Name: - Sandeep Sharma

Date : - 9/7/2020

Matplotlib allows to create reproducible figures programmatically.

Official Matplotlib web page: http://matplotlib.org/ (http://matplotlib.org/)

Installation

You'll need to install matplotlib first with either:

```
pip install matplotlib
```

or conda install matplotlib

Importing

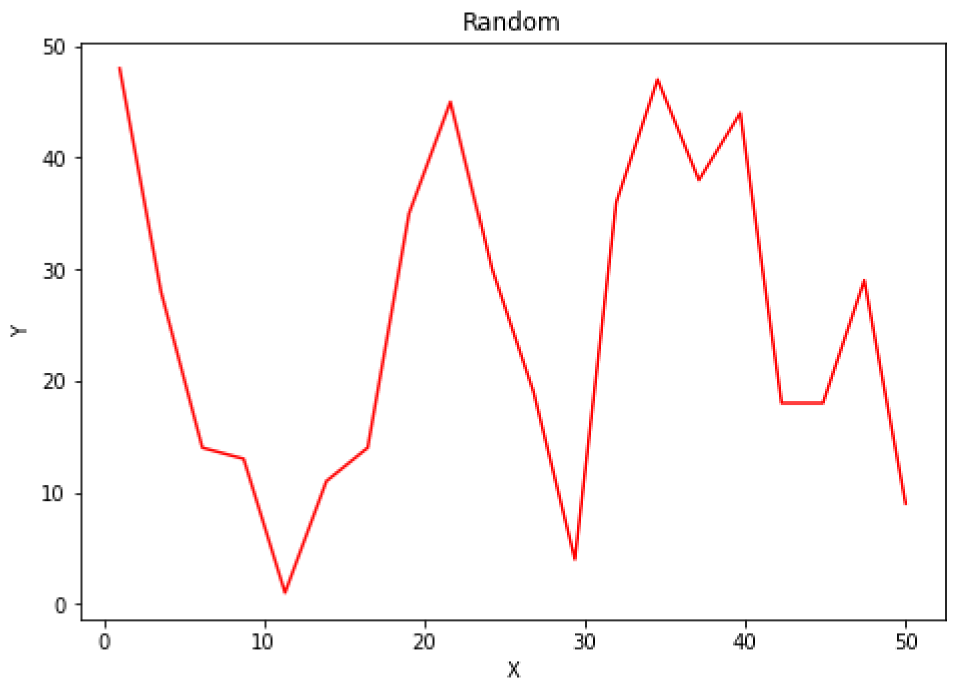
Object Oriented plots

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
```

```
In [2]: x = np.linspace(1,50,20)
        y = np.random.randint(1, 50, 20)
```

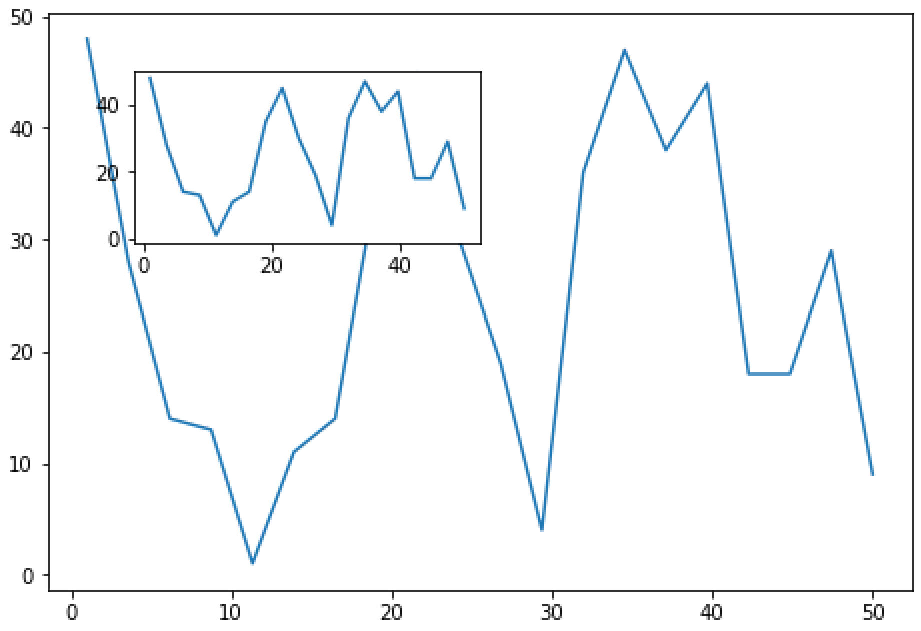
In [3]:
```
fig = plt.figure()
axes = fig.add_axes([0.5, 0.5, 1 ,1])
axes.plot (x,y,'r')
axes.set_xlabel ('X')
axes.set_ylabel ('Y')
axes.set_title ('Random')
```

Out[3]:
```
Text(0.5, 1.0, 'Random')
```



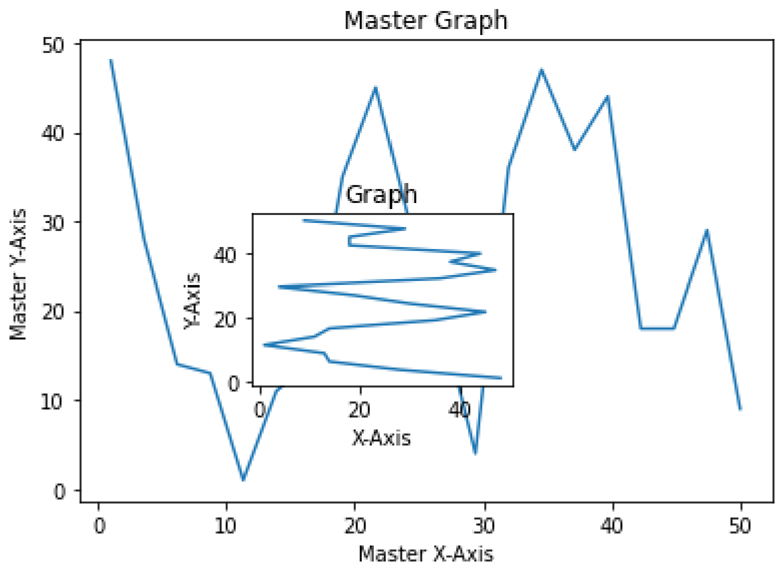
```
In [4]: fig1 = plt.figure()
        ax1 = fig1.add_axes([0, 0, 1 ,1])
        ax2 = fig1.add_axes([0.1, 0.6, 0.4, 0.3])
        ax1.plot(x, y)
        ax2.plot(x, y)
```

```
Out[4]: [<matplotlib.lines.Line2D at 0x27ec8736d00>]
```



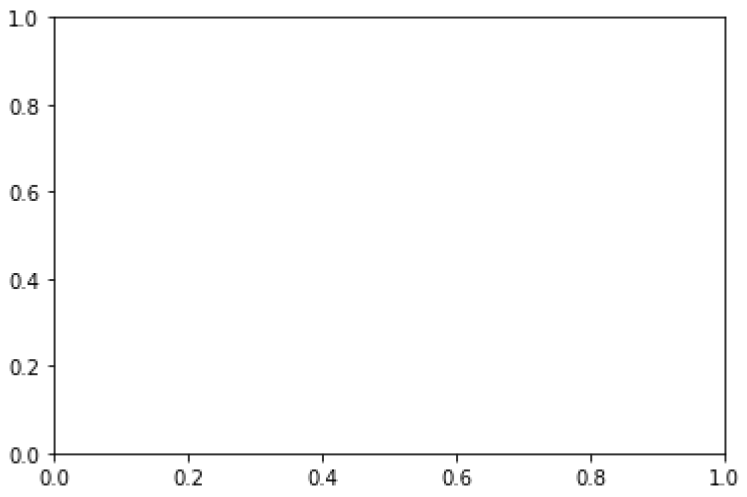
```
In [5]: fig1 = plt.figure()
        x1 = fig1.add_axes([0.1, 0.1, 0.8, 0.8])
        x1.plot(x, y)
        y1 = fig1.add_axes([0.3, 0.3, 0.3, 0.3])
        y1.plot(y, x)

        x1.set_xlabel('Master X-Axis')
        x1.set_ylabel('Master Y-Axis')
        x1.set_title('Master Graph')

        y1.set_xlabel('X-Axis')
        y1.set_ylabel('Y-Axis')
        y1.set_title('Graph')
```

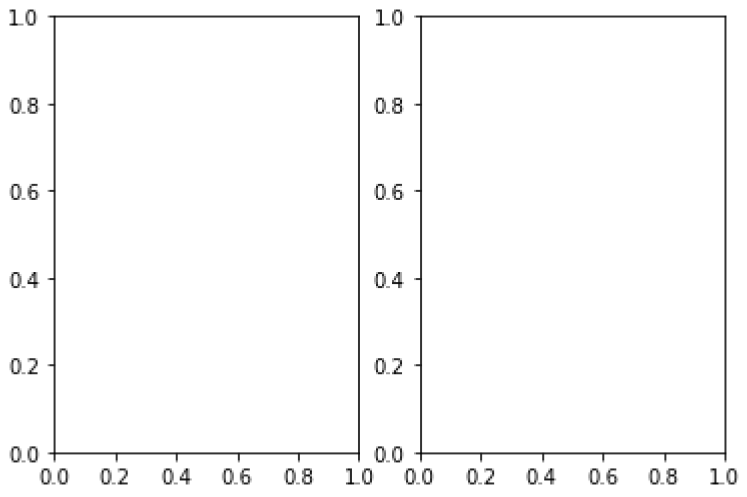
```
Out[5]: Text(0.5, 1.0, 'Graph')
```

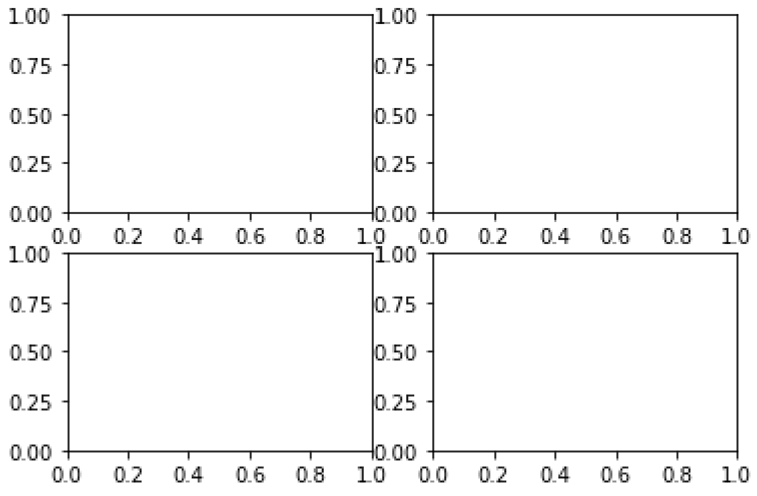


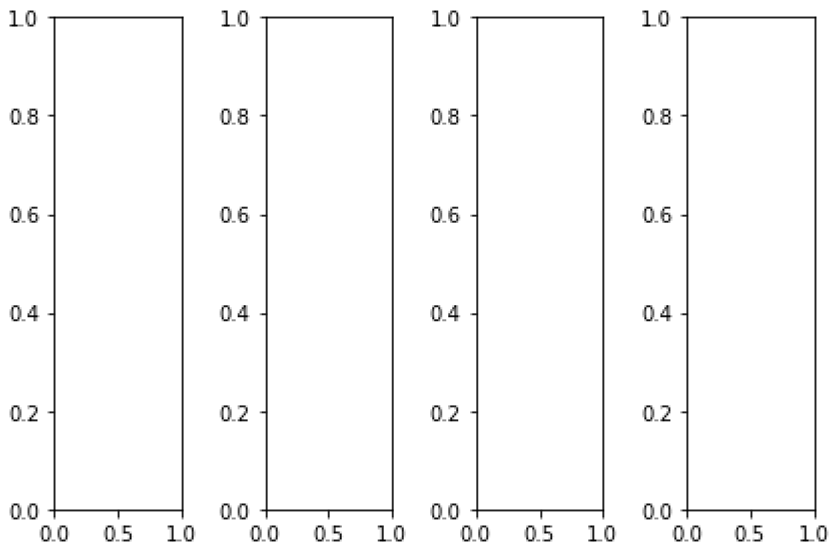
A common issue with matplolib is overlapping subplots or figures. We can use fig.tight_layout() or plt.tight_layout() method, which automatically adjusts the positions of the axes on the figure canvas so that there is no overlapping.

In [6]:
```python
# subplot in OOP and its variations
fig, ax = plt.subplots()    # Single subplot
fig, ax = plt.subplots(1,2)    # two subplots
fig, ax = plt.subplots(2,2)    # four subplots
fig, ax = plt.subplots(1,4)    # four subplots
plt.tight_layout()
```


0.0
0.2
0.4
0.6
0.8
1.0

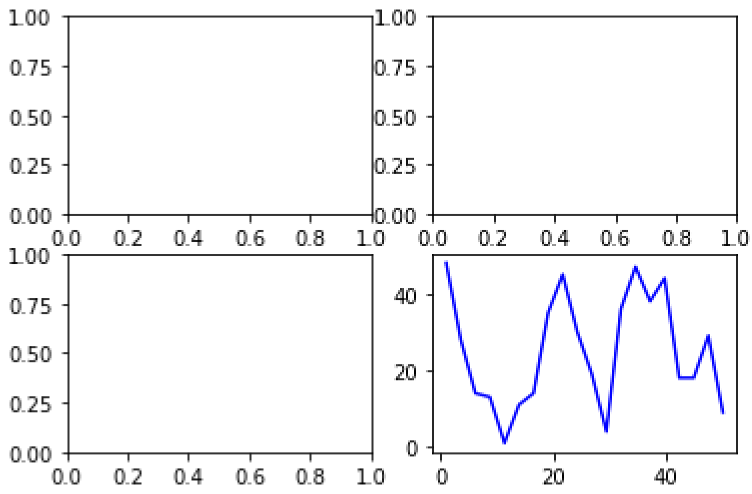

0.0
0.2
0.4
0.6
0.8
1.0


0.00
0.25
0.50
0.75
1.00
0.0
0.2
0.4
0.6
0.8
1.0



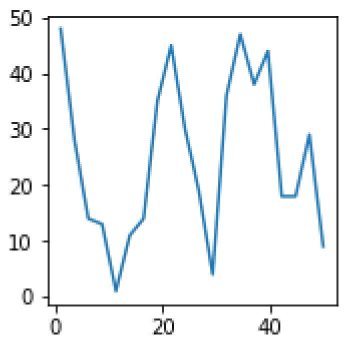
```
In [7]: # To fill value in 4th figure
        fig, ax = plt.subplots(2,2)
        ax[(1,1)].plot(x,y,'b')
```

```
Out[7]: [<matplotlib.lines.Line2D at 0x27ec8bbf550>]
```



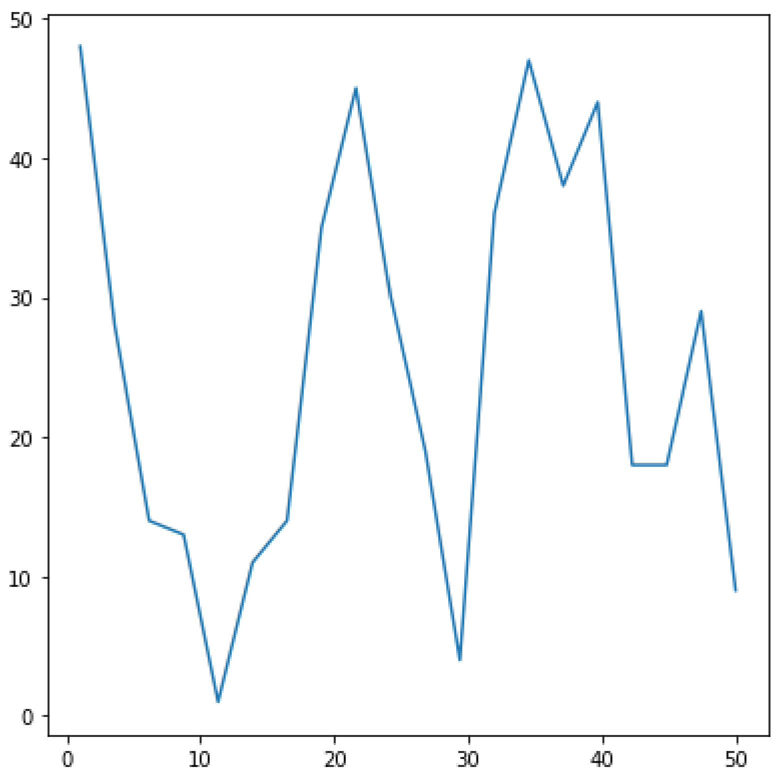
```
In [8]: #Figure Size
        fig = plt.figure(figsize = (2,2))
        axes = fig.add_axes([0,0,1,1])
        axes.plot(x,y)
```

```
Out[8]: [<matplotlib.lines.Line2D at 0x27ec8c19970>]
```



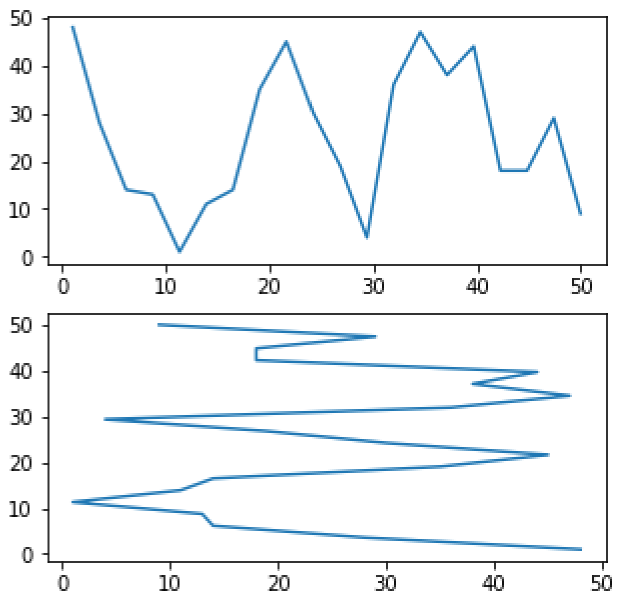
In [9]:
```
# 5,5 dimensions
fig = plt.figure(figsize = (5,5))
axes = fig.add_axes([0,0,1,1])
axes.plot(x,y)
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x27ec8c7efa0>]
```



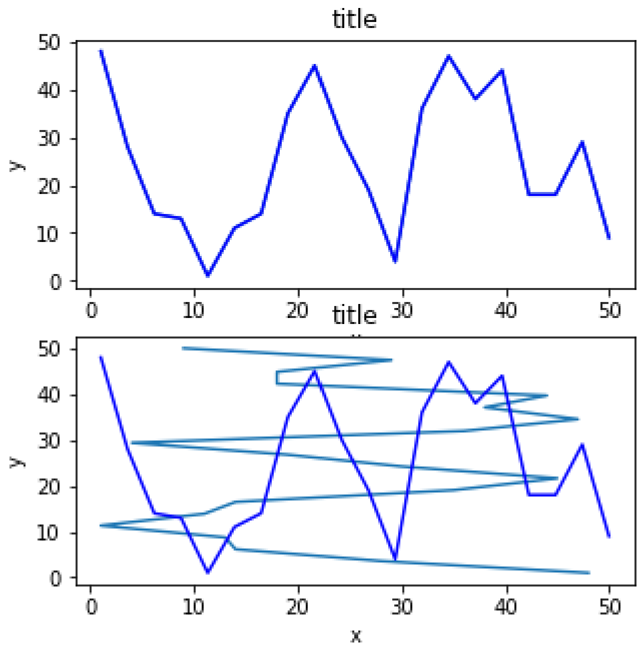
In [10]:
```
# Fig size in subplots
fig, axes = plt.subplots(2,1, figsize = (5,5))
axes[0].plot(x,y)
axes[1].plot(y,x)
```

Out[10]:
```
[<matplotlib.lines.Line2D at 0x27ec896be80>]
```



In [11]:
```
# By using for loop
for ax in axes:
    ax.plot(x, y, 'b')
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_title('title')

# Display the figure object
fig
```

Out[11]:



```
In [12]: #for using different function for different plots
         fig, axes = plt.subplots(1,2)
         axes[0].plot(x,y)
         axes[0].set_xlabel('x')
         axes[0].set_ylabel('y')
         axes[0].set_title('1st plot')
         axes[1].plot(y,x,'r')
         axes[1].set_xlabel('x')
         axes[1].set_ylabel('y')
         axes[1].set_title('2nd plot')
```

```
Out[12]: Text(0.5, 1.0, '2nd plot')
```

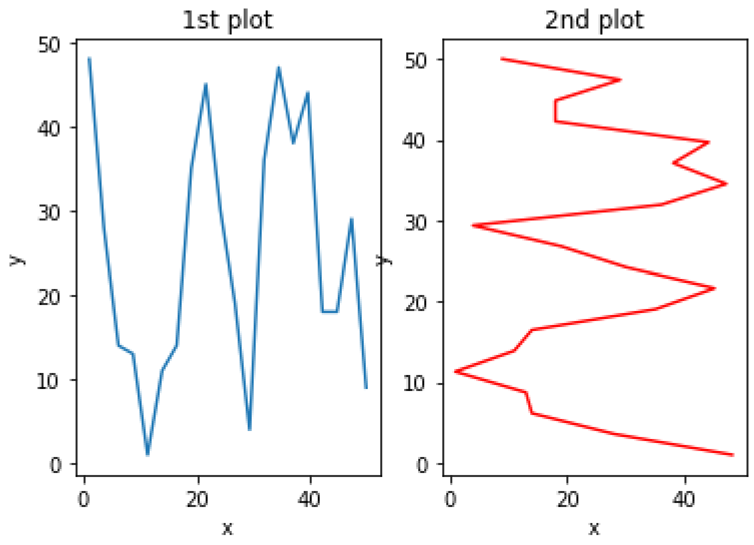


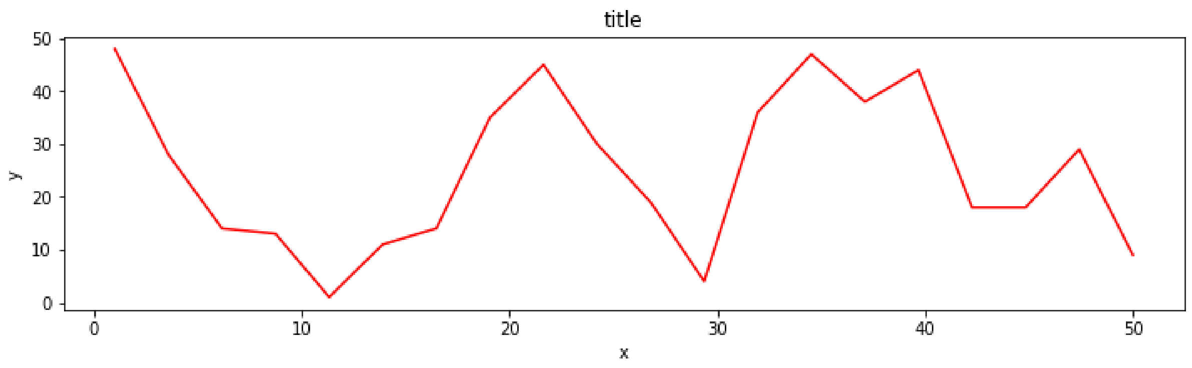
Figure size, aspect ratio and DPI

Matplotlib allows the aspect ratio, DPI and figure size to be specified when the Figure object is created. We can use the figsize and dpi keyword arguments. •figsize is a tuple of the width and height of the figure in inches •dpi is the dots-per-inch (pixel per inch).

```
In [13]: fig = plt.figure(figsize=(8,4), dpi=100)

         <Figure size 800x400 with 0 Axes>
```

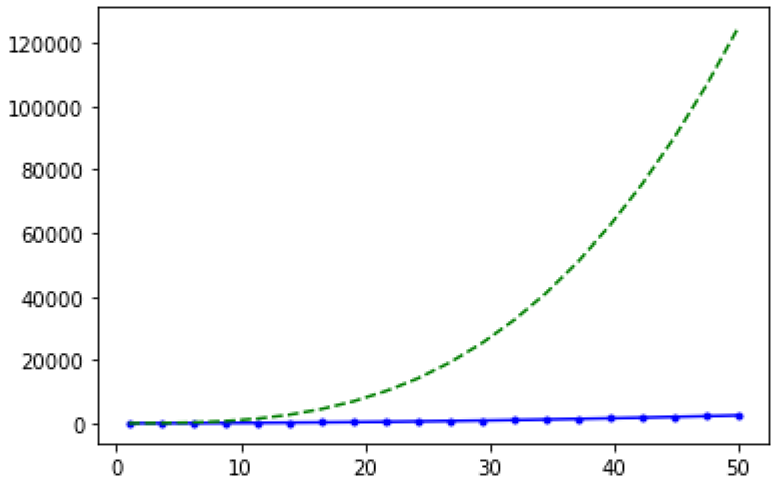
```
In [14]: fig, axes = plt.subplots(figsize=(12,3))

         axes.plot(x, y, 'r')
         axes.set_xlabel('x')
         axes.set_ylabel('y')
         axes.set_title('title')
```

```
Out[14]: Text(0.5, 1.0, 'title')
```



```
In [15]: # MATLAB style line color and style
         fig, ax = plt.subplots()
         ax.plot(x, x**2, 'b.-') # blue line with dots
         ax.plot(x, x**3, 'g--') # green dashed line
```

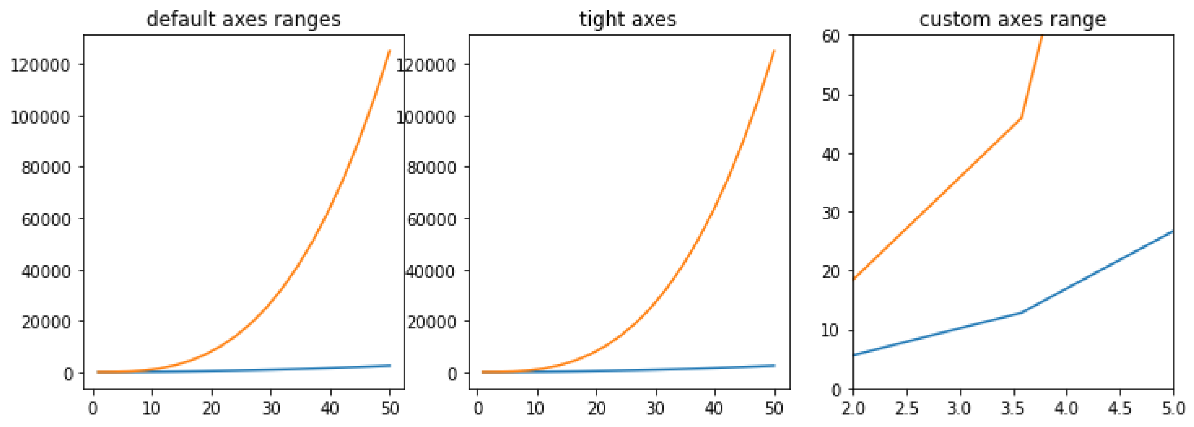
```
Out[15]: [<matplotlib.lines.Line2D at 0x27ec8960130>]
```



Plot range

We can configure the ranges of the axes using the set_ylim and set_xlim methods in the axis object, or axis('tight') for automatically getting "tightly fitted" axes ranges:

```
In [16]: fig, axes = plt.subplots(1, 3, figsize=(12, 4))

         axes[0].plot(x, x**2, x, x**3)
         axes[0].set_title("default axes ranges")

         axes[1].plot(x, x**2, x, x**3)
         axes[1].axis('tight')
         axes[1].set_title("tight axes")

         axes[2].plot(x, x**2, x, x**3)
         axes[2].set_ylim([0, 60])
         axes[2].set_xlim([2, 5])
         axes[2].set_title("custom axes range");
```


default axes ranges
tight axes
custom axes range