Assignment 02: Errors
Assigned: 30th September 2019
Due: 18th October 2019 at 5 PM

Note: Please upload your solution as an ipynb or a PDF file to the Canvas page.

The purpose of this assignment is to develop your skills in computing absolute and relative errors. Use 64 bit floating point representation unless otherwise specified (this is the default in Python).

1. if $|x| < 1$ it is known that:

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \dots$$

For this series, compute for $x = -0.1$ the absolute and relative errors for 3, 5 and 7 terms in the series.

2. Evaluate e^{-5} using two approaches:

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \dots$$

and

$$e^{-x} = \frac{1}{e^x} = \frac{1}{1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots}$$

compare with the true value of 6.737947×10^{-3} for $x = 5$. Use 20 terms to evaluate each series and compute the absolute and relative errors.

3. The function $\cos(x)$ can be approximated using the Taylor series expansion as follows:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n}$$

   (a) Calculate the absolute and relative error associated with the Taylor series expansion of $\cos(\pi/4)$ using:
      i. one term in the expansion,
      ii. two terms in the expansion, and
      iii. three terms in the expansion.

   (b) Compute the number of terms required to have a solution that is accurate for 10 significant figures.

4. The infinite series

$$f(n) = \sum_{i=1}^{n} \frac{1}{i^4}$$

converges on a value of $f(n) = \pi^4/90$ as n approaches infinity. Write a program in single precision to calculate $f(n)$ for $n = 10,000$ by computing the sum from $i = 1$ to $10,000$. Using 16, 32 and 64 bit floating point representations, compute the absolute and relative errors. Explain your results.

Hint: use `np.float16(x)` to represent a variable `x` as a 16-bit floating point number. Use the same representation for both the variable used for the summation and inside the iteration loop.

5. Let $x^* = x \pm \varepsilon_x$ and $y^* = y \pm \varepsilon_y$ are the absolute errors associated with the variables. Compute the absolute error accumulation for the subtraction operation $x^* - y^*$. (Refer to error accumulation section in the handout).

6. Let $x^* = x(1 \pm \eta_x)$ and $y^* = y(1 \pm \eta_y)$ are the relative errors associated with the variables. What is the relative error of multiplication $(x^* y^*)$. Assume $\eta_x \eta_y$ is negligible as η_x and η_y are small enough.

7. The "divide and average" method is an old-time method for approximating the square root of any positive number x, and can be formulated as

$$g = \frac{g + x/g}{2}$$

Write a well-structured function to implement this algorithm that computes the square root of a positive number up to 5 significant figures. Use `math.sqrt(x)` to get the exact value.

Hint: Square root of a number x is y such that $y * y = x$. The recipe for deducing the square root of a number x:

(a) Start with a guess, g. What could be a good initial guess?

(b) If $g * g$ is *close enough* to x, stop and say g is the answer. This is where the 5 signficant figures condition should be applied.

(c) Otherwise make a *new guess* by averaging g and x/g as: $g = \frac{g+x/g}{2}$

(d) Using a new guess, *repeat* the process until *close enough*.

Check if your solution works for $x = 16$ and $x = \sqrt{2}$. Compute the relative error in both cases.

Example calculation:

g	$g * g$	x/g	$\frac{g+x/g}{2}$
3	9	16/3	4.16
4.17	14.36	3.837	4.0035
4.0035	16.0277	3.997	4.000002