THỰC HÀNH ĐẠI SỐ TUYẾN TÍNH

TÀI LIỆU PHỤC VỤ SINH VIÊN NGÀNH KHOA HỌC DỮ LIỆU

Nhóm biên soạn: TS. Hoàng Lê Minh – Huỳnh Thái Học – Khưu Minh Cảnh

TP.HCM - Năm 2019

ÔN THI CUỐI KÌ THỰC HÀNH ĐẠI SỐ TUYẾN TÍNH

Mục tiêu:

Bài ôn tập thực hành đại số tuyến tính:

- *Các nội dung chính sinh viên được học.*
- *Giải một số bài tập trong chương trình đã học.*

Nội dung chính:

Ôn tập gồm 10 bài. Yêu cầu thực hiện mỗi bài tập gồm: mô tả lý thuyết (chứng minh nếu có), đoạn code viết bằng ngôn ngữ Python để xử lý và các mô tả cần thiết khác (như mở rộng, những quan điểm/ứng dụng sinh viên đề xuất,…).

1. Bài tập 1

Đề bài: Ba ông Cảnh, Tùng, Tùng dự định sẽ mua 4 món Gạo, Bún, Bánh ngọt và Bánh mì với số lượng khác nhau. Được biết, 4 loại hàng hóa này được bày bán tại 2 siêu thị (S1 và S2). Giả định, nhu cầu của mỗi người và giá mỗi sản phẩm được cho ở mỗi cửa hàng như dưới đây:

Bảng nhu cầu (P)	***Gạo***	***Bún***	***Bánh ngọt***	***Bánh mì***
Ông Cảnh	6	5	3	1
Ông Tùng	3	6	2	2
Ông Hiếu	3	4	3	1

Giá tại mỗi siêu thị:

Bảng giá (Q)	S1	S2
Gạo	1.50	1.00
Bún	2.00	2.50
Bánh ngọt	5.00	4.50
Bánh mì	16.00	17.00

Hãy tìm các siêu thị thích hợp cho mỗi ông để mỗi người mua được rẻ nhất. Biết rằng do chi phí lưu thông, mỗi ông chỉ đi được 1 siêu thị.

Hướng dẫn giải:

Nếu chọn siêu thị 1 và P2, người P1 sẽ phải trả số tiền tương ứng là:

$$6*\ 1.50 + 5*\ 2.00 + 3*\ 5.00 + 1*\ 16.00 = 50 \text{ đồng}$$

Và:

$$6*\ 1.00 + 5*\ 2.50 + 3*\ 4.50 + 1*\ 17.00 = 49 \text{ đồng}$$

Như vậy, đối với ông Cảnh, giá ở siêu thị S2 rẻ hơn giá ở siêu thị S1. Chúng ta có thể thực hiện phép nhân ma trận:

$$Ma\ trận\ n\square u\ cầu: P = \begin{pmatrix} 6 & 5 & 3 & 1 \\ 3 & 6 & 2 & 2 \\ 3 & 4 & 3 & 1 \end{pmatrix}; Ma\ trận\ giá: Q = \begin{pmatrix} 1.50 & 1.00 \\ 2.00 & 2.50 \\ 5.00 & 4.50 \\ 16.00 & 17.00 \end{pmatrix}$$

Và kết quả là:

$$P.Q = \begin{pmatrix} 50 & 49 \\ 58.50 & 61 \\ 43.50 & 43.50 \end{pmatrix}$$

Từ đó, chúng ta thấy ông Cảnh nên mua ở siêu thị S2, ông Tùng nên mua ở siêu thị S1 và ông Hiếu có thể mua ở 1 trong hai siêu thị vì giá bằng nhau.

Lệnh nhân ma trận P và Q và lệnh tìm phần tử nhỏ nhất trong kết quả:

```
>>> import numpy as np

>>> P = np.array([[6,5,3,1],[3,6,2,2],[3,4,3,1]])

>>> Q = np.array([[1.5, 1.0],[2.0, 2.5],[5.0, 4.5], [16.0, 17.0]])

>>> PQ = np.dot(P,Q)

>>> PQ.T

array([[50. , 58.5, 50. ],
```

[49. , 61. , 49.]])

>>> np.**fmin**((PQ.T)[0],(PQ.T)[1]) # Hàm numpy.fmin lấy giá trị cực tiểu mỗi cột của ma trận

array([49. , 58.5, 49.])

***Trao đổi và bình luận**: Đây chỉ đơn thuần là bài toán nhân ma trận. Gồm ma trận nhu cầu và ma trận giá bán. Sau đó tìm giá trị lớn nhất tương ứng với mỗi cột. Bài toán có thể mở rộng ở chiều ngược lại là các siêu thị đánh giá tiềm năng khách hàng thường mua và đưa ra các bảng giá hợp lý để thu hút khách hàng hoặc đạt được lợi nhuận cao. Nghĩa là xây dựng chiến lược về giá hoặc phân khúc đúng khách hàng để các siêu thị/nhà hàng/cửa hàng bán quần áo/… bán được nhiều hàng hóa và thu được nhiều lợi nhuận. Sinh viên sẽ tiếp tục được nghiên cứu các mô hình bài toán này trong các môn tối ưu hóa và lý thuyết thống kê.*

2. Bài tập 2

Đề bài: Viết chương trình và các câu lệnh để tính số Pi ở dạng liên phân số, cho dãy lưu trữ số Pi như sau: 3, 7, 15, 1, 292, 1, 1, 1, 1, 2.

Hướng dẫn giải:

- Biểu diễn liên phân số bằng ma trận:

Liên phân số, tiếng Anh gọi là Continued Fraction, là một dạng biểu diễn các số thực (hữu tỉ và vô tỉ) dưới dạng phân số nhiều tầng. Ví dụ:

$$\frac{9}{7} = 1 + \frac{1}{3 + \frac{1}{2}}$$

Người ta chứng minh được rằng có một cách xác định p_n và q_n trong phân số $\frac{p_n}{q_n}$bằng cách sau:

$$\begin{pmatrix} c_0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} c_1 & 1 \\ 1 & 0 \end{pmatrix}\dots\begin{pmatrix} c_n & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} p_n & p_{n-1} \\ q_n & q_{n-1} \end{pmatrix}, n = 0,1,2,\dots$$

Với số π, chúng ta có thể tính toán theo dãy số mô tả liên phân số.

- Đoạn lệnh Python xử lý:

>>> **import** numpy **as** np

```
>>> c = [3,7,15,1,292,1,1,1,2]
>>> M = np.mat([[1,2],[3,4]])  # khởi tạo ma trận 2x2, là ma trận tạm thời
>>> for i in range(len(c)):
        ci = np.mat([[1,1],[1,0]])
        ci[0, 0] = c[i]
        if (i==0):
            M = ci
        else:
            M = M.dot(ci)
>>> print(M)
>>> print (M[0,0]/M[1,0])
…… # sinh viên phải viết ra số Pi!
```

3. Bài tập 3

Đề bài: Số Fibonacci

Hãy chọn và trình bày một phương pháp tính toán số Fibonacci. Sau đó minh họa bằng chương trình viết trên thư viện numpy (Python). Hãy cho biết (viết vào giấy bài làm) 100 số Fibonacci đầu tiên. Giả định số Fibonacci được cho gồm các số F0 = 0; F1 = 1; F2 = 1,…

Hướng dẫn giải:

Gọi F là dãy Fibonacci, chúng ta có:

$$F_n = F_{n-1} + F_{n-2}$$

Để xây dựng một mô hình tuyến tính, ta cần bổ sung thêm một phương trình xác định F_{n-1}:

$$F_{n-1} = F_{n-1} + (0).F_{n-2}$$

Từ hai phương trình trên, chúng ta có thể xây dựng được hệ phương trình tuyến tính:

$$\begin{cases} F_n = F_{n-1} + F_{n-2} \\ F_{n-1} = F_{n-1} + (0).F_{n-2} \end{cases}$$

Thể hiện dưới dạng ma trận là:

$$\begin{bmatrix} F_n \\ F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_{n-1} \\ F_{n-2} \end{bmatrix}$$

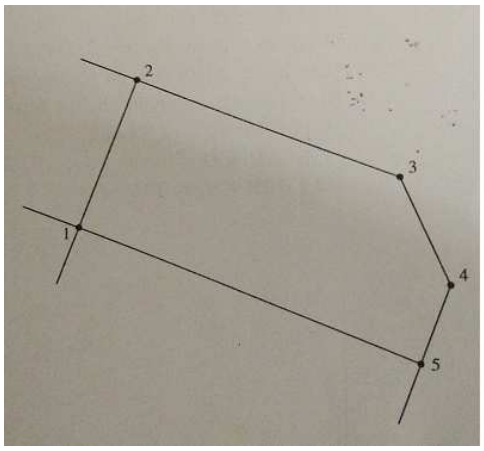
Với giá trị vector ban đầu là: $[F_1 \quad F_0]^T = [1 \quad 0]^T$

Đoạn lệnh tính toán 100 số Fibonacci đầu tiên:

```
>>> import numpy as np
>>> A = np.array( [ [1,1], [1,0] ] )
>>> b = np.array([1, 0])
>>> n = 100
>>> for i in range(n):
        b = A.dot(b)
        print(b)
```

4. Bài tập 4

Đề bài: Tìm diện tích của mảnh đất sau đây



BẢNG KÊ TỌA ĐỘ

Điểm	X(mét)	Y(mét)
1	1181128.25	603263.70
2	1181136.18	603266.68
3	1181130.97	603279.69
4	1181125.77	603281.91
5	1181122.06	603280.42

Hướng dẫn giải:

- **Phương pháp tính diện tích của tập điểm:**

Tính toán diện tích đa giác có n điểm. *Diện tích đa giác bằng tổng các diện tích của các tam giác ghép thành nên đa giác đó.*

$$Diện\ tích_P = \frac{1}{2}\left(\begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} + \begin{vmatrix} x_2 & y_2 \\ x_3 & y_3 \end{vmatrix} .. + \begin{vmatrix} x_{n-1} & y_{n-1} \\ x_n & y_n \end{vmatrix} + \begin{vmatrix} \boldsymbol{x_n} & \boldsymbol{y_n} \\ \boldsymbol{x_1} & \boldsymbol{y_1} \end{vmatrix}\right)$$

Tương ứng bên trên, ta có các điểm 1 là (x_1, y_1),…

- **Thực hiện tính toán diện tích trên:**

Xây dựng dữ liệu các điểm:

>>> X = [28.25, 36.18, 30.97, 25.77, 22.06, 28.25]

>>> Y = [63.70, 66.68, 79.69, 81.91, 80.42, 63.70]

Lưu ý: các điểm đều được tịnh tiến 1 vector k = (-1181100, -603200) nên diện tích không thay đổi nhưng giá trị tính được làm gọi lại. Ví dụ: từ giá trị 1181128.25 trở thành 28.25.

Tính toán diện tích theo công thức trên:

>>> S = 0

>>> **for** i in **range**(len(X)-1):

    S = S + (X[i]*Y[i+1]-X[i+1]*Y[i])

>>> print (S/2.0)

…….. # sinh viên điền kết quả

Kết luận: diện tích khu đất gần bằng …….. m2.

5. Bài tập 5

Đề bài: Bằng phương pháp bình phương cực tiểu giải bài toán sau:

Một người đi xạ trị bệnh ung thư bằng việc bơm thuốc vào người. Khi vừa xạ trị xong, lượng thuốc tồn động trong người ở mức 10 đơn vị. Sau đó 1 giờ sau lượng thuốc còn lại là 8; lần lượt 2, 3, 4 giờ sau lượng thuốc còn lại là 7, 5 và 2. Hãy viết các lệnh bằng Python để tìm phương trình tuyến tính bằng phương pháp bình phương cực tiểu với tập các cặp số (x, y) để tìm công thức giảm lượng thuốc đối với bệnh nhân theo thời gian:

$$(x, y) = \{(0, 10), (1, 8), (2, 7), (3, 5), (4, 2)\}$$

Hướng dẫn giải:

- **Lý thuyết:**

Thực hiện việc giải hệ trên:

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix}, Y = \begin{bmatrix} 10 \\ 8 \\ 7 \\ 5 \\ 2 \end{bmatrix}$$

Ta có:

$$A = (X^T X)^{-1} X^T Y = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

- **Đoạn mã xử lý:**

```
def bpcuctieu(a):
    b = []
    o1 = []
    for i in range(len(a)):
            o1.append(1.0)      # tạo cột 1
            b.append((i+1)*1.0)   # tạo cột [1 2 3 4 5…]
    x = np.array([o1, b])  # xay dung ma tran de tinh (xây dựng ma trận X)
    u = np.dot(x, x.T)  # luu y truong hop nay nguoc lai! (về vấn đề chuyển vị ma trận!)
    y = np.array(a)
    from numpy import linalg
    u_1 =np.linalg.inv(u)
    v = np.dot(u_1, x) # khong can x.T
    A = np.dot(v, y)
    return A


# Thực thi đoạn mã:
>>> dulieu = [10.0, 8.0, 7.0, 5.0, 2.0]
>>> bpcuctieu(dulieu)
```

…… # sinh viên ghi kết quả tính toán và diễn giải phương trình.

Và tính toán giá trị.

Hướng dẫn ghi kết quả:

Ví dụ: nếu kết quả ra là array([-0.2, 1.2]) thì sinh viên phải viết vào **array([-0.2, 1.2])** và ghi phương trình tìm được là: **-0.2 + 1.2x** và tính toán giá trị tại thời điểm tiếp theo: x = 6 → -0.2+1.2x = -0.2+1.2x6 = 7.0.

6. Bài tập 6

Đề bài: Tính tỉ lệ thất nghiệp

Tỉ lệ thất nghiệp ở một thành phố thường thay đổi theo thời gian. Xét mô hình đơn giản sau, được gọi là mô hình Markov. Mô hình mô tả xác suất của sự chuyển dịch từ có việc làm sang thất nghiệp. Chúng ta có các giả định với thời gian được xét theo đơn vị là tuần:

- Với 1 người thất nghiệp ở tuần nào đó, người đó sẽ có xác suất p có việc làm ở tuần tiếp theo. Và như vậy, nghĩa là người đó sẽ có xác suất $1 - p$ vẫn thất nghiệp ở tuần tiếp theo.
- Tương tự, với một người đang có việc làm, gọi q là xác suất người đó vẫn còn đi làm. Điều này có nghĩa là xác suất người đó thất nghiệp là $1 - q$.

Với $p = 0.136; q = 0.998$. Tại thời điểm t_0, $t = 0$,

Giả định chúng ta có: $x_0 = 0.95$ và $y_0 = 0.05$. Tìm tỉ lệ thất nghiệp trong tuần thứ 100 giả định các hệ số p và q không thay đổi.

Hướng dẫn giải:

- **Diễn giải lý thuyết:**

Ta có, nếu gọi:

- x_t là người đang có việc làm ở tuần t.
- y_t là người đang thất nghiệp ở tuần t.

Khi đó, chúng ta có hệ phương trình:

$$\begin{cases} x_{t+1} = qx_t + py_t \\ y_{t+1} = (1-q)x_t + (1-p)y_t \end{cases}$$

Thể hiện ở dạng ma trận: $v_{t+1} = Av_t$, với:

$$A = \begin{pmatrix} q & p \\ 1-q & 1-p \end{pmatrix}, v_t = \begin{pmatrix} x_t \\ y_t \end{pmatrix}$$

Gọi A là ma trận chuyển trạng thái và v_t là vector trạng thái của hệ thống. Trạng thái sau thứ t tuần được mô tả lần lượt như sau:

- Tuần 1: $v_1 = Av_0$
- Tuần 2: $v_2 = Av_1 = A(Av_0) = A^2v_0$
- Tuần 3: $v_3 = Av_2 = A(A^2v_0) = A^3v_0$
- → Tuần thứ k: $v_t = A^tv_0$

Giả định chúng ta có số liệu sau: $p = 0.136$; $q = 0.998$. Tại thời điểm t_0, nghĩa là $t = 0$, giả định chúng ta có: $x_0 = 0.95$ và $y_0 = 0.05$. Trạng thái sau 100 tuần về tình trạng thất nghiệp/có việc làm của thành phố sẽ là:

$$\begin{pmatrix} x_{100} \\ y_{100} \end{pmatrix} = \begin{pmatrix} 0.998 & 0.136 \\ 0.002 & 0.864 \end{pmatrix}^{100} \cdot \begin{pmatrix} 0.95 \\ 0.05 \end{pmatrix} = \cdots$$

- **Lệnh xử lý bằng Python:**

```
>>> A = np.array([[0.998, 0.136],[0.002, 0.864]])

>>> v = np.array([0.95, 0.05])

>>> A2 = np.dot(A,A)

>>> for i in range(99):

        A2 = np.dot(A2, A)

>>> np.dot(A2, v)

array([0.98550724, 0.01449276])
```

Vậy tỉ lệ thất nghiệp ở tuần thứ 100 là: 1,449%

7. Bài tập 7

Đề bài: Tìm ánh xạ tuyến tính f khi biết f: $\square^2 \to \square^3$ có cơ sở của $\square^2$ là

$$B = \{u_1 = (1,2); u_2 = (3,5)\}$$

Và

$$f(u_1) = (1;1;2), f(u_2) = (4;2;1)$$

Hướng dẫn giải:

Giải hệ với $u = (x, y)$ bất kỳ để tìm ra α và β. Hệ:

$$\begin{bmatrix} 1 & 3 & x \\ 2 & 5 & y \end{bmatrix}$$

Giải ra ta được:

$$\begin{cases} \alpha = -5x + 3y \\ \beta = 2x - y \end{cases}$$

Từ công thức $u = \alpha u_1 + \beta u_2$, thay thế vào công thức $f(u) = \alpha f(u_1) + \beta f(u_2)$ để tìm được ánh xạ tuyến tính. [đáp án: $f(x, y) = (3x - y, -x + y, -8x + 5y)$.

Thể hiện bằng các lệnh Python, sử dụng gói tính toán hình thức **sympy**:

- **Viết được các câu lệnh bằng Python để giải được alpha, beta.**

```
>>> import sympy as sp
>>> a, b = sp.symbols('a b')
>>> x, y = sp.symbols('x y')
>>> sp.solve([a+3*b-x, 2*a+5*b-y],[a,b])
…….. #  điền kết quả (nghiệm)
```

$$\begin{cases} \alpha = -5x + 3y \\ \beta = 2x - y \end{cases}$$

- **Viết được các câu lệnh bằng Python để tìm được ánh xạ tuyến tính f(x,y).**

Từ công thức $u = \alpha u_1 + \beta u_2$, ta thay thế vào công thức $f(u) = \alpha f(u_1) + \beta f(u_2)$ để tìm được ánh xạ tuyến tính. Thực hiện code như sau:

```
>>> fu1 = np.array([1,1,2])
>>> fu2 = np.array([4,2,1])
>>> fu = a*fu1 + b*fu2
>>> print (fu)
……..…….. # (lưu ý f(u) chưa biểu diễn dạng xy)
```

>>> fu = **a.subs**(a, -5*x + 3*y)*fu1 + **b.subs**(b, 2*x - y)*fu2 # thay giá trị tìm được ở trên vào

>>> print (fu)

……..…….. # điền kết quả cuối cùng vào.

8. Bài tập 8

Đề bài: Chứng minh tập W là tập các ma trận cấp 2 đối xứng (nghĩa là có tính chất $A^T = A$, với A^T là ma trận chuyển vị của A) là không gian con của không gian các ma trận $M_{2\times 2}$ đối với phép cộng ma trận và phép nhân một số thực.

Hướng dẫn giải:

Điều 1: Tập W rõ ràng không phải là tập rỗng.

Điều 2: Kiểm tra 2 tính chất theo định nghĩa:

$A_1 \in W, A_2 \in W \Longrightarrow (A_1 + A_2)^T = A_1^T + A_2^T = A_1 + A_2$, được tính chất: $A_1 + A_2 \in W$.

$c \in \mathbb{R}, A \in W \Longrightarrow (cA)^T = cA^T = cA$, ta được thêm tính chất: $cA \in W$

Thể hiện các minh chứng trên bằng phần mềm, cụ thể gói phần mềm **sympy** hỗ trợ tính toán hình thức (**chỉ sử dụng các biến hình thức, không cần thay giá giá trị cụ thể vào biến**). Lưu ý: một ma trận bậc 2 có tính chất $A^T = A$, nghĩa là ma trận đối xứng như sau:

$$A = \begin{pmatrix} x & y \\ y & x \end{pmatrix}$$

Thực hiện các lệnh trong gói thư viện sympy:

>>> import numpy as np

>>> import sympy as sp

>>> x, y = sp.symbols('x y') # khai báo 2 biến x và y

>>> A = **sp.Matrix**([[x, y],[y, x]]) # lưu ý: kiểu ma trận của sympy là ‘Matrix’ (viết hoa)

>>> x1, y1 = sp.symbols('x1 y1')

>>> A1 = **sp.Matrix**([[x1, y1],[y1, x1]])

>>> x2, y2 = sp.symbols('x2 y2')

>>> A2 = sp.Matrix([[x2, y2],[y2, x2]])

>>> A1.T

……# ← sinh viên **phải** điền kết quả

……

>>> (A1+A2).**T**

……# ← sinh viên **phải** điền kết quả

……

>>> ((A1+A2).T).**equals**(A1+A2)

……# ← sinh viên **phải** điền kết quả

>>> c = **sp.symbols**('c')

>>> print (c*A)

……# ← sinh viên **phải** điền kết quả

>>> ((c*A).T).**equals**(c*A)

……# ← sinh viên **phải** điền kết quả

Đó là điều cần chứng minh (sử dụng công cụ máy tính).

9. Bài tập 9

Đề bài: Chứng minh rằng tập $S = \{v_1, v_2\} = \{(1,1), (1,-1)\}$ là một cơ sở của $\mathbb{R}^2$.

Hướng dẫn giải:

Cơ sở nghĩa là phải thỏa 2 tính chất: vừa là tập sinh và vừa độc lập tuyến tính. Nghĩa là phải kiểm 2 tính chất:

- Kiểm tính chất tập sinh: $\forall u = (u_1, u_2) \in \mathbb{R}^2, c_1v_1 + c_2v_2 = u \Longrightarrow \begin{cases} c_1 + c_2 = u_1 \\ c_1 - c_2 = u_2 \end{cases}$ luôn có nghiệm (định thức khác 0), mang ý nghĩa luôn có nghiệm.
- Kiểm tính chất độc lập tuyến tính: kiểm hệ $c_1v_1 + c_2v_2 = 0 \; hay \begin{cases} c_1 + c_2 = 0 \\ c_1 - c_2 = 0 \end{cases}$ có định thức khác 0 (nghĩa là có nghiệm duy nhất).

Từ đó, chúng ta có thể sử dụng sympy để giải hệ trên:

>>> **import numpy as np**

>>> import sympy as sp

- Kiểm tập sinh:

>>> c1, c2 = sp.symbols('c1 c2')

>>> u1, u2 = sp.symbols('u1 u2')

>>> sp.**solve**([c1+c2-u1,c1-c2-u2])

……# ← sinh viên **phải** điền kết quả

- Kiểm độc lập tuyến tính:

Giải cụ thể với u=0:

>>> u1 = 0

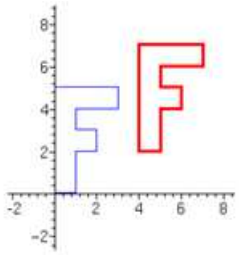
>>> u2 = 0

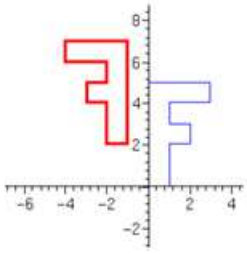
>>> sp.**solve**([c1+c2-u1,c1-c2-u2])

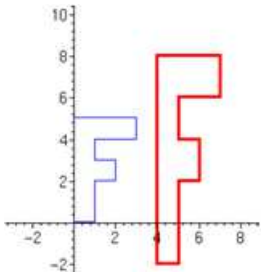
……# ← sinh viên **phải** điền kết quả

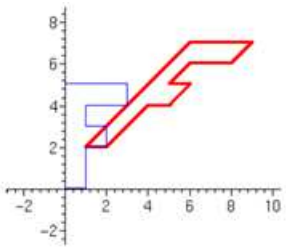
10. Bài tập 10

Đề bài: Hãy tìm vector *V* và ma trận *A* theo 4 biến đổi sau:

a. 

c. 

b. 

d. 

Hãy viết các đoạn mã xử lý bằng Python đối với 2 phép biến đổi ở câu a và b.

Hướng dẫn giải:

Chữ F gốc (ban đầu) trong các hình là tập các vector được hình thành từ tập điểm có thứ tự như sau: $P = \{(0,0),(0,5),(3,5),(3,4),(1,4),(1,3),(2,3),(2,2),(1,2),(1,0)\}$. Theo đó, các biến đổi trên lần lượt là:

a. Biến đổi tịnh tiến. Cụ thể: từ (x, y) thành $(x + a, y + b)$, nghĩa là ở dạng ma trận:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} x + a \\ y + b \end{pmatrix}$$

Với giá trị $a = 4$ và $b = 2$ (dời tọa độ $(0, 0)$ sang tọa độ $(4, 2)$.

Sinh viên thực hiện các lệnh sau:

```
>>> import numpy as np
>>> P = np.array([[0,0,3,3,1,1,2,2,1,1],[0,5,5,4,4,3,3,2,2,0]])
>>> vecdelta = np.array([4,2])
>>> P_caua = (P.T + vecdelta).T    # sinh viên có thể tách làm từng lệnh để hiểu
>>> print (P_caua)
……..
…….. # sinh viên kiểm tra so với hình
```

b. Biến đổi tịnh tiến và co giãn. Cụ thể từ (x, y) thành $(kx + a, ly + b), k, l > 0$. Nghĩa là ở dạng ma trận:

$$\begin{pmatrix} k & 0 \\ 0 & l \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} kx + a \\ ly + b \end{pmatrix}$$

Hệ số k, l sẽ được biện luận thêm như sau:

+ Co (contraction): $0 < k, l < 1$

+ Giãn (expansion): $k, l > 1$

Với giá trị $a = 4$ và $b = -2$ (dời tọa độ $(0, 0)$ sang tọa độ $(4, -2)$.

Sau đó thực hiện phép co giãn trục x vẫn giữ nguyên nhưng trục y gấp đôi, nghĩa là:

$k = 1.0$ và $l = 2.0$

```
>>> import numpy as np
>>> P = np.array([[0,0,3,3,1,1,2,2,1,1],[0,5,5,4,4,3,3,2,2,0]])
>>> vecdelta = np.array([4,-2])
>>> matran_biendoi = np.array([[1.0, 0.0],
                               [0.0, 2.0]])
>>> P_caub = (P.T @ matran_biendoi + vecdelta).T
 >>> print (P_caub)
……..
…….. # sinh viên kiểm tra so với hình
```

c. Biến đổi tịnh tiến và đối xứng theo trục y. Cụ thể từ (x, y) thành $(-x + a, y + b)$. Nghĩa là dạng ma trận tương ứng:

$$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} -x + a \\ y + b \end{pmatrix}$$

d. Biến đổi tịnh tiến và shearing. Cụ thể từ (x, y) thành $(x + py + a, qx + y + b)$, một trong hai giá trị p và q bằng 0. Nghĩa là ở dạng ma trận là:

$$\begin{pmatrix} 1 & p \\ q & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} x + py + a \\ qx + y + b \end{pmatrix}$$