シンプソン公式とリチャードソン補外

樋口さぶろお

龍谷大学理工学部数理情報学科

数値計算法 L09(2010-06-18)

今日の目標

1. 台形公式 → シンプソン公式で, 必要な精度が出る分割数で, 数値積分するプログラムが書けるようになろう.
2. 補間と補外の考えが説明できるようになろう.

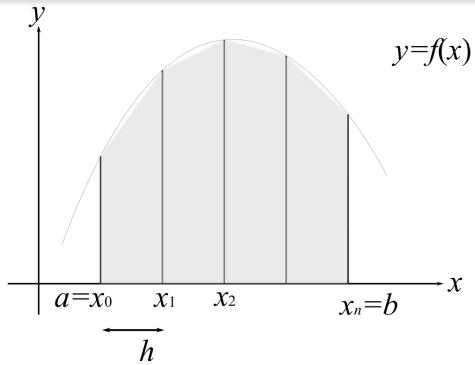
hig3.net

数値積分の台形公式

$$\int_a^b f(x)\,\mathrm{d}x = S_n + \mathcal{O}(n^{-2}).$$

$$S_n = \left[\frac{1}{2}f(x_0) + \sum_{j=1}^{n-1} f(x_j) + \frac{1}{2}f(x_n)\right] \times h$$

$$= \left[\frac{1}{2}f(a) + \sum_{j=1}^{n-1} f(a+jh) + \frac{1}{2}f(b)\right] \times h$$



- n 分割数

台形公式の漸増法

分割数 n での台形公式による近似.

$h = \frac{b-a}{n}$.

$$S_n = [\frac{1}{2}f(a) + \sum_{j=1}^{n-1} f(x+jh) + \frac{1}{2}f(b)] \times h.$$

分割数 $2n$ での台形公式による近似.

$h' = \frac{b-a}{2n} = \frac{h}{2}$.

$$\begin{aligned}
S_{2n} &= \left[\frac{1}{2}f(a) + \sum_{j=1}^{2n-1} f(x+jh') + \frac{1}{2}f(b)\right] \times h' \\
&= \left[\frac{1}{2}f(a) + \sum_{k=1}^{n-1} f(x+(2k)h') + \sum_{k=1}^{n} f(x+(2k-1)h') + \frac{1}{2}f(b)\right] \times h' \\
&= \left[\frac{1}{2}f(a) + \sum_{k=1}^{n-1} f(x+(k)h) + \frac{1}{2}f(b)\right] \cdot \frac{h}{2} + \left[\sum_{k=1}^{n} f(x+(2k-1)h')\right] h' \\
&= \frac{1}{2}S_n + \sum_{j=1,3,5,\ldots,2n-1} f(x+jh')h'
\end{aligned}$$

台形公式の漸増計算

栗原 §5.5

漸増計算の漸化式

S_n: n 分割での台形公式による近似値は, 次の漸化式で求められる.
$S_1 = \frac{1}{2}(f(a) + f(b)) \times (b - a)$

$$S_{2n} = \frac{1}{2}S_n + \sum_{j=1,3,5,\ldots,2n-1} f(x + jh')h'$$

上の公式を使って $S_1, S_2, S_4, \ldots$ と計算していって, だんだん正確にしていける. 誤差の評価値 $|S_{2n} - S_n|$ が指定の値より小さくなったらそこでやめる.

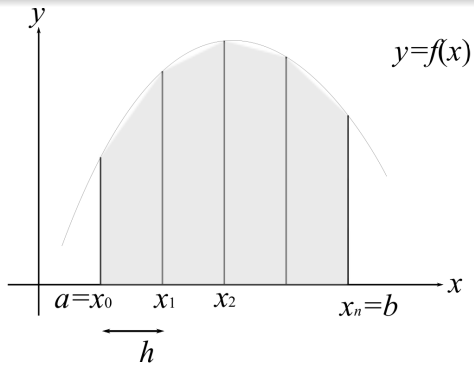
台形公式で，必要な精度が出るまで分割数を増やして数値積分アルゴリズム

```
1  double f(double x); /*被積分関数．与える．*/
2
3  int main(void){
4    double a=(); /* 下端．与える．*/
5    double b=(); /* 上端．与える．*/
6
7    double n=(); /* 分割数．与える．*/
8    double h;    /* 刻み幅 */
9    double eps=(); /*許せる誤差の上限．与える．*/
10
11   int j;
12   double s;
13   double sprev;
14
15   h=(b-a)/n;
16
17   n=1;
18   sprev=();    /* n=1 台形公式の結果 */
19   while(1){
20     n=n*2;
21     s=();  /* 分割数nの台形公式の結果．
22               先週のmainの中身みたい*/
23
24     if(fabs(s-sprev)<eps){
25       break;
26     }
27     sprev=s;
28   }
29   printf("%f\n",sum);
30   return 0;
31 }
```

数値積分のシンプソン公式

$$\int_a^b f(x)\,\mathrm{d}x = T_{2n} + \mathcal{O}((2n)^{-4}).$$

$$\begin{aligned} T_{2n} &= \frac{1}{3}(4S_{2n} - S_n) \\ &= \frac{1}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \ldots \\ &\quad + \ldots + 2f(x_{2n-2}) + 4f(x_{2n-1}) + f(x_{2n})] \times h \\ &= \frac{1}{3}\left[f(a) + f(b) + 4\sum_{k=1}^{n} f(x_{2k-1}) + 2\sum_{k=1}^{n-1} f(x_{2k})\right] \times h. \end{aligned}$$



- $2n$ 分割数

シンプソン公式で，必要な精度が出るまで分割数を増やして数値積分アルゴリズム

```
1  double f(double x); /*被積分関数．与える．*/
2
3  int main(void){
4    double a=(); /* 下端．与える．*/
5    double b=(); /* 上端．与える．*/
6
7    double n=(); /* 分割数．与える．*/
8    double h;    /* 刻み幅 */
9    double eps=(); /*許せる誤差の上限．与える．*/
10
11   int j;
12   double s;
13   double sprev;
14   double t;
15   double tprev;
16
17
18   h=(b-a)/n;
19
20   n=2;
21   sprev=();   /* n=2k=2 台形公式の結果 */
22   tprev=();   /* n=2k=2 シンプソン公式の結果 */
23   while(1){
24     n=n*2;
25     s=();  /* 分割数n=2kの台形公式の結果．
26               先週のmainの中身みたい*/
27     t=()   /* s と sprev からn=2kシンプソン公式の結果 */
28     if(fabs(t-tprev)<eps){
29       break;
30     }
31     sprev=s;
32     tprev=t;
33   }
34   printf("%f\n",sum);
35   return 0;
36 }
```

プチテスト Web 返却中