BUSITEMA UNIVERSITY

Pursuing excellence

FACULTY OF ENGINEERING AND TECHNOLOGY

A REPORT ON CONTRUCTION OF CLASSES AND SUB CLASSES IN MATLAB

BY GROUP 19

COURSE UNIT: COMPUTOR PROGRAMING

LECTURER: Mr. MASERUKA BENEDICTO

ACKNOWLEDGEMENT

First and foremost, we would like to thank the Almighty God for giving us the strength to carry on with our research in Group 19. We would love to extend our gratitude to all the persons with whose help we managed to make it this far. The willingness of each one of us to invest time and provide constructive feedback has been immensely valuable in this assignment. We wish to extend our gratitude to our lecturer for his consistent guidance and valuable insights throughout this assignment. His teaching and encouragement made it possible for us to understand and practically apply concepts of data importing, organization, and storage in MATLAB.

We also thank our group members for their cooperation and contribution. Each member actively participated in research, coding, and report writing, which ensured the success of this work. Finally, we would like to express our gratitude to all the sources and references that have been cited in this report

ABSTRACT

We started our first meeting for research on 18th, October, 2025 in the university library out of which we were exposed to various concepts and applied a variety of data programming techniques. These techniques were applied together with our knowledge from numerical methods to make functions that can automatically find solutions to said numerical problems if given enough data to work with.

DEDICATION

We dedicate this report to all the individuals especially Group 19 members, who have been there with us in the process of formulating and compiling this report. To our lecturer Mr. Maseruka Benedicto whose guidance and expertise have been invaluable, your mentorship and insightful feedback have shaped our understanding.

DECLARATION

We hereby certify and confirm that the information in this report is out of our own efforts, research and it has never been submitted in any institution for any academic award.

STUDENT NAME	**REG.NO**	**COURSE**	**SIGNATURE**
NTALE JOASH KATEREGA	BU/UG/2024/2595	WAR	
WANYAMA JOSEPH EROGO	BU/UP/2024/1077	WAR	
KAKULU ALFRED	BU/UP/2024/0981	MEB	
OKORI DARIOUS	BU/UP/2024/1061	WAR	
ALITEMA VICTOR	BU/UP/2024/5098	WAR	
AMASO SUSAN	BU/UP/2024/5435	PTI	
MAATE PHILEMON	BU/UP/2024/0830	AMI	
SIKUKU BELIZER RUTH	BU/UP/2024/0846	AMI	
TWICHIRIZE FLORENCE	BU/UP/2022/1836	WAR	
OKWI NICHOLAS	BU/UP/2024/4457	WAR	

APPROVAL

We are presenting this report which has been written and produced under our efforts. We carried out research on visualizing our data into plots and graphs that are well labeled ready for easy interpretation by the final user.

DATE OF SUBMISSION: …………………….

SIGNATURE: …………………………….

Table Of Contents

List of Acronyms/Abbreviations.

MATLAB –Matrix Laboratory.

GUI – Graphics user interface

1 CHAPTER 1: INTRODUCTION

1.1 Background

Matrix Laboratory, or just MATLAB, is a high-speed programming language and computational environment employed in engineering. It was first developed in the late 1970s by computer science professor, Cleve Moler, who desired to provide his students with access to sets of mathematical software without their having to learn to program in Fortran themselves.

1.2 Historical Development

Early Development: The initial release of MATLAB, in the latter 1970s, as an interactive matrix calculator, was in Fortran. It consisted of rudimentary matrix operations and was built upon two early numerical libraries, LINPACK, for linear algebra computations, and EISPACK, to solve eigenvalue problems.

Commercialization: The program entered commercial status in 1984, when Moler, in conjunction with Jack Little and Steve Bangert, began MathWorks. This release marked an extensive revision, as it was fully implemented in C and considerably increased in features, including user-defined functions, toolboxes, and graphical user interfaces, significantly broadening the ways in which it could be utilized.

Expansion through Toolboxes: Until the late 1980s, MATLAB had expanded considerably beyond its original limits. The introduction of toolboxes enabled having specialist applications in signal processing and control systems, and others. At this point, MathWorks also added Simulink, which also became a graphical environment to model and simulate in a dynamic state system.

Recent Advances: Since its past updates, MATLAB has also evolved to meet researchers', engineers', and educators' needs as it advances in this direction. New versions have added capabilities, including the Live Editor, which supports combining code, visualizations, and descriptive text in interactive documents. These advances demonstrate how MATLAB has been evolving to become an adaptable infrastructure supporting both research in academics and industrial practice.

2 CHAPTER 2: STUDY METHODOLOGY

2.1 Introduction

It is one of the core competencies of engineering and data science to create and manipulate algorithms to solve functions and equations. MATLAB provides a suite of functions and loops through which equations can be manipulated to find solutions using numerical method.

For the first part of this assignment, we were provided equations that required roots or solutions. We were to implement the use of function handles to continue the working on the equations in various methods i.e.; Newton Raphson method, Bisection method, Secant Method and Fixed iteration method.

The Objectives were:

- To construct classes functions that could solve numerical methods.
- To apply the functions on real world problems that require solutions.
- To visualize the time taken for each method on graphs and using the information to see the most suitable method.
- To compare the average time taken for each method.

This project made us get better with MATLAB's potential in functions analysis and data visualization. The project showed us how functions can be used to solve problems and equations whether multivariable or single variable and displayed for best choice. These skills provide an avenue for future works in engineering, research, and decision-making activities.

2.2 Design Process

1. As a group we decided to review all our methods of solving equations using numerical approximations.
2. We went through different ideologies, flow charts and pseudocodes. That we would then apply into our final scripts.
3. We organized meetings during our available time where we went through lecture notes and modules to come up with possible lines of code to put in our script.

4. We inquired from other groups about their progress and refined some of ideas from them.
5. The code for both numbers was written down.
6. Under Visualization plots were created to highlight the time taken difference.
7. Debugging was done in the presence of all members that were available to get a better understanding of how it worked.
8. Documentations was carried out in report making and presentation drafting.

3 CHAPTER 3: METHODOLOGY

3.1 Testing Our Classes

3.1.1 Differential Solutions

```
f = @(x) x - 2*sin(x);
f_d = @(x) 1 - 2*cos(x);e



Test_Raphoson = NewtonRaphson(f, f_d, 2, 5e-6);
Test_Raphoson.Solution();
```

```
Newton-Raphson Root = 1.89549
```

```
Secant_Test = Secant(f, 2, 1.9, 5e-6);
Secant_Test.Solution();
```

```
Secant Root = 1.89549
```

3.1.2 Integral Solutions

```
f_ode = @(x, y) sin(x) - y;


Euler_Test = Euler(f_ode, 0, 1, 0.2, 2);
Euler_Test.Solution();
```

```
Euler Method Results:
     x        y   
    ___    _______

      0          1
    0.2        0.8
    0.4    0.67973
    0.6    0.62167
    0.8    0.61027
      1    0.63168
    1.2    0.67364
    1.4    0.72532
    1.6    0.77735
    1.8    0.82179
      2     0.8522
```

```
RK_Test = RungeKutta(f_ode, 0, 1, 0.2, 2);
RK_Test.Solution();
```

```
Runge-Kutta Method Results:
     x         y   
    ___     _______

      0           1
    0.2      0.8374
    0.4     0.73967
    0.6     0.69288
    0.8     0.68433
      1     0.70241
    1.2     0.73664
    1.4     0.77764
    1.6     0.81724
    1.8     0.84848
      2     0.86573
```

```
classdef Secant < Numerical_Methods
    properties (Access = private)
        X_0
        X_1
    end

    methods
        function obj = Secant(f, x0, x1, tol)
            obj@Numerical_Methods(f, tol);
            obj.X_0 = x0;
            obj.X_1 = x1;
        end

        function obj = Solution(obj)
            x0 = obj.X_0;
            x1 = obj.X_1;
            f = obj.F;

            x2 = x1 - f(x1)*(x1 - x0)/(f(x1) - f(x0));

            if abs(x2 - x1) < obj.Tol
                fprintf('Secant Root = %.5f\n', x2);
                obj.X_0 = x2;
            else
```

```
                obj.X_0 = x1;
                obj.X_1 = x2;
                obj = obj.Solution();
            end
        end
    end
end



classdef NewtonRaphson < Numerical_Methods
    properties (Access = private)
        F_d
        X_0
    end

    methods
        function obj = NewtonRaphson(f, f_d, x0, tol)
            obj@Numerical_Methods(f, tol);
            obj.F_d = f_d;
            obj.X_0 = x0;
        end

        function obj = Solution(obj)
            x0 = obj.X_0;
            x1 = x0 - obj.F(x0)/obj.F_d(x0);

            if abs(x1 - x0) < obj.Tol
                fprintf('Newton-Raphson Root = %.5f\n', x1);
                obj.X_0 = x1;
            else
                obj.X_0 = x1;
                obj = obj.Solution();
            end
        end
    end
end


classdef Euler < Numerical_Methods
    properties (Access = public)
        X_0
        Y_0
        H
        X_end
    end

    methods
        function obj = Euler(f, x0, y0, h, x_end)
            obj@Numerical_Methods(f);
            obj.X_0 = x0;
            obj.Y_0 = y0;
            obj.H = h;
            obj.X_end = x_end;
        end

        function Solution(obj)
            x = obj.X_0:obj.H:obj.X_end;
            y = zeros(size(x));
            y(1) = obj.Y_0;

            for i = 1:length(x)-1
                y(i+1) = y(i) + obj.H * obj.F(x(i), y(i));
            end

            fprintf('\nEuler Method Results:\n');
            disp(table(x', y', 'VariableNames', {'x', 'y'}));
        end
    end
end


classdef (Abstract) Numerical_Methods
    properties (Access = public)
        F
        Tol
    end

    methods
        function obj = Numerical_Methods(f, tol)
            if nargin < 2
                tol = 5e-6; % default tolerance
            end
            obj.F = f;
            obj.Tol = tol;
        end
    end

    methods (Abstract)
        obj = Solution(obj)
    end
end
```

4 CHALLENGES

- Limited time given for the assignment to be completed.
- Referencing errors at times made the work hectic
- Lack of concentration due to the different course units being handled at the same time

5 RECOMMENDATIONS

- We recommend that the lecturer to always give us ample time to complete the assignment.

6 CONCLUSION AND LEARNING EXPERIENCE

Upon assignment completion, we really appreciated the MATLAB especially from modules 3 and 5to the coverage. This embedded a real-life application to plots into the different engineering aspects. We gained a deeper rhythm on algorithm development, structured programming and also the previous modules 1-4. This experience was of utmost importance to all of us.