COMP 576 Assignment 0

Travis Benedict trb6

Task 1

active environment : None user config file : C:\Users\travi.condarc populated config files : C:\Users\travi.condarc conda version : 4.7.10 conda-build version : 3.17.8 python version : 3.6.8.final.0 virtual packages : base environment : C:\Users\travi\Anaconda3 (writable) channel URLs : https://repo.anaconda.com/pkgs/main/win-64 (https://repo.anaconda.com/pkgs/main/win-64) https://repo.anaconda.com/pkgs/main/noarch (https://repo.anaconda.com/pkgs/main/noarch) https://repo.anaconda.com/pkgs/r/win-64 (https://repo.anaconda.com/pkgs/r/win-64) https://repo.anaconda.com/pkgs/r/noarch (https://repo.anaconda.com/pkgs/r/noarch) https://repo.anaconda.com/pkgs/msys2/win-64 (https://repo.anaconda.com/pkgs/msys2/win-64) https://repo.anaconda.com/pkgs/msys2/noarch (https://repo.anaconda.com/pkgs/msys2/noarch) package cache : C:\Users\travi\Anaconda3\pkgs C:\Users\travi.conda\pkgs C:\Users\travi\AppData\Local\conda\conda\pkgs envs directories : C:\Users\travi\Anaconda3\envs C:\Users\travi.conda\envs C:\Users\travi\AppData\Local\conda\conda\envs platform : win-64 user-agent : conda/4.7.10 requests/2.21.0 CPython/3.6.8 Windows/10 Windows/10.0.18362 administrator : False netrc file : None offline mode : False

Task 2

```
In [2]: from numpy import *
        import scipy.linalg

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

```
In [3]: ndim(a)

Out[3]: 2
```

```
In [4]: size(a)

Out[4]: 6
```

```
In [5]: a.shape

Out[5]: (2, 3)
```

```
In [6]: a.shape[2-1]

Out[6]: 3
```

```
In [7]: array([[1.,2.,3.], [4.,5.,6.]])

Out[7]: array([[1., 2., 3.],
               [4., 5., 6.]])
```

```
In [8]: block([[a,b], [a,b]])

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

```
In [9]: a[-1]

Out[9]: array([4, 5, 6])
```

```
In [10]: a[1,2]

Out[10]: 6
```

```
In [11]: a[1:]

Out[11]: array([[4, 5, 6]])
```

```
In [12]: a[0:1]

Out[12]: array([[1, 2, 3]])
```

```
In [13]: a[-5:]

Out[13]: array([[1, 2, 3],
                [4, 5, 6]])
```

```
In [14]: a[0:3][:,4:9]

Out[14]: array([], shape=(2, 0), dtype=int32)
```

```
In [15]: a[ix_([1],[0,2])]

Out[15]: array([[4, 6]])
```

```
In [16]: a[ 2:21:2,:]

Out[16]: array([], shape=(0, 3), dtype=int32)
```

```
In [17]: a[ ::2,:]

Out[17]: array([[1, 2, 3]])
```

```
In [18]: a[ ::-1,:]

Out[18]: array([[4, 5, 6],
                [1, 2, 3]])

In [19]: a[r_[:len(a),0]]

Out[19]: array([[1, 2, 3],
                [4, 5, 6],
                [1, 2, 3]])

In [20]: a.T

Out[20]: array([[1, 4],
                [2, 5],
                [3, 6]])

In [22]: a.conj().T

Out[22]: array([[1, 4],
                [2, 5],
                [3, 6]])

In [24]: a.T @ b

Out[24]: array([[32, 37, 42],
                [43, 50, 57],
                [54, 63, 72]])

In [25]: a * b

Out[25]: array([[ 4, 10, 18],
                [28, 40, 54]])

In [28]: c = a / b
         c

Out[28]: array([[0.25      , 0.4       , 0.5       ],
                [0.57142857, 0.625     , 0.66666667]])

In [29]: a**3

Out[29]: array([[  1,   8,  27],
                [ 64, 125, 216]], dtype=int32)

In [30]: (c>0.5)

Out[30]: array([[False, False, False],
                [ True,  True,  True]])

In [31]: nonzero(c>0.5)

Out[31]: (array([1, 1, 1], dtype=int64), array([0, 1, 2], dtype=int64))
```

```
In [32]: a[:,nonzero(c>0.5)[0]]

Out[32]: array([[2, 2, 2],
                [5, 5, 5]])

In [46]: a[:,a[1,:].T>0.5]

Out[46]: array([[1, 2, 3],
                [4, 5, 6]])

In [50]: a[a<5]=0
         a

Out[50]: array([[0, 0, 0],
                [0, 5, 6]])

In [51]: a = array([[1,2,3],[4,5,6]])

In [53]: a * (a>2)

Out[53]: array([[0, 0, 3],
                [4, 5, 6]])

In [55]: a[:] = 3
         a

Out[55]: array([[3, 3, 3],
                [3, 3, 3]])

In [56]: a = array([[1,2,3],[4,5,6]])

In [57]: y = a.copy()
         y

Out[57]: array([[1, 2, 3],
                [4, 5, 6]])

In [58]: y = a[1,:].copy()
         y

Out[58]: array([4, 5, 6])

In [60]: y = a.flatten()
         y

Out[60]: array([1, 2, 3, 4, 5, 6])

In [61]: arange(1.,11.)

Out[61]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
```

```
In [62]: arange(1.,11.)[:, newaxis]

Out[62]: array([[ 1.],
                [ 2.],
                [ 3.],
                [ 4.],
                [ 5.],
                [ 6.],
                [ 7.],
                [ 8.],
                [ 9.],
                [10.]])
```

```
In [63]: zeros((3,4))

Out[63]: array([[0., 0., 0., 0.],
                [0., 0., 0., 0.],
                [0., 0., 0., 0.]])
```

```
In [65]: zeros((3,4,5))

Out[65]: 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 [67]: ones((3,4))

Out[67]: array([[1., 1., 1., 1.],
                [1., 1., 1., 1.],
                [1., 1., 1., 1.]])
```

```
In [68]: eye(3)

Out[68]: array([[1., 0., 0.],
                [0., 1., 0.],
                [0., 0., 1.]])
```

```
In [69]: diag(a)

Out[69]: array([1, 5])
```

In [71]:
```
diag(a,1)
```

Out[71]:
```
array([2, 6])
```

In [72]:
```
random.rand(3,4)
```

Out[72]:
```
array([[0.45229196, 0.47181368, 0.92197959, 0.12101331],
       [0.2634483 , 0.52354106, 0.14775323, 0.77087795],
       [0.63667355, 0.67937637, 0.82300262, 0.1723521 ]])
```

In [73]:
```
linspace(1,3,4)
```

Out[73]:
```
array([1.        , 1.66666667, 2.33333333, 3.        ])
```

In [74]:
```
mgrid[0:9.,0:6.]
```

Out[74]:
```
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 [75]:
```
ogrid[0:9.,0:6.]
```

Out[75]:
```
[array([[0.],
        [1.],
        [2.],
        [3.],
        [4.],
        [5.],
        [6.],
        [7.],
        [8.]]), array([[0., 1., 2., 3., 4., 5.]])]
```

```
In [76]: tile(a, (3, 2))

Out[76]: array([[1, 2, 3, 1, 2, 3],
                [4, 5, 6, 4, 5, 6],
                [1, 2, 3, 1, 2, 3],
                [4, 5, 6, 4, 5, 6],
                [1, 2, 3, 1, 2, 3],
                [4, 5, 6, 4, 5, 6]])

In [77]: hstack((a,b))

Out[77]: array([[1, 2, 3, 4, 5, 6],
                [4, 5, 6, 7, 8, 9]])

In [78]: vstack((a,b))

Out[78]: array([[1, 2, 3],
                [4, 5, 6],
                [4, 5, 6],
                [7, 8, 9]])

In [79]: a.max()

Out[79]: 6

In [80]: a.max(0)

Out[80]: array([4, 5, 6])

In [81]: a.max(1)

Out[81]: array([3, 6])

In [82]: maximum(a, b)

Out[82]: array([[4, 5, 6],
                [7, 8, 9]])

In [83]: v = array([1,2,3,4,5])

In [85]: linalg.norm(v)

Out[85]: 7.416198487095663

In [86]: logical_and(a,b)

Out[86]: array([[ True,  True,  True],
                [ True,  True,  True]])

In [88]: logical_or(a,b)

Out[88]: array([[ True,  True,  True],
                [ True,  True,  True]])
```

```
In [89]: a & b

Out[89]: array([[0, 0, 2],
                [4, 0, 0]], dtype=int32)

In [90]: a | b

Out[90]: array([[ 5,  7,  7],
                [ 7, 13, 15]], dtype=int32)

In [106]: square = abs(random.rand(3,3))

In [107]: linalg.inv(square)

Out[107]: array([[-0.05104404,  4.45217034, -2.07471012],
                 [-0.04307021, -4.72659258,  3.33665002],
                 [ 1.57892669, -2.69237727,  0.86330558]])

In [108]: linalg.pinv(a)

Out[108]: array([[-0.94444444,  0.44444444],
                 [-0.11111111,  0.11111111],
                 [ 0.72222222, -0.22222222]])

In [98]: linalg.matrix_rank(a)

Out[98]: 2

In [110]: linalg.solve(square, b.T)

Out[110]: array([[ 9.60841482, 16.58766336],
                 [-3.7853436 , -8.08438191],
                 [-1.96634612, -2.71678114]])

In [109]: linalg.svd(square)

Out[109]: (array([[-0.4502919 ,  0.89015271,  0.06975208],
                  [-0.46407318, -0.16658337, -0.86999199],
                  [-0.7628062 , -0.42412041,  0.48810714]]),
           array([1.72269349, 0.61244861, 0.12395005]),
           array([[-0.79338026, -0.57251704, -0.20681394],
                  [ 0.05685505, -0.40795736,  0.911229  ],
                  [-0.6060654 ,  0.71119269,  0.3562158 ]]))

In [116]: linalg.eig(square)

Out[116]: (array([ 1.51524251+0.j        , -0.19739496+0.21758096j,
                  -0.19739496-0.21758096j]),
           array([[ 0.61523002+0.j        , -0.51132592-0.24473726j,
                   -0.51132592+0.24473726j],
                  [ 0.41157985+0.j        ,  0.39492737+0.41245666j,
                    0.39492737-0.41245666j],
                  [ 0.6723794 +0.j        ,  0.59376877+0.j        ,
                    0.59376877-0.j        ]]))
```

```
In [118]: scipy.linalg.eig(square, square)

Out[118]: (array([1.+0.j, 1.+0.j, 1.+0.j]),
           array([[-1.        ,  0.31622777, -0.72760688],
                  [-0.        ,  0.47128383,  0.18001071],
                  [-0.        ,  0.8233417 , -0.66195497]]))
```

```
In [119]: scipy.linalg.qr(square)

Out[119]: (array([[-0.468278  ,  0.73688295,  0.48756459],
                  [-0.50671819,  0.22808585, -0.83139252],
                  [-0.72384555, -0.63638068,  0.26658441]]),
           array([[-1.36925677, -0.97327629, -0.3673949 ],
                  [ 0.        , -0.30929243,  0.45826172],
                  [ 0.        ,  0.        ,  0.30879495]]))
```

```
In [120]: scipy.linalg.lu(square)

Out[120]: (array([[0., 1., 0.],
                  [0., 0., 1.],
                  [1., 0., 0.]]), array([[1.        , 0.        , 0.        ],
                  [0.64693083, 1.        , 0.        ],
                  [0.70003634, 0.58644333, 1.        ]]), array([[ 0.99113042,  0.90132
          944,  0.05662818],
                  [ 0.        , -0.35524624,  0.62365117],
                  [ 0.        ,  0.        , -0.37141897]]))
```

```
In [124]: sort(square)

Out[124]: array([[0.22785156, 0.64119283, 0.66028568],
                 [0.03395888, 0.42263158, 0.69382731],
                 [0.05662818, 0.90132944, 0.99113042]])
```

```
In [126]: argsort(square[:,2])

Out[126]: array([1, 2, 0], dtype=int64)
```

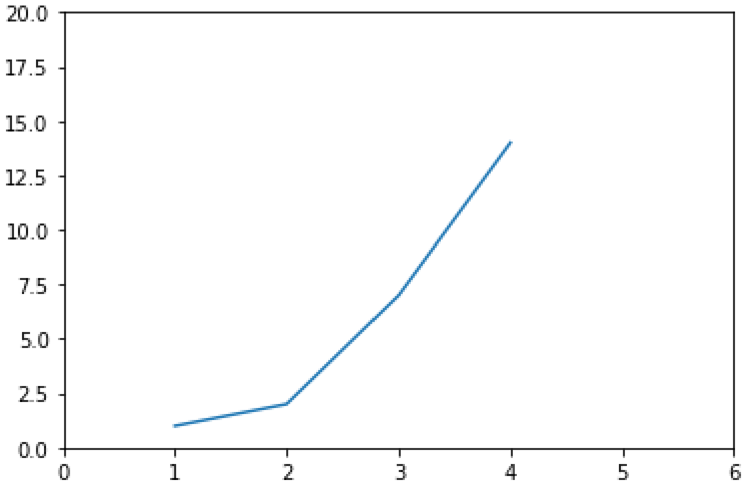
```
In [130]: linalg.lstsq(square, a[1,:])

          C:\Users\travi\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureWar
          ning: `rcond` parameter will change to the default of machine precision times
          ``max(M, N)`` where M and N are the input matrix dimensions.
          To use the future default and silence this warning we advise to pass `rcond=N
          one`, to keep using the old, explicitly pass `rcond=-1`.
            """Entry point for launching an IPython kernel.

Out[130]: (array([ 9.60841482, -3.7853436 , -1.96634612]),
           array([], dtype=float64),
           3,
           array([1.72269349, 0.61244861, 0.12395005]))
```

```
In [131]: unique(a)

Out[131]: array([1, 2, 3, 4, 5, 6])
```

```
In [132]: a.squeeze()

Out[132]: array([[1, 2, 3],
                 [4, 5, 6]])
```

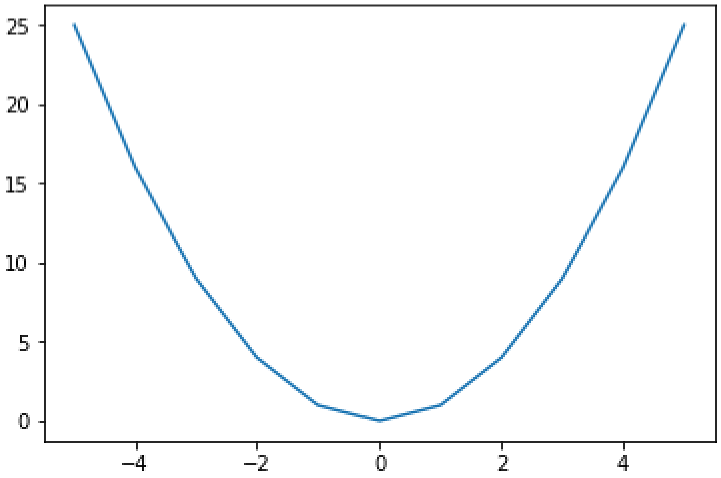
Task 3

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



Task 4

```
In [4]: xs = [x for x in range(-5,6)]
        x2 = [x**2 for x in xs]

        plt.plot(xs, x2)
        plt.show()
```



Task 5

https://github.com/travisbenedict (https://github.com/travisbenedict)

Task 6

https://github.com/travisbenedict/COMP-576-HW-0 (https://github.com/travisbenedict/COMP-576-HW-0)