Newton's Method

A Program for the

TI 83/84 Calculator

A program is a series of steps executed by a computer — for example the one that is at the heart of a graphing calculator. The language used is based on the computer programming language BASIC (short for Beginners All-purpose Symbolic Instruction Code). It allows you to construct programs to automate long, repetitive calculations or special purpose techniques.

This program calculates an estimate of root for a given function using Newton's Method. The technique uses the derivative of the function to give the tangent line to the function as an approximation for the function. This program will also serve as a good introduction to the basic idea (pun intended) of writing a program.

To get started, press the [PRGM] key on the calculator. The menu gives you three options with the preselected option [PRGM] showing you programs available to run. Use the arrow keys and the [ENTER] key to select a program ready to run. The [EDIT] option allows you to edit any programs in your calculator. To enter a new program, which is what we are doing, choose [NEW] (or had you already figured that out?).

When you choose [Create New], the calculator displays:

PROGRAM
Name=

The [ALPHA] key has already been selected, put in the name of your program using the letters shown on the base of the calculator in green to the upper right of the keys. Give your program a nice snappy, memorable name like NEWTON.

When you press [ENTER] after putting in the name, the calculator show

PROGRAM:***(Your program name)***
:

The colon starts the line where you enter the program. To locate the calculator commands listed, when you are editing a program, press PRGM again, and you will get menus of program commands so you don't have to search in CATALOG. If you are searching in the catalog for a command, pressing a letter key takes you to that letter in the alphabetical list saving you a lot of scrolling time. Here are the commands for this program and a short commentary on what they do and how they are entered.

Input "START = ",x

> The calculator will display what is between the quote marks. When program runs, the calculator displays START = with a question mark then waits until you put in a starting value estimate of the root. The Input command is in the catalog. Enter the letters S, T, A, R, and T using the [ALPHA] key and the green letters above the keys (e.g. To get "S" press [ALPHA] the [LN]). Use the [x,T,Θ,n] key for the x here. The entry between quotation marks is displayed as entered and has no effect on the program itself. You can be creative here.

1→N

Stores the value 1 in a counter called N. The → is the [STO>] key. The counter counts the number of iterations the program runs. The program will run until the change from one estimate to the next is less than the accuracy limit .0000000001.

Lbl 1

This is program marker 1. When the program executes GOTO 1, it goes to this marker (*Label*ed 1) then goes on to the step following this point in the program.

x–Y1/Y2→R

To enter Y1 and Y2 press the [VARS] key, select [Y-VARS] 1:Function to enter these functions. Before you run this program, you will need to define the function under Y1 and the derivative of the function under Y2 using the [Y=] as if you were going to draw the graphs.

This is the step that actually uses Newton's method to calculate a new estimate of the root. It stores the calculated value in R. When you run the program, you will enter the function in graphing mode under Y1. Enter the derivative of the function under Y2. When the program does x–Y1/Y2, it is calculating a new estimate of the root, and storing it in R.

If abs (x-R) < E-10

Find the abs command using the [MATH] key and select NUM from the menu. The E is the scientific notation key [2ND] [EE] and use the [(-)] key for the exponent.

This step tests the difference between the new estimate and the old estimate. If the absolute value of the difference is less than .000000001 (10^{-10}), the program executes the very next line of code. Here, this will go to program marker 2 which is where the program will finish. Otherwise, if it hasn't reached the level of accuracy set (a FALSE answer to the test), it skips to the next line which will send it through the cycle for another iteration.

Note that you can change the criteria for accuracy if you like. Less accuracy will take fewer cycles through Newton's Method, but the answer will be, well, less accurate.

Goto 2

When the If statement is TRUE, the program executes this step which jumps down to Lbl 2 where the program ends.

R→x

When the If statement is FALSE, the program skips the previous step and jumps here. Now it prepares to make another calculation cycle.

The calculated new estimate is moved to the x value and becomes the starting point for the calculation of a newer estimate.

```
N+1 → N
```

Since the program is going to calculate a new estimate, the counter is incremented.

```
Goto 1
```

Sends the program back to Lbl1 where it will go through the loop one more time.

```
Lbl 2
```

When the the set accuracy is reached, the program jumps to this point. Now the results will be displayed.

```
Disp “Root = “,R
```

The program displays the result of the calculation of the root. The Disp command displays whatever is in the quotes followed by the value contained in R. You may omit the label inside the quotes after the command and just list the variable.

```
Disp “Iter = “,N
```

The program lists the number of cycles it took to achieve the level of accuracy listed. Then it falls off the end of the program and displays DONE.

When you have completed entering the program. Use [2ND] [QUIT] to exit programming mode.

Before you run this program, go to GRAPH mode and enter the equation for Y1. Graph the program and estimate the *x* value(s) where the curve crosses the x-axis. These locations are the roots and your estimates will be your starting values.

Also, before you run this program, you must have the derivative of the equation entered as Y2. The intercepts of this graph are not related to the roots you are finding, but they will indicate where any relative minima or maxima occur.

Now when you press the [PRGM] key you should see your program ready to execute. Will it run correctly the first time? Oh, I hope so, but Murphy’s Law seems to apply with particular vengeance to programmers. If there is an error, select the goto option, and see if you can fix whatever is wrong.

When your estimate is near the actual root, this technique converges very quickly. However, there are regions of chaotic turbulence where convergence can be slow and erratic. These particular areas are of interest to mathematicians who study Chaos Theory, which examines behaviors of dynamical system that are extremely sensitive to initial conditions. Take a look. The results can be puzzling.