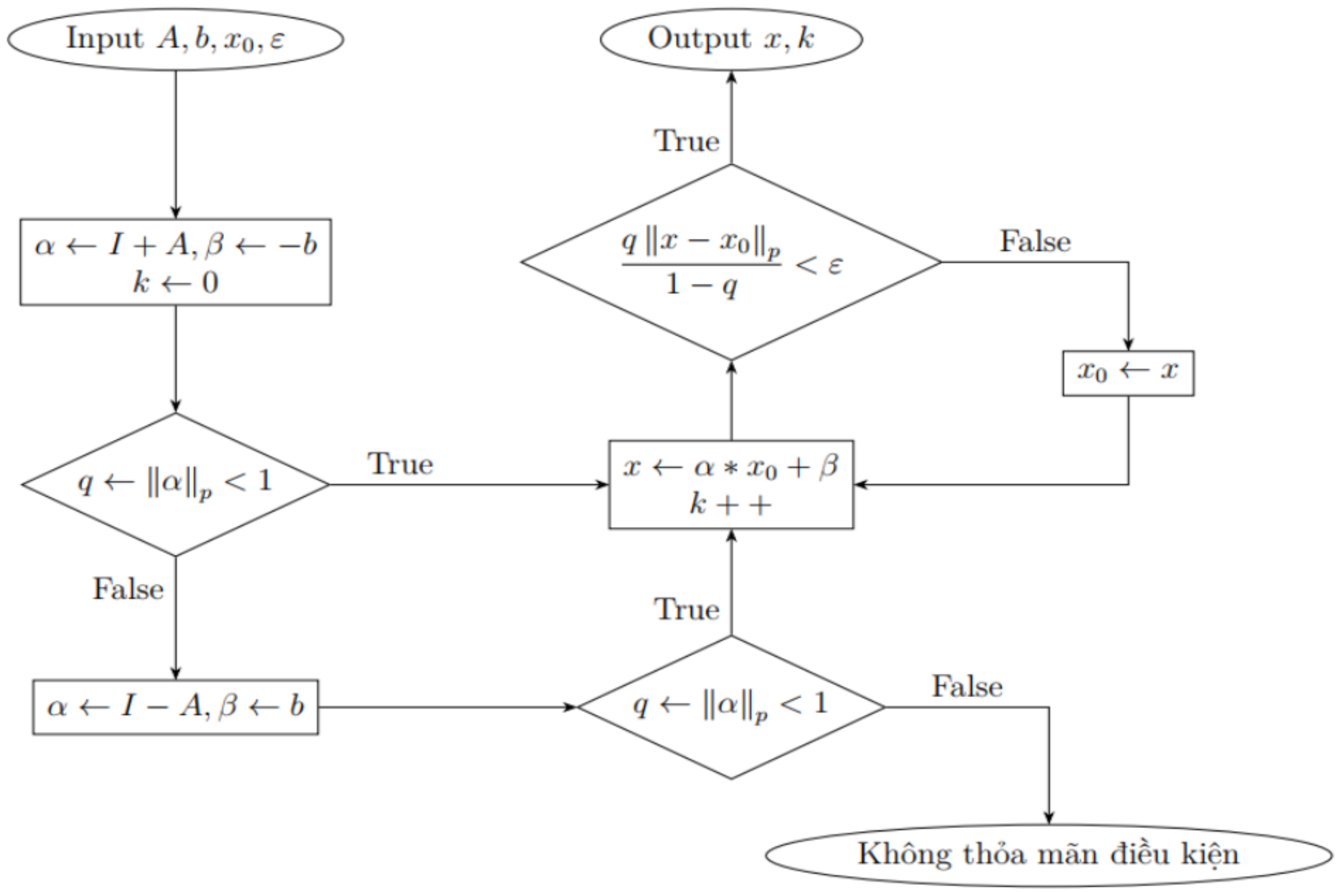
I. Thuật toán tổng quát

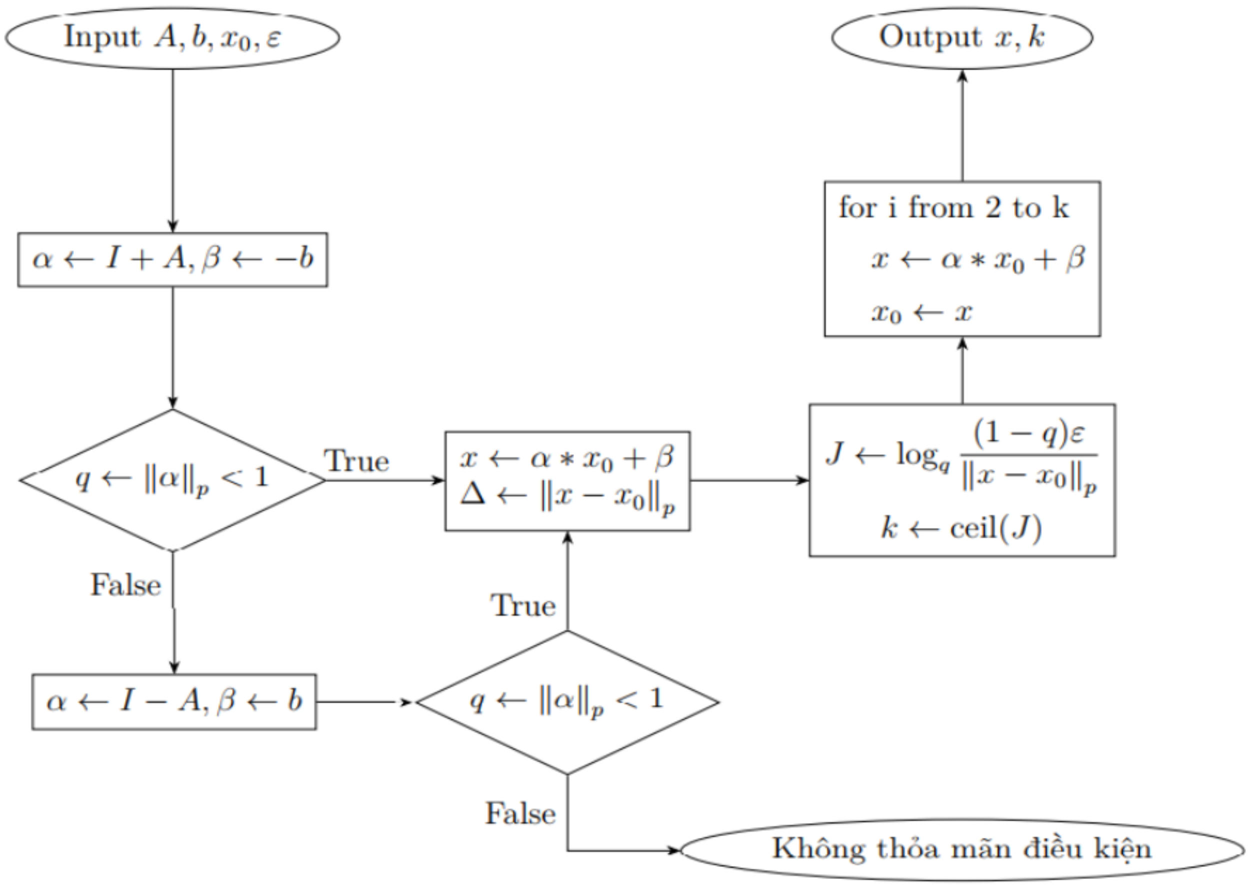
1. Lặp đơn hậu nghiệm

Sơ đồ khối cho sai số hậu nghiệm:

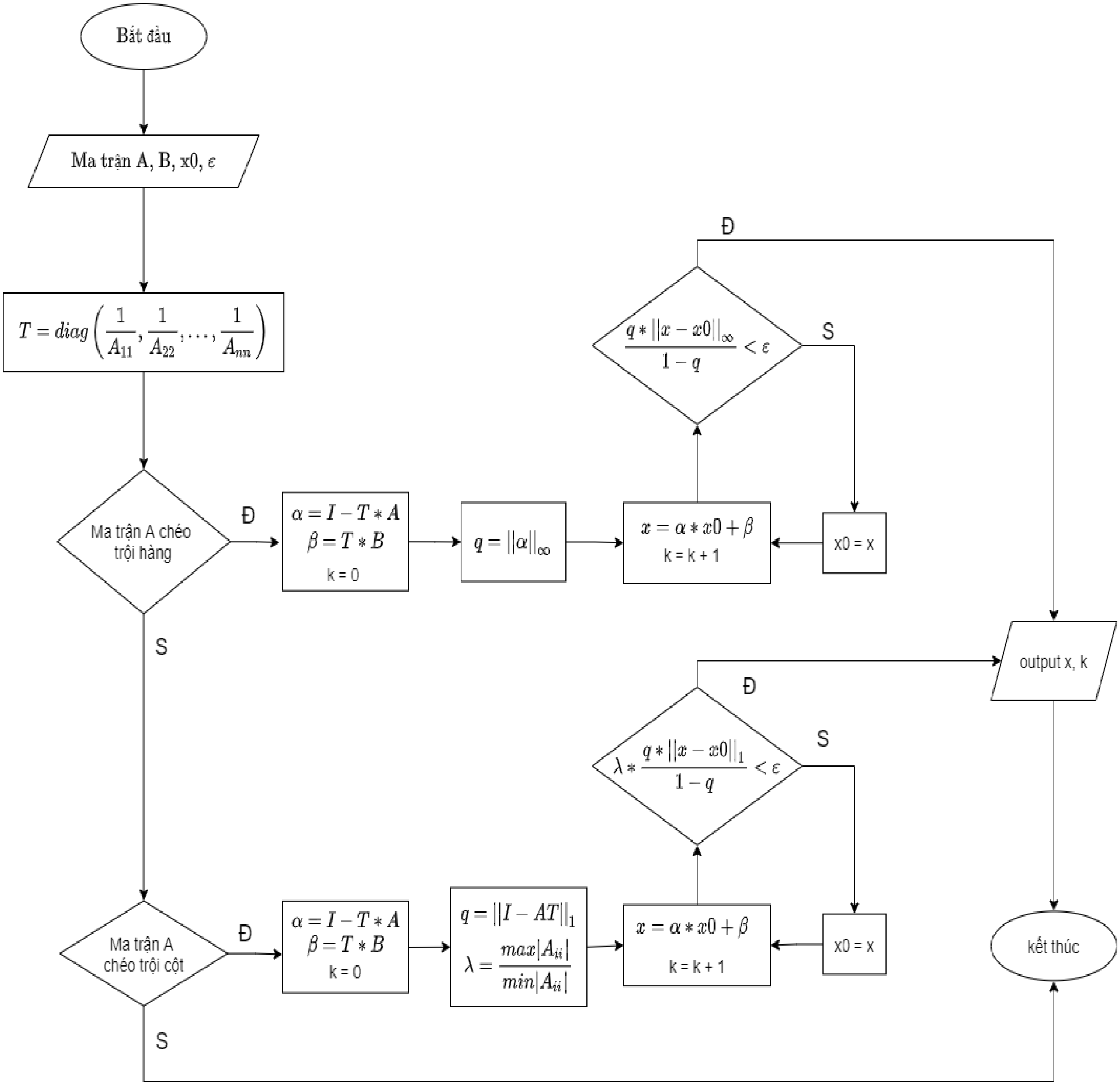


2. Lặp đơn tiên nghiệm

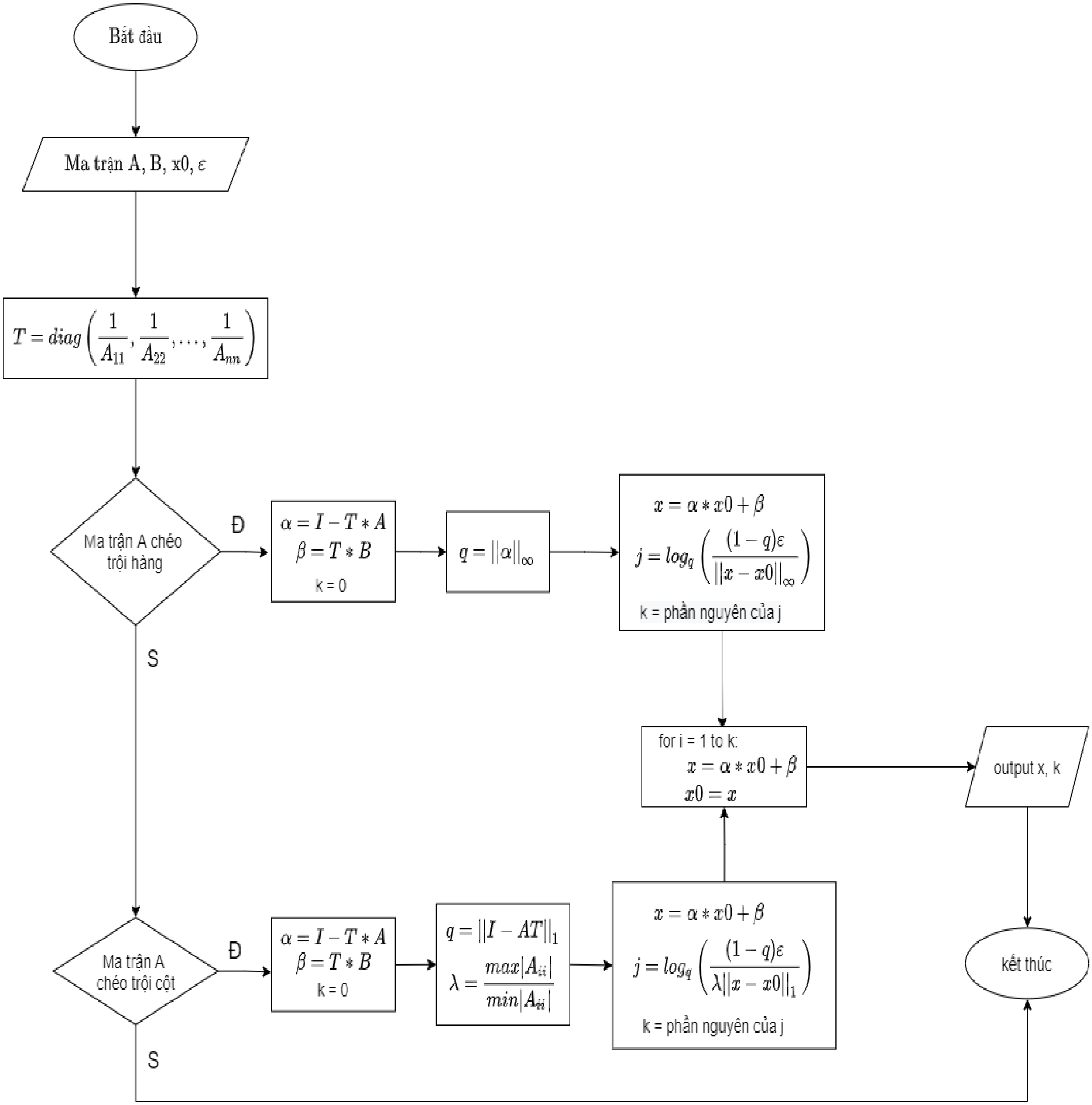
Sơ đồ khối cho công thức tiên nghiệm:



3. Lặp Jacobi hậu nghiệm.



4. Lặp Jacobi tiên nghiệm



II. Thuật toán chi tiết. Giả mã

1. Hàm tính chuẩn của ma trận

input: ma trận A, loại chuẩn lựa chọn: type (với type = 1: chuẩn hàng; type = 2: chuẩn cột; type = 3: chuẩn Euclid; type = 4: chuẩn trị riêng
output: giá trị ||A||

```
function getNorm:
        norm = 0
        if type = 1:
              for i = 1 to A.row:
                      norm1 = 0
                      for j = 1 to A.col:
                            norm1  = norm1 + |Aij|
                      if norm1 > norm:
                              norm = norm1
       return norm
       if type = 2:
              for i = 1 to A.col:
                      norm1 = 0
                      for j = 1 to A.row:
                            norm1 = norm1 + |Aij|
                      if norm1 > norm:
                              norm = norm1
       return norm
       if type = 3:
              for i = 1 to A.row:
                      norm1 = 0
                      for j = 1 to A.col:
```

```
                norm1 = norm1 + A_ij * A_ij
    return √norm1
    if type = 4:
        ma trận B = A^T * A
        for i = 1 to A.row:
            X[i][1] = 1
        t0 = 0
        t1 = 0
        do:
            t0 = t1
            X = B * X
            s = X[1][1]
            for i = 2 to A.row – 1:
                if |X_{i, 1}| > s:
                    j = i
                    s = |X_{i, 1}|
            t1 = X[j][0]
            for i = 0 to A.row – 1:
                X[i][0] = X[i][0] / t1
        while (|t1 – t0| > 10^-5)
        return √t1
```

1. Lặp đơn.

2. Hàm lặp theo phương pháp lặp đơn

input: ma trận A, B, X0, epsi
output: nghiệm X và số lần lặp k hoặc thông báo không chạy được lặp đơn

2.1. Lặp đơn hậu nghiệm

```
Function singLoopHN:
    check = 0
    alpha = I – A //I là ma trận đơn vị cùng cỡ với ma trận A
    for i = 1 to 4:
        if getNorm(alpha, i) < 1:
            check = i
            beta = B
            thoát khỏi vòng lặp for
    if check = 0:
        alpha = E + A
        for i = 1 to 4:
            if getNorm(alpha, i) < 1:
                check = i
                beta = - B
                thoát khỏi vòng lặp for
        if check = 0:
            print "Không thể lặp đơn"
            kết thúc chương trình
    q = getNorm(alpha, check)
    k = 0
    X = X0
    do:
        X0 = X
        X = alpha * X0 + beta
        k ++
    while q * getNorm(X – X0, check) > epsi * (1 – q)
    print "nghiệm là: " X
    print "số lần lặp: " k
    kết thúc chương trình
```

2.2. Lặp đơn tiên nghiệm

```
Function singLoopTN:
    check = 0
    alpha = I – A //I là ma trận đơn vị cùng cỡ với ma trận A
    for i = 1 to 4:
        if getNorm(alpha, i) < 1:
            check = i
            beta = B
            thoát khỏi vòng lặp for
    if check = 0:
        alpha = E + A
        for i = 1 to 4:
            if getNorm(alpha, i) < 1:
                check = i
                beta = - B
                thoát khỏi vòng lặp for
        if check = 0:
            print "Không thể lặp đơn"
            kết thúc chương trình
    q = getNorm(alpha, check)
    X = alpha * X0 + beta
    j  = log(epsi * (1 – q) / getNorm(X – X0, check)) / log(q)
    k = phần nguyên của j
    for i = 1 to k:
        X = alpha * X0 + beta
        X0 = X
    print "Nghiệm là: " X
    print "Số lần lặp là: " k
    kết thúc chương trình
```

2. Lặp Jacobi.

1. Hàm kiểm tra tính chéo trội của ma trận A

1.1. Kiểm tra tính chéo trội hàng

```
input: Ma trận A
output: true nếu A chéo trội hàng, false nếu A không chéo trội hàng
Function check_row:
        for i = 1 to A.row:
              max = A[i][i]
              for j = 1 to A.row:
                      if i ≠ j:
                              max = max - |A[i][j]|
              if max <= 0:
                      return false
        return true
```

1.2. Kiểm tra tính chéo trội cột

```
input: ma trận A
output: true nếu A chéo trội cột, false nếu A không chéo trội cột
function check_col:
        for i = 1 to A.row:
              max = A[i][i]
              for j = 1 to A.row:
                      if i ≠ j:
                              max = max - |A[j][i]|
              if max <= 0:
```

```
                return false
        return true
```

2. Hàm lặp Jacobi

input: ma trận A, B, X0, epsi
output: nghiệm X, số lần lặp k
2.1. Hàm lặp Jacobi hậu nghiệm

```
Function jacobiLoopHN:
        for i = 1 to A.row:
                for j = 1 to A.row:
                        if i = j:
                                T[i][j] = 1 / A[i][i]
                        else:
                                T[i][j] = 0
        if check_row = true:
                alpha = I – T * A
                beta = T * B
                k = 0
                q = getNorm(alpha, 1)
                X = X0
                do:
                        X0 = X
                        X = alpha * X0 + beta
                        k = k + 1
                while q * getNorm(X – X0, 1) < epsi * (1 – q)
                print "nghiệm là: " X
```

```
        print "số lần lặp: " k
        kết thúc chương trình
    if check_col = true:
        max = |A[1][1]|
        min = |A[1][1]|
        alpha = I – T*A
        beta = T * B
        k = 0
        q = getNorm(I – A*T, 2)
        for i = 2 to A.row:
            if |A[i][i]| > max:
                max = |A[i][i]|
            if |A[i][i]| < min:
                min = |A[i][i]|
        λ = max / min
        X = X0
        do:
            X0 = X
            X = alpha * X0 + beta
            k = k + 1
        while λ*q*getNorm(X – X0, 2) < epsi * (1 – q)
        print "nghiệm là: " X
        print "số lần lặp: " k
        kết thúc chương trình
    else:
        print "ma trận A không chéo trội, không lặp Jacobi được"
        kết thúc chương trình
```

2.2. Lặp Jacobi theo công thức tiên nghiệm

Function jacobiLoopTN:

```
for i = 1 to A.row:
    for j = 1 to A.row:
        if i = j:
            T[i][j] = 1 / A[i][i]
        else:
            T[i][j] = 0
if check_row = true:
    alpha = I – T * A
    beta = T * B
    q = getNorm(alpha, 1)
    X = alpha * X0 + beta
    j = log((1 – q) * epsi / getNorm(X – X0, 1)) / log (q)
    k = phần nguyên của j
    for i = 1 to k:
        X = alpha * X0 + beta
        X0 = X
    print "nghiệm là: " X
    print "số lần lặp: " k
    kết thúc chương trình
if check_col = true:
    max = |A[1][1]|
    min = |A[1][1]|
    alpha = I – T*A
    beta = T * B
    q = getNorm(I – A*T, 2)
    for i = 2 to A.row:
        if |A[i][i]| > max:
            max = |A[i][i]|
        if |A[i][i]| < min:
            min = |A[i][i]|
    λ = max / min
```

```
        X = alpha * X0 + beta
        j = log((1 – q)*epsi / (λ * getNorm(X – X0, 2)) / log (q)
        k = phần nguyên của j
        for i = 1 to k:
                X = alpha * X0 + beta
                X0 = X
        print “nghiệm là: “ X
        print “số lần lặp: “ k
        kết thúc chương trình
    else:
        print “ma trận A không chéo trội, không lặp Jacobi được”
        kết thúc chương trình
```

III. Ưu và nhược điểm của phương pháp

1. Ưu điểm

- Phương pháp lặp đơn giải quyết được sự bất ổn định của nghiệm khi giải hệ bằng phương pháp đúng (Gauss, Gauss-Jordan, Choleski)
- Tối ưu được bộ nhớ
- Dễ cài đặt trên máy tính
- Chi phí xử lí và tốc độ hội tụ nhanh hơn phương pháp tính toán trực tiếp

2. Nhược điểm

- Phương pháp lặp đơn: chỉ lặp được khi $||I – A|| < 1$ hoặc $||I + A|| < 1$
- Phương pháp lặp Jacobi: chỉ lặp được khi ma trận A là chéo trội hàng hoặc chéo trội cột

- ⇨ Lớp phương trình đại số tuyến tính giải được bằng phương pháp lặp đơn và lặp Jacobi là tương đối hẹp

*Chú ý: Tốc độ hội tụ của phương pháp lặp đơn và lặp Jacobi thì chậm hơn nhiều so với phương pháp lặp Seidel và Gauss Seidel

* Phương pháp lặp Jacobi chỉ giải được với điều kiện ma trận A là ma trận vuông

Qua thuật toán và hệ thống ví dụ, ta rút ra kết luận sau:
Đối với phương pháp lặp đơn

- Ưu điểm:
  1. Chi phí xử lí tính toán và tốc độ hội tụ nghiệm nhanh hơn so với phương pháp trực tiếp
  2. Hạn chế nhiều sai số do tính toán
- Nhược điểm:
  1. Chỉ xử lí được với một số ma trận thỏa mãn điều kiện có chuẩn nhỏ hơn 1
  2. Không hiệu quả với các ma trận cỡ nhỏ

Đối với phương pháp lặp Jacobi

- Ưu điểm:
  1. Chi phí xử lí tính toán và tốc độ hội tụ nghiệm nhanh hơn so với phương pháp trực tiếp
  2. Hạn chế nhiều sai số do tính toán
- Nhược điểm:
  1. Tuy khắc phục được nhược điểm về chuẩn của phương pháp lặp đơn, nhưng phạm vi sử dụng vẫn còn hẹp vì phải thỏa mãn điều kiện khác (tính chéo trội)
  2. Không hiệu quả với các ma trận cỡ nhỏ

IV. Tóm tắt phương pháp.

Phương pháp chỉ áp dụng được với những ma trận vuông

1. Phương pháp lặp đơn.

Đưa phương trình Ax = B về dạng X = α X + β qua 2 cách:

+ Cách 1: α = A + I, β = - B

+ Cách 2: α = I - A, β = B

a. Điều kiện của phương pháp: $q = \|\alpha\| < 1$
b. Dãy lặp của phương pháp: $x_n = x_{n-1} *_{\alpha} + \beta$
c. Công thức sai số (điều kiện dừng lặp)
- Công thức hậu nghiệm: $\|x_n - x^*\| <= \frac{q}{1-q} * \|x_n - x_{n-1}\|$
- Công thức tiên nghiệm: $\|x_n - x^*\| <= \frac{q^n}{1-q} * \|x_1 - x_0\|$

2. Phương pháp lặp Jacobi. Dựa trên ý tưởng phương pháp lặp đơn giải quyết các bài toán mà $\|\alpha\| >= 1$, và ma trận A là ma trận chéo trội hàng hoặc ma trận chéo trội cột.

a. Điều kiện của phương pháp:
- Ma trận A là ma trận chéo trội hàng hoặc ma trận chéo trội cột

• Ma trận chéo trội hàng

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{m} |a_{ij}|$$

• Ma trận chéo trội cột

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{m} |a_{ji}|$$

+ Ma trận A là ma trận chéo trội hàng: các chuẩn dùng ở đây là chuẩn hàng

$$T = diag\left(\frac{1}{a_{11}}, \frac{1}{a_{22}}, \cdots, \frac{1}{a_{mm}}\right);$$

$$Ax = b \Leftrightarrow x = (I - TA)x + Tb,$$

$$B = I - TA,\ d = Tb$$

$$x^{(0)} \in \mathbb{R}^m,\ \ x^{(n+1)} = Bx^{(n)} + d.$$

⇨ Công thức sai số: q = ||B||

+ Công thức hậu nghiệm: $||x_n - x^*|| <= \frac{q}{1-q} * ||x_n - x_{n-1}||$

+ Công thức tiên nghiệm: $||x_n - x^*|| <= \frac{q^n}{1-q} * ||x_1 - x_0||$

+ Ma trận A là ma trận chéo trội cột: Các chuẩn dùng ở đây là chuẩn cột

: $x^{(n)} = Bx^{(n-1)} + d$ với $\begin{cases} B = I - TA \\ d = Tb \\ q = \|B_1\|_1 \end{cases}$

⇨ Công thức sai số:

$$\lambda = \frac{\max |a_{ii}|}{\min |a_{ii}|}$$

⇨

+ Hậu nghiệm:

$$\boxed{\|x^{(n)} - x^*\| \leq \frac{\lambda q}{1-q}\|x^{(n)} - x^{(n-1)}\|}$$

+ Tiên nghiệm:

$$\boxed{\|x^{(n)} - x^*\| \leq \frac{\lambda q^n}{1-q}\|x^{(1)} - x^{(0)}\|}$$