MATLAB

Lecture 3

Relational Operators

- MATLAB uses *mostly* standard relational operators

equal	==
Not equal	~=
greater than	>
less than	<
greater or equal	>=
less or equal	<=

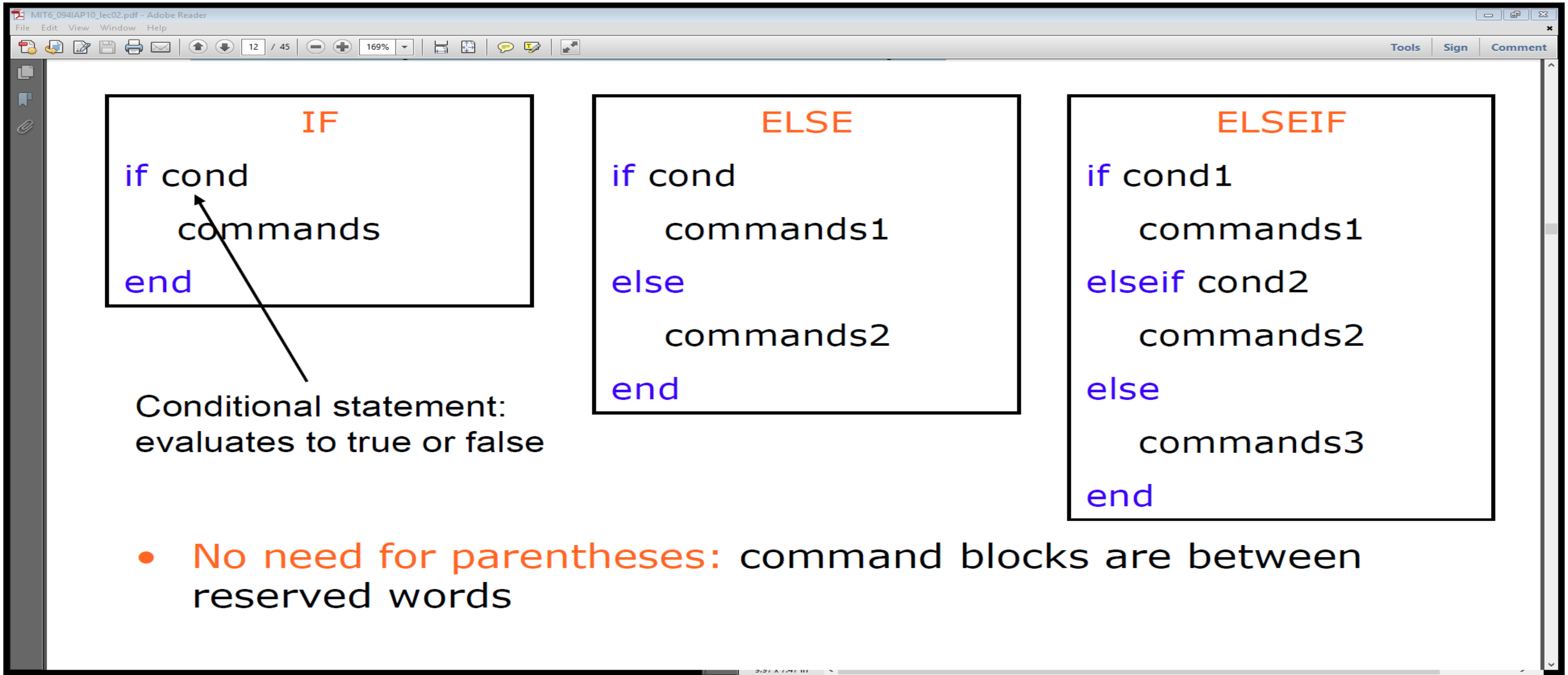
Relational Operators

<table>
<tr><th>Logical operators</th><th>elementwise</th><th>short-circuit (scalars)</th></tr>
<tr><td>And</td><td>&</td><td>&&</td></tr>
<tr><td>Or</td><td>|</td><td>| |</td></tr>
<tr><td><b>Not</b></td><td>~</td><td></td></tr>
<tr><td>Xor</td><td>xor</td><td></td></tr>
<tr><td>All true</td><td>all</td><td></td></tr>
<tr><td>Any true</td><td>any</td><td></td></tr>
</table>

- Boolean values: zero is false, nonzero is true
- See **help .**for a detailed list of operators

if/else/elseif

- Conditional Statements
- Basic flow-control, common to all languages
- MATLAB syntax is somewhat unique


MIT6_094IAP10_lec02.pdf - Adobe Reader
File Edit View Window Help
12 / 45
169%
Tools
Sign
Comment

IF

```
if cond
    commands
end
```

Conditional statement: evaluates to true or false

ELSE

```
if cond
    commands1
else
    commands2
end
```

ELSEIF

```
if cond1
    commands1
elseif cond2
    commands2
else
    commands3
end
```

- No need for parentheses: command blocks are between reserved words

Conditional Statements

- Matlab program :
- lane
- 1 ....
- 2 ....
- ....
- 100 **if** cond
- 101 command 1
- ....
- 199 command 99
- 200 **end**
- 201
- 202
- ....

if cond is **true** we doing commands 1-99, then go to lane 201

if it **is not true** we are not doing any commands 1-99 and going directly to lane 201 from lane 100

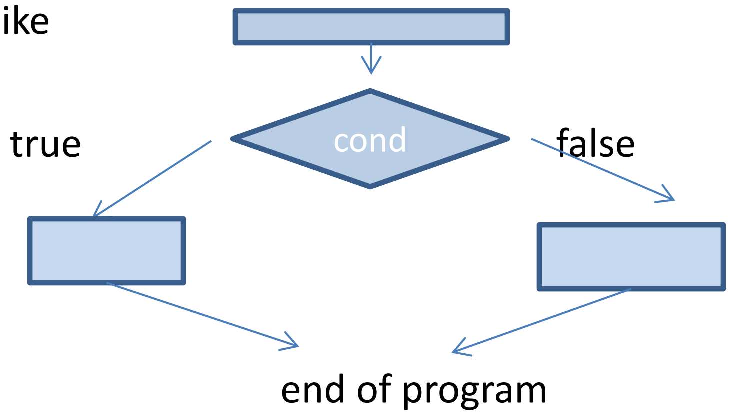
Conditional Statements

- Matlab program :
- lane
- 1
- 2
- ....
- 100 **if** cond
- 101 command 1
- ....
- 199 command 99
- 200 **end**
- 201
- 202
- ....

if cond is **true** we doing commands 1-99, then go to lane 201

if it **is not true** we are not doing any commands 1-99 and going directly to lane 201 from lane 100

- Command 99 end of program **break**
- Then our program look like



Conditional Statements

```
a=5;
b=10;

        if b>a
            disp(' true')
            break
        end

disp ('false')
break

z=10
k=20
```

for

- **For** loops: use for a known number of iterations

- MATLAB syntax: loop variable
- **for** n=1:100
- commands
- **end** command block
  - The loop variable Is defined as a vector
  - Is a scalar within the command block
  - Does not have to have consecutive values (but it's usually cleaner if they're consecutive)
- The command block
  - Anything between the **for** line and the **end** Loop

for

- Example:

```
• for n=1:10
•     disp n
•     disp('Hello Word!')
• end
```

while

- The **while** is like a more general for loop:
  - Don't need to know number of iterations

```
While   cond
        commands
end
```

- The command block will execute while the conditional expression is true
- Beware of infinite loops!

while

```
a=5;
b=10;

        while b>a
              disp(' true')
              a=a+1;
        end
```

  - This program will print ‘true’ 5 times

while

```
• a=5;
• b=10;
•
•           while b>a
•                  disp(' true')
•           end
```

– This program will print ‘true’ infinitive number of times

while

```
end
```

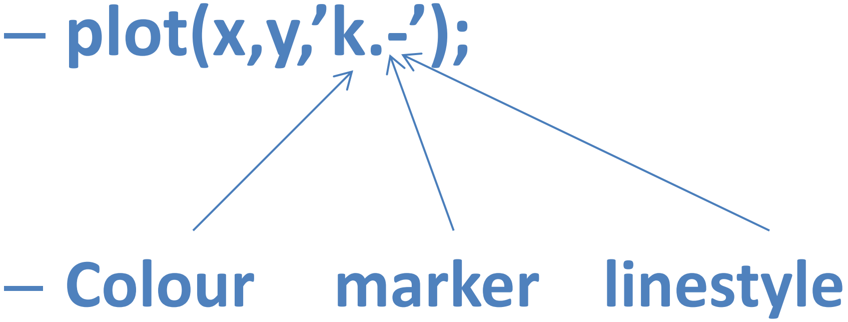
- Example find fixed point of $x_{n+1} = \frac{x_n}{2} + \frac{3}{2x_n}$ (they are $x = \pm\sqrt{3}$

```
xold = 2; xnew = 1;
while abs(xnew-xold) > 1e-5
    xold = xnew;
    xnew = xnew/2+3/(2*xnew);
end
```

This will produce approximation to $\sqrt{3}$.

Plot Options

- Can change the line colour, marker style, and line style by adding a string argument

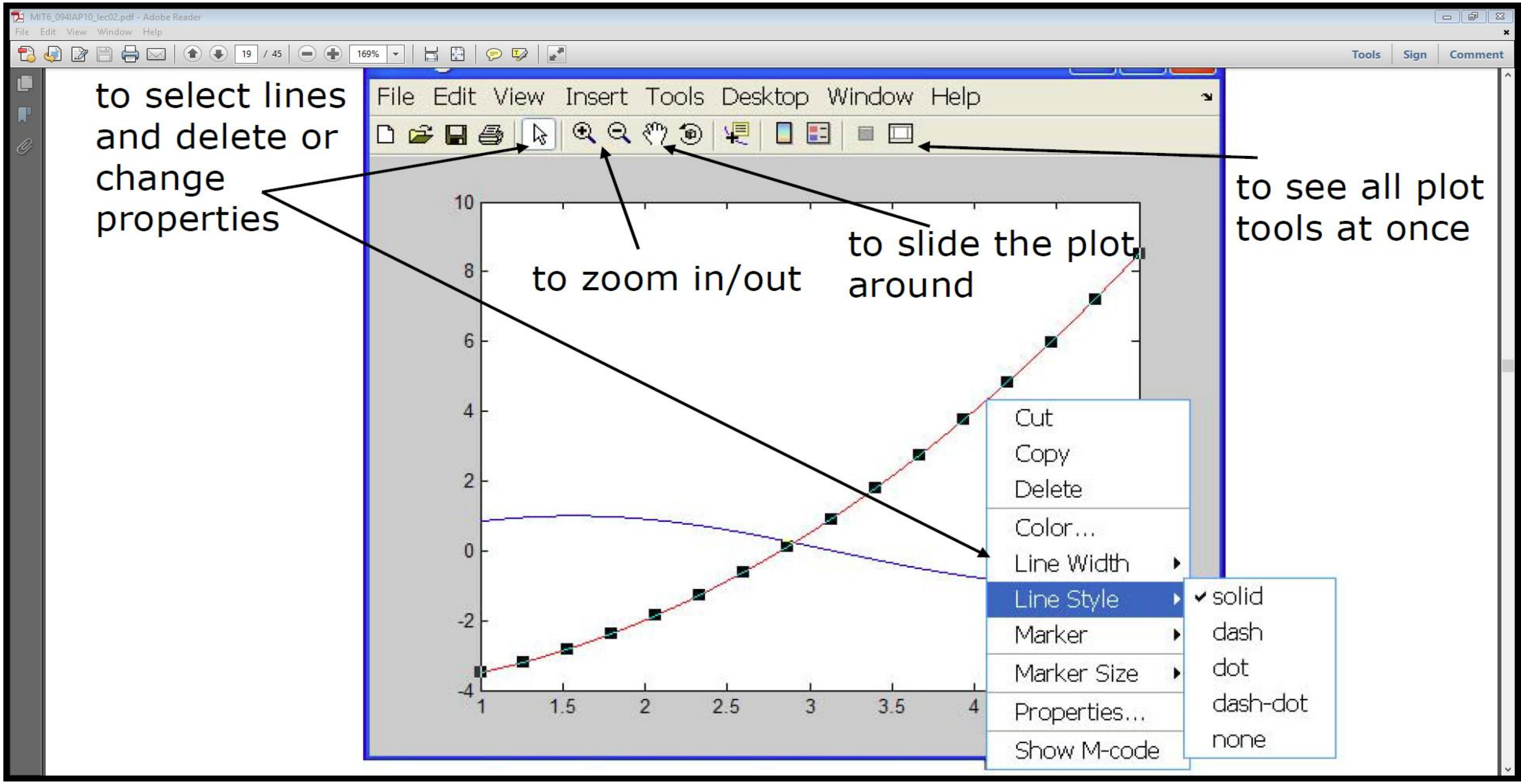


  - plot(x,y,’k.-’);
  - Colour marker linestyle

- •Can plot without connecting the dots by omitting line style argument»
  - plot(x,y,’.’)
- •Look at **help plot** for a full list of colours, markers, and linestyles

Plot Options

- When we build a plot we can modify everything



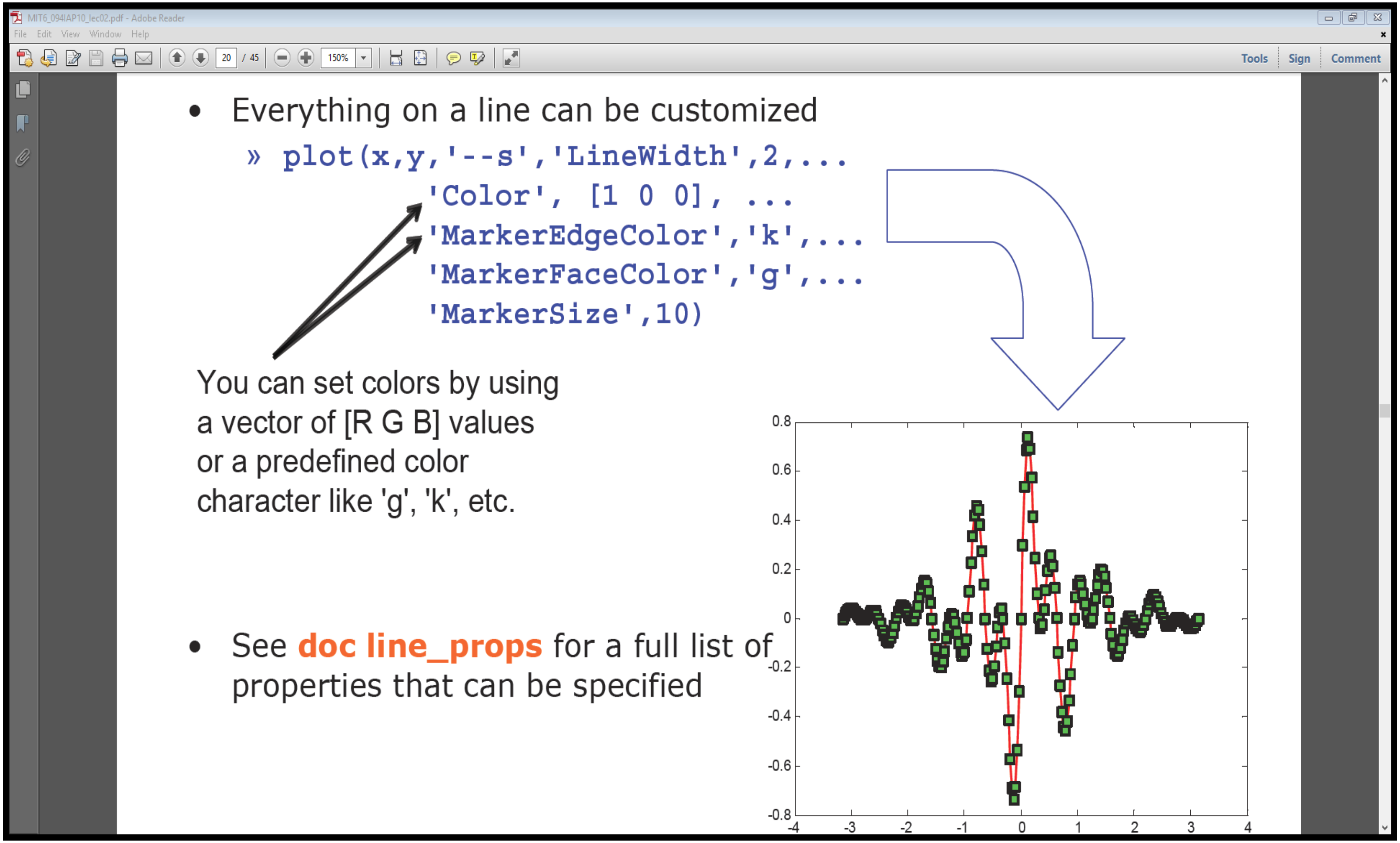
to select lines and delete or change properties

to zoom in/out

to slide the plot around

to see all plot tools at once

Plot Options



- Everything on a line can be customized

```
» plot(x,y,'--s','LineWidth',2,...
           'Color', [1 0 0], ...
           'MarkerEdgeColor','k',...
           'MarkerFaceColor','g',...
           'MarkerSize',10)
```

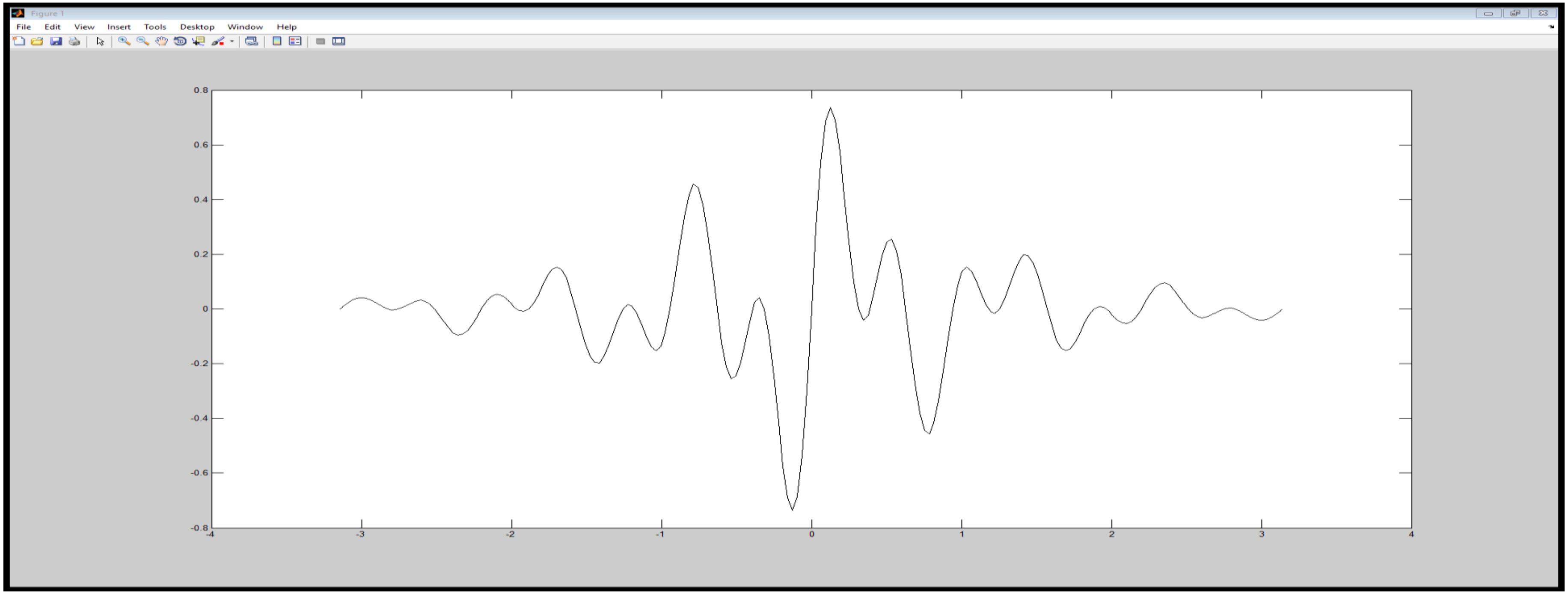
You can set colors by using a vector of [R G B] values or a predefined color character like 'g', 'k', etc.

- See **doc line_props** for a full list of properties that can be specified

Plot Options

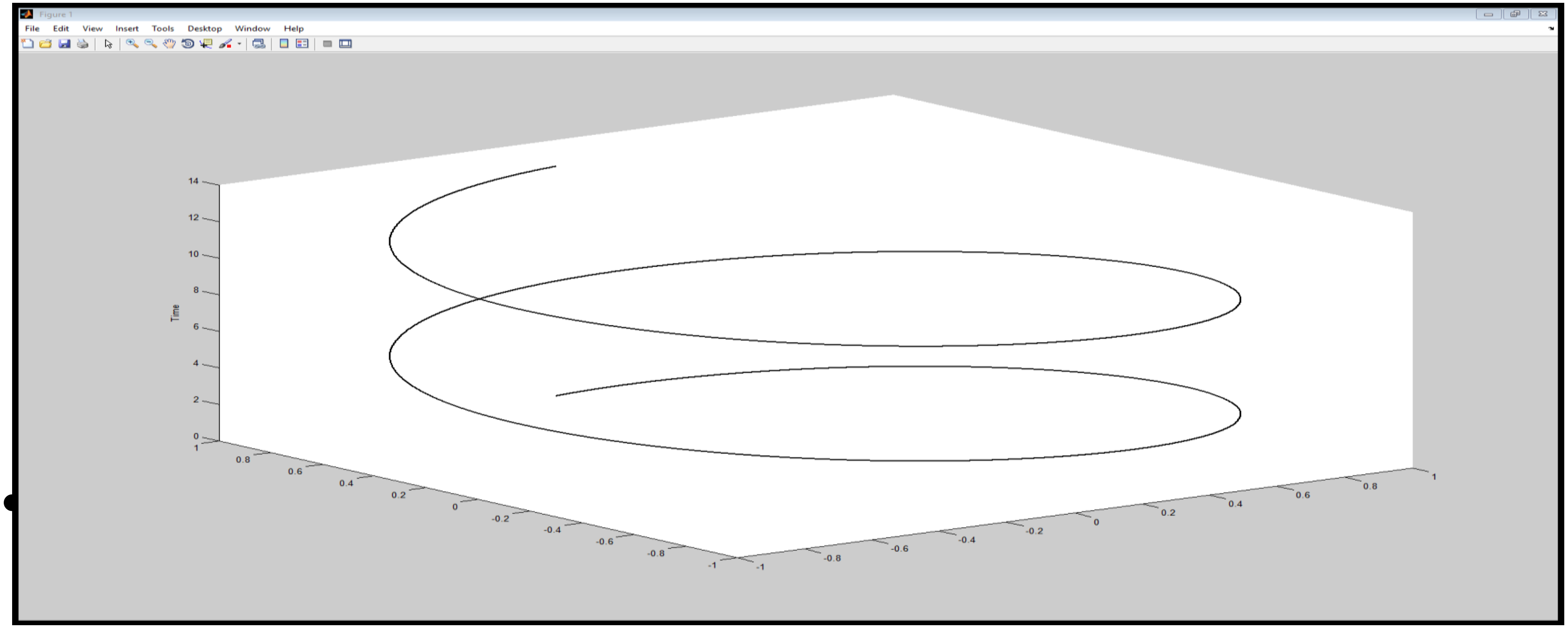
- Example

- x=-pi:pi/100:pi;
- y=cos(4*x).*sin(10*x).*exp(-abs(x));
- plot(x,y,'k-');



Plot Options

- We can plot in 3 dimensions just as easily as in previous example
  - time=0:0.001:4*pi;
  - x=sin(time);
  - y=cos(time);
  - z=time;
  - plot3(x,y,z,'k','LineWidth',2);
  - zlabel('Time');



Axis Modes

- **Built-in axis modes**
- **axis square**
  - makes the current axis look like a box
- **axis tight**
  - fits axes to data
- **axis equal**
  - makes x and y scales the same
- **axis xy**
  - puts the origin in the bottom left corner (default for plots)
- **axis ij**
  - puts the origin in the top left corner (default for matrices/images)

Plot Options

- Example
  - alpha = 0.5; beta = 2; N = 10;
  - x=[1:1:N]';
  - y=alpha + beta * x + randn(N,1);
  - save data x y;
  - figure(1);
  - plot(x,y,'b',x,alpha+x*beta,':r')
  - plot(x,y,'b',x,alpha+x*beta,':r','LineWidth',3) % width of line

- FIGURE(H) makes H the current figure, forces it to become visible,
- and raises it above all other figures on the screen. If Figure H
- does not exist, and H is an integer, a new figure is created with
- handle H.

Plot Options

- **hold on**
    - keeps the figure and allows to plot over existing lines:

    - % x=[1:1:N]';
    - % y=alpha + beta * x + randn(N,1);
    -
    - plot(x,y,'b','LineWidth',2)
    - hold on;
    - plot(x,alpha+x*beta,':r','LineWidth',1)
    - hold off;
    - legend('data','fitted data');
    - xlabel('x'); ylabel('y');
    - text(5, 1,'add text here')
    - title('Ordinary Least Square Regression')

Plot Options

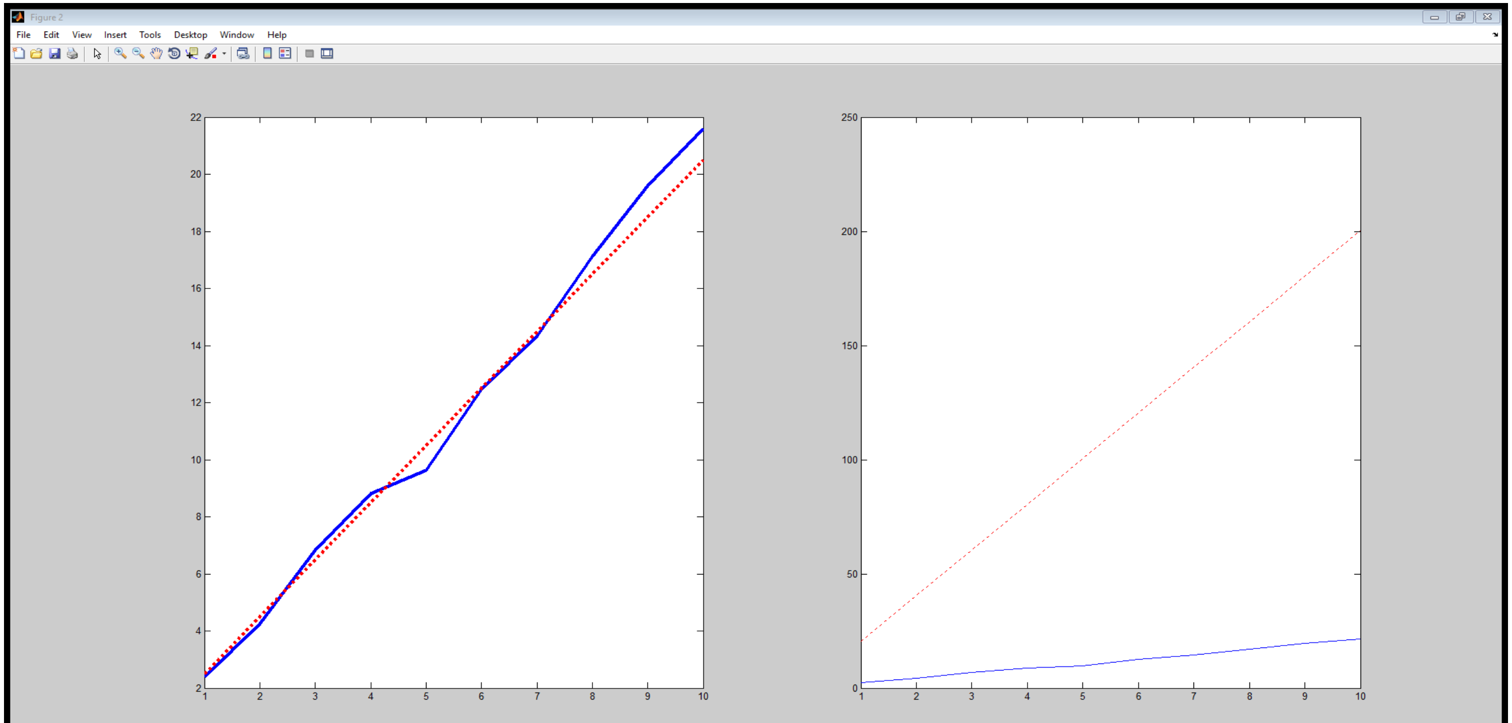
- Subplot(m,n,k)
  - breaks the Figure window into an m-by-n matrix of small axes
  - k position of picture in matrix

- Other words – subplot allow us to create many pictures in one page

Plot Options

- Example
  - figure(2)
  - subplot(1,2,1); plot(x,y,'b',x,alpha+x*beta,':r','LineWidth',3)
  - subplot(1,2,2); plot(x,y,'b',x,alpha+10*x*beta,':r')

- In this figure will be 1 ‘row’ and 2 ‘columns’ of pictures
- First picture second picture



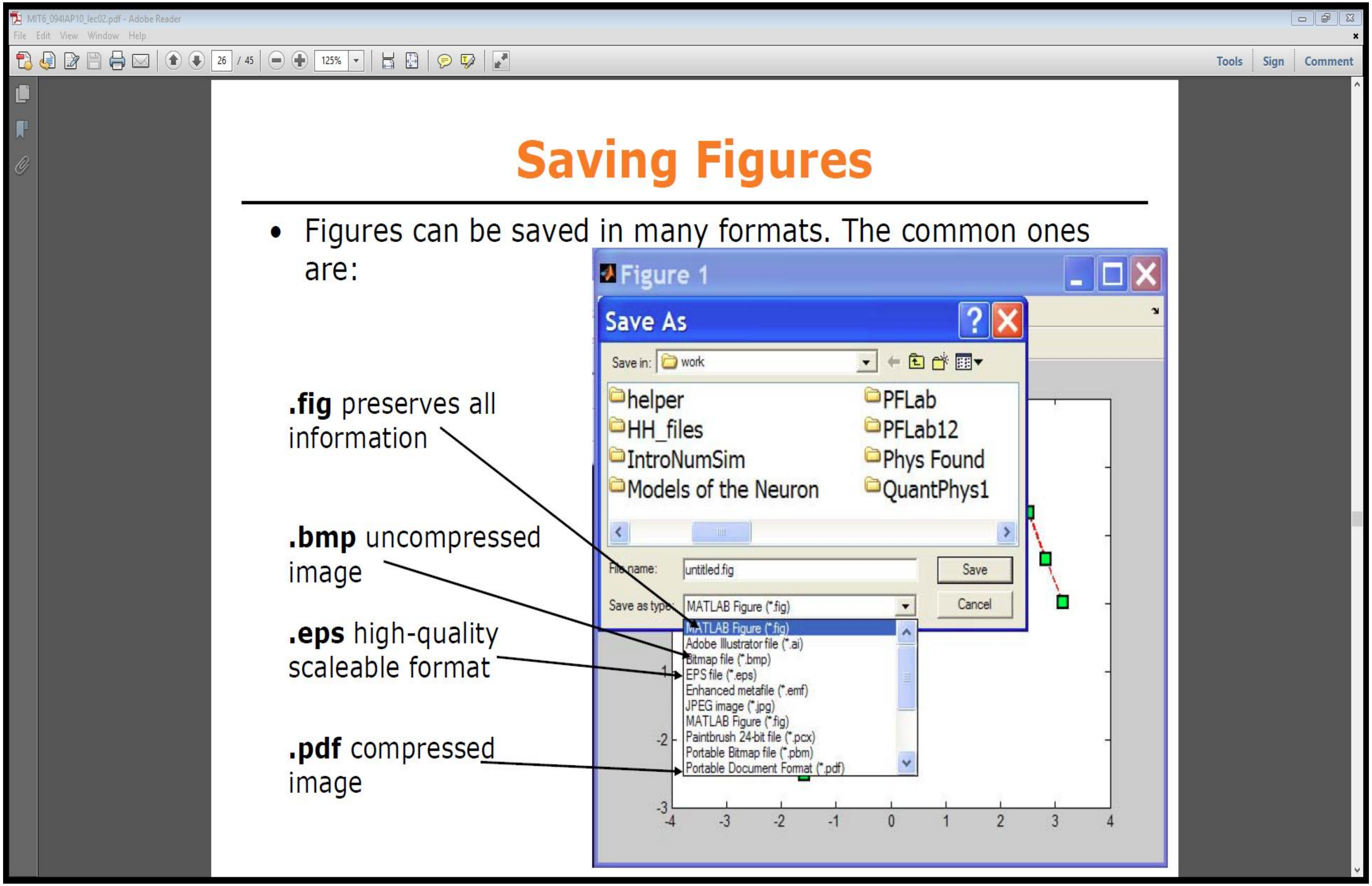
Plot Options

- Example
    - figure(3)
    - subplot(2,2,1); plot(x,y,'b',x,alpha+x*beta,':r','LineWidth',3)
    - subplot(2,2,2); plot(x,y,'b',x,alpha+x*beta,':r')
    - subplot(2,2,3); plot(x,y,'bn',x,alpha+x*beta,':m')
    - subplot(2,2,4); plot(x,y,'g:',x,alpha+x*beta,'y-.','LineWidth',6)
- Matrix (2X2), 4 pictures

Plot Options

- Exporting pictures
  - 1. File -> Export Setup
  - 2. Apply to figure
  - 3. Change the shape as you like
  - 4. Export -> choose any format you like
  - If you do not press Apply to figure, the saved file will not keep the chosen shape.

Plot Options



Saving Figures

- Figures can be saved in many formats. The common ones are:

.fig preserves all information

.bmp uncompressed image

.eps high-quality scaleable format

.pdf compressed image

Save in: work

File name: untitled.fig

Save as type: MATLAB Figure (*.fig)

3D plots

- It is more common to visualize *surfaces* in 3D
- Example:

F(x,y)=sin(x)cos(y)

Surf puts vertices at specified points in space x,y,z, and connects all the vertices to make a surface

The vertices can be denoted by matrices X,Y,Z

How can we make these matrices?

built-in function: **meshgrid**

3D plots

- Make the x and y vectors

```
x=-pi:0.1:pi;
y=-pi:0.1:pi;
```

- Use **meshgrid** to make matrices (this is the same as loop)

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

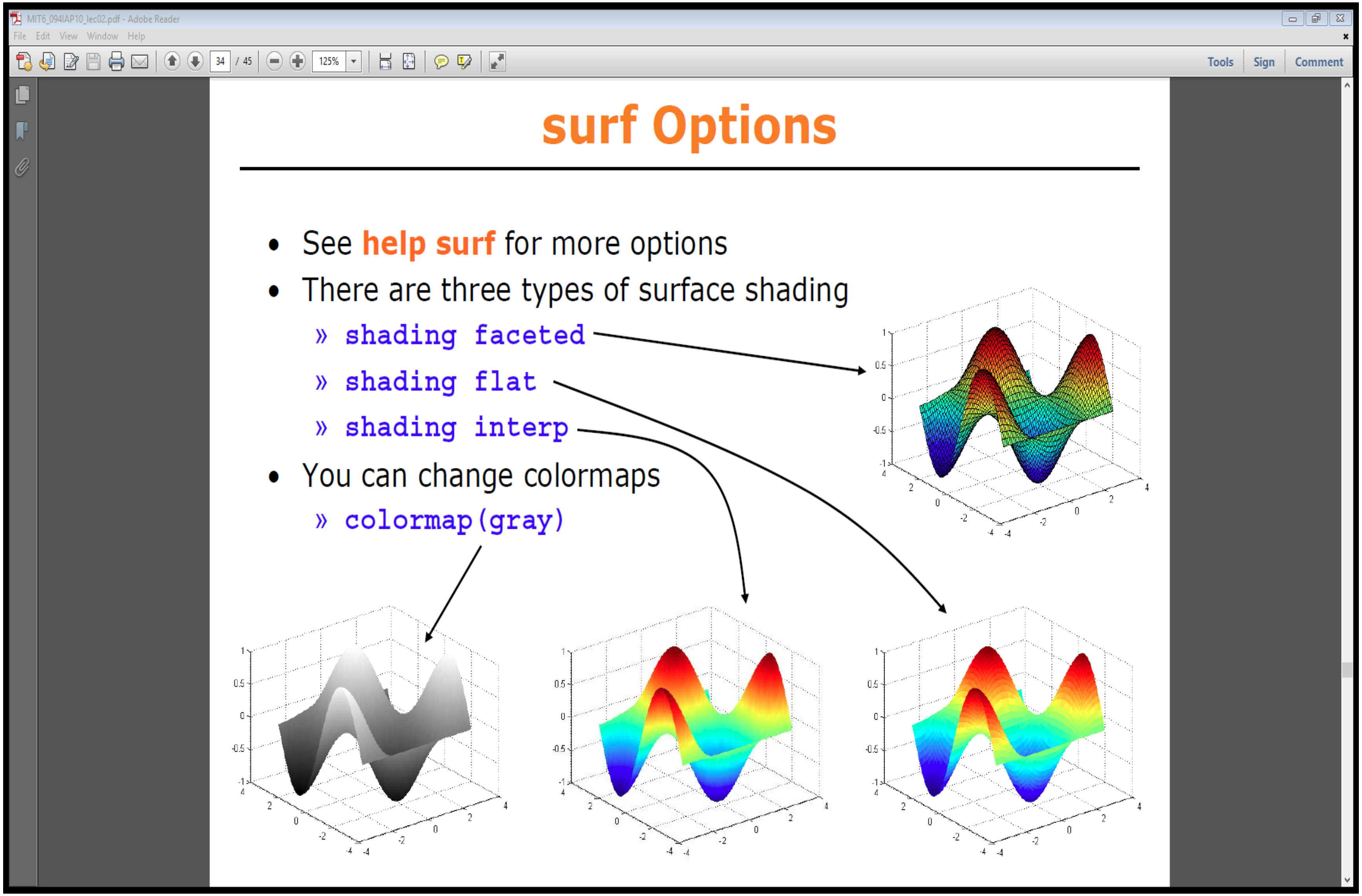
- to get function values, evaluate the matrices

```
Z =sin(X).*cos(Y);
```

- Plot the surface

```
surf(X,Y,Z)
surf(x,y,Z);
```

3D plots


MIT6_094IAP10_lec02.pdf - Adobe Reader
File Edit View Window Help
34 / 45
125%
Tools
Sign
Comment
1
0.5
0
-0.5
-1
4
2
0
-2
-4

surf Options

- See **help surf** for more options
- There are three types of surface shading

```
» shading faceted
» shading flat
» shading interp
```

- You can change colormaps

```
» colormap(gray)
```

3D plots

- **Contour**
- You can make surfaces two-dimensional by using contour
- **contour(X,Y,Z,'LineWidth',2)**
- takes same arguments as surf
- colour indicates height
- can modify linestyle properties
- can set colormap»
- **hold on**»
- **mesh(X,Y,Z)**

3D plots

- More examples
    - phi = 3;
    - c = [0.1:0.1:5];
    - n = [0.0:0.1:1];
    - [C,N] = meshgrid(c,n);
    - U = log(C) - N.^(1+phi)/(1+phi);
    - figure(4)
    - subplot(1,2,1); surf(C,N,U);
    - colormap('HSV');
    - xlabel('consumption'); ylabel('labour'); zlabel('utility')
    - subplot(1,2,2)
    - contour(C,N,U,'ShowText','on'); xlabel('consumption');
    - ylabel('labour')

3D plots

- More examples

```
for n=1:6
        figure(n)
        x=-pi:0.1:pi;
        y=-pi:0.1:pi;
        [X,Y]=meshgrid(x,y);

        Z =sin(n*X).*cos(Y/n);
        surf(X,Y,Z)

end
```

3D plots

- Same result using **While**

```
n=1;
while n<=6
     figure(n)
     x=-pi:0.1:pi;
     y=-pi:0.1:pi;

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

    Z =sin(n*X).*cos(Y/n);

    surf(X,Y,Z)
    n=n+1;
end
```