Chapter 6. 2-D and 3-D Plotting and Animation
Outline

• 6.1 Introduction
• 6.2 Anatomy of 2-D (x−y) Plots in MATLAB
• 6.3 Line Specifications
• 6.4 Accessing Plot Properties via the `get` and `set` Functions
• 6.5 Text Strings
• 6.6 Overlays, Legends, Subplots, and Multiple Figures
• 6.7 Special Types of 2-D Plots
• 6.8 3-D Plots
• 6.9 Changing Color Palette
• 6.10 Easy-to-Use Function Plots
• 6.11 Plot Editing
• 6.12 Interactive Plotting and Animation
• 6.13 Requirements to Engineering Plots
Plotting Basics

```matlab
x=0:pi/20:pi;
y=sin(x);
plot(x,y)

delimiters
```

```matlab
x=linspace(0,pi,20);
y=sin(x);
plot(x,y)

fplot('sin(x)',[0,pi])

plot(0:pi/20:pi,sin(0:pi/20:pi))
```

```matlab
plot(sin(0:pi/20:pi))
```

![Image of plots](image-url)
Plot’s Anatomy

ezplot('sin(x)')

figure
axis
line

title
xlabel

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
Enhancing Plot Readability

- `title('text')` – places text in a title at the top of a plot
- `axis([xmin xmax ymin ymax])` – sets the minimum and maximum limits of the x- and y-axes. (If you want to specify just one limit but want MATLAB to autoscale the other, use `Inf` or `–Inf` for autoscaled limits, or use `xlim` and `ylim` functions.) You may also use one of the following:
  - `axis square` – selects the axes’ limits so that the plot will be square
  - `axis equal` – assures that x-and y-axes have same tick mark spacing
  - `axis tight` – sets the limits to the range of the data
  - `axis manual` – freezes the scaling at the current limits
  - `axis auto` – returns axis scaling to its default autoscaling mode
  - `axis vis3d` – freezes aspect ratio properties to enable rotation of 3-D objects and overrides stretch-to-fill
- `v=axis` – returns the current axis scaling in the vector `v` (so that you might use the same scaling for another plot by calling `axis(v)`)
- `grid` – displays gridlines at the tick marks corresponding to the tick labels. Use `grid on` to turn on gridlines and `grid off` to turn off grid lines. When used by itself, `grid` toggles the grid switch on or off
- `xlabel('text')` – places a text label on the x-axis (abscissa)
- `ylabel('text')` – places a text label on the y-axis (ordinate)
- `text(x,y,'text')` – adds the text in the quotes to location (x,y) on the current axes
- `gtext('text')` – places the text in Figure window at a point specified by the click of the mouse
- `annotation(annotation_type)` – creates the specified annotation type, such as line, arrow, double arrow, text arrow, textbox, ellipse, and rectangle
grid, axis, and Text Strings

```matlab
>> t=linspace(0,2*pi);
>> line(cos(t),sin(t)), grid on

>> t=linspace(0,2*pi);
>> line(cos(t),sin(t)), axis equal

>> title('My Plot')
>> xlabel('x-axis'), ylabel('y-axis')
>> text(0.5,0.5,'Text')

>> annotation('arrow',[0.5 0.7],[0.7 0.8])
>> gtext('Chosen Location')
```
Text Strings Modifiers

Stream modifiers:

\bf - bold font
\it - italics font
\sl - oblique font (rarely available)
\rm - normal font
\fontname{fontname} - specify the name of the font family to use
\fontsize{fontsize} - specify the font size (in font units)

(The first four modifiers are mutually exclusive. Stream modifiers remain in effect until the end of the string or only within the context defined by braces {}.)

The subscript character “_” and the superscript character “^” modify the character or substring defined in braces immediately following.

To print the special characters used to define the TeX strings when Interpreter is TeX, prefix them with the backslash “\” character, for instance: \|, \{, \}, \_, \^.
## Special Symbols

<table>
<thead>
<tr>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>\alpha</td>
<td>α</td>
<td>\nu</td>
<td>ν</td>
<td>\Gamma</td>
<td>Γ</td>
<td>\equiv</td>
<td>≡</td>
</tr>
<tr>
<td>\beta</td>
<td>β</td>
<td>\xi</td>
<td>ξ</td>
<td>\Delta</td>
<td>Δ</td>
<td>\leq</td>
<td>≤</td>
</tr>
<tr>
<td>\gamma</td>
<td>γ</td>
<td>\pi</td>
<td>π</td>
<td>\Theta</td>
<td>Θ</td>
<td>\geq</td>
<td>≥</td>
</tr>
<tr>
<td>\delta</td>
<td>δ</td>
<td>\varpi</td>
<td>σ</td>
<td>\Lambda</td>
<td>Λ</td>
<td>\neq</td>
<td>≠</td>
</tr>
<tr>
<td>\epsilon</td>
<td>ε</td>
<td>\rho</td>
<td>ρ</td>
<td>\Xi</td>
<td>Ξ</td>
<td>\clubsuit</td>
<td>♣</td>
</tr>
<tr>
<td>\zeta</td>
<td>ζ</td>
<td>\sigma</td>
<td>σ</td>
<td>\Pi</td>
<td>Π</td>
<td>\diamondsuit</td>
<td>♥</td>
</tr>
<tr>
<td>\eta</td>
<td>η</td>
<td>\varsigma</td>
<td>ζ</td>
<td>\Sigma</td>
<td>Σ</td>
<td>\heartsuit</td>
<td>♥</td>
</tr>
<tr>
<td>\theta</td>
<td>θ</td>
<td>\tau</td>
<td>τ</td>
<td>\Upsilon</td>
<td>Υ</td>
<td>\spadesuit</td>
<td>♠</td>
</tr>
<tr>
<td>\vartheta</td>
<td>ϑ</td>
<td>\upsilon</td>
<td>υ</td>
<td>\Phi</td>
<td>Φ</td>
<td>\leftarrow</td>
<td>←</td>
</tr>
<tr>
<td>\iota</td>
<td>ι</td>
<td>\phi</td>
<td>φ</td>
<td>\Psi</td>
<td>ψ</td>
<td>\leftarrow</td>
<td>←</td>
</tr>
<tr>
<td>\kappa</td>
<td>κ</td>
<td>\chi</td>
<td>χ</td>
<td>\Omega</td>
<td>Ω</td>
<td>\uparrow</td>
<td>↑</td>
</tr>
<tr>
<td>\lambda</td>
<td>λ</td>
<td>\psi</td>
<td>ψ</td>
<td>\Im</td>
<td>ℑ</td>
<td>\rightarrow</td>
<td>→</td>
</tr>
<tr>
<td>\mu</td>
<td>μ</td>
<td>\omega</td>
<td>ω</td>
<td>\Re</td>
<td>ℜ</td>
<td>\downarrow</td>
<td>↓</td>
</tr>
</tbody>
</table>

---

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
<table>
<thead>
<tr>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
<th>Character Sequence</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>\otimes</td>
<td>⊗</td>
<td>\forall</td>
<td>∀</td>
<td>\circ</td>
<td>◦</td>
<td>\aleph</td>
<td>א</td>
</tr>
<tr>
<td>\cap</td>
<td>∩</td>
<td>\exists</td>
<td>∃</td>
<td>\pm</td>
<td>±</td>
<td>\wp</td>
<td>∅</td>
</tr>
<tr>
<td>\supset</td>
<td>⊃</td>
<td>\ni</td>
<td>∅</td>
<td>\sim</td>
<td>~</td>
<td>\oslash</td>
<td>Ø</td>
</tr>
<tr>
<td>\int</td>
<td>∫</td>
<td>\cong</td>
<td>≡</td>
<td>\propo</td>
<td>≅</td>
<td>\supseteq</td>
<td>⊃</td>
</tr>
<tr>
<td>\lfloor</td>
<td>\lfloor</td>
<td>\approx</td>
<td>≈</td>
<td>\partial</td>
<td>∂</td>
<td>\subset</td>
<td>⊂</td>
</tr>
<tr>
<td>\lceil</td>
<td>\lceil</td>
<td>\oplus</td>
<td>⊕</td>
<td>\bullet</td>
<td>•</td>
<td>\o</td>
<td>0</td>
</tr>
<tr>
<td>\perp</td>
<td>⊥</td>
<td>\cup</td>
<td>∪</td>
<td>\div</td>
<td>÷</td>
<td>\nabla</td>
<td>∇</td>
</tr>
<tr>
<td>\wedge</td>
<td>∧</td>
<td>\subseteq</td>
<td>⊆</td>
<td>\infty</td>
<td>∞</td>
<td>\ldots</td>
<td>…</td>
</tr>
<tr>
<td>\rceil</td>
<td>\rceil</td>
<td>\in</td>
<td>∈</td>
<td>\cdotp</td>
<td>·</td>
<td>\prime</td>
<td>′</td>
</tr>
<tr>
<td>\vee</td>
<td>∨</td>
<td>\lceil</td>
<td>⌈</td>
<td>\neg</td>
<td>¬</td>
<td>\emptyset</td>
<td>∅</td>
</tr>
<tr>
<td>\langle</td>
<td>⟨</td>
<td>\rangle</td>
<td>⟩</td>
<td>\times</td>
<td>×</td>
<td>\mid</td>
<td></td>
</tr>
<tr>
<td>\surd</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>\copyright</td>
<td>©</td>
</tr>
</tbody>
</table>
Adding Equations to the Plot

```matlab
fplot('0.25*exp(-.006*t)*sin(0.2*t)',[0 900])
a='fontsize{12}\{\itAe}^{\alpha<\beta}\sin\beta\{\itttt}';
b='   \ \ \ \alpha<\beta';
title(strcat(a,b))
xlabel('\bf\itTime\rm, \musec')
ylabel('\fontname{Times}\fontsize{12}Amplitude')
t1='\Delta\tau={\tau_f\over N-1}';
text(0.1,0.7,'$$' t1 '$$', ... 'Interpreter','latex')
t2=['\bf A_1=\left\{\matrix{-1&0&\cr 1&-1&\cr -1&0&\cr}\right\}';
text(0.35,0.7,'$$' t2 '$$', ... 'Interpreter','latex')
t3=['\dot x(t)={dx\over d\tau}d\tau=\tau''(\tau)\lambda(\tau)'];
text(0.65,0.7,'$$' t3 '$$', ... 'Interpreter','latex')
axis off
```

\[ \Delta\tau = \frac{\tau_f}{N-1} \]

\[ A_1 = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ -1 & 0 \end{bmatrix} \]

\[ \dot x(t) = \frac{dx}{d\tau} \frac{d\tau}{dt} = x'(\tau)\lambda(\tau) \]
Multiple Plots and Line Specifications

\begin{verbatim}
x=0:pi/20:pi;
y1=sin(x); y2=cos(x);
plot(x,y1), hold
plot(x,y2)
legend('sin','cos')
\end{verbatim}

\begin{verbatim}
x=0:pi/20:pi;
y1=sin(x); y2=cos(x);
plot(x,y1,x,y2)
legend('sin','cos')
\end{verbatim}

\begin{verbatim}
x=0:pi/20:pi;
y1=sin(x); y2=cos(x);
plot(x,y1,x,y2,'-.sm')
legend('sin','cos')
\end{verbatim}
legend(...,'Location',Loc) adds a legend in the specified location, **Loc**, with respect to the axes. **Loc** may be either a 1x4 position vector or one of the following strings:

- 'North' ('n') inside plot box near top
- 'South' ('s') inside bottom
- 'East' ('e') inside right
- 'West' ('w') inside left
- 'NorthEast' ('ne') inside top right (default)
- 'NorthWest' ('nw') inside top left
- 'SouthEast' ('se') inside bottom right
- 'SouthWest' ('sw') inside bottom left
- 'NorthOutside' ('no') outside plot box near top
- 'SouthOutside' ('so') outside bottom
- 'EastOutside' ('eo') outside right
- 'WestOutside' ('wo') outside left
- 'NorthEastOutside' ('neo') outside top right
- 'NorthWestOutside' ('nwo') outside top left
- 'SouthEastOutside' ('seo') outside bottom right
- 'SouthWestOutside' ('sw') outside bottom left
- 'Best' ('b') least conflict with data in plot
- 'BestOutside' ('bo') least unused space outside plot

legend(...,'Location',Loc,'Orientation ','horizontal ','Box ','off ')
## LineSpec Options

<table>
<thead>
<tr>
<th>Line Types</th>
<th>Data Markers†</th>
<th>Colors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solid line</td>
<td>Dot (.)</td>
<td>Black k</td>
</tr>
<tr>
<td>Dashed line</td>
<td>Asterisk (*)</td>
<td>Blue b</td>
</tr>
<tr>
<td>Dash-dotted line</td>
<td>Cross (x)</td>
<td>Cyan c</td>
</tr>
<tr>
<td>Dotted line</td>
<td>Plus sign (+)</td>
<td>Green g</td>
</tr>
<tr>
<td></td>
<td>Circle (o)</td>
<td>Magenta m</td>
</tr>
<tr>
<td></td>
<td>Square (□)</td>
<td>Red r</td>
</tr>
<tr>
<td></td>
<td>Diamond (◇)</td>
<td>White w</td>
</tr>
<tr>
<td></td>
<td>Pentagram (star)</td>
<td>Yellow y</td>
</tr>
<tr>
<td></td>
<td>Hexagram (star of David)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Left triangle (◄)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Up triangle (▲)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Right triangle (►)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Down triangle (▼)</td>
<td></td>
</tr>
</tbody>
</table>

† Note that circle and all following markers are filled markers (you may fill them with color)

- **LineWidth** – specifies the width of the line (in points)
- **MarkerEdgeColor** – specifies the color of the marker or the edge color for filled markers
- **MarkerFaceColor** – specifies the color of the face of filled markers
- **MarkerSize** – specifies the size of the marker (in points)
Markers and Line Styles

>> x=-pi:pi; y=sin(x);
>> line(x,y,'Color','r','Marker','p',
    'LineWidth',3,'MarkerSize',7)

>> x=-pi:pi; y=sin(x);
>> plot(x,y.^2,'--s','MarkerSize',10,...
    'MarkerFaceColor','r')

>> x=linspace(0,pi); plot(x,cos(6*x).*exp(-x),'^')

>> x=linspace(0,pi); xm=linspace(0,pi,20);
>> plot(x,cos(6*x).*exp(-x),xm,cos(6*xm).*exp(-xm),'^')

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means,
x=linspace(0,15,200);
y1=10*exp(-0.36*x).*sin(2.7*x);
y2=200*exp(-0.05*x).*sin(0.8*x);
[ax,h1,h2]=plotyy(x,y1,x,y2);
xlabel('Time, s')
set(get(ax(1),'Ylabel'),'String','Short Period Motion ($\alpha$, $^o$)')
set(get(ax(2),'Ylabel'),'String','Phugoid Motion (h, ft)')
set(h2,'LineStyle',':')
title('T-37 Longitudinal Dynamics')

Prior to R2014b

Starting from R2014b
Finding a Graphics Object

```matlab
>> x = linspace(-pi,pi,50);
>> hp = plot(x,sin(x),x,cos(x),'rp',x,sin(x).*cos(x),'m+:');
>> legend([hp(1) hp(3)],'f_1=sin(x)','f_2=sin(x)*cos(x)','location','ne')

>> h1 = findobj('Marker','p');
>> delete(h1)
>> h2 = findobj('Tag','legend');
>> set(h2,'Color','g')
```

Comments:

a) the `findobj` call returns handles of the root object and all its descendants
b) the `get(gcf,'children')` call returns 2x1 graphics array composed of two handles, one of which is a legend handle

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
The subplot Function

```matlab
subplot(221), ezplot('sin(x)')
subplot(222), ezplot('cos(x)')
subplot(223), ezplot('sin(x).^2')
subplot(224), ezplot('cos(x).^2')
```

```matlab
subplot(2,2,1), ezplot('sin(x)')
subplot(2,2,2), ezplot('cos(x)')
subplot(2,2,[3 4]), ezplot('sin(x).*cos(x)')
```
Variety of the 2-D Plots

- loglog
- semilogy
- polar
- area
- pie
- barh
- hist
- errorbar
- compass
- feather
- stem
- stairs
- scatter
- fill
- triplot
generation=1:5;
number=[4.2 33.6 42.7 9.5 1];
price=[0.1 0.2 1 30 150];
bar(generation,number,'y')
xlabel('Generation')
ylabel('Number manufactured, thousands')
h1=gca;
% Setting the new axes atop the first ones
h2=axes('Position',get(h1,'Position'));
% Adding the second plot to the new axes
semilogy(generation,price,'LineWidth',3)
% Modifying second axes settings
set(h2,'Color','none','YAxisLocation', ...    'right', 'XLim',get(h1,'XLim'), ...    'XTickLabel',[], 'TickLength',[0 0])
text(3.5,10,'Price','Rotation',58)
ylabel('Price, M$'), title('Jet Fighters')
Two Types of 3-D Plots

Trajectories

\[ t = 0:pi/50:10*pi; \]
\[ \text{plot3} (\sin(t),\cos(t),t) \]
\[ \text{grid on, axis square} \]

Surfaces

\[ [x,y]=\text{meshgrid}(\text{linspace}(0,2*pi,100)); \]
\[ z=\sin(x).*\cos(y).^2; \]
\[ \text{plot3}(x,y,z) \]
\[ \text{axis equal} \]

\[ t=0:.1:10; y=\exp(-(.1+i)*t); \]
\[ \text{stem3}(\text{real}(y),\text{imag}(y),t) \]
\[ \text{hold on} \]
\[ \text{plot3}(\text{real}(y),\text{imag}(y),t,'r') \]
\[ \text{hold off, view(-39.5,62)} \]

\[ \text{view (az,el)} \]
Variety of Surface Plots

mesh

meshc

meshz

surf

surfc

waterfall
Similarities in 2-D and 3-D

- **quiver**
- **contour**
- **bar**

- **quiver3**
- **contour3**
- **bar3**
**Shapes**

```matlab
% cylinder
h = findobj('Type','surface');
C = rand(size(get(h,'CData')));
set(h,'CData',C)
axis square

% cylinder
hc = get(gca,'Children')
rotate(hc,[1 1 1],45)
set(hc,'FaceColor','g')
axis equal

% sphere
hold on
[x,y,z] = sphere;
surf(x+2,y,z)
axis equal

% ellipsoid
ellipsoid(0,0,0,2,10,1)
hold, axis equal
[x,y,z] = cylinder(0.5,40);
surf(x,y,z+0.5)
shading interp
view([8,14])
colormap colorcube

% ellipsoid
ellipsoid(0,0,0,2,2,1)
h = get(gca,'Children');
rotate(h,[3 3 3],34)
z = get(h,'Zdata');
shading flat
set(h,'Zdata',z+2), axis equal
hs = mesh([-2 2; -2 2],...
[-2 -2; 2 2]),2.5*ones(2));
set(hs,'FaceColor','c')
```
Using Different Color Maps and colormap Editor

```matlab
>> colormap('name')
Jet
HSV
Hot
Cool
Spring
Summer
Autumn
Winter
Gray
Bone
Copper
Pink
Lines
[x,y]=meshgrid(linspace(0,2*pi,100));
z=sin(x).*cos(y).^2;
surf(x,y,z)
colormap('Cool')
axis equal
set(gca,'Visible','off')
```

Prior to R2014b

After R2014b

All rights reserved. No part of this publication may be reproduced, distributed, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
fill3 and patch Functions

\begin{verbatim}
x = [0 1; 1 0; 0 1];
y = [1 0; 1 1; 0 1];
z = [1 1; 1 1; 0 1];
c = [1 1; 0 0; 1 1];
fill3(x,y,z,c)
xlabel('x'), ylabel('y')
t=linspace(0,2*pi,11);
x=cos(t); y=sin(t);
x(2:2:10)=.4*cos(t(2:2:10));
y(2:2:10)=.4*sin(t(2:2:10));
patch(x,y,'b')
axis square

d = [0 0 1; 0 1 1; 0 1 0];
y = [0 1 1; 0 0 0; 1 1 1];
z = [0 0 1; 0 1 1; 0 1 0];
patch(x,y,z,-z)
axis square, view([35,35])
zlabel('z'), ylabel('y')
\end{verbatim}
Creating a 3-D Object Using Patch Graphics

```matlab
fx=[6.6 5.3 2.3 1.5 -2 -2.5 -10 -10 0 5.3 5.3]'; % fuselage
fy=zeros(11,1);
fz=[1.2 1.7 1.9 1.2 1.2 2.6 1.1 0.7 -0.2 0.3 0.8]';
cx=[2.3 0 -2.5 -2 1.5]'; % cockpit canopy
cy=zeros(5,1);
cz=[1.9 3.1 2.6 1.2 1.2]';
wingx=2.3*[1 -3 -3+.6 -3+.6 -3 -3+.6 -3+.6 -3 -3 1]/4; % wing
wingy=[8.5 8.5 7.8 7.8 2.2 2.2 -2.2 -2.2 -7.8 -7.8 -8.5 -8.5]';
wingz=zeros(12,1);
ax=2.3*[-3+.6 -3 -3 -3+.6]'/4; % right aileron
ay=[7.8 7.8 2.2 2.2]';
az=zeros(4,1);
hsx=1.3*[0 -1 -1 0]'-8.1; % horizontal stabilizer
hsy=5.1*[1 1 -1 -1]'/2;
hsz=0.7*ones(4,1);
ex=0.6*[0 -1 -1 0]-'9.4; % elevator
ey=5.1*[1 1 -1 -1]'/2;
ez=0.7*ones(4,1);
vsx=[0 -1.7 -2.7 -1.8]'-7.3; % vertical stabilizer
vsy=zeros(4,1);
vsz=[1.65 4.5 4.5 1.25]';
rx=[-2.7 -3.3 -3.1 -1.8]'-7.3; % rudder
ry=zeros(4,1);
rz=[4.5 4.5 1.5 1.25]';

pf=patch(fx,fy,fz,'c'); pw=patch(wingx,wingy,wingz,'c');
pc=patch(cx,cy,cz,'b'); ps=patch([hsx vsx],[hsy vsy],[hsz vsz],'c');
pu=patch([ax ax ex rx],[ay -ay ey ry],[az az ez rz],'m')
haircraft=[pf pc pw ps pu];
axis equal, axis off, view(135,20)
for i=1:1000
    rotate(haircraft,[25,35,1])
end
```

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
Property Editor
Saving Plots

print ('-dbitmap','Name')
print ('-dtiffn','Name')
print ('-dbmp','Name')
print ('-djpeg','Name')
plot(sin(0:pi/20:pi))
Accessing Graphics Properties via Property Editor

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.

Engineering Computations and Modeling in MATLAB/Simulink
plot(sin(0:pi/20:pi))

h_line = plot(sin(0:pi/20:pi));

get(h_line)

Prior to R2014b

Color: [0 0 1]
EraseMode: 'normal'
LineStyle: '-'
LineWidth: 0.5000
Marker: 'none'
MarkerSize: 6
MarkerEdgeColor: 'auto'
MarkerFaceColor: 'none'

...
Exploiting Parent – Children Relationship

Parent ➔ Children

Figure ➔ Axis ➔ Line

```
h_line= plot(sin(0:pi/20:pi));
```

```
gcf
```

```
get(0,'children')
```

```
get(0,'CurrentFigure')
```

```
get(gca,'parent')
```

```
get(gca,'children')
```

```
get(gcf,'children')
```

```
get(gcf,'children'), 'children')
```

```
get(gca,'children')
```

```
get(gca,'parent')
```

```
get(h_line,'parent')
```

```
get(h_line,'children')
```

Comment: any object has a single parent, but may have several children, e.g. several axes (and legends) within one figure, several lines within one axis.
Changing Graphics Object Properties

Prior to R2014b

```matlab
h_line = plot(sin(0:pi/20:pi));
get(h_line)
```

Starting from R2014b

```matlab
h_line = plot(sin(0:pi/20:pi));
get(h_line)
```

```matlab
set(h_line,'Color','r','LineWidth',2.5);
set(h_line,'ydata',get(h_line,'ydata')*5);
```

Prior to R2014b

```matlab
set(H,'PropertyName',PropertyName,.PropertyValue)
```

Starting from R2014b

```matlab
H.PropertyName = PropertyValue
```

```matlab
h_line.Color = 'r';
h_line.LineWidth = 2.5;
h_line.YData = 5*h_line.YData;
```
Tools for Creating Animations

‘Creating Movies’

figure('Renderer','zbuffer')

% Defining a membrane
r = [0:0.05:1]';       % Radius vector
phi = 0:pi/20:2*pi;   % Phi angle vector
x = r*cos(phi);       % x-coordinates of a grid
y = r*sin(phi);       % y-coordinates of a grid
z = besselj(1,3.8316*r)*cos(phi);

% Plotting the membrane
mesh(x,y,z)
xlabel('x-axis'), ylabel('y-axis')
zlabel('z-axis'), axis tight
YY=axis; set(gca,'zlim',YY(5:6))
set(gca,'nextplot','replacechildren');

% Creating movie frames
for j = 1:20
    mesh(x,y,sin(2*pi*j/20)*z,z);
    F(j) = getframe;
end

% Starting the movie
k=questdlg('Ready to watch the movie?',...
' 'Start the Movie', 'Yes', 'No', 'Yes');

% Playing the movie ten times
if char(k(1))=='Y'
movie(F,10)
end

‘Erase Mode’

% Plotting a sinusoid
x=0:0.1:2*pi;       % Defines the x scale
y=sin(x);           % Computes sin(x)
z=cos(x);           % Computes cos(x)
plot(x,y);          % Plots sin(x) curve
set(gca,'xlim',[-2 2*pi],'ylim',[-1 1]);
set(gca,'XTick',[-2:2:2*pi])
set(gca,'XTickLabel',[-2;2*pi])
xlabel('x'), ylabel('y=f(x)')

% Getting a handle to the line
h_line=get(gca,'children');

% Changing line properties
for i=1:1000
    pause(0.005)
    w=i/1000;
    d=(1-w)*y+w*z;
    set(h_line,'ydata',d,'EraseMode','none');
end

(‘normal’, ‘background’, ‘xor’)

Double strikethrough text should be removed if using R2014b+ 
Animated Rotation

figure('color','w')
quiver3(0,0,0,1.5,0,0,'LineWidth',2), hold on
quiver3(0,0,0,0,1.5,0,'LineWidth',2)
quiver3(0,0,0,0,0,1.5,'LineWidth',2)
text(1.5,0,0,'n_1'); text(0,1.5,0,'n_2'); text(0,0,1.5,'n_3')
axis([-1 1 -1 1 -1 1]), view(130,30)
expm(1, 1), ylabel('y_i'), zlabel('z_i')
R=eye(3);

for i = 1:200
    psi=4*pi*(i-1)/99; phi=2*pi*(i-1)/99; theta=pi*(i-1)/99;
    R=Euler2DCM(psi,theta,phi);
    for j=1:3
        set(h(j),'UData',R(j,1),'VData',R(j,2),'WData',R(j,3));
        set(ht(j),'Position',R(j,:));
    end
    phid=mod(phi*180/pi,360);     if phid>180,    phid=phid-360;     end
    thetad=mod(theta*180/pi,360); if thetad>180, thetad=thetad-360; end
    psid=mod(psi*180/pi,360);
    set(ha(1),'String',int2str(psid) ' ^o');
    set(ha(2),'String',int2str(thetad) ' ^o');
    set(ha(3),'String',int2str(phid) ' ^o');
    pause(0.01)
end

function R=Euler2DCM(psi,theta,phi)
    Rpsi = [ cos(psi) sin(psi) 0; ...
           -sin(psi) cos(psi) 0; 0 0 1];
    Rtheta = [cos(theta) 0 -sin(theta); 0 1 0; ...
               sin(theta) 0 cos(theta)];
    Rphi = [1 0 0; 0 cos(phi) sin(phi); ...
            0 -sin(phi) cos(phi)];
    R = Rphi*Rtheta*Rpsi;

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
Embedded Means for Animations

- `comet` instead of `plot`
- `comet3` instead of `plot3`

*Also, see* `animateline`, `addpoint`, `drawnow`
Saving Animations

>> aviobj = VideoWriter('sin2cos.avi');
open(aviobj);

>> aviobj;
writeVideo(aviobj,F);

>> aviobj;
close(aviobj);

% Plotting a sinusoid
x=0:0.1:2*pi;
y=sin(x);
z=cos(x);
plot(x,y)
set(gcf,'Renderer','zbuffer');
set(gca,'xlim',[0 2*pi],'ylim',[-1 1]);
set(gca,'XTick',[0:pi:2*pi])
set(gca,'XTickLabel',{'0';'pi';'2pi'})
xlabel('x'), ylabel('y=f(x)')

% Getting a handle to the line
h_line=get(gca,'children');

% Changing line properties
for i=1:100
  % Setting the weighting coefficient w
  w(i)=i/100;
  % Blending sin(x) and cos(x) using w
  d=(1-w(i))*y+w(i)*z;
  % Changing ydata for the line
  set(h_line,'ydata',d,EraseMode','normal');
  F = getframe;
  writeVideo(aviobj,F);
end

close(aviobj);

Settings the Renderer property to 'zbuffer' or 'painters' works around limitations of 'getframe' with the OpenGL renderer on some Windows systems.

<table>
<thead>
<tr>
<th>VideoWriter Method</th>
<th>File: C:sin2cos.avi</th>
<th>Format: avi</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Properties</td>
<td>'sin2cos.avi'</td>
<td>'avi'</td>
</tr>
<tr>
<td>Video Properties</td>
<td>'3', '429', '545', '100', '24', '75'</td>
<td></td>
</tr>
<tr>
<td>ColorChannels</td>
<td>'RGB24'</td>
<td>'RGB24'</td>
</tr>
<tr>
<td>FrameFormat</td>
<td>'Motion JPEG'</td>
<td>'Motion JPEG'</td>
</tr>
<tr>
<td>Quality</td>
<td>'H.264'</td>
<td>'H.264'</td>
</tr>
<tr>
<td>Codec</td>
<td>'Motion JPEG 2000'</td>
<td>'Motion JPEG 2000'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'None'</td>
<td>'None'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'Motion JPEG'</td>
<td>'Motion JPEG'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'Grayscale'</td>
<td>'Grayscale'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'Indexed'</td>
<td>'Indexed'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'mp4'</td>
<td>'mp4'</td>
</tr>
<tr>
<td>Vdcode</td>
<td>'m2'</td>
<td>'m2'</td>
</tr>
</tbody>
</table>
Note, that the Indeo5 codec (used by default by the `movie2avi` and `avifile` functions) and others were shipped with Windows XP, Windows XP SP2, and prior versions, however they are not shipped with the following operating systems: Windows XP SP1, Windows XP x64, Windows Vista (32/64), and Windows 7 (32/64). You need to install these codecs separately or use no compression.

The frame height and width will be padded to be a multiple of four as required by majority of codices.

After R2014b VideoWriter should be used (movie2avi is removed).
Graphical Input From Mouse

axis([0 10 0 10]), hold on
xy = []; n = 0; % Initially, the list of points is empty
%% Picking up multiple points by clicking left mouse button
% (Right mouse button means you are picking the last point)
but = 1;
while but == 1
    [xi,yi,but] = ginput(1);
    plot(xi,yi,'ro')
    n = n+1;
    xy(:,n) = [xi;yi];
end
%% Interpolating with a spline curve and finer spacing
n = n+1;
xy(:,n+1) = [xi;yi];
t = 1:n; ts = 1:0.1:n;
xys = spline(t,xy,ts);
%% Plot the interpolated curve
plot(xys(1,:),xys(2,:),'b-'); hold off

[X,Y] = ginput(N) gets N points from the current axes and returns the x- and y-coordinates in length N vectors X and Y (data points are entered by pressing a mouse button or any key on the keyboard except carriage return, which terminates the input before N points are entered.)

[X,Y,BUTTON] = ginput(N) returns a third result, BUTTON, that contains a vector of integers specifying which mouse button was used (1,2,3 from left).
All rights reserved. No part of this publication may be reproduced, distributed, or transmitted, unless for course participation, by any means, or stored in a database or retrieval system, without the prior written permission of the Publisher and/or Author. Contact the American Institute of Aeronautics and Astronautics, Professional Development Programs, Suite 500, 1801 Alexander Bell Drive, Reston, VA 20191-4344.
The End of Chapter 6

Questions?