
np.mgrid[0:9.,0:6.]
```

Out[60]:

```
array([[[0., 0., 0., 0., 0., 0.],
        [1., 1., 1., 1., 1., 1.],
        [2., 2., 2., 2., 2., 2.],
        [3., 3., 3., 3., 3., 3.],
        [4., 4., 4., 4., 4., 4.],
        [5., 5., 5., 5., 5., 5.],
        [6., 6., 6., 6., 6., 6.],
        [7., 7., 7., 7., 7., 7.],
        [8., 8., 8., 8., 8., 8.]],

       [[0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
```

```
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.],
        [0., 1., 2., 3., 4., 5.]]])
```

In [61]:

```
np.ogrid[0:9.,0:6.]
```

Out[61]:

```
[array([[0.],
        [1.],
        [2.],
        [3.],
        [4.],
        [5.],
        [6.],
        [7.],
        [8.]]),
 array([[0., 1., 2., 3., 4., 5.]])]
```

In [62]:

```
np.ix_(np.r_[0:9.],np.r_[0:6.])
```

Out[62]:

```
(array([[0.],
        [1.],
        [2.],
        [3.],
        [4.],
        [5.],
        [6.],
        [7.],
        [8.]]),
 array([[0., 1., 2., 3., 4., 5.]]))
```

In [63]:

```
np.meshgrid([1,2,4],[2,4,5])
```

Out[63]:

```
[array([[1, 2, 4],
        [1, 2, 4],
        [1, 2, 4]]),
 array([[2, 2, 2],
        [4, 4, 4],
        [5, 5, 5]])]
```

In [64]:

```
np.tile(g,(3,4))
```

Out[64]:

```
array([[3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.],
```

```
       [3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.],
       [3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.],
       [3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.],
       [3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.],
       [3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3., 3.]])
```

In [65]:

```
np.concatenate((c,d),0)
```

Out[65]:

```
array([1., 2., 3., 2., 3., 4.])
```

In [66]:

```
np.vstack((c,d))
```

Out[66]:

```
array([[1., 2., 3.],
       [2., 3., 4.]])
```

In [69]:

```
c.max()
```

Out[69]:

```
3.0
```

In [70]:

```
c.max(0)
```

Out[70]:

```
3.0
```

In [71]:

```
g.max(1)
```

Out[71]:

```
array([3., 3.])
```

In [72]:

```
np.maximum(c,d)
```

Out[72]:

```
array([2., 3., 4.])
```

In [74]:

```
np.sqrt(c@c)
```

Out[74]:

```
3.7416573867739413
```

In [75]:

```
np.linalg.norm(c)
```

Out[75]:

```
3.7416573867739413
```

In [76]:

```
np.logical_and(c,d)
```

Out[76]:

```
array([ True,  True,  True])
```

In [77]:

```
np.logical_or(c,d)
```

Out[77]:

```
array([ True,  True,  True])
```

In [78]:

```
i = np.array([[1,2,3],[4,5,6],[7,10,9]])
lina.eig(i)
```

Out[78]:

```
(array([16.82540654+0.j, -0.56655283+0.j, -1.2588537 +0.j]),
 array([[-0.22187905, -0.86720234, -0.7307057 ],
        [-0.49983984,  0.47988984, -0.16747068],
        [-0.83721552,  0.13291287,  0.66183288]]))
```

In [80]:

```
lina.inv(i)
```

Out[80]:

```
array([[-1.25      ,  1.        , -0.25      ],
       [ 0.5       , -1.        ,  0.5       ],
       [ 0.41666667,  0.33333333, -0.25      ]])
```

In [81]:

```
lina.pinv(i)
```

Out[81]:

```
array([[-1.25      ,  1.        , -0.25      ],
       [ 0.5       , -1.        ,  0.5       ],
       [ 0.41666667,  0.33333333, -0.25      ]])
```

In [82]:

```
lina.solve(i,c)
```

Out[82]:

```
array([ 1.26882631e-16, -0.00000000e+00,  3.33333333e-01])
```

In [83]:

```
lina.svd(i)
```

Out[83]:

```
(array([[-0.20157473, -0.72595537, -0.65753816],
        [-0.48945851, -0.50683053,  0.70961481],
        [-0.8484091 ,  0.46487806, -0.25316081]]),
 array([17.86008107,  1.32709403,  0.50628624]),
 array([[-0.45342865, -0.63462942, -0.62581783],
        [ 0.3774178 ,  0.4993747 , -0.77985942],
        [ 0.80743932, -0.58980539,  0.01308965]]))
```

In [84]:

```
lina.qr(i)
```

Out[84]:

```
(array([[-0.12309149,  0.69631062, -0.70710678],
        [-0.49236596, -0.66149509, -0.56568542],
        [-0.86164044,  0.27852425,  0.42426407]]),
 array([[ -8.1240384 , -11.32441717, -11.07823419],
        [  0.        ,   0.87038828,   0.62667956],
```

```
        [  0.        ,   0.        ,  -1.69705627]]))
```

In [85]:

```
lina.lu(i)
```

Out[85]:

```
(array([[0., 0., 1.],
        [0., 1., 0.],
        [1., 0., 0.]]),
 array([[ 1.        ,  0.        ,  0.        ],
        [ 0.57142857,  1.        ,  0.        ],
        [ 0.14285714, -0.8       ,  1.        ]]),
 array([[ 7.        , 10.        ,  9.        ],
        [ 0.        , -0.71428571,  0.85714286],
        [ 0.        ,  0.        ,  2.4       ]]))
```

In [86]:

```
np.fft.fft(i)
```

Out[86]:

```
array([[ 6. +0.j       , -1.5+0.8660254j, -1.5-0.8660254j],
       [15. +0.j       , -1.5+0.8660254j, -1.5-0.8660254j],
       [26. +0.j       , -2.5-0.8660254j, -2.5+0.8660254j]])
```

In [87]:

```
np.fft.ifft(i)
```

Out[87]:

```
array([[ 2.         +0.j         , -0.5        -0.28867513j,
        -0.5        +0.28867513j],
       [ 5.         +0.j         , -0.5        -0.28867513j,
        -0.5        +0.28867513j],
       [ 8.66666667+0.j         , -0.83333333+0.28867513j,
        -0.83333333-0.28867513j]])
```

In [88]:

```
np.sort(i)
```

Out[88]:

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  9, 10]])
```

In [89]:

```
j = np.array([[1,2,3],[4,5,6],[7,8,9]])
lina.lstsq(i,j)
```

Out[89]:

```
(array([[ 1.00000000e+00,  5.00000000e-01,  2.24258837e-16],
        [ 1.05692096e-15,  4.99600361e-16, -4.44089210e-16],
        [ 1.22191171e-16,  5.00000000e-01,  1.00000000e+00]]),
 array([], dtype=float64),
 3,
 array([17.86008107,  1.32709403,  0.50628624]))
```

In [90]:

```
np.unique(i)
```

Out[90]:

```
array([ 1,  2,  3,  4,  5,  6,  7,  9, 10])
```

In [91]:

```
i.squeeze()
```

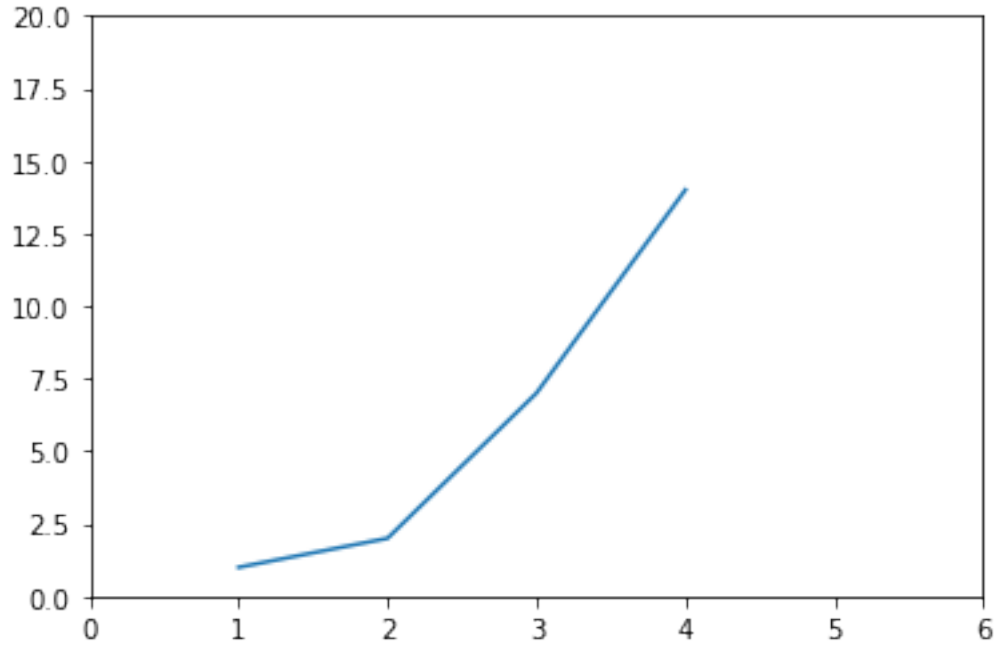
Out[91]:

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7, 10,  9]])
```

TASK3

In [1]:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,2,7,14])
plt.axis([0, 6, 0, 20])
plt.show()
```

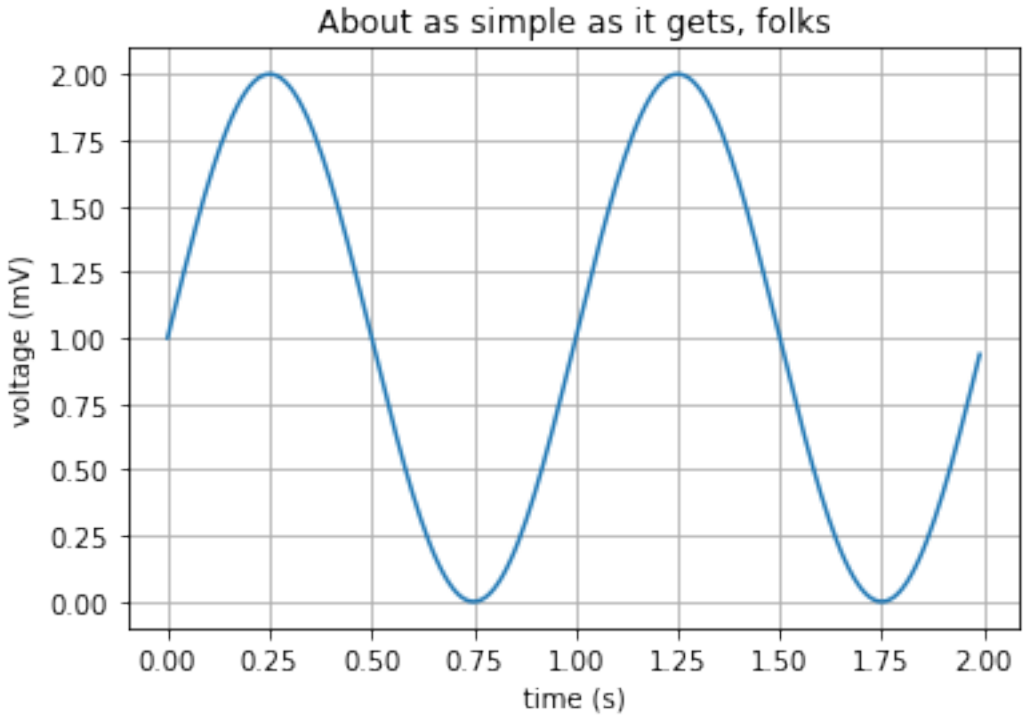


```
t = np.arange(0.0, 2.0, 0.01)
s = 1 + np.sin(2 * np.pi * t)

fig, ax = plt.subplots()
ax.plot(t, s)

ax.set(xlabel='time (s)', ylabel='voltage (mV)',
       title='About as simple as it gets, folks')
ax.grid()

fig.savefig("test.png")
plt.show()
```



3. Version Control System (GitHub)

GitHub Account: ansarrice

4. Integrated Development Environment (PyCharm)

https://github.com/ansarrice/COMP576/tree/master