+4 for presentation

CS375 HW2

Ryan Scherbarth, University of New Mexico

September 2024

1. Binary and Floating Point numbers

   (a) Find the binary representation of the number -26.1

$$
\begin{aligned}
26_2 &= 11010_2 \\
0.1_{10} &= 0.1 * 2 = 0.2 \ (0) \\
&= 0.2 * 2 = 0.4 \ (0) \\
&= 0.4 * 2 = 0.8 \ (0) \\
&= 0.8 * 2 = 1.6 \ (1) \\
&= 0.6 * 2 = 1.2 \ (1) \\
&= 0.2 * 2 = 0.4 \ (0) \\
&= \dots \\
0.1_{10} &= 0.000110011\dots_2 \\
26.1_2 &=
\end{aligned}
$$

negative? repeat bar? -2

$$\boxed{11010.000110011\dots_2}$$

   (b) Find the double precision machine number which represents -26.1

Double: [Sign (1)] [Exponent (11)] [Mantissa (52)]

We can immediately set the sign bit to 1 to represent a negative number. Next, we need to shift the bits s.t. we are at one digit less than the most significant to the left of the decimal point.

$$11010.000110011 = 1.1010000110011 \text{ x}2^4$$

Now, we can determine the exponent:

$$
\begin{aligned}
\text{bias} &= 1023 \\
\text{exponent} &= 4 + 1023 = 1027 \\
1027_{10} &= 0100\ 0000\ 0011_2
\end{aligned}
$$

The mantissa ends off with a repeating sequence: 1100, so we can expand to fill the full mantissa.

$\boxed{\text{1 10000000011 1010000110011001100110011001100110011001100110011001}}$

round last digits -1

(c) Convert your answer back to base 10.

-6

$$\left|\frac{fl(-26.1) - -26.1}{-26.1}\right| = 0$$

this is not a true statement. you cannot do thi

$$\frac{\epsilon_m}{2} = 1.7763...$$

The relative error is less than half machine epsilon.

2. Cancellation, Precision and Loss of Precision

(a) Write naive code to evaluate $f(x) = \dfrac{1-(1-x)^3}{x}$

$$\boxed{fx = (1-(1-x)^3)/x;}$$

code? -3

(b) Evaluate the function for $x = 10^{-1}, 10^{-2}, \ldots, 10^{-14}$

$$
\begin{aligned}
f(x^{-1}) &= 2.709999999999999\\
f(x^{-2}) &= 2.970099999999998\\
f(x^{-3}) &= 2.997000999999999\\
f(x^{-4}) &= 2.999700010000161\\
f(x^{-5}) &= 2.999970000083785\\
f(x^{-6}) &= 2.999997000041610\\
f(x^{-7}) &= 2.999999698660716\\
f(x^{-8}) &= 2.999999981767587\\
f(x^{-9}) &= 2.999999915154206\\
f(x^{-10}) &= 3.000000248221113\\
f(x^{-11}) &= 3.000000248221113\\
f(x^{-12}) &= 2.999933634839635\\
f(x^{-13}) &= 3.000932835561798\\
f(x^{-14}) &= 2.997602166487923
\end{aligned}
$$

(c) Determine number of accurate digits in your answer.

Loss of Precision Theorem:

$$\text{if } 10^{-p} \leq 1 - \frac{y}{x} \leq 10^{-q}, \text{ and } p, q > 0$$
$$L(x, y) = x - y$$

$$\begin{aligned} x^{-1} &= 1 \\ x^{-2} &= 2 \\ x^{-3} &= 3 \\ x^{-4} &= 4 \\ x^{-5} &= 5 \\ x^{-6} &= 6 \\ x^{-7} &= 7 \\ x^{-8} &= 8 \\ x^{-9} &= 7 \\ x^{-10} &= 7 \\ x^{-11} &= 7 \\ x^{-12} &= 4 \\ x^{-13} &= 3 \\ x^{-14} &= 3 \end{aligned}$$

why is this almost bell-curve shaped?

-4

We see that overall the function is very inefficient, but becomes even more inefficient when using sufficiently large numbers.

(d) Rearrange the function to avoid catastrophic cancelling for small x

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

Matlab function"

$$fx = 3 - 3 * x + x^2;$$

this is not a coded function-3

(e) Repeat part b with the optimized function

$$f(x^{-1}) = 2.710000000000000$$
$$f(x^{-2}) = 2.970100000000000$$
$$f(x^{-3}) = 2.997001000000000$$
$$f(x^{-4}) = 2.999700010000000$$
$$f(x^{-5}) = 2.999970000083785$$
$$f(x^{-6}) = 2.999997000001000$$
$$f(x^{-7}) = 2.999999700000010$$
$$f(x^{-8}) = 2.999999970000000$$
$$f(x^{-9}) = 2.999999997000000$$
$$f(x^{-10}) = 2.999999999700000$$
$$f(x^{-11}) = 2.999999999970000$$
$$f(x^{-12}) = 2.999999999997000$$
$$f(x^{-13}) = 2.999999999999700$$
$$f(x^{-14}) = 2.999999999999970$$

(f) Table showing approximations, true values, adn rel error for each.

use scientific notation for error

x	Approximation $f(x)$	True Value $f(x)$	Relative Error
10^{-1}	2.709999999999999	2.710000000000000	0.000000000000001
10^{-2}	2.970099999999998	2.970100000000000	0.000000000000002
10^{-3}	2.997000999999999	2.997001000000000	0.000000000000001
10^{-4}	2.999700010000161	2.999700010000000	0.000000000000161
10^{-5}	2.999970000083785	2.999970000083785	0.000000000000000
10^{-6}	2.999997000041610	2.999997000001000	0.000000000040610
10^{-7}	2.999999698660716	2.999999700000010	0.000000001339294
10^{-8}	2.999999981767587	2.999999970000000	0.000000011767587
10^{-9}	2.999999915154206	2.999999997000000	0.000000081845794
10^{-10}	3.000000248221113	2.999999999700000	0.000000248521113
10^{-11}	3.000000248221113	2.999999999970000	0.000000248251113
10^{-12}	2.999933634839635	2.999999999997000	0.000066365157365
10^{-13}	3.000932835561798	2.999999999999700	0.000932835562098
10^{-14}	2.997602166487923	2.999999999999970	0.002397833512047

-4

Here we see that our result matches our previous prediction, as we increase the number of digits after the decimal point, we see the relative error increases.

does this make sense??

3. Taylor Series Errors

(a) How many terms in the approx. $E_n(x) = \frac{(x-c)^{n+1}}{(n+1)!} f^{n+1}(\epsilon)$ for error $< 2\text{x}10^{-8}$ for $x \exists [0, \frac{\pi}{2}]$

The derivatives of $\cos(x)$ alternate between cos, sin, and their negatives.
cos can be expanded;

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

The maximum value of $E_n(x) = |\frac{x^{n+1}}{(n+1)!} f^{n+1}(\epsilon)|$ is 1 when $f(x) = \cos(x)$, therefore;

$$E_n(x) \leq \frac{x^{n+1}}{(n+1)!} < 2\text{x}10^{-8} \text{ for } x \exists [0, \frac{\pi}{2}]$$
$$E_n(x) \leq \frac{x^{n+1}}{(n+1)!}$$

Given that the worst case will be $x = \frac{\pi}{2}$, we can find the lowest possible n that satisfies the threshold, which happens to be $n = 13$.

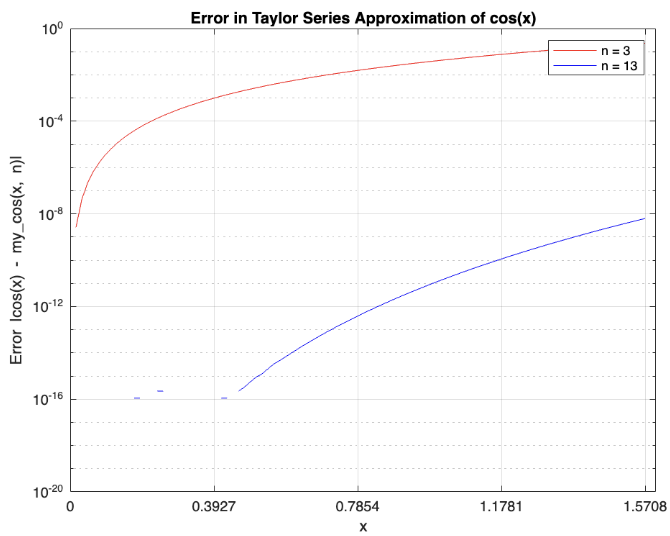
$$\boxed{E_n(13) = \frac{(\frac{\pi}{2})^{14}}{14!} \approx 6.39\text{x}10^{-9}}$$

(b) Function my-cos to approx. cos using a Taylor Series for order n.

```
function approx_cos = my_cos(x, n)
    approx_cos = 0;

    for k = 0:n
        term = ((-1)^k * x^(2*k)) / factorial(2*k);
        approx_cos = approx_cos + term;
    end
end
```

(c) Plot Results



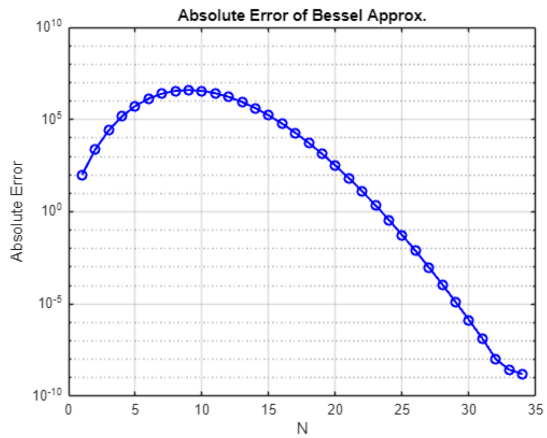
4. Taylor Series and Loss of Precision

(a) Matlab function to evaluate $\sum_{k=0}^{N}(-1)^k \frac{1}{(k!)^2}(\frac{x}{2})^{2k}$

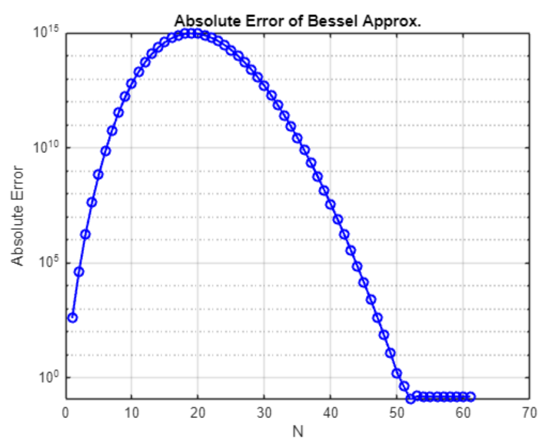
```
function result = ApproxBesselJ0(x, N)
    result = 0;

    for k = 0:N
        term = ((-1)^k / (factorial(k)^2)) * (x/2)^(2*k);
        result = result + term;
    end
end
```

(b) Compare from $N = 1 \ldots 20$ s.t. $\dfrac{1}{(M!)^2}(\dfrac{x}{2})^{2M} < 10^{-8}$, for $x = 20$



(c) Repeat with $x = 40$



-1

As we increase x, the $(M!)^2$ term increases much faster than $(\frac{x}{2})^{2M}$ term. As n gets large, the difference between these numbers increases. As we subtract an increasing small number (relative to the first term) from an increasingly large number, we increase our loss of precision.

this actually comes from subtracting very large