Introduction to Scientific Computing
Lecture 5

Professor Hanno Rein

Last updated: October 14, 2017

0 Root finding (continued)

0.3 Bisection method

All the methods we will discuss to solve the root finding problem are iterative. That means we start with a guess and improve upon our guess during every iteration. The bisection method is one such example.

Let's formulate the problem a bit more precisely. You are given a one dimensional real function on the interval $I = [a, b]$. You can assume that the function is continuous. The problem is then to find the value $d \in [a, b]$ for which $f(d) = c$. You are given c, usually we have $c = 0$.

The bisection method works as follows. We first divide the interval $[a, b]$ into two intervals $[a, \frac{a+b}{2}]$ and $[\frac{a+b}{2}, b]$. We then evaluate the function at the middle point $\frac{a+b}{2}$. If the value $f(\frac{a+b}{2})$ is smaller than c, then we know that our answer must lie in the upper interval $[\frac{a+b}{2}, b]$. Similarly, if the value $f(\frac{a+b}{2})$ is larger than c, then we know that our answer must lie in the lower interval $[a, \frac{a+b}{2}]$. We now have a better estimate of the value d. By simply repeating the process, we can make it more and more accurate. Because we divide the interval in half every time, the method converges quickly.

Below is an illustration of the bisection method and an implementation in pseudo code.

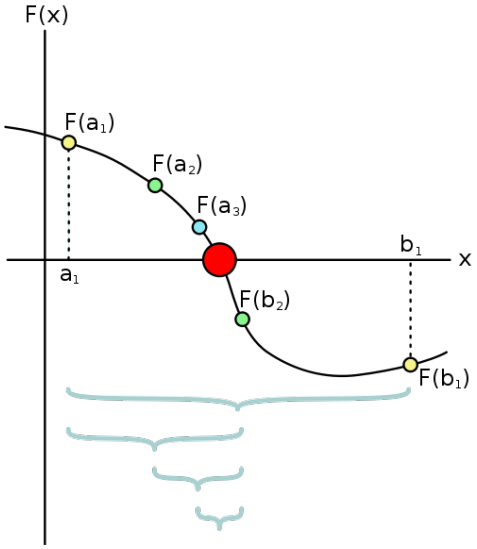

Figure 1: Bisection methods. Source: Wikipedia.

```
set a
set b
fa = f(a)
fb = f(b)
while ( (b-a) > epsilon ){
    m = (a+b)/2
    fm = f(m)
    if ( (fm>0 and fa<0) or (fm<0 and fa>0) ){
        b = m
        fb = f(b)
    }else{
        a = m
        fa = f(a)
    }
}
print [a:b]
```

Code 1: Pseudo code of the bisection method.

Ok. So the method improves the result at every iteration and the interval gets small. The question is: Using standard double floating point precision, how many iteration steps do we roughly need to converge to machine precision? The answer is 52 as there are 52 bits in the mantissa.

Note that we didn't use the actual value of $f(\frac{a+b}{2})$, just its sign. We're throwing away information that we could have used. There are much better methods than the bisection method that make use of this information.

0.4 Linear interpolation method

The next best thing one can do is the linear interpolation method, also known as the double false position method.

This method works similarly to the bisection method by shrinking the interval $[a, b]$, but instead of always dividing it in half, this method makes does a better estimate. First, it calculates the values $f(a)$ and $f(b)$. Then it interpolates the function f between these two values with a linear function. We can easily calculate the root of the linear function, let's call it c. We then use c as the new middle point to create two intervals $[a, c]$ and $[c, b]$. From there on we proceed the same way as in the bisection method. We calculate $f(c)$ and decide which interval is the interesting one. This method is slighlty faster at converging as we make use of the function evaluation $f(c)$ at the point c.

0.5 Newton's method

Newton's method is another iterative way of finding a root of a function $f(x)$. Contrary to the other methods we have talked about before, we have to be able to evaluate the derivate of f as well as the function f itself at any point. This is not always possible, depending on how complicated the function is. If no analytic function can be written down for the derivate, one can try to calculate the derivate numerically (e.g. using finite differences). However, evaluating the derivatives can be very expensive. On the positive side, if it is easy to calculate the derivative, then Newton's method is very fast and converges very quickly.

Let's spend a bit of time discussing how to numerically calculate a derivative. A common method is called *finite difference* which approximate the derivative $f'(x)$ by the expression

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}.$$

Which spacing h to choose is difficult to answer in general. The smaller it is the closer the estimate to the real value of the derivative. However, the h has to remain finite and if it gets too small, we might run into floating point issues. This is an important example of why it it important to understand the limitations of floating point numbers. The term $f(x+h)$ and $f(x)$ are almost identical if h is small. Thus, calculating the difference will result in a large floating point error.

The expression above is a one sided finite difference. One can also make a central difference:

$$f'(x) \approx \frac{f(x+0.5h) - f(x-0.5h)}{h},$$

which has often better properties. But let's go back to our root finding method.

In Newton's method we need one starting point of x, a guess. Let's call this x_0. During every iteration, we improve upon our guess using the formula

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

We continue iterating until we have achieved a precision that is good enough for our purpose. Often, people look at the change from x_n to x_{n+1} as a measure of how close we are to the real root.

There are several problems with Newton's method. We already mentioned that the function needs to be differentiable on the entire interval. Furthermore, it turns out that Newton's method might sometimes not converge at all if the first derivate is not *well* behaved. In such a case we might overshoot and never converge. One such example is the function

$$f(x) = |x|^{1/4}.$$

Another problem are stationary points of the function f (i.e. the derivative is zero). In that case we divide by zero and the method breaks down.

One of the most important uses of Newton's method is in optimization. Optimization refers to minimization or maximization of functions. We already know one example, the least square fit.

0.6 Multiple roots

In many cases, functions have multiple roots. For example

$$f(x) = x^2 - 1$$

has roots at $x_0 = 1$ and $x_1 = -1$. How can we find multiple roots? All the above methods give us only one root. Well, the short answer is that in general we might not be able to find all the root of a function. Especially if you think of a function like $\sin(x)$ which has an infinite number of roots.

However, there is a trick that allows us to find at least multiple roots in simple functions. Suppose the function you are trying to find the root is $f(x)$ and you have already found a root x_0. Then, look at the function

$$h(x) = \frac{f(x)}{x - x_0}$$

and find its roots. The figure below shows the function $f(x) = x^2 - 1$ in red. Suppose our first attempt at root finding resulted in the value $x_0 = 1$. Then, the function h is $h(x) = f(x)/(x-1)$, which is plotted in blue. You can see that the function h has now only one root, namely the one we missed before at $x_1 = -1$.

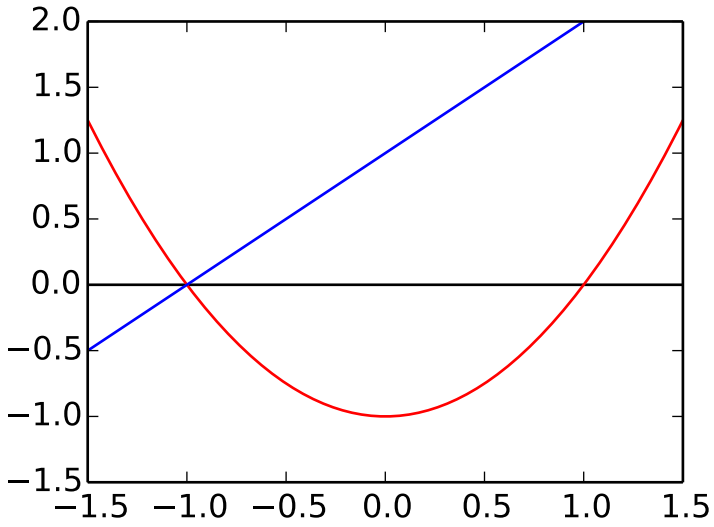

Figure 2: Finding multiple roots.