Rounding errors

Example

Show demo: "Waiting for 1".

Determine the double-precision machine representation for 0.1

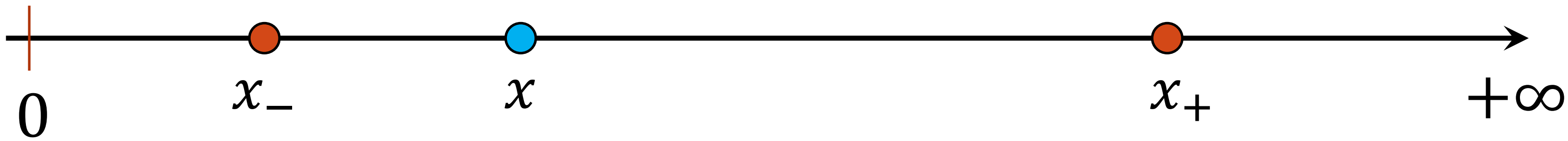
$0.1 = (0.000110011\,\overline{0011}\,\ldots\,)_2 = (1.100110011\,\ldots)_2 \times 2^{-4}$

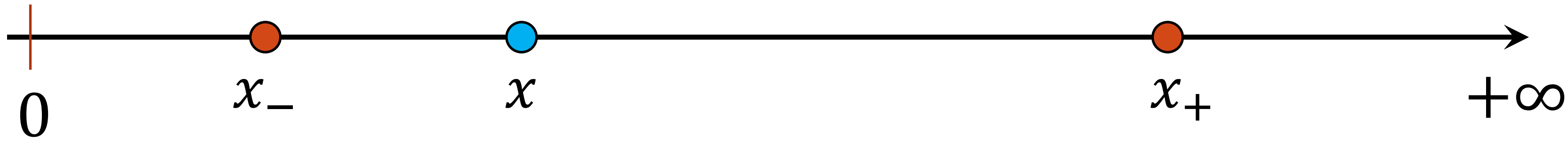
Machine floating point number

- Not all real numbers can be exactly represented as a machine floating-point number.
- Consider a real number in the normalized floating-point form:

$$x = \pm 1.b_1 b_2 b_3 \ldots b_n \ldots \times 2^m$$

- The real number x will be approximated by either x_- or x_+, the nearest two machine floating point numbers.





Exact number: $x = 1.b_1b_2b_3 \ldots b_n \ldots \times 2^m$

$$x_- = 1.b_1b_2b_3 \ldots b_n \times 2^m$$

$$x_+ = 1.b_1b_2b_3 \ldots b_n \times 2^m + \underbrace{0.000 \ldots 01}_{\epsilon_m} \times 2^m$$

Gap between x_+ and x_-: $|x_+ - x_-| = \epsilon_m \times 2^m$

Examples for single precision:

x_+ and x_- of the form $q \times 2^{-10}$

x_+ and x_- of the form $q \times 2^{4}$:

x_+ and x_- of the form $q \times 2^{20}$:

x_+ and x_- of the form $q \times 2^{60}$:

The interval between successive floating point numbers is not uniform: the interval is smaller as the magnitude of the numbers themselves is smaller, and it is bigger as the numbers get bigger.

Gap between two successive machine floating point numbers

A "toy" number system can be represented as $x = \pm 1.b_1 b_2 \times 2^m$ for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

$(1.00)_2 \times 2^0 = 1$
$(1.01)_2 \times 2^0 = 1.25$
$(1.10)_2 \times 2^0 = 1.5$
$(1.11)_2 \times 2^0 = 1.75$

$(1.00)_2 \times 2^1 = 2$
$(1.01)_2 \times 2^1 = 2.5$
$(1.10)_2 \times 2^1 = 3.0$
$(1.11)_2 \times 2^1 = 3.5$

$(1.00)_2 \times 2^2 = 4.0$
$(1.01)_2 \times 2^2 = 5.0$
$(1.10)_2 \times 2^2 = 6.0$
$(1.11)_2 \times 2^2 = 7.0$

$(1.00)_2 \times 2^3 = 8.0$
$(1.01)_2 \times 2^3 = 10.0$
$(1.10)_2 \times 2^3 = 12.0$
$(1.11)_2 \times 2^3 = 14.0$

$(1.00)_2 \times 2^4 = 16.0$
$(1.01)_2 \times 2^4 = 20.0$
$(1.10)_2 \times 2^4 = 24.0$
$(1.11)_2 \times 2^4 = 28.0$

$(1.00)_2 \times 2^{-1} = 0.5$
$(1.01)_2 \times 2^{-1} = 0.625$
$(1.10)_2 \times 2^{-1} = 0.75$
$(1.11)_2 \times 2^{-1} = 0.875$

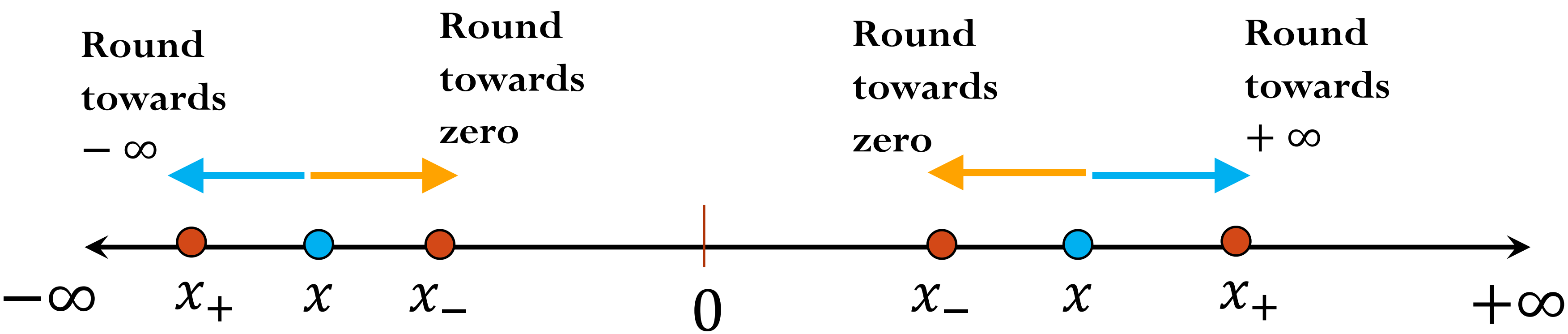
$(1.00)_2 \times 2^{-2} = 0.25$
$(1.01)_2 \times 2^{-2} = 0.3125$
$(1.10)_2 \times 2^{-2} = 0.375$
$(1.11)_2 \times 2^{-2} = 0.4375$

$(1.00)_2 \times 2^{-3} = 0.125$
$(1.01)_2 \times 2^{-3} = 0.15625$
$(1.10)_2 \times 2^{-3} = 0.1875$
$(1.11)_2 \times 2^{-3} = 0.21875$

$(1.00)_2 \times 2^{-4} = 0.0625$
$(1.01)_2 \times 2^{-4} = 0.078125$
$(1.10)_2 \times 2^{-4} = 0.09375$
$(1.11)_2 \times 2^{-4} = 0.109375$

Rounding

The process of replacing x by a nearby machine number is called rounding, and the error involved is called **roundoff error.**



Round by chopping:

	x is positive number	x is negative number
Round up (ceil)		
Round down (floor)		

Round to nearest:

Rounding (roundoff) errors

Consider rounding by chopping:

- **Absolute error:**

- **Relative error:**

Rounding (roundoff) errors

x_- $\quad x = 1.b_1b_2b_3 \ldots b_n \ldots \times 2^m$ $\quad x_+$

$$\frac{|\tilde{x} - x|}{|x|} \leq 2^{-23} \approx 1.2\times10^{-7}$$

Single precision: Floating-point math consistently introduces relative errors of about 10^{-7}. Hence, single precision gives you **about 7 (decimal) accurate digits.**

$$\frac{|\tilde{x} - x|}{|x|} \leq 2^{-52} \approx 2.2\times10^{-16}$$

Double precision: Floating-point math consistently introduces relative errors of about 10^{-16}. Hence, double precision gives you **about 16 (decimal) accurate digits.**

Iclicker question

Assume you are working with IEEE single-precision numbers. Find the smallest number a that satisfies

$$2^8 + a \neq 2^8$$

A) 2^{-1074}
B) 2^{-1022}
C) 2^{-52}
D) 2^{-15}
E) 2^{-8}

Demo

Floating point arithmetic

Consider a number system such that $x = \pm 1.b_1 b_2 b_3 \times 2^m$
for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

Rough algorithm for addition and subtraction:

1. Bring both numbers onto a common exponent
2. Do "grade-school" operation
3. Round result

- **Example 1: No rounding needed**

$$a = (1.101)_2 \times 2^1$$
$$b = (1.001)_2 \times 2^1$$

Floating point arithmetic

Consider a number system such that $x = \pm 1.b_1 b_2 b_3 \times 2^m$ for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

- **Example 2: Require rounding**

$$a = (1.101)_2 \times 2^0$$
$$b = (1.000)_2 \times 2^0$$

- **Example 3:**

$$a = (1.100)_2 \times 2^1$$
$$b = (1.100)_2 \times 2^{-1}$$

Mathematical properties of FP operations

Not necessarily associative:

For some x , y, z the result below is possible:

$$(x + y) + z \neq x + (y + z)$$

Not necessarily distributive:

For some x , y, z the result below is possible:

$$z\,(x + y) \neq z\,x + z\,y$$

Not necessarily cumulative:

Repeatedly adding a very small number to a large number may do nothing

Demo: FP-arithmetic

Floating point arithmetic

Consider a number system such that $x = \pm 1.b_1 b_2 b_3 b_4 \times 2^m$
for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

- **Example 4:**

$$a = (1.1011)_2 \times 2^1$$
$$b = (1.1010)_2 \times 2^1$$

Cancellation

Demo

$$a = 1.a_1a_2a_3a_4a_5a_6 \ldots a_n \ldots \times 2^{m1}$$
$$b = 1.b_1b_2b_3b_4b_5b_6 \ldots b_n \ldots \times 2^{m2}$$

Suppose $a \approx b$ and single precision (without loss of generality)

$$a = 1.a_1a_2a_3a_4a_5a_6 \ldots a_{20}a_{21}10a_{24}a_{25}a_{26}a_{27} \ldots \times 2^m$$

$$b = 1.a_1a_2a_3a_4a_5a_6 \ldots a_{20}a_{21}11b_{24}b_{25}b_{26}b_{27} \ldots \times 2^m$$

Example of cancellation:

Cancellation

$a = 1.a_1a_2a_3a_4a_5a_6 \ldots a_n \ldots \times 2^{m1}$
$b = 1.b_1b_2b_3b_4b_5b_6 \ldots b_n \ldots \times 2^{m2}$

For example, assume single precision and $m1 = m2 + 18$ (without loss of generality), i.e. $a \gg b$

$$fl(a) = 1.a_1a_2a_3a_4a_5a_6 \ldots a_{22}a_{23} \times 2^{m+18}$$

$$fl(b) = 1.b_1b_2b_3b_4b_5b_6 \ldots b_{22}b_{23} \times 2^{m}$$

$$\begin{array}{rl} & 1.a_1a_2a_3a_4a_5a_6 \ldots a_{22}a_{23} \times 2^{m+18} \\ + & 0.0000 \ldots 001b_1b_2b_3b_4b_5 \times 2^{m+18} \\ \hline \end{array}$$

In this example, the result $fl(a+b)$ only included 6 bits of precision from $fl(b)$. Lost precision!

Loss of Significance

How can we avoid this loss of significance? For example, consider the function $f(x) = \sqrt{x^2 + 1} - 1$

If we want to evaluate the function for values x near zero, there is a potential loss of significance in the subtraction.

Loss of Significance

Re-write the function as $f(x) = \frac{x^2}{\sqrt{x^2+1}-1}$ (no subtraction!)

Example:

If $x = 0.3721448693$ and $y = 0.3720214371$ what is the relative error in the computation of $(x - y)$ in a computer with five decimal digits of accuracy?