Matplotlib Tutorial

CS-328 Numerical Methods for Visual Computing and Machine Learning

Ekrem Fatih Yilmazer and Lovro Nuic
Slides by: Jan Bednarik and Tizian Zeltner

September 25, 2025

Introduction

About Matplotlib:

- Python open-source library for potting 2D/3D plots.
- Originally designed as a free substitute of Matlab.
- No need to memorize all the commands - use doc and stackoverflow.

Introduction

About Matplotlib:

- Python open-source library for potting 2D/3D plots.
- Originally designed as a free substitute of Matlab.
- No need to memorize all the commands - use doc and stackoverflow.

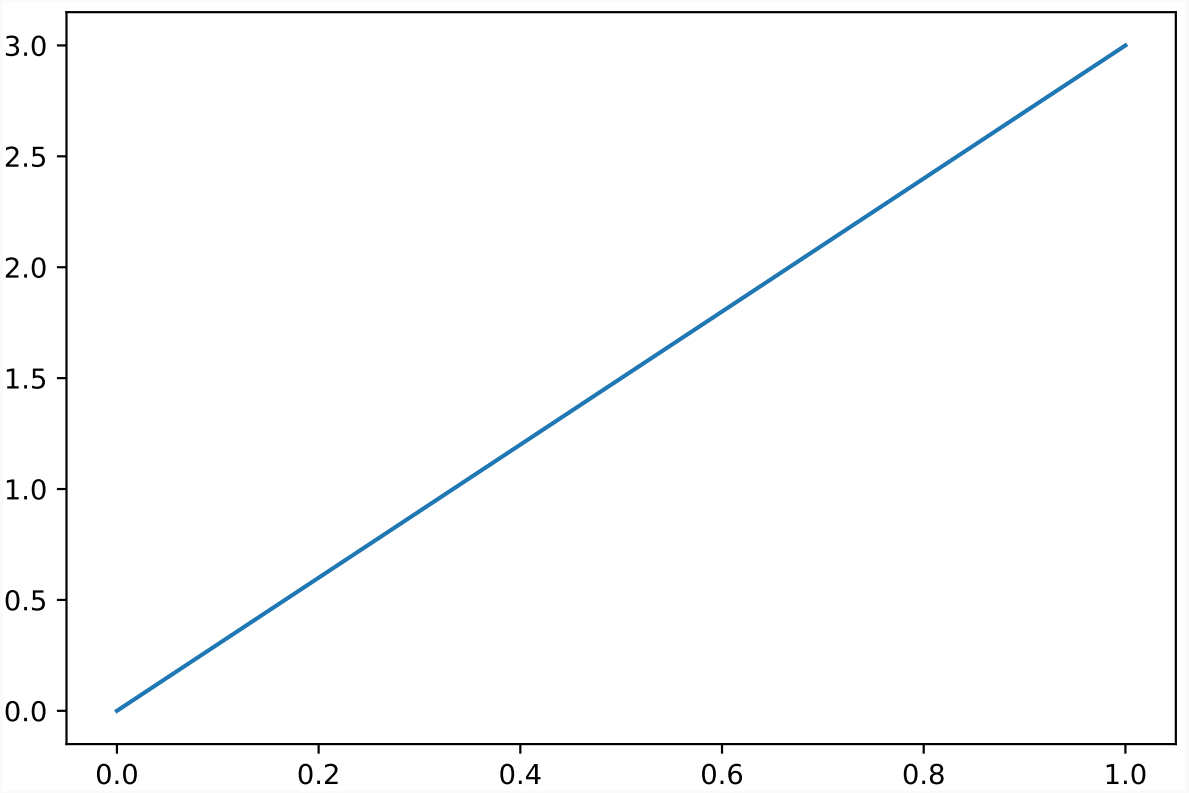
Introduction

About Matplotlib:

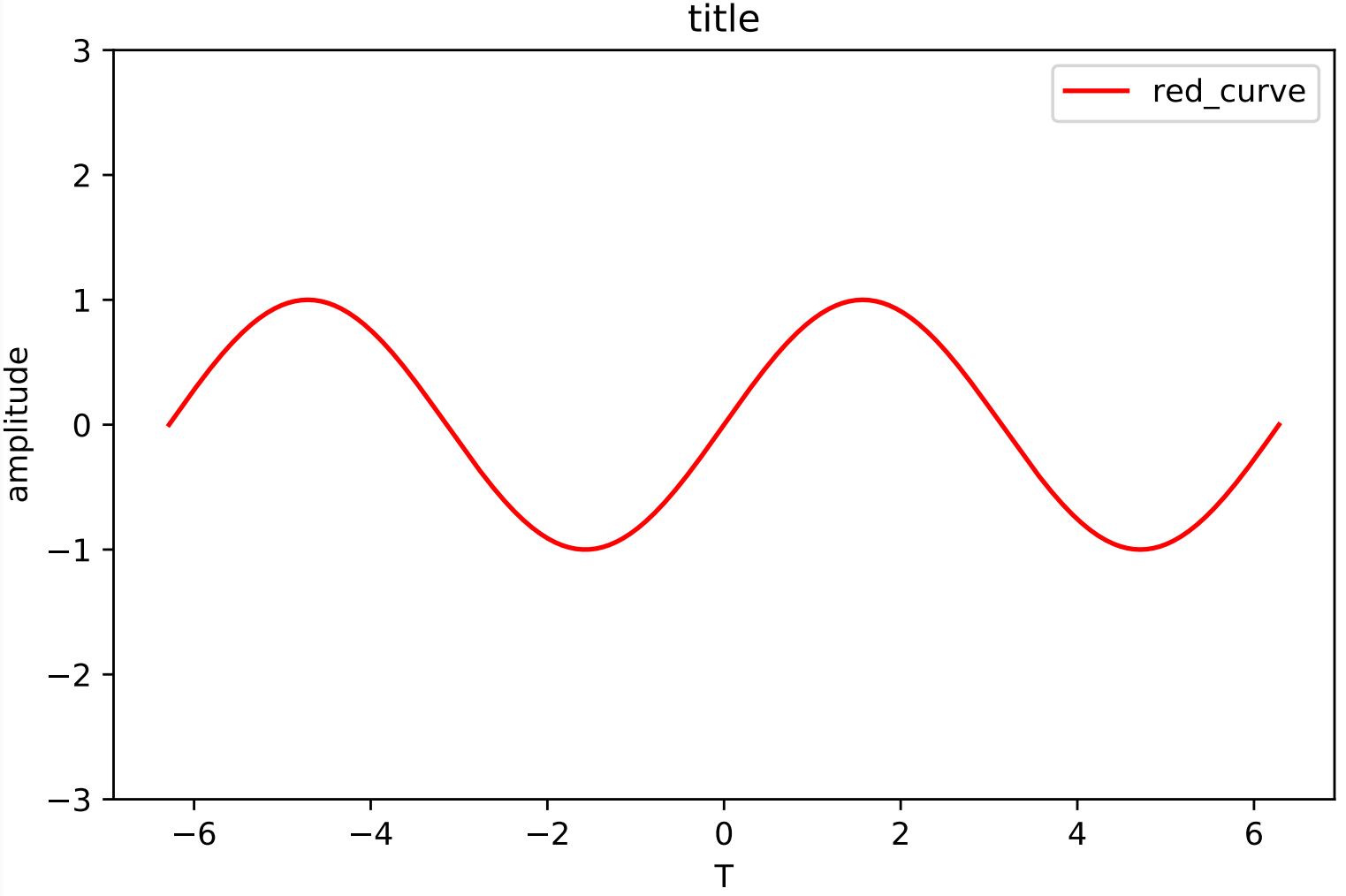
- Python open-source library for potting 2D/3D plots.
- Originally designed as a free substitute of Matlab.
- No need to memorize all the commands - use doc and stackoverflow.
- ChatGPT can write code for you.
- Different APIs - we use `pyplot` .

'Hello World' Plot

```
%matplotlib inline
import matplotlib.pyplot as plt
plt.plot([0, 3])  # Plots a line y = 3x in range [0, 1]
```



Basic Plotting Pipeline - 2D line example



Basic Plotting Pipeline - 2D line example

```
# 1) Prepare data.
x = np.linspace(-2.0 * np.pi, 2.0 * np.pi, 1000)
y = np.sin(x)
```

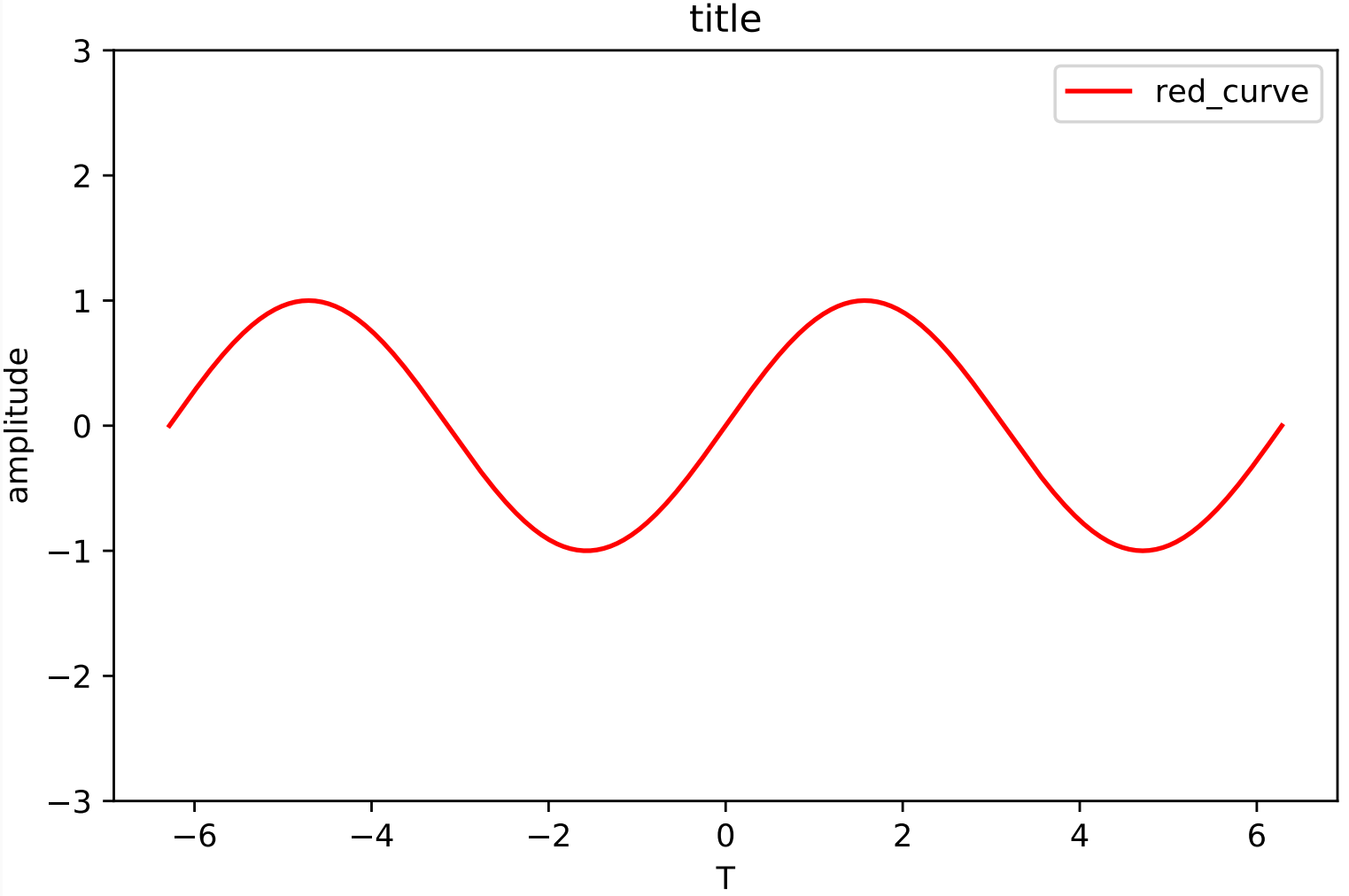
```
# 2) Add a title.
plt.title('title')
```

```
# 4) Plot data
plt.plot(x, y)
```

```
# 3) Adjust axes (scale, labels, limits, ...)
plt.xlabel('T')
plt.ylabel('amplitude')
plt.ylim([-3., 3.])
```

```
# 5) Add a legend.
plt.legend()
```

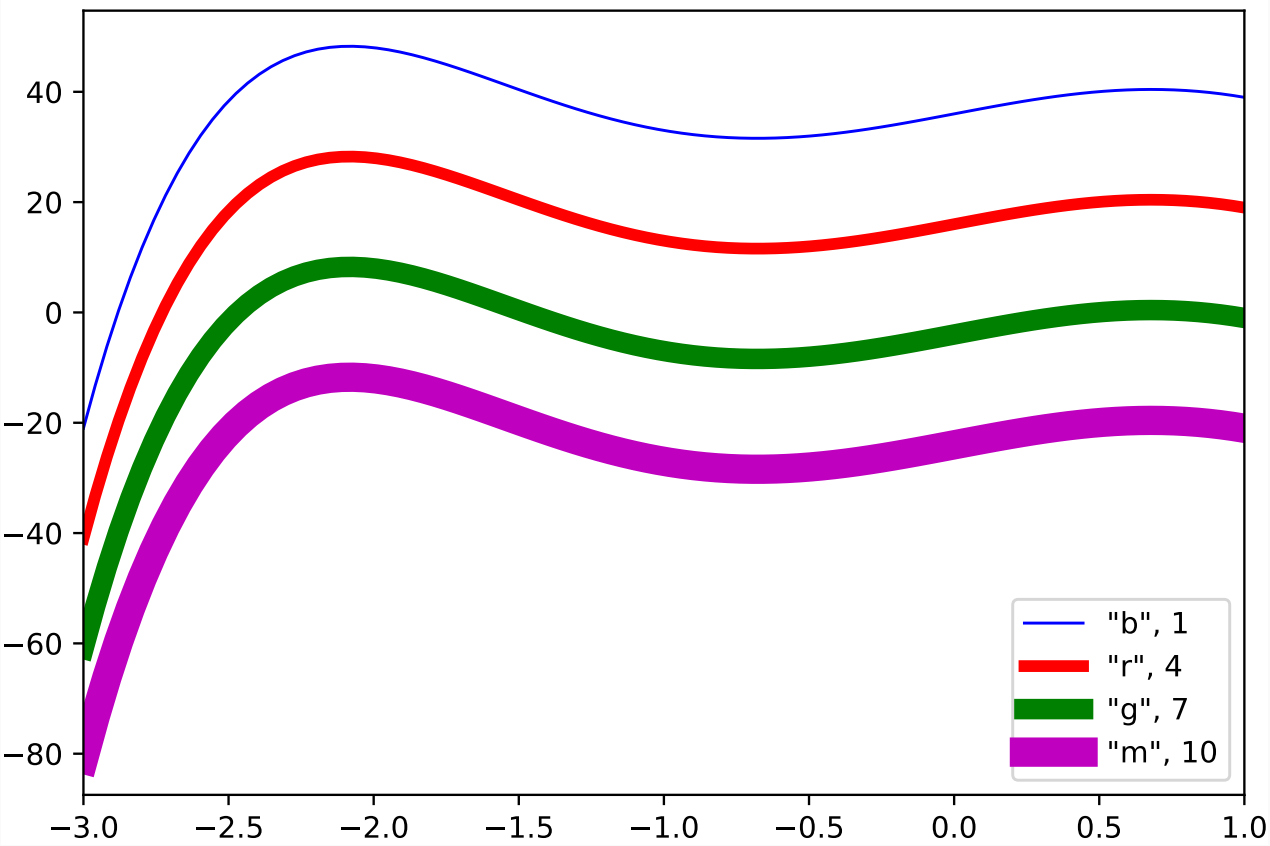
Basic Plotting Pipeline - 2D line example



- **Larger font size** especially when the figure is used in paper/report.

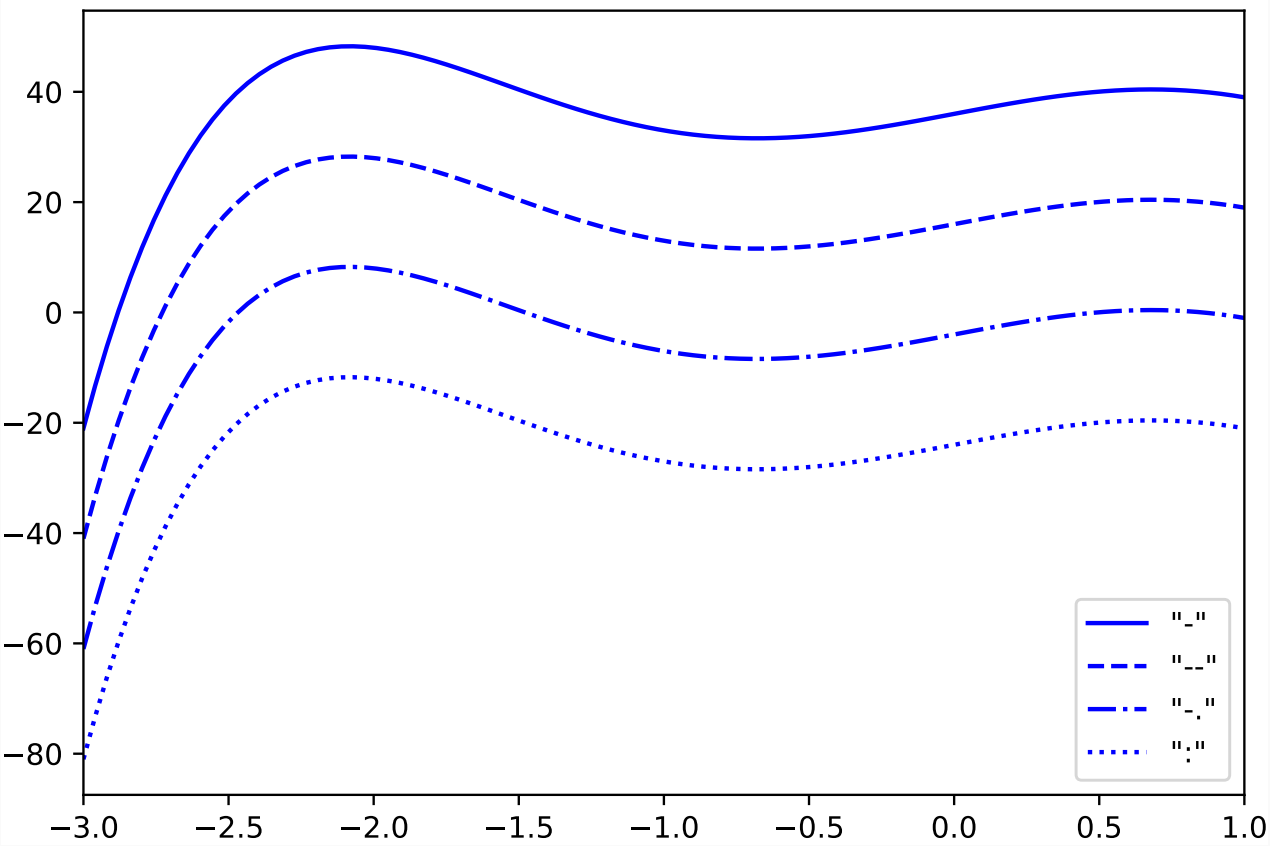
Adjusting Line Properties - Line Width and Color

```
plt.plot(x, y, linewidth=1, color='r') # 'r','g','b','y',...
```



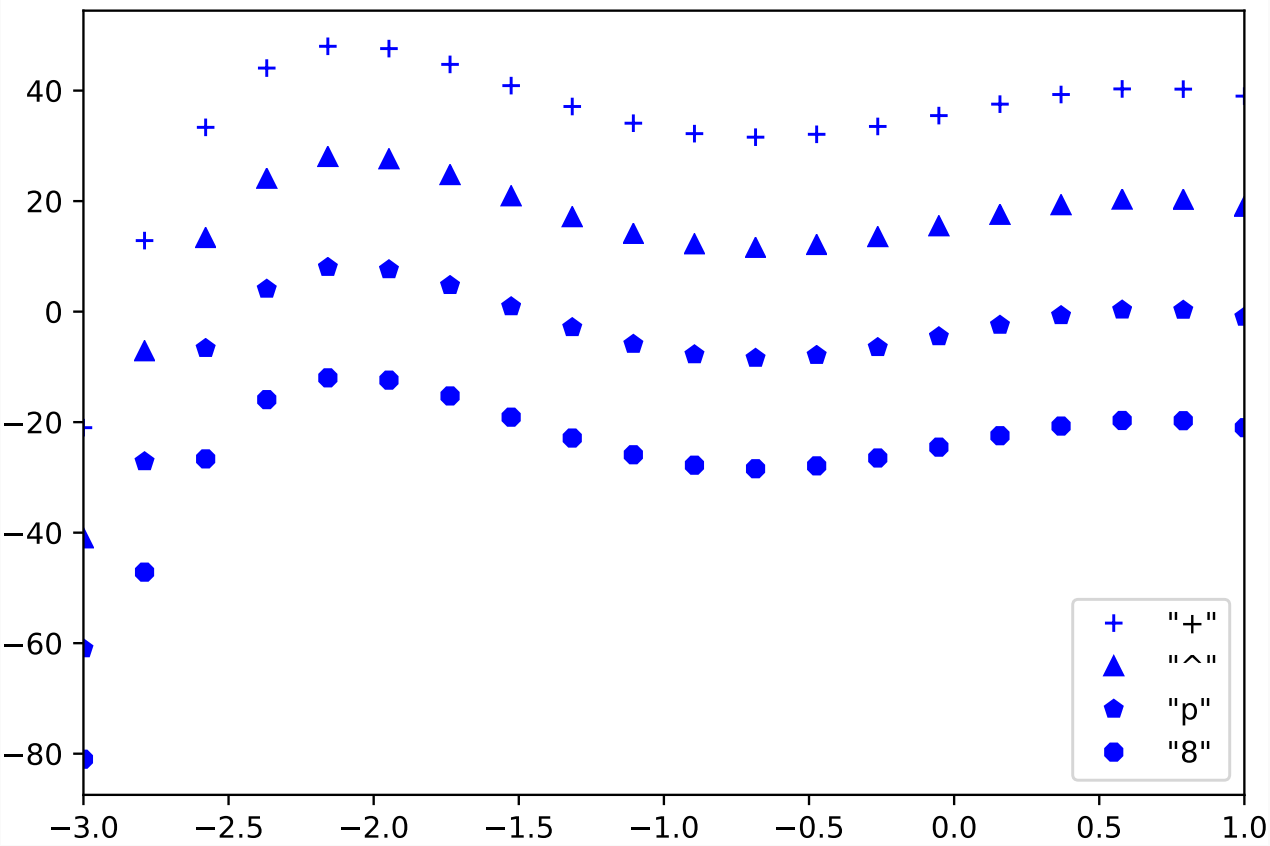
Adjusting Line Properties - Line Style

```
plt.plot(x, y, linestyle='-') # '-', '--', '-.', ':'
```



Adjusting Line Properties - Markers

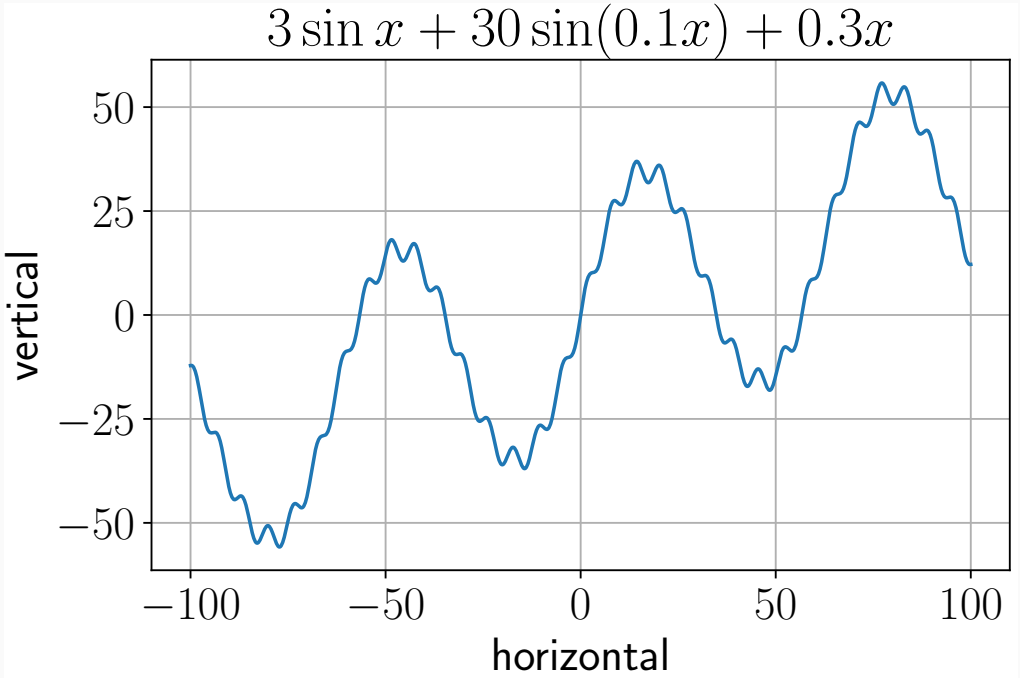
```
plt.plot(x, y, linestyle='', marker='+') # 'x','8','p',...
```



Axes Labels, Title, Grid

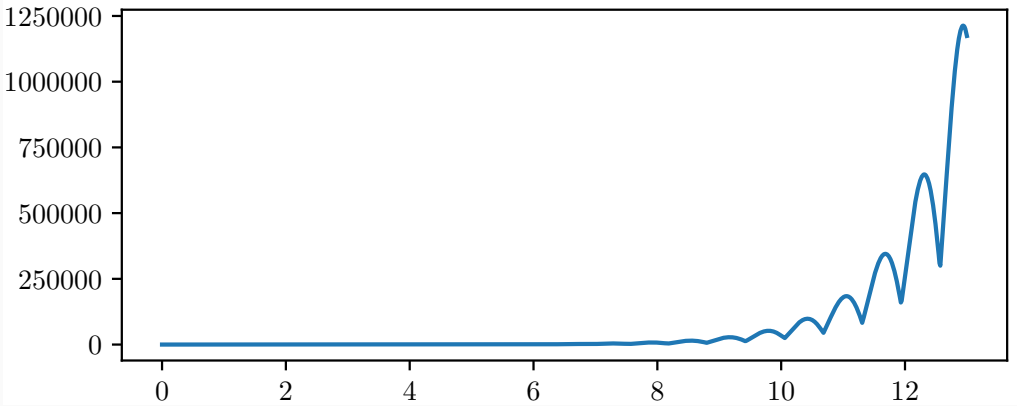
```
# TeX formulas rendering support.
from matplotlib import rc
rc('text', usetex=True)
```

```
plt.plot(x, y)
plt.xlabel('horizontal'); plt.ylabel('vertical')
plt.title('$3\sin{x} + 30\sin(0.1x) + 0.3x$' )
plt.grid()
```

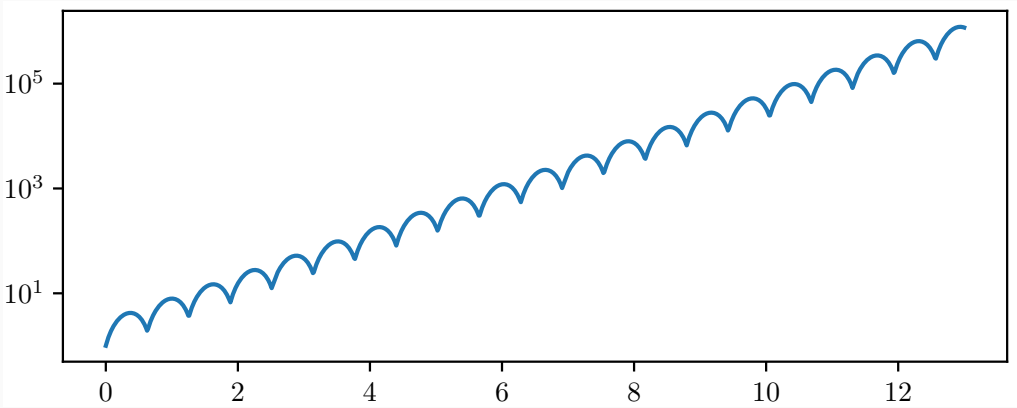


Adjusting Axes - Scale

```
plt.plot(x, y)
```

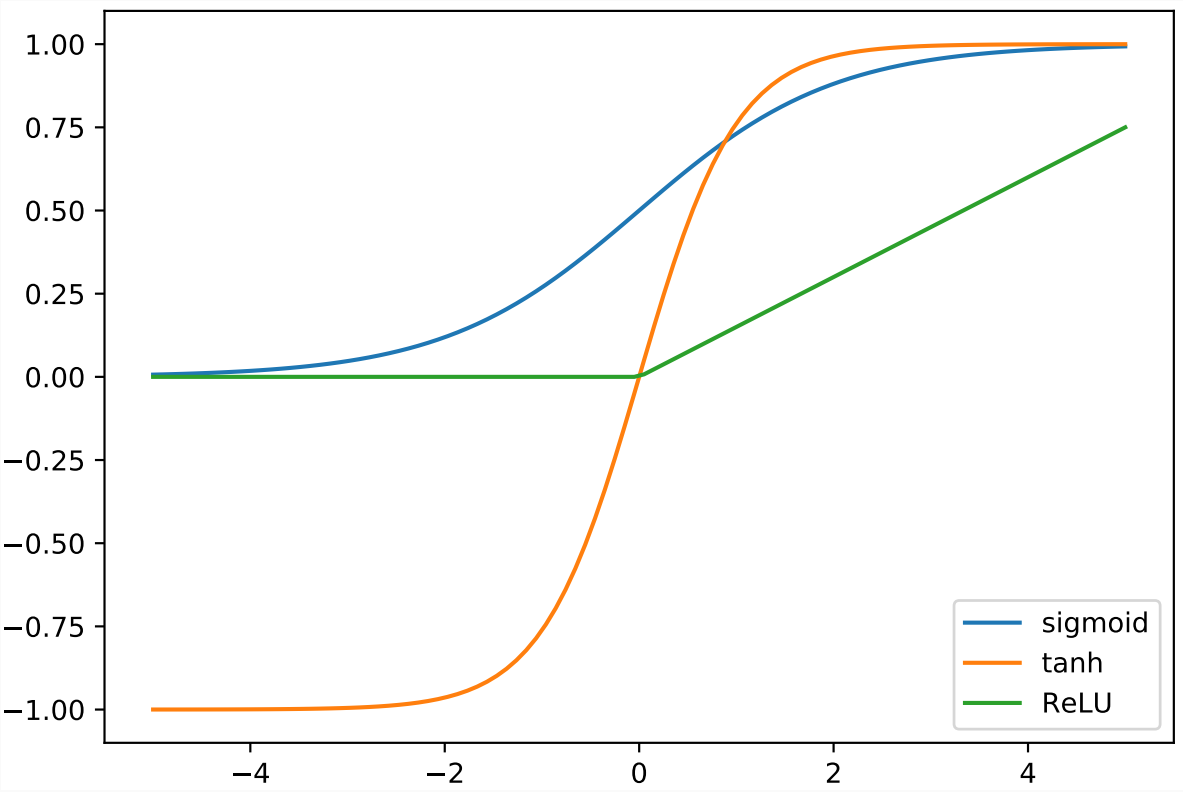



```
plt.semilogy(x, y) # plt.semilogx(), plt.loglog()
```



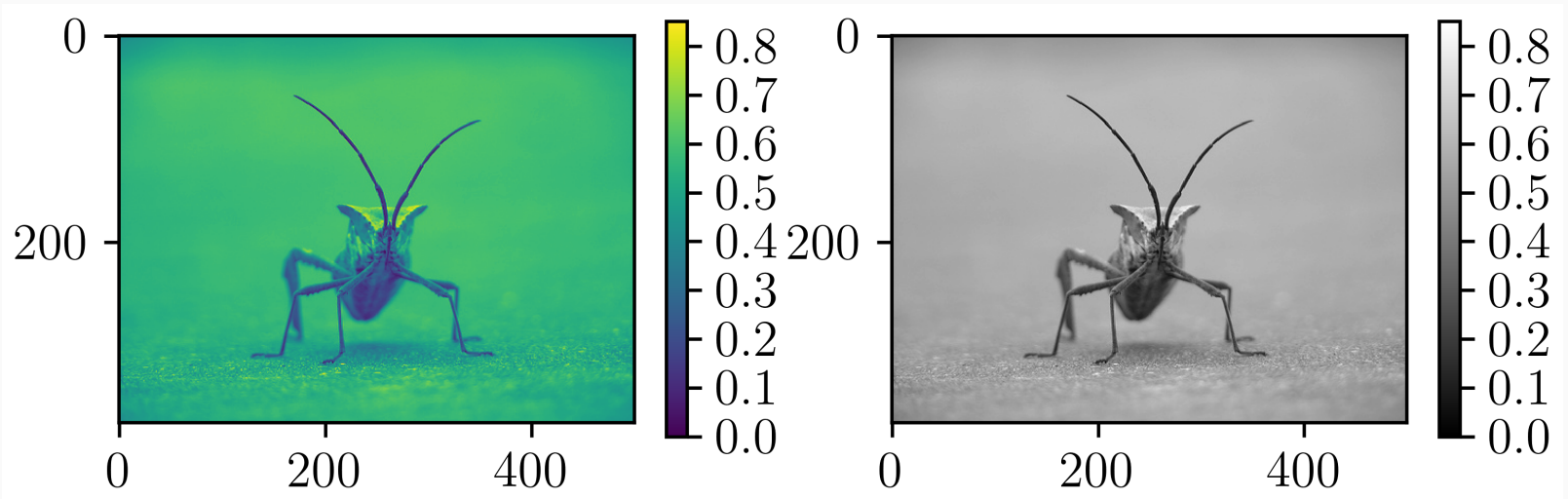
Plotting Multiple 2D Lines, Adding a Legend

```
plt.plot(x, y1, label='sigmoid')
plt.plot(x, y2, label='tanh')
plt.plot(x, y3, label='ReLU')
plt.legend(loc='lower right') # 'best','upper center',...
```

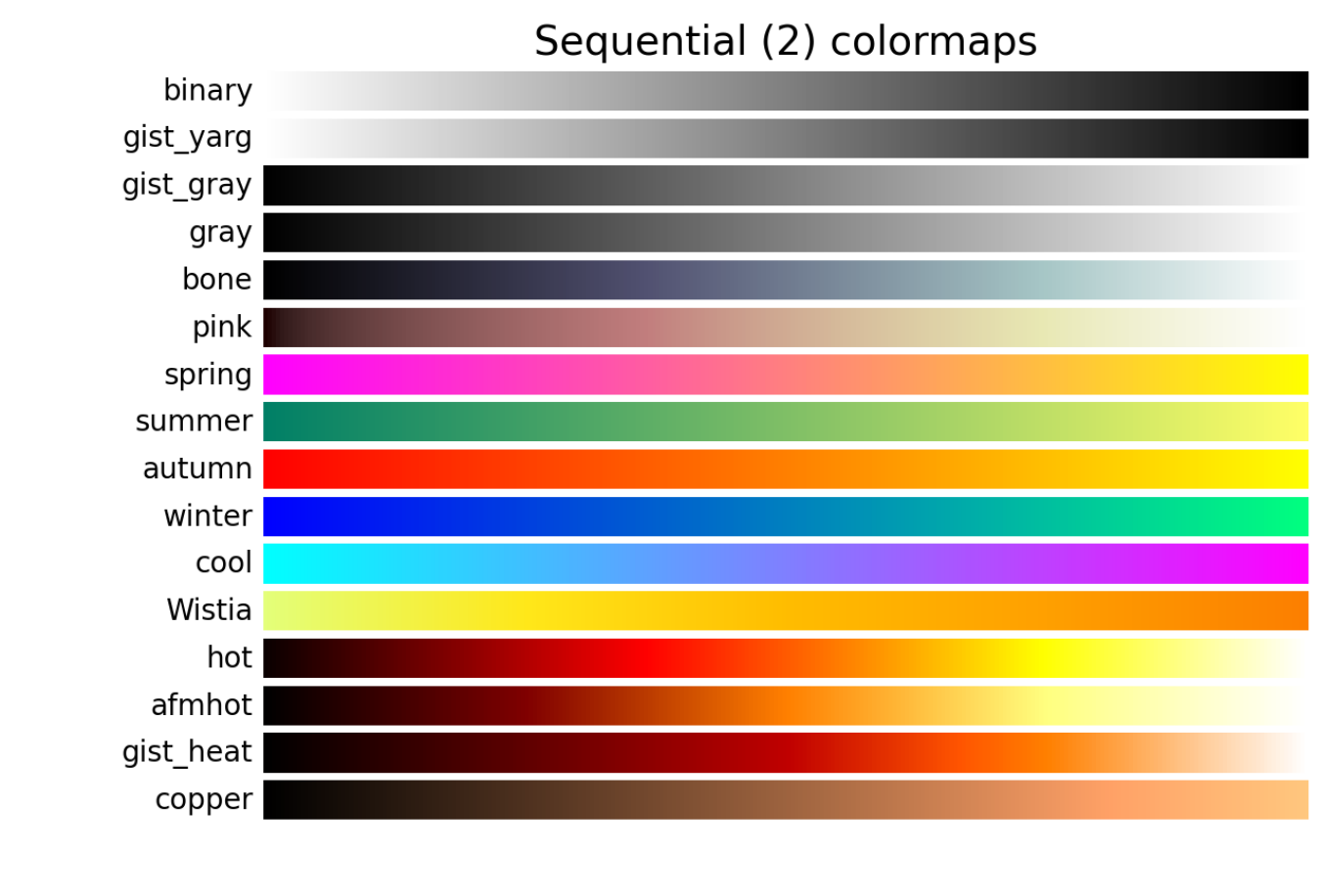


Plotting an Image

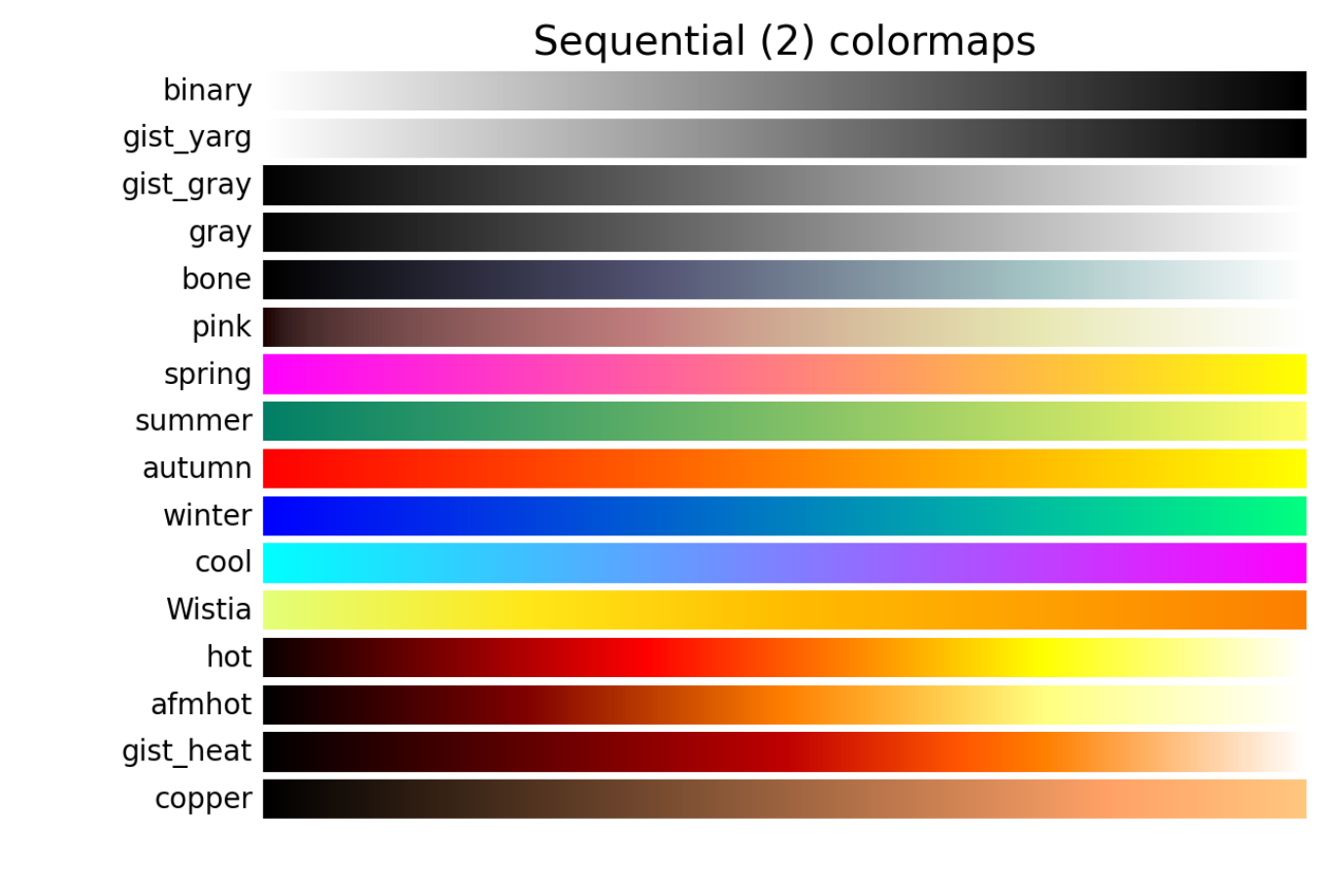
```
import matplotlib.image as mpimg
img = mpimg.imread('stinkbug.png')

plt.imshow(img)
plt.imshow(img, cmap='gray') # 'hot','plasma','inferno',...
plt.colorbar()
```



Plotting an Image



Plotting an Image

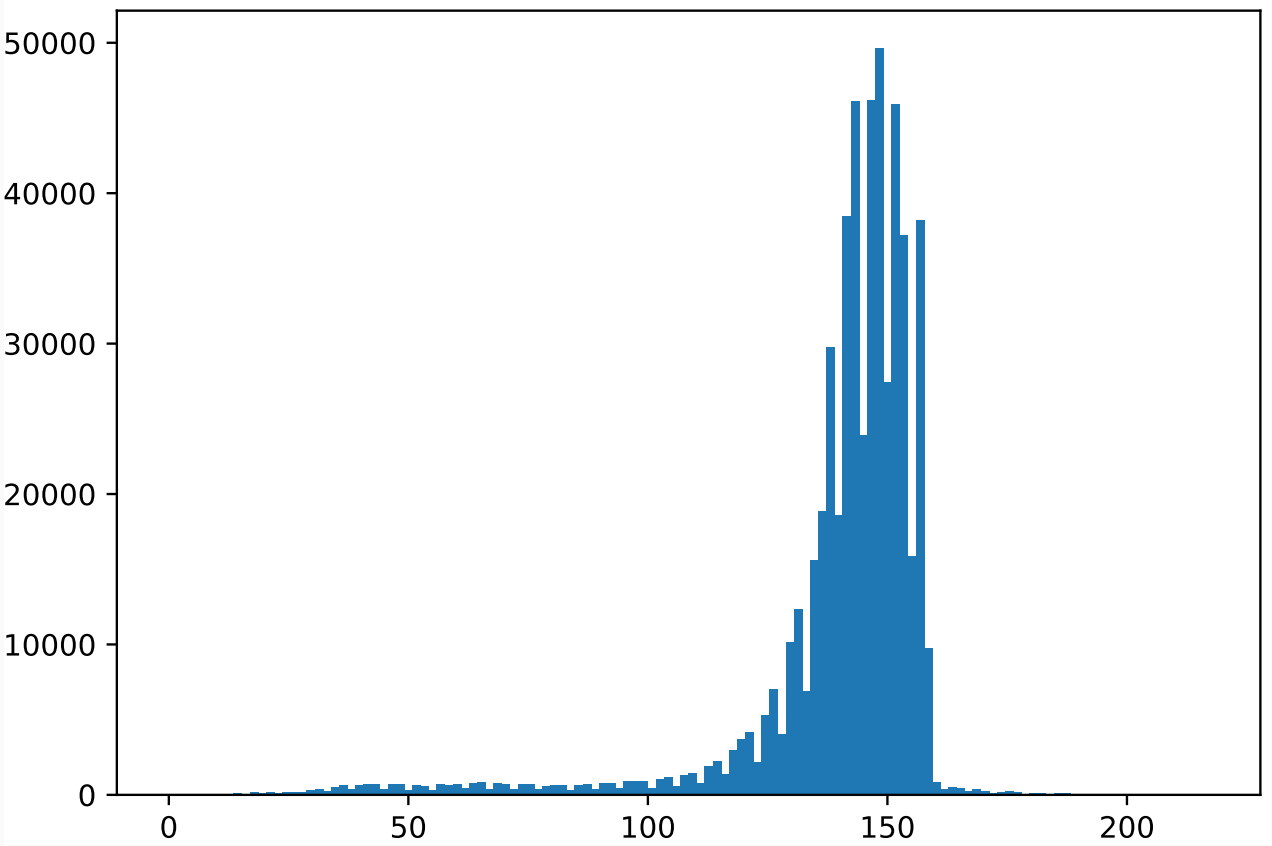


Always use pre-defined cmaps if possible.

- Vision deficiencies, perceptual uniformity, etc.

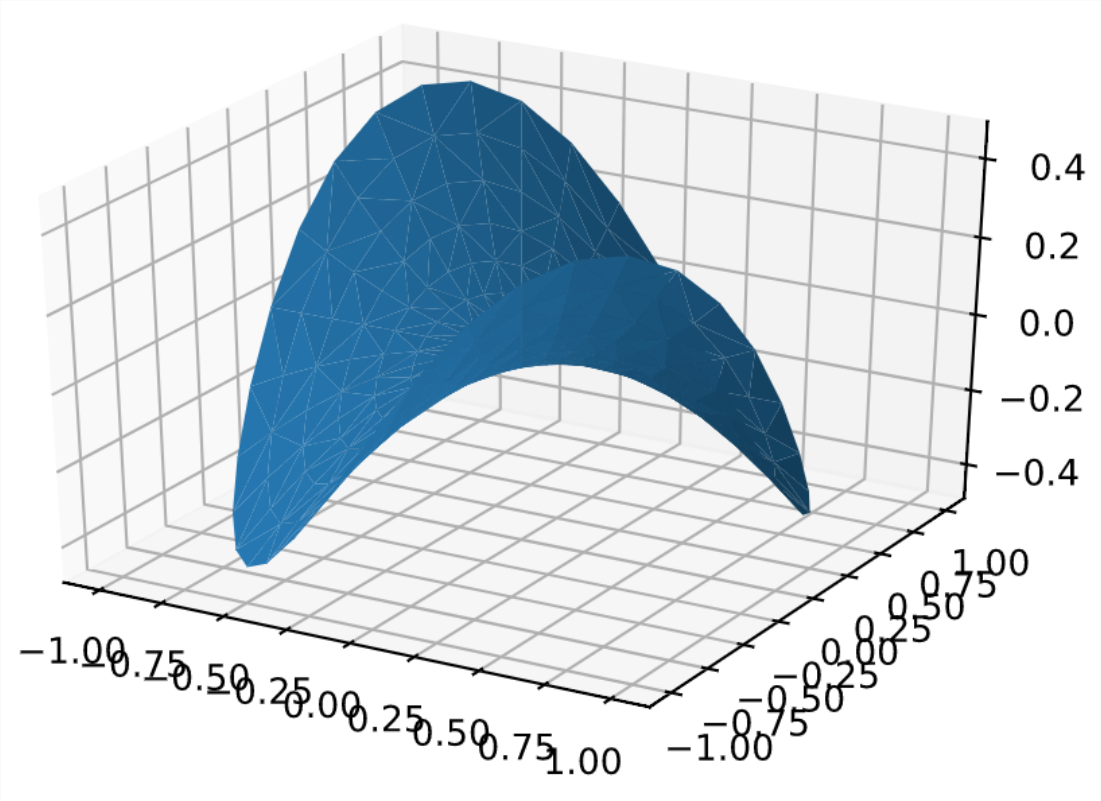
Plotting Histogram

```
_ = plt.hist(img.flatten(), bins=128)
```



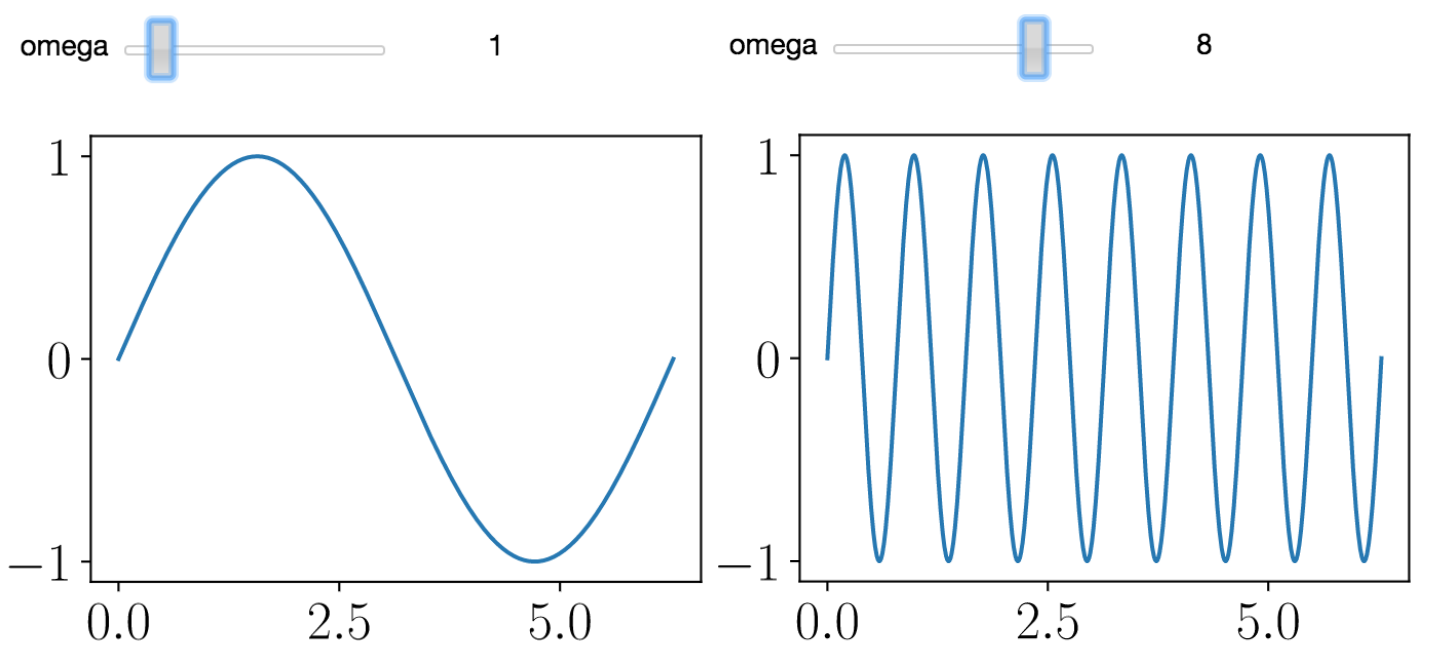
3D Plots

```
from mpl_toolkits.mplot3d import Axes3D
... # Prepare x, y, z data
plot_trisurf(x, y, z)
```



Interactivity - Slider

```
from ipywidgets import interact

@interact(omega=(0, 10, 1)) # min, max, step
def plotSin(omega):
        x = np.linspace(0.0, 2 * np.pi, 1000)
        y = np.sin(omega * x)
        plt.plot(x, y)
```



References

- Official tutorials:
  https://matplotlib.org/users/tutorials.html
- Official Matplotlib gallery (incl. source code):
  https://matplotlib.org/gallery.html
- General Concepts FAQ section:
  https://matplotlib.org/faq/usage_faq.html