



# Matplotlib使用教程

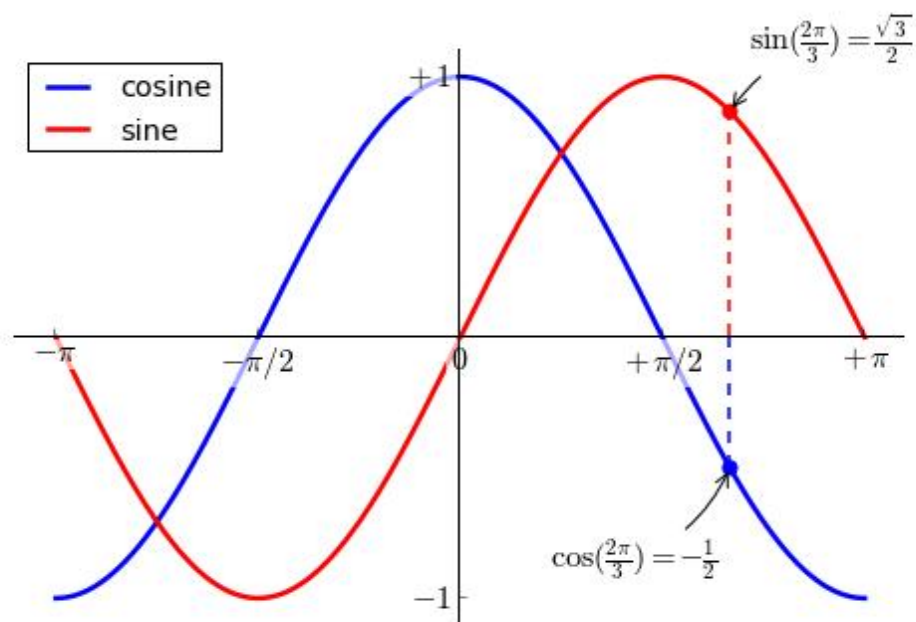
黄春林

# 简单介绍

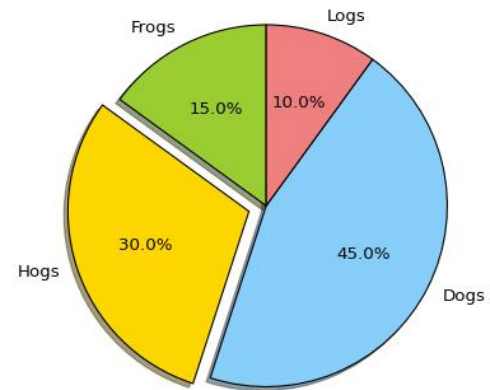
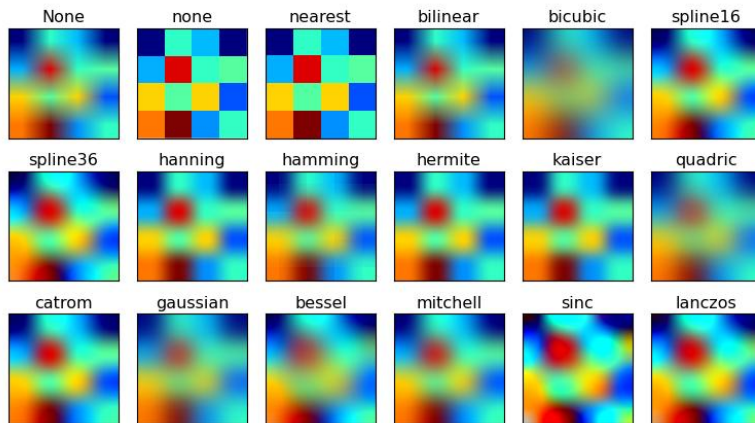
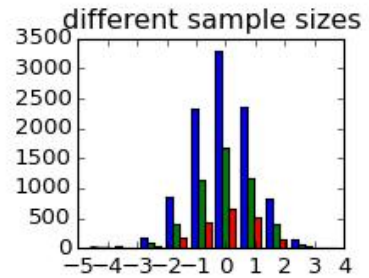
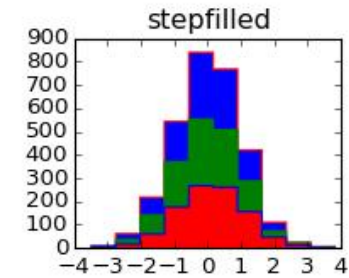
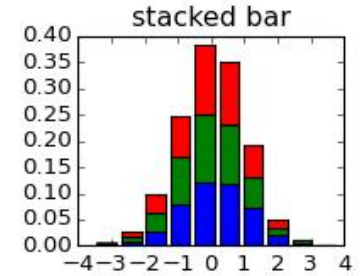
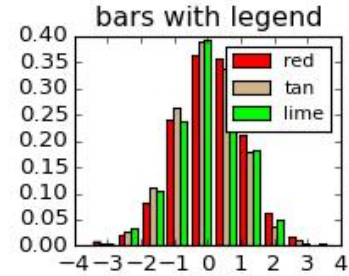
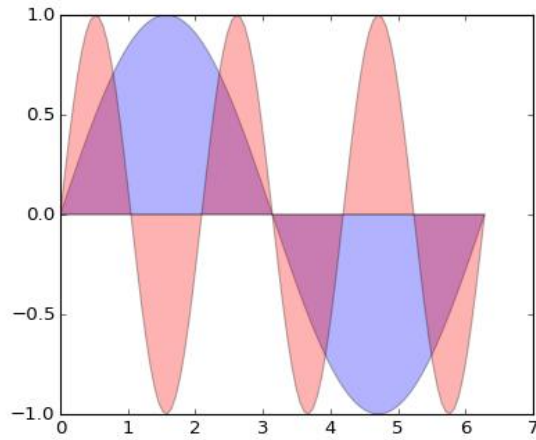
- Most popular plotting library for python
- Created by John Hunter (1968-2012)
- Has a lot in common with MatLab's plotting library, both functionally and syntactically.
- <http://matplotlib.org/> 中的 **Example** 和 **Gallery**



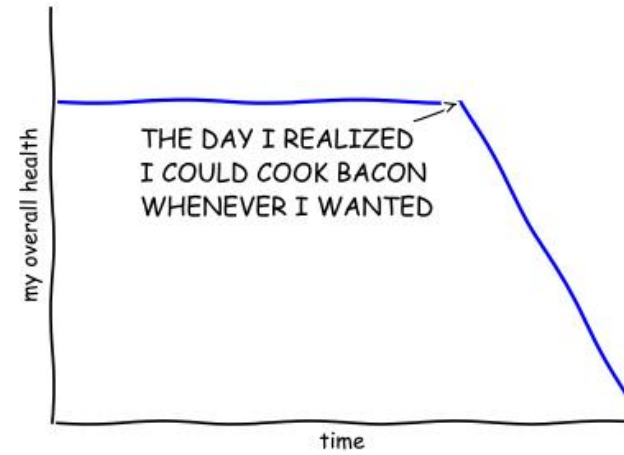
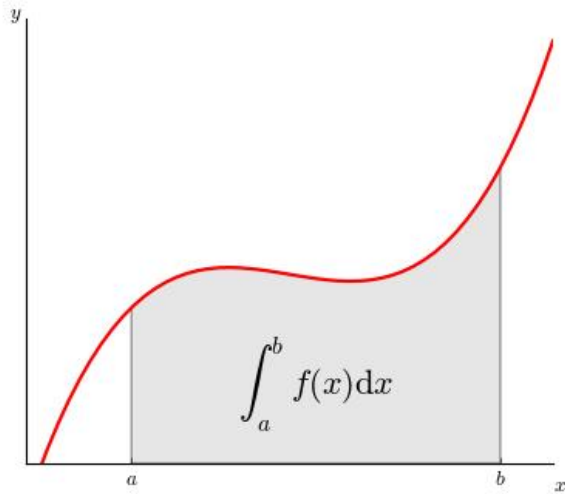
# 美观



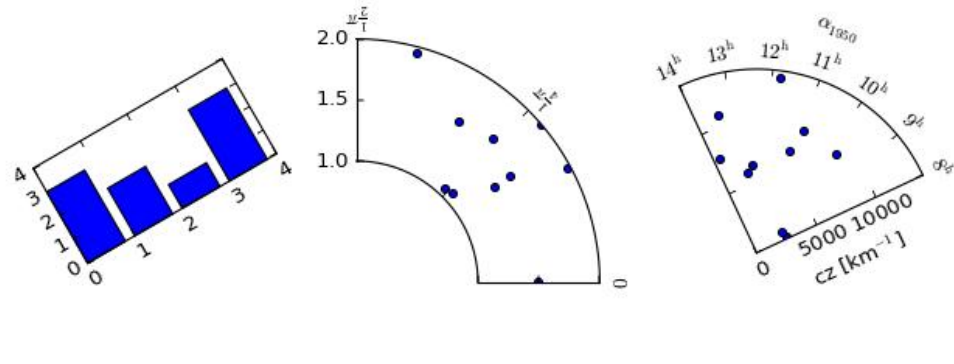
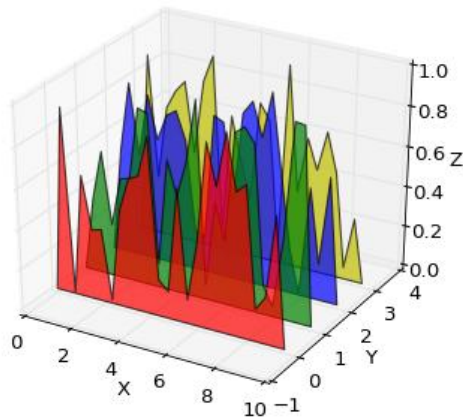
# 花样多



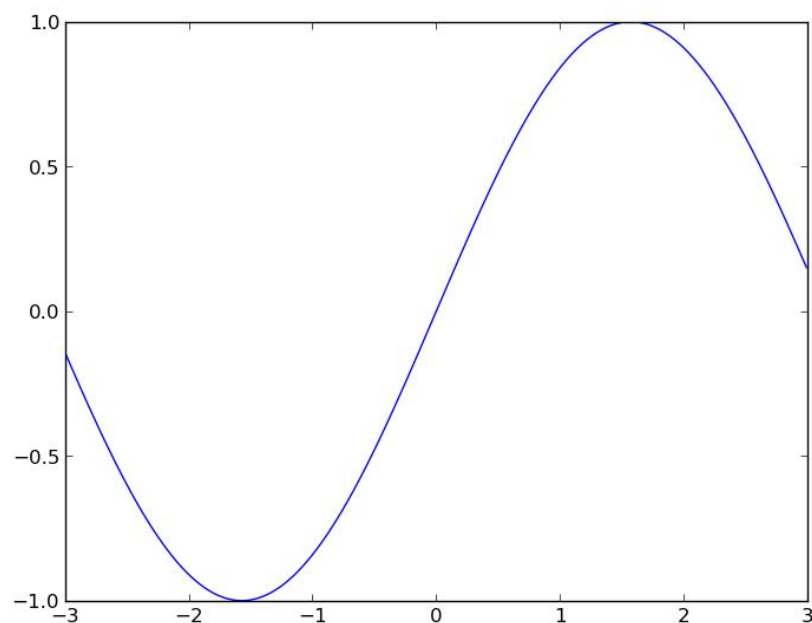
# 花样多



"Stove Ownership" from xkcd by Randall Monroe



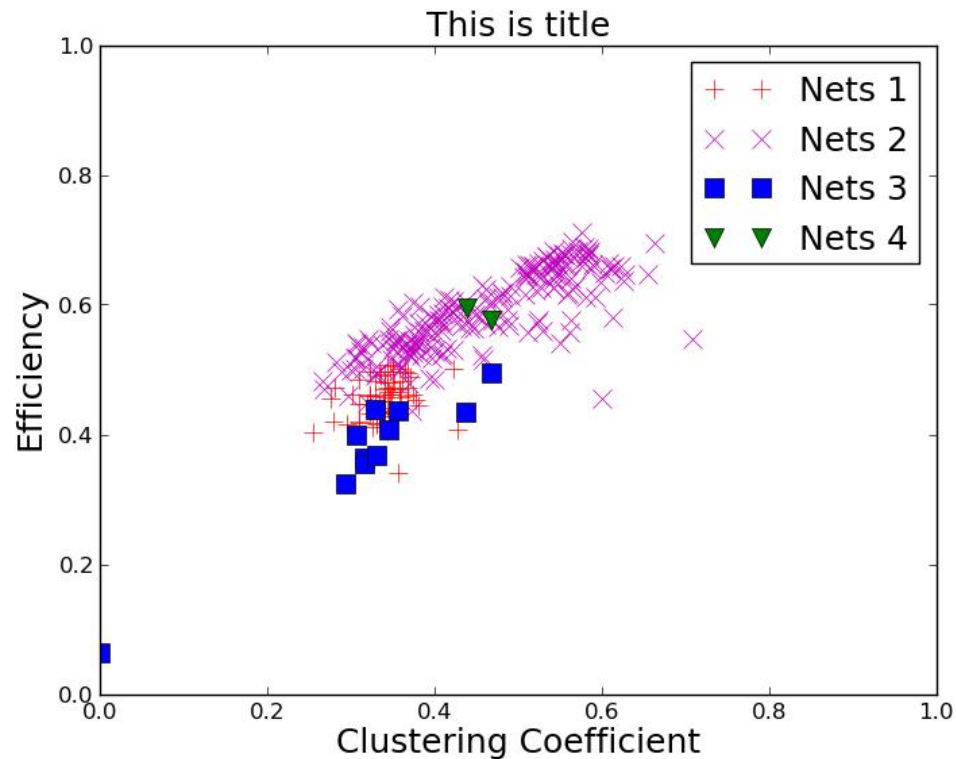
# 最简单的画图脚本



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

```
x=np.arange(-3,3,0.01)  
plt.plot(x,sin(x))  
plt.show()
```

# 从一个例子学起



```
#!/usr/bin/python

import numpy as np
import matplotlib.pyplot as plt

fs = 18
ms = 10

all_files = [ 'nets' + str(i) + '.txt' for i in np.arange(1,5) ]

plot_file = "dots.ex1.png"

data = [ np.loadtxt(all_files[i]) for i in range(0, len(all_files))]

subdata = data[0]
plt.plot(subdata[:,6], subdata[:,9], 'r+', markersize = ms, label = 'Nets 1')
subdata = data[1]
plt.plot(subdata[:,6], subdata[:,9], 'mx', markersize = ms, label = 'Nets 2')
subdata = data[2]
plt.plot(subdata[:,6], subdata[:,9], 'bs', markersize = ms, label = 'Nets 3')
subdata = data[3]
plt.plot(subdata[:,6], subdata[:,9], 'gv', markersize = ms, label = 'Nets 4')

plt.title('This is title', fontsize=fs)
plt.xlabel('Clustering Coefficient', fontsize=fs)
plt.ylabel('Efficiency', fontsize=fs)
plt.xlim((0,1))
plt.ylim((0,1))
for label in plt.gca().xaxis.get_ticklabels():
    label.set_fontsize(fs)
for label in plt.gca().yaxis.get_ticklabels():
    label.set_fontsize(fs)

plt.legend(loc=0, fontsize=fs)

plt.savefig(plot_file, bbox_inches='tight')
```

```
#!/usr/bin/python
```

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

```
fs = 18  
ms = 10
```

```
eps: plot_file = "dots.ex1.eps"  
pdf: plot_file = "dots.ex1.pdf"  
jpg: plot_file = "dots.ex1.jpg"  
ps: plot_file = "dots.ex1.ps"
```

```
all_files = [ 'nets' + str(i) + '.txt' for i in np.arange(1,5) ]
```

```
plot_file = "dots.ex1.png"
```

```
data = [ np.loadtxt(all_files[i]) for i in range(0, len(all_files))] 
```

( 待续 )



# 文件的格式（与matplotlib无关）

除了#开头的行以外，  
必须全部是数字

```
spring@Cunlunin: ~/Documents/CDC.files/done.papers/PNAS.pa...  * spring@Cunlunin: /media/spring/vvORR/workspace/Exp
1 # index size edges density average_degree average_strength transitivity average_path_len
2 1 109 728 0.123683 13.3578 161.413 0.357562 2.27115 1.33539 0.500351
3 2 113 721 0.113938 12.7611 166.425 0.352073 2.40629 1.31566 0.479327
4 3 107 669 0.117969 12.5047 161.028 0.344682 2.60395 1.35447 0.465328
5 4 107 594 0.104743 11.1028 108.449 0.280951 2.43767 1.32705 0.472418
6 5 107 784 0.138247 14.6542 171.925 0.422711 2.32816 1.38586 0.500685
7 6 113 683 0.107933 12.0885 153.239 0.322037 2.561 1.34077 0.461823
8 7 116 637 0.0955022 10.9828 156.828 0.381426 2.60585 1.34234 0.44522
9 8 113 721 0.113938 12.7611 134.779 0.349323 2.31762 1.45141 0.471882
10 9 110 447 0.0745621 8.12727 94.2727 0.279387 2.77314 1.45311 0.419136
11 10 94 415 0.0949439 8.82979 112.213 0.275481 2.51384 1.27526 0.455487
12 11 102 483 0.0937682 9.47059 123.157 0.322886 2.55318 1.37508 0.432667
13 12 128 796 0.0979331 12.4375 169.312 0.360009 2.53925 1.36148 0.456385
14 13 142 1162 0.116072 16.3662 237.113 0.34337 2.43802 1.29922 0.480107
15 14 141 1146 0.116109 16.2553 209.475 0.334889 2.32847 1.29258 0.490804
16 15 142 1218 0.121666 17.1549 212.493 0.344131 2.21307 1.27281 0.506854
17 16 144 1181 0.114705 16.4028 217.306 0.334578 2.26777 1.31857 0.496615
18 17 143 1000 0.0984931 13.986 226.14 0.373769 2.49837 1.2692 0.461795
19 18 146 1442 0.136231 19.7534 237.712 0.350236 2.28758 1.30791 0.506086
20 19 146 1083 0.102315 14.8356 208.397 0.346318 2.39879 1.33441 0.461083
21 20 151 1332 0.117616 17.6424 231.735 0.347251 2.28124 1.30113 0.497071
22 21 153 1252 0.107671 16.366 231.216 0.328791 2.3581 1.3075 0.483087
23 22 151 1205 0.106402 15.9603 232.675 0.33985 2.37572 1.37125 0.479929
24 23 155 1501 0.125765 19.3677 271.652 0.349539 2.33238 1.29693 0.496255
25 24 153 1059 0.0910733 13.8431 193.725 0.329297 2.4859 1.37914 0.459652
26 25 150 1100 0.098434 14.6667 223.093 0.348827 2.49826 1.29105 0.460735
27 26 150 1272 0.113826 16.96 216.787 0.323056 2.25754 1.31288 0.497248
28 27 149 1089 0.0987666 14.6174 194.872 0.340882 2.42291 1.31401 0.469959
29 28 149 1300 0.117903 17.4497 266.215 0.361846 2.41475 1.2952 0.482786
30 29 151 1176 0.103841 15.5762 196.291 0.309905 2.33227 1.28445 0.483981
31 30 151 1289 0.113819 17.0728 245.073 0.34061 2.2921 1.31368 0.493607
32 31 148 1119 0.102868 15.1216 216.122 0.34438 2.42627 1.30015 0.472496
33 32 147 1117 0.104091 15.1973 202.082 0.378156 2.65343 1.42389 0.454091
34 33 155 1511 0.126602 19.4968 261.419 0.370815 2.44214 1.3379 0.489167
35 34 152 1045 0.0910596 13.75 191.763 0.322696 2.4932 1.30269 0.459574
36 35 147 1273 0.118628 17.3197 224.844 0.367379 2.29951 1.34774 0.49396
```

实际上是第7列

```
subdata = data[0]
plt.plot(subdata[:,6], subdata[:,9], 'r+', markersize = ms, label = 'Nets 1')
subdata = data[1]
plt.plot(subdata[:,6], subdata[:,9], 'mx', markersize = ms, label = 'Nets 2')
subdata = data[2]
plt.plot(subdata[:,6], subdata[:,9], 'bs', markersize = ms, label = 'Nets 3')
subdata = data[3]
plt.plot(subdata[:,6], subdata[:,9], 'gv', markersize = ms, label = 'Nets 4')

plt.title('This is title', fontsize=fs)
plt.xlabel('Clustering Coefficient', fontsize=fs)
plt.ylabel('Efficiency', fontsize=fs)
plt.xlim((0,1))
plt.ylim((0,1))
for label in plt.gca().xaxis.get_ticklabels():
    label.set_fontsize(fs)
for label in plt.gca().yaxis.get_ticklabels():
    label.set_fontsize(fs)

plt.legend(loc=0, fontsize=fs)

plt.savefig(plot_file, bbox_inches='tight')
```

# plot 函数

- 画点或者线

```
plot(x, y) # plot x and y using default line style and color
```

```
plot(x, y, 'bo') # plot x and y using blue circle markers
```

```
plot(y) # plot y using x as index array 0..N-1
```

```
plot(y, 'r+') # ditto, but with red plusses
```

- 如果x或者y是二维矩阵，则相当于画列向量

- 可以在一个plot函数中画多个点或线

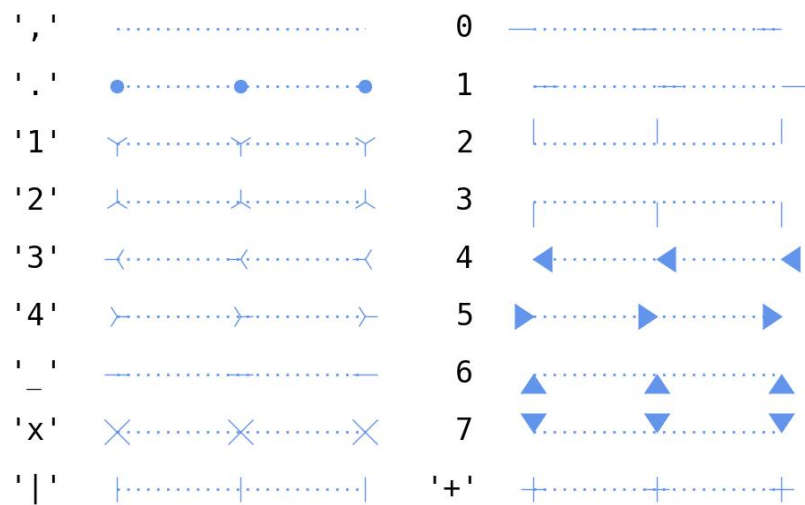
```
plot(x1, y1, 'g^', x2, y2, 'g-') plot(x1, y1, x2, y2, 'g-')
```

- 参数有fmt形式和kwargs形式

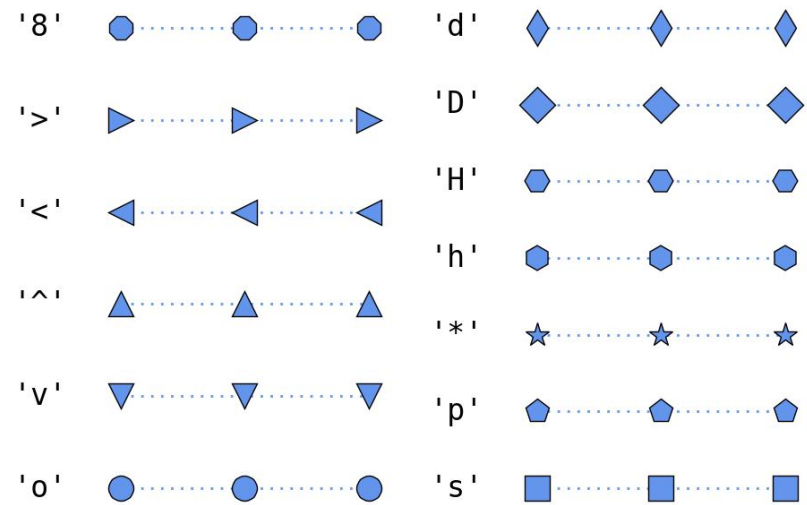
```
'g^-' VS color='g', marker='^', ls='-'
```

# Markers

un-filled markers



filled markers



# Line styles





























































































































































line styles



# Colors

- 单词，如red
- 字母，如r
- 6个16进制数，如'#FF0000'或'#ff0000'
- 含有三(RGB)或四(RGBA)个元素(0~1)的元组，如(1, 0, 0)或(1,0,0,1)，只用于kwargs
- 灰度字符串，如'0.8'



	black		k		dimgray		dimgrey
	gray		grey		darkgrey		darkgray
	silver		lightgrey		lightgray		gainsboro
	whitesmoke		white		w		snow
	rosybrown		lightcoral		indianred		brown
	firebrick		maroon		darkred		red
	r		mistyrose		salmon		tomato
	darksalmon		coral		orangered		lightsalmon
	sienna		chocolate		peru		saddlebrown
	sandybrown		peachpuff		burlywood		linen
	bisque		darkorange		blanchedalmond		antiquewhite
	tan		navajowhite		wheat		papayawhip
	moccasin		orange		goldenrod		oldlace
	floralwhite		darkgoldenrod		khaki		cornsilk
	gold		lemonchiffon		beige		palegoldenrod
	darkkhaki		ivory		y		lightyellow
	lightgoldenrodyellow		olive		darkolivegreen		yellow
	olivedrab		yellowgreen		sage		greenyellow
	chartreuse		lawngreen		darkseagreen		lightsage
	darksage		honeydew		limegreen		palegreen
	lightgreen		forestgreen		lime		darkgreen
	green		g		mintcream		seagreen
	mediumseagreen		springgreen		turquoise		mediumspringgreen
	mediumaquamarine		aquamarine		lightcyan		lightseagreen
	mediumturquoise		azure		teal		paleturquoise
	darkslategrey		darkslategray		aqua		darkcyan
	c		cyan		lightblue		darkturquoise
	cadetblue		powderblue		steelblue		deepskyblue
	skyblue		lightskyblue		lightslategray		aliceblue
	dodgerblue		lightsteelblue		cornflowerblue		slategrey
	slategray		lavender		midnightblue		royalblue
	ghostwhite		mediumblue		blue		navy
	darkblue		darkslateblue		mediumslateblue		b
	slateblue		indigo		darkorchid		mediumpurple
	blueviolet		thistle		plum		darkviolet
	mediumorchid		darkmagenta		m		violet
	purple		orchid		mediumvioletred		fuchsia
	magenta		lavenderblush		palevioletred		deeppink
	hotpink		lightpink				crimson
	pink						

# 一个字母可表示的颜色

- r 红色
- g 绿色
- b 蓝色
- c cyan
- m 紫色
- y 土黄色
- k 黑色
- w 白色



# plot函数的参数

- 可使用Line2D的所有参数
  - [http://matplotlib.org/api/pyplot\\_api.html#matplotlib.pyplot.plot](http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.plot)
  - class matplotlib.lines.Line2D(xdata, ydata, linewidth=None, linestyle=None, color=None, marker=None, markersize=None, markeredgewidth=None, markeredgecolor=None, markerfacecolor=None, markerfacecoloralt=u'none', fillstyle=u'full', antialiased=None, dash\_capstyle=None, solid\_capstyle=None, dash\_joinstyle=None, solid\_joinstyle=None, pickradius=5, drawstyle=None, markevery=None, \*\*kwargs)

# Line2D可以指定的参数

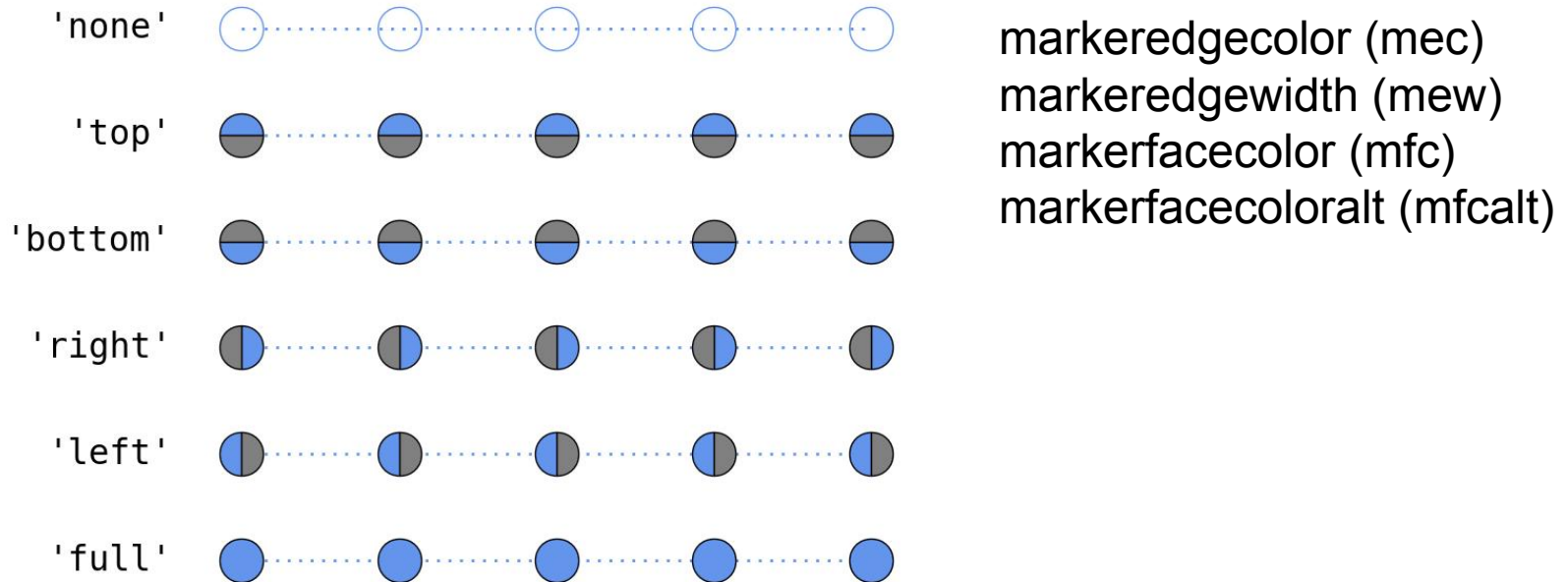
Property	Description
alpha	float (0.0 transparent through 1.0 opaque)
antialiased or aa	[True   False]
axes	an Axes instance
color or c	any matplotlib color
drawstyle	['default'   'steps'   'steps-pre'   'steps-mid'   'steps-post']
fillstyle	['full'   'left'   'right'   'bottom'   'top'   'none']
label	string or anything printable with '%s' conversion
linestyle or ls	['-'   '--'   '-.'   ':'   'None'   ''   '']
linewidth or lw	float value in points

# Line2D可以指定的参数

Property	Description
marker	A valid marker style
markeredgecolor or mec	any matplotlib color
markeredgewidth or mew	float value in points
markerfacecolor or mfc	any matplotlib color
markerfacecoloralt or mfcalt	any matplotlib color
markersize or ms	float
markevery	[None   int   length-2 tuple of int   slice   list/array of int   float   length-2 tuple of float]
xdata	1D array
ydata	1D array
visible	[True   False]

# fill style

fill style



1. 默认替代颜色为无色，不是灰色
2. 只设置color，相当于同时设置了mec和mfc

```
subdata = data[0]
plt.plot(subdata[:,6], subdata[:,9], 'r+', markersize = ms, label = 'Nets 1')
subdata = data[1]
plt.plot(subdata[:,6], subdata[:,9], 'mx', markersize = ms, label = 'Nets 2')
subdata = data[2]
plt.plot(subdata[:,6], subdata[:,9], 'bs', markersize = ms, label = 'Nets 3')
subdata = data[3]
plt.plot(subdata[:,6], subdata[:,9], 'gv', markersize = ms, label = 'Nets 4')
```

实际上是第7列

可改为 `ms = ms`

```
plt.title('This is title', fontsize=fs)
plt.xlabel('Clustering Coefficient', fontsize=fs)
plt.ylabel('Efficiency', fontsize=fs)
plt.xlim((0,1))
plt.ylim((0,1))
for label in plt.gca().xaxis.get_ticklabels():
    label.set_fontsize(fs)
for label in plt.gca().yaxis.get_ticklabels():
    label.set_fontsize(fs)

plt.legend(loc=0, fontsize=fs)

plt.savefig(plot_file, bbox_inches='tight')
```

# title 函数

- `matplotlib.pyplot.title(s, *args, **kwargs)`
  - label: str
  - fontdict: dict
    - 默认为: `{'fontsize': rcParams['axes.titlesize'], 'fontweight' : rcParams['axes.titleweight'], 'verticalalignment': 'baseline', 'horizontalalignment': 'center'}`
  - loc : `{'center', 'left', 'right'}`, str, optional

# title 函数

- 可使用Text的所有参数
  - `class matplotlib.text.Text(x=0, y=0, text=u", color=None, verticalalignment=u'baseline', horizontalalignment=u'left', multialignment=None, fontproperties=None, rotation=None, linespacing=None, rotation_mode=None, **kwargs)`

# Text可以指定的参数

Property	Description
alpha	float (0.0 transparent through 1.0 opaque)
axes	an Axes instance
backgroundcolor	any matplotlib color
color or c	any matplotlib color
family or fontfamily or fontname or name	[FONTNAME   'serif'   'sans-serif'   'cursive'   'fantasy'   'monospace' ]
fontproperties or font_properties	a matplotlib.font_manager.FontProperties instance
horizontalalignment or ha	[ 'center'   'right'   'left' ]
label	string or anything printable with '%s' conversion
position	(x,y)



# Text可以指定的参数

Property	Description
rotation	[ angle in degrees   'vertical'   'horizontal' ]
size or fontsize	[size in points   'xx-small'   'x-small'   'small'   'medium'   'large'   'x-large'   'xx-large' ]
stretch or fontstretch	[a numeric value in range 0-1000   'ultra-condensed'   'extra-condensed'   'condensed'   'semi-condensed'   'normal'   'semi-expanded'   'expanded'   'extra-expanded'   'ultra-expanded' ]
style or fontstyle	[ 'normal'   'italic'   'oblique' ]
text	string or anything printable with '%s' conversion
verticalalignment or va or ma	[ 'center'   'top'   'bottom'   'baseline' ]
visible	[True   False]
weight or fontweight	[a numeric value in range 0-1000   'ultralight'   'light'   'normal'   'regular'   'book'   'medium'   'roman'   'semibold'   'demibold'   'demi'   'bold'   'heavy'   'extra bold'   'black' ]
x	float
y	float

# xlabel, ylabel

- 设置横坐标和纵坐标的名字
- 用法与 **title** 一样

# 设置坐标轴

- 坐标范围设置

- 设置x轴范围：xlim函数

- 设置y轴范围：ylim函数

```
xmin, xmax = xlim() # return the current xlim
```

```
xlim( xmin, xmax ) # set the xlim to xmin, xmax
```

```
xlim( xmin, xmax ) # set the xlim to xmin, xmax
```

```
xlim(xmax=3) # adjust the max leaving min unchanged
```

```
xlim(xmin=1) # adjust the min leaving max unchanged
```

# 设置线性坐标系与对数坐标系

- `xscale/yscale(scale, **kwargs)`
  - `scale`的3种取值: 'linear' | 'log' | 'symlog'

Properties	Description	Available scale
<code>basex/basey</code>	The base of the logarithm	log, symlog
<code>subsx/subsy</code>	Where to place the subticks between each major tick. Should be a sequence of integers.	log, symlog
<code>nonposx/nonposy</code>	['mask'   'clip']; non-positive values in x or y can be masked as invalid, or clipped to a very small positive number.	log
<code>linthreshx/linthreshy</code>	The range (-x, x) within which the plot is linear (to avoid having the plot go to infinity around zero).	symlog
<code>linscalex/linscaley</code>	This allows the linear range (-linthresh to linthresh) to be stretched relative to the logarithmic range. Its value is the number of decades to use for each half of the linear range.	symlog

# 在对数坐标系下画图

- `matplotlib.pyplot.semilogx(*args, **kwargs)`
- `matplotlib.pyplot.semilogy(*args, **kwargs)`
- `matplotlib.pyplot.loglog(*args, **kwargs)`
- 参数同`plot (Line2D)`
- 注意与前面`xscale/yscale`函数的区别
  - `xscale/yscale` 只设置坐标系，没有画图功能

# 自定义坐标

- `matplotlib.pyplot.xticks(*args, **kwargs)`
- `matplotlib.pyplot.yticks(*args, **kwargs)`

## – 可使用Text的参数

```
# return locs, labels where locs is an array of tick locations and  
# labels is an array of tick labels.
```

```
locs, labels = plt.xticks()
```

```
# set the locations of the xticks
```

```
plt.xticks( np.arange(6) )
```

```
# set the locations and labels of the xticks
```

```
plt.xticks( np.arange(5), ('Tom', 'Dick', 'Harry', 'Sally', 'Sue') )
```

```
plt.xticks( np.arange(12), calendar.month_name[1:13], rotation=17 )
```

```
# 另外一种方法
```

```
plt.gca().set_xticks(np.arange(12))
```

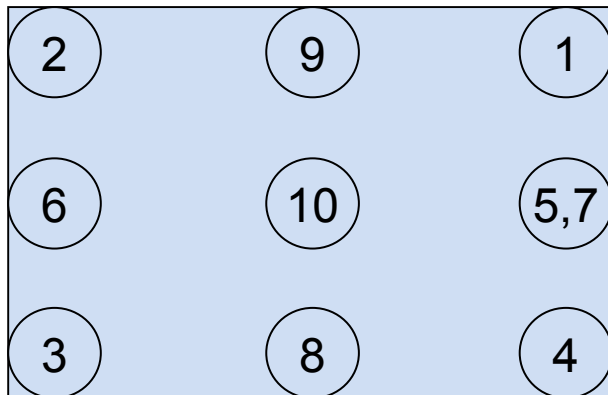
```
plt.gca().set_xticklabels(calendar.month_name[1:13], rotation=17)
```

# 图例的用法

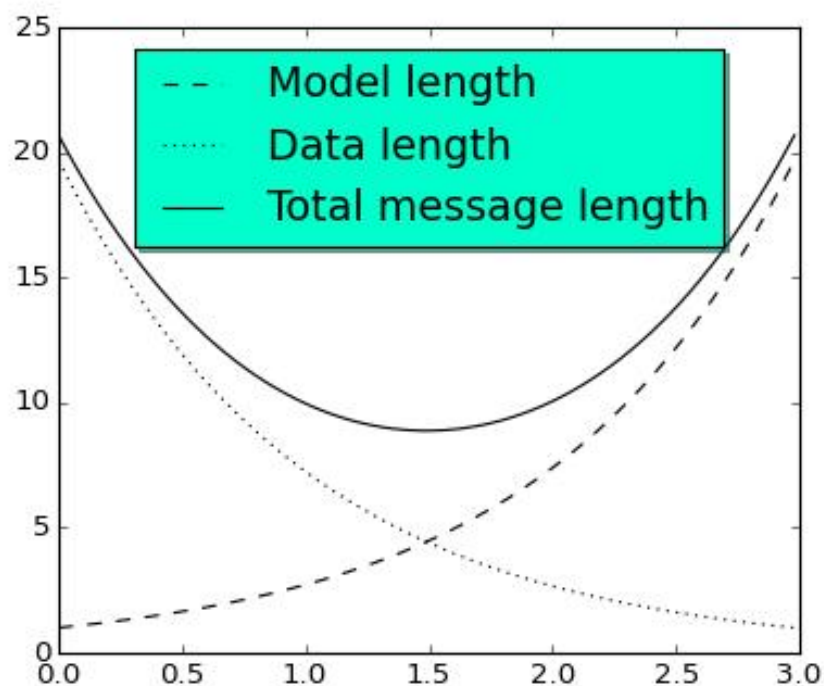
- `matplotlib.pyplot.legend(*args, **kwargs)`

- `loc`: default 0
- `fontsize`
- `numpoints`
- `shadow`
- `bbox_to_anchor`
- ...

Location String	Location Code
'best'	0
'upper right'	1
'upper left'	2
'lower left'	3
'lower right'	4
'right'	5
'center left'	6
'center right'	7
'lower center'	8
'upper center'	9
'center'	10



# 设置图例背景色和阴影



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

```
# Make some fake data.
a = b = np.arange(0,3, .02)
c = np.exp(a)
d = c[::-1]
```

```
# Create plots with pre-defined labels.
plt.plot(a, c, 'k--', label='Model length')
plt.plot(a, d, 'k:', label='Data length')
plt.plot(a, c+d, 'k', label='Total message length')
```

```
legend = plt.legend(loc='upper center',
shadow=True, fontsize='x-large')
```

```
# Put a nicer background color on the legend.
legend.get_frame().set_facecolor('#00FFCC')
```

```
plt.show()
```



# 保存图片

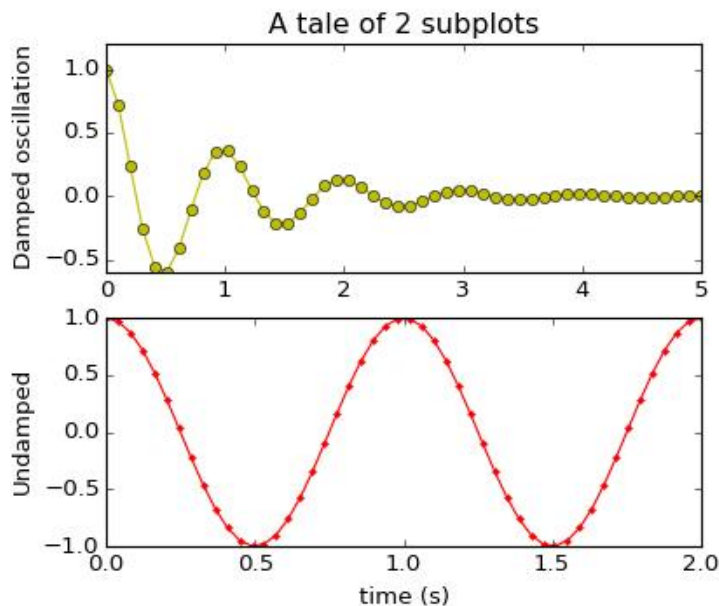
- `matplotlib.pyplot.savefig(*args, **kwargs)`
  - 文件名是必需参数
  - `plt.savefig(plot_file, bbox_inches='tight')`
    - `bbox_inches='tight'` 去掉不需要的白边
    - 输出ps格式文件时无法使用这个参数
  - 其他参数
- 如果只想在屏幕上显示出来
  - `plt.show()`

# 网格开关

- `matplotlib.pyplot.grid(b=None, which=u'major', axis=u'both', **kwargs)`
  - `b`: [True | False], 是否打网格, 默认反转**b**
  - `which`: ['major' | 'minor' | 'both'], 在哪种坐标处打网格, 默认为'major'
  - `axis`: ['both' | 'x' | 'y'], 横线和竖线|只竖线|只横线, 默认为'both'
  - 还可使用Line2D中的参数
  - 最简单的使用: `plt.grid()`

# 多图

- `matplotlib.pyplot.subplot(*args, **kwargs)`
  - `subplot(nrows, ncols, plot_number)`
  - 如果数字都小于10, 可以写成`subplot(211)`



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

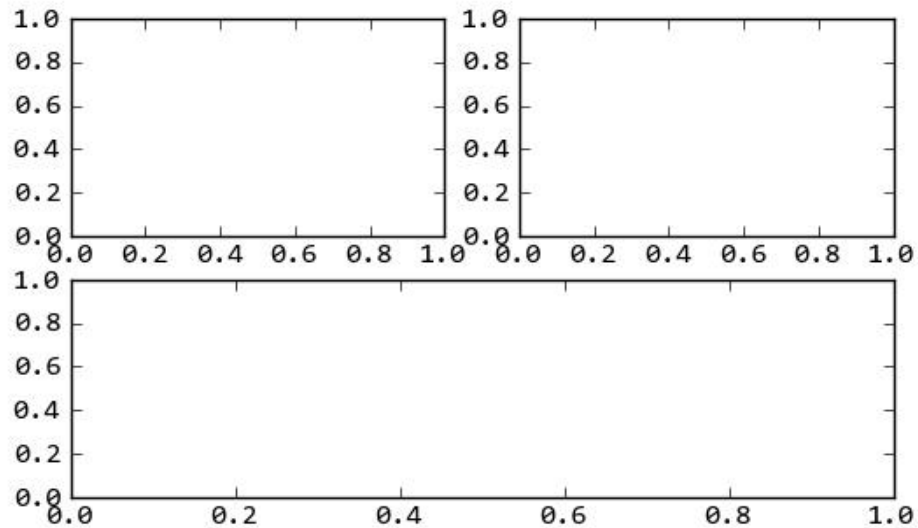
```
x1 = np.linspace(0.0, 5.0)
x2 = np.linspace(0.0, 2.0)
y1 = np.cos(2 * np.pi * x1) * np.exp(-x1)
y2 = np.cos(2 * np.pi * x2)
```

```
plt.subplot(2, 1, 1)
plt.plot(x1, y1, 'yo-')
plt.title('A tale of 2 subplots')
plt.ylabel('Damped oscillation')
```

```
plt.subplot(2, 1, 2)
plt.plot(x2, y2, 'r.-')
plt.xlabel('time (s)')
plt.ylabel('Undamped')
```

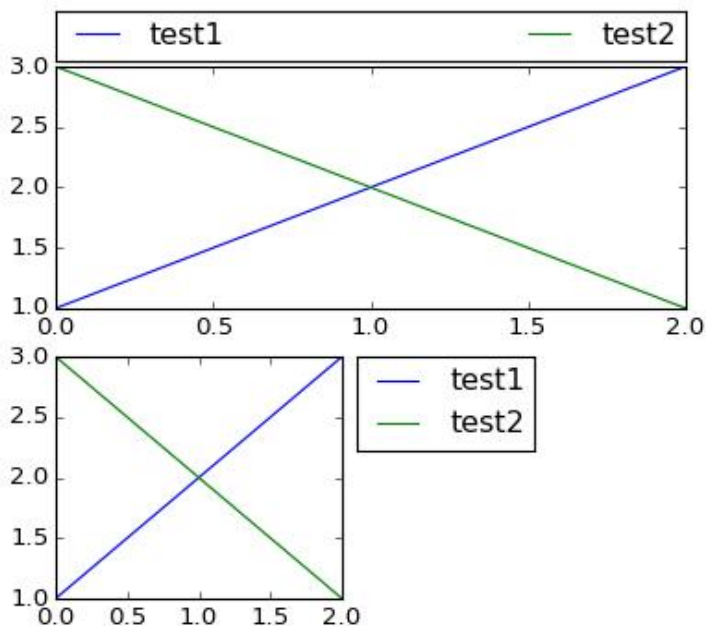
```
plt.show()
```

# 多图的一个例子



```
plt.subplot(221) # 第一行的左图  
plt.subplot(222) # 第一行的右图  
plt.subplot(212) # 第二整行  
plt.show()
```

# 把图例放在图片以外的地方



四元组：(loc的x相对坐标, loc的y相对坐标, 相对长度, 相对高度)  
二元组：四元组的前两个

```
import matplotlib.pyplot as plt
```

```
plt.subplot(211)  
plt.plot([1,2,3], label="test1")  
plt.plot([3,2,1], label="test2")  
# Place a legend above this legend, expanding  
# itself to fully use the given bounding box.  
plt.legend(bbox_to_anchor=(0., 1.02, 1., .102),  
loc=3, ncol=2, mode="expand",  
borderaxespad=0.)
```

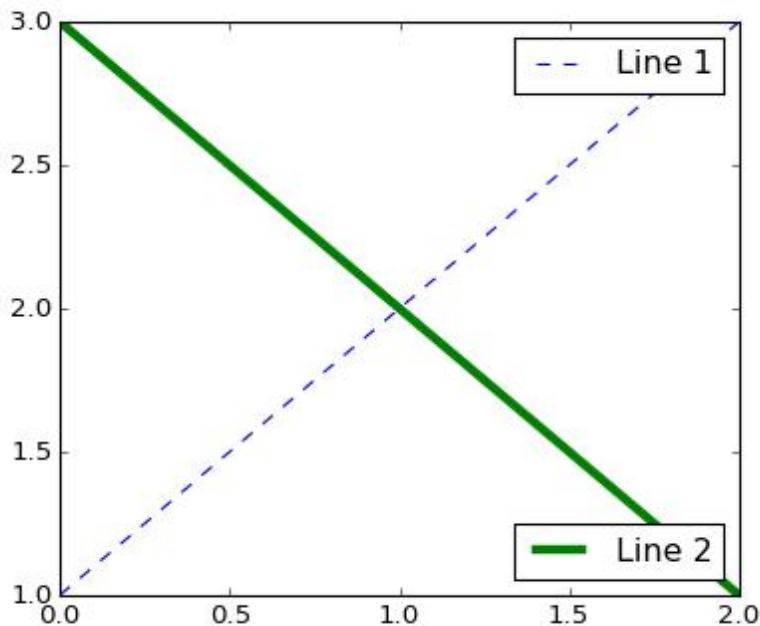
```
plt.subplot(223)
```

图例与坐标轴的距离为0

```
plt.plot([1,2,3], label="test1")  
plt.plot([3,2,1], label="test2")  
# Place a legend to the right of this smaller  
# figure.  
plt.legend(bbox_to_anchor=(1.05, 1), loc=2,  
borderaxespad=0.)
```

```
plt.show()
```

# 图例分开放置



没有运行成功!

```
import matplotlib.pyplot as plt
```

```
line1, = plt.plot([1,2,3], label="Line 1",  
linestyle='--')
```

```
line2, = plt.plot([3,2,1], label="Line 2",  
linewidth=4)
```

```
# Create a legend for the first line.
```

```
first_legend = plt.legend(handles=[line1], loc=1)
```

```
# Add the legend manually to the current Axes.
```

```
ax = plt.gca().add_artist(first_legend)
```

```
# Create another legend for the second line.
```

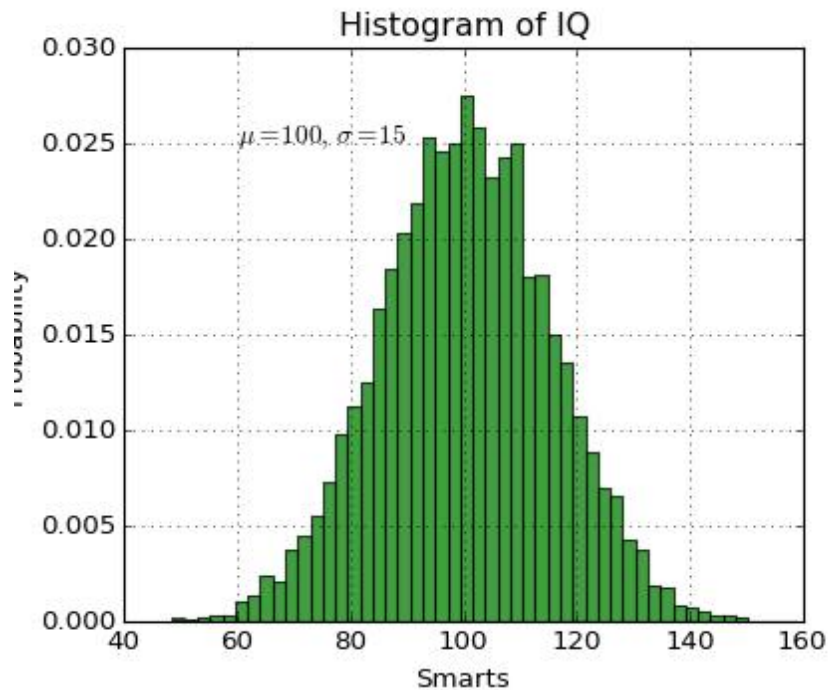
```
plt.legend(handles=[line2], loc=4)
```

```
plt.show()
```

# 直方图

- 两种方法
  - hist
  - bar/hbar

# hist



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

```
mu, sigma = 100, 15
x = mu + sigma * np.random.randn(10000)
```

```
# the histogram of the data
```

```
n, bins, patches = plt.hist(x, 50, normed=1,
                             facecolor='g', alpha=0.75)
```

```
plt.xlabel('Smarts')
```

```
plt.ylabel('Probability')
```

```
plt.title('Histogram of IQ')
```

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

```
plt.axis([40, 160, 0, 0.03])
```

```
plt.grid(True)
```

```
plt.show()
```



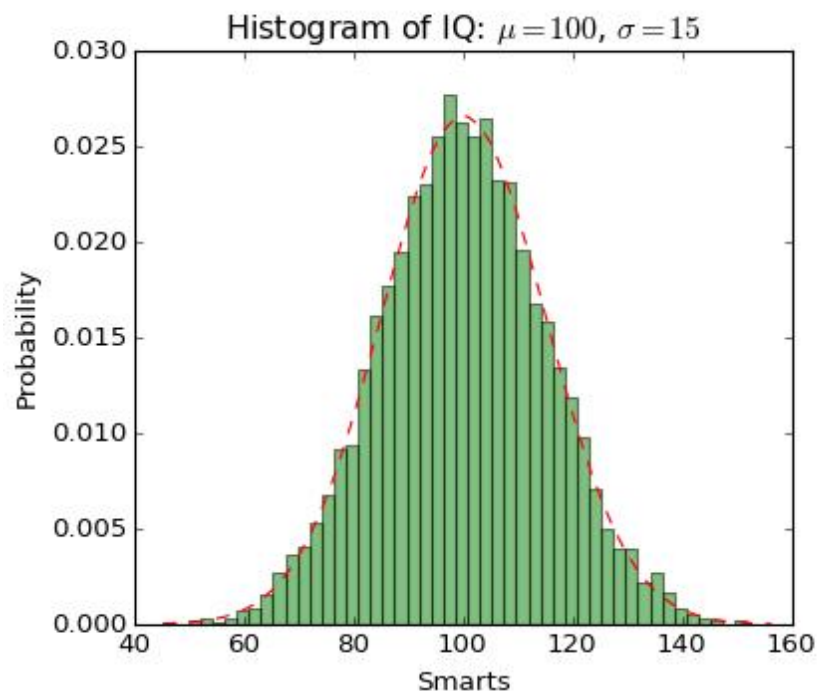
# hist函数

- `matplotlib.pyplot.hist(x, bins=10, range=None, normed=False, weights=None, cumulative=False, bottom=None, histtype=u'bar', align=u'mid', orientation=u'vertical', rwidth=None, log=False, color=None, label=None, stacked=False, hold=None, **kwargs)`
  - `x`: 一个列表或者多个列表（表示不同数据集，长度可以不一致）
  - `range`: 元组
  - `weights`: `x`里每个元素对bin高度的贡献（默认为1）
  - `bottom`: 数字或者长度为bins的列表
  - `histtype`: ['bar' | 'barstacked' | 'step' | 'stepfilled']
  - `align`: ['left' | 'mid' | 'right']
  - `orientation`: ['horizontal' | 'vertical']
  - `rwidth`: bar相对bin的宽度
  - `color`: 一种颜色或者颜色列表（针对不同数据集）

# text函数

- `matplotlib.pyplot.text(x, y, s, fontdict=None, withdash=False, **kwargs)`
  - Add text in string `s` to axis at location `x, y`, data coordinates
  - 注意这里的坐标是绝对坐标，与图例不同
  - 可使用**Text**的所有参数

# 直方图与相应折线图



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

# example data
mu = 100 # mean of distribution
sigma = 15 # standard deviation of distribution
x = mu + sigma * np.random.randn(10000)

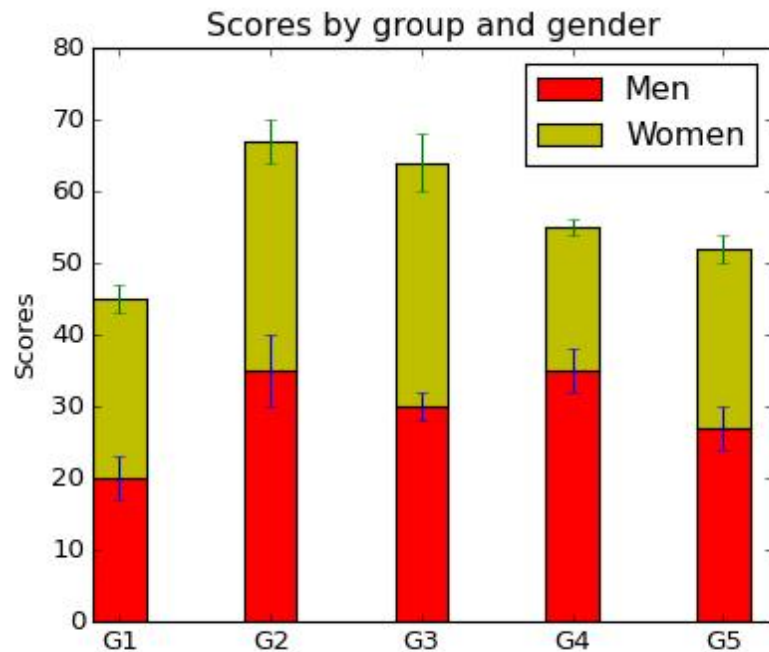
num_bins = 50
# the histogram of the data
n, bins, patches = plt.hist(x, num_bins, normed=1,
                             facecolor='green', alpha=0.5)
# add a 'best fit' line
y = mlab.normpdf(bins, mu, sigma)
plt.plot(bins, y, 'r--')
plt.xlabel('Smarts')
plt.ylabel('Probability')
plt.title(r'Histogram of IQ:  $\mu=100$ ,  $\sigma=15$ ')

# Tweak spacing to prevent clipping of ylabel
plt.subplots_adjust(left=0.15)
plt.show()
```

# bar/barh

- `matplotlib.pyplot.bar(left, height, width=0.8, bottom=None, hold=None, **kwargs)`
- `matplotlib.pyplot.barh(bottom, width, height=0.8, left=None, hold=None, **kwargs)`
  - 画`left`, `left + width`, `bottom`, `bottom + height`这四条线内的矩形
  - 其他参数（两个函数一样）：`color`, `edgecolor`, `linewidth`, `xerr`, `yerr`, `ecolor` (error bar的颜色), `capsize` (errorbar的大小), `error_kw`, `align` ('`edge`' or '`center`'), `orientation` ('`vertical`' or '`horizontal`'), `log`
- 适合在求出数据的概率分布后使用。

# 使用bar的一个例子



```
#!/usr/bin/env python
import numpy as np
import matplotlib.pyplot as plt
```

```
N = 5
```

```
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 = np.arange(N) # the x locations for the groups
```

```
width = 0.35 # the width of the bars: can also be
len(x) sequence
```

```
p1 = plt.bar(ind, menMeans, width, color='r',
yerr=womenStd)
```

```
p2 = plt.bar(ind, womenMeans, width, color='y',
bottom=menMeans, yerr=menStd)
```

```
plt.ylabel('Scores')
```

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

```
plt.xticks(ind+width/2., ('G1', 'G2', 'G3', 'G4', 'G5'))
```

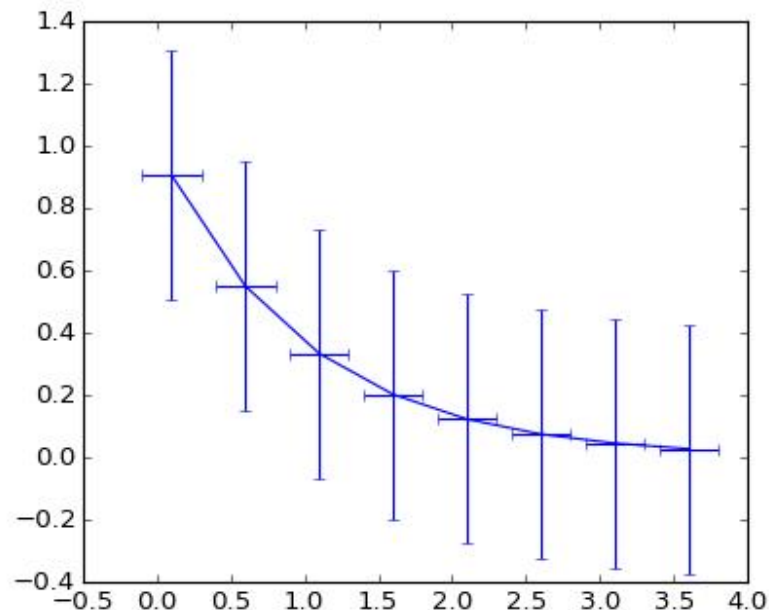
```
plt.yticks(np.arange(0,81,10))
```

```
plt.legend( (p1[0], p2[0]), ('Men', 'Women'))
```

```
plt.show()
```

# errorbar的画法

- `matplotlib.pyplot.errorbar(x, y, yerr=None, xerr=None, fmt='u',  
ecolor=None, elinewidth=None, capsize=3, barsabove=False,  
lolims=False, uplims=False, xlolims=False, xuplims=False,  
errorevery=1, capthick=None, hold=None, **kwargs)`
  - 可以使用`plot`函数的参数



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

# example data
x = np.arange(0.1, 4, 0.5)
y = np.exp(-x)

plt.errorbar(x, y, xerr=0.2, yerr=0.4)
plt.show()
```

# 支持中文

- 涉及到字体问题
- 请上网自行解决

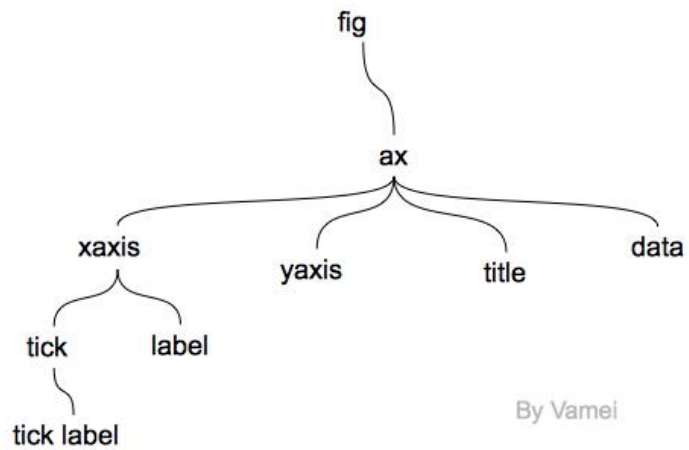
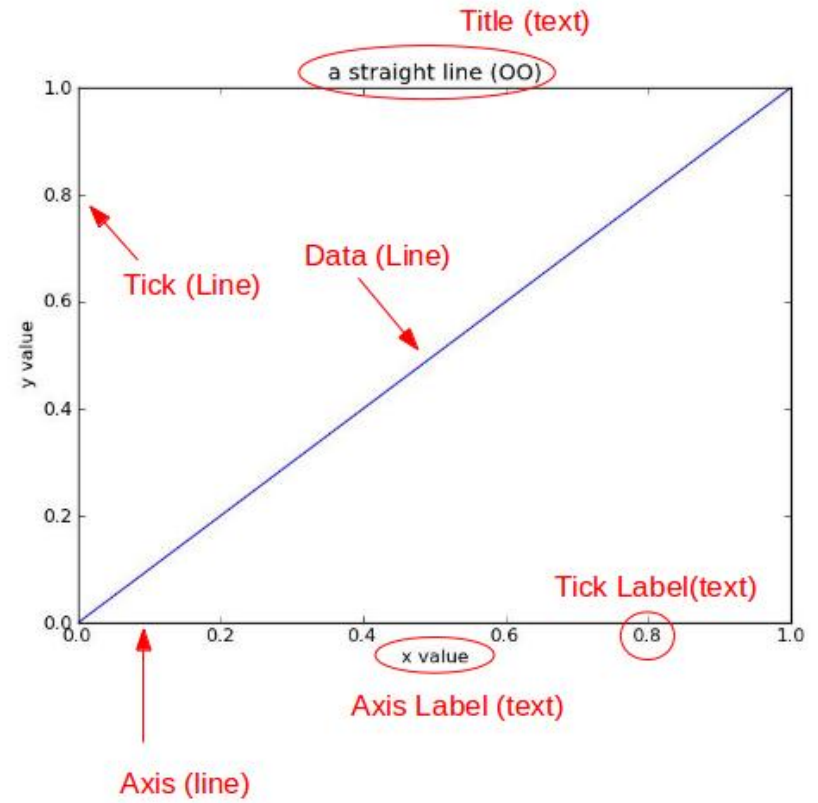
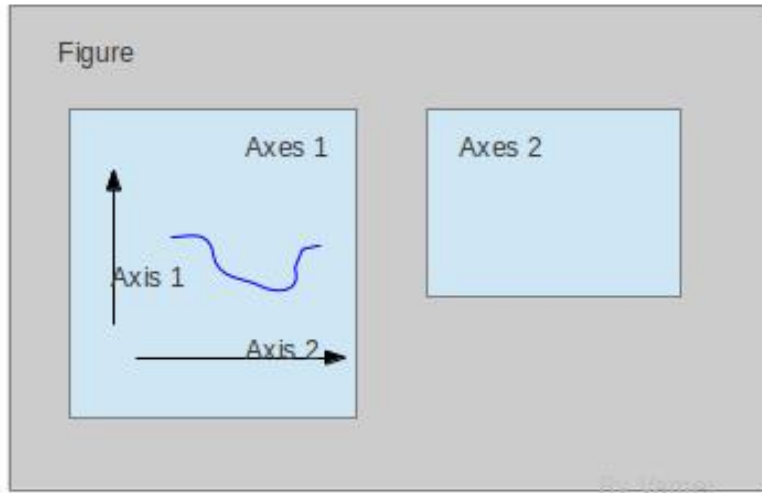
# 用matplotlib画图的第二种方法

面向对象绘图

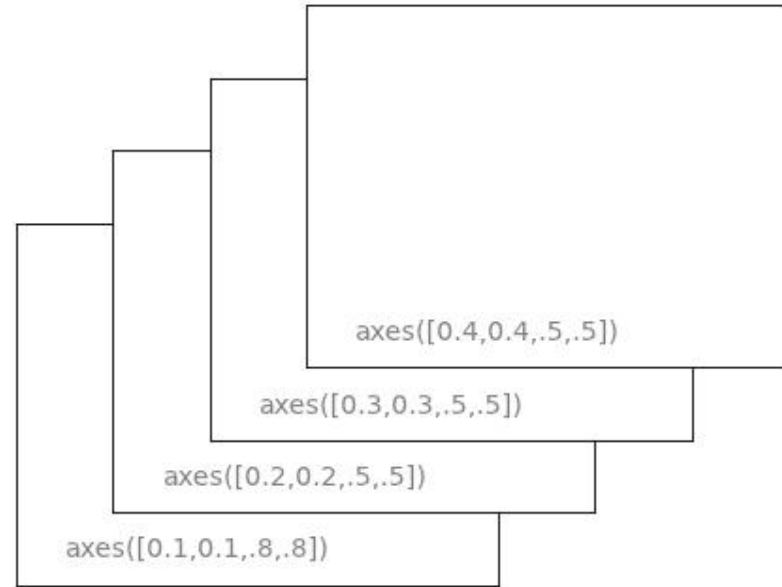
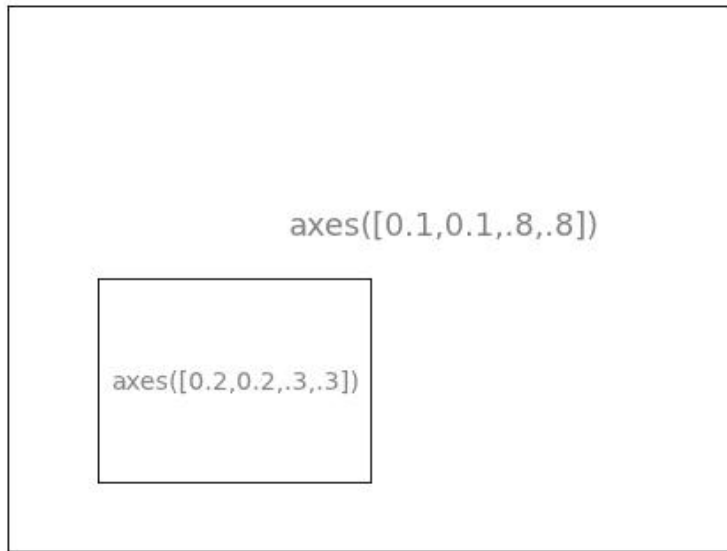


# 流程

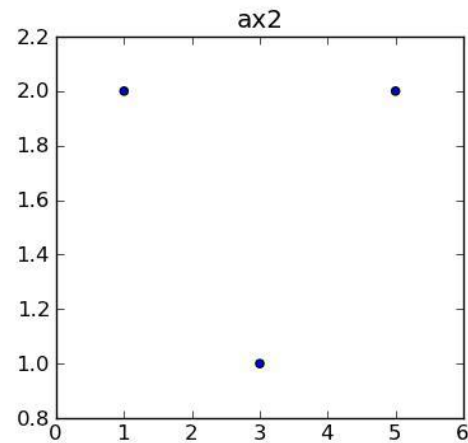
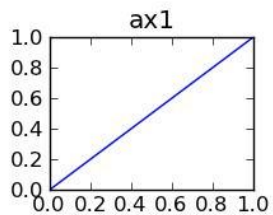
- **matplotlib API**包含有三层，**Artist**层处理所有的高层结构，例如处理图表、文字和曲线等的绘制和布局。通常我们只和**Artist**打交道，而不需要关心底层的绘制细节。
- 直接使用**Artists**创建图表的标准流程如下
  - 创建**Figure**对象
  - 用**Figure**对象创建一个或者多个**Axes**或者**Subplot**对象
  - 调用**Axes**等对象的方法创建各种简单类型的**Artists**



# subplots与axes的区别



# 第一个例子



```
import matplotlib.pyplot as plt
```

```
fig = plt.figure()
```

```
# first axes
```

```
ax1 = fig.add_axes([0.1, 0.1, 0.2, 0.2])
```

```
line, = ax1.plot([0,1], [0,1])
```

```
ax1.set_title("ax1")
```

```
# second axes
```

```
ax2 = fig.add_axes([0.4, 0.3, 0.4, 0.5])
```

```
sca = ax2.scatter([1,3,5],[2,1,2])
```

```
ax2.set_title("ax2")
```

```
plt.show()
```

# figure类

- `add_axes(*args, **kwargs)`
  - `rect = l,b,w,h`
  - `fig.add_axes(rect)`
  - `fig.add_axes(rect, frameon=False, axisbg='g')`
  - `fig.add_axes(rect, polar=True)`
  - `fig.add_axes(rect, projection='polar')`
  - `fig.add_axes(ax)`
  - `fig.add_axes(rect, label='axes1')`

# figure类

- `add_subplots(*args, **kwargs)`
  - `fig.add_subplot(111)`
  - `fig.add_subplot(1,1,1)`
  - `fig.add_subplot(sub)` # sub is an instance
- `gca(**kwargs)`
  - Get the current axes, creating one if necessary
- `get_axes()`

# figure类

- `legend(handles, labels, *args, **kwargs)`
  - `legend( (line1, line2, line3), ('label1', 'label2', 'label3'), 'upper right')`
- `savefig(*args, **kwargs)`
- `sca(a)`
  - Set the current axes to be a and return a
- `show(warn=True)`
- `suptitle(t, **kwargs)`
  - Add a centered title to the figure.
- `text(x, y, s, *args, **kwargs)`

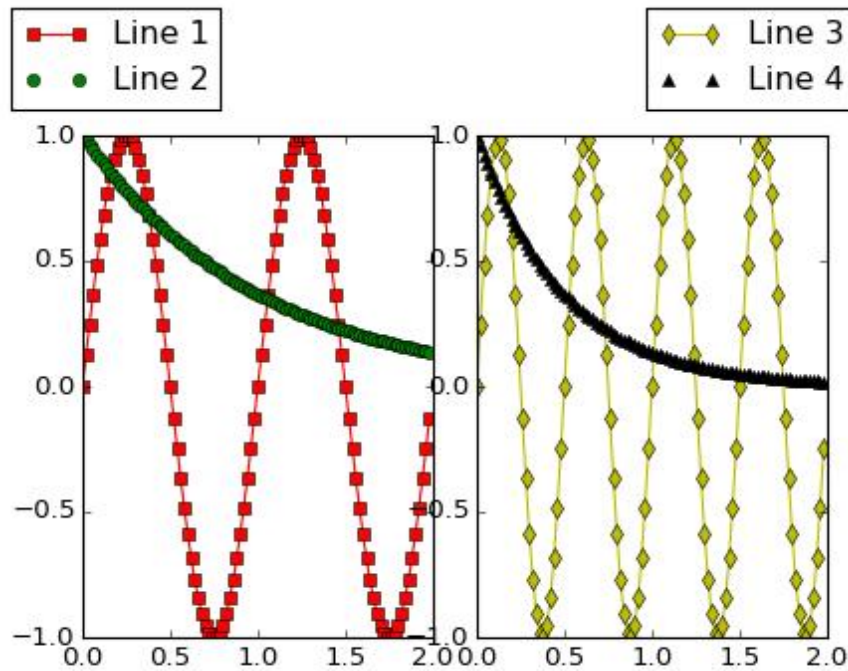
# figure类

## Others

- `get/set_edgecolor()`
- `get/set_facecolor()`
- `get/set_figwidth()`
- `get/set_figheight()`
- ....



# figure类



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

```
fig = plt.figure()
ax1 = fig.add_axes([0.1, 0.1, 0.4, 0.7])
ax2 = fig.add_axes([0.55, 0.1, 0.4, 0.7])
```

```
x = np.arange(0.0, 2.0, 0.02)
y1 = np.sin(2*np.pi*x)
y2 = np.exp(-x)
l1, l2 = ax1.plot(x, y1, 'rs-', x, y2, 'go')
```

```
y3 = np.sin(4*np.pi*x)
y4 = np.exp(-2*x)
l3, l4 = ax2.plot(x, y3, 'yd-', x, y4, 'k^')
```

```
fig.legend((l1, l2), ('Line 1', 'Line 2'),
           'upper left')
fig.legend((l3, l4), ('Line 3', 'Line 4'),
           'upper right')
plt.show()
```

# axes类

- `add_artist(a)`
- `add_line(line)`
- `axis(*v, **kwargs)`
- `bar(left, height, width=0.8, bottom=None, **kwargs)`
- `barh(bottom, width, height=0.8, left=None, **kwargs)`
- `pyplot, figure, axes`很多函数是一样的, `axes`在很多函数名称上加了 `set_`或`get_`。

# axis类（函数不全）

- `cla()`: clear
- `get_gridlines()`
- `get_label()`
- `get/set_label_text()`
- `get_major/minor_ticks(numticks=None)`
- `get_major/minorticklabels()`
- `get_major/minorticklines()`
- `get/set_ticklabels(minor=False, which=None)`
- `get_ticklines(minor=False)`
- `grid(b=None, which=u'major', **kwargs)`
- `set_ticks(ticks, minor=False)`

# axis的一些应用

- 选择是否显示刻度值
  - x轴上，1为下，2为上
  - y轴上，1为左，2为右

```
for tick in ax.xaxis.get_major_ticks():  
    tick.label1On = True  
    tick.label2On = False  
for tick in ax.yaxis.get_major_ticks():  
    tick.label1On = False  
    tick.label2On = False
```

# axis的一些应用

- 选择如何显示刻度

```
ax.xaxis.set_ticks_position('none')  
ax.yaxis.set_ticks_position('right')
```

# axis的一些应用

- 完全自定义坐标轴刻度及刻度值

```
ax.set_xticks([-250, -200, -150, -100, -50, 0, 50, 100, 150, 200])  
ax.set_xticklabels(['-250', ", ", '-150', ", ", '-50', ", ", '50', ", ", '150', "])  
ax.set_yticks([-40, -20, 0, 20, 40, 60, 80])  
ax.set_yticklabels(["", ", ", ", ", ", ", ", "])
```

# axis的一些应用

- 分别设置x轴和y轴上刻度值的字体大小

```
for tick in ax.xaxis.get_major_ticks():  
    tick.label1.set_fontsize(18)  
for tick in ax.yaxis.get_major_ticks():  
    tick.label1.set_fontsize(18)
```

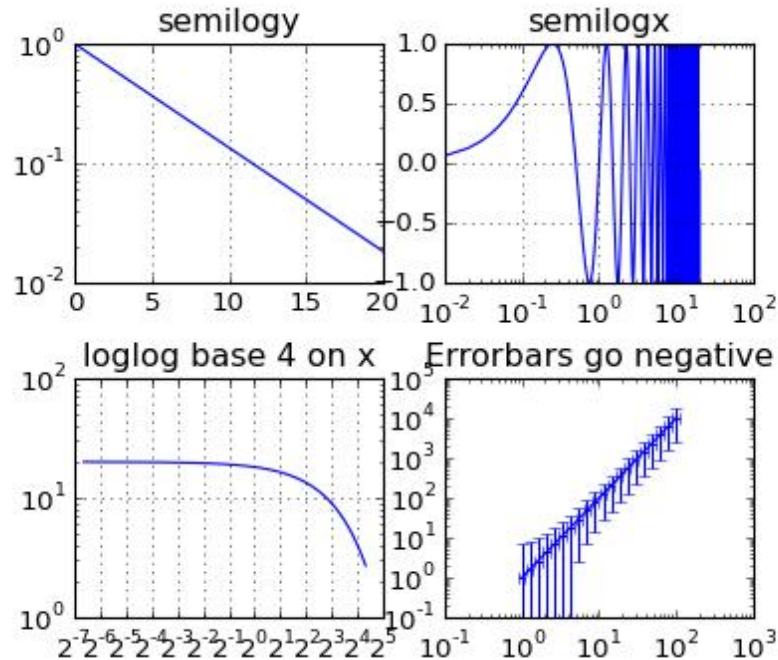
- 对比

```
for label in plt.gca().xaxis.get_ticklabels():  
    label.set_fontsize(18)  
for label in plt.gca().yaxis.get_ticklabels():  
    label.set_fontsize(18)
```

- 这样也可以设置刻度值字体大小

```
matplotlib.rc('xtick', labelsizes=20)  
matplotlib.rc('ytick', labelsizes=20)
```

# 一个例子



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

```
plt.subplots_adjust(hspace=0.4)
t = np.arange(0.01, 20.0, 0.01)
```

```
# log y axis
plt.subplot(221)
plt.semilogy(t, np.exp(-t/5.0))
plt.title('semilogy')
plt.grid(True)
```

```
# log x axis
plt.subplot(222)
plt.semilogx(t, np.sin(2*np.pi*t))
plt.title('semilogx')
plt.grid(True)
```

```
# log x and y axis
plt.subplot(223)
plt.loglog(t, 20*np.exp(-t/10.0), basex=2)
plt.grid(True)
plt.title('loglog base 4 on x')
```

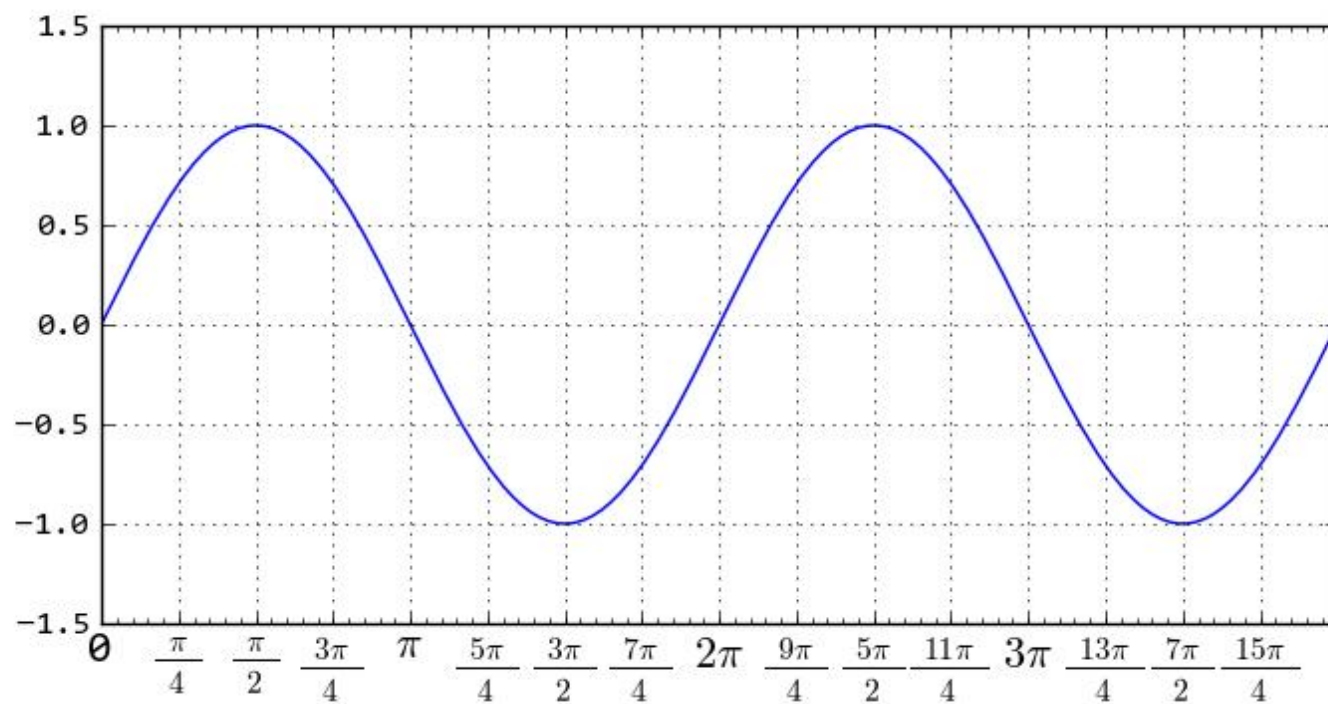
```
# with errorbars: clip non-positive values
ax = plt.subplot(224)
ax.set_xscale("log", nonposx='clip')
ax.set_yscale("log", nonposy='clip')
```

```
x = 10.0**np.linspace(0.0, 2.0, 20)
y = x**2.0
plt.errorbar(x, y, xerr=0.1*x, yerr=5.0+0.75*y)
ax.set_ylim(ymin=0.1)
ax.set_title('Errorbars go negative')
```

```
plt.show()
```

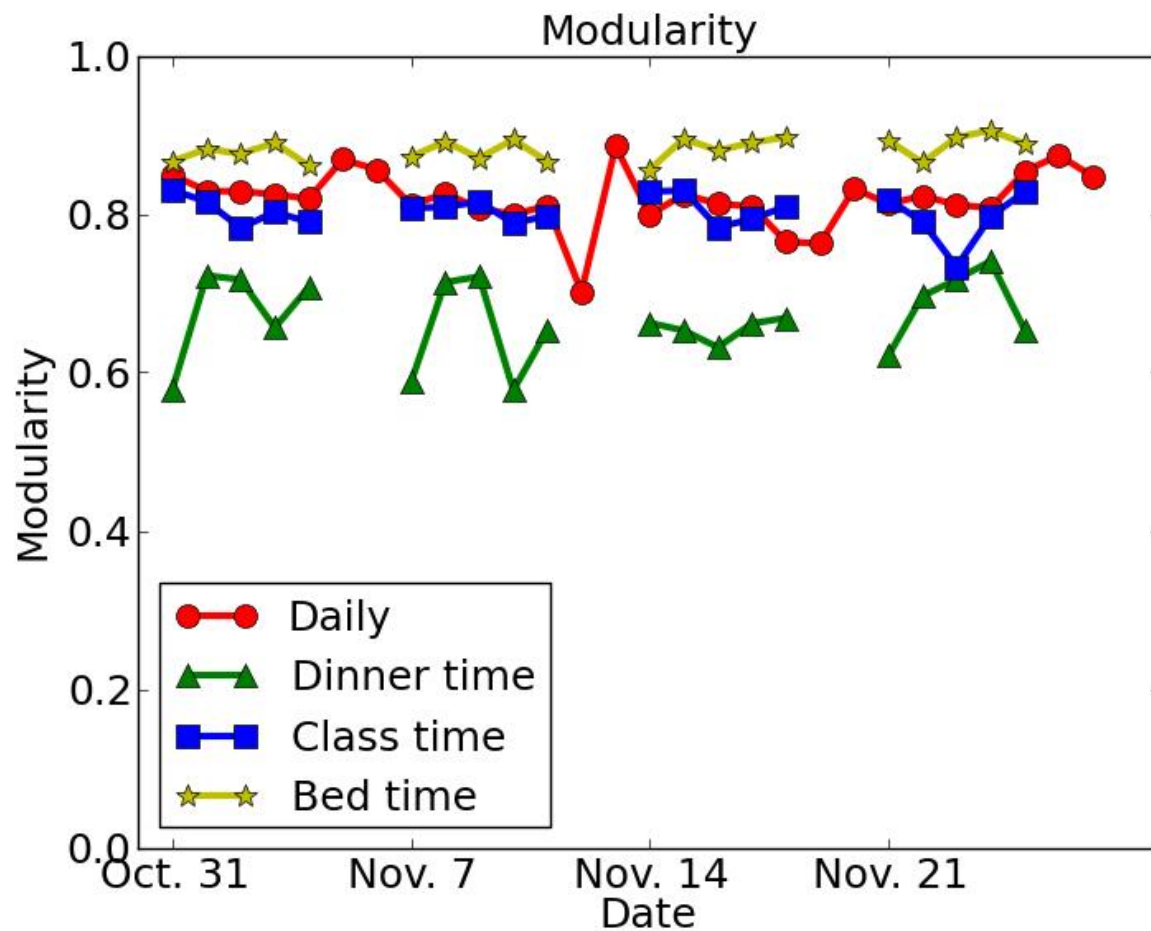


# 一个例子



See final\_example02.py

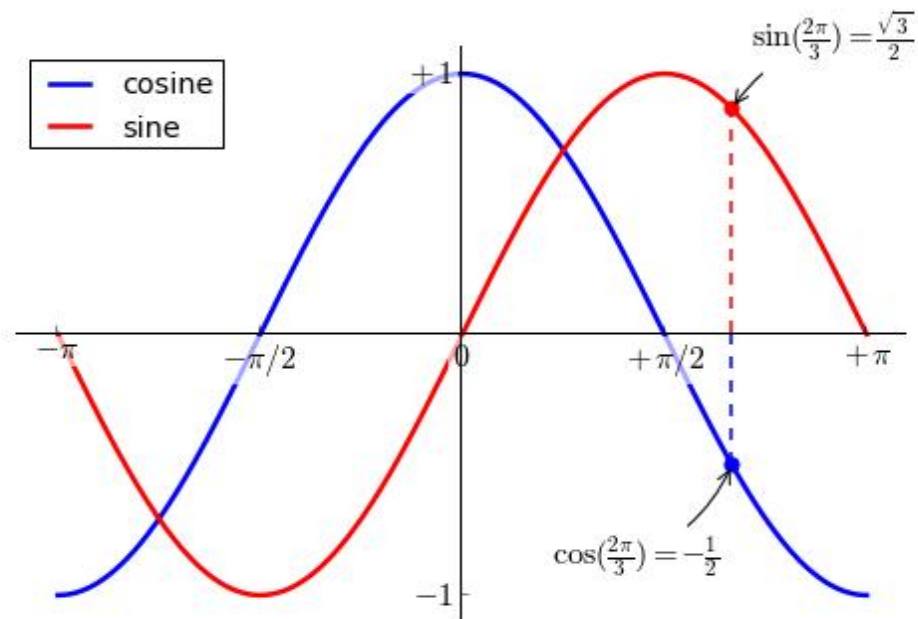
# 一个例子



See final\_example03.py

# 用matplotlib画图的第三种方法

用pylab模块快速画图  
(与第一种方法类似)



See final\_example04.py (用的是pylab)

详细过程:

<http://www.labri.fr/perso/nrougier/teaching/matplotlib/#beyond-this-tutorial>

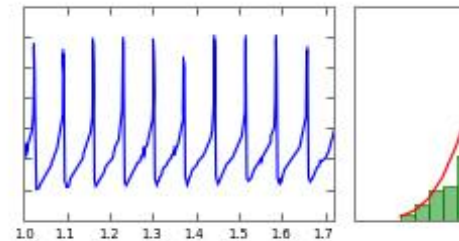
# 参考建议



[home](#) [examples](#) [gallery](#) | [pyplot](#) | [docs](#) »

## Introduction

matplotlib is a python 2D plotting library which produces publication quality figures in a variety of shell (ala MATLAB<sup>®</sup> or Mathematica<sup>®†</sup>), web application servers, and six graphical user interfaces.



matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, [thumbnail gallery](#), and [examples](#) directory.

For simple plotting the `pyplot` interface provides a MATLAB-like interface, particularly when combined with the `Figure` interface or via a set of functions familiar to MATLAB users.