Video 1: Rounding errors

A number system can be represented as $x = \pm 1.b_1b_2b_3b_4 \times 2^m$ for $m \in [-6,6]$ and $b_i \in \{0,1\}$.

Let's say you want to represent the decimal number 19.625 using the binary number system above. Can you represent this number exactly?

$(19.625)_{10} = (10011.101)_2 = (1.0011101)_2 \times 2^4$

$1.0011 \times 2^4 = 19$

$1.0100 \times 2^4 = 20$

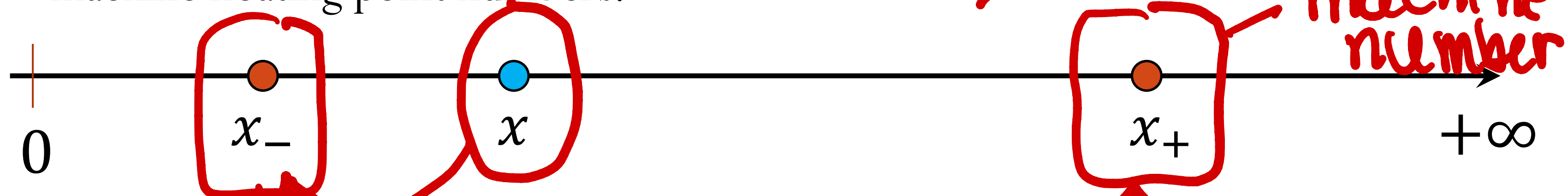
Machine floating point number

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

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

$b_{n+1} b_{n+2} \ldots$

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

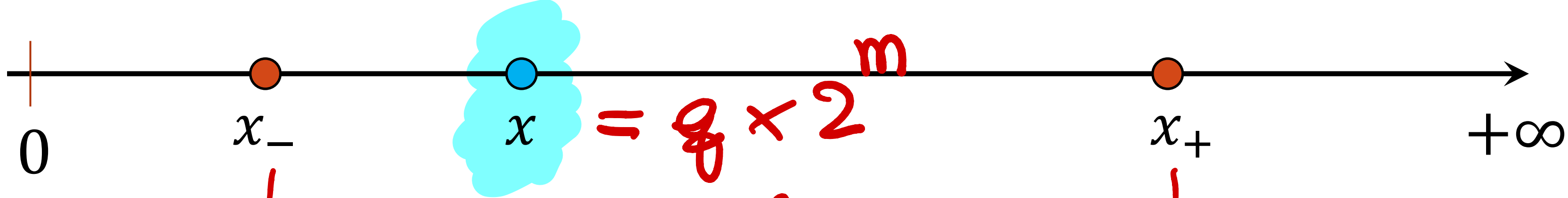


$X_- = 1.b_1 b_2 b_3 \ldots b_n \times 2^m$

last bit

$X_+ = \underbrace{X_-} + \underbrace{0.0000\ldots 01}_{2^{-n}} \times 2^m$

$2^{-n} \times 2^m = \epsilon_m \times 2^m$



$x = q \times 2^m$

$$x_+ = x_- + \epsilon_m \times 2^m$$

$$|(x_+) - (x_-)| = \epsilon_m \times 2^m$$

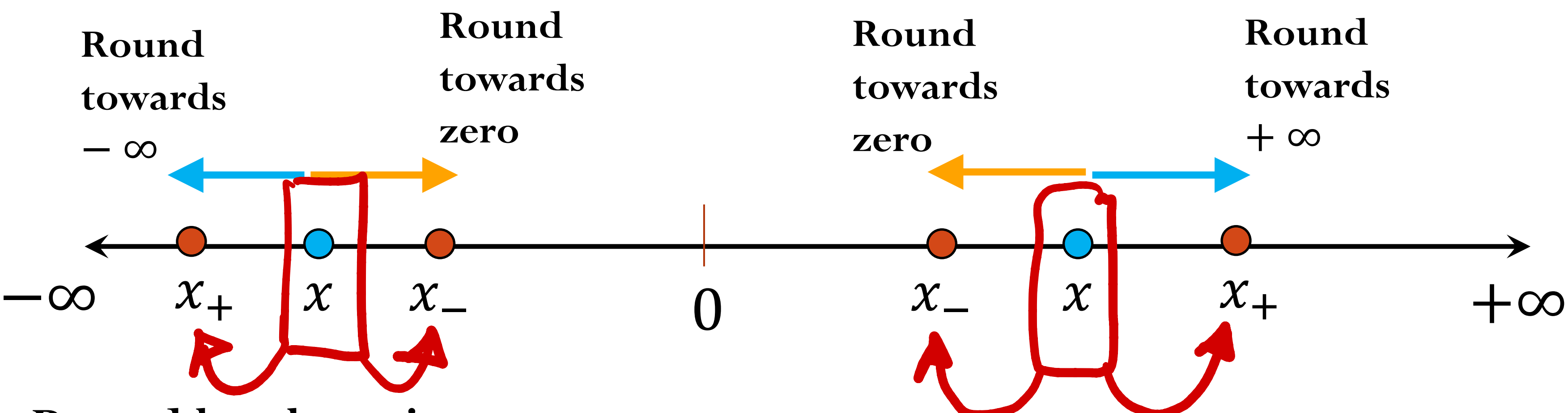
$$\text{gap} = \epsilon_m \times 2^m$$

larger # $\longrightarrow$ larger gap

Rounding

$x \longrightarrow fl(x) \quad = |fl(x) - x|$

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



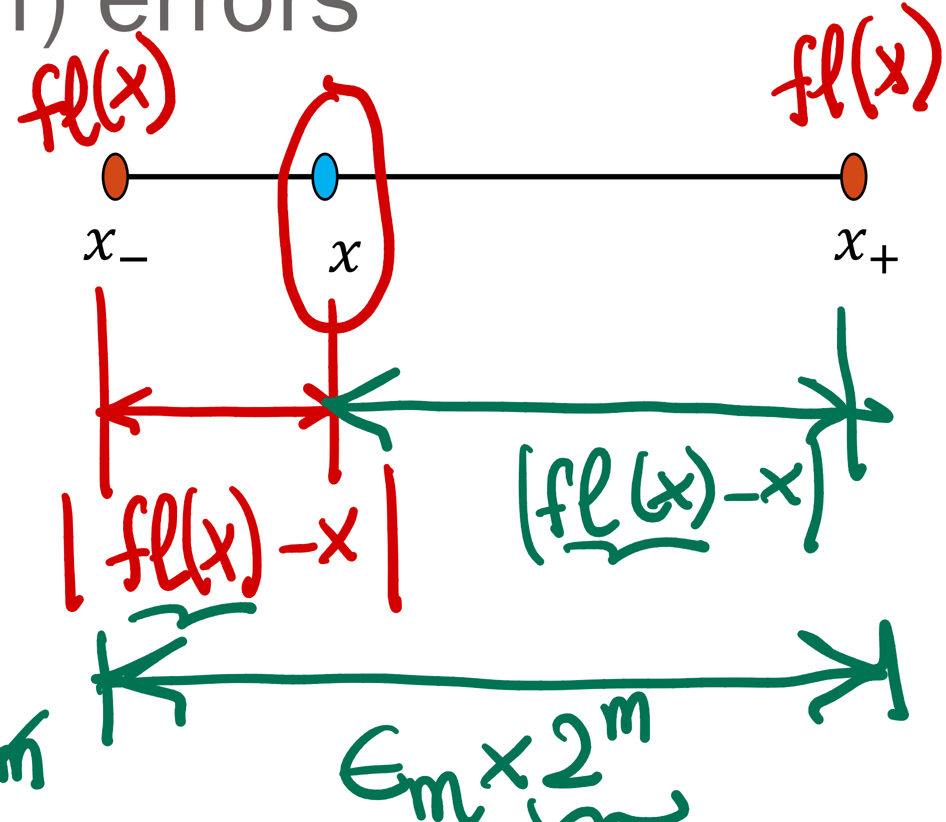
Round by chopping:

	x is positive number	x is negative number
Round up (ceil)	$fl(x) = x_+$ Rounding towards $+\infty$	$fl(x) = x_-$ Rounding towards zero
Round down (floor)	$fl(x) = x_-$ Rounding towards zero	$fl(x) = x_+$ Rounding towards $-\infty$

Round to nearest: either round up or round down, whichever is closer

Rounding (roundoff) errors

Consider rounding by chopping:



- **Absolute error:**

$$|fl(x)-x| \leq \epsilon_m \times 2^m$$

- **Relative error:**

$$\frac{|fl(x)-x|}{|x|} \leq \frac{\epsilon_m \times \cancel{2^m}}{|1.b_1b_2\ldots b_n \times \cancel{2^m}|} = \frac{\epsilon_m}{\underbrace{|1.b_1b_2\ldots b_n|}_{[1,2)}}$$

$$\boxed{\frac{|fl(x)-x|}{|x|} \leq \epsilon_m}$$

Rounding (roundoff) errors

e_r

$$\frac{|fl(x) - x|}{|x|} \leq \epsilon_m$$

The relative error due to rounding (the process of representing a real number as a machine number) **is always bounded by machine epsilon.**

IEEE Single Precision

$$\frac{|fl(x) - x|}{|x|} \le 2^{-23} \approx 1.2\times10^{-7}$$

$e_r \lesssim 1.2\times10^{-7} \lesssim 5\times10^{-7}$

$e_r \lesssim 5\times10^{-7}$

$fl(x) \rightarrow$

7

IEEE Double Precision

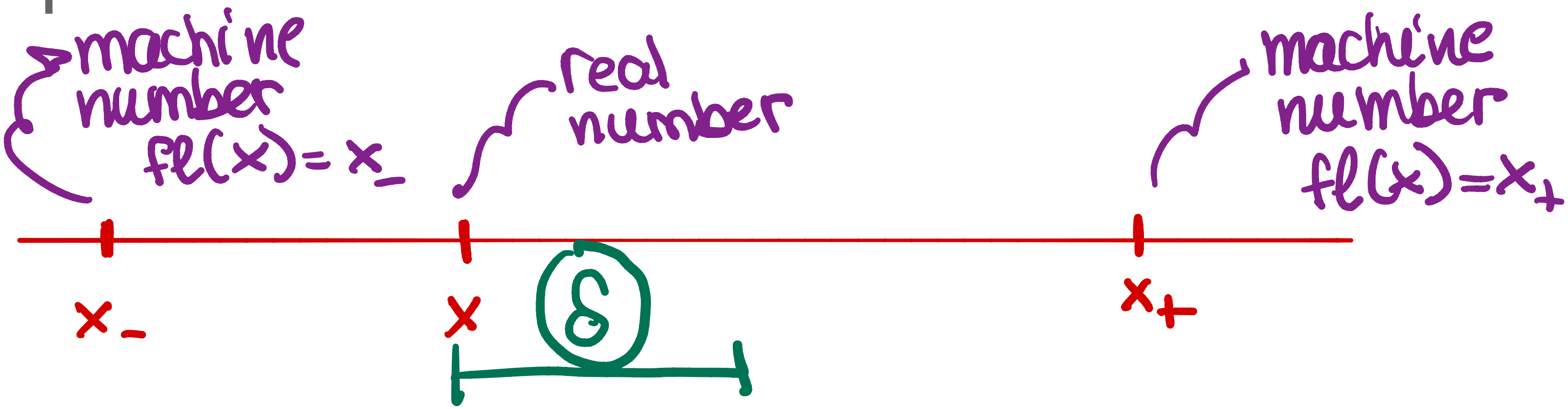
$$\frac{|fl(x) - x|}{|x|} \le 2^{-52} \approx 2.2\times10^{-16}$$

$e_r \lesssim 2.2\times10^{-16}$

$e_r \lesssim 5\times10^{-16}$

16 decimal

Gap between two machine numbers



$\hat{x} = fl(x)$

$fl(x+\delta) = fl(x)$

gap

$\delta < \text{gap}$

Gap between two machine numbers

Rule of Thumbs

Binary $x = q \times 2^m$

$x = 2^8$

$\delta \leq \epsilon_m 2^m \Rightarrow \delta \leq 2^{-23} 2^8$

(single) $\delta \leq 2^{-15}$

$fl(x+\delta) = fl(x)$

$\delta < 2^{-15} \rightarrow fl(x+\delta) = fl(x)$

$\delta > 2^{-15} \rightarrow fl(x+\delta) \neq fl(x)$

Decimal: $x = q \times 10^m$

$x = 4.5 \times 10^4$

double

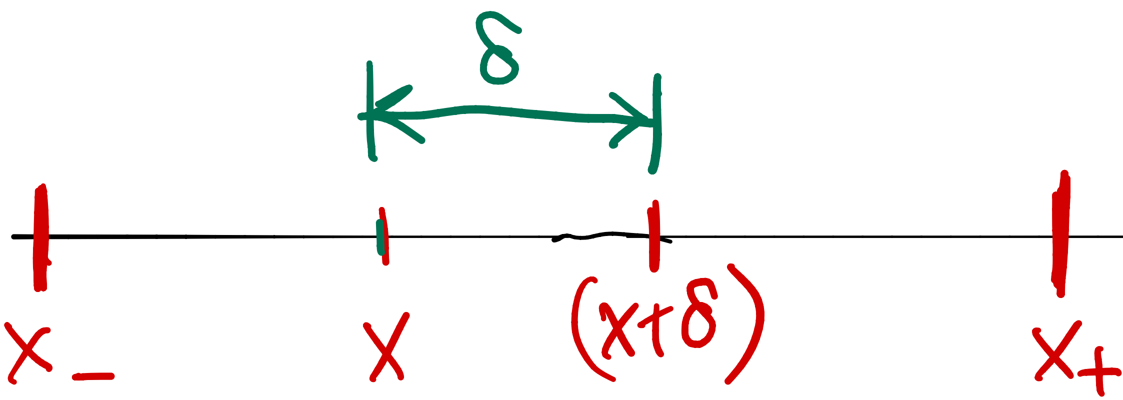
$\delta \leq gap \Rightarrow \delta \leq 10^{-16} 10^4$

$\delta \leq 10^{-12}$

$\delta < 10^{-12} \rightarrow fl(x+\delta) = fl(x)$

$\delta > 10^{-12} \rightarrow fl(x+\delta) \neq fl(x)$

Gap between two machine numbers



$$fl(x) = \hat{x} = \begin{cases} x_+ \\ x_- \end{cases}$$

$$fl(x+\delta) = \hat{x} = fl(x)$$

What is the smallest δ such that $fl(x+\delta) = fl(x) \Rightarrow \delta < \text{gap}$!

$\delta = \text{gap}$

$\delta = \epsilon_m \times 2^m$

$$x + \delta = q \times 2^m + \epsilon_m \times 2^m = \underbrace{(q + \epsilon_m)}_{\neq q} \times 2^m$$

In practice (Rule of Thumb)

~~Show Ipython notebook demos~~

Binary base $x = q \times 2^m$

$fl(x+\delta) = fl(x)$

$\delta < \epsilon_m 2^m$

Example

$x = 2^8$

$\delta < 2^{-23} 2^8 = 2^{-15}$ $\boxed{\delta < 2^{-15}}$

if $\delta < 2^{-15} \Rightarrow fl(x+\delta) = fl(x)$

otherwise $fl(x+\delta) \neq fl(x)$

Decimal base

$x = q \times 10^m$

Example $x = 4.5 \times 10^4$

Double Precision

$\delta < 10^{-16} \times 10^4$

$\boxed{\delta < 10^{-12}}$

Video 2: Arithmetic with machine numbers

Mathematical properties of FP operations

Not necessarily associative:

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

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

```
In [5]: (np.pi+1e100)-1e100
Out[5]: 0.0

In [6]: (np.pi)+(1e100-1e100)
Out[6]: 3.141592653589793

In [7]: b = 1e80
        a = 1e2
        print(a + (b - b) )
        print((a + b) - b )

100.0
0.0
```

Not necessarily distributive:

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

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

```
In [3]: print(100*(0.1 + 0.2))
        print(100*0.1 + 100*0.2)

30.000000000000004
30.0

In [4]: 100*(0.1 + 0.2) == 100*0.1 + 100*0.2
Out[4]: False
```

Not necessarily cumulative:

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

Floating point arithmetic (basic idea)

$$x = (-1)^{\boldsymbol{s}}\, 1.\boldsymbol{f} \times 2^{\boldsymbol{m}}$$

- First compute the exact result
- Then round the result to make it fit into the desired precision

- $x + y = fl(x + y)$

- $x \times y = fl(x \times y)$

Floating point arithmetic

Consider a number system such that $x = \pm 1.b_1b_2b_3 \times 2^m$ for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

$n=3$
$p=4$

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**

$$(1.101)_2 \times 2^1$$
$$\oplus\ (1.001)_2 \times 2^1$$
$$\overline{10.110 \times 2^1}$$

$$a = (1.101)_2 \times 2^1$$
$$b = (1.001)_2 \times 2^1$$
$$c = a + b = \underline{\underline{(10.110)_2 \times 2^1}} = (1.011)_2 \times 2^2$$

Floating point arithmetic

Consider a number system such that $x = \pm 1.b_1b_2b_3 \times 2^m$ for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

- **Example 2: Require rounding**

$$\begin{cases} a = (1.101)_2 \times 2^0 \\ b = (1.000)_2 \times 2^0 \end{cases}$$

$$c = a + b = (10.101)_2 \times 2^0 \longrightarrow 1.0101 \times 2^1$$

$$fl(a+b) = 1.010 \times 2^1$$

- **Example 3:**

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

$$c = a + b = (1.100)_2 \times 2^1 + (0.011)_2 \times 2^1 = (1.111)_2 \times 2^1$$

Floating point arithmetic

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

n=4 → p=5

- **Example 4:**

$$\left\{\begin{array}{l} a = (1.1011)_2 \times 2^1 \\ b = (1.1010)_2 \times 2^1 \end{array}\right\}$$

$$c = a - b = (0.0001)_2 \times 2^1$$

$$1.\boxed{?} \times 2^1$$

$$fl(a-b) = 1.\underbrace{0000} \times 2^1$$

$$\begin{array}{r} 1.1011 \times 2^1 \\ -\ 1.1010 \times 2^1 \\ \hline \underbrace{0.000}1 \times 2^1 \end{array}$$

not signi bits

Floating point arithmetic

Consider a number system such that $x = \underbrace{\pm 1.b_1b_2b_3b_4}_{p=5} \times 2^m$ for $m \in [-4,4]$ and $b_i \in \{0,1\}$.

- **Example 4:**

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

$$c = a - b = (0.0001)_2 \times 2^1$$

$p = 1$

Or after normalization: $c = (1.????)_2 \times 2^{-3}$

- There is not data to indicate what the missing digits should be.
- Machine fills them with its best guess, which is often not good (usually what is called spurious zeros).
- Number of significant digits in the result is reduced.
- This phenomenon is called **Catastrophic Cancellation**.

Loss of significance

Assume a and b are real numbers with $a \gg b$. For example

$$a = 1.a_1a_2a_3a_4a_5a_6 \ldots a_n \ldots \times 2^0$$
$$b = 1.b_1b_2b_3b_4b_5b_6 \ldots b_n \ldots \times 2^{-8}$$

In Single Precision, compute $(a + b)$ $n=23$

$$fl(a) = 1.a_1a_2a_3a_4a_5a_6a_7a_8a_9 \ldots a_{22}a_{23} \times 2^0$$
$$0.000000001b_1b_2 \ldots b_{14}b_{15} \times 2^0$$

$fl(a+b) \Rightarrow$ 15 bits of $\underline{\underline{b}}$

Cancellation

Assume a and b are real numbers with $a \approx b$.

$$a = 1.a_1a_2a_3a_4a_5a_6 \ldots a_n \ldots \times 2^m$$
$$b = 1.b_1b_2b_3b_4b_5b_6 \ldots b_n \ldots \times 2^m$$

In single precision (without loss of generality), consider this example:

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

$$b - a = 0.0000 \ldots 000\boxed{1}\times 2^m \qquad 2^{-23}$$

$$fl(b-a) = 1.\underbrace{\underline{000 - \ldots 00}}_{\text{not sig.}} \times 2^{-23} \times 2^m$$

Examples:

1) ***a* and *b* are real numbers with same order of magnitude** $(a \approx b)$. They have the following representation in a decimal floating point system with 16 decimal digits of accuracy:

$$fl(a) = 3004.45$$
$$fl(b) = 3004.46$$

How many accurate digits does your answer have when you compute $b - a$?

$$\begin{array}{r} 3004.46 \\ -\ 3004.45 \\ \hline 0000.01 \end{array}$$

11 digits

11 digits

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.

Assume you are performing this computation using a machine with 5 decimal accurate digits. Compute $f(10^{-3})$

$$f(10^{-3}) = \underbrace{\sqrt{10^{-6} + 1}}_{1} - 1$$

$$1 - 1$$

$$= \emptyset$$

$$\begin{array}{r} 1.000000 \\ +\ 0.000001 \\ \hline 1.000001 \end{array}$$

Loss of Significance

$(a-b)(a+b)=a^2-b^2$

Re-write the function $\boxed{f(x)=\sqrt{x^2+1}-1}$ to avoid subtraction of two numbers with similar order of magnitude

$$f(x)=\left(\sqrt{x^2+1}-1\right)\left(\frac{\sqrt{x^2+1}+1}{\sqrt{x^2+1}+1}\right)=\frac{\left(\sqrt{x^2+1}\right)^2-(1)^2}{\sqrt{x^2+1}+1}$$

$$=\frac{x^2+1-1}{\sqrt{x^2+1}+1}$$

$$\boxed{f(x)=\frac{x^2}{\sqrt{x^2+1}+1}}$$

$$f(10^{-3})=\frac{10^{-6}}{1+1}=\frac{10^{-6}}{2}\ //$$

Example:

round-down

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?

$fl(x) = 0.37214$

$fl(y) = 0.37202$

error roundoff

$$\left| \frac{fl(x) - x}{x} \right| = 1.308 \times 10^{-5}$$

$x - y = 0.0001234322$

$fl(x-y) = 0.00012$

$$e_r = \left| \frac{(x-y) - fl(x-y)}{(x-y)} \right| \approx 3 \times 10^{-2}$$