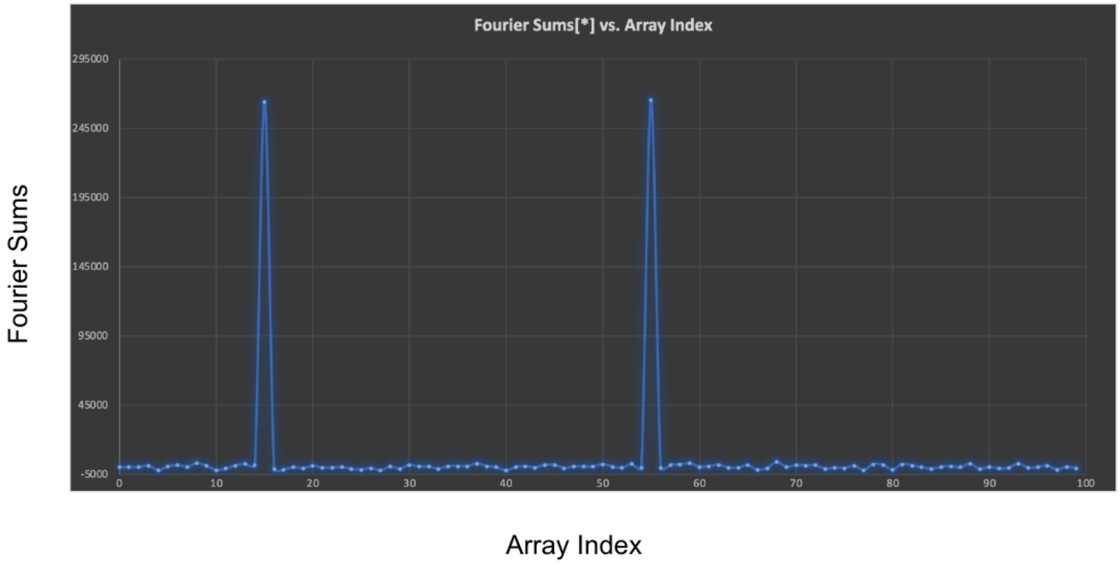
Author: Clinton Lohr
Email: lohrcl@oregonstate.edu

Date: 06/07/2023
Course: CS 475 - Intro to Parallel Programming
Project: 07 - Fourier Analysis using MPI

Fourier Sums vs. Array Index Scatterplot

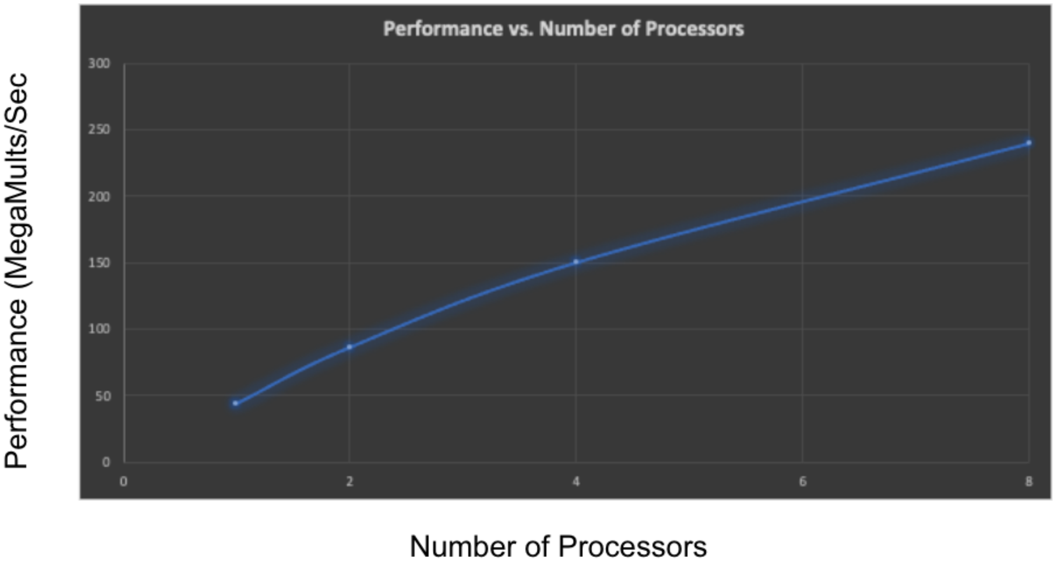
Array indices only consider 100 periods and not the total number of elements.



Based on the scatterplot, the secret sine-waves in the signal are at array indices 15 and 55, where there are significant spikes.

Performance vs. Number of MPI Processors

All runs for different MPI Processor counts used an input array of 1,048,579 elements. Array size was divided by the number of MPI processors used to determine how many elements each processor would handle.



Looking at the Performance vs. Number of Processors graph, there are distinct patterns that can be explained through parallelism. Considering the first point in which only one MPI processor was assigned, we have a performance number just below 50 Mega Multiplies per second. This is the slowest of the times for all trials. The reason being that all computation was handled by one processor.

The second point on the graph is at 2 for the number of MPI processors used. The performance number of this run was a little less than double that of the run using just one processor. This number is to be expected as it makes sense that if we divide the workload amongst two processors, the performance will double.

Again, for the third run we doubled the number of MPI processors, dividing the work amongst four processors. This also resulted in the expected performance numbers. The performance was a little less than double that of the performance when using two processors.

The final run used eight MPI processors to divide up the array. As expected, performance improved from the previous run in which four MPI processors were used. One trend that was different between runs using four MPI processors and eight MPI processors and the previous runs, was that there wasn’t as significant of a performance increase. This is likely due to the processors getting closer to the maximum that they can compute. As the number of processors increases, the curve will flatten out even more as the overhead stays the same.