The difference between the speed of the benchmark-blas and benchmark-blocked was very significant. However, the difference in speed between the benchmark-naive and benchmark-blocked were not so significant. In order to increase the speed of dgemm-blocked I tried multiple strategies. Some worked very well and some barely made any change. I noticed that manipulating the block size had different effects based on if you increased the block size or decreased it. The strategy I tried to implement for the code was loop unrolling. I tried to use different modules such as #pragma to get the compiler to automatically unroll the loop but nothing seemed to work for me as #pragma was ignored by the compiler. So instead, I manually unrolled the loops in the code. I did it to where the matrix multiplication can iterate 4 times in a single iteration because this is what loop unrolling does in order to increase the speed of a program. This strategy increased the speed of the blocked code for me but for some reason the code fails even after compiling to the very last test size matrix. I've tried countless things to try and fix this problem, but nothing has seemed to work. Overall, I found it very interesting and challenging to try and increase the speed/performance of the algorithm.