Python

Introdução à Linguagem

Prof. Luiz Fernando Carvalho

luizfcarvalho@utfpr.edu.br

Sumário

- Visão geral das bibliotecas usadas em ciência de dados:
  - numpy
  - matplotlib;
- Exemplos de uso.

Numerical Python (Numpy)

- Base para a computação numérica no Python;
- Permite a resolução de problemas matemáticos e científicos;
- Permite o uso facilitado de vetores e matrizes;
- Evita uso de laços de repetição...

```
import numpy as np

vet = np.array([1, 2, 3, 4, 5])

print(vet)
print(type(vet))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

Numerical Python (Numpy)

```
import numpy as np

matriz = np.array([[1, 2], [3, 4]])
print(matriz)
```

```
[[1 2]
 [3 4]]
```

- matriz.shape
- matriz.size
- matriz.ndim

Numerical Python (Numpy)

```
import numpy as np

matriz = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(matriz)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Slicing

```
matriz[0, 1]    matriz[:, 1]    matriz[0:2, 1]    matriz[-1,:]

matriz[2, 0]    matriz[0,:]    matriz[2,1:2]    matriz[0:,-2]
```

Numerical Python (Numpy)

```
import numpy as np

matriz_zeros = np.zeros([2, 3], dtype=int)
print(matriz_zeros)
```

```
[[0 0 0]
 [0 0 0]]
```

```
matriz_uns = np.ones(3)
print(matriz_uns)
```

```
[1 1 1]
```

```
matriz_identidade = np.eye(3)
print(matriz_identidade)
```

```
[[1 0 0]
 [0 1 0]
 [0 0 1]]
```

Numerical Python (Numpy)

```
arange(inicio=0, fim, intervalo=1, dtype=int)
```

```
import numpy as np

vet = np.arange(100)
vet2 = np.arange(0, 100, 2)
vet3 = np.arange(50, 0, -1)
```

Qual o conteúdo de vet, vet2 e vet3?

Numerical Python (Numpy)

```
np.random.randint(inicio=0, fim, tamanho)
```

```
import numpy as np

vetor_aleatorio = np.random.randint(0, 100, 10)
matriz_aleatoria = np.random.randint(0, 100, (3, 3))
```

Numerical Python (Numpy)

```
import numpy as np

def vetor(vet):
    for i in range(10):
        vet2 = vet * 2

def lista(l):
    for i in range(10):
        lista2 = [x*2 for x in l]

vet = np.arange(1000000)
l = list(range(1000000))
```

Numpy é mais rápido que listas!

No terminal Ipython

```
%timeit vetor(vet)
28.8 ms ± 770 µs per loop (mean ± std. dev. of 7 runs, 10 loops each)

%timeit lista(l)
1.21 s ± 25.5 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

Numerical Python (Numpy)

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad B = \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

a) 2 * B

b) A + B

c) A - B

d) A * B

d) Valor máximo de A

e) Valor mínimo de B

https://matplotlib.org/gallery.html

Matplolib

- Gráficos simples...

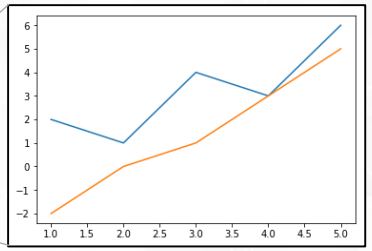
```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 x = np.array([1, 2, 3, 4, 5])
5 y = np.array([2, 1, 4, 3, 6])
6
7 plt.plot(x, y)
8 plt.show()
```

Troque o comando `plot` da linha 7 por `scatter` e depois para `bar`

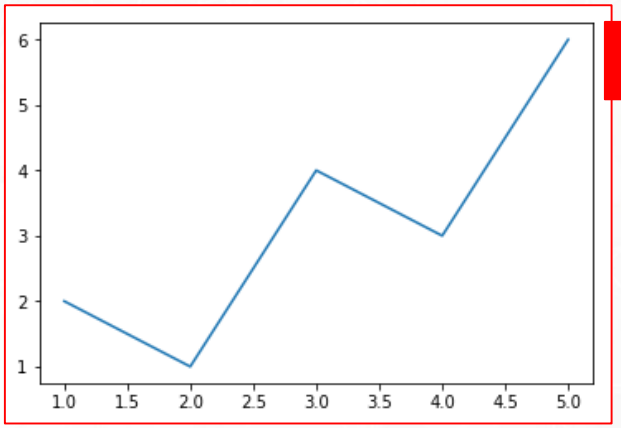
Matplolib

- Duas curvas dentro do mesmo gráficos simples...

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  x = np.array([1, 2, 3, 4, 5])
5  y = np.array([2, 1, 4, 3, 6])
6  y2 = np.array([-2, 0, 1, 3, 5])
7
8  plt.plot(x, y) #plota a primeira curva
9  plt.plot(x, y2) #plota a segunda curva
10 plt.show()
```

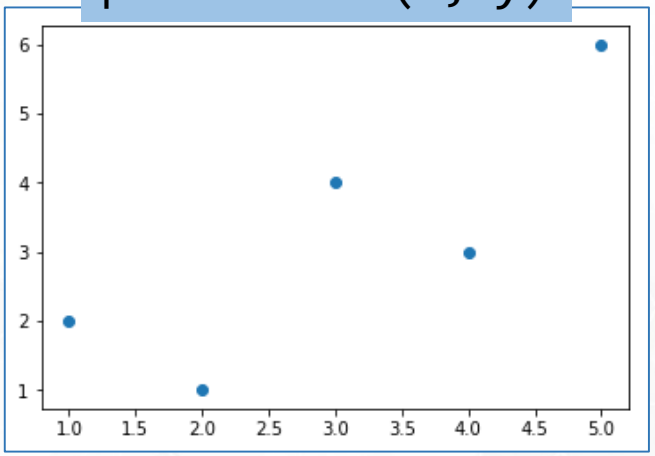


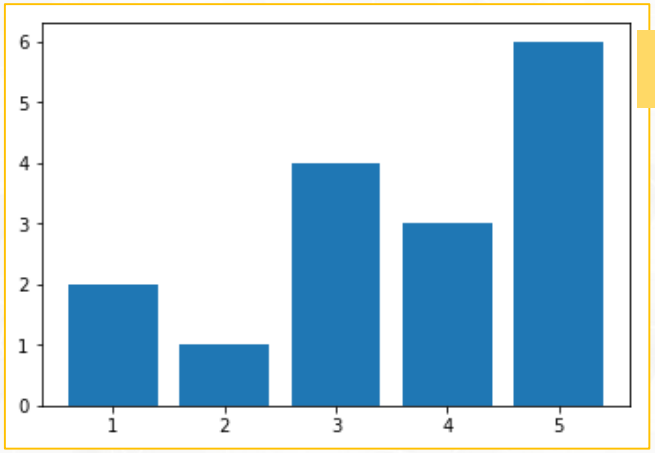
Matplolib




```
plt.plot(x, y)
```

```
plt.scatter(x, y)
```



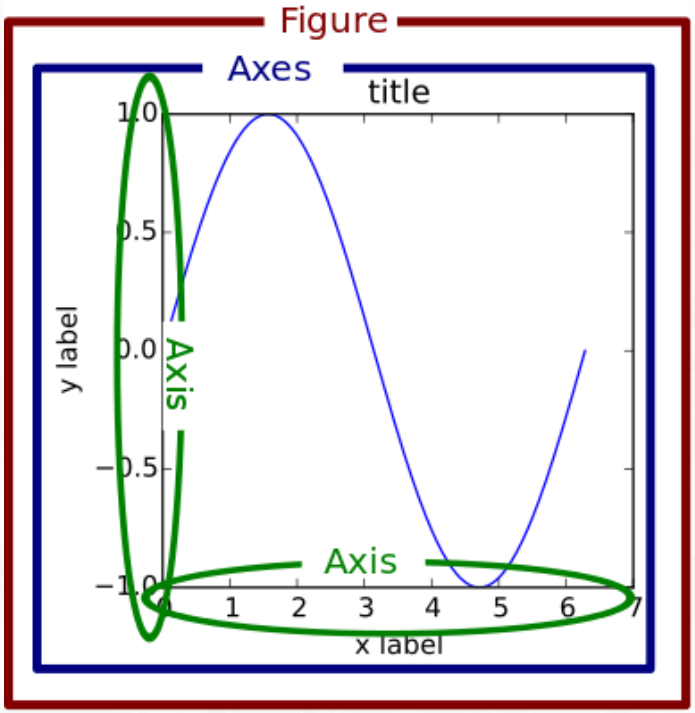



```
plt.bar(x, y)
```

```
x = np.array([1, 2, 3, 4, 5])
y = np.array([2, 1, 4, 3, 6])
```

Matplolib

- **Figure:** é o espaço onde tudo é desenhado para composição da imagem;
- **Axes:** Podem ser adicionados ao Figure. Axes é uma área em que os dados são plotados e tem rótulos, eixos, etc. relacionados a eles



Matplolib

- Sintaxe básica:

Apenas um axes

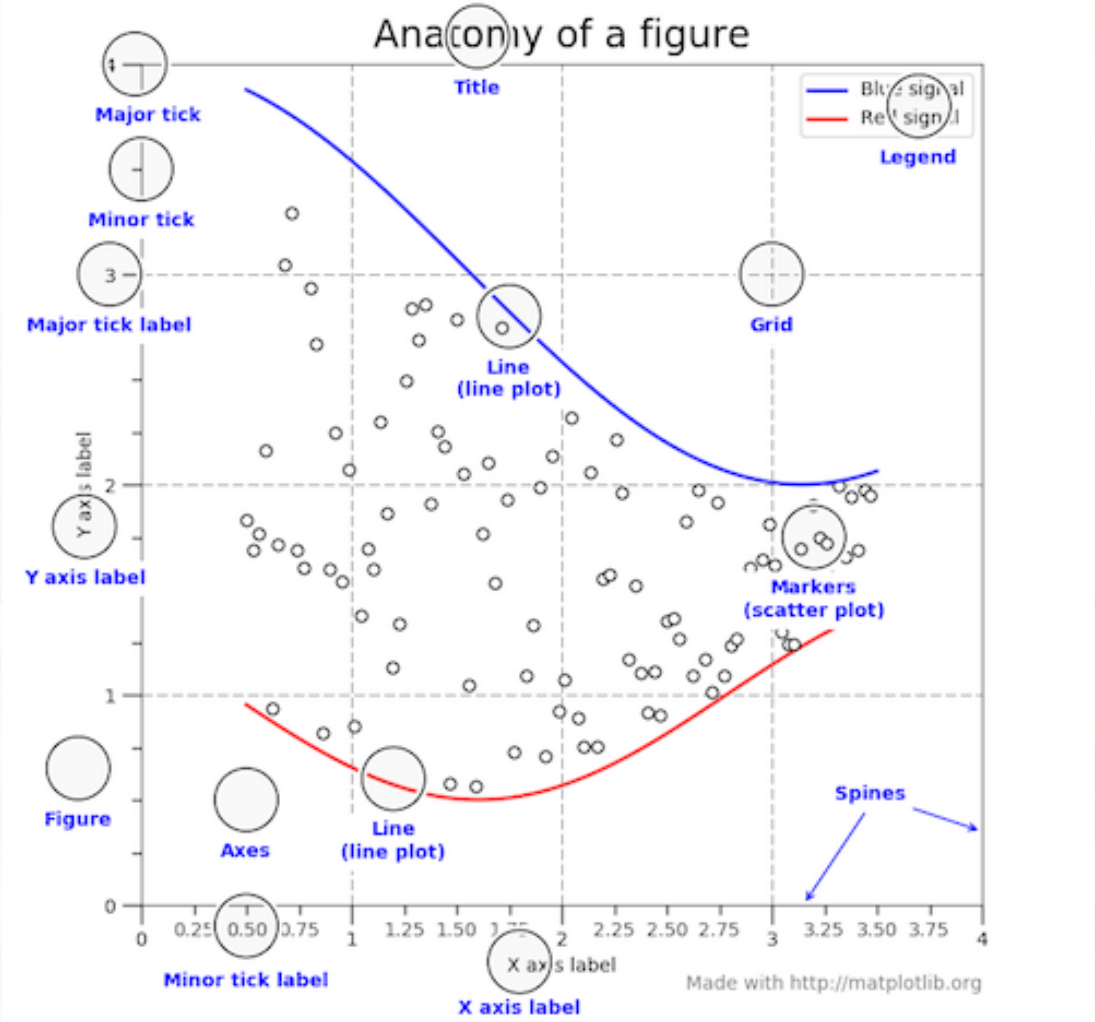
```
fig, ax = plt.subplots()
```

fig é onde ficará o axes

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  x = np.array([1, 2, 3, 4, 5])
5  y = np.array([2, 1, 4, 3, 6])
6
7  fig, ax = plt.subplots()
8  ax.plot(x, y)
9  plt.show()
```

Matplolib

Anatomia de um gráfico do matplotlib



Matplolib

- Melhorando o gráfico:

```
fig, ax = plt.subplots()
```

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  x = np.array([1, 2, 3, 4, 5])
5  y = np.array([2, 1, 4, 3, 6])
6
7  fig, ax = plt.subplots(figsize=(8, 5))
8  ax.plot(x, y)
9  ax.set_xlabel('Eixo X')
10 ax.set_ylabel('Eixo Y')
11 plt.show()
```

O tamanho da figura é dado em polegadas (largura, altura)

Matplolib

- Modificando o estilo e cor da linha

Caractere	Descrição
‘--’	Linha tracejada
‘-.’	Tracejada com pontilhado
‘:’	Pontilhada
‘.’	Ponto
‘o’	Círculo
‘s’	Quadrado
‘^’ ou ‘<‘ ou ‘^’ ou ‘v’	Triângulo
‘h’	Hexágono
‘*’	estrela

Caractere	Cor
‘b’	Blue
‘g’	Green
‘r’	red
‘k’	Black
‘y’	yellow

```
ax.plot(x, y, ‘g--’)
```

```
ax.plot(x, y, ‘r*’)
```

```
ax.plot(x, y, ‘k.’)
```

https://matplotlib.org/2.1.1/api/_as_gen/matplotlib.pyplot.plot.html

Matplolib

- Melhorando o gráfico:

```
 1  import numpy as np
 2  import matplotlib.pyplot as plt
 3
 4  x = np.array([1, 2, 3, 4, 5])
 5  y = np.array([2, 1, 4, 3, 6])
 6
 7  fig, ax = plt.subplots(figsize=(8, 5))
 8  ax.plot(x, y, 'g--', label='Crescimento')
 9  ax.set_xlabel('Eixo X')
10  ax.set_ylabel('Eixo Y')
11  ax.grid() #habilita o grid do gráfico
12  ax.legend() #adiciona a legenda
13  plt.show()
```

Cada curva dentro de um axes pode ter um `label`. Esse `label` será usado para geração da legenda

Matplolib

- Dois axes na mesma figura

`ncols` é o número de colunas que a `fig` vai ter

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  x = np.arange(0, 50)
5  y = np.random.randint(100, size=50)
6  y2 = np.random.randint(20, size=50)
7
8  fig, (ax, ax2) = plt.subplots(figsize=(8, 5), ncols=2, nrows=1)
9  ax.plot(x, y, 'g--', label='Crescimento')
10 ax.set_xlabel('Eixo X')
11 ax.set_ylabel('Eixo Y')
12 ax.grid() #habilita o grid do gráfico
13 ax.legend() #adiciona a legenda
14
15 ax2.scatter(x, y2, label = 'Pontos')
16 ax2.legend() #adiciona legenda do ax2
17 plt.show()
```

Matplolib

- Salvar a figura

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  x = np.arange(0, 50)
5  y = np.random.randint(100, size=50)
6  y2 = np.random.randint(20, size=50)
7
8  fig, (ax, ax2) = plt.subplots(figsize=(8, 5), ncols=2, nrows=1)
9  ax.plot(x, y, 'g--', label='Crescimento')
10 ax2.scatter(x, y2, label = 'Pontos')
11 plt.show() #não usamos o  show quando vamos salvar
12 plt.tight_layout() #tira espaços extras na borda da imagem
13 plt.savefig('graficos.png')
```