

Profiling

1. Introduction and Algorithm Description

To ensure maximum memory efficiency, we use a [derived formula](#) for Sample Standard Deviation, we used the fact that the only repetitive calculations were the incrementing of the count of the elements, the [sum of the elements](#) and the [sum of the squares](#) of the elements of the sequence whose Sample StdDev is to be calculated.

$$s = \sqrt{\frac{1}{N-1} \left(\sum_{i=1}^N x_i^2 - N\bar{x}^2 \right)}$$
$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i \quad \text{and} \quad \sum_{i=1}^N x_i^2$$

That way the space complexity is $O(1)$.

The stddev module uses our core math library to check for its effectivity.

2. Profiling Results

We tested [10](#), [10^3](#) and [10^6](#) random decimal numbers

2.1 Test Case: 10 Elements

```
File Edit Selection View Go Run Terminal Help
* timo@rahau:~/V5_project_2/arc$ python3 -m cProfile -s tottime stddev.py < ten_numbers.txt
402921.387122399
136 function calls (135 primitive calls) in 0.001 seconds

Ordered by: internal time

ncalls  tottime  percall  ctime  percall  filename:lineno(function)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:1183(get_data)
1 0.000 0.000 0.000 0.000 stddev.py:15(main)
1 0.000 0.000 0.000 0.000 (built-in method builtins.print)
1 0.000 0.000 0.000 0.000 (built-in method marshal.load)
28 0.000 0.000 0.000 0.000 (built-in method builtins.read)
3 0.000 0.000 0.000 0.000 (built-in method posix.stat)
1 0.000 0.000 0.000 0.000 (built-in method io.open_code)
14 0.000 0.000 0.000 0.000 math.lib.py:174(clear_result)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:1062(get_code)
1 0.000 0.000 0.000 0.000 (method 'disable' of '_lispof.profiler' objects)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:1591(find_spec)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:1240(find_spec)
1 0.000 0.000 0.000 0.000 (method 'read' of 'io.BufferedReader' objects)
2 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:482(cache_from_source)
1 0.000 0.000 0.000 0.000 math.lib.py:121(module)
1 0.000 0.000 0.001 0.001 <frozen importlib._bootstrap>:1349(find_and_load)
6 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:126(path_join)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:74(new_...)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:1591(get_spec)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:162(enter_...)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:951(load_unlocked)
11 0.000 0.000 0.000 0.000 math.lib.py:124(square)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:384(acquire)
1 0.000 0.000 0.000 0.000 math.lib.py:141(sqrt)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:802(spec_from_file_location)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:734(join_module_attrs)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:611(get_cached)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:48(new_module)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap>:426(get_module_lock)
1 0.000 0.000 0.001 0.001 <frozen importlib._bootstrap>:1304(find_and_load_unlocked)
1 0.000 0.000 0.001 0.001 stddev.py:1(module)
1 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:989(exec_module)
14 0.000 0.000 0.000 0.000 math.lib.py:12(validate_numbers)
2 0.000 0.000 0.000 0.000 <frozen importlib._bootstrap_external>:132(path_split)
```

2.2 Test Case: 1 000 elements

```
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
time@tableau:/VS_project_calc/VS_project_2/src$ python3 -m cProfile -s tottime stddev.py < thousand_numbers.txt
506061.279770868
8269 function calls (8259 primitive calls) in 0.011 seconds

Ordered by: internal time

ncalls  tottime  percall  cuptime  percall  filename:lineno(function)
1      0.003    0.003    0.000    0.000  stddev.py:15(round)
2000   0.002    0.000    0.000    0.000  (built-in method builtins.round)
1004   0.001    0.000    0.000    0.000  math.lib.py:17(clean_result)
1001   0.001    0.000    0.000    0.000  math.lib.py:135(square)
1      0.001    0.001    0.001    0.001  (method 'read' of '_io.BufferedReader' objects)
1004   0.001    0.000    0.001    0.000  math.lib.py:12(validate_numbers)
1000   0.000    0.000    0.000    0.000  (method 'split' of 'str' objects)
1009   0.000    0.000    0.000    0.000  (built-in method builtins.isinstance)
1      0.000    0.000    0.000    0.000  (built-in method io.open code)
1028   0.000    0.000    0.000    0.000  (built-in method builtins.abs)
1      0.000    0.000    0.000    0.000  (built-in method marshal.loads)
1      0.000    0.000    0.000    0.000  (built-in method builtins.print)
1      0.000    0.000    0.000    0.000  (built-in method posix.stat)
1      0.000    0.000    0.001    0.001  frozen importlib._bootstrap_external:1062(get_code)
4      0.000    0.000    0.000    0.000  frozen codecs:119(decode)
1      0.000    0.000    0.001    0.001  frozen importlib._bootstrap_external:1181(get_data)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:124(setdefault)
1      0.000    0.000    0.002    0.002  frozen importlib._bootstrap:1349(find_and_load)
1      0.000    0.000    0.000    0.000  (method 'disable' of '_lsprof.profiler' objects)
1      0.000    0.000    0.000    0.000  (method 'exit' of '_io._IOBase' objects)
4      0.000    0.000    0.000    0.000  (built-in method codecs.utf_8_decode)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:74(new_)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:751(compile_bytecode)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:1246(find_spec)
1      0.000    0.000    0.000    0.000  math.lib.py:1(modules)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:1593(find_spec)
1      0.000    0.000    0.000    0.000  math.lib.py:141(sqrt)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:666(classify_pyc)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:426(get_module_lock)
1      0.000    0.000    0.000    0.000  stddev.py:1(stddev_calc)
1      0.000    0.000    0.011    0.011  stddev.py:1(modules)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:384(acquire)
1      0.000    0.000    0.002    0.002  frozen importlib._bootstrap:911(load_unlocked)
2      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:482(cache_from_source)
```

2.3 Test Case: 1 000 000 elements

```
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
time@tableau:/VS_project_calc/VS_project_2/src$ python3 -m cProfile -s tottime stddev.py < million_numbers.txt
577530.2901111
8004766 function calls (8004765 primitive calls) in 3.427 seconds

Ordered by: internal time

ncalls  tottime  percall  cuptime  percall  filename:lineno(function)
1999962  1.029    0.000    1.029    0.000  (built-in method builtins.round)
1      0.043    0.043    3.426    3.426  stddev.py:15(round)
1000004  0.448    0.000    1.573    0.000  math.lib.py:17(clean_result)
1000001  0.429    0.000    2.394    0.000  math.lib.py:135(square)
1000004  0.246    0.000    0.292    0.000  math.lib.py:12(validate_numbers)
1000000  0.113    0.000    0.113    0.000  (method 'split' of 'str' objects)
1000009  0.146    0.000    0.146    0.000  (built-in method builtins.isinstance)
1000028  0.096    0.000    0.096    0.000  (built-in method builtins.abs)
2280   0.082    0.000    0.000    0.000  (built-in method codecs.utf_8_decode)
2280   0.083    0.000    0.000    0.000  frozen codecs:119(decode)
1      0.000    0.000    0.000    0.000  (built-in method builtins.print)
1      0.000    0.000    0.000    0.000  (built-in method io.open code)
3      0.000    0.000    0.000    0.000  (built-in method posix.stat)
1      0.000    0.000    0.000    0.000  (method 'disable' of '_lsprof.profiler' objects)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:1062(get_code)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:1246(find_spec)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:1593(find_spec)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:1349(find_and_load)
1      0.000    0.000    0.000    0.000  math.lib.py:1(modules)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:1491(get_spec)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:74(new_)
1      0.000    0.000    0.000    0.000  math.lib.py:141(sqrt)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:384(acquire)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:426(get_module_lock)
1      0.000    0.000    0.000    0.000  (method 'read' of '_io.BufferedReader' objects)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:886(module_from_spec)
6      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:124(path_join)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap:124(setdefault)
2      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:482(cache_from_source)
1      0.000    0.000    3.427    3.427  stddev.py:1(modules)
1      0.000    0.000    0.000    0.000  stddev.py:1(stddev_calc)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:1183(get_data)
1      0.000    0.000    0.000    0.000  frozen importlib._bootstrap_external:751(compile_bytecode)
```

3. Observations

The time needed for calculating the Standard Deviation for 10 and 1 000 elements is negligible.

When it comes to the 1 000 000 elements test case, we can observe that the most time consuming method is the python built in method *round()* which comes from our core *math_lib.py* math library which is called every iteration as well as the other time consuming methods are, such as *_clean_result* and our *square()* method, these numbers are not surprising due to the immense number of iterations each of these methods are called in.

Bottlenecks: *round()*, *_clean_result*, *square()*, *stddev*'s main loop

4. Suggestions

To get rid of or minimise the time taken by the *round()* method, our *math_lib.py* would need a special *square_raw* method which would be called in the main loop of our *stddev* module omitting the repeated rounding and only rounding the number after all the iterations, possibly trading accuracy for a few seconds of a quicker calculation of [Stddev](#)