7 Scalability testing

7.1 Summary of the obtained results from the scalability testing

This was our first time exploring the usage of AMD GPUs, with the added difficulty of environment issues (which are now solved) and Cray compiler bugs which needed a few workarounds. As such, we concentrated on verifying the system scaling for a single test. Typical use cases are smaller than that, and will be explored as part of future access calls.

It was a bit concerning that GPU calculation with more than 4 nodes would freeze indefinitely; this severely limited the scope of the scalability testing. However, we expect this limitation to disappear in future updates for both LUMI, and the library we use as a GPU interface, ELPA. That said, we can estimate that a single GPU was roughly equivalent to 8 CPUs.

7.2 Images or graphics showing results from the scalability testing

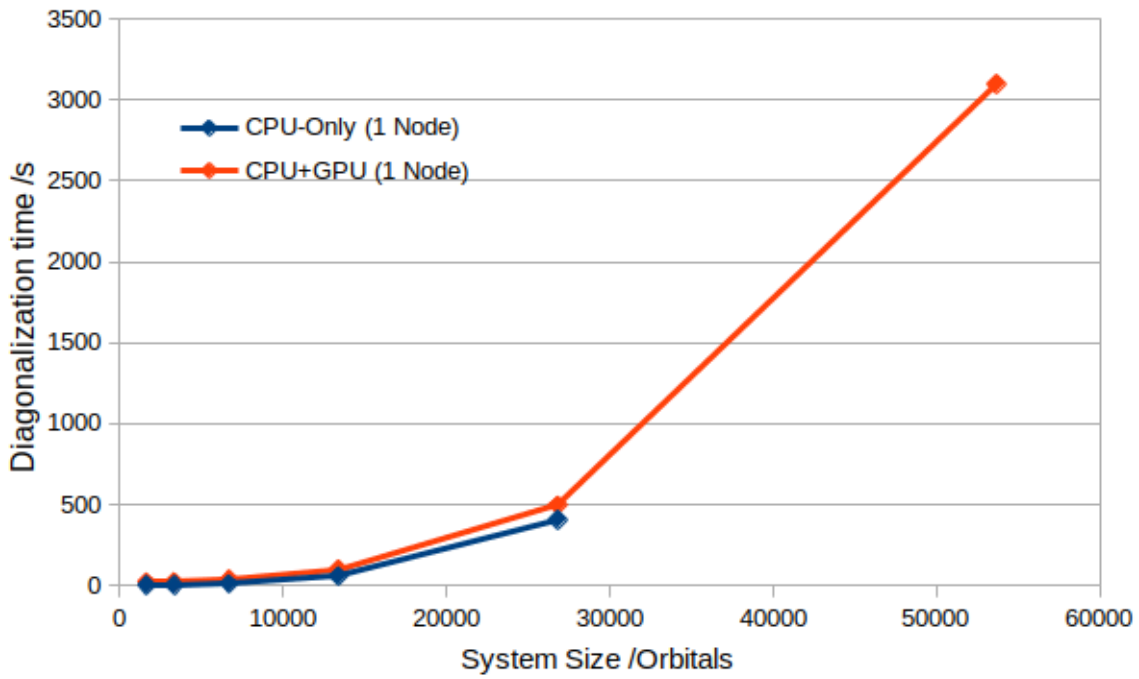

Figure 1. Sample increase in cost for the gold-water system for CPU-GPU calculations on a LUMI-G node (128 CPU cores, 8 GPUs) in comparison with CPU-only calculations on a LUMI-C node (64 CPU cores). It shows the cubic scaling of the diagonalization and density matrix building part of the code, which covers 95% of the total time. CPU+GPU calculations were able to achieve a larger system due to the extra memory available on standard LUMI-G nodes (512 GB vs 256 GB on standard LUMI-C nodes).

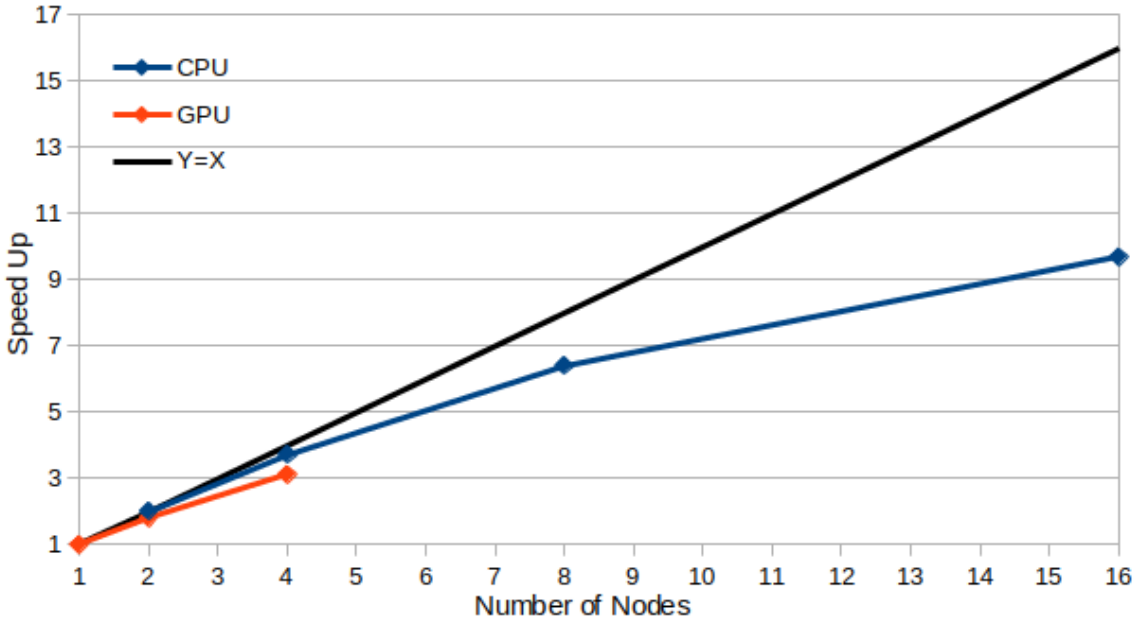

Figure 2. Strong scaling for the gold-water system consisting of 54k orbitals (see data below) on LUMI-G in comparison with the strong scaling on LUMI-C. GPU calculations for more than 4 nodes were not possible because simulations froze indefinitely.

7.3 Data to deploy scalability curves (if applicable)

A. Diagonalization scaling cost - CPU (LUMI-C)

Number of orbitals	Wall clock time (s)
1,677	0.936
3,354	2.524
6,708	10.286
13,416	59.578
26,832	408.528

Table 1. Diagonalization scaling for a Gold-Water system. These tests were performed on 128 CPU cores, using a single node from the LUMI-C partition.

B. Diagonalization scaling cost - CPU+GPU (LUMI-G)

Number of orbitals	Wall clock time (s)
1,677	22.109
3,354	24.076
6,708	36.682
13,416	101.734
26,832	496.756
53,664	3,097.579

Table 2. Diagonalization scaling for a Gold-Water system. These tests were performed in 64 CPUs plus 8 GPUs, using a single node from the LUMI-G partition.

C. Strong scaling curve - CPU (LUMI-C)

Number of cores	Wall clock time (s)	Twice the speed-up vs 2 nodes	Number of nodes	Number of processes
256	1,605.494	2.00	2	256
512	863.469	3.72	4	512
1024	501.303	6.41	8	1024
2048	331.869	9.68	16	2048

Table 3. Strong scaling for a Gold-Water system containing 53664 orbitals. These tests were performed in nodes consisting of 128 CPU cores, from the LUMI-C CPU partition. Note that the simulation didn't fit in a single LUMI-C node, hence the definition of the speed-up vs. the calculation on two nodes.

D. Strong scaling curve - GPU (LUMI-G)

Number of cores	Number of GPUs	Wall clock time (s)	Speed-up vs one node	Number of nodes	Number of processes
64	8	3,097.579	1.00	1	63
128	16	1,718.730	1.80	2	126
256	32	995.097	3.11	4	252

Table 4. Strong scaling for a Gold-Water system containing 53,664 orbitals. These tests were performed in nodes consisting of 64 CPUs plus 8 GPUs, from the GPU partition.

7.4 Publications or reports regarding the scalability testing

Please use the following format: Author(s). "Title". Publication, volume, issue, page, month year.

None to date. Please note that information about the deployment and testing of SIESTA on EuroHPC machines in general is/will be available in the project repository of the MaX Centre of Excellence (https://www.max-centre.eu/project-repository) and on the SIESTA benchmarks page (https://siesta-project.org/siesta/benchmarks/).

8 Results on Input/Output

8.1 Size of the data and/or the number of files

Please fill in the information in the box below (maximum 300 words).

Outputs for these tests are relatively small. In the larger test cases, total output was around 300MB of data across 30-40 output files, most of which are not actually needed to evaluate performance.

8.2 Usage of MPI-IO features, if applicable

Please fill in the information in the box below (maximum 300 words).

We did not explore MPI-IO features since this is not a limiting factor in our code, even though we do support it via NetCDF.