LECTURE 5

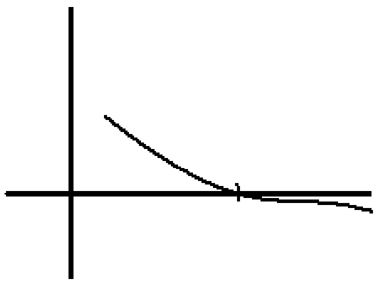
Solution of Nonlinear Equations

1. Introduction

In the next series of lectures we shall discuss the problem of identifying the **roots** of equations and systems of equations..

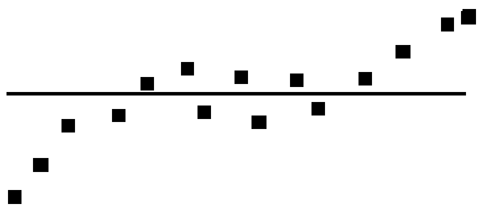
The basic formulation of the problem in the simplest case is this: given a function $f(x)$ of one variable, find all x such that $f(x) = 0$.

All the algorithms we shall present are based on the following geometric idea. Suppose a continuous function $f(x)$ has a zero on the interval $[a, b]$. Then its graph must look something like



Well, not exactly like this, but we do know that the graphs has to cross the x-axis somewhere between the points $x = a$ and $x = b$. A graphical solution to the problem would then just amount to identifying at which point(s) the graph of $f(x)$ crossses the x-axis.

Unfortunately, this graphical method is not really practical for finding highly accurate solutions. The reason for this is that once you start focussing in on the behavior of $f(x)$ when $f(x)$ is extremely close to zero, is that floating point errors become more and more significant so that near a zero of $f(x)$ a point plot may look rather erratic.



2. Bisection Algorithm

This method for finding roots is based on the following fundamental theorem from Calculus.

THEOREM 5.1. ***(Intermediate Value Theorem).*** *If f is a continuous function on the interval $[a, b]$, and then if ξ is a number between $f(a)$ and $f(b)$, then there exists a point $c \in [a, b]$ such that $f(c) = \xi$.*

This theorem is more than an mathematical abstraction of the graphical method mentioned above. Certainly, it says that if 0 is a number between $f(a)$ and $f(b)$, then there has to be a point c between a and b such that $f(c) = 0$. But it also furnishes us with an algorithm for finding c.

Suppose

$$f(a) < 0 < f(b).$$

Then let c be the point half-way between a and b

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

If we evaluate $f(c)$ then either

- $f(c) = 0$, in which case we have found a zero of f
- $f(c) < 0$. In this case, the Intermediate Value Theorem tells us that there must be a zero between c and b.
- $f(c) > 0$. In this case, the Intermediate Value Theorem tells us that there must be a zero between a and c.

If the first case holds, then we are finished. If the second case holds, then we can set $a_1 = c$ and $b_1 = b$ and

$$c_1 = \frac{a_1 + b_1}{2}$$

and fix the position of the zero to lie either at c_1, or between a_1 and c_1, or between c_1 and b_1. Similarly, if the third case holds, we can fix the position of the zero between either c and $(b+c)/2$, or between $(b+c)/2$ and c.

EXAMPLE 5.2. Starting with $a = 0$ and $b = 2$, use the bisection algorithm described above to find a zero of

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

- Now

$$\begin{aligned} f(a) &= f(0) = -2 \\ f(b) &= f(2) = 2 \end{aligned}$$

  So we must have a zero of f somewhere between $x = 0$ and $x = 2$. If we evaluate f at the midpoint $c = 1$ between 0 and 2, we find

$$f(1) = -1 < 0$$

  So the zero must lie between $x = 1$ and $x = 2$. The midpoint of $(1, 2)$ is 1.5, and there

$$f(1.5) = 0.25 > 0$$

  Therefore the zero has to line within the interval $(1, 1.5)$. Calculating f at the midpoint of that interval yields

$$f(1.25) = -0.4375 < 0$$

  so the zero must lie within $(1.25, 1.50)$. Calculating f at the midpoint of that interval yields

$$f(1.375) = -0.109375 < 0$$

So now we know the zero lies within $(1.375, 1.50)$. Iterating once more we find

$$f\left(\frac{1.375+1.50}{2}\right) = f(1.4375) = 0.066406250$$

so the zero lies within $(1.375, 1.4375)$.

Of course, the correct answer to this example is $\sqrt{2} = 1.41421356\ldots$.

3. Coding the Bisection Algorithm

Let us now figure out how to tell a computer to carry out the bisection algorithm.

We'll begin by first working out the logic of the algorithm.

1. We start with a function f and two numbers a and b such that $f(a) < 0 < f(b)$.
2. We calculate the midpoint between a and b

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

   and

   $$f(c)$$

3. We now have three possibilitites
   (a) $f(c) = 0$. In this case we terminate the program since we've found a zeron.
   (b) $f(c) < 0$. In this case, the zero mus lie between c and b, and so we loop back to step 2, with the original values of a and b now being replaced, respectively, by c and b.
   (c) $f(c) > 0$. In this case, the zero mus lie between a and c, and so we loop back to step 2, with the original values of a and b now being replaced, respectively, by a and c.

In Maple, the problem in the example above would code as follows:

```
f := x -> x^2 -2;
a := 0.0;
b := 2.0;
c := (a+b)/2;
while (f(c) <> 0) do
     if (f(c) < 0) then
          a := c;
     else
          b := c;
     fi;
od;
c;
```

Now if you try to run this code you'll see find that Maple has gotten stuck in an infinite loop. The reason for this, is that the condition we looking for, $f(c) = 0$, is never actually going to be met. In order to make this algorithm palatable to Maple we are going to have to tell Maple when our answer is good enough.

Now there's actually three natural ways to control the loop.

First, we can instruct Maple to cease calculating after a certain number of iterations, say 1000. To implement this idea we would change the line

while (f(c) <>0) do

to

for i to 1000 do

Secondly, we can allow Maple to stop once $f(c)$ is within a certain distance of 0, say within 0.00001. This can be implemented by inserting a statement like

if |f(c)| < 0.00001 then break;

within the do loop

Finally, we can allow Maple to stop once c is determined to lie within a small enough interval. This can be implemented via a statement like

if |a-b| < 0.00001 then break;

Thus, a fully functioning Maple routine might be

```
f := x -> x^2 -2;
a := 0.0;
b := 2.0;
c := (a+b)/2;
for i from 1 to 1000 do
     if (f(c) < 0) then
          a := c;
     else
          b := c:
     fi;
     c := (a+b)/2:
     if |f(c)| < 0.00001 then break fi:
     if |a-b| < 0.00001 then break fi:
od:
c;
```