

COMP ENG 3SK3 Winter 2011

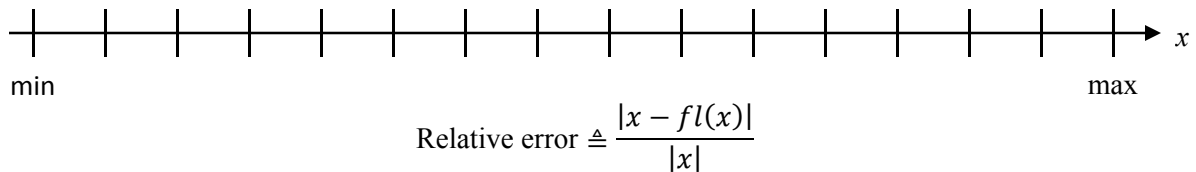
Assignment #1 (Solutions)

- a) Show why we cannot minimize the absolute error and relative error simultaneously in computer arithmetic, given the number of bits in machine word.

For a given number of bits, b , in machine word, $fl(x)$, only 2^b numbers can be represented exactly out of all numbers $\{x: -\infty < x < \infty, x \in \mathbb{R}\}$.

$$\text{Absolute error} \triangleq |x - fl(x)|$$

To minimize absolute error for all x , $fl(x)$ needs to always be a close approximation of x , regardless of the value of x . Therefore, $fl(x)$, the representation of x , should be equally spaced between a minimum and maximum.



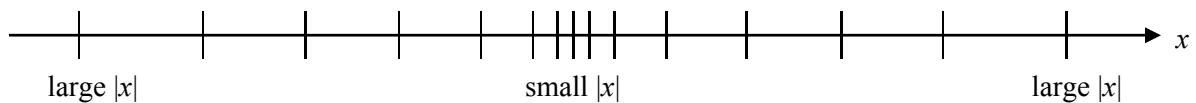
However, for equally spaced representations, relative error goes to ∞ when $|x|$ is very small.

$$\lim_{|x| \rightarrow 0} \frac{|x - fl(x)|}{|x|} = \infty$$

To minimize relative error for small $|x|$, $fl(x)$ has to be a closer approximation of x as $|x|$ gets smaller. $fl(x)$ should thus be spaced closer together for smaller $|x|$. But since the total number of exact machine representable numbers is finite and fixed at 2^b , the more clustered exactly representable numbers are near the origin, the more sparse exactly representable numbers $fl(x)$ are distributed as $|x|$ increases. For very large $|x|$, adjacent $fl(x)$'s do not have to be very close, because relative error approaches to 0 as $|x|$ approaches infinity,

$$\lim_{|x| \rightarrow \infty} \frac{|x - fl(x)|}{|x|} = 0$$

Therefore, minimizing relative errors requires uneven partition of the real line as below. But this increases the absolute error for large $|x|$ values.



Hence, absolute error and relative error cannot be minimized simultaneously by the same machine representation of numbers.

b) Prove that the two definitions of machine precision, as given in the bottom of page 18 and the top of page 19 in the course notes, are equivalent.

The real value, x , can be represented exactly with an infinite number of bits as

$$x = 1. \underbrace{XX \dots XX}_{t-1 \text{ bits}} \underbrace{XX \dots XX}_{\infty \text{ bits}} \times b^e$$

But, for chopping, the floating point representation, $fl(x) = 1. \underbrace{XX \dots XX}_{t-1 \text{ bits}} \times b^e$

$$\begin{aligned} |x - fl(x)| &< b^{-(t-1)} \times b^e = b^{1-t} \times b^e \\ |x| &\geq 1.00 \dots 00 \times b^e \\ \frac{|x - fl(x)|}{|x|} &< \frac{b^{1-t} \times b^e}{b^e} = b^{1-t} \end{aligned}$$

From the 1st definition, ϵ_{mach} = maximum relative error = $\max \left\{ \frac{|x - fl(x)|}{|x|} \right\} = b^{1-t}$ for chopping.

For rounding, maximum relative error is just half of that for chopping, $\epsilon_{mach} = \frac{b^{1-t}}{2}$.

$$\therefore \text{For binary, } b = 2, \quad \epsilon_{mach} = \begin{cases} 2^{1-t}, & \text{for chopping} \\ 2^{-t}, & \text{for rounding} \end{cases} \quad (1)$$

From the 2nd definition, $\begin{aligned} \text{if } \epsilon < \epsilon_{mach}, & \quad fl(1 + \epsilon) = fl(1) = 1 \\ \text{if } \epsilon \geq \epsilon_{mach}, & \quad fl(1 + \epsilon) > 1 \end{aligned}$

For ϵ_{mach} to be the smallest increment in the real value x to give an increment in the floating point representation $fl(x)$ after chopping, ϵ_{mach} needs to be the value of the least significant mantissa bit, which is $b^{-(t-1)} = b^{1-t}$.

For rounding, ϵ_{mach} is again half of that for chopping, $\epsilon_{mach} = \frac{b^{1-t}}{2}$.

$$\therefore \text{For binary, } b = 2, \quad \epsilon_{mach} = \begin{cases} 2^{1-t}, & \text{for chopping} \\ 2^{-t}, & \text{for rounding} \end{cases} \quad (2)$$

(2) from the 2nd definition is equivalent to (1) from the 1st definition.