APPM 4650 — Floating point arithmetic

Warm-up: What is numerical Analysis?

Floating point arithmetic
Questions:

- How many real numbers are there?
- Can the machine represent all of them?

Answer:

- Infinitely many.
- No. The machine cannot represent all real numbers.

The largest number that you can represent on the machine is 1.79×10^{308}. This is called the *overflow* number (denoted OFL). The smallest number that you can represent on the machine is 2.23×10^{-308}. This is called the *underflow* number (denoted UFL).
On the machine, numbers are represented by a truncated binary expansion of the form

$$\alpha = (a_0 2^0 + a_1 2^{-1} + a_2 2^{-2} + \cdots + a_p 2^{-(p-1)}) 2^e$$

where p is the precision and e is the exponent; i.e. the largest number that when 2^e is factored out of the series, the leading term is a constant.

Definition 0.1 *The representation of a real number on the machine is called its* floating point representation.
For an real number x that is representable with a float, $fl(x)$ denotes its floating point representation.

		p	Machine epsilson ϵ_m
Types of precision	Single	24	$\epsilon_m = 2^{-23} \approx 10^{-8}$
	Double	53	$\epsilon_m = 2^{-52} \approx 10^{-16}$

Definition 0.2 *The most accurately we can approximate a real number (in the relative sense) is* machine epsilon. *i.e let $\tilde{x} = fl(x)$, then $\frac{|x-\tilde{x}|}{|x|} \leq \epsilon_m$.*

Example: Write all the numbers that can be represented on the machine in the interval $[2, 4]$.
Solution: On the machine,

$$[2, 4] = \{2, 2 + 2^{-51}, 2 + 3(2^{-51}), \ldots, 4\}$$

Property: Let $\mathbb{F} = \{0, \text{all numbers that can be expressed via a float}\}$ denote the set of all floats (floating point numbers).

Definition 0.3 *Let $\tilde{x}$ be an approximation of x.*

- *The* absolute error *in the approximation is* $|x - \tilde{x}|$.
- *The* relative error *in the approximation is* $\frac{|x-\tilde{x}|}{|x|}$.

For all practical systems $0 < UFL < \epsilon_m < OFL$.

Exceptional values			
	Inf	infinity	?/0, or go over OFL
	NaN	not a number	0/0, 0*Inf or Inf/Inf

When to be careful with floating point arithmetic:

- Addition and subtraction can lead to a loss of some to all digits. This causes the most trouble.
- Multiplication of two precision digit numbers may not be representable; i.e. go into OFL.
- The quotient of two precision digit numbers may contain more than the number of digits available.
  Ex: nonterminating binary expansions such as 1/10.

The last two rarely cause trouble.
Let's look at some examples of floating point arithmetic causing trouble.
Example: Let $x = 1.92403 \times 10^2$ and $y = 6.3578 \times 10^{-1}$. Evaluate $x + y$ in 6 digit arithmetic.
Solution: $x + y = 1.9309 \times 10^2$ in 6 digit arithmetic. The last two digits of y do not contribute.
Example: Consider the infinite series $\sum_{n=1}^{\infty} 1/n$. Does this series converge? Does the series converge on the machine?
Solution: The series does not converge but the machine will think that it does since there exists an N such that $1/N < \epsilon_m$.
Note that if the user of the machine is not knowledgable about what the output should be, the finite answer may be used when it should not be. This type of error has led to many enigneering failures.
Example: Let ϵ be slightly smaller than ϵ_m . What is $fl(1 + \epsilon) - fl(1 - \epsilon)$? What is the real answer?
Solution: The float answer is 0 while the real answer is 2ϵ. So no significant digits are correct. This type of trouble is often encountered when solving PDEs in geometries that corners and edges.
Example: Consider the task of finding x such that $ax^2 + bx + c = 0$. We know that $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$. What happens if $-b \approx \sqrt{b^2 - 4ac}$? Can we fix the computation?
Solution: If $-b \approx \sqrt{b^2 - 4ac}$, we can get 0 even when the answer is far from it. This can be corrected by rescaling (i.e. mutliplying by the conjugate) and evaluating

$$x = \frac{2c}{-b \pm \sqrt{b^2 - 4ac}}$$

for the trouble root.