Solving Linear Systems by using Jacobi, GS, SOR

2016039034박준형
2018021794염태은

Contents

Review

Jacobi, GS, SOR

Review : Jacobi Method & Gauss – Seidel Method

Jacobi Method

A = M – N

M = D

N = L+U

$$M = \begin{bmatrix} a_{11} & & & \\ & \ddots & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}, \quad N = -\begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}.$$

Gauss – Seidel Method

A = M – N

M = D – L

N = U

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b}.$$

$$(D - L)\mathbf{x}^{(k+1)} = \mathbf{b} + U\mathbf{x}^{(k)}$$

Review

▪ Jacobi Method

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j\neq i} a_{ij}x^{(k)}\right]$$

▪ Gauss – Seidel Method

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right]$$

▪ Successive Over - Relaxation Method

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij}x_j^{(k)}\right] + (1-\omega)x_i^{(k)}$$

Problem #1

Part 2

Problem #1

A = gallery ('poisson' , n)

설명: 푸아송 방정식의 블록 삼중대각 행렬(희소 행렬)

구문:

- `A = gallery('poisson',n)`은 `n×n` 메시에 5점 연산자를 사용한 푸아송 방정식의 이산화에서 얻은 차수 `n^2`의 블록 삼중대각 희소 행렬을 반환합니다.

$$A = \begin{pmatrix} 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & \cdots & 0 \\ 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 4 \end{pmatrix} \qquad x = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\Rightarrow \quad Ax = b$$

Part 3

Code : Jacobi Method

```
%% Algorithm: Jacobi Method
function [x,itr] = jacobi(A,b,x,tol)
normVal=Inf;
itr = 0;
n=size(x,1);
while normVal>tol
    xold=x;

    for i=1:n

        sigma=0;

        for j=1:n

            if j~=i
                sigma=sigma+A(i,j)*xold(j);
            end

        end

        x(i)=(b(i)-sigma)/A(i,i);
    end

    itr=itr+1;
    normVal=norm(xold-x);
end

```

```
n=5;
A=gallery('poisson',n);
n=n^2;
b=A*ones(n,1);
tol=1e-5;
x=zeros(n,1);

```

Part 3

Code : Gauss – Seidel Method

```
%% Algorithm: Gauss Seidel Method
function [x,itr] = gs(A,b,x,tol)
normVal=1;
itr = 0;
n=size(x,1);
while normVal>tol
    xold=x;

    for i=1:n

        sigma=0;

        for j=1:i-1
                sigma=sigma+A(i,j)*x(j);
        end

        for j=i+1:n
                sigma=sigma+A(i,j)*xold(j);
        end

        x(i)=(1/A(i,i))*(b(i)-sigma);
    end

    itr=itr+1;
    normVal=norm(xold-x);
end

```

Part 3

Code : Successive Over - Relaxation Method

```
%% Algorithm: SOR Method
function [x,itr] = sor(A,b,x,w,tol)
normVal=Inf;
itr = 0;
n=size(x,1);
while normVal>tol
    xold=x;

    for i=1:n
        sigma=0;

          for j=1:i-1
                  sigma=sigma+A(i,j)*x(j);
          end

          for j=i+1:n
                  sigma=sigma+A(i,j)*xold(j);
          end

        x(i)=w*(1/A(i,i))*(b(i)-sigma)+(1-w)*xold(i);
    end

    itr=itr+1;
    normVal=norm(xold-x);
end
```

$$\omega = \frac{2}{1+\sqrt{1-[\rho(T_j)]^2}}.$$

```
%% Find optimal omega
D=diag(1./diag(A));
L =-tril(A,-1);
U = -triu(A,1);
T=D*(L+U);
p = max(eig(T));
w = 2/(1+sqrt(1-p^2));
```

Part 4
Conclusion

Iteration number	**n = 5**	**n = 10**
Jacobi	78	255
Gauss-Seidel	43	137
SOR	17	31
	$\omega = 1.33$	$\omega = 1.56$