MFC: An open-source high-order multi-component, multi-phase, and multi-scale compressible flow solver

Spencer H. Bryngelson,[1,*] Kevin Schmidmayer,[1] Vedran Coralic,[2] Jomela C. Meng,[3] Kazuki Maeda,[4] and Tim Colonius[1]

[1]*Division of Engineering and Applied Science, California Institute of Technology, Pasadena, CA 91125, USA*
[2]*Prime Air, Amazon Inc, Seattle, WA 98108, USA*
[3]*Bosch Research and Technology Center, Sunnyvale, CA 94085, USA*
[4]*Department of Mechanical Engineering, University of Washington, Seattle, WA 98195, USA*

The MFC is an open-source tool for solving multi-component, multi-phase, and bubbly flows. It is capable of efficiently solving a wide range of physically relevant flows, including droplet atomization, shock–bubble interaction, and gas bubble cavitation. We present the 5- and 6-equation thermodynamically-consistent diffuse interface models we use to handle such flows, which are coupled to high-order interface capturing methods, HLL-type Riemann solvers, and TVD time-integration schemes that capable of simulating unsteady flows with strong shocks. The numerical methods are implemented in a flexible, modular framework that is amenable to future development. The MFC is validated via comparisons to experimental results for shock–bubble aspherical collapses and shock–droplet and shock–water-cylinder interactions. We verify that our numerical method is indeed high-order accurate and free of spurious oscillations by considering isentropic vortex, material-interface advection, and gas–liquid Riemann problems.

Keywords: computational fluid dynamics, multiphase flow, diffuse interface method, compressible flow, ensemble averaging, bubble dynamics

PROGRAM SUMMARY

Title of program: MFC (Multi-component Flow Code)
Licensing provisions: GNU General Public License 3
Permanent link to code/repository: `https://mfc-caltech.github.io`
Operating systems under which the program has been tested: UNIX, Mac OSX, Windows
Programming language used: Fortran 90 and Python
Nature of problem: Computer simulation of multiphase flows requires careful physical model selection and sophisticated treatment of spatial and temporal derivatives in order to keep solutions both thermodynamically consistent and free of spurious oscillations. Further, keeping such methods high-order accurate is necessary to permit reasonable simulation times. These problems are particularly challenging for compressible flows, which can consist of shock waves, though they are of great importance in numerous biomedical and defense applications.
Solution method: The present software incorporates multiple state-of-the-art physical models and numerical schemes for treatment of compressible multi-phase and multi-component flows. Additional physical effects and sub-grid models are included, such as an ensemble-averaged bubbly flow model. The architecture was designed such that additional development is simple.

[*] Corresponding author: spencer@caltech.edu

I. INTRODUCTION

The multicomponent flow code (MFC) is an open source high-order interface-capturing solver for multiphase, multicomponent, and bubbly flows. Multiphase flows include the liquid/gas configurations we consider herein, and multicomponent configurations include gases or liquids with different physical properties.

Multiphase flows are commonplace in engineering applications. For example, bubbly, cavitating flow phenomena are of critical importance in many biomedical settings, including the development of artificial heart valves and pumps [10], understanding of injury due to blast trauma [47, 67], and improving shock- and burst-wave lithotripsy treatments [16, 36, 65]. Such bubbles are also pervasive when considering liquid flows around hydrofoils, submarines, and high-velocity projectiles [61, 63, 69] and during underwater explosions [22, 43]. Unfortunately, the cavitation that occurs in many of these cases is detrimental, causing noise and material deterioration [75, 85]. They also occur naturally and are known to occur during mantis [59, 60] and pistol shrimp strikes [7, 46] as well as in vascular plant tissues [64]. Another flow of great interest is the evolution of flowing droplets. The study of droplet aero-breakup is motivated by the ubiquity of high-speed droplet applications, such as the atomization of a droplet in combustion systems [54, 55, 72], the raindrop damaging erosion of aircraft surfaces during supersonic flight [21, 40], and shock-wave attenuation in nuclear blasts [13]. The associated phenomena often include the dissemination of secondary droplets, though a definitive understanding of this breakup remains an open challenge [45]. Of considerable importance is also the formation and subsequent fragmentation (via, e.g. Rayleigh–Plateau instability) of liquid jets. Simulation of such configurations are motivated by applications associated with microjets, including needle-free injections into the skin [76].

Robust simulation of such flow phenomena requires a numerical methods that maintain conservation at the discrete level, suppresses oscillations near discontinuities, and preserves numerical stability. For such simulations to be computational efficient, the methods used should also be high-order accurate away from discontinuitiies. The schemes that have the potential to achieve these are generally divided into interface-tracking and interface-capturing methods Fuster [24]. Interface-tracking methods treat material interfaces are sharp features in the flow. This is naturally advantageous as two fluids that share the interface can have different equations of state and interfacial physics are straightforward to implement. Unfortunately, interface tracking schemes do not naturally enforce discrete and total conservation properties, though modern examples are staring to include this [12, 20, 25, 50]. Interface-tracking in generally can be subdivided into volume of fluid methods, [20, 25, 41], free-Lagrange methods [5, 82], front-tracking methods [14, 26], and level-set/ghost fluid methods [2, 12, 30, 34, 49, 50, 58]. Interface capturing methods instead treat interfaces as discontinuities in material properties, generally represented by a set of advected volume fractions. Their main advantage is that achieving conservation is simple; indeed it is sufficient to simply solve the continuity, momentum and energy equations in conservative form. Unfortunately, these methods relaxes the requirements of interfaces to be sharp, and thus material interfaces diffuse due to numerical errors over time, generally to a small though finite region [39, 62, 73]. This results in physical mixture properties that must be treated in a thermodynamically consistent way and advected in a numerically consistent manner in order to avoid spurious oscillations near interfaces; fortunately such methods have already been developed [38, 39] Diffuse interfaces also mean that a numerically coherent interface does not exist, and thus interfacial physics are more challenging to implement. However, it has shown possible to implement both interfacial heat transfer [52] and capillary effects via conservative formulations [53, 62, 72].

Herein, we chose to use an interface capturing scheme as discrete conservation is of principal interest to many of the key phenomenologies outlined above. Further, they are generally more efficient than interface-tracking methods [57] and their complexity does not increase with simulation dimensionality or the number of components. For this we also need a governing set of physical model equations. It is known that the 5- and 6-equation models [3, 42, 52, 70] are sufficient for representing the physical disequilibriums of interest. These methods are mass- and pressure-disequilibrium models, respectively, though ultimately they are mechanical equilibrium models as they relax disequilibriums at the end of each time step. They conserve the mass of each fluid in the flow and advects the volume fraction of each component. These methods explicitly tracking each fluid in the flow, which facilitates the treatment of mixtures composed of more than two fluids and implementation of interfacial physics. The numerical method we implement is discretely conservative and solves the conservative form of the compressible flow equations. In order to maintain a non-oscillatory behavior near material interfaces, the material advection equations must be written in a form that is consistent with the conservative governing equations [1]. For this, mass fractions, volume fractions, or specific functions of the equation of state parameters are typically evolved. The equations are motion are then closed by a thermodynamically consistent set of mixture rules, as derived by assuming the mixtures are in mechanical equilibrium [3]. The governing equations must also be cast in a finite-volume framework and discretized with a non-oscillatory spatial and temporal method, with the primitive state variables, rather than the conservative ones, spatially reconstructed [38]. In order to achieve high-order accuracy while maintaining non-oscillatory behavior away from material interfaces, high-order shock capturing methods are generally used. In particular, it is known that WENO spatial reconstructions work well [6, 19, 32, 37], particularly when they are coupled with an HLL-type approximate Riemann solver [80, 81] and a TVD time stepper [28].

Herein we detail version 1.0 of the MFC. In section II we present an overview of the what is included in the public MFC package, including its organization, features, and logistics. In section **??** we briefly describe the data structures used by the MFC. The physical models employed by MFC are presented in section III, including the associated mixture rules, governing equations, equations of state, and our implementation of an ensemble-averaged bubbly flow model. The numerical methods used to solve the associated equations are presented in section IV. A series of example cases simulated using the MFC are discussed in section V; these provide a verification and validation of our methods, along with a demonstration of the capabilities of the package. Lastly, a discussion of the outlook of the MFC is included in section VII.

II. OVERVIEW AND FEATURES

A. Package, installation, and testing

The MFC is freely available at `mfc-caltech.github.io`. The source codes are written using Fortran 90 with MPI bindings used for parallel communication; the input files are generated using Python scripts. Installation of the MFC requires the FFTW package for cylindrical coordinate treatment, and, optionally, Silo and its dependencies for post treatment of data files. We only provide the FFTW package with the MFC due to size constraints, though the others can be readily found online.

The MFC package includes several directories; their organization and descriptions are shown in table I. New users should consult the "CONFIGURE" and "INSTALL" files for instructions on how to compile MFC. In brief, the user is required to ensure that an MPI Fortran compiler and Python are loaded. In terms of libraries, the FFTW package must be located by the makefiles; this can be done by pointing Makefile.user to the correct local location or by installing the included FFTW distribution in the installers directory. HDF5 and Silo, which are required for the post_process code, require a more involved installation. Specific directions for both Mac OSX (via Homebrew)

Name	**Description**
doc/	Documentation
installers/	Package installers: Includes FFTW
lib/	Libraries
src/	Source code
______ master_scripts/	Python modules and dictionaries, optional source files
____ pre_process_code/	Generates initial conditions and grids
_____ simulation_code/	Flow solver
___ post_process_code/	Processes simulation data
tests/	Test cases to ensure software is operating as intended
AUTHORS	List of contributors and their contact information
CONFIGURE	Package configuration guide
COPYRIGHT	Copyright notice
INSTALL	Installation guide
LICENSE	The GNU public license file
Makefile.in	Makefile input; generally does not need to be modified
Makefile.user	User inputs for compilation; requires attention from user
Makefile	Targets: all (default), pre_process, simulation, post_process, check

TABLE I. Descriptions of the files and directories included in the MFC.

and Unix distributions are included in the "CONFIGURE" file.

Once the software has been built, the "check" target of make should be called; it runs multiple tests (which are located in the tests directory) to ensure that the functionalities of the MFC are working as intended.

B. Features

The MFC features a fully parallel environment using message passing interfaces (MPI). Computationally, it includes structured cartesian and cylindrical grids with mesh stretching available. At domain edges characteristic-based Thompson boundary conditions as well as traditional periodic and free-slip options. From a modeling point of view, the number of components and phases are entirely flexible and the 5- and 6-equation models are implemented, as discussed in section III. Ensemble-averaged dilute bubbly flow modeling is also available, including options for Gilmore and Keller–Miksis single-bubble options (see section III C). The numerical methods are discussed in section IV, they include include 1-, 3-, and 5-th order accurate WENO reconstructions on optionally the primitive, conservative, or characteristic variables. Within each finite volume, high-order evaluation is available for the cell-averaged variables via Gaussian quadrature for multi-dimensional problems. The shock-capturing schemes are paired with optionally either HLL, HLLC, or exact Riemann solvers. For time-stepping, 1–5-th order accurate Runge–Kutta time stepping available. Additional options include support for one-way acoustic generating sources. These features will of course evolve with time. Additional features are available, though are currently considered experimental.

C. Software structure

The codes are modular and their full descriptions and available on the Doxygen documentation. We briefly describe them here. The software is executed in this order [pre, sim post], though the post processing step is structured to be optional in most cases.

1. *Pre processing*

The pre_process code generates initial conditions and spatial grids from the physical patches specified in the Python input file, and outputs them into binary files that are subsequently read by the simulation code. Specifically, this involves allocating and writing either a cartesian or cylindrical mesh, with the option of mesh stretching, according to the input parameters. The specified physical variables for each patch are transformed into their conservative form and written in a way consistent with the mesh. Individual Fortran modules are used to conduct each component; they read input values and export mesh and initial condition files, assign then distributing global variables via MPI, perform variable transformations, generate grids, parse and assign patch types, and check that specified input variables are physically consistent and that specified options do not contradict each other.

2. *Simulation*

The simulation code, given the initial condition files generated by the pre_process code, solves the corresponding governing flow equations with the specified boundary conditions using a high-order interface-capturing numerical method. The model flow equations are described in section III and the numerical methods are discussed in section IV. Simulation continues according to the number of time steps indicated. The simulation code outputs restart files that can be used to either restart the simulation or post_process the associated data, as well as run-time information, and, optionally, human-readable output data. Again, individual Fortran modules are used to conduct each software component, which includes reading and exporting data and grid files, performing Fourier transforms, assigning and distributing global variables via MPI, performing variable transformations, computing time and spatial derivatives using WENO and the Riemann solver specified, computing boundary values, including ensemble-averaged bubbly flow physics, and checking that the input variables are valid.

3. *Post processing*

The post_process code reads simulation data and outputs HDF5 files that include variables and derived variables as specified in the input file. Since the simulation code has an option to output human readable data, post processing is not essential to the usage of MFC, but is a useful tool, especially for large or parallel data structures. Specifically, it reads the restart files output by the simulation code and distinct time intervals, and computes the derived quantities necessary. The HDF5 database is then generated and exported. Again, individual Fortran modules perform the associated tasks, including reading data, parameter conversion, assigning and distributing global MPI variables, computing Fourier transforms, exporting HDF5 Silo databases, and checking that the input parameters are consistent.

	Name	Description
Input	input.py	Input parameters
	____ pre_process.inp	Pre-process input parameters, auto-generated
	_____ simulation.inp	Simulation input parameters, auto-generated
	___ post_process.inp	Post-process input parameters, auto-generated
Output	run_time.inf	Run-time information including current simulation time and CFL
	D/	Formatted simulation output files
	p_all/	Binary simulation restart files (`parallel_IO` disabled)
	restart_data/	Lustre restart files (`parallel_IO` enabled)
	silo_HDF5/	Silo post-process files (`format = 1`)
	binary/	Binary post-process files (`format = 2`)

TABLE II. Input/output files in the case directory.

D. Description of input/output files

We next describe the contents of a case-specific simulation directory and its logistics. The specific file structure is shown in table II. A Python script (input.py) is used to generate the input files ($*$.inp) for the source codes and execute an MFC component (one of pre_process, simulation, or post_process) either in the active window or a submitted batch script. This Python file contains the input parameters available for the MFC. The MFC, depending upon the component used and options selected, will generate a number of files and directories. If enabled, the run_time.inf file will be generated by the simulation component and includes details about the current time step, simulation time, and stability criterion. Directories that contain binary restart data and output files for visualization or further post treatment are generated, again depending upon the options, by the simulation and post_process components.

III. PHYSICAL MODEL AND GOVERNING EQUATIONS

The mechanical-equilibrium compressible multicomponent flow models we present can be written as

$$\frac{\partial \boldsymbol{q}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{q}) + \boldsymbol{h}(\boldsymbol{q}) \nabla \cdot \boldsymbol{u} = \boldsymbol{s}(\boldsymbol{q}), \tag{1}$$

where $\boldsymbol{q}$ is the state vector, $\boldsymbol{F}$ is the flux tensor, $\boldsymbol{u}$ is the velocity field, and $\boldsymbol{h}$ and $\boldsymbol{s}$ are non-conservative quantities we describe subsequently.

A. 5-equation model

We first introduce our implementation of the thermodynamically consistent mechanical-equilibrium model of Kapila et al. [42]. Our multi-component implementation can be used for N_k components, though we present a two-component ($N_k = 2$) configuration here for demonstration purposes. It consists of five partial differential equations as

$$\boldsymbol{q} = \begin{bmatrix} \alpha_1 \\ \alpha_1\rho_1 \\ \alpha_2\rho_2 \\ \rho\boldsymbol{u} \\ \rho E \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \alpha_1\boldsymbol{u} \\ \alpha_1\rho_1\boldsymbol{u} \\ \alpha_2\rho_2\boldsymbol{u} \\ \rho\boldsymbol{u}\otimes\boldsymbol{u} + p\boldsymbol{I} \\ (\rho E + p)\,\boldsymbol{u} \end{bmatrix}, \quad \boldsymbol{h} = \begin{bmatrix} -\alpha_1 - K \\ 0 \\ 0 \\ \boldsymbol{0} \\ 0 \end{bmatrix}, \quad \boldsymbol{s} = \boldsymbol{0}, \tag{2}$$

where ρ, $\boldsymbol{u}$, and p are the mixture density, velocity, and pressure, respectively, and α_k is the volume fraction of component k. The mixture total and internal energies are $E = e + \|\boldsymbol{u}\|^2/2$ and

$$e = \sum_{k=1}^{N_k} Y_k e_k\left(\rho_k, p\right), \tag{3}$$

respectively, where $Y_k = \alpha_k\rho_k/\rho$ are the mass fractions. We close (3) using the stiffened-gas equation of state, which is chosen for its ability to faithfully model both liquids and gases [56]; for component k it is

$$p_k = (\gamma_k - 1)\rho_k e_k - \gamma_k \pi_{\infty,k}, \tag{4}$$

where γ is the specific heat ratio and π_∞ is the liquid stiffness (gases have $\pi_\infty = 0$) [48]. For liquids, these are usually interpreted as fitted parameters from shockwave Hugoniot data [27, 51]. The speed of sound of each component is then

$$c_k = \sqrt{\frac{\gamma_k(p_k + \pi_{\infty,k})}{\rho_k}}. \tag{5}$$

The K term in $\boldsymbol{h}$ of (2) represents expansion and compression in mixture regions. For an $N_k = 2$ configuration it is

$$K = \frac{\rho_2 c_2^2 - \rho_1 c_1^2}{\frac{\rho_2 c_2^2}{\alpha_2} + \frac{\rho_1 c_1^2}{\alpha_1}} \tag{6}$$

and the mixture speed of sound c follows as the so-called Wood speed of sound [84, 86]

$$\frac{1}{\rho c^2} = \sum_{k=1}^{N_k} \frac{\alpha_k}{\rho_k c_k^2}. \tag{7}$$

Some models, such as those of Allaire et al. [3] and Massoni et al. [52], do not include this K term in (2) and thus do not strictly obey the second-law of thermodynamics, nor reproduce the correct mixture speed of sound. These properties are known to be important for many problems, including those of interest here, such as the cavitation of gas bubbles [71].

The equations are closed by the usual set of mixture rules

$$1 = \sum_{k=1}^{N_k} \alpha_k, \quad \rho = \sum_{k=1}^{N_k} \alpha_k\rho_k, \quad \text{and} \quad \rho\varepsilon = \sum_{k=1}^{N_k} \rho_k\varepsilon_k. \tag{8}$$

B. 6-equation model

While the 5-equation model described in section III A is efficient and represents the correct physics, the $K\nabla\cdot\boldsymbol{u}$ term that makes the model thermodynamically consistent can sometimes introduce numerical instabilities [71]. In such cases, a pressure-disequilibrium has been shown to be preferable [71]. We also support the 6-equation pressure-disequilibrium model of Saurel et al. [70], which for a two-component configuration is expressed as

$$\boldsymbol{q} = \begin{bmatrix} \alpha_1 \\ \alpha_1\rho_1 \\ \alpha_2\rho_2 \\ \rho\boldsymbol{u} \\ \alpha_1\rho_1 e_1 \\ \alpha_2\rho_2 e_2 \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \alpha_1\boldsymbol{u} \\ \alpha_1\rho_1\boldsymbol{u} \\ \alpha_2\rho_2\boldsymbol{u} \\ \rho\boldsymbol{u}\otimes\boldsymbol{u} + p\boldsymbol{I} \\ \alpha_1\rho_1 e_1\boldsymbol{u} \\ \alpha_2\rho_2 e_2\boldsymbol{u} \end{bmatrix}, \quad \boldsymbol{h} = \begin{bmatrix} -\alpha_1 \\ 0 \\ 0 \\ \boldsymbol{0} \\ \alpha_1 p_1 \\ \alpha_2 p_2 \end{bmatrix}, \quad \boldsymbol{s} = \begin{bmatrix} \mu\delta p \\ 0 \\ 0 \\ \boldsymbol{0} \\ -\mu p_I\delta p \\ \mu p_I \delta p \end{bmatrix}, \tag{9}$$

where $\boldsymbol{s}$ represents the relaxation of pressures between components with coefficient μ. The interfacial pressure is

$$p_I = \frac{z_2 p_1 + z_1 p_2}{z_1 + z_2}, \tag{10}$$

where $z_k = \rho_k c_k$ is the acoustic impedance of component k and

$$\delta p = p_1 - p_2, \tag{11}$$

is the pressure difference. Since $p_1 \neq p_2$, the total energy equation of the mixture is replaced by the internal-energy equation for each component. The mixture speed of sound is defined according to

$$c^2 = \sum_{k=1}^{2} Y_k c_k^2, \tag{12}$$

though after applying the numerical infinite pressure-relaxation procedure detailed in section IV C the effective mixture speed of sound matches (7).

C. Bubbly flow model

1. *Implementation*

The MFC includes support for the ensemble-phase-averaged bubbly flow model of Zhang and Prosperetti [87], and our implementation of it matches that of Bryngelson et al. [11]. The bubble population has void fraction α_b, which is assumed to be small, and the carrier components have mixture pressure p_l. The equilibrium radii of the bubble population are represented discretely as $\boldsymbol{R}_o$, which are N_{bin} bins of an assumed log-normal PDF with standard deviation σ_p [17]. The instantaneous bubble radii are a function of these equilibrium states as $\boldsymbol{R}(\boldsymbol{R}_o) = \{R_1, R_2, \dots, R_{N_{\text{bin}}}\}$. The total mixture pressure is modified as

$$p = (1-\alpha_b)p_l + \alpha_b \left(\frac{\overline{\boldsymbol{R}^3 \boldsymbol{p}_{bw}}}{\overline{\boldsymbol{R}^3}} - \rho \frac{\overline{\boldsymbol{R}^3 \dot{\boldsymbol{R}}^2}}{\overline{\boldsymbol{R}^3}} \right), \tag{13}$$

where $\dot{\boldsymbol{R}}$ are the bubble radial velocities and $\boldsymbol{p}_{bw}$ are the bubble wall pressures. Overbars $\bar{\cdot}$ denote the usual moments with respect to the log-normal PDF. The bubble void fraction is advected as

$$\frac{\partial \alpha_b}{\partial t} + \boldsymbol{u} \cdot \nabla \alpha_b = 3\alpha_b \frac{\overline{\boldsymbol{R}^2 \dot{\boldsymbol{R}}}}{\overline{\boldsymbol{R}^3}}, \tag{14}$$

and the bubble dynamic variables are evolved as

$$\frac{\partial n \boldsymbol{\phi}}{\partial t} + \nabla \cdot (n \boldsymbol{\phi} \boldsymbol{u}) = n \dot{\boldsymbol{\phi}}, \tag{15}$$

where $\boldsymbol{\phi} \equiv \left\{ \boldsymbol{R}, \dot{\boldsymbol{R}}, \boldsymbol{p}_b, \boldsymbol{m}_v \right\}$ (see section III C 2) and n is the conserved bubble number density per unit volume

$$n = \frac{3}{4\pi} \frac{\alpha_b}{\overline{\boldsymbol{R}^3}}. \tag{16}$$

2. Single-bubble dynamics

We model the single-bubble dynamics under the assumption that the bubbles remain a spherical, ideal, and spatially uniform gas region, which does not interact with other bubbles, break-up, or coalesce. Our model includes the thermal effects, viscous and acoustic damping, and phase change. The bubble radial accelerations $\ddot{R}$ are computed by the Keller–Miksis equation [44]:

$$R\ddot{R}\left(1 - \frac{\dot{R}}{c_b}\right) + \frac{3}{2}\dot{R}^2\left(1 - \frac{\dot{R}}{3c_b}\right) = \frac{p_{bw} - p_l}{\rho}\left(1 + \frac{\dot{R}}{c_b}\right) + \frac{R\dot{p}_{bw}}{\rho c_b}, \tag{17}$$

where c_b is the usual speed of sound associated with the bubble and

$$p_{bw} = p_b - \frac{4\mu\dot{R}}{R} - \frac{2\sigma}{R} \tag{18}$$

is the bubble wall pressure, for which p_b is the internal bubble pressure, σ is the surface tension coefficient, and μ is the liquid viscosity. The evolution of p_b is evaluated using the model of Ando [4]:

$$\dot{p}_b = \frac{3\gamma_b}{R}\left(\mathfrak{R}_v T_{bw} \dot{m}_v - \dot{R} p_b + \frac{\gamma_b - 1}{\gamma_b} \lambda_{bw} \left.\frac{\partial T}{\partial r}\right|_{r=w}\right), \tag{19}$$

where T is the temperature, λ is the thermal conductivity, $\mathfrak{R}_v$ is the gas constant and γ_b is the specific heat ratio of the gas. Mass transfer of the bubble contents follows the reduced model of Preston et al. [66] as

$$\dot{m}_v = \frac{\mathcal{D}\rho_{bw}}{1 - \chi_{vw}} \left.\frac{\partial \chi_v}{\partial r}\right|_w. \tag{20}$$

IV. SOLUTION METHOD

Our numerical scheme generally follows that of Coralic and Colonius [19]. The spatial discretization of (1) in three-dimensional Cartesian coordinates is

$$\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\boldsymbol{F}^x(\boldsymbol{q})}{\partial x} + \frac{\boldsymbol{F}^y(\boldsymbol{q})}{\partial y} + \frac{\boldsymbol{F}^z(\boldsymbol{q})}{\partial z} = \boldsymbol{s}(\boldsymbol{q}) - \boldsymbol{h}(\boldsymbol{q})\nabla\cdot\boldsymbol{u}, \tag{21}$$

where $\boldsymbol{F}^{x_i}$ are the $i \in (x, y, z)$-direction flux vectors and the treatment of $\nabla\cdot\boldsymbol{u}$ is discussed later.

A. Treatment of spatial derivatives

We use a finite volume method to treat the spatial derivatives. The finite volumes are

$$I_{i,j,k} = [x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{j+1/2}] \times [z_{k-1/2}, z_{k+1/2}]. \tag{22}$$

We spatially integrate (21) within each cell-centered finite volume as

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{q}_{i,j,k}}{\mathrm{d}t} =& \frac{1}{\Delta x_i}[\boldsymbol{F}^x_{i-1/2,j,k} - \boldsymbol{F}^x_{i+1/2,j,k}] + \\ & \frac{1}{\Delta y_j}[\boldsymbol{F}^y_{i,j-1/2,k} - \boldsymbol{F}^y_{i,j+1/2,k}] + \\ & \frac{1}{\Delta z_k}[\boldsymbol{F}^z_{i,j,k-1/2} - \boldsymbol{F}^z_{i,j,k+1/2}] + \boldsymbol{s}(\boldsymbol{q}_{i,j,k}) - \boldsymbol{h}(\boldsymbol{q}_{i,j,k})(\nabla\cdot\boldsymbol{u})_{i,j,k},\end{aligned} \tag{23}$$

where

$$\boldsymbol{q}_{i,j,k} = \frac{1}{V_{i,j,k}} = \iiint_{I_{i,j,k}} \boldsymbol{q}(x, y, z, t)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z, \tag{24}$$

$$\boldsymbol{s}_{i,j,k} = \frac{1}{V_{i,j,k}} = \iiint_{I_{i,j,k}} \boldsymbol{s}(x, y, z, t)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z, \tag{25}$$

$$\boldsymbol{F}_{i+1/2,j,k} = \frac{1}{\Delta y_j \Delta z_k} \iint_{A_{i+1/2,j,k}} \boldsymbol{F}(x, y, z, t)\,\mathrm{d}y\mathrm{d}z, \tag{26}$$

are cell-volume and face averages, for which $\Delta x_i = x_{i+1/2} - x_{i-1/2}$ are the mesh spacings (Δy and Δz have the same form), and $V_{i,j,k}$ and $A_{i+1/2,j,k}$ are the cell volumes and face areas.

The MFC can approximate this equation using high-order quadratures as opposed to simple cell-centered averages. This approach computes the flux surface integrals and source terms using a two-point, fourth-order, Gaussian quadrature rule, e.g.

$$\boldsymbol{F}_{i+1/2,j,k} = \frac{1}{4}\sum_{m=1}^{2}\sum_{l=1}^{2} \boldsymbol{F}(\boldsymbol{q}(x_{i+1/2}, y_{j_l}, z_{k_m})) \tag{27}$$

where l and m are the Gaussian quadrature point indices and

$$y_{j_l} = y_j + (2l+1)\frac{\Delta y_j}{2\sqrt{3}} \quad \text{and} \quad z_{k_m} = z_k + (2m+1)\frac{\Delta z_k}{2\sqrt{3}} \tag{28}$$

The divergence terms are treated using a midpoint rule

$$
(\nabla \cdot \boldsymbol{u})_{i,j,k} = \\
\frac{1}{\Delta x_i}(u_{i+1/2,j,k} - u_{i-1/2,j,k}) + \frac{1}{\Delta y_j}(v_{i,j+1/2,k} - v_{i,j-1/2,k}) + \frac{1}{\Delta z_k}(w_{i,j,k+1/2} - w_{i,j,k-1/2}), \tag{29}
$$

where $\boldsymbol{u} = \{u, v, w\}$ are the cell-averaged velocity components computed analogous to (27).

To avoid spurious oscillations at material interfaces, we ultimately evaluate the fluxes by reconstructing the primitive variables at the cell faces via a 5th-order-accurate WENO scheme [19] (though lower orders of accuracy are also included). This allows us to apply a Riemann solver, and so

$$
\boldsymbol{F}_{i+1/2,j,k} = \frac{1}{4}\sum_{m=1}^{2}\sum_{l=1}^{2} \widehat{\boldsymbol{F}}(\boldsymbol{q}^L_{i+1/2,j,k}, \boldsymbol{q}^R_{i+1/2,j,k}), \tag{30}
$$

where $\widehat{\boldsymbol{F}}$ is the numerical flux function of the Riemann solver. We use the HLLC approximate Riemann solver to compute the fluxes [81], though other Riemann solvers are also supported.

B. Time integration

Once the spatial derivatives have been approximated, (21) becomes a semi-discrete system of ordinary differential equations in time. We treat the temporal derivative using a Runge–Kutta time-marching scheme for the state variables. In order to achieve high-order accuracy and avoid spurious oscillations, we use the third-order-accurate total variation diminishing scheme of Gottlieb and Shu [28]:

$$
\begin{aligned}
\boldsymbol{q}^{(1)}_{i,j,k} &= \boldsymbol{q}^{n}_{i,j,k} + \Delta t \boldsymbol{L}(\boldsymbol{q}^{n}_{i,j,k}), \\
\boldsymbol{q}^{(2)}_{i,j,k} &= \frac{3}{4}\boldsymbol{q}^{n}_{i,j,k} + \frac{1}{4}\boldsymbol{q}^{(1)}_{i,j,k} + \frac{1}{4}\Delta t \boldsymbol{L}(\boldsymbol{q}^{(1)}_{i,j,k}), \\
\boldsymbol{q}^{n+1}_{i,j,k} &= \frac{1}{3}\boldsymbol{q}^{n}_{i,j,k} + \frac{2}{3}\boldsymbol{q}^{(2)}_{i,j,k} + \frac{2}{3}\Delta t \boldsymbol{L}(\boldsymbol{q}^{(2)}_{i,j,k}),
\end{aligned} \tag{31}
$$

where (1) and (2) are intermediate time-step stages, $\boldsymbol{L}$ represents the right hand side of (23), and n is the time-step index. We note that the MFC also supports Runge–Kutta schemes of orders 1–5.

C. Pressure-relaxation procedure

The pressure-disequilibrium model (9) requires a pressure-relaxation procedure to converge to an equilibrium pressure. We use the infinite-relaxation procedure of Saurel et al. [70]. At each time step it solves the non-relaxed, hyperbolic equations ($\mu \to 0$) using a first-order explicit time step integration, then relaxes the disequilibrium pressures for $\mu \to +\infty$. This procedure is performed at each Runge–Kutta stage, and so their is a unique pressure at the end of each stage and the 5- and 6-equation models reconstruct the same variables. As a result, simulations of the pressure-disequilibrium model are only modestly more expensive than the 5-equation models (about 5% for spherical bubble collapses [71]).

V. SIMULATION VERIFICATION AND VALIDATION

We next present several test cases that validate and verify the MFC's capabilities. These include one-, two-, and three-dimensional cases that span a wide variety of flow problems.

A. Shock–bubble interaction

We first consider a Mach 1.22 shockwave impinging a 5 cm diameter spherical helium bubble in air. This problem was investigated via experimental methods by Haas and Sturtevant [29] and has been previously used as a validation case for multicomponent flow simulations [23, 33, 77]. Our simulations are performed using an axisymmetric configuration; further simulation specifications can be found in Coralic and Colonius [19].

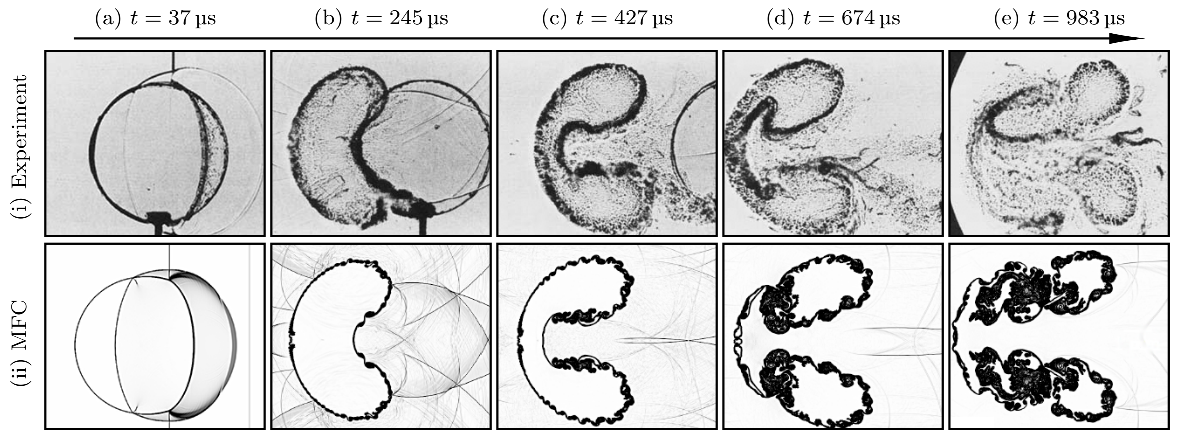

FIG. 1. Comparison between (i) experimental shadowgraphs of Haas and Sturtevant [29] and (ii) numerical Schlieren visualizations [68] using the MFC at select times (a)–(e) as labeled. Experimental images are ©Cambridge University Press 1987.

Visualizations of the shock impinging the bubble and subsequent breakup and vortex ring production are shown in figure 1. We see that the simulation results qualitatively match those of the experiment. Importantly, no spurious oscillations can be seen in the numerical schlieren images, despite their sensitivity to small density differences. We quantitatively compare our results to the experiment by considering velocity measurements of key flow features; Haas and Sturtevant [29] measured the velocity of the incident, reflected, and transmitted shocks, as well as the up and down-stream interfaces and jets. Our simulation results are within 10% of the experiments for all cases, and are generally within about 5% of the experimental means. Our results are also consistent with those computed independently via the level set [31] and diffuse-interface methods [74], including the Kelvin–Helmholtz instability that develops along the bubble interface (see figure 1 (b)–(e)).

B. Shock–droplet/cylinder interaction

We next consider air shocks interacting with liquid media in three dimensions. These problems are more computationally challenging, primarily due to the larger density ratio. The first case we analyze consists of a Mach 1.47 shock impinging a 4.8 mm diameter liquid water cylinder. The simulation parameterization can be found in Meng and Colonius [54].

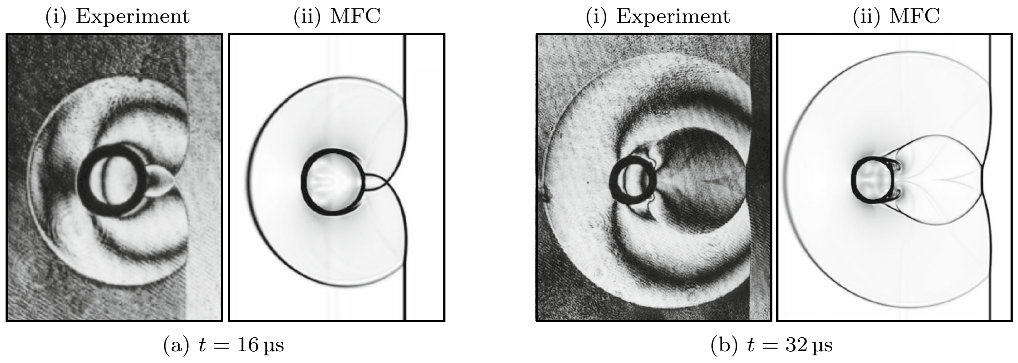

FIG. 2. Comparison between (i) holographic interferograms [35] and (ii) numerical Schlieren visualizations using the MFC at select times (a) and (b) as labeled. The experimental images are reprinted from Igra and Takayama [35].

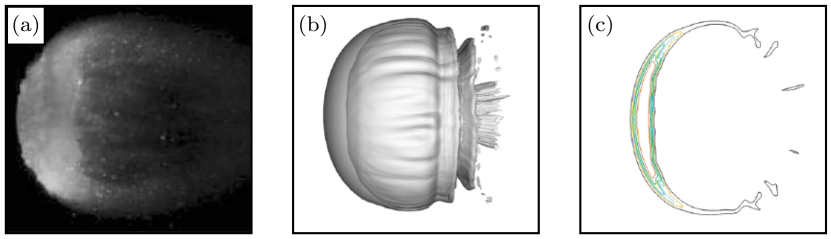

FIG. 3. Comparison of (a) experimental [78] and (b) numerical water droplets isosurface $\alpha_l = 0.01$. (c) is an isocontours of α_l ranging from 0.01 to 0.99.

Figure 2 shows a visualiation of experimental and our numerical results as the shock passes over the liquid cylinder. At early times it is difficult to assess the cylinder's deformation, so instead we compare the primary and secondary waves that are generated. In figure 2 (a) we see that the primary wave system, including the incident and reflected shock, have the same locations for both experimental and numerical results. The secondary wave system is generated when the Mach stems on both sides of the cylinder converge to rear stagnation point; in figure 2 (b) that these also match closely.

We also consider the breakup of a spherical water droplet due to a Mach 0.59 helium shock, following the experimental conditions of Theofanous et al. [78]. A full exposition of the simulation conditions can be found in Meng and Colonius [55]. Figure 3 shows their experimental image and volume fraction isosurfaces and sliced isocontours from our simulations. We use a small $\alpha_l = 0.01$ value for the isosurface of figure 3 (b) for comparison purposes since images from experiments are often obscured by the fine mist generated. While it is challenging to obtain accurate timing data from the experiments, a qualitative agreement between experiment and simulation are still observed for the shear-induced entrainment of the droplet.

C. Spherical bubble cavitation

Numerical simulation of cavitating spherical gas bubbles is challenging because mixture-region compressibility must be properly treated, discrete conservation must be enforced, and sphericity should be maintained in the presence of large density and pressure ratios. We consider a collapsing and rebounding air bubble in water at 10 times higher pressure as a test of the capabilities of the

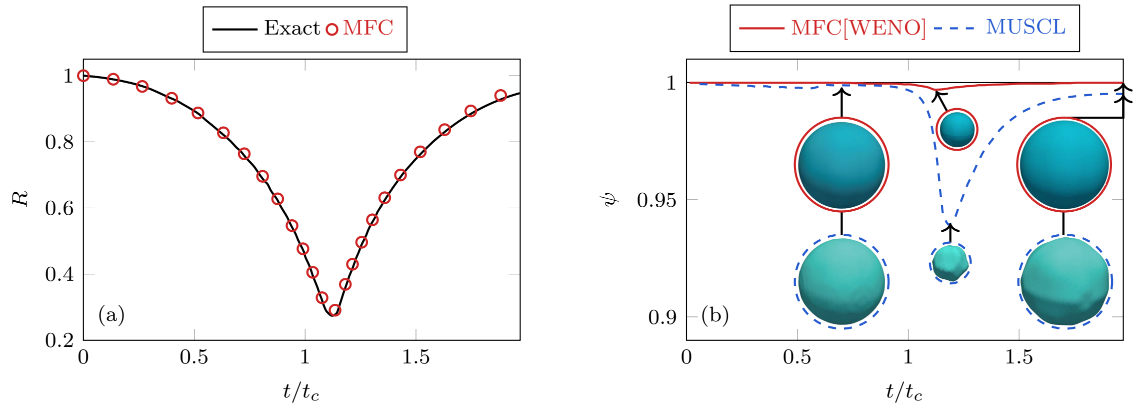

FIG. 4. Evolution of (a) the dimensionless bubble radius R and (b) its sphericity ψ. In (b) the nominal bubble shapes, represented by $\alpha_l = 0.5$ isosurfaces, are shown at select times.

MFC; specific simulation specifications were presented in Schmidmayer et al. [71].

Figure 4 (a) shows the evolution of the bubble radius; it reaches a minimum near the nominal Rayleigh collapse time t_c [9], then rebounds, as expected. The radius for our simulations is computed from the nominal $\alpha_l = 0.5$ volume-averaged bubble surface; this closely matches the solution expected following the Keller–Miksis equation [44]. We assess the quality of our simulation via computation of the bubble sphericity during the collapse-rebound process. Figure 4 (b) shows this sphericity ψ of the bubble, which is defined following Wadell [83] and a value of 1 indicates a spherical bubble We see that the WENO scheme used by the MFC is able to better maintain sphericity than a MUSCL scheme formulated for the same diffuse-interface model [71].

D. Isentropic and Taylor–Green vortices

We next consider two- and three-dimension vortex problems as a means to verify our solutions of the flow equations. The two-dimension problem we consider is the evolution of a steady, inviscid, isentropic, ideal-gas vortex. This problem has been used previously to assess the convergence properties of high-order WENO schemes for smooth solutions to the Euler equations [6, 79]. We use it here to verify that we obtain high-order accuracy away from shocks and material interfaces; details regarding the numerical parameters and exact problem formulation can be found in Coralic and Colonius [19].

Figure 5 (a) shows the density error for both low- and high-order finite volume cell-averaging (following section IV). Since the solution should be steady, the error is computed as the deviation from the initial condition after 1 dimensionless time unit as a function of the grid size, for which N is the number of finite volumes in one spatial direction. The convergence is 2nd- and 5th-order accurate for 2nd- and 4th-order-accurate cell-averaging schemes, respectively. Thus, we conclude that for multi-dimensional problems the 4th-order-accurate cell averaging we employ is required to achieve 5th-order accuracy associated with the WENO reconstructions.

We use the three-dimensional Taylor–Green vortex problem of Brachet et al. [8] to study the production of small length scales, including vortex stretching and dissipation. The simulation parameters again follow from Coralic and Colonius [19]. Figure 5 (b) shows the dimensionless dissipation rate of the kinetic energy ε in dimensionless time t, as computed over the entire computational domain. We see that the vortex stretching grows until $t \approx 5$, after which the effects of viscous dissipation begin to dominate. The results of the MFC closely match those of the direct solution computed by Brachet et al. [8] using spectral methods.

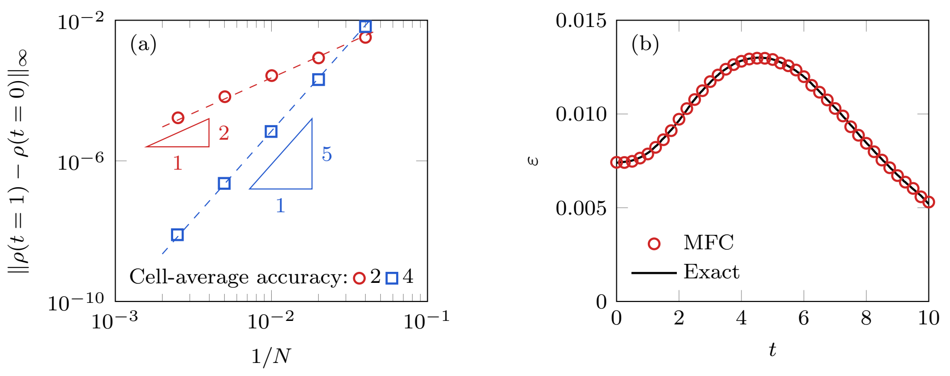

FIG. 5. (a) L_∞ density error associated with an isentropic steady vortex for different cell-averaged accuracies as labeled. (b) Non-dimensional kinetic energy dissipation rate associated with the three-dimensional Taylor–Green vortex problem; the MFC solution is compared to the direct spectral solution of Brachet et al. [8].

E. Further verification

The MFC has been verified using several other problems. Perhaps of most interest are the one-dimensional test cases used to clearly demonstrate the accuracy it is capable of. For example, of great importance are the development of spurious oscillations at material interfaces, which can significantly pollute simulation quality. To determine if such oscillations appear when using our method, Coralic and Colonius [19] considered the advection of an isolated air–water interface at constant velocity in a periodic domain. It was shown that when conservative variables are reconstructed, the interface is corrupted by spurious oscillations and the pressures can even become negative. Whereas when primitive variables are reconstructed, their character is oscillation free for both velocity and pressure down to round off error.

It is also challenging to predict the correct position and speed of waves that emit from shock–interface interactions. While the shock–bubble interaction problem considered in section V A showed that our method can approximate these quantities when comparing to experiments, it is helpful to consider a similar problem in one-dimension where exact solutions are available. Following Liu et al. [49], Coralic and Colonius [19] used the methods employed by the MFC to analyze a Mach 8.96 helium shockwave impinging an air interface. It was shown that the numerical results quantitatively matches the associated exact solution, correctly identifying the position and speed of all waves in the problem while avoiding any spurious oscillations. Of similar character is the gas–liquid shock tube problem of Cocchi et al. [15], which has been used as a model for underwater explosions. For this, Coralic and Colonius [19] also showed that, using our method, the numerical solution matches the exact one and correctly identifies the position and speed of all waves.

Finally, the ensemble-averaged bubbly flow model introduced in section III C was verified by simulating a weak acoustic pulse impinging a dilute bubble screen and comparing to the linearized bubble dynamic results of Commander and Prosperetti [18]. We saw that the measured phase speed and acoustic attenuations, computed via the method of Ando [4], match the expected results. Further, Bryngelson et al. [11] showed that this method quantitatively matches the volume-averaged formulation of the same problem.

VI. PARALLEL PERFORMANCE BENCHMARKS

It is important to ensure that our parallel implementation can utilize modern, large computer resources. To do this, we benchmark the MFC's parallel architecture via the usual scaling and speedup tests.

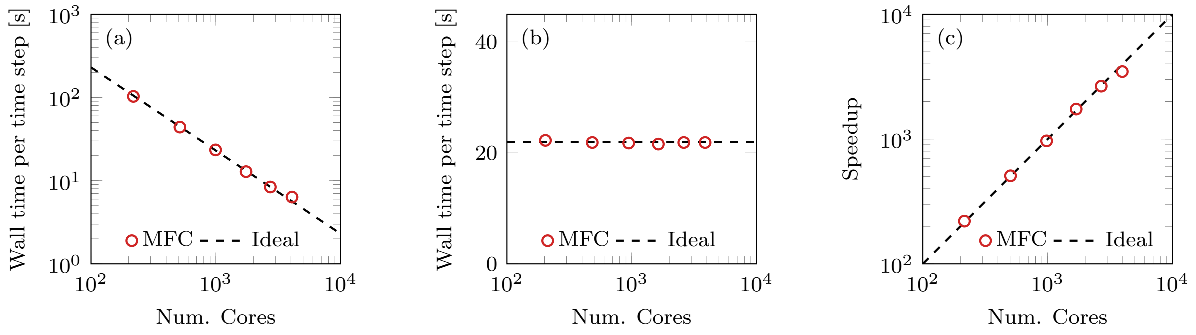

FIG. 6. MFC performance benchmarks: (a) strong scaling, (b) weak scaling, and (c) speedup tests.

Figure 6 shows parallel performance benchmarks of the MFC. The strong scaling test of figure 6 (a) measures how the solution time varies for a fixed problem size as the number of computing cores varies, the weak scaling test (figure 6 (b)) measures how well the computational load is balanced across the available cores by measuring solution time while fixing the grid size distributed to each core and varying the number of cores (thus changing the problem size overall), and the speedup test (figure 6 (c)) measures how the solution time increases with respect to serial computation as the number of cores varies. Thus, speedup is defined by the ratio of time cost of a parallel simulation with a certain of cores to a serial computation. The strong scaling and speedup tests are carried out on a 500^3 grid, while a constant load of 50^3 cells per core is maintained during the weak scaling test. For all tests the MFC performs very near the ideal threshold until the number of cores is 4096, at which point the results deviate slightly from ideal.

VII. CONCLUSIONS

We presented the MFC, a user-fieldly open-source tool capable of simulating multi-component, multi-phase, and multi-scale flows. It uses state-of-the-art diffuse-interface models, coupled with high-order interface capturing and Riemann solvers, to represent multi-material dynamics. The MFC includes a variety of flow models and numerical methods, including spatial and temporal orders of accuracy, that are useful when considering the computational requirements of challenging open problems. It also includes options for additional physics and modeling techniques, including a sub-grid ensemble-averaged bubbly flow model.

We also described the requirements to build the MFC and its design. This included external open-source software libraries that were readily available online. The MFC was divided into three main components that initialize and simulate the flow, then process the exported simulation data. Each of these components is modular, and thus can be readily modified by new developers. They are coupled together via an intuitive input Python script that automatically generates the required Fortran input files and executes the software component. The exported simulation files can be readily analyzed or treated via parallel post processing.

Finally, we presented a comprehensive set of both validation and verification studies of the MFC. Validation was performed via comparisons to shock-bubble, shock-droplet, and shock-water-cylinder experiments, while verification was obtained via numerical experiments involving isentropic and Taylor–Green vortices, as well as advected- and interface-interaction problems. A set of tests also showed that the MFC was able to perform near the ideal threshold of parallel computational efficiency.

ACKNOWLEDGEMENTS

The authors are grateful for the input of Dr. Benedikt Dorschner. This work was supported in part by multiple past grants from the US National Institute of Health (NIH), the US Office of Naval Research (ONR), and the US National Science Foundation (NSF), as well as current NIH grant number 2P01-DK043881 and ONR grant numbers N0014-17-1-2676 and N0014-18-1-2625. The computations presented here utilized the Extreme Science and Engineering Discovery Environment, which is supported under NSF grant number CTS120005. K.M. acknowledges support from the Funai Foundation for Information Technology via the Overseas Scholarship.

[1] Abgrall, R. (1996). How to prevent pressure oscillations in multicomponent flow calculations: a quasi conservative approach. *J. Comp. Phys.*, 125:150–160.

[2] Abgrall, R. and Karni, S. (2001). Computations of compressible multifluids. *J. Comp. Phys.*, 169:594–623.

[3] Allaire, G., Clerc, S., and Kokh, S. (2002). A five-equation model for the simulation of interfaces between compressible fluids. *J. Comp. Phys.*, 181:577–616.

[4] Ando, K. (2010). *Effects of polydispersity in bubbly flows.* PhD thesis, California Institute of Technology.

[5] Ball, G. J., Howell, B. P., Leighton, T. G., and Schofield, M. (2000). Shock-induced collapse of a cylindrical air cavity in water: a free-Lagrange simulation. *Shock waves*, 10:265–276.

[6] Balsara, D. S. and Shu, C. W. (2000). Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. *J. Comp. Phys.*, 160:405–452.

[7] Bauer, R. T. (2004). *Remarkable shrimps: adaptations and natural history of the carideans*, volume 7. University of Oklahoma Press.

[8] Brachet, M. E., Meiron, D. I., Orszag, S. A., Nickel, B. G., Morf, R. H., and Frisch, U. (1983). Small-scale structure of the Taylor–Green vortex. *J. Fluid Mech.*, 130:411–452.

[9] Brennen, C. E. (1995). *Cavitation and bubble dynamics.* Oxford University Press.

[10] Brennen, C. E. (2015). Cavitation in medicine. *Interface Focus*, 5(5).

[11] Bryngelson, S. H., Schmidmayer, K., and Colonius, T. (2019). A quantitative comparison of phase-averaged models for bubbly, cavitating flows. *Int. J. Mult. Flow*, 115:137–143.

[12] Chang, C.-H., Deng, X., and Theofanous, T. G. (2013). Direct numerical simulation of interfacial instabilities: a consistent, conservative, all-speed, sharp-interface method. *J. Comp. Phys.*, 242:946–990.

[13] Chauvin, A., Daniel, E., Chinnayya, A., Massoni, J., and Jourdan, G. (2015). Shock waves in sprays: numerical study of secondary atomization and experimental comparison. *Shock waves*, 26(4):403–415.

[14] Cocchi, J. P. and Saurel, R. (1997). A Riemann problem based method for the resolution of compressible multimaterial flows. *J. Comp. Phys.*, 137:265–298.

[15] Cocchi, J. P., Saurel, R., and Loraud, J. C. (1996). Treatment of interface problems with Godunov-type schemes. *Shock waves*, 5:347–357.

[16] Coleman, A. J., Saunders, J. E., Crum, L., and Dyson, M. (1987). Acoustic cavitation generated by an extracorporeal shockwave lithotripter. *Ultrasound Med. Biol.*, 13(2):69–76.

[17] Colonius, T., Hagmeijer, R., Ando, K., and Brennen, C. E. (2008). Statistical equilibrium of bubble oscillations in dilute bubbly flows. *Phys. Fluids*, 20(040902).

[18] Commander, K. W. and Prosperetti, A. (1989). Linear pressure waves in bubbly liquids: Comparison between theory and experiments. *J. Acoustic. Soc. Am.*, 85(732).

[19] Coralic, V. and Colonius, T. (2014). Finite-volume WENO scheme for viscous compressible multicomponent flow problems. *J. Comp. Phys.*, 219(2):715–732.

[20] Denner, F., Xiao, C.-N., and van Wachem, B. G. M. (2018). Pressure-based algorithm for compressible interfacial flows with acoustically-conservative interface discretisation. *J. Comp. Phys.*, 367:192–234.

[21] Engel, O. G. (1958). Fragmentation of waterdrops in the zone behind an air shock. *J. Res. Natl. Bur. Stand*, 60(3):245–280.

[22] Etter, P. C. (2013). *Underwater acoustic modeling and simulation.* CRC Press.

[23] Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S. (1999). A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comp. Phys.*, 152:457–492.

[24] Fuster, D. (2018). A review of models for bubble clusters in cavitating flows. *Flow, Turbulence and Combustion*, pages 1–40.

[25] Fuster, D. and Popinet, S. (2018). An all-Mach method for the simulation of bubble dynamics problems in the presence of surface tension. *J. Comp. Phys.*, 374:752–768.

[26] Glimm, J., Li, X. L., Liu, Y., Xu, Z. L., and Zhao, N. (2003). Conservative front tracking with improved accuracy. *SIAM J. Numer. Anal.*, 41:1926–1947.

[27] Gojani, A. B., Ohtani, K., Takayama, K., and Hosseini, S. H. R. (2009). Shock Hugoniot and equations of states of water, castor oil, and aqueous solutions of sodium chloride, sucrose and gelatin. *Shock waves*, 26:63–68.

[28] Gottlieb, S. and Shu, C.-W. (1998). Total variation diminishing Runge–Kutta schemes. *Math. Comput.*, 67(221):73–85.

[29] Haas, J. F. and Sturtevant, B. (1987). Interaction of weak shock waves with cylindrical and spherical gas inhomogeneities. *J. Fluid Mech.*, 181:41–76.

[30] Han, L. H., Hu, X. Y., and Adams, N. A. (2014). Adaptive multi-resolution method for compressible multi-phase flows with sharp interface model and pyramid data structure. *J. Comp. Phys.*, 262:131–152.

[31] Hejazialhosseini, B., Rossinelli, D., Bergdorf, M., and Koumoutsakos, P. (2010). High order finite volume methods on wavelet-adapted grids with local time-stepping on multicore architectures for the simulation of shock-bubble interactions. *J. Comp. Phys.*, 229:8364–8383.

[32] Henrick, A. K. and an d J. M. Powers, T. D. A. (2005). Mapped weighted essentially non-oscillatory schemes: achieving optimal order near critical points. *J. Comp. Phys.*, 207:542–567.

[33] Hu, X. Y. and Khoo, B. C. (2004). An interface interaction method for compressible multifluids. *J. Comp. Phys.*, 198(35–64).

[34] Hu, X. Y., Khoo, B. C., Adams, N. A., and Huang, F. L. (2006). A conservative interface method for compressible flows. *J. Comp. Phys.*, 219(2):553–578.

[35] Igra, D. and Takayama, K. (2001). Numerical simulation of shock wave interaction with a water column. *Shock waves*, 11:219–228.

[36] Ikeda, T., Yoshizawa, S., Masakata, T., Allen, J. S., Takagi, S., Ohta, N., Kitamura, T., and Matsumoto, Y. (2006). Cloud cavitation control for lithotripsy using high intensity focused ultrasound. *Ultrasound Med. Biol.*, 32(9):1383–1397.

[37] Jiang, G. S. and Schu, C. W. (1996). Efficient implementation of weighted ENO schemes. *J. Comp. Phys.*, 126:202–228.

[38] Johnsen, E. and Colonius, T. (2006). Implementation of weno schemes in compressible multicomponent flow problems. *J. Comp. Phys.*, 219:715–732.

[39] Johnsen, E. and Ham, F. (2012). Preventing numerical errors generated by interface-capturing schemes in compressible multi-material flows. *J. Comp. Phys.*, 231:5705–5717.

[40] Joseph, D. D., Belanger, J., and Beavers, G. (1999). Breakup of a liquid drop suddenly exposed to a high-speed airstream. *Int. J. Mult. Flow*, 25(6):1263–1303.

[41] Kannan, K., Kedelty, D., and Herrmann, M. (2018). An in-cell reconstruction finite volume method for flows of compressible immiscible fluids. *J. Comp. Phys.*, 373:784–810.

[42] Kapila, A., Menikoff, R., Bdzil, J., Son, S., and Stewart, D. (2001). Two-phase modeling of DDT in granular materials: Reduced equations. *Phys. Fluids*, 13:3002–3024.

[43] Kedrinskii, V. (1976). Negative pressure profile in cavitation zone at underwater explosion near free surface. *Acta Astro.*, 3(7-8):623–632.

[44] Keller, J. B. and Miksis, M. (1980). Bubble oscillations of large amplitude. *J. Acoustic. Soc. Am.*, 68(628).

[45] Khosla, S., Smith, C. E., and Throckmorton, R. P. (2006). Detailed understanding of drop atomization by gas crossflow using the volume of fluid method. In *19th Annual Conference on Liquid Atomization and Spray Systems.* ILASS.

[46] Koukouvinis, P., Bruecker, C., and Gavaises, M. (2017). Unveiling the physical mechanism behind pistol shrimp cavitation. *Sci. Rep.*, 7(1):13994.

[47] Laksari, K., Assari, S., Seibold, B., Sadeghipour, K., and Darvish, K. (2015). Computational simulation of the mechanical response of brain tissue under blast loading. *Biomech. Model. Mechanobiol.*, 14(3):459–472.

[48] Le Métayer, O., Massoni, J., and Saurel, R. (2004). Elaborating equations of state of a liquid and its vapor for two-phase flow models. *Int. J. Therm. Sci.*, 43:265–276.

[49] Liu, T. G., Khoo, B. C., and Yeo, K. S. (2003). Ghost fluid method for strong shock impacting on material interface. *J. Comp. Phys.*, 190:651–681.

[50] Liu, W., Yuan, L., and Shu, C. (2011). A conservative modification to the ghost fluid method for compressible multiphase flows. *Commun. Comput. Phys.*, 10:785–806.

[51] Marsh, S. P. (1980). *LASL Shock Hugoniot Data, Los Alamos Series on Dynamic Material Properties.* University of California Press, Berkeley.

[52] Massoni, J., Saurel, R., Nkonga, B., and Abgrall, R. (2002). Some models and Eulerian methods for interface problems between compressible fluids with heat transfer. *Int. J. Heat Mass Transfer*, 45:1287–1307.

[53] Meng, J. C. (2016). *Numerical simulations of droplet aerobreakup.* PhD thesis, California Institute of Technology.

[54] Meng, J. C. and Colonius, T. (2015). Numerical simulations of the early stages of high-speed droplet breakup. *Shock waves*, 25(4):399–414.

[55] Meng, J. C. and Colonius, T. (2018). Numerical simulation of the aerobreakup of a water droplet. *J. Fluid Mech.*, 835:1108–1135.

[56] Menikoff, R. and Plohr, B. J. (1989). The Riemann problem for fluid-flow of real materials. *Rev. Mod. Phys.*, 61(1):75–130.

[57] Mirjalili, S., Ivey, C. B., and Mani, A. (2018). Comparison between the diffuse interface and volume of fluid methods for simulating two-phase flows. *arXiv preprint arXiv:1803.07245.*

[58] Pan, S., Han, L., Hu, X., and Adams, N. A. (2018). A conservative interface-interaction method for compressible multi-material flows. *J. Comp. Phys.*, 371:870–895.

[59] Patek, S. and Caldwell, R. (2005). Extreme impact and cavitation forces of a biological hammer: strike forces of the peacock mantis shrimp odontodactylus scyllarus. *J. Exp. Biol.*, 208(19):3655–3664.

[60] Patek, S. N., Korff, W. L., and Caldwell, R. (2004). Biomechanics: deadly strike mechanism of a mantis shrimp. *Nature*, 428(6985):819.

[61] Pelanti, M. and Shyue, K.-M. (2014). A mixture-energy-consistent six-equation two-phase numerical model for fluids with interfaces, cavitation and evaporation waves. *J. Comp. Phys.*, 259(331–357).

[62] Perigaud, G. and Saurel, R. (2005). A compressible flow model with capillary effects. *J. Comp. Phys.*, 209:139–178.

[63] Petitpas, F., Massoni, J., Saurel, R., Lapebie, E., and Munier, L. (2009). Diffuse interface models for high speed cavitating underwater systems. *Int. J. Mult. Flow*, 35(8):747–759.

[64] Pickard, W. (1981). The ascent of sap in plants. *Prog. Biophy. Molec. Biol.*, 37:181–229.

[65] Pishchalnikov, Y. A., Sapozhnikov, O. A., Bailey, M. R., Williams, J. C., Cleveland, R. O., Colonius, T., Crum, L. A., Evan, A. P., and McAteer, J. A. (2003). Cavitation bubble cluster activity in the breakage of kidney stones by lithotripter shockwaves. *J. Endourol.*, 17(7):435–446.

[66] Preston, A., Colonius, T., and Brennen, C. E. (2007). A reduced-order model of diffusion effects on the dynamics of bubbles. *Phys. Fluids*, 19(123302).

[67] Proud, W. G., Nguyen, T.-T. N., Bo, C., Butler, B. J., Boddy, R. L., Williams, A., Masouros, S., and Brown, K. (2015). The high-strain rate loading of structural biological materials. *Metall. Mater. Trans. A*, 46(10):4559–4566.

[68] Quirk, J. J. and Karni, S. (1996). On the dynamics of a shock-bubble interaction. *J. Fluid Mech.*, 318:129–163.

[69] Saurel, R., Petitpas, F., and Abgrall, R. (2008). Modelling phase transition in metastable liquids: Application to cavitating and flashing flows. *J. Fluid Mech.*, 607:313–350.

[70] Saurel, R., Petitpas, F., and Berry, R. A. (2009). Simple and efficient relaxation methods for interfaces separating compressible fluids, cavitating flows and shocks in multiphase mixtures. *J. Comp. Phys.*, 228(5):1678–1712.

[71] Schmidmayer, K., Bryngelson, S. H., and Colonius, T. An assessment of multicomponent flow models and interface capturing schemes for spherical bubble dynamics. *under review.*

[72] Schmidmayer, K., Petitpas, F., Daniel, E., Favrie, N., and Gavrilyuk, S. L. (2017). A model and numerical method for compressible flows with capillary effects. *J. Comp. Phys.*, 334:468–497.

[73] Shyue, K. M. (1999). A fluid-mixture type algorithm for compressible multicomponent flow with van der waals equation of state. *J. Comp. Phys.*, 456:43–88.

[74] So, K. K., Hu, X. Y., and Adams, N. A. (2012). Anti-diffusion interface sharpening technique for two-phase compressible flow simulations. *J. Comp. Phys.*, 231(11):4304–4323.

[75] Streeter, V. (1983). Transient cavitating pipe flow. *J. Hydraulic Eng.*, 109(11):1407–1423.

[76] Tagawa, Y., Oudalov, N., Ghalbzouri, A. E., Sun, C., and Lohse, D. (2013). Needle-free injection into skin and soft matter with highly focused microjets. *Lab Chip*, 13(7):1357–1363.

[77] Terashima, H. and Tryggvason, G. (2009). A front-tracking/ghost-fluid method for fluid interfaces in compressible flows. *J. Comp. Phys.*, 228:4012–4037.

[78] Theofanous, T. G., Mitkin, V. V., Ng, C. L., Chang, C.-H., Deng, X., and Sushchikh, S. (2012). The physics of aerobreakup. part II. Viscous liquids. *Phys. Fluids*, 24:022104.

[79] Titarev, V. A. and Toro, E. F. (2004). Finite-volume WENO schemes for three-dimensional conservation laws. *J. Comp. Phys.*, 201:238–260.

[80] Toro, E. (2009). *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction.* Springer, Dordrecht, New York.

[81] Toro, E., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the HLL-Riemann solver. *Shock waves*, 4(1):25–34.

[82] Turangan, C. K., Ball, G. J., Jamaluddin, A. R., and Leighton, T. G. (2017). Numerical studies of cavitation erosion on an elastic–plastic material caused by shock-induced bubble collapse. *Proc. Royal Soc. A*, 473(2205):20170315.

[83] Wadell, H. (1935). Volume, shape, and roundness of quartz particles. *J. Geol.*, 43(3):250–280.

[84] Wallis, G. B. (1969). *One-dimensional two-phase flow.* McGraw-Hill.

[85] Weyler, M., Streeter, V., and Larsen, P. (1971). An investigation of the effect of cavitation bubbles on the momentum loss in transient pipe flow. *J. Basic Eng.*, 93(1):1–7.

[86] Wood, A. B. (1930). A textbook of sound. *G. Bell and Sons LTD, London.*

[87] Zhang, D. Z. and Prosperetti, A. (1994). Ensemble phase-averaged equations for bubbly flows. *Phys. Fluids*, 6(2956).