week09-00

March 24, 2025

1 Sorting...

[1]:
```
import numpy as np

L = np.random.rand(20)
L
```

[1]:
```
array([0.15496462, 0.43771374, 0.7691895 , 0.07377782, 0.79910591,
       0.56433127, 0.73494644, 0.13545665, 0.66473965, 0.19518556,
       0.21220215, 0.33412276, 0.94886737, 0.4219934 , 0.98005206,
       0.42314489, 0.67667685, 0.610882  , 0.28401761, 0.12808084])
```

[2]:
```
L.sort()
L
```

[2]:
```
array([0.07377782, 0.12808084, 0.13545665, 0.15496462, 0.19518556,
       0.21220215, 0.28401761, 0.33412276, 0.4219934 , 0.42314489,
       0.43771374, 0.56433127, 0.610882  , 0.66473965, 0.67667685,
       0.73494644, 0.7691895 , 0.79910591, 0.94886737, 0.98005206])
```

[3]:
```
np.flip(L)
```

[3]:
```
array([0.98005206, 0.94886737, 0.79910591, 0.7691895 , 0.73494644,
       0.67667685, 0.66473965, 0.610882  , 0.56433127, 0.43771374,
       0.42314489, 0.4219934 , 0.33412276, 0.28401761, 0.21220215,
       0.19518556, 0.15496462, 0.13545665, 0.12808084, 0.07377782])
```

[5]:
```
rng = np.random.default_rng()

import pprint
float_formatter = "{:.5f}".format
np.set_printoptions(formatter={'float_kind':float_formatter})

colors = [ "Gainsboro",
           "Gamboge",
           "Glossy grape",
           "Gold (metallic)",
           "Gold (Crayola)",
           "Golden poppy",
           "Golden yellow",
           "Goldenrod",
           "Gotham green",
           "Granite gray",
           "Granny Smith apple",
           "Gray (web)",
           "Gray (X11 gray)",
           "Green",
           "Green (Crayola)",
           "Green (web)",
           "Green (Munsell)",
           "Green (pigment)",
           "Green-blue",
           "Green Lizard"
           ]

vals = rng.random(20)

{ colors[i]:float(vals[i]) for i in range(20) }
```

```
[5]: {'Gainsboro': 0.06757705361621236,
      'Gamboge': 0.6814114044289292,
      'Glossy grape': 0.7277477398996349,
      'Gold (metallic)': 0.2668064570821498,
      'Gold (Crayola)': 0.7745500036614242,
      'Golden poppy': 0.19907674132921627,
      'Golden yellow': 0.3086028613701629,
      'Goldenrod': 0.43478402025095453,
      'Gotham green': 0.30007057477468146,
      'Granite gray': 0.9523168993121307,
      'Granny Smith apple': 0.1460670657546972,
      'Gray (web)': 0.7686009347303981,
      'Gray (X11 gray)': 0.0034087779127690565,
      'Green': 0.25618679306680814,
      'Green (Crayola)': 0.5852818538285638,
      'Green (web)': 0.7002443464296474,
      'Green (Munsell)': 0.24841718303543836,
      'Green (pigment)': 0.026560789524888828,
      'Green-blue': 0.3865936880142321,
      'Green Lizard': 0.6988164796183741}
```

```
[7]: vals
```

```
[7]: array([0.06758, 0.68141, 0.72775, 0.26681, 0.77455, 0.19908, 0.30860,
            0.43478, 0.30007, 0.95232, 0.14607, 0.76860, 0.00341, 0.25619,
            0.58528, 0.70024, 0.24842, 0.02656, 0.38659, 0.69882])
```

```
[6]: np.argsort(vals)
```

```
[6]: array([12, 17,  0, 10,  5, 16, 13,  3,  8,  6, 18,  7, 14,  1, 19, 15,  2,
            11,  4,  9])
```

```
[10]: np.array([ vals[i] for i in np.argsort(-vals)])[0:3]
```

```
[10]: array([0.95232, 0.77455, 0.76860])
```

```
[ ]: { colors[i]:vals[i] for i in np.argsort( - vals) }
```

```
[ ]: np.array([ colors[i] for i in np.argsort(vals) ])
```

```
[12]: np.array([ colors[i] for i in np.argsort( -vals) ])[0:3]
```

```
[12]: array(['Granite gray', 'Gold (Crayola)', 'Gray (web)'], dtype='<U18')
```

2 Example: financial markets, again briefly

Recall our model:

	bull	*bear*	*recession*
followed by bull	0.90	0.15	0.25
followed by bear	0.075	0.80	0.25
followed by recession	0.025	0.05	0.50

```
[13]: import numpy.linalg as npl

      states = [ "bull", "bear", "recession" ]
      1
      A = np.array([[0.90 , 0.15 , 0.25],
                    [0.075, 0.80 , 0.25],
                    [0.025, 0.05 , 0.50]])
```

```
[14]: vals, vecs = npl.eig(A)
      vals
```

```
[14]: array([1.00000, 0.74142, 0.45858])
```

Now, recall that the long term behavior of the market is predicted by the *normalized* 1-eigenvector.

```
[15]: w = vecs[:,0]

      # not normalized!
      (w, sum(w))
```

```
[15]: (array([0.89087, 0.44544, 0.08909]), np.float64(1.4253932901995967))
```

```
[16]: # normalized
      nw = 1/sum(w) * w
      (nw,sum(nw))
```

```
[16]: (array([0.62500, 0.31250, 0.06250]), np.float64(1.0))
```

Now we can display the market states in descending order of their long-term probabilities:

```
[18]: { states[i] : float(nw[i])  for i in np.argsort(nw) }
```

```
[18]: {'recession': 0.0625, 'bear': 0.3125, 'bull': 0.625}
```

```
[20]: def market_probabilities(A):
          vals, vecs = npl.eig(A)

          # get the 1-eigenvectors,as a  list (which should be of length 1)
          oe = [ vecs[:,i] for i in range(len(vals)) if vals[i] - 1 < 1E-5 ]
          w = oe[0]

          nw = 1/sum(w) * w

          return { states[i] : float(nw[i]) for i in np.argsort(-nw) }
```

```
[21]: market_probabilities(A)
```

```
[21]: {'bull': 0.625, 'bear': 0.3125, 'recession': 0.0625}
```

```
[22]: B = np.array([[0.80 , 0.15 , 0.25],
                    [0.075, 0.80 , 0.25],
                    [0.035, 0.05 , 0.50]])
```

```
[23]: market_probabilities(B)
```

```
[23]: {'bull': 0.5228037465286232,
       'bear': 0.3933745222671231,
       'recession': 0.0838217312042538}
```

```
[ ]:
```