Numerical Instability Caused by Cancellation

Group 2 - Ferschl Martin, Laktaoui Mohammed, Reiter Roman, Zenkic Mirza

November 30, 2025

1 Task specification

We aim to evaluate the function

$$f(x) = \exp(x) - 1$$

on a computer (in double precision arithmetic) for very small values of |x|, i.e., $0 < |x| \ll 1$.

1. Experiment on the computer using double precision arithmetic (you may assume that the exp function is implemented correctly in double precision). Choose x = $x = 10^{-k}$, k = 1, 2, 3, 4, 5,... For comparison, you can obtain an "almost exact" value of the function f using the Taylor expansion of exp(x) about x = 0, for example, up to the 10th degree (this is a brute-force method). For comparison of the results, plot the relative error between the two evaluation methods on a logarithmic scale.

2. By comparing the two methods, you will see that the direct evaluation of f (x) in its given form is numerically unstable, i.e., it yields very inaccurate results for $|x| \to 0$. Can you explain this observation?

2 Method

Using the Taylor expansion,

$$\exp(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!} \quad \Rightarrow \quad \exp(x) - 1 = \sum_{n=1}^{\infty} \frac{x^n}{n!} \approx \sum_{n=1}^{N} \frac{x^n}{n!},$$

we adopt $N = 10$ terms as the reference (*brute-force*) value for small x. The relative error we report is

$$\text{rel. error} = \frac{|f_{\text{direct}}(x) - f_{\text{ref}}(x)|}{|f_{\text{ref}}(x)|}.$$

3 Implementation

Below is the Python code used to generate the numerical results and the plot. It computes the direct value $\exp(x) - 1$, the Taylor reference, prints a table, and saves the figure as `relative_error.png`.

```
import numpy as np
import matplotlib.pyplot as plt

def taylor_exp_minus_one(x, n_terms=10):
    x = np.array(x, dtype=np.float64)
    s = np.zeros_like(x)
    term = None
    for k in range(1, n_terms + 1):
        term = x if k == 1 else term * x / k
        s += term
    return s

# x = 10^-k, k = 1..15
k_vals = np.arange(1, 16)
x = 10.0 ** (-k_vals)

f_direct = np.exp(x) - 1.0
f_ref = taylor_exp_minus_one(x, n_terms=10)
rel_err = np.abs(f_direct - f_ref) / np.abs(f_ref)

print("  k        x          direct          taylor          rel. error")
for k, xv, fd, ft, err in zip(k_vals, x, f_direct, f_ref, rel_err):
    print(f"{k:3d}  {xv:8.1e}  {fd: .3e}  {ft: .3e}  {err: .3e}")

plt.figure()
plt.loglog(x, rel_err, "o-")
plt.gca().invert_xaxis()
plt.xlabel(r"$x = 10^{-k}$")
plt.ylabel(r"relative error")
plt.title(r"Relative error of $\exp(x)-1$ vs Taylor reference")
plt.grid(True, which="both", ls="--")
plt.tight_layout()
plt.savefig("relative_error.png", dpi=200)
plt.close()
```

4 Results

4.1 Numerical table

Table 1 shows a sample of the numerical results for representative k (Values are produced by the Python script).

Table 1: Direct evaluation vs. Taylor reference for $x = 10^{-k}$.

k	x	direct	Taylor ref.	rel. error
1	1.000×10^{-1}	1.052×10^{-1}	1.052×10^{-1}	7.917×10^{-16}
2	1.000×10^{-2}	1.005×10^{-2}	1.005×10^{-2}	1.087×10^{-14}
3	1.000×10^{-3}	1.001×10^{-3}	1.001×10^{-3}	4.291×10^{-14}
4	1.000×10^{-4}	1.000×10^{-4}	1.000×10^{-4}	4.327×10^{-13}
5	1.000×10^{-5}	1.000×10^{-5}	1.000×10^{-5}	9.702×10^{-12}
6	1.000×10^{-6}	1.000×10^{-6}	1.000×10^{-6}	3.798×10^{-11}
7	1.000×10^{-7}	1.000×10^{-7}	1.000×10^{-7}	5.663×10^{-10}
8	1.000×10^{-8}	1.000×10^{-8}	1.000×10^{-8}	1.108×10^{-8}
9	1.000×10^{-9}	1.000×10^{-9}	1.000×10^{-9}	8.224×10^{-8}
10	1.000×10^{-10}	1.000×10^{-10}	1.000×10^{-10}	8.269×10^{-8}
11	1.000×10^{-11}	1.000×10^{-11}	1.000×10^{-11}	8.274×10^{-8}
12	1.000×10^{-12}	1.000×10^{-12}	1.000×10^{-12}	8.890×10^{-5}
13	1.000×10^{-13}	9.992×10^{-14}	1.000×10^{-13}	7.993×10^{-4}
14	1.000×10^{-14}	9.992×10^{-15}	1.000×10^{-14}	7.993×10^{-4}
15	1.000×10^{-15}	1.110×10^{-15}	1.000×10^{-15}	1.102×10^{-1}

4.2 Error plot

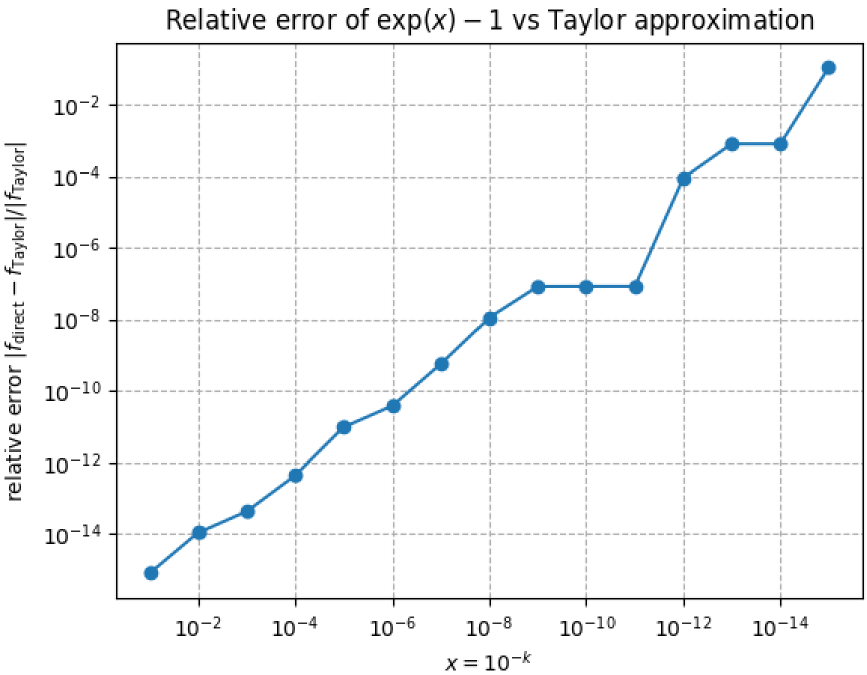

Figure 1: Relative error of the direct evaluation $\exp(x) - 1$ against the Taylor reference, for $x = 10^{-k}$. Both axes use logarithmic scaling (the x-axis is shown decreasing to the right).

5 Discussion

For very small x, $\exp(x) \approx 1 + x + \frac{x^2}{2} + \cdots$. The direct subtraction $\exp(x) - 1$ removes the leading 1, leaving a result dominated by rounding error (catastrophic cancellation). The absolute error is on the order of machine epsilon, while the true value is $\mathcal{O}(x)$, so the relative error grows like

$\varepsilon / |x|$ as $|x| \to 0$. For sufficiently small x, $\exp(x)$ may even round to 1, yielding a direct result of exactly 0.

6 Conclusion

The direct evaluation of $\exp(x) - 1$ is numerically unstable for small x due to cancellation. A stable alternative is to evaluate a series expansion near zero or to use a specialized routine such as `expm1(x)` provided by many standard libraries.