Some consequences of floating-point arithmetic error:

1) Loss of significant digits:

This can be seen by drawing a line and marking all the numbers that can be represented by IEEE-754 format.

\[ \cdots \quad 0 \quad \cdots \quad 1 \quad \text{machine epsilon} = 2^{-52} \]

\[ \frac{1}{2^{1022}} \]

\[ 1 = (1.00\ldots0)_2 \times 2^0 \]

next number = (1.00\ldots01)_2 \times 2^0

machine \( \varepsilon = (0.0\ldots01)_2 \times 2^0 = 2^{-52} \)

The red dots represent the number that can be represented with exactness by the IEEE format. We see that these dots get sporadic when moving toward 0 (or \( \infty \)). Thus, if \( x \gg y \) (\( x \) is much bigger than \( y \)) then \( x + y \) will be rounded to \( x \).

Loss of significant digits can also occur when multiplying too big number by a too small number. Some significant digits of the smaller number is lost due to rounding. Then this roundoff error is magnified by multiplication with the large number.

\[ \frac{x (\sqrt{x+1} - \sqrt{x})}{\frac{\text{large}}{\text{small}}} = \frac{x}{\sqrt{x+1} + \sqrt{x}} \]

better for calculation

In Matlab, try two methods for \( x = 10^{200} \):

2) Overflow and underflow:

This is caused by multiplying two too big numbers (overflow)
or two too small numbers (underflow).

\[ E^2: \]

Compute \( \sqrt{x^2 + y^2} \) for \( x = 10^{200} \), \( y = 11^{200} \).

The size of \( z = \sqrt{x^2 + y^2} \) is of the same order as the size of \( x \). But if one squares \( x \) and \( y \), it will result in overflow (recall that the largest number that can be represented in IEEE 754 is about \( 10^{308} \)). Instead, one can compute \( z \) another way:

\[ z = x \sqrt{1 + \left( \frac{y}{x} \right)^2} \]

3) Issue with choosing too small \( h \) in consideration of the limit as \( h \to 0 \):

Consider function \( f(x) = x^2 \). We want to evaluate \( f'(1) \) on computer. By definition,

\[ f'(1) = \lim_{h \to 0} \frac{f(1+h) - f(1)}{h} = \lim_{h \to 0} \frac{(1+h)^2 - 1}{h} \]

Thus,

\[ f'(1) \approx \frac{(h+1)^2 - 1}{h} \]

\( h \) would be considered as 0 in the IEEE 754 format if \( h \leq (0.00...01)_{2} \times 2^{-1022} = 2^{-1024} \)

1+\( h \) would be considered as 1 if \( 1 \leq 1+h \leq (1.0...01)_{2} \times 2^{0} = 1+2^{-52} \)

Therefore, if \( h \) is about \( 2^{-52} \) (or less), 1+\( h \) is considered as 1 and the numerator is equal to zero. The denominator is nonzero as long as \( h > 2^{-1024} \).
$2^{-52} \approx 10^{-16}$

One can check with Matlab that

$$\frac{(1+h)^2 - 1}{h} = 0 \quad \text{when} \quad h = 10^{-16}$$

This error is caused by the nature of floating-point format.

One should be aware of issues like this when programming an algorithm. For example, it is sufficient to select a small $h$ of order $10^{-8}$, but not as small as $10^{-14}$. 