Some myths about floating-point arithmetic

Steven G. Johnson

September 13, 2013

(This list is adapted from the "prevalent misconceptions about floating-point arithmetic" by William Kahan's 2004 presentation "How Java's Floating-Point Hurts Everyone Everywhere.")

As in Trefethen's book, we denote floating point operations by $\oplus, \otimes, \ldots$. We denote the set of floating-point numbers by $\mathbb{F}$, and $\mathrm{fl}(x)$ denotes the closest element of $\mathbb{F}$ to $x \in \mathbb{R}$. Assuming x does not overflow or underflow (exceed the max/min exponent), two key facts are that $|\,\mathrm{fl}(x) - x| \le \epsilon |x|$, where ϵ is the machine precision, and that (assuming IEEE "correct rounding") $x \quad y = \mathrm{fl}(x \cdot y)$ for binary operations $\cdot \in \{\times, \pm, /\}$. The other key fact is to understand that $\mathbb{F}$ is a specific set of rational numbers: p-digit integers multiplied by powers of two (in binary floating-point) or powers of 10 (in decimal floating point).

A number of pernicious myths about floating-point arithmetic are prevalent. They include:

- A unpredictable random number of order ϵ is added to every result. e.g. $1 \oplus 1$ may give $2 \pm \epsilon$, and $0 \otimes x$ may give $\pm\epsilon$. **False**. (e.g. $1 \oplus 1$ always gives exactly 2, and $0 \otimes x$ always gives exactly 0 [unless x is $\pm$Inf or NaN], since 2 and 0 are exactly representable.)

- Integer arithmetic is more accurate than floating-point arithmetic. **False**. (See above: integer arithmetic is performed exactly in floating-point.)

- Integer arithmetic is much faster than floating-point arithmetic. **False** on any modern general-purpose CPU. (True in 1985, or on small embedded systems.)

- Computational **precision** (the number of digits stored) is the same thing as the computational **accuracy**. **False**. (Numbers can be much more accurate than the number of digits stored, e.g. integers are stored exactly, or much less accurate, e.g. due to error accumulation.)

- "Arithmetic much more precise than the data it operates upon is needless, and wasteful." (Kahan) **False**: even if you only need 3 significant digits in the final result, you may need many more digits at intermediate steps.

- Floating-point arithmetic performs "random" rounding of decimal fractions, e.g. 0.1 or 3.1415. **True** for *binary* floating-point, but **False** for *decimal* floating-point (available in many software libraries).

- "In floating–point arithmetic nothing is ever exactly 0 ; but if it is, no useful purpose is served by distinguishing +0 from -0." (Kahan) **False.**

- "Progress is inevitable: When better formulas are found, they supplant the worse." (Kahan) Sadly, **False**.