

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



BÁO CÁO GIẢI TÍCH SỐ

CHỦ ĐỀ 16: TÌM GIÁ TRỊ RIÊNG TRỌI VÀ GIÁ TRỊ RIÊNG TRỌI TIẾP THEO

Nhóm 13:

Phạm Tuấn Anh - 20195840
Cao Đình Đạt - 20195850
Nguyễn Việt Hoàng - 20195878
Trần Đức Toàn - 20195929
Vũ Văn Xúng - 20195943

Lớp: Toán Tin 01 K64

Giảng viên hướng dẫn: TS. Hà Thị Ngọc Yến

MỤC LỤC

MỤC LỤC	2
LỜI NÓI ĐẦU.....	3
PHẦN I: CƠ SỞ TOÁN HỌC CỦA PHƯƠNG PHÁP LŨY THỪA.....	4
PHẦN II: PHƯƠNG PHÁP LŨY THỪA ĐỂ TÍNH GẦN ĐÚNG GIÁ TRỊ RIÊNG TRỘI	5
PHẦN III: PHƯƠNG PHÁP XUỐNG THANG TÌM GIÁ TRỊ TRỘI RIÊNG TIẾP THEO.	14
PHẦN IV: THUẬT TOÁN VÀ CHƯƠNG TRÌNH	18
PHẦN V: KẾT LUẬN	29

LỜI NÓI ĐẦU

Giải tích số là tên thường gọi của một lĩnh vực toán học chuyên nghiên cứu các phương pháp số giải gần đúng các bài toán thực tế được mô hình hóa bằng ngôn ngữ toán học. Đặc biệt trong thời đại công nghệ 4.0, khi nhân loại hướng tới hầu như bất kể hoạt động nào của con người đều có thể được thay thế thông minh bởi máy móc, tự động hóa; khi sự nghiên cứu thuật toán là thách thức cho con người, thì vai trò giải tích số lại càng quan trọng. Nó đóng vai trò cơ sở, nền tảng cho xây dựng thuật toán trong tương lai.

Để có được lời giải cho bất kì bài toán nào cũng đòi hỏi phải có các dữ kiện của bài toán được thu thập bằng cách đo đạc, thống kê,.. và sau đó là xây dựng mô hình bài toán rồi thực thi chúng. Nhiều bài toán trong thực tế được quy về việc giải hệ thống các phương trình, có thể với kích cỡ đầu vào ma trận lớn. Nếu giải bằng các phương pháp trong đại số tuyến tính, thì việc tính toán là không thể với ma trận kích cỡ lớn vì độ phức tạp. Do vậy việc nghiên cứu giải số là hết sức cần thiết. Có rất nhiều phương pháp số trong việc tìm giá trị riêng và vector riêng, nói chung được chia thành hai loại: giải đúng và giải gần đúng. Trong khuôn khổ bài viết này, chúng em xin phép được trình bày về phương pháp tìm gần đúng giá trị riêng trội và giá trị riêng trội tiếp theo và các véc-tơ riêng tương ứng. Các giá trị riêng trội của ma trận cùng các véc-tơ riêng tương ứng được áp dụng trong nhiều ứng dụng như: xử lý ảnh, mô hình kinh tế động,..

Trong bài viết về giải thuật lũy thừa này chúng em chia làm 5 phần chính: Phần 1: Giới thiệu cơ sở toán học của phương pháp cho chúng ta thấy rõ được bài toán được giải quyết như thế nào qua một vài kết quả quan trọng trong đại số tuyến tính. Phần tiếp theo, phần 2 trình bày về chi tiết phương pháp và xây dựng công thức của phương pháp lũy thừa. Phần 3 trình bày nội dung phương pháp xuống thang để tìm giá trị riêng trội tiếp theo. Phần 4 đưa ra thuật toán cùng sơ đồ khối để minh họa cho phương pháp, bên cạnh đó cũng đưa ra các ví dụ cho bạn đọc nắm được phương pháp dễ dàng hơn và trình bày chương trình được viết trên ngôn ngữ Python. Phần cuối đưa ra kết luận, những đánh giá tổng quan và ứng dụng của phương pháp tong thực tiễn.

Chúng em xin chân thành cảm ơn TS. Hà Thị Ngọc Yên vì những bài giảng của cô để chúng em hoàn thành tài liệu này.

Trong quá trình làm báo cáo, dù rất cẩn thận và tập trung, nhóm vẫn không tránh khỏi những sai sót và khiếm khuyết, rất mong nhận được sự góp ý của cô cùng các bạn trong lớp.

PHẦN I: CƠ SỞ TOÁN HỌC CỦA PHƯƠNG PHÁP LŨY THỪA

1. Khái niệm giá trị riêng và vector riêng

Cho ma trận A là ma trận vuông cấp n trên trường số ($K \in \mathbb{R}; \mathbb{C}$). Số $\lambda \in K$ được gọi là giá trị riêng của ma trận A nếu tồn tại một vectơ $u \neq 0, u \in K^n$ sao cho $Au = \lambda u$. Khi đó vector u được gọi là vector riêng của ma trận A ứng với trị riêng λ .

2. Tính chất của giá trị riêng và vector riêng

- Giá trị riêng λ chính là nghiệm của phương trình $\det(A - \lambda I) = 0$. Hay còn được gọi là phương trình đặc trưng của ma trận A
- Một giá trị riêng có thể có nhiều vector riêng
- Mỗi vector riêng chỉ ứng với 1 giá trị riêng duy nhất
- Nếu ma trận A có giá trị riêng là 0 thì A không khả nghịch. Ngược lại, nếu mọi giá trị riêng của A khác 0 thì A khả nghịch.

3. Một số cách tìm giá trị riêng và vector riêng

1. Giải phương trình $\det(A - \lambda I) = 0$ tìm các giá trị riêng. Ứng với mỗi giá trị riêng λ ta giải hệ phương trình tuyến tính thuần nhất $(A - \lambda I)u = 0$.
2. Sử dụng phương pháp Danhilepski đưa ma trận A về dạng ma trận có phương trình đặc trưng theo công thức để giải và tìm giá trị riêng
3. Sử dụng phương pháp lũy thừa để tính gần đúng giá trị riêng trội và vector riêng tương ứng.

Phương pháp đầu chỉ mang tính chất lý thuyết. Phương pháp Danhilepski tuy đơn giản và nhanh nhưng có một nhược điểm lớn, đó là phải giải một phương trình đa thức bậc n . Điều này khiến cho việc tính toán các giá trị riêng của ma trận có cấp lớn là điều vô cùng khó khăn. Còn phương pháp lũy thừa tuy tốc độ hội tụ chậm, nhưng lại có thể tính được giá trị riêng của mọi ma trận.

PHẦN II: PHƯƠNG PHÁP LŨY THỪA ĐỂ TÍNH GẦN ĐÚNG GIÁ TRỊ RIÊNG TRỌI

Bài toán: Trong kỹ thuật, ta thường xuyên làm việc với các ma trận lớn và việc tính toán chính xác các trị riêng theo các phương pháp thông thường đã biết rất khó khăn. Điều này đòi hỏi ta phải tìm ra 1 phương pháp khác giúp ta giải quyết các ma trận lớn. Phương pháp tính gần đúng trị riêng của ma trận tiêu biểu nhất là phương pháp lũy thừa.

1. Khái niệm giá trị riêng trội.

Giả sử ma trận A cấp n có đủ n trị riêng thực hoặc phức (đơn hoặc bội) và chúng thỏa mãn điều kiện:

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$$

Khi đó λ_1 được gọi là giá trị riêng trội của ma trận A . Vector ứng với λ_1 được gọi là vector riêng trội của ma trận A .

Không phải ma trận nào cũng có giá trị riêng trội. Ví dụ ma trận

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

không có giá trị riêng trội vì 2 giá trị riêng của nó có trị tuyệt đối bằng nhau.

Tương tự, dựa vào các trường hợp nghiệm của phương trình đặc trưng của ma trận A ta có thể có nhiều trường hợp khác nhau của các giá trị riêng.

2. Tìm trị riêng trội đơn, thực.

Giả sử ma trận A cấp n có đủ n trị riêng thực hoặc phức, được sắp xếp theo thứ tự môđun giảm dần sao cho $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$ (1) tức là ma trận A có trị riêng trội.

Các vector riêng tương ứng là $X_1, X_2, X_3, \dots, X_n$ là hệ vector độc lập tuyến tính

Khi đó, ta có vector Y bất kỳ đều là tổ hợp tuyến tính của hệ các vector riêng của A

$$Y = \sum_{i=1}^n C_i X_i = C_1 X_1 + C_2 X_2 + \dots + C_n X_n \text{ với}$$

Ta thực hiện việc tính dãy:

$$AY = A \sum_{i=1}^n C_i X_i = \sum_{i=1}^n C_i AX_i = \sum_{i=1}^n C_i \lambda_i X_i$$

Do ta đã có $AX_i = \lambda_i X_i$. Tiếp tục tính

$$A^2Y = A \sum_{i=1}^n C_i \lambda_i X_i = \sum_{i=1}^n C_i \lambda_i^2 X_i$$

$$A^3Y = A \sum_{i=1}^n C_i \lambda_i^2 X_i = \sum_{i=1}^n C_i \lambda_i^3 X_i$$

... ..

$$A^mY = \sum_{i=1}^n C_i \lambda_i^m X_i \quad (2)$$

Với $C_1 \neq 0$ ta có

$$A^mY = C_1 \lambda_1^m X_1 + \sum_{i=2}^n C_i \lambda_i^m X_i = \lambda_1^m \left[C_1 X_1 + \sum_{i=2}^n C_i \frac{\lambda_i^m}{\lambda_1^m} X_i \right]$$

Do điều kiện (1) nên khi $m \rightarrow \infty$ thì $\left(\frac{\lambda_i}{\lambda_1}\right)^m \rightarrow 0$. Và khi đó, ta có

$$A^mY \rightarrow C_1 \lambda_1^m X_1 \text{ khi } (m \rightarrow \infty)$$

Hay với m đủ lớn thì $A^mY \approx C_1 \lambda_1^m X_1$

Và $A^{m+1}Y \approx C_1 \lambda_1^{m+1} X_1$

$$A(A^mY) \approx \lambda_1 (A^mY)$$

Vậy, đẳng thức trên cho thấy A^mY là vector riêng của ma trận A .

Ta có thể và nên thu nhỏ các vector A^mX sau mỗi lần tính để giảm khối lượng tính toán vì nếu như m lớn thì các hệ số của vector A^mX cũng sẽ rất lớn và việc xảy ra tràn số là khó tránh khỏi.

Để tính toán trị riêng của ma trận từ dãy các vector A^mY , trong sách giáo trình giải tích số của thầy Lê Trọng Vinh có đưa ra công thức:

$$\lambda_1 \approx \frac{(A^{m+1}Y)_j}{(A^mY)_j} \quad \text{với } j = \underline{1, n} \quad (*)$$

Là tỉ số giữa các thành phần thứ $j = \underline{1, n}$ của các vector $(A^{m+1}Y)$ và (A^mY) . Và việc tính toán các vector A^mY sẽ kết thúc khi các tỉ số $(*)$ xấp xỉ bằng nhau. Tuy nhiên công thức này không hoàn toàn đúng trong mọi trường hợp. Ta sẽ xét ví dụ sau:

Ví dụ 1: cho ma trận $A = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$ ma trận này có 2 giá trị riêng $\lambda_1 = 2$ và $\lambda_2 = 1$ thỏa mãn điều kiện (1). Ta thực hiện phương pháp lũy thừa và có được kết quả sau

Y	AY	A^2Y	A^3Y	A^4Y	...	A^mY
1	1	1	1	1	...	1
1	2	4	8	16	...	2^m

Có thể thấy ở ví dụ trên, dãy A^mY có hội tụ về họ vector $k(0,1)$ chính là vector riêng ứng với trị riêng trội $\lambda = 2$, thế nhưng các tỉ số (*) lại không đổi trong suốt quá trình tính toán. Vậy nếu sử dụng công thức (*), quá trình lặp sẽ diễn ra mãi mãi. Thay vào đó, người báo cáo xin đưa ra cách làm khác như sau:

1. Sau mỗi lần tính được 1 vector A^mY , ta sẽ thu nhỏ vector sao cho hệ số lớn nhất bằng 1
2. So sánh các hệ số tương ứng giữa A^mY và $A^{m+1}Y$, nếu chúng gần xấp xỉ nhau hoặc sai lệch không đáng kể thì có thể dừng quá trình tính toán
3. Để tính trị riêng, ta chỉ việc lấy vector A^mY cuối cùng vừa tính được, tính tiếp $A^{m+1}Y$ mà không thu nhỏ, lấy tỷ số tọa độ lớn nhất của 2 vector $A^{m+1}Y$ và A^mY làm trị riêng trội.

Theo cách làm này, ví dụ 1 sẽ cho dãy vector như sau:

Y	AY	A^2Y	A^3Y	...	A^7Y	A^8Y
1	0.5	0.25	0.125	...	0.008	0.004
1	1	1	1	...	1	1

Theo bảng trên, sai lệch giữa 2 vector A^7Y và A^8Y là 0.004 tương đối nhỏ và có thể chấp nhận được (nếu cần chính xác hơn có thể tiếp tục tính cho đến khi đạt được sai số mong muốn).

Từ đây ta tìm được vector riêng tương ứng là X

Ta tính vector riêng theo công thức: $\lambda = \frac{X'AX}{X'X}$

Tốc độ hội tụ: ta sẽ xét 2 ví dụ sau:

Xét ma trận $A = \begin{bmatrix} 7 & 8 \\ 10 & 9 \end{bmatrix}$ quá trình tính trị riêng trội chính xác đến chữ số thứ 3 sau dấu phẩy được thể hiện qua bảng sau:

Y	AY	A ² Y	A ³ Y	A ⁴ Y
1	0.789	0.801	0.800	0.800
1	1	1	1	1

Tương tự với ma trận $B = \begin{bmatrix} -4 & 10 & 7 & 5 \end{bmatrix}$ ta cũng có bảng kết quả sau:

Y	BY	B ² Y	...	B ⁶⁶ Y	B ⁶⁷ Y	B ⁶⁸ Y
1	0.5	0.941	...	0.715	0.714	0.714
1	1	1	...	1	1	1

Nhận xét: Với ma trận A, chỉ cần 4 lần nhân, ta đã thu được dãy vector hội tụ đến một xấp xỉ vừa ý. Thế nhưng với ma trận B thì lại mất 68 lần mới đạt được điều tương tự. Lý do là vì ma trận A có 2 giá trị riêng là $\lambda_1 = 17$ và $\lambda_2 = 7$. Còn ma trận B có 2 giá trị riêng là $\lambda_1 = 10$ và $\lambda_2 = -9$. Ở trên ta đã rút ra kết luận khi $m \rightarrow \infty$ thì $\left(\frac{\lambda_i}{\lambda_1}\right)^m \rightarrow 0$. Và khi đó, ta có

$$A^m Y \rightarrow C_1 \lambda_1^m X_1 \text{ khi } (m \rightarrow \infty)$$

Với m tương đương với số phép nhân mà ta phải thực hiện. Vậy nếu m tối thiểu nhỏ thì tốc độ hội tụ nhanh. Tức là tỷ số $\left|\frac{\lambda_i}{\lambda_1}\right|$ càng nhỏ thì tốc độ hội tụ càng nhanh. Đó là lý do tại sao đối với ma trận A, phương pháp lũy thừa chỉ cần 4 lần tính, nhưng với ma trận B thì lại cần đến gần 70 lần tính.

3. Tìm trị riêng trội, thực bội r .

Giả sử λ_1 thực và bội r , tức là:

$$\{\lambda_1 = \lambda_2 = \dots = \lambda_r \mid \lambda_r > |\lambda_{r+1}| \geq |\lambda_{r+2}| \geq \dots \geq |\lambda_n|\} \quad (3)$$

Khi đó, biểu thức (2) sẽ có dạng:

$$A^m Y = \lambda_1^m (C_1 X_1 + C_2 X_2 + \dots + C_r X_r) + \sum_{i=r+1}^n C_i \lambda_i^m X_i$$

Tương tự như trường hợp trước, khi m đủ lớn, ta có:

$$A(A^m Y) \approx \lambda_1(A^m Y)$$

Với cách làm tương tự với trường hợp λ_1 thực, đơn ta sẽ tìm được giá trị riêng và vector riêng tương ứng.

Tuy nhiên, trong trường hợp λ_1 bội r , sẽ có r vector riêng độc lập tuyến tính ứng với λ_1 nhưng ta chỉ có thể tìm được 1 vector. Và trong quá trình tính toán, dựa vào các vector $A^m Y$, sẽ không thể xác định được liệu λ_1 là đơn hay bội r vì trong cả 2 trường hợp, biểu hiện của các vector đều như nhau. Điều này khiến cho việc tìm các giá trị riêng còn lại gặp nhiều khó khăn do ta không biết ma trận có n trị riêng đơn hay n trị riêng có bội.

4. Trường hợp λ_1 và λ_2 thực trái dấu.

Giả sử $\lambda_1 = -\lambda_2$, khi đó ta có:

$$\{\lambda_1 = -\lambda_2 \mid |\lambda_1| = |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|\} \quad (4)$$

Khi đó:

$$AY = \lambda_1 (C_1 X_1 - C_2 X_2) + \sum_{i=3}^n C_i \lambda_i X_i$$

$$A^2 Y = \lambda_1^2 (C_1 X_1 + C_2 X_2) + \sum_{i=3}^n C_i \lambda_i^2 X_i$$

$$A^3 Y = \lambda_1^3 (C_1 X_1 - C_2 X_2) + \sum_{i=3}^n C_i \lambda_i^3 X_i$$

... ..

$$A^{2k-1} Y = \lambda_1^{2k-1} (C_1 X_1 - C_2 X_2) + \sum_{i=3}^n C_i \lambda_i^{2k-1} X_i$$

$$A^{2k}Y = \lambda_1^{2k}(C_1X_1 + C_2X_2) + \sum_{i=3}^n C_i\lambda_i^{2k}X_i$$

Theo giả thiết (4), khi k đủ lớn:

$$A^{2k-2}Y \approx \lambda_1^{2k-2}(C_1X_1 + C_2X_2)$$

$$A^{2k-1}Y \approx \lambda_1^{2k-1}(C_1X_1 - C_2X_2)$$

$$A^{2k}Y \approx \lambda_1^{2k}(C_1X_1 + C_2X_2)$$

Ta nhận thấy rằng, khi $k \rightarrow \infty$ các vector liên nhau trong dãy lũy thừa không có dấu hiệu hội tụ, thế nhưng các vector ở các bước lũy thừa cùng chẵn hoặc cùng lẻ thì lại cùng nhau tạo thành dãy vector hội tụ. Và đồng thời:

$$\begin{aligned} A^{2k}Y &\approx \lambda_1^{2k}(C_1X_1 + C_2X_2) = \lambda_1^2\lambda_1^{2k-2}(C_1X_1 + C_2X_2) = \lambda_1^2A^{2k-2}Y \\ &\Rightarrow A^{2k}Y \approx \lambda_1^2A^{2k-2}Y \quad (5) \end{aligned}$$

Từ (5), ta có thể tính được xấp xỉ của λ_1^2 tương tự như cách làm ở 2 trường hợp trước rồi từ đó tính được xấp xỉ của λ_1^2 . Để tính được vector riêng tương ứng với λ_1 và $\lambda_2 = -\lambda_1$ ta sẽ đưa các biểu thức trên về dạng $AX = \lambda X$. Ta xét hệ:

$$\begin{aligned} \{A^{2k-1}Y \approx \lambda_1^{2k-1}(C_1X_1 - C_2X_2) \quad A^{2k}Y \approx \lambda_1^{2k}(C_1X_1 + C_2X_2)\} \\ \Leftrightarrow \{A^{2k}Y + \lambda_1A^{2k-1}Y \approx \lambda_1^{2k}2C_1X_1 \quad A^{2k}Y - \lambda_1A^{2k-1}Y \approx \lambda_1^{2k}2C_2X_2\} \\ \Leftrightarrow \{A(A^{2k-1}Y + \lambda_1A^{2k-2}Y) \approx \lambda_1^{2k}2C_1AX_1 \quad A(A^{2k-1}Y - \lambda_1A^{2k-2}Y) \approx \lambda_1^{2k}2C_2AX_2\} \\ \Leftrightarrow \{A(A^{2k-1}Y + \lambda_1A^{2k-2}Y) \approx \lambda_1(\lambda_1^{2k}2C_1X_1) \quad A(A^{2k-1}Y - \lambda_1A^{2k-2}Y) \approx \lambda_2(\lambda_1^{2k}2C_2X_2)\} \end{aligned}$$

$$\begin{aligned}\Leftrightarrow \{A\left(A^{2k}Y + \lambda_1 A^{2k-1}Y\right) &\approx \lambda_1\left(A^{2k}Y + \lambda_1 A^{2k-1}Y\right) A\left(A^{2k}Y - \lambda_1 A^{2k-1}Y\right) \\ &\approx \lambda_2\left(A^{2k}Y + \lambda_1 A^{2k-1}Y\right)\end{aligned}$$

Đến đây ta đã có thể kết luận $A^{2k}Y + \lambda_1 A^{2k-1}Y$ và $A^{2k}Y - \lambda_1 A^{2k-1}Y$ lần lượt là các vector riêng ứng với λ_1 và $\lambda_2 = -\lambda_1$.

5. Trường hợp λ_1 và λ_2 là phức liên hợp.

Giả sử ma trận A có $\lambda_1 = \underline{\lambda_2}$ (phức liên hợp) và:

$$|\lambda_1| = |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n| \quad (6)$$

Biểu thức (2) được viết lại thành:

$$A^m Y = \lambda_1^m C_1 X_1 + \lambda_2^m C_2 X_2 + \sum_{i=3}^n C_i \lambda_i^m X_i$$

Cùng với các lập luận tương tự, từ giả thiết (6), khi m đủ lớn, ta có:

$$\begin{aligned} \{A^m Y \approx \lambda_1^m C_1 X_1 + \lambda_2^m C_2 X_2 \quad A^{m+1} Y \approx \lambda_1^{m+1} C_1 X_1 + \lambda_2^{m+1} C_2 X_2 \quad A^{m+2} Y \\ \approx \lambda_1^{m+2} C_1 X_1 + \lambda_2^{m+2} C_2 X_2 \end{aligned}$$

$$\Rightarrow A^{m+2} Y - (\lambda_1 + \lambda_2) A^{m+1} Y + \lambda_1 \lambda_2 A^m Y = 0 \quad (7)$$

Ta thấy trong suốt quá trình tính toán, dãy các vector không hề hội tụ và các dãy con của nó cũng vậy. Thế nhưng khi m đạt được một giá trị nhất định, 3 vector liên tiếp có dấu hiệu tổ hợp tuyến tính với nhau.

Đặt $p = -(\lambda_1 + \lambda_2)$, $q = \lambda_1 \lambda_2$. Khi đó, λ_1 và λ_2 là nghiệm của phương trình:

$$Z^2 + pZ + q = 0 \quad (8)$$

Viết lại phương trình (7): $A^{m+2} Y + p A^{m+1} Y + q A^m Y = 0 \quad (9)$

Công việc cần làm là tìm λ_1 và λ_2 , tức là tìm nghiệm của phương trình (8).

Từ phương trình (9), ta chọn 2 tọa độ bất kì của các vector (chẳng hạn như $j = r$ và $j = s$), kết hợp với phương trình (8) ta có hệ 3 phương trình:

$$\begin{aligned} \{Z^2 + pZ + q = 0 \quad (A^{m+2} Y)_r - (\lambda_1 + \lambda_2) (A^{m+1} Y)_r + \lambda_1 \lambda_2 (A^m Y)_r \\ = 0 \quad (A^{m+2} Y)_s - (\lambda_1 + \lambda_2) (A^{m+1} Y)_s + \lambda_1 \lambda_2 (A^m Y)_s \\ = 0 \quad (10) \end{aligned}$$

Hệ phương trình (10) có 3 ẩn là 1, p và q và là hệ phương trình thuần nhất. Để hệ có nghiệm khác không thì định thức phải bằng 0. Tức là:

$$\begin{vmatrix} Z^2 & Z & 1 \\ (A^{m+2}Y)_r & (A^{m+1}Y)_r & (A^mY)_r \\ (A^{m+2}Y)_s & (A^{m+1}Y)_s & (A^mY)_s \end{vmatrix} = 0$$

Với các tọa độ tính được từ dãy vector, phương trình trên chính là phương trình (8). Giải ta được cặp nghiệm phức chính là λ_1 và λ_2 .

Để tìm vector riêng, ta sử dụng cách làm tương tự như khi λ_1 và λ_2 là thực trái dấu, ta có:

$$\{A^{m-1}Y \approx \lambda_1^{m-1}C_1X_1 + \lambda_2^{m-1}C_2X_2 \quad A^mY \approx \lambda_1^mC_1X_1 + \lambda_2^mC_2X_2$$

$$\begin{aligned} \Leftrightarrow \{A^mY - \lambda_1A^{m-1}Y &\approx C_2\lambda_2^m(\lambda_2 - \lambda_1)X_2 \quad A^mY - \lambda_2A^{m-1}Y \\ &\approx C_1\lambda_1^m(\lambda_1 - \lambda_2)X_1 \end{aligned}$$

Nhân A vào các vế:

$$\begin{aligned} \Leftrightarrow \{A(A^mY - \lambda_1A^{m-1}Y) &\approx \lambda_2(A^mY - \lambda_1A^{m-1}Y) \quad A(A^mY - \lambda_2A^{m-1}Y) \\ &\approx \lambda_1(A^mY - \lambda_2A^{m-1}Y) \end{aligned}$$

Vậy $A^mY - \lambda_1A^{m-1}Y$ là vector riêng ứng với λ_2 ,
 $A^mY - \lambda_2A^{m-1}Y$ là vector riêng ứng với λ_1 .

PHẦN III: PHƯƠNG PHÁP XUỐNG THANG TÌM GIÁ TRỊ TRỌI RIÊNG TIẾP THEO.

**Ý tưởng phương pháp:*

- Đối với bài toán tìm nghiệm của đa thức, sau khi tìm được $x = x_0$ là nghiệm của $P(x)$, ta loại bỏ nghiệm đã tìm được bằng cách chia $P(x)$ cho đơn thức tương ứng. Khi đó, ta xét:

$$Q(x) = \frac{P(x)}{x - x_0}$$

- Đối với bài toán tìm giá trị riêng trội tiếp theo của ma trận, ta sẽ đưa giá trị riêng vừa tìm được về 0, biến đổi ma trận A ban đầu thành một ma trận mới có các giá trị riêng (trừ giá trị riêng trội vừa tìm) giống A và giá trị riêng trội vừa tìm chuyển thành 0.

**Nội dung phương pháp*

Xét ma trận $A = [a_{ij}]$ là ma trận vuông cấp n có đủ n giá trị riêng là $\lambda_1, \lambda_2, \dots, \lambda_n$ và các véc-tơ riêng tương ứng là X_1, X_2, \dots, X_n

Qua phương pháp lũy thừa, ta đã tìm được giá trị riêng trội của A là λ_1 và véc-tơ riêng tương ứng là V_1

Chuẩn hóa V_1 , ta được véc-tơ có thành phần thứ s bằng 1. Ta gọi véc-tơ mới là $V_1 = (v_1, v_2, \dots, 1, \dots, v_n)^t$

Xét ma trận θ phụ thuộc vào véc-tơ V_1 và chỉ số s nên kí hiệu là $\theta(V_1, s)$

$$\theta(X^{(1)}, i) = \begin{bmatrix} 1 & 0 & \dots & 0 & -x_1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & -x_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & -x_n & 0 & \dots & 1 \end{bmatrix}$$

Với mọi véc-tơ $Z = (z_1, z_2, \dots, z_n)^t$ ta đều có:

$$\theta(x,s)Z = \begin{pmatrix} z_1 - x_1 z_s \\ z_2 - x_2 z_s \\ \dots\dots\dots \\ 0 \\ \dots\dots\dots \\ z_n - x_n z_s \end{pmatrix} = Z - z_s X$$

Do đó, ta có:

$$\theta(x,s)X = X - x_s X = X - 1X = 0$$

Nhận xét: $A_2 = \theta(V_1, s_1)A = A - V_1 a_{s1}$ với $a_s = (a_{s11}, a_{s12}, \dots, a_{s1n})$ là hàng thứ s_1 của ma trận A

Ta sẽ chứng minh ma trận A_2 có các giá trị riêng $\lambda_2, \lambda_3, \dots, \lambda_n$ và 0; các véc-tơ riêng tương ứng là $\theta X_2, \theta X_3, \dots, \theta X_n$ và X_1 .

Đầu tiên, chứng minh 0 và V_1 lần lượt là trị riêng và véc-tơ riêng tương ứng của A_2

Thật vậy, ta có:

$$\begin{aligned} A_2 V_1 &= (\theta(V_1, s_1)A)V_1 = \theta(V_1, s_1)(AV_1) = \theta(V_1, s_1)\lambda_1 V_1 \\ &= \lambda_1(\theta(V_1, s_1)V_1) = 0 \end{aligned}$$

Do đó, 0 và V_1 lần lượt là trị riêng và véc-tơ riêng tương ứng của A_2

Tiếp theo, với $k=2,3,\dots,n$, chứng minh λ_k và $\theta(V_1, s_1) V_k$ là trị riêng và vector riêng của A_2

Suy ra:

$$\begin{aligned} A_2(\theta(V_1, s_1)V_k) &= (\theta(V_1, s_1)A)(\theta(V_1, s_1)V_k) \\ &= (\theta(V_1, s_1)A)(V_k - v_{k(s1)}V_1) \\ &= \theta(V_1, s_1)AV_k - \theta(V_1, s_1)AV_{k(s1)}V_1 \end{aligned}$$

$$= \theta(V_1, s_1) A V_k$$

$$= \theta(V_1, s_1) \lambda_k V_k$$

$$= \lambda_k \theta(V_1, s_1) V_k$$

Nên suy ra λ_k và θV_k lần lượt là trị riêng và véc-tơ riêng của ma trận A_2 với mọi $k=2,3,\dots,n$

Như vậy, sau khi biến đổi ma trận A thành ma trận A_2 , thực hiện theo phương pháp lũy thừa một lần nữa, ta sẽ tìm được giá trị riêng trội tiếp theo là λ_2 . Tuy nhiên, vấn đề đặt ra là *tìm véc-tơ riêng tương ứng với λ_2 của ma trận A như thế nào.*

Để tìm véc-tơ riêng tương ứng với λ_2 của ma trận A , ta xuất phát từ véc-tơ riêng θV_2 của ma trận A_2 .

Ta có:

$$V_2 = \theta(V_1, s_1) V_2 + (V_2)_{s_1} V_1$$

Suy ra

$$A V_2 = A [\theta(V_1, s_1) V_2 + (V_2)_{s_1} V_1]$$

$$= A \theta(V_1, s_1) V_2 + (V_2)_{s_1} A V_1$$

$$= A \theta(V_1, s_1) V_2 + (V_2)_{s_1} \lambda_1 V_1$$

Vì V_2 là véc-tơ riêng ứng với λ_2 của ma trận A nên:

$$A V_2 = \lambda_2 V_2$$

$$= \lambda_2 (\theta(V_1, s_1) V_2 + (V_2)_{s_1} V_1)$$

$$= \lambda_2 \theta(V_1, s_1) V_2 + (V_2)_{s_1} \lambda_2 V_1$$

Suy ra:

$$A \theta(V_1, s_1) V_2 + (V_2)_{s_1} \lambda_1 V_1 = \lambda_2 \theta(V_1, s_1) V_2 + (V_2)_{s_1} \lambda_2 V_1$$

$$\Rightarrow (A - \lambda_2 E) \theta(V_1, s_1) V_2 = (V_2)_{s_1} (\lambda_2 - \lambda_1) V_1$$

Nếu $\lambda_2 = \lambda_1$ ta có $(A - \lambda_2 E)\theta(V_1, s_1)V_2 = 0$, suy ra $\theta(V_1, s_1)V_2$ chính là véc-tơ riêng tương ứng với λ_2 của ma trận A. Hay nói cách khác, trong trường hợp này $(V_2)_{s_1} = 0$

Nếu $\lambda_2 \neq \lambda_1$. Vì V_1 có phần tử thứ s_1 bằng 1, ta có thể tính $(V_2)_{s_1}$ bằng cách chia phần tử thứ s_1 của $(A - \lambda_2 E)\theta X_2$ cho $(\lambda_2 - \lambda_1)$, tức là ta có:

$$x_i^{(2)} = \frac{((A - \lambda_2 E)\theta(V_1, s_1)V_2)_i}{\lambda_2 - \lambda_1}$$

Như vậy, sau k - 1 lần xuống thang ta tìm được (λ_i, v_i) , với $i = 1, 2, \dots, k - 1, A'_{k=2}$

Tìm giá trị riêng, vector riêng tiếp theo từ ma trận A'

$$A_k = \theta(V_{k-1}, s_{k-1})A_{k-1}$$

$$A_k V_{k-1} = \theta(V_{k-1}, s_{k-1})\lambda_{k-1}V_{k-1} = 0$$

$$A_k \theta(V_{k-1}, s_{k-1})v_i = \theta(V_{k-1}, s_{k-1})0v_{k-1} = 0, i < k - 1$$

$$A_k \theta(V_{k-1}, s_{k-1})v_i = \lambda_i \theta(V_1, s_1)v_i, i > k - 1$$

PHẦN IV: THUẬT TOÁN VÀ CHƯƠNG TRÌNH

➤ **Sơ đồ, thuật toán tổng quan:**

- **Các gói xử lý dữ liệu:**

1. Gói lũy thừa:

- Đầu vào: Ma trận A, chỉ số m
- Trả về ma trận $A * (B[m])$ ứng với ma trận $A_{m+1}Y$

2. Gói kiểm tra:

Thực hiện phép trừ $A_{m+1}Y - A_mY$ bằng các vectơ $B[m+1] - B[m]$

Trả về $\max(|B[m+1] - B[m]|)$

3. Gói xử lý dữ liệu:

- Ban đầu lưu $B[0] = Y$
- Tính các A_mY và lưu vào $B[m]$
- Thực hiện kiểm tra ($B[m]$), so sánh với epsilon để chia trường hợp

4. Gói chuẩn hóa Vector:

- Tìm phần tử max trong vector V, lưu lại thành V_{max}
- Trả về Vector V/V_{max}

5. Gói chuyển ma trận (xuống thang ma trận)

- Đầu vào là ma trận A, vector V tại phần tử cuối cùng của mảng B.
- Xây dựng vectơ Theta từ ma trận đơn vị và vectơ V.
- $A' = \text{theta} \cdot A$

6. Gói tìm chuẩn cho V:

Gói này sẽ đưa vectơ V là vectơ riêng của A từ vectơ V'

- Đầu vào : lamda vừa tìm được, lamda của lượt tìm trước, V' vừa tìm được của A', V chuẩn của A ban đầu tại lượt trước, w(vị trí của Vmax trong V' của A')

+ Nếu lamda == lamda trước: trả về vecto vừa tìm được chính bằng vecto chuẩn của A ban đầu cần tìm.

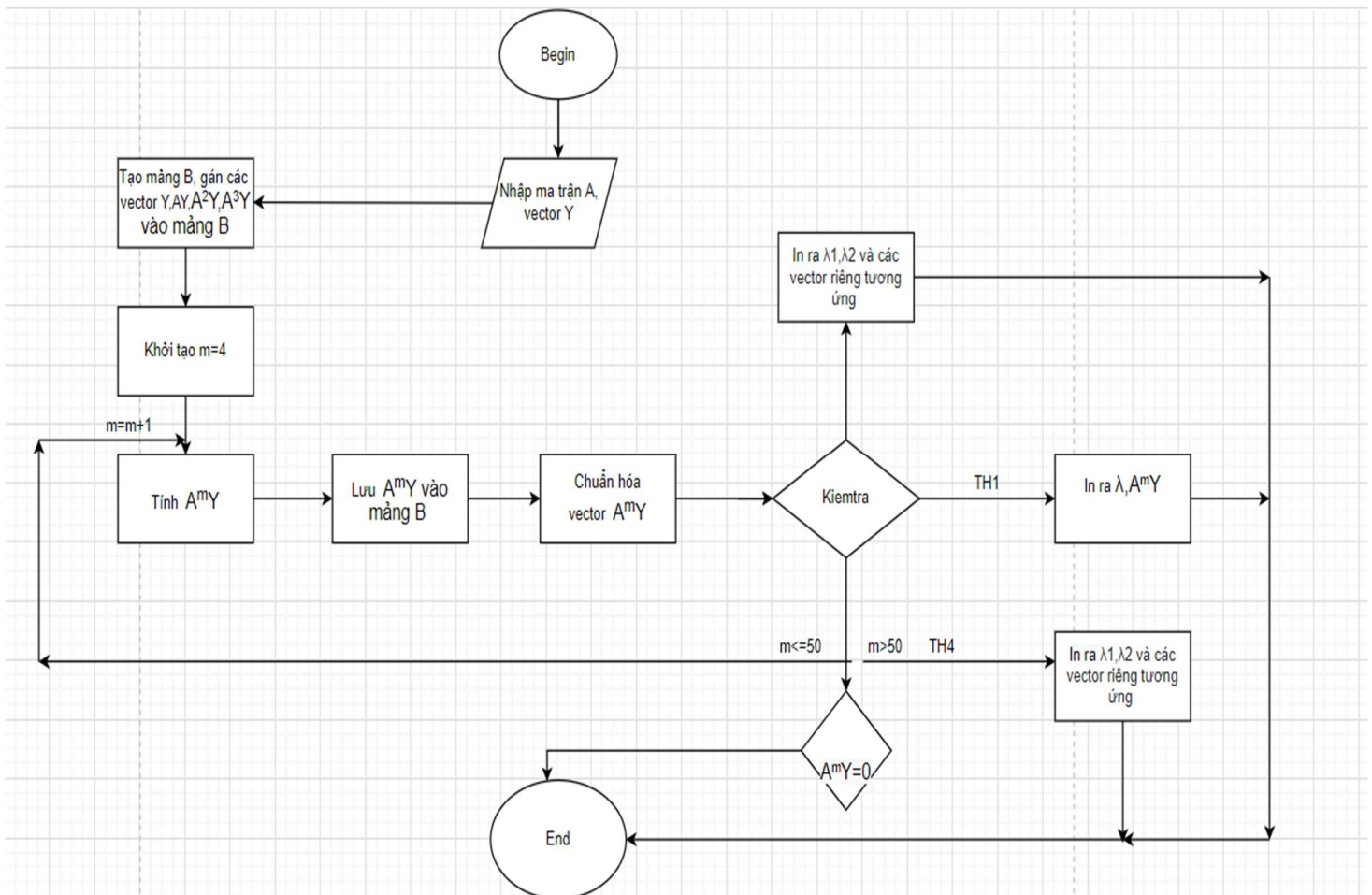
+ Nếu lamda != lamda trước:

Tính $Vw = (A(\text{ban đầu}) - \text{lamda} \cdot E(\text{matran đơn vị})) * V(\text{vừa tìm được})$

Tính hệ số $c = Vw[w][0] / (\text{lamda} - \text{lamda trước})$

Trả về vecto chuẩn của A ban đầu: $V' + c * V$

- Thuật toán lũy thừa:
- Sơ đồ thuật toán:



- **Thuật toán tổng quan:**

B1: Nhập ma trận A và vector Y, nhập vào một ε

B2: Tạo mảng B lưu vector

B3: Lưu Y vào B[0], tính các vector A^1Y, A^2Y, A^3Y rồi lưu vào mảng B theo thứ tự

B4: Khởi tạo $m=4$

B5 Tính A^mY

B6: Chuẩn hóa vector:

Tìm phần tử có trị tuyệt đối lớn nhất trong A^mY , đặt là t

Gán $A^mY = A^mY/t$

B7: Lưu các vector A^mY vào vị trí tiếp theo ở mảng B

B8: Kiểm tra

Nếu A là ma trận thực:

Tính $B[m] - B[n]$

Trả về phần tử có trị tuyệt đối lớn nhất trong $B[m] - B[n]$

Nếu A là ma trận phức:

Tính $B[m] - B[n]$

Trả về phần tử có modul lớn nhất trong $B[m] - B[n]$

B9: While:

Nếu $A^mY = 0$ thì thoát chương trình

Nếu $Kiemtra(B[m]-B[m-1]) < \varepsilon$ thì cho vào TH1

Nếu $Kiemtra(B[m] - B[m-2]) < \varepsilon$ thì cho vào TH3

$m=m+1$, quay lại B4

Nếu lặp đến $m=50$ mà không thuộc TH1 hay TH3 thì thoát khỏi vòng lặp, và nhảy sang TH4

B10: Xử lí

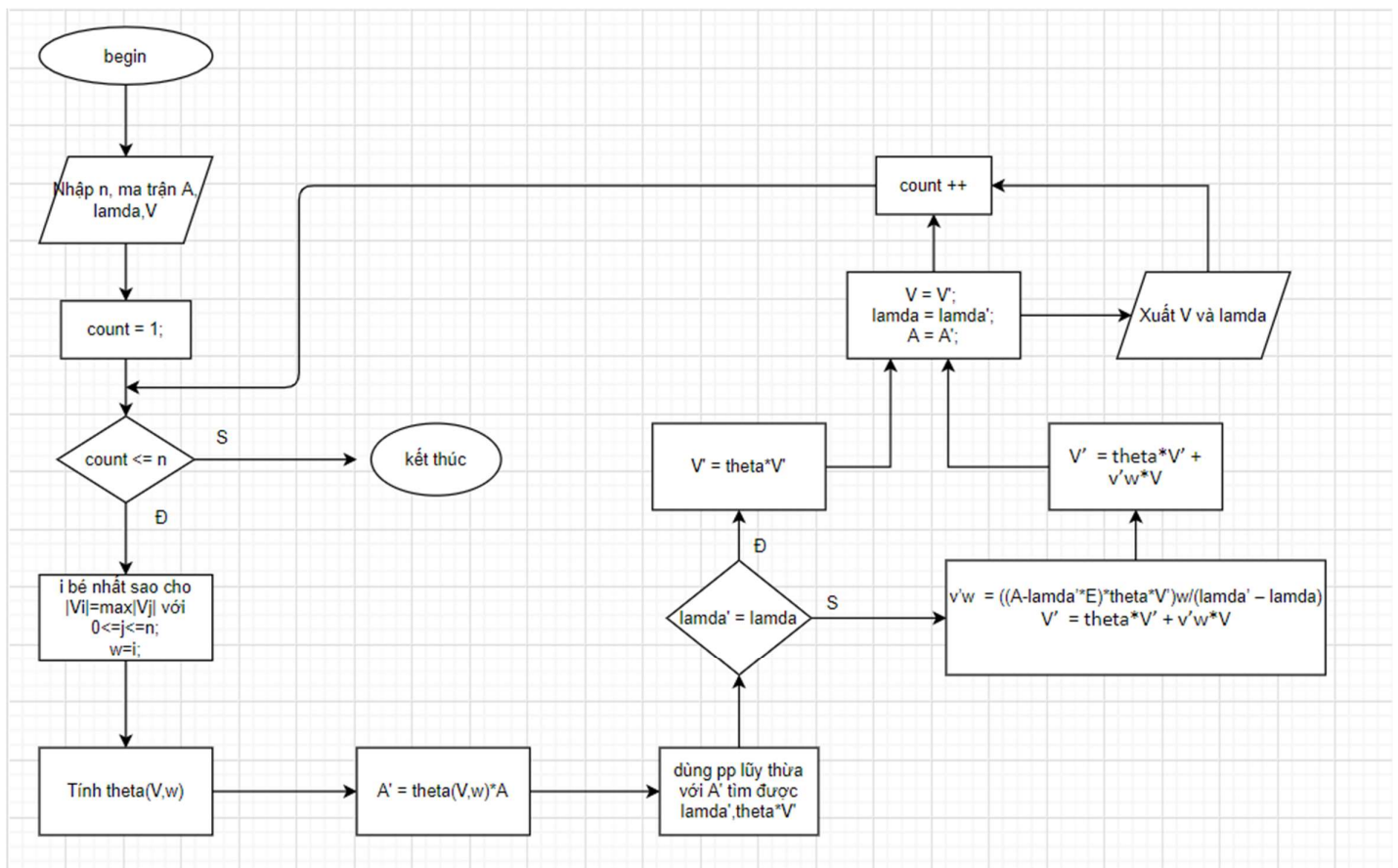
Nếu thuộc TH1, in ra trị riêng $\lambda = \frac{A^m Y(j)}{A^{m-1} Y(j)}$ và vector riêng $A^m Y$

Nếu thuộc TH3, in ra trị riêng $\lambda_1 = -\lambda_2 = \sqrt{\frac{A^{2k+2} Y}{A^{2k} Y}}(j)$ và 2 vector riêng tương ứng là $X_1 = A^{2k} Y + \lambda_1 A^{2k-1} Y$ và $X_2 = A^{2k} Y - \lambda_1 A^{2k-1} Y$

Nếu thuộc TH4 thì in ra màn hình trị riêng và vector riêng sau khi được tính toán

B11: Kết thúc chương trình

- Thuật toán xuống thang:
 - Sơ đồ:



- Thuật toán tổng quan:

Input: matrix A, n, lamda, vecto riêng V

Output: các trị riêng và vecto riêng gần đúng của ma trận A

B0: Nhập ma trận A, số n là cấp của ma trận, trị riêng trội lamda, vecto riêng V tương ứng với lamda

B1. Khởi tạo count = 1;

B2: While(count <= n):

B3: Chuẩn hóa vecto

B3.1 Khởi tạo $i = 1$; $w = 0$; $MAXV = |V[0]|$

B3.2 While $i < N$:

B3.3 So sánh $MAXV < |V[i]|$ thì gán $MAXV = |V[i]|$,
 $w = i$; chuyển đến bước 3.4

B3.4 $i = i + 1$;

B3.5 Quay lại bước 3.2 hoặc S: xuất ra $V = V/V_w$ và kết thúc vòng lặp

hoặc Đ: tiếp tục vòng lặp

B4: Xây dựng ma trận theta(V,w)

B4.1 Khởi tạo $j = 0$; $k = 0$;

B4.2 While ($j < n$; $k < n$)

if($j = k$) $theta[j][k] = 1$;

Else $theta[j][k] = 0$;

$j++$; $k++$;

B4.3 for($j = 0$; $j < n$; $j++$)

$Theta[j][w] = theta[j][w] - V[j]$;

B5. Phân chính:

B5.1 $A' = theta(V,w)*A$;

B5.2 Áp dụng pp lũy thừa với ma trận A' tìm được lamda', $theta*V'$ ứng với lamda' của A'

B5.3 Tìm V' của ma trận A thông qua vecto $thetaV'$

If lamda' = lamda thì $V' = thetaV'$;

Else lamda' != lamda thì $v'w = ((A - lamda'*E)*theta*V')w/(lamda' - lamda)$

$V' = theta*V' + v'w*V$;

B6. $V = V'$; lamda = lamda', $A = A'$ và xuất ra giá trị V và lamda;

B7. count ++

Nếu count < n quay lại bước 2;

Nếu count >= n thì kết thúc chương trình

4 Chương trình:

```
import numpy as np
import math
import sys

# Load file
f = open("input2.txt", "r")
temp = sys.stdout
sys.stdout = open('output.txt', 'w')
a = f.readline()
n = int(a)
print('Ma tran co: {} x {}'.format(n, n))
b = f.readline()
b = b.split()
X = []
for i in b:
    X.append(float(i))
X = np.array(X)
Y = X.reshape(n, 1)
c = f.read()
c = c.split()
A = []
G = []
for i in c:
    A.append(float(i))
    G.append(float(i))
G = np.array(G).reshape(n, n)
A = np.array(A).reshape(n, n)
#G = np.array(G).reshape(n, n)
print('Ma tran A la: ')
print(A)
print(' '*100)
E = 0.001

#Chuẩn hóa
def Chuan_Hoa(M):
    maxi = abs(M[0][0])
    c = M[0][0]
    for i in range(len(M)):
        if maxi < abs(M[i][0]):
            maxi = abs(M[i][0])
            c = M[i][0]
    matrix = M / c
    return matrix

#Kiểm tra
def Kiem_tra(B, m, n):
    A = B[m] - B[n]
    return max(abs(A))

#Lũy thừa
def Luy_thua(A, B, m):
    M = A.dot(B[m])
    return M
    #B= B[m-1] . A

#Xử lý
def Xu_li(A, Y):
    B = []
    B.append(Y)
    Z = np.zeros((n, 1))
    B.append(Chuan_Hoa(Luy_thua(A, B, 0)))
```

```

B.append(Chuan_Hoa(Luy_thua(A, B, 1)))
B.append(Chuan_Hoa(Luy_thua(A, B, 2)))
m = 3
TH = 4
while True:
    M = []
    F = B[-1]
    # for i in range (len(F)):
    #     x = round(F[i][0], 4)
    #     M.append(x)
    #M = np.array(M).reshape(n, 1)
    if sum(F == Z) == n:
        sys.exit()
    if m == 218:
        break
    if Kiem_tra(B, m - 1, m - 2) <= E:
        TH = 1
        break
    if Kiem_tra(B, m - 1, m - 3) <= E:
        TH = 3
        break
    B.append(Chuan_Hoa(Luy_thua(A, B, m)))
    m += 1
return m, TH, B

#Chuyển ma trận xuống thang
def Chuyen_MT(A, VTR):
    B = A
    V = VTR.reshape(n,)
    theta = np.eye(n, dtype=float)
    Vmax = max(V)
    loca = 0
    for i in range (n):
        if V[i] == Vmax:
            loca += i
            break
    E = theta[:, loca : loca +1] - VTR.reshape(n, 1)
    theta[:, loca : loca +1] = E
    A = theta.dot(A)
    # print(loca)
    # sys.exit()
    return loca , A

def timVchuan(lam, lamt, vec, vect, w):
    if lam == lamt:
        return vec
    elif lam != lamt:
        E = np.eye(n)
        v_w = G - lam * E
        v_w = v_w.dot(vec)
        c = v_w[w][0] / (lam - lamt)
        return vec + c * vect

# c = ((A - lamda * E)* theta * V')w / (lam - lamt)

#Tim
C=[]
D=[]
for i in range(n):
    m, TH, B = Xu_li(A, Y)
    if i >= 1 :
        print("Ma tran A sau khi xuong thang: ")
        print(A)
        print("\n")

    if TH == 1:
        VTR = B[-1]
        #VTRt = B[-2]
        VTR_T = VTR.reshape(1, n)
        AX = A.dot(VTR)
        XTAX = (VTR_T.dot(AX))
        XTX = (VTR_T.dot(VTR))
        lamda = XTAX[0] / XTX[0]
        lamda = round(lamda[0], 4)
        # C.append(lamda)

```



```

# > VTR chuwa C, >lamda2 chuan
print("TH1")
for j in range(len(B)):
    print("A{}Y = ".format(j))
    print(B[j])
    print("\n")
print('Sau {} lan lap'.format(m))
print('Gia tri rieng thu {} là: '.format(i+1), lamda)
print('Vector rieng tuong ung là: ')
#print(VTR)
VTRc= VTR
if i >= 1:
    VTRc = timVchuan(lamda, C[-1], VTR, D[-1], loca)
D.append(VTRc)
C.append(lamda)
print(Chuan_Hoa(VTRc))
loca, A = Chuyen_MT(A, VTR)
Y = np.random.rand(n, 1)
print('_' * 100)

elif TH == 3:
    print("TH3")
    for j in range(len(B)):
        print("A{}Y = ".format(j))
        print(B[j])
        print("\n")
    AmY = A.dot(B[-1])
    Am1Y = A.dot(AmY)
    Am2Y = A.dot(Am1Y)
    lamda_N = 0
    lamda_N1 = -1e18-1
    lamda_N2 = 1e18+1
    for j in range(n):
        if AmY[j][0] == 0:
            continue
        lamda_N1 = max((Am2Y[j][0]) / (AmY[j][0]), lamda_N1)
        lamda_N2 = min((Am2Y[j][0]) / (AmY[j][0]), lamda_N2)
        if abs(lamda_N2) > abs(lamda_N1):
            lamda_N = lamda_N2
        else:
            lamda_N = lamda_N1
    lamda_N = math.sqrt(lamda_N)
    if i >= 1:
        if round(lamda_N, 1) == round(C[-1], 1):
            lamda_N = - lamda_N
    VTR_N = Am1Y + lamda_N * AmY
    VTR_N = Chuan_Hoa(VTR_N)
    #TR_N = Chuan_Hoa(VTR_N)
    print("THUOC TH3")
    print('Sau {} lan lap'.format(m))
    print('Gia tri rieng la: ', round(lamda_N, 4))
    print('Vector tuong ung la: ')
    VTRc= VTR_N
    if i >= 1:
        VTRc = timVchuan(lamda_N, C[-1], VTR_N, D[-1], loca)
    D.append(VTRc)
    C.append(lamda_N)
    print(Chuan_Hoa(VTRc))
    loca, A = Chuyen_MT(A, VTR_N)
    Y = np.random.rand(n, 1)
    print('_' * 100)

elif TH == 4:
    print("TH4")
    for j in range(len(B)):
        if j == 0:
            print("Y =")
        else:
            print("A{}Y = ".format(j))
            print(B[j])
            print("\n")
    AmY = B[-1]
    Am1Y = A.dot(AmY)
    Am2Y = A.dot(Am1Y)
    a_1 = Am2Y[0][0]
    a_2 = Am2Y[1][0]

```

```

b_1 = Am1Y[0][0]
b_2 = Am1Y[1][0]
c_1 = AmY[0][0]
c_2 = AmY[1][0]
a = (c_1 * b_2 - b_1 * c_2)
b = (a_1 * c_2 - c_1 * a_2)
c = (b_1 * a_2 - a_1 * b_2)
# b= str(b)
# c= str(c)
print("TH4")
print('Ta thu duoc phuong trinh: Z^2 + {0}Z + {1} = 0 '.format(str(b), str(c)))
denta = b ** 2 - 4 * a * c
if denta >= 0:
    print('Phuong trinh co denta >= 0')
else:
    lamda_1 = complex(-b / (2 * a), -math.sqrt(abs(denta)) / (2 * a))
    lamda_2 = complex(-b / (2 * a), math.sqrt(abs(denta)) / (2 * a))
    VTR_1 = Am1Y - lamda_1 * AmY
    VTR_2 = Am1Y - lamda_2 * AmY
    print('Gia tri rieng la: {}'.format(lamda_1))
    print('Vector rieng tuong ung la: ')
    print(VTR_1)
    print('*' * 100)
    print('Gia tri rieng la: {}'.format(lamda_2))
    print('Vector rieng tuong ung la: ')
    print(VTR_2)
# A = Chuyen_MT(A, VTR_1)
# Y = np.random.rand(n, 1)
break

sys.stdout.close()
sys.stdout = temp

```

Ví dụ:

Sau đây sẽ là một số ma trận được chạy chương trình trên trình trên

TH1: Trường hợp trị riêng trội thực, đơn bội:

$$A = \begin{pmatrix} 8 & 6 & 10 & 10 \\ 9 & 1 & 10 & 5 \\ 1 & 3 & 1 & 8 \\ 10 & 6 & 10 & 1 \end{pmatrix}$$

Sau 6 lần lặp
 Giá trị riêng thu 1 là: 24.3481
 Vector riêng tương ứng là:
 $\begin{bmatrix} 1. \\ 0.7290593 \\ 0.40726386 \\ 0.79006226 \end{bmatrix}$

Sau 13 lần lặp
 Giá trị riêng thu 2 là: -7.5251
 Vector riêng tương ứng là:
 $\begin{bmatrix} 0.19559602 \\ -0.94904289 \\ 1. \\ -0.73424849 \end{bmatrix}$

Sau 6 lần lặp
 Giá trị riêng thu 3 là: -4.9851
 Vector riêng tương ứng là:
 $\begin{bmatrix} -0.01614028 \\ 0.07831356 \\ -0.52934341 \\ 1. \end{bmatrix}$

Sau 3 lần lặp
 Giá trị riêng thu 4 là: -0.8391
 Vector riêng tương ứng là:
 $\begin{bmatrix} -0.01614028 \\ 0.07831356 \\ -0.13986364 \\ 1. \end{bmatrix}$

TH2: Giá trị riêng trội đối nhau

$$A = \begin{bmatrix} 13 & 0 & 0 \\ 0 & -13 & 0 \\ 0 & 0 & -2 \end{bmatrix}$$

Sau 7 lần lặp
 Giá trị riêng là: 13.0
 Vector tương ứng là:
 $\begin{bmatrix} 1.0000000e+00 \\ 0.0000000e+00 \\ 1.3277383e-07 \end{bmatrix}$

Sau 7 lần lặp
 Giá trị riêng thu 2 là: -13.0
 Vector riêng tương ứng là:
 $\begin{bmatrix} -0.0000000e+00 \\ 1.0000000e+00 \\ 1.04143893e-05 \end{bmatrix}$

Sau 3 lần lặp
 Giá trị riêng thu 3 là: -2.0
 Vector riêng tương ứng là:
 $\begin{bmatrix} -0. \\ 0. \\ 1. \end{bmatrix}$

TH3 : Giá trị riêng phức:

$$A = \begin{pmatrix} -2 & 1 & 1 & 1 \\ -7 & -5 & -2 & -1 \\ 0 & -1 & -3 & -2 \\ -1 & 0 & -1 & 0 \end{pmatrix}$$

Ta thu được phương trình: $Z^2 + -6.628734553391537Z + -16.57183638347884 = 0$

Giá trị riêng là: $(-4+1.9999999999999991j)$

Vector riêng tương ứng là:

```
[ [ 0.87561986+0.24876027j]
  [-0.37809932-2.j          ]
  [-0.87561986-0.24876027j]
  [ 0.          +0.j          ] ]
```

Giá trị riêng là: $(-4-1.9999999999999991j)$

Vector riêng tương ứng là:

```
[ [ 0.87561986-0.24876027j]
  [-0.37809932+2.j          ]
  [-0.87561986+0.24876027j]
  [ 0.          +0.j          ] ]
```

PHẦN V: KẾT LUẬN

Ưu và nhược điểm của phương pháp lũy thừa:

a. Ưu điểm:

- i. Có thể tính trị riêng của các ma trận cấp lớn
- ii. Dễ hiểu, dễ nhớ
- iii. Linh hoạt, có thể thay đổi để tìm trị riêng nhỏ nhất

b. Nhược điểm:

- i. Tốc độ hội tụ không ổn định
- ii. Một số ma trận không thể áp dụng phương pháp này do khi xấp xỉ nhiều lần, tính hội tụ không còn đúng.
- iii. Dễ bị tràn số nếu không thu nhỏ vector

Phương pháp lũy thừa có ứng dụng quan trọng trong nhiều lĩnh vực khoa học kỹ thuật, đặc biệt là tin học.