Python Matplotlib

Han-Wei Shen
The Ohio State University
First Thing First

• Install the Matplotlib package if you have not had it already
  – Typically Matplotlib is installed together with numpy and scipy
  – Example: Use Anaconda Python
    (https://store.continuum.io/cshop/anaconda/)
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

```python
import matplotlib.pyplot as plt
import numpy as np
```

```python
plt.figure()
plt.plot(np.array[4,6,3,10,7])
plt.xlabel('input parameter (seconds)')
plt.show()
plt.savefig('myplot.png')
```
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

```python
import matplotlib.pyplot as plt
import numpy as np
```

```python
plt.figure()
# default argument is empty, but you use a number, a string, etc. as the input
plt.plot(np.array([4,6,3,10,7]))
# use line graph here
plt.xlabel('input parameter (seconds)')
# label the x axis
plt.show()
# draw to a window
plt.savefig('myplot.png')
# save to a file
```
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

    import matplotlib.pyplot as plt
    import numpy as np

• Create a figure as the drawing area. You can have multiple figures in a program

    plt.figure()  // default argument is empty, but you use a number, a string, etc. as the input

    plt.plot(np.array[4,6,3,10,7])  // use line graph here

    plt.xlabel('input parameter (seconds)')  // label the x axis

    plt.show()  // draw to a window

    plt.savefig('myplot.png')  // save to a file
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

  import matplotlib.pyplot as plt
  import numpy as np

• Create a figure as the drawing area. You can have multiple figures in a program

  plt.figure() // default argument is empty, but you use a numbr, a string, etc. as the input

  plt.plot(np.array[4,6,3,10,7]) // use line graph here
  plt.xlabel('input parameter (seconds)') // label the x axis
  plt.show() // draw to a window
  plt.savefig('myplot.png') // save to a file
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

```python
import matplotlib.pyplot as plt
import numpy as np
```

• Create a figure as the drawing area. You can have multiple figures in a program

```python
plt.figure()  # default argument is empty, but you use a number, a string, etc. as the input
```

• Plot the data using a plot type of your choice (line graph, bar chart, histogram, pie chart, etc)

```python
plt.plot(np.array[4,6,3,10,7])  # use line graph here
```
A Typical Matplotlib Program

- Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)
  
  ```python
  import matplotlib.pyplot as plt
  import numpy as np
  ```

- Create a figure as the drawing area. You can have multiple figures in a program
  
  ```python
  plt.figure()  // default argument is empty, but you use a numbr, a string, etc. as the input
  ```

- Plot the data using a plot type of your choice (line graph, bar chart, histogram, pie chart, etc)
  
  ```python
  plt.plot(np.array([4,6,3,10,7]))  // use line graph here
  ```

- Add title, labels, tickmarks, legends, etc.
  
  ```python
  plt.xlabel('input parameter (seconds)')  // label the x axis
  ```
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

```python
import matplotlib.pyplot as plt
import numpy as np
```

• Create a figure as the drawing area. You can have multiple figures in a program

```python
plt.figure()  # default argument is empty, but you use a numbr, a string, etc. as the input
```

• Plot the data using a plot type of your choice (line graph, bar chart, histogram, pie chart, etc)

```python
plt.plot(np.array([4,6,3,10,7]))  # use line graph here
```

• Add title, labels, tickmarks, legends, etc.

```python
plt.xlabel(‘input parameter (seconds)’)  # label the x axis
```

• Display the plot either to an interactive window or a file

```python
plt.show()  # draw to a window
plt.savefig(‘myplot.png’)  # save to a file
```
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

```python
import matplotlib.pyplot as plt
import numpy as np
```

• Create a figure as the drawing area. You can have multiple figures in a program

```python
plt.figure()  # default argument is empty, but you use a numbr, a string, etc. as the input
```

• Plot the data using a plot type of your choice (line graph, bar chart, histogram, pie chart, etc)

```python
plt.plot(np.array([4,6,3,10,7]))  # use line graph here
```

• Add title, labels, tickmarks, legends, etc.

```python
plt.xlabel('input parameter (seconds)')  # label the x axis
```

• Display the plot either to an interactive window or a file

```python
plt.show()  # draw to a window
plt.savefig('myplot.png')  # save to a file
```
A Typical Matplotlib Program

• Start your Python program by including matplotlib and numpy packages (numpy arrays are typically used as the input to matplotlib)

  import matplotlib.pyplot as plt
  import numpy as np

• Create a figure as the drawing area. You can have multiple figures in a program

  plt.figure()  // default argument is empty, but you use a numbr, a string, etc. as the input

• Plot the data using a plot type of your choice (line graph, bar chart, histogram, pie chart, etc)

  plt.plot(np.array([4,6,3,10,7]))  // use line graph here

• Add title, labels, tickmarks, legends, etc.

  plt.xlabel('input parameter (seconds)')  // label the x axis

• Display the plot either to an interactive window

  plt.show()  // draw to a window
  plt.savefig('myplot.png')  // save to a file
Display Multiple Plots

• Use ‘subplot()’ command to display multiple plots together

```python
plt.subplot(211)  # create a 2x1 grid for subplots, and you will draw on the first subplot
plt.plot(np.array([4, 6, 3, 10, 7]))  # use line graph here
plt.subplot(212)
plt.plot(np.random.uniform(10, 15, 20))
plt.show()
```
Display Multiple Plots

• Use ‘subplot()’ command to display multiple plots together

```python
plt.subplot(211)  # create a 2x1 grid for subplots, and you will draw on the first subplot
plt.plot(np.array([4,6,3,10,7]))  # use line graph here
plt.subplot(212)
plt.plot(np.random.uniform(10,15,20))
plt.show()
```
Display Multiple Plots

- Use ‘subplot()’ command to display multiple plots together

```python
plt.subplot(211)  # create a 2x1 grid for subplots, and you will draw on the first subplot
plt.plot(np.array([4,6,3,10,7]))  # use line graph here
plt.subplot(212)
plt.plot(np.random.uniform(10,15,20))
plt.show()
```
Tick Marks, Labels, and Grid

• You can customize tick marks, labels, and grid

```python
plt.plot(np.random.uniform(10,15,20))
plt.xticks(np.arange(1,20))
plt.ylabel('random')
plt.grid()
plt.show()
```
Tick Marks, Labels, and Grid

- You can customize tick marks, labels, and grid

```python
plt.plot(np.random.uniform(10,15,20))
plt.xticks(np.arange(1,20))
plt.ylabel('random')
plt.grid()
plt.show()
```
Tick Marks, Labels, and Grid

• You can customize tick marks, labels, and grid

```python
plt.plot(np.random.uniform(10,15,20))
plt.xticks(np.arange(1,20))
plt.ylabel('random')
plt.grid()
plt.show()
```
Multiple Graphs and Legend

• You can put multiple graphs in a single plot and add legend

```python
x = np.arange(-2*np.pi, 2*np.pi, 0.1)
plt.plot(x, np.sin(x), label='graph1')
plt.plot(x, np.cos(x), label='graph2')
plt.legend()
plt.show()
```
Multiple Graphs and Legend

• You can put multiple graphs in a single plot and add legend

```python
x = np.arange(-2*np.pi, 2*np.pi, 0.1)
plt.plot(x, np.sin(x), label='graph1')
plt.plot(x, np.cos(x), label = 'graph2')
plt.legend()
plt.show()
```
Bar Charts

- Use the function `bar(xlist, ylist)` to draw a bar chart

```python
x = [1, 2, 3, 4, 5]
y = [7, 2, 3, 5.5, 1]
plt.xticks(x, ['b1', 'b2', 'b3', 'b4', 'b5'], rotation=-60)
plt.bar(x, y, align='center')
plt.grid()
plt.show()
```
Pie Chart

• Use the function `pie(list, labels)` to plot a pie chart

```python
a = np.random.uniform(1,10,7)
L = ['b1', 'b2', 'b3', 'b4', 'b5', 'b6', 'b7']
plt.pie(a, labels = L)
plt.title('A pie chart example')
plt.show()
```
Histogram

- Use the function `hist(values, bins)` to display a histogram for a distribution

```python
x = np.random.uniform(0,0.1,1000)  // create a normal distribution
plt.hist(x,100)  // 100 bins
plt.show()
```
x = np.random.uniform(1,100,50)
y = np.random.normal(20, 10, 50)
plt.scatter(x,y)
plt.show()

x = np.random.uniform(1,100,50)
plt.boxplot(x)
plt.show()
Display Images

# load image data
data = np.fromfile('HeadMRVolume.raw', dtype='uint8')
data = data.reshape(42, 62, 48)

# plot slices
plt.imshow(data[20, :, :])
plt.gca().invert_yaxis()
plt.show()

plt.imshow(data[:, 20, :])
plt.gca().invert_yaxis()
plt.show()

plt.imshow(data[:, :, 20])
plt.gca().invert_yaxis()
plt.show()