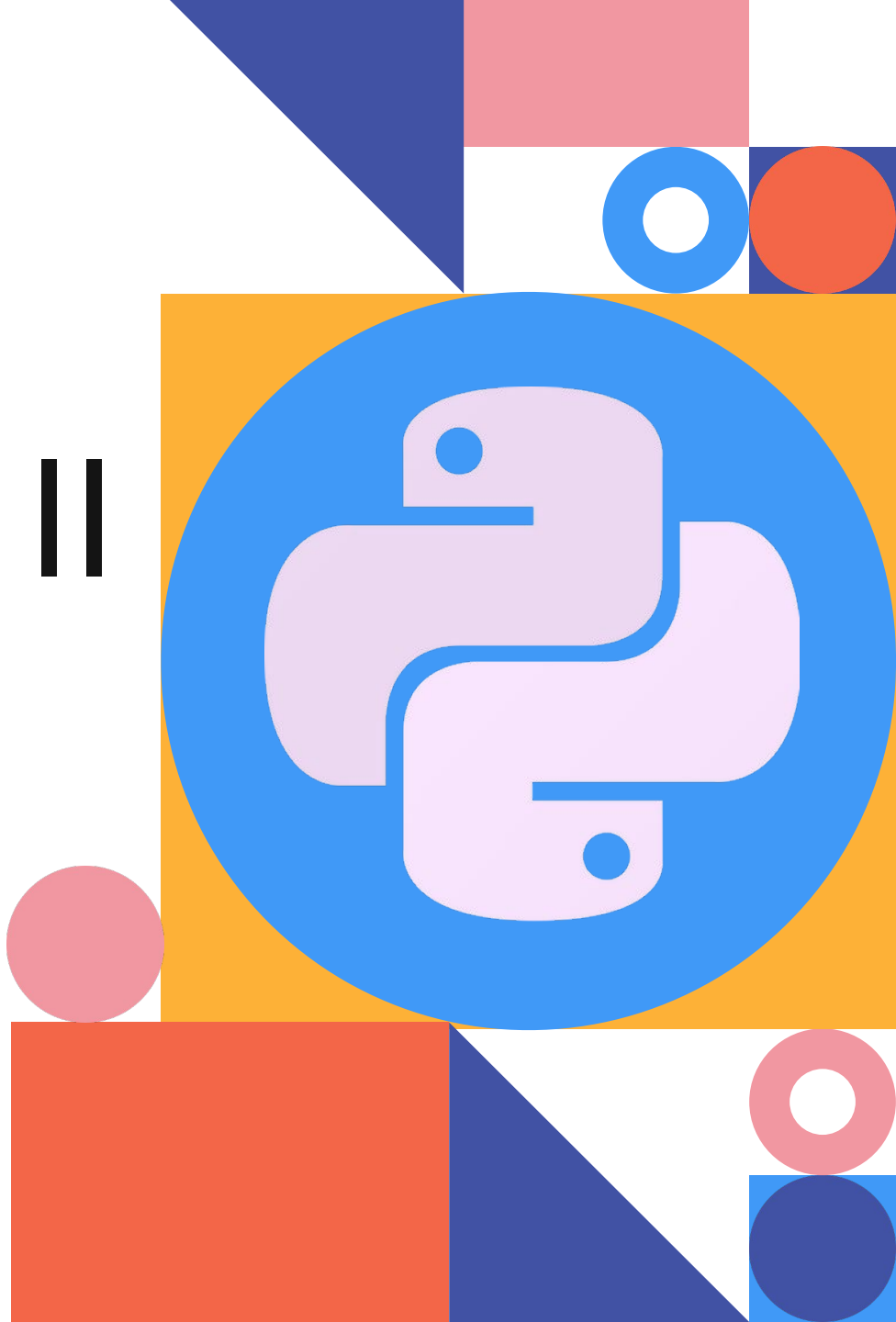
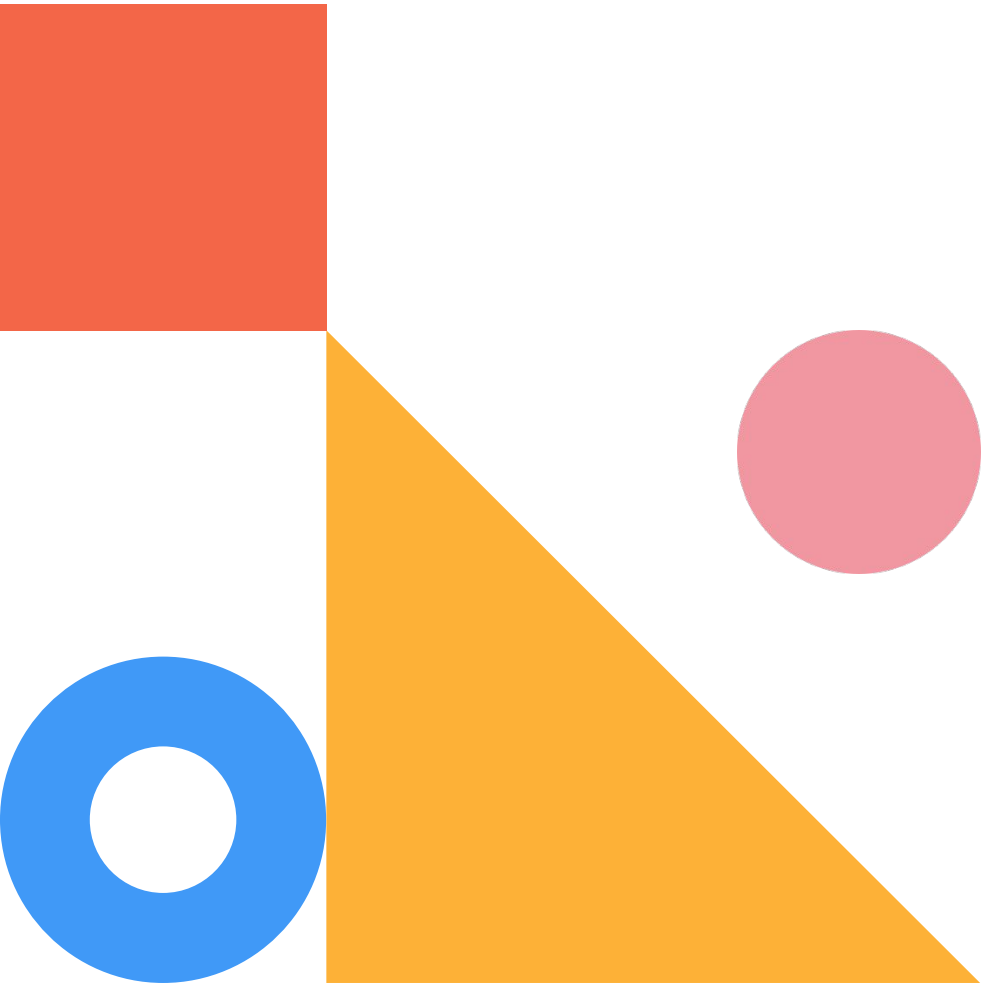
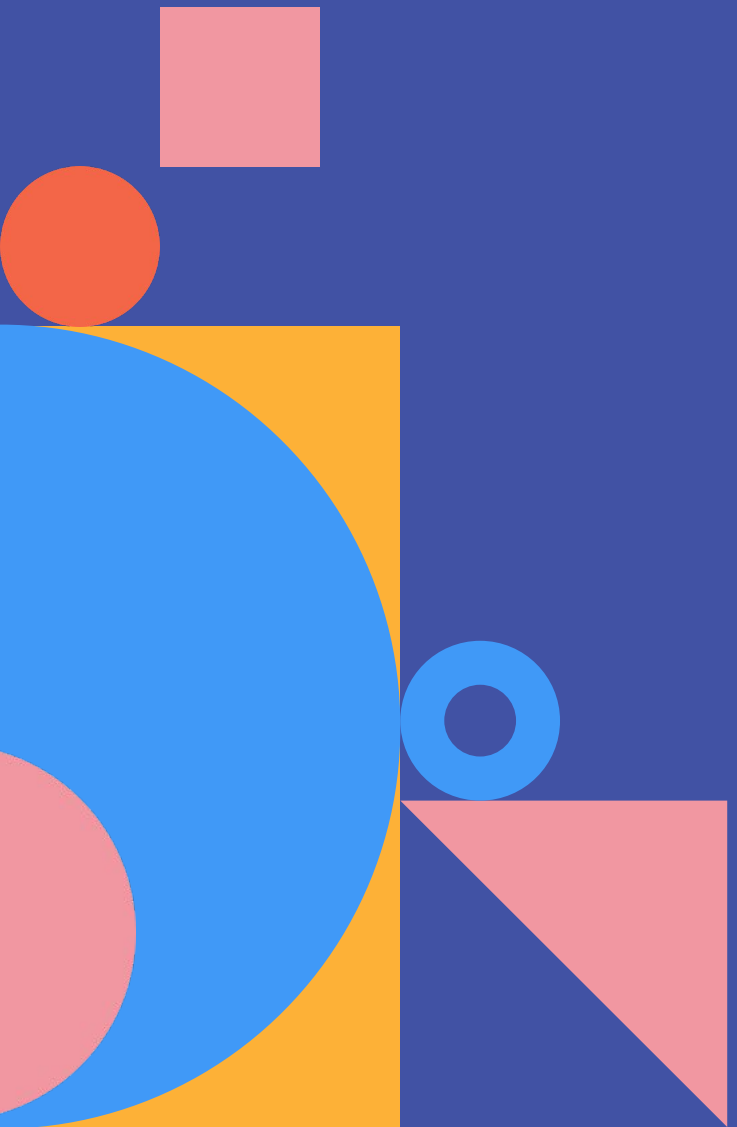
Data Visualization II



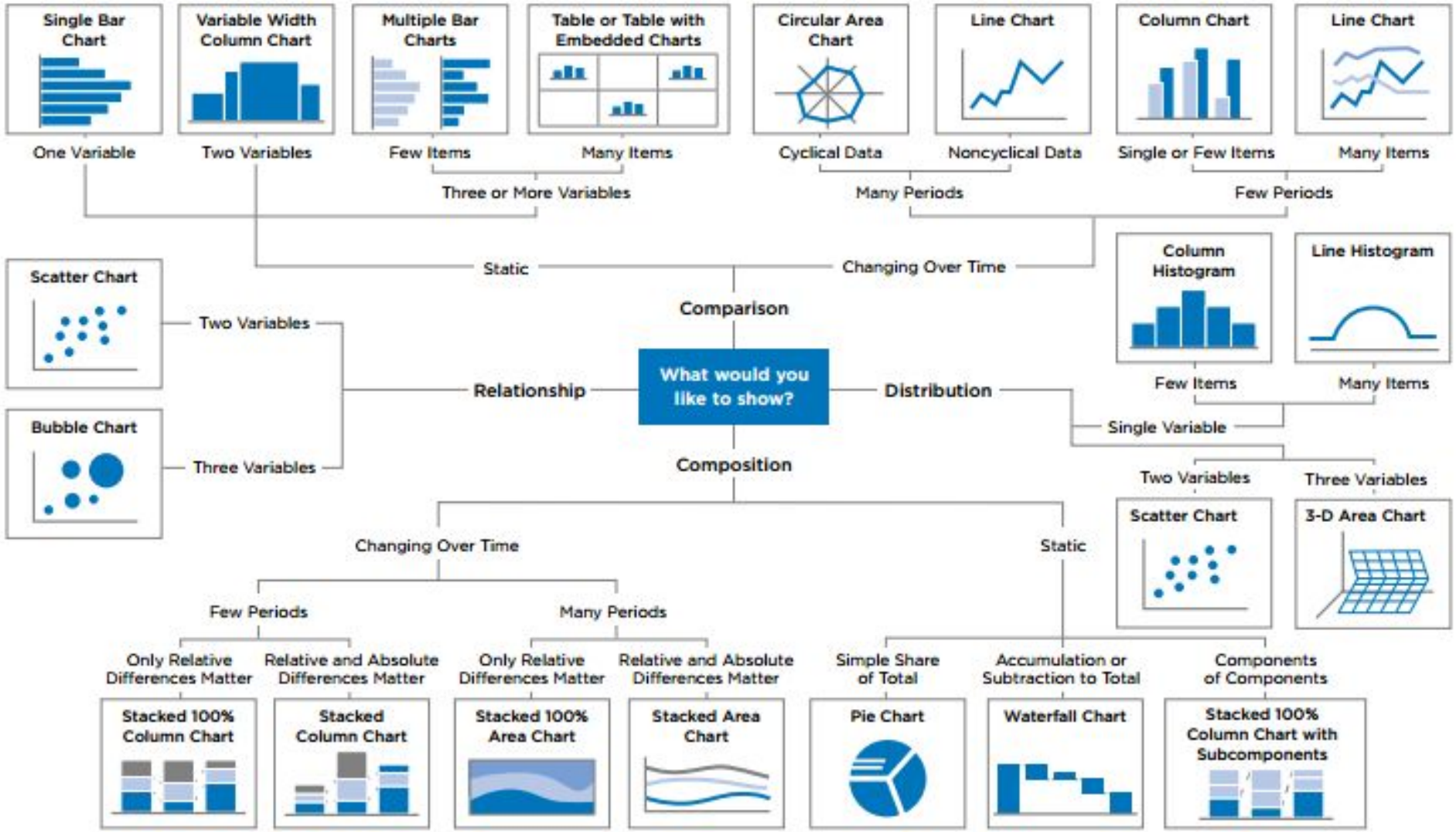
Yan-Fu Kuo
Dept. of Biomechatronics Engineering
National Taiwan University

Contents





01 Charts


What would you like to show?
Relationship
Distribution
Composition
Comparison
Static
Changing Over Time
Two Variables
Three Variables
Single Bar Chart
Variable Width Column Chart
Multiple Bar Charts
Table or Table with Embedded Charts
Circular Area Chart
Line Chart
Column Chart
Line Chart
One Variable
Two Variables
Few Items
Many Items
Three or More Variables
Cyclical Data
Noncyclical Data
Many Periods
Single or Few Items
Many Items
Few Periods
Scatter Chart
Bubble Chart
Column Histogram
Line Histogram
Few Items
Many Items
Single Variable
Two Variables
Three Variables
Scatter Chart
3-D Area Chart
Changing Over Time
Static
Few Periods
Many Periods
Only Relative Differences Matter
Relative and Absolute Differences Matter
Only Relative Differences Matter
Relative and Absolute Differences Matter
Simple Share of Total
Accumulation or Subtraction to Total
Components of Components
Stacked 100% Column Chart
Stacked Column Chart
Stacked 100% Area Chart
Stacked Area Chart
Pie Chart
Waterfall Chart
Stacked 100% Column Chart with Subcomponents

Quiz

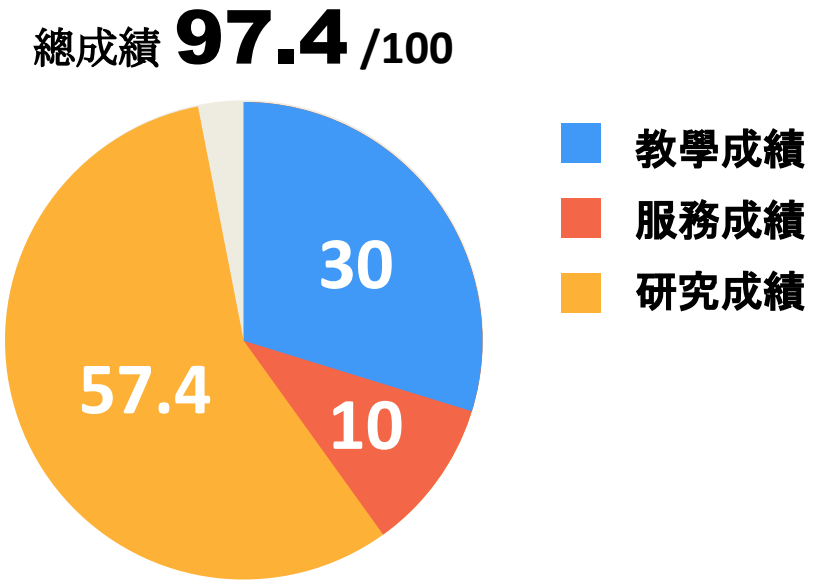
Which chart is misused?

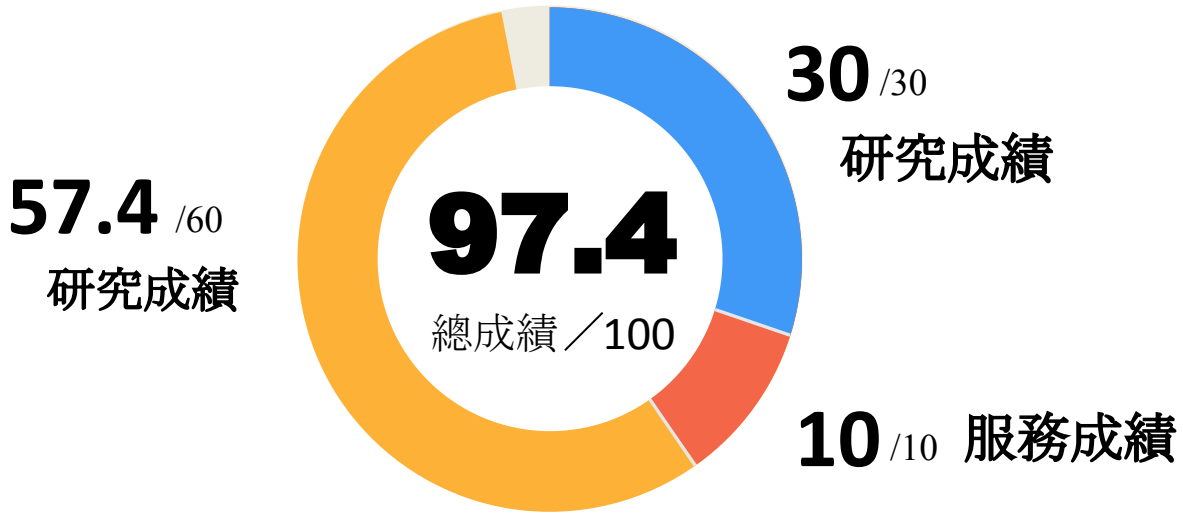
Data Looks Better Naked

DARKHORSE ANALYTICS

Another Example

How would you improve the chart?





Further Resources

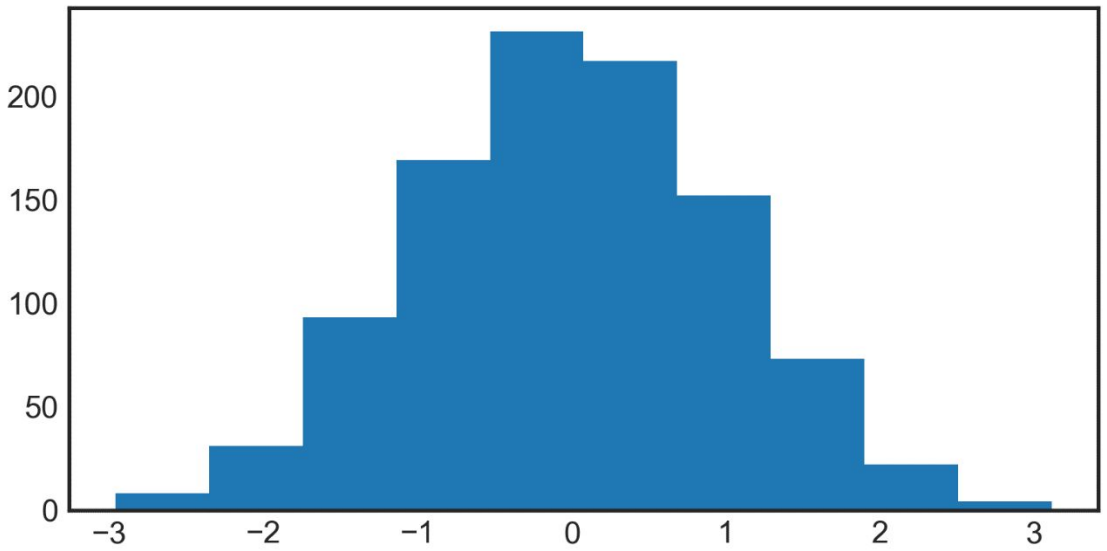
The 56 Best — And Weirdest — Charts We Made In 2019, FiveThirtyEight
https://fivethirtyeight.com/features/the-56-best-and-weirdest-charts-we-made-in-2019/

A year in Graphic detail, The Economist
https://infographics.economist.com/2019/AChristmasGiftForYou/AYearInGraphicDetail.pdf

Het jaar in interactieve datajournalistiek, DE TIJD
https://www.tijd.be/dossiers/terugblik-2019/het-jaar-in-interactieve-datajournalistiek/10193680.html

Histogram

```
data = np.random.randn(1000)
plt.hist(data)
```

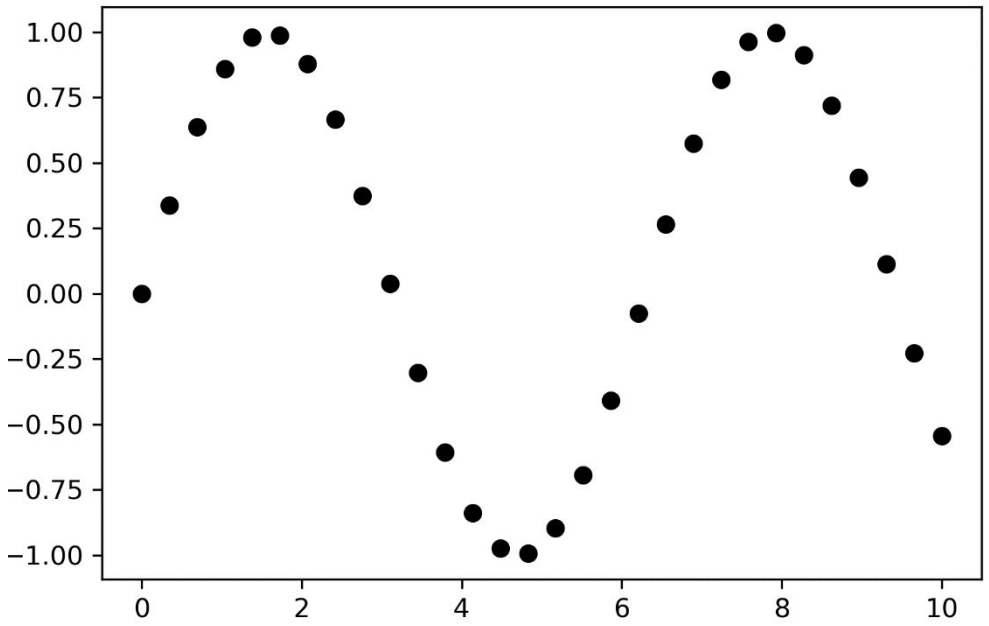


properties:

- bins
- range
- density
- weights
- bottom
- histtype
- color
- stacked

Scatter Plot

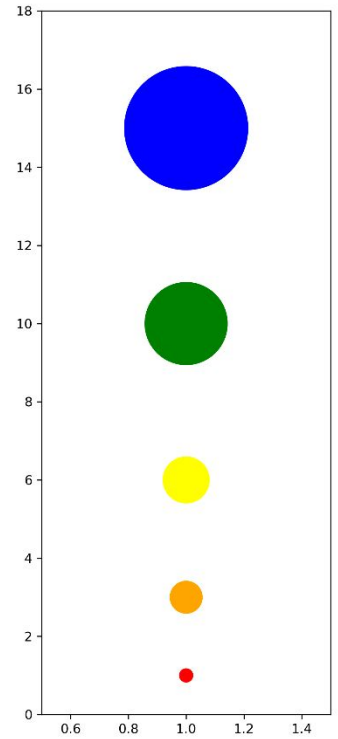
```
plt.scatter(x, y, marker='o')
```



`plt.scatter` create scatter plots where the properties of each individual point (size, face color, edge color, etc.) can be individually controlled or mapped to data.

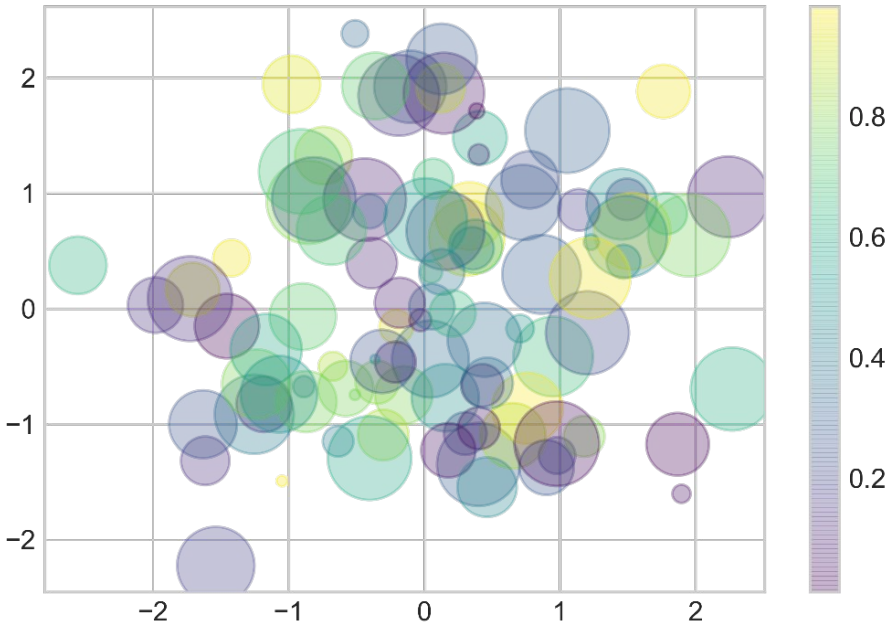
An Example Scatter Plot

```
x = [1, 1, 1, 1, 1]

y = [1, 3, 6, 10, 15]

sizevalues = [100, 600, 1250, 4000, 9000]

plotcolor = ['red','orange','yellow','green','
blue']

plt.scatter(x, y, s=sizevalues, c=plotcolor)
```



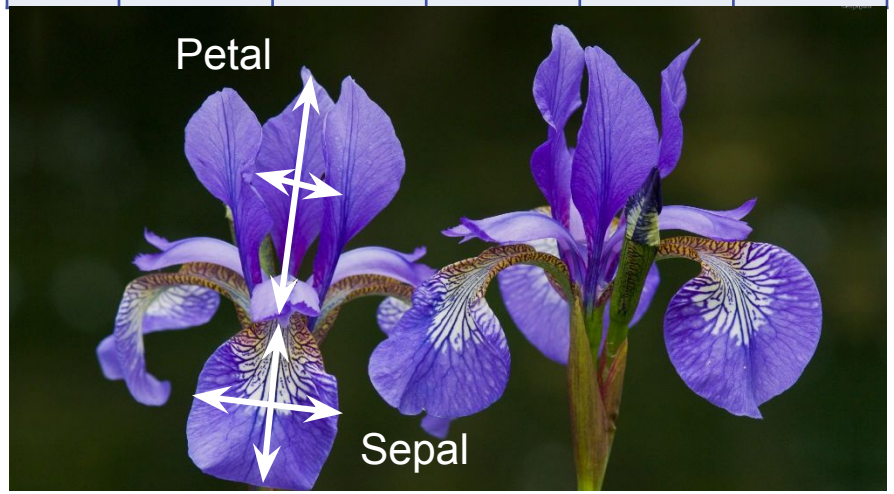
An Example Scatter Plot

```
rng = np.random.RandomState(0)
x = rng.randn(100)
y = rng.randn(100)
colors = rng.rand(100)
sizes = 1000 * rng.rand(100)

plt.scatter(x, y, c=colors, s=sizes,
            alpha=0.3, cmap='viridis')
plt.colorbar();  # show color scale
```

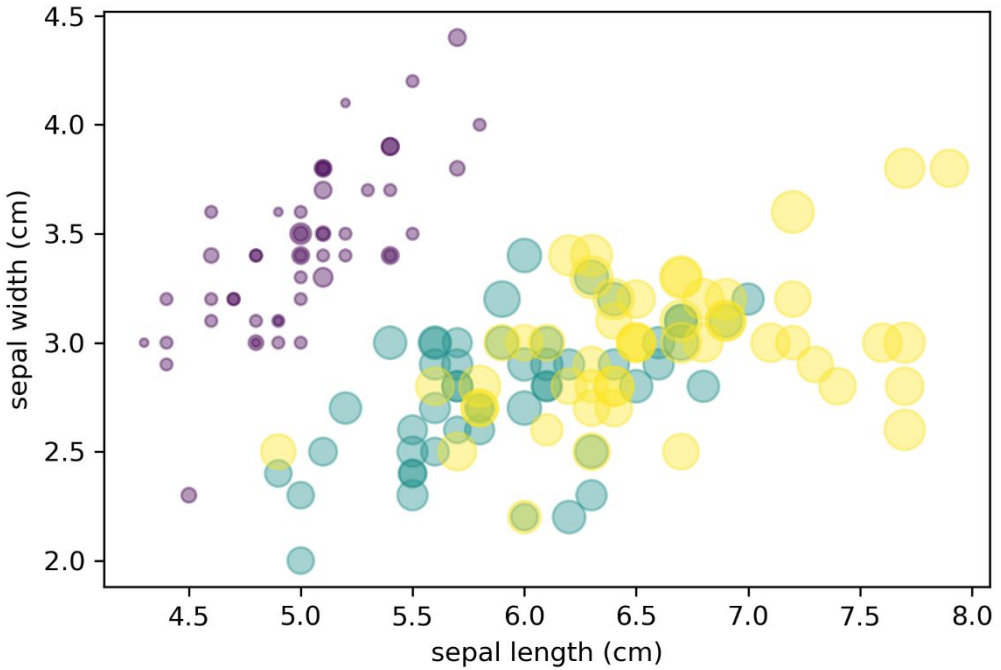


Exercise – The Iris Dataset (15mins)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
…					
150	5.9	3.0	5.0	1.8	Virginica



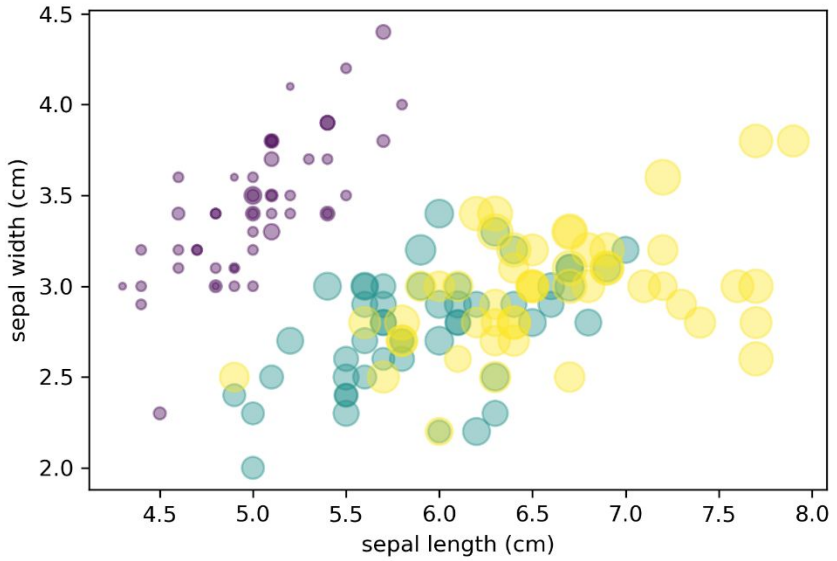
Complete the provided template to plot the figure below.



Exercise – The Iris Dataset (15mins)

Complete the provided template to plot the figure below.
You may want to check the schema of `iris` first.

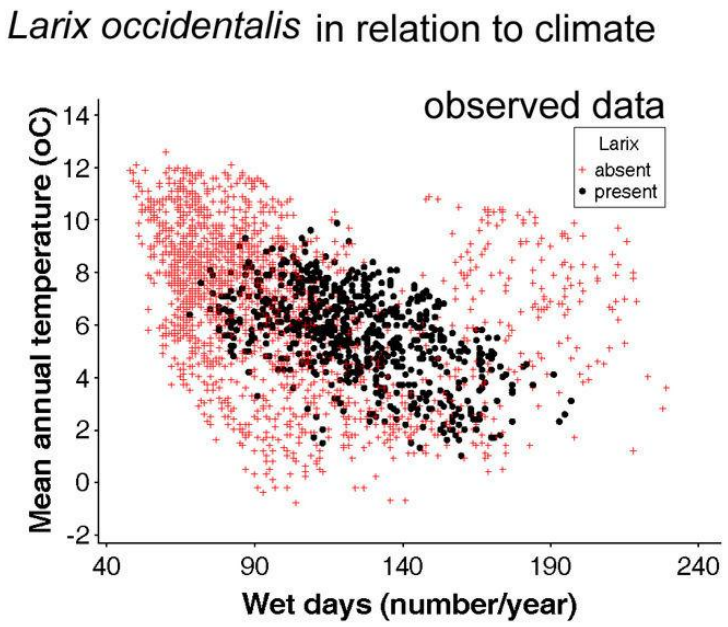
```
from sklearn.datasets import load_iris
iris = load_iris()
features = iris.data.T

plt.scatter(____, ____, alpha=0.4,
            s=100*____, c=____, cmap='viridis')
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
```

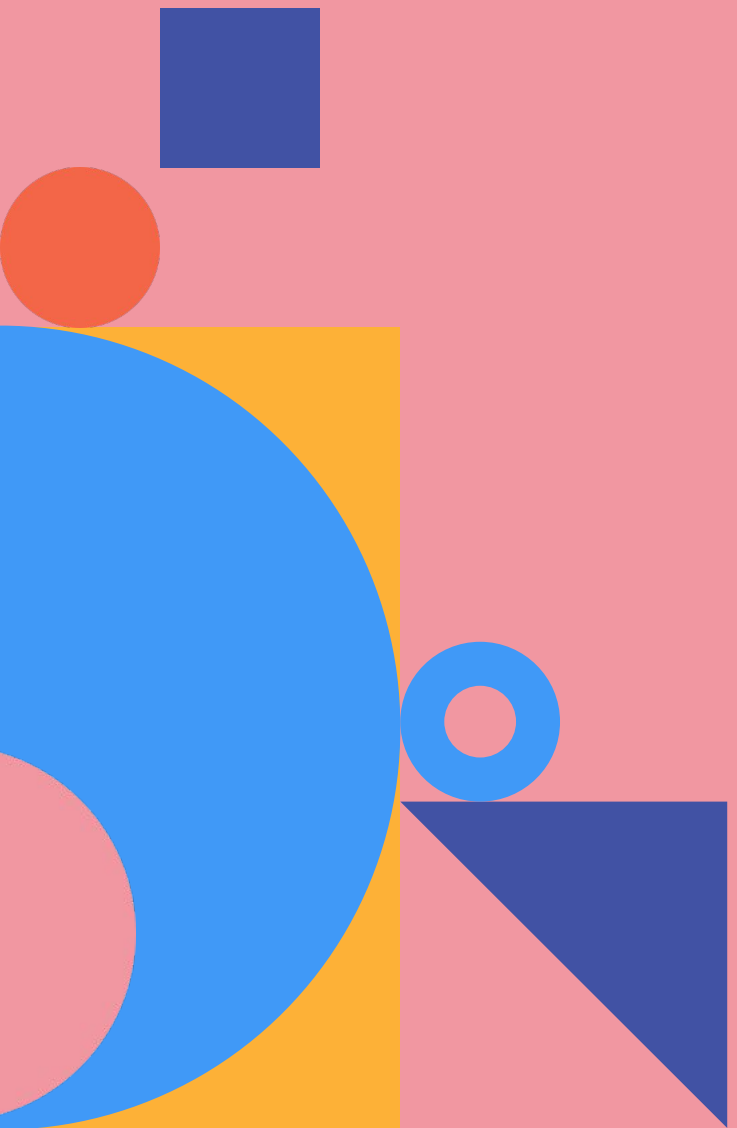


A Note on Efficiency

Aside from the different features available in plt.plot and plt.scatter, why might you choose to use one over the other?

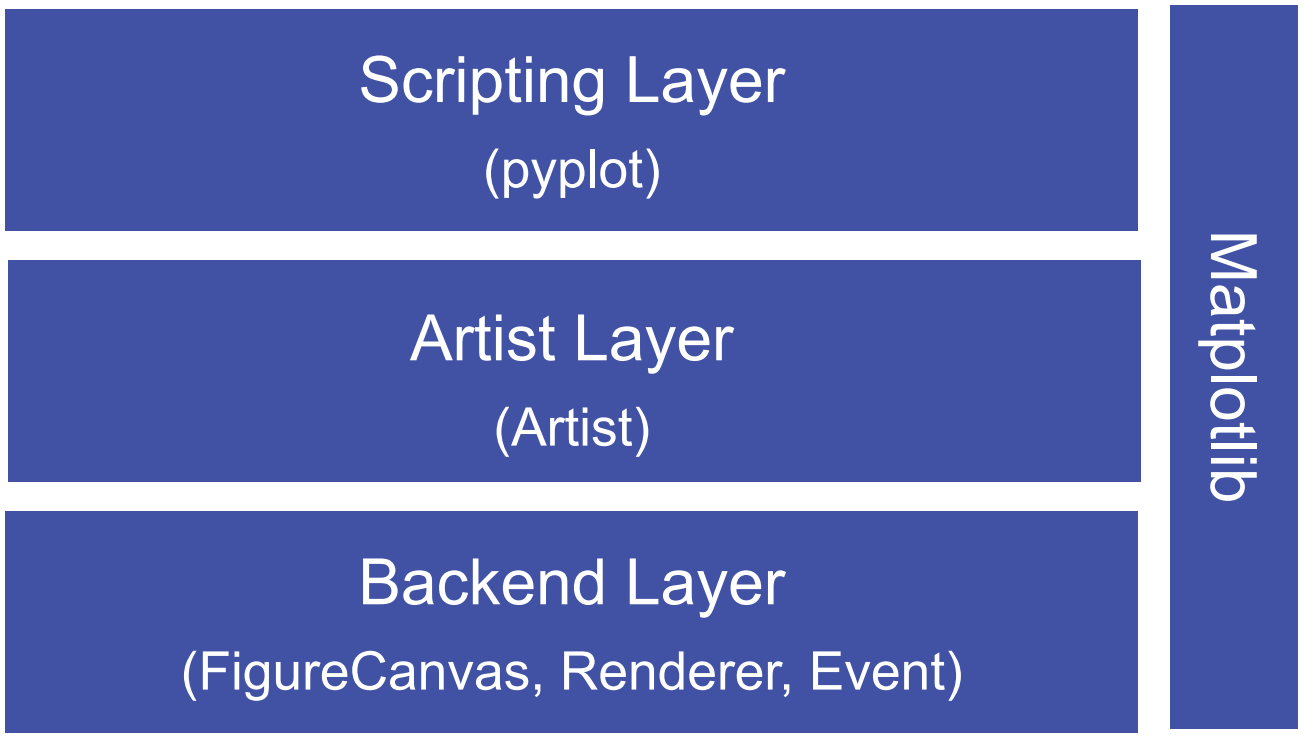
Which function should you use when plotting the figure on the right?





02 More on Matplotlib

Architecture



Backend Layer

FigureCanvas

encapsulates the concept of a surface to draw onto (e.g. "the paper")

Renderer

does the drawing (e.g. "the paintbrush")

Event

handles user inputs such as keyboard and mouse events.

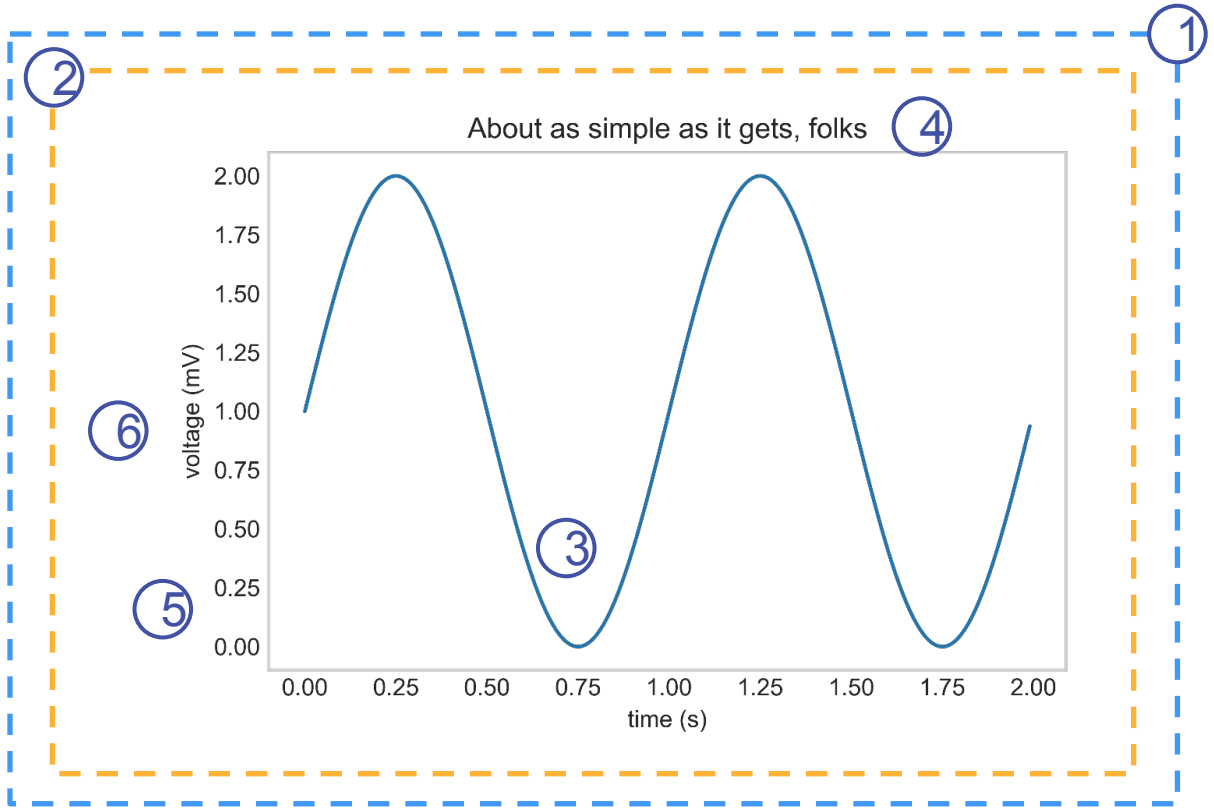
Backend Layer

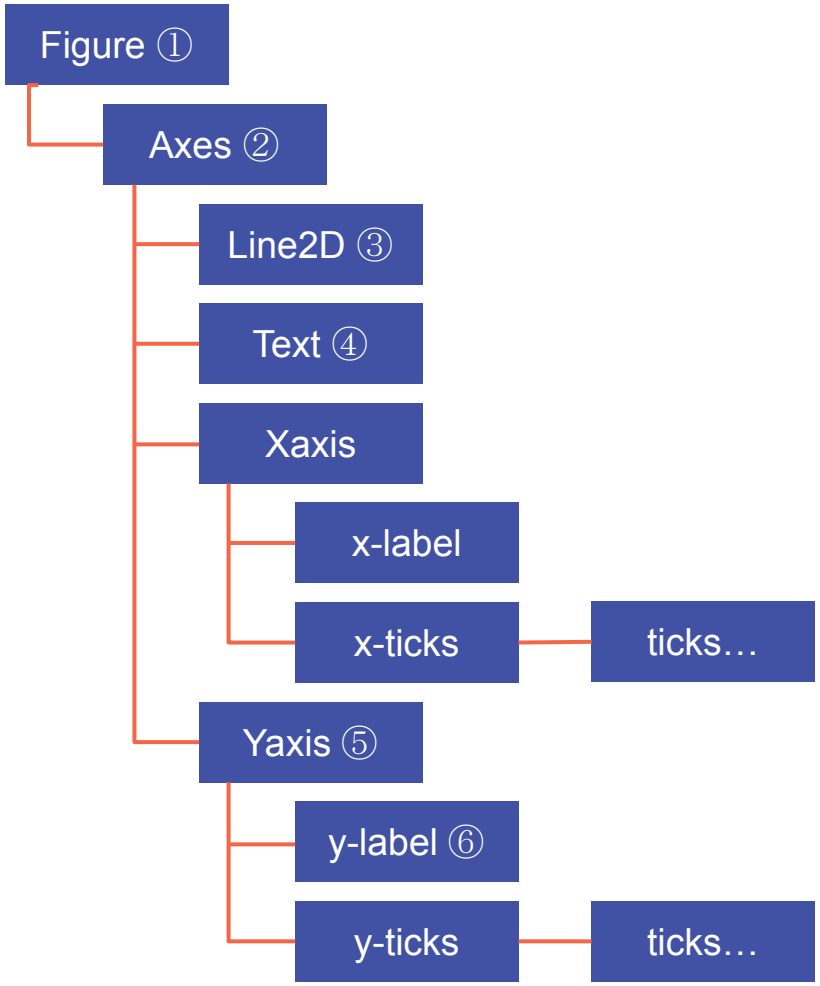
Run the script and press ‘t’ on your keyboard

```
def on_press(event):
    if event.inaxes is None: return
    for line in event.inaxes.lines:
        if event.key=='t':
            visible = line.get_visible()
            line.set_visible(not visible)
    event.inaxes.figure.canvas.draw()

fig, ax = plt.subplots(1)
fig.canvas.mpl_connect('key_press_event', on_press)
ax.plot(np.random.rand(2, 20))
plt.show()
```

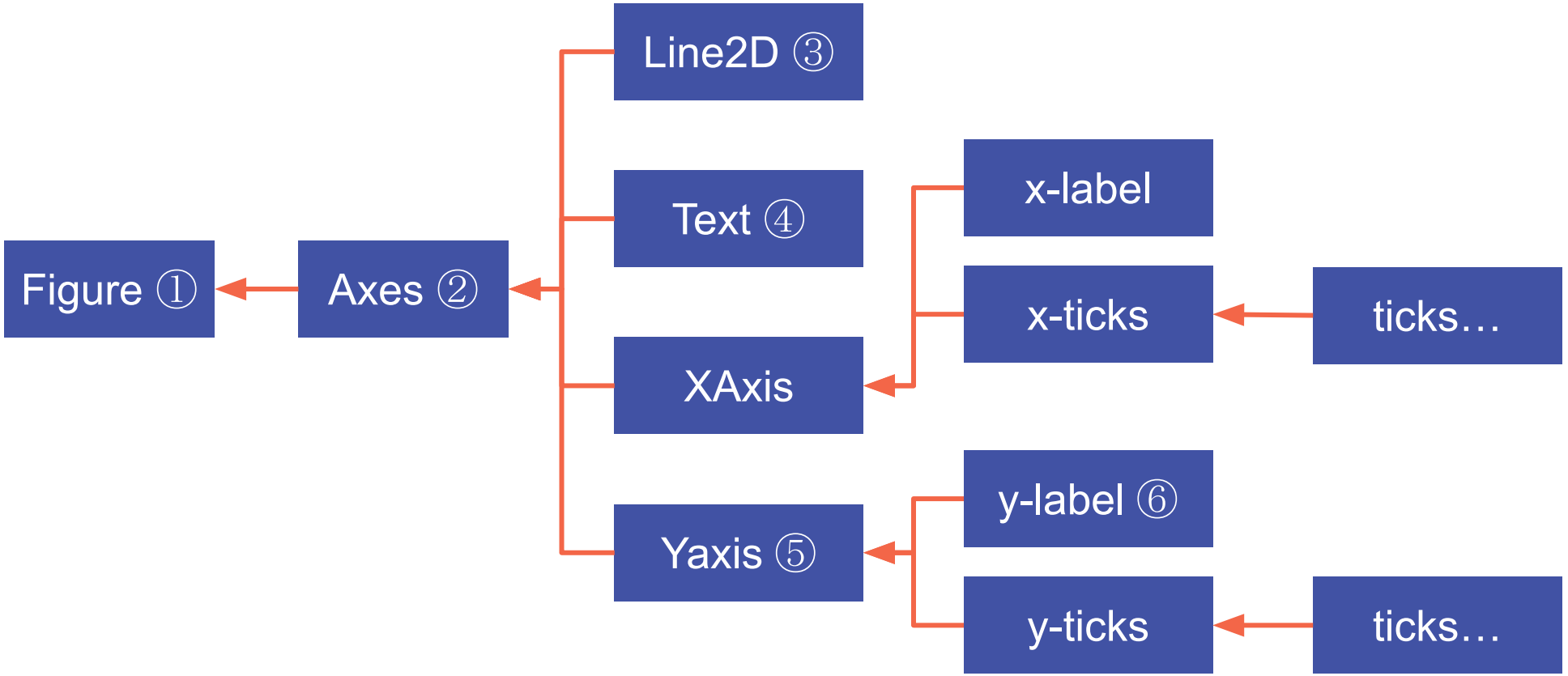
Artist Layer

Everything you see in a matplotlib Figure is an Artist instance





Artist Instances Hierarchy



Artist Layer

```
import matplotlib.pyplot as plt

fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)

t = np.arange(0.0, 1.0, 0.01)
s = np.sin(2*np.pi*t)
line, = ax.plot(t, s, color='blue', lw=2)
```

Try

```
>>> ax.lines[0]
```

Remove instances (one or the other)

```
del ax.lines[0]

ax.lines.remove(line)
```

Two Interfaces

MATLAB-style interface

```
plt.figure()  # create a plot figure

# create the first of two panels and
set current axis)
# (rows, columns, panel number)
plt.subplot(2, 1, 1)
plt.plot(x, np.sin(x))

plt.subplot(2, 1, 2)
plt.plot(x, np.cos(x));
```

Object-oriented interface

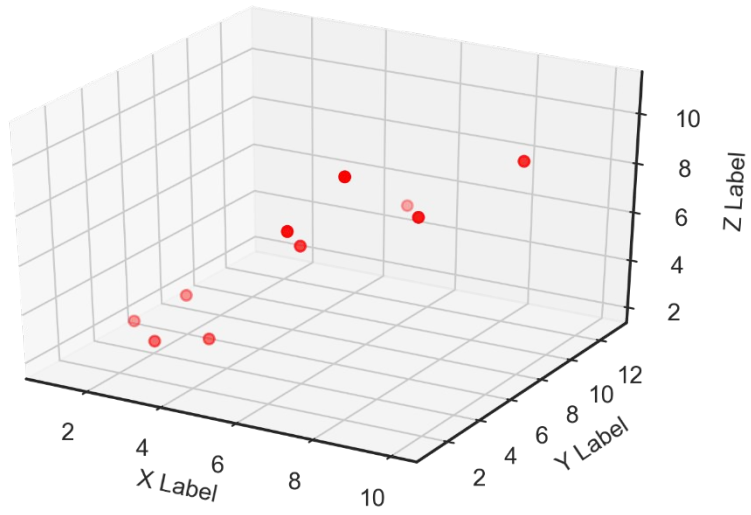
```
# First create a grid of plots
# ax will be an array of two Axes obje
cts
fig, ax = plt.subplots(2)

# Call plot() method on the appropriat
e object
ax[0].plot(x, np.sin(x))
ax[1].plot(x, np.cos(x));
```

Matplotlib Gotchas

`plt.plot()`	→	`ax.plot()`
`plt.legend()`	→	`ax.legend()`
`plt.xlabel()`	→	`ax.set_xlabel()`
`plt.ylabel()`	→	`ax.set_ylabel()`
`plt.xlim()`	→	`ax.set_xlim()`
`plt.ylim()`	→	`ax.set_ylim()`
`plt.title()`	→	`ax.set_title()`

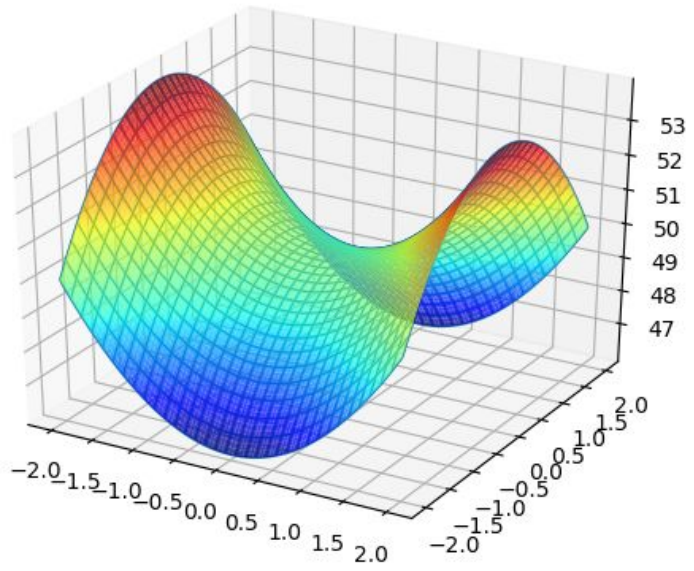
3D Scatter Plot

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x =[1,2,3,4,5,6,7,8,9,10]
y =[5,6,2,3,13,4,1,2,4,8]
z =[2,3,3,3,5,7,9,11,9,10]

ax.scatter(x, y, z, c='r', marker='o')

ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')
```



3D surface

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
...
Z = 50 + X**2 - Y**2
ax.plot_surface(X, Y, Z, cmap=cm.rainbow)
```



How to construct X and Y?

```
X = np.arange(-5, 5, 0.25)
Y = np.arange(-5, 5, 0.25)
X, Y = np.meshgrid(X, Y)
```