Numerical Differentiation is Hard

When deriving numerical differentiation formulae we made the assumption that we could evaluate f at any $x \in (a,b)$. Actually we can't. We can approximate x by a float, and compute a floating point approximation to f at that float. Let's write that approximation as

$$\overline{f(x)} \equiv \mathrm{fl}(f(\mathrm{fl}(x)).$$

Note that $\overline{f(x)}$ depends not only on f and x, but also on our machine precision, μ, and the algorithm and code for evaluating f.

So now let's look more closely at our finite difference formulas. The general formula

$$f'(x) = P'(x) + R'(x)$$

accounts for the truncation error, R, but doesn't account for rounding errors, so write

$$P'(x) = \overline{P'(x)} + E(x),$$

where $\overline{P'(x)}$ is our floating point approximation to $f'(x)$. Let's bound the the total error

$$f'(x) - \overline{P'(x)} = R'(x) + E(x) :$$

A $(k+1)$-point difference formula has the form $P'(x_0) \approx \frac{1}{h}\sum_{i=0}^{k} c_i f(x_i)$, so

$$f'(x_0) = P'(x_0) + R'(x_0) = \frac{1}{h}\sum_{i=0}^{k} c_i f(x_i) + ch^k f^{(k+1)}(\xi).$$

To find $E(x)$ we need to find the errors in evaluating f, and the additional errors in computing the sum above. We don't know these errors, so we let $f(x) = \overline{f(x)} + e(x)$ account for both the errors in evaluating f *and* evaluating the sum. $e(x)$ will depend on μ, the c_i, and the condition numbers for "evaluate f at x_j" (which depend on $|f'(x_j)|$).

$$\begin{aligned} \overline{P'(x_0)} &= \tfrac{1}{h}\textstyle\sum_{i=0}^{k} c_i(f(x_i) + e(x_i)) \\ &= P'(x_0) + \tfrac{1}{h}\textstyle\sum_{i=0}^{k} c_i e(x_i). \end{aligned}$$

Now if $|e(x)| \le M_r$ in (a,b) and $|c_i| \le C$, we can finally bound the rounding error:

$$|E(x_0)| \le \frac{1}{h}(k+1)CM_r.$$

If $|f^{(k+1)}(x)| \le M_t$ in (a,b), we can bound the truncation error and write the total error

$$|R'(x_0) + E(x_0)| \le |R'(x_0)| + |E(x_0)| \le ch^k M_t + (k+1)CM_r/h.$$

Here is the crux of a fundamental difficulty in scientific computation: *discrete calculus*. There are only 2 parameters which we can control, h and k. Fix $k \in \mathbb{Z}^+$. How can we make the truncation error smaller? What does that do to the rounding error? How do we make the rounding error smaller, and what does that do to the truncation error? The situation is ameliorated somewhat by increasing k (why?), but there is a similar phenomenon with k: increasing k demands more of f, reduces the truncation error, and increases the rounding error (by increasing both $k+1$ *and* C). In our choice of h and k we try to steer between two monsters, hoping to minimize our loss...