Python

Introduction to Matplotlib

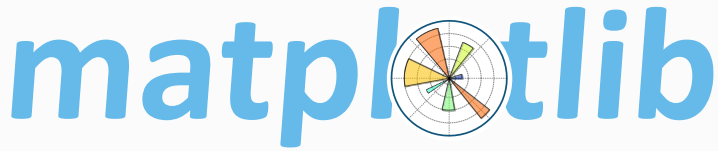
In this lecture

- Introduction to Matplotlib
- Functional approach
  - Basic plots
  - Subplots
- OOP approach
  - Figures and Axes
  - Scatter plots
  - Pie Charts
  - Bar Graphs and Histograms
  - Boxplots

Matplotlib

Matplotlib

- **Matplotlib** is a package that plots visualisations in Python.
- It can generate Line plots, Histogram, Scatter Plots, 3D plots and plot images by mapping pixels to coordinates on x and y axes.
- It provides an Object-Oriented API – applied by Tkinter– and has also kept its legacy procedural "pylab" interface, which resembles MATLAB from the 1980s
- More documentation available at: https://matplotlib.org



Functional approach

Imports

```
In[ ]: 1 | import pandas as pd
       2 | import numpy as np
       3 | import matplotlib.
```

```
matplotlib.afm
matplotlib.animation
matplotlib.artist
matplotlib.axes
matplotlib.axis
matplotlib.backend_bases
matplotlib.backend_managers
matplotlib.backend_tools
matplotlib.backends
matplotlib.bezier
```

Imports

In[ ]:

```
1 | import pandas as pd
2 | import numpy as np
3 | import matplotlib.
```

```
matplotlib.patches
matplotlib.path
matplotlib.patheffects
matplotlib.projections
matplotlib.pylab
matplotlib.pyplot
matplotlib.quiver
matplotlib.rcsetup
matplotlib.sankey
matplotlib.scale
```

Imports

In[ ]:

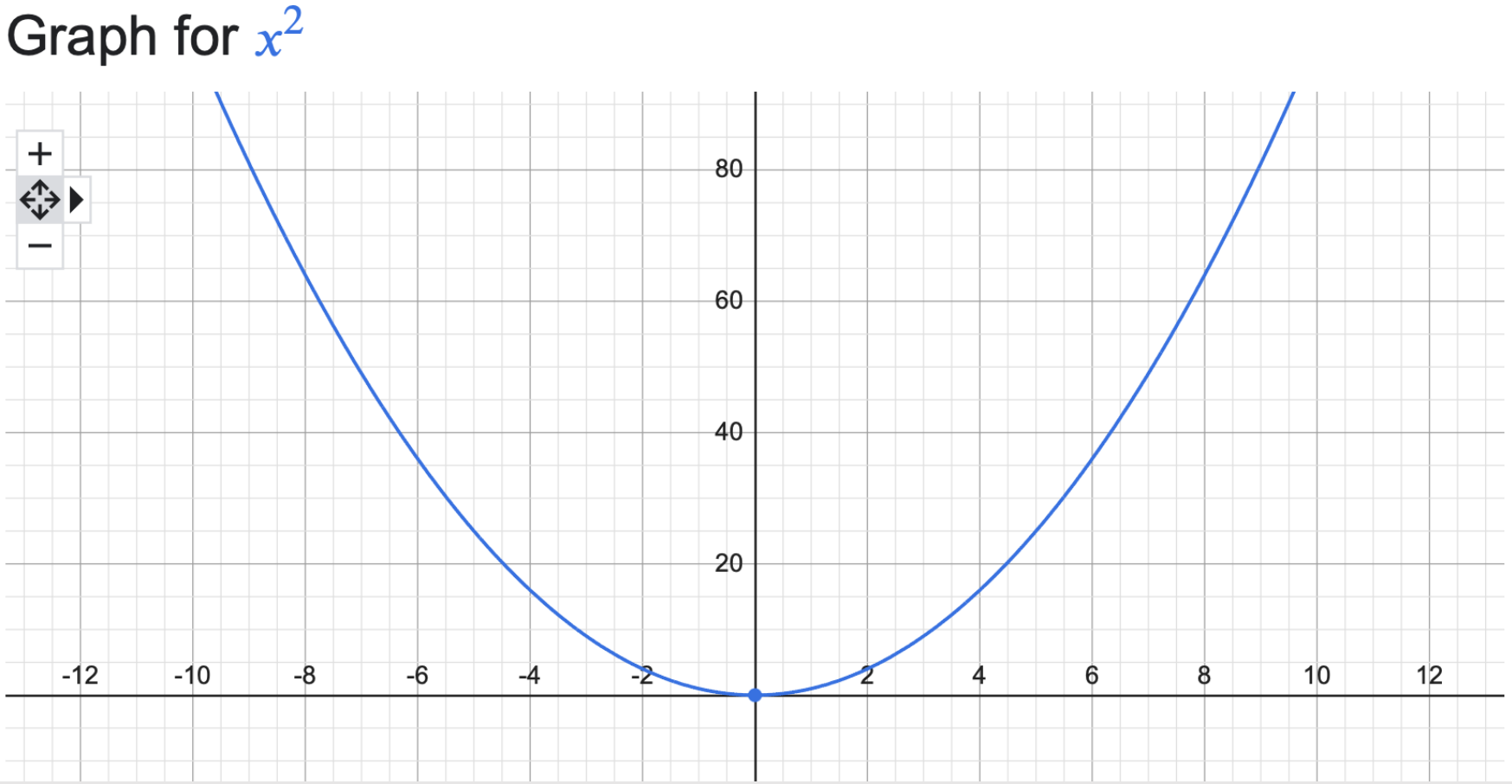
```
1 | import pandas as pd
2 | import numpy as np
3 | import matplotlib.pyplot as plt
```

Imports

In[ ]:
```
1 | import pandas as pd
2 | import numpy as np
3 | import matplotlib.pyplot as plt
```

Written in the first cell to reduce duplication across the following cells.

Graph for quadratic function $y=x^2$



Set up x axis

In[ ]:
```
1 | x = np.arange(1, 10, 1)
2 | x
```

Set up x axis

In[ ]:
```
1 | x = np.arange(1, 10, 1)
2 | x
```

Out[]:
```
array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

Set up x & y axes

In[ ]:
```
1 | x = np.arange(1, 10, 1)
2 | y = x**2
3 | y
```

Set up x & y axes

In[ ]:
```
1 | x = np.arange(1, 10, 1)
2 | y = x**2
3 | y
```

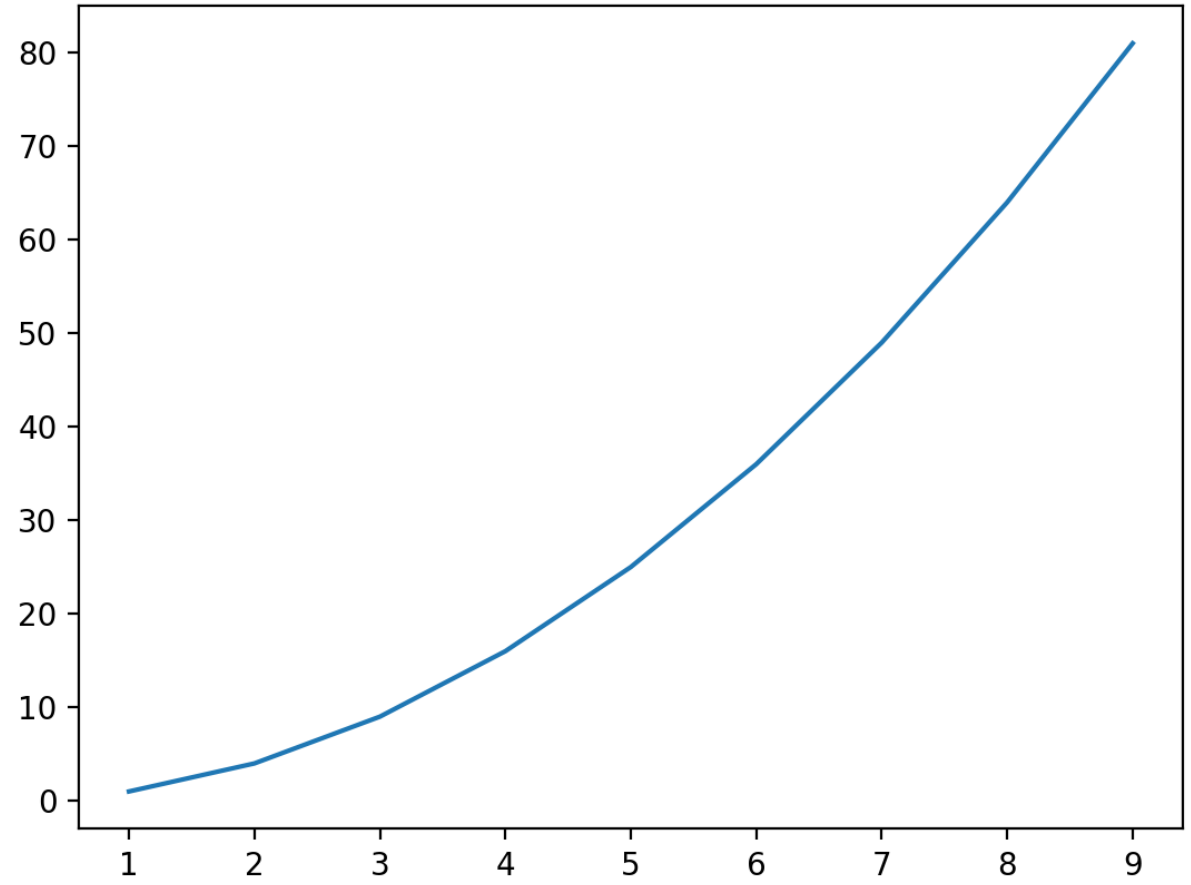
Out[]: array([1, 4, 9, 16, 25, 36, 49, 64, 81])

Plot x and y

In[ ]:
```
1 | plt.plot(x, y)
2 | plt.show()
3 |
```

Basic plot

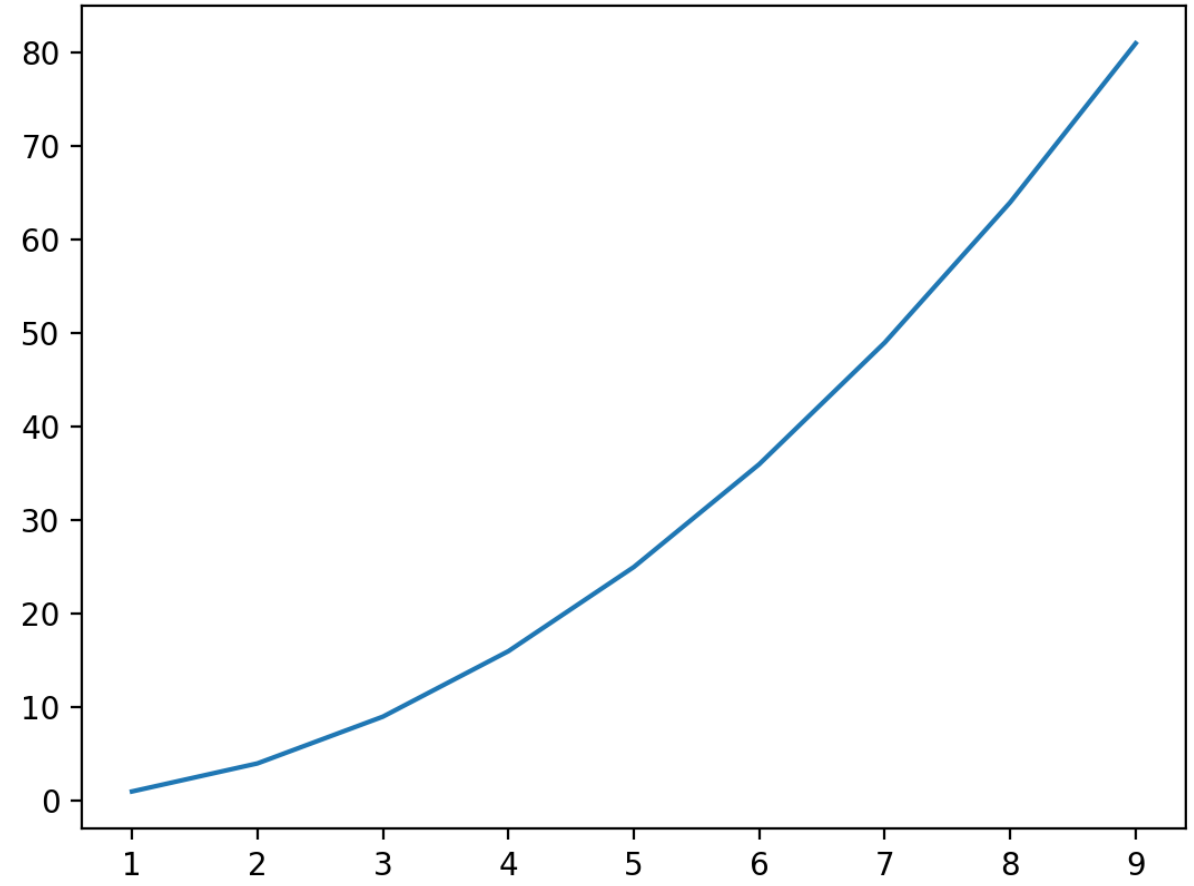
Out[]:



Basic plot

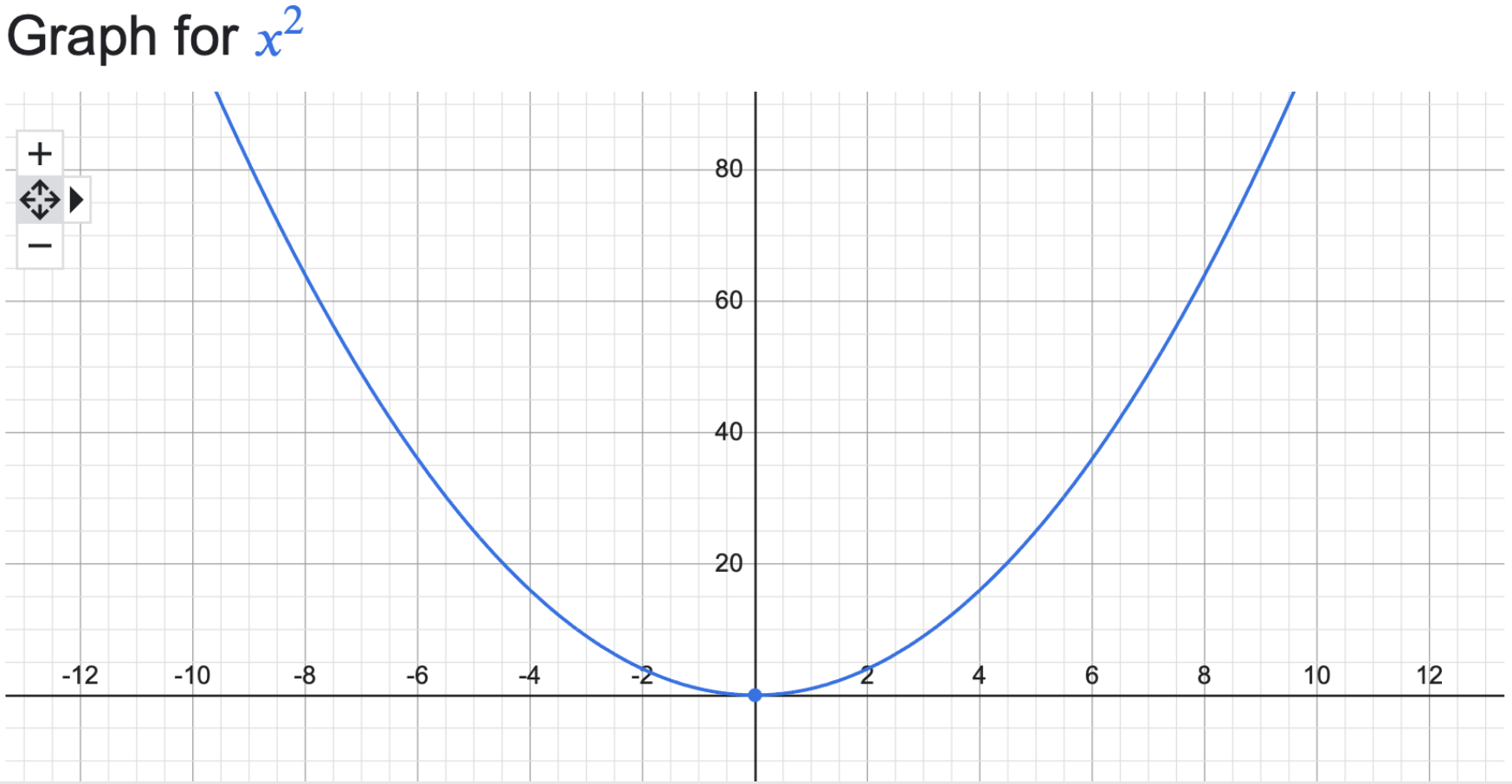
Out[]:

y:
81
64
49
36
25
16
9
4
1



x : array([1, 2, 3, 4, 5, 6, 7, 8, 9])

Graph for quadratic function $y=x^2$



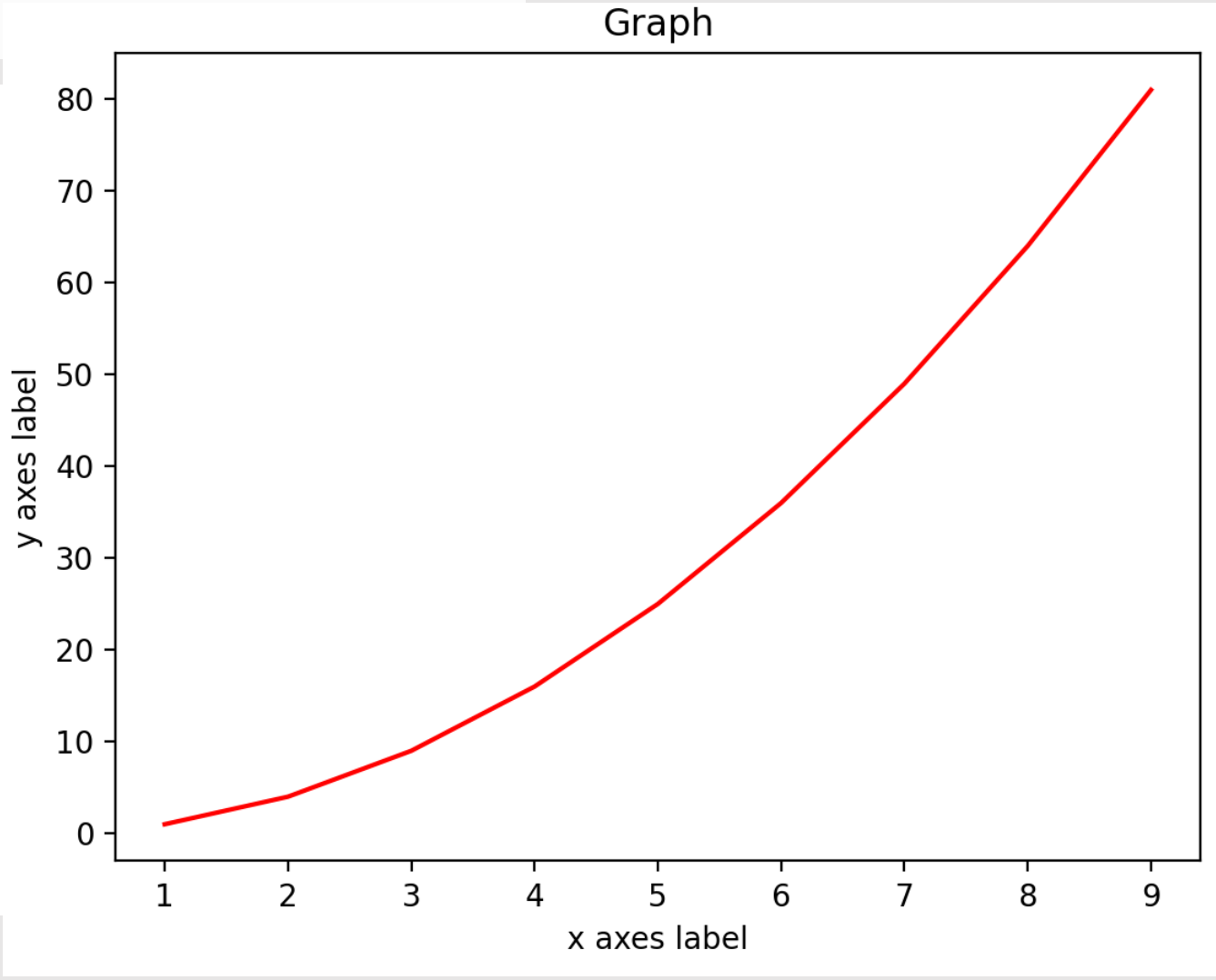
Add labels

In[ ]:

```
1 | plt.plot(x, y, 'r') # r for red- matlab
2 |
3 | plt.xlabel('x axes label')
4 | plt.ylabel('y axes label')
5 | plt.title('Graph')
6 |
7 | plt.show()
```

Labels and red

Out[]:



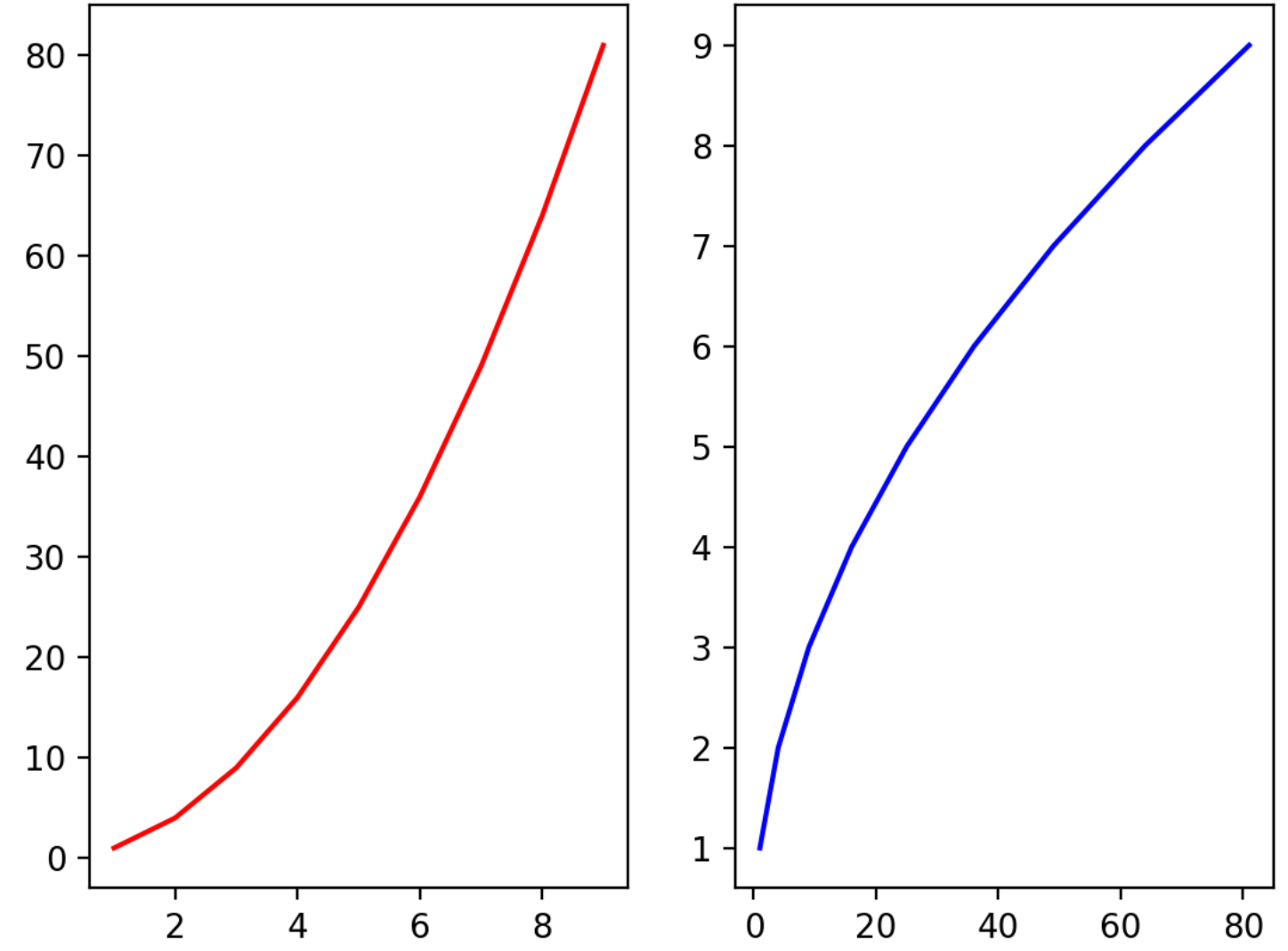
Subplots

In[ ]:

```
plt.subplot(1, 2, 1)
plt.plot(x, y, 'r') # r for red- matlab

plt.subplot(1, 2, 2)
plt.plot(y, x, 'b') # b for blue-matlab

plt.show()
```

Sub plots

Out[]:



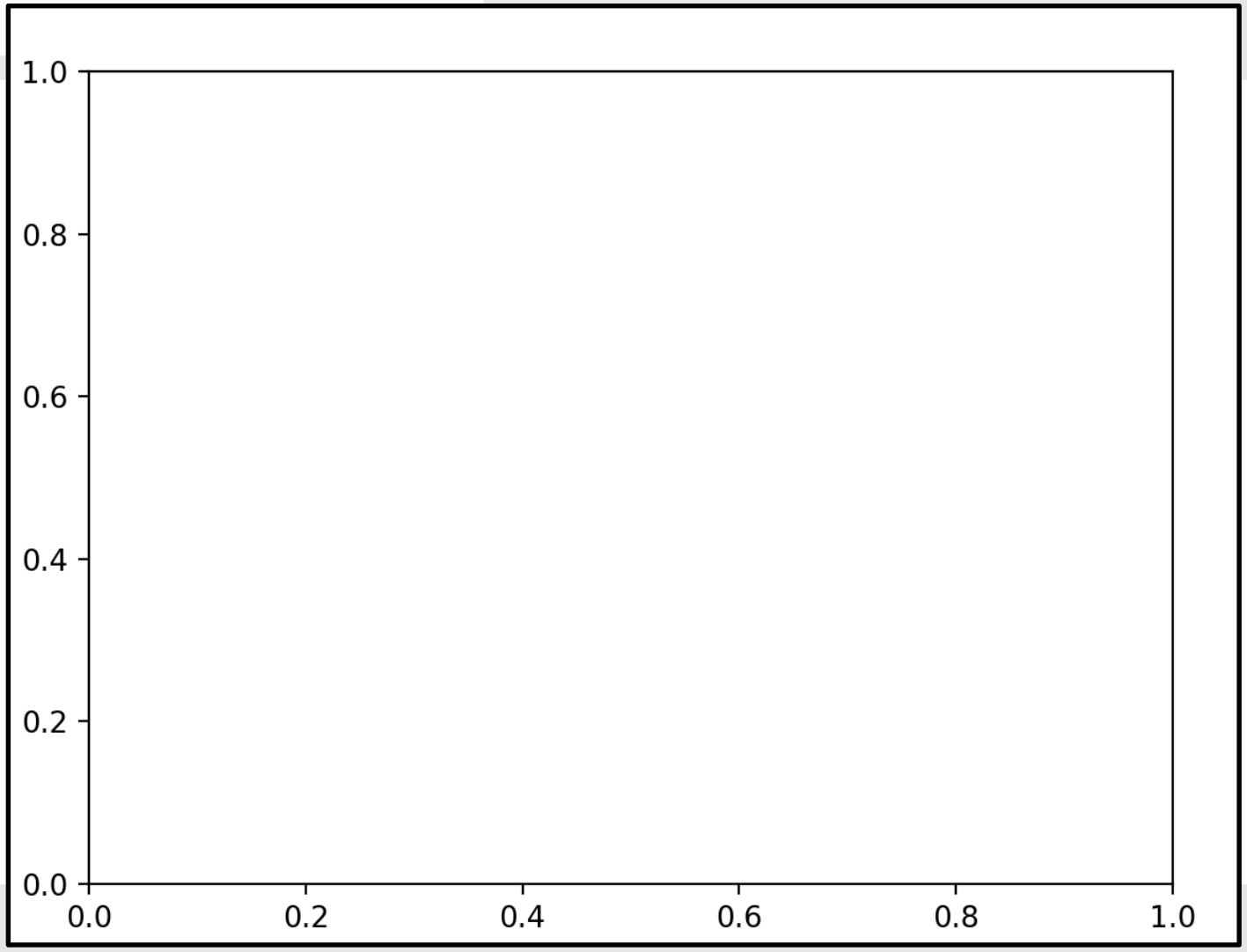
OOP approach

Set up the Figure

In[ ]:
```
fig = plt.figure() #blank canvas

axes = fig.add_axes([0.1, 0.1, 0.8,0.8])
                    #left, bottom, width, height
```

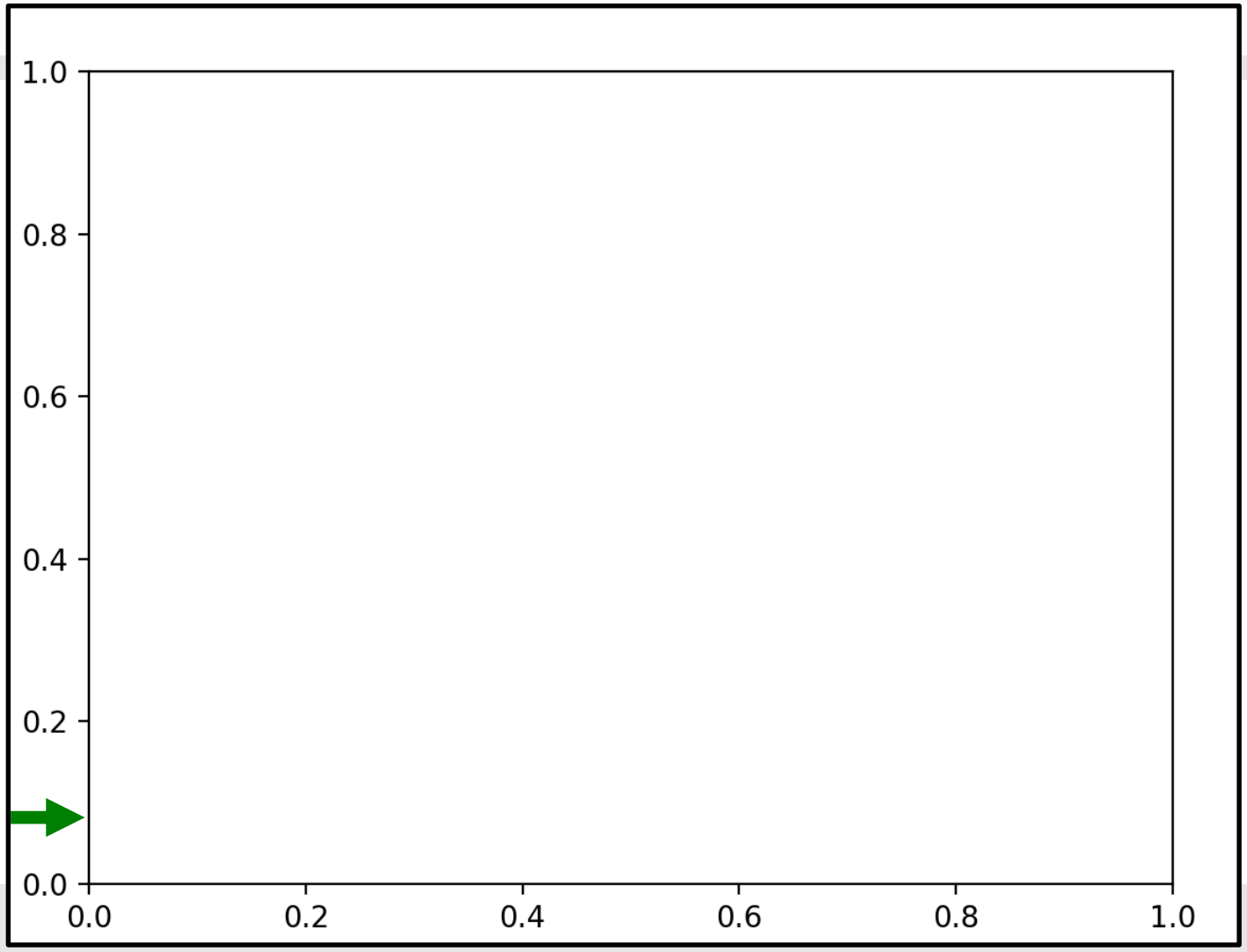
Blank Canvas

```
Out[]:
```



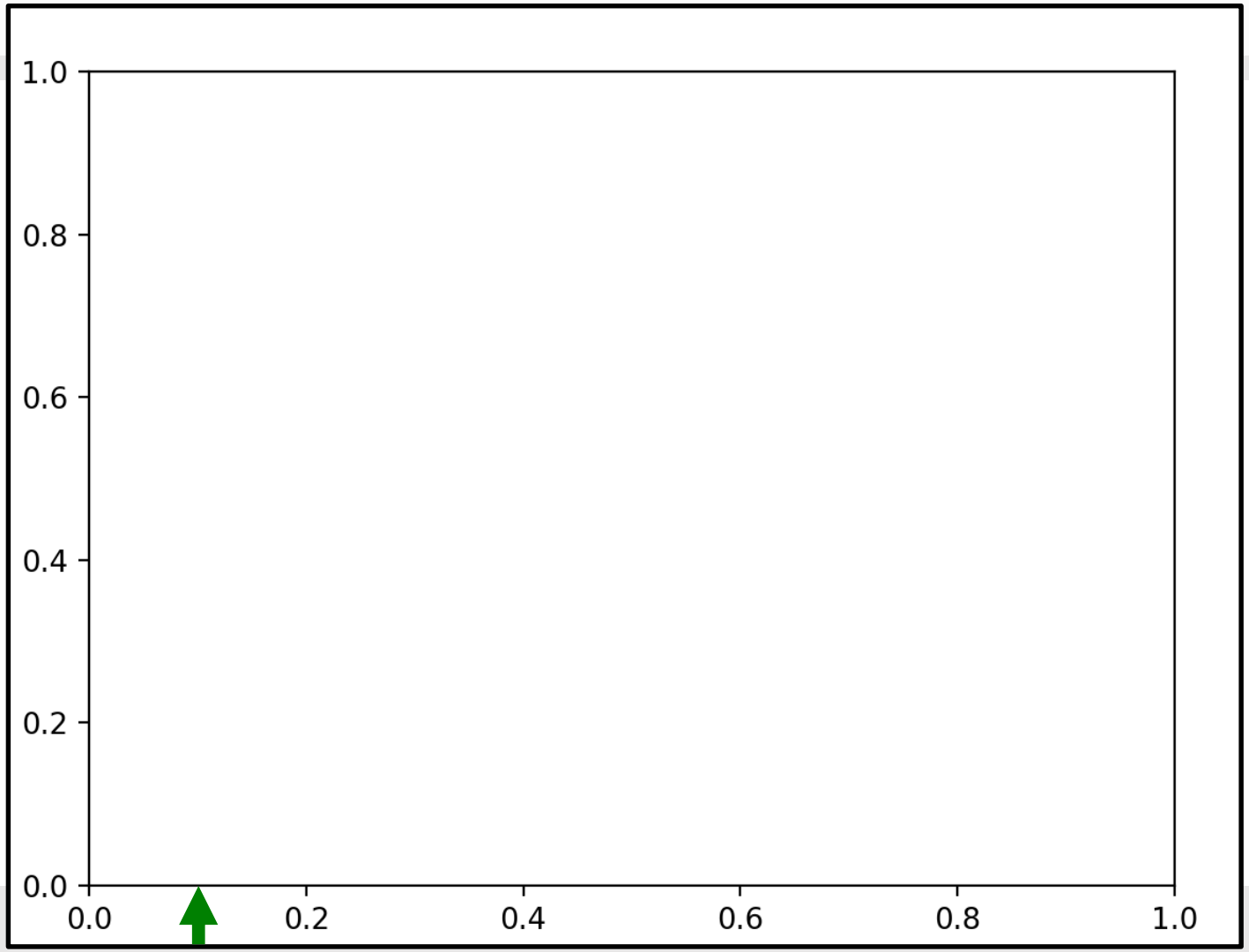
```
fig.add_axes([0.1, 0.1, 0.8, 0.8])
```

Out[]:



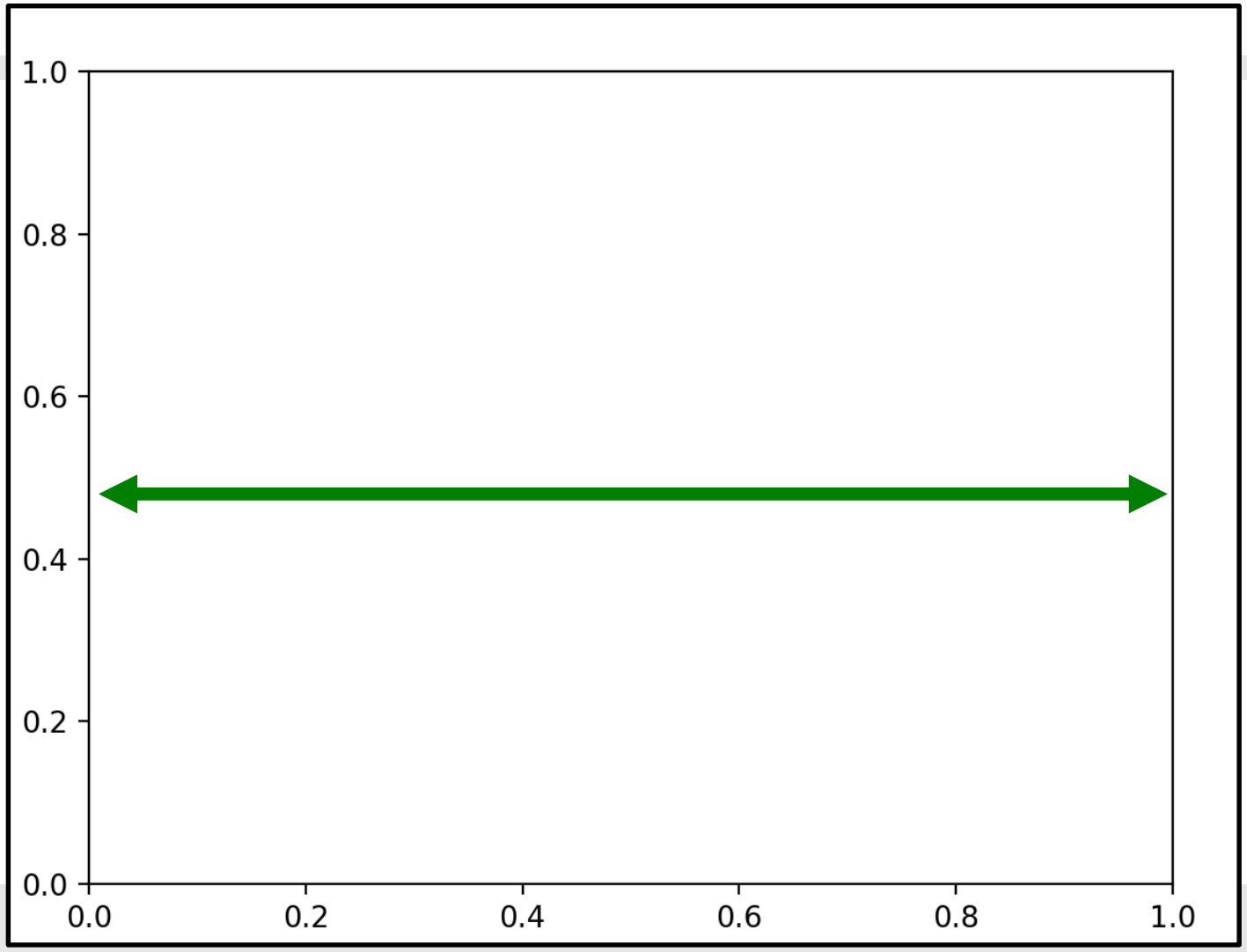
```
fig.add_axes([0.1, 0.1, 0.8, 0.8])
```

Out[]:



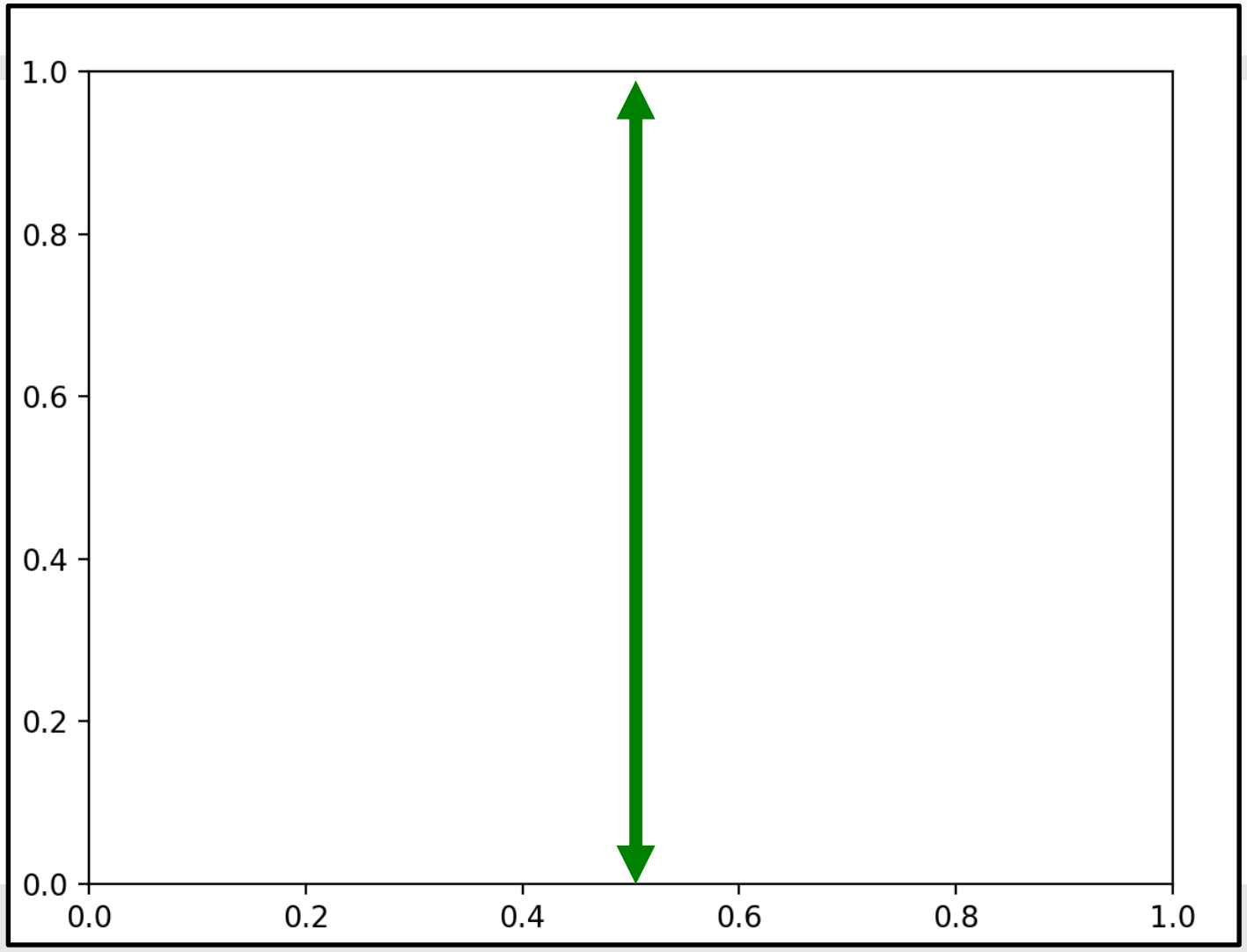
```
fig.add_axes([0.1, 0.1, 0.8, 0.8])
```

Out[]:



```
fig.add_axes([0.1, 0.1, 0.8, 0.8])
```

Out[]:



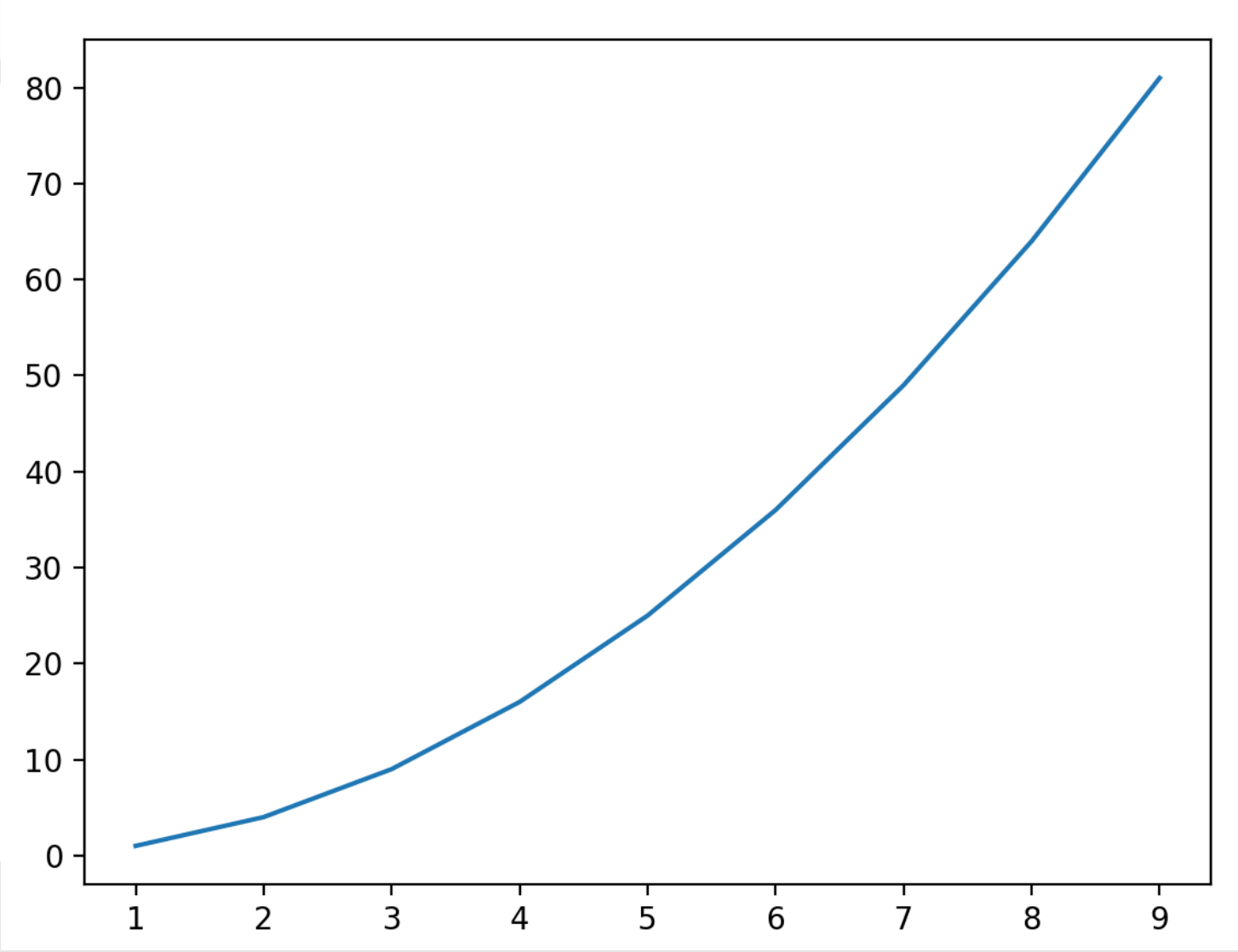
Set up the Figure

In[ ]:

```
1 | fig = plt.figure() #blank canvas
2 |
3 | axes = fig.add_axes([0.1, 0.1, 0.8,0.8])
4 |                     #left, bottom, width, height
5 | axes.plot(x, y)
```

With data

```
Out[]:
```

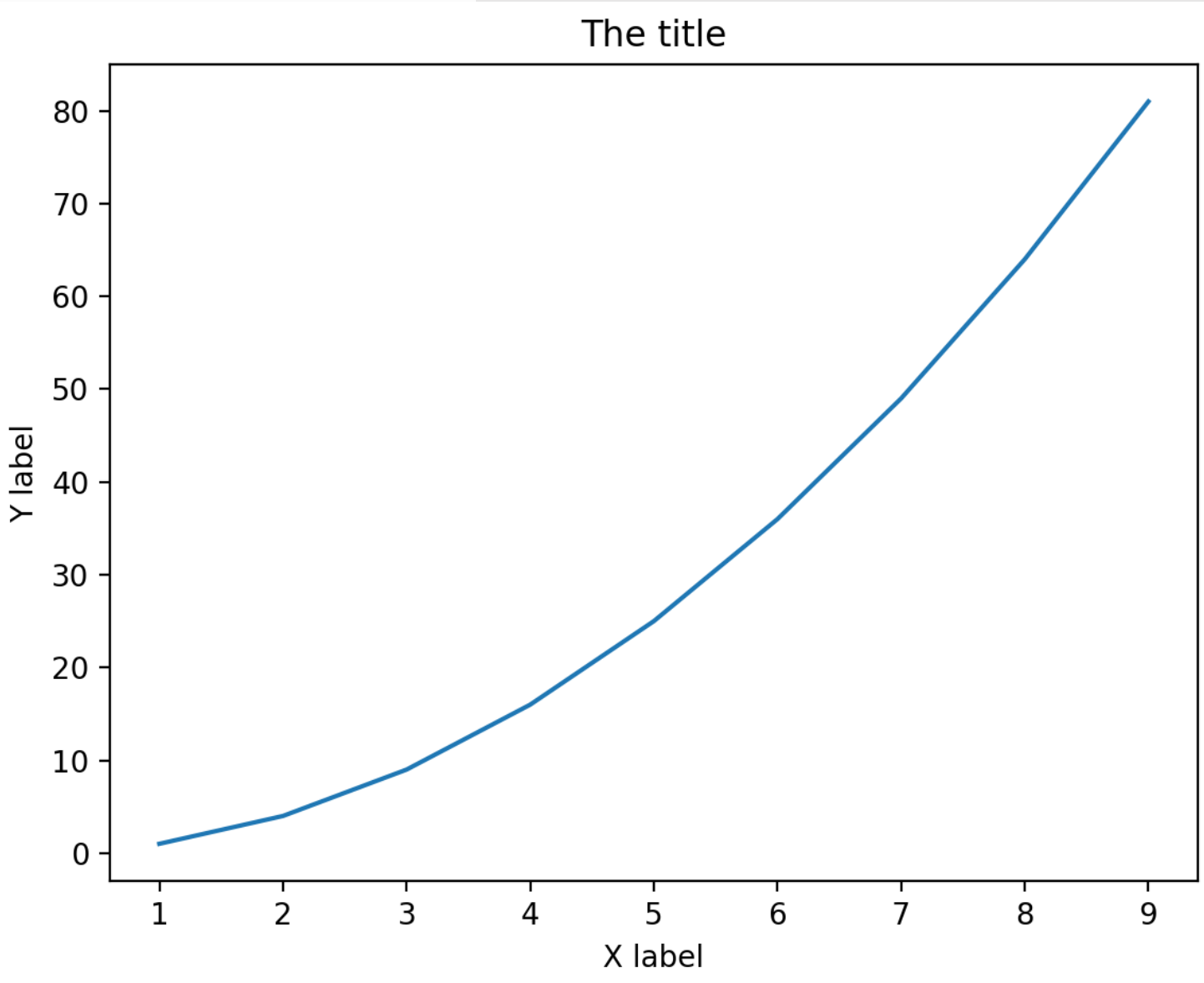


Add labels

In[ ]:
```
1 | axes.set_xlabel('X label’)
2 | axes.set_ylabel('Y label’)
3 | axes.set_title('The title')
```

With labels

```
Out[]:
```

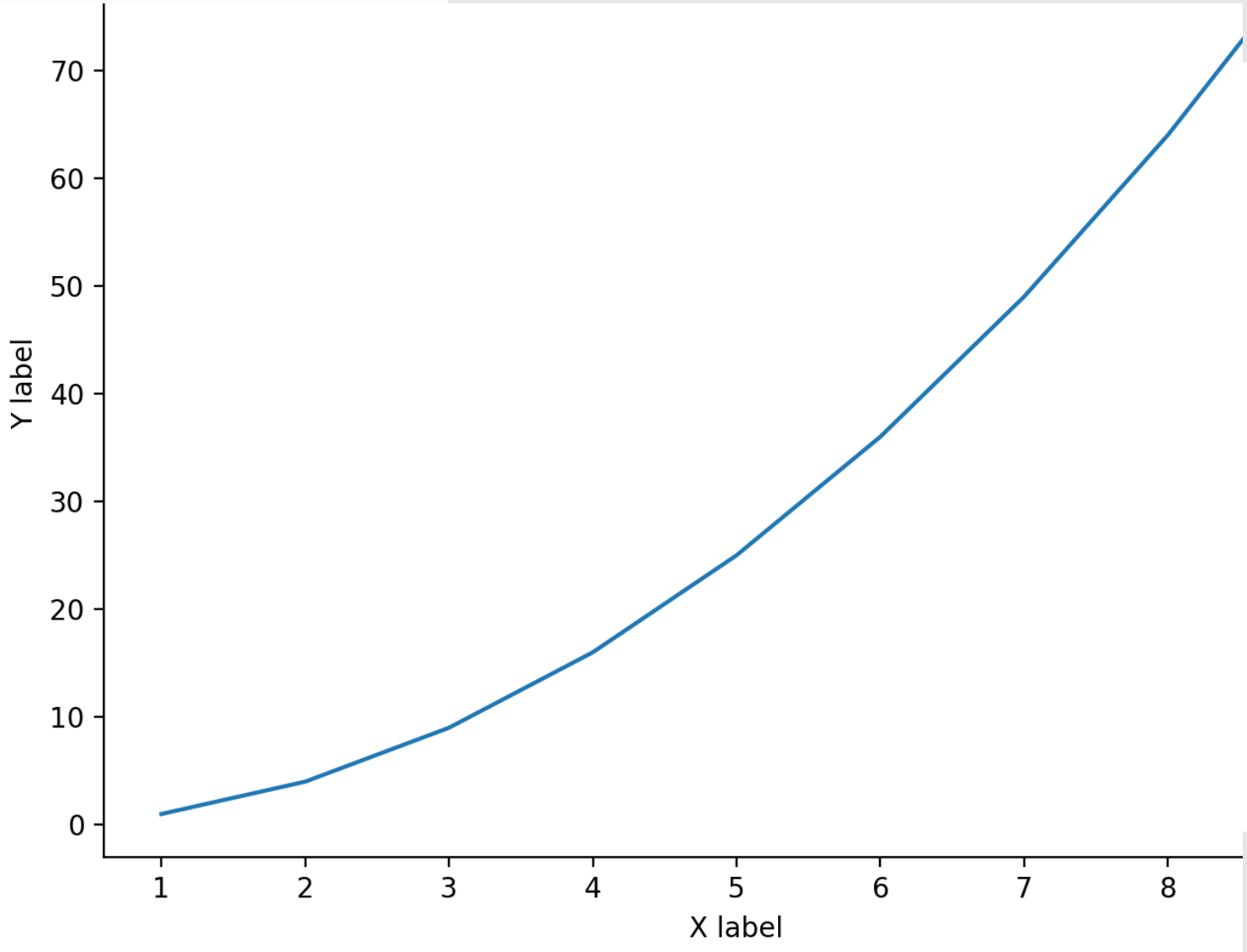


Adjust the frame

In[ ]:
```
1 | fig = plt.figure() #blank canvas
2 |
3 | axes = fig.add_axes([0.1, 0.1, 1, 1])
4 |                     #left, bottom, width, height
```

Lost the title…

Out[]:

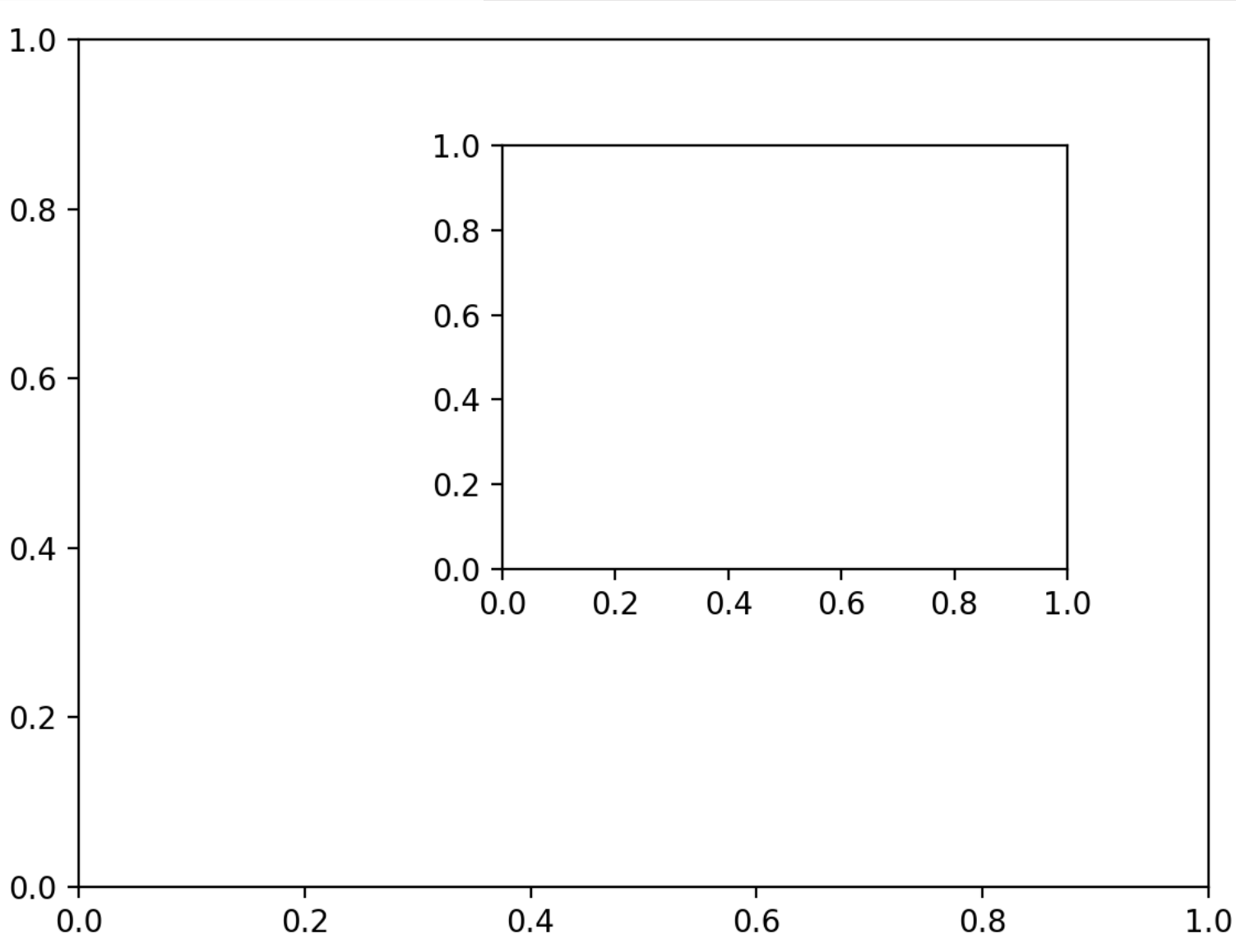


Setting up two plots

In[ ]:
```
1 | fig = plt.figure() #blank canvas
2 |
3 | axes = fig.add_axes([0.1, 0.1, 0.8,0.8])
4 | axes = fig.add_axes([0.4, 0.4, 0.4,0.4])
```

Subplots

Out[]:



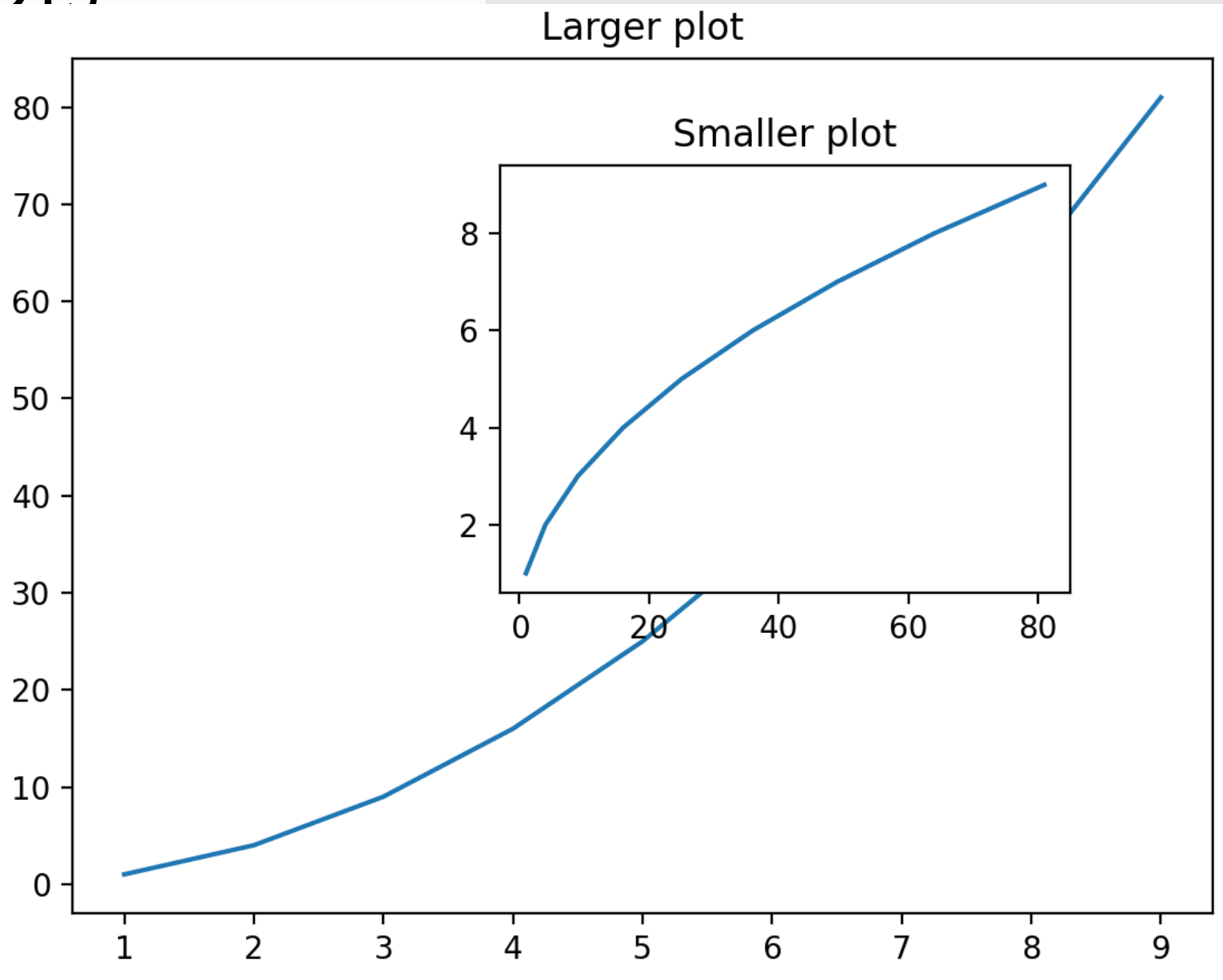
Setting up two plots

In[ ]:

```
1 | axes1.plot(x,y)
2 | axes2.plot(y,x)
3 |
4 | axes1.set_title('Larger plot')
5 | axes2.set_title('Smaller plot')
```

Subplots

Out[]:

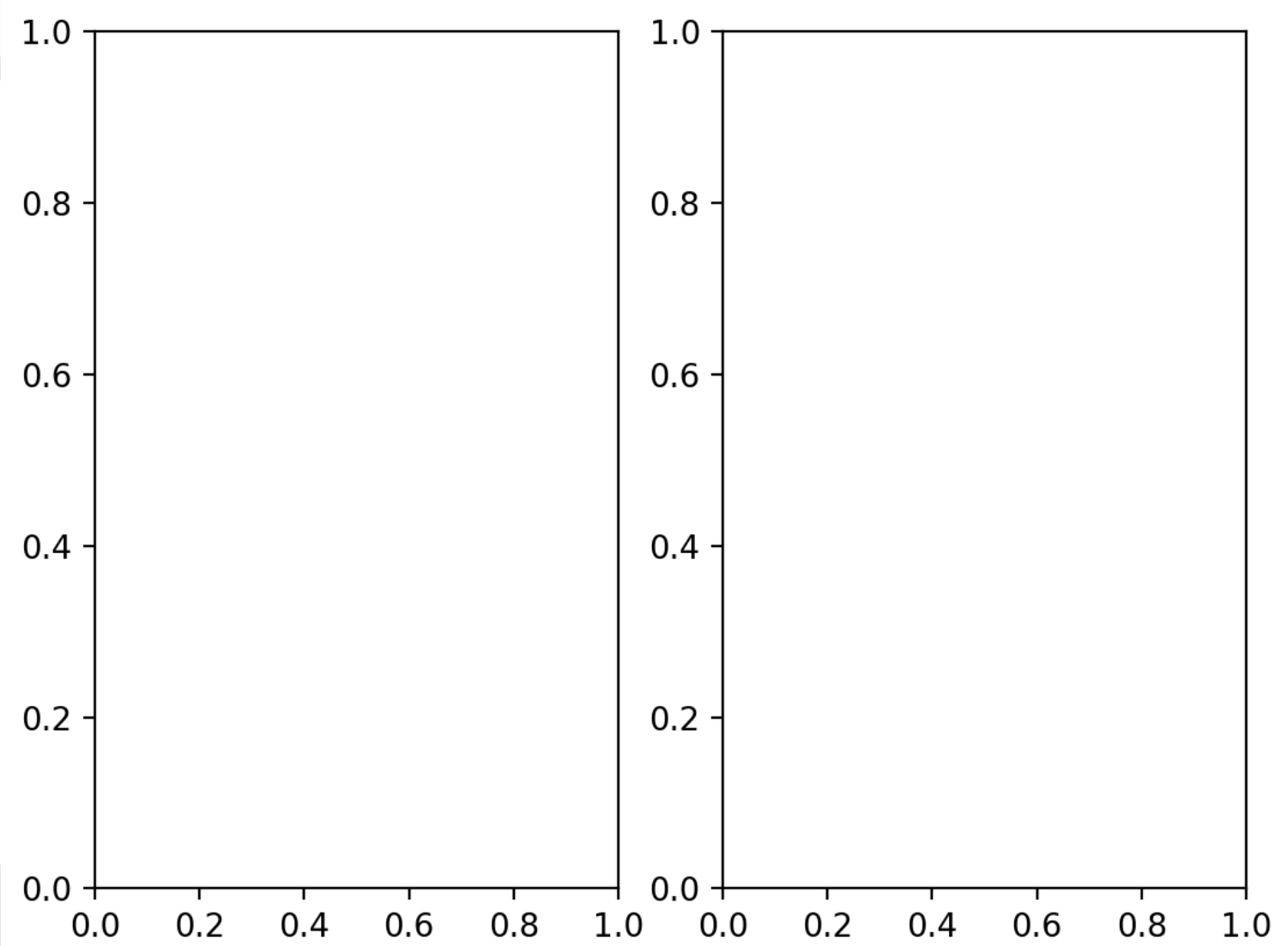


Separate plots

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=1, ncols=2)
2 |
```

Separate plots

Out[]:

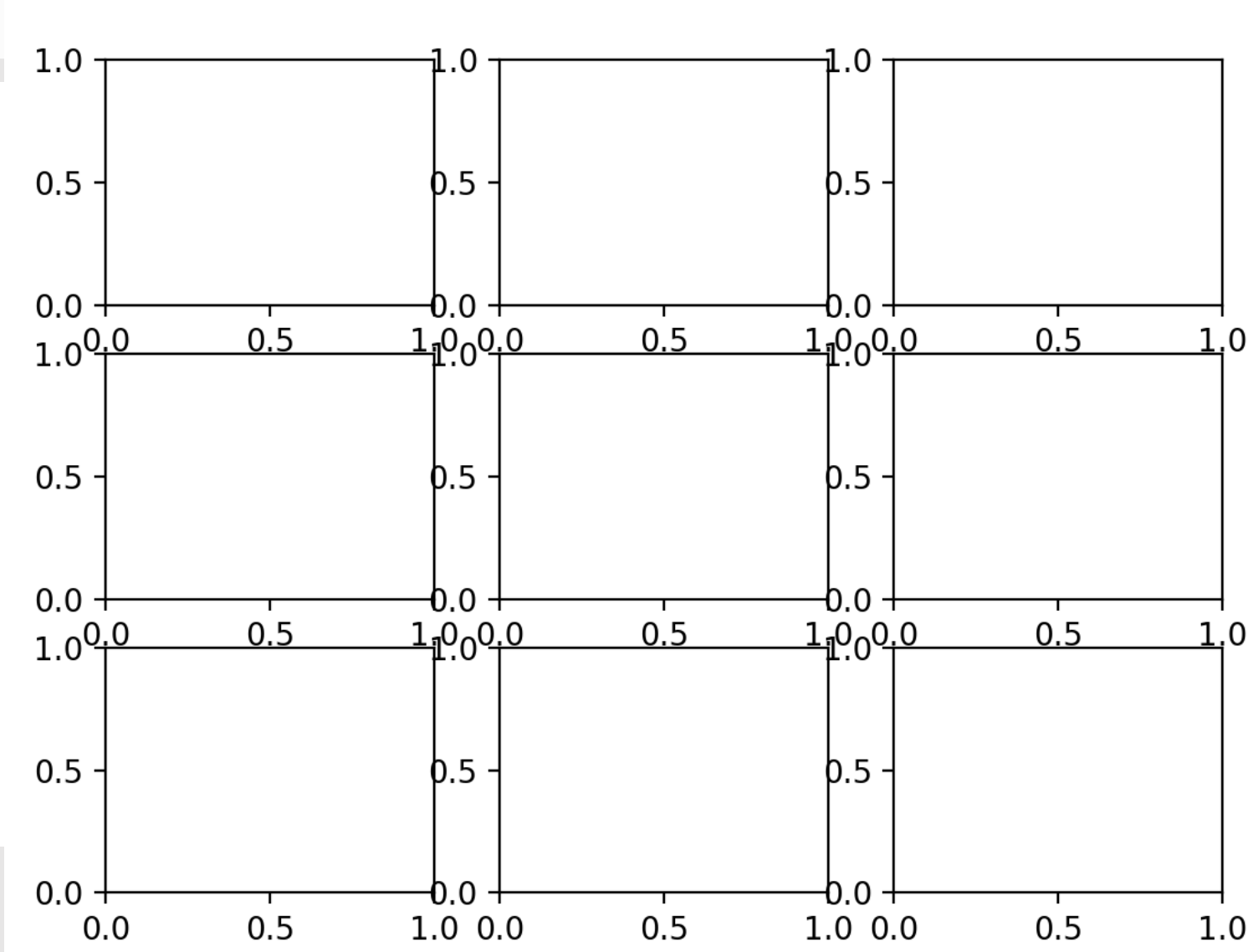


Separate plots

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 |
```

Separate plots

```
Out[]:
```

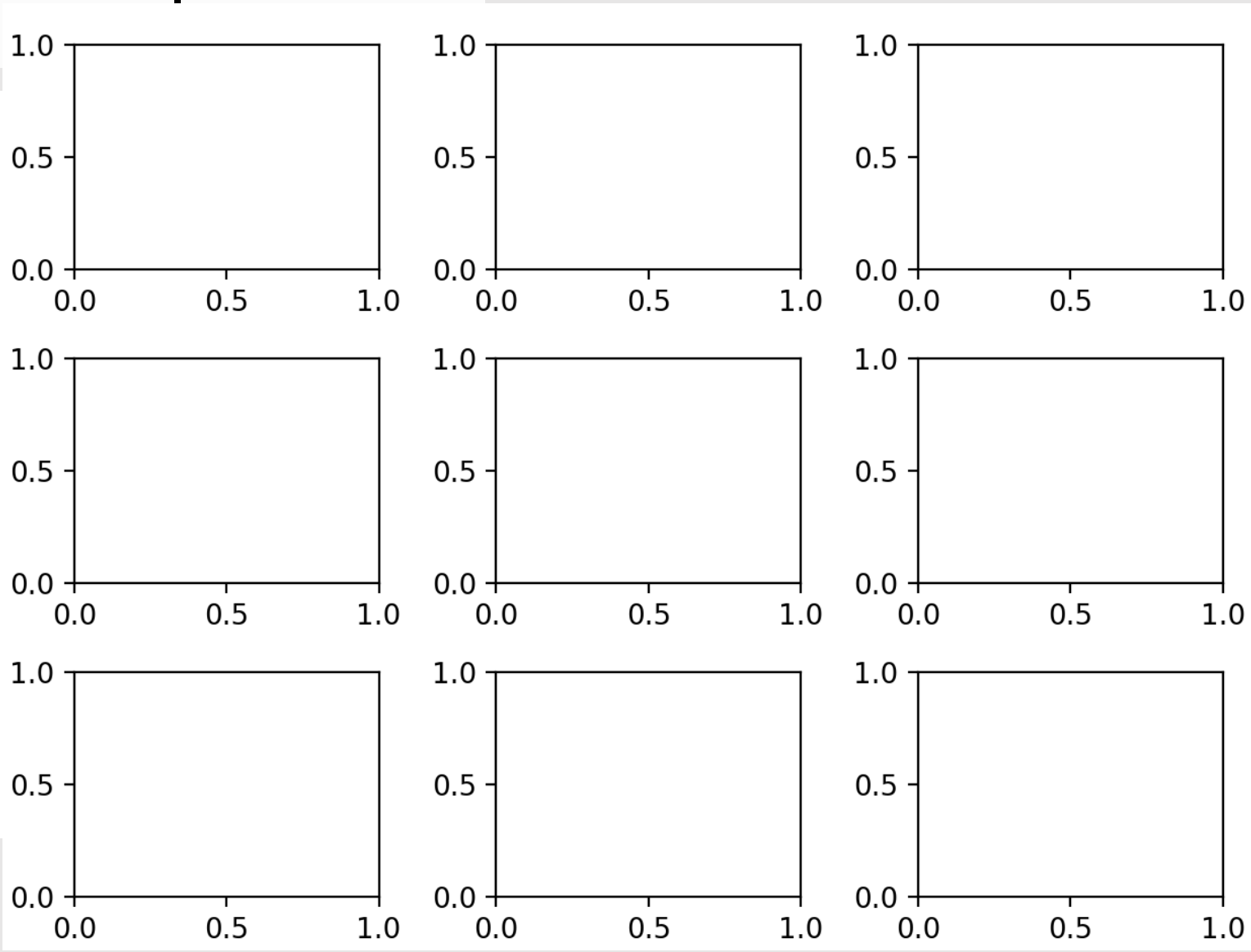


Separate plots

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 | plt.tight_layout()
3 |
```

Separate plots

Out[]:



Separate plots

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 | plt.tight_layout()
3 |
```

Axes

Axes

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 | axes
3 |
```

Axes

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 | axes
3 |
```

Out[]:
```
array([[<AxesSubplot:>,<AxesSubplot:>,<AxesSubplot:>],
       [<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>],
       [<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>]],
      dtype=object)
```

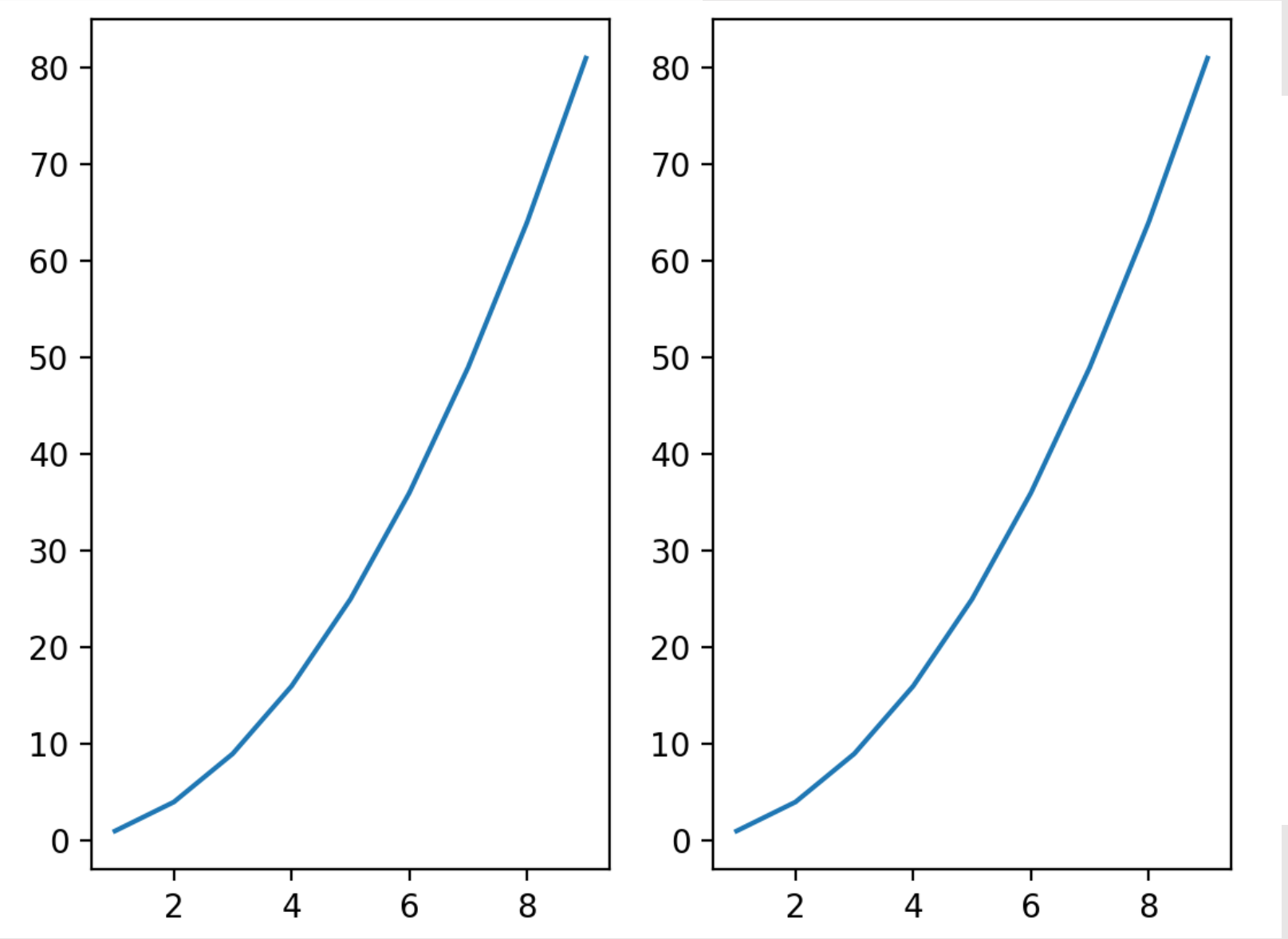
Iterating through axes

In[ ]:

```
1 | fig, axes = plt.subplots(nrows=1, ncols=2)
2 |
3 | for current_ax in axes:
4 |     current_ax.plot(x, y)
```

Iterating through axes

Out[]:



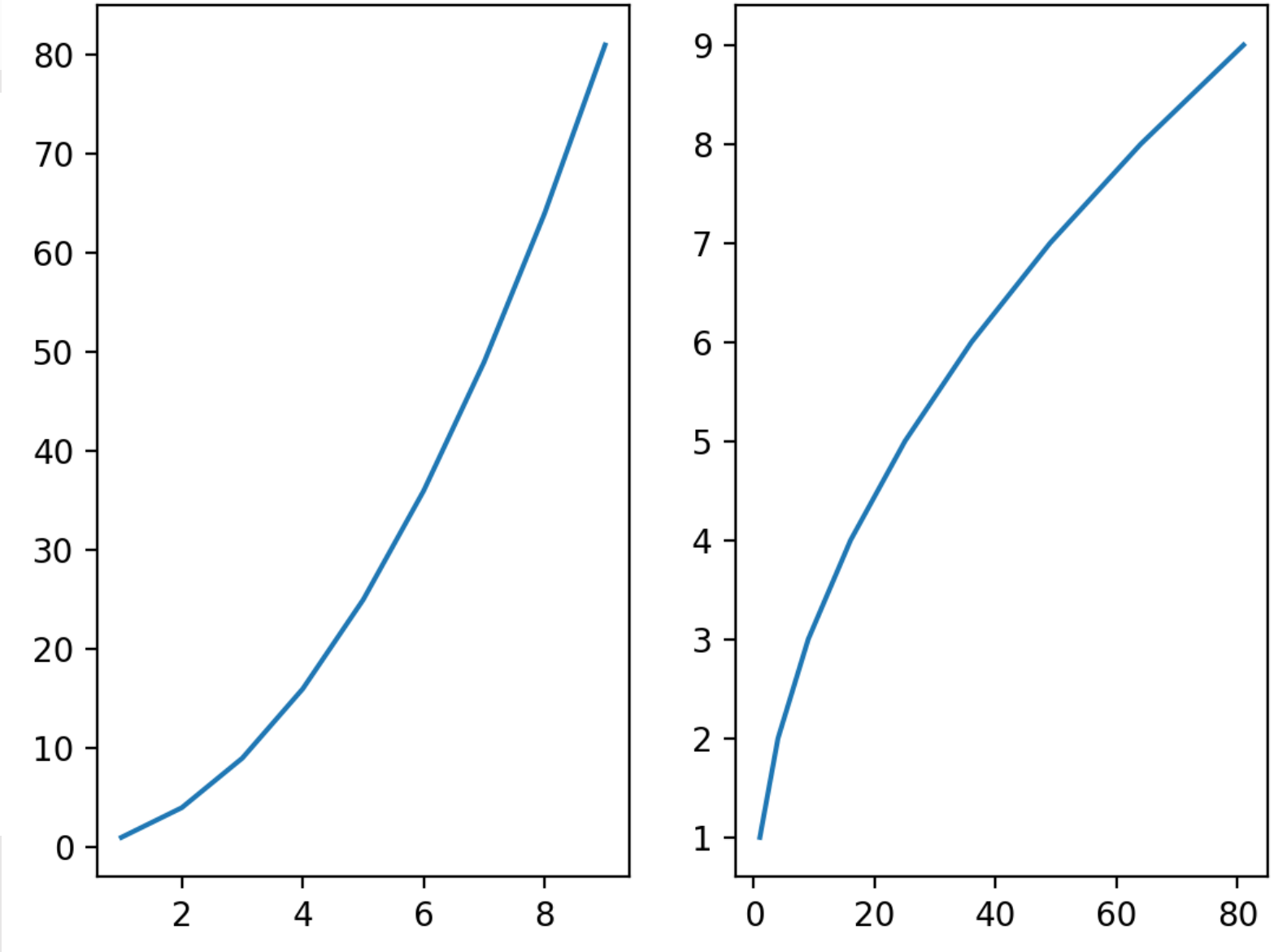
Axes subscript

In[ ]:

```
1 | fig, axes = plt.subplots(nrows=1, ncols=2)
2 |
3 | axes[0].plot(x,y)
4 | axes[1].plot(y,x)
```

Axes subscript

Out[]:

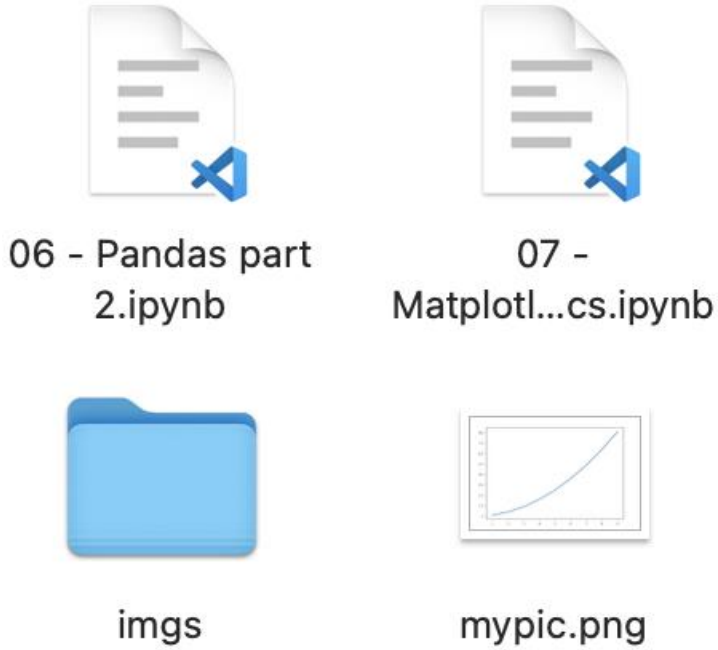


Export Figures

plt.savefig()

In[ ]:
```
1 | plt.plot(x, y)
2 | plt.show()
3 | plt.savefig('mypic.png', dpi=200)
```

Export


06 - Pandas part 2.ipynb
07 - Matplotl...cs.ipynb
imgs
mypic.png

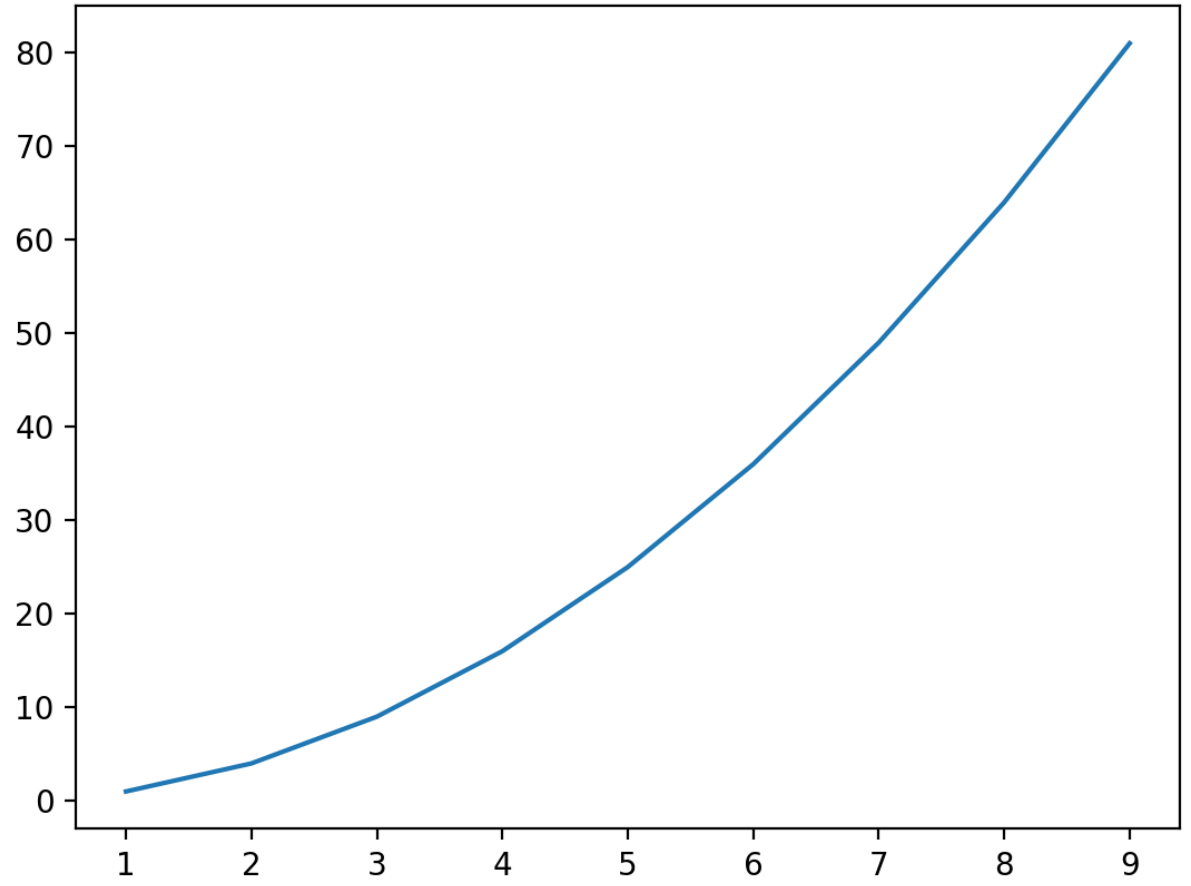

80
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9

fig.savefig()

In[ ]:
```
1 | fig, axes = plt.subplots(nrows=3, ncols=3)
2 |
3 | fig.savefig('mypic.png', dpi=200)
```

Figure Size

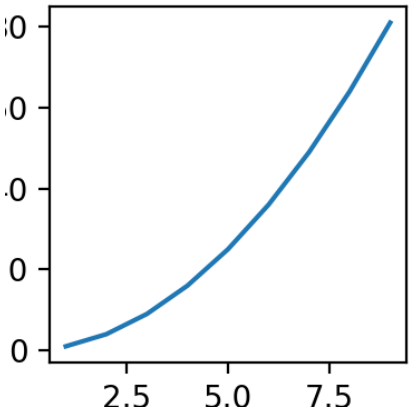
Figsize(2,2)

In[ ]:

```
fig = plt.figure(figsize=(2,2))
ax = fig.add_axes([0.1,0.1,0.8,0.8])
ax.plot(x,y)
```

Small figure

Out[]:



Figsize(8,8)

In[ ]:

```
1 | fig = plt.figure(figsize=(8,8))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 | ax.plot(x,y)
```

Large

Out[]:

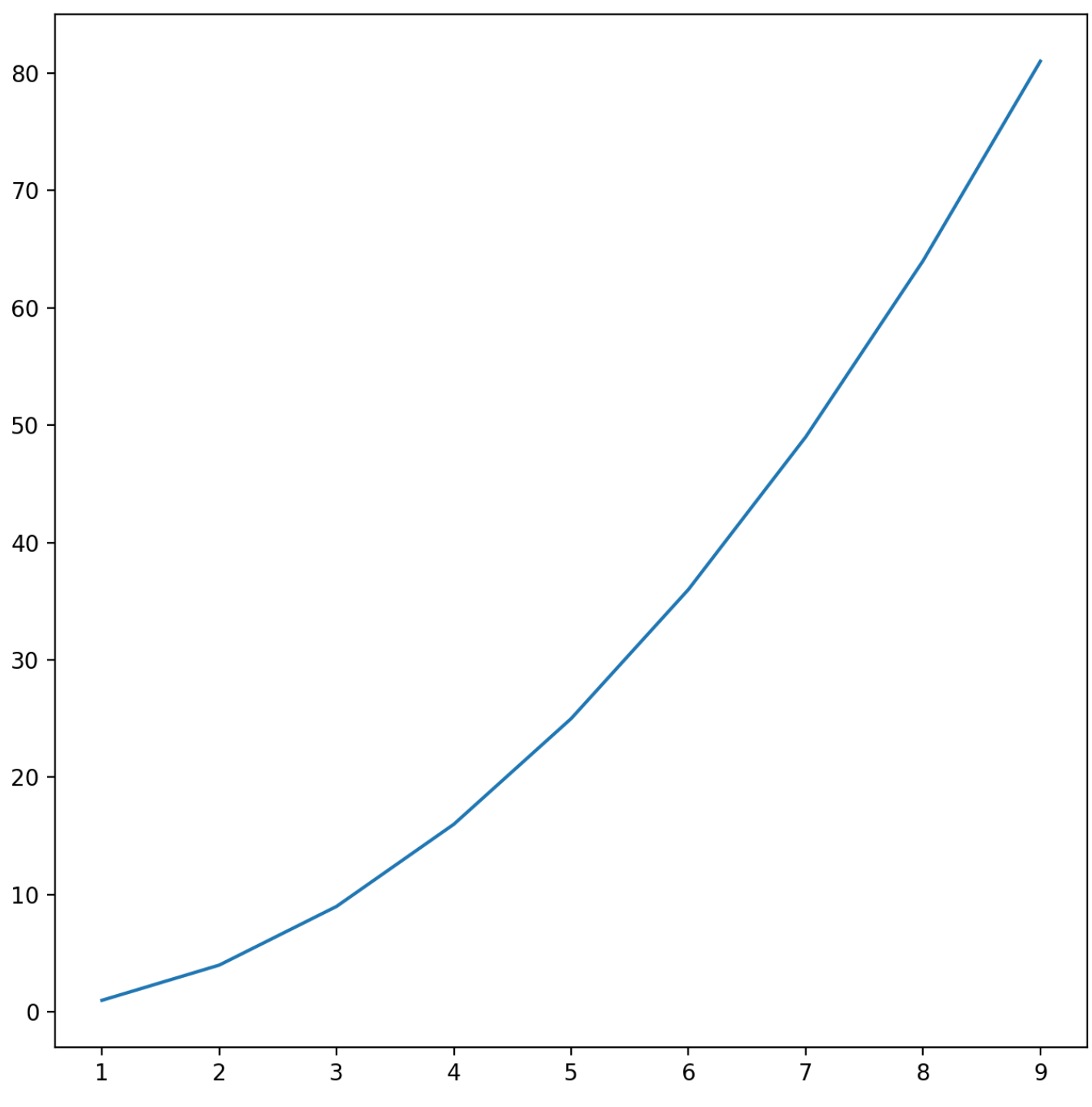
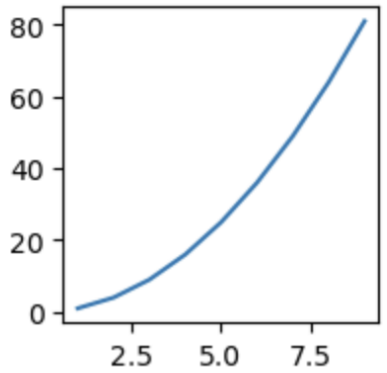
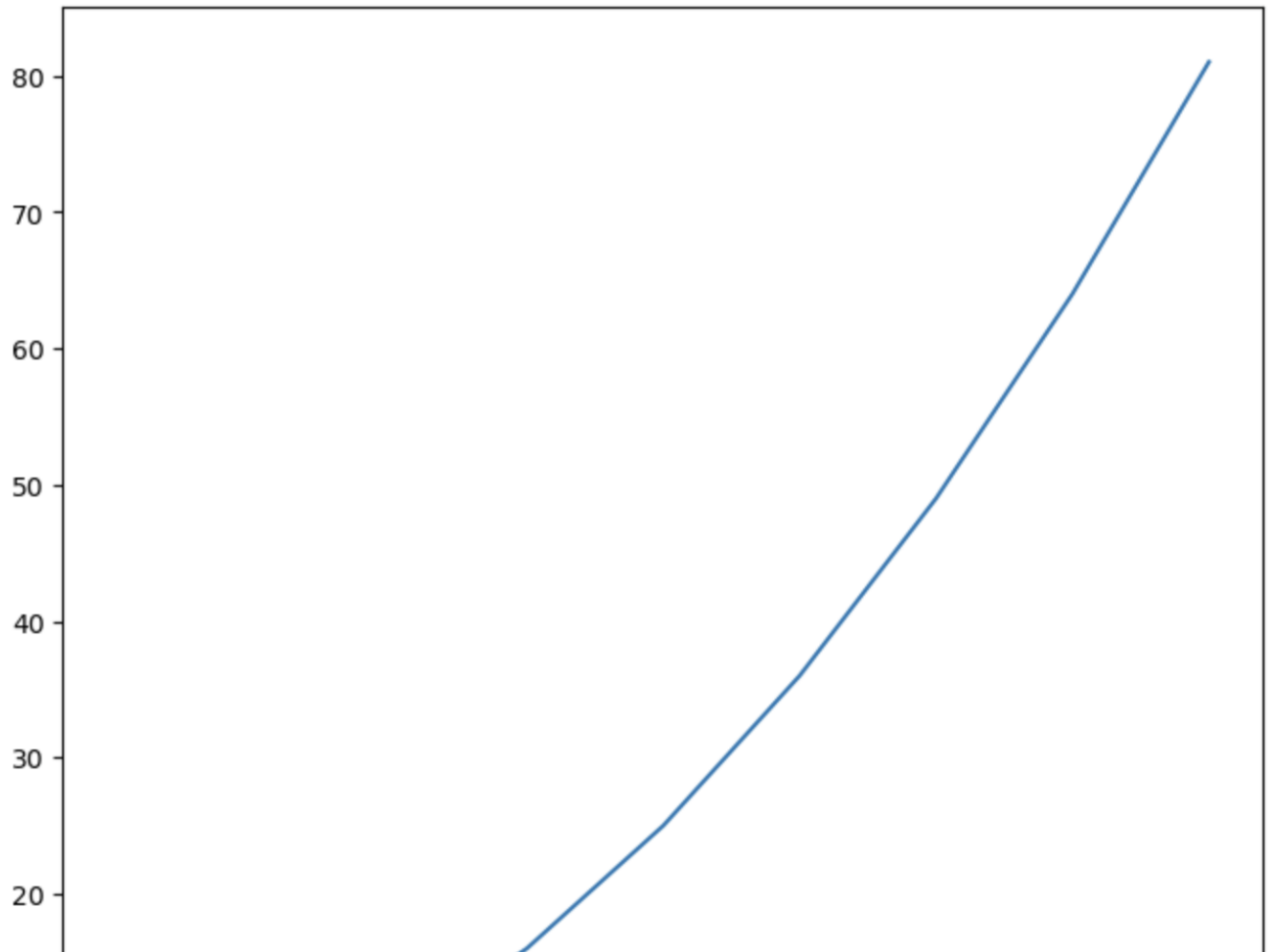


Figure size

In [52]:
```
fig = plt.figure(figsize=(2,2))

ax = fig.add_axes([0.1,0.1,0.8,0.8])
ax.plot(x,y)

#fig.savefig('smallfig.png', dpi=200)
```



In [53]:
```
fig = plt.figure(figsize=(8,8))

ax = fig.add_axes([0.1,0.1,0.8,0.8])
ax.plot(x,y)

#fig.savefig('largefig.png', dpi=200)
```



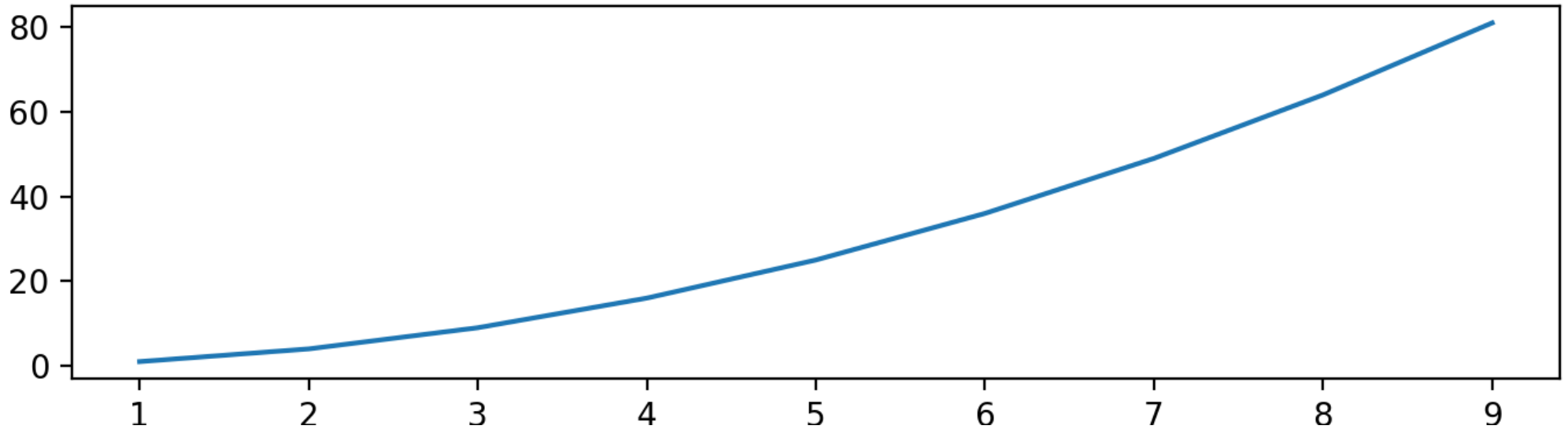
Figsize(8,2)

In[ ]:

```
1 | fig = plt.figure(figsize=(8,2))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 | ax.plot(x,y)
```

Wide figure

Out[]:



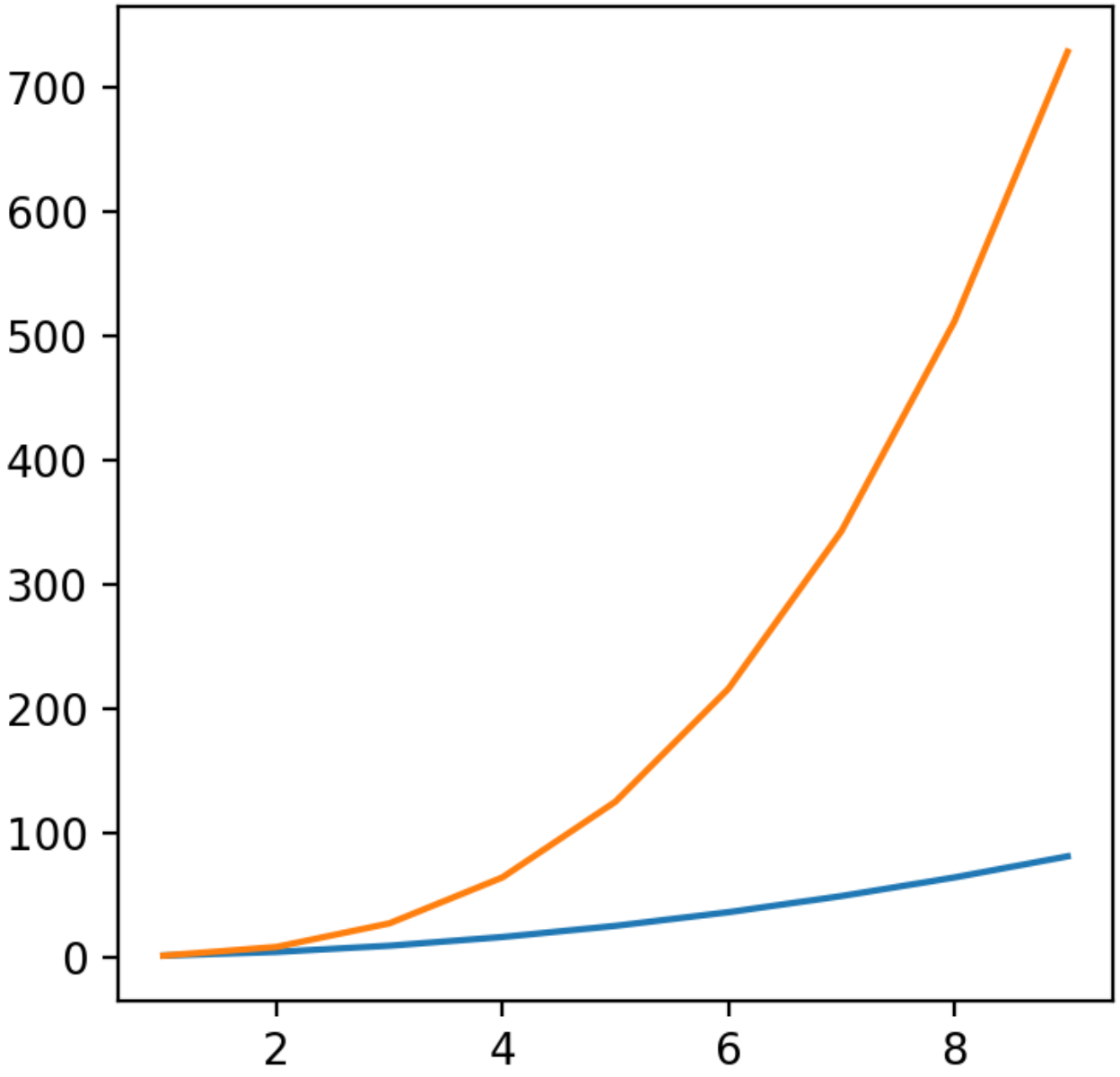
Multiple plots on axes

Multiple plots

In[ ]:
```
fig = plt.figure(figsize=(4,4))
ax = fig.add_axes([0.1,0.1,0.8,0.8])

ax.plot(x,x**2)
ax.plot(x,x**3)
```

Multiple plots

Out[]:



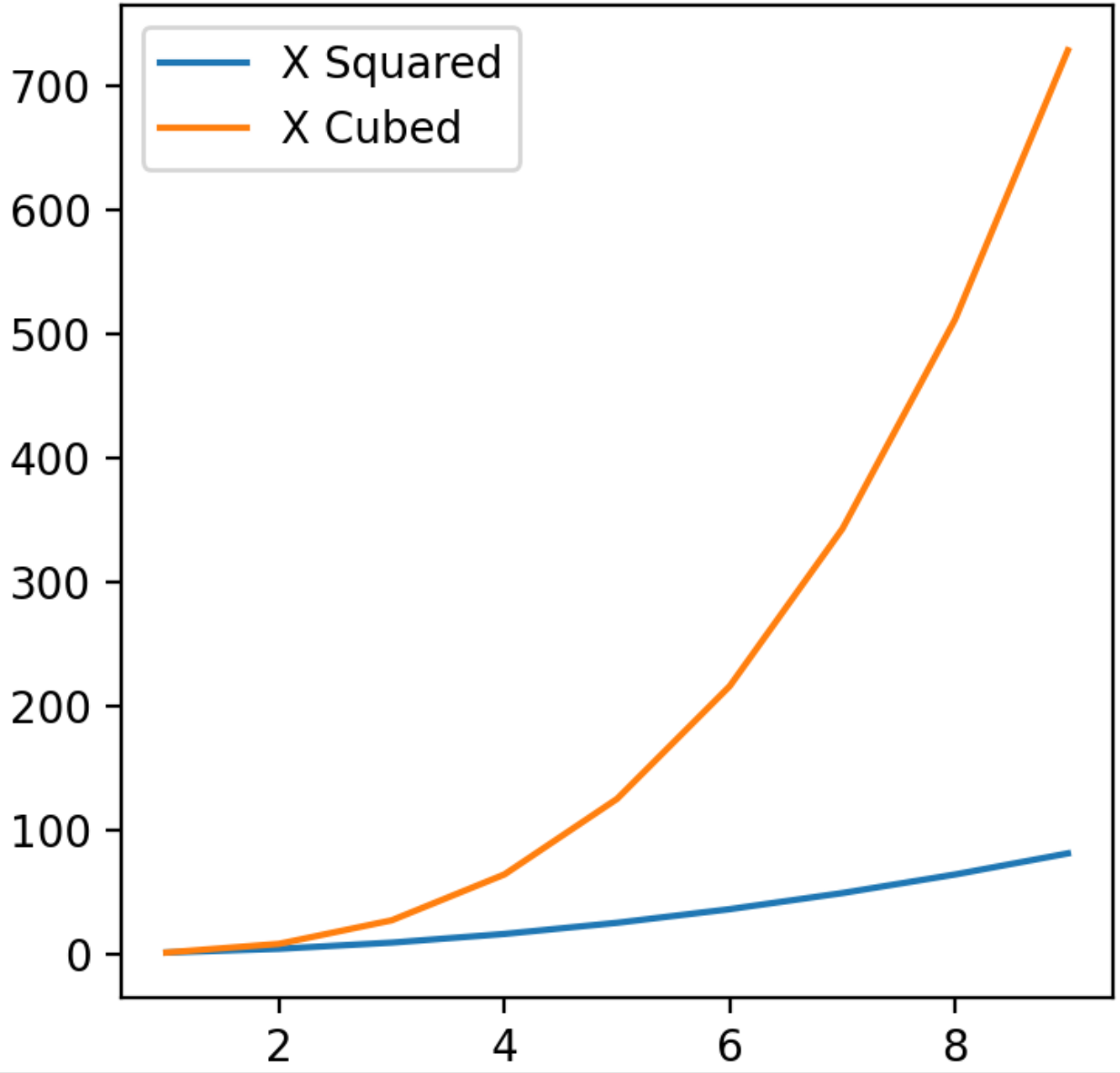
Added legend

In[ ]:

```
fig = plt.figure(figsize=(4,4))
ax = fig.add_axes([0.1,0.1,0.8,0.8])

ax.plot(x,x**2, label = "X Squared")
ax.plot(x,x**3, label = "X Cubed")
ax.legend() # requires labels
```

Multiple plots

Out[]:



Additional formatting

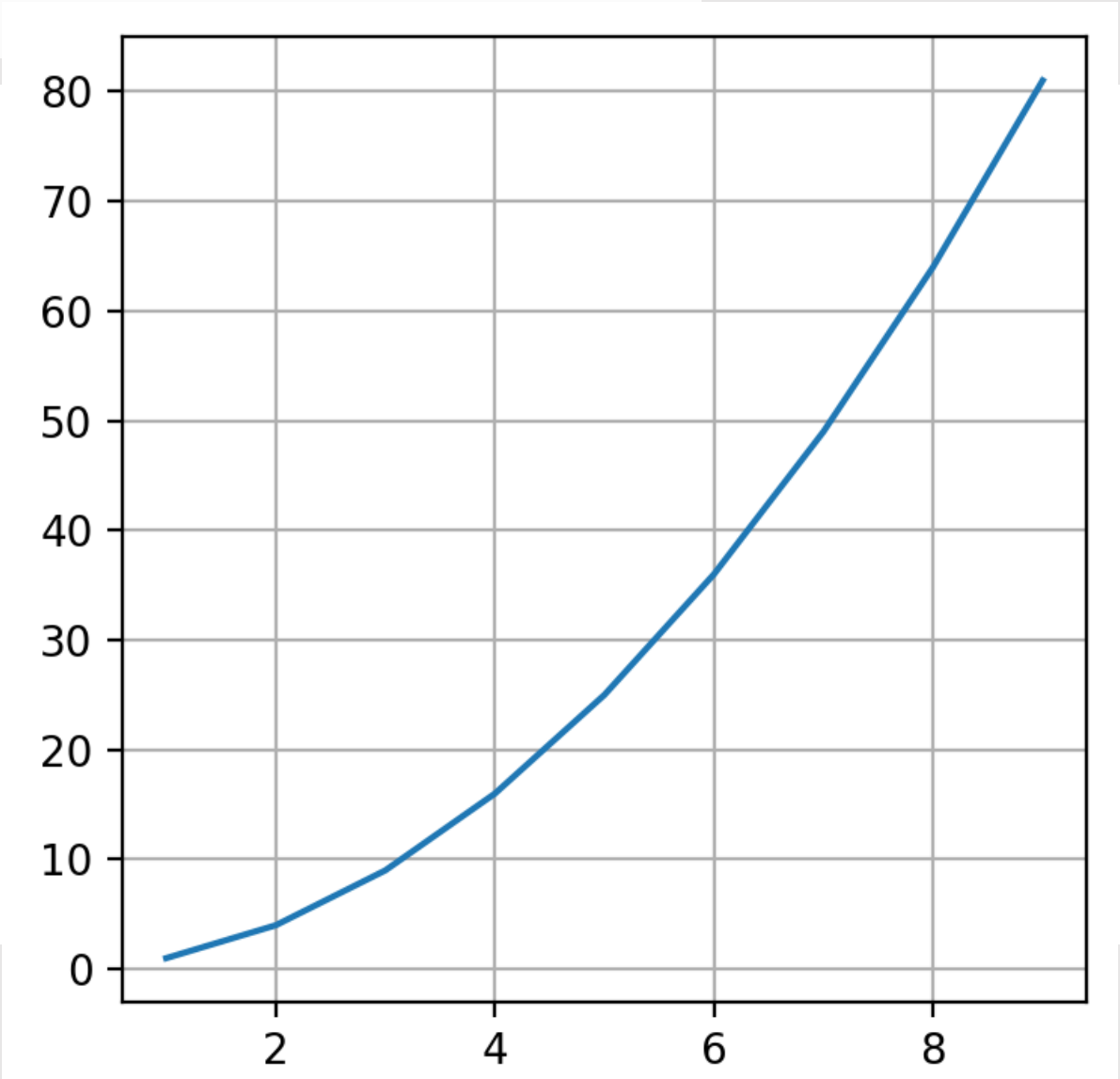
Grid formatting

In[ ]:

```
1 | fig = plt.figure(figsize=(4,4))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 |
4 | ax.plot(x, y)
5 | plt.grid()
```

Grid formatting

Out[]:



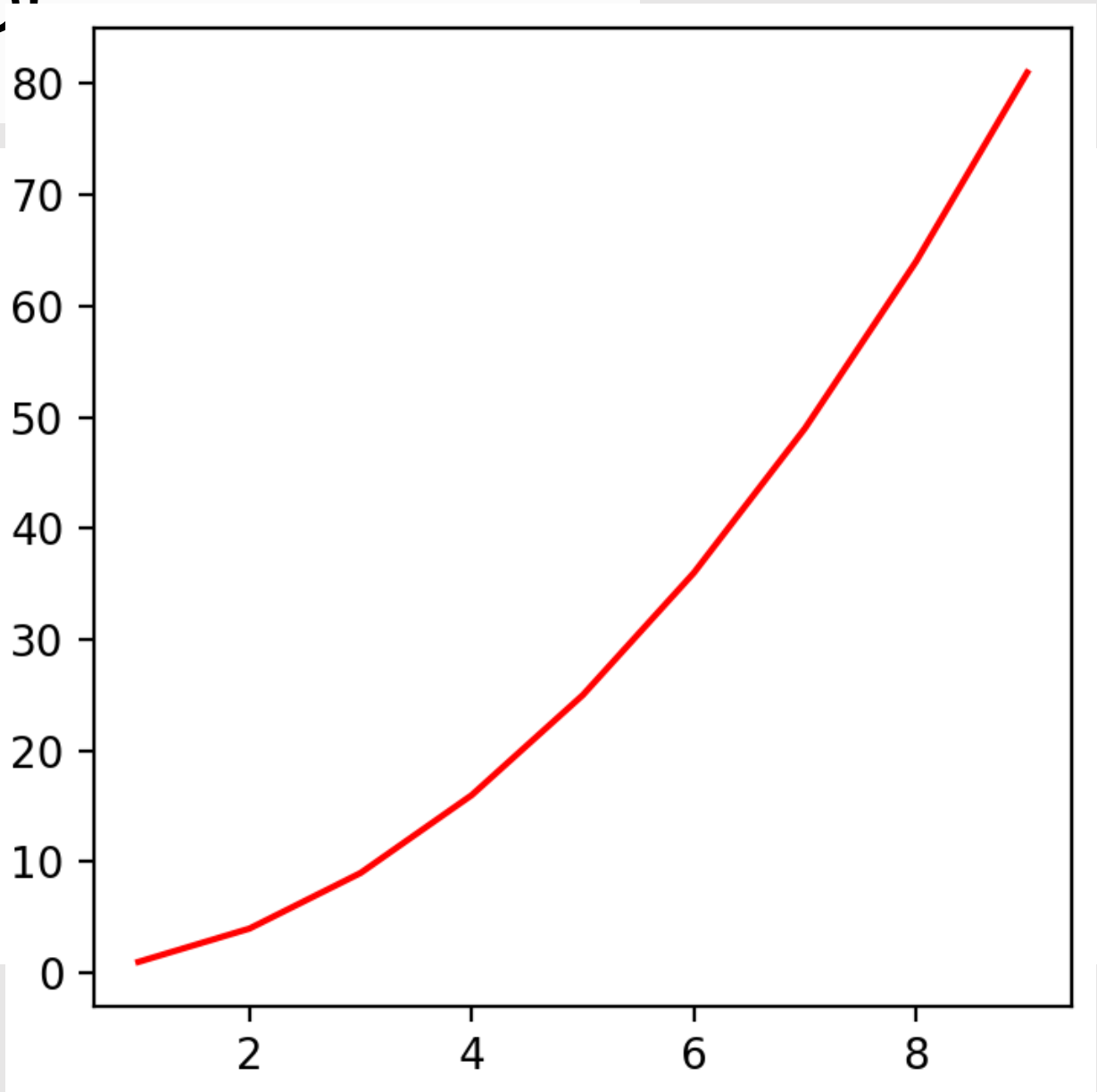
Color with figures

In[ ]:

```
1 | fig = plt.figure(figsize=(4,4))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 |
4 | ax.plot(x, y, color = 'red')
```

Red color

Out[]:



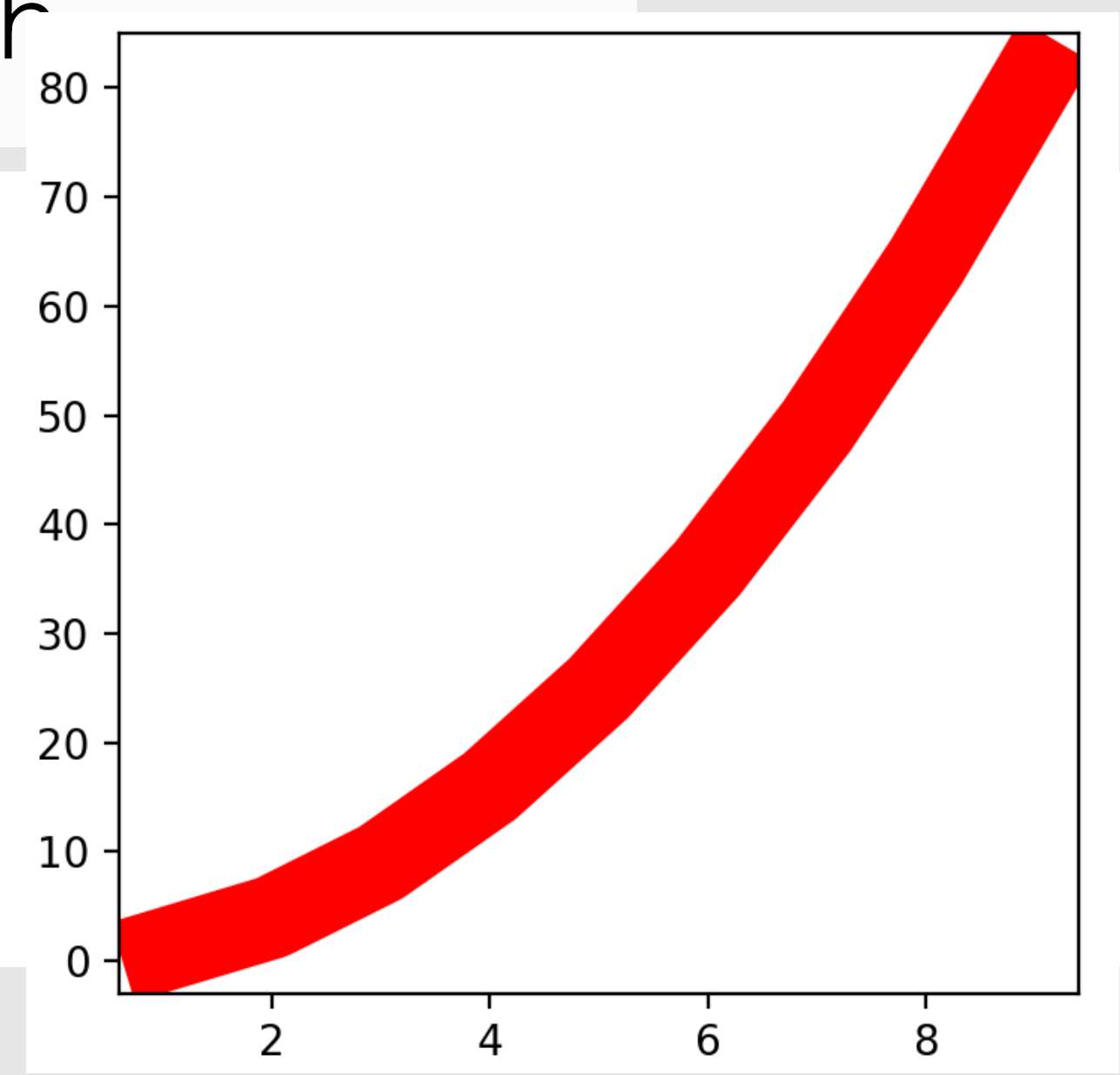
Linewidth

In[ ]:

```
1 | fig = plt.figure(figsize=(4,4))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 |
4 | ax.plot(x, y, color='red', linewidth=20)
```

Linewidth

Out[]:



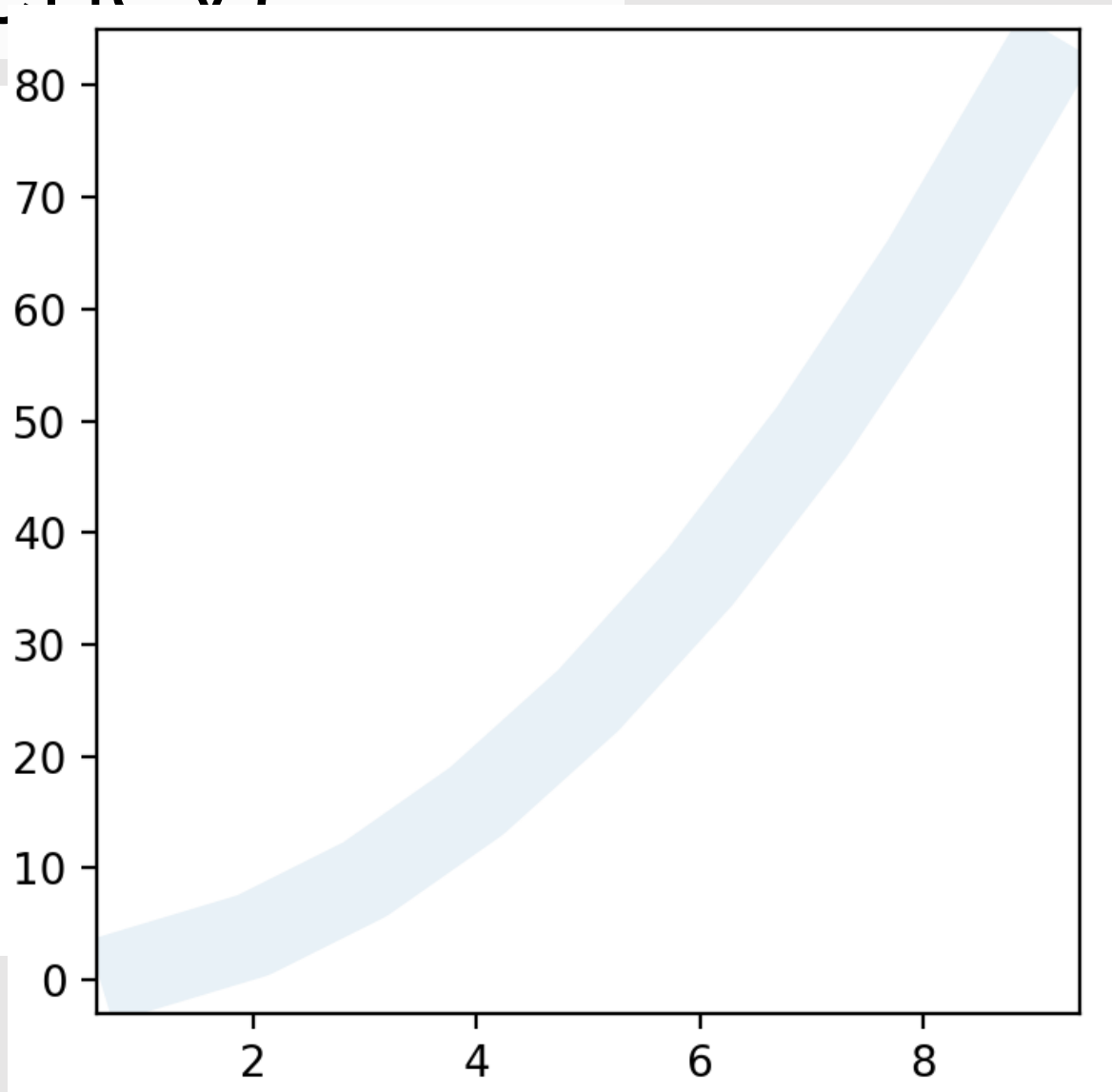
Alpha (transparency)

In[ ]:

```
1 | fig = plt.figure(figsize=(4,4))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 |
4 | ax.plot(x, y, linewidth=20, alpha = 0.1)
```

Alpha (transparency)

Out[]:

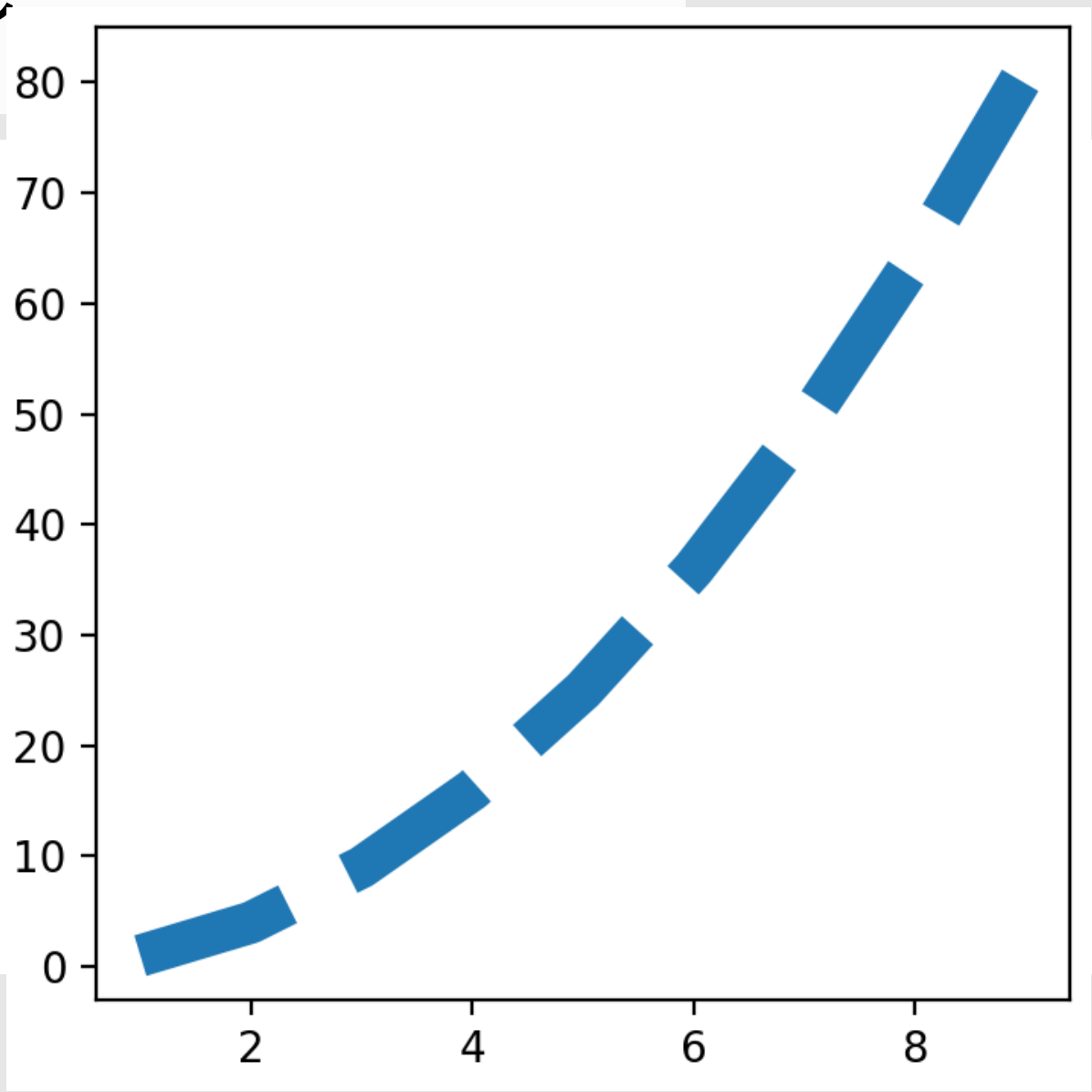


Linestyle

In[ ]:
```
1 | fig = plt.figure(figsize=(4,4))
2 | ax = fig.add_axes([0.1,0.1,0.8,0.8])
3 |
4 | ax.plot(x, y, linewidth=20, linestyle="--")
```

Linestyle

Out[]:



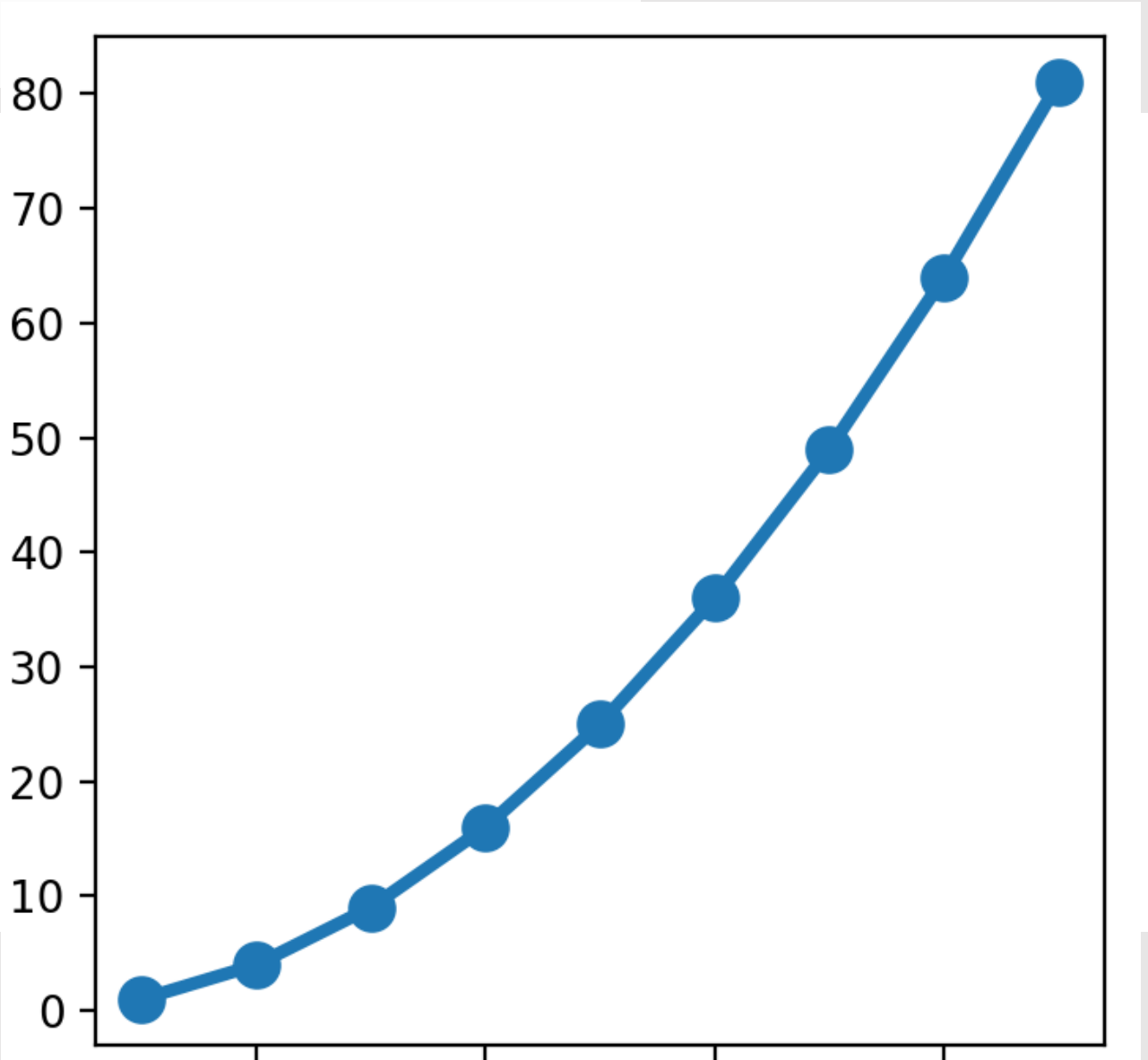
Marker

In[ ]:

```
fig = plt.figure(figsize=(4,4))
ax = fig.add_axes([0.1,0.1,0.8,0.8])

ax.plot(x, y, linewidth=3,
        marker = "o", markersize = "10")
```

marker = 'o'

Out[]:



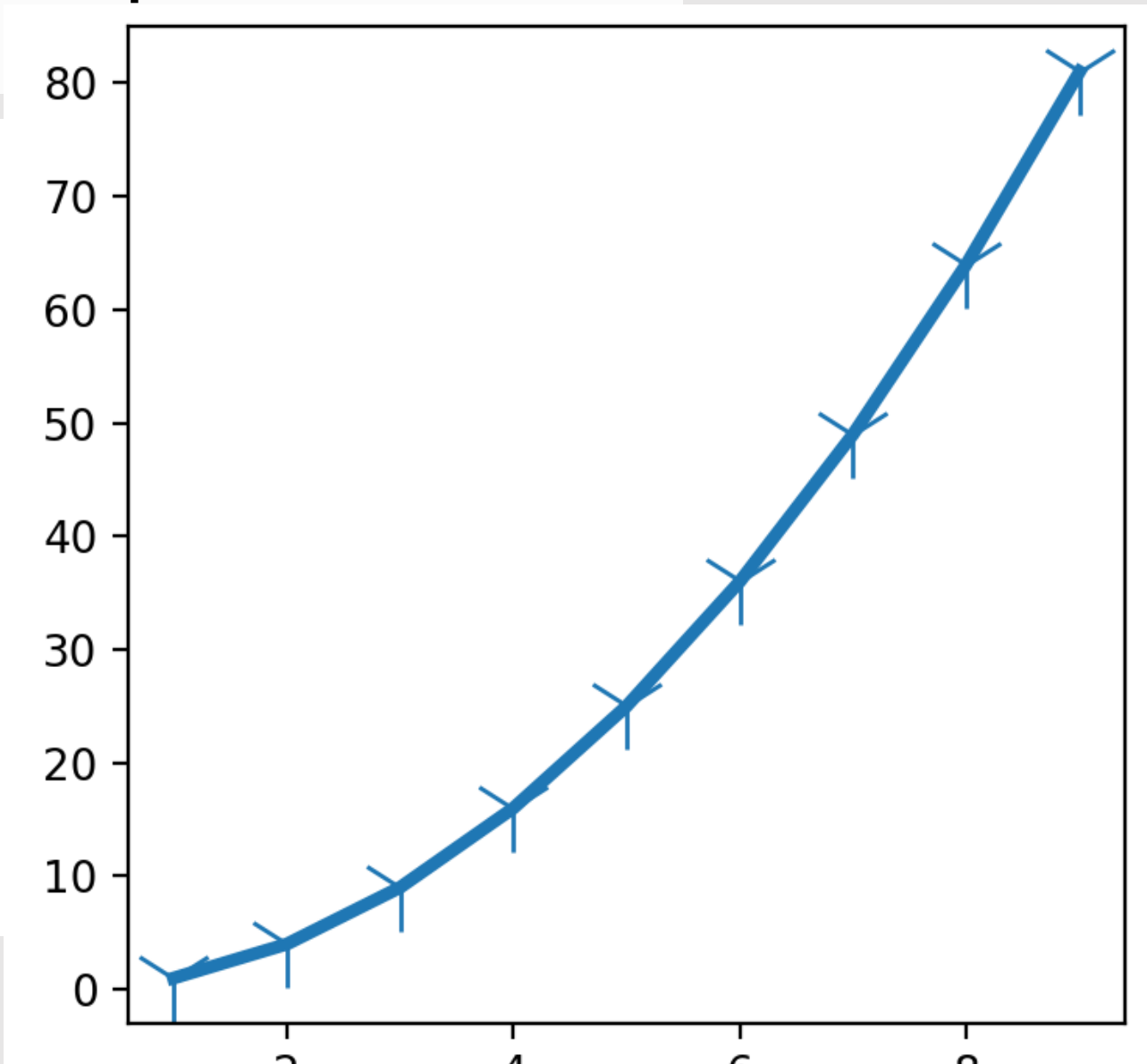
Marker

In[ ]:

```
fig = plt.figure(figsize=(4,4))
ax = fig.add_axes([0.1,0.1,0.8,0.8])

ax.plot(x, y, linewidth=3,
        marker = "1", markersize = "20")
```

marker = '1'

Out[]:



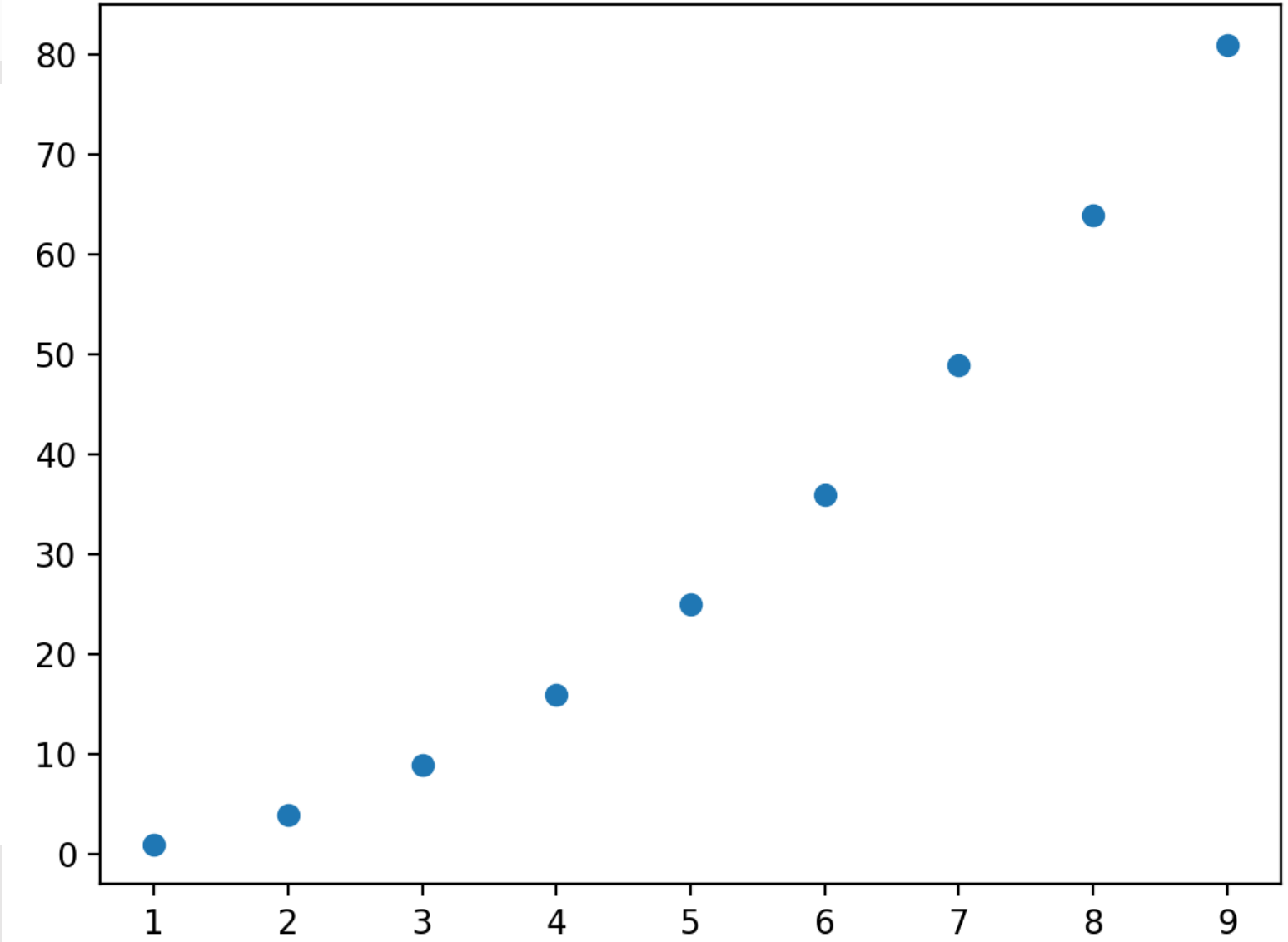
Scatter plots

Basic Scatter plot

In[ ]:
```
1 | plt.scatter(x, y)
2 |
```

Scatter

Out[]:

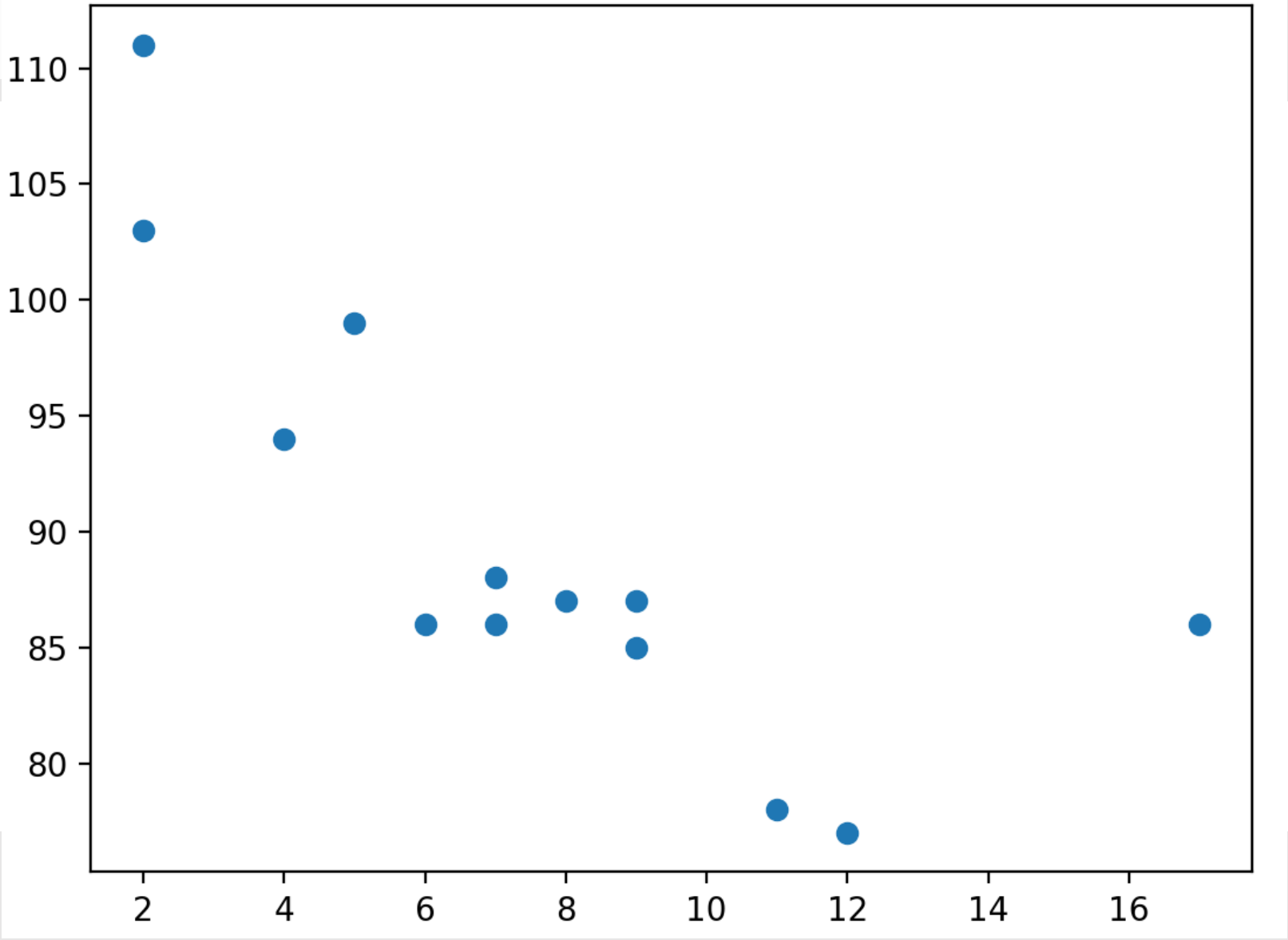


Plot a dataset

In[ ]:
```
1 | x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
2 | y = [99,86,87,88,111,86,103,87,94,78,77]
3 | plt.scatter(x, y)
```

Scatter

Out[]:



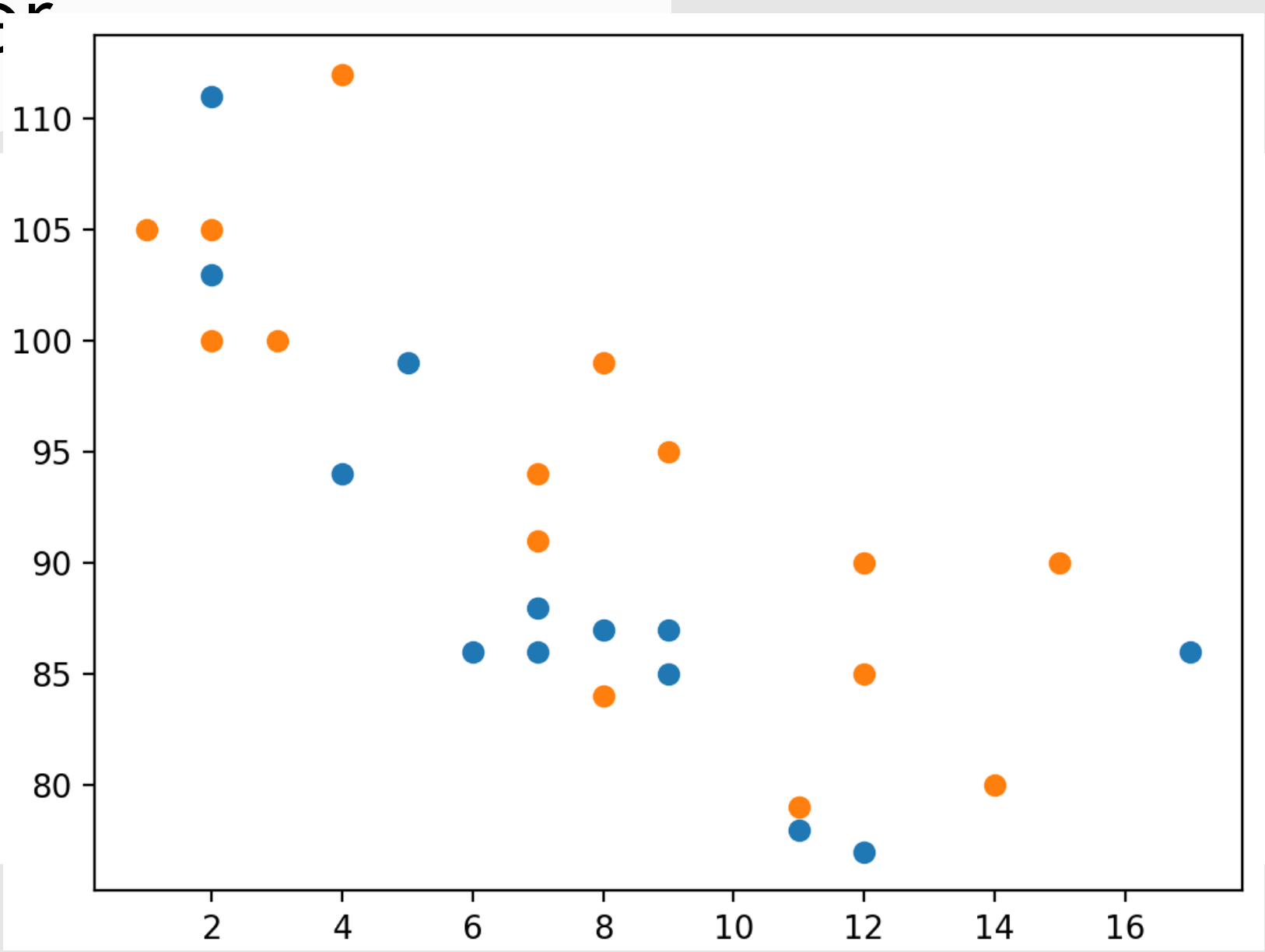
Plot two datasets

In[ ]:

```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77]
plt.scatter(x, y)

x = [2,2,8,1,15,8,12,9,7,3,11,4,7]
y = [100,105,84,105,90,99,90,95,94,100]
plt.scatter(x, y)
```

Scatter

Out[]:



Line of Best Fit

In[ ]:

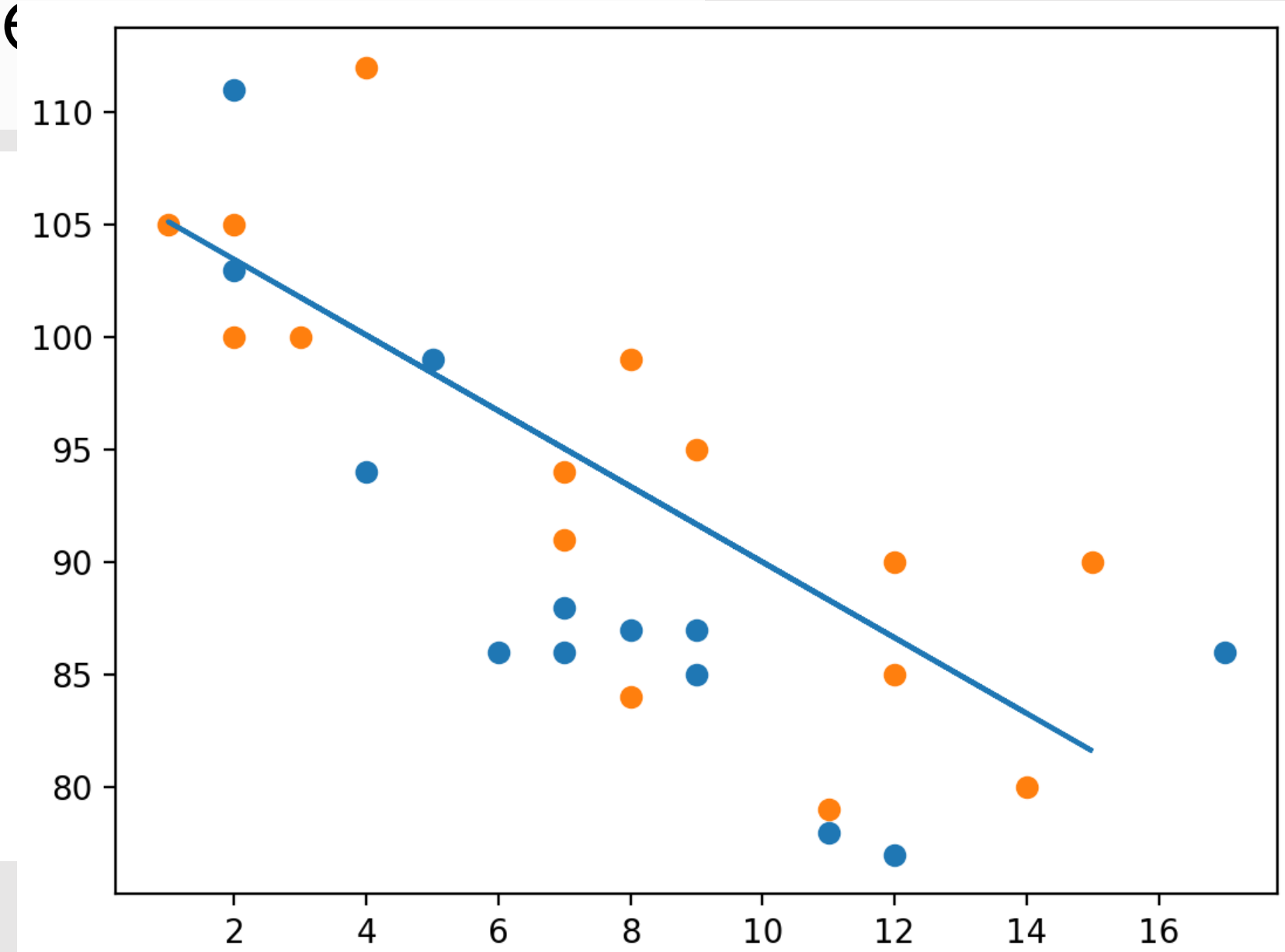
```
1 | a, b = np.polyfit(x, y, 1)
2 | plt.plot(x, a*x+b)
3 |
```

Line of Best Fit

In[ ]:
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77]
plt.scatter(x, y)
x = [2,2,8,1,15,8,12,9,7,3,11,4,7]
y = [100,105,84,105,90,99,90,95,94,100]
plt.scatter(x, y)
a, b = np.polyfit(x, y, 1)
plt.plot(x, a*x+b)
```

Scatte

Out[]:



Pie Chart

Basic pie

In[ ]:
```
1 | data = [35, 25, 25, 15]
2 |
3 | plt.pie(data)
```

Pie chart

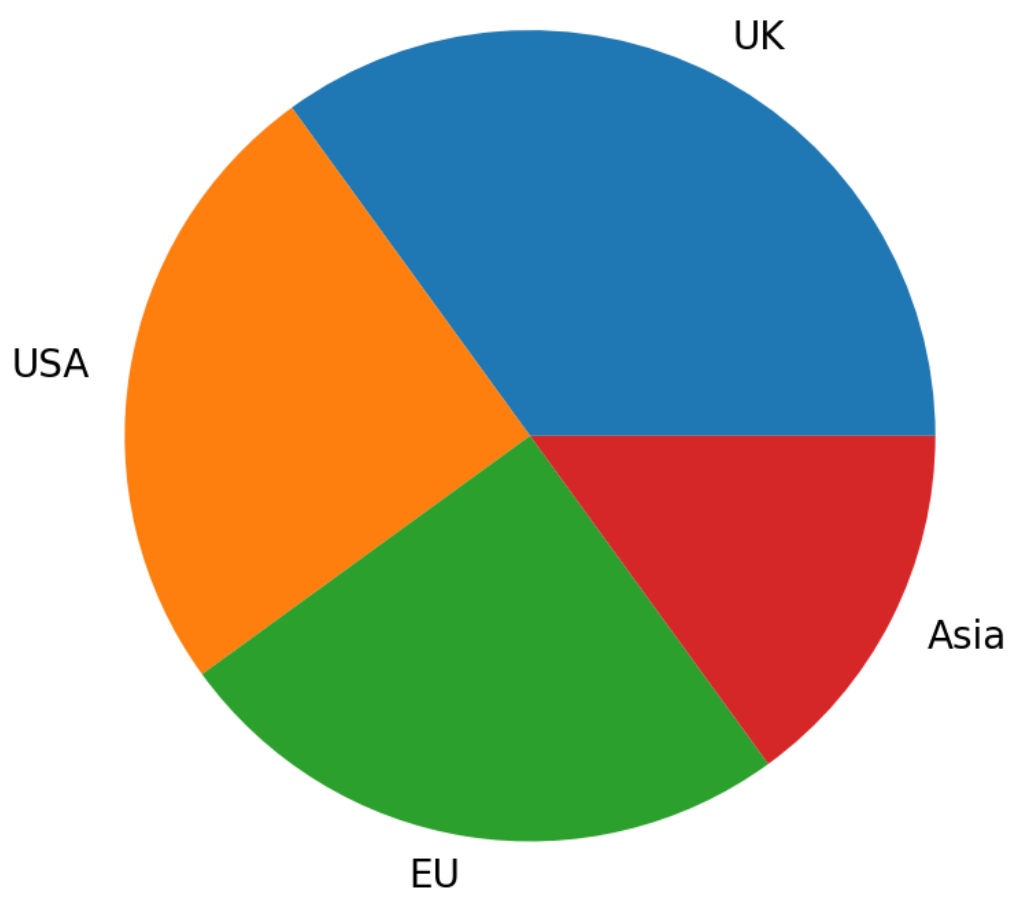
Out[]:

Pie with labels

In[ ]:
```
1 | data = [35, 25, 25, 15]
2 | l = ["UK", "USA", "EU", "Asia"]
3 | plt.pie(data, labels = l)
```

Pie chart labels

Out[]:

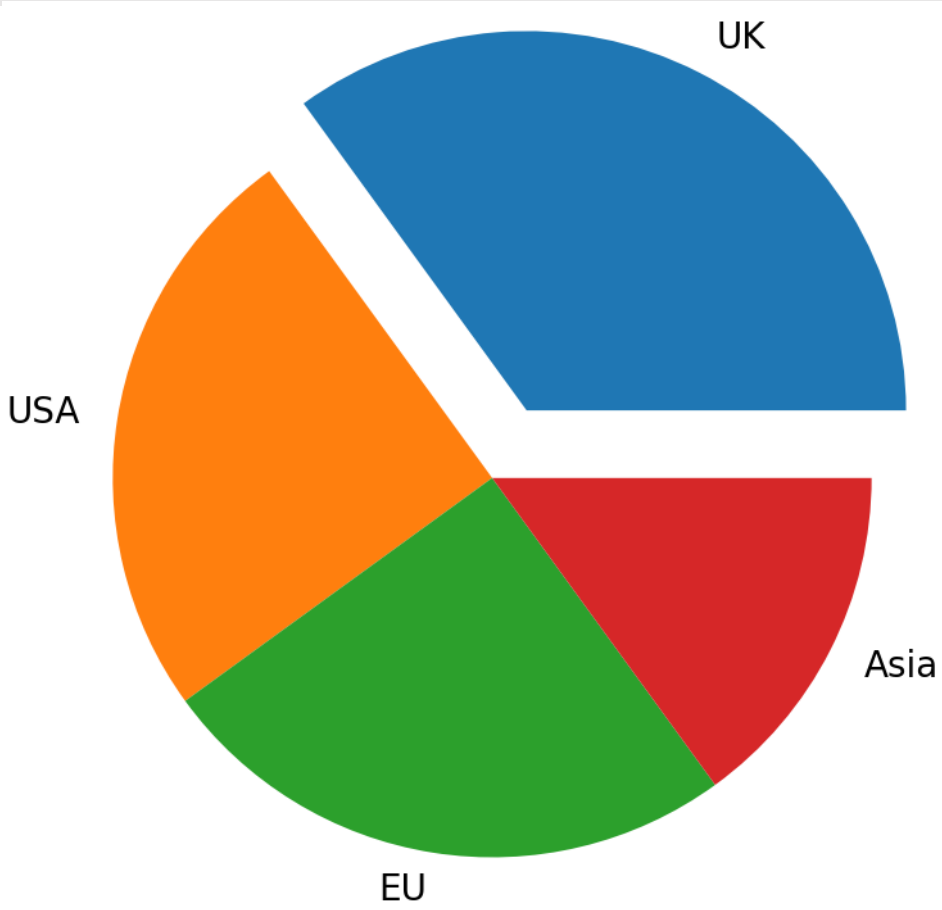


Wedge potraction

In[ ]:
```
data = [35, 25, 25, 15]
l = ["UK", "USA", "EU", "Asia"]
ex = [0.2, 0, 0, 0]
plt.pie(data, labels=l, explode=ex)
```

Pie chart explode

Out[]:



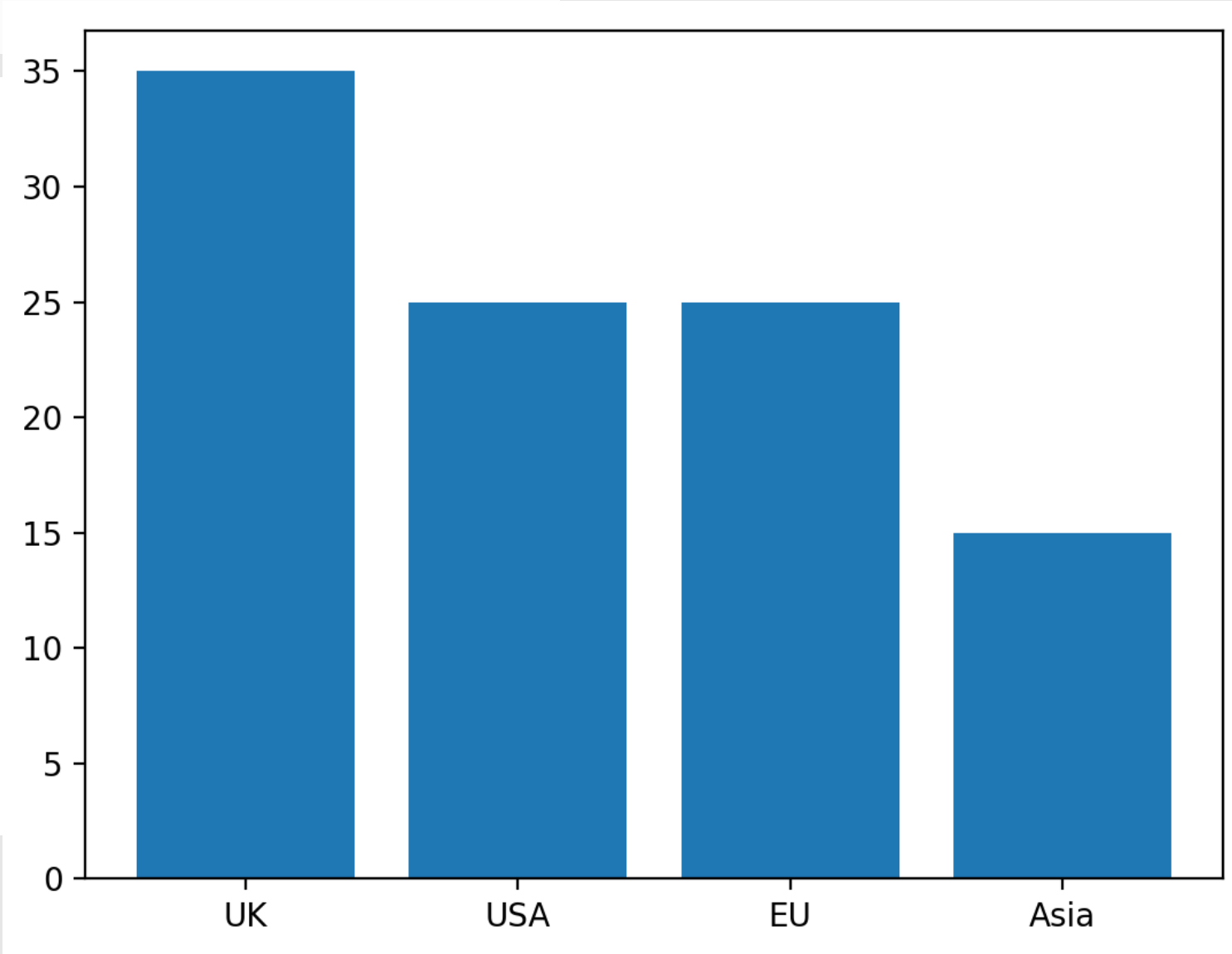
Bar Graph

Simple Bar

In[ ]:
```
1 | data = [35, 25, 25, 15]
2 | labels = ["UK", "USA", "EU", "Asia"]
3 |
4 | plt.bar(labels, data)
```

Bar graph

Out[]:



Histograms

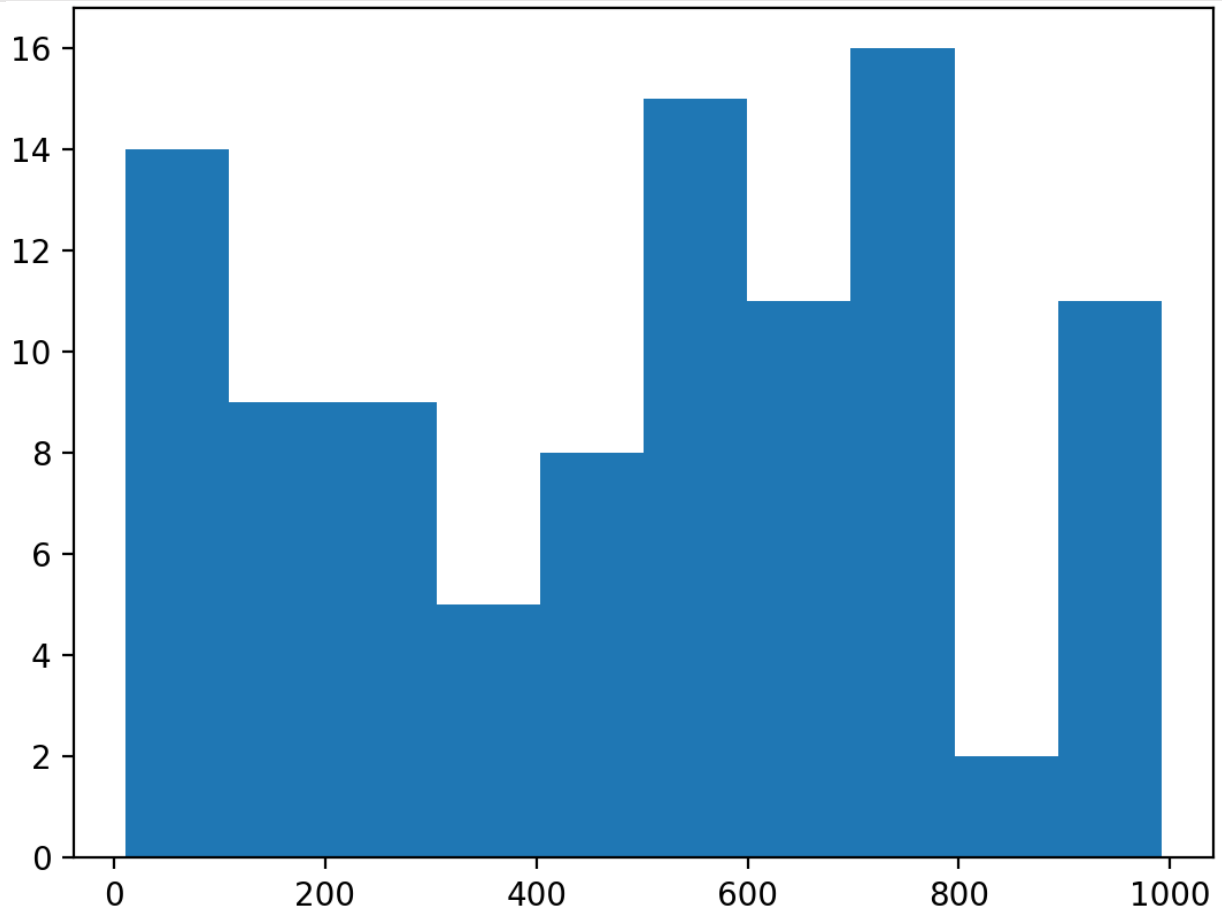
Histogram

In[ ]:

```
1 | from random import sample
2 | data = sample(range(1, 1000), 100)
3 | plt.hist(data)
```

Basic Histogram

Out[]:



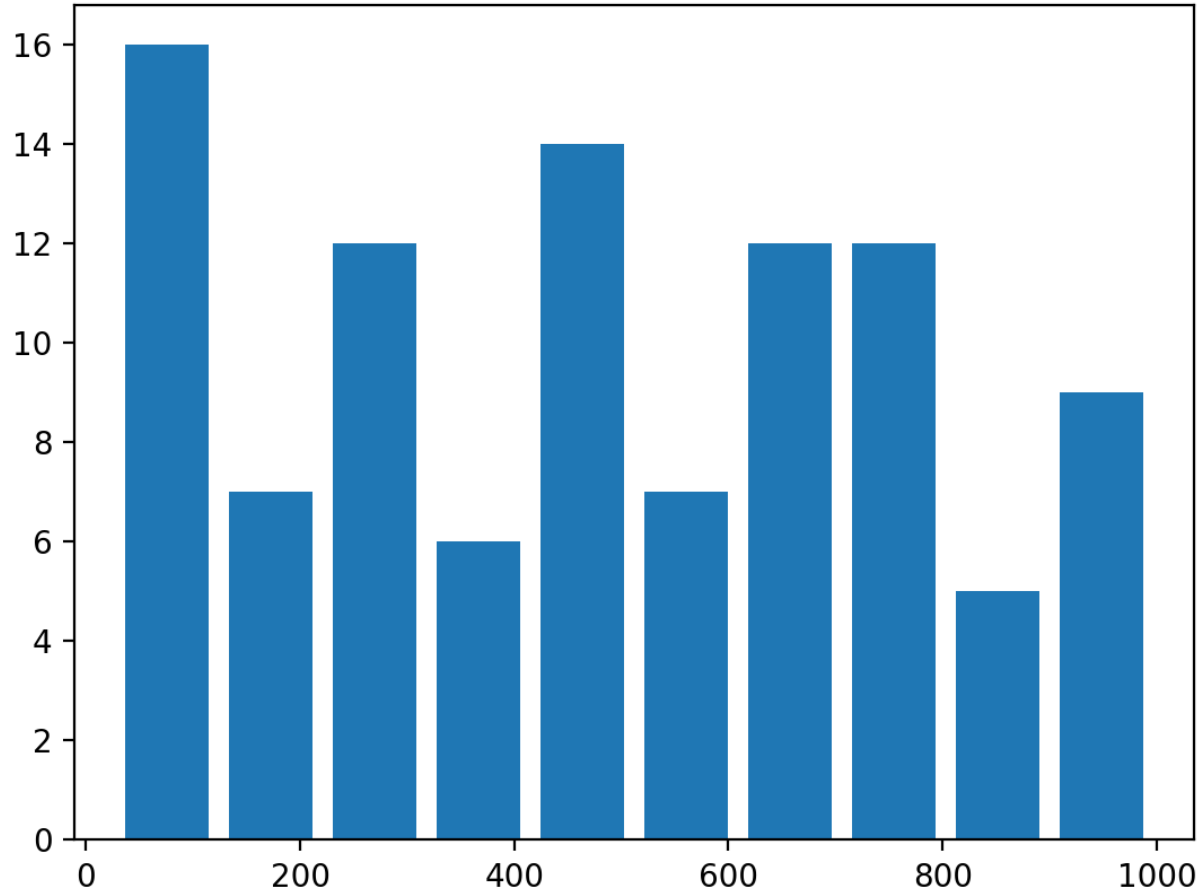
Histogram

In[ ]:

```
1 | from random import sample
2 | data = sample(range(1, 1000), 100)
3 | plt.hist(data, rwidth = 0.8)
```

With spaces

```
Out[]:
```

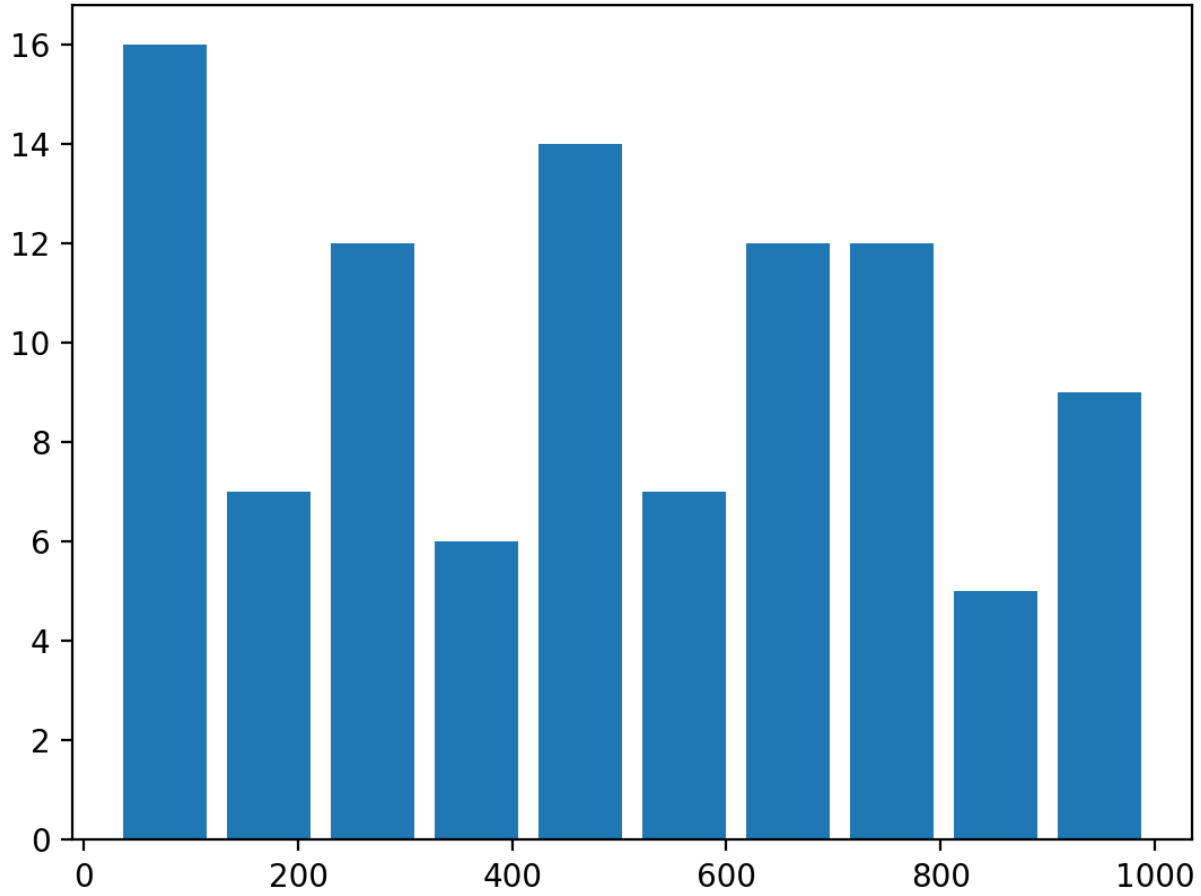


Bins - intervals

In[ ]:
```
1 | from random import sample
2 | data = sample(range(1, 1000), 100)
3 | plt.hist(data, bins = 10, rwidth = 0.8)
```

10 bins

Out[]:

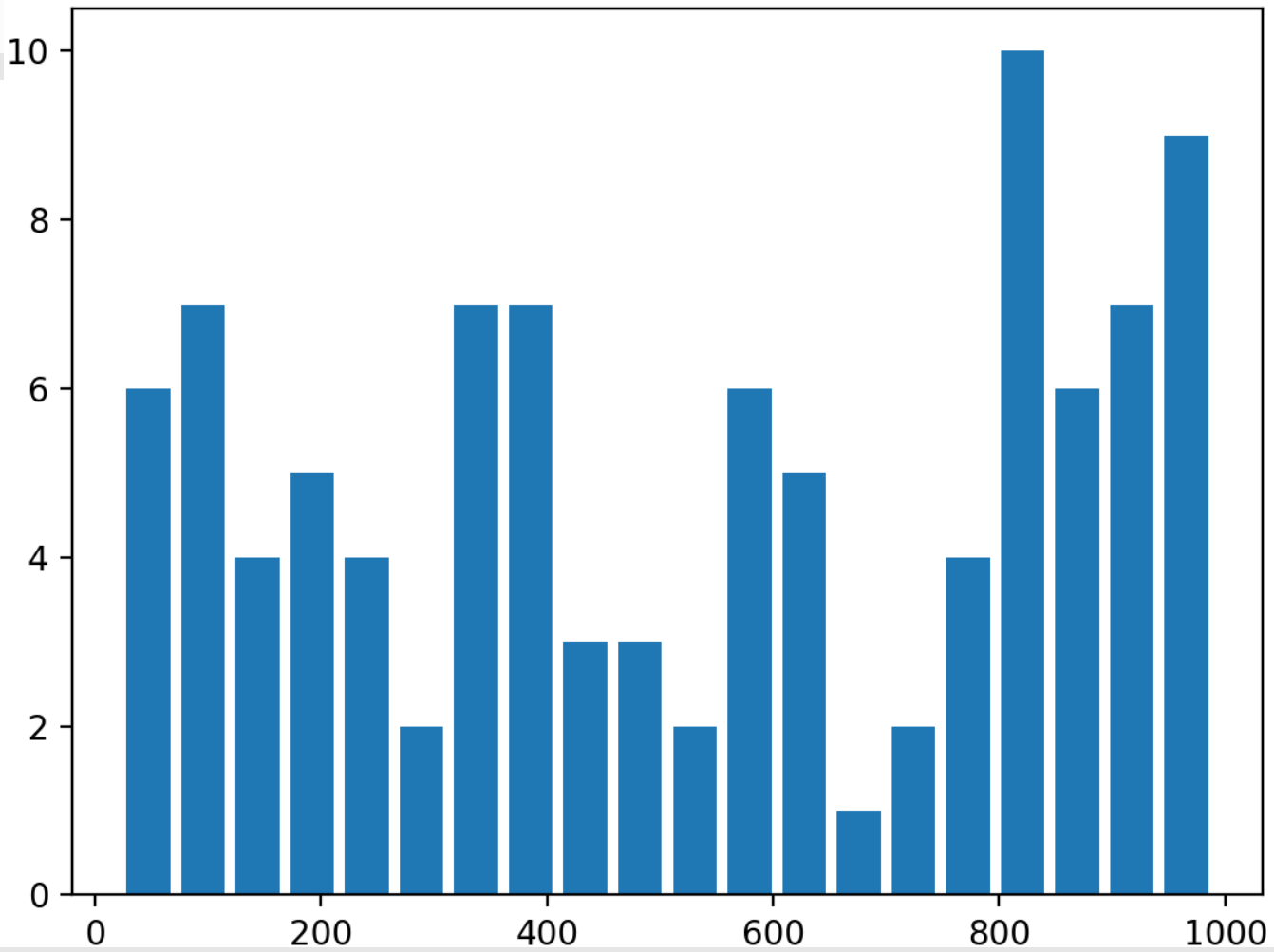


20 bins

In[ ]:
```
1 | from random import sample
2 | data = sample(range(1, 1000), 100)
3 | plt.hist(data, bins = 20, rwidth = 0.8)
```

20 bins

Out[]:



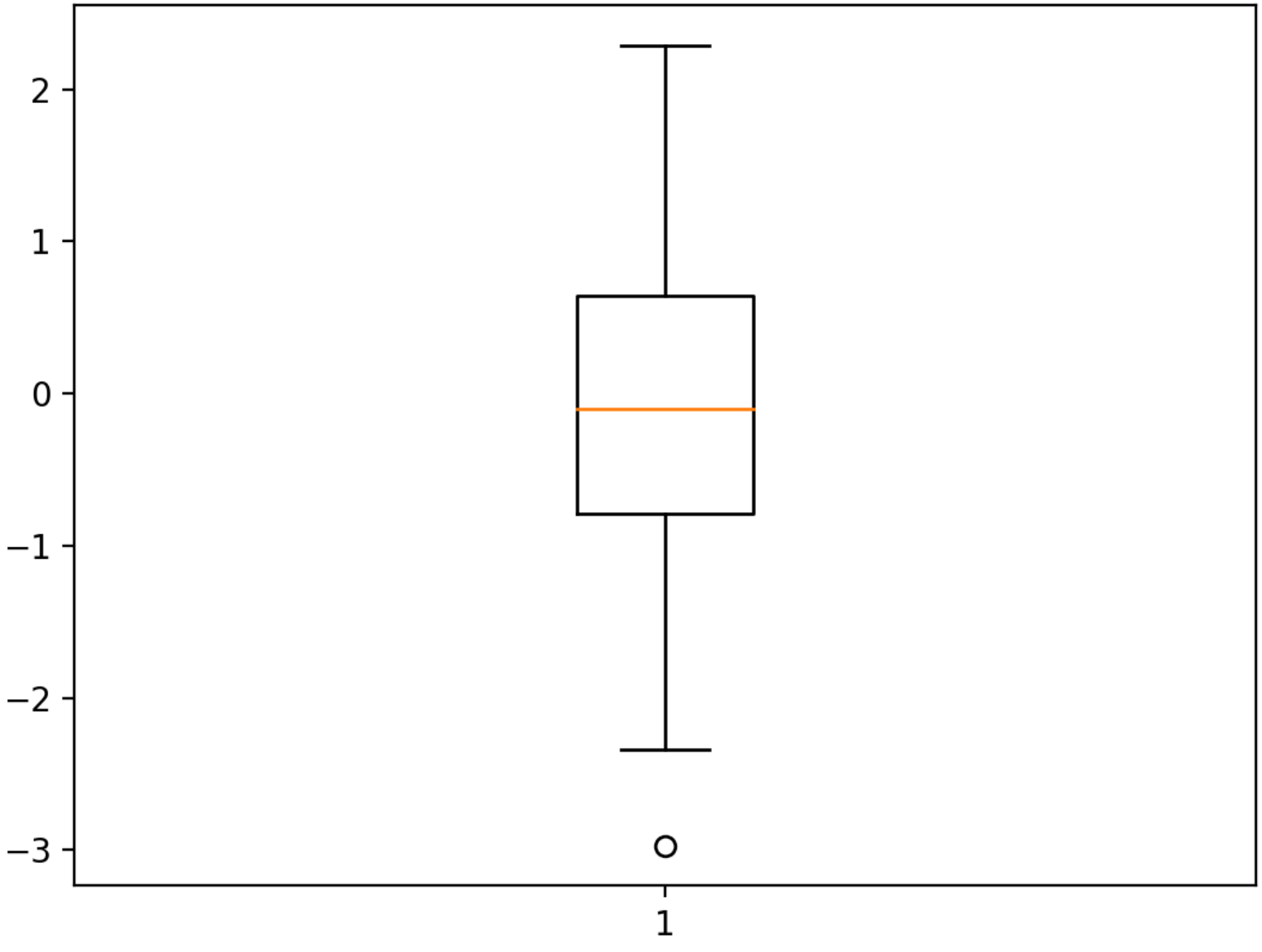
Box plots

Box Plot

In[ ]:
```
1 | std = 1
2 | data = [np.random.normal(0, std, 100)]
3 | plt.boxplot(data, vert = True)
```

Box Plot

Out[]:



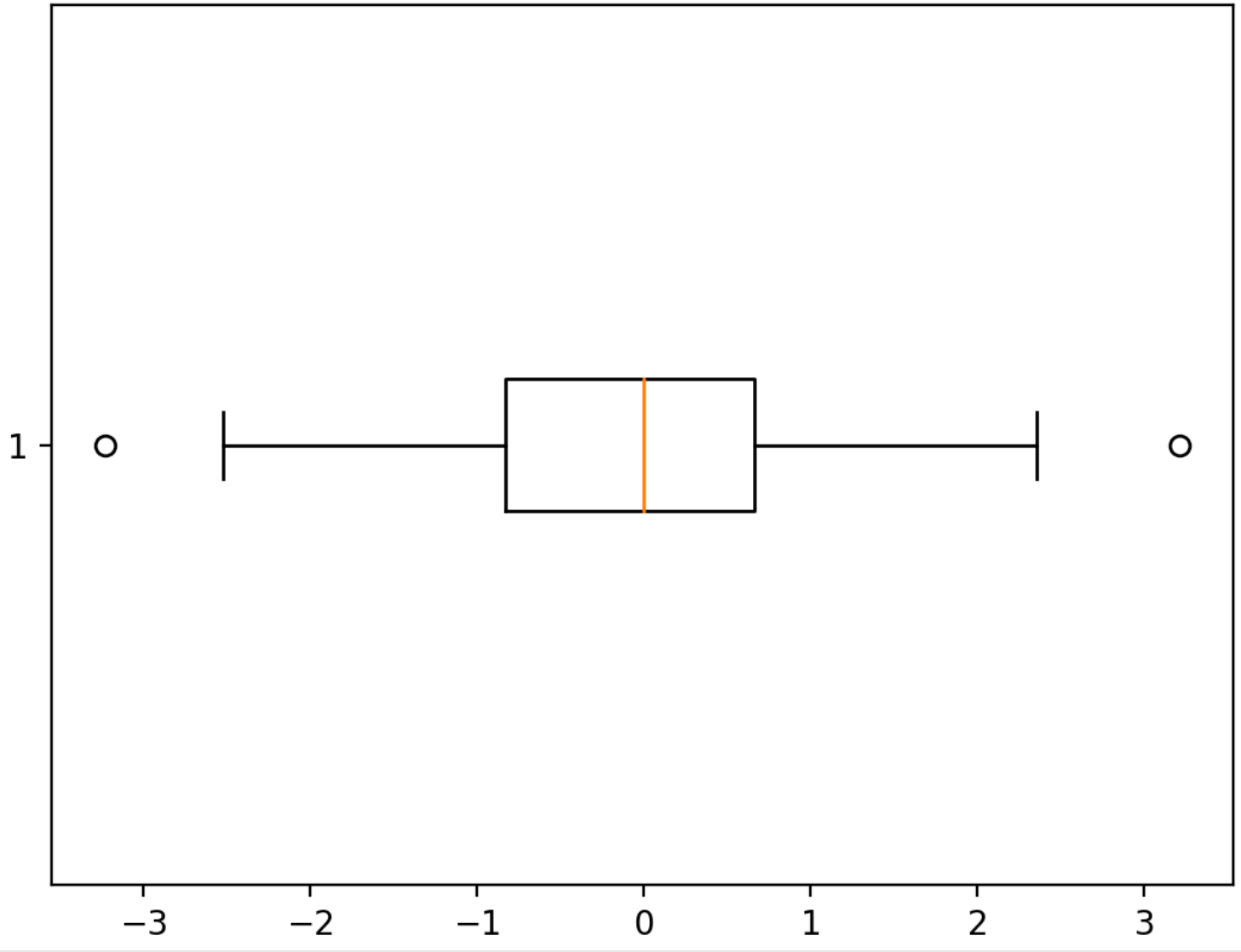
Box Plot

In[ ]:

```
1 | std = 1
2 | data = [np.random.normal(0, std, 100)]
3 | plt.boxplot(data, vert = False)
```

Box Plot

Out[]:



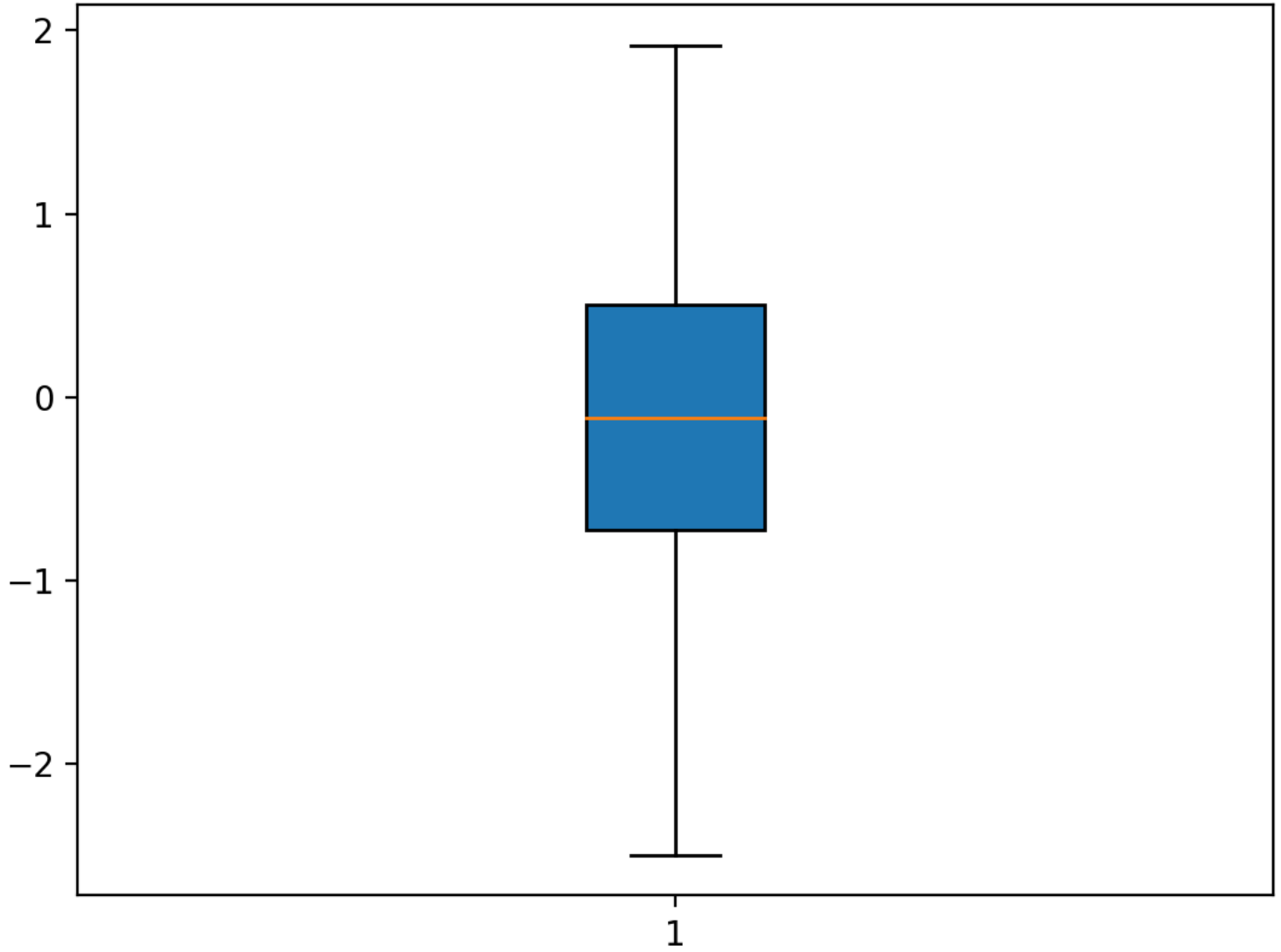
Box Plot

In[ ]:

```
1 | std = 1
2 | data = [np.random.normal(0, std, 100)]
3 | plt.boxplot(data,vert=True,patch_artist=True)
```

Box Plot

Out[]:



Box Plot

In[ ]:

```
std = 1
data = [np.random.normal(0, std, 100)
        for std in range(1,4)]
plt.boxplot(data,vert=True,patch_artist=True)
```

Box Plot

Out[]: