Surfaces
in
Matlab

Math 50C — Multivariable Calculus

David Arnold
David-Arnold@Eureka.redwoods.cc.ca.us

Abstract

In this exercise you will learn how to surfaces in Matlab. The exercises involve functions mapping $\mathbf{R}^2$ into $\mathbf{R}$, such as the function defined by $f(x, y) = x^2 + y^2$. *Prerequisites: Some knowledge of elementary plotting in Matlab. Also, knowledge of Matlab's array operations is useful.*

1 Introduction

This is an interactive document designed for online viewing. We've constructed this onscreen documents because we want to make a conscientious effort to cut down on the amount of paper wasted at the College. Consequently, printing of the onscreen document has been purposefully disabled. However, if you are extremely uncomfortable viewing documents onscreen, we have provided a print version. If you click on the Print Doc button, you will be transferred to the print version of the document, which you can print from your browser or the Acrobat Reader. We respectfully request that you only use this feature when you are at home. Help us to cut down on paper use at the College.

Much effort has been put into the design of the onscreen version so that you can comfortably navigate through the document. Most of the navigation tools are evident, but one particular feature warrants a bit of explanation. The section and subsection headings in the onscreen and print documents are interactive. If you click on any section or subsection header in the onscreen document, you will be transferred to an identical location in the print version of the document. If you are in the print version, you can make a return journey to the onscreen document by clicking on any section or subsection header in the print document.

Finally, the table of contents is also interactive. Clicking on an entry in the table of contents takes you directly to that section or subsection in the document.

1.1 Working with Matlab

This document is a working document. It is expected that you are sitting in front of a computer terminal where the Matlab software is installed. You are not supposed to read this document as if it were a short story. Rather, each time your are presented with a Matlab command, it is expected that you will enter the command, then hit the Enter key to execute the command and view the result. Furthermore, it is expected that you will ponder the result. Make sure that you completely understand why you got the result you did before you continue with the reading.

2 Surfaces in Matlab

In single variable calculus, we studied functions that mapped the real numbers into the real numbers. In symbols, $f : \mathbf{R} \to \mathbf{R}$. In the plane ($\mathbf{R}^2$), the graph of this function f is defined as follows.

Definition 1

Suppose that $f : \mathbf{R} \to \mathbf{R}$. Then the graph of f is

$$\{(x, f(x)) : \ x \in \text{Domain of } f\}$$

That is, the graph of f is the set of all ordered pairs that satisfy the equation of f. For example, suppose that $f : \mathbf{R} \to \mathbf{R}$ is defined by the equation $f(x) = x^2$. An easy calculation shows that $f(2) = 4$. Therefore, the ordered pair $(2, 4)$ lies on the graph of f.

Now, suppose that we have a function that maps elements in the plane onto the real line. That is, $f : \mathbf{R}^2 \to \mathbf{R}$. Then the graph of f lies in 3-space ($\mathbf{R}^3$) and is defined as follows.

Definition 2

Suppose that $f : \mathbf{R}^2 \to \mathbf{R}$. Then the graph of f is

$$\{(x, y, f(x, y)) : \ (x, y) \in \text{Domain of } f\}$$

Thus, the graph of f is the set of all ordered triples that satisfy the equation of f. For example, suppose that $f : \mathbf{R}^2 \to \mathbf{R}$ is defined by the equation $f(x, y) = x^2 + y^2$. Then an easy calculation shows that $f(2, 1) = 5$. Therefore, $(2, 1, 5)$ is an ordered triple that lies on the graph of f. This point is shown in **Figure 1**.

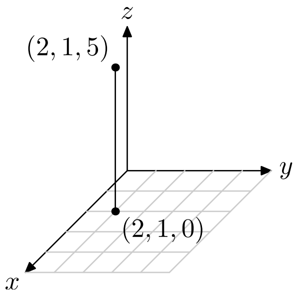

Figure 1 Point $(2, 1, 5)$ is on the graph of f.

To plot the graph of $f(x) = x^2$ in the plane, we begin by making a table of points that satisfy the equation, as shown in **Table 1**

x	-3	-2	-1	0	1	2	3
$f(x) = x^2$	9	4	1	0	1	4	9

Table 1 A table of points satisfying $f(x) = x^2$.

However, if $f(x, y) = x^2 + y^2$, the domain of f lies in $\mathbf{R}^2$. Thus, for each ordered pair (x, y), our function computes an output $z = f(x, y)$. It is the ordered triple (x, y, z) that must be plotted. We must come up with a new strategy for creating a table of points that satisfy the equation $f(x, y) = x^2 + y^2$. Matlab accomplishes this with the `meshgrid` command.

```
>> [X,Y]=meshgrid([1,2,3,4,5])
X =
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
Y =
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4
     5     5     5     5     5
```

This rather cryptic output warrants extensive explanation. Actually, the output is easily understood if one superimposes the matrix Y onto the matrix X to obtain a grid of ordered pairs.

$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$
$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$
$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$
$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$
$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$

Table 2 Interpreting the output of `meshgrid`.

Therefore, **Table 2** contains a set of points in the plane that we will substitute into the function $f(x, y) = x^2 + y^2$. Matlab's array smart operators make this an easy proposition.

```
>> Z=X.^2+Y.^2
Z =
     2     5    10    17    26
     5     8    13    20    29
    10    13    18    25    34
    17    20    25    32    41
    26    29    34    41    50
```

The alert reader will want use their calculator (mental calculations are also good) to check that these points actually satisfy the equation $f(x, y) = x^2 + y^2$. It is now an easy task to plot the surface to which these points belong. The following command was used to produce the image in **Figure 2**.

```
>> mesh(X,Y,Z)
```

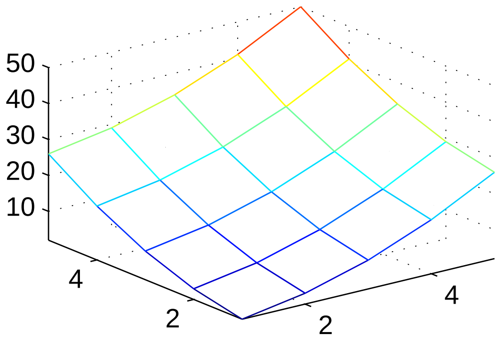

Figure 2 Plotting the surface $f(x,y) = x^2 + y^2$.

The help file for the meshgrid command is informative.

```
>> help meshgrid

 MESHGRID   X and Y arrays for 3-D plots.
   [X,Y] = MESHGRID(x,y) transforms the domain specified by vectors
   x and y into arrays X and Y that can be used for the evaluation
   of functions of two variables and 3-D surface plots.
   The rows of the output array X are copies of the vector x and
   the columns of the output array Y are copies of the vector y.

   [X,Y] = MESHGRID(x) is an abbreviation for [X,Y] = MESHGRID(x,x).
```

In our previous example, we used the form `[X,Y]=meshgrid([1,2,3,4,5])`. As stated in the helpfile above, this is an abbreviation for the command `[X,Y]=meshgrid([1,2,3,4,5],[1,2,3,4,5])`. We'll extend the domain and enter values for `x` and `y` separately in the following example.

```
>> x=-3:3
x =
   -3    -2    -1     0     1     2     3
>> y=-3:3
y =
    -3    -2    -1     0     1     2     3
```

We can now create a grid of domain points with the following command.

```
>> [X,Y]=meshgrid(x,y)
X =
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
```

```
Y =
    -3    -3    -3    -3    -3    -3    -3
    -2    -2    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1    -1    -1
     0     0     0     0     0     0     0
     1     1     1     1     1     1     1
     2     2     2     2     2     2     2
     3     3     3     3     3     3     3
```

Note that each row of matrix X contains the vector x and each column of matrix Y contains the vector y. In this manner, we create a grid of points to enter into the function $z = x^2 + y^2$. This is easily done in Matlab.

```
>> Z=X.^2+Y.^2
Z =
    18    13    10     9    10    13    18
    13     8     5     4     5     8    13
    10     5     2     1     2     5    10
     9     4     1     0     1     4     9
    10     5     2     1     2     5    10
    13     8     5     4     5     8    13
    18    13    10     9    10    13    18
```

The point represented by the upper left corner of matrices X and Y is the point $(-3, -3)$. Substituting in $z = x^2+y^2$, we have $z = (-3)^2 + (-3)^2 = 18$, which agrees with the value in the upper left corner of matrix Z. The careful reader will take time to check the remaining values of matrix Z.

It is now an easy task to sketch the surface to which these points belong. The following command was used to produce the image in **Figure 3**.

```
>> mesh(X,Y,Z)
```

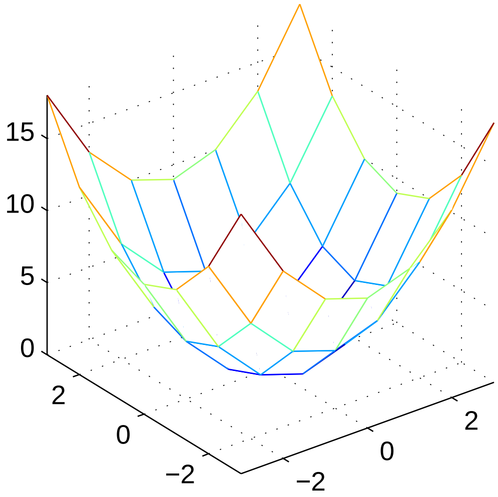

Figure 3 The surface $z = x^2 + y^2$ on the domain $\{(x, y) : -3 \le x \le 3,\ -3 \le y \le 3\}$.

2.1 Refining the Plot

Recall how Matlab's `plot` command was used to draw the graphs of functions.

```
>> x=linspace(0,2*pi);
>> y=sin(2*x);
>> plot(x,y)
```

This set of commands would plot the points in the vectors x and y, then line segments were used to connect consecutive points. If you plotted enough points, then the plot took on the shape of a smooth curve. Too few points and your plot had a "jagged" look.

When Matlab plots a surface, a similar thing occurs. Matlab plots the points, then neighboring points are connected with segments. The surface takes on the appearance of a mesh, where each set of four neighboring points seemed to be joined with small quadrilaterals. Again, if you plot too few points, the surface takes on a "jagged" look and feel. To draw a smoother surface, plot more points. The following commands were used to draw the image in **Figure 4**.

```
>> x=-3:.2:3;
>> y=-3:.2:3;
>> [X,Y]=meshgrid(x,y);
>> Z=X.^2+Y.^2;
>> mesh(X,Y,Z)
```

Matlab has a number of functions that are useful for annotating your plots. For example, we added axis labels and a title to the plot in **Figure 4** with these commands.

```
>> xlabel('x-axis')
>> ylabel('y-axis')
>> zlabel('z-axis')
>> title('The plot of z = x^2 + y^2.')
```

3 Script Files

Working at the command line in Matlab has its advantages and disadvantages. It's convenient to enter a command and see the immediate effect. However, it's annoying to type in a list of sequential commands, only to find you have made a mistake. Retyping a sequence of commands to correct a mistake, even with Matlab's up-arrow facility for replaying commands[1], is tedious and inefficient.

The solution lies in a type of file called a *script file*. In Matlab, a script file is just a sequence of commands, similar to what you enter at the command prompt, but saved in a file for future use. With Matlab's built-in editor, it's a simple matter to create a script file. Start by entering the following command at the Matlab prompt to open the editor.

```
>> edit
```

When the editor opens, enter the following list of commands.

---

[1] Each time you press the up-arrow on your keyboard, Matlab replays the previous command in its command history.

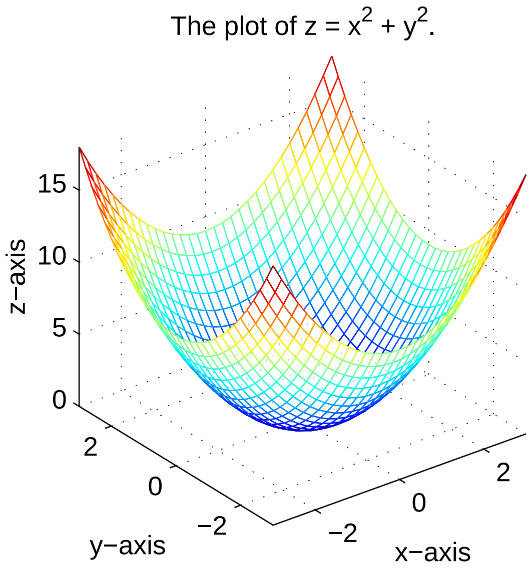

Figure 4 A smoother plot of the surface $f(x, y) = x^2 + y^2$.

```
close all
x=-3:.2:3;
y=-3:.2:3;
[X,Y]=meshgrid(x,y);
Z=X.^2+Y.^2;
mesh(X,Y,Z)
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('The plot of z = x^2 + y^2.')
axis tight
```

Next, you must save the file in a folder on your Matlab path. It is essential that you develop a good strategy for saving your work. Let me propose a sample strategy. Later, you can rework this strategy should you find something more suitable for your particular needs.

Open the Windows Explorer by holding down the control key and pressing the letter E (just once) on the keyboard. Again, the key combination is Ctrl+E.

- If you are at school, select your personal H drive and create a new folder called Math50C.
- If you are at home, select the C drive and create a new folder called Math50C.

Once you've created the Math50C folder, double-click this folder to open it. We've just created the Math50C folder, so there will be nothing in it. Now create a new folder inside the Math50C folder called MatlabPrograms. Note the case of the letters and the fact that there are no spaces in the folder name[2]. Now, in the future, save all of your Matlab programs for multivariable calculus in this folder.

---

[2] It's a very bad practice to have spaces in filenames. MacIntosh users are particularly susceptible to this disease.

You can now save your script file in this folder. Select Save As from the File menu of the editor, then browse to the folder /Math50c/MatlabPrograms and save the file as **newfigure.m**. That's all there is to it!

To run your new script file, return to the Matlab command window. You must first change the current working directory to point to /Math50C/MatlabPrograms folder. This can be accomplished in a number of ways.

- If you are using Matlab 6, you can select *Current Directory* from the View menu. In the navigation window, you can enter the path to the directory you wish to make the current directory. For example, at school you would enter H:/Math50C/MatlabPrograms. However, if you are at home, you might enter C:/Math50C/MatlabPrograms. There is also a Browse button available that you can use in the usual manner dictated by your operating system. You might also note that there is a history of recently visited folders. This can also be used to quickly revisit a folder.
- I find it easiest to just type the command cd H:/Math50C/MatlabPrograms at the Matlab prompt. The cd command stands for "change directory."

Return to the Matlab prompt and enter the command `pwd`. This command is an old Unix command that responds with the "present working directory." If you have followed the directions as above, Matlab should respond with something like the following.

```
>> pwd
ans =
h:\Math50C\MatlabPrograms
>>
```

You can get a listing of the contents in the current directory with either of the commands **dir** or **ls**. If you see the file **newfigure.m** listed among the contents, then you are ready to execute the script file. The script file is executed by typing its name at the Matlab prompt.

```
>> newfigure
```

Hit the Enter key to execute the command. Each line in the script file will be executed in turn, just as if you were entering them individually at the Matlab prompt. The result of executing the script **newfigure** is shown in Figure 5.

The beauty of this method is obvious. Should you make a mistake or decide to make additions to the code, revisit the script file in the editor, make your changes, then execute the script at the Matlab prompt. Indeed, this is the most efficient way to work in Matlab.

4 Examples

The best way to learn is by performing a number of examples. Let's begin. In the first example, z is a function of x and y, but only x is explicitly mentioned in the formula for z.

Example 1

Sketch the graph of $z = x^2$ over the region in the xy-plane defined by $D = \{(x, y) : -1 \le x \le 1,\ -1 \le y \le 1\}$.

The following commands produce the image in Figure 6.

```
close all
x=-1:.1:1;
```

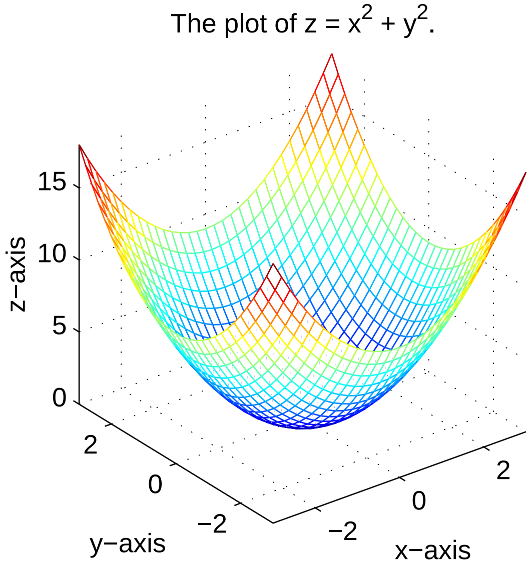

Figure 5 Producing an image with the script file **newfigure.m**.

```
y=-1:.1:1;
[xx,yy]=meshgrid(x,y);
zz=xx.ˆ2;
mesh(xx,yy,zz);
title('The graph of z=xˆ2')
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
axis tight
```

With a little adjustment, it's easy to draw the graph of $y = f(x,z)$ or $x = f(y,z)$.

Example 2

Sketch the graph of

$$y = x^2 - z^2$$

over the region in the xz-plane defined by $D = \{(x,z) : -1 \le x \le 1,\ -1 \le z \le 1\}$.

The following commands produce the image in **Figure 7**.

```
close all
x=-1:.1:1;
z=-1:.1:1;
[xx,zz]=meshgrid(x,z);
yy=xx.ˆ2-zz.ˆ2;
mesh(xx,yy,zz);
xlabel('x-axis')
ylabel('y-axis')
```

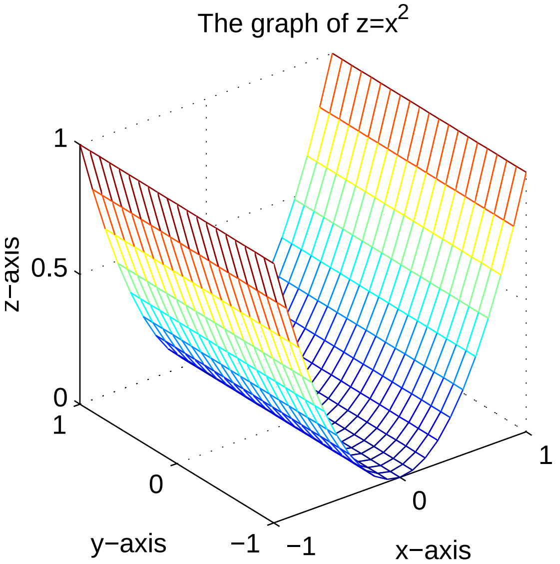

Figure 6 The graph of $z = x^2$ on the domain $\{(x,y);\,-1 \le x \le 1,\,-1 \le y \le 1\}$.

```
zlabel('z-axis')
title('The graph of y=x^2-z^2')
axis tight
```

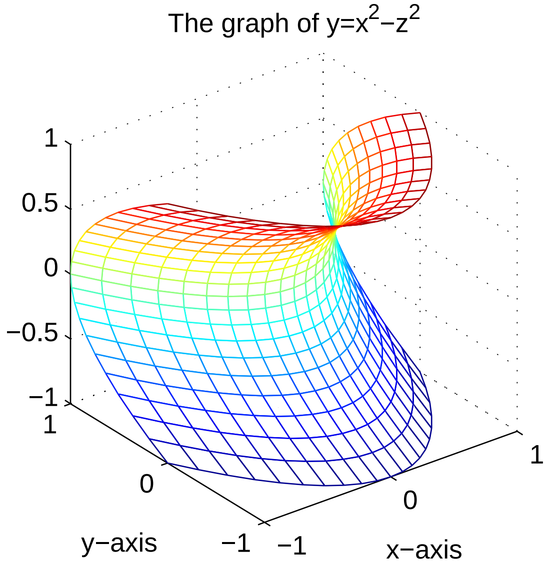

Figure 7 The graph of $y = x^2 - z^2$ on the domain $\{(x,z) : -1 \le x \le 1,\,-1 \le z \le 1\}$.

In the next example, x is a function of y and z, but only z is explicitly mentioned.

Example 3

Sketch the graph of $x = z^3$ over the region in the yz-plane defined by $\{(y,z) : -1 \le y \le 1,\,-1 \le z \le 1\}$.

The following commands produce the image in **Figure 8**.

```
close all
y=-1:.1:1;
z=-1:.1:1;
[yy,zz]=meshgrid(y,z);
xx=zz.^3;
mesh(xx,yy,zz);
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
title('The graph of x=z^3')
axis tight
```

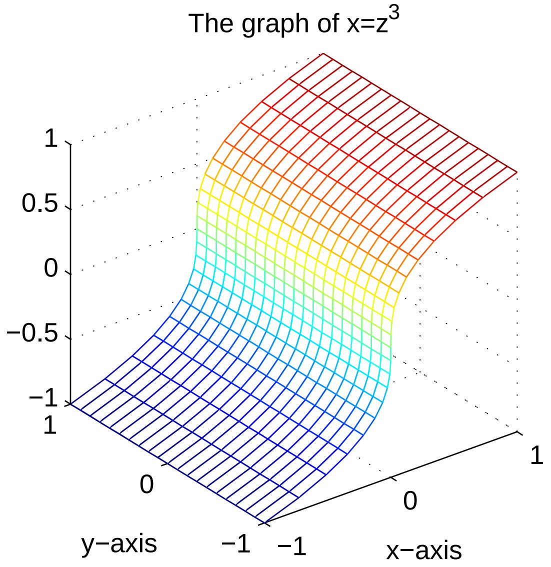

Figure 8 The graph of $z = x^3$ on the domain $\{(x, y) : -1 \le x \le 1, -1 \le y \le 1\}$.

5 A Useful Script

It's hard to differentiate the x-axis from the y-axis in a three dimensional plot if the axes are not labelled. Consequently, you'll find yourself repeatedly typing the following commands.

```
>> xlabel('x-axis')
>> ylabel('y-axis')
>> zlabel('z-axis')
```

After a while, this is annoying, particularly when you can automate this task. Open Matlab's editor and enter the following commands in a new script file.

```
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
```

Furthermore, you may have noted that the axes in the figures we've drawn thus far do not match the usual orientation we use in class. Take the last figure you drew and click the **rotate** icon in the figure window, then use the mouse to rotate the figure. As you rotate your figure, note that the azimuth and elevation are reported in the lower left corner of the figure window. Rotate the figure until the azimuth equals 130 degrees and the elevation equals 40 degrees. Or, equivalently, enter the following command at the Matlab prompt.

```
>> view([130,40])
```

This will rotate the image into the view shown in **Figure 9**. Note that the positive direction on both the axes matches the same common orientation we use on the whiteboard during class.

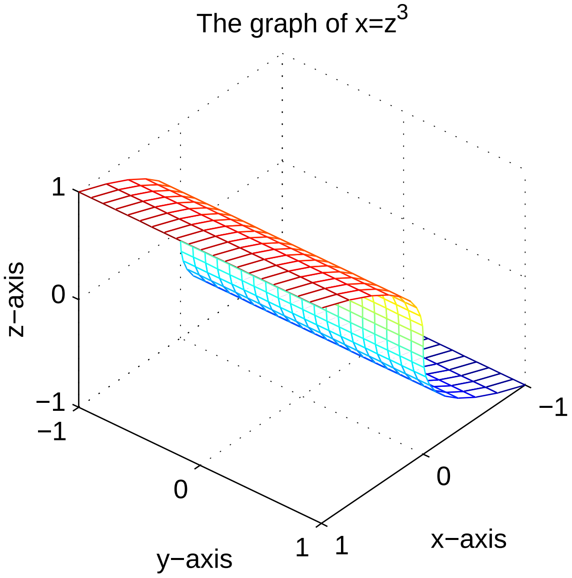

Figure 9 Rotate with `view([130,40])`.

Because this is the orientation we use, put this line into your script file as well.

```
xlabel('x-axis')
ylabel('y-axis')
zlabel('z-axis')
view([130,40])
```

Now, save the file as **xyz.m**. Put the file in a directory on Matlab's path. A good place to save this file is in your /Math50C/MatlabPrograms folder. Now, any time you type **xyz** at the Matlab prompt or enter **xyz** in a script, the script will execute, labelling the axes and changing the orientation.

6 Exercises

Sketch the surface represented by each of the following equations. Obtain hardcopy printouts to hand in during class.

1. Sketch the graph of

$$z = x^2 - 4y^2$$

over the region in the xy-plane defined by $D = \{(x, y) : -2 \le x \le 2,\ -2 \le y \le 2\}$.

2. Sketch the graph of

$$y = x - z^2$$

over the region in the xz-plane defined by $D = \{(x, z) : -2 \le x \le 2,\ -2 \le z \le 2\}$.

3. Sketch the graph of

$$x = \cos(y)$$

over the region in the yz-plane defined by $D = \{(y, z) : -\pi \le y \le \pi,\ -\pi \le z \le \pi\}$.

4. Sketch the graph of

$$z = \sin(x^2 + y^2)$$

over the region in the xy-plane defined by $D = \{(x, y) : -1.6 \le x \le 1.6,\ -1.6 \le y \le 1.6\}$