Course Notes: Mathematics

What Is a Matrix ?

- Collection of numbers ordered in rows and columns
- Scalars are all numbers from algebra of dimension 0, can be considered as 1_x1 matrix
- Vectors are matrices with either 1 row or 1 column depending whether we have row vector or column vector
- Matrices can be added, subtracted and multiplied

Scalars

(2) or (12)

2_x1 column vector of length 2

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

1_x3 row vector of length 3

$$\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

What Is a Matrix ?

! The length of a vector is the number of elements in it

$$A=\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

A is $\mathbf{m_x n}$ matrix with $\mathbf{a}_{ik}$ denoting the element on $\mathbf{i}^{\mathbf{th}}$ row and $\mathbf{k}^{\mathbf{th}}$ column

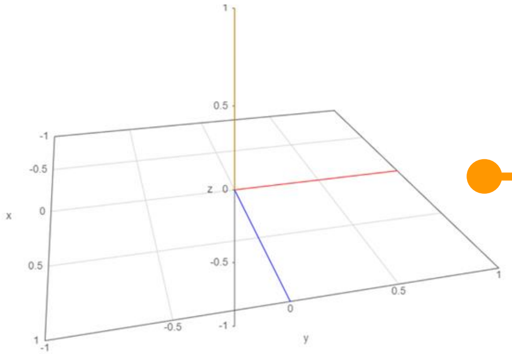
NB: When coding, indexing starts from 0, so the element in 1st row and 1st column is denoted as a_{00}

Linear Algebra and Python Implementation

- Scalars are like a point

- Vectors are lines with direction

- Matrices represent planes, any 2-dimensional plane can be represented by a matrix



$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

! The 3D plane can be represented by the matrix

Linear Algebra and Python Implementation

- The simplest and most flexible way to work with matrices in Python is by using arrays via the **NumPy** library

Vectors

```
v=np.array([5,-2,4])
v
```

```
array([ 5, -2,  4])
```

Matrices

```
m=np.array([[5,12],[-3,0]])
m
```

```
array([[ 5, 12],
       [-3,  0]])
```

Row vector

```
v.reshape(1,3)
```

```
array([[ 5, -2,  4]])
```

'reshape' gives an array a new shape without changing its data

Tensors, Addition and Subtraction of Matrices

- Tensor is a collection of matrices

Creating a tensor

```
m1=np.array([[5,12,6],[-3,0,14]])
m2=np.array([[9,8,7],[1,3,-5]])
```

```
t=np.array([m1,m2])
t
```

```
array([[[ 5, 12,  6],
        [-3,  0, 14]],

       [[ 9,  8,  7],
        [ 1,  3, -5]]])
```

Adding vectors

```
v1=np.array([1,2,3,4,5])
v2=np.array([5,4,3,2,1])
v1+v2
```

```
array([6, 6, 6, 6, 6])
```

$$\begin{pmatrix}1\\2\\3\\4\\5\end{pmatrix}+\begin{pmatrix}5\\4\\3\\2\\1\end{pmatrix}=\begin{pmatrix}6\\6\\6\\6\\6\end{pmatrix}$$

Adding vectors with a scalar

```
v1=np.array([1,2,3,4,5])
v1+1
```

```
array([2, 3, 4, 5, 6])
```

NB: In order to add and subtract matrices and vectors their forms must match

! Addition and subtraction is done elementwise for both vectors and matrices

Transpose

! The operation of transposing turns a row vector into column one and vice-versa

$$\begin{pmatrix}1\\2\\3\end{pmatrix}^T = \begin{pmatrix}1 & 2 & 3\end{pmatrix} \quad \text{and} \quad \begin{pmatrix}1 & 2 & 3\end{pmatrix}^T = \begin{pmatrix}1\\2\\3\end{pmatrix}$$

! Transposing a matrix turns each row into column

$$\begin{pmatrix}1 & 2 & 3\\4 & 5 & 6\end{pmatrix}^T = \begin{pmatrix}1 & 4\\2 & 5\\3 & 6\end{pmatrix} \quad \text{and} \quad \begin{pmatrix}1 & 4\\2 & 5\\3 & 6\end{pmatrix}^T = \begin{pmatrix}1 & 2 & 3\\4 & 5 & 6\end{pmatrix}$$

Transposing an array in Python

```
A=np.array([[5,12,6],[-3,0,14]])
A
```

```
array([[ 5, 12,  6],
       [-3,  0, 14]])
```

```
A.T
```

```
array([[ 5, -3],
       [12,  0],
       [ 6, 14]])
```

Dot Product

Dot product of vectors

```
x=np.array([2,8,-4])
y=np.array([1,-7,3])
np.dot(x,y)
```

```
-66
```

$$\begin{pmatrix} 2 \\ 8 \\ -4 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ -7 \\ 3 \end{pmatrix} = 2x1+8x(-7)+(-4)x3=-66$$

NB: We can take dot product only between vectors of equal length, the result of which is a scalar

Dot product between scalars

```
np.dot(5,6)
```

```
30
```

! This is just normal multiplication, but remember that Python interprets scalar as 1x1 arrays

Multiplication of vector by a scalar

```
x=np.array([1,2,3])
5*x
```

```
array([ 5, 10, 15])
```

! The resulting vector consists of scaled components of the initial one

Dot Product of Matrices

Multiplying matrix by a scalar

```
A=np.array([[5,12,6],[-3,0,14]])
3*A
```

```
array([[15, 36, 18],
       [-9,  0, 42]])
```

$$3*\begin{pmatrix} 5 & 12 & 6 \\ -3 & 0 & 14 \end{pmatrix} = \begin{pmatrix} 3x15 & 3x12 & 3x6 \\ 3x(-3) & 3x0 & 3x14 \end{pmatrix}$$

! Scale each element in the matrix by the scalar

Multiplication of matrices

```
A=np.array([[5,12,6],[-3,0,14]])
B=np.array([[2,-1],[8,0],[3,0]])
```

```
np.dot(A,B)
```

```
array([[124,  -5],
       [ 36,   3]])
```

NB: Can only multiply an $m_x n$ with an $n_x k$ matrix; the importance here is that the 2nd dimension of the first matrix matches the 1st dimension of the second one to give an $m_x k$ matrix

! Perform the dot product of each row with from 1st matrix with each column from the 2nd one

! Dot product of i^{th} row vector of 1st matrix with j^{th} column vector of 2nd matrix gives the element on position (i, j) in the resulting matrix