## Floating Point Representation and Arithmetic

1. Machine representation of floating point numbers.
   - If $x \in \Re$ then $x \approx fl(x) = (\text{mantissa})(\text{base})^{(\text{exponent})}$.
   - Example: IEEE 32 bit standard (see Section 2.2).
     - 8 bit exponent (base is two)
     - 23 bit normalized mantissa. Leading one is implicit.
     - "Machine epsilon" $\varepsilon = 2^{-23}$. This is a parameter that quantifies the relative precision of a particular floating point system. Note that $\varepsilon$ is the smallest positive number such that $fl(1+\varepsilon) > 1$.

2. Round-off error: occurs whenever a number cannot be exactly represented in the available number of bits.

3. Round-off error in representing a number:

$$\left|\frac{x - fl(x)}{x}\right| \leq \frac{1}{2}\varepsilon \qquad \text{or} \qquad fl(x) = x(1+\delta), \text{ for some } |\delta| \leq \frac{1}{2}\varepsilon.$$

   This says that the relative error in simply representing a real number in floating point can be as much as $\varepsilon$ (or, more precisely, $\frac{1}{2}\varepsilon$). Here are a couple of implications of this fact:

   - With IEEE 32 bit floating point ($\varepsilon = 2^{-23} \approx 10^{-7}$) you can never expect to have more than about 6 decimal digits of accuracy in any calculation.
   - When comparing two numbers $x$ and $y$, one should assume they are equal if

$$|x - y| < \varepsilon \max\{|x|, |y|\}.$$

     Why? Because two numbers that differ by less than this value could not be distinguished once they are represented in floating point.)

4. Propagation of round-off errors:
   - Multiplication—relatively safe:

$$\begin{aligned} fl(x) \cdot fl(y) &= x(1+\delta_x) \cdot y(1+\delta_y) \\ &= xy(1 + \delta_x + \delta_y + \delta_x\delta_y) \end{aligned}$$

   - Addition—problems if $x + y$ is small but $x$ and $y$ large:

$$\begin{aligned} fl(x) + fl(y) &= x(1+\delta_x) + y(1+\delta_y) \\ &= (x+y)(1 + \frac{x}{x+y}\delta_x + \frac{y}{x+y}\delta_y) \end{aligned}$$

   - An example of "catastrophic cancellation" error. Suppose we can store only 4 decimal digits in our machine.

| True | Computed | Signif. Digs. |
|---|---|---|
| 1.2376 | 1.238 | 4 |
| −1.2361 | −1.236 | 4 |
| 0.0015 | 0.002 | 1 |

     Note: $0.002 = 0.0015(1 + 0.33)$, i.e,. we have a 33% error in our answer.