

# Lab Instructions - session 3

## Row and Column Space, Linear Maps

### Column Space and Row Space

The following code creates a figure with two subplots. In the left subplot, we plot a bunch of random 3D points in the column space of matrix **A**. The right subplot shows a set of 2D points in the row space of **A**.

**plot1.py**

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# create a 3 x 2 matrix
A = np.array([[1, 2],
              [3, 4],
              [-2,1]])

fig = plt.figure()

# A 1 by 2 subplot grid, subplot 1 (3D)
ax1 = fig.add_subplot(1,2,1, projection='3d')
ax1.set_title('column space')

for i in range(200):
    # create a random column vector
    u = np.random.randn(2,1)
    # create a point in the column space of A
    v = A @ u
    ax1.scatter(v[0,0], v[1,0], v[2,0], color='b')

# A 1 by 2 subplot grid, subplot 2 (2D)
ax2 = fig.add_subplot(1,2,2)
ax2.set_title('row space')

for i in range(200):
    # create a random row vector
    u = np.random.randn(1,3)
    # create a point in the row space of A
    v = u @ A
    ax2.plot(v[0,0], v[0,1], 'ro')
plt.show()
```

- Rotate the 3D plot. Do all the points lie in a lower-dimensional subspace?

- What is the dimension of the column space? What is the dimension of the row space?

## Task 1 - Practice vectorized coding

You have to write the above without using the `for` loops. To create an  $m$  by  $n$  (normally distributed) random matrix use `np.random.randn(m, n)`. Notice that for a 2 by  $n$  matrix `A` containing  $n$  points as its columns, you may plot the points by giving the list of the x- and y-coordinates as the first and second argument of the `plot` function respectively:

```
ax.plot(A[0,:], A[1,:], 'o')
```

Similarly, for a 3 by  $n$  matrix containing 3D points, you may use

```
ax.scatter(A[0,:], A[1,:], A[2,:])
```

Likewise, you may plot the points represented as rows of a matrix.

## Task 2

Repeat task 1 for the matrix

```
1,  2  
3,  6  
-2, -4
```

- What are the dimensions of the row and column spaces?

## Task 3

Create a 2 by 3 subplot using `fig.add_subplot(2,3,i, projection='3d')` for plotting the column and row spaces of the following 3 by 3 matrices:

```
A = 1,  2,  1,  
    2, -1, -1,  
   -1,  1, -2  
  
B = 1,  2, -3  
    3,  1,  1  
    2,  1,  0  
  
C = 1,  2, -3  
    3,  6, -9  
   -2, -4,  6
```

The row and column spaces must be plotted in the subplot's first and second rows, respectively. The columns of the subplot correspond to the matrices **A**, **B**, and **C**.

- Rotate the plots. For each matrix, what are the dimensions of the row and the column spaces?
- What can you say about the row and column spaces of a matrix?
- Plot (the points in) the row and column spaces of matrix B in the same axes using two different colours. Repeat the same for matrix C. Are the row and column spaces of matrices equal in general?

## Linear Transformations

Remember representing the shape of a face as a set of points from the previous lab. Here, we apply a linear transformation to each point.

`face1.py`

```
import matplotlib.pyplot as plt
import numpy as np
from face_data import Face1, edges

def plot_face(plt,X,edges,color='b'):
    "plots a face"
    plt.plot(X[:,0], X[:,1], 'o', color=color)

    for i,j in edges:
        xi,yi = X[i]
        xj,yj = X[j]

        plt.plot((xi,xj), (yi,yj), '-', color=color)

    plt.axis('square')
    plt.xlim(-100,100)
    plt.ylim(-100,100)

th = np.pi/6
A = np.array([[np.cos(th), np.sin(th)],
              [-np.sin(th), np.cos(th)]])

X = Face1 @ A
plot_face(plt, X, edges, color='b')
plt.show()
```

- Why does the above rotates the face counterclockwise, while the matrix **A** corresponds to a 30 degrees clockwise rotation (-30°)?

## Task 4 - Linear Transformations

- Animate the face to rotate around the origin by varying  $\theta$  from 0 to  $2\pi$ . Use what you learned from the previous lab.
- Apply a scaling transformation:

```
A = [[ α,      0  ],
      [ 0,      α  ]]
```

- Animate by varying  $\alpha$  from  $3/4$  to  $4/3$ .
- What happens when alpha is negative?

- Apply a non-uniform scaling transformation:

```
A = [[ α,      0  ],
      [ 0,      β  ]]
```

- Animate by varying  $\alpha$  from  $3/4$  to  $4/3$  and taking  $\beta = 1/\alpha$ .

- Shear the face (horizontally) by applying the transformation

```
A = [[ 1,      0  ],
      [ s,      1  ]]
```

- Animate by varying  $s$  from **-0.7** to **0.7**.
- The matrix **A** above represents a vertical shear. Why does it perform a horizontal shear here?