COMP576 ASSIGN0 SOLUTIONS

Syed Saqib Habeeb

September 2024

1 Task 1

Input command:

```
$ conda info
```

Output is

```
         active environment : base
    active env location : /home/saqib/anaconda3
            shell level : 1
       user config file : /home/saqib/.condarc
 populated config files :
          conda version : 4.12.0
    conda-build version : 3.21.8
         python version : 3.9.12.final.0
       virtual packages : __linux=6.8.0=0
                          __glibc=2.35=0
                          __unix=0=0
                          __archspec=1=x86_64
       base environment : /home/saqib/anaconda3  (writable)
      conda av data dir : /home/saqib/anaconda3/etc/conda
  conda av metadata url : None
           channel URLs : https://repo.anaconda.com/pkgs/main/linux-64
                          https://repo.anaconda.com/pkgs/main/noarch
                          https://repo.anaconda.com/pkgs/r/linux-64
                          https://repo.anaconda.com/pkgs/r/noarch
          package cache : /home/saqib/anaconda3/pkgs
                          /home/saqib/.conda/pkgs
       envs directories : /home/saqib/anaconda3/envs
                          /home/saqib/.conda/envs
               platform : linux-64
             user-agent : conda/4.12.0 requests/2.27.1 CPython/3.9.12 Linux/6.8.0-40
                            -generic ubuntu/22.04.4 glibc/2.35
                UID:GID : 1000:1000
             netrc file : None
           offline mode : False
```

2 Task 2

Import the required libraries and run the codes below to get the output

```
import numpy as np
import random
import math

arr = np.random.rand(10)
print(arr)

array([0.92106603, 0.86627046, 0.27751017, 0.13211007, 0.76317295,
   0.39007386, 0.96359257, 0.03097473, 0.48309004, 0.26189913])

arr.ndim

1

arr.size

10

arr.shape

(10,)

a=np.array([[1.,2.,3.],[4.,5.,6.],[7.,8.,9.]]) //define a 2d array
a.shape[2-1] //shape of second dimension

3

a

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

arr[-1]

0.26189912785354075

b=np.block([[1,2],[3,4]])
b

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

a[1,2]

6.0
```

```
a[1,:]
```

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

```
a[:2,:]
```

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

```
a[-2:,]
```

```
array([[4., 5., 6.],
   [7., 8., 9.]])
```

```
a[0:1,1:2]
```

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

```
a[np.ix_([1, 2], [0, 2])]
```

```
array([[4., 6.],
   [7., 9.]])
```

```
a[::2, :]
```

```
array([[1., 2., 3.],
   [7., 8., 9.]])
```

```
a[::-1,:]
```

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

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

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

```
a.T
```

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

```
a.conj().transpose()
```

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

```
b=a.T
```

```
a@b
```

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

```
a[:2,:]
```

```
array([[ 14.,  32.,  50.],
   [ 32.,  77., 122.],
   [ 50., 122., 194.]])
```

```
a*b
```

```
array([[ 1.,  8., 21.],
   [ 8., 25., 48.],
   [21., 48., 81.]])
```

```
a/b
```

```
array([[1.        , 0.5       , 0.42857143],
   [2.        , 1.        , 0.75      ],
   [2.33333333, 1.33333333, 1.        ]])
```

```
a**3
```

```
array([[  1.,   8.,  27.],
   [ 64., 125., 216.],
   [343., 512., 729.]])
```

```
a>=5
```

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

```
np.nonzero(a > 5)
```

```
(array([1, 2, 2, 2]), array([2, 0, 1, 2]))
```

```
v=np.array([1,2,3])
a[:,np.nonzero(v > 1)[0]]
```

```
array([[2., 3.],
   [5., 6.],
   [8., 9.]])
```

```
a[:, v.T > 2]
```

```
array([[3.],
   [6.],
   [9.]])
```

```
a[a < 5]=0
a
```

```
array([[0., 0., 0.],
   [0., 5., 6.],
   [7., 8., 9.]])
```

```
a=np.array([[1.,2.,3.],[4.,5.,6.],[7.,8.,9.]])
a1 = a * (a > 5)
a1
```

```
array([[0., 0., 0.],
   [0., 0., 6.],
   [7., 8., 9.]])
```

```
a1[:] = 3
a1
```

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

```
i = a.copy()
i
```

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

```
y = a[1, :].copy()
y
```

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

```
i=a.flatten()
i
```

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

```
i=a.T
i=i.flatten()
i
```

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

np.arange(1., 11.)

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

np.arange(10.)

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

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

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

j=np.zeros((3, 4))
j

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

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

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.]]])

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

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

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

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

```
np.diag(a)
```

```
array([1., 5., 9.])
```

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

```
array([[1, 0, 0],
   [0, 2, 0],
   [0, 0, 3]])
```

```
from numpy.random import default_rng
rng = default_rng(42)
rng.random((3, 4))
```

```
array([[0.77395605, 0.43887844, 0.85859792, 0.69736803],
   [0.09417735, 0.97562235, 0.7611397 , 0.78606431],
   [0.12811363, 0.45038594, 0.37079802, 0.92676499]])
```

```
np.linspace(1,4,5)
```

```
array([1.  , 1.75, 2.5 , 3.25, 4.  ])
```

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

```
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.]]])

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

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

   np.ix_([1,2,4],[2,4,5])

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

   np.tile(a, (2, 2))

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

   np.concatenate((a,b),1)

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

   np.concatenate((a,b))

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

   a.max()

   a.max(1)
```

```
    a.max(1)

    array([3., 6., 9.])

    np.maximum(a, b)

    array([[1., 2., 7.],
       [4., 5., 8.],
       [7., 8., 9.]])

    np.linalg.norm(v)

    3.7416573867739413

    np.logical_and(a,b)

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

    np.logical_or(a,b)

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

    c = 1 & 2
    c

    0

    c= 1 | 2
    c

    3

    c=np.array([[1,2,3],[0,1,0],[1,1,1]])
    c


array([[1, 2, 3],
       [0, 1, 0],
       [1, 1, 1]]

    np.linalg.inv(c)

    array([[-0.5, -0.5,  1.5],
       [ 0. ,  1. ,  0. ],
       [ 0.5, -0.5, -0.5]])
```

```
np.linalg.pinv(c)
```

```
array([[-5.00000000e-01, -5.00000000e-01,  1.50000000e+00],
   [ 2.83091776e-16,  1.00000000e+00, -1.78386441e-16],
   [ 5.00000000e-01, -5.00000000e-01, -5.00000000e-01]])
```

```
np.linalg.matrix_rank(a)
```

```
2
```

```
np.linalg.matrix_rank(c)
```

```
3
```

```
np.linalg.lstsq(a, b)
```

```
(array([[ 0.        ,  1.        , -3.05555556],
    [ 0.        ,  0.33333333,  0.11111111],
    [ 0.        , -0.33333333,  3.27777778]]),
 array([], dtype=float64),
 2,
 array([1.68481034e+01, 1.06836951e+00, 3.33475287e-16]))
```

```
U, S, Vh = np.linalg.svd(a)
V = Vh.T
print(U,S,V)
```

```
[[-0.21483724  0.88723069  0.40824829]
 [-0.52058739  0.24964395 -0.81649658]
 [-0.82633754 -0.38794278  0.40824829]] [1.68481034e+01 1.06836951e+00 3.33475287e-16]
 [-0.57236779 -0.07568647  0.81649658]
 [-0.66506441  0.62531805 -0.40824829]]
```

```
c=np.array([[2,-1],[-1,2]])
np.linalg.cholesky(c)
```

```
array([[ 1.41421356,  0.        ],
   [-0.70710678,  1.22474487]])
```

```
D,V = np.linalg.eig(a)
print(D,V)
```

```
[ 1.61168440e+01 -1.11684397e+00 -9.75918483e-16] [[-0.23197069 -0.78583024  0.40824829]
 [-0.52532209 -0.08675134 -0.81649658]
 [-0.8186735   0.61232756  0.40824829]]
```

```
Q,R = np.linalg.qr(a)
print(Q,R)
```

```
[[-0.12309149  0.90453403  0.40824829]
 [-0.49236596  0.30151134 -0.81649658]
 [-0.86164044 -0.30151134  0.40824829]] [[-8.12403840e+00 -9.60113630e+00 -1.10782342e+0
 [ 0.00000000e+00  9.04534034e-01  1.80906807e+00]
 [ 0.00000000e+00  0.00000000e+00 -7.58790979e-16]]
```

```
np.fft.fft(a)
```

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

```
np.fft.ifft(a)
```

```
np.fft.ifft(a)
```

```
b=np.array([[1,7,5],[3,9,6],[2,8,4]])
np.sort(b,axis=0)
```

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

```
np.sort(b,axis=1)
```

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

```
I = np.argsort(c[:, 0]); c = a[I,:]
c
```

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

```
a1=np.array([[1,2,3],[2,4,6]])
b1=np.array([[2,4,6],[1,2,3]])
x = np.linalg.lstsq(a1,b1 )
x
```

```
(array([[0.05714286, 0.11428571, 0.17142857],
    [0.11428571, 0.22857143, 0.34285714],
    [0.17142857, 0.34285714, 0.51428571]]),
 array([], dtype=float64),
 1,
 array([8.36660027e+00, 4.77448860e-16]))
```

```
x=np.array([1,1,1,2,2,3,3,2,1,3,4,5,9,85,6,7,5,52,8,6,2,10,100,0,88,0,55,7,3,33,66,99,69
```

```
np.unique(x)
```

```
array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  33,  52,
        55,  66,  69,  85,  88,  90,  99, 100])
```

```
print(b.squeeze())
print(b.size)
```

```
[[1 7 5]
 [3 9 6]
 [2 8 4]]
9
```

3 Task 3

The output of the code

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,2,7,14])
plt.axis([0, 6, 0, 20])
plt.savefig('./figs/task3.png')
plt.show()
```

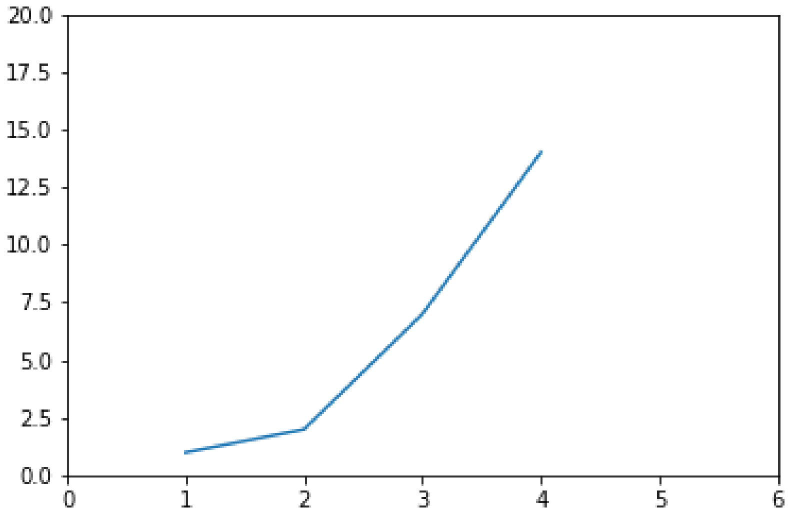

Fig. 3: Plot obtained in task 3

4 Task 4

The output of the code

```
x= [1,2,3,4,5,6]
y= [6,5,4,3,2,1]
z= [2,4,6,8,10,12]
plt.scatter(x,z,label = 'x')
plt.plot(y,z,label='y',color='red')
plt.xlabel("X Axis")
plt.ylabel("Y Axis")
plt.title("task 4 plot")
plt.legend()
plt.savefig('./figs/task4.png')
plt.show()
```

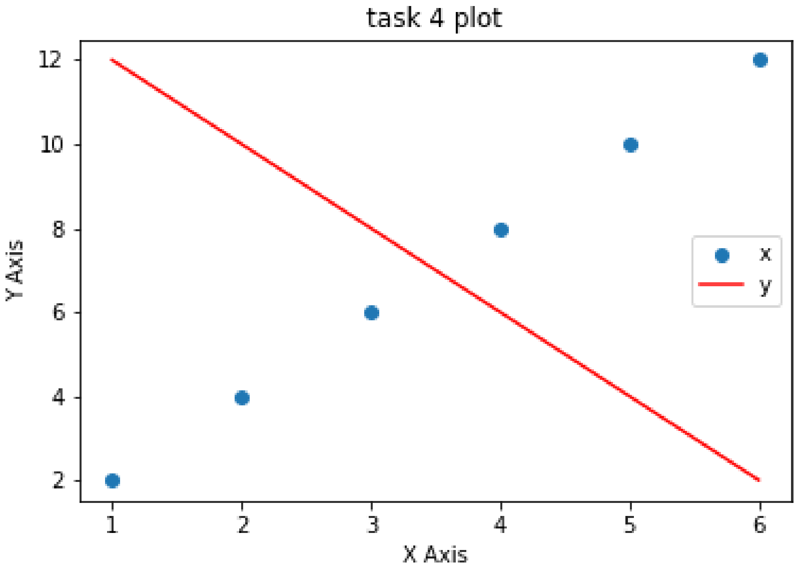

Fig. 4: Plot obtained in task 4

5 Task 5

Here is the link to my github profile.
https://github.com/syedsaqibhabeeb

6 Task 6

Here is the link to my project on github.
https://github.com/syedsaqibhabeeb/elec576_dl