Module One - Vectors, Lines, Planes, and Quadratic Surfaces

MAT325: Calculus III: Multivariable Calculus

Student Name

Date

Introduction:

Vectors, lines, planes, and quadratic surface are some of the basic quantities studied in Module One. Being able to plot, visualize, and perform computations with these types of objects is important, and MATLAB has a variety of functions that are useful for this purpose.

In this MATLAB assignment, you'll learn how to define vectors, plot vectors, compute dot products, compute cross products, and plot lines, planes, and surfaces. Many of the plotting skills introduced in this module should be helpful throughout the course.

Review the code and comments provided in the "Examples" section below, and then use this information to complete the problems listed in the "Problems" section.

Make sure to run your code so all relevant computations/results are displayed, delete the "Introduction" and "Examples" sections, and then export your work as a PDF file for submission (your submission only needs to contain the "Problems" section that you completed).

Examples:

Example 1 - Vectors

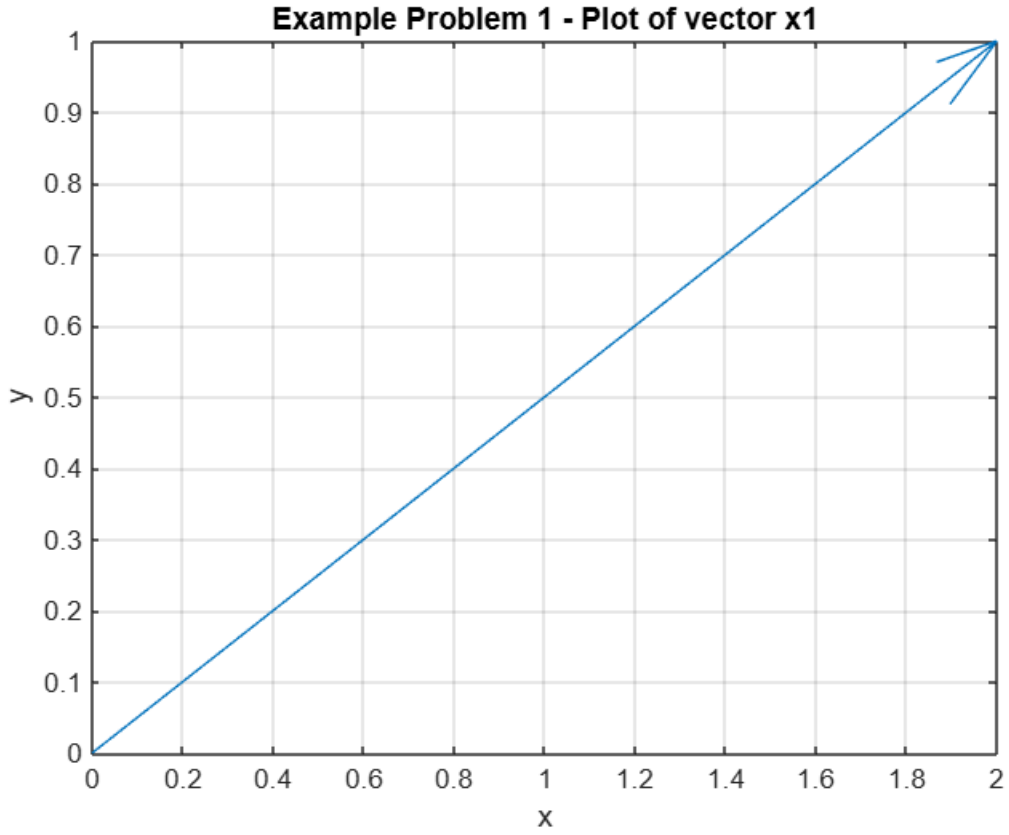
```
% Example 1 Code - Vectors

% Row vectors can be defined be listing numbers within square brackets.
% For example, the code below creates a 2-dimensional vector x1 = <2, 1>
% as well as a 3-dimensional vector x2 = <1, 0.5, -2>

x1 = [2 1]
```

```
x1 = 1×2
     2     1
```

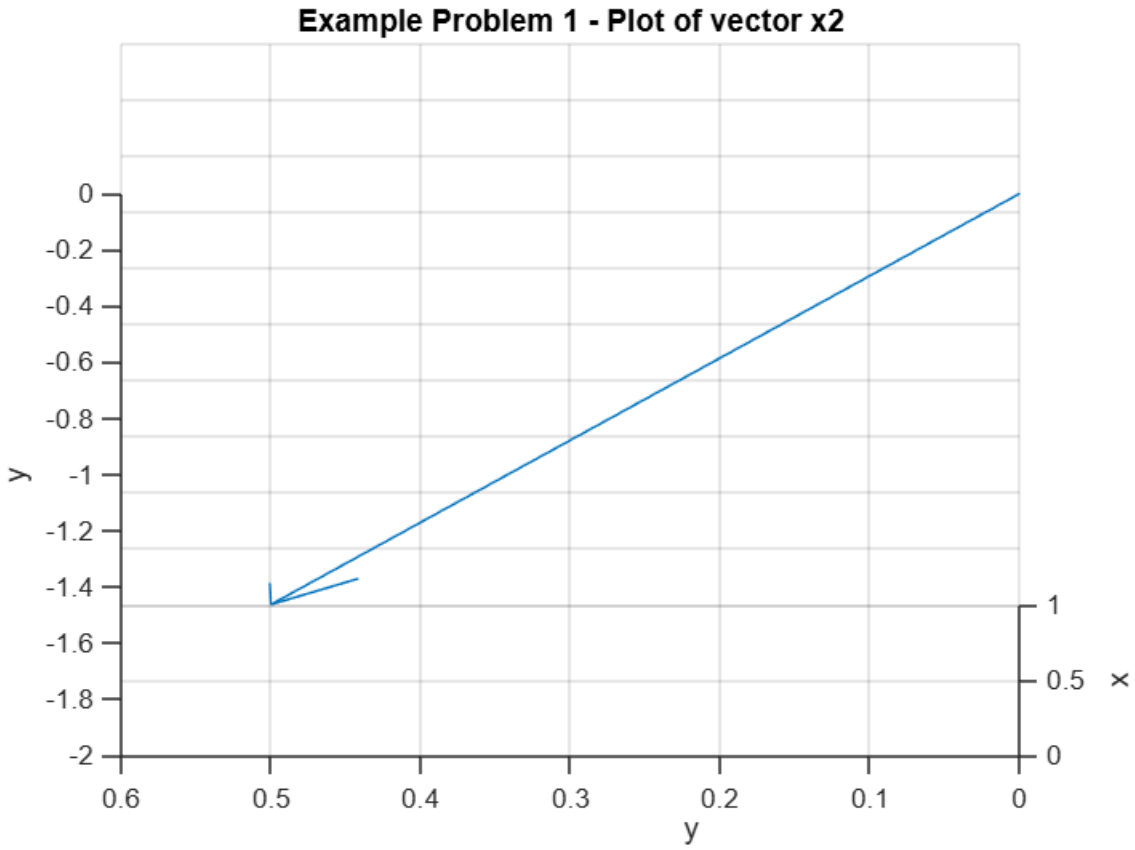
```
x2 = [1 0.5 -2]
```

```
x2 = 1×3
    1.0000    0.5000   -2.0000
```

```
% Note, the lines above were not "terminated" with a semi-colon ";".
% Terminating with ";"  keeps the line from outputing to the display and
% sometimes this is desired. However, if you do want to see the value, remove
% the semi-colon to allow the value to print out. For example, the quantity
z has
% been defined below, but since the line of code ended with a semi-colon, it
% is not printed out when the script is run.
z = [1 2 3];

% The quiver() function can be used to plot 2-dimensional vectors, and the
% quiver3() function can be used to plot 3-dimensional vectors. The code
% below shows an example of plotting the vectors x1 and x2 defined above.

% First, let's make a plot of x1. This vector starts at the origin (0,0),
% so the first inputs to the quiver() function will always be 0,0.
% While sometimes useful, the final 'off' argument ensure automatic scaling
% is disabled.
figure;
quiver(0,0,x1(1),x1(2),'off');

% It's usually best practice to title your figure and label the axes. The
% xlabel(), ylabel(), and title() functions are used for this.
xlabel('x');
ylabel('y');
title('Example Problem 1 - Plot of vector x1');

% Plotting on a grid often improves the look of a plot. The "grid on"
% command will turn on a grid.
grid on;
```




```
% We can repeat similar steps to create a 3-D plot of vector x2.
% The view() command is useful for 3D plots to set a useful viewing angle
% of the plot. The "hold on" command can be used if more plots are going to
% be made to the same figure.
figure;
quiver3(0,0,0,x2(1),x2(2),x2(3),'off');
hold on;
xlabel('x');
ylabel('y');
zlabel('y');
title('Example Problem 1 - Plot of vector x2');
grid on;
view([-90 15]);
```



Example 2 - Dot and Cross Products

```
% Example 2 Code - Dot and Cross Products

% MATLAB has functions dot() and cross() that can be used to easily compute
% the dot product or cross product between two vectors.

% Consider the vectors
x1 = [1 0 0];
x2 = [0 1 0];
x3 = [0 0 1];

% We see these vectors are just the standard basis vectors for R3. x1 lies
% perfectly along the x-axis, x2 along the y-axis, and x3 along the z-axis.
% We compute the dot product between x1 and x2 as
dot(x1,x2)
```

```
ans =
0
```

```
% We compute the cross product between x1 and x2 as
cross(x1,x2)
```

```
ans = 1×3
```

```
     0     0     1
```

```
% Not surprisingly, the dot product is zero since these vectors are at
% a right angle (i.e. orthogonal to each other). Similarly, the cross
% product between x1 and x2 is [0 0 1], the vector x3.
```

Example 3 - Lines and Planes

```
% The "clear all" command will clear all workspace variables. This is often
% useful if you'd like to reset your workspace, or if you might be
% re-defining quantities that have already been previously defined.
clear all;

% The "syms" command is used to define symbolic variables within the MATLAB
% workspace. You can define all symbols in a single list.
syms t;

% Consider the parametric equations for a line on the interval -1 <= t <=1
% given by: x(t) = 3t+2, y(t) = 1-t, and z(t) = 2t+4.
% We can easily define these parametric equations in MATLAB.
% For example, defining x(t) as function of the variable "t", the quantity
x(t)
% will now be a symbolic function within the MATLAB workspace. We can make
% other definitions for y(t) and z(t).
x(t) = 3*t +2
```

$$x(t) = 3t + 2$$

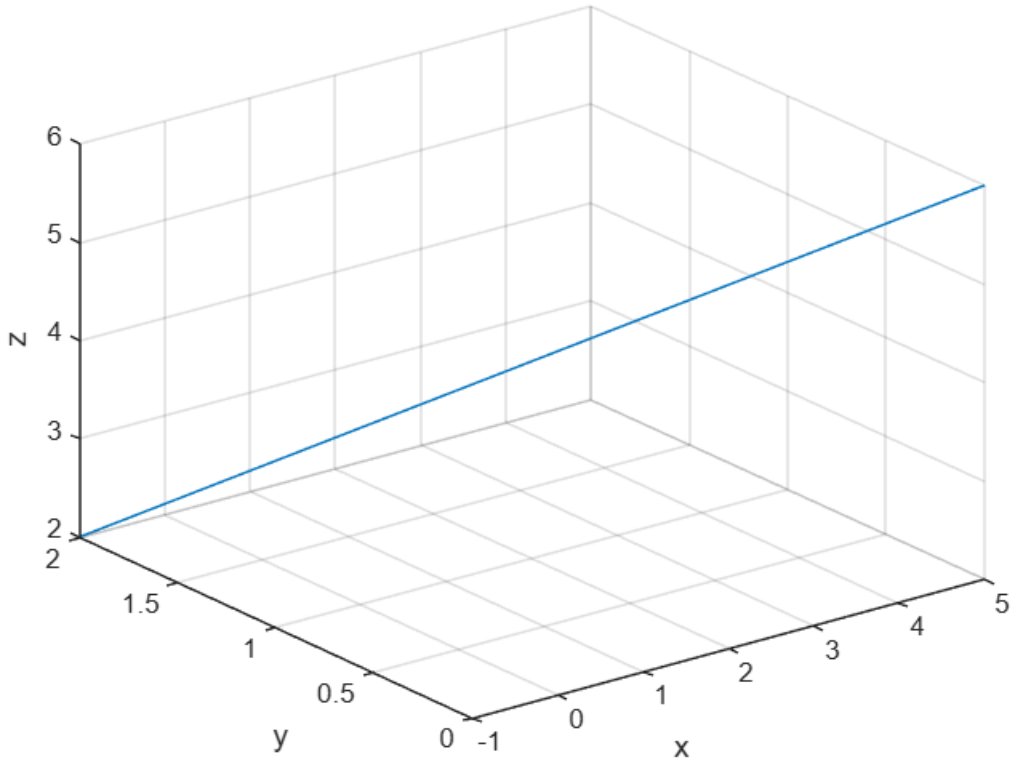
```
y(t) = 1-t
```

$$y(t) = 1 - t$$

```
z(t) = 4+2*t
```

$$z(t) = 2t + 4$$

```
% The "fplot3" function plots a symbolic function over some interval. The
% second argument [-1,1] tells fplot3 to plot from t = -1 to t = 1.
figure;
fplot3(x(t),y(t),z(t),[-1,1]);
xlabel('x');
ylabel('y');
zlabel('z');
```

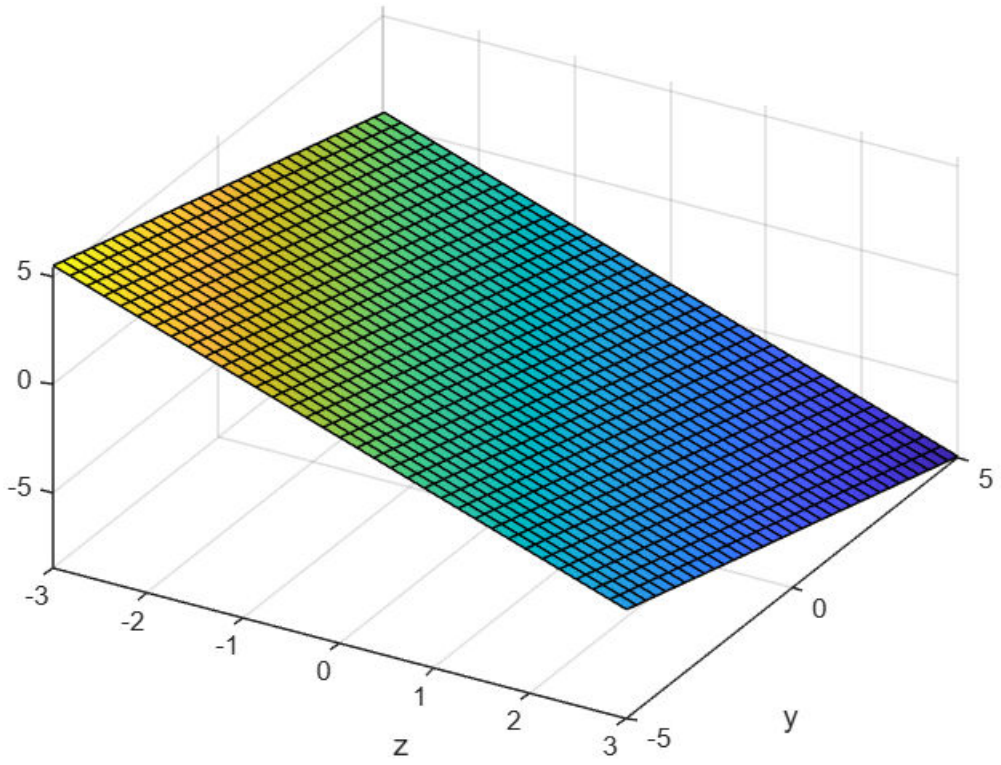



```
% Similar methods can be used to plot surfaces or planes via the fsurf3()
% function. For example, consider the plane defined by the equation:
% 3x + y + 2z + 3 = 0.
%
% Solving for z yields z = (0.5)*(-3x-y-3);

% Define needed symbols
syms x y;

% Compute z
z = 0.5*(-3*x - y - 3)
```

```
z =
```

$$-\frac{3x}{2}-\frac{y}{2}-\frac{3}{2}$$

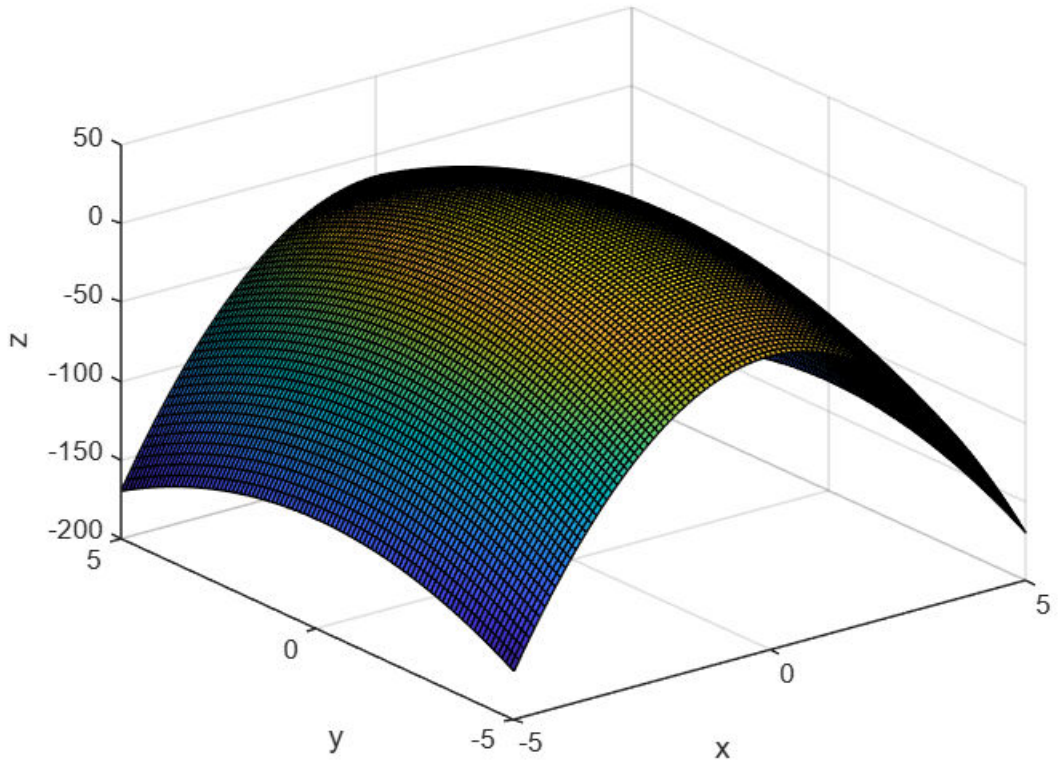
```
% Plot the plane
% The second argument sets bounds [xMin, xMax, yMin, yMax]
figure
fsurf(z,[-3,3,-5,5]);
xlabel('x');
ylabel('y');
xlabel('z');
view([30 45]);
```



Example 4 - Surfaces

```
% Using symbolic functions as in Example 3 works well when a simple
% equation can be found for the line, plane, or surface. In some cases,
% this isn't as simple as desired, but numerical methods may still be
% useful. The code below shows an example of using a meshgrid() to construct
% sets of x-y points that can then be evaluated using surf(). This approach
% can be more computationally expensive, but might offer flexibility when
% functions have a more complicated form.

x = linspace(-5,5,100); %100 points from -5 to 5
y = linspace(-5,5,100); %100 points from -5 to 5

% compute all combinations of x y points via meshgrid function
[X,Y] = meshgrid(x,y);

% compute quadratic surface z = 25(1 - x^2/4 - y^2/16);
Z = 25*(1 - (X.^2)./4 - (Y.^2)./16);

figure;
surf(X,Y,Z);
xlabel('x');
ylabel('y');
```

```
zlabel('z');
```



Problems:

Problem 1: Use MATLAB to define the vectors $\mathbf{x}_1 =< 1, 2, -4 >$, $\mathbf{x}_2 =< 2, 0, 1 >$, and $\mathbf{x}_3 =< -1, -2, 3 >$. Use the quiver3() function to plot all vectors on a single figure. Set an appropriate view in your plot, label all axes, and title your figure. Also, compute the dot product between vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, and compute the cross product between vectors $\mathbf{x}_2$ and $\mathbf{x}_3$.

```
% Problem 1 Code Here
x1 = [ 1, 2, -4 ];
x2 = [ 2, 0, 1 ];
x3 = [ -1, -2, 3];

quiver3(0, 0, 0, x1(1), x1(2), x1(-4));
hold on;
quiver3(0, 0, 0, x2(2), x2(0), x2(1));
quiver3(0, 0, 0, x3(-1), x3(-2), x3(3));
hold off;
grid on;

xlabel('x');
ylabel('y');
```

```
zlabel('z');
title('Problem 1 - Graph');

dot(x1, x2);
cross(x2, x3);
```

Problem 2: Consider the two planes defined by the equations $2x + y + z = 3$ and $x - 4y + 3z = 2$. Use MATLAB and the surf() or fsurf() function to plot the two planes in the same figure for the interval $x \in [-3, 3]$ and $y \in [-3, 3]$. Define normal vectors for each plane and compute the angle (in degrees) between the two planes.

```
% Problem 2 Code Here
x = linespace(-3, 3, 100);
y = linespace(-3, 3, 100);

[X, Y] = meshgrid(x, y);

Z1 = 3- 2* X- Y;
Z2 = (2- X+ 4* Y)/ 3

figure;
fsurf(@(x, y) 3- 2* x- y, [-3, 3, -3, 3]);
hold on;
fsurf(@(x, y)(2- x+ 4* y)/ 3, [-3, 3, -3, 3]);

xlabel('x');
ylabel('y');
zlabel('z');

view(30,40);
grid on;

title('Problem 2 - Graph');

n1 = [2, 1, 1];
n2 = [1, -4, 3];
cosTheta = dot(n1, n2) / norm(n1) * norm(n2));
theta = acos(cosTheta);
```

Problem 3: Consider the quadratic surface defined by the equation $z = \frac{x^2}{9} - \frac{y^2}{16}$.

Use MATLAB and the surf() or fsurf() function to plot the surface over the interval

$x \in [-4, 4]$, $y \in [-3, 3]$. **Choose an appropriate view to best visualize the surface and label axes appropriately.**

```
% Problem 3 Code Here
x = linespace(-4, 4, 100);
y = linespace(-3, 3, 100);

[X, Y] = meshgrid(x,y);
Z = X^2/9 - Y^2/16;

figure;
surf(X, Y, Z);

xlabel('x');
ylabel('y');
zlabel('z');

view(30, 45);
grid on;

title('Proble 3 - Graph');
```