

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

## Assignment 0

### Task 1:

```
$ conda info
```

```
1      active environment : base
2      active env location : /Users/mason/opt/anaconda3
3          shell level : 1
4      user config file : /Users/mason/.condarc
5  populated config files : /Users/mason/.condarc
6          conda version : 24.7.1
7      conda-build version : 24.7.1
8          python version : 3.9.19.final.0
9              solver : libmamba (default)
10     virtual packages : __archspec=1=cannonlake
11                       __conda=24.7.1=0
12                       __osx=10.16=0
13                       __unix=0=0
14     base environment : /Users/mason/opt/anaconda3 (writable)
15     conda av data dir : /Users/mason/opt/anaconda3/etc/conda
16     conda av metadata url : None
17     channel URLs : https://repo.anaconda.com/pkgs/main/osx-64
18                  https://repo.anaconda.com/pkgs/main/noarch
19                  https://repo.anaconda.com/pkgs/r/osx-64
20                  https://repo.anaconda.com/pkgs/r/noarch
21     package cache : /Users/mason/opt/anaconda3/pkgs
22                  /Users/mason/.conda/pkgs
23     envs directories : /Users/mason/opt/anaconda3/envs
24                  /Users/mason/.conda/envs
25         platform : osx-64
26     user-agent : conda/24.7.1 requests/2.32.3 CPython/3.9.19
27                ↪ Darwin/22.3.0 OSX/10.16 solver/libmamba
28                ↪ conda-libmamba-solver/24.7.0 libmambapy/1.5.8 aau/0.4.4 c/. s/.
29                ↪ e/.
27     UID:GID : 501:20
28     netrc file : None
29     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.

1 In [1]: import numpy as np
2
3 In [2]: import scipy.linalg
4
5 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]])
6
7 In [4]: np.ndim(a)
8 Out[4]: 2
9
10 In [5]: a.ndim
11 Out[5]: 2
12
13 In [6]: np.size(a)
14 Out[6]: 20
15
16 In [7]: a.size
17 Out[7]: 20
18
19 In [8]: np.shape(a)
20 Out[8]: (5, 4)
21
22 In [9]: a.shape
23 Out[9]: (5, 4)
24
25 In [10]: n = 2
26
27 In [11]: a.shape[n-1]
28 Out[11]: 4
29
30 In [12]: np.array([[1., 2., 3.], [4., 5., 6.]])
31 Out[12]:
32 array([[1., 2., 3.],
33        [4., 5., 6.]])
34
35 In [13]: b = a*-1
36
37 In [14]: c = a*2
38
39 In [15]: d = a*-3
```

```
40
41 In [16]: np.block([[a, b], [c, d]])
42 Out[16]:
43 array([[ 1,  2,  3,  4, -1, -2, -3, -4],
44        [ 5,  6,  7,  8, -5, -6, -7, -8],
45        [ 9, 10, 11, 12, -9, -10, -11, -12],
46        [13, 14, 15, 16, -13, -14, -15, -16],
47        [17, 18, 19, 20, -17, -18, -19, -20],
48        [ 2,  4,  6,  8, -3, -6, -9, -12],
49        [10, 12, 14, 16, -15, -18, -21, -24],
50        [18, 20, 22, 24, -27, -30, -33, -36],
51        [26, 28, 30, 32, -39, -42, -45, -48],
52        [34, 36, 38, 40, -51, -54, -57, -60]])
53
54 In [17]: from numpy.random import default_rng
55
56 In [18]: rng = default_rng(42)
57
58 In [19]: a = rng.random((23, 11))
59
60 In [20]: a[-1]
61 Out[20]:
62 array([0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
63        0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
64        0.45389688])
65
66 In [21]: a[1, 4]
67 Out[21]: 0.2272387217847769
68
69 In [22]: a[1]
70 Out[22]:
71 array([0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
72        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
73        0.35452597])
74
75 In [23]: a[1, : ]
76 Out[23]:
77 array([0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
78        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
79        0.35452597])
80
81 In [24]: a[:5]
82 Out[24]:
83 array([[0.77395605, 0.43887844, 0.85859792, 0.69736803, 0.09417735,
84        0.97562235, 0.7611397 , 0.78606431, 0.12811363, 0.45038594,
85        0.37079802],
86        [0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
```

```

87         0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
88         0.35452597],
89     [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721 ,
90     0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
91     0.32582536],
92     [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
93     0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
94     0.31236664],
95     [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
96     0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
97     0.70516538]])
98
99 In [25]: a[0:5, :]
100 Out[25]:
101 array([[0.77395605, 0.43887844, 0.85859792, 0.69736803, 0.09417735,
102         0.97562235, 0.7611397 , 0.78606431, 0.12811363, 0.45038594,
103         0.37079802],
104        [0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
105        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
106        0.35452597],
107        [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721 ,
108        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
109        0.32582536],
110        [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
111        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
112        0.31236664],
113        [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
114        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
115        0.70516538]])
116
117 In [26]: a[0:5]
118 Out[26]:
119 array([[0.77395605, 0.43887844, 0.85859792, 0.69736803, 0.09417735,
120         0.97562235, 0.7611397 , 0.78606431, 0.12811363, 0.45038594,
121         0.37079802],
122        [0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
123        0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
124        0.35452597],
125        [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721 ,
126        0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
127        0.32582536],
128        [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
129        0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
130        0.31236664],
131        [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
132        0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
133        0.70516538]])

```

```
134
135 In [27]: a[-5:]
136 Out[27]:
137 array([[0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
138         0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
139         0.1218325 ],
140        [0.50632993, 0.69426244, 0.58111661, 0.19977565, 0.80412453,
141         0.71540713, 0.738984 , 0.13105775, 0.1237538 , 0.92756255,
142         0.39757819],
143        [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
144         0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
145         0.1403396 ],
146        [0.41911432, 0.96623191, 0.59604255, 0.93302322, 0.80436092,
147         0.4673816 , 0.78476345, 0.01783678, 0.109144 , 0.82942861,
148         0.79681709],
149        [0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
150         0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
151         0.45389688]])
152
153 In [28]: a[0:3, 4:9]
154 Out[28]:
155 array([[0.09417735, 0.97562235, 0.7611397 , 0.78606431, 0.12811363],
156        [0.22723872, 0.55458479, 0.06381726, 0.82763117, 0.6316644 ],
157        [0.466721 , 0.04380377, 0.15428949, 0.68304895, 0.74476216]])
158
159 In [29]: a[np.ix_([1, 3, 4], [0, 2])]
160 Out[29]:
161 array([[0.92676499, 0.82276161],
162        [0.37045971, 0.18947136],
163        [0.8322598 , 0.38747838]])
164
165 In [30]: a[2:21:2,: ]
166 Out[30]:
167 array([[0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721 ,
168         0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
169         0.32582536],
170        [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
171         0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
172         0.70516538],
173        [0.55920716, 0.3039501 , 0.03081783, 0.43671739, 0.21458467,
174         0.40852864, 0.85340307, 0.23393949, 0.05830274, 0.28138389,
175         0.29359376],
176        [0.16127178, 0.50104478, 0.1523121 , 0.69632038, 0.44615628,
177         0.38102123, 0.30151209, 0.63028259, 0.36181261, 0.08764992,
178         0.1180059 ],
179        [0.45577629, 0.20236336, 0.30595662, 0.57921957, 0.17677278,
180         0.85661428, 0.75851953, 0.71946296, 0.43209304, 0.62730884,
```

```

181         0.58409797],
182     [0.58106114, 0.3468698 , 0.59091549, 0.02280387, 0.95855921,
183      0.48230344, 0.78273523, 0.08273   , 0.48665833, 0.49070699,
184      0.93782645],
185     [0.10857574, 0.67224009, 0.28123378, 0.65942263, 0.72699461,
186      0.76864749, 0.10774095, 0.91601185, 0.23021399, 0.03741256,
187      0.55485247],
188     [0.43509706, 0.99237556, 0.89167727, 0.74860802, 0.89079249,
189      0.89344664, 0.51885836, 0.31592905, 0.77201243, 0.66166126,
190      0.37365773],
191     [0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
192      0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
193      0.1218325 ],
194     [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
195      0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
196      0.1403396 ]])
197
198 In [31]: a
199 Out[31]:
200 array([[0.77395605, 0.43887844, 0.85859792, 0.69736803, 0.09417735,
201         0.97562235, 0.7611397 , 0.78606431, 0.12811363, 0.45038594,
202         0.37079802],
203        [0.92676499, 0.64386512, 0.82276161, 0.4434142 , 0.22723872,
204         0.55458479, 0.06381726, 0.82763117, 0.6316644 , 0.75808774,
205         0.35452597],
206        [0.97069802, 0.89312112, 0.7783835 , 0.19463871, 0.466721  ,
207         0.04380377, 0.15428949, 0.68304895, 0.74476216, 0.96750973,
208         0.32582536],
209        [0.37045971, 0.46955581, 0.18947136, 0.12992151, 0.47570493,
210         0.22690935, 0.66981399, 0.43715192, 0.8326782 , 0.7002651 ,
211         0.31236664],
212        [0.8322598 , 0.80476436, 0.38747838, 0.2883281 , 0.6824955 ,
213         0.13975248, 0.1999082 , 0.00736227, 0.78692438, 0.66485086,
214         0.70516538],
215        [0.78072903, 0.45891578, 0.5687412 , 0.139797  , 0.11453007,
216         0.66840296, 0.47109621, 0.56523611, 0.76499886, 0.63471832,
217         0.5535794 ],
218        [0.55920716, 0.3039501 , 0.03081783, 0.43671739, 0.21458467,
219         0.40852864, 0.85340307, 0.23393949, 0.05830274, 0.28138389,
220         0.29359376],
221        [0.66191651, 0.55703215, 0.78389821, 0.66431354, 0.40638686,
222         0.81402038, 0.16697292, 0.02271207, 0.09004786, 0.72235935,
223         0.46187723],
224        [0.16127178, 0.50104478, 0.1523121 , 0.69632038, 0.44615628,
225         0.38102123, 0.30151209, 0.63028259, 0.36181261, 0.08764992,
226         0.1180059 ],
227        [0.96189766, 0.90858069, 0.69970713, 0.26586996, 0.96917638,
```

```

228     0.7787509 , 0.71689019, 0.4493615 , 0.27224156, 0.09639096,
229     0.9026024 ],
230     [0.45577629, 0.20236336, 0.30595662, 0.57921957, 0.17677278,
231     0.85661428, 0.75851953, 0.71946296, 0.43209304, 0.62730884,
232     0.58409797],
233     [0.6498466 , 0.08444432, 0.4158074 , 0.04161417, 0.49399082,
234     0.32986121, 0.14452419, 0.10340297, 0.58764457, 0.17059297,
235     0.92512012],
236     [0.58106114, 0.3468698 , 0.59091549, 0.02280387, 0.95855921,
237     0.48230344, 0.78273523, 0.08273 , 0.48665833, 0.49070699,
238     0.93782645],
239     [0.57172805, 0.4734894 , 0.26697566, 0.331569 , 0.5206724 ,
240     0.43891146, 0.02161208, 0.82629192, 0.89616077, 0.14024909,
241     0.55403614],
242     [0.10857574, 0.67224009, 0.28123378, 0.65942263, 0.72699461,
243     0.76864749, 0.10774095, 0.91601185, 0.23021399, 0.03741256,
244     0.55485247],
245     [0.37092228, 0.82978974, 0.80825147, 0.31713889, 0.9528994 ,
246     0.29091784, 0.51505713, 0.25596509, 0.93604357, 0.16460782,
247     0.04491062],
248     [0.43509706, 0.99237556, 0.89167727, 0.74860802, 0.89079249,
249     0.89344664, 0.51885836, 0.31592905, 0.77201243, 0.66166126,
250     0.37365773],
251     [0.09446667, 0.74678961, 0.26246052, 0.93681315, 0.24097058,
252     0.12275793, 0.83111267, 0.15328432, 0.17926831, 0.59938279,
253     0.87456204],
254     [0.19643467, 0.31032367, 0.77740484, 0.97182643, 0.50074119,
255     0.1438975 , 0.01393629, 0.22965603, 0.13182222, 0.67765867,
256     0.1218325 ],
257     [0.50632993, 0.69426244, 0.58111661, 0.19977565, 0.80412453,
258     0.71540713, 0.738984 , 0.13105775, 0.1237538 , 0.92756255,
259     0.39757819],
260     [0.30094869, 0.48858405, 0.66286421, 0.95562326, 0.28644623,
261     0.92480843, 0.02485949, 0.55519804, 0.63397511, 0.1058974 ,
262     0.1403396 ],
263     [0.41911432, 0.96623191, 0.59604255, 0.93302322, 0.80436092,
264     0.4673816 , 0.78476345, 0.01783678, 0.109144 , 0.82942861,
265     0.79681709],
266     [0.23264074, 0.53076959, 0.60601582, 0.86773895, 0.60310716,
267     0.41257157, 0.37418404, 0.42588209, 0.65193103, 0.86749063,
268     0.45389688]])
269
270 In [32]: a =
271     ↪ np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])
272
273 In [33]: a[:, :2]
274 Out[33]:

```

```
274 array([[ 1,  2,  3,  4],
275         [ 9, 10, 11, 12],
276         [17, 18, 19, 20]])
277
278 In [34]: a[::-1,:]
279 Out[34]:
280 array([[17, 18, 19, 20],
281        [13, 14, 15, 16],
282        [ 9, 10, 11, 12],
283        [ 5,  6,  7,  8],
284        [ 1,  2,  3,  4]])
285
286 In [35]: a[np.r_[len(a),0]]
287 Out[35]:
288 array([[ 1,  2,  3,  4],
289        [ 5,  6,  7,  8],
290        [ 9, 10, 11, 12],
291        [13, 14, 15, 16],
292        [17, 18, 19, 20],
293        [ 1,  2,  3,  4]])
294
295 In [36]: a.transpose()
296 Out[36]:
297 array([[ 1,  5,  9, 13, 17],
298        [ 2,  6, 10, 14, 18],
299        [ 3,  7, 11, 15, 19],
300        [ 4,  8, 12, 16, 20]])
301
302 In [37]: a.T
303 Out[37]:
304 array([[ 1,  5,  9, 13, 17],
305        [ 2,  6, 10, 14, 18],
306        [ 3,  7, 11, 15, 19],
307        [ 4,  8, 12, 16, 20]])
308
309 In [38]: a.conj().transpose()
310 Out[38]:
311 array([[ 1,  5,  9, 13, 17],
312        [ 2,  6, 10, 14, 18],
313        [ 3,  7, 11, 15, 19],
314        [ 4,  8, 12, 16, 20]])
315
316 In [39]: a.conj().T
317 Out[39]:
318 array([[ 1,  5,  9, 13, 17],
319        [ 2,  6, 10, 14, 18],
320        [ 3,  7, 11, 15, 19],
```



```
321         [ 4,  8, 12, 16, 20]))
322
323 In [40]: b = a.T
324
325 In [41]: a @ b
326 Out[41]:
327 array([[ 30,   70,  110,  150,  190],
328        [  70,  174,  278,  382,  486],
329        [ 110,  278,  446,  614,  782],
330        [ 150,  382,  614,  846, 1078],
331        [ 190,  486,  782, 1078, 1374]])
332
333 In [42]: b = 3
334
335 In [43]: a * b
336 Out[43]:
337 array([[ 3,  6,  9, 12],
338        [15, 18, 21, 24],
339        [27, 30, 33, 36],
340        [39, 42, 45, 48],
341        [51, 54, 57, 60]])
342
343 In [44]: a/b
344 Out[44]:
345 array([[0.33333333, 0.66666667, 1.          , 1.33333333],
346        [1.66666667, 2.          , 2.33333333, 2.66666667],
347        [3.          , 3.33333333, 3.66666667, 4.          ],
348        [4.33333333, 4.66666667, 5.          , 5.33333333],
349        [5.66666667, 6.          , 6.33333333, 6.66666667]])
350
351 In [45]: a**3
352 Out[45]:
353 array([[ 1,   8,  27,  64],
354        [125, 216, 343, 512],
355        [ 729, 1000, 1331, 1728],
356        [2197, 2744, 3375, 4096],
357        [4913, 5832, 6859, 8000]])
358
359 In [46]: (a > 0.5)
360 Out[46]:
361 array([[ True,  True,  True,  True],
362        [ True,  True,  True,  True],
363        [ True,  True,  True,  True],
364        [ True,  True,  True,  True],
365        [ True,  True,  True,  True]])
366
367 In [47]: np.nonzero(a > 0.5)
```

```
368 Out[47]:
369 (array([0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4]),
370  array([0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3]))
371
372 In [48]: v = np.array([1, 0, 0, 1])
373
374 In [49]: a[:,np.nonzero(v > 0.5)[0]]
375 Out[49]:
376 array([[ 1,  4],
377        [ 5,  8],
378        [ 9, 12],
379        [13, 16],
380        [17, 20]])
381
382 In [50]: a[:, v.T > 0.5]
383 Out[50]:
384 array([[ 1,  4],
385        [ 5,  8],
386        [ 9, 12],
387        [13, 16],
388        [17, 20]])
389
390 In [51]: a = a/10
391
392 In [52]: a[a < 0.5]=0
393
394 In [53]: a
395 Out[53]:
396 array([[0. , 0. , 0. , 0. ],
397        [0.5, 0.6, 0.7, 0.8],
398        [0.9, 1. , 1.1, 1.2],
399        [1.3, 1.4, 1.5, 1.6],
400        [1.7, 1.8, 1.9, 2. ]])
401
402 In [54]: a =
403 ↪ np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16],[17,18,19,20]])
404
405 In [55]: a = a/10
406
407 In [56]: a * (a > 0.5)
408 Out[56]:
409 array([[0. , 0. , 0. , 0. ],
410        [0. , 0.6, 0.7, 0.8],
411        [0.9, 1. , 1.1, 1.2],
412        [1.3, 1.4, 1.5, 1.6],
413        [1.7, 1.8, 1.9, 2. ]])
```

```
414 In [57]: a[:] = 3
415
416 In [58]: a
417 Out[58]:
418 array([[3., 3., 3., 3.],
419        [3., 3., 3., 3.],
420        [3., 3., 3., 3.],
421        [3., 3., 3., 3.],
422        [3., 3., 3., 3.]])
423
424 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]])
425
426 In [60]: y = x.copy()
427
428 In [61]: y
429 Out[61]:
430 array([[ 1,  2,  3,  4],
431        [ 5,  6,  7,  8],
432        [ 9, 10, 11, 12],
433        [13, 14, 15, 16],
434        [17, 18, 19, 20]])
435
436 In [62]: y = x[1, :].copy()
437
438 In [63]: y
439 Out[63]: array([5, 6, 7, 8])
440
441 In [64]: y = x.flatten()
442
443 In [65]: y
444 Out[65]:
445 array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
446        18, 19, 20])
447
448 In [66]: np.arange(1., 11.)
449 Out[66]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
450
451 In [67]: np.r_[1.:11.]
452 Out[67]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
453
454 In [68]: np.r_[1:10:10j]
455 Out[68]: array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
456
457 In [69]: np.arange(10.)
458 Out[69]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
459
```

```
460 In [70]: np.r_[0:]
461 Out[70]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
462
463 In [71]: np.r_[0:10j]
464 Out[71]: array([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
465
466 In [72]: np.arange(1.,11.)[:, np.newaxis]
467 Out[72]:
468 array([[ 1.],
469        [ 2.],
470        [ 3.],
471        [ 4.],
472        [ 5.],
473        [ 6.],
474        [ 7.],
475        [ 8.],
476        [ 9.],
477        [10.]])
478
479 In [73]: np.zeros((3, 4))
480 Out[73]:
481 array([[0., 0., 0., 0.],
482        [0., 0., 0., 0.],
483        [0., 0., 0., 0.]])
484
485 In [74]: np.zeros((3, 4, 5))
486 Out[74]:
487 array([[[0., 0., 0., 0., 0.],
488        [0., 0., 0., 0., 0.],
489        [0., 0., 0., 0., 0.],
490        [0., 0., 0., 0., 0.]],
491
492       [[0., 0., 0., 0., 0.],
493        [0., 0., 0., 0., 0.],
494        [0., 0., 0., 0., 0.],
495        [0., 0., 0., 0., 0.]],
496
497       [[0., 0., 0., 0., 0.],
498        [0., 0., 0., 0., 0.],
499        [0., 0., 0., 0., 0.],
500        [0., 0., 0., 0., 0.]])
501
502 In [75]: np.ones((3, 4))
503 Out[75]:
504 array([[1., 1., 1., 1.],
505        [1., 1., 1., 1.],
506        [1., 1., 1., 1.]])
```

```
507
508 In [76]: np.eye(3)
509 Out[76]:
510 array([[1., 0., 0.],
511        [0., 1., 0.],
512        [0., 0., 1.]])
513
514 In [77]: a = np.array([[3,2],[2,1]])
515
516 In [78]: np.diag(a)
517 Out[78]: array([3, 1])
518
519 In [79]: np.diag(v, 0)
520 Out[79]:
521 array([[1, 0, 0, 0],
522        [0, 0, 0, 0],
523        [0, 0, 0, 0],
524        [0, 0, 0, 1]])
525
526 In [80]: rng = default_rng(42)
527
528 In [81]: rng.random((3, 4))
529 Out[81]:
530 array([[0.77395605, 0.43887844, 0.85859792, 0.69736803],
531        [0.09417735, 0.97562235, 0.7611397 , 0.78606431],
532        [0.12811363, 0.45038594, 0.37079802, 0.92676499]])
533
534 In [82]: np.linspace(1,3,4)
535 Out[82]: array([1.          , 1.66666667, 2.33333333, 3.          ])
536
537 In [83]: np.mgrid[0:9.,0:6.]
538 Out[83]:
539 array([[[0., 0., 0., 0., 0., 0.],
540        [1., 1., 1., 1., 1., 1.],
541        [2., 2., 2., 2., 2., 2.],
542        [3., 3., 3., 3., 3., 3.],
543        [4., 4., 4., 4., 4., 4.],
544        [5., 5., 5., 5., 5., 5.],
545        [6., 6., 6., 6., 6., 6.],
546        [7., 7., 7., 7., 7., 7.],
547        [8., 8., 8., 8., 8., 8.]],
548
549        [[0., 1., 2., 3., 4., 5.],
550         [0., 1., 2., 3., 4., 5.],
551         [0., 1., 2., 3., 4., 5.],
552         [0., 1., 2., 3., 4., 5.],
553         [0., 1., 2., 3., 4., 5.]])
```

```
554         [0., 1., 2., 3., 4., 5.],
555         [0., 1., 2., 3., 4., 5.],
556         [0., 1., 2., 3., 4., 5.],
557         [0., 1., 2., 3., 4., 5.]])
558
559 In [84]: np.meshgrid([1,2,4],[2,4,5])
560 Out[84]:
561 [array([[1, 2, 4],
562         [1, 2, 4],
563         [1, 2, 4]]),
564  array([[2, 2, 2],
565         [4, 4, 4],
566         [5, 5, 5]])]
567
568 In [85]: np.ix_([1,2,4],[2,4,5])
569 Out[85]:
570 (array([[1],
571         [2],
572         [4]]),
573  array([[2, 4, 5]]))
574
575 In [86]: m = 1
576
577 In [87]: n = 2
578
579 In [88]: np.tile(a, (m, n))
580 Out[88]:
581 array([[3, 2, 3, 2],
582        [2, 1, 2, 1]])
583
584 In [89]: b = np.array([[2, 3],[0,5]])
585
586 In [90]: np.concatenate((a,b),1)
587 Out[90]:
588 array([[3, 2, 2, 3],
589        [2, 1, 0, 5]])
590
591 In [91]: np.hstack((a,b))
592 Out[91]:
593 array([[3, 2, 2, 3],
594        [2, 1, 0, 5]])
595
596 In [92]: np.column_stack((a,b))
597 Out[92]:
598 array([[3, 2, 2, 3],
599        [2, 1, 0, 5]])
600
```

```
601 In [93]: np.c_[a,b]
602 Out[93]:
603 array([[3, 2, 2, 3],
604        [2, 1, 0, 5]])
605
606 In [94]: np.concatenate((a,b))
607 Out[94]:
608 array([[3, 2],
609        [2, 1],
610        [2, 3],
611        [0, 5]])
612
613 In [95]: np.vstack((a,b))
614 Out[95]:
615 array([[3, 2],
616        [2, 1],
617        [2, 3],
618        [0, 5]])
619
620 In [96]: np.r_[a,b]
621 Out[96]:
622 array([[3, 2],
623        [2, 1],
624        [2, 3],
625        [0, 5]])
626
627 In [97]: a.max()
628 Out[97]: 3
629
630 In [98]: np.nanmax(a)
631 Out[98]: 3
632
633 In [99]: a.max(0)
634 Out[99]: array([3, 2])
635
636 In [100]: a.max(1)
637 Out[100]: array([3, 2])
638
639 In [101]: np.maximum(a, b)
640 Out[101]:
641 array([[3, 3],
642        [2, 5]])
643
644 In [102]: v
645 Out[102]: array([1, 0, 0, 1])
646
647 In [103]: np.sqrt(v @ v)
```

```
648 Out[103]: 1.4142135623730951
649
650 In [104]: np.linalg.norm(v)
651 Out[104]: 1.4142135623730951
652
653 In [105]: np.logical_and(a,b)
654 Out[105]:
655 array([[ True,  True],
656        [False,  True]])
657
658 In [106]: np.logical_or(a,b)
659 Out[106]:
660 array([[ True,  True],
661        [ True,  True]])
662
663 In [107]: a & b
664 Out[107]:
665 array([[2, 2],
666        [0, 1]])
667
668 In [108]: a | b
669 Out[108]:
670 array([[3, 3],
671        [2, 5]])
672
673 In [109]: np.linalg.inv(a)
674 Out[109]:
675 array([[ -1.,  2.],
676        [ 2., -3.]])
677
678 In [110]: np.linalg.pinv(a)
679 Out[110]:
680 array([[ -1.,  2.],
681        [ 2., -3.]])
682
683 In [111]: np.linalg.matrix_rank(a)
684 Out[111]: 2
685
686 In [112]: a
687 Out[112]:
688 array([[3, 2],
689        [2, 1]])
690
691 In [113]: b = np.array([[1],[1]])
692
693 In [114]: np.linalg.solve(a, b)
694 Out[114]:
```



```

695 array([[ 1.],
696         [-1.]])
697
698 In [115]: b = b.T
699
700 In [116]: np.linalg.solve(a.T, b.T).T
701 Out[116]: array([[ 1., -1.]])
702
703 In [117]: a = np.array([[5,7,6,5],[7,10,8,7],[6,8,10,9],[5,7,9,10]])
704
705 In [118]: U, S, Vh = np.linalg.svd(a)
706
707 In [119]: U
708 Out[119]:
709 array([[ -0.38026207, -0.39630556, -0.09330504, -0.83044375],
710        [ -0.52856785, -0.61486128,  0.30165233,  0.50156506],
711        [ -0.55195485,  0.27160104, -0.76031843,  0.2085536 ],
712        [ -0.52092478,  0.62539618,  0.56764067, -0.12369746]])
713
714 In [120]: S
715 Out[120]: array([3.02886853e+01, 3.85805746e+00, 8.43107150e-01, 1.01500484e-02])
716
717 In [121]: Vh.T
718 Out[121]:
719 array([[ -0.38026207, -0.39630556, -0.09330504, -0.83044375],
720        [ -0.52856785, -0.61486128,  0.30165233,  0.50156506],
721        [ -0.55195485,  0.27160104, -0.76031843,  0.2085536 ],
722        [ -0.52092478,  0.62539618,  0.56764067, -0.12369746]])
723
724 In [122]: S**2
725 Out[122]: array([9.17404460e+02, 1.48846073e+01, 7.10829666e-01, 1.03023482e-04])
726
727 In [123]: Vh@Vh.T
728 Out[123]: array([[ 1.00000000e+00, -7.05395086e-17,  4.76007668e-17,
729                    -1.07622236e-16],
730                  [-7.05395086e-17,  1.00000000e+00,  2.11109113e-16,
731                    2.12834394e-16],
732                  [ 4.76007668e-17,  2.11109113e-16,  1.00000000e+00,
733                    -1.38049606e-16],
734                  [-1.07622236e-16,  2.12834394e-16, -1.38049606e-16,
735                    1.00000000e+00]])
736
737 In [124]: U@np.diag(S)@Vh
738 Out[124]:
739 array([[ 5.,  7.,  6.,  5.],
740        [ 7., 10.,  8.,  7.],
741        [ 6.,  8., 10.,  9.]])

```

```
742         [ 5.,  7.,  9., 10.]])
743
744 In [125]: np.linalg.cholesky(a)
745 Out[125]:
746 array([[ 2.23606798e+00,  0.00000000e+00,  0.00000000e+00,
747         0.00000000e+00],
748        [ 3.13049517e+00,  4.47213595e-01,  0.00000000e+00,
749         0.00000000e+00],
750        [ 2.68328157e+00, -8.94427191e-01,  1.41421356e+00,
751         0.00000000e+00],
752        [ 2.23606798e+00, -6.32813837e-16,  2.12132034e+00,
753         7.07106781e-01]])
754
755 In [126]: D,V = np.linalg.eig(a)
756
757 In [127]: D
758 Out[127]: array([3.02886853e+01, 3.85805746e+00, 1.01500484e-02, 8.43107150e-01])
759
760 In [128]: V
761 Out[128]:
762 array([[ -0.38026207, -0.39630556, -0.83044375, -0.09330504],
763        [ -0.52856785, -0.61486128,  0.50156506,  0.30165233],
764        [ -0.55195485,  0.27160104,  0.2085536 , -0.76031843],
765        [ -0.52092478,  0.62539618, -0.12369746,  0.56764067]])
766
767 In [129]: b = np.eye(4)
768
769 In [130]: D,V = scipy.linalg.eig(a, b)
770
771 In [131]: D
772 Out[131]:
773 array([3.02886853e+01+0.j, 3.85805746e+00+0.j, 1.01500484e-02+0.j,
774        8.43107150e-01+0.j])
775
776 In [132]: V
777 Out[132]:
778 array([[ 0.38026207, -0.39630556,  0.83044375, -0.09330504],
779        [ 0.52856785, -0.61486128, -0.50156506,  0.30165233],
780        [ 0.55195485,  0.27160104, -0.2085536 , -0.76031843],
781        [ 0.52092478,  0.62539618,  0.12369746,  0.56764067]])
782
783 In [133]: D,V = scipy.sparse.linalg.eigs(b, k=2)
784
785 In [134]: D
786 Out[134]: array([1.+0.j, 1.+0.j])
787
788 In [135]: V
```

```

789 Out[135]:
790 array([[ -0.96312067+0.j,  0.33743833+0.j],
791        [ -0.08300551+0.j, -0.31645308+0.j],
792        [ 0.01517654+0.j,  0.45922686+0.j],
793        [-0.25549625+0.j, -0.75835558 +0.j]])
794
795 In [136]: Q,R = np.linalg.qr(a)
796
797 In [137]: Q
798 Out[137]:
799 array([[ -0.43033148, -0.08439495, -0.36952858, -0.81923192],
800        [ -0.60246408, -0.57388564, -0.25706336,  0.49153915],
801        [ -0.51639778,  0.81019149, -0.12853168,  0.24576958],
802        [ -0.43033148, -0.08439495,  0.88365529, -0.16384638]])
803
804 In [138]: R
805 Out[138]:
806 array([[ -11.61895004, -16.18046376, -16.43866265, -15.31980079],
807        [  0.          , -0.43885373,  2.2449056 ,  2.00859974],
808        [  0.          ,  0.          ,  2.39390251,  4.03268141],
809        [  0.          ,  0.          ,  0.          , -0.08192319]])
810
811 In [139]: Q@R
812 Out[139]:
813 array([[ 5.,  7.,  6.,  5.],
814        [ 7., 10.,  8.,  7.],
815        [ 6.,  8., 10.,  9.],
816        [ 5.,  7.,  9., 10.]])
817
818 In [140]: P,L,U = scipy.linalg.lu(a)
819
820 In [141]: P
821 Out[141]:
822 array([[0., 0., 0., 1.],
823        [1., 0., 0., 0.],
824        [0., 1., 0., 0.],
825        [0., 0., 1., 0.]])
826
827 In [142]: L
828 Out[142]:
829 array([[ 1.          ,  0.          ,  0.          ,  0.          ],
830        [ 0.85714286,  1.          ,  0.          ,  0.          ],
831        [ 0.71428571,  0.25        ,  1.          ,  0.          ],
832        [ 0.71428571,  0.25        , -0.2         ,  1.          ]])
833
834 In [143]: U
835 Out[143]:

```

```

836 array([[ 7.          , 10.          ,  8.          ,  7.          ],
837         [ 0.          , -0.57142857,  3.14285714,  3.          ],
838         [ 0.          ,  0.          ,  2.5         ,  4.25        ],
839         [ 0.          ,  0.          ,  0.          ,  0.1         ]])
840
841 In [144]: P@L@U
842 Out[144]:
843 array([[ 5.,  7.,  6.,  5.],
844        [ 7., 10.,  8.,  7.],
845        [ 6.,  8., 10.,  9.],
846        [ 5.,  7.,  9., 10.]])
847
848 In [145]: scipy.sparse.linalg.cg(a,np.array([1,1,1,1]))
849 Out[145]: (array([ 20., -12., -5.,  3.]), 0)
850
851 In [146]: np.fft.fft(a)
852 Out[146]:
853 array([[23.+0.j, -1.-2.j, -1.+0.j, -1.+2.j],
854        [32.+0.j, -1.-3.j, -2.+0.j, -1.+3.j],
855        [33.+0.j, -4.+1.j, -1.+0.j, -4.-1.j],
856        [31.+0.j, -4.+3.j, -3.+0.j, -4.-3.j]])
857
858 In [147]: np.fft.ifft(a)
859 Out[147]:
860 array([[ 5.75+0.j , -0.25+0.5j , -0.25+0.j , -0.25-0.5j ],
861        [ 8.  +0.j , -0.25+0.75j, -0.5  +0.j , -0.25-0.75j],
862        [ 8.25+0.j , -1.  -0.25j, -0.25+0.j , -1.  +0.25j],
863        [ 7.75+0.j , -1.  -0.75j, -0.75+0.j , -1.  +0.75j]])
864
865 In [148]: np.sort(a)
866 Out[148]:
867 array([[ 5,  5,  6,  7],
868        [ 7,  7,  8, 10],
869        [ 6,  8,  9, 10],
870        [ 5,  7,  9, 10]])
871
872 In [149]: np.sort(a, axis=1)
873 Out[149]:
874 array([[ 5,  5,  6,  7],
875        [ 7,  7,  8, 10],
876        [ 6,  8,  9, 10],
877        [ 5,  7,  9, 10]])
878
879 In [150]: I = np.argsort(a[:, 0])
880
881 In [151]: b = a[I,:]
882

```

```
883 In [152]: b
884 Out[152]:
885 array([[ 5,  7,  6,  5],
886        [ 5,  7,  9, 10],
887        [ 6,  8, 10,  9],
888        [ 7, 10,  8,  7]])
889
890 In [153]: Z = np.array([[0,1],[1,1],[2,1],[3,1]])
891
892 In [154]: y = np.array([-1, 0.2, 0.9, 2.1])
893
894 In [155]: x = np.linalg.lstsq(Z, y,rcond=None)
895
896 In [156]: x
897 Out[156]: (array([ 1.  , -0.95]), array([0.05]), 2, array([4.10003045,
898 ↪ 1.09075677]))
899
900 In [157]: Z@x[0]
901 Out[157]: array([-0.95,  0.05,  1.05,  2.05])
902
903 In [158]: q=1
904
905 In [159]: scipy.signal.resample(y, int(np.ceil(len(y)/q)))
906 Out[159]: array([-1.  ,  0.2,  0.9,  2.1])
907
908 In [160]: np.unique(a)
909 Out[160]: array([ 5,  6,  7,  8,  9, 10])
910
911 In [161]: a = np.array([[0], [1], [2]])
912
913 In [162]: a.squeeze()
914 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.

1 In [1]: import matplotlib.pyplot as plt
2 Matplotlib is building the font cache; this may take a moment.
3
4 In [2]: plt.plot([1,2,3,4], [1,2,7,14])
5 Out[2]: [<matplotlib.lines.Line2D at 0x7fd741fc6f40>]
6
7 In [3]: plt.axis([0, 6, 0, 20])
8 Out[3]: (0.0, 6.0, 0.0, 20.0)
9
10 In [4]: plt.show()
11 2024-09-08 17:18:45.301 python[20991:3560702] +[CATransaction synchronize] called
   ↳ within transaction
12 2024-09-08 17:18:45.609 python[20991:3560702] +[CATransaction synchronize] called
   ↳ within transaction
```

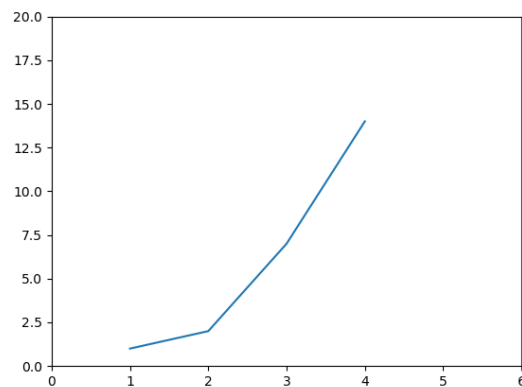


Figure 1: Output from Task 3

**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.

1 In [1]: import numpy as np
2
3 In [2]: import matplotlib.pyplot as plt
4
5 In [3]: x = np.linspace(0,100,10001)
6
7 In [4]: y = np.cos(x)*np.sqrt(x)
8
9 In [5]: plt.plot(x, y, 'r-')
10 Out[5]: [<matplotlib.lines.Line2D at 0x7fadd2bb5df0>]
11
12 In [6]: plt.grid()
13
14 In [7]: plt.show()
15 2024-09-08 18:07:00.499 python[21981:3600780] +[CATransaction synchronize] called
    ↳ within transaction
16 2024-09-08 18:07:00.823 python[21981:3600780] +[CATransaction synchronize] called
    ↳ within transaction
17 2024-09-08 18:07:08.918 python[21981:3600780] +[CATransaction synchronize] called
    ↳ within transaction
18 2024-09-08 18:07:09.789 python[21981:3600780] +[CATransaction synchronize] called
    ↳ within transaction
```

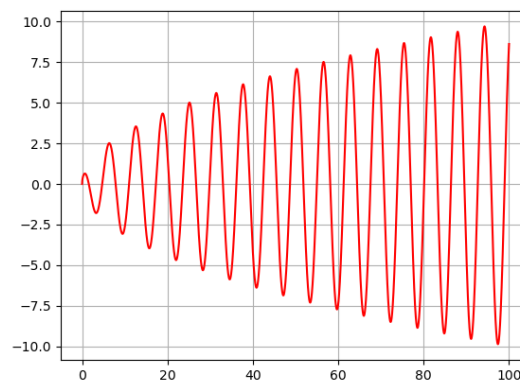


Figure 2: Output from Task 4

**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>