SBT 645 Introduction to Scientific Computing in Sports Science

#14

De Motu Animalium G.Borelli (1680)

SERDAR ARITAN

serdar.aritan@hacettepe.edu.tr

Biyomekanik Araştırma Grubu
www.biomech.hacettepe.edu.tr
Spor Bilimleri Fakültesi
www.sbt.hacettepe.edu.tr
Hacettepe Universitesi, Ankara, Türkiye
www.hacettepe.edu.tr

Speeding Up Python

When people argue about programming languages, a common critique of Python is, "It's slow." This is occasionally followed by, "A program written in C will run a thousand times faster." Such generalization carry little weight. **Python is often fast enough, and a well-written Python program can run significantly faster than a poorly-written C program.** Plus, Moore's Law implies that computers today are over a thousand times faster than those of 15 years ago: You can do with Python today what was only possible with a highly optimized, compiled program in 2000.

IP[y]: IPython

Profiling

Before proceeding, I offer this advice: If your program already runs fast enough, ***do not bother with profiling and optimization***. There are an endless number of interesting problems waiting to be solved, and the question of how to improve the performance of a particular program by 20 percent is probably not one of them.

The goal of this lecture and its sequel is to provide some tools and tips for improving the performance of Python programs.

How Long Does It Really Take?

To improve the performance of a program, it is useful to gather quantitative data on how long it takes to run. This is called profiling.

- **The `%time` Command**

`%time` is a basic stopwatch. It will tell you how much time elapses on your computer's internal clock while a command is executed.

How Long Does It Really Take?

```
%time 2**100
Wall time: 0 ns
Out[1]: 1267650600228229401496703205376

%time pow(2, 100)
Wall time: 0 ns
Out[2]: 1267650600228229401496703205376
```

Run the commands above several times. You may notice minor differences in the elapsed times, as well as a few significant variations. To get an accurate measure of performance, it is best to average over many repetitions of the same command.

How Long Does It Really Take?

The %timeit Command

Try the same operations as before, but use the %timeit command instead of %time:

```
%timeit pow(2, 100)
265 ns ± 3.05 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

%timeit 2**100
224 ns ± 1.26 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)
```

This means that Python inserted the command 2**100 inside a loop and carried out the operation million times. It evaluated 7 such loops.

How Long Does It Really Take?

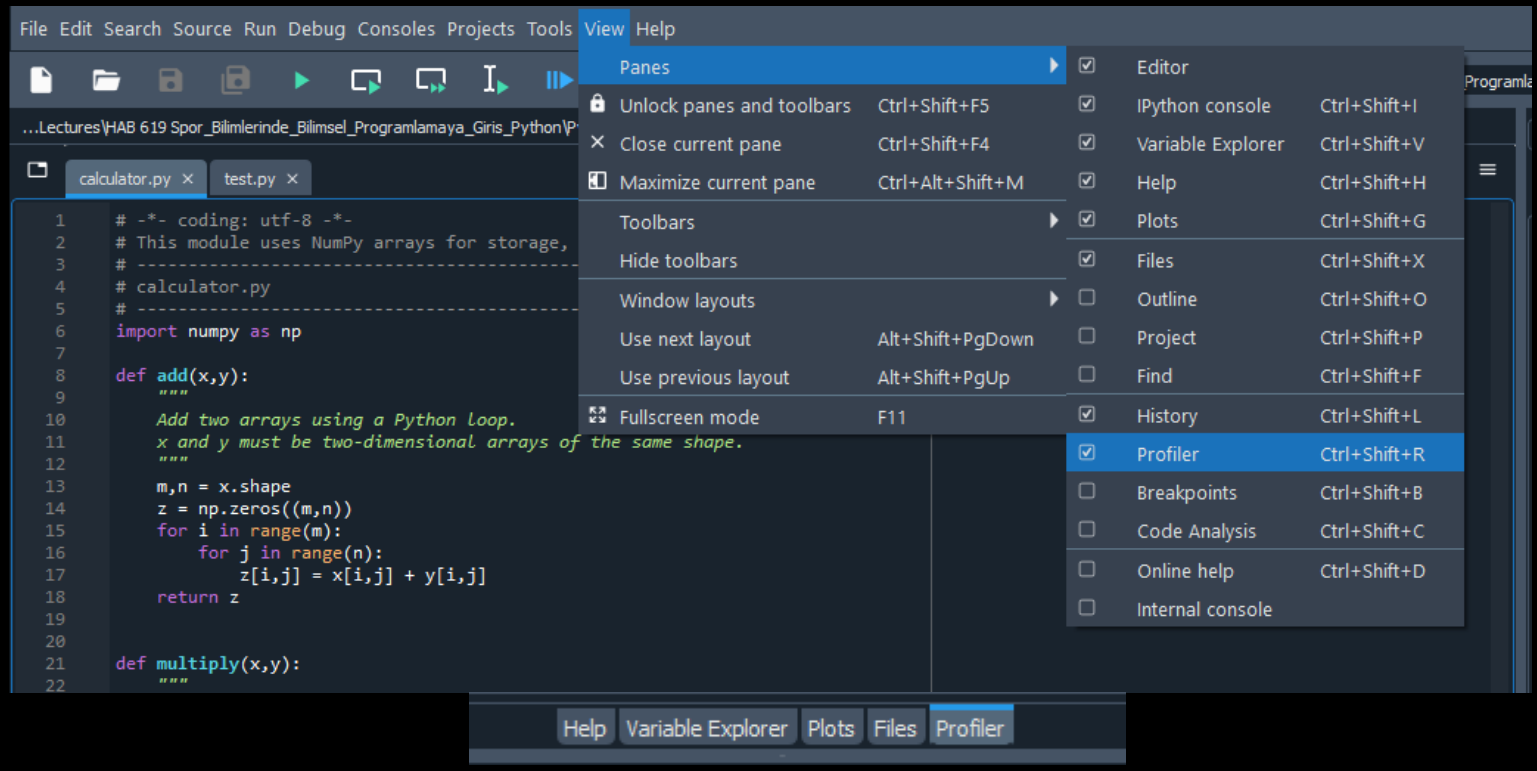
The %timeit Command

```
%timeit -r 10 -n 1000000 pow(2,100)
263 ns ± 4.47 ns per loop (mean ± std. dev. of 10 runs, 1000000 loops each)
```

`-r` tells Python to set the number of repetitions to whatever number comes next. Likewise, the `-n` tells Python to set the number of iterations in each loop to whatever number comes next.

Profiling

The Profiler pane recursively determines the run time and number of calls for every function and method called in a file, breaking down each procedure into its smallest individual units.



IPython

Profiling

```
# calculator.py
import numpy as np

def add(x,y):
m,n = x.shape
    z = np.zeros((m,n))
    for i in range(m):
        for j in range(n):
            z[i,j] = x[i,j] + y[i,j]
    return z

def multiply(x,y):
    m,n = x.shape
    z = np.zeros((m,n))
    for i in range(m):
        for j in range(n):
            z[i,j] = x[i,j] * y[i,j]
    return z

def sqrt(x):
    from math import sqrt
    m,n = x.shape
    z = np.zeros((m,n))
    for i in range(m):
        for j in range(n):
            z[i,j] = sqrt(x[i,j])
    return z

def hypotenuse(x,y):
xx = multiply(x,x)
    yy = multiply(y,y)
    zz = add(xx, yy)
    return sqrt(zz)
```

```
# test.py
import numpy as np
import calculator as calc

M = 10**3
N = 10**3

A = np.random.random((M,N))
B = np.random.random((M,N))

calc.hypotenuse(A,B)
```

Profiling

IPython

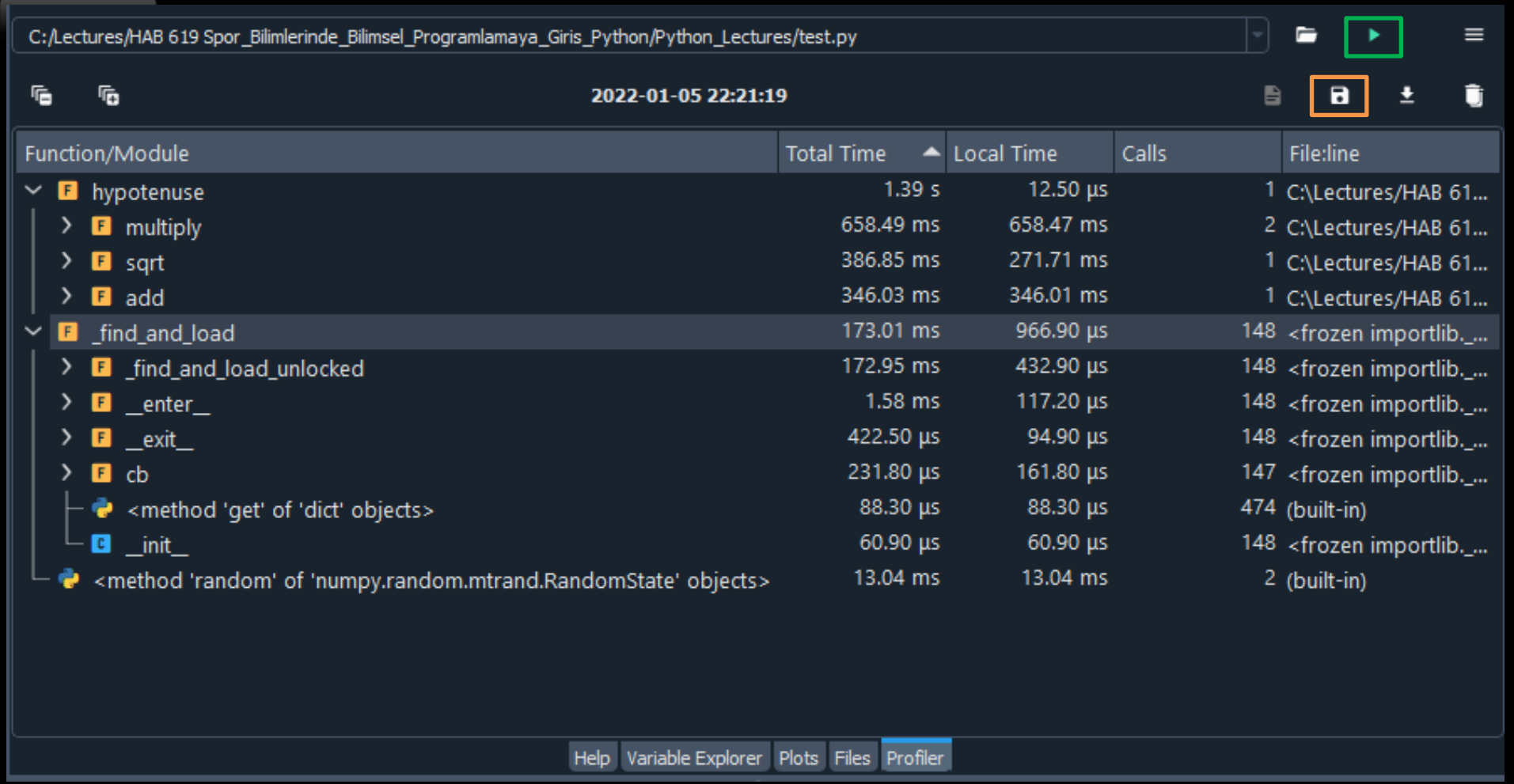
C:/Lectures/HAB 619 Spor_Bilimlerinde_Bilimsel_Programlamaya_Giris_Python/Python_Lectures/test.py

2022-01-05 22:21:19

Function/Module	Total Time	Local Time	Calls	File:line
hypotenuse	1.39 s	12.50 µs	1	C:\Lectures/HAB 61...
multiply	658.49 ms	658.47 ms	2	C:\Lectures/HAB 61...
sqrt	386.85 ms	271.71 ms	1	C:\Lectures/HAB 61...
add	346.03 ms	346.01 ms	1	C:\Lectures/HAB 61...
_find_and_load	173.01 ms	966.90 µs	148	<frozen importlib._...
_find_and_load_unlocked	172.95 ms	432.90 µs	148	<frozen importlib._...
__enter__	1.58 ms	117.20 µs	148	<frozen importlib._...
__exit__	422.50 µs	94.90 µs	148	<frozen importlib._...
cb	231.80 µs	161.80 µs	147	<frozen importlib._...
<method 'get' of 'dict' objects>	88.30 µs	88.30 µs	474	(built-in)
__init__	60.90 µs	60.90 µs	148	<frozen importlib._...
<method 'random' of 'numpy.random.mtrand.RandomState' objects>	13.04 ms	13.04 ms	2	(built-in)

Help | Variable Explorer | Plots | Files | Profiler

Profiling


C:/Lectures/HAB 619 Spor_Bilimlerinde_Bilimsel_Programlamaya_Giris_Python/Python_Lectures/test.py
2022-01-05 22:21:19
Function/Module
Total Time
Local Time
Calls
File:line
hypotenuse 1.39 s 12.50 µs 1 C:\Lectures/HAB 61...
multiply 658.49 ms 658.47 ms 2 C:\Lectures/HAB 61...
sqrt 386.85 ms 271.71 ms 1 C:\Lectures/HAB 61...
add 346.03 ms 346.01 ms 1 C:\Lectures/HAB 61...
_find_and_load 173.01 ms 966.90 µs 148 <frozen importlib._...
_find_and_load_unlocked 172.95 ms 432.90 µs 148 <frozen importlib._...
__enter__ 1.58 ms 117.20 µs 148 <frozen importlib._...
__exit__ 422.50 µs 94.90 µs 148 <frozen importlib._...
cb 231.80 µs 161.80 µs 147 <frozen importlib._...
<method 'get' of 'dict' objects> 88.30 µs 88.30 µs 474 (built-in)
__init__ 60.90 µs 60.90 µs 148 <frozen importlib._...
<method 'random' of 'numpy.random.mtrand.RandomState' objects> 13.04 ms 13.04 ms 2 (built-in)
Help
Variable Explorer
Plots
Files
Profiler

IPython

Profiling

C:/Lectures/HAB 619 Spor_Bilimlerinde_Bilimsel_Programlamaya_Giris_Python/Python_Lectures/test.py

2022-01-05 22:43:42

Function/Module	Total Time	Diff	Local Time	Diff	Calls
hypotenuse	1.41 s	+18.52 ms	10.80 µs	-1.70 µs	
multiply	668.24 ms	+9.75 ms	668.21 ms	+9.75 ms	
sqrt	386.55 ms	-295.10 µs	271.59 ms	-127.20 µs	
add	355.11 ms	+9.07 ms	355.08 ms	+9.07 ms	
_find_and_load	171.79 ms	-1.21 ms	959.20 µs	-7.70 µs	
_find_and_load_unlocked	171.74 ms	-1.21 ms	441.20 µs	+8.30 µs	
__enter__	1.58 ms	+1.70 µs	121.50 µs	+4.30 µs	
__exit__	391.20 µs	-31.30 µs	78.30 µs	-16.60 µs	
cb	224.70 µs	-7.10 µs	155.60 µs	-6.20 µs	
<method 'get' of 'dict' objects>	87.30 µs	-1000.00 ns	87.30 µs	-1000.00 ns	
__init__	63.00 µs	+2.10 µs	63.00 µs	+2.10 µs	
<method 'random' of 'numpy.random.mtrand.RandomState' objects>	12.96 ms	-81.80 µs	12.96 ms	-81.80 µs	

Help | Variable Explorer | Plots | Files | Profiler