Department: Tsinghua Shenzhen International Graduate School
Course Title: Engineering Mathematics
Class: Shenzhen Big Data 22 class
Name: Tianhe Wu
ID: 2022214378
Major: Big Data Enginnering

Experiment 1: Series Summation of Hamming

Date: 2022.10.6

The code of this experiment and the output values of each task are open source:
https://github.com/TianheWu/Tsinghua-Engineering-Mathematics

NOTES:

1. All output values are saved in .txt format.
2. The final_results.txt is the output value of problem (b) and the output values of all comparative experiments are stored in the results folder.

---

1. 实验题目名称

   Hamming 级数求和问题：

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

   取 2001 个 x 值，即 $x = 0.000, 0.001, 0.002, ..., 2.000$，计算所有关于 x 的级数，并要求**误差**控制在 $0.5e-12$ 精度范围。

   (a) 分别计算级数中 $k = 1000$ 及 $k = 100000$ 时结果并加以分析，比较两者算法复杂度；
   (b) 给出合适的级数求和误差控制算法，考虑如何减少计算的步骤和时间；
   (c) 分析新算法的误差，并对比评价新旧算法。

2. 目的和意义

   Hamming 级数求和是一个著名的问题，在做完所有实验后，我认为其最重要的意义就是**提升分母幂次，降低时间复杂度**。该问题引发了一系列对在给定误差范围内降低算法复杂度的思考。

3. 算法

   针对问题 (a)，有以下分析。

   我们先对 Hamming 级数求和问题给出最原始的计算方式：

$$\sum_{k=1}^{N} \frac{1}{k(k+x)} = \sum_{k=1}^{\infty} \frac{1}{k(k+x)}, N \longrightarrow \infty$$

可以发现，当我们的 N 取到无穷的时候，就可以逼近真实解，结果可以被控制在误差范围内。下方给出对应的伪代码求解算法：

Algorithm 1: Basic Algorithm of Series Summation of Hamming

input : The maximum value of x and the value of N

output: The value $\psi(x)$ of series summation of Hamming for each x

for $x \leftarrow 0.000$ **to** 2.000 **do**
  $\psi(x)$=0;
  for $k \leftarrow 1$ **to** N **do**
    $\psi(x) \leftarrow \psi(x) + \frac{1}{k(k+x)}$;
    $k \leftarrow k+1$;
  end
  $x \leftarrow x + 0.001$;
end

对于 $k = 1000$ 与 $k = 100000$ 时，上述算法用时为 $5ms$ 与 $552ms$。我们将其与在误差 $0.5e-12$ 内的结果（可看作精确值）进行对比，计算每一项与精确值的差的绝对值，最后求和得到 S。

$$S = \sum_{i=1}^{M} |\hat{y}_i - y_i|$$

不同 N 值与对应的运行时间还有与精确值的差值如下表 1 所示：

Table 1: Error results and running time for different values of N

N	1e3	1e4	1e5	1e6	1e7	1e8
S	1.99900	0.20000	0.02000	0.00200	0.00020	1.91e-5
Running time(ms)	5	56	552	5451	54818	532882

结论：可以观察到当计算级数和时，计算 k 的次数越多时，也就是 N 越大时计算的误差总和 S 越小，而如果不对其进行优化的话即使 N 的取值达到了 $1e8$，S 已经很小，很接近 $\varepsilon = 0.5e-2$ 的误差要求，只要继续增大 N，一定可以满足误差要求。但是同时可以从图 1 观察到，算法的运行时间也在恐怖地增加。由上表或下图可以发现，当 x 的数量 M 固定时，运行的时间是线性增加的。上述算法的运行时间复杂度为 $O(MN)$，空间复杂度为 $O(1)$。比较 $k = 1000$ 与 $k = 100000$ 时，只需代入即可。

针对问题 (b) 的误差控制算法构建，有以下分析。

令 M 为 x 的个数，此时算法的时间复杂度为 $O(MN)$。由于给定的 x 值是有限固定的 M，所以为了减少计算的时间复杂度，只有采取降低 N 的措施。而 N 的取值影响的是结果 $\psi(x)$ 的精确度。因此，可以从对 $\psi(x)$ 误差的分析进行入手。想要达到的预期是在 N 值降低时，保证精确度。

当 k 的上限取到 N 时，此时的误差为：

$$\sum_{k=N+1}^{\infty} \frac{1}{k(k+x)} \leq \sum_{k=N+1}^{\infty} \frac{1}{k^2} \leq \varepsilon$$

其中 ε 为规定的误差大小，可以发现对相同的 N 来讲存在如下关系：

$$\sum_{k=N+1}^{\infty} \frac{1}{k^r} < \cdots < \sum_{k=N+1}^{\infty} \frac{1}{k^4} < \sum_{k=N+1}^{\infty} \frac{1}{k^3} < \sum_{k=N+1}^{\infty} \frac{1}{k^2} \leq \varepsilon$$

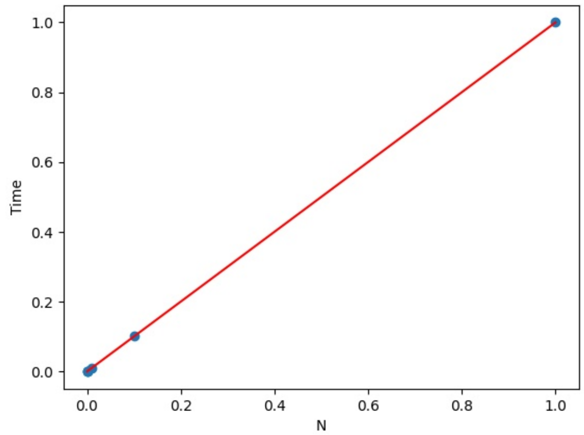

Figure 1: Running Time with N

可以观察到，如果此时要求每一个误差均相等，则当 r 越小时，所对应的 N 越大。可以通过如下方式证明：

$$\sum_{k=N+1}^{\infty} \frac{1}{k^r} < \int_N^{\infty} \frac{1}{x^r} dx, k > 1, r \geq 1$$

由于题目的已知条件，k 与 r 满足上述条件，令现有误差小于题目给定的误差：

$$\sum_{k=N+1}^{\infty} \frac{1}{k^r} < \int_N^{\infty} \frac{1}{x^r} dx = \frac{1}{r-1} N^{1-r} \leq \varepsilon$$

可得：

$$N \geq \sqrt[1-r]{(r-1)\varepsilon} = f(r) \tag{1}$$

由 (1)可得，当误差 ε 给定时，r 越大，N 越小。因此可以通过**提高分母中 k 的次数来达到减小 N 的目的**。

Table 2: The change of Error $f(r)$ with r

r	2	3	4	5	6	7	8	9	10
$\lceil f(r) \rceil$	2e12	1e6	8736	841	210	84	44	27	19

结合表 1 与表 2 可得，当 $r = 4$ 时，$N \geq 8736$ 时，运行时间是可以控制在 $100ms$ 以内的，$100ms$ 是一个理想的运行时间。因此我们接下来要尝试将分母中 k 的指数提升。

针对上述问题，可以对 Hamming 级数求和公式进行转换：

$$\psi(x) = \sum_{k=1}^{\infty} \frac{1}{k(k+x)} = \sum_{k=1}^{\infty} \frac{1}{x}(\frac{1}{k} - \frac{1}{k+x}) = \frac{1}{x}(\sum_{k=1}^{\infty} \frac{1}{k} - \sum_{k=1}^{\infty} \frac{1}{k+x}) \tag{2}$$

其中，对于 $\sum_{k=1}^{\infty}\frac{1}{k+x}$，我们可以对其进行展开：

$$\begin{aligned}\sum_{k=1}^{\infty}\frac{1}{k+x}&=\frac{1}{1+x}+\frac{1}{2+x}+\frac{1}{3+x}+...+\frac{1}{n+x},n\longrightarrow\infty\\&=(\frac{1}{x}+\frac{1}{1+x}+\frac{1}{2+x}+\frac{1}{3+x}+...+\frac{1}{n+x})-\frac{1}{x},n\longrightarrow\infty\\&=\sum_{k=1}^{\infty}\frac{1}{k-1+x}-\frac{1}{x}\end{aligned}$$

将上述结果代入公式 (2)，可以得到：

$$\begin{aligned}\psi(x)=\sum_{k=1}^{\infty}\frac{1}{k(k+x)}&=\frac{1}{x}(\sum_{k=1}^{\infty}\frac{1}{k}-\sum_{k=1}^{\infty}\frac{1}{k-1+x}+\frac{1}{x})\\&=\frac{1}{x}(\sum_{k=1}^{\infty}\frac{1}{k}-\frac{1}{k-1+x}+\frac{1}{x})\\&=\frac{1}{x}[(x-1)\sum_{k=1}^{\infty}\frac{1}{k(k+x-1)}+\frac{1}{x}]\\&=\frac{1}{x}[(x-1)\psi(x-1)+\frac{1}{x}]\\&=\frac{x-1}{x}\psi(x-1)+\frac{1}{x^2}\end{aligned}$$

经整理, 可得递推公式 (3):

$$\psi(x)=\frac{x-1}{x}\psi(x-1)+\frac{1}{x^2} \tag{3}$$

经上述递推公式 (3)，可以计算，可以得到 $\psi(1)=1$。该方法可以起到两个作用，规避掉计算 $\psi(0)$，同时可以降低算法的时间复杂度。

我们对 Hamming 级数求和公式再次进行转换：

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

我们令：

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

此时可计算 $t(2)$ 的值：

$$
\begin{aligned}
t(2) &= \sum_{k=1}^{\infty} \frac{1}{k(k+2)(k+1)} \\
&= \sum_{k=1}^{\infty} \frac{1}{2k} + \frac{1}{2(k+2)} - \frac{1}{k+1} \\
&= \frac{1}{2} \sum_{k=1}^{\infty} \frac{1}{k} + \frac{1}{k+2} - \frac{2}{k+1} \\
&= \frac{1}{2}(\frac{3}{2} + \sum_{k=3}^{\infty} \frac{2}{k} - \sum_{k=1}^{\infty} \frac{2}{k+1}) \\
&= \frac{1}{2}(\frac{3}{2} - 1) \\
&= \frac{1}{4}
\end{aligned}
$$

针对上述转换的式子继续进行变换：

$$
\begin{aligned}
\phi(x) &= (1-x) \sum_{k=1}^{\infty} \frac{1}{k(k+x)(k+1)} + 1 \\
&= (1-x)(\sum_{k=1}^{\infty} \frac{1}{k(k+x)(k+1)} - t(2) + t(2)) + 1 \\
&= (1-x)(\sum_{k=1}^{\infty} \frac{1}{k(k+x)(k+1)} - \sum_{k=1}^{\infty} \frac{1}{k(k+2)(k+1)} + \frac{1}{4}) + 1 \\
&= (1-x)(\sum_{k=1}^{\infty} \frac{1}{k(k+x)(k+1)} - \frac{1}{k(k+2)(k+1)} + \frac{1}{4}) + 1 \\
&= (1-x)(\sum_{k=1}^{\infty} \frac{2-x}{k(k+x)(k+1)(k+2)} + \frac{1}{4}) + 1 \\
&= (1-x)(2-x) \sum_{k=1}^{\infty} \frac{1}{k(k+x)(k+1)(k+2)} + \frac{1}{4}(1-x) + 1
\end{aligned}
$$

此时分母中 k 的幂次达到了 4 次方，N 的值可以设置为 8736 使得最终的误差在 $\varepsilon = 0.5e - 12$ 以内，此时的运行时间可以在 $100ms$ 内。我们构建下述伪代码进行计算：

Algorithm 2: Power4 Algorithm of Series Summation of Hamming

input : The maximum value of x and the value of N
output: The value $\psi(x)$ of series summation of Hamming for each x

for $x \leftarrow 0.000$ **to** 2.000 **do**
  $\psi(x)$=0;
  for $k \leftarrow 1$ **to** N **do**
    $\psi(x) \leftarrow \psi(x) + (1-x)(2-x)\frac{1}{k(k+x)(k+1)(k+2)}$;
    $k \leftarrow k+1$;
  end
  $\psi(x) \leftarrow \psi(x) + \frac{1}{4}(1-x) + 1$;
  $x \leftarrow x + 0.001$;
end

针对问题 (c)，有以下分析。

由 (b) 中分析一致，新算法的误差可以通过表 1 和表 2 可知，我们选了一个在 $100ms$ 内的 N 的取值范围，此时只要 $N \geq 8736$ 就**一定**可以在误差 $\varepsilon = 0.5e - 12$ 内。与精确值的比较（控制误差在 0.5e-12 以内）可以见下表：

Table 3: Error results and running time for different values of N for Algorithm Power4

N	1e3	2e3	3e3	4e3	5e3	6e3	7e3	8e3	8736	1e4
S	3.4e-7	4.2e-8	1.2e-8	4.7e-9	1.9e-9	8e-10	2.7e-10	0	0	0
Running time(ms)	8	7	22	15	22	28	40	48	49	62

结论：可以发现，新算法相比旧算法所用的时间大大减少，并且误差从 $8e3$ 开始，就为 0，满足题目要求。而旧算法虽然通过增加 N 来达到满足误差条件，但是所用的时间太大，本质就是 $O(MN)$ 中，N 过大。新旧算法对于空间的消耗均可看作 $O(1)$。

算法加速

由公式 3 可知，如果我们已知了 $0 \leq x < 1$ 中的所有值，当 $x \geq 1$ 时的结果都可以通过递推关系求出。因此我们考虑在范围 $0 \leq x < 1$ 内，采用算法 2 进行计算，而 $1 \leq x \leq 2$ 时采用递推公式 3 计算。

Table 4: Error results and running time for different values of N for Algorithm DP

N	1e3	8736	1e4	1e5
S	0.028	0.028	0.028	0.028
Running time(ms)	4	24	30	287

结论：由表 4 和表 3 可得，利用递推的算法能节省一半的时间，但是误差非常大。其原因在于给定的 x 是小数，而计算出的小数本身就不是精确值，所以当递推出新的值时，由于存在误差累积，所以新的值也存在误差。如果给定的 x 均为整数的话，那么该递推算法便可排上用场，在线性时间 $O(M)$ 内得出结果。