Abstract
We present a SIMD implementation of the pseudo-Verlet list neighbour search algorithm [1]. We have applied our strategy to the SPH-based cosmological code SWIFT [2]. We report an overall speed-up over the scalar version of 2.24x, 2.43x and 4.07x for AVX, AVX2 and AVX-512 instruction sets respectively.

Problem
Particle-based simulations evolve a system of particles via a set of pair-wise interactions. Short-range interactions are evaluated by computing pair-wise distances between particles and checking that they lie within a cut-off radius, \( h \), of each other. Efficient neighbour finding is performed using a mesh of cells with edge length matching the particle cut-off radius. Particles are then interacted within pairs of neighbouring cells (See Fig. 1).

Solution
Neighbour finding is the most time consuming part of the computation. This time can be reduced by using the pseudo-Verlet list algorithm (See Fig. 2):
1. Project the particles onto the axis joining the centres of the two cells and sort them along the axis
2. Calculate the separation between particles in the left-hand cell \( a \), \( p_i^{(a)} \), and particles in the right hand cell \( b \), \( p_j^{(b)} \), that lie within \( h \), the cut-off radius of \( p_i^{(a)} \), on the sorted axis.
3. If any particles \( p_j^{(b)} \) satisfy the criterion \( \| p_i^{(a)} - p_j^{(b)} \|_2 < h \), an interaction is computed.

SIMD Implementation
The overall vectorisation strategy involved picking one particle, \( p_i^{(a)} \), from cell \( a \) and interacting it with a vector length of, \( p_j^{(b)} \), particles that are within range from cell \( b \). The contributions from the \( p_j^{(b)} \) that are not within range are masked out.

The following optimisations were also performed:
- Place particles into a cache formed from a SoA (Structure of Arrays)
- Sort the cache and only populate it with particles that are within range
- Make use of horizontal add intrinsics to lower memory access
- Limit the number of particles looped over (See Fig. 2)

Conclusion
We presented an efficient SIMD implementation of the pseudo-Verlet list algorithm. When implemented in the SWIFT code using the AVX, AVX2 and AVX-512 instruction sets this algorithm reached speed-ups of 2.24x, 2.43x and 4.07x respectively when compared to a scalar version.

Results
The results displayed below show the speed-up gained from each instruction set.

Table 1: Median times and corresponding vectorization speed-ups for the idealised case of one particle directly interacting with 2500 other particles without any distance checks.

<table>
<thead>
<tr>
<th>Machine Name</th>
<th>CFLAGS</th>
<th>Scalar Time (ms)</th>
<th>Vectorised Time (ms)</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>COSMA5</td>
<td>-avX2</td>
<td>0.048</td>
<td>0.0084</td>
<td>5.68x</td>
</tr>
<tr>
<td>Hamilton</td>
<td>+x08E+XVX2</td>
<td>0.038</td>
<td>0.0005</td>
<td>67.7x</td>
</tr>
<tr>
<td>Kyll</td>
<td>-avX12+AVX512</td>
<td>0.170</td>
<td>0.0079</td>
<td>21.30x</td>
</tr>
</tbody>
</table>
