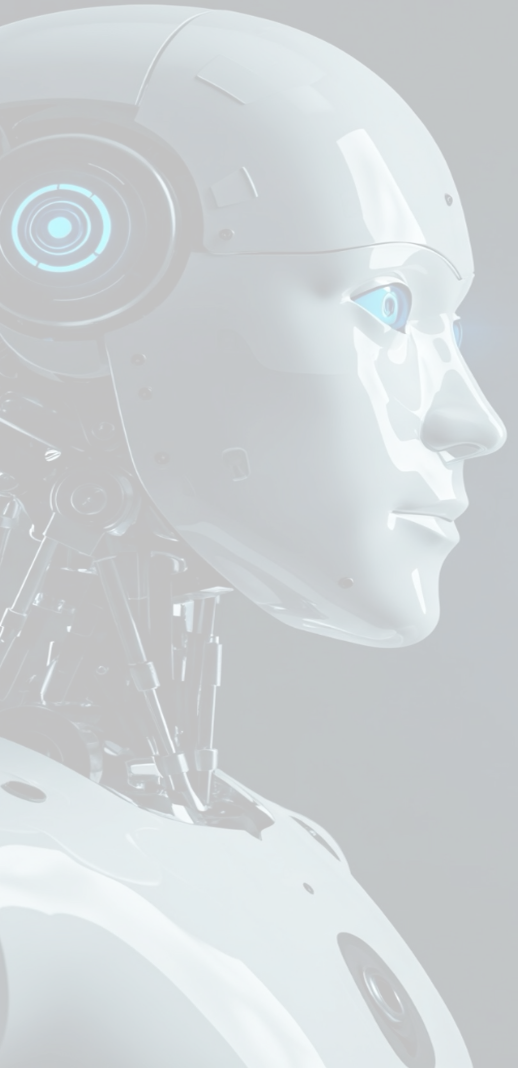


TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI TP.HCM
VIỆN CÔNG NGHỆ THÔNG TIN VÀ ĐIỆN, ĐIỆN TỬ

PHƯƠNG PHÁP TOÁN CHO MÁY HỌC

CHƯƠNG 2: KHÔNG GIAN VECTOR VÀ ÁNH XẠ TUYẾN TÍNH

TS. Trần Thế Vinh

KHÔNG GIAN VECTOR VÀ ÁNH XẠ TUYẾN TÍNH

Không gian véc tơ giúp biểu diễn dữ liệu trong không gian nhiều chiều, trong khi ánh xạ tuyến tính hỗ trợ việc biến đổi dữ liệu qua các phép toán ma trận.

Ứng dụng:

- **Chuẩn và khoảng cách** (L1, L2, L∞) giúp đo lường sự tương đồng giữa các dữ liệu, đặc biệt quan trọng trong bài toán phân loại và hồi quy.
- **Ma trận đặc biệt và chéo hóa** giúp đơn giản hóa bài toán tối ưu hóa và tăng tốc độ tính toán trong Machine Learning.

KHÔNG GIAN VECTOR

1. Các khái niệm về không gian vector:

Định nghĩa:

Không gian vector (không gian tuyến tính) là một tập hợp X với các đối tượng vector trên một trường số thực hoặc phức cùng với 2 phép toán:

- Cộng vector: $\forall x, y \in X\ thì\!: x + y \in X$.
- Nhân vô hướng: $\forall x \in X,$ với mọi số $\lambda \in \mathbb{R}(hoặc\ \mathbb{C}), thì\!:\ \lambda . x \in X$

Sao cho thoả mãn các tính chất sau:

- ✓ Tính kết hợp: $\forall x, y, z \in X\ thì\!: (x + y) + z = x + (y + z)$
- ✓ Tính giao hoán: $\forall x, y \in X\ thì\!: x + y \in X\ và\!: y + x = x + y$
- ✓ Tồn tại phần tử trung lập: có phần tử vector $\mathcal{O}$ sao cho: $x + \mathcal{O} = x, \forall x \in X$
- ✓ Tồn tại phần tử đối: $\forall x \in X, \exists$ phần tử $\in X, kí\ hiệu\!: -x$ sao cho $x + (-x\ ) = \mathcal{O}$
- ✓ Phân phối với phép cộng vector: $\alpha.(x + y) = \alpha.x + \alpha.y, \forall x, y \in X, \forall số\ \alpha$
- ✓ Phân phối với phép cộng vô hướng: $(\alpha + \beta).x = \alpha.x + \beta.x, \forall x \in X, \forall số\ \alpha, \beta.$
- ✓ Tính kết hợp của nhân vô hướng: $\alpha.(\beta.x) = (\alpha\beta).x, \forall x \in X, \forall số\ \alpha, \beta.$
- ✓ Phần tử đơn vị của hô hướng: $1.x = x, \forall x \in X$

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

Ví dụ:

1. Tập: $\mathbb{R}^n = \{x = (x_1, x_2, \dots, x_n): x_1, x_2, \dots, x_n \in \mathbb{R}\}$, với phép cộng (+) và phép nhân với số thực:

   $$\forall x, y \in \mathbb{R}^n: x = (x_1, x_2, \dots, x_n), y = (y_1, y_2, \dots, y_n); \alpha \in \mathbb{R}:$$

   $$x + y = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n);\ \alpha . x = (\alpha x_1, \alpha x_2, \dots, \alpha x_n)$$

   là một không gian vector (vector **0** =(0, 0,..., 0))

2. Với phép cộng 2 ma trận và phép nhân ma trận với số thực (phức), $\mathrm{R}^{\mathrm{m}\times\mathrm{n}}$ (hoặc $\mathbb{C}^{\mathrm{m}\times\mathrm{n}}$) trở thành không gian vector (vector **0** = $\mathcal{O}_{m\times n}$) .

3. $C_{[a,\, b]}$ là tập tất cả các hàm số liên tục trên [a, b], với phép cộng hai hàm số, phép nhân hàm số với số thực, trở thành một k.gian véc tơ (vector **0** là hàm đồng nhất = 0 trên [a, b].

4. $P_n[x]$ là tập tất cả các đa thức của biến x, có bậc không quá n, với phép cộng các hàm số, phép nhân h.số với số thực, là không gian vector.

   Trong chương này , chúng ta đặc biệt khảo sát một số nội dung quan trọng liên quan đến các không gian véc tơ: $\mathbb{R}^n$, $\mathrm{R}^{\mathrm{m}\times\mathrm{n}}$ (hoặc $\mathbb{C}^{\mathrm{m}\times\mathrm{n}}$).

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

2. Bao tuyến tính (Tập hợp sinh) của một hệ vector:

Cho một tập hợp vector X:

$$X = \{x_1, x_2, \dots, x_n\}$$

Bao tuyến tính của X (ký hiệu span(X), là tập hợp các vector có thể biểu diễn dưới dạng tổ hợp tuyến tính:

$$span(X) = \{\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n | \alpha_1, \alpha_2, \dots, \alpha_n \in \mathbb{R}\}$$

Ví dụ:

Xét hai vector trong không gian 2 chiều:

$$x_1 = (1, 0); x_2 = (0, 1);$$

Bao tuyến tính của nó là:

$$span(\{x_1, x_2\}) = \{\alpha_1.1 + \alpha_2.0, \alpha_1.0 + \alpha_2.1 | \alpha_1, \alpha_2 \in \mathbb{R}\} = \{\alpha_1, \alpha_2 | \alpha_1, \alpha_2 \in \mathbb{R}\}$$

Nhận xét:

- *Mọi vector trong không gian 2 chiều đều có thể được biểu diễn bằng 2 vector này.*
- *Vậy bao tuyến tính của 2 vector này chính là toàn bộ không gian 2 chiều.*

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

2. Hệ vector độc lập tuyến tính:

Một tập hợp vector $\{x_1, x_2, \dots, x_n\}$ trong không gian vector X được gọi là độc lập tuyến tính nếu phương trình sau chỉ có một nghiệm duy nhất là tất cả các hệ số đều bằng 0(hay tổ hợp tuyến tính tầm thường):

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = \mathcal{O} \Leftrightarrow \alpha_1 = \alpha_2 = \cdots = \alpha_n = 0$$

Nói cách dễ hiểu hơn:
Mỗi vector trong không gian vector X đóng góp một hướng mới, độc lập với các hướng của các vector còn lại.

Ví dụ:

$$x_1 = (1, 0); x_2 = (0, 1)$$

Thay vào phương trình ta có:

$$\alpha_1(1,0) + \alpha_2(0,1) = (0,0)$$

Tách từng toạ độ:

$$(\alpha_1.1 + \alpha_2.0, \alpha_1.0 + \alpha_2.1) = (0,0)$$

Ta có hệ phương trình:

$$\begin{cases} \alpha_1.1 + \alpha_2.0 = 0 \\ \alpha_1.0 + \alpha_2.1 = 0 \end{cases} \Leftrightarrow \begin{cases} \alpha_1 = 0 \\ \alpha_2 = 0 \end{cases}$$

Vậy 2 vector x_1, x_2 là độc lập tuyến tính.

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

2. Hệ vector phụ thuộc tuyến tính:

Một tập hợp vector $\{x_1, x_2, \dots, x_n\}$ trong không gian vector X được gọi là phụ thuộc tuyến tính nếu tồn tại một tổ hợp tuyến tính không tầm thường(tức là có ít nhất một hệ số khác 0) sao cho:

$$\exists \alpha_i \neq 0, \qquad \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = 0$$

Nói cách dễ hiểu hơn:
Có ít nhất một vector trong hệ có thể được tạo ra từ các vector còn lại.

Ví dụ:

$$x_1 = (1,0); x_2 = (0,1); x_3 = (2,2)$$

Thay vào phương trình ta có:

$$\alpha_1(1,0) + \alpha_2(0,1) + \alpha_3(2,2) = (0,0)$$

Tách từng toạ độ:

$$(\alpha_1.1 + \alpha_2.0 + \alpha_3.2, \alpha_1.0 + \alpha_2.1 + \alpha_3.2) = (0,0)$$

Ta có hệ phương trình:

$$\begin{cases} \alpha_1.1 + \alpha_3.2 = 0 \\ \alpha_2.1 + \alpha_3.2 = 0 \end{cases}$$

Đặt $\alpha_3 = -1$:

$$\begin{cases} \alpha_1 = 2 \\ \alpha_2 = 2 \end{cases}$$

Vậy ta có một tổ hợp tuyến tính không tầm thường là: $x_3 = 2x_1 + 2x_2$
Kết luận: x_3 có thể được tạo từ x_1 và x_2, do đó tập hợp $\{x_1, x_2, x_3\}$ phụ thuộc tuyến tính

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

Nhận xét	Giải thích	Ví dụ:
Nếu X chứa vector 0, thì X phụ thuộc tuyến tính	Vì vector 0 luôn có thể được biểu diễn bằng tổ hợp tuyến tính không tầm thường	$X = \{(1,0), (0,1), (0,0)\}$ phụ thuộc tuyến tính
Nếu X độc lập tuyến tính, thì $B \subset X$ cũng độc lập tuyến tính. Nếu B là phụ thuộc tuyến tính thì X cũng phụ thuộc tuyến tính	Vì nếu một tập con có sự phụ thuộc tuyến tính thì tập chứa nó cũng sẽ phụ thuộc.	$X = \{(1,0), (0,1)\}$ độc lập thì mọi tập con (1,0); (0,1) cũng độc lập
Mọi hệ vector độc lập tuyến tính cực đại trong cùng một không gian có cùng số vector	Vì số vector độc lập tối đa chính là số chiều của không gian đó	Trong $\mathbb{R}^2$, mọi hệ độc lập tuyến tính cực đại điều có 2 vector.

KHÔNG GIAN VECTOR

Ứng dụng:

Bao tuyến tính rất quan trọng trong giảm số chiều và biểu diễn dữ liệu.

Giảm số chiều dữ liệu:

- Trong PCA(Principal Component Analysis), tìm một tập hợp vector cơ sở có thể biểu diễn dữ liệu một cách tốt nhất.
- Các thành phần chính là một tập hợp các vector trực giao

Ví dụ:

- Một tập dữ liệu ảnh có 1000px (1000 chiều) , nhưng phần lớn thông tin nằm trên một mặt phẳng 10 chiều trong không gian 1000 chiều
- PCA tìm ra mặt phẳng 10 chiều này (bao tuyến tính của 10 vector chính), giúp giảm số chiều xuống còn 10 mà vẫn giữ phần lớn thông tin.

Biểu diễn dữ liệu:

- Trong hồi quy tuyến tính (Linear Regression), ta có tập các đặt trưng $\{X_1, X_2, \dots, X_n\}$ và muốn biểu diễn đầu ra

$$Y = \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_n X_n + \epsilon$$

- Nếu các đặt trưng X_i không tạo thành một không gian đầy đủ (span) của dữ liệu thì mô hình sẽ không dự đoán chính xác.
- Nếu một số đặc trưng có thể biểu diễn bằng các đặc trưng khác(phụ thuộc tuyến tính) thì có đa cộng tuyến (multicollinearity), làm cho mô hình bị kém hiệu quả.

Ví dụ:

- Nếu ta muốn dự đoán giá nhà dựa vào diện tích và số phòng, nhưng bỏ qua vị trí, có thể mô hình không đủ thông tin để dự đoán chính xác.
- Khi đó, tập các đặc trưng không tạo thành không gian đầy đủ, dẫn đến mô hình kém chính xác.

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

Ứng dụng:

Độc lập tuyến tính giúp chọn đặc trưng và xây dựng mạng Neural Networks.

Chọn lọc đặc trưng:

- Nếu một tập đặc trưng (features) có quá nhiều chiều và một số chiều là dư thừa, ta cần chọn một tập con độc lập tuyến tính để huấn luyện mô hình.
- Điều này giúp giảm overfitting, tăng tốc độ tính toán, và làm mô hình dễ hiểu hơn.

Ví dụ:

Nếu có 2 đặc trưng X_1= chiều dài (cm) và X_2= chiều dài (inch), chúng phụ thuộc tuyến tính ($X_2 = 2.54X_1$), nên có thể loại bớt một trong hai.

Neural Networks:

- Khi xây dựng các lớp ẩn trong mạng neural, ta muốn các hàm kích hoạt (activation functions) tạo ra các biểu diễn độc lập tuyến tính của dữ liệu.
- Nếu các giá trị đầu ra của một lớp bị phụ thuộc tuyến tính, mạng sẽ không học được thông tin mới.
- Mô hình sẽ bị nghẽn thông tin và không thể biểu diễn được các mối quan hệ phức tạp trong dữ liệu

```python
import numpy as np

def input_vector(dim, index):
    """Nhập một vector có độ dài dim"""
    print(f"Nhập vector {index} có {dim} phần tử:")
    return np.array([float(input(f"Phần tử {i + 1}: ")) for i in range(dim)])

def is_linearly_independent(vectors):
    """Kiểm tra tập vector có độc lập tuyến tính không"""
    matrix = np.column_stack(vectors)
    rank = np.linalg.matrix_rank(matrix)
    return rank == len(vectors)

# 1. Nhập số chiều của không gian vector
dim = int(input("Nhập số chiều của không gian vector (n): "))

# 2. Nhập số lượng vector cần kiểm tra
num_vectors = int(input("Nhập số lượng vector: "))

# 3. Nhập danh sách vector
vectors = [input_vector(dim, i+1) for i in range(num_vectors)]

# Hiển thị các vector đã nhập
print("\nCác vector đã nhập:")
for i, v in enumerate(vectors, 1):
    print(f"v{i} = {v}")

# 4. Thực hiện phép cộng vector (nếu có ít nhất 2 vector)
if num_vectors > 1:
    sum_vector = sum(vectors)
    print("\nTổng của các vector:", sum_vector)

# 5. Thực hiện phép nhân vô hướng
alpha = float(input("\nNhập một số vô hướng để nhân với vector đầu tiên: "))
scaled_vector = alpha * vectors[0]
print(f"Nhân vector v1 với {alpha}:", scaled_vector)

# 6. Kiểm tra phần tử trung lập (vector 0)
zero_vector = np.zeros(dim)
print("\nVector không (trung lập phép cộng):", zero_vector)

# 7. Tìm phần tử đối của từng vector
opposite_vectors = [-v for v in vectors]
print("\nPhần tử đối của từng vector:")
for i, v in enumerate(opposite_vectors, 1):
    print(f"Phần tử đối của v{i}: {v}")

# 8. Kiểm tra độc lập tuyến tính
if is_linearly_independent(vectors):
    print("\nCác vector là độc lập tuyến tính.")
else:
    print("\nCác vector phụ thuộc tuyến tính.")

# 9. Xác định số chiều của không gian vector sinh ra
basis = np.column_stack(vectors)
dimension = np.linalg.matrix_rank(basis)
print("Số chiều của không gian vector là:", dimension)
```

KHÔNG GIAN VECTOR

Cơ sở của không gian vector:

Hệ vector S được gọi là một cơ sở của không gian vector X khi và chỉ khi thoả mãn một trong các điều kiện sau:

- S là một tập hợp sinh và độc lập tuyến tính trong X
- S là một tập hợp sinh cực tiểu của X (tức là nếu bỏ đi bất kỳ vector nào trong S, tập hợp sẽ không còn sinh được X)
- S là một tập hợp độc lập tuyến tính cực đại trong X (tức là nếu thêm bất kỳ vector nào vào S, tập hợp sẽ trở thành phụ thuộc tuyến tính)
- Mọi vector trong X đều có thể biểu diễn duy nhất theo tổ hợp tuyến tính của các vector trong S.
- Số phần tử của S bằng số chiều của không gian vector X(dim(X)).

Ví dụ:
Xét không gian $\mathbb{R}^3$ và tập hợp 3 vector sau:

$$S = \{(1,0,0), (0,1,0), (0,0,1)\}$$

Mọi vector trong $\mathbb{R}^3$, chẳng hạn (a,b,c), đều có thể viết dưới dạng tổ hợp tuyến tính:

$$(a, b, c) = a(1,0,0) + b(0,1,0) + c(0,0,1)$$

do đó S sinh ra toàn bộ không gian $\mathbb{R}^3$

- Không có vector nào trong tập có thể biểu diễn bằng các vector còn lại, do đó S độc lập tuyến tính
- Nếu bỏ đi bất kỳ vector nào, ta không thể sinh ra toàn bộ $\mathbb{R}^3$ nữa.
- Nếu thêm một vector mới, tập sẽ trở thành phụ thuộc tuyến tính.
- Số vector trong S đúng bằng số chiều của $\mathbb{R}^3$ (3 vector)

```python
import numpy as np

def is_basis(vectors):
    """
    Kiểm tra xem tập vectors có phải là một cơ sở của không gian hay không.
    """
    # Chuyển danh sách vector thành ma trận
    matrix = np.array(vectors).T  # Chuyển vector thành cột

    # Lấy số vector và số chiều của không gian
    num_vectors, dimension = matrix.shape[1], matrix.shape[0]

    # Điều kiện: Số vector phải bằng số chiều của không gian
    if num_vectors != dimension:
        return False, "Số vector không bằng số chiều của không gian => Không phải cơ sở."

    # Kiểm tra độc lập tuyến tính: Hạng của ma trận phải bằng số chiều không gian
    rank = np.linalg.matrix_rank(matrix)
    if rank < dimension:
        return False, "Tập vector phụ thuộc tuyến tính => Không phải cơ sở."

    return True, "Tập vector là một cơ sở của không gian."

# Nhập số chiều của không gian
try:
    n = int(input("Nhập số chiều của không gian vector (n): "))
    if n <= 0:
        raise ValueError
except ValueError:
    print("Lỗi: n phải là số nguyên dương!")
    exit()

# Nhập các vector từ bàn phím
vectors = []
print(f"Nhập {n} vector, mỗi vector gồm {n} phần tử (cách nhau bởi dấu cách):")
for i in range(n):
    try:
        vector = list(map(float, input(f"Vector {i+1}: ").split()))
        if len(vector) != n:
            raise ValueError
        vectors.append(vector)
    except ValueError:
        print("Lỗi: Mỗi vector phải có đúng n phần tử!")
        exit()
print(vectors)
# Kiểm tra xem tập vector có phải là cơ sở không
is_basis_result, message = is_basis(vectors)
print(message)
```

KHÔNG GIAN VECTOR

Số chiều của không gian vector:

Với S là một cơ sở của X, ta có:

$$\dim(X) = \begin{cases} n, \text{ nếu S gồm n vector} \\ +\infty, \text{ nếu S có vô số vector} \end{cases}$$

Ta gọi dim(X) là số chiều của không gian vector .

- Nếu $\dim(X) = n$, ta nói X là không gian hữu hạn chiều (n chiều).
- Nếu $\dim(X) = +\infty$ ta nói X là không gian vô hạn chiều.

Hạng của một hệ vector B là số vector độc lập tuyến tính cực đại trong B.

Ví dụ:

1. $\Phi = \{e_1, e_2, \dots, e_n\}, \left(e_j = \left(0, \dots, 0, \underbrace{1}_{j}, 0, \dots, 0\right), j = 1,2, \dots, n\right)$ là một cơ sở trong $\mathbb{R}^n$***, gọi là cơ sở chính tắc của*** $\mathbb{R}^n$**.** Vì thế $\mathbb{R}^n$ là không gian n chiều

2. $\Phi = \{\varphi_0, \varphi_1, \dots, \varphi_n\}$ $(\varphi_0(x) = 1, \varphi_1(x) = x, \dots, \varphi_n(x) = x^n, \forall x \in \mathbb{R})$ là một cơ sở ***(gọi là cơ sở chính tắc)*** trong $P_n[x]$**.** Vậy $P_n[x]$ là không gian n + 1 chiều.

3. $\wp[x]$ (tập tất cả các đa thức) và $C_{[a,\,b]}$ là các không gian vô hạn chiều.

4. Trong không gian vector $\mathrm{R}^{m\times n}$, hệ véc tơ

$$\Phi = \left\{\varphi_{ij} = (a_{rs})_{m\times n}: a_{rs} = \begin{cases} 1, \text{khi } r = i, s = j \\ 0, \text{khi } r \neq i, \text{hoặc } s \neq j \end{cases}, i = \overline{1, m}, j = \overline{1, n}\right\},$$

là một cơ sở, gọi là cơ sở chính tắc của $\mathrm{R}^{m\times n}$.

```python
import numpy as np

def find_vector_space_dimension(vectors):
    """
    Xác định số chiều của không gian vector dựa trên hạng của ma trận chứa các vector.
    """
    matrix = np.array(vectors)  # Chuyển danh sách vector thành ma trận
    rank = np.linalg.matrix_rank(matrix)  # Tính hạng của ma trận
    return rank

# Nhập số vector
try:
    n = int(input("Nhập số vector: "))
    vectors = []

    # Nhập từng vector
    for i in range(n):
        vec = list(map(float, input(f"Nhập vector {i+1} (các phần tử cách nhau bởi dấu cách): ").split()))
        vectors.append(vec)

    # Xác định số chiều của không gian vector
    dim = find_vector_space_dimension(vectors)
    print(f"Số chiều của không gian vector: {dim}")

except ValueError:
    print("Lỗi: Hãy nhập số nguyên hợp lệ và các phần tử số thực cách nhau bởi dấu cách!")
```

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Các khái niệm về không gian chuẩn:

Định nghĩa 1:

Không gian định chuẩn $(X, \|.\|)$ là một không gian vector có một chuẩn được xác định, tức là một ánh xạ $\|.\|: X \to [0, +\infty)$ thoả mãn ba tính chất:

- Tính dương xác định: $\|x\| = 0 \Leftrightarrow x = \mathcal{O}$ (vector không);
- Tính đồng nhất thuần: $\|\alpha . x\| = |\alpha| . d(x), \forall x \in X, \forall \alpha \in K$;
- Bất đẳng thức tam giác: $\|x + y\| \leq \|x\| + \|y\|, \forall x, y \in X$

lúc đó $\|x\|$ là chuẩn (hay độ dài) của vector x và $d(x, y) = \|x - y\|$ là khoảng cách giữa 2 phần tử x và y.

Vector x có độ dài = 1, gọi là vector đơn vị.

Với không gian định chuẩn $(X, \|.\|)$, $a \in X, r > 0$:

- $B(a, r) = \{u \in X: \|u - a\| < r\}$: Hình cầu mở tâm a, bán kính r
- $\overline{B}(a, r) = \{u \in X: \|u - a\| \leq r\}$: Hình cầu đóng tâm a, bán kính r
- $S(a, r) = \{u \in X: \|u - a\| = r\}$: Mặt cầu tâm a, bán kính r

```python
import numpy as np

class NormedSpace:
    def __init__(self, vector):
        self.vector = np.array(vector)

    def l1_norm(self):
        """Chuẩn L1 (Manhattan)"""
        return np.sum(np.abs(self.vector))

    def l2_norm(self):
        """Chuẩn L2 (Euclidean)"""
        return np.sqrt(np.sum(self.vector ** 2))

    def linf_norm(self):
        """Chuẩn vô hạn (L∞)"""
        return np.max(np.abs(self.vector))

    def check_triangle_inequality(self, other_vector):
        """Kiểm tra bất đẳng thức tam giác: ||x + y|| ≤ ||x|| + ||y||"""
        other_vector = np.array(other_vector)
        norm_x = self.l2_norm()
        norm_y = np.sqrt(np.sum(other_vector ** 2))
        norm_x_plus_y = np.sqrt(np.sum((self.vector + other_vector) ** 2))
        return norm_x_plus_y <= norm_x + norm_y

# Nhập vector từ bàn phím
def get_vector_from_input():
    vector_str = input("Nhập vector (các phần tử cách nhau bằng dấu cách): ")
    vector = list(map(float, vector_str.split()))
    return vector

# Nhập vector x và y từ bàn phím
print("Nhập vector x:")
x = NormedSpace(get_vector_from_input())

print("Nhập vector y:")
y = NormedSpace(get_vector_from_input())

# Tính các chuẩn
print(f"L1-norm của x: {x.l1_norm()}")
print(f"L2-norm của x: {x.l2_norm()}")
print(f"L∞-norm của x: {x.linf_norm()}")

# Kiểm tra bất đẳng thức tam giác
print(f"Bất đẳng thức tam giác có đúng không? {x.check_triangle_inequality(y.vector)}")
```

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Định nghĩa 2:

Mỗi ánh xạ: $\langle .,. \rangle : X \times X \to \mathbb{R}$ được gọi là một tích vô hướng trên X, nếu thỏa mãn các điều kiện:

- Tính đối xứng: $\langle x, y\rangle = \langle y, x\rangle, \forall x, y \in X$

(điều này có nghĩa là thứ tự của các vector trong phép tính vô hướng không ảnh hưởng đến kết quả)

- Tính xác định dương: $\langle x, x\rangle \geq 0, \forall x \in X,$ và $\langle x, x\rangle = 0 \Leftrightarrow x = \mathcal{O}$

(Tích vô hướng của một vector với chính nó luôn không âm và chỉ bằng 0 khi vector là vector $\mathcal{O}$

- Bất đẳng thức Cauchy-Schwarz: $|\langle x, y\rangle|^2 \leq \langle x, x\rangle + \langle y, y\rangle, \forall x, y \in X.$

(Đây là bất đẳng thức quan trọng, đảm bảo rằng góc giữa 2 vector có giá trị hợp lệ trong không gian số thực)

Ứng dụng trong không gian vector:

Giả sử X là không gian vector có tích vô hướng $\langle .,. \rangle$

Chuẩn cảm sinh: Tích vô hướng $\langle .,. \rangle$ cảm sinh một chuẩn (norm) $\|.\|$ trên X:

$$\|x\| = \sqrt{\langle x, x\rangle}$$

Chuẩn này cho biết độ dài của vector trong không gian vector.

Khoảng cách giữa 2 vector: Dựa vào chuẩn, ta có thể tính khoảng cách giữa 2 vector như sau:

$$d(x, y) = \|x - y\| = \sqrt{\langle x - y, x - y\rangle}$$

Hệ trực giao và hệ trực chuẩn:

Nếu $\langle x, y\rangle = 0$, ta nói x và y trực giao nhau, kí hiệu: $x \perp y$.

- Hệ vector S được gọi là ***hệ trực giao*** nếu: $\forall x, y \in X, x \neq y$ thì $x \perp y$.
- Một cơ sở S gồm các vector trực giao nhau gọi là một ***cơ sở trực giao***
- Hệ vector S được gọi là ***hệ trực chuẩn*** nếu S là hệ trực giao, đồng thời mọi vector trong hệ đều là vector đơn vị.
- Một cơ sở gồm các véc tơ trực chuẩn gọi là một ***cơ sở trực chuẩn***

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Một số chuẩn thường dùng các phông gian $\mathbb{R}^n$ và $\mathbb{R}^{m\times n}$:

Trên một không gian vector có thể đưa ra nhiều chuẩn khác nhau, tức là nhiều cá đo độ dài vector khác nhau.
Trên $\mathbb{R}^n$, xây dựng như sau: $\forall x, y \in \mathbb{R}^n: x = (x_1, x_2, \dots, x_n), y = (y_1, y_2, \dots, y_n)$

Chuẩn Euclide trong $\mathbb{R}^n$:
Tích vô hướng: $\langle x, y\rangle = \sum_{i=1}^{n} x_i . y_i$ cảm sinh chuẩn sau gọi là chuẩn Euclide***:***

$$\|x\| = \langle x, x\rangle^{1/2} = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}$$

và khoảng cách Euclide:

$$d(x, y) = \|x - y\| = \langle x - y, x - y\rangle^{1/2}$$
$$= \left(\sum_{i=1}^{n} (x_i - y_i)^2\right)^{1/2}$$

$\mathbb{R}^n$ với tích vô hướng nói trên được gọi là không gian Euclide n chiều

Chuẩn Manhattan $\mathbb{R}^n$: Đo khoảng cách giữa 2 điểm bằng tổng độ lệch tuyệt đối theo từng toạ độ:

$$d(x, y) = \sum |x_i - y_i|$$

Khoảng cách này còn gọi là khoảng cách taxi, vì nó đo đường đi theo lưới ô vuông (đi theo các trục toạ độ thay vì đường chéo)

Chuẩn Chebyshev $\mathbb{R}^n$: Đo khoảng cách giữa 2 điểm bằng giá trị tuyệt đối lớn nhất của các toạ độ:

$$d(x, y) = \max|x_i - y_i|$$

Chuẩn này hữu ích trong hệ thống mà chỉ cần một sự khác biệt lớn nhất là đủ để quyết định khoảng cách.

Ví dụ:
Giả sử $x = (3,4), y = (0,0)$, khi đó:
Chuẩn Euclide:

$$d(x, y) = \sqrt{(3-0)^2 + (4-0)^2} = \sqrt{9+16} = 5$$

Chuẩn Manhattan:

$$d(x, y) = |3-0| + |4-0| = 7$$

Chuẩn Chebyshev:

$$d(x, y) = \max(|3-0|, |4-0|) = 4$$

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Ứng dụng:

1. Chuẩn hoá dữ liệu (Normalization)
    - Trong xử lý dữ liệu, việc chuẩn hoá giúp các đặc trưng có cùng thang đo, giúp thuật toán hội tụ nhanh hơn
    - Các kỹ thuật như Min-Max Scaling, Z-score Normalization dựa trên chuẩn L_2(chuẩn Euclid) hoặc chuẩn L_1(chuẩn Manhattan).
    - Ví dụ: Chuẩn hoá một vector đặc trưng x theo chuẩn L_2(chuẩn Euclid):

    $$x' = \frac{x}{\|x\|_2}$$

    Giúp các giá trị nằm trong khoảng [-1, 1] hoặc [0,1], giúp tối ưu mô hình.
2. Định nghĩa khoảng cách trong ML:
    - Các thuật toán máy học thường dựa vào khoảng cách giữa các điểm dữ liệu để đưa ra quyết định
    - Các chuẩn phổ biến để đo khoảng cách:

    Chuẩn Euclid (L_2): $d(x, y) = \sqrt{\sum(x_i - y_i)^2}$

    Chuẩn Manhattan(L_1): $d(x, y) = \sum|x_i - y_i|$

    Chuẩn Chebyshev(L_∞): $d(x, y) = \max|x_i - y_i|$
    - Thuật toán KNN (K-Nearest Neighbors) khi tìm điểm gần nhất, dùng khoảng cách Euclid hoặc Manhattan để xác định hàng xóm gần nhất.

3. Giúp Gradient Descent và Điều chuẩn mô hình
    - Trong DL, Gradient Descent sử dụng chuẩn L_2 để đo độ dốc của hàm loss.
    - Regularization:
        - L1 Regularization (Lasso Regression): $\lambda \sum|x_i|$
        - L2 Regularization (Ridge Regression): $\lambda \sum x_i^2$
        - Dùng để giảm overfitting, tăng khả năng tổng quát của mô hình.
4. Mô hình không gian vector trong NLP:
    - Trong Xử lý ngôn ngữ tự nhiên (NLP), các từ được biểu diễn dưới dạng vector trong không gian định chuẩn.
    - Ví dụ: Google Translate, Chatbot,.. khoảng cách giữa các từ được tính bằng cosine similarity: $\cos(x, y) = \frac{x.y}{\|x\|_2\|y\|_2}$
5. Xử lý ảnh (Computer Vision):
    - Trong **nhận diện khuôn mặt**, vector đặc trưng khuôn mặt thường được so sánh bằng **khoảng cách Euclid**. Nếu vector đặc trưng khuôn mặt của 2 người là x và y, ta so sánh: $d(x, y) = \sqrt{\sum(x_i - y_i)^2}$ khoảng cách càng nhỏ thì 2 khuôn mặt càng giống nhau, ngược lại càng lớn thì càng khác nhau.

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Một số bài toán cơ bản trong $\mathbb{R}^n$:

Tọa độ vector theo cơ sở: Ta đã biết

$$\Phi = \{e_1, e_2, \dots, e_n\}, (e_j = \left(0, \dots, 0, \underbrace{1}_{j}, 0, \dots, 0\right), j = 1,2, \dots, n)$$

là một cơ sở của $\mathbb{R}^{\mathrm{n}}$ và vì vậy $\mathbb{R}^n$ là không gian vector n chiều và mọi cơ sở của $\mathbb{R}^n$ đều gồm n vector.

Giả sử $\mathrm{S} = \{\mathrm{u}_1, \mathrm{u}_2, \dots, \mathrm{u}_\mathrm{n}\}$ là một cơ sở có sắp thứ tự của $\mathbb{R}^n$, khi đó mỗi $\mathrm{x} \in \mathbb{R}^n$
có biểu diễn duy nhất: $\mathrm{x} = \mathrm{t}_1.\mathrm{u}_1 + \mathrm{t}_2.\mathrm{u}_2, \dots + \mathrm{t}_\mathrm{n}.\mathrm{u}_\mathrm{n}$
và $\mathrm{t}_1, \mathrm{t}_2, \dots, \mathrm{t}_\mathrm{n}$ là ***các tọa độ của vector x trong cơ sở sắp thứ tự S***. Kí hiệu:
$(\mathrm{x})_\mathrm{S} = (\mathrm{t}_1, \mathrm{t}_2, \dots, \mathrm{t}_\mathrm{n})$ (gọi là ***dòng hay hàng tọa độ của x trong S***)
$[\mathrm{x}]_\mathrm{S} = {(\mathrm{x})_\mathrm{S}}^\mathrm{T}$ (chuyển vị của $(\mathrm{x})_\mathrm{S}$) (gọi là ***cột tọa độ của x trong S***)

Trên $\mathbb{R}^n$ với tích vô hướng Euclide, cơ sở Φ là một cơ sở trực chuẩn, còn gọi là cơ sở chính tắc của $\mathbb{R}^n$.
$\forall x \in \mathbb{R}^n: x = (x_1, x_2, \dots, x_n) \Leftrightarrow \mathrm{x} = x_1.\mathrm{e}_1 + x_2.\mathrm{e}_2 + \cdots + x_n.\mathrm{e}_\mathrm{n}$
Như vậy $(x_1, x_2, \dots, x_n)$ chính là dòng tọa độ của x trong cơ sở chính tắc Φ

Ví dụ:
Giả sử ta có một hệ cơ sở trong $\mathbb{R}^2$:

$$\mathrm{S} = \{\varphi_1 = (1,1), \varphi_2 = (1,-1)\}$$

Và ta có vector φ =(3,1), ta cần tìm các toạ độ x_1, x_2 sao cho:

$$\varphi = x_1\varphi_1 + x_2\varphi_2$$

Tức là giải hệ phương trình: $\begin{cases} x_1.1 + \mathrm{x}_2.1 = 3 \\ \mathrm{x}_1.1 + \mathrm{x}_2.(-1) = 1 \end{cases} \Leftrightarrow \begin{cases} x_1 = 2 \\ x_2 = 1 \end{cases}$

Vậy toạ độ của φ theo cơ sở S là (2,1) $\varphi = 2\varphi_1 + \varphi_2$

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Tìm hạng của một hệ vector trong $\mathbb{R}^n$

$\forall u \in \mathbb{R}^n$, để đơn giản ta kí hiệu dòng và cột tọa độ trong cơ sở chính tắc của u là: (u) và [u] tương ứng.

Cho hệ vector $B = \{u_1, u_2, \dots, u_m\} \subset \mathbb{R}^n$, hạng của hệ vector B là số kí hiệu: rank(B) = số vector độc lập tuyến tính cực đại trong B. Dễ thấy nếu m > n thì B phụ thuộc tuyến tính, vì thế:
$0 \le \text{rank}(B) \le n\,;\ \text{rank}(B) = 0 \Leftrightarrow$ B chỉ chứa vector không .

Gọi: $([u_1]\ \ [u_2] \dots [u_m])$ là ma trận liên kết cột của hệ B, $\big((u_1)\ \ (u_2) \dots (u_m)\big)$ là ma trận liên kết hàng của hệ B. Một trong 2 ma trận này ta kí hiệu là: B^*.
Định lý: Hạng của một ma trận A bằng hạng của hệ vector cột (hoặc hàng) của A
Từ đó suy ra: rank (B) = rank (B^*), và như vậy bài toán tìm hạng của hệ vector B chuyển về bài toán tìm hạng của ma trận liên kết (hàng hoặc cột) của B.
VD1. Cho B = {u = (1,0,2,1), v = (2,2,1,2), w =(0,2,0,1), z = (-1,0,1,0)}, tìm rank(B)
Ta tìm sơ đồ biến đổi ma trận liên kết hàng của B:

$$B^* = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 2 & 2 & 1 & 2 \\ 0 & 2 & 0 & 1 \\ -1 & 0 & 1 & 0 \end{bmatrix} \xrightarrow[h4+h1\to h4]{h2-2h1\to h2} \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 2 & -3 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 3 & 1 \end{bmatrix} \xrightarrow[h4-(h3-h2)\to h4]{h3-h2\to h3} \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 2 & -3 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = B^{**}$$

(B^{**} là ma trận bậc thang)
Vậy rank(B) = rank(B^*) = rank (B^{**}) = 3
Chú ý:
(1)Hệ véc tơ B là một cơ sở trong $\mathbb{R}^n \Leftrightarrow$ B gồm đúng n vector độc lập tuyến tính.
(2) Trong các không gian $\mathbb{R}^n, M_{n\times m}(\mathbb{R}), P_n[x]$, khi xét ma trận liên kết của một hệ vector, người ta thường xét trong cơ sở chính tắc và khi nói tới ma trận liên kết của một hệ vector mà không giải thích gì thêm, thì hiểu là xét trong cơ sở chính tắc.

```python
import numpy as np

def rank_of_vectors(dim, num_vectors):
    """
    Tính hạng của một hệ vector trong ℝ^n
    :param dim: Chiều không gian của vector (số phần tử trong mỗi vector)
    :param num_vectors: Số lượng vector trong hệ
    :return: Hạng của hệ vector
    """
    vectors = []

    print(f"Nhập {num_vectors} vector trong không gian ℝ^{dim}:")
    for i in range(num_vectors):
        vector = list(map(float, input(f"Nhập vector {i+1} (cách nhau bằng dấu cách): ").split()))
        if len(vector) != dim:
            print(f"Lỗi: Vector phải có đúng {dim} phần tử. Hãy nhập lại!")
            return
        vectors.append(vector)

    # Chuyển danh sách vector thành ma trận và tính hạng
    matrix = np.array(vectors)
    rank = np.linalg.matrix_rank(matrix)

    print(f"\nHạng của hệ vector: {rank}")

# Nhập chiều không gian và số vector
dim = int(input("Nhập chiều không gian của vector (n): "))
num_vectors = int(input("Nhập số vector trong hệ: "))

# Gọi hàm tính hạng
rank_of_vectors(dim, num_vectors)
```

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Ý nghĩa:

Hạng của một hệ vector cho biết số chiều tối đa của không gian mà hệ vector đó có thể sinh ra. Điều này giúp ta hiểu được:

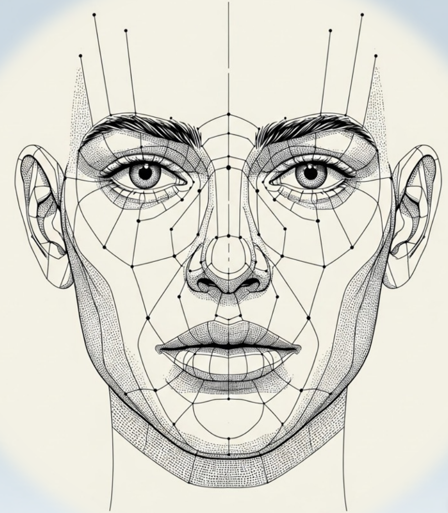
- ✓ Hệ vector có độc lập tuyến tính hay không
- ✓ Hệ vector có thể sinh ra toàn bộ không gian hay chỉ một phần của không gian đó
- ✓ Một vector trong hệ có thể biểu diễn dưới dạng tổ hợp tuyến tính của các vector khác hay không

Ví dụ:

- Giả sử có một tập hợp các camera quan sát trong một toà nhà. Nếu hạng của tập vector liên quan đến vị trí camera nhỏ hơn số camera, điều đó có nghĩa một số camera có thể bị lãng phí (chúng không mang thêm thông tin mới so với các camera khác).

Ứng dụng:

- Trong ML và AI, vector được dùng để biểu diễn dữ liệu, hình ảnh, giọng nói, giúp máy tính hiểu và xử lý thông tin từ thế giới thực.
- Hạng vector được sử dụng để giảm chiều dữ liệu, giúp mô hình học nhanh hơn và chính xác hơn.
- Ví dụ: Trong nhận diện khuôn mặt, mỗi khuôn mặt có thể được biểu diễn bằng một tập hợp vector đặc trưng. Nếu một số vector có thể được biểu diễn bởi các vector khác, thì ta có thể bỏ đi (giảm chiều dữ liệu) mà vẫn dữ được thông tin quan trọng.



KHÔNG GIAN VECTOR

Không gian định chuẩn:

Đổi cơ sở trong $\mathbb{R}^n$:
Trong không gian vector $\mathbb{R}^n$, một cơ sở là một tập hợp gồm n vector độc lập tuyến tính, dùng để biểu diễn mọi vector khác trong không gian đó.

Đổi cơ sở có nghĩa là chuyển toạ độ của một vector từ một cơ sở B sang một cơ sở khác B′. Điều này được thực hiện bằng cách tìm ma trận chuyển cơ sở, giúp biến đổi toạ độ của vector từ hệ này sang hệ khác.

Ma trận chuyển cơ sở:
Giả sử trong $\mathbb{R}^n$, ta có:

- Cơ sở cũ $B = \{u_1, u_2, \dots, u_n\}$
- Cơ sở mới $\mathrm{B}' = \{v_1, v_2, \dots, v_n\}$
- Một vector u có toạ độ trong cơ sở cũ là $[u]_B$
- Ta muốn tìm toạ độ trong cơ sở mới $[u]_{\mathrm{B}'}$

Bước 1:
Lập ma trận chuyển cơ sở P, trong đó các cột là toạ độ của B′ theo B:

$$v_1 = \mathrm{a}_{11}u_1 + \mathrm{a}_{12}u_2 + \cdots + \mathrm{a}_{1n}u_n$$
$$v_2 = \mathrm{a}_{21}u_1 + \mathrm{a}_{22}u_2 + \cdots + \mathrm{a}_{2n}u_n$$
$$\dots\dots$$
$$v_n = \mathrm{a}_{n1}u_1 + \mathrm{a}_{n2}u_2 + \cdots + \mathrm{a}_{nn}u_n$$

Ta có ma trận chuyển cơ sở P từ B′ sang B:

$$P = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

Tức là, mỗi vector trong cơ sở mới v_i được viết bằng tổ hợp tuyến tính của các vector trong cơ sở cũ
Bước 2: Toạ độ mới được tính bằng công thức:

$$[u]_{\mathrm{B}'} = P^{-1}\,[u]_B$$

Trong đó P^{-1} là ma trận nghịch đảo của P.

Ví dụ:
Giả sử trong $\mathbb{R}^2$, ta có:

- Cơ sở cũ $B = \{(1,1), (1,-1)\}$
- Cơ sở mới $\mathrm{B}' = \{(2,3), (-1,1)\}$
- Vector cần đổi là $u = (4,2)$ theo cơ sở cũ.

Bước 1:
Lập ma trận chuyển cơ sở P.
Cơ sở mới B′ được viết theo cơ sở cũ là:

$$\mathrm{P} = \begin{bmatrix} 5/2 & 0 \\ -1/2 & -1 \end{bmatrix}$$

Bước 2: Tìm toạ độ mới.
Tính nghịch đảo của P:

det(P)=(5/2).(-1)– (-1/2).0= -5/2

$$P^{-1} = \frac{1}{\det(P)} . \begin{bmatrix} -1 & 1/2 \\ 0 & 5/2 \end{bmatrix} = \begin{bmatrix} 2/5 & -1/5 \\ 0 & -1 \end{bmatrix}$$

Nhân với toạ độ u trong cơ sở cũ:

$$[u]_{\mathrm{B}'} = \begin{bmatrix} 2/5 & -1/5 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 6/5 \\ -2 \end{bmatrix}$$

Không gian định chuẩn:

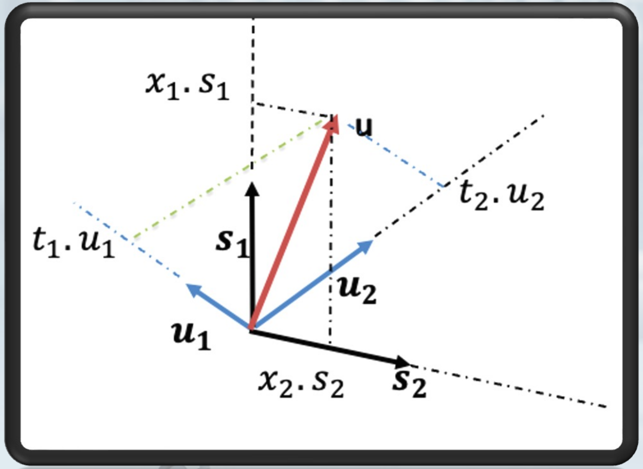
Biểu diễn bằng hình học:

Để dễ hình dung, xét 2 cơ sở

$S = \{s_1, s_2\}$, $B = \{u_1, u_2\}$ trong $\mathbb{R}^2$:

$u = x_1.s_1, +x_2.s_2$

$= t_1.u_1, +t_2.u_2$



Ví dụ:

1. Trong $\mathbb{R}^3$ cho: B = {v_1= (1,0,1), v_2= (1, 1, 0), v_3= (0, 1, 1)};
   S= {s_1 = (1, 2, 3), s_2 = (0, 1, 0), s_3 = (0, 0, 1)}
   (1) C/m S và B là các cơ sở của $\mathbb{R}^3$;(2) Tìm ma trận chuyển từ B sang S; (3) Tìm tọa độ của véc tơ: u = 2. s_1 – 3s_2 + 5.s_3 trong cơ sở B.

Giải: Có $\det(S^*) = 1 \neq 0, \det(B^*) = 2 \neq 0$, nên S, B là các cơ sở của $\mathbb{R}^3$.

Có sơ đồ biến đổi:

$$(B^*|S^*) = \left(\begin{array}{ccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 2 & 1 & 0 \\ 1 & 0 & 1 & 3 & 0 & 1 \end{array}\right) \xrightarrow[h3-h1+h2\to h3]{h2-h3-h1\to h2} \left(\begin{array}{ccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & -1 \\ 0 & 0 & 2 & 4 & 1 & 1 \end{array}\right)$$

$$\xrightarrow{2h1-h2\to h1} \left(\begin{array}{ccc|ccc} 2 & 0 & 0 & 2 & -1 & 1 \\ 0 & 2 & 0 & 0 & 1 & -1 \\ 0 & 0 & 2 & 4 & 1 & 1 \end{array}\right) \xrightarrow{\substack{\frac{1}{2}h1\to h1 \\ \frac{1}{2}h2\to h2;\ \frac{1}{2}h3\to h3}} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -0{,}5 & 0{,}5 \\ 0 & 1 & 0 & 0 & 0{,}5 & -0{,}5 \\ 0 & 0 & 1 & 2 & 0{,}5 & 0{,}5 \end{array}\right)$$

$$\Rightarrow P_{(B\to S)} = A = \begin{pmatrix} 1 & -0{,}5 & 0{,}5 \\ 0 & 0{,}5 & -0{,}5 \\ 2 & 0{,}5 & 0{,}5 \end{pmatrix}.$$

Ta có: $[u]_B = A.[u]_S = \begin{pmatrix} 1 & -0{,}5 & 0{,}5 \\ 0 & 0{,}5 & -0{,}5 \\ 2 & 0{,}5 & 0{,}5 \end{pmatrix}.\begin{pmatrix} 2 \\ -3 \\ 5 \end{pmatrix} = \begin{pmatrix} 6 \\ -4 \\ 1 \end{pmatrix}$

1. Trong $\mathbb{R}^4$ cho:
   B = {v_1= (1,1,1,1), v_2= (0, 1, 1,1), v_3= (1, 0,1,1), v_4 = (1,1,0,1)};
   S = {s_1 = (1, 1, 0, 0), s_2 = (0, 1, 1, 0), s_3 = (0, 0,1, 1), s_4 = (1, 0, 0, 0)}
   (1) C/m S và B là các cơ sở của $\mathbb{R}^4$; (2)Tìm ma trận chuyển từ B sang S.
   (3) Tìm tọa độ của véc tơ: u trong cơ sở B, biết $(u)_S = (1, -2, 3, -4)$.

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Quá trình trực giao hóa của Gram-Schmidt là một phương pháp trong đại số tuyến tính dùng để biến một tập hợp các vector độc lập tuyến tính thành một tập hợp các vector trực giao (hoặc trực chuẩn nếu chuẩn hoá) trong một không gian vector có tích trong.

Cho V là không gian vetor có tích vô hướng, $B = \{v_1, v_2, \dots, v_n\}$ là họ vector độc lập tuyến tính của V. Từ B, cần xây dựng một họ vector trực chuẩn $S = \{u_1, u_2, \dots, u_n\}$ sao cho span(S) = span(B). Quá trình gồm các bước:

B1: Chọn: $\mathrm{u}_1 = \frac{1}{\|\mathrm{v}_1\|}.\mathrm{v}_1$, ta có: $\|\mathrm{u}_1\| = 1$ và: $\mathrm{span}\{\mathrm{u}_1\} = \mathrm{span}\{\mathrm{v}_1\}$

B2: Tìm u_2 để họ $\{\mathrm{u}_1, \mathrm{u}_2\}$ trực chuẩn, bằng cách đặt: $\mathrm{u}_2' = \mathrm{v}_2 + \mathrm{t}\mathrm{u}_1$, và chọn t từ: $\langle \mathrm{u}_2', \mathrm{u}_1\rangle = 0 \Leftrightarrow \mathrm{t} = -\langle \mathrm{v}_2, \mathrm{u}_1\rangle$. Đặt: $\mathrm{u}_2 = \frac{1}{\|\mathrm{u}_2'\|}.\mathrm{u'}_2$, thì $\{\mathrm{u}_1, \mathrm{u}_2\}$ là họ trực chuẩn và ta có: $\mathrm{span}(\{\mathrm{u}_1, \mathrm{u}_2\}) = \mathrm{span}(\{\mathrm{v}_1, \mathrm{v}_2\})$.

B3: *Giả sử đã xây dựng được họ:* $S_{k-1} = \{u_1, u_2, \dots, u_{k-1}\}$ là họ trực chuẩn sao cho: $span(S_i) = span(\{v_1, v_2, \dots, v_i\}), \forall i = \overline{1, k-1}$.
Khi đó ta tiếp tục xây dựng: $u_k' = v_k + t_1 u_1 + t_2 u_2 + \dots + t_{k-1} u_{k-1}$,
Với t_j tìm được từ đ/k: $\langle u_k', u_j\rangle = 0, hay\ t_j = -\langle v_k, u_j\rangle, \forall j = \overline{1, k-1}$.
Từ đó chọn: $u_k = \frac{1}{\|u_k'\|}.u_k'$, và nhận được: $S_k = \{u_1, u_2, \dots, u_k\}$ là họ trực chuẩn mà: $span(S_k) = span(\{v_1, v_2, \dots, v_k\})$.

Quá trình xây dựng họ vector trực chuẩn $S = \{u_1, u, \dots, u_n\}$ nói trên được gọi là quá trình trực chuẩn hóa Gram- Smidt.
Như vậy bằng quy nạp, ta đã chứng minh được kết quả sau:
Đ.lí 4: Với $B = \{v_1, v_2, \dots, v_n\}$ là họ vector độc lập tuyến tính trong không gian vector thực V, ta tìm được họ vector trực chuẩn $S = \{u_1, u_2, \dots, u_n\}$ sao cho: $span(\{u_1, u_2, \dots, u_k\}) = span(\{v_1, v_2, \dots, v_k\}), với\ mọi\ k = 1,2, \dots, n$.

Không gian định chuẩn:

Ví dụ:
Trong $\mathbb{R}^3$,bằng quá trình Gram-Schmidt hãy trực chuẩn hóa hệ vector:

$$B = \{v_1 = (1,1,1), v_2 = (1,1,0), v_3 = (0,1,1)\}$$

Giải: Dễ thấy B là họ vector độc lập tuyến tính trong $\mathbb{R}^3$.

Đặt $u_1 = \frac{1}{\|v_1\|}.v_1 = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right)$; $u_2' = v_2 + tu_1$, *với* $t = -\langle v_2, u_1\rangle = -\frac{2}{\sqrt{3}} \Rightarrow u_2' = \left(\frac{1}{3}, \frac{1}{3}, -\frac{2}{3}\right)$. *Chọn:* $u_2 = \frac{1}{\|u_2'\|}.u_2' = \left(\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, -\frac{2}{\sqrt{6}}\right)$.

$$\text{Đặt}: u_3' = v_3 + au_1 + bu_2, \text{với}: a = -\langle v_3, u_1\rangle = -\frac{2}{\sqrt{3}}; b = -\langle v_3, u_2\rangle = \frac{1}{\sqrt{6}}$$

$\Rightarrow u_3' = \left(-\frac{1}{2}, \frac{1}{2}, 0\right)$. Chọn $u_3 = \frac{1}{\|u_3'\|}.u_3' = \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0\right)$. Hệ cần tìm là:

$$S = \{u_1 = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right), u_2 = \left(\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, -\frac{2}{\sqrt{6}}\right), u_3 = \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0\right)\}.$$

Bài tập: Trong $\mathbb{R}^3$,bằng quá trình Gram-Smidt hãy trực chuẩn hóa hệ vector:

$$B = \{v_1 = (1,-1,1), v_2 = (-1,1,1), v_3 = (1,1,-1)\}.$$

KHÔNG GIAN VECTOR

Không gian định chuẩn:

Ứng dụng:

Trong ML, nếu các đặc trưng có tương quan cao, nó có thể gây ra các vấn đề trong huấn luyện mô hình như sau:

- Ảnh hưởng đến độ chính xác.
- Gây ra vấn đề khi sử dụng hồi quy tuyến tính hoặc PCA.

Ví dụ:

- Giả sử bạn có 2 biến đầu vào trong mô hình hồi quy tuyến tính (Linear Regression):
  - ✓ $v_1 = (170, 165, 180)$. (chiều cao)
  - ✓ $v_2 = (70, 65, 80)$ (cân nặng)

  Hai biến chiều cao và cân nặng có tương quan cao.

  Sử dụng Gram-Schmidt để trực giao hoá tạo ra biến $v_2' = (-1.06, -4.07, 4.76)$ trực giao với v_1 giúp giảm tương quan và cải thiện độ chính xác của mô hình.

  v_2' có thể được hiểu là phần trọng lượng dư thừa hoặc thiếu hụt không phụ thuộc vào chiều cao.

```python
import numpy as np

def gram_schmidt(V):
    """
    Thực hiện trực chuẩn hóa Gram-Schmidt trên tập hợp vector V.

    Args:
        V: Ma trận numpy (m x n), mỗi cột là một vector đầu vào.

    Returns:
        U: Ma trận chứa các vector trực chuẩn.
    """
    V = np.array(V, dtype=np.float64)
    m, n = V.shape
    U = []

    for i in range(n):
        u_i = V[:, i]  # Lấy vector hiện tại
        for u_j in U:  # Loại bỏ thành phần chiếu lên các vector trước đó
            proj = (np.dot(u_j, u_i) / np.dot(u_j, u_j)) * u_j
            u_i = u_i - proj

        norm_u_i = np.linalg.norm(u_i)  # Chuẩn của vector
        if norm_u_i > 1e-10:  # Kiểm tra nếu vector không bị triệt tiêu hoàn toàn
            U.append(u_i / norm_u_i)  # Chuẩn hóa vector

    return np.array(U).T  # Chuyển dạng ma trận để mỗi cột là một vector trực chuẩn

# Nhập số vector và số chiều từ bàn phím
num_vectors = int(input("Nhập số lượng vector: "))
vector_dim = int(input("Nhập số chiều của vector: "))

# Nhập các vector từ bàn phím
vectors = []
print(f"Nhập từng vector (mỗi phần tử cách nhau bằng dấu cách):")
for i in range(num_vectors):
    vector = list(map(float, input(f"Vector {i+1}: ").split()))
    if len(vector) != vector_dim:
        print("❌ Lỗi: Số phần tử không đúng, vui lòng nhập lại!")
        exit()
    vectors.append(vector)

# Chuyển đổi sang numpy array và thực hiện Gram-Schmidt
V = np.array(vectors).T  # Chuyển ma trận để mỗi cột là một vector
U = gram_schmidt(V)

# In kết quả
print("\n🔹 Các vector trực chuẩn hóa:")
print(U.T)  # Chuyển lại dạng vector ban đầu
```

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Các khái niệm về ánh xạ tuyến tính:
Ánh xạ tuyến tính là một khái niệm quan trọng trong đại số tuyến tính, giúp mô tả các phép biến đổi giữa các không gian vector.
Định nghĩa: Cho X, Y là các không gian vector trên cùng một trường số K (thực hoặc phức)
Mỗi ánh xạ $f: X \to Y$ được gọi là một ánh xạ tuyến tính (hay một đồng cấu tuyến tính) từ X sang Y, nếu có các tính chất:

1. Tính cộng tính: $f(u + v) = f(u) + f(v), \forall u, v \in X$
2. Tính thuần nhất: $f(\alpha.u) = \alpha.f(u), \forall \alpha \in K, \forall u \in X$,

Nói cách khác, ánh xạ tuyến tính bảo toàn phép cộng và phép nhân vô hướng.

Ảnh và nhân của ánh xạ tuyến tính:

- Ảnh của f (ký hiệu $Im(f)$):

$$\text{Với ánh xạ tuyến tính } f: X \to Y, \text{ ký hiệu: } Im(f) = \{f(u):\ u \in X\}$$

Tập hợp tất cả các giá trị mà f có thể đạt được trong Y

- Nhân của f (ký hiệu $Ker(f)$):

$$Ker(f) = \{u \in X:\ f(u) = \mathcal{O}_Y \text{ (vector không của Y}\}$$

Tập hợp tất cả các vector trong X bị ánh xạ về vector $\mathcal{O}$ trong Y

- Điều kiện để một ánh xạ là tuyến tính:

$$\text{Một ánh xạ } f: X \to Y \text{ là ánh xạ tuyến tính khi và chỉ khi: } f(\alpha.u + \beta.v) = \alpha.f(u) + \beta.f(v), \forall u, v \in X, \forall \alpha, \beta \in K$$

Tức là nó phải giữ được cả phép cộng và phép nhân vô hướng đồng thời.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Một số định lý quan trọng:

Giả sử f: X →Y là ánh xạ tuyến tính. Khi đó:

1. $f(\mathcal{O}_X) = \mathcal{O}_Y$. (Ánh xạ tuyến tính luôn ánh xạ vector $\mathcal{O}$ về vector $\mathcal{O}$)
2. $\mathrm{Im}(f)$ là không gian con của Y, $Ker(f)$ là không gian con của X.
3. f là toàn ánh ⇔ $\mathrm{Im}(f) = \mathrm{Y}$ (tức là f bao phủ toàn bộ không gian Y)
4. f là đơn ánh ⇔ $Ker(\mathrm{f}) = \{\mathcal{O}_X\}$ (tức là ánh xạ không làm sụp đổ bất kỳ vector nào khác ngoài vector $\mathcal{O}$)
5. Nếu S là hệ vector độc lập tuyến tính trong X thì f(S) là hệ vector độc lập tuyến tính trong $Im(f)$
6. Nếu X là không gian vector n chiều thì: $\dim(\mathrm{Im}(f)) + \dim(\mathrm{Ker}(f)) = n$.

   Đây được gọi là định lý hạng-nhân(rank-nullity theore), trong đó:

   - $\dim(\mathrm{Im}(f))$ là hạng của ánh xạ tuyến tính f(số chiều của ảnh)
   - $\dim(\mathrm{Ker}(f))$ là số khuyết của của ánh xạ tuyến tính f (số chiều của nhân)

Ví dụ: $f: R^{m\times n} \to R^{n\times m}, f(A) = f(A^T)$ (phép chuyển vị) là một ánh xạ truyến tính từ $R^{m\times n}$ sang $R^{n\times m}$. f là một song ánh, vì $\mathrm{Im(f)} = R^{n\times m}$ (toàn ánh) và $\mathrm{Ker(f)} = \{\mathrm{A} \in \mathrm{R}^{m\times n}\colon \mathrm{A}^T = \mathcal{O}_{n\times m}\} = \mathcal{O}_{m\times n}$ (đơn ánh)

Suy ra: hạng của f = $\dim(\mathrm{Im}(f)) = \dim(R^{n\times m})$ = m.n, và f có số khuyết = 0.

Bài tập: Cho $f: \mathrm{R}^2 \to \mathrm{R}^3: f(x,y) = (x, y, x+y)$

a. Chứng minh f là ánh xạ tuyến tính

b. Tìm $Im(f)$ và $Ker(f)$. Tìm hạng và số khuyết của f.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Phép biến đổi tuyến tính:
Mỗi ánh xạ tuyến tính f: X $\rightarrow$ X từ một không gian vector X vào chính nó gọi là một phép biến đổi tuyến tính trong X.

Ví dụ1: Phép đối xứng qua đường phân giác thứ nhất:

$$\text{Xét ánh xạ tuyến tính } f:\ \mathrm{R}^2 \rightarrow \mathrm{R}^2: \mathrm{f}(\mathrm{x},\mathrm{y}) = (\mathrm{y},\mathrm{x}), \forall(\mathrm{x},\mathrm{y}) \in \mathrm{R}^2$$

Đây là phép đối xứng qua đường phân giác y=x.
Ta có thể kiểm tra f là một song ánh vì:

- Đơn ánh: Nếu $f(x,y) = (0{,}0) \Longrightarrow (y,x) = (0{,}0) \Longrightarrow x = 0, y = 0$
- Toàn ánh: Với mọi $(\mathrm{a},\mathrm{b}) \in \mathrm{R}^2$, ta có thể tìm được (x,y) sao cho $f(x,y) = (a,b)$ bằng cách đặt $x = b\ y = a$

Vậy f là một phép biến đổi tuyến tính và là một song ánh

Ví dụ 2: Phép ánh xạ ma trận về ma trận đường chéo:

$$\text{Xét ánh xạ tuyến tính } f: R^{n\times n} \rightarrow R^{n\times \mathrm{n}} \text{ được xác định bởi } f(\mathrm{A}) = \mathrm{A}^*,$$

trong đó A^* là ma trận đường chéo với các phần tử trên đường chéo chính bằng chính các phần tử của A
Dễ dàng kiểm chứng f là một phép biến đổi tuyến tính trong $R^{n\times \mathrm{n}}$ có:

- hạng(f) = dim(Im(f)) = n
- số khuyết là: dim(Ker(f)) = n.n – n = (n – 1).n

Ví dụ 3: Phép đạo hàm đa thức:
Xét ánh xạ f trên không gian đa thức bậc n:

$$f: P_n[x] \rightarrow P_n[x]: f(p) = p'(\text{phép lấy đạo hàm}), \forall p \in P_n[x].$$

Dễ thấy f là phép biến đổi tuyến tính trong $P_n[x]$ với Im(f) = $P_{n-1}[x]$, Ker(f) = $P_0[x]$ (các đa thức bậc 0), có hạng(f) = dim($P_{n-1}[x]$) = n và số khuyết = 1

Bài tập: Cho $A = \left(a_{ij}\right)_{m\times n}$, c/m: f: $\mathbb{R}^{\mathrm{n}} \rightarrow \mathbb{R}^{\mathrm{m}}$: $[f(x)] = A.[x]$, ($[x]$ là cột tọa độ của x trong cơ sở chính tắc) là ánh xạ tuyến tính. Hãy mô tả Ker(f)

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Định nghĩa:
Giả sử X, Y là hai không gian vector trên cùng một trường số K
Mỗi ánh xạ tuyến tính f: X → Y có thể thuộc một trong các loại sau:

1. Đơn cấu (Injective – One-to-One):
   - f được gọi là đơn cấu nếu $f(x) = 0 \Rightarrow x = 0$.
   - Tức là ker của f chỉ chứa phần tử 0: $Ker(f) = \{0\}$
   - Điều này có nghĩa là f không làm mất thông tin, tức là không có 2 phần tử khác nhau trong X ánh xạ vào cùng một phần tử trong Y
2. Toàn cấu (Surjective- Onto):
   - f được gọi là toàn cấu nếu mọi phần tử trong Y đều có ít nhất một ảnh ngược trong X
   - Tức là tầm ảnh (image) của f bằng toàn bộ Y: Im(f)=Y
   - Điều này có nghĩa là f phủ toàn bộ không gian đích, không có phần tử nào trong Y bị bỏ sót.
3. Đẳng cấu(Isomorphism):
   - Nếu f vừa đơn cấu vừa toàn cấu, thì f là một đẳng cấu.
   - Khi đó tồn tại ánh xạ ngược f^{-1}: X → Y cũng là một ánh xạ tuyến tính.

Định lý:
Hai không gian vector hữu hạn chiều X và Y trên cùng một trường K là đẳng cấu với nhau khi và chỉ khi chúng có cùng số chiều:

$$\dim(X) = \dim(Y)$$

Nhờ vào kết quả trên mà đối với các phép biến đổi tuyến tính trên các không gian vector hữu hạn chiều, chỉ cần khảo sát trên không gian $\mathbb{R}^n$

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Ánh xạ tuyến tính trong không gian hữu hạn chiều:

Giả sử X là một không gian vector có số chiều n, với một cơ sở B = $\{u_1, u_2, ..., u_n\}$, và Y là một không gian vector có số chiều m, với một cơ sở S = $\{v_1, v_2, \dots, v_m\}$.
Cho một ánh xạ tuyến tính f: X $\rightarrow$ Y . Do tính chất của không gian vector, mọi vector u $\in X$ có thể biểu diễn duy nhất dưới dạng tổ hợp tuyến tính của các vector trong cơ sở B:

$$u = \alpha_1 u_1 + \alpha_2 u_2 + \cdots + \alpha_n u_n$$

Do f là ánh xạ tuyến tính, ta có:

$$f(u) = \alpha_1 f(u_1) + \alpha_2 f(u_2) + \cdots + \alpha_n f(u_n)$$

Vậy để xác định hoàn toàn f, ta chỉ cần biết ảnh của các vector trong cơ sở B, tức là $f(u_1), f(u_2), \dots, f(u_n)$.
Gỉa sử rằng trong không gian Y, mỗi $f(u_j)$ có toạ độ trong cơ sở S là:

$$f(u_j) = a_{1j} v_1 + a_{2j} v_2 + \cdots + a_{mj} v_m$$

Khi đó, ma trận A của ánh xạ tuyến tính f trong các cơ sở B và S là:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Trong đó, cột thứ j của ma trận A chính là toạ độ của $f(u_j)$ trong cơ sở S.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Ánh xạ tuyến tính trong không gian hữu hạn chiều:
Công thức tổng quát cho ảnh của một vector tuỳ ý:

Mọi vector $u \in X$ có thể biểu diễn theo cơ sở B:

$$u = \sum_{j=1}^{n} x_j u_j$$

với $[u]_B = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ là toạ độ của u trong cơ sở B

Khi áp dụng ánh xạ f:

$$f(u) = f\left(\sum_{j=1}^{n} x_j u_j\right) = \sum\nolimits_{j=1}^{n} x_j f(u_j)$$

Thay $f(u_j) = \sum_{i=1}^{m} a_{ij} v_i$:

$$f(u) = \sum\nolimits_{j=1}^{n} x_j \sum\nolimits_{i=1}^{m} a_{ij} v_i = \sum\nolimits_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j\right) v_i$$

Toạ độ $f(u)$ trong cơ sở S là:

$$[f(u)]_S = \begin{bmatrix} \sum_{j=1}^{n} a_{1j} x_j \\ \sum_{j=1}^{n} a_{2j} x_j \\ \vdots \\ \sum_{j=1}^{n} a_{mj} x_j \end{bmatrix}$$

Hay: $[f(u)]_S = A[u]_B$

Nhận xét:

- Tương ứng 1-1 giữa ma trận $A \in \mathrm{R}^{m\times n}$ và ánh xạ tuyến tính f: X → Y

Ý nghĩa:

Mỗi ánh xạ tuyến tính f: X → Y (trong đó X có không gian chiều n và Y có không gian chiều m) đều có thể được biểu diễn dưới dạng một ma trận A kích thước mxn. Đồng thời, mỗi ma trận A cũng xác định một ánh xạ tuyến tính duy nhất f: X → Y

- Hạng của ma trận A bằng số chiều của $Im(f)$

Ý nghĩa:

Hạng của ma trận A chính là số chiều của ảnh của ánh xạ tuyến tính f tức là số chiều của không gian $Im(f)$. Do đó, ta có: rank(A)=dim(Im(f))

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Ví dụ: Cho ánh xạ f: $\mathbb{R}^3 \to \mathbb{R}^4$, f(x, y, z) = (2x, x + y, x – y , x + z)
(1) Chứng minh f là ánh xạ tuyến tính; (2) Tìm Imf, Kerf; (3)Tìm ma trận A của f trong cặp cơ sở chính tắc của $\mathbb{R}^3$ và $\mathbb{R}^4$.
Giải:

1. (S/v tự kiểm chứng)
2. $Kerf = \{(x,y,z) \in \mathbb{R}^3 : (2x, x+y, x-y, x+z) = (0,0,0,0)\} = \{(0,0,0)\}$
   Cơ sở chính tắc của $\mathbb{R}^3$ là $\Phi_3 = \{e_1 = (1,0,0), e_2 = (0,1,0), e_3 = (0,0,1)\}$.
   Ta có:
   $f(e_1) = (2, 1, 1, 1)$, $f(e_2) = (0, 1, -1, 0)$, $f(e_3) = (0, 0, 0, 1)$
   $Imf = Span\{f(e_1), f(e_2), f(e_3)\} = Span\{(2,1,1,1), (0,1,-1,0), (0,0,0,1)\}$.
   Imf là kg con ba chiều của $\mathbb{R}^4$, nhận $\{f(e_1), f(e_2), f(e_3)\}$ làm một cơ sở.
3. Cơ sở chính tắc của $\mathbb{R}^4$ là $\Phi_4 = \{u_1 = (1,0,0,0), u_2 = (0,1,0,0), u_3 = (0,0,0,1), u_4 = (0,0,0,1)\}$.
   Ta có: $f(e_1) = (2, 1, 1, 1) = 2.u_1 + u_2 + u_3 + u_4$; $f(e_2) = (0, 1, -1, 0) = 0.u_1 + 1.u_2 - 1.u_3 + 0.u_4$; $f(e_3) = (0, 0, 0, 1) = 0.u_1 + 0.u_2 + 0.u_3 + u_4$;
   Ma trận của f trong cặp cơ sở chính tắc của $\mathbb{R}^3$ và $\mathbb{R}^4$ là:
   $$A = \begin{pmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$
   Ma trận A có hạng bằng 3 (số chiều của Imf).

```python
import numpy as np
def is_linear_mapping(A):
    """
    Kiểm tra ánh xạ có phải là ánh xạ tuyến tính không.
    """
    return True  # Mọi ánh xạ biểu diễn bởi ma trận đều là tuyến tính
def find_image(A):
    """
    Tìm ảnh của ánh xạ tuyến tính (Im(f)), tức là không gian cột của ma trận A.
    """
    U, S, Vt = np.linalg.svd(A)  # Phân rã SVD để tìm các vector độc lập
    rank = np.linalg.matrix_rank(A)  # Hạng của A
    image_basis = U[:, :rank]  # Cơ sở của không gian cột (Im(f))
    return image_basis
def find_kernel(A):
    """
    Tìm hạt nhân của ánh xạ tuyến tính (Ker(f)), tức là tập nghiệm của Ax = 0.
    """
    m, n = A.shape
    U, S, Vt = np.linalg.svd(A)  # Phân rã SVD để tìm không gian null
    null_space = Vt[len(S):].T  # Vector ứng với giá trị suy biến
    return null_space
# Nhập kích thước của ma trận ánh xạ tuyến tính
m = int(input("Nhập số hàng của ma trận ánh xạ: "))  # Số chiều đích
n = int(input("Nhập số cột của ma trận ánh xạ: "))  # Số chiều miền xác định
# Nhập ma trận từ bàn phím
print(f"Nhập ma trận ánh xạ tuyến tính {m}x{n} (mỗi hàng nhập trên 1 dòng, các phần tử cách nhau dấu cách):")
A = np.array([list(map(float, input().split())) for _ in range(m)])
# Kiểm tra ánh xạ có phải tuyến tính không
if is_linear_mapping(A):
    print("\n✅ Ánh xạ f là tuyến tính.")
# Tìm ảnh của f (Im(f))
image = find_image(A)
print("\n🔷 Ảnh của f (Im(f)):")
print(image)
# Tìm hạt nhân của f (Ker(f))
kernel = find_kernel(A)
print("\n🔷 Hạt nhân của f (Ker(f)):")
print(kernel if kernel.size > 0 else "ø (Không có vector nào)")
# Tính hạng và số khuyết
rank = np.linalg.matrix_rank(A)
nullity = A.shape[1] - rank  # Số khuyết = số biến - hạng
print(f"\n✅ Hạng của f (rank): {rank}")
print(f"✅ Số khuyết của f (nullity): {nullity}")
# Nhập vector cần ánh xạ
print("\nNhập vector đầu vào để ánh xạ (kích thước phải là {}x1):".format(n))
vector = np.array(list(map(float, input().split()))).reshape(n, 1)
# Kiểm tra kích thước hợp lệ
if vector.shape[0] != n:
    print("\n⚠️ Lỗi: Kích thước vector đầu vào không khớp với miền xác định!")
else:
    result = A @ vector  # Tính f(vector) = A * vector
    print("\n🔷 Kết quả ánh xạ của vector:")
    print(result)
```

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Định lý:
Cho f, g là các ánh xạ tuyến tính từ không gian vector n chiều X sang không gian vector m chiều Y. Giả sử A và G lần lượt là ma trận biểu diễn của f và g trong cặp cơ sở B của X và S của Y.

1. Ánh xạ tuyến tính $a.f\,(a \in K)$ có ma trận tương ứng là $a.$A
$$(a.f)(x) = a.f(x) = a(A.[x]_B) = a.A.[x]_B$$
Do đó, ma trận của ánh xạ $a.f$ chính $a.$A
2. Ánh xạ tuyến tính $f + g$ có ma trận tương ứng A $+ G$
$$(f+g)(x) = f(x) + g(x) = A.[x]_B + \mathrm{G}[x]_B = (A+G)[x]_B$$
Do đó, ma trận biểu diễn của $f + g$ trong cặp cơ sở B – S chính là A $+ G$

Chú ý:

1. Nếu f: X $\longrightarrow$ X là ánh xạ tuyến tính và B là một cơ cở có thứ tự của X, thì ma trận của f trong cặp cơ sở B – B gọi là ma trận của f trong cơ sở B.
2. Nếu $f: \mathbb{R}^n \longrightarrow \mathbb{R}^m$ là ánh xạ tuyến tính, thì ma trận của f trong cặp cơ sở chính tắc của $\mathbb{R}^n$ và của $\mathbb{R}^m$ được gọi đơn giản là ma trận chính tắc của f

Ví dụ: Giả sử ánh xạ tuyến tính $f: \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ có ma trận chính tắc $A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix}$ và ánh xạ tuyến tính $h: \mathbb{R}^3 \longrightarrow \mathbb{R}^2$ có ma trận chính tắc $H = \begin{pmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \end{pmatrix}$. Hãy tìm $h \circ f(2,-1)$

Giải: Ánh xạ $h \circ f: \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ là ánh xạ tuyến tính có ma trận chính tắc là:
$$C = H.A = \begin{pmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \end{pmatrix}.\begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$
Vậy : $\left(h \circ f(2,-1)\right)^T = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}.\begin{bmatrix} 2 \\ -1 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \end{bmatrix}$, hay: $h \circ f(2,-1) = (3,-1)$.

KHÔNG GIAN VECTOR

Ma trận của phép biến đổi tuyến tính qua phép đổi cơ sở:
Trong mục này ta luôn giả thiết V là không gian vector n chiều.
Ma trận đồng dạng
Định nghĩa: Giả sử A và B là hai ma trận vuông cấp n. Nói ***B đồng dạng với A***, nếu tồn tại một ma trận vuông cấp n khả nghịch P sao cho:

$$B = P^{-1}.A.P$$

Nhận xét: Nếu B đồng dạng với A thì A cũng đồng dạng với B, vì:

$$B = P^{-1}.A.P \Leftrightarrow A = P.B.P^{-1} = Q^{-1}.B.Q\ (Q = P^{-1}).$$

Vì vậy ta nói: A và B đồng dạng với nhau, và ký hiệu: $\boldsymbol{A \sim B}$.
Đ.lý 5: Giả sử $f: V \longrightarrow V$ là phép biến đổi tuyến tính trên V. Khi đó nếu A là m.trận của f trong cơ sở B và A' là ma trận của của f trong cơ sở B', thì A và A' đồng dạng với nhau và: $A' = P^{-1}.A.P$, trong đó: $P = P_{(B \to B')}$.

C/m: Theo giả thiết, ta có: $\begin{cases} A.[x]_B = [f(x)]_B \\ A'.[x]_{B'} = [f(x)]_{B'} \\ [x]_B = P.[x]_{B'} \end{cases}, \forall x \in V.$

Từ đó: $[f(x)]_{B'} = P^{-1}.[f(x)]_B = P^{-1}.A.[x]_B = P^{-1}.A.P.[x]_{B'}$
Hệ thức này cho thấy ma trận của f trong cơ sở B' là: $A' = P^{-1}.A.P$.
Ví dụ: Cho: $f: \mathbb{R}^2 \longrightarrow \mathbb{R}^2: f(x,y) = (x + y, 4y - 2x)$. Hãy tìm ma trận của f trong cơ sở $B' = \{u_1 = (1,1), u_2 = (1,2)\}$.
Giải: Với cơ sở chính tắc: $\Phi = \{e_1 = (1,0), e_2 = (0,1)\}$, ta có:
$f(e_1) = (1,-2), f(e_2) = (1,4)$, vậy f có ma trận chính tắc: $\begin{bmatrix} 1 & 1 \\ -2 & 4 \end{bmatrix}$. Ma trận chuyển từ Φ sang B' là: $P = P_{(\Phi \to B')} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$ có:
$P^{-1} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$. Từ đó, theo đ.lí 5, ta có ma trận của f trong cơ sở B' là: $A' = P^{-1}.A.P = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$

KHÔNG GIAN VECTOR

Trị riêng và vector riêng của phép biến đổi tuyến tính:

Cho X là không gian vector trên trường K, $f\colon X \to X$ là một phép biến đổi tuyến tính. Nếu có $x \in X, x \neq \mathcal{O}$ và số $\lambda \in K$, sao cho: $f(x) = \lambda.x$ thì ta nói: λ là một trị riêng của f và x là vector riêng của f ứng với trị riêng λ.

Ví dụ: Với ánh xạ tuyến tính f: $\mathbb{R}^2 \to \mathbb{R}^2$, f(x, y) = (y, x), ta có:

* $f(e_1) = f(1,0) = (0,1) = e_2; f(e_2) = f(0,1) = (1,0) = e_1$. Vậy f có ma trận chính tắc: $A = ([f(e_1)], [f(e_2)]) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

* $f(x, y) = \lambda(x, y) \Leftrightarrow$ (y, x) = λ.(x, y) ⇔ λ = 1, x = y, hoặc λ = - 1, x = - y,
hoặc λ = 0, x = y = 0. Vậy f có hai trị riêng: λ = 1, λ = - 1. Các vector riêng ứng với λ = 1 là: (x, x) (x ≠ 0). Các vector riêng ứng với λ = -1 là: (x, - x) (x ≠ 0).

Ví dụ: Tìm trị riêng và véc tơ riêng của: $f\colon \mathbb{R}^3 \to \mathbb{R}^3 : f(x, y, z) = (z, x, y)$,

Ví dụ: $C^1(\mathbb{R})$ là không gian các hàm số một biến số thực, có đạo hàm liên tục trên $\mathbb{R}$, và $C(\mathbb{R})$ là k.gian các hàm số một biến số thực, liên tục trên $\mathbb{R}$. Với phép cộng các hàm và phép nhân một hàm với một số thực thì $C^1(\mathbb{R}), C(\mathbb{R})$ là các không gian vector thực ($C^1(\mathbb{R})$ là không gian con của $C(\mathbb{R})$). Xét ánh xạ: $d\colon C^1(\mathbb{R}) \to C(\mathbb{R}) : d(f) = f'$ thì d là ánh xạ tuyến tính. Khi đó hệ thức:

$$d(f) = \lambda.f \ (f \neq 0) \Leftrightarrow f' = \lambda.f \ (f \neq 0) \Leftrightarrow f(x) = c.e^{\lambda x} (c \neq 0), \forall x \in \mathbb{R}.$$

Cho thấy mọi số thực λ đều là trị riêng của ánh xạ d (phép lấy đạo hàm) và khi đó các vector riêng tương ứng là họ hàm: $f(x) = c.e^{\lambda x} (c \neq 0)$.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Trị riêng và vector riêng của ma trận:

Ma trận trực giao:

Định nghĩa 1: $U \in R^{n\times n}$ được gọi là một *ma trận trực giao* nếu: $\boldsymbol{U.U^T = U^T.U = I_n}$

Định nghĩa2: $U \in \mathbb{C}^{n\times n}$ được gọi là một ma trận Unitary nếu $\boldsymbol{U.U^H = U^H.U = I_n}$

VD. Xét: I_n, $A = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix}$, $B = \frac{1}{3}\begin{bmatrix} -2 & 2 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}$; $U = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix}$

Dễ thấy I_n, A, B là các ma trận trực giao, U là ma trận Unitary

Các tính chất của ma trận trực giao

1. Nếu U là ma trận trực giao thì U^T cũng là ma trận trực giao và det(U) = 1
2. Ma trận vuông **U** là ma trận trực giao ⇔ U khả nghịch và $U^{-1} = U^T$
3. Ma trận trực giao U tương ứng với phép quay vector trong $\mathbb{R}^n$:

$$\forall x, y \in \mathbb{R}^n, \text{thì: } (Ux)^T.(Uy) = x^TU^T.Uy = x^T(U^T.U).y = x^T.y$$

   tức là phép biến đổi trực giao bảo toàn tích vô hướng của hai vector
   (ý nghĩa hình học: Ma trận trực giao U xác định một ánh xạ tuyến tính:

$$f: R^n \to R^n: f(x) = U.x$$

   mà nó bảo toàn góc giữa hai vector.)

KHÔNG GIAN VECTOR

Trị riêng và vector riêng của một ma trận vuông:

Định nghĩa: Cho m.trận vuông $A \in \mathrm{R}^{n\times n}$ và không gian vector X trên trường số K. Nếu có số $\lambda \in$ K và vector $\mathrm{x} \in \mathrm{X}, \mathrm{x} \neq \mathcal{O}$ sao cho: $\mathrm{A.x} = \lambda.\mathrm{x}$, thì ta nói λ là một trị riêng của ma trận A và x là một vector riêng của A tương ứng với trị riêng λ.

Nhận xét:

1. Trị riêng và vector riêng của ma trận vuông A cũng chính là trị riêng và vector riêng của phép biến đổi tuyến tính trong không gian n chiều V, nhận A làm ma trận trong một cơ sở B nào đó của V.
2. Nếu x là véc tơ riêng của A ứng với trị riêng λ, thì với mọi hằng số $c \neq 0$, véc tơ $c.x$ cũng là một vector riêng của A ứng với trị riêng λ.

VD1: Với $A = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$ và véc tơ $x = (2, 1)$, ta có: $A.x^T = \begin{pmatrix} 4 \\ 2 \end{pmatrix} = 2.x^T$. Vậy $\lambda = 2$ là một trị riêng của A và $x = (2, 1)$ là một vector riêng của A ứng với trị riêng $\lambda = 2$.

Phương trình đặc trưng. Việc tồn tại trị riêng λ và vector riêng tương ứng của ma trận A là sự tồn tại nghiệm không tầm thường của hệ phương trình tuyến tính: $A.x^T = \lambda.x^T$ (1)

hay: $(A - \lambda I).x^T = 0.$ (2)

Điều này có nghĩa là: $\det(A - \lambda I) = 0$ (3)

Đây là phương trình để tìm trị riêng λ, gọi là *phương trình đặc trưng* của ma trận vuông A.

$\det(A - \lambda I)$ là một đa thức đối với λ, gọi là *đa thức đặc trưng* của ma trận A.

KHÔNG GIAN VECTOR

Chú ý:

• (3) là phương trình đại số cấp n nên có n nghiệm trên trường số phức. Vậy một ma trận vuông cấp n có đúng n trị riêng (thực hoặc phức, đơn hoặc bội) khi xét trong không gian vector phức. Số bội của nghiệm λ của phương trình đặc trưng (3) được gọi là số bội đại số của λ.

Với λ là một nghiệm phức thì nó là một trị riêng của ma trận A nếu không gian vector là trên trường phức. x là một vector riêng ứng với trị riêng λ khi và chỉ khi x là nghiệm khác không của hệ phương trình tuyến tính thuần nhất (2). Như đã biết: tập nghiệm của (2) là một không gian con của $\mathbb{R}^{\mathrm{n}}$, gọi là không gian con riêng ứng với trị riêng λ, ký hiệu là N_λ:

$$N_\lambda = \{x \in \mathbb{R}^{\mathrm{n}}:\ (A - \lambda I).x^T\ = \mathcal{O}\}$$

Khi đó số chiều của N_λ được gọi là số bội hình học của λ.

Đ.lý 1*(về trị riêng của ma trận đối xứng)*. Nếu A là ma trận vuông cấp n, đối xứng thì A có đúng n trị riêng thực (gồm cả số bội của nghiệm) và n véc tơ riêng trực chuẩn tương ứng.

KHÔNG GIAN VECTOR

Cách tìm trị riêng và vector riêng của ma trận vuông A:

Từ các kết quả trên, có thể tìm trị riêng và véc tơ riêng của ma trận vuông A theo các bước sau:

1. Giải phương trình đặc trưng: $\det(A - \lambda I) = 0$, mỗi nghiệm của phương trình này là một trị riêng.
2. Với mỗi trị riêng λ, ta giải hệ phương trình: $(A - \lambda I).X = \mathcal{O}$,
   Từ đó tìm được không gian con riêng tương ứng N_λ.
   Tập các véc tơ riêng của A tương ứng với trị riêng λ là $N_\lambda \setminus \{\mathcal{O}\}$

VD2: Tìm các trị riêng và các véc tơ riêng tương ứng của ma trận: $A = \begin{bmatrix} -2 & 5 \\ -1 & 2 \end{bmatrix}$

Giải: A có phương trình đặc trưng:

$$\det(A - \lambda I) = 0 \Leftrightarrow \begin{vmatrix} -2-\lambda & 5 \\ -1 & 2-\lambda \end{vmatrix} = 0 \Leftrightarrow \lambda^2 + 1 = 0 \Leftrightarrow \left[\begin{array}{l} \lambda = \mathrm{i} \\ \lambda = -i \end{array}\right.,$$

Vậy A không có trị riêng thực (mà có 2 trị riêng phức $\lambda = i$ và $\lambda = -i$)

Vì thế A không có véc tơ riêng trên kgvt thực.

Tuy nhiên nếu xét trên kgvt phức thì hệ phương trình:

$$(A - \lambda I).X = 0 \Leftrightarrow \begin{cases} (-2-\lambda)x + 5y = 0 \\ -x + (2-\lambda)y = 0 \end{cases} \Leftrightarrow \begin{cases} (\lambda^2+1)y = 0 \\ x = (2-\lambda)y \end{cases}.$$

Với $\lambda = i$, hệ có tập nghiệm:

$$N_\lambda = \{(x,y): y = t(\text{tùy ý}), x = (2-i)t\} = \{t.(2-i,1): t \in \mathbb{R}\} = span\{u_1\}(u_1 = (2-i,1))$$

Tập các véc tơ riêng tương ứng là: $\{t.(2-i,1): 0 \neq t \in \mathbb{R}\}$

Với $\lambda = -i$, hệ có tập nghiệm:

$$N_\lambda = \{(x,y): y = t(\text{tùy ý}), x = (2+i)t\} = \{t.(2+i,1): t \in \mathbb{R}\} = span\{u_1\}(u_1 = (2+i,1))$$

Tập các véc tơ riêng tương ứng là: $\{t.(2+i,1): 0 \neq t \in \mathbb{R}\})$

KHÔNG GIAN VECTOR

Ví dụ: Tìm các trị riêng và các véc tơ riêng tương ứng của ma trận:

$$B = \begin{bmatrix} 3 & -2 & 0 \\ -2 & 3 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

Giải: Phương trình đặc trưng:

$$\det(B - \lambda.I) = 0 \Leftrightarrow \begin{vmatrix} 3-\lambda & -2 & 0 \\ -2 & 3-\lambda & 0 \\ 0 & 0 & 5-\lambda \end{vmatrix} = 0 \Leftrightarrow \left[\begin{array}{l} \lambda = 1(\text{đơn}) \\ \lambda = 5(bội\ 2) \end{array}\right.$$

Hệ phương trình:

$$(B - \lambda I).X = \mathcal{O} \Leftrightarrow \begin{bmatrix} 3-\lambda & -2 & 0 \\ -2 & 3-\lambda & 0 \\ 0 & 0 & 5-\lambda \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} (3-\lambda)x - 2y = 0 \\ -2x + (3-\lambda)y = 0\,. \\ (5-\lambda)z = 0 \end{cases} \quad (*). \text{ Từ đó:}$$

- Với $\lambda = 1$, (*) có không gian nghiệm:

$$N_\lambda = \{(t, t, 0) : t \in \mathbb{R}\} = span\{(1,1,0)\}$$

Các véc tơ riêng tương ứng là: $\{(t, t, 0) : 0 \neq t \in \mathbb{R}\}$

- Với $\lambda = 5$, (*) có không gian nghiệm:

$$N_\lambda = \{(t, -t, s) : t, s \in \mathbb{R}\} = \{t.(1, -1, 0) + s.(0,0,1), t, s \in \mathbb{R}\} = span\{(1, -1, 0), (0,0,1)\}$$

Các véc tơ riêng tương ứng là: $\{(t, -t, s) : t, s \in \mathbb{R}, t^2 + s^2 > 0\}$

KHÔNG GIAN VECTOR

Trị riêng và vector riêng của phép biến đổi tuyến tính trong không gian hữu hạn chiều:

Giả sử V là không gian vector n chiều, f: V $\rightarrow V$ là phép biến đổi tuyến tính và B = $\{u_1, u_2, \dots, u_n\}$ là một cơ sở sắp thứ tự trên V. Trong cơ sở B, f có ma trận A là ma trận vuông cấp n: $A = ([f(u_1)]_B, [f(u_2)]_B, \dots, [f(u_n)]_B)$

Vì:

$$f(x) = \lambda.x \Leftrightarrow A.[x]_B = \lambda.[x]_B$$

nên vector x là vector riêng của f, ứng với trị riêng λ, khi và chỉ khi véc tơ $[x]_B$ là vector riêng của ma trận A ứng với trị riêng λ.

Do đó việc tìm giá trị riêng và vector riêng của một phép biến đổi tuyến tính f: V $\rightarrow V$ trong không gian n chiều V được chuyển về tìm giá trị riêng và vector riêng của ma trận theo các bước sau:

1. Xác định một cơ sở sắp thứ tự B = $\{u_1, u_2, \dots, u_n\}$ trên V và tìm ma trận A của phép biến đổi tuyến tính f trong cơ sở B.
2. Tìm trị riêng và vector riêng của ma trận A. Lưu ý rằng: mỗi vector riêng của ma trận A là một vector $(x_1, x_2, \dots, x_n) \in \mathbb{R}^n$.

Khi đó vector : $x = x_1.u_1 + x_2.u_2 + \cdots + x_n.u_n$ $(\in V)$ là vector riêng của phép biến đổi tuyến tính f.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Ví dụ: Chứng minh phép chuyển vị ma trận vuông cấp 2 là một phép biến đổi tuyến tính và tìm tất cả các trị riêng và véc tơ riêng của phép chuyển vị này.

Giải: Phép chuyển vị ma trận vuông cấp 2 là ánh xạ $f\colon M_{2\times2}(\mathbb{R}) \to M_{2\times2}(\mathbb{R})\colon f(D) = D^T, \forall D \in M_{2\times2}(\mathbb{R})$ Dễ thấy f là một ánh xạ tuyến tính. Để tìm trị riêng và véc tơ riêng của f, ta thực hiện các bước:

B1. Xét cơ sở chính tắc của $M_{2\times2}(\mathbb{R})$ là:

$$\Phi = \left\{\varphi_{11} = \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}, \varphi_{12} = \begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}, \varphi_{21} = \begin{bmatrix}0 & 0\\1 & 0\end{bmatrix}, \varphi_{22} = \begin{bmatrix}0 & 0\\0 & 1\end{bmatrix}\right\},$$

ta có: $f(\varphi_{11}) = \varphi_{11}, f(\varphi_{12}) = \varphi_{21}, f(\varphi_{21}) = \varphi_{12}, f(\varphi_{22}) = \varphi_{22}$, nên trong cơ sở này, f có ma trận:

$$A = ([f(\varphi_{11})]_\Phi, [f(\varphi_{12})]_\Phi, [f(\varphi_{21})]_\Phi, [f(\varphi_{22})]_\Phi) = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 0 & 1 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\end{bmatrix}$$

B2. Tìm trị riêng và véc tơ riêng của ma trận A.

Xét phương trình đặc trưng:

$$\det(A - \lambda.I) = 0 \Leftrightarrow \begin{vmatrix}1-\lambda & 0 & 0 & 0\\0 & -\lambda & 1 & 0\\0 & 1 & -\lambda & 0\\0 & 0 & 0 & 1-\lambda\end{vmatrix} = 0 \Leftrightarrow (\lambda - 1)^3(\lambda + 1) = 0 \Leftrightarrow \left[\begin{array}{l}\lambda = -1(đơn)\\ \lambda = 1(bội\ 3)\end{array}\right.$$

Hệ phương trình:

$$(A - \lambda I).X = \mathcal{O} \Leftrightarrow \begin{bmatrix}1-\lambda & 0 & 0 & 0\\0 & -\lambda & 1 & 0\\0 & 1 & -\lambda & 0\\0 & 0 & 0 & 1-\lambda\end{bmatrix}.\begin{bmatrix}x\\y\\z\\t\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix} \Leftrightarrow \begin{cases}(1-\lambda)x = 0\\ -\lambda y + z = 0\\ y - \lambda z = 0\\ (1-\lambda)t = 0\end{cases} \qquad (**)$$

- Ứng với $\lambda = -1$, (**) có nghiệm: $(x = 0, y = -z, z\ (tùy\ ý), t = 0)$, tức là:

$$N_{(\lambda=-1)} = \{(0, -z, z, 0) : z \in \mathbb{R}\} = Span\{(0, -1, 1, 0)\}$$

- Ứng với $\lambda = 1$, (**) có nghiệm: $(x\ (tùy\ ý), y(tùy\ ý), z = y, t\ (tùy\ ý))$, tức là:

$$N_{(\lambda=1)} = \{(x, y, y, t) : x, y, t \in \mathbb{R}\} = Span\{(1,0,0,0), (0,1,1,0), (0,0,0,1)\}$$

Từ đó suy ra: phép chuyển vị ma trận vuông cấp 2 là ánh xạ f đang xét có các trị riêng: $\lambda = -1$ (đơn) và $\lambda = 1$ (bội 3), trong đó:

Ứng với $\lambda = -1$, có các véc tơ riêng: $D = \begin{bmatrix}0 & -z\\z & 0\end{bmatrix}, \forall z \in \mathbb{R}, z \neq 0$ Với $\lambda = 1$, có các véc tơ riêng: $D = \begin{bmatrix}x & y\\y & t\end{bmatrix}, \forall x, y, t \in \mathbb{R}, x^2 + y^2 + t^2 > 0$.

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Chéo hoá ma trận:

Vấn đề chéo hóa một ma trận.
Đối với một phép biến đổi tuyến tính f trong k.gian n chiều V, ma trận của f trong cơ sở B nào đó phụ thuộc vào B. Khi cơ sở B thay đổi thì ma trận của f cũng thay đổi. Vấn đề đặt ra là:

1. Người ta mong muốn tìm được một cơ sở nào đó của V sao cho m.trận của f trong đó có dạng đơn giản là dạng chéo. Tức là có hay không một cơ sở của V sao cho m.trận của f là ma trận chéo?
2. Đối với V là không gian vector n chiều có tích vô hướng, thì có hay không một cơ sở trực giao trong V sao cho m.trận của f trong cơ sở này là m.trận chéo?

Định nghĩa: Cho ma trận vuông A. Ta nói: Ma trận A chéo hóa được, nếu tồn tại một ma trận khả nghịch P sao cho $P^{-1}.A.P$ là một ma trận đường chéo. Khi đó nói ma trận P làm chéo hóa ma trận A. Ngoài ra nếu P là một ma trận trực giao thì ta nói A chéo hóa trực giao được.
Giải bài toán chéo hóa ma trận
Định lý: $A \in R^{n\times n}$ chéo hóa trực giao được ⇔ A có n véc tơ riêng trực chuẩn. Có thể tóm tắt chứng minh định lý này như sau:
Nếu A có n véc tơ riêng trực chuẩn: $p_1, p_2, \dots, p_n$ ứng với các trị riêng $\lambda_1, \lambda_2, \dots, \lambda_n$, thì ma trận $P = \big[[p_1]\ [p_2] \dots [p_n]\big]$ là ma trận trực giao sẽ làm chéo hóa A và: $P^{-1}.A.P = D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$.
Nếu A chéo hóa trực giao được bởi ma trận trực giao P gồm các cột $p_1, p_2, \dots, p_n$ thì đương nhiên các cột này là các véc tơ trực giao khác không, hơn nữa có thể xem chúng là các véc tơ trực chuẩn (vì nếu không thì ta chuẩn hóa các véc t ơ này). Như vậy: $P^{-1}.A.P = D$ là ma trận chéo với các phần tử chéo là $\lambda_1, \lambda_2, \dots, \lambda_n$. Khi đó $\lambda_1, \lambda_2, \dots, \lambda_n$ là các trị riêng của A và $p_1, p_2, \dots, p_n$ là các véc tơ riêng tương ứng.
Định lý:
A chéo hóa trực giao được ⇔ A là ma trận đối xứng.
Nếu A là m.trận đối xứng thì:
các véc tơ riêng thuộc các k.gian riêng khác nhau sẽ trực giao nhau theo tích vô hướng Euclid trong $\mathbb{R}^n$;
số bội hình học của mỗi trị riêng bằng số bội đại số của nó, tức là: nếu λ là một nghiệm bội k của phương trình đặc trưng thì k.gian riêng N_λ sẽ có số chiều là k.
Nhận xét:
A là chéo hóa được ⇔ A đồng dạng với một m.trận đường chéo.
Nếu P làm chéo hóa A:

$$P^{-1}.A.P = D, \text{thì: } A^k = P.D^k.P^{-1},\ \forall k \in \mathbb{N}^* \text{ và nếu A khả nghịch thì: } A^{-1} = P.D^{-1}.P^{-1}$$

KHÔNG GIAN VECTOR

Quy trình chéo hóa trực giao một ma trận A đối xứng như sau:

1. Tìm các trị riêng $\lambda_1, \lambda_2, \dots, \lambda_n$ (có thể trùng nhau do nghiệm bội) và các k.gian riêng tương ứng. Tìm các véc tơ cơ sở cho mỗi k.gian riêng.
2. Áp dụng quá trình trực giao hóa Gram-Smidt cho mỗi cơ sở của các k.gian riêng để thu được cơ sở trực chuẩn cho mỗi k.gian này. Từ đó nhận được n véc tơ riêng trực chuẩn: $p_1, p_2, \dots, p_n$.
3. Lập ma trận P gồm n cột là n véc tơ $p_1, p_2, \dots, p_n$. Ma trận P là trực giao, sẽ làm chéo hóa A: $P^{-1}.A.P = D(\text{các phần tử chéo là } \lambda_1, \lambda_2, \dots, \lambda_n)$.

VD 5. Tìm m.trận trực giao P làm chéo hóa ma trận: $B = \begin{bmatrix} 3 & -2 & 0 \\ -2 & 3 & 0 \\ 0 & 0 & 5 \end{bmatrix}$

Giải: Trong ví dụ 3, ta đã tìm được các trị riêng $\lambda_1 = 1$(đơn) và $\lambda_2 = 5$(bội 2) và N_{λ_1} có cơ sở là một véc tơ $v_1 = (1,1,0)$, N_{λ_2} có cơ sở là 2 véc tơ $v_2 = (1,-1,0), v_3 = (0,0,1)$. Áp dụng quá trình trực giao hóa Gram-Smidt:

- Đối với cơ sở $\{v_1\}$, nhận được cơ sở trực chuẩn $\left\{ u_1 = \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, 0\right)\right\}$
- Đối với cơ sở $\{v_2, v_3\}$, nhận được cơ sở trực chuẩn $\{u_2 = \left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, 0\right), u_3 = (0,0,1)\}$. Ma trận trực giao làm chéo hóa B là: $P = [[\mathrm{u}_1]\ [\mathrm{u}_2]\ [\mathrm{u}_3]]$

KHÔNG GIAN VECTOR

Ánh xạ tuyến tính:

Ma trận xác định dương và bán xác định dương.

Định nghĩa: Cho ma trận đối xứng Cho $A, B \in \mathbb{R}^{n\times n}$, gọi:

1. A là ma trận ***xác định dương*** (*positive definite*), kí hiệu: $A \succ 0$ nếu:
$$(x).A.[x] > 0, \forall x \in \mathbb{R}^n, x \neq \mathcal{O}$$
2. A là ma trận ***bán xác định dương*** (*positive semidefinite*), kí hiệu: $A \succcurlyeq 0$ nếu:
$$(x).A.[x] \geq 0, \forall x \in \mathbb{R}^n$$
3. A là ma trận ***xác định âm*** (*negative definite*), kí hiệu: $A \prec 0$, nếu:
$$(x).A.[x] < 0, \forall x \in \mathbb{R}^n, x \neq \mathcal{O}$$
4. A là ma trận ***bán xác định âm*** (*negative semidefinite*), kí hiệu: $A \preccurlyeq 0$, nếu:
$$(x).A.[x] \leq 0, \forall x \in \mathbb{R}^n$$
5. Kí hiệu $A \succ B(\succcurlyeq, \prec, \preccurlyeq) \Leftrightarrow A - B \succ 0(A \succcurlyeq B, A \prec B, A \preccurlyeq B)$
6. A là ma trận không xác định dấu, nếu $\exists u, v \in \mathbb{R}^n: (u).A.[u] > 0, (v).A.[v] < 0$

Chú ý:

- $A \succ 0$ (hoặc $A \prec 0$) $\Rightarrow A \succcurlyeq 0$ **(hoặc** $A \preccurlyeq 0$**)**
- Một ma trận Hermitian $A \in \mathbb{C}^{n\times n}$được gọi là xác định dương, nếu: $(x)^H.A.[x] > 0, \forall x \in \mathbb{C}^n, x \neq \mathcal{O}$ (Tương tự cho các khái niệm Ma trận Hermitian bán xác định dương, xác định âm, bán xác định âm).
- Thực tế người ta thường quan tâm đến tính bán xác định dương
- Các đặc tính nói trên của ma trận được dùng trong các bài toán tối ưu, cực trị.

Ví dụ:

$$\text{Xét các ma trận: } A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, B = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Dễ thấy:

A là xác định dương, B là bán xác định dương, $-A$ là xác định âm, $-B$ là bán xác định âm, C không phải bán xác định dương, không phải bán xác định âm.

Ánh xạ tuyến tính:

Các tính chất:

1. Ma trận xác định dương có tất cả các trị riêng đều là các số thực dương
2. Nếu A là xác định dương thì A khả nghịch và $\det(A) > 0$ và:$(x).A.[x] = 0 \Leftrightarrow A.[x] = \mathcal{O}$
3. (*Tiêu chuẩn Sylvester*). Với ma trận đối xứng $A = (a_{ij})_{n\times n}$
   - A là xác định dương (bán xác định dương) $\Leftrightarrow$ mọi m.trận vuông con góc trái $A_k = (a_{ij})_{k\times k}$ $(k = 1, 2,..., n)$ của A đều có $\det(A_k) > 0$ $(\det(A_k) \geq 0)$.
   - A là xác định âm $\Leftrightarrow$ mọi ma trận vuông con góc trái $A_k = (a_{ij})_{k\times k}$ $(k = 1, 2,..., n)$ của A có $\det(A_k) > 0$ $(\det(A_k) \geq 0)$. nếu k chẵn và $\det(A_k) < 0$ $(\det(A_k) \leq 0)$. nếu k lẻ.
4. $\forall B \in R^{m\times n}$ thì $A = B^H.B$ là m.trận xác định dương
5. (Khai triển Cholesky): Mọi m.trận Hermitian bán xác định dương đều biểu diễn được duy nhất dưới dạng: $A = L.L^H$, trong đó L là một m.trận tam giác dưới có các phần tử trên đường chéo đều là các số thực dương.
6. Nếu A là m.trận bán xác định dương thì: $(x).A.[x] = 0 \Leftrightarrow A.[x] = \mathcal{O}$

VD7. Cho các ma trận

$$A = \begin{bmatrix} 5 & 2 & -1 \\ 2 & 1 & -1 \\ -1 & -1 & 3 \end{bmatrix}, B = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, C = \begin{bmatrix} -5 & -2 & 1 \\ -2 & -1 & 1 \\ 1 & 1 & -3 \end{bmatrix}, D = \begin{bmatrix} 1 & 1-i \\ i & -1 \\ 0 & 1 \end{bmatrix}$$

Ánh xạ tuyến tính:

- Xét A, có: $\det(A_1) = 5 > 0, \det(A_2) = \begin{vmatrix} 5 & 2 \\ 2 & 1 \end{vmatrix} = 1 > 0$, $\det(A_3) = \det(A) = 1 > 0$. Vậy A là m.trận xác định dương
- Xét B: $\det(B_1) = 1 > 0, \det(B_2) = \det(B) = 0$. Vậy B bán xác định dương
- Xét C: $\det(C_1) = -5 < 0, \det(C_2) = \begin{vmatrix} -5 & -2 \\ -2 & -1 \end{vmatrix} = 1 > 0$, $\det(C_3) = \det(C) = -1 < 0$. Vậy C là m.trận xác định âm
- Xét m.trận $U = D^H.D = \begin{bmatrix} 1 & -i & 0 \\ 1+i & -1 & 1 \end{bmatrix}.\begin{bmatrix} 1 & 1-i \\ i & -1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$

Theo T/c 4 hoặc theo tiêu chuẩn Sylvester, ta có U là m.trận xác định dương.

Vết của ma trận vuông:

$$\text{Cho } A = \left(a_{ij}\right)_{n\times n} \in R^{n\times n}$$

Định nghĩa:
Vết của ma trận A, kí hiệu trace(A), là tổng các phần tử trên đường chéo:

$$\mathrm{trace}(A) = a_{11} + a_{22} + \cdots + a_{nn}$$

VD8. Trong VD7, có trace(A) = 5 + 1 + 3 = 9

Các tính chất.

1. $\mathrm{trace}\left(A^T\right) = \mathrm{trace}(A) = \lambda_1 + \lambda_2 + \cdots + \lambda_n$ (Tổng các trị riêng)

$$\mathrm{trace}(A.B) = \mathrm{trace}(B.A);\ \mathrm{trace}(A+B) = \mathrm{trace}(A) + \mathrm{trace}(B)$$

2. $\forall X$ (khả nghịch) $\in R^{n\times n}$ **:** $\mathrm{trace}(X.A.X^{-1}) = \mathrm{trace}(X^{-1}.A.X) = \mathrm{trace}(A)$

3. $\forall X \in R^{m\times n}$: $\mathrm{trace}\left(X.X^T\right) = \mathrm{trace}\left(X^T.X\right) = \|X\|_F^2$

Vết của m.trận vuông có nhiều ứng dụng trong tối ưu.