# HW4 Performance Analysis

## CUDA-

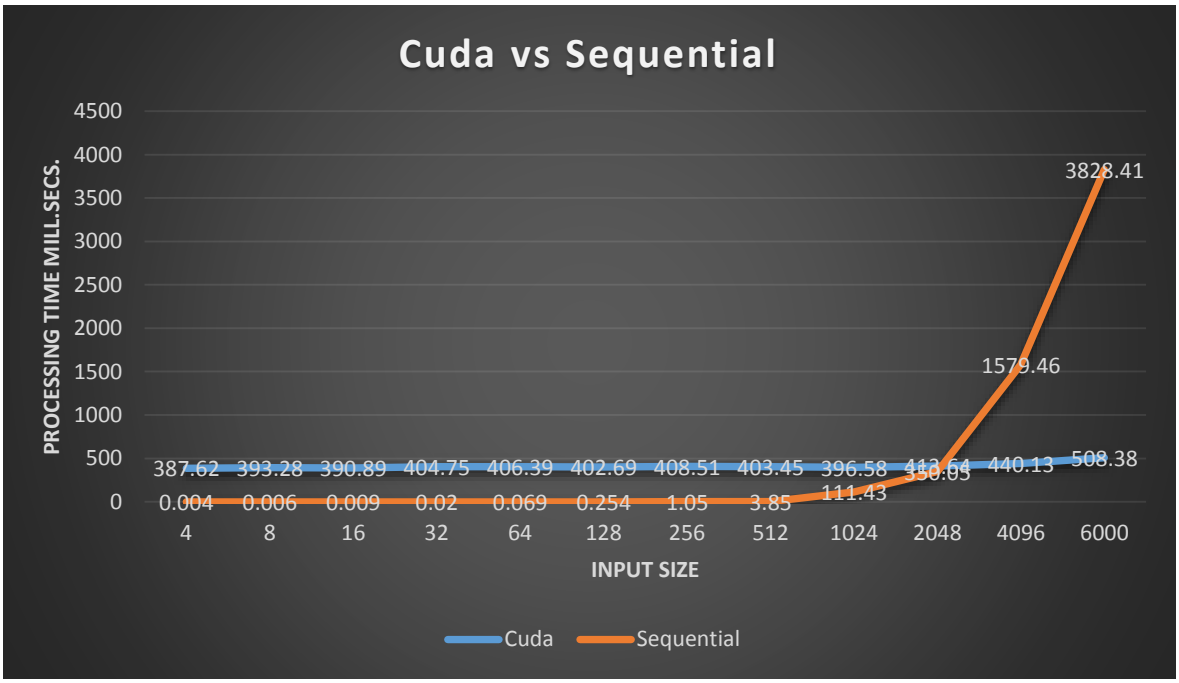
**Correctness argument:** The operation of calculating mean of the columns, standard deviation of the columns, variance of each element and the normalization are performed on the GPU. The total number of threads is kept same as the number of columns in the first two operations, and it is kept same as the number of elements in the last two operations. A local copy of matrix A is maintained on the device.

**Different Versions tried:**

1. The first version of the program performed the entire normalization operation at once. The number of threads were always kept same as the number of columns present in the matrix. Same threads were performing the normalization operation on all the elements of a column.

2. In the second version, I split the operation into four sub-operations. The sub-operations which were column level were performed using the number of threads same as the number of columns. The sub-operation which were element level operation, and had no dependency on other elements of the column were performed using the number of threads same as the number of elements. It provided a good speedup.

**Performance analysis:**




Following table shows the comparison between the sequential version and the CUDA version of the program. For less number of elements, the CUDA takes more time, as the data transfer between the host and the device requires some time. The sequential version is quite fast for less number of elements. As the size of input grows, the time required for parallel version is almost constant, but the time required for the sequential version increases drastically. **The number of threads per block on GPU is kept same as the Warp size of the GPU i.e. 32.**

| Input Size | Processing time CUDA | Processing time SEQ |
|---|---|---|
| **4** | 387.62 | 0.004 |
| **8** | 393.28 | 0.006 |
| **16** | 390.89 | 0.009 |
| **32** | 404.75 | 0.02 |
| **64** | 406.39 | 0.069 |
| **128** | 402.69 | 0.254 |
| **256** | 408.51 | 1.05 |
| **512** | 403.45 | 3.85 |
| **1024** | 396.58 | 111.43 |
| **2048** | 413.64 | 350.05 |
| **4096** | 440.13 | 1579.46 |
| **6000** | 508.38 | 3828.41 |

**Speed up:**

$$Sp = Ts/Tp$$

For input size N=6000,

Ts = 3828.41 ms

Tp = 508.38 s

Sp =3828.41/508.38

**Sp = 7.53**