小数を分数で近似する

bunsu

有理数は分数で表わすことができますが，無理数($\sqrt{2}$など）や超越数（π やeなど）は分数で表わせず無限小数となります．

そこで，このような小数を分数で近似することを考えました．

アルゴリズム

小数を分数で表わすというのは，たとえば，π=3.141592653589…を，

3+10/71=3.1418511…（誤差0.008%）
とか
3+16/113=3.1415929…（誤差1/200万）

などのように近似することをいいます．

では，もとの小数との誤差が最小となる分数を発見する方法について考えてみます．

このアルゴリズムの厳密な数学的証明はできませんが，おおまかな考え方を説明します．

考え方のポイントとしては，もとの小数（以後真値T）と近似分数 B/A との誤差，

$\varDelta = T - B/A$

が小さくなる方向に探索を行ないます．

まず，最初の近似分数を真値より小さくなるように設定（$T>B/A$）します．このとき，

$\varDelta = T - B/A > 0$

となります．

次に，近似分数の分子を固定して分母から-1すると，近似分数は大きくなり真値との誤差 $\varDelta$ は小さくなります．

そして分母をどんどん小さくしていけば，誤差 $\varDelta$ もどんどん小さくなり，ついには "$\varDelta<0$" になります．

このときの分数の値もしくはこの寸前の分数の値が "$\varDelta=0$" に最も近く，誤差が最小となるので，これを現時点での誤差最小分数として記録しておきます．

次に，これ以上分母から "-1" しても $\varDelta$ は増える一方なので，分子から "-1" して近似分数の値を小さくして再び "$\varDelta>0$" にします．

そして，この操作を繰り返していけば，最後にもっとも $\varDelta$ の小さな近似分数が記録されていることになります．

最小誤差探索法

以上の方法を「最小誤差探索法」と呼ぶことにします．

この方法を具体例でもっと詳しく説明します．例として π を取り扱うことにします．

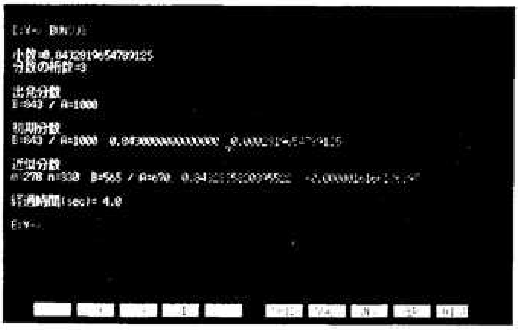
まず，分数の分母を何桁にするか決めます．ここでは，3桁として π の小数点以下4桁以降を切り捨てて，0.1415926…を0.141（整数部分は簡単にするため省略します）とします．

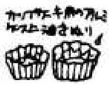
よって出発分数は141/1000となり，これが最小分数となります．そして，真値に最も近い分数は141/1000〜15/100の間にあることになります．

さて，出発分数の分母から "-1" して分数の値を大きくします．そして，これが真値より大きくなるまで "-1" します．

このときのひとつ前の分数またはこのときの分数で，真値との差が現時点での最小になるので，これを記録しておき

実行画面





ます。

この例では，141/1000から出発して141/999，141/998…となり，141/995が真値より大きくなる分数です．この場合は，ひとつ前の141/996が真値との誤差が最小の分数です。

次に，現在の分数の分子から-1して分数の値を真値より小さくします。ここでは，140/996となります。

そして，これをまた出発分数としてこれらの操作を繰り返し，15/100になったら終了します．そうすると，誤差が最小の分数が記録されていることになります．

この例では，16/113です．

最小誤差計算法（改良案）

もっと探索の数を減らす方法を考えました。

前述の方法により，真値 T との誤差が負となるひとつ前の分数（初期分数）を求めます．これを "B/A" とします．

ここで，真値との誤差は，

$$\Delta(0,0)=T-B/A>0$$

となります．

次に，B/A の分子から "$-m$" して，分母から "$-n$" した "$(B-m)/(A-n)$" について，

$$\Delta(m,n)=T-(B-m)/(A-n)>0$$

とすると，

$$\begin{aligned}\Delta(m,n)-\Delta(0,0)&=B/A-(B-m)/(A-n)\\&=(mA-nB)/A(A-n)\end{aligned}$$

です．

ここで，$\Delta(m,n)$ が初期誤差 $\Delta(0,0)$ より小さくなるには，

$$\Delta(m,n)<\Delta(0,0)$$
$$\Delta(m,n)-\Delta(0,0)<0$$

だから，

[illegible]

となります．

m を1，2，3…と変化させて，条件式 "$mA<nB$" に合う n を求めます．そして，このときの "$(B-m)/(A-n)$" を計算して記録しておきます，

この例では，初期分数は141/996で，$m=1$，2，3，4…に対して $n=8$，15，22，29…となります．

さらに計算を続けてゆき，"15/100" になったら終了します，そうすると，誤差が最小の分数が記録されていることになります，この例では，前述の方法と同じ "16/113" です．この方法を最小誤差計算法と呼ぶことにします，

もっと改良を考えてみました．

真値との誤差 $\Delta(m,n)$ を，m を x 軸に $\Delta(m,n)$ を y 軸にとってグラフにしてみると，図のように単調に増加していき突然小さくなりまた単調増加を繰り返しているように見えます（数学的な証明はできませんが）．

そして，$\Delta(m,n)=0$ の線を横切っているので，$\Delta(m,n)$ の値が－から＋に転じたときの前後の値を調べて，より誤差の小さい方を現時点での最小誤差分数とすればよいことが分かります．

プログラムについて

bunsu.c は最小誤差探索法のプログラムで，**bunsu2.c** は最小誤差計算法のプログラムです．**bunsu3.c** は最小誤差計算法の改良版です．

なお，このプログラムは "Turboc 2.0" で開発しました．整数は **long int** を用い，実数は **double** を使ったので，分数の分母の桁数は9桁まで計算できます．小数点以下の精度は16桁までは信用できます．

π についていろいろな桁数で近似分数を求めた結果を表に示しました．一般に分母の桁数を多くするほど，真値との誤差は小さくなると思われますが，分母が4桁のときは3桁のときとまったく同じになってしまいました．

また，最小誤差計算法の方が最小誤差探索法より約2～3倍程速度が速くなりましたが，思ったほどの改善ではありませんでした．最小誤差計算法改良版は計算法とあまり変わらない速度だったので，プログラムは単純な方が良いということでしょうか．

＊

「女房酔わせてどうするの」じゃありませんが，小数を分数に近似してもどうということはありません．しかし，π の近似分数「16/113」は有名ですので役に立つかもしれません．

そのほか，e などについて自分で試してみてください．

表1　π の近似値

[illegible]	[illegible]	[illegible]
1	1/7	3.1｜128571428571428
2	14/99	3.141｜41414141414
3	16/113	3.141592｜9203539823
4	144/1017	〃
5	14093/99532	3.141592653｜6189366
6	140914/995207	3.14159265358｜86504

▶初めまして，EJと申す者でございます．I/O読者にはまだまだ8bit機ユーザーの方が沢山いてうれしいです．私の所有機種はX1-turboZIIとPC-1261，PC-E200です．ところで，8月号P214の，がいちさん，私のターボZでは2HDを2Dフォーマットできますよ．(でもできた!使ったディスクはフジのMD2HSUPER-HR(おのゴールドハブリングがついてるやつです)で，現在3枚枚は2Dフォーマットしてます．

図 1

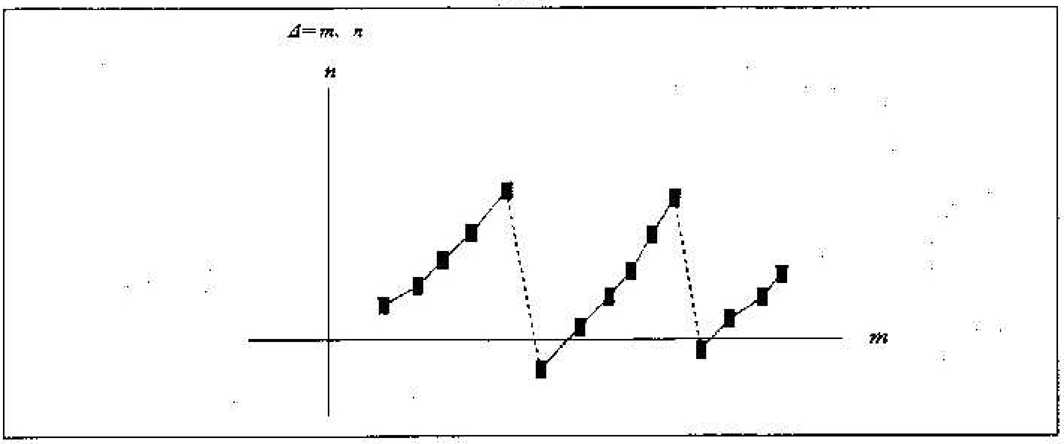
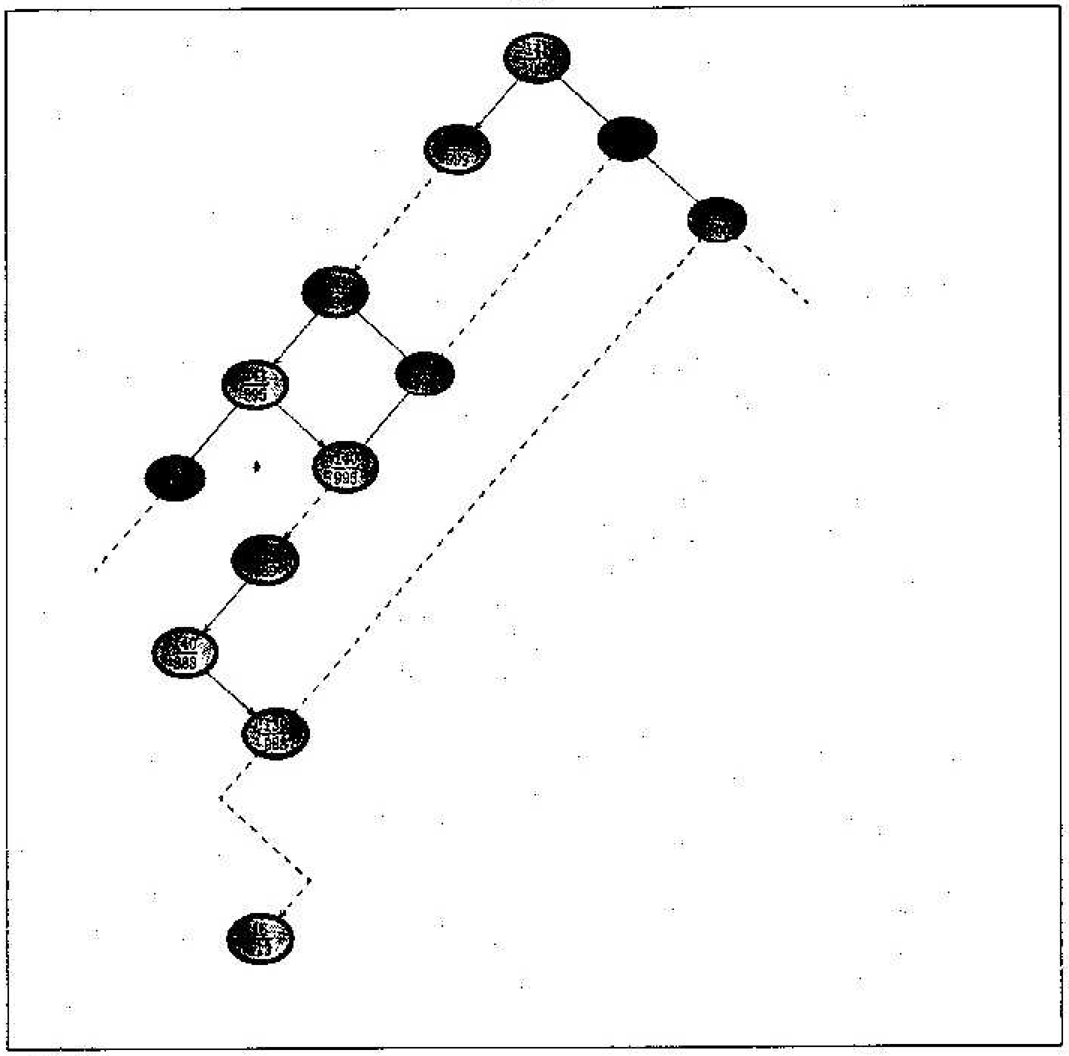


図 2



P.S.私のまわりはみんな98ユーザーだよー。でもただ1人ターボZ日もってる人がいるからまだいいけど。

(EJ)

bunsu.c

```
/*  bunsu.c 完成版 ver1.2  1991.3.1   Turbo C ver2.0  SmallModel  by S.Furuta
                     小数を分数で近似する
                     最小誤差探索法                                          */


#include <stdio.h>
#include <math.h>
#include <time.h>

double value;

main()
{
        int kazu;
        double vl,val;
        long int va,vb;
        time_t start,finish;
        double times;

        printf("\n小数=");                        /* valueに小数 */
        scanf("%lf",&value);
        printf("分数の桁数=");                    /* 分数の桁数 */
        scanf("%d",&kazu);                        /* vlに分子   */
        modf(value*pow10(kazu),&vl);              /* valに分母  */
        val=pow10(kazu);
        vb=(long int)vl;
        va=(long int)val;

        printf("\n出発分数  B=%ld / A=%ld\n\n",vb,va);

        time(&start);

        DownPath(kazu,vb,va);

        time(&finish);
        times=difftime(finish,start);
        printf("経過時間(sec)= %.1f\n",times);

}
/* 下降して探索する */
DownPath(kazu,vb,va)
int kazu;
long int vb,va;
{
        long int A,B;
        double vv,BA;
        double val,vl,dlta,dl;
        double va_min,vb_min,dlta_min;

        vl=pow10(kazu-1);                         /* 分母の下限 */
        vv=(long int)vl;
        vl=(double)vb;
        val=(double)va;
        dlta=value-(vl/val);
        dlta_min=dlta;

        while ((va>=vv)&&(vb>0)) {
            while ((dlta>=0)&&(va>vv)) {
                --va;                             /* 分母を-1 */
                vl=(double)vb;
                val=(double)va;
                dlta=value-(vl/val);              /* 真値との誤差 */
            }

            dl=value-(vl/(va+1));                 /* ひとつ手前の値 */
            if (dlta<0) dlta=-dlta;
            if (dlta>dl) {
                if (dl<=dlta_min) {
                    vb_min=vb;
                    va_min=va+1;
                    dlta_min=dl;
                }
            }
            else {
                if (dlta<=dlta_min) {
                    vb_min=vb;
                    va_min=va;
                    dlta_min=dlta;
                }
            }

            --vb;                                 /* 分子を-1 */
            vl=(double)vb;
            val=(double)va;
            dlta=value-(vl/val);
        }

        A=(long int)va_min;
        B=(long int)vb_min;
        BA=vb_min/va_min;
        dlta=value-BA;

        printf("\n近似分数\n");
        printf("B=%ld / A=%ld  %.16f  %.16f\n\n",B,A,BA,dlta);

}
```

bunsu2.c

```
/*  bunsu2.c 完成版 ver1.3  1991.3.2   Turbo C ver2.0  SmallModel  by S.Furuta
                     小数を分数で近似する
                     最小誤差計算法                                          */


#include <stdio.h>
#include <math.h>
#include <time.h>

double value;
double vb_min,va_min,dlta_min;

main()
{
        int kazu;
        double vl,val;
        long int va,vb;
        time_t start,finish;
        double times;

        printf("\n小数=");                        /* valueに小数 */
        scanf("%lf",&value);
        printf("分数の桁数=");                    /* 分数の桁数 */
        scanf("%d",&kazu);                        /* vlに分子   */
        modf(value*pow10(kazu),&vl);              /* valに分母  */
        val=pow10(kazu);
        vb=(long int)vl;
        va=(long int)val;

        printf("\n出発分数\n");
        printf("B=%ld / A=%ld\n\n",vb,va);

        time(&start);

        InitPath(&vb,&va);
        DownPath(vb,va,kazu);

        time(&finish);
        times=difftime(finish,start);
        printf("\n経過時間(sec)= %.1f\n",times);

}
/* 初期分数を探索する */
InitPath(vb,va)
long int *vb,*va;
{
        double val,vl,dlta,dl;

        do {
            --(*va);
            vl=(double)(*vb);                     /* 分子 */
            val=(double)(*va);                    /* 分母 */
            dlta=value-(vl/val);                  /* 真値との誤差 */
        } while (dlta>=0);

        dl=value-((double)(*vb))/((double)(*va+1));
        if (dl<-dlta) {
            dlta_min=dl;
            va_min=0;
            vb_min=0;
        }
        else {
            dlta_min=-dlta;
            va_min=0;
            vb_min=1;
        }

        ++(*va);
        vl=(double)(*vb);
        val=(double)(*va);
        dlta=value-(vl/val);

        printf("初期分数\n");
        printf("B=%ld / A=%ld  %.16f  %.16f\n",*vb,*va,vl/val,dlta);

}
/* 漸化式から探索する */
DownPath(vb,va,kazu)
long int vb,va;
int kazu;
{
        double val,vl,dlta,BA;
        long int m,x,y,z;
        double vm,vx,vy,vz,vn;

        vl=pow10(kazu-1);                         /* 分母の下限 */
        vv=(long int)vl;
        vl=(double)vb;                            /* 分子 */
        val=(double)va;                           /* 分母 */
        x=va;
        y=vb;

        m=1;
        while ((x>=vv)&&(y>0)) {
            vm=(double)m;                         /* 分子, 分母の計算 */
            vn=ceil((vm*val)/vl);
            vy=vl-vm;
            vx=val-vn;
            dlta=value-(vy/vx);                   /* 真値との誤差 */

            m=(long int)vm;
            n=(long int)vn;
            y=(long int)vy;
            x=(long int)vx;
            if ((x<vv)||(y<=0)) break;
            if (dlta>=0) {                        /* 現時点の誤差最小分数を記録 */
                if (dlta<=dlta_min) {
                    dlta_min=dlta;
                    va_min=vx;
                    vb_min=vy;
                }
            }
            else {
                if (-dlta<=dlta_min) {
                    dlta_min=-dlta;
                    va_min=vx;
                    vb_min=vy;
                }
```

I/Oプラザ

Sep. 1991

▶ある日生物の授業でスゴイビデオを見ました。一言で表現するなら「超残酷な"しぼり"のビデオ」です。カーテンで締め切って暗くした実験室の中で、皆騒ぎするのも惜しんで見入っていました。常軌を逸した過激さでしたが、怖いもの見たさというやつか、はたまた野次馬の暴走する好奇心か、結局熱心に見ていたのでした。
[illegible]カイコガの幼虫を縛って部分的に成長を抑制してしまうのですから、資料集などの図で見るよりも映像で見るほ

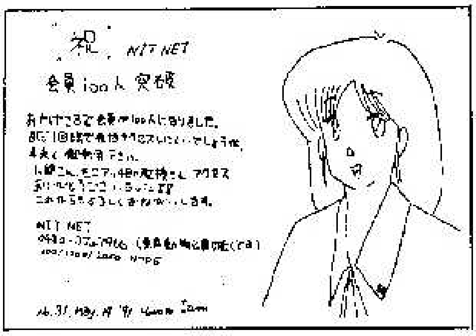
```
        }
        m++;
    }

    m=(long int)vm_min;
    n=(long int)vn_min;
    y=(long int)(vl-vm_min);
    x=(long int)(val-vn_min);
```

```
    BA=(vl-vm_min)/(val-vn_min);
    dlta=value-BA;

    printf("\n近似分数\n");
    printf("m=%ld n=%ld  B=%ld / A=%ld  %.16f  %.11f\n",m,n,y,x,BA,dlta);

}
```

bunsu3.c

```
/*  bunsu3.c 完成版 ver1.1  1991.3.3   Turbo C ver2.0  SmallModel  by S.Furuta
                    小数を分数で近似する
                    最小限遡行探索法（改良版）                              */


#include <stdio.h>
#include <math.h>
#include <time.h>

double value;
double vn_min,vm_min,dlta_min;

main()
{
    int kazu;
    double vl,val;
    long int va,vb;
    time_t start,finish;
    double times;

    printf("\n小数=");                     /* valueに小数 */
    scanf("%lf",&value);
    printf("分数の桁数=");
    scanf("%d",&kazu);                     /* 分数の桁数 */
    modf(value*pow10(kazu),&vl);           /* vlに分子   */
    val=pow10(kazu);                       /* valに分母  */
    vb=(long int)vl;
    va=(long int)val;

    printf("\n出発分数\n");
    printf("B=%ld / A=%ld\n\n",vb,va);

    time(&start);

    InitPath(&vb,&va);
    DownPath(vb,va,kazu);

    time(&finish);
    times=difftime(finish,start);
    printf("\n経過時間(sec)= %.1f\n",times);

}
/* 初期分数を探索する */
InitPath(vb,va)
long int *vb,*va;
{
    double val,vl,dlta,dl;

    do {
        --(*va);
        vl=(double)(*vb);                  /* 分子 */
        val=(double)(*va);                 /* 分母 */
        dlta=value-(vl/val);               /* 真値との誤差 */
    } while (dlta>=0);

    dl=value-((double)(*vb))/((double)(*va+1));
    if (dl<-dlta) {
        dlta_min=dl;
        vm_min=0;
        vn_min=0;
    }
    else {
        dlta_min=-dlta;
        vm_min=0;
        vn_min=1;
    }

    ++(*va);
    vl=(double)(*vb);
    val=(double)(*va);
    dlta=value-(vl/val);
    printf("初期分数\n");
    printf("B=%ld / A=%ld  %.16f  %.16f\n",*vb,*va,vl/val,dlta);

}
/* 条件式から探索する */
DownPath(vb,va,kazu)
long int vb,va;
int kazu;
{
    double val,vl,dlta,dl,BA;
    long int m,n,y,x;
    double vm,vn,vx,vy,vv;
    double v_m,v_n,v_x,v_y,dl_ta;

    vv=pow10(kazu-1);                      /* 分母の下限 */
    vv=(long int)vv;
    vl=(double)vb;                         /* 分子 */
    val=(double)va;                        /* 分母 */
    x=va;
    y=vb;
    dlta=0;

    m=1;
    while ((x>vv)&&(y>0)) {
        dl=dlta;
        vm=(double)m;                          /* 分子，分母の計算 */
        vn=ceil(vm*val/vl);
        vy=vl-vm;
        vx=val-vn;
        dlta=value-(vy/vx);                    /* 真値との誤差 */

        m=(long int)vm;
        n=(long int)vn;
        y=(long int)vy;
        x=(long int)vx;
        if ((x<vv)||(y<=0)) break;
        if ((dlta>dl)&&(dl<0)&&(dlta>0)) {     /* 単調増加で，－から＋へ */
            v_m=vm-1;
            v_n=ceil(v_m*val/vl);
            v_y=vl-v_m;
            v_x=val-v_n;
            dl_ta=value-(v_y/v_x);                 /* 真値との誤差 */
            if (dl_ta<0) dl_ta=-dl_ta;
            if (dl_ta<=dlta) {                     /* 現時点の誤差最小分数を記録 */
                if (dl_ta<=dlta_min) {
                    dlta_min=dl_ta;
                    vm_min=v_m;
                    vn_min=v_n;
                }
            }
            else {
                if (dlta<=dlta_min) {
                    dlta_min=dlta;
                    vm_min=vm;
                    vn_min=vn;
                }
            }
        }
        m++;
    }

    m=(long int)vm_min;
    n=(long int)vn_min;
    y=(long int)(vl-vm_min);
    x=(long int)(val-vn_min);
    BA=(vl-vm_min)/(val-vn_min);
    dlta=value-BA;
    printf("\n近似分数\n");
    printf("m=%ld n=%ld  B=%ld / A=%ld  %.16f  %.11f\n",m,n,y,x,BA,dlta);

}
```

▶埼玉県・いずみ北斗



▶広島県・絵夢絶斗

P.S.背景部分だけ見んで"勘違い"しなかった人はおられますか？

（紅屋烔希）