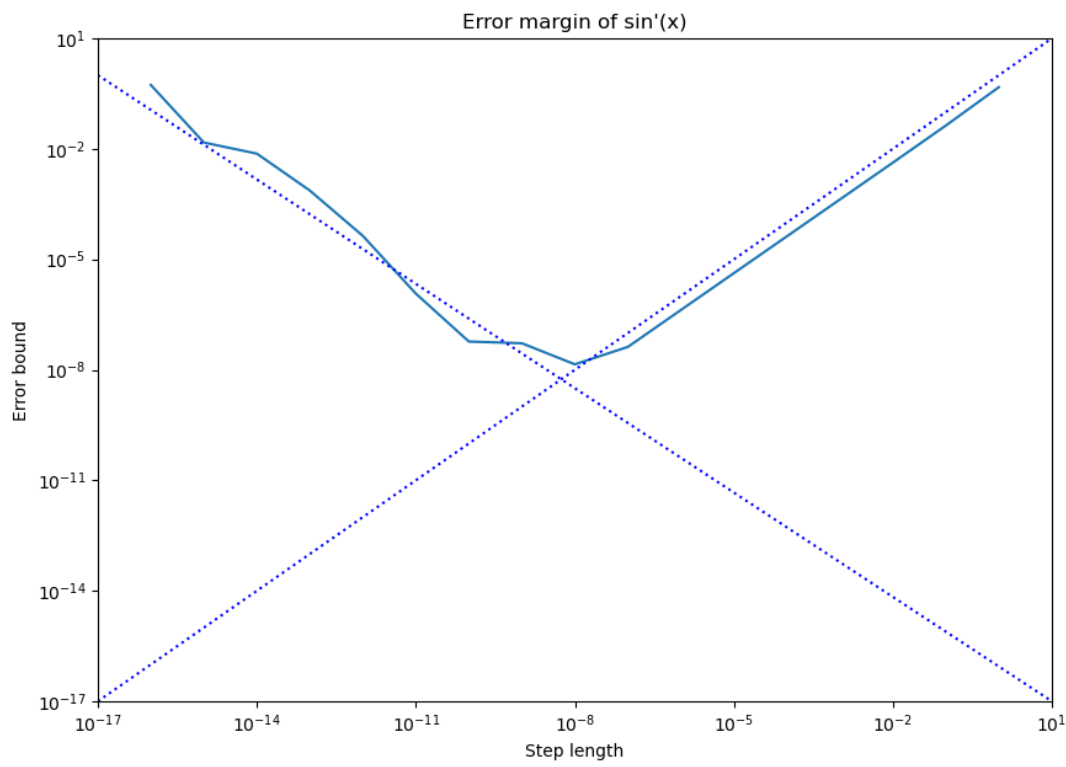


Experiment 1 Report

第一章上机题1

我们复现了课本例1.4，使用 $\sin' 1 = \lim_{h \rightarrow 0} \frac{\sin(1+h) - \sin 1}{h}$ 计算了 $\sin x$ 在 $x = 1$ 处的导数及绝对误差，并绘制了图1-2：



其中所有的计算都使用双精度浮点数进行。可以看出，舍入误差（由浮点数精度导致）近似地与步长成反比，这是因为在计算式中，分子部分的舍入误差基本是个定值，因而误差与分母（步长）成反比；而截断误差基本与步长成正比，这是因为由泰勒展开，上式的截断误差约为 $\frac{h^2}{2} \sin''(1)$ ，与步长二次方基本成正比。两种误差之和为总误差，在 10^{-9} 左右达到最小值。

第一章上机题3

我们在不同精度下计算了 $\sum \frac{1}{n}$ 。

使用单精度浮点数计算

单精度浮点数的机器精度 $\epsilon_{mach} = 2^{-24}$ ，为估计何时求和值不再变化，我们用 $\ln n + \gamma, \gamma \approx 0.57721566$ 估算上述级数和，则有当 $\frac{1/n}{\ln n + \gamma} \leq \frac{\epsilon_{mach}}{2}$ 时级数求和不再变化。利用计算软件求得 $n \geq 2\,209\,628$ 。

我们使用 `numpy.float32` 做单精度浮点数计算，逐个计算 $\frac{1}{n}$ 的值并加到答案上，直到当前答案与上一步的答案相等为止。程序显示，计算到 $n = 2\,097\,152$ 时结果不再发生变化（与理论分析差别不大），固定为 15.403683，共花费 973.95 毫秒。

使用双精度浮点数计算

我们使用 `numpy.float_` 做双精度浮点数计算，对前2 097 152项进行求和，得到结果为15.133306，共花费739.86毫秒。假如认为双精度浮点数的计算结果为真实值，那么绝对误差为0.2704，相对误差约1.787%。可以看出，单精度浮点数在本任务中的误差是不可忽略的。

假如像上一部分那样在双精度下求和直到结果不发生变化, 由于双精度浮点数的机器精度 $\epsilon_{mach} = 2^{-53}$, 故使用相同方式计算得到 $n \geq 5.22654 \times 10^{14}$ 。程序运行时间因此估算为 $\frac{739.86}{2\ 097\ 152} ms * 5.22654 \times 10^{14} \approx 2134$ 天, 可以认为是超过本问题计算限度的。

程序结果截图

```
(base) casorazitora@allaxomai programming_assignments % python ./Ex1/assignment_1.py
2097152 single-precision iterations in 0.97395 seconds, yielding 15.403683
100%|██████████████████████| 2097151/2097151 [00:00<00:00, 2916619.17it/s]
Double precision calculations yield 15.133306 in 0.73986 seconds
Result difference: -0.2703764904991992
(base) casorazitora@allaxomai programming_assignments %
```

