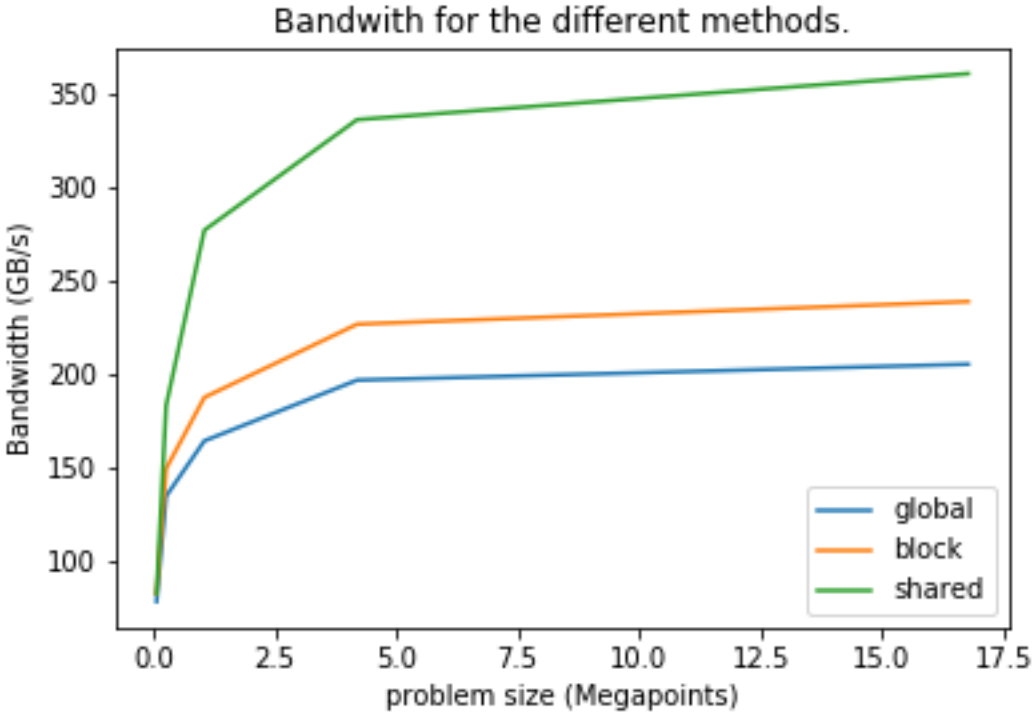
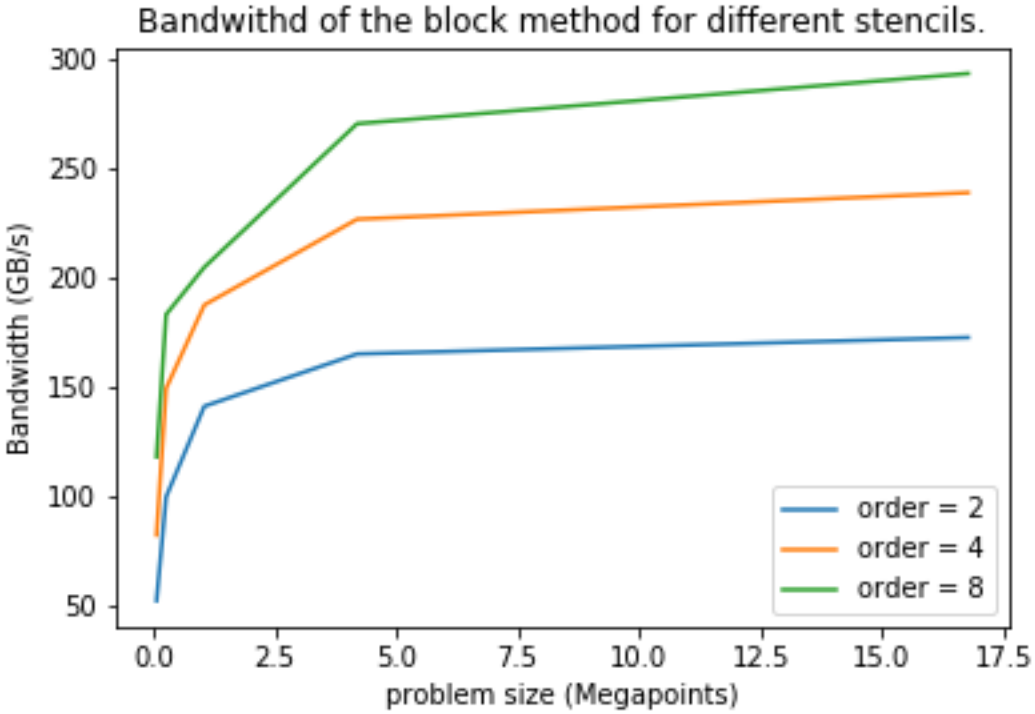
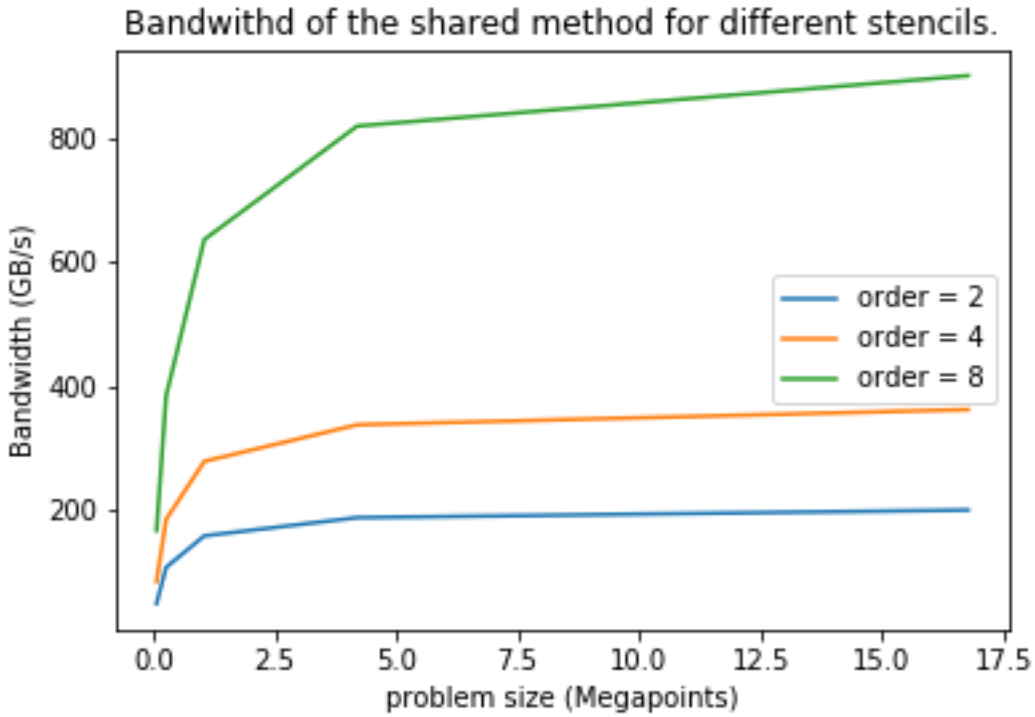
Question 3


Bandwith for the different methods.
Bandwidth (GB/s)
350
300
250
200
150
100
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
problem size (Megapoints)
global
block
shared


Bandwithd of the block method for different stencils.
Bandwidth (GB/s)
300
250
200
150
100
50
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
problem size (Megapoints)
order = 2
order = 4
order = 8



Question 4

Except for the smallest grid size the performance in order from worst to best is global, block and shared. For the lowest grid size not all the resources cannot be utilized simultaneously, to the idle time as a fraction of the total time will be larger. This is why the different methods perform the same for the smaller grid sizes. The higher order methods perform better than the lower order ones due to the memory copying needed to for a single mesh point. Because any memory request will copy an entire cache line no matter how many elements are actually used. The higher order methods will use more of the elements in the row of the stencil, which is coalesced in memory, and therefore make more efficient use of cache lines. Since the bandwidth calculation depends on stencil size the bandwidth will increase for higher order methods.

The block method will perform better than the global method because the there is less memory traffic in total. In the global method each thread will request the memory for its operation after which the allocted memory in the cache may dissapear. In the block method one thread will perform multiple operations. This keeps the requested memory in the cache for longer which will make access to it faster for the thread and other threads in the warp. This is the same reason why the shared method performs best. In this method all the memory required by a single block is stored locally and remains there until the block is finished. This significantly reduces the memory traffic and explains why the shared memory so much faster.