FURB
UNIVERSIDADE DE BLUMENAU

Introdução à Computação Científica com Python

Leonardo K. Kewitz
github.com/kewitz

FURB
UNIVERSIDADE DE BLUMENAU

Por que Python?

- Fácil de ler, fácil de aprender.
- Alto nível.
- Interpretada e dinâmica.
  - Programação iterativa e prototipagem rápida.
- Procedural, Orientada ou Funcional.
- De uso geral mas que possui bom desempenho para computação numérica.
- Facilmente integrável. (C, C++, Fortran)
- Open Source.

FURB
UNIVERSIDADE DE BLUMENAU

Spyder

- Scientific PYthon Development EnviRonment.
- IDE Python semelhante ao MATLAB.
- Desenvolvido em Python e Qt.
- Por padrão já contém diversas bibliotecas para computação científica:
  - Scipy
  - Numpy
  - Matplotlib

FURB
UNIVERSIDADE DE BLUMENAU

Python vs. C

```python
import numpy
msg = "Hello World!"
print msg

A = 20

if A > 10:
     print “Maior que 10.”

for i in range(10):
     print i

def greet(name):
    print "Hello %s!" % name


greet("World")
```

```cpp
#include <string>
string msg = "Hello World!";
cout << msg;

int A = 20;

if (A > 10) {
     cout << “Maior que 10.”;
}

for(int i = 0; i < 10; i++) {
     cout << i;
}

void greet(string name){
     cout << “Hello ” << name << “!”;
}

greet(“World”);
```

Números

```
>>> type(3)
<type 'int'>
>>> type(3.0)
<type 'float'>
>>> type((3+1j))
<type 'complex'>
>>> 3/5
0
>>> 3.0/5
0.6
>>> 3/5.0
0.6
>>> 3.0//5.0
0.0
>>> (3+1j) / 2
```

FURB
UNIVERSIDADE DE BLUMENAU

Vetores

```
>>> v = range(10)
>>> print v
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> v[:4]
[0, 1, 2, 3]
>>> v[4:]
[4, 5, 6, 7, 8, 9]
>>> v[:-2]
[0, 1, 2, 3, 4, 5, 6, 7]
>>> len(v)
10
>>> v[-1]
9
```

FURB
UNIVERSIDADE DE BLUMENAU

Funcional

```
>>> print v
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> [i**2 for i in v]
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
>>> [i for i in v if i > 5]
[6, 7, 8, 9]
>>> [(x, y) for x in [1,2,3] for y in [3,1,4] if x != y]
[(1, 3), (1, 4), (2, 3), (2, 1), (2, 4), (3, 1), (3, 4)]
>>> a, b = 2, 1
>>> a + b
3
```

FURB
UNIVERSIDADE DE BLUMENAU

Numpy

- Biblioteca numérica para Python.
- Introduz objetos de vetor e matriz melhores que os nativos do Python.
- Ferramentas para facilitar integração com outras linguagens.
- Inclui funções para álgebra linear, transformadas de Fourier e outras.

FURB
UNIVERSIDADE DE BLUMENAU

Numpy

```
>>> from numpy import *
>>> arange(5)
array([0, 1, 2, 3, 4])
>>> arange(5)**2
array([ 0,  1,  4,  9, 16])
>>> v = linspace(0, pi, 4).reshape((2,2))
>>> print v
[[ 0.         ,  1.04719755],
 [ 2.0943951 ,  3.14159265]]
>>> print v[:,0]
[ 0.          2.0943951]
>>> cross([1,0,0],[0,1,0])
array([0, 0, 1])
```

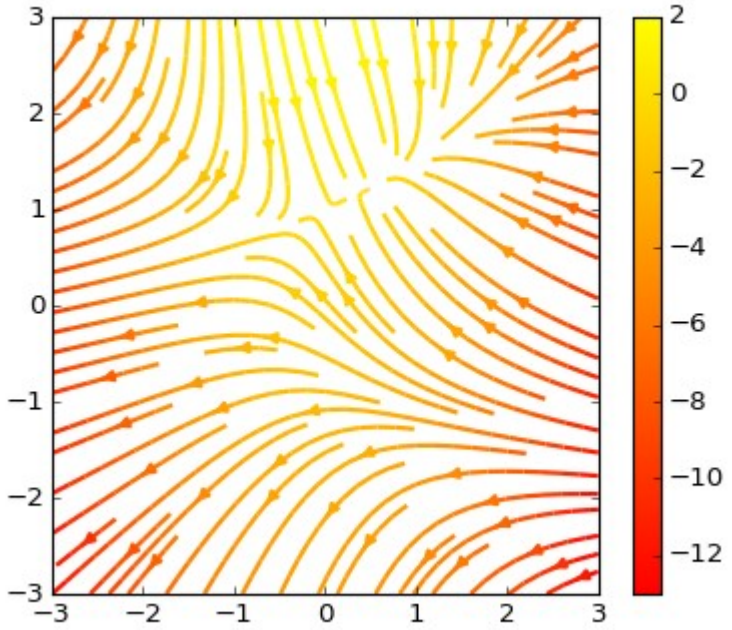
Exemplo #1

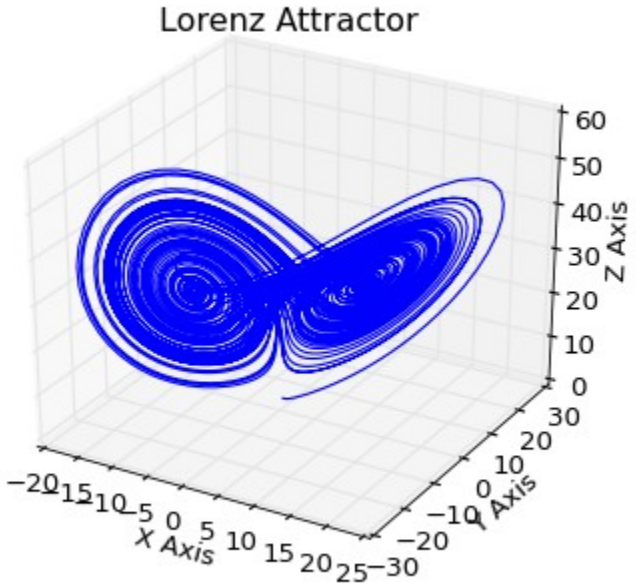
- Criar função `fib(n)` que calcula a série de Fibonacci até `n`.

```
def fib(n):
    ...
    while i < n:
        …
        print i
```

Matplotlib

- Biblioteca para plotagem 2D.
- Suporta animações.





Exemplo #2

- Criar um vetor tempo de 0 à 2*Pi.
- Plotar sin(t) e cos(t).

```
from numpy import *
import matplotlib.pyplot as plt

t = linspace(...)
plt.plot(...)
```

Exemplo #3

- Plotar u(t)*sin(t) para -pi <= t <= 2pi
- Implementar função lambda u(t).
  - u(t) = 0 para t < 0; 1 para t >= 0