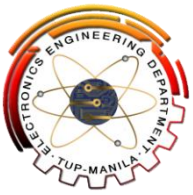
TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES
Manila
COLLEGE OF ENGINEERING
Department of Electronics Engineering
AECE6-L
Circuits 2, Laboratory



PROJECT No.1
SimpifyMi
(1/3 Simpson's Rule with n Segment and Matrix Inversion)

Submitted By:

AQUINO, Seamus Rod C.
MANZANO, Justin Tracy Q.
BS ECE 2C

Submitted To:

ENGR. GLENN C. VIRREY
Instructor

AUGUST 15 , 2020
Second Semester, S.Y. 2019-2020

A. INTRODUCTION

SimpifyMi is a simple mathematical program for students, educators, and engineers both in and out of the classroom. It has been developed on the basis of partial fulfillment of requirements in Advanced ECE Mathematics Laboratory and carried out by our engineering students for the benefit of the educational e-learning sector.

SimpifyMi offers two numerical method functions. The Simpify is a program run by MATLAB that evaluates definite integrals by using 1/3 Simpson's Rule. Mi is a program that uses MATLAB to evaluate an inverse matrix.

MATRIX INVERSION

Matrix inversion is the process of finding the **matrix** B that satisfies the prior equation for a given invertible **matrix A**.

The inverse of a square matrix A, sometimes called a reciprocal matrix, is a matrix A^{-1} such that

$$\mathsf{A}\mathsf{A}^{-1} = \mathsf{I},$$

where I is the identity matrix.

If A is an n×n matrix and I be an n×n identity matrix, then the n×n matrix B (also called as B=A^{-1}) said to be inverse matrix such that AB=BA=I or AAA^{-1}=A^{-1}A=I. Note that, all the square matrices are not invertible. If the square matrix has invertible matrix or non-singular if and only if its determinant value is non-zero. Moreover, if the square matrix A is not invertible or singular if and only if its determinant is zero.

$$[A] = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$[A]^{-1} = \frac{1}{det[A]} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

A square matrix that is not invertible is called singular or degenerate. A square matrix is singular if and only if its determinant is zero. Singular matrices are rare in the sense that if a

square matrix's entries are randomly selected from any finite region on the number line or complex plane, the probability that the matrix is singular is 0, that is, it will "almost never" be singular.

Non-square matrices (m-by-n matrices for which $m \neq n$) do not have an inverse. However, in some cases such a matrix may have a left inverse or right inverse. If A is m-by-n and the rank of A is equal to n ($n \leq m$), then A has a left inverse, an n-by-m matrix B such that $BA = \mathbf{I}_n$. If A has rank m ($m \leq n$), then it has a right inverse, an n-by-m matrix B such that $AB = \mathbf{I}_m$.

ONE-THIRD SIMPSON'S RULE WITH N-SEGMENTS

Simpson's Rule is a numerical method that approximates the value of a definite integral by using quadratic functions. This method is named after the English mathematician Thomas Simpson (1710−1761). Simpson's Rule is based on the fact that given three points, we can find the equation of a quadratic through those points.

SIMPSON'S RULE FORMULA

$$\text{Area} = \int_a^b f(x)\,dx$$

$$\approx \frac{\Delta x}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \ \ldots + 4y_{n-1} + y_n)$$

$$\text{where } \Delta x = \frac{b-a}{n}$$

The trapezoidal rule was based on approximating the integrand by a first order polynomial, and then integrating the polynomial over interval of integration. Simpson's 1/3 rule is an extension of Trapezoidal rule where the integrand is approximated by a second order polynomial.

We can re-write Simpson's Rule by grouping it as follows:

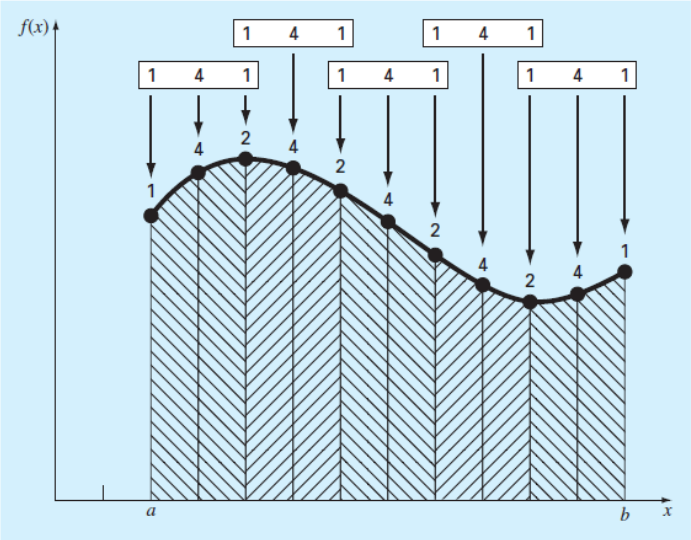
$$\int_a^b f(x)\,dx \approx \frac{\Delta x}{3}[y_0 + 4(y_1 + y_3 + y_5 + \ldots) + 2(y_2 + y_4 + y_6 + \ldots) + y_n]$$

This gives us an easy way to remember Simpson's Rule:

$$\int_a^b f(x)\,dx \approx \frac{\Delta x}{3}[\text{FIRST} + 4(\text{sum of ODDs}) + 2(\text{sum of EVENs}) + \text{LAST}]$$

COMPOSITE SIMPSON'S 1/3 RULE

The relative weights are depicted above the function values. Note that the method can be employed only if the number of segments is even.



SOFTWARE

MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building
- Math and computation
-

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation.

AppDesigner powered by MATLAB

App Designer is an interactive development environment for designing an app layout and programming its behavior. It provides a fully integrated version of the MATLAB® Editor and a large set of interactive UI components. It also offers a grid layout manager to organize your user interface, and automatic reflow options to make your app detect and respond to changes in screen size. It lets you distribute apps by packaging them into installer files directly from the App Designer toolstrip, or by creating a standalone desktop or web app (requires MATLAB Compiler™).

B. SCOPE AND LIMITATIONS OF THE PROGRAM

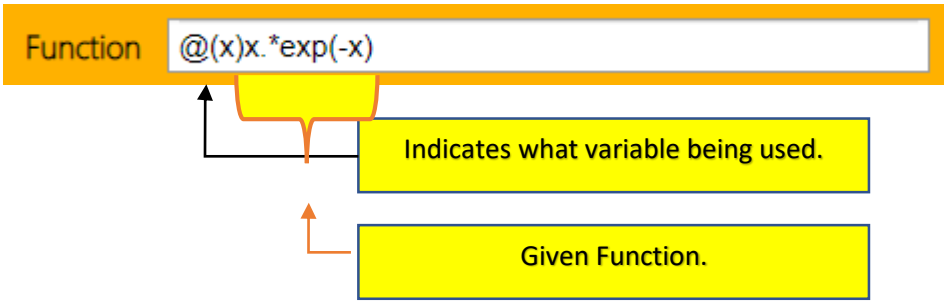
This program (SimpifyMi) about Matrix Inversion and 1/3 Simpson's Rule with n Segment aims to help the students, educators, and engineers to double-check their answers about the two numerical methods that have been mention. Specifically, the Simpify covers all definite integral that have limits of + or - infinity. On the other hand, the scope of the Mi program is for all.

C. Instructions on How to Use the Program per Numerical Method

1/3 Simpson's Rule with Multiple Segments

- **Launch Simpify,** select the simplify from the menu on the start page, the click launch. Simpify pop up window will open.
- **Enter your definite integral problem.**

1. Function, follow the format.



1.a) Declare what variable being used in the given function.

Example:

@(x)x.^3 **@(t)sin(t)**

@ (y)y+5

1.b Enter the Upper limit, Lower limit and the segments from the given Function

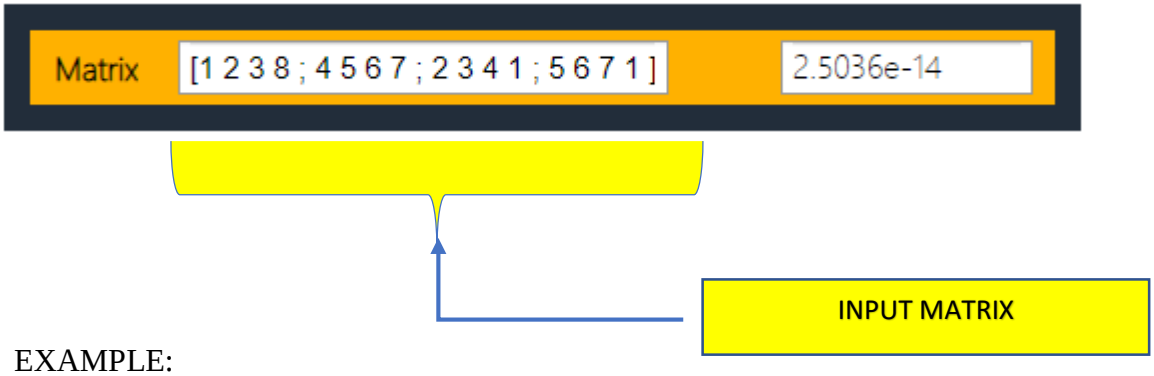
- **Click estimate,** it displays the graph, interval, estimation, exactness and the percent error.

Matrix Inversion

- **Launch Mi,** select the Mi from the menu on the start page, the click launch. Mi pop up window will open.

- **Enter your input matrix**

  1. Matrix, Follow the format.

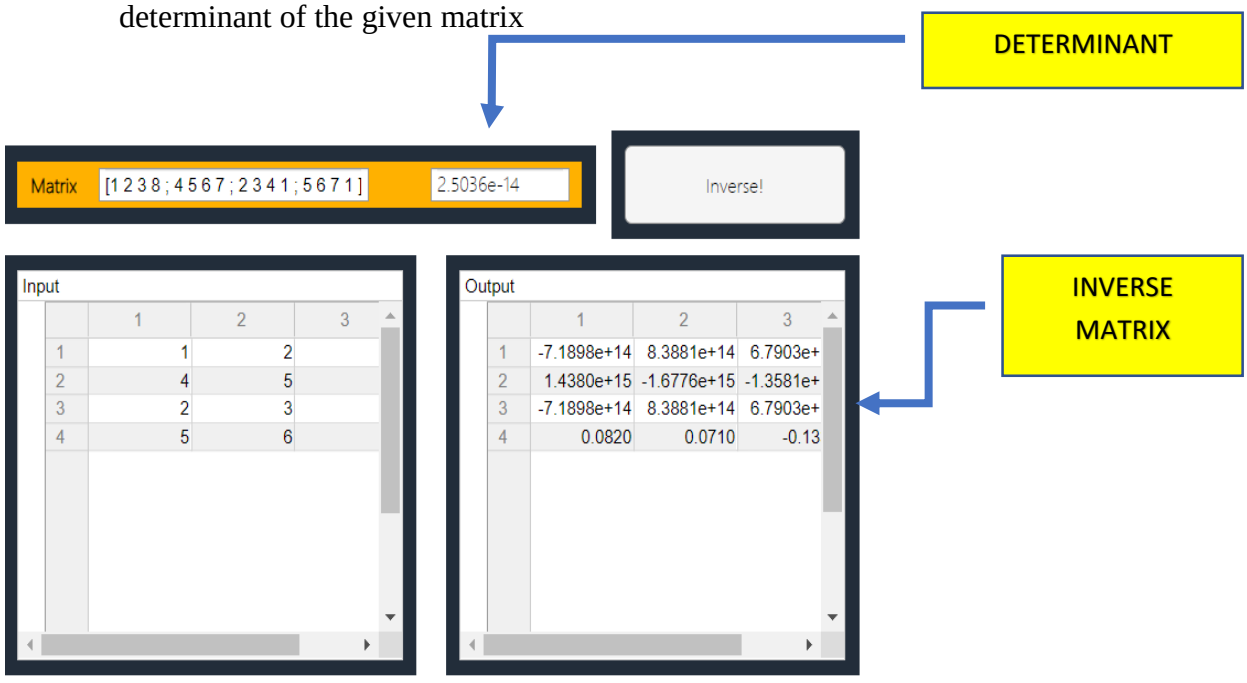


EXAMPLE:

[1 2 3 ; 4 5 6 ; 4 5 7] [4+3i 3 ; 2 2+2i]

[1 2 ; 3 4]

- **Click Inverse button,** it displays the input matrix, the inverse matrix and the determinant of the given matrix



D. ENCOUNTERED PROBLEMS AND SOLUTIONS

Pre-development

Initial thoughts & Planning

Given the during pre-community quarantine, along with the suspension of classes, the problem of having little to no knowledge about the numerical methods that was given nor utilizing MATLAB which is a computational language that is being utilized to teach numerical methods became most prominent.

- ***Problem 1: Knowledge deficit***

  Although matrix inversion and the utilization of MATLAB to quickly evaluate matrices were already taught, there was still a big *information* gap that must be bridged. Lacking knowledge on both Simpsons' method of estimating definite integrals, as well as the basic syntax and capabilities of MATLAB, which were the most prominent hurdles to overcome led to one and only solution. The only solution for lack of information is to study and practice the concepts well. Also, using the Programming Laboratory Manual given by the faculty significantly helped the learning stage of MATLAB basics that was utilized for the project. Another way to lessen development time is by dividing the workload between the developers. Through this method, the concepts can be split between people reducing precious development time while developing mastery over their specific field of choice.

Program Concept

Conceptualizing a program over the period of a pandemic is one hard task. Due to other subjects also making announcements for final requirements to solidify the end of the semester time was divided and progress of the software to a dramatic halt but soon follows, its re-emergence. Simpify was created using GUIDE, while Mi was made on App designer.

- ***Problem 2: GUIDE vs. App designer***

  MATLAB offers two-ways in order to create computation software. GUIDE was the pioneer of GUI building within the software. Its capabilities are simple and basic, it uses handles as the main function to call UI elements. But as counterweight to the good reliable simplicity, it lacks a number of elements. To add, it offers no UI customization and style at all. In general GUIDE was simple and plain, both in terms of its capabilities

and customization, it was made simple for simple things. On the other hand, App Designer was used to design apps which can be installed in MATLAB itself. It was two-steps higher from GUIDE. Handles were completely changed to “app.” to call UI elements, which is easy to remember. It is fairly customizable, and has a lot of potential for software development and was recommended over GUIDE. These upgrades became good optimization solutions to prevent problems to be encountered in the future. Also, the program was migrated from MATLAB version 2018b to 2019a.

Software Development

UI Layout/Design

- ***Problem 3: Migrating GUIDE .fig as App Designer .mlapp***

  As mentioned earlier Simpify was first conceptualized using GUIDE. Migrating and converting .fig files to .mlapp was one of the reasons why it was important to use a later version of MATLAB. The lack of extensions and toolboxes (incomplete installation) led to numerous fatal errors and crashes during the migration of Simpify. It cost a clean installation of MATLAB 2019a, unfortunately due to compatibility issues, Simpify was rebuilt from the ground up as a mlapp file.

- ***Problem 4: UI Overhaul***

  Switching to App designer allowed the developers to completely overhaul the User-Interface design of the program. But even so, the developers lack experience in designing a user-interface. Since the specific layout were not planned, it took a lot of time coming up with designs and color schemes that would fill the UI. Using extensive research and referencing, the developers come up with a design based on a logo (credited to the owner). Where its color schemes were used for Simpify and Matrix Inversion.

Code Implementation Challenges

- ***Problem 5: Data types***

  The first common mistake is mishandling data types. App designer elements require specific data types. Use numeric and text edit fields wisely, some UI elements can only accept a certain data type as value. For example, a text edit field containing numbers would not be evaluated because everything written on a text edit field is

classed as a string. Changing the edit field to numeric is one solution but the str2num() function can be used as well. Aside from str2num there are other kinds of functions and all have their specific utilizations.

- ***Problem 6: Displaying Figures***

  Displaying figures is a very tricky job. A full grasp of the Parent-Child concepts since the functions for visualization are different from normal. In order to plot graphs call a function, then locate the UI Figure where it will be displayed, followed by other layout options. Example is when plotting a graph. It uses this syntax plot(app.UIfigure,data). Whilst for tables you can utilize the uitable(app.UIfigure(can be parent class),"Data",data) and so on.

- ***Problem 7: Symbolic Package vs Function Handles***

  There are two ways of encoding variables in MATLAB. Symbolic is declarative, it requires you to declare the character that will be treated as a variable in the initial run of a function, whilst using Function handles give the flexibility of letting the user define the variable to be used. Simpify requires a function as an input, using Symbolic means that once the character was not declared as a variable during initialization, it can lead to an error. Example syms are x c v. These three letters are no longer variables but now symbolics. If an input of 4t is encoded, t will be considered a variable and prevent the program from working. In order to bring flexibility, function handles using anonymous variables was used. Eg. @(d)d+5d*2 where d is now an anonymous variable making the input a function handle.

- ***Problem 8: Value Changing Input***

  Adding callbacks for textbox gives real time update and can help visualize the data that is being encoded, a good examples is when encoding a matrix string as a MATLAB double can be updated to a matrix represented in tables that updates real time as the user inputs values, however this update is quite slow. Although advantageous, it will not work the way as intended, a solution can be a button that displays the value of input or no using the callback at all.

- ***Problem 9: Parent-Child Class Relationships***

  In plotting tabular data you can make use of panels to replace default UI elements such as axes and tables. A panel is grouping UI, that will make everything within in a subpanel, this the Parent-Child Class Relationships. Panels can be used as UI elements to plot any kind of figure rather than looking for a specific UI element.

- ***Problem 10: Path Setting Errors***

  Any errors on environment path can cause custom static images and other custom UI inputs to not load properly, errors messages such as matlab.ui will show paragraphs of error messages. To combat this, it is preferred to keep all files related to the project on one folder, and add that specific folder as a MATLAB path. Home>Set path>Add folder

Debugging

Parsing & Installation

- ***Problem 11: Compiling MATLAB to C++ library***

  Parsing the app to a C++ library is not possible because there are functions that only MATLAB can perform and cannot be compiled to C++. This means that the program will still be running in a MATLAB environment MATLAB Runtime when compiled as a standalone app. It is necessary for Runtime to be installed for the app to run. Another solution is compiling the program as a MATLAB app, which is much smaller in size, easy to parse and can be installed on a device with MATLAB installed.

- ***Problem 12: Installing as a MATLAB App***

  Along with the zip files for the program is a text file which contains instructions on how to run the app properly. The parsed .mlapp can only access the files necessary for the program as long as that window is open. In the case of this program which closes, the main app to switch functions, the system will not find the files and will load an incomplete app. The solution for this is already written for the text file.

Final Optimizations

This section contains minor fixes and optimizations to improve app functionality. As its pledge to look professional, it has to act professionally too.

- ***Window titling***

  How sharp the final output could look, the professional standard of the app can be broken at first read of the title bar “UI Figure”. Rename this to an app title before parsing.

- ***Resize function***

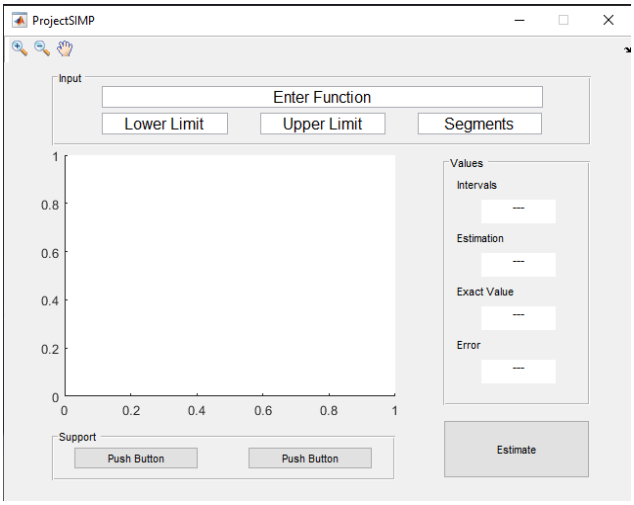
  There are UI elements that are non-scalable thus they will not stretch nor contract when the window is resized. In left enabled, it give the function to resize the app window and can ruin the layout.

- ***Scrollable***

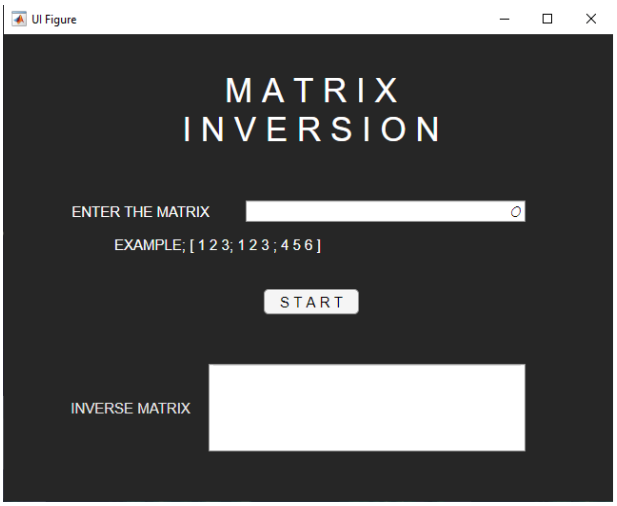
  When using Parent-Child Relationships be sure to enable scrollable. The default position of a figure plotted to a Parent holder is above. This problem occurs prominently with tables, and will leave nothing but an empty space instead of a figure visualization. Enable scrollable in panels.

- ***Editable Texts***

  When using edit fields as output boxes (outputs results) be sure to disable editing so the answer may not be altered or erased. It can cause loss of data and time.



Simpify GUIDE concept



Matrix Inversion App designer concept