数值分析上机习题报告 (1)

张宏毅 1500017736

March 5, 2017

Problem 1

Description

利用 Taylor 展开公式

$$\mathrm{e}^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

编一段小程序上机用单精度计算 e^x 的函数值。分别取 $x = 1, 5, 10, 15, 20, -1, -5, -10, -15, -20$, 观察所得结果是否合理，如不合理请分析原因并给出解决方法。

Solution

我们利用 e^x 的 Taylor 级数的前 n 项和来作为 e^x 的近似值。在计算的过程中，我们可以认为，当通项 $a_k = x^k/k!$ 的绝对值足够小（比如小于 $\epsilon = 10^{-10}$）时，部分和已经基本收敛，近似值的计算就可以结束了。在实际计算中，注意到通项 $a_k = x^k/k!$ 之间存在关系 $a_k = a_{k-1} \cdot (x/k)$, 因而通项可以通过递推得到，免去重复计算。具体代码（C）如下：

```c
#include <stdio.h>
#include <math.h>

int main() {
   int x[] = {1, 5, 10, 15, 20, -1, -5, -10, -15, -20};
   for (int i = 0; i < 10; ++i) { // For each x
      float ans = 0, add = 1;     // Initialize ans
      for (int k = 1; fabs(add) > 1e-10; ++k) {
         ans += add;              // Add a_{k-1} to ans
         add = add * x[i] / k;    // Calculate a_k
      }
      printf("exp(%d) = %f\n", x[i], ans);
   }
}
```

计算得到的数值结果列表如表 1(a)。对于 x 是正数的情况，估计值基本符合预期，具有较小的相对误差。而对于 x 是负数的情况，计算中产生了非常大的相对误差，甚至出现了指数函数结果为负的情形！这是非常不合理的。

表 1: 利用 Taylor 展开计算 $\exp(x)$

(a) 原始的数值计算结果

x	e^x 的估计值	x	e^x 的估计值
1	2.718282	−1	0.367879
5	148.413193	−5	0.006738
10	22026.468750	−10	−0.000063
15	3269017.500000	−15	0.021234
20	485165216.000000	−20	−2.756675

(b) 改进的数值计算结果

x	e^x 的估计值
−1	0.367879
−5	0.006738
−10	0.000045
−15	0.000000
−20	0.000000

究其原因，主要是因为当 $x < 0$ 时，通项 $a_k = x^k/k!$ 的符号是交错的，就会很容易出现两个绝对值相近的数字相减的情形，导致有效数字丢失。这样有效数字的丢失所带来的误差不断被放大，最后甚至会导致计算结果中一个有效数字都没有（如 $x = -20$ 时）。一个可行的解决方法为：当 $x < 0$ 时，利用等式 $\mathrm{e}^x = 1/\mathrm{e}^{-x}$，先计算 e^{-x} 的近似值再取倒数，这样就可以避免两个符号相反的数字相加的情形，且保证了最终结果一定不会为负。经过改进后求得的数值结果列表如表 1(b)。

Problem 2

Description

对于积分

$$I_n = \int_0^1 \frac{x^n}{x+5}\mathrm{d}x, \qquad n = 0, 1, 2, \cdots,$$

（1）证明递推公式

$$\begin{cases} I_0 = \ln 1.2, \\ I_n = -5I_{n-1} + \dfrac{1}{n}, \qquad n = 1, 2, \cdots; \end{cases}$$

（2）用上述递推公式计算 $I_1, I_2, \cdots, I_{20}$，观察数值结果是否合理，并说明原因。

Solution

先证明（1）中的递推公式成立。容易计算得初始条件

$$I_0 = \int_0^1 \frac{1}{x+5}\mathrm{d}x = \ln(x+5)|_0^1 = \ln 1.2.$$

且当 $n \geq 1$ 时有递推关系

$$I_n = \int_0^1 \frac{x^{n-1}(x+5) - 5x^{n-1}}{x+5}\mathrm{d}x = \int_0^1 x^{n-1}\mathrm{d}x - 5\int_0^1 \frac{x^{n-1}}{x+5}\mathrm{d}x = -5I_{n-1} + \frac{1}{n}.$$

因而（1）中的递推公式成立。利用该递推公式依次计算 $I_1, I_2, \cdots, I_{20}$，具体代码如下：

```c
#include <stdio.h>
#include <math.h>

int main() {
   float integral = log(1.2);
   for (int i = 1; i <= 20; ++i) {
      integral = -5 * integral + 1.0/i;
      printf("I_%d = %f\n", i, integral);
   }
}
```

计算得到的数值结果如表 2 所示。从表中可以看出，当 $n < 10$ 时，计算的结果至少还在区间 $(0, 1)$ 内，尚可接受，而当 $n > 10$ 时，积分的估计值出现了不可逆转的误差。理论上，简单分析可知 $0 < I_{n+1} < I_n \leq I_1 < 1$，即 $\{I_n\}$ 是位于区间 $(0, 1)$ 内的单调递减数列，因而 $n \geq 10$ 时的数值计算结果既不合理也不可靠。

表 2: 利用递推公式（1）计算 I_n

n	I_n 的估计值	n	I_n 的估计值	n	I_n 的估计值	n	I_n 的估计值
1	0.088392	6	0.024381	11	−0.161940	16	550.045654
2	0.058039	7	0.020951	12	0.893034	17	−2750.169434
3	0.043138	8	0.020245	13	−4.388246	18	13750.903320
4	0.034309	9	0.009886	14	22.012659	19	−68754.460938
5	0.028457	10	0.050570	15	−109.996628	20	343772.375000

对该现象可以做出如下原因分析：设积分的精确值为 I_n，近似值为 $\tilde{I}_n$，则序列 $\{I_n\}$ 和 $\{\tilde{I}_n\}$ 均符合递推公式（1），由此知 $(I_n - \tilde{I}_n) = -5(I_{n-1} - \tilde{I}_{n-1})$。若记 $e_n = |I_n - \tilde{I}_n|$ 为计算积分 I_n 时的绝对误差，则 $e_n = 5e_{n-1}$，进而有

$$e_n = 5^n e_0.$$

这也就意味着，如果在计算 $I_0 = \ln 1.2$ 时产生了一个绝对误差 e_0，那么在递推的过程中，绝对误差会被指数级地放大。当 $n = 10$ 时，放大的倍数 $5^{10} \approx 10^7$，导致了非常严重的有效数字丢失，因而利用该递推公式来计算 I_n 是不合适的。可以考虑提高数字精度、改变递推方式或者采用其他的数值积分方法来减小误差（双精度浮点数比单精度在这个问题中产生的计算误差要小得多）。