Lecture 13 - *Floating-Point Arithmetic*

OBJECTIVE:
Floating-point arithmetic is the hardware analogue of scientific notation. Before we can assess the accuracy of the algorithms of numerical linear algebra, we must examine this topic.

◇ LIMITATIONS OF DIGITAL REPRESENTATIONS

Digital computers must represent real numbers with a *finite number of bits*; i.e., only a finite set of numbers can be represented.

This leads to 2 limitations:

1. the numbers cannot be arbitrarily large or arbitrarily small

2. they cannot be arbitrarily close together (i.e., there must be <u>gaps</u> between numbers!)

The first limitation is often not of concern: IEEE double precision supports a largest number of 1.79×10^{308} and a smallest number of 2.23×10^{-308}.

In other words, *overflow* and *underflow* are not common problems.

However, the gaps between numbers can pose real problems if you're not careful!

In IEEE double precision, the interval $[1, 2]$ is represented by the discrete subset

$$1, 1 + 2^{-52}, 1 + 2 \times 2^{-52}, 1 + 3 \times 2^{-52}, \ldots, 2 \quad (1)$$

Similarly, the interval $[2, 4]$ is represented by the same set of numbers multiplied by 2

$$2, 2 + 2^{-51}, 2 + 2 \times 2^{-51}, 2 + 3 \times 2^{-51}, \ldots, 4$$

In general, the interval $[2^j, 2^{j+1}]$ is represented by 2^j times (1).

Note 1. *The gaps between numbers* in a relative sense *is never larger than* $2^{-52} \approx 2.22 \times 10^{-16}$.
This seems unimportant!
It is — but only if your algorithm is stable!

◇ FLOATING-POINT NUMBERS

In this situation, the "position" of the decimal point is stored as an exponent, separate from the digits.

This leads to gaps between numbers that are equal on a relative basis (but grow in absolute value as the numbers themselves grow).

This is in contrast to *fixed-point numbers*, where the gaps between numbers are fixed.

e.g., in a 3-digit fixed-point representation $0.abc$, the difference between adjacent numbers is always 0.001.
(*verify!*)

Consider a discrete subset $\mathbb{F}$ of the real numbers $\mathbb{R}$ to be our floating-point number system. The elements of $\mathbb{F}$ are the number 0 together with all numbers of the form

$$x = \pm \left(\frac{m}{\beta^t} \right) \beta^e$$

$\beta \geq 2$ is an integer called the *base* (or radix)
$\rightarrow$ usually $\beta = 2$.

$t \geq 1$ is an integer called the *precision*
($t = 53$ for IEEE double precision).

m is an integer satisfying $\beta^{t-1} \leq m \ \leq \beta^t - 1$.

e is called the *exponent*.

With these restrictions, the choices of m and e are unique (and thus so is the representation of every number in $\mathbb{F}$).

Note 2. $\left(\frac{m}{\beta^t}\right) \leq 1$ *and is called the fraction or* mantissa *of* x.

$\diamond$ MACHINE EPSILON

The resolution of $\mathbb{F}$ is typically defined as half the distance between 1 and the next larger floating-point number.

This number is known as *machine epsilon* and represents a measure of the worst case of the relative amount by which a given real number is rounded off when represented as a machine number (i.e., a number of $\mathbb{F}$)

$$\epsilon_{\text{machine}} = \frac{1}{2}\beta^{1-t}$$

Another way to view this is

$$\text{for all } x \in \mathbb{R}, \text{ there is an } x' \in \mathbb{F} \text{ such that}$$

$$|x - x'| \leq \epsilon_{\text{machine}}|x|$$

We can define a function

$$\text{fl} : \mathbb{R} \to \mathbb{F}$$

as a function that takes a real number and *rounds it off* to the nearest floating-point number.

So,

$$\text{for all } x \in \mathbb{R}, \text{ there is an } \epsilon \text{ satisfying } |\epsilon| \leq \epsilon_{\text{machine}}$$

$$\text{such that } \text{fl}(x) = x(1 + \epsilon)$$

i.e., *the difference between a real number and its closest floating-point number is always smaller than* $\epsilon_{\text{machine}}$ *in relative terms.*

◇ FLOATING-POINT ARITHMETIC

The classic arithmetic operations are $+, -, \times, \text{ and } /$.

These are of course operations on elements of $\mathbb{R}$.

On a computer, they have analogous operations on $\mathbb{F}$. We denote these by $\oplus, \ominus, \otimes, \text{ and } \oslash$.

These operations are most naturally defined as follows:

Let $x, y \in \mathbb{F}$.

Let $*$ stand for one of $+, -, \times,$ and $/$,
and let $\circledast$ be its floating-point analogue.

Then

$$x \circledast y = \mathrm{fl}(x * y)$$

This leads us to the "fundamental axiom of the floating-point arithmetic".

$$\text{for all } x, y \in \mathbb{F}, \text{ there is an } \epsilon \text{ satisfying } |\epsilon| \leq \epsilon_{\text{machine}}$$

$$\text{such that } x \circledast y = (x * y)(1 + \epsilon)$$

i.e., *every operation of floating-point arithmetic is exact up to a relative error of size at most* $\epsilon_{\text{machine}}$.

Note 3. *We are often interested in* $x, y \in \mathbb{R}$ *not* $x, y \in \mathbb{F}$*!*

In this case,

$$\begin{aligned} x \circledast y &= \mathrm{fl}(\mathrm{fl}(x) * \mathrm{fl}(y)) \\ &= \mathrm{fl}(x(1+\epsilon_1) * y(1+\epsilon_2)) \\ &= x * y(1+\epsilon_1+\epsilon_2)(1+\epsilon) \\ &= x * y(1+\epsilon_1+\epsilon_2+\epsilon) \qquad \text{(verify!)} \end{aligned}$$