Maple Lab 5

Many practical problems are equivalent to solving an equation of the form $f(x) = 0$. These include the problem of finding the x-intercepts of a graph, the problem of finding mininum and maximum of a function, the problem of finding roots of a polynomial,... Some simple equations can be solved analytically—that is, there is an explicit and precise formula for the solution—such as the linear equations $ax + b = 0$ and the quadratic equations $ax^2 + bx + c = 0$. Most equations cannot be solved analytically, but can be solved numerically (approximately). Two numerical methods we have learned in Calculus I are the bisection method and the Newton's method. In this lab, you will learn how to use Maple to solve numerically the equation $f(x) = 0$ using

- bisection method,
- Newton's method.

1 Practice

Let us solve the equation $\sin x = \frac{1}{x}$. There are many ways to convert this equation to the form $f(x) = 0$ such as $\sin x - \frac{1}{x} = 0$ or $x - \frac{1}{\sin x} = 0$ or $x \sin x - 1 = 0$, etc. Each of these form corresponds to a different function f. Let us choose $f(x) = x \sin x - 1$ because it is a nice function (continuous and differentiable everywhere).

Bisection method

Recall the Intermediate Value Theorem:

> *Let f be a continuous function on an interval $[a, b]$. Suppose $f(a)$ and $f(b)$ have different signs. Then there exists $c \in (a, b)$ such that $f(c) = 0$.*

The number c in the theorem is a *root* of the function $f(x)$. Although the theorem does not tell us what the root is, it does tell us where to find it. For example, you can check with calculator that $f(0) < 0$ and $f(2) > 0$. The Intermediate Value Theorem guarantees that $f(x)$ has a root in the interval $(0, 2)$. The graph of the function (Figure 1) confirms that fact.

```
f:=x->x*sin(x)-1;
plot(f(x),x=-10,10);
```

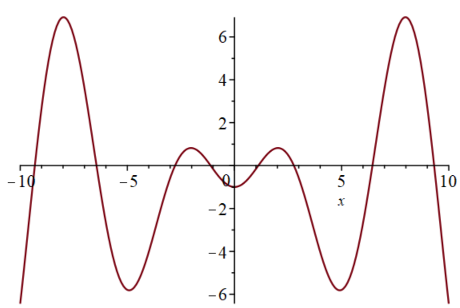


Figure 1

The bisection method goes as follows. The first step is setting $a = 0$, $b = 2$, and $d = (a + b)/2$. If $f(a)$ and $f(d)$ have different signs then we look for a root in the interval $[a, d]$. If $f(a)$ and $f(d)$ have the same sign then $f(b)$ and $f(d)$ must have different signs and we should look for a root in the interval $[d,b]$. After the first step, the interval to search for the root is cut down by half in length. The new interval is either $[a,d]$ or $[d,b]$. Viewing this interval as if it were the original interval $[a,b]$, we repeat the above procedure. The interval is cut down by 4 times and 8 times after the second step and third step, respectively. Of course, the iteration can go on forever. To implement the procedure on the computer, you need to specify a *terminating condition*. You may want the iteration to stop

- after a certain number of steps, or
- when certain precision is reached (the approximated root is within an allowable error from the exact root).

If you want to terminate after n steps, the algorithm is as follows:

```
1) i = 1
2) d = (a+b)/2
3) Check i < n.
If false then stop the procedure.
If true then check the sign of f(a)f(d).
        * If f(a)f(d) < 0 then the interval [a,b] is reset by [a,d].
        Otherwise, the interval [a,b] is reset by [d,b].
        * Update d = (a+b)/2
        * i = i + 1
        * Go back to check i < n at the beginning of Step 3.
```

In Maple, we first declare the variables:

```
a:=0.0; b:=2.0; d:=(a+b)/2; n:=10;
```

and then use the "for" loop (press **Shift+Enter** after each line and **Enter** after the last line):

```
for i from 1 to n do
    if f(a)*f(d) < 0 then
       b := d;
    else
       a := d;
    end if;
    d := (a + b)/2;
end do:
```

The colon after "end do" is used to suppress the intermediate outputs. This procedure consists of $n = 10$ iterations. The reason why we type `a:=0.0` instead of `a:=0` is so that Maple understands a as a decimal-point number (i.e. of "float" data type), not an integer. As we keep updating the values of a and b in the for loop, they become decimal-point numbers, not integers. If you only write `a:=0`, Maple may throw an error when it later tries to assign a by a decimal-point number. The width of the interval $[a,b]$ is now equal to the original length, which was $2 - 0 = 2$, divided by 2^{10}. Now type

```
[a,b];
d;
```

to see the latest interval and its mid-point.

If you want to terminate when the error between $(a+b)/2$ and the exact root is less than an allowable error epsilon, the algorithm is as follows:

```
1) d = (a+b)/2
2) While b-a is still greater than epsilon, do the following:
        * If f(a)f(d) < 0 then the interval [a,b] is reset by [a,d].
        Otherwise, the interval [a,b] is reset by [d,b].
        * Update d = (a+b)/2
        * Go back to check b-a at the beginning of Step 2.
```

In Maple, we first declare the variables:

```
a := 0.0; b := 2.0; d:= (a+b)/2; epsilon := 0.0001;
```

and then use the "while" loop (press **Shift+Enter** after each line and **Enter** after the last line):

```
while b-a>epsilon
do
   if f(a)*f(d) < 0 then
      b := d;
   else
      a := d;
   end if;
   d := (a + b)/2;
end do:
```

Now type

```
[a,b];
d;
```

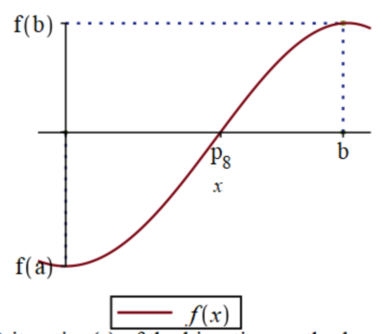
to see the latest interval and its mid-point. The midpoint d is the approximate root that we were looking for.

To see an animation of the bisection method, we first call the Numerical Analysis package:

```
with(Student[NumericalAnalysis]);
```

and then enter the command

```
Bisection(f(x), x = [0.0, 2.0], output = animation, tolerance = 0.0001,
stoppingcriterion = absolute);
```



8 iteration(s) of the bisection method applied to $f(x) = x\sin(x) - 1$ with initial points $a = 0.$ and $b = 2.0.$

Figure 2

Click on the picture. You will see on the Menu bar the Play button and a box called FPS (Frame Per Second). You can set FPS to 1 or 2 (then press Enter) to get a slow animation. Then press the Play button. It will animate the iteration for you. For more animation options for the bisection method, see this documentation page.

Newton's method

For Newton's method, the root is the limit of the sequence x_n defined recursively as follows:

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

Given the value of x_0, you can find the first n terms x_1,x_2,...,x_n of the sequence by the algorithm:

```
1) set a value for x0
2) set x_prev = x0
3) for i = 1 to n, do the following
        x = x_prev - f(x_prev)/f'(x_prev)
        print the value of x
        set x_prev = x
```

To animate this iteration, we use the command (with $x_0 = 0.1$):

```
Newton(f(x), x = 0.1, output = animation, tolerance = 0.0001,
stoppingcriterion = absolute);
```

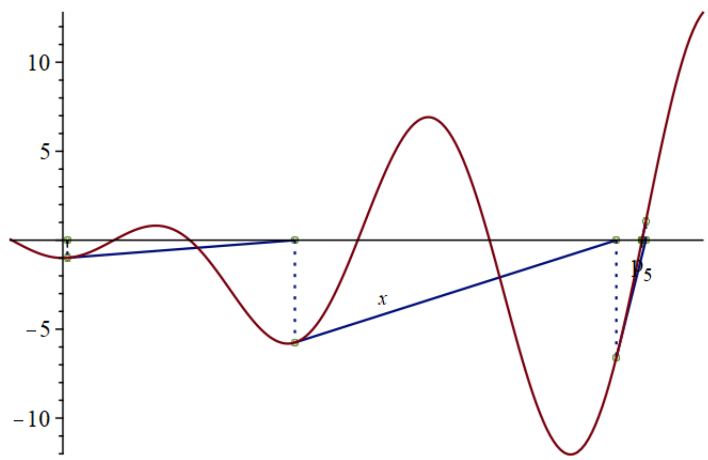


Figure 3

For more animation options for the Newton's method, see this documentation page.

2 To turn in

Consider the function $f(x) = x^4 - 4x^3 - 2x^2 + 3x + 1$.

1. Graph the function. How many roots does it have?

2. Write a "while" loop to approximate the *second largest* root of $f(x)$ using the bisection method. The allowable error is 0.0001. Show animation.

3. Write a "for" loop to find x_8 using the Newton's method. Test with two different values of x_0, namely $x_0 = 0.2$ and $x_0 = 0.4$. Show animation for each case.

4. What value of x_0 should you take so that x_8 in the Newton's method approximates the second smallest root of $f(x)$?