Práctica 0: vectorización

Pedro Antonio González Calero

Vectorización

Vectorización

```
import time
import numpy as np

def dot_product(x1, x2):
    """Calcula el producto escalar con un bucle
    y devuelve el tiempo en milisegundos"""
    tic = time.process_time()
    dot = 0
    for i in range(len(x1)):
        dot += x1[i] * x2[i]
    toc = time.process_time()
    return 1000 * (toc - tic)

def fast_dot_product(x1, x2):
    """Calcula el producto escalar vectorizado
    y devuelve el tiempo en milisegundos"""
    tic = time.process_time()
    dot = np.dot(x1, x2)
    toc = time.process_time()
    return 1000 * (toc - tic)
```

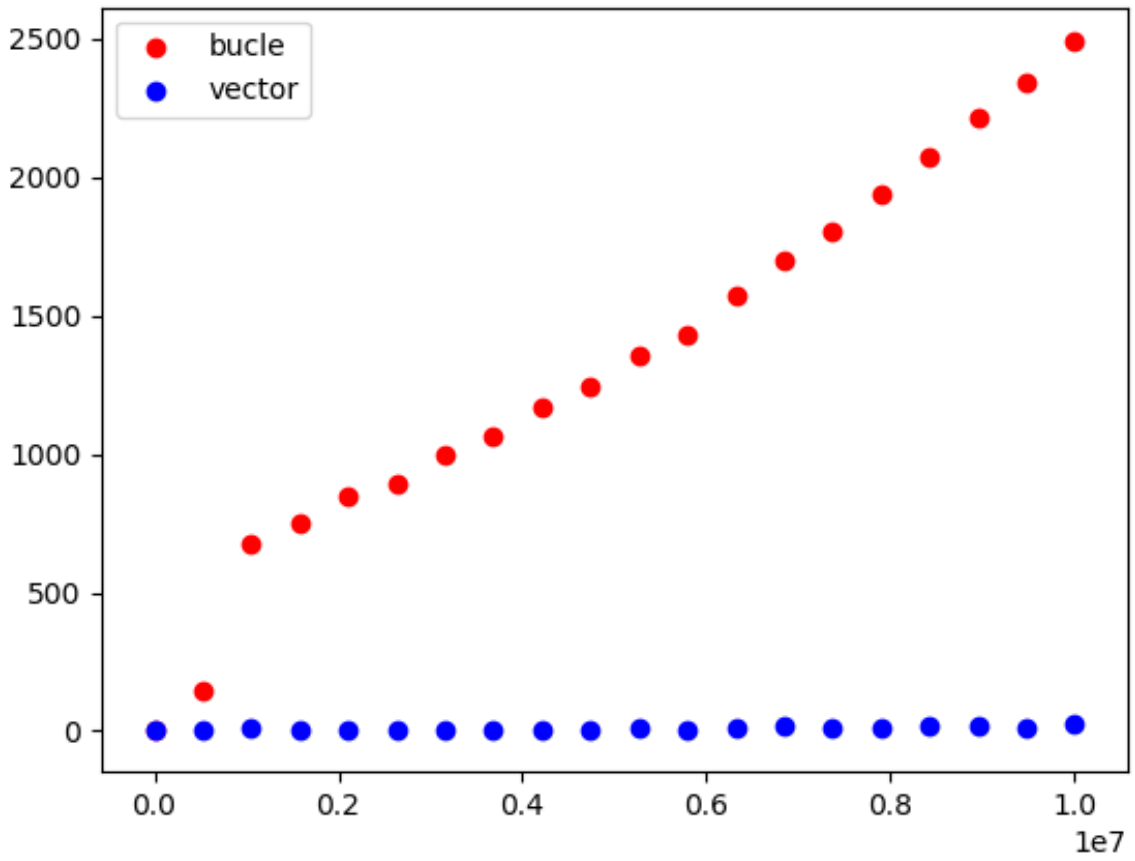
Vectorización

```
import matplotlib.pyplot as plt

def compara_tiempos():
    sizes = np.linspace(100, 10000000, 20)
    times_dot = []
    times_fast = []
    for size in sizes:
        x1 = np.random.uniform(1, 100, int(size))
        x2 = np.random.uniform(1, 100, int(size))
        times_dot += [dot_product(x1, x2)]
        times_fast += [fast_dot_product(x1, x2)]

    plt.figure()
    plt.scatter(sizes, times_dot, c='red', label='bucle')
    plt.scatter(sizes, times_fast, c='blue', label='vector')
    plt.legend()
    plt.savefig('time.png')
```

Vectorización

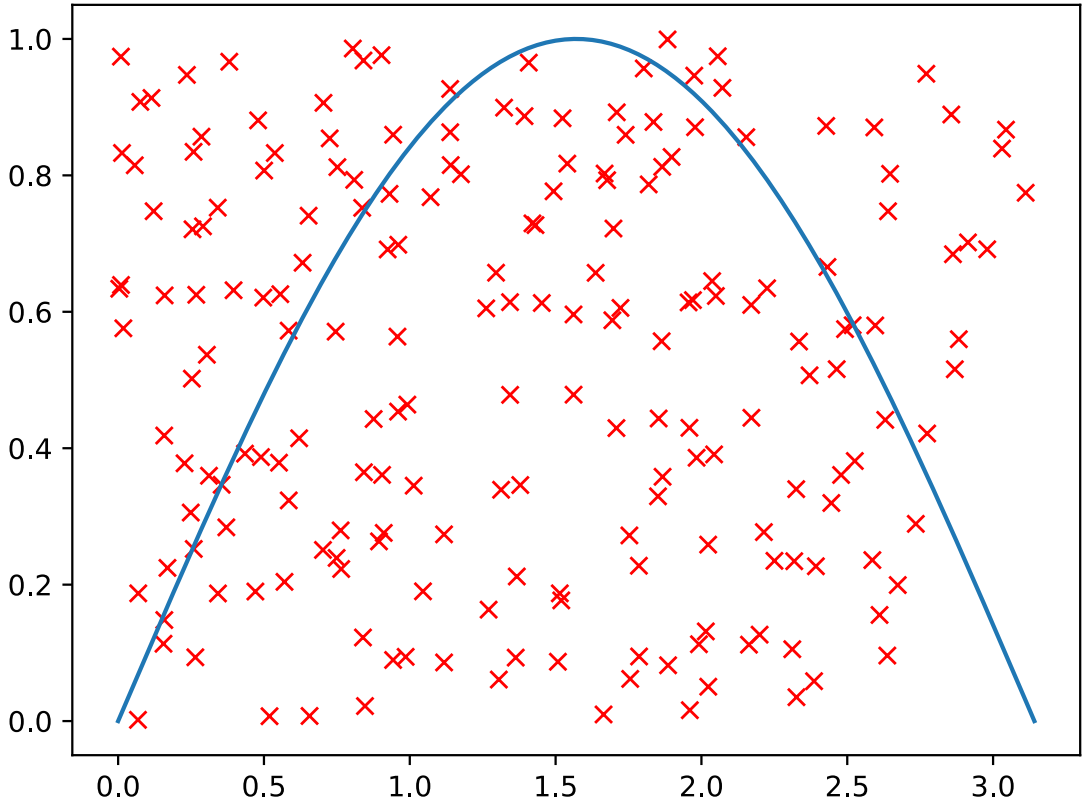


Práctica 0

Integración por el método de Monte Carlo

$$I = \int_a^b f(x)dx = F(b) - F(a)$$

$$I \approx \frac{N_{debajo}}{N_{total}}(b-a)M$$



Pista: cálculo vectorizado

```
In [1]: import numpy as np

In [2]: a = np.array([1,2,3,4,5])
In [4]: b = np.array([3,3,3,3,3])

In [6]: a < b
Out[6]: array([ True,  True, False, False, False])

In [7]: sum(a < b)
Out[7]: 2
```