FDM流れ解析　No.2
SOR法及びCG法プログラム

2007年6月

後　保範 (東京工芸大学)

目次

1. Gauss-Seidel法

Ax=bの解xを反復計算する
$A=\{a_{ij}\}$, $x=\{x_i\}$, $b=\{b_i\}$

Gauss-Seidel法では下記を反復計算

$$for \quad k=1;n$$

$$x_k=\left(b_k-\sum_{i=1}^{k-1}a_{ki}x_i-\sum_{i=k+1}^{n}a_{ki}x_i\right)/a_{kk}$$

2. SOR法

SOR法は下記を収束まで反復計算する
$\omega(1<\omega<2)$は加速係数である。

$$for \ k=1;n$$

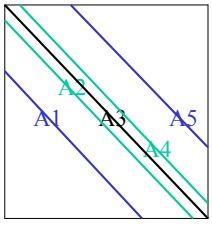
$$r=\left(b-\sum_{i=1}^{k-1}a_{ki}x_i-\sum_{i=k+1}^{n}a_{ki}x_i\right)/a_{kk}-x_k$$

$$x_k=x_k+\omega r$$

2.1 2次元差分法用SOR法

- 下記を $\|r\|_2\le\varepsilon$(収束条件) となるまで反復計算

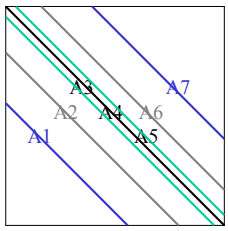
```
for j=1;ny
  for i=1;nx
    r = (b(i,j) - A1(i,j)*x(i,j-1)
             - A2(i,j)*x(i-1,j)
             - A4(i,j)*x(i+1,j)
             - A5(i,j)*x(i,j+1) )
            /A3(i,j) - x(i,j)
    x(i,j) = x(i,j) + ω*r
```



2.2 3次元差分法用SOR法

- 下記 $\|r\|_2\le\varepsilon$(収束条件) となるまで反復計算

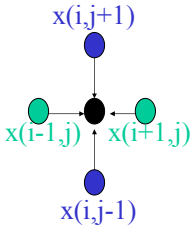
```
for k=1,nz
 for j=1,ny
  for i=1,nx
    r = (b(i,j,k) - A1(i,j,k)*x(i,j,k-1)
             - A2(i,j,k)*x(i,j-1,k)
             - A3(i,j,k)*A(i-1,j,k)
             - A5(i,j,k)*A(i+1,j,k)
             - A6(i,j,k)*A(i,j+1,k)
             - A7(i,j,k)*A(i,j,k+1) )
            /A4(i,j,k) - x(i,j,k)
    x(i,j,k) = x(i,j,k) + ω*r
```



3. 並列用SOR法

- SOR法が並列化不可の理由(2次元FDM)

x(i,j)の計算に
x(i-1,j),x(i+1,j)
x(i,j-1),x(i,j+1)
の4点が関与
iの順でもjの順でも直前に計算したものを利用

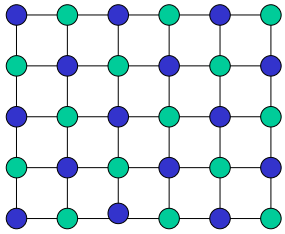


7

3.1 Even-Odd SOR法

- Even-Odd SOR法の計算順序(2次元)

●と●
を交互に
計算する



8

3.2 並列SOR法プログラム(2次元)

```
for j=1,ny
  for i=mod(j-1,2)+1,nx,2
    x(i,j) = 差分計算式( ●の格子点)

for j=1,ny
  for i=mod(j,2)+1,nx,2
    x(i,j) = 差分計算式( ●の格子点)
```

9

4. CG法の導出と性質

- 導出条件(k段からk+1段への反復)

(1) 下記2ステップで計算

$x_{k+1} = x_k + \alpha_k p_k$

$p_{k+1} = r_{k+1} + \beta_k p_k$

(2) αの導出

$F(x_{k+1})=F(r_{k+1})=(r_{k+1},A^{-1}r_{k+1})$を最小

(3) βの導出

$(p_{k+1},Ap_k)=0$、即ちp_{k+1}とp_kはA直交

10

4.1 α計算式の算出

$$
\begin{aligned}
F(r_{k+1}) &= (b - Ax_{k+1}, A^{-1}(b - Ax_{k+1})) \\
&= (b, A^{-1}b) - 2(b, x_{k+1}) + (x_{k+1}, Ax_{k+1}) \\
&= (b, A^{-1}b) - 2(b, x_k) + 2\alpha_k(b, p_k) + (x_k, Ax_k) \\
&\quad + 2\alpha_k(p_k, Ax_k) + \alpha_k^2(p_k, Ap_k)
\end{aligned}
$$

$F(r_{k+1})$を最小にする、即ち$\partial F(r_{k+1})/\partial\alpha_k = 0$

これから、次式が得られる。

$$-2(b, p_k) + 2(p_k, Ax_k) + 2\alpha_k(p_k, Ap_k) = 0$$

従って、α_kは次のようになる。

$$\alpha_k = \frac{(p_k, b) - (p_k, Ax_k)}{(p_k, Ap_k)} = \frac{(p_k, r_k)}{(p_k, Ap_k)}$$

11

4.2 β計算式の算出

$$
\begin{aligned}
(p_{k+1}, Ap_k) &= (r_{k+1} + \beta_k p_k, Ap_k) \\
&= (r_{k+1}, Ap_k) + \beta_k(p_k, Ap_k) \\
&= 0
\end{aligned}
$$

これより、β_kが次のように定まる。

$$\beta_k = \frac{-(r_{k+1}, Ap_k)}{(p_k, Ap_k)}$$

12

4.3 CG法の性質

- 各種の直交性

$(p_i, Ap_j) = 0 \quad (i \neq j)$; 共役直交性

$(r_i, r_j) = 0 \quad (i \neq j)$; 残差の直交性

$(r_i, Ap_j) = 0 \quad (i \neq j, i \neq j+1)$; 三重対角性

- pとrの関係

$$(p_i, r_j) = \begin{cases} 0 & (i < j) \\ (r_j, r_j) & (i \geq j) \end{cases}$$

$$(r_i, Ap_i) = (p_i, Ap_i)$$

13

4.4 n回で収束する理由

- n次元の連立一次方程式Ax=bは数学的には最大n回の反復で収束する。
- その理由は残差ベクトルrが

$(r_i, r_j) = 0 \quad i \neq j$

となるため。n次元のベクトルは n本以上の1次独立なベクトルは存在しない。

- ただし、この仮定は数値計算では丸め誤差のため成立しない。

14

4.5 CG法の計算式(No.1)

初期値x_0を用意する.

$p_0 = r_0 = b - Ax_0$

$k = 0,1,2,\cdots$と収束するまで反復する.

$$\alpha_k = (p_k, r_k)/(p_k, Ap_k)$$

$$x_{k+1} = x_k + \alpha_k p_k, \quad r_{k+1} = r_k - \alpha_k Ap_k$$

$$\beta_{k+1} = -(r_{k+1}, Ap_k)/(p_k, Ap_k)$$

$$p_{k+1} = r_{k+1} + \beta_k p_k$$

15

4.6 計算式の一部変更

- 変更後のほうが計算量が少ない
  一方，直交性は変更前の方が良い
- αの計算式

$$\alpha_k = \frac{(p_k, r_k)}{(p_k, Ap_k)} = \frac{(r_k, r_k)}{(p_k, Ap_k)}$$

- βの計算式 $r_{k+1} = r_k - \alpha_k Ap_k$等を使用

$$\beta_k = \frac{-\alpha_k (r_{k+1}, Ap_k)}{\alpha_k (r_k, Ap_k)} = \frac{(r_{k+1}, r_k) - (r_{k+1}, \alpha_k Ap_k)}{(r_k, r_{k+1}) + \alpha_k (r_k, Ap_k)} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}$$

16

4.7 CG法の計算式(No.2)

初期値x_0を用意する.

$p_0 = r_0 = b - Ax_0$

$k = 0,1,2,\cdots$と収束するまで反復する.

$$\alpha_k = (r_k, r_k)/(p_k, Ap)$$

$$x_{k+1} = x_k + \alpha_k p_k, \quad r_{k+1} = r_k - \alpha_k Ap_k$$

$$\beta_{k+1} = (r_{k+1}, r_{k+1})/(r_k, r_k)$$

$$p_{k+1} = r_{k+1} + \beta_k p_k$$

17

5. CG法プログラム (No.1)

反復計算で $Ax = b$の解xを求める.

$p = r = b - Ax$, xは初期値

以下$\|r\|_2 \leq \varepsilon$(収束条件)となるまで反復する.

$$q = Ap, \quad \lambda = (p, q), \quad \alpha = (p, r)/\lambda$$

$$x = x + \alpha p, \quad r = r - \alpha q$$

$$\beta = -(r, q)/\lambda$$

$$p = r + \beta p,$$

18

5.1 CG法プログラム (No.2)

反復計算で $Ax = b$ の解 x を求める.

$p = r = b - Ax$, xは初期値

$\lambda_0 = (r, r)$

以下 $\|r\|_2 \le \varepsilon$(収束条件)となるまで反復する.

$q = Ap$, $\alpha = \lambda_0 / (p, q)$

$x = x + \alpha p$, $r = r - \alpha q$

$\lambda_1 = (r, r)$, $\beta = \lambda_1 / \lambda_0$

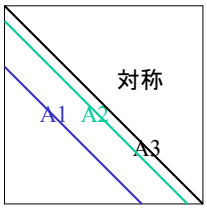
$p = r + \beta p$, $\lambda_0 = \lambda_1$

19

5. 2 q=Apの計算(2次元)

- q=Apの計算

```
for j=1,ny
 for i=1,nx
  q(i,j)=A1(i,j)*p(i,j-1)
       +A2(i,j)*p(i-1,j)
       +A3(i,j)*p(i,j)
       +A2(i+1,j)*p(i+1,j)
       +A1(i,j+1)*p(i,j+1)
```

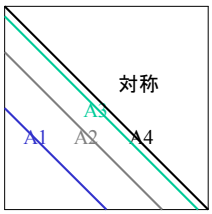


20

5. 3 q=Apの計算(3次元)

- q=Apの計算

```
for k=1,nz
 for j=1,ny
  for i=1,nx
   q(i,j,k)=A1(i,j,k)*p(i,j,k-1)
         +A2(i,j,k)*p(i,j-1,k)
         +A3(i,j,k)*P(i-1,j,k)
         +A4(i,j,k)*P(i,j,k)
         +A3(i+1,j,k)*p(i+1,j,k)
         +A2(i,j+1,k)*p(i,j+1,k)
         +A1(i,j,k+1)*P(i,j,k+1)
```



21

6. ICCG法

- 今回省略
- 必要に応じて、作成し説明する

22

7. MICCG法

- 今回省略
- 必要に応じて作成し説明する

23