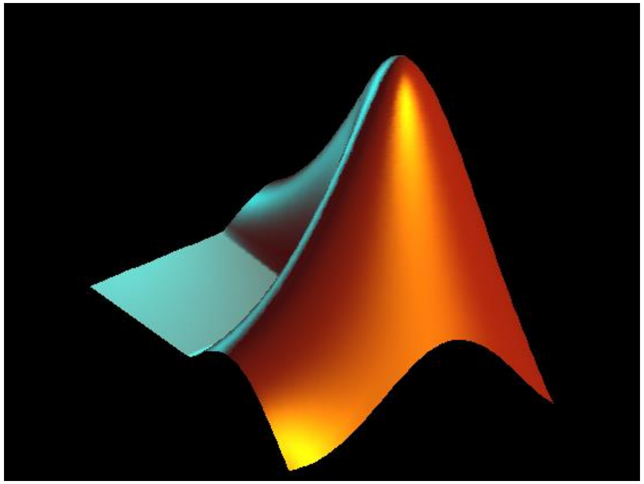
An Overview of 2D and 3D Coordinate Systems



Topic 5

Outline

- Introduction, Euclidean Spaces
- 2D Cartesian, Rotation of Axes
- 2D Polar
- 3D Cartesian
- (3D) Cylindrical
- (3D) Spherical

MATLAB Examples & Problems

- 2D Polar Plots
- 3D Line Plots
- 3D Surface Plots

Introduction

Coordinate system: a system which uses an ordered list of numbers to uniquely represent the location of a point in a given domain (line, plane, space, etc.)

Euclidean spaces

Recall that $\mathbb{R}$ denotes the set of real numbers, (-∞, ∞).

The set $\mathbb{R}^n$, which is the set of all *n*-tuples $(x_1, x_2, \ldots.., x_n)$, is called **Euclidean *n*-space.** An *n*-tuple is simply an ordered list of *n* elements where *n* is a finite non-negative integer.

An *n*-tuple can either represent a point or a position vector in *n*-dimensional space.

For example $(\sqrt{2}, -3, \pi^2, 0, {}^8/_7)$ can represent a point in 5D space or the position vector of this point.

2D and 3D Space

Examples

$\mathbb{R}$ or $\mathbb{R}^1$ - 1D space or **real number line**
$\mathbb{R}^2$ - 2D space or **(2D) plane**
$\mathbb{R}^3$ - **3D space**

We tend to think of vectors as directed line segments, especially in 2D and 3D space.

$c \in \mathbb{R}$ may represent

- a *point* in 1D space (the location of c on the real number line)
- a *vector* pointing from 0 to c along the real number line
- the *scalar (numerical value)* c

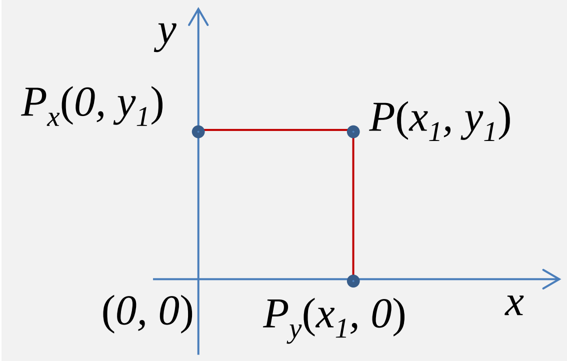
2D Cartesian Coordinate System

- x and y coordinate axes intersect at right angles
- each point in the plane is assigned a unique (x, y) pair of coordinates

$P(x_1, y_1)$: point in 2D plane

P_x and P_y : perpendicular projections of P onto the coordinate axes

x_1 : $O\,P_x$ distance
y_1 : $O\,P_y$ distance



Example (explicit) functions:

$y = 2$ line

$y = x^2$ parabola

$y = 1/x$ hyperbola

$y = \sqrt{4 - x^2}$ semi-circle

Example (implicit) functions:

$x^2 + y^2 = 4$ circle

$\frac{x^2}{4} + \frac{y^2}{9} = 1$ ellipse

$y^2 - x = 1$ parabola

$\frac{x^2}{4} - \frac{y^2}{9} = 1$ hyperbola

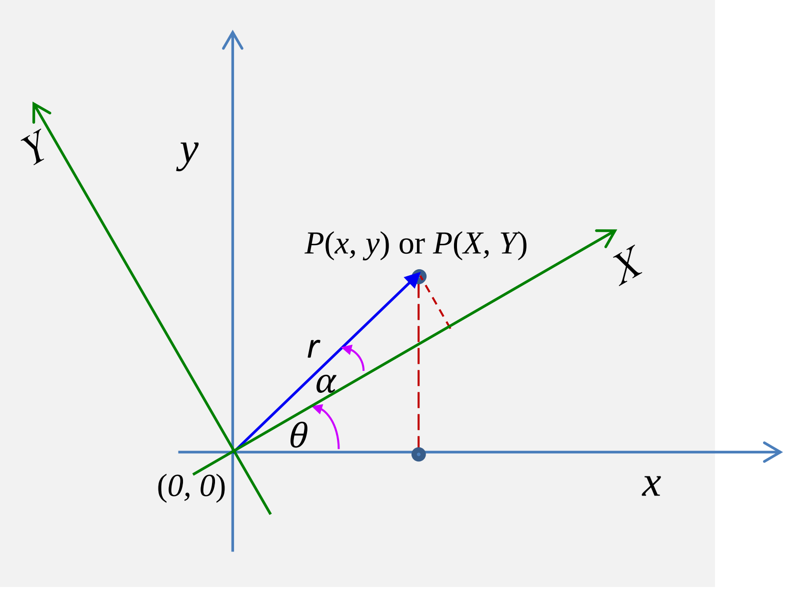
Rotation of Axes

$P(x, y)$: point in 2D plane

X and Y are obtained by rotating the x and y axes about the origin by θ

$P(X, Y)$: coordinates in the (X,Y) system

angle between x axis and r: $\alpha + \theta$
angle between X axis and r: α



Old and new coordinates:

$X = r\cos(\alpha) \qquad Y = r\sin(\alpha)$

$x = r\cos(\theta + \alpha) \qquad y = r\sin(\theta + \alpha)$

using the sine and cosine addition formulas

$$x = X\cos(\theta) - Y\sin(\theta)$$

$$y = X\sin(\theta) + Y\cos(\theta)$$

Matrix form:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}$$

We can use this relationship to plot a variety of shapes rotated about the origin. Some good examples and the details of the derivation can be found [here]

2D Polar Coordinate System

- *x* and *y* coordinate axes intersect at right angles
- each point in the plane can be assigned an (r, θ) pair of coordinates
- polar coordinates are not unique

$P(r, \theta)$: point in 2D plane

r : OP distance

θ: *angle between* $\overrightarrow{OP}$ *and x axis*

Example:

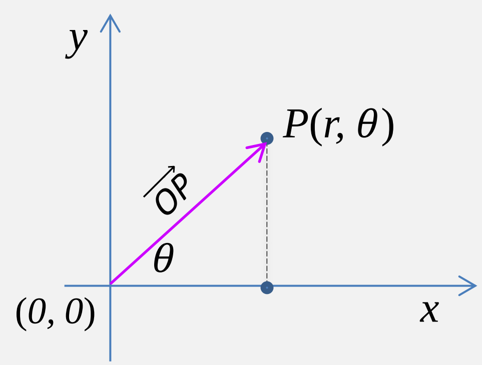
Cartesian coordinates: $\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)$

Polar coordinates:

$\left(1, \frac{\pi}{4}\right), \left(1, \frac{9\pi}{4}\right), \left(1, -\frac{7\pi}{4}\right)$, *etc*

general formula:

$\left(1, \frac{\pi}{4} + 2n\pi\right)$, *where* $n \in \mathbb{Z}$



Cartesian $\leftrightarrow$ polar conversion

$$x = r\cos(\theta)$$

$$y = r\sin(\theta)$$

$$r = \sqrt{x^2 + y^2}$$

$$\tan(\theta) = y/x$$

Cartesian vs Polar Equations

Example Curve	Cartesian Equation	Polar equation
Circle (centred at origin)	$x^2 + y^2 = 4$	$r = 2$
Circle (translated)	$(x-1)^2 + y^2 = 1$	$r = 2\cos(\theta)$
Ellipse (translated)	$\frac{(x+3)^2}{25} + \frac{y^2}{16} = 1$	$r = \frac{16}{5 + 3\cos(\theta)}$
Vertical line	$x = 4$	$r = 4\sec(\theta)$
Horizontal line	$y = 2$	$r = 2\csc(\theta)$
Line through the origin	$y = x$	$\theta = \pi/4$
Parabola	$x^2 - 2y = 1$	$r = \frac{1}{1 - \sin(\theta)}$

3D Cartesian Coordinate System

- *x*, *y* and *z* axes are pairwise perpendicular
- each point is assigned a unique (*x*, *y*, *z*) triplet

<u>Examples of 3D surfaces</u>:

$x = 2$ vertical plane

$x + y + z = 0$ plane through the origin

$x + y + z = 3$ plane bypassing the origin

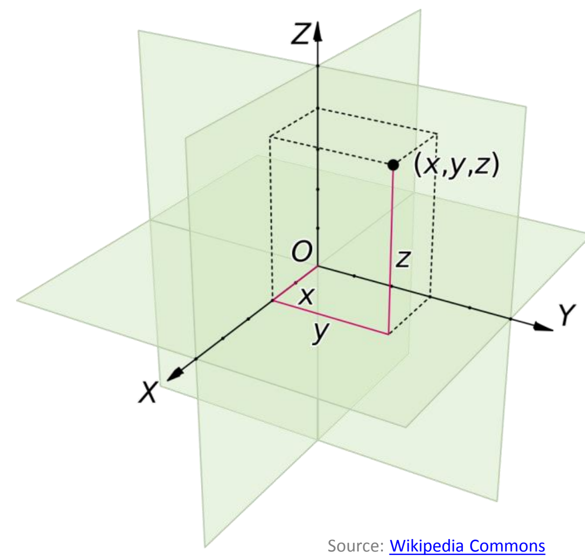
$x^2 + y^2 = 4$ cylinder centred on the *z* axis

$x^2 + y^2 + z^2 = 1$ unit sphere

$z = \sqrt{4x^2 + y^2}$ elliptic cone

*check MOLE for additional reading on **quadric surfaces***

right-handed 3D Cartesian coordinate system



Source: Wikipedia Commons

(3D) Cylindrical Coordinate System

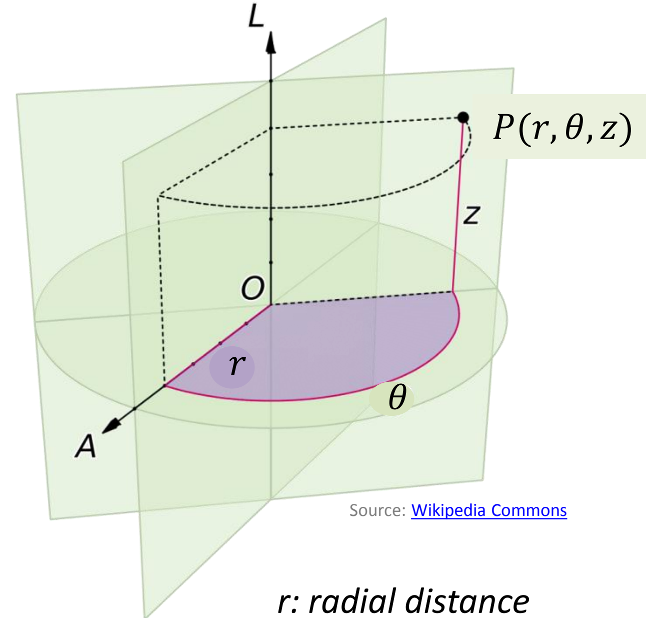
- extension of polar coordinates into 3D by the addition of the *z* axis
- each point can be assigned an (*r*, *θ*, *z*) triplet (not unique!)

Cartesian ↔ cylindrical conversion

$$x = r\cos(\theta) \qquad r = \sqrt{x^2 + y^2}$$

$$y = r\sin(\theta) \qquad \tan(\theta) = y/x$$

$$z = z \qquad z = z$$



Source: Wikipedia Commons

r: radial distance
θ: azimuth
z: height

Examples
(try to convert back to Cartesian coordinates)

$r = 2$ cylinder

$r^2 + z^2 = 25$ sphere

$z = r$ cone

$z^2 = 1 - 4r^2$ ellipsoid

(3D) Spherical Coordinate System

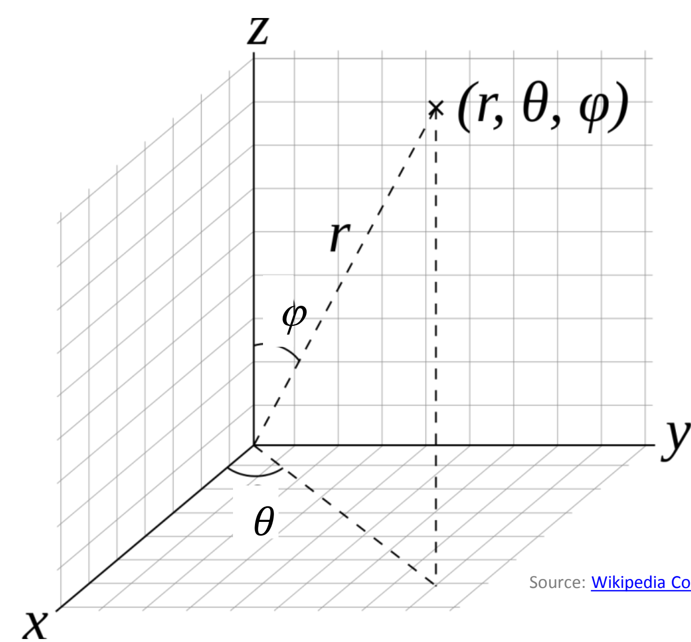
- extension of polar coordinates into 3D
- each point can be assigned an (r, θ, φ) triplet (not unique!)

Cartesian ↔ spherical conversion

$$x = r\sin(\varphi)\cos(\theta) \qquad r = \sqrt{x^2 + y^2 + z^2}$$

$$y = r\sin(\varphi)\sin(\theta) \qquad \tan(\theta) = y/x$$

$$z = r\cos(\varphi) \qquad \tan(\varphi) = (\sqrt{x^2 + y^2})/z$$



Source: Wikipedia Commons

r: radial distance
θ: azimuth
φ: polar angle

Examples
(try to convert back to Cartesian coordinates)

$r = 2$ sphere

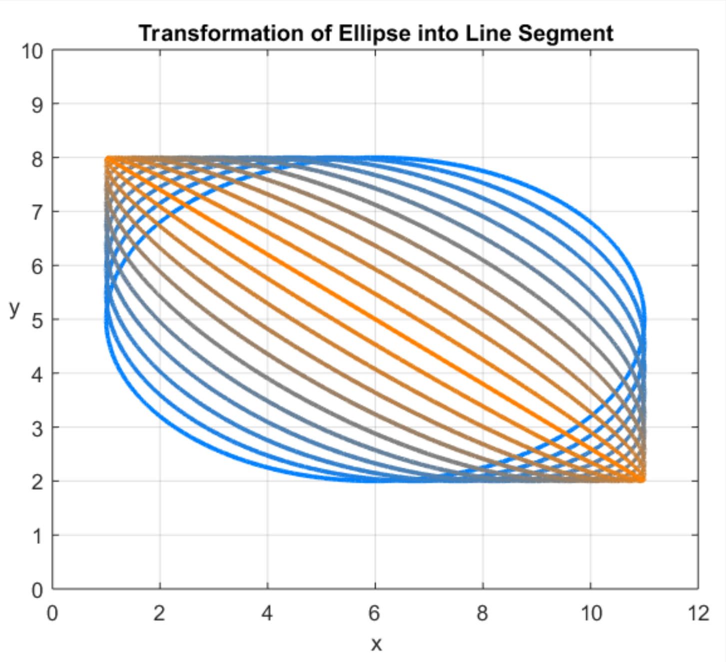
$\varphi = \pi/4$ half-cone

$\theta = \pi/2$ half-plane (*yz*-plane where *y*>0)

$r\sin(\varphi) = 2$ cylinder

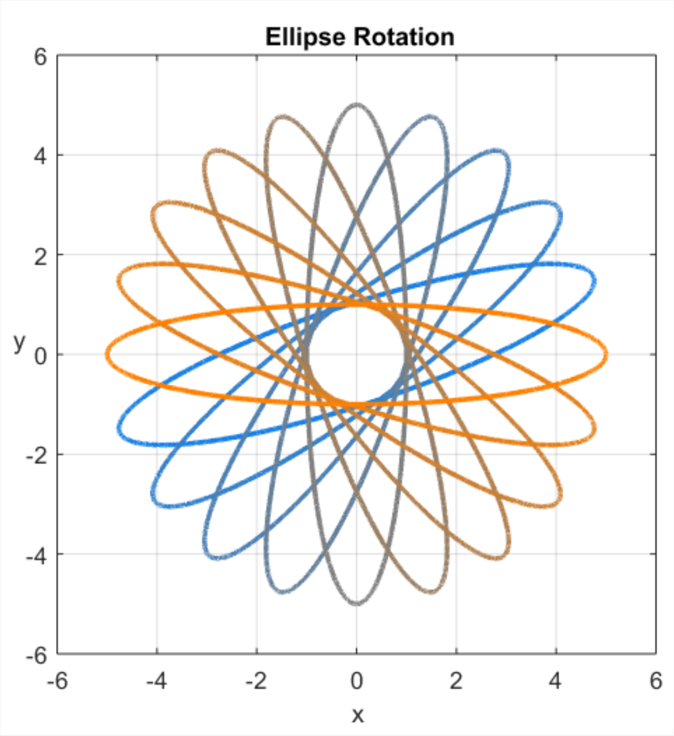
Example 1
Deformation

```
close all
% plot area settings
title('Transformation of Ellipse into Line Segment')
hold on, axis equal, axis([0,12,0,10])
grid on, box on
xlabel('x'), ylabel('y', 'rot',0)

% Ellipse geometry
% major & minor axes
a=5; b=3;
% centre coordinates
c=[6,5];
% angle parameter
t=-pi:0.001:pi;

n=10;
for k=0:n
% keep adding pi/20 to the angle in the x coordinate
x=a*cos(t+k*pi/20)+c(1);  % x(t) parametric equation
y=b*sin(t)+c(2);    % leave t in y(t) unchanged
plot(x,y,'color', [k/n 1/2 1-k/n],'LineWidth',2)
%pause(1)
end
```

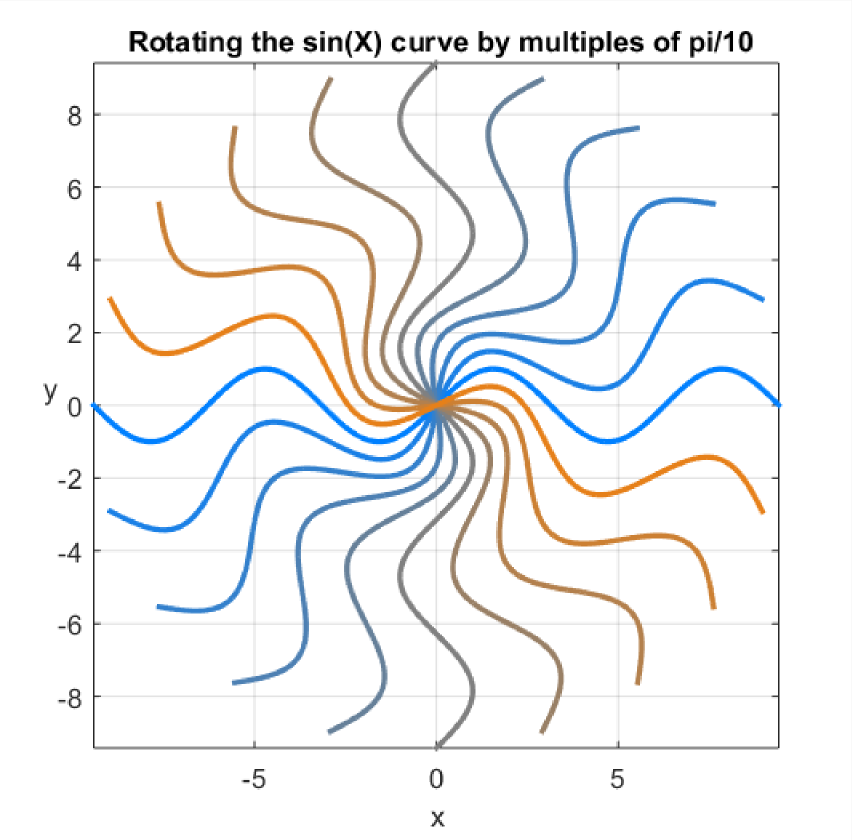


Example 2
Rotation

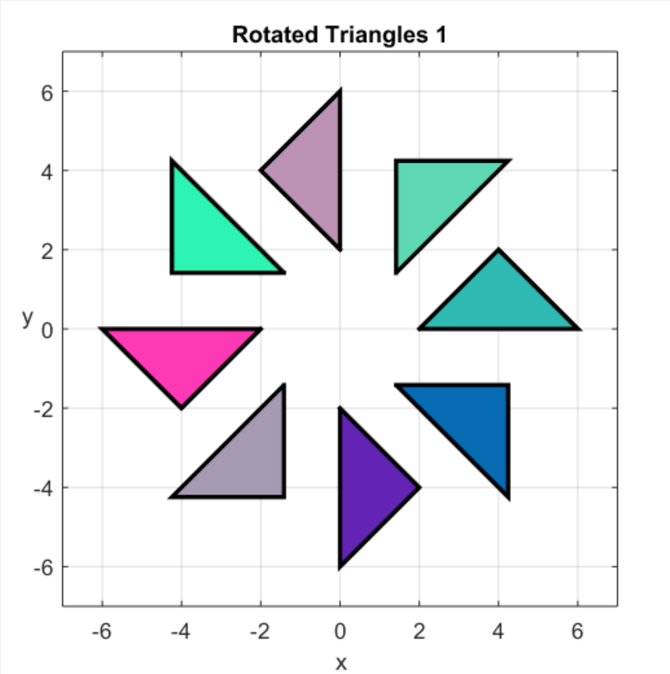
```
%starting ellipse: a=5 & b = 1 centred at the origin
close all
% plot area settings
title(' Ellipse Rotation')
hold on
axis equal, axis([-6 6, -6 6])
grid on, box on
xlabel('x'), ylabel('y', 'rot',0)

% Ellipse geometry
a=5; b=1;
t=-pi:0.001:pi;
% unrotated ellipse
X=a*cos(t); Y=b*sin(t);

n=10;
for k=1:n
% angle of rotation: (k*pi/n)
x=X.*cos(pi*k/n)-Y.*sin(pi*k/n); % coordinates in the old system
y=X.*sin(pi*k/n)+Y.*cos(pi*k/n);
plot(x,y,'color', [k/n 1/2 1-k/n],'LineWidth',2)
% pause(1)
end
```

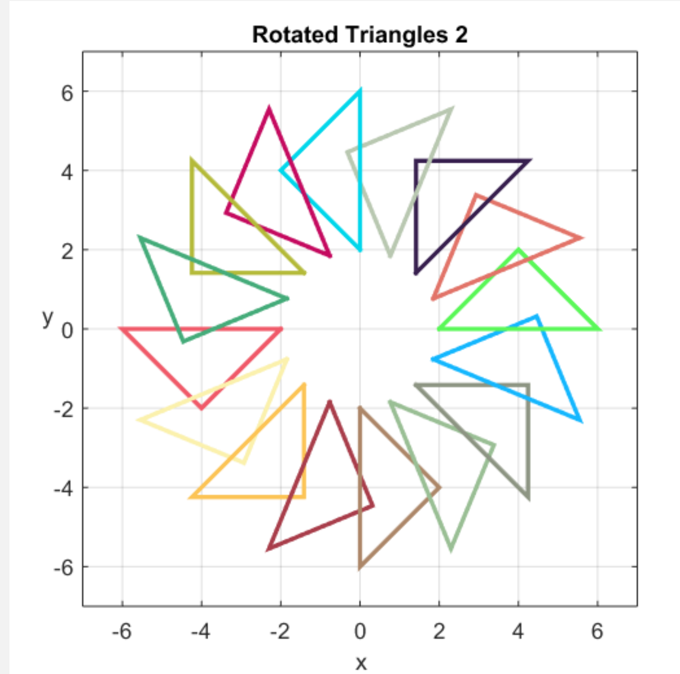


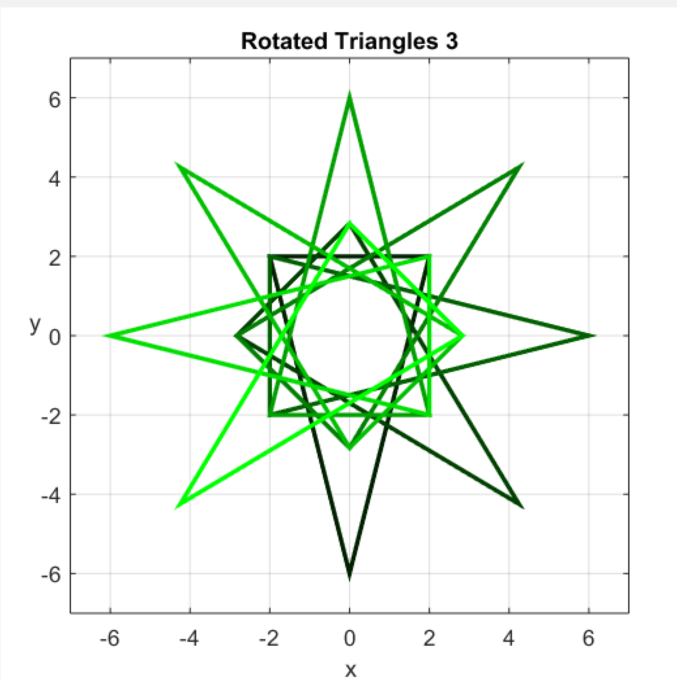
try to do something similar - use any shapes

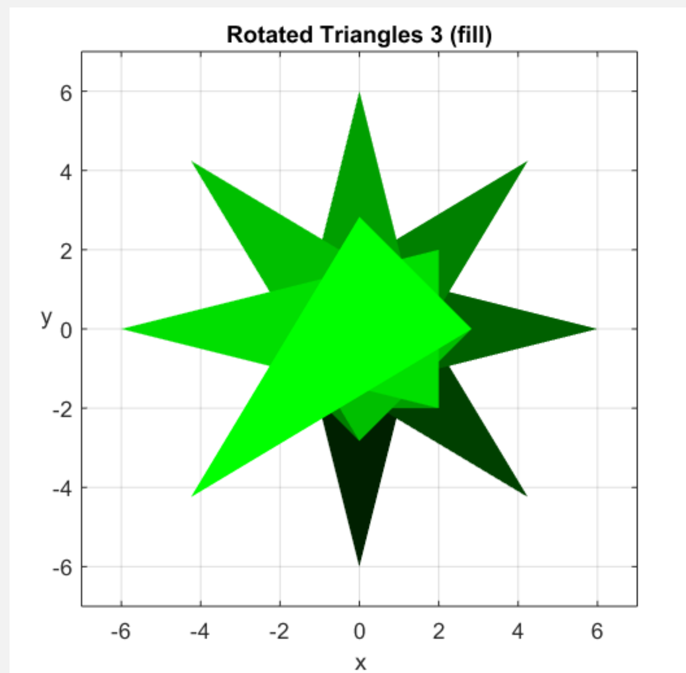


Example 3 Rotated Triangles







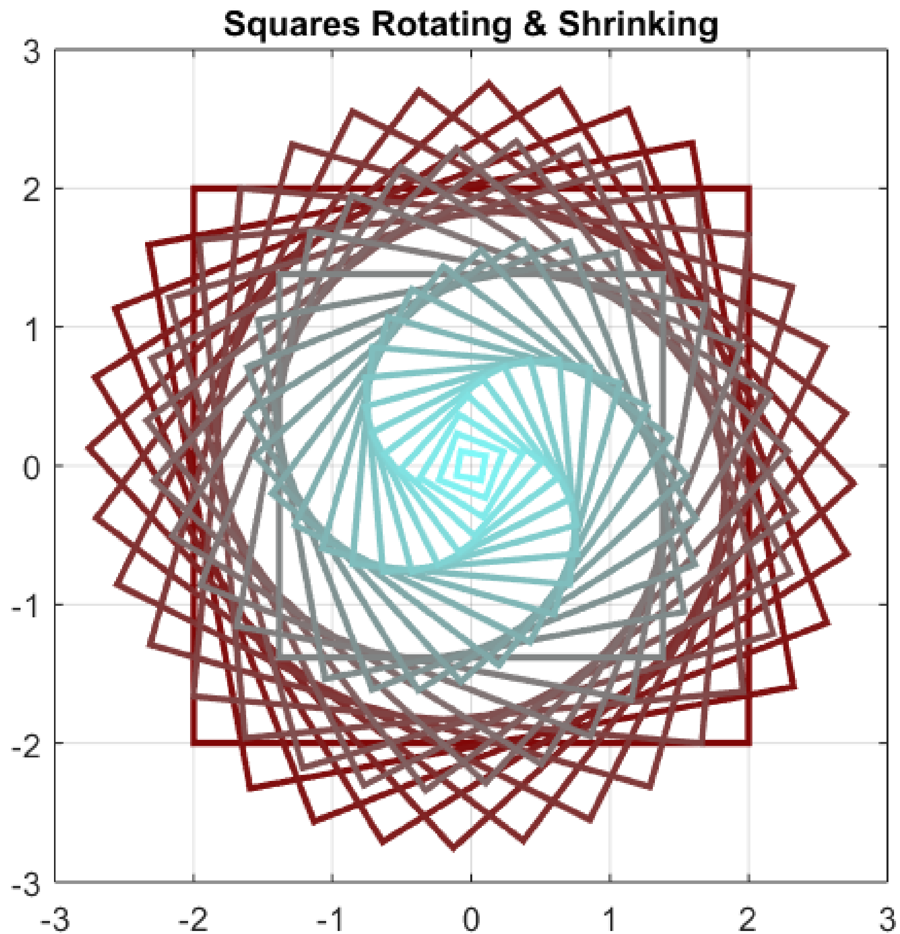


try to do something similar

use any shapes

the matrix form of the rotation equations may be easier to work with

Example 4
Squares



try to do something similar

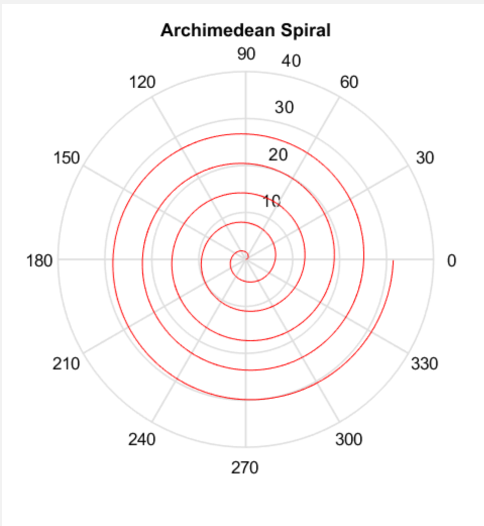
use any shapes

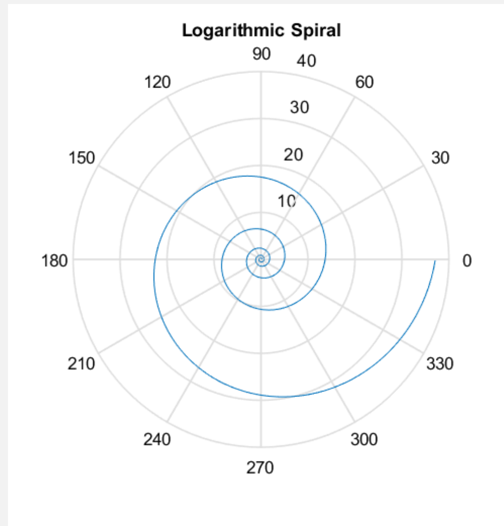
the matrix form of the rotation equations may be easier to work with

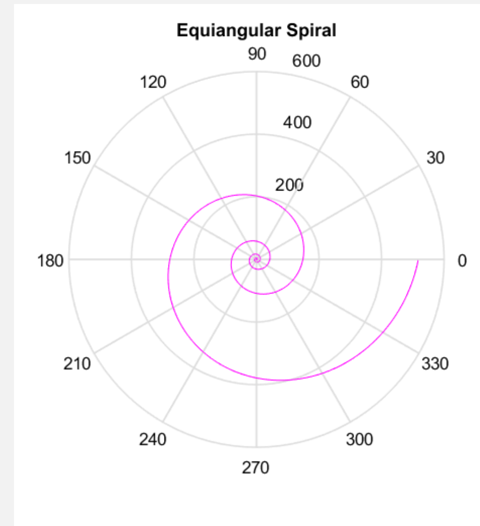
shrinking effect:

in the example the shrinking effect was achieved using the factor $\cos\left(\frac{\pi k}{2m}\right)$ where *k* is the loop counter and *m* is the maximum of *k*

polar(theta, r) creates a plot of the polar coordinates theta (angle) and r (radius)





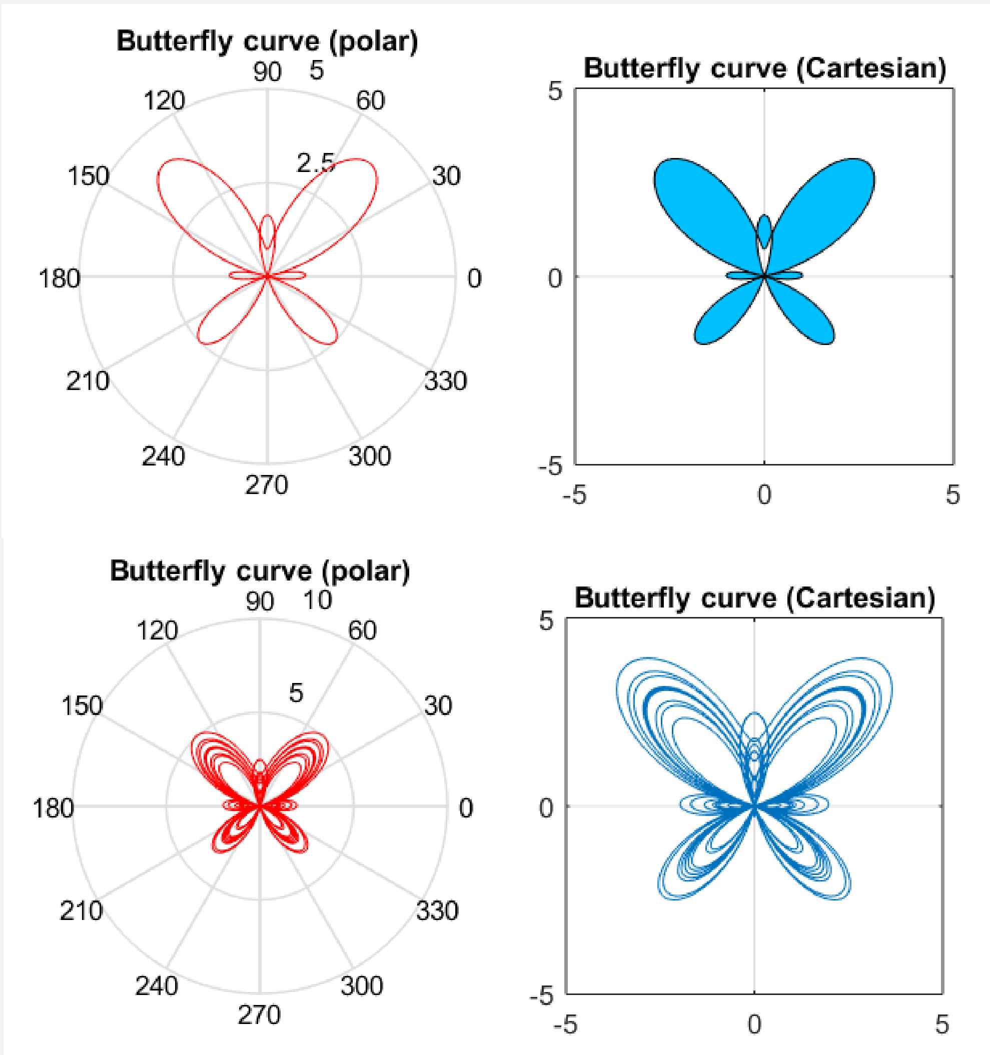



```
t=0:0.01:10*pi;
%r=t;  % Archimedean spiral
%r=exp(t*cot(7*pi/16)); % Equiangular spiral
r=0.25*exp(t/(2*pi)); % Logarithmic
polar(t,r)
```

The polar function does NOT allow the user to change the linewidth or specify RGB colours the same way the plot function does.

Butterfly curve in polar coordinates

$$r(t) = e^{\sin(t)} - 2\cos(4t) + \sin^5\left(\frac{2t-\pi}{24}\right)$$



$t \in [0, 2\pi]$

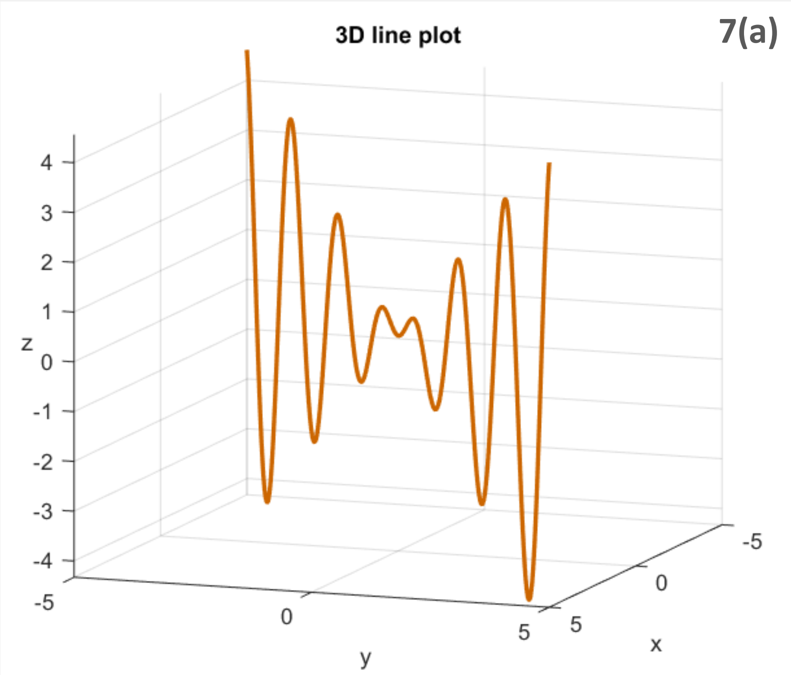
The Cartesian plot is shown using fill

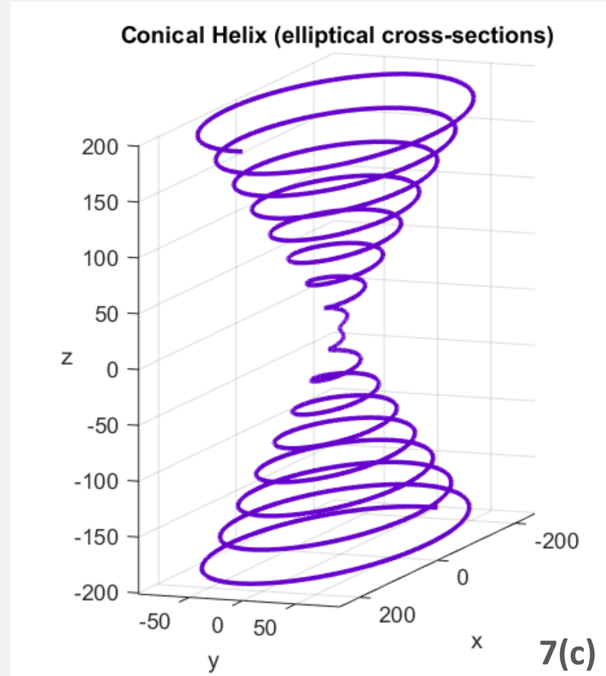
$t \in [0, 50\pi]$

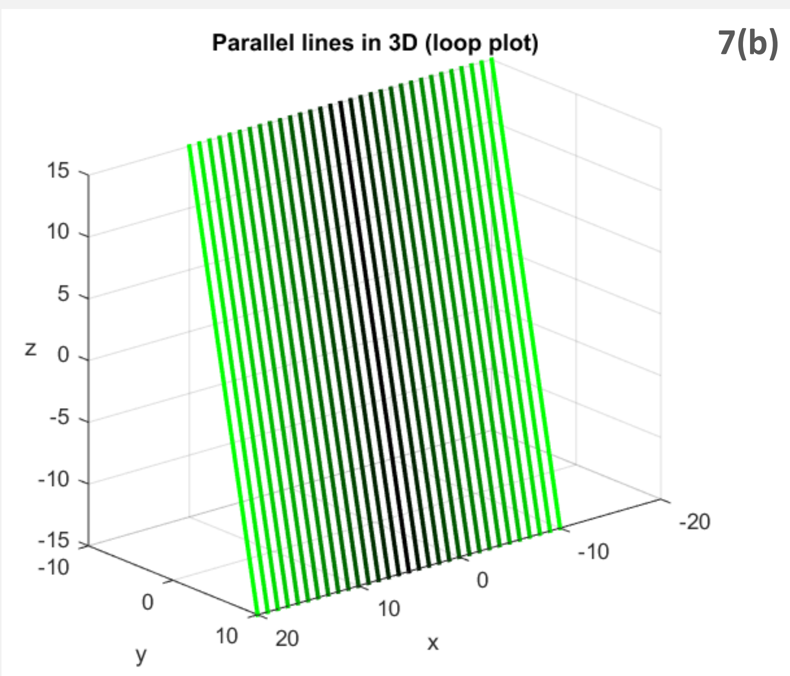
there is no easy way to set the axes when using the polar function

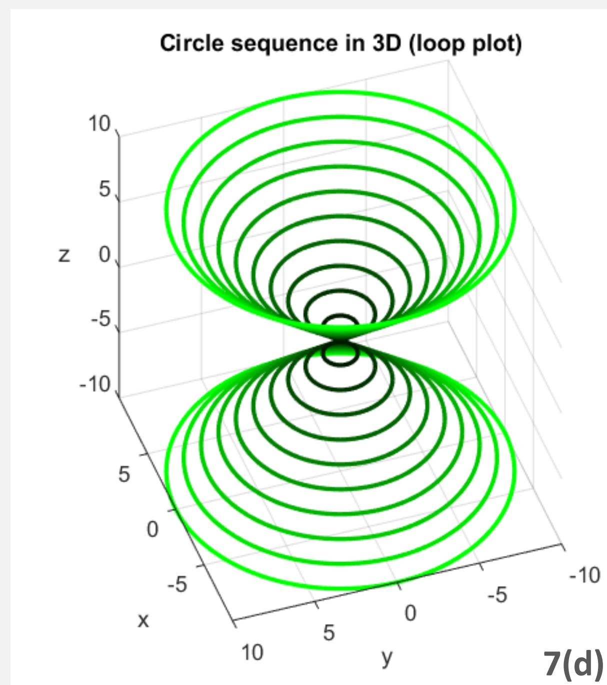
Example 7

3D Line Plots









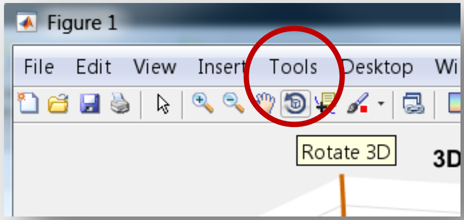
```
% Example 7(a)

close all
view([110,10]) % viewpoint (azimuth & elevation)

% define a parameter
t=-5:0.01:5;

% define x(t), y(t), z(t)
x=t;
y=t;
z=t.*sin(4*t);

% plot3 can be customised the same way as plot
plot3(x,y,z, 'color', [0.8 0.4 0], 'linewidth',2)
title('3D line plot')
xlabel('x'); ylabel('y'); zlabel('z', 'rot',0)
grid on
axis equal
```

try to create something similar to 7(b)

you will need to know the parametric equation of a 3D line

When using the plot3 function, it is often useful to define the *x*, *y* and *z* coordinates in terms of a parameter, e.g. *x*(*t*), *y*(*t*), *z*(*t*).

To see how the **azimuth** and **elevation** angles are defined, click [here](#)
The view can also be changed from the figure window.



```
% Example 7(c) - Conical Helix

close all
view([110,10]) % viewpoint (azimuth & elevation)

% define a parameter
t=-16*pi:0.01:16*pi;

% define x(t), y(t), z(t)
x=5*t.*cos(t);
y=2*t.*sin(t);
z=4*t;

% plot3 can be customised the same way as plot
plot3(x,y,z, 'color', [ 0.4 0 0.8], 'linewidth',2)
title('Conical Helix (elliptical cross-sections)')
xlabel('x'); ylabel('y'); zlabel('z', 'rot',0)
grid on
axis equal
```

The conical helix is a space curve.

A good collection of equations of plane/space curves and surfaces can be found [here]

Experiment with other curves – try your own ideas.

```
% Example 7(d) - Horizontal Circles
close all

view([-109,42]) % set azimuth & elevation

t=-5:0.01:5;

hold on
for k=-10:10
x=k*sin(t);
y=k*cos(t);
z=k+0*x; % z cannot be a scalar, must have the same size as x and y
plot3(x,y,z,'color', [0 abs(k)/10,0], 'linewidth', 2)
end

title('Circle sequence in 3D (loop plot)')
xlabel('x'); ylabel('y'); zlabel('z', 'rot',0)
grid on
axis equal
```

Try to replace the circle with another shape – use your imagination.

Plotting a 3D surface in the form $z = f(x, y)$

- define the range of x and y coordinates

```
X = -3:0.1:3;
Y = -3:0.1:3;
```

- create an *xy* grid using the **meshgrid** function

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

- define the surface z=f(x,y)

```
z=1-x.^2.*sin(y);
```

- use the **surf** or **mesh** function to create the 3D surface plot

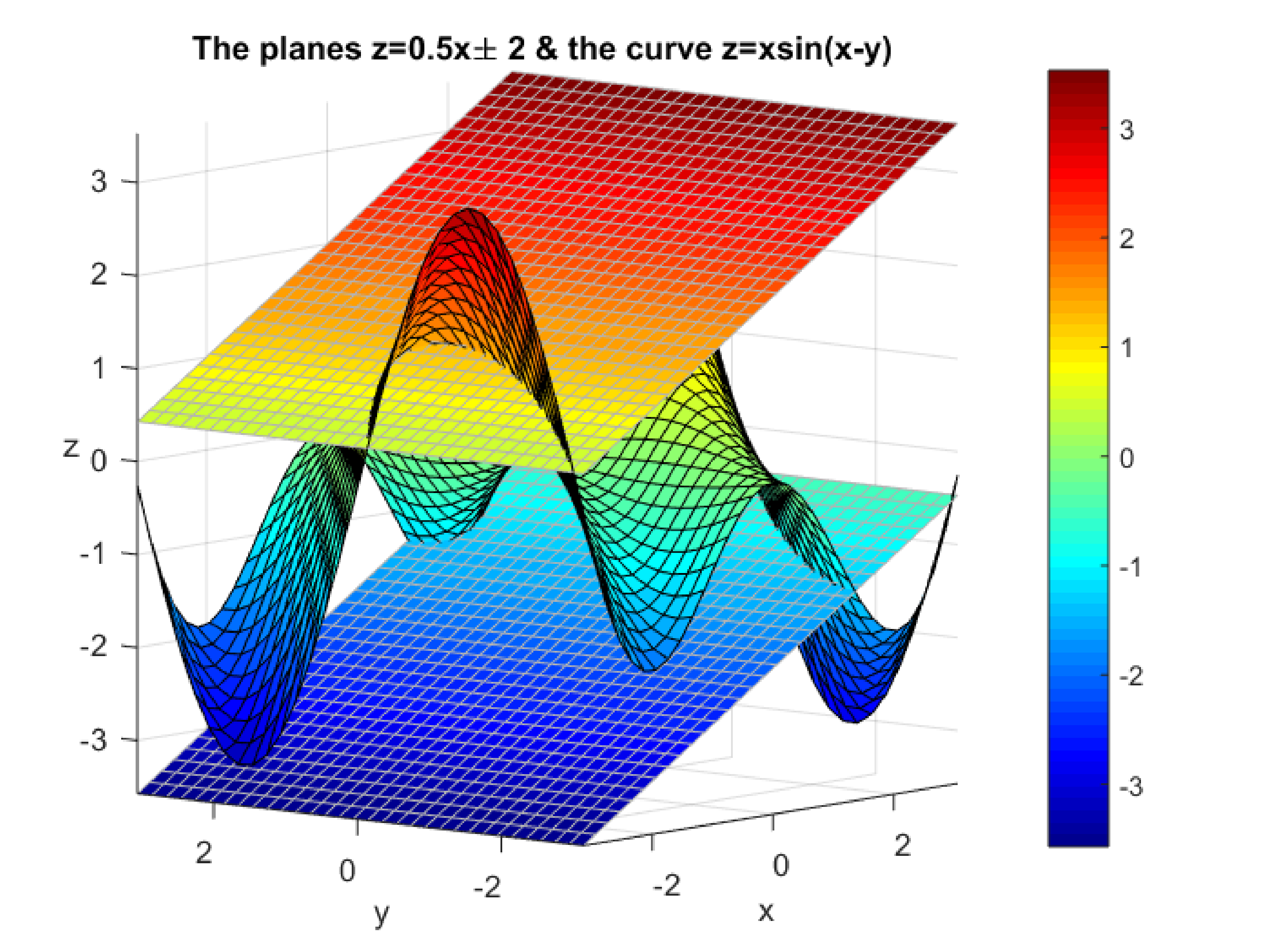
```
surf(x,y,z)
mesh(x,y,z)
```

A few useful links:

A detailed handout on 3D graphics can be found [here](#)

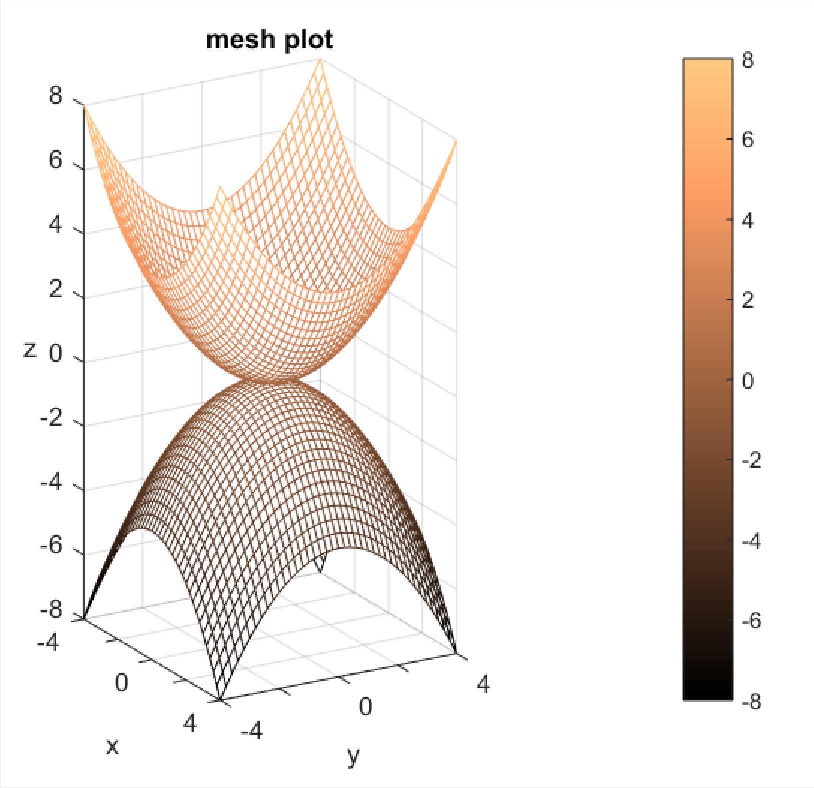
http://uk.mathworks.com/help/matlab/surface-and-mesh-plots-1.html

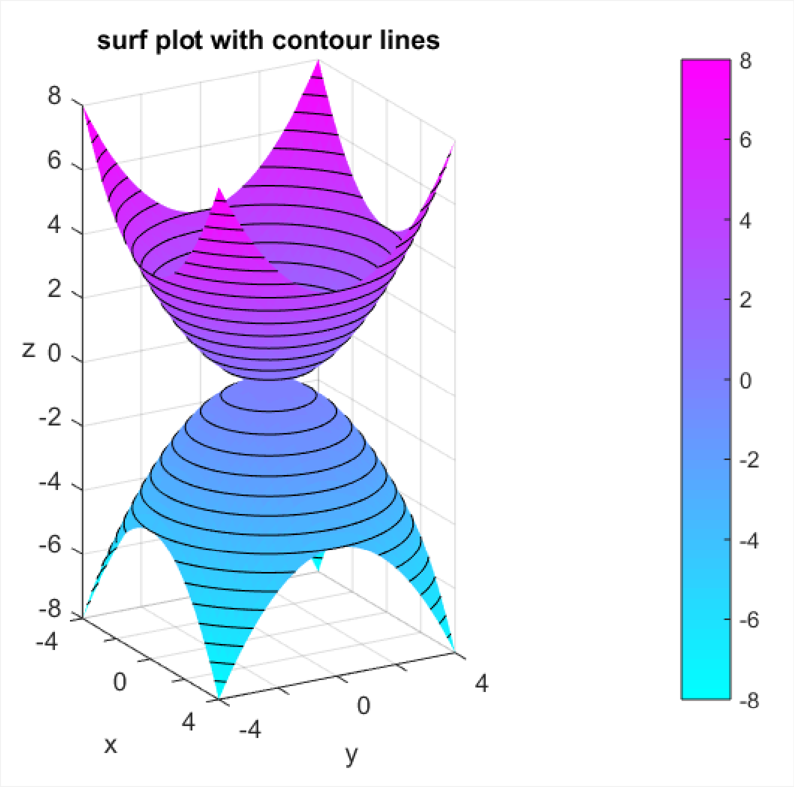
http://uk.mathworks.com/help/matlab/ref/contour3.html


The planes z=0.5x± 2 & the curve z=xsin(x-y)
z
3
2
1
0
-1
-2
-3
y
2
0
-2
x
-2
0
2
3
2
1
0
-1
-2
-3

Example 8(a)

```
% Example 8(a)

X = -pi:.2:pi;
Y = -pi:.2:pi;
[x,y] = meshgrid (X,Y);

close all, hold on, grid on
view([-50,8])

% define surfaces in the form z=f(x,y)
z1=x.*sin(x-y); % trigonometric surface
z2=0.5*x-2; % plane 1
z3=0.5*x+2; % plane 2

surf(x,y,z1);
surf(x,y,z2, 'edgecolor', [0.7 0.7 0.7]);
surf(x,y,z3, 'edgecolor', [0.7 0.7 0.7]);

colormap(jet)
axis equal tight
xlabel('x');
ylabel('y');
zlabel('z', 'rot',0)
title('The planes z=0.5x\pm 2 & the curve z=xsin(x-y) ')
```





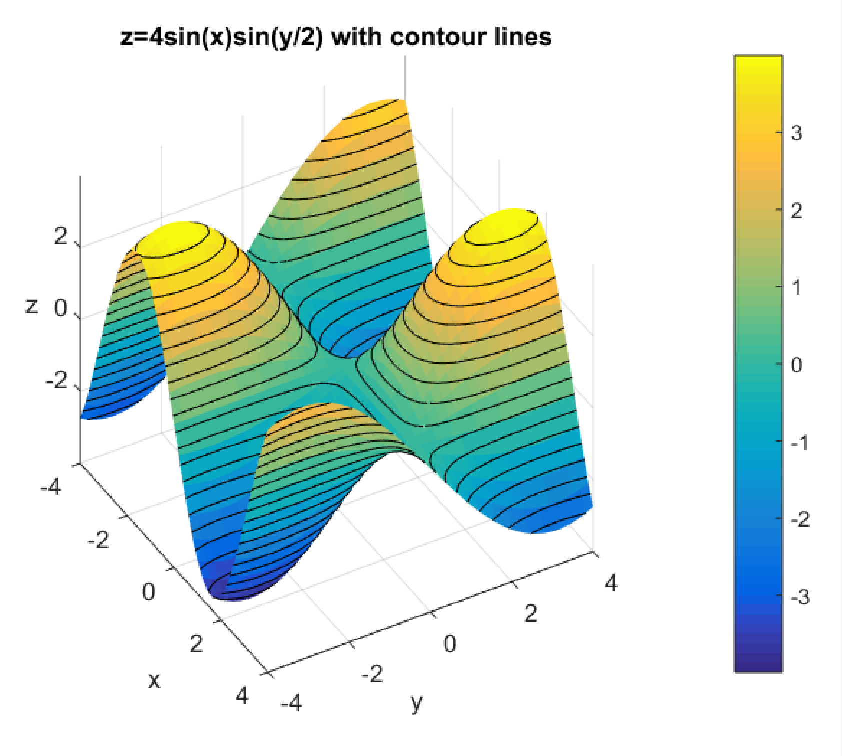
Information on colormaps:

http://uk.mathworks.com/help/matlab/ref/colormap.html

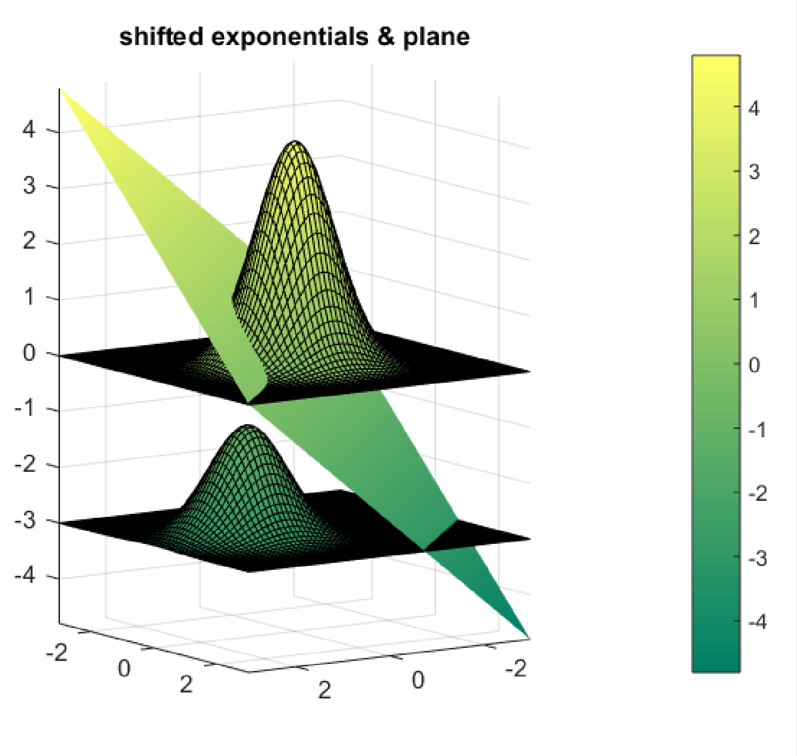
Example 8(b)

```
% Example 8(b)

 X = -4:.2:4; Y = -4:.2:4;
[x,y] = meshgrid (X,Y);

close all, hold on, grid on, view([60,20])

% paraboloid equation
z=(x/2).^2+(y/2).^2;
% using mesh
mesh(x,y,z)
mesh(x,y,-z)
colormap(copper)

% % using surf and contour3
% surf(x,y,z,'edgecolor','none')
% % turn the paraboloid upside-down
% surf(x,y,-z,'edgecolor','none')
% % add 15 black contour lines to each surface
% contour3(x,y,z,15,'k')
% contour3(x,y,-z,15,'k')
% colormap(cool)

xlabel('x'); ylabel('y'); zlabel('z', 'rot',0)
title('mesh plot'), axis equal tight
```

Example 8(c)



Example 8(d)

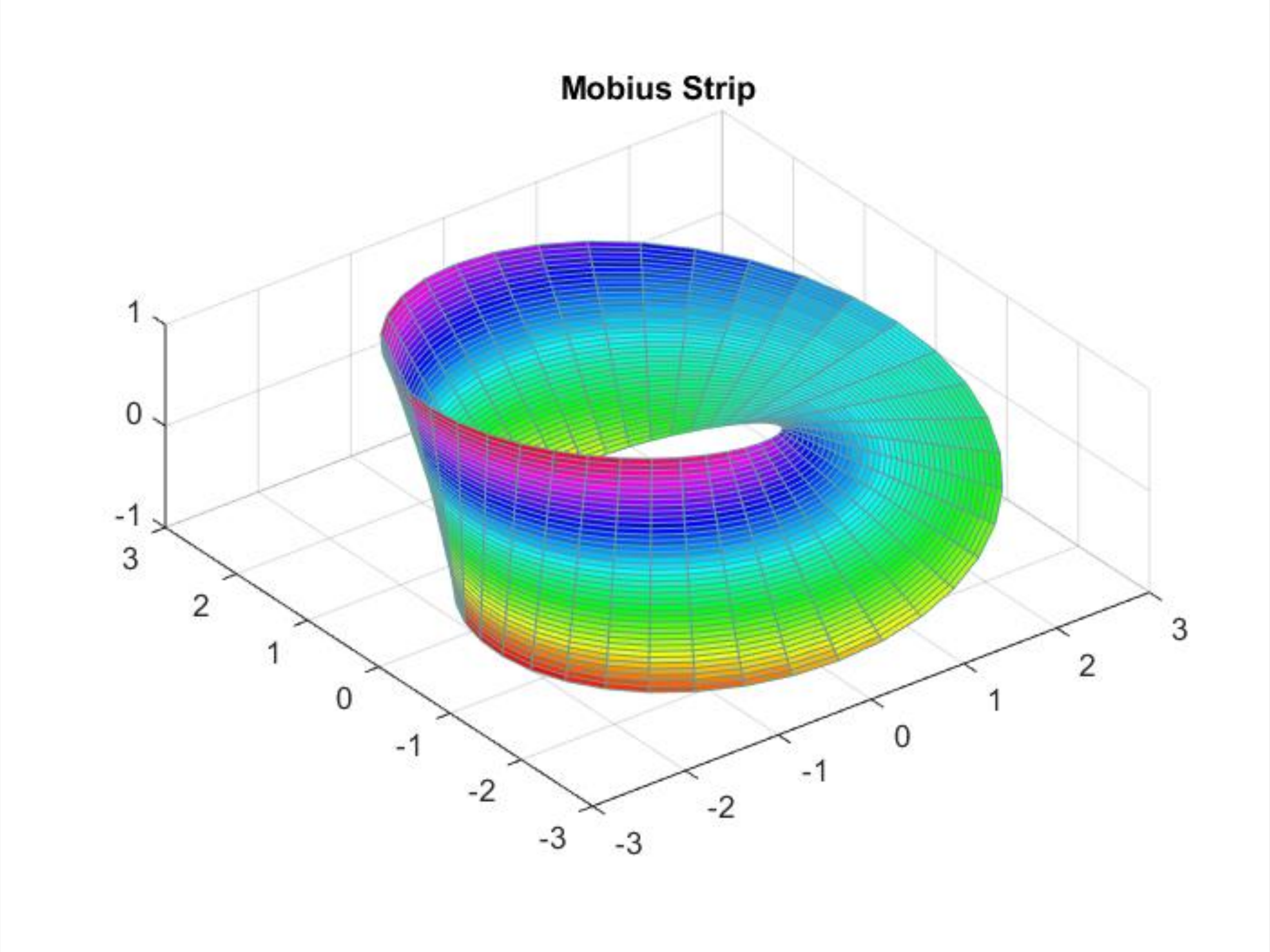


```
% Example 8(c)
% using the default colormap (parula)

X = -4:.2:4;
Y = -4:.2:4;
[x,y] = meshgrid (X,Y);

close all, hold on, grid on
view([60,40])

z=4*sin(y/2).*sin(x);

% plot surface elements without borders
surf(x,y,z, 'edgecolor', 'none')

% add 20 black contour lines
contour3(x,y,z,20,'k')

xlabel('x'); ylabel('y'); zlabel('z', 'rot',0)
title('z=4sin(x)sin(y/2) with contour lines')
axis equal tight
```

Experiment with different $z(x, y)$ functions!

```
% Example 8(d)

 X = -3:.1:3; Y = -3:.1:3;
[x,y] = meshgrid (X,Y);

close all, hold on, grid on
view([146,10])

% define the plane
z1=0.8*(x-y);
surf(x,y,z1,'edgecolor', 'none')

% upper exponential
z2=4*exp(-(x.^2+y.^2));
surf(x,y,z2)

% shifted shorter exponential shape below the tall one
z3=2*exp(-((x-1).^2+y.^2))-3;
surf(x,y,z3)

colormap(summer)
title('shifted exponentials & plane')
axis equal tight
```

Example 9



```
% Example 9

% Mobius strip parametric equation
% S(u,v)= [x(u,v), y(u,v), z(u,v)]
% S(u,v) = [(2+v*cos(u)/2)*cos(u), (2+v*cos(u)/2)*sin(u), v*sin(u)/2]

close all, view([-40,35])

U = linspace(0,2*pi,40); % parameter 1
% using linspace to make sure that 2pi is included

V = -1:0.05:1;              % parameter 2
[u,v] = meshgrid(U,V);      % create uv grid

% define x, y and z coordinates in terms of u and v
x = (2+v.*cos(u/2)).*cos(u); % x(u,v)
y = (2+v.*cos(u/2)).*sin(u); % y(u,v)
z = v.*sin(u/2); % z(u,v)

surf(x,y,z, 'edgecolor', [0.4 0.6 0.6]);
colormap(hsv)
title('Mobius Strip');
axis equal
axis([-3 3 -3 3 -1 1]);
```

Example 9

Parametric Surface

Möbius strip