1 问题描述

Matlab内置函数humps为

$$f(x)=\frac{1}{(x-0.3)^2+0.01}+\frac{1}{(x-0.9)^2+0.04}-6$$

要求：
(1)采用数值微分方法计算函数在0.5处的一阶导数值，并分析误差。
(2)采用不同的数值积分方法计算函数在[0,1]上的积分结果，并分析误差。

2 问题分析

对f(x)求微分，得到解析解

$$f^{'}(x)=-\frac{2(x-0.3)}{((x-0.3)^2+0.01)^2}-\frac{2(x-0.9)}{((x-0.9)^2+0.04)^2}$$

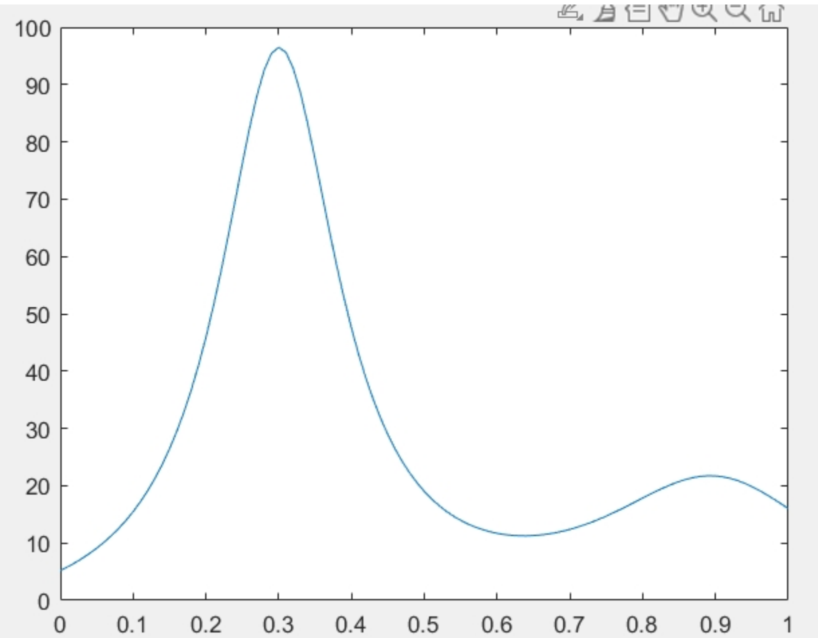
把0.5代入，$f^{'}(0.5)=-140$

对f(x)求积分，得到解析解

$$\int f(x)dx=10arctan(10x-3)+5arctan(5x-4.5)-6x+c$$

把[0，1]代入，用计算器算得$\int_0^1 f(x)dx=29.858325395498675089500892382438$

原函数图象为



3 代码实现

3.1 微分算法

差商近似的高精度微分公式，由

$$f'(x_i)=\frac{-f(x_{i+2})+4f(x_{i+1})-3f(x_i)}{2h}+O(h^2)$$

```
function result=diff(f,x0,h)
    result=(-f(x0+2*h)+4*f(x0+h)-3*f(x0))/(2*h);
end
```

一阶微分三点公式。

n=2，插值节点为$x_k=x_0+kh(k=0,1,2)$——一阶微分三点公式

$$f'(x_0)=\frac{1}{2h}\left[-3f(x_0)+4f(x_1)-f(x_2)\right]+\frac{h^2}{3}f^{(3)}(\xi)$$

三点向前差商

$$f'(x_1)=\frac{1}{2h}\left[-f(x_0)+f(x_2)\right]-\frac{h^2}{6}f^{(3)}(\xi)$$

中心差商

$$f'(x_2)=\frac{1}{2h}\left[f(x_0)-4f(x_1)+3f(x_2)\right]+\frac{h^2}{3}f^{(3)}(\xi)$$

三点向后差商

用中心差商公式

```
function result=diff_three(f,x0,h)
    result=(f(x0+h)-f(x0-h))/(2*h);
end
```

3.2 积分算法

Newton-Cotes积分

分别定义梯形公式，辛普森公式，辛普森3/8公式

```
function result=trape(f,X)
    result=(X(2)-X(1))*(f(X(2))+f(X(1)))/2;
end
function result=simpson(f,X)
    h=(X(3)-X(1))/2;
    result=h*(f(X(1))+4*f(X(2))+f(X(3)))/3;
end
function result=simpsom8(f,X)
    h=(X(4)-X(1))/3;
    result=3*h*(f(X(1))+3*f(X(2))+3*f(X(3))+f(X(4)))/8;
end
```

分别意味着区间等分为1，2，3块时，对应的X分别有2，3，4的端点

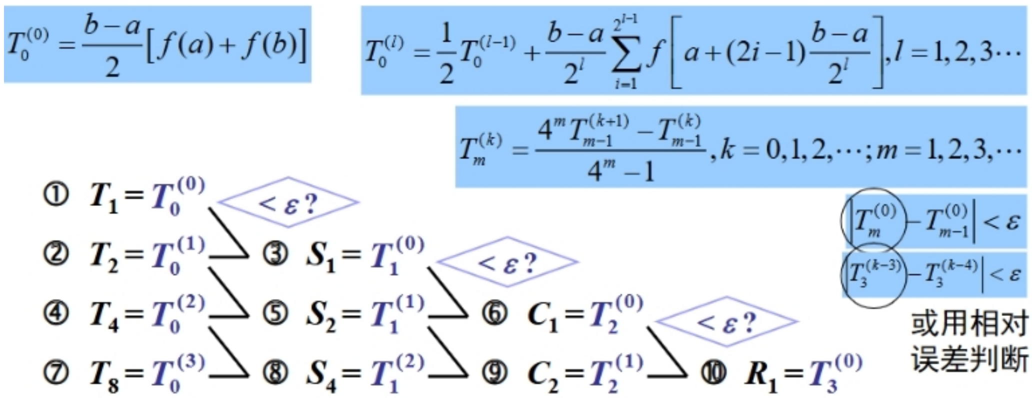
```
function X=separate(d,a,b)%均分点的函数
    X=[];
    h=(b-a)/(d-1);
    for i=1:d
        x=a+h*(i-1);
        X=[X,x];
    end
end
```

但是仅仅将其分成1、2、3块，精度远远不够。需要分成更多块，需要复合Newton-Cotes公式，其实本质上就是把这些块按照3、2或1将相邻块组合起来，再分别用对应的公式，再将每块的结果加起来。

将区间分成120块，分别全用辛普森3/8公式，全用辛普森公式，全用梯形公式和两个辛普森3/8公式加三个辛普森公式的组合。

```
x=separate(121,0,1);
sum1=0;%全用辛普森3/8公式
sum2=0;%全用辛普森公式
sum3=0;%全用梯形公式
sum4=0;%两个辛普森3/8公式加三个辛普森公式
for(i=1:40)
    sum1=sum1+simpson8(f_handle,x(3*i-2:3*i+1));
end
for(i=1:60)
    sum2=sum2+simpson(f_handle,x(2*i-1:2*i+1));
end
for(i=1:120)
    sum3=sum3+trape(f_handle,x(i:i+1));
end
for(i=1:20)
    sum4=sum4+simpson8(f_handle,x(3*i-2:3*i+1));
end
for(i=30:59)
    sum4=sum4+simpson(f_handle,x(2*i+1:2*i+3));
end
```

龙贝格积分



用如图所示的公式来计算

```
function [T,S,C,R]=romberg(k,f,a,b)
    T(1)=(b-a)*(f(b)+f(a))/2;
    for i=1:k
        sum=0;
        for j=1:2^(i-1)
            sum=sum+f(a+(2*j-1)*(b-a)/(2^i));
        end
        T(i+1)=T(i)/2+(b-a)*sum/(2^i);
    end
    for i=1:k
        S(i)=(4*T(i+1)-T(i))/3;
    end
    for i=1:k-1
        C(i)=(16*S(i+1)-S(i))/15;
    end
```

```
    for i=1:k-2
        R(i)=(64*C(i+1)-C(i))/63;
    end
end
```

其中k是最大指数，T是梯形序列，S是simpson序列，C是cotes序列，R是romberg序列。先求出全部的T序列，然后求出S，再依次求出C与R。

高斯积分

选用legendre多项式，默认上下限为[-1,1]，需要先转化原函数。

$$x = a_0 + a_1 x_d$$
$$a_0 = \frac{a+b}{2}$$
$$a_1 = \frac{b-a}{2}$$

有$x = \frac{1}{2} + \frac{1}{2}x_d$，$dx = \frac{1}{2}dx_d$

原积分变为：

$$\int_{-1}^{1}\left(\frac{1}{(0.5x_d+0.2)^2+0.01}+\frac{1}{(0.5x_d-0.4)^2+0.04}-6\right)\times\frac{1}{2}dx_d$$

n	x_k	A_k
1	0	2
2	±0.5773502692	1
3	±0.7745966692 0	0.5555555556 0.8888888889
4	±0.8611363116 ±0.3399810436	0.3478548451 0.6521451549
5	±0.9061798459 ±0.5384693101 0	0.2369268851 0.4786286705 0.5688888889
6	±0.9324695142 ±0.6612093865 ±0.2386191861	0.1713244924 0.3607615730 0.4679139346
7	±0.9491079123 ±0.7415311856 ±0.4058451514 0	0.1294849662 0.2797053915 0.3818300505 0.4179591837
8	±0.9602898565 ±0.7966664774 ±0.5255324099 ±0.1834346425	0.1012285363 0.2223810345 0.3137066459 0.3626837834

由节点与系数表，做出n=2~6的五个函数

```
function result=gauss2(f)
    result=f(0.5773502692)+f(-0.5773502692);
end
function result=gauss3(f)

 result=0.5555555556*f(0.7745966692)+0.5555555556*f(-0.7745966692)+0.8888888889*f
(0);
end
function result=gauss4(f)
    result=0.3478548451*(f(0.8611363116)+f(-0.8611363116))+0.6521451549*
(f(0.3399810436)+f(-0.3399810436));
end
function result=gauss5(f)
    result=0.2369268851*(f(0.9061798459)+f(-0.9061798459))+0.4786286705*
(f(0.5384693101)+f(-0.5384693101))+0.5688888889*f(0);
```

```
end
function result=gauss6(f)
    result=0.1713244924*(f(0.9324695142)+f(-0.9324695142))+0.3607615730*
(f(0.6612093865)+f(-0.6612093865))+0.4679139346*
(f(0.2386191861)+f(-0.2386191861));
end
```

本质上是纯加减乘除运算。

4 结果分析

```
function result=error(y_true,y_pred)
    result=abs(y_true-y_pred)/abs(y_true);
end
```

定义求相对误差的函数

4.1 微分结果

真实值为-140

步长取0.1	步长取0.01	步长取0.001
高精度微分公式：	高精度微分公式：	高精度微分公式：
-113.0656	-139.3320	-139.9926
0.1924	0.0048	5.2575e-05
一阶微分三点公式：	一阶微分三点公式：	一阶微分三点公式：
-178.7798	-140.3723	-140.0037
0.2770	0.0027	2.6572e-05

可见步长越小，预测值越准确，相对误差越小，而且一阶微分三点公式近似解的精度要高于高精度微分公式。

4.2 积分结果

真实值为29.858325395498675089500892382438

Newton-Cotes积分

不用复合公式时：

梯形公式	辛普森公式	辛普森3/8公式
10.5882	16.1961	39.5023
0.6454	0.4576	0.3230

此时辛普森3/8公式精度最高

用复合公式时，共分成120块：

全辛普森3/8	全辛普森	全梯形
29.8583	29.8583	29.8575
1.1394e-08	5.0449e-09	2.8704e-05

二十个辛普森3/8与三十个辛普森

29.8583

2.6371e-08

发现复合辛普森的误差最小，复合梯度的误差最大，而复合辛普森3/8与两者混用的误差均大于复合辛普森小于复合梯形。

符合理论值，理论上复合辛普森有四阶收敛性，收敛速度快，精度高；而复合梯度公式随着分的区间数增大，虽然总的误差会先变小，但是舍入误差会增大，最终精度受到限制。

龙贝格积分

10.5882	14.7941	30.1141	29.3312	29.8108	0.0016
16.1961	35.2207	29.0702	29.9707		0.0038
36.4890	28.6602	30.0307			0.0058
28.5359	30.0525				0.0065

自上而下每行依次为T，S，C，R序列。右侧自上而下为四个序列最后一个值的相对误差。

理论上其误差在四个序列中应该是递减的，即龙贝格积分精度最高。但是其真实结果是反过来的，可能是误差传递累积的问题。继续提高k的值。

当k=6时

10.5882	14.7941	30.1141	29.3312	29.8108	29.8463	29.8553
16.1961	35.2207	29.0702	29.9707	29.8581	29.8583	
36.4890	28.6602	30.0307	29.8506	29.8583		
28.5359	30.0525	29.8477	29.8585			

1.0092e-04

6.4035e-08

6.2093e-07

4.7543e-06

此时T序列的精度遇到了瓶颈，S序列精度最高。

当k=8时

10.5882	14.7941	30.1141	29.3312	29.8108	29.8463	29.8553	29.8576	29.8581
16.1961	35.2207	29.0702	29.9707	29.8581	29.8583	29.8583	29.8583	
36.4890	28.6602	30.0307	29.8506	29.8583	29.8583	29.8583		
28.5359	30.0525	29.8477	29.8585	29.8583	29.8583			

6.3068e-06

2.4278e-10

7.0547e-13

1.0968e-12

此时T与S序列的精度均不如C序列的精度，

当k=10时

列 1 至 10

10.5882	14.7941	30.1141	29.3312	29.8108	29.8463	29.8553	29.8576	29.8581	29.8583

列 11

29.8583

16.1961	35.2207	29.0702	29.9707	29.8581	29.8583	29.8583	29.8583	29.8583	29.8583
36.4890	28.6602	30.0307	29.8506	29.8583	29.8583	29.8583	29.8583	29.8583	
28.5359	30.0525	29.8477	29.8585	29.8583	29.8583	29.8583	29.8583		

3.9417e-07

9.4784e-13

1.1899e-16

3.5696e-16

此时C与R序列都精度极高且非常接近。均比T和S序列高。

由此我们可以得出，R序列的精度上限最高，但是到达精度上限要求的k也越大。因此才会出现在k较小时精度不如T或R序列的现象，因为T与R序列在k较小时就能表现出较高的精度，但是随着k增大，算法的上限低。

高斯积分

选用legendre多项式，n取2，3，4，5，6

n=2时	n=3时	n=4时	n=5时	n=6时
34.5155	19.3931	38.2042	27.8015	27.8952
0.1560	0.3505	0.2795	0.0689	0.0657

随着n的增大，误差逐渐变小，理论上n足够大，精度就足够。虽然这种算法的精度远不如前两种。但是高斯积分算法的运算及其简单，只涉及函数值的加减乘除运算，不涉及拟合、递归等等，公式简单。但是缺点是规律性不强，改变n需要改变所有函数值及其系数，没有通解，要取更高精度解只能查节点与系数表。

5 个人心得

对于微分问题，只要步长够小，无论是高精度微分公式还是一阶微分三点公式，其精度都足够。运用差商近似可以将微分转化为线性运算，简化了计算过程。

对于积分问题，Newton-Cotes积分的辛普森积分精度最高，收敛性也最好，将积分区间分成尽量小的区间，再运用复合辛普森公式，一般可以得到较为精确的结果，而且原始公式简单，复合公式也不复杂，计算量不大。Romberg积分收敛速度最快，精度最高，k=10的时候，误差的数量级已经可以达到10^{-16}，缺点是原始公式与算法较为复杂，计算量大，时间复杂度高。高斯积分法的公式最简单，缺点是没有通解，只能查系数节点表，而且精度较差，但是鉴于其朴素的原始公式，能够达到这种精度也是非常强大的。