TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC
School of Applied Mathematics and Informatics

Giải tích số

Phương pháp Cholesky giải hệ phương trình tuyến tính

Giảng viên hướng dẫn: TS. Hà Thị Ngọc Yến
Nhóm sinh viên thực hiện: Nhóm 14

Trần Thành Đạt	20185439
Dương Thị Phương Thảo	20185478
Nguyễn Trần Thức	20185483
Lê Thị Thành	20185477
Lê Hữu Đức Long	20185464

Mục lục

1 Mở đầu

1.1 Giới thiệu chung

Giải tích số (Numerical Analysis) là một môn khoa học nghiên cứu cách giải đúng và gần đúng, chủ yếu là giải số, các phương trình, các bài toán xấp xỉ hàm số và các bài toán tối ưu.

Ban đầu, toán học phát sinh do nhu cầu giải các bài toán thực tế như tính diện tích đất đai, quỹ đạo của các vì sao, đường đi của các con tàu buôn trên biển,... nên thuật ngữ Toán học đồng nghĩa với Toán học tính toán. Cùng với sự phát triển của toán học và các ngành khoa học khác, toán học được chia thành toán lý thuyết và toán ứng dụng. Tất cả những nhà toán học vĩ đại như Newton, Euler, Lagrange, Gauss,... đều có những công trình nền móng trong Giải tích số.

Trong nhiều năm trở lại đây, đặc biệt là những năm 80, Giải tích số đặc biệt phát triển cùng với sự phát triển của Tin học. Nếu toán lý thuyết chỉ quan tâm đến việc chứng minh tồn tại nghiệm, khảo sát dáng điệu và một số tính chất định tính của nghiệm thì toán tính đề xuất các thuật toán giải trên máy tính. Giải tích số đặc biệt quan tâm đến các vấn đề: thời gian máy, bộ nhớ cần sử dụng, tốc độ hội tụ và sự ổn định của thuật toán.

Xấp xỉ hàm số, giải gần đúng các phương trình và giải gần đúng các bài toán tối ưu là ba nhiệm vụ chính vô cùng quan trọng của Giải tích số. Trong bài báo cáo này, chúng em sẽ nghiên cứu một phương pháp được sử dụng để giải đúng nghiệm của hệ phương trình tuyến tính bậc nhất $Ax = b$, đó là phương pháp Cholesky.

Bài báo cáo chuẩn bị trong thời gian ngắn nên chúng em không tránh khỏi có những sai sót, rất mong được cô và các bạn góp ý. Chúng em xin chân thành cảm ơn cô Hà Thị Ngọc Yến đã giảng dạy và hướng dẫn chúng em hoàn thành báo cáo này.

1.2 Giới thiệu đề tài

Các phương pháp trực tiếp giải đúng hệ phương trình đại số tuyến tính thường chỉ được dùng đối với bài toán có kích cỡ nhỏ. Với các bài toán cỡ lớn, ta phải thực hiện một số lượng tính toán khổng lồ. Ví dụ như phương pháp Gauss - Jordan với hệ phương trình tuyến tính bậc nhất n ẩn cần thực hiện số lượng phép tính

- Nhân chia $N = \dfrac{n}{3}(n^2 + 3n - 1)$
- Cộng trừ $C = \dfrac{n}{6}(2n^2 + 3n - 5)$

Khi đó, số lượng phép tính tương đương $\dfrac{2n^3}{3}$

Khi thực hiện các phép tính, ta luôn phải thực hiện làm tròn số, tính toán với số gần đúng. Mà với hệ phương trình đại số tuyến tính, một chút thay đổi trong hệ số có thể dẫn tới nghiệm của hệ ra sai lệch lớn khi tính ra nghiệm. Ví dụ

$$\begin{cases} 400x_1 - 201x_2 = 200 \\ -800x_1 + 401x_2 = -200 \end{cases} \Rightarrow \begin{cases} x_1 = -100 \\ x_2 = -200 \end{cases}$$

Tuy nhiên:

$$\begin{cases} 401x_1 - 201x_1 = 200 \\ -800x_1 + 401x_2 = -200 \end{cases} \Rightarrow \begin{cases} x_1 = 40000 \\ x_2 = 79800 \end{cases}$$

Đây được gọi là sự bất ổn định của hệ phương trình đại số tuyến tính.

Do đó với các bài toán có kích cỡ lớn, ta sẽ dùng các phương pháp với lượng tính toán ít hơn, và cũng sẽ phải ổn định hơn. Đối với những bài toán như thế, ta sẽ sử dụng phương pháp Cholesky.

2 Phân rã ma trận (Matrix Decomposition)

2.1 Ý tưởng

Khi giải đúng hệ phương trình tuyến tính, ta dùng phép khử Gauss để đưa về ma trận tam giác, sau đó dùng quy trình nghịch để tìm nghiệm. Ý tưởng ở đây là đưa ma trận liên kết thành ma trận tam giác nhưng không thông qua phép khử, mà qua phép phân tích thành tích của các ma trận đặc biệt (ma trận tam giác, ma trận đường chéo, ma trận trực giao), ta gọi đó là các phương pháp của Matrix Decomposition.

Các phương pháp Matrix Decomposition thường gặp bao gồm:

- Phân tách LU: Phân tích ma trận liên kết thành tích của một ma trận tam giác trên và một ma trận tam giác dưới.
- Phân tách QR: Phân tích ma trận liên kết thành tích của một ma trận trực giao và một ma trận tam giác trên.
- Phân tách Cholesky: Phân tích ma trận liên kết thành tích của hai ma trận chuyển vị của nhau.
- Phân tách trị riêng (Eigendecomposition).
- Phân tách giá trị suy biến/ Giá trị kì dị (Singular Value Decomposition).

Trong báo cáo này, ta chỉ đi sâu vào hai phương pháp có độ phức tạp nhỏ nhất là phân tách LU và phân tách Cholesky.

2.2 Phương pháp tách LU

Phương pháp LU phát biểu rằng một ma trận A bất kỳ có thể phân rã thành:

$$A = LU$$

với L là ma trận tam giác dưới, U là ma trận tam giác trên:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \ddots & \vdots \\ & & & u_{nn} \end{bmatrix}$$

Tuy nhiên, để tìm các ẩn không hề dễ, bởi ta phải giải một hệ phương trình với $n^2 + n$ ẩn nhưng chỉ có n^2 phương trình (tương ứng với n^2 phần tử của ma trận tích A). Hệ này có thể vô nghiệm hoặc vô số nghiệm, tương ứng là có vô số cách phân tích A thành LU.

Để giảm tải số lượng ẩn và dễ tính toán, người ta cố định các giá trị, thường là các phần tử trên đường chéo chính của một trong hai ma trận thừa số, là 1. Khi đó, phân rã trở thành:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \times \begin{bmatrix} 1 & u_{12} & \cdots & u_{1n} \\ & 1 & \cdots & u_{2n} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}$$

Do đó, với phương trình

$$Ax = b \Leftrightarrow LUx = b$$

ta có thể viết lại thành (với $n = 3$):

$$\begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Lúc này, có thể dễ dàng tìm x dựa vào quy trình nghịch của phương pháp Gauss.

Sau khi phân tách, ta có hệ phương trình:

$$\begin{cases} l_{11} = a_{11} \\ l_{11}u_{1k} = a_{1k}, k = \overline{2,n} \\ l_{i1} = a_{i1}, i = \overline{2,n} \\ l_{i1}u_{1k} + l_{i2}u_{2k} + \cdots + l_{ii}u_{ik} = a_{ik}, k = \overline{2,n}, i = \overline{2,n} \end{cases}$$

Thoạt nhìn, đây là một hệ phi tuyến, song thực chất, ta có thể giải các giá trị sau dựa vào các giá trị vừa tìm được trước đó. Nghiệm của hệ được tổng quát hóa:

$$\begin{cases} l_{i1} = a_{i1}, i = \overline{1,n} \\ u_{1k} = \dfrac{a_{1k}}{a_{11}}, k = \overline{2,n} \end{cases}$$

$$\begin{cases} l_{it} = a_{it} - l_{i1}u_{1t} - l_{i2}u_{2t} - \cdots - l_{i,t-1}u_{t-1,t}, i \geq t \\ u_{tk} = a_{tk} - \dfrac{l_{t1}u_{1k} + l_{t2}u_{2k} + \cdots + l_{t,t-1}u_{t-1,k}}{l_{tt}}, k \geq t \end{cases}$$

Sau khi đã tách $A = LU$, ta lần lượt dùng quy trình nghịch của phương pháp Gauss để giải phương trình $Ly = b$ và $Ux = y$ và tìm được x.

2.3 Phép phân tách Cholesky

Phép tách Cholesky được đặt theo tên của **André Louis Cholesky**, một sỹ quan quân đội Pháp làm việc tại phòng Trắc địa.

Ý tưởng của Cholesky là phân tích ma trận liên kết A thành tích của hai ma trận U và $L = U^T$ là ma trận chuyển vị của U:

$$A = LU = U^T U$$

hay

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{12} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} u_{11} & & & \\ u_{12} & u_{22} & & \\ \vdots & \vdots & \ddots & \\ u_{1n} & u_{2n} & \cdots & u_{nn} \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \ddots & \vdots \\ & & & u_{nn} \end{bmatrix}$$

Dễ thấy, để tìm được các u_{ij}, ta cần giải một hệ phương trình gồm $\frac{n^2+n}{2}$ ẩn với n^2 phương trình (có phương trình trùng lặp do sự đối xứng của U^T và U). Điều này là có thể thực hiện được.

Sau khi đã mô hình phân tách, ta bắt đầu đi tìm các u_{ij}, các u_{ij} là nghiệm của hệ:

$$\begin{cases} u_{11}^2 = a_{11} \\ u_{11}u_{1k} = a_{1k}, k = \overline{2,n} \\ u_{12}^2 + u_{22}^2 = a_{22} \\ u_{12}u_{1k} + u_{22}u_{2k} = a_{2k}, k = \overline{3,n} \\ u_{1i}^2 + u_{2i}^2 + \ldots + u_{ii}^2 = a_{ii}, i = \overline{3,n} \\ u_{1i}u_{1k} + u_{2i}u_{2k} + \ldots + u_{ii}u_{ik} = a_{ik}, k = \overline{i+1,n}, i = \overline{3,n-1} \end{cases}$$

Giả sử tất cả các điều kiện xác định thỏa mãn, ta có nghiệm của hệ là:

$$\begin{cases} u_{11} = \sqrt{a_{11}} \\ u_{1k} = \frac{a_{1k}}{u_{11}}, k = \overline{2,n} \\ u_{22} = \sqrt{a_{22} - u_{12}^2} \\ u_{2k} = \dfrac{a_{2k} - u_{12}u_{1k}}{u_{22}}, k = \overline{3,n} \\ \ldots\ldots\ldots\ldots . \end{cases}$$

hay một cách tổng quát:

$$\begin{cases} u_{ii} = \sqrt{a_{ii} - \left(u_{1i}^2 + u_{2i}^2 + \cdots + u_{i-1,i}^2\right)} \\ u_{ik} = \dfrac{a_{ik} - (u_{1i}u_{1k} + u_{2i}u_{2k} + \cdots + u_{i-1,i}u_{i-1,k})}{u_{ii}} \end{cases}, k = \overline{i+1,n}, i = \overline{2,n-1}.$$

2.4 Đánh giá tổng quan

Dưới đây là bảng đánh giá độ phức tạp thuật toán, không gian chiếm, độ ổn định [1] của phép khử Gauss, phép phân tách LU, phép tách Cholesky:

	Khử Gauss	Phép tách LU	Phép tách Cholesky
Độ phức tạp	$\frac{2}{3}n^3 + O(n^2)$	$\frac{2}{3}n^3 + O(n^2)$	$\frac{1}{3}n^3 + O(n^2)$
Bộ nhớ chiếm dụng	1	2 (do phải lưu trữ L, U)	1 (chỉ cần lưu trữ U)
Độ ổn định	Ổn định	Không ổn định	Ổn định

[1] là độ đo sự thay đổi của nghiệm nếu các hệ số của ma trận liên kết thay đổi. Độ ổn định của hệ A được tính bằng cond(A)=$\|A\| \cdot \|A^{-1}\|$, khi cond(A)≈ 1, ta nói ma trận ổn định, cond(A) càng lớn, ma trận càng bất ổn, hay còn gọi là càng **gần suy biến**.

Phép phân hủy LU có độ phức tạp bằng phép khử Gauss, tuy nhiên với trường hợp ma trận b thay đổi, LU tỏ ra nhanh hơn Gauss vì không phải khử lại từ đầu!

3 Các vấn đề của phương pháp Cholesky

3.1 Cơ sở toán

Với những vấn đề từ việc giải hệ phương trình trên để tìm ra các u_{ii}, ta nhận thấy, để tồn tại phân tách Cholesky, những điều sau phải thỏa mãn:

- A phải là một ma trận đối xứng, bởi $A^T = (U^T U)^T = U^T U = A$.
- $u_{ii} > 0$. Mà $\det(A) = \det(U^T)\det(U) = \prod u_{ii}^2$, nên với mọi $u_{ii} > 0$, ta có $\det(A) > 0$, dẫn tới A phải là một ma trận không suy biến với định thức dương.
- $a_{ii} - (u_{1i}^2 + u_{2i}^2 + \cdots + u_{i-1,i}^2) > 0$, cho thấy điều kiện cần là các a_{ii} phải dương. Dễ thấy điều này sẽ đạt được nếu A là một ma trận xác định dương. Thật vậy, giả sử A là ma trận xác định dương, chọn $x^T = [0, \ldots, 0, 1, 0, \ldots, 0](x_i = 1)$. Dạng toàn phương $x^T Ax = e_i^T Ae_i$ viết lại bằng:

$$[0,\ldots,0,1,0,\ldots,0]\begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}\begin{bmatrix} 0 \\ \ldots \\ 1 \\ \ldots \\ 0 \end{bmatrix} = [0,\ldots,0,1,0,\ldots,0]\begin{bmatrix} a_{1i} \\ \ldots \\ a_{ii} \\ \ldots \\ a_{ni} \end{bmatrix} = a_{ii} > 0$$

Định lý 3.1 Phân tách Cholesky

Với ma trận A đối xứng xác định dương (SPD), tồn tại duy nhất một ma trận tam giác trên U với đường chéo chính dương, thỏa mãn $A = U^T U$.

Trước tiên ta có bổ đề:

Bổ đề 3.1. (Phần bù Schur). Giả định A là một ma trận vuông cấp n, đối xứng và xác định dương, được chia khối thành :

$$A = \begin{bmatrix} a_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

ta định nghĩa phần bù Schur với a_{11} là :

$$S := A_{22} - \frac{1}{a_{11}} A_{21} A_{12}.$$

Thì S cũng là một ma trận đối xứng xác định dương.

Chứng minh bổ đề:

- Dễ thấy S là một ma trận đối xứng, bởi lẽ A_{22} và $A_{21}A_{12} = A_{21}A_{21}^T$ đều là ma trận đối xứng.

- Để chứng minh S xác định dương, ta lấy $x \neq 0$ bất kỳ, và định nghĩa $y = -\dfrac{A_{21}^T x}{a_{11}}$, được:

$$x^T S x = \begin{bmatrix} y \\ x \end{bmatrix}^T \begin{bmatrix} a_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix} > 0$$

vì A xác định dương. Thật vậy, ta có:

$$\begin{aligned}
\begin{bmatrix} y \\ x \end{bmatrix}^T A \begin{bmatrix} y \\ x \end{bmatrix} &= \begin{bmatrix} -\frac{1}{a_{11}} A_{12} x & x^T \end{bmatrix} \begin{bmatrix} a_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} -\frac{1}{a_{11}} A_{12} x \\ x \end{bmatrix} \\
&= \begin{bmatrix} -\frac{1}{a_{11}} A_{12} x & x^T \end{bmatrix} \begin{bmatrix} -A_{12}x + A_{12}x \\ -\frac{1}{a_{11}} A_{21}A_{12}x + A_{22}x \end{bmatrix} \\
&= \begin{bmatrix} -\frac{1}{a_{11}} A_{12} x & x^T \end{bmatrix} \begin{bmatrix} 0 \\ -\frac{1}{a_{11}} A_{21}A_{12}x + A_{22}x \end{bmatrix} \\
&= -\frac{1}{a_{11}} x^T A_{21} A_{12} x + x^T A_{22} x \\
&= x^T S x
\end{aligned}$$

Vậy S là ma trận đối xứng xác định dương. ■

Chứng minh Phân tách Cholesky: Ta sẽ chứng minh bằng quy nạp.

- Với $n = 1$, dễ thấy kết quả đúng với ma trận 1×1: $A = a_{11}$. Trong trường hợp này, A là SPD, dẫn tới a_{11} là thực dương, và nhân tố Cholesky là $u_{11} = \sqrt{a_{11}}$, với tính duy nhất khi ta chọn u_{ii} dương.

- Giả sử kết quả đúng với ma trận SPD $A \in \mathcal{R}^{(n-1)\times(n-1)}$, ta sẽ chứng minh nó đúng với $A \in \mathcal{R}^{n\times n}$.
  Lấy $A \in \mathcal{R}^{n\times n}$ là SPD, viết lại A dưới dạng phân khối như trên **Bổ đề 3.1**, ta có ma trận S là SPD. Theo giả thiết quy nạp, S sẽ có phân tách Cholesky. Vì vậy, ta có thể viết lại $S = U_S^T U_S$, với U_S là một ma trận tam giác trên với kích thước $(n-1)\times(n-1)$. Ta định nghĩa:

$$U := \begin{bmatrix} \sqrt{a_{11}} & \frac{1}{\sqrt{a_{11}}} A_{12} \\ 0 & U_S \end{bmatrix}$$

  Định nghĩa của U là có nghĩa bởi $a_{11} > 0$. Và cũng dễ thấy U chính là ma trận tam giác trên. Mặt khác, do $A_{21}^T = A_{12}$ nên ta có:

$$\begin{aligned}
U^T U &= \begin{bmatrix} \sqrt{a_{11}} & 0 \\ \frac{1}{\sqrt{a_{11}}} A_{21} & U_S^T \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \frac{1}{\sqrt{a_{11}}} A_{12} \\ 0 & U_S \end{bmatrix} \\
&= \begin{bmatrix} a_{11} & A_{12} \\ A_{21} & \frac{1}{a_{11}} A_{21}A_{12} + U_S^T U_S \end{bmatrix} \\
&= \begin{bmatrix} a_{11} & A_{12} \\ A_{21} & \frac{1}{a_{11}} A_{21}A_{12} + S \end{bmatrix} \\
&= A
\end{aligned}$$

Đó chính là phân rã Cholesky của A. Kết thúc quy nạp. ■

Định lý 3.2

Với ma trận A là một ma trận phức khả nghịch, tồn tại phân tách Cholesky của A nếu và chỉ nếu A là ma trận *Hermitian* [a] xác định dương.

[a] ma trận vuông có các phần tử trên đường chéo là các số thực dương, các phần tử đối xứng qua đường chéo là các số phức liên hợp.

Việc chứng minh định lý này tương tự với cách chứng minh ma trận A là ma trận thực. Chỉ khác là thay vì dùng các chuyển vị, ta dùng các liên hợp.

Như vậy, ta đã biết rằng một ma trận A đối xứng xác định dương (xác định dương thì sẽ không suy biến), luôn có phân tách Cholesky. Tuy nhiên, trong thực tế, rất khó để một ma trận đạt được các điều trên. Ta sẽ phân chia các điều kiện của A để giải quyết từng vấn đề. Các vấn đề điều kiện của A :

- A có thể là ma trận không đối xứng?
- A có thể là ma trận không xác định?
- A có thể là ma trận suy biến?

3.2 Giải quyết các vấn đề của phương pháp Cholesky

3.2.1 Tính đối xứng của A

Như đã thấy, A bắt buộc phải là một ma trận đối xứng mới tồn tại ma trận U. Tuy nhiên, với A không đối xứng, ta có thể biến đổi A. Xét phương trình:

$$Ax = b \Leftrightarrow A^T Ax = A^T b$$

Nhân A^T vào hai vế là hoàn toàn hợp lý bởi lẽ A là ma trận khác ma trận 0.

Bằng cách gán $b := A^T b, A := A^T A$, dễ thấy ma trận A lúc này đã là ma trận đối xứng. Bài toán được giải quyết với ma trận liên kết mới A đối xứng. Ta chỉ cần kiểm tra tính xác định dương của ma trận.

3.2.2 Tính xác định dương

A là xác định dương, bởi nếu A không xác định dương, khi tính các u_{ii}, các phép khai căn số âm là vô nghĩa. Tuy nhiên, đó chỉ là về mặt toán học.

Thay vì tính trên trường số thực, ta có thể tính toán trên trường phức. Quy trình nghịch của Gauss sẽ tự nó khử đi các phần ảo để được nghiệm là một số thực.

Như vậy, các phần tử trên đường chéo chính của ma trận U sẽ tồn tại các số phức, nhưng không vấn đề, vì ta chỉ cần quan tâm tới kết quả. Vấn đề về tính xác định âm đã được giải quyết.

3.2.3 Tính suy biến

Như đã chỉ ra $\det(A) = \prod u_{ii}^2$. Vì vậy, nếu A suy biến, dẫn tới một u_{ii} nào đó bằng 0. Lúc đó phép tính của ta là không xác định.

Tổng kết lại, sau khi đã phân tích các điều kiện, ta thấy để áp dụng Cholesky, A chỉ cần là một ma trận vuông không suy biến!

4 Thiết kế thuật toán

4.1 Thuật toán cho phân rã Cholesky

4.1.1 Thuật toán bằng lời

INPUT: Ma trận A.

OUTPUT: Ma trận L (hoặc U)

Bước 1: $l_{11} = \sqrt{a_{11}}$

Bước 2: $j = 2, \ldots, n$, đặt $l_{j1} = \dfrac{a_{j1}}{l_{11}}$

Bước 3: $i = 2, \ldots, n-1$ làm **Bước 4** và **Bước 5**.

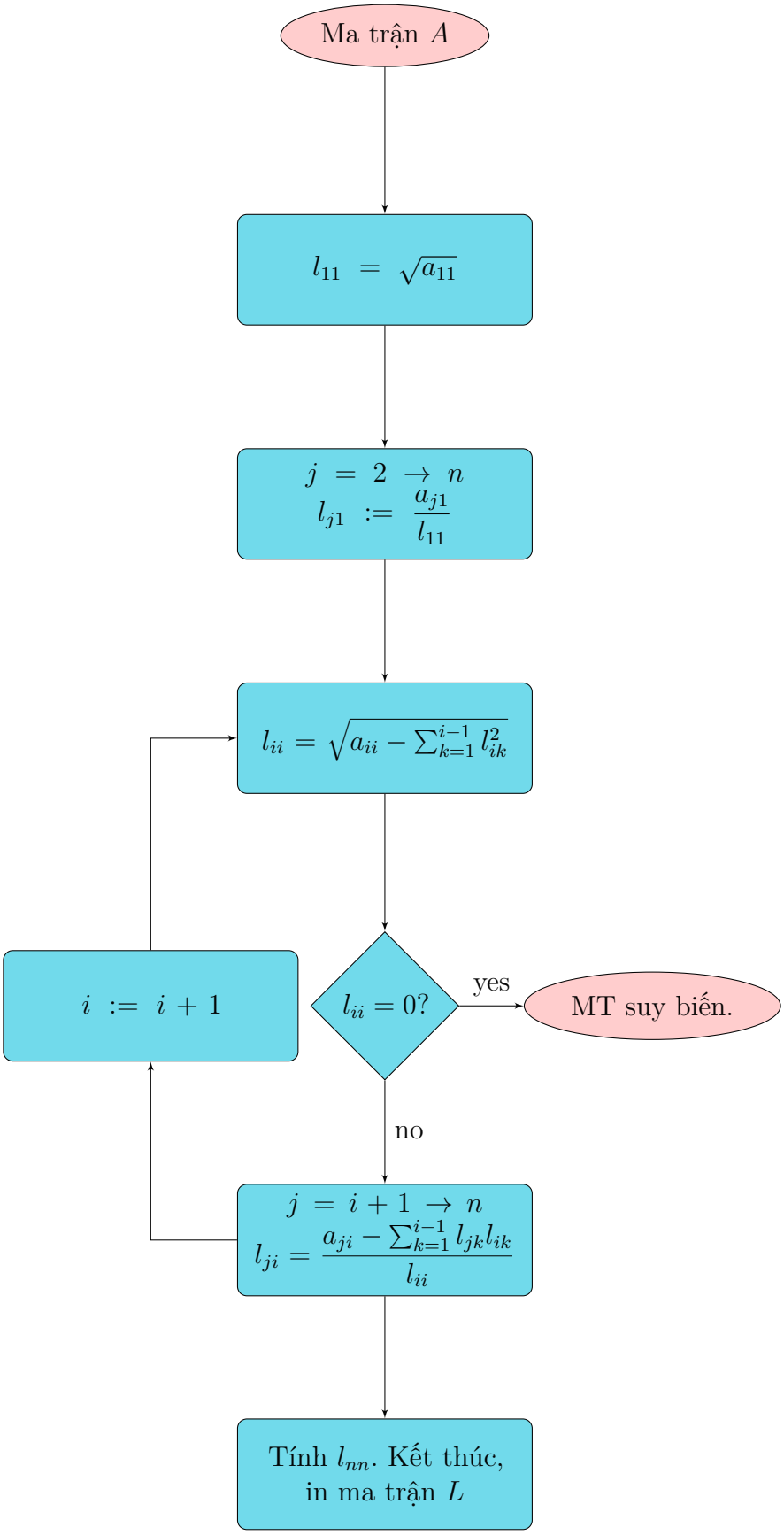
Bước 4: Tính $l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}$. Nếu $l_{ii} = 0$, dừng thuật toán, thông báo không thể giải.

Bước 5: Cho $j = i+1, \ldots, n$, tính $l_{ji} = \dfrac{a_{ji} - \sum_{k=1}^{i-1} l_{jk} l_{ik}}{l_{ii}}$

Bước 6: Tính $l_{nn} = \sqrt{a_{nn} - \sum_{k=1}^{n-1} l_{nk}^2}$

Bước 7: Output, kết thúc thuật toán.

4.1.2 Thuật toán bằng sơ đồ khối



4.2 Thuật toán cho giải hệ phương trình bằng phân rã Cholesky

4.2.1 Thuật toán bằng lời

INPUT: Ma trận A, b

OUTPUT: Nghiệm x của hệ.

Bước 1: Nhập ma trận A, b.

Bước 2: Kiểm tra ma trận A có vuông không. Nếu không vuông dừng thuật toán và thông báo không giải được.

Bước 3: Kiểm tra tính đối xứng. Nếu đối xứng, thực hiện bước 5. Nếu không đối xứng, làm bước 4.

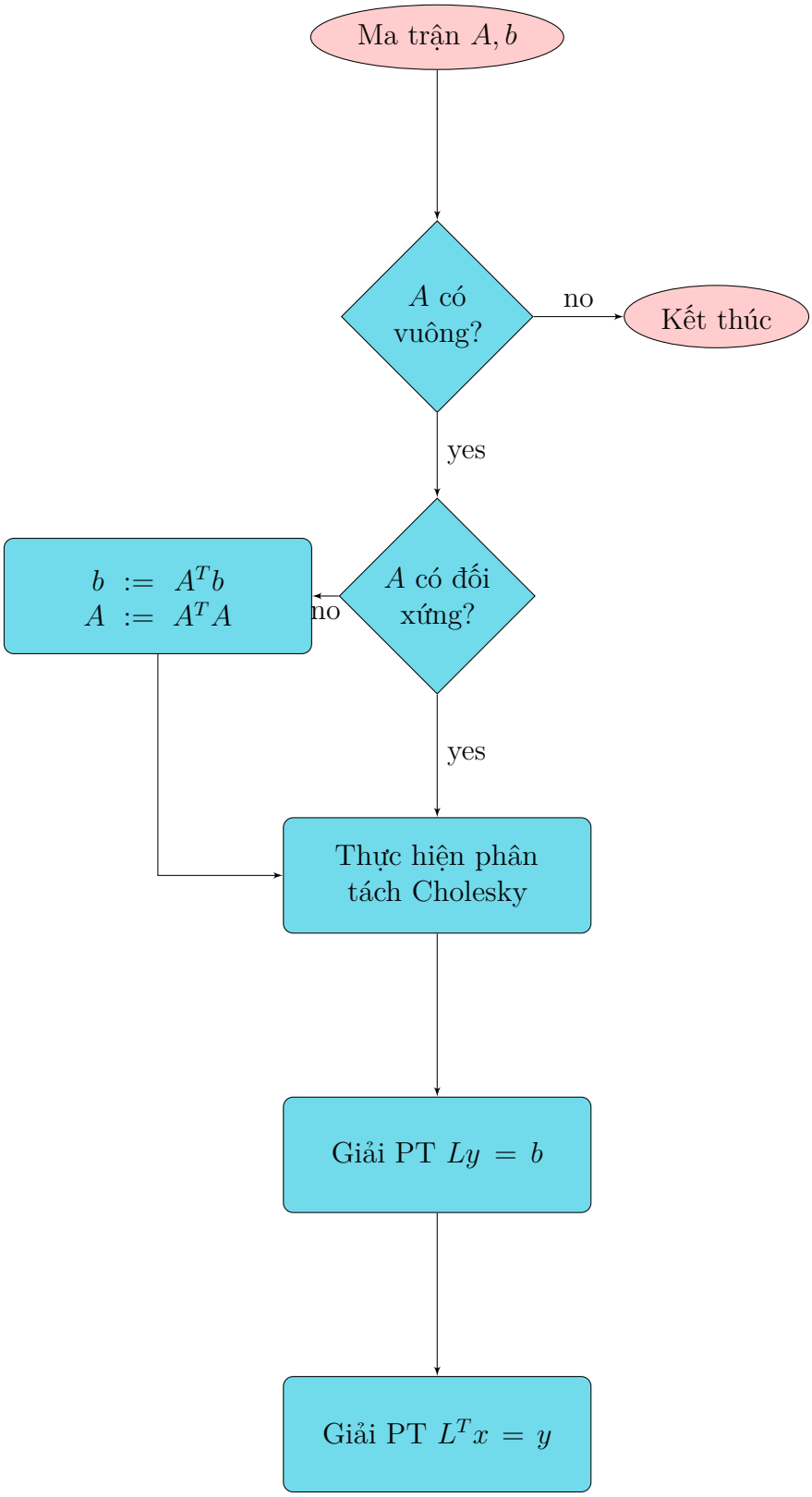
Bước 4: Gán $b := A^T b$ và $A := A^T A$.

Bước 5: Thực hiện phân tách Cholesky để tìm ma trận U hoặc L.

Bước 6: Thực hiện quy trình nghịch của PP Gauss giải phương trình $U^T y = b (Ly = b)$.

Bước 7: Thực hiện quy trình nghịch của PP Gauss giải phương trình $Ux = y (L^T x = y)$.

4.2.2 Thuật toán sử dụng sơ đồ khối



5 Độ phức tạp và tính ổn định của thuật toán

5.1 Độ phức tạp thuật toán

Độ phức tạp Cholesky	
Phép tính	Độ phức tạp
Phép cộnng	$\frac{n^3}{6} + O(n^2)$
Phép nhân	$\frac{n^3}{6} + O(n^2)$
Phép chia	$\frac{n^2}{2} + O(n)$
Phép khai căn	$O(n)$
Tổng	$\simeq (\frac{n^3}{3}) + O(n^2)$

Độ phức tạp chỉ bằng một nửa Gauss, đồng thời nếu phải giải nhiều hệ phương trình với b thay đổi và ma trận liên kết A giữ nguyên, phép tách U^TU được lưu lại và việc còn lại chỉ là thực hiện quy trình thế nghịch của Gauss chứ không phải thực hiện từ đầu.

5.2 Code chương trình

Kiểm tra ma trận có vuông không:

```
def IsVuong(matrix):
  m,n = np.shape(matrix)
  if (m==n):
    print('A la ma tran vuong.')
    return True
  else:
    print('A khong phai la ma tran vuong')
    return False
```

Kiểm tra tính đối xứng của A:

```
def IsSymmetry(matrix):
    n,_ = np.shape(matrix)
    for i in range(n):
        for j in range(i+1,n):
            if matrix[i,j] != matrix[j,i]:
                print('A khong doi xung.')
                return False
    print('A la ma tran doi xung.')
    return True
```

Tính phân rã Cholesky của ma trận

```
def Decomposition(matrix):
  a = np.array(matrix, dtype = complex)
```

```
  n,_ = np.shape(a)
  L = np.zeros((n,n), complex)
  for j in range(n):
    for i in range(j,n):
      if i == j:
        sumk = 0
        for k in range(j):
          sumk += L[j,k]**2
        L[i,j] = np.sqrt(a[i,j]-sumk)
      else:
        sumk = 0
        for k in range(j):
          sumk += L[i,k]*L[j,k]
        L[i,j] = (a[i,j]-sumk)/L[j,j]
  pro = 1.0
  for k in range(n):
    pro *= L[k,k]
  pro = round(pro)
  if pro == 0:
    return None
  print('Ma tran tam giac duoi :\n', L)
  return L
```

Giải hệ phương trình AX=B (A là ma trận tam giác trên/ dưới)

```
def Solve(matrixA, matrixB, L = True):
    y = np.zeros_like(matrixB, complex)
    x = np.zeros_like(matrixB, complex)
    n,_ = np.shape(matrixB)
    for i in range(n):
        sumk_y = sumk_x = 0
        j = n-1-i
        for k in range(i):
            sumk_y += matrixA[i,k] * y[k,0]
            sumk_x += matrixA[j,n-1-k] * x[n-1-k,0]
        y[i,0] = (matrixB[i,0]-sumk_y) / matrixA[i,i]
        x[j,0] = (matrixB[j,0]-sumk_x) / matrixA[j,j]
    if L == True:
        return y
    else:
        return x.real
```

Hàm giải HPT bằng phương pháp Cholesky.

```
def Cholesky(matrixA,matrixB):
  print('Input:\n A=', matrixA, '\n B=', matrixB)
```

```
  vuong = IsVuong(matrixA)
  if vuong == False:
    return print('Khong the giai he.')
  doixung = IsSymmetry(matrixA)
  if doixung == False:
    matrixB = matrixA.T @ matrixB
    matrixA = matrixA.T @ matrixA
    print('Ma tran sau khi bien doi ve doi xung
    \nA =', matrixA, '\nB=', matrixB)
  L = Decomposition(matrixA)
  if L is None:
    print('Ma tran suy bien, khong the giai')
    return None
  print('Nghiem he la:')
  y = Solve(L,matrixB,True)
  x = Solve(L.T,y,False)
  return x
```

6 Hệ thống ví dụ

Ví dụ 1: Giải HPT

$$\begin{pmatrix} 1 & 3 & -2 & 0 & -2 \\ 3 & 4 & -5 & 1 & -3 \\ -2 & -5 & 3 & -2 & 2 \\ 0 & 1 & -2 & 5 & 3 \\ -2 & -3 & 2 & 3 & 4 \end{pmatrix} X = \begin{pmatrix} 0,5 \\ 5,4 \\ 5 \\ 7,5 \\ 3,3 \end{pmatrix}$$

Xét ví dụ trên, ma trận liên kết A là một ma trận xác định không dương. Và chính vì thế, ma trận tam giác sau phân tách Cholesky là một ma trận phức. Tuy nhiên, qua quá trình nghịch của Gauss, nghiệm giải ra vẫn là một nghiệm thực:

```
Input:
 A= [[ 1.   3. -2.   0. -2.]
    [ 3.   4. -5.   1. -3.]
    [-2. -5.   3. -2.   2.]
    [ 0.   1. -2.   5.   3.]
    [-2. -3.   2.   3.   4.]]
 B= [[0.5]
    [5.4]
    [5. ]
    [7.5]
    [3.3]]
```

```
A la ma tran vuong.
A la ma tran doi xung.
Ma tran tam giac duoi :
[[ 1. +0.j    0. +0.j       0. +0.j       0.+0.j      0. +0.j     ]
 [ 3. +0.j    0. +2.2360j   0. +0.j       0.+0.j      0. +0.j     ]
 [-2. +0.j    0. -0.4472j   0. +0.8944j   0. +0.j     0. +0.j     ]
 [ 0. +0.j    0. -0.4472j   0. +2.0124j   3.0413+0.j 0. +0.j     ]
 [-2. +0.j    0. -1.3416j   0. +1.5652j   2.2193+0.j 0. +0.8219j]]
Nghiem he la:
[[-6.1]
 [-2.2]
 [-6.8]
 [-0.9]
 [ 0.2]]
```

Và có cùng kết quả khi sử dụng hàm giải phương trình có sẵn của Python

```
SD ham co san cua numpy:
 [[-6.1]
  [-2.2]
  [-6.8]
  [-0.9]
  [ 0.2]]
```

Ví dụ 2: Giải HPT

$$\begin{pmatrix} 1 & 1 & -1 \\ 2 & 1 & -1 \\ 4 & -1 & -3 \end{pmatrix} X = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$$

Ở ví dụ này, ma trận liên kết A là ma trận không đối xứng và xác định dương. Lúc này, thuật toán sẽ biến đổi ma trận A về ma trận đối xứng rồi tiếp tục thực hiện phân tách Cholesky:

```
Input:
 A= [[ 1.  1. -1.]
     [ 2.  1. -1.]
     [ 4. -1. -3.]]
 B= [[ 1.]
     [-2.]
     [ 1.]]
A la ma tran vuong.
A khong la ma tran doi xung.
Ma tran sau khi bien doi ve dang doi xung:
A = [[ 21.  -1. -15.]
     [ -1.   3.   1.]
```

```
      [-15.    1.   11.]]
B= [[ 1.]
    [-2.]
    [-2.]]
Ma tran tam giac duoi :
 [[ 4.58257569+0.j  0.         +0.j  0.         +0.j]
  [-0.21821789+0.j  1.71824939+0.j  0.         +0.j]
  [-3.27326835+0.j  0.1662822 +0.j  0.50800051+0.j]]
Nghiem he la:
[[-3.  ]
 [-0.25]
 [-4.25]]
```

và kết quả trả giống nghiệm của khi sử dụng hàm sẵn:

```
Su dung ham co san cua Numpy:
 [[-3.  ]
  [-0.25]
  [-4.25]]
```

Ví dụ 3: Giải HPT

$$\begin{pmatrix} 1 & 1 & -1 \\ 1 & 2 & 0 \\ -1 & 0 & 4 \end{pmatrix} X = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Ở ví dụ này, ma trận liên kết là một ma trận đối xứng và xác định dương.

```
Input:
 A= [[ 1.   1. -1.]
     [ 1.   2.  0.]
     [-1.   0.  4.]]
 B= [[1.]
     [2.]
     [3.]]
A la ma tran vuong.
A la ma tran doi xung.
Ma tran tam giac duoi:
 [[ 1.         +0.j  0.         +0.j  0.         +0.j]
  [ 1.         +0.j  1.         +0.j  0.         +0.j]
  [-1.         +0.j  1.         +0.j  1.41421356+0.j]]
Nghiem he la:
[[ 3. ]
 [-0.5]
 [ 1.5]]
```

```
Su dung ham co san cua Numpy:
 [[ 3. ]
  [-0.5]
  [ 1.5]]
```

Ví dụ 4: Giải HPT

$$\begin{pmatrix} 2 & 3 & 5 \\ 4 & 6 & 10 \\ 7 & 6 & 5 \end{pmatrix} X = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Ví dụ 4 này, ma trận A là ma trận suy biến. Khi ma trận là suy biến, bài toán không thể giải được theo phương pháp Cholesky!

```
Input:
 A= [[ 2.   3.   5.]
     [ 4.   6.  10.]
     [ 7.   6.   5.]]
 B= [[1.]
     [2.]
     [3.]]
A la ma tran vuong.
A khong la ma tran doi xung.
Ma tran sau khi bien doi ve doi xung:
A = [[ 69.   72.   85.]
     [ 72.   81.  105.]
     [ 85.  105.  150.]]
B= [[31.]
   [33.]
   [40.]]
Ma tran suy bien, khong the giai.
```

7 Ứng dụng của phương pháp Cholesky

7.1 Giải hệ phương trình

Như ta đã thấy, việc giải hệ phương trình tuyến tính bằng phương pháp Cholesky là nhanh hơn đối với các phương pháp tiền nhiệm trước đó. Trong thực tế, đối với những bài toán có ma trận đặc thù (ví dụ đối xứng xác định dương), hoặc các bài toán có ma trận liên kết không đổi, việc giải bằng phương pháp phân tách Cholesky là tối ưu.

7.2 Bình phương tối thiểu

Xét phương trình $Ax = b$, trong đó $A \in \mathbb{C}^{m\times n}$ với $m > n$. Ta nghiên cứu bài toán: tìm $\hat{x} \in \mathbb{C}^n$ sao cho:

$$\|b - A\hat{x}\|_2 = \min_{x\in\mathbb{C}^n} \|b - Ax\|_2$$

Đây chính là bài toán bình phương tối thiểu.
Trước khi giải bài toán, ta có các quan sát sau:

- $\hat{b} = A\hat{x}$ nằm trong không gian cột của A ($\mathcal{C}(\mathcal{A})$.
- $\hat{b}$ là phần tử của $\mathcal{C}(A)$ gần nhất với b, làm cho nó trở thành phép chiếu trực giao của b vào $\mathcal{C}(A)$
- Như vậy, phần dư $b-\hat{b}$ là trực giao với $\mathcal{C}(A)$. Ta tách $b = z+w$, trong đó $z \in \mathcal{C}(A), w \in \mathcal{C}(A)^\perp$
  Vì $w \in \mathcal{C}(A)^\perp \Rightarrow w \in \mathcal{N}(A^H)$ (trong đó $\mathcal{N}(A)$ là không gian vô hiệu của A (hay còn gọi là hạt nhân))
  Dẫn tới $A^H w = 0 \Leftrightarrow A^H(b-z) = 0 \Leftrightarrow A^H(b - A\hat{x}) = 0$
- Đồng nghĩa với $A^H A\hat{x} = A^H b$, đây được gọi là *normal equations*.
- Nếu A có các cột độc lập tuyến tính, thì $rank(A) = n, \mathcal{N}(A) = \emptyset$ và $A^H A$ không suy biến, khi đó:

$$\hat{x} = \left(A^H A\right)^{-1} A^H b$$

Như vậy, để giải bài toán bình phương tối thiểu bằng phương pháp Cholesky, ta cần:

- Đặt $B = A^H A$. Tốn mn^2 vòng lặp
- Sử dụng Cholesky để tách $B = LL^H$. Tốn $\dfrac{n^3}{3}$ vòng lặp
- Tính $y - A^H b$. Tốn $2mn$ vòng lặp.
- Giải hệ $Lz = y$. Tốn n^2 vòng lặp.
- Giải hệ $L^H\hat{x} = z$. Tốn n^2 vòng lặp.

7.3 Mô phỏng Monte Carlo

Ma trận Cholesky biến đổi một vectơ của các biến ngẫu nhiên có phân phối chuẩn không tương quan (tức là độc lập) thành một vectơ của các biến ngẫu nhiên có tương quan (tức là phụ thuộc) có phân phối chuẩn. Những biến thể ngẫu nhiên có tương quan hiện nay có thể được sử dụng trong mô phỏng Monte Carlo (lớp thuật toán mô phỏng hướng dữ liệu từ đầu vào thường là ngẫu nhiên và đầu ra là một kết quả chưa được xác định trước)

Một thí dụ nho nhỏ là chúng ta tính diện tích một hình tròn. Các thuật toán tất định sẽ tính bằng cách lấy $S = \pi R^2$ với R là bán kính đường tròn. Nhưng mô phỏng Monte Carlo giả sử rằng chưa biết công thức đó, nó mô phỏng bằng cách giả sử có một đường tròn và một

hình vuông ngoại tiếp đường tròn đó. Mô phỏng sẽ tạo ra ngẫu nhiên một điểm thuộc hình vuông ngoại tiếp đường tròn, và mỗi lần ghi lại xem điểm đó có thuộc vào đường tròn hay không. Sau vô cùng lớn lần tạo lập, tỷ lệ diện tích hình tròn trên diện tích hình vuông được tính bằng $\frac{\text{số điểm tạo được trong hình tròn}}{\text{tổng số điểm tạo được trong hình vuông}}$.

Tưởng tượng ta muốn tạo ra nhiều biến ngẫu nhiên phân phối chuẩn có tương quan, nhưng lại không muốn xử lý một mô hình đa biến phân phối chuẩn cồng kênh. Phân tách Cholesky giúp ta làm điều đó.

Giả sử có 3 biến ngẫu nhiên chuẩn $X_i \sim \mathbb{N}(0, 1)$, ba biến này có:

Ma trận hiệp phương sai:

$$\Sigma = \begin{bmatrix} 10.0 & -2.0 & 2.0 \\ -2.0 & 20.0 & 0.5 \\ 2.0 & 0.5 & 0.5 \end{bmatrix}$$

Ma trận hệ số tương quan:

$$\mathbb{P}_\rho = \begin{bmatrix} 1.00 & -0.14 & 0.89 \\ -0.14 & 1.00 & 0.16 \\ 0.89 & 0.16 & 1.00 \end{bmatrix}$$

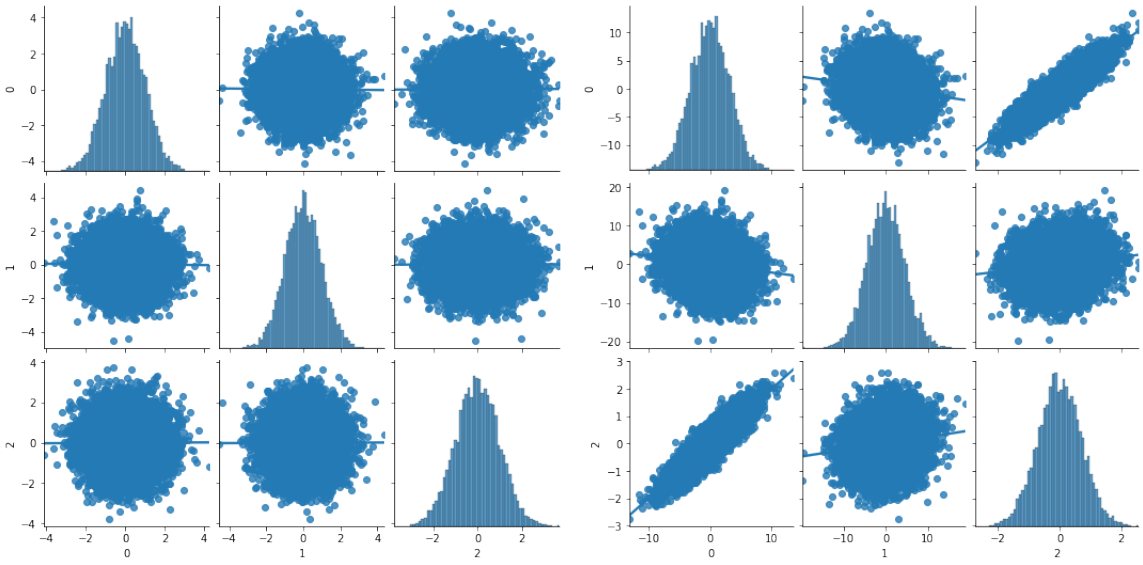
Độ lêch chuẩn:

$$\sigma = \begin{bmatrix} 3.16 \\ 4.47 \\ 0.71 \end{bmatrix}$$

Ta tìm phân tích Cholesky của ma trận hiệp phương sai và nhân nó với ma trận của các biến ngẫu nhiên không tương quan để tạo ra các biến tương quan:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
x_uncor = np.random.normal(0, 1, (3, 10000))
cov = np.array([[ 10.0, -2.00,  2.00],
                [-2.00, 20.00,  0.50],
                [ 2.00,  0.50,  0.50]])
L = np.linalg.cholesky(cov)
x_cor = np.dot(L, x_uncor)
corr_simulated = pd.DataFrame(x_cor).T.corr()
std_ = np.sqrt(np.diag(cov))
corr_empirical = cov / np.outer(std_, std_)
x_uncor = pd.DataFrame(x_uncor.T)
x_cor = pd.DataFrame(x_cor.T)
sns.pairplot(x_uncor, kind="reg")
sns.pairplot(x_cor, kind="reg")
```

Kết quả: Các biến từ không tương quan (được tạo random bằng hàm với kỳ vọng bằng 0 và phương sai bằng 1) đã được đưa về tương quan bằng một phân tách Cholesky đối với một ma trận hiệp phương sai.



Đây là một phương pháp tuyệt vời, nhất là khi trường phái khoa học hướng dữ liệu đang dần phát triển hơn trong thời đại Big data như hiện nay.

8 Kết luận

Qua bài báo cáo vừa rồi, nhóm chúng em đã sơ lược qua về ý tưởng, cơ sở Toán học, các vấn đề phát sinh và giải thuật của phép tách Cholesky. Bên cạnh đó, thuật toán, ví dụ và các ứng dụng khác của phương pháp cũng được đề cập để làm rõ cho sự *lợi hại* của phương pháp tách này trong khoa học hiện đại. Phương pháp Cholesky thực chất vẫn còn rất nhiều điều hay cần được khám phá, song trong nội dung môn Giải tích số thì nhóm đã giới hạn qua những kiến thức tổng quan nhất của phương pháp Cholesky. Mong cô và các bạn đọc cho ý kiến để bài báo cáo được hoàn thiện thêm về chất và lượng. Nhóm xin chân thành cảm ơn!

Tài liệu

[1] Gene H. Golub and Charles F. Van Loan. 1996. *Matrix computations* (3rd ed.). Johns Hopkins University Press, USA.

[2] Prof. L. Vandenberghe, UCLA. 2020. *Applied Numerical Computing*, ECE133A.

[3] Hà Thị Ngọc Yến. 2020. *Giải tích số MI3041*, Viện Toán Ứng dụng và Tin học, Đại học Bách Khoa Hà Nội.

[4] Lê Trọng Vinh. 2007. *Giải tích số*. Nhà Xuất bản Khoa học Kỹ thuật, VN.

[5] Phạm Kỳ Anh. 1996. *Giải tích số*. Nhà xuất bản Đại học Quốc gia Hà Nội, VN.

[6] Richard L. Burden, J. Douglas Faires. 2011. *Numerical Analysis*, 9th Edition, Brooks/Cole, Cengage Learning.

[7] Timothy Sauer. 2018. *Numerical Analysis*, 3rd Edition, Pearson's r, USA.

[8] McSweeney, Thomas. 2017. *Modified Cholesky Decomposition and Applications*, Manchester Institute for Mathematical Sciences School of Mathematics, The University of Manchester.