CSC349A Numerical Analysis

Lecture 3

Rich Little

University of Victoria

2018

Table of Contents I

Round-off errors

Round-off errors originate from two factors:

- finite representations of possibly infinitely long numbers
- finite range of values from a possibly infinite range

All dependent on the *word size* - maximum size of the string of bits used.

Number systems

- **Decimal:**
  - base-10
  - digits: 0,1,2,3,4,5,6,7,8,9
  - powers of 10 positional system
  - Ex: 86409
- **Binary:**
  - base-2
  - digits: 0,1
  - powers of 2 positional system
  - Ex: 101011

Computer representation

- Positive integers
  - What we can represent depends on the word size
  - Ex: 8-bits, then $43_{10} = 00101011_2$
  - What is the range? $2^8 = 256$ values from 0 to $2^8 - 1$
  - Ex: 16-bits, then $43_{10} = 0000000000101011_2$
  - What is the range? $2^{16} = 65,536$ values from 0 to $2^{16} - 1$

Computer representation

- Negative integers
  - Signed magnitude method - use leftmost bit for the sign
  - usually 0 for '+' and 1 for '-'
  - Ex: 8-bits, then $-43_{10} = 10101011_2$
  - What is the range? only 7 bits so 128 values with a lead 0 and 128 with a lead 1
  - But, two of them - 10000000 and 00000000 - represent the same number, 0
  - We usually let 10000000 = -128 giving the range -128 to 127

Real numbers

- How do we interpet real numbers in decimal?
- Ex: Consider 82.3801
- What about in binary?
- Ex: Consider 101.1101
- Going from decimal to binary with real numbers?
- Now, how do we represent them in a computer?

Table of Contents I

Floating-point number system

- A floating-point number system is a finite approximation to the (infinite) real/complex number system, of the form:

$$\pm m \times b^{e} \tag{1}$$

- The m is called the **mantissa**, b is the **base** and e is the **exponent**.

Normalized floating-point number system

- In normalized floating-point number systems real numbers are represented in the form:

$$\pm 0.d_1 d_2 d_3 \ldots d_k \times b^e \tag{2}$$

- Where the first digit of the *mantissa* should be non-zero, i.e. $1 \leq d_1 \leq b - 1$.
- The remaining digits can be zero and are also constrained by the base, i.e. $0 \leq d_i \leq b - 1$.
- Because $d_1 \neq 0$, k is the number of significant digits in the mantissa, and is called the *precision* of the floating point system.

Errors in floating-point representation

There are a number of inherent errors in this system, some more obvious than others.

- Large negative and positive numbers fall outside the finite range of the system (overflow).
- Because of normalization, very small (close to 0) negative and positive numbers fall outside the range (underflow).
- Only a finite number of values can be represented in the range (round-off error)
- The distance between two consecutive floating-point numbers increases as the numbers get larger

Rounding and chopping

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...	0.0001
rounding	$0.6667x10^0$	0.0000333...	0.00005

Table: Rounding and chopping

Question:

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**?

Unit round-off

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.