Journal of Computational Physics 288 (2015) 1–18

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

MRAG-I2D: Multi-resolution adapted grids for remeshed vortex methods on multicore architectures

Diego Rossinelli [a], Babak Hejazialhosseini [a], Wim van Rees [a], Mattia Gazzola [a,b], Michael Bergdorf [c], Petros Koumoutsakos [a,*]

[a] *Computational Science and Engineering Laboratory, Department of Mechanical and Process Engineering, ETH Zurich, Switzerland*
[b] *School of Engineering and Applied Science, Harvard University, USA*
[c] *D.E. Shaw Research, New York, USA*

ARTICLE INFO

Article history:
Received 23 May 2014
Received in revised form 28 October 2014
Accepted 21 January 2015
Available online 2 February 2015

Keywords:
Wavelets
Adapted grids
Local time-stepping
Remeshing
Multicore architectures
Multipole methods
Bluff-body flows

ABSTRACT

We present MRAG-I2D,[1] an open source software framework, for multiresolution simulations of two-dimensional, incompressible, viscous flows on multicore architectures. The spatiotemporal scales of the flow field are captured by remeshed vortex methods enhanced by high order average-interpolating wavelets and local time-stepping. The multiresolution solver of the Poisson equation relies on the development of a novel, tree-based multipole method. MRAG-I2D implements a number of HPC strategies to map efficiently the irregular computational workload of wavelet-adapted grids on multicore nodes. The capabilities of the present software are compared to the current state-of-the-art in terms of accuracy, compression rates and time-to-solution. Benchmarks include the inviscid evolution of an elliptical vortex, flow past an impulsively started cylinder at $Re = 40$–$40\,000$ and simulations of self-propelled anguilliform swimmers. The results indicate that the present software has the same or better accuracy than state-of-the-art solvers while it exhibits unprecedented performance in terms of time-to-solution.

1. Introduction

The efficient simulation of incompressible flows at high Reynolds numbers requires the prudent allocation of computational elements to resolve the governing structures of the flow field. In simulations of flows with confined vortex structures (wakes, jets, etc.), discretizing the vorticity–velocity ($\omega - \mathbf{u}$) formulation of the Navier–Stokes equations (NSE) allows that computational elements are distributed only in the, limited, non-zero vorticity areas of the flow field. In addition, if we adopt the Lagrangian description of these equations, we can employ computational vortex elements that move adaptively with the velocity of the flow. These two concepts of adaptivity are key elements of remeshed vortex methods [1,2], where in addition to Lagrangian particles a grid is used to regularize particle locations and ensure the convergence of the method.

In recent years, remeshed vortex methods have been enhanced [3,4] by adapting the size of the computational elements (h-adaptation), in the spirit of Adaptive Mesh Refinement (AMR) [5]. A further extension [6], takes advantage of remeshing to introduce wavelet-based, multiresolution criteria [7] in the distribution of computational elements. It is important to

* Corresponding author.
E-mail address: petros@ethz.ch (P. Koumoutsakos).

[1] MRAG-I2D: *M*ulti-*R*esolution *A*dapted *G*rids for *I*ncompressible flow simulations in *2D*.

http://dx.doi.org/10.1016/j.jcp.2015.01.035
0021-9991/

note that enhanced adaptivity and a high compression rate in the number of employed computational elements may not be ultimately accompanied with computational savings, due to the increase of computational complexity of the associated algorithms. This fact that is often overlooked in multiresolution simulations and it is the focus of the present work.

Kevlahan and Vasilyev [8,9] investigated 2D flows past an impulsively started and moving cylinder at $30 < Re < 10^5$ using, both $\mathbf{u} - p$ and $\omega - \mathbf{u}$ NSE and the Adaptive Wavelet Collocation Method (AWCM) and report compression rates of up to 250–1000 : 1 compared to grid-based simulations at uniform resolution, and compression rates of about 10 : 1 compared to vortex methods. Furthermore, they observed a 70-250X improvement in time-to-solution (TTS) with respect to pseudospectral methods. Schneider and Farge [10] performed 2D simulations of the flow past an impulsively started cylinder at $Re = 3000$ with the wavelet-based Coherent Vortex Simulation (CVS) scheme combined with the Brinkman penalization technique for the $\omega - \mathbf{u}$ formulation of the NSE. They report a 10 : 1 compression rate and a TTS improvement of 3X over DNS simulations at uniform resolution. Deriaz and Perrier [11] investigated the use of divergence-free wavelets for the incompressible NSE in the context of turbulent flows. For the 2D problem of vortex merging, they showed bitrate-distortion plots for a disparate range of compression rates, without however reporting performance metrics.

Alam et al. [12] studied the simultaneous space–time wavelets solution of nonlinear parabolic PDEs with AWCM. For the 2D problem of vortex merging, they report a 2-4X improvement in TTS and a compression rate up to 20 : 1 compared to simulations at uniform resolution. Domingues et al. [13] coupled third-order average-interpolating wavelets with second-order local time-stepping schemes, observing TTS improvements of up to 25X over uniform simulations. Martin and Colella [14] presented a block-structured AMR, finite volume solver for incompressible flow simulations. For the 2D investigation of periodic shear layers they report a performance overhead of about 25–30% for dynamically adapting the grid on 3 levels of resolution. Roma et al. [15] coupled an immersed boundary method with AMR to study a 2D flow-mediated interaction between an elastic balloon and a wall. They do not report compression rates and performance numbers.

Wavelet and AMR approaches have been discussed in a number of papers (see [16] as well as [7] and references therein). Wavelets are capable of providing a consistent multiresolution framework that leads to higher data compressions, optimal convergence rate and automatic error control. On the other hand, wavelet-based solvers suffer from "three major curses" [16]: the efficient representation of complex geometries and irregular domains, the curse of anisotropy (for example the incapability of refining in just one arbitrary direction), and the high computational overhead due to their intricate data structures. These issues have been so far the key roadblocks on the widespread implementation of these methods on multicore and manycore platforms.

MRAG-I2D We provide this "missing link" by the present software that couples average-interpolating wavelets [17] and their multiresolution analysis with remeshed vortex methods on single multicore nodes. For the flow problems considered herein, the $\omega - \mathbf{u}$ formulation of the NSE is well suited, as the vorticity field is confined near the solid boundaries and the wake. This confinement is also an essential property for the efficient representation the flow field with wavelets. Here, wavelets are combined with local time-stepping (LTS) [13,18] schemes to further decrease the number of arithmetic operations. The resulting numerical schemes are inevitably complex. The underlying algorithm involves irregular computational patterns as well as unbalanced workloads. An efficient computer implementation requires that we effectively utilize the parallelism offered by current multicores[2] to exhibit good performance as quantified by high compression rates in computational elements and reduced time to solution.

The present MRAG-I2D framework efficiently maps wavelet-adapted grids and LTS schemes onto modern computing architectures. The software development relies on the integration of several HPC components designed and tested over the last 4 years [19–23] on both NUMA and heterogeneous CPU–GPU platforms. MRAG-I2D was developed by keeping high productivity as the foremost priority: the technical details about multiresolution analysis, wavelets and adapted grids are abstracted away. This allows users to formulate simulations without the need of explicitly interacting with a highly sophisticated spatiotemporally-adaptive machinery. The present software have been used for the simulations of bluff body flows [23], flow-mediated interactions between two cylinders [24], for the analysis of optimal shapes in anguilliform swimming at intermediated Reynolds numbers [25], and the investigation of reinforcement learning within the context of self-propelled bodies [26].

The present paper is organized as follows. In Section 2 we first discuss the governing equations and the treatment of deforming solid boundaries as well as the two-way flow-structure coupling. In Section 3 we discuss the numerical schemes employed for the discretization of the governing equations. In Section 4 we describe how the software effectively maps the workload onto a single multicore node. In Section 5, we examine the simulation of the evolution of an elliptical vortex, the flow past an impulsively-started cylinder at $Re = 40$–$40\,000$ as well as anguilliform swimming. We also report performance and data compression comparisons with respect to the state-of-the-art. Concluding remarks are presented in Section 6.

2. Governing equations

We consider two dimensional incompressible viscous flows in a domain (Σ) with a collection of N immersed, moving bodies. We denote by $\Omega_{i=1,\dots,N}$ the support of the bodies, which are assumed to be of the same density as the fluid

[2] Namely, instruction-, data-, and thread-level parallelism (ILP, DLP and TLP, respectively).

($\rho = 1$). The flow field is described by the NSE (Eqs. (1)–(2)) along with the no-slip boundary condition at the surface of the bodies $\partial\Omega_i$ (Eq. (3)). In the case of self-propelled bodies the feedback from the fluid to the bodies is governed by Newton's equations of motion (Eqs. (4)–(5)):

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \nu\nabla^2\mathbf{u}, \quad \mathbf{x} \in \Sigma \setminus \{\Omega_i,\ i = 1, ..N\} \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \qquad \mathbf{x} \in \Sigma \setminus \{\Omega_i,\ i = 1, ..N\} \tag{2}$$

$$\mathbf{u} = \mathbf{u}_i, \qquad \mathbf{x} \in \{\partial\Omega_i,\ i = 1, ..N\} \tag{3}$$

$$m_i\ddot{\mathbf{x}}_i = \mathbf{F}_i^H, \tag{4}$$

$$\frac{d(I_i\dot{\theta}_i)}{dt} = \mathbf{M}_i^H, \tag{5}$$

where ν is the kinematic viscosity, $\mathbf{u}_i$, $\mathbf{x}_i$, m_i, I_i and $\dot{\theta}_i$ are, respectively, the velocity, the position of the center of mass, the mass, the moment of inertia and the angular velocity of the body i, while $\mathbf{F}_i^H$ and $\mathbf{M}_i^H$ are the hydrodynamic force and torque exerted by the fluid on the body i. We note that the velocity field $\mathbf{u}_i = \mathbf{u}_i^t + \mathbf{u}_i^r + \mathbf{u}_i^{def}$ associated with the body i is the sum of three terms, namely the rigid, translational $\mathbf{u}_i^t$ and rotational $\mathbf{u}_i^r$ velocities, and an imposed deformation velocity field $\mathbf{u}_i^{def}$. The NSE can be rewritten in the $\omega - \mathbf{u}$ formulation:

$$\frac{\partial \omega}{\partial t} + (\mathbf{u}\cdot\nabla)\omega = \nu\nabla^2\omega, \tag{6}$$

$$\nabla^2\mathbf{u} = -\nabla\times\omega, \tag{7}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{8}$$

where $\omega = \nabla\times\mathbf{u}$ denotes the vorticity.

Flow-structure interaction The geometry of each body is represented by a characteristic function $\chi_i(t)$ and its kinematics by a deformation velocity field $\mathbf{u}_i^{def}(t)$. The characteristic function χ_i localizes the geometry in space by assuming a value of one inside the body and zero outside ($0 \le \chi_i \le 1$). The dynamics of the bodies, governed by their rigid components of motion (translational $\mathbf{u}_i^t$ and rotational $\mathbf{u}_i^r$ velocities), are the result of the flow-structure interaction captured by the solver.

The method of the Brinkman volume penalization [27–30] is used to enforce the no-slip boundary condition (Eq. (3)) at the surface of the body. Eq. (3) is replaced by adding the forcing term to the momentum equation (Eq. (1)). The penalized equations are applied then over the entire domain:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \nu\nabla^2\mathbf{u} + \sum_i \underbrace{\lambda\chi_i(\mathbf{u}_i - \mathbf{u})}_{\text{penalization term}}, \quad \mathbf{x} \in \Sigma \tag{9}$$

where $\lambda \gg 1$ is the penalization factor, and $\mathbf{u}_i = \mathbf{u}_i^t + \mathbf{u}_i^r + \mathbf{u}_i^{def}$.

The feedback from the flow to the body is captured through a projection approach [31–33]:

$$\mathbf{u}_i^t = \frac{1}{m_i}\int_\Sigma \chi_i\mathbf{u}d\mathbf{x}, \tag{10}$$

$$\dot{\theta}_i = \frac{1}{I_i}\int_\Sigma \chi_i(\mathbf{x} - \mathbf{x}_i)\times\mathbf{u}d\mathbf{x}. \tag{11}$$

These equations are in turn used to move the body and enforce the consistency of the extended flow field with respect to the body motion. We remark that the Brinkman penalization requires special care whenever the prescribed deformation velocity field is non-solenoidal ($\nabla\cdot\mathbf{u}_i^{def} \neq 0$). This introduces a dilatation source term and must be considered in the Helmholtz decomposition of the global velocity field. Eq. (7) is therefore replaced by:

$$\nabla^2\psi = -\omega, \tag{12}$$

$$\nabla^2\phi_i = \chi\nabla\cdot\mathbf{u}_i^{def}, \tag{13}$$

$$\mathbf{u} = \nabla\times\psi + \nabla\phi_i, \tag{14}$$

where ψ and ϕ are the stream function and velocity potential, respectively.

Time discretization The governing Navier–Stokes equations are advanced in time from t^n to t^{n+1} with a splitting scheme as follows:

$$\sigma^n = \sum_{i=1}^{N} \chi_i \nabla \cdot \mathbf{u}_i^{def,n} \tag{15}$$

$$\nabla^2 \psi^n = -\omega^n \tag{16}$$

$$\nabla^2 \phi^n = \sigma^n \tag{17}$$

$$\mathbf{u}^n = \nabla \times \psi^n + \nabla \phi^n \tag{18}$$

$$\mathbf{u}_i^{t,n} = \frac{1}{m_i} \int_{\Sigma} \chi_i^n \mathbf{u}^n dx \tag{19}$$

$$\dot{\theta}_i^n = \frac{1}{I_i} \int_{\Sigma} \chi_i^n \left(\mathbf{x} - \mathbf{x}_i^n\right) \times \mathbf{u}^n d\mathbf{x} \tag{20}$$

$$\mathbf{u}_i^{r,n} = \dot{\theta}_i^n \times \left(\mathbf{x} - \mathbf{x}_i^n\right) \tag{21}$$

$$\mathbf{u}_\lambda^n = \frac{\mathbf{u}^n + \lambda \Delta t \sum_{i=1}^{N} \chi_i^n (\mathbf{u}_i^{t,n} + \mathbf{u}_i^{r,n} + \mathbf{u}_i^{def,n})}{1 + \lambda \Delta t \sum_{i=1}^{N} \chi_i^n} \tag{22}$$

$$\omega_\lambda^n = \nabla \times \mathbf{u}_\lambda^n \tag{23}$$

$$\frac{\partial \omega_\lambda^n}{\partial t} = \nu \nabla^2 \omega_\lambda^n \tag{24}$$

$$\frac{\partial \omega_\lambda^n}{\partial t} + \nabla \cdot \left(\mathbf{u}_\lambda^n \, \omega_\lambda^n\right) = 0 \tag{25}$$

$$\omega^{n+1} = \omega_\lambda^{n+1} \tag{26}$$

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \mathbf{u}_i^{t,n} \Delta t^n \tag{27}$$

$$\theta_i^{n+1} = \theta^n + \dot{\theta}_i^n \Delta t^n. \tag{28}$$

3. Numerical schemes

In this section we discuss the adaptive, spatiotemporal discretization of Eqs. (15)–(28).

Adapted grids Biorthogonal wavelets are used to build a multiresolution analysis [34] of the flow quantities. With the Fast Wavelet Transform (FWT) we analyze the flow field and decompose it into a set of scaling and detail coefficients. The adapted grid is obtained by discarding grid points associated with detail coefficients with negligible energy.

In order to be able to employ standard finite-difference (FD) schemes on the adaptive grid, we create a frame of uniform resolution around a grid point. This frame is obtained by reconstructing *ghost* values, obtained as linear combinations of the surrounding active scaling coefficients. Because of the way ghosts are built, they do not affect the detail coefficients and are thus invisible to the FWT. The specific weights and source points of a ghost depend on the chosen wavelets as well as the local structure of the adapted grid [20,35]. First and second derivatives of the flow field are constructed with third-order accurate FD stencils. In order to simulate free-space boundary conditions, the stencils are completely one-sided at the boundaries of the computational domain, towards the domain interior. The stencil progressively becomes symmetric and centered for grid points away from the domain boundaries.

MRAG-I2D exclusively relies on first-generation wavelets, whose multiresolution analysis is constructed with dilated and translated versions of just one mother wavelet. Despite the moment-preserving features of second-generation wavelets, first-generation wavelets in this context are computationally advantageous. Their simpler structure enable the acceleration of the computation of the ghost values: the reconstruction pattern can be captured and stored into a lookup table for future reuse [19].

First-generation wavelets have been designed for infinite-length signals. Unfortunately, the shape of the wavelet function cannot be modified so as to take into account the proximity of the computational domain boundaries. This issue introduces complications in prescribing conditions as well as performing grid refinement or compression at the domain boundary. We address this issue by keeping a zone of uniform (coarse) resolution adjacent to the domain boundaries throughout the simulation. This allows us to straightforwardly prescribe boundary conditions (in the same way as for uniform resolution solvers) and avoid the difficulties involved in extending the grid to an infinite domain.

The grid is dynamically readapted according to the necessary changes in resolution, which are detected by performing FWTs on a frequent basis and examining the energy of the detail coefficients. We perform the FWTs on the quantities $\mathbf{u}$,

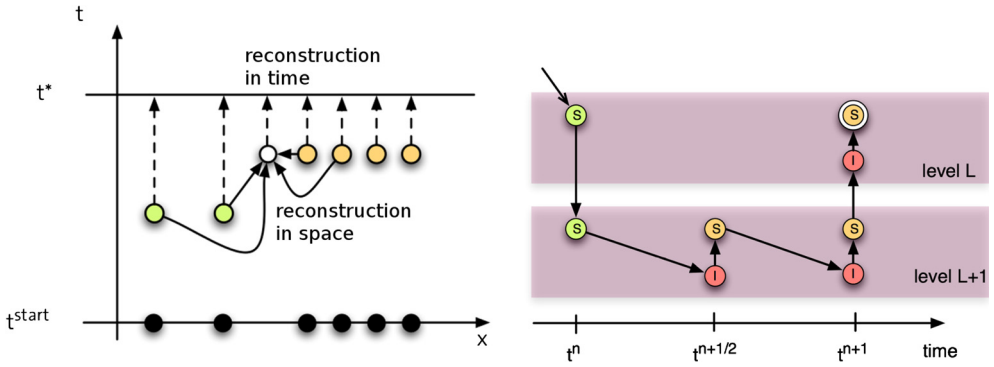


Fig. 1. Space–time diagram for the ghost reconstruction on an adapted grid (left) and associated marching graph for a second-order accurate LTS (right). Grid points (denoted in black) have different spatiotemporal scales and coordinates. All the points are used for the RHS evaluation at $t = t^*$. The red and orange nodes in the marching graph involve an RHS evaluation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

ω and χ_i. We refine grid points where the detail of any quantity exceeds $\epsilon_{\text{refinement}}$ and compress grid points where all the details of all the quantities are less than $\epsilon_{\text{compress}}$.

Local time-stepping Spatially-adapted grids indirectly decrease the amount of operations by removing computational elements. Operations can be further decreased by employing LTS schemes, which have been shown to accelerate the time-to-solution (TTS) by over one order of magnitude [12,13,35–37]. LTS schemes exploit the spatial locality of the time-stepping stability condition; grid points are grouped according to their time-stepping constraint and the flow quantities at coarse resolution are integrated with larger time steps thus avoiding unnecessary right-hand side (RHS) evaluations. To perform a global time-step, the LTS visit repeatedly these groups by evaluating the RHS and updating their values. Here we consider a second-order accurate total-variation diminishing LTS scheme [20].

The evaluation of the RHS is performed by considering the time-reconstructed values of the grid points. Fig. 1 (left) illustrates this process, which involves the space–time reconstruction of a high resolution ghost. As depicted in Fig. 1 (right), high-resolution grid points are subject to more RHS evaluations than coarser elements (four instead of two, for the illustrated case).

For the flow problems considered herein, by increasing the resolution the time-stepping stability condition of the viscous term is expected to asymptotically dominate the over the other time-stepping constraints (since $\delta t < \delta x_{\text{smallest}}{}^2/4\nu$, where $\delta x_{\text{smallest}}$ denotes the highest refinement allowed). We therefore employ an LTS scheme to integrate the viscous term and avoid unnecessary computation at coarser resolutions. Further details can be found in [23].

Multiresolution CFL-particles Particles, often coupled with Lagrangian descriptions of the governing Navier–Stokes equations as in vortex methods, provide automatic adaptivity and a minimal number of computational elements [1,38–40]. The accuracy of the method is ensured by a remeshing process where particle strengths are interpolated through a high-order moment conserving kernel (here we consider M'_4 [40,41]) onto an underlying regular grid at the end of each advection step [1,2]. The regularity of the underlying grid enables also the use of wavelet based adaptivity for particle methods [4,6].

In this work we consider a simpler multiresolution particle method, employed exclusively for the integration of the convection term. The method leaves the structure of the adapted grid unchanged after remeshing (no refinement or compression operations are performed at this stage). We discretize the vorticity field ω by means of particles, which are advected by the velocity field:

$$\frac{\partial \omega}{\partial t} + (\mathbf{u} \cdot \nabla)\omega = 0. \tag{29}$$

We first reconstruct a cloud of ghosts around an active scaling coefficient. The scaling coefficient and ghosts are then treated as particles: their location is integrated according to the velocity field. Despite possible local changes in resolution, the ghosts will always share the same resolution of the scaling coefficient. The scaling coefficient and the surrounding ghosts thus have the same particle size. After moving the particles, we remesh them at the grid points location thus obtaining the new solution. The cloud size is large enough to guarantee the partition of unity of the particle contributions within the range of one grid spacing.

Particles are advected with a second order Runge–Kutta time stepping scheme. When particles are advected with no CFL-like constraint, remeshing can take place across two different levels of resolutions. To avoid this issue we constrain the time step to a global CFL number less than 1. The CFL number is computed by considering the smallest grid spacing and maximum velocity over the entire grid. Since high-order wavelets are able to sense the signal in their surrounding, they anticipate the need for resolution adjustments in order to correctly accommodate the new solution.

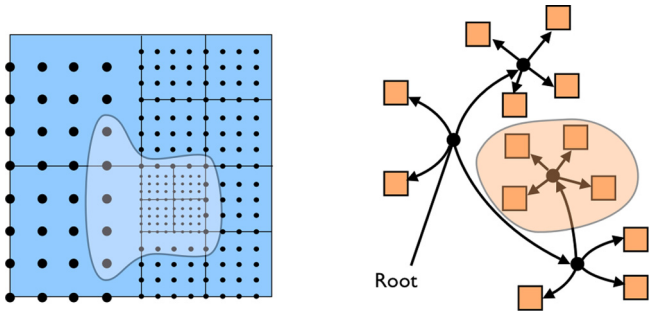


Fig. 2. Representation of blocks of 4×4 scaling coefficients in the physical space of the adapted grid (left) and associated data structure (right).

This limitation has negligible or no practical impact for bluff body flows. For the numerical investigations herein, the particle time-step is bound by the Lagrangian CFL condition and not the CFL condition, due to the strong velocity gradients around the bodies.[3]

Full algorithm To summarize, the spatiotemporal adapted discretization of Eqs. (15)–(28) is obtained by combining average-interpolating wavelets with an LTS scheme. The simulation consists of a series of global time-steps. Before each step, we perform FWTs and we refine the adapted grid where new scales are expected to emerge. The velocity field is then reconstructed with a tree code. Without changing the grid structure, Eqs. (15)–(28) are solved at the location of the scaling coefficients, in sequence. For the integration of the viscous term, an LTS scheme is used, where scaling coefficients are visited multiple times. For the integration of the convection term, we use a multiresolution particle method. The step is finalized by performing an FWT and removing the scaling coefficients associated with negligible details.

4. Multiresolution flow simulations on multicore architectures

Here we discuss the performance-aware design of MRAG-I2D to map the simulation onto multicores.

Structured blocks Wavelet-adapted grids are dyadic recursive structures entirely encompassing the computational domain. The grid points are the leaves of a tree. If the tree is fine-grained, the resulting compression rate is high. The downside of such a structure is the number of internal nodes, which is approximately the same as the number of grid points. This leads to detrimental performance issues when we operate on the grid points. Firstly, the parallelism in a tree traversal, necessary when evaluating the RHS or performing the FWT, is limited by the largest sequential task, corresponding to the path from the root to the farthest leaf. This issue grows along with the tree depth and thus the compression rate. Secondly, the performance of the access pattern incurred by fine-grained tree traversals is likely to be latency-bound because of the many memory indirections involved. The resulting utilization of the memory bandwidth is poor. The impact of this issue is further exacerbated if we take into account that the operational intensities [42] of the compute kernels considered here, except for those related to the tree code evaluation, are expected to be up to 4 FLOP/B (thus far below any ridge point).

To expose data-level parallelism (DLP) scaling coefficients are grouped into fixed-size structured blocks (Fig. 2, left), at the expense of a decreased compression rate. The grid is then represented with a tree which contains blocks as leaves (Fig. 2, right). Here blocks contain 32×32 points, the tree granularity is thus coarsened by 3 orders of magnitude with respect to single scaling coefficients.

Further details can be found in [20,35,43]. The introduction of blocks brings a number of performance benefits:

- **Higher utilization of the memory bandwidth.** A grid block is represented contiguously in memory. The evaluation of the RHS and FWT mostly exhibits unit-stride accesses. This is essential to extract decent memory bandwidth on multicore platforms.
- **Relaxation of latency-induced bottlenecks.** The tree contains $\approx 10^3$ less nodes than the one without blocks. This leads to a drastic decrease of the memory indirections involved in the tree traversal.
- **Enabling DLP.** The scaling coefficients within a block have the same resolution. The resulting computational workload is thus homogeneous. This allows us to consider both explicit and implicit vectorization for the evaluation of the RHS evaluation, the FWT as well as the tree code.
- **Enhancement of ILP.** By introducing blocks we observe a decrease in the number of ghosts. This in turn decreases the number of branching instructions resolved at runtime. By enabling DLP, the number of executed instructions to process a block decreases up to a factor of four to eight.[4] This indirectly enhances the ILP: the multicore out-of-order backend executes portions of code that are up to four times bigger, potentially better hiding data hazards.

[3] We observe $\delta x^2_{\text{smallest}} / \|\mathbf{u}\|_\infty \|\nabla \mathbf{u}\|_\infty \approx 10$ for all flow-body interaction problems presented herein.
[4] Considering the SSE and AVX SIMD-width, respectively.

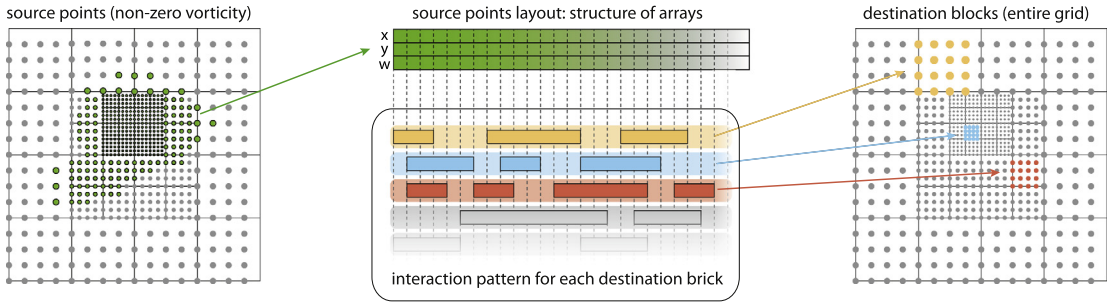


Fig. 3. Source grid points in the tree code (left), the direct interactions table contained in the plan (middle) relative to the adapted grid (right).

TLP issues and work-stealing Blocks are processed in parallel. The work associated to a block, being it FWT or RHS evaluation, is assigned exclusively to one thread. Because of the coarse-grained nature of the blocks, the simulation is subject to significant losses in the overall exposed TLP. This might adversely impact the overall performance. A second TLP issue is the work imbalance associated to the ghost reconstructions, since their cost depends on the local structure of the grid.

Here TLP issues are addressed by considering concurrent task-schedulers based on Work–Stealing, a scheduling algorithm developed by Blumofe et al. [44,45] to balance the work on each core. As soon as one parallel worker finishes its work, it steals tasks from the queue of another worker, in a breadth-first theft and depth-first fashion. Work-stealing is used both to address the work imbalance in the ghost reconstruction as well as to address irregular computational patterns of the tree code.

Tree code We solve the Poisson equation by employing a tree code [46] and considering multipoles expansions [46]. We reconstruct the velocity from the vorticity field according to the Biot–Savart integral. This approach naturally handles the free-space boundary conditions on the velocity. The advantages of using a tree code in the context of adapted grids are twofold. Firstly, its cost is factored into two separate terms: the number of sources (vorticity ω) and the number of targets/receivers (velocity $\mathbf{u}$). For the velocity reconstruction of N_{source} particles at N_{dest} locations, tree codes lead to the decreased computational cost of $\mathcal{O}(N_{\text{dest}} \log N_{\text{source}})$. Since the vorticity field has local support, the amount of sources is a fraction of the total computational elements. Secondly, tree codes and the block-structured multiresolution grid share similar or equivalent nested and dyadic structures. This enables the simplification of the computational patterns and a consequent performance boost that would not be possible with other Poisson solvers (e.g. those based on non-uniform FFT [47] or multigrid [48]).

Evaluations are always performed between single multipole expansions and full grid blocks. Prior to the evaluation, we construct a logical plan with all local and far-field interactions necessary to reconstruct the entire velocity field as illustrated in Fig. 3. The evaluation translates the plan into nested parallel tasks, which are then effectively scheduled with the Work–Stealing algorithm. Further details can be found in [49].

Software The code is written in C++, except for the HPC parts (about 10% of the total number of lines) which are explicitly vectorized with SSE intrinsics. Well-balanced workloads are carried out with OpenMP [50] (both compiler directives and library). TLP issues are addressed with the Intel Threading Building Blocks (TBB) [51,52], consisting of a library and a set of C++ headers. The code is organized into two layers: MRAG and MRAGapps. The MRAG layer deals with the technical details of the chosen wavelets, multiresolution analysis, TLP issues and the core algorithms behind spatiotemporally adapted grids [23]. MRAG is built upon 3th- and 5th-order average-interpolating wavelets as well as 2nd- and 4th-order interpolating wavelets. The MRAGapps layer contains instead application-specific client codes, such as the I2 client. MRAG-I2D relies on the Visualization Toolkit (VTK) [53] for the output and the postprocessing the simulation results. The code is on GitHub: https://github.com/cselab/MRAG-I2.

5. Applications

We demonstrate the accuracy and capabilities of the present software over several benchmark problems and applications pertaining in particular to flows past multiple deforming bodies. For these flow problems we allow the software to dynamically adjust the grid resolution within the minimum of 8×8 blocks and the maximum of 2048×2048 blocks. The simulations were performed on 4-way hexa-core AMD Opteron 6174 nodes with 64 GB of RAM, and 4-way quad-core AMD Opteron 8380 nodes with 32 GB of RAM.

5.1. Evolution of an inviscid elliptical vortex

We consider the evolution of an elliptical vortex in an inviscid fluid. This flow has been extensively studied in the context of remeshing with vortex methods [2]. We choose the initial condition according to case 1 in [2] defined by:

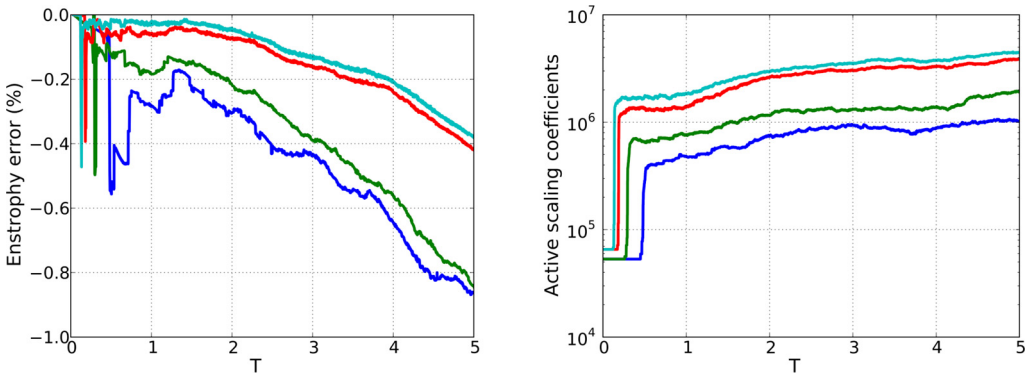


Fig. 4. Error in total enstrophy (left) and number of active scaling coefficients as a function of time for the evolution of an inviscid ellipse, for four different refinement and compression tolerances. The refinement tolerances are 1×10^{-1} (blue), 5×10^{-2} (green), 1×10^{-2} (red) and 5×10^{-3} (cyan). The compression tolerances are ten times smaller than the corresponding refinement tolerance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Resolutions and the corresponding compression factors of each of the refinement/compression tolerance combinations for the inviscid evolution of an elliptical vortex at $T = 5$.

Refinement tolerance	1×10^{-1}	5×10^{-2}	1×10^{-2}	5×10^{-3}
Compression tolerance	1×10^{-2}	5×10^{-3}	1×10^{-3}	5×10^{-4}
N_{eff} per dimension	4096	4096	4096	4096
N_{adaptive} per dimension	1008	1384	1967	2111
Compression factor	**16.5**	**8.8**	**4.3**	**3.8**

$$\omega_0(r) = \Lambda \begin{cases} 1 - f_q(r/R_0) & \text{if } r/R_0 \leq 1.0 \\ 0 & \text{else} \end{cases}, \tag{30}$$

where $f_q(z) = \exp(-(q/z)\exp(1/(z-1)))$. For the elliptical profile we consider an aspect ratio of 2, so that $r(x, y) = \sqrt{(2x)^2 + y^2}$. We set the parameters $\Lambda = 20$ and $q = 2.56085$, as in [2].

The inviscid nature of the fluid enables the use of some robust diagnostics for the errors made in solving the Euler equations. We monitor the error in the total enstrophy:

$$\frac{\mathcal{E}(t) - \mathcal{E}(0)}{\mathcal{E}(0)},$$

where the enstrophy is:

$$\mathcal{E}(t) = \int \omega^2(\mathbf{x}, t)\, d\mathbf{x}.$$

We use $R_0 = 0.4$ and place the elliptical vortex in the center of our unit square domain with effective resolution of 4096×4096, with 0.1 as CFL number.

Four different refinement and compression thresholds are chosen to show their effect on the error and compression rate. We vary the refinement tolerance as 1×10^{-1}, 5×10^{-2}, 1×10^{-2} and 5×10^{-3}, and for each case we choose the compression tolerance to be ten times smaller than the corresponding refinement tolerance.

Fig. 4 (left) shows that the error in enstrophy at $T = 5$ is only about 0.85% for the coarsest simulation, and decreases further to 0.4% for the finest. The reference simulations [2] show an error around this time between 1.5% and 2.5%, although with a smaller number of computational elements than employed herein. The increase in the number of scaling coefficients reduces the compression rate at $T = 5$ from 16 to about 4 for the finest grid, as quantified in Table 1. We note that the coarsest grid simulation, and to a lesser extent also the other tolerance combinations, show an increase in the enstrophy error on the first big refinement of the grid between $T = 0.1$ and $T = 0.5$.

In Fig. 5, the vorticity field and grid blocks are shown for the case where the refinement tolerance is 1×10^{-2} (the red curve in Fig. 4). At $T = 1$ the grid has only reached the maximum level of refinement in the center of the core, whereas the filaments are still smooth enough to be captured by lower levels of resolution. At $T = 5$, in contrast, the filaments are much thinner and the grid is refined to the maximum allowed level to capture their fine scales.

5.2. Flow past an impulsively-started cylinder at Re = 40–40 000

This flow problem involves a cylinder at rest for $t < 0$ in a fluid flow. At $t \geq 0$ the flow has a velocity of U_∞. The Reynolds number of this problem is defined as $Re = U_\infty D/\nu$ where D is the diameter and ν is the kinematic viscosity of the fluid. The cylinder experiences a drag force. The non-dimensional drag coefficient is defined as:

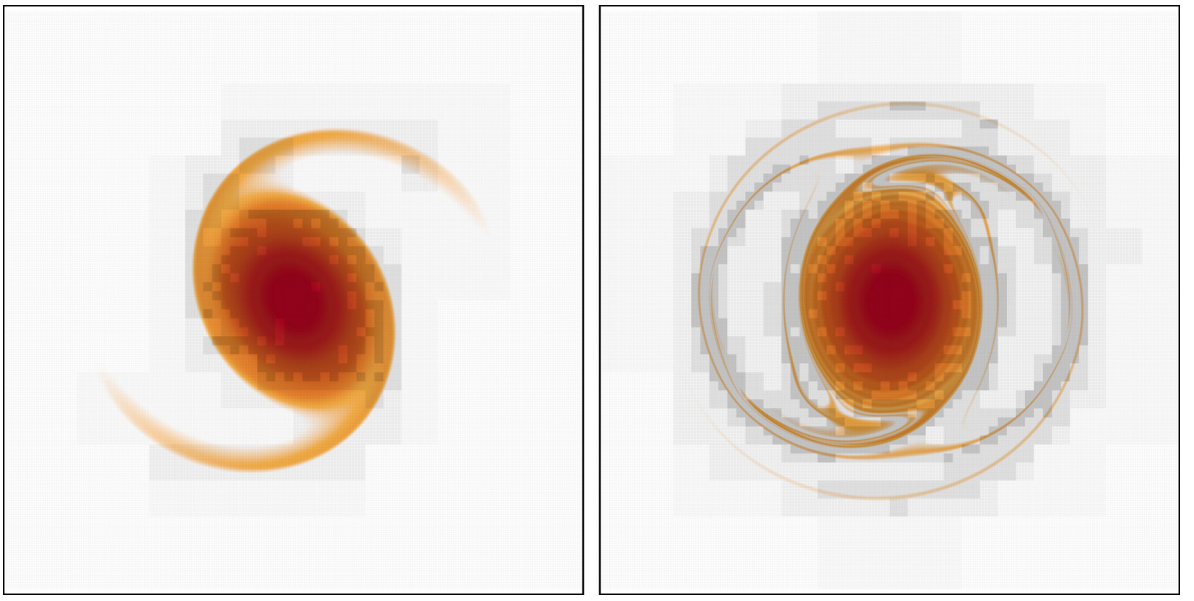


Fig. 5. Vorticity field at $T = 1$ (left) and $T = 5$ (right) overlaid with the adaptive computational grid for refinement and compression tolerances of 10^{-2} and 10^{-3}, respectively.

Table 2
Points per radius (ppr) of the cylinder, effective and adaptive resolutions and the corresponding compression factors for $Re = 40$–$40\,000$.

Re	40	100	3000	3000	9500	40 000
ppr	51	102	204	820	3276	13 104
N_{eff} per dimension	4096	8162	16 384	16 384	65 536	65 536
N_{adaptive} per dimension	404	525	690	1172	1210	1810
Compression factor	**102**	**240**	**562**	**195**	**2928**	**1311**
Measured at	$T = 40$	$T = 40$	$T = 20$	$T = 65$	$T = 5$	$T = 3$

$$C_D = \frac{F}{1/2DU_\infty^2},$$

where F is the force in the streamwise direction. The non-dimensional time of this problem is $T = 2U_\infty t/D$, where t is the dimensional time.

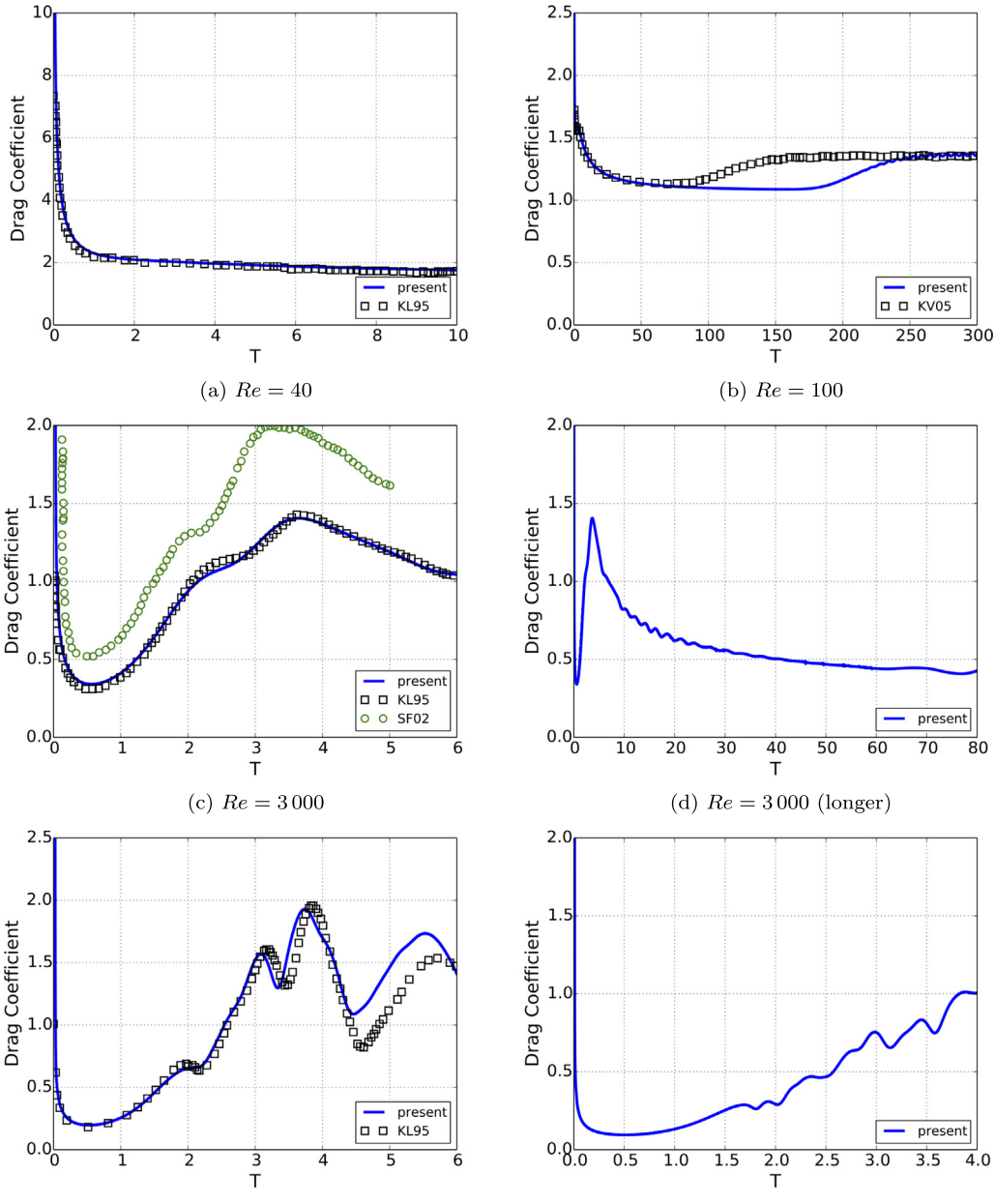
We simulate this problem at $Re = 40, 100, 3000, 9500$ and $40\,000$, with spatial resolutions given in Table 2. We use $U_\infty = 0.1$, $D = 0.0125$ (except for $Re = 40\,000$ where $D = 0.05$), penalization parameter $\lambda = 10^4$, $CFL = 0.01$ and the refinement and compression tolerances are 10^{-3} and 10^{-4}, respectively.

Fig. 6 reports the drag coefficient versus T. The drag curves at $Re = 40$ (a) and 100 (b) are compared with those reported in [1,8] and show good agreement. The present solver maintains the symmetry of the vortical structures at $Re = 100$ for much longer than the one reported by Kevlahan and Vasilyev [8].

In Fig. 6(c), we plot the drag at $Re = 3000$ from the present simulations as well as from those in [1,10]. The present method clearly shows more agreement with the reference compared to [10]. Our drag curve is also in very good agreement with that presented in [8] (using velocity-pressure formulation) until $T = 6$ for a similar number of points per radius. At $Re = 3000$, we continued the simulation up to time $T = 90$ and visualized the vorticity field in Fig. 7 at four different times. The simulations show the growth of the initial major vortex pair into a large recirculation region behind the cylinder, with a corresponding decrease in the drag coefficient plotted in Fig. 6(d). At around $T = 45$ the vortex pair becomes unstable, the main symmetry in the flow is broken and each vortex rolls up and sheds into the wake. The loss of this recirculation region immediately results in an increase in the drag coefficient. After $T = 70$, smaller vortex pairs start to shed downstream.

Increasing the Reynolds number further, we show the vorticity field at $Re = 9500$ in Fig. 8. The main flow features of this case have been extensively studied in [1] and the current results show quantitative and qualitative agreement with the results reported in this reference. We notice that this is the first adaptive simulation at $Re = 9500$ which shows such a close agreement to the reference (see [23]).

Finally, we present the drag curve of the flow past an impulsively-started cylinder at $Re = 40\,000$ in Fig. 6(f) and the associated vorticity field in Fig. 9. Rather than the formation of a few strong vortices as in $Re = 9500$, we observe a rapid succession of smaller vortices breaking off from the shear layer. This is reflected in the drag curve, where a big increase in the drag coefficient corresponding to the shedding of a large vortex pair (as in Fig. 6(e)) is absent, and the drag coefficient



(a) $Re = 40$ (b) $Re = 100$

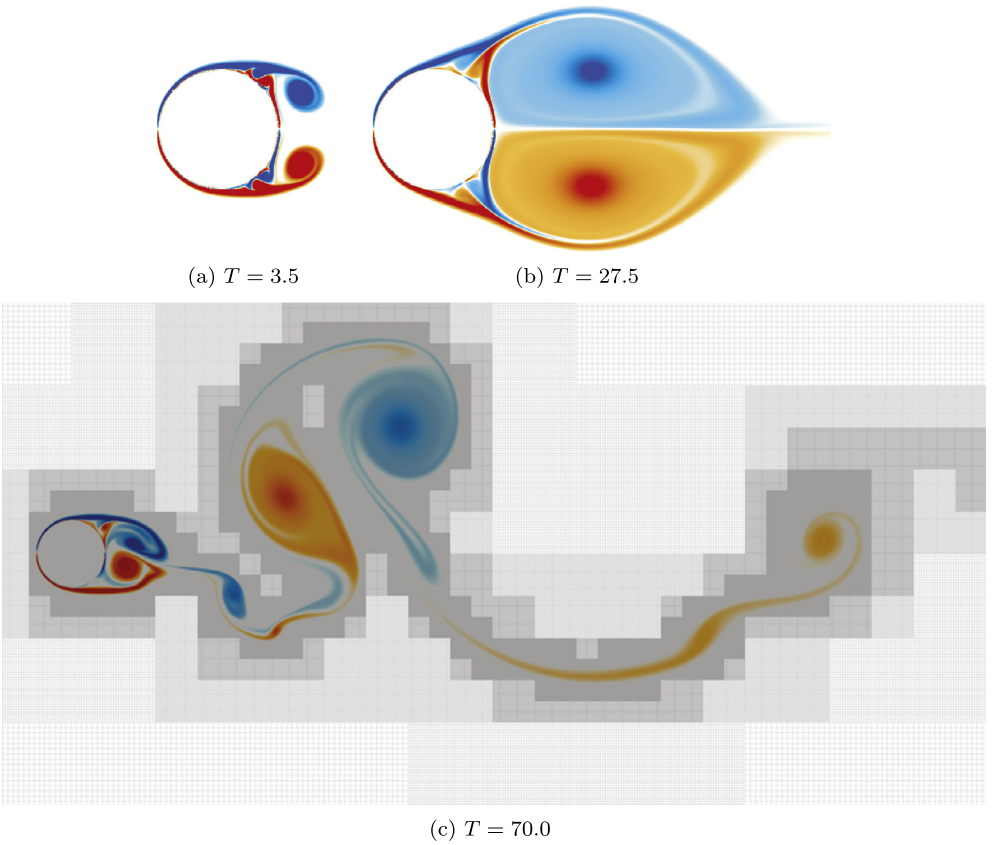
(c) $Re = 3\,000$ (d) $Re = 3\,000$ (longer)

(e) $Re = 9\,500$ (f) $Re = 40\,000$

Fig. 6. Evolution of the drag coefficient at $Re = 40$–$40\,000$. Reference data (empty symbols) is taken from [1] (KL95), [8] (KV05) and [10] (SF02).

stays low with some small oscillations starting around $T = 1.7$. These local oscillations correspond to the separation of the main vortices and their interaction with smaller filaments of opposite strength.

The compression rates associated with these simulations range from two to three orders of magnitude (Table 2) and are in general observed to increase with the Reynolds number. At high Reynolds numbers, typically the vortical structures in the flow become more compact, with steep gradients along their boundaries. We compare the TTS of the uniform resolution simulation of [54] at $Re = 3000$ with $16\,384 \times 4096$ grid points to our adaptive one at the same effective resolution in Fig. 10 (left). We notice that for a large fraction of the simulation, the adaptive solver performs two orders of magnitude faster than the uniform resolution one. Fig. 10 (right) depicts the evolution of the number of active scaling coefficients up to $T = 70$ for simulations with different compression thresholds. We notice that at early times, flow structures and the adaptive grids are very similar as the region in the vicinity of the boundary of the cylinder is refined to the maximum allowable resolution. As vortical structures move farther from the cylinder, adaptive grids with different thresholds show significant difference. The two largest thresholds result in a very early loss of symmetry and the shedding of the first vortex pair ($T = 60$ with an increase in the number of active scaling coefficients). On the other hand, the number of scaling coefficients does not



(a) $T = 3.5$ (b) $T = 27.5$

(c) $T = 70.0$

Fig. 7. Vorticity field of the flow past an impulsively-started cylinder at $Re = 3000$ at different times. Vorticity color map is scaled from -100 to 100.

show an appreciable difference for the two smallest thresholds. For this reason, we report the evolution of the drag as well as the compression rates and timings of $Re = 3000$ at compression threshold of 10^{-4}. The left picture of Fig. 11 reports the TTS for the simulation at $Re = 40\,000$. The right picture shows the nominal peak compute performance versus the measured performance on the AMD 8380 node during the evaluation of the tree code. The measured performance is about 170–190 GFLOP/s, corresponding to 55% of the nominal peak in average. Overall, the simulation reaches about 40% of the nominal peak.

5.3. Self-propelled swimming

We present validation studies and new results for the simulation of a single anguilliform swimmer with identical geometry and deformation as in [55]. The Reynolds number in this case can be defined as $Re = (L^2/T)/\nu$, where L is the swimmer's length, T is the period of the traveling wave defining its motion, and ν is the kinematic viscosity. We show results at $Re = 7143$ and 15 000 for the forward and lateral velocity (in fish-lengths/cycle unit) of the swimmers in Fig. 12.

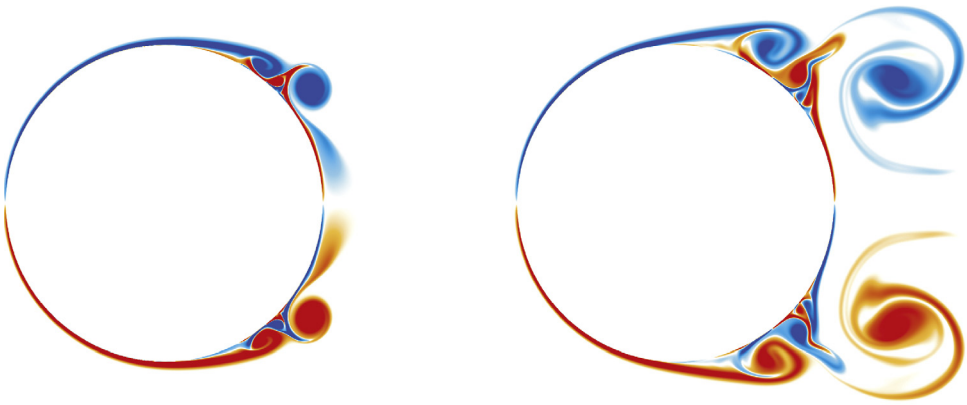
We compare our results at $Re = 7143$ with those reported in [55], obtained using a finite volume scheme on a body-fitted mesh. Resolutions and compression factors of the two simulations are presented in Table 3. Fig. 13 depicts the vorticity field of an anguilliform swimmer at $Re = 15\,000$ as well as the adapted the grid. We note that the swimmer at this Reynolds number reaches a slightly larger forward velocity compared with $Re = 7143$. The structure of the wake is similar to the lower Reynolds number case, showing a regular staggered array of opposite-signed vortices connected by thin filaments.

We observe a $10:1$ compression rate over the simulation at uniform resolution. Larger compression rates are hindered by the large number of high resolution blocks needed to resolve the wake.

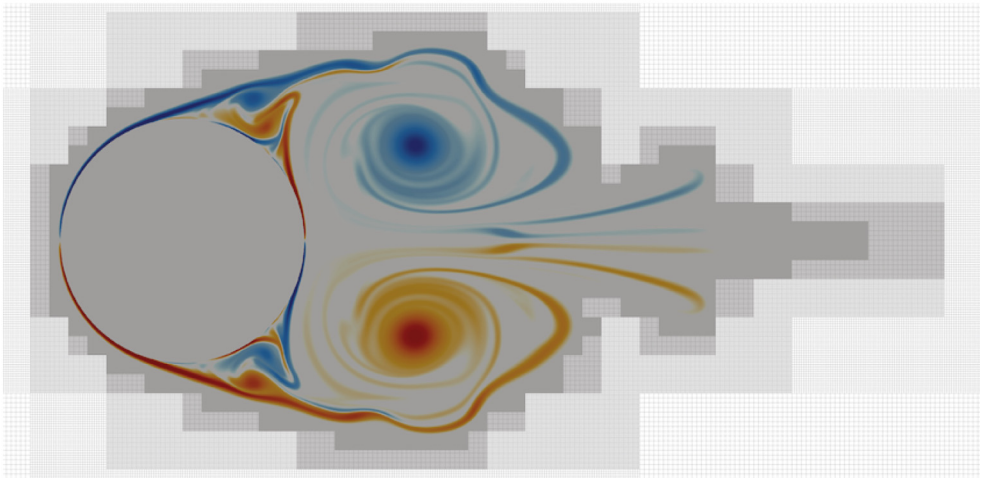
We compare the TTS of the anguilliform swimming at uniform resolution simulation [56] with 4096×2048 grid points to our adaptive one at the same effective resolution in Fig. 14. We notice that the adaptive solver performs more than seven times faster than the uniform resolution one.

5.4. Multi-object cases

We present two flow problems where we simulate the interaction of flow with multiple objects. We illustrate the flow field for the real-time simulation of an array of wind turbines at $Re = 300$ [23] in Fig. 15 (top). In this case, the location

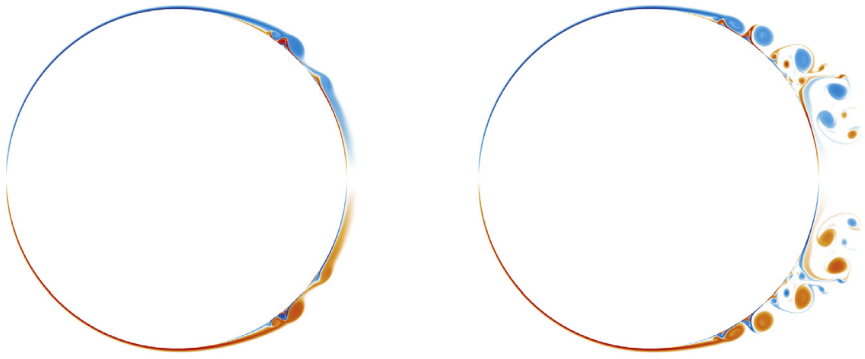


(a) $T = 2.1$ (b) $T = 3.8$



(c) $T = 11.0$

Fig. 8. Vorticity field of the flow past an impulsively-started cylinder at $Re = 9500$ at different times. Vorticity color map is scaled from -400 to 400.



(a) $T = 1.2$ (b) $T = 2.4$

Fig. 9. Vorticity field of the flow past an impulsively-started cylinder at $Re = 40\,000$ at different times. Vorticity color map is scaled from -600 to 600.

of turbine axes is fixed and they are allowed to rotate according to their interaction with a free stream flow. We then demonstrate the flexibility of the proposed tool for the simulation of self-propelled swimming of five fish with different shapes and motion parameters (at $Re = 5000$) in Fig. 15 (bottom).

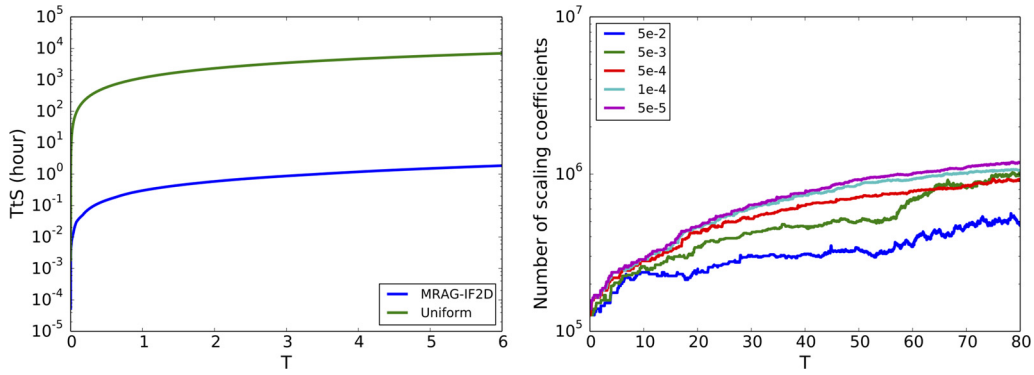


Fig. 10. Left: TTS in wall clock hours versus non-dimensional physical time (T) for the flow past an impulsively-started cylinder at $Re = 3000$. Right: Number of active scaling coefficients versus non-dimensional physical time (T) for the flow past an impulsively-started cylinder at $Re = 3000$ with different compression thresholds.

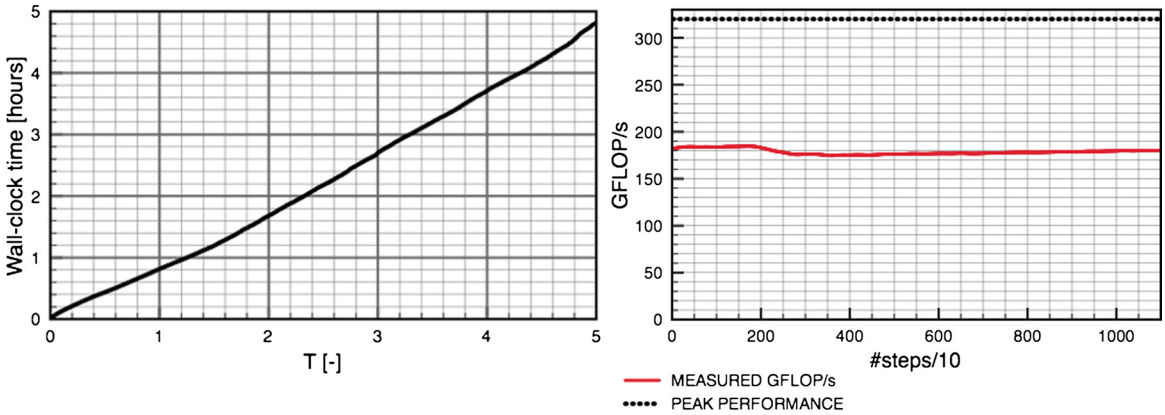


Fig. 11. TTS and measured performance during the evaluation of the tree code (right) for the flow simulation past an impulsively started cylinder.

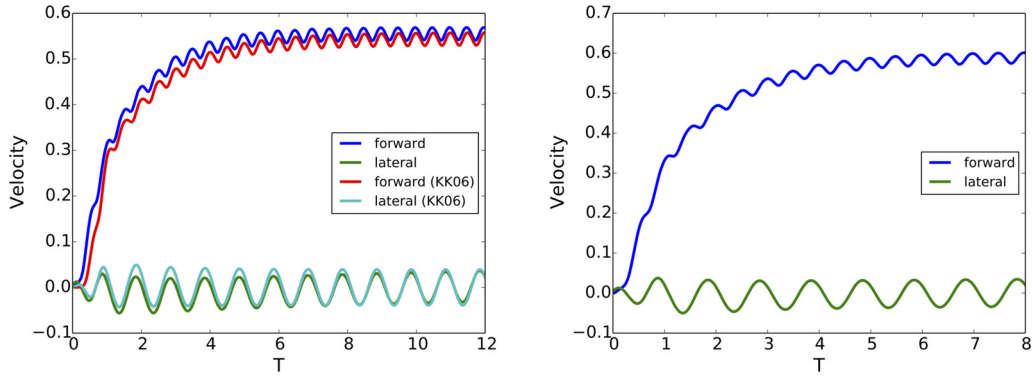


Fig. 12. Evolution of the forward and the lateral velocities of an anguilliform swimmer at $Re = 7143$ (left) and at $Re = 15\,000$ (right). Reference data for $Re = 7143$ is taken from [55].

Table 3
Points per chord (ppc) of the swimmer, effective and adaptive resolutions and the corresponding compression factors for the simulations at $Re = 7143$ and $Re = 15\,000$.

Re	7143	15 000
ppc	819	1638
N_{eff} per dimension	8192	16 384
N_{adaptive} per dimension	1438	1843
Compression factor	**32**	**80**
Measured at	$T = 8.5$	$T = 5.2$

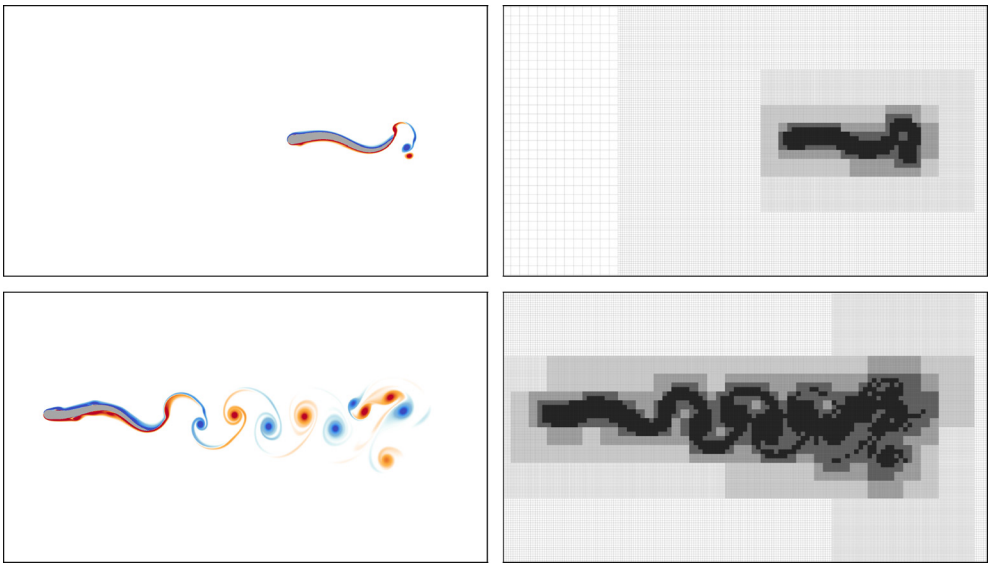


Fig. 13. Vorticity field (left) and the adaptive grid (right) for an anguilliform swimmer at $Re = 15\,000$ at $T = 1.0$ (top) and $T = 5.2$ (bottom).

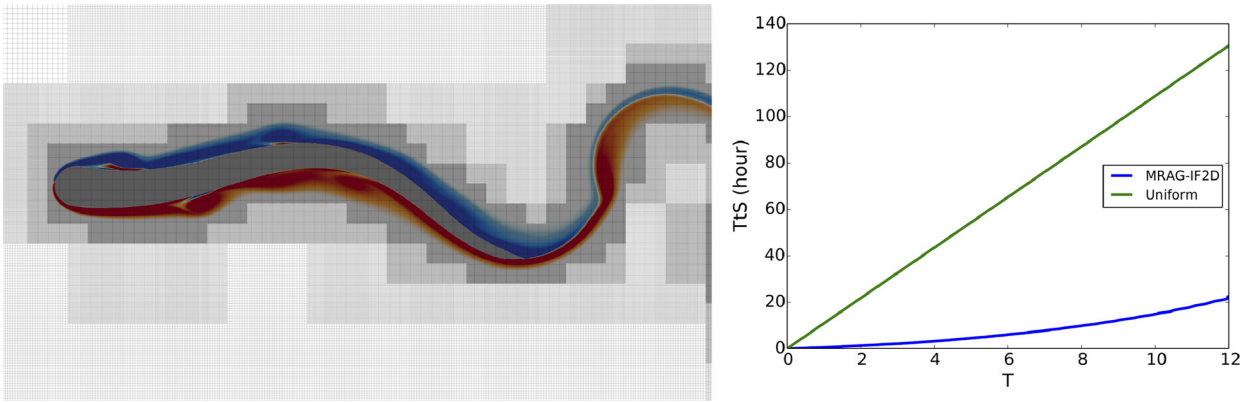


Fig. 14. Closeup of the vorticity field and the adaptive grid for an anguilliform swimmer at $Re = 15\,000$ at $T = 5$ (left), and TTS in wall clock hours versus non-dimensional physical time (T) for anguilliform swimmer at $Re = 7143$.

6. Conclusions

We have presented MRAG-I2D, an open source software for two-dimensional incompressible flow simulations on multicore architectures. We describe the numerical schemes and the key HPC strategies employed in MRAG-I2D to effectively map the numerical methods on multicore architectures. Multiple spatiotemporal flow scales are captured by combining wavelets with local time-stepping schemes. The Poisson equation is solved using a tree based multipole solver designed for block-structured multiresolution grids that is efficiently implemented on multicore architectures. We have examined accuracy, memory footprint and performance of simulations for a variety of flow problems. For the flow past an impulsively-started cylinder at $Re = 40$–9500, MRAG-I2D demonstrates high accuracy, in computing the drag coefficient, over the current state-of-the-art. We have also presented simulations at $Re = 40\,000$, characterized by multiple roll-ups of the initial thin vortex layer, suggesting an instability mechanism for the onset of unsteady separation at high Re number flows. Further tests include simulations of self-propelled anguilliform swimming and comparison with related works. We report significant improvements in time-to-solution over uniform resolution remeshed vortex methods: about 10X for anguilliform swimming and 100X for flow past an impulsively-started cylinder at $Re = 3000$. MRAG-I2D maintains overall a sustained performance at 40% of the nominal peak, reaching about 55% during the execution of the most compute intensive kernels. The use of interpolating wavelets on the interval along with the block structure of our code enables unprecedented compression rates that facilitate the I/O of such simulations. We have achieved large compression rates of computational elements: 32X and 80X for swimming at $Re = 7143$ and $15\,000$, respectively and 100X–1300X for flow past impulsively-started cylinders up to $Re = 40\,000$. Compared to the current state-of-the-art, MRAG-I2D shows substantial improvements in performance and productivity while reaching equivalent or better accuracy.

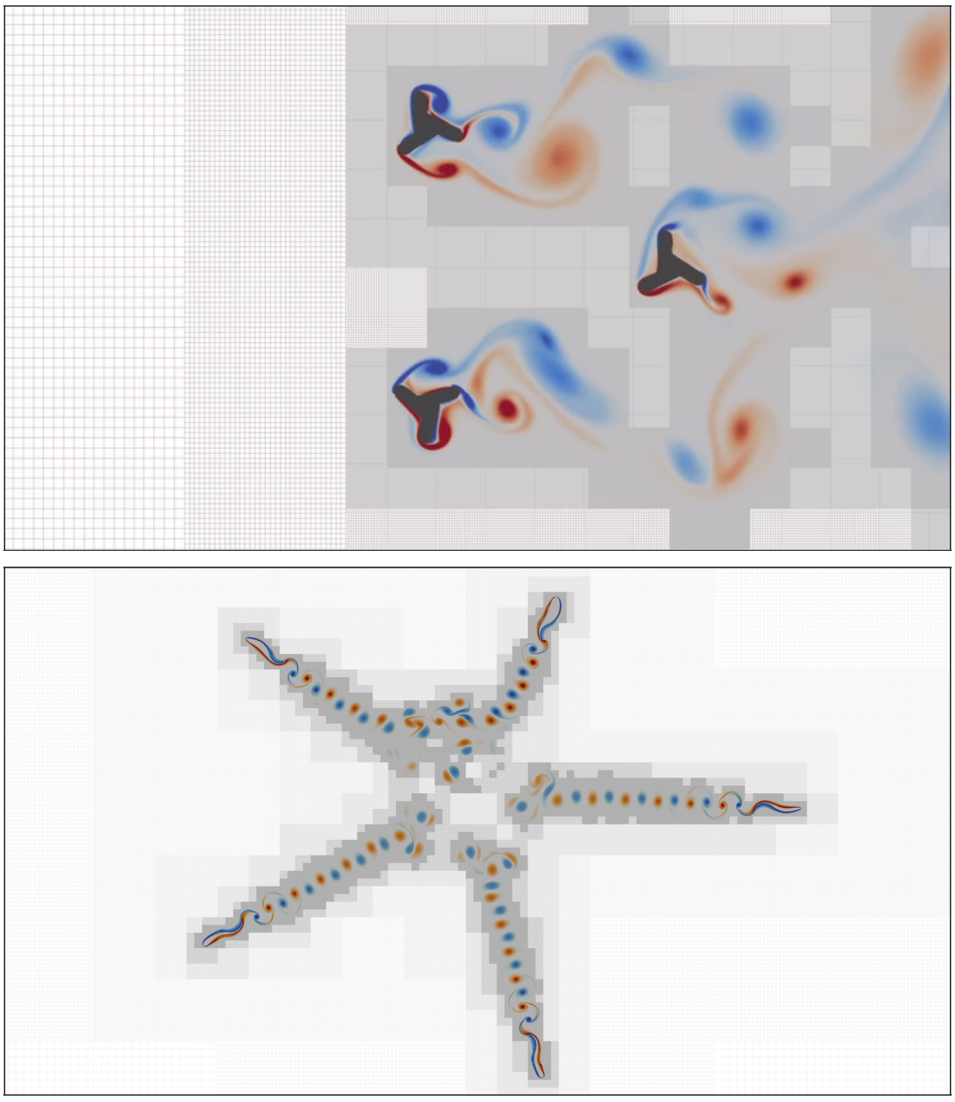


Fig. 15. Vorticity field and the adaptive grid in an array of three wind turbines at $Re = 300$ (top) and diverging fish school at $Re = 5000$ (bottom).

The software is built upon a performance-aware design and a set of abstraction layers that facilitates rapid prototyping of new simulations. The presented software can be downloaded from GitHub, https://github.com/cselab/MRAG-I2D. We are currently testing MRAG-I3D for the 3D multiresolution flow simulations on heterogeneous, massively parallel computer architectures.

Acknowledgements

The authors would like thank Manfred Quack and Roman Schaerer for their contributions to the software development of MRAG-I2, the Swiss National Science Foundation (SNSF), the Brutus cluster team of ETH Zurich as well as the user support team at the Swiss Supercomputing Center (CSCS), project s500.

Appendix A. Ghosts, LTS and CFL-particles

In this section we provide additional diagrams in order to better elucidate the construction of ghosts, the LTS, and the multiresolution CFL-particles.

The sketch in Fig. 16 illustrates an example of a ghost reconstruction in order to provide a uniform resolution frame (of width 1) around the grid point coarse c^0_{-1}. A detailed description of about ghost reconstruction can be found in [22]. The sketch in Fig. 17 illustrates the gain provided by a first-order LTS. The example considers a grid consisting of gridpoints of three different resolution levels. The details about the employed LTS can be found in [20].

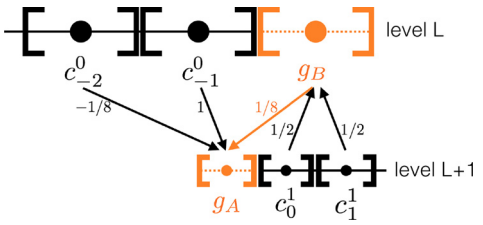


Fig. 16. Reconstruction of the ghost g_A, across a resolution jump of one for the case of third-order average interpolating wavelets. The arrows, and their associated weights, denote contributions from the grid points to the ghosts g_A and g_B. The secondary ghost g_B is needed to evaluate g_A but does not depend on g_A when using average-interpolating wavelets (no loop in the dependency graph).

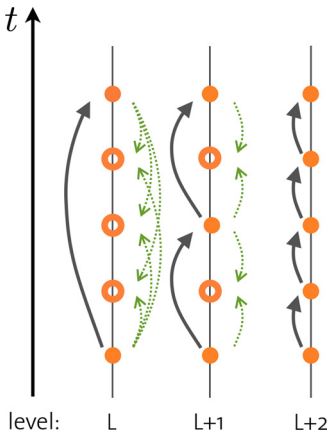


Fig. 17. LTS example for three grid points at different levels of resolution for an advection problem. Every black arrow requires one evaluation of the right hand side and one integration step to advance the solution (solid orange circles). Green dashed arrows denote the time interpolation of missing data (empty orange circles) necessary for evaluating the RHS on finer levels. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

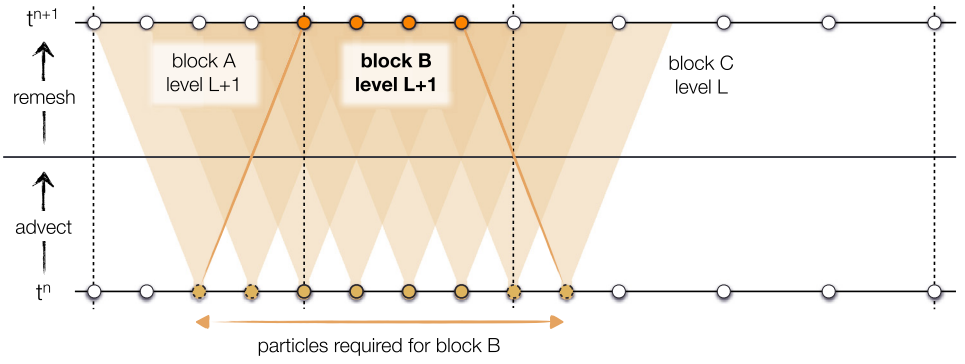


Fig. 18. Spatiotemporal range of the CFL-particles for one advection step.

Fig. 18 details the advection and remeshing step using CFL-particles on multiresolution block-structured grids in a 1D example. During the particle advection stage, assuming $CFL < 1$, a particle may move at most one grid point to the left or to the right of its original position. During the remeshing stage, assuming here an interpolation kernel with support of two grid points, the particle contributes to the grid point on its left and the one on its right. In order to capture all contributions to the interior grid points of block B, we therefore need to initialize particles at tn inside the block plus a ghost region of two grid points on either side. These ghost points are created using wavelet-based reconstruction, resulting in an extended set of uniformly spaced particles. After advection and remeshing, only the particles' contributions to the interior grid points of block B are kept. This approach is followed for each block and allows us to consistently deal with jumps in resolution between neighboring blocks. It also enables the independent, parallel processing of each block after the reconstruction of their respective ghost points.

References

[1] P. Koumoutsakos, A. Leonard, High-resolution simulations of the flow around an impulsively started cylinder using vortex methods, J. Fluid Mech. 296 (1995) 1–38.
[2] P. Koumoutsakos, Inviscid axisymmetrization of an elliptical vortex, J. Comput. Phys. 138 (2) (1997) 821–857.
[3] G.H. Cottet, P. Koumoutsakos, M.L.O. Salihi, Vortex methods with spatially varying cores, J. Comput. Phys. 162 (1) (2000) 164–185.
[4] M. Bergdorf, G. Cottet, P. Koumoutsakos, Multilevel adaptive particle methods for convection–diffusion equations, Multiscale Model. Simul. 4 (1) (2005) 328–357.
[5] A.S. Almgren, J.B. Bell, P. Colella, L.H. Howell, M.L. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, J. Comput. Phys. 142 (1) (1998) 1–46.
[6] M. Bergdorf, P. Koumoutsakos, A lagrangian particle–wavelet method, Multiscale Model. Simul. 5 (3) (2006) 980–995.
[7] K. Schneider, O. Vasilyev, Wavelet methods in computational fluid dynamics, Annu. Rev. Fluid Mech. 42 (1) (2010) 473–503.
[8] N. Kevlahan, O. Vasilyev, An adaptive wavelet collocation method for fluid–structure interaction at high Reynolds numbers, SIAM J. Sci. Comput. 26 (6) (2005) 1894–1915.
[9] O. Vasilyev, N. Kevlahan, Hybrid wavelet collocation – Brinkman penalization method for complex geometry flows, Int. J. Numer. Methods Fluids 40 (3–4) (2002) 531–538.
[10] K. Schneider, M. Farge, Adaptive wavelet simulation of a flow around an impulsively started cylinder using penalisation, Appl. Comput. Harmon. Anal. 12 (3) (2002) 374–380.
[11] E. Deriaz, V. Perrier, Divergence-free and curl-free wavelets in two dimensions and three dimensions: application to turbulent flows, J. Turbul. (7) (2006).
[12] J.M. Alam, N.K.R. Kevlahan, O.V. Vasilyev, Simultaneous space–time adaptive wavelet solution of nonlinear parabolic differential equations, J. Comput. Phys. 214 (2) (2006) 829–857.
[13] M. Domingues, S. Gomes, O. Roussel, K. Schneider, An adaptive multiresolution scheme with local time stepping for evolutionary PDEs, J. Comput. Phys. 227 (2008) 3758–3780.
[14] D.F. Martin, P. Colella, A cell-centered adaptive projection method for the incompressible Euler equations, J. Comput. Phys. 163 (2) (2000) 271–312.
[15] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534.
[16] A. Cohen, Adaptive methods for PDE's: wavelets or mesh refinement? preprint arXiv:math/0212414, 2002.
[17] D.L. Donoho, Interpolating wavelet transforms, Department of Statistics, Stanford University 2 (3), 1992, preprint.
[18] E. Bacry, S. Mallat, G. Papanicolaou, A wavelet based space–time adaptive numerical method for partial differential equations, Modél. Math. Anal. Numér. 26 (7) (1992) 793–834.
[19] D. Rossinelli, M. Bergdorf, B. Hejazialhosseini, P. Koumoutsakos, Wavelet-based adaptive solvers on multi-core architectures for the simulation of complex systems, in: Euro-Par '09: Proceedings of the 15th International Euro-Par Conference on Parallel Processing, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 721–734.
[20] B. Hejazialhosseini, D. Rossinelli, M. Bergdorf, P. Koumoutsakos, High order finite volume methods on wavelet-adapted grids with local time-stepping on multicore architectures for the simulation of shock–bubble interactions, J. Comput. Phys. 229 (22) (2010) 8364–8383.
[21] D. Rossinelli, B. Hejazialhosseini, M. Bergdorf, P. Koumoutsakos, Wavelet-adaptive solvers on multi-core architectures for the simulation of complex systems, Concurr. Comput.: Pract. Exper. 23 (2) (2011) 172–186.
[22] D. Rossinelli, B. Hejazialhosseini, D. Spampinato, P. Koumoutsakos, Multicore/multi-GPU accelerated simulations of multiphase compressible flows using wavelet adapted grids, SIAM J. Sci. Comput. 33 (2) (2011) 512–540.
[23] D. Rossinelli, Multiresolution flow simulations on multi/many-core architectures, PhD thesis, ETH Zurich, 2011, http://dx.doi.org/10.3929/ethz-a-006483930.
[24] M. Gazzola, C. Mimeau, A. Tchieu, P. Koumoutsakos, Flow mediated interactions between two cylinders at finite re numbers, Phys. Fluids 24 (4) (2012) 043103.
[25] W.M. van Rees, M. Gazzola, P. Koumoutsakos, Optimal shapes for anguilliform swimmers at intermediate Reynolds numbers, J. Fluid Mech. 722 (5) (2013).
[26] M. Gazzola, B. Hejazialhosseini, P. Koumoutsakos, Reinforcement learning and wavelet adapted vortex methods for simulations of self-propelled swimmers, SIAM J. Sci. Comput. 36 (3) (2014) B622–B639, http://dx.doi.org/10.1137/130943078.
[27] E. Arquis, J. Caltagirone, On the hydrodynamical boundary-conditions along a fluid layer porous-medium interface – application to the case of free convection, C. R. Acad. Sci., Ser. II 299 (1) (1984) 1–4.
[28] P. Angot, C. Bruneau, P. Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, Numer. Math. 81 (4) (1999) 497–520.
[29] G. Carbou, P. Fabrie, Boundary layer for a penalization method for viscous incompressible flow, Adv. Differ. Equ. 8 (2003) 1453–1480.
[30] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.
[31] N. Patankar, N. Sharma, A fast projection scheme for the direct numerical simulation of rigid particulate flows, Commun. Numer. Methods Eng. 21 (8) (2005) 419–432.
[32] M. Coquerelle, G. Cottet, A vortex level set method for the two-way coupling of an incompressible fluid with colliding rigid bodies, J. Comput. Phys. 227 (21) (2008) 9121–9137.
[33] M. Gazzola, W.M. van Rees, P. Koumoutsakos, C-start: optimal start of larval fish, J. Fluid Mech. 698 (2012) 5–18.
[34] A. Cohen, I. Daubechies, J. Feauveau, Biorthogonal bases of compactly supported wavelets, Commun. Pure Appl. Math. 45 (5) (1992) 485–560.
[35] D. Rossinelli, B. Hejazialhosseini, M. Bergdorf, P. Koumoutsakos, Wavelet-adaptive solvers on multi-core architectures for the simulation of complex systems, Concurr. Comput.: Pract. Exper. 23 (2) (2011) 172–186.
[36] M. Domingues, S. Gomes, O. Roussel, K. Schneider, Space–time adaptive multiresolution methods for hyperbolic conservation laws: applications to compressible Euler equations, Appl. Numer. Math. 59 (9) (2009) 2303–2321.
[37] O. Roussel, K. Schneider, A. Tsigulin, H. Bockhorn, A conservative fully adaptive multiresolution algorithm for parabolic PDEs, J. Comput. Phys. 188 (2) (2003) 493–523.
[38] G. Cottet, P. Poncet, Advances in direct numerical simulations of 3d wall-bounded flows by vortex-in-cell methods, J. Comput. Phys. 193 (1) (2004) 136–158.
[39] P. Ploumhans, G. Winckelmans, Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry, J. Comput. Phys. 165 (2) (2000) 354–406.
[40] G. Cottet, P. Koumoutsakos, Vortex Methods, Theory and Practice, Cambridge University Press, 2000.
[41] J.J. Monaghan, Extrapolating *b* splines for interpolation, J. Comput. Phys. 60 (2) (1985) 253–262.
[42] S. Williams, A. Waterman, D. Patterson, Roofline: an insightful visual performance model for multicore architectures, Commun. ACM 52 (4) (April 2009) 65–76.
[43] B. Hejazialhosseini, D. Rossinelli, C. Conti, P. Koumoutsakos, High throughput software for direct numerical simulations of compressible two-phase flows, in: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, 2012, pp. 16:1–16:12.

[44] R. Blumofe, C. Joerg, B. Kuszmaul, C. Leiserson, K. Randall, Y. Zhou, Cilk – an efficient multithreaded runtime system, J. Parallel Distrib. Comput. 30 (8) (1995) 207–216.
[45] R.D. Blumofe, C.E. Leiserson, Scheduling multithreaded computations by work stealing, J. ACM 46 (5) (1999) 720–748.
[46] J. Barnes, P. Hut, A hierarchical o(N-log-N) force-calculation algorithm, Nature 324 (6096) (1986) 446–449.
[47] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data, SIAM J. Sci. Comput. 14 (6) (1993) 1368–1393.
[48] W. Hackbusch, Multi-grid Methods and Applications, vol. 4, Springer-Verlag, Berlin, 1985.
[49] W.M. van Rees, B. Hejazialhosseini, D. Rossinelli, P. Hadjidoukas, P. Koumoutsakos, High performance CPU/GPU multiresolution Poisson solver, Adv. Parallel Comput. 163 (2014) 481–490.
[50] O.A.R. Board, OpenMP application program interface, Technical report, OpenMP Architecture Review Board, 2008.
[51] A. Robison, M. Voss, A. Kukanov, Optimization via reflection on work stealing in TBB, IEEE International Symposium on Parallel and Distributed Processing 1 (8) (2008) 598–605.
[52] G. Contreras, M. Martonosi, Characterizing and improving the performance of intel threading building blocks, in: IEEE International Symposium on Workload Characterization (IISWC), 2008, pp. 57–66.
[53] W. Schroeder, K. Martin, B. Lorensen, An Object-Oriented Approach to 3D Graphics, Prentice Hall, 1997.
[54] D. Rossinelli, M. Bergdorf, G. Cottet, P. Koumoutsakos, GPU accelerated simulations of bluff body flows using vortex particle methods, J. Comput. Phys. 229 (9) (2010) 3316–3333.
[55] S. Kern, P. Koumoutsakos, Simulations of optimized anguilliform swimming, J. Exp. Biol. 209 (24) (2006) 4841–4857.
[56] M. Gazzola, P. Chatelain, W. van Rees, P. Koumoutsakos, Simulations of single and multiple swimmers with non-divergence free deforming geometries, J. Comput. Phys. 230 (19) (2011) 7093–7114.