ECE 4122 Final Report

Group members: Christopher Chen

Yee Aung

Nick Landi

Mohammad Karim

Nathan Olivares

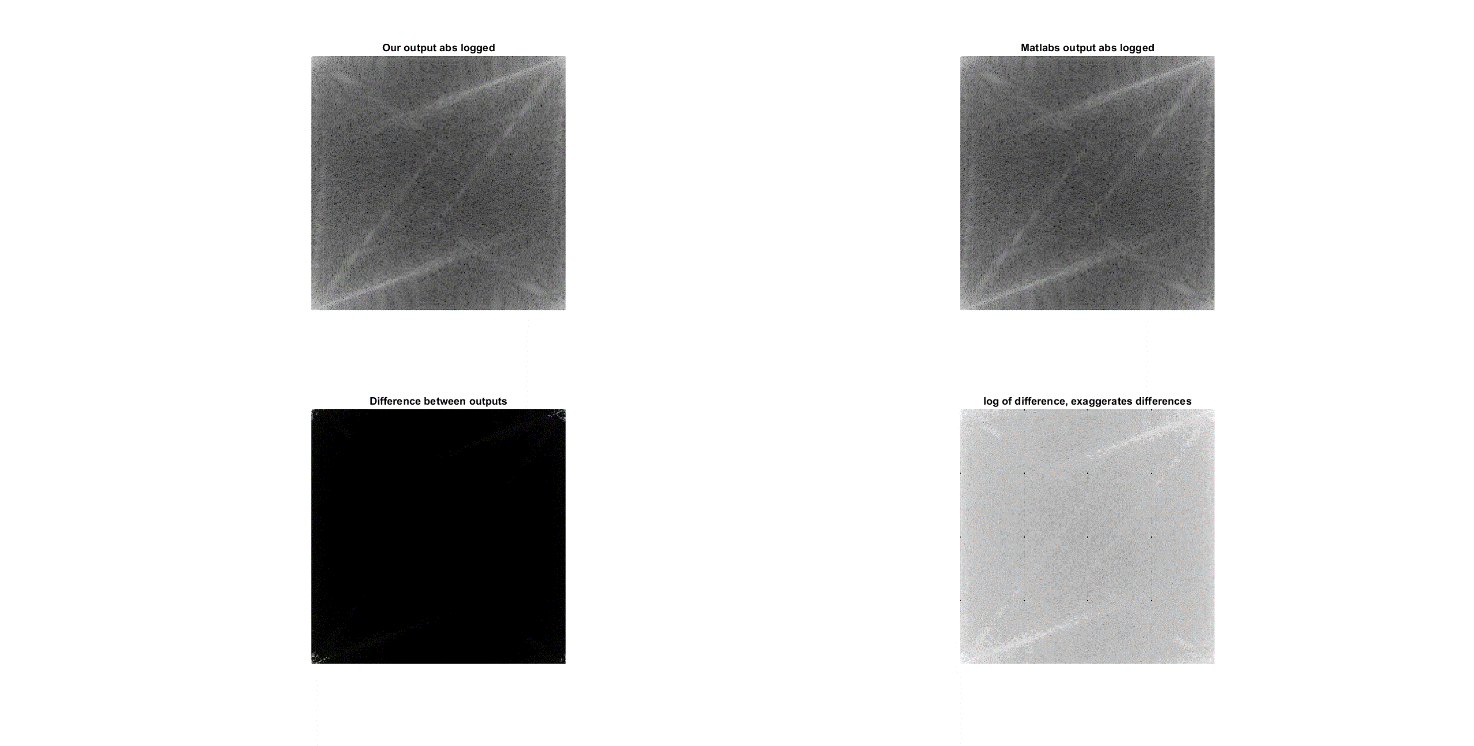
**We did the extra credit part where we implemented c-make for all three methods in one text file.**

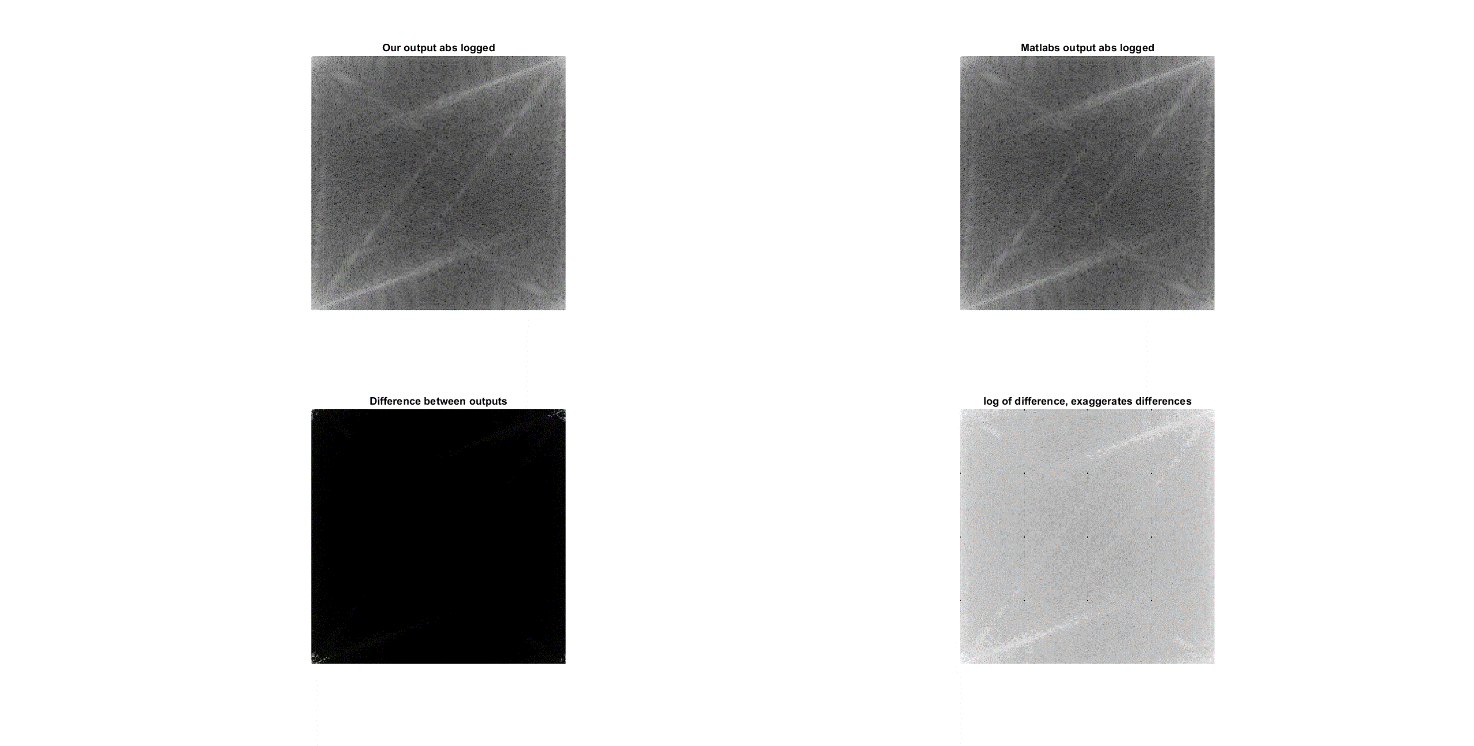
Runtimes for CUDA, MPI, and Multi-Thread implementations.

Above runtime graph measure the duration of DFT implementation across three different programming methods which include CUDA, MPI and C-Threads. The runtime was measured in milliseconds and it accounts for reading the input file, performing the parallel processing regarding to its method and writing an output file. As you can see, CUDA and MPI implementation have roughly the same runtime while threads runtime increases as the dimension of the input text file increases. MPI used Cooley-Turkey FFT algorithm and was tested using 8 processors across different dimensions while CUDA and C-Threads implementation utilizes the regular DFT algorithm. MPI lags behind CUDA’s runtime due to the number send and receive communication between different processors when performing the parallel processing but CUDA was able to avoid this by using global GPU memory. C-Threads suffers the lengthiest runtime due to the code being tested only on 16-threads. The runtime of C-Threads can be further reduced by using FFT algorithm instead of the DFT and by running on more threads.

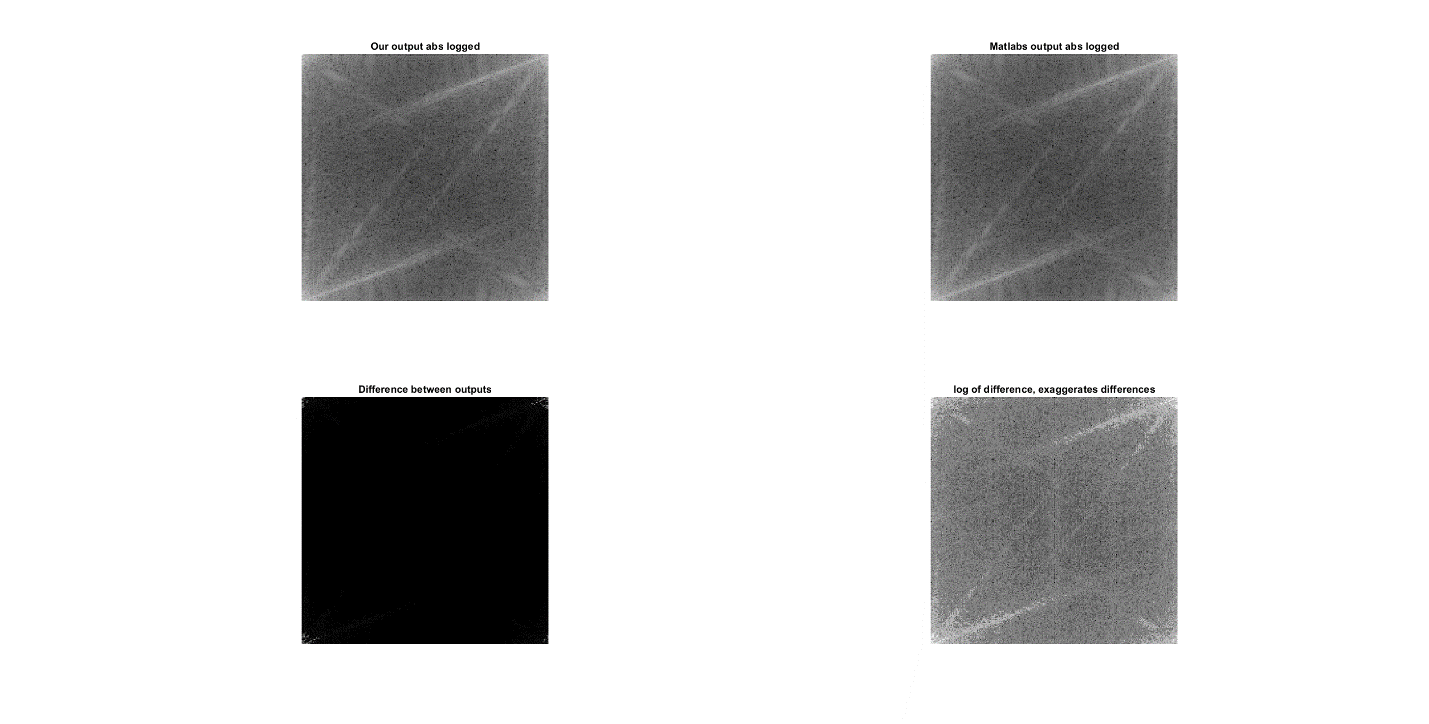
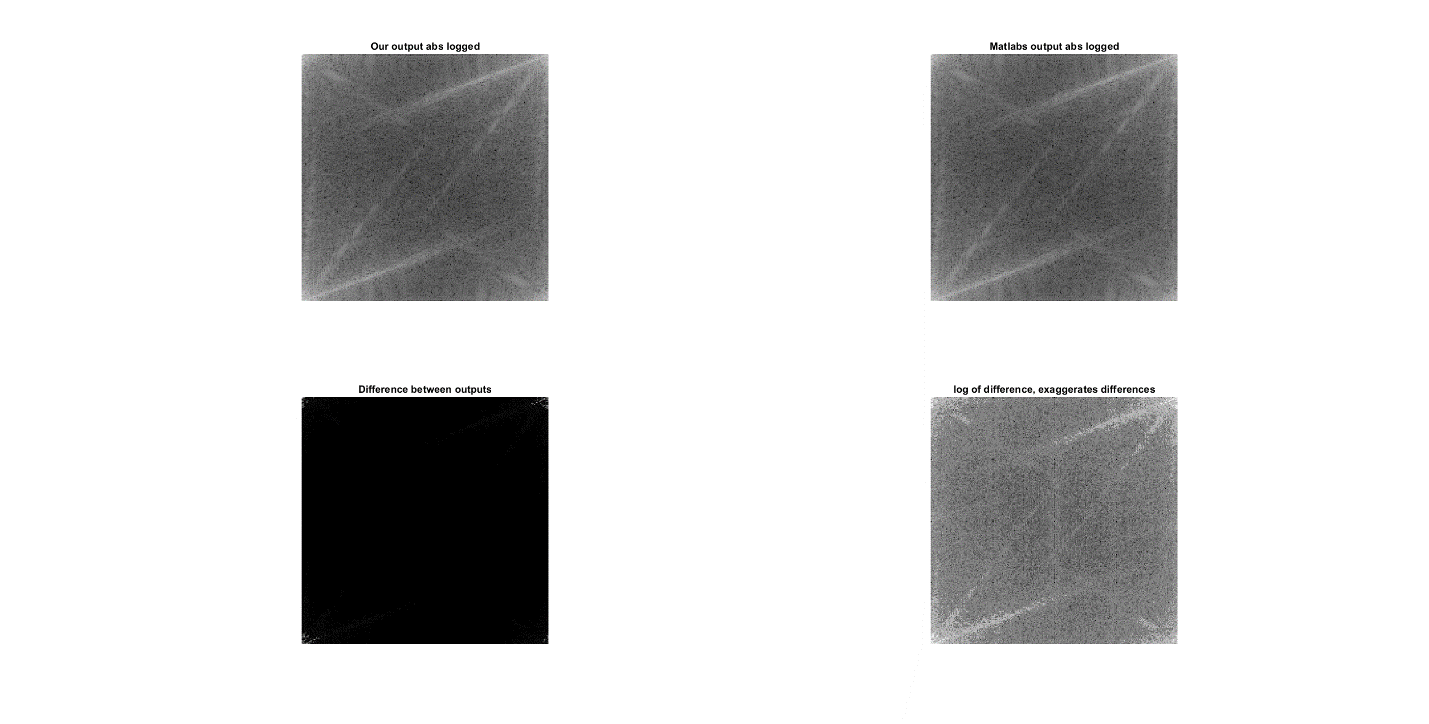
We included out result of Tower256.txt output and compared it to MATLAB FFT2 output. As you can see below, there is not much discrepancies between each of the methods compare to MATLAB’s function.

Implementation output pictures:

1.  Cthreads Tower256:



1. MPI Tower256:



3) CUDA Tower256:

