**Mục lục**

[I. Ý tưởng phương pháp 2](#_Toc104529263)

[II. Lý thuyết ma trận 2](#_Toc104529264)

[1. Các phép biến đổi ma trận 2](#_Toc104529265)

[2. Ma trận bậc thang 2](#_Toc104529266)

[3. Hạng của ma trận 3](#_Toc104529267)

[III. Cách chọn phần tử khử 3](#_Toc104529268)

[IV. Cách xét nghiệm 3](#_Toc104529269)

[IV. Công thức khử 3](#_Toc104529270)

[V. Thuật toán 4](#_Toc104529271)

[1. Thuật toán tổng quát 4](#_Toc104529272)

[a. Bằng chữ: 4](#_Toc104529273)

[2. Thuật toán chi tiết 6](#_Toc104529274)

[a. Tìm phần tử khử 6](#_Toc104529275)

[b. Lưu vị trí 8](#_Toc104529276)

[c. quá trình thuận 10](#_Toc104529277)

[d. Chuẩn hóa hệ số 12](#_Toc104529278)

[V. đánh giá thuật toán 16](#_Toc104529279)

[1. Ưu điểm. 16](#_Toc104529280)

[2. Nhược điểm 16](#_Toc104529281)

# I. Ý tưởng phương pháp

- hạn chế sai số tính toán khi gặp phép chia cho số gần 0 bằng cách chọn phần tử khử thích hợp

- Quy trình thuận: dùng phép khử để đưa tất cả các phần tử (trừ phần tử khử) trên cột chứa phần tử khử về 0.

- Quy trình nghịch: đánh giá rank của ma trận hệ số và ma trận bổ sung rồi đưa ra kết luận về nghiệm của phương trình

# II. Lý thuyết ma trận

## 1. Các phép biến đổi ma trận

- đổi chỗ hai hàng

- Nhân một số khác 0 bất kì vào một hàng

- Nhân một hàng với một số rồi cộng với hàng khác

## 2. Ma trận bậc thang

- Một ma trận được gọi là ma trận bậc thang nếu thỏa mãn

+ nếu có hàng 0 (mọi phần tử trên hàng bằng 0) thì nó phải nằm dưới các hàng có phần tử khác 0

+ với hai hàng khác 0, phần tử khác 0 đầu tiên kể từ trái của hàng dưới nằm ở bên phải cột chứa phần tử khác 0 đầu tiên của hàng trên

## 3. Hạng của ma trận

- Hạng của ma trận là số hàng khác 0 của ma trận đó

# III. Cách chọn phần tử khử

- Ưu tiên 1: chọn sao cho | = 1. Để tránh phép chia

- Ưu tiên 2: chọn sao cho | =

Chú ý: phần tử khử thứ k được chọn từ các hàng với cột không chứa các phần tử khử đã chọn trước đó

# IV. Cách xét nghiệm

- gọi rank(A), rank(B) là hạng của ma trận A và ma trận bổ sung B:

+ rank(A) rank(B): hệ vô nghiệm

+ rank(A) rank(B) n: hệ có nghiệm duy nhất

+ rank(A) rank(B) < n: hệ có vô số nghiệm

# IV. Công thức khử

Ta có ma trân aij với kích thước là m x n, là phần tử khử thứ k với p là hàng giải và q là cột giải

- Đặt

- Lấy hàng p nhân với mỗi rồi lấy lần lượt hệ số ở hàng i = 1, 2, … trừ đi các phần tử tương ứng trong cột của hàng p ta thu được công thức khử

# V. Thuật toán

## 1. Thuật toán tổng quát

## a. Bằng chữ:

input: m, n, A, B

trong đó: m, n là số hàng và số cột của ma trận A

output: Kết quả nghiệm của hệ phương trình

Bước 1: Nhập input

Bước 2: Khởi tạo i = 1 để tính số lần chạy

Bước 3: Nếu i <= min(m, n) thì chuyển sang bước 4, ngược lại thì chuyển sang bước 9

Bước 4: Tìm phần tử khử apq của ma trận hệ số

Bước 5: Nếu apq 0 thì chuyển sang bước 6, ngược lại thì chuyển sang bước 8

Bước 6: Lưu vị trí của phần tử khử

Bước 7: Biến đổi ma trận để khử các phần tử trên cột q (trừ hàng p) thành 0

Bước 8: Tăng i++ rồi quay lại bước 3

Bước 9: Chuẩn hóa hệ số

Bước 10: Kiểm tra rank của ma trận và đánh giá nghiệm

Bước 11: in ra output

b. bằng sơ đồ khối

Diagram

Description automatically generated

## 2. Thuật toán chi tiết

### a. Tìm phần tử khử

\* Bằng chữ:

input: A

output: ptk

trong đó: ptk là phần tử khử.

Bước 1: nhập input

Bước 2: tạo biến i = 1 để đo số lần lặp

Bước 3: nếu i <= m thì chuyển sang bước 4, ngược lại thì chuyển sang bước 11

Bước 4: kiểm tra trùng vị trí, nếu trùng thì chuyển sang bước 9, ngược lại thì chuyển sang bước 5

Bước 5: Nếu h[i] có phần tử 1 hoặc -1 thì chuyển sang bước 10, ngược lại chuyển sang bước 6

trong đó: h[i] là tập các hệ số trong hàng i

Bước 6: gán MaxH = MaxH[i, n]

Trong đó: MaxH[i, n] là giá trị cho phần tử có trị tuyệt đối lớn nhất trong hàng i từ cột 1 đến cột n

Bước 7: Nếu MaxH > ptk thì chuyển sang bước 8, ngược lại thì chuyển sang bước 9

Bước 8: gán ptk = MaxH

Bước 9: tăng i++ và quay lại bước 3

Bước 10: gán ptk = 1

Bước 11: in ra output

\* Bằng sơ đồ khối

Diagram

Description automatically generated

### b. Lưu vị trí

\* Bằng chữ

Input: ptk, k, A[][]

Trong đó: k là phần tử lần thứ k

Output: Lh[k], Lc[k]

Trong đó: Lh[] là mảng lưu vị trí hàng đã chọn, Lc[] là mảng lưu vị trí của cột đã chọn

Bước 1: nhập input

Bước 2: khởi tạo i = 1 để đo số lần lặp

Bước 3: khởi tạo j = 1 để đo số lần lặp

Bước 4: nếu j <= n thì chuyển sang bước 5, ngược lại chuyển sang bước 7

Bước 5: nếu A[i][j] = ptk thì chuyển sang bước 8, ngược lại thì chuyển sang bước 6

Bước 6: tăng j++ và quay lại bước 4

Bước 7: tăng i++ và quay lại bước 3

Bước 8: in ra output

\* Bằng sơ đồ khối

Diagram

Description automatically generated

### c. quá trình thuận

\* Bằng chữ

Input: Lh[k], Lc[k], A

Output: A mới

Bước 1: nhập input

Bước 2: gán p = Lh[k], q = Lh[k]

Bước 3: khởi tạo i = 1 để đếm số lần lặp

Bước 4: nếu i <= m thì chuyển sang bước 5, ngược lại thì chuyển sang bước 12

Bước 5: nếu i p thì chuyển sang bước 6, ngược lại thì chuyển sang bước 11

Bước 6: gán alpha =

Bước 7: tạo j = 1 để đếm số lần lặp

Bước 8: nếu j <= n + 1 thì chuyển sang bước 9, ngược lại thì chuyển sang bước 11

Bước 9: tính

Bước 10: tăng j ++ rồi quay lại bước 8

Bước 11: tăng i ++ rồi quay lại bước 4

Bước 12: in ra output

\* Bằng sơ đồ khối

Diagram

Description automatically generated

### d. Chuẩn hóa hệ số

Input: A[

Output: A sau khi chuẩn hóa

Bước 1: nhập input

Bước 2: khởi tạo i = 1 để đo số lần lặp

Bước 3: nếu i <= s thì chuyển sang bước 4, ngược lại thì chuyển sang bước 9

trong đó: s là số phần tử của mảng Lh[]

Bước 4: Khởi tạo biến j = 1 để đo số lần lặp

Bước 5: nếu j <= n + 1 thì chuyển sang bước 6, ngược lại thì chuyển sang bước 8

Bước 6: tính A[Lh[i]][j] =

Bước 7: tăng k ++ rồi quay lại bước 5

Bước 8: tăng u ++ rồi quay lại bước 3

Bước 9: in ra output

\* Bằng sơ đồ khối

Diagram

Description automatically generated

6. Kiểm tra rank ma trận

Input: A[][]

Output: rankA, rankB

Bước 1: nhập input

Bước 2: khởi tạo rankA = 0, rankB = 0

Bước 3: khởi tạo i = 1 để đo số vòng lặp

Bước 4: nếu i <= m thì chuyển sang bước 5, ngược lại thì chuyển sang bước 10

Bước 5: nếu MaxH[i, n] 0 thì chuyển sang bước 6, ngược lại thì chuyển sang bước 7

Bước 6: tăng rankA ++

Bước 7: nếu MaxH[i, n + 1] 0 thì chuyển sang bước 8, ngược lại chuyển sang bước 9

Bước 8: tăng rankB ++

Bước 9: tăng i++ rồi quy lại bước 4

Bước 10: in ra output

\* Bằng sơ đồ khối

Diagram

Description automatically generated

# V. đánh giá thuật toán

1. Ưu điểm.

* Giảm được sai số trong quá trình tính toán khi chia cho số gần 0
* Dễ lập trình tính toán trên máy tính
* Độ chính xác của nghiệm cao
* Giải được các phương trình đại số tuyến tính khi ma trận A bất kì còn với Phương pháp lặp đơn, lặp Jacobi, pp Seidel và Gauss – Seidel điều kiện ma trận A phải là ma trận vuông
* Biểu diễn được nghiệm trong trường hợp vô số nghiệm

1. Nhược điểm

* Độ phức tạp thuật toán lớn, do mỗi lần ta đều phải chọn phần tử giải.

\*Chú ý:

* Đối với những ma trận hệ số đơn giản, hoặc có thể coi là “ước”, “bội” của nhau thì kết quả nhận được sẽ hoàn toàn đúng, các phép tính đơn giản sẽ không có ảnh hưởng đến sai số
* Đối với những ma trận cỡ lớn thì nên sử dụng các phương pháp lặp để giải vì tốc độ hội tụ nó sẽ nhanh hơn rất nhiều so với phương pháp Gauss - Jordan