龙贝格积分上机报告

计试81 白思雨 2186123935

一、算法原理

对于复化梯形求积公式而言，由下式

$$I[f] \approx T_{2n} + \frac{1}{3}(T_{2n} - T_n)$$

$$|I[f] - T_{2n}| \approx \frac{1}{3}|T_{2n} - T_n|$$

知，可取积分的近似值为

$$I[f] \approx T_{2n} + \frac{1}{4-1}(T_{2n} - T_n)\overline{T}_{2n}$$

对于复化辛普森求积公式有

$$I[f] - S_n = \frac{b-a}{2880}h^4 f^4(\eta), a \leqslant \eta \leqslant b,$$

$$I[f] - S_{2n} = -\frac{b-a}{2880}\left(\frac{h}{2}\right)^4 f^4(\eta_1), a \leqslant \eta_1 \leqslant b.$$

假定$f^4(x)$在区间$[a,b]$上连续且变化不大，可以认为$f^4(\eta) \approx f^4(\eta_1)$，将以上两式相除得

$$\frac{I[f] - S_n}{I[f] - S_{2n}} \approx 16$$

由此解得

$$I[f] \approx S_{2n} + \frac{1}{4^2-1}(S_{2n} - S_n) \triangleq \bar{S}_{2n}$$

同理，对复化科茨求积公式有

$$I[f] \approx C_{2n} + \frac{1}{4^3-1}(C_{2n} - C_n) \triangleq \bar{C}_{2n}$$

下面对上述式子作进一步的分析，看一看它们的实质和它们之间的关系.

$$\begin{aligned}
\bar{T}_{2n} &= T_{2n} + \frac{1}{3}(T_{2n} - T_n) = \frac{1}{3}(4T_{2n} - T_n) \\
&= \frac{1}{3}\left\{4\left[\frac{1}{2}T_n + \frac{h}{2}\sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right)\right] - T_n\right\} \\
&= \frac{1}{3}\left\{T_n + 2h\sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right)\right\} \\
&= \frac{1}{3}\left\{\frac{h}{2}\left[f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)\right] + 2h\sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right)\right\} \\
&= \frac{h}{6}\left\{f(a) + 2\sum_{i=1}^{n-1} f(x_i) + 4\sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right) + f(b)\right\} = S_n
\end{aligned}$$

即

$$S_n = T_{2n} + \frac{1}{4-1}(T_{2n} - T_n)$$

同理可得

$$C_n = S_{2n} + \frac{1}{4^2-1}(S_{2n} - S_n)$$

令 $R_n = C_{2n} + \frac{1}{4^3-1}(C_{2n} - C_n)$，称为龙贝格积分公式。

龙贝格积分法是将积分区间$[a,b]$逐次分半，将上述公式逐次递推计算，以得到较高精度的积分近似值。龙贝格积分法的计算公式如下：

$$T_1 = \frac{b-a}{2}[f(a)+f(b)]$$

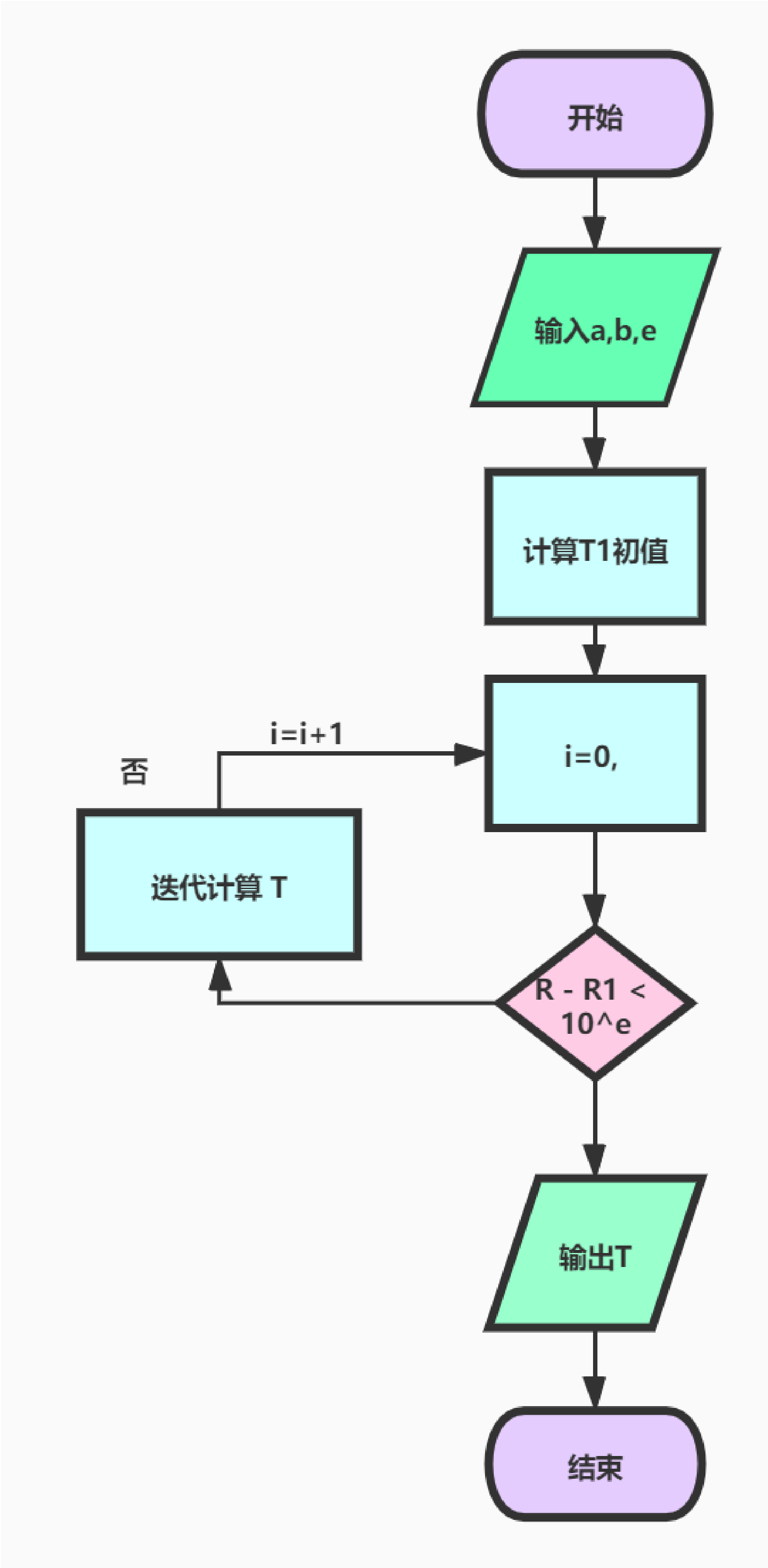
$$T_{2^{k+1}} = T_{2^k} + \frac{b-a}{2^{k+1}}\sum_{i=1}^{2^k} f\left(a+(2i-1)\frac{b-a}{2^{k+1}}\right), k=0,1,2,\cdots,$$

$$\begin{cases} S_{2^k} = T^{2^{k+1}} + \dfrac{1}{4-1}(T^{2^{k+1}} - T^{2^k}), \\ C_{2^k} = S^{2^{k+1}} + \dfrac{1}{4^2-1}(S^{2^{k+1}} - S^{2^k}),\ k=0,1,2,\cdots, \\ R_{2^k} = C^{2^{k+1}} + \dfrac{1}{4^3-1}(C^{2^{k+1}} - C^{2^k}), \end{cases}$$

计算过程按顺序进行。

若$|R_{2^{k+1}} - R_{2^k}| < \varepsilon$，则取$I[f] \approx R_{2^{k+1}}$；否则，继续计算，直到满足精度要求为止。

二、程序框图



三、程序及使用说明

```
#include<iostream>
#include<cmath>
using namespace std;

double a, b, fa, fb;
double T, T1, S, S1, C, C1, R, R1, e;

double fx1(double x)
{
	double fx = 1/(1 + x);
	return fx;
}

double fx2(int t, double a, double b)//T 的高次迭代函数
{
	double m = (1 << t), Tk = 0, Tkk, z;
	for(int i = 1;i <= m;++i)
	{
		z = a + (2 * i - 1) * (b - a) / (2 * m);
		Tk += fx1(z);
	}
	return Tkk = (b - a) * Tk / (m * 2);
}

int main()
{
	cout << "请输入积分下界 a: \n";		//输入 X 下界
	cin >> a;
	cout << "请输入积分下上界 b: \n";//输入 X 下界
	cin >> b;
	cout << "请输入计算精度 e: \n";
	cin >> e;
	//计算 f(x) 函数上下界的值
	fa = fx1(a);
	fb = fx1(b);
	//输出上下界函数值、上下界 fx 函数值、计算精度 e
	cout << "a = " << a << "\nb = " << b << "\nfa = " << fa <<
"\nfb = " << fb << "\ne = " << e << "\n";
	T1 = (b - a) * (fa + fb) / 2;
	for(int i = 0;;++i)
	{
		double Tkk = fx2(i, a, b);
		//cout << Tkk;
```

```
			T = T1 / 2 + Tkk;
			T1 = T;
			S = T + (T - T1) / (4 - 1);
			S1 = S;
			if(i)
			{
				C = S + (S - S1) /(4 * 4 - 1);
				C1 = C;
			}
			if(i > 1)
			{
				R = C + (C - C1) /(4 * 4 * 4 - 1);
				if(abs(R - R1) < pow(10,e))break;
				R1 = R;
			}
	}
	cout << "积分计算数值为: " << T;
	return 0;
}
```

说明:需要手动修改函数 fx1,其余相关说明已在代码中注释,

直接运行即可。

四、算例及计算结果

习题 6.2.1

```
C:\Users\62613\Desktop\数值分析\计试81_白思雨_实验报告\龙贝格积分\lbgjf.exe
请输入积分下界 a:
0
请输入积分下上界 b:
1
请输入计算精度 e:
-7
a = 0
b = 1
fa = 1
fb = 0.5
e = -7
积分计算数值为: 0.693147
--------------------------------
Process exited after 2.926 seconds with return value 0
请按任意键继续. . .
```

可见实验结果与原结果相符,实验成功,