Jerico Sabile 2021-00075

CMSC 180
Introduction to Parallel Computing
Laboratory Research Problem 01:
Computing the Pearson Correlation Coefficient of a Matrix and a Vector

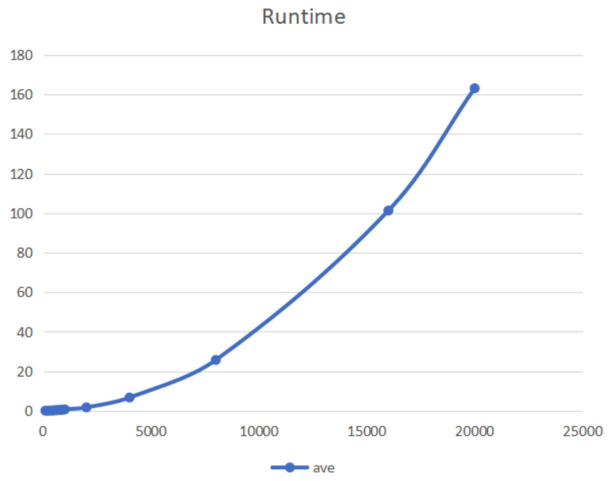
Research Question 1: What do you think is the complexity of solving the Pearson Correlation Coefficient vector of an n x n square matrix X with a n x 1 vector y? (hint: CMSC 142)

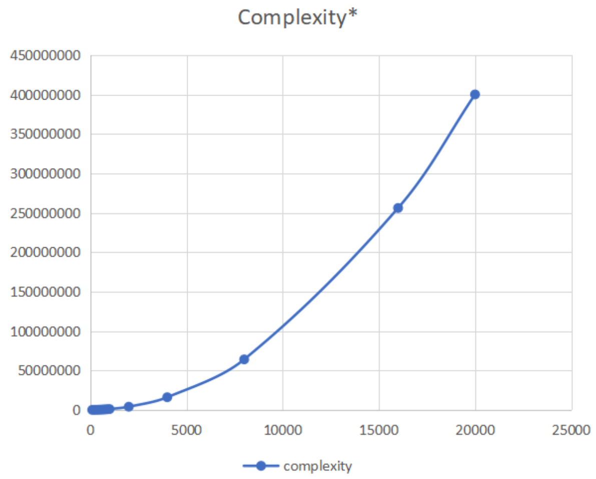
$O(n^2)$

n	**Time Elapsed (seconds)**			**Average Runtime (seconds)**	**Complexity***
	Run 1	**Run 2**	**Run 3**		
100	0.0070	0.0079	0.0071	0.0073	10 000
200	0.0274	0.0277	0.0269	0.0273	40 000
300	0.0612	0.0603	0.0591	0.0602	90 000
400	0.1066	0.1089	0.1056	0.1070	160 000
500	0.1665	0.1664	0.1655	0.1661	250 000
600	0.2388	0.2406	0.2360	0.2385	360 000
700	0.3255	0.3222	0.3217	0.3231	490 000
800	0.4226	0.4314	0.4218	0.4253	640 000
900	0.5362	0.5480	0.5380	0.5407	810 000
1000	0.6573	0.6688	0.6737	0.6666	1 000 000
2000	1.5561	1.5409	1.4650	1.5207	4 000 000
4000	6.2504	6.9516	6.8140	6.6720	16 000 000
8000	25.2671	25.9006	25.8702	25.6793	64 000 000
16000	101.2493	102.3986	101.4780	101.7086	256 000 000
20000	163.1231	158.4946	167.4386	163.0188	400 000 000

Research Question 2: Were you able to run up to n > 10,000,000? If so, can you make it higher to 50,000,000 or even 100,000,000? If not, why do you think so and what do you need to do to make it so?

> **No, I was not able to run up to n > 10,000,000. I think the reason behind this is the time complexity of the program is quadratic. As n increases, the time also increases quadratically.**





Research Question 3: Do the two lines agree, at least in the form? If not, provide an explanation why so?

> **Yes the two lines have the same trend or form. The two lines resemble a quadratic line or equation.**

Research Question 4: Discuss ways on how we can make it better (lower average runtime) without using any extra processors or cores (notice that the word “ways” is in plural form).

> **One way to make the program have a lower average runtime is to analyze the algorithm for optimizations. In the code I used, instead of computing the sumX and sumY two times per loop, computing both and storing it in the variable will avoid redundant computations. Another way to lower the runtime is to use multiple threads to compute the Pearson correlation coefficient of j columns.**