CSC336

Numerical Methods

Sinan Li

2025

CONTENTS

Part I

Part 1

CHAPTER

Introduction 1

1.1 Motivation

$$\text{Math Textbook} + \text{Laptop} + \text{Coding} \overset{?}{\Longrightarrow} \text{Compute Accurate Solution}$$

Consider the McLaurin series expansion of the function $f(x) = e^x$:

$$\begin{aligned} e^x &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \\ &= \sum_{n=0}^{\infty} \frac{x^n}{n!} \end{aligned}$$

The issue is that we cannot compute to infinity. We need to introduce partial sums

$$S_n = \sum_{i=0}^{n} \frac{x^i}{i!}$$

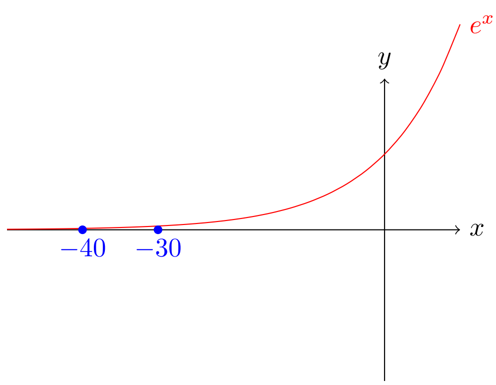
We could iterate over n until $|S_n - S_{n-1}| <$ tolerance.

x	0	1	10	20	40
Num Terms to "Converge"	2	13	42	69	104

We observe that the running time is dependent on the value of x. We need to find a better way to compute the sum – with more consistent running time.

Using the python program,

- When $x = -30$, convergence happened after 97 terms, to -6.0×10^{-5}.
- When $x = -40$, convergence happened after 124 terms, to approximately -5.9×10^{0}.



Clearly, we have inaccuracy when $x = -40$, as $0 < e^x < 1$ for all $x < 0$. The math textbooks' s techniques does not always provide good computational algorithms.

Course goal:

Show computational algorithms and discuss why they are good.

Example (e^x Better Algorithm)**.** A better algorithm is as follows

- Find k such that $r = \frac{x}{k}$ exactly with $\|r\| < 1$.
- Compute $e^r = e^{x/k}$ using the McLaurin series.
- Then, $e^x = (e^r)^k$.

◇

> **Remark** Error due to Catasrophic Cancellation
> When we subtract two numbers that are very close to each other, we lose precision.

1.2 Topics

- Computer Arithmetic and Computational Errors (Chap. 1)
  - Floating Point Arithmetic
  - Two Concepts
    - The conditioning of a math problem
    - the numerical stability of an algorithm
- Solving Systems of Linear Equations (Chap. 2)
  - Solve $Ax = b$ for x
- Solving Non-linear Equations (Chap. 5)
  Fine x s.t. $f(x) = 0$ or $g(x) = 0$ or $f(x) = g(x)$.
- Interpolation (Chap. 7)
  - Given the set of data
    $$\{(t_i, y_i)\}_{i=0}^n \qquad \text{or} \qquad \{(t_i, f(t_i))\}_{i=0}^n$$
    come up with a function $g(t)$ that approximates the data.

CHAPTER

Computer Arithmetic and Computational Errors 2

2.1 Numerical Stability

There is only finite space in computer. How would we store π, an irrational number? We can't. We can only store an approximation of π. How does introduction of approximations affect the accuracy of our computations?

Example. Suppose we want to compute the value for the sequence of integrals

$$y_n = \int_0^1 \frac{x^n}{x+5}\,dx$$

for $n = 0, 1, 2, \ldots, 8$, with 3 decimal digits of accuracy.

There are several properties that I can claim:

- $y_n > 0$ for all n, since the integrand $\frac{x^n}{x+5} > 0$ for all $x \in (0,1)$.
- $y_{n+1} < y_n$ for all n, since the integrand $\frac{x^{n+1}}{x+5} = x \cdot \frac{x^n}{x+5} < \frac{x^n}{x+5}$ for all $x \in (0,1)$.

There is not closed-form solution to this problem.

$$\begin{aligned}
x^n &= x^n \cdot \frac{x+5}{x+5} && \text{for } x \in (0,1)\\
x^n &= \frac{x^{n+1}}{x+5} + \frac{5x^n}{x+5}\\
\int_0^1 x^n\,dx &= \int_0^1 \frac{x^{n+1}}{x+5}\,dx + 5\int_0^1 \frac{x^n}{x+5}\,dx\\
\frac{1}{n+1}x^{n+1}\Big|_0^1 &= y_{n+1} + 5y_n\\
y_{n+1} &= \frac{1}{n+1} - 5y_n
\end{aligned}$$

Fortunately,

$$\begin{aligned}
y_0 &= \int_0^1 \frac{1}{x+5}\,dx && = \ln(x+5)\Big|_0^1\\
&= \ln 6 - \ln 5\\
&= \ln\frac{6}{5} \doteq 0.182
\end{aligned}$$

By the recurrence,

$$\begin{aligned}
y_1 &= \frac{1}{1} - 5y_0 \doteq 1 - 5(0.182) = 0.0900 \\
y_2 &= \frac{1}{2} - 5y_1 \doteq 0.5 - 5(0.0900) = 0.0500 \\
y_3 &= \frac{1}{3} - 5y_2 \doteq 0.333 - 5(0.0500) = 0.0830 \\
y_4 &= \frac{1}{4} - 5y_3 \doteq 0.25 - 5(0.0830) = -0.165
\end{aligned}$$

Clearly, something went wrong. We have a negative value for y_4, which is impossible. We also have $y_3 > y_2$. The problem is that we are using floating point arithmetic, which is not exact. We are losing precision in our calculations.

What if we leave y_0 as an unevaluated term?

$$\begin{aligned}
y_1 &= 1 - 5y_0 \\
y_2 &= \frac{1}{2} - 5y_1 \\
&= -\frac{9}{2} + 25y_0 \\
y_3 &= \frac{1}{3} - 5y_2 \\
&= \frac{137}{6} - 125y_0 \\
y_4 &= \frac{1}{4} - 5y_3 \\
&= -\frac{1367}{12} + 625y_0
\end{aligned}$$

We approximated $y_0 = \ln\frac{6}{5} \approx 0.182$. We know that the true value of $y_0 \in [0.1815, 0.1825]$. Another way to express y_0 is $y_0 = 0.182 + E$, where $|E| \leq 0.0005 = 5 \times 10^{-4}$ is the error in our approximation.

Substituting this into the formula for y_4, we get

$$\begin{aligned}
y_4 &= -\frac{1367}{12} + 625(0.182 + E) \\
&= -113.91\dot{6} + 113.75 + 625E \\
&= -0.1\dot{6} + 625E
\end{aligned}$$

where

$$625E \leq 625 \times 5 \times 10^{-4} = 0.3125$$

and

$$y_4 < y_0 \doteq 0.182$$

so our propagated error is greater than the quantity to compute. ♢

A lesson learned from the previous example is that the math textbook algorithms does not necessarily produce good computational algorithms. This algorithms for computing y_n is said to be an **numerically unstable algorithm**, since a small error was magnified by the algorithm. We want the algorithms to be **numerically stable**.

Definition 2.1.1 Numerically Unstable

An algorithm is said to be **numerically unstable** if the error in the output is not bounded by the error in the input.

Definition 2.1.2 Numerical Stability

An algorithm is said to be **numerically stable** if the error in the output is bounded by the error in the input.

Part II

Appendices

BIBLIOGRAPHY

[1] Danelnov, *Plantilla latex*, https://github.com/Danelnov/Plantilla-latex, 2022.

[2] Material Design. “The color system.” Section: Tools for Picking Colors. (2024), [Online]. Available: https://m2.material.io/design/color/the-color-system.html#tools-for-picking-colors.

INDEX