応用解析
レポート

氏名: 今田将也 (IMADA, Masaya)
学生番号: 09430509

出題日: 2019 年 08 月 29 日
提出日: 2019 年 08 月 05 日
締切日: 2019 年 08 月 08 日

1　概要

応用解析レポート課題として，

$$\int_1^2 \left(\frac{1}{x^2} + \frac{3}{x^6}\right) dx$$

を台形積分およびシンプソン積分で計算を C 言語にて行わせた．

本レポートでは，その計算結果と考察をまとめた．また，ロンバーグ積分による計算も行い表にまとめた．

2　プログラムの使用法

本プログラムは台形積分とシンプソン積分を `n=2,4,8,16` の四通りの分割数で計算するためのプログラムである．処理結果を標準出力に出力する．プログラムは第 6 節に添付している．

3　結果

まず，プログラムを実行した台形積分とシンプソン積分の結果を以下に示す．

```
/*daikei*/
n=2
S1= 1.42812822
|S1-S|= 0.34687822

n=4
S1= 1.17841633
|S1-S|= 0.09716633

n=8
S1= 1.10644500
|S1-S|= 0.02519500
```

n	台形積分	誤差	シンプソン積分	誤差
2	1.42812822	0.34687822	1.18802512	0.10677512
4	1.17841633	0.09716633	1.09517903	0.01392903
8	1.10644500	0.02519500	1.08245456	0.00120456
16	1.08761162	0.00636162	1.08133383	0.00008383

表 1: 台形積分とシンプソン積分の結果と誤差の表

```
n=16
S1= 1.08761162
|S1-S|= 0.00636162

/*sympthon*/
n=2
S1= 1.18802512
|S1-S|= 0.10677512

n=4
S1= 1.09517903
|S1-S|= 0.01392903

n=8
S1= 1.08245456
|S1-S|= 0.00120456

n=16
S1= 1.08133383
|S1-S|= 0.00008383
```

なお，両方の数値積分の絶対誤差を表 1 にまとめた．

4 考察

ここでは，概要で挙げた以下の項目について考察を述べる．

1. 台形積分についての考察
2. シンプソン積分についての考察

4.1 台形積分についての考察

台形積分の結果を見ると S1-S の絶対誤差は，n=2 と n=4 を見ると約 0.25 誤差が小さくなっている．また，n=4 と n=8 を見ると約 0.07 だけ小さくなっており，n=8 と n=16 を見ると約 0.02 だけ小さくなっており，真の値に近づいていることがわかり，次第にその絶対誤差が小さくなっている．

n	$I_1(n)$	$I_2(n)$	$I_3(n)$	$I_4(n)$	$I_5(n)$
1	2.14843750	-	-	-	-
2	1.42812822	1.18802512	-	-	-
4	1.17841633	1.09517903	1.08898930	-	-
8	1.10644500	1.08245456	1.08160626	1.08148907	-
16	1.08761162	1.08133383	1.08125912	1.08125361	1.08125268

表 2: ロンバーグ積分の結果の表

また，相対誤差は n=2 で約 0.32 そして，n=16 で約 0.005 と減っているいることもわかった．

n=2 と n=4 では後者のほうが約 0.28 倍精度が良くなっており，n=4 と n=8 でも，約 0.26 倍精度が良くなっている．n=8 と n=16 においては，約 0.25 倍精度が

従って，台形積分は約 $\frac{1}{4}$ 倍ずつ誤差が少なくなると言えるだろう．

4.2　シンプソン積分についての考察

シンプソン積分の結果を見ると S1-S の絶対誤差は，n=2 と n=4 を見るとその誤差は約 0.092 小さくなっている．また，n=4 と n=8 を見ると約 0.012 だけ小さくなっており，n=8 と n=16 を見ると約 0.001 だけ小さくなっており，真の値に近づいていることがわかり，次第にその絶対誤差が台形積分よりはるかに小さくなっている．

また，相対誤差は n=2 で約 0.098 そして，n=16 で約 0.000077 と大きく減っているいることもわかった．

n=2 と n=4 では後者のほうが約 0.1304 倍精度が良くなっており，n=4 と n=8 でも，約 0.0864 倍精度が良くなっている．n=4 と n=8 でも，約 0.0695 倍精度が

従って，シンプソン積分は次第に約 $\frac{1}{16}$ 倍ずつ誤差が少なくなっていくと考える．

5　ロンバーグ積分

課題の式をロンバーグ積分にて，n_{max} まで行った結果も示す．

```
/*Romberg*/
I1( 1)= 2.14843750
I1( 2)= 1.42812822 I2( 2)= 1.18802512
I1( 4)= 1.17841633 I2( 4)= 1.09517903 I3( 4)= 1.08898930
I1( 8)= 1.10644500 I2( 8)= 1.08245456 I3( 8)= 1.08160626 I4( 8)= 1.08148907
I1(16)= 1.08761162 I2(16)= 1.08133383 I3(16)= 1.08125912 I4(16)= 1.08125361 I5(16)= 1.08125268
```

なお，結果を表 2 にまとめた

6　作成したプログラム

作成したプログラムを以下に添付する．

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>

#define S 1.08125000
#define a 1
#define b 2
```

```

double f(double x){
    double y;
    y=(1/x/x)+(3/x/x/x/x/x/x);
    return y;
}

void daikei(){
    int i,n,m;
    double S1,xi,yi,y0,yn,h;

    for(m=2;m<=16;m*=2){
        n=m;
        y0=f(a);
        yn=f(b);
        h=fabs(b-a)/n;
        S1=(y0+yn)/2.0;//最初と最後の項を計算しとく

        for(i=1;i<n;i++){//残りの項を足していく
            xi=a+h*i;//1つ隣へ
            yi=f(xi);
            S1+=yi;
        }
        S1=S1*h;
        printf("n=%d\nS1=%11.8f\n",m,S1);
        printf("|S1-S|=%11.8f\n\n",fabs(S1-S));

    }
}
void sympthon(){
    int i,n,m;
    double S1,xi,yi,y0,yn,h;

    for(m=2;m<=16;m*=2){
        n=m;
        y0=f(a);
        yn=f(b);
        h=fabs(b-a)/n;
        S1=(y0+yn);

        for(i=1;i<n/2;i++){//繰り返しはnまで/2
            xi=a+h*2*i;//二つ隣まで
            yi=4*f(xi-h)+2*f(xi);//公式4*F_1+2*F_2+4*F_3+...+2*F_n-1
            S1=S1+yi;
        }
        S1=S1+4*f(b-h);//4*F_n項目-1
        S1=h*S1/3;
        printf("n=%d\nS1=%11.8f\n",m,S1);
        printf("|S1-S|=%11.8f\n\n",fabs(S1-S));
    }
}
void Romberg(){
    //横に一段ずつ，上の段を使って計算していく
    //I(2),I(4)を先に計算しているわけではなさそう？
    int n,k,i,j,g;
    double h,T1,s,m,x,t;

    double T[10][10];

    n=1;
    h=b-a;
    T[1][1]=h*(f(a)+f(b))/2;//I(1);
    printf("I1(%2d)=%11.8f\n",n,T[1][1]);

    for(k=1;k<=log2(16.0);k++){
        s=0;
        //I(1)を使ってノート 6.2
        for(i=1;i<=n;i++){
            x=a+(i-0.5)*h;
            s=s+f(x);
        }
        s=(T[k][1]+h*s)/2;
        /*
        I(n/2)を使って I(n)を出す？ノート k + 1 段目の初期値？
        6.2()
```

```
        I(2)の値のこと？
        */

        h=h/2;//幅半分
        n=n*2;//分割数２倍

        m=1;
        for(j=1;j<=k;j++){/*漸化式のところ */
            t=T[k][j];//I(n/2)と初回は//I(1)
            T[k+1][j]=s;//I(n)を使って初回は//I(2)に値が入る
            m=m*4;
            s=(m*s-t)/(m-1);//I1(2)が出た
        }
        T[k+1][j]=s;//段の最後に結果を代入

        for(g=1;g<=k+1;g++){
            printf("I%d(%2d)=%11.8f ",g,n,T[k+1][g]);
        }
        printf("\n");
    }
    return;
}

int main(int argc, char *argv[]){
    printf("/*daikei*/\n");
    daikei();
    printf("/*sympthon*/\n");
    sympthon();
    printf("/*Romberg*/\n");
    Romberg();
    return 0;
}
```