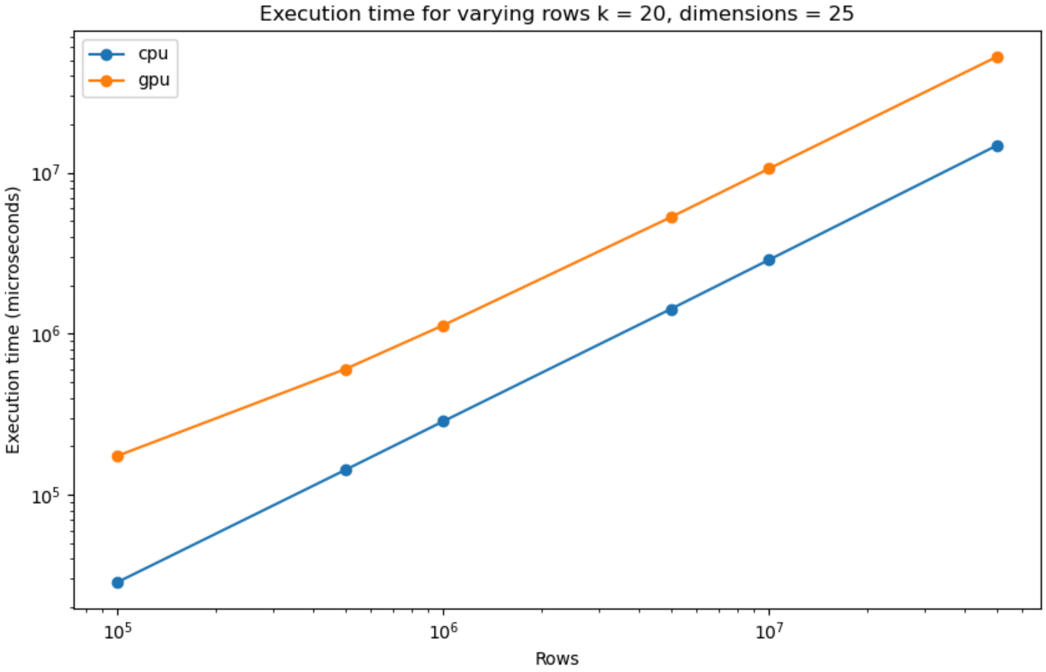
For this assignment, I compared my fastest CPU implementation with KMCuda. I created datasets with 25 dimensions and 100,000, 500,000, 1 million, 5 million, 10 million, and 50 million rows. I tested with k = 20. Interestingly, my CPU implementation was faster than KMCuda in all cases. This was not the result I expected, as running this algorithm on a GPU should theoretically reduce execution time. So, I tweaked the parameters such as yinyang, device, and precision. None of which resulted in any meaningful change. I also switched to using k++, however, interestingly, this had very little impact on execution time. Suggesting that my CPU implementation converges quicker despite choosing initial random points.


Execution time for varying rows k = 20, dimensions = 25
cpu
gpu
Execution time (microseconds)
10^7
10^6
10^5
10^5
10^6
10^7
Rows