Lec 08: Graphics in MATLAB

Anonymous Functions

Anonymous Functions

Mathematical functions such as

$$f_1(x) = \cos x \sin\big(\cos(\tan x)\big),$$
$$f_2(\theta) = (\cos 3\theta + 2\cos 2\theta)^2,$$
$$f_3(x, y) = \frac{\sin(x+y)}{1 + x^2 + y^2},$$

can be defined in MATLAB using *anonymous functions*:

```
f1 = @(x) cos(x).*sin(cos(tan(x)));
f2 = @(th) ( cos(3*th) + 2*cos(2*th) ).^2;
f3 = @(x, y) sin(x + y)./(1 + x.^2 + y.^2);
```

Anonymous Functions – Syntax

Take a closer look at one of them.

```
f1 = @(x) cos(x).*sin(cos(tan(x)));
```

- `f1` : the function name or the *function handle*
- `@` : marks the beginning of an anonymous function
- `(x)` : denotes the function (input) argument
- `cos(x).*sin(cos(tan(x)))` : MATLAB expression defining $f_1(x)$

Examples

Expressions in function definitions can get very complicated. For example,

```
h1 = @(x) [2*x, sin(x)];
h2 = @(x) [2*x, sin(x); 5*x, cos(x); 10*x, tan(x)];
r = @(a,b,m,n) a + (b-a)*rand(m,n);
```

Exercise: Different Ways of Defining a Function

The function

$$f_4(\theta; c_1, c_2, k_1, k_2) = (c_1 \cos k_1\theta + c_2 \cos k_2\theta)^2$$

can be defined in two different ways:

```
f4s = @(th,c1,c2,k1,k2) c1*cos(k1*th) + c2*cos(k2*th)
f4v = @(th,c,k) c(1)*cos(k(1)*th) + c(2)*cos(k(2)*th)
```

Question

Use `f4s` and `f4v` to define yet another anonymous functions for

- $g(\theta) = 3\cos(2\theta) - 2\cos(3\theta)$
- $h(\theta) = 3\cos(\theta/7) + \cos(\theta)$

Exercise: Understanding Anonymous Functions

Type in the following statements in MATLAB:

```
f1 = @(x) cos(x).*sin(cos(tan(x)));
f2 = @(th) ( cos(3*th) + 2*cos(2*th) ).^2;
x1 = 5; y1 = f1(x1)
x2 = [5:-2:1]; y2 = f1(x2)
TH = diag(0:pi/2:2*pi); R = f2(TH)
```

Question

1. What are the types of the input and output variables?
   - `x1` and `y1`
   - `x2` and `y2`
   - `TH` and `R`
2. Which of the three outputs will be affected if elementwise operations were not used in the definition of `f1` and `f2`?

2-D Graphics

The `PLOT` Function

To draw a curve in MATLAB:

- Construct a pair of n-vectors `x` and `y` corresponding to the set of data points $\{(x_i, y_i) \mid i = 1, 2, \ldots, n\}$ which are to appear on the curve in that order.
- Then type `plot(x, y)`.

For example:

```
x = linspace(0, 2*pi, 101);
y = sin(x);
plot(x, y)                          % or simply plot(x, sin(x))
```

or

```
f = @(x) 1 + sin(2*x) + cos(4*x);       % anonymous function
x = linspace(0, 2*pi, 101);
plot(x, f(x))
```

Example: Wiggly Curve

First, run the following script.

```
1 f1 = @(x) cos(x).*sin(cos(tan(x)));
2 x = 2*pi*[0:.0001:1];   % or  x = linspace(0, 2*pi, 10001);
3 plot(x, f1(x))
4 shg
```

Play Around!

Observe what happens after applying the following modifications one by one.

- Change line 3 into `plot(x, f1(x), 'r')`.

Example: Wiggly Curve

First, run the following script.

```
1 f1 = @(x) cos(x).*sin(cos(tan(x)));
2 x = 2*pi*[0:.0001:1];    % or  x = linspace(0, 2*pi, 10001);
3 plot(x, f1(x))
4 shg
```

Play Around!

Observe what happens after applying the following modifications one by one.

- Change line 3 into `plot(x, f1(x), 'r')`.
- Change line 3 into `plot(x, f1(x), 'r--')`.

Example: Wiggly Curve

First, run the following script.

```
1 f1 = @(x) cos(x).*sin(cos(tan(x)));
2 x = 2*pi*[0:.0001:1];    % or  x = linspace(0, 2*pi, 10001);
3 plot(x, f1(x))
4 shg
```

Play Around!

Observe what happens after applying the following modifications one by one.

- Change line 3 into `plot(x, f1(x), 'r')`.
- Change line 3 into `plot(x, f1(x), 'r--')`.
- After line 3, add `axis equal, axis tight`.

Example: Wiggly Curve

First, run the following script.

```
1 f1 = @(x) cos(x).*sin(cos(tan(x)));
2 x = 2*pi*[0:.0001:1];    % or  x = linspace(0, 2*pi, 10001);
3 plot(x, f1(x))
4 shg
```

Play Around!

Observe what happens after applying the following modifications one by one.

- Change line 3 into `plot(x, f1(x), 'r')`.
- Change line 3 into `plot(x, f1(x), 'r--')`.
- After line 3, add `axis equal, axis tight`.
- Then add `text(4.6, -0.3, 'very wiggly')`.

Example: Wiggly Curve

First, run the following script.

```
1 f1 = @(x) cos(x).*sin(cos(tan(x)));
2 x = 2*pi*[0:.0001:1];   % or  x = linspace(0, 2*pi, 10001);
3 plot(x, f1(x))
4 shg
```

Play Around!

Observe what happens after applying the following modifications one by one.

- Change line 3 into `plot(x, f1(x), ’r’)`.
- Change line 3 into `plot(x, f1(x), ’r--’)`.
- After line 3, add `axis equal, axis tight`.
- Then add `text(4.6, -0.3, ’very wiggly’)`.
- Then add `xlabel(’x axis’), ylabel(’y axis’), title(’A wiggly curve’)`.

Note: Line Properties

- To specify line properties such as colors, markers, and styles:

```
plot(x, y, '--')                        % dashed line
plot(x, y, 'g:')                        % dotted line in green
plot(x, y, '.', 'MarkerSize', 3)        % adjust marker size
plot(x, y, 'b-', 'LineWidth', 5)        % adjust line width
```

Colors

b	blue
g	green
r	red
c	cyan
m	magenta
y	yellow
k	black
w	white

Markers

.	point
o	circle
x	x-mark
+	plus
*	star
s	square
d	diamond

Line Styles

-	solid
:	dotted
-.	dashdot
--	dashed

Note: Labels and Saving

- To label the axes and the entire plot, add the following after `plot` statement:

```
xlabel('x axis')
ylabel('y axis')
title('my awesome graph')
```

- Save figures using `print` function. Multiple formats are supported.

```
print -dpdf 'wiggly'
% or print('-dpdf', 'wiggly')                             [pdf]

print -djpeg 'wiggly'
% or print('-djpeg', 'wiggly')                           [jpeg]

print -deps 'wiggly'
% or print('-deps', 'wiggly')                             [eps]
```

Note: Drawing Multiple Figures

- To plot multiple curves:

```
plot(x1, y1, x2, y2, x3, y3, ...)
```

- To create a legend, add

```
legend('first graph', 'second graph', 'third graph', ...)
```

Note: Miscellaneous Commands

- `shg`: (show graph) to bring Figure Window to the front
- `figure`: to open a new blank figure window
- `clf`: (clear figure) to clear previously drawn figures
- `axis equal`: to put axes in equal scaling
- `axis tight`: to remove margins around graphs
- `axis image`: same as `axis equal` and `axis tight`
- `grid on`: to put light gray grid lines

Exercise

Question

Do the following:

- Define $f(x) = x^3 + x$ as an anonymous function.
- Find f' and f'' and define them as anonymous functions.
- Plot all three functions in one figure in the interval $[-1, 1]$.
- Include labels and title in your plot.
- Add legend to the graph.
- Save the graph as a pdf file.

Multiple Figures – Stacking

To draw multiple curves in one plotting window as in Figure 1:

- One liner:

```
plot(x1, y1, x2, y2, x3, y3)
```

- Or, add curves one at a time using `hold` command.

```
plot(x1, y1)
hold on
plot(x2, y2)
plot(x3, y3)
```

  - `hold on`: holds the current plot for further overlaying
  - `hold off`: turns the *hold* off

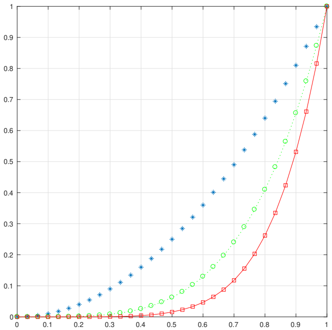


Figure 1: Multiple curves in one plot

Multiple Figures – Subplots

To plot multiple curves separately and arrange them as in Figure 2:

```
subplot(1,3,1)
plot(x1, y1)
subplot(1,3,2)
plot(x2, y2)
subplot(1,3,3)
plot(x3, y3)
```

`subplot(m,n,p)`:

- `m`, `n`: determine grid dimensions
- `p`: determines grid is to be used

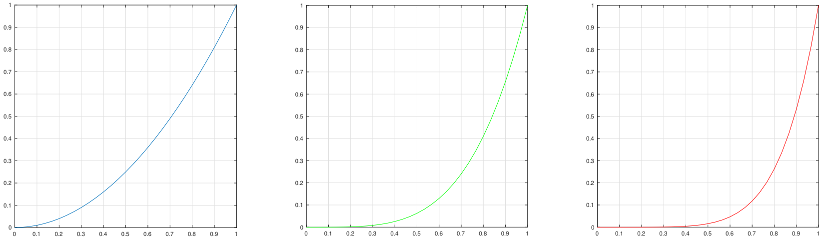


Figure 2: Multiple plots in 1×3 grids

Exercise: Multiple Figures

Do It Yourself

Generate Figures 1 and 2.

- Common: Generating sample points

```
x = linspace(0, 1, 101);
y1 = x.^2; y2 = x.^4; y3 = x.^6;
```

- Figure 1:

```
hold off
plot(x, y1, '*')
hold on
plot(x, y2, 'g:o')
plot(x, y3, 'r-s')
```

- Figure 2:

```
subplot(1, 3, 1)
plot(x, y1)
subplot(1, 3, 2)
plot(x, y2, 'g')
subplot(1, 3, 3)
plot(x, y3, 'r')
```

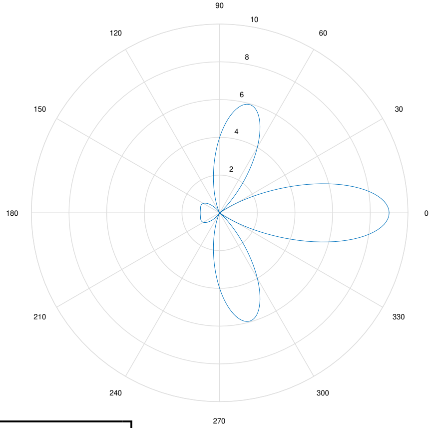
The POLAR Function

To draw the polar curve $r = f(\theta)$, for $\theta \in [a, b]$:

- Grab n sample points $\{(\theta_i, r_i) \mid r_i = f(\theta_i),\ 1 \le i \le n\}$ on the curve and form vectors `th` and `r`.
- Then type `polar(th, r)`.
- For example, to plot

$$r = f_2(\theta) = (\cos 3\theta + 2\cos 2\theta)^2,$$

for $\theta \in [0, 2\pi]$:




```
th = linspace(0, 2*pi, 361);
f2 = @(th) (cos(3*th) + 2*cos(2*th)).^2;
polar(th, f2(th));
```

Exercise: Drawing Polar Curves

Question

1. Draw the graph of two-petal leaf given by
$$r = f(\theta) = 1 + \sin(2\theta)\,, \quad \theta \in [0, 2\pi]\,.$$
2. Draw the graphs of
$$r = f(\theta - \pi/4)\,, \quad r = f(\theta - \pi/2)\,, \quad r = f(\theta - 3\pi/4)$$
on the same plotting window.
3. Does your figure make sense?

3-D Graphics

Curves and the `PLOT3` Function

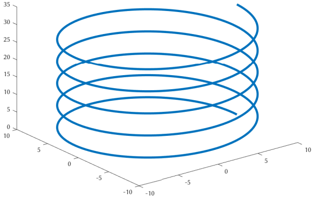
Curves in $\mathbb{R}^3$ are plotted in an analogous fashion.

- Grab n sample points $\{(x_i, y_i, z_i) \mid i = 1, 2, \ldots, n\}$ on the curve and form vectors `x`, `y`, and `z`.
- Then type `plot3(x, y, z)`.
- For example, to plot the helix given by the parametrized equation

$$\mathbf{r}(t) = \langle 10\cos(t), 10\sin(t), t\rangle,$$

for $t \in [0, 10\pi]$:

```
t = linspace(0, 10*pi, 1000);
plot3(10*cos(t), 10*sin(t), t);
```



Exercise: Corkscrew

Question

Modify the code to generate a corkscrew by putting the helix outside of an upside down cone.

Hint: Use $\mathbf{r}(t) = \langle t\cos(t), t\sin(t), t\rangle$.

Surfaces and the SURF Function

To plot the surface of $z = f(x, y)$ on $R = [a, b] \times [c, d]$:

- Collect samples points on the intervals $[a, b]$ and $[c, d]$ and form vectors `x` and `y`.
- Based on `x` and `y`, generate grid points $\{(x_i, y_j) \mid i = 1, 2, \ldots, m, j = 1, 2, \ldots, n\}$ on the domain R and separate coordinates into matrices `X` and `Y` using `meshgrid`.
- Type `surf(X, Y, f(X,Y))`

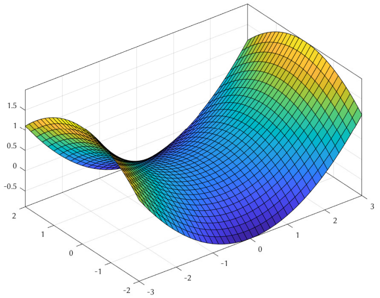
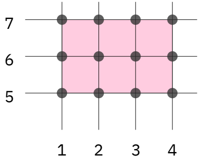
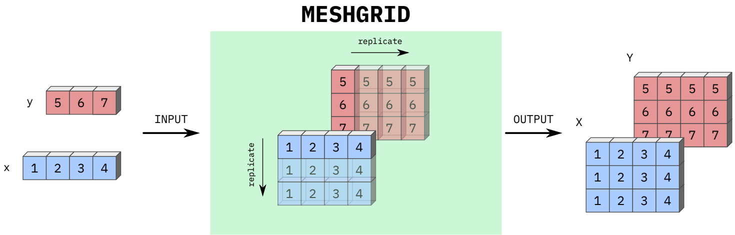


Figure 3: Graph of $z = \frac{2}{9}(x^2 - y^2)$ on $[-3, 3] \times [-2, 2]$

Note: How MESHGRID Work

```
>> x = [1 2 3 4]; y = [5 6 7];
>> [X, Y] = meshgrid(x,y)
X =
      1      2      3      4
      1      2      3      4
      1      2      3      4
Y =
      5      5      5      5
      6      6      6      6
      7      7      7      7
```





Example: Saddle

Question

Plot the saddle parametrized by

$$\frac{z}{c} = \frac{x^2}{a^2} - \frac{y^2}{b^2}$$

for your choice of a, b, and c.

```
x = linspace(-3, 3, 13);
y = linspace(-2, 2, 9);
[X, Y] = meshgrid(x, y);
a = 1.5; b = 1.5; c = .5;
g2 = @(x,y) c*( x.^2 /a^2 - y.^2 /b^2);
surf(X, Y, g2(X,Y))
axis equal, box on
```

Figure 3 was generated using this code.

Example: Oblate Spheroid

The figure for Problem 5 of Homework 1 was generated by the following code.[1]

```
a = 1; b = 1.35; c = 1;
nr_th = 41; nr_ph = 31;
x = @(th, ph) a*cos(th).*sin(ph);
y = @(th, ph) b*sin(th).*sin(ph);
z = @(th, ph) c*cos(ph);
th = linspace(0, 2*pi, nr_th);
ph = linspace(0, pi, nr_ph);
[T, P] = meshgrid(th, ph);
surf(x(T,P), y(T,P), z(T,P))
colormap(winter)
axis equal, axis off, box off
```

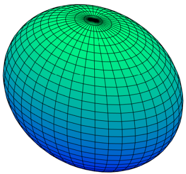


Figure 4: Oblate spheroid.

[1] The code is originally from **LM** (`sphere.m`); some parameters and the color specs were modified.