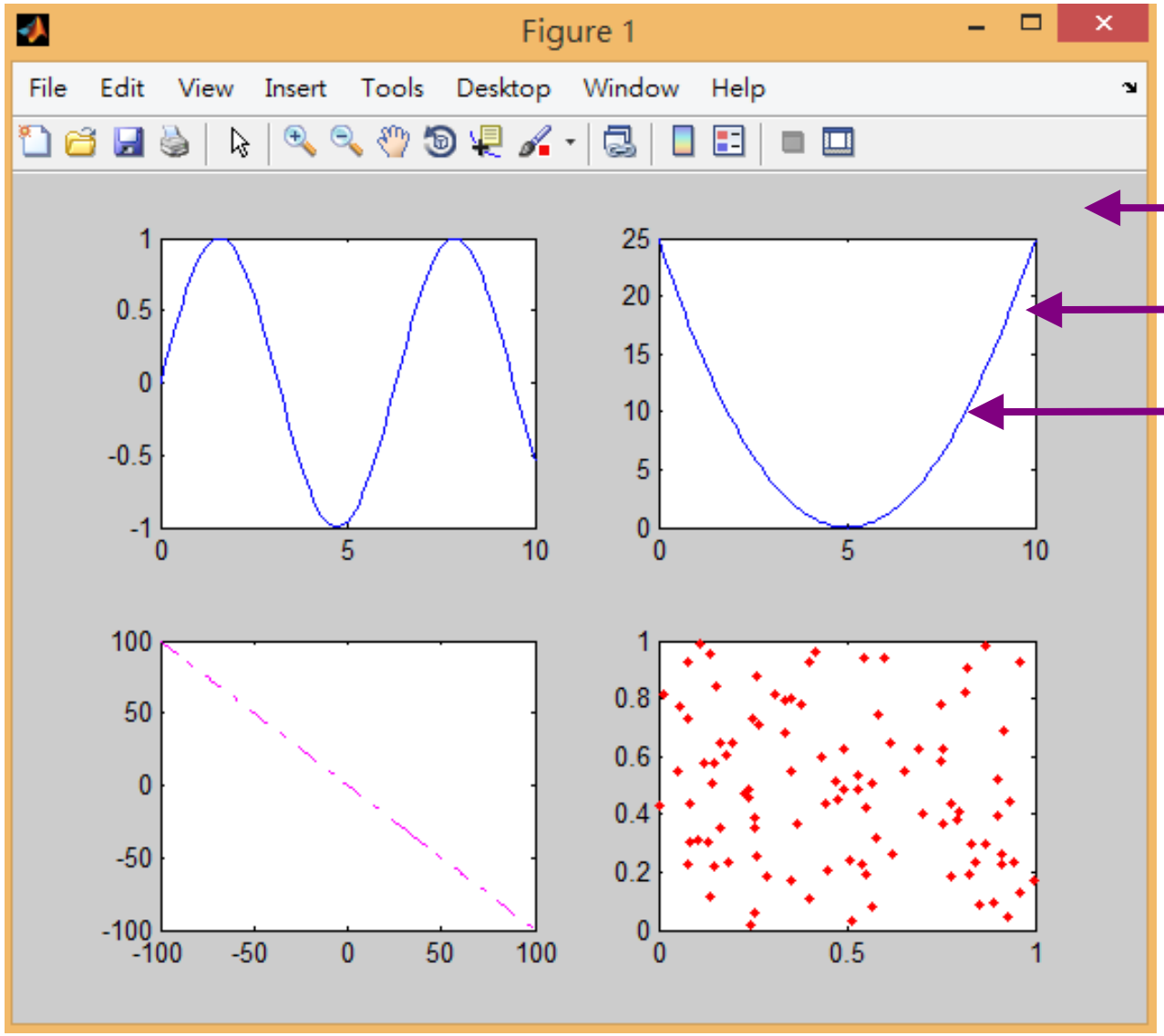
Data Visualization: Plots and Images (1)

Graph Objects

Basic concepts: Three levels of graph objects in MATLAB:

- **Figure**: Each figure appears in a separate **figure window**. All other graph objects have to appear in figures.
- **Axes**: An axes specifies a particular coordinate system for drawing the data.
  - An "axes" determines where to place plots and other graph objects.
  - The display of an axes can be in 2-D (default) or 3-D. (An axes is intrinsically 3-D. When it's displayed in 2-D, the third dimension is just perpendicular to the screen.)
- **Plots**: A "plot" is a particular visualization of a set of data, such as a curve. Each plot has to be drawn within an **axes**.

Graph Objects

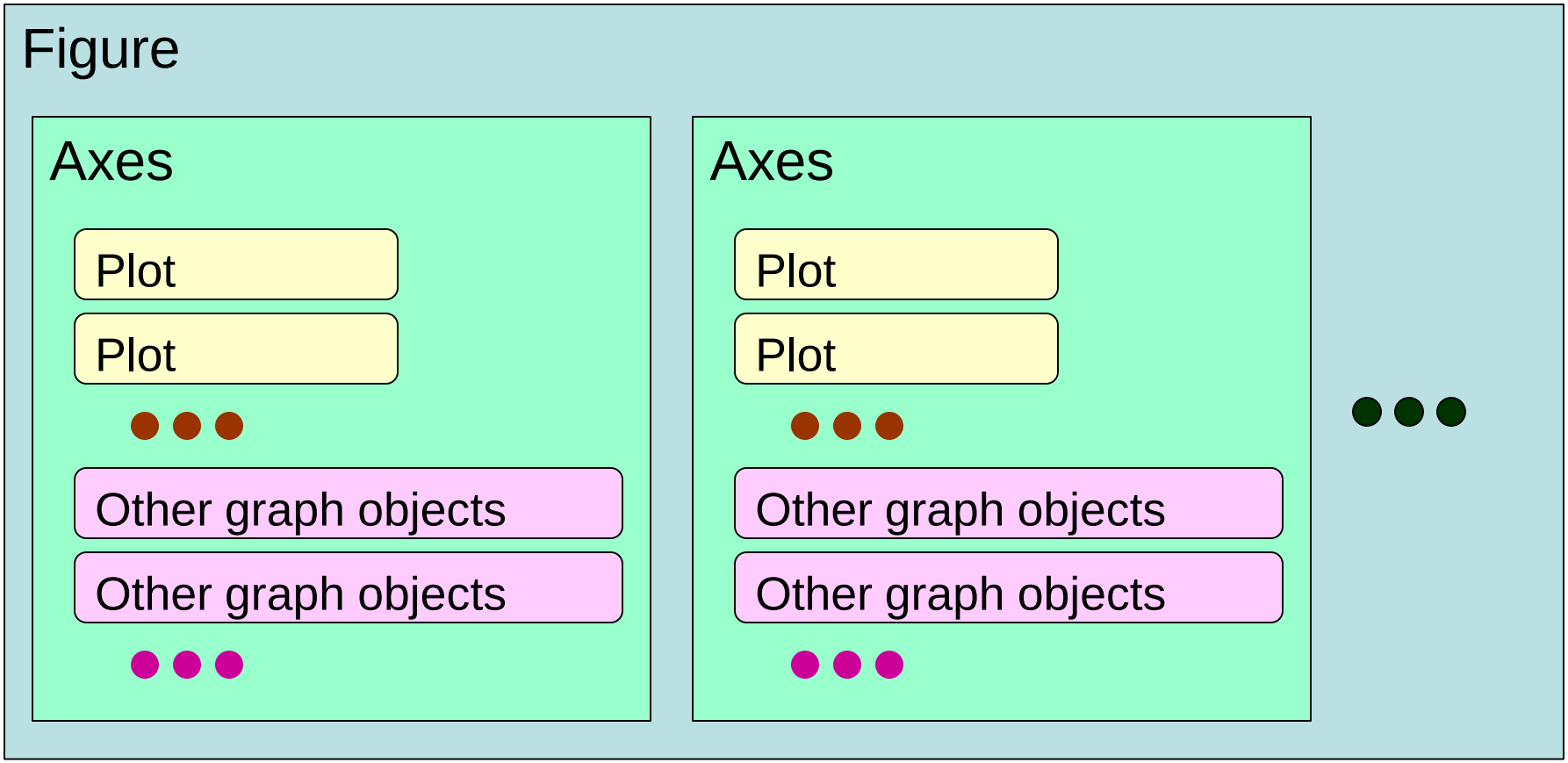


Figure

Axes

Plot (many types)

Graph Objects Hierarchy



"Other graph objects" can be texts or annotations you want to add, such as figure titles, legends, arrows, etc.

Creating/Accessing Graph Objects

- An object is returned when created explicitly (such as by functions **`figure`**, **`axes`**, **`plot`**, etc.). Keep this object if you may need to modify the object later.
- When not specified, graph operations are applied to the **current** (most recently used) **figure** and **axes**.
  - Functions **`gcf`** and **`gca`** return the object that is the **current figure** and **current axes**, respectively.
  - A **container object** is automatically created for an operation if needed. (Example: A **figure** and an **axe**s are created if you just call **`plot`**.)
  - A figure can also be identified with a positive integer. This is most useful when specifying figures using **`figure(n)`**.

Multiple Axes in a Figure

- Specifically call **`axes`** for creating each axes:
  - Manually controlled position/size for each axes. The positions/sizes are relative to the containing figure.
- Call function **`subplot`** before drawing operations:
  - Automatically controlled position/size for each axes.

Basic 2-D Plots

- Basic form: `plot(X,Y)`
  - Here `X` and `Y` are vectors of the same length.
  - If `X` is omitted (called with syntax `plot(Y)`), MATLAB generates `X` using `1:length(Y)`.
  - This function creates what is called a **line plot** (the default plot type) in MATLAB.
- Specifying the axes when plotting: `plot(ax, X, Y)`
  - This syntax applies to other plotting functions as well.

Basic 2-D Plots

We can set basic plot properties within the call to **`plot`**:

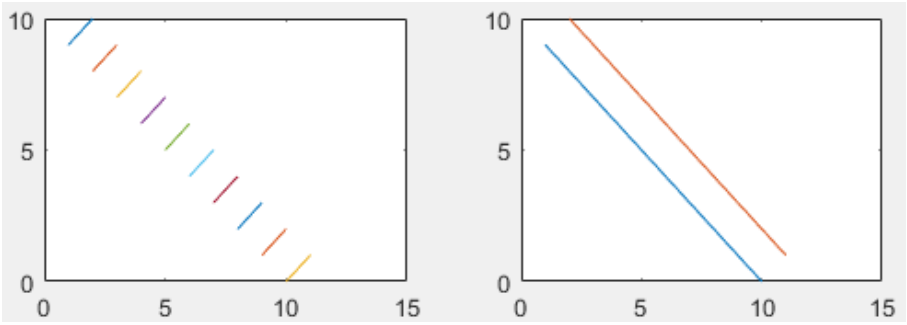
- Example: **`plot(X,Y,'o-m')`**
- Setting basic colors
- Setting line types
- Setting marker types
- Draw scatter plots by specifying marker types but no line types.
- The use of name-value pairs of properties:
  - Examples: **`'linewidth'`**, **`'color'`**, …
  - Many more; check MATLAB documentation
  - Specifying RGB colors

Multiple 2-D Plots in One Axes

Drawing multiple plots on the same axes:

- Method #1: Use **`X`** and **`Y`** that are 2-D arrays of the same size.
  - Each corresponding pair of column vectors in **`X`** and **`Y`** will become a line plot.
  - Line colors are assigned automatically if not specified.
  - Example codes:

```
x=1:10; y=10-x;
plot([x;x+1],[y;y+1]);
plot([x;x+1]',[y;y+1]');
```



- Method #2: Use **`hold on`** for the axes:
  - **`hold on/off`**: Whether old contents are retained when drawing new plots in the current axes.

Axis Properties of 2-D Plots

*Note: Do not confuse the terms **axis** and **axes** in Matlab plots!*

Controlling the axes:

- Setting axis ranges:
  - **`axis([xmin xmax ymin ymax])`**
  - **`axis auto/manual/tight`**
- Axis/box visibility:
  - **`axis on/off`**
  - **`box on/off`** (the outer box)
- Aspect ratio: **`axis normal/equal/square`**
- Direction (where the origin is): **`axis xy/ij`**

Adding Text in 2-D Plots

Marking the axes:

- Marking on **x** and **y** axis:
  - Functions **`xlabel`** and **`ylabel`**:
  - Use **`set(gca, …)`** with name-value pairs:
    - Properties: **`'xtick'`**, **`'xticklabel'`**, **`'ytick'`**, **`'yticklabel'`**, …
- Function **`title`**:
- Function **`legend`**: For drawing the legend
- Function **`text`**: For drawing texts
  - Properties: **`'fontname'`**, **`'fontweight'`**, **`'fontsize'`**, **`'color'`**, **`'verticalalignment'`**, **`'horizontalalignment'`**, …

Graph Object Properties

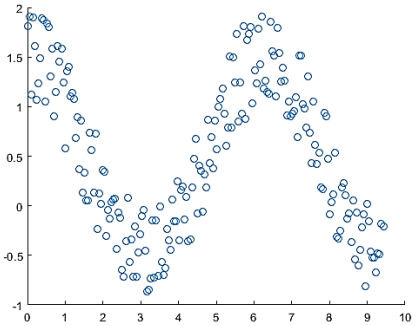
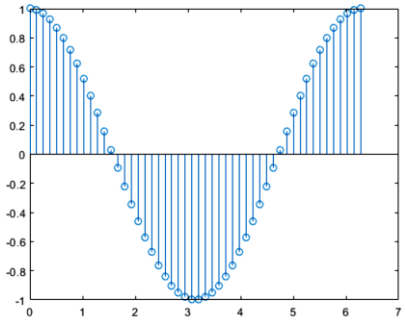
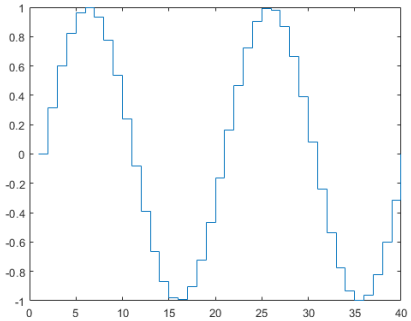
- The various graph objects have many properties that can be queried and set.
  - Too many to list them here; check the documentation.
- Many properties can be set with name-value pairs during or after the object's creation.
- To query the properties of an existing object:
  - `get(object, name-value pairs …)`
- To set the properties of an existing object:
  - `set(object, name-value pairs …)`

Additional 2-D Plot Types

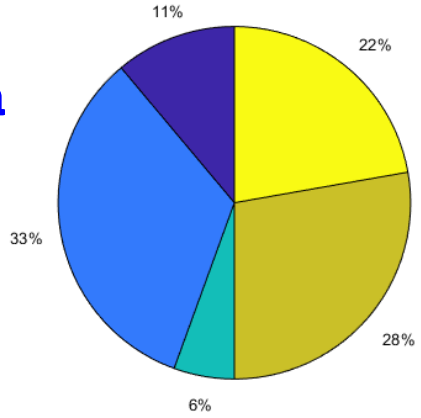
- Plot in polar coordinates: Function **`polar`**
- Log-scale x-y plots:
  - Function **`loglog`**:
  - Functions **`semilogx`** and **`semilogy`**:
  - These functions are used in place of **`plot`**.
- Two **y** axis on the left and right sides; there are actually two overlapping **axes**:
  - Function: **`plotyy(X1,Y1,X2,Y2)`**

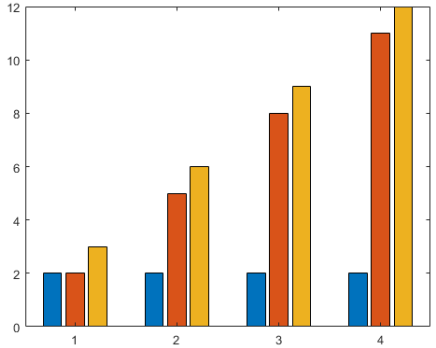
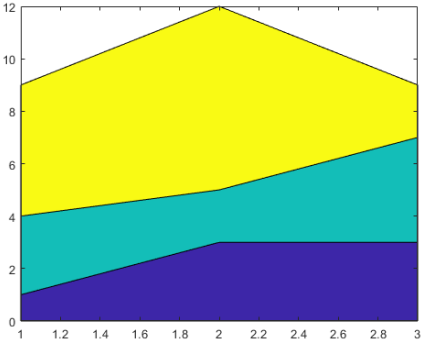
Additional 2-D Plot Types

- For discrete numerical data:
  - Functions **`scatter`**, **`stem`** and **`stair`**:

- Additional plot types:
  - **`pie`**
  - **`bar`**, **`barh`**
  - **`area`**



Function Plots

- While we can always plot a function $f(x)$ by sampling it over some points of x, MATLAB also provides some methods for very convenient visualization of functions:
  - Function **`func`**: Plot curves of $y=f(x)$ in 2-D.
  - Function **`fimplicit`**: Plot curves in 2-D corresponding to points satisfying $f(x,y)=0$.
  - Function **`func3`**: Plot parametric curves in 3-D defined by $(x(t), y(t), z(t))$.
- We will demonstrate these functions when we get to talk about **function handles** later in the semester.