Chapter 10
Three-Dimensional Plots

Three-dimensional (3-D) plots can be a useful way to present data that consists of more than two variables. MATLAB provides various options for displaying three-dimensional data. They include line and wire, surface, mesh plots, and many others. The plots can also be formatted to have a specific appearance and special effects. Many of the three-dimensional plotting features are described in this chapter. Additional information can be found in the Help Window under **Plotting and Data Visualization**.

In many ways this chapter is a continuation of Chapter 5, where two-dimensional plots were introduced. The 3-D plots are presented in a separate chapter because not all MATLAB users use them. In addition, new users of MATLAB will probably find it easier to practice 2-D plotting first and learn the material in Chapters 6–9 before attempting 3-D plotting. It is assumed throughout the rest of this chapter that the reader is familiar with 2-D plotting.

10.1 LINE PLOTS

A three-dimensional line plot is a line that is obtained by connecting points in three-dimensional space. A basic 3-D plot ‘is created with the `plot3` command, which is very similar to the `plot` command and has the form:

```
plot3(x,y,z,'line specifiers','PropertyName',property value)
```

`x`, `y`, and `z` are vectors of the coordinates of the points.

(Optional) Specifiers that define the type and color of the line and markers.

(Optional) Properties with values that can be used to specify the line width, and marker's size and edge and fill colors.

- The three vectors with the coordinates of the data points must have the same number of elements.

- The line specifiers, properties, and property values are the same as in 2-D plots (see Section 5.1).

For example, if the coordinates x, y, and z are given as a function of the parameter t by

$$x = \sqrt{t}\sin(2t)$$
$$y = \sqrt{t}\cos(2t)$$
$$z = 0.5t$$

a plot of the points for $0 \le t \le 6\pi$ can be produced by the following script file:

```
t=0:0.1:6*pi;
x=sqrt(t).*sin(2*t);
y=sqrt(t).*cos(2*t);
z=0.5*t;
plot3(x,y,z,'k','linewidth',1)
grid on
xlabel('x'); ylabel('y'); zlabel('z')
```

The plot shown in Figure 10-1 is created when the script is executed.

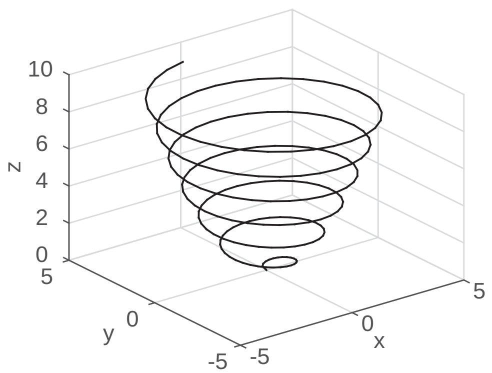


Figure 10-1: A plot of the function $x = \sqrt{t}\sin(2t)$, $y = \sqrt{t}\cos(2t)$, $z = 0.5t$ **for** $0 \le t \le 6\pi$.

10.2 Mesh and Surface Plots

Mesh and surface plots are three-dimensional plots used for plotting functions of the form $z = f(x, y)$ where x and y are the independent variables and z is the dependent variable. It means that within a given domain the value of z can be calculated for any combination of x and y. Mesh and surface plots are created in three steps. The first step is to create a grid in the x y plane that covers the domain of the function. The second step is to calculate the value of z at each point of the grid. The third step is to create the plot. The three steps are explained next.

Creating a grid in the *x y* plane (Cartesian coordinates):

The grid is a set of points in the x y plane in the domain of the function. The density of the grid (number of points used to define the domain) is defined by the user. Figure 10-2 shows a grid in the domain $-1 \leq x \leq 3$ and $1 \leq y \leq 4$. In this

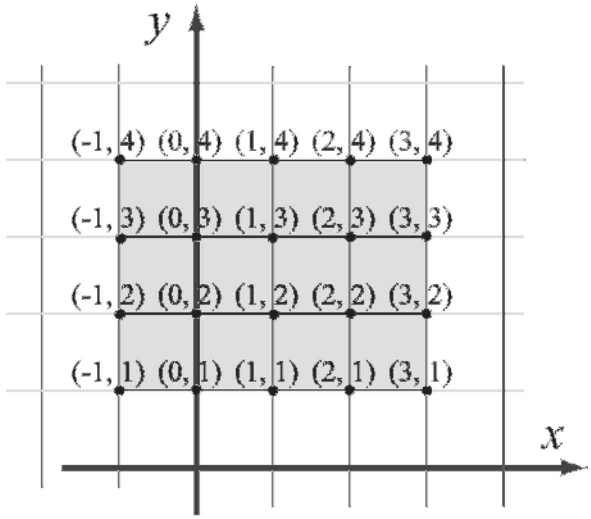


Figure 10-2: A grid in the *x y* plane for the domain $-1 \leq x \leq 3$ and $1 \leq y \leq 4$ with spacing of 1.

grid the distance between the points is one unit. The points of the grid can be defined by two matrices, X and Y. Matrix X has the x coordinates of all the points, and matrix Y has the y coordinates of all the points:

$$X = \begin{bmatrix} -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 4 \end{bmatrix} \quad \text{and} \quad Y = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 \\ 3 & 3 & 3 & 3 & 3 \\ 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The X matrix is made of identical rows since in each row of the grid the points have the same x coordinate. In the same way the Y matrix is made of identical columns since in each column of the grid the y coordinate of the points is the same.

MATLAB has a built-in function, called `meshgrid`, that can be used for creating the X and Y matrices. The form of the `meshgrid` function is:

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

`X` is the matrix of the x coordinates of the grid points.
`Y` is the matrix of the y coordinates of the grid points.

`x` is a vector that divides the domain of x.
`y` is a vector that divides the domain of y.

In the vectors `x` and `y` the first and last elements are the respective boundaries of the domain. The density of the grid is determined by the number of elements in the vectors. For example, the mesh matrices X and Y that correspond to the grid in Figure 10-2 can be created with the `meshgrid` command by:

```
>> x=-1:3;
>> y=1:4;
>> [X,Y]=meshgrid(x,y)
X =
    -1     0     1     2     3
    -1     0     1     2     3
    -1     0     1     2     3
    -1     0     1     2     3
Y =
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4
```

Once the grid matrices exist, they can be used for calculating the value of z at each grid point.

Calculating the value of z at each point of the grid:

The value of z at each point is calculated by using element-by-element calculations in the same way it is used with vectors. When the independent variables x and y are matrices (they must be of the same size), the calculated dependent variable is also a matrix of the same size. The value of z at each address is calculated from the corresponding values of x and y. For example, if z is given by

$$z = \frac{xy^2}{x^2 + y^2}$$

the value of z at each point of the grid above is calculated by:

```
>> Z = X.*Y.^2./(X.^2 + Y.^2)
```

```
Z =
   -0.5000         0    0.5000    0.4000    0.3000
   -0.8000         0    0.8000    1.0000    0.9231
   -0.9000         0    0.9000    1.3846    1.5000
   -0.9412         0    0.9412    1.6000    1.9200
```

Once the three matrices have been created, they can be used to plot mesh or surface plots.

Making mesh and surface plots:

A mesh or surface plot is created with the `mesh` or `surf` command, which has the form:

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

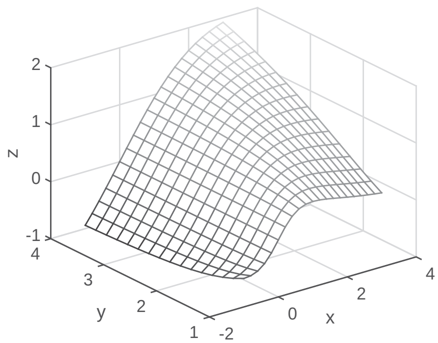
where `X` and `Y` are matrices with the coordinates of the grid and `Z` is a matrix with the value of z at the grid points. The mesh plot is made of lines that connect the points. In the surface plot, areas within the mesh lines are colored.

As an example, the following script file contains a complete program that creates the grid and then makes a mesh (or surface) plot of the function $z = \frac{xy^2}{x^2 + y^2}$ over the domain $-1 \le x \le 3$ and $1 \le y \le 4$.

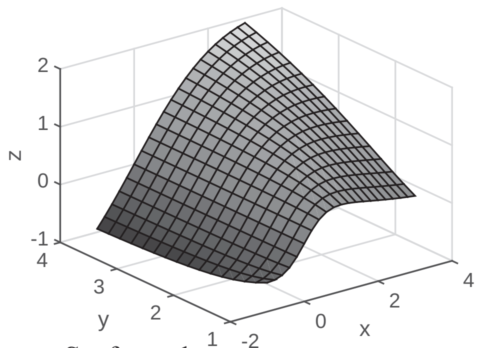
```
x=-1:0.1:3;
y=1:0.1:4;
[X,Y]=meshgrid(x,y);
Z=X.*Y.^2./(X.^2+Y.^2);
mesh(X,Y,Z)
xlabel('x'); ylabel('y'); zlabel('z')
```

Type `surf(X,Y,Z)` for surface plot.

Note that in the program above the vectors `x` and `y` have a much smaller spacing than the spacing earlier in the section. The smaller spacing creates a denser grid. The figures created by the program are:



Mesh plot



Surface plot

Additional comments on the mesh command:

- The plots that are created have colors that vary according to the magnitude of z. The variation in color adds to the three-dimensional visualization of the plots. The color can be changed to be a constant either by using the Plot Editor in the Figure Window (select the edit arrow, click on the figure to open the Property Editor Window, then change the color in the Mesh Properties list), or by using the `colormap(C)` command. In this command `C` is a three-element vector in which the first, second, and third elements specify the intensity of Red, Green, and Blue (RGB) colors, respectively. Each element can be a number between 0 (minimum intensity) and 1 (maximum intensity). Some typical colors are:

  C = [0 0 0] black   C = [1 0 0] red   C = [0 1 0] green
  C = [0 0 1] blue   C = [1 1 0] yellow   C = [1 0 1] magenta
  C = [0.5 0.5 0.5] gray

- When the mesh command executes, the grid is on by default. The grid can be turned off with the `grid off` command.
- A box can be drawn around the plot with the `box on` command.
- The `mesh` and `surf` commands can also be used with the form `mesh(Z)` and `surf(Z)`. In this case the values of `Z` are plotted as a function of their addresses in the matrix. The row number is on the x axis and the column number is on the y axis.

There are several additional plotting commands that are similar to the `mesh` and `surf` commands that create plots with different features. Table 10-1 shows a summary of the mesh and surface plotting commands. All the examples in the table are plots of the function $z = 1.8^{-1.5\sqrt{x^2+y^2}}\sin(x)\cos(0.5y)$ over the domain $-3 \le x \le 3$ and $-3 \le y \le 3$.

Table 10-1: Mesh and surface plots

<table>
<tr><th>Plot type</th><th>Example of plot</th><th>Program</th></tr>
<tr><td>Mesh Plot

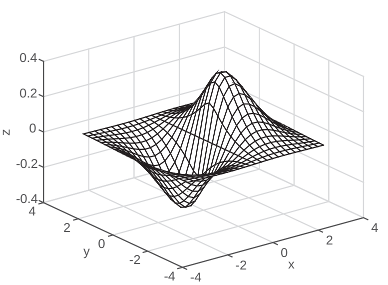
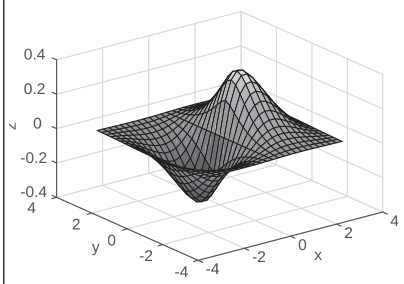
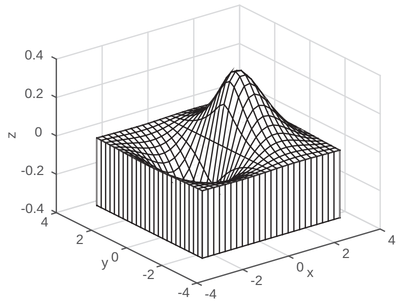
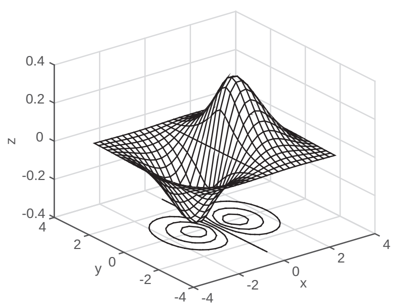
Function format:
<code>mesh(X,Y,Z)</code></td><td>
</td><td><pre>x=-3:0.25:3;
y=-3:0.25:3;
[X,Y] = meshgrid(x,y);
Z=1.8.^(-1.5*sqrt(X.^2+
Y.^2)).*cos(0.5*Y).*sin(X);
mesh(X,Y,Z)
xlabel('x');
ylabel('y')
zlabel('z')</pre></td></tr>
</table>

Table 10-1: Mesh and surface plots (Continued)

Plot type	Example of plot	Program

<u>Surface Plot</u>
 	
Function format:
`surf(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+`
		`Y.^2)).*cos(0.5*Y).*sin(X);`
		`surf(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

<u>Mesh Curtain Plot</u> (draws a curtain around the mesh)
 	
Function format:
`meshz(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+`
		`Y.^2)).*cos(0.5*Y).*sin(X);`
		`meshz(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

<u>Mesh and Contour Plot</u> (draws a contour plot beneath the mesh)
 	
Function format:
`meshc(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+`
		`Y.^2)).*cos(0.5*Y).*sin(X);`
		`meshc(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

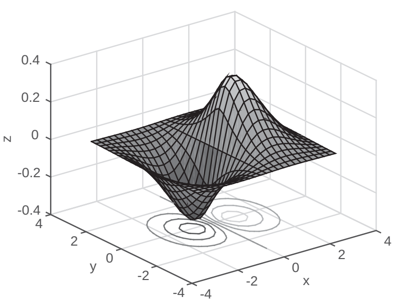
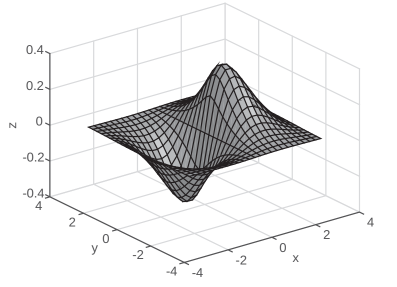
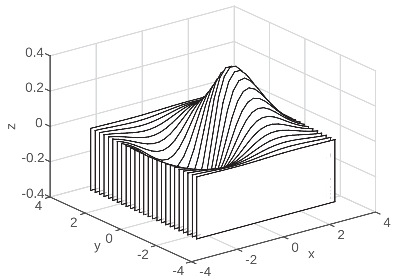
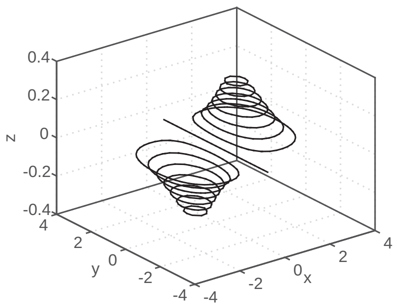
<u>Surface and Contour Plot</u> (draws a contour plot beneath the surface)
 	
Function format:
`surfc(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+`
		`Y.^2)).*cos(0.5*Y).*sin(X);`
		`surfc(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

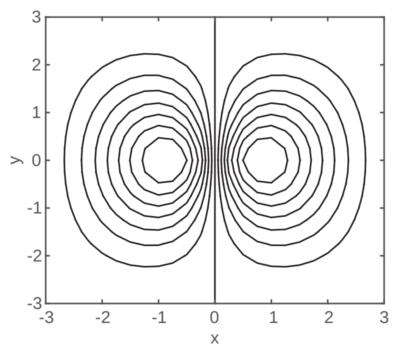
Table 10-1: Mesh and surface plots (Continued)

Plot type	Example of plot	Program

<u>Surface Plot with Lighting</u>
 	
Function format:
`surfl(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+Y.^2)).*cos(0.5*Y).*sin(X);`
		`surfl(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

<u>Waterfall Plot</u> (draws a mesh in one direction only)
 	
Function format:
`waterfall(X,Y,Z)`
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+Y.^2)).*cos(0.5*Y).*sin(X);`
		`waterfall(X,Y,Z)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

<u>3-D Contour Plot</u>
 	
Function format:
`contour3(X,Y,Z,n)`
 	
`n` is the number of contour levels (optional)
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+Y.^2)).*cos(0.5*Y).*sin(X);`
		`contour3(X,Y,Z,15)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

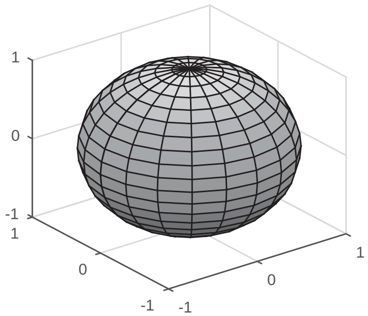
<u>2-D Contour Plot</u> (draws projections of contour levels on the x y plane)
Function format:
`contour(X,Y,Z,n)`
`n` is the number of contour levels (optional)
		`x=-3:0.25:3;`
		`y=-3:0.25:3;`
		`[X,Y ]=meshgrid(x,y);`
		`[X,Y] = meshgrid(x,y);`
		`Z=1.8.^(-1.5*sqrt(X.^2+Y.^2)).*cos(0.5*Y).*sin(X);`
		`contour(X,Y,Z,15)`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

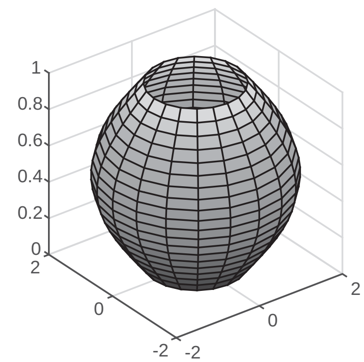
10.3 PLOTS WITH SPECIAL GRAPHICS

MATLAB has additional functions for creating various types of special three-dimensional plots. A complete list can be found in the Help Window under Plotting and Data Visualization. Several of these 3-D plots are presented in Table 10-2. The examples in the table do not show all the options available

Table 10-2: Specialized 3-D plots

Plot type	Example of plot	Program

<u>Plot a Sphere</u>
 	
Function format:
`sphere`
Returns the *x*, *y*, and *z* coordinates of a unit sphere with 20 faces.
`sphere(n)`
Same as above with *n* faces.
		`sphere`
		
		`or:`
		
		`[X,Y,Z]=sphere(20);`
		`surf(X,Y,Z)`

<u>Plot a Cylinder</u>
 	
Function format:
`[X,Y,Z]=`
`cylinder(r)`
Returns the *x*, *y*, and *z* coordinates of cylinder with profile *r*.
		`t=linspace(0,pi,20);`
		`r=1+sin(t);`
		`[X,Y,Z]=cylinder(r);`
		`surf(X,Y,Z)`
		`axis square`

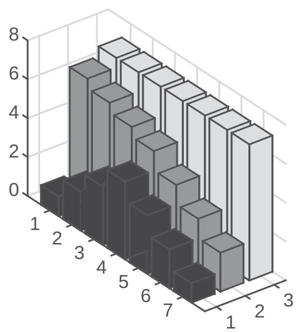
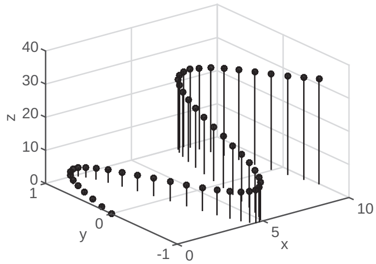
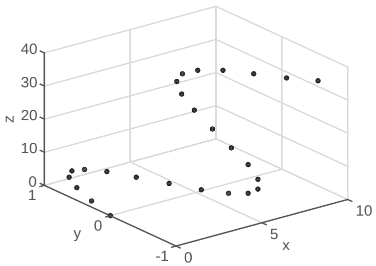
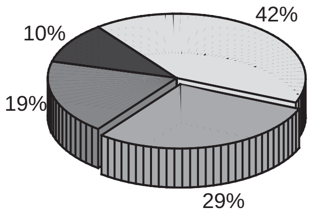
<u>3-D Bar Plot</u>
 	
Function format:
`bar3(Y)`
Each element in `Y` is one bar. Columns are grouped together.
		`Y=[1 6.5 7; 2 6 7;`
		`3 5.5 7; 4 5 7; 3 4`
		`7; 2 3 7; 1 2 7];`
		`bar3(Y)`

Table 10-2: Specialized 3-D plots (Continued)

Plot type	Example of plot	Program

3-D Stem Plot
(draws sequential points with markers and vertical lines from the *x y* plane)
 	
Function format:
`stem3(X,Y,Z)`
		`t=0:0.2:10;`
		`x=t;`
		`y=sin(t);`
		`z=t.^1.5;`
		`stem3(x,y,z,'fill')`
		`grid on`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

3-D Scatter Plot
 	
Function format:
`scatter3(X, Y,Z)`
		`t=0:0.4:10;`
		`x=t;`
		`y=sin(t);`
		`z=t.^1.5;`
		`scatter3(x,y,z,'filled')`
		`grid on`
		`colormap([0.1 0.1 0.1])`
		`xlabel('x');`
		`ylabel('y')`
		`zlabel('z')`

3-D Pie Plot
 	
Function format:
`pie3(X, explode)`
		`X=[5 9 14 20];`
		`explode=[0 0 1 0];`
		`pie3(X,explode)`
		
		`explode` is a vector (same length as `X`) of 0's and 1's. 1 offsets the slice from the center.

with each plot type. More details on each type of plot can be obtained in the Help Window, or by typing `help command_name` in the Command Window.

Polar coordinates grid in the *x y* plane:

A 3-D plot of a function in which the value of z is given in polar coordinates (for example $z = r\theta$) can be done by following these steps:

- Create a grid of values of θ and r with the `meshgrid` function.
- Calculate the value of z at each point of the grid.

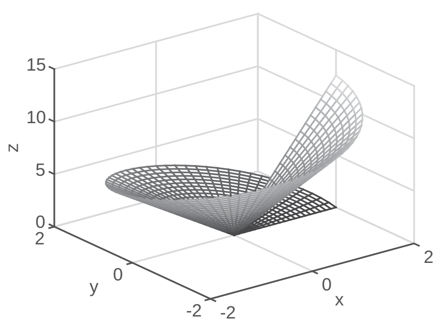
- Convert the polar coordinates grid to a grid in Cartesian coordinates. This can be done with MATLAB's built-in function pol2cart (see example below).
- Make a 3-D plot using the values of z and the Cartesian coordinates.

For example, the following script creates a plot of the function $z = r\theta$ over the domain $0 \le \theta \le 360°$ and $0 \le r \le 2$.

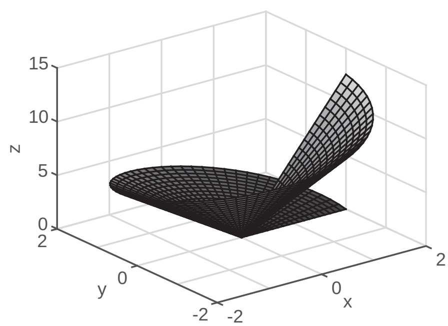
```
[th,r]=meshgrid((0:5:360)*pi/180,0:.1:2);
Z=r.*th;
[X,Y] = pol2cart(th,r);
mesh(X,Y,Z)
```

Type surf(X,Y,Z) for surface plot.

The figures created by the program are:



Mesh plot



Surface plot

10.4 *THE* view *COMMAND*

The view command controls the direction from which the plot is viewed. This is done by specifying a direction in terms of azimuth and elevation angles, as seen in Figure 10-3, or by defining a point in space from which the plot is viewed. To set the viewing angle of the plot, the view command has the form:

```
view(az,el) or view([az,el])
```

- az is the azimuth, which is an angle (in degrees) in the x y plane measured relative to the negative y axis direction and defined as positive in the counterclockwise direction.
- el is the angle of elevation (in degrees) from the x y plane. A positive value corresponds to opening an angle in the direction of the z axis.
- The default view angles are $az = -37.5°$, and $el = 30°$.

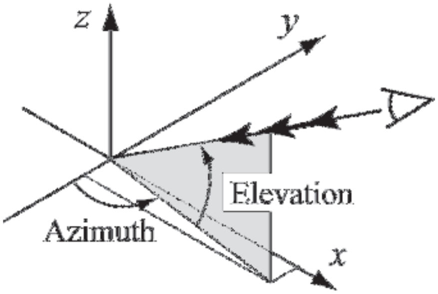
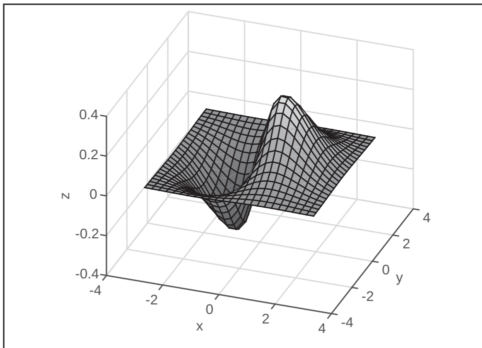


Figure 10-3: Azimuth and elevation angles.

As an example, the surface plot from Table 10-1 is plotted again in Figure 10-4, with viewing angles *az* = 20° and *el* = 35°.



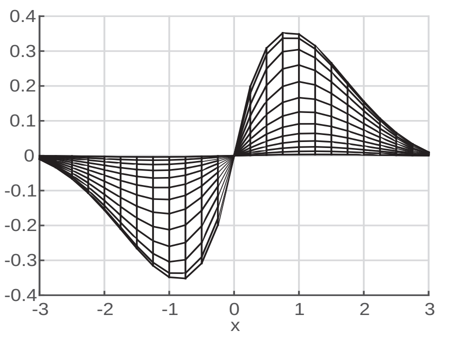

```
x=-3:0.25:3;
y=-3:0.25:3;
[X,Y]=meshgrid(x,y);
Z=1.8.^(-1.5*sqrt(X.^2+
Y.^2)).*cos(0.5*Y).*sin(X);
surf(X,Y,Z)
view(20,35)
```

Figure 10-4: A surface plot of the function $z = 1.8^{-1.5\sqrt{x^2+y^2}}\sin(x)\cos(0.5y)$ **with viewing angles of *az* = 20° and *el* = 35°.**

- With the choice of appropriate azimuth and elevation angles, the `view` command can be used to plot projections of 3-D plots on various planes according to the following table:

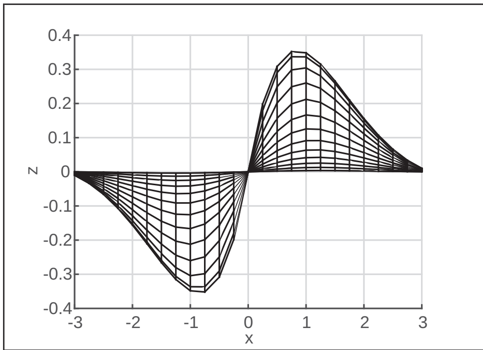
Projection plane	*az* value	*el* value
x y (top view)	0	90
x z (side view)	0	0
y z (side view)	90	0

An example of a top view is shown next. Figure 10-5 shows the top view of the function that is plotted in Figure 10-1. Examples of projections onto the *x z* and *y z* planes are shown next, in Figures 10-6 and 10-7, respectively. The figures show mesh plot projections of the function plotted in Table 10-1.




```
t=0:0.1:6*pi;
x=sqrt(t).*sin(2*t);
y=sqrt(t).*cos(2*t);
z=0.5*t;
plot3(x,y,z,'k','linewidth',1)
view(0,90)
grid on
xlabel('x'); ylabel('y')
zlabel('z')
```

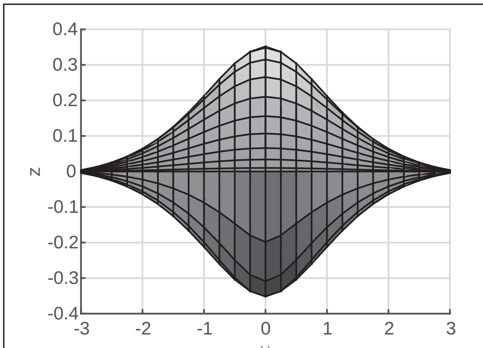
Figure 10-5: A top view plot of the function $x = \sqrt{t}\sin(2t)$, $y = \sqrt{t}\cos(2t)$, $z = 0.5t$ for $0 \le t \le 6\pi$.




```
x=-3:0.25:3;
y=-3:0.25:3;
[X,Y]=meshgrid(x,y);
Z=1.8.^(-1.5*sqrt(X.^2+
Y.^2)).*cos(0.5*Y).*sin(X);
mesh(X,Y,Z)
view(0,0)
```

Figure 10-6: Projections onto the x z plane of the function.

$$z = 1.8^{-1.5\sqrt{x^2+y^2}}\sin(x)\cos(0.5y).$$




```
x=-3:0.25:3;
y=-3:0.25:3;
[X,Y]=meshgrid(x,y);
Z=1.8.^(-1.5*sqrt(X.^2+
Y.^2)).*cos(0.5*Y).*sin(X);
surf(X,Y,Z)
view(90,0)
```

Figure 10-7: Projections onto the y-z plane of the function.

$$z = 1.8^{-1.5\sqrt{x^2+y^2}}\sin(x)\cos(0.5y).$$

- The view command can also set a default view:

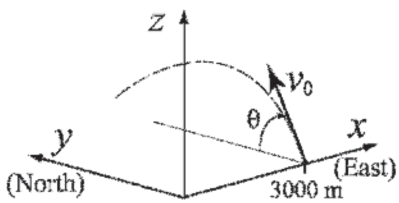
`view(2)` sets the default to the top view, which is a projection onto the x-y plane with $az = 0°$, and $el = 90°$.
`view(3)` sets the default to the standard 3-D view with $az = -37.5°$ and $el = 30°$.

- The viewing direction can also be set by selecting a point in space from which the plot is viewed. In this case the `view` command has the form `view([x,y,z])`, where `x`, `y`, and `z` are the coordinates of the point. The direction is determined by the direction from the specified point to the origin of the coordinate system and is independent of the distance. This means that the view is the same with point [6, 6, 6] as with point [10, 10, 10]. Top view can be set up with [0, 0, 1]. A side view of the x z plane from the negative y direction can be set with [0, –1, 0], and so on.

10.5 *Examples of MATLAB Applications*

Sample Problem 10-1: 3-D projectile trajectory

A projectile is fired with an initial velocity of 250 m/s at an angle of θ = 65° relative to the ground. The projectile is aimed directly north. Because of a strong wind blowing to the west, the projectile also moves in this direction at a constant speed of 30 m/s. Determine and plot the trajectory of the projectile until it hits the ground. For comparison, plot also (in the same figure) the trajectory that the projectile would have had if there was no wind.



Solution

As shown in the figure, the coordinate system is set up such that the x and y axes point in the east and north directions, respectively. Then the motion of the projectile can be analyzed by considering the vertical direction z and the two horizontal components x and y. Since the projectile is fired directly north, the initial velocity v_0 can be resolved into a horizontal y component and a vertical z component:

$$v_{0y} = v_0\cos(\theta) \quad \text{and} \quad v_{0z} = v_0\sin(\theta)$$

In addition, due to the wind the projectile has a constant velocity in the negative x direction, $v_x = -30$ m/s.

The initial position of the projectile (x_0, y_0, z_0) is at point (3000, 0, 0). In the vertical direction the velocity and position of the projectile are given by:

$$v_z = v_{0z} - gt \quad \text{and} \quad z = z_0 + v_{0z}t - \frac{1}{2}gt^2$$

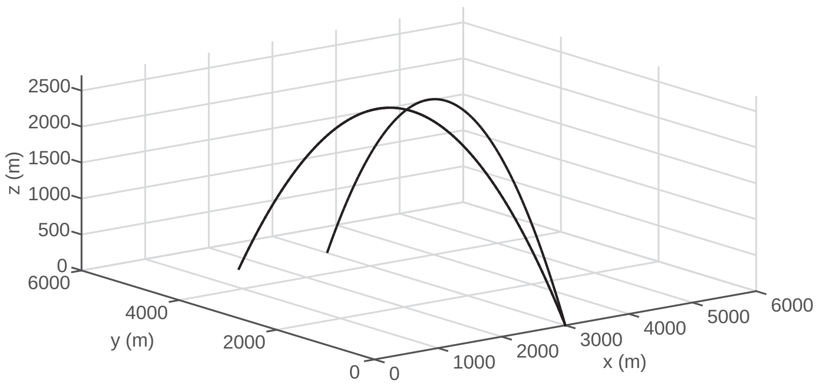
The time it takes the projectile to reach the highest point $(v_z = 0)$ is $t_{hmax} = \frac{v_{0z}}{g}$. The total flying time is twice this time, $t_{tot} = 2t_{hmax}$. In the horizontal direction the velocity is constant (both in the x and y directions), and the position of the projectile is given by:

$$x = x_0 + v_x t \quad \text{and} \quad y = y_0 + v_{0y} t$$

The following MATLAB program written in a script file solves the problem by following the equations above.

```
v0=250; g=9.81; theta=65;
x0=3000; vx=-30;
v0z=v0*sin(theta*pi/180);
v0y=v0*cos(theta*pi/180);
t=2*v0z/g;
tplot=linspace(0,t,100);       Creating a time vector with 100 elements.
z=v0z*tplot-0.5*g*tplot.^2;
y=v0y*tplot;                   Calculating the x, y, and z coordinates of the projectile at each time.
x=x0+vx*tplot;
xnowind(1:length(y))=x0;       Constant x coordinate when no wind.
plot3(x,y,z,'k-',xnowind,y,z,'k--')     Two 3-D line plots.
grid on
axis([0 6000 0 6000 0 2700])
xlabel('x (m)'); ylabel('y (m)'); zlabel('z (m)')
```

The figure generated by the program is shown below.



Sample Problem 10-2: Electric potential of two point charges

The electric potential V around a charged particle is given by

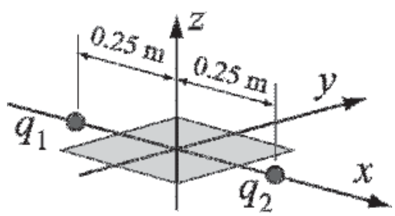
$$V = \frac{1}{4\pi\varepsilon_0}\frac{q}{r}$$

where $\varepsilon_0 = 8.8541878 \times 10^{-12} \frac{\text{C}}{\text{N m}^2}$ is the permittivity constant, q is the magnitude of the charge in coulombs, and r is the distance from the particle in meters. The electric field of two or more particles is calculated by using superposition. For example, the electric potential at a point due to two particles is given by

$$V = \frac{1}{4\pi\varepsilon_0}\left(\frac{q_1}{r_1} + \frac{q_2}{r_2}\right)$$

where q_1, q_2, r_1, and r_2 are the charges of the particles and the distance from the point to the corresponding particle, respectively.

Two particles with a charge of $q_1 = 2 \times 10^{-10}$ C and $q_2 = 3 \times 10^{-10}$ C are positioned in the $x\,y$ plane at points (0.25, 0, 0) and (–0.25, 0, 0), respectively, as shown. Calculate and plot the electric potential due to the two particles at points in the $x\,y$ plane that are located in the domain $-0.2 \le x \le 0.2$ and $-0.2 \le y \le 0.2$ (the units in the $x\,y$ plane are meters). Make the plot such that the $x\,y$ plane is the plane of the points, and the z axis is the magnitude of the electric potential.



Solution

The problem is solved by following these steps:

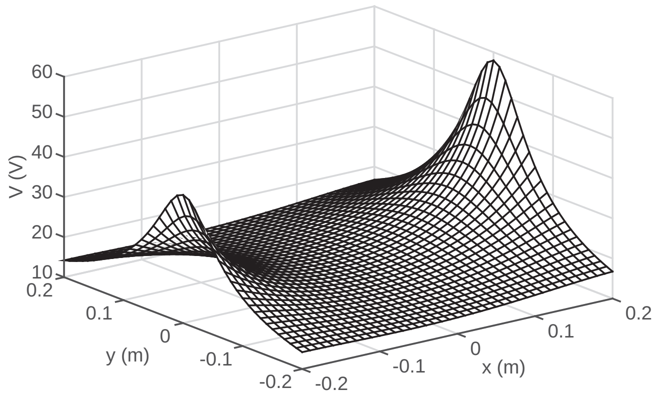
(*a*) A grid is created in the $x\,y$ plane with the domain $-0.2 \le x \le 0.2$ and $-0.2 \le y \le 0.2$.
(*b*) The distance from each grid point to each of the charges is calculated.
(*c*) The electric potential at each point is calculated.
(*d*) The electric potential is plotted.

The following is a program in a script file that solves the problem.

```
eps0=8.85e-12; q1=2e-10; q2=3e-10;
k=1/(4*pi*eps0);
x=-0.2:0.01:0.2;
y=-0.2:0.01:0.2;
[X,Y]=meshgrid(x,y);                    Creating a grid in the x y plane.
r1=sqrt((X+0.25).^2+Y.^2);    Calculating the distance r1 for each grid point.
```

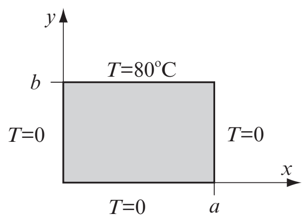
```
r2=sqrt((X-0.25).^2+Y.^2);     Calculating the distance r2 for each grid point.
V=k*(q1./r1+q2./r2);           Calculating the electric potential V at each grid point.
mesh(X,Y,V)
xlabel('x (m)'); ylabel('y (m)'); zlabel('V (V)')
```

The plot generated when the program runs is:



Sample Problem 10-3: Heat conduction in a square plate

Three sides of a rectangular plate (a = 5 m, b = 4 m) are kept at a temperature of 0°C and one side is kept at a temperature T_1 = 80°C, as shown in the figure. Determine and plot the temperature distribution $T(x, y)$ in the plate.



Solution

The temperature distribution, $T(x, y)$ in the plate can be determined by solving the two-dimensional heat equation. For the given boundary conditions $T(x, y)$ can be expressed analytically by a Fourier series (Erwin Kreyszig, *Advanced Engineering Mathematics*, John Wiley and Sons, 1993):

$$T(x, y) = \frac{4T_1}{\pi} \sum_{n=1}^{\infty} \frac{\sin\left[(2n-1)\frac{\pi x}{a}\right]}{(2n-1)} \frac{\sinh\left[(2n-1)\frac{\pi y}{a}\right]}{\sinh\left[(2n-1)\frac{\pi b}{a}\right]}$$

A program in a script file that solves the problem is listed below. The program follows these steps:

(*a*) Create an X, Y grid in the domain $0 \le x \le a$ and $0 \le y \le b$. The length of the plate, a, is divided into 20 segments, and the width of the plate, b, is divided into 16 segments.

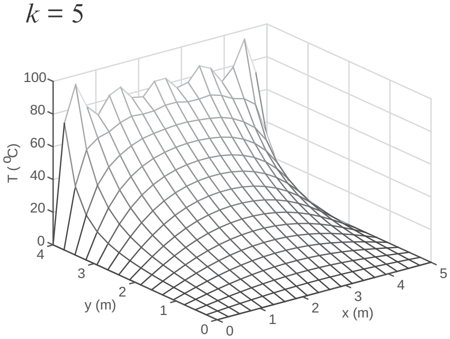
(*b*) Calculate the temperature at each point of the mesh. The calculations are done point by point using a double loop. At each point the temperature is determined by adding k terms of the Fourier series.

(c) Make a surface plot of T.

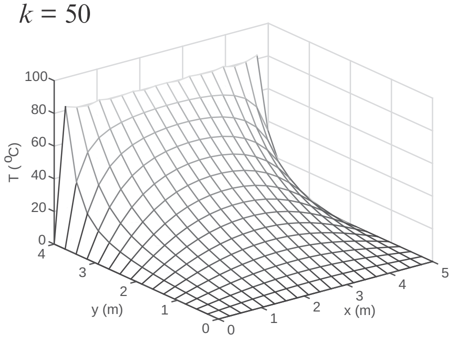
```
a=5; b=4; na=20; nb=16; k=5; T0=80;
clear T
x=linspace(0,a,na);
y=linspace(0,b,nb);
[X,Y]=meshgrid(x,y);                  Creating a grid in the x y plane.
for i=1:nb                            First loop, i, is the index of the grid's row.
    for j=1:na                        Second loop, j, is the index of the grid's column.
        T(i,j)=0;
        for n=1:k                     Third loop, n, is the nth term of the Fourier
            ns=2*n-1;                 series, k is the number of terms.
      T(i,j)=T(i,j)+sin(ns*pi*X(i,j)/a).*sinh(ns*pi*Y(i,j)/
a)/(sinh(ns*pi*b/a)*ns);
        end
        T(i,j) = T(i,j)*4*T0/pi;
    end
end
mesh(X,Y,T)
xlabel('x (m)'); ylabel('y (m)'); zlabel('T ( ^oC)')
```

The program was executed twice, first using five terms (k = 5) in the Fourier series to calculate the temperature at each point, and then with k = 50. The mesh plots created in each execution are shown in the figures below. The temperature should be uniformly 80°C at y = 4 m. Note the effect of the number of terms (k) on the accuracy at y = 4 m.

k = 5



k = 50



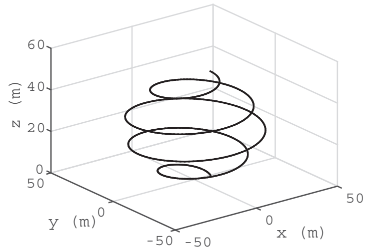
10.6 PROBLEMS

1. The position of a moving particle as a function of time is given by:

$$x = 0.01(30 - t)^2 \sin(2t) \qquad y = 0.01(30 - t)^2 \cos(2t) \qquad z = 0.5\,t^{1.5}$$

Plot the position of the particle for $0 \le t \le 20$ s.

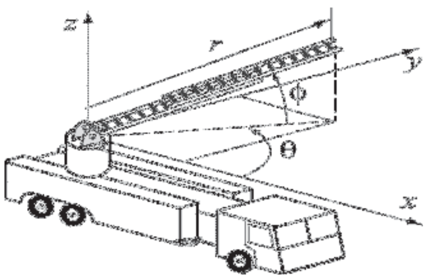
2. A staircase of height h is modeled by the parametric equations:



$$x = r\cos(t) \quad y = r\sin(t) \quad z = \frac{ht}{2\pi n}$$

where $r = h[2 + 5\sin(t/8)]/10$, $n = 4$, and $h = 50$ m is the staircase height. Make a 3-D plot (shown) of the staircase. (Create a vector t for the domain 0 to $2\pi n$, and use the `plot3` command.)

3. The ladder of a fire truck can be elevated (increase of angle ϕ), rotated about the z axis (increase of angle θ), and extended (increase of r). Initially the ladder rests on the truck ($\phi = 0$, $\theta = 0$, and $r = 8$ m). Then the ladder is moved to a new position by raising the ladder at a rate of 5 deg/s, rotating at a rate of 8 deg/s, and extending the ladder at a rate of 0.6 m/s. Determine and plot the position of the tip of the ladder for 10 s.



4. Make a 3-D surface plot of the function $z = \frac{x^2}{4} + 2\sin^2(0.7y)$ in the domain $-4 \le x \le 4$ and $-3 \le y \le 3$.

5. Make a 3-D surface plot of the function $z = -0.7x^4 - 0.7y^4$ in the domain $-2 \le x \le 2$ and $-2 \le y \le 2$.

6. Make a 3-D surface plot of the function $z = -1.4xy^3 + 1.4yx^3$ in the domain $-2 \le x \le 2$ and $-2 \le y \le 2$.

7. Make a 3-D mesh plot of the function $z = \frac{-\cos 2R}{e^{0.2R}}$, where $R = \sqrt{x^2 + y^2}$ in the domain $-5 \le x \le 5$ and $-5 \le y \le 5$.

8. Make a 3-D surface plot of the function $z = \cos(0.7x + 0.7y)\cos(0.7x - 0.7y)$ in the domain $-\pi \le x \le \pi$ and $-\pi \le y \le \pi$.

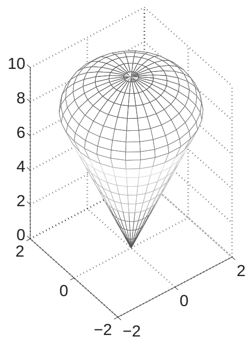
9. Make a plot of the ice cream cone shown in the figure. The cone is 8 in. tall with a 4-in. diameter base. The ice cream at the top is a 4-in. diameter hemisphere.
A parametric equation for the cone is:

$$x = r\cos\theta\ ,\quad y = r\sin\theta\ ,\quad z = 4r$$
$$\text{with } 0 \le \theta \le 2\pi \text{ and } 0 \le r \le 2$$

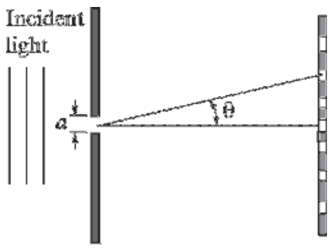
A parametric equation for a sphere is:

$$x = r\cos\theta\sin\phi\ ,\quad y = r\sin\theta\sin\phi\ ,\quad z = r\cos\phi$$

with $0 \le \theta \le 2\pi$ and $0 \le \phi \le \pi$

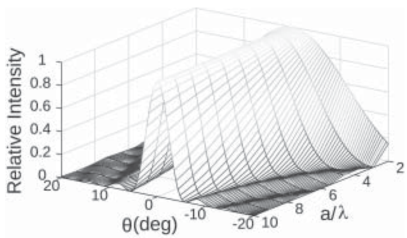


10. A monochromatic light that passes through a slit produces on a screen a diffraction pattern consisting of bright and dark fringes. The intensity of the bright fringes, I, as a function of θ can be calculated by:



$$I = I_{\max}\left(\frac{\sin\alpha}{\alpha}\right)^2$$

where $\alpha = \frac{\pi a}{\lambda}\sin\theta$, λ is the light wave length, a is the width of the slits. Make a 3-D plot (shown) that shows the relative intensity $I/I_{\max}$ as a function of θ for $-20° \le \theta \le 20°$, and a function of a/λ for $2 \le a/\lambda \le 10$.



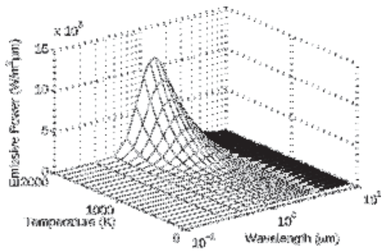
11. Molecules of a gas in a container are moving around at different speeds. Maxwell's speed distribution law gives the probability distribution $P(v)$ as a function of temperature and speed:

$$P(v) = 4\pi\left(\frac{M}{2\pi RT}\right)^{3/2} v^2 e^{-Mv^2/2RT}$$

where M is the molar mass of the gas in kg/mol, $R = 8.31$ J/(mol K), is the gas constant, T is the temperature in kelvins, and v is the molecule's speed in m/s.

Make a 3-D plot of $P(v)$ as a function of v and T for $0 \le v \le 1{,}000$ m/s and $70 \le T \le 320$ K for oxygen (molar mass 0.032 kg/mol).

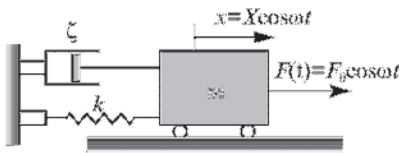
12. Plank's distribution law gives the black-body emissive power (amount of radiation energy emitted) as a function of temperature and wavelength:



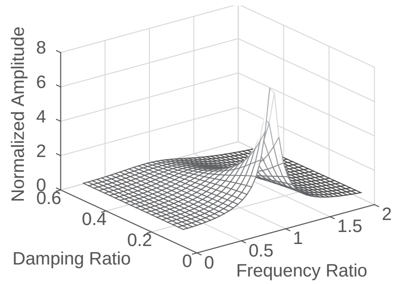
$$E = \frac{C_1}{\lambda^5\left[e^{C_2/\lambda T} - 1\right]} \qquad \left(\frac{\text{W}}{\text{m}^2\mu\text{m}}\right)$$

where $C_1 = 3.742 \times 10^8$ Wμm^4/m^2, $C_2 = 1.439 \times 10^4$ μmK, T is the temperature in degrees K, and λ is the wavelength in μm. Make a 3-D plot (shown in the figure) of E as a function of λ ($0.1 \le \lambda \le 10$ μm) and T for $100 \le T \le 2000$ K. Use a logarithmic scale for λ. This can be done with the command: `set(gca,'xscale','log')`.

13. Consider steady-state vibration of a friction-free spring-mass-damper system subjected to harmonic applied force. The normalized amplitude of the mass is given by:

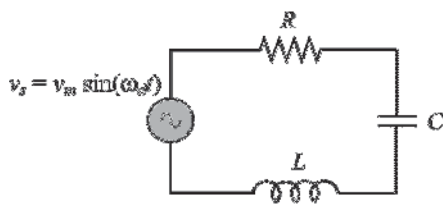


$$\frac{Xk}{F_0} = \frac{1}{\sqrt{(1-r^2)^2 + (2\zeta r)^2}}$$



where $r = \omega/\omega_n$ is the frequency ratio, and ζ is the damping ratio. Make a 3-D plot (shown) of the normalized amplitude (z axis) as a function of the frequency ratio for $0 \le r \le 2$, and a function of the damping ratio for $0.05 \le \zeta \le 0.5$.

14. An RLC circuit with an alternating voltage source is shown. The source voltage v_s is given by $v_s = v_m \sin(\omega_d t)$, where $\omega_d = 2\pi f_d$, in which f_d is the driving frequency. The amplitude of the current, I, in this circuit is given by:



$$I = \frac{v_m}{\sqrt{R^2 + [\omega_d L - 1/(\omega_d C)]^2}}$$

where R and C are the resistance of the resistor and capacitance of the capacitor, respectively. For the circuit in the figure $C = 15 \times 10^{-6}$ F, $L = 240 \times 10^{-3}$ H, and $v_m = 24$ V.

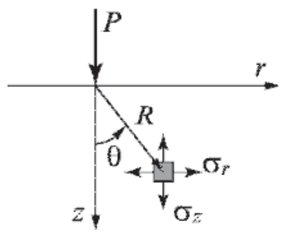
(a) Make a 3-D plot of I (z axis) as a function of ω_d (x axis) for

$60 \le f \le 110$ Hz, and as a function of R (y axis) for $10 \le R \le 40\,\Omega$.

(*b*) Make a plot that is a projection on the $x\,z$ plane. Estimate from this plot the natural frequency of the circuit (the frequency at which I is maximum). Compare the estimate with the calculated value of $1/(2\pi\sqrt{LC})$.

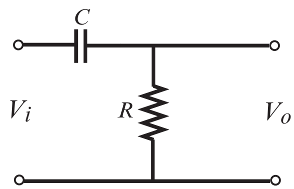
15. In the solution of elasticity problem of a normal point load applied to the surface of a half plane that was solved by Boussinesq in 1878, the stresses σ_r and σ_z are given by:

$$\sigma_z = -\frac{3Pz^3}{2\pi R^5} \quad \text{and} \quad \sigma_r = \frac{P}{2\pi}\left[\frac{1-2\nu}{R(R+z)} - \frac{3r^2 z}{R^5}\right]$$



where ν is Poisson's ratio. For $P = 2,000$ lb and $\nu = 0.3$, plot the stress components (each in a separate figure) as a function of r and z in the domain $0 \le \theta \le 90°$ and $0.02 \le R \le 0.1$ in. Plot the coordinates r and z in the horizontal plane and the stresses in the vertical direction.

16. A high-pass filter passes signals with frequencies that are higher than a certain cutoff frequency. In this filter the ratio of the magnitudes of the voltages is given by:



$$\left|\frac{V_o}{V_i}\right| = \frac{\omega RC}{\sqrt{1+\omega^2 R^2 C^2}}$$

where $\omega = 2\pi f$ is the frequency of the input signal.

(*a*) Make a 3-D mesh plot of $\left|\frac{V_o}{V_i}\right|$ (z axis) as a function of f (x axis) for $1 \le f \le 10^6$ Hz, and as a function of RC (y axis) for $0.4 \times 10^{-4} \le RC \le 6 \times 10^{-3}$ s. Use a logarithmic scale for the x axis. This can be done by typing the MATLAB command `set(gca,'Xscale','log')` following the `mesh` command. A vector with constant spacing on a logarithmic scale can be created with the command `logspace(a,b,n)`.

(*b*) Make a plot that is a projection on the $x\,z$ plane.

17. The equation for the streamlines for uniform flow over a cylinder is

$$\psi(x, y) = y - \frac{y}{x^2+y^2}$$

where ψ is the stream function. For example, if $\psi = 0$, then $y = 0$. Since the equation is satisfied for all x, the x axis is the zero ($\psi = 0$) streamline. Observe that the collection of points where $x^2 + y^2 = 1$ is also a streamline. Thus, the stream function above is for a cylinder of radius 1. Make a 2-D contour plot of the streamlines around a cylinder with 1 in. radius. Set up the domain for x and y to range between –3 and 3. Use 100 for the number of contour levels. Add to the figure a plot of a circle with a radius of 1. Note that MATLAB also plots streamlines inside the cylinder. This is a mathematical artifact.

18. The deflection w of a clamped circular membrane of radius r_d subjected to pressure P is given by (small deformation theory):

$$w(r) = \frac{Pr_d^4}{64K}\left[1-\left(\frac{r}{r_d}\right)^2\right]^2$$

where r is the radial coordinate, and $K = \frac{Et^3}{12(1-\nu^2)}$, where E, t, and ν are the elastic modulus, thickness, and Poisson's ratio of the membrane, respectively. Consider a membrane with $P = 15$ psi, $r_d = 15$ in., $E = 18\times10^6$ psi, $t = 0.08$ in., and $\nu = 0.3$. Make a surface plot of the membrane.

19. The Verhulst model, given in the following equation, describes the growth of a population that is limited by various factors such as overcrowding and lack of resources:

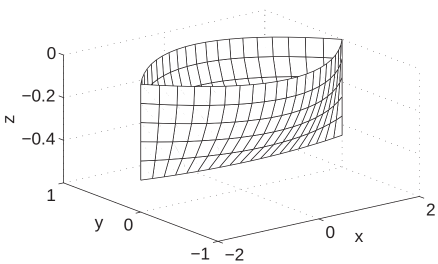
$$N(t) = \frac{N_\infty}{1+\left(\frac{N_\infty}{N_0}-1\right)e^{-rt}}$$

where $N(t)$ is the number of individuals in the population, N_0 is the initial population size, N_∞ is the maximum population size possible due to the various limiting factors, and r is a rate constant. Make a surface plot of $N(t)$ versus t and N_∞ assuming $r = 0.1\,\text{s}^{-1}$, and $N_0 = 10$. Let t vary between 0 and 100 and N_∞ between 100 and 1,000.

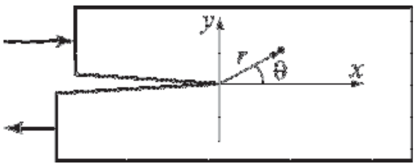
20. The geometry of a ship hull (Wigley hull) can be modeled by the equation:

$$y = \pm\frac{B}{2}\left[1-\left(\frac{2x}{L}\right)^2\right]\left[1-\left(\frac{2z}{T}\right)^2\right]$$

where x, y, and z are the length, width, and height, respectively. Use MATLAB to make a 3-D figure of the hull as shown. Use $B = 1.2$, $L = 4$, $T = 0.5$, $-2 \le x \le 2$, and $-0.5 \le z \le 0$.



21. The stress fields near a crack tip of a linear elastic isotropic material for mode II loading are given by:



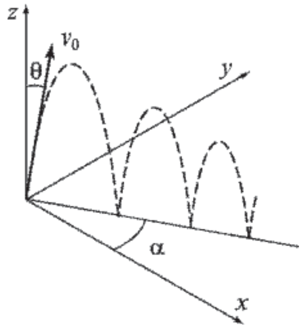
$$\sigma_{xx} = -\frac{K_{II}}{\sqrt{2\pi r}}\sin\frac{\theta}{2}\left(2+\cos\frac{\theta}{2}\cos\frac{3\theta}{2}\right)$$

$$\sigma_{yy} = \frac{K_{II}}{\sqrt{2\pi r}}\sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2}$$

$$\sigma_{xy} = \frac{K_{II}}{\sqrt{2\pi r}}\cos\frac{\theta}{2}\left(1-\sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)$$

For K_{II} = 300 ksi$\sqrt{\text{in.}}$ plot the stresses (each in a separate figure) in the domain $0 \le \theta \le 90°$ and $0.02 \le r \le 0.2$ in. Plot the coordinates x and y in the horizontal plane, and the stresses in the vertical direction.

22. A ball thrown up falls back to the floor and bounces many times. For a ball thrown up in the direction shown in the figure, the position of the ball as a function of time is given by:



$$x = v_x t \qquad y = v_y t \qquad z = v_z t - \frac{1}{2}gt^2$$

The velocities in the x and y directions are constants throughout the motion and are given by $v_x = v_0\sin(\theta)\cos(\alpha)$ and $v_y = v_0\sin(\theta)\sin(\alpha)$. In the vertical z direction the initial velocity is $v_z = v_0\cos(\theta)$, and when the ball impacts the floor its rebound velocity is 0.8 of the vertical velocity at the start of the previous bounce. The time between bounces is given by $t_b = 2v_z/g$, where v_z is the vertical component of the velocity at the start of the bounce. Make a 3-D plot (shown in the figure) that shows the trajectory of the ball during the first five bounces. Take $v_0 = 20$ m/s, $\theta = 30°$, $\alpha = 25°$, and $g = 9.81$ m/s^2.