Assignment I (MA226)

Name: Sourav Bikash

Roll No.:11012338

Submission Date: 08/01/2013 Time:23:59 hrs.

Aim of the Problem:

Approximate solution of the root of the equation: $f(x)= 3x^2 – e^x$ using Newton-Raphson Method and implementing it with R and C/C++.

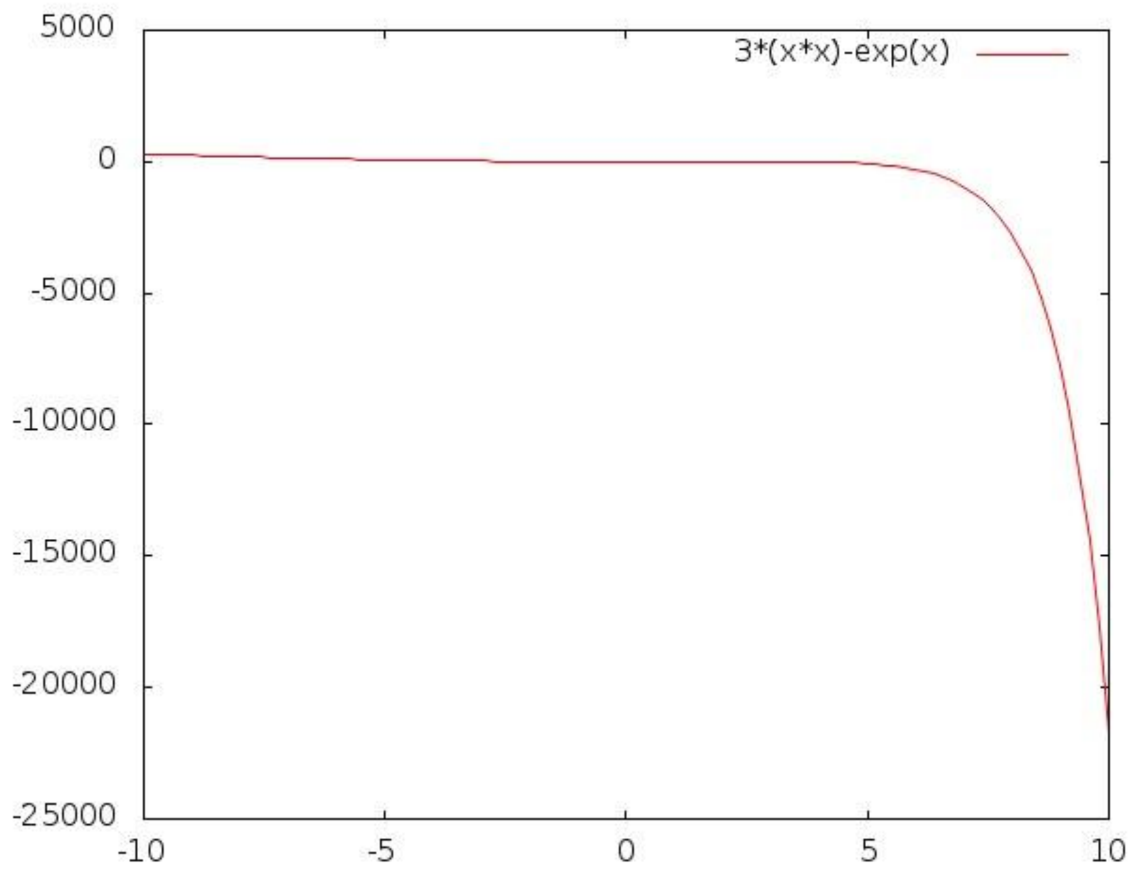
Mathematical Analysis/Theory:

The Newton-Raphson method is an iterative process for solving the root of the equation f(x) = 0. According to the method, starting with an initial guess of x_0 , apply the iterative formula:

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

Where f' denotes the derivative of the function. The iteration stops until you arrive at an acceptable limit $|x_{n+1} – x_n| < \mu$ where μ is some pre-specified tolerance value.

Generating the graph:



The graph was generated by a GNUPLOT software.

The following functions come into use:

`gnuplot>plot<function>`

To save it as a image file one can use: "set terminal .jpeg"→"set output"filename.jpeg"→replot

Inference from the Graph:

We first draw a graph to see the general shape of the function. If we don't get it right, that will tell us on the domain of the function. Moreover, we will have a rough idea of where the root lies on the x-axis. That will then be helpful for the initial guess. In our case the guess is a value less than the actual root.

Therefore let us select $X_0=-10$.

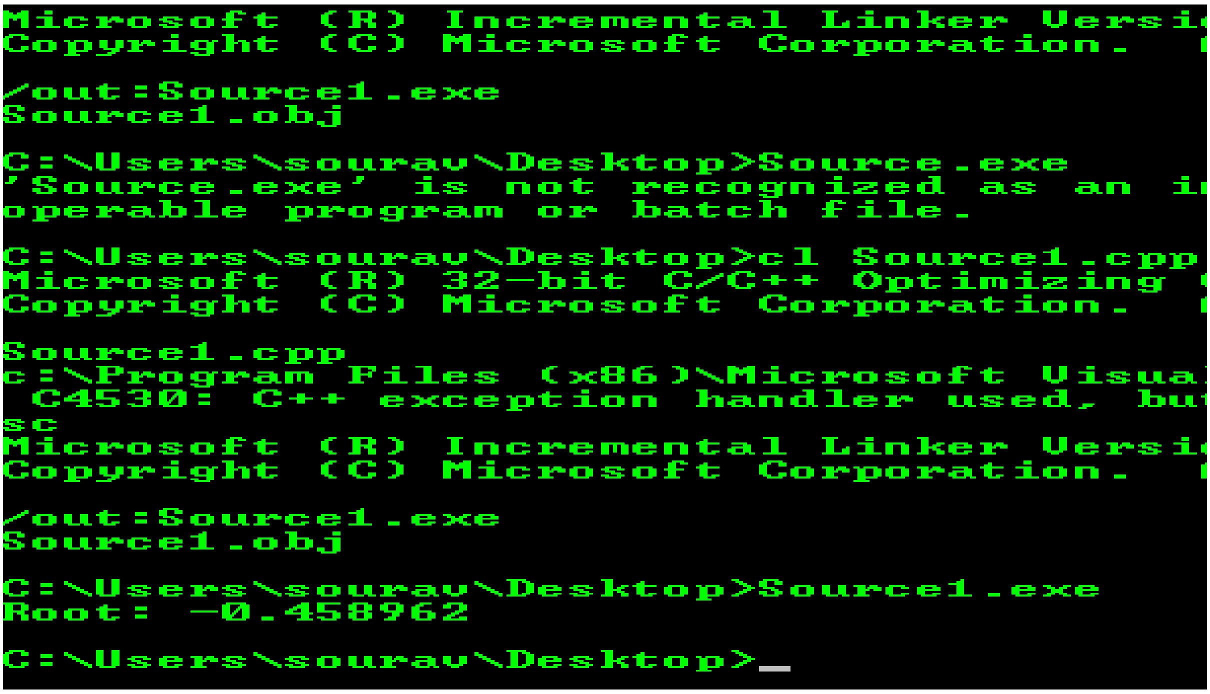
Implementation of the Newton-Raphson Algorithm:

Then, we are ready for the implementation of the *Newton-Raphson* algorithm that will give us the root of **f(x)**.

Implementation using C++:

```
#include<iostream>
#include<cmath>
using namespace std;

double newtonRaphson (double x)
{
  return x - (3*x*x-exp(x)) / (6*x-exp(x));//using the given function and its
derivative
}

int main ()
{
        double x = -10.0;         // the initial guess close and < to the actual root
from the graph
        double error = 0.00001;    // 0.00001 is the error level we wish
        double x1;

        do
        {
                x1 = x;
                x = newtonRaphson(x);
        }
        while (abs(x1 - x) > error);     // while loop because the
                                        // number of iterations is not
                                        // known beforehand
        cout << "Root: " << x << endl;

        return 0;
}
```

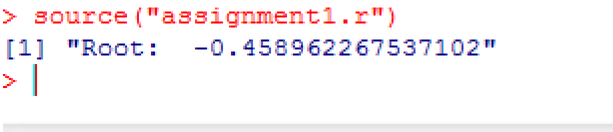
The output of the program can be seen from the terminal window below: **root=-0.458962.**



```
Microsoft (R) Incremental Linker Versi
Copyright (C) Microsoft Corporation.

/out:Source1.exe
Source1.obj

C:\Users\sourav\Desktop>Source.exe
'Source.exe' is not recognized as an i
operable program or batch file.

C:\Users\sourav\Desktop>cl Source1.cpp
Microsoft (R) 32-bit C/C++ Optimizing
Copyright (C) Microsoft Corporation.

Source1.cpp
c:\Program Files (x86)\Microsoft Visua
 C4530: C++ exception handler used, bu
sc
Microsoft (R) Incremental Linker Versi
Copyright (C) Microsoft Corporation.

/out:Source1.exe
Source1.obj

C:\Users\sourav\Desktop>Source1.exe
Root: -0.458962

C:\Users\sourav\Desktop>_
```

Implementation using R:

```
newtonRaphson <- function (x){
      x - ((3*x*x-exp(x)) / (6*x-exp(x)))#inputfunction
}


x <- -10.0 # initial guess based on the graph
x1 <- 0    # sets x1 to 0 before the first loop
error <- 0.00001#the error is specified


while (abs(x1 - x) > error){
      x1 <- x
      x <- newtonRaphson(x)
}#using a while loop for the problem


print (paste("Root: ", x))# the required output obtained
```



```
> source("assignment1.r")
[1] "Root:  -0.458962267537102"
>
```

Finally, the result given in **R** is approximately the same as the one we obtained with the **C/C++** snippet. We are done.

Approximate Root=-0.458962267

(Implementation was done in windows environment)

Conclusion:

Approximate Root= -0.458962

We can also obtain another root on the positive side of the x-axis by choosing the guess value say X_0=+10.

Then we obtain a positive approx. root=3.73308.

Therefore,

The roots are=-0.458962, 3.73308.