EE202 Lab#6   3D Graphics

How to Copy your Figures to Word

1. Save your graph in a format than can be imported into Word (e.g., bmp, jpeg).
2. To save time you can simple Select Edit => Copy Figure and paste directly into Word. When you turn in your lab please insert each picture after the m-file of command line(s) that generated it.
3. To save even more time…Question 1 will show you a way to directly print your M-File along with its output.

How to Organize and Clean up Your Work

To clean up past work, place the following line at the beginning of your M-File(s).

```
clear,clc, close all  %  The close all command closes all figure windows
```

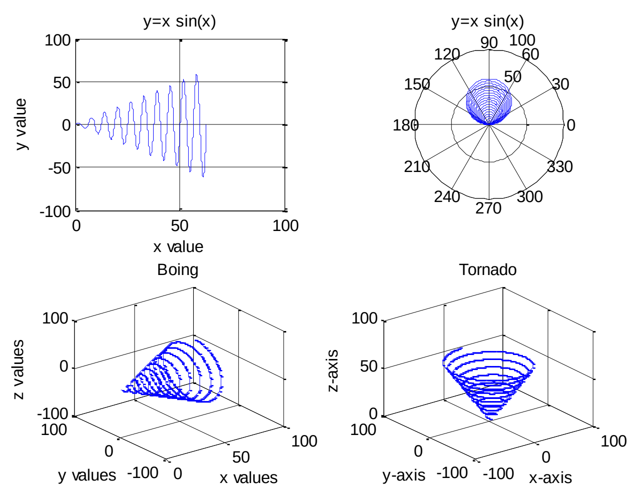
To make more than one figure in an M file, use the function `figure(n)`, where **n** is the next figure to be drawn.

1. [8.2] Make a new M-File named Tornado. Add the following comment at the beginning of your script. %% Lab 6 Question 1 Tornado

   Create a vector x of values from 0 to 20 π, with a spacing of π/100. Define vectors y and z as:

   $y = x\sin(x)$

   $z = x\cos(x)$

   Using the subplot function, graph the solution to the following four questions on a single figure as shown here.

   (a) Create an x-y plot of x and y. Turn grid on and specify solid gridlines.

   (b) Create a polar plot of x and y. To get the solid gridlines shown in the figure you will need to use a custom polar function. Download from the class website the `polarEE202` M-file and place in your current directory. Now in place of the polar function type: `polarEE202(x,y)`

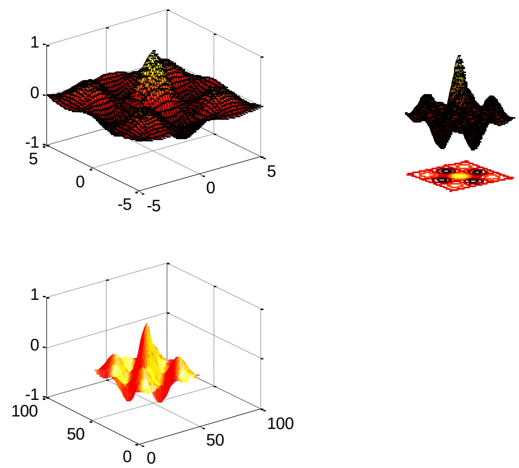
   (c) Using the plot3 functions, create a three-dimensional line plot of x, y, z. Don't forget a title, labels, and solid gridlines (see Sample Figure).

   (d) Figure out how to adjust your input order to the plot3 function to create a graph that looks like a tornado. Hint: with three inputs (x, y, z) there are six unique input combinations possible, you have already tried one - so you have a 20% chance of getting it right with the first guess.

   (e) Just for fun, once you can create a tornado, replace the plot3 function with the comet3 function. Do not turn in this solution.

Matlab can format your M-file and the resulting outputs for publishing (i.e., a format you can turn in with your lab) by selecting File → Publish To →Word Document. If the conversion fails then try File → Publish To HTML, open in IE or Firefox and convert to PDF or simply print.

2. [8.2] Make a new M-File named Surf3Dplots. Add the following comment at the beginning of your script. %% Lab 6 Question 2 Surf 3D Plots. Create a vector u of values from -5 to 5, with a spacing of 0.2 units. Define vector [x,y] as a mesh grid of u (`[x,y] = meshgrid(u, u)`). Read page 102 of your textbook to learn more about the meshgrid function. Using the subplot function, make three 3D surface plots as described below, of the function:

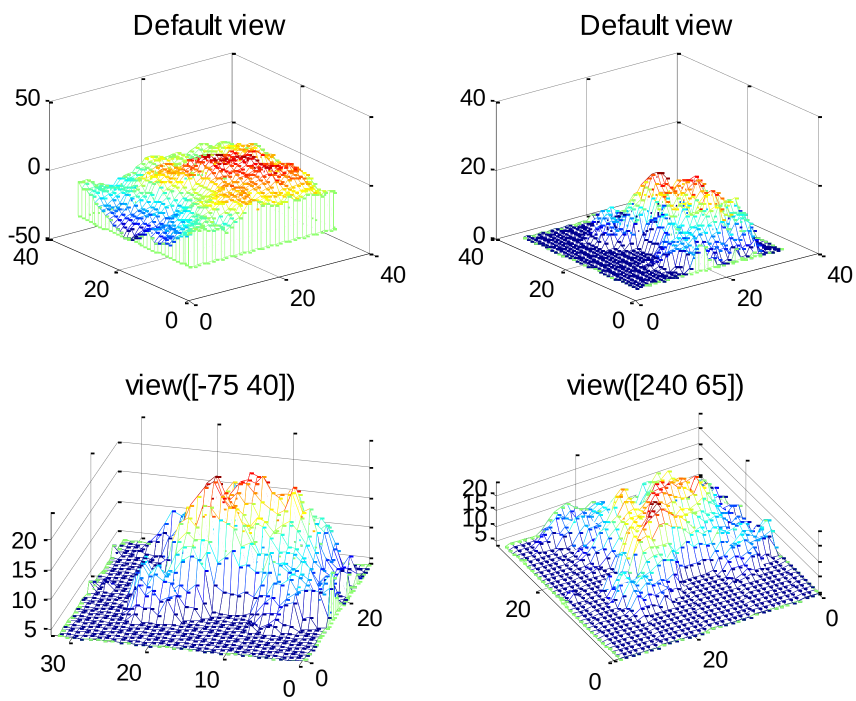


$$z = \cos x \cos y \, e^{\frac{-\sqrt{x^2+y^2}}{4}}$$
$$|x| \leq 5, |y| \leq 5$$

```
(a)    surf(x,y,z)
(b)    surfc(z)
       view(-37.5,20)
       axis('off')
(c)    surfl(z)
       shading interp
       colormap hot
```

Publish the M-File and resulting output to a Word or HTML document as previously described (see end of question 1).

3. Make a new M-File named Bisland. Add the following comment at the beginning of your script. %% Lab 6 Question 3 Bisland. Create the Bisland Fractal landscape defined on pages 108 to 109 of the textbook. To save you time I have included in the lab6 folder the **land** function. After entering the code provided by the author, verify that you are getting the same figure shown in Figure 8.19 and reproduced here. Next, create your own landscapes by commenting out the first line in the Bisland M-File (`randn('state'10);`). Publish the M-File and resulting output to a Word or HTML document as previously described (see end of question 1).



4. In this problem we are going to have some fun with 3D objects. Make a new M-File named Cylinder_3D. Add the following comment at the beginning of your script. %% Lab 6 Question 4 Cylinder 3D. I would recommend as your first line, code to clear variables and functions from memory; erase the command window; and close all the open figure windows.

```
clear,clc, close all %  The close all command closes all figure windows
```

Next, define a linear array **z** containing 101 points equally spaced at interval 0.01 along a unit height (i.e., 0 to 1). You may use colon notation or the `linspace` function. The spacing (0.01) will determine the resolution of our 3D object along the z-axis. Now all we need is the outline of our to-be-drawn 3D object.

```
r = sin(3*pi*z)+1.5; % define radius at each point
```

We will use this sine wave function to define the radius of our 3D object, at each point along the z-axis. Let us take an initial look at what the basic shape of our 3D object will look like.

(a) Add the following comment at this point in your script. %% (a) Sinewave. During the creation of our 3D objects we will make four (4) figures.

```
figure(1)              % shape outline
```

Applying what you learned in the 2D lab, plot a line along the unit height **z** on the y-axis and the radius at each point (**r**) on the x-axis. Set the `axis` as `square` and the range from 0 to 2.5 along the x-axis and from 0 to 1 along the y-axis. Do not forget to title and label the axis of your figure.

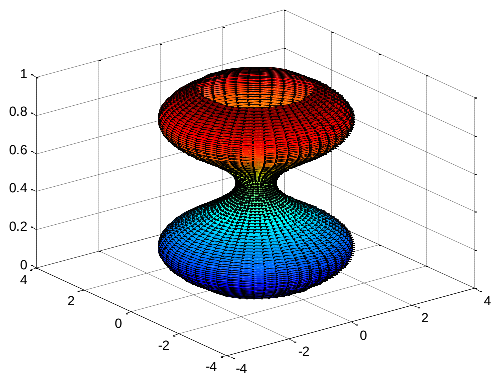
(b) Add the following comment at this point in your script. %% (b) Overhead View. Matlab has specialized graphics functions – cylinder, sphere, and ellipsoid – for generating cylindrical, spherical, and ellipsoidal surfaces. While we can generate all these surfaces using the mathematical equations defining these shapes, these functions make our life a little easier.

Define a new `figure` 2. Using the cylinder function we will now wrap our sine wave function (see figure 1) around the z-axis.

```
[x,y,z] = cylinder(r,50); % wrap sinewave around z-axis
```

To get a preview of your 3D object, temporarily omit the output arguments.

```
cylinder(r,50);
```



Omitting the output arguments `[x,y,z]` causes the cylinder to be displayed with a SURF function. In order to work with our 3D object we will need to assign an output to the cylinder function and then use this output to generate a 3-D colored surface plot.

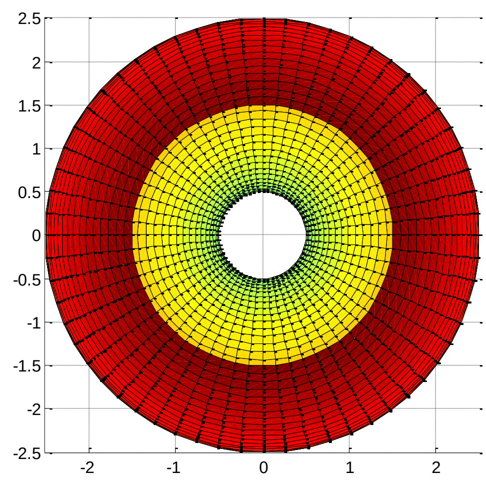
```
[x,y,z] = cylinder(r,60);
c = surf(x,y,z);
```

Within the world of object oriented programming the output argument **c** is known as an "instance" of the object. You can think of it as giving us a way of accessing the properties of the 3-D surface plot.

Now let us take a closer look at the input arguments (r, 50). We know vector **r** is the sinewave function which we want to use to define the surface of our 3D cylindrical object; the number 50 defines the number of points to be computed around the circumference of each circle. In other words, the resolution of our figure in the x and y axes. To see this point more clearly we are going to change our perspective (viewpoint).

Set the `axis` as `square` and the range from -2.5 to 2.5 along the x-axis and from -2.5 to 2.5 along the y-axis. Change the view to directly overhead.

```
view(0,90);
```



From this perspective you can clearly see the 60 points along the circumference defining each circle.

Turn this version of figure 2 in with your lab.

(c) Add the following comment at this point in your script. %% (c) Three Questions and 3D Surface Plot. Let us take a moment to review. (c.1) What is the resolution of your drawing along the z-axis? This is a unitless number relative to a figure of unit height (i.e., 0 to 1). (c.2) What is the resolution in degrees of the 3D cylindrical object about the z-axis (x-y plane)?

In the last section we looked at how different viewpoints can give us insight into our 3D figures. Define a new figure (3). Regenerate our 3D surface plot with axis square. Enter the view command with output argument az (azimuth) and el elevation.

```
[az,el] = view       % returns the current azimuth and elevation.
```

Run your M-file. (c.3) What is the default argument az (azimuth) and el (elevation)? It should **not** be 0 and 90. Do not forget to highlight your answers in your lab report

Change the position you view your 3D object from to something more interesting. Next, change the color of your surface drawing. Turn in this as your figure (3). Include in the **title** of this figure the azimuth and elevation from which the figure was drawn.

(d) Add the following comment at this point in your script. %% (d) Spruce Up The Cylinder. Define a new figure 4. Copy your code to recreate the 3D drawing from part d. Set the view to azimuth -38.5 and elevation 26. It is finally time to spruce up your 3D object. We will do this by changing the properties of the object **c**. For example to set the 'facealpha' property of the object c, you would enter:

```
set(c, 'facealpha',0.8);        % set the face alpha (opacity) to 80%
```

Next, define the RGB values of a dark brown color.

```
dark_brown = [.32 .19 .19];     % define dark_brown
```

- Set the edgecolor property of our object to dark brown.
- Set the edgealpha property to 0.1. This will make the edges almost transparent.

Now let us define the light source and how the object will reflect the light.

```
light('position',[2 -2 .1],'Style','infinite');% create light source
lighting phong;      % define how the light is reflected
```

Finally, let us view our creation without any axes and the frame

```
set(gca,'visible','off');       % remove the axes and the frame
```

Turn in this as your figure (4).

(e) Add the following comment at this point in your script. %% (e) Vase. Define a new figure 5. In this part of the lab we will turn our 3D object into a vase. To accomplish this goal all we need to do is cut-off the top of our 3D object.

```
z1 = z;
z1(60:101,:)=NaN;  % z1(51:101,:)=NaN;
```

- Create a new 3D surface drawing using x, y, and z1. Assign the instance of the 3D object to variable **c2**. Again `square` the axis.
- Copy the code for light, facecolor, etc. here from the previous drawing.
- Change the lighting from `phone` to `flat`.
- Change the view to azimuth -40 elevation 20.

Now all you need is a 3D printer.

Publish the M-File and resulting output to a Word or HTML document as previously described (see end of question 1). Highlight answers to part c.

5. Hopefully, the last section has given you some ideas on how you could make your own unique 3D object. That is exactly what I want you to do for this question. Please be original. Do not copy another student's work or simply grab something off the internet.

   Begin by making a new M-File named My_3D_Art. Add the following comment at the beginning of your script. %% Lab 6 Question 5 My 3D Art. When you have finished your creation publish your M-File and resulting output to a Word or HTML document as previously described (see end of question 1).

Please obtain my signature for questions 3 and 5. Be ready to explain your thought process and ideas for coming up with your cool 3D graphs.