



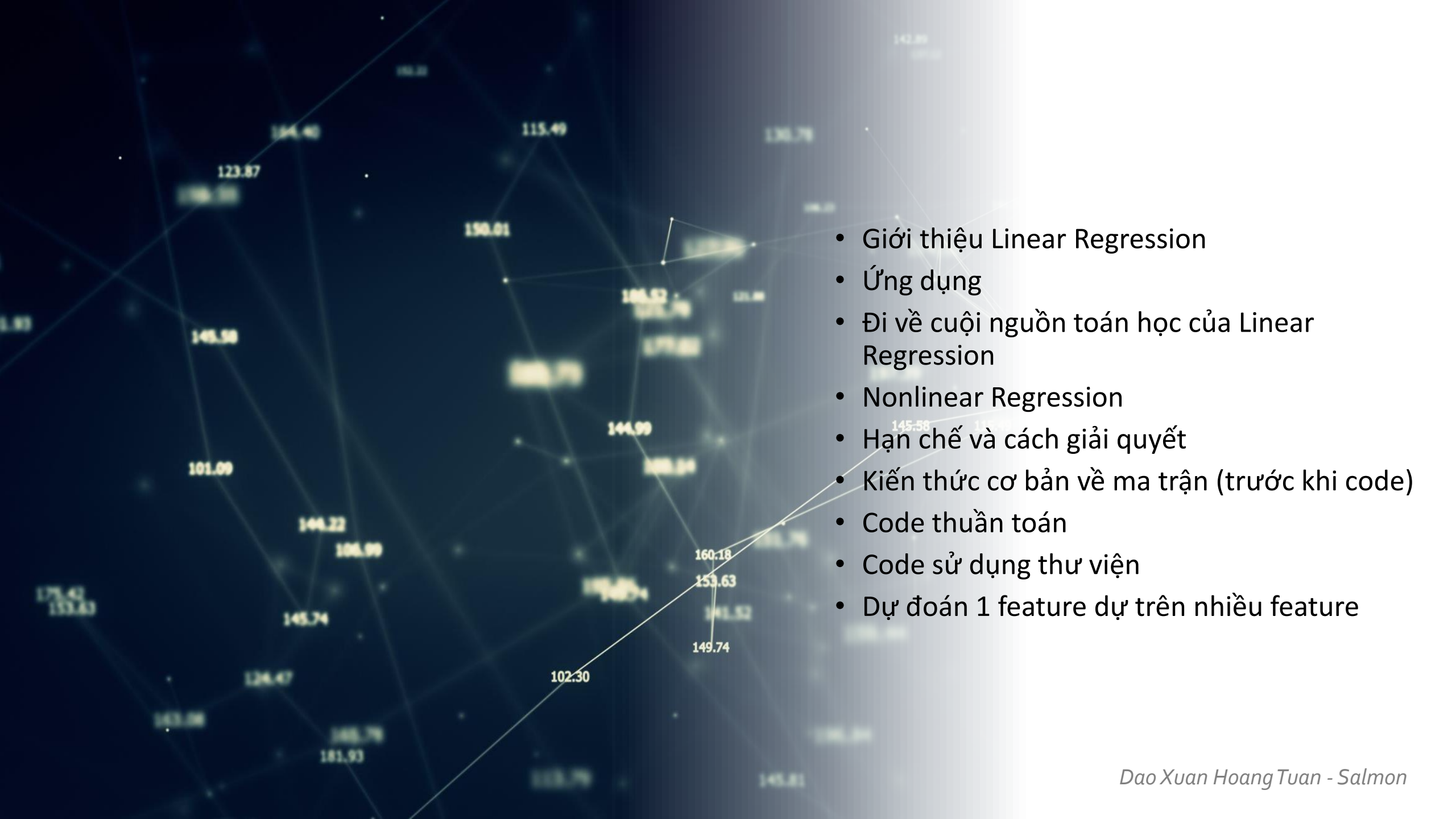
AI Faster Team

Chủ đề: Linear Regression

(Giải quyết từ gốc)

Người soạn: Đào Xuân Hoàng Tuấn

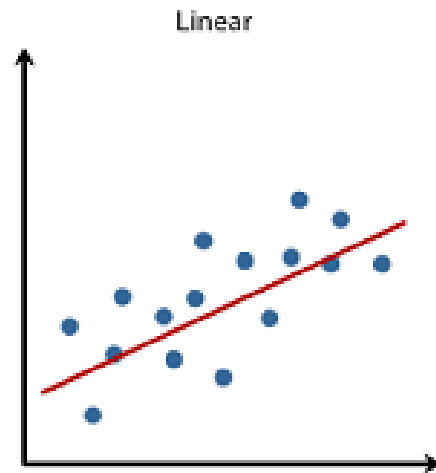
Tham khảo nhiều nguồn bao gồm Sách Linear algebra của Gilbert Strang (giáo sư đại học MIT)

- 
- Giới thiệu Linear Regression
 - Ứng dụng
 - Đi về cuối nguồn toán học của Linear Regression
 - Nonlinear Regression
 - Hạn chế và cách giải quyết
 - Kiến thức cơ bản về ma trận (trước khi code)
 - Code thuần toán
 - Code sử dụng thư viện
 - Dự đoán 1 feature dự trên nhiều feature

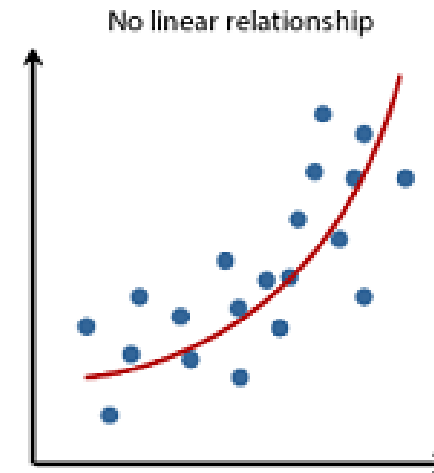
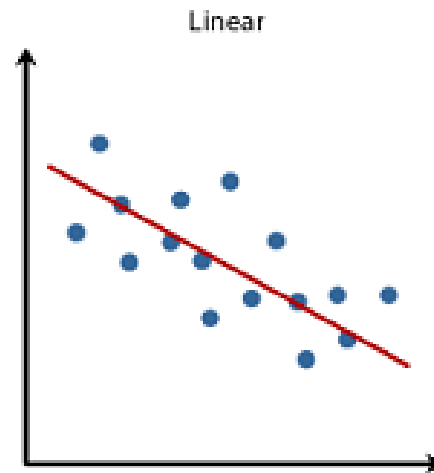
Linear Regression

(Hồi Quy Tuyến Tính)

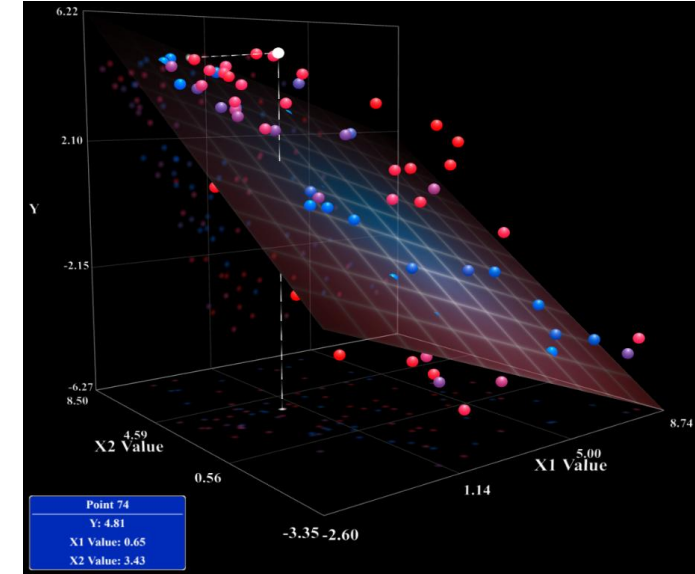
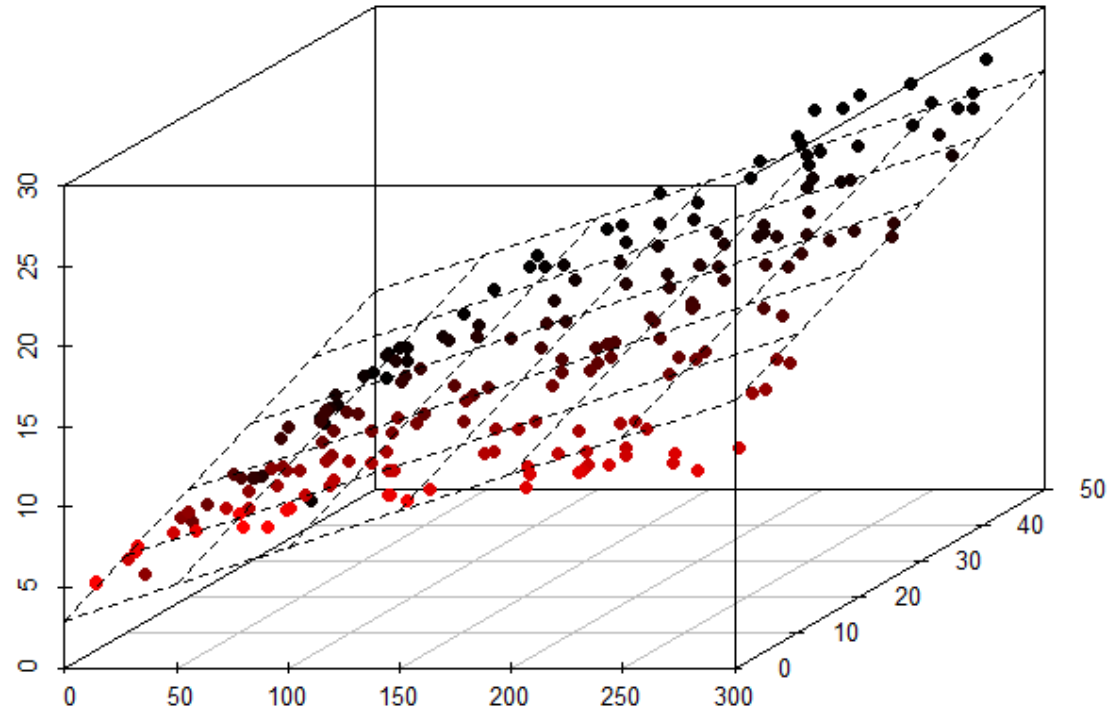
- Một trong những thuật toán cơ bản nhất của Machine Learning.
- Thuộc nhóm Supervised learning (Học có giám sát).



Linear regression



Nonlinear regression



Linear Regression 3D

Ví dụ về Linear Regression

Sinh viên

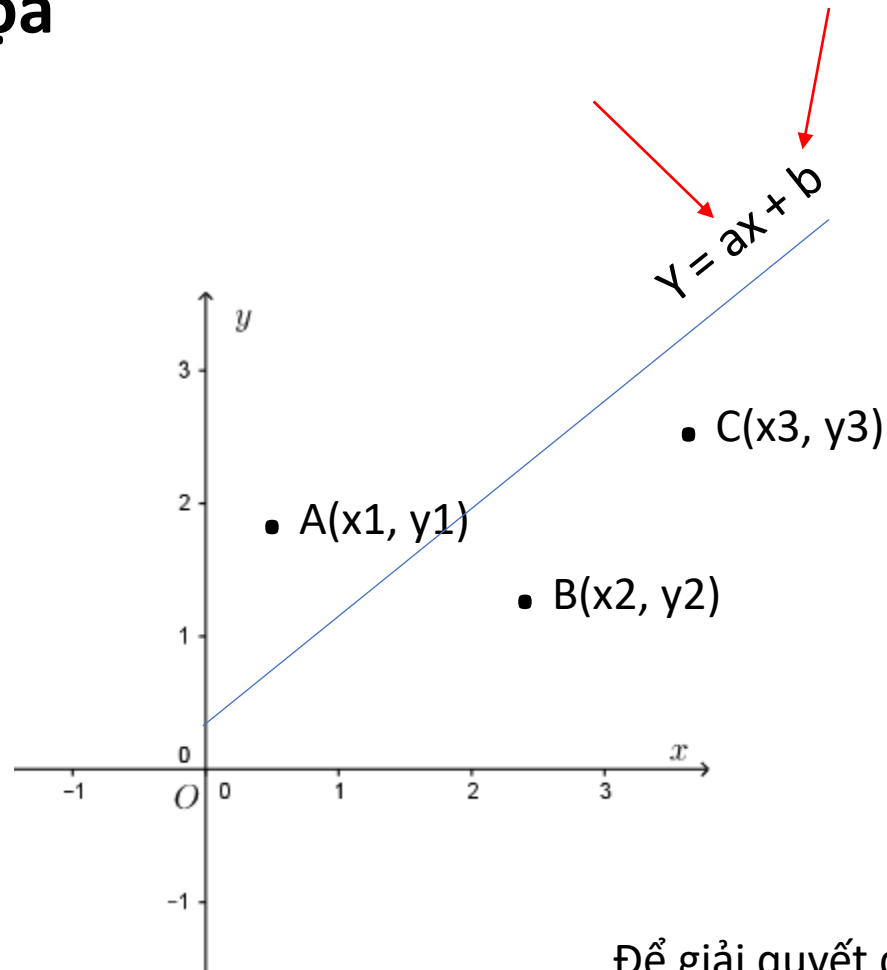
- Feature 1 (chăm chỉ)
- Feature 2 (thông minh)
- Feature 3 (ý chí)
- Feature 4 (...)
- ...
- Feature n (Điểm toán)

Linear Regression Model

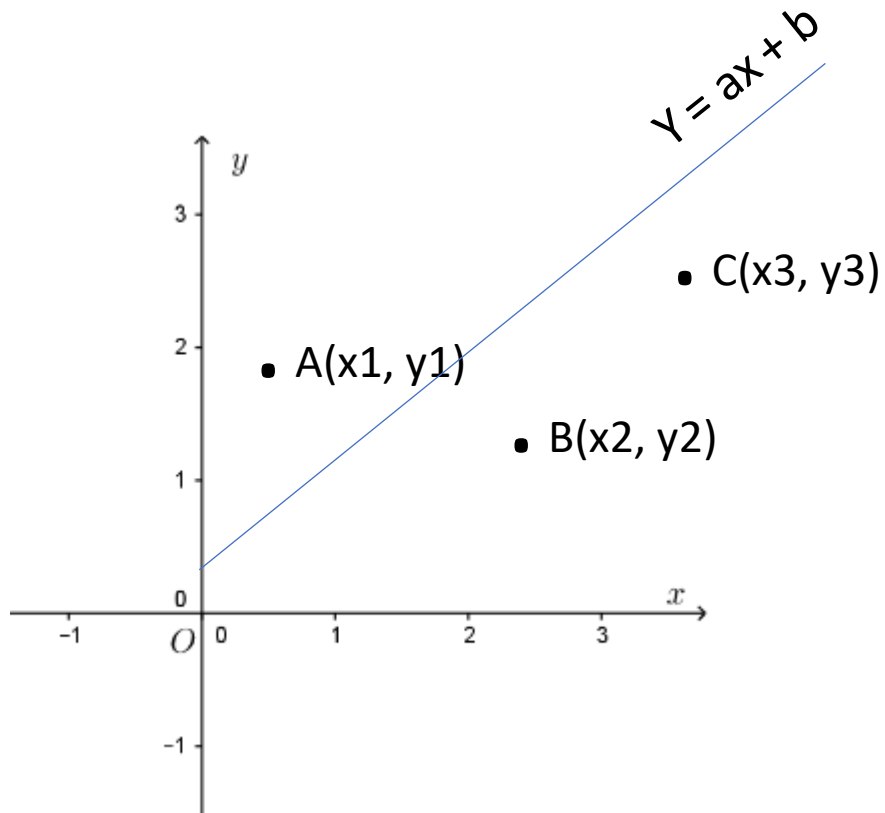
Dự đoán 1 trong số N
feature của sinh viên

Đi về cuối nguồn toán học của Linear Regression

Bài toán minh họa



Để giải quyết được bài toán
=> Thì mục đích cuối cùng là cần tìm a và b



$A \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$ Kí hiệu cho vector A (2x1) 2 dòng , 1 cột

Tương tự với B, C

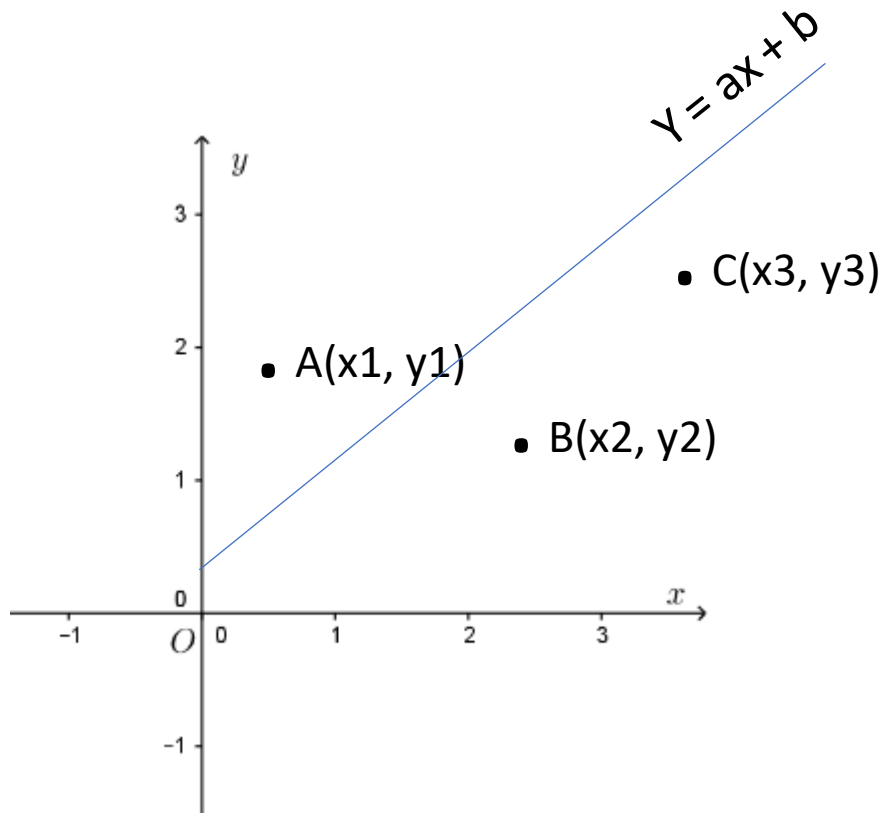
Mình giả sử có một đường thẳng đi qua 3 điểm này, thì:

$$y_1 = ax_1 + b$$

$$y_2 = ax_2 + b$$

$$y_3 = ax_3 + b$$

Nếu hệ 2 ẩn có 2 phương trình thì chắc chắn sẽ có nghiệm, tuy nhiên lại có thêm một điểm C (3 phương trình, 2 ẩn) cho nên hệ phương trình sẽ vô nghiệm
=> Vì thế chúng ta sẽ tìm đường thẳng đi gần nhất 3 điểm đấy



$$y_1 = ax_1 + b$$

$$y_2 = ax_2 + b$$

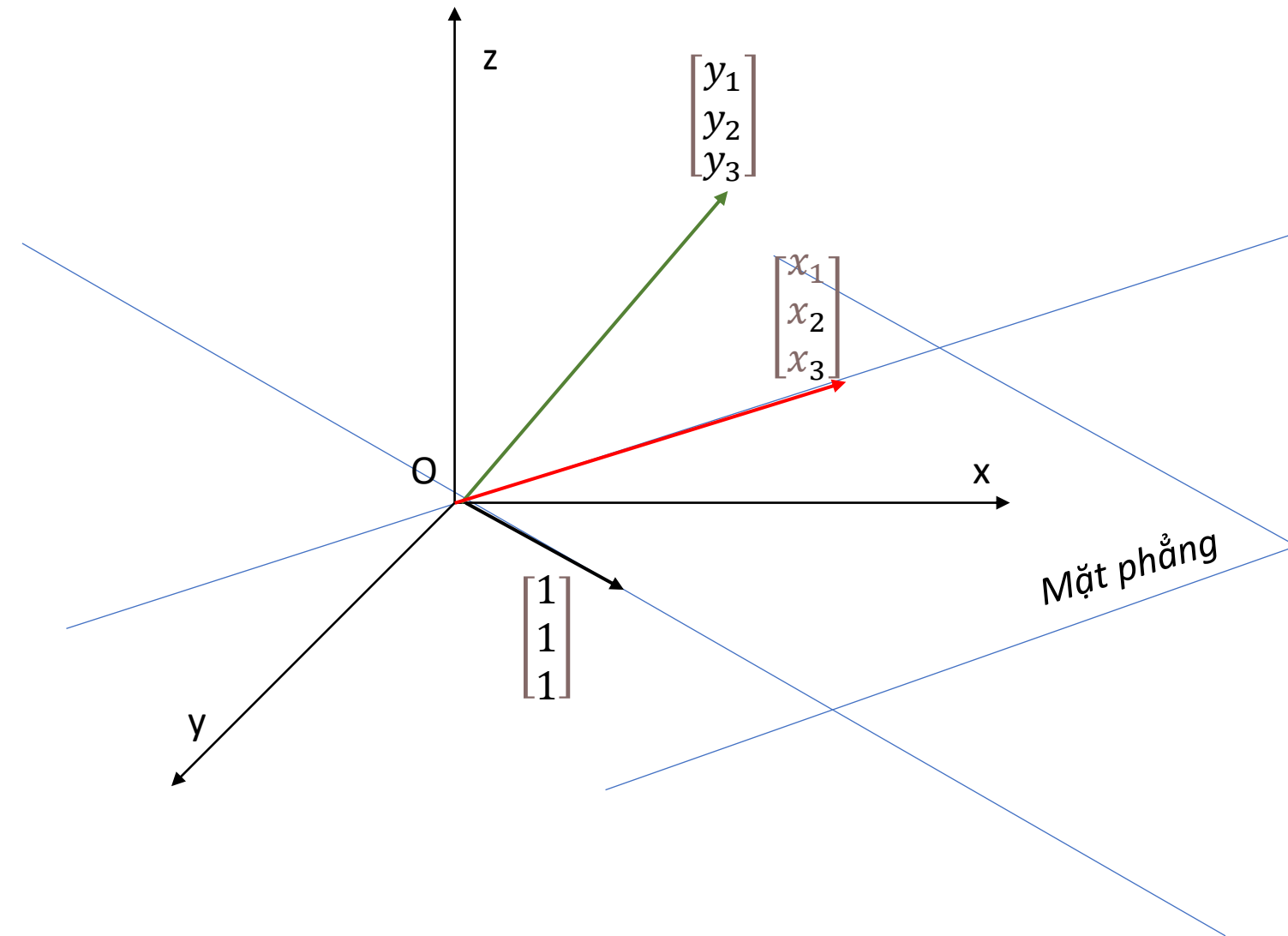
$$y_3 = ax_3 + b$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Đây là không gian 3 chiều

- Chuyển từ 3 phương trình sang 1 phương trình

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

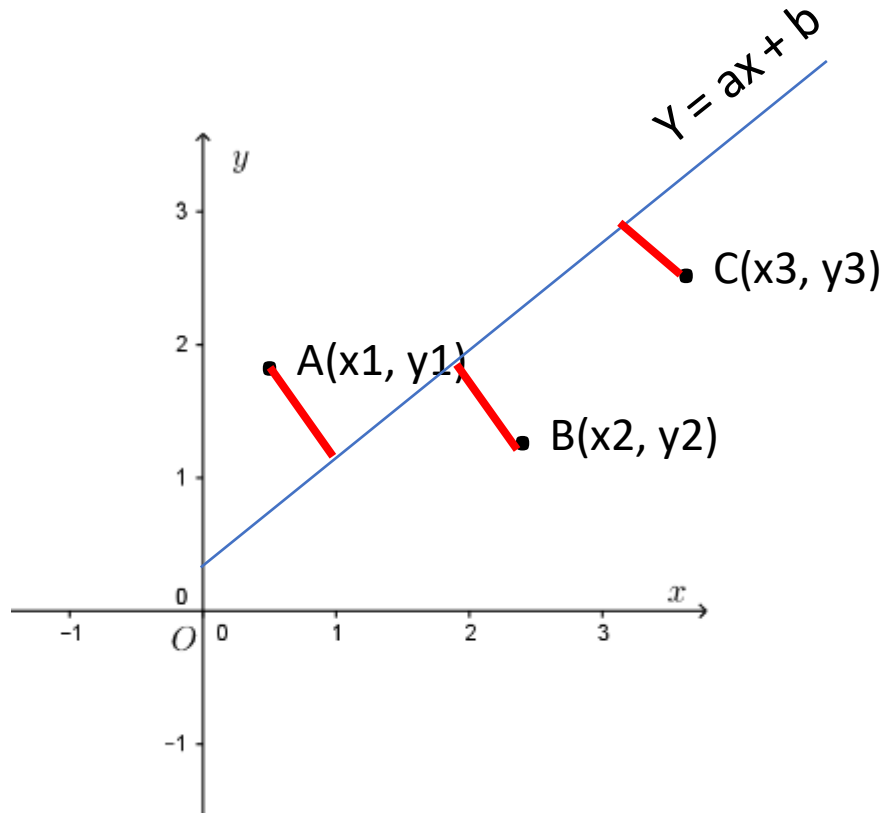


- Nhân một số với một vector
- Cộng 2 vector
- $\text{Number} * \text{vector} + \text{number} * \text{vector} = \text{mặt phẳng}$

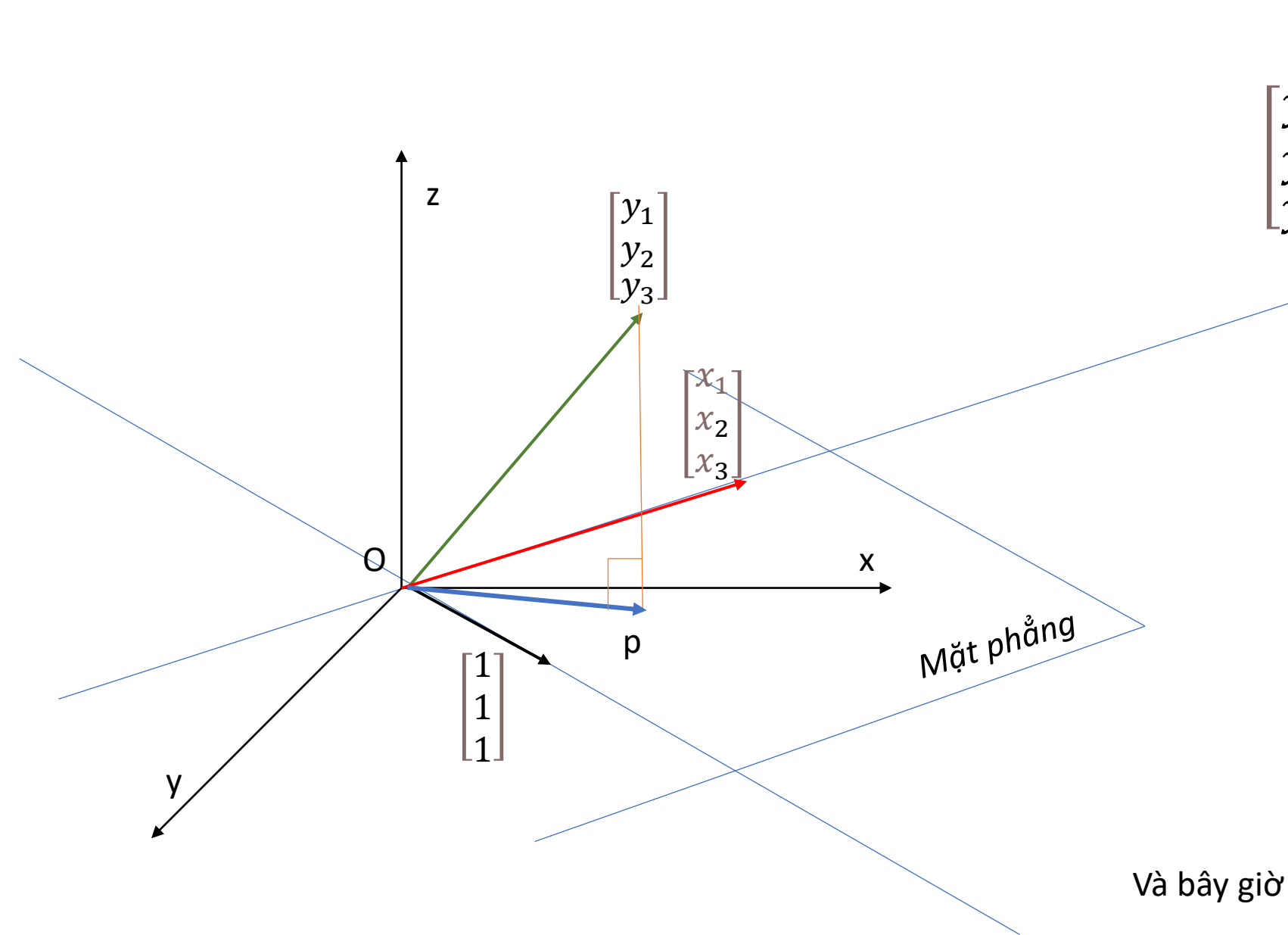
Vì vector **green** không nằm trong mặt phẳng **blue**
 => Không bao giờ có trường hợp dấu bằng xảy ra



Mẫu chốt của vấn đề



Làm sao để tổng bình phương (phương sai) Min?



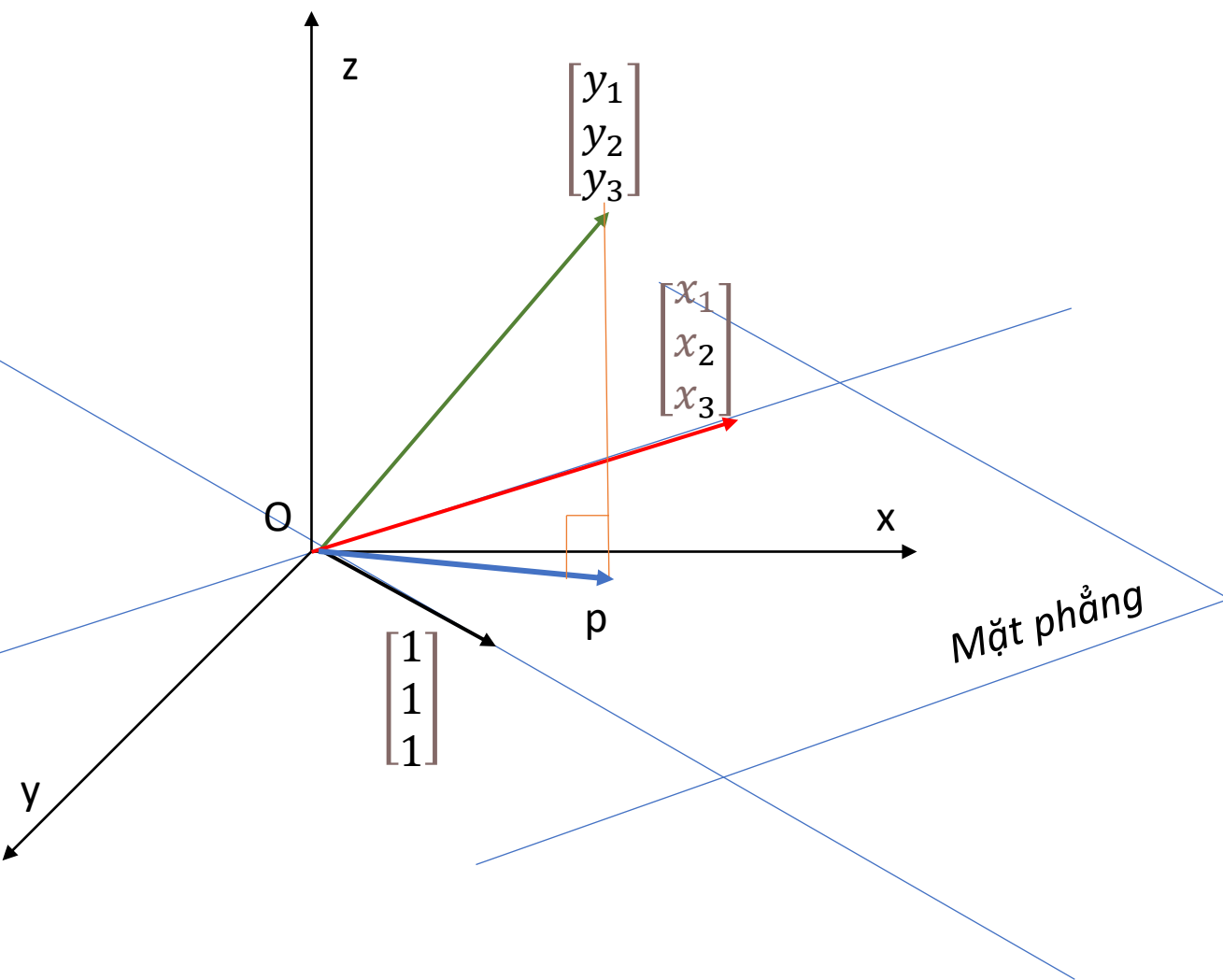
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Mặt phẳng

Giả sử vector p là vector gần vector **green** nhất thì p thì điều kiện để vector p gần vector **green** nhất là “vector p phải là hình chiếu của vector **green** xuống mặt phẳng **blue**”

Green vuông góc với p

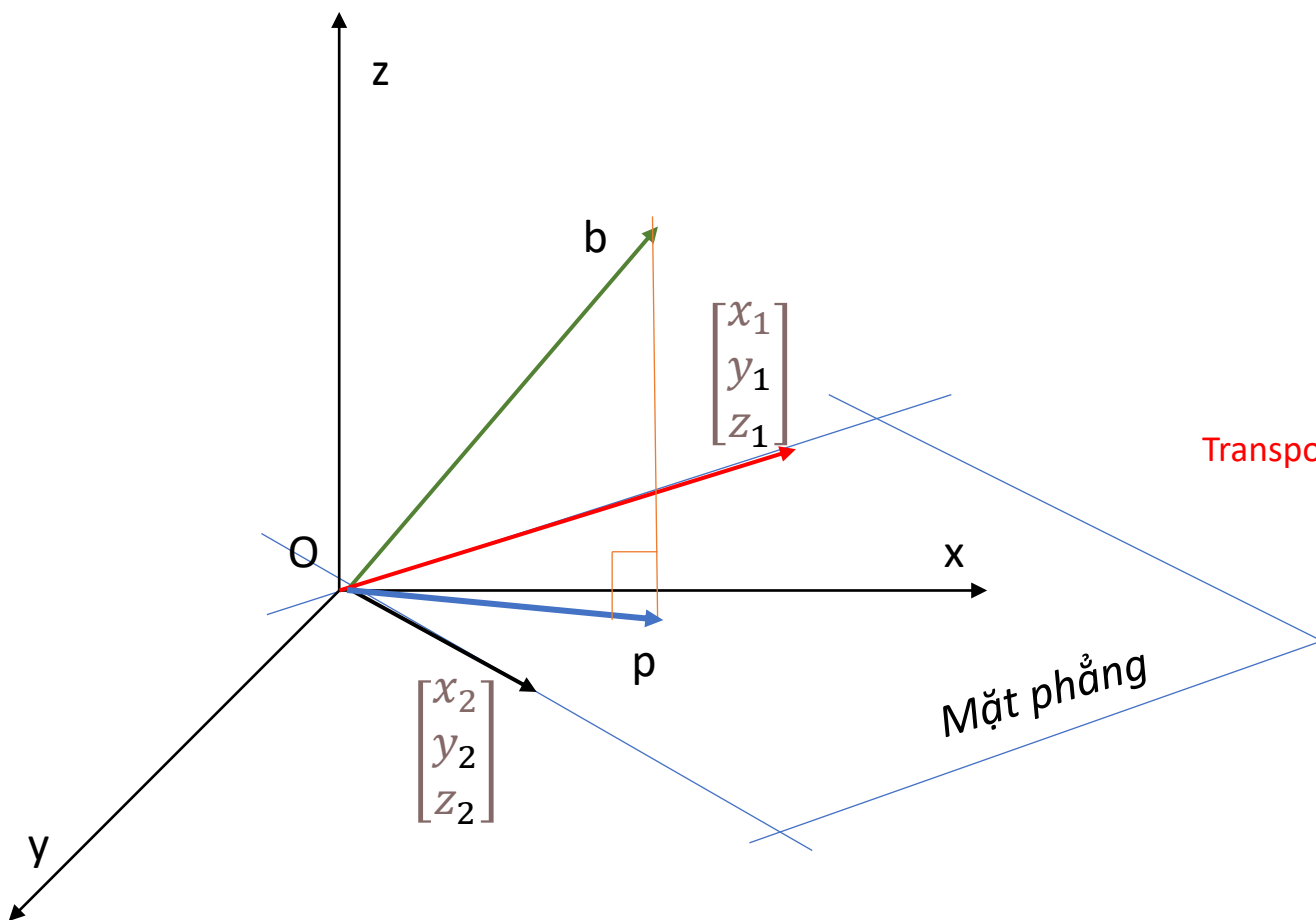
Và bây giờ vector p sẽ có dạng $a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$



Biết p sẽ tính được a , b và a , b chính là thứ mình cần tìm

*Quay về với bài toán tìm hình chiếu của vector xuống một mặt phẳng mà hồi cấp 3 chúng ta đã học
(hóa ra nó áp dụng cho machine learning 😊)*

Tìm hình chiếu...



Nếu bạn có 2 vector vuông góc với nhau thì nhân lại = 0 $\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \perp \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$

$$\text{Thì } \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = [x_1, y_1] \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = 0$$

$$= x_1x_2 + y_1y_2 = 0$$

Transpose (nghịch đảo) trong ĐSTT

Cũng có thể viết là:

$$\mathbf{a}_1 \perp \mathbf{a}_2 \Rightarrow x_1x_2 + y_1y_2 = \mathbf{a}_1^T \mathbf{a}_2 = 0$$

Kiến thức này giống như vector chỉ phương với vector pháp tuyến ngày xưa chúng ta học vậy

Nếu bạn có 2 vector vuông góc với nhau thì nhân lại = 0 $\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \perp \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$

$$\text{Thì } \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = [x_1, y_1] \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = 0$$
$$= x_1 x_2 + y_1 y_2 = 0$$

Transpose (nghịch đảo) trong ĐSTT

Cũng có thể viết là:

$$\mathbf{a}_1 \perp \mathbf{a}_2 \Rightarrow x_1 x_2 + y_1 y_2 = \mathbf{a}_1^T \mathbf{a}_2 = 0$$

Kiến thức này giống như vector chỉ phương với vector pháp tuyến ngày xưa chúng ta học vậy

Chứng minh bằng tích vô hướng

Tương tự như tích vô hướng giữa 2 vector

Theo đại số, tích vô hướng là tổng các tích tọa độ tương ứng giữa chúng.

$$\vec{u} \times \vec{v} = x_1 x_2 + y_1 y_2$$

Theo hình học, tích vô hướng là tích độ lớn của 2 vector và cos của góc giữa chúng

$$|\vec{u}| \cdot |\vec{v}| \cdot \cos(\alpha)$$

Mà $\cos(90^\circ) = 0$ nên từ những điều trên ta chắc chắn rằng 2 vector vuông góc nhân nhau = 0

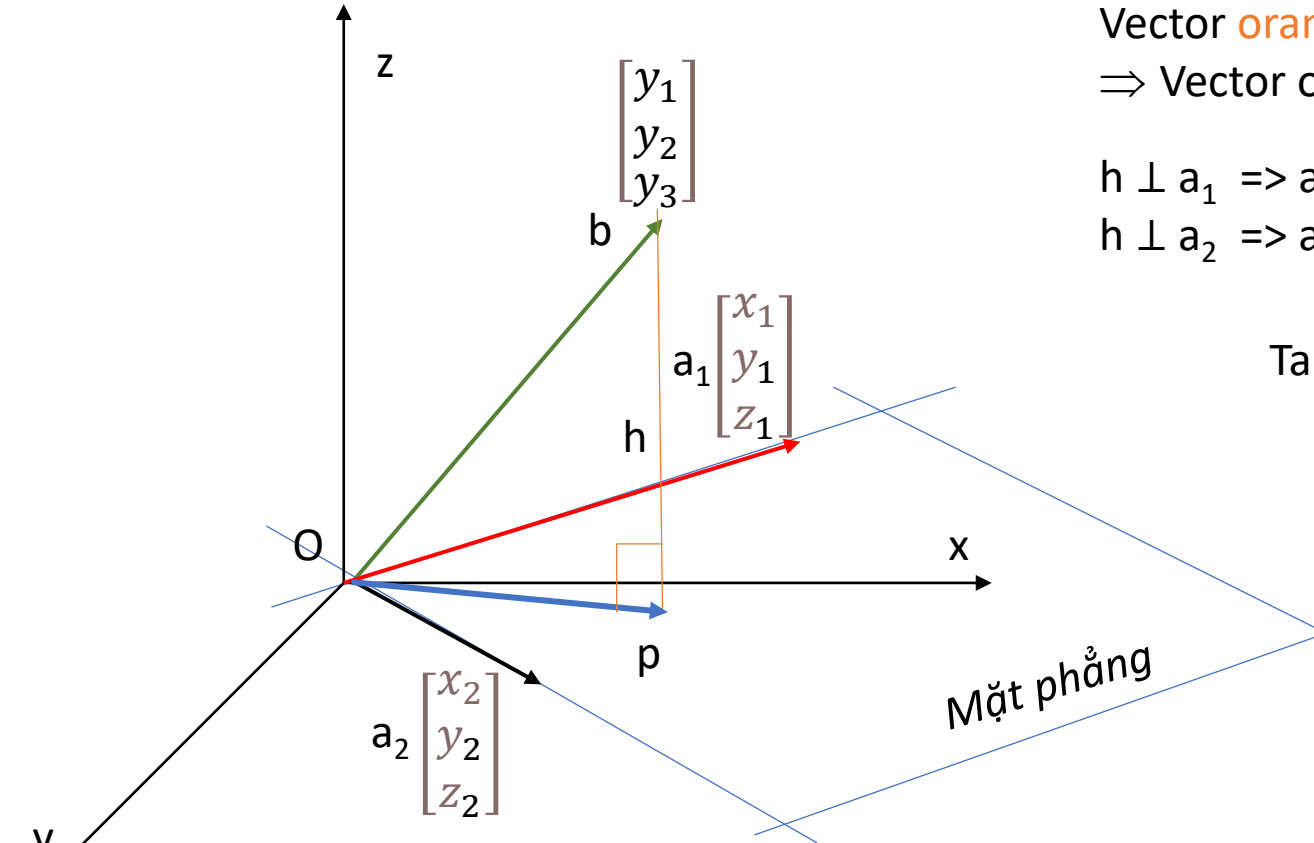
Tích vô hướng của hai vector $\mathbf{A} = [A_1, A_2, \dots, A_n]$ và $\mathbf{B} = [B_1, B_2, \dots, B_n]$ được định nghĩa như sau:^[1]

$$\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^n A_i B_i = A_1 B_1 + A_2 B_2 + \dots + A_n B_n$$

Nên cho dù có bao nhiêu chiều thì chỉ cần 2 vector đó vuông góc với nhau chúng nhân lại sẽ = 0

Tổng quát:

Áp dụng vào bài toán tìm hình chiếu...



Vector **orange** sẽ vuông góc với tất cả vector thuộc mặt phẳng **blue**
 \Rightarrow Vector orange sẽ vuông góc với a_1, a_2

$$\begin{aligned} h \perp a_1 &\Rightarrow a_1^T h = 0 & [x_1 \ y_1 \ z_1] * h &= 0 \\ h \perp a_2 &\Rightarrow a_2^T h = 0 & [x_2 \ y_2 \ z_2] * h &= 0 \end{aligned} \quad (1)$$

Ta có thể viết lại như sau

Khởi tạo ma trận A như dưới đây

$$A = [a_1 \ a_2] = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \\ z_1 & z_2 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix} \quad (2)$$

Từ (1) và (2) $\Rightarrow A^T h = 0$

Và $h = b - p$ (Quy tắc trừ vector) ($h = b + (-p) = b - p$)

$$\Rightarrow A^T(b-p) = 0$$

$$\Rightarrow A^T b - A^T p = 0$$

$$\begin{aligned} h \perp a_1 &\Rightarrow a_1^T h = 0 & [x_1 \ y_1 \ z_1] * h &= 0 \\ h \perp a_2 &\Rightarrow a_2^T h = 0 & [x_2 \ y_2 \ z_2] * h &= 0 \end{aligned} \quad (1)$$

Ta có thể viết lại như sau

$$A = [a_1 \ a_2] = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \\ z_1 & z_2 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix} \quad (2)$$

Từ (1) và (2) $\Rightarrow A^T h = 0$

Và $h = b - p$ (Quy tắc trừ vector) ($h = b + (-p) = b - p$)

$$\Rightarrow A^T(b-p) = 0$$

$$\Rightarrow A^T b - A^T p = 0$$

Vì p là vector hình chiếu nên là vector gần nhất với các điểm và có dạng như bên

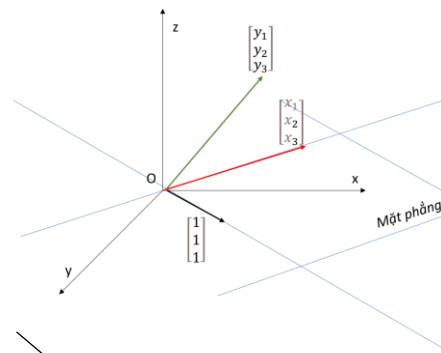
\Rightarrow Ta có thể viết **$p = Ax$**

Trong đó:

A : là ma trận (2)

x : là ma trận chứa 2 biến a, b như hình bên $x = [a \ b]$

Và ma trận x là kết quả chúng ta cần tìm cho bài toán này



Quay lại bài toán của chúng ta

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Vector nối tất cả các điểm theo giả thuyết (Không tồn tại)

Vector hình chiếu p (vector gần nhất với vector vế trái)

Và bây giờ ta có công thức:

$$A^T b - A^T A x = 0$$

$$\Rightarrow A^T b = A^T A x$$

$$\Rightarrow x = (A^T A)^{-1} A^T b \text{ (Thực hiện bước chuyển vế)}$$

(Chuyển vế ma trận nên phải sử dụng Inverse (nghịch đảo))

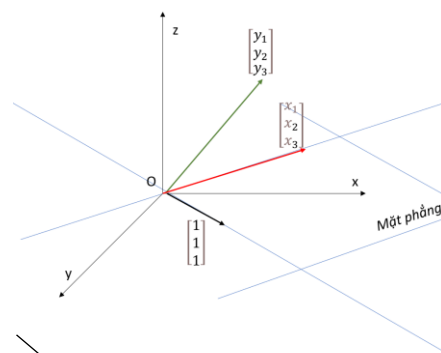
Vì trong ma trận không có chia mà chỉ có nghịch đảo

Ví dụ trực quan:

$$2x = 3y$$

$$\Leftrightarrow \frac{2x}{y} = 3$$

$$\Leftrightarrow 2x * y^{-1} = 3 \text{ (Vì } y^{-1} = 1/y \text{)}$$



Quay lại bài toán của chúng ta

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Vector nối tất cả các điểm theo giả thuyết (Không tồn tại)



Vector hình chiếu p (vector gần nhất với vector vế trái)

$$A = [a_1 \ a_2] = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \\ z_1 & z_2 \end{bmatrix}$$

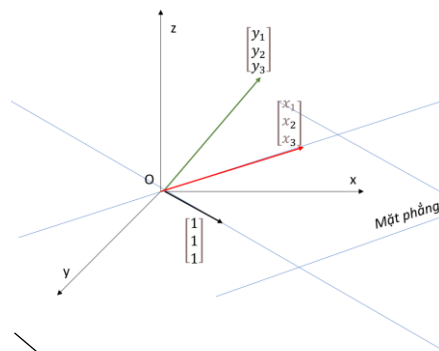
$$x = [a \ b]$$

$$\begin{aligned}
A^T b - A^T A x &= 0 \\
\Rightarrow A^T b &= A^T A x \\
\Rightarrow x &= (A^T A)^{-1} A^T b
\end{aligned}$$

Công thức cuối cùng:
 $x = (A^T A)^{-1} A^T y$

$$A = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix} \quad b \text{ chính là } y \text{ (vector giả thuyết)} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

$$x = \begin{bmatrix} a \\ b \end{bmatrix}$$



Quay lại bài toán của chúng ta

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = a \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Vector nối tất
cả các điểm
theo giả thuyết
(Không tồn tại)

Vector hình chiếu p
(vector gần nhất với vector vế trái)

Vậy đối với không gian 4 chiều, 5 chiều, 6 chiều, có tương tự như vậy không?

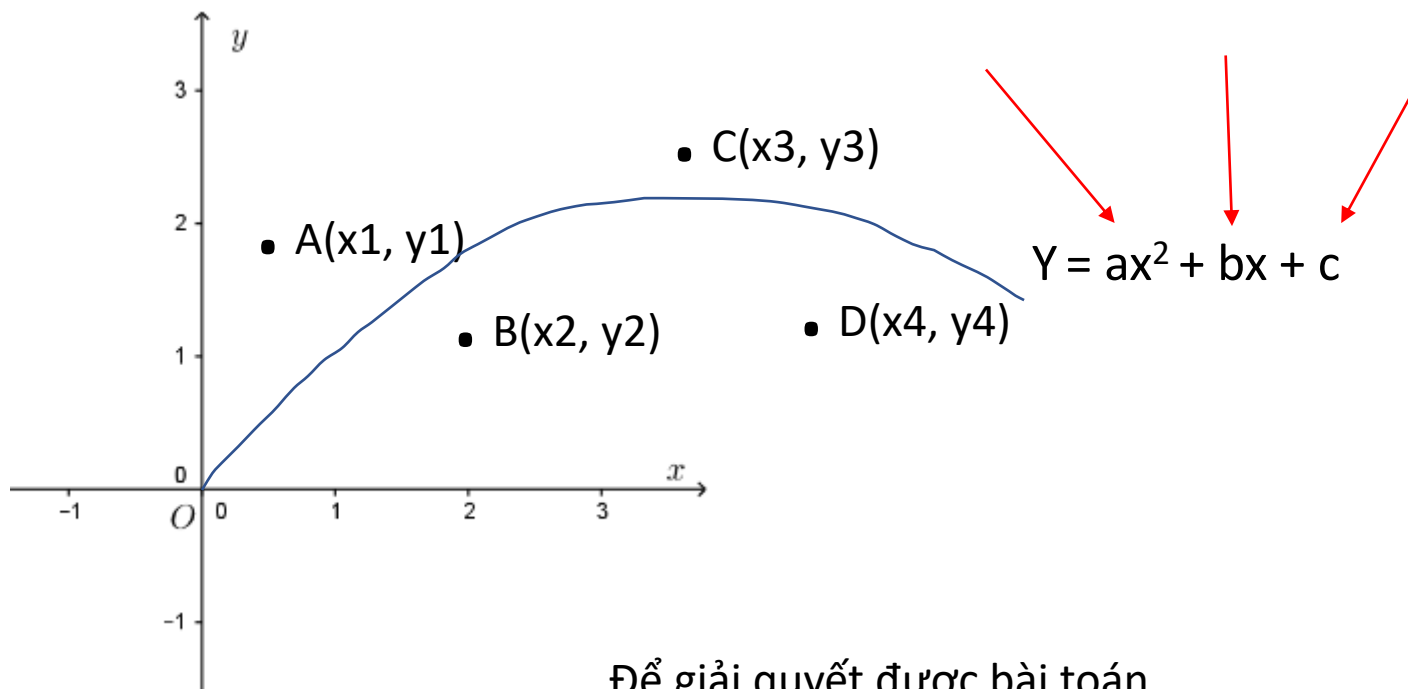
Vậy nếu không phải đường thẳng mà là đường cong?

Vậy công thức này có mặt hạn chế nào?

Tất cả sẽ được giải đáp ở những slide tiếp theo

Đối với đường cong

Mình luôn vẽ được một parabol đi qua 3 điểm, nên đối với bài toán này cần có thêm điểm D nữa



Để giải quyết được bài toán
=> Thì mục đích cuối cùng là cần tìm a, b, c

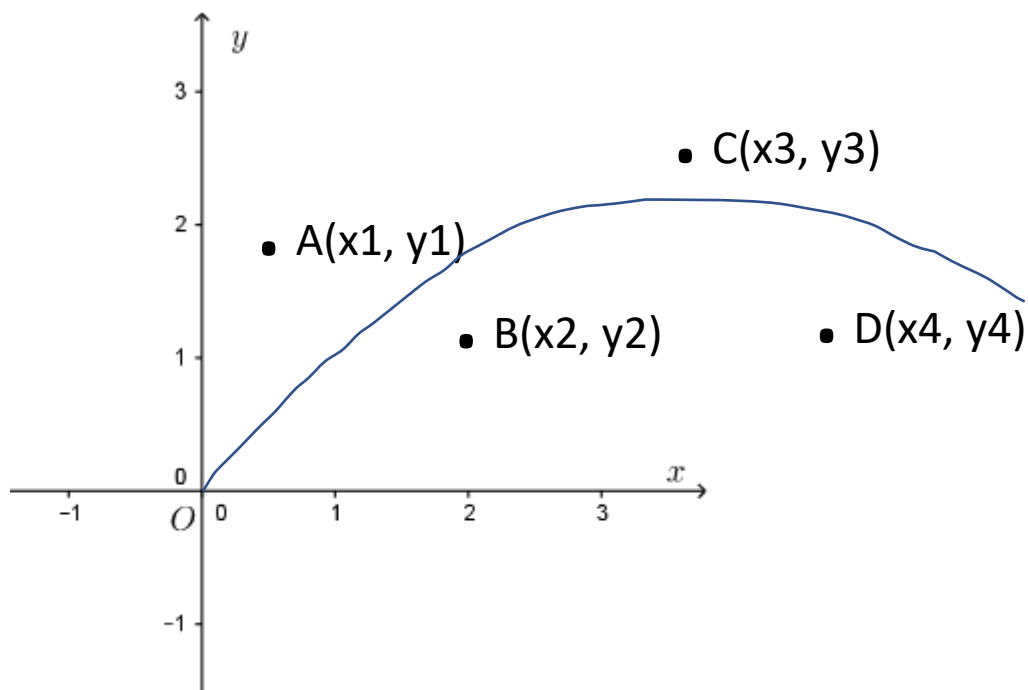
$$Y = ax^2 + bx + c$$

$$y_1 = ax_1^2 + bx_1 + c$$

$$y_2 = ax_2^2 + bx_2 + c$$

$$y_3 = ax_3^2 + bx_3 + c$$

$$y_4 = ax_4^2 + bx_4 + c$$

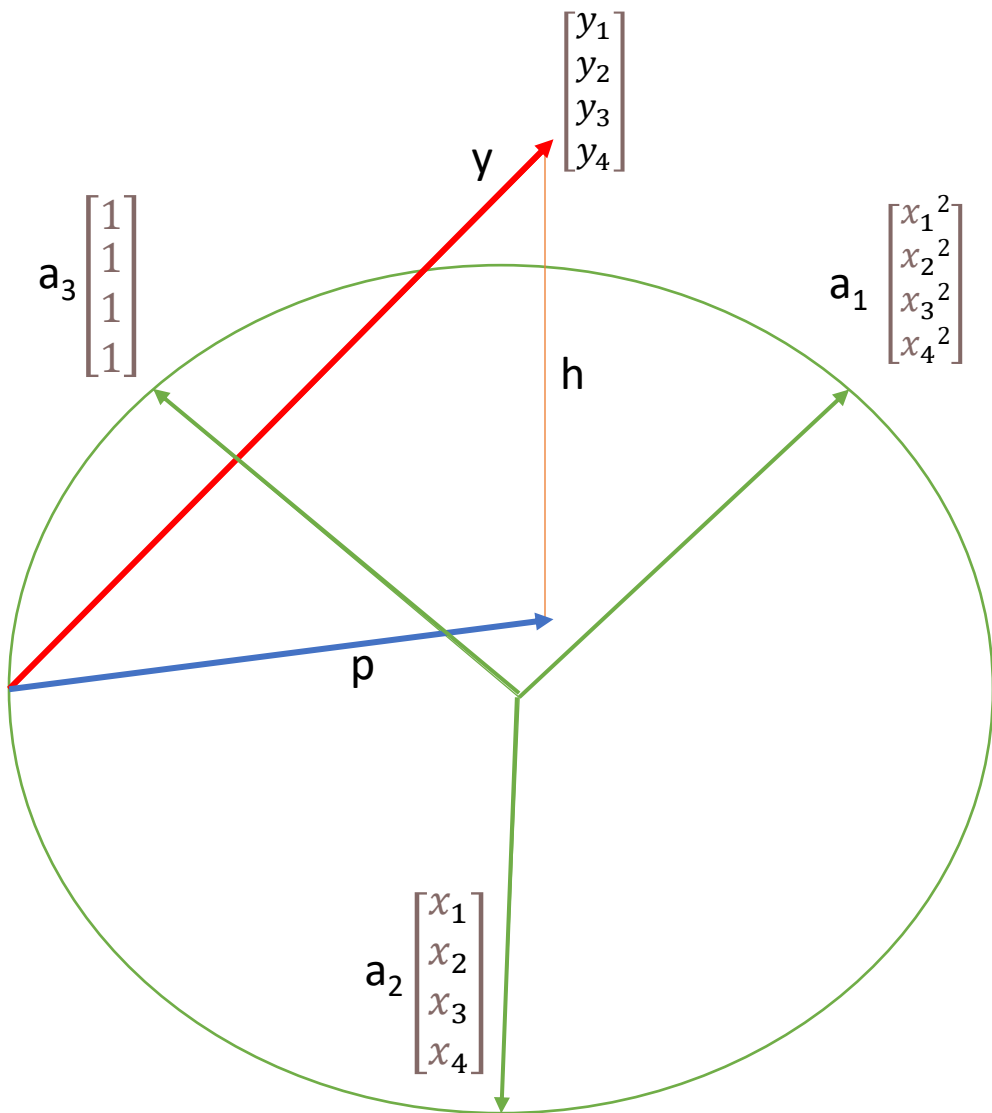


$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = a \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ x_4^2 \end{bmatrix} + b \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + c \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Không gian 3 chiều

Không gian 4 chiều

Từ kiến thức phần trước vậy suy ra bài toán này là tìm hình chiếu của vector lên một không gian 3 chiều



$$A = [a_1 \ a_2 \ a_3] = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \\ x_4^2 & x_4 & 1 \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} \quad x = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Công thức cuối cùng: $x = (A^T A)^{-1} A^T y$

Với không gian nhiều chiều hơn cũng tương tự như vậy...

Vấn đề:

- *Khi mà bài toán trở nên nhiều chiều thì rất tốn tài nguyên của máy tính, vì khi tính mũ -1 phải dùng Gram–Schmidt (máy tính không đủ mạnh)*

Cách giải quyết:

- *Sử dụng Gradient Descent để điều chỉnh (random) để tìm được phương sai Min của Linear regression*

Kiến thức về ma trận trước khi code

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$$

Mỗi cột của ma trận là một vector

Knowledge:

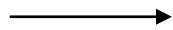
- Transpose (A^T): Lật ma trận lại (hàng \rightarrow cột, cột \rightarrow hàng)
- Shape: chiều (kích thước) của ma trận $A(2 \times 3)$
- Cộng 2 ma trận: Thực chất là cộng các vector lại với nhau $\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} + \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} = \begin{bmatrix} 2 & 6 & 10 \\ 4 & 8 & 12 \end{bmatrix}$
- Inverse (A^{-1})
- Det
- Eigen vector
- $Ax = b$
- Linear combination (cộng 2 vector nhiều chiều (có chứa biến)) là bài toán tìm x, y (x, y là độ dẫn của các vector sao cho khi nó cộng vào nhau nó tạo được một vector khác) và nó là bài toán to của bài linear regression, trong khi linear regression tìm vector gần nhất, thì bài toán này giải được khi vector nằm trên mặt phẳng do vẽ phải (bài toán minh họa) tạo ra
- Projection: hình chiếu

Nhân 2 ma trận

Ta chỉ nhân được 2 ma trận khi số cột của ma trận này bằng số hàng của ma trận kia

$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \end{bmatrix} 1 + \begin{bmatrix} 3 \\ 4 \end{bmatrix} 1 + \begin{bmatrix} 5 \\ 6 \end{bmatrix} 2 = \begin{bmatrix} 14 \\ 24 \end{bmatrix}$$

Biểu diễn trong python

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$


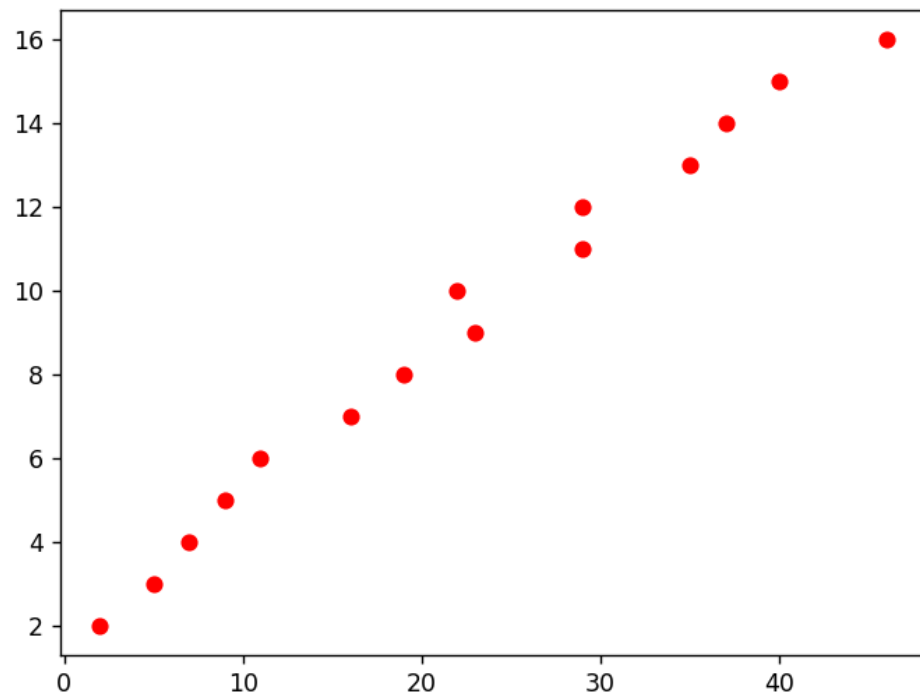
List A = [1, 2, 3]

List A = [[1],
[2],
[3]]

List A = [[1], [2], [3]]

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

A = [[1, 4],
[2, 5],
[3, 6]]



```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

#random data
A = [2, 5, 7, 9, 11, 16, 19, 23, 22, 29, 29, 35, 37, 40, 46]
b = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

#Visualize data
plt.plot(A, b, 'ro')
plt.show()
```

Code deploy dữ liệu


```

import numpy as np
import matplotlib
import matplotlib.pyplot as plt

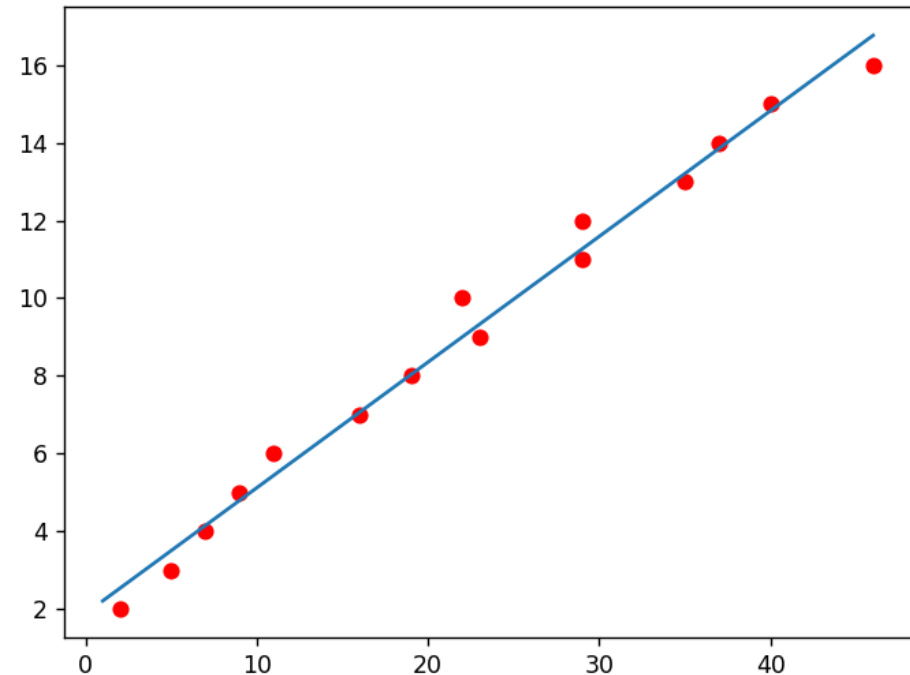
#random data
A = [2, 5, 7, 9, 11, 16, 19, 23, 22, 29, 29, 35, 37, 40, 46]
b = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

# x = (A^T A)^-1 A^T y
A = np.array([A]).T # Doi thanh hang doc
b = np.array([b]).T
plt.plot(A, b, 'ro')
ones = np.ones((A.shape[0], A.shape[1]), dtype = np.int8) # Tao vector
A = np.concatenate((A, ones), axis = 1) # Gop vector them vao ben phai
x = np.linalg.inv(A.transpose().dot(A)).dot(A.transpose()).dot(b)
print(x)

#Du doan
x_test = 12
y_test = x_test*x[0][0] + x[1][0]
print("Ket qua: ", y_test)

#Visualize data
x0 = np.array([[1, 46]]).T
y0 = x0*x[0][0] + x[1][0]
plt.plot(x0, y0)
plt.show()

```



```

[[0.32361847]
 [1.88039364]]
Predict for 12 là: 5.7638152914458765

```

Code thuần toán học

```

import numpy as np
import matplotlib
import matplotlib.pyplot as plt

from sklearn import linear_model
A = [2, 5, 7, 9, 11, 16, 19, 23, 22, 29, 29, 35, 37, 40, 46]
b = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]
A = np.array([A]).T # Đoi thành hàng dọc
b = np.array([b]).T

LR = linear_model.LinearRegression()

#Train
LR.fit(A, b)
print(LR.coef_) # a
print(LR.intercept_) # b (độ dốc)

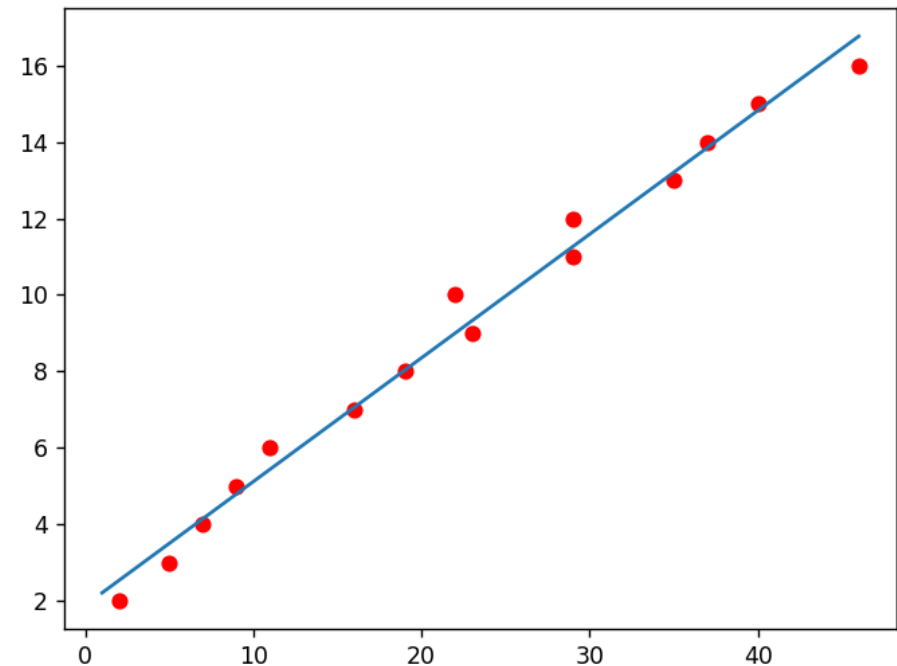
plt.plot(A, b, 'ro')

#Predict
x_test = 12
y_test = x_test*LR.coef_ + LR.intercept_
print(f"Predict for {x_test} là: {y_test}")

#Estimate (chuẩn đoán 1 đồng dữ liệu)
x0 = np.array([[1, 46]]).T
y0 = x0*LR.coef_ + LR.intercept_

plt.plot(x0, y0)
plt.show()

```



Code bằng thư viện

```

import numpy as np
import matplotlib
import matplotlib.pyplot as plt

# random data
b = [2,5,7,9,11,16,19,23,22,29,29,35,37,40,46,42,39,31,30,28,20,15,10,6]
A = [2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25]

#  $x = (A^T A)^{-1} A^T y$ 
A = np.array([A]).T # Doi thanh hang doc
b = np.array([b]).T
plt.plot(A, b, 'ro')

x_square = np.array([A[:, 0]**2]).T
A = np.concatenate((x_square, A), axis = 1) # Gop vector them vao ben phai

ones = np.ones((A.shape[0], 1), dtype = np.int8) # Tao vector 1
A = np.concatenate((A, ones), axis = 1) # Gop vector them vao ben phai

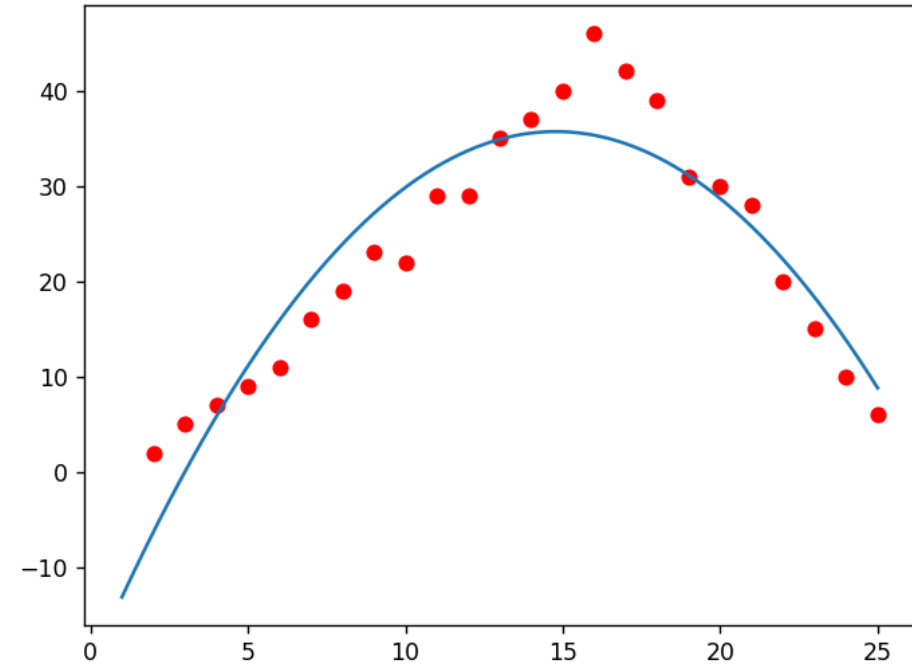
x = np.linalg.inv(A.transpose().dot(A)).dot(A.transpose()).dot(b)
print(x)

# a, b, c
#  $y = ax^2 + bx + c$ 

#Du doan
x_test = 12
y_test = x_test*x_test*x[0][0] + x_test*x[1][0] + x[2][0]
print(f"Predict for {x_test} là: {y_test}")

#Visualize data
x0 = np.linspace(1, 25, 10000)
print(x0)
y0 = x0*x0*x[0][0] + x0*x[1][0] + x[2][0]
plt.plot(x0, y0)
plt.show()

```



Nonlinear Regression code thuần toán học

Sinh viên

- Feature 1 (chăm chỉ)
- Feature 2 (thông minh)
- Feature 3 (ý chí)
- Feature 4 (...)
- ...
- Feature n (Điểm toán)



Dự đoán 1 trong số N
feature của sinh viên

Quay lại với bài này, thì để giải quyết chúng ta chỉ cần giải phương trình sau tương tự như bài trước:

$$Y = ax_1 + bx_2 + cx_3 + \dots + d$$

Đưa vào Sklearn

Ví dụ giải bài toán “dự đoán điểm toán của một sinh viên dựa vào sự thông minh và chăm chỉ”

$$z = ax + by + c$$

x: sự thông minh

y: sự chăm chỉ

z: Điểm toán

$$X = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$A = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \\ x_4 & y_4 & 1 \end{bmatrix}$$

$$z_1 = ax_1 + by_1 + c$$

$$z_2 = ax_2 + by_2 + c$$

$$z_3 = ax_3 + by_3 + c$$

$$z_4 = ax_4 + by_4 + c$$