Subplots

Subplots

You can create subplots in two different ways:

`fig.add_subplot()`

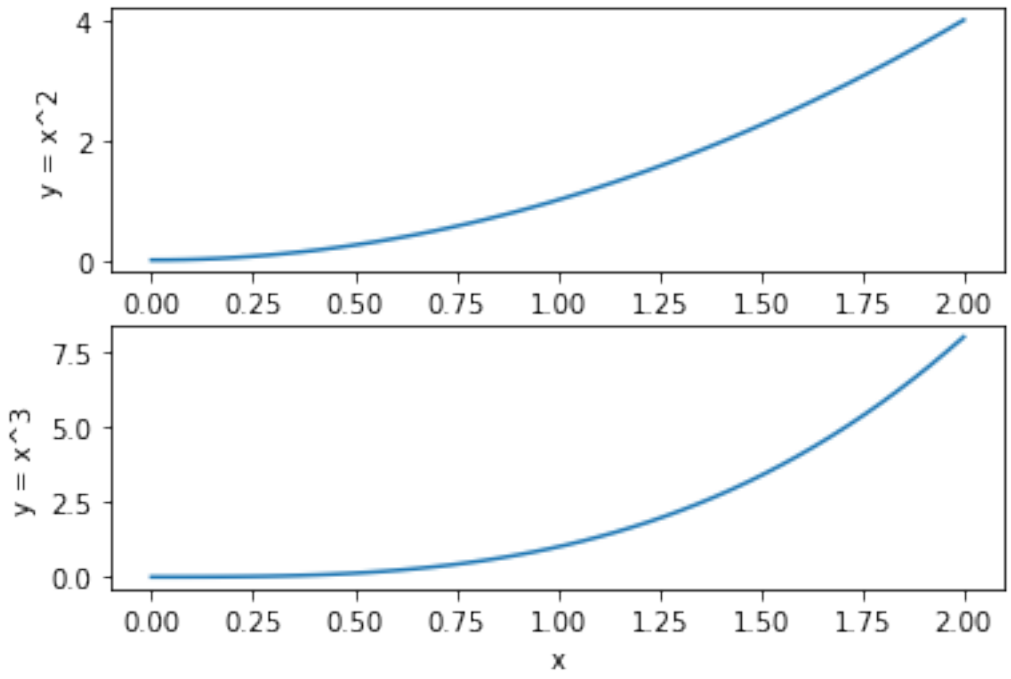
One way to add subplots is by creating a figure and calling the `fig.add_subplot()` method to add an axis to it with (one of) the call signature:

`fig.add_subplot(nrows, ncols, index)`

where `nrows` and `ncols` are the total number of rows and columns of axis and `index` is the position on the grid of axis.

Consider the plot with two rows and a single column:

[6]:
```
x = np.linspace(0, 2)


fig = plt.figure()

#Top axis
ax0 = fig.add_subplot(2, 1, 1)
ax0.plot(x , x**2)
ax0.set_xlabel('x') #Note `set_xlabel` instead of `xlabel`
ax0.set_ylabel('y = x^2')

#Bottom axis
ax1 = fig.add_subplot(2, 1, 2)
ax1.plot(x, x*x*x)
ax1.set_xlabel('x')
ax1.set_ylabel('y = x^3')

plt.show()
```



Refer to the documentation for additional options.

`plt.subplots()`

An alternative way to create subplots is to use the `plt.subplots()` function which returns the figure object and a tuple of axis. The call signature is:

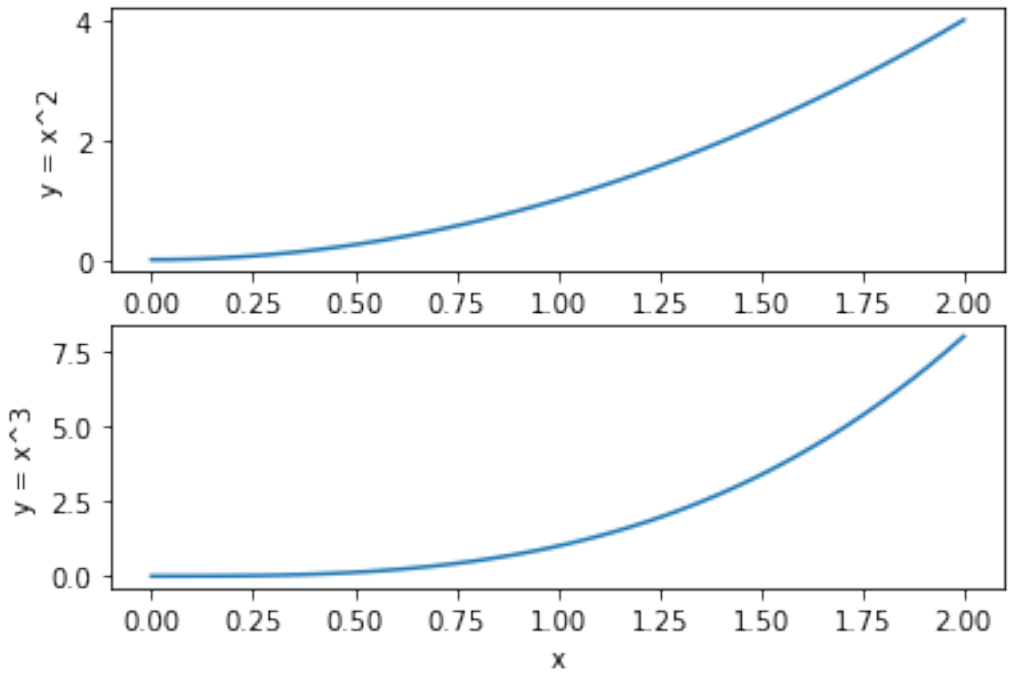
```
plt.subplots(nrows = 1, ncols = 1)
```

where `nrows` and `ncols` are the number of rows an columns as before.

Let's recreate the previous plot using this function:

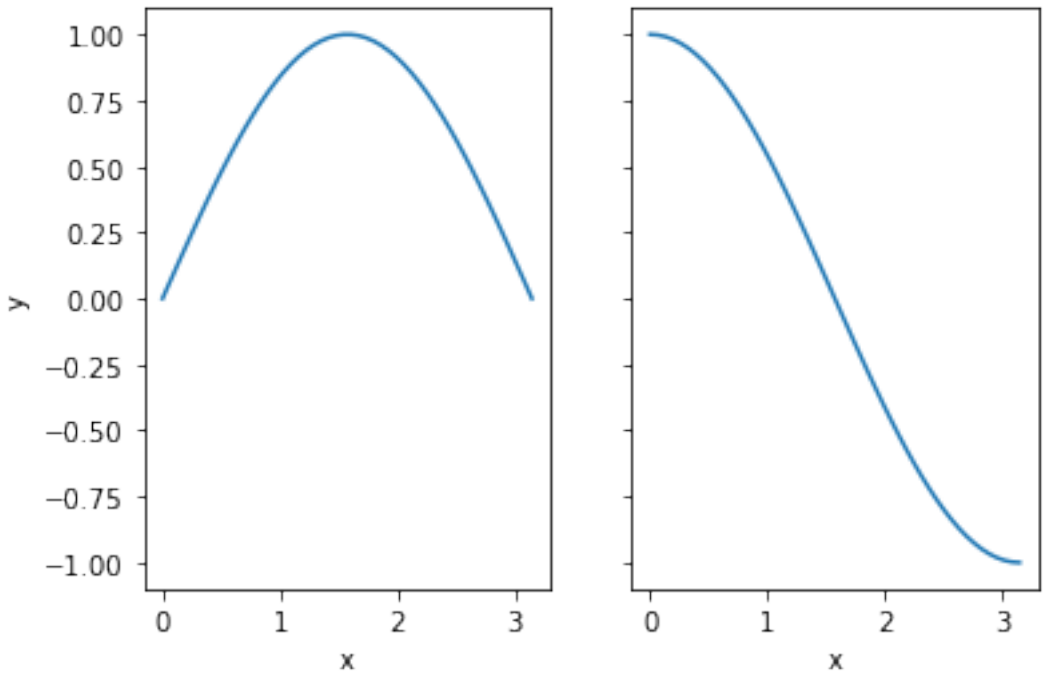
[7]:
```
x = np.linspace(0, 2)

fig, ax = plt.subplots(2, 1)

#Top axis
ax[0].plot(x , x**2)
ax[0].set_xlabel('x') #Note `set_xlabel` instead of `xlabel`
ax[0].set_ylabel('y = x^2')

#Bottom axis
ax[1].plot(x, x*x*x)
ax[1].set_xlabel('x')
ax[1].set_ylabel('y = x^3')
```

```
plt.show()
```



A couple of additional keyword arguments are `sharex` and `sharey`. These take boolean values. If true the subplots will share the relevant axis's ticks. For example:

[11]:
```
x = np.linspace(0, np.pi)

fig, ax = plt.subplots(1, 2, sharey = True)

ax[0].plot(x, np.sin(x))
ax[0].set_xlabel('x')

ax[1].plot(x, np.cos(x))
ax[1].set_xlabel('x')

ax[0].set_ylabel('y') #You can set this for the other axis

plt.show()
```

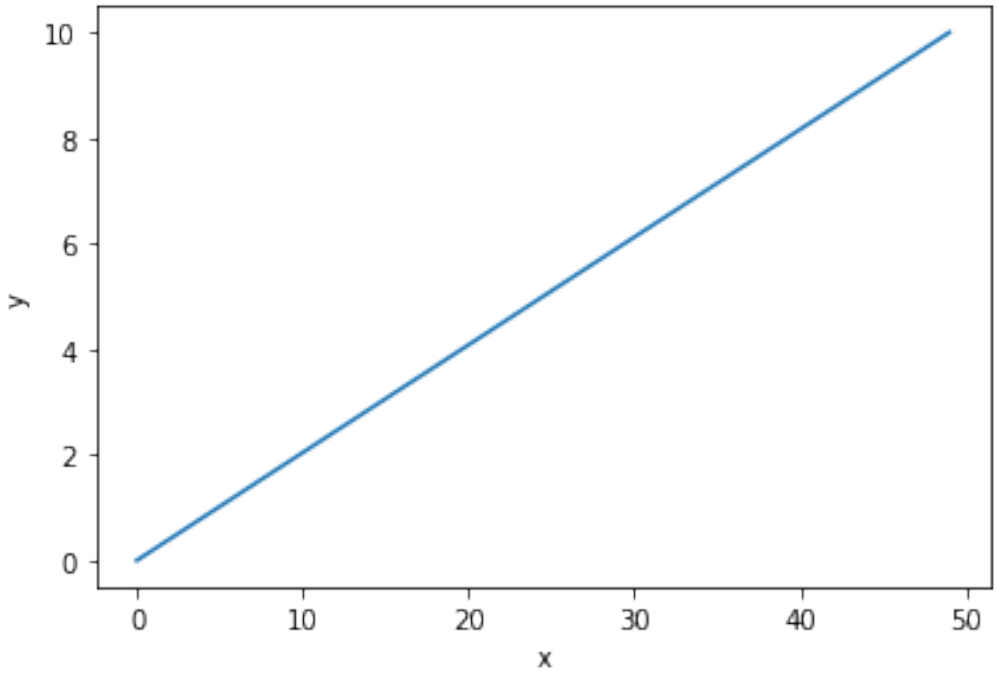


Refer to the documentation for additional options.

Using Subplots For General Plots

The subplot functions above are also used in general practice to create single axis plots, due to the ability to create a reference to the axis, which grants further customization. Simply:

[12]:
```
fig = plt.figure()
ax = fig.add_subplot()

ax.plot(np.linspace(0, 10))
ax.set_xlabel('x')
ax.set_ylabel('y')

plt.show()
```


10
8
6
4
2
0
y
0
10
20
30
40
50
x