# Khulna University of Engineering & Technology, Khulna

Department of Biomedical Engineering

SESSIONAL REPORT

Course No: **BME 2152**

Experiment No : **06**

Name of the Experiment: **Finding the root of polynomial equation using Newton Raphson Method.**

Remarks:

Name: **Md. Fuad Hasan Hamim**

Roll : **2215029**

Year : **2nd**

Term : **1st**

Date of performance: 22 January 2025

Date of submission : 29 January 2025

## Objectives:

The main objectives of this sessional are:

- To learn how to find out roots of polynomial equation using Newton Raphson Method in MATLAB.
- To know the operation of polyder() function in MATLAB.

## Introduction:

**Newton Raphson Method** or **Newton Method** is a powerful technique for solving equations numerically. It is most used for approximation of the roots of the real-valued functions. Newton Rapson Method was developed by Isaac Newton and Joseph Raphson, hence the name Newton Rapson Method.

The Newton-Raphson method and bracketing methods like bisection and false position work differently. Newton-Raphson needs one good guess close to the root and uses the function's derivative, making it faster but less reliable if the guess is poor or the function behaves unpredictably. Bracketing methods, on the other hand, require two guesses that enclose the root, guaranteeing convergence as long as the function is continuous in that range. They don't need derivatives, making them simpler but slower. Newton-Raphson is best for speed, while bracketing methods are more reliable for finding roots.

When we use the Newton-Raphson method, we start by rewriting the equation in the form, where represents the function whose root we want to find. Then, we calculate the derivative of the function, because it's a key part of the process. After that, we choose an initial guess that we think is close to the actual root. Picking a good initial guess can make the method work much better.

Using the formula, we substitute our current guess into and to calculate the next approximation. This step adjusts our guess based on the slope of the function, helping us get closer to the root.

We repeat this process, using each new approximation to calculate the next one. After every step, we check if the difference between two successive guesses, is smaller than a tolerance level we've set, like. If it is, or if the function value is very close to zero, we stop because we've found an accurate approximation of the root. This method works well as long as our initial guess is good and the function behaves nicely, meaning isn't zero or undefined near the root.

The Newton-Raphson method is a fast and efficient way to find roots of equations, especially when the initial guess is close to the actual root. It converges quickly and works well for equations that are otherwise difficult to solve. The method uses the slope of the function, which helps it make accurate approximations in fewer steps.

However, it has some drawbacks. The method relies heavily on a good initial guess; a poor guess can lead to wrong results or failure to converge. If the derivative of the function is zero or undefined during the process, the method cannot continue. It may also struggle with equations that have multiple roots or sharp changes. Additionally, calculating the derivative can sometimes be complicated or time-consuming.

**Formula:**

The iterative formula for the Newton Raphson Method is:

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

Where $x_n$ is the current approximation, $f(x_n)$ is the function value at $x_n$, $f'(x_n)$ is the derivative of $f(x)$ at $x_n$ and $x_{n+1}$ is the next approximation.

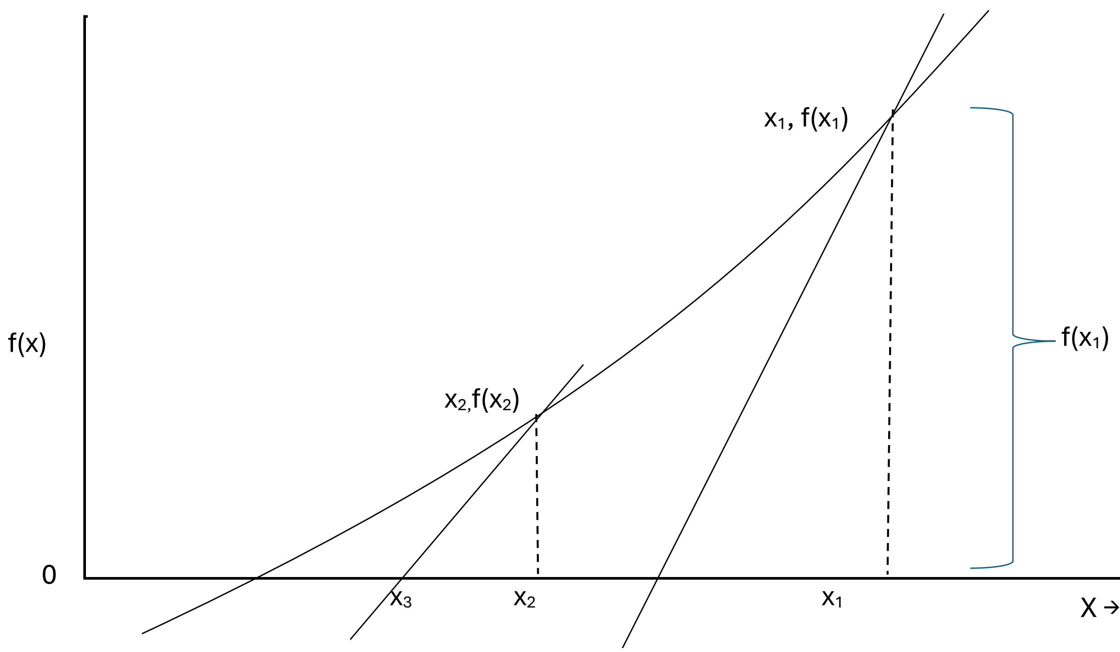


**Figure: Newton- Raphson Method.**

## Algorithm for Newton Raphson Method:

Step 1. Input equation

Step 2. Input guess value

Step 3. Find derivative of the equation

Step 4. $x_2 = x_1 - a(x_1)/b(x_1)$

Step 5. Find $a(x_2)$

Step 6. If $a(x_2) = 0$; print: The root is $x_2$ and terminate the loop

Step 7. If $|(x_2-x_1)/x_2| < eps$; print: The root is $x_2$ and terminate the loop

Else $x_1 = x_2$

Step 8. Go to step 4 and repeat.

**Task 1 || Finding the root of the polynomial equation by using Newton Raphson Method.**

**Code:**

```
clc;
clear all;

eps =0.000001;
a=input('Co-effiicients of the equation: ');
x1= input('Guess value: ');
b=polyder(a);

for n = 1:50
        x2= (x1- (polyval(a,x1)/polyval(b,x1)));
        c = polyval(a,x2);

if c==0
    fprintf('Root is x2: %g', x2);
    return;
end

if (abs((x2-x1)/x2) < eps)
    fprintf('Root is x2: %g\n', x2);
return

else
    x1 = x2;
end
end
```

| **Output:** |
|---|
| Co-effiicients of the equation: [1 -3 2]<br>Guess value: 0<br>Root is x2: 1<br>>><br><br>Co-effiicients of the equation: [1 -2 1 -1]<br>Guess value: 2<br>Root is x2: 1.75488<br>>><br><br>Co-effiicients of the equation: [1 -4 -10]<br>Guess value: -1<br>Root is x2: -1.74166<br>>><br><br>Co-effiicients of the equation: [4 -14 6]<br>Guess value: 10<br>Root is x2: 3<br>>> |
| **Comment:** |
| The code works well to find the root of a polynomial using the Newton-Raphson method and gives the correct result. |

## Conclusion:

In this sessional, we used the Newton-Raphson method in MATLAB to find the roots of a polynomial equation. By working with functions like polyder() and polyval(), we saw how the method quickly converges to an accurate solution when a good initial guess is provided.

This session showed that while the method is fast and effective, it depends heavily on the choice of the starting point and the behavior of the function near the root. Overall, the Newton-Raphson method proved to be a reliable tool for solving equations, giving accurate results in just a few steps.

## References:


[1] S. C. Chapra and R. P. Canale, "Numerical Methods for Engineers," ScienceDirect, 2010. [Online]. Available: https://www.sciencedirect.com/.

[2] R. Paul, "Newton-Raphson Method: Formula, Steps, and Examples," *GeeksforGeeks*, 2021. [Online]. Available: https://www.geeksforgeeks.org/newton-raphson-method/.