Lecture 10

Wednesday, January 29, 2020

The bisection method is based on the idea of

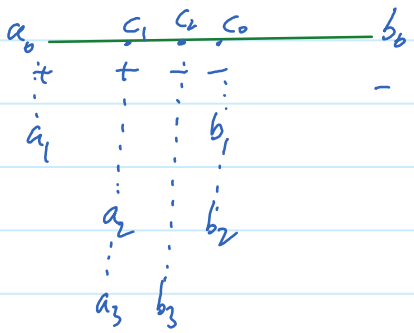
- binary search,
- Intermediate Value theorem (Bolzano, 1817).

Starting from an interval $[a_0, b_0]$ where $f(a_0)$ and $f(b_0)$ have different signs, we compute the midpoint $c_0 = \frac{a_0 + b_0}{2}$ and check the sign of $f(c_0)$.

If $f(c_0)$ and $f(a_0)$ have different signs then

$$a_1 = a_0, \quad b_1 = c_0.$$

Otherwise, $a_1 = c_0$ and $b_1 = b_0$.



By taking c_n as an approximate root, the error (difference between true root and approximate root) is estimated as follows:

$$|x^* - c_n| \leq \frac{1}{2^{n+1}} (b_0 - a_0).$$

If we want the error to be under some permitted error ε then we only need to choose n large enough such that

$$\frac{1}{2^{n+1}} (b_0 - a_0) < \varepsilon$$

This is equivalent to $$n > \log_2 \frac{b_0 - a_0}{\varepsilon} - 1.$$

Ex:

Find a root of $f(x) = x^3 - 2x - 2$ on the interval $[0, 2]$.

By taking $a_0 = 0$, $b_0 = 2$ we see that $f(a_0) = f(0) < 0$ and $f(b_0) = f(2) > 0$.

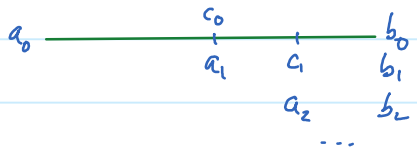
$$c_0 = \frac{a_0+b_0}{2} = \frac{0+2}{2} = 1, \quad f(c_0) = f(1) < 0$$

Thus, $a_1 = c_0 = 1$ and $b_1 = b_0 = 2$

Then $$c_1 = \frac{a_1+b_1}{2} = \frac{1+2}{2} = 1.5,$$

$$f(c_1) = f(1.5) = -1.6250 < 0.$$

Hence $a_2 = c_1 = 1.5$ and $b_2 = b_1 = 2$.



How close is c_4 to the true root x^*?

$$|x^* - c_4| \le \frac{1}{2^{4+1}}(b_0 - a_0) = \frac{1}{32}(2-0) = 0.0625.$$

How to compute x^* approximately with error less than 10^{-3}?
We want that $|x^* - c_n| < \varepsilon = 10^{-3}$. For this, we need

$$n > \log_2\left(\frac{b_0-a_0}{\varepsilon}\right) - 1 = \log_2 \frac{2}{10^{-3}} - 1 = \log_2(2000) - 1$$

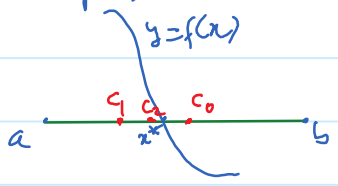
$$\approx 9.97$$

We can take $n = 10$. Then c_{10} approximates x^* with error less than 10^{-3}.
[See an example of writing Matlab code on the .m file on course website.]

* Observations:

- Bisection method guarantees success: the candidates $c_0, c_1, c_2, c_3, \ldots$ will converge to a root. Moreover, we know how many steps needed to be done to get an approximate root within some prescribed error.
- Bisection method is not sensitive to how close c_n is to the true root.

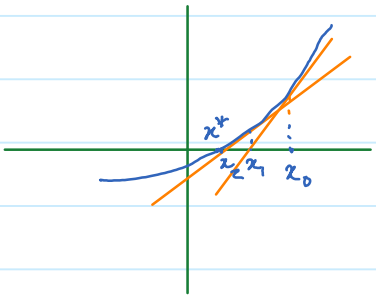
For example,



On the picture, c_0 is already close to x^*. But bisection method doesn't record that information. It continues to bisect the next interval, giving

q further from the true root x^* Because the bisection method only cares about dividing the interval, rather than making use of good qualities of function f, the convergence rate is rather slow.

Newton's method complements the bisection method: although convergence is not guaranteed, the rate of convergence is much faster. The idea of Newton's method is as follows:



Pick a starting point x_0 (preferably close to the true root)

Then draw a tangent line to the graph of f at the point $(x_0, f(x_0))$. The intersection between this line and the x-axis gives x_1. We then repeat the process, viewing x_1 as the starting point.

We see on the picture that the sequence $x_0, x_1, x_2, \ldots$ converges very quickly to the true root x^*. The order of finding this sequence is

$$x_0 \longrightarrow x_1 \longrightarrow x_2 \longrightarrow x_3 \longrightarrow \cdots$$

which means: once we pick x_0, we can find x_1. Then we can find x_2. Then we can find x_3, and so on.

The question is: given x_n, how to find x_{n+1}?

The tangent line of the graph of f at point $(x_n, f(x_n))$ has slope equal to $f'(x_n)$. The equation of this line is

$$y - f(x_n) = f'(x_n)(x - x_n).$$

To find the intercept with the x-axis, we set $y = 0$.

$$-f(x_n) = f'(x_n)(x - x_n).$$

The root x of this equation is x_{n+1}. Hence,

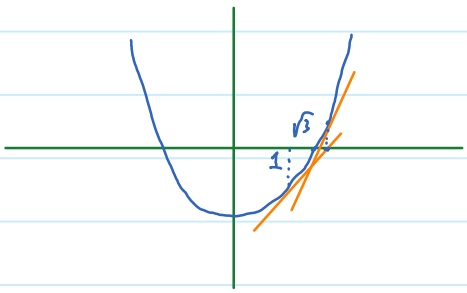
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

This is known as the recursive formula or iteration formula of the Newton's method.

Ex:

Find an approximate value of $\sqrt{3}$ by using Newton's method to find the positive root of $f(x) = x^2 - 3$.

We have $f'(x) = 2x$. Because $1 < \sqrt{3} < 2$, it is natural to choose the initial point $x_0 = 1$ (or 2).



The recursive formula of Newton's method is

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

$$= x_n - \frac{x_n^2 - 3}{2x_n}$$

$$= \frac{x_n}{2} + \frac{3}{2x_n}.$$

Thus, $x_{n+1} = \frac{x_n}{2} + \frac{3}{2x_n}$

Then $x_1 = \frac{x_0}{2} + \frac{3}{2x_0} = \frac{1}{2} + \frac{3}{2} = 2,$

$$x_2 = \frac{x_1}{2} + \frac{3}{2x_1} = \frac{2}{2} + \frac{3}{2\times 2} = 1.75,$$

$$x_3 = \frac{x_2}{2} + \frac{3}{2x_2} = \frac{1.75}{2} + \frac{3}{2\times 1.75} = 1.7321.$$

We see that we get quite a good approximation of $\sqrt{3}$ via only simple iterations.