HW1 Question 3

Ben Wolfe

Concept Overview: Loss of Significance

- When machines perform subtraction on numbers that are very close in size, significance and/or accuracy can be lost:
  - $5.4322 - 5.4321 = .0001 = 1 * 10^{-4}$
  - Started with 5 significant digits, only 1 remains
- In some situations, this can be avoided

Our Problem

a) Come up with a different expression that computes **y = x –sin x** without suffering from loss of significance.

y(2^-31) = 1.68e-29, but machine will return 0.0e+0

Idea: Expand sin x in Taylor Series

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

Part a) Continued

$$y(x) = x - \sin x \approx x - (x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!}) = \boldsymbol{\frac{x^3}{3!} - \frac{x^5}{5!} + \frac{x^7}{7!}}$$

- We no longer have subtraction of similar sized numbers
- Also examine nested form:

$$y(x) \approx x^3(\frac{1}{3!} + x^2(-\frac{1}{5!} + \frac{1}{7!}x^2)$$

Part b): Results

	x	Test	Result	Nested Result
0	2^-1	2.057446e-02	2.057447e-02	2.057602e-02
1	2^-3	3.252666e-04	3.252666e-04	3.252673e-04
2	2^-5	5.086015e-06	5.086015e-06	5.086015e-06
3	2^-7	7.947262e-08	7.947262e-08	7.947262e-08
4	2^-9	1.241763e-09	1.241763e-09	1.241763e-09
5	2^-11	1.940255e-11	1.940255e-11	1.940255e-11
6	2^-13	3.031649e-13	3.031649e-13	3.031649e-13
7	2^-15	4.736950e-15	4.736952e-15	4.736952e-15
8	2^-17	7.401459e-17	7.401487e-17	7.401487e-17
9	2^-19	1.156412e-18	1.156482e-18	1.156482e-18
10	2^-21	1.805239e-20	1.807004e-20	1.807004e-20
11	2^-23	2.779327e-22	2.823443e-22	2.823443e-22
12	2^-25	3.308722e-24	4.411630e-24	4.411630e-24
13	2^-27	0.000000e+00	6.893172e-26	6.893172e-26
14	2^-29	0.000000e+00	1.077058e-27	1.077058e-27
15	2^-31	0.000000e+00	1.682903e-29	1.682903e-29
16	2^-33	0.000000e+00	2.629536e-31	2.629536e-31

Part c): Discussion

- This method does not have loss of significance because subtraction of close numbers does not occur as was the case for the original function
- Truncation error from Taylor Series
- Cannot always limit loss of significance

Appendix: Code

```
def loss_of_significance(x):
    return x**3/6 - x**5/120 + x**7/5040

vals = np.linspace(1,33,17)
x = np.float64(2.**-vals)

test = x - np.sin(x)
result = loss_of_significance(x)

# Use Horner's algorithm
a = np.array([0,0,0,1/6,0,-1/120,1/5040])
result_nested_eval = polynomial_evaluation(a,x)

pd.DataFrame({'x': [f'2^-{int(val)}' for val in vals],
              'Test': test, 'Result': result,
              'Nested Result': result_nested_eval})
```