How can I use tight_layout() in Matplotlib to improve the spacing and layout of my plot?

Authored by

stats writer

June 25, 2024

Tight_layout() is a function in the Matplotlib library that can be used to improve the spacing and layout of a plot. It automatically adjusts the size and position of the axes, labels, and other elements in the plot to prevent them from overlapping and improve the overall appearance of the plot. This can be particularly useful when creating complex or multi-panel plots. By using tight_layout(), users can easily achieve a more polished and professional look for their plots.

Use tight_layout() in Matplotlib

You can use the tight_layout() function in Matplotlib to automatically adjust the padding between and around subplots.

The following example shows how to use this function in practice.

Example: How to Use tight_layout() in Matplotlib

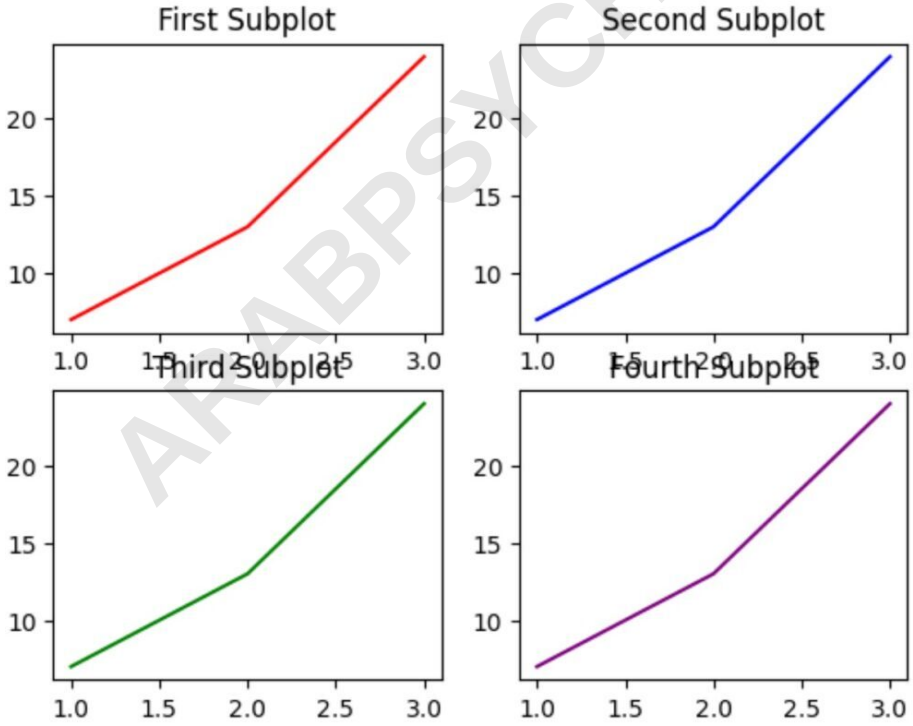
Suppose we use Matplotilb to create four subplots in a 2×2 grid:

import matplotlib.pyplotas plt

#define data
x =
y =

#define layout for subplots
fig, ax = plt.subplots(2, 2)

```
#define subplot titles
ax.plot(x, y, color='red')
ax.plot(x, y, color='blue')
ax.plot(x, y, color='green')
ax.plot(x, y, color='purple')

#add title to each subplot
ax.set_title('First Subplot')
ax.set_title('Second Subplot')
ax.set_title('Third Subplot')
ax.set_title('Fourth Subplot')
```



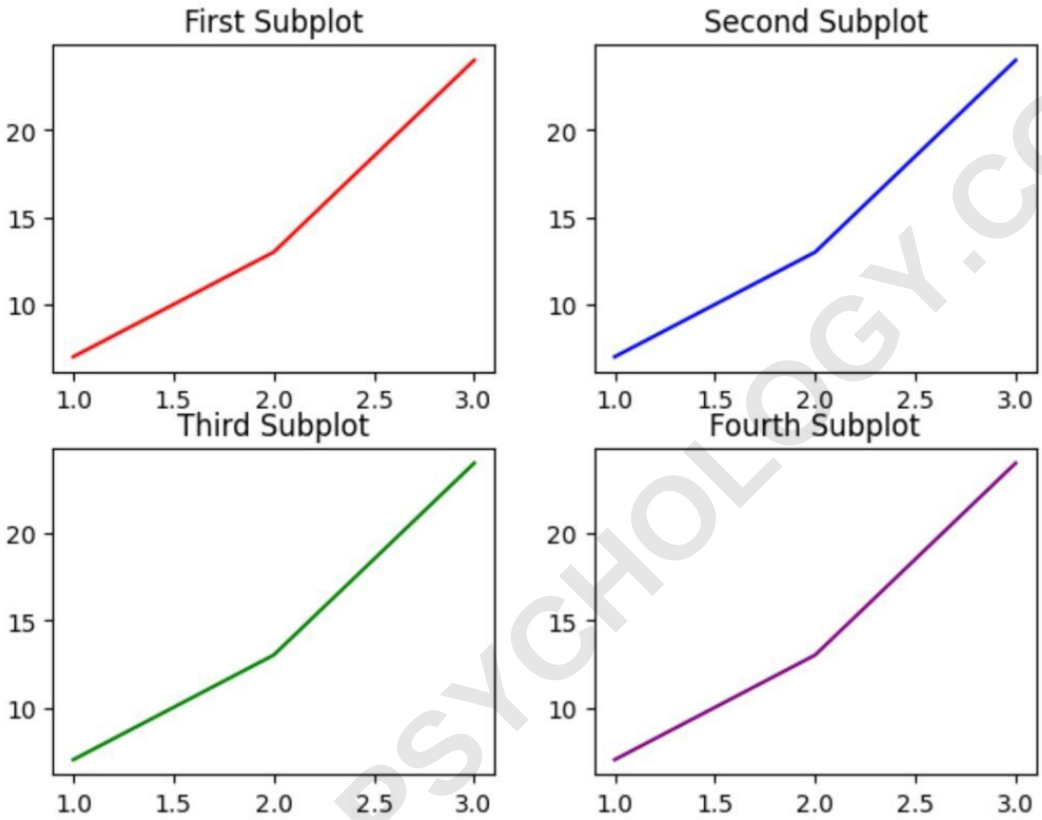
Notice that there is minimal padding between the

subplots, which causes the titles to overlap in some places.

By specifying fig.tight_layout() we can automatically adjust the padding between and around subplots:

```
import matplotlib.pyplotas plt

#define data
x =
y =

#define layout for subplots
fig, ax = plt.subplots(2, 2)

#specify a tight layout
fig.tight_layout()

#define subplot titles
ax.plot(x, y, color='red')
ax.plot(x, y, color='blue')
ax.plot(x, y, color='green')
ax.plot(x, y, color='purple')

#add title to each subplot
ax.set_title('First Subplot')
```

```
ax.set_title('Second Subplot')
ax.set_title('Third Subplot')
ax.set_title('Fourth Subplot')
```



Notice that the padding between and around the subplots has been adjusted so that the plots no longer overlap in any area.

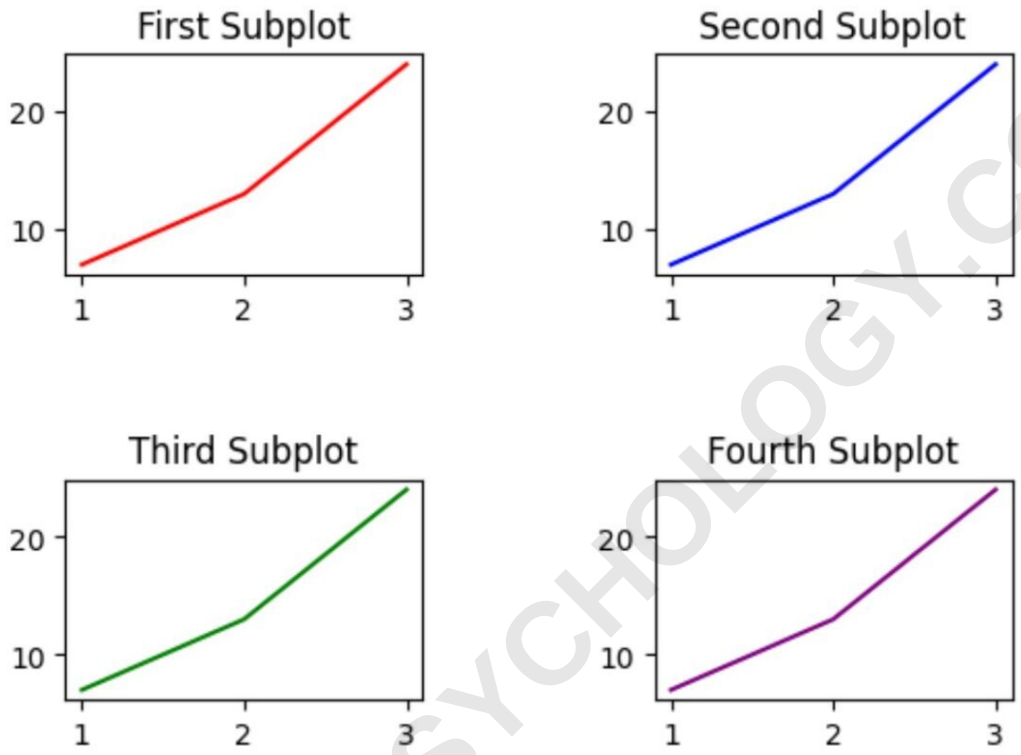
Note that the tight_layout() function takes a pad argument to specify the padding between the figure edge and the edges of subplots, as a fraction of the font

size.

The default value for pad is 1.08. However, we can increase this value to increase the padding around the plots:

```
import matplotlib.pyplotas plt

#define data
x =
y =

#define layout for subplots
fig, ax = plt.subplots(2, 2)

#specify a tight layout with increased padding
fig.tight_layout(pad=5)

#define subplot titles
ax.plot(x, y, color='red')
ax.plot(x, y, color='blue')
ax.plot(x, y, color='green')
ax.plot(x, y, color='purple')

#add title to each subplot
ax.set_title('First Subplot')
```

```
ax.set_title('Second Subplot')
ax.set_title('Third Subplot')
ax.set_title('Fourth Subplot')
```



Notice that the padding around the plots has increased noticeably.

Feel free to adjust the value for the pad argument to increase the padding around the plots as much as you'd like.

The following tutorials explain how to perform other

common tasks in Matplotlib: