# Phương pháp chia đôi

Các bước của phương pháp chia đôi được mô tả như sau:

Đầu vào:

* Hàm số f(x)
* Hai giá trị a và b, sao cho f(a) và f(b) có giá trị khác dấu
* Số lần lặp tối đa N
* Độ chính xác epsilon

Đầu ra:

* Nghiệm x của phương trình f(x) = 0

Mã giả:

1. Đặt i = 0
2. Lặp cho đến khi i = N hoặc độ chính xác đủ đáp ứng:

a. Đặt c = (a + b) / 2

b. Tính giá trị của hàm số f(c)

c. Nếu f(c) = 0 hoặc độ chính xác đủ đáp ứng, trả về c là kết quả

d. Nếu f(a) và f(c) có giá trị khác dấu, đặt b = c

e. Ngược lại, đặt a = c

f. Đặt i = i + 1 và quay lại bước 2

# Phương pháp điểm bất động

Đầu vào:

* Hàm số f(x)
* Hàm số g(x)
* Giá trị x0 là điểm bắt đầu
* Số lần lặp tối đa N
* Độ chính xác epsilon

Đầu ra:

* Giá trị x\* là nghiệm của phương trình f(x) = 0 với độ chính xác đủ đáp ứng

Mã giả:

1. Đặt i = 0 và x = x0
2. Lặp cho đến khi i = N hoặc độ chính xác đủ đáp ứng:

a. Đặt x = g(x)

b. Tính sai số ||x - g(x)|| và kiểm tra xem nó có đạt đến độ chính xác mong muốn không

c. Nếu đạt độ chính xác đủ đáp ứng, trả về x là kết quả

d. Ngược lại, đặt i = i + 1 và quay lại bước 2

# Phương pháp newton

Đầu vào:

* Hàm số f(x)
* Đạo hàm f'(x)
* Giá trị x0 là điểm bắt đầu
* Số lần lặp tối đa N
* Độ chính xác epsilon

Đầu ra:

* Giá trị x\* là nghiệm của phương trình f(x) = 0 với độ chính xác đủ đáp ứng

Mã giả:

1. Đặt i = 0 và x = x0
2. Lặp cho đến khi i = N hoặc độ chính xác đủ đáp ứng:

a. Tính giá trị f(x) và f'(x) tại x

b. Nếu f'(x) = 0 thì dừng phương pháp vì không thể tính được tiếp theo

c. Đặt x = x - f(x)/f'(x)

d. Tính sai số ||f(x)|| và kiểm tra xem nó có đạt đến độ chính xác mong muốn không

e. Nếu đạt độ chính xác đủ đáp ứng, trả về x là kết quả

f. Ngược lại, đặt i = i + 1 và quay lại bước 2

# Phương pháp dây cung

Đầu vào:

* Hàm số f(x)
* Giá trị x0 và x1 là hai điểm bắt đầu
* Số lần lặp tối đa N
* Độ chính xác epsilon

Đầu ra:

* Giá trị x\* là nghiệm của phương trình f(x) = 0 với độ chính xác đủ đáp ứng

Mã giả:

1. Đặt i = 0 và x0 = x0, x1 = x1
2. Lặp cho đến khi i = N hoặc độ chính xác đủ đáp ứng:

a. Tính giá trị f(x0) và f(x1)

b. Tính x2 = x1 - f(x1)\*(x1-x0)/(f(x1)-f(x0))

c. Tính sai số ||x2 - x1|| và kiểm tra xem nó có đạt đến độ chính xác mong muốn không

d. Nếu đạt độ chính xác đủ đáp ứng, trả về x2 là kết quả

e. Ngược lại, đặt x0 = x1, x1 = x2, i = i + 1 và quay lại bước 2

# Phương pháp điểm sai

Đầu vào:

* Hàm số f(x)
* Hàm số g(x) sao cho x = g(x) ⬄ f(x) = 0
* Điểm khởi đầu x0
* Số lần lặp tối đa N
* Độ chính xác epsilon

Đầu ra:

* Giá trị x\* là nghiệm của phương trình f(x) = 0 với độ chính xác đủ đáp ứng

Mã giả:

1. Đặt i = 0 và x = x0
2. Lặp cho đến khi i = N hoặc độ chính xác đủ đáp ứng:

a. Tính giá trị xnew = g(x) với công thức lặp đơn g(x)

b. Tính sai số ||xnew - x|| và kiểm tra xem nó có đạt đến độ chính xác mong muốn không

c. Nếu đạt độ chính xác đủ đáp ứng, trả về xnew là kết quả

d. Ngược lại, đặt x = xnew, i = i + 1 và quay lại bước 2

# Doolittle

Đầu vào:

* Ma trận hệ số A kích thước n x n
* Ma trận vế phải b kích thước n x 1

Đầu ra:

* Nghiệm của hệ phương trình Ax = b

Mã giả:

1. Khởi tạo ma trận tam giác dưới L và ma trận tam giác trên U với giá trị ban đầu bằng 0
2. Với i = 1 đến n:

a. Gán U[i][i] bằng 1

b. Với j = i đến n: Tính tổng L[j][k] \* U[k][i] với k từ 1 đến i - 1, gán vào U[j][i]

c. Với j = i + 1 đến n: Tính tổng L[j][k] \* U[k][i] với k từ 1 đến i - 1, chia cho U[i][i] và gán vào L[j][i]

d. Lưu U[i][i] vào biến pivot e. Với j = i + 1 đến n:

i. Tính tổng L[i][k] \* U[k][j] với k từ 1 đến i - 1, gán vào U[i][j]

ii. Tính A[j][i] - tổng L[j][k] \* U[k][i] với k từ 1 đến i - 1, chia cho pivot và gán vào U[j][i]

1. Giải hệ phương trình Ly = b với phương pháp giải tam giác dưới
2. Giải hệ phương trình Ux = y với phương pháp giải tam giác trên

Trong đó, để giải hệ phương trình tam giác dưới Ly = b, ta sử dụng các bước sau:

1. Khởi tạo vector y với giá trị ban đầu bằng 0
2. Với i = 1 đến n:

a. Gán y[i] bằng b[i]

b. Với j = 1 đến i - 1: Trừ L[i][j] \* y[j] từ y[i]

c. Chia y[i] cho L[i][i]

1. Trả về vector y là nghiệm của hệ phương trình Ly = b

Để giải hệ phương trình tam giác dưới Ly = b, ta sử dụng các bước sau:

1. Khởi tạo vector x với giá trị ban đầu bằng 0
2. Với i = n đến 1:

a. Gán x[i] bằng y[i]

b. Với j = i + 1 đến n: Trừ U[i][j] \* x[j] từ x[i]

c. Chia x[i] cho U[i][i]

1. Trả về vector x là nghiệm của hệ phương trình Ux = y

# Gauss-Seidel

Đầu vào: ma trận A kích thước nxn, vector b kích thước n, vector x(k) kích thước n, sai số ε

Đầu ra: vector x(k+1) kích thước n

1. Khởi tạo biến k = 0

2. Khởi tạo vector x(k+1) với giá trị ban đầu bằng vector x(k)

3. Lặp lại các bước 4 đến 8 cho đến khi đạt được giá trị cần thiết:

a. Tăng biến k lên 1 đơn vị

b. Khởi tạo biến delta = 0

c. Với i = 1 đến n:

i. Khởi tạo biến sigma = 0

ii. Với j = 1 đến i-1:

Tính giá trị sigma = sigma + A[i][j] \* x(k+1)[j]

iii. Với j = i+1 đến n:

Tính giá trị sigma = sigma + A[i][j] \* x(k)[j]

iv. Gán giá trị mới cho x(k+1)[i] = (1/A[i][i]) \* (b[i] - sigma)

v. Tính giá trị mới cho delta = max(delta, |x(k+1)[i] - x(k)[i]|)

d. Nếu delta < ε, dừng vòng lặp và trả về vector x(k+1) là nghiệm của hệ phương trình Ax = b

4. Kết thúc

# Jacobi

Input: ma trận A kích thước nxn, vector b kích thước n, vector x(k) kích thước n, sai số ε

Output: vector x(k+1) kích thước n

1. Khởi tạo biến k = 0

2. Khởi tạo vector x(k+1) với giá trị ban đầu bằng vector x(k)

3. Lặp lại các bước 4 đến 8 cho đến khi đạt được giá trị cần thiết:

a. Tăng biến k lên 1 đơn vị

b. Khởi tạo biến delta = 0

c. Với i = 1 đến n:

i. Khởi tạo biến sigma = 0

ii. Với j = 1 đến n:

Nếu j ≠ i, tính giá trị sigma = sigma + A[i][j] \* x(k)[j]

iii. Gán giá trị mới cho x(k+1)[i] = (1/A[i][i]) \* (b[i] - sigma)

iv. Tính giá trị mới cho delta = max(delta, |x(k+1)[i] - x(k)[i]|)

d. Nếu delta < ε, dừng vòng lặp và trả về vector x(k+1) là nghiệm của hệ phương trình Ax = b

4. Kết thúc

# Khử Gauss

Input: ma trận A kích thước nxn, vector b kích thước n

Output: vector x kích thước n

1. Khởi tạo ma trận A' bằng cách nối ma trận A và vector b

2. Cho i chạy từ 1 đến n-1:

a. Tìm phần tử lớn nhất a[i][j] với j từ i đến n

b. Hoán đổi hai hàng i và k (với a[k][j] là phần tử lớn nhất tìm được ở bước a)

c. Cho j chạy từ i+1 đến n:

i. Tính hệ số u = a[j][i]/a[i][i]

ii. Cho k chạy từ i+1 đến n+1:

- a[j][k] = a[j][k] - u \* a[i][k]

3. Khởi tạo vector x kích thước n với giá trị ban đầu x[n] = a[n][n+1]/a[n][n] và x[i] = (a[i][n+1] - sum(a[i][j]\*x[j], j=i+1 đến n))/a[i][i] với i từ n-1 đến 1

4. Trả về vector x là nghiệm của hệ phương trình Ax = b

5. Kết thúc

# Cramer

Input: ma trận A kích thước nxn, vector b kích thước n

Output: vector x kích thước n

1. Tính định thức detA của ma trận hệ số A

2. Cho i chạy từ 1 đến n:

a. Tạo một bản sao của ma trận A và thay cột thứ i bằng vector b

b. Tính định thức detAi của ma trận mới này

c. Gán giá trị nghiệm xi bằng detAi/detA

3. Trả về vector x là nghiệm của hệ phương trình Ax = b

4. Kết thúc

# Lagrange

Input: một mảng điểm [(x0, y0), (x1, y1), ..., (xn, yn)], và một giá trị x

Output: giá trị của đa thức nội suy Lagrange tại x

1. Khởi tạo giá trị của đa thức nội suy f(x) bằng 0

2. Cho i chạy từ 0 đến n:

a. Khởi tạo giá trị của đa thức Lagrange Li(x) bằng 1

b. Cho j chạy từ 0 đến n:

i. Nếu j khác i, tính Li(x) bằng Li(x) \* (x - xj) / (xi - xj)

c. Cộng tổng Li(x)\*yi vào giá trị của đa thức nội suy f(x)

3. Trả về giá trị của đa thức nội suy f(x) tại x

4. Kết thúc

# Nội suy newton

Input: một mảng điểm [(x0, y0), (x1, y1), ..., (xn, yn)], và một giá trị x Output: giá trị của đa thức nội suy Newton tại x

1. Khởi tạo mảng f với giá trị y0, y1, ..., yn
2. Với mỗi i từ 1 đến n: a. Với mỗi j từ n đến i: i. Tính giá trị f[j] = (f[j] - f[j-1]) / (x[j] - x[j-i])
3. Khởi tạo giá trị của đa thức nội suy f(x) bằng f[n]
4. Cho i chạy từ n-1 đến 0: a. Cập nhật giá trị đa thức nội suy f(x) bằng f[i] + (x - xi) \* f(x, i+1)
5. Trả về giá trị của đa thức nội suy f(x) tại x
6. Kết thúc

# Spline

Input: một mảng điểm [(x0, y0), (x1, y1), ..., (xn, yn)], và một giá trị x Output: giá trị của đa thức nội suy Spline tại x

1. Sắp xếp các điểm theo thứ tự tăng dần của giá trị x
2. Tính các độ dốc giữa các điểm theo công thức: a. Cho h[i] = x[i+1] - x[i] (khoảng cách giữa hai điểm liên tiếp) b. Cho b[i] = (y[i+1] - y[i]) / h[i] (độ dốc giữa hai điểm liên tiếp)
3. Khởi tạo các giá trị a, c, d bằng 0
4. Tính các giá trị u, v, và z theo công thức:

a. Khởi tạo giá trị u[0] = 0 và z[0] = 0

b. Cho i chạy từ 1 đến n-1:

i. Tính giá trị u[i] = h[i] / (h[i] + h[i-1]) \* u[i-1] + 2

ii. Tính giá trị z[i] = (b[i] - b[i-1]) / (h[i] + h[i-1]) - h[i-1] \* z[i-1] / (h[i] + h[i-1])

c. Tính giá trị v bằng (3 \* b[n-1] / h[n-1]) - (3 \* b[n-2] / h[n-2]) - z[n-2] \* h[n-2]

1. Tính các giá trị c và d theo công thức:

a. Khởi tạo giá trị c[n-1] và d[n-1] bằng 0

b. Cho i chạy từ n-2 đến 0:

i. Tính giá trị c[i] = (z[i] - h[i] \* c[i+1]) / u[i]

ii. Tính giá trị d[i] = (c[i+1] - c[i]) / (3 \* h[i])

1. Tìm đoạn chứa giá trị x, xác định giá trị của đa thức nội suy tại x theo công thức: a. Cho i là chỉ số của đoạn chứa giá trị x

b. Tính giá trị của đa thức nội suy tại x bằng công thức: f(x) = y[i] + b[i] \* (x - x[i]) + c[i] \* (x - x[i])^2 + d[i] \* (x - x[i])^3

1. Trả về giá trị của đa thức nội suy Spline tại x
2. Kết thúc.