Chapter 2

Root Approximations

2.1 Newton's Method for Root Approximation

Newton's (or Newton–Raphson) method can be used to approximate the roots of any linear or non–linear equation of any degree. This is an iterative (repetitive procedure) method and it is derived with the aid of Figure 2.1.

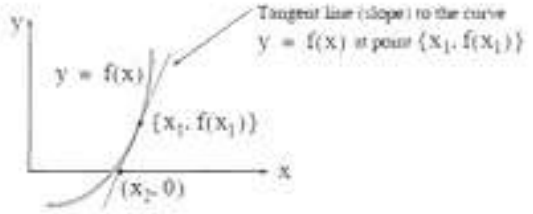


Figure 2.1. Newton's method for approximating real roots of a function

We assume that the slope is neither zero nor infinite. Then, the slope (first derivative) at $x = x_1$ is

$$f'(x_1) = \frac{y - f(x_1)}{x - x_1}$$

$$y - f(x_1) = f'(x_1)(x - x_1) \qquad (2.1)$$

The slope crosses the x – axis at $x = x_2$ and $y = 0$. Since this point $[x_2, f(x_2)] = (x_2, 0)$ lies on the slope line, it satisfies (2.1). By substitution,

$$0 - f(x_1) = f'(x_1)(x_2 - x_1)$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \qquad (2.2)$$

and in general

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad 2.3$$

2.2 Approximations with Spreadsheets

In this section, we will go through several examples to illustrate the procedure of using a spreadsheet such as Excel* to approximate the real roots of linear and non–linear equations.

We recall that there is a standard procedure for finding the roots of a cubic equation; it is included here for convenience.

A cubic equation of the form

$y^3 + py^2 + qy + r = 0$
can be reduced to the simpler form
$x^3 + ax + b = 0$
where

$$x = y + \frac{p}{3} \qquad a = \frac{1}{3}(3q - p^2) \qquad b = \frac{1}{27}(2p^3 - 9pq + 27r)$$

For the solution it is convenient to let

$$A = \sqrt[3]{\frac{-b}{2} + \sqrt{\frac{b^2}{4} + \frac{a^3}{27}}} \qquad B = -\sqrt[3]{\frac{-b}{2} + \sqrt{\frac{b^2}{4} + \frac{a^3}{27}}}$$

Then, the values of x for which the cubic equation of (2.11) is equal to zero are

$$x_1 = A + B \qquad x_2 = -\frac{A+B}{2} + \frac{A-B}{2}\sqrt{-3} \qquad x_3 = -\frac{A+B}{2} - \frac{A-B}{2}\sqrt{-3}$$

If the coefficients p, q, and r are real, then

If $\frac{b^2}{4} + \frac{a^3}{27} > 0$ *one root will be real and the other two complex conjugates*

If $\frac{b^2}{4} + \frac{a^3}{27} < 0$ *the roots will be real and unequal*

If $\frac{b^2}{4} + \frac{a^3}{27} = 0$ *there will be three real roots with at least two equal*

While MATLAB handles complex numbers very well, spreadsheets do not. Therefore, unless we know that the roots are all real, we should not use a spreadsheet to find the roots of a cubic equation by substitution in the above formulas. However, we can use a spreadsheet to find the real root since in any cubic equation there is at least one real root. For real roots, we can use a spreadsheet to define a range of x alues with small increments and compute the corresponding values of y = f(x) Then, we can plot y versus x to observe the values of x that make

f(x) = 0 . This procedure is illustrated with the examples that follow.

Note: In our subsequent discussion we will omit the word *cell* and the key <*enter*>. Thus B3, C11, and so on will be understood to be cell B3, cell C11, and so on. Also, after an entry has been made, it will be understood that the <*enter*> key was pressed.

Example 2.3

Compute the roots of the polynomial

$y = f(x) = x^3 - 7x^2 + 16x - 12$

using Excel

Solution:

We start with a blank worksheet. In an Excel worksheet, a *selected* cell is surrounded by a heavy border. We select a cell by moving the thick hollow white cross pointer to the desired cell and we *click*. For this example, we first select A1 and we type **x**. We observe that after pressing the <*enter*> key, the next cell moves downwards to A2; this becomes the next selected cell. We type 0.00 in A2. We observe that this value is displayed just as 0, that is, without decimals. Next, we type 0.05 in A3. We observe that this number is displayed exactly as it was typed.

We will enter more values in column A, and to make all values look uniform, we *click* on letter ***A***
on top of column A. We observe that the entire column is now highlighted, that is, the background on the monitor has changed from white to black. Next, from the *Tools* drop menu of the Menu bar, we choose *Options* and we click on the *Edit* tab. We *click* on the *Fixed Decimal* check box to place a check mark and we choose 2 as the number of decimal places. We repeat these steps for Column B and we choose 3 decimal places. Then, all numbers that we will type in Column A will be fixed numbers with two decimal places, and the numbers in Column B will be fixed with three decimal places.

To continue, we select A2, we *click* and holding the mouse left button down, we drag the mouse
down to A3 so that both these two cells are highlighted; then we release the mouse button.

When properly done, A2 will have a white background but A3 will have a black background. We will now use the *AutoFill** feature to fill in the other values of in Column A. We will use values in 0.05 increments up to 5.00. Column *A* now contains 100 values of from 0.00 to 5.00 in increments of 0.05.

Next, we select B1, and we type **f(x)**. In B2, we type the equation formula with the = sign in front of it, that is, we type = A2^3-7*A2^2 + 16*A2-2
where A2 represents the first value of x = 0.00
We observe that B2 displays the value –12.000 This is the value of f (x) when x = 0.00 Next, we want to copy this formula to the range
B3:B102 (the colon : means B3 through B102). With B2 still selected, we *click* on *Edit* on the main taskbar, and we *click* on *Copy*. We select the range B3:B102 with the mouse, we release the mouse button, and we observe that this range is now highlighted. We click on *Edit*, then on *Paste* and we observe that this range is now filled with the values of f (x) Alternately, we can use the *Copy* and *Paste* icons of the taskbar.
To plot f (x) versus , we click on the *Chart Wizard* icon of the Standard Toolbar, and on the *Chart type* column we click on XY (Scatter). From the displayed charts, we choose the one on top of the right side (the smooth curves without connection points). Then, we *click* on *Next*, *Next*, *Next,* and *Finish*. A chart similar to the one on Figure 2.4 appears.

x	f(x)
0.00	-12.000
0.05	-11.217
0.10	-10.469
0.15	-9.754
0.20	-9.072
0.25	-8.422
0.30	-7.803
0.35	-7.215
0.40	-6.656
0.45	-6.126
0.50	-5.625
0.55	-5.151

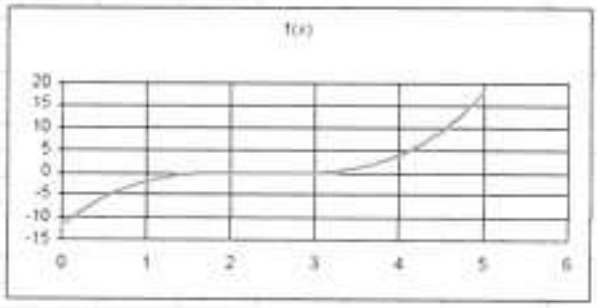


Figure 2.4. Plot of the equation of Example 2.3.

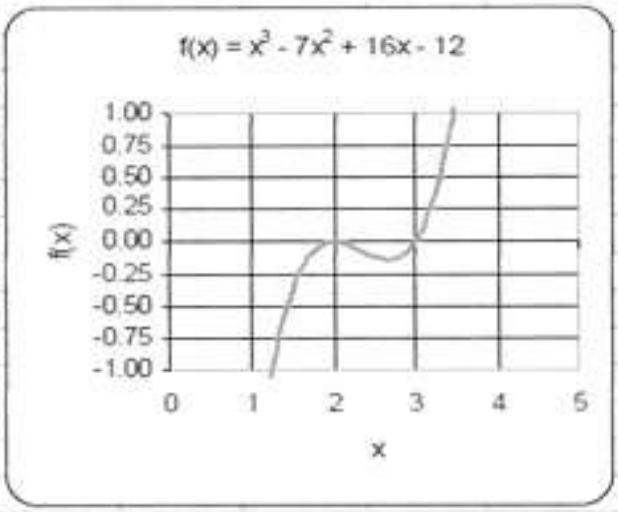
We will modify this plot to make it more presentable, and to see more precisely the x – axis crossing(s), that is, the roots of f(x). This is done with the following steps:

1. We *click* on the *Series 1* box to select it, and we delete it by pressing the *Delete* key.
2. We *click* anywhere inside the graph box. Then, we see it enclosed in six black square handles. From the *View* menu, we *click* on Toolbars, and we place a check mark on *Chart*. The *Chart* menu appears in two places, on the main taskbar and below it in a box where next to it is another small box with the hand icon. *Note:* The *Chart* menu appears on the main taskbar and on the box below it, only when the graph box is selected, that is, when it is enclosed in black square handles. From the *Chart* menu box (below the main taskbar), we select *Value (X) axis*, and we *click* on the small box next to it (the box with the hand icon). Then, on the *Format axis* menu, we *click* on the *Scale* tab and we make the following entries:

Minimum: 0.0
Maximum: 5.0
Major unit: 1.0
Minor unit: 0.5
We *click* on the *Number* tab, we select *Number* from the *Category* column, and we type 0 in the *Decimal places* box. We *click* on the *Font* tab, we select any font, *Regular style, Size* 9. We *click* on the *Patterns* tab to select it, and we *click* on *Low* on the *Tick* mark labels (lower right box). We *click* on OK to return to the graph.
3. From the *Chart* menu box we select *Value (Y) axis* and we *click* on the small box next to it (the box with the hand icon). On the *Format* axis menu, we *click* on the *Scale* tab, and we make the
following entries:
Minimum: □1.0
Maximum: 1.0
Major unit: 0.25
Minor unit: 0.05
We *click* on the *Number* tab, we select *Number* from the *Category* column, and we select 2 in the *Decimal places* box. We *click* on the *Font* tab, select any font, *Regular style, Size 9*. We *click* on the *Patterns* tab, and we *click* on *Outside* on the *Major tick* mark type (upper right box). We *click* on *OK* to return to the graph.
4. We *click* on *Chart* on the main taskbar, and on the *Chart Options*. We *click* on *Gridlines,* we place check marks on *Major gridlines* of both *Value (X) axis* and *Value (Y) axis*. Then, we *click* on the *Titles* tab and we make the following entries:
Chart title: f(x) = the given equation (or whatever we wish)
Value (X) axis: x (or whatever we wish) *Value (Y) axis:* y=f(x) (or whatever we wish)
5. Now, we will change the background of the plot area from gray to white. From the *Chart* menu box below the main task bar, we select *Plot Area* and we observe that the gray background of the plot area is surrounded by black square handles. We *click* on the box next to it (the box with the hand

icon), and on the *Area* side of the *Patterns* tab, we *click* on the white square which is immediately below the gray box. The plot area on the chart now appears on white background.
6. To make the line of the curve f(x) thicker, we *click* at any point near it and we observe that several black square handles appear along the curve. *Series 1* appears on the *Chart* menu box. We *click* on the small box next to it, and on the *Patterns* tab. From the *Weight* selections we select the first of the thick lines. Finally, to change *Chart Area* square corners to round, we select *Chart Area* from the *Chart* menu, and on the *Patterns* tab we place a check mark on the *Round corners* box. The plot now resembles the one shown in Figure 2.5 where we have shown partial lists of x and f (x) The given polynomial has two roots at $x = 2$ and the third root is $x = 3$
We will follow the same procedure for generating the graphs of the other examples which follow; therefore, it is highly recommended that this file is saved with any name, say *poly01.xls* where.*xls* is the default extension for file names saved in Excel.

x	f(x)
0.00	-12.000
0.05	-11.217
0.10	-10.469
0.15	-9.754
0.20	-9.072
0.25	-8.422
0.30	-7.803
0.35	-7.215
0.40	-6.656
0.45	-6.126
0.50	-5.625
0.55	-5.151
0.60	-4.704
0.65	-4.283
0.70	-3.887
0.75	-3.516
0.80	-3.168
0.85	-2.843
0.90	-2.541
0.95	-2.260
1.00	-2.000



	x	f(x)	x	f(x)
	1.90	-0.011	2.90	-0.081
	1.95	-0.003	2.95	-0.045
Roots →	2.00	0.000 →	3.00	0.000
	2.05	-0.002	3.05	0.055
	2.10	-0.009	3.10	0.121

f(x) =0 at x=2 (double root) and at x=3

Figure 2.5. Modified plot of the equation of Example 2.3.

2.3 The Bisection Method for Root Approximation

The *Bisection* (or *interval halving*) method is an algorithm* for locating the real roots of a function.

The objective is to find two values of x, say x_1 and x_2 so that $f(x_1)$ and $f(x_2)$ have opposite signs, that is, either $f(x_1) > 0$ and $f(x_2) < 0$, or $f(x_1) < 0$ and $f(x_2) > 0$ If any of these two conditions is satisfied, we can compute the midpoint x_m of the interval $x_1 \leq x \leq x_2$ with

$$Xm = \frac{x1+x2}{2}$$

Knowing x_m we can find $f(x_m)$ Then, the following decisions are made:

1. If $f(x_m)$ and $f(x_1)$ have the same sign, their product will be positive, that is $f(x_m) \cdot f(x_1) > 0$ This indicates that x_m and x_1 are on the left side of the *x-axis* crossing as shown in Figure 2.11 In this case, we replace x_1 with x_m .

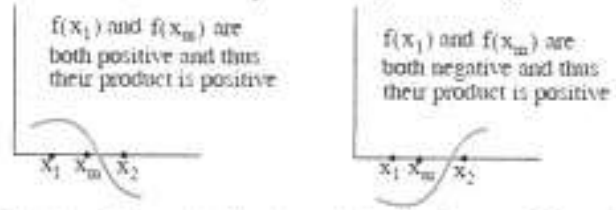


Figure 2.11. Sketches to illustrate the bisection method when $f(x_1)$ and $f(x_m)$ have same sign

2. If $f(x_m)$ and $f(x_1)$ have opposite signs, their product will be negative, that is, $f(x_m) \cdot f(x_1) < 0$. This indicates that x_m and x_2 are on the right side of the x-axis crossing as in Figure 2.12. In this case, we replace x_2 with x_m.

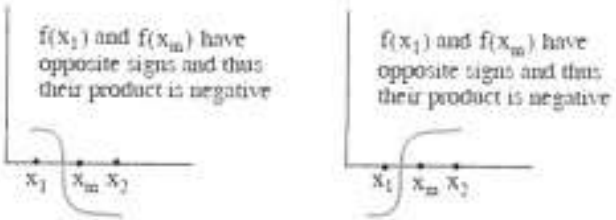


Figure 2.12. Sketches to illustrate the bisection method when $f(x_1)$ and $f(x_m)$ have opposite signs

After making the appropriate substitution, the above process is repeated until the root we are seeking has a specified tolerance. To terminate the iterations, we either:

a. specify a number of iterations

b. specify a tolerance on the error of $f(x)$

We will illustrate the *Bisection Method* with examples using both MATLAB and Excel.

Example 2.7

Use the Bisection Method with MATLAB to approximate one of the roots of

$$y = f(x) = 3x^5 - 2x^3 + 6x - 8 \qquad (2.19)$$

by

a. by specifying 16 iterations, and using a **for end** loop MATLAB program

b. by specifying 0.00001 tolerance for f(x), and using a **while end** loop MATLAB program

Solution:

This is the same polynomial as in Example 2.4.

a. The **for end** loop allows a group of functions to be repeated a fixed and predetermined number of times. The syntax is:

```
for x = array
commands...
end
```

Before we write the program script, we must define a function assigned to the given polynomial and save it as a *function m-file*. We will define this function as *funcbisect01* and will save it as *funcbisect01.m*.

```
function y= funcbisect01(x).
y = 3 .* x .^ 5 - 2 .* x .^ 3 + 6 .* x - 8;
% We must not forget to type the semicolon at the end of the line above;
% otherwise our script will fill the screen with values of y
```

On the script below, the statement for k = 1:16 says for k = 1, k = 2, ..., k = 16, evaluate all commands down to the **end** command. After the k = 16 iteration, the loop ends and any commands after the end are computed and displayed as commanded.

Let us also review the meaning of the fprintf('%9.6f %13.6f \n', xm,fm) line. Here, %9.6f and %13.6f are referred to as *format specifiers* or *format scripts*; the first specifies that the value of xm must be expressed in *decimal format* also called *fixed point format*, with a total of 9 digits, 6 of which will be to the right of the decimal point. Likewise, fm must be expressed in *decimal format* with a total of 13 digits, 6 of which will be to the right of the decimal point. Some other specifiers are %e for scientific format, %s for string format, and %d for integer format. For more information, we can type help fprintf. The special format \n specifies a linefeed, that is, it prints everything specified up to that point and starts a new line. We will discuss other special formats as they appear in subsequent examples.

The script for the first part of Example 2.7 is given below:

```
x1=1; x2=2;                          % We know this interval from Example 2.4, Figure 2.6
disp('  xm          fm')             % xm is the average of x1 and x2, fm is f(xm)
disp('-------------------------')     % insert line under xm and fm
for k=1:16;
    f1=funcbisect01(x1); f2=funcbisect01(x2);
xm=(x1+x2) / 2; fm=funcbisect01(xm);
fprintf('%9.6f %13.6f \n', xm,fm) % Prints xm and fm on same line;
if (f1*fm<0)
  x2=xm;
else
   x1=xm;
   end
end
```

When this program is executed, MATLAB displays the following:

```
  xm              fm
-------------------------
 1.500000     17.031250
 1.250000      4.749023
 1.125000      1.308441
 1.062500      0.038318
 1.031250     -0.506944
 1.046875     -0.241184
 1.054688     -0.103195
 1.058594     -0.032885
 1.060547      0.002604
 1.059570     -0.015168
 1.060059     -0.006289
 1.060303     -0.001844
 1.060425      0.000380
 1.060364     -0.000732
 1.060394     -0.000176
 1.060410      0.000102
```

We observe that the values are displayed with 6 decimal places as we specified, but for the
integer part unnecessary leading zeros are not displayed.

b. The **while end** loop evaluates a group of commands an indefinite number of times. The syntax

is:
while expression
commands...

end
The commands between **while** and **end** are executed as long as all elements in expression are
true. The script should be written so that eventually a ***false*** condition is reached and the loop
then terminates.
There is no need to create another function *m–file*; we will use the same as in part a. Now we
type and execute the following **while end** loop program.

```
x1=1; x2=2; tol=0.00001;
disp('   xm          fm'); disp('-----------------------')
while (abs(x1-x2)>2*tol);
  f1=funcbisect01(x1); f2=funcbisect01(x2); xm=(x1+x2)/2;
  fm=funcbisect01(xm);
  fprintf('%9.6f %13.6f \n', xm,fm);
  if (f1*fm<0);
    x2=xm;
  else
   x1=xm;
  end
end
```

When this program is executed, MATLAB displays the following:

xm	fm
1.500000	17.031250
1.250000	4.749023
1.125000	1.308441
1.062500	0.038318
1.031250	-0.506944
1.046875	-0.241184
1.054688	-0.103195
1.058594	-0.032885
1.060547	0.002604
1.059570	-0.015168
1.060059	-0.006289
1.060303	-0.001844
1.060425	0.000380
1.060364	-0.000732
1.060394	-0.000176
1.060410	0.000102
1.060402	-0.000037
1.060406	0.000032
1.060404	-0.000003

Next, we will use an Excel spreadsheet to construct a template that approximates a real root of a function with the bisection method. This requires repeated use of the IF function which has the following syntax.

=IF(logical_test,value_if_true,value_if_false)

where
logical_test: any value or expression that can be evaluated to true or false.
value_if_true: the value that is returned if logical_test is true. If logical_test is true and value_if_true is omitted, true is returned. Value_if_true can be another formula.

value_if_false is the value that is returned if logical_test is false. If *logical_test* is false and value_if_false is omitted, false is returned. Value_if_false can be another formula.
These statements may be clarified with the following examples.
=IF(C11>=1500,A15, B15):If the value in C11 is greater than or equal to *1500*, use the value in A15; otherwise use the value in B15.

=IF(D22<E22, 800, 1200):If the value in D22 is less than the value of E22, assign the number
800; otherwise assign the number *1200*.
=IF(M8<>N17, K7*12, L8/24):If the value in *M8* is not equal to the value in *N17*, use the value in
K7 multiplied by *12*; otherwise use the value in *L8* divided by 24.

2.5 Exercises

1. Use MATLAB to sketch the graph $y = f(x)$ for each of the following functions, and verify from the graph that $f(a)$ and $f(b)$, where a and b defined below, have opposite signs. Then, use Newton's method to estimate the root of $f(x) = 0$ that lies between a and b.

   a. $f_1(x) = x^4 + x - 3$ $\quad a = 1 \quad b = 2$

   b. $f_2(x) = \sqrt{2x+1} - \sqrt{x+4}$ $\quad a = 2 \quad b = 4$

   Hint: Start with $x_0 = (a+b)/2$

2. Repeat Exercise 1 above using the Bisection method.

3. Repeat Example 2.5 using MATLAB.

   Hint: Use the procedure of Example 2.2