

# Python for scientific research

## Plotting with Matplotlib

John Joseph Valletta

University of Exeter, Penryn Campus, UK

June 2017



Researcher  
Development

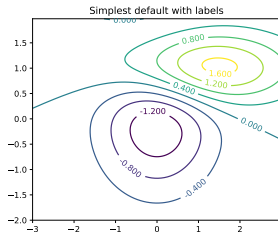
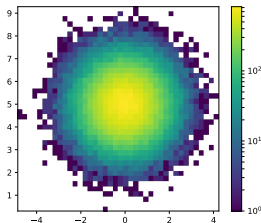
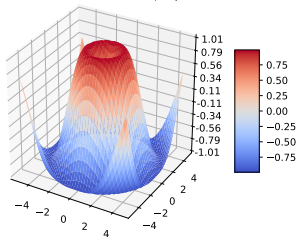
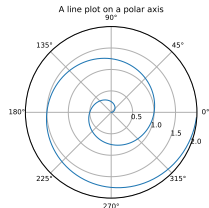
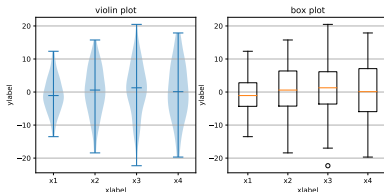
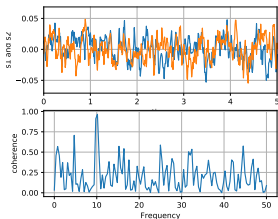


# What we've done so far

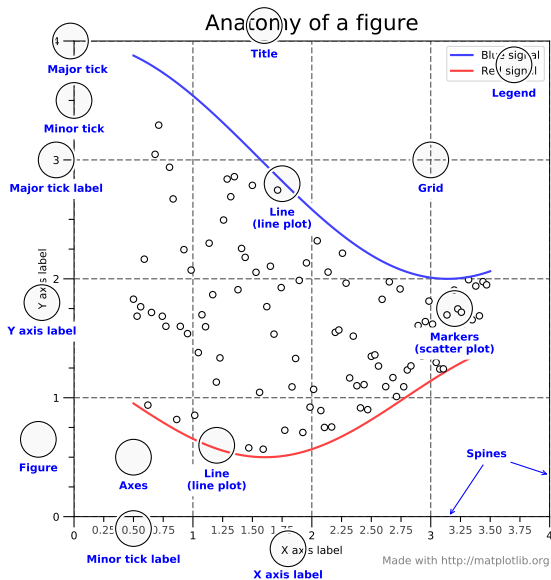
- 1 Declare variables using built-in data types and execute operations on them
- 2 Use flow control commands to dictate the order in which commands are run and when
- 3 Encapsulate programs into reusable functions, modules and packages
- 4 Using NumPy and SciPy for numerical computations
- 5 **Next:** Introducing Matplotlib, Python's plotting library

# Introduction

- Matplotlib is a 2D and 3D plotting library that produces publication-ready scientific figures in most formats (e.g PNG, EPS)



# Anatomy of a figure



# My first plot

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

# Generate sinusoidal data
x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

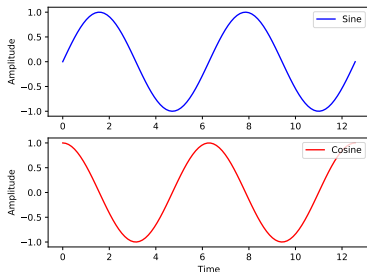
# Plot sinusoidal curve
plt.plot(x, y, color="blue")
plt.xlabel("Time")
plt.ylabel("Amplitude")
plt.title("My first Python plot")
```



# Subplots

```
plt.subplot(2, 1, 1) # 2 rows, 1 column, first plot
plt.plot(x, np.sin(x), color="b", label="Sine")
plt.ylabel("Amplitude")
plt.legend(loc="upper right")

plt.subplot(2, 1, 2) # 2 rows, 1 column, second plot
plt.plot(x, np.cos(x), color="r", label="Cosine")
plt.xlabel("Time")
plt.ylabel("Amplitude")
plt.legend(loc="upper right")
```

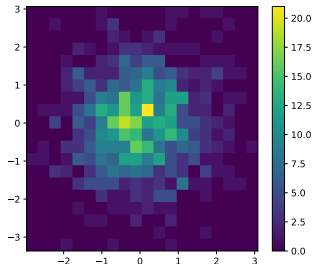
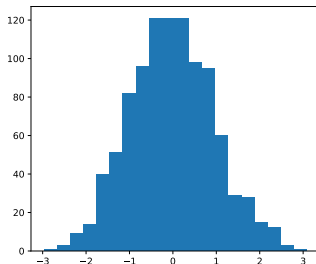


# Histograms

```
# Generate 1000 normally distributed numbers
x1 = np.random.randn(1000)
x2 = np.random.randn(1000)

# 1D histogram (x1)
plt.subplot(1, 2, 1)
plt.hist(x1, bins=20)

# 2D histogram (x1 vs x2)
plt.subplot(1, 2, 2)
plt.hist2d(x1, x2, bins=20)
plt.colorbar()
```

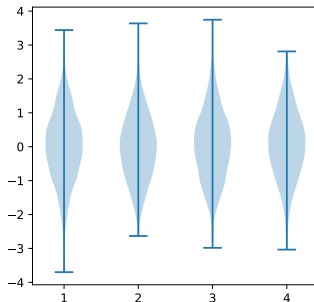
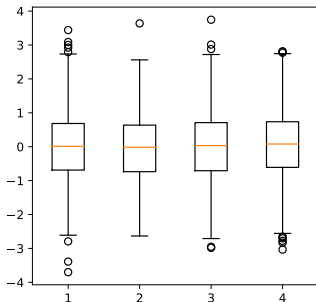


# Boxplots and violin plots

```
# Generate four sets of random numbers
data = [np.random.randn(1000) for i in range(4)]

# Boxplot
plt.subplot(1, 2, 1)
plt.boxplot(data)

# Violin plot
plt.subplot(1, 2, 2)
plt.violinplot(data)
```



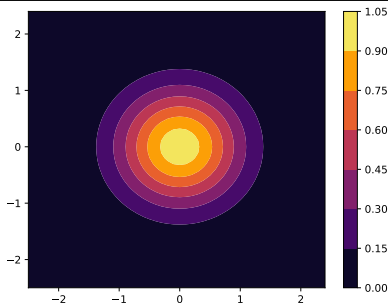


# Contour plot

```
from matplotlib import cm

# Generate some x, y, z data
x = np.arange(-2.5, 2.5, 0.1)
y = np.arange(-2.5, 2.5, 0.1)
x, y = np.meshgrid(x, y)
z = np.exp(-x**2 - y**2)

# Contour plot
plt.contourf(x, y, z, cmap=cm.inferno)
plt.colorbar()
```



# 3D surface

```
from mpl_toolkits.mplot3d import Axes3D

# Create figure amenable for 3D plotting
hFig = plt.figure()
hAx = hFig.gca(projection="3d")

# Plot the surface
hSurf = hAx.plot_surface(x, y, z, cmap=cm.inferno)
plt.colorbar(hSurf, shrink=0.5)
```

