University of Puerto Rico

Mayagüez Campus

Electrical & Computer Engineering Department

Programming Assignment 1: Is Python fast or slow? - Report

Made by: Juan D. Pérez Sepúlveda,

Jeremy N. Alicea Vázquez,

Jeanpaul D. Ortiz Rodriguez

Made to: Prof. Jose F. Vega Riveros

Course: ICOM 5015

Date: February 28, 2024

Table of Content

Introduction:

When developing code some important factors to take into consideration are complexity of the code and execution time. We have resources at our disposal that can help reduce execution time allowing our code to become more efficient while making the algorithm simpler. Python libraries, a collection of related modules, contains bundles of code that can be used repeatedly in different programs making the process easier for programmers. NumPy is a Python library that provides multidimensional array objects, various derived objects (such as masked arrays and matrices) and an assortment of routines for fast operations on arrays.

Questions:

What is the purpose of the assignment?

The purpose of the assignment is to determine what form of operation process is more efficient to calculate the product of one-dimensional and two-dimensional arrays. We'll be comparing the iterating algorithm developed with one that makes use of python libraries such as NumPy to observe how each algorithm performs under three cases: when the size of both arrays is less than 10 elements, when the size of both arrays is 10 elements or more, but less than 100 and when the size of the array is 100 elements or more.

What is the key question or the hypothesis of this assignment?

Which algorithm will perform faster? The expected outcome is that the NumPy algorithm performs better than the one we made because it is already one of the most used python libraries. NumPy is very helpful for performing logical and mathematical calculations on arrays and matrices [1]. NumPy uses less memory and storage space, which generates an advantage that offers better performance in execution speed.

What are the key concepts taken into consideration for the design of the experiment?

The key concepts to take into consideration are the variety in the dimensions of the array which vary and must be compatible for the product of the arrays to be realized, knowing how the arrays are multiplied, knowing how time can be measured to calculate the execution, knowing how the numpy library works to use it and compare the results with the method without using the library and how to save the results and display them.

How did you set up the experiment (dimensions in 1-D and 2-D arrays)?

The way we set up the experiment was by defining a function which has the size of the arrays as a parameter. This function creates a one-dimensional array and a two-dimensional array with the size passed as a parameter initializing them to zero each position. Then, take these two arrays and place random numbers from 0 to 20 in each position of the arrays. The next thing we do is save the time in a variable called start before starting to multiply the arrays. Afterwards, the operation of multiplying the arrays is done and then the current time is saved again in the variable called end. This concludes the part of performing the calculations of the product of the matrices without the numpy library. Using the numpy library, using the same arrays with the same numbers, we create a new variable npstart to save the time before calling the function to multiply the arrays. Next, we create a variable called numpyResult which stores the result of multiplying the two arrays using numpy's multiply function. The next thing to do is create another variable and assign the time to it after the function has been executed. Finally, the size of the arrays and the time it takes to execute are printed, which is calculated with the time after execution minus the time after execution. Three different loops are used that are repeated 10 times each with a size variation of 0-9, 10-99 and 100-999.

How do the concepts identified before, together with authoritative information sources (technical literature on Python and NumPy) guide your analysis and interpretation of the data obtained from your experiments?

The interpretation of experimental results draws from key principles highlighted in the official documentation for Python and NumPy. NumPy enhances efficiency through vectorized operations, allowing for array-wide computations without the need for explicit iteration, thanks to its foundation in C and Fortran's optimized code. The design of NumPy arrays for greater memory compactness minimizes the amount of memory needed to perform a task, which becomes increasingly beneficial as dataset sizes expand. The results from our tests show exactly that, the timing is similar with the smaller arrays, but as the arrays get bigger in size, so does the difference in time between raw Python and NumPy. The transition to executing operations in compiled C code marks a significant departure from Python's interpretative execution, offering a notable increase in speed. The documentation not only deepens the understanding of these optimizations but also underscores the marked advantages of NumPy for handling large-scale scientific computations. As we scale up the dimensions of the arrays, these combined optimizations underscore the strategic fit of NumPy for complex, data-heavy scientific tasks.

As we can notice, when the dimensions of the product of 1-D Array and 2-D Array are less than 10 we can see that the numbers without using the numpy library are smaller than using NumPy library. When the size is greater than 9 we can notice a difference in which as the size increases,

the execution time also increases. In addition to this, analyzing the times of both implementations with sizes greater than 10, the execution time using the numpy library is less than without using the library in each of the executions. This tells us that for small numbers the method without using the numpy library is better than using it, but as the size increases, using the numpy is better. One of the reasons why the times calculated using the numpy library are lower than without using it is because numpy is written in the C language and therefore, the program uses fewer lines of code and is vectorized code which makes it more concise and easier to read [2][3]. According to Geek for Geeks, vectorization is used to speed up the Python code without using loop [4].

Tables

Table 1: Arrays with dimensions less than 10:

1-D size:	2-D size:	Time without NumPy (ms):	Time with NumPy (ms):
2	2x2	0.002	0.016
3	3x3	0.003	0.008
4	4x4	0.003	0.009
5	5x5	0.005	0.015
6	6x6	0.007	0.012
7	7x7	0.008	0.018
8	8x8	0.010	0.015
9	9x9	0.012	0.017

Table 2: Arrays with dimensions greater than 9 but less than 100:

1-D size:	2-D size:	Time without NumPy (ms):	Time with NumPy (ms):
14	14x14	0.027	0.026
19	19x19	0.049	0.037
20	20x20	0.053	0.033
22	22x22	0.064	0.043
45	45x45	0.258	0.147
47	47x47	0.279	0.146
47	47x47	0.279	0.162
51	51x51	0.327	0.162
77	77x77	0.742	0.388

91	91x91	1.056	0.547

Table 3: Arrays with dimensions greater than 99 but less than 1000:

1-D size:	2-D size:	Time without NumPy (ms):	Time with NumPy (ms):
233	233 x 233	6.684	3.151
240	240 x 240	7.002	3.286
306	306 x 306	11.6	5.036
343	343 x 343	14.844	6.639
397	397 x 397	20.668	9.201
495	495 x 495	31.527	13.571
541	541 x 541	38.223	16.195
820	820 x 820	88.863	37.194
932	932 x 932	114.982	48.464
993	993 x 993	130.308	54.247

Figures

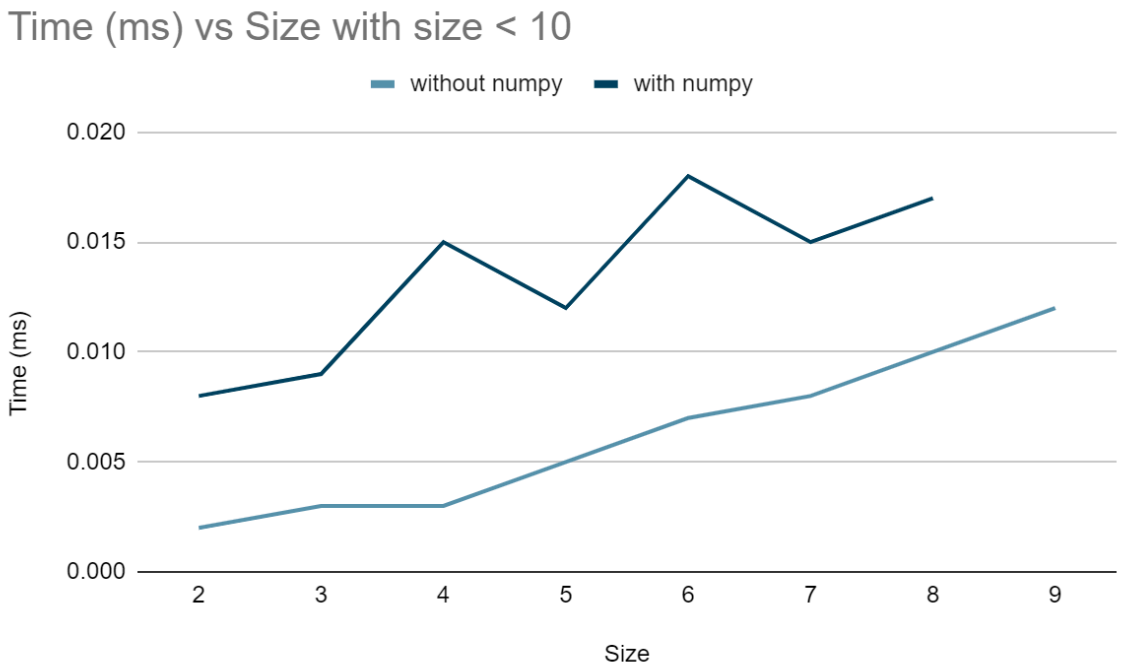

Figure 1: Time vs Size with the size of the arrays less than 10.

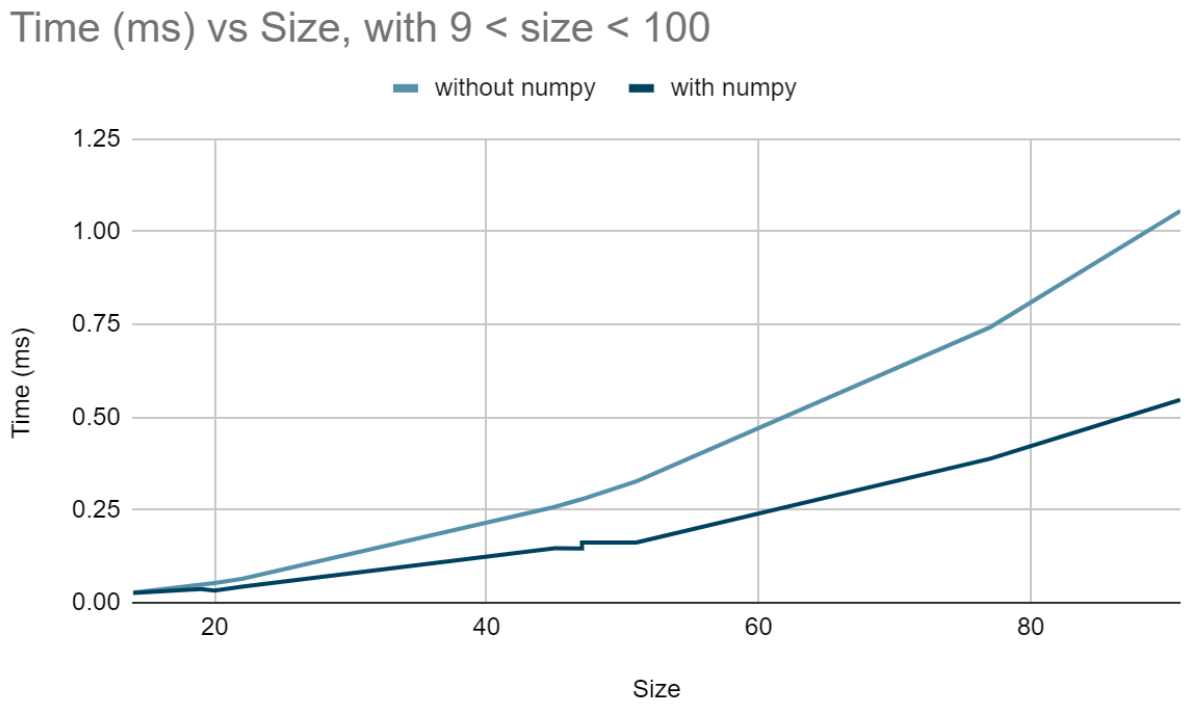

Figure 2: Time vs Size with the size of the arrays greater than 9 but less than 100.

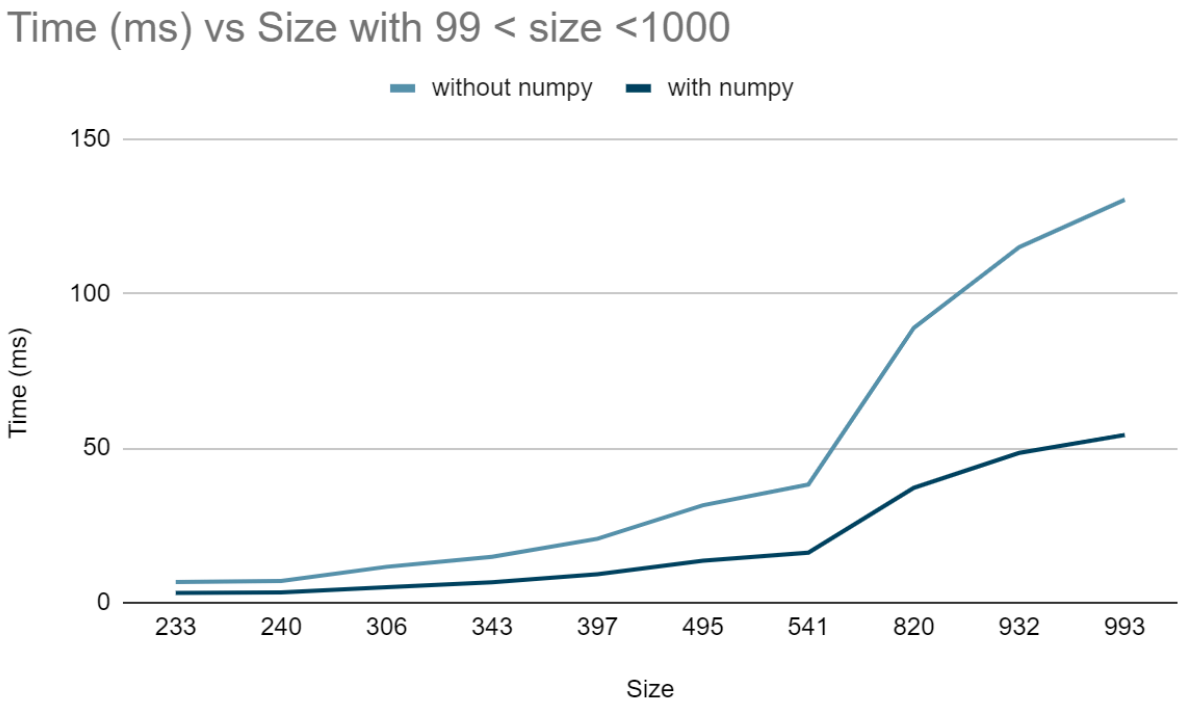

Figure 3: Time vs Size with the size of the arrays greater than 99 but less than 1000.

Conclusion:

From the experiment we can conclude that Numpy is a very useful library especially when dealing with operations in arrays. We can determine by the cases illustrated in Table 2 and 3 that the algorithm when it includes NumPy is faster by an average of 45% when compared with the algorithm without it making it more efficient.The efficiency of NumPy can be clearly seen in the Figures 2 and 3 where the growth of execution time is much slower than the algorithm without NumPy. We can also determine with the experiment that when the array dimensions are less than 10 the algorithm without NumPy is faster, the smaller the smaller the array the faster it executes, illustrated in Table 3.

What we learned

After concluding the different tests, we learned how important libraries are for programming, in this case NumPy for high-level calculations. While Python is capable of doing array multiplication, NumPy takes advantage of compiled programming languages, like C, to perform the same calculations more efficiently. On top of that, we used the Time library, to simplify the process of timing each test. This not only highlights the versatility of Python as a language, but

also underscores the critical role of optimized libraries like NumPy in computational efficiency and enabling more sophisticated scientific and data-driven analyses.

Task division:

The coding section of the assignment was done independently by each member which were compared after to make sure the process was done correctly and had similar outputs. We divided the report task with the suggested key components the professor specified:

Juan D. Pérez Sepúlveda did the sections that address:
a)What are the key concepts taken into consideration for the design of the experiment? b)How did you set up the experiment (dimensions in 1-D and 2-D arrays)? Made the Figures 1, 2, 3 and Table 3.

Jeremy N. Alicea Vázquez did the sections that address:

a)What is the purpose of the assignment? b)What is the key question or the hypothesis of this assignment? c)What do you conclude from this experiment?, Did the introduction and Tables 1 and 2.

Jeanpaul D. Ortiz Rodriguez did the sections that address:

a)How do the concepts identified before, together with authoritative information sources (technical literature on Python and NumPy) guide your analysis and interpretation of the data obtained from your experiments? b)What did you learn from this assignment?

References:

[1] D. Team, “NumPy : the most used Python library in Data Science,” *Data Science Courses | DataScientest*, Mar. 13, 2023. https://datascientest.com/en/numpy-the-python-library-in-data-science#:~:text=What%20are%20the%20advantages%20of (accessed Feb. 29, 2024)

[2] “Building from source — NumPy v1.26 Manual,” *numpy.org*. https://numpy.org/doc/stable/user/building.html (accessed Feb. 29, 2024).

[3] NumPy, “What is NumPy? — NumPy v1.19 Manual,” *numpy.org*, 2022. https://numpy.org/doc/stable/user/whatisnumpy.html (accessed Feb. 29, 2024)

[4] GeekForGeeks, “Vectorization in Python,” *GeeksforGeeks*, Apr. 18, 2019. https://www.geeksforgeeks.org/vectorization-in-python/ (accessed Feb. 29, 2024)

[5] parthmanchanda81, “Libraries in Python,” *GeeksforGeeks*, Oct. 16, 2021. https://www.geeksforgeeks.org/libraries-in-python/