

**《数值计算方法》第五章作业**

**姓名与学号**  王德茂（3220105563）

**指导教师**  徐祖华

**年级与专业**  机器人2202

**所在学院**  控制科学学院

**第五章作业**

1. **问题提出：**

假设通过某电阻器的电流可以表示为函数：



并且电阻是电流的函数，

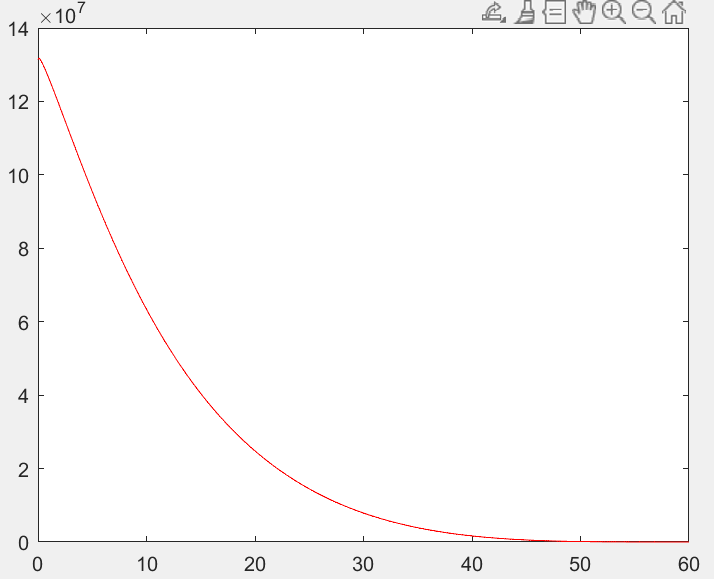


采用不同的数值积分方法计算t=0到60之间的平均电压，并分析误差。

1. **解决思路：**

这个可以直接采用老师所讲授的Newton-Cotes 积分、龙贝格（Romberg）积分以及高斯（Gauss）求积公式来计算。

在开始之前，我们画出函数的大致图像：



可以看出，函数的变化速度极快。

我们采用的主要脚本为:

I = @(x) (60 - x )^2 +(60 - x )\*sin(sqrt(x));

R = @(x) 10\*I(x) + 2\* I(x)^(2/3) ;

count = 50;

x = 0 : 60/count :60 ;

y = (count+1) ;

for i = 1 : 1 : count +1

    y(i) = I(x(i))\*R(x(i));

end

plot(x,y,"r-")

disp("当分割区间数为："+ count)

disp("trapezoid的结果：")

disp(trapezoid(y , count ));

disp("simpson的结果：")

disp(simpson(y , count ) );

disp("Romberg的结果：")

Romberg();

for tar = [5 ,10 , 50 ,100 ]

    disp("Gauss的结果：")

    disp("当分割区间数为："+ tar )

    disp(Gauss(tar))

end

1. **Newton-Cotes 积分**
2. 原理

在数值分析上，梯形法则和辛普森法则均是数值积分的方法。它们都是计算定积分的。

这两种方法都属于牛顿-柯特斯公式。它们以函数于等距n+1点的值，取得一个n次的多项式来近似原来的函数，再行求积。

1. 代码实现
2. 梯形公式

function [out] = trapezoid(y,count)

    sum = 0 ;

    for i = 2 : count

        sum = sum + 2 \* y(i) ;

    end

    out = 60 \*(sum + y(1) + y(count+1)) / 2 / count ;

end

1. Simpson公式

function [out] = simpson(y , count )

    h = 60 / count ;

    sum = 0;

    for i = 2 : 2 :count

        sum = sum + 4 \* y(i) + 2\* y(i+1);

    end

    sum = ( sum + y(1) - y(count+1) ) \* h /3 ;

    out = sum ;

end

1. 结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分割区间数 | trapezoid的结果： | 误差：% | simpson的结果： | 误差：% |
| 3 | 3.0696e+07 | 16.45 | 2.5973e+07 | -1.46 |
| 5 | 2.7714e+07 | 5.14 | 2.6009e+07 | -1.33 |
| 10 | 2.6614e+07 | 0.96 | 2.6247e+07 | -0.42 |
| 50 | 2.6359e+07 | 0.00 | 2.6353e+07 | 0.00 |
| 100 | 2.6359e+07 | 0.00 | 2.6358e+07 | 0.00 |

1. 误差分析

为了保证精度，采用复合求积的方法：先将积分区间分成几个小区间，并在每个小区间上用低阶Newton-Cotes公式计算积分的近似值，然后对这些近似值求和，从而得到所求积分的近似值。

这里一开始我们将分割数取值较高，所以精度已经比较高。

我们知道，复合梯形公式的误差可由以下计算得到：



**复合梯形公式具有二阶收敛性。**

**复合simpson公式具有四阶收敛性。**



1. **Romberg法**
2. 原理

龙贝格求积公式也称为逐次分半加速法。它是在梯形公式、辛普森公式和柯特斯公式之间的关系的基础上，构造出一种加速计算积分的方法。 作为一种外推算法，它在不增加计算量的前提下提高了误差的精度。

在等距基点的情况下，用计算机计算积分值通常都采用把区间逐次分半的方法进行。这样，前一次分割得到的函数值在分半以后仍可被利用,且易于编程。

1. 代码实现

I = @(x) (60 - x )^2 +(60 - x )\*sin(sqrt(x));

R = @(x) 10\*I(x) + 2\* I(x)^(2/3) ;

T = (5) ;

S = (4) ;

C = (3) ;

r = (2) ;

j = 0;

for count = 2 : 2 : 10

    j = j + 1 ;

    x = 0 : 60/count :60 ;

    y = (count+1) ;

    for i = 1 : 1 : count +1

        y(i) = I(x(i))\*R(x(i));

    end

    T(j) = trapezoid(y , count) ;

end

for i = 1 : 4

    S(i) = 4/3 \* T(i+1) - 1/3 \*T(i);

end

for i = 1 : 3

    C(i) = S(i+1) \*16 /15 - S(i) /15 ;

end

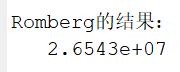
for i = 1 : 2

    r(i) = 64/63 \*C(i+1) - 1/63 \*C(i);

end

disp(r(2))

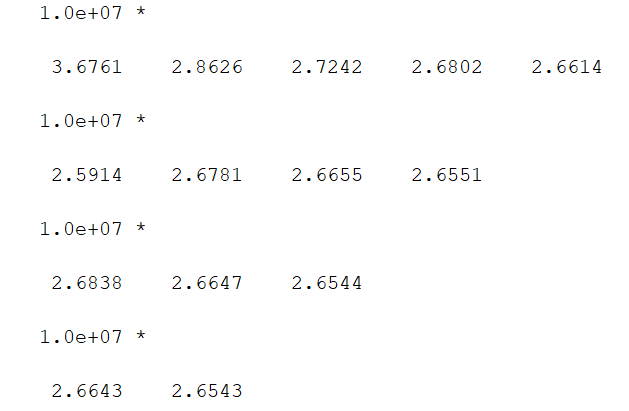
1. 结果



1. 误差

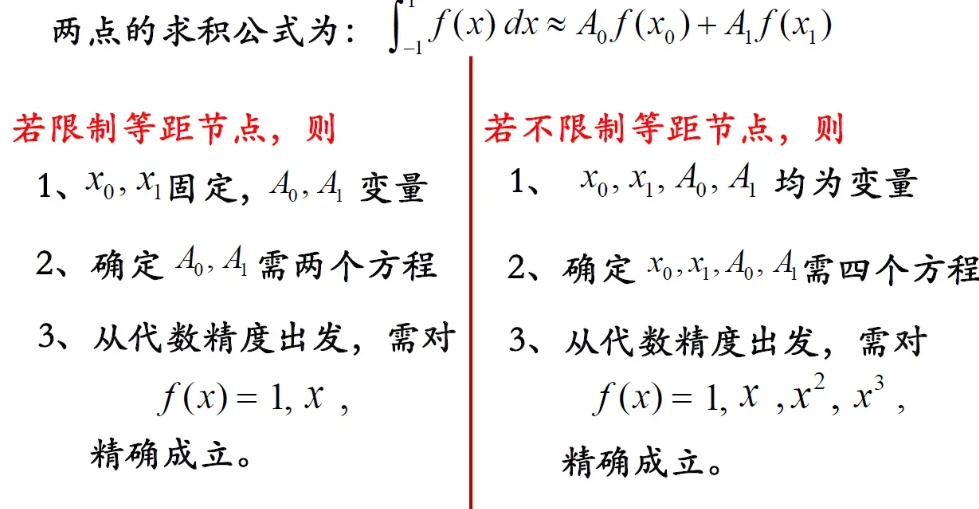
我们对最后所得到的结果进行计算，可以得到：

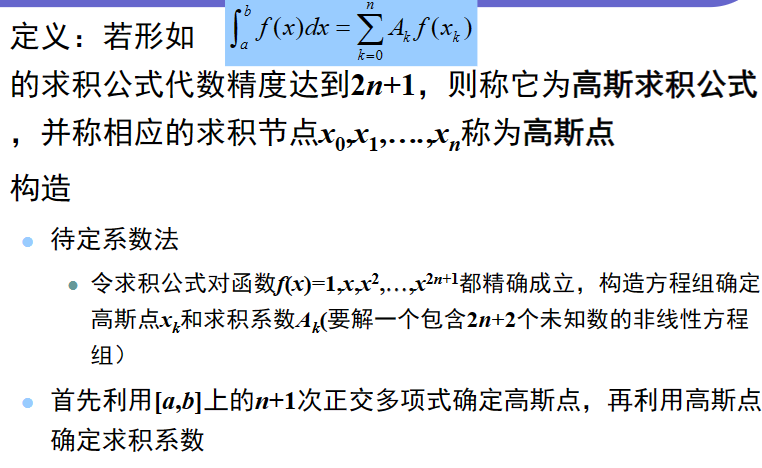
效果非常显著，而计算量，只需作少量的四则运算没有涉及到求函数值，可忽略不计。



1. **高斯（Gauss）求积公式**
2. 原理：

定理：n+1个求积结点的数值积分公式，其代数精度至多为2n+1





1. 代码实现：

function [out] = Gauss(tar)

    I = @(x) (60 - x )^2 +(60 - x )\*sin(sqrt(x));

    R = @(x) 10\*I(x) + 2\* I(x)^(2/3) ;

    x = 0 : 60 / tar : 60 ;

    sum = 0 ;

    for i  = 1 : 1 : tar

        temp = ( (x(i)+x(i+1) ) + (x(i+1)-x(i))\*(-sqrt(3)/3))/2 ;

        sum = sum + I(temp)\*R(temp) ;

        temp = ( (x(i)+x(i+1) ) + (x(i+1)-x(i))\*(sqrt(3)/3))/2 ;

        sum = sum + I(temp)\*R(temp) ;

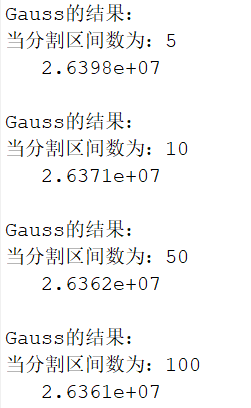
    end

    out = sum /tar \*30;

    out = out /60 ;

end

1. 结果



1. 误差分析：

我们知道，n+1个求积结点的数值积分公式，其代数精度至多为2n+1。

在这里，我们将积分区间分割成多个小区间，在每一个小区间内，我们采用了两点Gauss-Legendre公式。

|  |  |  |
| --- | --- | --- |
| 分割区间数 | 结果 | 误差% |
| 5 | 2.6398e+07 | 0.15 |
| 10 | 2.6371e+07 | 0.05 |
| 50 | 2.6362e+07 | 0.00 |
| 100 | 2.6361e+07 | 0.00 |

根据计算可得，GAUSS法的结果在计算量较少时即可达较高的精度。

1. **小结**

通过本次作业的实践，我们深入掌握了数值积分方法的应用，包括Newton-Cotes 积分、Romberg法和高斯求积公式。通过对比不同方法的结果和误差分析，我们更好地理解了数值积分的原理和应用场景。这次作业为我们提供了宝贵的实践经验，加深了对数值计算方法的理解和运用能力。