Outline

- 6.1 Introduction
- 6.2 Anatomy of a 2-D \((x-y)\) Plot in MATLAB
- 6.3 Line Specifications
- 6.4 Accessing and Modifying Properties of the Graphics Objects
- 6.5 Text Annotations
- 6.6 Overlays, Legends, Subplots, and Multiple Figures
- 6.7 2-D Plotting Functions
- 6.8 3-D Plotting Functions
- 6.9 Changing Color Palette
- 6.10 Easy-to-Use Plotting Functions
- 6.11 Plot Editing
- 6.12 Interactive Plotting and Animation
- 6.13 Requirements to Engineering Plots
ezplot('sin(x)')

figure

axis

line

title

xticks

xticklabels

xlabel

Default line color: [0.00,0.45,0.74] (before R2014b used to be [0,0,1])

Brush/Select Data
Data Tips
Pan
Zoom In
Zoom Out
Restore View

Link Plot
Insert Colorbar
Insert Legend

Edit Plot
Open Property Inspector

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>plot(X,Y,…)</code></td>
<td>creates a 2-D plot of data in Y versus the corresponding values in X (this function also creates the figure, axes and line objects)</td>
</tr>
<tr>
<td><code>line(X,Y,…)</code></td>
<td>adds a line defined by vectors X and Y (or multiple lines if either X and Y, or both are matrices) to the current axes. Unlike the <code>plot</code> function, <code>line</code> does not delete other graphics objects or reset axes properties</td>
</tr>
<tr>
<td><code>ezplot(f,…)</code></td>
<td>plots symbolic expression, equation, or function ( f(x) ) or ( f(x, y) ). The default range for ( x ) or both ( x ) and ( y ) is ([-2\pi; 2\pi])</td>
</tr>
<tr>
<td><code>ezplot(funx,funy,…)</code></td>
<td>plots the parametrically defined planar curve ( x = fun_x(t) ) and ( y = fun_y(t) ) over the default range ( 0 \leq t \leq 2\pi ) (similar function: <code>ezplot3</code>)</td>
</tr>
<tr>
<td><code>fplot(f,…)</code></td>
<td>plots the curve defined by the function ( y = f(x) ) over the default interval ([-5; 5]) for ( x ) (similar functions: <code>fsurf</code>, introduced in R2016a)</td>
</tr>
<tr>
<td><code>fplot(funx,funy,…)</code></td>
<td>plots the curve defined parametrically, as ( x = f_x(t) ) and ( y = f_y(t) ), over the default interval ([-5; 5]) for ( x ) (similar functions: <code>fplot3</code> and <code>fsurf</code>, both introduced in R2016a)</td>
</tr>
<tr>
<td><code>fimplicit(f,…)</code></td>
<td>plots the implicit function ( f(x,y) = 0 ) defined by a function handle ( f ) over the default interval ([-5; 5]) for ( x ) and ( y ) (introduced in R2016b) (similar functions <code>fcontour</code> and <code>fimplicit3</code>, introduced in R2016a and R2016b, respectively)</td>
</tr>
</tbody>
</table>
Plotting Basics

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

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

```matlab
plot(0:pi/20:pi, sin(0:pi/20:pi))
line(0:pi/20:pi, sin(0:pi/20:pi))
fplot(@(x) sin(x), [0, pi])
```
ezplot('sin(x)+sin(y)-sin(x.*y)')
grid, axis equal

fimplicit(@(x,y)sin(x)+sin(y)-sin(x.*y),
2*pi*[-1 1 -1 1])
grid, axis equal

fplot(@(t)sin(2*t),@(t)sin(2.5*t))
grid, axis equal

f(x) f(x,y)=0 x(t), y(t)

ax=gca;
S=sym(ax.XLim(1):pi/2:ax.XLim(2));
ax.XTick=double(S);
ax.XTickLabel=arrayfun(@texlabel,S,'UniformOutput',false);

Try this code to change the x-axis tick labels
enchanting Plot Readability

\texttt{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 \texttt{Inf} or \texttt{–Inf} for autoscaled limits, or use \texttt{xlim} and \texttt{ylim} functions.) You may also use one of the following:

- \texttt{axis square} – selects the axes’ limits so that the plot will be square
- \texttt{axis equal} – assures that x-and y-axes have same tick mark spacing
- \texttt{axis tight} – sets the limits to the range of the data
- \texttt{axis vis3d} – freezes aspect ratio properties to enable rotation of 3-D objects and overrides stretch-to-fill
- \texttt{axis manual} – freezes the scaling at the current limits
- \texttt{axis auto} – returns axis scaling to its default autoscaling mode
- \texttt{axis ij} – reverses a direction of the y-axis with the values increasing top to bottom

\texttt{v=axis} – returns the current axis scaling in the vector \texttt{v} (so that you might use the same scaling for another plot by calling \texttt{axis(v)})*

\texttt{axis off} – turns visibility of axes lines and background off

\texttt{grid} – displays gridlines at the tick marks corresponding to the tick labels. Use \texttt{grid on} to turn on gridlines and \texttt{grid off} to turn off grid lines. When used by itself, \texttt{grid} toggles the grid switch \texttt{on} or \texttt{off}.

\texttt{grid minor} – toggles the visibility of the minor grid lines (not, not all types of charts support it)

* You can also use \texttt{linkaxes} to synchronize limits of multiple axes
Enhancing Plot Readability

- `title('text')` - places text in a title at the top of a plot
- `title({'First line';'Second line'})` - creates a two-line title
- `sgtitle('text')` - adds a title above the grid of subplots in the current figure (introduced in R2018b)

- `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 mouse click

- `annotation(annotation_type)` - creates the specified annotation type, such as line, arrow, double arrow, text arrow, textbox, ellipse, and rectangle (introduced in R2016b)

- `xticks(ticks)` - sets the x-axis tick values for the current axis as a vector of increasing numeric numbers, \(ticks\), specifying locations along the x-axis where the tick marks appear. Similarly, the `yticks` and `zticks` functions are available
- `xticklabels(labels)` - sets the x-axis tick labels for the current axes as a string array or a cell array of character vectors \(labels\). Also, see `xtickangle` and `xtickformat`. Similar functions exist for y- and z-axis
grid, axis, and Annotations

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

```
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: \|, \{, \}, \_, \^.
<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>\leftrightarrow</td>
<td>↔</td>
</tr>
<tr>
<td>\iota</td>
<td>i</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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
## 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>\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>ϝ</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>\propto</td>
<td>∝</td>
<td>\supseteq</td>
<td>⊃</td>
</tr>
<tr>
<td>\rfloor</td>
<td>\rfloor</td>
<td>\approx</td>
<td>≈</td>
<td>\partial</td>
<td>∂</td>
<td>\subset</td>
<td>⊂</td>
</tr>
<tr>
<td>\lfloor</td>
<td>\lfloor</td>
<td>\oplus</td>
<td>⊕</td>
<td>\bullet</td>
<td>•</td>
<td>\emptyset</td>
<td>Ø</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>\wedge</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>\vee</td>
<td>\lceil</td>
<td>\lceil</td>
<td>\neg</td>
<td>¬</td>
<td>\emptyset</td>
<td>Ø</td>
</tr>
<tr>
<td>\angle</td>
<td>\langle</td>
<td>\rangle</td>
<td>\rangle</td>
<td>\times</td>
<td>×</td>
<td>\mid</td>
<td>\mid</td>
</tr>
<tr>
<td>\surd</td>
<td>\surd</td>
<td>\copyright</td>
<td>©</td>
<td></td>
<td></td>
<td></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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Adding Equations to the Plot

```matlab
fplot('0.25*exp(-.006*t)*sin(0.2*t)', [0 900])
a = '\textit{Ae}^{-\alpha \textit{itt}} \sin \beta\textit{itt}';
b = '\alpha << \beta';
title(strcat(a, b))
xlabel('\textbf{itTime}\ \text{rm, \ musec}')
ylabel('\textit{fontname}\{Times\}\textit{fontsize}\{12\}\textit{Amplitude}')
```

You may also try `text(.3,.3,\text{texlabel}'beta12*e^{-alpha*t}')\text{)}\text{)\text{)\text{)}}`, which uses the `\text{texlabel}` function to convert Greek variable names into a character vector that is displayed as Greek letters: $\beta_{12}e^{-\alpha t}$.
The easiest way to visualize what additional capabilities the LaTeX interpreter of MATLAB offers is to click on the Insert equation icon of the Live Editor’s Insert tab, which opens the Equation tab.

For advanced users
Markers and Line Styles

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

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

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

```matlab
>> x=-pi:pi; y=sin(x);
>> line(x,y,'Color','r','Marker','p',...
     'LineWidth',3,'MarkerSize',7)
```
### LineSpec Options

**plot(x,y,LineSpec)**

<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>Red r</td>
</tr>
<tr>
<td>Dashed line</td>
<td>Asterisk (*)</td>
<td>Green g</td>
</tr>
<tr>
<td>Dash-dotted line</td>
<td>Cross (x)</td>
<td>Blue b</td>
</tr>
<tr>
<td>Dotted line</td>
<td>Plus sign (+)</td>
<td>Cyan c</td>
</tr>
<tr>
<td></td>
<td>Circle (o)</td>
<td>Magenta m</td>
</tr>
<tr>
<td></td>
<td>Square (□)</td>
<td>Yellow y</td>
</tr>
<tr>
<td></td>
<td>Diamond (◇)</td>
<td>Black k</td>
</tr>
<tr>
<td></td>
<td>Pentagram (star)</td>
<td>White w</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)
### Properties of Line Object

**AlignVertexCenters:** 'off'  
**Annotation:** [1x1 ... Annotation]  
**BeingDeleted:** 'off'  
**BusyAction:** 'queue'  
**ButtonDownFcn:** ''  
**Children:** [0x0 GraphicsPlaceholder]  
**Clipping:** 'on'  
**Color:** [0 0.4470 0.7410]  
**CreateFcn:** ''  
**DataTipTemplate:** [1x1 ... DataTipTemplate]  
**DeleteFcn:** ''  
**DisplayName:** ''  
**HandleVisibility:** 'on'  
**HitTest:** 'on'  
**Interruptible:** 'on'  
**LineJoin:** 'round'  
**LineStyle:** '-'  
**LineWidth:** 0.5000  
**Marker:** 'none'  
**MarkerEdgeColor:** 'auto'  
**MarkerFaceColor:** 'none'  
**MarkerIndices:** [1 2 3 4 5 6 7 8 9 10]  
**MarkerSize:** 6  
**Parent:** [1x1 Axes]  
**PickableParts:** 'visible'  
**Selected:** 'off'  
**SelectionHighlight:** 'on'  
**Tag:** ''  
**Type:** 'line'  
**UIContextMenu:** [0x0 GraphicsPlaceholder]  
**UserData:** []  
**Visible:** 'on'  
**XData:** [1 2 3 4 5 6 7 8 9 10]  
**XDataMode:** 'auto'  
**XDataSource:** ''  
**YData:** [10 9 8 7 6 5 4 3 2 1]  
**YDataSource:** ''  
**ZData:** [1x0 double]  
**ZDataSource:** ''

---

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Properties of Axes Object

- **ALim**: [0 1]  
  - **ALimMode**: 'auto'  
- **AlphaScale**: 'linear'  
- **Alphamap**: [1x64 double]  
- **AmbientLightColor**: [1 1 1]  
- **BeingDeleted**: 'off'  
  - **Box**: 'on'  
  - **BoxStyle**: 'back'  
- **BusyAction**: 'queue'  
- **ButtonDownFcn**: ''  
- **CLim**: [0 1]  
  - **CLimMode**: 'auto'  
- **CameraPosition**: [5.5000 5.5000 17.3205]  
  - **CameraPositionMode**: 'auto'  
- **CameraTarget**: [5.5000 5.5000 0]  
  - **CameraTargetMode**: 'auto'  
- **CameraUpVector**: [0 1 0]  
  - **CameraUpVectorMode**: 'auto'  
- **CameraViewAngle**: 6.6086  
  - **CameraViewAngleMode**: 'auto'  
- **Children**: [1x1 Line]  
- **Clipping**: 'on'  
- **ClippingStyle**: '3dbox'  
- **Color**: [1 1 1]  
  - **ColorOrder**: [7x3 double]  
- **ColorOrderIndex**: 2  
- **ColorScale**: 'linear'  
  - **Colormap**: [64x3 double]  
- **CreateFcn**: ''  
- **CurrentPoint**: [2x3 double]  
  - **DataAspectRatio**: [4.5000 4.5000 1]  
  - **DataAspectRatioMode**: 'auto'  
- **DeleteFcn**: ''  
  - **FontAngle**: 'normal'  
  - **FontName**: 'Helvetica'  
  - **FontSize**: 10  
  - **FontSizeMode**: 'auto'  
- **FontSmoothing**: 'on'  
  - **FontUnits**: 'points'  
  - **FontWeight**: 'normal'  
- **GridAlpha**: 0.1500  
- **GridAlphaMode**: 'auto'  
  - **GridColor**: [0.1500 0.1500 0.1500]  
- **GridColorMode**: 'auto'  
- **GridLineStyle**: '-'  
- **HandleVisibility**: 'on'  
  - **HitTest**: 'on'  
  - **Interactions**: [1x ...DefaultAxesInteractionSet]  
  - **Interruptible**: 'on'  
  - **LabelFontSizeMultiplier**: 1.2000  
  - **Layer**: 'bottom'  
- **Legend**: [0x0 GraphicsPlaceholder]  

The last 17 lines are repeated for Y and Z axes
### Properties of Figure Object

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alphamap</td>
<td>[1×64 double]</td>
</tr>
<tr>
<td>BeingDeleted</td>
<td>'off'</td>
</tr>
<tr>
<td>BusyAction</td>
<td>'queue'</td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>''</td>
</tr>
<tr>
<td>Children</td>
<td>[1×1 Axes]</td>
</tr>
<tr>
<td>Clipping</td>
<td>'on'</td>
</tr>
<tr>
<td>CloseRequestFcn</td>
<td>'closereq'</td>
</tr>
<tr>
<td>Color</td>
<td>[0.9400 0.9400 0.9400]</td>
</tr>
<tr>
<td>Colormap</td>
<td>[64×3 double]</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>''</td>
</tr>
<tr>
<td>CurrentAxes</td>
<td>[1×1 Axes]</td>
</tr>
<tr>
<td>CurrentCharacter</td>
<td>''</td>
</tr>
<tr>
<td>CurrentObject</td>
<td>[0×0 GraphicsPlaceholder]</td>
</tr>
<tr>
<td>CurrentPoint</td>
<td>[0 0]</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>''</td>
</tr>
<tr>
<td>DockControls</td>
<td>'on'</td>
</tr>
<tr>
<td>FileName</td>
<td>''</td>
</tr>
<tr>
<td>GraphicsSmoothing</td>
<td>'on'</td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>'on'</td>
</tr>
<tr>
<td>InnerPosition</td>
<td>[1440 918 560 420]</td>
</tr>
<tr>
<td>IntegerHandle</td>
<td>'on'</td>
</tr>
<tr>
<td>Interruptible</td>
<td>'on'</td>
</tr>
<tr>
<td>InvertHardcopy</td>
<td>'on'</td>
</tr>
<tr>
<td>KeyPressFcn</td>
<td>''</td>
</tr>
<tr>
<td>KeyReleaseFcn</td>
<td>''</td>
</tr>
<tr>
<td>MenuBar</td>
<td>'figure'</td>
</tr>
<tr>
<td>Name</td>
<td>''</td>
</tr>
<tr>
<td>NextPlot</td>
<td>'add'</td>
</tr>
<tr>
<td>Number</td>
<td>1</td>
</tr>
<tr>
<td>NumberTitle</td>
<td>'on'</td>
</tr>
<tr>
<td>OuterPosition</td>
<td>[1432 910 576 513]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PaperOrientation</td>
<td>'portrait'</td>
</tr>
<tr>
<td>PaperPosition</td>
<td>[1.3333 3.3125 5.8333 4.3750]</td>
</tr>
<tr>
<td>PaperPositionMode</td>
<td>'auto'</td>
</tr>
<tr>
<td>PaperSize</td>
<td>[8.5000 11]</td>
</tr>
<tr>
<td>PaperType</td>
<td>'usletter'</td>
</tr>
<tr>
<td>PaperUnits</td>
<td>'inches'</td>
</tr>
<tr>
<td>Parent</td>
<td>[1×1 Root]</td>
</tr>
<tr>
<td>Pointer</td>
<td>'arrow'</td>
</tr>
<tr>
<td>PointerShapeCData</td>
<td>[16×16 double]</td>
</tr>
<tr>
<td>PointerShapeHotSpot</td>
<td>[1 1]</td>
</tr>
<tr>
<td>Position</td>
<td>[1440 918 560 420]</td>
</tr>
<tr>
<td>Renderer</td>
<td>'opengl'</td>
</tr>
<tr>
<td>RendererMode</td>
<td>'auto'</td>
</tr>
<tr>
<td>Resize</td>
<td>'on'</td>
</tr>
<tr>
<td>Scrollable</td>
<td>'off'</td>
</tr>
<tr>
<td>SelectionType</td>
<td>'normal'</td>
</tr>
<tr>
<td>SizeChangedFcn</td>
<td>''</td>
</tr>
<tr>
<td>Tag</td>
<td>''</td>
</tr>
<tr>
<td>ToolBar</td>
<td>'auto'</td>
</tr>
<tr>
<td>Type</td>
<td>'figure'</td>
</tr>
<tr>
<td>UIContextMenu</td>
<td>[0×0 GraphicsPlaceholder]</td>
</tr>
<tr>
<td>Units</td>
<td>'pixels'</td>
</tr>
<tr>
<td>UserData</td>
<td>[]</td>
</tr>
<tr>
<td>Visible</td>
<td>'on'</td>
</tr>
<tr>
<td>WindowButtonDownFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowButtonMotionFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowButtonUpFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowKeyPressFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowKeyReleaseFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowScrollWheelFcn</td>
<td>''</td>
</tr>
<tr>
<td>WindowState</td>
<td>'normal'</td>
</tr>
<tr>
<td>WindowStyle</td>
<td>'normal'</td>
</tr>
</tbody>
</table>
Multiple Plots and Line Specifications

```matlab
x=0:pi/20:pi; y1=sin(x); y2=cos(x); plot(x,y1), hold on plot(x,y2) legend('sin', 'cos')
```

```matlab
x=0:pi/20:pi; y1=sin(x); y2=cos(x); plot(x,y1,x,y2) legend('sin', 'cos')
```

```matlab
x=0:pi/20:pi; y1=sin(x); y2=cos(x); plot(x,y1,x,y2, '-.sm') legend('sin', 'cos')
```

Line colors are automatically cycled

R2014b

Figure 1

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Adding Reference Lines

The Statistics and Machine Learning Toolbox features `refline` and `refcurve` for adding a straight line or polynomial in the current axis, `lsline` superimposing a least-squares line on each scatter plot, and `gline` drawing a line segment by clicking the pointer at the two endpoints.

What about basic MATLAB?

```matlab
function y=horline(Y,sp)
def='k--';                  % default settings
if nargin==0; Y=0; end % no input arguments
if nargin<=1; sp=def; end % less than two input arg.
linecolors='rgbcmykw';   % define line colors
[c,n]=intersect(sp,linecolors);  % find color spec.
if isempty(n), c=def(1); else, sp(n)=[]; end % no color
if isempty(sp), sp=def(2:3); end % no line style
yy=xlim;                 % find x-limits
line(yy,Y*[1 1],'Color',c,'LineStyle',sp)

a=randn(50,1); plot(a,'o')
horline(mean(a),'r')
horline(mean(a)+2*std(a),'b-.')
horline(mean(a)-2*std(a),'-b')
horline
```

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Legend and its Location

`legend(lbl1, ..., lbln, ["Location", Lcn])` adds a legend with `lbl1, ..., lbln` text strings in the specified location, `Lcn`, with respect to the axes, where `Lcn` may be one of the following character vectors (or string scalars):

- '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' ('swo') outside bottom left
- 'Best' ('b') least conflict with data in plot
- 'BestOutside' ('bo') least unused space outside plot
**Legend Options**

`legend(subset,labels,...)`

Only includes items in the legend for the data series listed in a vector of graphics objects `subset` (labels can be a single cell array of character vectors, a string array, or a character array labels)

`legend([GrObj1, GrObj2, ...],labels,...)`

If needed, you can create the group objects (which includes multiple graphics objects, as a child of the current axes) using the `hggroup` function

`legend(...,'Color','y','TextColor','r', 'NumColumns',2,'Box','off', 'Orientation','horizontal', 'AutoUpdate','off')`

Specifies label properties via name-value pair arguments

```matlab
h1=fplot(@sin,'-.'); hold
fplot(@(x)sin(x).*cos(x),':');
h2=fplot(@cos,'--');
legend([h1 h2],{"sin(x)" "cos(x)"})
hlgd=findall(gcf,'type','legend');
hlgd.AutoUpdate='off';
fplot(@(x)sin(x)+cos(x))
```
h1=fplot(@sin,'-.'); hold
fplot(@(x)sin(x).*cos(x),':');
h2=fplot(@cos,'--');
legend([h1 h2],{"sin(x)" "cos(x)"},'AutoUpdate','off');
fplot(@(x)sin(x)+cos(x))
hlgd=findall(gcf,'type','legend')
hlgd.Color='y';

Starting from R2017a, the legend automatically updates when you add or remove data series from the axes, unless... you set the 'AutoUpdate' property of the legend to 'off'.
Utilizing Two y-axes

Prior to R2014b

```matlab
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);
set(get(ax(1),'Ylabel'),'String','Short Period Motion (\alpha, ^o)')
set(get(ax(2),'Ylabel'),'String','Phugoid Motion (h, ft)')
set(h2,'LineStyle',':')
xlabel('Time, s')
title('T-37 Longitudinal Dynamics')
```

Starting from R2016a

```matlab
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);
yyaxis left, plot(x,y1); ylabel('Short Period Motion (\alpha, ^o)')
yyaxis right, plot(x,y2,:); ylabel('Phugoid Motion (h, ft)')
xlabel('Time, s')
title('T-37 Longitudinal Dynamics')
```
Finding a Graphics Object

```matlab
>> x = linspace(-pi,pi,50);
>> hp = plot(x,sin(x),'b-',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','se')
```

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

or
```
>> h2.Color='g';
```

Comments:

a) The `h=findobj` call returns handles of the root object and all its descendants
b) The `get(gcf,'children')` call returns 2-by-1 graphics array composed of the Legend and Axes objects
c) Try `allchild(gcf)`, `findall(gcf)`, `allchild(gca)`, and `findall(gca)`

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
The subplot Function

```matlab
subplot(2,2,1), ezplot('sin(x)')
subplot(2,2,2), ezplot('cos(x)')
subplot(223), ezplot('sin(x).^2')
subplot(224), ezplot('cos(x).^2')
```
More About the Grids

H=subplot(221); ezplot('sin(x)')

creates an Axes object H enabling modifying any property, e.g.

H.GridColor='r';

<table>
<thead>
<tr>
<th>Axes Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XTick, YTick, ZTick</td>
<td>Location of tick marks and major grid lines for each axis direction</td>
</tr>
<tr>
<td>XGrid, YGrid, ZGrid</td>
<td>Display of major grid lines for each axis direction</td>
</tr>
<tr>
<td>XMinorGrid, YMinorGrid, ZMinorGrid</td>
<td>Display of minor grid lines for each axis direction</td>
</tr>
<tr>
<td>LineWidth</td>
<td>Line width of grid lines, axes box outline, and tick marks</td>
</tr>
<tr>
<td>GridLineStyle</td>
<td>Major grid line style</td>
</tr>
<tr>
<td>MinorGridLineStyle</td>
<td>Minor grid line style</td>
</tr>
<tr>
<td>GridColor</td>
<td>Major grid line color</td>
</tr>
<tr>
<td>MinorGridColor</td>
<td>Minor grid line color</td>
</tr>
<tr>
<td>GridAlpha</td>
<td>Major grid line transparency</td>
</tr>
<tr>
<td>MinorGridAlpha</td>
<td>Minor grid line transparency</td>
</tr>
<tr>
<td>Layer</td>
<td>Location of grid lines in relation to the plotted data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PolarAxes Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ThetaTick, RTick</td>
<td>Location of tick marks and major grid lines for each axis direction</td>
</tr>
<tr>
<td>ThetaGrid, RGrid</td>
<td>Display of major grid lines for each axis direction</td>
</tr>
<tr>
<td>ThetaMinorGrid, RMinorGrid</td>
<td>Display of minor grid lines for each axis direction</td>
</tr>
<tr>
<td>LineWidth</td>
<td>Width of outline, tick marks, and grid lines</td>
</tr>
<tr>
<td>GridLineStyle</td>
<td>Major grid line style</td>
</tr>
<tr>
<td>MinorGridLineStyle</td>
<td>Minor grid line style</td>
</tr>
<tr>
<td>GridColor</td>
<td>Major grid line color</td>
</tr>
<tr>
<td>MinorGridColor</td>
<td>Minor grid line color</td>
</tr>
<tr>
<td>GridAlpha</td>
<td>Major grid line transparency</td>
</tr>
<tr>
<td>MinorGridAlpha</td>
<td>Minor grid line transparency</td>
</tr>
<tr>
<td>Layer</td>
<td>Location of grid lines in relation to the plotted data</td>
</tr>
</tbody>
</table>

For advanced users
%% Hourly air and dew point temperatures
A=[
    60 42
    63 42
    ...
];

%% Hourly sky condition
B=[
    "Fair"
    "Fair"
    "Fair"
    ...
];

C=nan(length(A),1); C(B~="Fair")=3;

stime=datenum('02-01-2018 00:54');
etime=datenum('02-03-2018 23:54');
t=linspace(stime,etime,3*24);
plot(t,A,'.-.',t,A(:,1)-A(:,2),'m.--'), grid, hold
bar(t,C,'y')
datetick('x','HH:MM PM')
xlabel('Time'), ylabel('Temperature, ^oF')
legend('Air','Dew point','\Delta T','Overcast','Location','E')
Variety of the 2-D Plots

- stem
- stairs
- loglog
- semilogx
- semilogy
- scatter
- errorbar
- polarplot
- compass
- polarhistogram
- polaraxes
- feather
- area
- pie
- fill
- rectangle

Introduced in R2016a
Introduced in R2016b

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Variety of the 2-D Plots

barh
histogram
plotmatrix
binscatter
histogram2
contour
ggeoplot
ggeobubble
triplot
stackedplot
parallelplot
heatmap
wordcloud

Introduced in R2018b
Introduced in R2019a
Introduced in R2017a
Introduced in R2017b
Introduced in R2015b

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
For advanced users

```
P=[-10000 1
-8000 5
-6500 5
...];

%% Defining axes using the subplot function
subplot 211
    plot(P(:,1),P(:,2)/1000,'o-.','markersize',3), grid, hold
    plot(P(end-2:end,1),P(end-2:end,2)/1000,'pr','markersize',6)
    xlabel('Year'), ylabel('Population, billion')
    xlim([-8000 2100])
    set(gca,'Color',[0.98 0.98 0.85]);

subplot 212
    plot(P(2:end,1),diff(P(:,2))./diff(P(:,1)),...
        'o-.','markersize',3), grid, hold
    plot(P(end-1:end,1),diff(P(end-2:end,2))./diff(P(end-2:end,1)),...
        'pr','markersize',6)
    xlim([-8000 2100])
    xlabel('Year'), ylabel('Yearly increase, million')
    set(gca,'Color',[0.98 0.98 0.85]);

%% Explicitly-defined axes positions (for two insert plots)
ax1=axes('Position',[0.2 0.68 0.5 0.22]);
semilogy(ax1,P(:,1),P(:,2)/1000,'o-.','markersize',2), grid, hold
semilogy(ax1,P(end-2:end,1),P(end-2:end,2)/1000,'pr','markersize',4)
xlim([-4000 2100]), ylabel('log scale, billion')

ax2=axes('Position',[0.2 0.20 0.5 0.22]);
semilogy(ax2,P(2:end,1),diff(P(:,2))./diff(P(:,1)),...
    'o-.','markersize',2), grid, hold
semilogy(ax2,P(end-1:end,1),diff(P(end-2:end,2))./diff(P(end-2:end,1)),...
    'pr','markersize',4)
xlim([-4000 2100]), ylabel('log scale, million')
```

---

For participation, in any form or by any means, contact the American Institute of
Aeronautics and Astronautics, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Overlaying One Plot over Another

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')
The Polar Plots

- `polarplot`
- `polarscatter`
- `polarhistogram`
- `compass`

**Introduced in R2016b**

- `rlim` sets or queries the r-axis limits for polar axes
- `rticks` sets or queries the r-axis tick values
- `rticklabels` sets or queries the r-axis tick labels
- `rtickformat` specifies the r-axis tick label format
- `rtickangle` rotates the r-axis tick labels
- `thetalim` sets or queries the theta-axis limits for polar axes
- `thetaticks` sets or queries the theta-axis tick values
- `thetaticklabels` sets or queries the theta-axis tick labels
- `thetatickformat` specifies the theta-axis tick label format
- `polaraxes` creates polar axes

```matlab
theta=0:0.01:2*pi;
rho=10*sin(.5*theta).*cos(.5*theta+pi/2);
pax=polaraxes;
polarplot(theta,rho,'-.','LineWidth',2)
pax.ThetaDir='clockwise';
pax.ThetaZeroLocation='top';
pax.FontSize = 12;
pax.ThetaColor='r';
rticks([1 5 9])
rticklabels({'r=1nmi','r=5nmi','r=9nmi'})
thetaticks(0:90:270)
thetaticklabels({'N','E','S','W'})
```
Two Types of 3-D Plots

Trajectories

t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t,'b')
grid on, axis square

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

Surfaces

[X,Y]=meshgrid(linspace(0,2*pi,100));
Z=sin(X).*cos(Y).^2;
plot3(X,Y,Z)
axis equal

view(az,el)
Variety of the Surface Plots

\[ [X,Y]=\text{meshgrid}(\text{linspace}(0,2\pi,100)); \ Z=\sin(X) .* \cos(Y); \]

\text{mesh}(X,Y,Z) \quad \text{meshc}(X,Y,Z) \quad \text{meshz}(X,Y,Z), \ hidden \ off

\text{surf}(X,Y,Z) \quad \text{surfc}(X,Y,Z) \quad \text{surfl}(X,Y,Z)

All these functions allow passing in just the grid vectors as the first two arguments (i.e. not necessarily a full grid matrices)
Variety of the Surface Plots

$[X,Y]=\text{meshgrid}(\text{linspace}(0,2*\pi,100)); Z=\sin(X) \cdot \cos(Y);$ 

ribbon($Y,Z$)  
waterfall($X,Y,Z$)  
tetramesh

timesh($\text{delaunay}(x,y)$) 

trimesh($\text{delaunay}(x,y)$) 
trisurf($\text{delaunay}(x,y)$) 
trisurf

Engineering Computations and Modeling in MATLAB/Simulink
Similarities in 2-D and 3-D

quiver

contour

bar

quiver3

contour3

bar3

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Linking Multiple Axes

Equating axes limits using the `axis` function

```matlab
[x,y]=meshgrid(-3:1/8:3); z=peaks(x,y);
subplot(231), meshc(x,y,z), title('meshc')
axis([-3 3 -3 3 -10 5]); v=axis; hidden off
subplot 232, meshz(x,y,z), axis(v); title('meshz')
ax=gca; ax.Clipping='off';
subplot 233, surfc(x,y,z,'FaceAlpha',0.7)
axis(v), title('surfc')
subplot 234, surfl(x,y,z), shading interp,
axis(v), title('surfl')
subplot 235, waterfall(x,y,z-5),
axis(v), title('waterfall')
subplot 236, ribbon(y,z)
```

Equating axes limits using the `linkaxes` function

```matlab
a(1)=subplot(211);
x=randi(19,100,1);
y=rand(100,1);
stem(x,y)
a(2)=subplot(212);
histogram(x)
linkaxes(a,'x')

Link x-axis only
'x'

Link y-axis only
'y'

Link x-axis and y-axis
'xy'

Remove linking
'off'
```

Also, see `linkdata` and `linkprop` functions
Shapes and Shades

The default is shading faceted
fill3 and patch Functions

```matlab
x = [0 1; 1 0; 0 1];
y = [1 0; 1 1; 0 1];
z = [1 1; 1 1; 0 1];
c = [1 0; 0 1; 1 1];
fill3(x, y, z, c)
xlabel('x'), ylabel('y')
```

```matlab
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
```

```matlab
x=[0 0 0 0; 0 1 1 0; 0 1 1 0]';
y=[0 1 1 0; 0 0 0 0; 1 1 1 1]';
z=[0 0 1 1; 0 0 1 1; 0 0 1 1]';
patch(x, y, z, -z)
axis square, view([35,35])
zlabel('z'), ylabel('y')
```

Jet

Parula

Jet

Parula

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Surface and Patch Objects

Lighting

```
ezsurf('sin(x^2+y^2)/(x^2+y^2)^0.5',pi*[-1 1 -1 1])
    shading interp
    light('Color','w','Position',[-4 0 0],'Style','local')
material shiny
xlabel('x axis'), ylabel('y axis'), zlabel('z axis')
```

camlight('headlight'), camlight('right'), camlight('left') - creates a light at the camera position, right and up from camera (default), and left and up from camera, respectively

camlight(az,el), lightangle(az,el) - creates a light at the specified azimuth (az) and elevation (el) with respect to the camera position

camlight(..., 'style') - defines the style argument using one of the following two values:

  * **local** (default) - the light is a point source that radiates from the location in all directions
  * **infinite** - the light shines in parallel rays

**lighting** flat - produces uniform lighting across each of the faces of the faceted object

**lighting** gouraud - calculates the vertex normals and interpolates linearly across the curved faces

**material** shiny - the object has a high specular reflectance relative to the diffuse and ambient light, and the color of the specular light depends only on the color of the light source

**material** dull - the object reflects more diffuse light and has no specular highlights, but the color of the reflected light depends only on the light source

**material** metal - the object has a very high specular reflectance, very low ambient and diffuse reflectance, and the color of the reflected light depends on both the color of the light source and the color of the object

For advanced users
Camera and Light Objects

For advanced users

orbit camera motion control
orbit scene light motion control
pan/tilt
move horizontally
move vertically
zoom
roll
selecting principal axis (x, y, z, no)
adding light
orthographic projection
perspective projection
resetting camera and scene light
stopping camera/light motion

camlight creates a light at the camera position (headlight, left, right)
lighting produces uniform lighting across each of the faces of the faceted object (flat, gouraud)
material sets the reflectance properties so that the object (shiny, dull, metal)
Using Different Color Maps and colormap Editor

```
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

>> colormappeditor
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 -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);
rx=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')

for i=1:1000
    rotate(haircraft,[25,35],1)
    pause(0.001)
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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Property Editor (before R2018b)
Saving Plots

```matlab
print ('-dbitmap', 'Name')
print ('-dtiff', 'Name')
print ('-dbitmap', 'Name')
print ('-dtiffn', 'Name')
print ('-djpeg', 'Name')
```
Accessing Graphics Properties via Property Editor
Graphics Objects Hierarchy

Root (Computer Screen)

Figure

UI Objects

Figure

Axes

Core Graphics Objects

Group Objects

Annotation Objects

Uicontrol

Uicontrols

Uibuttongroup

Uimenu

Uicontextmenu

Uipanel

Axes

Hidden Annotation Axes

Uimenu

Uicontrol

Image

Light

Line

Patch

Rectangle

Surface

Text

UI Objects

Prior to R2014b

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

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

```
get(h_line)
```

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

...
Exploiting Parent – Children Relationship

- **Figure**
- **Axis**
- **Line**

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

- `gcf` - Get the current figure handle
- `gca` - Get the current axes handle
- `h_figure = get(gca, 'parent')`
- `h_axis = get(gcf, 'children')`
- `h_line = get(gca, 'children')`
- `h_figure = get(0, 'children')`
- `h_figure = get(0, 'CurrentFigure')`

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) % Prior to R2014b
```

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

Starting from R2014b

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

% 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');
if char(k(1))=='Y'
    movie(F,10)
end
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)
xlabel('x_i'), ylabel('y_i'), zlabel('z_i');
R=eye(3);
h(1)=quiver3(0,0,0,R(1,1),R(1,2),R(1,3),'m','Linewidth',3);
h(2)=quiver3(0,0,0,R(2,1),R(2,2),R(2,3),'m','Linewidth',3);
h(3)=quiver3(0,0,0,R(3,1),R(3,2),R(3,3),'m','Linewidth',3);
ht(1)=text(R(1,1),R(1,2),R(1,3),'f{b_1}');
ht(2)=text(R(2,1),R(2,2),R(2,3),'f{b_2}');
ht(3)=text(R(3,1),R(3,2),R(3,3),'f{b_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',['\phi = ' int2str(phid) ' ^o']);
set(ha(2),'String',['\theta = ' int2str(thetad) ' ^o']);
set(ha(3),'String',['\psi = ' int2str(psid) ' ^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;

Replaced with angle2dcm in 2017a
Saving Animations

Setting the Renderer property to zbuffer or Painters works around limitations of getframe with the OpenGL renderer on some Windows systems.

```matlab
>> aviobj

aviobj =

VideoWriter

General Properties:
Filename:     'sin2cos.avi'
Path:         'C:\Users\Oleg\Documents\MATLAB'
FileFormat:   'avi'
Duration:      3.3333

Video Properties:
ColorChannels:            3
Height:                   429
Width:                    545
FrameCount:               100
FrameRate:                30
VideoBitsPerPixel:        24
VideoFormat:              'RGB24'
VideoCompressionMethod:   'Motion JPEG'
Quality:                  75

Methods
% 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(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)')

% Creating the Line object
h_line=get(gca,'children');

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

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

close(aviobj);
```
Creating Animated Lines

```matlab
x=linspace(0,4*pi,400);
y=sin(x)+0.5*rand(1,length(x));

h=animatedline('Marker','p','Color','b','LineStyle','-.');
axis([min(x),max(x),min(y),max(y)]), grid
xlabel("x-axis"), ylabel("y-axis")

for k = 1:length(x)
    addpoints(h,x(k),y(k));  
    drawnow
end
```

*Introduced in R2016b*
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
  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).
ellipsoid(0,0,0,2,10,1,40), hold
[x,y,z]=cylinder(0.5,40); axis equal
surf(0.8*x,3*y,1.5*z+0.5)
shading interp
light('Co','y','Po',[-4 0 -2],'St','lo')
material metal
set(gca,'Visible','off'), colormap jet

uicontrol('Style','pushbutton','String','Zoom In',...
    'Position',[20 20 80 30],...
    'Callback','if camva <= 1; return; else; camva(camva-1); end');
uicontrol('Style','pushbutton','String','Zoom Out',...
    'Position',[100 20 80 30],...
    'Callback','if camva >= 179; return; else; camva(camva+1); end');
E-Books and Code by Cleve Moler

Numerical Computing

Experiments with MATLAB

https://www.mathworks.com/moler.html

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, 12700 Sunrise Valley Drive, Suite 200, Reston, VA 20191-5807.
Requirement to Engineering Plots

- Choose the **type** of the plot and its **scale** wisely to have the best possible (that is, readable and understandable) representation of what you intend to show by this plot.
- Use a **title** to identify what the plot is, especially when there are multiple plots of a similar type.
- **Label** each axis and show the units used (place them in parentheses or after a comma right after the axis’ title and be consistent for all the plots).
- Add **annotations** including visualizing equations, indicating specific parameters, adding basic data statistics, as necessary.
- Assure that each axis has regularly spaced **tick marks** at convenient intervals to allow quick estimates of values.
- Distinguish each line not only by a different color (which MATLAB does for you automatically) but also by a **different line style** (and/or marker). While you can see the difference between the colored lines on the screen, it may disappear when the plot is printed out on a monochrome printer or converted into grayscale in a journal publication.
- Place a **legend** distinguishing and explaining different curves.
- Do not overload one plot by placing too much information on it, so if appropriate, use **multiple axes**.
- **Measured data** are usually represented by markers at each data point (different symbols then can be used for different sets of data). Connecting the data points assumes a relationship in data that may not exist.
- On the contrary, **equations** are usually plotted as lines, not as individual data points.
The End of Chapter 6

Questions?