International Journal of Multiphase Flow 140 (2021) 103615

Contents lists available at ScienceDirect

International Journal of Multiphase Flow

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

An analysis of the performance enhancement with adaptive mesh refinement for spray problems

Chia-Wei Kuo, Mario F. Trujillo*

Department of Mechanical Engineering, University of Wisconsin-Madison, 1513 University Av. Madison WI 53706 United States

ARTICLE INFO

Article history:
Received 23 June 2020
Revised 20 November 2020
Accepted 23 February 2021
Available online 6 April 2021

Keywords:
Adaptive mesh refinement
Atomization
Spray problem
Openfoam
Volume-of-Fluid

ABSTRACT

Adaptive mesh refinement (AMR) provides an attractive means of significantly reducing computational costs while simultaneously maintaining a high degree of fidelity in regions of the domain requiring it. In the present work, an analysis of the performance of AMR supported by simulations is undertaken for liquid injection and spray formation problems. These problems are particularly challenging from a computational cost perspective since the associated interfacial area typically grows by orders of magnitude, leading to similar growth in the number of highly refined cells. While this increase in cell numbers directly contributes to a declining performance for AMR, a second less obvious factor is the decaying trend for the cell-based speedup, Θ. A theoretical analysis is presented, leading to a closed-form estimate for this cell-based speedup, namely $\Theta_E = \sqrt{\kappa_{F,SM}}/\sqrt{\kappa_{F,AMR}}$, where κ_F is the Frobenius condition number, and SM corresponds to a static mesh case. It is shown that for spray formation problems, the typical growth in $\kappa_{F,AMR}$ is more pronounced than $\kappa_{F,SM}$ causing a decline in Θ and consequently diminishing the AMR performance. Additional contributing sources are also examined, which include the role of load balancing and the choice of linear solvers for the Poisson system.

1. Introduction

The use of adaptive mesh refinement (AMR) is widely accepted as a computationally efficient way to solve various types of partial differential equations (Harten and Hyman, 1983; Berger and Oliger, 1984; Berger and Colella, 1989; Miller and Miller, 1981; Adjerid and Flaherty, 1986; Brandt, 1977; Babuvška and Rheinboldt, 1978; Verfürth, 1994; Dörfler, 1996; Riviere et al., 1999; Hornung and Trangenstein, 1997; DeZeeuw and Powell, 1993) by dynamically allocating a high level of numerical fidelity in areas requiring it. A common way to achieve this is to reduce the local grid spacing (Δx) dynamically in regions of high-fidelity demand, which can be identified through the use of a cost function. Some examples of these cost functions include local truncation errors based on Richardson extrapolation (Berger and Oliger, 1984; Berger and Colella, 1989), magnitude of the velocity gradient (Miller and Miller, 1981), a fraction of the maximum total velocity difference (DeZeeuw and Powell, 1993), smearing of a shock discontinuity (Harten and Hyman, 1983), and jumps in flux variables (Adjerid and Flaherty, 1986; Hornung and Trangenstein, 1997). Besides its abundant application in Computational Fluid Dynamics (CFD), AMR has also been successfully applied in cosmological hydrodynamics (Teyssier, 2002), magnetohydrodynamics (Balsara, 2001), and strain localization problems (Ortiz and Quigley, 1991) among other fields.

For two-phase flow problems, the interface is the most obvious choice for the cost function since good numerical accuracy is required in its advection and the calculation of its dynamics, e.g., surface tension. For instance, in the work of Theodorakakos and Bergeles (2004), the tagging of the interfacial region is achieved by monitoring the domain where the liquid fraction varies between 0.2 and 0.8. In the paper by Malik et al. (2007), the tagging is instead executed where the normalized curvature (product of curvature and the cell size) is below a specified constant that is typically 0.2 (Fuster et al., 2009). Common standard problems that are employed in evaluating the degree of AMR acceleration include rotation of Zalesak sphere (Anjos et al., 2014; Laurmaa et al., 2016), droplet deformation in the 3D vortical flow (Anjos et al., 2014; Chen and Yang, 2014), Rayleigh-Taylor instability (Zuzio and Estivalezes, 2011; Xie et al., 2014), and the secondary breakup of a droplet (Strotos et al., 2016; Yang et al., 2016; Tavangar et al., 2015; Jain et al., 2015). Among these works, the investigations by Laurmaa et al. (2016), and Zuzio and Estivalezes (2011) further demonstrate that the smaller the value of the minimum grid size, $\Delta x_{\min}$, the better the odds of attaining a higher AMR speedup. Overall in all of these cases, significant benefits are reported with the use of AMR.

* Corresponding author.
E-mail address: mtrujillo@wisc.edu (M.F. Trujillo).

https://doi.org/10.1016/j.ijmultiphaseflow.2021.103615
0301-9322/

Considering the case of liquid jet injection and spray formation, an inspection of the literature (Tonini et al., 2008; Xue and Kong, 2009; Lebas et al., 2009; Fuster et al., 2009; Herrmann, 2010; Shinjo and Umemura, 2010; Jiang et al., 2010; Li and Soteriou, 2012; Chen et al., 2013; Fuster et al., 2013; Jarrahbashi and Sirignano, 2014; Ling et al., 2015; Arienti and Sussman, 2015) reveals that AMR is not overwhelmingly used. From the studies that do employ AMR (Tonini et al., 2008; Xue and Kong, 2009; Fuster et al., 2009; Li and Soteriou, 2012; Chen et al., 2013; Fuster et al., 2013; Arienti and Sussman, 2015), no comparisons are performed against a well-designed static refined mesh, e.g., a static mesh (SM) that is octree refined only in the region where the interface is expected to be present within the computational time period. For liquid injection problems, this region can be reasonably estimated apriori. Usually, in the literature, AMR versus SM comparisons are performed for situations where the SM is refined at the maximum level everywhere in the domain, even in regions far from the interface. Under such conditions, almost regardless of the performance of AMR, it is expected that using AMR will provide significant computational cost savings.

In relatively recent conference presentations (Kuo and Trujillo, 2019; 2018), the present authors have provided empirical evidence indicating a declining trend in AMR performance with time for liquid injection simulations. This behavior was attributed to a substantial growth in the interfacial area, which resulted in the respective creation of a large number of refined cells in AMR. Hence, with this substantial growth of cells, the size of the system being solved increases, and thus it comes as no surprise that a significant slow down in speed is observed. However, beyond the empirical evidence, we did not provide any underlying arguments supported by theoretical developments to explain the trends observed, so the present work endeavors to remedy the situation by providing such analysis. Because the cost of computations is primarily driven by the solution of the Pressure Poisson Equation, particularly for CFD problems (Löhner et al., 2011; Jiang and Lai, 2016; Kwak and Kiris, 2010; Johnston and Liu, 2002), the present work focuses on this aspect. Regardless, for every computational result shown, the dominating costs of the Poisson solution are confirmed. Specifically, the emphasis of the present work is on the changing linear algebra characteristics of the Poisson system, as AMR unfolds. While different advanced variations of AMR algorithm and adaptations are currently in use and development, in the present work our particular algorithm employs the standard block-based AMR library of OpenFOAM 2.1.1.

The contents of the paper are as follows. An analysis of the linear system solution for the Poisson system is detailed in Section 2, where the main results from this analysis form the theoretical foundation for interpreting the subsequent computations. An overview of the solver is provided in Section 3. The computations first consider a standard two-phase problem consisting of a standing wave in Section 4.1. This is followed by progressively more complicated problems, including liquid injection in a small domain in Section 4.2 and liquid injection in a larger domain in Section 4.3, where the latter case reflects the size of computations often found in engineering and scientific studies of sprays. Towards the end of the paper in Section 4.4, we examine other potential contributors to the recorded AMR slowdown, including load balancing or choice of the linear solver to evaluate whether they can help explain the trends recorded. Finally, a summary of the work, along with concluding thoughts, are communicated in Section 5.

2. Analysis of AMR performance

The general Poisson linear system to be analyzed is of the form

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where A is a real $N \times N$ symmetric matrix (it can be shown that under finite volume discretization, the resulting Poisson system is symmetric). A popular procedure for tackling these types of systems is through the use of preconditioned conjugate gradient methods (Quan et al., 2009; Anjos et al., 2014; Theodorakakos and Bergeles, 2004; Liu, 2013; Moayedi et al., 2017). Let P^{-1} denote the preconditioner, then the system to be solved is

$$P^{-1}A\mathbf{x} = B\mathbf{x} = P^{-1}\mathbf{b}. \tag{2}$$

For large systems, a popular family of preconditioners is the incomplete Cholesky (IC) factorization (Ascher and Greif, 2011). The overall goal is to ensure that the system, $B\mathbf{x} = P^{-1}\mathbf{b}$, has much better clustering of the eigenvalues when compare to A resulting in significant improvements in convergence.

Defining the iteration index as k, the exact solution to Eq. (1) as $\mathbf{x}$, the k^{th} iterative solution as $\mathbf{x}_k$, the associated error as $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}$, the energy norm for this error can then be defined as

$$||\mathbf{e}_k||_B = \sqrt{(\mathbf{e}_k)^T(B)\mathbf{e}_k}. \tag{3}$$

The corresponding error bound is given as a theorem in Ascher and Greif (2011, pg. 187) and in LeVeque (2007, pg. 93) as

$$\frac{||\mathbf{e}_k||_B}{||\mathbf{e}_0||_B} \leq 2\left(\frac{\sqrt{\kappa_2(B)}-1}{\sqrt{\kappa_2(B)}+1}\right)^k. \tag{4}$$

In this expression, the condition number κ_2 is given as λ_1/λ_N, i.e., the ratio of largest to the smallest eigenvalue of the symmetric positive definite matrix B.

An issue with the use of κ_2 is that while it can be readily computed, an order of magnitude estimate of its value based on the size of the system, N, is not automatically apparent. As an alternative, we choose to work with a quantity that is explicitly related to the size of the problem, i.e., the Frobenius condition number, κ_F, defined as (Chehab and Raydan, 2008)

$$\begin{aligned} &\kappa_F(B) = N + f_B(N), \text{ where} \\ &f_B(N) = \frac{1}{2}\left[||B-(B)^{-1}||_F^2 - \left(||B||_F - ||(B)^{-1}||_F\right)^2\right], \end{aligned} \tag{5}$$

where the Frobenius norm is defined as

$$||B||_F^2 = Tr[(B)^T(B)]. \tag{6}$$

Here Tr is the trace operation. It follows from the Frobenius condition number that $\kappa_F(B) > \kappa_2(B)$ (Chehab and Raydan, 2008, pg. 2091) and (Trefethen and Bau, 1997)[Theorem 5.3].

In what follows, it is tacitly assume that the condition numbers are based on B, i.e. $\kappa_F = \kappa_F(B)$ and $\kappa_2 = \kappa_2(B)$. Returning to the upper bound for the error in Eq. (4), we can expand the following quantity as

$$\begin{aligned} \frac{\sqrt{\kappa_2}-1}{\sqrt{\kappa_2}+1} &= \left(1 - \frac{2}{\sqrt{\kappa_2}+1}\right) \leq \left(1 - \frac{2}{\sqrt{\kappa_F}+1}\right) \\ &= \frac{\sqrt{\kappa_F}-1}{\sqrt{\kappa_F}+1} (\text{since } \kappa_F > \kappa_2). \end{aligned} \tag{7}$$

Introducing this into Eq. (4) gives

$$\frac{||\mathbf{e}_k||_B}{||\mathbf{e}_0||_B} \leq 2\left(\frac{\sqrt{\kappa_F}-1}{\sqrt{\kappa_F}+1}\right)^k = 2\left(1 - \frac{2}{\sqrt{\kappa_F}+1}\right)^k. \tag{8}$$

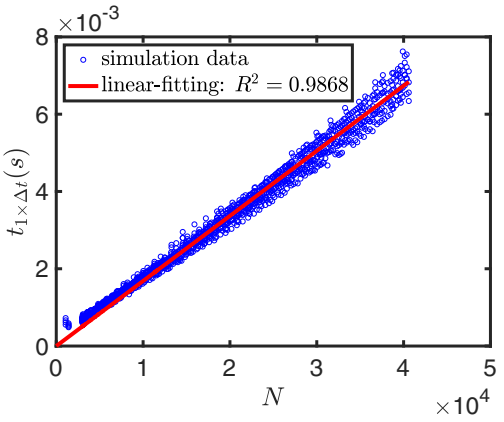

Fig. 1. Computation time for one iteration of linear solve (Poisson system) versus number of computational cells, N, corresponding to the small domain case discussed below in Section 4.2.

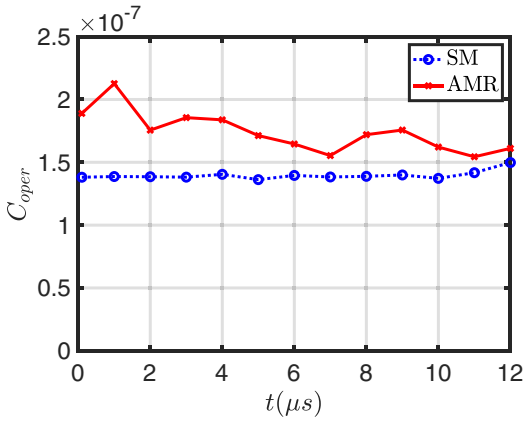

Fig. 2. Time traces of C_{oper} of AMR contrasted with SM for the small domain case elaborated in Section 4.2.

Manipulating this expression yields

$$\begin{aligned}\log\left(\frac{\|\mathbf{e}_k\|_B}{\|\mathbf{e}_0\|_B}\right) &\le \log\left[2\left(1-\frac{2}{\sqrt{\kappa_F}+1}\right)^k\right]\\ &= \log 2 + k\log\left(1-\frac{2}{\sqrt{\kappa_F}+1}\right)\\ &= \log 2 - k\left[\frac{2}{\sqrt{\kappa_F}+1}+\frac{1}{2}\left(\frac{2}{\sqrt{\kappa_F}+1}\right)^2+\mathcal{O}\left(\frac{2}{\sqrt{\kappa_F}+1}\right)^3\right]\\ &\quad(\text{ log expansion [53] pg.126})\\ \log\left(\frac{\|\mathbf{e}_k\|_B}{\|\mathbf{e}_0\|_B}\right) &\le \log 2 - \frac{2k}{\sqrt{\kappa_F}+1}\,(\text{since}\,\kappa_F \gg 1).\end{aligned} \tag{9}$$

Solving for k, we have

$$k \le \frac{1}{2}(\sqrt{\kappa_F}+1)\log\left(2\frac{||\mathbf{e}_0||_B}{||\mathbf{e}_k||_B}\right). \tag{10}$$

The expression on the right of this inequality is the upper bound on the iteration number required to reduce the error for the linear solve from the initial error $||\mathbf{e}_0||_B$ to $||\mathbf{e}_k||_B$.

In the present CFD solution procedure, at each physical time step, the solution of momentum is performed along with the solution for the advection of the gas-liquid interface through a Volume-of-Fluid (VoF) formulation. During this step, the Poisson system is solved in an iterative fashion, where a predetermined error tolerance is assigned. This error tolerance is directly related to $||\mathbf{e}_k||_B/||\mathbf{e}_0||_B$. To estimate the computational cost of this solution, let the upper bound on the number of iterations to achieve the specified tolerance be denoted as n_{iter}. Then the associated computational cost for the Poisson solution is

$$t^{\Delta t} = n_{iter} t_{1\times\Delta t}, \tag{11}$$

where $t_{1\times\Delta t}$ is the computation time per one iteration of the linear solve. Here $t_{1\times\Delta t}$ is proportional to the number of floating-point operations, and the number of operations is proportional to N (Saad, 2003, pg. 281). Both of these statements lead to

$$t^{\Delta t} = n_{iter} C_{oper} N, \tag{12}$$

where C_{oper} is a constant of proportionality. In our own computations, we have confirmed this assertion of $t^{\Delta t}$ being linearly related to N as shown in Fig 1. While values for C_{oper} for either AMR and SM are similar, these constants are not exactly the same as illustrated in Fig. 2. However, for the sake of providing an estimate for the computational efficiency, we will assume they are equal.

Combining Eqs. (10) and (12) by letting n_{iter} be the upper bound on the iterations required for the reduction of $||\mathbf{e}_0||_B$ to $||\mathbf{e}_{n_{iter}}||_B$, we obtain an *upperbound estimate for the computational cost for the Poisson system* as

$$t_U^{\Delta t} = \frac{1}{2}(\sqrt{\kappa_F}+1)\log\left(2\frac{||\mathbf{e}_0||_B}{||\mathbf{e}_{n_{iter}}||_B}\right)C_{oper}N. \tag{13}$$

This expression clearly shows that this upper bound estimate is proportional to the size of the solution vector, i.e., through N, and directly related to the square root of the associated Frobenius condition number for the matrix B.

In comparing the performance of calculations using the SM and AMR strategies, the most straightforward metric is the ratio of respective computation times, namely $t_{SM}^{\Delta t}/t_{AMR}^{\Delta t}$ or simply the *speedup*. Another metric that has been employed in multiphase flow work (Fuster et al., 2009; Zuzio and Estivalezes, 2011; Akhtar and Kleis, 2013) is the ratio of computational time per computational cells, namely

$$\Theta = \frac{t_{SM}^{\Delta t}/N_{SM}}{t_{AMR}^{\Delta t}/N_{AMR}}. \tag{14}$$

This quantity, Θ, represents a comparison of the computational efficiency between the SM and AMR strategies and is referred to in the present work as the *cell-based speedup* between SM and AMR. Using this metric, we can express the computational time ratio as

$$\frac{t_{SM}^{\Delta t}}{t_{AMR}^{\Delta t}} = \Theta\frac{N_{SM}}{N_{AMR}}. \tag{15}$$

If we consider an ideal case, for instance $\Theta = 1$, where the computational time per cell is the same between AMR and SM, then Eq. (15) states that $t_{SM}^{\Delta t}/t_{AMR}^{\Delta t} = N_{SM}/N_{AMR}$. This means that the speedup is directly related to the ratio of computational cells used in SM over AMR, and since it is generally understood that $N_{SM} \gg N_{AMR}$, this implies an expected large value for the speedup. This is, in essence, the *promise of AMR*, that with a significantly reduced number of computational cells, we can proportionally achieve a significant speedup.

Based on the upper bound expression provided in Eq. (13), we can arrive at an estimate for the speedup given by

$$\frac{t_{SM}^{\Delta t}}{t_{AMR\,E}^{\Delta t}} = \frac{(\sqrt{\kappa_{F,SM}}+1)\log\left(2\frac{||\mathbf{e}_0||_{B,SM}}{||\mathbf{e}_{n_{iter}}||_{B,SM}}\right)N_{SM}}{(\sqrt{\kappa_{F,AMR}}+1)\log\left(2\frac{||\mathbf{e}_0||_{B,AMR}}{||\mathbf{e}_{n_{iter}}||_{B,AMR}}\right)N_{AMR}}. \tag{16}$$

Since this is an estimate for the speedup based on upper bounds for $t_{SM}^{\Delta t}$ and $t_{AMR}^{\Delta t}$, it is denoted with subscript E. The ratio of initial ($||\mathbf{e}_0||_B$) and final ($||\mathbf{e}_{n_{iter}}||_B$) error differs usually by orders of magnitude for both SM and AMR. If we can assume that the difference between initial error levels between SM ($||\mathbf{e}_0||_{B,SM}$)

and AMR ($||\mathbf{e}_0||_{B,AMR}$) is , $\mathcal{O}(1)$ the expression above can be approximated by

$$\frac{t_{SM}^{\Delta t}}{t_{AMR\,E}^{\Delta t}} \cong \frac{\sqrt{\kappa_{F,SM}}}{\sqrt{\kappa_{F,AMR}}}\frac{N_{SM}}{N_{AMR}} = \frac{\sqrt{N_{SM}+f_B(N_{SM})}}{\sqrt{N_{AMR}+f_B(N_{AMR})}} \times \frac{N_{SM}}{N_{AMR}}, \tag{17}$$

where we have made use of the fact that $\sqrt{\kappa_{F,SM}} \gg 1$ and $\sqrt{\kappa_{F,AMR}} \gg 1$. In comparing this equation to Eq. (15), we see that Θ is not necessarily a constant, especially for the AMR case where N_{AMR} can vary considerably in time. This expression also highlights the fact that the speedup is not simply related to N_{SM}/N_{AMR} as we shall verify in Section 4. Furthermore, based on this development the estimate for cell-based speedup becomes

$$\Theta_E \cong \frac{\sqrt{\kappa_{F,SM}}}{\sqrt{\kappa_{F,AMR}}} = \frac{\sqrt{N_{SM}+f_B(N_{SM})}}{\sqrt{N_{AMR}+f_B(N_{AMR})}}. \tag{18}$$

In Section Appendix A, an expression for the Frobenius condition number is derived in terms of the eigenvalues of the matrix B as shown in Eq. (42). This yields the following expression

$$\Theta_E \cong \frac{\sqrt{\kappa_{F,SM}}}{\sqrt{\kappa_{F,AMR}}} = \frac{(\sum_{i=1}^{N_{SM}} \lambda_{i,SM}^2)^{1/4}(\sum_{i=1}^{N_{SM}} \lambda_{i,SM}^{-2})^{1/4}}{(\sum_{i=1}^{N_{AMR}} \lambda_{i,AMR}^2)^{1/4}(\sum_{i=1}^{N_{AMR}} \lambda_{i,AMR}^{-2})^{1/4}}, \tag{19}$$

where $\lambda_{i,SM}$ and $\lambda_{i,AMR}$ are respectively the eigenvalues of their corresponding B matrices. However, the estimation of the eigenvalues is not straightforward since the underlying coefficients of the matrix B are variable. These coefficients are functions of local density and volume flux, which have drastic temporal changes in a two-phase flow. Hence, a further simplification of Θ_E beyond the point shown in Eq. (19) is difficult. However, there are two trends that are noteworthy. The first is that κ_F increases with time even for the SM case, as shown in the small domain cases discussed in Section 4.2. The second trend is that N_{AMR} increases tremendously in liquid injection and atomization problems. Hence, even if the magnitudes of the eigenvalues remain fairly equal, the overall sum would increase, leading to an overall increase in $\kappa_{F,AMR}$. Since this second contribution is absent in SM, this behavior would favor an increase in AMR that is more pronounced than SM, leading to a decrease in Θ_E. Again this is supported by the results from the liquid injection cases. Prior to the examination of the AMR performance via computation, the following section gives a brief overview of the solver.

3. Numerical solver

To provide a quantitative assessment of the previous analysis, the algebraic Volume-of-Fluid solver, `interFoam`, and its AMR adaption, `interDyMFoam`, is employed to solve the transport equation for the cell-based liquid fraction. The `interFoam`solver forms a part of a larger open-source distribution of computational mechanics solvers and C++ libraries of OpenFOAM version 2.1.1, which is designed for finite volume discretization on collocated grids for the solution of two-phase incompressible flows. The `interFoam` solver has been previously verified and validated in a previous publication from our group (Deshpande et al., 2012). It has also been used to study the physics associated with spray formation and interfacial instabilities (Agarwal and Trujillo, 2020; 2018; Trujillo et al., 2018; Deshpande et al., 2015).

The two-phase solution begins with an interface advection step, which is handled via an algebraic VoF scheme based on a compressive interface capturing methodology advanced by Ubbink and Issa (1999) and Rusche (2003) with contributions from Henry Weller. The two-phase momentum equation is composed of an initial predictor step, which produces a velocity $\mathbf{U}$, which is typically not divergent free. This is followed by a pressure Poisson solution having the following form (Deshpande et al., 2012, pg. 10)

$$\sum_f \left(\frac{1}{A_p}\right)_f (\Delta_f^{\perp} P)|\mathbf{S}_f| = \sum_f \mathbf{U}_f \cdot \mathbf{S}_f, \tag{20}$$

where the summation is taken over all faces, f, of a given computational cell. The interpolation of the predictor velocity to the cell faces is denoted by $\mathbf{U}_f$; the cell face area vector by $\mathbf{S}_f$, and the operator $\Delta_f^{\perp}$ is the gradient normal to each cell face, where P is the pressure. Due to the two-phase flow conditions, the coefficients $(A_p)_f$ are variable (Deshpande et al., 2012, pg. 10). It can be shown that the resulting system is symmetric and that it is positive definite. The Poisson system is solved using a Preconditioned Conjugate Gradient (PCG) method, with Diagonal Incomplete Cholesky (DIC) as the preconditioner. The convergence criterion employed is 10^{-7} based on the normalized residual of the original linear system, and this criterion is used in all cases presented in this paper.

After the Poisson solution, a corrector step follows. Subsequent to this step, a mesh adaptation is performed where the cost function employed is $|\nabla\alpha| \times \Delta x$. Here $|\nabla\alpha|$ is the magnitude of the liquid fraction gradient, and Δx is the local cell size. The selection of this cost function follows published procedures used in two-phase flow AMR cases as documented in Fuster et al. (Fuster et al., 2009, pg. 554). Refinement occurs if

$$|\nabla\alpha| \times \Delta x > \epsilon_1, \tag{21}$$

where ϵ_1 is a user-defined threshold value. Under refinement, the given cell is recursively split into eight smaller child cells following the octree structure shown in Fig. 3. For a N_{level} AMR, where the cell is refined $(N_{level}-1)$ times, the sizes of the child cells, denoted as Δx_c, are given by

$$\Delta x_c = \Delta x_p \big/ 2^{N_{level}-1}. \tag{22}$$

Here Δx_p is the parent cell size. For the sake of narrowing the scope of the present investigation, we mainly consider a three-level AMR ($N_{level} = 3$), that is, only three levels of grid size, Δx_1, Δx_2, and Δx_3 are considered, where $\Delta x_1 = \Delta x_p$, $\Delta x_2 = \Delta x_1/2$ and $\Delta x_3 = \Delta x_2/2$ (although a higher AMR is also available in `interDyMFoam`). Once a refinement has taken place, the flow variable is mapped from the parent cell to the child cells by assigning them the same value,

$$\xi_{c,i} = \xi_p, \;\; i = 1, 2, 3, ., 8, \tag{23}$$

where $\xi_{c,i}$ is the value of the child cell i, and ξ_p is the value of the parent cell. Other schemes for mapping parent-child or child-parent are presented in the literature (Theodorakakos and Bergeles, 2004; Xue and Kong, 2009), which apply to more general grid structures.

Conversely, if any cell satisfies the coarsening criterion given by

$$|\nabla\alpha| \times \Delta x < \epsilon_2, \tag{24}$$

the cluster of the associated cells will be combined to recover the parent cell, as shown in Fig. 3. Here ϵ_2 is also a user-defined threshold with $\epsilon_2 < \epsilon_1$. The flow variable for the parent cell is obtained from

$$\xi_p = \frac{1}{8}\sum_{i=1}^{8} \xi_{c,i}. \tag{25}$$

Overall, Eqs. (23) and (25) apply to all flow variables, namely liquid fraction, velocity, and pressure, which are cell-centered. The values for ϵ_1 and ϵ_2 are adjusted so that the gas-liquid interface is bounded by roughly five cells on either side of the interface. This provides adequate support for the advection operation and allows us to take between three to five global time steps between each

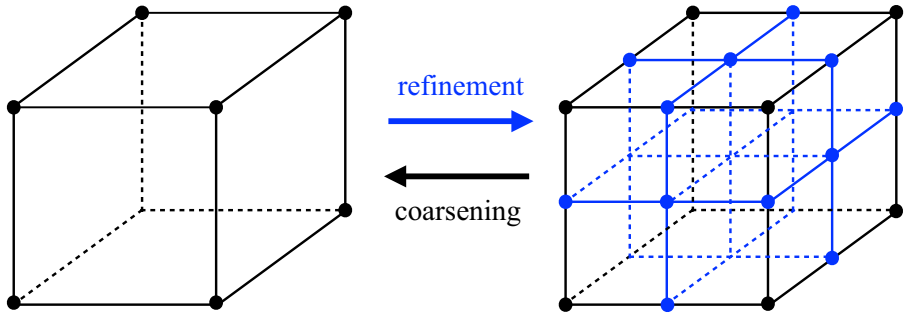

Fig. 3. Mesh operation steps of octree-based AMR employed in the present study (all hexahedral cells): refinement (from left to right) and coarsening (from right to left).

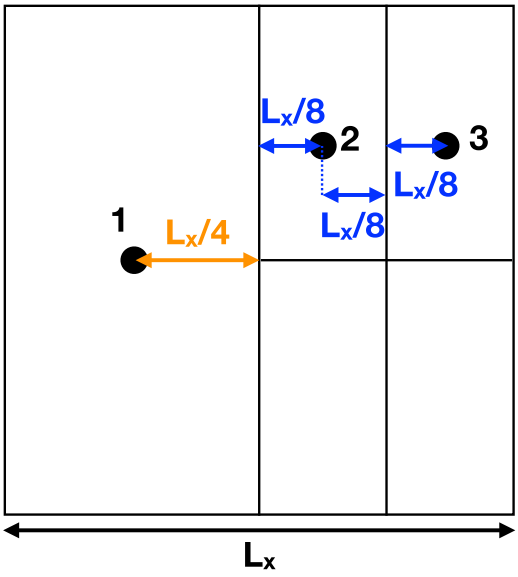

Fig. 4. Examples of grid interfaces for gradient and flux calculations.

mesh adaptation operation. However, towards the end of the computation, the refinement region fills a sizable fraction of the entire domain due to the intensity of the liquid breakup process, and under these conditions, the mesh adaptation frequency can be extended to approximately every ten times steps without ill effect.

In addition to cell-centered quantities being mapped among refined and coarse regions, gradient and flux information are also handled accordingly at the boundaries between coarse and refined regions (grid interface). For instance, the x-component of the gradient of a flow variable ξ at the grid interface is calculated as follows,

$$\begin{aligned}\partial_x \xi_f^{(1,2)} &= \frac{\xi_2 - \xi_1}{L_x/4 + L_x/8},\\ \partial_x \xi_f^{(2,3)} &= \frac{\xi_3 - \xi_2}{L_x/8 + L_x/8},\end{aligned} \tag{26}$$

pertaining to the configuration depicted in Fig. 4. Here ξ_i is the value of ξ at cell i and $\partial_x \xi_f^{(j,k)}$ is the derivative evaluated at the boundary between cells j and k. Likewise, the volume flux between cells j and k, denoted by $\phi_f^{(j,k)}$, is calculated as

$$\begin{aligned}\phi_f^{(1,2)} &= \left[\mathbf{U}_1 \frac{L_x/8}{L_x/4 + L_x/8} + \mathbf{U}_2 \frac{L_x/4}{L_x/4 + L_x/8}\right] \cdot \mathbf{S}_{f_1},\\ \phi_f^{(2,3)} &= \left[\mathbf{U}_2 \frac{L_x/8}{L_x/8 + L_x/8} + \mathbf{U}_3 \frac{L_x/8}{L_x/8 + L_x/8}\right] \cdot \mathbf{S}_{f_2},\end{aligned} \tag{27}$$

where $\mathbf{S}_{f_1}$ is the product of the cell face area and its unit normal outward vector pertaining to faces $f_1 = \{\partial\Omega_1 \cap \Gamma_{\Delta x}^{(1,2)}\}$. Similarly, $\mathbf{S}_{f_2}$ pertains to the faces $f_2 = \{\partial\Omega_2 \cap \Gamma_{\Delta x}^{(2,3)}\}$.

4. Results and discussion

The section begins by computing and analyzing the AMR performance for a standard flow problem that often appears in the multiphase flow literature (Zuzio and Estivalezes, 2011; Popinet, 2009). The purpose of this first exercise is to evaluate whether a similar level of performance is obtained in the current computations.

The next set of exercises consists of liquid injection and spray formation, which is the main focus of the work. In the first set of calculations, a relatively small domain under carefully selected conditions is employed to allow for seamless calculation of all relevant linear algebra metrics, e.g., the Frobenius condition number. The goal here is to provide insights originating from the analysis into the observed drop in AMR speedup. The second set of injection problems are much larger and reflect the type of cases executed in physics and engineering pursuits. The examination is focused on determining whether the pattern observed for the smaller injection cases is similarly observed in the larger spray problems. Since all of this work is based on the conjugate gradient method, in the last section, a multigrid scheme for solving the Poisson system is employed to detect whether similar patterns of performance are observed. Furthermore, the potential issue of load balancing is discussed and supported by additional injection calculations.

4.1. Standing wave cases

An illustration of the standing wave cases is shown in Fig. 5 corresponding to both the AMR and SM strategies. The extent of the domain in the horizontal and vertical direction is respectively 100 μm and 400 μm. The initial conditions for velocity and interface position are

$$\begin{aligned}\mathbf{u}(\mathbf{x}, t = 0) &= \mathbf{0}\\ y_\Gamma(t = 0) &= [200 + 5sin(2\pi x/100)]\mu m.\end{aligned} \tag{28}$$

At the bottom of the domain, a zero Neumann condition for the pressure and the liquid fraction is employed; the no-slip boundary condition is used for the velocity. On the left and right faces, a zero gradient condition is prescribed. On the front and back faces, a periodic condition is employed. At the top of the domain, we use a zero Neumann condition for liquid fraction and velocity and a zero Dirichlet condition for total pressure. The AMR refinement criterion is $|\nabla\alpha| \times \Delta x > 10^{-4}$, while the coarsening criterion is $|\nabla\alpha| \times \Delta x < 0.5 \times 10^{-4}$. Additionally, the physical properties are included in Table 1. Here ρ and μ are mass density and viscosity, respectively, σ is the surface tension coefficient, and subscripts (L, G) denote liquid and gas, respectively.

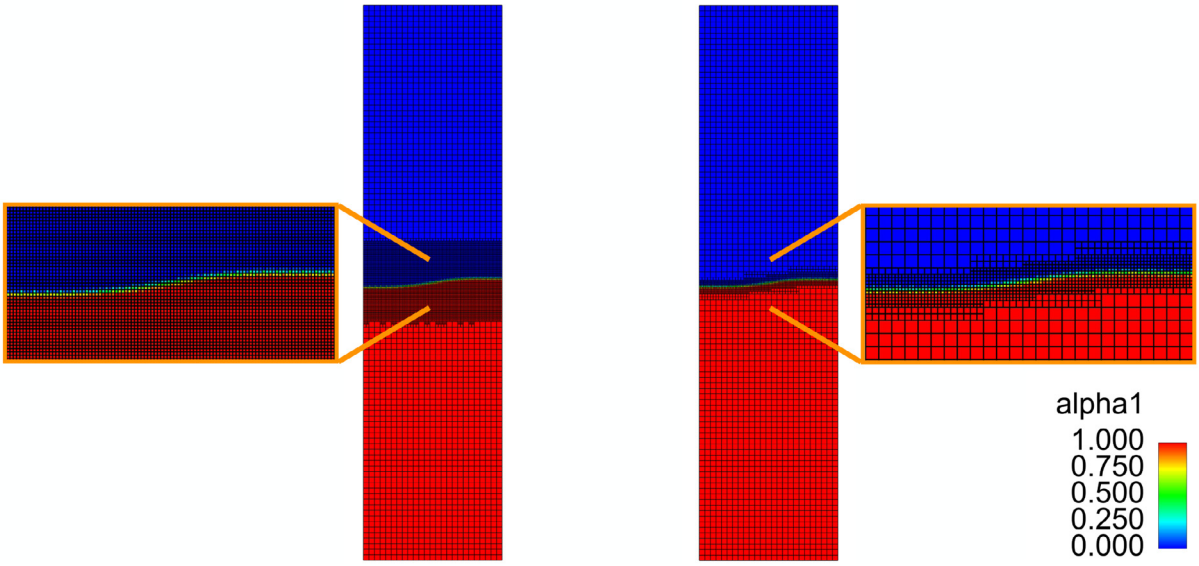

Fig. 5. Illustration of the standing wave problem with SM on the left and AMR on the right. The AMR process results in a much tighter high-resolution band of cells surrounding the interface in comparison to the SM strategy.

Table 1
Physical properties for the standing wave case.

Liquid density	$\rho_L = 997$ kg/m^3
Liquid viscosity	$\mu_L = 8.899\times 10^{-4}$ kg/(m-s)
Gas density	$\rho_G = 1.177$ kg/m^3
Gas viscosity	$\mu_G = 1.846\times 10^{-5}$ kg/(m-s)
Coefficient of surface tension	$\sigma = 0.072$ kg/s^2

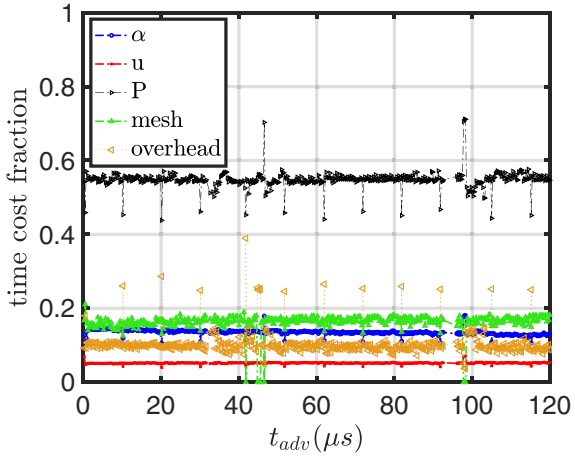

Fig. 6. Time history of the AMR computational time of each calculation (per mesh adaptation interval) over the total computation time (per mesh adaptation interval) for the standing wave problem. The time instant where mesh adaptation is executed is denoted as t_{adv}.

To confirm whether the Poisson system is the most dominant factor in the AMR total computation time, Fig. 6 plots the ratio of time cost for each calculation over total time cost per mesh adaptation interval. The calculations include interface transport, two-phase momentum, Poisson solution, refinement and coarsening of the mesh, and overhead. The difference between the total time cost and the joint cost of all calculations is viewed as overhead costs. Also, during the mesh adaption interval, mesh coarsening/refinement occurs only once, while interface transport, two-phase momentum, and Poisson solution occur at each time step within this interval. The results show that the Poisson solution hovers around 55% of the total computation time, confirming our expectations. For the overhead costs, they are on par with the liquid fraction advection and momentum predictor but well below the Poisson solution.

The performance results in terms of $t_{SM}^{\Delta t}/t_{AMR}^{\Delta t}$ and cell-based speedup, Θ, are shown respectively in Figs. 7a and 7 b. In terms of the recorded values of Θ from the simulation results, these are consistently around one for the entire computation time window. With values of Θ at approximately one and $t_{SM}^{\Delta t}/t_{AMR}^{\Delta t}$ remaining nearly at four, implies that the ratio N_{AMR}/N_{SM} is similarly constant at ≈ 0.25. This behavior indicates that the morphology of the two-phase flow is not substantially changing in time, i.e., while the interface $\Gamma(t)$ is unsteady, the extent of this interface $|\Gamma(t)|$ does not change significantly. In subsequent sections, we will see that this characteristic plays a fundamental role in the performance of AMR. The results presented here confirm the superior performance obtained with AMR and corroborate previous results (Zuzio and Estivalezes, 2011; Popinet, 2009) with this standing wave problem.

4.2. Small domain liquid injection cases

For the small domain liquid injection cases, the region of investigation corresponds to $(L_x, L_y, L_z) = (75, 200, 500)\mu m$ as shown in Fig. 8, where the injector orifice is centered on the left-hand $x-y$ plane. Again the primary reason for examining the performance of AMR under these conditions is that it allows for the direct evaluation of the Frobenius condition number and related parameters. These calculations are performed on a single CPU core (2.4 GHz Xeon E5645), avoiding any potential load balancing issues, which are discussed in Section 4.4. In the present simulations, the domain is initially quiescent, and at $t = 0$, injection is initiated via the following inlet profile,

$$\mathbf{U}(r) = U_{inj}(1 - r/R)^{1/7}\mathbf{e_z}, \tag{29}$$

where r is the radial coordinate, and the injection occurs in the $\mathbf{e}_z$ direction. The one-seventh power law stems from the scaling of the mean injection velocity profile in channel flows (Schlichting et al., 1974). The parameters characterizing the injection conditions are included in Table 2. In choosing the physical properties, the main motivator was ensuring that hydrodynamic breakup occurs relatively soon after injection, primarily to minimize the extent of the domain. Again having the focus of the

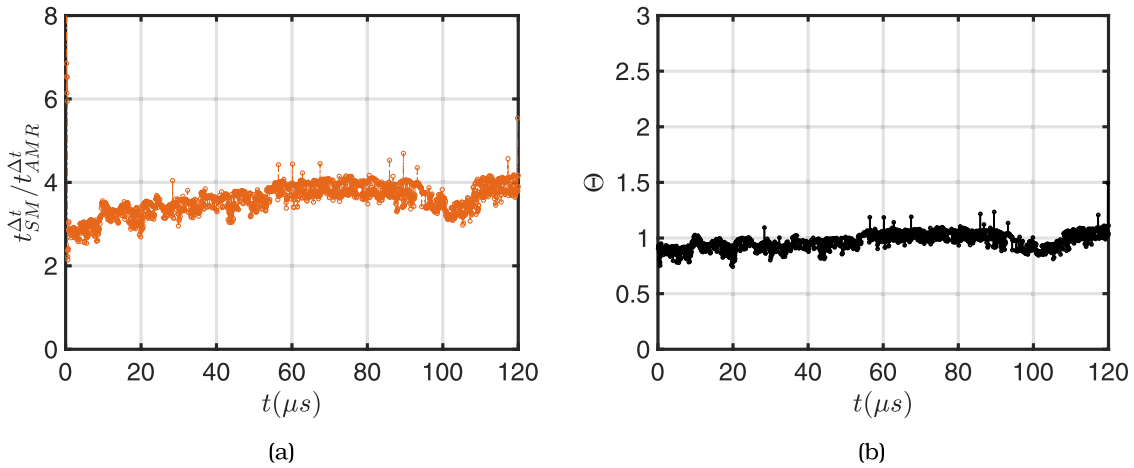

Fig. 7. AMR performance metrics for the standing wave problem.

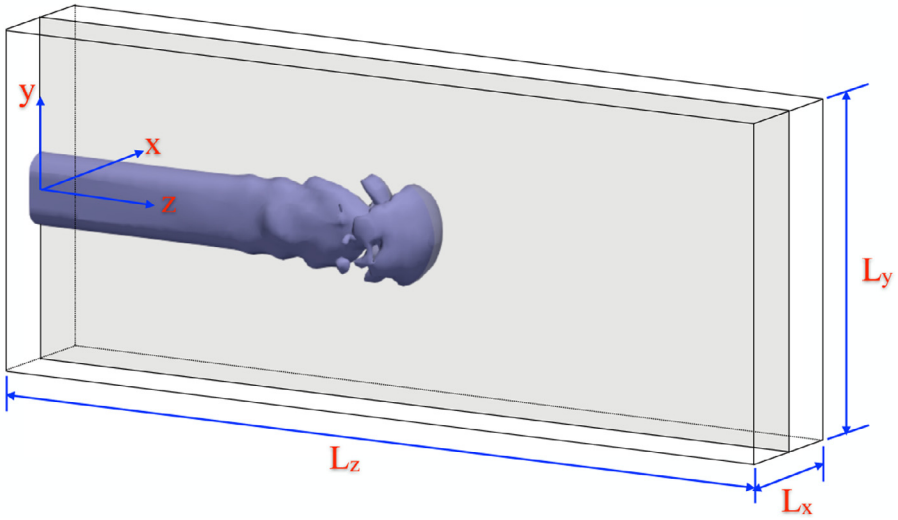

Fig. 8. Illustration of the configuration used in the liquid injection simulations (small domain).

Table 2
Physical properties and injection conditions for the small domain cases.

Injection speed	$U_{inj} = 50$ m/s
Injector radius	$R = 25\ \mu m$
Liquid density	$\rho_L = 1000\ kg/m^3$
Liquid viscosity	$\mu_L = 1\times\ 10^{-5}$ kg/(m-s)
Gas density	$\rho_G = 100\ kg/m^3$
Gas viscosity	$\mu_G = 1\times\ 10^{-6}$ kg/(m-s)
Coefficient of surface tension	$\sigma = 0.01\ kg/s^2$
Number of computational cells in SM	$N_{SM} = 38,880$

present work on the numerical performance issue, we are not interested in obtaining physical quantities from the simulations, and thus, the computations are, to a certain degree, physically unresolved. The AMR refinement criterion is $|\nabla\alpha| \times \Delta x > 10^{-4}$, while the coarsening criterion is $|\nabla\alpha| \times \Delta x < 0.9 \times 10^{-4}$.

A three-level mesh refinement is employed, where the minimum grid size, $\Delta x_{\min} = 4.17$ m. For the SM cases, the refined region, Ω_{ref}, i.e. where $\Delta x = \Delta x_{\min}$ is assigned as

$$\Omega_{ref} = \Big\{(x,y,z) \in \mathbb{R}^3 \mid x \in (-27.5, 27.5)\mu m, \\ y \in (-85, 85)\mu m, z \in (0, 475)\mu m\Big\}. \quad (30)$$

At the wall surrounding the nozzle exit, the no-slip boundary condition for the velocity is prescribed and a zero Neumann condition for the liquid fraction and pressure. On the remaining part of boundaries, we employ a zero Dirichlet condition for pressure, which specifies the total pressure to be zero, and zero Neumann condition for liquid fraction and velocity except for reverse flow, in which case this boundary switches to zero Dirichlet condition.

A visualization of the results is presented in Fig. 9, which shows the deformation of the liquid jet leading into the formation of a mushroom structure at the jet tip. As the calculation proceeds, smaller liquid structures are observed becoming detached from the main jet, and towards the end of the computation, the jet tip is on the verge of breaking up. The underlying AMR grid is also shown accompanying the visualization of the liquid jet.

An examination of the AMR computational cost for all calculations in Fig. 10 reveals that the Poisson costs remain the most significant contributor to the overall calculation burden. The time to adapt the mesh and the overhead costs are not negligible but are noticeably smaller than the Poisson solution. In terms of the AMR behavior, the ratio of computational time per time step shown in Fig. 11a indicates that in contrast to the standing wave problem, there is a precipitous decrease in performance. The theoretical estimate is superimposed on these calculations, which match reasonably well with the recorded simulation results and reproduce the observed trend.

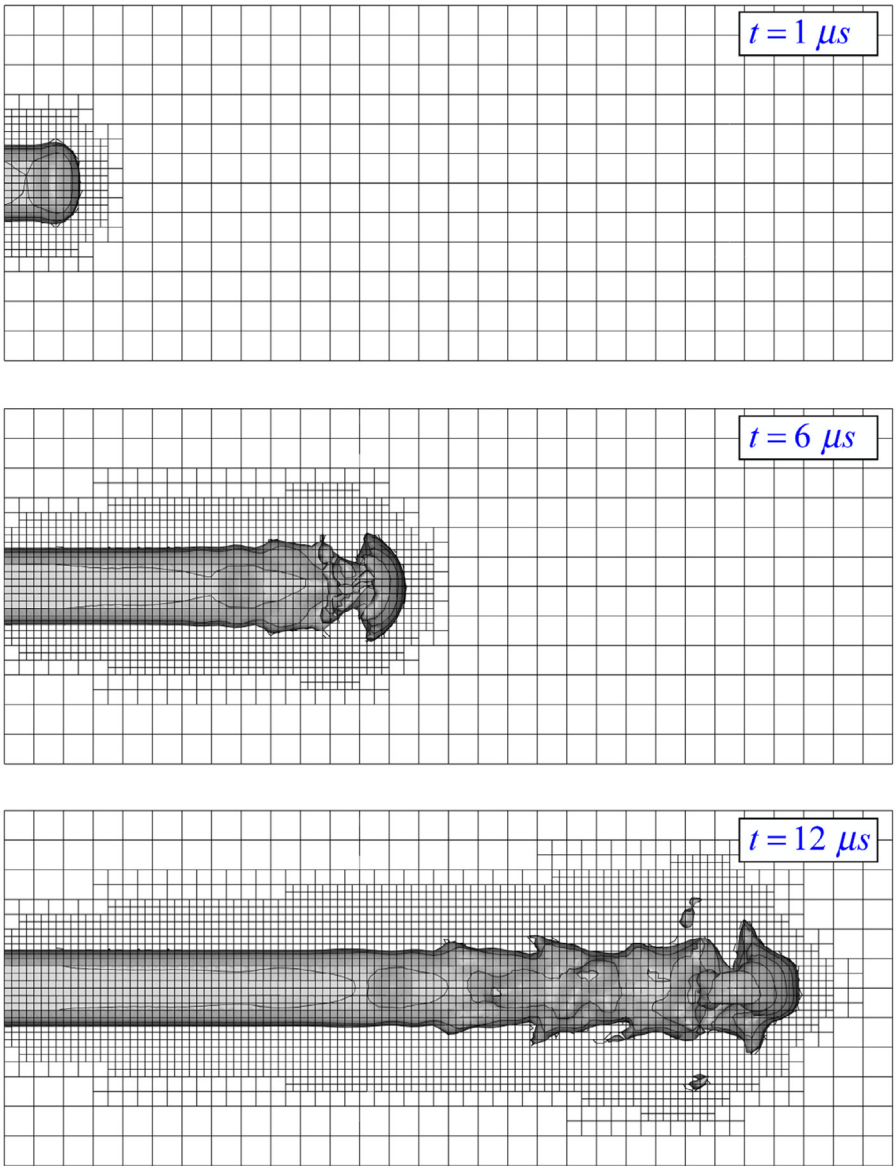

Fig. 9. Evolution of the liquid jet injection process and underlying AMR for the small domain case.

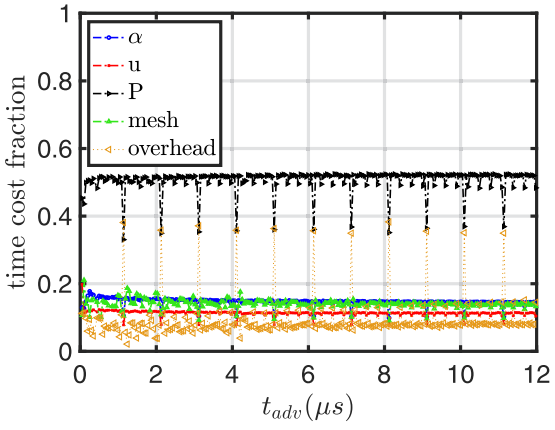

Fig. 10. Time history of the AMR computational time of each calculation (per mesh adaptation interval) over the total computation time (per mesh adaptation interval) for the small domain injection problem. The time instant where mesh adaptation is executed is denoted as t_{adv}.

An ideal case is also included in Fig. 11a, which treats Θ as being equal to one. Under these ideal conditions, the speedup becomes $t_{SM}^{\Delta t}/t_{AMR}^{\Delta t} = N_{SM}/N_{AMR}$, i.e., it scales with the reduction in total number of computational cells. As evidenced in Fig. 11a, this ideal case matches very well with the simulation results, which indicates that the decaying performance of AMR is *being largely fueled by an increase in computational cells*. This is a noticeable difference between spray problems and other two-phase flow problems. In sprays, there is a tremendous growth of the interfacial area while hydrodynamic breakup and liquid penetration take place. The interface monotonically grows, eventually occupying almost the entire domain.

A secondary contributor to the worsening performance of AMR is also a slight decline in cell-based speedup, as shown in Fig. 11a. Beginning at approximately $t = 2\mu s$, the behavior for Θ shows a declining trend, and beyond roughly $t = 8\mu s$, its value becomes less than one. Superimposed on the simulation results, the theoretical estimate, Θ_E, is also included, and while it overpredicts the actual Θ during the early phase of injection, it agrees relatively well in the latter part of the computation time. Since it was in this

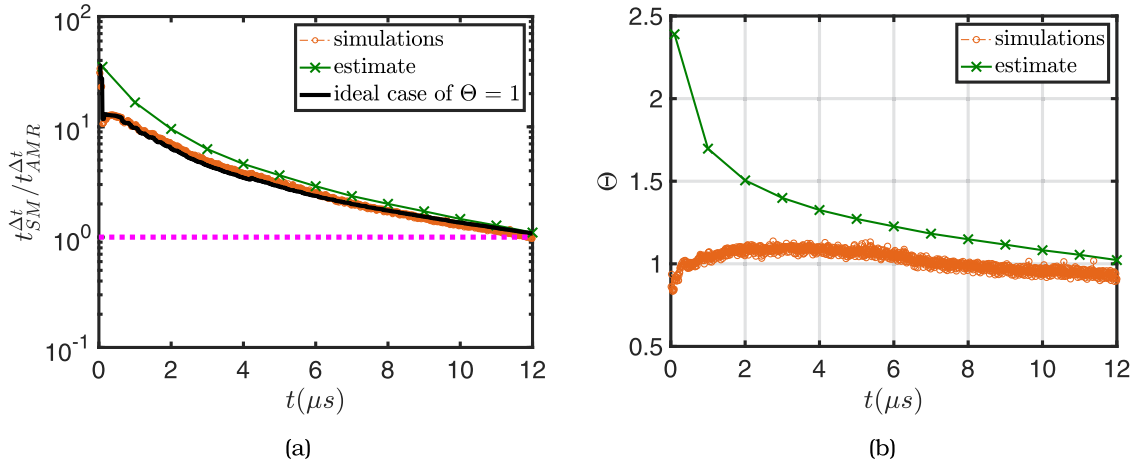

Fig. 11. Decay of the AMR speedup shown in (a) and associated cell-based speedup (b) for the small domain case.

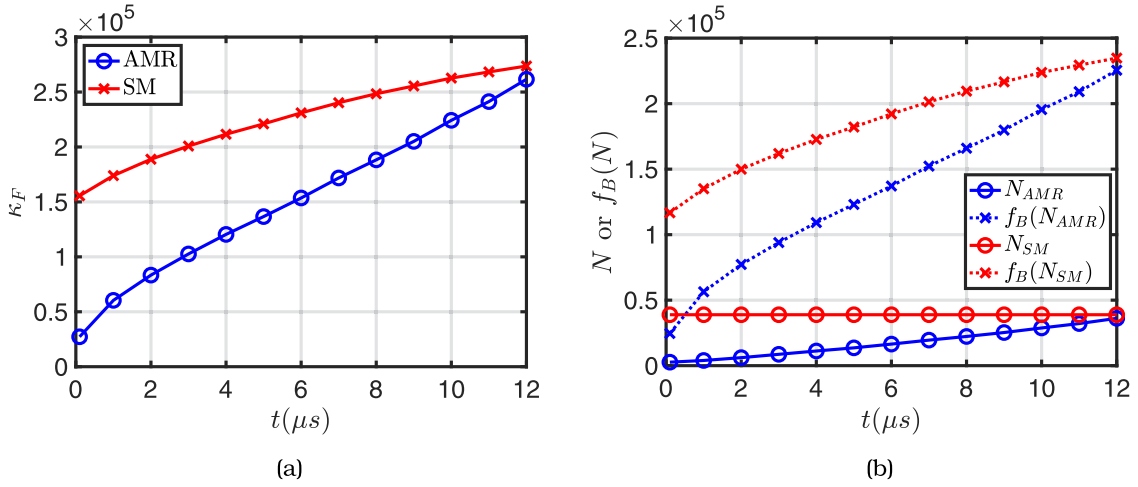

Fig. 12. Growth of the Frobenius condition number shown in (a) and associated components (b) for the small domain case.

latter part when we observed the declining efficiency of AMR, we can look at the theoretical estimate for explanations.

Again the theoretical estimate is given by the ratio of square roots of the Frobenius number (See Eq. 18 and surrounding text) $(\sqrt{\kappa_{F,SM}}/\sqrt{\kappa_{F,AMR}})$. While values for κ_F increase in time for both AMR and SM, as shown in Fig. 12a, the growth is quite more pronounced for the AMR case. This results in the declining trend observed in Fig. 11a. To inspect what contributes more meaningfully to $\kappa_{F,AMR}$, a plot of its components is included in Fig. 12b for both SM and AMR. The variation in N_{AMR} definitely plays a role in the observed growth in $\kappa_{F,AMR}$. However, the more significant contributions originate from $f_B(N_{AMR})$ (see Eq. 5 for the definitions), which produces in large part the decrease in the observed performance of Θ_E and similarly for Θ.

4.3. Large domain liquid injection cases

In the present computations, the shape of the domain is the same as the one shown in Fig. 8 with the exception that the extent is now $(L_x, L_y, L_z) = (1260, 1260, 9000)\mu m$. The physical properties for these cases corresponding to Diesel injection in a pressurized air environment and are listed in Table 3. Also, three different injection conditions are considered corresponding to increases in U_{inj}.

Table 3
Physical properties and injection conditions for the larger domain cases.

Injection velocity	$U_{inj} = (100, 200, 300)$ m/s
Injector radius	$R = 45\mu m$
Liquid density	$\rho_L = 688.03$ kg/m^3
Liquid viscosity	$\mu_L = 4.78 \times 10^{-4}$ kg/(m-s)
Gas density	$\rho_G = 50$ kg/m^3
Gas viscosity	$\mu_G = 1.88 \times 10^{-5}$ kg/(m-s)
Coefficient of surface tension	$\sigma = 0.02$ kg/s^2
Number of computational cells in SM	$N_{SM} = 59,301,368$

The simulations are initiated by injecting fluid via the following velocity profile,

$$\mathbf{U}(r) = U_{inj}(1 - r/R)^{1/7}\mathbf{e_z} + 0.01U_{inj}[1 - (1 - r/R)^{1/7}]\mathbf{e_r}. \quad (31)$$

Here we impose a radial component to reflect some extent the small deviation from pure axial flow observed in typical engineering nozzle flows such as the ECN Spray A configuration (Agarwal and Trujillo, 2018). The calculation is performed in parallel using 180 CPU cores (2.5 GHz Intel Xeon). The AMR refinement criterion is $|\nabla\alpha| \times \Delta x > 10^{-4}$, while the coarsening criterion is $|\nabla\alpha| \times \Delta x < 0.9 \times 10^{-4}$.

The time evolution of the spray as well as the underlying AMR are shown in Figs. 13 and 14, respectively ($U_{inj} = 300$ m/s). In the beginning phase of injection, the jet tip is still present as a

Fig. 13. Snapshots of the large liquid injection case ($U_{inj} = 300$ m/s) at three different times displays the evolution of the hydrodynamic breakup process.

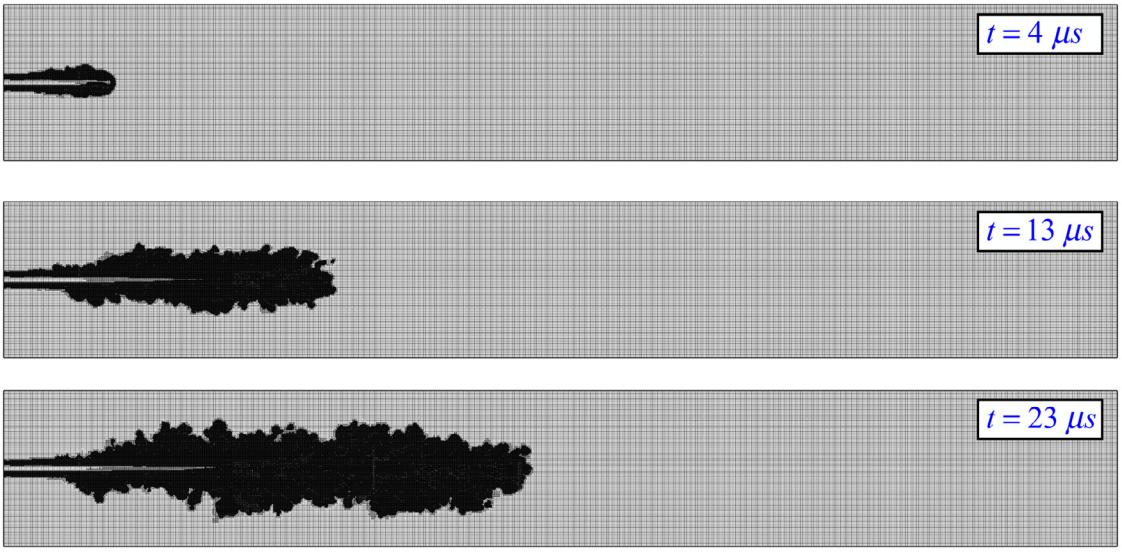

Fig. 14. Evolution of the AMR grid corresponding to the same snapshots captured in Fig. 13.

mushroom-shaped object. As the calculation proceeds, this tip becomes fragmented, creating a population of droplets, which are propelled both radially and axially. This progression of droplets is what drives the local refinement. Since the population of droplets and distorted liquid structures is numerous, the adaptive mesh grows to fill a relatively large domain. As shown in Fig. 15, the Poisson system solution remains the largest fraction of the computation cost hovering above 65% for much of the AMR computation. The overhead costs are higher than all other costs except pressure. Overhead varies between approximately 17% and 33% of the pressure cost for this large domain case.

Concerning the performance of AMR, a similar pattern to the previous small-domain cases is observed. The results for speedup and cell-based speedup are shown in Figs. 16a and 16b with respect to three different injection conditions. The most noticeable difference for the larger cases is that the drop in AMR performance is even more accentuated. In fact, towards the end of the computation, the benefits offered by AMR are practically negligible as $t^{\Delta t}_{AMR}$ is roughly the same as $t^{\Delta t}_{SM}$. This drop in performance occurs faster with increasing injection speed as the cloud of droplets and ligaments occupy the physical domain more quickly.

While we may be tempted to blame responsibility for the decaying AMR performance to the much larger rise in N_{AMR}, this is not the full story. A significant part of this observed behavior is due to a decrease in the cell-based speedup, Θ, as shown in Fig. 16b. In the latter part of the computation, Θ dips below one, which indicates that the computational efficiency of AMR is lower than SM. In terms of linear algebra metrics, this implies that the Frobenius condition number for AMR is larger than the corresponding values for SM. Unfortunately for this case, due to its size, which is approximately 1525 times larger in N_{SM} than for the small domain case, we are unable to provide Frobenius condition metrics.

An insightful exercise is to compare the performance of these AMR calculations with the ideal case ($\Theta = 1$). The results, in terms of the histories of the computational time ratios, are shown in

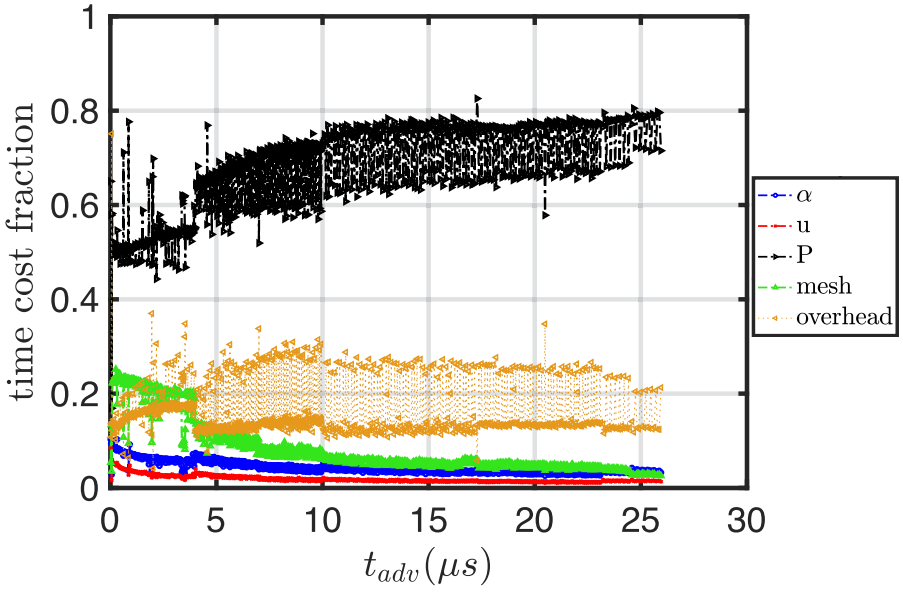

Fig. 15. Time history of the AMR computational time of each calculation (per mesh adaptation interval) over the total computation time (per mesh adaptation interval) for the large domain injection problem (U_{inj} = 300 m/s).

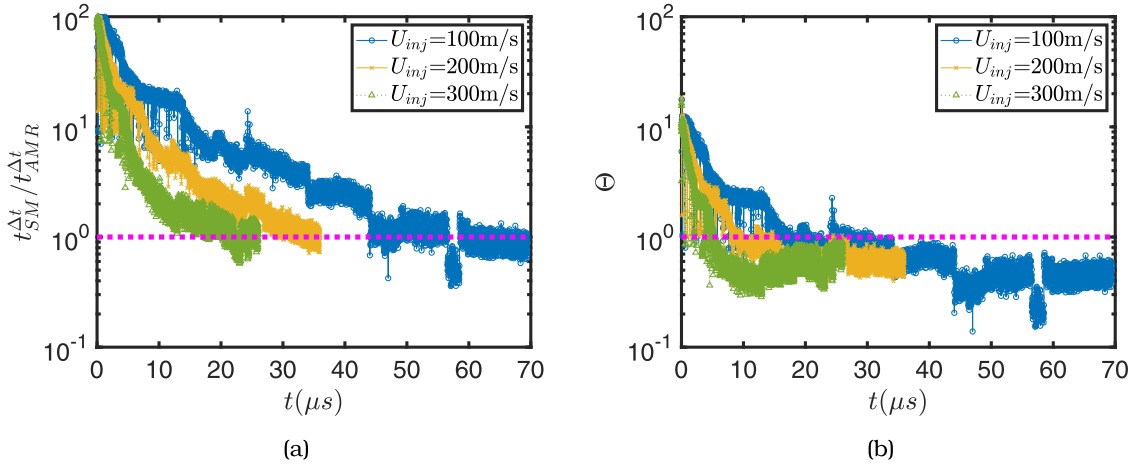

Fig. 16. Decay of AMR performance as measured by (a) the speedup and (b) cell-based speedup corresponding to the large domain cases.

Figs. 17a, 17b, and 17c corresponding to the different injection speeds. The trend captured in all three injection conditions indicates that at an early time instant when the number of computational cells in AMR is low, the performance is better than the ideal case ($\Theta > 1$). A likely reason for this behavior can be found by considering the theoretical estimate for Θ, i.e. $\Theta_E = \sqrt{\kappa_{F,SM}}/\sqrt{\kappa_{F,AMR}}$. For instance, during the early injection period, $\kappa_{F,AMR} \ll \kappa_{F,SM}$ due in part to the much lower number of cells in AMR. However, once the liquid population envelops a greater portion of the domain, the reverse trend unfolds. During this latter period, which highlights the greater portion of the total computation time, the efficiency of AMR becomes worse than SM. While the computational cost savings in terms of pure $t^{\Delta t}$ is better than SM; it ceases to reflect the benefits afforded by a substantially lower computational cell count. A time history of the ratio N_{AMR}/N_{SM} for each of the three injection cases included in Fig. 17d highlights this observation.

4.4. Other potential contributors to AMR performance deterioration

Besides the Poisson solution costs, there may be other contributors to the decline of AMR performance that merit exploration. A common source of concern is load imbalance issues. In the present computations, domain decomposition based on `Scotch` method is used for parallelization (Guide, 2012), and under this methodology, a load imbalance would result if a given part of the domain has a much higher grid density than another part. Under this scenario, CPU(s) handling calculations in the higher grid density region would naturally be more strained than those CPU(s) responsible for the lowest grid density region and would cause a bottleneck in performance. The highest grid density is located where atomization is most vigorous and is characterized by a local cell size equal to $\Delta x_{\min}$. In both AMR and SM, there are CPUs that are handling calculations in this high grid density region, and thus both methods are exposed to equivalent levels of bottleneck behavior.

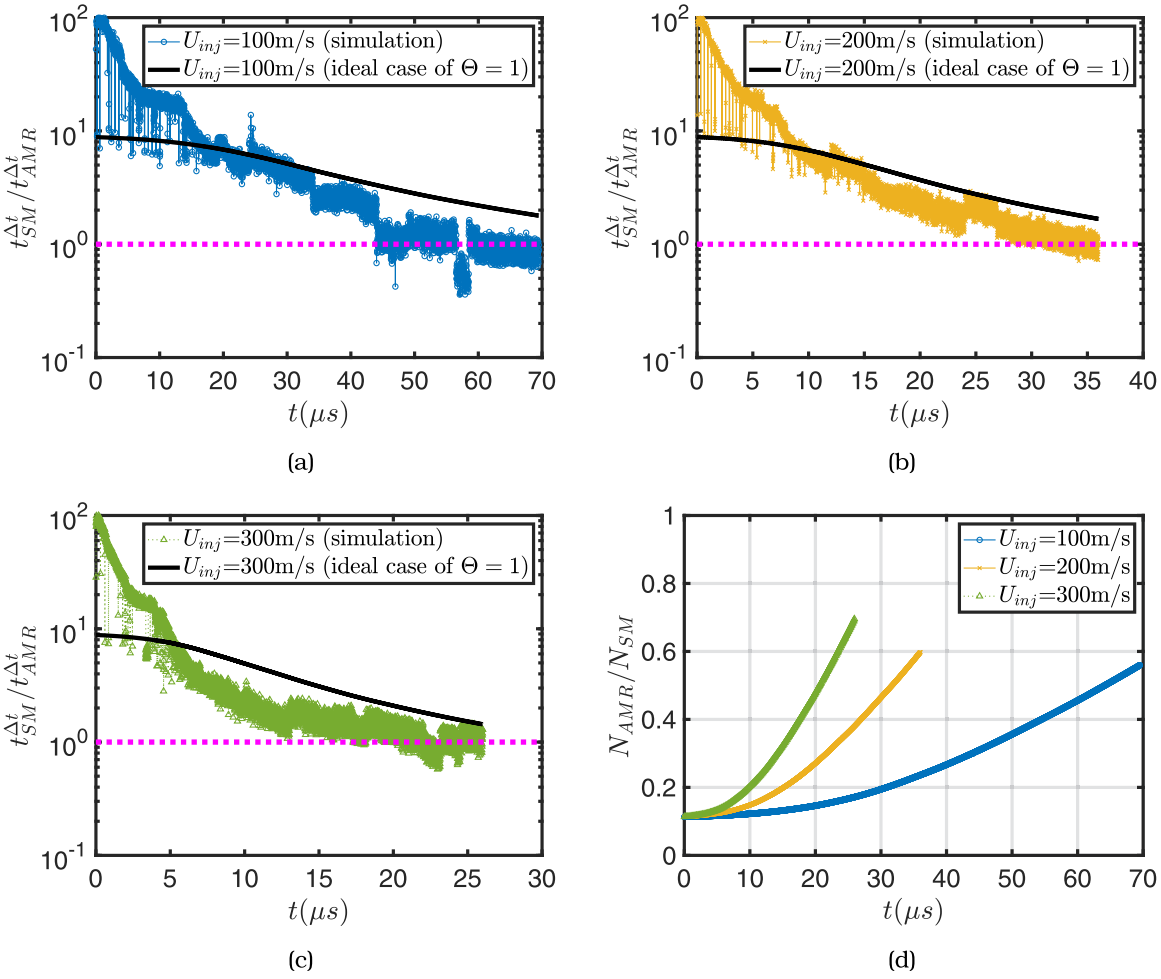

Fig. 17. Comparison of the recorded AMR performance against the ideal case of $\Theta = 1$ (a,b,c) indicates that beyond the initial period, the recorded AMR performance declines substantially even to the point where it has a speedup near or below one, even though (d) the number of computational cells for AMR is uniformly lower than SM (large domain case).

Nevertheless, for the small domain cases, the drop in performance measured by Θ is approximately 15% (Fig. 11a), while for the larger domain cases, the drop in Θ is slightly more than one order of magnitude (Fig. 16b). Hence, it is natural to question whether the trends observed are, in fact, due to a progressively diminishing value for $\sqrt{\kappa_{F,SM}}/\sqrt{\kappa_{F,AMR}}$ or a load balance problem. Since values for the Frobenius condition number are unavailable for the larger domain cases and the performance deterioration for AMR manifests itself more strongly for larger cases, one option for studying the behavior is to consider the largest reasonable case for which a single CPU core calculation can be performed. This is the case examined in the present work. For the sake of curiosity, additional calculations performed with 4 and 8 CPU cores are also reported.

The conditions for these load balancing calculations match those of Table 2 with the exception that $R = 20\mu m$ and $U_{inj} = 150 m/s$, and that the domain extent is $(L_x, L_y, L_z) = (160, 160, 1700)\mu m$. The total number of cells for the SM case is $N_{SM} = 2{,}785{,}280$, and the AMR refinement criterion is $|\nabla\alpha| \times \Delta x > 10^{-6}$, and for coarsening $|\nabla\alpha| \times \Delta x < 0.5 \times 10^{-6}$.

The results in terms of the time evolution for Θ are presented in Figs. 18a and 18b. While the values are not expected to be the same, since the parallel scaling is not 100%, the overall declining trend is similar among all cases considered. For these calculations, the reference is the single-core computation since, by definition, no load balancing issues exist under this mode of calculation. In the beginning phase of the calculation, the 4-core and 8-core executions have lower performance than the single-core case but still provide significant benefits in terms of speedup over the SM case. However, as the calculation proceeds, all cases exhibit the same declining trend, with the 8-core execution showing marginal gains over the other cases. Hence, the trend of downward AMR performance is exhibited even in single-core computations showing a drop in performance in a speedup by approximately two orders of magnitude and a drop in a cell-based speedup by approximately one order of magnitude.

A second concern is the choice of a numerical method for solving the Poisson system. In the present work, we employed the PCG solver as outlined in Section 2, and while we do not intend to engage in an exhaustive examination of the solution characteristics subject to various linear system solution methods, it is instructive to at least consider an alternative method that is common in CFD. This alternative method is the Generalized Geometric-Algebraic Multi-grid solver, which is exercised on the large domain spray case (U_{inj} = 300 m/s) using the same DIC preconditioner as the PCG calculations. The results are presented in Fig. 19a and reveal that using a different linear solve leads to the same pattern of decaying AMR performance. The ideal case ($\Theta = 1$) is employed

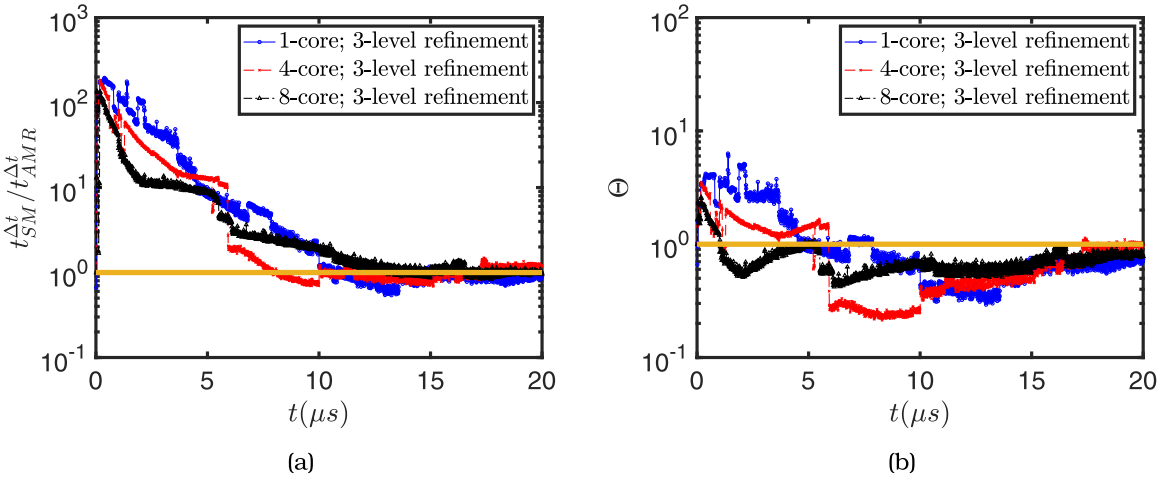

Fig. 18. AMR performance employing 1, 4, and 8 CPU core executions shows a similar pattern of decay.

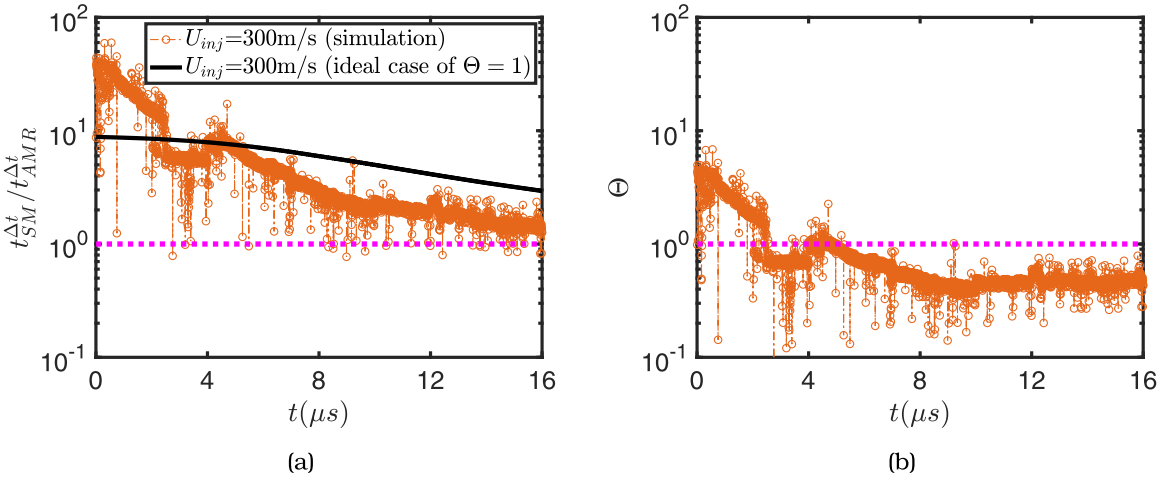

Fig. 19. AMR performance solving the PPE using Generalized Geometric-Algebraic Multi-grid corresponding to the large liquid injection case (U_{inj} = 300 m/s).

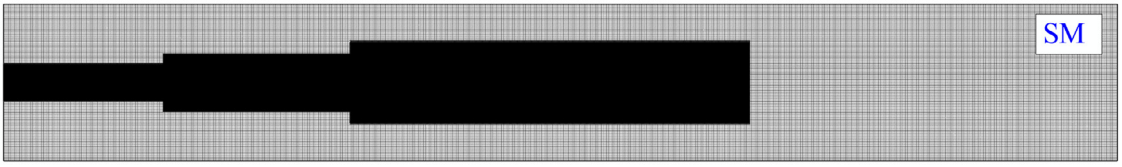

Fig. 20. Illustration of an octree-refined static mesh using a telescoping strategy for handling spray problems. With increasing distance from injection, the radial spread of the liquid grows, and thus the extent of radial refinement conforms to this spread. Mesh refinement does not encompass the entire length of the domain; hence, the finest level of resolution is only provided in the near field or during the initial period of injection. For studies focused only on primary atomization physics or liquid core breakup (Agarwal and Trujillo, 2018), this mesh refinement strategy is adequate.

to illustrate this behavior. Towards the end of the computation, the computational time per timestep is almost the same between AMR and SM, although the number of cells employed in AMR is substantially lower. Additionally, the cell-based speedup dips below one, as shown in Fig. 19b, in a similar fashion to the PCG cases reported previously.

5. Conclusions

Due to the dominant cost of Pressure Poisson solution in the simulation of spray problems, the present work has focused on the computational costs of this solution as a way to understand the declining trend in AMR performance. From the definition of speedup, which is repeated here as $t^{\Delta t}_{SM}/t^{\Delta t}_{AMR} = \Theta(N_{SM}/N_{AMR})$, two contributions to the performance of AMR are the cell-based speedup (Θ) and the ratio of computational cells. For spray problems, which are characterized by a huge increase in interfacial area, it is logical to expect a decrease in speedup as the liquid injection and atomization unfold, since the interfacial area growth is intrinsically associated with an increase in N_{AMR}. However, as shown in the present work, a less obvious contributor is the decrease in the cell-based speedup.

As derived in Section 2, a theoretical estimate for the cell-based speedup is obtained, namely $\Theta_E = \sqrt{\kappa_{F,SM}}/\sqrt{\kappa_{F,AMR}}$, where κ_F is the Frobenius condition number for the respective SM and AMR Poisson systems. It is shown that as the calculation of atomization unfolds, there is a noticeable decline in $\kappa_{F,SM}/\kappa_{F,AMR}$ leading directly to the decrease in cell-based speedup. Under certain conditions, AMR does not lead to an advantage, i.e., $t^{\Delta t}_{SM}/t^{\Delta t}_{AMR}$ is at or below one, even though the total number of computa-

tional cells in AMR is lower than the corresponding SM. To examine why κ_F increases more pronouncedly for AMR versus SM, we can refer to the eigenvalue product equality with κ_F, i.e. $\kappa_F = \sqrt{\sum_{i=1}^{N}\lambda_i^2}\sqrt{\sum_{i=1}^{N}\lambda_i^{-2}}$. Since N naturally increases with AMR but not with SM, even if the magnitude of $\lambda_i, i = 1, 2, \ldots, N$, remain quite the same, this behavior would lead to the more rapid growth of $\kappa_{F,AMR}$ compared with $\kappa_{F,SM}$.

We should keep in mind that for arriving at the previously mentioned conclusion, the AMR computations were performed against a well-tailored static mesh, which employs octree refinement but in a fixed or static manner. This is depicted in Fig. 20. In general, if comparisons are performed against a highly refined mesh, where the grid spacing is $\Delta x_{\min}$ everywhere in the computational domain, the speedup values will almost certainly stay well above one for the entire duration of the simulation. Also, these observations hold for a particular type of two-phase flow problems, such as those characterizing sprays, where an explosive growth of interfacial area occurs. As shown in the above injection cases, for these problems, interfacial area production is so severe that AMR typically results in an entire region being completely refined rather than the case for more mild two-phase problems, where generally isolated narrow bands of refinement would characterize the mesh. Hence, under atomization conditions, an alternative is to employ static refinement in the local region associated with the high interfacial area density rather than incur the additional costs of AMR.

In many two-phase flow problems not characterized by huge increases of interfacial area, variations in (N_{SM}/N_{AMR}) are expected to be mild, resulting in near steady values for Θ. The standing wave case examined in Section 4.1 falls under this category. Under these conditions, the speedup values will be in proportion to (N_{SM}/N_{AMR}) (see content around Eq. 15), and the computational cost savings will scale directly with the savings in the number of computational cells.

Declaration of Competing Interest

None.

Acknowledgements

The support from the National Science Foundation (Number 1703825) is gratefully acknowledged. Our thanks are also due to the Center for High Throughput Computing (CHTC) at UW-Madison for providing computing resources, along with Joshua Leach, for administering the computing resources within the research group. Additionally, this work partially used the Extreme Science and Engineering Discovery Environment (XSEDE) Bridges regular memory at the Pittsburgh Supercomputing Center through allocation TG-CTS180037. The authors are grateful for the access granted to this resource.

Appendix A. Simplified expression for κ_F

To further explore the makeup of Θ_E, let us return to the original definition of κ_F (Eq. 5) and focus on $f_B(N)$. Since B is a symmetric matrix, this yields $B^T = B$ and $(B^{-1})^T = (B^T)^{-1} = B^{-1}$ (Lay et al., 2016, pg. 107). Thus $f_B(N)$ can be recast as,

$$\begin{aligned} f_B(N) &= \frac{1}{2}\left\{Tr\left[\left(B-B^{-1}\right)^T\left(B-B^{-1}\right)\right]\right. \\ &\quad \left.-\left(||B||_F^2 - 2||B||_F||B^{-1}||_F + ||B^{-1}||_F^2\right)\right\} \\ &= \frac{1}{2}\left\{Tr\left[(B^T-B^{-1})(B-B^{-1})\right]\right. \\ &\quad \left.-\left[Tr(B^TB) - 2\sqrt{Tr(B^TB)}\sqrt{Tr[(B^{-1})^TB^{-1}]} + Tr\left[(B^{-1})^TB^{-1}\right]\right]\right\} \\ &= \frac{1}{2}\left\{Tr\left[B^TB - B^{-1}B - B^TB^{-1} + (B^{-1})^2\right]\right. \\ &\quad \left.-\left[Tr(B^2) - 2\sqrt{Tr(B^2)}\sqrt{Tr[(B^{-1})^2]} + Tr\left[(B^{-1})^2\right]\right]\right\} \\ &= \frac{1}{2}\left\{Tr(B^2) - 2N + Tr[(B^{-1})^2]\right. \\ &\quad \left.-\left[Tr(B^2) - 2\sqrt{Tr(B^2)}\sqrt{Tr[(B^{-1})^2]} + Tr[(B^{-1})^2]\right]\right\} \\ &= \frac{1}{2}\left\{-2N - \left[-2\sqrt{Tr(B^2)}\sqrt{Tr[(B^{-1})^2]}\right]\right\} \\ &= -N + \sqrt{Tr(B^2)}\sqrt{Tr(B^{-2})}. \end{aligned} \tag{32}$$

Introducting Eq. (32) into Eq. (5) gives

$$\kappa_F(B) = N + f_B(N) = \sqrt{Tr(B^2)}\sqrt{Tr(B^{-2})}. \tag{33}$$

To reduce this further, we note that B is symmetric, so it is therefore orthogonally diagonalizable (Lay et al., 2016, pg. 398). Moreover, it admits a spectral decomposition (Lay et al., 2016, pg. 399), namely

$$B = QDQ^{-1}, \tag{34}$$

where Q is the matrix whose columns are the eigenvectors of B, and D is the diagonal matrix composed of the eigenvalues of B. Additionally,

$$B^2 = (QDQ^{-1})(QDQ^{-1}) = QD^2Q^{-1}, \tag{35}$$

which can be generalized to n, i.e. $B^n = QD^nQ^{-1}$. Since B and D are similar matrices, they have the same traces (Lay et al., 2016, pg. 296), thus

$$Tr(D) = \sum_{i=1}^{N}\lambda_i = Tr(B) \tag{36}$$

and

$$Tr(B^2) = Tr(D^2) = \sum_{i=1}^{N}\lambda_i^2. \tag{37}$$

For the B^{-1} case, we have that

$$\begin{aligned} B^{-1}B &= I \\ B^{-1}(QDQ^{-1}) &= I(\text{from Eq. 34}) \\ B^{-1}(QDQ^{-1})(QD^{-1}Q^{-1}) &= I(QD^{-1}Q^{-1}) \\ B^{-1} &= QD^{-1}Q^{-1}, \end{aligned} \tag{38}$$

which similarly leads to

$$B^{-2} = QD^{-2}Q^{-1}, \tag{39}$$

or more generally to $B^{-n} = QD^{-n}Q^{-1}$. To obtain a simplified relation for $Tr(B^{-2})$ requires an explicit expression for D^{-1}, which can be obtained as follows,

$$\begin{aligned} \sum_{k=1}^{N} D_{ik}(D^{-1})_{kj} &= \delta_{ij}(\delta_{\text{ij}}\text{is the Kronecker delta}) \\ \lambda_i\sum_{k=1}^{N}\delta_{ik}(D^{-1})_{kj} &= \delta_{ij}\ (\text{summation convention not implied}) \\ \lambda_i\delta_{ii}(D^{-1})_{ij} &= \delta_{ij}(\text{summation convention not implied}) \\ (D^{-1})_{ij} &= \lambda_i^{-1}\delta_{ij}. \end{aligned} \tag{40}$$

Not surprising ly, D^{-1} is also a diagonal matrix. By inspection it follows that $(D^{-2})_{ij} = \lambda_i^{-2}\delta_{ij}$ (summation convention not implied).

From Eq. (39), B^{-2} and D^{-2} are similar matrices. Therefore, they both have the same traces, namely

$$Tr\left(B^{-2}\right) = Tr\left(D^{-2}\right) = \sum_{i=1}^{N} \lambda_i^{-2}. \quad (41)$$

Putting Eqs. (37) and (41) into Eq. (33) gives a final expression for the Frobenius condition number as

$$\kappa_F(B) = \sqrt{Tr(B^2)}\sqrt{Tr\left(B^{-2}\right)} = \left(\sum_{i=1}^{N} \lambda_i^2\right)^{1/2} \left(\sum_{i=1}^{N} \lambda_i^{-2}\right)^{1/2}. \quad (42)$$

References

Adjerid, S., Flaherty, J.E., 1986. A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations. SIAM J. Numer. Anal. 23 (4), 778–796.

Agarwal, A., Trujillo, M.F., 2018. A closer look at linear stability theory in modeling spray atomization. Int. J. Multiphase Flow 109, 1–13.

Agarwal, A., Trujillo, M.F., 2020. The effect of nozzle internal flow on spray atomization. Int. J. Engine Res. 21 (1), 55–72.

Akhtar, M.W., Kleis, S.J., 2013. Boiling flow simulations on adaptive octree grids. Int. J. Multiphase Flow 53, 88–99.

Anjos, G.R., Borhani, N., Mangiavacchi, N., Thome, J.R., 2014. A 3d moving mesh finite element method for two-phase flows. J. Comput. Phys. 270, 366–377.

Arienti, M., Sussman, M., 2015. A High-Fidelity Study of High-Pressure Diesel Injection. Technical Report. SAE Technical Paper.

Ascher, U., Greif, C., 2011. A first course in numerical methods. SIAM.

Babuvška, I., Rheinboldt, W.C., 1978. Error estimates for adaptive finite element computations. SIAM J. Numer. Anal. 15 (4), 736–754.

Balsara, D.S., 2001. Divergence-free adaptive mesh refinement for magnetohydrodynamics. J. Comput. Phys. 174 (2), 614–648.

Berger, M.J., Colella, P., 1989. Local adaptive mesh refinement for shock hydrodynamics. J. Comput. Phys. 82 (1), 64–84.

Berger, M.J., Oliger, J., 1984. Adaptive mesh refinement for hyperbolic partial differential equations. J. Comput. Phys. 53 (3), 484–512.

Brandt, A., 1977. Multi-level adaptive solutions to boundary-value problems. Math. Comput. 31 (138), 333–390.

Chehab, J.-P., Raydan, M., 2008. Geometrical properties of the frobenius condition number for positive definite matrices. Linear Algebra Appl. 429 (8–9), 2089–2097.

Chen, X., Ma, D., Yang, V., Popinet, S., 2013. High-fidelity simulations of impinging jet atomization. Atomization Sprays 23 (12).

Chen, X., Yang, V., 2014. Thickness-based adaptive mesh refinement methods for multi-phase flow simulations with thin regions. J. Comput. Phys. 269, 22–39.

Deshpande, S.S., Anumolu, L., Trujillo, M.F., 2012. Evaluating the performance of the two-phase flow solver interfoam. Computational science & discovery 5 (1), 014016.

Deshpande, S.S., Gurjar, S.R., Trujillo, M.F., 2015. A computational study of an atomizing liquid sheet. Physics of Fluids 27 (8), 082108.

DeZeeuw, D., Powell, K.G., 1993. An adaptively refined cartesian mesh solver for the euler equations. J. Comput. Phys. 104 (1), 56–68.

Dörfler, W., 1996. A convergent adaptive algorithm for poisson's equation. SIAM J. Numer. Anal. 33 (3), 1106–1124.

Fuster, D., Bagué, A., Boeck, T., Le Moyne, L., Leboissetier, A., Popinet, S., Ray, P., Scardovelli, R., Zaleski, S., 2009. Simulation of primary atomization with an octree adaptive mesh refinement and VOF method. Int. J. Multiphase Flow 35 (6), 550–565.

Fuster, D., Matas, J.-P., Marty, S., Popinet, S., Hoepffner, J., Cartellier, A., Zaleski, S., 2013. Instability regimes in the primary breakup region of planar coflowing sheets. J. Fluid Mech. 736, 150–176.

Guide, O.U., 2012. The open source CFD toolbox user guide, Version 2.1.1. OpenCFD Limited. https://github.com/OpenFOAM/OpenFOAM-2.1.x/blob/master/doc/Guides-a4/UserGuide.pdf.

Harten, A., Hyman, J.M., 1983. Self adjusting grid methods for one-dimensional hyperbolic conservation laws. J. Comput. Phys. 50 (2), 235–269.

Herrmann, M., 2010. Detailed numerical simulations of the primary atomization of a turbulent liquid jet in crossflow. J. Eng. Gas Turbine Power 132 (6), 061506.

Hornung, R.D., Trangenstein, J.A., 1997. Adaptive mesh refinement and multilevel iteration for flow in porous media. J. Comput. Phys. 136 (2), 522–545.

Jain, M., Prakash, R.S., Tomar, G., Ravikrishna, R.V., 2015. Secondary breakup of a drop at moderate weber numbers. Proc. R. Soc. A 471 (2177), 20140930.

Jarrahbashi, D., Sirignano, W.A., 2014. Vorticity dynamics for transient high-pressure liquid injection. Physics of Fluids 26 (10), 73.

Jiang, X., Lai, C.-H., 2016. Numerical techniques for direct and large-eddy simulations. CRC Press.

Jiang, X., Siamas, G.A., Jagus, K., Karayiannis, T.G., 2010. Physical modelling and advanced simulations of gas–liquid two-phase jet flows in atomization and sprays. Prog. Energy Combust. Sci. 36 (2), 131–167.

Johnston, H., Liu, J.-G., 2002. Finite difference schemes for incompressible flow based on local pressure boundary conditions. J. Comput. Phys. 180 (1), 120–154.

Kuo, C.W., Trujillo, M.F., 2018. Benefits of AMR for atomization calculations. In: ICLASS 2018, 14th Triennial International Conference on Liquid Atomization and Spray Systems. Institute of Liquid Atomization and Spray Systems, Chicago, IL.

Kuo, C.W., Trujillo, M.F., 2019. Speedup analysis of adaptive mesh refinement in the simulation of spray formation. In: ILASS-Americas 30th Annual Conference on Liquid Atomization and Spray Systems. Institute of Liquid Atomization and Spray Systems, Tempe, AZ, pp. 1–7.

Kwak, D., Kiris, C.C., 2010. Computation of viscous incompressible flows. Springer Science & Business Media.

Laurmaa, V., Picasso, M., Steiner, G., 2016. An octree-based adaptive semi-lagrangian VOF approach for simulating the displacement of free surfaces. Computers & Fluids 131, 190–204.

Lay, D.C., Lay, S.R., McDonald, J.J., 2016. Linear algebra and its applications 5th edition. Pearson.

Lebas, R., Menard, T., Beau, P.-A., Berlemont, A., Demoulin, F.-X., 2009. Numerical simulation of primary break-up and atomization: DNS and modelling study. Int. J. Multiphase Flow 35 (3), 247–260.

LeVeque, R.J., 2007. Finite difference methods for ordinary and partial differential equations. SIAM.

Li, X., Soteriou, M., 2012. Prediction of high density-ratio liquid jet atomization in crossflow using high fidelity simulations on HPC. In: 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, p. 175.

Ling, Y., Zaleski, S., Scardovelli, R., 2015. Multiscale simulation of atomization with small droplets represented by a lagrangian point-particle model. Int. J. Multiphase Flow 76, 122–143.

Liu, X., 2013. Parallel modeling of three-dimensional variably saturated ground water flows with unstructured mesh using open source finite volume platform openFOAM. Engineering Applications of Computational Fluid Mechanics 7 (2), 223–238.

Löhner, R., Mut, F., Cebral, J.R., Aubry, R., Houzeaux, G., 2011. Deflated preconditioned conjugate gradient solvers for the pressure-poisson equation: extensions and improvements. Int. J. Numer. Methods Eng. 87 (1–5), 2–14.

Malik, M., Fan, E.S.-C., Bussmann, M., 2007. Adaptive VOF with curvature-based refinement. Int. J. Numer. Methods Fluids 55 (7), 693–712.

Miller, K., Miller, R.N., 1981. Moving finite elements. i. SIAM J. Numer. Anal. 18 (6), 1019–1032.

Moayedi, H., Amanifard, N., Deylami, H.M., Dolati, F., 2017. Numerical investigation of using micropolar fluid model for EHD flow through a smooth channel. J. Electrostat. 87, 51–63.

Ortiz, M., Quigley Iv, J.J., 1991. Adaptive mesh refinement in strain localization problems. Comput. Methods Appl. Mech. Eng. 90 (1–3), 781–804.

Popinet, S., 2009. An accurate adaptive solver for surface-tension-driven interfacial flows. J. Comput. Phys. 228 (16), 5838–5866.

Quan, S., Lou, J., Schmidt, D.P., 2009. Modeling merging and breakup in the moving mesh interface tracking method for multiphase flow simulations. J. Comput. Phys. 228 (7), 2660–2675.

Riviere, B., Wheeler, M.F., Girault, V., 1999. Improved energy estimates for interior penalty, constrained and discontinuous galerkin methods for elliptic problems. part i. Comput. Geosci. 3 (3–4), 337–360.

Rusche, H., 2003. Computational fluid dynamics of dispersed two-phase flows at high phase fractions. Imperial College London (University of London) PhD dissertation.

Saad, Y., 2003. Iterative methods for sparse linear systems, Vol. 82. siam.

Schlichting, H., et al., 1974. Boundary-layer theory. Springer.

Shinjo, J., Umemura, A., 2010. Simulation of liquid jet primary breakup: dynamics of ligament and droplet formation. Int. J. Multiphase Flow 36 (7), 513–532.

Strotos, G., Malgarinos, I., Nikolopoulos, N., Gavaises, M., 2016. Predicting droplet deformation and breakup for moderate weber numbers. Int. J. Multiphase Flow 85, 96–109.

Tavangar, S., Hashemabadi, S.H., Saberimoghadam, A., 2015. Cfd simulation for secondary breakup of coal–water slurry drops using openfoam. Fuel Process. Technol. 132, 153–163.

Teyssier, R., 2002. Cosmological hydrodynamics with adaptive mesh refinement-a new high resolution code called RAMSES. Astronomy & Astrophysics 385 (1), 337–364.

Theodorakakos, A., Bergeles, G., 2004. Simulation of sharp gas–liquid interface using VOF method and adaptive grid local refinement around the interface. Int. J Numer Methods Fluids 45 (4), 421–439.

Tonini, S., Gavaises, M., Theodorakakos, A., 2008. Modelling of high-pressure dense diesel sprays with adaptive local grid refinement. Int. J. Heat Fluid Flow 29 (2), 427–448.

Trefethen, L.N., Bau III, D., 1997. Numerical linear algebra. SIAM.

Trujillo, M.F., Gurjar, S., Mason, M., Agarwal, A., 2018. Global characterization of the spray formation process. Atomization Sprays 28 (9), 811–835.

Ubbink, O., Issa, R.I., 1999. A method for capturing sharp fluid interfaces on arbitrary meshes. J. Comput. Phys. 153 (1), 26–50.

Verfürth, R., 1994. A posteriori error estimation and adaptive mesh-refinement techniques. J. Comput. Appl. Math. 50 (1–3), 67–83.

Xie, Z., Pavlidis, D., Percival, J.R., Gomes, J.L., Pain, C.C., Matar, O.K., 2014. Adaptive unstructured mesh modelling of multiphase flows. Int. J. Multiphase Flow 67, 104–110.

Xue, Q., Kong, S.-C., 2009. Development of adaptive mesh refinement scheme for engine spray simulations. Computers & Fluids 38 (4), 939–949.

Yang, W., Jia, M., Sun, K., Wang, T., 2016. Influence of density ratio on the secondary atomization of liquid droplets under highly unstable conditions. Fuel 174, 25–35.

Zuzio, D., Estivalezes, J.L., 2011. An efficient block parallel AMR method for two phase interfacial flow simulations. Computers & Fluids 44 (1), 339–357.