

Notes on rounding modes and measures of error

Let x represent the “exact” value of some quantity, and let \hat{x} denote an approximation to x . There are two principal ways to quantify the error in \hat{x} . The *absolute error* is

$$\text{abserr}(\hat{x}) = |\hat{x} - x|, \quad (1)$$

and, if $x \neq 0$, the *relative error* is

$$\text{relerr}(\hat{x}) = \frac{|\hat{x} - x|}{|x|} = \frac{\text{abserr}(\hat{x})}{|x|}. \quad (2)$$

The relative error puts the absolute error into context. For example, if $\hat{x} = 199$ and $x = 200$, then the absolute error in \hat{x} is 1, and the corresponding relative error is 0.5 percent. However, if $\hat{x} = 2$ and $x = 1$, then the absolute error is still 1, but the relative error is 100 percent. If the “true” value x is unknown, as is often the case in practice, but an estimate of $\text{abserr}(\hat{x})$ is available, then it may be acceptable to approximate $\text{relerr}(\hat{x})$ as $\text{abserr}(\hat{x})/|\hat{x}|$.

Now consider the error that is incurred by rounding the real number x to the floating-point number \hat{x} . To motivate the analysis, we first consider base-10 floating-point arithmetic with six digits, i.e., normalized floating-point numbers of the form $\hat{x} = a_0.a_1a_2a_3a_4a_5 \times 10^e$.

Assume that floating-point values are rounded toward 0 (chopped rounding). In this case, the floating-point number \hat{x} is obtained from the base-10 representation of x by truncating all the digits after a_5 . The absolute error that is incurred, $|\hat{x} - x|$, is largest when the leading truncated digits are 9’s. In this case, the absolute error in \hat{x} is not larger than

$$0.00000999999 \dots \times 10^e,$$

or 10^{e-5} . The same is true when x is positive and the rounding is toward $-\infty$. If x is positive and we round toward $+\infty$, then $|\hat{x} - x|$ is maximal when most of the digits following a_5 are 0; the rounding error is likewise bounded by 10^{e-5} . Analogous arguments give the same bound for negative x .

When we round to the nearest floating-point number, the rounding error is largest when x is exactly halfway between two floating-point numbers. This occurs when $a_6 = 5$ and all subsequent digits are 0. Thus,

$$|\hat{x} - x| \leq 0.000005 \times 10^e = \frac{1}{2} \times 10^{e-5}.$$

It is straightforward to extend this analysis to significands with any number of digits, which yields the following result.

Theorem 1 Consider decimal floating-point numbers with $d + 1$ digits of precision, i.e., floating-point numbers of the form

$$a_0.a_1a_2 \dots a_d \times 10^e.$$

The maximum absolute error incurred by rounding any representable real number is no larger than

- $\frac{1}{2} \times 10^{e-d}$ when rounding to nearest;
- 10^{e-d} for all other rounding modes.

This reasoning can be extended to floating-point numbers in any other even base b . The proof of the following generalization of Theorem 1 is left as an exercise.

Theorem 2 *In floating-point arithmetic with $d + 1$ base- b digits, the maximum absolute error in rounding any representable real number is no greater than*

- $\frac{1}{2} \times b^{e-d}$ when rounding to nearest;
- b^{e-d} for all other rounding modes.

Unit in the last place

We consider floating-point numbers in the usual form

$$\hat{x} = \pm a_0.a_1a_2 \cdots a_d \times b^e, \quad (3)$$

where the representation is assumed to be normalized unless otherwise stated. Suppose that the real number x is representable in the form (3) using one of the usual rounding modes. Our previous discussion implies that either all the digits in the significand of \hat{x} are identical to the leading $d + 1$ digits in the base- b representation of x , or they can be made identical to the leading $d + 1$ digits of x by altering a_d by one unit.

This characterization of rounding error is so convenient in practice that it has a special name. Given a normalized floating-point number of the form (3), a *unit in the last place*, abbreviated ulp, is the change in \hat{x} that arises when the last digit, a_d , of the significand of \hat{x} is altered by one unit. In other words, 1 ulp refers to a change in the *significand* of \hat{x} by b^{-d} and a change in the *absolute value* of \hat{x} by b^{e-d} .

Example 1. Consider base-10 floating-point numbers of the form

$$\hat{x} = a_0.a_1a_2a_3a_4a_5 \times 10^e.$$

In this case, 1 ulp corresponds to the change in the absolute value of \hat{x} when the last digit in the significand is altered by one unit:

$$1 \text{ ulp} = 0.00001 \times 10^e = 10^{e-5}.$$

More generally, if there are $d + 1$ digits of precision (i.e., d digits to the right of the decimal point), then $1 \text{ ulp} = 10^{e-d}$. In 6-digit decimal arithmetic, for instance,

- If $x = 1.23456$ and $\hat{x} = 1.23458$, then the absolute error in \hat{x} is 2 ulp;
- If $y = 123.456$ and $\hat{y} = 123.458$, then the absolute error in \hat{y} also is 2 ulp.

□

The ulp provides a convenient way to quantify the absolute error introduced by rounding. When rounding to nearest, $x \models 1.23456 \times 10^2$ whenever $123.4555 < x < 123.4565$. In this interval, the absolute error incurred by rounding does not exceed $\frac{1}{2}$ ulp, i.e., $\frac{1}{2} \times 10^{-3}$.

Example 2. The approximation of $\pi = 3.14159265\dots$ by the floating-point number $\hat{\pi} = 3.14159 \times 10^0$ when rounding to nearest incurs an absolute error of about 0.265 ulp. When rounding toward infinity, $\pi \models 3.14160 \times 10^0 = \hat{\pi}$, and $|\hat{\pi} - \pi| \approx 0.735$ ulp. □

Example 3. The base-2 representation of $x = \frac{21}{32}$ is $1.01010_2 \times 2^{-1}$. No rounding error occurs when x is rounded to the nearest 5-bit normalized floating-point number. When x is rounded to the nearest 4-bit floating-point number, the rounding error is $\frac{1}{2}$ ulp. ($x \models \hat{x} = 1.010_2 \times 2^{-1}$, because we round to the nearest even digit.) □

The binary representation of $\frac{1}{10}$

One aspect of binary floating-point arithmetic that surprises many people at first is that many common decimal numbers, such as $\frac{1}{10}$, cannot be represented exactly. Let us compute the binary expansion

$$\frac{1}{10} = a_1 \times 2^{-1} + a_2 \times 2^{-2} + a_3 \times 2^{-3} + a_4 \times 2^{-4} + \dots \quad (4)$$

We will show that the coefficients a_i form an eventually repeating sequence. First observe that $\frac{1}{16} < \frac{1}{10} < \frac{1}{8}$; hence $a_1 = a_2 = a_3 = 0$ and $a_4 = 1$. The remainder is

$$\frac{1}{10} - \frac{1}{16} = \frac{3}{80} > \frac{1}{32},$$

so $a_5 = 1$. Therefore, the base-2 representation of $\frac{1}{10}$ is $1.100\dots \times 2^{-4}$. We continue with

$$\frac{1}{10} - \frac{1}{16} - \frac{1}{32} = \frac{1}{160} > \frac{1}{256},$$

so $a_6 = a_7 = 0$ and $a_8 = 1$. Now $\frac{1}{160} - \frac{1}{256} = \frac{3}{1280} > \frac{1}{512}$, so $a_9 = 1$. The remainder is

$$\frac{1}{160} - \frac{1}{256} - \frac{1}{512} = \frac{1}{2560} < \frac{1}{2048},$$

so $a_{10} = a_{11} = 0$. Thus, the base-2 representation of $\frac{1}{10}$ is

$$1.1001100\dots \times 2^{-4}.$$

Might the significand consist of the repeating bit pattern 1100? Note that $0.1100_2 = \frac{3}{4}$, and each successive four-bit string 1100 corresponds to a value that is 2^{-4} times smaller than the previous one. In other words,

$$\begin{aligned}
 1.100\overline{1100}_2 \times 2^{-4} &= 2^{-3} \times (0.1100\overline{1100}_2) \\
 &= 2^{-3} \times \left(\frac{3}{4} + \frac{3}{4} \times 2^{-4} + \dots \right) \\
 &= 2^{-3} \times \left(\frac{\frac{3}{4}}{1 - 2^{-4}} \right) \\
 &= 2^{-3} \times \frac{4}{5} \\
 &= \frac{1}{10}.
 \end{aligned} \tag{5}$$

Therefore, $\frac{1}{10}$ has a repeating base-2 representation—and so cannot be represented exactly with a finite number of bits!

Example 4. In IEEE single precision, the normalized representation of the number $\frac{1}{10}$ is truncated after 24 bits. Under chopped rounding,

$$\frac{1}{10} \models 1.10011001100110011001100 \times 2^{-4}, \tag{6}$$

which corresponds to the 6-term sum

$$\begin{aligned}
 &2^{-3} \times \left(\frac{3}{4} + \frac{3}{4} \times 2^{-4} + \frac{3}{4} \times 2^{-8} + \frac{3}{4} \times 2^{-12} + \frac{3}{4} \times 2^{-16} + \frac{3}{4} \times 2^{-20} \right) \\
 &= 2^{-3} \times \left(\frac{\frac{3}{4}(1 - 2^{-24})}{1 - 2^{-4}} \right) = \frac{1}{8} \times \frac{16}{15} \times \frac{3}{4} \times (1 - 2^{-24}) = \frac{1}{10} \times (1 - 2^{-24}),
 \end{aligned} \tag{7}$$

which is slightly less than $\frac{1}{10}$. In fact, it's about .0999999940, corresponding to an absolute error of approximately 6.0×10^{-8} . \square

The machine epsilon

Suppose that x is a nonzero real number that is representable by a floating-point number \hat{x} in the form (3). What can we say about the relative error that occurs when x is rounded to \hat{x} ? Theorem 2 implies that, for all rounding modes, the relative error in \hat{x} is

$$\frac{|\hat{x} - x|}{|x|} \leq \frac{b^{e-d}}{|x|} \leq \frac{b^{e-d}}{b^e} = b^{-d}. \tag{8}$$

In other words, the relative error in rounding is bounded by b^{-d} , which is just 1 ulp in the floating-point representation of 1.0. The quantity b^{-d} is called the *machine epsilon*, denoted ε .

Example 5. In decimal floating-point arithmetic with 6 digits, nonzero numbers are represented in the form

$$\pm a_0.a_1a_2a_3a_4a_5 \times 10^e,$$

where $1 \leq a \leq 9$, which implies that the machine epsilon is $\varepsilon = 10^{-5}$. The relative error in rounding any representable real number x to a floating-point number \hat{x} in this format is never larger than ε , and in a round-to-nearest-mode, the relative rounding error is bounded by $\frac{1}{2}\varepsilon$. For instance, the absolute error in rounding $x = 123.4567$ to $\hat{x} = 1.23457 \times 10^2$ (i.e., rounding to nearest) is 0.0003, and the relative error is

$$\frac{0.0003}{123.4567} \approx 2.43 \times 10^{-6} = 0.243\varepsilon.$$

□

Example 6. An IEEE single-precision number is a base-2 floating-point number with a 24-bit significand. The machine epsilon is 2^{-23} , or approximately 1.19×10^{-7} . For this reason, some people say that IEEE single-precision arithmetic is approximately equivalent to 7 decimal digits.

□

Example 7. An IEEE double-precision number is a binary floating-point number with a 53-bit significand. The machine epsilon is 2^{-52} , or approximately 2.22×10^{-16} (roughly equivalent to 16 decimal digits). □

Using Eq. (8), the results of Theorem 2 can be re-cast in the following statement.

Theorem 3 *If x is a nonzero real number that is representable as a normalized floating-point number \hat{x} with $d + 1$ base- b digits of precision, then there is a real number δ such that*

$$\hat{x} = x(1 + \delta), \tag{9}$$

where $|\delta| \leq \varepsilon$ in all rounding modes. (Here ε is the machine epsilon, $\varepsilon = b^{-d}$.) When rounding to the nearest floating-point number, $|\delta| \leq \frac{1}{2}\varepsilon$.