European Journal of Mechanics / B Fluids 102 (2023) 91–102

Contents lists available at ScienceDirect

European Journal of Mechanics / B Fluids

journal homepage: www.elsevier.com/locate/ejmflu

Cost vs Accuracy: DNS of turbulent flow over a sphere using structured immersed-boundary, unstructured finite-volume, and spectral-element methods

Francesco Capuano [a], Nikolaos Beratlis [b], Fengrui Zhang [b], Yulia Peet [b], Kyle Squires [b], Elias Balaras [c,*]

[a] Department of Fluid Mechanics, Universitat Politècnica de Catalunya · BarcelonaTech (UPC), Barcelona 08019, Spain
[b] School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ, 85287, USA
[c] Department of Mechanical and Aerospace Engineering, George Washington University, Washington, DC, 20052, USA

ARTICLE INFO

Article history:
Received 28 February 2023
Received in revised form 25 July 2023
Accepted 26 July 2023
Available online 6 August 2023

Keywords:
Direct Numerical Simulation (DNS)
Immersed-boundary method
Spectral Element Method
Nek5000
OpenFOAM
Comparative study

ABSTRACT

We report a comparative study of three numerical solvers for the direct numerical simulation of the flow over a sphere at Re = 3700. A high-order spectral-element code (Nek5000), a general purpose, unstructured finite-volume solver (OpenFOAM) and an in-house Cartesian solver using the immersed-boundary method (IBM) are employed for the analysis; results are compared against previous numerical and experimental data. Numerical results show that Nek5000 and the IBM code operate within a similar computational performance range, in terms of cost-vs-accuracy analysis, for both global parameters as well as local flow features. On the other hand, OpenFOAM needed a significantly higher number of degrees of freedom (and, overall, a higher cost) to match some of the basic features of the flow, such as the length of the recirculation bubble forming downstream the sphere. For the finest grid resolutions, the three codes are in good agreement for most of the analyzed flow metrics. Overall, our results suggest that high-order methods and second-order, energy-conserving approaches based on the IBM may be both viable options for high-fidelity scale-resolving simulations of turbulent flows with separation.

1. Introduction

The computational fluid dynamics (CFD) community is currently devoting major research efforts towards enabling high-fidelity, eddy-resolving simulations of complex flows within turnaround times that are compatible with industrial design cycles [1,2]. The key to achieving this goal is arguably the adoption of numerical discretization algorithms that are simultaneously accurate, robust, flexible and, at the same time, efficient (i.e., with a high accuracy/cost ratio). The fulfillment of such a set of conflicting requirements continues to thrill the CFD community, especially when it comes to choosing the appropriate method for a specific field of applications.

Traditionally, industrial CFD simulations have been carried out using low-order finite-volume or finite-difference schemes, either on unstructured meshes, or in conjunction with the immersed boundary method (IBM) to cope with complex or moving boundaries. On the other hand, recent years have seen the development of high-order techniques, such as discontinuous Galerkin and spectral-element methods, that potentially combine high accuracy with geometric flexibility and ease of parallelization. Today the whole spectrum of methodologies is within reach of the community by the availability of open-source software (e.g., OpenFOAM [3], Basilisk [4], Nek5000 [5], PyFR [6], etc.). As a result, there is a growing interest in understanding and comparing the accuracy and efficiency of various approaches/solvers. Although there are numerous studies comparing the accuracy of different solvers in terms of their ability to capture certain flow related metrics, the computational cost and therefore the overall efficiency is rarely addressed in a quantitative manner. This is primarily because the latter is difficult to measure as it can be affected by several factors (flow specifics, solution algorithm, parallelization strategy, computer hardware, etc.).

The following section provides a brief literature overview of previous works, where different approaches and/or solvers have been systematically assessed and compared. Subsequently, the test case considered in this work is discussed in detail, alongside with the objectives of the paper.

* Corresponding author.
E-mail address: balaras@gwu.edu (E. Balaras).

https://doi.org/10.1016/j.euromechflu.2023.07.008
0997-7546/

1.1. Overview of previous works comparing different approaches/solvers

A systematic platform to comparatively assess the cost-efficiency of several CFD approaches is provided by The International Workshops on High-Order CFD Methods, first introduced in 2012 and held alternately between USA and EU [7]. Multiple solvers utilizing formulations of varying order of accuracy are compared for a series of flow problems of increasing complexity. The comparison is done on the basis of *error vs. cost*. The latter is typically estimated with the open-source tool *TauBench* [8], which provides a work-unit on each computer to non-dimensionalize the total CPU time. From the vast array of problems considered, the ones with smooth solutions and simpler geometries have demonstrated that higher-order solvers performed better than standard second-order methods; on the other hand, the results for non-smooth solutions/geometries and turbulent flows are currently inconclusive and require further research [9,10]. A later study by Moin & Verzicco [11] also argued that while spectral and high-order methods remain superior for smooth and transitional flows, their benefits are significantly curtailed in complex turbulence simulations, and advocated the use of energy-preserving second-order methods. Recently, Vermeire et al. [12] compared the cost/accuracy performance of a high-order unstructured solver deployed on GPU platforms to a standard finite-volume approach commonly used in industrial applications. In particular, the GPU optimized PyFR solver [6] was compared to STAR-CCM+ [13] running on CPUs for turbulent flow problems of increasing complexity. Overall the PyFR simulations of the Taylor–Green vortex problem were consistently cheaper and more accurate than STAR-CCM+ in terms of predicting the dissipation rates based on enstrophy. For more complex cases, such as the flow over a cylinder or an airfoil both codes produced results that were in agreement with the literature at comparable cost. They noted, however, that for the case of STAR-CCM+ this was only possible with a third-order scheme and implicit time stepping; lower order schemes were found to produce excessive dissipation.

Capuano et al. [14] reported a comparative study between the spectral-element solver Nek5000 [5] and the OpenFOAM finite-volume solver [3] for the case of direct numerical simulations (DNS) of synthetic jets. Overall, the mean flow and macroscopic quantities of engineering interest (e.g., saddle point position and stroke length) were predicted accurately by both codes at high resolution. At lower resolution, OpenFOAM over-predicted the residual turbulent fluctuations and therefore failed to capture the partial relaminarization of the near-field occurring within the suction phase, supposedly due to its non-conservative numerics. In terms of computational performance, Nek5000 was found to be slower than OpenFOAM in terms of CPU time elapsed per unit time step and per unit grid point. On the other hand, when both cost and solution accuracy were taken into account, Nek5000 outperformed OpenFOAM in all numerical experiments. This is in line with the results reported by Kooij et al. [15] in their comparison of four codes for DNS of Rayleigh–Bénard convection. In this case, however, both Nek5000 and OpenFOAM were outperformed by a dedicated, second-order finite-difference code and a specialized fourth-order finite-volume solver. Another comparison between OpenFOAM and Nek5000 in scale-resolving simulations of canonical wall turbulence also reported superior cost-efficiency of the latter [16]. Very recently, Saini et al. [17] compared the spectral/*hp* solver *Nektar++* and OpenFOAM for the complex dilution port flows of gas turbine combustors. They found that, for a similar accuracy, a fifth-order LES computation could be 3–8 times less expensive than OpenFOAM.

The IBM gained enormous popularity over the past twenty years but has not been as often included in cost/accuracy studies. Schäfer et al. [18], for example, compared two implementations of the IBM against a body-conforming simulation provided by Nek5000 (taken as the reference) for the DNS of the flow over a transverse bar in the fully turbulent regime. In particular, a direct-forcing method [19] available in the compact finite-difference code Incompact3D [20] and a continuous forcing approach [21] employed by the pseudo-spectral code SIMSON [22] were assessed. They found that both IBM codes operated within a similar computational performance range and were able to reproduce the main flow features when compared to the reference simulation. It should be acknowledged that the immersed body considered in this study was aligned with the underlying Cartesian mesh, and thus did not constitute a challenging test for this formulation.

1.2. Aim of the study and test problem

This work aims to comparatively assess the cost-vs-accuracy for methods and solvers applicable to scale-resolving simulations of turbulent incompressible flows with separation. This belongs to the family of problems for which current studies on cost-vs-accuracy are inconclusive, and indicate that further research is needed [9,10]. It is also a class of applications where classical RANS models have well-known difficulties in reliably capturing separated flows, thereby advocating the use of scale-resolving techniques for the purpose. The flow around a sphere is an example of a prototypical case representative of this class of problems. Depending on the Reynolds number, Re, ($Re = U_\infty D/\nu$, where U_∞ is the freestream velocity, D is the diameter of the sphere and ν is the kinematic viscosity of the fluid) the state of the boundary layers on the surface of the sphere changes from laminar to turbulent and the flow is characterized as sub-critical and super-critical respectively. The value of the Reynolds number that marks this transition is around $Re \sim 5 \times 10^5$. The wake of the sphere is turbulent in all cases, provided that $Re > 1000$ [23]. Eddy-resolving simulations across the full range of Reynolds numbers are challenging due to: (i) the large disparity of scales that need to be captured; (ii) the presence of both laminar and turbulent flow regions; (iii) the need to solve a global equation for pressure in incompressible flows that adds significantly to the computational cost. The first challenge is partly due to the large range of scales typical of high Reynolds number flows, whose accurate resolution demands vast computational resources that need to be devoted for predicting the evolution of the laminar/turbulent boundary layers on the surface of the sphere, which are usually very thin. The sub-critical regime represents a particularly suitable test case for code comparison, as there are many important features that need to be accurately predicted, including: (i) thin boundary layers on the front part of the sphere; (ii) flow separation near the equator of the sphere, (iii) shear layer detachment and instability, with associated characteristic frequency, and (iv) transition to turbulence in the wake and macro-scale vortex shedding. In this study the Reynolds number was set to $Re = 3700$ to allow for comparisons with previous computational and experimental work, summarized in Table 1. The choice of the Reynolds number also allowed for moderately large grids and affordable computations.

The paper is organized as follows. Section 2 describes the set-up of the test case and the three numerical approaches utilized in this work. Numerical results are reported in Section 3, along with a critical discussion of the comparative performances of the three codes. Concluding remarks are finally given in Section 4.

Table 1
Experimental and numerical studies for the flow over a sphere at $Re = 3700$.

Authors	Year	Approach	#dof	Notes
Rodriguez et al. [24]	2011	DNS	9.48M	Unstr. collocated, KEP
Yun et al. [25]	2006	LES	3.25M	Structured IBM, KEP
Sakamoto & Haniu [26]	1990	Exp.	–	Hot wire + flow visualizations
Kim & Durbin [27]	1988	Exp.	–	Hot wire + pressure probes
Schlichting [28]	1961	Exp.	–	–

2. Set-up of numerical simulations

In the following sections we will examine the performance of three representative numerical techniques: (i) second-order unstructured finite-volume method; (ii) second-order Cartesian method with IBM; and (iii) spectral-element method, when used for the DNS of the flow over a sphere at $Re = 3700$. We selected two well-known, open-source implementations for methods (i) and (iii), i.e. OpenFOAM and Nek5000 respectively. Although there are open-source implementations for IBM, their userbase is not as extensive as the other two solvers and the adoption of one over another will be fairly arbitrary. For this reason we selected our in-house code as a characteristic example of a Cartesian IB solver, which has been extensively validated in a variety of complex turbulent flow problems [29]. Results are discussed thoroughly in terms of global efficiency, as well as with regards to the local quality of the flow fields produced by the various codes for various levels of mesh refinement. Comparisons with available numerical and experimental data are also discussed.

The dynamics of the flow problem under consideration is governed by the Navier–Stokes equations for incompressible flow. A cylindrical domain as shown in Fig. 1a was used in all simulations. The letters r, ϕ, and x denote the radial, azimuthal and axial directions respectively, while θ denotes the polar angle, which varies from 0° at the stagnation point at the front of the sphere to 180° at the opposite end. The center of the sphere is located at $r = 0, x = 0$. The inlet, outlet and freestream boundaries are located at $5D$, $20D$ and $7D$ from the center respectively. This choice was based on previous computational studies of the same problem at a range of Reynolds numbers utilizing the solvers considered in this work [30–32].

The computational grid for all different methods was designed to meet the following criteria: (i) place a minimum number of points in the laminar boundary layer along the upstream portion of the sphere, which is nonetheless sufficient to correctly capture the onset of separation; (ii) resolve the energetic turbulent structures in the near wake, $x/D \leq 5$. To estimate the thickness of the laminar boundary layer, which will then set the scale for the local grid resolution, a reduced set of equations is considered representing a thin, axisymmetric, boundary layer:

$$\frac{\partial (r_0 u_s)}{\partial s} + r_0 \frac{\partial u_n}{\partial n} = 0 \tag{1a}$$

$$u_s \frac{\partial u_s}{\partial s} + u_n \frac{\partial u_s}{\partial n} = U \frac{dU}{dx} + \frac{1}{Re} \frac{\partial^2 u_s}{\partial n^2}, \tag{1b}$$

where s and n are the unit vectors in an orthogonal system tangent and normal to the surface of the sphere (see Fig. 1a), r_0 is the distance from the center to the surface of the sphere, and U is the freestream velocity. The above equations are parabolic and are numerically solved using a finite-difference downstream marching technique (see [33] for details). The freestream velocity distribution, $U(\theta)$, is taken from the measurements by Fage [34] and fits the following polynomial curve:

$$U(\theta)/U_\infty = 1.5\theta - 0.4371\theta^3 + 0.1481\theta^5 - 0.0423\theta^7, \tag{2}$$

where θ denotes the polar angle from the front stagnation point. Fig. 1b shows the evolution of the boundary layer thickness, $\delta(\theta)$, on the front part of the sphere for three different Reynolds numbers, $Re = 370$, $Re = 3700$ and $Re = 37000$. For all Reynolds numbers the boundary layer thickness grows in a similar way and scales by the established $Re^{-0.5}$ law. We should note that the boundary layer thickness increases almost linearly from $\theta = 0°$ to $\theta = 30°$ and then almost exponentially until separation at $\theta = 84°$ (as predicted by this laminar boundary layer model). The $\delta(\theta)/D$ above was used to guide the design of the computational grid near the surface of the sphere; care was taken to ensure that the grid resolution at the wall is comparable for the three solvers. In the next subsections we provide information on the specific setup for each solver.

2.1. Spectral-element method: Nek5000

The open source spectral-element code Nek5000 [5] is employed in this work as a representative example of high-order methods. Nek5000 is a scalable solver that has been widely used for computations of fluid flows, heat transfer and magnetohydrodynamics, among others. The incompressible flow branch of Nek5000 is utilized in the current study. With a spectral-element formulation, the computational domain is divided into N_e non-overlapping elements, which are related to the reference elements in the computational domain $[-1, 1]^{\text{dim}}$ by an appropriate transformation, where dim = 3 for the three-dimensional cases. In the version of Nek5000 used in the current study, velocity and pressure are approximated with a P_N/P_{N-2} formulation as:

$$\begin{aligned} \mathbf{u}(\xi, \eta, \zeta) &= \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{k=0}^{N} \mathbf{u}_{ijk} l_i(\xi) l_j(\eta) l_k(\zeta) \\ p(\xi, \eta, \zeta) &= \sum_{i=0}^{N-2} \sum_{j=0}^{N-2} \sum_{k=0}^{N-2} p_{ijk} l_i(\xi) l_j(\eta) l_k(\zeta), \end{aligned} \tag{3}$$

where $l_i(\xi), l_j(\eta), l_k(\zeta)$ are the Lagrangian interpolating polynomial functions that satisfy the cardinality property $l_i(\phi_j) = \delta_{ij}$ on Gauss–Legendre–Lobatto (GLL) points for velocity and Gauss–Legendre (GL) points for pressure, respectively, and δ_{ij} refers to the Kronecker delta function. In the current cases the polynomial order is chosen as $N = 5$. A semi-implicit time advancement scheme is used, where convection terms are marched explicitly with the second-order extrapolation in time and diffusion terms are integrated implicitly with the second-order backward difference scheme. The code is fully dealiased using an over-integration technique [35], the velocity is solved using the preconditioned conjugate gradient (PCG) method, and the pressure solver uses the iterative generalized mean residual (GMRES) method in the Krylov subspace. Overlapping additive Schwarz method is used as a preconditioner for the pressure problem [36], and XXT method is employed as a coarse grid solver. XXT method is based on a Cholesky factorization of a matrix A into XX^T, where the sparse matrix X retains the sparsity pattern of A in maximum [37]. The tolerance for the coarse grid solver was set to 10^{-6}. Further details of the current spectral-element methodology can be found in Refs. [31,38,39].

Three sets of meshes are constructed for this study, referred to as a coarse mesh, a medium mesh, and a fine mesh, respectively.

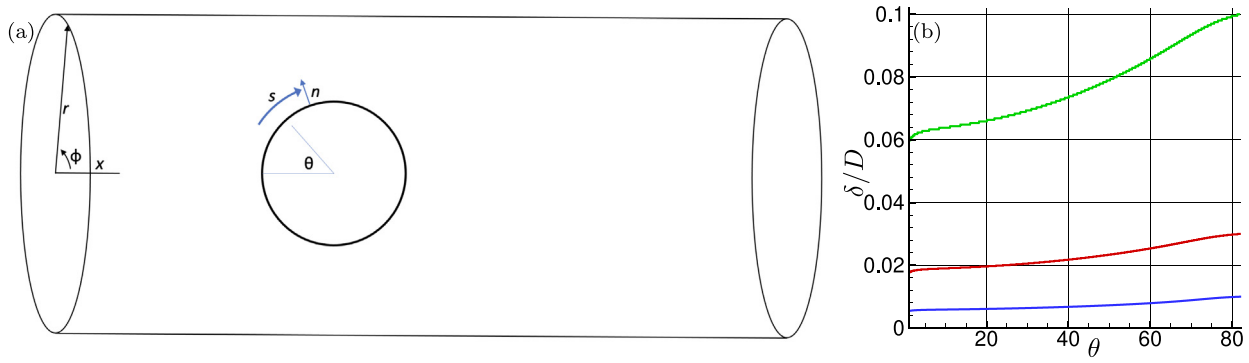

Fig. 1. (a) Sketch showing the computational domain and the local coordinate system. (b) Boundary layer thickness on a surface of a sphere at various Reynolds numbers predicted by Eqs. (1a)–(1b). —— $Re = 370$; —— $Re = 3700$; —— $Re = 37000$.

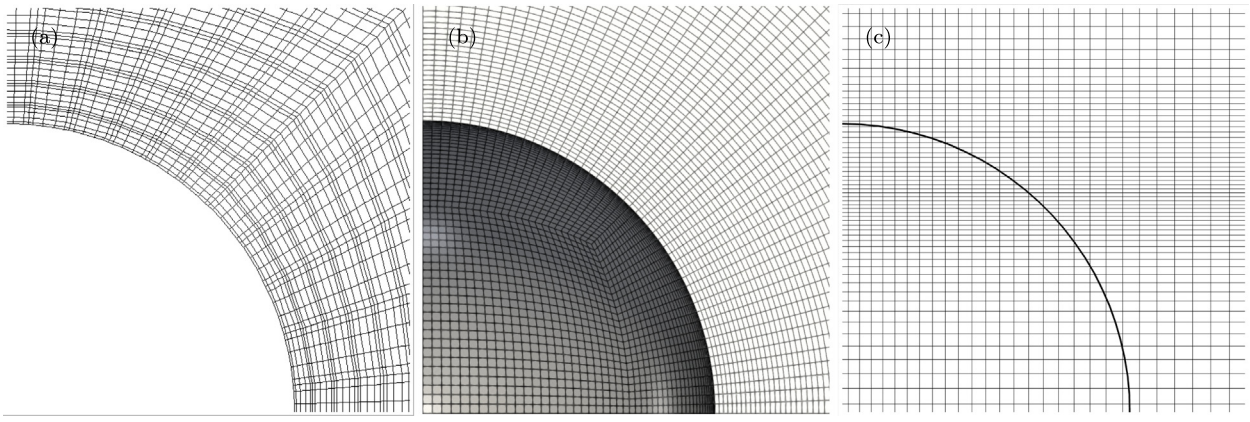

Fig. 2. Comparison of the coarsest mesh around the sphere used in each code, (a) Nek5000 (b) OpenFOAM and (c) IBM.

O-grid mesh is utilized in each case, with a local refinement within the boundary layer around the sphere, with a minimum element height $\Delta r_{min} = 0.015D$ in a fine case. The mesh is entirely hexahedral and is generated in Ansys Icem, upon which it is converted to the format that can be read by Nek5000 using the open-source exo2nek converter [40]. The solid boundary of a sphere is approximated with 5th-order polynomials defined on GLL points on the surface of the sphere to correspond to the accuracy of the interior solution. The total number of degrees of freedom (DOFs) for the coarse, medium and fine cases are approximately 2.8 millions, 6.5 millions and 9.8 millions, respectively. A view of the mesh around the sphere for the coarse grid is shown in Fig. 2a. The no-slip wall boundary condition is employed for the surface of the sphere. A uniform Dirichlet boundary condition is used for the inflow boundary, while all the other boundaries of the cylindrical computational domain are set to the stabilized outflow boundary condition [41]. The time step was set to $2 \times 10^{-3} D/U$ for all the cases. This resulted in the CFL numbers being 0.65, 0.77 and 0.88 for the coarse, medium and fine grids, respectively. While time step could be further increased while still maintaining stable simulations, which would result in shorter wall-clock times, this was not done in the current study to eliminate the sensitivity of the grid refinement to a time step size. The evolution of the forces on the sphere was monitored and when the forces reached a quasi-steady state behavior (at a time of approximately $200D/U$) statistics were sampled over time. Overall, the simulations were integrated in time for $200D/U$ to get rid of transient effects, and the time-averaged statistics were sampled for another $300D/U$.

2.2. Unstructured finite-volume method: OpenFOAM

OpenFOAM is a general purpose open-source platform with a wide range of possible applications. Written in C++, it comes with several native solvers and also offers the possibility to create custom codes. The native incompressible flow solver *pimpleFoam* from Version 7.0 of the software is used in this work [3]. The discretization is based on a finite-volume approach with collocated arrangement, with both convective and diffusive terms approximated by second-order centered schemes. A blended central-upwind scheme for convection is also available in OpenFOAM but all computations in this work were stable with the purely central scheme. Time integration is achieved by a second-order Crank–Nicolson method. The overall solution methodology is based on the PIMPLE algorithm, which is a slightly modified version of the original PISO formulation [42]. The reader is referred to [43,44] for a detailed description of the spatial discretization methods employed by OpenFOAM. Here we highlight that the standard velocity interpolation procedure utilized in *pimpleFoam*, which is based on a cell center distance weighted average, is not discretely kinetic-energy preserving [45]. Also, while the collocated mesh approach offers advantages in terms of implementation and storage of flow variables, the solution of the Poisson equation for pressure can lead to the well-known checkerboard phenomenon. Several remedies have been proposed in the literature to address this issue; in OpenFOAM, a specific cell-face velocity interpolation method is adopted that has been shown to introduce dissipation into the numerical solution [46]. For a thorough discussion of the numerical errors associated with the standard OpenFOAM incompressible solvers, the reader is referred to recent work by Komen et al. [47,48]. In this work, the pressure linear system is

solved by a generalized geometric–algebraic multi-grid (GAMG) method with a tolerance of 10^{-7}.

Three O-grid meshes, each with a successive 50% refinement, were created using the built-in OpenFOAM tool *blockMesh*. A cylindrical outer domain was used in all cases. We chose this type of grid over an unstructured one because it produces high quality meshes and also gives good control of the grid spacing around the sphere and in the wake. The grid is qualitatively similar to the one used in the Nek5000 simulations (see Fig. 2a–b). The grid around the sphere was stretched in the radial direction to resolve the boundary layer while uniform elements were employed in the other two directions around the sphere. For the finest grid the minimum element height at the surface of the sphere is $\Delta r_{min} = 0.0036D$ and $0.0078D$ in the other two directions. The grid was kept uniform in a region in the near wake up to $x = 3.8D$ with an axial resolution $\Delta x = 0.009D$ and a resolution of $0.016D$ in the other two directions. The total number of elements for the coarse, medium and fine cases are approximately 3.7, 13.5 and 30 million nodes respectively.

No-slip wall boundary condition was specified at the sphere, while symmetry was used at the outer cylindrical boundaries and an outlet boundary condition at the outflow. The time step for the finest grid was set to $\Delta t = 2.5 \times 10^{-3} D/U$ ensuring the Courant number was slightly below 1.0. The simulation was integrated in time for $200D/U$ to get rid of transient effects and time-averaged statistics were sampled for another $300D/U$.

2.3. Immersed-boundary method

The Navier–Stokes equations for viscous incompressible flow are solved on a structured grid in cylindrical coordinates. The governing equations are advanced in time using a semi-implicit projection method, treating the explicit part with a 3rd order Runge–Kutta scheme, and the implicit part with a 2nd order Crank–Nicolson scheme. All spatial derivatives are approximated using second-order central-differences on a staggered grid; this method is discretely kinetic-energy-preserving (KEP) in the inviscid limit [45]. To overcome the severe time step limitation imposed by the nature of the cylindrical coordinate grid, the viscous and convective terms are treated implicitly in the azimuthal direction, near the centerline, and in the radial direction near the top of the sphere. The pressure Poisson equation, which enables the projection of the predicted velocity into a divergence-free field, is solved using a direct solver. The eptadiagonal matrix is initially reduced to a series of pentadiagonal problems via Fast Fourier Transforms (FFTs) along the azimuthal direction, which are then solved with a divide-and-concur strategy [49]. The code is parallelized using a domain decomposition approach in the streamwise direction, where all communication between processors is handled utilizing Message Passing Interface (MPI) library calls.

The geometry of the sphere is represented by a Lagrangian grid consisting of triangular elements. The requirement for the Eulerian grid to conform to the body is relaxed, and the non-slip boundary conditions are imposed using an immersed-boundary formulation. Details on the boundary condition reconstruction and implementation within the framework of the fractional step scheme discussed above can be found in [50,51]. The overall formulation is 2nd-order accurate both in space and time. An extensive validation on practical flow problems over a wide range of Reynolds numbers can be found in [52–58].

The design of the grid for this solver is not trivial since the grid does not conform to the surface. The distribution of grid points in the radial and axial directions around the sphere for the intermediate grid is shown in Fig. 3a–b respectively. To control the distribution of the points inside the boundary layer the orientation of the outward normal with respect to the radial and axial directions was taken into account. In particular at $\theta \sim 84°$, the normal pointing away from the sphere is almost aligned with the radial direction. The radial grid needs to be fine there while the axial grid can be relaxed. Since $\delta \sim 0.03D$ at $\theta \sim 84°$ the radial resolution dr is $0.0045D$ resulting in approximately 7 points within the boundary layer. At $\theta = 45°$ ($r/D \sim 0.35$, $z/D \sim 0.35$) the outward normal is not aligned with either axis. To maintain the same resolution normal to the surface of the sphere the grid resolution is increased to $\Delta r = \Delta z = 0.003D$, resulting in approximately 6 points inside the boundary layer. At the stagnation point, where the normal is aligned with the axial direction, the axial grid is refined while the radial grid is coarsened. There $dz = 0.003D$, which puts 5–6 points inside the boundary layer. Finally at the back of the sphere the axial grid resolution is slightly relaxed since the flow is separated. Three different grids each with 50% refinement were employed; a coarse with $275 \times 84 \times 150$ points, an intermediate with $375 \times 118 \times 225$ points and a fine with $550 \times 161 \times 300$ points in the axial, radial and azimuthal directions respectively. The CFL number for all simulations was set to $CFL = 1.2$. The evolution of the forces on the sphere was monitored and when the forces reached a quasi-steady state behavior statistics were sampled over time. As for the other solvers above, the flow was sampled over $300D/U$ units after a transient of $200D/U$.

2.4. Test matrix

Table 2 summarizes the simulations performed in this study. Overall, 9 simulations were carried out, with coarse, medium and fine grids employed in each solver. The goal is to establish that results for each solver converge as the grid is refined before proceeding with comparing results among the solvers. It is worth noticing that the medium OpenFOAM grid (OF2) had a similar number of cells as the DNS performed by Rodriguez et al. [24], while the coarsest one (OF1) was slightly finer than the IBM-based LES carried out by Yun et al. [25]. We should also note that the initial condition is an important factor that can affect the overall cost of the simulations. For all three solvers we initialized the solution with the freestream velocity for the coarsest grid cases. For the finer grids, to minimize the transient towards a quasi-periodic state that is independent of the initial condition, we started with an interpolated solution from the coarser levels.

3. Results and discussion

3.1. Integral parameters and grid convergence

Fig. 4 shows contours of the time-averaged azimuthal vorticity, ω_y, along with isolines of the time-averaged streamwise velocity, U, for the full test matrix. It is evident that all solvers capture the main flows features for all three grid resolutions. In all cases, for example, the laminar boundary layer separates and forms a well-defined shear layer that encapsulates a large recirculation area in the near wake. Small differences in the length and structure of the recirculation bubble between the different solvers can be observed, but overall the mean flow structure is the same. Table 2 lists the predicted drag coefficient, $C_D = 2F_D/\rho U_\infty^2 A$ (where F_D is the drag force on the sphere, ρ is the fluid density, and A is the projected frontal area), Strouhal number, $St = fD/U_\infty$ (where f is the shedding frequency), recirculation bubble length, L/D, and the location where the laminar boundary layer separates indicated by the separation angle, θ_s. For the nine computations considered in this work the error for each variable with respect to the one predicted by finest grid in the Nek5000 simulation (N3) is also indicated. Available results from experiments and

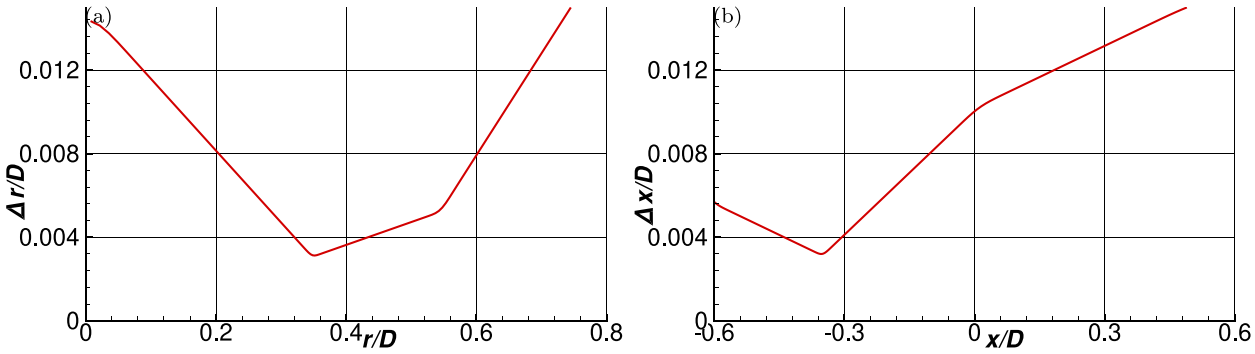

Fig. 3. Grid resolution around the sphere for the intermediate grid for the IBM solver (a) radial direction and (b) axial direction.

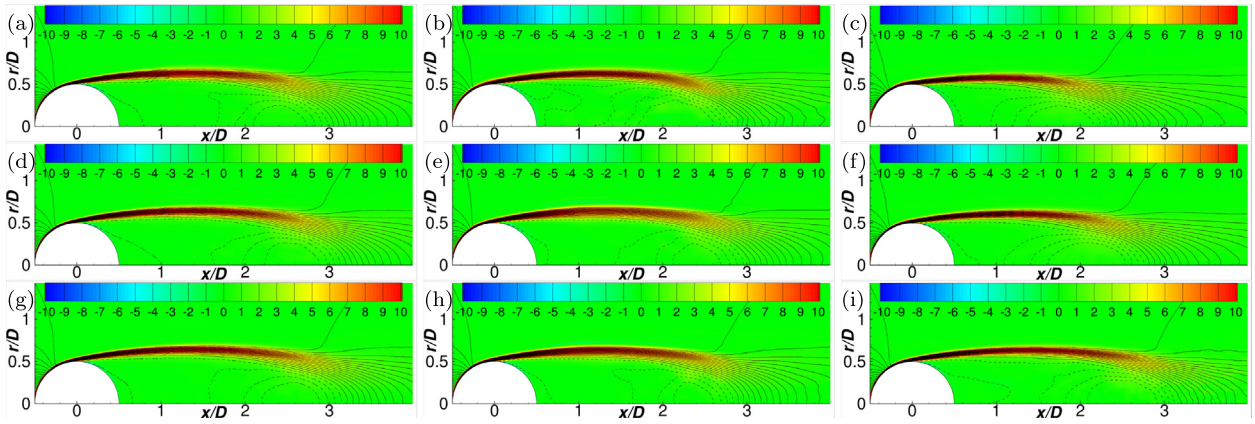

Fig. 4. Averaged azimuthal vorticity, $\omega_\theta D/U_\infty$, at an $x-r$ plane. Superimposed are isolines of the streamwise velocity. (a) coarse grid Nek5000; (b) coarse grid OpenFOAM; (c) coarse grid IBM; (d) medium grid Nek5000; (e) medium grid OpenFOAM; (f) medium grid IBM; (g) fine grid Nek5000; (h) fine grid OpenFOAM; (i) fine grid IBM.

Table 2
Main flow parameters provided by: (i) the simulations carried out in this work (top part of the table) and (ii) previous references (bottom part of the table). C_D is the drag coefficient, St the Strouhal number, L/D the non-dimensional length of the recirculation bubble, and θ_s the angle at which boundary layer separation occurs on the sphere. The percentage errors in the simulations are with respect to case N3.

Run/Reference	C_D	ΔC_D	St	ΔSt	L/D	ΔL	θ_s	$\Delta\theta$
N1	0.381	1.06%	0.23	4.54%	2.5	−5.4%	90.2°	0.33%
N2	0.376	−0.27%	0.22	0.00%	2.65	−0.6%	89.9°	0.00%
N3	0.377	–	0.22	–	2.67	–	89.9°	–
OF1	0.402	6.63%	0.21	−4.54%	2.22	−14.2%	88.5°	−1.6%
OF2	0.386	2.39%	0.21	−4.54%	2.48	−5.9%	89.2°	−0.7%
OF3	0.384	1.86%	0.21	−4.54%	2.36	−9.8%	89.8°	−0.1%
IBM1	0.356	−5.62%	0.24	9.09%	1.88	−24.9%	103.0°	14.5%
IBM2	0.370	−1.87%	0.23	2.22%	2.28	−12.3%	95.1°	5.8%
IBM3	0.371	−1.60%	0.22	0.00%	2.64	−0.9%	91.6°	1.9%
Rodriguez et al. [24]	0.394		0.215		2.28		89.4°	
Yun et al. [25]	0.355		0.21		2.62		90°	
Sakamoto et al. [26]	–		0.204		–		–	
Kim & Durbin [27]	–		0.225		–		–	

computations in the literature have also been added. A detailed study on the effects of grid resolution and time sample for the Nek5000 simulations can be found in [31].

For the case of Nek5000 all the above quantities converge and the difference between the two finest grids, N2 and N3, is less than 1%. The same applies to the IBM solver where all quantities converge as the grid is refined and the error on the finest grid (IBM3) is within 2% of N3. For the case of OpenFOAM the drag coefficient, C_D, also converges as the grid is refined and is within 2% of N3. The same applies to the separation angle, θ_s. The error on the recirculation bubble, however, does not converge monotonically and is relatively large when compared to N3 (>5%). In the process of designing the OpenFOAM grids we found that L/D is very sensitive to the adopted grid types and particularly to the way the mesh transitions from the boundary layer to the wake. The selected configuration was the one minimizing such errors. Note that St number predicted by OpenFOAM is not affected by the grid resolution and is 4.5% lower than N3.

At the bottom part of Table 2 we also include the results from past experiments [26,27] and eddy resolving simulations [24,25] at the same Reynolds number range. Although not all the parameters listed above are provided, overall the present computations are in agreement within 5%. One notable difference is the recirculation length L/D in the DNS by Rodriguez et al. [24], which is underpredicted by 12% with respect to N3 and IBM3. We should note that given the lack of a comprehensive set of reference

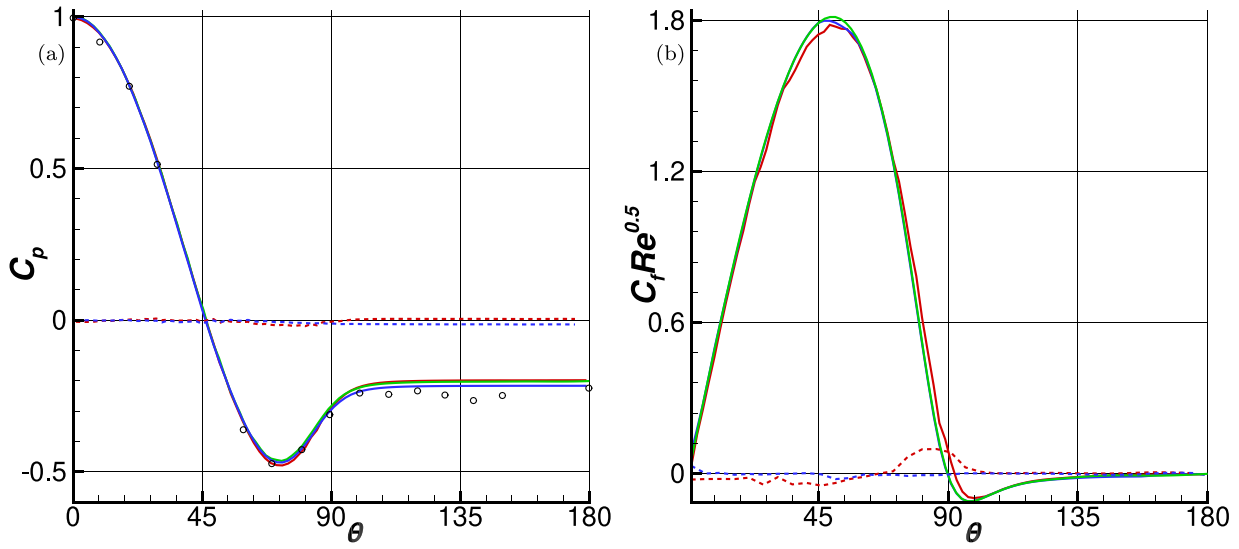

Fig. 5. Distribution of (a) average pressure coefficient C_p and (b) average skin friction coefficient C_f. Lines represent: IBM3; N3; OF3; diff. IBM-Nek5000; diff. OpenFOAM-Nek5000; ∘ Kim & Durbin [27] at $Re = 4200$.

experimental results we will use the grid-converged, spectral-element results for case N3 as the reference to compute all related errors involved in the assessment of cost-vs-accuracy below.

3.2. Pressure and skin-friction coefficients

An accurate prediction of the local pressure and friction forces on the surface of the sphere is of utmost importance. Fig. 5a shows the time-averaged pressure coefficient, $C_p = 2p/\rho U^2$ (where p is the pressure), for the three different solvers on the finest grids. At the front part of the sphere the pressure coefficient is in very good agreement among the three solvers, as well as with the experiments in [27], which are at a slightly higher Reynolds number, $Re = 4200$. At the back of the sphere, N3 and IBM3 are in excellent agreement, while the pressure predicted by OF3 does not recover as quickly. The latter is expected, given the discrepancies on the recirculation bubble length previously described. A similar trend is also indicated by the experimental results.

The time-averaged skin friction coefficient, $C_f = 2\tau_w/\rho U^2$ (where τ_w is the wall shear stress), is shown in Fig. 5b. Also in this case the agreement between the different solvers is very good. In particular, OF3 and N3 agree very well with each other while the peak of C_f in IBM3 is marginally lower and the separation point is overpredicted by two degrees. We should note that in the case of IBM methods the computation of C_p and (especially) C_f, which involves velocity derivatives, is not trivial. The IBM solver utilized in this work uses the method proposed by Wang et al. [59] for the computation of the hydrodynamic forces. The details of the formulation and extensive analysis of the accuracy and sensitivity of hydrodynamic forces on the grid resolution for various problems including fluid–structure interactions can be found in [59].

3.3. Mean velocity profiles

Profiles of the time-averaged axial velocity at three streamwise locations are shown in Fig. 6 for cases N3, OF3 and IBM3. The locations were selected to match the ones where experimental measurements have been reported by Kim & Durbin [27]. At $x/D = 0.2$, that is shortly after separation, the velocity profiles agree very well between the different codes. Agreement with the experiment is also excellent. At $x/D = 1.6$, which is inside the recirculation bubble in the near wake, the overall agreement is good with the exception that mean profile predicted by OF3 shows that the shear layer is shifted slightly downwards. This is consistent with the vorticity contours shown in Fig. 4, and is most likely due to numerical dissipation. At $x/D = 3.0$, which is at the end of the recirculation bubble, the differences are more pronounced. While the profiles for the IBM solver and Nek5000 agree well, OF3 does not predict flow reversal near the centerline due to the shorter recirculation bubble. The experimental results also point to a shorter recirculation area compared to the one predicted on N3 and IBM3. This is more clearly demonstrated in Fig. 7, where the evolution of the streamwise velocity along the centerline downstream of the sphere for N3, OF3 and IBM3 is shown. Here the difference between OpenFOAM and the other two solvers is more pronounced. The length of the recirculation bubble is challenging to predict in experiments and simulations as it experiences a periodic shrinking and enlargement motion known as a "bubble pumping", with a relatively low frequency. The reported frequencies for a sphere at this Reynolds number are in the range of $0.01 - 0.02\, D/U_\infty$ [24,31]. To capture such low-frequency motions, the statistical sampling time needs to enclose at least one or several bubble pumping cycles. The statistical averaging time of $300\, D/U_\infty$ employed in the current study is sufficient for this purpose. It was verified that extending the statistical averaging time to $900 D/U_\infty$ did not change the results [31].

3.4. Velocity fluctuations

We next look at each code's capabilities in predicting turbulent quantities. Fig. 8 shows contours of the streamwise turbulent intensities $u'_x u'_x$ and cross Reynolds stress $u'_x u'_r$ for the three solvers at the highest grid resolution. The dashed lines in the left part of the figure represent the location where profiles of $u'_x u'_x$ and $u'_x u'_r$ are extracted and shown in Fig. 9. All solvers predict a similar evolution of the near wake velocity fluctuations. Close examination of the streamwise component, $u'_x u'_x$, reveals that the detached shear layer in OpenFOAM becomes unstable a little closer to the sphere, while the magnitude of both $u'_x u'_x$ and $u'_x u'_r$ is generally higher. This trend is confirmed in all profiles shown in Fig. 9. Already at $x = 1.6D$ the peak in both quantities is over-predicted by OpenFOAM, while the maximum discrepancy is seen at $x = 2.5D$, where the peaks are roughly double in magnitude. This behavior is probably due to the early breakdown

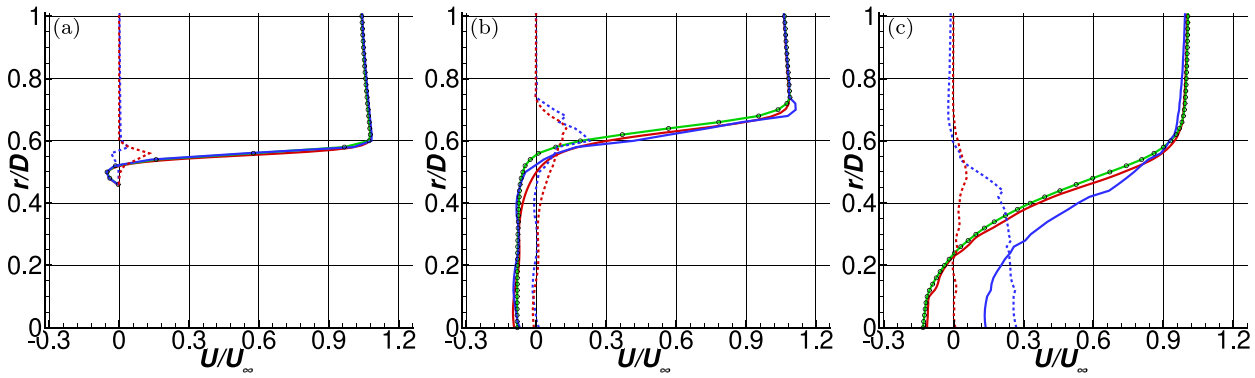

Fig. 6. Comparison of the average axial velocity U profiles among the codes at three different locations, (a) $x = 0.2D$ (b) $x = 1.6D$ and (c) $x = 3.0D$. Lines represent: IBM; Nek5000; OpenFOAM; - - - diff. IBM-Nek5000; - - - diff. OpenFOAM-Nek5000; ∘ Yun et al. [25].

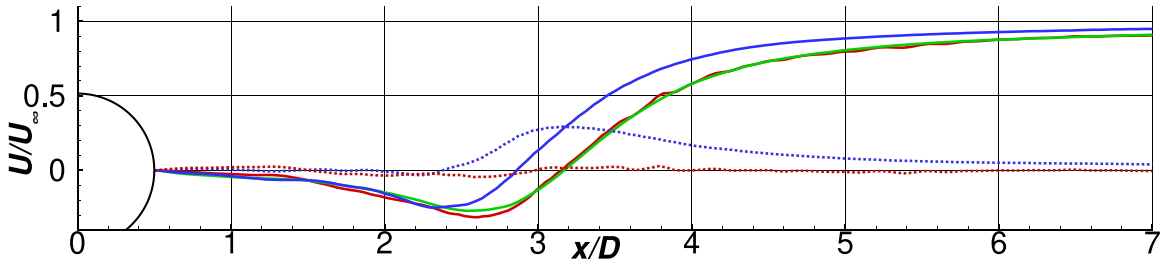

Fig. 7. Comparison of the average axial velocity U profiles among the codes along the centerline. Lines represent: IBM; Nek5000; OpenFOAM; - - - diff. IBM-Nek5000; - - - diff. OpenFOAM-Nek5000.

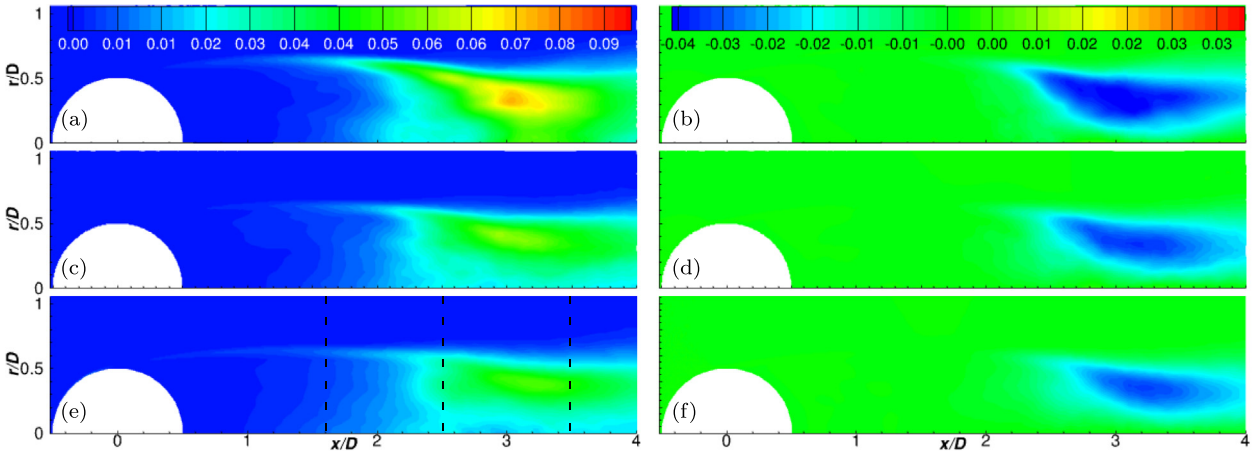

Fig. 8. Contours of the streamwise turbulent intensity $u'_x u'_x$ on the left and turbulent Reynolds stresses $u'_x u'_r$ on the right. (a) $u'_x u'_x$, OpenFOAM; (b) $u'_x u'_r$, OpenFOAM; (c) $u'_x u'_x$, Nek5000; (d) $u'_x u'_r$, Nek5000; (e) $u'_x u'_x$, IBM; (f) $u'_x u'_r$, IBM; The vertical dashed line correspond to the location of the profiles in Fig. 9.

of the shear layer and the shorter recirculation bubble. Overall the predictions by Nek5000 and IBM are in good agreement, and the finite-difference solver slightly under-predicts $u'_x u'_x$.

3.5. Cost-vs-accuracy

All Nek5000 simulations were conducted on the KNL nodes of the Stampede-2 supercomputer at the Texas Advanced Computing Center located at The University of Texas at Austin [60]. They utilize Intel Xeon Phi 7250 CPUs with a clock rate of 1.4 GHz and 96 GB DDR4 plus 16 GB high-speed MCDRAM for RAM. For the IBM and OpenFOAM computations we used the PEGASUS HPC cluster at the George Washington University consisting of Dual 20-Core 3.70 GHz Intel Xeon Gold 6148 processors with 96 GB of 2666MHz DDR4 ECC Register DRAM [61]. The specifics for each computation on the corresponding high-performance computing (HPC) resource are given in Table 3. The runs for all the solvers utilized different number of cores (<408 for Nek5000, <200 for OpenFoam, and <50 for IBM) and the corresponding wall-times (WT) for the same temporal sample ($300D/U_\infty$ units) were $30.9 < WT < 40.8$ for Nek5000, $36.7 < WT < 290$ for OpenFoam, and $2.3 < WT < 21.3$ for IBM. A direct comparison of the costs in Table 3 is not trivial given that the three solvers utilize different methods for spatial and temporal discretization as well as different parallelization strategies. In addition, different HPC platforms were utilized for Nek5000 as compared to OpenFOAM and IBM. Vermeire et al. [12], for example, introduced a cost metric referred to as resource utilization, which is the product of the cost of the hardware being used for a simulation in dollars and the amount of time that hardware has been utilized in seconds, with units \$ × sec. This metric effectively normalizes the computational cost by the price of the hardware used. A more

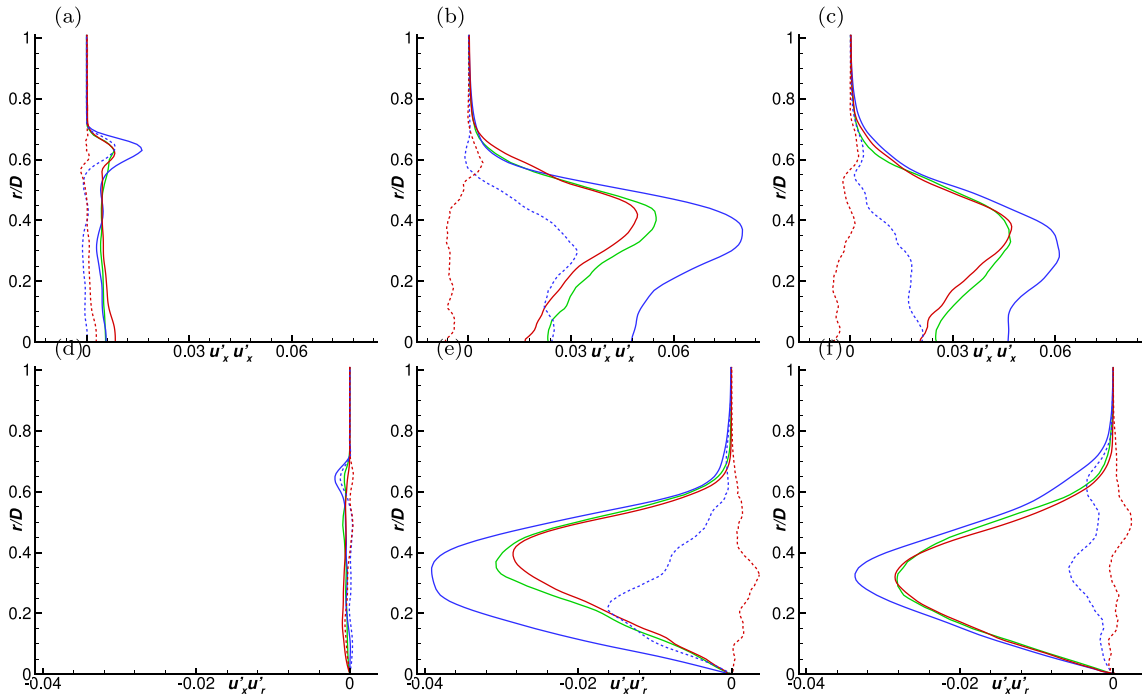

Fig. 9. Comparison of profiles of the streamwise turbulent intensity $u'_x u'_x$ and Reynolds stresses $u'_x u'_r$. (a) $u'_x u'_x$, at $x = 1.6D$; (b) $u'_x u'_x$, at $x = 2.5D$; (c) $u'_x u'_x$, at $x = 3.5D$; (d) $u'_x u'_r$ at $x = 1.6D$; (e) $u'_x u'_r$ at $x = 2.5D$; (f) $u'_x u'_r$ at $x = 3.5D$. Lines represent: IBM; Nek5000; OpenFOAM; diff. IBM-Nek5000; diff. OpenFOAM-Nek5000.

Table 3
Runtime data for each computation. Note that the number of timesteps and wall-time are given for a typical computation where equations are advanced in time for $300D/U_\infty$.

Run	HPC resource	# cores	grid size [$\times 10^6$]	# timesteps [$\times 10^4$]	wall-time [hours]
N1		136	2.8	15.0	30.9
N2	Stampede2	272	6.5	15.0	40.8
N3		408	9.8	15.0	40.7
OF1		40	3.7	6.0	36.7
OF2	Pegasus	80	13.4	8.5	248.0
OF3		200	29.7	12.0	290.0
IBM1		25	3.6	3.5	2.3
IBM2	Pegasus	25	10.3	5.2	10.2
IBM3		50	27.2	7.2	21.3

Table 4
Computational performances parameters. See text for details.

Run	$\tau[s]$	#dof/processor [$\times 10^3$]	$t^* \times N_p$	$\frac{t^* \times N_p}{\#\text{dof}}$ [$\times 10^{-6}$]	W
N1		20.6	1.0	0.36	0.14
N2	101.44	23.9	2.6	0.40	0.36
N3		24.0	3.9	0.40	0.54
OF1		92.5	5.96	1.61	0.33
OF2	14.76	167.5	56.4	4.21	4.47
OF3		148.5	117.9	3.97	13.10
IBM1		145	0.4	0.11	0.01
IBM2	14.76	410	1.2	0.11	0.06
IBM3		544	3.6	0.13	0.24

frequently used metric is the normalized CPU time utilized by a particular computation on a specific HPC platform. Typically the open-source tool *TauBench* [8] provides a work-unit on each computer to non-dimensionalize the CPU time. Given that the HPC architectures utilized in the computations in the present work are similar we will use the latter to obtain comparable cost estimates.

The results are reported in Table 4. To establish the work-unit for each of the utilized HPC platforms we used *TauBench*, and the resulting, work-unit, τ (in seconds), is listed in the second column. There is a significant difference in the execution time for the benchmark between Stampede2 and Pegasus since the latter utilizes newer CPUs. The normalized, elapsed CPU time per time step is indicated as $t^* = t/\tau$. Note that t was obtained by averaging the elapsed CPU time (i.e., the actual wall time) over the number of time-steps of each simulation. We did not perform scalability tests for the solvers to determine the overall efficiency, and the selection of degrees-of-freedom per core (*#dof*/core) was based on published results and prior experience with the solvers. In particular, Axtmann & Rist [62] include scaling results for the same solver of OpenFOAM utilized in the present work and report linear scaling when the $\#dof/\text{core} > 18,500$. We are well within this range because in our simulations the $\#dof/\text{core} > 92,000$. For the case of Nek5000, Rezaeiravesh et al. [63] report scaling results for a similar size problem that utilizes approximately 6.5×10^6 *dof*, demonstrating linear scaling to more that 1000 cores. This agrees with our own scaling tests conducted on KNL nodes (not shown here) and is in the range of $\#dof/core$ for all Nek5000 computations in the present work. The in-house IBM solver scales linearly for the reported $\#dof/core$ [52]. We should also note that all IBM simulations are conducted on a single node. We can therefore assume that all solvers were used within a very similar parallel efficiency range. The total cost can be defined as $t^* \times N_p$, where N_p is the number of processors used for each run. A metric that is directly linked to the efficiency of each solver is the normalized cost per timestep per dof ($t^* \times N_p/dof$), which is also listed in Table 4. Based on this metric OpenFOAM is the slowest among the three codes; it was, on average, slower by a factor ≈30 with respect to the IBM code and ≈10× slower with respect to Nek5000, in line with recent work [16]. As expected, the IBM code (a structured finite-difference solver) is the fastest one: roughly 3 times faster than Nek5000, on average. This can

be attributed to a superior efficiency of the Poisson solver for the pseudo-pressure implemented in the IBM code. The IBM pressure Poisson solver results in an epta-diagonal matrix which is first decomposed into a series of penta-diagonal systems of equations using trigonometric transformations along the periodic azimuthal direction. These are directly inverted using a divide-and-concur strategy [64]. We note that we did not attempt any optimization of performance in Nek5000 simulations, where a baseline XXT coarse grid pressure solver was used. Further optimization of performance of Nek5000 is possible, for example, with a newer algebraic multi-grid (AMG) solver [65,66], by replacing an over-integration with spectral filtering [67] for stabilization, and with $P_N - P_N$ formulations [68]. It is worth noting that the time step size is not involved in this comparison, which can be misleading as all three solvers use different time steps. To account for such differences we define the cost of each simulation as the CPU hours required to compute one flow-through time (D/U_∞). The resulting Work Unit (W) is thus defined as:

$$W = \frac{t^* \times N_p}{3600} \times \frac{D/U_\infty}{\Delta t}, \tag{4}$$

where Δt is the time step size. Even though the codes employ different time integration schemes, the time step was not drastically different for the various runs, $\Delta t \approx O(10^{-3})$.

The work unit W is shown in Fig. 10 for all nine computations versus the error on the integral parameters listed in Table 2. Nek5000 rapidly converges towards negligible errors at the medium grid resolution for all quantities. For the IBM solver the errors are significant for the coarsest grid but also converge rapidly as the grid is refined. For OpenFOAM, grid convergence is not as rapid and some quantities such as the Strouhal number, St and the length of the recirculation bubble, L/D, still have errors of the order of 5% even on the finest grid. We did not attempt further grid refinement for OpenFOAM given that the unit-cost, W, for the finest grid is 13.1, which is already an order of magnitude higher than the other two solvers, which are clearly more efficient. The power metrics for Nek5000 could be further improved by utilizing larger time steps. However, the sensitivity of results to a temporal refinement was not performed in the current study. Overall, if one sets the threshold of the error to be $\leq$2%, the IBM solver delivers it at the lowest cost ($W = 0.24$).

We should note that grid generation costs, which can be an important factor, are not included in the above cost-vs-accuracy estimates. For OpenFOAM and Nek5000 similar O-grid meshes were generated, using *blockMesh* and *Ansys Icem* respectively. The actual computational time to generate the grid in both cases was always below 1 CPU hour. Grid design, however, was found to affect OpenFOAM much more that the other two solvers, especially in the region of the near wake, where the laminar boundary layer detaches forming shear layers that consequently become unstable. This led us to utilize O-grid type meshes for OpenFOAM, rather then the fully unstructured grids with specific boundary layer zones (generated using the utility *snappyHexMesh*), which were found to be less robust. The latter requires additional input from the user and the computational cost is in the range of 60 CPU hours for producing grids with resolution equivalent to the finest O-grid considered above. Note that it also takes several attempts to fine-tune the grid properties, and as a result, the actual cost is a multiple of the above.

4. Summary

We performed a comparative study of the cost-vs-accuracy behavior of three radically different CFD approaches for the prototypical case of the flow over a sphere at subcritical Reynolds number, $Re = 3700$. Despite its relative simplicity, this problem incorporates many complex flow features and can be considered as a representative benchmark for the broad class of turbulent flows with separation. Furthermore, it has proven to be highly challenging for previous computational studies. Indeed, previous DNS and LES studies provide a relatively large range of variation for the results, suggesting that even bulk flow parameters are highly sensitive to numerical errors.

Each solver was tested at three grid resolutions. The grid design and overall *dof* used in each solver were very different but compatible with each method. Attention was paid to achieve comparable resolution in critical areas of the flow, such as the thin laminar boundary layers on the upstream part of the sphere and the detached shear layers in the near wake. All solvers converged to similar results at the finest grid resolutions. Nek5000 and IBM rapidly converged to errors of the order of 2% while for OpenFOAM the Strouhal number and the length of the recirculation bubble stagnated to around 5% even on the finest grid. To facilitate a direct comparison in terms of computational cost we established a work unit, W, that accounts for differences in hardware using *TauBench*. Based on this metric, the IBM fulfills the same accuracy requirements (i.e. 2%) with the lowest cost, while OpenFOAM turns out to be the most expensive. We should also note that we did not extend the comparison between the different solvers to higher-order moments and/or spectra that can be extracted from DNS of such flows, and for which there were no experimental reference results available for this case. Conservative, higher-order codes may have an advantage in such comparisons, but one can also argue that higher-order moments are not typically of interest in practical applications. This can be the topic of a future study focusing on acoustics, for example, in the context of cost-vs-accuracy comparisons.

Finally, caution should be taken in terms of generalizing the reported findings. Based on the selected prototype benchmark, our results suggest that high-order methods and second-order, kinetic-energy-preserving approaches based on the IBM may be both viable and efficient options for high-fidelity scale-resolving simulations of turbulent separated flows, when compared to standard finite-volume approaches. In complex flow scenarios, however, the actual geometry and boundary conditions are of course very important factors and may mitigate such differences (i.e. sharp edged objects, highly three-dimensional flows), or favor one approach over another. For the class of flows represented by the example we consider in this work, the main factor that can tip the balance related to cost-vs-accuracy is the Reynolds number. This is because immersed boundary methods are typically implemented within the framework of conservative, structured Cartesian solvers, which lack control of the grid resolution in areas of high velocity gradients such as boundary layers and shear layers (see for example [59]). As we discussed extensively for the case of the sphere (see Section 2), for example, the boundary layer thickness grows by the established $Re^{0.5}$ law. When generating an O-type boundary conforming grid, it is sufficient to refine the grid in one coordinate direction (normal to the sphere) and increase the number of points in a boundary layer for high Reynolds numbers. For an immersed boundary formulation however, refinement is required in all three coordinate directions to cluster more points in the boundary layer, and the cost grows faster with the Reynolds number when compared to boundary conforming formulations. This limitation can be nonetheless mitigated by the implementation of appropriate wall models. On the other hand, the IBM has a significant and undoubted advantage in the case of moving bodies and fluid–structure interaction problems, especially when large motions/deformations are present. Even in the case of fixed bodies, the cost (in terms of both human effort and CPU time) needed to generate suitable body-conforming grids can represent a significant bottleneck, especially in the context of large parameter studies. This aspect was not taken into account in the cost metric considered in the present work, and could further tip the balance towards IBM-based approaches.

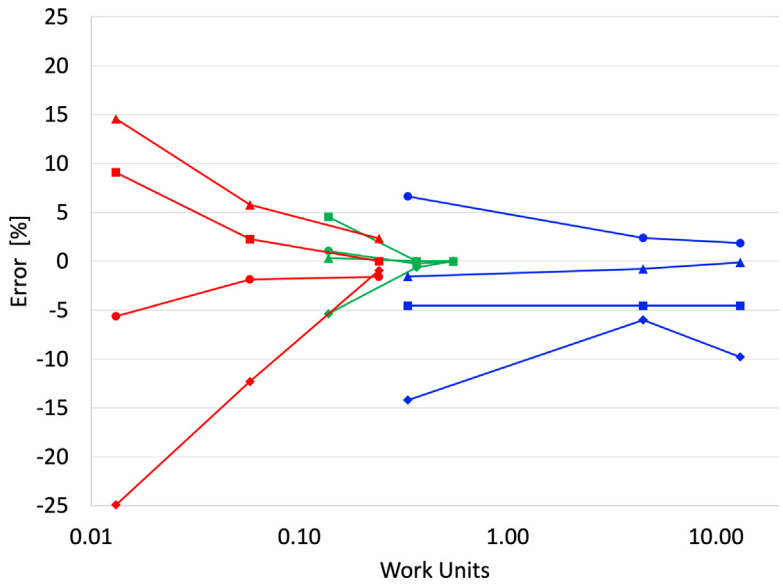

Fig. 10. Cost (W) vs. accuracy (% error) comparison for all cases. Symbols indicate errors on: ∘ C_D; □ St; ◊ L/D; △ θ_s. Nek5000; OpenFOAM; IBM.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Yulia Peet reports financial support was provided by National Science Foundation.

Data availability

Data will be made available on request

Acknowledgments

FZ and YP would like to acknowledge the support of NSF CBET 1707075 and 1944568 grants for this work. FC is a Serra Húnter fellow.

References

[1] R. Löhner, Towards overcoming the LES crisis, Int. J. Comput. Fluid Dyn. 33 (3) (2019) 87–97.
[2] K.A. Goc, O. Lehmkuhl, G.I. Park, S.T. Bose, P. Moin, Large eddy simulation of aircraft at affordable cost: a milestone in computational fluid dynamics, Flow 1 (2021).
[3] OpenFoam, https://www.openfoam.com.
[4] Basilisk, http://basilisk.fr.
[5] Nek5000, https://nek5000.mcs.anl.gov.
[6] PyFR, http://www.pyfr.org.
[7] The international workshops on high-order CFD methods, 2019, URL https://www1.grc.nasa.gov/research-and-engineering/hiocfd/.
[8] Taubench, 2022, Accessed: 2022-07-18, http://www.ipacs-benchmark.org/index.php?s=description&bm=TAUBENCH.
[9] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, M. Visbal, High-order CFD methods: current status and perspective, Internat. J. Numer. Methods Fluids 72 (8) (2013) 811–845, http://dx.doi.org/10.1002/fld.3767, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.3767, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.3767.
[10] 5th international workshop on high order CFD methods, Cenaero, 2017, https://how5.cenaero.be/.
[11] P. Moin, R. Verzicco, On the suitability of second-order accurate discretizations for turbulent flow simulations, Eur. J. Mech. B Fluids 55 (2016) 242–245.
[12] B.C. Vermeire, F.D. Witherden, P.E. Vincent, On the utility of GPU accelerated high-order methods for unsteady flow simulations: A comparison with industry-standard tools, J. Comput. Phys. 334 (2017) 497–521.
[13] STAR-CCM+, https://www.plm.automation.siemens.com/global/en/products/simcenter/STAR-CCM.html.
[14] F. Capuano, A. Palumbo, L. de Luca, Comparative study of spectral-element and finite-volume solvers for direct numerical simulation of synthetic jets, Comput. & Fluids 179 (2019) 228–237.
[15] G.L. Kooij, M.A. Botchev, E.M. Frederix, B.J. Geurts, S. Horn, D. Lohse, E.P. van der Poel, O. Shishkina, R.J. Stevens, R. Verzicco, Comparison of computational codes for direct numerical simulations of turbulent Rayleigh–Bénard convection, Comput. & Fluids 166 (2018) 1–8.
[16] S. Rezaeiravesh, R. Vinuesa, P. Schlatter, On numerical uncertainties in scale-resolving simulations of canonical wall turbulence, Comput. & Fluids 227 (2021) 105024.
[17] V. Saini, H. Xia, G.J. Page, Comparison of finite-volume and spectral/hp methods for large-eddy simulation of combustor port flow, AIAA J. (2022) 1–17.
[18] K. Schäfer, P. Forooghi, S. Straub, B. Frohnapfel, A. Stroh, Direct numerical simulations of a turbulent flow over wall-mounted obstacles—a comparison of different numerical approaches, in: ERCOFTAC Workshop Direct and Large Eddy Simulation, Springer, 2019, pp. 91–96.
[19] E. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (1) (2000) 35–60.
[20] S. Laizet, E. Lamballais, High-order compact schemes for incompressible flows: A simple and efficient method with quasi-spectral accuracy, J. Comput. Phys. 228 (16) (2009) 5989–6015.
[21] D. Goldstein, R. Handler, L. Sirovich, Modeling a no-slip flow boundary with an external force field, J. Comput. Phys. 105 (2) (1993) 354–366.
[22] M. Chevalier, A. Lundbladh, D.S. Henningson, Simson–a pseudo-spectral solver for incompressible boundary layer flow, 2007.
[23] A.G. Tomboulides, S.A. Orszag, Numerical investigation of transitional and weak turbulent flow past a sphere, J. Fluid Mech. 416 (2000) 45–73, http://dx.doi.org/10.1017/S0022112000008880.
[24] I. Rodriguez, R. Borell, O. Lehmkuhl, C.D.P. Segarra, A. Oliva, Direct numerical simulation of the flow over a sphere at Re=3700, J. Fluid Mech. 679 (2011) 263–287.
[25] G. Yun, D. Kim, H. Choi, Vortical structures behind a sphere at subcritical Reynolds numbers, Phys. Fluids 18 (1) (2006) 015102.
[26] H. Sakamoto, H. Haniu, A study on vortex shedding from spheres in a uniform flow, 1990.
[27] H. Kim, P. Durbin, Observations of the frequencies in a sphere wake and of drag increase by acoustic excitation, Phys. fluids 31 (11) (1988) 3260–3265.
[28] H. Schlichting, J. Kestin, Boundary Layer Theory, McGraw-Hill, 1979, vol. 7th edn..
[29] J. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, J. Comput. Phys. 215 (1) (2006) 12–40.
[30] C.E. Smith, N. Beratlis, E. Balaras, K. Squires, M. Tsunoda, Numerical investigation of the flow over a golf ball in the subcritical and supercritical regimes, Int. J. Heat Fluid Flow 31 (3) (2010) 262–273.

[31] F. Zhang, Y.T. Peet, The dynamics of coherent structures in a turbulent wake past a sphere at Re=3700, in: Proceedings of 12th Int. Symp. on Turbulence and Shear Flow Phenomena, Osaka, Japan, 2022.
[32] N. Beratlis, E. Balaras, K. Squires, On the origin of the drag force on dimpled spheres, J. Fluid Mech. 879 (2019) 147–167, http://dx.doi.org/10.1017/jfm.2019.647.
[33] F.M. White, Viscous Fluid Flow, McGraw-Hill, New York, 1991.
[34] A. Fage, Experiments on a sphere at critical Reynolds numbers, Aero. Res. Council Gt. Brit. (1936).
[35] J. Malm, P. Schlatter, P.F. Fischer, D.S. Henningson, Stabilization of the spectral element method in convection dominated flows by recovery of skew-symmetry, J. Sci. Comput. 57 (2) (2013) 254–277.
[36] P.F. Fischer, An overlapping Schwarz method for spectral element solution of the incompressible Navier–Stokes equations, J. Comput. Phys. 133 (1) (1997) 84–101.
[37] H.M. Tufo, P.F. Fischer, Fast parallel direct solvers for coarse grid problems, J. Parallel Distrib. Comput. 61 (2) (2001) 151–177.
[38] M.O. Deville, P.F. Fischer, E.H. Mund, High-Order Methods for Incompressible Fluid Flow, Cambridge University Press, Cambridge, UK, 2002.
[39] B.E. Merrill, Y.T. Peet, P.F. Fischer, J. Lottes, A spectrally accurate method for overlapping grid solution of incompressible Navier-Stokes equations, J. Comput. Phys. 307 (2016) 60–93.
[40] P. Fischer, J. Lottes, S. Kerkemeier, O. Marin, K. Heisey, A. Obabko, E. Merzari, Y. Peet, Nek5000 User's Manual, Technical Report ANL/MCS-TM-351, Argonne National Laboratory, 2015, http://nek5000.mcs.anl.gov.
[41] S. Dong, G.E. Karniadakis, C. Chrissostomidis, A robust and accurate outflow boundary condition for incompressible flow simulations on severely-truncated unbounded domains, J. Comput. Phys. 261 (2014) 83–105.
[42] R.I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, J. Comput. Phys. 62 (1) (1986) 40–65.
[43] H. Jasak, Error analysis and estimation for the finite volume method with applications to fluid flows, 1996.
[44] E. De Villiers, The Potential of Large Eddy Simulation for the Modeling of Wall Bounded Flows, Imperial College of Science, Technology and Medicine, 2006.
[45] G. Coppola, F. Capuano, L. de Luca, Discrete energy-conservation properties in the numerical simulation of the Navier–Stokes equations, Appl. Mech. Rev. 71 (1) (2019).
[46] V. Vuorinen, J.-P. Keskinen, C. Duwig, B.J. Boersma, On the implementation of low-dissipative Runge–Kutta projection methods for time dependent flows using OpenFOAM®, Comput. & Fluids 93 (2014) 153–163.
[47] E.M. Komen, E.M. Frederix, T. Coppen, V. D'Alessandro, J. Kuerten, Analysis of the numerical dissipation rate of different Runge–Kutta and velocity interpolation methods in an unstructured collocated finite volume method in OpenFOAM®, Comput. Phys. Comm. 253 (2020) 107145.
[48] E.M. Komen, J.A. Hopman, E. Frederix, F.X. Trias, R.W. Verstappen, A symmetry-preserving second-order time-accurate PISO-based method, Comput. & Fluids 225 (2021) 104979.
[49] J. Toivanen, T. Rossi, A parallel fast direct solver for block tridiagonal systems with separable matrices of arbitrary dimension, SIAM J. Sci. Comput. 20 (2001).
[50] E. Balaras, Modeling complex boundaries using an external force field on fixed cartesian grids in large-eddy simulations, Comput. & Fluids 33 (3) (2004) 375–404.
[51] J. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, J. Comput. Phys. 215 (1) (2006) 12–40.
[52] E. Balaras, S. Schroeder, A. Posa, Large-eddy simulations of submarine propellers, J. Ship Res. 59 (4) (2015) 227–237.
[53] A. Posa, A. Lippolis, R. Verzicco, E. Balaras, Large-eddy simulations in mixed-flow pumps using an immersed-boundary method, Comput. & Fluids 47 (1) (2011) 33–43, http://dx.doi.org/10.1016/j.compfluid.2011.02.004.
[54] A. Posa, A. Lippolis, E. Balaras, Large-eddy simulation of a mixed-flow pump at off-design conditions, ASME J. Fluids Eng. 137 (10) (2015) 101302, http://dx.doi.org/10.1115/1.4030489.
[55] A. Posa, E. Balaras, A numerical investigation of the wake of an axisymmetric body with appendages, J. Fluid Mech. 792 (2016) 470–498.
[56] M. Rahromostaqim, A. Posa, E. Balaras, Numerical investigation of the performance of pitching airfoils at high amplitudes, AIAA J. 54 (8) (2016) 2221–2232.
[57] A. Pal, S. Sarkar, A. Posa, E. Balaras, Direct numerical simulation of stratified flow past a sphere at a subcritical Reynolds number of 3700 and moderate froude number, J. Fluid Mech. 826 (2017) 5–31.
[58] A. Posa, E. Balaras, Large-eddy simulations of a notional submarine in towed and self-propelled configurations, Comput. & Fluids 165 (2018) 116–126.
[59] S. Wang, M. Vanella, E. Balaras, A hydrodynamic stress model for simulating turbulence/particle interactions with immersed boundary methods, J. Comput. Phys. 382 (2019) 240–263, http://dx.doi.org/10.1016/j.jcp.2019.01.010, URL https://www.sciencedirect.com/science/article/pii/S0021999119300270.
[60] Stampede2 user guide, https://portal.tacc.utexas.edu/user-guides/stampede2.
[61] GW high performance computing, https://hpc.gwu.edu.
[62] G. Axtmann, U. Rist, Scalability of openfoam with large eddy simulations and DNS on high-performance systems, in: W.E. Nagel, D.H. Kröner, M.M. Resch (Eds.), High Performance Computing in Science and Engineering ´16, Springer International Publishing, Cham, 2016, pp. 413–424.
[63] S. Rezaeiravesh, R. Vinuesa, P. Schlatter, On numerical uncertainties in scale-resolving simulations of canonical wall turbulence, Comput. & Fluids 227 (2021) 105024, http://dx.doi.org/10.1016/j.compfluid.2021.105024, URL https://www.sciencedirect.com/science/article/pii/S0045793021001900.
[64] T. Rossi, J. Toivanen, A Parallel Fast Direct Solver, Vol.21, Technical Report, 1996, pp. 1–20.
[65] N. Offermans, O. Marin, M. Schanen, J. Gong, P. Fischer, P. Schlatter, A. Obabko, A. Peplinski, M. Hutchinson, E. Merzari, On the strong scaling of the spectral element solver Nek5000 on petascale systems, in: Proceedings of the Exascale Applications and Software Conference 2016, 2016, pp. 1–10.
[66] N. Offermans, A. Peplinski, O. Marin, E. Merzari, P. Schlatter, Performance of preconditioners for large-scale simulations using Nek5000, in: Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2018, Springer, Cham, 2020, pp. 263–272.
[67] P. Fischer, J. Mullen, Filter-based stabilization of spectral element methods, C. R. Acad. Sci.-Series I-Math. 332 (3) (2001) 265–270.
[68] A. Tomboulides, J. Lee, S. Orszag, Numerical simulation of low Mach number reactive flows, J. Sci. Comput. 12 (2) (1997) 139–167.