1 Calculus review

The notion of limits underlies all of calculus.

Definition 1.1. Let $f(x)$ be a function defined on a subset of the real numbers. Then we write

$$\lim_{x \to x_0} f(x) = L$$

if for any $\varepsilon > 0$, there exists $\delta > 0$ such that

$$0 < |x - x_0| < \delta \implies |f(x) - L| < \varepsilon.$$

We can also define limits of sequences.

Definition 1.2. Let $\{x_n\}_{n=1}^{\infty}$ be a sequence of real numbers. Then we write

$$\lim_{n \to \infty} x_n = x$$

if for any $\varepsilon > 0$, there exists $N > 0$ such that $|x - x_n| < \varepsilon$ whenever $n > N$.

Using this, we can define continuous functions.

Definition 1.3. A function $f(x)$ defined on a subset of the real numbers is continuous at x_0 if

$$\lim_{x \to x_0} f(x) = f(x_0).$$

The set of all continuous functions on a set of real numbers X is denoted by $C(X)$. In the special case when X is an interval (a, b) or $[a, b]$, we often write $C(a, b)$ or $C[a, b]$ instead. We also use limits to define derivatives.

Definition 1.4. Let $f(x)$ be a function defined on a subset of the real numbers. Then $f(x)$ is differentiable at x_0 if either of the following limits exist:

$$f'(x_0) = \lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

Many fundamental results in calculus are vital for understanding and applying numerical methods. The most important results are Taylor's theorem, whose usefulness stems from the ability to approximate a given function at a point in terms of its derivatives along with a known error term giving the validity of the approximation and the mean value theorem, which allows us to approximate an average of a function in terms of its derivative. To indicate that a function is n-times differentiable, rather than merely continuous, we use the notations $C^n(X), C^n(a, b), C^n[a, b]$.

Theorem 1.5 (Taylor's theorem). *Suppose that $f \in C^{n+1}[a, b]$, meaning that f is $(n+1)$-times differentiable on the interval $[a, b]$, and let $x_0 \in [a, b]$. Then for any $x \in [a, b]$, there is $\xi(x) \in [a, b]$ such that*

$$f(x) = f(x_0) + \cdots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)^{n+1}.$$

Theorem 1.6 (Mean Value theorem)**.** *If f is differentiable on an interval (a, b), then there is some $c \in (a, b)$ such that*

$$f'(x) = \frac{f(b) - f(a)}{b - a}.$$

We will see applications of these theorems over and over again in this class.

2 Floating Point Arithmetic

In many contexts, we assume that the numbers we work with are exact. However, computers cannot reasonably be expected to represent infinitely many numbers, so we need a system to represent numbers on a computer. Internally, computers represent everything using *bitstrings*: sequences consisting of 0s and 1s. If we want to represent natural numbers using bitstrings, we can use *binary*:

$$\begin{aligned} 0 &\to 0 \\ 1 &\to 1 \\ 10 &\to 2 \\ 11 &\to 3 \\ \ldots& \\ b_i b_{i-1} \ldots b_1 b_0 &\to b_i 2^i + b_{i-1} 2^{i-1} + \cdots + 2^1 b_1 + b_0 \end{aligned}$$

However, this doesn't help us at all to represent arbitrary numbers, such as π.

Definition 2.1 (IEEE 754)**.** Under the IEEE 754 standard, a 64 bit *floating point number* $sc_1 \ldots c_{11} f_1 \ldots f_{52}$ represents the number

$$(-1)^s 2^{c-1023}(1 + 0.f)$$

where $c = c_1 \ldots c_{11}$ is the *characteristic* and $0.f = 0.f_1 \ldots f_{52}$ is the *mantissa*. In addition to the "normal" numbers defined above, we also have "subnormal numbers" allowing for gradual underflow by defining special behavior if $C = 0$ using

$$(-1)^s 2^{-1022} 0.f.$$

Subnormal numbers help to avoid a lot of underflow issues that arise when working with small numbers.

Question 2.2. What is the largest 64-bit floating point number? The smallest? Roughly how many decimal digits can we get using these binary floating point representations?

Question 2.3. How would you find a floating point approximation for π?

Let $\varepsilon = 2^{-52}$. Different values of c yield numbers in different intervals:

$$\begin{aligned}
&\vdots\\
c = 1022 = 1111111110_2 &\to \frac{1}{2}, \frac{1}{2} + \frac{\varepsilon}{2}, \dots 1 - \frac{\varepsilon}{2} \in [\frac{1}{2}, 1)\\
c = 1023 = 1111111111_2 &\to 1, 1 + \varepsilon, \dots 2 - \varepsilon \in [1, 2)\\
c = 1024 = 10000000000_2 &\to 2, 2 + 2\varepsilon, \dots, 4 - 2\varepsilon \in [2, 4)\\
&\vdots
\end{aligned}$$

Question 2.4. How many intervals are represented in floating point arithmetic? Are they spread out evenly or do they bunch up? Why might this be useful?

Definition 2.5. If x is a real number, then its floating point representation $fl : \mathbb{R} \to FP$ is defined by setting $fl(x)$ to be the closest floating point number to x. (Ties can be resolved e.g. by rounding towards 0.) This process is called *rounding* and the resulting error

$$x - fl(x)$$

is called the *roundoff* error. It is often more meaningful to consider the *relative* rounding error

$$\left|\frac{x - fl(x)}{x}\right|.$$

Relative error is often more important than error itself because if x is large, an error of 1 may be negligible, whereas if x is small, then an error of 1 may be unacceptable.

It's sometimes impossible to deliver an exact floating point approximation of the correct answer to a problem due to fundamental limitations in certain approaches. Therefore it is of utmost importance to deliver a number along with an error estimate. Computations are meaningless without knowing bounds on the error.

Example 2.6.

$$fl(1 + \frac{\varepsilon}{2}) = 1$$

This example demonstrates the importance of relative error – it is meaningless to ask for an algorithm to deliver an approximation with error less than $\frac{\varepsilon}{2}$ for floating point numbers between 1 and 2 since there is no way to represent such numbers in floating point. This would be like your computer telling you "I know the answer, but I can't tell you."

$$fl(1 - \frac{\varepsilon}{2}) = 1 - \frac{\varepsilon}{2}$$

$$fl(1 - \frac{\varepsilon}{4}) = 1$$

$$fl(\frac{1}{2} - \frac{\varepsilon}{4}) = \frac{1}{2} - \frac{\varepsilon}{4}$$

The main idea of this class is that numerical analysis is all about delivering the correct answer, correctly rounded, as efficiently as possible. It will often be the case that the theoretical answer is simple, but impossible to store exactly on a computer. Because of this, there is often a large gap between theory and practice and the goal of numerical analysis is to bridge that gap.

3 Examples

Where do roundoff errors occur? Unfortunately: everywhere.

Example 3.1 (Quadratic formula)**.** Recall that given a quadratic polynomial $ax^2 + bx + c$ (with $a \neq 0$ and $b^2 - 4ac \geq 0$), we can find its zeroes using the *quadratic formula*

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

We will see in this example where floating point arithmetic can go wrong. Remember that floating point uses bitstrings to represent numbers. This complicates matters unnecessarily, so for these examples we will approximate floating point arithmetic using four-digit decimal arithmetic. (This means that any intermediate results can only be stored up to four decimal places.)

Consider the polynomial $x^2 + 62.10x + 1$. Using e.g. WolframAlpha, we find the zeroes

$$x = -0.01610723, -62.08390.$$

If we use four-digit decimal arithmetic with the quadratic formula, we find the following:

$$\frac{-62.10 \pm \sqrt{(62.10)^2 - 4(1)(1)}}{2(1)}$$

$$\frac{-62.10 \pm \sqrt{3856 - 4}}{2}$$
$$\frac{-62.10 \pm \sqrt{3852}}{2}$$
$$\frac{-62.10 \pm 62.06}{2}$$
$$\frac{-124.2}{2}$$
$$-.02, -62.10.$$

Comparing these to the actual answers from above, we find relative errors

$$\left|\frac{-.02 - -.01610723}{-.01610723}\right| \approx .24$$

$$\left|\frac{-62.10 - -62.08390}{-62.08390}\right| \approx .00026$$

In other words, the error in our understanding of the larger root is much (almost 1000 times!) smaller than our understanding of the smaller root; the relative error in the computation of the smaller root is larger than the root itself.

Question 3.2. Why did this happen?

What can we do about this?

1. Increase the accuracy of arithmetic operations (typically expensive, and may not always be available)
2. Reformulate the computation
3. Use a better method (next class)

Question 3.3. How can we fix this to get a better approximation of the other root?

Instead of directly using the quadratic formula, we can first rationalize the numerator:

$$\frac{-b \pm \sqrt{b^2-4ac}}{2a} \cdot \frac{-b \mp \sqrt{b^2-4ac}}{-b \mp \sqrt{b^2-4ac}} = \frac{b^2-(b^2-4ac)}{2a(-b \mp \sqrt{b^2-4ac})} = \frac{2c}{-b \mp \sqrt{b^2-4ac}}$$

and then recompute the roots:

$$\frac{2(1)}{-62.10 \mp \sqrt{62.10^2-4(1)(1)}}$$

$$\frac{2}{-62.10 \mp \sqrt{3856-4}}$$

$$\frac{2}{-62.10 \mp \sqrt{3852}}$$

$$\frac{2}{-62.10 \mp 62.06}$$

$$-.01610, -50.00$$

and we now find the relative errors:

$$\left|\frac{-.01610 - -.01610723}{-.01610723}\right| \approx .00045$$

$$\left|\frac{-50.00 - -62.08390}{-62.08390}\right| \approx .19$$

and the situations have reversed. The error in the smaller zero has improved significantly while the error in the larger zero has worsened. This example illustrates the difficulty of computing with small numbers. Moral of the story: try to avoid computing 0 unless necessary.

Another common situation where roundoff errors occur is in polynomial evaluation.

Example 3.4 (polynomial evaluation)**.** Evaluate the polynomial

$$f(x) = x^3 - 6.1x^2 + 3.2x + 1.5$$

at $x = 4.71$ using three digit decimal arithmetic. (This means that any intermediate results can only be stored up to three decimal places.) Using e.g. WolframAlpha, we evaluate

$$f(4.71) = -14.263899.$$

Using three digit decimal arithmetic, we compute the intermediate results:

$$x = 4.71$$

$$3.2x = 15.1$$
$$x^2 = 22.2$$
$$6.1x^2 = 135$$
$$x^3 = 105$$
$$f(4.71) = 105 - 153 + 15.1 + 1.5 = -13.4.$$

The relative error here is

$$\left| \frac{-13.4 - -14.263899}{-14.263899} \right| \approx .06.$$

Question 3.5. Why did we lose accuracy here?

As before, we can do better by reformulating the computation.

$$f(x) = x^3 - 6.1x^2 + 3.2x + 1.5 = ((x - 6.1)x + 3.2)x + 1.5$$

as we now compute

$$((4.71 - 6.1)4.71 + 3.2)4.71 + 1.5$$
$$((-1.39)(4.71) + 3.2)4.71 + 1.5$$
$$(-6.54 + 3.2)4.71 + 1.5$$
$$-3.34(4.71) + 1.5$$
$$-15.7 + 1.5$$
$$-14.2$$

for a relative error of

$$\left| \frac{-14.2 - -14.263899}{-14.263899} \right| \approx .0025.$$

Similarly to the previous example, this example illustrates the importance of doing as few floating point calculations as possible. By using nesting (you should always use nesting when evaluating polynomials), we can increase the accuracy of our results by decreasing the number of arithmetic operations required.

Question 3.6. This shows that floating point arithmetic is not associative. Why is that?