The Bisection Method

Devere Anthony Weaver

Theorem (Intermediate Value Theorem)**.** *If $f \in C[a, b]$ and k is any number between $f(a)$ and $f(b)$, then there exists a number $c \in [a, b]$ for which $f(c) = k$.*

Bisection Technique

A basic problem in numerical approximation is the root-finding problem. This involved finding a rot of an equation of the form $f(x) = 0$ for some given function f. A root of this equation is also called a zero of the function.

The Bisection method (Binary-search) is a technique for finding roots that is based on the Intermediate Value Theorem mentioned above.

We suppose f is continuous on $[a, b]$, with $f(a)$ and $f(b)$ of the opposite sign. This is key for the the Bisection method won't necessarily work unless the end points of the given interval are of different signs. Then the Intermediate Value Theorem implies that a number p exists in (a, b) wit $f(p) = 0$. This procedure works when there is more than one root in the given interval; however, we assume a unique root exists in the interval for simplicity. The method repeatedly halves (bisects) subintervals of $[a, b]$ and locates the half containing p. This method belongs to a class of algorithms that attempt to find roots using iterative methods.

To begin, set $a_1 = a$, $b_1 = b$ and $p_1 = \frac{a_1+b_1}{2}$ (i.e. p_1 is the midpoint of the interval). Then, evaluate the function $f(p_1)$. If $f(p_1) = 0$, then the point $p = p_1$ and the root has been found. If $f(p_1) \neq 0$, then evaluate $f(a_1)$. There are two possible outcomes,

- If $f(p_1)f(a_1) > 0$, then the root p lies to the right of p_1 (i.e. $p \in (p_1, b_1)$). We then shift the interval to the right of p_1 by setting $a_2 = p_1$ and $b_2 = b_1$. This gives the new interval $[a_2, b_2] = [p_1, b_1]$.
- If $f(p_1)f(a_1) < 0$, then the root p lies to the left of p_1 (i.e. $p \in (a_1, p_1)$). We then shift the interval to the left of p_1 by setting $a_2 = a_1$ and $b_2 = p_1$. This gives the new interval $[a_2, b_2] = [a_1, p_1]$.

This method is then applied to the new subinterval $[a_2, b_2]$ and iteratively to each subsequent subinterval until a good approximation for the root p is found. A *good* approximation for the root is determined by the implementer after a specific number of iterations, a specific error tolerance is reached, or some other metric.

To see an example of the Bisection method implemented using a Python notebook, follow the link: The Bisection Method

---

Theorem. *Suppose that $f \in C[a,b]$ and $f(a)f(b) < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating to zero p of f with*

$$|p_n - p| \leq \frac{b-a}{2^n}, \text{ when } n \geq 1.$$

This theorem can be used to approximate the number of iterations needed to achieve a specified error. As an example, use this theorem to determine the number of iterations needed to solve $f(x) = x^3 + 4x^2 - 10 = 0$ with accuracy 10^{-3} using $a_1 = 1, b_1 = 2$. Then,

$$\begin{aligned}
\Rightarrow |p_n - p| \leq \frac{b-a}{2^n} &= 2^{-n}(b-a) < 10^{-3} \\
&= 2^{-n} < 10^{-3} \\
&= -n \log 2 < \log 10^{-3} \\
&= -n \log 2 < -3 \\
&= -n < \frac{-3}{\log 2} \\
&= n > -\frac{-3}{\log 2} \approx 9.96\ldots.
\end{aligned}$$

Hence, roughly 10 iterations will be needed to achieve an approximation for the root of f that is accurate within 10^{-3}.

It should be noted that error analysis gives only a bound for the number of iterations, in some cases the bound is much larger than the actual number required. As a final note, the computation of a the midpoint of an interval is slightly different when it is implemented on a digital computer since we need to account for round-off error. To compute the midpoint on a computer for $[a_n, b_n]$, instead use

$$p_n = a_n + \frac{b_n - a_b}{2}.$$

Here, when $b_n - a_n$ is near max precision for a machine, the correction may be in error, but the error doesn't significantly affect the value of p_n. Using the other form to compute a midpoint, it is entirely possible to return a point that does not exist in the interval since round off is not accounted for.