Scalars, Vectors, Matrices & Tensors

An introduction to Linear Algebra (Part 1)

Contents

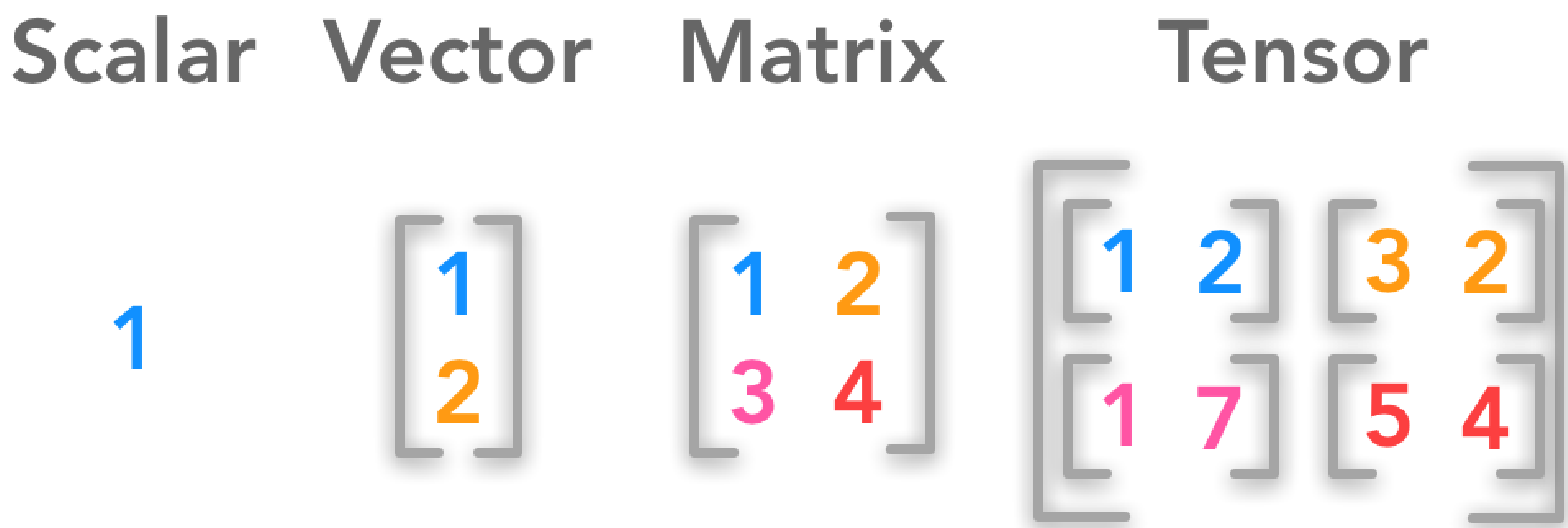
Introduction

- A scalar is a single number
- A vector is an array of numbers.
- A matrix is a 2-D array
- A tensor is a n-dimensional array with n>2

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{bmatrix}$$

Introduction



Conventions

- scalars are written in lowercase and italics. For instance: n
- vectors are written in lowercase, italics and bold type. For instance: $\mathbf{x}$
- matrices are written in uppercase, italics and bold. For instance: $\mathbf{X}$

Transposition

The superscript $^{\mathrm{T}}$ is used for transposed matrices.

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix}$$

$$\boldsymbol{A}^{\mathrm{T}} = \begin{bmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \end{bmatrix}$$

Vector Transpose

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

Matrix Transpose

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

Addition

- Matrices can be added if they have the same shape:

  A+B=C

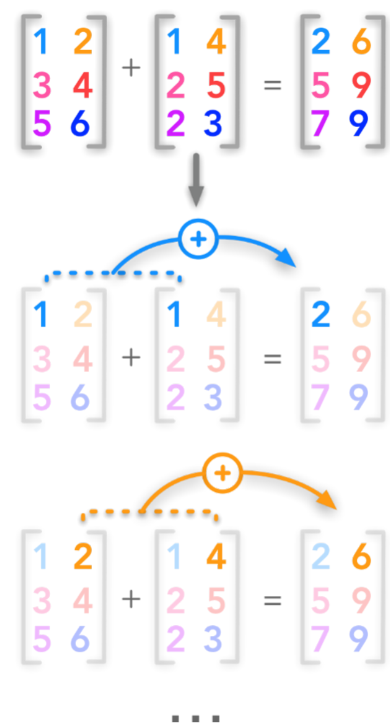
- Each cell of **A** is added to the corresponding cell of **B**:

  $\mathbf{A}_{i,j} + \mathbf{B}_{i,j} = \mathbf{C}_{i,j}$

ii is the row index and jj the column index.

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} + \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \\ B_{3,1} & B_{3,2} \end{bmatrix} = \begin{bmatrix} A_{1,1}+B_{1,1} & A_{1,2}+B_{1,2} \\ A_{2,1}+B_{2,1} & A_{2,2}+B_{2,2} \\ A_{3,1}+B_{3,1} & A_{3,2}+B_{3,2} \end{bmatrix}$$

The shape of $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are identical.



Broadcasting

- Numpy can handle operations on arrays of different shapes.
- The smaller array will be extended to match the shape of the bigger one.

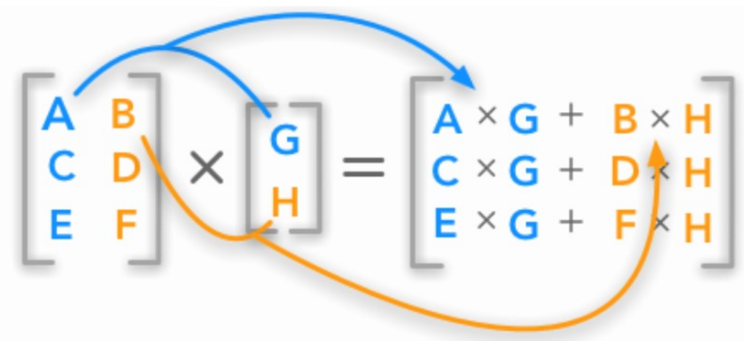
$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} + \begin{bmatrix} B_{1,1} \\ B_{2,1} \\ B_{3,1} \end{bmatrix}$$

is equivalent to

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} + \begin{bmatrix} B_{1,1} & B_{1,1} \\ B_{2,1} & B_{2,1} \\ B_{3,1} & B_{3,1} \end{bmatrix} = \begin{bmatrix} A_{1,1} + B_{1,1} & A_{1,2} + B_{1,1} \\ A_{2,1} + B_{2,1} & A_{2,2} + B_{2,1} \\ A_{3,1} + B_{3,1} & A_{3,2} + B_{3,1} \end{bmatrix}$$

Matrix Multiplication

- The standard way to multiply matrices is not to multiply each element of one with each element of the other (called the *element-wise product*) but to calculate the sum of the products between rows and columns.
- The number of columns of the first matrix must be equal to the number of rows of the second matrix.
- The matrix product, also called **dot product**, is calculated as following:



The dot product between a matrix and a vector

Matrix Multiplication

- The dot product can be formalized through the following equation:

$$C_{i,j} = A_{i,k}B_{k,j} = \sum_k A_{i,k}B_{k,j}$$

- Properties of the dot product:
  - 1. $(\boldsymbol{AB})^{\mathrm{T}} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}$
  - 2. Matrix mutliplication is distributive $\boldsymbol{A}(\boldsymbol{B}+\boldsymbol{C}) = \boldsymbol{AB}+\boldsymbol{AC}$
  - 3. Matrix mutliplication is associative $\boldsymbol{A}(\boldsymbol{BC}) = (\boldsymbol{AB})\boldsymbol{C}$
  - 4. Matrix multiplication is not commutative $\boldsymbol{AB} \neq \boldsymbol{BA}$
  - 5. Vector multiplication is commutative $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}$

System of linear equations

- A system of equations is a set of multiple equations (at least 1). For instance we could have:

$$\begin{cases} y = 2x + 1 \\ y = \frac{7}{2}x + 3 \end{cases}$$

- It is defined by its number of equations and its number of unknowns.
- In this example, there are 2 equations (the first and the second line) and 2 unknowns (x and y).
- In addition we call this a system of linear equations because each equation is linear.

Using matrices to describe the system

- Matrices can be used to describe a system of linear equations of the form **Ax=b**

$$
\begin{aligned}
&A_{1,1}x_1 + A_{1,2}x_2 + A_{1,n}x_n = b_1 \\
&A_{2,1}x_1 + A_{2,2}x_2 + A_{2,n}x_n = b_2 \\
&\cdots \\
&A_{m,1}x_1 + A_{m,2}x_2 + A_{m,n}x_n = b_n
\end{aligned}
$$

- The left-hand side can be considered as the product of a matrix **A** containing weights for each variable (n columns) and each equation (m rows):

Using matrices to describe the system

- The left-hand side can be considered as the product of a matrix **A** containing weights for each variable (n columns) and each equation (m rows):

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{bmatrix}$$

- with a vector **x** containing the n unknowns
- The dot product of **A** and **x** gives a set of equations

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$$

Using matrices to describe the system: Example

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1x_1 + 2x_2 \\ 3x_1 + 4x_2 \end{bmatrix}$$

Matrix form of a system of linear equations

Using matrices to describe the system

$$\begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{bmatrix}$$

Or simply:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

Identity matrices

- The identity matrix $\mathbf{I}_n$ is a special matrix of shape (n×n) that is filled with 0 except the diagonal that is filled with 1.

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

A 3 by 3 identity matrix

- An identity matrix can be created with the Numpy function *eye()*
- When 'apply' the identity matrix to a vector the result is this same vector:

$$\boldsymbol{I}_n \boldsymbol{x} = \boldsymbol{x}$$

Inverse Matrices

- The matrix inverse of **A** is denoted $\mathbf{A}^{-1}$. It is the matrix that results in the identity matrix when it is multiplied by **A**

$$\mathbf{A}^{-1}\mathbf{A}=\mathbf{I}_n$$

Solving a system of linear equations

- The inverse matrix can be used to solve the equation **Ax**=**b** by adding it to each term:

$$\boldsymbol{A}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

- Since we know by definition that $A^{-1}A=I$, we have:

$$\boldsymbol{I}_n\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

- We saw that a vector is not changed when multiplied by the identity matrix. So we can write:

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

Singular matrices

- Some matrices are not invertible. They are called **singular**.

Thank You!!!!