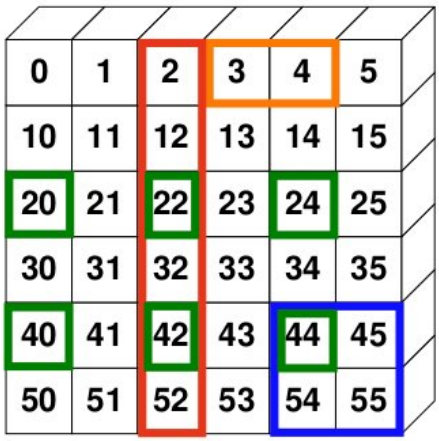
Tutorial Básico de numpy



NumPy é um pacote de Python que suporta operações com vetores e matrizes e é essencial para a computação científica com Python. O NumPy é baseado em C, portanto tem um desempenho superior se comparado às operações com vetores originais do Python. Neste post eu quero mostrar uma introdução básica ao NumPy para os iniciantes.

INSTALAÇÃO DO NUMPY

Primeiro vamos instalar o NumPy. No Windows, basta baixar a última versão do numpy no site do SourceForge e instalar facilmente com o arquivo .exe. No Linux, instale o pacote "python-numpy" disponível nos repositórios da sua distribuição ou use o pip:

```
pip install numpy
```

Pronto, agora é só abrir o console Python e importar o pacote:

```
>>> import numpy as np
```

CRIAÇÃO DE UM ARRAY NUMPY

Para criar um array, é bem simples:

```
>>> a = np.array([0,1,2,3,4,5])
>>> a
array([0,1,2,3,4,5])
```

A função *array* do NumPy, recebe uma lista de Python e transforma em um array NumPy. Você pode checar o tipo:

```
>>> type(a)
```

E o tipo dos elementos:

```
>>> a.dtype
```

Para criar matrizes multidimensionais é bem simples também:

```
>>> a = np.array([[0,1,2,3], [4,5,6,7], [8,9,10,11]])
```

A função *arange* é bem parecida com a função *range*, só que retorna um array ao invés de uma lista:

```
>>> x = np.arange(11.)
array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.])
```

É possível definir mais parâmetros pra função *arange*:

```
>>> x = np.arange(10, 30, 5) #(limite inferior, limite superior, passo)
array([10, 15, 20, 25])
```

TAMANHO DO ARRAY

A propriedade *shape* mostra o tamanho de cada dimensão da matriz:

```
>>> a.shape
(3,4)
```

É possível, também, modificar essa propriedade:

```
>>> a.shape = (2,6)
>>> a
array([[ 0, 1, 2, 3, 4, 5],
[ 6, 7, 8, 9, 10, 11]])
```

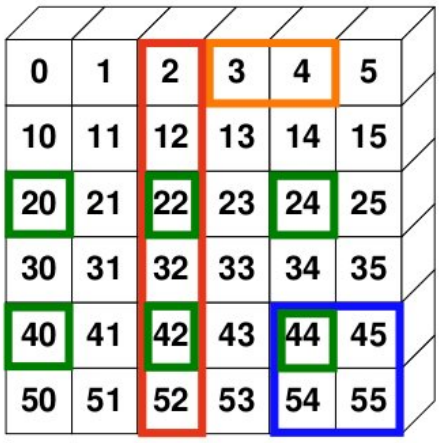
ACESSANDO ELEMENTOS

Para acessar elementos:

```
>>> a[1,3]
9
```

É possível acessar vários elementos ao mesmo tempo:

```
>>>a[0,3:5]
array([3, 4])
>>>a[4:,4:]
array([[44, 45],
[54, 55]])
>>>a[:,2]
array([2,12,22,32,42,52])
>>>a[2::2,::2]
array([[20, 22, 24],
[40, 42, 44]])
```



Fonte: SciPy 2011 - Jonathan Rocher

MATRIZ TRANSPOSTA

Para obter a matriz transposta existem duas formas:

```
>>> a.transpose()
>>> a.T
array([[ 0, 4, 8],
[ 1, 5, 9],
[ 2, 6, 10],
[ 3, 7, 11]])
```

OUTRAS FUNÇÕES IMPORTANTES DO NUMPY

A função *sum* soma todos os elementos do array:

```
>>> np.sum(a)
66
```

Podemos usar o parâmetro axis e determinar em qual eixo queremos a soma:

```
>>> np.sum(a, axis=0)
array([12, 15, 18, 21])
>>> np.sum(a, axis=1)
array([ 6, 22, 38])
```

Alternativamente, podemos usar o método *sum*:

```
>>> a.sum()
66
>>> a.sum(axis=0)
array([12, 15, 18, 21])
```

As funções amin e amax retornam o valor mínimo e o valor máximo do array, respectivamente:

```
>>> b = np.array([3.4, 5., 33., 8.])
>>> np.amin(b)
3.4
>>> np.amax(b)
33.0
```

argmin e *argmax* retornam o índice do menor valor e do maior valor do array, respectivamente:

```
>>> b.argmax()
2
>>> b.argmin()
0
```

O atributo *flat* retorna um iterator que permite acessar elementos de um array multidimensional como se ele fosse uma lista:

```
>>> a = np.array([[0,1,2,3], [4,5,6,7], [8,9,10,11]])
>>> a.flat
>>> a.flat[:]
array([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11])
```