一次元熱伝導方程式の差分法による解法

2007年4月18日

中島 研吾

並列計算プログラミング（616-2057）・先端計算機演習I（616-4009）

一次元熱伝導方程式(1/7)

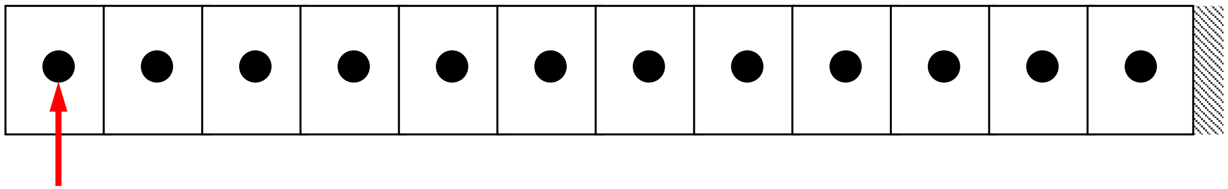
支配方程式:簡単のため熱伝導率=1

$$\frac{d^2\phi}{dx^2} + BF = 0, \quad \phi = 0 @ x = 0, \frac{d\phi}{dx} = 0 @ x = x_{\max}$$

$$\phi = -\frac{1}{2} BF\, x^2 + BF\, x_{\max}\, x$$

φ=0 断熱

実際は以下のような離散化をしているので注意が必要



断熱となっているのはこの面，しかし温度は計算されない

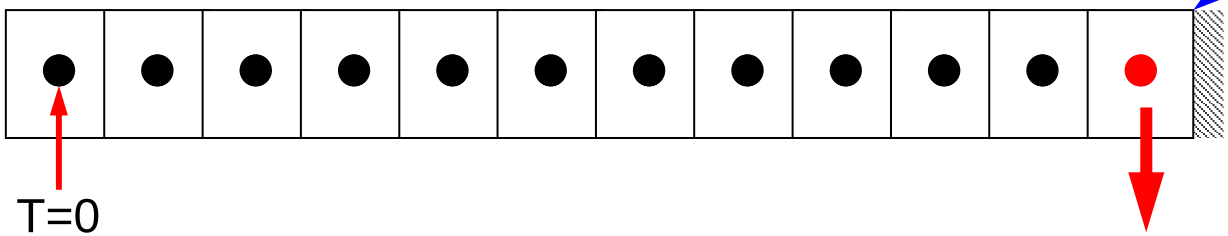
φ=0

一次元熱伝導方程式(2/7)

解析解

$$\phi = -\frac{1}{2} BF\, x^2 + BF\, x_{max}\, x$$

断熱となっているのはこの面，しかし温度は計算されない（$X=X_{max}$）。



T=0
X=0

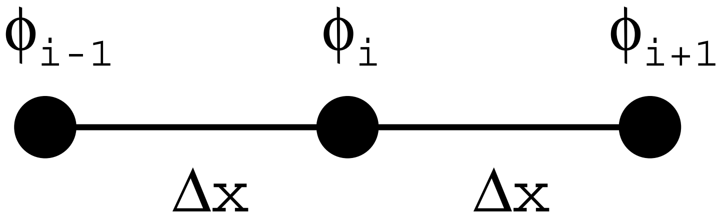
Δx=1.d0，メッシュ数=50，とすると，X_{max}=49.5，
●の点のX座標は49.0となる。BF=1.0d0とすると●での温度は：

$$\phi = -\frac{1}{2} 49^2 + 49.5 \times 49 = -1200.5 + 9850.5 = 1225$$

念のため・・・・差分について

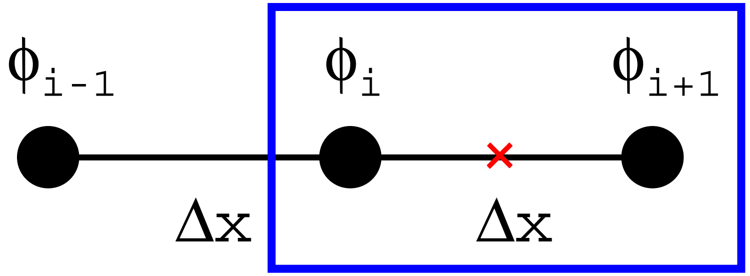
- 差分法：Finite Difference Method
- マクロな微分
  - 微分係数を数値的に近似する手法

- 以下のような一次元系を考える



直感的・・・というか安易な定義

- ×（iとi+1の中点）における微分係数



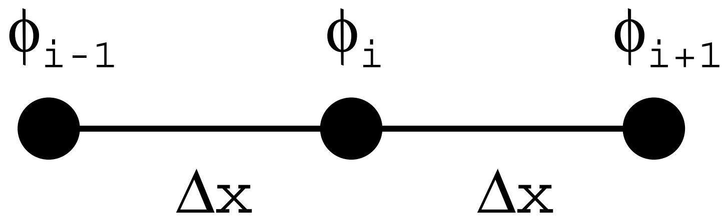
$$\left(\frac{d\phi}{dx}\right)_{i+1/2} \approx \frac{\phi_{i+1}-\phi_i}{\Delta x}$$

Δx→0となると微分係数の定義そのもの

- iにおける二階微分係数

$$\left(\frac{d^2\phi}{dx^2}\right)_i \approx \frac{\left(\frac{d\phi}{dx}\right)_{i+1/2}-\left(\frac{d\phi}{dx}\right)_{i-1/2}}{\Delta x} = \frac{\frac{\phi_{i+1}-\phi_i}{\Delta x}-\frac{\phi_i-\phi_{i-1}}{\Delta x}}{\Delta x} = \frac{\phi_{i+1}-2\phi_i+\phi_{i-1}}{\Delta x^2}$$

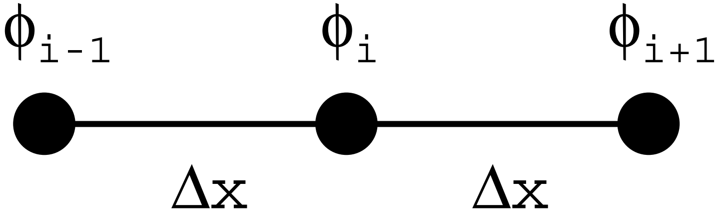
厳密な定義：Taylor展開（1/3）



$$\phi_{i+1} = \phi_i + \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i + \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

$$\phi_{i-1} = \phi_i - \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i - \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

厳密な定義：Taylor展開(2/3)



前進差分

$$\phi_{i+1} = \phi_i + \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i + \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

$$\frac{\phi_{i+1} - \phi_i}{\Delta x} = \left(\frac{\partial \phi}{\partial x} \right)_i + \underline{\frac{(\Delta x)}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i + \frac{(\Delta x)^2}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots}$$

打ち切り誤差が
Δxのオーダー
（一次精度）

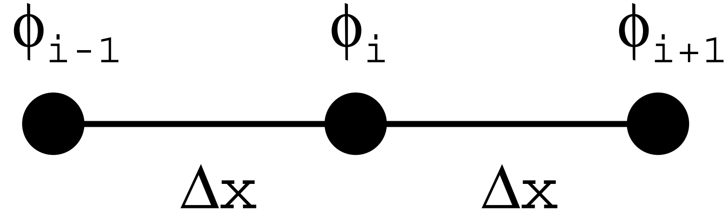
後退差分

$$\phi_{i-1} = \phi_i - \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i - \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

$$\frac{\phi_i - \phi_{i-1}}{\Delta x} = \left(\frac{\partial \phi}{\partial x} \right)_i + \underline{\frac{(\Delta x)}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i + \frac{(\Delta x)^2}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots}$$

打ち切り誤差が
Δxのオーダー
（一次精度）

厳密な定義：Taylor展開（3/3）



中央差分，中心差分

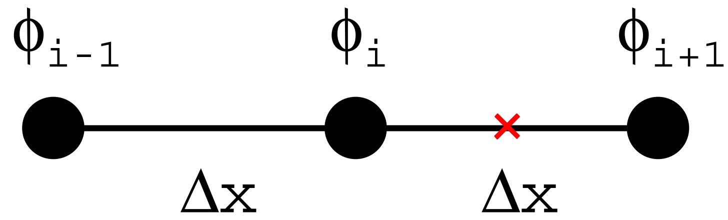
$$\phi_{i+1} = \phi_i + \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i + \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

$$\phi_{i-1} = \phi_i - \Delta x \left(\frac{\partial \phi}{\partial x} \right)_i + \frac{(\Delta x)^2}{2!} \left(\frac{\partial^2 \phi}{\partial x^2} \right)_i - \frac{(\Delta x)^3}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots$$

$$\frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x} = \left(\frac{\partial \phi}{\partial x} \right)_i + \underline{\frac{2 \times (\Delta x)^2}{3!} \left(\frac{\partial^3 \phi}{\partial x^3} \right)_i \cdots}$$

打ち切り誤差が
（Δx）^{2}のオーダー
（二次精度）

安易な定義：実は二次精度だった



$$\phi_{i+1} = \phi_{i+1/2} + \Delta x/2\left(\frac{\partial \phi}{\partial x}\right)_{i+1/2} + \frac{(\Delta x/2)^2}{2!}\left(\frac{\partial^2 \phi}{\partial x^2}\right)_{i+1/2} + \frac{(\Delta x/2)^3}{3!}\left(\frac{\partial^3 \phi}{\partial x^3}\right)_{i+1/2} \cdots$$

$$\phi_{i} = \phi_{i+1/2} - \Delta x/2\left(\frac{\partial \phi}{\partial x}\right)_{i+1/2} + \frac{(\Delta x/2)^2}{2!}\left(\frac{\partial^2 \phi}{\partial x^2}\right)_{i+1/2} - \frac{(\Delta x/2)^3}{3!}\left(\frac{\partial^3 \phi}{\partial x^3}\right)_{i+1/2} \cdots$$

$$\frac{\phi_{i+1} - \phi_i}{\Delta x} = \left(\frac{\partial \phi}{\partial x}\right)_{i+1/2} + \underline{\frac{2\times(\Delta x/2)^2}{3!}\left(\frac{\partial^3 \phi}{\partial x^3}\right)_{i+1/2}} \cdots$$

打ち切り誤差が
$(\Delta x)^2$のオーダー
（二次精度）

二点間の中点で二次精度，それ以外の点では一次精度・・・ということもできる。
Δxが均一でない場合も同様のことが起こる。

一次元熱伝導方程式（3/7）
連立一次方程式の解法：古典的反復法（1）

- 差分法による離散化

$$\left(\frac{d^2\phi}{dx^2}\right)_i \approx \frac{\left(\frac{d\phi}{dx}\right)_{i+1/2} - \left(\frac{d\phi}{dx}\right)_{i-1/2}}{\Delta x} = \frac{\frac{\phi_{i+1}-\phi_i}{\Delta x} - \frac{\phi_i - \phi_{i-1}}{\Delta x}}{\Delta x} = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}$$

- 各要素における線形方程式は以下のような形になる

$$A_L(i)\times\phi_{i-1} + A_D(i)\times\phi_i + A_R(i)\times\phi_{i+1} = BF(i)$$

$$A_L(i) = \frac{1}{\Delta x^2},\ A_D(i) = -\frac{2}{\Delta x^2},\ A_R(i) = \frac{1}{\Delta x^2}$$

一次元熱伝導方程式（4/7）
連立一次方程式の解法：古典的反復法（2）

各要素における線形方程式

$$A_L(i)\times\phi_{i-1} + A_D(i)\times\phi_i + A_R(i)\times\phi_{i+1} = BF(i)$$

$$A_L(i) = \frac{1}{\Delta x^2},\ A_D(i) = -\frac{2}{\Delta x^2},\ A_R(i) = \frac{1}{\Delta x^2}$$

解

$$\phi(i) = \frac{BF(i) - A_L(i)\times\phi_{i-1} - A_R(i)\times\phi_{i+1}}{A_D(i)}$$

- 各点において`φ(i)`の値の変化が無くなるまで反復を繰り返す。
  - 収束：convergence
- 前の反復における解を `φ0(i)`とする

N=8の場合の行列の係数

自分とその周囲のみに非ゼロ成分:疎行列

	1	2	3	4	5	6	7	8
1	AD1	AR1						
2	AL2	AD2	AR2					
3		AL3	AD3	AR3				
4			AL4	AD4	AR4			
5				AL5	AD5	AR5		
6					AL6	AD6	AR6	
7						AL7	AD7	AR7
8							AL8	AD8

$$A_L(i)\times\phi(i-1)+A_D(i)\times\phi(i)+A_R(i)\times\phi(i+1)=BF(i)$$

$$A_L(i)=\frac{1}{\Delta x^2},\ A_D(i)=-\frac{2}{\Delta x^2},\ A_R(i)=\frac{1}{\Delta x^2}$$

一次元熱伝導方程式(5/7)
連立一次方程式の解法:古典的反復法(3)

- ヤコビ法(Jacobi)
  - ϕ_{i-1}およびϕ_{i+1}の値として前の反復における **φ0(i-1),φ0(i+1)** を使用する。
  - 収束は遅い。

$$\phi(i) = \frac{BF(i) - A_L(i) \times \phi 0(i-1) - A_R(i) \times \phi 0(i+1)}{A_D(i)}$$

- ガウス=ザイデル(Gauss-Seidel)法
  - 計算の終了した値 ϕ_{i-1}については最新の値 **φ(i-1)**, ϕ_{i+1}の値としては前の反復における値 **φ0(i+1)** を使用する。
  - Jacobi法より速い。

$$\phi(i) = \frac{BF(i) - A_L(i) \times \phi(i-1) - A_R(i) \times \phi 0(i+1)}{A_D(i)}$$

一次元熱伝導方程式(6/7)
連立一次方程式の解法:古典的反復法(4)

- SOR(Successive-Over Relaxation)法
  - Gauss-Seidel法によって求められた解を ϕ_{GS} とすると, $\phi(i)=\phi 0(i)+\omega(\phi_{GS}-\phi 0(i))$

$$\phi_{GS}(i)=\frac{BF(i)-A_L(i)\times\phi(i-1)-A_R(i)\times\phi 0(i+1)}{A_D(i)}$$

$$\phi_{SOR}(i)=\phi 0(i)+\omega\times\left[\phi_{GS}(i)-\phi 0(i)\right]$$

一次元熱伝導方程式(7/7)
連立一次方程式の解法:古典的反復法(5)

- SOR(Successive-Over Relaxation)法(続き)
  - **ω=1**の場合はGauss-Seidelと同じ，**ω>1**の場合を過緩和(over relaxation)，**ω<1**の場合を不足緩和(under relaxation)と呼ぶ。通常 **1<ω<2** の値を使用する。
  - **ω>1**とすることによって収束が加速される場合がある。
    - 値が大きすぎると発散する場合がある。
  - 一次元線形問題における最適値(メッシュ数=N)，Nが大きくなると2に近づく:境界条件等によって変わる

$$\varpi_{opt} \approx \frac{2}{1+\sin\left(\pi / N\right)}$$

サンプルコード：一次元熱伝導方程式
intro.tarの中にある

FORTRAN

```
heat_jacobi.f
heat_gs.f
heat_sor.f
```

C

```
heat_jacobi.c
heat_gs.c
heat_sor.c
```

input.dat, cinput.dat

50	N	メッシュ数
1.d0 1.d0	dx, BF	メッシュ幅，体積発熱量
50000	ITERmax	最大反復回数
1.d-7 -1.00	EPS, OMEGA	打切誤差， SORのω

実行例：Jacobi法（./ja）

```
1000     iters,   RESID=   3.911949E-01      PHI(N)=  4.724513E+02
2000     iters,   RESID=   2.350935E-01      PHI(N)=  7.746137E+02
3000     iters,   RESID=   1.406316E-01      PHI(N)=  9.555996E+02
...
29000    iters,   RESID=   2.213721E-07      PHI(N)=  1.225000E+03
30000    iters,   RESID=   1.324140E-07      PHI(N)=  1.225000E+03
30548    iters,   RESID=   9.991504E-08      PHI(N)=  1.225000E+03

1        0.000000E+00      0.000000E+00
2        4.899999E+01      4.900000E+01
3        9.699999E+01      9.700000E+01
4        1.440000E+02      1.440000E+02
5        1.900000E+02      1.900000E+02
...

41       1.180000E+03      1.180000E+03
42       1.189000E+03      1.189000E+03
43       1.197000E+03      1.197000E+03
44       1.204000E+03      1.204000E+03
45       1.210000E+03      1.210000E+03
46       1.215000E+03      1.215000E+03
47       1.219000E+03      1.219000E+03
48       1.222000E+03      1.222000E+03
49       1.224000E+03      1.224000E+03
50       1.225000E+03      1.225000E+03
```

反復回数
最大残差
$\phi(50)$

数値解，解析解

$$\phi = -\frac{1}{2}49^2 + 49.5 \times 49 = -1200.5 + 9850.5 = 1225$$

実行例：Gauss-Seidel法（./gs）

```
1000    iters,  RESID=  4.591084E-01    PHI(N)=  7.785284E+02
2000    iters,  RESID=  1.642708E-01    PHI(N)=  1.065259E+03
3000    iters,  RESID=  5.877313E-02    PHI(N)=  1.167848E+03
...
14000   iters,  RESID=  7.227382E-07    PHI(N)=  1.224999E+03
15000   iters,  RESID=  2.585828E-07    PHI(N)=  1.225000E+03
15925   iters,  RESID=  9.993005E-08    PHI(N)=  1.225000E+03

1       0.000000E+00    0.000000E+00
2       4.899999E+01    4.900000E+01
3       9.699999E+01    9.700000E+01
4       1.440000E+02    1.440000E+02
5       1.900000E+02    1.900000E+02
...

41      1.180000E+03    1.180000E+03
42      1.189000E+03    1.189000E+03
43      1.197000E+03    1.197000E+03
44      1.204000E+03    1.204000E+03
45      1.210000E+03    1.210000E+03
46      1.215000E+03    1.215000E+03
47      1.219000E+03    1.219000E+03
48      1.222000E+03    1.222000E+03
49      1.224000E+03    1.224000E+03
50      1.225000E+03    1.225000E+03
```

反復回数
最大残差
$\phi(50)$

数値解，解析解

実行例：SOR法（./sor）

```
### OMEGA=     1.881838E+00
1000 iters, RESID=    4.921991E-07 PHI(N)=      1.225000E+03
1091 iters, RESID=    9.923362E-08 PHI(N)=      1.225000E+03

1        0.000000E+00     0.000000E+00
2        4.899999E+01     4.900000E+01
3        9.699999E+01     9.700000E+01
4        1.440000E+02     1.440000E+02
5        1.900000E+02     1.900000E+02
...

41       1.180000E+03     1.180000E+03
42       1.189000E+03     1.189000E+03
43       1.197000E+03     1.197000E+03
44       1.204000E+03     1.204000E+03
45       1.210000E+03     1.210000E+03
46       1.215000E+03     1.215000E+03
47       1.219000E+03     1.219000E+03
48       1.222000E+03     1.222000E+03
49       1.224000E+03     1.224000E+03
50       1.225000E+03     1.225000E+03
```

OMEGA（この場合は最適値）

反復回数
最大残差
$\phi(50)$

数値解，解析解

一次元熱伝導方程式：Jacobi法 heat_jacobi.f（1/3）

```
!C
!C   1D Poisson Equation Solver by
!C   Jacobi Method
!C
!C   d/dx(dPHI/dx) + BF = 0
!C   PHI=0@x=0
!C
      program JACOBI_poi
      implicit REAL*8 (A-H,O-Z)

      integer :: N, ITERmax
      real(kind=8) :: dx, RESID, dPHI, dPHImax, BF, EPS
      real(kind=8), dimension(:), allocatable :: PHI, RHS, PHI0
      real(kind=8), dimension(:), allocatable :: rAD, AR, AL

!C
!C-- INIT.
      open  (11, file='input.dat', status='unknown')
       read (11,*) N
       read (11,*) dX, BF
       read (11,*) ITERmax
       read (11,*) EPS
      close (11)
```

一次元熱伝導方程式：Jacobi法 heat_jacobi.f（2/3）

```
      allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

       PHI = 0.d0
       PHI0= 0.d0

       AR = 1.d0/dX
       AL = 1.d0/dX
      rAD = 1.d0/(-2.d0/dX)
      RHS=  -BF * dX

       AL (1)= 0.d0
      rAD (1)= 1.d0
       RHS(1)= 0.d0

       AR (N)= 0.d0
      rAD (N)= 1.d0/(-1.d0/dX)
```

$$\frac{d^2\phi}{dx^2} + BF = 0, \quad \phi = 0 @ x = 0, \frac{d\phi}{dx} = 0 @ x = x_{\max}$$

一次元熱伝導方程式：Jacobi法 heat_jacobi.f（2/3）

```
allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

 PHI = 0.d0
 PHI0= 0.d0

 AR = 1.d0/dX
 AL = 1.d0/dX
rAD = 1.d0/(-2.d0/dX)
RHS=  -BF * dX

 AL (1)= 0.d0
rAD (1)= 1.d0
 RHS(1)= 0.d0

 AR (N)= 0.d0
rAD (N)= 1.d0/(-1.d0/dX)
```

$$\left(\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}\right)\times V + BF \times V = 0$$

$$\left(\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}\right)\times \Delta x + BF \times \Delta x = 0$$

一次元熱伝導方程式：Jacobi法 heat_jacobi.f（2/3）

```
      allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

       PHI = 0.d0
       PHI0= 0.d0

       AR = 1.d0/dX
       AL = 1.d0/dX
      rAD = 1.d0/(-2.d0/dX)
      RHS=  -BF * dX

       AL (1)= 0.d0
      rAD (1)= 1.d0
       RHS(1)= 0.d0

       AR (N)= 0.d0
      rAD (N)= 1.d0/(-1.d0/dX)
```

$$rA_D(i) = \frac{1}{A_D(i)}$$

割り算は計算時間がかかるため

$$\left(\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}\right) \times \Delta x + BF \times \Delta x = 0$$

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x} = \underbrace{\left(\frac{1}{\Delta x}\right)}_{\text{AL}}\phi_{i-1} - \underbrace{\left(\frac{2}{\Delta x}\right)}_{\text{AD}}\phi_i + \underbrace{\left(\frac{1}{\Delta x}\right)}_{\text{AR}}\phi_{i+1} = \underbrace{-BF \times \Delta x}_{\text{RHS}}$$

一次元熱伝導方程式：Jacobi法 heat_jacobi.f（2/3）：境界条件の処理

```
AR  (N)=  0.d0
rAD(N)=  1.d0/(-1.d0/dX)
```

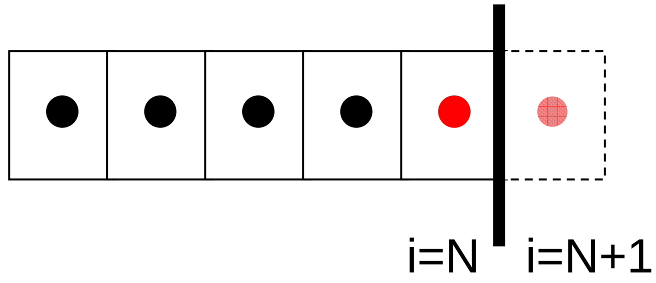
$$\frac{d\phi}{dx}=0 @ x=x_{\max} \Rightarrow \frac{\phi_{N+1}-\phi_N}{\Delta x}=0$$

$$\left(\frac{\phi_{N+1}-2\phi_N+\phi_{N-1}}{\Delta x^2}\right)\times\Delta x+BF\times\Delta x=0$$

$$\Rightarrow\left(\frac{-\phi_N+\phi_{N-1}}{\Delta x^2}\right)\times\Delta x+BF\times\Delta x=0$$

$$\Rightarrow(0)\phi_{N+1}+\left(\frac{-1}{\Delta x}\right)\phi_N+\left(\frac{1}{\Delta x}\right)\phi_{N-1}=-BF\times\Delta x$$

AL **AD** **AR** **RHS**



境界面で断熱条件が成立するためには，$\phi_{N+1}=\phi_N$を満たすような仮想的な要素があると都合が良い

一次元熱伝導方程式:Jacobi法 heat_jacobi.f（3/3）

```
!C
!C-- ITERATIONs
      do iter= 1, ITERmax
        dPHImax= -1.d0
        do i= 2, N
          RESID  = RHS(i) - AL(i)*PHI0(i-1) - AR(i)*PHI0(i+1)
          dPHI   = RESID*rAD(i) - PHI0(i)
          dPHImax= dmax1 (dabs(dPHI), dPHImax)
          PHI(i) = PHI0(i) + dPHI
        enddo

        do i= 2, N
          PHI0(i) = PHI(i)
        enddo

        if (dPHImax.lt.EPS) exit
      enddo

!C
!C-- OUTPUT
      Xmax= (dfloat(N-1)+0.5d0)*dX
      do i= 1, N
        XX= dfloat(i-1)*dX
        T = -0.5d0*BF*(XX**2) + BF*Xmax*XX
        write (*,'(i8, 2(1pe16.6))') i, PHI(i), T
      enddo

      end program JACOBI_poi
```

$$\phi(i) = \frac{RHS(i) - A_L(i)\times\phi 0(i-1) - A_R(i)\times\phi 0(i+1)}{A_D(i)}$$

$$= \left[\frac{RHS(i) - A_L(i)\times\phi 0(i-1) - A_R(i)\times\phi 0(i+1)}{A_D(i)} - \phi 0(i)\right] + \phi 0(i)$$

$$T = \phi(analy.) = -\frac{1}{2} BF\, x^2 + BF\, x_{\max}\, x$$

一次元熱伝導方程式：SOR法 heat_sor.f（1/3）

```
!C
!C   1D Poisson Equation Solver by
!C   SOR (Successive Over Relaxation) Method
!C
!C   d/dx(dPHI/dx) + BF = 0
!C   PHI=0@x=0
!C
      program SOR_poi
      implicit REAL*8 (A-H,O-Z)

      integer :: N, ITERmax
      real(kind=8) :: dx, RESID, dPHI, dPHImax, BF, EPS
      real(kind=8), dimension(:), allocatable :: PHI, RHS, PHI0
      real(kind=8), dimension(:), allocatable :: rAD, AR, AL

!C
!C-- INIT.
      open  (11, file='input.dat', status='unknown')
       read (11,*) N
       read (11,*) dX, BF
       read (11,*) ITERmax
       read (11,*) EPS, OMEGA
      close (11)

      if (OMEGA.le.0.d0) then
        PI= 4.d0 * atan(1.d0)
        OMEGA= 2.d0/(1.d0+dsin(PI/dfloat(N)))
      endif
```

一次元熱伝導方程式：SOR法 heat_sor.f（1/3）

```
!C
!C   1D Poisson Equation Solver by
!C   SOR (Successive Over Relaxation) Method
!C
!C   d/dx(dPHI/dx) + BF = 0
!C   PHI=0@x=0
!C
      program SOR_poi
      implicit REAL*8 (A-H,O-Z)

      integer :: N, ITERmax
      real(kind=8) :: dx, OMEGA, RESID, dPHI, dPHImax, BF, EPS
      real(kind=8), dimension(:), allocatable :: PHI, RHS
      real(kind=8), dimension(:), allocatable :: rAD, AR, AL

!C
!C-- INIT.
      open  (11, file='input.dat', status='unknown')
       read (11,*) N
       read (11,*) dX, BF
       read (11,*) ITERmax
       read (11,*) EPS, OMEGA
      close (11)

      if (OMEGA.le.0.d0) then
        PI= 4.d0 * atan(1.d0)
        OMEGA= 2.d0/(1.d0+dsin(PI/dfloat(N)))
      endif
```

ωの最適値
（<0.0が入力された場合）

一次元熱伝導方程式：SOR法 heat_sor.f（2/3）

```
      allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

       PHI = 0.d0
       PHI0= 0.d0

       AR = 1.d0/dX
       AL = 1.d0/dX
      rAD = 1.d0/(-2.d0/dX)
      RHS=  -BF * dX

       AL (1)= 0.d0
      rAD (1)= 1.d0
       RHS(1)= 0.d0

       AR (N)= 0.d0
      rAD (N)= 1.d0/(-1.d0/dX)
```

$$\frac{d^2\phi}{dx^2} + BF = 0, \quad \phi = 0 @ x = 0, \frac{d\phi}{dx} = 0 @ x = x_{\max}$$

一次元熱伝導方程式：SOR法 heat_sor.f（2/3）

```
      allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

       PHI = 0.d0
       PHI0= 0.d0

       AR = 1.d0/dX
       AL = 1.d0/dX
      rAD = 1.d0/(-2.d0/dX)
      RHS=  -BF * dX

       AL (1)= 0.d0
      rAD (1)= 1.d0
       RHS(1)= 0.d0

       AR (N)= 0.d0
      rAD (N)= 1.d0/(-1.d0/dX)
```

$$\left(\frac{\phi_{i+1}-2\phi_i+\phi_{i-1}}{\Delta x^2}\right)\times V+BF\times V=0$$

$$\left(\frac{\phi_{i+1}-2\phi_i+\phi_{i-1}}{\Delta x^2}\right)\times \Delta x+BF\times \Delta x=0$$

一次元熱伝導方程式：SOR法 heat_sor.f （2/3）

```
allocate (PHI(N+1), rAD(N), AR(N), AL(N), RHS(N), PHI0(N))

 PHI = 0.d0
 PHI0= 0.d0

 AR = 1.d0/dX
 AL = 1.d0/dX
rAD = 1.d0/(-2.d0/dX)
RHS=  -BF * dX

 AL (1)= 0.d0
rAD (1)= 1.d0
 RHS(1)= 0.d0

 AR (N)= 0.d0
rAD (N)= 1.d0/(-1.d0/dX)
```

$$rA_D(i) = \frac{1}{A_D(i)}$$

割り算は計算時間がかかるため

$$\left(\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}\right) \times \Delta x + BF \times \Delta x = 0$$

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x} = \underbrace{\left(\frac{1}{\Delta x}\right)}_{\mathbf{AL}}\phi_{i-1} - \underbrace{\left(\frac{2}{\Delta x}\right)}_{\mathbf{AD}}\phi_i + \underbrace{\left(\frac{1}{\Delta x}\right)}_{\mathbf{AR}}\phi_{i+1} = \underbrace{-BF \times \Delta x}_{\mathbf{RHS}}$$

一次元熱伝導方程式：SOR法 heat_sor.f（3/3）

```
!C
!C-- ITERATIONs
      do iter= 1, ITERmax
        dPHImax= -1.d0
        do i= 2, N
          RESID   = RHS(i) - AL(i)*PHI(i-1) - AR(i)*PHI(i+1)
          dPHI    = OMEGA * (RESID*rAD(i)-PHI(i))
          dPHImax= dmax1 (dabs(dPHI), dPHImax)
          PHI(i) = PHI(i) + dPHI
        enddo

        if (dPHImax.lt.EPS) exit
      enddo

!C
!C-- OUTPUT
      Xmax= (dfloat(N-1)+0.5d0)*dX
      do i= 1, N
        XX= dfloat(i-1)*dX
        T = -0.5d0*BF*(XX**2) + BF*Xmax*XX
        write (*,'(i8, 2(1pe16.6))') i, PHI(i), T
      enddo

      end program SOR_poi
```

$$\phi_{GS}(i) = \frac{BF(i) - A_L(i)\times\phi(i-1) - A_R(i)\times\phi 0(i+1)}{A_D(i)}$$

$$\phi_{SOR}(i) = \phi 0(i) + \omega\times\left[\phi_{GS}(i) - \phi 0(i)\right]$$

JacobiとSOR

JACOBI

```
      do iter= 1, ITERmax
        do i= 2, N
          RESID   = RHS(i) - AL(i)*PHI0(i-1) - AR(i)*PHI0(i+1)
          dPHI    = RESID*rAD(i) - PHI0(i)
          PHI(i)  = PHI0(i) + dPHI
        enddo

        do i= 2, N
          PHI0(i)  = PHI(i)
        enddo
      enddo
```

反復の間φの値は不変(φ0)

SOR

```
      do iter= 1, ITERmax
        do i= 2, N
          RESID   = RHS(i) - AL(i)*PHI(i-1) - AR(i)*PHI(i+1)
          dPHI    = OMEGA * (RESID*rAD(i)-PHI(i))
          PHI(i)  = PHI(i) + dPHI
        enddo
      enddo
```

反復の間φの値は常に最新値を使用

動作確認：もしできたら

- 各プログラム をコンパイル，ランさせる。
- 「input.dat」，「cinput.dat」を変更して ./sor 実行
  - OMEGA=1.00とした場合の収束回数が ./gs の場合と一致することを確認
  - OMEGAを1.00から増加させると，収束回数が減少することを確認
  - OMEGAが最適値（OMEGA<0とした場合）を上回る場合の収束回数
    - 実際はOMEGA=1.94程度が最適値