University of Victoria Computer Science

CSC349A Numerical Analysis

Lecture 5

R. Little and G. Tzanetakis

University of Victoria

2025

Table of Contents I

Rounding and chopping

University of Victoria Computer Science

There are two methods of representing a real number p in floating-point: **rounding** and **chopping**.
For example let $b = 10$, $k = 4$ and $p = 2/3$. Then:

	p*	absolute error	relative error
chopping	$0.6666x10^0$	$0.0000666\ldots$	0.0001
rounding	$0.6667x10^0$	$0.0000333\ldots$	0.00005

Table: Rounding and chopping

Question:

University of Victoria Computer Science

What is the maximum possible relative error in the k-digit, base b, floating point representation p^* of a real number p with (a) **chopping**? (b) **rounding**?

Answer for (a) chopping

University of Victoria
Computer Science

Suppose $p \in [b^{t-1}, b^t)$. How many values are in this interval? What is the distance between adjacent values?

Answer for (a) chopping

Upper bound on absolute error of p^*? Upper bound on the relative error?

Unit round-off

University of Victoria Computer Science

Definition

The quantity b^{1-k} is called the **unit round-off** (or the **machine epsilon**). Note that it is independent of t and the magnitude of p. The number $k-1$ indicates approximately the number of significant base b digits in a floating-point approximation to a real number p.

Answer for (b) rounding

Upper bound on absolute error of p^*? Upper bound on the relative error?

Table of Contents I

Floating-point arithmetic

University of Victoria Computer Science

Floating-point arithmetic is a simulation of real arithmetic.

- We will use the notation fl to denote the floating-point representation of a real number x as $fl(x)$.
- Also, the floating-point representation of arithmetic operations such as:

$$fl(a+b), fl(a-b), fl(a \times b), fl(a/b)$$

  where a and b are floating-point numbers.
- The implementation of these floating-point operations (in either software or hardware) depends on several factors, and includes for example choices such as whether to use rounding or chopping and the number of significant digits used for floating-point addition and subtraction.

Idealized floating-point arithmetic

University of Victoria Computer Science

Definition

For simplicity, we will consider only **"idealized" floating-point arithmetic** which is defined as follows. Let $\bullet$ denote any of the basic arithmetic operations $+ - \times /$ and let x and y denote floating point numbers. $fl(x \bullet y)$ is obtained by performing **exact arithmetic** on x and y, and then **rounding or chopping** this result to k significant digits.

Note 1

Although no actual digital computers or calculators implement floating-point arithmetic that way (it's too expensive as it would require a very long accumulator for doing addition and subtraction), idealized floating-point arithmetic:

- Behaves very much like any actual implementation
- Is very simple to do in hand computations
- Has accuracy almost identical to that of any implementation

Note 2

If *fl* is applied to an arithmetic expression containing more than one arithmetic operation, then each of the arithmetic operations must be replaced by its corresponding floating-point operation.

Note 3

With idealized floating-point arithmetic, the maximum relative error in $fl(x \bullet y)$ is the same as the maximum relative error in converting a real number z to floating-point form. Thus, for a **single** floating-point operation $+ - \times /$, the **relative error is very small**: it is $< b^{1-k}$ with chopping, or $\frac{1}{2} b^{1-k}$ (with rounding). However, the relative error in a floating-point computation **might be large** if more than one floating-point operation is performed.

Example

Let $b = 10$, $k = 4$, $x = 0.1234 \times 10^0$, $y = -0.5508 \times 10^{-4}$, and $z = -0.1232 \times 10^0$.
(a) Calculate $x + y$, $fl(x + y)$, and the relative erorr between the two.

Example continued

(b) Now calculate $x + y + z$, $fl(x + y + z)$, and the relative erorr between the two.

Table of Contents I

Subtractive cancellation

University of Victoria Computer Science

- Loss of significant digits due to the subtraction of *nearly* equal floating-point numbers.
- In the case of $+$, $\times$ and $/$, there is little significant loss of precision.
- However, this is not the case with subtraction, where close to all the significant digits may be lost.
- Handout 4 presents four examples of this and some of the ways of avoiding it.

Example 1

Using $b = 10$, $k = 4$ idealized floating-point arithmetic evaluate $fl(\sqrt{x^2 + 1} - x)$ at $x = 65.43$ using rounding.

Example 1 continued

Can we solve this problem more accurately?

Example 2

Using $b = 10$, $k = 4$ idealized floating-point arithmetic evaluate $fl(x - \sin x)$ at $x = 0.01234$ using chopping.

Example 2 continued

How do we solve this one more accurately?