Assignment 2 - Matrix Multiplication
COMP30250 - Parallel and Cluster Computing

October 29, 2021

Aleryc SERRANIA - 21204068

1 Introduction

This report was created with a Jupyter notebook. You can find the original source of this notebook in the same folder (`report.ipynb`). Source code of the implementations of the algorithms can be found inside `src` folder. A Makefile is provided to compile the code.

The variants for this assignments are the following:

1. Manually written straightforward non-blocked ijk algorithm;
2. Blocked ijk algorithm using ATLAS calls to compute multiplication of blocks;
3. Blocked kij algorithm using ATLAS calls to compute multiplication of blocks;
4. Compare the fastest program with BLAS dgemm routine.

I built a python script `run_exp.py` to get results for different sizes of matrix and block and also to run multiple iterations of the same configuration to get an average of the execution time (25 iterations). The results are compiled inside `data.csv`.

2 Detailed performance results for each algorithm

Timing unit is second.

2.1 Straightforward non-blocked ijk

```
[3]:             filename  matrix_size        timing
     0  non_blocked_ijk.out            2  2.000000e-07
     1  non_blocked_ijk.out            4  7.600000e-07
     2  non_blocked_ijk.out            8  3.480000e-06
     3  non_blocked_ijk.out           16  2.420000e-05
     4  non_blocked_ijk.out           32  2.048400e-04
     5  non_blocked_ijk.out           64  1.452720e-03
     6  non_blocked_ijk.out          128  1.265856e-02
     7  non_blocked_ijk.out          256  1.499471e-01
     8  non_blocked_ijk.out          512  1.256371e+00
     9  non_blocked_ijk.out         1024  2.541357e+01
```

2.2 Blocked ijk with ATLAS

```
[4]:                 filename  matrix_size  block_size    timing
     20  blocked_ijk_atlas.out            2           2  0.000017
     21  blocked_ijk_atlas.out            4           2  0.000020
     22  blocked_ijk_atlas.out            4           4  0.000018
     23  blocked_ijk_atlas.out            8           2  0.000026
     24  blocked_ijk_atlas.out            8           4  0.000018
     25  blocked_ijk_atlas.out            8           8  0.000017
     26  blocked_ijk_atlas.out           16           2  0.000055
     27  blocked_ijk_atlas.out           16           4  0.000026
     28  blocked_ijk_atlas.out           16           8  0.000020
     29  blocked_ijk_atlas.out           16          16  0.000021
     30  blocked_ijk_atlas.out           32           2  0.000341
     31  blocked_ijk_atlas.out           32           4  0.000081
     32  blocked_ijk_atlas.out           32           8  0.000043
     33  blocked_ijk_atlas.out           32          16  0.000034
     34  blocked_ijk_atlas.out           32          32  0.000035
     35  blocked_ijk_atlas.out           64           2  0.002313
     36  blocked_ijk_atlas.out           64           4  0.000451
     37  blocked_ijk_atlas.out           64           8  0.000183
     38  blocked_ijk_atlas.out           64          16  0.000140
     39  blocked_ijk_atlas.out           64          32  0.000140
     40  blocked_ijk_atlas.out           64          64  0.000138
     41  blocked_ijk_atlas.out          128           2  0.018559
     42  blocked_ijk_atlas.out          128           4  0.003531
     43  blocked_ijk_atlas.out          128           8  0.001394
     44  blocked_ijk_atlas.out          128          16  0.001014
     45  blocked_ijk_atlas.out          128          32  0.000939
     46  blocked_ijk_atlas.out          128          64  0.000588
     47  blocked_ijk_atlas.out          128         128  0.000648
     48  blocked_ijk_atlas.out          256           2  0.163757
     49  blocked_ijk_atlas.out          256           4  0.030790
     50  blocked_ijk_atlas.out          256           8  0.011338
     51  blocked_ijk_atlas.out          256          16  0.008187
     52  blocked_ijk_atlas.out          256          32  0.007289
     53  blocked_ijk_atlas.out          256          64  0.004172
     54  blocked_ijk_atlas.out          256         128  0.004046
     55  blocked_ijk_atlas.out          256         256  0.004101
     56  blocked_ijk_atlas.out          512           2  1.307175
     57  blocked_ijk_atlas.out          512           4  0.247788
     58  blocked_ijk_atlas.out          512           8  0.093341
     59  blocked_ijk_atlas.out          512          16  0.063817
     60  blocked_ijk_atlas.out          512          32  0.058434
     61  blocked_ijk_atlas.out          512          64  0.032584
     62  blocked_ijk_atlas.out          512         128  0.030997
     63  blocked_ijk_atlas.out          512         256  0.030732
```

```
64  blocked_ijk_atlas.out         512         512   0.029372
65  blocked_ijk_atlas.out        1024           2  13.861305
66  blocked_ijk_atlas.out        1024           4   3.207080
67  blocked_ijk_atlas.out        1024           8   0.949980
68  blocked_ijk_atlas.out        1024          16   0.530183
69  blocked_ijk_atlas.out        1024          32   0.478315
70  blocked_ijk_atlas.out        1024          64   0.268475
71  blocked_ijk_atlas.out        1024         128   0.251911
72  blocked_ijk_atlas.out        1024         256   0.245424
73  blocked_ijk_atlas.out        1024         512   0.228122
74  blocked_ijk_atlas.out        1024        1024   0.225221
```

2.3 Blocked kij with ATLAS

```
[5]:                  filename  matrix_size  block_size    timing
     75   blocked_kij_atlas.out            2           2  0.000015
     76   blocked_kij_atlas.out            4           2  0.000015
     77   blocked_kij_atlas.out            4           4  0.000015
     78   blocked_kij_atlas.out            8           2  0.000020
     79   blocked_kij_atlas.out            8           4  0.000016
     80   blocked_kij_atlas.out            8           8  0.000017
     81   blocked_kij_atlas.out           16           2  0.000062
     82   blocked_kij_atlas.out           16           4  0.000023
     83   blocked_kij_atlas.out           16           8  0.000017
     84   blocked_kij_atlas.out           16          16  0.000017
     85   blocked_kij_atlas.out           32           2  0.000290
     86   blocked_kij_atlas.out           32           4  0.000075
     87   blocked_kij_atlas.out           32           8  0.000040
     88   blocked_kij_atlas.out           32          16  0.000034
     89   blocked_kij_atlas.out           32          32  0.000030
     90   blocked_kij_atlas.out           64           2  0.002284
     91   blocked_kij_atlas.out           64           4  0.000540
     92   blocked_kij_atlas.out           64           8  0.000202
     93   blocked_kij_atlas.out           64          16  0.000157
     94   blocked_kij_atlas.out           64          32  0.000138
     95   blocked_kij_atlas.out           64          64  0.000128
     96   blocked_kij_atlas.out          128           2  0.017508
     97   blocked_kij_atlas.out          128           4  0.003818
     98   blocked_kij_atlas.out          128           8  0.001504
     99   blocked_kij_atlas.out          128          16  0.001120
     100  blocked_kij_atlas.out          128          32  0.000930
     101  blocked_kij_atlas.out          128          64  0.000595
     102  blocked_kij_atlas.out          128         128  0.000671
     103  blocked_kij_atlas.out          256           2  0.138930
     104  blocked_kij_atlas.out          256           4  0.028898
     105  blocked_kij_atlas.out          256           8  0.011014
     106  blocked_kij_atlas.out          256          16  0.007977
```

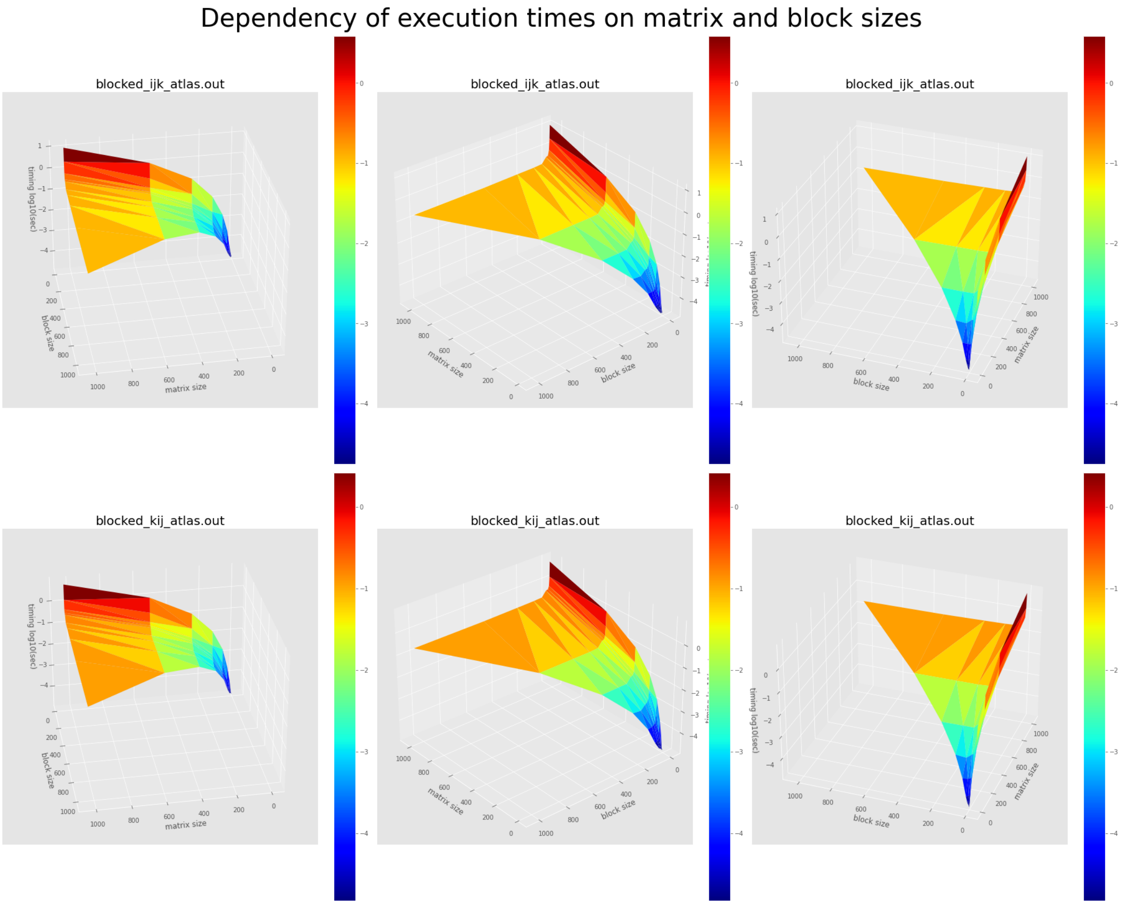
```
107  blocked_kij_atlas.out          256          32  0.007328
108  blocked_kij_atlas.out          256          64  0.004165
109  blocked_kij_atlas.out          256         128  0.004155
110  blocked_kij_atlas.out          256         256  0.004052
111  blocked_kij_atlas.out          512           2  1.107016
112  blocked_kij_atlas.out          512           4  0.228654
113  blocked_kij_atlas.out          512           8  0.087474
114  blocked_kij_atlas.out          512          16  0.063513
115  blocked_kij_atlas.out          512          32  0.057866
116  blocked_kij_atlas.out          512          64  0.032614
117  blocked_kij_atlas.out          512         128  0.030717
118  blocked_kij_atlas.out          512         256  0.030706
119  blocked_kij_atlas.out          512         512  0.030226
120  blocked_kij_atlas.out         1024           2  8.920727
121  blocked_kij_atlas.out         1024           4  1.823004
122  blocked_kij_atlas.out         1024           8  0.701317
123  blocked_kij_atlas.out         1024          16  0.509449
124  blocked_kij_atlas.out         1024          32  0.476489
125  blocked_kij_atlas.out         1024          64  0.269782
126  blocked_kij_atlas.out         1024         128  0.255873
127  blocked_kij_atlas.out         1024         256  0.245888
128  blocked_kij_atlas.out         1024         512  0.229296
129  blocked_kij_atlas.out         1024        1024  0.226316
```

2.4 BLAS dgemm routine

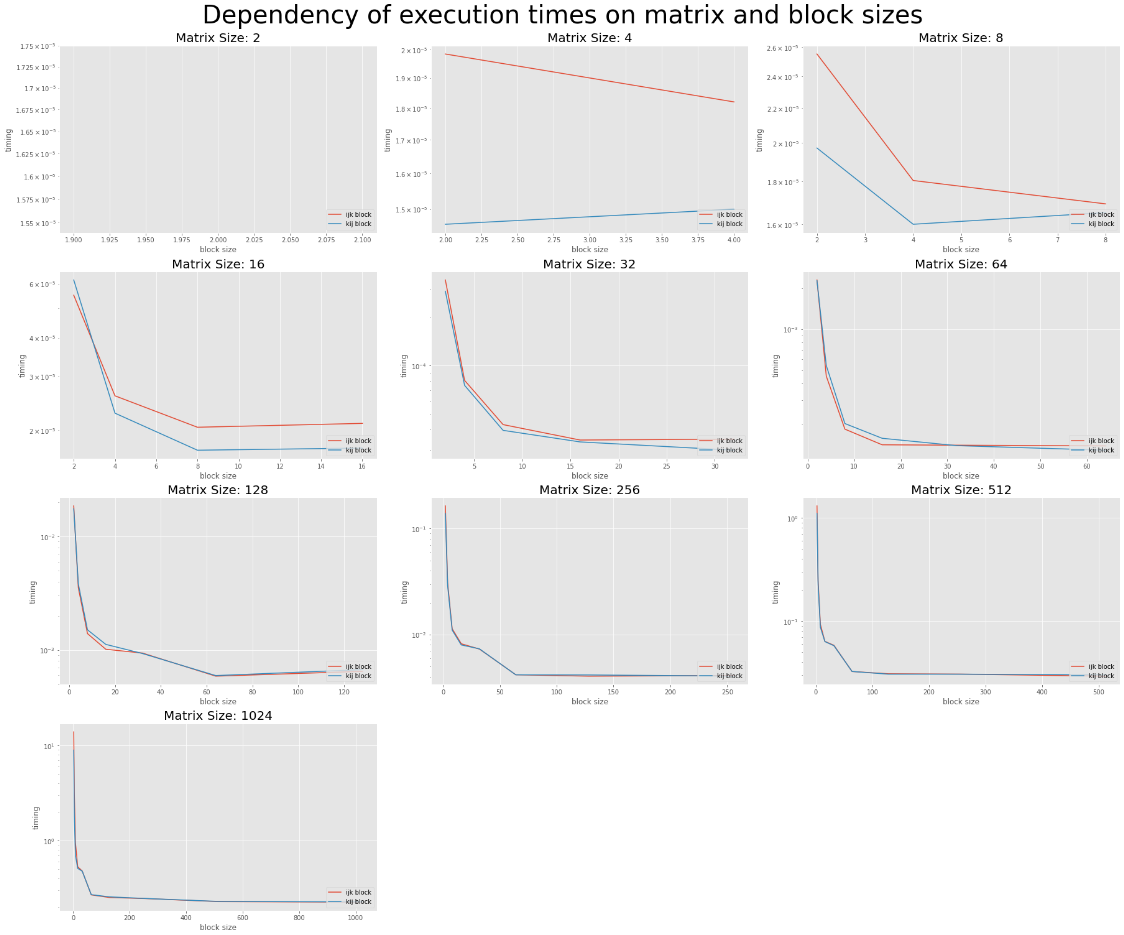
```
[6]:            filename  matrix_size    timing
     10  blas_routine.out            2  0.000023
     11  blas_routine.out            4  0.000025
     12  blas_routine.out            8  0.000027
     13  blas_routine.out           16  0.000026
     14  blas_routine.out           32  0.000039
     15  blas_routine.out           64  0.000144
     16  blas_routine.out          128  0.000635
     17  blas_routine.out          256  0.004034
     18  blas_routine.out          512  0.030143
     19  blas_routine.out         1024  0.225817
```

3 Execution times of blocked algorithms in function of matrix size

We can plot the dependency of the execution on the matrix and block sizes in a 3d plot. I am using a log scale for the execution time since the differences in time are either really small or really big.



We can also plot in 2d for each matrix size:



As we can see from these plots, the two blocked algorithm are similar in shape and in execution times. Blocking appears to not help with performance since there is no significant speedup when the block size is half the size of the matrix

3.1 Speedup of the blocked algorithms over the non-blocked one

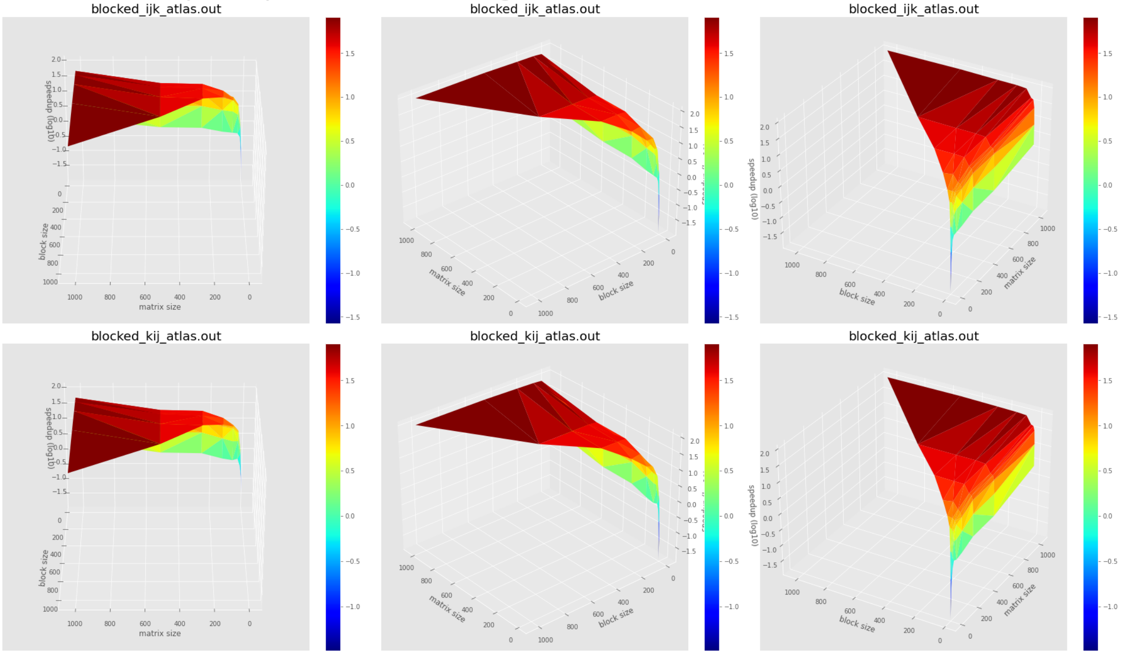
Speedup is computed as following:

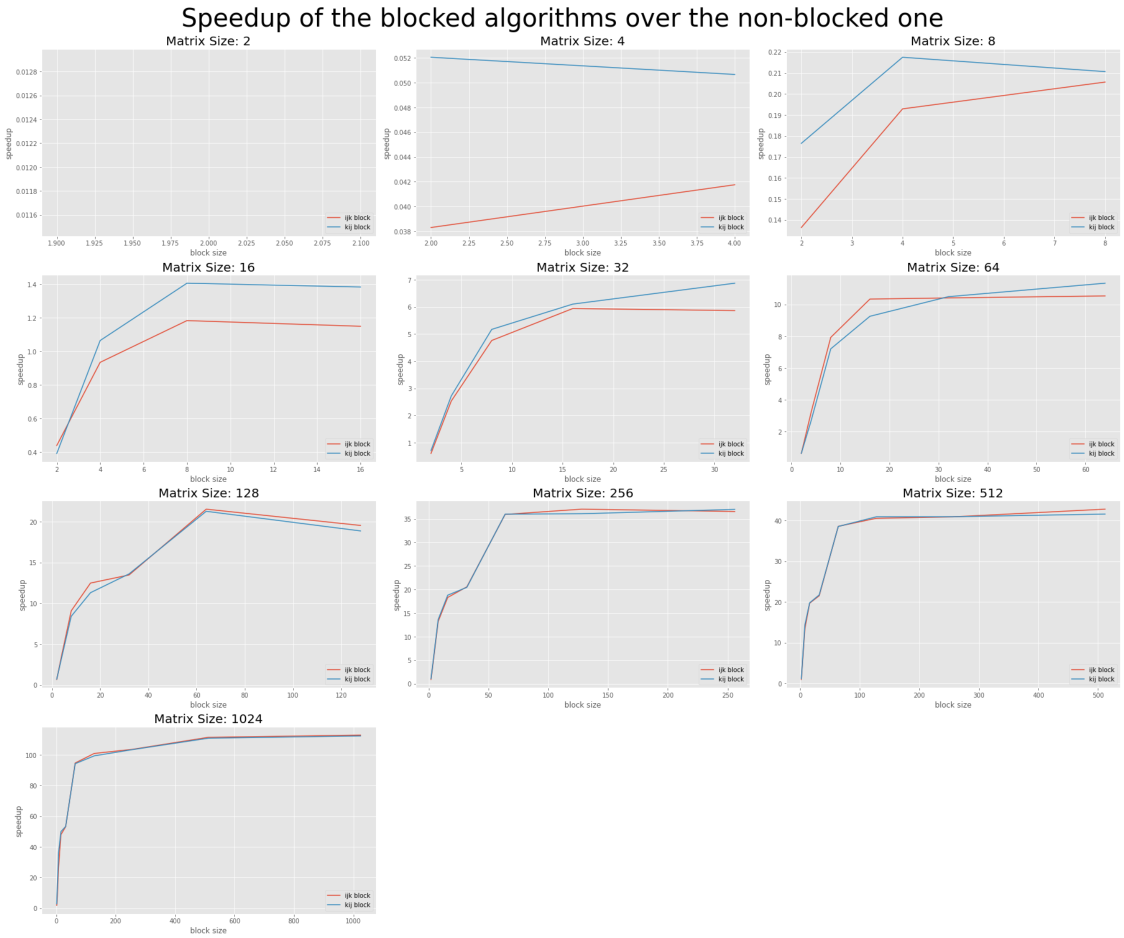
$$S(m, b) = \frac{T_{nonblocked}(m)}{T_{blocked}(m, b)}$$

where m is matrix size and b is block size

We can plot the speedup in 3d and 2d:

Speedup of the blocked algorithms over the non-blocked one



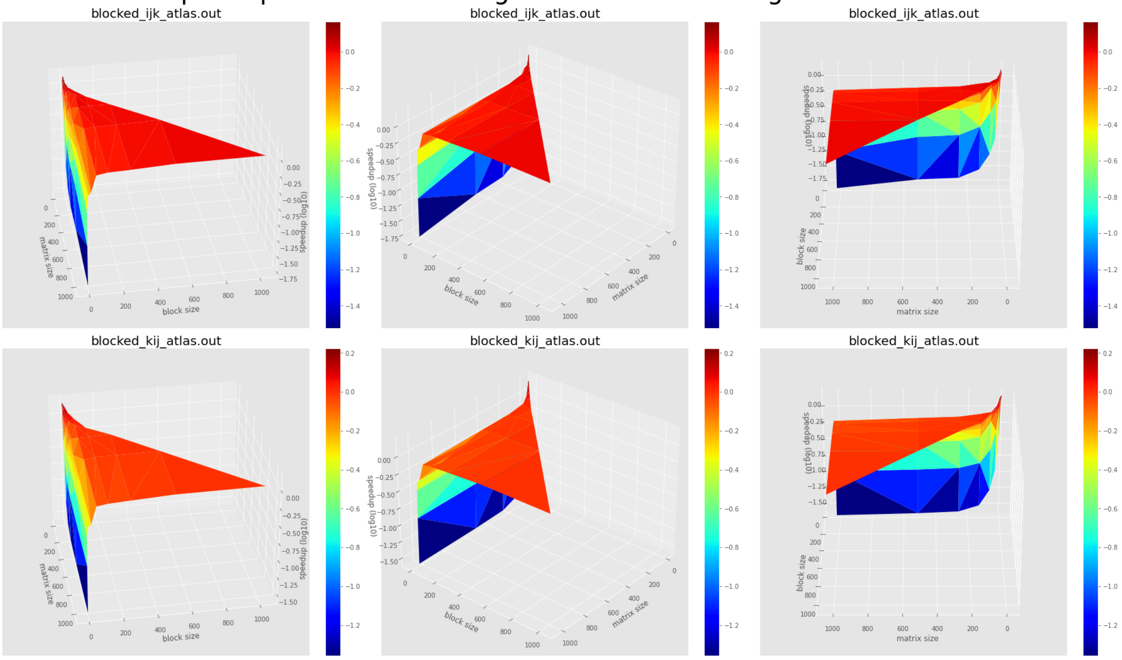


As we can see, the speedup can be up to a factor of 10^1 for large matrices ($m > 64$) and 10^2 for larger ones ($m > 1024$). For smaller matrix, we notice worse performance in comparison with the straightforward algorithm. There is no significant speedup between the ijk and kij algorithm any matrix size. However, I don't think we ca not conclude right now that blocked algorithms are faster than non-blocked ones since we are using ATLAS calls for the blocked algorithms which are already much faster than any naive implementation.

3.2 Comparison with BLAS dgemm

This time, we will compare our blocked algorithms with BLAS dgemm routine.

Speedup of the blocked algorithms over BLAS dgemm routine



blocked_ijk_atlas.out

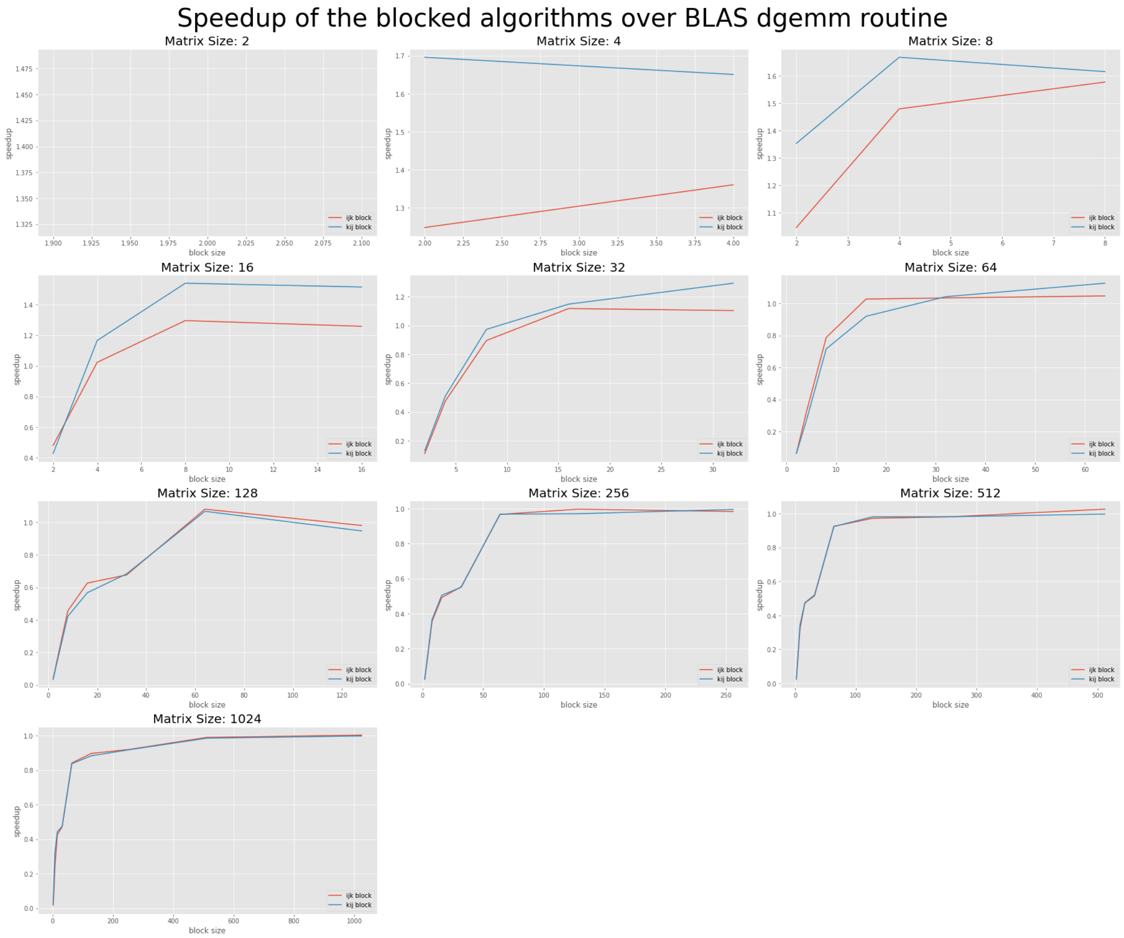
blocked_ijk_atlas.out

blocked_ijk_atlas.out

blocked_kij_atlas.out

blocked_kij_atlas.out

blocked_kij_atlas.out



The speedup is generally under 1 for matrices with $m > 64$ and any block size. Blocking appears to lower the performance for matrix larger than 64x64 when using ATLAS calls. I expected blocking algorithms to give better performance since they reduce cache misses by reusing the same small blocks frequently. One reason I can think of why blocking in this case doesn't improve the performance is the `dgemm` routine's cost of overhead makes the multiple calls inefficient for small matrices. It means that for N_b number of blocks, we have $N_b.t_{overhead} + t_{procblocked} < t_{proc}$.