

**IS5306-NUMERICAL METHODS**

**Project Work: Group 03**

**Group Members:**

|  |  |
| --- | --- |
| EG/2021/4410 | ARACHCHI W.A.H.U.W. |
| EG/2021/4412 | ARACHCHI W.A.T.T.W. |
| EG/2021/4414 | ARIVANAN V. |
| EG/2021/4415 | ARIVARASAN J. |
| EG/2021/4417 | ASHFAQ M.R.M. |
| EG/2021/4418 | ATHAPATHTHU A.A.V.S. |
| EG/2021/4419 | ATHTHANAYAKA K.A.L.G. |
| EG/2021/4424 | BALASOORIYA J.M. |
| EG/2021/4426 | BANDARA A.W.M.L.M. |
| EG/2021/4432 | BANDARA K.M.T.O.N. |
| EG/2021/4433 | BANDARA L.R.T.D. |
| EG/2021/4434 | BANDARA M.M.D.L. |
| EG/2021/4437 | BANDARA R.M.U.T. |
| EG/2021/4447 | BRAVIN K. |
| EG/2021/4449 | BULEGODAARACHCHI D.C. |

**Table of Contents**

[**1** **The Selected non linear equation and its relevance** 3](#_Toc186224390)

[**2** **Manual Solutions** 5](#_Toc186224391)

[**2.1** **Bisection Method** 5](#_Toc186224392)

[**2.2** **Newton-Raphson Method** 7](#_Toc186224393)

[**2.3** **Fixed Point Method** 10](#_Toc186224394)

[**3** **MathLab Implementation** 12](#_Toc186224395)

[**3.1** **Bisection Method** 12](#_Toc186224396)

[**3.2** **Newton-Raphson Method** 13](#_Toc186224397)

[**3.3** **Fixed Point Method** 14](#_Toc186224398)

[**4** **The Comparison and Discussion of the Findings.** 15](#_Toc186224399)

# **The Selected non linear equation and its relevance**

This equation has been selected due to its significance in modeling real-world systems where nonlinear relationships between variables need to be analyzed. It serves as a representative mathematical model that can be applied in a variety of scientific and engineering applications, including those in the computer engineering field.

To solve this equation, we aim to determine its root within the interval [1,2] [1,2], where the equation is expressed as:

*f* (*x*)=*x*3−*x*−2

The selection of this nonlinear equation is motivated by its practical applications in computer engineering systems where nonlinear relationships often emerge. These relationships are commonly encountered in areas such as algorithm optimization, signal processing, and hardware design.

Example in Real-Life Scenario:

Consider the behavior of a computer processor or a circuit board under varying operational conditions:

* Thermal Management in Processors: As processors work, they experience varying thermal loads, which affect the overall performance. Nonlinear equations like

*f* (*x*)=*x*3−*x*−2 can be used to model temperature changes across components as power consumption varies.

* Power Efficiency in Electronics: In systems with power regulation circuits, the relationship between voltage, current, and resistance can often be nonlinear, necessitating the use of nonlinear equations for accurate modeling and analysis.

These examples illustrate the importance of solving equations such as

*f* (*x*)=*x*3−*x*−2

in predicting and optimizing system performance in computer hardware design and thermal management.

Nonlinear Equation Appears in Such Systems;

* Processor Performance and Heat Dissipation: Nonlinear equations are used to model how heat generated by processors influences the overall system performance. As clock speeds increase or system workloads change, thermal effects become nonlinear, requiring numerical methods to predict temperatures and adjust cooling mechanisms.
* Optimization Algorithms: Nonlinear equations appear in optimization problems, particularly in the context of algorithmic complexity and resource allocation. In scenarios such as load balancing or memory management, the relationship between system parameters is often nonlinear, making the need for accurate solutions critical.
* Signal Processing: Nonlinear systems are prevalent in signal processing, where algorithms need to handle varying frequencies or power levels. The relationship between input signals and outputs in certain filters or transformations is often nonlinear, and solving such equations is necessary to ensure precision in the processing systems.

This equation is highly relevant in computer engineering and computational applications, including:

* Circuit Design: Nonlinear equations are crucial when modeling the performance of integrated circuits under varying conditions like power fluctuations or heat exposure.
* Algorithm Efficiency: Cubic equations can be used to describe the computational complexity of certain algorithms, especially in scenarios where resource consumption grows nonlinearly with problem size, such as in cryptography or machine learning.
* Simulations and Modeling: In computational simulations, particularly those modeling processor behavior or electronic systems, solving nonlinear equations accurately is essential to predict performance, optimize resource usage, and ensure system stability.

# **Manual Solutions**

## **Bisection Method**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| , interval [1,2] | | | | | = | T3-3T-2 | = | **0** |
| f (1) | = | 13- 1 - 2 |  |  |
|  | = | -2 |  | **F (1) < 0** |
|  | | | | |
| f (2) | = | 23 -2 -2 |  |  |
|  | = | 4 |  | **F (2) > 0** |
| Since f(1)<0. f(2) > 0; a root exists between 1 and 2 | | | | |
| **C** | = |  |  |  |
|  | = | 1.5 |  |  |
| f (1.5**)** | = | 1.53 -1.5 -2 |  |  |
|  | = | -0.125 |  |  |
| * Therefore, updated Interval [1.5,2] | | | | |

**Table 1:Iteration Table.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Iterations** | **a** | **F(a)** | **b** | **F(b)** | **c** | **F(c)** | **update** |
| 1 | 1.00000000 | -2.00000000 | 2.00000000 | 4.00000000 | 1.50000000 | -0.12500000 | a=c |
| 2 | 1.50000000 | -0.12500000 | 2.00000000 | 4.00000000 | 1.75000000 | 1.60937500 | b=c |
| 3 | 1.50000000 | -0.12500000 | 1.75000000 | 1.60937500 | 1.62500000 | 0.66601562 | b=c |
| 4 | 1.50000000 | -0.12500000 | 1.62500000 | 0.66601562 | 1.56250000 | 0.25219727 | b=c |
| 5 | 1.50000000 | -0.12500000 | 1.56250000 | 0.25219727 | 1.53125000 | 0.05911255 | b=c |

* So **1.53125000** considered as a **root** according to Bisection Method after 5 iterations

## **Newton-Raphson Method**

Initial guess :

**Newton-Raphson formula**

n = 0

n = 1

n = 2

**Table 2:Iteration Table.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Iteration |  |  |  |  | Absolute error (in%) |
| 1 | 1.5 | -0.125 | 5.7500 | 1.52174 | 1.4260 |
| 2 | 1.52174 | 0.00214 | 5.94708 | 1.52138 | 0.0236 |
| 3 | 1.52138 | 0.0000017 | 5.94379 | 1.52138 | 0.0000 |
| 4 | 1.52138 | 0.0000017 | 5.94379 | 1.52138 | 0.0000 |
| 5 | 1.52138 | 0.0000017 | 5.94379 | 1.52138 | 0.0000 |

## **Fixed Point Method**

->

Therefore:

Let interval as [1,2]

=

= 1.442

=

= 1.5874

> 1 < 2

Hence g(x) є [1,2]

h(x) = g(x) -x ; ұ x є [1,2]

h(1) = g(1) -1 >0

h(2) = g(2) -2 <0

Derivative of g(x)

g′(x)=

For the interval [1,2]:

x =1; g′(1) = = 0.154

x =2; g′(2) = = 0.118

Since ∣g′(x)∣<1| for all x∈[1,2] the fixed-point iteration will converge.

Initial Guess:

n=0 = 1.51829

Error || |1.51829-1.5| = 0.01829

n=1 = 1.52093

Error || |1.52093-1.51829| = 0.00264

n=2 = 1.5213149

Error || |1.5213149-1.52093| = 0.00038

n=3 = 1.52137

Error || |1.52137-1.52131| = 0.00006

n=4 = 1.5213783

Error || |1.52138-1.52137| = 0.00001

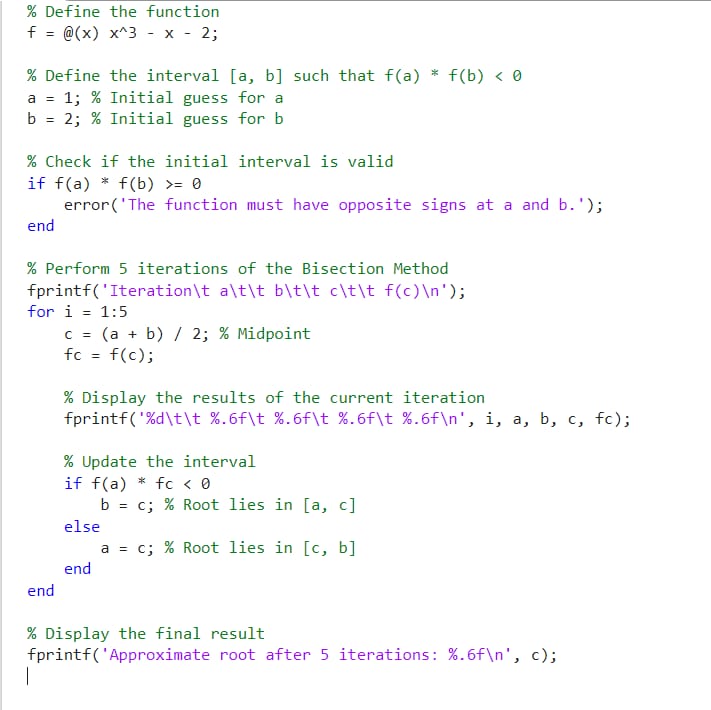
After 5 iterations, the approximate root of is x=1.545

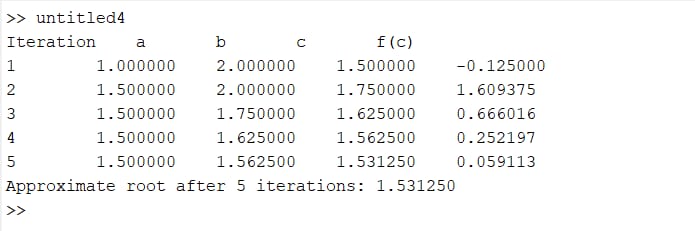
**Table 3:Iteration Table.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Iterations** |  |  | **Error** |
| 0 | 1.5 | 1.51829 | 0.01829 |
| 1 | 1.51829 | 1.52093 | 0.00264 |
| 2 | 1.52093 | 1.5213149 | 0.00038 |
| 3 | 1.5213149 | 1.52137 | 0.00006 |
| 4 | 1.52137 | 1.5213783 | 0.00001 |

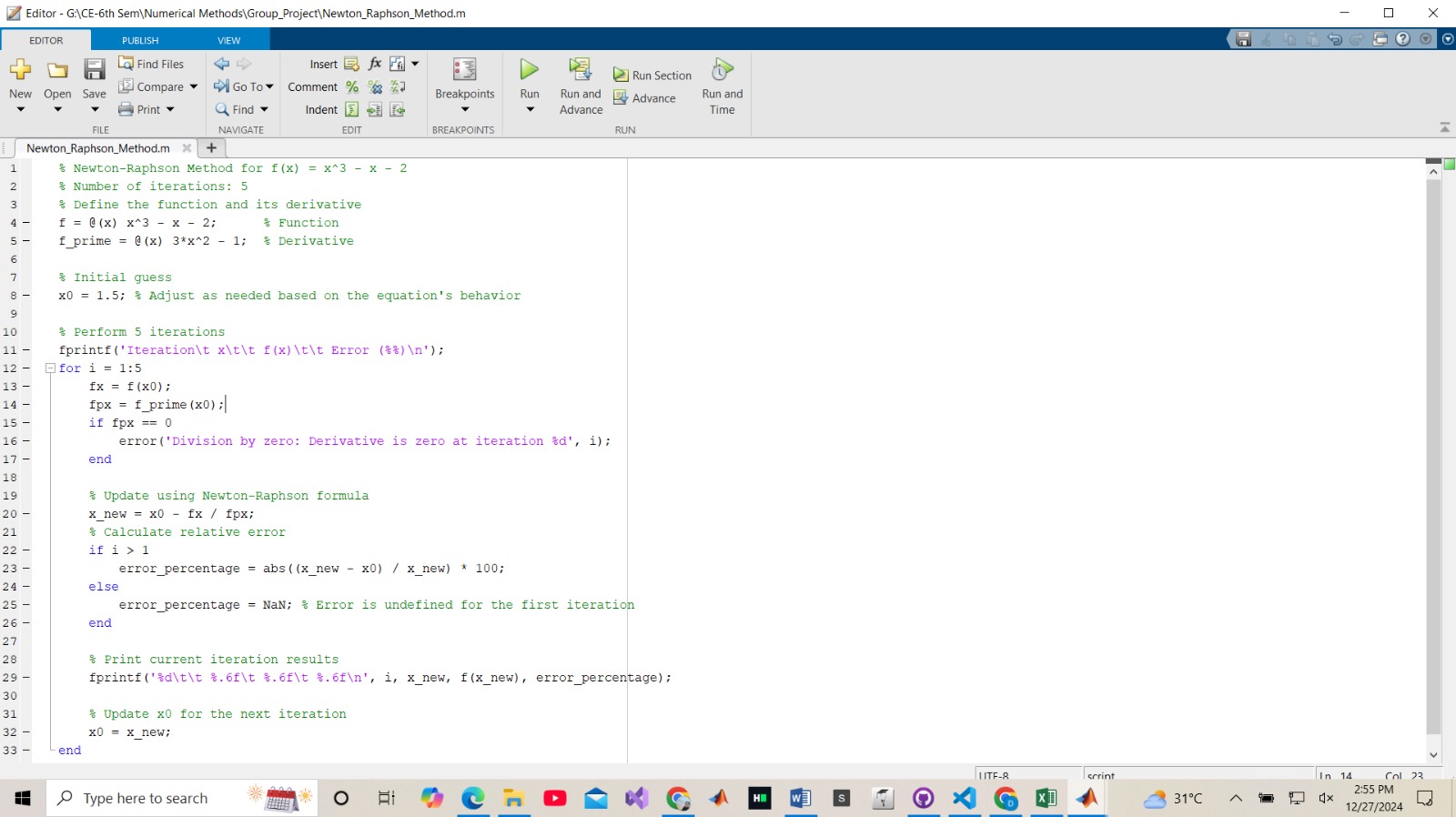
# **MathLab Implementation**

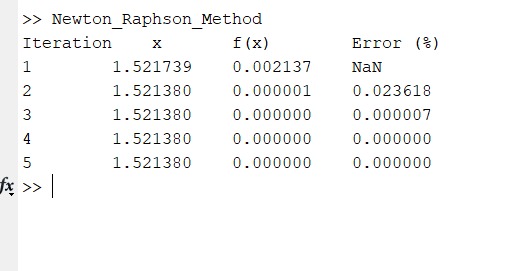
## **Bisection Method**



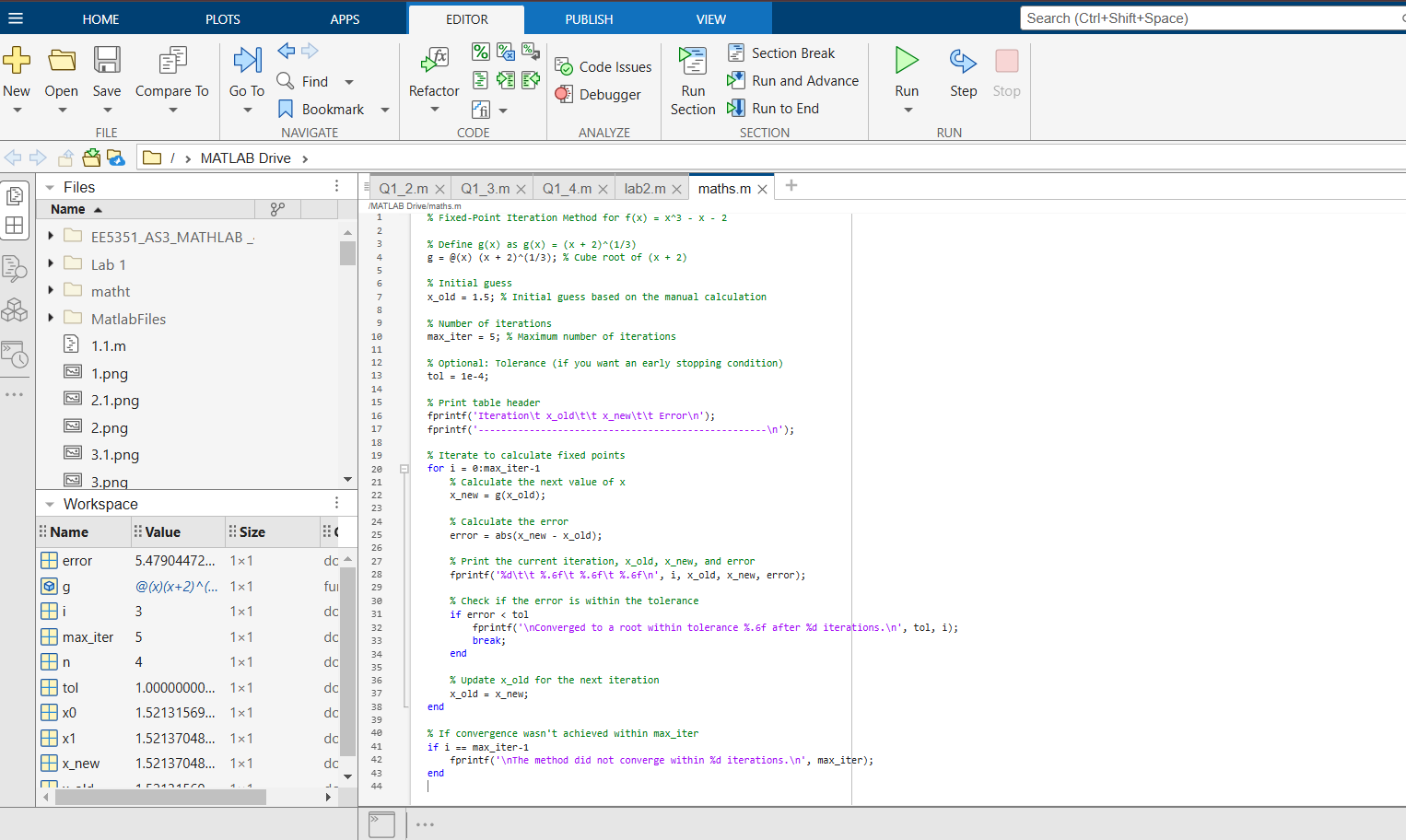
Output:

## **Newton-Raphson Method**



Output

## **Fixed Point Method**



A screenshot of a computer code

Description automatically generatedOutput:

# **The Comparison and Discussion of the Findings.**

**Discussion**

**Accuracy of Roots Found**

Using the Bisection Method the root converges to x≈1.531 with a specified tolerance.This method guarantees a root within the interval [a,b] as long as f(a)⋅f(b)<0. Its accuracy improves linearly, but achieving high precision requires more iterations. On the other hand, the Newton-Raphson Method achieves high accuracy quickly due to its quadratic convergence, especially when the initial guess is close to the root. So for x0=1.5 (initial guess), the method converges to x≈1.521 within five iterations. Fixed-Point Iteration depends heavily on the choice of g(x); poorly chosen transformations can slow convergence or even prevent it.

**Speed of Convergence**

The Bisection Method converges slowly compared to the other two, with about 20 iterations needed to reach a precision of 10^{-6}. In contrast, the Newton-Raphson Method converges quadratically and typically requires only 4-5 iterations to achieve the same precision, provided the initial guess is good. Fixed-Point Iteration is slower than Newton-Raphson and can require as many as 15-20 iterations to achieve similar precision, depending on g(x).

**Ease of Application and Challenges**

The Bisection Method is simple and reliable, requiring no derivative calculations. However, it is slower and requires f(a)⋅f(b)<0, which may not always be easy to identify. The Newton-Raphson Method is efficient and fast but requires a good initial guess and accurate calculation of f′(x). If the guess is far from the root or f′(x)=0, the method may fail. Fixed-Point Iteration is straightforward to implement once g(x) is defined, but selecting an appropriate g(x) is challenging. Poor choices can lead to divergence.

**Recommendations**

* For guaranteed convergence: Use the Bisection Method, which is best for reliable results when little is known about the function.
* For speed and efficiency: Use the Newton-Raphson Method, especially if a good initial guess and derivative are available.
* For simplicity: Use Fixed-Point Iteration, but only with a well-chosen transformation g(x).

Each method serves a unique purpose and is suited to different situations based on the problem's requirements.