Mục lục

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

PHƯƠNG PHÁP DANILEVSKI TÌM GIÁ TRỊ RIÊNG VÀ VECTOR RIÊNG

BÁO CÁO
GIẢI TÍCH SỐ

Giảng viên hướng dẫn: TS. Hà Thị Ngọc Yến
Sinh viên thực hiện: Phạm Tiến Duật
Lớp: KSTN Toán Tin K62

HÀ NỘI, 1/2022

LỜI MỞ ĐẦU

Trong chương trình đại số tuyến tính ta đã làm quen với các khái niêm trị riêng và vector riêng. Nhưng việc tìm các trị riêng và vector riêng đó chỉ là trên những ma trận có kích thước nhỏ. Trong thực tế các bài toán đặt ra luôn rất phức tạp buộc ta phải làm việc với những ma trận có kích thức rất lớn, khi đó các phương pháp thông thường như việc giải thủ công đa thức đặc trưng là việc rất khó khăn thậm chí là cả đối với máy tính. Chính vì vậy vấn đề đặt ra là cần phải xây dựng một phương pháp giúp công việc này trở lên dễ dàng hơn, khối lượng tính toán ít hơn nhưng vẫn đảm bảo độ chính xác cần thiết.

Phương pháp Danilevski là một trong những phương pháp để giải bài toán tìm trị riêng vector riêng với khối lượng tính toán ít, thuật toán dễ hiểu đơn giản nhưng khá hiệu quả. Tư tưởng của phương pháp là sử dụng những phép biến đổi đưa ma trận A ban đầu về dạng có cấu trúc đơn giản hơn, từ đó dễ dàng tìm được đa thức đặc trưng và vector riêng.

Bài báo cáo không thể hoàn thiện nếu thiếu đi những góp ý và nhẫn xét quý báu của cô và các bạn, đặc biệt là những nhận xét và sự hướng dẫn tận tình của cô Hà Thị Ngọc Yến, người đã giúp em có được cái nhìn hoàn thiện và hiểu sâu hơn về phương pháp. Cuối cùng em xin gửi lời cảm ơn và biết ơn sâu sắc đến cô đã giúp đỡ em rất nhiều trong thời gian vừa qua!

Chương 1

Kiến thức chuẩn bị

1.1 Trị riêng và vector riêng của ma trận

Cho $A = (a_{ij})$ là ma trận vuông cỡ $n \times n$. Nếu $Au = \lambda u$, khi đó λ và u tương ứng được gọi là giá trị riêng và vector riêng của ma trận A.
Giá trị riêng của A là nghiệm của đa thức đặc trưng :

$$K_A(\lambda) = \det(\lambda I_n - A)$$

Lưu ý: $K_A(\lambda)$ là đa thức một biến:

$$K_A(\lambda) = \lambda^n + p_1\lambda^{n-1} + ... + p_{n-1}\lambda + p_n.$$

Vector riêng của A là nghiệm của hệ thuần nhất:

$$(\lambda I_n - A)X = 0$$

1.2 Hệ số của đa thức đặc trưng

Đa thức đặc trưng của ma trận A được xác định như sau:

$$K_A(\lambda) = \det(\lambda I_n - A) = \begin{vmatrix} \lambda - a_{11} & -a_{12} & \dots & -a_{1n} \\ -a_{21} & \lambda - a_{22} & \dots & -a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \dots & \lambda - ann \end{vmatrix}$$

$$= \lambda^n - \sigma_1 \lambda n - 1 - ... + (-1)^n \sigma_n.$$

trong đó

$$\sigma_1 = \sum_{i=1}^{n} a_{ii} = trace(A)$$

là tổng các định thức con bậc 1 trên đường chéo chính của A,

$$\sigma_2 = \sum_{i<j} \begin{vmatrix} a_{ii} & a_{ij} \\ a_{ji} & a_{jj} \end{vmatrix}$$

là tổng của các định thức con bậc 2 trên đường chéo chính của A,

$$\sigma_3 = \sum_{i<j<k} \begin{vmatrix} a_{ii} & a_{ij} & a_{ik} \\ a_{ji} & a_{jj} & a_{ik} \\ a_{ki} & a_{kj} & a_{kk} \end{vmatrix}$$

là tổng của các định thức con bậc 3 trên đường chéo chính của A, và cứ tiếp tục. Đến cuối cùng ta có,

$$\sigma_n = \det(A).$$

Nhận xét: Ta thấy có tất cả C_n^k định thức con bậc k trên đường chéo chính của ma trận A. Khi đó số phép tính cần thực hiện để tính hệ số của đa thức đặc trưng là:

$$C_n^1 + C_n^2 + ... + C_n^n = 2^n - 1$$

phép tính.

1.3 Khai triển định thức ma trận theo Laplace

Phần bù đại số (i, j) của ma trận A là A_{ij} xác định bởi:

$$A_{ij} = (-1)^{i+j} D_{ij}$$

trong đó D_{ij} là định thức con của A được tạo ra bằng cách xóa đi hàng thứ i cột thứ j của ma trận A.
Với $i, j \in \{1, 2, ..., n\}$. Khi đó định thức của A thỏa mãn:

$$\begin{aligned} |A| &= a_{i1}A_{i1} + a_{i2}A_{i2} + ... + a_{in}A_{in} \\ &= a_{j1}A_{j1} + a_{j2}A_{j2} + ... + a_{jn}A_{j2n} \\ &= \sum_{j'=1}^{n} a_{ij'}A_{ij'} = \sum_{i'=1}^{n} a_{i'j}A_{i'j} \end{aligned}$$

các biểu thức trên lần lượt được gọi là khai triển Laplace theo hàng i và theo cột j của ma trận A.

1.4 Ma trận đồng dạng

Cho A và B là 2 ma trận vuông cấp n. A và B được gọi là đồng dạng với nhau nếu tồn tại ma trận T không suy biến ($\det T \neq 0$) sao cho:

$$B = T^{-1}AT$$

ký hiệu là: $A \sim B$.

1.5 Tính chất

Cho A và B là 2 ma trận vuông. Ta có các tính chất sau:

a) Nếu X là vector riêng của ma trận A ứng với trị riêng λ thì CX (C=const) cũng là vector riêng ứng với trị riêng λ đó.

b) Nếu $A \sim B$ thì $B \sim A$

c) Nếu $A \sim B, B \sim C$ thì $A \sim C$.

d) Nếu $A \sim B$ thì A, B có cùng trị riêng.

e) Nếu $A = \begin{bmatrix} B & D \\ 0 & C \end{bmatrix}$ hoặc $A = \begin{bmatrix} B & 0 \\ D & C \end{bmatrix}$ với B, C là các ma trận vuông thì $\det A = \det B . \det C$.

Chương 2

Tìm giá trị riêng bằng phương pháp Danilevski

Ta thấy, để tìm giá trị riêng của một ma trận ta cần tìm đa thức đặc trưng của ma trận đó và từ đó tính nghiệm của đa thức. Phương pháp đơn giản nhất để tìm hệ số của đa thức đặc trưng đó là sử dụng khai triển trực tiếp (phần 1.2), tuy nhiên độ phức tạp của thuật toán sẽ là $2^n - 1$. Hay khi sử dụng khai triển Laplace để tính định thức $\det(\lambda I_n - A)$, ta cũng cần phải dùng $n.n!$ phép tính.

Ý tưởng của phương pháp Danilevski là đưa ma trận ban đầu A về ma trận đồng dạng F mà việc tính đa thức đặc trưng của ma trận F có thể dễ dàng thực hiện. Từ đó, tìm được trị riêng của F, đồng thời cũng là trị riêng của A (ma trận F là ma trận Frobenius).

2.1 Ma trận Frobenius

2.1.1 Khái niệm

Ma trân Frobenius là ma trận có dạng

$$F = \begin{pmatrix} p_1 & p_2 & p_3 & \dots & p_{n-1} & p_n \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix}$$

2.1.2 Đa thức đặc trưng của mà trận Frobenius

Ta đặt:

$$F_n = \det(F - \lambda E) = \begin{vmatrix} p_1 - \lambda & p_2 & p_3 & \dots & p_{n-1} & p_n \\ 1 & -\lambda & 0 & \dots & 0 & 0 \\ 0 & 1 & -\lambda & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & -\lambda \end{vmatrix}$$

Sử dụng khai triển Laplace theo cột n, ta có:

$$F_n = p_n(-1)^{1+n}M_{1n} + (-\lambda)(-1)^{n+n}M_{nn}$$

trong đó,

$$M_{1n} = \begin{vmatrix} 1 & -\lambda & 0 & \dots & 0 & 0 \\ 0 & 1 & -\lambda & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 1 \end{vmatrix}$$

,

$$M_{nn} = \begin{vmatrix} p_1 - \lambda & p_2 & p_3 & \dots & p_{n-2} & p_{n-1} \\ 1 & -\lambda & 0 & \dots & 0 & 0 \\ 0 & 1 & -\lambda & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & 1 \end{vmatrix}$$

Dễ thấy,

$$M_{1n} = 1 v M_{nn} = F_{n-1}$$

Nên ta có công thức truy hồi:

$$F_n = (-\lambda)F_{n-1} + (-1)^{n+1}p_n$$

Khai triển công thức truy hồi:

$F_n =$

$=(-\lambda)F_{n-1} + (-1)^{n+1}p_n$

$=(-\lambda)\left[(-\lambda)F_{n-2} + (-1)^n p_{n-1}\right] + (-1)^{n+1}p_n$

$=(\lambda)^2 F_{n-2} + (-1)^n(-\lambda)p_{n-1} + (-1)^{n+1}p_n$

$=\dots=(-1)^n\left[\lambda^n - p_1\lambda^{n-1} - \dots - p_{n-1}\lambda - p_n\right]$

2.2 Phương pháp cho trường hợp đơn giản

Mục đích chung của phương pháp là đưa ma trận ban đầu, qua một số hữu hạn các bước về dạng của ma trận Frobenius. Trong trường hợp lý tưởng, ta giả sử các phần tử $a^{(i+1)}_{n-i,n-i-1}$ tại bước biến đổi thứ $i+1$ là khác không. Các bước thực hiện như sau:
Bước 1: Đặt ma trận ban đầu là $A = A^{(1)}$

$$A^{(1)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1,n-1}^{(1)} & a_{1,n}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \dots & a_{2,n-1}^{(1)} & a_{2,n}^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n-1,1}^{(1)} & a_{n-1,2}^{(1)} & \dots & a_{n-1,n-1}^{(1)} & a_{n-1,n}^{(1)} \\ a_{n,1}^{(1)} & a_{n,2}^{(1)} & \dots & a_{n,n-1}^{(1)} & a_{n,n}^{(1)} \end{bmatrix}$$

Đưa ma trận $A^{(1)}$ về mà trận đồng dạng $A^{(2)}(A^{(1)} \sim A^{(2)})$, với ma trận $A^{(2)}$ có dạng như sau:

$$A^{(2)} = \begin{bmatrix} a_{11}^{(2)} & a_{12}^{(2)} & \dots & a_{1,n-1}^{(2)} & a_{1,n}^{(2)} \\ a_{21}^{(2)} & a_{22}^{(2)} & \dots & a_{2,n-1}^{(2)} & a_{2,n}^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n-1,1}^{(2)} & a_{n-1,2}^{(2)} & \dots & a_{n-1,n-1}^{(2)} & a_{n-1,n}^{(2)} \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$

Ta biến đổi $A^{(2)} = M_1 A^{(1)} M_1^{-1}$, với các ma trận M_1, M_1^{-1} như sau:

$$M_1 = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n-1} & a_{n,n} \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}$$

và,

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -\frac{a_{n,1}^{(1)}}{a_{n,n-1}^{(1)}} & -\frac{a_{n,2}^{(1)}}{a_{n,n-1}^{(1)}} & \dots & -\frac{1}{a_{n,n-1}^{(1)}} & -\frac{a_{n,n}^{(1)}}{a_{n,n-1}^{(1)}} \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}$$

Bước i: Biến đổi ma trận $A^{(i)}$ thành ma trận $A^{(i+1)}$, với $A^{(i+1)} = M_i A^{(i)} M_i^{-1}$

$$M_i = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots & \vdots \\ a^{(i)}_{n-i+1,1} & a^{(i)}_{n-i+1,2} & \dots & a^{(i)}_{n-i+1,n-i} & \dots & a^{(i)}_{n-i+1,n-1} & a^{(i)}_{n-i+1,n} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix}$$

$$M_i^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ -\frac{a^{(i)}_{n-i+1,1}}{a^{(i)}_{n-i+1,n-i}} & -\frac{a^{(i)}_{n-i+1,2}}{a^{(i)}_{n-i+1,n-i}} & \dots & -\frac{1}{a^{(i)}_{n-i+1,n-i}} & \dots & -\frac{a^{(i)}_{n-i+1,n}}{a^{(i)}_{n-i+1,n-i}} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & \dots & 1 \end{bmatrix}$$

Thực hiện $n-1$ lần phép biến đổi như trên, ta thu được ma trận $A^{(n)} = F$ đồng dạng với ma trận ban đầu.

$$A^{(n)} = F = M_{n-1}A^{(n-1)}M_{n-1}^{-1} = M_{n-1}M_{n-2}A^{(n-2)}M_{n-2}^{-1}M_{n-1}^{-1}$$

$$= \dots$$

$$= \Big(M_{n-1}M_{n-2}\dots M_2M_1)A_n^{(1)}(M_1^{-1}M_2^{-1}\dots M_{n-2}^{-1}M_{n-1}^{-1}\Big)$$

Đặt $M = M_{n-1}M_{n-2}\dots M_2M_1$ thì
$M^{-1} = (M_{n-1}M_{n-2}\dots M_2M_1)^{-1} = M_1^{-1}M_2^{-1}\dots M_{n-2}^{-1}M_{n-1}^{-1}$
Khi đó $F = A^{(n)} = MA^{(1)}M^{-1} = MAM^{-1}$ hay $F \sim A$.
Nhận xét: Khối lượng tính toán từ A đến F cỡ $N = n^3 - n^2$ phép nhân và chia. Ta thấy phương pháp Danilevski có khối lượng tính toán giảm đáng kể so với phương pháp thông thường.

2.3 Phương pháp cho các trường hợp đặc biệt

Ở trường hợp lý tưởng, ta giả thiết phần tử $a^{(k)}_{n-k+1,n-k}$ của ma trận $A^{(k)}$ khác không. Tuy nhiên, sau các phép biến đổi có thể xảy ra các trường hợp mà $a^{(k)}_{n-k+1,n-k} = 0$. Ở phần này, ta sẽ đưa ra phương pháp để xử lý các trường hợp đặc biệt đó.

2.3.1 Trường hợp 1

Trong hàng $n-k+1$ của ma trận $A^{(k)}$ tồn tại phần tử $a^{(k)}_{n-k+1,j} \neq 0$ với $j < n-k$. Khi đó ma trận $A^{(k)}$ có dạng:

$$A^{(k)} = \begin{bmatrix} a^{(k)}_{11} & \dots & a^{(k)}_{1,j} & a^{(k)}_{1,j+1} & \dots & a^{(k)}_{1,n-k} & a^{(k)}_{1,n-k+1} & \dots & a^{(k)}_{1,n} \\ a^{(k)}_{21} & \dots & a^{(k)}_{2,j} & a^{(k)}_{2,j+1} & \dots & a^{(k)}_{2,n-k} & a^{(k)}_{2,n-k+1} & \dots & a^{(k)}_{2,n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a^{(k)}_{n-k,1} & \dots & a^{(k)}_{n-k,j} & a^{(k)}_{n-k,j+1} & \dots & a^{(k)}_{n-k,n-k} & a^{(k)}_{n-k,n-k+1} & \dots & a^{(k)}_{n-k,n} \\ 0 & \dots & a^{(k)}_{n-k+1,j} & 0 & \dots & 0 & a^{(k)}_{n-k+1,n-k+1} & \dots & a^{(k)}_{n-k+1,n} \\ 0 & \dots & 0 & 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}$$

Trong trường hợp này, ta hoán vị cột j và cột $n-k$, hàng j và hàng $n-k$ sẽ thu được một ma trận mới $\overline{A}^{(k)} = CA^{(k)}C^{-1}$ đồng dạng với ma trận $A^{(k)}$ ban đầu. Với ma trận C như sau ($C = C^{-1}$)

$$C = \begin{bmatrix} 1_{1,1} & 0_{1,2} & \dots & 0_{1,n-k} & 0_{1,j+1} & \dots & 0_{1,n-k-1} & 0_{1,j} & \dots & 0_{1,n} \\ 0_{2,1} & 1_{2,2} & \dots & 0_{2,n-k} & 0_{2,j+1} & \dots & 0_{2,n-k-1} & 0_{2,j} & \dots & 0_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0_{j-1,1} & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0_{j,1} & 0 & \dots & 0 & 0 & \dots & 0 & 1_{j,j} & \dots & 0 \\ 0_{j+1,1} & 0 & \dots & 0 & 1_{j+1,j+1} & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0_{n-k,1} & 0 & \dots & 1_{n-k,n-k} & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0_{n,1} & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 1_{n,n} \end{bmatrix}$$

Dễ thấy, ma trận $\overline{A}^{(k)}$ có phần tử $\overline{a}^{(k)}_{n-k+1,n-k} \neq 0$ thoả mãn để tiếp tục biến đổi theo phương pháp trong trường hợp đơn giản.

2.3.2 Trường hợp 2

Ta xét trường hợp các phần tử hàng $n-k+1$ đứng trước phần tử $a^{(k)}_{n-k+1,n-k}$ đều bằng không. Hay $a^{(k)}_{j,n-k} = 0 \quad \forall j = 1, n-k$.
Khi đó ma trận $A^{(k)}$ có dạng:

$$A^{(k)} = \begin{bmatrix} a^{(k)}_{11} & \dots & a^{(k)}_{1,n-k-1} & a^{(k)}_{1,n-k} & a^{(k)}_{1,n-k+1} & \dots & a^{(k)}_{1,n-1} & a^{(k)}_{1,n} \\ a^{(k)}_{21} & \dots & a^{(k)}_{2,n-k-1} & a^{(k)}_{2,n-k} & a^{(k)}_{2,n-k+1} & \dots & a^{(k)}_{2,n-1} & a^{(k)}_{2,n} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a^{(k)}_{n-k,1} & \dots & a^{(k)}_{n-k,n-k-1} & a^{(k)}_{n-k,n-k} & a^{(k)}_{n-k,n-k+1} & \dots & a^{(k)}_{n-k,n-1} & a^{(k)}_{n-k,n} \\ 0 & \dots & 0 & 0 & a^{(k)}_{n-k+1,n-k+1} & \dots & a^{(k)}_{n-k+1,n-1} & a^{(k)}_{n-k+1,n} \\ 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$

Ta có thể biểu diễn ma trận $A^{(k)}$ thành ma trận khối như sau:

$$A^{(k)} = \begin{bmatrix} A^{(k1)} & A^{(k2)} \\ 0 & A^{(k3)} \end{bmatrix}$$

Với:

$$A^{(k1)} = \begin{bmatrix} a_{11}^{(k)} & \dots & a_{1,n-k-1}^{(k)} & a_{1,n-k}^{(k)} \\ a_{21}^{(k)} & \dots & a_{2,n-k-1}^{(k)} & a_{2,n-k}^{(k)} \\ \vdots & \ddots & \vdots & \vdots \\ a_{n-k,1}^{(k)} & \dots & a_{n-k,n-k-1}^{(k)} & a_{n-k,n-k}^{(k)} \end{bmatrix}$$

$$A^{(k2)} = \begin{bmatrix} a_{1,n-k+1}^{(k)} & \dots & a_{1,n-1}^{(k)} & a_{1,n}^{(k)} \\ a_{2,n-k+1}^{(k)} & \dots & a_{2,n-1}^{(k)} & a_{2,n}^{(k)} \\ \vdots & \ddots & \vdots & \vdots \\ a_{n-k,n-k+1}^{(k)} & \dots & a_{n-k,n-1}^{(k)} & a_{n-k,n}^{(k)} \end{bmatrix}$$

$$A^{(k3)} = \begin{bmatrix} a_{n-k+1,n-k+1}^{(k)} & \dots & a_{n-k+1,n-1}^{(k)} & a_{n-k+1,n}^{(k)} \\ 1 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 \\ 0 & \dots & 1 & 0 \end{bmatrix}, \quad 0 = \begin{bmatrix} 0 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & 0 \end{bmatrix}$$

Ta thấy $A^{(k3)}$ là một ma trận Frobenius cấp k, và từ tính chất (1.5e) ta tiếp tục biến đổi theo phương pháp Danilevski với ma trận $A^{(k1)}$ vuông có cấp $n-k$. Từ đó, thu được trị riêng của ma trận A bao gồm các trị riêng của $A^{(k1)}$ và $A^{(k3)}$

Chương 3

Tìm vector riêng bằng phương pháp Danilevski

3.1 Phương pháp cho trường hợp đơn giản

Giả sử Y là vectơ riêng ứng với trị riêng λ của ma trận F. Khi đó, λ cũng là trị riêng của ma trận A, ta có:

$$MAM^{-1}Y = \lambda Y \Leftrightarrow AM^{-1}Y = \lambda M^{-1}Y$$

$$\text{Đặt } X = M^{-1}Y = (M_{n-1}M_{n-2}\dots M_2M_1)^{-1}Y \text{ hay } X = \left(M_1^{-1}M_2^{-1}\dots M_{n-2}^{-1}M_{n-1}^{-1}\right)Y$$

$$\Rightarrow AX = \lambda X$$

Vậy $X = \left(M_1^{-1}M_2^{-1}\dots M_{n-2}^{-1}M_{n-1}^{-1}\right)Y$ là vector riêng ứng với giá trị riêng λ của ma trận A.
Ta cần tìm vector riêng của ma trận F, ta có:

$$FY = \lambda Y \quad \text{với} \quad Y = (y_1, y_2, \dots, y_n)^t,$$

$$\Leftrightarrow \begin{bmatrix} p_1 & p_2 & \dots & p_{n-1} & p_n \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \lambda \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} p_1y_1 + p_2y_2 + \ldots + p_ny_n = 0 \\ y_1 = \lambda y_2 \\ y_2 = \lambda y_3 \\ \ldots \\ y_{n-2} = \lambda y_{n-1} \\ y_{n-1} = \lambda y_n \end{cases}$$

Chọn $y_n = 1$, suy ra: $Y = \left(\lambda^{n-1}, \lambda^{n-2}, \ldots, \lambda, 1\right)^t$. Từ đó, ta thu được vector riêng X tương ứng với giá trị riêng λ của ma trận A.

3.2 Phương pháp cho các trường hợp đặc biệt

3.2.1 Trường hợp 1

Trong hàng $n-k+1$ của ma trận $A^{(k)}$ tồn tại phần tử $a^{(k)}_{n-k+1,j} \neq 0$ với $j < n-k$. Khí đó vector riêng của A như sau:

$$X = M^{-1}Y = \left(M_1^{-1}M_2^{-1} \ldots C \ldots M_{n-2}^{-1}M_{n-1}^{-1}\right) Y$$

với ma trận C được cho như ở mục (2.3.1).

3.2.2 Trường hợp 2

Ta xét trường hợp các phần tử hàng $n-k+1$ đứng trước phần tử $a^{(k)}_{n-k+1,n-k}$ đều bằng không. Hay $a^{(k)}_{j,n-k} = 0 \quad \forall j = 1, n-k$.

Như đã trình bày ở phần (2.3.2), ma trận $A^{(k)}$ có dạng khối như sau:

$$A^{(k)} = \begin{bmatrix} A^{(k1)} & A^{(k2)} \\ 0 & A^{(k3)} \end{bmatrix}$$

Ta sẽ có gắng triệt tiêu ma trận $A^{(k2)}$ với phép biến đổi đồng dạng. Giả sử ta muốn triệt tiêu cột $q\,(q < n)$ của ma trận $A^{(k)}$ (nằm trong ma trận $A^{(k2)}$).
Biến đổi đồng dạng $\widehat{A}^{(k)} = S_q A^{(k)} S_q^{-1}$ với:

$$S_q = \begin{bmatrix} 1 & 0 & 0 & \dots & -a_{1q} & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & -a_{2q} & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & -a_{3q} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -a_{n-k+1,q} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix}$$

và,

$$S_q^{-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & a_{1q} & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & a_{2q} & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & a_{3q} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{n-k+1,q} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix}$$

Tiếp tục biến đổi các cột như trên đến cột thứ $n-1$, ta thu được ma trận như sau:

$$\widehat{A}^{(k)} = \begin{bmatrix} a_{11}^{(k)} & \dots & a_{1,n-k-1}^{(k)} & a_{1,n-k}^{(k)} & 0 & \dots & 0 & b_{1,n}^{(k)} \\ a_{21}^{(k)} & \dots & a_{2,n-k-1}^{(k)} & a_{2,n-k}^{(k)} & 0 & \dots & 0 & b_{2,n}^{(k)} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-k,1}^{(k)} & \dots & a_{n-k,n-k-1}^{(k)} & a_{n-k,n-k}^{(k)} & 0 & \dots & 0 & b_{n-k,n}^{(k)} \\ 0 & \dots & 0 & 0 & a_{n-k+1,n-k+1}^{(k)} & \dots & a_{n-k+1,n-1}^{(k)} & b_{n-k+1,n}^{(k)} \\ 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \dots & 0 & 0 & 1 & \dots & 1 & 0 \end{bmatrix}$$

Tại đây, ta lại xét 2 trường hợp sau:

- $b_{j,n}^{(k)} = 0 \; \forall \; j = \overline{1, n-k}$
- Tồn tại j sao cho $b_{j,n}^{(k)} \neq 0$

3.2.2.1 Trường hợp 2.1

Với $b_{j,n}^{(k)} = 0 \quad \forall j = \overline{1, n-k}$, khi đó ma trận $A^{(k)}$ có dạng:

$$A^{(k)} = \begin{bmatrix} A^{(k1)} & 0 \\ 0 & A^{(k3)} \end{bmatrix}$$

Ta có $A^{(k3)}$ là một ma trận Frobenius. Ta tiếp tục thực hiện phương pháp Danilevski để đưa ma trận $A^{(k1)}$ về dạng ma trận Frobenius.
Cuối cùng, ma trận A ban đầu sẽ được biến đổi thành một ma trận khối Frobenius có dạng:

$$F = \begin{bmatrix} F_1 & 0 & \dots & 0 \\ 0 & F_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & F_m \end{bmatrix}$$

- Ta xây dựng cách tìm vector riêng của ma trận khối F như sau:

Giả sử $Y = (y_1, y_2, \ldots, y_m)^t$ là vector riêng ứng với trị riêng λ của ma trận F. Với y_i là các vector có cấp tương ứng với ma trận $F_i \quad \forall i = \overline{1, m}$
Ta có:

$$\begin{bmatrix} F_1 & 0 & \ldots & 0 \\ 0 & F_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & F_m \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \lambda \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Mặt khác, ta lại có:

$$\begin{bmatrix} F_1 & 0 & \ldots & 0 \\ 0 & F_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & F_m \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} F_1 y_1 \\ F_2 y_2 \\ \vdots \\ F_m y_m \end{bmatrix}$$

Nên:

$$F_i y_i = \lambda y_i, \quad \forall i = \overline{1, m}$$

Nếu λ không là trị riêng của F_i suy ra $y = 0$. Mặt khác, vì λ là trị riêng của ma trận F nên tồn tại ít nhất một ma trận F_i nào đó nhận λ làm trị riêng.

Giả sử λ là trị riêng của các ma trận $F_{m_1}, F_{m_2}, \ldots, F_{m_k}, y_{m_i}$ là vector riêng ứng với trị riêng λ của ma trận F_{m_i}.

Khi đó ta xây dựng vector riêng của F ứng λ như sau:

$$v_{m_i} = \begin{bmatrix} 0 \\ \ldots \\ y_{m_i} \\ \ldots \\ 0 \end{bmatrix}$$

Việc tìm vector riêng x_{m_i} ứng với trị riêng λ của ma trận A ban đầu, ta vẫn thực hiện tương tự như (3.1).

3.2.2.2 Trường hợp 2.2

Nếu $\exists j$ sao cho $b_{j,n}^{(k)} \neq 0$($b_{j,n}^{(k)}$ là phần tử đầu tiên khác không đi từ dưới lên trên).

Ta xây dựng ma trận U_j như sau:

$$U_j = \begin{bmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ & & & 0 & 1 & 0 & \dots & 0 \\ & & & 0 & 0 & 1 & \dots & 0 \\ & & & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & 0 & 0 & 0 & \dots & 1 \\ & & & 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

Phần khối ma trận chuyển hàng (cột) có cấp $n - j + 1$. Dễ thấy, ma trận nghịch đảo của ma trận U_j là ma trận chuyển vị của nó:

$$U_j^{-1} = U_j^T$$

Thực hiện phép gán $\widehat{A}^{(k)} = U_j A^{(k)} U_j^T$, bắt đầu vòng lặp mới, thực hiện biến đổi từ đầu $\widehat{A}^{(k)}$ sử dụng phương pháp Danilevski.

Chương 4

Thuật toán và các kết quả số

4.1 Thuật toán

Áp dụng các kiến thức lý thuyết đã trình bày ở trên, ta xây dựng thuật toán Danilevski tìm trị riêng và vector riêng như sau:

Algorithm 1 Danilevski

```
Danilevski(A):
    list_eigenvalues = []
    list_eigenvectors = []
    n = A.shape[0]
    m = n
    k = n
    back = eye(n)
    while k > 1 :
        if A[k, k − 1]! = 0 :
            A, inverseM = findSimpleA(A, k)
            back = back * inverseM
        else:
            non = False
            # Tìm A[k, j]
            for j = 1, 2, ..., k − 2 :
                if A[k, j]! = 0 :
                    + Hoán vị cột j và cột k-1 của ma trận A
                    + Hoán vị hàng j và hàng k-1 của ma trận A
                    + Hoán vị cột j và cột k-1 của ma trận back
                    k = k + 1
                    non = True
                    break
```

```
            if not non:
                # Khử cột từ k đến m-1
                for j = k, k + 1, ..., m − 1 :
                    + Tìm M, M1 khử cột j của ma trận A
                    + A = M ∗ A ∗ M1
                    + back = back*M1
                tt = False
                for j = k − 1, k − 2, ..., 1 :
                # Tìm b khác 0
                    if A[j, m]! = 0 :
                        + Tìm ma trận U va U nghịch đảo
                        A = U ∗ A ∗ U′
                        back = back*U’
                        k = m + 1
                        tt = True
                        break
                if tt == False:
                    X = A[k : m, k : m]
                    t = X.shape[0]
                    lamda = findValue(X)
                    list_eigenvalues.append(lamda)
                    + Với mỗi trị riêng trong lamda:
                        + Tìm vector riêng y của ma trận Frobenius X
                        + Tìm vector riêng x của ma trận A ban đầu
                        list_eigenvectors.append(x)
                    m = k
                    A = A[1:m, 1:m]
                    back = eye(m)
        k = k - 1
    lam = findValue(A)
    list_eigenvalues.append(lam)
    + Với mỗi trị riêng trong lam:
        + Tìm vector riêng y của ma trận Frobenius A
        + Tìm vector riêng x của ma trận A ban đầu
        list_eigenvectors.append(x)
```

4.2 Kết quả số

Ví dụ 4.2.1

Tìm trị riêng và vector riêng của ma trận:

$$A = \begin{pmatrix} 1 & 2 & 4 & 16 \\ 2 & 7 & 25 & 125 \\ 4 & 25 & -3 & 81 \\ 16 & 125 & 81 & -111 \end{pmatrix}$$

Ta có các bước giải của phương pháp Danilevski như sau:

```
A 0  =
[[   1    2    4   16]
 [   2    7   25  125]
 [   4   25   -3   81]
 [  16  125   81 -111]]
Simple case
****************************************************************************
A 1  =
[[ 2.09876543e-01 -4.17283951e+00  4.93827160e-02  2.14814815e+01]
 [-2.93827160e+00 -3.15802469e+01  3.08641975e-01  1.59259259e+02]
 [ 8.07407407e+00 -1.61429630e+03 -7.46296296e+01  2.64791111e+04]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  1.42108547e-14]]
Simple case
****************************************************************************
A 2  =
[[ 1.89005644e-01  2.58492788e-03  2.42294927e-01 -4.69651111e+01]
 [ 4.99974827e+03 -1.06189006e+02  2.36260057e+04  5.78746035e+05]
 [ 0.00000000e+00  1.00000000e+00  0.00000000e+00 -3.63797881e-12]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  1.42108547e-14]]
Simple case
****************************************************************************
A 3  =
[[-1.06000000e+02  2.36590000e+04  5.75492000e+05 -3.44200000e+05]
 [ 1.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  1.00000000e+00  0.00000000e+00 -3.63797881e-12]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  1.42108547e-14]]
****************************************************************************
```

Kết quả thu được như sau:

```
Vector rieng cua:  -206.87706426657527
[[-0.06538821]
 [-0.54536325]
 [-0.32914132]
 [ 1.        ]]
Vector rieng cua:  -23.086712601608887
[[  2.19890733]
 [ 17.1510161 ]
 [-25.81662027]
 [  1.        ]]
Vector rieng cua:  0.5841075540698107
[[-118.50791328]
 [  15.78293687]
 [   0.43016805]
 [   1.        ]]
Vector rieng cua:  123.37966931411435
[[0.18084432]
 [1.2700577 ]
 [0.89788825]
 [1.        ]]
```

Ví dụ 4.2.2

Tìm trị riêng và vector riêng của ma trận sau:

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 6 & 7 & 8 \\ 3 & 7 & 0 & 0 \\ 4 & 8 & 0 & 1 \end{pmatrix}$$

Ta có các bước giải của phương pháp Danilevski như sau:

```
A 0  =
[[1 2 3 4]
 [2 6 7 8]
 [3 7 0 0]
 [4 8 0 1]]
Case 1
********************************************************************************
Doi vi tri cot:
[[0 7 3 0]
 [7 6 2 8]
 [3 2 1 4]
 [0 8 4 1]]
Simple case
********************************************************************************
A 1  =
[[ 0.    1.    0.75 -0.75]
 [ 7.    2.    0.5   7.5 ]
 [68.   16.    6.    75.  ]
 [ 0.    0.    1.    0.  ]]
Simple case
********************************************************************************

A 2  =
[[-4.2500e+00  6.2500e-02  3.7500e-01 -5.4375e+00]
 [-3.1300e+02  1.2250e+01  9.6500e+01 -3.9975e+02]
 [ 0.0000e+00  1.0000e+00  0.0000e+00  0.0000e+00]
 [ 0.0000e+00  0.0000e+00  1.0000e+00  0.0000e+00]]
Simple case
********************************************************************************
A 3  =
[[   8.  129. -107.    3.]
 [   1.    0.    0.    0.]
 [   0.    1.    0.    0.]
 [   0.    0.    1.    0.]]
********************************************************************************
```

Kết quả thu được như sau:

```
Vector rieng cua:  -8.552000310353327
[[-0.52065297]
 [-0.93367355]
 [ 0.94687482]
 [ 1.        ]]
Vector rieng cua:  0.029057125094363694
[[-1.62545157]
 [ 0.69135792]
 [-1.26816508]
 [ 1.        ]]
Vector rieng cua:  0.766185720044297
[[ 1.15292158]
 [-0.60568758]
 [-1.01939813]
 [ 1.        ]]
Vector rieng cua:  15.756757465214665
[[0.64014782]
 [1.52452078]
 [0.79915483]
 [1.        ]]
```

Ví dụ 4.2.3

Tìm trị riêng và vector riêng của ma trận sau:

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Ta có các bước giải của phương pháp Danilevski như sau:

```
A 0  =
[[2 1 0]
 [0 2 0]
 [0 0 1]]
Case 2
********************************************************************************
Bien doi ve dang co cot n - b(i,n)
A 1  =
[[2 1 0]
 [0 2 0]
 [0 0 1]]
Case 2.1
********************************************************************************
```

```
A 1  =
[[2 1]
 [0 2]]
Case 2
********************************************************************************
Bien doi ve dang co cot n - b(i,n)
A 2  =
[[2 1]
 [0 2]]
Case 2.2
********************************************************************************
Nhan A 2  voi U( 3 )
[[2. 0.]
 [1. 2.]]
********************************************************************************
A 1  =
[[2. 0.]
 [1. 2.]]
Simple case
********************************************************************************
A 2  =
[[ 4. -4.]
 [ 1.  0.]]
********************************************************************************
```

Kết quả thu được như sau:

```
Vector rieng cua:  1.0
[[0.]
 [0.]
 [1.]]

Vector rieng cua:  2.0
[[1.]
 [0.]
 [0.]]
Vector rieng cua:  2.0
[[1.]
 [0.]
 [0.]]
```

Ví dụ 4.2.4

Tìm trị riêng và vector riêng của ma trận sau:

$$A = \begin{pmatrix} -2 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & -2 \end{pmatrix}$$

Ta có các bước giải của phương pháp Danilevski như sau:

```
A 0  =
[[-2  1  0  0  0]
 [ 1 -2  1  0  0]
 [ 0  1 -2  1  0]
 [ 0  0  1 -2  1]
 [ 0  0  0  0 -2]]
Case 2
*****************************************************************************
Bien doi ve dang co cot n - b(i,n)
A 1  =
[[-2  1  0  0  0]
 [ 1 -2  1  0  0]
 [ 0  1 -2  1  0]
 [ 0  0  1 -2  1]
 [ 0  0  0  0 -2]]
Case 2.2
*****************************************************************************
Nhan A 1  voi U( 5 )
[[-2.  1.  0.  0.  0.]
 [ 1. -2.  1.  0.  0.]
 [ 0.  1. -2.  0.  1.]
 [ 0.  0.  0. -2.  0.]
 [ 0.  0.  1.  1. -2.]]
*****************************************************************************
A 0  =
[[-2.  1.  0.  0.  0.]
 [ 1. -2.  1.  0.  0.]
 [ 0.  1. -2.  0.  1.]
 [ 0.  0.  0. -2.  0.]
 [ 0.  0.  1.  1. -2.]]
Simple case
*****************************************************************************
A 1  =
[[-2.  1.  0.  0.  0.]
 [ 1. -2.  1.  0.  0.]
 [ 0.  1. -2.  0.  1.]
 [ 0.  1.  0. -4. -3.]
 [ 0.  0.  0.  1.  0.]]
Case 1
*****************************************************************************
Doi vi tri cot:
[[-2.  0.  1.  0.  0.]
 [ 0. -2.  1.  0.  1.]
 [ 1.  1. -2.  0.  0.]
 [ 0.  0.  1. -4. -3.]
 [ 0.  0.  0.  1.  0.]]
Simple case
*****************************************************************************
```

```
A 2  =
[[ -2.   0.   1.   4.   3.]
 [  0.  -2.   1.   4.   4.]
 [  1.   1.  -6. -11.  -6.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.   1.   0.]]
Simple case
****************************************************************************
A 3  =
[[ -2.   0.   1.   4.   3.]
 [  0.  -8. -21. -20.  -5.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.   1.   0.]]
Case 2
****************************************************************************
Bien doi ve dang co cot n - b(i,n)
A 4  =
[[ -2.   0.   0.   0.  -1.]
 [  0.  -8. -21. -20.  -5.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.   1.   0.]]
Case 2.2
****************************************************************************
Nhan A 4  voi U( 5 )
[[ -8. -21. -20.  -5.   0.]
 [  1.   0.   0.   0.   0.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.  -1.  -2.]]
****************************************************************************
A 0  =
[[ -8. -21. -20.  -5.   0.]
 [  1.   0.   0.   0.   0.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.  -1.  -2.]]
Simple case
****************************************************************************
A 1  =
[[ -8. -21. -20.   5.  10.]
 [  1.   0.   0.   0.   0.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.  -1.  -2.   0.]
 [  0.   0.   0.   1.   0.]]
Simple case
****************************************************************************
```

```
A 2  =
[[ -8. -21.  20.  45.  10.]
 [  1.   0.   0.   0.   0.]
 [  0.  -1.  -2.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.   1.   0.]]
Simple case
****************************************************************************
A 3  =
[[-8. 21. 62. 45. 10.]
 [-1. -2.  0.  0.  0.]
 [ 0.  1.  0.  0.  0.]
 [ 0.  0.  1.  0.  0.]
 [ 0.  0.  0.  1.  0.]]
Simple case
****************************************************************************
A 4  =
[[-10. -37. -62. -45. -10.]
 [  1.   0.   0.   0.   0.]
 [  0.   1.   0.   0.   0.]
 [  0.   0.   1.   0.   0.]
 [  0.   0.   0.   1.   0.]]
****************************************************************************
```

Kết quả thu được như sau:

```
Vector rieng cua:  -3.6180340018430295
[[-1.61803403e+00]
 [ 2.61803408e+00]
 [-2.61803413e+00]
 [ 1.61803400e+00]
 [ 9.47428020e-08]]
Vector rieng cua:  -2.6180339643220423
[[ 6.18034019e-01]
 [-3.81966015e-01]
 [-3.81966049e-01]
 [ 6.18033964e-01]
 [ 6.75169360e-08]]
Vector rieng cua:  -2.0
[[ 1.]
 [ 0.]
 [-1.]
 [ 0.]
 [ 1.]]
Vector rieng cua:  -1.3819660245211556
[[ 6.18034005e-01]
 [ 3.81966013e-01]
 [-3.81966032e-01]
 [-6.18033975e-01]
 [ 3.66802837e-08]]
Vector rieng cua:  -0.38196600931377245
[[-1.61803400e+00]
 [-2.61803400e+00]
 [-2.61803401e+00]
 [-1.61803399e+00]
 [ 1.40114356e-08]]
```

4.3 Đánh giá

Phương pháp Danilevski là phương pháp tìm đúng trị riêng và vector riêng của ma trận vuông ban đầu, tức là không có sai số tính toán, chỉ có sai số phương pháp.

So với phương pháp khai triển trực tiếp tìm đa thức đặc trưng, phương pháp Danilevski tỏ ra hiệu quả hơn với độ phức tạp cỡ $O(n^3)$.

Với các ma trận cỡ lớn, phương pháp phải giải phương trình đa thức đặc trưng bậc lớn, nên không tránh khỏi sai số.

Tài liệu tham khảo

[1] Giáo trình giải tích số - Lê Trọng Vinh - Nhà xuất bản khoa học kĩ thuật

[2] Phương pháp tính và matlab - PGS.TS Lê Trọng Vinh, TH.S Trần Minh Hoàn - Nhà xuất bản Bách Khoa, Hà Nội