Linear-Algebra-Numpy-Examples

February 8, 2018

0.1 Linear Algebra and Numpy

Examples taken from "An Introduction to Data Analysis through a Geometric Lens" (copyright: Jeff M. Phillips)

```
In [1]: import numpy as np
        from numpy import linalg as LA
```

```
In [2]: #create an array, a row vector
        v = np.array([1,2,7,5])
```

```
In [3]: v
```

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

```
In [4]: v[2]
```

```
Out[4]: 7
```

```
In [5]: #create a n=2 x d=3 matrix
        A = np.array([[3,4,3],[1,6,7]])
```

```
In [6]: A
```

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

```
In [7]: # matrix entry at specified position
        A[1,2]
```

```
Out[7]: 7
```

```
In [8]: #restrict A to a given range of rows and columns
        A[:, 1:3]
```

```
Out[8]: array([[4, 3],
               [6, 7]])
```

```
In [9]: #adding and multiplying vectors
        u = np.array([3,4,2,2])
```

```
In [10]: #elementwise add
         v+u

Out[10]: array([4, 6, 9, 7])

In [11]: #elementwise multiply
         v*u

Out[11]: array([ 3,  8, 14, 10])

In [12]: # dot product
         v.dot(u)

Out[12]: 35

In [13]: np.dot(u,v)

Out[13]: 35

In [14]: #matrix multiplication
         B = np.array([[1,2],[6,5],[3,4]])
         A.dot(B)

Out[14]: array([[36, 38],
                [58, 60]])

In [15]: x = np.array([3,4])

In [16]: B.dot(x)

Out[16]: array([11, 38, 25])

In [17]: LA.norm(v)

Out[17]: 8.8881944173155887

In [18]: # Other norms: L1
         LA.norm(v,1)

Out[18]: 15.0

In [19]: #transpose
         A.T

Out[19]: array([[3, 1],
                [4, 6],
                [3, 7]])

In [20]: A

Out[20]: array([[3, 4, 3],
                [1, 6, 7]])
```

```
In [21]: # vectors always print in row format
         x.T

Out[21]: array([3, 4])

In [22]: # vectors always print in row format
         x

Out[22]: array([3, 4])

In [23]: LA.matrix_rank(A)

Out[23]: 2

In [24]: C = np.array([[1,2],[3,5]])

In [25]: LA.inv(C)

Out[25]: array([[-5.,  2.],
                [ 3., -1.]])

In [26]: C.dot(LA.inv(C))

Out[26]: array([[ 1.,  0.],
                [ 0.,  1.]])

In [ ]:
```