Matplotlib Python Module

Computing for Data Analytics (CPSC 4800)

Mourad Bouguerra
mbouguerra@langara.ca

Langara College

June, 2022

Lesson's Outline

Learning Objectives

Learning Objectives

✎ Upon completion of this lesson, you will be able to

- ❒ visualize your data using different types of plots
- ❒ setup plot labels, title and legend
- ❒ apply different styles to your plot

Learning Objectives

Learning Objectives

✎ Upon completion of this lesson, you will be able to

- ❒ visualize your data using different types of plots
- ❒ setup plot labels, title and legend
- ❒ apply different styles to your plot

Learning Objectives

Learning Objectives

✎ Upon completion of this lesson, you will be able to

- ❒ visualize your data using different types of plots
- ❒ setup plot labels, title and legend
- ❒ apply different styles to your plot

Learning Objectives

Learning Objectives

✎ Upon completion of this lesson, you will be able to

- ❒ visualize your data using different types of plots
- ❒ setup plot labels, title and legend
- ❒ apply different styles to your plot

NumPy Methods

Using arange() Method

- NumPy method
  - arange(start,stop,step) returns
    - ✔ evenly spaced values over an open interval
    - ✔ including start
    - ✔ excluding stop

```
[x for x in np.arange(start=0,stop=1,step=.1)]
```

Using linspace() Method

- NumPy method
  - linspace(start,stop,num) returns
    - ✔ evenly spaced num values over a closed interval
    - ✔ including both start and open
- The step between the generated list item is computed as follows
  - $step = \frac{stop - start}{num - 1}$

```
[x for x in np.linspace(start=0,stop=1,num=11)]
```

Class Activity

- What is the output of the following Python script?

```
import numpy as np
x = [ w for w in np.linspace(start=-10,stop=10,num=5)]
print(x)
```

Chinese Proverb

Tell Me & I Forget,
Teach Me & I Remember,
Involve Me & I Learn

Class Activity

- What is the output of the following Python script?

```python
import numpy as np
x = [w for w in np.linspace(start=-2,stop=2,num=5)]
print(x)
```

Chinese Proverb

I Hear & I Forget, I See & I Remember, I Do & I Understand

Using Matplotlib

Matplotlib Python Module

- Matplotlib is a comprehensive Python module for creating
  - static and interactive data visualization

Matplotlib Installation

- Matplotlib can be installed from terminal using one of the following
  - command-lines

```
python -m pip install matplotlib
pip install matplotlib
conda install matplotlib
```

Using Matplotlib

Matplotlib Python Module

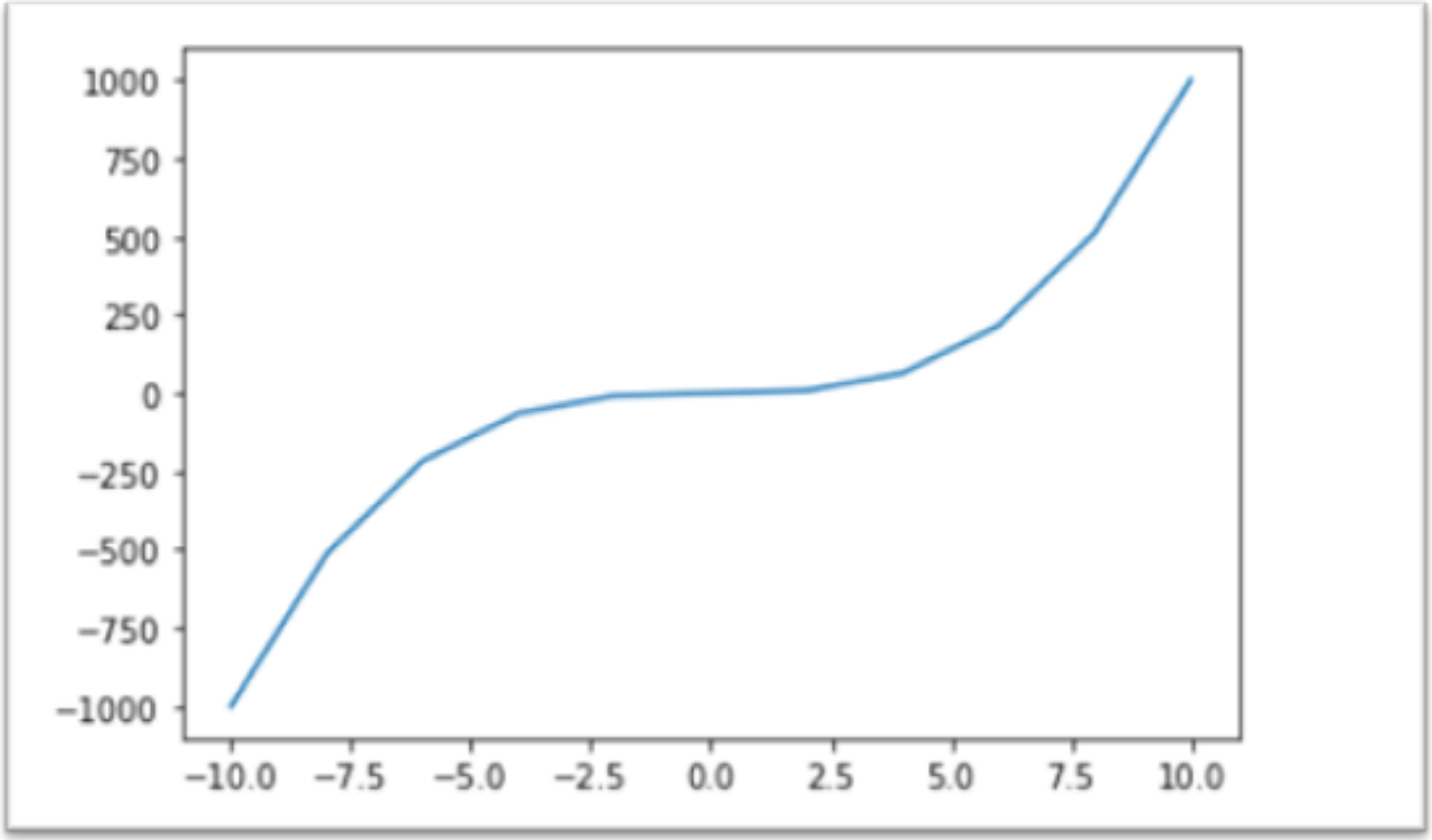
- To automatically display a static plot in a Jupyter notebook use
  - %matplotlib inline
- To automatically display an interactive plot in a Jupyter notebook use
  - %matplotlib notebook

Matplotlib Python Module

- To display a plot
  - x and y should be specified
- In Python code, you have to call
  - plt.show() method

```
%matplotlib inline
x = np.linspace(start=-10,stop=10,num=11)
y = x**3
plt.plot(x,y)
```

Static Grid

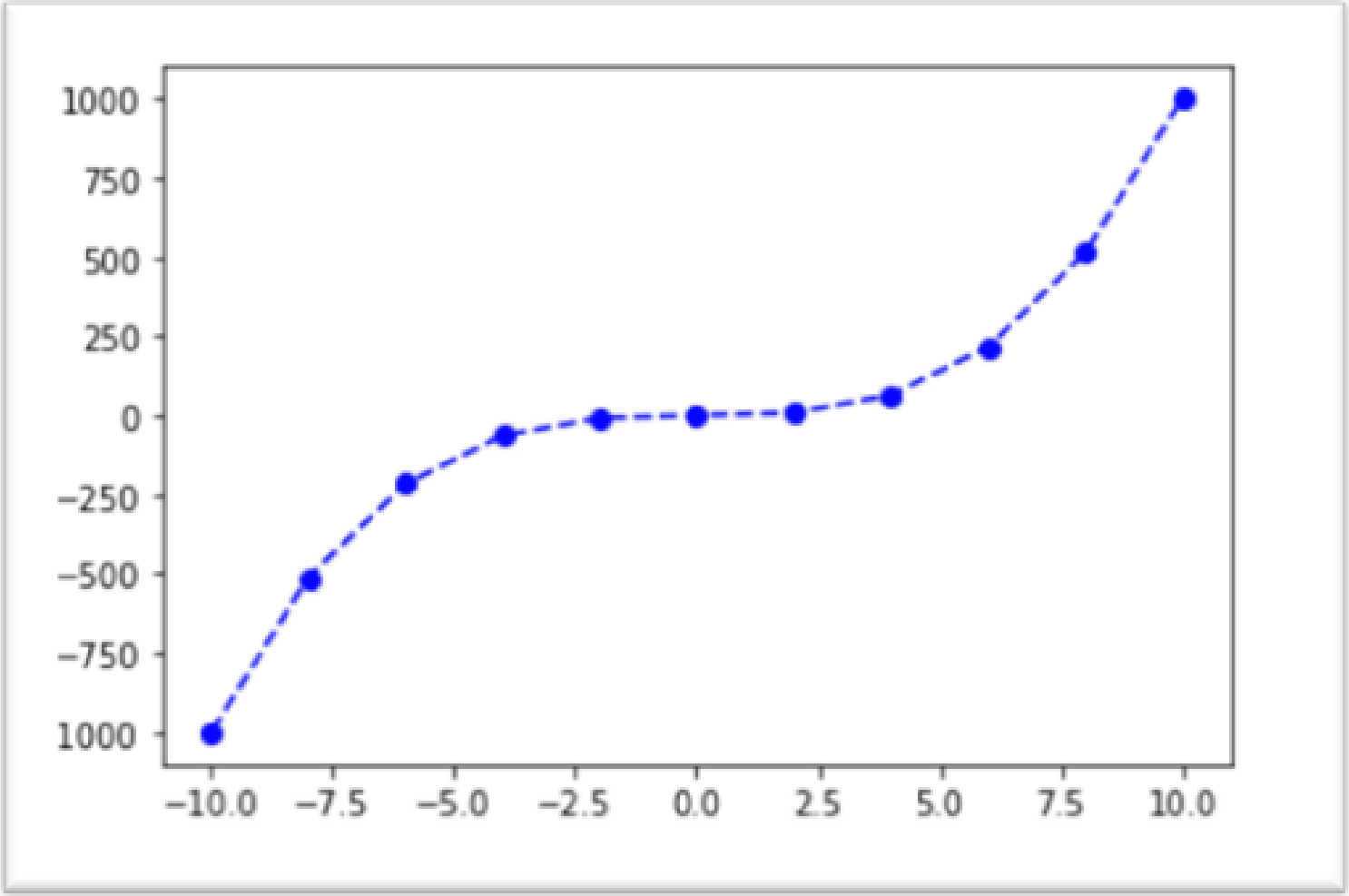


Matplotlib Python Module

- You can also specify
  - Color, marker, and line style

```
%matplotlib inline
x = np.linspace(start=-10,stop=10,num=11)
y = x**2
plt.plot(x,y,color='b',linestyle ='--',marker='o')
```

Static Plot

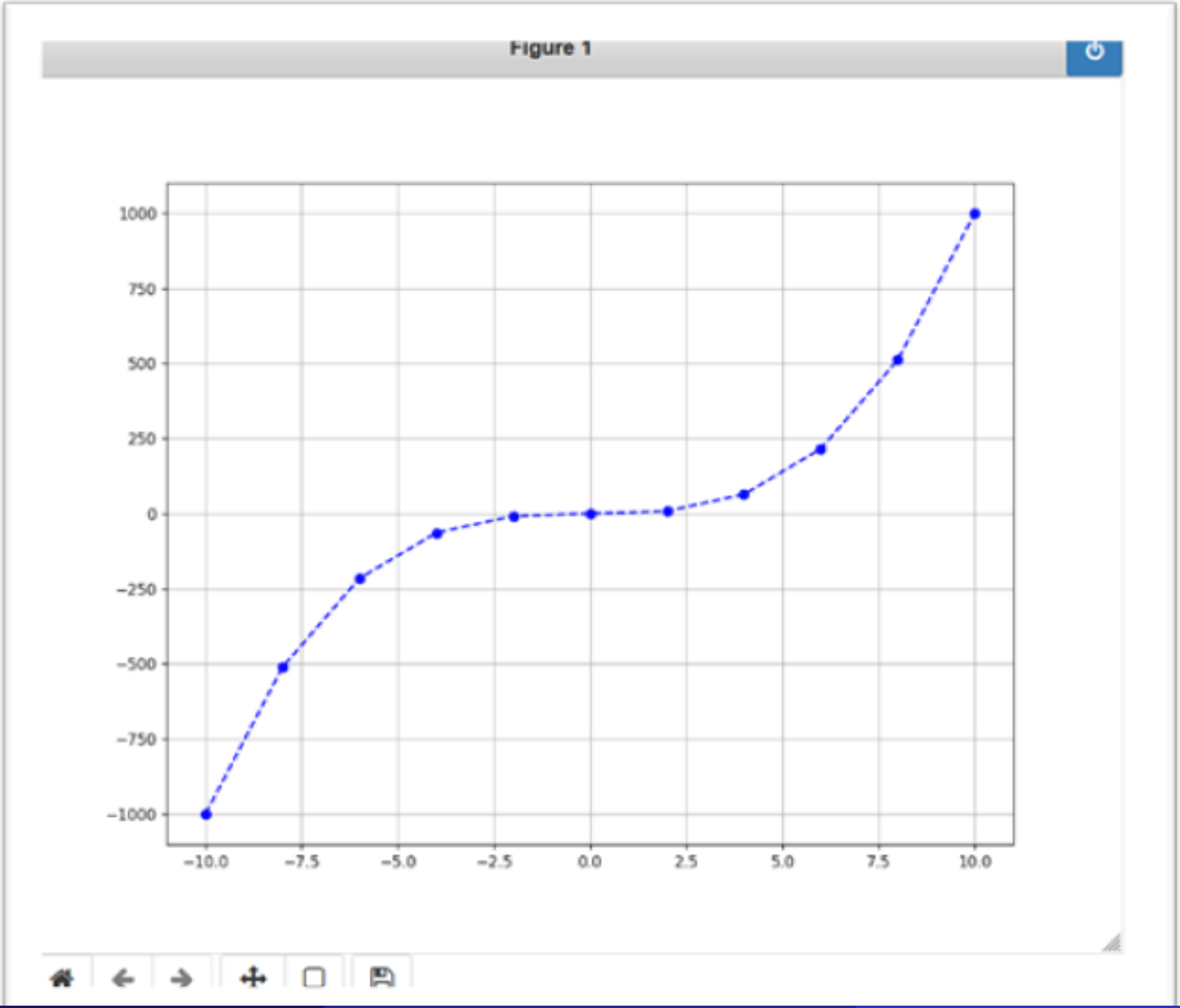


Matplotlib Python Module

- You can also specify
  - Color, marker, and line style
- You add a grid

```
%matplotlib notebook
x = np.linspace(start=-10,stop=10,num=11)
y = x**3
plt.plot(x,y,color='b',linestyle ='--',marker='o')
plt.grid()
```

Interactive Gride



Using Matplotlib

Matplotlib Python Module

- Different type of plots
  - line
  - scatter
  - histogram
  - boxplot
  - bar

Using Matplotlib

```
# default is a line plot
plt.plot(np.random.randn(100))
# scatter plot requires x and y
plt.scatter(range(100),np.random.randn(100))
# histogram
plt.hist(np.random.randn(100))
# boxplot
plt.boxplot(np.random.randn(100))
# bar plot requires height
data = pd.Series(np.random.randint(1,10,size=(100,)))
data_frequency = data.value_counts()
plt.bar(x=range(len(data_frequency)),height=data_frequency)
```

Matplotlib Colors

Matplotlib Python Module

- The color parameter can be specified by
  - using basic color names
  - using hexadecimal string
  - using grayscale number between .0 and .1

```
plt.plot(x,y,color='red')
plt.plot(x,y,color='r')
plt.plot(x,y,color='#ee0903')
plt.plot(x,y,color='.65')
```

Matplotlib Buitin Basic Colors

Matplotlib Basic Colors

Color	Shorthand
red	r
green	g
blue	b
cyan	c
magenta	m
yellow	y
black	k
white	w

Matplotlib Markers

Matplotlib Python Module

- All possible Matplotlib markers are defined in

  ➡ https://matplotlib.org/stable/api/markers_api.html

```
plt.plot(x,y,marker='.')
plt.plot(x,y,marker='o')
plt.plot(x,y,marker='v')
plt.plot(x,y,marker='^')
```

Matplotlib Markers

marker	symbol	description
"."	•	point
",."	·	pixel
"o"	●	circle
"v"	▼	triangle_down
"^"	▲	triangle_up
"<"	◀	triangle_left
">"	▶	triangle_right

Figure: `https://matplotlib.org/stable/api/markers_api.html`

Matplotlib Line Styles

Matplotlib Python Module

- All possible Matplotlib line styless are defined in

  ➡ https://matplotlib.org/3.5.0/gallery/lines_bars_and_markers/linestyles.html

```
plt.plot(x,y,linestyle='solid')
plt.plot(x,y,linestyle='dotted')
plt.plot(x,y,linestyle='dashed')
plt.plot(x,y,linestyle='-')
plt.plot(x,y,linestyle=':')
plt.plot(x,y,linestyle='--')
```

Matplotlib Simple Line Styles

Matplotlib Simple Line Styles

Line Style	Shorthand
`solid`	`-`
`dotted`	`:`
`dashed`	`-`
`dashdot`	`-.`

Matplotlib Title and Labels

Matplotlib Python Module

- To add title and labels for x and y axes
  - plt.title()
  - plt.xlabel()
  - plt.ylabel()

```
plt.title(r'$y=x^2$')
plt.xlabel('x')
plt.ylabel('y')
```

Matplotlib Title and Labels

Matplotlib Python Module

- To add a legend
  - add a label argument to each plot
  - use plt.legend() method

```
x = np.linspace(start=-10,stop=10,num=11)
y = x**2
plt.plot(x,x,color='y',marker='o',linewidth=5,label='identity')
plt.plot(x,y,color='#1122ff',marker='v',label='Quadratic')
plt.title(r'$y=x^2$')
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc=1)
plt.style.use('dark_background')
plt.grid()
```

Matplotlib Legend

Location String	Location Code
'best'	0
'upper right'	1
'upper left'	2
'lower left'	3
'lower right'	4
'right'	5
'center left'	6
'center right'	7
'lower center'	8
'upper center'	9
'center'	10

Figure: `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.legend.html`

Matplotlib Plot Styles

Matplotlib Python Module

- To get all available styles

```
from matplotlib import style
print(plt.style.available)
```

- To change to a given style

```
plt.style.use('dark_background')
```

- To change back to a default style

```
plt.style.use('default')
```

Matplotlib Plot Styles

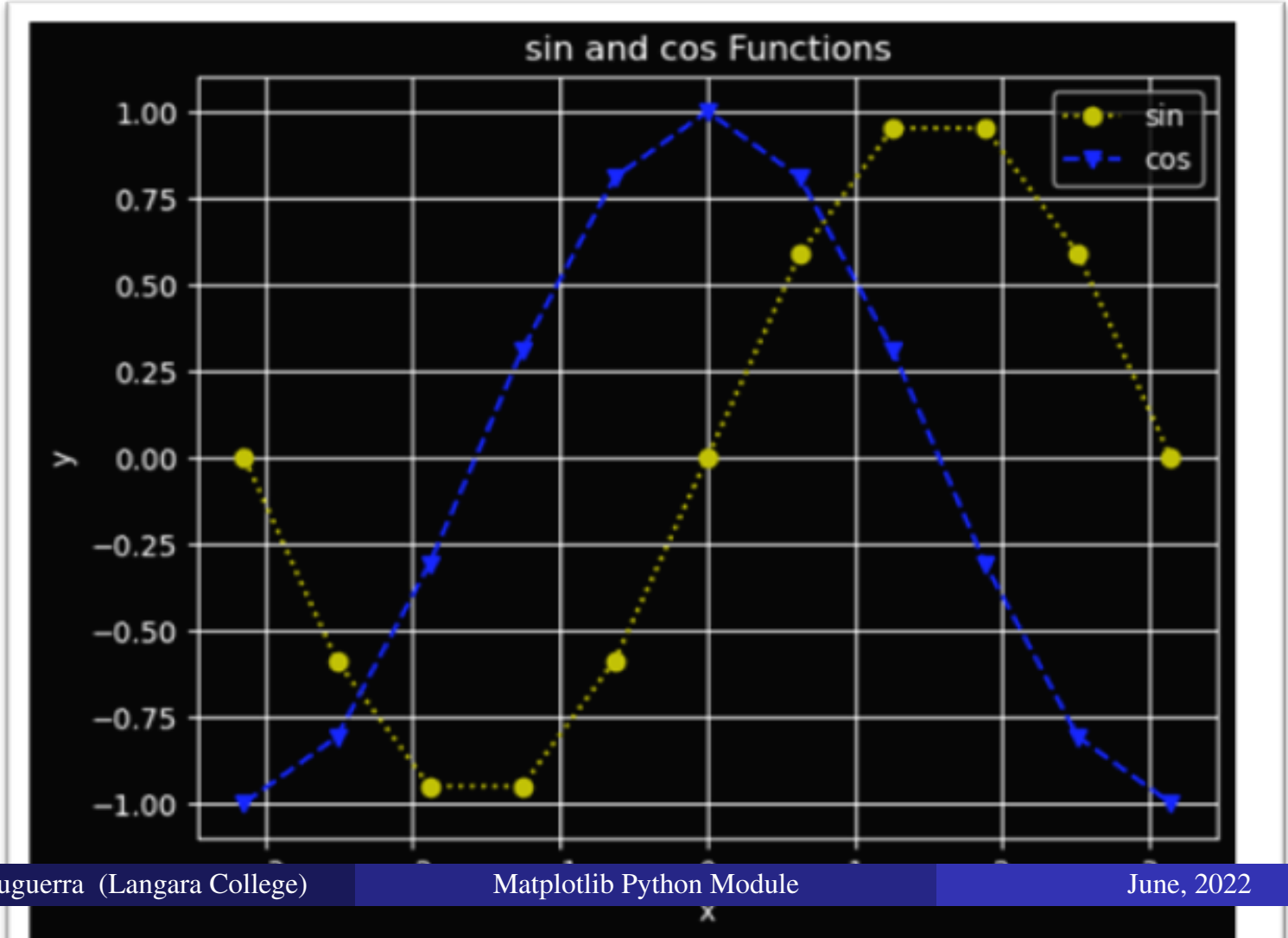
Matplotlib Python Module

- All the Matplotlib available styles are

  - **Solarize_Light2**
  - **_classic_test_patch**
  - **bmh**
  - **classic**
  - **dark_background**
  - **fast**
  - **fivethirtyeight**
  - **ggplot**
  - **grayscale**
  - **seaborn**
  - **seaborn-bright**
  - **seaborn-colorblind**
  - **seaborn-dark**
  - **seaborn-dark-palette**
  - **seaborn-darkgrid**
  - **seaborn-deep**
  - **seaborn-muted**
  - **seaborn-notebook**
  - **seaborn-paper**
  - **seaborn-pastel**
  - **seaborn-poster**
  - **seaborn-talk**
  - **seaborn-ticks**
  - **seaborn-white**
  - **seaborn-whitegrid**
  - **tableau-colorblind10**

Class Activity

- Plot the sin and cos functions over the interval $[-\pi, +\pi]$ as shown in the following figure?



Using Matplotlib Figure Object

Matplotlib Python Module

- To add a figure object

```
fig = plt.figure(figsize=(10,8),dpi=300)
```

- To setup a four subplots in the figure
  - in a 2x2 layout

```
ax1 = fig.add_subplot(2,2,1)
ax2 = fig.add_subplot(2,2,2)
ax3 = fig.add_subplot(2,2,3)
ax4 = fig.add_subplot(2,2,4)
```

Using Matplotlib Figure Object

Matplotlib Python Module

- To setup a four subplots in the figure
  - in a 2x2 layout

```
ax1 = fig.add_subplot(2,2,1)
ax2 = fig.add_subplot(2,2,2)
ax3 = fig.add_subplot(2,2,3)
ax4 = fig.add_subplot(2,2,4)
```

- To add four subplots to the figure

```
plt.style.use('Solarize_Light2')
ax1.hist(np.random.randn(100), bins=20, color='m', alpha=0.3,edgecolor='black')
ax2.scatter(np.arange(30), np.arange(30) + 3 * np.random.randn(30),alpha=.4)
ax3.plot(x,y,color='y',marker='>',linewidth=2)
ax4.plot(x,x,color='y',marker='s',linewidth=3)
fig
```

Using Matplotlib Figure Object

Matplotlib Python Module

- To save a Matplotlab plot use
  - savefig() method

```
x = np.linspace(start=-10,stop=10,num=11)
y = x**3
plt.grid()
plt.plot(x,y,color='b',linestyle =':',marker='^',
    markersize=10,alpha=.4)
plt.savefig('results/cubic.png')
plt.savefig('results/cubic.pdf')
```

Pandas Scatter Plot

- Pandas uses the plot() method to create different plots by specifying
  - the kind parameter

```
movies = pd.read_csv('data/movies.csv')
long_movies = movies.sort_values(by='duration',
    ascending=False)[0:10]
long_movies.plot(x='title',y='duration',kind='scatter'
    ,marker='s',color='b',alpha=.3,figsize=(10,8),s=50)
plt.title('Top 10 Longest Movies')
plt.xticks(rotation=90)
plt.grid()
plt.show()
```

Class Activity

- Produce a scatter plot for the top 10 short movies?

Chinese Proverb

Tell Me & I Forget,
Teach Me & I Remember,
Involve Me & I Learn

Pandas Histogram Plot

- Pandas uses the plot() method to create different plots by specifying
  - the kind parameter

```
movies.plot(y='duration',kind='hist', alpha=.3,figsize
    =(10,8),bins=20,edgecolor='red')
# or
movies.duration.plot(kind='hist', alpha=.3,figsize=
    (10,8),bins=20,edgecolor='red')
plt.grid()
plt.show()
```

Pandas Boxplot

- Pandas uses the plot() method to create different plots by specifying
  - the kind parameter

```
movies.boxplot('duration', figsize=(10,8))
plt.title('Duration Boxplot')
plt.grid()
plt.show()
```

Pandas Barplot

- Pandas uses the plot() method to create different plots by specifying
  - the kind parameter

```
long_movies = movies.sort_values(by='duration',
    ascending=False)[1:10]
long_movies.plot(x='title',y='duration',kind='bar',
    color='b',alpha=.3,figsize=(10,8))
plt.title('Top 10 Longest Movies')
plt.xticks(rotation=90)
plt.grid()
plt.show()
```

Pandas Horizontal Barplot

- Pandas uses the plot() method to create different plots by specifying
  - the kind parameter

```
long_movies = movies.sort_values(by='duration',
    ascending=False)[1:10]
long_movies.plot(x='title',y='duration',kind='barh',
    color='b',alpha=.3,figsize=(10,8))
plt.title('Top 10 Longest Movies')
plt.xticks(rotation=90)
plt.grid()
plt.show()
```