Homework 2

Due 01/27/2020

1. In this problem, we will use Taylor approximation to approximate the integral

$$I = \int_1^2 \frac{e^x - 1}{x} dx.$$

Let us denote $f(x) = \frac{e^x-1}{x}$.

(a) Derive a formula for the n'th Taylor polynomial about $x_0 = 0$, called $p_n(x)$, of f. Use the summation symbol Σ to write $p_n(x)$.
Hint: use the Taylor approximation of the function e^x.

(b) Write the integral $I_n = \int_1^2 p_n(x)dx$ using Σ symbol and without integral signs.

(c) How large should n be so that I_n approximates I with an error less than $\epsilon = 10^{-5}$?

(d) With a value of n found in Part (c), write a Matlab code to compute I_n. Matlab has a built-in function called 'int' to compute approximately I. Try the following:

```
format long
f = @(x) (exp(x)-1)./x
integral(f,1,2)
```

Double check if I_n indeed approximates I with error less than 10^{-5}.

2. Let us consider the following toy model of the IEEE double precision floating-point format. This toy model makes it simpler to demonstrate how addition and multiplication of floating-point numbers work.

The sequence of 8 bits

$$\underbrace{c_0}_{\text{sign part}} \quad \underbrace{b_1 \quad b_2 \quad b_3 \quad b_4}_{\text{exponent part}} \quad \underbrace{a_1 \quad a_2 \quad a_3}_{\text{mantissa part}}$$

represents a number $x = \sigma \cdot \bar{x} \cdot 2^e$ where σ, $\bar{x}$, e are determined as follows:

$$\begin{aligned} \sigma &= \begin{cases} 1 & \text{if} \quad c_0 = 0, \\ -1 & \text{if} \quad c_0 = 1, \end{cases} \\ E &= (b_1 b_2 b_3 b_4)_2 \end{aligned}$$

- If $1 \leq E \leq 14$ then

$$\begin{aligned} e &= E - 7, \\ \bar{x} &= (1.a_1 a_2 a_3)_2 \end{aligned}$$

- If $E = 0$ then $e = -6$ and $\bar{x} = (0.a_1 a_2 a_3)_2$.
- If $E = 15$ then $x = \pm\infty$ (depending on the sign σ).

(a) Find the dynamic range and machine epsilon of this floating-point number format.

(b) What numbers are represented by the bit sequences 11001001, 00000000, 11111000 ?

3. There are only 256 different sequences of 8 bits. Thus, the sequence of 8 bits in Problem 2 cannot represent precisely every real number. It can represent *precisely* only 254 real numbers and $\pm\infty$. However, any real number can be represented *approximately* by a bit sequence. The principle is simple: *given a real number x, we look for the number y among those 256 numbers that is closest to x. Then x is represented by the bit sequence that represents y.*

   The method is as follows:

   - Write x is binary form. For example, $6.3 = (110.010011001\ldots)_2$.
   - Shift the binary point to the form $1.c_1c_2c_3\ldots$ by choosing an exponent $-6 \le e \le 7$. For example, $6.3 = (1.10010011001\ldots)_2 \times 2^2$.
   - Round the mantissa to 3 digits after the dot. For example, $6.3 \approx (1.101)_2 \times 2^2$.
   - Find the value of σ, $\bar{x}$, e. For example, these values in the case $x = 6.3$ are $\sigma = 1$, $\bar{x} = (1.101)_2$ and $e = 2$. The bit sequence that represents 6.3 is therefore 01001101.

   Note that in the second step, it may be impossible to choose e between -6 and 7. An example is when $e > 7$. In this case, the number is "*too big*" and is approximated by $\pm\infty$ (depending on the sign σ). Another example is when $e < -6$. In this case, one will shift the binary point one digit to the left to get the form $(0.1c_1c_2c_3\ldots)_2$. The new exponent is now $e+1$. If the new exponent is equal to -6 then one proceeds to Step 3 and 4. If the new exponent is still less than -6, the number x is "*too close to zero*" and thus is approximated by 0.

   (a) Represent the decimal numbers 1, 5.5, 12.9, 1000, 0.0001 in the floating-point format $x = \sigma \cdot \bar{x} \cdot 2^e$ and bit sequence described in Problem 2.

   (b) Find the smallest number larger than 5.5 that can be represented precisely by the floating-point format in Problem 2. The same question for 12.9 and 100.25.