Matplotlib

Pacote para fazer desenhos e manipulações de imagens em Python. Ele é muito extenso, tem 2303 páginas e sua última versão deve ser buscada na internet via `matplotlib.pdf`.

Instalação Tendo acesso à Internet, entre no diretório de instalação do Python, no subdiretório `Scripts` dentro da árvore de instalação do Python. Abrindo uma janela CMD nesse diretório execute

```
pip install matplotlib
```

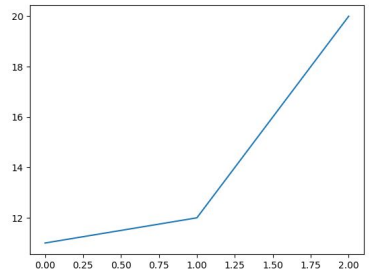
Se não souber fazer isto ou não tiver direitos na máquina em questão, mude de estratégia e use o Winpython. Este pacote bem como muitos outros, já está pré-instalado no `winpython`.

Importação Há duas maneiras de importar. A mais simples é `>>> from matplotlib import *` Aqui todos os objetos do pacote são incluidos no espaço de nomes atual. A segunda maneira, mais higiênica, é atribuir um nome ao pacote e referenciar seus objetos qualificando-os por este nome. O comando agora é `>>> import matplotlib.pyplot as plt`. O nome do pacote agora é `plt`. Pode-se usar qualquer outro nome, mas este material (e 99% da comunidade Python) o conhece como plt. Não esqueça de importar também o pacote numérico (`import numpy as np`).

Modos de uso Há 2 modos de uso do pacote: usar a interface procedural `pyplot` ou usar a API nativa da matplotlib, esta mais orientada a objetos. A primeira é constituída por métodos (como xlabel, legend, etc) e elas podem ser usadas de duas maneiras: se chamadas sem argumentos, devolvem o valor atual do parâmetro e se se chamadas com parâmetros elas atualizam este valor.

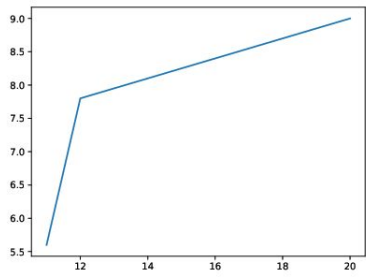
Plot Cria gráficos `xy`. Crie as variáveis a plotar. Se for uma única série, ela será considerada como y (e neste caso x serão seus índices).

```
import numpy as np
import matplotlib.pyplot as plt
a=[11,12,20]
lixo=plt.plot(a)
plt.show()
```



Ou se quiser crie um conjunto de valores x, y como dois vetores.

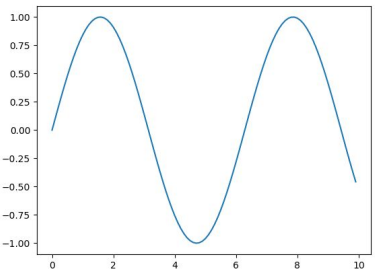
```
a=[11,12,20]
b=[5.6, 7.8, 9.0]
lixo=plt.plot(a,b)
```



Diversas curvas podem ser desenhadas no mesmo gráfico, bastando chamar o plot com uma matriz (cada coluna será uma curva) ou chamando várias vezes a plot() uma vez para cada coluna e vendo tudo de uma vez com plt.show().

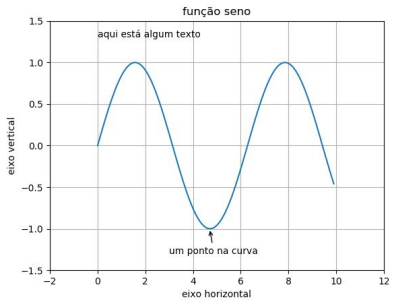
Usando uma função

```
x = np.arange(0.,10.,0.1)
x
array([0. , 0.1, 0.2, 0.3, 0.4, 0.5,
  0.6, 0.7, 0.8, 0.9, 1. , 1.1, 1.2,
  1.3, 1.4, 1.5, 1.6, 1.7, 1.8,
  ... , 9.9])
y = np.sin(x)
ll = plt.plot(x,y)
plt.show()
```



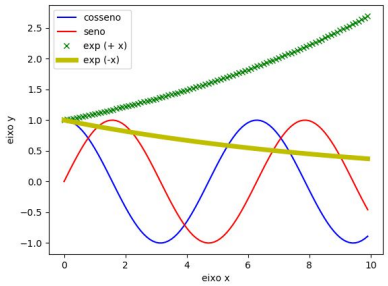
Melhorando o visual

```
xl=plt.xlabel('eixo horizontal')
yl=plt.ylabel('eixo vertical')
ttl=plt.title('função seno')
ax=plt.axis([-2,12,-1.5,1.5])
grd=plt.grid(True)
txt=plt.text(0,1.3,'aqui está algum texto')
ann=plt.annotate('um ponto na curva',
 xy=(4.7,-1),xytext=(3, -1.3),
 arrowprops=dict(arrowstyle='->'))
```



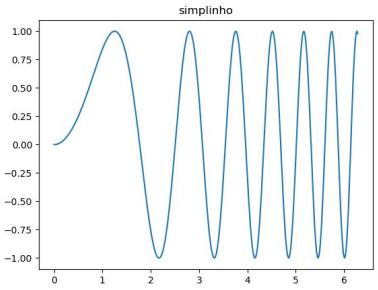
Mais

```
x=np.arange(0.,10,0.1)
a=np.cos(x)
b=np.sin(x)
c=np.exp(x/10)
d=np.exp(-x/10)
la=plt.plot(x,a,'b-',label='cosseno')
lb=plt.plot(x,b,'r-',label='seno')
lc=plt.plot(x,c,'gx',label='exp(+x)')
ld=plt.plot(x,d,'y-',linewidth=5,
   label='exp(-x)')
ll=plt.legend((loc='upper left'))
lx=plt.xlabel('eixo x')
ly=plt.ylabel('eixo y')
plt.show ()
```

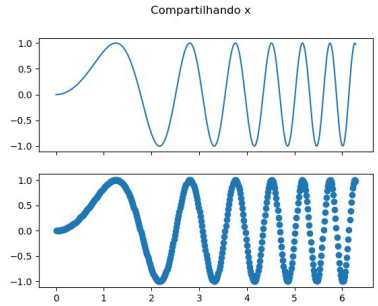


subplots Permite colocar diversos gráficos no mesmo desenho. O formato da chamada é `subplots(lin,col,sharex=T/F,sharey=T/F)`. PS: scatter é para desenhar o ponto sozinho, sem ligação. Acompanhe

```
x=np.linspace(0, 2*np.pi, 400)
y=np.sin(x**2)
fig,ax=plt.subplots()
ax.plot(x,y)
ax.set_title('simplinho')
plt.show()
```




```
...
f,axarr=plt.subplots(2, sharex=True)
adj = plt.subplots_adjust(hspace=0.4,wspace=0.4) #
f.suptitle('Compartilhando x')
axarr[0].plot(x,y)
axarr[1].scatter(x,y)
plt.show()
```



Obviamente se o conjunto de subplots for (2,2), a indexação sera [0,0], depois [0,1] depois [1,0] e assim por diante.

Tortas Eis o formato do comando `pie`

```
matplotlib.pyplot.pie(x, explode=None,
labels=None, colors=None, autopct=None,
pct-distance=0.6, shadow=False,
labeldistance=1.1, startangle=None,
radius=None, counterclock=True,
```



```
wedgeprops=None, textprops=None,
center=(0, 0), frame=False,
rotatelabels=False,hold=None,data=None)
```

As fatias estão dadas por `x`. Cada fatia ocupa $x/sum(x)$. Se a soma de x é menor do que 1, os valores de x indicam a fração e não são normalizados. As fatias são desenhadas em sentido anti-horário a partir do eixo x.

explode O parâmetro *explode* se não for *None* será um array de tamanho $len(x)$ indicando qual a fração do raio que cada fatia deve ser deslocada.

labels é uma sequencia de strings indicando o que é cada fatia.

colors é uma sequencia de cores. Se usado *None* usa-se a sequencia padrão.

autopct Pode ser *None*, um string ou uma função. Diz para identificar a fatia com um número. Se for uma função ela será chamada. A *pctdistance* indica onde por este rótulo.

shadow é um booleano, indica se haverá sombra.

labeldistance o default é 1.1 e indica a distância radial onde os labels vão ser desenhados.

startangle onde começa a primeira fatia (a partir do eixo x).

radius O raio da pizza. Se *None* vale 1.

counterclock Booleano, indica o sentido antihorário ou não.

wedgeprops é um dicionário (como em `wedgeprops = {'linewidth': 3}` . Para saber mais veja a lista de argumentos do objeto `wedge`.

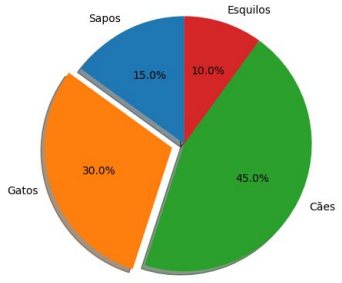
textprops é um dicionário com atributos de texto.

center a localização do centro da pizza (default 0,0)

frame Booleano. Se true desenha um frame.

rotatelabels Booleano: rotaciona cada fatia no ângulo da fatia.

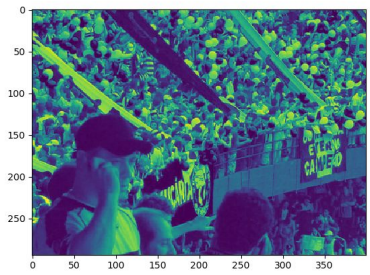
```
labels='Sapos','Gatos','Cães','Esquilos'
sizes = [15, 30, 45, 10]
explode = (0, 0.1, 0, 0)
fig1, ax1 = plt.subplots()
ax1.pie(sizes,explode=explode,
 labels=labels,autopct='%1.1f%%',
 shadow=True,startangle=90)
ax1.axis('equal')
plt.show()
```



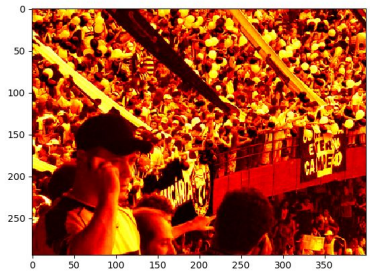
Brincando com imagens

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.image as mpimg
img=mpimg.imread('c:/p/python/coxa.png')
#print(img.shape) é (294, 400, 3)
imgplot=plt.imshow(img)
plt.show(imgplot)
```

```
lum_img=img[:,:,0]
imgplot=plt.imshow(lum_img)
plt.show(imgplot)
```



```
im22=plt.imshow(lum_img,cmap='hot')
plt.show(im22)
```



Diferença entre subplot e subplots

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(1234567)
data=np.random.randn(2,200)
  #array 2L x 200C # dp=1 e media 0
fig,ax=plt.subplots(2,2)
ax[0,0].hist(data[0])
ax[0,0].set(ylabel='eixo y')
ax[1,0].scatter(data[0],data[1])
ax[0,1].plot(data[0],data[1])
ax[1,1].hist2d(data[0],data[1])
plt.show()

fig2=plt.figure(1)
sp=plt.subplot(221)
l1=plt.hist(data[0])
ly=plt.ylabel("eixo y")
sp=plt.subplot(222)
l1=plt.scatter(data[0],data[1])
sp=plt.subplot(223)
l1=plt.plot(data[0],data[1])
sp=plt.subplot(224)
l1=plt.hist2d(data[0],data[1])
plt.show()
```

Exemplo Completo

```
# caogato
import matplotlib.pyplot as plt
import numpy as np
def caogato():
 ca=[23,34,24,28,67,54,76,99,23,26,66,55]
 ga=[11,28,26,39,56,77,81,89,56,66,45,69]
 plt.plot(ca,'r-',label='cães')
 plt.plot(ga,'g-',label='gatos')
 plt.legend()
 plt.title('Cães e gatos em Curitiba')
 plt.xlabel('ao longo dos meses')
 plt.ylabel('milhares de animais')
 plt.axis([0,11,0,120])
 plt.xticks(np.arange(12),('jan','fev',
 'mar','abr','mai','jun','jul','ago',
 'set','out','nov','dez'),rotation=30)
 plt.text(5,100,'mês de\n cachorro louco')
 plt.grid()
 plt.savefig('c:/p/python/caogato.png')
 plt.show()
caogato()
```