

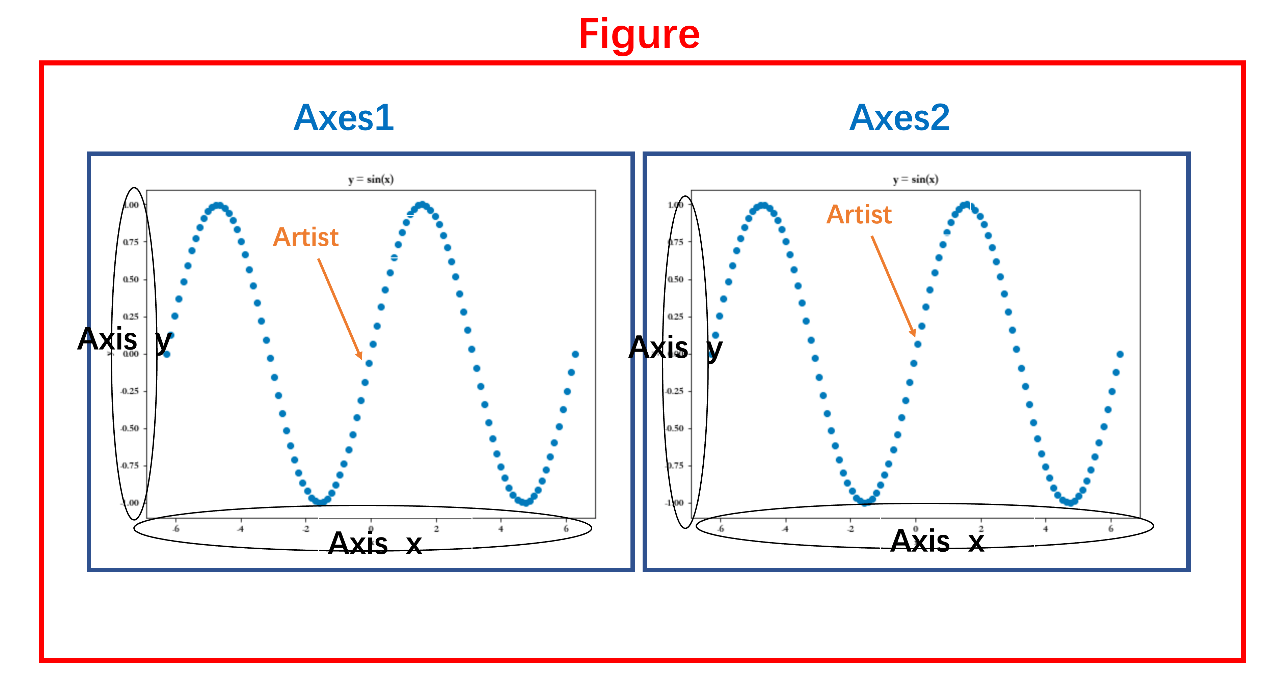
## 绘图元素

一般来说，Mayplotlib绘图元素包括：基本绘图类型与容器绘图类型。下面，我们逐一看看：

（1）基本绘图类型（graphic primitives）：点 (marker)、线 (line)、文本 (text)、图例 (legend)、网格线 (grid)、 标题 (title)、图片 (image) 等；

（2）容器绘图类型（containers）：

* Figure：最重要的元素，代表整个图像，所有的其他元素都是绘制在其上（如果有多个子图，子图也绘制在figure上）。Figure 对象包含一些特殊的 artist 对象，如图名 (title)、图例 (legend)。
* Axes：第二重要的元素，代表 subplot（子图），数据都是显示在这个区域。一个Figure至少含有一个Axes对象，当绘制多个子图时就是axes列表
* Axis：代表坐标轴对象，本质是一种带装饰的 spines，一般分为 xaxis 和 yaxis，Axis对象主要用于控制数据轴上的刻度位置和显示数值。
* Spines：表示数据显示区域的边界，可以显示或不显示。
* Artist：表示任何显示在 Figure 上的元素，Artist 是很通用的概念，几乎任何需要绘制的元素都可以当成是 Artist，但是一个 Artist 只能存在于一个 Axes 之上。



一般来说，要使用Matplotlib画出一副图表，需要设置一个容器绘图类型，再在容器内添加基本绘图对象，如数据元素：点 (marker)、线 (line)、文本 (text)、图例 (legend)、网格线 (grid)、 标题 (title)、图片 (image) 等。除了数据元素以外，还需要设置**图表元素**，包括图表尺寸、坐标轴的轴名及其标签、刻度、图例、网格线等。

## 图表元素

| **函数** | **函数功能** | **核心参数设置举例** |
| --- | --- | --- |
| figure() | 设置图表的大小figsize与分辨率dpi | plt.figure(figsize=(6,3), dpi=75) |
| title() | 设置图表标题 | plt.title("这是一个标题") |
| xlabel()/ylabel() | 设置x轴和y轴标签 | plt.xlabel("这是x轴标签")/plt.ylabel("这是y轴标签") |
| axis()/xlim()/ylim() | 设置x轴和y轴的显示范围 | plt.axis(xmin=0, xmax=1, ymin=0.5)/ plt.xlim(0,1)/plt.ylim(0,1) |
| xticks()/yticks() | 设置x轴和y轴的刻度数值 | plt.xticks(labels=['A', 'B', 'C', 'D'])/plt.yticks(list(np.arange(11)) |
| grid() | 设置网格线 | plt.grid(b=True, which='both', axis='y') # b: 是否显示网格线 True/None; # which: 网格线显示的尺度。字符串，可选参数，取值范围为{'major', 'minor', 'both'}，默认为'both'。'major'为主刻度、'minor'为次刻度。# axis: 选择网格线显示的轴。字符串，可选参数，取值范围为{'both', 'x', 'y'}，默认为'both'`。 |
| legend() | 设置图表的图例 | plt.legend("这是一个图例", loc='best') # loc表示图例的位置，可选：0-'best', 1-'upper right',2-'upper left',3-'lower left',4-'lower right',5-'right',6-'center left',7-'center right', # 8-'lower center',9-'upper center',10-'center' |

## 能画哪些图

绘图函数图表类型绘图函数参数

### plt.plot()折线图，其实也可以画散点图

* x
* y
* color 线条颜色
* linestyle线条类型
* linewidth 线条宽度
* marker 点的形状
* markeredgecolor 点的边框颜色
* markeredgewidth 点的边框的宽度
* markerfacecolor 点的填充颜色
* markersize 点的大小
* label 线图的标签（常配合legend）

### plt.scatter()散点图、气泡图

* x
* y
* s 散点的大小（size）
* c 散点的颜色（color）
* marker 散点的形状类型
* linewidths 散点的边框宽度
* edgecolors 散点的边框颜色
* label 点图的标签（常配合legend）

### plt.bar()柱状图、堆叠柱状图

* x
* height 柱子的高度
* width 柱子的宽度
* align 柱子与x的对齐方式，有：{'center'居中对齐, 'edge'左侧对齐}，默认为'center'
* color 柱子的填充颜色
* edgecolor 柱子边框的颜色
* linewidth 柱子边框线的宽度

### plt.barh()横向柱状图

* y
* height 柱子的高度
* width 柱子的宽度
* align 柱子与x的对齐方式
* color 柱子的填充颜色
* edgecolor 柱子边框的颜色
* linewidth 柱子边框线的宽度

### plt.hist()直方图

* x
* bins 划分组数
* range 统计的范围，默认为(x.min(), x.max())
* density 频数直方图还是频率直方图
* align 柱子与x的对齐方式
* color 颜色
* label 直方图的标签（常配合legend）

### plt.boxplot() 箱线图

* x
* notch 是否凹口展示
* sym 散点的形状
* vert 水平箱线图还是竖向箱线图
* widths 箱子的宽度
* label 箱线图的标签（常配合legend）

### plt.fill\_between() 面积图、填充图

* x
* y
* facecolor 填充颜色
* edgecolor 边缘线条颜色
* linewidth 边缘线宽度
* where 填充的范围
* interpolate interpolate只有在使用了where参数同时两条曲线交叉时才有效, 使用这个参数会把曲线交叉处也填充使得填充的更完整
* alpha 透明度
* label 面积图的标签（常配合legend）

### plt.stackplot() 堆叠面积图、河流图

* x 尺寸为N的一维数组
* y 尺寸为(M,N)的二维数组
* baseline 基线，取值范围为{'zero', 'sym', 'wiggle', 'weighted\_wiggle'}：'zero'：以0为基线，比如绘制简单的堆积面积图；'sym'：以0上下对称，有时被称为主题河流图；'wiggle'：所有序列的斜率平方和最小；'weighted\_wiggle'： 类似于'wiggle'，但是增加各层的大小作为权重。绘制出的图形也被称为流图（streamgraph）。
* 堆叠面积图的标签（常配合legend）

### plt.pie() 饼状图

* **x** (每一块)的比例，如果sum(x) > 1会使用sum(x)归一化
* colors 填充的颜色
* labels (每一块)饼图外侧显示的说明文字
* radius 控制饼图半径

### plt.errorbar() 误差折线图

* x
* y
* yerr y轴的误差范围
* xerr x轴的误差范围
* fmt fmt参数的值和plot方法中指定点的颜色，形状，线条风格的缩写方式相同，如：fmt='co--'
* ecolor 误差线的颜色
* elinewidth 误差线的宽度
* ms 数据点的大小
* mfc 数据点的填充颜色
* mec 数据点边缘颜色
* capthick 误差线的粗细

### plt.axhline() 水平直线

* y
* xmin 取值位于[0,1]之间，取值0时在图的左边界，取值1时在图的右边界
* xmax 取值位于[0,1]之间，0取值时在图的左边界，取值1时在图的右边界
* color 线条颜色
* linestyle 线条类型
* linewidth 线条宽度
* label 线图的标签（常配合legend）

### plt.axvline() 竖向直线

* x
* ymin 取值位于[0,1]之间，取值0时在图的左边界，取值1时在图的右边界
* ymax 取值位于[0,1]之间，0取值时在图的左边界，取值1时在图的右边界
* color 线条颜色
* linestyle 线条类型
* linewidth 线条宽度
* label 线图的标签（常配合legend）

### plt.axhspan() 水平矩形带

* ymin 矩形条带下边界的y坐标
* ymax 矩形条带上边界的y坐标
* facecolor 填充颜色
* alpha 透明度
* edgecolor 边缘线条颜色
* linestyle 线条类型
* linewidth 线条宽度
* label 条带标签（常配合legend）

### plt.axvspan() 竖直矩形带

* xmin 矩形条带左边界的x坐标
* xmax 矩形条带右边界的x坐标
* facecolor 填充颜色
* alpha 透明度
* edgecolor 边缘线条颜色
* linestyle 线条类型
* linewidth 线条宽度
* label 条带标签（常配合legend）

### plt.text() 文本

* x 文本位置的x坐标
* y 文本位置的y坐标
* s 文本
* fontdict 使用字典dict形式定义文本的字体属性，如：

fontdict =   
dict(  
fontsize=12,  
color='r',  
family='monospace',*#字体,可选'serif', 'sans-serif', 'cursive', 'fantasy', 'monospace'*  
weight='bold',*#磅值，可选'light', 'normal', 'medium', 'semibold', 'bold', 'heavy', 'black'*   
)

### plt.annotate() 添加带箭头的文本标注

* s 标注的文本内容
* xy 代表标注点的坐标位置
* xytext 标注的文字的坐标
* arrowprops 箭头参数,参数类型为字典dict，如：

arrowprops=dict(  
arrowstyle = '->',  # 箭头的风格  
connectionstyle = 'arc3, # 连接方式  
rad=0.2' # 弯曲的角度  
)

### 柱状图和直方图的区别！！！

直方图和柱状图的区别是：

1、代表的意义不同，直方图所代表的是数据的具体分布，而柱状图所代表的则是数据的大小。

2、X轴的意思不同，在直方图中X轴所代表的是定量数据，柱状图里面的X轴所代表的则是分类数据。所以直方图里面的柱子是不能随意移动的，而且X轴上的区间需要连续，但是柱状图上的柱子却能够随意的排序移动。

3、柱子的形态不同，直方图上的柱子其宽度是可以不一样的，也就是有宽的，有窄的，但是柱状图上面的柱子宽度是需要一致的。因为柱状图上的宽度并没有数值的含义，但是直方图上面的柱子是代表了区间的长度，所以宽度是根据区间来调整的。