# Numerical mathematics

## Introduction

Numerical mathematics

**Numerical mathematics** is devoted to methods that seek an approximate but sufficiently accurate solution of problems in various fields. A **simplified mathematical model** of the problem is used; its partial tasks consist of various mathematical problems.

The following mathematical problems are often involved:

1. solution of systems of linear equations,
2. solution of differential equations,
3. calculation of integrals,
4. evaluations of function values,
5. estimation of errors in calculations,
6. ...

Typically, a computer calculation is involved.

From the history

- Error in the Patriot missile defense system (February 25th, 1991)

$$(0.1)_{10} = (0.0001100110011001100110011001100110011...)_2$$

- Explosion of the Ariane 5 rocket (June 4th, 1996) conversion from a 64-bit floating point number to a 16-bit signed integer

- ...

This does not mean that approximation methods do not work. In the vast majority of cases they work well, but it is important to know how reliable they are.

## Origin of errors

Category of errors

We will use different **approximations** to design the algorithm. We will therefore make various kind of mistakes, which can be divided according to their origin:

1. errors in the **model**: the mathematical model to solve the problem is somehow simplified.

2. errors in the **data**: data often come from measurements that do not have absolute accuracy.

3. errors in the **algorithm**: we don't have to have an algorithm that finds the exact solution in a finite number of steps.

4. **rounding** errors: errors occur during the calculation itself (e.g., during arithmetic operations).

Apart from data errors, we will give examples of all other kinds of errors. We start with rounding errors, which are given by the fact that the algorithm need a computer to do the hard work.

# Computer arithmetics

## Representation with floating point

Representation with floating point

To store a number in computer we usually use the binary number system.

$$(6)_{10} = (110)_2 \qquad (0.1)_{10} = (0.0001100110011001100110011001100110011...)_2$$

For non-integers, one can use the **scientific notation**. In the binary base a number $x$ is represented as

$$x = \pm\, m \cdot 2^e.$$

$m$ - **mantissa/significand** having a fixed number of digits / fixed length; these digits are also called **significant digits**.

$e$ - **exponent** having a fixed number of digits / fixed length.

IEEE-754

A number $x$ is represented by its sign $s$ and by the numbers $e$ and $m$.
The standard `IEEE-754` defines the following lengths of $e$ and $m$ and their interpretation.

| precision | length of $m$ | $d$ = length of $e$ | $b$ |
|---|---|---|---|
| `binary32` / single precision | 23 | 8 | 127 |
| `binary64` / double precision | 52 | 11 | 1023 |
| `binary128` / quadruple precision | 112 | 15 | 16383 |

- if $e = 2^d - 1$ and $m \neq 0$, then $x = \texttt{NaN}$ (Not a Number)
- if $e = 2^d - 1$ and $m = 0$ and $s = 0$, then $x = +\texttt{Inf}$
- if $e = 2^d - 1$ and $m = 0$ and $s = 1$, then $x = -\texttt{Inf}$
- if $0 < e < 2^d - 1$, then $x = (-1)^s \cdot (1.m)_2 \cdot 2^{e-b}$ (so-called **normalized numbers**)
- if $e = 0$ and $m \neq 0$, then $x = (-1)^s \cdot (0.m)_2 \cdot 2^{-b+1}$ (so-called **subnormal/unnormalized numbers**)

- if $e = 0$ and $m = 0$ and $s = 0$, then $x = 0$
- if $e = 0$ and $m = 0$ and $s = 1$, then $x = -0$

---

Machine numbers (1/3)

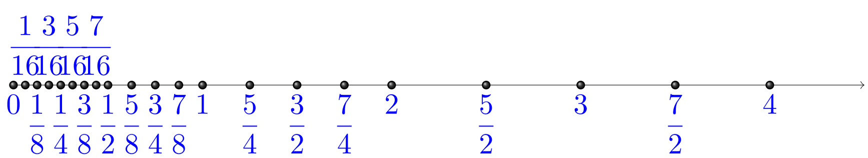
The numbers that can be represented as floating point numbers (with selected finite lengths of $m$ and $e$) are called **machine numbers**.

Example: take $m$ of length $2$ bits, $e$ of length $3$ bits, and $b = 3$.

We obtain the following set of numbers (we consider only positive elements)

$$\left\{0, \frac{1}{16}, \frac{1}{8}, \frac{3}{16}, \frac{1}{4}, \frac{5}{16}, \frac{3}{8}, \frac{7}{16}, \frac{1}{2}, \frac{5}{8}, \frac{3}{4}, \frac{7}{8}, 1, \frac{5}{4}, \frac{3}{2}, \frac{7}{4}, 2, \frac{5}{2}, 3, \frac{7}{2}, 4, 5, 6, 7, 8, 10, 12, 14\right\}$$

Subnormal numbers are in brown.




The set of all machine numbers with a given precision has little in common with the set of real numbers. It resembles more to a finite subset of integers.

---

Machine numbers (2/3)

Denote the set of machine numbers by $F$.

The set $F$ has the largest and the smallest positive elements as follows:

| precision | max. no. | min. pos. normalized | min. pos. subnormal |
|---|---|---|---|
| single | $(2 - 2^{-23}) \cdot 2^{127}$ $\approx 3.4 \cdot 10^{38}$ | $2^{-126}$ $\approx 1.2 \cdot 10^{-38}$ | $2^{-126-23} = 2^{-149}$ $\approx 1.4 \cdot 10^{-45}$ |
| double | $(2 - 2^{-52}) \cdot 2^{1023}$ $\approx 1.8 \cdot 10^{308}$ | $2^{-1022}$ $\approx 2.2 \cdot 10^{-308}$ | $2^{-1022-52} = 2^{-1074}$ $\approx 4.9 \cdot 10^{324}$ |

---

Machine numbers (3/3)

$F$ is characterized by the **machine epsilon** $\epsilon_F$, which is the difference between $1.0$ and the smallest number in $F$ larger than $1$.

For single precision we have $\epsilon_F = 2^{-23}$, for double $2^{-52}$.

**Proposition 1.** *The distance between any two neighboring normalized numbers in $F$ is at least $\frac{\epsilon_F}{2}$ and at most $\epsilon_F$.*

---

Representation of real numbers (1/3)

Let $fl : \mathbb{R} \to F$ be the mapping which assigns to any $x \in \mathbb{R}$ the closest machine number.

The "closest" is given by the method chosen: rounding ("ties to even"), chopping (rounding towards $0$),...

When trying to represent a number which is out of the representable range, an **overflow** or **underflow** is returned.

**Definition 2.** *Let a number $\alpha$ be an approximate value of a number $a$.*

- *The* ***absolute error*** *is the value $|\alpha - a|$.*
- *For $a \neq 0$, the* ***relative error*** *is $\dfrac{|\alpha - a|}{|a|}$.*

---

Representation of real numbers (2/3)

In single precision, suppose that a number $x \in \mathbb{R}$ lies in the normalized range, i.e.,

$$x = q \cdot 2^{\ell}, \quad \text{where } 1 \leq q < 2 \text{ and } -126 \leq \ell \leq 127.$$

What is the **error** due to the rounding or chopping when the closest machine number is chosen?

Let's round towards $0$, i.e., chop off bits which do not fit into the significand (for positive numbers).

If $\quad x = (1.b_1 b_2 \cdots b_{22} b_{23} b_{24} \cdots)_2 \cdot 2^{\ell} \qquad$ then $\quad fl(x) = (1.b_1 b_2 \cdots b_{23}) \cdot 2^{\ell}$.

The absolute error and the absolute errors are respectively:

$$|x - fl(x)| \le 2^{-23+\ell} \qquad \text{and} \qquad \frac{|x - fl(x)|}{|x|} \le \frac{2^{-23+\ell}}{q \cdot 2^{\ell}} \le 2^{-23}.$$

---

Representation of real numbers (3/3)

The threshold of relative error is called the **unit roundoff error** and is denoted by $\mathbf{u}$. Thus, in the single precision with chopping we have $\mathbf{u} = 2^{-23}$.

Attention, this number is sometimes called **machine epsilon**.

If we use mathematical rounding, we obtain $\mathbf{u} = 2^{-24}$.

**Proposition 3.** *Let $x \in \mathbb{R}$ be greater than the smallest normalized number of $F$ and smaller than the greatest normalized number of $F$. We have*

$$fl(x) = x(1+\delta), \quad \textit{where} \ \ |\delta| \le \mathbf{u},$$

## Arithmetic operations

---

Arithmetic operations - error

**Proposition 4.** *Let $x, y \in F$ and $\odot$ be the operation of addition, multiplication or division. If there is no overflow or underflow, then we have*

$$fl(x \odot y) = (x \odot y)(1+\delta), \quad \textit{where} \ \ |\delta| \le \mathbf{u},$$

In general: If we operate with more numbers, it is better to start with the smallest ones.

---

Arithmetic operations - a demonstration

Let $f : \mathbb{R}^2 \mapsto \mathbb{R}$ be a mapping given by

$$f(x,y) = 333.75y^6 + x^2\left(11x^2y^2 - y^6 - 121y^4 - 2\right) + 5.5y^8 + \frac{x}{2y}.$$

Let us evaluate $f(77617, 33096)$:

| | |
|---|---|
| `SageMath` (precision 23 bits) | $1.17260$ |
| `SageMath` (precision 24 bits) | $-6.33825 \cdot 10^{-29}$ |
| `SageMath` (precision 53 bits) | $-1.18059162071741 \cdot 10^{21}$ |
| `SageMath` (precision 54 bits) | $1.18059162071741 \cdot 10^{21}$ |
| `SageMath` (precision 100 bits) | $1.1726039400531786318588349045$ |
| `SageMath` (precision 121 bits) | $1.172603940053178631858834904520183 71$ |
| `SageMath` (precision 122 bits) | $-0.827396059946821368141165095479816292$ |

The exact solution is $-\dfrac{54767}{66192} \approx -0.827396$.

[S. M. Rump: *Algorithms for verified inclusions - theory and practice*, ..., 1988]

---

Loss of significant digits (1/3)

Errors while doing arithmetical operations can accumulate.

Big problems can be caused by the so-called **cancellation**.

Let us illustrate this on an example. Imagine that our computer calculates in basis $10$ and uses $10$ significant digits.

We want to evaluate $x - \sin(x)$ for $x = \dfrac{1}{15}$.

$$\begin{aligned} x &\leftarrow 6.6666\ \ 66667 \quad \cdot 10^{-2} \\ \sin(x) &\leftarrow 6.6617\ \ 29492 \quad \cdot 10^{-2} \\ x - \sin(x) &\leftarrow 0.0049\ \ 37175 \quad \cdot 10^{-2} \\ x - \sin(x) &\leftarrow 4.9371\ \ 75000 \quad \cdot 10^{-5} \end{aligned}$$

The last $3$ zeros are not *correct* significant digits.

Let us calculate the relative error.

---

Loss of significant digits (2/3)

$$\frac{\left|\left(\frac{1}{15} - \sin\left(\frac{1}{15}\right)\right) - fl\left(fl\left(\frac{1}{15}\right) - \sin\left(fl\left(\frac{1}{15}\right)\right)\right)\right|}{\left|\frac{1}{15} - \sin\left(\frac{1}{15}\right)\right|} \approx 1.4 \cdot 10^{-7}.$$

That is a lot in comparison to

$$\frac{|x - fl(x)|}{|x|} \le 5 \cdot 10^{-10}.$$

**Proposition 5.** *Let $x$ and $y$ be normalized machine numbers and $x > y > 0$.*

*If $2^{-p} \le 1 - \frac{y}{x} \le 2^{-q}$ for some positive integers $p$ and $q$, then **at most** $p$ and **at least** $q$ significant binary bits are lost when performing the operation $x - y$.*

---

Loss of significant digits (3/3)

Cancellation can be avoided by using the following techniques:

- rationalizing the problem, i.e., using rational numbers and avoiding the subtraction in floating points arithmetics,
- using series expansions (such as Taylor series),
- using other identities,...

## Errors - conclusion

---

Errors - conclusion

Origins of errors:

- rounding errors and their accumulation,
- cancellation.

The errors on the inputs may also play an important role. Those errors are given by the origin of the input which may be the output of another calculation or a measurement.

A few final notes:

- increased precision may not lead to a more precise result,
- cancellation can be useful - it may cancel rounding errors,
- few operations with small numbers do not imply a small error.

One of the problems of machine numbers (`IEEE-754`) is in the ignorance of the created error.

There are some alternatives:

- Exact arithmetics: $\mathbb{Z}$, $\mathbb{Q}$ or $GF(p)$ (it is not always possible or suitable).
- Interval arithmetics (we work with intervals instead of points). (`IEEE 1788-2015`).
- Unum.