Journal of Computational Physics 493 (2023) 112474

Contents lists available at ScienceDirect

Journal of Computational Physics

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

Efficient solution of bimaterial Riemann problems for compressible multi-material flow simulations

Wentao Ma, Xuning Zhao, Shafquat Islam, Aditya Narkhede, Kevin Wang *

Department of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA

ARTICLE INFO

Article history:
Received 14 March 2023
Received in revised form 7 August 2023
Accepted 29 August 2023
Available online 4 September 2023

Keywords:
Multiphase flow
Multi-material flow
Riemann problem
Equation of state
Compressible flow

ABSTRACT

When solving compressible multi-material flow problems, an unresolved challenge is the computation of advective fluxes across material interfaces that separate drastically different thermodynamic states and relations. A popular idea in this regard is to locally construct bimaterial Riemann problems, and to apply their exact solutions in flux computation. For general equations of state, however, finding the exact solution of a Riemann problem is expensive as it requires nested loops. Multiplied by the large number of Riemann problems constructed during a simulation, the computational cost often becomes prohibitive. The work presented in this paper aims to accelerate the solution of bimaterial Riemann problems without introducing approximations or offline precomputation tasks. The basic idea is to exploit some special properties of the Riemann problem equations, and to recycle previous solutions as much as possible. Following this idea, four acceleration methods are developed, including (1) a change of integration variable through rarefaction fans, (2) storing and reusing integration trajectory data, (3) step size adaptation, and (4) constructing an R-tree on the fly to generate initial guesses. The performance of these acceleration methods is assessed using four example problems in underwater explosion, laser-induced cavitation, and hypervelocity impact. These problems exhibit strong shock waves, large interface deformation, contact of multiple (> 2) interfaces, and interaction between gases and condensed matters. For all the problems, the acceleration methods are able to significantly reduce the computational cost without affecting solver robustness or solution accuracy. In different cases, the solution of bimaterial Riemann problems is accelerated by 37 to 87 times. As a result, the total cost of advective flux computation, which includes the exact Riemann problem solution at material interfaces and the numerical flux calculation over the entire computational domain, is accelerated by 18 to 81 times.

1. Introduction

Fluid flows involving multiple materials or phases arise in many scientific and engineering disciplines. In this paper, we consider a class of problems in which different materials (or phases) are separated by sharp interfaces, and the flow needs to be treated as being compressible. Many fluid dynamics problems involving bubbles, droplets, phase transitions, and chemical reactions fall into this category (e.g., [1–3]). Some problems in solid and soft matter mechanics also belong to this

* Corresponding author.
E-mail address: kevinwgy@vt.edu (K. Wang).

https://doi.org/10.1016/j.jcp.2023.112474
0021-9991/

category, when the material behaves like a compressible fluid due to some special loading condition (e.g., [4,5]). Because of compressibility, the physical model typically involves a thermodynamic equation of state (EOS) for each material. Across a material interface, discontinuity arises not only in some state variables (e.g. density, internal energy), but also in the EOS. These discontinuities can be quite significant, if the interface separates gaseous and condensed phases.

Different numerical methods have been developed to represent material interfaces. In Lagrangian [6,7] and Arbitrary Lagrangian-Eulerian (ALE) [8,9] frameworks, the computational mesh is updated to conform to the material interface. In the Eulerian framework, the material interface is tracked using numerical methods such as interface capturing [10,11], level set [12], volume of fluid [13], and front tracking [14]. Regardless of the method used to represent interfaces, the spatial discretization of the governing equations (e.g. Navier-Stokes equations) at an interface presents new challenges not encountered in single-phase flow simulations. Many popular numerical flux functions (e.g. Roe, HLLC, HLLE) assume a single EOS. Therefore, they cannot be directly applied at material interfaces. It is also well known that spurious oscillations often occur near material interfaces, even if the applied numerical scheme performs well for single-phase flows (e.g., [15,16]). For example, Brummelen et al. showed that when an interface capturing method is used, spurious pressure oscillation may occur due to the loss of pressure-invariance property in discretization [17]. Fedkiw et al. developed the ghost fluid method, in which ghost states on both sides of a material interface are constructed [18]. In order to suppress an "overheating" phenomenon, one needs to apply the "isobaric fixing" technique that requires solving another auxiliary partial differential equation [18,19]. Some authors have also reported that the original ghost fluid method may produce inaccurate results when an interface is hit by a shock wave, or the density or EOS jumps significantly across it [20,15,21].

In the past two decades, many authors have proposed the idea of constructing bimaterial Riemann problems across material interfaces, and applying their exact solutions in flux computation [15,22,23,21,24–26,11,27–29,3,30]. For example, Liu et al. incorporated exact Riemann solvers in their ghost fluid method, in order to correctly predict the double rarefaction waves in nearly cavitating flow conditions [15]. Later, Wang et al. used the exact Riemann solutions to populate not only ghost cells but also the real fluid cells near material interfaces, thereby eliminating the need of isobaric fix [22]. Cocchi and Saurel developed a prediction-correction scheme in which the solutions of exact Riemann problems are used to correct flow states near contact discontinuities [25]. Igra and Takayama modified this method to make it work in conjunction with the level-set method [26]. Also, Kitamura et al. conducted a comparative study of the AUSM (Advection Upstream Splitting Method) family of schemes, and found that when combined with the exact Riemann problem solver, AUSM schemes are applicable to more challenging problems with larger pressure ratios [11]. Recently, Farhat et al. developed the FIVER (FInite Volume method with Exact multiphase Riemann solvers) method, in which the exact solutions of bimaterial Riemann problems are used directly to compute advective fluxes, instead of populating ghost or real fluid cells [27,28,31–33]. FIVER has been applied to study several shock-dominated multiphase flow and fluid-structure interaction problems, including underwater explosion and implosion [34,35], pipeline explosion [31], cavitation erosion [36–39], laser-induced vaporization [1], and hypervelocity impact [5]. Furthermore, some authors have proposed to extend exact Riemann problem solvers to account for additional interfacial physics, such as surface tension and phase transition [29,3,30].

A unique advantage of the constructed bimaterial Riemann problems is that they account for not only the discontinuity of state variables (e.g. density, internal energy) across a material interface, but also the change of thermodynamic EOS. With this in mind, one may expect this idea to be more impactful for problems that require sophisticated, highly nonlinear EOS. In reality, however, most of the efforts reviewed above deal with simple EOS, such as perfect and stiffened gases. For general EOS, computational cost becomes a major issue, as finding the exact solution of a Riemann problem requires nested loops [40,41]. In particular, when the solution contains a rarefaction fan, each iteration in the outer loop requires numerically integrating two state variables through the fan, with the upper bound of the integral being an unknown variable [41]. As a result, finding the exact solution of a Riemann problem often requires evaluating the EOS thousands of times. In contrast, approximate Riemann solvers (e.g. Roe, HLLC) only call the EOS a few times. Therefore, although the exact Riemann problem solver is only applied within a lower-dimensional subset (i.e., material interfaces) of the computational domain, its cost can be much higher than the approximate Riemann solver that is applied in the entire domain. This dramatic increase of computational cost often makes the simulation unfeasible.

In this work, our objective is to accelerate the solution of bimaterial Riemann problems without introducing approximations or requiring additional precomputation. Our basic idea is to exploit some special properties of the Riemann problem equations, and to recycle previous solutions (obtained during the same simulation) as much as possible. Following this idea, we present four acceleration methods that can be used either in combination or separately. Three of the four methods are designed to accelerate the numerical integration through rarefaction fans, as it is the most expensive part of the solution procedure. We show that through a change of integration variable, the upper bound of the integral can be determined. We also show that the integral curve (i.e., isentrope) is uniquely determined by the inputs of each constructed Riemann problem. Therefore, the numerical integration performed within each iteration of the solution procedure is along the same curve, only with a different end point. In light of this observation, we present a method that eliminates repeated calculations by storing and reusing previous integration trajectories. The conventional solution procedure uses a fixed integration step size. We show that the nonlinearity of the EOS and the high degree of automation needed here allow us to benefit from step size adaptation. Furthermore, the cost of the exact Riemann problem solver also depends on the quality of the initial guesses. The fourth acceleration method is designed to generate better initial guesses, by storing previous Riemann solutions in a five-dimensional R-tree [42,43] and finding initial guesses through nearest-neighbor search. To assess the performance of these acceleration methods, we solve four example problems in underwater explosion, laser-induced cavitation,

and hypervelocity impact. These problems exhibit strong shock waves, large interface deformation, contact of multiple (> 2) interfaces, and interaction between gases and condensed matters. The robustness, accuracy, and computational efficiency of the acceleration methods are put to the test.

A major difference between the current work and previous acceleration efforts (e.g., [44,15,45,28]) is that here we do not introduce approximations or require any offline precomputation. Previously, Colella and Glaz accelerated single-material Riemann problem solvers by developing a local parametrization for the EOS [44]. This method assumes slow variation of the thermodynamic state, which may not be true in challenging multi-material flow problems. Liu et al. found that their ghost fluid method equipped with a specifically designed approximate two-phase Riemann problem solver cannot provide correct solutions under nearly cavitating flow conditions [15]. Exact solutions of Riemann problems were thus employed to address the two strong rarefaction waves in this situation. Farhat et al. avoided online numerical integration by tabulating and interpolating the Riemann invariants on sparse grids [45,28]. For the problem they considered, which involves the stiffened gas EOS and the Jones-Wilkins-Lee (JWL) EOS, only one two-dimensional sparse grid needs to be constructed. But for arbitrary EOS, two three-dimensional sparse grids need to be constructed for each material. This offline precomputation can be expensive. It also requires a high level of expertise from the user, who is expected to specify, before the simulation, the resolution of the sparse grids as well as the lower and upper bounds of each tabulated variable. These complexities motivated us to pursue a different path, that is, to accelerate the Riemann solver without introducing approximations or requiring offline precomputation.

The remainder of this paper is organized as follows. In Sec. 2, we present the governing compressible Navier-Stokes equations and the bimaterial Riemann problem with general convex and smooth EOS. We also briefly describe the aforementioned FIVER method, which is adopted in this work to perform numerical tests. In Sec. 3, we discuss the computational efficiency of the exact Riemann problem solver in detail. The acceleration methods mentioned above are presented. In Sec. 4, we present the numerical tests and discuss the performance of the acceleration methods. Finally, we provide a summary in Sec. 5.

2. Governing equations and solution framework

2.1. Navier-Stokes equations with general equations of state

In general, the dynamics of compressible multi-material flows are governed by Navier-Stokes equations, which enforce the conservation of mass, momentum, and energy. In the literature, the exact solutions of bimaterial Riemann problems are used mostly to compute the advective fluxes (e.g., [28,3]). Therefore, we write the Navier-Stokes equations as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \mathcal{F}(\boldsymbol{U}) = \boldsymbol{\mathcal{S}}, \qquad \text{in } \Omega \times (0,\ t_{\max}], \tag{1}$$

which emphasizes the advective fluxes $\mathcal{F}(\boldsymbol{U})$. Here, $\boldsymbol{U}$ denotes the conservative state vector. t denotes time, and $\boldsymbol{\mathcal{S}}$ represents collectively the additional terms that may need to be included in the conservation laws, depending on the specific flow being analyzed. Examples include viscous fluxes, heat diffusion fluxes, heat sources, and body forces. Because bimaterial Riemann problems are neither constructed based on these terms nor used to evaluate them, we do not formulate $\boldsymbol{\mathcal{S}}$ explicitly. Ω denotes the spatial domain of the flow, and $t_{\max}$ the maximum physical time of interest. $\boldsymbol{U}$ and $\mathcal{F}(\boldsymbol{U})$ are given by

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho e_t \end{bmatrix}, \quad \mathcal{F}(\boldsymbol{U}) = \begin{bmatrix} \rho \boldsymbol{u}^T \\ \rho \boldsymbol{u} \otimes \boldsymbol{u} + p \boldsymbol{I} \\ (\rho e_t + p) \boldsymbol{u}^T \end{bmatrix}, \tag{2}$$

where ρ, e_t, and p denote the density, total energy per unit mass, and pressure, respectively. $\boldsymbol{u} = [u, v, w]^T$ is the velocity vector. $\boldsymbol{I}$ denotes the 3×3 identity matrix. In addition,

$$e_t = e + \frac{1}{2}|\boldsymbol{u}|^2, \tag{3}$$

where e is the internal energy per unit mass.

We assume that the fluid domain Ω consists of multiple non-overlapping open sets (i.e. subdomains) occupied by different materials, i.e.,

$$\Omega = \bigcup_{m=1}^{M} \mathrm{cl}(\Omega_m), \quad \Omega_m \cap \Omega_n = \emptyset,\ \forall m,\ n\ (m \neq n), \tag{4}$$

where $\mathrm{cl}(\Omega_m)$ denotes the closure of Ω_m. Within each subdomain Ω_m, a thermodynamic equation of state (EOS) of the form

$$p = p(\rho, e) \tag{5}$$

is needed to close the governing equations. The speed of sound, c, is given by

$$c = \sqrt{\frac{\partial p}{\partial \rho}\Big|_e + \frac{p}{\rho^2}\frac{\partial p}{\partial e}\Big|_\rho}. \tag{6}$$

Here, we introduce a few specific examples that are used in the numerical tests in Sec. 4. Nonetheless, the algorithms presented in this paper are designed to be applicable to general convex and smooth equations of state in the form of (5).

As a generalization of perfect gas, the noble-Abel stiffened gas equation of state [46] is given by

$$p = (\gamma - 1)\frac{\rho(e - e_c)}{1 - \rho b} - \gamma p_c, \tag{7}$$

where γ, e_c, b, and p_c are constant parameters. The sound speed is given by

$$c\,(p, \rho) = \sqrt{\frac{\gamma(p + p_c)}{\rho - \rho^2 b}}. \tag{8}$$

The Jones-Wilkins-Lee (JWL) equation of state is widely used to model the thermodynamics of detonation products [47]. It has the expression

$$p = \omega\rho e + f(\rho), \tag{9}$$

where

$$f(\rho) = A_1\left(1 - \frac{\omega\rho}{R_1\rho_0}\right)\exp\Big(-\frac{R_1\rho_0}{\rho}\Big) + A_2\left(1 - \frac{\omega\rho}{R_2\rho_0}\right)\exp\Big(-\frac{R_2\rho_0}{\rho}\Big). \tag{10}$$

Here, ρ_0, ω, A_1, A_2, R_1, and R_2 are constant parameters. The sound speed is given by

$$c(p, \rho) = \sqrt{\frac{\gamma p - f(\rho) + \rho f'(\rho)}{\rho}}. \tag{11}$$

The Mie-Grüneisen equation of state is often used to model solids and liquids under high pressure. In this work, we adopt the formulation given in [48], i.e.,

$$p = \frac{\rho_0 c_0^2 \eta}{(1 - s\eta)^2}\Big(1 - \frac{1}{2}\Gamma_0\eta\Big) + \rho_0\Gamma_0 e, \tag{12}$$

with $\eta = 1 - \rho_0/\rho$. ρ_0 and c_0 denote the density and bulk speed of sound in the ambient condition. s is the slope of the Hugoniot curve. Γ_0 is the Grüneisen parameter in the ambient condition. In this case, the sound speed is given by

$$c(p, \rho) = \sqrt{\frac{\rho_0^2 c_0^2}{\rho^2}\frac{1 + (s - \Gamma_0)\eta}{(1 - s\eta)^3} + \frac{p}{\rho^2}\Gamma_0\rho_0}. \tag{13}$$

For these examples, both e and c can be evaluated analytically in closed-form. There are more complicated EOS for which one or both of them can only be evaluated through numerical iteration (e.g. [49]). As a result, the computational cost may increase drastically.

If the term $\boldsymbol{\mathcal{S}}$ in (1) involves temperature, a temperature law is also needed for each material. However, the temperature law is not involved in the construction and solution of exact Riemann problems. Therefore, additional details are omitted for the sake of conciseness.

2.2. Formulation of bimaterial Riemann problems at material interfaces

At the interface between two adjacent material subdomains, normal velocity and pressure are assumed to be continuous, i.e.,

$$\begin{aligned}(\boldsymbol{u}_l - \boldsymbol{u}_r)\cdot\boldsymbol{n} &= 0,\\ p_l - p_r &= 0,\end{aligned} \qquad \text{on } \partial\Omega_l \cap \partial\Omega_r, \tag{14}$$

where $\boldsymbol{n}$ denotes the unit normal to the material interface, and the subscripts l and r indicate the one-sided limits of the state variable ($\boldsymbol{u}$ or p) obtained within the two subdomains. At each point on the interface, the bimaterial Riemann problem is defined as the 1D Euler equations [25,26,15,23,21,24,11,27,28],

$$\frac{\partial \boldsymbol{q}}{\partial \tau} + \frac{\partial \mathcal{F}(\boldsymbol{q})}{\partial \xi} = 0, \tag{15}$$

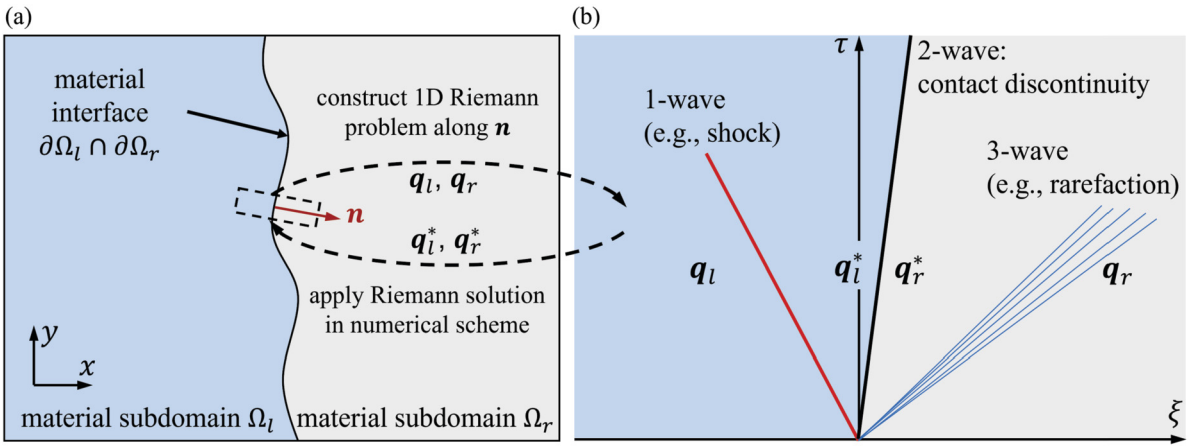

Fig. 1. Construction and solution of a bimaterial Riemann problem at material interface.

with a piecewise constant initial condition obtained from the original 2D or 3D flow field (Fig. 1). Here, τ is a fictitious time coordinate, ξ is the spatial coordinate, which in principle is along the direction of $\boldsymbol{n}$. The initial condition, which features a discontinuity at the material interface, is given by

$$\boldsymbol{q}(\xi, 0) = \begin{cases} \boldsymbol{q}_l, & \text{if } \xi \le 0, \\ \boldsymbol{q}_r, & \text{if } \xi > 0, \end{cases} \tag{16}$$

with

$$\boldsymbol{q}_l \equiv \begin{bmatrix} \rho_l \\ \rho_l u_l \\ \rho_l e_{tl} \end{bmatrix} = \begin{bmatrix} \rho_l \\ \rho_l (\boldsymbol{u}_l \cdot \boldsymbol{n}) \\ \rho_l e_l + \frac{1}{2}\rho_l (\boldsymbol{u}_l \cdot \boldsymbol{n})^2 \end{bmatrix}, \quad \boldsymbol{q}_r \equiv \begin{bmatrix} \rho_r \\ \rho_r u_r \\ \rho_r e_{tr} \end{bmatrix} = \begin{bmatrix} \rho_r \\ \rho_r (\boldsymbol{u}_r \cdot \boldsymbol{n}) \\ \rho_r e_r + \frac{1}{2}\rho_r (\boldsymbol{u}_r \cdot \boldsymbol{n})^2 \end{bmatrix}. \tag{17}$$

Solving the Riemann problem exactly generally means finding a weak entropy solution that consists of three waves, as illustrated in Fig. 1(b). The one in the middle (i.e., the 2-wave) is a contact discontinuity that tracks the motion of the material interface. It separates the two material subdomains in which different EOS should be applied. Each of the other two waves can be either a shock wave or a rarefaction fan. These three waves divide the ξ-τ space into four regions, plus the interior of any rarefaction fan. The outer two regions have not been affected by the shock and/or rarefaction yet, so the initial fluid states $\boldsymbol{q}_l$ and $\boldsymbol{q}_r$ sill hold there (Fig. 1(b)). The state variables in the two middle regions next to the material interface are unknowns that need to be solved for. Following the convention in the literature ("star states"), these variables are identified by a superscript $*$. It should be noted that the aforementioned three-wave solution structure requires the EOS to be convex and smooth [50]. Given the convexity and smoothness of the EOS, the Riemann solution is unique if the EOS in the meantime satisfies the "medium condition" in [50].

Across the contact discontinuity, pressure and velocity are continuous, i.e.,

$$\begin{aligned} p_l^* &= p_r^*, \\ u_l^* &= u_r^*. \end{aligned} \tag{18}$$

Therefore, the solution of the exact Riemann problem automatically satisfies the interface conditions (14).

Across a shock wave, the Rankine-Hugoniot jump conditions can be derived from (15). After some algebraic manipulations, we get the following two equations.

$$e_K(p_K^*, \rho_K^*) - e_K(p_K, \rho_K) + \frac{1}{2}(p_K + p_K^*)\left(\frac{1}{\rho_K^*} - \frac{1}{\rho_K}\right) = 0 \tag{19}$$

and

$$u_K^* = u_K \mp \sqrt{-(p_K^* - p_K)\left(\frac{1}{\rho_K^*} - \frac{1}{\rho_K}\right)}, \tag{20}$$

where $K = l, r$. Equation (19) only involves the thermodynamic variables, and is known as the Hugoniot equation. The function $e_K(p, \rho)$ evaluates internal energy using the EOS that holds in material subdomain Ω_K. In (20), the minus sign holds for $K = l$, and the plus sign holds for $K = r$.

Through a rarefaction fan, two Riemann invariants can be derived from the 1D Euler equations (15). They are

$$s - s_K = 0 \tag{21}$$

and

$$u - u_K \pm \int_{\rho_K}^{\rho} \frac{c_K(s_K, \rho)}{\rho} d\rho = 0, \tag{22}$$

where s, u, ρ denote the entropy, velocity, and density evaluated at any point within a rarefaction fan, including its boundaries. In (22), the plus sign applies to the 1-wave (i.e., $K = l$), and the minus sign applies to the 3-wave (i.e., $K = r$). The first Riemann invariant (21) indicates that entropy is constant through a rarefaction fan. Therefore, the sound speed can be evaluated from $c^2 = \frac{\partial p}{\partial \rho}\Big|_s$. Integrating this relation through the rarefaction fan, i.e., from the ambient state to the star state, gives

$$p_K^* = p_K + \int_{\rho_K}^{\rho_K^*} c_K^2(s_K, \rho) d\rho, \tag{23}$$

where the function $c_K(s, \rho)$ evaluates the sound speed using the EOS for material K. Similarly, substituting $\rho = \rho_K^*$ and $u = u_K^*$ into (22) yields

$$u_K^* = u_K \mp \int_{\rho_K}^{\rho_K^*} \frac{c_K(s_K, \rho)}{\rho} d\rho, \tag{24}$$

where the minus sign applies to the 1-wave (i.e., $K = l$) and the plus sign applies to the 3-wave (i.e., $K = r$). Although these two equations involve entropy in the evaluation of sound speed, the actual computation does not necessarily need it. We can write the sound speed c as a function of p and ρ, as given in equations (8), (11), and (13).

To determine whether each of the 1 and 3 waves is a shock or a rarefaction, we enforce an entropy condition. Each of the two waves is determined to be a shock wave if and only if $p_K^* > p_K$, where $K = l$ for the 1-wave, and $K = r$ for the 3-wave.

In summary, the solution of the exact Riemann problem, $\boldsymbol{q}_l^*$ and $\boldsymbol{q}_r^*$, is determined by Eqs. (18), (19), (20), (23), (24), and the above entropy condition. The shock speed, u_s, and the state variables through any rarefaction fan can be easily derived based on $\boldsymbol{q}_l^*$ and $\boldsymbol{q}_r^*$. For a general equation of state, the exact Riemann problem cannot be solved analytically. The numerical solution methods are presented in Sec. 3 in detail.

2.3. Interface tracking and discretization of governing equations

We briefly describe the interface tracking and spatial discretization methods adopted in this work, thereby providing a clear context for the construction and solution of bimaterial Riemann problems. Nonetheless, the algorithms for solving the Riemann problems — to be introduced in Sec. 3 — are not restricted to these methods.

For a problem that involves M (≥ 2) materials, there can be as many as $C_2^M = M(M-1)/2$ material interfaces. Instead of tracking each one of them explicitly, we solve $M-1$ level set equations, i.e.

$$\frac{\partial \phi^{(m)}(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{u} \cdot \nabla \phi^{(m)} = 0, \quad m = 1, 2, \cdots M-1, \tag{25}$$

to track the boundaries of the first $M-1$ subdomains (Fig. 2). Here, $\phi^{(m)}(\boldsymbol{x}, t)$ denotes a level set function, initialized to be the signed shortest distance from $\boldsymbol{x}$ to $\partial\Omega_m$. Any point $\boldsymbol{x} \in \Omega$ is within subdomain Ω_m ($m = 1, 2, \cdots M-1$) at time t if and only if $\phi^{(m)}(\boldsymbol{x}, t) < 0$. The last subdomain is simply

$$\Omega_M = \Omega \setminus \Big(\bigcup_{m=1}^{M-1} \mathrm{cl}(\Omega_m) \Big). \tag{26}$$

Notably, all the level set equations share the same velocity field $\boldsymbol{u}$, which prevents spurious overlapping and separation of subdomains. The interface between each pair of materials — e.g. m and n ($1 \leq m < n \leq M$) — is simply given by

$$\partial\Omega_m \cap \partial\Omega_n = \begin{cases} \Big\{ \boldsymbol{x} \in \Omega, \ \ \phi^{(m)}(\boldsymbol{x}, t) = \phi^{(n)}(\boldsymbol{x}, t) = 0 \Big\}, & n < M, \\ \Big\{ \boldsymbol{x} \in \Omega, \ \ \phi^{(k)}(\boldsymbol{x}, t) = 0 \ \text{ iff } \ k = m \Big\}, & n = M. \end{cases} \tag{27}$$

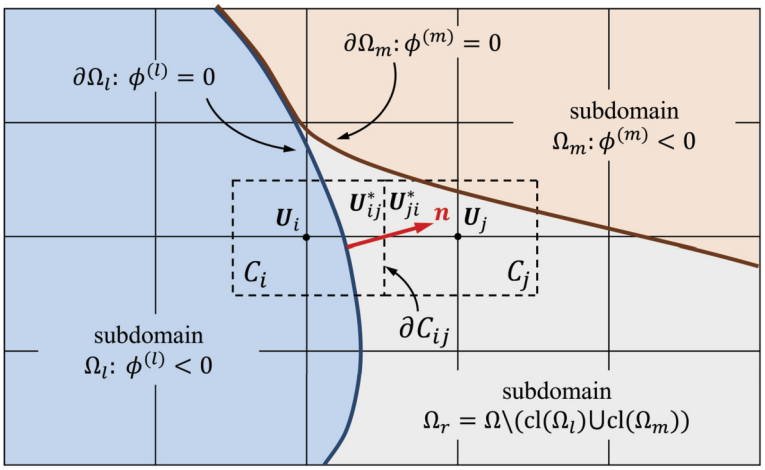

Fig. 2. Interface tracking and discretization of governing equations: The level set method and FIVER.

We discretize the governing equations (1) using the FIVER method [51,1]. Applying a standard finite volume method to the left hand side of (1) yields

$$\frac{\partial \boldsymbol{U}_i}{\partial t} + \frac{1}{\|C_i\|} \sum_{j \in N(i)} \int_{\partial C_{ij}} \mathcal{F}(\boldsymbol{U}) \cdot \boldsymbol{n}_{ij} dS = \boldsymbol{\mathcal{S}}^h, \tag{28}$$

where C_i denotes a control volume in Ω, $\|C_i\|$ being its volume. $\boldsymbol{U}_i$ denotes the cell-average of $\boldsymbol{U}$. $N(i)$ is the set of nodes that are connected to node i by an edge. $\partial C_{ij} = \partial C_i \cap \partial C_j$ is the cell interface between C_i and C_j. $\boldsymbol{\mathcal{S}}^h$ represents a numerical approximation of $\boldsymbol{\mathcal{S}}$, which does not need to be based on the same finite volume method. Between cell C_i and each neighboring cell C_j, a numerical advective flux is computed to approximate the surface integral in (28). If i and j belong to the same material subdomain, a conventional flux function (e.g. local Lax-Friedrichs (LLF), Roe, HLLC) is directly applied. If they belong to different material subdomains (e.g. Ω_l and Ω_r in Fig. 2), a 1D bimaterial Riemann problem is constructed using the method described in Sec. 2.2. Its solution, $\boldsymbol{q}^*_l$ and $\boldsymbol{q}^*_r$, are mapped back to 2D or 3D, generating interface states U^*_{ij} and U^*_{ji}. Then, the advective fluxes on the two sides of ∂C_{ij} are computed as

$$\begin{aligned} F_{ij} &= \left\|\partial C_{ij}\right\| \Phi\left(\boldsymbol{U}_{ij}, \boldsymbol{U}^*_{ij}, n_{ij}, \mathrm{EOS}(\Omega_l)\right) \\ F_{ji} &= \left\|\partial C_{ij}\right\| \Phi\left(\boldsymbol{U}_{ji}, \boldsymbol{U}^*_{ji}, n_{ji}, \mathrm{EOS}(\Omega_r)\right), \end{aligned} \tag{29}$$

where $\left\|\partial C_{ij}\right\|$ is the area of ∂C_{ij}, and Φ denotes a conventional numerical flux function. $\boldsymbol{U}_{ij}$ and $\boldsymbol{U}_{ji}$ are interface states reconstructed using the MUSCL (monotonic upwinding scheme for conservation laws) method and a slope limiter. Additional details about FIVER can be found in [51,1].

3. Efficient solution of bimaterial Riemann problems

3.1. Conventional solution procedure and its efficiency issues

We start with summarizing the conventional method for solving Riemann problems (e.g., [40,41]), while at the same time, emphasizing its high computational cost. This cost may not be a practical issue if the user only needs to solve one or a few Riemann problems. Nonetheless, it can dominate the entire simulation cost, when bimaterial Riemann problems are constructed and solved along each edge in the mesh that penetrates a material interface, although the union of all the interfaces is still a lower-dimensional subset of the computational domain.

The equations that govern the three-wave entropy solution of the Riemann problem are Eqs. (18) to (24), together with the entropy condition described in Sec. 2.2. The unknowns are p^* $(= p^*_l = p^*_r)$, u^* $(= u^*_l = u^*_r)$, ρ^*_l, and ρ^*_r. We define function $f(p)$ as

$$f(p) = u^*_l(p;\ \rho_l, u_l, p_l, \mathrm{EOS}_l) - u^*_r(p;\ \rho_r, u_r, p_r, \mathrm{EOS}_r), \tag{30}$$

where the subscripts l and r indicate the material subdomains separated by the interface. For an arbitrary input p, the first term of $f(p)$ evaluates the interface velocity u^*_l that complies with an interface pressure $p^* = p$, and the left ambient state ρ_l, u_l, and p_l. Specifically, the entropy condition is first examined to determine whether the 1-wave is a shock or a

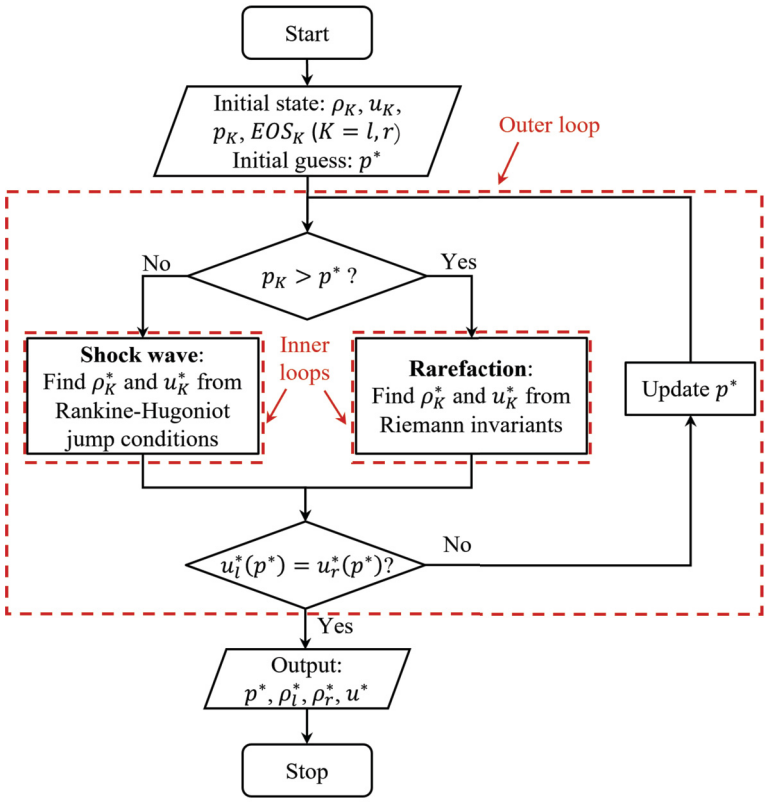

Fig. 3. The conventional procedure for solving bimaterial Riemann problems.

rarefaction. If it is a shock, Eqs. (19) and (20) are solved for u_l^*, with $K = l$ and $p_K^* = p$. If it is a rarefaction, Eqs. (23) and (24) are solved for u_l^*, also with $K = l$ and $p_K^* = p$. Similarly, the second term of $f(p)$ evaluates the interface velocity u_r^* that complies with an interface pressure $p^* = p$, and the right ambient state ρ_r, u_r, and p_r. Again, it starts with enforcing the entropy condition, followed by the solution of either (19) and (20), or (23) and (24), depending on whether the 3-wave is a shock or a rarefaction. Because of (18), the true solution of interface pressure, p^*, is a root of $f(p)$, i.e.

$$f(p^*) = 0. \tag{31}$$

Equation (31) is nonlinear, and for general equations of state it cannot be solved analytically. The standard method is to solve it numerically through an iterative procedure, such as the one illustrated in Fig. 3. Starting with an initial guess of p^*, in each iteration, we first apply the entropy condition to determine whether each of the 1 and 3 waves is a shock or a rarefaction. If it is a shock wave, ρ_K^* can be first obtained by solving Eq. (19) using an iterative method (e.g. the secant method). Then, u_K^* can be obtained using Eq. (20).

If it is a rarefaction fan, u_K^* and ρ_K^* are computed from Eqs. (23) and (24). For simple equations of state such as perfect gas, the integrals therein can be evaluated analytically in closed-form (e.g. see [28]). However, for an arbitrary equation of state, the two integrals need to be evaluated using a numerical integration method. For example, using the fourth-order Runge-Kutta method, in each step, the pressure and velocity are evaluated as

$$p^{(i+1)} = p^{(i)} + \frac{1}{6}\left[\hat{c}_1^2 + 2\left(\hat{c}_2^2 + \hat{c}_3^2\right) + \hat{c}_4^2\right]\Delta\rho, \tag{32}$$

$$u^{(i+1)} = u^{(i)} \mp \frac{1}{6}\left[\frac{\hat{c}_1}{\hat{\rho}_1} + 2\left(\frac{\hat{c}_2}{\hat{\rho}_2} + \frac{\hat{c}_3}{\hat{\rho}_3}\right) + \frac{\hat{c}_4}{\hat{\rho}_4}\right]\Delta\rho, \tag{33}$$

where $p^{(i)}$ and $u^{(i)}$ are the pressure and density evaluated from the previous step. In the second equation, the minus sign applies to the 1-wave, and the plus sign applies to the 3-wave. $\hat{c}_i$ and $\hat{\rho}_i$ $(i = 1, 2, 3, 4)$ are sound speed and density evaluated at intermediate stages within the step, specifically,

Table 1
Number of EOS evaluations required by exact and approximate Riemann problem solvers.

	Exact Riemann solver	Approx. Riemann solver		
		LLF	Roe	HLLC
Shock ($=2N_{\text{out}}N_{\text{sh}}$)	50	–	–	–
Rarefaction ($=4N_{\text{out}}N_{\text{rf}}$)	4000	–	–	–
Total*	4050	4	3	5

* Assuming one shock wave and one rarefaction fan in the solution structure (Fig. 1(b)).

$$\hat{c}_1 = c_K\left(p^{(i)},\ \hat{\rho}_1\right),\ \text{with}\ \hat{\rho}_1 = \rho^{(i)},$$
$$\hat{c}_2 = c_K\left(p^{(i)} + 0.5\Delta\rho\hat{c}_1^2,\ \hat{\rho}_2\right),\ \text{with}\ \hat{\rho}_2 = \rho^{(i)} + 0.5\Delta\rho,$$
$$\hat{c}_3 = c_K\left(p^{(i)} + 0.5\Delta\rho\hat{c}_2^2,\ \hat{\rho}_3\right),\ \text{with}\ \hat{\rho}_3 = \rho^{(i)} + 0.5\Delta\rho,$$
$$\hat{c}_4 = c_K\left(p^{(i)} + \Delta\rho\hat{c}_3^2,\ \hat{\rho}_4\right),\ \text{with}\ \hat{\rho}_4 = \rho^{(i)} + \Delta\rho.$$

It should be noted that in Eq. (23), the unknown variable is the integral's upper bound, ρ_K^*, while the integration result (p^*) on the left hand side is given. In order to find ρ_K^* that yields the given p^*, a trial-and-error procedure is needed. For example, Kamm [41] proposed an iterative procedure to find a constant $\Delta\rho$ that would allow the numerical integration to end at p_K^*. In each iteration, the entire course of integration is repeated with an updated $\Delta\rho$. A more efficient method is to update $\Delta\rho$ in each step of the numerical integration process, based on the difference between the current pressure $p^{(i)}$ and the given p^*.[1] Either way, the computational cost is higher than standard numerical integration, where the lower and upper bounds of the integral are both given.

If the computed interface velocities, u_l^* and u_r^*, satisfy

$$\left|u_l^*(p^*) - u_r^*(p^*)\right| < \epsilon_u, \tag{34}$$

an approximate solution of p^* has been obtained. Here, ϵ_u denotes the error tolerance. ρ_l^* and ρ_r^* have also been obtained through the solution of the Hugoniot equation (19) and/or the Riemann invariant (23). u^* can be set by

$$u^* = \frac{1}{2}\left(u_l^* + u_r^*\right). \tag{35}$$

The propagation speed of any shock wave can be calculated easily using the Rankine-Hugoniot jump conditions. If needed, the solution of ρ, u, and p within any rarefaction fan can be obtained from the numerical integration procedure as a byproduct.

If (34) is not satisfied, the nonlinear equation solver (i.e., the outer loop in Fig. 3) enters the next iteration with an updated guess of p^*, which can be obtained using the secant method.

Compared to approximate Riemann problem solvers, such as LLF [52], Roe and HLLC [53], the cost of the exact Riemann problem solver presented above is significantly higher, as it involves nested loops (Fig. 3). This difference can be examined by counting and comparing the number of times the EOS are evaluated. For the exact Riemann problem solver, let N_{out} denote the number of iterations in the outer loop. Let N_{sh} be the average number of iterations in the inner loop for solving the shock Hugoniot equation. Let N_{rf} be the number of integration steps required to go through any rarefaction fan. The number of EOS evaluations — including the evaluation of c, e, and p — is 2 within each step of the shock Hugoniot equation solver, and 4 within each integration step. Therefore, executing the algorithm shown in Fig. 3 requires at least

$$N_{\text{exact}} = N_{\text{out}}(2N_{\text{sh}} + 4N_{\text{rf}}) \tag{36}$$

calls to the EOS, if we assume the solution structure consists of one shock wave and one rarefaction fan. In comparison, the LLF, Roe, and HLLC solvers require 4, 3, and 5 calls to the EOS, respectively. Table 1 shows a typical scenario with $N_{\text{out}} = N_{\text{sh}} = 5$ and $N_{\text{rf}} = 200$, in which the cost of the exact Riemann problem solver is roughly 1000 times that of approximate Riemann problem solvers, in terms of the number of EOS evaluations.

The exact Riemann problem solver is only applied across or near material interfaces, which is a lower-dimensional subset of the computational domain. In comparison, approximate Riemann solvers are generally applied within the entire domain. Although the number of times the exact Riemann problem solver is used is smaller, its cost can still dominate the entire simulation cost. As an example, consider a square domain in 2D, discretized uniformly into n elements in both directions. The total number of edges along which numerical fluxes need to be computed is roughly $2n^2$. Let us assume that $2n$ of

[1] This method is adopted in the baseline simulations presented in Sec. 4.

them penetrate material interfaces. In this case, the ratio between the number of approximate Riemann problems and that of exact Riemann problems is approximately n. Based on the statistics shown in Table 1, for $n = 200$, exact Riemann problem solutions account for more than 80% of the computational cost, measured by the number of EOS evaluations.

In numerical tests, we have found that during flux calculations, over 95% of the computation time can be consumed by the exact Riemann problem solver. This number is higher than the estimate obtained above, for at least two reasons. First, in some cases, the numerical integration procedure requires more than 200 steps (i.e., $N_{rf} > 200$) to reach a sufficient degree of accuracy (e.g. see Fig. 24 in Sec. 4.4). Second, in parallel computations, the mesh is usually partitioned without knowledge of the location of material interfaces, which results in load imbalance. Some processors may need to solve a larger number of exact Riemann problems, if the subdomain assigned to it covers a larger fraction of material interface. For the most burdened processor, the ratio of approximate to exact Riemann problems is often lower than 200, which was assumed in the above estimation.

Next, we propose several techniques that can significantly reduce the computational cost of the exact Riemann problem solver, without sacrificing accuracy or introducing precomputation efforts (e.g. sparse-grid tabulation [28]).

3.2. Change of integration variable

As mentioned in Sec. 3.1, when evaluating the integrals in (23) and (24), the upper bound ρ_K^* is unknown. A trial-and-error procedure is required to find the correct ρ_K^* that makes Eq. (23) hold for the given p^*. We show that this trial-and-error procedure can be eliminated by rewriting (23) and (24) in an alternative form, in which the integration variable is p instead of ρ. Then, the upper bound of the integral becomes p^*, which is specified within the outer loop.

The two Riemann invariants, (23) and (24), can be expressed in a differential form,

$$\begin{cases} \dfrac{dp}{d\rho} &= c_K^2(s_K,\rho), \\ \dfrac{du}{d\rho} &= \mp\dfrac{c_K(s_K,\rho)}{\rho}, \\ p(\rho_K) &= p_K, \\ u(\rho_K) &= u_K, \end{cases} \tag{37}$$

for all $\rho \in [\rho^*, \rho_K]$ and $p \in [p^*, p_K]$. In the second equation, the minus sign applies to the 1-wave ($K = l$), and the plus sign applies to the 3-wave ($K = r$). Through the rarefaction fan, entropy does not change, so pressure $p(s_K, \rho)$ is essentially a single variable function of density, i.e., $p(\rho)$. Also, the first equation in (37) indicates that $p(\rho)$ increases monotonically with respect to ρ since its derivative is $c^2(s_K, \rho)$. Therefore, the function $p(\rho)$ has an inverse function $\rho(p)$. Its derivative, $\dfrac{d\rho}{dp}$, can be obtained easily from the first equation of (37), i.e.,

$$\frac{d\rho}{dp} = \frac{1}{dp/d\rho} = \frac{1}{c_K^2(s_K,\rho(p))} = \frac{1}{c_K^2(s_K,p)}. \tag{38}$$

Then, $\dfrac{du}{dp}$ can be derived from the second equation in (37) and (38) using the chain rule, i.e.,

$$\frac{du}{dp} = \frac{du}{d\rho}\frac{d\rho}{dp} = \mp\frac{c_K(s_K,p)}{\rho(p)}\frac{1}{c_K^2(s_K,p)} = \mp\frac{1}{c_K(s_K,p)\rho(p)}. \tag{39}$$

Using (38) and (39), (37) can be rewritten as

$$\begin{cases} \dfrac{d\rho}{dp} &= \dfrac{1}{c_K^2(s_K,p)}, \\ \dfrac{du}{dp} &= \mp\dfrac{1}{c_K(s_K,p)\rho(p)}, \\ \rho(p_K) &= \rho_K, \\ u(p_K) &= u_K, \end{cases} \tag{40}$$

for all $\rho \in [\rho^*, \rho_K]$ and $p \in [p^*, p_K]$. Integrating the ODEs in (40) from p_K to p^* yields

$$\rho_K^* = \rho_K + \int_{p_K}^{p^*} \frac{1}{c_K^2(s_K,p)}dp, \tag{41}$$

$$u_K^* = u_K \mp \int_{p_K}^{p^*} \frac{1}{c_K(s_K,p)\rho(p)}dp. \tag{42}$$

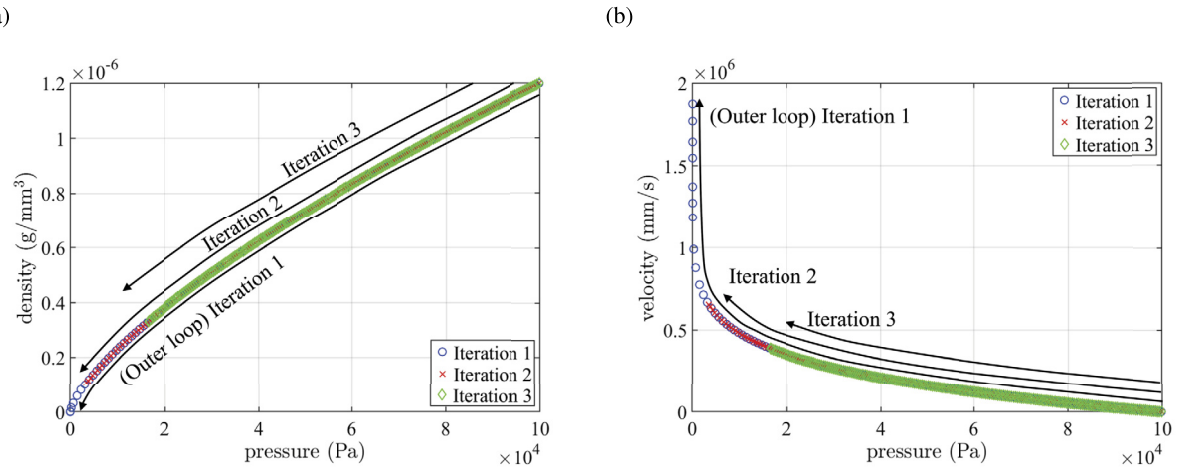

Fig. 4. Integration trajectories through the 1-wave of the example Riemann problem in three successive outer loop iterations.

Again, we can express sound speed c as a function of p and ρ. Hence, a closed-form formulation of entropy is not needed. We use (41) and (42) to replace (23) and (24). In this way, the integration interval is fixed, and the computational overhead associated with the aforementioned trial-and-error procedure is eliminated.

3.3. Storing and reusing integration trajectory data

We show that the integrations needed to resolve rarefaction fans, i.e. (41) and (42), can be accelerated by storing and reusing previous results. Our idea is based on the following two observations.

(1) Although the iterative solution of exact Riemann problems (i.e., the outer loop in Fig. 3) requires repeated evaluation of (41) and (42), the initial state of the integration is always the same, namely, $\{p_K, \rho_K, u_K\}$. Only p^* varies from one iteration to the next.
(2) Once p_K, ρ_K, and u_K are fixed, the integrands in (41) and (42) are both single variable functions of p.

(1) is true because the initial state, $\{p_K, \rho_K, u_K\}$, is determined by the construction of the Riemann problem, and the iterative solution procedure does not change it. (2) is true because entropy is constant through a rarefaction fan, and its value is determined uniquely by the initial state, $\{p_K, \rho_K, u_K\}$.

Indeed, throughout the solution procedure (i.e., the outer loop), we always integrate along the same isentrope, defined by the initial state, and characterized by a single variable, p. Only the final state varies in different iterations. Therefore, from one iteration to the next, the "overlapped" part of the integration trajectory is first evaluated, then discarded, and then re-evaluated again.

This redundancy can be shown clearly using an example problem. We consider a Riemann problem (i.e., (15)) with initial condition

$$(\rho, u, p, \text{EOS}) = \begin{cases} (1.2 \times 10^{-6}\ \text{g/mm}^3, 0.0\ \text{mm/s}, 1.0 \times 10^5\ \text{Pa, stiffened gas}) & \text{if} \quad \xi < 0\ \text{mm} \\ (2.204 \times 10^{-3}\ \text{g/mm}^3, 4.0 \times 10^5\ \text{mm/s}, 1.0 \times 10^5\ \text{Pa, Mie-Grüneisen}) & \text{if} \quad \xi > 0\ \text{mm}. \end{cases} \tag{43}$$

The parameters in the stiffened gas EOS are given by $\gamma = 1.4$, $e_c = 0$, $b = 0$, and $p_c = 0$. Therefore, it degenerates to a perfect gas. The parameters in the Mie-Grüneisen EOS are given by $p_0 = 2.204 \times 10^{-3}$ g/mm^3, $c_0 = 2.22 \times 10^6$ mm/s, $s = 1.61$, and $\Gamma_0 = 0.65$. The solution has a double-rarefaction wave structure. 200 steps are specified for the numerical integration.

Fig. 4 shows the integration trajectories through the 1-wave, obtained in 3 successive iterations in the outer loop. Clearly, these curves all start from the same point at $p = 1.0 \times 10^5$ Pa (the right end). They follow the same trajectory, only end at different points. It can be observed that the trajectory traversed in the first iteration is the same as that in the next two iterations. In iterations 2 and 3, only the last point (i.e., the end point) really needs to be evaluated. Nonetheless, the algorithm recalculates the entire trajectory from the right end. Therefore, the vast majority of the computations performed in iterations 2 and 3 are redundant and unnecessary.

To eliminate this redundancy, we propose to store and reuse the numerically obtained integration trajectory. Let $\boldsymbol{\chi}$ denote this trajectory, which is a discrete sequence of state variables $\{p, \rho, u\}$ on the isentrope. At the beginning of the outer loop (see Fig. 3), $\boldsymbol{\chi}$ is initialized to contain only the start point, $\{p_K, \rho_K, u_K\}$. In each iteration of the outer loop, the standard numerical integration procedure is replaced by Algorithm 1.

Algorithm 1 has two main steps. In Step 1, we find the point in $\boldsymbol{\chi}$ that is closest to the given p^*, but bigger. We start the numerical integration from this point, thereby avoiding redundant calculations as much as possible. In Step 2, we perform the numerical integration, and expand $\boldsymbol{\chi}$ with new points obtained on the isentrope.

Algorithm 1: Numerical integration of (41) and (42) with a stored trajectory.

Input: Start point $\{p_K, \rho_K, u_K\}$, end point $p^* < p_K$, and step size Δp.
Stored trajectory $\boldsymbol{\chi}$, with N denoting its size.

Step 1: Reset the start point based on the stored trajectory $\boldsymbol{\chi}$:
for $s = N,\ N-1,\ \cdots,\ 1$ **do**
  if $p^* \le p_s$ **then**
    $\left\{p^{(0)}, \rho^{(0)}, u^{(0)}\right\} \leftarrow \{p_s, \rho_s, u_s\}$ ▷ reuse the stored trajectory
    $\Delta p \leftarrow \min\left(\Delta p, p_s - p^*\right)$
    break
  end
end
Step 2: Integrate, and update the stored trajectory
$i \leftarrow 0$
while $p^{(i)} > p^*$ **do**
  $\Delta p \leftarrow \min(\Delta p,\ p^{(i)} - p^*)$
  $\left\{p^{(i+1)}, \rho^{(i+1)}, u^{(i+1)}\right\} \leftarrow \boldsymbol{I}\left(\left\{p^{(i)}, \rho^{(i)}, u^{(i)}\right\}, \Delta p\right)$ ▷ integrate one step by 4th-order Runge-Kutta
  if $p^{(i+1)} < p_N$ **then**
    $N \leftarrow N + 1$
    $\{p_N, \rho_N, u_N\} \leftarrow \left\{p^{(i+1)}, \rho^{(i+1)}, u^{(i+1)}\right\}$ ▷ add new point to $\boldsymbol{\chi}$
  end
  $i \leftarrow i + 1$
end
Output: $\rho_K^* = \rho^{(i)}$, $u_K^* = u^{(i)}$, and $\boldsymbol{\chi}$

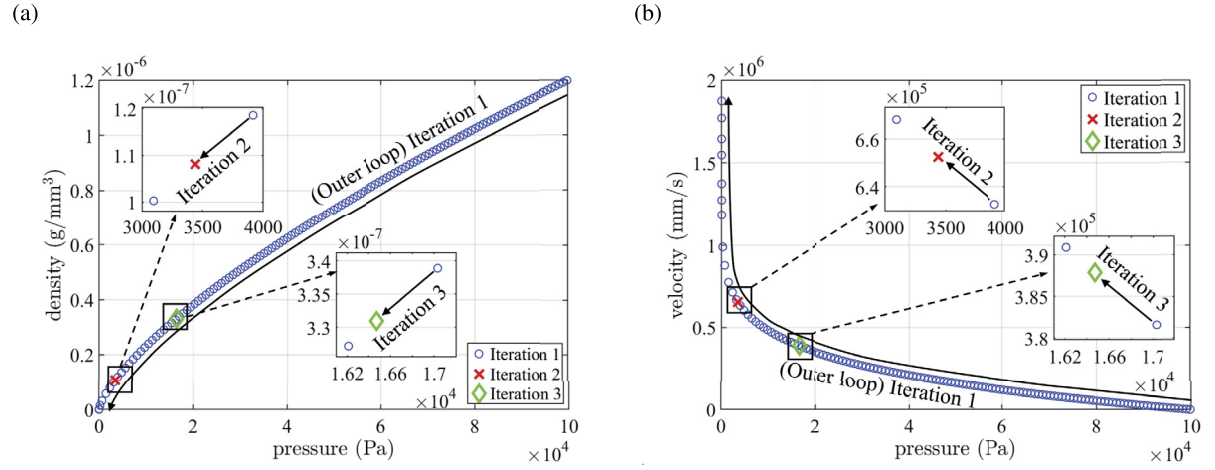

Fig. 5. Integration trajectories in three successive outer loop iterations, with Algorithm 1.

It is possible that both the 1-wave and the 3-wave are rarefactions. Therefore, two separate trajectories, namely $\boldsymbol{\chi}_l$ and $\boldsymbol{\chi}_r$, should be constructed.

Consider the same example problem shown in Fig. 4. If we apply Algorithm 1, iterations 2 and 3 can be completed in one integration step (Fig. 5), as opposed to hundreds. This leads to a significant reduction of computational cost.

3.4. Adaptive step size

The accuracy and cost of the numerical integration performed in Step 2 of Algorithm 1 depend heavily on the step size, Δp. When a user solves a small number of Riemann problems one by one, it may be possible to specify an appropriate step size for each problem based on the user's experience. Nonetheless, when exact Riemann problems are constructed and solved in 2D and 3D multi-material simulations, manually specifying an integration step size for each Riemann problem is impossible. A common practice is to specify a fixed number of integration steps for all the exact Riemann problems constructed throughout the simulation (e.g., [28,1,5]). Clearly, this is not optimal from both accuracy and efficiency standpoints. For different Riemann problems, the number of integration steps required to achieve a certain degree of accuracy varies with the EOS and the integration interval. Moreover, for each Riemann problem, applying a constant Δp through the entire rarefaction fan is also problematic, as $\rho(p)$ and $u(p)$ are often highly nonlinear. Their slopes may change rapidly through a rarefaction. For example, in the case shown in Fig. 4(b), the slope of $u(p)$ varies by 6 orders of magnitude — from -2 mm/(s Pa) to -4×10^6 mm/(s Pa) — along the trajectory.

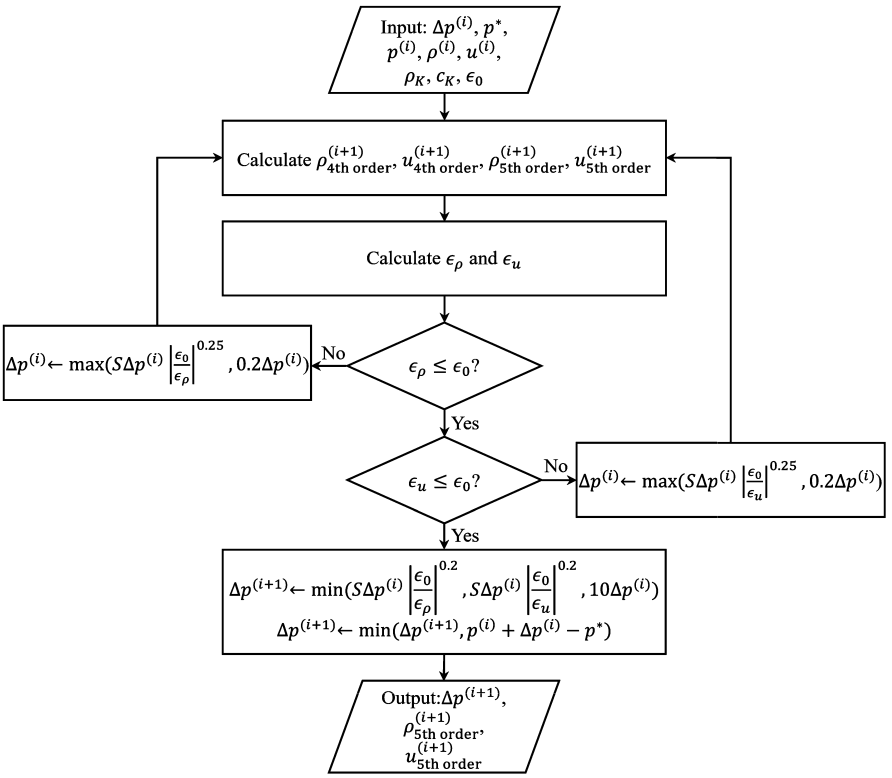

Fig. 6. Algorithm 2: One step of numerical integration through a rarefaction fan with adaptive step size.

To resolve this issue, we propose to apply the adaptive Runge-Kutta methods, which automatically adjust the step size based on the local, single-step error of the integration. This method (Algorithm 2) is described in Fig. 6 in the form of a flowchart. Here, we apply the Runge-Kutta-Fehlberg method [54] as an example, which embeds a fourth-order formulation inside a fifth-order one. The difference between the fourth- and fifth-order results serves as an approximation of the local error. Applying this method to (41) and (42), the fourth-order integration formulas for density and velocity are given by

$$\rho^{(i+1)}_{\text{4th order}} = \rho^{(i)} + \left(\frac{2825}{27,648} \frac{1}{\hat{c}_1^2} + \frac{18,575}{48,384} \frac{1}{\hat{c}_3^2} + \frac{13,525}{55,296} \frac{1}{\hat{c}_4^2} + \frac{277}{14,336} \frac{1}{\hat{c}_5^2} + \frac{1}{4} \frac{1}{\hat{c}_6^2} \right) \Delta p^{(i)}, \tag{44}$$

$$u^{(i+1)}_{\text{4th order}} = u^{(i)} \mp \left(\frac{2825}{27,648} \frac{1}{\hat{\rho}_1 \hat{c}_1} + \frac{18,575}{48,384} \frac{1}{\hat{\rho}_3 \hat{c}_3} + \frac{13,525}{55,296} \frac{1}{\hat{\rho}_4 \hat{c}_4} + \frac{277}{14,336} \frac{1}{\hat{\rho}_5 \hat{c}_5} + \frac{1}{4} \frac{1}{\hat{\rho}_6 \hat{c}_6} \right) \Delta p^{(i)}. \tag{45}$$

Similarly, the fifth-order integration formulas are given by

$$\rho^{(i+1)}_{\text{5th order}} = \rho^{(i)} + \left(\frac{37}{378} \frac{1}{\hat{c}_1^2} + \frac{250}{621} \frac{1}{\hat{c}_3^2} + \frac{125}{594} \frac{1}{\hat{c}_4^2} + \frac{512}{1771} \frac{1}{\hat{c}_6^2} \right) \Delta p^{(i)}, \tag{46}$$

$$u^{(i+1)}_{\text{5th order}} = u^{(i)} \mp \left(\frac{37}{378} \frac{1}{\hat{\rho}_1 \hat{c}_1} + \frac{250}{621} \frac{1}{\hat{\rho}_3 \hat{c}_3} + \frac{125}{594} \frac{1}{\hat{\rho}_4 \hat{c}_4} + \frac{512}{1771} \frac{1}{\hat{\rho}_6 \hat{c}_6} \right) \Delta p^{(i)}. \tag{47}$$

In (45) and (47), the minus sign applies to the 1-wave, and the plus sign applies to the 3-wave. $\hat{c}_i$ and $\hat{\rho}_i$ $(i = 1, 2, \cdots, 6)$ are sound speed and density evaluated at intermediate stages, specifically,

$$
\begin{aligned}
&\hat{c}_1 = c_K\left(p^{(i)},\ \hat{\rho}_1\right),\ \text{with}\ \hat{\rho}_1 = \rho^{(i)},\\
&\hat{c}_2 = c_K\left(p^{(i)} + \frac{1}{5}\Delta p^{(i)},\ \hat{\rho}_2\right),\ \text{with}\ \hat{\rho}_2 = \rho^{(i)} + \frac{1}{5}\frac{1}{\hat{c}_1^2}\Delta p^{(i)},\\
&\hat{c}_3 = c_K\left(p^{(i)} + \frac{3}{10}\Delta p^{(i)},\ \hat{\rho}_3\right),\ \text{with}\ \hat{\rho}_3 = \rho^{(i)} + \frac{3}{40}\frac{1}{\hat{c}_1^2}\Delta p^{(i)} + \frac{9}{40}\frac{1}{\hat{c}_2^2}\Delta p^{(i)},\\
&\hat{c}_4 = c_K\left(p^{(i)} + \frac{3}{5}\Delta p^{(i)},\ \hat{\rho}_4\right),\ \text{with}\ \hat{\rho}_4 = \rho^{(i)} + \frac{3}{10}\frac{1}{\hat{c}_1^2}\Delta p^{(i)} - \frac{9}{10}\frac{1}{\hat{c}_2^2}\Delta p^{(i)} + \frac{6}{5}\frac{1}{\hat{c}_3^2}\Delta p^{(i)},\\
&\hat{c}_5 = c_K\left(p^{(i)} + \Delta p^{(i)},\ \hat{\rho}_5\right),\ \text{with}\ \hat{\rho}_5 = \rho^{(i)} - \frac{11}{54}\frac{1}{\hat{c}_1^2}\Delta p^{(i)} + \frac{5}{2}\frac{1}{\hat{c}_2^2}\Delta p^{(i)} - \frac{70}{27}\frac{1}{\hat{c}_3^2}\Delta p^{(i)} + \frac{35}{27}\frac{1}{\hat{c}_4^2}\Delta p^{(i)},\\
&\hat{c}_6 = c_K\left(p^{(i)} + \frac{7}{8}\Delta p^{(i)},\ \hat{\rho}_6\right),\\
&\quad\text{with}\ \hat{\rho}_6 = \rho^{(i)} + \frac{1631}{55,296}\frac{1}{\hat{c}_1^2}\Delta p^{(i)} + \frac{175}{512}\frac{1}{\hat{c}_2^2}\Delta p^{(i)} + \frac{575}{13,824}\frac{1}{\hat{c}_3^2}\Delta p^{(i)} + \frac{44,275}{110,592}\frac{1}{\hat{c}_4^2}\Delta p^{(i)} + \frac{253}{4096}\frac{1}{\hat{c}_5^2}\Delta p^{(i)}.
\end{aligned}
\tag{48}
$$

The coefficients utilized in the above formulas were developed by Cash and Karp [55]. They allow the intermediate stages involved in the fourth-order formula to be re-used in the fifth-order one.

We normalize the errors in density and velocity using the initial state, i.e.,

$$\epsilon_\rho = \left|\frac{\rho^{(i+1)}_{\text{4th order}} - \rho^{(i+1)}_{\text{5th order}}}{\rho_K}\right| + \tilde{\epsilon}, \tag{49}$$

$$\epsilon_u = \left|\frac{u^{(i+1)}_{\text{4th order}} - u^{(i+1)}_{\text{5th order}}}{c_K}\right| + \tilde{\epsilon}, \tag{50}$$

where ρ_K and c_K are density and sound speed in the initial state. Because ϵ_ρ and ϵ_ρ will appear in the denominator, a small positive number, $\tilde{\epsilon}$, is added to avoid division by zero. In all the numerical tests presented in this paper, it is set to 1×10^{-30}. Then, the integration step size Δp is adjusted according to the two errors and an error tolerance ϵ_0. For example, a new step size, Δp^*, can be calculated based on the density error, ϵ_ρ, by

$$\Delta p^* = S\cdot\Delta p^{(i)}\cdot\left(\frac{\epsilon_0}{\epsilon_\rho}\right)^\alpha. \tag{51}$$

Here, $S\in(0,1)$ is introduced as a safety factor, because ϵ_ρ is only an estimate of the true error. In this work, we set $S=0.9$, following [56]. α is set to 0.2 if $\epsilon_\rho\le\epsilon_0$, and 0.25 otherwise [56].

If $\epsilon_\rho\le\epsilon_0$, we set

$$\Delta p^{(i+1)} = \Delta p^*, \tag{52}$$

and move on to examine the error in velocity (Fig. 6). If $\epsilon_\rho>\epsilon_0$, we repeat step i with a smaller step size, i.e.

$$\Delta p^{(i)} = \Delta p^*. \tag{53}$$

Following [57], we cap the variation of Δp within one step at a factor of 10 for increase, and a factor of 5 for decrease. Next, the same procedure is applied to update Δp using the velocity error ϵ_u, as shown in Fig. 6.

To apply the adaptive step size within Algorithm 1, we note that the input Δp in Algorithm 1 becomes $\Delta p^{(0)}$. It does not need to be determined precisely, as the actual step size will be calculated automatically in each integration step. In practice, we set $\Delta p^{(0)}$ to be a small number to avoid the repetition of the first integration step (i.e., $i=0$). Because of step size adaptation, every point (i.e., $\{p_s\mid s\in[1,N]\}$) stored on the integration trajectory $\boldsymbol{\chi}$ is quality checked in terms of accuracy. The stored step sizes (e.g., p_s-p_{s+1}) automatically satisfy the requirement on local accuracy. This fact can be utilized to determine the new initial step size $\Delta p^{(0)}$ when we reset the integration start point. If the new start point p_s is not the last stored point on $\boldsymbol{\chi}$, the inequality $p_s>p^*>p_{s+1}$ must hold. This means the new initial step size $\Delta p^{(0)}$ can be set by

$$\Delta p^{(0)} \leftarrow p_s - p^*. \tag{54}$$

In this way, the new integration is completed in only one step while satisfying accuracy requirements. If the new start point p_s is the last stored point on $\boldsymbol{\chi}$, we set the new initial step size to

$$\Delta p^{(0)} \leftarrow \min\left(p_{s-1}-p_s,\ p_s-p^*\right). \tag{55}$$

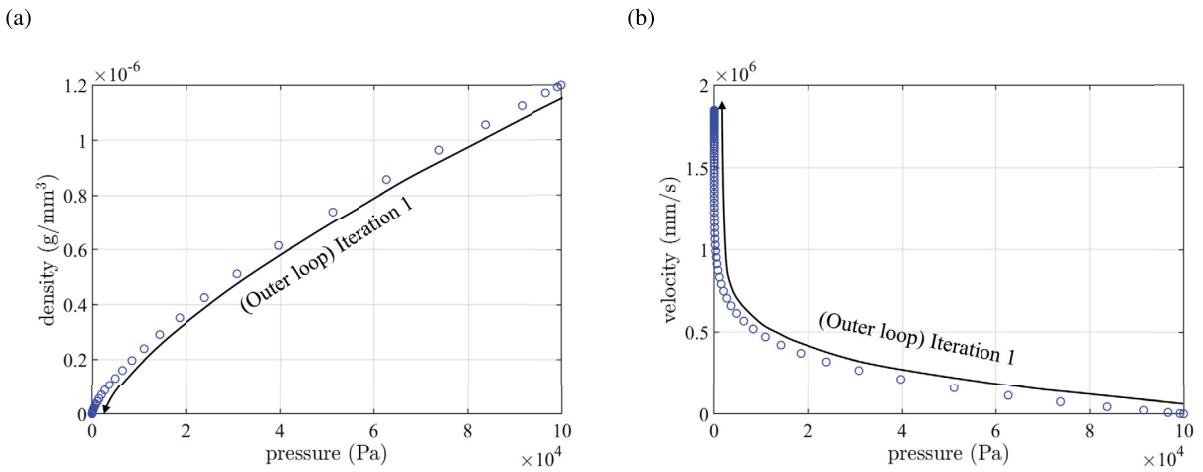

Fig. 7. Repetition of iteration 1 in Fig. 4 with adaptive step size.

Therefore, (54) and (55) are used to replace line 5 in Algorithm 1 when adaptive step size is applied.

Repeating the solution of the Riemann problem given in (43) with adaptive integration step size gives Fig. 7. Comparing with the result shown in Fig. 4, the advantage brought by the adaptive step size is clear. Smaller step sizes are applied near the left end, where the slopes of $\rho(p)$ and $u(p)$ vary rapidly. At the right end, although the starting step size, $\Delta p^{(0)}$ is small, the algorithm is able to quickly increase its value to adapt to the slow variation of slopes.

3.5. Obtaining initial guesses using R-tree data structure

The computational cost of exact Riemann problem solution depends on the number of iterations performed in the outer loop (Fig. 3). When the error tolerance is fixed, the number of iterations is largely determined by the quality of the initial guesses. A popular method to specify initial guesses is to apply the linear acoustic theory [41]. Assuming that two initial guesses are needed, such as in the Secant method, the first one, $p^{*(0)}$, is given by

$$p^{*(0)} = \left[C_r p_l + C_l p_r + C_l C_r \left(u_l - u_r\right)\right] / \left(C_l + C_r\right), \tag{56}$$

where l and r refer to the two materials, c denotes the sound speed, and $C = \rho c$. The second initial guess, $p^{*(1)}$, is obtained by applying a different sound speed. Specifically,

$$p^{*(1)} = \left[\overline{C}_r p_l + \overline{C}_l p_r + \overline{C}_l \overline{C}_r \left(u_l - u_r\right)\right] / \left(\overline{C}_l + \overline{C}_r\right), \tag{57}$$

where $\overline{C}_K$ $(K = l, r)$ is given by

$$\overline{C}_K = \begin{cases} \left|p^{*(0)} - p_K\right| / \left|u_K^{*(0)} - u_K\right| & \text{if} \quad u_K^{*(0)} \neq u_K, \\ \rho_K c_K & \text{if} \quad u_K^{*(0)} = u_K. \end{cases} \tag{58}$$

Here, $u_K^{*(0)}$ denotes the velocity obtained with pressure $p^{*(0)}$.

When the Riemann problem is highly nonlinear, the initial guesses obtained using the linear acoustic theory may not be close to the true solution. To mitigate this issue, we introduce another method based on storing and reusing the solutions of previously solved Riemann problems. The motivation of this data-based method is that in the course of a multi-material simulation, a large number of bimaterial Riemann problems are constructed and solved. Some of them may have similar inputs, hence similar solutions. As an example, Fig. 8 visualizes the Riemann problems solved in the simulation presented in Sec. 4.3, within 10 time steps. Some clusters can be easily identified in this data set. Our basic idea is to store the solutions of solved exact Riemann problems using an efficient data structure, and to generate the initial guesses for a new problem through nearest neighbor search.

Notably, the bimaterial Riemann problem is 5D, with $(\rho_l, p_l, \rho_r, p_r, u_r - u_l)$ as the independent variables. Also, this data set needs to be frequently updated and accessed. For these reasons, we adopt the R-tree data structure, which is specifically designed for efficient indexing and searching of multidimensional data [42,43].

Unlike the idea of constructing a sparse grid before each simulation [28], the R-tree data set is designed to be constructed during the simulation (i.e., "on the fly"). Specifically, this method involves three components.

- **Construction of R-tree.** In the course of a simulation, after solving each bimaterial Riemann problem, we pair the solution p^* with the corresponding 5D inputs, and insert the pair into the R-tree.

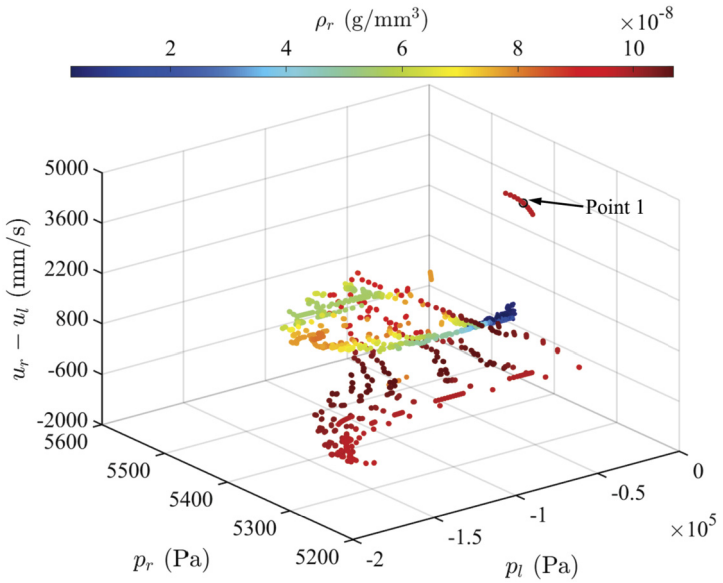

Fig. 8. Visualization of the bimaterial Riemann problems solved in the simulation presented in Sec. 4.3 during 10 time steps. This 5D data set is displayed with p_l, p_r, and $u_r - u_l$ as the three spatial coordinates. The color of each data point indicates the value of ρ_r, and the size of the point indicates the value of ρ_l. "Point 1" is used to demonstrate the initial guess improvement from the nearest neighbor search.

Table 2
Comparison of the state variables at Point 1 in Fig. 8 and its nearest neighbor in the R-tree.

	ρ_l (g/mm^3)	p_l (Pa)	ρ_r (g/mm^3)	p_r (Pa)	$u_r - u_l$ (mm/s)	p^* (Pa)
Point 1	9.9628×10^{-4}	-4.4378×10^4	9.8425×10^{-8}	5.3338×10^3	4.3761×10^3	5.21797×10^3
Nearest neighbor	9.9628×10^{-4}	-4.4266×10^4	9.8425×10^{-8}	5.3338×10^3	4.3757×10^3	5.21798×10^3

- **Normalization of the five (5) coordinates.** The 5 coordinates of the database represent density, pressure, and velocity. Their numerical values often differ by several orders of magnitudes, depending on their units (e.g., see Fig. 8). Directly performing nearest-neighbor search on these dimensional coordinates is problematic, as the evaluation of distance would be biased towards coordinates with larger numerical values. Therefore, we normalize the 5 coordinates using a lower bound and an upper bound, so that each normalized, non-dimensional coordinate varies between 0 and 1.
- **Nearest-neighbor search.** Once a new Riemann problem is constructed, we apply a bounding-box based nearest-neighbor search algorithm [58,59] to find the data point in the R-tree whose inputs are closest to the new Riemann problem. Its solution, p^*, is taken as the first initial guess for solving the new Riemann problem. The second initial guess, if needed, can be obtained either by Eq. (57) or by applying a perturbation to the first initial guess.

Consider the same example problem shown in Fig. 8. In this figure, Point 1 marks a new Riemann problem that is constructed during the simulation. Table 2 shows the state variables — including both the 5 inputs and the solution p^* — at this point and its nearest neighbor in the existing R-tree. Clearly, the two points are close to each other. Using the solution at the nearest neighbor as the first initial guess, it takes only 1 iteration to solve the new Riemann problem at Point 1. In comparison, it takes 8 iterations to converge to the same error tolerance, if the first initial guess is generated using the linear acoustic theory, i.e., by (56).

This work utilizes the R-tree data structure implemented in the widely used Boost Generic Geometry Library [60]. In particular, the insertion of data points, query of the nearest neighbor, and searching of the lower and upper bounds for normalization are performed by built-in functions in the Boost library. More details about the creating, modifying and using R-tree data structure in Boost library can be found in [60]. For the numerical experiments in Sec. 4, we store the Riemann problems solved in the previous time step, and perform the normalization at the beginning of each time step.

In summary, Fig. 9 presents the new algorithm for solving an exact bimaterial Riemann problem, which incorporates the acceleration methods presented in Sec. 3.2, 3.3, 3.4, and 3.5.

4. Numerical experiments

We assess the performance of the acceleration methods presented in Sec. 3.2 – 3.5 using four example problems that involve multiple (2 or 3) materials and significant discontinuities across their interfaces. For conciseness, the four acceleration methods are referred to as Methods 1 to 4 in this section.

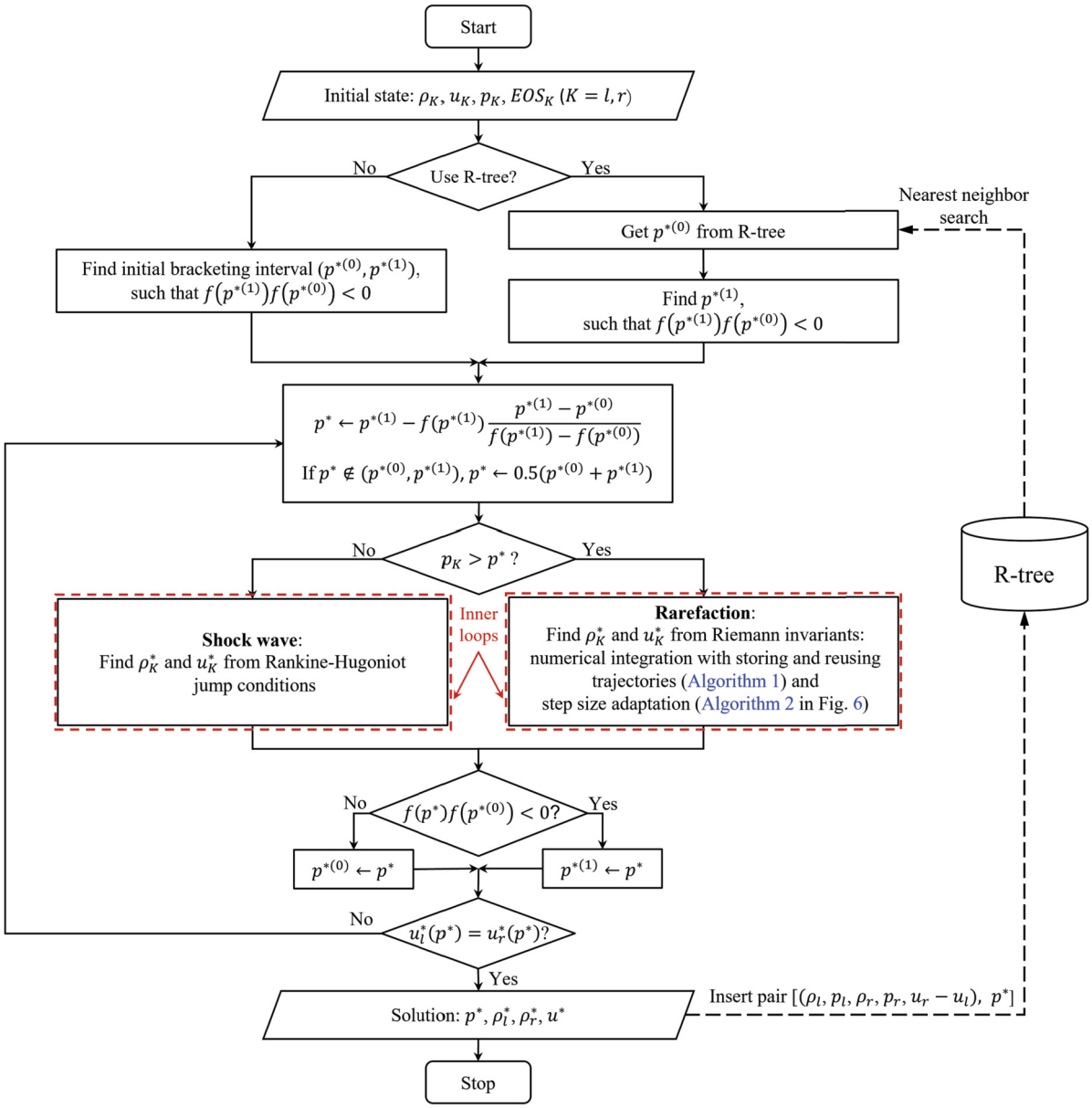

Fig. 9. Improved solution procedure for bimaterial Riemann problems in multi-material flow simulaitons.

- Method 1: Change of integration variable (Sec. 3.2).
- Method 2: Storing and reusing integration trajectory data (Sec. 3.3).
- Method 3: Adaptive step size (Sec. 3.4).
- Method 4: Obtaining initial guesses using R-tree data structure (Sec. 3.5).

The simulations presented in this section are performed using the M2C (Multiphysics Modeling and Computation) solver [61] and the Tinkercliffs computer cluster at Virginia Tech. A standalone version of the exact Riemann problem solver is available at [62].

4.1. One-dimensional benchmark problem

The 1D bimaterial Riemann problem defined by (43) is solved numerically using the FIVER method described in Sec. 2.3. This problem models a condensed phase (soda lime glass) moving away from a gas (air) at high speed (400 m/s). The exact solution is used to verify the numerical solver. At any time $t > 0$, density jumps by 4 orders of magnitude across the

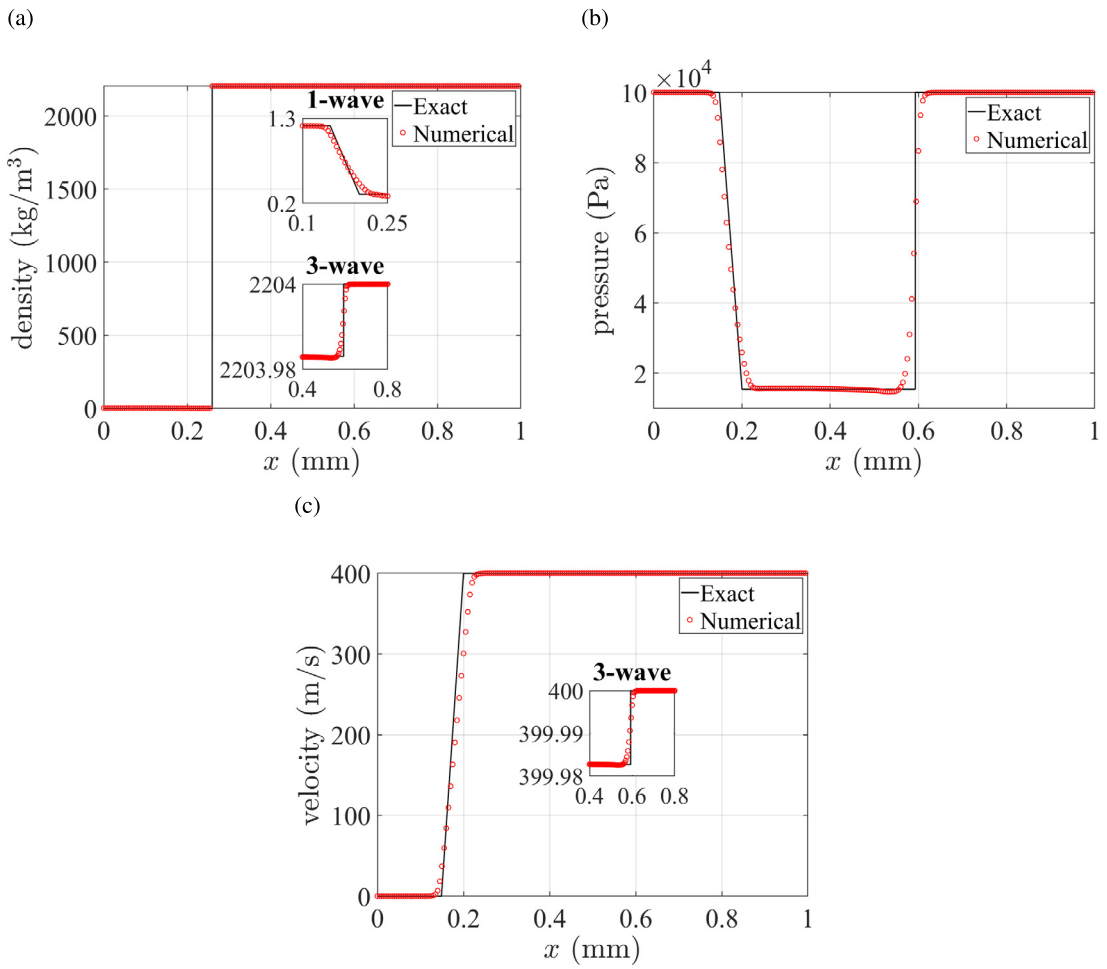

Fig. 10. A 1D bimaterial Riemann problem: Density, pressure, and velocity distributions at $t = 0.15$ μs.

material interface, from 0.3 kg/m^3 to 2203.98 kg/m^3. Therefore, this example problem also challenges the robustness of the solver.

To separate the effects of different acceleration methods, we perform four simulations in which the exact Riemann problems constructed during the simulation are solved using different methods.

- Simulation (i): using the baseline method presented in Sec. 3.1;
- Simulation (ii): with acceleration Methods (1) and (2);
- Simulation (iii): with acceleration Methods (1), (2), and (3); and
- Simulation (iv): with acceleration Methods (1), (2), (3), and (4).

The computational domain extends from $x = 0$ to 1.0 mm. The material interface is initially at $x = 0.2$ mm. 200 rectangular elements are used to discretize the domain. In Simulations (iii) and (iv), the error tolerance for controlling the adaptive step size (Eqs. (49) and (50)) is set to 1.0×10^{-9}. In Simulations (i) and (ii), we need to specify a fixed number of integration steps. To have a fair comparison in terms of solution accuracy, we set the number of integration steps to be the maximum integration steps required in (iii) to reach the error tolerance, which is 5, 307.

The density, pressure, and velocity results obtained from Simulation (iv) at $t = 0.15$ μs are shown in Fig. 10, in comparison with the exact solution. For all the three variables, the simulation result matches closely the exact solution. The results obtained from Simulations (i), (ii), and (iii) are essentially the same as that from Simulation (iv), as the acceleration methods do not introduce new approximations. Therefore, their results are not shown separately.

Fig. 11 compares the computational cost of the four simulations up to $t = 0.15$ μs. Here, we measure the computation time for evaluating the advective fluxes, which includes two parts: the time spent on solving the exact Riemann problems at the material interface, and the time spent on evaluating the numerical flux functions (LLF in this case) over the entire domain. The effect of the acceleration methods is found to be significant. When all the four methods are adopted (i.e., in

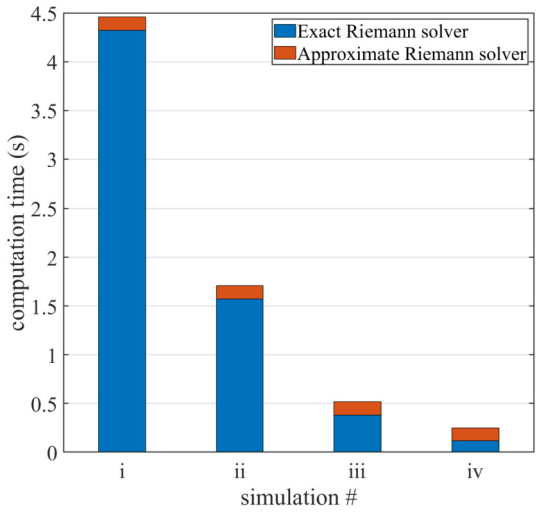

Fig. 11. A 1D bimaterial Riemann problem: Comparison of the flux computation time of different simulations.

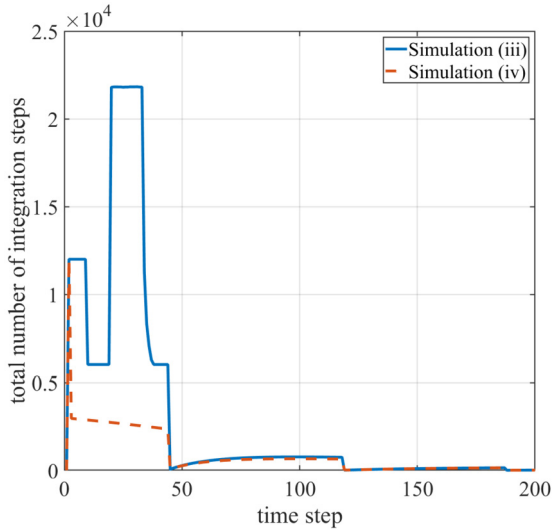

Fig. 12. A 1D bimaterial Riemann problem: Effect of R-tree on the total number of integration steps.

Simulation (iv)), the time consumed by the exact Riemann solver is reduced by a factor of 37.6 compared to the baseline simulation (from 4.32 s to 0.123 s). The total computation time for evaluating the advective fluxes is also significantly reduced, by a factor of 17.8 (from 4.46 s to 0.26 s).

Comparing the computation times of Simulations (iii) and (iv), it can be observed that searching for the initial guesses from the R-tree reduces the cost of exact Riemann problem solutions by about 70%. This reduction can be explained by Fig. 12, which shows the number of integration steps in the first 200 time steps. At each time step, because the initial guesses provided by the R-tree are closer to the true solution, the integration interval is shorter in Simulation (iv), which leads to a smaller number of integration steps. The computational overhead associated with updating, normalizing, and accessing the R-tree is found to be less than 10% of the time spent on solving the exact Riemann problems. This overhead is acceptable, as expected.

4.2. Underwater explosion

We simulate the expansion of a high-pressure gas bubble in a liquid, which can be viewed as a model problem for underwater explosion. Following [63], we specify a bubble with radius $r = 0.2$ at $t = 0$. The (non-dimensional) initial conditions inside and outside the bubble are given by

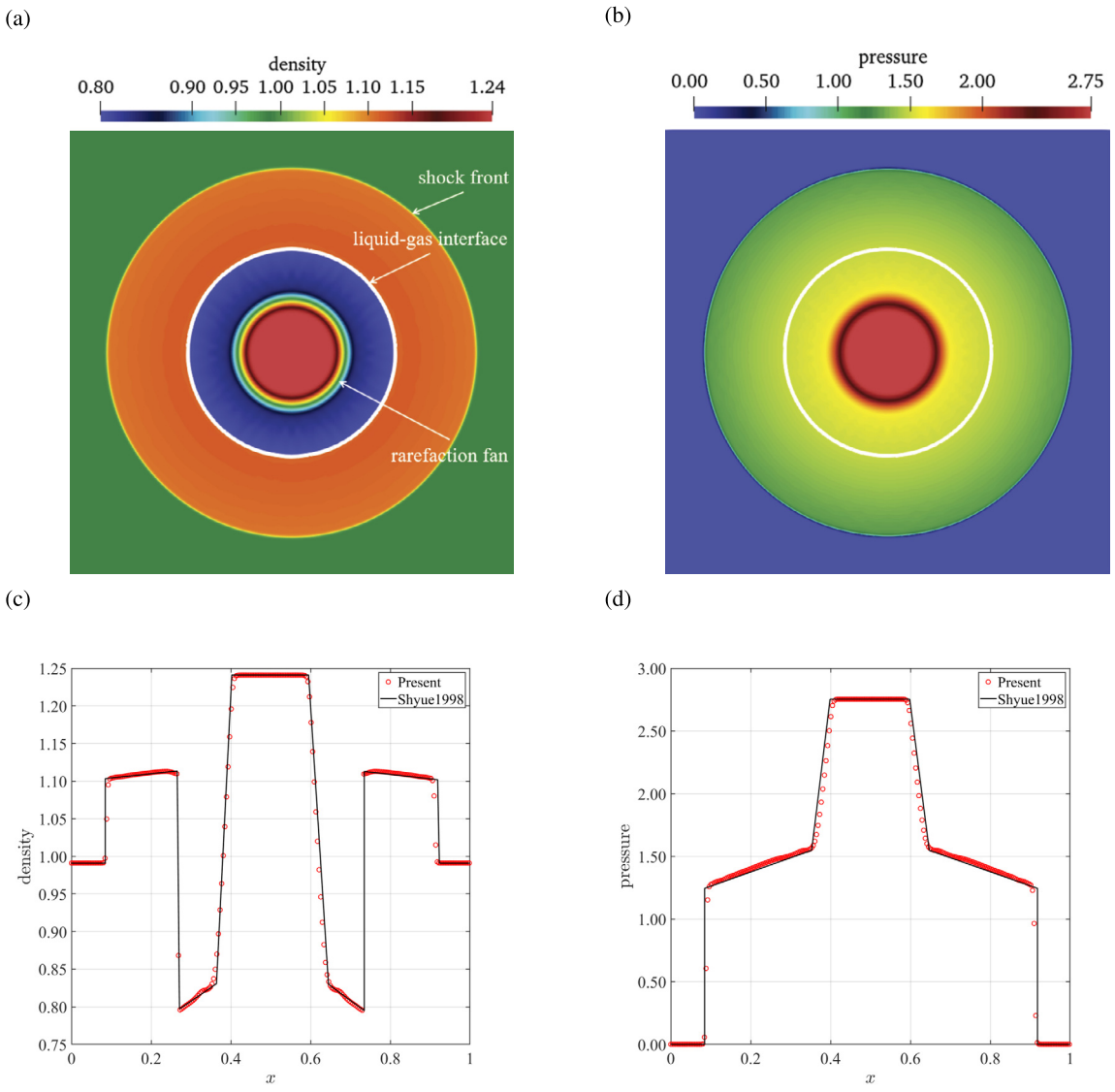

Fig. 13. Underwater explosion: simulation results at $t = 0.058$. (a) density field, (b) pressure field, (c) density along the horizontal symmetric axis, (d) pressure along the horizontal symmetric axis.

$$(\rho, u, p, EOS) = \begin{cases} (0.991, 0, 3.059 \times 10^{-4}, \text{stiffend gas}) & \text{liquid phase, outside the bubble,} \\ (1.241, 0, 2.753, \text{stiffened gas}) & \text{gas phase, inside the bubble.} \end{cases} \tag{59}$$

The parameters in the stiffened gas EOS, i.e., Eq. (7), are $\gamma = 5.5$, $e_c = 0$, $b = 0$ and $p_c = 1.505$ for the liquid phase, and $\gamma = 1.4$, $e_c = 0$, $b = 0$, and $p_c = 0$ for the gas phase. The Euler equations are solved, which means in Eq. (1), $\boldsymbol{\mathcal{S}} = \mathbf{0}$. The computational domain is 2D, extending from 0.0 to 1.0 in both x and y directions. It is discretized by a uniform, 500×500 grid. The Roe's flux function is adopted to compute the advective fluxes. Again, the four simulations designed in Sec. 4.1 are performed for this problem. All the simulations are parallelized to run on 256 CPU cores. Same as in the previous case, the error tolerance for adaptive step size control is set to 1.0×10^{-9} in Simulations (iii) and (iv). Reaching this tolerance requires up to 1,092 integration steps. This number is then specified as the fixed number of integration steps in Simulations (i) and (ii), in order to ensure the same level of accuracy.

The results obtained from the four simulations are essentially the same, as the acceleration methods do not introduce new approximations. As an example, Fig. 13 shows the result of Simulation (iv) at $t = 0.058$. In this figure, Subfigures (a) and (b) visualize the density and pressure fields. Subfigures (c) and (d) plot the same solution fields along the horizontal centerline of the computational domain, in comparison with the results given in [63]. For both density and pressure, the present result is in close agreement with the reference. The liquid-gas interface, the outgoing shock wave, and the incoming rarefaction fan are all captured accurately.

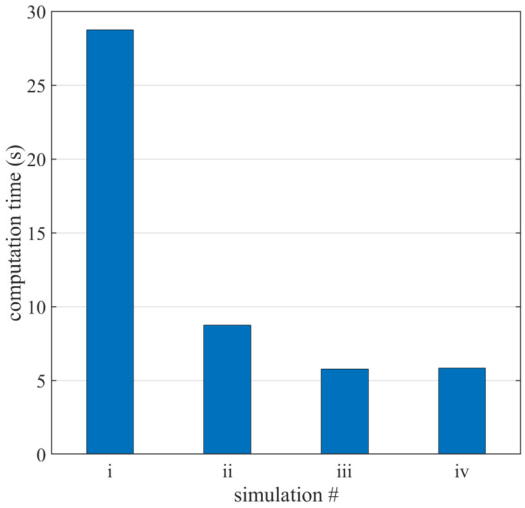

Fig. 14. Underwater explosion: Computational cost of exact Riemann problem solutions in the first 200 time steps.

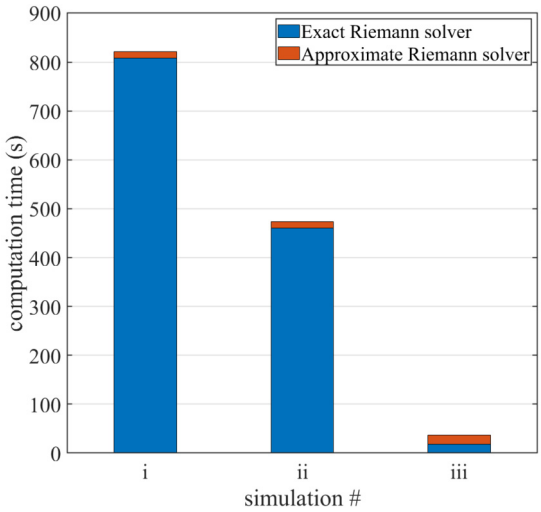

Fig. 15. Underwater explosion: Comparison of the flux computation time of different simulations.

Fig. 14 shows the computational cost of exact Riemann problem solutions within the first 200 time steps, when the exact Riemann problems are highly nonlinear. After storing and reusing the integration trajectories (i.e., Simulation (ii)), the cost of numerical integration becomes trivial after the first iteration in the outer loop. As a result, the computation time is reduced by a factor of 3 (9 s vs. 29 s), compared to the baseline Simulation (i). After adopting step size adaptation (i.e., Simulation (iii)), the computation time is further reduced by 34%, because fewer integration steps are needed. Comparing the costs of Simulations (iii) and (i), it is found that the first three acceleration methods, when combined together, provides a 5-fold cost reduction. By obtaining the initial guesses from R-tree (i.e., Simulation (iv)), however, the computational cost increases slightly by 0.86% compared to Simulation (iii). This is because the reduction in Riemann problem solution time is offset by the computational overhead associated with updating and maintaining the R-tree. Specifically, the Riemann problem solution time is reduced by 10.35% compared to Simulation (iii), but the computational overhead amounts to 11.21%. The benefit brought by the R-tree is less significant than in the previous case. This can be attributed to the fact that the density jump across the material interface is less than a factor of 2 in this case, while it was 4 orders of magnitude in the previous case. Overall, we have found that unless the density and EOS across the material interface are extremely different, the initial guesses provided by the R-tree may not be significantly better than those obtained by the linear acoustic theory. For this reason, Simulation (iv) is excluded from the subsequent test cases, except one in Sec. 4.4, where the R-tree's benefits overtake its cost due to the significant discontinuity across the material interface.

Furthermore, Fig. 15 compares the computation time of Simulations (i)(ii)(iii) up to $t = 0.058$, or 13,029 time steps. It can be observed that when acceleration Methods (1), (2), and (3) are applied, the solution of exact Riemann problems is

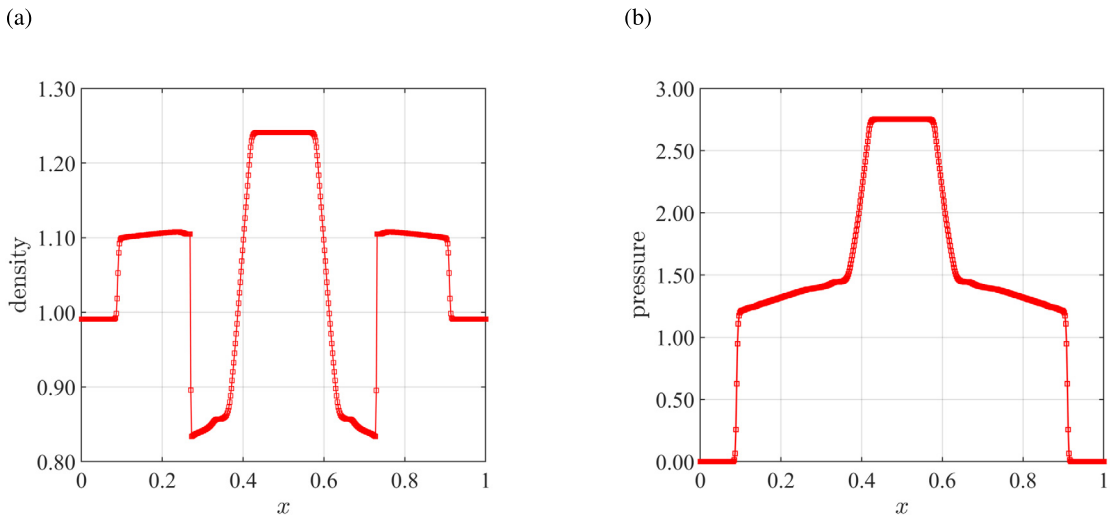

Fig. 16. Underwater explosion with the JWL EOS: Simulation result at $t = 0.058$, along the horizontal centerline of the computational domain.

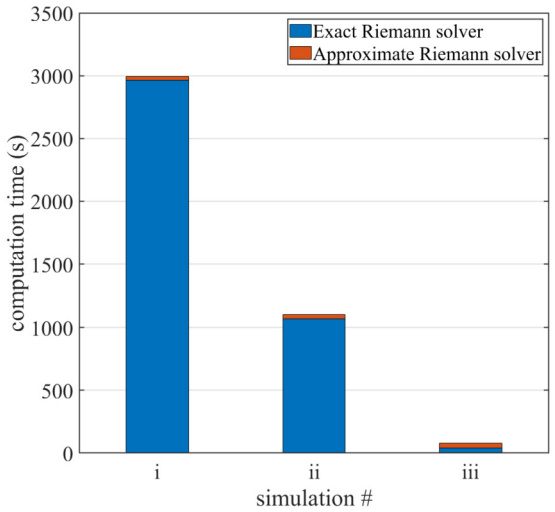

Fig. 17. Underwater explosion with the JWL EOS: Comparison of the flux computation time of different simulations.

accelerated by 45 times (from 808 s to 18 s). The total time on advective flux computation is reduced by a factor of 22 (from 822 s to 37 s).

In the aforementioned simulations, the gas inside the bubble is modeled using the stiffened gas EOS, following the example problem described in [63]. In general, the thermodynamics of gaseous explosion products can be better described using the JWL EOS (9). Compared to stiffened gas, the JWL EOS is highly nonlinear as it involves exponential functions. Also, ρ can no longer be solved analytically from p and e. Instead, an iterative numerical method is needed. To further assess the performance of the acceleration methods, we perform another set of simulations in which the gas inside the bubble is modeled using the JWL EOS. Its parameters are set by $\omega = 0.28$, $A_1 = 37.12$, $A_2 = 0.323$, $R_1 = 4.15$, $R_2 = 0.95$, and $\rho_0 = 1.241$, so that the results are comparable with those obtained using the stiffened gas EOS. Fig. 16 shows the density and pressure distributions at $t = 0.058$ along the horizontal centerline of the computational domain.

Fig. 17 shows the performance of the accelerated Riemann problem solver, in the presence of the JWL EOS. Again, the computation time is calculated up to $t = 0.058$. Comparing with Fig. 15, it is clear that the JWL EOS leads to a higher computational cost. The effect of the acceleration methods is even more significant. When Methods (1), (2), and (3) are combined (i.e., Simulation (iii)), the solution of exact Riemann problems is accelerated by 76 times (39 s vs. 2,963 s). As a result, the total cost of flux computation is reduced by a factor of 38 (80 s vs. 2,994 s).

4.3. Laser-induced cavitation

Next, we assess the performance of the acceleration methods using an example problem of cavitation induced by a long-pulse laser, first presented in [1]. A Holmium:YAG laser with a 2.12×10^{-3} mm wavelength, a 10.2 μs pulse duration, and a peak power of 2.854 kW is considered. The cylindrical laser fiber has a radius of 0.1825 mm, and is placed within a water tank. The energy of the laser pulse is high enough to vaporize a small volume of liquid water in front of the laser fiber, thereby generating a vapor bubble. The two-phase liquid-vapor flow is modeled as a compressible and inviscid flow. A radiative heat source is added to the energy conservation equation. Therefore, the term $\boldsymbol{\mathcal{S}}$ in (1) has the form

$$\boldsymbol{\mathcal{S}} = \begin{bmatrix} 0 \\ \mathbf{0} \\ -\nabla \cdot \boldsymbol{q}_r \end{bmatrix}, \tag{60}$$

where $\boldsymbol{q}_r$ denotes the radiative heat flux, given by

$$\boldsymbol{q}_r = L\boldsymbol{s}. \tag{61}$$

Here, $L = L(\boldsymbol{x})$ denotes laser radiance at position $\boldsymbol{x}$. It characterizes the laser intensity at that position. $\boldsymbol{s}$ denotes the propagation direction of the laser light. It is also a function of $\boldsymbol{x}$ because the laser beam simulated here has a 7.5° diverging angle. L is governed by the following laser radiation equation that enforces conservation of energy.

$$\nabla \cdot (L\boldsymbol{s}) = \nabla L \cdot \boldsymbol{s} + (\nabla \cdot \boldsymbol{s})L = -\mu_\alpha L. \tag{62}$$

Here, μ_α is the laser absorption coefficient, which depends on the laser wavelength, the fluid material, and the fluid temperature. Same as in [1], the method of latent heat reservoir is applied to model the phase transition (i.e., vaporization). In addition, the cylindrical symmetry of the fluid flow with respect to the central axis of the laser fiber is leveraged to allow simulations on 2D grids.

Both the liquid and vapor phases are modeled by the stiffened gas EOS. The parameters are $\gamma = 6.12$, $e_c = 0$, $b = 0$, and $p_c = 343$ MPa for the liquid phase, and $\gamma = 1.34$, $e_c = 0$, $b = 0$, and $p_c = 0$ for the vapor. In the latter case, the stiffened gas degenerates to the perfect gas. Additional details about this problem — including its background, novelty, the model equations, and the numerical methods — can be found in [1].

The computational domain is 2D, discretized by a non-uniform Cartesian grid with approximately 338, 000 elements. In the most refined region, the element size is around 2.5×10^{-3} mm. Simulations (i), (ii), and (iii) defined in Sec. 4.1 are performed for this example problem, on 256 CPU cores. Again, the error tolerance for integration step adaptation is set to 1.0×10^{-9} in Simulation (iii), which translates to 841 integration steps for Simulations (i) and (ii).

Fig. 18 presents the laser radiance and fluid velocity fields obtained from Simulation (iii). The solutions obtained from the other two simulations are the same. Initially, the entire fluid domain is occupied by the liquid phase. The laser source power rapidly grows from 0 to its peak value during the first 0.2 μs. Then, it stays at the peak value until $t = 10.0$ μs. At around 4.8 μs, the simulation predicts the formation of a vapor bubble right in front of the laser fiber tip. Phase transition lasted around 0.025 μs. Afterwards, the main flow features are the expansion of the bubble, the propagation of vaporization-induced acoustic waves, and laser heating. Fig. 19 presents the fluid pressure field at six time instances. Subfigures (1) and (2) show the acoustic waves induced by the sudden increase of the fluid's internal energy due to laser radiation. Subfigures (3) and (4) demonstrate the formation and expansion of the vapor bubble, as well as the associated acoustic waves. The laser power vanishes at $t = 10.2$ μs. Afterwards, the bubble continues to grow and gradually deforms into a teardrop shape, which is shown in Subfigures (5) and (6). The same shape has been observed in laboratory experiments using the same type of laser [64]. This type of non-spherical, beam dependent bubbles is a unique feature of long-pulse laser-induced cavitation.

Fig. 20 compares the computation times of Simulations (i)(ii)(iii) up to $t = 50$ μs. The advective fluxes are evaluated using the LLF numerical flux function. It can be observed that the solution of exact Riemann problems is accelerated by 51 times (350 s vs. 17, 899 s), when Methods (1), (2), and (3) are used. In the baseline Simulation (i), over 95% of the flux computation time is spent on solving the exact Riemann problems. As a result, after the acceleration, the total flux computation time is reduced by a factor of 26 (from 18, 233 s to 696 s).

4.4. Hypervelocity impact in a fluid environment

Finally, we consider an example problem of hypervelocity impact, first presented in [5]. In this problem, a copper projectile impacts onto a soda lime glass (SLG) target in a neon gas environment. If the projectile's velocity is sufficiently high, the peak pressure inside the projectile and the target can be much higher than the material's Hugoniot elastic limit. In this scenario, the solid materials can be modeled as compressible fluids. Unlike conventional impact simulations that consider only the projectile and the target, we perform a fluid-solid coupled analysis that includes the ambient neon gas as a material subdomain. Therefore, this problem involves three material subdomains, and three material interfaces (neon-copper, copper-SLG, and SLG-neon). Two level set equations are solved to track the boundaries of the copper and SLG subdomains,

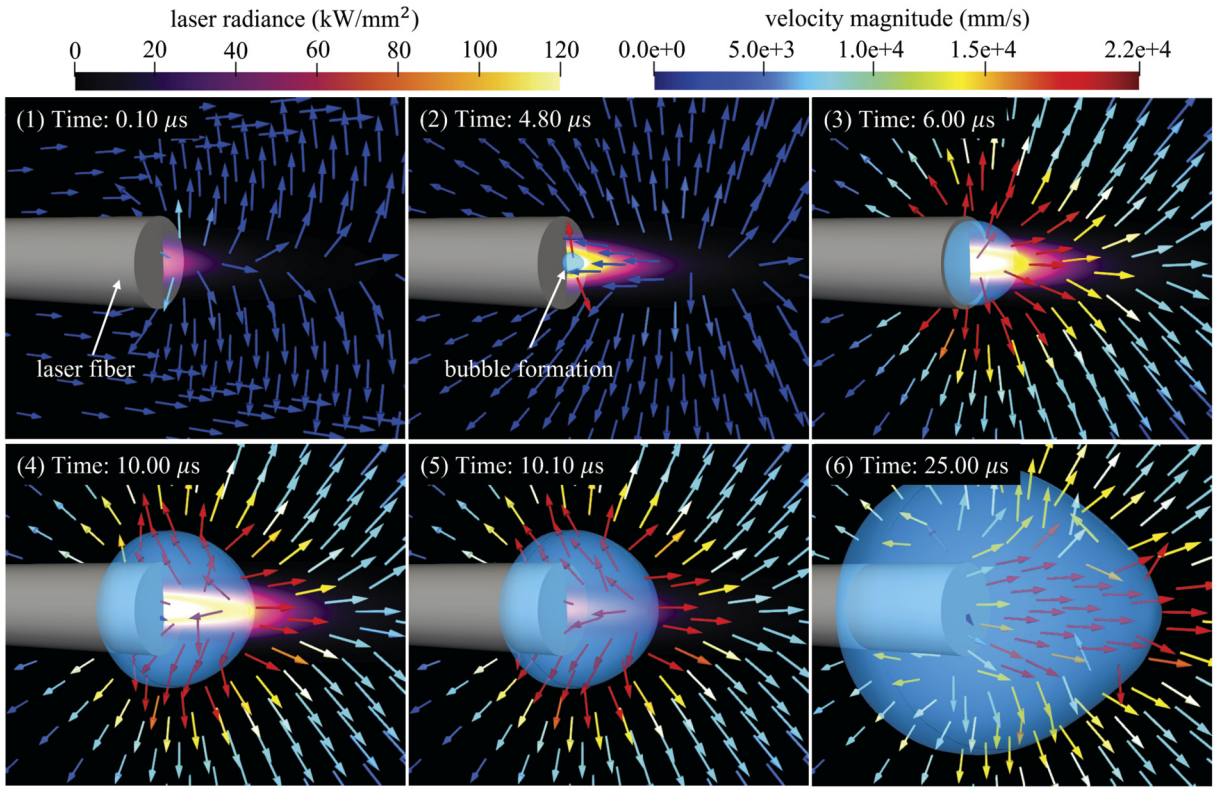

Fig. 18. Laser-induced cavitation: Laser radiation, bubble formation, and the fluid velocity field.

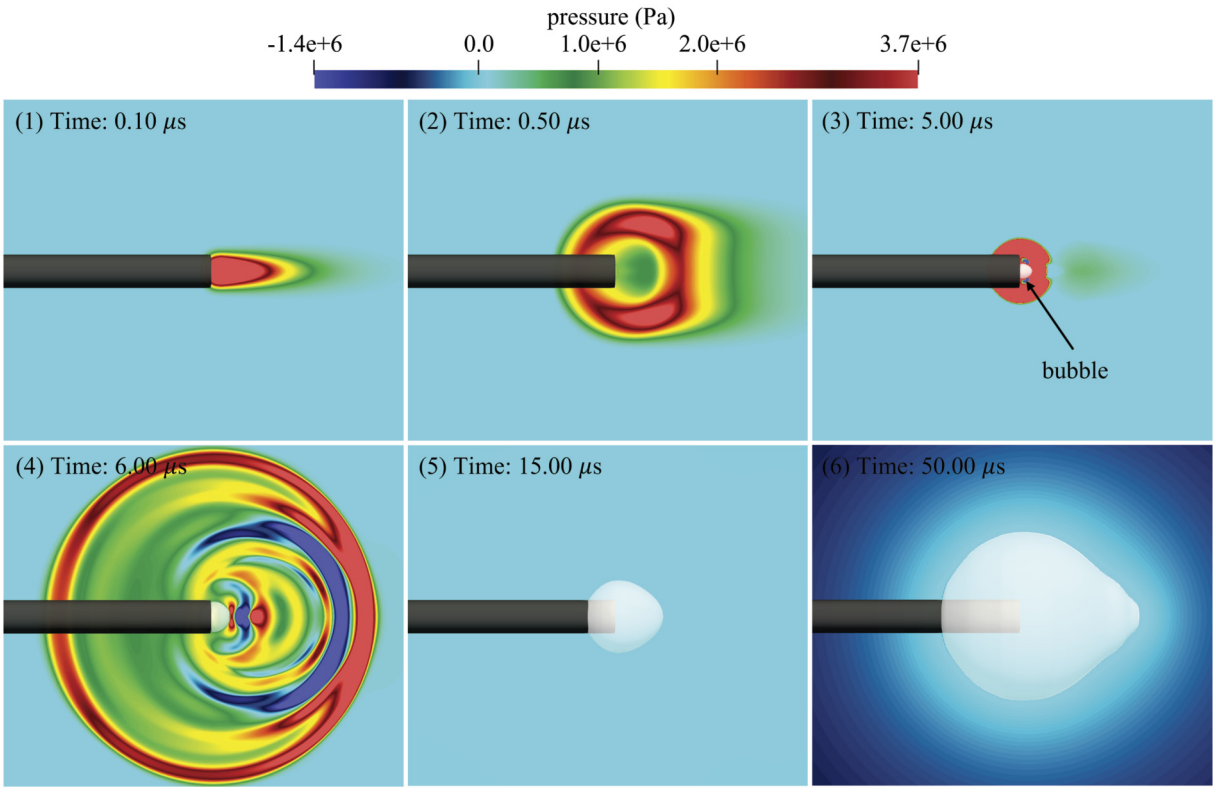

Fig. 19. Laser-induced cavitation: The pressure field.

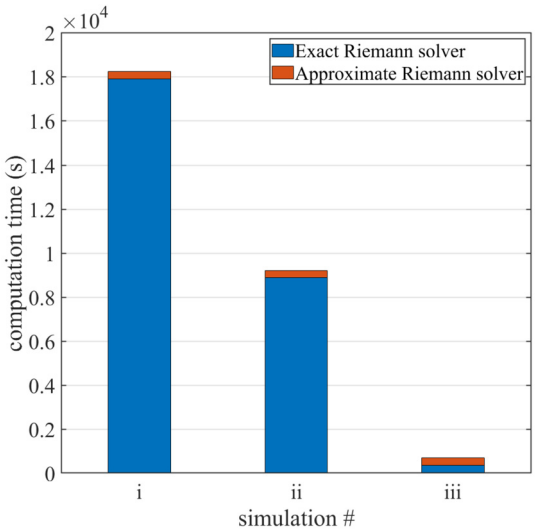

Fig. 20. Laser-induced cavitation: Comparison of the flux computation time of different simulations.

using the method described in Sec. 2.3. This problem also challenges the robustness of the solver as it exhibits extreme mechanical and thermodynamic states, and the EOS vary significantly across the fluid-solid interfaces.

Following [5], the effects of viscosity and heat diffusion are neglected. The impact problem is analyzed in two stages, namely the flight of the projectile in the fluid medium, followed by the collision of the projectile onto the target. The first stage produces a bow shock that reaches the target before the projectile. This flow field can be obtained by either a conventional body-fitted CFD analysis ([5]), or a multi-material simulation in which the projectile surface is tracked using the level set method. To demonstrate the accelerated Riemann problem solver, we first present a simulation that employs the second approach. The copper projectile is a cylinder with a spherical leading edge. Its thermodynamics is modeled using the Mie-Grüneisen EOS (i.e., Eq. (12)), with parameters $c_0 = 3.93$ km/s, $s = 1.5$, $\rho_0 = 8,960$ kg/m^3, and $\Gamma_0 = 2.12$ [5]. The neon gas is modeled as a perfect gas, with $\gamma = 1.667$. The initial pressure within the copper projectile and neon gas are both 1.0×10^5 Pa. The copper projectile has an initial density of $8,960$ kg/m^3 and neon gas 0.82 kg/m^3. Therefore, the density ratio across the projectile surface reaches 4 orders of magnitude.

Leveraging the cylindrical symmetry of the problem, we use a 2D non-uniform Cartesian mesh. In the most refined region, the element size is around 0.25 mm. Simulations (i), (ii), (iii), and (iv) defined in Sec. 4.1 are performed, each one using 256 CPU cores. The number of integration steps in Simulations (i) and (ii) is set to $8,915$, which is determined by the step size adaptation of Simulation (iii), to achieve an error tolerance of 1.0×10^{-9}. This number of integration steps is greater than those in previous test cases, because of the significant discontinuity across the material interface and the extreme physical conditions.

Fig. 21 presents the pressure and velocity fields obtained from the simulation at $t = 8.87$ µs. As expected, a bow shock appears in front of the projectile, and rarefaction waves are generated at the projectile's back surface. The computation times of the four simulations up to $t = 40.2$ µs are presented in Fig. 22. The solution of exact Riemann problems is accelerated by 87 times (from $1,970.5$ s to 22.7 s), after applying the four methods. In the baseline simulation, over 99% of the flux computation time is spent on solving the exact Riemann problems. As a result, the total advective flux computation time is reduced by a factor of 81.3, when the four acceleration methods are applied. In particular, after using the initial guesses searched from R-tree, the flux computation time further drops by 58.7% (from 58.7 s to 24.2 s). The computational time spent on updating, normalizing, and accessing the R-tree is only 1.08 s. Therefore, in this test case with very significant discontinuity across the projectile surface, the benefits of Method 4 far overtake its cost.

Next, we present a simulation of the second stage of the impact problem. The projectile's velocity is set to 4 km/s. A body-fitted CFD analysis is performed to generate the initial flow field, following [5]. In this case, the projectile is a long copper cylinder with a spherical leading edge. Its radius is 5 mm. The material properties of the copper projectile and the neon gas are identical to the previous test case. The SLG target is modeled using the stiffened gas EOS (i.e., Eq. (7)), with $\gamma = 3.9$, $e_c = 0$, $b = 0$, and $p_c = 2.62$ GPa. Additional details about this test case can be found in [5].

The 2D computational domain is discretized by a non-uniform Cartesian grid with approximately $240,000$ elements. In the most refined region, the grid size is around 0.1 mm. Simulations (i), (ii), and (iii) defined in Sec. 4.1 are performed on 256 CPU cores. The number of integration steps in Simulations (i) and (ii) is set to $8,596$, which is determined by the step size adaptation of Simulation (iii), to achieve an error tolerance of 1.0×10^{-9}. The number of integration steps is greater than those in previous sections, because of the extreme physical conditions generated by the impact.

Fig. 23 presents the pressure, velocity, and temperature fields at four different time instances. The surfaces of the copper projectile and the SLG target are obtained by extracting the zero level set of the two level set functions. At $t = 0.75$ µs, a

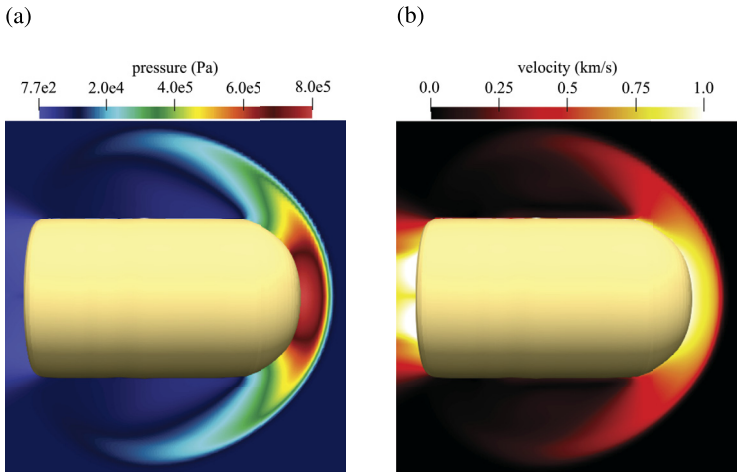

Fig. 21. Simulation of flow around a copper projectile before collision: Pressure and velocity fields at $t = 8.87$ µs. Velocity of projectile: 1 km/s.

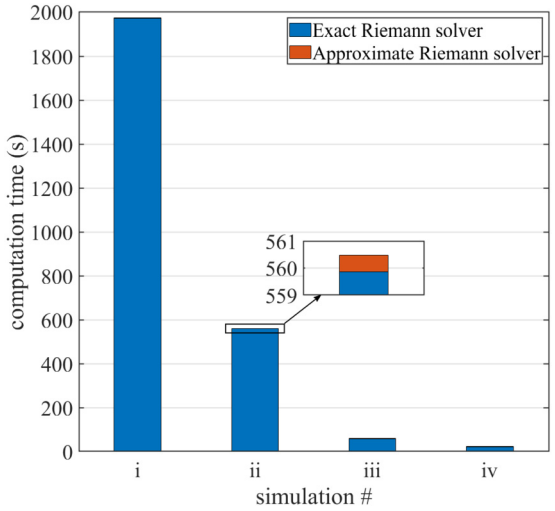

Fig. 22. Simulation of flow around a copper projectile before collision: Comparison of the flux computation time.

time instance shortly after the contact of the two objects, two shock waves propagate in the projectile and the target. The pressure behind the shocks exceeds 20 GPa. Similar pressure magnitudes were observed in previous experiments [65], also using SLG as the target. In the velocity and temperature fields, the bow shock attached to the projectile is formed due to its hypersonic flight. It is not visible in the pressure field because the scale is selected to highlight the much higher pressure in the solid materials. Because of the impact, the neon gas between the target and the projectile are compressed rapidly, resulting in a region of high pressure and high temperature. As time advances, a crater is formed in the target. The SLG material from the rim of the crater is ejected in the neon gas, at a high velocity that amounts to the impact velocity. At $t = 5.93$ µs, the back wall of the target deforms under the impact of the shock wave. In the meantime, the compressive shock wave is reflected as a tensile wave, interfering with the incident shock wave. In the projectile, its leading edge morphs into a "mushroom head" shape, due to the rapid deceleration caused by the initially static target. This type of deformation has been observed in previous studies [66].

In Fig. 24, the pressure time history at a sensor location is plotted. Specifically, we have performed Simulation (iii) with different error tolerances (ϵ_0) for step adaptation. For comparison, the result from Simulation (i) with 8,596 integration steps is also shown. Except for the one with $\epsilon_0 = 10^{-6}$, all the other results match perfectly. This indicates that the flow field predictions can be affected by the error tolerance in the exact Riemann solver; and in certain cases, a very small tolerance — such as 10^{-9} in this case — is needed.

The computation times of Simulations (i)(ii)(iii) up to $t = 0.87$ µs are presented in Fig. 25. After applying Methods (1), (2), and (3), the solution of exact Riemann problems is accelerated by 83 times (from 8,956 s to 108 s). The total advective flux computation time is reduced by a factor of 79.

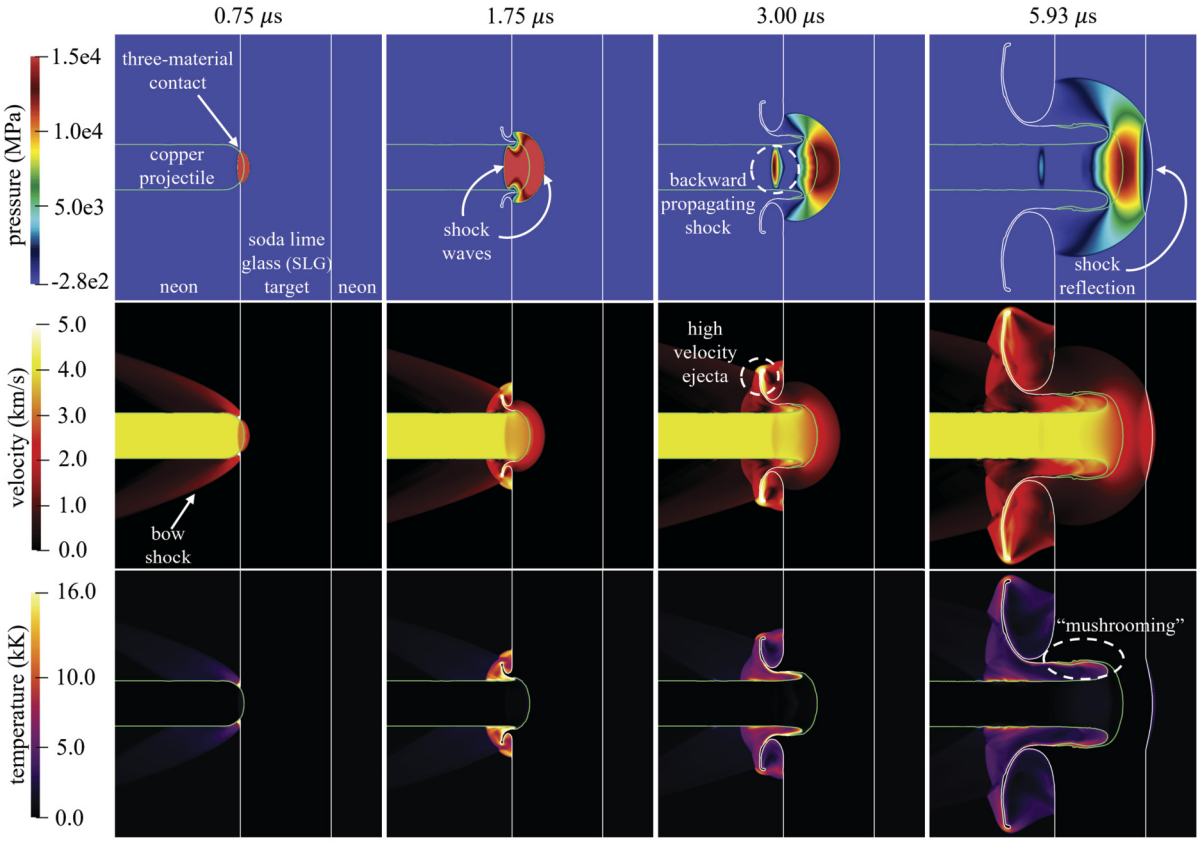

Fig. 23. Hypervelocity impact in a fluid environment: pressure, velocity, and temperature fields. Velocity of projectile: 4 km/s.

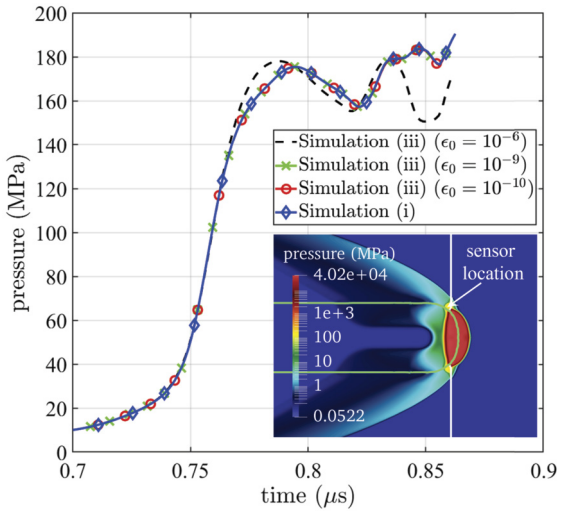

Fig. 24. Hypervelocity impact in a fluid environment: the time history of the pressure at the marked sensor location. The pressure field is from $t = 0.825$ μs.

5. Summary

We consider the numerical solution of compressible multi-material flow problems with sharp interfaces and significant discontinuities. In this context, the advective flux calculator can be viewed as the engine of the solver. Within the engine, the calculation of interfacial fluxes is arguably the most important (yet tricky) part. The idea of locally constructing and solving bimaterial Riemann problems across interfaces has gained popularity in the past two decades. It has a unique advantage in that the discontinuity of both state variables and the thermodynamic EOS are naturally accounted for. This

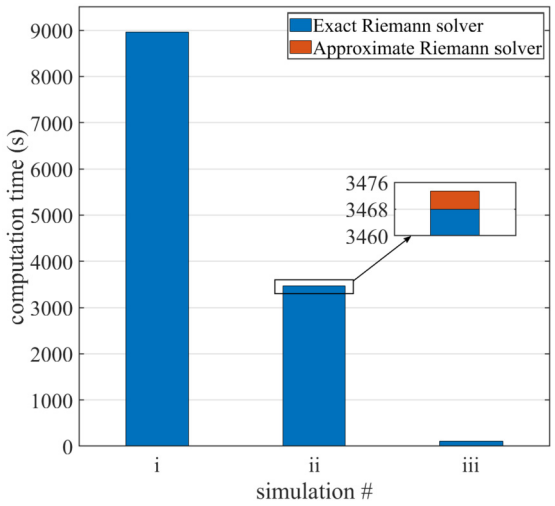

Fig. 25. Hypervelocity impact in a fluid environment: Comparison of the flux computation time of different simulations.

idea entails solving a large number — up to 10^8 in the test cases presented in Sec. 4 — of bimaterial Riemann problems on the fly. On the other hand, the off-the-shelf method for solving Riemann problems (Sec. 3.1) was developed earlier to solve individual problems on a case-by-case basis, and therefore not optimized for computational efficiency. As a result, applying it in multi-material flow solvers leads to a drastic — sometimes prohibitive — increase of computational cost, especially when complicated EOS are involved.

In this paper, we presented four methods to accelerate the solution of bimaterial Riemann problems. The main idea behind these methods is to exploit some special properties of the Riemann problem equations (Sec. 3.2, 3.3, and 3.4), and to store and reuse previous solutions as much as possible (Sec. 3.3 and 3.5). The most expensive part of the solution procedure is the numerical integration of state variables through rarefaction fans. Three of the four methods were designed to accelerate this part, exploiting a change of integration variable (Sec. 3.2), the recycle of integration trajectories (Sec. 3.3), and the adoption of adaptive step size (Sec. 3.4). The computational cost also depends heavily on the quality of the initial guess. In this regard, the fourth acceleration method (Sec. 3.5) stores previous inputs and solutions in a 5D R-tree, and performs a nearest-neighbor search to generate the initial guess for the next Riemann problem to be solved. Compared to previous efforts on the acceleration of Riemann solvers, the four methods presented in this paper do not introduce new approximations, nor do they require additional pre-processing work. Also, they do not depend on or conflict with each other. These methods have been implemented in a standalone Riemann problem solver [62] and a 3D multi-material flow solver [61], both of which are open source and under the GPL (Version 3) license.

We presented four example problems to demonstrate and assess these acceleration methods (Sec. 4). These problems represent different application areas including underwater explosion (Sec. 4.2), laser-induced cavitation (Sec. 4.3), and hypervelocity impact (Sec. 4.1 and 4.4). They involve some major computational challenges such as large interface motion and deformation, multiple (> 2) interfaces in contact, drastically different thermodynamic relations, and significant jumps of state variables across interfaces. For two example problems (Sec. 4.1 and 4.2), the proposed numerical methods are verified against reference solutions. For all the problems, the speedup obtained by the proposed acceleration methods was found to be significant. The time spent on solving exact Riemann problems was reduced by a factor of 37 to 87. As a result, the total time spent on calculating advective fluxes was also reduced by a factor of 18 to 81. Among the four acceleration methods, the first three were found to be effective in all the test cases. The last one (presented in Sec. 3.5) was found to be effective when the local density jump across the material interface is significant. While the simulations presented in this paper were performed using a specific solver that implements the FIVER method, the proposed acceleration methods were not tailored for this solver. They are generally applicable to multi-material and multiphase flow solvers that utilize the exact solution of Riemann problems; and similar acceleration effects can be expected.

CRediT authorship contribution statement

Wentao Ma: Conceptualization, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Xuning Zhao:** Methodology, Software, Validation. **Shafquat Islam:** Methodology, Software, Validation. **Aditya Narkhede:** Methodology, Software. **Kevin Wang:** Conceptualization, Funding acquisition, Investigation, Methodology, Software, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

The authors gratefully acknowledge the support of the National Science Foundation (NSF) under Award CBET-1751487, the support of the Office of Naval Research (ONR) under Award N00014-19-1-2102, and the support of the National Institutes of Health (NIH) under Award 2R01-DK052985-24A1. W.M. and K.W. also acknowledge the support of U.S. Department of Transportation (DOT) Pipeline and Hazardous Materials Safety Administration under contract number 693JK32250007CAAP.

References

[1] X. Zhao, W. Ma, K. Wang, Simulating laser-fluid coupling and laser-induced cavitation using embedded boundary and level set methods, J. Comput. Phys. 472 (2023) 111656.
[2] G. Xiang, B. Wang, Numerical study of a planar shock interacting with a cylindrical water column embedded with an air cavity, J. Fluid Mech. 825 (2017) 825–852.
[3] R.W. Houim, K.K. Kuo, A ghost fluid method for compressible reacting flows with phase change, J. Comput. Phys. 235 (2013) 865–900.
[4] J. Krimmel, T. Colonius, M. Tanguay, Simulation of the effects of cavitation and anatomy in the shock path of model lithotripters, Urol. Res. 38 (2010) 505–518.
[5] S.T. Islam, W. Ma, J.G. Michopoulos, K. Wang, Plasma formation in ambient fluid from hypervelocity impacts, Extrem. Mech. Lett. 58 (2023) 101927.
[6] J. VonNeumann, R.D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, J. Appl. Phys. 21 (3) (1950) 232–237.
[7] F. Vilar, C.-W. Shu, P.-H. Maire, Positivity-preserving cell-centered lagrangian schemes for multi-material compressible flows: from first-order to high-orders. part ii: the two-dimensional case, J. Comput. Phys. 312 (2016) 416–442.
[8] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, S. Galera, Reale: a reconnection-based arbitrary-lagrangian–eulerian method, J. Comput. Phys. 229 (12) (2010) 4724–4761.
[9] A. Marbœuf, A. Claisse, P. Le Tallec, Conservative and entropy controlled remap for multi-material ale simulations with space-staggered schemes, J. Comput. Phys. 390 (2019) 66–92.
[10] J. Banks, D. Schwendeman, A. Kapila, W. Henshaw, A high-resolution Godunov method for compressible multi-material flow on overlapping grids, J. Comput. Phys. 223 (1) (2007) 262–297.
[11] K. Kitamura, M.-S. Liou, C.-H. Chang, Extension and comparative study of ausm-family schemes for compressible multiphase flow simulations, Commun. Comput. Phys. 16 (3) (2014) 632–674.
[12] F. Gibou, R. Fedkiw, S. Osher, A review of level-set methods and some recent applications, J. Comput. Phys. 353 (2018) 82–109.
[13] D.J. Benson, Volume of fluid interface reconstruction methods for multi-material problems, Appl. Mech. Rev. 55 (2) (2002) 151–165.
[14] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2) (2001) 708–759.
[15] T. Liu, B. Khoo, C. Wang, The ghost fluid method for compressible gas–water simulation, J. Comput. Phys. 204 (1) (2005) 193–221.
[16] E. Johnsen, F. Ham, Preventing numerical errors generated by interface-capturing schemes in compressible multi-material flows, J. Comput. Phys. 231 (17) (2012) 5705–5717.
[17] E. van Brummelen, B. Koren, A pressure-invariant conservative Godunov-type method for barotropic two-fluid flows, J. Comput. Phys. 185 (1) (2003) 289–308.
[18] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (2) (1999) 457–492.
[19] R. Fedkiw, A. Marquina, B. Merriman, An isobaric fix for the overheating problem in multimaterial compressible flows, J. Comput. Phys. 148 (1999) 545–578.
[20] T. Liu, B. Khoo, K. Yeo, Ghost fluid method for strong shock impacting on material interface, J. Comput. Phys. 190 (2) (2003) 651–681.
[21] S.K. Sambasivan, H.S. UdayKumar, Ghost fluid method for strong shock interactions part 1: fluid-fluid interfaces, AIAA J. 47 (12) (2009) 2907–2922.
[22] C.W. Wang, T.G. Liu, B.C. Khoo, A real ghost fluid method for the simulation of multimedium compressible flow, SIAM J. Sci. Comput. 28 (1) (2006) 278–302.
[23] T.D. Aslam, A level set algorithm for tracking discontinuities in hyperbolic conservation laws ii: systems of equations, J. Sci. Comput. 19 (1–3) (2003) 37–62.
[24] H. Terashima, G. Tryggvason, A front-tracking method with projected interface conditions for compressible multi-fluid flows, Comput. Fluids 39 (10) (2010) 1804–1814.
[25] J.-P. Cocchi, R. Saurel, A Riemann problem based method for the resolution of compressible multimaterial flows, J. Comput. Phys. 137 (2) (1997) 265–298.
[26] D. Igra, K. Takayama, A high resolution upwind scheme for multi-component flows, Int. J. Numer. Methods Fluids 38 (10) (2002) 985–1007.
[27] C. Farhat, A. Rallu, S. Shankaran, A higher-order generalized ghost fluid method for the poor for the three-dimensional two-phase flow computation of underwater implosions, J. Comput. Phys. 227 (16) (2008) 7674–7700.
[28] C. Farhat, J.-F. Gerbeau, A. Rallu, Fiver: a finite volume method based on exact two-phase Riemann problems and sparse grids for multi-material flows with large density jumps, J. Comput. Phys. 231 (19) (2012) 6360–6379.
[29] W. Bo, X. Liu, J. Glimm, X. Li, A robust front tracking method: verification and application to simulation of the primary breakup of a liquid jet, SIAM J. Sci. Comput. 33 (4) (2011) 1505–1524.
[30] A. Jafarian, A. Pishevar, An exact multiphase Riemann solver for compressible cavitating flows, Int. J. Multiph. Flow 88 (2017) 152–166.
[31] K.G. Wang, P. Lea, C. Farhat, A computational framework for the simulation of high-speed multi-material fluid-structure interaction problems with dynamic fracture, Int. J. Numer. Methods Eng. 104 (7) (2015) 585–623.

[32] A. Main, X. Zeng, P. Avery, C. Farhat, An enhanced fiver method for multi-material flow problems with second-order convergence rate, J. Comput. Phys. 329 (2017) 141–172.
[33] J. Ho, C. Farhat, Discrete embedded boundary method with smooth dependence on the evolution of a fluid-structure interface, Int. J. Numer. Methods Eng. 122 (19) (2021) 5353–5383.
[34] C. Farhat, K. Wang, A. Main, S. Kyriakides, L.-H. Lee, K. Ravi-Chandar, T. Belytschko, Dynamic implosion of underwater cylindrical shells: experiments and computations, Int. J. Solids Struct. 50 (19) (2013) 2943–2961.
[35] W. Ma, X. Zhao, C. Gilbert, K. Wang, Computational analysis of bubble–structure interactions in near-field underwater explosion, Int. J. Solids Struct. 242 (2022) 111527.
[36] K.G. Wang, Multiphase fluid-solid coupled analysis of shock-bubble-stone interaction in shockwave lithotripsy, Int. J. Numer. Methods Biomed. Eng. 33 (10) (2017), https://doi.org/10.1002/cnm.2855.
[37] S. Cao, Y. Zhang, D. Liao, P. Zhong, K.G. Wang, Shock-induced damage and dynamic fracture in cylindrical bodies submerged in liquid, Int. J. Solids Struct. 169 (2019) 55–71.
[38] S. Cao, G. Wang, O. Coutier-Delgosha, K. Wang, Shock-induced bubble collapse near solid materials: effect of acoustic impedance, J. Fluid Mech. 907 (2021).
[39] G. Xiang, X. Ma, C. Liang, H. Yu, D. Liao, G. Sankin, S. Cao, K. Wang, P. Zhong, Variations of stress field and stone fracture produced at different lateral locations in a shockwave lithotripter field, J. Acoust. Soc. Am. 150 (2) (2021) 1013–1029, https://doi.org/10.1121/10.0005823.
[40] E.F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction, Springer Science & Business Media, 2013.
[41] J.R. Kamm, An exact, compressible one-dimensional Riemann solver for general, convex equations of state, Tech. Rep., Los Alamos National Lab. (LANL), Los Alamos, NM (United States), 2015.
[42] A. Guttman, R-trees: a dynamic index structure for spatial searching, in: ACM SIGMOD Conference, 1984.
[43] Y. Manolopoulos, A. Nanopoulos, A.N. Papadopoulos, Y. Theodoridis, R-Trees: Theory and Applications, 1st edition, Springer, London, 2006.
[44] P. Colella, H.M. Glaz, Efficient solution algorithms for the Riemann problem for real gases, J. Comput. Phys. 59 (2) (1985) 264–289.
[45] A. Rallu, A multiphase fluid-structure computational framework for underwater imploison problems, Ph.D. thesis, Stanford University, 2009.
[46] O. Le Métayer, R. Saurel, The noble-Abel stiffened-gas equation of state, Phys. Fluids 28 (2016) 046102, https://doi.org/10.1063/1.4945981.
[47] R. Menikoff, JWL equation of state, Tech. Rep., Los Alamos National Lab. (LANL), Los Alamos, NM (United States), 2015.
[48] A.C. Robinson, The mie-gruneisen power equation of state, Tech. Rep., Sandia National Lab. (SNL-NM), Albuquerque, NM (United States), 2019.
[49] J. Sanchez, Inelastic equation of state for solids, Comput. Methods Appl. Mech. Eng. 375 (2021) 113622.
[50] R. Menikoff, B.J. Plohr, The Riemann problem for fluid flow of real materials, Rev. Mod. Phys. 61 (1989) 75–130.
[51] A. Main, X. Zeng, P. Avery, C. Farhat, An enhanced FIVER method for multi-material flow problems with second-order convergence rate, J. Comput. Phys. 329 (2017) 141–172.
[52] V. Rusanov, On difference schemes of third order accuracy for nonlinear hyperbolic systems, J. Comput. Phys. 5 (3) (1970) 507–516.
[53] X. Hu, N.A. Adams, G. Iaccarino, On the hllc Riemann solver for interface interaction in compressible multi-fluid flow, J. Comput. Phys. 228 (17) (2009) 6572–6589.
[54] E. Fehlberg, Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems, Tech. Rep., National Aeronautics and Space Administration. (NASA), Washington, D.C, (United States), 1969.
[55] J.R. Cash, A.H. Karp, A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides, ACM Trans. Math. Softw. 16 (3) (1990) 201–222.
[56] S. Chapra, R. Canale, Numerical Methods for Engineers, 7th edition, McGraw-Hill Higher Education, New York, NY, 2015.
[57] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes: The Art of Scientific Computing, 3rd edition, Cambridge University Press, 2007.
[58] N. Roussopoulos, S. Kelley, F. Vincent, Nearest neighbor queries, SIGMOD Rec. 24 (2) (1995) 71–79.
[59] K.L. Cheung, A.W.-C. Fu, Enhanced nearest neighbour search on the r-tree, SIGMOD Rec. 27 (3) (1998) 16–21.
[60] Boost Generic Geometry Library, https://www.boost.org/doc/libs/1_82_0/libs/geometry/doc/html/index.html.
[61] M2C solver, Github, https://github.com/kevinwgy/m2c. (Accessed 17 February 2023).
[62] One-dimensional two-phase Riemann problem solver, Github, https://github.com/kevinwgy/riemann. (Accessed 17 February 2023).
[63] K.-M. Shyue, An efficient shock-capturing algorithm for compressible multicomponent problems, J. Comput. Phys. 142 (1) (1998) 208–242.
[64] D.S. Ho, D. Scialabba, R.S. Terry, X. Ma, J. Chen, G.N. Sankin, G. Xiang, R. Qi, G.M. Preminger, M.E. Lipkin, P. Zhong, The role of cavitation in energy delivery and stone damage during laser lithotripsy, J. Endourolog. 35 (6) (2021) 860–870.
[65] T. Kobayashi, T. Sekine, O. Fat'yanov, E. Takazawa, Q. Zhu, Radiation temperatures of soda-lime glass in its shock-compressed liquid state, J. Appl. Phys. 83 (3) (1998) 1711–1716.
[66] Y. Sun, C. Shi, Z. Liu, D. Wen, Theoretical research progress in high-velocity/hypervelocity impact on semi-infinite targets, Shock Vib. 2015 (2015) 1–15, https://doi.org/10.1155/2015/265321.