The Secant Method

Our last comment about Newton's method bemoaned its lack of generality. We address this here, and will end up with a general purpose method that is arguably faster than Newton's method.

The generality of Newton's method is compromised by the need to evaluate both $f(x_k)$ and $f'(x_k)$ at each iteration; and quite often f' is unknown (or even unknowable). The secant method is simply Newton's method with $f'(x_k)$ replaced by an approximation. To get more specific, recall that we can define $f'(a)$ as

$$f'(a) = \lim_{b \to a} \frac{f(a) - f(b)}{a - b}.$$

The limitations of our arithmetic prevent us actually taking the limit, but we can get an approximation to $f'(a)$ simply by taking $|a - b|$ small. If we use this approximation to the slope of the tangent line through $(x_k, f(x_k))$ we get the line

$$y - f(x_k) = \frac{f(x_k) - f(b)}{x_k - b}(x - x_k),$$

which is what your calculus text called a *secant line*.

Now, what should we use for our b? We want b to be close (but not equal) to x_k, and we will need to find $f(b)$. Since we computed $f(x_{k-1})$ in the previous iteration, a particularly efficient choice for b is simply $b = x_{k-1}$, giving

$$y - f(x_k) = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}(x - x_k).$$

Letting x_{k+1} be the x-intercept of this line gives the *secant iteration*

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}.$$

This was a linearization similar to Newton's method, but designed to avoid f' by approximating it in an efficient way. Does our approximation make the method slower? Yes and no. It usually takes the secant method more iterations than Newton's method to achieve the same accuracy. But (except for the first iteration), the secant method requires only one function evaluation per iteration as compared to two for Newton's method.

So who wins? You, of course, because you get to choose which method to use. Who wins is an important question, and we will have to build some machinery before we can answer. The short answer: secant is usually faster (but keep Newton's method in your toolbox!).