matplotlib.pypltot【plt接口】

1. figure(num=1, figsize=(15, 8), dpi=80)：创建一个画布，同时设置大小，分辨率；

8. get\_axes()：

2. gcf()：获得当前figure对象；

3. gca()：获取当前axes对象；

4. clf()：清理当前的figure对象；

5. cla()：清理当前的axes对象；

6. close()：释放figure所占用的；

7. getp(args1, args2)：打印对象的属性值，参数args1为对象，args2为属性名，若不指定则为所有属性和其值

3. subplots(nrows=x, ncols=y, figsize=plt.figaspect(1./2))：创建一个画布，同时创建x,y个子图，并同时Figure对象，Axes对象；

4. title()：设置图表标题；

5. grid()：设置图标网格；

6. axis([a,b,c,d])：设置横纵坐标轴范围，与ax.set\_xlim()/ax.set\_ylim()等效；

7. xlabel('xxx')/ylabel('xxx')：设置x、y轴标题；

8. xticks([a,b,c],[x,y,z])/yticks([a,b,c],[x,y,z])：设置x、y轴刻度及对应的名称；

9. xlim((x, y))/ylim((x, y))：设置x、y轴范围；

10. annotate('xxx', xy=(x0, y0), xytext=(x1, y1), xycoords='args1', weight ='args2', textcoords='args3', fontsize=16, arrowprops=dict(arrowstyle='->', connectionstyle="arc3, rad=.2"))：设置标注；参数：xxx：标注的内容；(x0, y0)：注释文字的坐标位置；xytext：注释文字的坐标位置；xycoords：所使用的坐标系【figure points：图左下角的点；figure pixels：图左下角的像素；figure fraction：图的左下部分；axes points：坐标轴左下角的点；axes pixels：坐标轴左下角的像素；axes fraction：左下轴的分数；data：使用被注释对象的坐标系统(默认)；offset points/ offset pixels：相对于被注释点xy的偏移量；weight：设置字体线型【ultralight, light, normal, regular, book, medium, roman, semibold, demibold, demi, bold, heavy, extra bold, black】；color：设置字体颜色【'b', 'g', 'r', 'c', 'm', 'y', 'k', 'w'】；arrowprops：设置箭头参数，参数类型为dict【arrowstyle：箭头的样式，参数'->','|-|','-|>'；connectionstyle：设置箭头的连线，曲线或直线，参数'arc3','arc','angle','angle3'；width：箭头的宽度(以点为单位)；headwidth：箭头底部以点为单位的宽度；headlength：箭头的长度(以点为单位)；shrink：箭头两端收缩的百分比；facecolor：箭头颜色】；bbox：给注释增加外框，参数类型为dict【boxstyle：设置风格，参数round，圆形方框；rarrow，箭头；facecolor，设置背景颜色；edgecolor，设置边框线条颜色；lineweigh，设置边框线型粗细；alpha，设置透明度】

11. text(x, y, s, fontdict, withdash, verticalalignment, horizontalalignment)：设置无指向型的注释，即只会在图中添加注释内容而无指向箭头；参数：x, y位置坐标；s：表示说明的文字；fontdict：字体；withdash：宽度；verticalalignment：垂直对齐方式【'center' | 'top' | 'bottom' | 'baseline'】, horizontalalignment，水平对齐方式【'center' | 'right' | 'left'】

12. tight\_layout()：避免出现叠影；

画图：plot(x, y1, color='red', linewidth=1.0, linestyle='--')，返回line对象；

matplotlib.figure.Figure类【画布】

用于绘图的图表类，负责管理一个图形窗口的子图(Axes)，各种图表组件的绘制，参数定义：num，图像编号或名称，数字为编号 ，字符串为名称；figsize，指定figure的宽和高，单位为英寸；dpi，参数指定绘图对象的分辨率，即每英寸多少个像素，缺省值为80；facecolor，背景颜色；edgecolor：边框颜色；frameon：是否显示边框；frameon：是否显示边框；

1. add\_subplot(x,y,z)：将画布分成x,y个子图，并返回第z个子图(Axes对象), 参数为x,y,z；

2. add\_axes([left, bottom, width, height])：新增子图，参数为列表，分别表示[left, bottom, width, height]占figure的百分比，返回该子图(Axes对象)；

3. add\_artist(a)：将由legend()函数产生的图例a添加至画布；

3. add\_gridspec(x,y)：将画图布局按照x,y进行“分区”，并返回GridSpec对象，该对象可作为add\_subplot的对象，如gs[0,:]；

4. align\_labels()：对齐子图标签，等同于fig.align\_xlabels(); fig.align\_ylabels();

5. autofmt\_xdate()/autofmt\_ydate()：改变x，y轴坐标的显示方法可以斜着表示,不用平着挤一堆；

6. suptitle(‘xxx’) ：设置标题；

7. subplots\_adjust(left=int, bottom= int, right= int, top= int, wspace= int, hspace= int)：设置窗口内边距，即留白。

7. gca()：就是获得当前的坐标轴的句柄，与plt.gca()等价；

9. set\_size\_inches(int,int)：设置figure的尺寸大小，参数为int；

10. set\_facecolor('g')：设置背景色；

11. savefig(fname)：将画布保存至文件路径；

12. clf()

matplotlib.axes.Axes类【子图】

1. set\_title('xxx')：对子图命名；

2. set\_xlim(a,b)/set\_ylim(a,b)：分别对x轴和y轴坐标的范围进行设置，(a~b)；

3. set\_xlabel('xxx')/ set\_ylabel('xxx')：分别对x轴和y轴的标题进行设置；

4. set\_xticks([1,2,3,4,5,6])/Axes.set\_yticks([1,2,3,4,5,6])：设置显示的刻度

5. set\_xticklabels/set\_yticklebels ([a,b,c,d], rotation=-30, fontproperties='SimHei', fontsize=12)：设置刻度标签的内容，旋转角度，字体及大小；

6. spines(['args'])：返回边框对象，其中参数为'right', 'top'

7. set\_axis\_off()：设置坐标轴是否显示；

8. set\_frame\_off()：设置坐标轴边框是否显示；

9. get\_legend()：获取legend对象；

10. get\_legend\_handles\_labels()：获取legend对象的句柄handles和标签labels；

9. legend(loc='upper left')：设置显示图例，与plt.legend()等同，参数【x, y (loc), 设置图例位置；fontsize，设置图例字体；frameon，edgecolor，facecolor，设置图例边框及背景；】

10. annotate()：设置标注；

10. tick\_params(axis/which/ direction, length/width, colors)：axis【‘x’, ‘y’, ‘both’】，设置轴的属性；which【'major', 'minor', 'both'】，设置主/副刻度线；direction【’in’, ‘out’, ’inout’】刻度线显示在绘图区内侧、外侧；length/width，分别用于设置刻度线的长度和宽度；

11. fill\_between('x', 'y1', 'y2', color='yellow', data=data\_obj)：填充两条线之间的颜色；

12. axhline/ axvline(0, color='gray', linewidth=2)：在水平或者垂直方向上画线；

13. get\_lines()：返回子图上有plot()函数生成的Line2D对象的列表

画图：plot；scatter；hist；bar；pie

xaxis/yaxis【Axis坐标轴】

1. set\_ticks\_position('args')：设置坐标刻度数字或名称的位置，参数：'bottom'’，返回坐标轴；

2. set\_visible(True)：设置坐标轴可见；

3. label()：获取坐标轴的label对象；

4. set\_major\_locator()/ set\_minor\_locator()：设置主副坐标，参数MultipleLocator(int)，定义主刻度标签的刻度间隔；

5. grid(True, which='major/minor')：设置坐标轴的网格使用定义的主刻度格式；

6. set\_axisline\_style("->", size = a) ：设置坐标轴箭头属性，和大小

7. set\_axis\_direction('top')：设置刻度标识位置

8. get\_major/minor\_ticks()：获取主/副坐标轴

9. get\_ major/minor\_locator()：获取主副坐标轴刻度的数值定位方式，返回locator对象；

10. set\_major/minor\_locator(MultipleLocator(xx/ plt.NullLocator()))：设置主副坐标轴刻度的数值定位方式，参数为MultipleLocator，定义为将主刻度标签设置为xx的倍数；

11. get\_ major/minor\_formatter()：获取主副坐标轴刻度的格式，返回formatter对象；

12. set\_ major/minor\_formatter(FormatStrFormatter(xx/plt.NullFormatter()))：设置主副坐标轴刻度的格式，参数为FormatStrFormatter；

matplotlib.spines【边框对象】

1. set\_position(('args1', 0))：设置边框位置，参数args1：outward，axes，data；

2. set\_color('none')：

matplotlib.legend【图例对象】

1. get\_texts()

matplotlib.ticker【坐标轴主副刻度对象】

1.

matplotlib.text.Annotation 【标注】

locator【坐标轴定位器】

1. AutoLocator()：默认值，自动对刻度线的位置进行设置；

2. MaxNLocator()：根据提供的刻度线的最大刻度，自动设置；

3. IndexLocator()：根据起始位置和间隔来设置刻度线；

4. MultipleLocator()：根据指定的间隔来设置刻度线；

5. FixedLocator()：根据提供的列表元素来设置刻度线；

6. LinearLocator()：从 min 到 max 均匀分布刻度

7. LogLocator()：从 min 到 max 按对数分布刻度

8. NullLocator()：不显示刻度线；

formatter【格式生成器】

1. NullFormatter()：刻度上无标签

2. IndexFormatter()：将一组标签设置为字符串

3. FixedFormatter()：手动为刻度设置标签

4. FuncFormatter()：用自定义函数设置标签

5. FormatStrFormatter()：为每个刻度值设置字符串格式

6. ScalarFormatter()：（默认）为标量值设置标签

7. LogFormatter()：对数坐标轴的默认格式生成器

matplotlib.lines.Line2D【线对象】

1. set\_antialiased(False)：关闭对象的反锯齿效果

2. get\_linewidth()：获取线宽；

matplotlib.widgets【视图对象】

1. AxesWidget(ax)：

2. Button(ax, label, image=None, color='0.85', hovercolor='0.95')：按钮

3. CheckButtons(ax, labels, actives=None)：复选框；

4. Cursor(ax, horizOn=True, vertOn=True, useblit=False, \*\*lineprops)：显示鼠标十字光标

5. EllipseSelector(ax, onselect, drawtype='box', minspanx=0, minspany=0, useblit=False, lineprops=None, rectprops=None, spancoords='data', button=None, maxdist=10, marker\_props=None, interactive=False, state\_modifier\_keys=None)：椭圆形选区

6. Lasso(ax, xy, callback=None, useblit=True)：锁套；

7. LassoSelector(ax, onselect=None, useblit=True, lineprops=None, button=None)：套索选区

8. LockDraw

9. MultiCursor(canvas, axes, useblit=True, horizOn=False, vertOn=True, \*\*lineprops)：同一图像内多子图共享光标信息；

10. PolygonSelector(ax, onselect, useblit=False, lineprops=None, markerprops=None, vertex\_select\_radius=15)：多边形选区

11. RadioButtons(ax, labels, active=0, activecolor='blue')：单选框

12. RectangleSelector(ax, onselect, drawtype='box', minspanx=0, minspany=0, useblit=False, lineprops=None, rectprops=None, spancoords='data', button=None, maxdist=10, marker\_props=None, interactive=False, state\_modifier\_keys=None)：矩形选取

13. Slider(ax, label, valmin, valmax, valinit=0.5, valfmt=None, closedmin=True, closedmax=True, slidermin=None, slidermax=None, dragging=True, valstep=None, orientation='horizontal', \*\*kwargs)

14. SpanSelector(ax, onselect, direction, minspan=None, useblit=False, rectprops=None, onmove\_callback=None, span\_stays=False, button=None)

15. SubplotTool(targetfig, toolfig)

16. TextBox(ax, label, initial='', color='.95', hovercolor='1', label\_pad=0.01)

17. ToolHandles(ax, x, y, marker='o', marker\_props=None, useblit=True)

18. Widget()

Formatter对象坐标轴轴字体设置

Artist：

Axes.plot()绘制曲线，返回Line2D类型的对象；

Axes.

Axes.scatter()绘制散点图，返回PathCol；lecton类型的对象

坐标轴matplotlib.axes.Axis类

X

图例matplotlib.legend.Legend类

其他设置：

plt.rcParams["font.family"]="SimHei" # 用于设置显示字体的名称

plt.rcParams["font.style"]="normal" # 用于设置字体风格，正常‘normal’ 或者斜体‘italic’

plt.rcParams["font.size"]="large" # 用于字体大小的设置，参数为证书字号或者“large”, “x-small”

7. axes

8. clear

clf

colorbar

contains

delaxes

dpi

draw

draw\_artist

execute\_constrained\_layout

figimage

frameon

get\_axes

get\_children

get\_constrained\_layout

get\_constrained\_layout\_pads

get\_default\_bbox\_extra\_artists

plt.与ax.区别：

方法1：

fig=plt.figure(num=1,figsize=(4,4)) # 生成了一个画布

plt.plot([1,2,3,4],[1,2,3,4]) # 在画布上隐式的生成一个画图区域来进行画图

plt.show()

方法2：

fig=plt.figure(num=1,figsize=(4,4)) # 生成了一个画布

ax=fig.add\_subplot(111) #将画布分为1x1的分布，并将第一个子图赋值给ax

ax.plot([1,2,3,4],[1,2,3,4]) # 在子区域生成一个子图

plt.show()

等同于

fig, ax = plt.subplots(nrows=1, ncols=1)

ax.plot([1,2,3,4],[1,2,3,4]) # 在子区域生成一个子图

plt.show()

方法3：

fig=plt.figure(num=1,figsize=(4,4)) #

plt.subplot(111) # 将画布分为1x1的分布，并选中第一个子图进行操作

plt.plot([1,2,3,4],[1,2,3,4]) # 在子区域生成一个子图

plt.show()

