Московский государственный технический университет
Факультет ИУ «Информатика и системы управления»
Кафедра ИУ-1 «Системы автоматического управления»

ОТЧЕТ

по лабораторной работе №1

«Ознакомление с основными компонентами среды MATLAB для разработки навигационных алгоритмов ИНС»

по дисциплине

«Интегрированные навигационные системы»

Выполнил: Машков И.И.
Группа: ИУ1-83
Проверил: Лукьянов В.В.

2024 г.

Цель работы: ознакомиться с основными компонентами среды MATLAB для последующей разработки навигационных алгоритмов ИНС.

Задание: из предложенных исходных файлов считать матрицы, выполнить требуемые преобразования. Полученные результаты загрузить в выходной файл и построить графики.

Практическая часть

1. Реализовать считывание входных данных из файла.

```
clc
clear variables
close all

%% 1. Input data reading from files

filename_A = 'A.txt';
filename_B = 'B.txt';
filename_C = 'C.txt';
filename_coord = 'coord_rad.txt';
filename_output = 'output.txt';
```

1.1. С помощью функции readmatrix считать матрицу из файла A.txt в переменную A. С помощью функции writematrix записать матрицу A в файл A_out.txt. Сравнить содержание двух файлов.

```
% Readmatrix
A = readmatrix(filename_A);
fprintf("A = \n");
disp(A);
```

На рисунках ниже (A.txt, A_out.txt) представлено содержание обоих файлов.

```
1  5 6 7
2  9 0 11
3  -2 14 44
4
```

```
1  5,6,7
2  9,0,11
3  -2,14,44
4
```

При сравнении данный файлов можно сказать, что в файле A.txt матрица выводится без запятых, а в файле A_out.txt – с запятыми.

1.2. С помощью функции load произвести загрузку матрицы из исходного файла B.txt в переменную B.

```
% Load
B = load(filename_B);
fprintf("B = \n");
disp(B);
```

1.3. Реализовать построчное считывание матрицы из файла C.txt с помощью функций fgetl и str2num в переменную C.

```
% Line by line reading
C = [];
C_fid = fopen(filename_C);
while ~feof(C_fid)
    tline = fgetl(C_fid);
    C = [C; str2num(tline)];
end
fprintf("C = \n");
disp(C);
```

На рисунке ниже представлены полученные матрицы A, B и C:

```
A =
     5     6     7
     9     0    11
    -2    14    44

B =
    -1    14     8
     9    12    33
     7   -14    11

C =
     5     0    12
     1    41    -3
     1     4    55
```

2. Выполнить матричные преобразования.

2.1. Найти определитель матрицы A, матрицу B умножить на число, матрицу C транспонировать, используя встроенные функции среды MATLAB det и C’.

```
% det, *, '
det_A = det(A);
B_multiplied = 5 * B;
C_transformed = C';

fprintf("det(A) = %d\n\n", det_A);
fprintf("5 * B = \n");
disp(B_multiplied);
fprintf("C^T = \n");
disp(C_transformed);
```

Вывод результатов представлен на ниже.

```
det(A) = -2396

5 * B =
    -5    70    40
    45    60   165
    35   -70    55

C^T =
     5     1     1
     0    41     4
    12    -3    55
```

2.2. Найти обратную матрицу A с помощью встроенной функции inv, проверить результат.

```
% Inverse matrix
inv_A = inv(A);
det_inv_A = det(inv_A);

fprintf("A^-1 = \n");
disp(inv_A);
fprintf("det(A^-1) = %f\n\n", det_inv_A);
fprintf("1/det(A) = %f\n\n", 1/det_A);
```

Вывод результатов представлен на рисунке ниже.

```
A^-1 =
    0.0643    0.0693   -0.0275
    0.1745   -0.0977   -0.0033
   -0.0526    0.0342    0.0225

det(A^-1) = -0.000417

1/det(A) = -0.000417
```

2.3. Задать единичную матрицу и считать элементы ее главной диагонали с помощью встроенных функций eye и diag.

```
% Eye matrix
eye_matrix = eye(3);
diag_elements = diag(eye_matrix);

fprintf("Eye matrix = \n");
disp(eye_matrix);
fprintf("Diag elements = \n");
disp(diag_elements);
```

Вывод результатов представлен на рисунке ниже.

```
Eye matrix =
     1     0     0
     0     1     0
     0     0     1

Diag elements =
     1
     1
     1
```

2.4. Вывести на экран матрицу C, определить ее размерность, сортировать ее по столбцам и по строкам по возрастанию и убыванию с помощью встроенных функций size и sort.

```
% C matrix transformations
size_C = size(C);

fprintf("C = \n");
disp(C);
fprintf("Size of C = \n");
disp(size_C);
fprintf("C sorted by rows ascending = \n");
disp(sort(C, 2));
fprintf("C sorted by rows descending = \n");
disp(sort(C, 2, 'descend'));
fprintf("C sorted by columns ascending = \n");
disp(sort(C));
fprintf("C sorted by columns descending = \n");
disp(sort(C, 'descend'));
```

Вывод результатов представлен на рисунке ниже.

```
C =
     5     0    12
     1    41    -3
     1     4    55

Size of C =
     3     3

C sorted by rows ascending =
     0     5    12
    -3     1    41
     1     4    55

C sorted by rows descending =
    12     5     0
    41     1    -3
    55     4     1

C sorted by columns ascending =
     1     0    -3
     1     4    12
     5    41    55

C sorted by columns descending =
     5    41    55
     1     4    12
     1     0    -3
```

2.5. Из матрицы C получить вектор-строку Q1 и вектор-столбец Q2 с помощью встроенной функции reshape. Сортировать вектор-строку по убыванию.

```
% Reshaping C matrix
Q1 = reshape(C, 1, []);
Q2 = reshape(C, [], 1);

fprintf("C Q1 row = \n");
disp(Q1);
fprintf("C Q2 column = \n");
disp(Q2);
```

Вывод результатов представлен на рисунке ниже.

```
C Q1 row =
     5     1     1     0    41     4    12    -3    55

C Q2 column =
     5
     1
     1
     0
    41
     4
    12
    -3
    55
```

2.6. Из матрицы B получить вектор-столбец Q1 способом группировки по столбцам командой B(:). Полученную матрицу записать в выходной файл output.txt с помощью встроенной функции writematrix.

```
% Writing B column
Q1 = B(:);
writematrix(Q1, filename_output);

fprintf("B Q1 column = \n");
disp(Q1);
```

Выводы результатов представлены на рисунках ниже.

```
B Q1 column =
    -1
     9
     7
    14
    12
   -14
     8
    33
    11
```

```
1   -1
2   9
3   7
4   14
5   12
6   -14
7   8
8   33
9   11
10
```

2.7. Из матрицы B получить вектор-строку Q2 способом группировки по столбцам.

```
% Writing B row
Q2 = B(:)';

fprintf("B Q2 row = \n");
disp(Q2);
```

Вывод результатов представлен на рисунке ниже.

```
B Q2 row =
    -1     9     7    14    12   -14     8    33    11
```

3. Отрисовка графиков

3.1 Задать функцию sin() и построить ее график в диапазоне $\pm 2\pi$ с помощью функций linespace и plot.

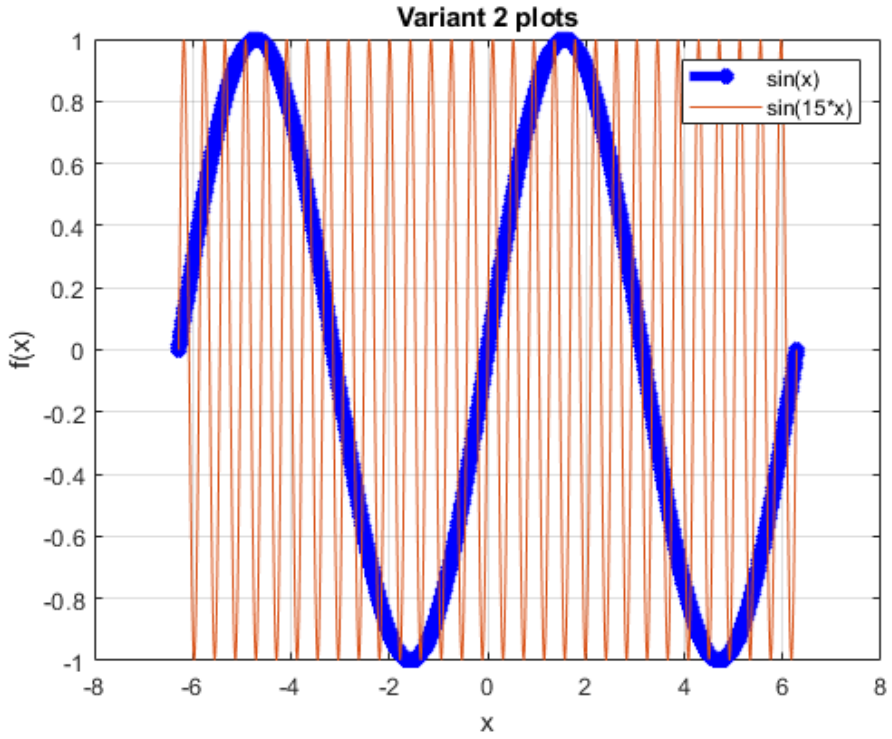
3.2 Согласно вариантам задать этому графику необходимые параметры.

3.3 Добавить на тот же рисунок (hold on) дополнительный график согласно варианту.

3. 4 К каждому графику задать подписи осей, заголовки и легенды.

```
%% 3. Plotting

% 1 plot with 2 functions
f1 = figure();

x = linspace(-2*pi, 2*pi, 1e3);
plot(x, sin(x), 'LineWidth', 4, 'Color', 'blue', 'LineStyle', '--', 'Marker', 'x');
hold on;
plot(x, sin(15 * x));
hold off;

grid on;
title('Variant 2 plots');
xlabel('x');
ylabel('f(x)');
legend('sin(x)', 'sin(15*x)');
```

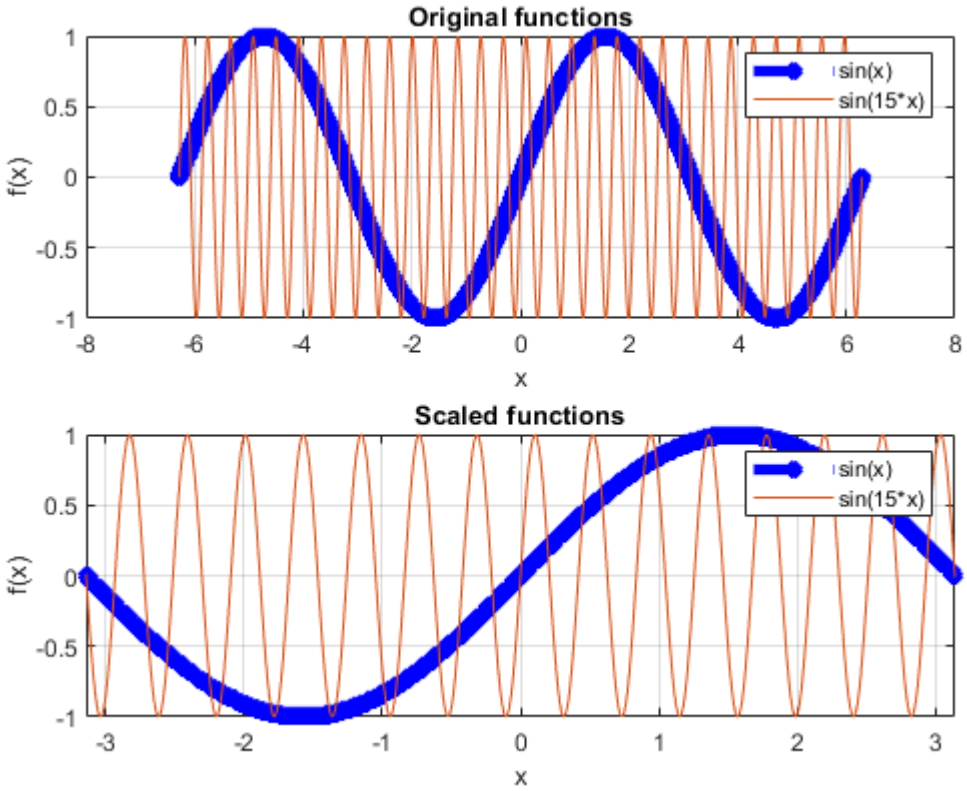
Вывод результатов представлен на рисунке ниже.



3.5 С помощью функции subplot() перенести исходные графики на первое полотно, а на втором сделать его приближение произвольного диапазона с помощью функции axis().

```
% 1st plot with 2 original functions and 2nd with 2 scaled functions
f2 = figure();

subplot(2, 1, 1);
x = linspace(-2*pi, 2*pi, 1e3);
plot(x, sin(x), 'LineWidth', 4, 'Color', 'blue', 'LineStyle', '--', 'Marker', 'x');
hold on;
plot(x, sin(15 * x));
hold off;

grid on;
title('Original functions');
xlabel('x');
ylabel('f(x)');
legend('sin(x)', 'sin(15*x)');

subplot(2, 1, 2);
x = linspace(-2*pi, 2*pi, 1e3);
plot(x, sin(x), 'LineWidth', 4, 'Color', 'blue', 'LineStyle', '--', 'Marker', 'x');
hold on;
plot(x, sin(15 * x));
hold off;

axis( [-pi, pi, -1, 1] );
grid on;
title('Scaled functions');
xlabel('x');
ylabel('f(x)');
legend('sin(x)', 'sin(15*x)');
```

Вывод результатов представлен на рисунке ниже.

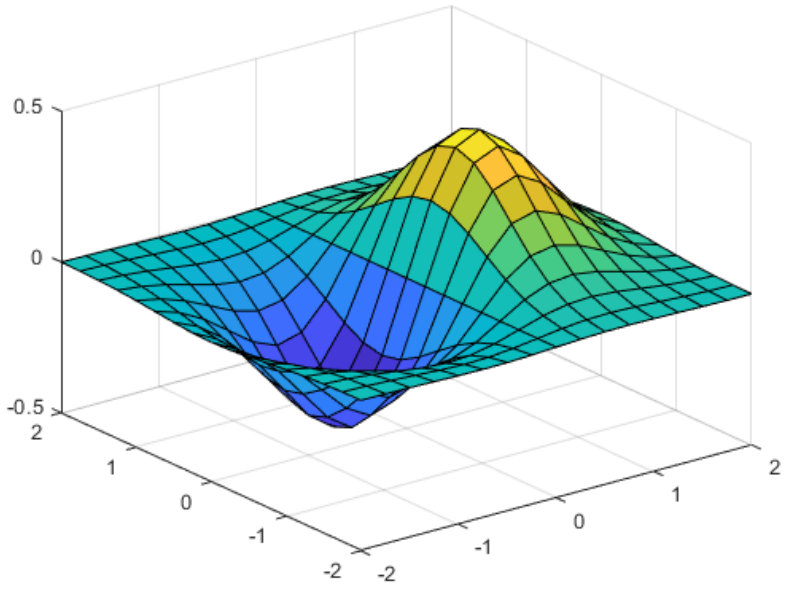


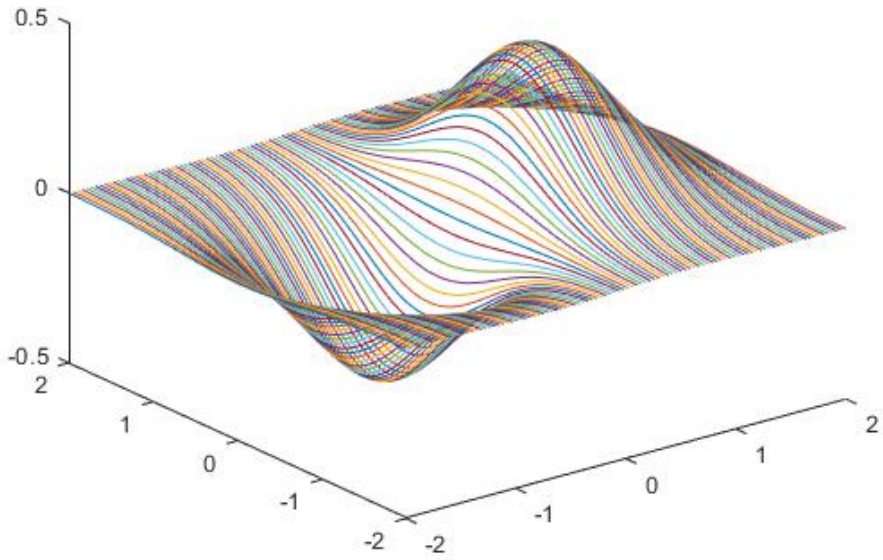
3.6 С помощью функции plot3(), построить график функции, а с помощью функции surf – поверхность:

$$z = x * exp(-x^2 - y^2)$$

```
% 3 dimensional plot
x = -2:0.25:2;
[X, Y] = meshgrid(x);
Z = X .* exp(-X.^2 - Y.^2);

f3 = figure();
surf(X, Y, Z);

x = linspace(-2, 2, 100);
y = linspace(-2, 2, 100);
[X, Y] = meshgrid(x, y);
Z = X .* exp(-X.^2 - Y.^2);

f4 = figure();
plot3(X, Y, Z);
```

Выводы результатов представлены на рисунках ниже.

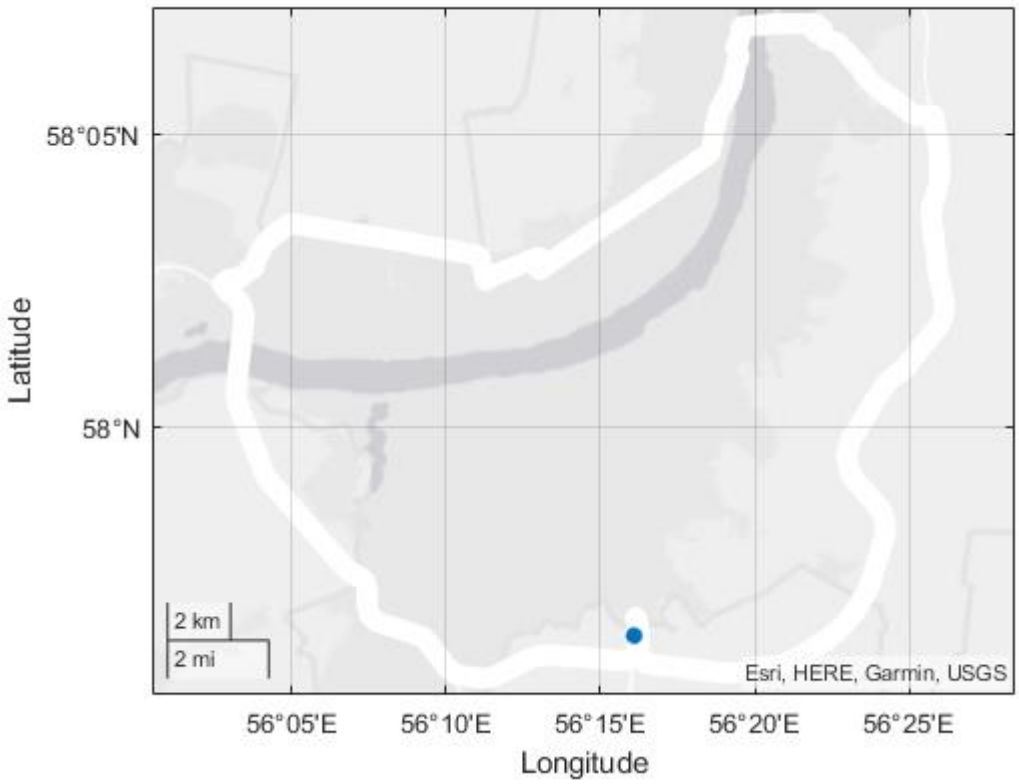




4. Из файла coord_rad.txt считать данные местоположения объекта, представленные в радианах, в виде матрицы. Преобразовать данные в градусы и нанести маршрут подвижного объекта на карту для поздних версий MATLAB с помощью функции geobubble(), для ранних версий – с помощью функции plot.

```
%% 4. Reading coords data from file

coords_data_rad = readmatrix(filename_coord);
rads_to_degrees = 180/pi;
coords_data_deg = rads_to_degrees * coords_data_rad;

f5 = figure();
degrees_table = array2table(coords_data_deg, 'VariableNames', {'Latitude', 'Longitude'});
geobubble(degrees_table, 'Latitude', 'Longitude');
```

Вывод результатов представлен на рисунке ниже.



Вывод

В данной работе проведено изучение основных элементов среды MATLAB с целью разработки навигационных алгоритмов ИНС. Рассмотрены методы загрузки данных из файлов, выполнение матричных операций, построение графиков функций и их визуализация. Также была рассмотрена функция, обеспечивающая работу с координатами подвижного объекта. Результаты лабораторной работы представлены в виде фрагментов кода, вывода в консоль и графиков.