关于matplotlib：sharey =’row’和sharey =’True’之间的区别

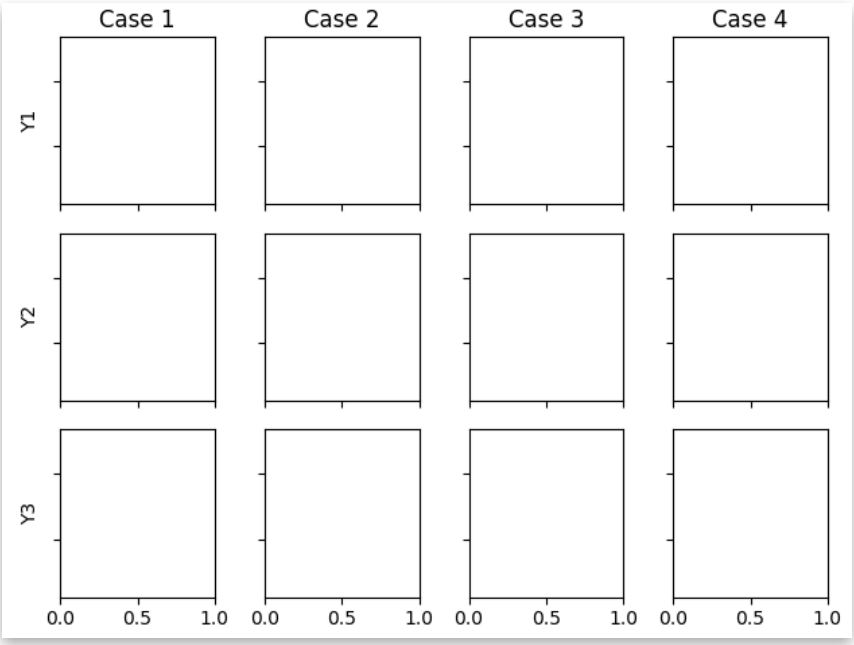
2020-10-30

matplotlib subplot

Difference between sharey='row' and sharey='True'

我正在考虑一个包含3行和4列的图，其中：

对于4个研究案例，要绘制3个因变量： `Y1` ， `Y2` 和 `Y3` ，而使用一个常见的 `X` 自变量。



在这种情况下，有：

1)从 `case i` 到 `case i+1` 时共享 `y` 轴

2)在 `case i` 中共享 `X` 轴

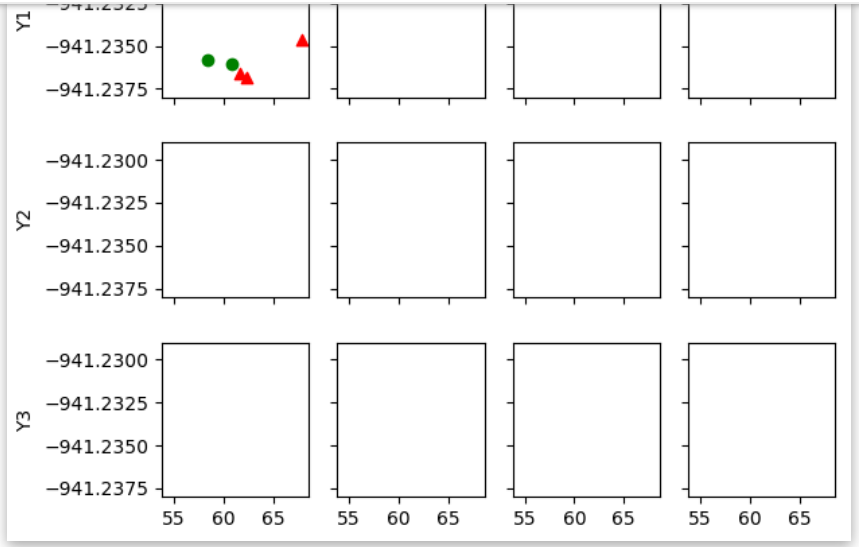
因此，原则上，人们会认为以下代码将产生所需的图(结果显示在上图)：

```
fig, axes = plt.subplots(ncols=4, nrows=3,\\
                         sharex=True, sharey=True,\\
                         subplot_kw=dict(adjustable='box-forced'))
```

如此处所述， `adjustable='box-forced'` 只是为了确保子图平方。

```
import matplotlib.pyplot as plt
import sys

fig, axes = plt.subplots(ncols=4, nrows=3,\\
                         sharex=True, sharey=True,\\
                         subplot_kw=dict(adjustable='box-forced'))

pad = 5
axes[0][0].annotate('Case 1', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

axes[0][1].annotate('Case 2', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

axes[0][2].annotate('Case 3', xy=(0.5, 1), xytext=(0, pad),
               xycoords='axes fraction', textcoords='offset points',
                size='large', ha='center', va='baseline')

axes[0][3].annotate('Case 4', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

#
axes[0][0].set_ylabel('Y1', fontsize=10)
axes[1][0].set_ylabel('Y2', fontsize=10)
axes[2][0].set_ylabel('Y3', fontsize=10)

E_C_I =  np.array([-941.23658347, -941.23685494, -941.23467666])
V_C_I =  np.array([ 61.66341, 62.342903,  67.9311515])
E_14 =  np.array([-941.22938469, -941.23583586, -941.23605613])
V_14 =  np.array([ 54.65693125,  58.47115725, 60.8626545 ])
P_C_I =  np.array([ 2.20068119,  1.33328211,  -4.28370285])
P_14 =  np.array([ 8.16605135,  7.54737315, 0.3909309 ])


axes[0][0].scatter(V_C_I, E_C_I, marker='^', color='red', label='Calcite I')#, s=100)
axes[0][0].scatter(V_14, E_14, marker='o', color='green', label='Calcite I')#, s=100)

axes[0][0].set_ylim(bottom=-941.238, top=-941.229)

plt.tight_layout()
axes[0][0].ticklabel_format(useOffset=False)
plt.show()
sys.exit()
```

一切似乎都很好：



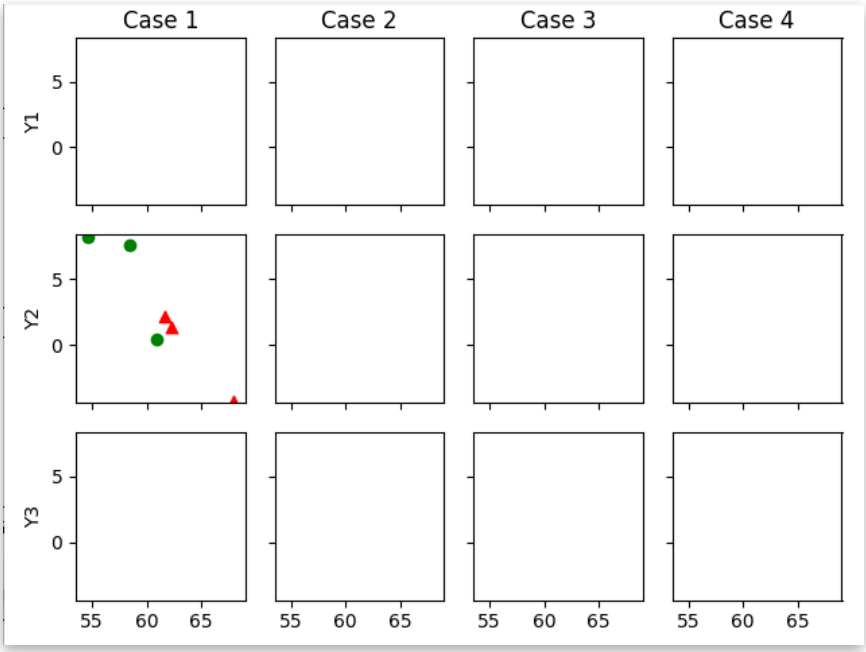
我已将情节强制为 `axes[0][0].set_ylim(bottom=-941.238, top=-941.229)`

当我尝试为 `Case 1` 绘制 `Y2` 与 `X` 时，以下代码应该可以工作：我基本上和以前一样，但是添加了 `axes[1][0]` 绘制指令：

```
import numpy as np
import matplotlib.pyplot as plt
import sys

fig, axes = plt.subplots(ncols=4, nrows=3,\\
                         sharex=True, sharey=True,\\
                         subplot_kw=dict(adjustable='box-forced'))

pad = 5
axes[0][0].annotate('Case 1', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

axes[0][1].annotate('Case 2', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

axes[0][2].annotate('Case 3', xy=(0.5, 1), xytext=(0, pad),
               xycoords='axes fraction', textcoords='offset points',
                size='large', ha='center', va='baseline')

axes[0][3].annotate('Case 4', xy=(0.5, 1), xytext=(0, pad),
                xycoords='axes fraction', textcoords='offset points',
                 size='large', ha='center', va='baseline')

#
axes[0][0].set_ylabel('Y1', fontsize=10)
axes[1][0].set_ylabel('Y2', fontsize=10)
axes[2][0].set_ylabel('Y3', fontsize=10)

E_C_I =  np.array([-941.23658347, -941.23685494, -941.23467666])
V_C_I =  np.array([ 61.66341, 62.342903,  67.9311515])
E_14 =  np.array([-941.22938469, -941.23583586, -941.23605613])
V_14 =  np.array([ 54.65693125,  58.47115725, 60.8626545 ])
```

```
38
39  axes[0][0].scatter(V_C_I, E_C_I, marker='^', color='red', label='Calcite I')#, s=100)
40  axes[0][0].scatter(V_14, E_14, marker='o', color='green', label='Calcite I')#, s=100)
41
42  axes[0][0].set_ylim(bottom=-941.238, top=-941.229)
43
44  axes[1][0].scatter(V_C_I, P_C_I, marker='^', color='red', label='Calcite I')#, s=100)
45  axes[1][0].scatter(V_14, P_14, marker='o', color='green', label='Calcite I')#, s=100)
46
47  axes[1][0].set_ylim(bottom=-4.4, top=8.4)
48
49  plt.tight_layout()
50  axes[0][0].ticklabel_format(useOffset=False)
51  plt.show()
52  sys.exit()
```

结果是 `axes[0][0]` 图已更改其比例，因此未显示任何数据：



我已经强制 `axes[0][0]` 和 `axes[0][1]` 来显示确实有数据的区域：

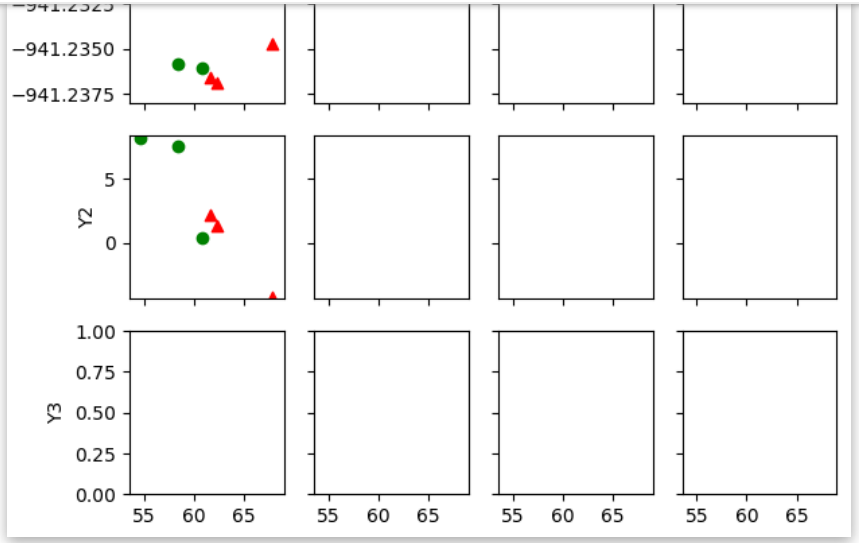
```
1  axes[0][0].set_ylim(bottom=-941.238, top=-941.229)
2  axes[1][0].set_ylim(bottom=-4.4, top=8.4)
```

但是， `axes[0][0]` 图上未显示任何数据。为什么会这样呢？

更新： `sharey='row'` 和 `sharey=True` 之间的区别已在出色的@DavidG答案中阐明。但是，我测试了 `sharex='col'` 和 `sharex=True` 之间的区别，并且我注意到：

```
1  fig, axes = plt.subplots(ncols=4, nrows=3,\\
2                           sharex=True, sharey='row',\\
3                           subplot_kw=dict(adjustable='box-forced'))
```

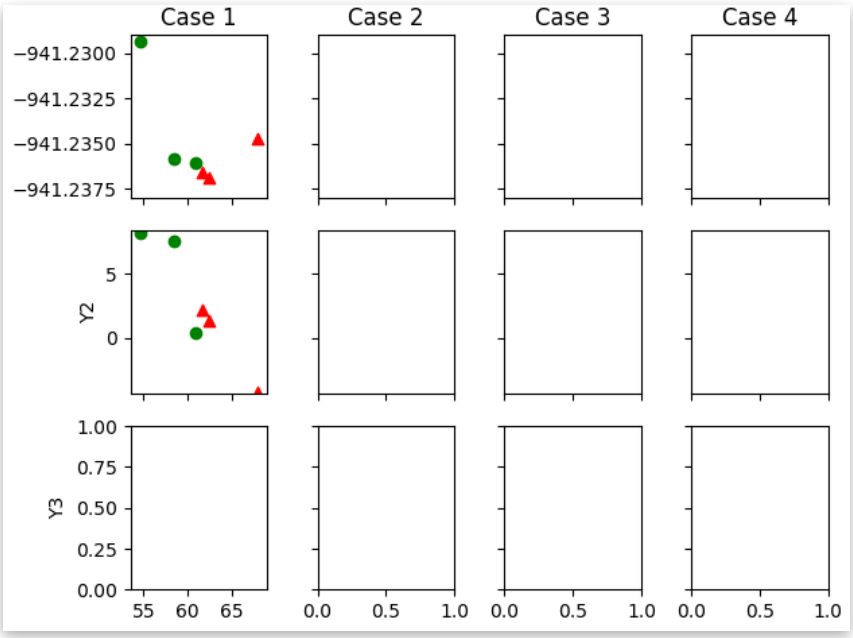
产生以下内容：



然而，

```
fig, axes = plt.subplots(ncols=4, nrows=3,\\
                         sharex='col', sharey='row',\\
                         subplot_kw=dict(adjustable='box-forced'))
```

一种在列之间留出一些空间，并破坏 `adjustable='box-forced'` 声明以使要绘制的子图平方：



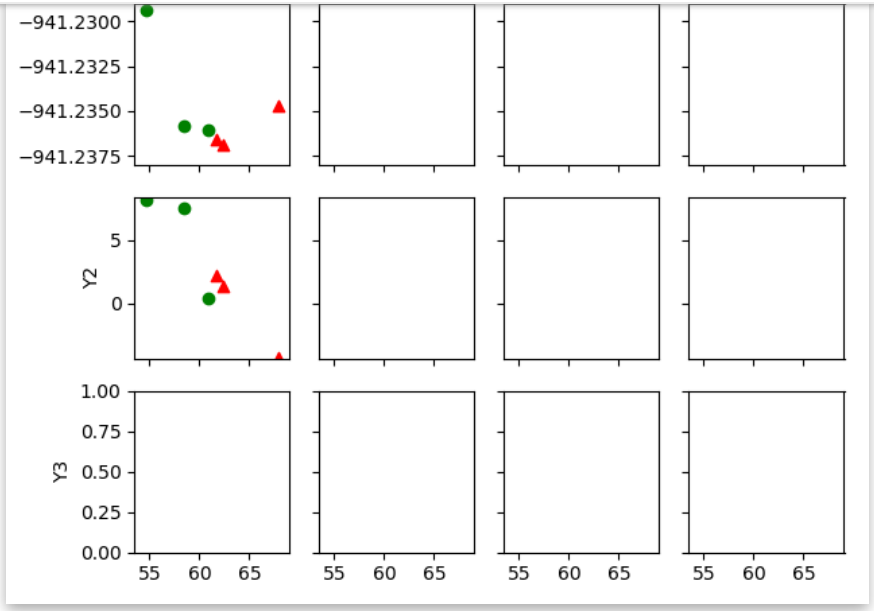
我想知道为什么会这样吗？

您已使用参数 `sharey=True` 将y轴应用于所有子图。

有一个方便的参数 `sharey='row'` ，它将使每行子图共享相同的y轴。 因此，将图形的创建更改为：

```
fig, axes = plt.subplots(ncols=4, nrows=3,\\
                         sharex=True, sharey='row',\\
                         subplot_kw=dict(adjustable='box-forced'))
```

码农家园



相关讨论