Usando Numpy arrays para realizar operações matemáticas em Python

Esse material irá tratar de:

- Matemática básica com Numpy arrays 1D
- Criação de arrays com arange e linspace
- Comparação de performance (velocidade de processamento) com listas

Criando uma lista, trabalhando com listas e arrays

In [12]:

```
import numpy as np
```

In [13]:

```
a = [1,2,3]
```

In [14]:

```
a
```

Out[14]:

```
[1, 2, 3]
```

In [15]:

```
a + a
```

Out[15]:

```
[1, 2, 3, 1, 2, 3]
```

In [17]:

```
b = np.array(a)
```

In [18]:

```
b + b
```

Out[18]:

```
array([2, 4,
6])
```

In [19]:

```
b + 1
```

Out[19]:

```
array([2, 3,
4])
```

In [20]:

```
b**2
```

Out[20]:

```
array([1, 4,
9])
```

In [22]:

```
np.sin(b)
```

Out[22]:

```
array([ 0.84147098,  0.90929743,
0.14112001])
```

Criando listas com funções básicas do python

In [25]:

```
range(5)
```

Out[25]:

```
range(0, 5)
```

Criando arrays com Numpy

In [28]:

```
np.arange(5)
```

Out[28]:

```
array([0, 1, 2, 3,
4])
```

Criando arrays com valores inicial e final e preenchido com elementos igualmente espaçados

In [30]:

```
np.linspace(0,5,11)
```

Out[30]:

```
array([ 0. ,  0.5,  1. ,  1.5,  2. ,  2.5,  3. ,  3.5,  4. ,  4.5,  5. ]
)
```

In [31]:

```
len(np.linspace(0,5,11))
```

Out[31]:

```
11
```

In [32]:

```
a
```

Out[32]:

```
[1, 2, 3]
```

In [33]:

```
aa = [x + y for x,y in zip(a,a)]
```

In [34]:

```
aa
```

Out[34]:

```
[2, 4, 6]
```

In [35]:

```
bb = b + b
```

In [36]:

```
bb
```

Out[36]:

```
array([2, 4,
6])
```

Função legal para medir performance de códigos (timeit)

In [37]:

```
a = range(1000)
```

In [38]:

```
timeit aa = [x + y for x,y in zip(a,a)]
```

```
1000 loops, best of 3: 180 µs per
loop
```

In [40]:

```
b = np.array(a)
```

In [41]:

```
timeit bb = b + b
```

```
The slowest run took 16.14 times longer than the fastest. This could mean t
hat an intermediate result is being cached
100000 loops, best of 3: 2.45 µs per loop
```

In [42]:

```
180/2.45
```

Out[42]:

```
73.46938775510203
```

por isso é melhor usar numpy com vetores grandes

In [ ]: