Case Study

On

Development of a General Linear Least Squares Solver in MATLAB

(Least squares fitting using normal equation)

Bachelor of Technology

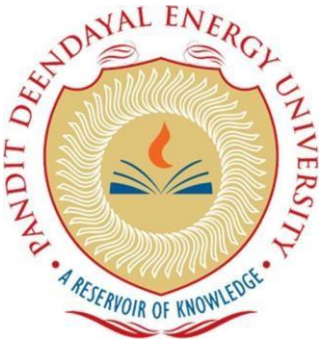
in

Mechanical Engineering

By

Mihir G. Gohil

(Roll. No: 23BME019)



School of Technology

Pandit Deendayal Energy University

Gandhinagar – 382426, Gujarat, India

February - 2026

Declaration

I hereby declare that the case study titled:
"Development of a General Linear Least Squares Solver in MATLAB with Application to Hooke's Law" is an original work carried out by me as part of the course requirements. The work presented in this report has been completed by me under the prescribed academic guidelines.
The results and analysis presented in this report are based on my own implementation in MATLAB. Any references used have been properly acknowledged.
I confirm that this work has not been submitted elsewhere for any other academic evaluation.

Mihir G. Gohil

Acknowledgements

I would like to express my sincere gratitude to our course instructor for providing the opportunity to undertake this case study on the Least Squares Method using MATLAB.
I am thankful for the guidance and support provided during the development of this project. This case study helped me gain practical understanding of linear regression, numerical optimization, and the application of mathematical concepts in engineering problems.
I also acknowledge the use of MATLAB software and relevant academic references that assisted in completing this work successfully.

ABSTRACT

In engineering and scientific analysis, many relationships between physical variables can be approximated using linear models. However, experimental measurements often contain errors that prevent data points from lying perfectly on a straight line. The Least Squares Method provides a systematic approach for determining the best-fit linear relationship by minimizing the total squared error between observed and predicted values.

This project presents the development of a general-purpose linear least squares solver in MATLAB capable of analyzing any dataset that follows a linear model of the form y=mx+cy = mx + cy=mx+c. The solver numerically determines optimal slope and intercept values by minimizing the sum of squared residuals. To validate the functionality of the solver, experimental load-extension data based on Hooke's Law was used as a practical example. The results demonstrate that the developed solver accurately determines linear parameters and can be applied to a wide range of engineering problems.

Table of Contents

1. INTRODUCTION

Linear relationships are fundamental in engineering and physical sciences. Many systems exhibit approximately linear behaviour within certain operating ranges. Examples include:

- Hooke's Law in spring systems
- Ohm's Law in electrical circuits
- Stress-strain relationship in elastic materials
- Calibration curves in sensors

In practical experiments, measured data rarely aligns perfectly with theoretical models due to:

- Measurement uncertainty
- Instrument precision limits
- Environmental disturbances
- Human error

To obtain a reliable mathematical representation of such data, regression techniques are employed. The Least Squares Method is one of the most widely used techniques for fitting linear models to experimental data.

The objective of this project is to develop a general MATLAB-based linear least squares solver and demonstrate its functionality using spring load-extension data.

2. MATHEMATICAL THEORY OF LEAST SQUARES

PAGE NO.
DATE:

Least Squares Derivation for a Simple Linear Regression Model

1. Assumption: Linear Model

We assume a linear relationship between the independent variable x and the dependent variable y:

$$y = mx + c$$

where: m slope of the regresion line
- c = intercept
- x_i = observed input values
- y_i = observed output values
- $\hat{y}_i$ = predicted output values

2. Predicted Value

For each data point x_i, the predicted value is:

$$y_i = mx_i + c$$

The residual r_i is defined as the difference between the observed and predicted values:

$$r_i = y_i - \hat{y}_i$$

Substituting for $\hat{y}_i$:

$$= r_i = y_i - (mx_i + c)$$

4. Objective Function: Sum of Squared Residuals

To determine the best-fit line, we minimize the total squared residual:

$$S = \sum_{i=1}^{n} r_i^2$$

Substituting r_i:

$$= S = \sum_{i=1}^{n} (y_i - (mx_i + c))^2$$

3. DEVELOPMENT OF GENERAL MATLAB SOLVER

The developed MATLAB program performs the following operations:

1. Accepts number of data points
2. Accepts any set of x and y values
3. Assumes linear model y = mx + c
4. Computes total squared error S for different values of m and c
5. Selects the values that produce minimum S
6. Displays equation and generates graphical plots

The solver uses a brute-force numerical approach to evaluate multiple combinations of slope and intercept values.

Although computationally slower than matrix-based methods, this approach provides conceptual clarity and ease of understanding.

4. APPLICATION EXAMPLE: HOOKE'S LAW

To validate the general solver, experimental data from a spring system was used.

According to Hooke's Law:

$F = kx$

F = Applied load

x = Extension

k = Spring stiffness

Since real measurements contain slight errors, the relation is written as:

$F = kx + c$

4.1 Experimental Dataset

Load (N)	Extension (mm)
10	1.1
20	2.1
30	3.1
40	4.2
50	5.0
60	6.2

5. MATLAB Implementation

5.1 Program Code

```
clc;
clear;
close all;

disp("================================================");
disp("   SIMPLE LEAST SQUARES LINE FITTING PROJECT   ");
disp("================================================");



n = input("Enter the number of data points: ");

x = zeros(1,n);
y = zeros(1,n);

disp("Enter the values of x and y one by one:");

for i = 1:n
    fprintf("\nPoint %d\n", i);
    x(i) = input("Enter x value: ");
    y(i) = input("Enter y value: ");
end


best_m = 0;
best_c = 0;
min_S  = inf;

for m = -5:0.01:5
    for c = -10:0.1:10


        S = 0;

        for i = 1:n
            y_pred = m*x(i) + c;
            error  = y(i) - y_pred;
            S = S + error^2;
        end
```

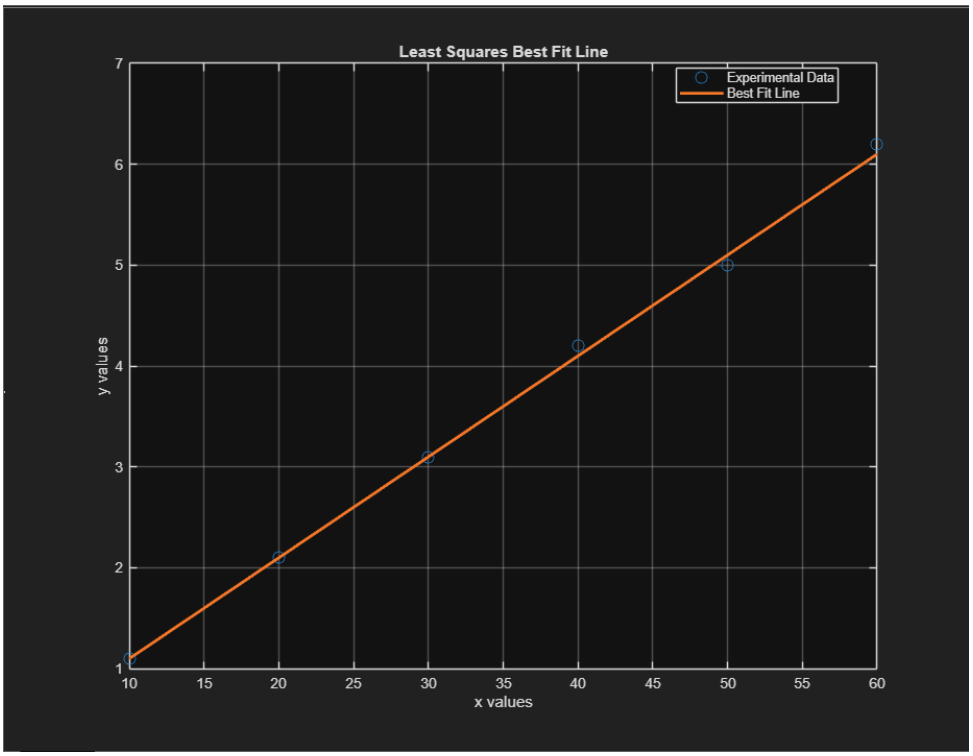
```
40
41
42          if S < min_S
43              min_S = S;
44              best_m = m;
45              best_c = c;
46          end
47      end
48  end
49
50
51  disp("-----------------------------------------------");
52  disp("Best Fit Line Found Using Least Squares Method:");
53  fprintf("Slope (m)     = %.4f\n", best_m);
54  fprintf("Intercept (c) = %.4f\n", best_c);
55  fprintf("Equation: y = %.4f*x + %.4f\n", best_m, best_c);
56  fprintf("Minimum Error S = %.6f\n", min_S);
57  disp("-----------------------------------------------");
58
59
60  y_fit = best_m*x + best_c;
61
62
63  figure;
64  plot(x, y, 'o', 'MarkerSize', 8);
65  hold on;
66  plot(x, y_fit, '-', 'LineWidth', 2);
67
68  xlabel('x values');
69  ylabel('y values');
70  title('Least Squares Best Fit Line');
71  legend('Experimental Data', 'Best Fit Line');
72  grid on;
73
74
75  residuals = y - y_fit;
76
77  figure;
78  stem(x, residuals, 'filled');
79
80  xlabel('x values');
81  ylabel('Residual Error (y - yfit)');
82  title('Residual Error Plot');
83  grid on;
```

5.2 MATLAB Output

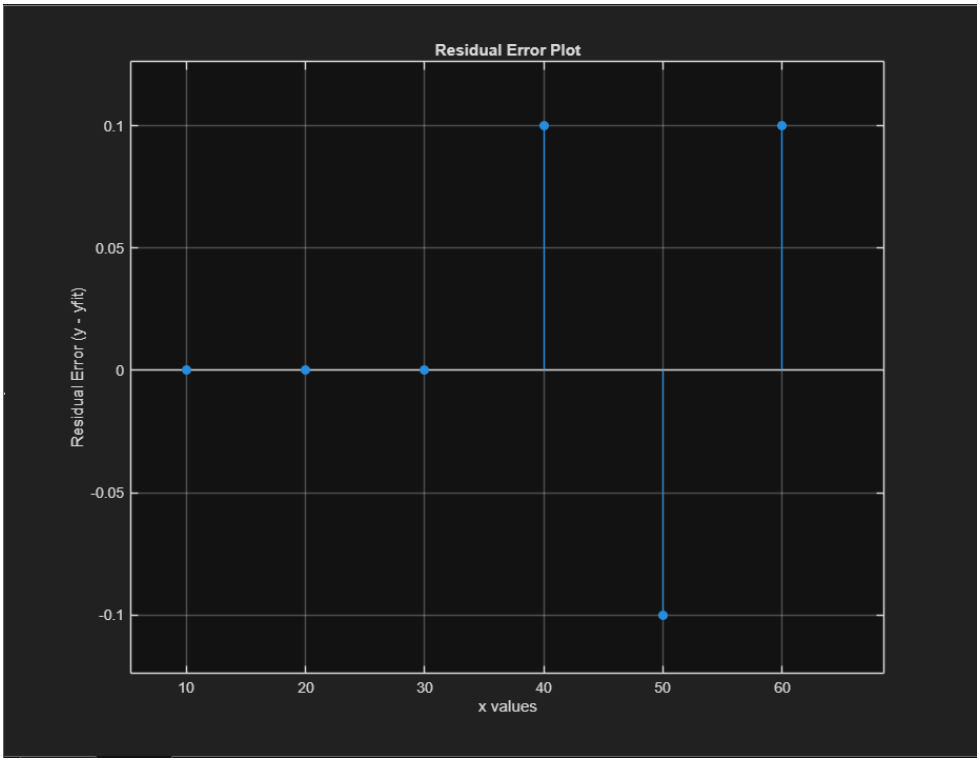
```
================================================
   SIMPLE LEAST SQUARES LINE FITTING PROJECT
================================================
Enter the number of data points: 6
Enter the values of x and y one by one:

Point 1
Enter x value: 10
Enter y value: 1.1

Point 2
Enter x value: 20
Enter y value: 2.1

Point 3
Enter x value: 30
Enter y value: 3.1

Point 4
Enter x value: 40
Enter y value: 4.2

Point 5
Enter x value: 50
Enter y value: 5.0

Point 6
Enter x value: 60
Enter y value: 6.2
------------------------------------------------
Best Fit Line Found Using Least Squares Method:
Slope (m)     = 0.1000
Intercept (c) = 0.1000
Equation: y = 0.1000*x + 0.1000
Minimum Error S = 0.030000
------------------------------------------------
```

6. Graphical Results

Plot 1: Least Squares Best Fit Line



Plot 2: Residual Error Plot



7. DISCUSSION

The developed solver successfully determines linear parameters for any dataset following the model $y = mx + c$.

The example using Hooke's Law demonstrates:

- Accurate determination of stiffness
- Small residual errors
- Effective minimization of squared error

Applications of the General Solver

The developed program can be used in:

- Stress-strain analysis
- Voltage-current characterization
- Calibration of instruments
- Material testing
- Experimental data approximation

Limitations

- Brute-force method is computationally intensive
- Accuracy depends on step size selection
- Applicable only to linear relationships

For large datasets, matrix-based or QR-decomposition methods are recommended.

8. CONCLUSION

A general linear least squares solver was successfully developed using MATLAB. The solver determines optimal slope and intercept values by minimizing total squared error. Hooke's Law was used as a validation example to demonstrate practical application.

The results confirm that the solver accurately models linear systems and can be extended to various engineering problems requiring linear approximation.