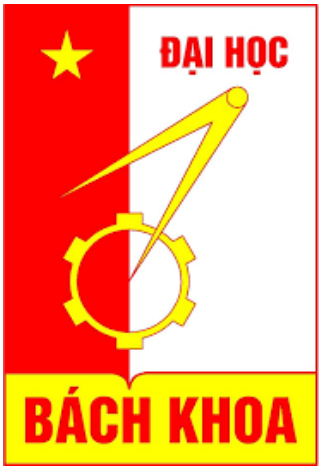
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



BÁO CÁO GIẢI TÍCH SỐ

CHỦ ĐỀ 24: TÌM GIÁ TRỊ RIÊNG TRỘI VÀ GIÁ TRỊ RIÊNG TRỘI TIẾP THEO

Nhóm sinh viên: Vũ Thị Kim Dung (MSSV: 20195959)

Lê Đức Tài (MSSV: 20195163)

Lớp: CTTN Toán Tin K64

Giảng viên hướng dẫn: TS. Hà Thị Ngọc Yến

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 lí 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. Phần này sẽ 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 chúng em xin 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, chúng em đưa ra thuật toán cùng một vài ví dụ điển hình để minh họa cho phương pháp, đồng thời cho bạn đọc nắm được phương pháp dễ dàng hơn và trình bày chương trình được viết trên ngôn ngữ Python.

Chúng em xin chân thành cảm ơn TS. Hà Thị Ngọc Yến 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 mình vẫn không tránh khỏi những sai sót và khiếm khuyết, rất mong nhận được sự góp ý của cô cùng các bạn trong lớp.

MỤC LỤC

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 $(K \in R; C)$. Số $\lambda \in K$ được gọi là giá trị riêng của ma trận A nếu tồn tại một véctơ $u \neq 0, u \in K^n$ 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 $det\ det\ (A - \lambda I) = 0$. 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**

- Giải phương trình $det\ det\ (A - \lambda I) = 0$ tìm các giá trị riêng. Ứng với mỗi giá trị riêng λ ta giải hệ phương trình tuyến tính thuần nhất $(A - \lambda I)u = 0$.
- 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
- Sử dụng phương pháp luỹ 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. Đều 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 luỹ 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 TRỘI

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 luỹ 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 thoả mãn điều kiện:
   $$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$
   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
   $$A = [2\ 0\ 0\ -2\ ]$$
   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 $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$ (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.awngffsss

$$Y = \sum_{i=1}^{n} C_i X_i = C_i X_i + C_2 X_2 + \cdots + C_n X_n \ với i$$

Ta thực hiện việc tính dãy:

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

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

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

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

… … …

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

Với $C_1 \neq 0$ ta có

$$A^mY = C_1\lambda_1^m X_1 + \sum_{i=2}^{n} C_i\lambda_i^m X_i = \lambda_1^m\left[C_1X_1 + \sum_{i=2}^{n} C_i\frac{\lambda_i^m}{\lambda_1^m}X_i\right]$$

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

$$A^mY \to C_1\lambda_1^m X_1 \ khi \ (m \to \infty)$$

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:

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

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 $A = [1\ 0\ 0\ 2\ ]$ ma trận này có 2 giá trị riêng $\lambda_1 = 2$ và $\lambda_2 = 1$ thoả mãn điều kiện (1). Ta thực hiện phương pháp luỹ 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:

- Sau mỗi lần tính được 1 vector A^mY, ta sẽ thu nhỏ vector sao cho hệ số lớn nhất bằng 1
- 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
- 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 toạ 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 tinh vector riêng theo công thức: $\lambda = \dfrac{X^tAX}{X^tX}$

Tốc độ hội tụ: ta sẽ xét 2 ví dụ sau:
Xét ma trận $A = [7\ 8\ 10\ 9\ ]$ 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^2Y	A^3Y	A^4Y
1	0.789	0.801	0.800	0.800

1	1	1	1	1

Tương tự với ma trận $B = [-4\ 10\ 7\ 5\ ]$ ta cung có bảng kết quả sau:

Y	BY	B^2Y	…	$B^{66}Y$	$B^{67}Y$	$B^{68}Y$
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 $m \to \infty$ thì $\left(\frac{\lambda_i}{\lambda_1}\right)^m \to 0$. Và khi đó, ta có

$$A^m Y \to C_1 \lambda_1^m X_1 \; khi \; (m \to \infty)$$

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 luỹ 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:

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

Khi đó, biểu thức (2) sẽ có dạng:

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

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. Đều 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:

$\{\lambda_1 = -\lambda_2 \ |\lambda_1| = |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$ (4)

Khi đó:

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

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

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

… …

$$A^{2k-1}Y = \lambda_1^{2k-1}(C_1X_1 - C_2X_2) + \sum_{i=3}^{n} C_i\lambda_i^{2k-1}X_i$$

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

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 $k \to \infty$ các vector liền nhau trong dãy luỹ thừa không có dấu hiệu hội tụ, thế nhưng các vector ở các bước luỹ thừa cùng chẵn hoặc cùng lẻ thì lại cùng nhau tạo thành dãy vector hội tụ. Và đồng thời:

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

$$\Rightarrow A^{2k}Y \approx \lambda_1^2A^{2k-2}Y \quad (5)$$

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:

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

$$\Leftrightarrow \{A^{2k}Y + \lambda_1A^{2k-1}Y \approx \lambda_1^{2k}2C_1X_1\ A^{2k}Y - \lambda_1A^{2k-1}Y \approx \lambda_1^{2k}2C_2X_2$$

$$\Leftrightarrow \{A\left(A^{2k}Y + \lambda_1A^{2k-1}Y\right) \approx \lambda_1^{2k}2C_1AX_1\ A\left(A^{2k}Y - \lambda_1A^{2k-1}Y\right) \approx \lambda_1^{2k}2C_2AX_2$$

$$\Leftrightarrow \{A\left(A^{2k}Y + \lambda_1A^{2k-1}Y\right) \approx \lambda_1(\lambda_1^{2k}2C_1X_1)\ A\left(A^{2k}Y - \lambda_1A^{2k-1}Y\right)$$
$$\approx \lambda_2(\lambda_1^{2k}2C_2X_2)$$

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

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:

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

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:

$$\{A^m Y \approx \lambda_1^m C_1 X_1 + \lambda_2^m C_2 X_2 \; A^{m+1} Y \approx \lambda_1^{m+1} C_1 X_1 + \lambda_2^{m+1} C_2 X_2 \; A^{m+2} Y \approx \lambda_1^{m+2} C_1 X_1 + \lambda_2^{m+2} C_2 X_2$$

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

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): $A^{m+2} Y + p A^{m+1} Y + q A^m Y = 0$ (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 toạ 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:

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

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:

$$\left| Z^2 \; Z \; 1 \; \left(A^{m+2} Y\right)_r \; \left(A^{m+1} Y\right)_r \; \left(A^m Y\right)_r \; \left(A^{m+2} Y\right)_s \; \left(A^{m+1} Y\right)_s \; \left(A^m Y\right)_s \right| = 0$$

Với các toạ độ tính được từ dãy vector, phương trình trên chính là phương trình (8). Giải ta được cặp ngiệm 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:

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

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

Nhân A vào các vế:

$$\Leftrightarrow \{A(A^mY - \lambda_1A^{m-1}Y) \approx \lambda_2(A^mY - \lambda_1A^{m-1}Y)\ A(A^mY - \lambda_2A^{m-1}Y) \approx \lambda_1(A^mY - \lambda_2A^{m-1}Y)$$

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.

Ví dụ: Cho ma trận

$A =$

có 2 trị riêng là $\lambda_1 = -4 + 2i$ và $\lambda_2 = -4 - 2i$. Quá trình thực hiện phương pháp luỹ thừa được thể hiện qua bảng sau:

Y	1	1	1	1
AY	-0.0667	1	0.4	0.1333
A^2Y	-0.3049	1	0.4512	0.0609
A^3Y	-0.5541	1	0.6464	0.0328
A^4Y	-0.9261	0.8131	1	0.0306
A^5Y	-0.9539	-0.0997	1	0.019
A^6Y	0.5481	1	-0,5697	-0.009
A^7Y	0.0878	1	-0.0945	-0.0028
A^8Y	-0.134	1	0.131	-0.001

Ta có thể thấy các vector (kế tiếp nhau hay có số bậc luỹ thừa cùng chẵn hoặc lẻ) không có dấu hiệu hội tụ, có thể kết luận đây là trường hợp trị riêng phức liên hợp.
Ta tiếp tục tính $A^9Y = (1.398; -4.322; -1.3906; 0)$ và $A^{10}Y = (-8.509; 14.607; 8.484; 0)$. Lý do ta phải tính thêm 2 vector vì ta đã thu nhỏ các vector A^6Y, A^7Y và ta cần 3 vector liên tiếp nhau nên phải tính tiếp 2 vector kết hợp với A^8Y tạo thành hệ (10) Chọn 2 toạ độ đầu của A^9Y và A^8Y ta lập được phương trình (8):

$$\begin{vmatrix} Z^2 & Z & 1 \\ -8.509 & 1.398 & -0.134 \\ 14.607 & -4.322 & 1 \end{vmatrix} = 0$$

Giải ta được $z \approx -4 \pm 2i$ đây chính là 2 giá trị riêng mà ta cần tìm. Việc tìm các vector riêng tương ứng là khá đơn giản với công thức đã tìm ra.

6. Tính trị riêng tiếp theo.
PHƯƠNG PHÁP XUỐNG THANG TÌM GIÁ TRỊ TRỘI 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 X_1

Chuẩn hóa X_1,ta được véc-tơ có thành phần thứ i bằng 1.Ta vẫn gọi véc-tơ mới là $X_1 = (x_1, x_2, \dots, 1, \dots, x_n)^t$

Xét ma trận θ phụ thuộc vào véc-tơ X_1 và chỉ số i nên kí hiệu là $\theta(X_1, i)$

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

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

$$\theta Z=\begin{vmatrix} z_1-x_1z_i \\ z_2-x_2z_i \\ \dots \\ 0 \\ \dots\dots \\ z_n-x_nz_i \end{vmatrix}=Z-z_iX$$

Do đó, với véc-tơ X_1 ta có $\theta X_1=0$

Nhận xét: $A_1=A-X_1a_i$ với $a_i=(a_{i1},a_{i2},\dots,a_{in})$ là hàng thứ i của ma trậnA

Ta sẽ chứng minh ma trận A_1 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 X_1 lần lượt là trị riêng và véc-tơ riêng tương ứng của A_1

Thật vậy, ta có:

$$A_1X_1=(\theta A)X_1=\theta(AX_1)=\theta\lambda_1X_1=\lambda_1(\theta X_1)=0$$

Do đó, 0 và X_1 lần lượt là trị riêng và véc-tơ riêng tương ứng của A_1

Tiếp theo, với k =2,3,…,n, gọi $x_i^{(k)}$ là phần tử thứ i của X_k. Khi đó ta có:

$$\theta X_k=X_k-x_i^{(k)}X_1$$

Suy ra:

$$A_1(\theta X_k) = (\theta A)(\theta X_k)$$
$$= (\theta A)\left(X_k - x_i^{(k)} X_1 \right)$$
$$= \theta A X_k - x_i^{(k)} \theta A X_1$$
$$= \theta A X_k$$
$$= \theta \lambda_k X_k$$
$$= \lambda_k (\theta X_k)$$

Nên suy ra $\lambda_k$và θX_k lần lượt là trị riêng và véc-tơ riêng của ma trận A_1 với mọi k=2,3,…,n

Như vậy, sau khi biến đổi ma trận A thành ma trận A_1, 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 θX_2 của ma trận A_1. Điều cần làm ở đây là tìm hệ số $x_i^{(2)}$

Ta có:

$$X_2 = \theta X_2 + x_i^{(2)} X_1$$

Suy ra

$$AX_2 = A\left(\theta X_2 + x_i^{(2)} X_1\right)$$
$$= A\theta X_2 + x_i^{(2)} A X_1$$
$$= A\theta X_2 + x_i^{(2)} \lambda_1 X_1$$

Vì X_2 là véc-tơ riêng ứng với λ_2 của ma trận A nên:

$$AX_2 = \lambda_2 X_2$$
$$= \lambda_2 \left(\theta X_2 + x_i^{(2)} X_1\right)$$

$$= \lambda_2 \theta X_2 + x_i^{(2)} \lambda_2 X_1$$

Suy ra:

$$A\theta X_2 + x_i^{(2)} \lambda_1 X_1 = \lambda_2 \theta X_2 + x_i^{(2)} \lambda_2 X_1$$

$$\Rightarrow \quad (A - \lambda_2 E)\theta X_2 = x_i^{(2)} (\lambda_2 - \lambda_1) X_1$$

+ Nếu $\lambda_2 = \lambda_1$ ta có $(A - \lambda_2 E)\theta X_2 = 0$, suy ra θX_2 chính là véc-tơ riêng tương ứng với λ_2 của ma trận A. Hay nói cách khác, trong trường hợp này $x_i^{(2)} = 0$

+ Nếu $\lambda_2 \neq \lambda_1$. Vì X_1 có phần tử thứ i bằng 1, ta có thể tính $x_i^{(2)}$bằng cách chia phần tử thứ i của $(A - \lambda_2 E)\theta X_2$ cho $(\lambda_2 - \lambda_1)$, tức là ta có:

$$x_i^{(2)} = \frac{\left((A - \lambda_2 E)\theta X_2\right)_i}{\lambda_2 - \lambda_1}$$

Tổng kết: Bằng phương pháp lũy thừa, ta tìm được giá trị riêng trội λ_1 và véc-tơ riêng tương ứng X_1 của ma trận A. Áp dụng phương pháp xuống thang, ta biến đổi ma trận A thành ma trận A_1. Lại áp dụng phương phương pháp lũy thừa đối với ma trận A_1 ta tìm được λ_2 là giá trị riêng trội của A_1, đồng thời là giá trị riêng trội tiếp theo của A. Từ véc-tơ θX_2 là véc-tơ riêng ứng với λ_2 của ma trận A_1, ta tìm được véc-tơ X_2 ứng với λ_2 của ma trận A. Cứ như thế tiếp tục, sau (n-1) lần xuống thang, ta tìm được đủ n trị riêng và n véc-tơ riêng tương ứng của ma trận A.

PHẦN III: THUẬT TOÁN VÀ CHƯƠNG TRÌNH

- **Thuật toán tổng quan:**

Tổng quan:

- B1: Nhập ma trận vuông cấp n và vector Y

- B2: Tính các vector và kiểm tra:

▪ Nếu các vector kề nhau hội tụ, đánh dấu là trường hợp 1.
▪ Nếu các vector có bậc luỹ thừa cùng chẵn hoặc lẻ hội tụ, đánh dấu là trường hợp 3.
▪ Nếu tính đến 200 lần mà ko có dấu hiệu thõa mãn hai trường hợp trên thì là trường hợp 4

- B3: Xử lý các trường hợp:

● TH1&2: Đưa ra trị riêng trội và vector tương ứng. Lặp/Xuống thang n lần để tính ma trận mới và tìm trị riêng tiếp theo.

● TH3&4: Đưa ra các trị riêng trội và vector riêng tương ứng. Kết thúc chương trình.

1. Gói xử lý dữ liệu

- **Gói Tính $A^m Y$:**

Lưu các cột $A^m Y$ liên tiếp thành một mảng B

Ngay từ đầu, véc-tơ Y được lưu tại vị trí B[0]

Đầu vào của gói này là ma trận A, mảng B đã có m cột : 0,1,2,…,m-1.

Gói này thực hiện việc tính và lưu cột thứ m ứng với các giá trị cột A^(m-1) Y dựa vào cột thứ m-1 và ma trận A.

- **Gói Chuẩn Hóa Vector:**
  Đầu vào là một vector M
  Tìm số có giá trị tuyết đối lớn nhất trong vector M là maxi
  Vector chuẩn hóa là M' = M/maxi

Gói Kiểm Tra

Đầu vào là mảng B và số m, n
Tính F = B[m] - B[n]
Tìm số có trị tuyệt đối lớn nhất trong F
Trả về trị tuyết đối của số đó
Đầu vào là ma trận A và vector Y:

Tạo một mảng B, lưu ví trí B[0] bằng vector Y

Tính 3 vector đầu tiên là AY, A^2Y, A^3Y lưu 3 vector này vào các vị trí tiếp theo của mảng B

Nếu Kiem_tra(B, m - 1, m - 2) <= E: là TH1, kết thúc vòng lặp

Nếu Kiem_tra(B, m - 1, m - 3) <= E: là TH3, kết thúc vòng lặp

Nếu lặp đến 200 lần mà ko có dấu hiệu hội tụ của hai trường hợp 1 và 3 thì là TH4, kết thúc vòng lặp

(E là sai số mong muốn giữa các vector)
Sau mỗi lần kiểm tra điều kiện xong thì ta tính A^mY tiếp theo và chuẩn hóa nó.
Lưu vector vừa chuẩn hóa vào vào các vị trí tiếp theo của mảng B
Gói này trả về số lần tính m, TH và mảng B

2 **Gói chuyển ma trận:**

Đầu vào là ma trận A và vector riêng của A
Tạo ma trận

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

Tính A' = $\theta\left(X^{(1)},i\right)$ A

Trả về A'

3 Tính toán các trường hợp:

Trường hợp 1:

- Tìm được vector riêng: X

  Tìm giá trị riêng: $\lambda = \frac{X^t AX}{X^t X}$

- Tính ma trận A mới để tìm trị riêng tiếp theo.

- Thay đổi vector Y

Trường hợp 3:

- Tính thêm $A^{m+1}Y$ và $A^{m+2}Y$

- Tìm tọa lớn nhất của $A^{m+1}Y$ và $A^{m+2}Y$

- Tính được trị riêng trái dấu là 2 căn của tỷ số giữa 2 toạ độ trên

- Tính vector riêng theo công thức.

- Tính ma trận A mới để tìm trị riêng tiếp theo.

Trường hợp 4:

- Tính thêm $A^{m+1}Y$ *và* $A^{m+2}Y$

- Thiết lập và giải phương trình (10) để tìm 2 giá trị riêng phức

- Tính vector riêng theo công thức đã tìm

- Kết thúc chương trình

4 Chương trình:

```
#Khai báo thư viện
import numpy as np
import  math
#Load file
f = open("input.txt", "r")
a = f.readline()
n =  int(a)
print('Ma trận cỡ: {} 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)
#print('Vector Y là: ',X)
# Y = np.random.rand(n, 1)
c = f.read()
c = c.split()
A = []
for i in c:
    A.append(float(i))
A = np.array(A).reshape(n, n)
print('Ma trận A là: ')
print(A)
print('_'*100)

E = 0.1
#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

#Xử lí
def Xu_li(A, Y):
    B = []
    B.append(Y)
    Z = np.zeros((n, 1))

    B.append(Luy_thua(A, B, 0))
    B.append(Luy_thua(A, B, 1))
    B.append(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 == 50:
            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
        m += 1
        B.append(Chuan_Hoa(Luy_thua(A, B, m - 1)))


    return m, TH, B
#Chuyển ma trận xuống thang
def Chuyen_MT(A, VTR):
    B = A
    V = VTR.reshape(n,)
    M = np.eye(n, dtype=float)
    maxi_1 = V.max()
    index = 0
    for i in range (n):
        if V[i] == maxi_1:
            index += i
            break
    E = M[: , index: index + 1] - VTR.reshape(n, 1)
    E[index][0] =0
    M[:, index: index + 1] = E
    A = M.dot(A)
    return A
for i in range(n):
    m, TH, B = Xu_li(A, Y)

    if TH == 1:
        VTR = B[-1]
        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)
        print('Sau {} lần lặp'.format(m))
        print('Giá trị riêng là: ', lamda)
        print('Vector riêng tương ứng là: ')
        print(VTR)
        A = Chuyen_MT(A, VTR)
        Y = np.random.rand(n,1)
        print(' ' * 100)
    elif TH == 3:
        AmY = A.dot(B[-1])
        Am1Y = A.dot(AmY)
        Am2Y = A.dot(Am1Y)
```

```
        lamda_N = max(Am2Y)/max(AmY)
        lamda_N = math.sqrt(lamda_N[0])

        VTR_N = Am1Y + lamda_N * AmY
        VTR_N = Chuan_Hoa(VTR_N)
        print('Sau {} lần lặp'.format(m))
        print('Giá trị riêng là: ', round(lamda_N, 4))
        print('Vector tương ứng là: ')
        print(VTR_N)
        A = Chuyen_MT(A, VTR_N)
        Y = np.random.rand(n, 1)
        print('_' * 100)

    elif TH == 4:
        M = B[-1]
        AmY = A.dot(M)
        Am1Y = A.dot(AmY)
        a_1 = Am1Y[0][0]
        a_2 = Am1Y[1][0]
        b_1 = AmY[0][0]
        b_2 = AmY[1][0]
        c_1 = M[0][0]
        c_2 = M[1][0]
        a = 1
        b = (a_1 * c_2 - c_1 * a_2) / (c_1 * b_2 - b_1 * c_2)
        c = (b_1 * a_2 - a_1 * b_2) / (c_1 * b_2 - b_1 * c_2)
        print('Phương trình tìm đc: Z^2 + {}Z + {} = 0 '.format(b, c))
        print()
        denta = b ** 2 - 4 * a * c
        if denta >= 0:
            print('Phương trình có 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('Giá trị riêng là: {}'.format(lamda_1))
            print('Vector riêng tương ứng là: ')
            print(VTR_1)
            print('*' * 100)
            print('Gía trị riêng là: {}'.format(lamda_2))
            print('Vector riêng tương ứng là: ')
            print(VTR_2)
        break
```

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

Trường hợp 1: Trường hợp trị riêng trội thực, đơn (bội một):

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

```
Giá trị riêng là:  24.3481
Vector riêng tương ứng là:
[[1.        ]
 [0.7290593 ]
 [0.40726386]
 [0.79006226]]
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  -4.6662
Vector riêng tương ứng là:
[[-0.        ]
 [ 1.        ]
 [-0.46923084]
 [ 0.02813595]]
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  -7.7985
Vector riêng tương ứng là:
[[-0.        ]
 [-0.        ]
 [-0.47564751]
 [ 1.        ]]
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  -0.839
Vector riêng tương ứng là:
[[-0.]
 [-0.]
 [ 1.]
 [-0.]]
```

TH2: Trường hợp trị riêng trội thực, bội:

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

```
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  13.0
Vector riêng tương ứng là:
[[1.00000000e+00]
 [1.32592775e-05]
 [1.00000000e+00]]
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  13.0
Vector riêng tương ứng là:
[[0.00000000e+00]
 [4.68897345e-06]
 [1.00000000e+00]]
--------------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  2.0
Vector riêng tương ứng là:
[[0.]
 [1.]
 [0.]]
--------------------------------------------------------------------------------------------------------
```

Trường hợp 3: Giá trị riêng trội đối nhau

$$A = \begin{pmatrix} 9 & 0 & 0 & 1 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & -9 & 0 \\ 0 & 0 & 0 & 6 \end{pmatrix}$$

*

```
----------------------------------------------------------------------------------------------------
Sau 7 lần lặp
Giá trị riêng là:  9.0
Vector tương ứng là:
[[1.]
 [0.]
 [0.]
 [0.]]
----------------------------------------------------------------------------------------------------
Sau 7 lần lặp
Giá trị riêng là:  -8.9911
Vector riêng tương ứng là:
[[-0.00000000e+00]
 [-1.21364868e-05]
 [ 1.00000000e+00]
 [-2.44251488e-02]]
----------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  6.0
Vector riêng tương ứng là:
[[0.00000000e+00]
 [6.27480006e-04]
 [0.00000000e+00]
 [1.00000000e+00]]
----------------------------------------------------------------------------------------------------
Sau 6 lần lặp
Giá trị riêng là:  2.0
Vector riêng tương ứng là:
[[0.]
 [1.]
 [0.]
 [0.]]
----------------------------------------------------------------------------------------------------
```

TH4 : Giá trị riêng phức:

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

```
--------------------------------------------------------------------------------------------------------
Phươnng trình tìm đc: Z^2 + 7.999999999999961Z + 19.99999999999972 = 0

Giá trị riêng là: (-3.9999999999999805-1.999999999999969j)
Vector riêng tương ứng là:
[[-8.13872519e-01+4.38959561e+00j]
 [-7.96531870e+00-6.01734065e+00j]
 [ 8.13872519e-01-4.38959561e+00j]
 [ 3.01980663e-14+3.79696274e-14j]]
********************************************************************************************************
Gía trị riêng là: (-3.9999999999999805+1.999999999999969j)
Vector riêng tương ứng là:
[[-8.13872519e-01-4.38959561e+00j]
 [-7.96531870e+00+6.01734065e+00j]
 [ 8.13872519e-01+4.38959561e+00j]
 [ 3.01980663e-14-3.79696274e-14j]]
```

Phần IV: Kết luận

2 Ưu và nhược điểm của phương pháp luỹ 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 luỹ thừa có ứng dụng quan trọng trong nhiều lĩnh vực khoa học kỹ thuật, đặc biệt là tin học.