# Introduction to Math for DS Group Task 1

IMDS Group 24
Zehao Qian, Mohammad Jamshaid Iqbal, Chloe Mendez

October 27, 2023

## 1 Question 1

Consider the line L in $R^3$ passing through the origin and the point P=(2,2,1)? Which of the following points is closest to L, and which is farthest from L?

- A: (8, 9, 0.9)
- B: (4, 4, 2.1)
- C: (0.9, 0.99, 0.49)
- D: (-2, -2, 1)
- E: (0, 2.1, 0)

### 1.1 AI Answers

Since we used multiple rounds of dialogue to try to correct the answers generated by LLM, it was not aesthetically pleasing to paste into the document due to too much content. That's why we've taken the approach of sharing a history of conversations.

**ChatGPT's Answer:**
https://chat.openai.com/share/6e64d98d-8d06-4575-bf97-e6e721612e47
**Bard's Answer:**
https://g.co/bard/share/983927e7e284

### 1.2 Analytics:

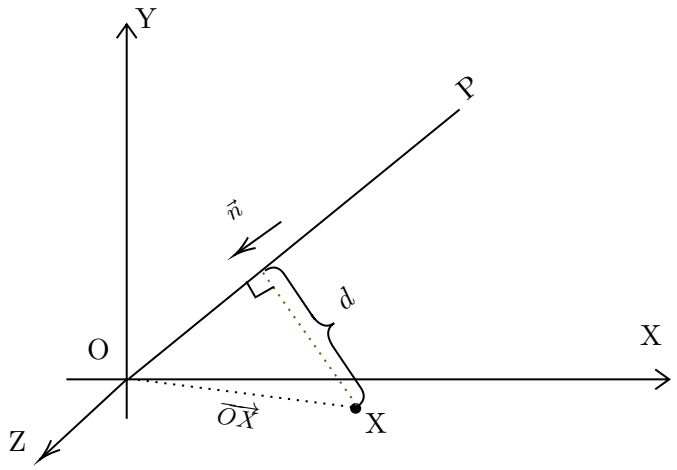
In three-dimensional space, we can calculate the distance from a point to a line using the following formula:

Let's say you have a point X with coordinates $(x_0, y_0, z_0)$, and a line defined by two points O$(0, 0, 0)$ and P$(2, 2, 1)$. The Projection formula is:

$$Proj_L X = \frac{\vec{X} \cdot \vec{OP}}{\vec{OP} \cdot \vec{OP}} * \vec{OP}$$

Then to find the distance, we will use:

$$distance = ||\vec{x} - Proj_L X||$$




### 1.3 Code for calculation

- We calculate the distance from A, B, C, D, E by using Python in a Jupyter Notebook and define a function called **point_to_line_distance**. The code is below:

```
import numpy as np

# Define the coordinates of the two points
# that define the line L
point_O = np.array([0, 0, 0])
point_P = np.array([2, 2, 1])

# Define the coordinates of points A, B, C, D, and E
point_A = np.array([8, 9, 0.9])
point_B = np.array([4, 4, 2.1])
point_C = np.array([0.9, 0.99, 0.49])
point_D = np.array([-2, -2, 1])
point_E = np.array([0, 2.1, 0])

# Function to calculate the distance from a point to a line


def point_to_line_distance(point, line_point_O, line_point_P):
    line_vector = line_point_P - line_point_O
    point_vector = point - line_point_O
    distance = np.linalg.norm(
```

```
        np.cross(line_vector, point_vector)) / np.linalg.norm(
            line_vector)
    return distance


# Calculate and print the distances
dist_A = point_to_line_distance(point_A, point_O, point_P)
dist_B = point_to_line_distance(point_B, point_O, point_P)
dist_C = point_to_line_distance(point_C, point_O, point_P)
dist_D = point_to_line_distance(point_D, point_O, point_P)
dist_E = point_to_line_distance(point_E, point_O, point_P)

print(f"Distance from point A to line L: {dist_A}")
print(f"Distance from point B to line L: {dist_B}")
print(f"Distance from point C to line L: {dist_C}")
print(f"Distance from point D to line L: {dist_D}")
print(f"Distance from point E to line L: {dist_E}")
```

The output is:

- Distance from point A to line L: 3.2365962917168947
- Distance from point B to line L: 0.09428090415820643
- Distance from point C to line L: 0.06574360974438671
- Distance from point D to line L: 1.885618083164127
- Distance from point E to line L: 1.565247584249853

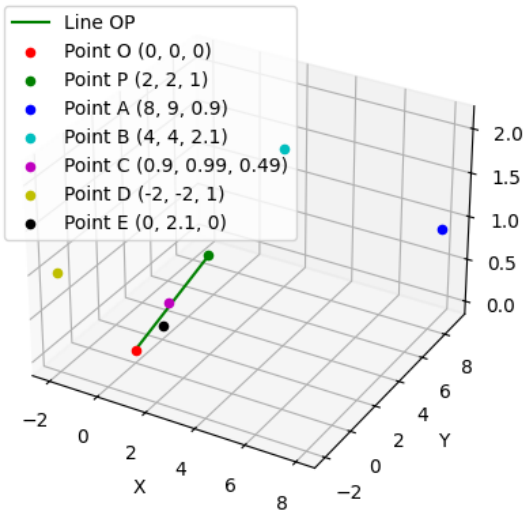



Figure 1: We draw this picture with Matplotlib in Jupyter Notebook

**So the conclusion is:** $d(A) > d(D) > d(E) > d(B) > d(C)$

### 1.4 AI Analytics

**ChatGPT** – the use of dot product in the numerator by ChatGPT is wrong for example in ProjLA, the dot product (8,9,0.9).(2,2,1) that ChatGPT is giving us is 26 whereas it should be 34.9. It made the same mistakes for all the Projections of the other points. Due to this mistake, it got the wrong projection which in turn gave the wrong distances from the defined point to the line L. ChatGPT gave the answer A to be the smallest distance and D to be the farthest. The smallest distance actually is C not A, the farthest distance is definitely D but the calculations are totally wrong.

**Bard** is giving a bit weird calculation as it is adding the point's vector by the points first number. Due to this the dot product, it is giving for each and every point is wrong and in turn the projection it gave was also wrong which resulted in taking out the wrong distances. But surprisingly the answers of the shortest distance and longest distance was right in Bard which is C and D respectively.

After asking both the chats to reconsider their answer they still were giving the same answer.

## 2 Question 2

Consider the function:

$$f(x,y) = e^{\frac{-(x^2+y^2)}{30}} \sin\frac{x^2+y^2}{5}$$

If we perform a single iteration of gradient descent starting at the point (5,-6), will we move further from the origin, or closer to it?

### 2.1 AI Answers

**ChatGPT's Answer:**
https://chat.openai.com/share/00f22dd3-31f5-4654-95fe-bae6bf2763f1
**Bard's Answer:**
https://g.co/bard/share/aae3031772cb

### 2.2 Analytics:

The gradient of $f(x,y)$:

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

1. Calculate the partial derivative of $f(x,y)$ with respect to $x$:

$$\frac{\partial f}{\partial x} = e^{\frac{-(x^2+y^2)}{30}}\left(-\frac{2x}{30}\right)\sin\frac{x^2+y^2}{5} + e^{\frac{-(x^2+y^2)}{30}}\cos\frac{x^2+y^2}{5}\left(\frac{2x}{5}\right)$$

2. Calculate the partial derivative of $f(x,y)$ with respect to $y$:

$$\frac{\partial f}{\partial y} = e^{\frac{-(x^2+y^2)}{30}}\left(-\frac{2y}{30}\right)\sin\frac{x^2+y^2}{5} + e^{\frac{-(x^2+y^2)}{30}}\cos\frac{x^2+y^2}{5}\left(\frac{2y}{5}\right)$$

The gradient at the point (5, -6):

$$\nabla f(5,-6) = \left(\frac{\partial f}{\partial x}(5,-6), \frac{\partial f}{\partial y}(5,-6)\right) = (0.26005, -0.3121)$$

Next, we calculate the negative gradient:

$$-\nabla f(5,-6) = -\left(\frac{\partial f}{\partial x}(5,-6), \frac{\partial f}{\partial y}(5,-6)\right)$$

$$a_0 = (5,-6)$$

$$a_1 = (5,-6) - \epsilon(0.26005, -0.3121)$$

Let $\epsilon = 0.01$, $a_1 = (4.9973995, -5.996879)$, so it is closer to the origin point.

## 2.3 AI Analytics

This negative gradient vector points in the direction of steepest decrease. If this vector points away from the origin, then a single iteration of gradient descent will move further from the origin. If it points towards the origin, it will move closer to the origin. You can analyze the components of the negative gradient to determine the direction.

Both ChatGPT and Bard got the final answer right but their calculations of the partial derivatives after inserting the x and y values are wrong. ChatGPT got dfx and dfy (0.0101, -0.0343) and Bard got (-2.74, -1.02) whereas the actual values are (0.26005, -0.3121).

We think that ChatGPT and Bard got it wrong because we gave them the input in natural language due to which they might have taken out the wrong derivatives which in result gave them the wrong answer for partial derivatives at (5,-6).