Assignment 2: Advanced Cryptography and Cryptanalysis (COSC5196)

Mihirkumar Mistry (Student ID: 249419480)
Divkumar Patel (Student ID: 249417620)
Group Number: 11

October 2024

Introduction

In this assignment, we explore the fundamentals of cryptography using the Caesar Cipher and MATLAB. We begin by setting up your programming environment and familiarizing with basic functions of MATLAB. The first part of this assignment, focuses on applying the Caesar Cipher with a key of 0, allowing us to observe the effects of encryption on text without any actual shifts. We analyze the letter frequencies of both the original and cipher-text, discovering how they remain identical when no shift occurs.

In second part, we adjust the encryption key to 1, generating a cipher-text that reflects a simple character shift. Through visualizations of letter frequencies, we learn how small changes in the key can alter the distribution of letters. Finally, we engage in cryptanalysis by determining the encryption key from a given cipher-text and automating this process through creating MATLAB code. This will deepen our understanding of cryptographic techniques and reinforce your programming skills.

Part-0: Introduction to MATLAB

We installed the MATLAB on our system. First of all, we go through the some of basic documentation and functions of the MATLAB like, MATLAB environment, Plotting, Numerical methods and programming methods. We will this knowledge for tasks, visualizing the data through the plots and graphs. [1]

Part-1: Plotting Letter Frequency with k=0

In this part, we have explored the MATLAB Caesar Cipher encryption code to encrypt the content of **PlainText1.txt** and generate an encrypted file called **CipherText1.txt**. For this step, we have used a key of K=0, which means no

shift will be applied. After that, we have compared the letter frequency graphs of both the original (plain-text) and the encrypted text to see how encryption impacts letter distribution.

Caesar Cipher encryption code (Key K=0)

This code will encrypt the plain-text using encryption key k=0.

```
%% Caesar Cipher encryption
% m = plain text string. Contains only a-to-z and space
% k = encryption key, ranges from 1 to 26
% cipherText = encrypted text k(m).

clc; clear all; close all;

m = fileread('PlainText1.txt'); % reading plaintext from text file
k = 0; % encryption key
ascii_m = double(m);         % ascii values of the string

%% Finding the locations of special characters
characters1 = find(ascii_m < 65);
characters2 = find(ascii_m == 96);
characters3 = find(ascii_m > 122);

%% special characters are replaced by space
ascii_m(characters1) = 32;
ascii_m(characters2) = 32;
ascii_m(characters3) = 32;


%% Encryption
ascii_cipherText = ascii_m+k;
wrap = find(ascii_cipherText>122); % wraping around if greater than
     'z'
ascii_cipherText(wrap) = ascii_cipherText(wrap)-26;
wrap = find(ascii_cipherText==96); % wraping around if greater than
     'z'
ascii_cipherText(wrap) = ascii_cipherText(wrap)-26;


%% restoring spaces
ascii_cipherText(characters1) = 32;
ascii_cipherText(characters2) = 32;
ascii_cipherText(characters3) = 32;

cipherText = char(ascii_cipherText);

%% Writing encrypted text in a text file
%fid = fopen('C:\Users\Administrator\Documents\MATLAB\CipherText1.
    txt','wt');
fid = fopen('Task_1_CipherText1.txt','wt');
fprintf(fid, '%s', cipherText);
fclose(fid);
```

Letter Frequency Distribution Plot Code

This code will plot the letter frequency distribution graph of the plain-text and cipher-text.

```
%% This program plots the letter frequency of the input text
clc; close all; clear all;

%% Reading ciphertext from file
cipherText = fileread('Task_1_CipherText1.txt');
ascii_cipherText = double(cipherText); %Converting string to
    numeric ASCII values

%% Reading plaintext from file
plainText = fileread('PlainText1.txt');
ascii_plainText = double(plainText); % converting string to numeric
     ASCII values

%% array declaration. Array size 1x26
frequency_cipher = zeros(1,26);
frequency_plain = zeros(1,26);

%% Counting frequency for small case letters
for i= 97:1:122
    frequency_cipher(i-96) = length(find(ascii_cipherText==i));
    frequency_plain(i-96) = length(find(ascii_plainText==i));
end

%% Counting frequency for capital case letters
for i= 65:1:90
    frequency_cipher(i-64) = frequency_cipher(i-64) + length(find(
    ascii_cipherText==i));
    frequency_plain(i-64) = frequency_plain(i-64) + length(find(
    ascii_plainText==i));
end

%% Normalizing to percentage value
frequency_cipher = frequency_cipher/sum(frequency_cipher)*100;
frequency_plain = frequency_plain/sum(frequency_plain)*100;

%% Ploting letter frequency for ciphrtext
subplot(2,1,1)
bar(frequency_cipher, 'red')
xlabel('Encrypted Alphabets (a to z i.e., 0 to 26)')
ylabel('Frequency (in %)')
title('Letter Frequency Plot for Ciphertext')
grid on

%% Ploting letter frequency for plaintext
subplot(2,1,2)
bar(frequency_plain, '')
xlabel('Plain Alphabets (a to z i.e., 0 to 26)')
ylabel('Frequency (in %)')
title('Letter Frequency Plot for Plaintext')
grid on
```

MATLAB interface image

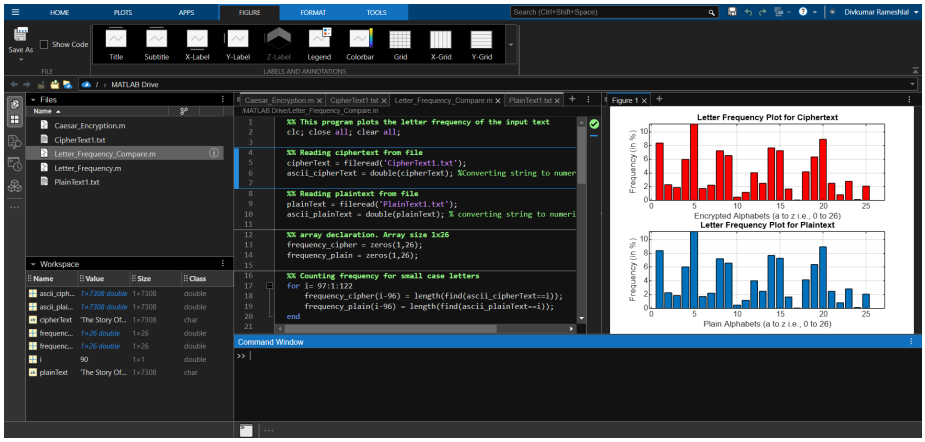

Figure 1: MATLAB interface image for part 1

Output

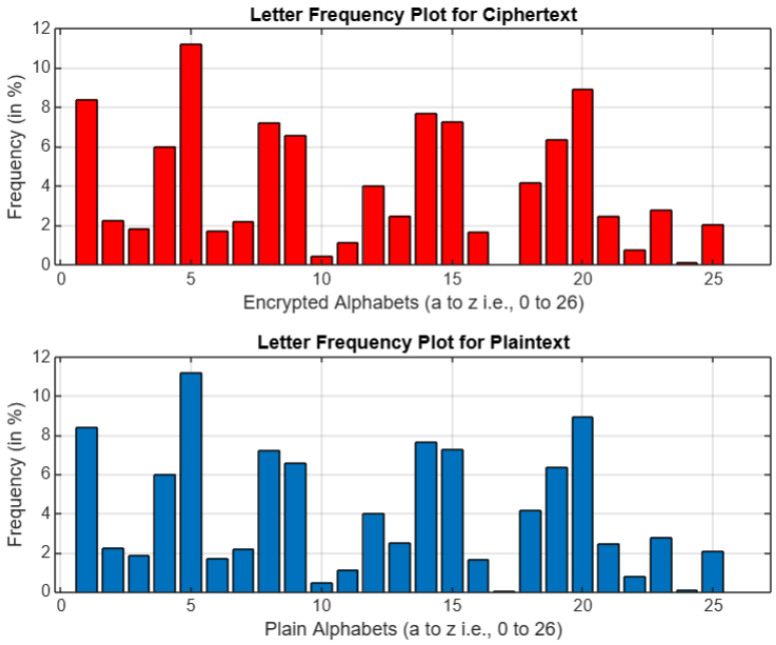

Figure 2: Part 1: plain-text and cipher-text frequency histogram

Here, key K=0 due to that both histogram are identical.

Part-2: Plotting Letter Frequency with k=1

In the second part, we use the same code of part-1 but with different key value **K=1**. Then, we will plot letter frequency histogram to compare plain-text and cipher-text letter frequency.

Caesar cipher encryption code (Key K=1)

This code will encrypt the plain-text using encryption key k=1.

```
%% Caesar Cipher encryption
% m = plain text string. Contains only a-to-z and space
% k = encryption key, ranges from 1 to 26
% cipherText = encrypted text k(m).

clc; clear all; close all;

```

```
m = fileread('PlainText1.txt'); % reading plaintext from text file
k = 1; % encryption key
ascii_m = double(m);          % ascii values of the string

%% Finding the locations of special characters
characters1 = find(ascii_m < 65);
characters2 = find(ascii_m == 96);
characters3 = find(ascii_m > 122);

%% special characters are replaced by space
ascii_m(characters1) = 32;
ascii_m(characters2) = 32;
ascii_m(characters3) = 32;


%% Encryption
ascii_cipherText = ascii_m+k;
wrap = find(ascii_cipherText>122); % wraping around if greater than
    'z'
ascii_cipherText(wrap) = ascii_cipherText(wrap)-26;
wrap = find(ascii_cipherText==96); % wraping around if greater than
    'z'
ascii_cipherText(wrap) = ascii_cipherText(wrap)-26;


%% restoring spaces
ascii_cipherText(characters1) = 32;
ascii_cipherText(characters2) = 32;
ascii_cipherText(characters3) = 32;

cipherText = char(ascii_cipherText);

%% Writing encrypted text in a text file
%fid = fopen('C:\Users\Administrator\Documents\MATLAB\CipherText1.
    txt','wt');
fid = fopen('Task_2_CipherText1.txt','wt');
fprintf(fid, '%s', cipherText);
fclose(fid);
```

Letter frequency distribution plot code

This code will plot the letter frequency distribution graph of the plain-text and cipher-text.

```
%% This program plots the letter frequency of the input text
clc; close all; clear all;

%% Reading ciphertext from file
cipherText = fileread('Task_2_CipherText1.txt');
ascii_cipherText = double(cipherText); %Converting string to
    numeric ASCII values

%% Reading plaintext from file
plainText = fileread('PlainText1.txt');
ascii_plainText = double(plainText); % converting string to numeric
    ASCII values
```

```
11
12 %% array declaration. Array size 1x26
13 frequency_cipher = zeros(1,26);
14 frequency_plain = zeros(1,26);
15
16 %% Counting frequency for small case letters
17 for i= 97:1:122
18     frequency_cipher(i-96) = length(find(ascii_cipherText==i));
19     frequency_plain(i-96) = length(find(ascii_plainText==i));
20 end
21
22 %% Counting frequency for capital case letters
23 for i= 65:1:90
24     frequency_cipher(i-64) = frequency_cipher(i-64) + length(find(
       ascii_cipherText==i));
25     frequency_plain(i-64) = frequency_plain(i-64) + length(find(
       ascii_plainText==i));
26 end
27
28 %% Normalizing to percentage value
29 frequency_cipher = frequency_cipher/sum(frequency_cipher)*100;
30 frequency_plain = frequency_plain/sum(frequency_plain)*100;
31
32 %% Ploting letter frequency for ciphrtext
33 subplot(2,1,1)
34 bar(frequency_cipher, 'red')
35 xlabel('Encrypted Alphabets (a to z i.e., 0 to 26)')
36 ylabel('Frequency (in %)')
37 title('Letter Frequency Plot for Ciphertext')
38 grid on
39
40 %% Ploting letter frequency for plaintext
41 subplot(2,1,2)
42 bar(frequency_plain, '')
43 xlabel('Plain Alphabets (a to z i.e., 0 to 26)')
44 ylabel('Frequency (in %)')
45 title('Letter Frequency Plot for Plaintext')
46 grid on
```

MATLAB interface image

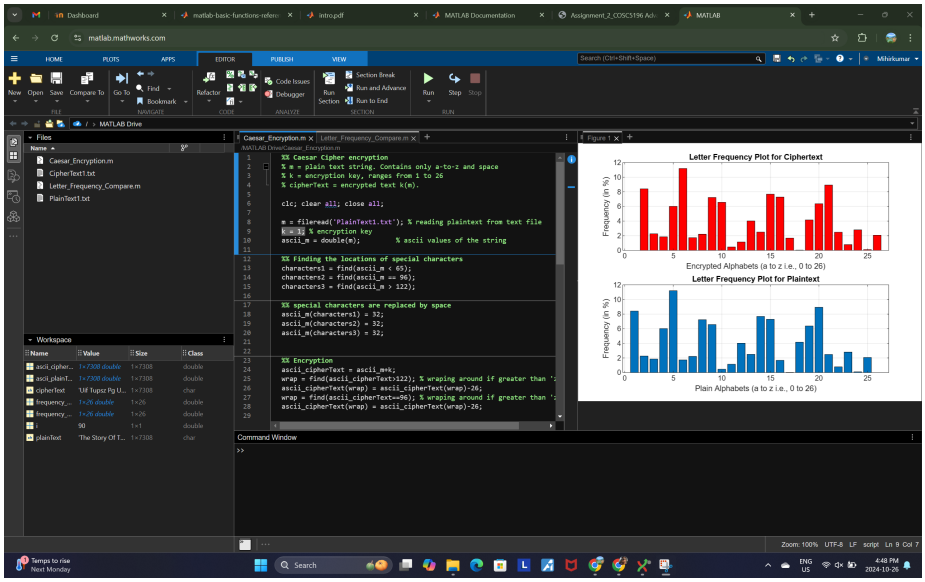

Figure 3: MATLAB interface image for part 2

Output

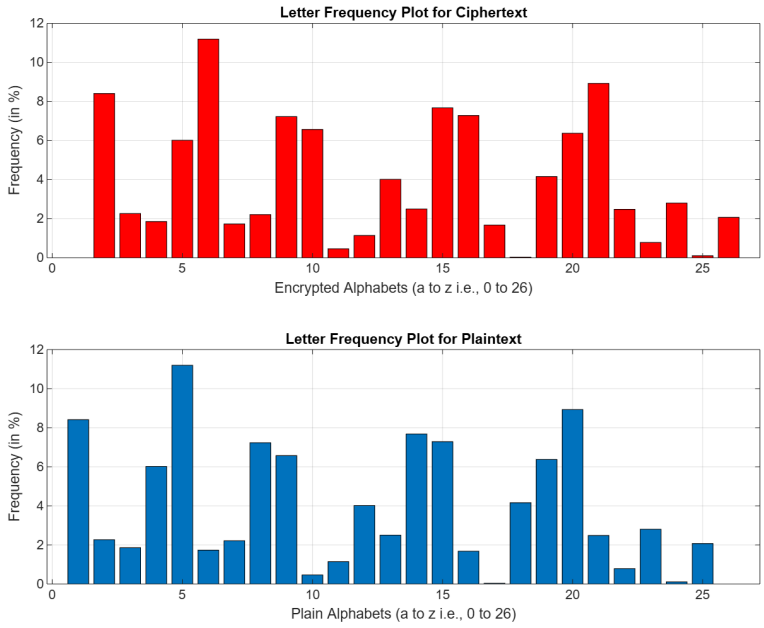

Figure 4: Part 2: plain-text and cipher-text frequency histogram

Here, We can see the difference between the plain-text and cipher-text histogram because of the key value K=1.

Part-3: Cryptanalysis

In the third part, we created a histogram displaying the letter frequencies in "CipherText2.txt".

Histogram of CipherText2.txt

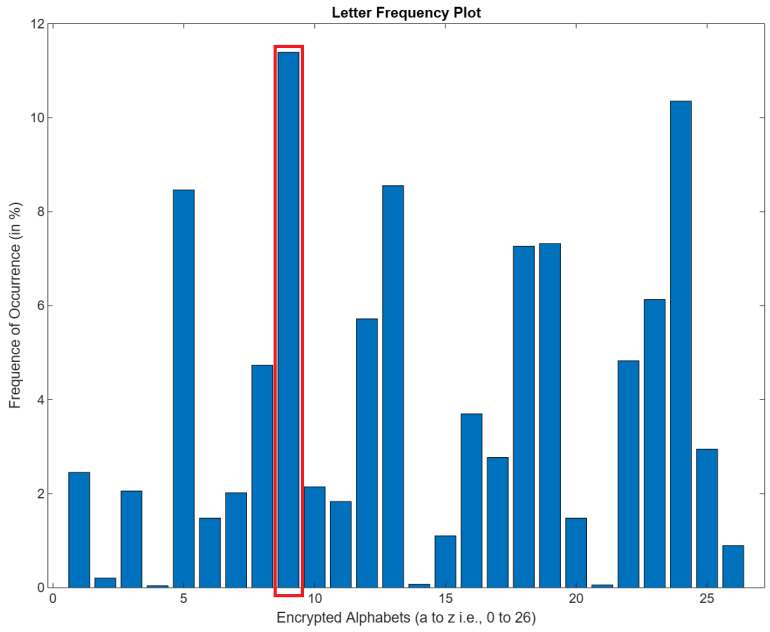

Figure 5: Histogram of CipherText2.txt

To interpret this, we compared the histogram with the typical letter frequency distribution of the English alphabet, where the letter **"E"** is known to have the highest frequency. By using this as a reference, we identified the letter with the highest frequency in our cipher-text, located in column 9 on the cryptanalysis graph. To calculate the key, we took the index of **"E"** (position 5 in the alphabet) and subtracted it from the index of the most frequent letter in our cipher-text (position 9). **This gives us a possible key of 4.** [2] Then we use the decryption algorithm from assignment-1 to decrypt the CipherText2.txt using the key value K=4 and we got the following result.

Caesar cipher decryption code from Assignment 1

```
"""
Created on Mon Oct  7 00:28:03 2024
@author: Mihirkumar Mistry
Student ID: 249419480
Group Number: 11
"""
# Take cipher text from the user
```

```
cipher_text = input('Enter cipher text:')
# Take key value from the user
key = int(input('Enter key:'))

# Caesar cipher decryption function
def caesar_cipher_decryption(cipher_text, key):
    plain_text = ""

    for char in cipher_text:
        # Check if the char is a alphabet or not
        if char.isalpha():
            # Get the ASCII value of the base char, based on the case
            start = ord('A') if char.isupper() else ord('a')
            # Finding the plaintext using character shifting algorithm
            decrypted_char = chr((ord(char) - start - key) % 26 + start)
            # Add the resulting char
            plain_text += decrypted_char
        else:
            # Keep the Non-alphabet character as it is
            plain_text += char

    return plain_text

# Call caesar_cipher_decryption with user input
print('Plaintext:',caesar_cipher_decryption(cipher_text, key))
```

Output

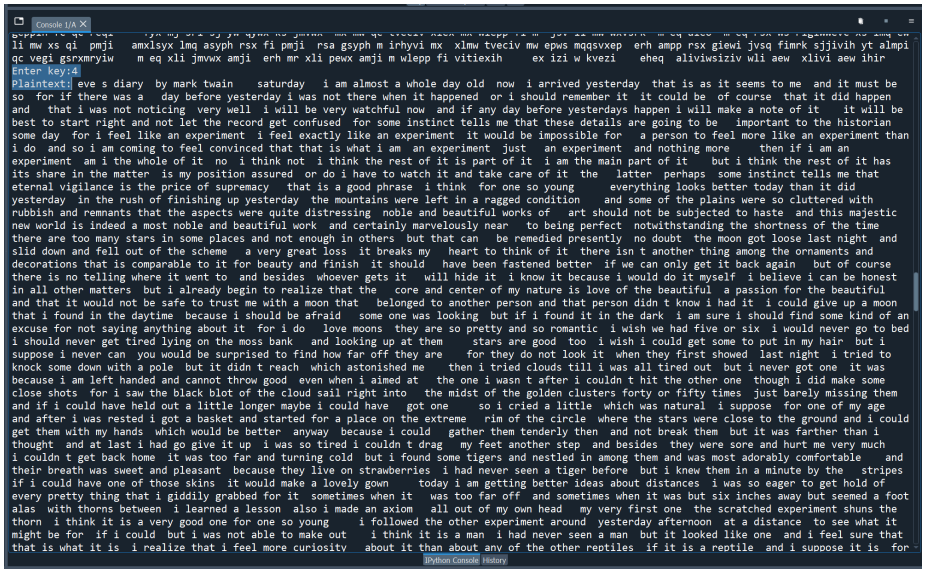

Figure 6: Part 3: Decrypted text using key value 4

Histogram of plain-text and cipher-text

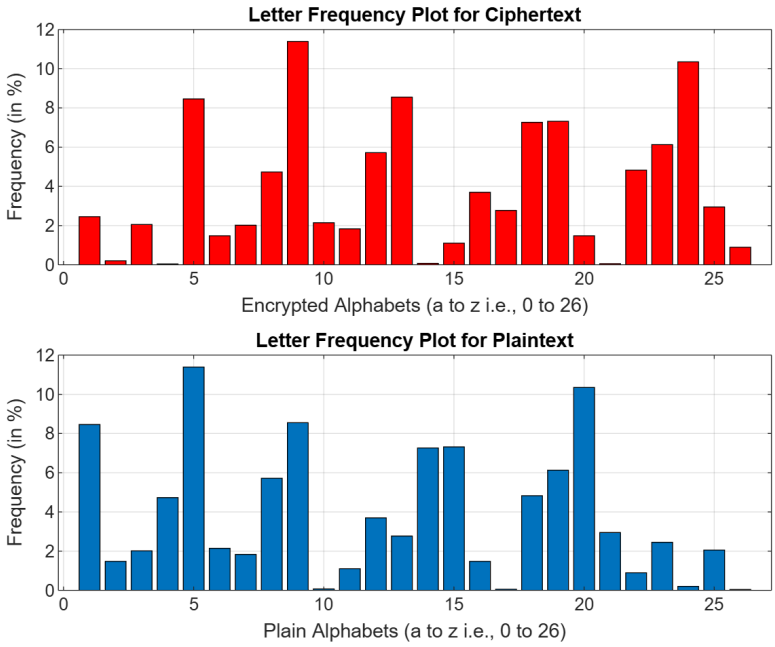

Figure 7: Part 3: Histogram of plain-text and cipher-text

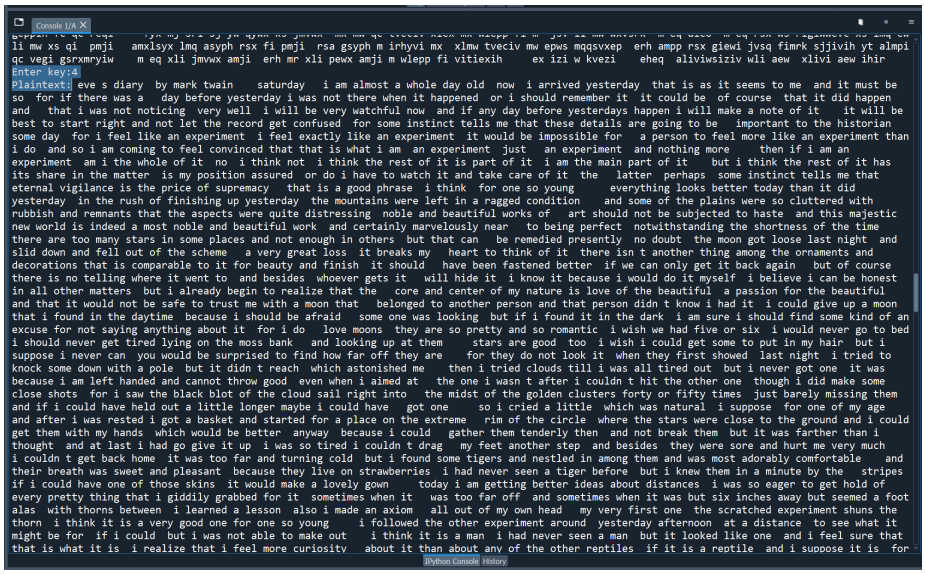

Figure 8: Part 3: Decrypted text using key value 4

Furthermore, We can answer the following questions using the output.

- What is the story name? **Answer: eve s diary**
- Who is the writer? **Answer: mark twain**

Part-4: Write Your Own Code

In the final part of our analysis, we developed a MATLAB code to perform cryptanalysis on the cipher-text. This code uses the approach we outlined in part 3 to identify the likely key, which it then uses to decrypt the cipher-text. [2]

Caesar cipher cryptanalysis code

This code will analyses the possible key for the cipher text and generate the text file of the plain-text.[3]

```
% This program plots the letter frequency of the input text
clc; close all; clear all;

%text = fileread('EvesDiaryMarkTwain.txt'); % reading text from
    file
text = fileread('CipherText2.txt'); % reading text from file
ascii_text = double(text); % converting string to numeric ASCII
    values
```

```
7
8  frequency = zeros(1,26); % array declaration. Array size 1x26
9
10 %% Counting frequency for small case letters
11 for i= 97:1:122
12     frequency(i-96) = length(find(ascii_text==i));
13 end
14
15 %% Counting frequency for capital case letters
16 for i= 65:1:90
17     frequency(i-64) = frequency(i-64) + length(find(ascii_text==i))
       ;
18 end
19
20
21 % Standard English letter frequencies (approximate)
22 english_freq = [8.167, 1.492, 2.782, 4.253, 12.702, 2.228, 2.015,
       6.094, ...
23                 6.966, 0.153, 0.772, 4.025, 2.406, 6.749, 7.507,
       1.929, ...
24                 0.095, 5.987, 6.327, 9.056, 2.758, 0.978, 2.361,
       0.150, ...
25                 1.974, 0.074];
26
27 % Normalize the frequencies of the ciphertext to compare with
       English frequencies
28 total_letters = sum(frequency);
29 normalized_frequency = (frequency / total_letters) * 100;
30
31 % Find the best shift by comparing each possible shift with
       standard frequencies
32 best_shift = 0;
33 min_difference = inf; % Start with a large number
34
35 for shift = 0:25
36     % Shift frequencies
37     shifted_frequency = circshift(normalized_frequency, -shift);
38
39     % Calculate the sum of absolute differences for this shift
40     difference = sum(abs(shifted_frequency - english_freq));
41
42     % Update the best shift if this one has a smaller difference
43     if difference < min_difference
44         min_difference = difference;
45         best_shift = shift;
46     end
47 end
48
49 % Decrypt the text using the best shift found
50 plaintext = char(ascii_text); % Initialize with original text
       structure
51 for i = 1:length(ascii_text)
52     if ascii_text(i) >= 65 && ascii_text(i) <= 90
53         % Uppercase letters
54         plaintext(i) = char(mod(ascii_text(i) - 65 - best_shift,
       26) + 65);
55     elseif ascii_text(i) >= 97 && ascii_text(i) <= 122
```

```
        % Lowercase letters
        plaintext(i) = char(mod(ascii_text(i) - 97 - best_shift,
    26) + 97);
    end
end

% Save the plaintext to a file
fileID = fopen('PlainText2.txt', 'w');
fprintf(fileID, '%s', plaintext);
fclose(fileID);

% Display the encryption key and success message
fprintf('The encryption/decryption key (shift) is: %d\n',
    best_shift);
disp('Decryption complete. Plaintext saved to PlainText2.txt.');
```

MATLAB interface image

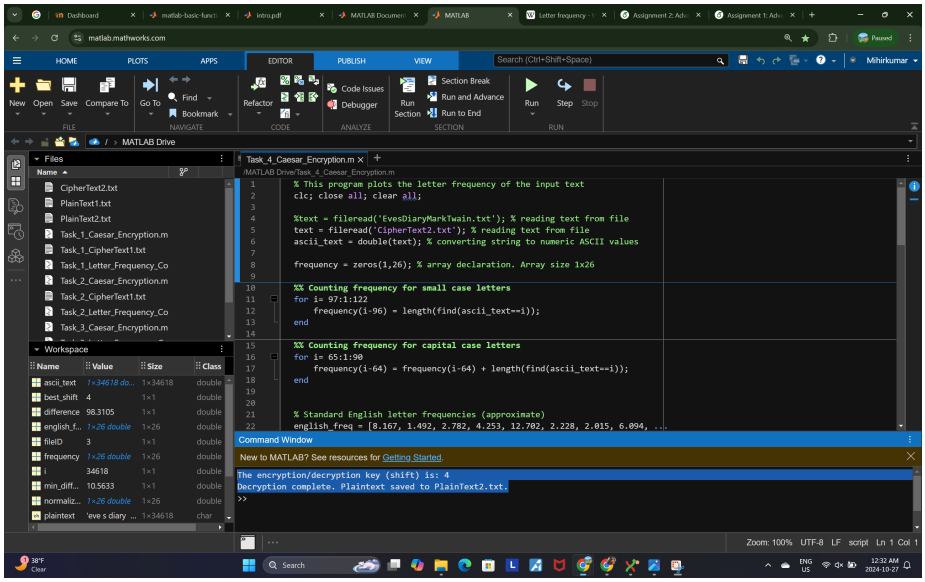

Figure 9: MATLAB interface image for part 4

Output

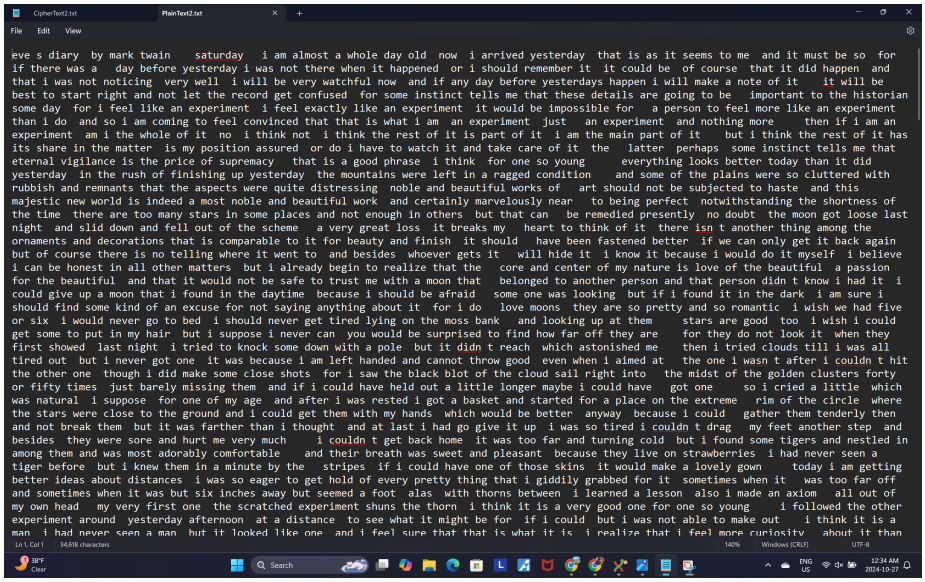

Figure 10: Part 4: Cryptanalysis output

Acknowledgment

- **Part-0: Introduction to MATLAB**
  - Divkumar Patel (Student Id: 249417620)
  - Mihirkumar Mistry (Student Id: 249419480)
- **Part-1: Plotting Letter Frequency with k=0**
  - Divkumar Patel (Student Id: 249417620)
- **Part-2: Plotting Letter Frequency with k=1**
  - Mihirkumar Mistry (Student Id: 249419480)
- **Part-3: Cryptanalysis**
  - Divkumar Patel (Student Id: 249417620)
  - Mihirkumar Mistry (Student Id: 249419480)
- **Part-4: Write Your Own Code**
  - Divkumar Patel (Student Id: 249417620)

  - Mihirkumar Mistry (Student Id: 249419480)

- **Assignment Report:**
  - Divkumar Patel (Student Id: 249417620)
  - Mihirkumar Mistry (Student Id: 249419480)

Conclusion

In conclusion, this assignment provided a hands-on exploration of cryptography fundamentals through the Caesar Cipher and MATLAB. Starting with basic encryption using a key of 0, we observed the unchanged letter frequency, reinforcing our understanding of how shifts affect text. Increasing the key to 1 showed how even small adjustments impact letter distribution. Finally, by developing MATLAB code for cryptanalysis, we applied these principles to decrypt unknown cipher-text, gaining insights into key discovery and decryption techniques. Overall, this exercise strengthened both our understanding of cryptographic concepts and our MATLAB programming skills.

References

[1] MATLAB. Matlab documentation.

[2] Wikipedia contributors. Letter frequency - wikipedia.

[3] GeeksforGeeks given i=G. Caesar cipher in cryptography.