Math 164: Week 2B

Fall 2024

First algorithm for $f'(x)$: forward difference

We use Taylor's Theorem to find an approximation of $f'(x)$, namely the derivative of $f(x)$.

$$f(x_0+h) = f(x_0)+f'(x_0)h+\frac{f''(x_0)}{2!}h^2+\ldots+\frac{f^{(n)}(x_0)}{n!}h^n+\frac{f^{(n+1)}(s)}{(n+1)!}h^{n+1}$$

Keep only the 0, 1 order terms in h:

$$f(x_0+h) = f(x_0) + f'(x_0)h + O(h^2)$$

Rewrite it in the form

$$f'(x_0) = \frac{f(x_0+h)-f(x_0)}{h} + O(h)$$

We learn two important points from this derivation

- Forward difference (numerical algorithm):

$$f'_{fwd}(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} \tag{1}$$

- Truncation error:

$$|f'(x_0) - f'_{fwd}(x_0)| = O(h) \tag{2}$$

Python example of forward difference

Round-off error can become larger than truncation error

Total error is given by

$$\textit{total error} = |f'_{fwd}(x) - f'(x)| \tag{3}$$

It includes two types of errors: truncation error and round-off error.

• Truncation error: mathematical error since $f'(x)$ is approximated by $f'_{fwd}(x)$

• Round-off error: numerical error since $f'(x), f'_{fwd}(x)$ need to be evaluated numerically.

The smallest round-off error of a double precision number is

$$\epsilon = 2^{-53} \approx 10^{-16} \tag{4}$$

Include the round-off error in the forward difference formula:

$$\begin{aligned}
total\ error &= \left| \frac{f(x+h) - f(x) + 2\epsilon}{h} - f'(x) \right| && (5) \\
&= \left| \underbrace{\frac{f(x+h) - f(x)}{h} - f'(x)}_{trunc\ err} + \underbrace{\frac{2\epsilon}{h}}_{r.o.\ err} \right| && (6) \\
&\leq \left| \underbrace{\frac{f''(s)}{2} h}_{trunc\ err} \right| + \underbrace{\frac{2\epsilon}{h}}_{r.o.\ err} && (7) \\
&\leq \underbrace{\frac{h}{2}}_{trunc\ err} + \underbrace{\frac{2\epsilon}{h}}_{r.o.\ err} && (8)
\end{aligned}$$

Truncation error decreases as h decreases, but round-off error increases as h decreases. Check this in Python.

$$error = \frac{h}{2} + \frac{2\epsilon}{h} \tag{9}$$

Its minimum value is attained at

$$\frac{derror}{dh} = \frac{1}{2} - \frac{2\epsilon}{h^2} = 0 \tag{10}$$

$$h = (4\epsilon)^{1/2} \approx \left(4 \cdot 10^{-16}\right)^{1/3} \approx 2 \cdot 10^{-8}. \tag{11}$$

Second algorithm for $f'(x)$: Centered difference

How can we improve the precision of evaluating $f'(x)$?

- Decreasing h to small values has limitations because of the round-off error.
  - To avoid round-off error, we must switch to higher-precision floating system
- Alternative is to find algorithms that have higher precision

Use Taylor's approximation again, but in a different way

$$f(x_0 + h) = f(x_0 + h) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(s)h^3 \quad (12)$$

$$f(x_0 - h) = f(x_0 + h) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(t)h^3 \quad (13)$$

Subtract two equations, then

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + Ch^3 \quad (14)$$

Centered difference

Then we obtain the centered difference:

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} + O(h^2) \tag{15}$$

which has numerical precision of $O(h^2)$.

Recall that forward difference

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} + O(h) \tag{16}$$

has numerical precision of $O(h)$.

• This means that truncation error goes to zero as h^2. If we decrease h by a factor 0.1, then the truncation error decreases by a factor of 0.01 in the centered difference.

• The forward difference decreases by a factor of 0.1.

Python example of centered difference

Round-off error of centered difference

We can follow the same analysis performed on the forward difference to estimate the total error of centered difference.

The only change is that the truncation is error is $O(h^2)$ (not $O(h)$). Then the total error is

$$error = Dh^2 + \frac{2\epsilon}{h} \tag{17}$$

Its minimum value is attained at

$$\frac{derror}{dh} = 2Dh - \frac{2\epsilon}{h^2} = 0 \tag{18}$$

$$h = \left(\frac{\epsilon}{D}\right)^{1/3} \approx \left(\frac{10^{-16}}{0.2}\right)^{1/3} \approx 10^{-5}. \tag{19}$$

Condition number

Let's step back and consider different types of errors that emerge when finding numerical algorithms to a mathematical problem.

A general mathematical problem can be described as:

given input data x, find the output y defined by $y = F(x)$.

Examples of mathematical problems:

- $F(x) = f(x)$ (evaluate the value of a function)
- $F(x) = f'(x)$ (evaluate the derivative of a function)

Then, we looked for numerical algorithms $\tilde{F}(x)$ that yields approximation of the output: $\tilde{y} = \tilde{F}(x)$.
Examples of numerical algorithms:

- $\tilde{F}(x) = T_n(x)$ (Taylor polynomial)
- $\tilde{F}(x) = f'_{fwd}(x)$ (forward difference)

Types of errors

input	math problem	numerical algo	
x	$F(x)$	$\tilde{F}(x)$	**numerical**
$x+\delta x$	$F(x+\delta x)$	$\tilde{F}(x+\delta x)$	**error**
	condition	stability	

So far we considered the numerical error, i.e., how good is the numerical approximation:

$$\textit{numerical error} = |F(x) - \tilde{F}(x)|. \tag{20}$$

Condition number is about how sensitive the mathematical problem is to errors in the input

$$\textit{absolute condition number} = \lim_{\delta x \to 0} \frac{|F(x) - F(x+\delta x)|}{|\delta x|} \tag{21}$$

$$\textit{relative condition number} = \lim_{\delta x \to 0} \frac{|F(x) - F(x+\delta x)|/|F(x)|}{|\delta x|/|x|} \tag{22}$$

Condition number of evaluating the value of a function

The mathematical problem is to evaluating the function $f(x)$. So, $F(x) = f(x)$.

$$abs\ cond\ num = \lim_{\delta x \to 0} \frac{|f(x) - f(x + \delta x)|}{|\delta x|} = |f'(x)| \tag{23}$$

$$rel\ cond\ num = \lim_{\delta x \to 0} \frac{|f(x) - f(x + \delta x)|/|f(x)|}{|\delta x|/|x|} = |xf'(x)/f(x)| \tag{24}$$

- Well-conditioned problems have small absolute or relative condition number (< 1).
- Example of an ill-conditioned problem: find the roots r of f such that $r \neq 0, f'(r) \neq 0$.