Graphics

Basic Plotting Functions

Creating a Plot

`plot(y)`: y is a vector, create piecewise linear graph of the element of y versus the index of the elements of y. **`plot(x, y)`** produces a graph of y versus x.

`plot(x1, y1, x2, y2, ..., xn, yn)` plots multiple data sets in one graph.

Graph Style

`xlabel` create the x_axis label

`ylabel` like xlabel

`title` create the title

```
title('The title', 'FontSize', 12)
```

`legend` create the legend of each line

```
legend('Line one', 'Line two', ..., 'Line n')
```

`plot(x, y, ’color_style_marker’)` specifies the line style

```
plot(x, y, 'r:+') % plots the data using a red-dotted line and places a + marker at each data point.
```

`color` can be `c, m, y, r, g, b, w, k`

`line style` can be `';-';, ';--';, ';:';, ';-.';, no character`

`marker style` can be `';+';, ';o';, ';*';, ';x';, ';s';, ';d';, ';^';, ';v';, ';<;';, ';>;';, ';p';, ';h';, no character`

```
plot(x1, y1, 'r:', x2, y2, 'r+')
```

Graphing Imaginary and Complex Data

When you pass complex values as arguments to **`plot`**, MATLAB ignores the imaginary part, except when you pass a single complex argument. For this special case, the command is a shortcut for a graph of the real part versus the imaginary part. Therefore,

```
plot(Z)
```

where Z is a complex vector or matrix, is equivalent to

```
plot(real(Z), image(Z))
```

Adding Plots to an Existing Graph

```
hold on
```

Figure Windows

```
figure(n) % make an existing figure window the current figure
figure % create a new figure window and make it the current figure
clf reset % reset the properties you set at the previous plot
```

Displaying Multiple Plots in One Figure

```
subplot(m ,n, p) % partitions the figure window into an m-by-n matrix of small subplots and selects the pth subplot for the current plot.
```

Controlling the Axes

Axis Limits

use the functions **`axis`**, **`xlim`**, **`ylim`**, **`zlim`**

```
axis([xmin, xmax, ymin, ymax, zmin, zmax])
axis auto % enable automatic limit selection
axis square % makes the x-axis and y-axis the same length
axis equal % makes the individual tick mark increments on the x-axes and y-axes the same length.
axis auto normal % default automatic mode
axis on % visible
axis off % invisible
grid on % grid line on
grid off % grid line off
```

Axis Labels and Titles

```
text(0.5,-1/3,'{\itNote the odd symmetry.}')
```

see also `annotation` function.

Creating Mesh and Surface Plots

About Mesh and Surface Plots

- **`mesh`** produces wireframe surfaces that color only the lines connecting the defining points.
- **`surf`** displays both the connecting lines and the faces of the surface in color.

Visualizing Functions of Two Variables

To display a function of two variables, $z = f(x, y)$,

1. Generate X and Y matrices consisting of repeated rows and columns, respectively, over the domain of the function.
2. Use X and Y evaluate and graph the function.

```
x = 1:3;
y = 1:5;
[X,Y] = meshgrid(x,y);
[X, Y] = meshgrid(-8:.5:8);
R = sqrt(X .^ 2 + Y .^ 2) + eps;
Z = sin(R) ./ R;
mesh(X, Y, Z)
```

基于矢量 x 和 y 中包含的坐标返回二维网格坐标。X 是一个矩阵，每一行是 x 的一个副本；Y 也是一个矩阵，每一列是 y 的一个副本。坐标 X 和 Y 表示的网格有 **`length(y)`** 个行和 **`length(x)`** 个列。

Color Surface

```
[X, Y] = meshgrid(-8:.5:8);
R = sqrt(X .^ 2 + Y .^ 2) + eps;
Z = sin(R) ./ R;
colormap hsv
colorbar
```

Making Surface Transparent

```
alpha(.4)
```

Illuminating Surface Plots with Lights

```
surf(X,Y,Z,'FaceColor','red','EdgeColor','none')
camlight left;
lighting phong
```

Display Images

Printing Graphics

```
print -dpng magicsquare.png
set(gcf, 'PaperPositionMode', 'auto')
print -dpng -r0 magicsquare.png
print -dtiff -r200 magicsquare.tiff
```

Working with Graphics Objects

```
y = [75 91 105 123.5 131 150 179 203 226 249 281.5];
bar(y,'FaceColor','green','EdgeColor','black','LineWidth',1.5)
```

Access Object Properties

```
x = 1:10;
y = x.^3;
h = plot(x,y);
h.Color = 'red'; % h = plot(x,y,'Color','red');
h.LineWidth
get(h)
    AlignVertexCenters: 'off'
            Annotation: [1×1 matlab.graphics.eventdata.Annotation]
          BeingDeleted: 'off'
            BusyAction: 'queue'
         ButtonDownFcn: ''
              Children: [0×0 GraphicsPlaceholder]
              Clipping: 'on'
                 Color: [1 0 0]
             CreateFcn: ''
             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: [1×1 Axes]
         PickableParts: 'visible'
              Selected: 'off'
    SelectionHighlight: 'on'
                   Tag: ''
                  Type: 'line'
         UIContextMenu: [0×0 GraphicsPlaceholder]
              UserData: []
               Visible: 'on'
                 XData: [1 2 3 4 5 6 7 8 9 10]
             XDataMode: 'manual'
           XDataSource: ''
                 YData: [1 8 27 64 125 216 343 512 729 1000]
           YDataSource: ''
                 ZData: [1×0 double]
           ZDataSource: ''
set(h)
    AlignVertexCenters: {'on'  'off'}
            BusyAction: {'queue'  'cancel'}
         ButtonDownFcn: {}
              Children: {}
              Clipping: {'on'  'off'}
                 Color: {1×0 cell}
             CreateFcn: {}
             DeleteFcn: {}
           DisplayName: {}
      HandleVisibility: {'on'  'callback'  'off'}
               HitTest: {'on'  'off'}
         Interruptible: {'on'  'off'}
              LineJoin: {'chamfer'  'miter'  'round'}
             LineStyle: {'-'  '--'  ':'  '-.'  'none'}
             LineWidth: {}
                Marker: {1×14 cell}
       MarkerEdgeColor: {'auto'  'none'}
       MarkerFaceColor: {'auto'  'none'}
         MarkerIndices: {}
            MarkerSize: {}
                Parent: {}
         PickableParts: {'visible'  'none'  'all'}
              Selected: {'on'  'off'}
    SelectionHighlight: {'on'  'off'}
                   Tag: {}
         UIContextMenu: {}
              UserData: {}
               Visible: {'on'  'off'}
                 XData: {}
             XDataMode: {'auto'  'manual'}
           XDataSource: {}
                 YData: {}
           YDataSource: {}
                 ZData: {}
           ZDataSource: {}
```

```
figure
y = magic(5);
h = plot(y) % h is a array of five graphic object

prop_name(1) = {'Marker'};
prop_name(2) = {'MarkerFaceColor'};

prop_values(1,1) = {'s'};
prop_values(1,2) = {h(1).Color};
prop_values(2,1) = {'d'};
prop_values(2,2) = {h(2).Color};
prop_values(3,1) = {'o'};
prop_values(3,2) = {h(3).Color};
prop_values(4,1) = {'p'};
prop_values(4,2) = {h(4).Color};
prop_values(5,1) = {'h'};
prop_values(5,2) = {h(5).Color};

set(h,prop_name,prop_values)
```

Passing Argument

```
function plotFunc(x)
    y = 1.5*cos(x) + 6*exp(-.1*x) + exp(.07*x).*sin(3*x);
    ym = mean(y);
    hfig = figure('Name','Function and Mean');
    hax = axes('Parent',hfig);
    plot(hax,x,y)
    hold on
    plot(hax,[min(x) max(x)],[ym ym],'Color','red')
    hold off
    ylab = hax.YTick;
    new_ylab = sort([ylab, ym]);
    hax.YTick = new_ylab;
    title ('y = 1.5cos(x) + 6e^{-0.1x} + e^{0.07x}sin(3x)')
    xlabel('X Axis')
    ylabel('Y Axis')
end

x = -10:.005:40;
plotFunc(x)
```

Finding the Handle of Existing Objects

Finding All Object of a Certain Type

```
h = findobj('Type','patch');
```

Finding Objects with Particular Property

```
plot(rand(5),'r:')
h = findobj('Type','line','Color','r','LineStyle',':');
```