HORIZON-EUROHPC-JU-2021-COE-01

Centre of Excellence in Exascale CFD

CEEC – Centre of Excellence in Exascale CFD

Grant Agreement Number: 101093393

D1.2 – Definition of a common benchmark test case

WP1: Exascale light-house cases

EuroHPC Joint Undertaking

Document Information

Deliverable Number	D1.2
Deliverable Name	Definition of a common benchmark test case
Due Date	31/12/2023 (PM 12)
Deliverable lead	BAM
Authors	Samuel Kemmler (BAM)
Responsible Author	Samuel Kemmler (BAM), samuel.kemmler@bam.de
Keywords	code capabilities, benchmark problems, performance metrics
WP	WP1
Nature	R
Dissemination Level	PU
Final Version Date	31/12/2023
Reviewed by	Tim Felle Olsen (DTU), Niclas Jansson (KTH)
MGT Board Approval	31/12/2023

Acknowledgment:

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark under grant agreement No 101093393.

Document History

Partner	Date	Comment	Version
WP1 contributors	06/12/23	Initial version	0.1
DTU	07/12/23	Revision	0.1
WP1 contributors	20/12/23	Revised version	0.2
BAM	21/12/23	Final version	1.0

Executive Summary

One of the primary goals within the scope of CEEC is improving the performance and energy efficiency of the five European HPC partner codes, driven by the collection of the six different light-house cases (LHCs). The deliverable D1.1 provides more details regarding the LHCs [1]. To achieve this goal, we face the dilemma that, on the one hand, the final LHCs are only fully developed towards the end of the project, but, on the other hand, the work on the performance and energy efficiency in WP2/3/4 should be able to start immediately. To overcome this dilemma, we introduce a set of benchmarks covering the key functionalities of the LHCs, allowing the performance optimizations in WP2/3/4 to start immediately based on the benchmarks. Each code/LHC pair has collaboratively developed one or more benchmarks that are common between the corresponding two partners steering their work. Due to the diverse nature of the LHCs and code methodologies, the benchmarks differ between the code/LHC pairs to precisely tune them towards the LHC needs.

We first give an overview of the code capabilities, strengths, and weaknesses for the five European HPC partner codes Flexi, Alya, waLBerla, Neko, and NekRS/Nek5000. Then, we introduce the benchmark cases for the five codes and the corresponding LHCs. Flexi introduces three distinct benchmark cases to cover the shock-boundary layer interaction and buffet on wings at the edge of the flight envelope required for LHC1. Those benchmarks are a Taylor Green vortex-, a wall model- and an LHC-related benchmark. Alya covers the high-fidelity aeroelastic simulation of LHC2 utilizing a Taylor Green vortex-, transient cantilever beam, and Turek and Hron FSI benchmark. waLBerla also provides three benchmarks for representing the functionality of a fully resolved coupled fluid-particle simulation. A channel flow benchmark covers the fluid phase, a percolation benchmark as an extension of the channel flow benchmark covers the coupling, and a settling spheres benchmark tests the particle dynamics. Neko introduces three benchmarks for LHC3 (Topology optimization of static mixers) and LHC6 (Merchant ship hull): a channel flow-, an immersed boundary method- and a filtering benchmark. Finally, NekRS/Nek5000 uses a GABLS benchmark tailored towards LHC5 (Simulation of Atmospheric Boundary Layer flows). After introducing the benchmarks, we define performance metrics that will be used to quantify the performance of the benchmarks. Scaling metrics and energy efficiency will be used among all codes. However, due to the diverse nature of the solvers, each code provides one or more individual performance metrics.

Contents

1 Introduction

The Center of Excellence in Exascale CFD (CEEC) aims to advance state-of-the-art CFD algorithms and models with the clear goal of enabling exascale performance. This will be demonstrated by the example of six lighthouse cases (LHCs), which have high relevance for scientific and industrial applications. The deliverable D1.1 provides more details regarding the LHCs [1].

To maximize the impact of the CEEC and to ensure that the developments help to meet the requirements of specified real-world CFD applications for future exascale systems, the CEEC will employ a co-design process. For this purpose, the CEEC project is organized around six LHCs, representing challenging problems in the field of CFD that require the capabilities of these future HPC systems. The LHCs will drive the development of the codes and workflows such that they will be ready to be used on exascale systems.

The overall goal of the work within this work package (WP1 — 'exascale light-house cases'), is to provide an overview of code capabilities, equations, regimes, strength and weaknesses as well as the definition of one or more common benchmark problems and performance metrics which are used in WPx. These developments are designed to meet the requirements of the six CEEC LHCs. The six LHCs considered cover a broad range of CFD applications:

- Shock-Boundary layer interaction and buffet on wings at the edge of the flight envelope
- High fidelity aeroelastic simulation of the SFB 401 wing in flight conditions
- Topology optimization of static mixers
- Localized erosion of an offshore wind-turbine foundation
- Simulation of Atmospheric Boundary Layer flows
- Merchant ship hull

To achieve our goal of advancing state-of-the-art CFD algorithms and models for the six LHCs, we face the problem that the final LHCs will only be finalized towards the end of the project. There, we introduce a set of benchmarks covering the critical functionalities of the LHCs, aiming to enable the performance optimization in WP2/3/4 to commence based on these benchmarks. As the benchmarks are carefully tailored towards the LHCs, the optimizations based on the benchmarks will directly benefit the LHC performance.

As the benchmarks' pivotal role is steering the work between WP1 and WP2/3/4, they must cover all the key functionalities within the diverse spectrum of the lighthouse cases. Given the wide-ranging nature of these cases, we must delineate the benchmarks individually for each lighthouse case/code. This approach will enable us to effectively tailor the performance optimization efforts in WP2/3/4 to each lighthouse case's unique characteristics and requirements, ensuring a more streamlined and targeted optimization process. Each code/LHC pair has collaboratively developed one or more benchmarks that are common between the corresponding two partners steering their work.

In this deliverable, we first give an overview of the code capabilities, strengths, and weaknesses for the five European HPC partner codes Flexi, Alya, waLBerla, Neko, and NekRS/Nek5000 in Section 2. Then, we introduce the benchmark cases for the five codes and the corresponding LHCs in Section 3. Finally, Section 4 defines performance metrics that will be used to quantify the performance of the benchmarks.

2 Overview of code capabilities, strength and weaknesses

In the following, we give an overview of the capabilities, strength and weaknesses of each code.

2.1 Flexi

The open-source CFD solver FLEXI focuses on the solution of the compressible Navier–Stokes–Fourier equations (NSE), i.e., the macroscopic level. The NSE are discretized in space via a high-order discontinuous Galerkin spectral element (DGSE) method. This renders the scheme highly efficient, especially on parallel systems. Non-linear stability due to underresolved turbulent structures is ensured via a split-form DGSE method. Special care has to be taken for discontinuities in the solution, such as shock waves, since high-order schemes are subject to oscillations at strong discontinuities. To alleviate these oscillations, discontinuities are adequately captured and a localized artificial viscosity approach based on a finite volume subcell scheme (similar to h-refinement) is applied to ensure a high-order accurate scheme in smooth regions. The NSE are integrated in time using an explicit, high-order, low-storage Runge–Kutta scheme. The primary focus of FLEXI is on large eddy simulations (LES) from wall-resolved to wall-modeled (and direct numerical simulations), where analytical and ODE-based wall models are available. Moreover, local mesh refinement via non-conforming elements is possible.

Turning to the strength and weaknesses of FLEXI, a strong advantage of FLEXI is that compressibility effects such as shock waves or acoustics can be easily considered since FLEXI relies on the compressible NSE. Combined with an explicit high-order time integration, this allows a highly accurate representation of physical effects even on the smallest time scales. However, explicit schemes impose a time step restriction, which can lead to a tremendous computational effort, depending on the size of the smallest physical time scale and the simulation runtime.

2.2 Alya

`ALYA/SOD2D` constitutes a versatile simulation framework for Fluid-Structure Interaction (FSI) problems specially crafted for turbulent flow regimes and finite deformation structural analysis. This framework aims to capitalize on the robust solid mechanics solver and algebraic coupling strategies of `ALYA`, synergizing them with the exceptionally efficient compressible scale-resolving fluid solver of `SOD2D`.

`ALYA` is a continuous Galerkin Finite Element multiphysics simulation code designed to provide flexibility in coupling different physics thanks to its modular structure. In this sense, `ALYA`'s kernel provides the parallel infrastructure while each module deals with a specific physics. In the context of CEEC, the solid mechanics kernel is employed, which is built following a Total Lagrangian formulation with a Finite Element spatial discretization and the family of Newmark schemes for the temporal integration, leading to implicit and explicit solution schemes. When dealing with implicit methods, built-in solvers and external libraries, such as PETSc and MUMPS, are employed. In turn, the coupling is performed from an algebraic standpoint and is implemented as a multi-code approach by exploiting the domain decomposition via MPI. This renders the coupling independent of discretization while providing flexibility to implement several staggered solution algorithms, such as Jacobi or Gauss-Seidel, and the usage of high-performance algorithms, such as Dynamic Load Balance (DLB). At the same time, this strategy enables

the utilization of external solvers like `SOD2D` to deal with some of the physics involved. Under this setup, the current weakness is essentially the efficiency of the iterative solvers, algorithmically (in terms of iterations) and thus also computationally (in terms of time).

`SOD2D` is a hardware-accelerated continuous Galerkin Finite Elements simulation code conceived for simulating turbulent compressible and incompressible flows over intricate geometries. Its core relies on the Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS) approaches, but only the former is of interest to CEEC. In this regard, `SOD2D`'s LES mathematical model combines Spectral Finite Elements with operator splittings on the convective term and incorporates the Entropy Viscosity stabilization model, tailored explicitly to spectral elements. This novel approach addresses the challenge of minimizing artificial (numerical) diffusion with exceptional efficiency while ensuring proper stabilization of the solution. The main limitation of `SOD2D` is the need to use unstructured hexahedron elements mesh, which is not trivial in industrial geometries. To reduce this weakness, `SOD2D` is able to use unstructured meshes generated from the splitting of fully tetrahedron elements and gluing meshes through hanging nodes.

2.3 waLBerla

The multiphysics framework waLBerla employs the lattice Boltzmann method (LBM) for fluid dynamics, an alternative to conventional Navier-Stokes solvers. The LBM is a mesoscopic approach, i.e., neither individual fluid particles, nor the macroscopic quantities are simulated directly, but particle distribution functions are evolved over time in a lattice grid based on the Boltzmann equation. It can be shown by the Chapman–Enskog analysis that the Boltzmann equation recovers the Navier–Stokes equations under certain conditions. LBM is highly parallelizable, allowing for efficient use of massively parallel high-performance computing resources. It is well-suited for simulating complex flow phenomena, including multiphase flows (including bubbles), complex geometries/porous media, heat transfer, and particle transport. On the other hand, the LBM implementation in waLBerla is currently still weakly compressible.

2.4 Neko

Neko is a portable framework for high-order spectral element-based simulations on hexahedral meshes, mainly focusing on incompressible flow simulations. The framework is written in modern Fortran and adopts an object-oriented approach, allowing for multi-tier abstractions of the solver stack and facilitating various hardware backends, ranging from general-purpose processors, accelerators to vector processors, and as well as limited FPGA support. Neko focuses on single core/single accelerator efficiency via fast tensor product operator evaluations.

A matrix-free formulation is considered to achieve good performance in spectral element methods, where one always works with the unassembled matrix on a per-element basis. Gather–scatter operations are used to ensure continuity of functions on the element level, operating on both intra-node and inter-node element data. Currently, Neko uses MPI for inter-node parallelism and parallel I/O for production runs, but one-sided options such as Coarray Fortran-based gather-scatter kernels are under development.

Neko's solver is based on conformal function spaces for both the pressure and momentum equation (citations) which is referred as $P_N - P_N$ method. To advance the incompressible Navier-Stokes equations in time, an implicit explicit scheme relying on backward differ-

entiation and k-th order extrapolation is used. For the pressure, a Poisson equation is solved in each time step, followed by a Helmholtz equation for each of the velocity components. Iterative Krylov subspace methods are used for all systems. For the velocities, a preconditioned Conjugate Gradient (CG) method is applied in combination with a block Jacobi preconditioner. The Poisson equation is solved via a preconditioned Generalised Minimal Residual Method (GMRES).

Currently, the main weakness of Neko is the need to use conformal unstructured hexahedral meshes, which makes mesh generation a tedious and challenging task for complex geometries. Support for non-conformal meshes is planned for 2024, which will mitigate some of the mesh generation challenges in Neko.

2.5 NekRS/Nek5000

Nek5000/NekRS employ high-order spectral elements in which the solution, data, and test functions are represented as locally structured N^{th}-order tensor product polynomials on a set of E globally unstructured curvilinear hexahedral brick elements. The approach yields two principal benefits. First, for smooth functions such as solutions to the incompressible Navier–Stokes equations, high-order polynomial expansions yield exponential convergence with approximation order, implying a significant reduction in the number of unknowns ($\mathrm{n} = EN^3$) required to reach engineering tolerances. Second, the locally structured forms permit local lexicographical ordering with minimal indirect addressing and, crucially, the use of tensor-product sum factorization to yield low $\mathrm{O}(n)$ storage costs and $\mathrm{O}(nN)$ work complexities. The leading order $\mathrm{O}(nN)$ work terms can be cast as small dense matrix-matrix products (tensor contractions) with favorable $\mathrm{O}(N)$ work-to-storage ratios (computational intensity).

Time integration in Nek5000/NekRS is based on a semi-implicit splitting scheme using k^{th}-order backward differences (BDFk) to approximate the time derivative coupled with implicit treatment of the viscous and pressure terms and k^{th}-order extrapolation (EXTk) for the remaining advection and forcing terms. This approach leads to independent elliptic subproblems comprising a Poisson equation for the pressure, a coupled system of Helmholtz equations for the three velocity components, and an additional Helmholtz equation for the potential temperature.

The GPU-oriented NekRS, which is written in C++/OCCA, is the refactored version of Nek5000 which is written in F77/C. NekRS provides access to the standard Nek5000 interface and features (e.g., deformed geometry through an arbitrary Lagrangian–Eulerian formulation and overlapping domains), allowing users to leverage existing application-specific source code and data files on GPU-based platforms.

Nek5000, utilizing the high-order spectral element method, excels in delivering high accuracy and geometric flexibility, particularly for intricate fluid flow details and irregular geometries. Its optimal convergence rates enhance solution accuracy, even with relatively coarse grids. However, in simulations involving highly complex and dynamic geometries, such as wind turbines, additional considerations are vital for mesh generation to ensure conformity across elements. Recognized for exceptional performance, scalability, and portability in parallel computing, Nek5000 excels in large-scale simulations. Its versatility and open-source nature allow users to customize the source code. With NekRS, the software maintains strengths in performance, scalability, and accuracy, leveraging GPU computational power for accelerated simulations. The shift to GPU architecture introduces complexities, requiring user adaptation. As NekRS evolves rapidly, users must

stay current with the latest versions to benefit from ongoing improvements and address potential issues.

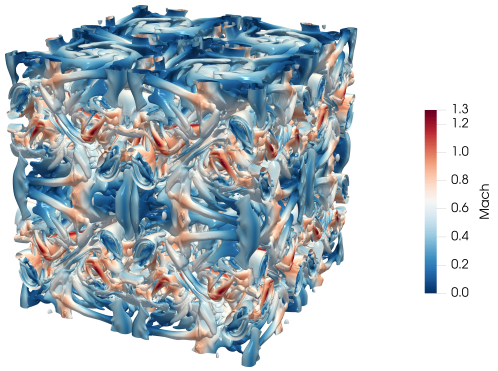

Figure 1: Instantaneous field solution of the compressible TGV visualized by iso-surfaces of the Q-criterion colored by the Mach number.

3 Benchmark cases

In the following, we establish one or more benchmarks per code. They cover the relevant functionality of the final LHC.

3.1 Flexi

The primary focus of LHC1 is on shock-boundary layer interaction and buffet on wings at the edge of the flight envelope. This renders a wall-resolved LES unfeasible such that a wall-modeled LES is utilized. In addition, to ensure non-linear stability due to underresolved turbulent structures and to alleviate oscillations due to strong shock waves, a split form DGSE method and a suitable shock capturing scheme based on a finite volume subcell approach are employed, respectively. To adequately assess and track the performance of these individual parts, it is necessary to define several distinct benchmark cases. This allows to identify and tackle potential bottlenecks such that further performance and accuracy improvements are possible and predefined milestones are reached. For this, three benchmark cases are defined, designed to evaluate first the core performance of FLEXI for the pure DGSE operator, second, for the machine learning based wall-model, and finally, a benchmark case that covers all functionalities relevant for LHC1.

Taylor Green Vortex Benchmark

With the incompressible Taylor Green vortex (TGV) ($M = 0.1$) benchmark test case, the performance and accuracy of the pure DGSE operator is assessed. To evaluate the shock-capturing capabilities of FLEXI, the compressible TGV with $M = 1.25$ will be investigated. The TGV is a common benchmark and composed of sinusoidal turbulent structures initialized in a periodic, Cartesian box which decays over time; the 3D compressible ($M = 1.2$) TGV is illustrated in Fig. 1. This renders the TVG suitable to measure the parallel efficiency and scale-resolving abilities of the baseline solver. This already includes LHC-related features such as the lifting procedure to compute flow gradients, anti-aliasing methods such as the split form DGSEM, and several post-processing routines necessary for the LHC1, such as time-averaging.

Wall-Model Benchmark

With this benchmark test case, the performance and accuracy of the machine-learning enhanced wall model have to be evaluated. On the one hand, this benchmark case should cover complex situations such as transition, shock-boundary layer interactions, and boundary layers with adverse pressure gradients to which the wall model will be applied. On the other hand, the test case should be straightforward to evaluate without any stability issues, e.g., due to shock waves, and easily accessible for performance and scaling analyses. To ensure this, the wall models are evaluated using canonical test cases such as a turbulent channel flow and a periodic hill test case.

LHC Related Benchmark

This benchmark test case will represent a generalization of LHC1 and will be chosen to assess the performance and accuracy of the solver FLEXI with respect to the features used in the LHC1. Hence, this benchmark case completely covers all relevant properties of LHC1. This includes wall-modeled boundary conditions, including the corresponding evaluation of the interface within the flow domain, a representative fraction of shock captured elements for the shock buffet case, lifting procedure to compute flow gradients, anti-aliasing methods such as the split form DGSE method, and further code analyzing steps. In addition, it should be a test case that can be easily used for scaling tests and various performance considerations within CEEC. Considering the fact that FLEXI is based on hexahedral elements, the underlying mesh will be a Cartesian box on which all necessary features are evaluated.

With these benchmark test cases, we can guarantee that potential bottlenecks in the code are tackled to enable detailed performance improvements, especially for LHC1, which are independent of the architecture of the operating system.

3.2 Alya

The LHC2 focuses on the high-fidelity aeroelastic simulation of the SFB 401 wing in a transonic regime ($\mathrm{Ma} = 0.8$), also known as the HIRENASD wind tunnel model. Thus, the LHC2 relies on a Fluid-Structure Interaction (FSI) simulation modeling the interaction between the turbulent flow field and the flexural behavior of the wing. Consequently, the multiphysics simulation framework must encompass three features: the fluid solver, the structural solver, and the two-way coupling driver.

In the modular context of `ALYA/SOD2D`, each feature of FSI features can easily be isolated. Accordingly, three benchmark tests are proposed to assess a specific feature necessary for the LHC2. This division not only enables the evaluation of the performance independently but also aids in pinpointing potential bottlenecks. The three benchmark tests include i) the Taylor-Green Vortex (TGV) problem for the fluid solver, ii) a cantilever beam subjected to transient load for the structural solver, and iii) the Turek and Hron configuration for the two-way FSI coupling strategy.

Taylor-Green Vortex at $M = 1.25$

The first test focuses on the accuracy and performance of the fluid solver that lies to `SOD2D` solver. The well-known Taylor-Green vortex configuration at a high Mach number is employed for this purpose.

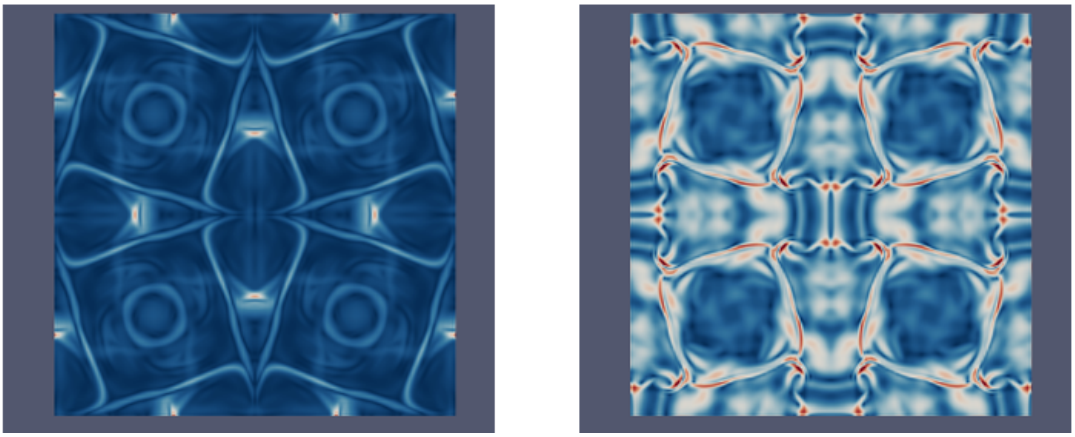

Figure 2: Slices of $\|\nabla p\|$ at $x = \pi$ illustrating the development and evolution of shocks and shocklets: left) t = 15 s and right) t = 20 s.

As previously stated, the LHC2 is designed to model fluid behavior within the transonic regime. Consequently, the impact of compressibility and the formation of shock waves on wing surfaces may become pronounced. As a result, the TGV at $M = 1.25$ serves as an excellent benchmark to evaluate the precision and efficacy of the fluid solver. The objective is to assess the low-dissipation shock-capturing algorithms of SOD2D for turbulent compressible flow fields (i.e., in scenarios where turbulence interacts with flow discontinuities, as shown in Fig. 2). The solver should achieve this without introducing undue dispersion, embodying characteristics akin to a high-order Total Variation Diminishing algorithm but with minimal numerical dissipation.

Regarding efficiency, the simplicity of the TGV mesh allows us to test performance and how it is affected by the different interpolation orders without the burden of generating complex geometrical meshes, allowing us to really push the number of degrees of freedom of our analysis.

Transient Cantilever Beam test

The second benchmark test evaluates the accuracy and performance of the structural solver in `ALYA` through the computation of the transient response of a cantilever beam subjected to external forces.

The significance of this specific benchmark case lies on two fronts. Firstly, considering the targeted fluid-structure problem in LHC2 is the HIRENASD wing model, the cantilever beam geometry and problem statement offer a simplified representation of the wing structure. Secondly, given the availability of analytical data for this geometry, it enables the verification of the correct implementation of the finite element method using quadratic elements. These elements have previously shown efficacy in mitigating locking effects and enhancing simulation accuracy.

Beyond the physics, this benchmark case serves as a means to monitor the performance enhancements of the structural mechanics module. The simplicity of this geometry streamlines the mesh refinement procedure, allowing for a thorough examination of code scalability. Consequently, the finite element assembly and the linear solvers' performance are explicitly studied in this context.

Turek and Hron FSI test

The last benchmark test is the Turek and Hron test, and it targets the assessment of the accuracy and performance of the coupling strategy for fluid-structure interaction problems conducted by the kernel of `ALYA`.

Although the regime of the fluid is not the same as in LHC2, the Turek and Hron configuration facilitates the benchmarking of the core of the aforementioned coupling. The test consists of an incompressible laminar flow around a fixed cylinder with a cantilever beam of an elastic Saint Venant-Kirchhoff material attached as shown in Fig. 3. Therefore, how the fluid and the structure interact is close to the HIRENASD wing one, providing insight into the coupling methods used and the interfaces between Alya and SOD2D. It is worth noting that the Aiken acceleration scheme is employed herein to attain a precise solution due to the ratio of fluid and solid densities being one, provoking the well-known added mass effect. Hence, this test also challenges the convergence of the coupling strategy.

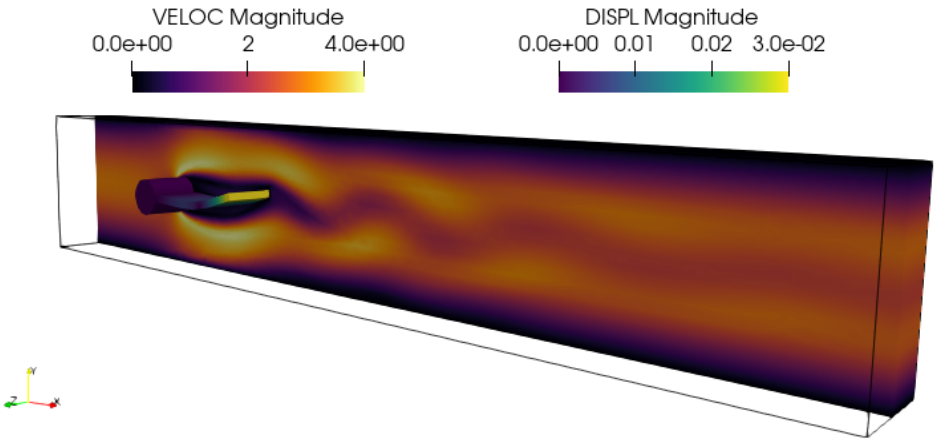

Figure 3: Displacement and velocity fields of the FSI test case.

3.3 waLBerla

LHC4 (Localized erosion in offshore wind-turbine foundations) utilizing the waLBerla framework is a fluid-solid coupled micromechanical simulation focusing on the behavior of seabed foundations. This simulation combines a range of methodologies, namely the lattice Boltzmann method for the fluid simulation, the discrete element method for the particle simulation, and a two-way coupling of these two components. Given the diverse nature of this LHC, it is essential to define several performance benchmarks to separate the evaluation of different components within the simulation. This division allows us a better understanding of the individual performance characteristics and potential bottlenecks. As a result, we have defined three distinct benchmarks, each tailored to assess a specific aspect of the simulation.

Channel Flow Benchmark

This benchmark is dedicated to evaluating the performance of the lattice Boltzmann method. Channel flow is an established benchmark, enabling us to gauge the base efficiency of the fluid simulation without any particle interactions. Fig. 4 visualizes a 2D slice of the 3D channel flow benchmark. It is driven by an inflow (velocity) boundary on

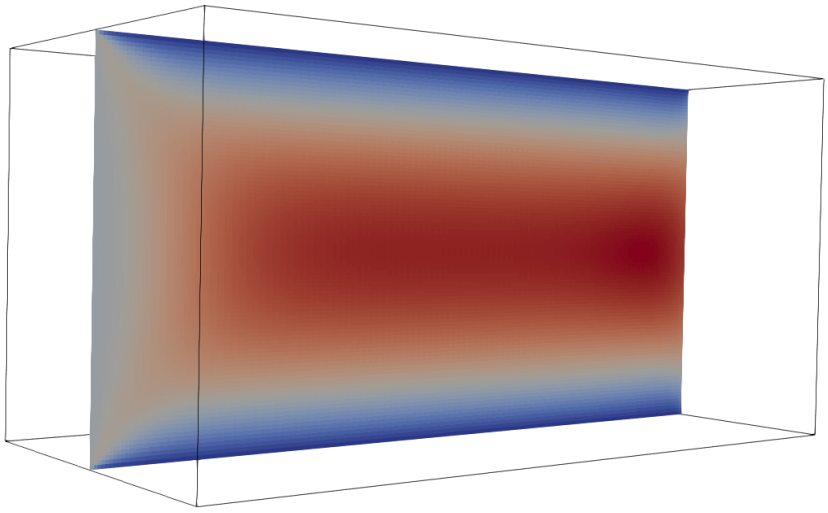

Figure 4: 2D slice of the channel flow benchmark. Colors indicate the velocity. Flow is governed by a velocity boundary on the left and density on the right.

the left and outflow (density) on the right. The Hagen–Poiseuille equation describes the flow.

For such a simple channel flow using the lattice Boltzmann method, setting up a performance model is straightforward, which allows us to evaluate the performance and relate it to the maximum achievable performance.

Percolation Flow Benchmark

The Percolation flow benchmark replicates the channel flow but introduces a two-way coupling with the fixed particles of a porous granular medium. Before the fluid dynamics, the particles are mapped into the fluid domain. Afterward, the hydrodynamic forces and torques acting on the particles are computed. The particles remain fixed in this benchmark, meaning no particle dynamics are involved. This allows us a performance comparison with the channel flow benchmark to assess the penalty on the performance and scalability introduced by the coupling compared to a pure lattice Boltzmann simulation. Fig. 5 visualizes a 2D slice of the 3D percolation flow benchmark. The particles are ordered in a regular grid with a fixed offset in the x-direction (flow direction) and without inter-particle contacts. Every second layer in the flow direction has an offset in y- and z-direction, resulting in a chessboard pattern. Darcy's law describes the flow.

Settling Spheres Benchmark

The third benchmark does not consider fluid dynamics at all but consists of spherical particles settling under the influence of gravity. This benchmark focuses on the performance of the particle dynamics simulation. Fig. 6 illustrates the settling after a certain number of time steps. Initially, the particles are ordered in a regular grid, not touching each other. At the end of the simulation, all particles reach a stable position within the settled (quasi-static) arrangement of the granular sample.

By introducing three benchmarks covering the three main components of our numerical model, we gain a comprehensive understanding of the performance of LHC4. Each bench-

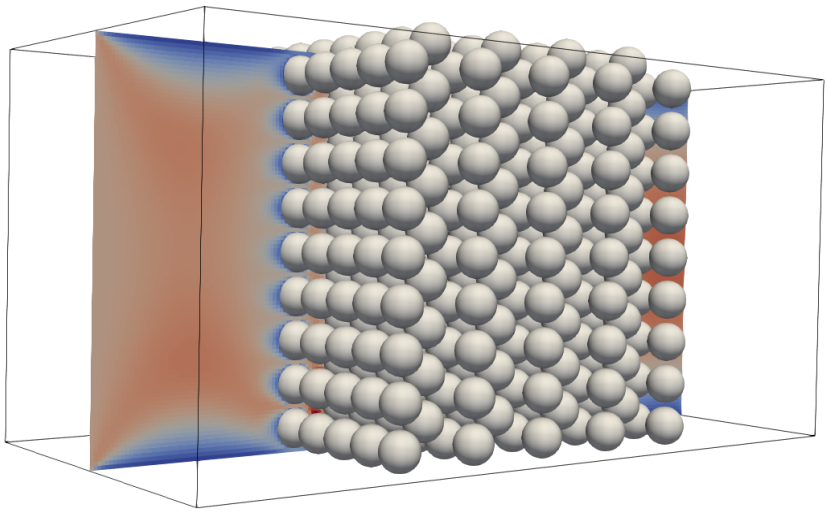

Figure 5: 2D slice of the percolation flow benchmark. Flow is governed by a velocity boundary on the left and density on the right.

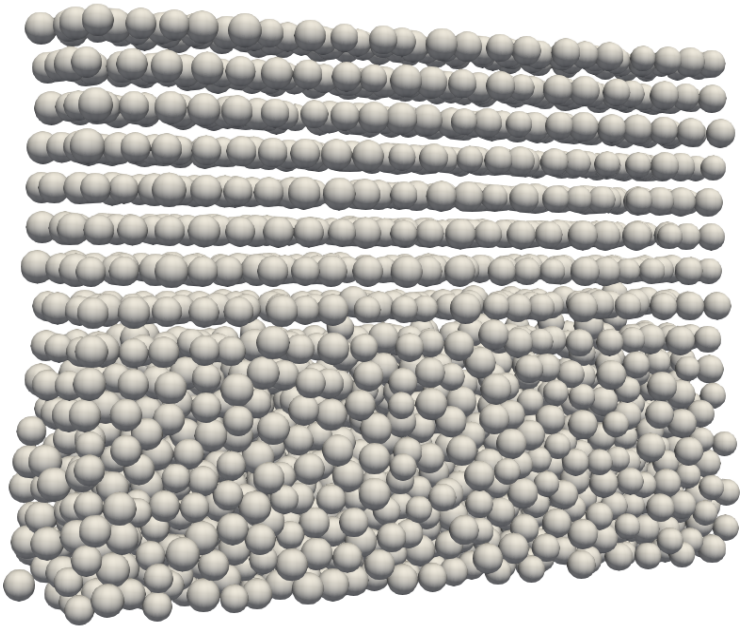

Figure 6: Settling spheres benchmark.

mark addresses a specific facet of the simulation, ensuring that we can make informed decisions about system optimization and resource allocation for complex fluid-solid coupling scenarios.

3.4 Neko

Neko will be used for LHC3 (Topology optimization of static mixers) and LHC6 (Merchant ship hull) to predict the flow in a static mixer configuration and the flow around a Japanese Bulk Carrier. For the LHCs, DNS, LES, and wall-modeled LES simulations are envisaged. The proposed benchmark cases are intended to pave the way for the LHC simulations by addressing particular features and functionalities to be tested.

Channel Flow at $Re_\tau = 300$ Benchmark

For LHC3, the turbulent internal flow inside a static mixer has to be considered, whereas the geometry is of a relatively complex type and subject to optimization. In the first step, an internal flow configuration, here a classical channel flow at $Re_\tau = 300$, is selected, to benchmark the performance of the code Neko against reference DNS data like averaged velocities, second order turbulence statistics and averaged wall shear stress τ_w. A pressure gradient drives the flow, and periodic boundary conditions are applied in the streamwise and spanwise directions.

Since the mixing of species or fluid phases is of interest in LHC3, this case will also be used to evaluate the correct implementation of the Eularian-Eularian approach to handle the transport of a passive scalar, for example.

Benchmark for Immersed Boundary Method

The design representation of the static mixer in LHC3 utilizes the immersed boundary method, which is currently under development in the framework of WP3. Before applying this method within the optimization cycle, a benchmark is required to prove the correctness of the implementation on the one hand and to obtain initial performance metrics on the other hand. At this stage of the project, the test case is intended to be composed out of a channel flow configuration (as above) with an immersed cylinder located on the channel center line.

Filtering Benchmark

For the topology optimisation (LHC3), repeated filter operations are needed to go between the various density formulations of the design, and implement size control. In our previous work, this was achieved using convolutional filters. These are efficient mathematically, however they pose significant performance issues when implemented in a parallel code. Global reduction operations involving neighbours that are within the kernel size are necessary. In particular for unstructured meshes, or meshes with local refinement, such convolutions are costly. Therefore, we are investigating the use of differential (or casual) filters also for density-based topology optimisation. In this way, the existing operators of the code (Neko) can be re-used to implement a filter with required properties. The benchmark for the filter is thus intended to monitor both the computational efficiency and the quality of the results as the implementation is adapted. We will also consider channel flow geometry with immersed obstacles (similar as for the IBM benchmark), and apply different filter implementations with different filter charactersitics.

Channel Flow at $Re_\tau = 5200$: Wall-Model Benchmark

For LHC6, the flow around a ship hull is of particular interest. For this case, the Reynolds number is estimated to be in the order of $\mathcal{O}(Re_L \approx 10^9)$, which indicates wall-modeled LES (WMLES) to be the route of choice to predict the flow field. Within WP4, current advances in machine learning (ML) are combined with wall-models. To benchmark this approach, a channel flow at $Re_\tau = 5200$ is considered. For this case, wall-resolved and wall-modeled LES will be performed to compare results arising from WMLES in combination with ML-based sub-models, like velocity profiles second order turbulence statistic, with results from wall-resolved simulations or simulations based on classical algebraic wall-models. Besides flow physics, the general performance of the ML-based sub-models is an important parameter to be evaluated to verify a possible gain (time to solution) of this approach.

Complex Flow Benchmark: Hill or Step

Since the flow around a ship hull is more complex than a simple channel flow scenario, for example, due to the curved surfaces and adverse pressure gradients or flow separation, a further benchmark is required to evaluate the ML-based wall models. The benchmark has to cover the more complex flow behavior while being straightforward in terms of implementation and computational cost. For this purpose, a canonical test case, the turbulent flow over a periodic hill (or forward-facing step), will be investigated to evaluate the performance of the wall models.

3.5 NekRS/Nek5000

In conjunction with LHC5 (Simulation of Atmospheric Boundary Layer flows), we will perform high-resolution LES to investigate the stable and convective Atmospheric Boundary Layers (ABL) to examine the quality of LES solutions and, in particular, their dependence on the mesh, subgrid-scale (SGS) parameters, numerical discretization, and surface boundary conditions.

To achieve this, we plan to leverage the high-order Nek5000 and NekRS codes, complemented by wall models grounded in the Monin–Obukhov (M–O) similarity theory. These wall models are specifically tailored for rough surfaces and are well-suited for variational formulation approaches like the Spectral Element Method (SEM). Collaborative efforts with scientists from Argonne National Laboratory (ANL) and the National Renewable Energy Laboratory (NREL) will be integral for cross-verification and validation of LES results and associated wall models on both CPU and GPU platforms.

GABLS benchmark

In order to facilitate model and code inter-comparison, we have identified the Global Energy and Water Cycle Experiment (GEWEX) Atmospheric Boundary Layer Study (GABLS) as a well-documented benchmark problem for stably stratified ABL. The stably stratified ABL, exemplified by scenarios such as the nocturnal ABL over land, serves as an excellent benchmark case as the characteristic turbulent scales are notably smaller. Consequently, the sensitivity of LES to SGS models is accentuated. In this paradigm, the ground temperature remains cooler than the air temperature, and this cooling trend persists throughout the simulation, with a domain size of 400 meters in all directions.

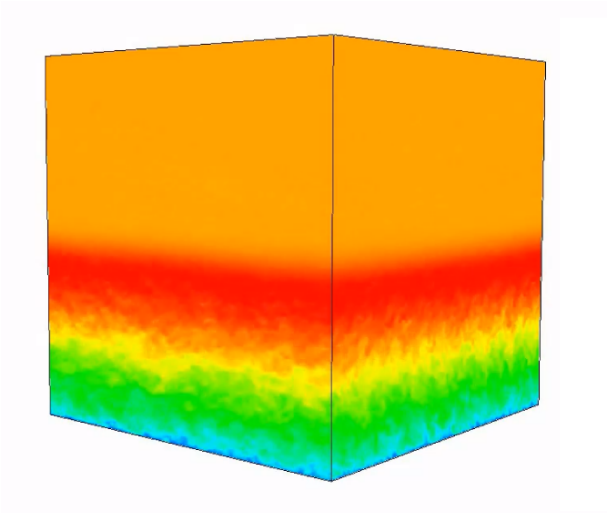

Figure 7: GABLS domain and x-velocity at $t = 7h$.

The temporal development of mean velocity and temperature profiles between Nek5000 and NekRS are comprehensively compared, along with the results of several other codes. Particularly, there is an interest in the height of the peak velocity within the low-level jet during the quasi-steady evolution of the GABLS problem (after approximately 7 hours), as well as its maximum value. Acknowledging the persistent requirement for improvements in SGS turbulence models, augmenting simulation resolution stands as a plausible strategy to alleviate the reliance on these models and improve accuracy.

An outcome of these comparative simulations is a dedicated initiative to rigorously validate and cross-verify the SGS models. Through this extensive benchmark study, our goal is to deepen our comprehension of ABL simulations, ultimately contributing to the improvement and refinement of LES models and codes within the field of atmospheric science.

4 Performance metrics

In this section, we first state the performance metrics we share among all partners and present the code-specific performance metrics.

4.1 Common performance metrics

We have decided to employ the "strong scaling," "weak scaling," and "I/O scaling" performance metrics as our standardized evaluation criteria for all benchmarks as they are independent of the applied methodology or application scenario. These metrics will provide a comprehensive view of our system's performance under different scalability scenarios.
Strong scaling measures how the performance of a parallel application changes if one increases the number of compute resources while keeping the problem size constant. Strong scaling gives insights if the goal is to solve a fixed-size problem as quickly as possible.
Weak scaling evaluates the ability of a parallel system to handle larger problem sizes efficiently by increasing both the number of compute units and the problem size proportionally. In weak scaling, the goal is to maintain a constant workload per compute resource. For example, ideal weak scaling would mean that doubling the problem size and the number of compute resources keeps the execution time constant.
I/O scaling, while being less commonly reported in literature than strong and weak scaling, is crucial for applications that heavily rely on data input and output operations, which is typically the case for simulations. I/O scaling measures the ability of a parallel system to manage increasing demands on data storage and transfer as more processors are used. In many cases, the performance of I/O operations can become a bottleneck, affecting the overall scalability of the application. Therefore, it should be an essential part of performance benchmarking and improvement.
Finally, energy efficiency is a key performance metric for large-scale CFD simulations, whereby higher efficiency values generally indicate a responsible use of (financial) resources and minimize the environmental footprint of the simulations.

4.2 Flexi

Apart from the common performance metrics, FLEXI uses the performance index (PID) as a primary performance metric. The PID describes the time required to advance one DOF for a single Runge–Kutta stage and is defined as follows

$$\mathrm{PID} = \frac{\text{wall-clock-time} \cdot \#\text{cores/devices}}{\#\text{DOF} \cdot \#\text{time steps} \cdot \#\text{RK-stages}}. \tag{1}$$

Since the PID indicates the execution time, a smaller PID implies better performance. Therefore, the PID allows us to compare the performance for cases with different problem sizes and resources. It is particularly suitable for explicit codes since the computational effort for each timestep is the same in contrast to iterative solvers. The PID can be used to estimate the computation times required for scaling simulation setups to larger simulation cases since it is almost constant for a given hardware and setup. In the case of FLEXI, the PID is the underlying metric for the common normalized benchmark metrics, such as strong and weak scaling. For better comparability between widely differing hardware architectures (e.g., a system with and without accelerators), it is worth considering the required power in the PID to take normalized energy usage into account.

4.3 Alya

Strong scalability

The performance assessment of `ALYA/SOD2D` relies on the Performance Optimisation and Productivity (PoP) Centre of Excellence in HPC, which consists of a multiplicative model for global efficiency. Theoretically, global efficiency can be obtained by performing a strong scalability test using a baseline with one single core. As this is impossible in practice due to time and memory constraints, using a baseline with more than one core will eventually deviate from the interpretation of the strong scalability. Under this context, two options exist. On the one hand, the strong scalability is computed using a baseline obtained with the minimum number of cores, which should be explicitly mentioned for correct interpretation. On the other hand, we can rely on absolute measures of the parallel efficiency PE, which can be expressed as $PE = LB \times CE$, where LB is the load balance and CE is the communication efficiency. These are absolute measures with values between 0 and 1 that can be obtained using libraries such as the Tracking Application Life Performance (TALP). Accordingly, the benchmarks for the LHC will provide the following:

- Strong scalability with baseline obtained with a minimum number of cores.
- Real measures of LB and PE using TALP library.

Timings

`ALYA` is tested continuously through a performance suite to monitor the performance variation as the code evolves with each merge to the main branch via the automatic execution of some benchmark tests, such as the ones proposed for CEEC. This monitoring saves the history of the optimizations implemented in `ALYA`. In turn, it serves as an alarm when a branch has worsened the general performance of the code or when the supercomputer tested (herein MN4) is failing (IO, gpfs, clock frequency, etc.). As `SOD2D` will be coupled as an external library, it will be assessed similarly for the benchmark cases herein studied. Some of the parameters to be considered will be:

- Solver efficiency
- Time to solve a DoF per GPU (SOD2D)
- Time to solve a DoF per CPU (ALYA)

4.4 waLBerla

For the fluid simulation utilizing the lattice Boltzmann method, which is employed in the channel flow and percolation flow benchmarks, a standard performance metric used for evaluation is MLUP/s (Million Lattice Updates Per Second). It is computed as

$$\text{MLUP/s} = \frac{\#\text{cells} \cdot \#\text{timesteps}}{\text{runtime}} \tag{2}$$

and is a fundamental indicator of these simulations' computational efficiency and capability. MLUP/s can be directly translated into DOF/s (Degrees Of Freedom per Second) by multiplying it by the number of degrees of freedom per cell, typically 19 or 27. For particle dynamics simulation, such as the one in the settling spheres benchmark, the relevant performance metric is PUp/s, which stands for Particles Updated Per Second. It is

computed as:

$$\mathrm{PUp/s} = \frac{\#\mathrm{particles} \cdot \#\mathrm{timesteps}}{\mathrm{runtime}} \tag{3}$$

and is a measure of how effectively the simulation can advance the positions and attributes of individual particles within the system over a given time frame.

4.5 Neko

The performance assessment for Neko follows the common performance metrics, with strong scalability as the primary performance figure of merit.

For the performance evaluation, runtime statistics in the form of the average time per timestep and per individual simulation component per step are collected. Furthermore, the performance assessments are conducted over an averaging window of a couple of hundred steps after the initial transient. The position and length of the averaging window are defined in the case's input file.

4.6 NekRS/Nek5000

To analyze and improve the efficiency of the two codes, scaling studies are conducted on our benchmark case. As a metric of the problem's size, we define $n = EN^3$, where n is the number of grid points, E the number of spectral elements, and N is the N^{th}-order polynomial basis. For the strong scaling study, for each case, the same spatial resolution (number of elements and polynomial order), timestep, and solver tolerances are maintained. To facilitate accurate timings, performance assessments are conducted over a time frame wherein the solutions exhibit a representative turbulent flow. For the weak-scale study, the domain height is fixed while the dimensions are expanded along the x and y directions (leading to an increase of n). To mitigate initial transient behavior, the average (wall) time per step, t_{step}, in seconds is measured over steps 101–200.

An essential metric for evaluating code performance and scaling capabilities is parallel efficiency. In the context of Nek5000 and NekRS, where the number of processing units P is represented by CPUs (cores) and GPUs, respectively, the parallel efficiency (P_{eff}) is defined as follows:

$$\mathrm{P_{eff}} = \frac{\mathrm{t_0} \cdot \mathrm{P_0}}{\mathrm{t_{step}} \cdot \mathrm{P}} \tag{4}$$

where P_0 is the smallest value of P processing units that will hold the given problem, and t_0 is the t_{step} cost corresponding to P_0.

In addition to the aforementioned aspects, each Nek5000 job meticulously monitors fundamental runtime statistics by employing MPI Wtime, complemented by cudaDeviceSynchronize or CUDA events for NekRS. This comprehensive tracking mechanism offers a detailed breakdown of crucial metrics related to code performance, including the computational cost of individual subroutines and I/O operations.

5 Summary

This deliverable presented the capabilities, strengths, and weaknesses of the five European HPC partner codes Flexi, Alya, waLBerla, Neko, and NekRS/Nek5000. Furthermore, we have introduced benchmark cases for the five codes and the corresponding LHCs. Finally, we have introduced the performance metrics we will use to investigate the performance of the given benchmarks. On the one hand, there are performance metrics that are relevant for all codes, namely strong scaling, weak scaling, I/O scaling, and energy efficiency. On the other hand, due to the diverse nature of the codes and methodologies, each code also provides individual performance metrics that are tailored to individual needs.

This deliverable tackles the dilemma that, on the one hand, the final LHCs are only fully developed towards the end of the project, but, on the other hand, the work on the performance and energy efficiency in WP2/3/4 should be able to start immediately. We address this issue by introducing different benchmark problems for each lighthouse case/code pair, whose purpose is to enable the performance optimization in WP2/3/4 to commence based on these benchmarks, as the lighthouse cases are not yet finalized.

The benchmarks defined in this deliverable now allow WP2/3/4 to implement their performance and energy efficiency optimizations based on the benchmarks. The specified performance metrics allow the quantification of different performance aspects and comparing them at different development stages and between the codes. However, due to the essential differences in the underlying solvers and equation systems, a direct comparison of performance indicators between the different codes is hardly possible. It should be treated with the utmost caution. Once the benchmarks are built into the CI/CD pipelines, they will allow us to monitor the performance of the most important LHC functionalities over time. The work in this deliverable will help to advance the European HPC partner codes to enable exascale performance for the six LHCs.

Bibliography

[1] CEEC Consortium. Deliverable D1.1 – CEEC exascale lighthouse cases and their needs, August 2023. Available via https://ceec-coe.eu/.