Project 1 Report

Engineering Analysis and Computation I

Ahmed Jaheen - 900212943
Mina Yasser - 900214039
Mohamed ElFahla - 900192563

Submitted to Dr. Mostafa Youssef
This report is prepared for ENGR3202, section 1

THE AMERICAN
UNIVERSITY IN CAIRO

School of Science and Engineering
The American University in Cairo
12/07/2023

List of contributors

1. **Ahmed Jaheen**
   Contributions: Wrote ***main.m****,* ***newtonRaphson.m****,* ***CalculateF.m****,* ***CalculateJacobian.m****. Also, the latex code for this report and the report.*

2. **Mina Yasser**
   Contributions: Wrote ***gaussElimination.m****,* ***predictPlateSize.m****. Also, the report.*

3. **Mohamed ElFahla**
   Contributions: Wrote ***gaussJordan.m****,* ***plotRelativeErrors.m****,* ***plotPressureVsRadius.m****. Also, the report.*

Abstract

Numerical analysis has been around for quite some time now, and it is prevalent in our everyday lives and will surely be present in the future. **MATLAB** has been the leading software for mathematical and numerical analysis for a while, and it is evident in the enormous power and wide capabilities it provides users with. The project consists of a numerical analysis of Pressures and radii to calculate the constant K. The main software used in this project is **MATLAB** which was programmed to output respective entropy, specific volume, coefficient of expansion, and other values based on the value input by the user. As coding amateurs, we depended on the slides provided in the lectures and other sources for guidance when necessary. The code was divided into several functions to use as few computations as possible (in order not to exhaust the resources).

Background review of the problem

The problem involves determining the constants k_1, k_2, and k_3 in a non-linear equation approximating the amount of pressure needed to sink a circular plate into soft soil. This equation is based on experiments conducted on different scales with varying radii and pressures. The soft ground is assumed to be homogeneous and lies above a rigid base soil, with a known thickness D. The goal is to find the constants k_1, k_2, and k_3, which depend on the composition of the soil and the sinkage distance d, but do not depend on the plate radius. To solve this problem, three experiments are conducted using three different circular plates. The plate is pressed into the soil for each experiment until it sinks a distance d, and the corresponding pressure required is measured. These experiments result in three non-linear equations relating the pressures (p_1, p_2, p_3) to the plate radii (r_1, r_2, r_3). The software package aims to solve this system of equations using the Newton-Raphson method. It prompts the user to enter the iterations' radii, pressures, and stopping criterion. The package offers the flexibility to choose between using Gauss elimination with partial pivoting or Naïve Gauss-Jordan to solve the linear systems that arise during the Newton-Raphson iterations.

Additionally, the software calculates and plots the approximate relative error in estimating the constantsk_1, k_2, and k_3 as a function of the number of iterations. It reports the final values of the constants and plots the pressure as a function of the radius. Furthermore, the package provides a bonus feature that allows the user to predict the smallest size of a circular plate required to sustain a specific load F with a given sinkage distance d. This prediction is achieved using the bisection method, which iteratively narrows down the range of possible plate radii until the desired load is reached. In summary, the software package addresses the problem of determining the constants in the non-linear equation for the pressure-sinkage relationship in soft soil. It provides a comprehensive solution that includes solving equations, analyzing errors, reporting results, and predicting plate sizes based on desired loads.

Background on the numerical methods

In this project, we employ numerical methods to solve the system of equations and approximate the values of the constants in the pressure-sinkage relationship for objects sinking into soft soil. Two main numerical methods are utilized: the Newton-Raphson method for solving non-linear equations and methods for solving linear systems. The Newton-Raphson method is an iterative numerical technique used to find the roots of non-linear equations. In our case, it allows us to solve non-linear equations relating to the pressures and plate radii.

The method starts with an initial guess for the values of the constants and iteratively refines them to converge toward the correct values. At each iteration, the method updates the guesses based on the current estimates and the derivatives of the equations. The process continues until the desired level of accuracy is achieved.

$$f(x) = 0; x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{1}$$

We have two options to solve the linear systems that arise during the Newton-Raphson iterations: Gauss elimination with partial pivoting and Naïve Gauss-Jordan method.

The Gauss elimination method with partial pivoting is used to solve linear equation systems. It involves performing row operations on the augmented matrix of the system to transform it into an upper triangular form and then back-substituting to find the solutions. Partial pivoting is used to avoid numerical instability and improve the accuracy of the solutions for a system of linear equations:

$$\begin{aligned} a_{11}x_1 + a12x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a22x_2 + \ldots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + am2x_2 + \ldots + a_{mn}x_n &= b_m \end{aligned}$$

where a_{ij} represents the coefficients, x_i represents the variables, and b_i represents the constants, the Gaussian elimination method can be applied to solve the system.

The Gaussian elimination algorithm consists of the following steps:

(i) Create an augmented matrix by combining the coefficient matrix and the constants vector:

$$[A \mid b]$$

(ii) Perform row operations to transform the augmented matrix into row-echelon form. The row operations include:
   (a) Scaling a row by a nonzero scalar.
   (b) Swapping rows.
   (c) Adding or subtracting rows.

(iii) Continue the row operations until the augmented matrix is in row-echelon form, which means:
   (a) The leftmost nonzero entry in each row is 1 (leading coefficient).
   (b) The leading coefficient of each row is to the right of the leading coefficient of the row above it.
   (c) Rows consisting entirely of zeros are at the bottom.

(iv) Apply back-substitution to obtain the solutions. Start from the bottom row and solve for each variable by substituting the known values.

The resulting values of the variables $x_1, x_2, \ldots, x_n$ are the solutions to the system of linear equations.

The Naïve Gauss-Jordan method is another approach to solving linear systems. It applies row operations on the augmented matrix to convert it into reduced row-echelon form. This method directly obtains the solutions without the need for back-substitution for a system of linear equations:

$$\begin{aligned} a_{11}x_1 + a12x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a22x_2 + \ldots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + am2x_2 + \ldots + a_{mn}x_n &= b_m \end{aligned}$$

where a_{ij} represents the coefficients, x_i represents the variables, and b_i represents the constants, the Gaussian-Jordan elimination method can be applied to solve the system.

The Gaussian-Jordan elimination algorithm consists of the following steps:

(i) Create an augmented matrix by combining the coefficient matrix and the constants vector:

$$[A \mid b]$$

(ii) Perform row operations to transform the augmented matrix into reduced row-echelon form. The row operations include:
   (a) Scaling a row by a nonzero scalar.
   (b) Swapping rows.

(c) Adding or subtracting rows.

(iii) Continue the row operations until the augmented matrix is in row-echelon form, which means:
   (a) The leftmost nonzero entry in each row is 1 (leading coefficient).
   (b) The leading coefficient of each row is to the right of the leading coefficient of the row above it.
   (c) All other entries in the column containing the leading coefficient are zero.

(iv) Apply back-substitution to obtain the solutions. Start from the bottom row and solve for each variable by substituting the known values.

The resulting values of the variables $x_1, x_2, \ldots, x_n$ are the solutions to the system of linear equations.

In our software package, the user has the flexibility to choose between these two methods for solving the linear systems that arise during the Newton-Raphson iterations. The choice depends on factors such as computational efficiency and numerical stability.

Additionally, the bisection method is utilized as a bonus feature to predict the minimum size of a circular plate required to sustain a specific load with a desired sinkage distance. The bisection method is an iterative technique that repeatedly divides an interval into two equal subintervals and determines in which subinterval the root lies. It is particularly useful when dealing with continuous functions and finding approximate solutions within a given interval.

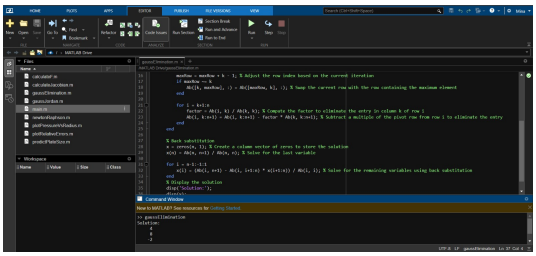
Overall, the numerical methods used in this project, including the Newton-Raphson method, Gauss elimination with partial pivoting, Naïve Gauss-Jordan method, and the bisection method, enable us to solve the system of equations accurately, approximate the values of the constants, and predict plate sizes for desired loads. These methods form the foundation of the software package, providing efficient and reliable numerical solutions to the problem at hand.

Validation for the functions

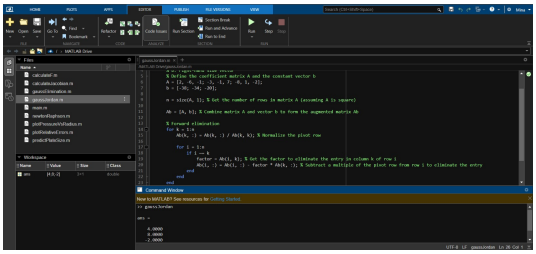
Firstly, we will test our gauss elimination with partial pivoting function and gauss jordan function in compare with the output values from the embedded MATLAB function for the following augmented matrix:

$$\left(\begin{array}{ccc|c} 2 & -6 & -1 & -38 \\ -3 & -1 & 7 & -34 \\ -8 & 1 & -2 & -20 \end{array}\right)$$

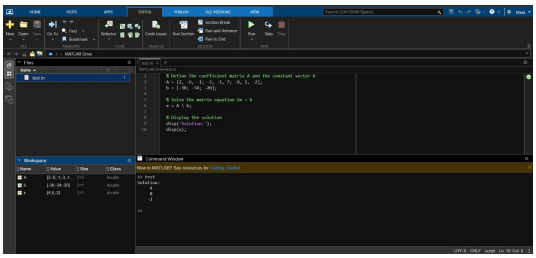
1. **Gauss elimination method with partial pivoting function:**



2. **Naïve Gauss-Jordan method function:**



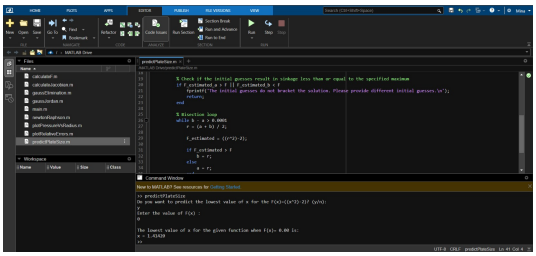
3. **Embedded MATLAB function:**



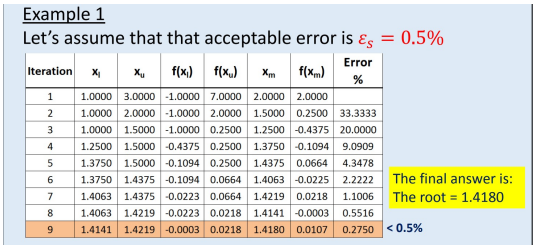
Next, we will test our bisection method function and compare it to an already solved example in the lecture notes with initial guesses 1 and 2 for finding the roots of the following function:

$$f(x) = x^2 - 2 \tag{2}$$

1. **Bisection method function:**



2. **Solved example from lecture notes:**



Example 1

Let's assume that that acceptable error is $\varepsilon_s = 0.5\%$

Iteration	x_l	x_u	$f(x_l)$	$f(x_u)$	x_m	$f(x_m)$	Error %
1	1.0000	3.0000	-1.0000	7.0000	2.0000	2.0000	
2	1.0000	2.0000	-1.0000	2.0000	1.5000	0.2500	33.3333
3	1.0000	1.5000	-1.0000	0.2500	1.2500	-0.4375	20.0000
4	1.2500	1.5000	-0.4375	0.2500	1.3750	-0.1094	9.0909
5	1.3750	1.5000	-0.1094	0.2500	1.4375	0.0664	4.3478
6	1.3750	1.4375	-0.1094	0.0664	1.4063	-0.0225	2.2222
7	1.4063	1.4375	-0.0223	0.0664	1.4219	0.0218	1.1006
8	1.4063	1.4219	-0.0223	0.0218	1.4141	-0.0003	0.5516
9	1.4141	1.4219	-0.0003	0.0218	1.4180	0.0107	0.2750

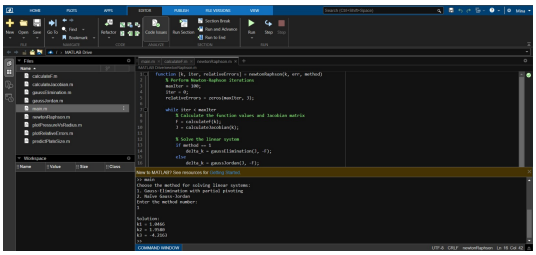
The final answer is:
The root = 1.4180

< 0.5%

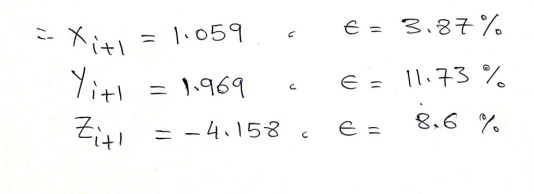
Finally, we will test our newton raphson method function and compare it to an already solved example in the lecture notes with initial guesses $(1.1, 2.2, -3.8)$ with only 1 iteration for finding the roots of the following equations:

$$x^2y + 10xz = -42$$
$$x + 10xy^2 + z = 37$$
$$yz + 2xy + x^2 = -3$$

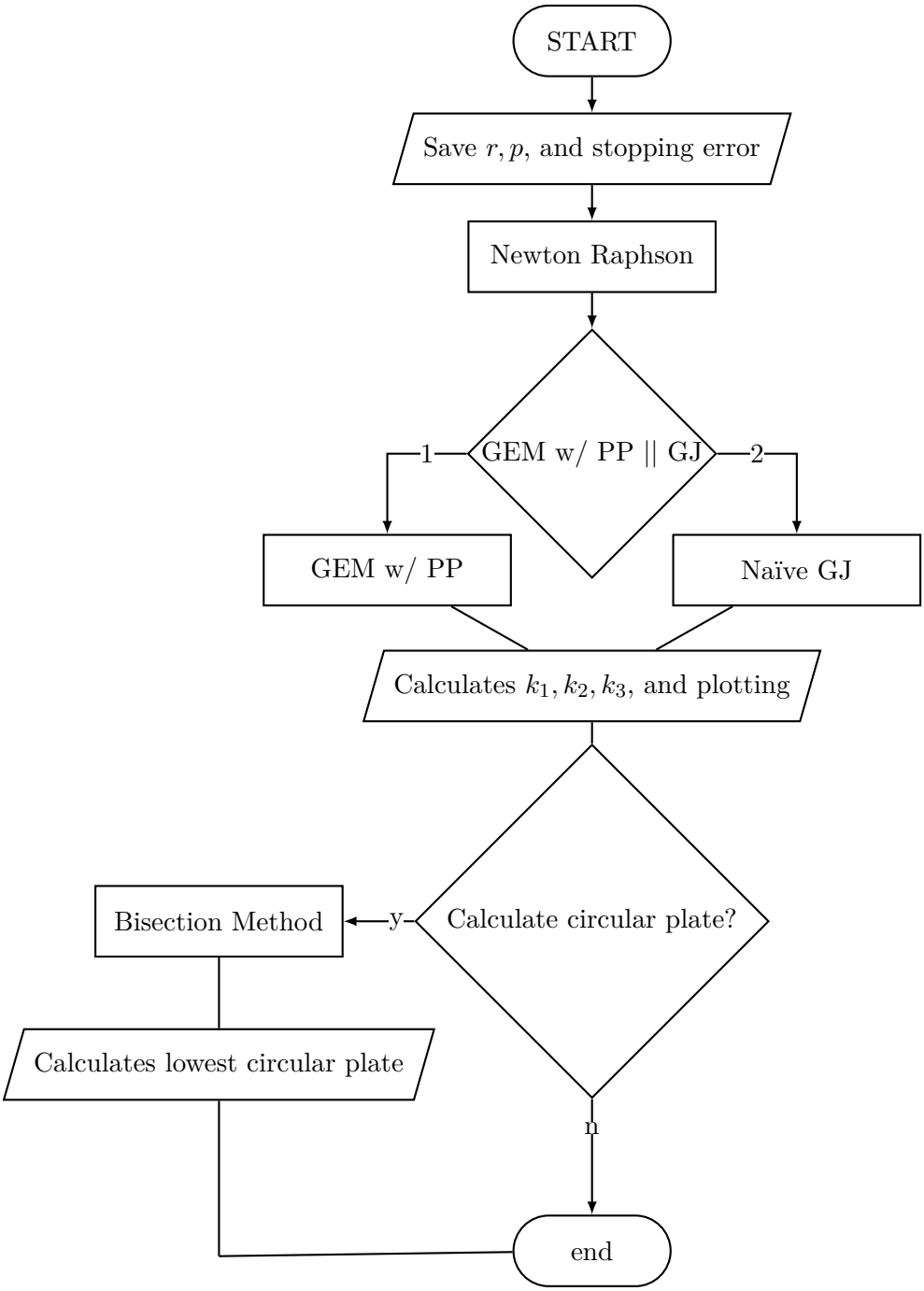
1. **Newton Raphson method function:**



2. **Solved example from lecture notes:**



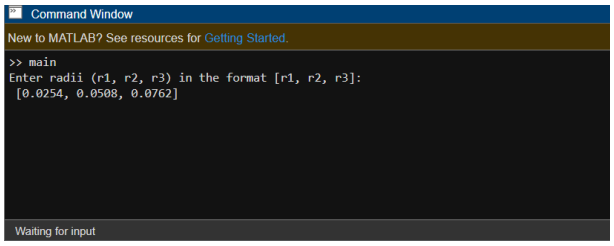
Graphical flow chart for the code



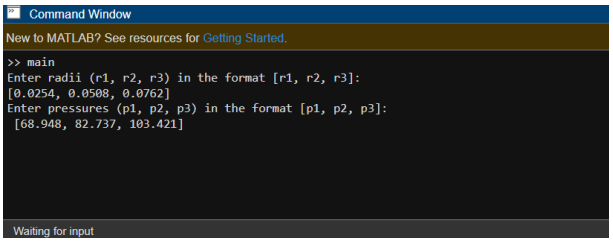
User Guide

The project is written in $MATLAB$, allowing you to compile it using $MATHWORKS$ online or offline platform. Once the compilation process is complete, the program will be initiated as following:

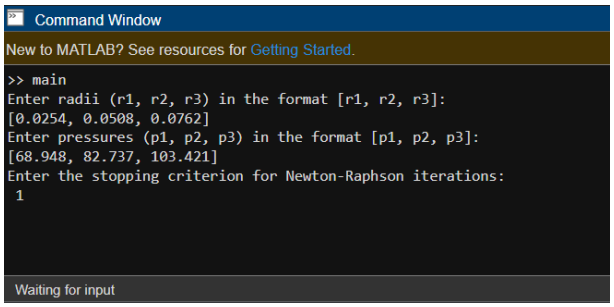
1. Firstly, the user has to write the radii in the following format $[r_1, r_2, r_3]$.



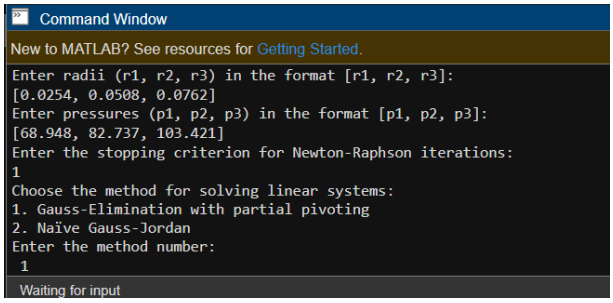
2. Secondly, the user has to write the pressures in the following format $[p_1, p_2, p_3]$.



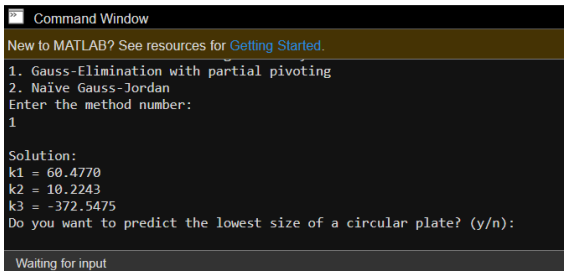
3. Thirdly, the user has to input stopping error for newton-raphson (preferably less than 3).

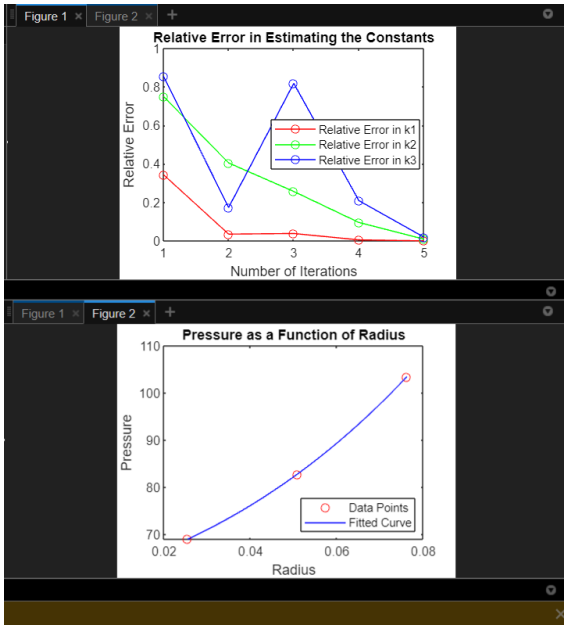


4. Fourthly, the user has to decide which method he would like to work with the linear equations output from newton-raphson, just enter 1 or 2.

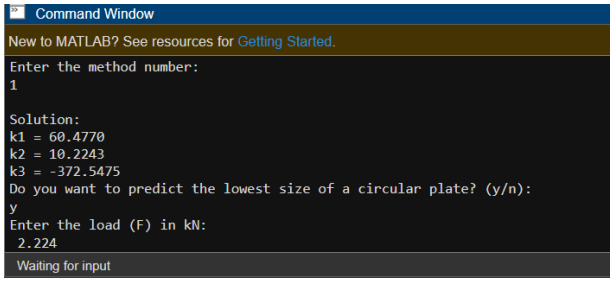


5. Fifthly, the code will output the k_1, k_2, and k_3 in a friendly way to the user. Also, it will output 2 plots; one for approximate relative error as a function of the number of iterations, the other the pressure as a function of the radius r.

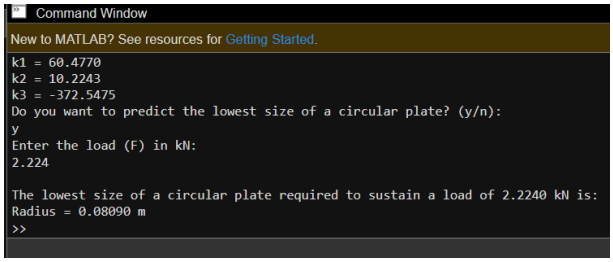




6. Next, the user will be asked if they are interested in predicting the lowest size of a circular plate that would be required to sustain a load F on this soil with a sinkage less than or equal d. y will proceed while n will exit the program. If y, the user has to input the value of F.



7. Finally, the user will receive the value of the lowest size of a circular plate, and the program will terminate.



Difficulties and sources of error

The code is relatively advanced to our level in programming, so we faced many difficulties that we resolved by brainstorming and using MathWorks's official documentations. When faced by error, we implemented meticulous error tracing and professional debugging manuals that proved useful in identifying the error and resolving the issue. Also, there was a problem with choosing the initial guess as it is not given to us and by choosing a random initial guess one time, it may result in an error as the answer may be one of a million answers that will cause the error. Also, the number of iterations used is quite small due to our laptops' weaknesses or low standers, which may cause the code to process for a long time. However, after quite some effort, the code was written at the end and ran smoothly and accurately.

Conclusion

In this project, we developed a MATLAB software package to solve a problem related to estimating the constants k_1, k_2, and k_3 in a non-linear equation using the Newton-Raphson method. The problem involved determining the pressure required to sink circular plates of different radii into soft soil.

The software package was designed to offer flexibility to the user by allowing them to input the radii and pressures and choose the stopping criterion and the method for solving linear systems (either Gauss elimination with partial pivoting or Gauss-Jordan).

We successfully implemented the Newton-Raphson method to solve the non-linear equations and calculate the approximate values of the constants k_1, k_2, and k_3 constants. The software also provided visualizations of the relative error in estimating the constants as a function of the number of iterations and plotted the pressure as a function of the radius.

Additionally, we included a bonus feature that allowed the user to predict the smallest size of a circular plate required to sustain a given load on the soil with a specific sinkage. The bisection method was employed to calculate the radius of the plate, and initial guesses were obtained using the pressure-radius relationship.

Overall, this software package provides an efficient and user-friendly tool for estimating the constants k_1, k_2, and k_3 based on experimental data. It can be used in various engineering applications that involve the interaction of soft soil with circular plates, such as foundation design and geotechnical analysis.

Bibliography

[1] Dr. Mostafa Youssef. (2023). *Lecture Notes*.

[2] The MathWorks Inc. (2023). *MATLAB Onramp, Natick, Massachusetts: The MathWorks Inc.*

[3] The MathWorks Inc. (2023). *Solving Nonlinear Equations with MATLAB, Natick, Massachusetts: The MathWorks Inc.*

[4] OpenAI. (2023). *ChatGPT* (June 18 version) [Large language model].