Matplotlib 系列之标注

2020-01-27 | Python， 数据可视化 | | | |

1.8k | 2 分钟

我们经常需要在绘图中添加一些箭头、文字之类的标注，让图片更易懂。

基本标注

我们通过 `annotate()` 这个函数来进行快速标注。其中几个比较重要的参数如下：

```
1  plt.annotate(s, xy, xytext, xycoords, textcoords, arrowprops, annotation_clip)
```

- `s`： 标注的文字（字符串）
- `xy`： 标注点的坐标（长度为 2 的序列），以 `xycoords` 为坐标系，标注点就是箭头指向的点
- `xytext`： 注释的坐标（长度为 2 的序列），以 `textcoords` 为坐标系
- `xycoords`： `coords` 是 `coordinates` 之意，箭头（`arrowprops`）的坐标系（字符串类型）

参数	坐标系
`figure points`	以 figure 的左下角为原点，以 point 为单位
`figure pixels`	以 figure 的左下角为原点，以 pixel 为单位
`figure fraction`	figure 左下角为原点，以 fraction（分数）为单位；0,0 左下角；1,1 右上角
`axes points`	以 axes 的左下角为原点，以 point 为单位
`axes pixels`	以 axes 的左下角为原点，以 pixel 为单位
`axes fraction`	以 axes 左下角为原点，以 fraction（分数）为单位；0,0 左下角；1,1 右上角
`data`	使用数据的坐标系，以数据坐标系的单位为单位
`polar`	`(theta,r)` 极坐标系；例如 `(np.pi/2,3)`

- `textcoords`： 指定注释的坐标体系，默认为 `xycoords`。`xycoords` 的参数都可以用在 `textcoords` 中。另外 `textcoords` 还多出两个参数，当且仅当 `textcoords` 默认使用 `xycoords` 的坐标系时使用

参数	坐标系	解释
`offset points`	偏移 `xy` 的量（以 point 为单位）	此时注释的坐标系取决于 `xycoords`
`offset pixels`	偏移 `xy` 的量（以 pixels 为单位）	此时注释的坐标系取决于 `xycoords`

- `arrowprops`：设置箭头形状 `xy` 与 `xytext` 之间的箭头，箭头的类型是 `~matplotlib.patches.FancyArrowPatch`，字典类型，不同的箭头形状有不同的属性

  1. 若 `arrowwprops = None`，则没有 箭头

  2. 如果 `arrowprops` 不包含 `arrowstyle` 这个 key那么 `arrowprops` 允许存在的 key 包括

Key	Description
width	箭头（箭头线）的宽度，以 point 为单位
shrink	从非箭头端向箭头段收缩（shrink）的 fraction（分数比：length of contraction /total length）；收缩的部分不显示
headwidth	箭头底部（也就是箭头）的宽度以 point 为单位
headlength	箭头底部（也就是箭头）的长度以 point 为单位
?	any key to `matplotlib.patches.FancyArrowPatch`

  3. 如果 `arrowprops` 包含 `arrowstyle` 这个 key，那么以面的 key 会被禁止，允许存在的 key 包括：

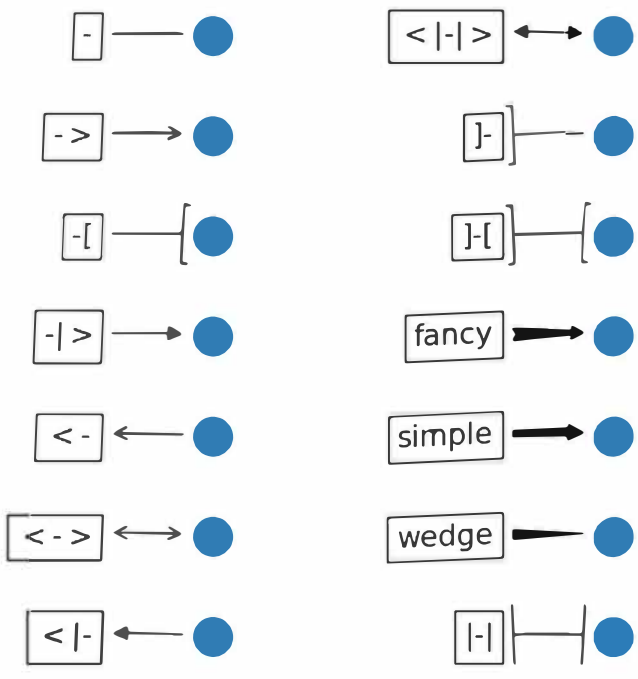
Key	Description
arrowstyle	箭头的样式
connectionstyle	两个点之间的连接路径的样式
relpos	default is (0.5, 0.5)
patchA	default is bounding box of the text
patchB	default is None
shrinkA	default is 2 points
shrinkB	default is 2 points
mutation_scale	default is text size (in points)
mutation_aspect	default is 1.
?	any key for `matplotlib.patches.PathPatch`

  4. `arrowstyle` 的设置如下，其中 name 为箭头形状，attrs 为可设置的属性：

Name	Attrs

Name	Attrs
-	None
->	head_length=0.4,head_width=0.2
-[	widthB=1.0,lengthB=0.2,angleB=None
|-|	widthA=1.0,widthB=1.0
-|>	head_length=0.4,head_width=0.2
<-	head_length=0.4,head_width=0.2
<->	head_length=0.4,head_width=0.2
<|-	head_length=0.4,head_width=0.2
<|-|>	head_length=0.4,head_width=0.2
fancy	head_length=0.4,head_width=0.4,tail_width=0.4
simple	head_length=0.5,head_width=0.5,tail_width=0.2
wedge	tail_width=0.3,shrink_factor=0.5

一些箭头（`fancy`、`simple`、`wedge`）仅适用于生成二次样条线段的连接样式。对于这些箭头样式，必须使用 `angle3` 或 `arc3` 连接样式。



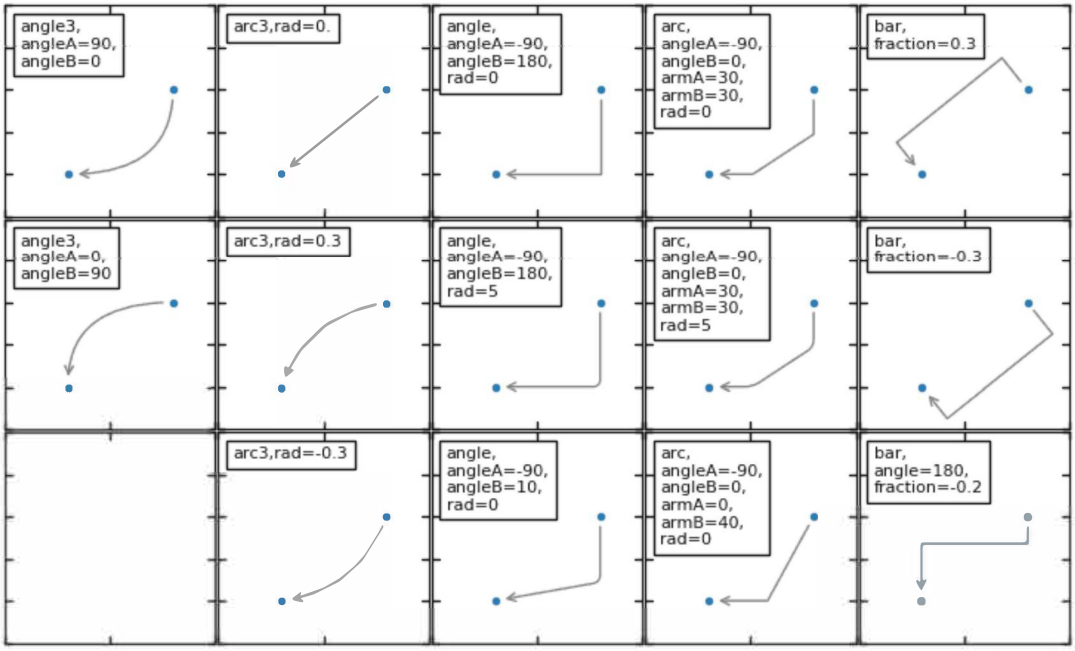
fancyarrow

5. `connectionstyle` 的设置如下：

名称	属性

名称	属性
angle	angleA=90,angleB=0,rad=0.0
angle3	angleA=90,angleB=0
arc	angleA=0,angleB=0,armA=None,armB=None,rad=0.0
arc3	rad=0.0
bar	armA=0.0,armB=0.0,fraction=0.3,angle=None

注意： `angle3` 和 `arc3` 中的 3 意味着所得到的路径是二次样条段（三个控制点）。当连接路径是二次样条时，可以使用一些箭头样式选项。



connectionstyle

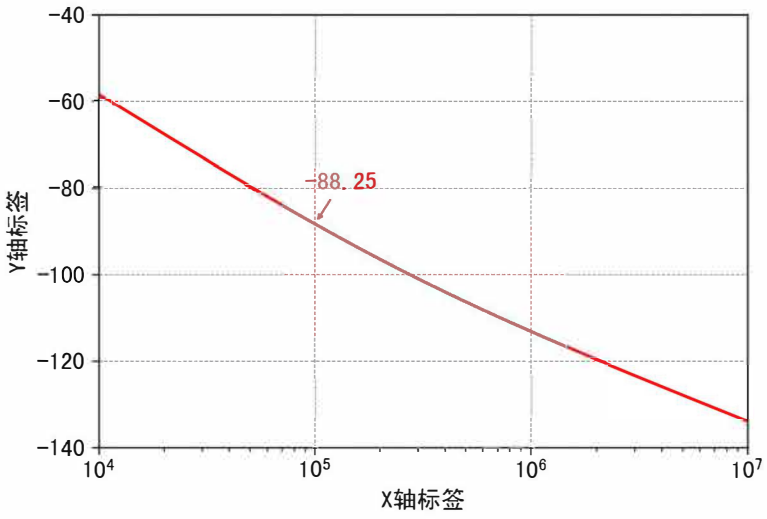
- `annotation_clip`：（bool 型参数），当注释超出轴区域时，控制注释的可见性。如果为 `True`，则只有当 `xy` 位于轴内时才会绘制注释。如果为 `False`，则无论其位置如何，都将始终绘制注释。默认值为 `None`，仅当 `xycoords` 为 `data` 时才表现为 `True`

小例子

下面是一个例子的绘图部分代码：

```
fig = plt.figure()
axes = fig.add_axes([0.15, 0.15, 0.75, 0.75])

# 数据曲线
l1, = axes.plot(df_pn0['freq'],df_pn0['pn'],color = 'r',label="")

# 标注
m1 = df_pn0[df_pn0['freq'] == 1e5]['pn'].values[0]
x1 = 1e5
text1 = str(round(m1,2))
```

```
# 箭头
arrowprops=dict(arrowstyle='-|>',mutation_scale=5)
axes.annotate(s=text1, xy=(x1,m1), xytext=(x1-10000, m1+8),arrowprops=arrowprops,color = 'r',fontsize = 12)

# 坐标轴设置
axes.set_xscale('log')
# 设置x、y轴范围
axes.set_xlim([1e4,1e7])
axes.set_ylim([-140,-40])

# 设置坐标轴标签
axes.set_xlabel("X轴标签",fontsize = 12)
axes.set_ylabel("Y轴标签",fontsize = 12)

# 设置刻度字体大小
plt.xticks(fontsize = 12)
plt.yticks(fontsize = 12)

# 格线
axes.grid(color='black', alpha=0.5, linestyle='dashed', linewidth=0.5)

plt.show()
```



标注小例子

其中标注部分的关键代码为：

```
m1 = df_pn0[df_pn0['freq'] == 1e5]['pn'].values[0]
x1 = 1e5
text1 = str(round(m1,2))

arrowprops=dict(arrowstyle='-|>',mutation_scale=5)
axes.annotate(s=text1, xy=(x1,m1), xytext=(x1-10000, m1+8),arrowprops=arrowprops,color = 'r',fontsize = 12)
```

文本标注

我们可以使用 `text()` 函数进行快速文本标注，其参数如下：

```
text(x, y, s, fontdict=None, withdash=False, **kwargs)
```

- x，y：（标量 scalars），放置文本的坐标位置，默认是 data coordinates。coordinate system 可以通过 transform 参数改变
- s：（字符串 str），标注的文本
- fontdict：（字典 dict），设置文本属性（字体大小、字体颜色等），例子：fontdict={'size': 16, 'color': 'r'}
- withdash：（布尔类型 bool）、（可选参数）、默认为 False，创建一个 ~matplotlib.text.TextWithDash 实例而不是 ~matplotlib.text.Text 实例