In [1]:
```
import numpy as np
```

vector

In [2]:
```
x = np.array([1,2,3,4])
y = np.ones(4)
z = np.array([1,2,3,0])
```

In [3]:
```
x
```

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

In [4]:
```
y
```

Out[4]:
```
array([1., 1., 1., 1.])
```

vector inner product

In [5]:
```
x.dot(y)
```

Out[5]:
```
10.0
```

vector outer product

In [6]:
```
np.outer(x, y)
```

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

element-wise product

In [7]:
```
x*z
```

Out[7]:
```
array([1, 4, 9, 0])
```

vector norm

In [8]:
```
#l1 norm of a vector
np.linalg.norm(x, 1)
```

Out[8]:
```
10.0
```

In [9]:
```
#l2 norm of a vector
np.linalg.norm(x, 2)
```

Out[9]:
```
5.477225575051661
```

matrix multiplication

In [10]:
```
a = np.array([[1, 2, 3], [4, 5, 6]])
b = np.array([[2,3,4, 5], [3,4, 5,3], [4,3,2,4]])
```

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

Out[11]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [12]:
```
b
```

Out[12]:
```
array([[2, 3, 4, 5],
       [3, 4, 5, 3],
       [4, 3, 2, 4]])
```

In [13]:
```
a.dot(b)
```

Out[13]:
```
array([[20, 20, 20, 23],
       [47, 50, 53, 59]])
```

matrix transpose

In [14]:
```
a.T
```

Out[14]:
```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

matrix rank

In [15]:
```
np.linalg.matrix_rank(b)
```

Out[15]:
```
3
```

matrix inverse

In [16]:
```
np.linalg.inv(b.dot(b.T))
```

Out[16]:
```
array([[ 0.2037037 , -0.11904762, -0.08201058],
       [-0.11904762,  0.15306122, -0.03741497],
       [-0.08201058, -0.03741497,  0.14247921]])
```

Singular Value Decomposition

In [17]:
```
u, s, vh = np.linalg.svd(b,full_matrices=True)
```

In [18]:
```
u.shape
```

Out[18]:
```
(3, 3)
```

In [19]:
```
s.shape
```

Out[19]:
```
(3,)
```

In [20]:
```
vh.shape
```

Out[20]:
```
(4, 4)
```

solve the system of linear equations

$$\begin{aligned} 4x_1 &- 5x_2 &= -13 \\ -2x_1 &+ 3x_2 &= 9. \end{aligned}$$

In [21]:
```
A = np.array([[4, -5], [-2, 3]])
A
```

Out[21]:
```
array([[ 4, -5],
       [-2,  3]])
```

In [22]:
```
b = np.array([-13, 9])
```

In [23]:
```
x = np.linalg.solve(A, b)
```

In [24]:
```
x
```

Out[24]:
```
array([3., 5.])
```

In [ ]:
```

```