Topic: Root Finding

Nick Battista

Date Created: 5/20/2014
Date Modified: 7/15/2016

1 BACKGROUND

The main problem we are interested in solving involves finding a root, $\alpha \in [a,b]$, such that, for some continuous function and differentiable function, $f(x)$, on $[a,b]$ that $f(\alpha) = 0$.

Root finding can be done in n-dimensions. In most cases, going from a 1-dimensional algorithm to n-dimensions is fairly straight forward, and the analysis is pretty much analogous, but a bit hairier. We will begin by introducing a few basic algorithms and their analysis for order of convergence to root.

2 THE BISECTION METHOD

The most natural algorithm to think about, *the bisection method*, involves successively cutting a search interval in half repeatedly until the root is found to a specified precision. However, it is clear that this is not the most efficient or elegant algorithm to find a root. (Although the author says it has many splendid applications when searching for points along arcs when functions are almost singular...mo' derivatives, mo' problems).

2.1 BISECTION ALGORITHM

Let $a \leq x_1 < x_2 \leq b$. If $f(x_1)$ and $f(x_2)$ have different signs, i.e. $f(x_1) \cdot f(x_2) = 0$. Let $\epsilon > 0$, then

while $err > \epsilon$ **do**

$c = \frac{x_1+x_2}{2}$
if $f(x_1) \cdot f(x_2) < 0$ **then**
$x_2 = c$
else
$x_1 = c$
end if
$err = x_2 - x_1$
end while

2.2 Analysis of Bisection / Why slow?

After k iterations, $x_n - x$, where $x_n = \frac{x_1+x_2}{2}$, satisfies $\|x_n - x\| < \frac{|b-a|}{2^{k+1}}$

Why slow? Consider we want to get 10 digit accuracy, then we need

$$\frac{1}{2^k} < 10^{-10}$$
$$k\log 2 > 10\log 10$$
$$k > 10\frac{\log 10}{\log 2} \approx 33 \text{ iterations}$$

If we know f is smooth(er), we can improve this drastically!

Conclusion: When to use the bisection method? If speed (or cuteness) doesn't matter (okay...), but if f is not smooth, bisection may be the way to go.

3 Fixed Point Iterations

General recursive algorithms, have a functional form such as $x_{n+1} = g(x_n, x_{n-1}, ..., x_{n-m})$. In the case of root finding we typically have the scenario, where $x_{n+1} = g(x_n)$.

If β is a fixed point of $g(x)$ then $\beta = g(\beta)$. The way we construct $g(x)$ is such that a fixed point of $g(x)$ will also be a root of $f(x)$. That would be groovy. Our root finding problem then manifests itself as finding a fixed point for $g(x)$. Is this just a new can of worms? Maybe? Sometimes?

However, if we have chosen $g(x)$ in a logically sound manner, we see we should get convergence to our desired root by simple criteria- If $|g'(\alpha)| < 1$ in some neighborhood of α, then

convergence is guaranteed. In more mathy language, if $x_{n+1} = g(x_n)$ such that $g : \mathbb{R} \to \mathbb{R}$, $g(\alpha) = \alpha$, and $|g'(\alpha)| < 1$, then in some neighborhood of α, e.g., if $|x - \alpha| < \epsilon$ for $\epsilon > 0$, then $x_n \to \alpha$.

3.1 Analysis of Fixed Point Iterations

We will now employ the power of Taylor series to do all the heavy lifting for us. Let α be a root of $f(x)$, i.e., $f(\alpha) = 0$ and let the error at the $(n+1)^{th}$ step we defined as,

$$e_{n+1} = x_{n+1} - \alpha.$$

Using cute, subtle math tomfoolery, we get

$$\begin{aligned} e_{n+1} &= x_{n+1} - \alpha \\ &= g(x_n) - \alpha \\ &= g(\alpha + e_n) - \alpha \\ &= \left[g(\alpha) + e_n g'(\alpha) + \frac{1}{2!} e_n^2 g''(\alpha) + \frac{1}{3!} e_n^3 g'''(\alpha) + \ldots \right] - \alpha \\ &= e_n g'(\alpha) + \frac{1}{2!} e_n^2 g''(\alpha) + \frac{1}{3!} e_n^3 g'''(\alpha) + \ldots, \end{aligned}$$

since $\alpha = g(\alpha)$ by definition of our fixed point. Hence we see that

$$e_{n+1} = e_n g'(\alpha) + \ldots = \mathcal{O}(e_n),$$

and if $|g'(\alpha)| < 1$ we get convergence to the root. Yay. Note, however, this analysis just showed that fixed point iterations (at worst) exhibit linear convergence. Nay.

4 Newton's Method

Probably the most widely used method, at least by Calculus 1 students for a few miserable days, is *Newton's Method*. Newton's method at its roots (ha) begins by looking at a general functions Taylor Series, doing a linear approximation, and using that information to develop an algorithm to approximate a root.

4.1 Derivation of Newton Method

Consider a function $f(x)$ that is at least C^2, but for our purposes here we'll assume it is C^∞ (We'll see why we want it to be at least C^2 when we do the error analysis). If we just write $f(x)$ as its Taylor series centered around some number x_n, we see

$$f(x) = f(x_n) + (x - x_n)f'(x_n) + \frac{(x-x_n)^2}{2!}f''(x_n) + \frac{(x-x_n)^3}{3!}f'''(x_n) + \ldots$$

Now do what engineers love and truncate after the linear term!

$$f(x) \approx f(x_n) + (x - x_n)f'(x_n)$$

The idea now is that if we solve for x in the equation above, it will approximate a root of $f(x)$. Solving for x we get

$$x = x_n - \frac{f(x_n)}{f'(x_n)}.$$

Finally, since we can do more than one iteration of Newton's (contrary to the dismay of Calculus students), if we let $x = x_{n+1}$ we get the usual form of Newton's,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

The other way, for those geometrically minded, to derive Newton's is to pick an initial guess, say x_n, and to approximate the function, $f(x)$, using a tangent line at x_n. Finding the zero of that tangent line is what we call the next guess for the root of $f(x)$.

4.2 Newton's Convergence Analysis

Since Newton's Method is an example of a fixed point iteration, we can use the same kind of Taylor series analysis to find its convergence order. Recall for Newton's,

$$x_{n+1} = g(x_n) = x_n - \frac{f(x_n)}{f'(x_n)}.$$

Let α be a root of $f(x)$, i.e., $f(\alpha) = 0$ and let the error at the $(n+1)^{th}$ step we defined as,

$$e_{n+1} = x_{n+1} - \alpha.$$

Again doing some cute mathematical slight of hand, we get

$$\begin{aligned} e_{n+1} &= x_{n+1} - \alpha \\ &= g(x_n) - \alpha \\ &= g(\alpha + e_n) - \alpha \\ &= \left[g(\alpha) + e_n g'(\alpha) + \frac{1}{2!} e_n^2 g''(\alpha) + \frac{1}{3!} e_n^3 g'''(\alpha) + \ldots \right] - \alpha \end{aligned}$$

Recall for fixed point schemes that $\alpha = g(\alpha)$, hence we find that

$$e_{n+1} = e_n g'(\alpha) + \frac{1}{2!} e_n^2 g''(\alpha) + \ldots$$

Now we see that $g'(\alpha) = 0$, i.e.,

$$g'(\alpha) = \frac{d}{dx}\bigg|_{x=\alpha} = \frac{d}{dx}\left(x - \frac{f(x)}{f'(x)} \right)\bigg|_{x=\alpha} = \frac{f(x)f''(x)}{f'(x)^2}\bigg|_{x=\alpha} = \frac{f(\alpha)f''(\alpha)}{f(\alpha)^2} = 0,$$

which shows Newton's method exhibits 2^{nd} order convergence to the root, i.e.,

$$e_{n+1} = \mathcal{O}(e_n^2).$$

Note we see above why $f(x)$ needs to be at least C^2 otherwise, the statement above has no meaning. Also, we find that if α is not a single root that the above analysis does not imply 2^{nd} order convergence.

4.3 What can go wrong with Newton's?

Things that can go wrong:

- If a root has multiplicity greater than 1, it exhibits less than 2^{nd} order convergence. You can find this order by standard Taylor series analysis. → **use *Aitken extrapolation* to bump up the order!**
- If $f(x)$ is not continuous or at least C^2
- Also, sometimes computing derivatives is a pain in the butt...

5 The Secant Method

Now Newton has a kid sister, the *Secant Method*. The essence of the Secant methods feels much of the same as Newton's; however, Secant uses the two previous iterates to compute the next approximation to the root. There are some very strong similarities.

Thinking geometrically, Secant's story begins with two guesses to the root. Rather than forming any tangent lines, as in Newton's case, Secant very cleverly forms a secant line between the two guesses. Analytically, we find the zero of that secant line and that root is our next guess, i.e,

Consider two points, x_n and x_{n-1}, with respective function values $f(x_n)$ and $f(x_{n-1})$. The equation of the secant line between the points is

$$y(x) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}(x - x_n) + f(x_1).$$

Setting that line equal to zero, and solving for x we find

$$x = x_n - \frac{f(x_n)}{\left[\frac{f(x_n)-f(x_{n-1})}{x_n - x_{n-1}}\right]}$$

Since x is our next guess we let $x_{n+1} = x$, we see

$$x_{n+1} = x_n - \frac{f(x_n)}{\left[\frac{f(x_n)-f(x_{n-1})}{x_n - x_{n-1}}\right]}$$

This has a very similar form to Newton's method; however, rather than having an actual derivative in the algorithm, Secant simply employs an approximation to a derivative.

6 Aitken Extrapolation

The essence of Aitken is very sly, it takes a first order fixed point method and cleverly transforms it into a second order method. The only downside is for every full iteration in the Aitken scheme, we have to do the original fixed point method twice. Bummer? Nah, let's check it out!

Recall our original problem is finding a root, α, of some continuous function $f(x)$. To do this, we tried to use some fixed point iterative scheme, $x_{n+1} = g(x_n)$ but only found that our

scheme exhibited linear convergence. Now let's derive Aitken's scheme and make a 2^{nd} order method.

6.1 Aitken Extrapolation Derivation

Consider two consecutive iterations of the original fixed point scheme with their associated errors,

$$\begin{aligned} x_{n+1} &= g(x_n) \qquad e_{n+1} = x_{n+1} - \alpha \\ x_{n+2} &= g(x_{n+1}) \qquad e_{n+2} = x_{n+2} - \alpha \end{aligned}$$

Remember since the fixed point scheme only showed linear convergence we have,

$$e_{n+1} = e_n g'(\alpha) + \mathcal{O}(e_n^2) + \ldots = Ce_n + \mathcal{O}(e_n^2),$$

From above we see that,

$$\begin{aligned} e_{n+1} &= x_{n+1} - \alpha = C(x_n - \alpha) \\ e_{n+2} &= x_{n+2} - \alpha = C(x_{n+1} - \alpha) \end{aligned} \tag{6.1}$$

Subtracting the top equation from the bottom, we find

$$x_{n+2} - x_{n+1} = C(x_{n+1} - x_n),$$

and hence we get that

$$C = \frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n}.$$

Now plugging that into the bottom equation in (6.2), we see

$$\begin{aligned} x_{n+2} - \alpha &= C(x_{n+1} - \alpha) \\ \alpha &= \frac{x_{n+2} - Cx_{n+1}}{1 - C} \\ \alpha &= x_{n+2} - \frac{(x_{n+2} - x_{n+1})^2}{x_{n+2} - 2x_{n+1} + x_n} \end{aligned} \tag{6.2}$$

6.2 Algorithm: Aitken Extrapolation

Let's discuss briefly how to implement this. It will help clarify all the elegance of this extrapolation scheme.

x_0 = initial guess, $n = 0, \epsilon > 0$
while $err > \epsilon$ **do**
  $y_1 = g(x_n)$
  $y_2 = g(y_1) = g(g(x_n))$
  $x_{n+1} = y_2 - \frac{(y_2 - y_1)^2}{y_2 - 2y_1 + x_0}$
  $err = x_{n+1} - x_n$
  $n = n + 1$
end while

It is clear that for each step in Aitken's we have to do roughly $3x$ the amount of work to get a 2^{nd} order scheme, since we have to evaluate what are essentially 3 separate fixed point iterative scheme computations.

To show how Aitken's is a fixed point scheme, consider the algorithm written as:

$$x_{n+1} = H(x_n) = g(g(x_n)) - \frac{(g(g(x_n)) - g(x_n))^2}{g(g(x_n)) - 2g(x_n) + x_n},$$

where $g(x)$ is the original 1^{st} order fixed point scheme.

6.3 Proof Aitken Extrapolation is 2^{nd} Order

There are two ways to go about this - the fun way, and the *more* fun way. I'll briefly discuss both ways, but we will only perform the first method...no one should have too much excitement, after all. The less fun method involves a calculation similar to the derivation of Aitken's Extrapolation itself; however, we will hold onto all the error terms and see how they change throughout the computation.
Recall in the derivation we had for a standard first order scheme, that

$$e_{n+1} = e_n g'(\alpha) + \frac{1}{2!} e_n^2 g''(\alpha) + \mathcal{O}(e_n^3) = Ce_n + De_n^2 + \mathcal{O}(e_n^3) \tag{6.3}$$

$$\tag{6.4}$$

$$e_{n+2} = e_{n+1} g'(\alpha) + \frac{1}{2!} e_{n+1}^2 g''(\alpha) + \mathcal{O}(e_{n+1}^3) = Ce_{n+1} + De_{n+1}^2 + \mathcal{O}(e_{n+1}^3) \tag{6.5}$$

$$\tag{6.6}$$

where $C = g'(\alpha) \neq 0$ and $D = \frac{1}{2!} g''(\alpha)$. Upon substituting the above two equations, we find

$$e_{n+2} - e_{n+1} = C(e_{n+1} - e_n) + D(e_{n+1}^2 - e_n^2) + \mathcal{O}(e_n^3),$$

since $|e_{n+1}| < |e_n|$.

Now substituting $e_{n+2} = x_{n+2} - \alpha$, $e_{n+1} = x_{n+1} - \alpha$, and $e_n = x_n - \alpha$, we get

$$\begin{aligned} x_{n+2} - x_{n+1} &= C(x_{n+1} - x_n) + D(x_{n+1} - x_n)(e_{n+1} + e_n) + \mathcal{O}(e_n^3) \\ &= C(x_{n+1} - x_n) + D(x_{n+1} - x_n)\mathcal{O}(e_n) + \mathcal{O}(e_n^3), \end{aligned}$$

since $e_{n+1} - e_n = \mathcal{O}(e_n)$. Now solving the above for C, we obtain

$$C = \frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n} + \mathcal{O}(e_n). \tag{6.7}$$

We find that C is only 1^{st}-order accurate, but will that be enough to to make the next iterate, 2^{nd}-order accurate? Let's check it out. Substitute (6.7) into (6.5), we find that

$$\begin{aligned} e_{n+2} = x_{n+2} - \alpha &= C(x_{n+1} - \alpha) + \mathcal{O}(e_n^2) \\ &= \left[\frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n} + \mathcal{O}(e_n) \right] (x_{n+1} - \alpha) + \mathcal{O}(e_n^2) \quad \text{where} \quad K = \frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n} \\ &= K(x_{n+1} - \alpha) + \mathcal{O}(e_n)\mathcal{O}(e_{n+1}) + \mathcal{O}(e_n^2), \qquad \text{since} \quad e_{n+1} = x_{n+1} - \alpha \\ &= K(x_{n+1} - \alpha) + \mathcal{O}(e_n^2), \qquad \text{as} \quad \mathcal{O}(e_n)\mathcal{O}(e_{n+1}) \le \mathcal{O}(e_n^2). \end{aligned}$$

Solving for α, we find

$$\alpha = \frac{x_{n+2} - K x_{n+1}}{1 - K} + \mathcal{O}(e_n^2),$$

and hence

$$\alpha = x_{n+2} - \frac{(x_{n+2} - x_{n+1})^2}{x_{n+2} - 2x_{n+1} + x_n} + \mathcal{O}(e_n^2).$$

Therefore our next iterate, α, is 2^{nd}-order accurate.
The other "straight-forward" proof one can do to prove the 2^{nd}-order accuracy bump from using Aitken's Extrapolation is by using Taylor Series, exclusively. We consider

$$x_{n+1} = H(x_n) = g(g(x_n)) - \frac{(g(g(x_n)) - g(x_n))^2}{g(g(x_n)) - 2g(x_n) + x_n},$$

and go through the same rigmarole as we did for Newton's Method, namely, computing

$$
\begin{aligned}
e_{n+1} = x_{n+1} - alpha &= H(x_n) - \alpha \\
&= H(\alpha - e_n) - \alpha \\
&= \left[H(\alpha) + e_n H'(\alpha) + \frac{1}{2!} e_n^2 H''(\alpha) + \mathcal{O}(e_n^3) \right] - \alpha \\
&= e_n H'(\alpha) + \mathcal{O}(e_n^3).
\end{aligned}
$$

Simply, show that $H'(\alpha) = 0$ and the proof is in the pudding. We note, if you ever want to feel like you're doing a lot of mathematical work, but at the same time, not a lot at all, we suggest you proceed with this calculation.