# Lesson: Finding Roots: Bisection Method

## Chapter 3: Finding Roots of $f(x) = 0$.

### What is a root?

If f(x) is a function and $f(x^*) = 0$ then $x^*$ is called a **root** of f(x). A root is sometimes referred to as a "zero" of the function since we are trying to find the place where a function is equal to zero. We typically denote the zero or root of a function by $x^*$.

We can also formulate the problem of find the intersection of two functions as a root finding problem: $f(x) = g(x) \Leftrightarrow (f - g)(x) = 0$

### Issues

a. Does the problem have a solution? (Existence)
b. If there is a solution, are we guaranteed that our algorithm will find it? (Convergence)
c. How efficient is our algorithm? How rapidly will it approximate the solution within a given error tolerance?
d. What if there is more than one solution? (Uniqueness) How do we get it to choose the root that we want?

### Methods

Root finding methods are **iterative** methods: we repeat a process over and over until we get an answer. However, we need a starting value or interval and we need to know when an answer is close enough to the correct answer.

There are two basic types of methods:
1. Bracketing Methods--Bisection
2. Fixed Point Methods--Newton, Secant

### Visually finding roots.

Perhaps the easiest way to find a root is by looking at a plot. If you have a plot of a function then you can point your finger at it and see where the function crosses the x-axis. Wherever the function crosses the x-axis, we have a root.

In Matlab, it is relatively easy to plot a function using the **plot** command. You can then click on the zoom button to zoom in on the plot. This is somewhat <u>imprecise</u> though. The great advantage of visually estimating a root, however, is that it gives you a way of bracketing the root within a particular interval. This estimate of the root can then be used in one of the other methods discussed below.

*Example:* Here we find the root of the function $y = e^{-x} + x - 2$.

```
>> x=[0:0.1:6];         % pick a region to look for a root
>> y=exp(-x)+x-2;
>> plot(x,y)            % plot the function
>> grid                 % this puts a grid on the plot so that you can see a root
                        % better.
```

It looks like from the plot that there is a root close to 2. Zooming in by using mouse clicks, the root appear to be about 1.84.

**Bisection Method**

Finding roots by bisection is very similar to finding roots visually. The difference is that there is a systematic approach to it instead of using an arbitrary sequence of mouse clicks.

1. You have a function. You need an initial estimate of where you think the root is. In particular, you need to choose an interval in which you think the root is lying.

   - Choose an interval [a,b].

2. Check to make sure that one of the endpoints isn't the root.

   - If |f(a)| < ftol OR |f(b)| < ftol, STOP .

   Also, stop if the width of the interval bracketing the root comes within some tolerance, xtol, then STOP

   - $|a - b| < xtol$

3. Verfiy that a root does in fact lie between a and b. How can you tell? f(a) and f(b) must have opposite signs. If one is positive, and the other negative, the function must have crossed zero somewhere in between. (Intermediate Value Theorem) (Remember, we're dealing with a continuous function here.)

   - If sign(f(a)) = sign (f(b)) [NO sign change], STOP.

4. Otherwise, cut the interval in half. Let c be the midpoint of [a,b]

   - $c = \frac{a+b}{2}$

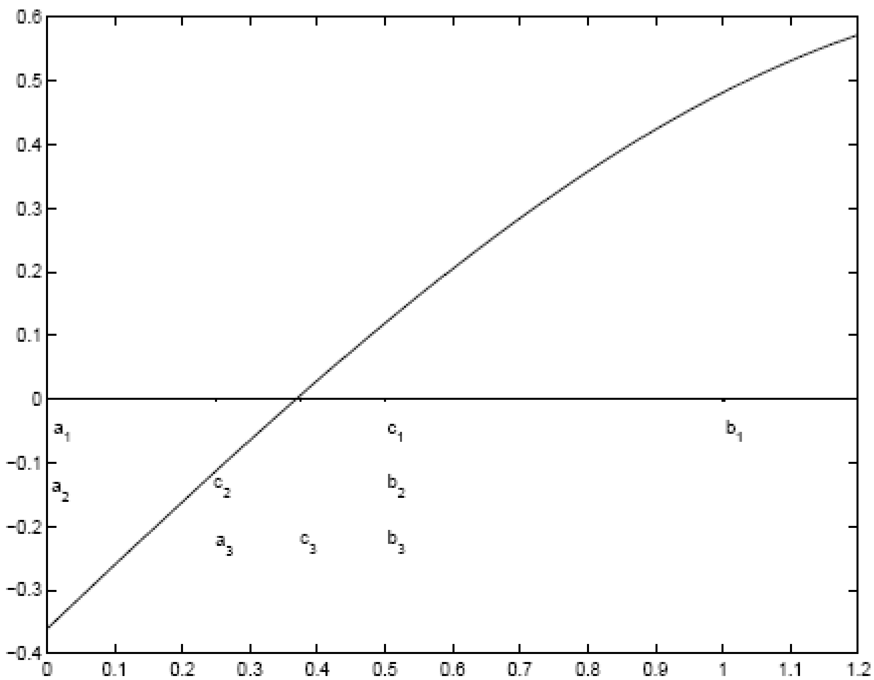
5. There are three possibilities now:
   a. c is the root,

b. the root is on the left side of c, in the interval [a,c],
c. the root is on the right side of c, in the interval [c,b].

- If |f(c)| < ftol, STOP.

- elseif sign(f(a)) ~= sign(f(c)) < 0, the root is on the left. Keep a, but let b = c. Go back to step (3).

- Else, the root is on the right. Keep b, but let a=c. Go back to step (3).




**Advantages/Disadvantages**

+ What if there's no root? This one's covered. If there's no root in the given interval, we'll know about it immediately -- there won't be a sign change.

-/+ More than one root? While no roots implies no sign change, the converse isn't true -- no sign change doesn't have to mean no roots. A function with two zeros in a given interval will not have a change in sign at the endpoints. You need to choose a starting interval which brackets only the root you want (which means you need a rough idea of where it is to start with)

- Roots of (even) multiplicity greater than one. Think of $f(x) = (x-1)^2$ . The graph touches the x-axis at x = 1, but it also turns on that point -- you never get a sign change, no matter what interval you start with. Bisection will never find that root.

+ One and only one root? Convergence to that root is guaranteed. Doesn't even matter how big that starting interval is; if there's only one root between a and b, we'll get there eventually.

- How fast does it get there? Not very. Bisection is the slowest method of all the methods we'll consider.

**Note:** **xtol** is used to test the magnitude by which the estimate of the root changes from iteration to iteration. **ftol** is to test the magnitude of the function f(x) at the current estimate of the root.

***Bisection Algorithm:***

```
Inputs: f, a, b, xtol, ftol   % f is a function m-file that defines the function f.
Output: c                     % the root of f(x)

fa = f(a); fb = f(b);

if |fa| < ftol
    c = a
    return
elseif |fb| < ftol
    c = b
    return
elseif |a − b| < xtol
    c = a
    return
end

while |a − b| > xtol  and  |f(c)| > ftol
    if sign(fa) = sign(fb)        % f(a) and f(b) have same sign
            %  Give a warning
            return
    else
            c = (a + b) / 2
            fc = f(c)
    end

    if sign(fa) ~= sign(fc)       % choose left half
            b = c
            fb = fc
    elseif sign(fc) ~= sign(fb) % choose right half
            a = c
            fa = fc
    end
end
```