7 Scalability testing

7.1 Scalability testing on VEGA-GPU

7.1.1 Summary of the obtained results from the scalability testing

Please show the scaling behavior of the application. Which progress was achieved? Does it fulfill the set expectations? If not, what were the reasons? (maximum 500 words)

Regarding the CPU-only calculations, results are as expected, showing the good strong scaling of SIESTA (even if not perfect) when going into 1k processes. The GPU calculations were trickier since this was our first time accessing this amount of resources; in fact, some of the test cases we had prepared proved to be *too small* to be useful. This is why some of the results herein might differ from those presented in the original proposal.

When compared to CPU-only calculations, GPU acceleration was an undeniable improvement. A single GPU was roughly equivalent to something between 16 and 32 CPUs. That said, setting the proper CPU bindings and understanding how to maximize GPU usage required a lot of testing that is not shown in this report.

One thing that these tests showed is the need to have a special look into CPU-GPU communications. The disparity shown in scalability plots for two systems of similar sizes can only be explained by differences in the parallelization. In any case, except for the very small systems, scalability for GPU was better than expected and we considered these tests to be extremely successful.

7.1.2 Images or graphics showing results from the scalability testing

All tables and figures (including photographs, schemas, graphs and diagrams) should be numbered with Arabic numerals (1, 2,...n) and include a descriptive caption. Please attach the images to this form (minimum resolution 300 dpi).

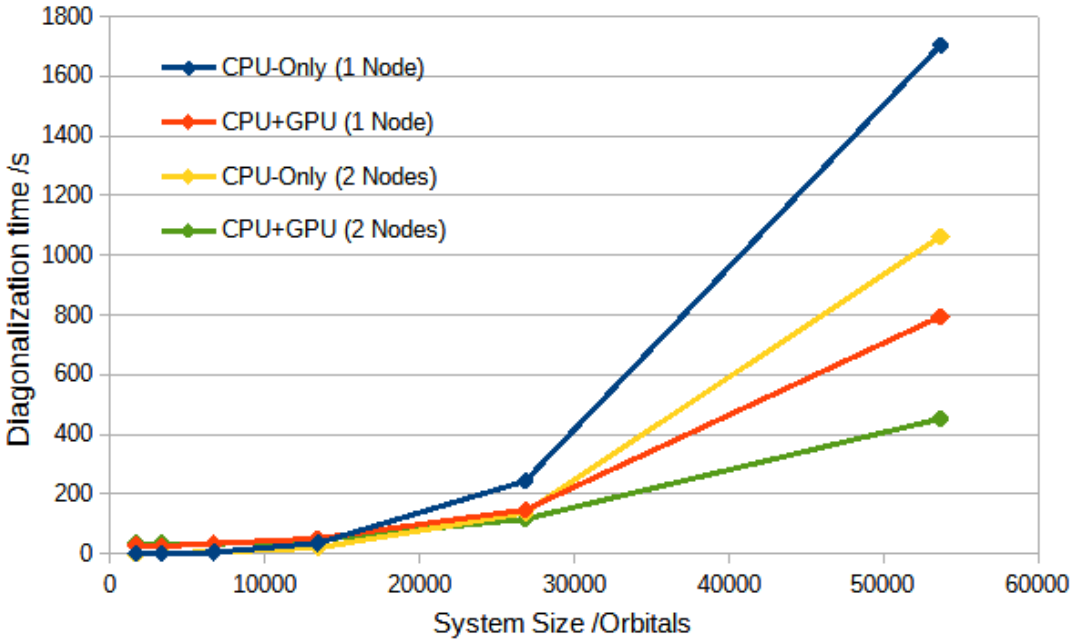

Figure 1. Sample increase in cost for the gold-water system with different configurations. This is just to show the cubic scaling of the diagonalization and density matrix building part of the code, which covers 95% of the total time.

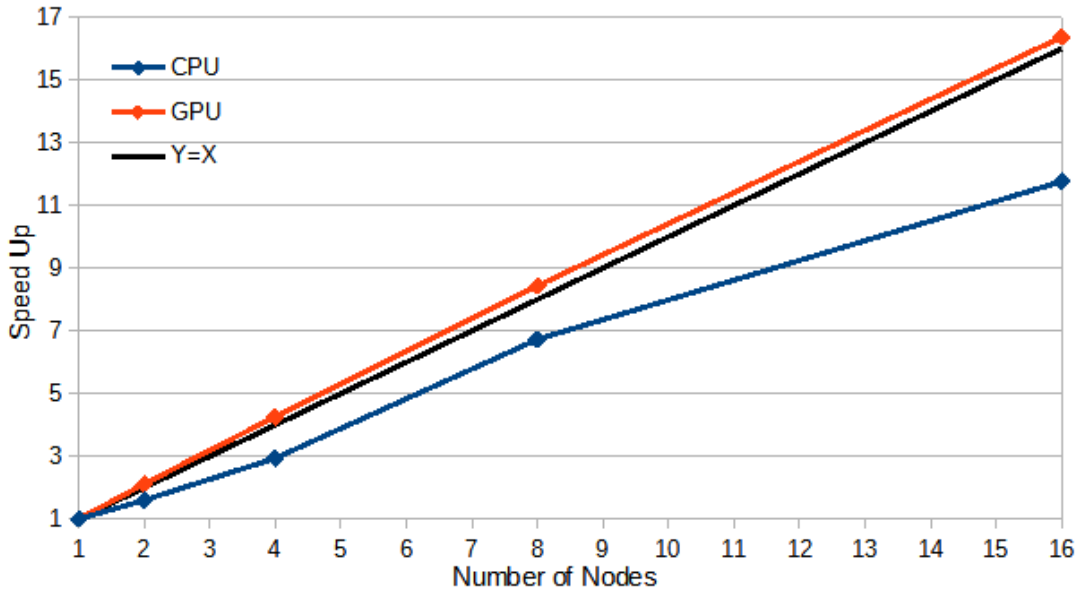

Figure 2. Strong Scaling for the gold-water system consisting of 54k orbitals (see data below). The GPU scaling being slightly *above* the X=Y line is curious but might be the consequence of numerical errors due to the timescale of the calculations.

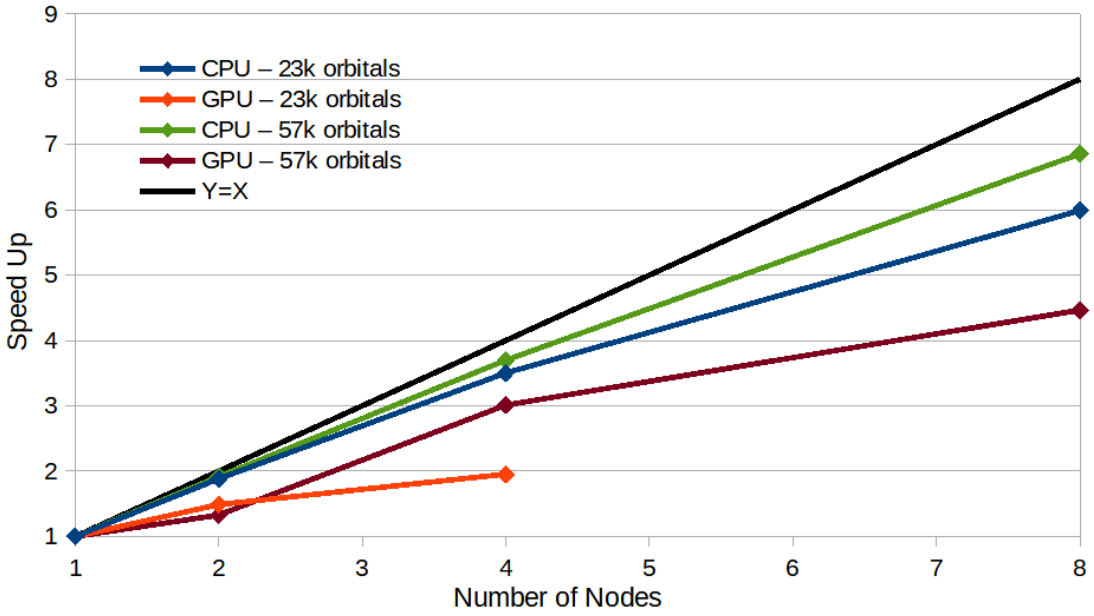

Figure 3. Strong Scaling for the Covid Spike protein system with two basis sets: SZ (23k orbitals) and SZP (57k orbitals). The under-performance of the GPU scalability in this case can be explained due to the under-saturation of the accelerator; in fact , the reason that the GPU line goes only up to 4 nodes for the smallest case is due to the fact that the system was *too small* to go higher.

7.1.3 Data to deploy scalability curves

A. Typical user test cases

Please include the data for each test case.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
64	350	1	1	64
128	200	1.75	2	128

Table 1. Typical user test cases. They were run as CPU-only tests. The tests consist of an SCF calculation for a gold slab with water molecules, with a total of 13k orbitals.

B. Strong scaling curve

Please include the data in order to deploy the scalability curve when the number of processors varies for a fixed total problem size.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
64	1703.906	1.00	1	64
128	1063.340	1.60	2	128
256	579.910	2.94	4	256
512	253.262	6.73	8	512
1024	144.893	11.76	16	1024

Table 2. Strong scaling data for CPU-only calculations. Gold-water system composed of 54k orbitals.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
64	3507.139	1.00	1	64
128	1830.331	1.92	2	128
256	948.456	3.70	4	256
512	511.505	6.86	8	512

Table 3. Strong scaling data for CPU-only calculations. COVID Spike protein composed of 58k orbitals.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
64	794.809	1.00	1	64
128	452.493	1.76	2	128
256	285.484	2.78	4	256
512	189.691	4.19	8	512
1024	141.541	5.62	16	1024

Table 4. Strong scaling data for CPU-GPU calculations. Gold-water system composed of 54k orbitals.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
64	794.268	1.00	1	64
128	598.506	1.33	2	128
256	263.879	3.01	4	256
512	177.884	4.47	8	512

Table 5. Strong scaling data for CPU-GPU calculations. COVID Spike protein composed of 58k orbitals.

C. Weak scaling curve

Please include the data in order to deploy the scalability curve when the number of processors varies for a fixed problem size per processor.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes	System Size (orbitals)
64	244.038	1.00	1	64	27k
128	1.063.340	0.22	2	128	54k

Table 6. Weak scaling example for gold-water surface, CPU-only calculation.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes	System Size (orbitals)
64	147.276	1.00	1	64	27k
128	452.493	0.33	2	128	54k

Table 7. Weak scaling example for gold-water surface, CPU-GPU calculation.

Doing full weak scaling plots for SIESTA does not make a lot of sense since they are just the composition of the computational cost (which scales cubically, as shown in 7.1.2) and the scalability shown under the strong scaling section (7.1.3.B).

7.1.4 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).

7.2 Scalability testing on LUMI-C

7.2.1 Summary of the obtained results from the scalability testing

Please show the scaling behavior of the application. Which progress was achieved? Does it fulfill the set expectations? If not, what were the reasons? (maximum 500 words)

Tests on LUMI-C were focused on the usage of the PEXSI algorithm, which, while more expensive than Scalapack regular diagonalization, has the advantage of lower scaling (linearly for extremely large systems, quadratic otherwise) and lower memory usage.

We did find the CPUs to be a bit underwhelming, since twice as much CPUs were needed to achieve similar results to those in VEGA. However, it is possible that compiler and library optimizations play a much larger role here, rather than the CPUs themselves.

We could not, unfortunately, scale to more than 32 nodes, and in some cases the limit was 16. Calculations either crashed with cryptic errors or the processes simply hanged idly; we suppose MPI communication was partly to blame for this behaviour.

The scalability itself started degrading, in general, after 1024 cores. We consider this a good result for regular calculations, but we did expect much better performance for the systems tested here.

We expect to repeat these tests, and use new benchmarks, in 2024, with the hope that a newer and better software stack in LUMI-C will greatly improve them.

7.2.2 Images or graphics showing results from the scalability testing

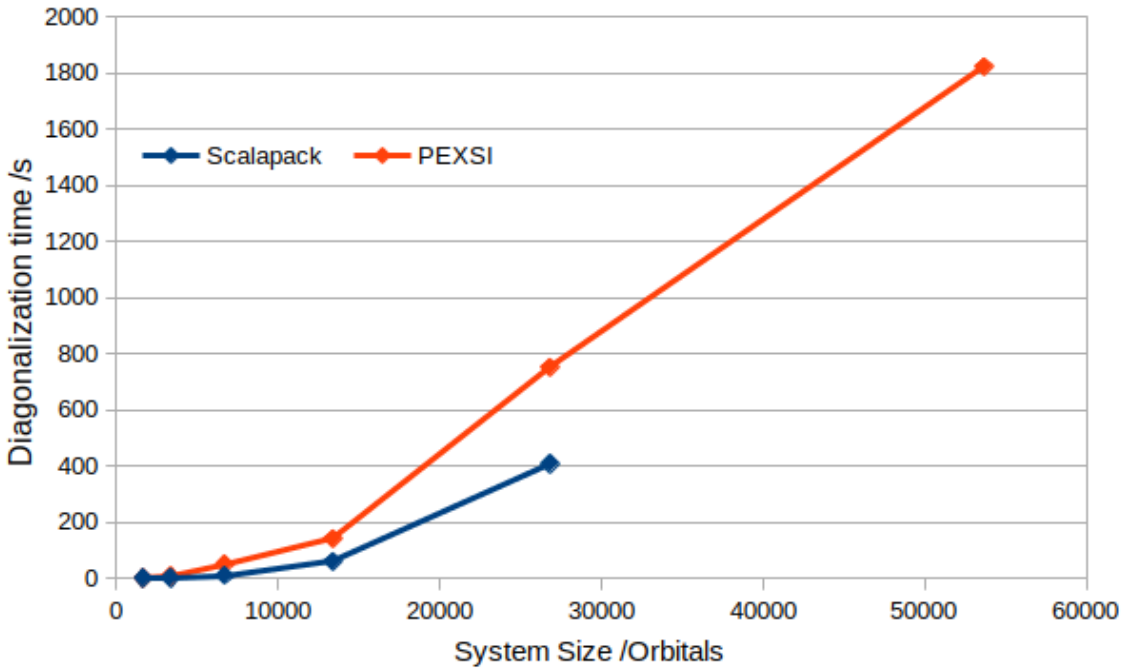

Figure 4. Sample increase in cost for the gold-water system using the standard Scalapack diagonalization and the PEXSI algorithm. This timing corresponds to just the building of the density matrix from the Hamiltonian, which accounts for 95% of the computational time. For Scalapack, it was not possible to achieve 54k orbitals in a single node due to memory limitations.

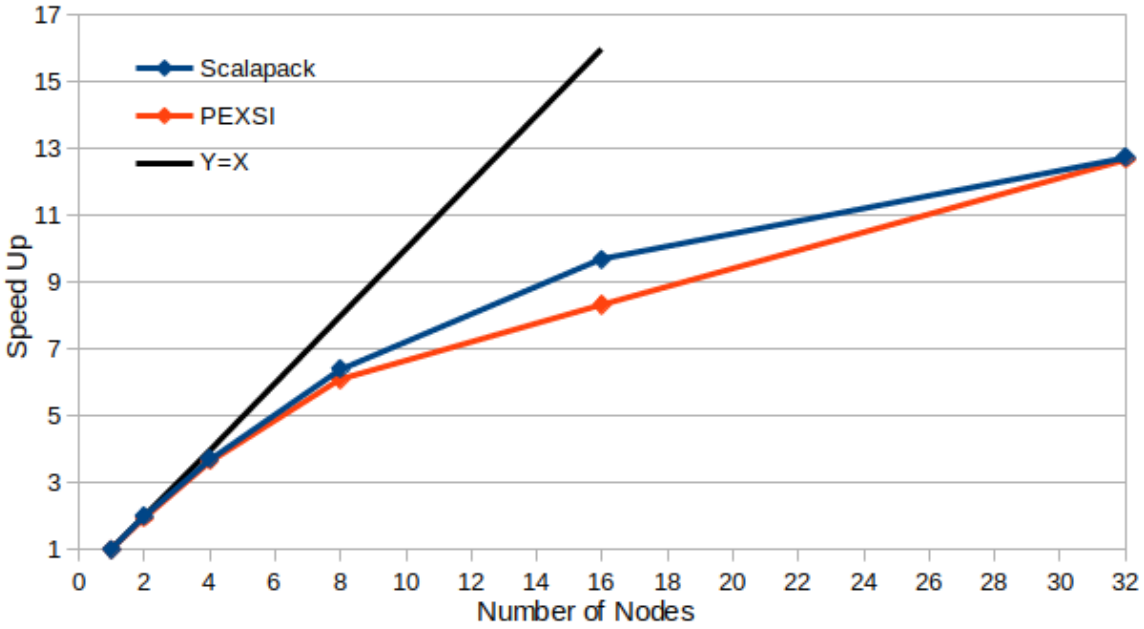

Figure 5. Strong Scaling for the gold-water system consisting of 54k orbitals (see data below).

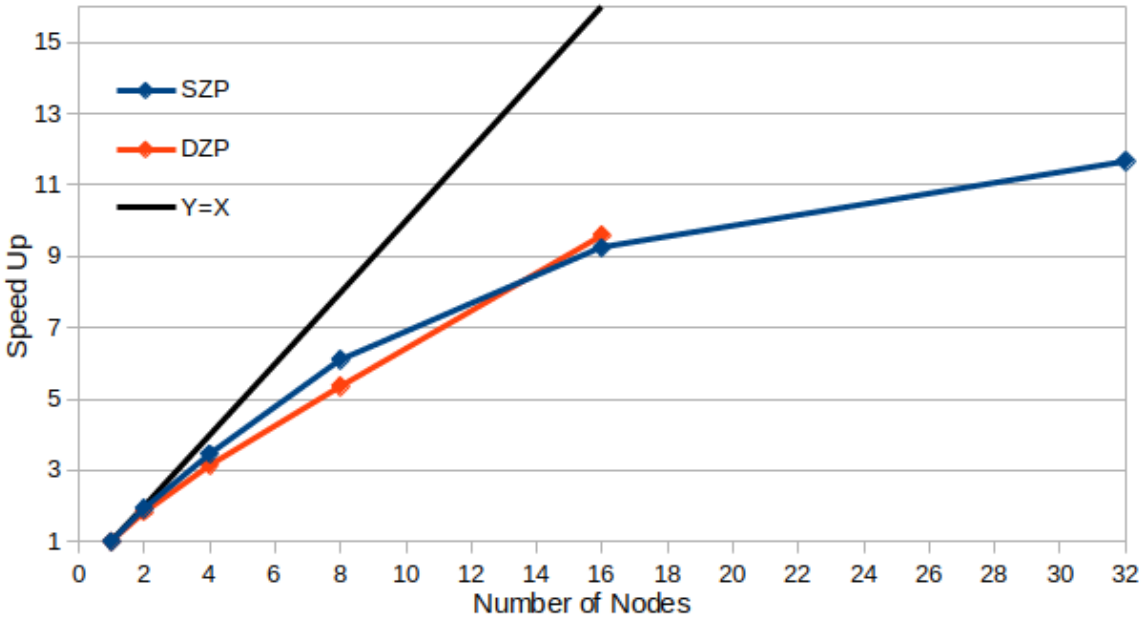

Figure 6. Strong Scaling for the COVID Spike Protein consisting of 57k orbitals (SZP) and 80k orbitals (DZP). Calculations for the DZP system with 32 nodes were not possible due to constant crashes, probably related to the amount of communications needed.

7.2.3 Data to deploy scalability curves

D. Typical user test cases

Please include the data for each test case.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
128	751.588	1.00	1	128
256	447.561	1.68	2	256

Table 8. Typical user test casesThese typical user test cases were run as CPU-only tests. The tests consist of an SCF calculation for a gold slab with water molecules, with a total of 27k orbitals, using the PEXSI algorithm.

E. Strong scaling curve

Please include the data in order to deploy the scalability curve when the number of processors varies for a fixed total problem size.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
256	1605.494	1.00	2	256
512	863.469	1.86	4	512
1024	501.303	3.20	8	1024
2048	331.869	4.84	16	2048
4096	252.084	6,37	32	4096

Table 9. Strong scaling for a system of gold-water of 54k orbitals, using the standard diagonalization algorithm. It was not possible to use a single node since the program ran out of memory.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
128	1821.075	1.00	1	128
256	933.966	1.95	2	256
512	497.070	3.66	4	512
1024	299.411	6.08	8	1024
2048	219.093	8.31	16	2048
4096	143.594	12.68	32	4096

Table 10. Strong scaling for a system of gold-water of 54k orbitals, using the PEXSI algorithm.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
128	1110.454	1.00	1	128
256	574.538	1.93	2	256
512	321.779	3.45	4	512
1024	182.038	6.10	8	1024
2048	120.209	9.24	16	2048
4096	95.197	11.66	32	4096

Table 11. Strong scaling for the COVID spike protein with an SZP basis set, with 57k orbitals, using the PEXSI algorithm.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes
128	2166.059	1.00	1	128
256	1180.194	1.84	2	256
512	690.729	3.14	4	512
1024	404.247	5.36	8	1024
2048	225.786	9.59	16	2048

Table 12. Strong scaling for the COVID spike protein with an DZP basis set, with 80k orbitals, using the PEXSI algorithm.

F. Weak scaling curve

Please include the data in order to deploy the scalability curve when the number of processors varies for a fixed problem size per processor.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes	System Size (orbitals)
128	408.528	1.00	1	128	27k
256	1605.494	0.51	2	256	54k

Table 13. Weak scaling example for gold-water surface, using the regular diagonalization algorithm.

Number of cores	Wall clock time (s)	Speed-up vs the first one	Number of nodes	Number of processes	System Size (orbitals)
128	751.588	1.00	1	128	27k
256	933.966	1.61	2	256	54k

Table 14. Weak scaling example for gold-water surface, using PEXSI.

Doing full weak scaling plots for SIESTA does not make a lot of sense since they are just the composition of the computational cost (which scales cubically for regular diagonalization and semi-quadratically for PEXSI, as shown in 8.2) and the scalability shown under the strong scaling section (8.3 B).

7.2.4 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).

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.