Jeff Anderson
CSCI 551 project 1

TIMES FOR ACTUAL RUNS

| | **RUN 1** | **RUN 2** | **RUN 3** |
|---|---|---|---|
| **Mmvec add** | 0.059233 | 0.063551 | 0.060659 |
| **Mmvec mult** | 163.334015 | 164.632050 | 163.045349 |
| **Mmnovec add** | 0.062356 | 0.062067 | 0.062743 |
| **Mmnovec mult** | 343.167145 | 334.637085 | 344.569427 |

Speedup for addition
0.062067 ÷ 0.059233 = 1.0478

Speedup for multiplication
334.637085 ÷ 163.045349 = 2.0524

Maximum resident set size (average of all 6 runs)
(1050532 + 1050484 + 1050596 +1050440 + 1050476 + 1050548) / 6 = 1050512.667

As per the ICC documentation, using an IKJ format (referred to as STRIDE-1) improves vectorization performance by working quicker through the different buckets. In IJK form, you would need the result from the last iteration from the previous loop. Every iteration in STRIDE-1 is only a partial calculation in the elements of C, like the tree-structured reduction method we discussed in class. It's working quicker through the different elements.

I used

```
-march=core-avx2 -O2 -D NOALIAS -D ALIGNED
```

For compile flags. Judging by the class spreadsheet, there were [seemingly] two ways to implement addition. I think that by using static arrays and aligning the data to 16 bit made the fastest addition I could have had. My fastest clock time ever was 0.0592.

The theoretical speedup would be roughly 8 times faster because the AVX2 has a 256 bit SIMD architecture. Because we are using floats, and floats take 32 bits of space, that would give us

$256 \div 32 = 8$. Of course, that is only theoretical and not a real-world scenario. My optimization report listed the estimated potential speedup for the add function at 7.140 and 6.030 for the multiplication function.

```
janderson1@o244-02:~/551/p1$ make
icc -o mmvec -Wall -Werror -std=c99 -march=core-avx2 -O2 -qopt-report=5 -qopt-report-phase=vec -D NOALIAS -D ALIGNED
mm.c
icc: remark #10397: optimization reports are generated in *.optrpt files in the output location
icc -o mmnovec -Wall -Werror -std=c99 -march=core-avx2 -O2 -no-vec -D NOALIAS -D ALIGNED mm.c
janderson1@o244-02:~/551/p1$ ls
Makefile  mm.c  mmnovec  mm.optrpt  mmvec
janderson1@o244-02:~/551/p1$ ./mmvec
R
8192

time for adding matrices = 0.065395
time for multiplying matrices = 162.995972
usage time = 164568520
Max RSS = 1050628
janderson1@o244-02:~/551/p1$ ./mmvec
R
8192

time for adding matrices = 0.060990
time for multiplying matrices = 164.472382
usage time = 165977077
Max RSS = 1050480
janderson1@o244-02:~/551/p1$ ./mmvec
R
8192

time for adding matrices = 0.063222
time for multiplying matrices = 164.710342
usage time = 166242858
Max RSS = 1050596
janderson1@o244-02:~/551/p1$ ./mmnovec
R
8192

time for adding matrices = 0.063019
time for multiplying matrices = 343.132477
usage time = 344708154
Max RSS = 1050592
janderson1@o244-02:~/551/p1$ ./mmnovec
R
8192

time for adding matrices = 0.063803
time for multiplying matrices = 342.322479
usage time = 343852201
Max RSS = 1050584
janderson1@o244-02:~/551/p1$ ./mmnovec
R
8192

time for adding matrices = 0.062555
time for multiplying matrices = 342.893311
usage time = 344458279
Max RSS = 1050596
janderson1@o244-02:~/551/p1$
```