Data Error and Computational Error.

compute $f(x)$ for a given problem.

- x true value of input
- $f(x)$ true value of output.
- $\hat{x}$ approximate input.
- $\hat{f}$ computed value of $f(x)$ (approximate of the function).

Total error : $\hat{f}(\hat{x}) - f(x)$

$$= \underbrace{\hat{f}(\hat{x}) - f(\hat{x})}_{\text{computational error}} + \underbrace{f(\hat{x}) - f(x)}_{\text{data error}}$$

- The algorithm that is chosen to compute $f(x)$ has no effect on the data error.
- We can only know the computational error and never the total error unless $f(x)$ is known.

Truncation and Rounding Errors

- truncation error : difference between true result and result produced by given algorithm using exact arithme

$$\underset{\substack{\uparrow \\ \text{true result}}}{\frac{1}{1-h}} = 1 + h + h^2 + h^3 + \cdots \approx \underbrace{1 + h + h^2}_{\text{algorithm}} + \underbrace{\mathcal{O}(h^3)}_{\substack{\text{truncation} \\ \text{error}}}$$

big-O of h^3

- Rounding error : difference between result produced by algorithm using exact arithmetic and result produced by the same algorithm using limited precision arithmetic

  example : $|1.085679 - 1.0857| = 0.000021$

- computational error = truncation error + rounding error

$$= \underbrace{\hat{f}(\hat{x}) - f(\hat{x})}$$

truncation error will come from the order of accuracy that which was used to compute $f(\hat{x})$ and $\hat{f}(\hat{x})$

rounding error comes from the precision used to represent $\hat{f}(\hat{x})$ or $\hat{x}$.

Absolute and Relative errors

- absolute error : approximate value − true value
- relative error : $\dfrac{\text{absolute value}}{\text{true value}}$

Example : Finite Difference Approximation

$$f'(x) \approx \frac{f(x+\varepsilon e) - f(x)}{\varepsilon}$$ is a first order

approximation. , where ε is very small, $\varepsilon \ll 1$ and e is a vector , $\vec{e} = [1, \dots, 1]$ $\|e\| = 1$

How do we derive this formula?

Finite-Difference Approach

From Taylor's Theorem, if f is twice differentiable, then f can be expanded at point x, such as

$$f(x+p) = f(x) + \nabla f(x)^T p + \tfrac{1}{2} p^T \nabla^2 f(x+tp) p \,, \quad t \in (0,1)$$

If L is a bound on the magnitude of the second derivative, $\nabla^2 f(x+tp)$, then

$$\|\nabla^2 f(x+tp)\| \leq L .$$

Thus, $\left\| f(x+p) - f(x) - \nabla f(x)^T p \right\| \leq \left(\tfrac{L}{2}\right) \|p\|^2 \leq \tfrac{1}{2} \|\nabla^2 f\| \cdot \|p\|^2$

Choose $p = \varepsilon e_i$, then

$$\| f(x+\varepsilon e_i) - f(x) - \nabla f(x)^T \varepsilon e_i \| \leq \left(\tfrac{L}{2}\right) \|\varepsilon e_i\|^2$$

Here ε is a very small scalar value, $\varepsilon \ll 1$ and e is a vector, $e = [1, \cdots, 1]$.

Then $\quad \| f(x+\varepsilon e_i) - f(x) - \nabla f(x)^T \varepsilon e_i \| \leq \left(\tfrac{L}{2}\right) \varepsilon^2 \cdot 1$

since $\|e\|^2 = 1$.

Next divide by ε,

$$\left\| \frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon} - \nabla f(x)^T e_i \right\| \leq \left(\tfrac{L}{2}\right) \varepsilon$$

$\nabla f(x)^T e_i$ is then the first derivative of $f(x)$, or $\frac{\partial f}{\partial x_i}$.

So,

$$\frac{\partial f}{\partial x_i} = \frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon} + \delta_e \quad , |\delta_e| \leq \left(\tfrac{L}{2}\right) \varepsilon .$$

This is simply called a forward difference formula. As $\varepsilon \rightarrow 0$, then

$$\frac{\partial f}{\partial x_i} = \frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon} + \mathcal{O}(\varepsilon) \quad , \text{ and}$$

$$\mathcal{O}(\varepsilon) = |\delta_e| \leq \left(\frac{L}{2}\right)\varepsilon \, , \quad \text{so} \quad \mathcal{O}(\varepsilon) = \|\delta_e\| \longrightarrow 0 \text{ as } \varepsilon \rightarrow 0.$$

And $\frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon}$ approximates the first derivative.

To improve the accuracy, a second order approximation can be derived, From the Taylor series expansion, if p is set to $+\varepsilon e_i$ and $-\varepsilon e_i$, then

$$f(x+\varepsilon e_i) = f(x) + \varepsilon \nabla f(x)^T + \frac{1}{2}\varepsilon^2 \nabla^2 f(x+t\varepsilon e_i) + \mathcal{O}(\varepsilon^3)$$

$$f(x-\varepsilon e_i) = f(x) - \varepsilon \nabla f(x)^T + \frac{1}{2}\varepsilon^2 \nabla^2 f(x - t\varepsilon e_i) + \mathcal{O}(\varepsilon^3)$$

substract the second from the first,

$$f(x+\varepsilon e_i) - f(x-\varepsilon e_i) = 2\varepsilon \nabla f(x)^T + \frac{1}{2}\varepsilon^2 \left(\nabla^2 f(x+t\varepsilon e_i) - \nabla^2 f(x-t\varepsilon e_i)\right) + \mathcal{O}(\varepsilon^3)$$

$$\frac{f(x+\varepsilon e_i) - f(x-\varepsilon e_i)}{2\varepsilon} = \nabla f(x) + \frac{1}{4}\varepsilon \left(\nabla^2 f(x+t\varepsilon e_i) - \nabla^2 f(x-t\varepsilon e_i)\right) + \mathcal{O}(\varepsilon^2)$$

as $\varepsilon \rightarrow 0$,

$$\frac{f(x+\varepsilon e_i) - f(x-\varepsilon e_i)}{2\varepsilon} = \frac{\partial f}{\partial x_i} + \frac{1}{4}\varepsilon\left(\frac{\partial^2 f}{\partial x_i^2} - \frac{\partial^2 f}{\partial x_i^2}\right) + \mathcal{O}(\varepsilon^2)$$

$$\frac{\partial f}{\partial x_i} = \frac{f(x+\varepsilon e_i) - f(x-\varepsilon e_i)}{2\varepsilon} + \mathcal{O}(\varepsilon^2)$$

when using a computer whose processor is based on IEEE floating-point arithmetic standard, the numerical value of $f(x)$ is bounded by u, where $u = 1.1 \times 10^{-16}$. 7/

Thus
$$\left| f(x)_{numerical} - f(x) \right| \leq u L_f$$
$$\left| f(x+\varepsilon e_i)_{numerical} - f(x+\varepsilon e_i) \right| \leq u L_f$$

where L_f is a bound on $|f(x)|$.

Therefore
$$\frac{\partial f}{\partial x_i}(x) = \frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon} + (L/2)\varepsilon + 2uL_f/\varepsilon.$$

Error is bounded by
$$\left(\frac{L}{2}\right)\varepsilon + \frac{2uL_f}{\varepsilon} = \delta_e$$

so
$$\delta_e \varepsilon^2 = \frac{4L_f u}{L}$$
$$\delta_e \varepsilon = \sqrt{u}$$
assuming $\frac{L_f}{L}$ is scaled constantly.

This gives us an estimate for ε.

Forward and Backward Error

If we want to compute $y = f(x)$, but then obtain an approximate value $\hat{y}$.

- The forward error : $\Delta y = \hat{y} - y$, where $\hat{y} = f(\hat{x})$.
- The backward error : $\Delta x = \hat{x} - x$

$$\begin{tikzcd}
x \arrow[r, "f"] \arrow[d, leftrightarrow, "\text{backward error}"'] & y \arrow[d, leftrightarrow, "\text{forward error.}"] \\
\hat{x} \arrow[r, "f"] & \hat{y}
\end{tikzcd}$$

Example 1

If $y = \sqrt{2}$, then $\hat{y} = 1.4$ is an approximation

Its absolute forward error , $|\Delta y| = |\hat{y} - y|$

$$= |1.4 - 1.414\ldots|$$

$$\approx 0.0142$$

relative forward error, $\dfrac{|\Delta y|}{y} \approx 1\%$

Since $\sqrt{1.96} = 1.4$, then

the absolute backward error $= |\Delta x| = |\hat{x} - x|$

$$= |1.96 - 2|$$

$$= 0.04$$

relative backward error, $\dfrac{|\Delta x|}{x} \approx 2\%$

Example 2

Approximate $f(x) = e^x$ with a simplar function, and evaluate its accuracy at $x = 1$.

We know that $f(x) = e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$

If we truncate the series as,

$$\hat{f}(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + O(x^4)$$

The forward error is $\left| \hat{f}(x) - f(x) \right|$

The backward error is $\left| \hat{x} = \log(\hat{f}(x)) - x \right|$

$$\left| \hat{f}(x) - f(x) \right| = \left| 2.666667 - 2.718282 \right| \approx + 0.05615$$

$$\left| \hat{x} - x \right| = \left| 0.980829 - 1 \right| = 0.019171.$$

Sensitivity and Conditioning

- Problem is sensitive or ill-conditioned, if relative change in solution can be much larger than that in input data or condition number $\gg 1$.
- Condition number :

$$\text{cond} : \frac{|\text{relative change is solution}|}{|\text{relative change in input data}|}$$

$$= \left| \frac{f(\hat{x}) - f(x)}{f(x)} \right| \Big/ \left| \frac{\hat{x} - x}{x} \right|$$

$$\text{cond} := \frac{\text{relative forward error}}{\text{relative backward error}}$$

Example 1: Given a function $f(x)$, evaluate the condition number if $\hat{x} = x + \Delta x$.

$$f(\hat{x}) = f(x + \Delta x)$$

$$\frac{f(\hat{x}) - f(x)}{f(x)} = \frac{f(x+\Delta x) - f(x)}{f(x)} \approx \frac{f'(x)\,\Delta x}{f(x)}$$

$$\text{condition number} \approx \frac{\left|\dfrac{f'(x)\,\Delta x}{f(x)}\right|}{\left|\dfrac{\Delta x}{x}\right|} = \left|\frac{x f'(x)}{f(x)}\right|$$

Example 2 : Evaluate the sensitivity of the tangent function, near $\pi/2$

$$\frac{\pi}{2} = 1.570796\ldots\ldots$$

$$f(x) = \tan(1.57079) \approx 1.58 \times 10^5$$

$$f(\hat{x}) = \tan(1.57078) \approx 6.124 \times 10^4$$

$$\text{condition number} = \frac{\left|\dfrac{6.124\times10^4 - 1.58\times10^5}{1.58\times10^5}\right|}{\left|\dfrac{1.57078 - 1.57079}{1.57079}\right|} = 9.6\times10^4$$

Stability

- Algorithm is stable if result produced is relatively insensitive to perturbations to the input, which also means that the condition number is very small.

Accuracy

- Difference between computed and true solution.

- Stability alone does not guarantee accurate results.

Floating Point Numbers

$$x = \pm .b_1 b_2 \dots\dots b_t \times \beta^e$$

- $b_1 b_2 \dots b_t$ is the mantissa or t = precision
- e is the exponent , $L \le e \le U$
- b_i are the base digits , $0 \le b_i \le \beta - 1$.
- β is the base.

example : if $x = 14.7$ $(\beta, t, L, U) = (10, 3, -9, 9)$

$$x = .147 \times 10^2$$

In computers we use IEEE Standards,

	β	t	L	U
IEEE (SP - single precision)	2	24	-126	127
IEEE (DP - double precision)	2	53	-1022	1023

IEEE SP translates to 8 significant digits in the decimal system and 16 in DP.

<u>Theorem</u> : Suppose we are given the set of floating point numbers $F(\beta, t, L, U)$ and that $x \in \mathbb{R}$ satisfies $\beta^L < |x| < \beta^U$. It follows that

$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\beta^{1-t}$$

→ $fl(x)$ is the floating-point approximation of a given real number x.

so $\frac{|fl(x) - x|}{|x|}$ is the relative error in representing real number x within range of floating-point system.

→ For IEEE SP $\frac{1}{2}\beta^{1-t} = \frac{1}{2}2^{1-24} = 5.96\text{e}-8$

IEEE DP $\frac{1}{2}\beta^{1-t} = \frac{1}{2}2^{1-53} = 1.11\text{e}-16$

→ Therefore the accuracy of the floating point system is given by $\frac{1}{2}\beta^{1-t}$ or also called <u>machine precision</u>.

Two Issues with Finite-precision Arithmetic

1) Rounding

- chop : round toward zero.
- round to nearest :

example :	2.442	2.4	2.4
	2.481	2.4	2.5
	2.591	2.5	2.6
		chop	round to nearest.

2) Cancellation

When two numbers that are almost identical except for the last few digits are used in subtraction, then the resulting number will have only a few digits of accuracy. If we use this new number in a calculation, then the results will only have a few digits of accuracy.

example : $a = 654738.2919$

$b = 654737.7921$

If we represent a as 0.6547383×10^6 and b as 0.6547378×10^6, then $a - b = 0.5$. However the true number is 0.4998.