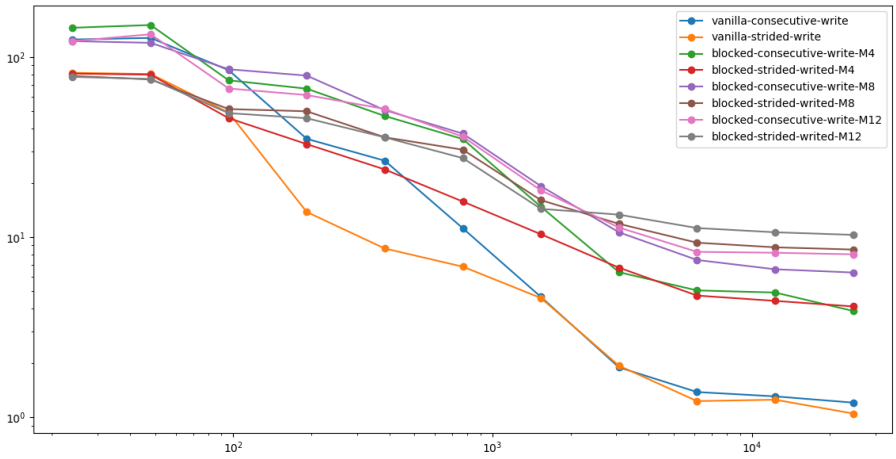
Exercise 2

1. The following figure shows the result on ijk-form and 4*4, 8*8, 12*12 blocks.



We can see:

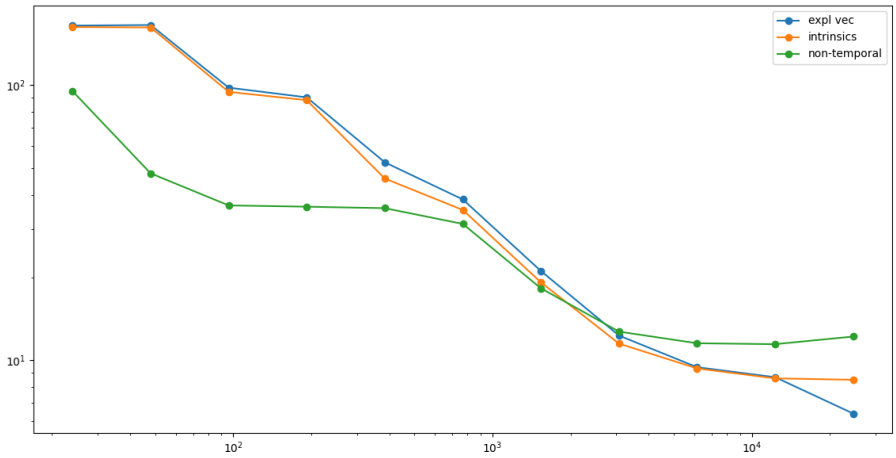
8*8 and 12*12 blocks perform better than 4*4 blocks and ijk-form.

Consecutive-write performs better with a smaller n.

Strided-write performs better with a larger n.

This transfer happens between n equals to 1536 (18MB) and 3072 (72MB), and I guess this is because the size of L3 cache (24MB)

2. The following figure shows the result on 3 different modes.



Here we can still find that non-temporal method becomes faster after n=3072 (72MB). It is also because the size of matrix is larger than the size of L3 cache (24MB) since then.

Memory alignment:

Computer's processor does not read from and write to memory in byte-sized chunks. Instead, it accesses memory in two-, four-, eight- 16- or even 32-byte chunks. So, if our

data has a data type that takes up, say, 32 bytes, we want its address to be aligned with the boundaries of the 32-byte chunk so that the data can be stored in a chunk.

If the memory is not properly aligned, the processor needs to read the first chunk of the unaligned address and shift out the bytes we don't need from the first chunk. Then it needs to read the second chunk of the unaligned address and shift out some of its information. Finally, the two are merged together for placement in the register. It's a lot of work.

In this experiment, we should always read or write several vectors with 4 or 8 doubles. So, it will take more time for processer to do what is described above.