

+++++

Lab 005

+++++

+++++

2154638

+++++

Mathematical Skills for Data Scientists

Lab Exercises 5

CSCM70

Lab 5 Solutions

Lab Exercise Sheet 5

Start Assignment

Due Monday by 23:59 Points 4 Submitting a file upload File types m, pdf, txt, and xls Available 7 Nov at 11:00 - 15 Nov at 2:00

You can find the second exercise sheet below. You have a full week to complete it.

[Lab ExerciseSheet5.pdf](#) ↓

Please upload the file as

- matlab m file (not live script mlx)
- text
- pdf
- xls

If you are doing multiple submissions, please include all the files in each submission. We will only mark the last submission (and the files attached to it)

[◀ Previous](#)[Next ▶](#)

Question 1: ->

Exercise 1 (Excel). Do linear regression on the following data:

x	y
1.1	2.7
1.2	2.8
2.4	5.2
3.1	6.8

What values for slope and intercept do you get? Try out forcing the intercept to be zero. Does the slope change? If so, why? 1 mark

Answer 1:



lab5_answer1.xlsx

Excel Sheet above.

Description :

In the below charts we've plotted the data given in question. The data is plotted for linear equation which can be represented as $y = mx + c$, with m as the slope of the line and c as the intercept of the line (the value of y where $x = 0$)

Chart 1 → Shows the best fitting line

- B8 is calculated as `=INDEX(LINEST(C2:C5,B2:B5),1)`
- C8 is calculated as `=INDEX(LINEST(C2:C5,B2:B5),2)`
- Dashed line represents extrapolation to the x and y axis

Chart 2 → shows the line where we've set the intercept as Zero

- B22 is calculated as `=INDEX(LINEST(C2:C5,B2:B5,0),1)`
- C22 is calculated as `=INDEX(LINEST(C2:C5,B2:B5,0),2)`
- Dashed line represents extrapolation to the x and y axis where it can be seen that y intercept is zero.

OUTPUT EXCEL:

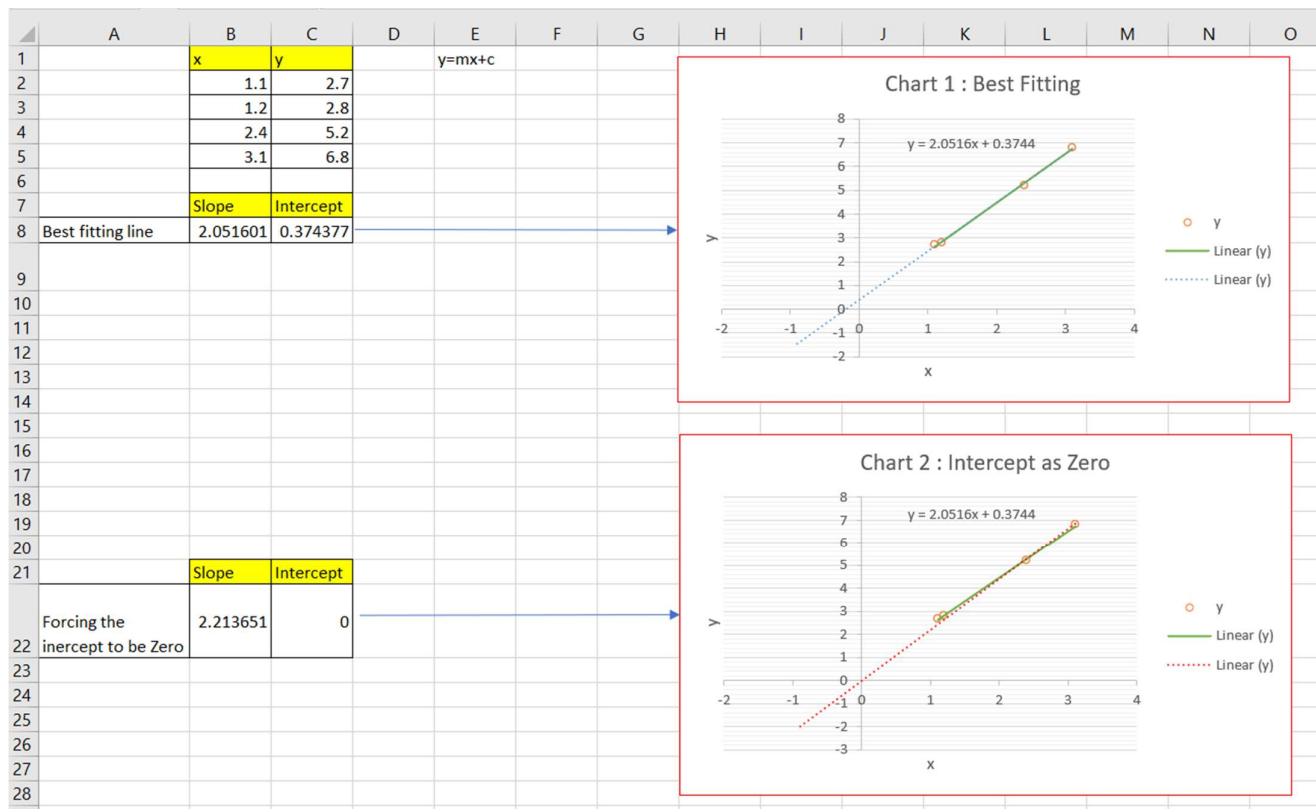


Image : Excel screenshot for Chart 1 and Chart 2

Yes, the slope changes as we tried to change the slope of the line by changing the intercept to be zero. It is also called regression through origin; we need this to interpret the real-world situation where most of the things starts from origin [3]. In other words, we are creating the variance through cluster sampling.

Question 2: ->

Exercise 2 (Matlab). Write a Matlab function that take a data matrix to find the covariance matrix (Input: data matrix and output: covariance of data matrix). You need to use the formula (in matrix form) discussed in the lectures (and not inbuilt command, cov). Use this to find the covariance matrix of the data matrix.

$$\begin{pmatrix} 1.1 & 2.7 & 3.2 & 2.1 \\ 1.2 & 2.8 & 3.4 & 1.8 \\ 2.4 & 5.2 & 1.9 & 2.5 \\ 3.1 & 6.8 & 3.5 & 2 \end{pmatrix}$$

Answer 2:

F = [1.1 2.7 3.2 2.1; 1.2 2.8 3.4 1.8; 2.4 5.2 1.9 2.5; 3.1 6.8 3.5 2]

cov_manual(F)



Without using the built-in function, we are finding the covariance of the data matrix F , using the formula discussed in class.

F be the data matrix. Then $F - 1_N \cdot \bar{X}$ is the normalized data matrix. Now

$$Q = \frac{1}{N-1} (F - 1_N \cdot \bar{X})^T (F - 1_N \cdot \bar{X})$$

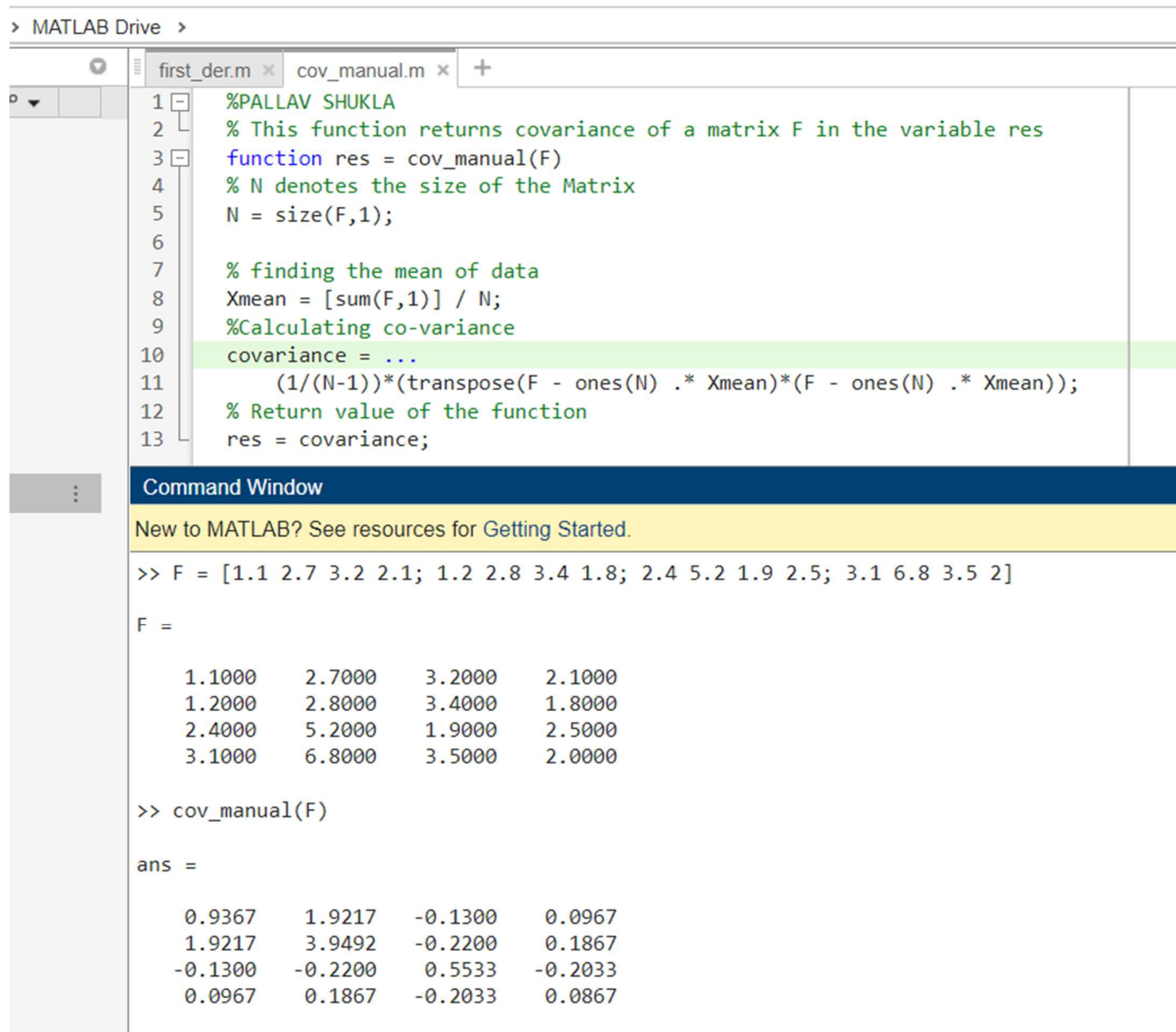
And 1_N = A square matrix of ones or all-ones matrix.

Code:

```
% CSCM 70 ----- LAB 05 -----
% CSCM 70 ----- 2154638 -----
% =====
% =====
% PALLAV SHUKLA

% ----- Covariance Matrix -----
% -----
% This function returns covariance of a matrix F in the variable res
function res = cov_manual(F)
% N denotes the size of the Matrix
N = size(F,1);
% finding the mean of data
Xmean = [sum(F,1)] / N;
%Calculating co-variance
covariance = ...
    (1/(N-1))*(transpose(F - ones(N) .* Xmean)*(F - ones(N) .* Xmean));
% Return value of the function
res = covariance;
% =====
% ----- END Covariance Matrix -----
% =====
```

OUTPUT MATLAB: COMMAND WINDOW



The image shows a MATLAB Command Window with two tabs: 'first_der.m' and 'cov_manual.m'. The 'cov_manual.m' tab is active, displaying a function definition for 'cov_manual'. The function calculates the covariance of a matrix 'F' and returns the result in 'res'. The function includes comments for each step: finding the mean, calculating the covariance, and returning the value. Below the function definition, the Command Window shows the execution of the function with a sample matrix 'F'.

```
1 %PALLAV SHUKLA
2 % This function returns covariance of a matrix F in the variable res
3 function res = cov_manual(F)
4 % N denotes the size of the Matrix
5 N = size(F,1);
6
7 % finding the mean of data
8 Xmean = [sum(F,1)] / N;
9 %Calculating co-variance
10 covariance = ...
11 (1/(N-1))*(transpose(F - ones(N) .* Xmean)*(F - ones(N) .* Xmean));
12 % Return value of the function
13 res = covariance;
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> F = [1.1 2.7 3.2 2.1; 1.2 2.8 3.4 1.8; 2.4 5.2 1.9 2.5; 3.1 6.8 3.5 2]
```

F =

1.1000	2.7000	3.2000	2.1000
1.2000	2.8000	3.4000	1.8000
2.4000	5.2000	1.9000	2.5000
3.1000	6.8000	3.5000	2.0000

```
>> cov_manual(F)
```

ans =

0.9367	1.9217	-0.1300	0.0967
1.9217	3.9492	-0.2200	0.1867
-0.1300	-0.2200	0.5533	-0.2033
0.0967	0.1867	-0.2033	0.0867

Image: 1

Comparing/Verifying with built-in function.

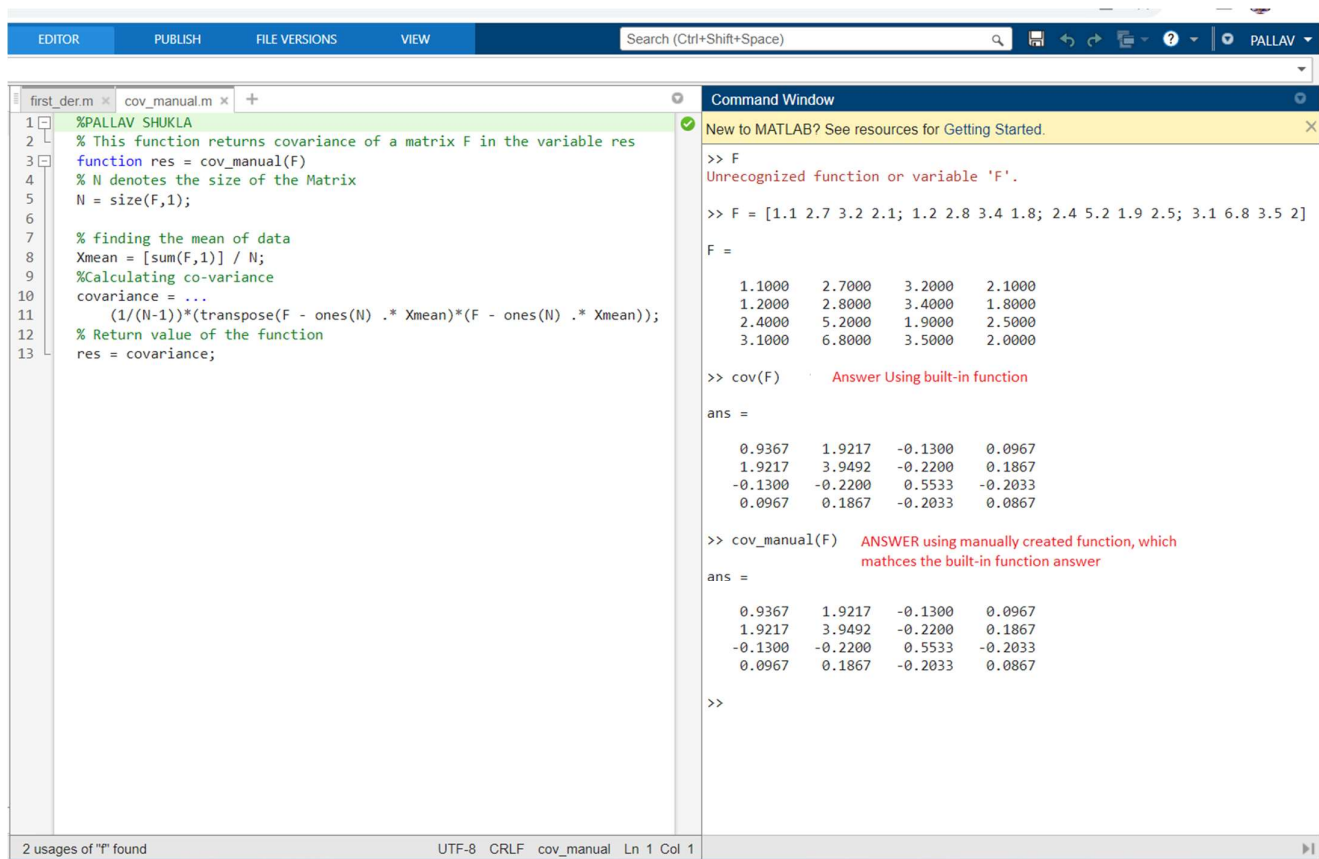


Image 2:

Question 3: ->

Exercise 3 (Matlab). Start with data $x_i = y_i = i$ for i ranging from 1 to 10. Add random perturbations of size up to 0.1 to each value. Compute the covariance matrix of the result, and obtain the eigenvectors and eigenvalues. Interpret. [Note: here you can use matlab commands for finding covariance ('cov') and eigenvalue/eigenvectors ('eig')] - 2 marks

Answer 3:

Generating the 10 random values using the formula:

➔ **CODE 1: From values - 0.1 to 0.1 as mentioned on either side.**

```
% CSCM 70 ----- LAB 05 -----
% CSCM 70 ----- 2154638 -----
% =====
% =====
% PALLAV SHUKLA

% ----- Random Perturbations -----
% -----
% fIRST Created the Matrix
>> F = [1 1; 2 2; 3 3; 4 4; 5 5; 6 6; 7 7; 8 8; 9 9; 10 10;]
% Defined Xi and Yi where Xi and Yi have a difference of .1(either side) and i has a
% range of 1 to 10
```

```

>> Xi = (0.1 + 0.1)*rand(1,10) - 0.1
>> Yi = (0.1 + 0.1)*rand(1,10) - 0.1

% Reshaped Xi and Yi
>> Xi_reshape = reshape(Xi,size(Xi,2),1)
>> Yi_reshape = reshape(Yi,size(Yi,2),1)
>> new_mat = [Xi_reshape Yi_reshape]

% Adding with F
Ultimate_mat = new_mat + F

% finding cov
>> cov(Ultimate_mat)
>> Ultimate_mat_new = cov(Ultimate_mat)

% Finding eigen
>> [V,D] = eig(Ultimate_mat_new)

% =====
% ----- END Random Perturbations -----
% =====

```

OUTPUT MATLAB: COMMAND WINDOW

```

>> F = [1 1; 2 2; 3 3; 4 4; 5 5; 6 6; 7 7; 8 8; 9 9; 10 10;]

```

```

F =

```

```

     1     1
     2     2
     3     3
     4     4
     5     5
     6     6
     7     7
     8     8
     9     9
    10    10

```

```

>> Xi = (0.1 + 0.1)*rand(1,10) - 0.1

```

```

Xi =

```

```

Columns 1 through 9

```

```

    0.0561   -0.0221   -0.0517   -0.0192   -0.0807   -0.0736    0.0884    0.0912    0.0150

```

```

Column 10

```

```

   -0.0880

```

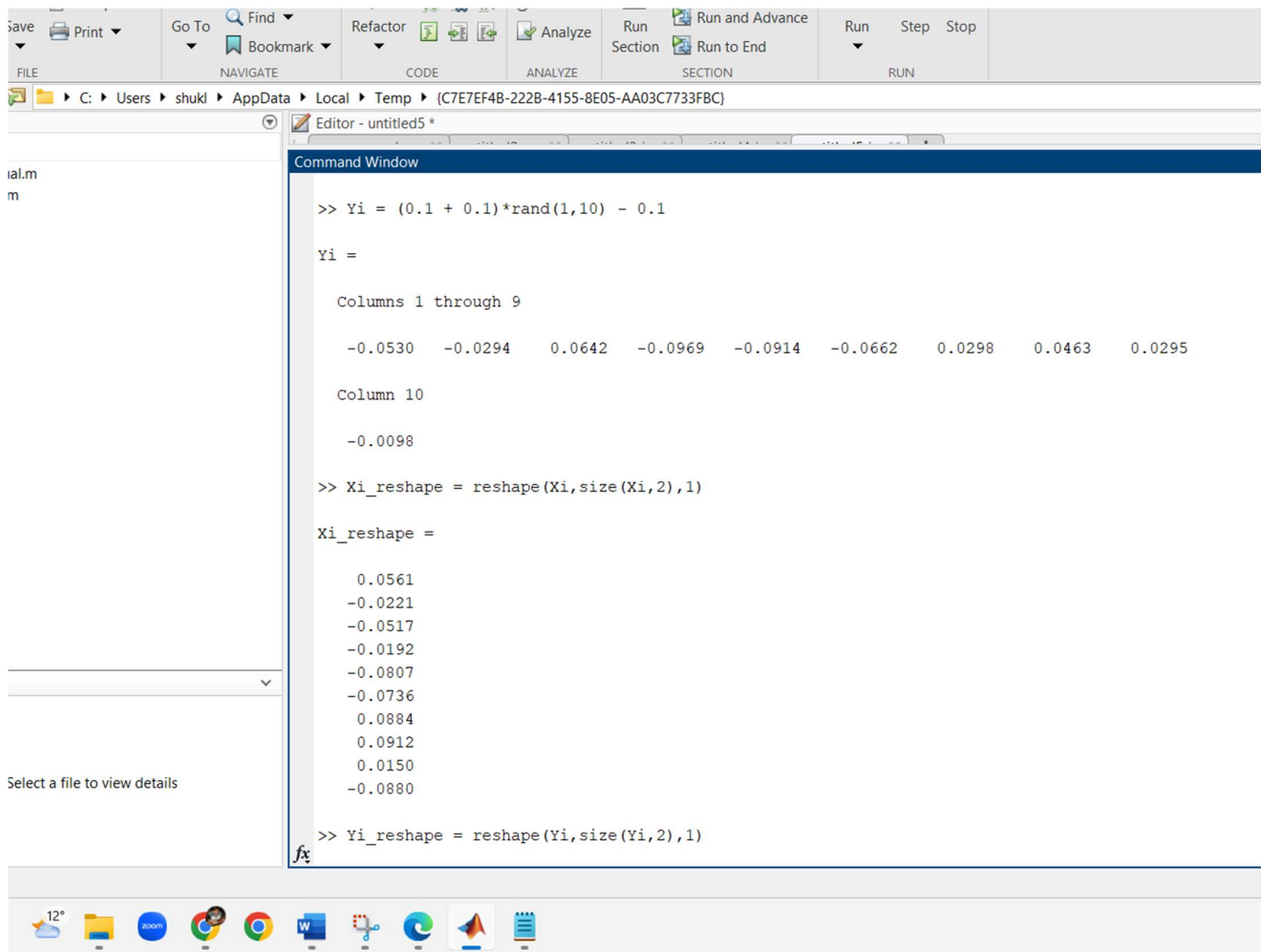


Image : 2

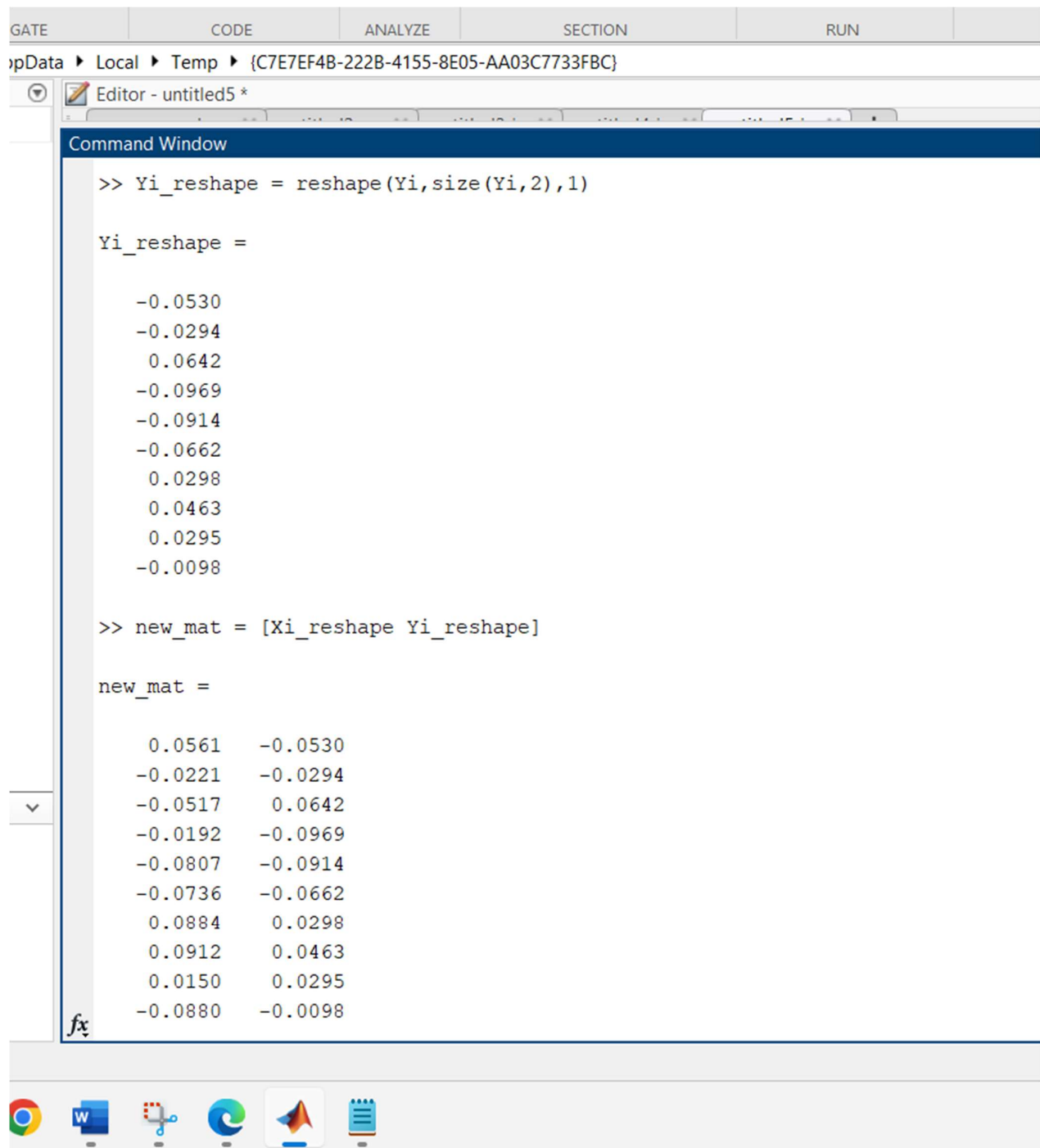


Image : 3

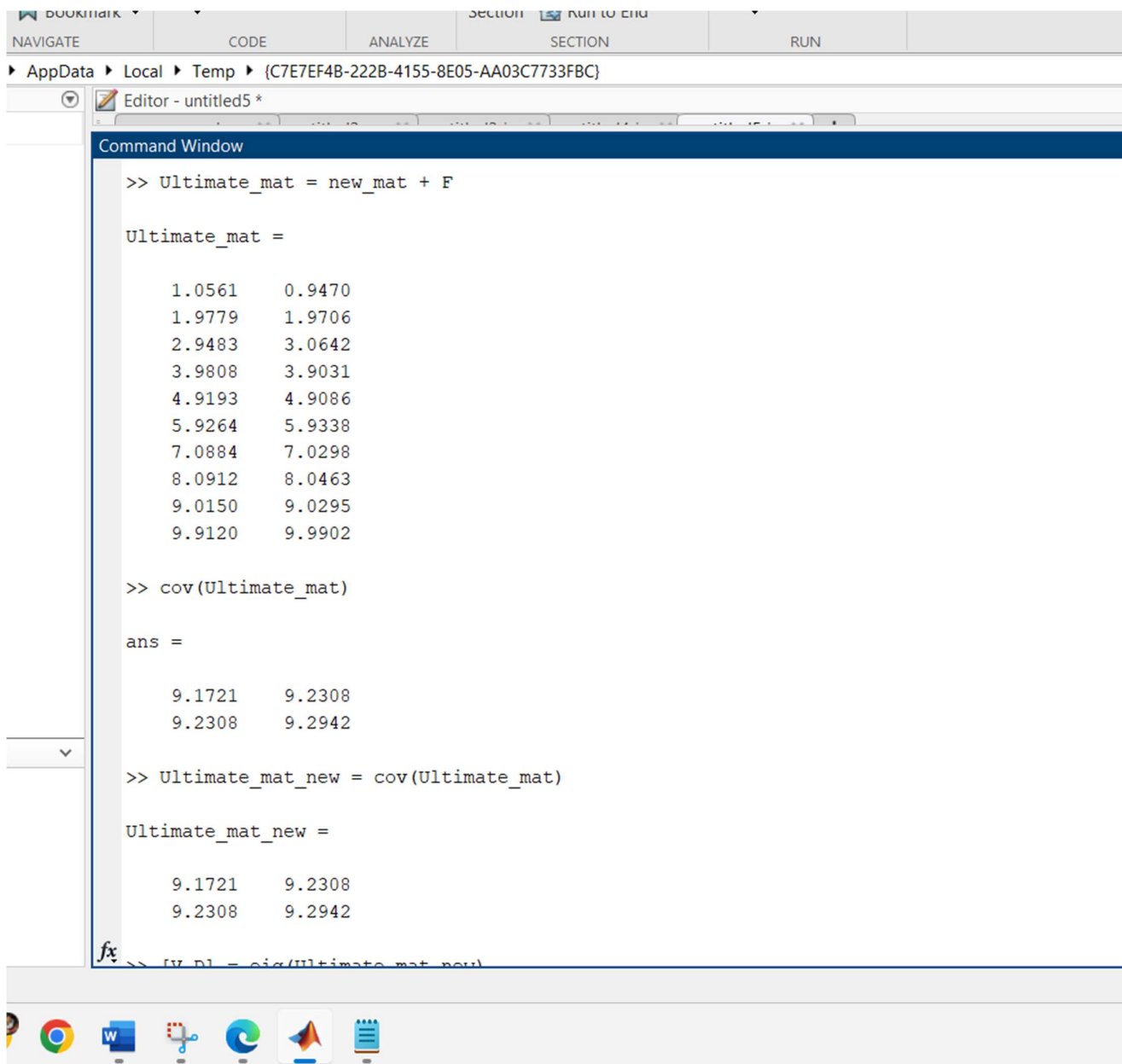


Image : 4

```
NAVIGATE CODE ANALYZE SECTION RUN
AppData Local Temp {C7E7EF4B-222B-4155-8E05-AA03C7733FBC}
Editor - untitled5 *
Command Window
9.9120 9.9902

>> cov(Ultimate_mat)

ans =

9.1721 9.2308
9.2308 9.2942

>> Ultimate_mat_new = cov(Ultimate_mat)

Ultimate_mat_new =

9.1721 9.2308
9.2308 9.2942

>> [V,D] = eig(Ultimate_mat_new)

V =

-0.7094 0.7048
0.7048 0.7094

D =

0.0022 0
0 18.4642

fx >>
```

Covariance of the matrix

eig(A) returns diagonal matrix D of eigenvalues and matrix V whose columns are the corresponding right eigenvectors, so that $A*V = V*D$.

Image : 5

Covariance is a measure of the extent to which corresponding elements from two sets of ordered data move in the same direction[4]. An eigenvector of A is a vector that is taken to a multiple of itself by the matrix transformation $T(x)=Ax$, which explains the terminology[5]. Here we have a-lot of directions and we are going for only 2 directions. In the above code First we create the F matrix, then we define Xi and Yi, as per question with constraints of 0.1 and range between 1 to 10 and then we reshape it for better understanding and adding F to the newly formed reshaped matrix then completing the answer with finding co-variance and Eigen vector and value. In total when the value changes for question it doesn't make such a big difference and if there is a change it goes for all the values in that direction.

➔ CODE 2: trying for values 0 to 0.1

```
% CSCM 70 ----- LAB 05 -----
% CSCM 70 ----- 2154638 -----
% =====
% =====
% PALLAV SHUKLA

% ----- Random Perturbations -----
% -----
% fIRST Created the Matrix
>> F = [1 1; 2 2; 3 3; 4 4; 5 5; 6 6; 7 7; 8 8; 9 9; 10 10;]

% Defined Xi and Yi where Xi and Yi have a difference of .1 and i has a
% range of 1 to 10
>> Xi = 0.1*rand(1,10)
>> Yi = 0.1*rand(1,10)

% Reshaped Xi and Yi
>> Xi_reshape = reshape(Xi,size(Xi,2),1)
>> Yi_reshape = reshape(Yi,size(Yi,2),1)
>> new_mat = [Xi_reshape Yi_reshape]

% Adding up with F
Ultimate_mat = new_mat + F

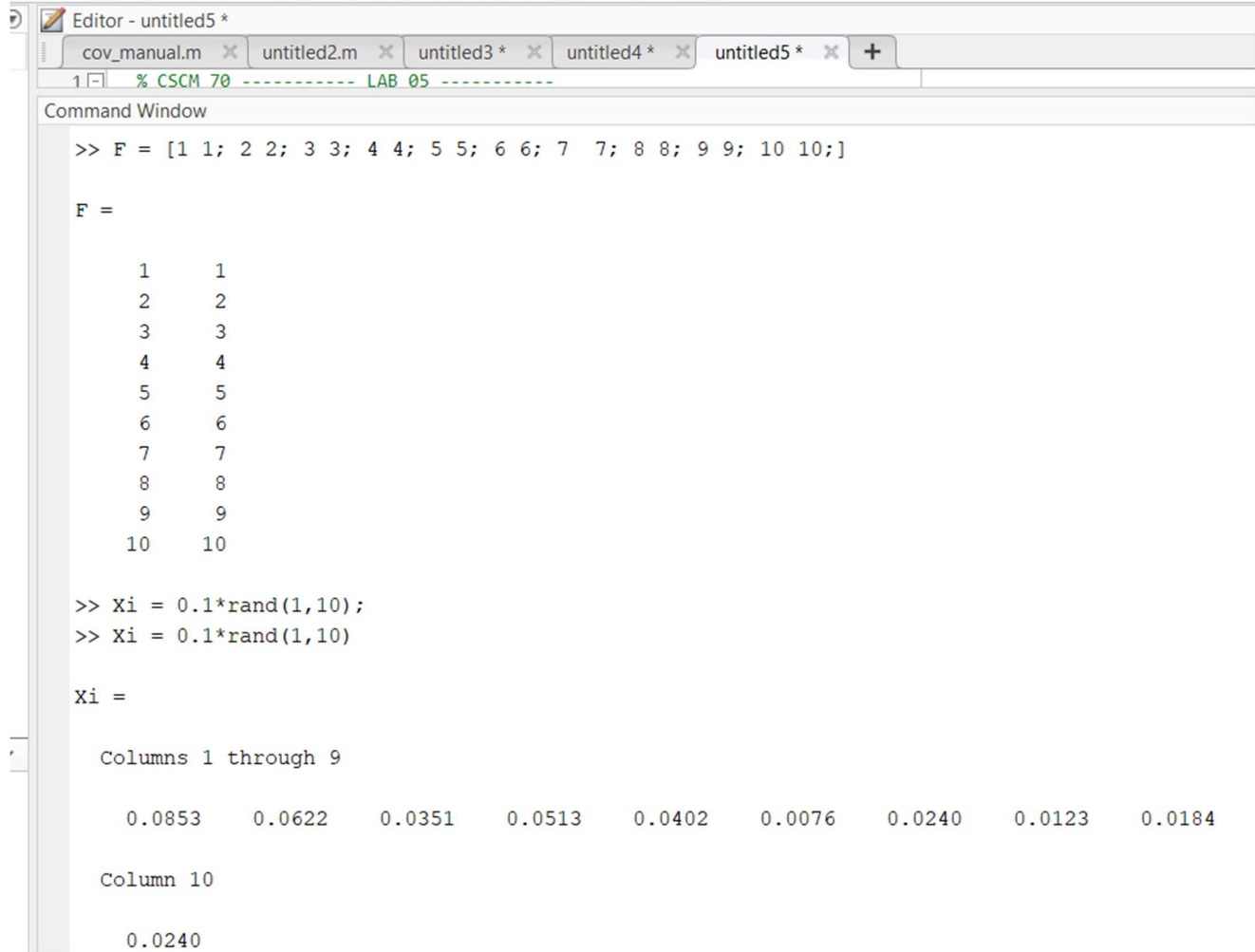
% finding cov
>> cov(Ultimate_mat)
>> Ultimate_mat_new = cov(Ultimate_mat)

% Finding eigen
>> [V,D] = eig(Ultimate_mat_new)

% =====
% ----- END Random Perturbations -----
% =====
```

OUTPUT MATLAB: COMMAND WINDOW

hukl ▸ AppData ▸ Local ▸ Temp ▸ {C7E7EF4B-222B-4155-8E05-AA03C7733FBC}



The image shows a MATLAB development environment. The top part is the 'Editor' window with the title 'Editor - untitled5 *'. It contains several tabs: 'cov_manual.m', 'untitled2.m', 'untitled3 *', 'untitled4 *', and 'untitled5 *'. The active tab is 'untitled5 *', which displays a line of MATLAB code: `1 % CSCM 70 ----- LAB 05 -----`. Below the editor is the 'Command Window'. It shows the execution of two MATLAB commands. The first command is `>> F = [1 1; 2 2; 3 3; 4 4; 5 5; 6 6; 7 7; 8 8; 9 9; 10 10;]`, which results in a 10x2 matrix `F` where each row contains two identical values from 1 to 10. The second command is `>> Xi = 0.1*rand(1,10);`, which is executed twice. The first execution results in a row vector `Xi` with 10 elements, displayed as 'Columns 1 through 9' followed by the values: 0.0853, 0.0622, 0.0351, 0.0513, 0.0402, 0.0076, 0.0240, 0.0123, and 0.0184. The second execution shows 'Column 10' with the value 0.0240.

```
Editor - untitled5 *
cov_manual.m x untitled2.m x untitled3 * x untitled4 * x untitled5 * x +
1 % CSCM 70 ----- LAB 05 -----

Command Window

>> F = [1 1; 2 2; 3 3; 4 4; 5 5; 6 6; 7 7; 8 8; 9 9; 10 10;]

F =

     1     1
     2     2
     3     3
     4     4
     5     5
     6     6
     7     7
     8     8
     9     9
    10    10

>> Xi = 0.1*rand(1,10);
>> Xi = 0.1*rand(1,10)

Xi =

Columns 1 through 9

    0.0853    0.0622    0.0351    0.0513    0.0402    0.0076    0.0240    0.0123    0.0184

Column 10

    0.0240
```

Image : 1

NAVIGATECODEANALYZESECTIONRUN

kl ▶ AppData ▶ Local ▶ Temp ▶ {C7E7EF4B-222B-4155-8E05-AA03C7733FBC}

Editor - untitled5 *

cov_manual.m ×untitled2.m ×untitled3 * ×untitled4 * ×untitled5 * ×+

11 % Defined Xi and Yi where Xi and Yi have a difference of .1 and i has a
12 % range of 1 to 10

Command Window

>> Yi = 0.1*rand(1,10)

Yi =

Columns 1 through 9

0.04170.00500.09030.09450.04910.04890.03380.09000.0369

Column 10

0.0111

>> Xi_reshape = reshape(Xi,size(Xi,2),1)

Xi_reshape =

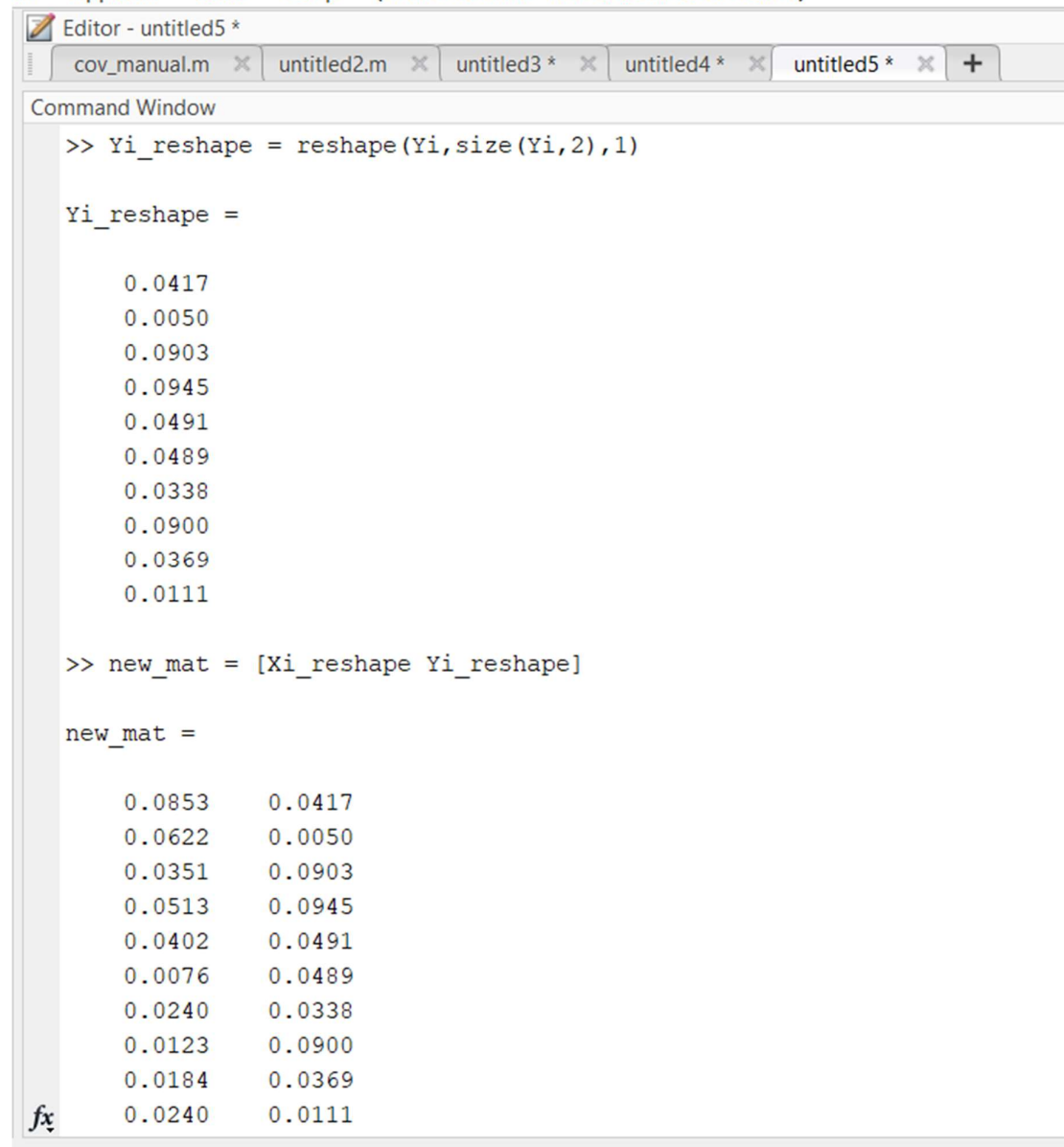
0.0853
0.0622
0.0351
0.0513
0.0402
0.0076
0.0240
0.0123
0.0184
0.0240

fx

Zoom: 90%UTF-8

Image: 2

kl ▶ AppData ▶ Local ▶ Temp ▶ {C7E7EF4B-222B-4155-8E05-AA03C7733FBC}



The image shows a MATLAB Command Window with the following content:

```
Editor - untitled5 *
cov_manual.m x untitled2.m x untitled3 * x untitled4 * x untitled5 * x +
Command Window
>> Yi_reshape = reshape(Yi,size(Yi,2),1)

Yi_reshape =

    0.0417
    0.0050
    0.0903
    0.0945
    0.0491
    0.0489
    0.0338
    0.0900
    0.0369
    0.0111

>> new_mat = [Xi_reshape Yi_reshape]

new_mat =

    0.0853    0.0417
    0.0622    0.0050
    0.0351    0.0903
    0.0513    0.0945
    0.0402    0.0491
    0.0076    0.0489
    0.0240    0.0338
    0.0123    0.0900
    0.0184    0.0369
    0.0240    0.0111
```

fx

Image: 3

```
hukl ▸ AppData ▸ Local ▸ Temp ▸ {C7E7EF4B-222B-4155-8E05-AA03C7733FBC}
Editor - untitled5 *
cov_manual.m x untitled2.m x untitled3 * x untitled4 * x untitled5 * x +
Command Window

>> Ultimate_mat = new_mat + F

Ultimate_mat =

    1.0853    1.0417
    2.0622    2.0050
    3.0351    3.0903
    4.0513    4.0945
    5.0402    5.0491
    6.0076    6.0489
    7.0240    7.0338
    8.0123    8.0900
    9.0184    9.0369
   10.0240   10.0111

>> cov(Ultimate_mat)

ans =

    9.0465    9.0931
    9.0931    9.1415

>> Ultimate_mat_new = cov(Ultimate_mat)
```

Image: 4

The image shows a MATLAB Command Window with the following code and output:

```
>> cov(Ultimate_mat)

ans =

    9.0465    9.0931
    9.0931    9.1415

>> Ultimate_mat_new = cov(Ultimate_mat)

Ultimate_mat_new =

    9.0465    9.0931
    9.0931    9.1415

>> [V,D] = eig(Ultimate_mat_new)

V =

   -0.7090    0.7053
    0.7053    0.7090

D =

    0.0008         0
         0   18.1872
```

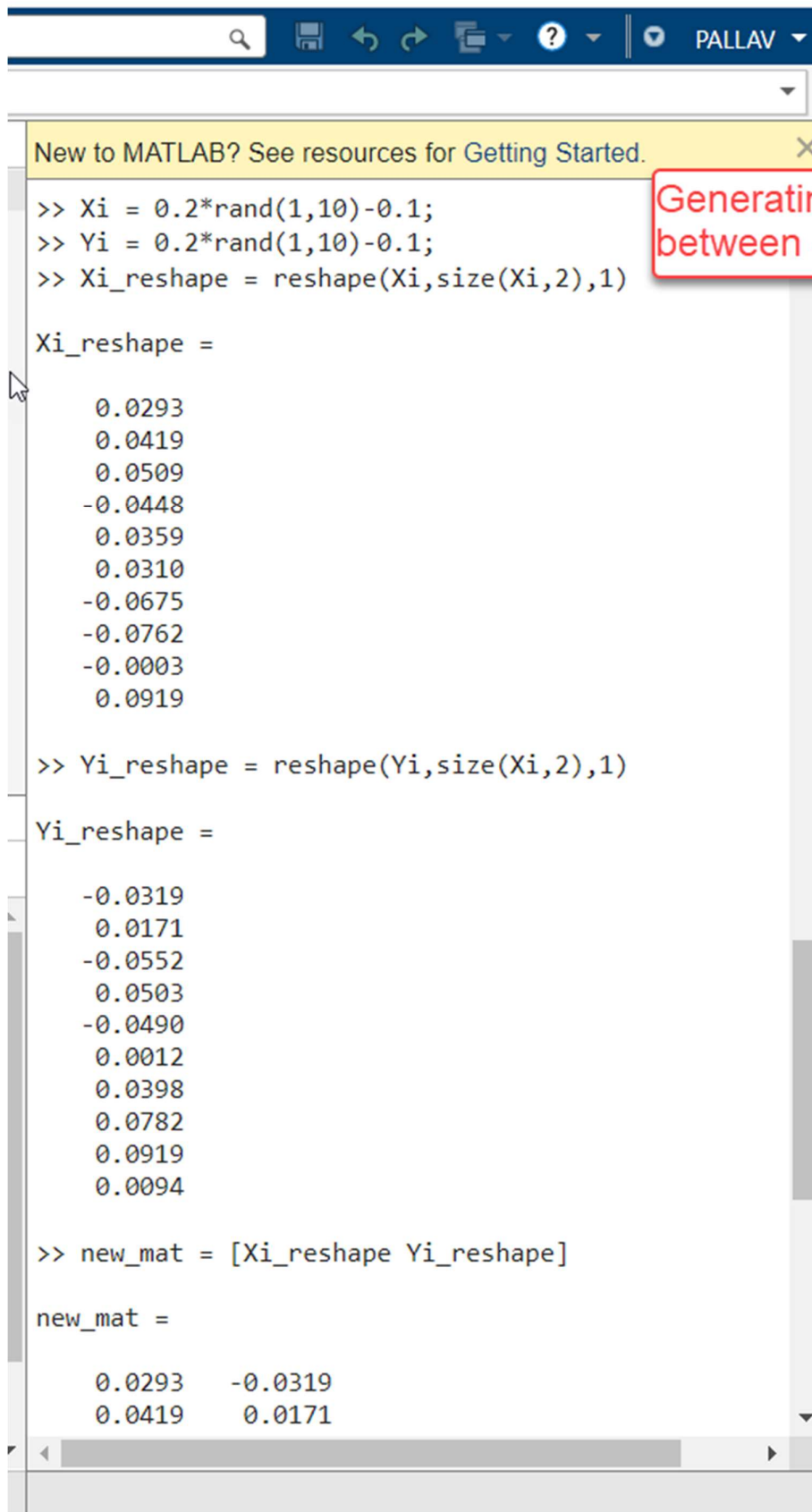
Annotations in red boxes:

- Covariance of the matrix** (next to the output of `cov(Ultimate_mat)` and `Ultimate_mat_new`)
- eig(A) returns diagonal matrix D of eigenvalues and matrix V whose columns are the corresponding right eigenvectors, so that $A*V = V*D$.** (next to the output of `[V,D] = eig(Ultimate_mat_new)`)

The Command Window prompt is `fx >> |`.

Image: 5

➔ Tried for the range of -0.1 to 0.1 without generating F-> Next page

A screenshot of the MATLAB Command Window. At the top, there is a toolbar with icons for search, save, undo, redo, copy, paste, and help, followed by the name 'PALLAV'. Below the toolbar is a yellow banner that reads 'New to MATLAB? See resources for Getting Started.' with a close button. The main area of the window contains MATLAB code and its output. The code generates two random vectors, Xi and Yi, and reshapes them into column vectors. A red callout box points to the first two lines of code. The output shows the values of Xi_reshape, Yi_reshape, and the resulting matrix new_mat.

```
>> Xi = 0.2*rand(1,10)-0.1;
>> Yi = 0.2*rand(1,10)-0.1;
>> Xi_reshape = reshape(Xi,size(Xi,2),1)

Xi_reshape =

    0.0293
    0.0419
    0.0509
   -0.0448
    0.0359
    0.0310
   -0.0675
   -0.0762
   -0.0003
    0.0919

>> Yi_reshape = reshape(Yi,size(Xi,2),1)

Yi_reshape =

   -0.0319
    0.0171
   -0.0552
    0.0503
   -0.0490
    0.0012
    0.0398
    0.0782
    0.0919
    0.0094

>> new_mat = [Xi_reshape Yi_reshape]

new_mat =

    0.0293   -0.0319
    0.0419    0.0171
```

Generating random numbers
between -0.1 and 0.1

Then finding the eigen values and eigen vectors

```
0.0919
0.0094

>> new_mat = [Xi_reshape Yi_reshape]

new_mat =
    0.0293   -0.0319
    0.0419    0.0171
    0.0509   -0.0552
   -0.0448    0.0503
    0.0359   -0.0490
    0.0310    0.0012
   -0.0675    0.0398
   -0.0762    0.0782
   -0.0003    0.0919
    0.0919    0.0094

>> cov_x_y = cov(new_mat)

cov_x_y =
    0.0030   -0.0018
   -0.0018    0.0026

>> [V,D] = eig(cov_x_y)

V =
   -0.6599   -0.7513
   -0.7513    0.6599

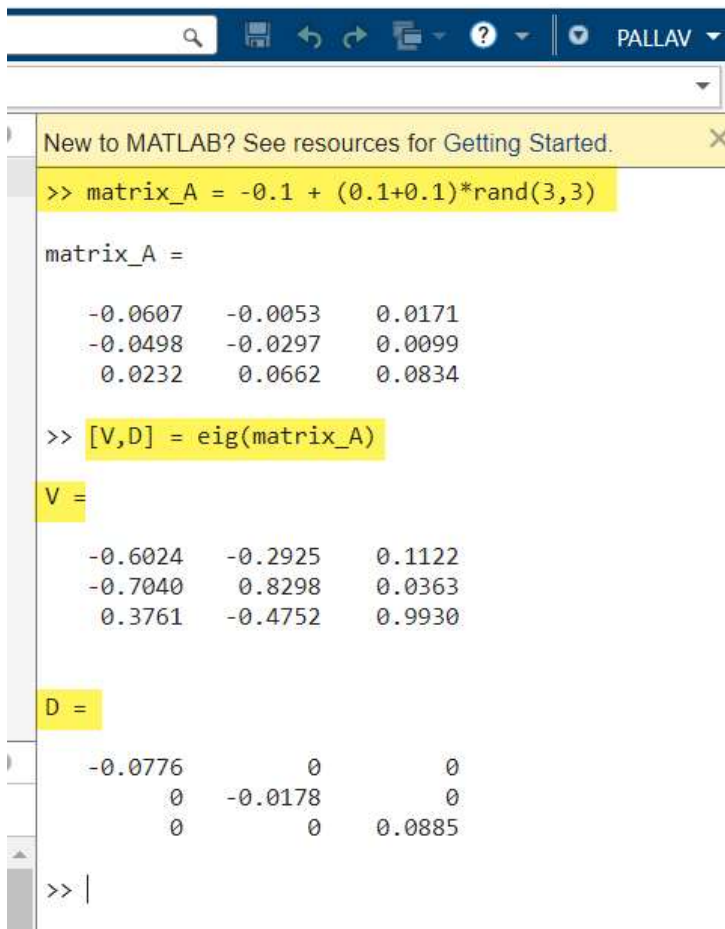
D =
    0.0010         0
         0    0.0046

>> |
```

Covariance of the matrix

`eig(A)` returns diagonal matrix `D` of eigenvalues and matrix `V` whose columns are the corresponding right eigenvectors, so that $A*V = V*D$.

→ Trying eigen function on randomly generated (3,3) matrix:



```
>> matrix_A = -0.1 + (0.1+0.1)*rand(3,3)

matrix_A =

   -0.0607   -0.0053    0.0171
   -0.0498   -0.0297    0.0099
    0.0232    0.0662    0.0834

>> [V,D] = eig(matrix_A)

V =

   -0.6024   -0.2925    0.1122
   -0.7040    0.8298    0.0363
    0.3761   -0.4752    0.9930

D =

   -0.0776         0         0
         0   -0.0178         0
         0         0    0.0885

>> |
```

References:

- 1.) Sir's Lecture.
- Panopto. Panopto. Retrieved November 10, 2022, from
<https://swanseauniversity.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=164ff299-6a2c-4aa2-af57-af4600c7f45c>
- 2.) Canvas Panopto. Panopto. Retrieved November 10, 2022, from
<https://swanseauniversity.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=164ff299-6a2c-4aa2-af57-af4600c7f45c>
- 3.) Intercept to zero .
When should we force the intercept to zero? / Researchgate. Retrieved November 14, 2022, from
<https://www.researchgate.net/post/When-should-we-force-the-intercept-to-zero>

4.) Covariance

Berman H.B., "*Variance-Covariance Matrix*", [online] Available at: <https://stattrek.com/matrix-algebra/covariance-matrix> URL [Accessed Date: 11/14/2022].

5.) EigenVector

Dan Margalit, J. R. *Interactive linear algebra*. Eigenvalues and Eigenvectors. Retrieved November 14, 2022, from <https://textbooks.math.gatech.edu/ila/eigenvectors.html>

End

+++++

Lab 005

+++++

+++++

2154638

+++++