Mason Weiss
ELEC 576 - Introduction to Deep Learning
Prof. Ankit Patel, Rice University
Due: September 18, 2024

Assignment 0

Task 1:

```
$ conda info
```

```
     active environment : base
    active env location : /Users/mason/opt/anaconda3
            shell level : 1
       user config file : /Users/mason/.condarc
 populated config files : /Users/mason/.condarc
          conda version : 24.7.1
    conda-build version : 24.7.1
         python version : 3.9.19.final.0
                 solver : libmamba (default)
       virtual packages : __archspec=1=cannonlake
                          __conda=24.7.1=0
                          __osx=10.16=0
                          __unix=0=0
       base environment : /Users/mason/opt/anaconda3  (writable)
      conda av data dir : /Users/mason/opt/anaconda3/etc/conda
  conda av metadata url : None
          channel URLs : https://repo.anaconda.com/pkgs/main/osx-64
                          https://repo.anaconda.com/pkgs/main/noarch
                          https://repo.anaconda.com/pkgs/r/osx-64
                          https://repo.anaconda.com/pkgs/r/noarch
         package cache : /Users/mason/opt/anaconda3/pkgs
                          /Users/mason/.conda/pkgs
       envs directories : /Users/mason/opt/anaconda3/envs
                          /Users/mason/.conda/envs
                platform : osx-64
              user-agent : conda/24.7.1 requests/2.32.3 CPython/3.9.19
              ↪  Darwin/22.3.0 OSX/10.16 solver/libmamba
              ↪  conda-libmamba-solver/24.7.0 libmambapy/1.5.8 aau/0.4.4 c/. s/.
              ↪  e/.
                 UID:GID : 501:20
              netrc file : None
            offline mode : False
```

Task 2: Run all of Python commands in the table "Linear Algebra Equivalents" in Numpy for MATLAB Users.

```
(base) mason@masons-mbp ~ % ipython
Python 3.9.19 (main, May  6 2024, 14:46:57)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.15.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import numpy as np

In [2]: import scipy.linalg

In [3]: a =
   ↪  np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])

In [4]: np.ndim(a)
Out[4]: 2

In [5]: a.ndim
Out[5]: 2

In [6]: np.size(a)
Out[6]: 20

In [7]: a.size
Out[7]: 20

In [8]: np.shape(a)
Out[8]: (5, 4)

In [9]: a.shape
Out[9]: (5, 4)

In [10]: n = 2

In [11]: a.shape[n-1]
Out[11]: 4

In [12]: np.array([[1., 2., 3.], [4., 5., 6.]])
Out[12]:
array([[1., 2., 3.],
       [4., 5., 6.]])

In [13]: b = a*-1

In [14]: c = a*2

In [15]: d = a*-3
```

```
In [16]: np.block([[a, b], [c, d]])
Out[16]:
array([[  1,   2,   3,   4,  -1,  -2,  -3,  -4],
       [  5,   6,   7,   8,  -5,  -6,  -7,  -8],
       [  9,  10,  11,  12,  -9, -10, -11, -12],
       [ 13,  14,  15,  16, -13, -14, -15, -16],
       [ 17,  18,  19,  20, -17, -18, -19, -20],
       [  2,   4,   6,   8,  -3,  -6,  -9, -12],
       [ 10,  12,  14,  16, -15, -18, -21, -24],
       [ 18,  20,  22,  24, -27, -30, -33, -36],
       [ 26,  28,  30,  32, -39, -42, -45, -48],
       [ 34,  36,  38,  40, -51, -54, -57, -60]])

In [17]: from numpy.random import default_rng

In [18]: rng = default_rng(42)

In [19]: a = rng.random((23, 11))

In [20]: a[-1]
Out[20]:
array([0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
       0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
       0.45389688])

In [21]: a[1, 4]
Out[21]: 0.2272387217847769

In [22]: a[1]
Out[22]:
array([0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
       0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
       0.35452597])

In [23]: a[1, :]
Out[23]:
array([0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
       0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
       0.35452597])

In [24]: a[:5]
Out[24]:
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, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
```

```
        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
        0.35452597],
       [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
        0.32582536],
       [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
        0.31236664],
       [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
        0.70516538]])

In [25]: a[0:5, :]
Out[25]:
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, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
        0.35452597],
       [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
        0.32582536],
       [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
        0.31236664],
       [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
        0.70516538]])

In [26]: a[0:5]
Out[26]:
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, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
        0.35452597],
       [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
        0.32582536],
       [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
        0.31236664],
       [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
        0.70516538]])
```

```
In [27]: a[-5:]
Out[27]:
array([[0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
        0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
        0.1218325 ],
       [0.50632993, 0.69426244, 0.58111661, 0.19977565, 0.80412453,
        0.71540713, 0.738984  , 0.13105775, 0.1237538 , 0.92756255,
        0.39757819],
       [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
        0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
        0.1403396 ],
       [0.41911432, 0.96623191, 0.59604255, 0.93302322, 0.80436092,
        0.4673816 , 0.78476345, 0.01783678, 0.109144  , 0.82942861,
        0.79681709],
       [0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
        0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
        0.45389688]])

In [28]: a[0:3, 4:9]
Out[28]:
array([[0.09417735, 0.97562235, 0.7611397 , 0.78606431, 0.12811363],
       [0.22723872, 0.55458479, 0.06381726, 0.82763117, 0.6316644 ],
       [0.466721  , 0.04380377, 0.15428949, 0.68304895, 0.74476216]])

In [29]: a[np.ix_([1, 3, 4], [0, 2])]
Out[29]:
array([[0.92676499, 0.82276161],
       [0.37045971, 0.18947136],
       [0.8322598 , 0.38747838]])

In [30]: a[2:21:2,:]
Out[30]:
array([[0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
        0.32582536],
       [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
        0.70516538],
       [0.55920716, 0.3039501 , 0.03081783, 0.43671739, 0.21458467,
        0.40852864, 0.85340307, 0.23393949, 0.05830274, 0.28138389,
        0.29359376],
       [0.16127178, 0.50104478, 0.1523121 , 0.69632038, 0.44615628,
        0.38102123, 0.30151209, 0.63028259, 0.36181261, 0.08764992,
        0.1180059 ],
       [0.45577629, 0.20236336, 0.30595662, 0.57921957, 0.17677278,
        0.85661428, 0.75851953, 0.71946296, 0.43209304, 0.62730884,
```

```
        0.58409797],
       [0.58106114, 0.3468698 , 0.59091549, 0.02280387, 0.95855921,
        0.48230344, 0.78273523, 0.08273   , 0.48665833, 0.49070699,
        0.93782645],
       [0.10857574, 0.67224009, 0.28123378, 0.65942263, 0.72699461,
        0.76864749, 0.10774095, 0.91601185, 0.23021399, 0.03741256,
        0.55485247],
       [0.43509706, 0.99237556, 0.89167727, 0.74860802, 0.89079249,
        0.89344664, 0.51885836, 0.31592905, 0.77201243, 0.66166126,
        0.37365773],
       [0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
        0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
        0.1218325 ],
       [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
        0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
        0.1403396 ]])

In [31]: a
Out[31]:
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, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
        0.35452597],
       [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
        0.32582536],
       [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
        0.31236664],
       [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
        0.70516538],
       [0.78072903, 0.45891578, 0.5687412 , 0.139797  , 0.11453007,
        0.66840296, 0.47109621, 0.56523611, 0.76499886, 0.63471832,
        0.5535794 ],
       [0.55920716, 0.3039501 , 0.03081783, 0.43671739, 0.21458467,
        0.40852864, 0.85340307, 0.23393949, 0.05830274, 0.28138389,
        0.29359376],
       [0.66191651, 0.55703215, 0.78389821, 0.66431354, 0.40638686,
        0.81402038, 0.16697292, 0.02271207, 0.09004786, 0.72235935,
        0.46187723],
       [0.16127178, 0.50104478, 0.1523121 , 0.69632038, 0.44615628,
        0.38102123, 0.30151209, 0.63028259, 0.36181261, 0.08764992,
        0.1180059 ],
       [0.96189766, 0.90858069, 0.69970713, 0.26586996, 0.96917638,
```

```
        0.7787509 , 0.71689019, 0.4493615 , 0.27224156, 0.09639096,
        0.9026024 ],
       [0.45577629, 0.20236336, 0.30595662, 0.57921957, 0.17677278,
        0.85661428, 0.75851953, 0.71946296, 0.43209304, 0.62730884,
        0.58409797],
       [0.6498466 , 0.08444432, 0.4158074 , 0.04161417, 0.49399082,
        0.32986121, 0.14452419, 0.10340297, 0.58764457, 0.17059297,
        0.92512012],
       [0.58106114, 0.3468698 , 0.59091549, 0.02280387, 0.95855921,
        0.48230344, 0.78273523, 0.08273   , 0.48665833, 0.49070699,
        0.93782645],
       [0.57172805, 0.4734894 , 0.26697566, 0.331569  , 0.5206724 ,
        0.43891146, 0.02161208, 0.82629192, 0.89616077, 0.14024909,
        0.55403614],
       [0.10857574, 0.67224009, 0.28123378, 0.65942263, 0.72699461,
        0.76864749, 0.10774095, 0.91601185, 0.23021399, 0.03741256,
        0.55485247],
       [0.37092228, 0.82978974, 0.80825147, 0.31713889, 0.9528994 ,
        0.29091784, 0.51505713, 0.25596509, 0.93604357, 0.16460782,
        0.04491062],
       [0.43509706, 0.99237556, 0.89167727, 0.74860802, 0.89079249,
        0.89344664, 0.51885836, 0.31592905, 0.77201243, 0.66166126,
        0.37365773],
       [0.09446667, 0.74678961, 0.26246052, 0.93681315, 0.24097058,
        0.12275793, 0.83111267, 0.15328432, 0.17926831, 0.59938279,
        0.87456204],
       [0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
        0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
        0.1218325 ],
       [0.50632993, 0.69426244, 0.58111661, 0.19977565, 0.80412453,
        0.71540713, 0.738984  , 0.13105775, 0.1237538 , 0.92756255,
        0.39757819],
       [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
        0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
        0.1403396 ],
       [0.41911432, 0.96623191, 0.59604255, 0.93302322, 0.80436092,
        0.4673816 , 0.78476345, 0.01783678, 0.109144  , 0.82942861,
        0.79681709],
       [0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
        0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
        0.45389688]])

In [32]: a =
    ↪  np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])

In [33]: a[::2, :]
Out[33]:
```

```
array([[ 1,  2,  3,  4],
       [ 9, 10, 11, 12],
       [17, 18, 19, 20]])

In [34]: a[::-1,:]
Out[34]:
array([[17, 18, 19, 20],
       [13, 14, 15, 16],
       [ 9, 10, 11, 12],
       [ 5,  6,  7,  8],
       [ 1,  2,  3,  4]])

In [35]: a[np.r_[:len(a),0]]
Out[35]:
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20],
       [ 1,  2,  3,  4]])

In [36]: a.transpose()
Out[36]:
array([[ 1,  5,  9, 13, 17],
       [ 2,  6, 10, 14, 18],
       [ 3,  7, 11, 15, 19],
       [ 4,  8, 12, 16, 20]])

In [37]: a.T
Out[37]:
array([[ 1,  5,  9, 13, 17],
       [ 2,  6, 10, 14, 18],
       [ 3,  7, 11, 15, 19],
       [ 4,  8, 12, 16, 20]])

In [38]: a.conj().transpose()
Out[38]:
array([[ 1,  5,  9, 13, 17],
       [ 2,  6, 10, 14, 18],
       [ 3,  7, 11, 15, 19],
       [ 4,  8, 12, 16, 20]])

In [39]: a.conj().T
Out[39]:
array([[ 1,  5,  9, 13, 17],
       [ 2,  6, 10, 14, 18],
       [ 3,  7, 11, 15, 19],
```

```
       [ 4,  8, 12, 16, 20]])

In [40]: b = a.T

In [41]: a @ b
Out[41]:
array([[  30,   70,  110,  150,  190],
       [  70,  174,  278,  382,  486],
       [ 110,  278,  446,  614,  782],
       [ 150,  382,  614,  846, 1078],
       [ 190,  486,  782, 1078, 1374]])

In [42]: b = 3

In [43]: a * b
Out[43]:
array([[ 3,  6,  9, 12],
       [15, 18, 21, 24],
       [27, 30, 33, 36],
       [39, 42, 45, 48],
       [51, 54, 57, 60]])

In [44]: a/b
Out[44]:
array([[0.33333333, 0.66666667, 1.        , 1.33333333],
       [1.66666667, 2.        , 2.33333333, 2.66666667],
       [3.        , 3.33333333, 3.66666667, 4.        ],
       [4.33333333, 4.66666667, 5.        , 5.33333333],
       [5.66666667, 6.        , 6.33333333, 6.66666667]])

In [45]: a**3
Out[45]:
array([[   1,    8,   27,   64],
       [ 125,  216,  343,  512],
       [ 729, 1000, 1331, 1728],
       [2197, 2744, 3375, 4096],
       [4913, 5832, 6859, 8000]])

In [46]: (a > 0.5)
Out[46]:
array([[ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True]])

In [47]: np.nonzero(a > 0.5)
```

```
Out[47]:
(array([0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4]),
 array([0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3]))

In [48]: v = np.array([1, 0, 0, 1])

In [49]: a[:,np.nonzero(v > 0.5)[0]]
Out[49]:
array([[ 1,  4],
       [ 5,  8],
       [ 9, 12],
       [13, 16],
       [17, 20]])

In [50]: a[:, v.T > 0.5]
Out[50]:
array([[ 1,  4],
       [ 5,  8],
       [ 9, 12],
       [13, 16],
       [17, 20]])

In [51]: a = a/10

In [52]: a[a < 0.5]=0

In [53]: a
Out[53]:
array([[0. , 0. , 0. , 0. ],
       [0.5, 0.6, 0.7, 0.8],
       [0.9, 1. , 1.1, 1.2],
       [1.3, 1.4, 1.5, 1.6],
       [1.7, 1.8, 1.9, 2. ]])

In [54]: a =
    ↪  np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])

In [55]: a = a/10

In [56]: a * (a > 0.5)
Out[56]:
array([[0. , 0. , 0. , 0. ],
       [0. , 0.6, 0.7, 0.8],
       [0.9, 1. , 1.1, 1.2],
       [1.3, 1.4, 1.5, 1.6],
       [1.7, 1.8, 1.9, 2. ]])
```

```
In [57]: a[:] = 3

In [58]: a
Out[58]:
array([[3., 3., 3., 3.],
       [3., 3., 3., 3.],
       [3., 3., 3., 3.],
       [3., 3., 3., 3.],
       [3., 3., 3., 3.]])

In [59]: x =
    ↪  np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])

In [60]: y = x.copy()

In [61]: y
Out[61]:
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20]])

In [62]: y = x[1, :].copy()

In [63]: y
Out[63]: array([5, 6, 7, 8])

In [64]: y = x.flatten()

In [65]: y
Out[65]:
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20])

In [66]: np.arange(1., 11.)
Out[66]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])

In [67]: np.r_[1.:11.]
Out[67]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])

In [68]: np.r_[1:10:10j]
Out[68]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])

In [69]: np.arange(10.)
Out[69]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
```

```
In [70]: np.r_[:10.]
Out[70]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])

In [71]: np.r_[:9:10j]
Out[71]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])

In [72]: np.arange(1.,11.)[:, np.newaxis]
Out[72]:
array([[ 1.],
       [ 2.],
       [ 3.],
       [ 4.],
       [ 5.],
       [ 6.],
       [ 7.],
       [ 8.],
       [ 9.],
       [10.]])

In [73]: np.zeros((3, 4))
Out[73]:
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])

In [74]: np.zeros((3, 4, 5))
Out[74]:
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 [75]: np.ones((3, 4))
Out[75]:
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

```
In [76]: np.eye(3)
Out[76]:
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])

In [77]: a = np.array([[3,2],[2,1]])

In [78]: np.diag(a)
Out[78]: array([3, 1])

In [79]: np.diag(v, 0)
Out[79]:
array([[1, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 1]])

In [80]: rng = default_rng(42)

In [81]: rng.random((3, 4))
Out[81]:
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]])

In [82]: np.linspace(1,3,4)
Out[82]: array([1.        , 1.66666667, 2.33333333, 3.        ])

In [83]: np.mgrid[0:9.,0:6.]
Out[83]:
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 [84]: np.meshgrid([1,2,4],[2,4,5])
Out[84]:
[array([[1, 2, 4],
        [1, 2, 4],
        [1, 2, 4]]),
 array([[2, 2, 2],
        [4, 4, 4],
        [5, 5, 5]])]

In [85]: np.ix_([1,2,4],[2,4,5])
Out[85]:
(array([[1],
        [2],
        [4]]),
 array([[2, 4, 5]]))

In [86]: m = 1

In [87]: n = 2

In [88]: np.tile(a, (m, n))
Out[88]:
array([[3, 2, 3, 2],
       [2, 1, 2, 1]])

In [89]: b = np.array([[2, 3],[0,5]])

In [90]: np.concatenate((a,b),1)
Out[90]:
array([[3, 2, 2, 3],
       [2, 1, 0, 5]])

In [91]: np.hstack((a,b))
Out[91]:
array([[3, 2, 2, 3],
       [2, 1, 0, 5]])

In [92]: np.column_stack((a,b))
Out[92]:
array([[3, 2, 2, 3],
       [2, 1, 0, 5]])
```

```
In [93]: np.c_[a,b]
Out[93]:
array([[3, 2, 2, 3],
       [2, 1, 0, 5]])

In [94]: np.concatenate((a,b))
Out[94]:
array([[3, 2],
       [2, 1],
       [2, 3],
       [0, 5]])

In [95]: np.vstack((a,b))
Out[95]:
array([[3, 2],
       [2, 1],
       [2, 3],
       [0, 5]])

In [96]: np.r_[a,b]
Out[96]:
array([[3, 2],
       [2, 1],
       [2, 3],
       [0, 5]])

In [97]: a.max()
Out[97]: 3

In [98]: np.nanmax(a)
Out[98]: 3

In [99]: a.max(0)
Out[99]: array([3, 2])

In [100]: a.max(1)
Out[100]: array([3, 2])

In [101]: np.maximum(a, b)
Out[101]:
array([[3, 3],
       [2, 5]])

In [102]: v
Out[102]: array([1, 0, 0, 1])

In [103]: np.sqrt(v @ v)
```

```
Out[103]: 1.4142135623730951

In [104]: np.linalg.norm(v)
Out[104]: 1.4142135623730951

In [105]: np.logical_and(a,b)
Out[105]:
array([[ True,  True],
       [False,  True]])

In [106]: np.logical_or(a,b)
Out[106]:
array([[ True,  True],
       [ True,  True]])

In [107]: a & b
Out[107]:
array([[2, 2],
       [0, 1]])

In [108]: a | b
Out[108]:
array([[3, 3],
       [2, 5]])

In [109]: np.linalg.inv(a)
Out[109]:
array([[-1.,  2.],
       [ 2., -3.]])

In [110]: np.linalg.pinv(a)
Out[110]:
array([[-1.,  2.],
       [ 2., -3.]])

In [111]: np.linalg.matrix_rank(a)
Out[111]: 2

In [112]: a
Out[112]:
array([[3, 2],
       [2, 1]])

In [113]: b = np.array([[1],[1]])

In [114]: np.linalg.solve(a, b)
Out[114]:
```

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

In [115]: b = b.T

In [116]: np.linalg.solve(a.T, b.T).T
Out[116]: array([[ 1., -1.]])

In [117]: a = np.array([[5,7,6,5],[7,10,8,7],[6,8,10,9],[5,7,9,10]])

In [118]: U, S, Vh = np.linalg.svd(a)

In [119]: U
Out[119]:
array([[-0.38026207, -0.39630556, -0.09330504, -0.83044375],
       [-0.52856785, -0.61486128,  0.30165233,  0.50156506],
       [-0.55195485,  0.27160104, -0.76031843,  0.2085536 ],
       [-0.52092478,  0.62539618,  0.56764067, -0.12369746]])

In [120]: S
Out[120]: array([3.02886853e+01, 3.85805746e+00, 8.43107150e-01, 1.01500484e-02])

In [121]: Vh.T
Out[121]:
array([[-0.38026207, -0.39630556, -0.09330504, -0.83044375],
       [-0.52856785, -0.61486128,  0.30165233,  0.50156506],
       [-0.55195485,  0.27160104, -0.76031843,  0.2085536 ],
       [-0.52092478,  0.62539618,  0.56764067, -0.12369746]])

In [122]: S**2
Out[122]: array([9.17404460e+02, 1.48846073e+01, 7.10829666e-01, 1.03023482e-04])

In [123]: Vh@Vh.T
Out[123]: array([[ 1.00000000e+00, -7.05395086e-17,  4.76007668e-17,
        -1.07622236e-16],
       [-7.05395086e-17,  1.00000000e+00,  2.11109113e-16,
         2.12834394e-16],
       [ 4.76007668e-17,  2.11109113e-16,  1.00000000e+00,
        -1.38049606e-16],
       [-1.07622236e-16,  2.12834394e-16, -1.38049606e-16,
         1.00000000e+00]])

In [124]: U@np.diag(S)@Vh
Out[124]:
array([[ 5.,  7.,  6.,  5.],
       [ 7., 10.,  8.,  7.],
       [ 6.,  8., 10.,  9.],
```

```
       [ 5.,  7.,  9., 10.]])

In [125]: np.linalg.cholesky(a)
Out[125]:
array([[ 2.23606798e+00,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00],
       [ 3.13049517e+00,  4.47213595e-01,  0.00000000e+00,
         0.00000000e+00],
       [ 2.68328157e+00, -8.94427191e-01,  1.41421356e+00,
         0.00000000e+00],
       [ 2.23606798e+00, -6.32813837e-16,  2.12132034e+00,
         7.07106781e-01]])

In [126]: D,V = np.linalg.eig(a)

In [127]: D
Out[127]: array([3.02886853e+01, 3.85805746e+00, 1.01500484e-02, 8.43107150e-01])

In [128]: V
Out[128]:
array([[-0.38026207, -0.39630556, -0.83044375, -0.09330504],
       [-0.52856785, -0.61486128,  0.50156506,  0.30165233],
       [-0.55195485,  0.27160104,  0.2085536 , -0.76031843],
       [-0.52092478,  0.62539618, -0.12369746,  0.56764067]])

In [129]: b = np.eye(4)

In [130]: D,V = scipy.linalg.eig(a, b)

In [131]: D
Out[131]:
array([3.02886853e+01+0.j, 3.85805746e+00+0.j, 1.01500484e-02+0.j,
       8.43107150e-01+0.j])

In [132]: V
Out[132]:
array([[ 0.38026207, -0.39630556,  0.83044375, -0.09330504],
       [ 0.52856785, -0.61486128, -0.50156506,  0.30165233],
       [ 0.55195485,  0.27160104, -0.2085536 , -0.76031843],
       [ 0.52092478,  0.62539618,  0.12369746,  0.56764067]])

In [133]: D,V = scipy.sparse.linalg.eigs(b, k=2)

In [134]: D
Out[134]: array([1.+0.j, 1.+0.j])

In [135]: V
```

```
Out[135]:
array([[-0.96312067+0.j,  0.33743833+0.j],
       [-0.08300551+0.j, -0.31645308+0.j],
       [ 0.01517654+0.j,  0.45922686+0.j],
       [-0.25549625+0.j, -0.7583558 +0.j]])

In [136]: Q,R = np.linalg.qr(a)

In [137]: Q
Out[137]:
array([[-0.43033148, -0.08439495, -0.36952858, -0.81923192],
       [-0.60246408, -0.57388564, -0.25706336,  0.49153915],
       [-0.51639778,  0.81019149, -0.12853168,  0.24576958],
       [-0.43033148, -0.08439495,  0.88365529, -0.16384638]])

In [138]: R
Out[138]:
array([[-11.61895004, -16.18046376, -16.43866265, -15.31980079],
       [  0.        ,  -0.43885373,   2.2449056 ,   2.00859974],
       [  0.        ,   0.        ,   2.39390251,   4.03268141],
       [  0.        ,   0.        ,   0.        ,  -0.08192319]])

In [139]: Q@R
Out[139]:
array([[ 5.,  7.,  6.,  5.],
       [ 7., 10.,  8.,  7.],
       [ 6.,  8., 10.,  9.],
       [ 5.,  7.,  9., 10.]])

In [140]: P,L,U = scipy.linalg.lu(a)

In [141]: P
Out[141]:
array([[0., 0., 0., 1.],
       [1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.]])

In [142]: L
Out[142]:
array([[ 1.        ,  0.        ,  0.        ,  0.        ],
       [ 0.85714286,  1.        ,  0.        ,  0.        ],
       [ 0.71428571,  0.25      ,  1.        ,  0.        ],
       [ 0.71428571,  0.25      , -0.2       ,  1.        ]])

In [143]: U
Out[143]:
```

```
array([[ 7.        , 10.        ,  8.        ,  7.        ],
       [ 0.        , -0.57142857,  3.14285714,  3.        ],
       [ 0.        ,  0.        ,  2.5       ,  4.25      ],
       [ 0.        ,  0.        ,  0.        ,  0.1       ]])

In [144]: P@L@U
Out[144]:
array([[ 5.,  7.,  6.,  5.],
       [ 7., 10.,  8.,  7.],
       [ 6.,  8., 10.,  9.],
       [ 5.,  7.,  9., 10.]])

In [145]: scipy.sparse.linalg.cg(a,np.array([1,1,1,1]))
Out[145]: (array([ 20., -12.,  -5.,   3.]), 0)

In [146]: np.fft.fft(a)
Out[146]:
array([[23.+0.j, -1.-2.j, -1.+0.j, -1.+2.j],
       [32.+0.j, -1.-3.j, -2.+0.j, -1.+3.j],
       [33.+0.j, -4.+1.j, -1.+0.j, -4.-1.j],
       [31.+0.j, -4.+3.j, -3.+0.j, -4.-3.j]])

In [147]: np.fft.ifft(a)
Out[147]:
array([[ 5.75+0.j  , -0.25+0.5j , -0.25+0.j  , -0.25-0.5j ],
       [ 8.  +0.j  , -0.25+0.75j, -0.5 +0.j  , -0.25-0.75j],
       [ 8.25+0.j  , -1.  -0.25j, -0.25+0.j  , -1.  +0.25j],
       [ 7.75+0.j  , -1.  -0.75j, -0.75+0.j  , -1.  +0.75j]])

In [148]: np.sort(a)
Out[148]:
array([[ 5,  5,  6,  7],
       [ 7,  7,  8, 10],
       [ 6,  8,  9, 10],
       [ 5,  7,  9, 10]])

In [149]: np.sort(a, axis=1)
Out[149]:
array([[ 5,  5,  6,  7],
       [ 7,  7,  8, 10],
       [ 6,  8,  9, 10],
       [ 5,  7,  9, 10]])

In [150]: I = np.argsort(a[:, 0])

In [151]: b = a[I,:]
```

```
In [152]: b
Out[152]:
array([[ 5,  7,  6,  5],
       [ 5,  7,  9, 10],
       [ 6,  8, 10,  9],
       [ 7, 10,  8,  7]])

In [153]: Z = np.array([[0,1],[1,1],[2,1],[3,1]])

In [154]: y = np.array([-1, 0.2, 0.9, 2.1])

In [155]: x = np.linalg.lstsq(Z, y,rcond=None)

In [156]: x
Out[156]: (array([ 1.  , -0.95]), array([0.05]), 2, array([4.10003045,
    ↪  1.09075677]))

In [157]: Z@x[0]
Out[157]: array([-0.95,  0.05,  1.05,  2.05])

In [158]: q=1

In [159]: scipy.signal.resample(y, int(np.ceil(len(y)/q)))
Out[159]: array([-1. ,  0.2,  0.9,  2.1])

In [160]: np.unique(a)
Out[160]: array([ 5,  6,  7,  8,  9, 10])

In [161]: a = np.array([[[0], [1], [2]]])

In [162]: a.squeeze()
Out[162]: array([0, 1, 2])
```

Task 3: Run the following script in IPython and paste the figure created by the script into your report.

```
(base) mason@masons-mbp Desktop % ipython
Python 3.9.19 (main, May  6 2024, 14:46:57)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.15.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import matplotlib.pyplot as plt
Matplotlib is building the font cache; this may take a moment.

In [2]: plt.plot([1,2,3,4], [1,2,7,14])
Out[2]: [<matplotlib.lines.Line2D at 0x7fd741fc6f40>]

In [3]: plt.axis([0, 6, 0, 20])
Out[3]: (0.0, 6.0, 0.0, 20.0)

In [4]: plt.show()
2024-09-08 17:18:45.301 python[20991:3560702] +[CATransaction synchronize] called
↪  within transaction
2024-09-08 17:18:45.609 python[20991:3560702] +[CATransaction synchronize] called
↪  within transaction
```

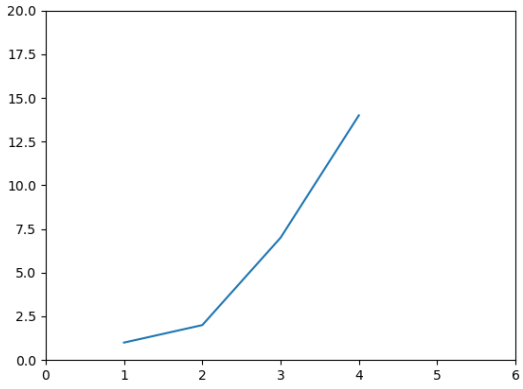

Figure 1: Output from Task 3

Task 4: Use Matplotlib to create a figure of your choice in IPython. Paste your code and figure into your report.

```
(base) mason@masons-mbp Desktop % ipython
Python 3.9.19 (main, May  6 2024, 14:46:57)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.15.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import numpy as np

In [2]: import matplotlib.pyplot as plt

In [3]: x = np.linspace(0,100,10001)

In [4]: y = np.cos(x)*np.sqrt(x)

In [5]: plt.plot(x, y, 'r-')
Out[5]: [<matplotlib.lines.Line2D at 0x7fadd2bb5df0>]

In [6]: plt.grid()

In [7]: plt.show()
2024-09-08 18:07:00.499 python[21981:3600780] +[CATransaction synchronize] called
    within transaction
2024-09-08 18:07:00.823 python[21981:3600780] +[CATransaction synchronize] called
    within transaction
2024-09-08 18:07:08.918 python[21981:3600780] +[CATransaction synchronize] called
    within transaction
2024-09-08 18:07:09.789 python[21981:3600780] +[CATransaction synchronize] called
    within transaction
```

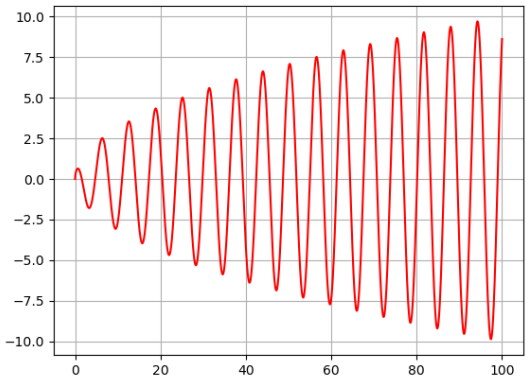

Figure 2: Output from Task 4

Task 5: Paste your VCS account into your report.

GitHub Account: https://github.com/masonweiss

Task 6: Start a new project in Pycharm. Commit and push your project to GitHub as a public project. Paste the link of your project in your report.

Repository Link: https://github.com/masonweiss/ELEC576