数値解析 C　(塩田)　2006 年 10 月 4 日

— 誤差の話 —

1. 丸め誤差

全ての数値は「コンピュータで扱える数値」に近似される。

例　$e^{\pi\sqrt{163}} = 262537412640768743.99999999999925\ldots$
は double 型で計算しても整数にしか見えない。

2. オーバーフロー・アンダーフロー

大き過ぎる数は表現できずにエラーを起こす。小さ過ぎる数は 0 にされる。

例

```
#include<stdio.h>
main()
{
        int i,n;
        float x;
        printf("1 を 10 で n 回割ってから、10 を n 回掛けると ...\n");
        for(n=35;n<=50;n++){
                x=1.0;
                for(i=1;i<=n;i++) x/=10;
                for(i=1;i<=n;i++) x*=10;
                printf("n = %d のとき x = %f\n",n,x);
        }
}
```

```
1 を 10 で n 回割ってから、10 を n 回掛けると ...

n = 38 のとき x = 1.000000
n = 39 のとき x = 1.000000
n = 40 のとき x = 0.999995
n = 41 のとき x = 0.999967
n = 42 のとき x = 1.000527
n = 43 のとき x = 0.994922
n = 44 のとき x = 0.980909
n = 45 のとき x = 1.401299
n = 46 のとき x = 0.000000
n = 47 のとき x = 0.000000
```

3. 累積誤差

塵も積もれば山となる。

例

```
#include<stdio.h>
#include<math.h>
#define pi 3.14159265358979324
main()
{
        int i,j,n;
        float x;
        printf("理論値は 0\n");
        for(j=1;j<=7;j++){
                n=(int)pow(10,j);
                x=0.0;
                for(i=1;i<=n;i++) x+=sin((2*pi*i)/n);
                printf("n = %8d のとき x = %f\n",n,x);
```

```
        }
}

理論値は 0
n =       10 のとき x = -0.000000
n =      100 のとき x = 0.000001
n =     1000 のとき x = -0.000015
n =    10000 のとき x = -0.000050
n =   100000 のとき x = -0.000020
n =  1000000 のとき x = -0.028050
n = 10000000 のとき x = 0.078272
```

4. 積み残し

$|a| \gg |b|$ のときに $a+b$ を計算すると b の下の方の桁が無視される。

例

```
#include<stdio.h>
#include<math.h>
main()
{
        int i,j,n;
        float x;
        printf("理論値は円周率 3.141592653 ...\n");
        printf("昇順に足す\n");
        for(j=1;j<=7;j++){
                n=(int)pow(10,j);
                x=0.0;
                for(i=1;i<=n;i++)
                        if(i%2) x+=4.0/(2*i-1); else x-=4.0/(2*i-1);
                printf("n = %8d のとき x = %f\n",n,x);
        }
        printf("降順に足す\n");
        for(j=1;j<=7;j++){
                n=(int)pow(10,j);
                x=0.0;
                for(i=n;i>=1;i--)
                        if(i%2) x+=4.0/(2*i-1); else x-=4.0/(2*i-1);
                printf("n = %8d のとき x = %f\n",n,x);
        }
}

/* 実行結果

理論値は円周率 3.141592653 ...
昇順に足す
n =       10 のとき x = 3.041840
n =      100 のとき x = 3.131593
n =     1000 のとき x = 3.140593
n =    10000 のとき x = 3.141498
n =   100000 のとき x = 3.141586
n =  1000000 のとき x = 3.141595
n = 10000000 のとき x = 3.141597
降順に足す
n =       10 のとき x = 3.041840
n =      100 のとき x = 3.131593
n =     1000 のとき x = 3.140593
n =    10000 のとき x = 3.141493
n =   100000 のとき x = 3.141583
n =  1000000 のとき x = 3.141592
n = 10000000 のとき x = 3.141593
```

5. 桁落ち

$a \quad b$ のときに $a - b$ を計算すると著しく精度が落ちる。

例

```
#include<stdio.h>
#include<math.h>
main()
{
int j;
        float a,b=1.0,c=1.0,x,y;
        printf("2 次方程式 a x^2 + x + 1 = 0 の解\n\n");
        for(j=1;j<=8;j++){
                a=pow(10,-j);
                x=(-b+sqrt(b*b-4*a*c))/(2.0*a);
                y=(2.0*c)/(-b-sqrt(b*b-4*a*c));
                printf("a = 10^(-%d) のとき (-b+sqrt(b^2-4ac))/(2a) = %f\n",j,x);
                printf("                   (2c)/(-b-sqrt(b^2-4ac)) = %f\n\n",y);
        }
}
```

```
2 次方程式 a x^2 + x + 1 = 0 の解

a = 10^(-1) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.127017
(2c)/(-b-sqrt(b^2-4ac)) = -1.127017

a = 10^(-2) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.010206
(2c)/(-b-sqrt(b^2-4ac)) = -1.010205

a = 10^(-3) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.001004
(2c)/(-b-sqrt(b^2-4ac)) = -1.001002

a = 10^(-4) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.000117
(2c)/(-b-sqrt(b^2-4ac)) = -1.000100

a = 10^(-5) のとき
(-b+sqrt(b^2-4ac))/(2a) = -0.999878
(2c)/(-b-sqrt(b^2-4ac)) = -1.000010

a = 10^(-6) のとき
(-b+sqrt(b^2-4ac))/(2a) = -0.998379
(2c)/(-b-sqrt(b^2-4ac)) = -1.000001

a = 10^(-7) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.043081
(2c)/(-b-sqrt(b^2-4ac)) = -1.000000

a = 10^(-8) のとき
(-b+sqrt(b^2-4ac))/(2a) = -1.490116
(2c)/(-b-sqrt(b^2-4ac)) = -1.000000
```