Problem Statement:

Understanding weather patterns and trends is essential for various applications such as agriculture, disaster management, and climate research. This project will involve analyzing and visualizing weather data to uncover insights and trends. By leveraging MATLAB's powerful data analysis and visualization capabilities, you will develop a comprehensive understanding of how to handle real-world data.

Objective:

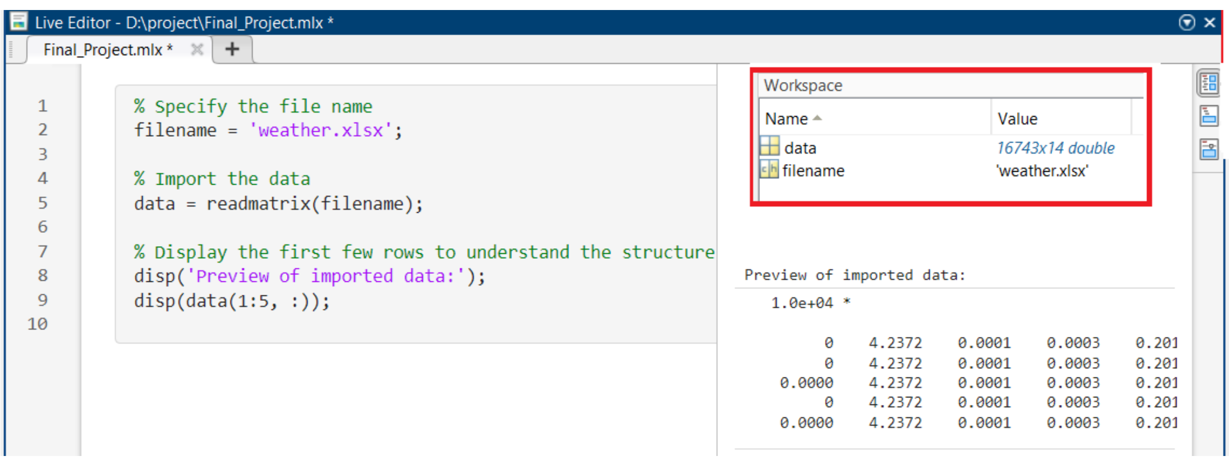
● To apply MATLAB fundamentals learned in previous assignments to a real-world data analysis project.
● To import, process, analyze, and visualize weather data using MATLAB.
● To develop a comprehensive report summarizing the findings.

Tasks to be Performed:

Data Acquisition and Preparation:

● **Task 1.1:** *Import weather data from a CSV file containing historical weather records (e.g., temperature, humidity, precipitation, wind speed).*
○ *Use the read matrix function to import data.*
○ *Inspect the data and handle any missing values or anomalies*.

Import weather data in matlab live script:



Red boxes are the data of weather import from a Excel sheet

Inspect the data:

Inspection of data determining the no. of rows and column are appeared from that excel sheet in matlab. Also the number of missing elements are reported in this code and see the result in display of live script from screen shot view.

Inspect the data

```
% Check for the size of the dataset
disp(['Data size: ', num2str(size(data))]);

% Use `summary` or manually check the range of values (if ap|
disp('Summary statistics:');
disp(['Min: ', num2str(min(data, [], 'all'))]);
disp(['Max: ', num2str(max(data, [], 'all'))]);

% Check for NaN values (missing values)
numMissing = sum(isnan(data), 'all');
disp(['Number of missing values: ', num2str(numMissing)]);
```

```
         0    4.2372    0.0001    0.0003    0.2016
    0.0000    4.2372    0.0001    0.0003    0.2016
         0    4.2372    0.0001    0.0003    0.2016
    0.0000    4.2372    0.0001    0.0003    0.2016

Data size: 16743     14

Summary statistics:
Min: -35
Max: 42736

Number of missing values: 66972
```

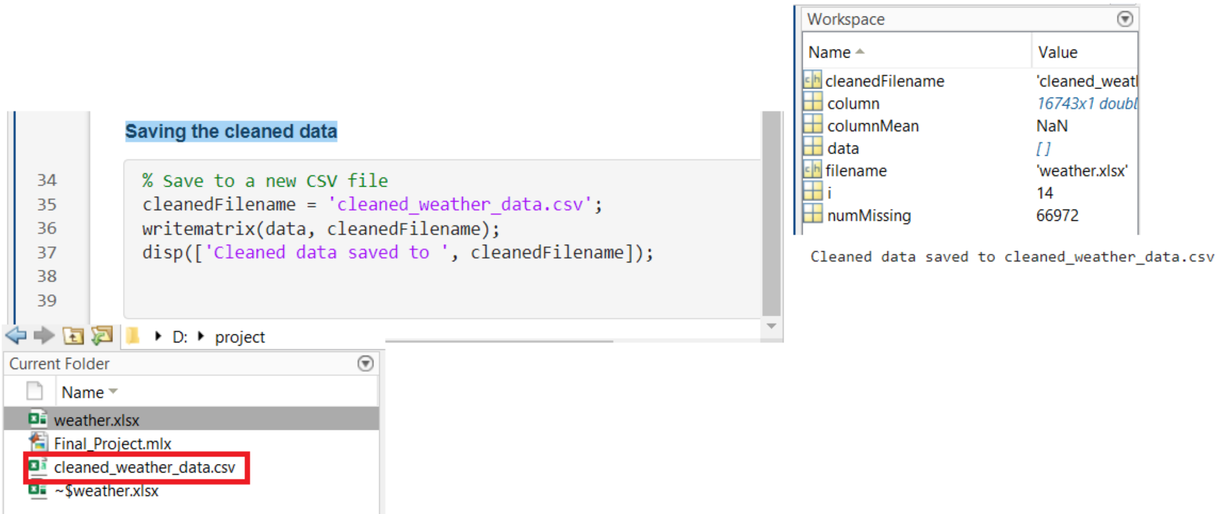
Handling the all rows and column missing values import from the excel sheet.

Handle Missing Values

```
% Replace NaN values with the mean of their respective colum
for i = 1:size(data, 2)
    column = data(:, i);
    if any(isnan(column))
        columnMean = mean(column, 'omitnan');
        column(isnan(column)) = columnMean;
        data(:, i) = column;
    end
end
disp('Missing values replaced with column means.');
% Remove rows containing NaN values
data = data(~any(isnan(data), 2), :);
disp('Rows with missing values removed.');
```

```
Data size: 16743     14
Summary statistics:
Min: -35
Max: 42736
Number of missing values: 66972

Missing values replaced with column means.

Rows with missing values removed.
```

Saving the cleaned data files code are generated. This is not so important but still do for cache cleared from the excel sheet.



- **Task 1.2:** *Organize the data into appropriate MATLAB data structures.*
  - *Use tables or matrices to store and manipulate the data.*
  - *Extract relevant columns for analysis.*

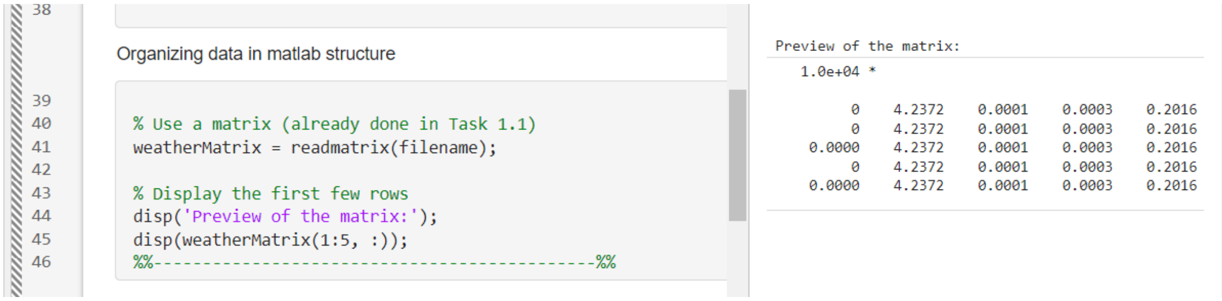
Organizing data in matlab structure and displaying the result from the import excel file mainly temperature, humidity precipitation and wind speed.

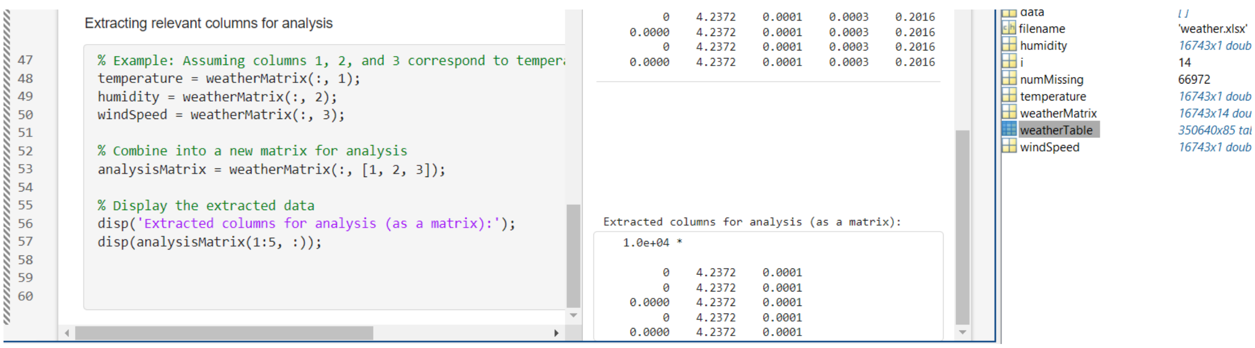
```
% Use a matrix (already done in Task 1.1)
weatherMatrix = readmatrix(filename);


% Display the first few rows
disp('Preview of the matrix:');
disp(weatherMatrix(1:5, :));
%%---------------------------------------------%%
```

Extracting relevant columns for analysis

```
% Example: Assuming columns 1, 2, and 3 correspond to temperature, humidity, and
wind speed
temperature = weatherMatrix(:, 1);
humidity = weatherMatrix(:, 2);
windSpeed = weatherMatrix(:, 3);


% Combine into a new matrix for analysis
analysisMatrix = weatherMatrix(:, [1, 2, 3]);


% Display the extracted data
```

```
disp('Extracted columns for analysis (as a matrix):');
disp(analysisMatrix(1:5, :));
```





Basic Data Analysis:

Task 2.1: *Calculate the basic statistical measures for each weather parameter (Mean, Median, Standard division).*

Use Build-in MATLAB functions for statistical analysis.

Task 2.2: Identify trends and patterns in the data.

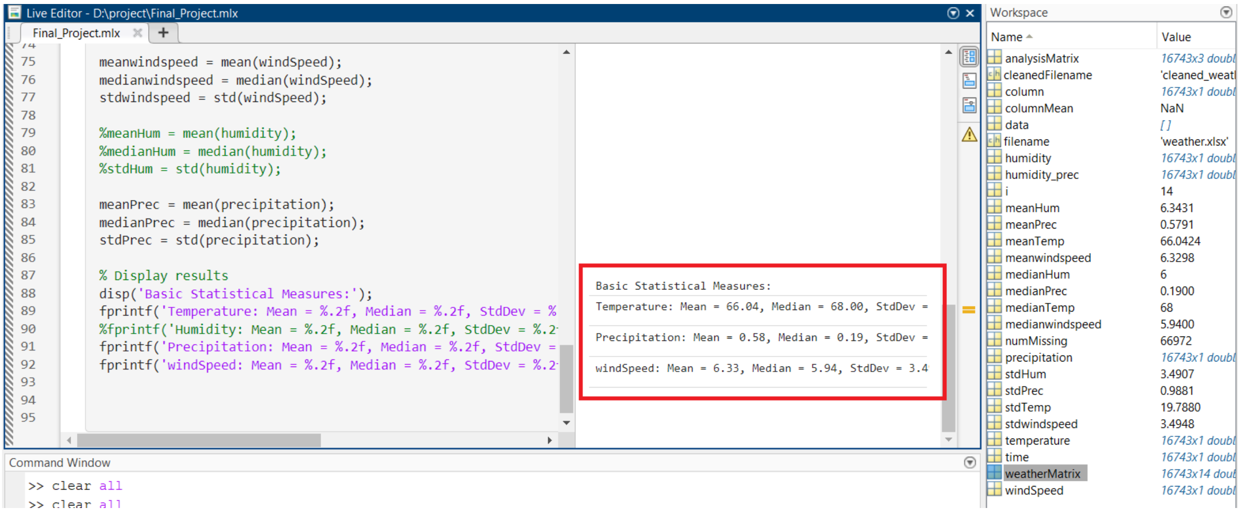
○ *Plot time series graphs for temperature, humidity, and precipitation.*
○ *Use moving averages to smooth the data and highlight trends.*

TASK 2.1: Calculate basic statistical measure

Code for basic statistical measure.

```
% Assuming 'weatherMatrix' contains the data


% Extract columns
         % Time (e.g., serial dates or indices)
temperature = weatherMatrix(:, 11);  % Temperature
```

```
     % Humidity
precipitation = weatherMatrix(:, 1);% Precipitation
windSpeed = weatherMatrix(:, 14);


% Calculate basic statistics
meanTemp = mean(temperature);
medianTemp = median(temperature);
stdTemp = std(temperature);


meanwindspeed = mean(windSpeed);
medianwindspeed = median(windSpeed);
stdwindspeed = std(windSpeed);


%meanHum = mean(humidity);
%medianHum = median(humidity);
%stdHum = std(humidity);


meanPrec = mean(precipitation);
medianPrec = median(precipitation);
stdPrec = std(precipitation);


% Display results
disp('Basic Statistical Measures:');
fprintf('Temperature: Mean = %.2f, Median = %.2f, StdDev = %.2f\n', meanTemp,
medianTemp, stdTemp);
%fprintf('Humidity: Mean = %.2f, Median = %.2f, StdDev = %.2f\n', meanHum,
medianHum, stdHum);
fprintf('Precipitation: Mean = %.2f, Median = %.2f, StdDev = %.2f\n', meanPrec,
medianPrec, stdPrec);
fprintf('windSpeed: Mean = %.2f, Median = %.2f, StdDev = %.2f\n', meanwindspeed,
medianwindspeed, stdwindspeed);
```

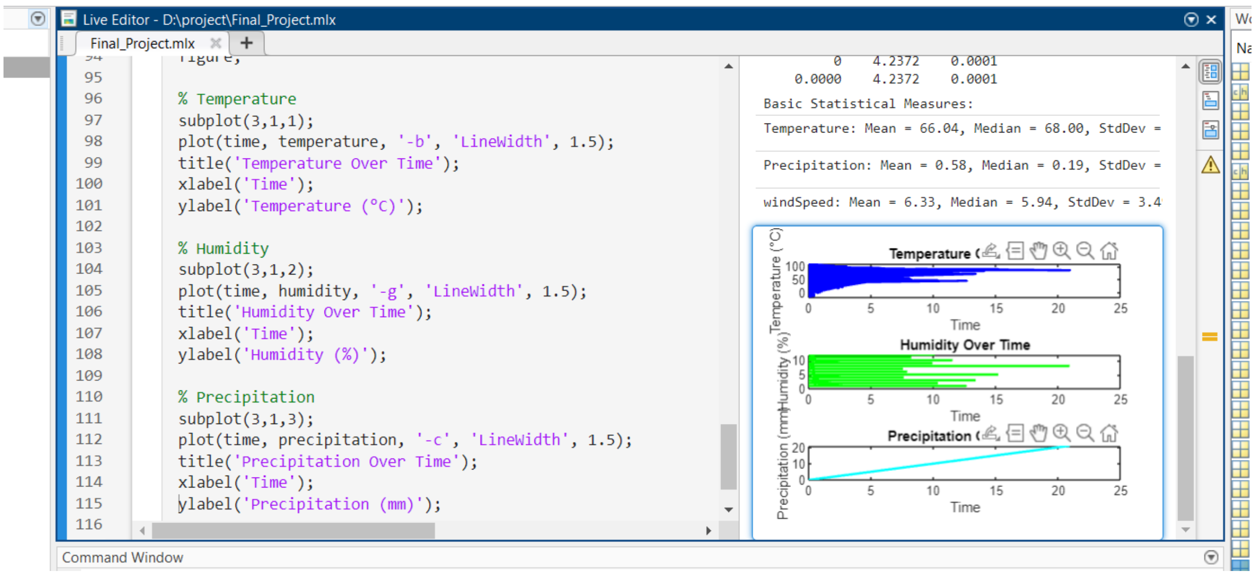


Code for time plot series

Plot for TASK 2.2

```
% Plot each parameter
figure;


% Temperature
subplot(3,1,1);
plot(time, temperature, '-b', 'LineWidth', 1.5);
title('Temperature Over Time');
xlabel('Time');
ylabel('Temperature (°C)');


% Humidity
subplot(3,1,2);
plot(time, humidity, '-g', 'LineWidth', 1.5);
title('Humidity Over Time');
xlabel('Time');
ylabel('Humidity (%)');


% Precipitation
```

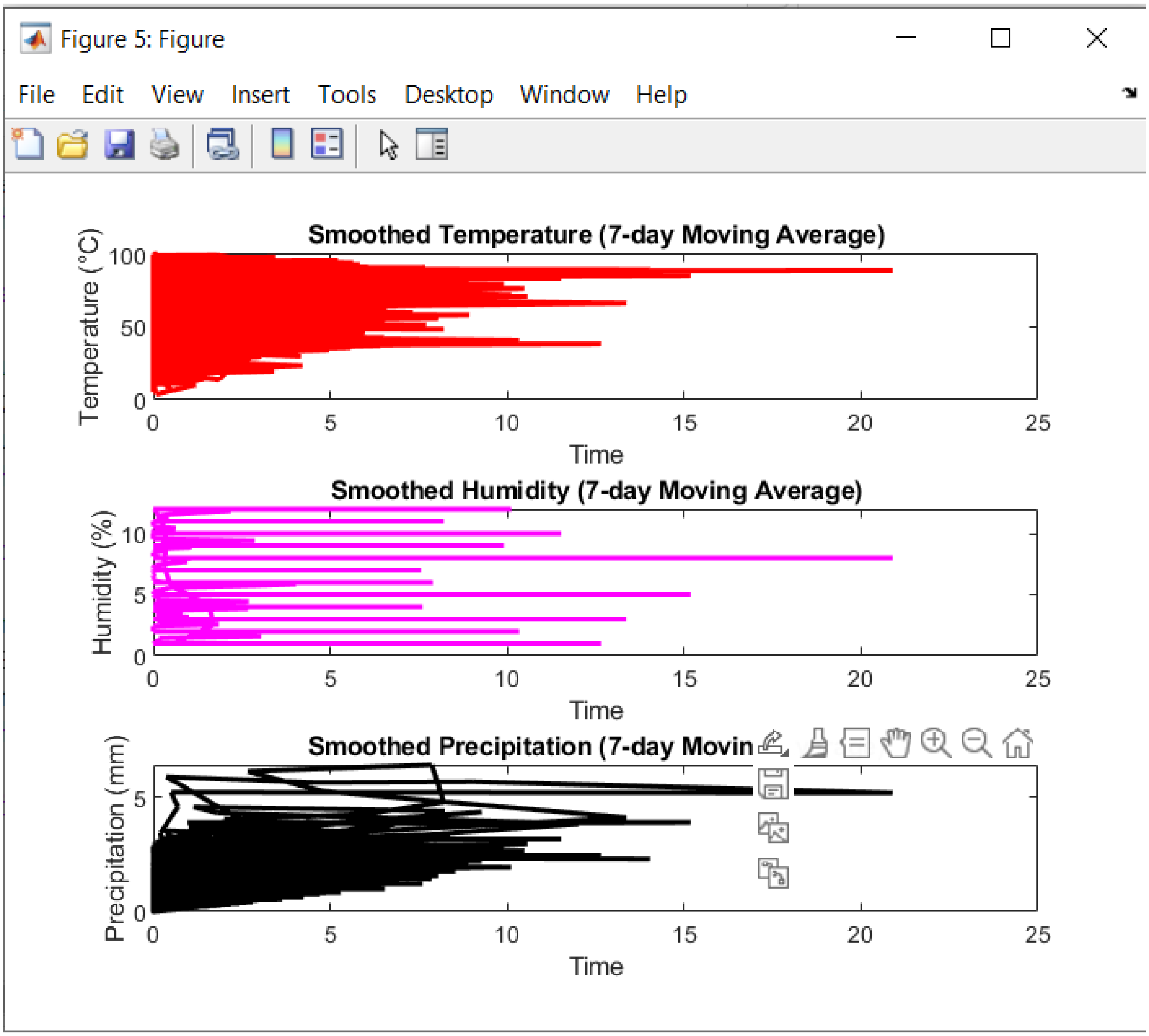
```
subplot(3,1,3);
plot(time, precipitation, '-c', 'LineWidth', 1.5);
title('windspeed');
xlabel('Time');
ylabel('Precipitation (m)');
```



Using moving average

```
% Define window size for moving average (e.g., 7 for weekly averages)
windowSize = 7;


% Apply moving averages
temp_smooth = movmean(temperature, windowSize);
hum_smooth = movmean(humidity, windowSize);
precip_smooth = movmean(precipitation, windowSize);


% Plot smoothed data
figure;


% Smoothed Temperature
subplot(3,1,1);
plot(time, temp_smooth, '-r', 'LineWidth', 1.5);
```

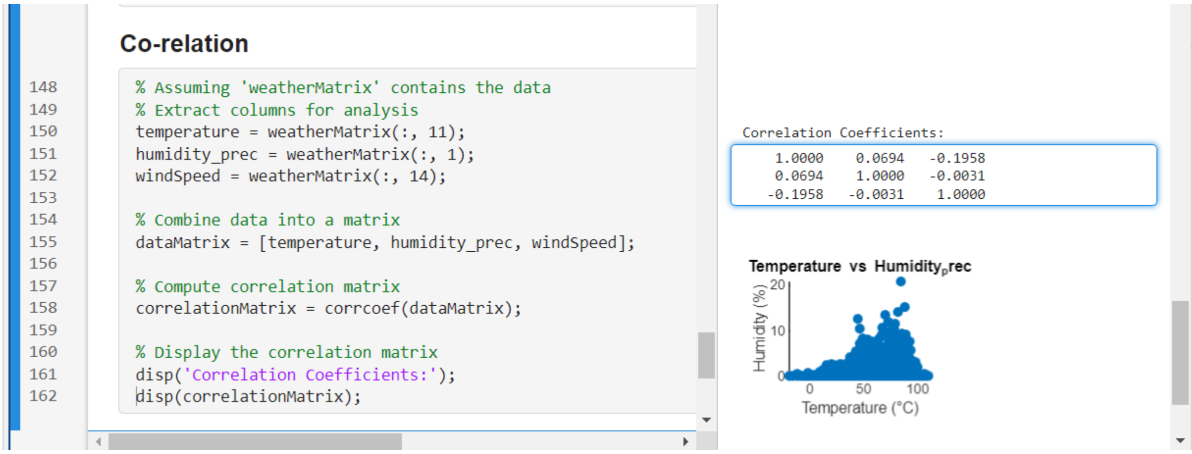
```
title('Smoothed Temperature (7-day Moving Average)');

xlabel('Time');

ylabel('Temperature (°C)');


% Smoothed Humidity
subplot(3,1,2);

plot(time, hum_smooth, '-m', 'LineWidth', 1.5);

title('Smoothed Humidity (7-day Moving Average)');

xlabel('Time');

ylabel('Humidity (%)');


% Smoothed Precipitation
subplot(3,1,3);

plot(time, precip_smooth, '-k', 'LineWidth', 1.5);

title('Smoothed Precipitation (7-day Moving Average)');

xlabel('Time');

ylabel('Precipitation (mm)');
```



Advanced Data Analysis:

● **Task 3.1:** *Perform correlation analysis between different weather parameters.*

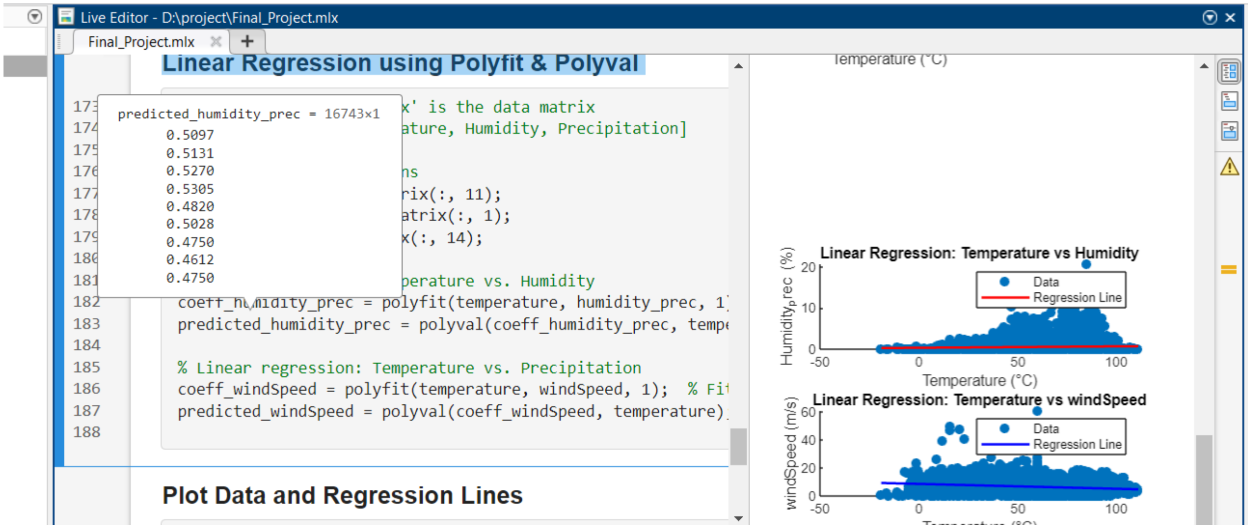
○ *Calculate correlation coefficients and create scatter plots to visualize relationships.*



● **Task 3.2:** *Implement linear regression to model the relationship between temperature and other weather parameters.*
○ *Use the polyfit and polyval functions to fit and evaluate the regression model.*
○ *Plot the regression line and analyze the results.*

Linear regression process is a technique of statistical analysis using least square technique.

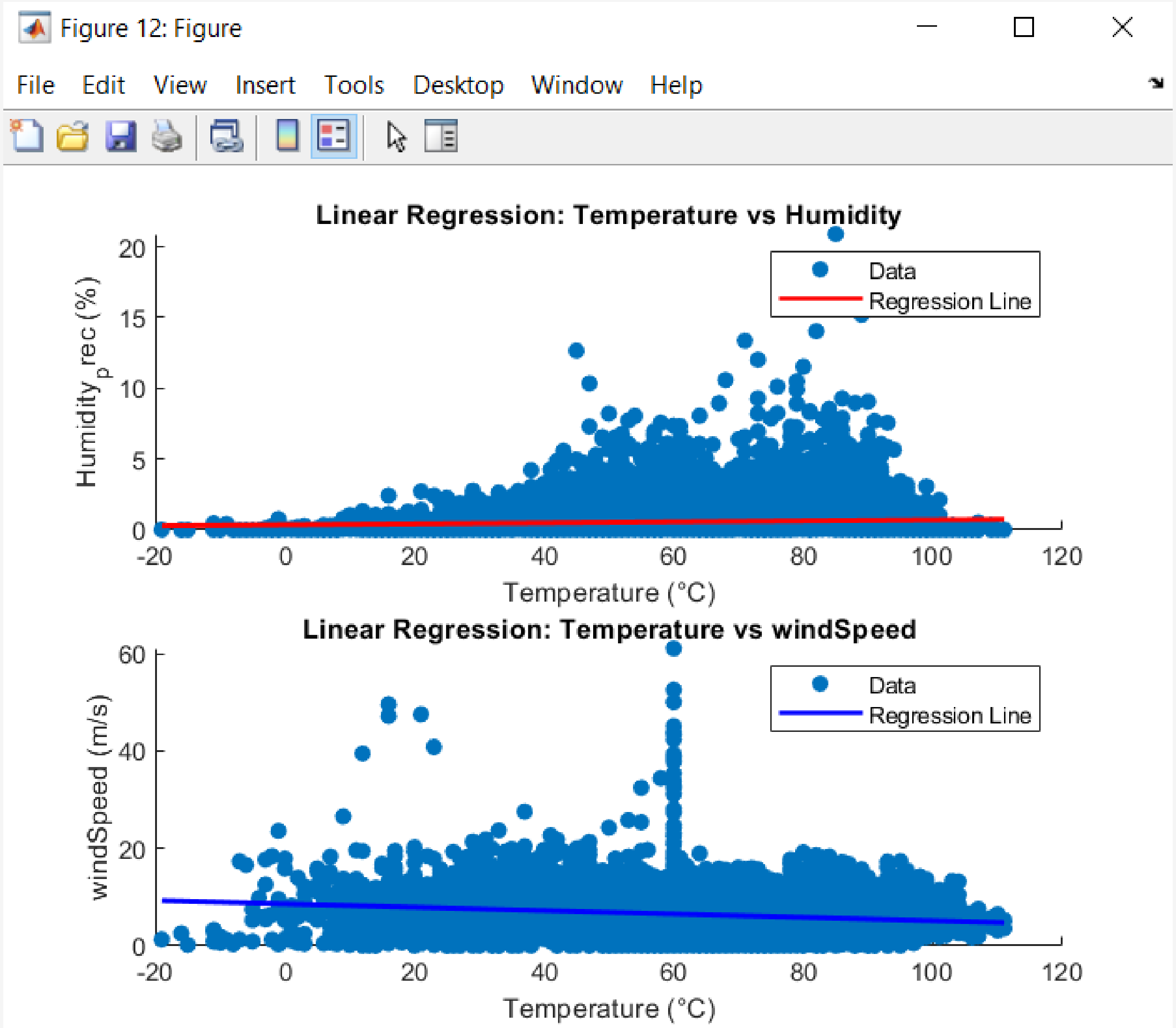
Linear Regression using Polyfit & Polyval



Plot Data and Regression Lines

```
figure;


% Plot: Temperature vs Humidity_prec
subplot(2, 1, 1);
scatter(temperature, humidity_prec, 'filled');  % Original data points
hold on;
plot(temperature, predicted_humidity_prec, '-r', 'LineWidth', 1.5);  % Regression
line
hold off;
title('Linear Regression: Temperature vs Humidity');
xlabel('Temperature (°C)');
ylabel('Humidity_prec (%)');
legend('Data', 'Regression Line');




% Plot: Temperature vs windSpeed
subplot(2, 1, 2);
scatter(temperature, windSpeed, 'filled');  % Original data points
hold on;
plot(temperature, predicted_windSpeed, '-b', 'LineWidth', 1.5);  % Regression
line
hold off;
title('Linear Regression: Temperature vs windSpeed');
```

```
xlabel('Temperature (°C)');
ylabel('windSpeed (m/s)');
legend('Data', 'Regression Line');
```

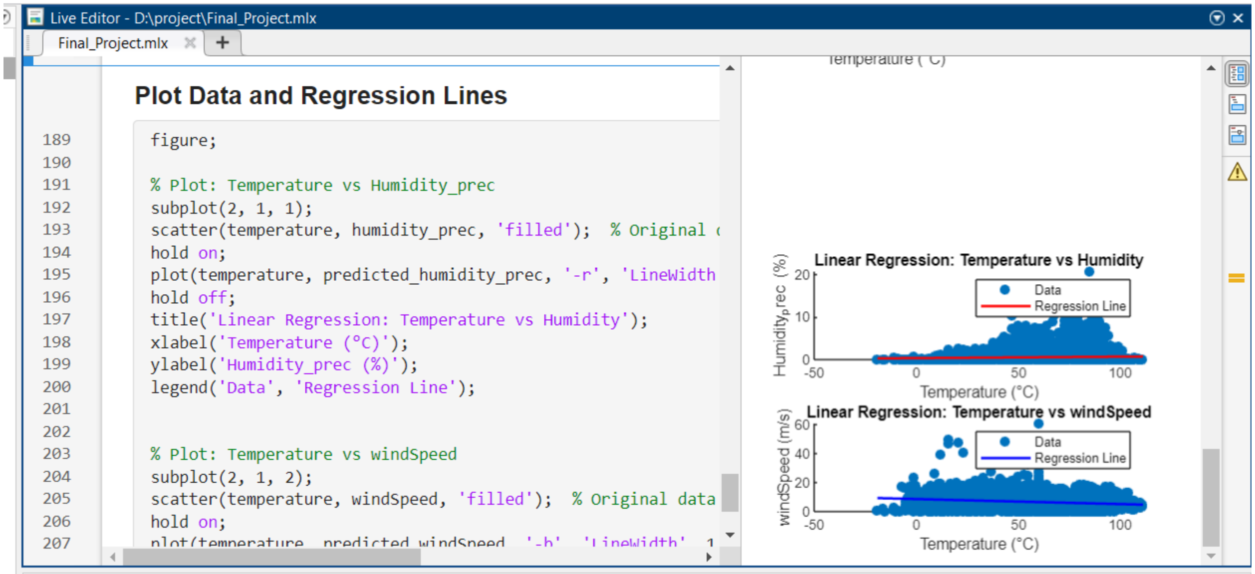


Data Visualization:

● **Task 4.1***: Create comprehensive visualizations to present the analysis.*
○ *Use subplot to create multiple plots in a single figure.*
○ *Customize plots with titles, labels, legends, and annotations.*

● **Task 4.2***: Develop 3D surface plots to visualize temperature variation over time and space.*
○ *Use the surf and mesh functions to create 3D plots*

4.1: subplot with legend & annotation

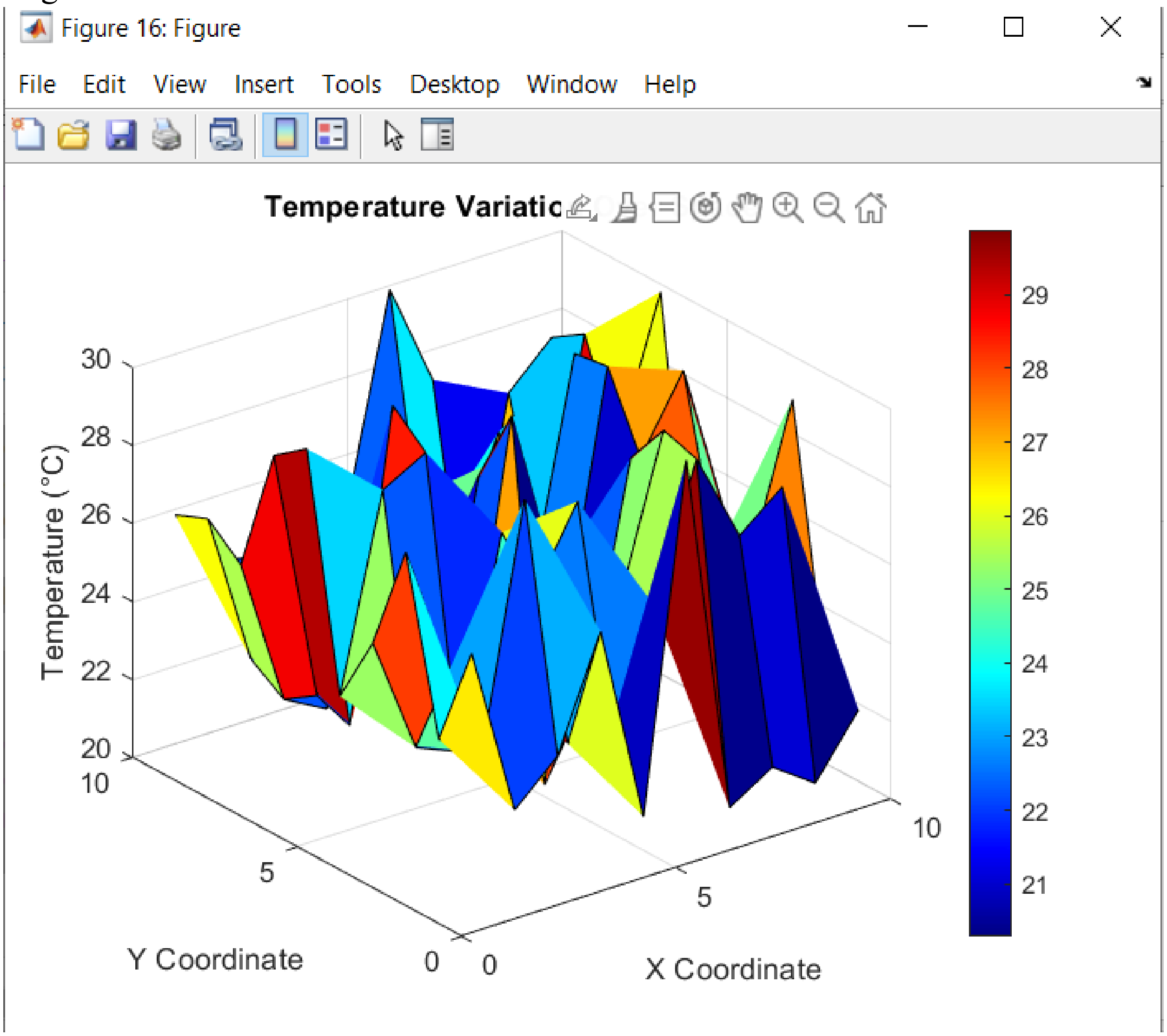


Task 4.2: Developing 3D surface plot for temperature variation

Code:

```
% Simulated data for spatial variation
[x, y] = meshgrid(1:10, 1:10);  % Create a 10x10 grid
z = rand(10, 10) * 10 + 20;      % Simulated temperature data


% Create 3D surface plot
figure;
surf(x, y, z);
title('Temperature Variation Over Space');
xlabel('X Coordinate');
ylabel('Y Coordinate');
zlabel('Temperature (°C)');
colormap('jet'); % Use a color map
colorbar;
```

Figure:



Mesh Plot for the weather report.

3D Mesh plot for temperature variation

```
figure;

mesh(x, y, z);

title('Temperature Variation Over Space (Mesh Plot)');

xlabel('X Coordinate');

ylabel('Y Coordinate');

zlabel('Temperature (°C)');
```

```
colormap('hot'); % Use a different color map

colorbar;
```