3-D Visualization and Colormaps

We can create plots of 3-D curves and points using

plot3(x, y, z, ...)

scatter3(x, y, z, ...)

```
% create evenly spaced angles for the spiral

angles = linspace(0, 6*pi, 100);

% create a 3D plot of a spiral curve and add axis labels

plot3(cos(angles), sin(angles), angles, 'g-*', 'LineWidth', 3);

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

% create a 3D scatter plot of random points
% rand(1,100) returns a vector of 100 random numbers between 0 and 1

scatter3(rand(1,100), rand(1,100), rand(1,100), 40, 'm', 'filled');

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

Suppose we want to graph the function

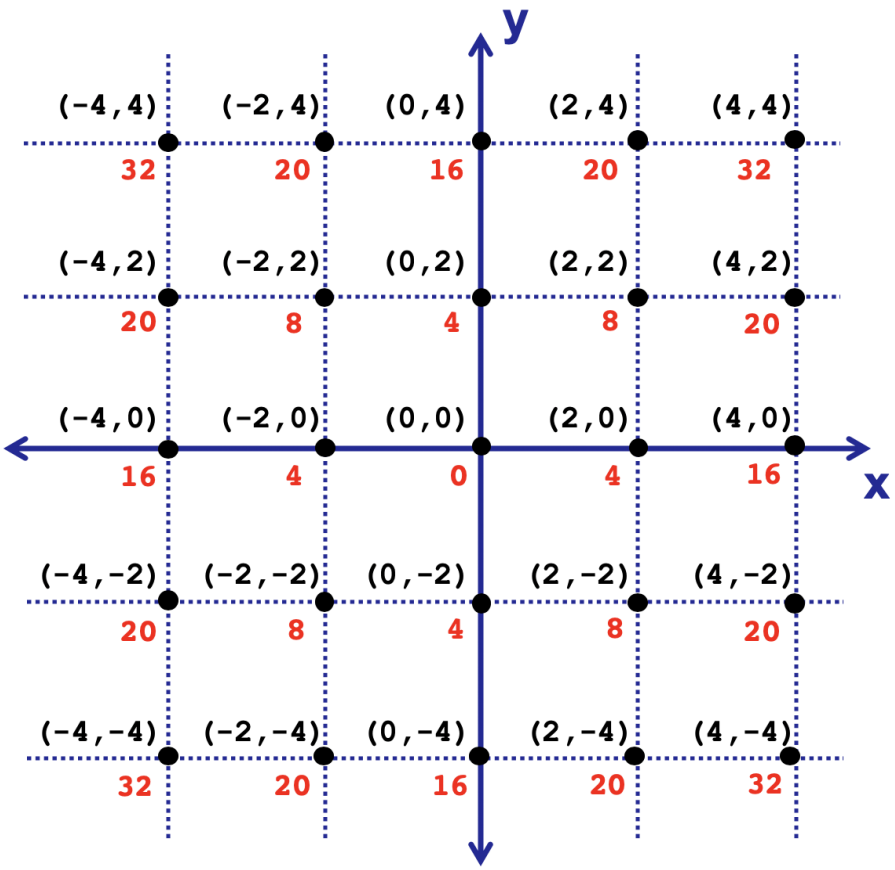
$$z = x^2 + y^2 \quad -4 \leq x,y \leq +4$$

We can get there in three steps:

```
% create matrices of x,y coordinates

[x,y] = meshgrid(-4:0.2:4);

% calculate z from the x,y coordinates

z = x.^2 + y.^2;

% display x,y,z coordinates as a mesh

mesh(x,y,z);

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

Let's examine **meshgrid** more closely.

The picture below shows a grid of regularly spaced points (black dots) whose x and y coordinates range from -4 to 4 in increments of 2. The (x,y) coordinates of each point are shown in black, and value of z is shown in red.



meshgrid returns two matrices that store the x and y coordinates for regularly spaced locations on a grid. The desired values for each coordinate are given as inputs.

```
% create 5x5 matrices of x,y coordinates, where both coordinates
% span the values [-4 -2 0 2 4]

[x,y] = meshgrid(-4:2:4)

% calculate z from the x,y coordinates

z = x.^2 + y.^2

% meshgrid can create matrices of coordinates with different
% values for x and y. Here the x values are [1 4 7 10] and
% y values are [-2 0 2]

[x,y] = meshgrid(1:3:10, -2:2:2)
```

The **peaks** function returns matrices of coordinates that can be used directly with **mesh**. The **surf** function displays a 3D surface, and **colorbar** shows the current colormap.

```
% get x,y,z coordinates (in 25x25 matrices) for the "peaks" function

[x,y,z] = peaks(25);

% draw the mesh in black

mesh(x,y,z,'EdgeColor','k');

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

% use the same x,y,z coordinates to display a surface and colorbar

surf(x,y,z)

colorbar

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

Change the colormap

```
% change the colormap (see MATLAB documentation for named colormaps)

surf(x,y,z)

colormap(spring)
colorbar

surf(x,y,z)

colormap(winter)
colorbar
```

How is the colormap represented and stored?

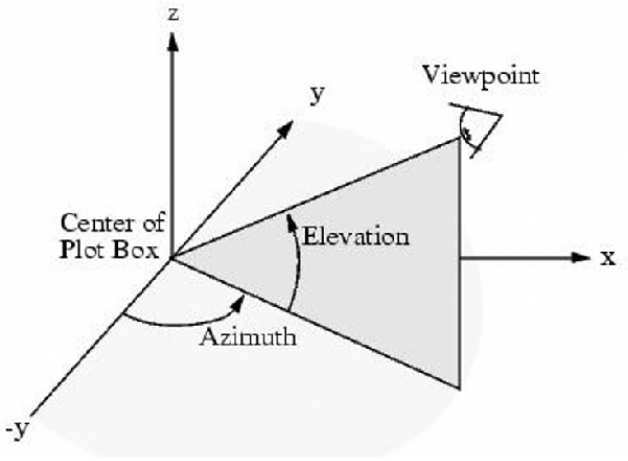
```
% create a colormap with 10 distinct colors from the jet colormap

surf(x,y,z)

cmap = colormap(jet(10));
colorbar('southoutside')

cmap
```

Make your own colormap

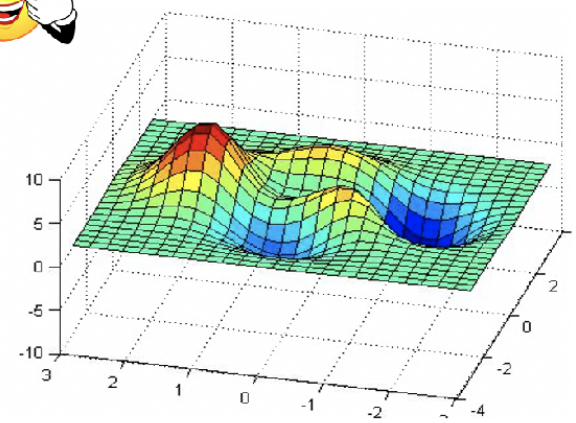
```
% make a colormap with 10 shades of purple, which all
% have equal amounts of red and blus

purples = zeros(10,3);
purples(:,1) = linspace(0,1,10);
purples(:,3) = linspace(0,1,10);

purples

surf(x,y,z)
colormap(purples)
```

Change your point of view

view(azimuth, elevation)



view(-75, 45)



default: view(-37.5, 30)

```
% display the surface and turn on an interactive tool
% to change the viewpoint

surf(x,y,z)

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

rotate3d on
```

Display your 3D object with smooth shading

```
% create matrices of x,y,z coordinates for the surface of a sphere

[x,y,z] = sphere(20);

surf(x,y,z)
```

```
axis equal

colormap(gray)

% view shading in different ways - facets have constant
% color within each patch

shading faceted

surf(x,y,z)

axis equal

% faceted shading with mesh lines removed

shading flat

surf(x,y,z)

axis equal

% smooth shading across the surface

shading interp
```

Add a light source and a shiny surface

```
% show a "peaks" surface with denser samples

[x,y,z] = peaks(100);

surf(x,y,z)

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

axis tight

colormap(copper)

shading interp

% add a light source in the direction of the vector [0 -1 0]

light('Position', [0 -1 0])

% make the surface shiny

material shiny
```