CMOR 420/520, Homework #2: LaTeX Submission

bi3

October 11, 2023

1 Compilation

```
gcc verification.c -I./include ./src/matrix.c
gcc timing.c -I./include ./src/matrix.c
```

2 Verification Output

```
Contiguous matrix multiplication error: 0.000000
Contiguous transpose matrix multiplication error: 0.000000
Non-contiguous matrix multiplication error: 0.000000
Non-contiguous transpose matrix multiplication error: 0.000000
```

3 Timing table

	$m = n = 1000$	$m = 2000$	$m = 3000$	$m = 4000$
matvec (Contig)	0.004300	0.016241	0.045804	0.229530
matvec transpose (Contig)	0.006213	0.060537	0.141352	0.391721
matvec (Non-contig)	0.004854	0.020777	0.044203	0.096590
matvec transpose (Non-contig)	0.010008	0.070714	0.144999	0.544047
matvec (Contig, -O3)	0.002107	0.009743	0.012147	0.037924
matvec transpose (Contig, -O3)	0.004206	0.039258	0.100988	0.258681
matvec (Non-contig, -O3)	0.001945	0.004696	0.015548	0.020373
matvec transpose (Non-contig, -O3)	0.003129	0.032746	0.137521	0.444403
MATLAB	0.0156027	0.0662907	0.1447487	0.2735435
MATLAB (transpose)	0.0091237	0.0194655	0.0476365	0.0955813

From the above, we see that calculating the product of a transpose matrix and x takes a longer time in C compared to regular matrix multiplication. Furthermore, initializing a matrix using contiguous memory leads to faster execution compared to non-contiguous memory. Unsurprisingly, the timings increase as the size of the matrix increases, and using the -O3 flag does speed up the calculations. Matlab appears to calculate much more slowly than almost all the C calculations with regular matrix multiplication. Curiously, it calculates the product of a transpose matrix faster then the regular product. This may be the result of different processes occurring on my computer at the time of writing (Python running calculations, all the tabs I have open, etc.) It could also be that Matlab is its own program and may have extra processes occurring in its background, while compiling C is a more simple process.