# 数値計算法
# 物理学実験 III(1994)

C 版第 1 版 (1994.10)
from PASCAL 版改訂第 3 版 (1990.4.7)
& 同改訂第 4 版 (1992.10.20)
©富阪幸治 (1990,1992,1994,1996)

目次

(0) この実験を通じて学ぶこと
(1) はじめに
  1.1 プログラム言語について
  1.2 まず小手調べ (連接)
  1.3 関数と分岐
  1.4 再帰呼び出し
(2) 常微分方程式の数値解法
  2.1 Euler 法
  2.2 繰り返し計算
  2.3 修正 Euler 法
  2.4 ルンゲ- クッタ法
(3) 偏微分方程式の数値解法
  3.1 熱伝導方程式
  3.2 陽解法
  3.3 境界条件と初期条件
  3.4 繰り返しと配列型
  3.5 Poisson 方程式
(4) 連立方程式の解法
  4.1 ガウスの消去法
(5) 数値積分
  5.1 台形則
  5.2 無限区間の積分
(付録)
  1 第 3 節への準備- 配列型
  2 グラフィクスの使い方
注　教科書 :Turbo C プログラミング (サイエンス社) 玉井浩著

# 0 ．この実験を通じて学ぶこと

この実験を通じて学ぶことは，3 点ある．

- 数値計算のいくつかの方法についてそれを身につける．
- プログラムによって，その方法を表現することが出来るようになる．
- データを読む，プログラムを書く，実行する，などの基本的動作をパソコンを使って行え、ＭＳ－ＤＯＳなどのオペレーティング・システムについても基本的動作については理解できるようになる．

# 1 ．初めに

ある方程式を解いて答を得ようとする場合に，答が「手」で求められることは，そんなに多くはない．我々のまわりの物理現象は，微分方程式で，表わされることが多いが，この解が，いつも既知の関数で表わされるわけではない．このような時，計算機を用いて解を得ようとする方法を，数値計算法と呼ぶ．もちろん $d^2y/dx^2 = -y$の解が $y = a\sin x + b\cos x$ と計算機で求められるのではなく，$x =$なになにの時 $y =$なになにと，数値で求められる，と言う意味である．

## 1 ．1　プログラム言語について

数値計算法自体は，「方法」であるから，数学的に記述できるが，実際に計算機に仕事をさせるとなると，計算機にわかる「記述法」で問題を記述してやる必要がある，これを，プログラム言語という．

ここでは，C という言語を，用いることにする．まず始めに，この C について学びながら，Fortran，BASIC 等，どの言語でも共通のプログラムの基本的構造を学ぶことにする．

## 1 ．2　まず小手調べ (プログラムは順番に)

例題 **1.1**　$a, b$ 2 つの整数をキーボードから入力し，その和，差，積，商を求めて，ディスプレイに出力せよ．

```
#include <stdio.h>
/*  REIDAI 1.1  */
main()
{
  int a,b,wa,sa,seki,shou;
  scanf("%d %d",&a,&b);
  printf("a=%d,  b=%d\n",a,b);
  wa=a+b;
  sa=a-b;
  seki=a*b;
  shou=a/b;
  printf("Wa=%d,  Sa=%d,  Seki=%d,  Shou=%d\n",wa,sa,seki,shou);
}
```

重要点

1. プログラムは，main() で始まる．実際の仕事はつぎの { で始まり，} で終わる．
2. #include <stdio.h>は標準の入出力をおこなうために必要な関数原型を読み込むことを指定しており、通常、入出力 (キーボードから数字を読んだり、ディスプレイに文字を書いたりする) のためには必須です。
3. int は変数の性格を示す．int は整数，float は実数，char は文字型．
   変数とは、様々なデータを記憶しておく場所に付いた名前で、電卓のメモリーの名前のようなもの。
4. scanf，printf は読み，書きを行なう関数．
5. =は代入で，右辺を計算して左辺に代入する事を示す．
   数学の等号と意味が異なることに注意。
6. 四則は，+，-，*，/で，また余りを求めるには%を用いる。 (注意) ただし、整数どうしの割り算の結果は切り捨てで整数で与えられる。
7. 変数は、先頭が英字、以降は英数字．
8. 数は、整数は、123、0 のように、実数は 123.4 または 1.234E2 のように書く。
9. この場合は、二つの整数をスペースで区切って並べて入力する。入力するとは、キーボードから打ち込んだあと、リターンキーを押すことである。
10. 様々な名前では大文字と小文字は別のものとして区別する。abc と Abc は異なる名前である。

問題 **1.1.a**　上のプログラムを，ふたつの実数の，和，差，積，商を求めるにはどうすればよいか．

問題 **1.2**　$\frac{(b+c)\times a}{a+b}$を計算して、変数 x に代入する文を書け。ただし、a、b、c は実数を記憶する変数であるとする。

重要事項　ここでみたように，プログラムの基本的構造の一つは順番に実行して行く (連接という) という構造である．このほかに後に出て来るが，ある条件によって処理が変わる (分岐§１．３)，おなじ手順を繰り返す (§２．２) といった基本的構造がある．

## １．３　関数と分岐

例題 **1.3**　実数 $x$ を読み，その 2 乗，3 乗，平方根，サイン，exp を求めよ．

```
#include <stdio.h>
#include <math.h>
/* REIDAI 1.3: Numerical Calculation */
main()
{
  float x,x2,x3,rx,sx,ex;
```

```
  scanf("%f",&x);

  x2=x*x;       /*  square       */
  x3=x*x*x;     /*  cube         */
  rx=sqrt(x);   /*  square root  */
  sx=sin(x);    /*  sine         */
  ex=exp(x);    /*  exponential  */

  printf("%f %f %f %f %f\n",x2, x3, rx, sx, ex);
}
```

注意数学の関数を利用するには#include <math.h> を指定して、数学ライブラリを利用可能にしておかねばならない。この他に，cos(x)，tan(x)，log(x)=$\log_e(x)$，fabs(x) などがある．関数は自分で作ることもできる．

例題 1.4　$x$ の 3 乗を求める関数を作れ．

```
#include <stdio.h>
#include <math.h>
/*  REIDAI 1.4  */
float Cube(float x);
main()
{
  float x,x2,x3,rx,sx,ex;
  scanf("%f",&x);

  x2=x*x;
  x3=Cube(x);  /* 関数 Cube を呼び出している */
  rx=sqrt(x);
  sx=sin(x);
  ex=exp(x);

  printf("%f %f %f %f %f\n",x2, x3, rx, sx, ex);
}

float Cube(float a)     /* ここからが自分で作った関数 */
{
  float a3;
  a3=a*a*a;
  return(a3);
}
```

重要事項

1. 関数は x(y) という形をしている。
2. 使う関数は事前に 「つかうよ」と **float** Cube(**float** x); で宣言する。ここで、実数の 3 乗を求めるので **float** x とし、その結果は、実数で求められるから **float** Cube としている。
3. x の値が、a にコピーされ、その 3 乗が計算されて、a3 に代入される。計算結果 (Cube(x) の値 ) を持ち帰るには、**return**(a3); を用いる。

簡単な問題 **1.4.a:** a3 を使わないで、a の 3 乗を関数 Cube の値とするにはどうすればよいか。

例題 **1.5** $a$ , $b$ の 2 つの整数のうち大きいものを求める関数 max をつくれ .

これには , プログラム構造の分岐を用いる . C で分岐を表現する最も普通の方法は , **if** 文 (教科書 p.24 ) を用いる .

```
if (条件)
  {
    ....; 条件が満足されるときに
    ....; 実行する手順 (文がひとつの時は{と}は不要 )
  }
else
  {
    ....; 条件が満足されないときに
    ....; 実行する手順 (これがないときには else 以下は省略 )
  }
```

したがって , この場合は

```
int max(int x,int y)
{
  int max;
  if (x>y)
    {
    max = x;
    }
  else
    {
    max = y;
    }
  return(max);
}
```

となる .

注意 } の後には ;を付けないこと。

問題 **1.6** 二数 $a$ , $b$ を読み込み , その最大値と最小値を出力するプログラムを作成せよ .

問題 **1.7** これを三数 $a$ , $b$ , $c$ の最大値と最小値を出力するものに変更せよ .

### １．４　再帰呼び出し

関数が自分自身を呼び出す構造になっているものを再帰呼び出しという（教科書 p.78）．ここでは階乗を計算する例を上げる．

例題 **1.8** $n!$を計算する関数を書け．

```
int kaijou(int k)
{
  if (k>1)
    {
    return(kaijou(k-1)*k);
    }
  else
    {
    return(1);
    }
}
```

この場合は，次章に出て来る繰り返しを使うことによってもプログラムできる．しかし，この例のように，再帰呼び出しを用いると非常に簡単にプログラムを書くことが出来る場合がある．

問題 **1.9** 例題を参考にして，$n$ が偶数の時は $n \cdot (n-2) \cdot (n-4) \cdot \cdots \cdot 2$ 及び，$n$ が奇数の時は $n \cdot (n-2) \cdot (n-4) \cdot \cdots \cdot 1$ を求めるプログラムを作成せよ．
ヒント $n$ が偶数か，奇数かは $\mathrm{n} \bmod 2$ が $0$ か $1$ かによって区別できる．

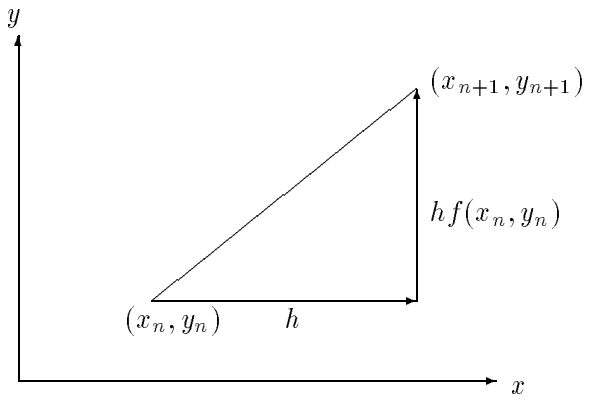
## ２．常微分方程式の数値解法

### ２．１　$Euler$法

一階の微分方程式 $dy/dx = f(x,y)$ を初期条件 $x = x_0$で $y = y_0$のもとに解くことにする。$h$ を「充分」に小さくとると、

$$y(x+h) \approx y(x) + hf(x,y), \tag{2.1}$$

これで、$(x_n, y_n)$ から $(x_{n+1} \equiv x_n + h, y_{n+1})$ の値が求められるので、これを繰り返していけば、微分方程式の解が得られる。




## ２．２　繰り返し計算

前の節から、微分方程式を解く主要な部分は、

```
y=y+f(x,y)*h;
x=x+h;
```

であることがわかった。これを繰り返し実行すれば微分方程式の解が得られることになる。x=x+h という文は、「x に記憶されている中身を h だけ増やして、また x に記憶させる。」という意味であることに注意。

ここで最後の基本的プログラム構造繰り返しがでてきた．C では繰り返しは、

```
while（条件）
  {......;          まず条件を調べ、
   ......;            満足されているなら {} の中の仕事が実行される。
   ......;            満足されていないなら } の次の文に移る。
  }                  これを、繰り返す。
```

という文で記述される。

例題 **2.1** として、１から１００までの整数の和を求めてみよう。

```
/*   Rei 2.1    */
#include <stdio.h>
main()
{
  int i,wa;
  i=1;
  wa=0;
  while (i<=100)
    {
     wa=wa+i;
     i=i+1;
     }
```

```
  printf("Sum of 1,2,...%d=%d\n",i-1,wa);
}
```

ここで、いくつかの簡便な記述法をまとめて覚えておこう。while を含む部分は、次のように書いてもよい。

```
  while (i<=100)
    {
     wa+=i;
     i++;
     }
```

ここで、使われているのは、複合代入演算とインクリメント、デクリメント演算である。

複合代入演算 wa+=i は、wa=wa+i と同じである。同じようにして、a-=b は、a=a-b と同じ意味である。

インクリメント演算とデクリメント演算 i++もしくは++i は、i を 1 増加させることを表す。つまり i=i+1 と同じである。

また、i- -もしくは- -i は、i を 1 減少させることを表す。つまり i=i-1 と同じである。

ただし前置演算子 j=++i と後置演算子 j=i++とは、意味が異なるので注意。つまり、j=++i は、まずインクリメントした後に代入が行なわれ、j=i++では、j=i の代入が行なわれた後、i の値だけがインクリメントされる。

たとえば，n=3 のとき，x=n++; の文の後では，x の値は 3 である (n はもちろん 4 ) が，x=++n; の後では，x の値も 4 になっている．

この二つを利用すると、while を含む部分は、さらに簡便に次のように書いてもよい。

```
  while (i<=100)
    {
     wa+=i++;
     }
```

**練習 2.2**　第 1 章で既に学んだ，階乗をもとめる関数を繰り返しを用いて計算するものに書き直せ．

**課題 2.3**　$dy/dx = x - y$　(この答えは (1) であるが )、これを Euler 法を用いて解け。初期値は $x = 0$ で $y = 0$ とする。

## 2．3　修正 *Euler* 法

Euler 法の計算法を、

$$\tilde{y}_{n+1} = y_n + f(x_n, y_n) \cdot h \qquad y_{n+1} = y_n + \frac{f(x_n, y_n) + f(x_{n+1}, \tilde{y}_{n+1})}{2} \cdot h \tag{2.2}$$

としたものを修正 Euler 法という。

**課題 2.4**　課題 1 をこの方法で解き、 (課題 2.3 との ) 誤差の差を調べよ。

## 2．4　ルンゲ– クッタ法

この Euler 法は、関数の変化が直線的でないと誤差が大きくなる。もう少し誤差の少ない方法としてルンゲ– クッタ法がある。

$$
\begin{aligned}
y_n^{(1)} &= y_n + \frac{h}{2} f_n^{(1)}, \quad & f_n^{(1)} &= f(x_n, y_n), \\
y_n^{(2)} &= y_n + \frac{h}{2} f_n^{(2)}, \quad & f_n^{(2)} &= f(x_n + \frac{h}{2}, y_n^{(1)}), \\
y_n^{(3)} &= y_n + h f_n^{(3)}, \quad & f_n^{(3)} &= f(x_n + \frac{h}{2}, y_n^{(2)}), \\
y_{n+1} &= y_n + \frac{h}{6}(f_n^{(1)} + 2f_n^{(2)} + 2f_n^{(3)} + f_n^{(4)}), \quad & f_n^{(4)} &= f(x_{n+1}, y_n^{(3)})
\end{aligned}
\tag{2.3}
$$

## 2．5　連立微分方程式

連立微分方程式

$$
\begin{aligned}
\frac{dy}{dx} &= f_y(x, y, z) \\
\frac{dz}{dx} &= f_z(x, y, z)
\end{aligned}
\tag{2.4}
$$

を解くには、どうしたらよいだろうか ?修正 Euler 法を例にとると、

$$
\begin{aligned}
\tilde{y}_{n+1} &= y_n + f_y(x_n, y_n, z_n) \cdot h \\
\tilde{z}_{n+1} &= z_n + f_z(x_n, y_n, z_n) \cdot h \\
y_{n+1} &= y_n + \frac{f_y(x_n, y_n, z_n) + f_y(x_{n+1}, \tilde{y}_{n+1}, \tilde{z}_{n+1})}{2} \cdot h \\
z_{n+1} &= z_n + \frac{f_z(x_n, y_n, z_n) + f_z(x_{n+1}, \tilde{y}_{n+1}, \tilde{z}_{n+1})}{2} \cdot h
\end{aligned}
\tag{2.5}
$$

とすればよい。プログラムで書けば、

```
while (x<=xmax}
  {
  y1=y+fy(x,y,z)*h;
  z1=z+fz(x,y,z)*h;
  y2=y+0.5*(fy(x,y,z)+fy(x+h,y1,z1))*h;
  z=z+0.5*(fz(x,y,z)+fz(x+h,y1,z1))*h;
  y=y2;
  x+=h;
  }
```

の様にすればよい。

質問　上のプログラムで一度 $y_2$に新しい $y$の値を代入した後、 2 行下であらためて $y$に代入している。これはなぜか ？

連立微分方程式を用いると、２階以上の微分方程式を解くことができる。$y' = z$と置くと$d^2y/dx^2 = f(x, y, y')$は次のように書き換えられる。

$$\frac{dz}{dx} = f(x, y, z)$$

$$\frac{dy}{dx} = z \tag{2.6}$$

**課題 2.5**　上の置き換えを利用して、$y'' = -y$（答えは言わずと知れた三角関数）を解け。初期条件は$x = 0$で、$y = 0, y' = 1$とせよ。

# ３．偏微分方程式

## ３．１　熱伝導方程式

鉄の棒の片方の端を，突然高温に熱したとする．熱伝導で，その端から他方へ，熱が流れ，しだいに全体が高温になるだろう．このときの温度の変化は，場所場所でどの様に変化するだろうか．これを記述する方程式が熱伝導方程式である．

図のように，棒の長さの方向に，$x$ 軸をとる．単位面積の断面積を通って流れる，熱エネルギーの量 $q$は，熱伝導率 $(\kappa)$ と温度勾配 $(\partial T/\partial x)$ の積で表される．

$$q = -\kappa \frac{\partial T}{\partial x} \tag{3.1}$$

ちなみに鉄の熱伝導率は，$15 \sim 50 \mathrm{Js^{-1}m^{-1}K^{-1}}$である．単位体積当りの熱エネルギーの時間変化は

$$\frac{\partial E}{\partial t} = -\frac{\partial q}{\partial x} \tag{3.2}$$

比熱を $c$ とすると，$\delta E = c\delta T$だから，比熱と熱伝導率が温度に依らずに一定とすると，温度変化の式は，

$$c\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2} \tag{3.3}$$

となる．

$$x' = \frac{x}{L}$$

$$u' = \frac{T}{T_0}$$

$$t' = \frac{t}{(cL^2/\kappa)}$$

と置くと，(３．３）式は，

$$\frac{\partial u'}{\partial t'} = \frac{\partial^2 u'}{\partial x'^2} \tag{3.4}$$

となる．$x'$、$u'$、$t'$はすべて、次元のない量になっている。このように、方程式を次元のない量（無次元量）に対するものに書き換えることを無次元化あるいは規格化という。

## ３．２　陽解法

書くのに面倒なので，これ以降（３．４）式の′は省略する．これを，この節では，差分法で解く．$x$ の区間 $0 \leq x \leq 1$ 上に等間隔 $(h)$ に $N+1$ 個の点を考え，そこでの温度変化を追跡する．温度を $u_0, u_1, u_2, \ldots, u_N$と書く．点の間の間隔が小さければ，微分を差分に書き直せる．つまり，

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_i}{h} \tag{3.5}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} \tag{3.6}$$

よって

$$u_i(t + \Delta t) = u_i(t) + \frac{\Delta t}{h^2}[u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)] \tag{3.7}$$

右辺は時間$t$の時の値のみで計算できるので，$\Delta t$ 後の$x_i$での温度が計算できる．これを，$0, 1, 2, \ldots, N$ についてやれば，$\Delta t$ 後の温度分布がわかる．

## ３．３　境界条件と初期条件

最初，この鉄の棒は $0^\circ$ C であった．$u_i = 0$ for $i = 1, 2, \ldots, N$である．これを，初期条件と呼ぶ．また，片方の端を加熱した後，高温の端は，$100^\circ$ C 低温の端は，$0^\circ$ C に保たれていたとする．いつも $u_0 = 0$ で $u_N = 1$ である $(T_0 = 100)$．これを境界条件という．

## ３．４　繰り返しと配列型

$u_0, u_1, \ldots, u_N$のような，変数の集まりをＣで使うには，配列を用いる．配列は，u[1] とか u[i] といったように使うが，最初に，個数（何番まで使うか）と型（実数か整数か）を指定する必要がある．何番から使うかは決まっている。最初は u[0] と決まっている。

```
float u[10];
```

これは，0 から１０まで、１１個の実数の要素を使うことを示す．

式（３．７）のような繰り返しは，もちろん，

```
i=1;
while (i<=n-1)
  {
  u[i]+=r*(u[i+1]-2*u[i]+u[i-1]);
  i++;
  }
```

としてもよい。（ここで $r$は$\Delta t/h^2$である．) i=0 と i=N では，いつも $u_i$は決まっているから計算する必要無し．

しかし，while に代わるもっと簡潔な表現として，

```
for（i=初期値; 条件判定; i を 1 増やす）
  {
  ..........
  ..........
  }
```

がある．これを用いいると、

```
for (i=1; i<=n-1; i++)
  {
  u[i]+=r*(u[i+1]-2*u[i]+u[i-1]);
  }
```

と書ける。while ループの初期化 (i=1;)、条件判定 ((i<=n-1))、そしてカウンタの再設定 (i++;) を for の () の中に詰め込んだ表現と見てよい。

問題 **3.1**　この上に示した，プログラムには，計算しても正しくできない (間違った答えが出る) 点がある．それはなにか ?i を 1 ，2 ，3 ，と変えながら，どういう計算が行われるか考えてみよ．

課題 **3.2**　この問題を，プログラムを作って解け．$t = 0.1$，$0.2$，$0.5$ の時の，温度分布を示せ．$r$ の値はどの程度にとればよいか ？

## 3 ． 5　ポアッソン方程式

重力や，静電気力などの逆 2 条に比例する力を及ぼすポテンシャルは

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} = 4\pi G\rho \tag{3.8}$$

に従う (静電気力の時は右辺は$\rho/\epsilon_0$になる)．

また，熱伝導の問題で，熱源からの発熱 H を含めると，

$$c\frac{\partial T}{\partial t} = \kappa\frac{\partial^2 T}{\partial x^2} + H \tag{3.9}$$

という方程式が得られるが，この発熱を含む熱伝導の定常解 (時間的に温度分布の変わらない状態 :$\partial u/\partial t \equiv 0$) は，

$$\frac{\partial^2 u}{\partial x^2} = -H \tag{3.10}$$

(ここで，H はもとの H を$\kappa T_0/L^2$ で割ったもの) という方程式を解けばよいことになる．これは，(3 ． 8 ) 式の独立変数が $x$ のみの場合にあたっている．

微分を差分に書き直すと，この式は

$$u_{i+1} - 2u_i + u_{i-1} = -H_i h^2 \tag{3.11}$$

となる．ここで境界条件，$u_0 = 1$，$u_N = 0$ を付加すると，$H_i$は与えられているので，この式は連立方程式—- $u_1, u_2, \ldots, u_{N-1}$に対して $N-1$ 本の式がある—-として，解くことができる．

$$\begin{pmatrix} -2 & 1 & 0 & \ldots & \ldots & 0 \\ 1 & -2 & 1 & 0 & \ldots & 0 \\ 0 & 1 & -2 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & 1 & -2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ \vdots \\ u_{N-1} \end{pmatrix} = - \begin{pmatrix} H_1 h^2 - u_0 \\ H_2 h^2 \\ H_3 h^2 \\ \vdots \\ \vdots \\ H_{N-1} h^2 - u_N \end{pmatrix} \tag{3.12}$$

次の節で，連立方程式の解法について学ぶことにする．

# 3 節への補遺

## 3 ． 4　陰解法

前節で，$r = \Delta t/h^2$ (1) となると，振動が起こって不安定になり，この方法では，真の解を求められないことが分かった．そこで，ここでは，$\Delta t$ をどの様にとっても，振動が起こらない方法について，学ぶ．

陽解法では，

$$u_i(t+\Delta t) = u_i(t) + \frac{\Delta t}{h^2}[u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)] \tag{3.7}$$

とした．陰解法と言うのは，右辺の$\partial^2 u/\partial x^2$の部分を，$u(t)$ だけを用いて書き表すのでなく，$u(t+\Delta t)$ をも用いる．すなわち，

$$u_i(t+\Delta t) = u_i(t)+\frac{\Delta t}{h^2}\{\theta[u_{i+1}(t)-2u_i(t)+u_{i-1}(t)]+(1-\theta)[u_{i+1}(t+\Delta t)-2u_i(t+\Delta t)+u_{i-1}(t+\Delta t)]\} \tag{3.8}$$

とする $(0 \le \theta < 1)$．$\theta = 1/2$ とする方法を，クランク ・ニコルソン法という．いかでは$\theta = 1/2$ の場合の解き方を調べる．

(3 ． 8 ) 式で $t + \Delta t$ での値　(新しい値 ) を左辺に集めると

$$-ru_{i-1}(t+\Delta t) + (2+2r)u_i(t+\Delta t) - ru_{i+1}(t+\Delta t) = ru_{i-1}(t) + (2-2r)u_i(t) + ru_{i+1}(t). \tag{3.9}$$

右辺の値は現在の値であるから，すでに分かっている．分からない $t + \Delta t$ の時の値を求めることを考えればよい．これは，連立方程式を解くことに帰着する．行列形式で書けば，

$$\begin{pmatrix} 2+2r & -r & 0 & \dots & \dots & 0 \\ -r & 2+2r & -r & 0 & \dots & 0 \\ 0 & -r & 2+2r & -r & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & -r & 2+2r & -r \\ 0 & 0 & \dots & 0 & -r & 2+2r \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ \vdots \\ u_{N-1} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ \vdots \\ b_{N-1} \end{pmatrix} \tag{3.10}$$

ここで，$u$ 現在の値を使って

$$b_i = ru_{i-1}(t) + (2-2r)u_i(t) + ru_{i+1}(t), \tag{3.11}$$

である．ただし，もし境界条件 $u_0 = 0$，$u_N = 1$ を課すのであれば，当然

$$b_1 = (2-2r)u_1(t) + ru_2(t), \tag{3.12a}$$

$$b_{N-1} = ru_{N-2}(t) + (2-2r)u_{N-1}(t) + 2r, \tag{3.12b}$$

としなければならない．

この連立方程式を解けば，$t + \Delta t$ の温度分布が求められる．求まった新しい $u$ を，今度は現在の値として右辺に代入すれば，また，次の新しい $u$ の値を求められる．これを繰り返すことによって，さらに任意の時間の温度分布が求められる．

課題　ガウスの消去法による連立方程式を解くプログラムに手を加えてクランク ・ニコルソン法によって，熱伝導を陰解法で解くプログラムを作れ．

# ４．連立方程式の解法

## ４．１　ガウスの消去法

解くべき連立方程式が

$$A\mathbf{x} = \mathbf{b} \tag{4.1}$$

であるとする．ここで，

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \tag{4.2}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \tag{4.3}$$

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{4.4}$$

である．

$$\begin{cases} a_{11}^{(1)}x_1 + a_{12}^{(1)}x_2 + \cdots + a_{1n}^{(1)}x_n = b_1^{(1)} & \#1 \\ a_{21}^{(1)}x_1 + a_{22}^{(1)}x_2 + \cdots + a_{2n}^{(1)}x_n = b_2^{(1)} & \#2 \\ a_{31}^{(1)}x_1 + a_{32}^{(1)}x_2 + \cdots + a_{3n}^{(1)}x_n = b_3^{(1)} & \#3 \\ \cdots & \\ a_{n1}^{(1)}x_1 + a_{n2}^{(1)}x_2 + \cdots + a_{nn}^{(1)}x_n = b_n^{(1)} & \#\mathrm{n} \end{cases} \tag{4.5}$$

消去の第１段では，(４．５）式の２番目以降の式から $x_1$を消す．まず$\#1$ に

$$m_{21} = \frac{a_{21}}{a_{11}} \tag{4.6}$$

を掛けて$\#2$ から引くと，

$$(a_{22}^{(1)} - m_{21}a_{12}^{(1)})x_2 + \cdots + (a_{2n}^{(1)} - m_{21}a_{1n}^{(1)})x_n = b_2^{(1)} - m_{21}b_1^{(1)} \tag{4.7}$$

となる．同じように，$\#1$ に

$$m_{31} = \frac{a_{31}}{a_{11}}$$

を掛けて$\#3$ から引くと，

$$(a_{32}^{(1)} - m_{31}a_{12}^{(1)})x_2 + \cdots + (a_{3n}^{(1)} - m_{31}a_{1n}^{(1)})x_n = b_3^{(1)} - m_{31}b_1^{(1)}$$

(４．５）式は

$$\begin{cases} a_{11}^{(1)}x_1 + & a_{12}^{(1)}x_2 + \cdots + a_{1n}^{(1)}x_n = b_1^{(1)} & \#1' \\ & a_{22}^{(2)}x_2 + \cdots + a_{2n}^{(2)}x_n = b_2^{(2)} & \#2' \\ & a_{32}^{(2)}x_2 + \cdots + a_{3n}^{(2)}x_n = b_3^{(2)} & \#3'2 \\ & \cdots & \\ & a_{n2}^{(2)}x_2 + \cdots + a_{nn}^{(2)}x_n = b_n^{(2)} & \#n' \end{cases}$$

ここで，$a_{ij}^{(2)}$と$b_i^{(2)}$は，次式で定義される．

$$\begin{cases} a_{ij}^{(2)} = a_{ij}^{(1)} - m_{i1}a_{1j}^{(1)} & i,j = 2,3,\ldots,n \\ b_i^{(2)} = b_i^{(1)} - m_{i1}b_1^{(1)} & i = 2,3,\ldots,n \end{cases} \tag{4.9}$$

次の段階では，#3' 以降の式から $x_2$を含む項を消去する．以下同様にして，次々に $x_i$を消していく．

この消去の手順をまとめると，

```
for (k=1; k <= n-1; k++)
  {
  d[k]=a[k][k];
  for (i=k+1; i<=n; i++)
    {
    m[i][k]=a[i][k]/d[k];
    for (j=k+1; j<=n; j++)
      {
      a[i][j]=a[i][j]-m[i][k]*a[k][j];
      }
    b[i]=b[i]-m[i][k]*b[k];
    }
  }
```

この操作で，元の方程式は

$$U\mathbf{x} = \mathbf{b} \tag{4.10}$$

$$U = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ & & \ddots & \vdots \\ & & & a_{nn}^{(n)} \end{pmatrix} \tag{4.10}$$

$$\mathbf{b} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(n)} \end{pmatrix} \tag{4.11}$$

という，形に変換されたことになる．$U$は上三角行列である．ここまでの操作を，<u>前進消去</u>という．

（４．１０）式を解くことはたやすい．つまり，$n$ 行目の式から，

$$x_n = b_n^{(n)}/a_{nn}^{(n)} \tag{4.13}$$

これを，$n-1$ 行目の式に代入すると，

$$x_{n-1} = (b_{n-1}^{(n-1)} - a_{n-1\ n}^{(n-1)}x_n)/a_{n-1\ n-1}^{(n-1)} \tag{4.14}$$

この二つを $n-2$ 行目の式に代入すると，

$$x_{n-2} = (b_{n-2}^{(n-2)} - a_{n-2\ n-1}^{(n-2)}x_{n-1} - a_{n-2\ n}^{(n-2)}x_n)/a_{n-2\ n-2}^{(n-2)} \tag{4.15}$$

以下同様で，下から順番に $x_i$を求めることができる．まとめると，

```
x[n]=b[n]/a[n][n];
for (i=n-1; i>=1; i--)
  {
  x[i]=b[i];
  for (j=i+1; j<=n; j++)
    {
    x[i]=x[i]-a[i][j]*x[j];
    }
  x[i]=x[i]/a[i][i];
  }
```

この部分を，後退代入という．
課題５　連立方程式を解くプログラムをつくり，それを利用して（3．1 2）式を解け．但し，熱源の分布としては $0 \leq x \leq 1$ では $H = 1$ とせよ．

# ５．数値積分

## ５．１台形則

定積分

$$I = \int_a^b f(x)dx, \tag{5.1}$$

を求めるのに，$[a, b]$ を $N$等分して，

$$h = (b - a)/N \tag{5.2}$$

$I$を次のように近似するのが，台形則である．

$$I_N = h\left[\frac{1}{2}f(a) + \sum_{n=1}^{N-1} f(a + nh) + \frac{1}{2}f(b)\right] \tag{5.3}$$

はじめから，$N$を決めて計算するのももちろんよいが，しだいに，$N$を増やして行って適当な精度で積分値が求められたら，終わるというプログラムが望ましい．
問題 $N = 1$ から $N = 2$,$N = 4$,... と分点の数を２倍に増やして行き，$I_N$と $I_{2N}$の差が$\epsilon$（これは十分に小さい数）よりも小さくなったら，$I$がもとまったとして結果を出力するプログラムを作れ．$(5.7)$ 式の積分の例を行え．

上の問題では，前に計算された $I_N$の値を使わないで次の $I_{2N}$を計算した．これは，効果的ではないので，前の値を使うように改良することを考える．

これには，次のようにやればよい．$N = 1$ のときは，

$$I_1 = [f(a) + f(b)]\frac{b - a}{2} \tag{5.4}$$

$N$を倍にして（h を半分にして）$I_{2N}$を計算するには，

$$J_N = h\sum_{n=1}^{N} f\left(a + (n - \frac{1}{2})h\right) \tag{5.5}$$

$$h = (b - a)/N$$

を求めて，$I_N$と平均すればよい．つまり，

$$I_{2N} = \frac{I_N + J_N}{2} \tag{5.6}$$

とすれば，刻み幅が，半分の台形則 $I_{2N}$になる．$|I_N - I_{2N}| < \epsilon$という条件を満足したら終わりということにすれば，良い．$\epsilon$は適当な $(10^{-6}$とか$)$ 小さい正の数である．

問題　次の積分値を求めよ．

$$I = \int_{-1}^{1} \frac{2}{1+x^2} dx (= \pi) \tag{5.7}$$

## 5．2　無限区間の積分

$$I = \int_0^{\infty} f(x) dx \tag{5.8}$$

で，$f(x)$ が非常に早く $0$ に近付く様なものを考える．

$$I(h) = h \sum_{n=0}^{\infty} f(nh) \tag{5.9}$$

こんどは，下から順番に加えて行って関数の値 $f(nh)$ が，それまでの，部分和の$\epsilon$倍にまで減ったところで打ち切ればよい．[1]　$(\sin x \exp^{-\frac{x^2}{2}}$のように，x の大きい所で，零点を持つものもあるので，$f(nh)$ だけでなく，$f((n+1)h)$,$f((n+1)h)$ も充分に小さくなっていることを確かめた方がよい．この手続きで，$I(h)$ ある $h$ に対する値が求められる．

　前の問題と同じように，$h$ を，半分にしていって，$|I(h) - I(h/2)|$ が充分に小さくなったら解が求まったと考えよう．

問題　次の積分値を求めよ．

$$I = 2\int_0^{\infty} \exp \frac{-x^2}{2} dx (= \sqrt{2\pi}) \tag{5.10}$$

この章の参考文献 :数値計算の常識，伊理正夫，藤野和建，共立出版．

# 付録 1　:3 節への準備– 配列型

　ここで，配列型について学ぶ．同じ性質のデータが多数ある時，変数名を多数使って a，b，c，d，e，f，g，などとするのは後の取扱に面倒である．こんな時はデータに順番をつけて $a_1, a_2, a_3, \ldots$ とする方がよい．このようなデータを C で取り扱うのが配列型　(教科書 5 6 ページ) である．配列型はプログラム上では a[1] とか a[i] というふうに表す　(a[i] は i=3 なら a[3] をあらわす)．変数が配列型であることは

```
int a[100];
float b[10];
```

[1] ここでは幅 1 高さ $f(nh)$ の面積と部分和で表される面積を比べている．

の様にして宣言する．こうすると，a[0]，a[1]，a[2]，...，a[99] の 100 個の整数型の記憶場所と，b[0]，b[1]，b[2]，...，b[9] の 10 個の実数型の記憶場所が準備される．最初が 0 から始まることに注意．

問題　c:test.dat というファイルに次のようなデータが入っている．

```
15   クラスの人数
88   １番の人の点数
45   ２番の人の点数
60   ３番の人の点数
......
90   １５番目の人の点数
```

これを読んで，平均値，標準偏差を求め，元のデータと共に出力するプログラムを作れ．

注意１　:平均値，標準偏差の求め方は教科書５７ページ参照

注意２　:キーボードからでなく c:test.dat というファイルからデータを読むには，(教科書９６及び１０２ページ) つぎのようにする．

```
main()
  {
  FILE *inf;                    ファイルポインター型変数 inf の宣言
......
......
  inf = fopen("c:test.dat", "r");
                  c:test.dat というファイルを inf と呼ぶ．
                  r は read 読み込むファイルとして使うことを示す．
                      従って内容が変化することはない．
                  その他 w は書き込むファイルとして，a は続けて書き込むファイルとして
                  オープンすることを示す．
  fscanf(inf,&a[i])
                  実際に読み出し，a[i] に記憶する．
                  scanf(の代わりに，ファイル inf から読むので fscanf(inf, をつける．
```

書き出す場合は fprintf(outf,.. のようになる．

問題　前の問題ができたら，さらに，クラスの最高点と最低点を求めて元のデータと共に出力するプログラムを作れ．

参考　:最大値の捜し方

```
saidai_chi=0;
for (i=1 ;i<= n ; i++)
  {
  if (a[i]> saidai_chi)
    {
    saidaichi=a[i];
    }
  }
end;
```

# 付録 2 :グラフィックの使い方

(1) 座標系 :画面の左上が (0,0) で，x-方向が 0 ~ 639，y-方向が 0 ~ 399 である．

(2) グラフィックスの準備 :まず，グラフィックスを使うためには#include として

```
#include <graphics.h>
```

と書く．
グラフィックスの初期化 :さらに，main() の適当なところに，図を描く準備として

```
gd = DETECT;
initgraph(&gd, &gm, "b:\\turbo_c\\bgi");
```

と記述する．ここで，gd，gm は int 型の変数である (従って宣言が必要である)．これでグラフィックが使えるようになる．

(3) グラフを書くには，プロットすべき座標の組が，$(x[0], y[0]), (x[1], y[1]), \ldots$ で与えられているとすると，

```
moveto(Fx(x[0]), Fy(y[0]));
for (i:=1; i<=n; ++i)
{
  lineto(Fx(x[i]), Fy(y[i]));
}
```

ここで，moveto(int x, int y) は点 (x,y) へ移動し (線は結ばない)，lineto(int x, int y) は今いる点から，点 (x,y) へ直線を結ぶ命令 (C の用語では関数) である．
また，Fx(x) は，座標 x の点を，画面の 0 ~ 639 のどこに対応させるかを決める関数．Fy(y) は，座標 y の点を，画面の 0 ~ 399 のどこに対応させるかを決める関数．例えば，

```
int Fx(x:real)
  {
    return(639*x);
  }
```

こうすると，x=(0 ~ 1) が Fx=(0 ~ 639) に移る．

(4) 終了処理 :

```
closegraph;
```

とすれば，画面が消えて元のターボ C の画面に戻る．消さないで残して置くには，次の例のように，何かを入力待にすればよい．この例では y を入力するとグラフは消える．

```
printf("Do you erase this graph?");
scanf("%c",&y);
if (y=='y') then closegraph;
```

y は文字型変数で，char y; で定義する．DOS の画面で，グラフィックスが消したい場合には，

```
B:\> g cls
```

というコマンドを利用する．また，グラフをコピーしたい場合は，

```
B:\> hardcopy
```

を一度実行しておけば，次のようにしてできるようになる．[copy] でグラフとテキスト（文字）が，[cntl]+[copy] でテキストだけが，[graph]+[copy] でグラフだけがハードコピーされる．