计算方法编程练习：级数求和

王章瀚　PB18111697

2020 年 3 月 19 日

1 Introduction

本次作业要求在给定的x下求以下级数的近似值：

$$\psi(x) = \sum_{k=1}^{+\infty} \frac{1}{k(k+x)}$$

其中分别取$x = 0.0,\ 0.5,\ 1.0,\ \sqrt{2},\ 10.0,\ 100.0,\ 300.0$，计算$\psi(x)$的近似值，要求截断误差在$10^{-6}$内。

输出格式：x和$\psi(x)$的值

$x = 0.0, y = 1.644934066848226$

$x = 0.5, y = 1.227411277760219$

2 Method

2.1 Method 1

由于题目要求的是近似值，且要求限制是截断误差。故应先计算，在截断误差的要求范围内，k应算到哪一步。

对于这个函数，由于题目要求的x满足$x \geq 0$，故对较大的k有：

$$\begin{aligned}
E(x) &= \sum_{k=MIN_K+1}^{+\infty} \frac{1}{k(k+x)} \\
&\leq \sum_{k=MIN_K+1}^{+\infty} \frac{1}{k^2} \\
&\leq \int_{MIN_K}^{+\infty} \frac{1}{k^2}\, dk \\
&= \frac{1}{MIN_K} \leq 10^{-6}
\end{aligned}$$

因此有：

$$MIN\_K \geq \frac{1}{10^{-6}} = 10^6$$

该方法的实现代码及效果放在了附录A.

2.2 Method 2

此外，根据课程主页上原题的提示，还有优化后的方法，即注意到$\psi(1) = 1$的事实，可以考虑求

$$\frac{\psi(x) - \psi(1)}{1 - x} = \sum_{k=1}^{+\infty} \frac{1}{k(k+1)(k+x)}$$

易见，这个函数相较于原本的$\psi(x)$收敛速度更快。 考虑到$0 \leq x \leq 300$类似地，可以算出原函数的截断误差$E(x)$满足，

$$\begin{aligned}
E(x) &\leq 300 \sum_{k=MIN_K+1}^{+\infty} \frac{1}{k(k+1)(k+x)} \\
&\leq \sum_{k=MIN_K+1}^{+\infty} \frac{300}{k^3} \\
&\leq \int_{MIN_K}^{+\infty} \frac{300}{k^3}\, dk \\
&= \frac{300}{2MIN_K^2} \leq 10^{-6}
\end{aligned}$$

因此有：

$$MIN\_K \geq \sqrt{\frac{150}{10^{-6}}}$$

这样的构造使得计算量大大减少。 该方法的实现代码及效果放在了附录B.

3 Result

按照Method1的算法，其输出结果如下：

x=0.000000,y=1.644933066848770e+00

x=0.500000,y=1.227410277760964e+00

x=1.000000,y=9.999990000010476e-01

x=1.414214,y=8.749819960221313e-01

x=10.000000,y=2.928958254023105e-01

x=100.000000,y=5.187277522689390e-02

x=300.000000,y=2.094121308480047e-02

按照Method2的算法，其输出结果如下：

x=0.000000,y=1.644934066826041e+00

x=0.500000,y=1.227411277749111e+00

x=1.000000,y=1.000000000000000e+00

x=1.414214,y=8.749829960301527e-01

x=10.000000,y=2.928968255968175e-01

x=100.000000,y=5.187377737541299e-02

x=300.000000,y=2.094221956986198e-02

从上述答案中可见，尽管使用Method1的算法在截断误差要求为10^{-6}的情况下，也能满足截断误差在10^{-6}范围内的要求，但如果提高截断误差精确度，则计算过于缓慢。相比之下Method2的算法，能够更快地得到计算结果。

4 Discussion

本题中，由于要求的截断误差不那么严格，因此暴力计算的方法并不会对效率造成太大影响。但如果要求精度更高，则需要考虑对要求的函数稍作修改，从而提高收敛速度等。

A Computer Code of Method 1

```
#include <cstdio>
#include <cmath>

// 设置允许截断误差值
```

```cpp
const double TRUNCATION_ERROR = 1e-6;
// 原方法下，计算在允许截断误差下的应计算到多少位k
const double MIN_K_0 = 1/TRUNCATION_ERROR;

// 们及的个数xx
const double array_x[] = { 0.0, 0.5, 1.0, pow(2, 0.5), 10.0, 100.0, 300.0 };
const int X_NUM = 7;

// 原方法
double psi_0(double x) {
    double answer = 0;
    double k = 1;
    // 直接对1/((x+k)x)求和
    while (k <= MIN_K_0) {
        answer += 1/(k*(k+x));
        k++;
    }
    return answer;
}

int main()
{
    for (double i : array_x) {
        printf("x=%lf,y=%.15e\n", i, psi_0(i));
    }
}
```

B Computer Code of Method 2

```cpp
#include <cstdio>
#include <cmath>
using namespace std;

// 设置允许截断误差值
const double TRUNCATION_ERROR = 1e-6;
// 新方法下，计算在允许截断误差下的应计算到多少位k
const double MIN_K = 150/pow(TRUNCATION_ERROR, 0.5);
// 原方法下，计算在允许截断误差下的应计算到多少位k
const double MIN_K_0 = 1/TRUNCATION_ERROR;

// 们及的个数xx
const double array_x[] = { 0.0, 0.5, 1.0, pow(2, 0.5), 10.0, 100.0, 300.0 };
const int X_NUM = 7;
```

```

// 经提示改进的方法
double psi(double x) {
    double answer = 0;
    double k = 1;
    // 先计算(psi(x)-psi(1))/(1-x)，其收敛速度更快
    while (k <= MIN_K) {
        answer += 1/(k*(k+x)*(k+1));
        k++;
    }
    return 1 + (answer)*(1 - x);
}

int main()
{
    for (double i : array_x) {
        printf("x=%lf,y=%.15e\n", i, psi_0(i));
    }
    printf("\n");

    for (double i : array_x) {
        printf("x=%lf,y=%.15e\n", i, psi(i));
    }
}
```