# 13.Matplotlib库

## 13.1. 基本配置

数据可视化是数据分析的一个重要工具，最常用的就是Matplotlib库

**【1】 要不要plt.show()**

ipython中可用魔术方法 %matplotlib inline，这样可以无需plt.show()

pycharm 中必须使用plt.show()

%**matplotlib** inline

**import** **matplotlib.pyplot** **as** **plt**

plt.style.use("seaborn-whitegrid")

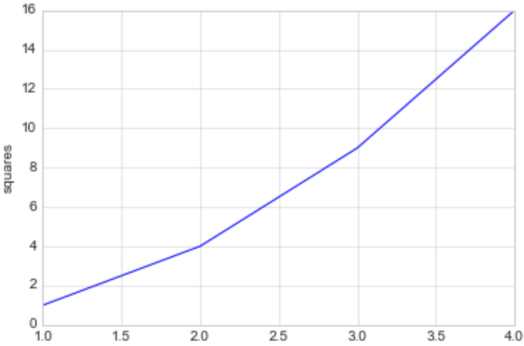
x = [1, 2, 3, 4]

y = [1, 4, 9, 16]

plt.plot(x, y)

plt.ylabel("squares")

*# plt.show()*



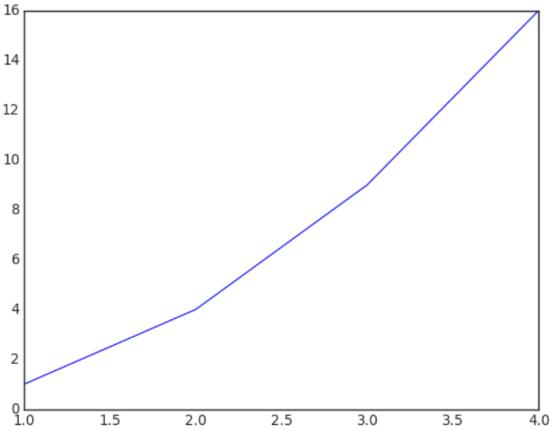
**【2】设置样式**

plt.style.available[:5]

['bmh', 'classic', 'dark\_background', 'fast', 'fivethirtyeight']

**with** plt.style.context("seaborn-white"):

plt.plot(x, y)



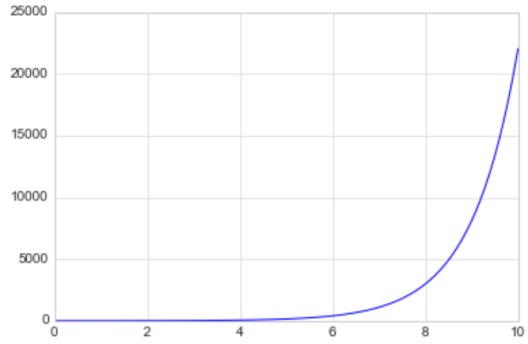
**【3】将图像保存为文件**

**import** **numpy** **as** **np**

x = np.linspace(0, 10 ,100)

plt.plot(x, np.exp(x))

plt.savefig("my\_figure.png")



## 13.2.Matplotlib库

### 13.2.1.折线图

%**matplotlib** inline

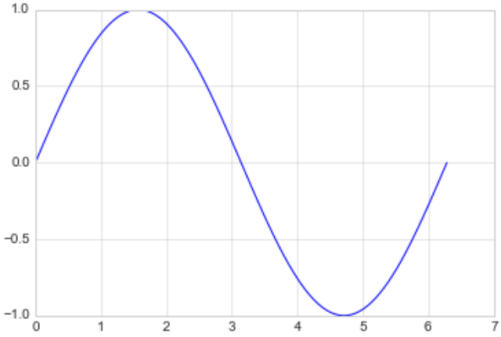
**import** **matplotlib.pyplot** **as** **plt**

plt.style.use("seaborn-whitegrid")

**import** **numpy** **as** **np**

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

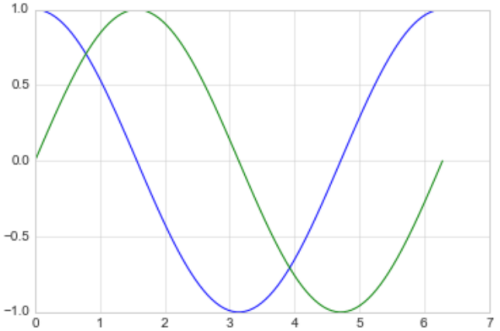


绘制多条曲线

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.cos(x))

plt.plot(x, np.sin(x))



**【1】调整线条颜色和风格**

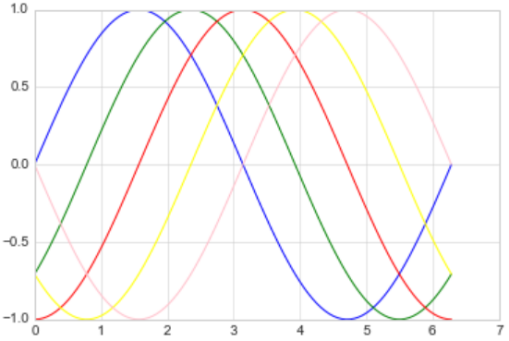
调整线条颜色

offsets = np.linspace(0, np.pi, 5)

colors = ["blue", "g", "r", "yellow", "pink"]

**for** offset, color **in** zip(offsets, colors):

plt.plot(x, np.sin(x-offset), color=color) *# color可缩写为c*



调整线条风格

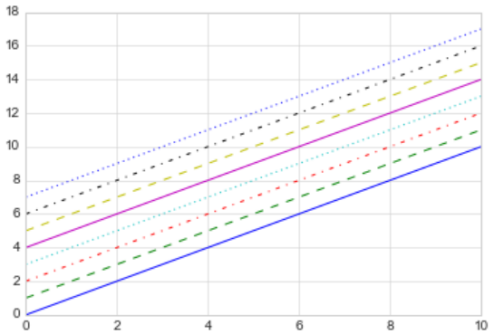
x = np.linspace(0, 10, 11)

offsets = list(range(8))

linestyles = ["solid", "dashed", "dashdot", "dotted", "-", "--", "-.", ":"]

**for** offset, linestyle **in** zip(offsets, linestyles):

plt.plot(x, x+offset, linestyle=linestyle) *# linestyle可简写为ls*



调整线宽

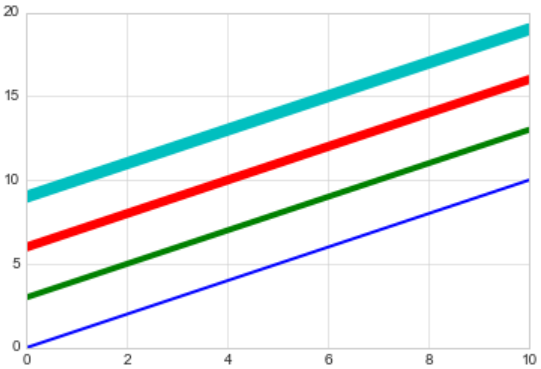
x = np.linspace(0, 10, 11)

offsets = list(range(0, 12, 3))

linewidths = (i\*2 **for** i **in** range(1,5))

**for** offset, linewidth **in** zip(offsets, linewidths):

plt.plot(x, x+offset, linewidth=linewidth) *# linewidth可简写为lw*



调整数据点标记

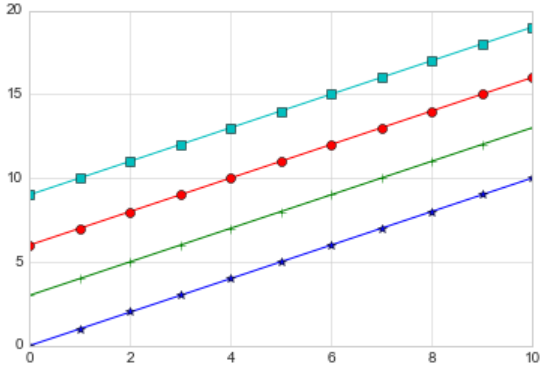
x = np.linspace(0, 10, 11)

offsets = list(range(0, 12, 3))

markers = ["\*", "+", "o", "s"]

**for** offset, marker **in** zip(offsets, markers):

plt.plot(x, x+offset, marker=marker)



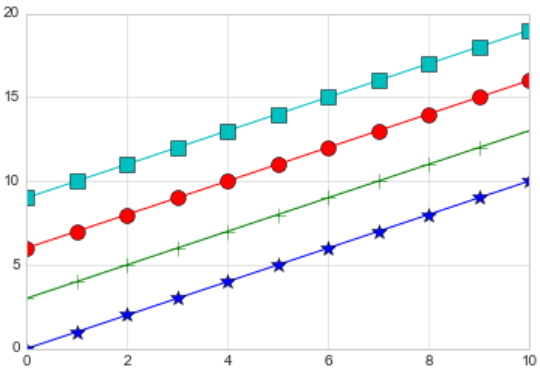
x = np.linspace(0, 10, 11)

offsets = list(range(0, 12, 3))

markers = ["\*", "+", "o", "s"]

**for** offset, marker **in** zip(offsets, markers):

plt.plot(x, x+offset, marker=marker, markersize=10) *# markersize可简写为ms*



颜色跟风格设置的简写

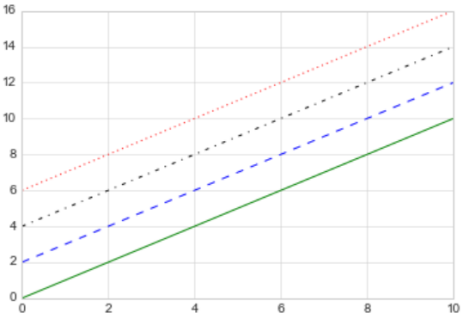
x = np.linspace(0, 10, 11)

offsets = list(range(0, 8, 2))

color\_linestyles = ["g-", "b--", "k-.", "r:"]

**for** offset, color\_linestyle **in** zip(offsets, color\_linestyles):

plt.plot(x, x+offset, color\_linestyle)



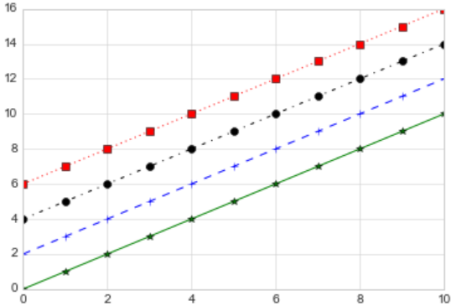
x = np.linspace(0, 10, 11)

offsets = list(range(0, 8, 2))

color\_marker\_linestyles = ["g\*-", "b+--", "ko-.", "rs:"]

**for** offset, color\_marker\_linestyle **in** zip(offsets, color\_marker\_linestyles):

plt.plot(x, x+offset, color\_marker\_linestyle)



其他用法及颜色缩写、数据点标记缩写等请查看官方文档，如下：

[https://matplotlib.org/api/\_as\_gen/matplotlib.pyplot.plot.html#matplotlib.pyplot.plot](https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html" \l "matplotlib.pyplot.plot)

**【2】调整坐标轴**

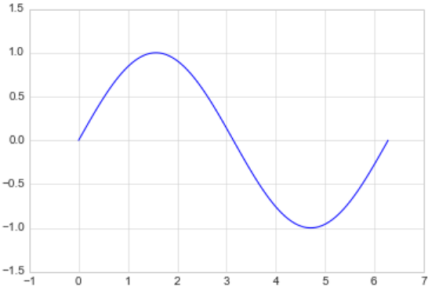
xlim, ylim

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

plt.xlim(-1, 7)

plt.ylim(-1.5, 1.5)

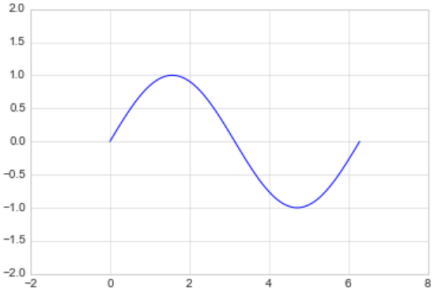


axis

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

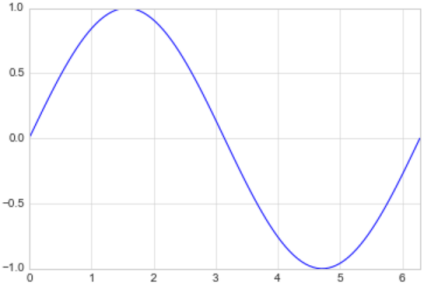
plt.axis([-2, 8, -2, 2])



x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

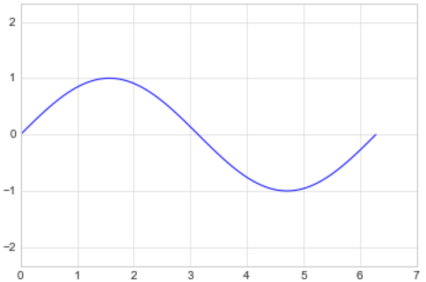
plt.axis("tight") *# tight表示紧凑的图像*



x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

plt.axis("equal") *# equal表示扁平的图像*



?plt.axis *# 查看还有哪些样式*

'on' Turn on axis lines and labels.

...

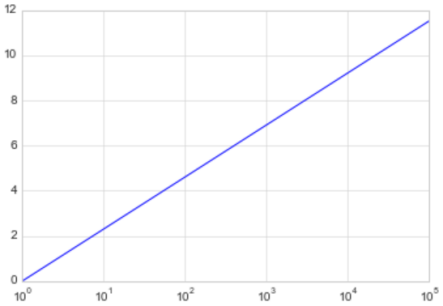
'square' Square plot; similar to 'scaled', but initially forcing

对数坐标

x = np.logspace(0, 5, 100)

plt.plot(x, np.log(x))

plt.xscale("log")

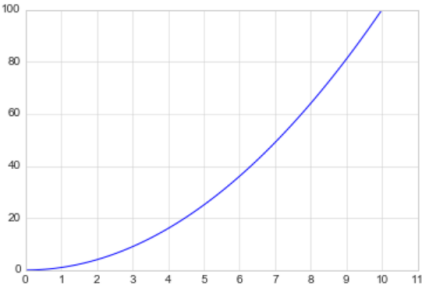


调整坐标轴刻度

x = np.linspace(0, 10, 100)

plt.plot(x, x\*\*2)

plt.xticks(np.arange(0, 12, step=1))

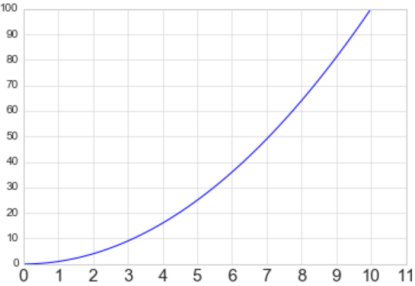


x = np.linspace(0, 10, 100)

plt.plot(x, x\*\*2)

plt.xticks(np.arange(0, 12, step=1), fontsize=15)

plt.yticks(np.arange(0, 110, step=10))

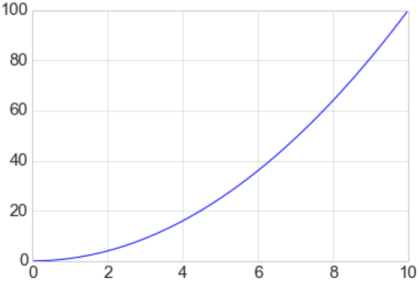


调整刻度样式

x = np.linspace(0, 10, 100)

plt.plot(x, x\*\*2)

plt.tick\_params(axis="both", labelsize=15)



**【3】设置图形标签**

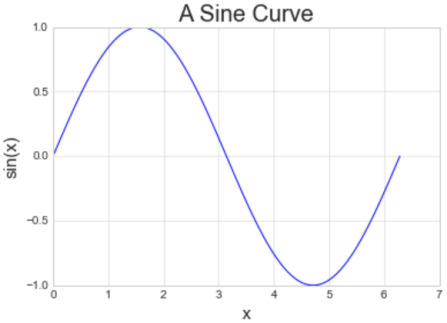
x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x))

plt.title("A Sine Curve", fontsize=20)

plt.xlabel("x", fontsize=15)

plt.ylabel("sin(x)", fontsize=15)



**【4】设置图例**

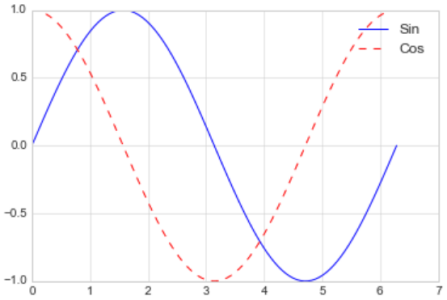
默认

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x), "b-", label="Sin")

plt.plot(x, np.cos(x), "r--", label="Cos")

plt.legend()



修饰图例

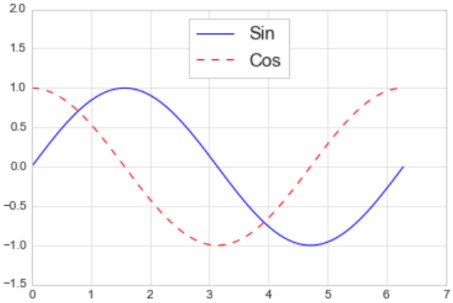
x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x), "b-", label="Sin")

plt.plot(x, np.cos(x), "r--", label="Cos")

plt.ylim(-1.5, 2)

plt.legend(loc="upper center", frameon=**True**, fontsize=15)



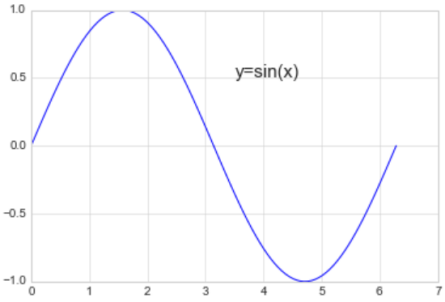
**【5】添加文字和箭头**

添加文字

x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x), "b-")

plt.text(3.5, 0.5, "y=sin(x)", fontsize=15)



添加箭头

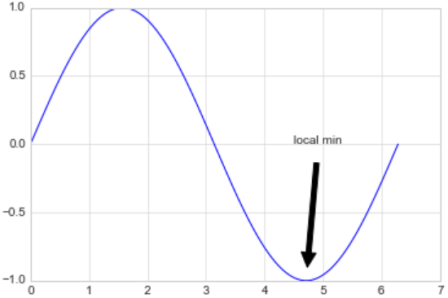
x = np.linspace(0, 2\*np.pi, 100)

plt.plot(x, np.sin(x), "b-")

plt.annotate('local min', xy=(1.5\*np.pi, -1), xytext=(4.5, 0),

arrowprops=dict(facecolor='black', shrink=0.1),

)

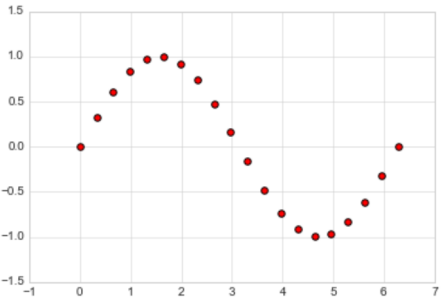


### 13.2.2.散点图

**【1】简单散点图**

x = np.linspace(0, 2\*np.pi, 20)

plt.scatter(x, np.sin(x), marker="o", s=30, c="r") *# s 大小 c 颜色*



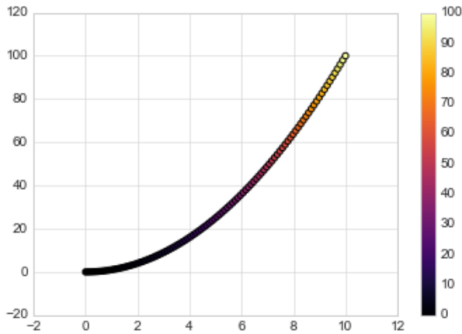
**【2】颜色配置**

x = np.linspace(0, 10, 100)

y = x\*\*2

plt.scatter(x, y, c=y, cmap="inferno") *# c=y表示颜色随着y的大小映射*

plt.colorbar()



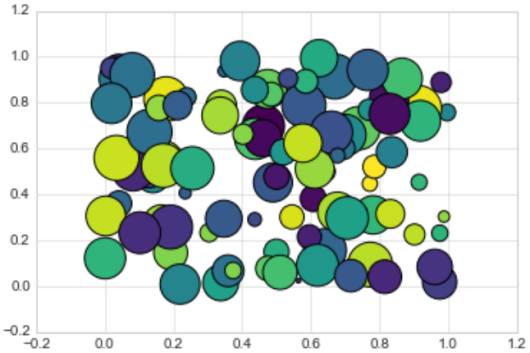
颜色配置参考官方文档

<https://matplotlib.org/examples/color/colormaps_reference.html>

**【3】根据数据控制点的大小**

x, y, colors, size = (np.random.rand(100) **for** i **in** range(4))

plt.scatter(x, y, c=colors, s=1000\*size, cmap="viridis")

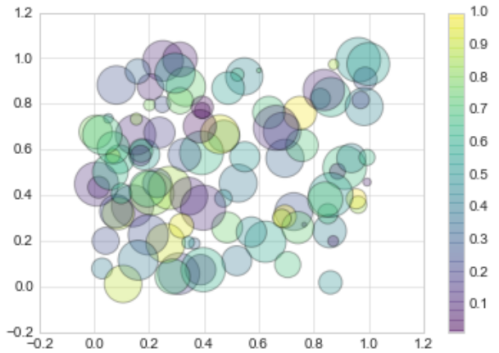


**【4】透明度**

x, y, colors, size = (np.random.rand(100) **for** i **in** range(4))

plt.scatter(x, y, c=colors, s=1000\*size, cmap="viridis", alpha=0.3)

plt.colorbar()



【例】随机漫步

**from** **random** **import** choice

**class** **RandomWalk**():

*"""一个生产随机漫步的类"""*

**def** \_\_init\_\_(self, num\_points=5000):

self.num\_points = num\_points

self.x\_values = [0]

self.y\_values = [0]

**def** fill\_walk(self):

**while** len(self.x\_values) < self.num\_points:

x\_direction = choice([1, -1])

x\_distance = choice([0, 1, 2, 3, 4])

x\_step = x\_direction \* x\_distance

y\_direction = choice([1, -1])

y\_distance = choice([0, 1, 2, 3, 4])

y\_step = y\_direction \* y\_distance

**if** x\_step == 0 **or** y\_step == 0:

**continue**

next\_x = self.x\_values[-1] + x\_step

next\_y = self.y\_values[-1] + y\_step

self.x\_values.append(next\_x)

self.y\_values.append(next\_y)

rw = RandomWalk(10000)

rw.fill\_walk()

point\_numbers = list(range(rw.num\_points))

plt.figure(figsize=(12, 6))

plt.scatter(rw.x\_values, rw.y\_values, c=point\_numbers, cmap="inferno", s=1)

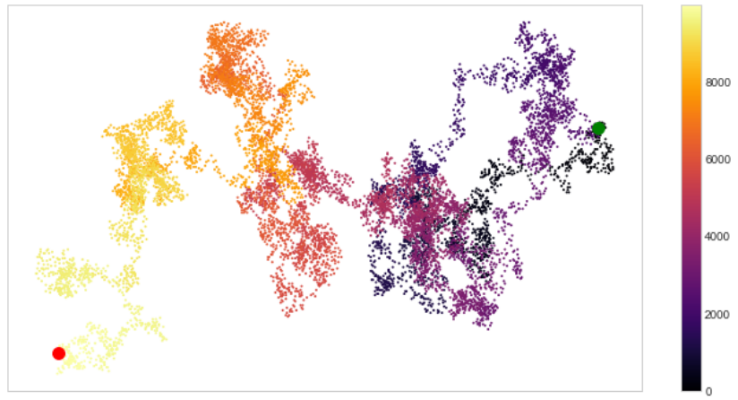
plt.colorbar()

plt.scatter(0, 0, c="green", s=100)

plt.scatter(rw.x\_values[-1], rw.y\_values[-1], c="red", s=100)

plt.xticks([])

plt.yticks([])



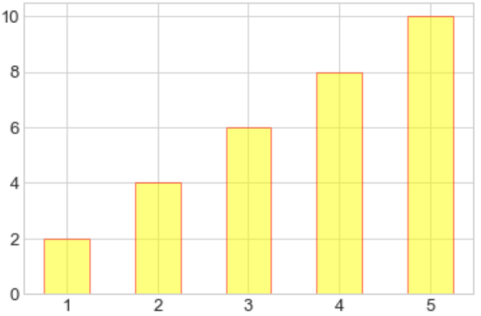
### 13.2.3.柱形图

**【1】简单柱形图**

x = np.arange(1, 6)

plt.bar(x, 2\*x, align="center", width=0.5, alpha=0.5, color='yellow', edgecolor='red')

plt.tick\_params(axis="both", labelsize=13)

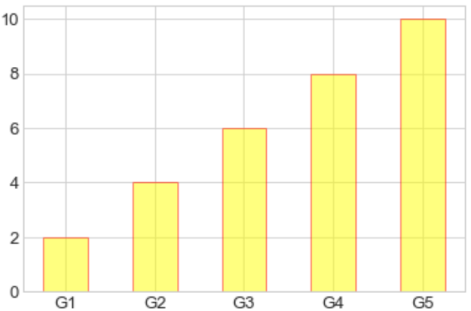


x = np.arange(1, 6)

plt.bar(x, 2\*x, align="center", width=0.5, alpha=0.5, color='yellow', edgecolor='red')

plt.xticks(x, ('G1', 'G2', 'G3', 'G4', 'G5'))

plt.tick\_params(axis="both", labelsize=13)

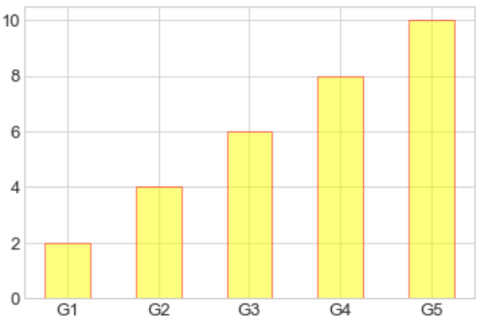


x = ('G1', 'G2', 'G3', 'G4', 'G5')

y = 2 \* np.arange(1, 6)

plt.bar(x, y, align="center", width=0.5, alpha=0.5, color='yellow', edgecolor='red')

plt.tick\_params(axis="both", labelsize=13)



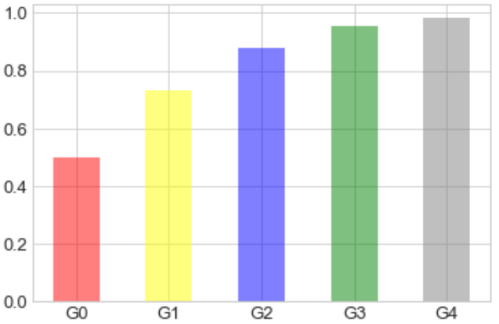
x = ["G"+str(i) **for** i **in** range(5)]

y = 1/(1+np.exp(-np.arange(5)))

colors = ['red', 'yellow', 'blue', 'green', 'gray']

plt.bar(x, y, align="center", width=0.5, alpha=0.5, color=colors)

plt.tick\_params(axis="both", labelsize=13)



**【2】累加柱形图**

x = np.arange(5)

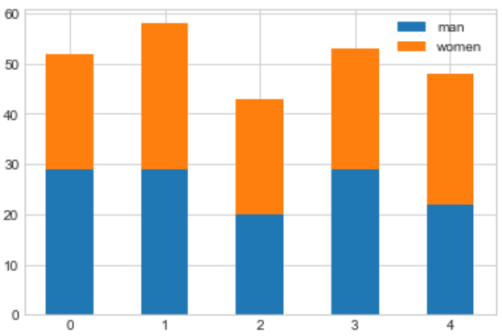
y1 = np.random.randint(20, 30, size=5)

y2 = np.random.randint(20, 30, size=5)

plt.bar(x, y1, width=0.5, label="man")

plt.bar(x, y2, width=0.5, bottom=y1, label="women")

plt.legend()



**【3】并列柱形图**

x = np.arange(15)

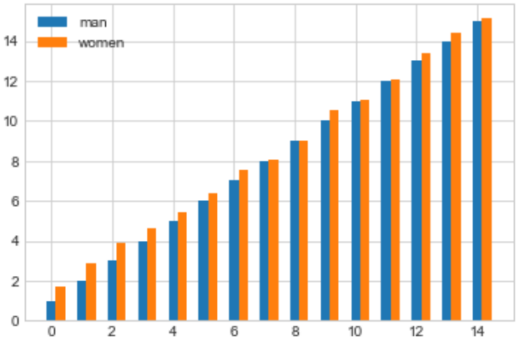
y1 = x+1

y2 = y1+np.random.random(15)

plt.bar(x, y1, width=0.3, label="man")

plt.bar(x+0.3, y2, width=0.3, label="women")

plt.legend()



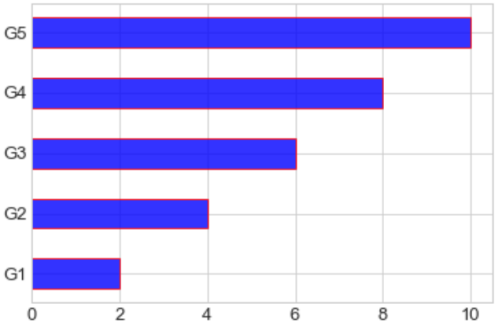
**【4】横向柱形图**

x = ['G1', 'G2', 'G3', 'G4', 'G5']

y = 2 \* np.arange(1, 6)

plt.barh(x, y, align="center", height=0.5, alpha=0.8, color="blue", edgecolor="red")

plt.tick\_params(axis="both", labelsize=13)



### 13.2.4.多子图

**【1】简单多子图**

**def** f(t):

**return** np.exp(-t) \* np.cos(2\*np.pi\*t)

t1 = np.arange(0.0, 5.0, 0.1)

t2 = np.arange(0.0, 5.0, 0.02)

plt.subplot(211)

plt.plot(t1, f(t1), "bo-", markerfacecolor="r", markersize=5)

plt.title("A tale of 2 subplots")

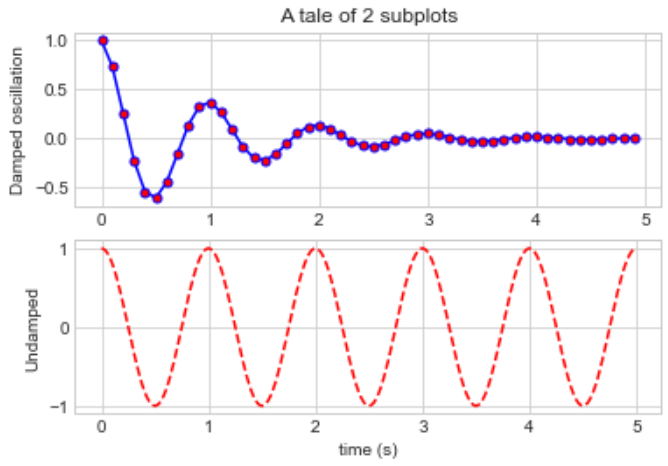
plt.ylabel("Damped oscillation")

plt.subplot(212)

plt.plot(t2, np.cos(2\*np.pi\*t2), "r--")

plt.xlabel("time (s)")

plt.ylabel("Undamped")



**【2】多行多列子图**

x = np.random.random(10)

y = np.random.random(10)

plt.subplots\_adjust(hspace=0.5, wspace=0.3)

plt.subplot(321)

plt.scatter(x, y, s=80, c="b", marker=">")

plt.subplot(322)

plt.scatter(x, y, s=80, c="g", marker="\*")

plt.subplot(323)

plt.scatter(x, y, s=80, c="r", marker="s")

plt.subplot(324)

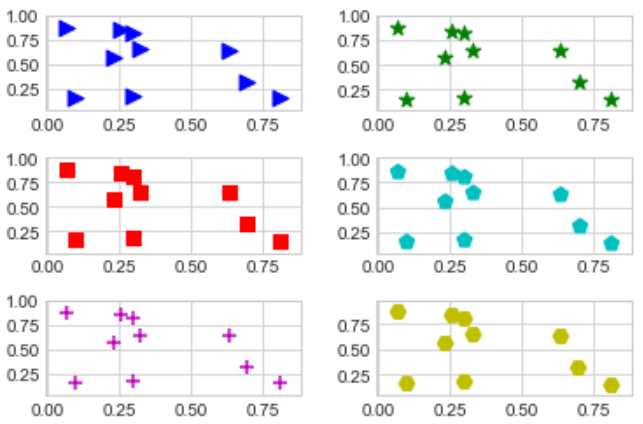
plt.scatter(x, y, s=80, c="c", marker="p")

plt.subplot(325)

plt.scatter(x, y, s=80, c="m", marker="+")

plt.subplot(326)

plt.scatter(x, y, s=80, c="y", marker="H")



**【3】不规则多子图**

**def** f(x):

**return** np.exp(-x) \* np.cos(2\*np.pi\*x)

x = np.arange(0.0, 3.0, 0.01)

grid = plt.GridSpec(2, 3, wspace=0.4, hspace=0.3)

plt.subplot(grid[0, 0])

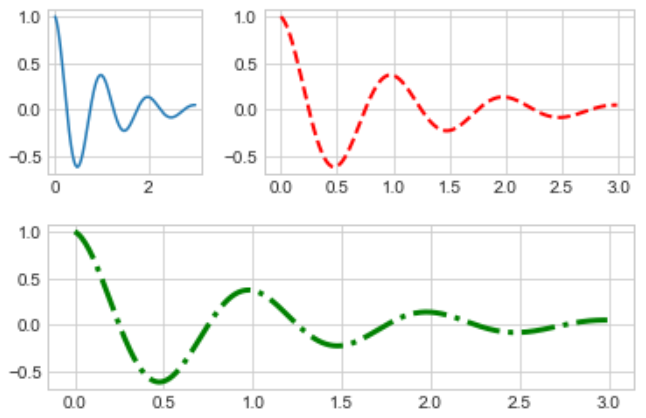
plt.plot(x, f(x))

plt.subplot(grid[0, 1:])

plt.plot(x, f(x), "r--", lw=2)

plt.subplot(grid[1, :])

plt.plot(x, f(x), "g-.", lw=3)



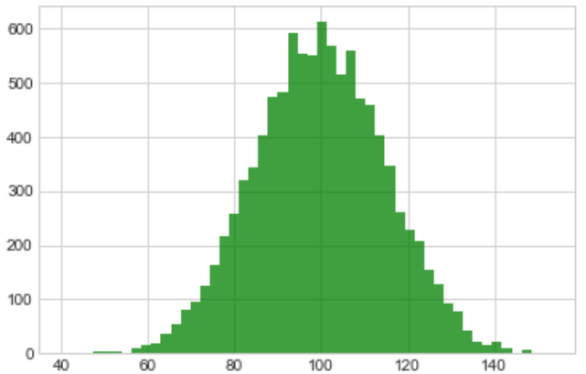
### 13.2.5.直方图

**【1】普通频次直方图**

mu, sigma = 100, 15

x = mu + sigma \* np.random.randn(10000)

plt.hist(x, bins=50, facecolor='g', alpha=0.75)



**【2】概率密度**

mu, sigma = 100, 15

x = mu + sigma \* np.random.randn(10000)

plt.hist(x, 50, density=**True**, color="r")

plt.xlabel('Smarts')

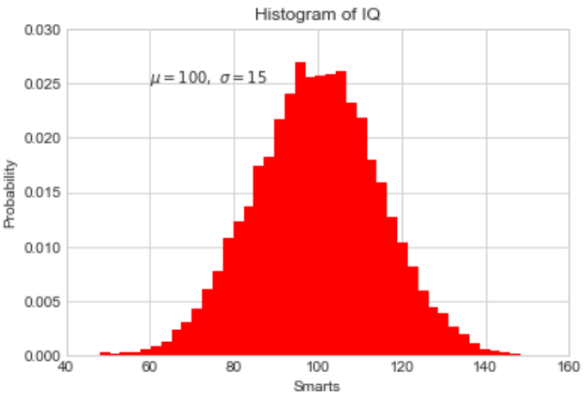
plt.ylabel('Probability')

plt.title('Histogram of IQ')

plt.text(60, .025, r'$\mu=100,\ \sigma=15$')

plt.xlim(40, 160)

plt.ylim(0, 0.03)



mu, sigma = 100, 15

x = mu + sigma \* np.random.randn(10000)

plt.hist(x, bins=50, density=**True**, color="r", histtype='step')

plt.xlabel('Smarts')

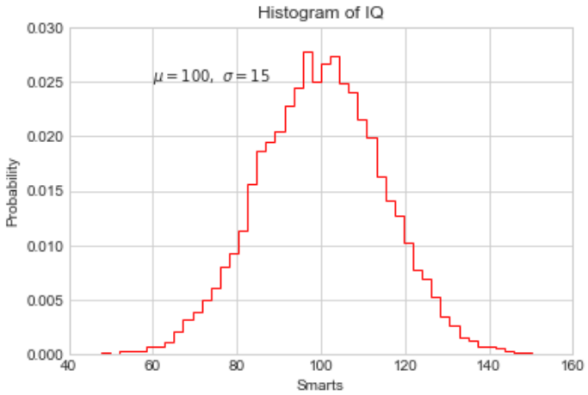
plt.ylabel('Probability')

plt.title('Histogram of IQ')

plt.text(60, .025, r'$\mu=100,\ \sigma=15$')

plt.xlim(40, 160)

plt.ylim(0, 0.03)



**from** **scipy.stats** **import** norm

mu, sigma = 100, 15

x = mu + sigma \* np.random.randn(10000)

\_, bins, \_\_ = plt.hist(x, 50, density=**True**)

y = norm.pdf(bins, mu, sigma)

plt.plot(bins, y, 'r--', lw=3)

plt.xlabel('Smarts')

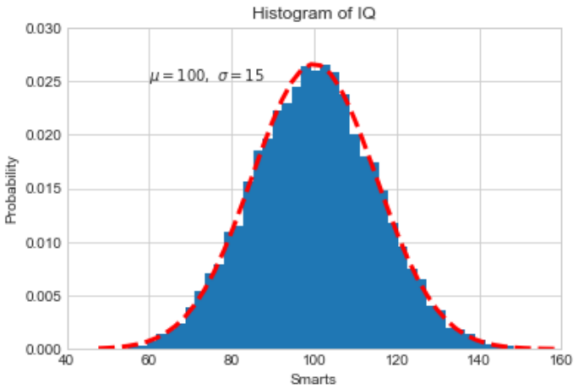
plt.ylabel('Probability')

plt.title('Histogram of IQ')

plt.text(60, .025, r'$\mu=100,\ \sigma=15$')

plt.xlim(40, 160)

plt.ylim(0, 0.03)



**【3】累计概率分布**

mu, sigma = 100, 15

x = mu + sigma \* np.random.randn(10000)

plt.hist(x, 50, density=**True**, cumulative=**True**, color="r")

plt.xlabel('Smarts')

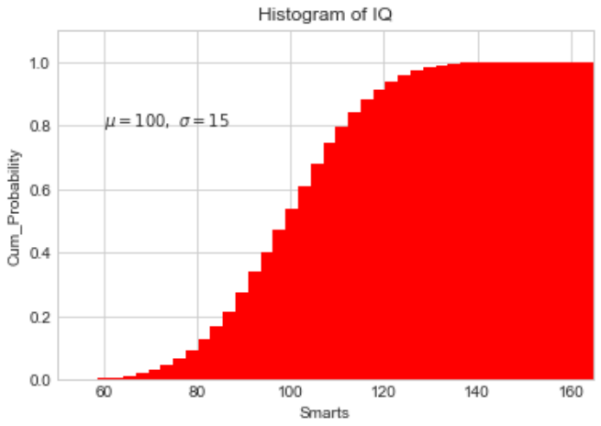
plt.ylabel('Cum\_Probability')

plt.title('Histogram of IQ')

plt.text(60, 0.8, r'$\mu=100,\ \sigma=15$')

plt.xlim(50, 165)

plt.ylim(0, 1.1)



【例】模拟投两个骰子

**class** **Die**():

"模拟一个骰子的类"

**def** \_\_init\_\_(self, num\_sides=6):

self.num\_sides = num\_sides

**def** roll(self):

**return** np.random.randint(1, self.num\_sides+1)

重复投一个骰子

die = Die()

results = []

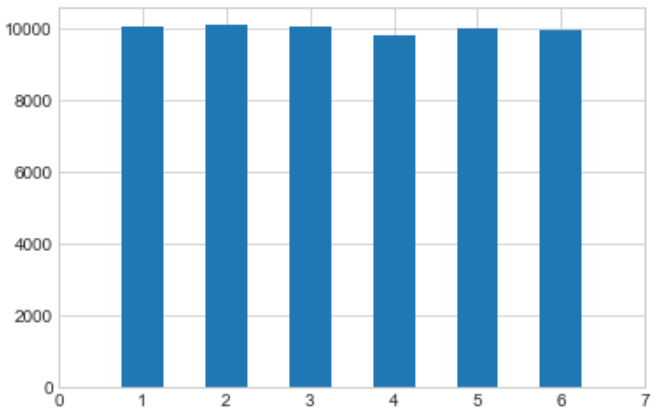
**for** i **in** range(60000):

result = die.roll()

results.append(result)

plt.hist(results, bins=6, range=(0.75, 6.75), align="mid", width=0.5)

plt.xlim(0 ,7)



重复投两个骰子

die1 = Die()

die2 = Die()

results = []

**for** i **in** range(60000):

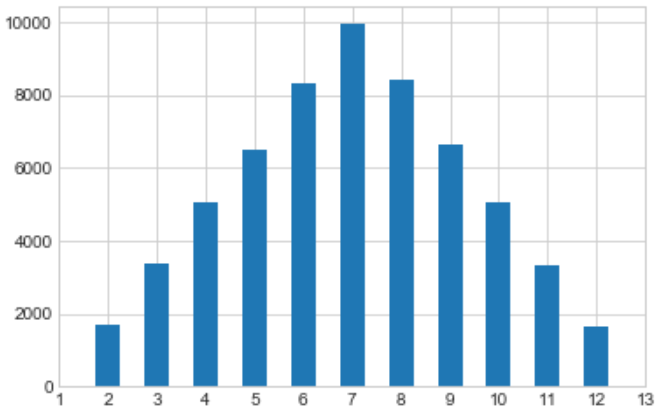
result = die1.roll()+die2.roll()

results.append(result)

plt.hist(results, bins=11, range=(1.75, 12.75), align="mid", width=0.5)

plt.xlim(1 ,13)

plt.xticks(np.arange(1, 14))



### 13.2.6.误差图

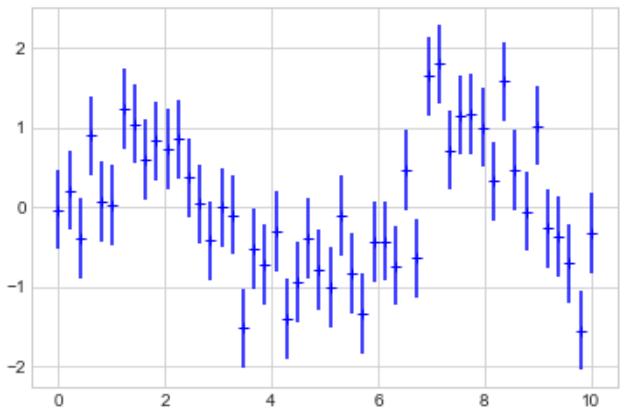
**【1】基本误差图**

x = np.linspace(0, 10 ,50)

dy = 0.5

y = np.sin(x) + dy\*np.random.randn(50)

plt.errorbar(x, y , yerr=dy, fmt="+b")



**【2】柱形图误差图**

menMeans = (20, 35, 30, 35, 27)

womenMeans = (25, 32, 34, 20, 25)

menStd = (2, 3, 4, 1, 2)

womenStd = (3, 5, 2, 3, 3)

ind = ['G1', 'G2', 'G3', 'G4', 'G5']

width = 0.35

p1 = plt.bar(ind, menMeans, width=width, label="Men", yerr=menStd)

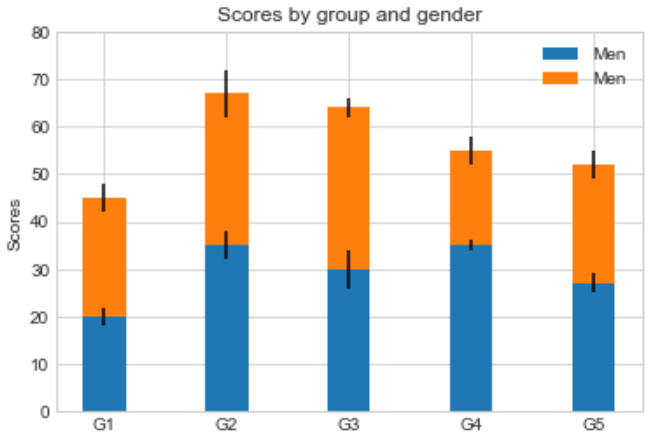
p2 = plt.bar(ind, womenMeans, width=width, bottom=menMeans, label="Men", yerr=womenStd)

plt.ylabel('Scores')

plt.title('Scores by group and gender')

plt.yticks(np.arange(0, 81, 10))

plt.legend()



### 13.2.7.面向对象的风格简介

**【例1】 普通图**

x = np.linspace(0, 5, 10)

y = x \*\* 2

fig = plt.figure(figsize=(8,4), dpi=80) *# 图像*

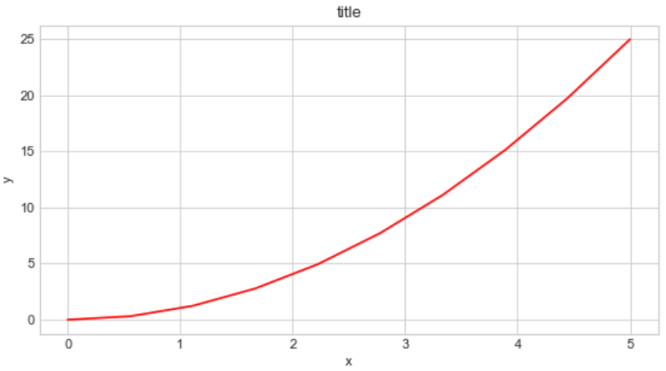
axes = fig.add\_axes([0.1, 0.1, 0.8, 0.8]) *# 轴 left, bottom, width, height (range 0 to 1)*

axes.plot(x, y, 'r')

axes.set\_xlabel('x')

axes.set\_ylabel('y')

axes.set\_title('title')



**【2】画中画**

x = np.linspace(0, 5, 10)

y = x \*\* 2

fig = plt.figure()

ax1 = fig.add\_axes([0.1, 0.1, 0.8, 0.8])

ax2 = fig.add\_axes([0.2, 0.5, 0.4, 0.3])

ax1.plot(x, y, 'r')

ax1.set\_xlabel('x')

ax1.set\_ylabel('y')

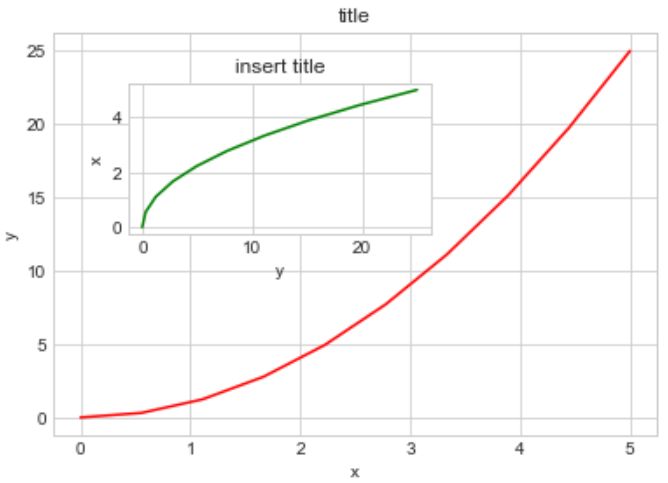
ax1.set\_title('title')

ax2.plot(y, x, 'g')

ax2.set\_xlabel('y')

ax2.set\_ylabel('x')

ax2.set\_title('insert title')



**【3】 多子图**

**def** f(t):

**return** np.exp(-t) \* np.cos(2\*np.pi\*t)

t1 = np.arange(0.0, 3.0, 0.01)

fig= plt.figure()

fig.subplots\_adjust(hspace=0.4, wspace=0.4)

ax1 = plt.subplot(2, 2, 1)

ax1.plot(t1, f(t1))

ax1.set\_title("Upper left")

ax2 = plt.subplot(2, 2, 2)

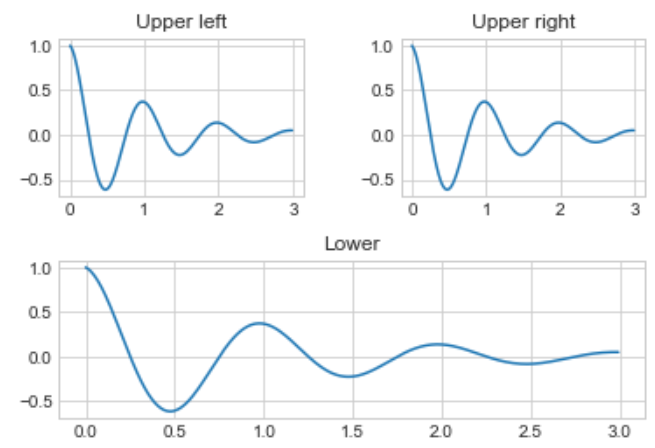
ax2.plot(t1, f(t1))

ax2.set\_title("Upper right")

ax3 = plt.subplot(2, 1, 2)

ax3.plot(t1, f(t1))

ax3.set\_title("Lower")



### 13.2.8.三维图形简介

**【1】三维数据点与线**

**from** **mpl\_toolkits** **import** mplot3d

ax = plt.axes(projection="3d")

zline = np.linspace(0, 15, 1000)

xline = np.sin(zline)

yline = np.cos(zline)

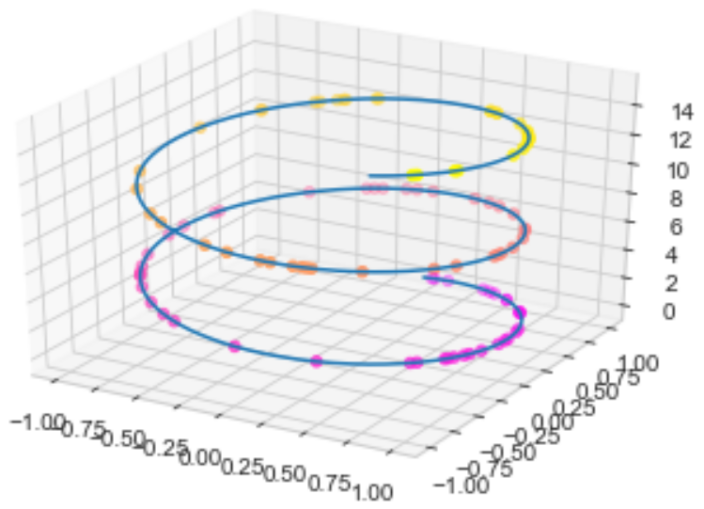
ax.plot3D(xline, yline ,zline)

zdata = 15\*np.random.random(100)

xdata = np.sin(zdata)

ydata = np.cos(zdata)

ax.scatter3D(xdata, ydata ,zdata, c=zdata, cmap="spring")



**【2】三维数据曲面图**

**def** f(x, y):

**return** np.sin(np.sqrt(x\*\*2 + y\*\*2))

x = np.linspace(-6, 6, 30)

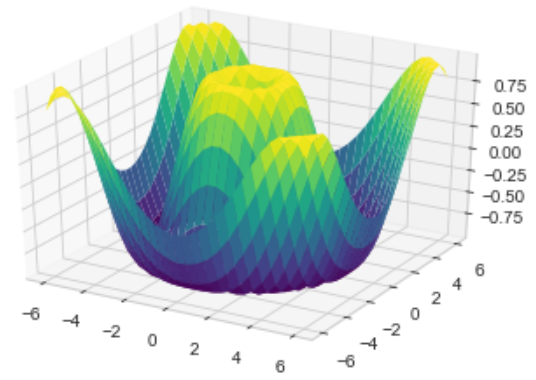
y = np.linspace(-6, 6, 30)

X, Y = np.meshgrid(x, y) *# 对x和y网格化*

Z = f(X, Y)

ax = plt.axes(projection="3d")

ax.plot\_surface(X, Y, Z, cmap="viridis")



## 13.3.Seaborn库

**【1】Seaborn 与 Matplotlib**

Seaborn 是一个基于 matplotlib 且数据结构与 pandas 统一的统计图制作库

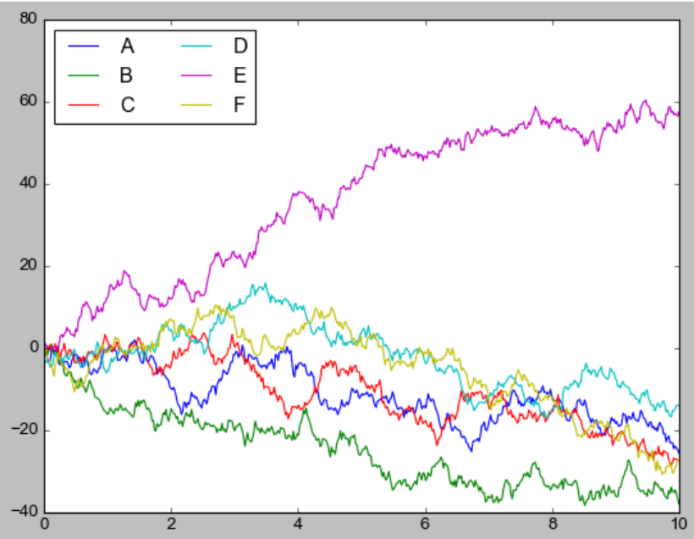
x = np.linspace(0, 10, 500)

y = np.cumsum(np.random.randn(500, 6), axis=0) *# 500行6列累计求和*

**with** plt.style.context("classic"):

plt.plot(x, y)

plt.legend("ABCDEF", ncol=2, loc="upper left")



**import** **seaborn** **as** **sns**

x = np.linspace(0, 10, 500)

y = np.cumsum(np.random.randn(500, 6), axis=0)

sns.set()

plt.figure(figsize=(10, 6))

plt.plot(x, y)

plt.legend("ABCDEF", ncol=2, loc="upper left")



**【2】柱形图的对比**

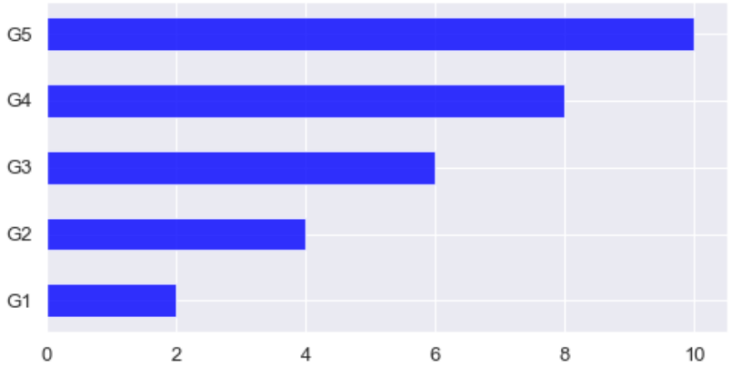
x = ['G1', 'G2', 'G3', 'G4', 'G5']

y = 2 \* np.arange(1, 6)

plt.figure(figsize=(8, 4))

plt.barh(x, y, align="center", height=0.5, alpha=0.8, color="blue")

plt.tick\_params(axis="both", labelsize=13)



**import** **seaborn** **as** **sns**

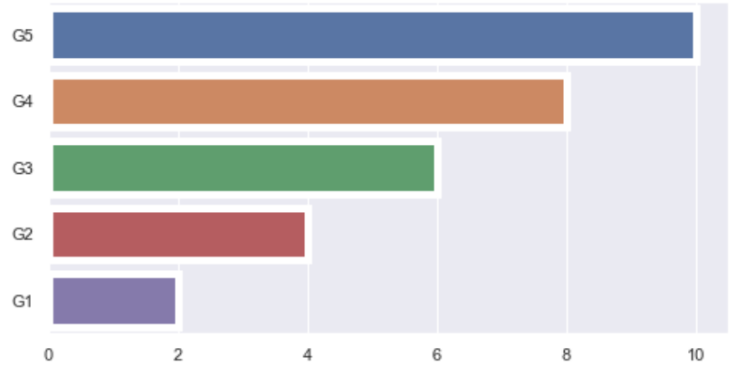
plt.figure(figsize=(8, 4))

x = ['G5', 'G4', 'G3', 'G2', 'G1']

y = 2 \* np.arange(5, 0, -1)

*#sns.barplot(y, x)*

sns.barplot(y, x, linewidth=5)



sns.barplot? *# 查看sns.barplot用法*

【3】以鸢尾花数据集为例

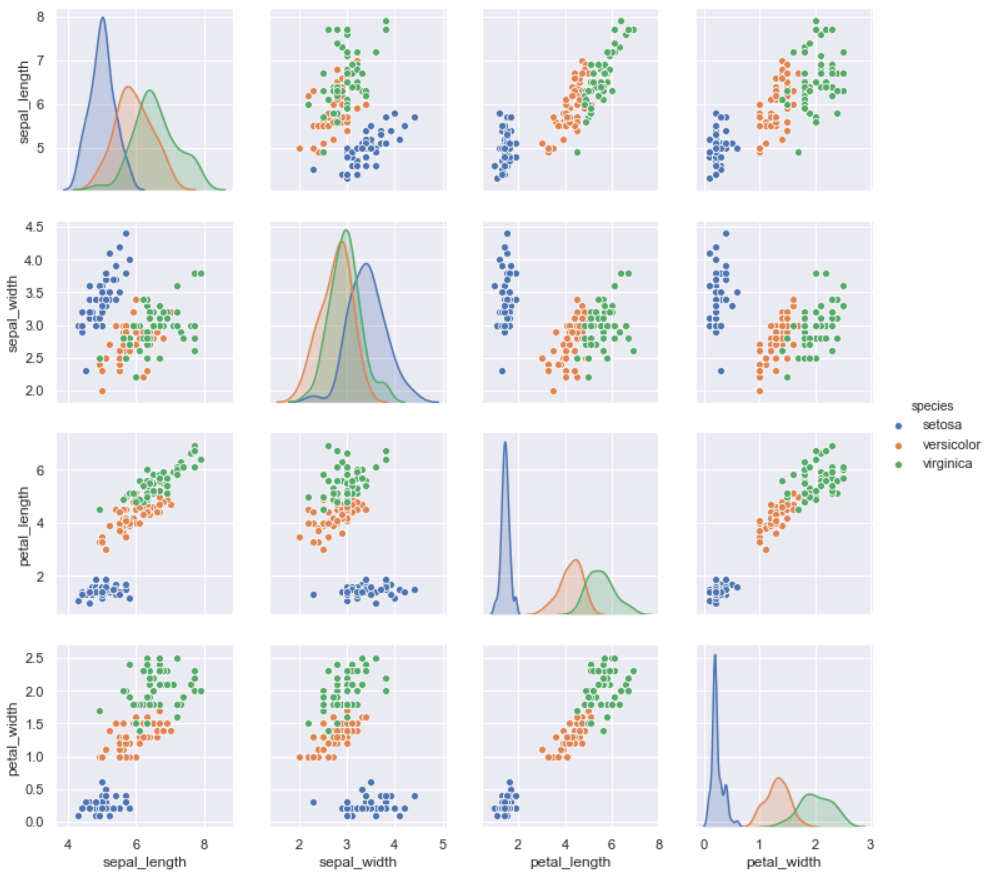
*#iris = sns.load\_dataset("iris")*

iris = pd.read\_csv("data/iris.csv")

iris.head()

|  | **sepal\_length** | **sepal\_width** | **petal\_length** | **petal\_width** | **species** |
| --- | --- | --- | --- | --- | --- |
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | setosa |

sns.pairplot(data=iris, hue="species")



## 13.4.Pandas 中的绘图函数概览

**【1】线形图**

**import** **pandas** **as** **pd**

df = pd.DataFrame(np.random.randn(1000, 4).cumsum(axis=0),

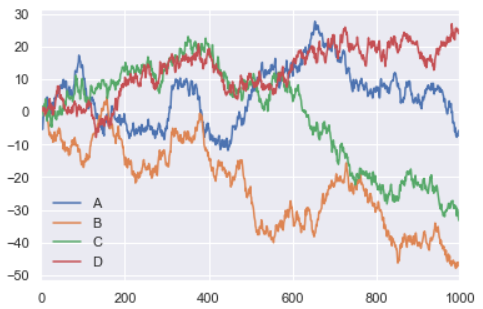
columns=list("ABCD"),

index=np.arange(1000))

df.head()

|  | **A** | **B** | **C** | **D** |
| --- | --- | --- | --- | --- |
| **0** | -1.311443 | 0.970917 | -1.635011 | -0.204779 |
| **1** | -1.618502 | 0.810056 | -1.119246 | 1.239689 |
| **2** | -3.558787 | 1.431716 | -0.816201 | 1.155611 |
| **3** | -5.377557 | -0.312744 | 0.650922 | 0.352176 |
| **4** | -3.917045 | 1.181097 | 1.572406 | 0.965921 |

df.plot()



df = pd.DataFrame()

df.plot?

**【2】柱形图**

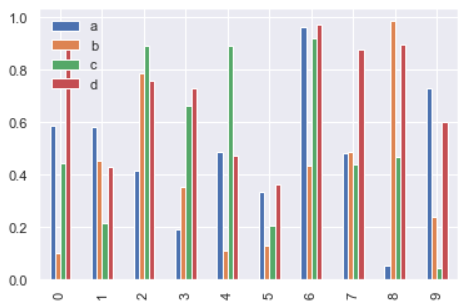
df2 = pd.DataFrame(np.random.rand(10, 4), columns=['a', 'b', 'c', 'd'])

df2

|  | **a** | **b** | **c** | **d** |
| --- | --- | --- | --- | --- |
| **0** | 0.587600 | 0.098736 | 0.444757 | 0.877475 |
| **1** | 0.580062 | 0.451519 | 0.212318 | 0.429673 |
| **...** | ... | ... | ... | ... |
| **9** | 0.730905 | 0.237166 | 0.043195 | 0.600445 |

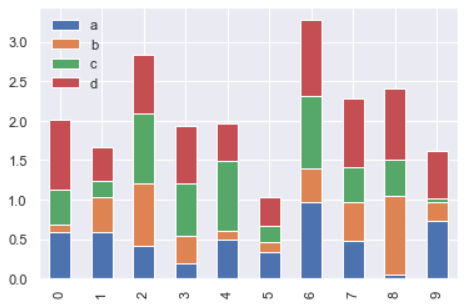
多组数据竖图

df2.plot.bar()



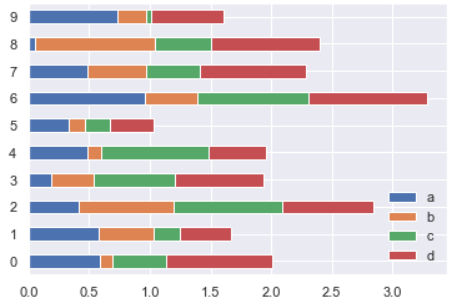
多组数据累加竖图

df2.plot.bar(stacked=**True**)



多组数据累加横图

df2.plot.barh(stacked=**True**)



**【3】直方图和密度图**

df4 = pd.DataFrame({"A": np.random.randn(1000) - 3, "B": np.random.randn(1000),

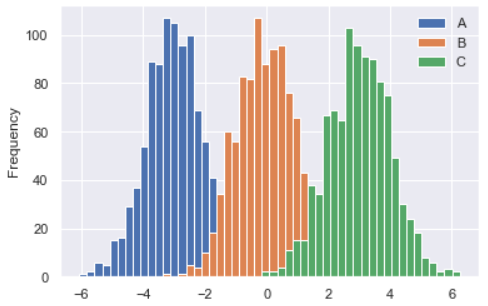
"C": np.random.randn(1000) + 3})

df4.head()

|  | **A** | **B** | **C** |
| --- | --- | --- | --- |
| **0** | -4.250424 | 1.043268 | 1.356106 |
| **1** | -2.393362 | -0.891620 | 3.787906 |
| **2** | -4.411225 | 0.436381 | 1.242749 |
| **3** | -3.465659 | -0.845966 | 1.540347 |
| **4** | -3.606850 | 1.643404 | 3.689431 |

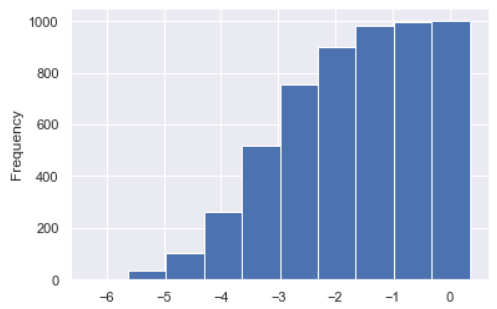
普通直方图

df4.plot.hist(bins=50)



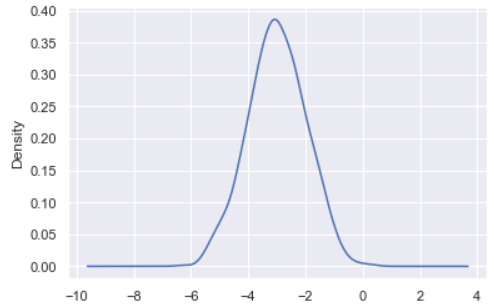
累加直方图

df4['A'].plot.hist(cumulative=**True**)



概率密度图

df4['A'].plot(kind="kde")



差分

df = pd.DataFrame(np.random.randn(1000, 4).cumsum(axis=0),

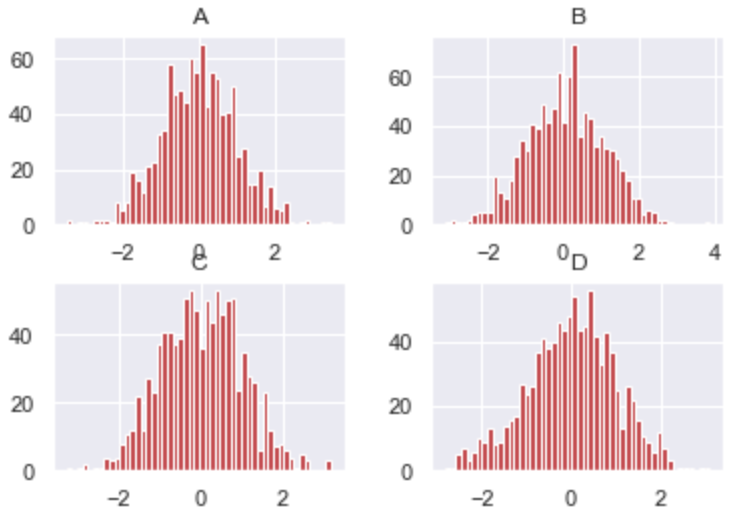
columns=list("ABCD"),

index=np.arange(1000))

df.head()

|  | **A** | **B** | **C** | **D** |
| --- | --- | --- | --- | --- |
| **0** | -0.277843 | -0.310656 | -0.782999 | -0.049032 |
| **1** | 0.644248 | -0.505115 | -0.363842 | 0.399116 |
| **2** | -0.614141 | -1.227740 | -0.787415 | -0.117485 |
| **3** | -0.055964 | -2.376631 | -0.814320 | -0.716179 |
| **4** | 0.058613 | -2.355537 | -2.174291 | 0.351918 |

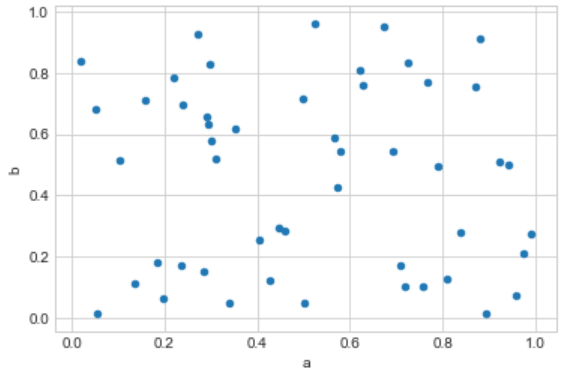
df.diff().hist(bins=50, color="r")



**【4】散点图**

df = pd.DataFrame(np.random.rand(50, 2), columns=['a', 'b'])

df.plot.scatter(x='a', y='b')



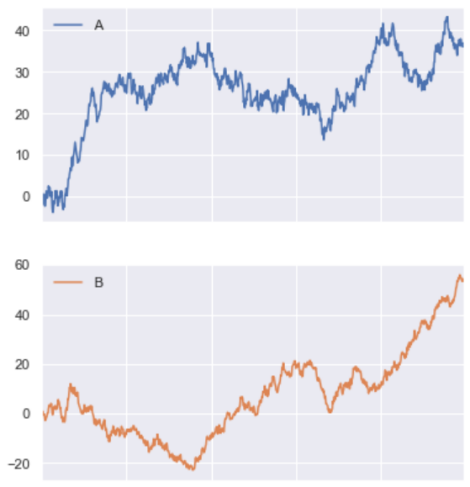
**【5】多子图**

df = pd.DataFrame(np.random.randn(1000, 4).cumsum(axis=0),

columns=list("ABCD"),

index=np.arange(1000))

df.plot(subplots=**True**, figsize=(6, 16))





设定图形安排

df.plot(subplots=**True**, layout=(2, 2), figsize=(16, 6), sharex=**False**)



其他内容请参考Pandas中文文档

[https://www.pypandas.cn/docs/user\_guide/visualization.html#plot-formatting](https://www.pypandas.cn/docs/user_guide/visualization.html" \l "plot-formatting)