

Lab 4: Lists and loops

SUNY Korea - Francois Rameau

GitHub Classroom



Lab 4 – List and loops

Vector and matrix manipulation

Now that you know lists and loops you can start working on exciting linear algebra problems

Today, we will address two topics:

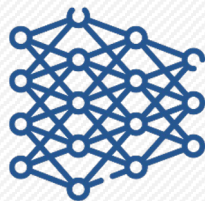
1. Dot product computation
2. Matrix averaging



Task 1: Dot product computation

The dot product is a mathematical operation that takes two vectors and produces a scalar value.

Dot product is everywhere



Machine
learning



Computer
Vision



Physics

Maybe you are already familiar with the dot product?

- Do you know what a correlation is?
- Have you ever heard of attention in AI?
- Have you ever checked the orthogonality between two vectors?

Task 1: Dot product computation

How to compute the dot product?

The dot product of two vectors of same length $a = [a_1 \ a_2 \ \cdots \ a_n]$ and $b = [b_1 \ b_2 \ \cdots \ b_n]$ can be written:

$$a \cdot b = \sum_{i=1}^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$$

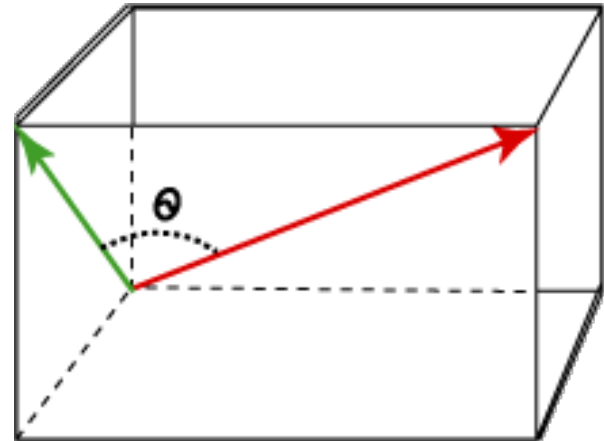
A concrete example

$$[1 \ 3 \ -5] \cdot [4 \ -2 \ -1] = (1 \times 4) + (3 \times -2) + (-5 \times -1) = 3$$

Task 1: Dot product computation

Let's compute the angle between two 3D vectors

$$\begin{aligned} a &= [a_x \quad a_y \quad a_z] \\ b &= [b_x \quad b_y \quad b_z] \\ \cos(\theta) &= \frac{a \cdot b}{|a||b|} \end{aligned}$$



Let's solve it in three steps

1. Compute the magnitude $|a|$ and $|b|$ of both vectors $|a| = \sqrt{a_x^2 + a_y^2 + a_z^2}$ ← In `magnitude(a, b)`
2. Compute the dot product between the vectors $a \cdot b$ ← In `dot(a, b)`
3. Compute the cosine distance (angle θ) with $\theta = \text{acos}\left(\frac{a \cdot b}{|a||b|}\right)$ ← In `cos_distance(a, b)`

Use `acos` from the module `math`!

Task 2: Matrix averaging

Matrix averaging, normalization or standardization is essential for many applications

1. **Numerical stability:** Normalizing matrices can prevent numerical errors when performing operations such as inversion, eigenvalue calculation, and matrix decomposition.
2. **Comparability:** Normalizing matrices can make it easier to compare them.
3. **Regularization:** Normalizing matrices can help prevent overfitting in machine learning models.
4. **Preprocessing:** Normalizing matrices is often an important preprocessing step in machine learning to improve performance.

Task 2: Matrix averaging


Now let's average a matrix using **nested loops**!

Matrix averaging

Matrix:

$$a = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,m} \end{bmatrix}$$

We will address the problem of averaging in two stages:

1. Compute the average of the matrix $\bar{a} = \frac{\sum_{i=1}^n \sum_{j=1}^m a_{i,j}}{n.m}$  **In matrix_average(a)**

2. Normalize each element by the average $a_{averaged} = \begin{bmatrix} a_{1,1}/\bar{a} & a_{1,2}/\bar{a} & \cdots & a_{1,m}/\bar{a} \\ a_{2,1}/\bar{a} & a_{2,2}/\bar{a} & \cdots & a_{2,m}/\bar{a} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}/\bar{a} & a_{n,2}/\bar{a} & \cdots & a_{n,m}/\bar{a} \end{bmatrix}$

 **In normalize_matrix(a)**