CSC 165
floating-points
resources: chapter 7 of course notes

We use a “floating-point system given a fixed base $\beta$, a fixed number of digits $t$, and a range $[e_{\min}, e_{\max}]$ of exponents (integers), we can represent only numbers of the form:

$$\pm d_0.d_1 \ldots d_{t-1} \times \beta^e,$$

where the $d_i \in [0, \beta - 1]$ are called the digits (and the sequence of digits is called the MANTISSA), and $e \in [e_{\min}, e_{\max}]$ is the EXPONENT. (There’s also a sign, costing at least a bit).
If $\beta = 10$, $t = 3$, $e_{\text{min}} = -4$, and $e_{\text{max}} = +4$, then we represent

- $1/4$ as $+0.25 \times 10^0$ or $+2.50 \times 10^{-1}$.
- $1/3$ as $+3.33 \times 10^{-1}$

**Normalized mantissa**: we require that the first digit $d_0 \neq 0$ unless we are representing 0 itself.
Using this normalized floating-point system, if $\beta = 10$, $t = 3$, $e_{\text{min}} = -4$, and $e_{\text{max}} = +4$

- The smallest positive number is $+1.00 \times 10^{-4} = 0.0001$.
- The largest positive number is $+9.99 \times 10^4 = 99900$.

If $\beta = 2$, $t = 3$, $e_{\text{min}} = -2$, $e_{\text{max}} = +3$. Numbers (other than 0) have the form

$$\pm 1.d_1 d_2 \times 2^e.$$  

- Smallest positive number: $(1.00)_2 \times 2^{-2} = 1/4$.
- Largest positive number: $(1.11)_2 \times 2^3 = 14$. 

rounding

Consider $\beta = 10$, you cannot represent $1/3$ exactly, so we use $3.33 \times 10^{-1}$ when $t = 3$.

**Question** How do we represent $e = 2.718281828\ldots$?

- Round to nearest: $2.72 \times 10^0$.
- Truncate to zero: $2.71 \times 10^0$. 
overflow – underflow

There is no way to represent a number larger/smaller than the largest/smallest floating-point number.

- **Overflow** in a floating-point system occurs when we want to use an exponent larger than $e_{\text{max}}$ to express a number.

- **Underflow** in a floating-point system occurs when we want to use an exponent smaller than $e_{\text{min}}$ to express a number.
absolute, relative

- **Absolute Error**: Difference between the true value we're trying to represent and the value of its floating-point representation. For example, the absolute error in our representation of $e$ is $|2.71 - 2.718281828\ldots| = |0.008281828\ldots|$.

- **Relative Error**: For $x \neq 0$, the relative error between the approximate value $x'$ and the "real" value $x$ is
  $$\frac{|x - x'|}{|x|}$$

  For example, $|1.1 - 1|/|1.1| = 0.0909 \approx 9\%$. However, $|100.1 - 100|/|100| = 0.000999\ldots \approx 0.1\%$. 
Relative error in round-to-nearest

If you had infinitely many digits and base $\beta$, you could

*exactly* represent a number $d_0.d_1d_2\ldots d_{t-1}\ldots \times \beta^e$

But, if you’re limited to $t$ digits, you have to round up or down:

$$d_0.d_1d_2\ldots d_{t-1} \times \beta^e$$

or

$$d_0.d_1d_2\ldots (d_{t-1} + 1) \times \beta^e$$
Relative error in round-to-nearest – continued

The difference between these two numbers is simply a 1 in the position occupied by $d_{t-1}$, for a difference of $0.00 \ldots 1 \times \beta^e = \beta^{e-(t-1)}$. Since we round to nearest, our error is at most half this value:

$$\frac{\beta^{e-(t-1)}}{2}.$$

de the leading digit, $d_0$, is non-zero, so the smallest denominator (hence the largest bound) is when $d_0 = 1$, giving a relative error of:

$$\frac{|\beta^{e-(t-1)}/2|}{|1.0 \ldots 0 \times \beta^e|} = \frac{\beta^{1-t}}{2}$$

This matches our intuition that by increasing $t$ (the number of digits) we get more precision.

**Question:** $\beta = 2, t = 3$, what is a bound on the relative error of round-to-nearest?
Addition

Example take $\beta = 10$, $t = 3$, $e \in [-2, +2]$, and consider the sum $x + y$ where $x = 1.65 \times 10^2$ and $y = 2.71 \times 10^1$. First we need to get a common exponent (meaning usually one of the numbers must be denormalized) before doing the addition:

$$
\begin{array}{c}
1.65 \times 10^2 \\
+ 0.271 \times 10^2 \\
\hline
1.921 \times 10^2
\end{array}
$$

and the answer must be rounded (and, if necessary, normalized) to $1.92 \times 10^2$. Note that during normalization the exponent can change (for example, if a carry happened in the first digit or if the first digit becomes zero).
we don’t need the exponents to agree: we can find the product of the mantissas first, then add the exponents together. To compute $x \cdot y$ in this system:

\[
1.65 \times 10^2 \\
\times 2.71 \times 10^1 \\
\hline
4.4715 \times 10^3
\]

which is rounded (and, if necessary, normalized) to $4.47 \times 10^3$. In practice, we usually only need to “remember” one extra digit (beyond the $t$ digits) to figure out the rounding direction, so the exact answer is never really computed (in this example, the “…15” won’t be computed).
Accumulation of error

In our same small number system, set $x = 1.00 \times 10^2$ and $y = 1.00 \times 10^{-1}$

Now

$(((((((x+y)+y)+y)+y)+y)+y)+y) \neq x + (y+y+y+y+y+y+y+y+y+y+y+y+y+y+y)$
Catasrophic cancellation

Consider the same floating-point system to compute $b^2 - 4ac$ for $b = 3.34$, $a = 1.22$, and $c = 2.28$. The exact value is $0.0292 = 2.92 \times 10^{-2}$, and this exact value is representable in our floating-point system.

\[
\begin{align*}
  b^2 &= (3.34)^2 \\
  &= 11.1556 \approx 1.12 \times 10^1 \\
  4ac &= 4 \times 1.22 \times 2.28 \\
  &= 4.88 \times 2.28 \\
  &= 11.1264 \approx 1.11 \times 10^1 \\
  b^2 - 4ac &\approx 1.12 \times 10^1 - 1.11 \times 10^1 \\
  &= 0.01 \times 10^1 = 1.00 \times 10^{-1}
\end{align*}
\]
Catastrophic cancellation – continued

Compared to our exact answer of $2.92 \times 10^{-2}$, this has a relative error of

$$\frac{|0.0292 - 0.1|}{0.0292} = \frac{0.0708}{0.0292} = 2.424\ldots > 240\%.$$ 

Subtracting two floating-point numbers that are very close together leaves very few significant digits—a great deal of information is lost. Since the true value is very small, the round-off error becomes much more significant, and sometimes becomes much larger than the value being computed.
Catastrophic cancellation – continued

The expression $b^2 - 4ac$ crops up in the solution to the quadratic equation $ax^2 + bx + c = 0$. The general form of the solution for the two roots $x_1$ and $x_2$ is

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}.$$

We may not have to worry about the large relative error in $b^2 - 4ac$, since it may be small in absolute value compared to $-b$.

Here’s a case where computing $x_1$ gives a fairly acceptable value using floating-point operations:

$$x_1 = \frac{-3.34 + \sqrt{0.1}}{2 \times 1.22} = \frac{-3.34 + 0.316}{2.44} = -1.24$$

Compare this to the result if there were no error in the computation of $b^2 - 4ac$, which is:

$$x_1 = \frac{-3.34 + \sqrt{0.0292}}{2 \times 1.22} = \frac{-3.34 + 0.171}{2.44} = -1.30$$

for a relative error of less than 5%.
**stability**

**Definition:** a formula (or algorithm) is called **unstable** iff errors in the input values get magnified during the computation (i.e., iff the relative error in the final answer can be larger than the relative error in the input values).

\[
\frac{|f(x) - f(x')|/|f(x)|}{|x - x'|/|x|}.
\]

If you take the limit as \(x' \to x\) (and assume that \(f\) is differentiable, and that \(f(x) \neq 0\)), this is:

\[
\lim_{x' \to x} \frac{|f(x) - f(x')|/|f(x)|}{|x - x'|/|x|} = \frac{|x|}{|f(x)|} \lim_{x' \to x} \frac{|f(x) - f(x')|}{|x - x'|} = \frac{|xf'(x)|}{|f(x)|}.
\]
solutions

In general, there are two ways to deal with unstable formulas or algorithms:

- Increase the precision (the number of significant digits). This does not change the fact that the formula or algorithm is unstable, but can help minimize the magnitude of the errors, for some inputs.

- Use a different, more stable algorithm or formula to compute the result. When possible, this is preferred.

What is the condition number for $f(x) = x^5$ and $f(x) = \cos(x)$?