Advection in one-dimensional space

Problem: Find the scalar field $u = u(x,t)$ governed by the equation:

$$\begin{cases} \dfrac{\partial u}{\partial t} + 2\pi \dfrac{\partial u}{\partial x} = 0, & x \in [0, 2\pi], \\ u(x,0) = \sin(x), \\ u(0,t) = -\sin(2\pi t). \end{cases}$$

Exact solution: $u(x,t) = \sin(x - 2\pi t)$.

Run times of dg-on-cuda: CPU (serial) execution on Nvidia Jetson Xavier NX (Carmel ARMv8.2 64-bit 6MB L2 + 4MB L3):

	1024 elements						8192 elements					
Approx. order	1	2	3	4	5	6	1	2	3	4	5	6
Time (ms)	2802	3628	4726	6366	8670	10436	18593	27856	37669	47379	61880	82965

GPU execution on Nvidia Jetson Xavier NX (Volta GPU with 384 CUDA cores) with no optimizations:

	1024 elements						8192 elements					
Approx. order	1	2	3	4	5	6	1	2	3	4	5	6
Time (ms)	5070	5240	5480	6910	9849	10732	6142	8250	10559	13858	19193	22060

GPU execution with coalesced global memory access:

	1024 elements						8192 elements					
Approx. order	1	2	3	4	5	6	1	2	3	4	5	6
Time (ms)	4602	4354	5947	7510	9622	10959	6510	7565	10298	13446	18600	21166

The reason coalesced access did not significantly improve the performance is that there is only up to 7 DOFs in an element and even without coalesced layout the situation is not too bad. The improvement is expected to be much more significant in 2D and 3D problems as each element can have much more DOFs, resulting in much worse situation in non-coalesced access.

Moving the matrices (for volume and surface integration) to constant memory for some reason did not improve much either:

	1024 elements						8192 elements					
Approx. order	1	2	3	4	5	6	1	2	3	4	5	6
Time (ms)	4648	4321	5432	6841	9440	10627	6100	7529	10144	13373	18594	21184

The block size of running all tests above was set to 1024. The table below shows how different block size could impact the performance (tested with 8192 elements of approximation order 6).

Block size	16	32	64	128	256	512	1024
Time (ms)	26971	15806	15861	15816	16747	16880	21194