数值分析 实验1-2

计52 路橙 2015010137

实验1 误差

实验要求

实验题目

已知$ln2 = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + ... + \frac{(-1)^{n-1}}{n} + ...$，令$x_n = \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k}$，则$x_n$构成逼近$ln2$的数列。根据交错级数和截断误差的知识，有估计式$|x_n - ln2| < \frac{1}{n+1}$。

记$|x_n - ln2| < \epsilon$，若取$\epsilon = \frac{1}{2} \times 10^{-4}$，试用单精度float计算$x_n$，问$n$何值时能满足精度要求？理论上的$n$值与实际计算的$n$值是否存在不同？为什么？（令$ln2$的准确值为0.693147190546）

初始数据

$\epsilon = \frac{1}{2} \times 10^{-4}$，$ln2 = 0.693147190546$

算法描述

伪代码

直接计算的方法

```
n <- 0
x <- 0
while |x - ln2| < epsilon:
    n++
    x += (-1)^(n-1) / n
```

即当x与$ln2$之差的绝对值大于ϵ时，不断增加x的项数，直到绝对误差小于给定的ϵ，则得到的n为第一个满足要求的n。（但误差较大，因为单次计算可能出现误差累积的偶然性）。

提高精度的计算方法

伪代码描述主要算法如下：

```
n <- 0
x <- 0
is_precise[0, 1, 2, 3, 4] <- false
while is_precise[] are not all true :
    n++
    x += (-1)^(n-1) / n
    if |x - ln2| > epsilon then:
```

```
        is_precise[0, 1, 2, 3, 4] <- false
    else:
        if is_precise[4] then break
        else if is_precise[3] then
            is_precise[4] <- true
        else if is_precise[2] then
            is_precise[3] <- true
        else if is_precise[1] then
            is_precise[2] <- true
        else if is_precise[0] then
            is_precise[1] <- true
        else is_precise[0] <- true
n <- n - 6 // 因为一共涉及了6次连续的判断，故开头的第一个n即为满足要求的结果。
```

即当x与$ln2$之差的绝对值大于ϵ时，不断增加x的项数，直到连续5次计算的绝对误差小于ϵ。这主要是考虑到单次计算可能有误差，需要保证精度在连续5次计算都收敛到对应精度才可以一定程度上确保计算的准确性。

程序编写时的主要问题

实验时要求使用**float**计算x_n，但float只能精确表示6到7位的精确有效数字，实验中给的初始数据$ln2$ 的有效数字共12位，这无法用float精确表示。

因此，为了满足要求及尽可能地提高精度，我采用如下方法：

1. 计算x_n时使用float计算，即通过如下方法计算x_n：

   ```
   float x = 0;
   ...
   x += (float)pow(-1, n - 1) / (float)n;
   ```

   由此得到的x_n是用float存储并进行中间计算的，满足了题目要求。

2. 在判断绝对误差时，为了利用$ln2$的准确值的精度，进行如下处理：

   ```
   double ln2 = 0.693147190546;
   double epsilon = 5e-5;
   ...
   while (abs(x - ln2) > epsilon)
   {
       ...
   }
   ```

   利用double可以准确存储12位有效数字精度的$ln2$。由于x为float，在while循环的判断时会被隐式类型转换为double，从而利用了$ln2$的12位精度，计算的结果也更精确。

程序清单说明

实验1在`1.cpp`下。使用的是标准c++库函数，直接编译即可运行。

其中`compute_float_x()`为x作为float存储并计算，`compute_float_x_precisely`为x作为float存储计算并且要求连续5次计算的绝对误差都满足要求才输出，`compute_double_x()`为x作为double类型存储并计算（用于比较，并不是实验要求）。

运行结果与分析

1. 程序运行结果：
    - x用float存储：当x的绝对误差小于给定ϵ时则输出，这样得到的$n=9013$；当要求连续5次绝对误差都小于给定ϵ时，得到的$n=11083$；
    - x用double存储：当x的绝对误差小于给定ϵ时则输出，这样得到的$n=9999$；当要求连续5次绝对误差都小于给定ϵ时，得到的$n=9999$；
2. 理论计算：由于$|x_n-ln2|<\frac{1}{n+1}$，为了保证误差的绝对值小于ϵ，一个充分条件是满足$\frac{1}{n+1}\leq\epsilon$，故需要保证$n\geq\frac{1}{\epsilon}-1=20000-1=19999$。
3. 对结果的分析：
    - 理论计算得到的n大于实际程序运行的结果，这是因为理论计算估计的上界并不精确，得到的n也只是一个充分条件，一定大于实际的n。
    - 同样的算法，x用float与用double存储得到的结果有一定的差距，这是因为float的有效数字只有7位，不断计算加法的过程中随着数字变化的值越来越小，舍入误差越来越明显，导致n较大时舍入误差逐渐增大，且有误差累积现象。但double类型的变量的有效位数可以满足$ln2$的12位有效位数，从而计算的结果很精确。
    - 连续5次绝对误差都小于ϵ时才返回，这样的算法一定程度上提高了计算的精度，即保证所得的x的值受误差累积的影响较小，在范围内收敛保证了得到的x一定满足要求。但由于float的浮点误差的舍入造成的数据波动，很难给出在某个确定的值之后x与$ln2$的绝对误差永远小于ϵ，但double类型可以满足在$n=9999$之后x与$ln2$的绝对误差都小于ϵ，这也进一步说明，float类型的浮点误差的积累会导致数据不断出现由于舍入造成的波动，序列的收敛性较差，但double类型在这样的精度上远远优于float类型，得到的序列波动小很多。

体会与展望

本次实验我较好地分析了float、double类型的精度导致的误差积累，以及体会到了实际计算与理论分析的不同。今后在编程的时候需要注意，对于精度要求较高的计算，一定要使用double类型，以减少误差积累造成的不利影响。

实验2 插值法

实验要求

实验题目

对 $[-5,5]$ 作等距划分 $x_i=-5+ih$，$h=\frac{10}{n}$，$i=0,1,\ldots,n$，并对 Runge 给出的函数 $f(x)=\frac{1}{1+16x^2}$ 作 Lagrange 插值何三次样条插值，观察 Runge 现象并思考改进策略。

1. 分别取 $n=10,20$ 作 Lagrange 代数插值 $L_{10}(x)$ 与 $L_{20}(x)$。
2. 分别取 $n=10,20$ 作第一类边界条件的三次样条插值 $S_{10}(x)$ 与 $S_{20}(x)$。
3. 给出 $f(x)$ 与 $L_{10}(x)$、$L_{20}(x)$、$S_{10}(x)$、$S_{20}(x)$ 在区间 $[-5,5]$ 的函数图像，观察其不同。
4. 考察上述两种插值函数在 $x=4.8$ 处的误差，并作分析和思考。

初始数据

给定区间[-5, 5]上插值节点为$x_i = -5 + ih$，$h = 10/n$，$i = 0, 1, 2, \dots n$

给定函数为$f(x) = \frac{1}{1+16x^2}$

算法描述

Lagrange插值

给定$n+1$个节点$x_0 < x_1 < \cdots < x_n$，要求构造一个n次插值多项式，满足$L_n(x_j) = y_j$，$j = 0, 1, \dots, n$，则称$L_n(x)$为拉格朗日插值多项式。

拉格朗日插值多项式的公式为：

$$L_n(x) = \sum_{k=0}^{n} y_k l_k(x)$$

其中$l_k(x)$为节点$x_0, x_1, \dots, x_n$上的n次插值基函数，它满足条件

$$l_k(x_j) = \begin{cases} 1, & k = j, \\ 0, & k \neq j, \end{cases} j, k = 0, 1, \dots, n$$

其计算公式为：

$$l_k(x) = \frac{(x - x_0)\cdots(x - x_{k-1})(x - x_{k+1})\cdots(x - x_n)}{(x_k - x_0)\cdots(x_k - x_{k-1})(x_k - x_{k+1})\cdots(x_k - x_n)}, k = 0, 1, \dots, n$$

因此，只需要计算出基函数，即可得到拉格朗日插值多项式。

第一类（一阶）边界条件的三次样条插值

第一类边界条件的三次样条插值公式为：

若$x \in [x_j, x_{j+1})$，则插值公式为：

$$S(x) = M_j \frac{(x_{j+1} - x)^3}{6h_j} + M_{j+1} \frac{(x - x_j)^3}{6h_j} + (y_j - \frac{M_j h_j^2}{6}) \frac{x_{j+1} - x}{h_j} + (y_{j+1} - \frac{M_{j+1} h_j^2}{6}) \frac{x - x_j}{h_j}, \\ j = 0, 1, \dots, n-1$$

其中$h_j = x_{j+1} - x_j, j = 0, 1, \dots, n-1$，而$M_j$的计算方法为如下线性方程的解：

$$\begin{bmatrix} 2 & \lambda_0 & & & \\ \mu_1 & 2 & \lambda_1 & & \\ & \cdots & \cdots & \cdots & \\ & & \mu_{n-1} & 2 & \lambda_{n-1} \\ & & & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \cdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \cdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

其中，

$$\mu_j = \begin{cases} 1, & j = n \\ \frac{h_{j-1}}{h_{j-1}+h_j}, & j = 1, 2, \ldots, n-1 \end{cases}$$

$$\lambda_j = \begin{cases} 1, & j = 0 \\ \frac{h_j}{h_{j-1}+h_j}, & j = 1, 2, \ldots, n \end{cases}$$

$$d_j = \begin{cases} \frac{6}{h_0}(f[x_0, x_1] - f'(x_0)), & j = 0 \\ 6f[x_{j-1}, x_j, x_{j+1}], & j = 1, 2, \ldots, n-1 \\ \frac{6}{h_{n-1}}(f'(x_n) - f[x_{n-1}, x_n]), & j = 0 \end{cases}$$

从而可通过使用"追赶法"求解上述方程。算法描述如下：

$$g_i = \begin{cases} \frac{d_0}{p_0}, & i = 0 \\ \frac{d_i - \mu_i g_{i-1}}{p_i}, & i = 1, 2, \ldots, n \end{cases}$$

$$r_i = \frac{\lambda_i}{p_i}, i = 0, 1, \ldots, n-1$$

$$p_i = \begin{cases} 2, & i = 0 \\ 2 - \mu_i r_{i-1}, & i = 1, 2, \ldots, n \end{cases}$$

接着可以通过如下递推得到M：

$$M_i = \begin{cases} g_n, & i = n \\ g_i - r_i M_{i+1}, & i = n-1, n-2, \ldots, 0 \end{cases}$$

至此，即可解得所需的三次样条插值函数。

程序清单说明

在`2.cpp`中包括了本次实验的代码，具体描述为：

- `f()`和`df()`函数为题目所给函数及其导数。
- `n_difference()`用递归的方法计算所给序列根据`f()`的n阶差分。
- `Lagrange()`函数用于计算拉格朗日插值。
- `Spline()`函数用于计算三次样条插值，其依赖的函数为：
  - `get_mu()`计算所有的μ_j的值。
  - `get_lambda()`计算所有的λ_j的值。
  - `get_d()`计算所有的d_j的值。
  - `solve_tridiagonal_matrix()`通过追赶法求解出所有的M_j。
  - `find_interval()`找出输入x属于插值节点的哪个区间，从而代入该区间的三次样条插值公式。
  - `get_S()`根据以上算出的参数得出最终的三次样条插值函数在x处的值。

运行结果与分析

对于给定的$x \in [-5, 5]$，运行`Spline(10, x)`、`Spline(20, x)`、`Lagrange(10, x)`、`Lagrange(20, x)`即可得出各个插值公式的结果。

1. 拉格朗日插值公式的结果如下：

$$
\begin{aligned}
L_{10}(x) = -6.35027 \times 10^{-5} x^{10} - 8.77783 \times 10^{-20} x^{9} + 0.00349662 x^{8} + 6.37922 \times 10^{20} x^{7} \\
- 0.0651818 x^{6} + 1.15715 \times 10^{-16} x^{5} + 0.489552 x^{4} - 5.47336 \times 10^{-17} x^{3} - 1.36898 x^{2} \\
- 1.50184 \times 10^{-16} x + 1
\end{aligned}
$$

$$
\begin{aligned}
L_{20}(x) = 5.56608 \times 10^{-8} x^{20} + 4.66411 \times 10^{-22} x^{19} - 5.36083 \times 10^{-6} x^{18} + 4.74261 \times 10^{-20} x^{17} \\
+ 0.000214093 x^{16} - 1.1642 \times 10^{-17} x^{15} - 0.00461755 x^{14} - 3.78374 \times 10^{-16} x^{13} \\
+ 0.0587499 x^{12} - 3.00659 \times 10^{-15} x^{11} - 0.452759 x^{10} - 1.11493 \times 10^{-14} x^{9} + 2.09177 x^{8} \\
- 2.25027 \times 10^{-14} x^{7} - 5.54502 x^{6} - 2.96517 \times 10^{-14} x^{5} + 7.72654 x^{4} - 9.42634 \times 10^{-15} x^{3} \\
- 4.81604 x^{2} - 8.12252 \times 10^{-16} x + 1
\end{aligned}
$$

2. 对于三次样条插值公式，这里给出M_j和h_j的对应取值：
   - $$
     h_j = \frac{10}{n} = \begin{cases} 1, & n = 10 \\ 0.5, & n = 20 \end{cases} \\
     j = 0, 1, \ldots, n
     $$
   - 而对于M_j：
     - $n = 10$时：

参数	数值
M_0	0.0251116
M_1	-0.0478097
M_2	0.175776
M_3	-0.6224
M_4	2.52353
M_5	-4.08529
M_6	2.52353
M_7	-0.6224
M_8	0.175776
M_9	-0.0478097
M_{10}	0.0251116

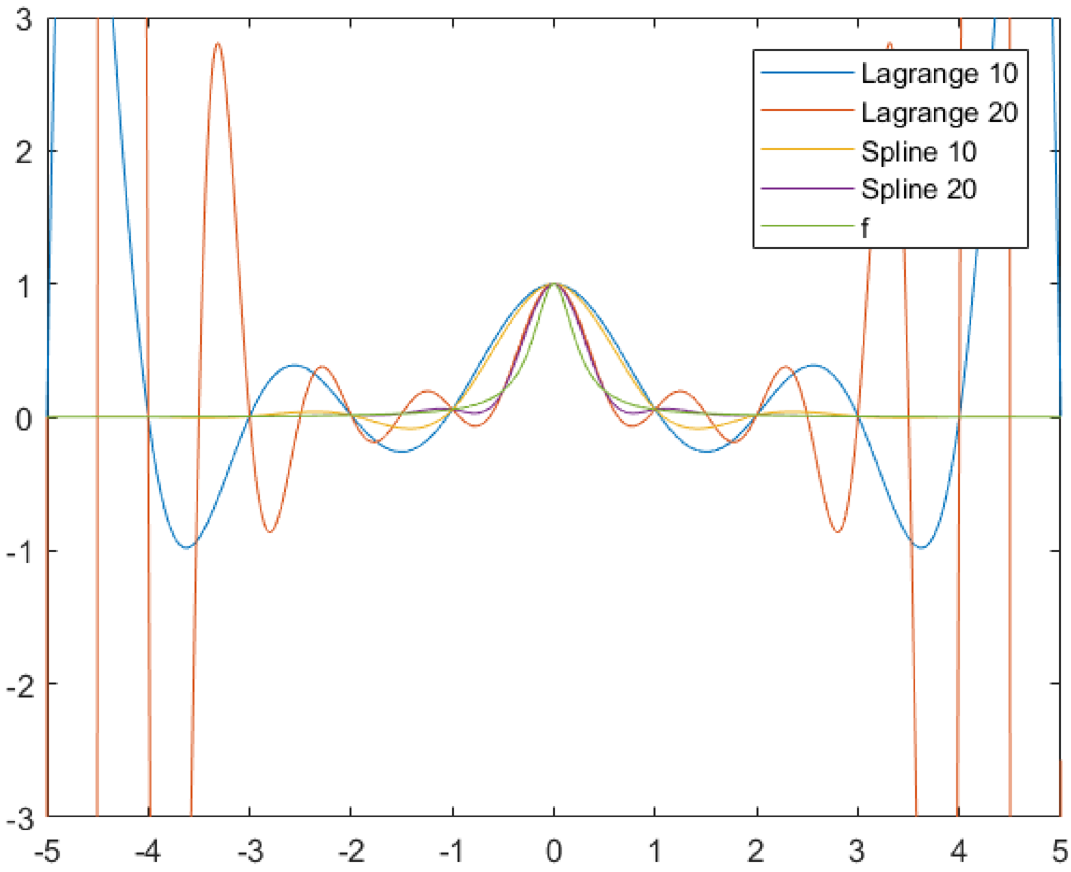
     - $n = 20$时：

参数	数值
M_0	0.000498448
M_1	0.00105866
M_2	0.000810192
M_3	0.00460371
M_4	-0.0039774
M_5	0.0397226
M_6	-0.0954124
M_7	0.489738
M_8	-1.37984
M_9	7.65474
M_{10}	-13.4274
M_{11}	7.65474
M_{12}	-1.37984
M_{13}	0.489738
M_{14}	-0.0954124
M_{15}	0.0397226
M_{16}	-0.0039774
M_{17}	0.00460371
M_{18}	0.000810192
M_{19}	0.00105866
M_{20}	0.000498448

插值函数公式

插值函数图像

绘制以上4个函数的图像如下：



可以看到，拉格朗日插值在$|x| > 3$时误差已经极大，不能用于插值估计，而三次样条插值可以较好地估计函数的值，可以作为函数的近似。

误差分析

在$x = 4.8$处，各个函数的函数值以及绝对误差为：

函数	函数值	绝对误差
$f(x)$	0.002705	0
$L_{10}(x)$	5.15132	5.14862
$L_{20}(x)$	-1080.74	1080.74
$S_{10}(x)$	0.003098	0.000393
$S_{20}(x)$	0.002704	0.000001

可以看出，拉格朗日插值在n较大时可观察到Runge现象，即随着n的增大，误差也相应地增大，这是由于n越大时拉格朗日插值的多项式次数也越大，在x较大时更快地趋于无穷大，因此误差会相应地增大。

而对于三次样条插值，随着n的增大，误差也会相应地减小，从而可以更好地近似原函数。

体会与展望

本次实验我手动实现了拉格朗日插值和三次样条插值，对算法的理解更深入了，也掌握了“追赶法”的算法，感觉十分有收获。此外，通过实验的观察，也明显地可以体会到两种插值方法的不同，优劣也高下立判。