Other Ways to Find Roots

Written by Dave Pawlowski

1 Indroduction

So far we've covered finding the roots of a quadratic. Now lets work on solving equations that are a bit more general. We'll cover two methods here, the bisection method and the Newton-Raphson method.

2 Bisection Method

Say that you are given a function, $f(x)$ over the interval $[a, b]$, and at some point in the interval $f(x)$ changes sign. The bisection method is thus, our first step is to divide the interval in half and note that f must change sign on either the right half or the left half of the interval (or be zero at the midpoint of the interval). Next, the interval $[a, b]$ is replaced with the half-interval in which $f(x)$ changes sign and we repeat the process. We iterate by halving the interval and selecting the sub-interval that has a sign change until the sub-interval has a length of ϵ, or our tolerance. If the sub-interval has the width of ϵ then we know that our approximation to the root (we choose our updated a or b) must be within the tolerance of the root, since the root is within the new range.

Note that since we are always reducing the interval by half, this method is linear in order of convergence. Additionally, as long as our function, $f(x)$ changes sign within our initial interval, $[a, b]$, the bisection method will converge. So it is an extremely reliable method, if not the most efficient. Note that this method will still work if there are more than one roots within our interval. In particular, it will always work for an odd number of roots. If a given function has an even number of roots, it is necessary to choose the initial interval such that it only brackets an odd subset.

In order to implement such an algorithm, one would use the following steps:

1. Input the range, $[a, b]$
2. Input the tolerance
3. Test the ends of the range to see if they are one of the roots or if they bracket a root
4. Begin iteration
5. set $c = \frac{a+b}{2}$

6. if $f(a)f(c) > 0$ then $a = c$ else $b = c$
7. repeat until $abs(a - b) < tolerance$

3 Newton-Raphson Method

In order to create an algorithm that is a bit more efficient than the bisection method, it is useful to have a little bit more information about the function. One such method is based on expanding our function $f(x)$ about the point $x = p$ as a Taylor series. If we do this, and only keep first order derivatives, we get:

$$f(p) = f(x) + f'(x)(p - x) + \dots.$$

It is important to note that since we got rid of higher order terms, we can't use this method to find an exact solution, but we can still get a pretty decent approximation. Solving for p in the above equation gives:

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

This defines an iterative method, so if we start with an initial guess p_0 then we would have a method that looks like:

$$p_{n+1} = p_n - \frac{f(p_n)}{f'(p_n)}$$

and we iterate over n until a sufficiently accurate value is reached.

Unlike the bisection method that we already discussed, there is no guarantee that Newton's method converges. This is obvious for the case where $f'(p_n) = 0$. Usually, Newton's method requires a that you choose your initial guess "close" to the actual solution. If this is done, the method converges quadraticaly.

In addition, since there is no guarantee that our method converges, it is always a good idea to limit the number of iterations that your program can execute, less it runs forever. If the method doesn't converge by the name you reach some maximum number of iterations, you give up and try to find an alternative.

Finally, in order to use Newton's method, you need to have knowledge of your function's derivative. Hopefully it is easy to find this analytically, as we have yet to discuss doing so numerically.

Implementation of Newton's method should follow the following steps:

1. Input the initial guess p
2. Test if $f(p) = 0$. If it is, you're done!
3. Begin iteration
4. set $p_{new} = p - f(p)/f'(p)$

5. repeat until $abs(p - p_{new}) < tolerance$ **or** $n > n_{max}$