Computer Physics Communications 251 (2020) 107093

Contents lists available at ScienceDirect

Computer Physics Communications

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

ECOGEN: An open-source tool for multiphase, compressible, multiphysics flows☆,☆☆

Kevin Schmidmayer [b], Fabien Petitpas [a,*], Sébastien Le Martelot [c], Éric Daniel [a]

[a] Aix Marseille Univ, CNRS, IUSTI, Marseille, France
[b] California Institute of Technology, Division of Engineering and Applied Science, Pasadena, CA, USA
[c] Centre National d'Études Spatiales, Paris, France

ARTICLE INFO

Article history:
Received 29 January 2019
Received in revised form 2 October 2019
Accepted 15 November 2019
Available online 6 December 2019

Keywords:
CFD code
Multiphase flows
Compressible
Multiphysics
Unstructured mesh
Adaptive mesh refinement

ABSTRACT

ECOGEN, a new open-source computational fluid dynamics code is presented. It is a multi-model tool devoted to the simulation of compressible flows. A large range of problems can be solved, from single-phase gas dynamics to multiphase, multiphysics flows including interface problems between pure fluids. This code is suited for strongly unsteady flows. The numerical solver of ECOGEN is implemented in a flexible structure making the code able to compute such complex flows on different kinds of discretization grids. The implemented hyperbolic solver is able to deal with Cartesian geometries as well as unstructured grids. A recent adaptive mesh refinement method is also implemented. Its numerical implementation is presented in detail to help the enthusiastic developer to contribute to this open-source project. Representative test cases are presented to show the tool abilities and to open the gate for future developments.

Program summary

Program title: ECOGEN
Program files doi: http://dx.doi.org/10.17632/5bvx4g39dw.1
Licensing provisions: GNU General Public License 3
Programming language: C++ and XML
Supplementary material: MPI Library required
Nature of problem: The code solves sets of partial differential equations of compressible, multiphase flows in the framework of diffuse-interface methods. It is dedicated to unsteady flows involving acoustic waves, shock waves and material interfaces between liquids and gases. Phase change problems (heating or cavitating flows) can be treated with respect to physical conservation principles and thermodynamics consistency.
Solution method: The numerical method is based on finite volume discretization involving approximate Riemann solvers on different multi-dimensional grids: Cartesian (with or without AMR algorithms) or unstructured. Time and space integration scheme is based on first and second-order methods using the MUSCL approach. The time integration is explicit, the time step obeys a CFL condition. The algorithm is using Message Passing Interface library for the treatment of communications in parallel simulations. Geometrical domain decomposition is automatically generated for Cartesian grids.
Additional comments:
Official web site: https://code-mphi.github.io/ECOGEN/
Official documentation: https://code-mphi.github.io/ECOGEN/docs/sphinx_docs/index.html

1. Introduction

Multiphase, compressible, multiphysics flows are present in numerous industrial and medical applications involving physical phenomena such as:

- Bubble dynamics and cavitation for the treatment of kidney stones in the context of shock-wave and burst-wave lithotripsy [1–10], for the improvement of the artificial

☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).
☆☆ The review of this paper was arranged by Prof. Hazel Andrew.
* Corresponding author.
E-mail addresses: kevinsch@caltech.edu, kevin.schmidmayer@gmail.com (K. Schmidmayer), fabien.petitpas@univ-amu.fr (F. Petitpas), sebastien.lemartelot@cnes.fr (S. Le Martelot), eric.daniel@univ-amu.fr (É. Daniel).

https://doi.org/10.1016/j.cpc.2019.107093
0010-4655/

heart valves and pumps [11], for the understanding of injury mechanisms during blast trauma [12–14] or of syringe failure by an autoinjector device during drug delivery [15,16], for liquid flows around hyper-velocity projectiles and submarine airfoils, or inside nozzles such as fuel injector systems [17–22].
- Breakup of liquid droplets induced by high-speed flows or shock waves for combustion systems where a liquid jet is atomized [23–28], for the erosion of aircraft surface during supersonic flights [29–32] or for nuclear security issue involving shock-wave attenuation [33,34].
- Formation and fragmentation of liquid jets [35,36] as for example for needle-free injections into skin with highly focused microjets [37,38].
- Impact of solids at high velocity for fracture and fragmentation in ductile materials undergoing impact loading, for hypervelocity impact on satellites or for blast effects on structure [39–44].
- Boiling flows [45–47].
- Deflagration-to-detonation transition in granular energetic materials [48] or detonation waves in condensed energetic materials [49].

Among the previous references, many have participated to the improvement of diffuse-interface models and schemes with the objective to understand, to precisely describe and to reliably predict the behavior of these complicated flows. One can also note additional contributions to the associated numerical methods from authors such as Toro [50], Dumbser et al. [51] and Shyue and Xiao [52] to name a few.

However, only home-made computational research tools have been developed in parallel by researchers in different teams. It appears that these home-made tools are developed for similar applications and involve similar methods. Their specificities often rely on the use of a given discretized grid or a given mathematical model. This is the reason why ECOGEN has been developed: It utilizes the modular aspect of the C++ language to treat different kinds of problems with different appropriate physics in a unique code by sharing common features issued from academic works (grids, equations of state, numerical solver). This sharing of complex features makes developers of the code able to implement quite simply a new flow model without paying any attention to what is already existing in the tool. Moreover, a peculiar attention is paid to conserve this virtue that makes it able to deal with different problems without compromising the performance.

If ECOGEN is one of the first open-source package for compressible, multiphase flow solutions in the framework of diffuse-interface methods, a little alternative exists with comparable features. OpenFOAM [53] can integrate interface-capturing methods but through external projects (not natively). Caltech develops an open-source package named MFC[1] [54] based on a WENO scheme for multiphase flow with similar approaches. In Schmidmayer et al. [55], one can find comparisons between ECOGEN's MUSCL and MFC's WENO schemes for treating spherical-bubble-dynamics problems. A trend is observed where the higher-order-accurate WENO scheme shows slightly better results than the lower-order-accurate MUSCL scheme for relatively slow interface dynamics while the opposite is observed for relatively fast interface dynamics. Note that the latter correspond to our previously cited, problem interests. Few of the main results of this comparison and single-node performance comparison of both codes are given herein. Furthermore, ECOGEN offers an Adaptive Mesh Refinement (AMR) technique [26] specifically made to treat fast dynamics problems in a very efficient manner and an unstructured mesh option.

ECOGEN means:

1. **E**volutive: Because of its modular aspect, it is quite obvious that the tool is dedicated to evolve according to the new models resulting from the last research works on multiphase flows.
2. **C**ompressible: The tool is devoted to compressible applications. The numerical solver is based on a hyperbolic solver using Godunov method and Riemann problems for intercell flux calculations [56].
3. **O**pen-source: It is free and distributed under the GNU General Public License (http://www.gnu.org/licenses/gpl.txt).
4. **G**enuine: The aim is to provide a truthful tool able to help the community for a better understanding on multiphase, compressible, multiphysics flows.
5. **E**asy: The code structure is designed for easier implementations of new models.
6. **N**-phase: If numerical tools already exist for solving single-phase flow problems, ECOGEN is more specifically designed for multiphase flow problems.

This paper presents in detail the version 1.0 of ECOGEN. Section 2 is an overview of the code, describing downloaded package, installation instructions, main CFD features and a brief description of Input/Output (I/O) files. Section 3 describes the code structure and the base ideas making ECOGEN modular. In Section 4, grid structures embedded in the code are detailed. Section 5 is devoted to the implemented mathematical models showing how the code is able to treat a variety of equations of state, flow models and their physical extensions. The numerical solver is also presented in this section. Section 6 presents validations against analytic solutions as well as experimental ones. A comparison of results obtained with MFC [54] is also presented. In Section 7, typical results obtained using ECOGEN in different flow situations illustrate the code possibilities. Finally, Section 8 provides elements in regard to ECOGEN's computational performance: Parallel scaling and single-node performance.

2. Overview

2.1. Package

The ECOGEN_V1.0 package is freely available at the following url: https://code-mphi.github.io/ECOGEN/. This package includes several folders and files organized and described below:

Folders:

- ECOGEN/src/folder including C++ source files.
- ECOGEN/libMeshes/folder including examples of unstructured meshes in *.geo format (gmsh[2] files [57]).
- ECOGEN/libEOS/folder including some possible parameters for different equations of state in XML files.
- ECOGEN/libTests folder including:
  - ECOGEN/libTests/referenceTestCases/folder organized as a test-case library according to the flow model (Euler-equation ECOGEN solver, Kapila's model for multiphase-flow ECOGEN solver, homogeneous-Euler-equation ECOGEN solver, etc.).
  - Four quick-manual XML files to help the user to create a new flow calculation with ECOGEN.

Files:

[1] MFC is a multi-component flow code made in the computational flow physics group at Caltech (California Institute of Technology).

[2] Gmsh is a simple and efficient free mesh generator written by Christophe Geuzaine and Jean-François Remacle (see: http://gmsh.info/).

- ECOGEN.xml: Main entry file to select running cases.
- Makefile: For compilation in Unix environment. This file may require some adaptations to the user's environment.
- LICENSE, COPYRIGHT and AUTHORS: Information files about authors and licensing.
- README.md: Information file.
- ECOGEN_documentation.pdf: User's guide including installation instructions, full description of input and output files, test case descriptions, etc.

2.2. Installation and testing

ECOGEN must be compiled with a C++ compiler. It also requires a functional system implementation of Message Passing Interface (MPI) library (not provided in the package). Installation of these prerequisites can be done using one of these three possibilities:

- Online installation guide available at: https://code-mphi.github.io/ECOGEN/start/
- In the official documentation online at: https://code-mphi.github.io/ECOGEN/docs/sphinx_docs/index.html section *installation instructions*.
- In the *ECOGEN_documentation.pdf* provided in the downloaded package.

2.3. Features

ECOGEN is including the following features for solving compressible-flow problems:

1. Cartesian meshes with or without AMR method [26], or unstructured meshes;
2. parallel computing using a domain decomposition (MPI library);
3. first- and second-order-accurate finite volume method based on MUSCL approach [50] and THINC interface-sharpening method for multiphase flow [52] on Cartesian grids (with or without AMR);
4. single-phase or multiphase flows using diffuse-interface method [58]. Depending on multiphase application, various interphase equilibria may be selected or not (velocities, pressures, temperatures, chemical potentials).
5. Depending on the flow model, additional physical effects are implemented:
    - Surface tension [27];
    - viscosity [59,60];
    - heat conduction [59];
    - phase transition [19,47,49];
    - volume forces (gravity).

This list of features represents a snapshot of ECOGEN features at the date of publication and would evolve in time.

2.4. Brief description of I/O files

ECOGEN settings are mainly managed via input XML files and GEO files for the unstructured meshes (gmsh format). Running the code creates output files including the computational results. The global input- and output-file structure is depicted in Fig. 1. ECOGEN XML-format files are located in the different folders and subfolders:

- ECOGEN.xml: Main input file driving the name of simulations to run.
- Input files for the material equation-of-state parameters are placed in the ECOGEN/libEOS/folder.
- mainV5.xml (input file specific to a given simulation): It includes the settings to control the name, the time, the output mode and the precision of the numerical scheme.
- meshV5.xml (input file specific to a given simulation): It contains parameters relative to the geometry (kind of mesh, dimensions, AMR, etc.).
- modelV4.xml (input file specific to a given simulation): It allows the user to set the physics of the simulation: Flow model, number of phases, equations of state, additional physics, source terms and relaxations.
- initialConditionsV4.xml (input file specific to a given simulation): All necessary settings for the initialization of the computational domain are set in this input file as well as specification of boundary conditions.

The GEO input files for the unstructured meshes are located in ECOGEN/libMeshes/folder.

Output files can be generated under either ASCII column files or XML, standard Visualization Tool Kit (VTK), ASCII (or binary) format. Output files under VTK file format can be easily read by common open-source visualization tool (*e.g.* Paraview [61] or VisIt [62]).

For a complete description of the use of input and output files, please refer to the official online documentation. The user can also look at the input-file possibilities within the four quick-manual files present in the ECOGEN/libTests/folder.

3. ECOGEN C++ structure

ECOGEN is developed in a way to make use of the modular aspect of the C++ language and especially inheritance and polymorphism through classes, their attributes and methods. Thanks to that, ECOGEN clearly distinguishes the input, the geometry, the flow model and the thermodynamics of fluids. In other words, this is a very interesting feature for the developers because it means, for example, that everything that belongs to the geometry (dimensions, meshes, etc.) can be modified without any care of the physics (flow models, additional physics, etc.) and reversely. It is also quite easy to add new equations of state as they are described in specific classes in the code. The global simplified structure idea of the code is summarized in Fig. 2.

To help the reader in the following, base classes are written in bold characters and derived classes are written in italic characters.

As ECOGEN is able to distinguish the geometry and the physics, links are ensured by specific computational-cell classes: **Cell** and **CellInterface**. Computational cells are objects from the base class **Cell**. Each cell contains the following attributes:

- Information for the number of equations: Number of phases, of transport equations and of additional physics;
- objects for phases, mixture, transports and quantities of additional physics;
- a flux object;
- AMR attributes (if activated);
- a pointer to its corresponding geometrical element;
- one pointer for each interface with other cells (*e.g.* 2, 4 and 8 cell interfaces per cell in Cartesian one, two and three dimensions, respectively).

A cell possesses pointers to its cell interfaces and not to the geometrical faces. Each cell interface will be associated to an object from base class **CellInterface** that contains the following attributes:

- Pointer to the used model;
- pointer to its corresponding geometrical face;
- pointers to its left and right cells;
- AMR attributes (if activated).

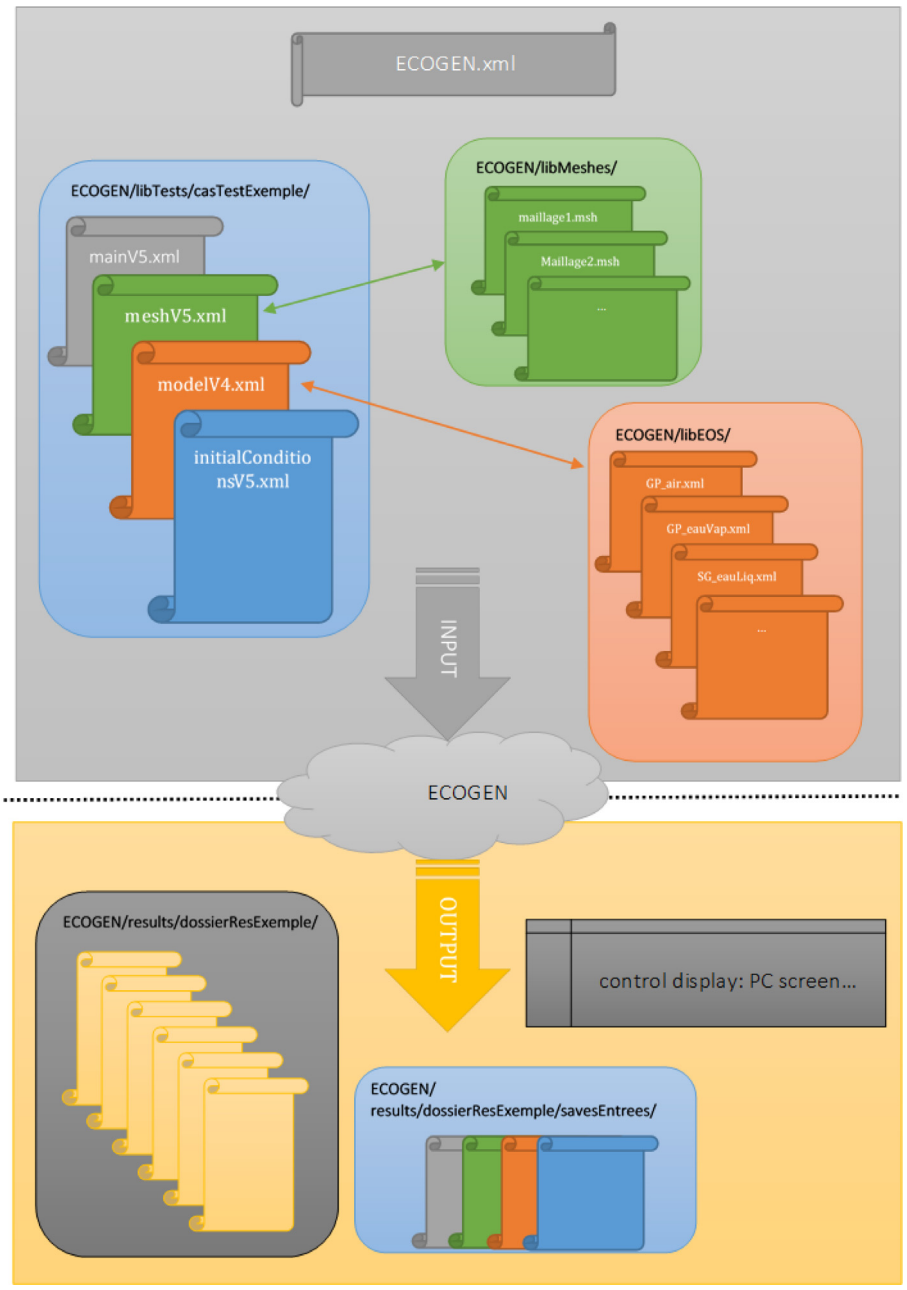

Fig. 1. Input- and output-file hierarchy.

Fig. 3 summarized the main architecture between the geometry and the physics.

Some specificities have to be added to these cells and cell interfaces if the second-order scheme is used. In that case, they are instantiated to derived classes *CellO2* and *CellInterfaceO2* that contain additional attributes. For the second-order cells, there are:

- Additional phases, mixture and transports for the second-order numerical scheme;
- additional conservative fluxes for the second-order numerical scheme.

And for the second-order cell interfaces:

- Slopes for the phases, mixture and transports.

Computational-domain boundaries are treated as derived class *BoundCond* from **CellInterface** base class. Their added attributes are only characteristics to compute the corresponding boundary condition involved.

In `ECOGEN`, a simulation is ensured by creation of a general object from base class **Run** that acts as a conductor between all `ECOGEN` classes and corresponding instanciated objects. This class possesses several important attributes:

- General information for the simulation (numbers of phases, source terms, precision order, simulation-time control, etc.);
- a **Model** object containing model information;
- a **Mesh** object containing geometrical information;
- an array of **EOS** objects containing fluid equation-of-state parameters;
- an array of **Cell** objects containing fluid-state variables;
- an array of **CellInterface** objects containing information on cell interfaces.
- arrays of **AddPhys**, **Source** and **Relaxation** objects.

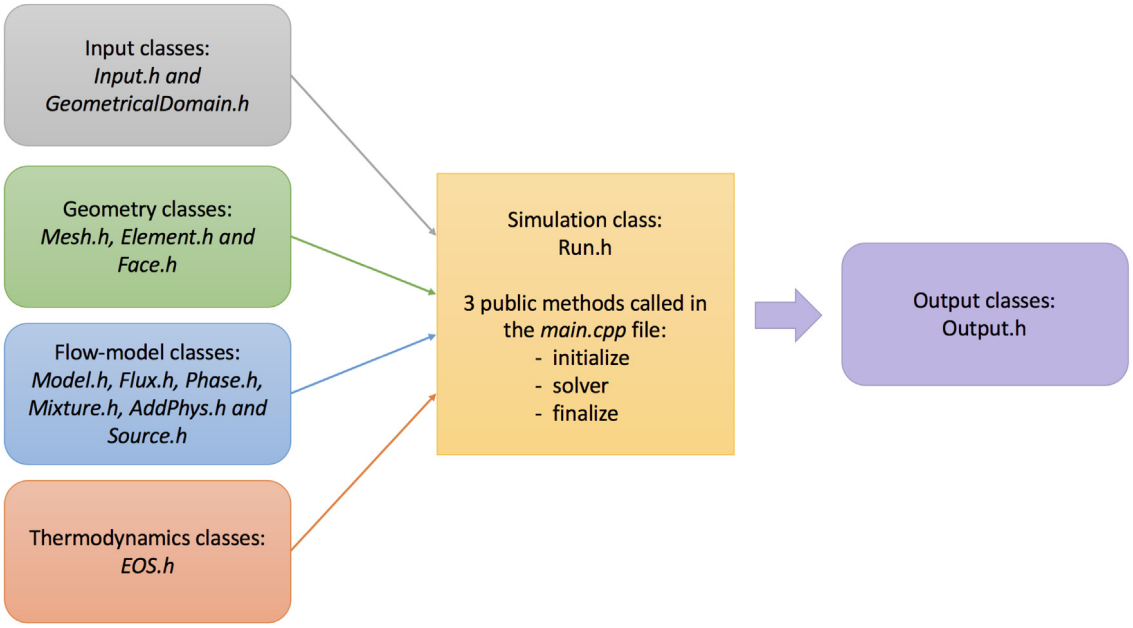

Fig. 2. The main classes of ECOGEN. Input, geometry, flow-model and thermodynamics are separated to simplify the code evolution.

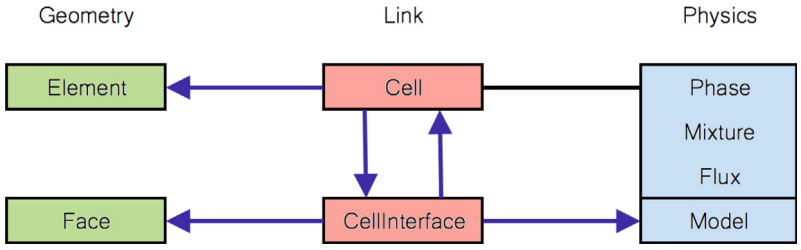

Fig. 3. Cell and **CellInterface** objects are linking the geometry and the physics in ECOGEN. Arrows correspond to the pointers between each class object. Black line means that a **Cell** object contains objects from derived classes of **Phase**, **Mixture** and **Flux**.

In other word, the **Run** class object is regrouping all information about a simulation in ECOGEN. It also contains the three main methods of ECOGEN to run a simulation. These methods are called in the *main.cpp* source file for each requested simulation:

- The Run::initialize method is in charge of setting initial conditions for the simulation.
- The Run::solver method contains the numerical scheme.
- The Run::finalize method is in charge of freeing up memory for the next simulation.

Fig. 4 shows a simplified global view of the ECOGEN structure. Boxes represent base classes and dashed arrows show the links between them with the appropriate variables involved. Find more information in the ECOGEN documentation ("doc" folder) made with Doxygen.[3]

Derived classes corresponding to geometry and physics are briefly detailed below.

3.1. Geometry classes

Geometry definition is ensured by classes derived from three base classes: **Mesh**, **Element** and **Face**.

A mesh object is derived from base class **Mesh**. It can either be Cartesian (*MeshCartesian*) or unstructured (*MeshUnStruct*), see Fig. 5.

Concerning the AMR method, only the Cartesian type of mesh is adapted yet and the way it is implemented in ECOGEN follows the lines of Schmidmayer et al. [26]. Notice that there is no specific Cartesian- or unstructured-parallel derived classes since the parallel part is generalized into the mesh classes. These mesh classes define how to build a mesh object which is composed of geometrical elements derived from base class **Element** and its corresponding geometrical faces derived from base class **Face** (Fig. 6). Element main attributes are the position (3 coordinates), the volume and a characteristic length for CFL condition. Face main attributes are the position, the surface, the normal, the tangent and the binormal. These last tree attributes represent a local base of vectors attached to the face.

3.2. Physical classes

To entirely describe the physics of a flow, three ingredients are needed: The base mathematical flow model, the equations of state describing the thermodynamical behavior of each fluid and the additional physics involved (*e.g.* surface tension, viscous effects, heat conduction).

3.2.1. Flow-model classes

Physics of the flow model are defined using four kinds of inter-linked base classes: **Model**, **Phase**, **Mixture** and **Flux**. The flow-model object is determined by the corresponding class derived from base class **Model** (see Fig. 7).

[3] see http://www.stack.nl/~dimitri/doxygen/.

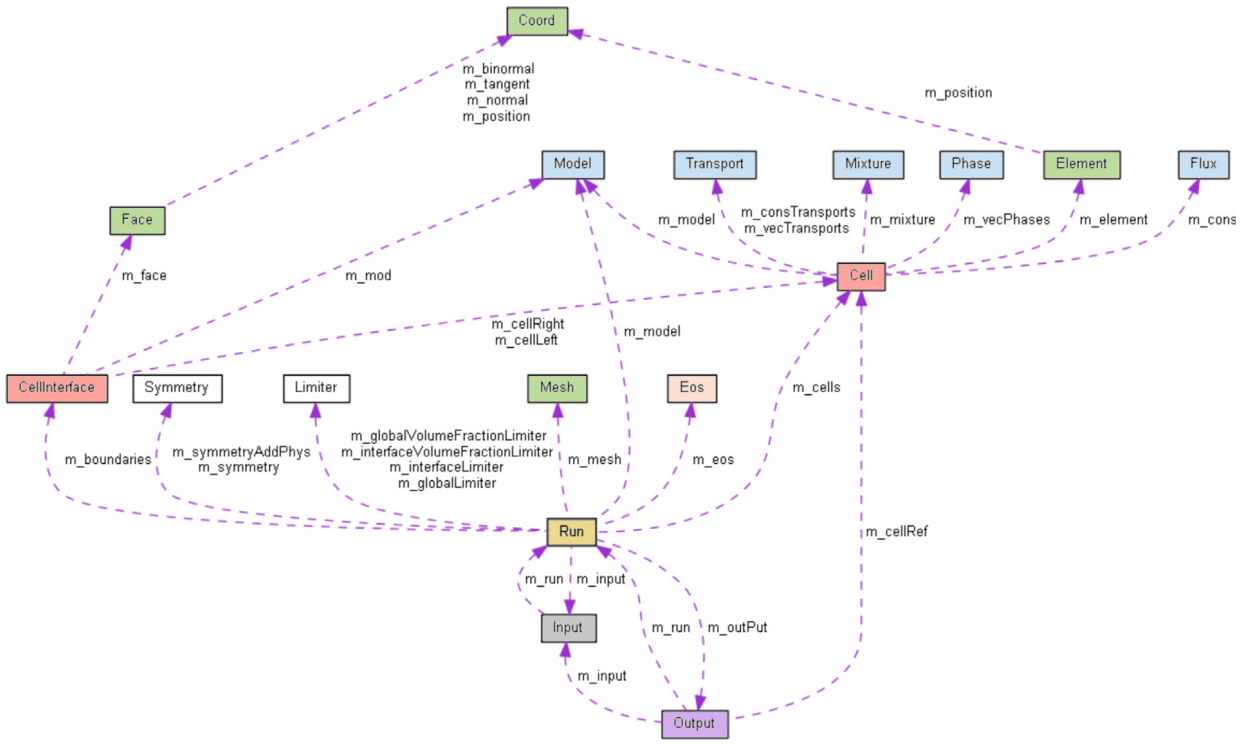

Fig. 4. Simplified global view of ECOGEN structure. Boxes represent base classes and dashed arrows show the links between them with the appropriate variables involved. Boxes are colored depending on their function (geometry, physics, etc.) in agreement with Figs. 2 and 3.

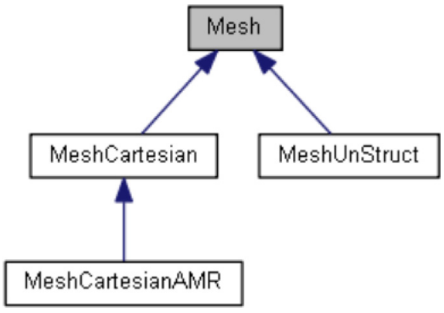

Fig. 5. Inheritance graph of **Mesh** class. A class arrow points to which base class it derives.

The model class is used to compute the different methods that are specific to a model. There is no specific attribute for the models. As example, Riemann solvers are methods implemented in model classes. Accordingly to the chosen model, the phases, mixtures and fluxes are based on the corresponding derived classes from base classes **Phase**, **Mixture** and **Flux**. Phase and mixture objects contain physical data of the fluids (state variables, velocities, etc.) accordingly to the chosen model (similar inheritance structures than in Fig. 7 for models exist for phases and mixtures). They are dedicated to be included in a computational cell (presented in the next section).

Objects instantiated as derived class from **Flux** (also similar inheritance structure than for models) contain a number of attributes function of the number of equations of the model. Fluxes are used for several purposes. The first one is to store, for each cell, either the sum of the computed fluxes at the cell interfaces or the unknown variables (conservative or not depending on the model), depending on which part of the time-stepping strategy is executed. The second one is to use the corresponding methods for the determination of the fluxes at cell interfaces.

Using inspiration of existing models to create four new classes derived from base classes **Model**, **Phase**, **Mixture** and **Flux** thus becomes possible in a total independence of geometry considerations. In other words, the interested developer can implement quite easily a new flow model in ECOGEN.

A typical example: The mechanical-equilibrium, multiphase flow model of Kapila et al. [48]

The multiphase flow model of Kapila et al. [48] has been widely used in the past decades to simulate interface problems as well as mixtures evolving in mechanical equilibrium in a flow [18,63]. Some interesting applications have been treated with extensions of this model. To simulate such flow with n phases involved and a mixture part, $n+1$ objects will be instantiated per computational cell. The n objects from *PhaseKapila* class include at least the following attributes:

- Volume fraction;
- density;
- pressure;
- pointer to the equation-of-state object.

And the object from *MixKapila* class includes at least the following mixture attributes:

- Density;
- pressure;
- velocity vector;
- total energy.

A flux object from *FluxKapila* includes the following attributes:

- Volume fraction of each phase;
- density of each phase,
- mixture momentum;
- mixture total energy.

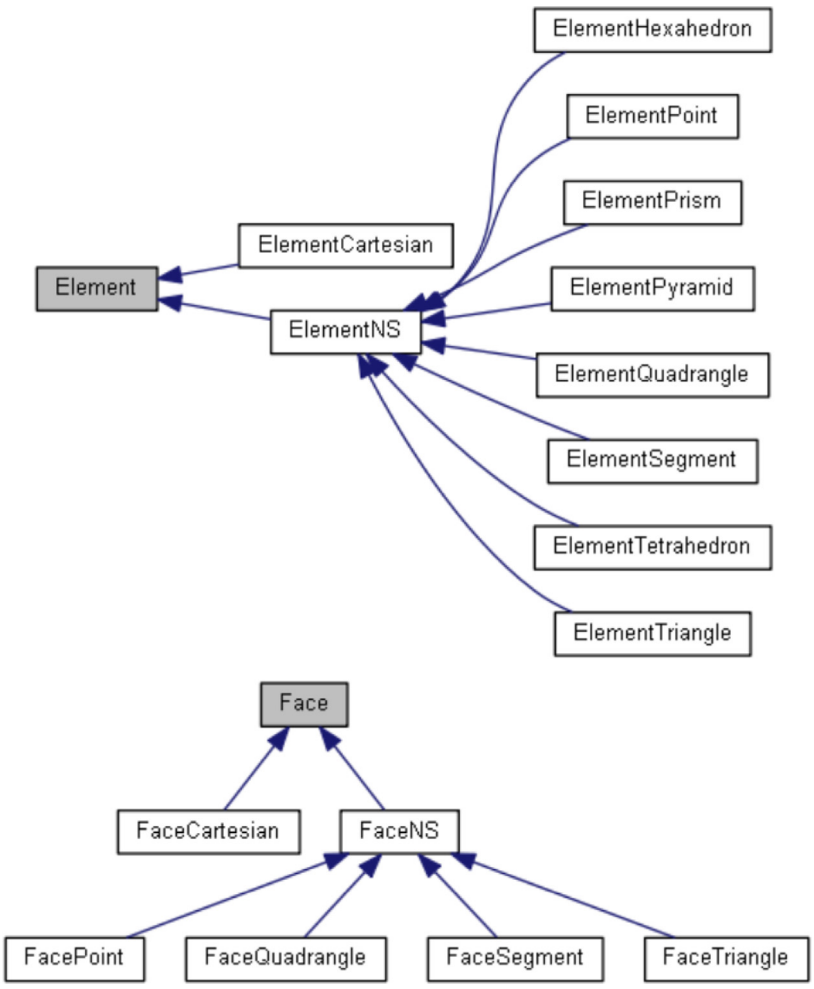

Fig. 6. Inheritance graphs of **Element** and **Face** classes.

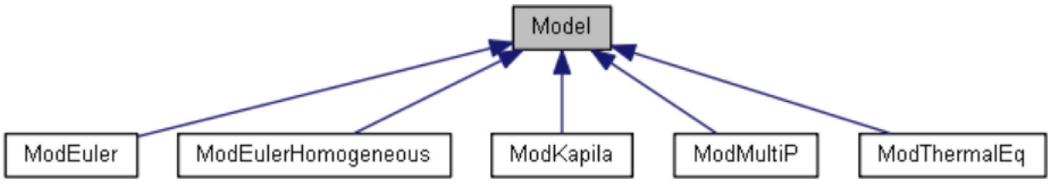

Fig. 7. Inheritance graph of **Model** class. 5 flow models are implemented in the version 1.0 of ECOGEN. Similar inheritance structures are present for **Phase**, **Mixture** and **Flux** classes.

3.2.2. Equation-of-state classes

The thermodynamical behavior of a fluid is described using an appropriate equation of state. The base class is named **EOS** and the inheritance graph of the class is shown in Fig. 8. Two equations of state are currently implemented: Ideal-gas and stiffened-gas [64,65] equations of state which allow a large range of flow simulations.

Each derived class contains a given number of attributes corresponding to the equation-of-state parameters as well as the needed methods used in ECOGEN to compute thermodynamics variables (*e.g.* temperature from pressure and density) or thermodynamical evolutions (*e.g.* isentropic process or process on the adiabatic Hugoniot curve). One can easily implement new EOS in ECOGEN by creating a new derived class of EOS.

3.2.3. Additional-transport-variable class

Objects instantiated as derived class from **Transport** class are initialized only if supplementary transport equations are required in the flow model. As example, when using the effects of surface tension coupled with the multiphase flow model of Kapila, an extra transport equation is needed to play the role of the color function [27].

3.2.4. Additional-physics classes

Is named "additional physics", the physical effects which are not taken into account in the base hyperbolic flow models. They may appear as fluxes, relaxation steps or source terms. In the case of class **AddPhys**, they are treated as fluxes following a splitting procedure, *i.e.*, the additional fluxes are calculated using the new primitive variables obtained from the hyperbolic step

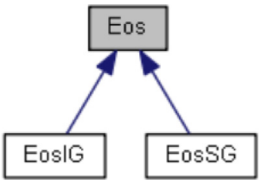

Fig. 8. Inheritance graph of **EOS** class. Two equations of state are implemented in version 1.0 of ECOGEN.

done with the **Model** base class introduced in Section 3.2.1. The source terms **Source** and the relaxation terms **Relaxation** follow the same procedure using each time the new primitive variables.

Fig. 9 shows the configurations of the different classes involved in the additional-physics part and details about the advancing procedure and the structure of these classes are given in Section 5 on the numerical solver of ECOGEN.

4. Data structure

Three kinds of grids are available: Unstructured grids, Cartesian grids and AMR Cartesian grids. Whatever the kind of grid is, the objects of classes **Element**, **Face**, **Cell** and **CellInterface** are all stored in 1D arrays but with different arrangements depending on the grid type.

4.1. Cartesian data structure

For Cartesian grids, ECOGEN is generating its own meshes. It also decomposed the Cartesian grid for multi-core simulations. An example of 3D Cartesian grid generated by ECOGEN is shown in Fig. 10. In this example, an array of 4400 Cartesian elements is initialized for a 4-core simulation. Note that in this peculiar example, the mesh is stretched in the X-direction forward and backward, starting from the middle. Stretching is possible in every direction allowing refined meshed zones in Cartesian computations.

For a mesh containing n_X, n_Y and n_Z elements in X-,Y- and Z-direction, respectively, two main arrays are generated:

- **Element** array: An array of $n_X \times n_Y \times n_Z$ Cartesian elements is generated where each element stores its geometrical information. In this array, the element e is stored with respect to the following formula:

$$\forall\,(i,j,k) \in ([0,n_X],[0,n_Y],[0,n_Z]), \quad e = i + j \times n_X + k \times n_X \times n_Y.$$

- **Face** array: An array of n_{faces} faces is generated to store geometrical information of each interface between two cells as well as boundary conditions with:

$$n_{faces} = (n_X+1)\times n_Y \times n_Z + n_X \times (n_Y+1) \times n_Z + n_X \times n_Y \times (n_Z+1).$$

Then, the data structure is ending with the generation of arrays of cells and cell interfaces dedicated to the storing of physical flow information. Each **Cell** is associated to an **Element** and each **CellInterface** is associated to a **Face**.

4.2. Dual-tree AMR data structure

For AMR simulations, the evolution of the flow dynamically modifies the data structure. The implemented AMR algorithm is based on dual trees for cells and faces, which is detailed in Schmidmayer et al. [26]. First, a base Cartesian grid is generated as presented in the previous section. Then, additional data will be initialized during the evolution of the simulation depending on geometrical regions that need to be refined. In those refined regions, a **Cell** originated from the base Cartesian grid becomes the root of a cell tree. In the same manner, the corresponding **CellInterface** (those linked to the refined cell) become roots of face trees. Both appearing tree structures are interlinked by additional information:

- A **Cell** is equipped with an integer for its level of refinement, a pointer to each of its child cell nodes, a pointer to each of its **CellInterface** and a pointer to new face trees (appearing inside the split cell).
- A **CellInterface** is equipped with an integer for its level of refinement and a pointer to each of its child face nodes.

A 2D example of this complex data structure is presented in Fig. 11. In this example, a cell is refined (top of the figure) and

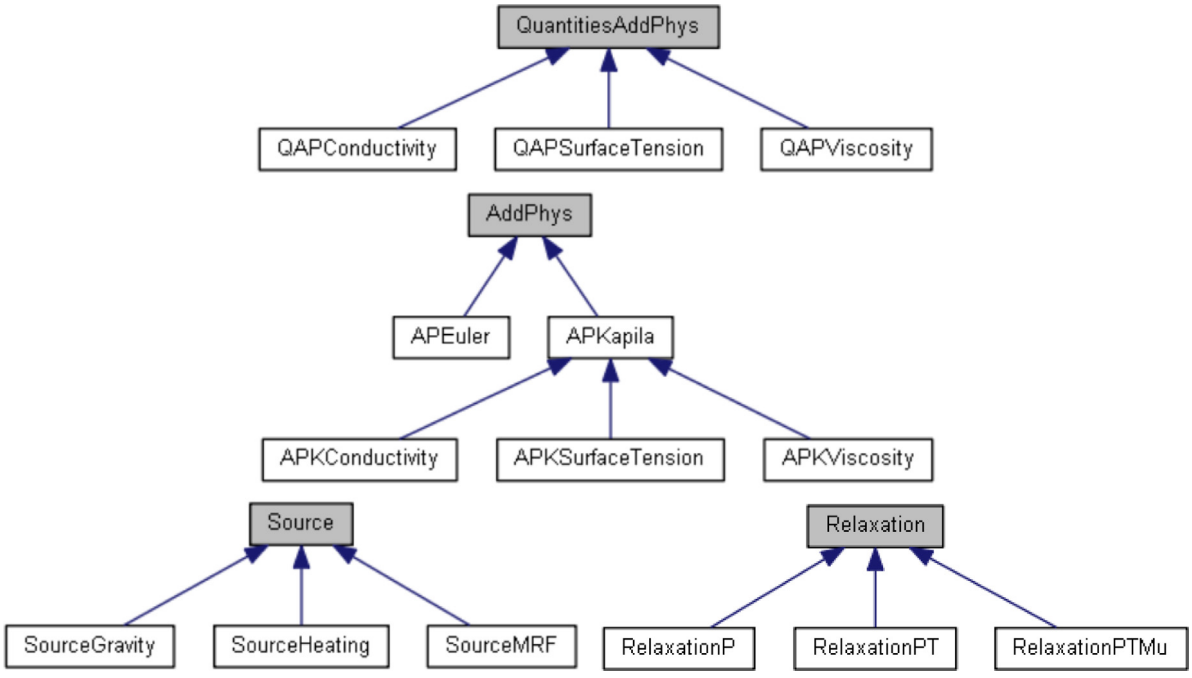

Fig. 9. Inheritance graphs of **QuantityAddPhys**, **AddPhys**, **Source** and **Relaxation** classes.

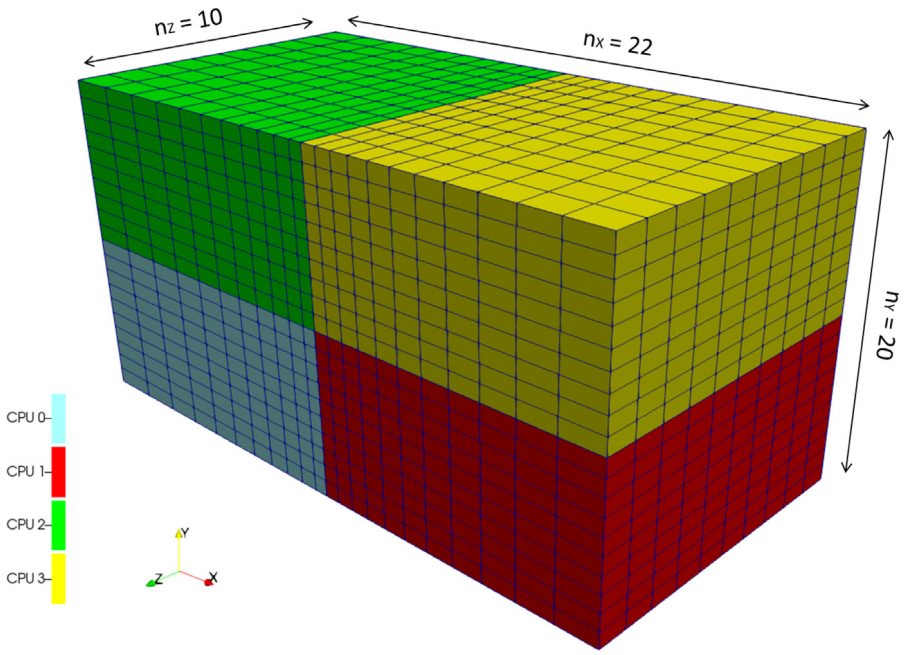

Fig. 10. 3D Cartesian grid example. The grid is generated for a 4-core simulation.

the figure is summarizing both induced tree data structures: The cell tree in the middle (with 4 level-1 cells) and the face trees in the bottom (4 level-0 face trees and 12 level-1 faces, including 4 new level-1 face trees). The adopted numeration is 'XYZ' with X being C (for cell) or B (for face), Y corresponds to level number (here, 0 or 1) and Z is the letter corresponding to the entity (A to N in the present case). Pointers between cells in cell tree, as well as pointers between faces in face trees, are shown with black solid lines. These pointers are organizing the tree structure. The other pointers involved in the method, which are pointers between cells and faces, are non-exhaustively presented in order to facilitate comprehension. These last pointers can be classified as follows:

- Pointers from cells to their external faces (2 in 1D, 4 in 2D and 6 in 3D). Such pointers are present for each cell and are needed for gradient calculations as well as slope determination for second-order scheme. Examples of such pointers are shown with yellow solid lines, where cell C0A points to the 4 faces B0A, B0B, B0C and B0D of level 0.
- Pointers from cells to internal face trees. When a cell is refined, new faces appear between child cells (1 in 1D, 4 in 2D and 12 in 3D). These faces have the possibility to become roots of new face trees which are linked to the parent cell thanks to pointers. These pointers are needed for refinement/unrefinement purpose. Examples of such pointers are shown with blue dotted lines, where cell C0A points to the 4 new faces B1K, B1L, B1M, and B1N.
- Pointers from faces to cells. Each face possesses two of these pointers which are used to compute hydrodynamic fluxes and update hydrodynamic part of the solution. These pointers are used only if the face is a leaf of a face tree. 3 examples corresponding to different situations are presented. In green dash-dotted lines are shown pointers from a face of level 1 (B1N) to two level-1 cells (C1C and C1D). The last example in red dashed lines shows a face (B1A) linked to a level-1 cell (C1A) and to another cell neighbor (level-0) of cell C0A or one of its child (level-1 cell) not shown in the figure.

This dual-tree structure induces an over-cost in regard to memory use compared to classical fully-threaded tree methods [66], but it improves efficiency by avoiding neighbor searches and thus simplifies the AMR global algorithm presented in Section 5.3. Full details of the AMR data structure are presented in [26].

Because of the recursive evolution algorithm, browsing of trees will lead to an important amount of tests to detect cell and face levels. To improve access in ECOGEN, cells and faces are listed for each level of the simulation in specific dynamic members of the **Run** class. These members are named *m_cellsLvl* and *m_boundariesLvl*, respectively. Indeed, loops over cells or faces become straightforward and constantly efficient.

We also note that within the version 1.0 of ECOGEN, AMR parallel balancing is not present. A simple static domain decomposition is thus used. The dynamic parallel balancing is going to be available within the next public version.

4.3. Unstructured meshes

For unstructured grids, the version 1.0 of ECOGEN is reading mesh files from Gmsh [57]: version 2 of MSH file format. Please refer to the Gmsh reference manual available online. Implemented unstructured finite elements are:

- 1D segment;
- 2D triangle;
- 2D quadrangle;
- 3D tetrahedron;
- 3D hexahedron;
- 3D prism;
- 3D pyramid.

5. Numerical solver

The present section explains the numerical methods and algorithms implemented to solve compressible flows in ECOGEN. The solver implemented in ECOGEN is explicit and solves hyperbolic systems of equations (conservative or not). The solver is particularly adapted to simulate unsteady flows. In this section is successively presented:

- The finite volume scheme and algorithm for conservation laws in the general case of unstructured meshes at first order;

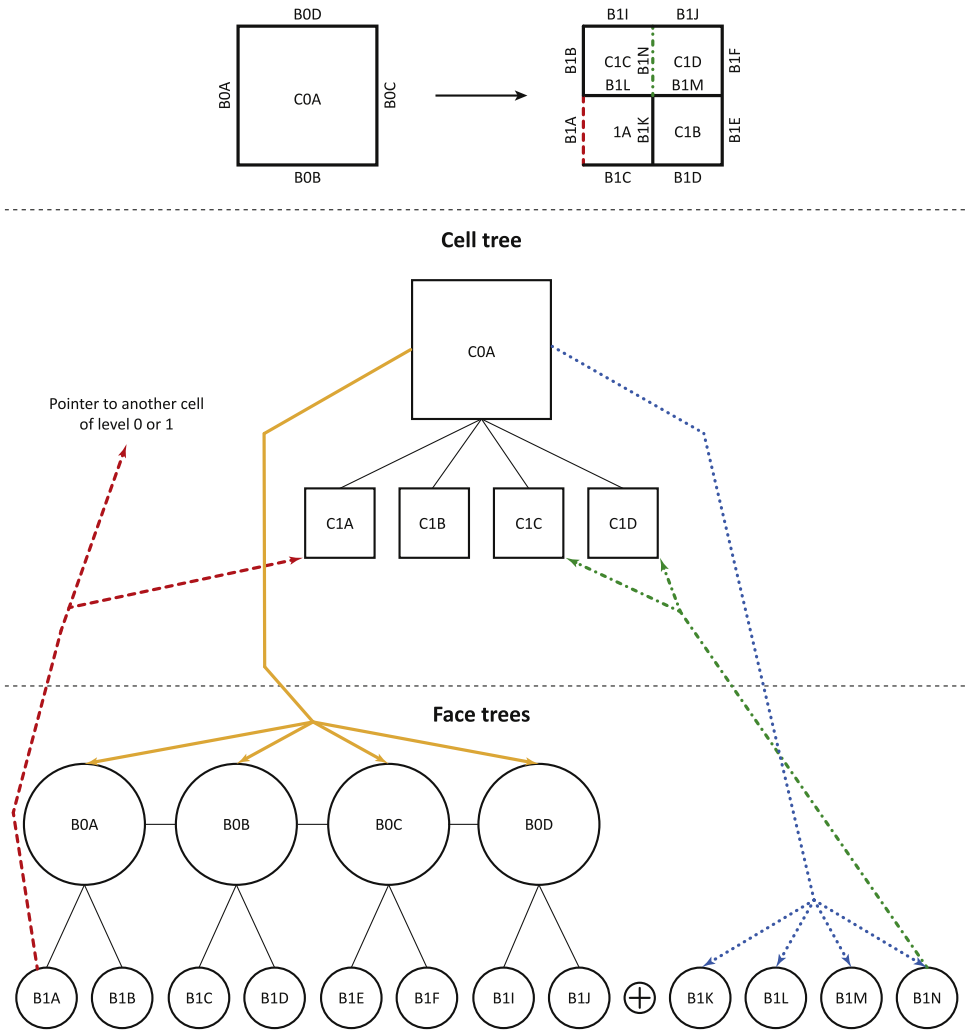

Fig. 11. 2D example of links between cell and face trees. Here, some details are given for a unique cell and its level of refinement. The top represents the cells. The middle and bottom sketches are representing the cell tree and face trees, respectively. Connections between both cell and face trees are presented for some typical situations.

- the treatment of non-conservative terms;
- the algorithm modifications induced by simulations using adaptive mesh refinement;
- the algorithm specificities induced by second-order accuracy;
- the additional finite volume scheme for additional physics;
- the source-term integration;
- the physical-relaxation methods for compressible multiphase flows.

The system of equations solved in ECOGEN can be written in the general vector form:

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

with $\mathbf{q}$ the vector of unknown variables, $\mathbf{F}$ the flux tensor and $\mathbf{u}$ the velocity field, while the vectors $\mathbf{p}$, $\mathbf{s}$ and $\mathbf{r}$ stand for the additional physics (*e.g.* surface tension or viscous dissipation), the source terms due to external effects and the relaxation effects between the phases, respectively. The vector $\mathbf{h}$ contains the quantities governed by non-conservative equations (*e.g.* the volume fraction).

The initial solution $\mathbf{q}^n$ is updated towards the solution at the time level $n+1$ according to the following chain of operators:

$$\mathbf{q}^{n+1} = \mathbf{L}_{\text{relax}} \mathbf{L}_{\text{source}} \mathbf{L}_{\text{addPhys}} \mathbf{L}_{\text{hyper}} \left(\mathbf{q}^n\right).$$

The sequence reads from the right to the left and is performed over one time step, each $\mathbf{L}_k$ operator is applied independently of the others. These operators are detailed in the next paragraphs.

5.1. Finite volume scheme for conservation laws

The base solver of ECOGEN solves a system of conservation laws in the following form:

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{q}) = \mathbf{0}. \tag{2}$$

Integration of System (2) on a computational cell i of volume V_i delimited by surface A of normal unit vector $\mathbf{n}$ (as shown in Fig. 12) reads:

$$\frac{\partial}{\partial t} \int_{V_i} \mathbf{q} \mathrm{dV} + \int_A \mathbf{F}(\mathbf{q}) \cdot \mathbf{n} \mathrm{dA} = \mathbf{0}. \tag{3}$$

The time evolution of the discretized System (3) is given for cell i by the following explicit scheme:

$$\mathbf{q}_i^{n+1} = \mathbf{q}_i^n - \frac{\Delta t}{V_i} \sum_{s=1}^{N} A_s \mathbf{F}_s^{\star} \cdot \mathbf{n}_s, \tag{4}$$

where $\mathbf{F}_s^{\star}$ represents the flux-tensor solution of the Riemann problem between left (L) and right (R) states separated by a surface of area A_s with respect to normal $\mathbf{n}_s$. Scheme (4) is restricted

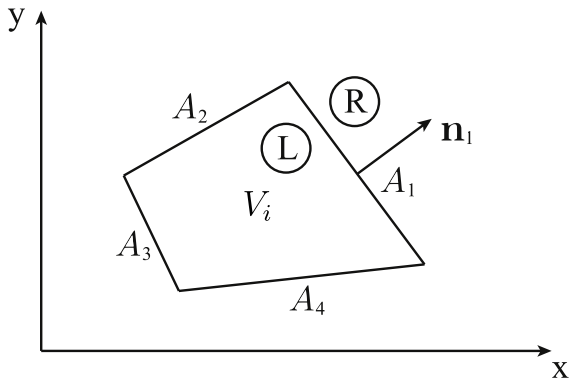

Fig. 12. Schematic view of a 2D computational cell i.

by a CFL criterion determined by solving the associated Riemann problems.

In ECOGEN, this evolution is ensured by the complementarity between cells and their interfaces described in the data structure of the preceding section and it is expressed through the simple following pseudo-algorithm:

1. — Flux computation and buffering —

   for (each cell interface s of area A_s) {
   Compute the hyperbolic flux tensor $\mathbf{F}_s^\star = \mathbf{F}_s^\star\left(\mathbf{q}_\mathrm{L}^n, \mathbf{q}_\mathrm{R}^n\right)$
   by solving the appropriate
   Riemann solver and buffering it into the corresponding
   left and right cell flux
   objects:
   $\widetilde{\mathbf{F}}_\mathrm{L} = \widetilde{\mathbf{F}}_\mathrm{L} - A_s\mathbf{F}_s^\star \cdot \mathbf{n}_s$;
   $\widetilde{\mathbf{F}}_\mathrm{R} = \widetilde{\mathbf{F}}_\mathrm{R} + A_s\mathbf{F}_s^\star \cdot \mathbf{n}_s$;
   }

2. — Unknown-variable cell evolution —

   for (each cell i) {
   $\mathbf{q}_i^{n+1} = \mathbf{q}_i^n + \frac{\Delta t}{V_i}\widetilde{\mathbf{F}}_i$;
   $\widetilde{\mathbf{F}}_i = \mathbf{0}$;
   }

This algorithm is considered as the advancing procedure $Ad\,(l)$ of refinement level l in Section 5.3 about AMR specificities.

5.2. Particular treatment of non-conservative terms

Multiphase models implemented in ECOGEN may present non-conservative terms which can be written under the general form $\mathbf{h}\,(\mathbf{q})\,\nabla\cdot\mathbf{u}$. Then, a special treatment dedicated to this kind of non-conservative terms is implemented in ECOGEN. The non-conservative part of the system of equations thus reads:

$$\frac{\partial \mathbf{q}}{\partial t} + \mathbf{h}\,(\mathbf{q})\,\nabla\cdot\mathbf{u} = \mathbf{0},$$

This form of non-conservative terms is then solved in ECOGEN using the following explicit scheme for cell i:

$$\mathbf{q}_i^{n+1} = \mathbf{q}_i^n - \frac{\Delta t}{V_i}\mathbf{h}\left(\mathbf{q}_i^n\right)\sum_{s=1}^{N} A_s\mathbf{u}_s^\star \cdot \mathbf{n}_s,$$

where $\mathbf{u}_s^\star$ represents the flow velocity solution of the previously solved Riemann problem on interface s. The flux buffering part of the pseudo-algorithm of Section 5.1 is modified as follows:

— Flux computation and buffering with non-conservative terms —

for (each cell interface s of area A_s) {
$\widetilde{\mathbf{F}}_\mathrm{L} = \widetilde{\mathbf{F}}_\mathrm{L} - A_s\left(\mathbf{F}_s^\star\left(\mathbf{q}_\mathrm{L}^n, \mathbf{q}_\mathrm{R}^n\right) + \mathbf{h}\left(\mathbf{q}_\mathrm{L}^n\right)\mathbf{u}_s^\star\right)\cdot\mathbf{n}_s$;
$\widetilde{\mathbf{F}}_\mathrm{R} = \widetilde{\mathbf{F}}_\mathrm{R} + A_s\left(\mathbf{F}_s^\star\left(\mathbf{q}_\mathrm{L}^n, \mathbf{q}_\mathrm{R}^n\right) + \mathbf{h}\left(\mathbf{q}_\mathrm{R}^n\right)\mathbf{u}_s^\star\right)\cdot\mathbf{n}_s$;
}

5.3. AMR specificities

ECOGEN is equipped with a new adaptive mesh refinement method using dual trees for cells and faces. The global method is presented in [26] and here is presented a summary of ECOGEN solver modifications needed for AMR simulations.

The efficiency of an AMR method requires the implementation of a specific time-stepping strategy based on two key points:

- Cells at different refinement levels evolve with different time steps according to their level of refinement. In order to maintain the global time-step coherence for unsteady simulations, if cells of level l evolve at a given time step, cells of level $l+1$ (two times smaller in each direction than cells of level l) will then evolve 2 times with a time step 2 times smaller. It thus avoids to compute the smallest time step of the entire computational domain, necessary for stability, for every cell but only for the ones where it is necessary, *i.e.*, the cells at the highest level (l_max). Then, it results in a saving of CPU time.
- This time-stepping strategy allows interleaving between time integration and tree refinement. It results in a saving of memory as it limits excessive buffer layer of refinement ahead a discontinuity [66].

Time steps at various levels thus are:

$$\Delta t\,(l) = 2^{l_\mathrm{min}-l}\Delta t,$$

where l_min is the minimum tree (refinement) level and Δt is the global time step.

The general integration procedure occurs at the different levels of the tree as an interleaving of advancing and refinement procedures. It is expressed as a recursive procedure $I(l_\mathrm{min})$ with:

$$\begin{aligned} I\,(l_\mathrm{min}) &= Ad\,(l_\mathrm{min})\,I\,(l_\mathrm{min}+1)\,R\,(l_\mathrm{min})\,, \\ I\,(l) &= Ad\,(l)\,I\,(l+1)\,Ad\,(l)\,I\,(l+1)\,R\,(l) \\ &\qquad \text{for} \quad l \neq (l_\mathrm{min}, l_\mathrm{max})\,, \\ I\,(l_\mathrm{max}) &= Ad\,(l_\mathrm{max})\,Ad\,(l_\mathrm{max})\,R\,(l_\mathrm{max})\,, \end{aligned} \tag{5}$$

where $Ad(l)$ represents the advancing procedure of level l described in the preceding section and $R(l)$ is the refinement /unrefinement procedure of level l detailed in [26]. All procedures in (5) are performed from right to left, *i.e.*, $R\,(l)$ first, and $Ad\,(l)$ last.

5.4. Second-order scheme

The second-order scheme used in ECOGEN is inspired by the MUSCL method where spacial reconstruction is achieved through cell slopes and where a two-step integration in time is used (the first step is a prediction for the second step). However, the prediction step in ECOGEN is not accomplished using the only geometry but also by computing a Riemann problem. The sequence is thus the following:

- Determine the boundary extrapolated values $\mathbf{q}_{\mathrm{L}}^{n}$ and $\mathbf{q}_{\mathrm{R}}^{n}$ for each cell using the usual piece-wise linear MUSCL reconstruction [50]: Determine the slope at each interface using the values $\mathbf{q}^{n}$ of its two neighboring cells and then, in each cell, use a slope limiter between its face slopes to determine the final cell slope. Finally, from these cell slopes, the boundary extrapolated values are obtained (Fig. 13).
- Compute the hyperbolic flux tensor $\mathbf{F}^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n}, \mathbf{q}_{\mathrm{R}}^{n}\right)$ by solving a first Riemann problem.
- Evolve the cell unknown variables $\mathbf{q}^{n}$ to a half time step to obtain $\mathbf{q}^{n+\frac{1}{2}}$ and thus finish the prediction step.
- Repeat the process to obtain the new boundary extrapolated values $\mathbf{q}_{\mathrm{L}}^{n+\frac{1}{2}}$ and $\mathbf{q}_{\mathrm{R}}^{n+\frac{1}{2}}$ from the piece-wise linear MUSCL reconstruction and then compute the new hyperbolic flux tensor $\mathbf{F}^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n+\frac{1}{2}}, \mathbf{q}_{\mathrm{R}}^{n+\frac{1}{2}}\right)$ by solving a second Riemann problem.
- Evolve the cell unknown variables $\mathbf{q}^{n}$ to a full time step to obtain $\mathbf{q}^{n+1}$ and thus finish the sequence.

In ECOGEN, the slopes are computed using the primitive variables and the available slope limiters are:

- Minmod [50,67];
- van Leer [68];
- van Albada [69];
- MC (Monotonized Central) [70];
- Superbee [67,71];
- THINC (Tangent of Hyperbola for INterface Capturing) interface-sharpening method [52,72] for interface between two fluids.

The simplified pseudo-algorithm of the second-order scheme to solve System (2) is:

1. — Determination of boundary extrapolated values —

   for (each cell interface) {
   Determine the cell-interface slope using the values $\mathbf{q}^{n}$ of the two neighboring cells;
   }

   for (each cell) {
   Determine the cell slope using a slope limiter between its face slopes;
   Extrapolate the values at the cell boundary to obtain $\mathbf{q}_{\mathrm{L}}^{n}$ and $\mathbf{q}_{\mathrm{R}}^{n}$;
   }

2. — Flux computation and buffering —

   for (each cell interface s of area A_s) {
   $\widetilde{\mathbf{F}}_{\mathrm{L}} = \widetilde{\mathbf{F}}_{\mathrm{L}} - A_s \mathbf{F}_s^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n}, \mathbf{q}_{\mathrm{R}}^{n}\right) \cdot \mathbf{n}_s$;
   $\widetilde{\mathbf{F}}_{\mathrm{R}} = \widetilde{\mathbf{F}}_{\mathrm{R}} + A_s \mathbf{F}_s^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n}, \mathbf{q}_{\mathrm{R}}^{n}\right) \cdot \mathbf{n}_s$;
   }

3. — Unknown-variable cell evolution for the prediction step —

   for (each cells i) {
   $\mathbf{q}_i^{n+\frac{1}{2}} = \mathbf{q}_i^{n} + \frac{\Delta t}{2V_i}\widetilde{\mathbf{F}}_i$;
   $\widetilde{\mathbf{F}}_i = \mathbf{0}$;
   }

4. — Determination of boundary extrapolated values —

   for (each cell interface) {
   Determine the cell-interface slope using the values $\mathbf{q}^{n+\frac{1}{2}}$ of the two neighboring cells;
   }

   for (each cell) {
   Determine the cell slope using a slope limiter between its face slopes;
   Extrapolate the values at the cell boundary to obtain $\mathbf{q}_{\mathrm{L}}^{n+\frac{1}{2}}$ and $\mathbf{q}_{\mathrm{R}}^{n+\frac{1}{2}}$;
   }

5. — Flux computation and buffering —

   for (each cell interface s of area A_s) {
   $\widetilde{\mathbf{F}}_{\mathrm{L}} = \widetilde{\mathbf{F}}_{\mathrm{L}} - A_s \mathbf{F}_s^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n+\frac{1}{2}}, \mathbf{q}_{\mathrm{R}}^{n+\frac{1}{2}}\right) \cdot \mathbf{n}_s$;
   $\widetilde{\mathbf{F}}_{\mathrm{R}} = \widetilde{\mathbf{F}}_{\mathrm{R}} + A_s \mathbf{F}_s^{\star}\left(\mathbf{q}_{\mathrm{L}}^{n+\frac{1}{2}}, \mathbf{q}_{\mathrm{R}}^{n+\frac{1}{2}}\right) \cdot \mathbf{n}_s$;
   }

6. — Unknown-variable cell evolution for the hyperbolic part —

   for (each cell i) {
   $\mathbf{q}_i^{\mathrm{h}} = \mathbf{q}_i^{n} + \frac{\Delta t}{V_i}\widetilde{\mathbf{F}}_i$;
   $\widetilde{\mathbf{F}}_i = \mathbf{0}$;
   }

where $\mathbf{q}_i^{\mathrm{h}}$ corresponds to the vector of unknown variables with the only hyperbolic part of System (1) taken into account. The $\mathbf{L}_{\mathrm{hyper}}$ operator is thus a combination of the four algorithms previously described in this section: The finite volume scheme is its basis, AMR and second-order method are optional extensions while the non-conservative algorithm is applied accordingly to the form of the equations.

5.5. Additional physics

The $\mathbf{L}_{\mathrm{addPhys}}$ operator describes additional physics not solved in the hyperbolic part of the multiphysics model (previous sections) and not related to the integration of source terms or relaxation procedures. Typically, this operator is applied to models that are split to be solved through particular procedures when these models either involve complex structures of Riemann solvers or when they have second-order terms, such as for surface tension [27], solid mechanics [41], shallow water [73], or viscosity and heat conduction [59,60,74]. Those types of physics often involve the calculation of gradients to compute the fluxes and the associated pseudo-algorithm is thus the following:

7. — Gradient computation —

   for (each cell i) {
   Compute the necessary gradients $\mathbf{g}_i = \mathbf{g}_i\left(\mathbf{q}_i^{\mathrm{h}}\right)$;
   }

8. — Flux computation and buffering with non-conservative terms —

   for (each cell interface s of area A_s) {
   $\widetilde{\mathbf{F}}_{\mathrm{L}} = \widetilde{\mathbf{F}}_{\mathrm{L}} - A_s\left(\mathbf{F}_s^{\mathrm{ap}}\left(\mathbf{q}_{\mathrm{L}}^{\mathrm{h}}, \mathbf{q}_{\mathrm{R}}^{\mathrm{h}}, \mathbf{g}_{\mathrm{L}}, \mathbf{g}_{\mathrm{R}}\right) + \mathbf{h}^{\mathrm{ap}}\left(\mathbf{q}_{\mathrm{L}}^{\mathrm{h}}, \mathbf{g}_{\mathrm{L}}\right)\mathbf{u}_s^{\mathrm{h}}\right) \cdot \mathbf{n}_s$;
   $\widetilde{\mathbf{F}}_{\mathrm{R}} = \widetilde{\mathbf{F}}_{\mathrm{R}} + A_s\left(\mathbf{F}_s^{\mathrm{ap}}\left(\mathbf{q}_{\mathrm{L}}^{\mathrm{h}}, \mathbf{q}_{\mathrm{R}}^{\mathrm{h}}, \mathbf{g}_{\mathrm{L}}, \mathbf{g}_{\mathrm{R}}\right) + \mathbf{h}^{\mathrm{ap}}\left(\mathbf{q}_{\mathrm{R}}^{\mathrm{h}}, \mathbf{g}_{\mathrm{R}}\right)\mathbf{u}_s^{\mathrm{h}}\right) \cdot \mathbf{n}_s$;
   }

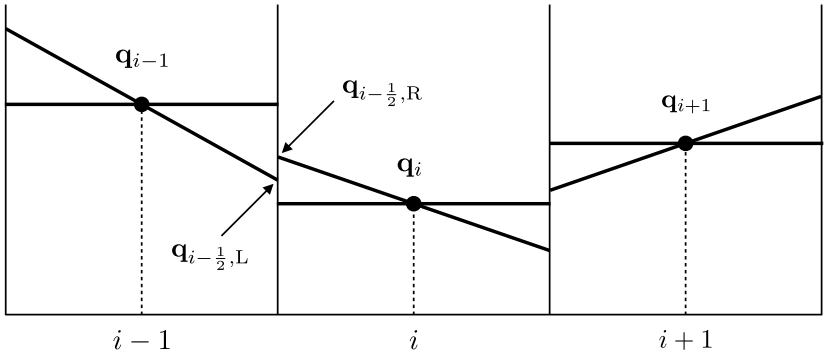

Fig. 13. Piece-wise linear MUSCL reconstruction for three successive computing cells $i-1$, i, $i+1$. Boundary extrapolated values at interface $i-\frac{1}{2}$ from cells $i-1$ and i are $\mathbf{q}_{i-\frac{1}{2},\mathrm{L}}$ and $\mathbf{q}_{i-\frac{1}{2},\mathrm{R}}$, respectively.

9. — Unknown-variable cell evolution for the additional physics part —

for (each cell i) {
$\mathbf{q}_i^{\mathrm{ap}} = \mathbf{q}_i^{\mathrm{h}} + \frac{\Delta t}{V_i}\widetilde{\mathbf{F}}_i$;
$\widetilde{\mathbf{F}}_i = \mathbf{0}$;
}

where the superscript "ap" stands for the additional physics.

5.6. Source-term integration

The source-term operator $\mathbf{L}_{\mathrm{source}}$ performs a time integration of a system of ordinary differential equations. In this ECOGEN released version, the gravity acceleration, the terms due to rotating frame used for rotating flows (Coriolis force, centrifugal force) and heat exchange through duct walls may be considered. Because there are constant source terms, a simple solver based on first-order Euler method is sufficient and leads to a one-step scheme:

10. — Unknown-variable cell evolution for the source-term integration part —

for (each cell i) {
$\mathbf{q}_i^{\mathrm{s}} = \mathbf{q}_i^{\mathrm{ap}} + \Delta t \mathbf{s}_i\,(\mathbf{q}^{\mathrm{ap}})$;
}

where the superscript "s" stands for the source terms.

5.7. Physical-relaxation methods

ECOGEN is primarily dedicated to multiphase flows simulation in the framework of diffuse-interface method [58]. The present released version only treats single velocity flows. More precisely, implemented models are particularly adapted to describe flows composed of several phases evolving with the same velocity. In other words, drag effects between phases are considered to locally (at the cell scale) be intense enough to consider that the time scale under interests does not require to take into account for multiple velocities. Nevertheless, thermodynamics variables may be considered as evolving separately. This concerns the pressure p, the temperature T and the chemical potential μ (or Gibbs free energy ε). For several implemented models, ECOGEN allows to occasionally perform a simulation with additional constraints on thermodynamics variables. These constraints are obtained by using physical-relaxation procedures. This kind of procedure is popular in the multiphase flow theory [75–78]. Implemented procedures systematically consist in finding the final equilibrium state "f" of an isolated thermodynamic system (the cell) from an initial disequilibrium state "0". This initial state is obtained at each time step as the solution result of the previous operator and the relaxed state corresponds to the application of the $\mathbf{L}_{relax}$ operator. The corresponding pseudo-algorithm is:

11. — Unknown-variable cell evolution for the relaxation part —

for (each cell i) {
$\mathbf{q}_i^{n+1} = \mathbf{L}_{\mathrm{relax}}\left(\mathbf{q}_i^{\mathrm{s}}\right)$;
}

where $\mathbf{L}_{\mathrm{relax}}$ corresponds to one of the relaxation procedures presented in Appendix.

6. Simulation validation

Several test cases that validate and verify ECOGEN's capabilities are presented in this section. These include one-, two-, and three-dimensional test cases that span a wide variety of flow problems.

6.1. Validation against analytic solutions

ECOGEN has been validated against analytic solutions over 1D and multidimensional, single phase and multi-phase flows:

- Simple 1D and 2D transport tests for single phase and multiphase configurations [26,79];
- 1D liquid–gas shock tubes with high-density and high-pressure ratios [26,55,79];
- 2D and 3D surface-tension tests to verify Laplace's law [26,27,79];
- 3D spherical bubble collapse to compare ECOGEN's numerical modeling to the semi-analytic solution given by the Keller–Miksis equation [80] (a compressible form of the Rayleigh–Plesset equation) [55]. Fig. 17, presented in Section 6.3, shows few of the main results of [55]. An example of spherical-bubble-collapse validation is given below in a 2D-axisymmetric context.

A spherical air bubble is initialized at atmospheric state in the corner of a 2D-axisymmetric domain filled with water at high pressure of $p_{\mathrm{w}} = 500 \times 10^5$ Pa (Fig. 14). The domain size is 6 mm x 6 mm. The initial radius of the bubble is $R_{\mathrm{b}} = 0.5$ mm. The left and bottom boundaries imply both symmetry condition. Non-reflecting (absorption) boundary conditions are used elsewhere.

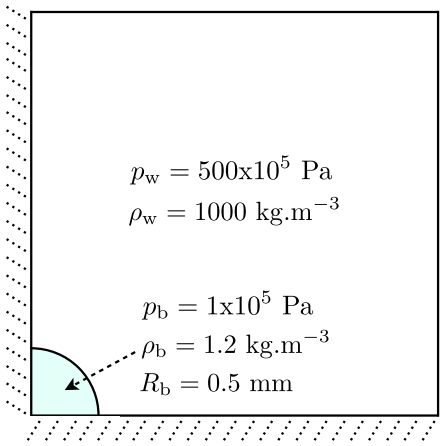

Fig. 14. Initial configuration of the test case of the spherical bubble collapse.

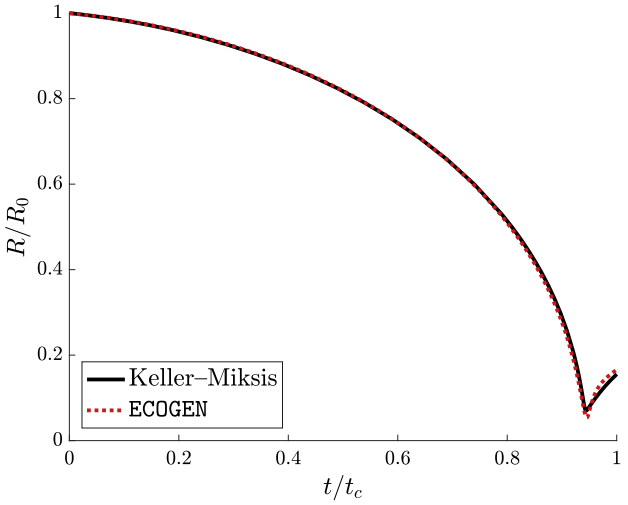

Fig. 15. Radial bubble-wall evolution. Bubble radius R is non-dimensionalized by the initial radius R_0 and time t is non-dimensionalized by the theoretical Rayleigh-collapse time t_c [83]. Comparison between the solutions from the Keller–Miksis equation [80] and the simulation.

The hyperbolic model of Kapila et al. [48] is used:

$$\begin{cases} \dfrac{\partial \alpha_1}{\partial t} + \mathbf{u}\cdot\nabla\alpha_1 & = K\nabla\cdot\mathbf{u}, \\ \dfrac{\partial \alpha_1\rho_1}{\partial t} + \nabla\cdot(\alpha_1\rho_1\mathbf{u}) & = 0, \\ \dfrac{\partial \alpha_2\rho_2}{\partial t} + \nabla\cdot(\alpha_2\rho_2\mathbf{u}) & = 0, \\ \dfrac{\partial \rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u} + p\mathbf{I}) & = \mathbf{0}, \\ \dfrac{\partial \rho E}{\partial t} + \nabla\cdot((\rho E + p)\,\mathbf{u}) & = 0, \end{cases} \tag{6}$$

where subscripts "1" and "2" correspond to one of the two phases, respectively. α_k and ρ_k are the volume fraction and density of phase k. $\rho = \sum_k \alpha_k\rho_k$, $\mathbf{u}$, p, $E = e + \frac{1}{2}\|\mathbf{u}\|^2$ and $e = \sum_k \alpha_k\rho_k e_k$ are the mixture density, velocity, pressure, total energy and internal energy, respectively. The term $K\nabla\cdot\mathbf{u}$ accounts for the differences in the acoustic behavior of both phases or in other words, for the differences in expansion and compression of each phase in mixture regions. K is given by:

$$K = \frac{\rho_2 s_2^2 - \rho_1 s_1^2}{\frac{\rho_2 s_2^2}{\alpha_2} + \frac{\rho_1 s_1^2}{\alpha_1}},$$

s_k being the speed of sound of phase k. A relaxation procedure is also used to solve the model. Complete details are available in [18].

The stiffened-gas parameters for the water are $\gamma_w = 4.4$ and $\pi_{\infty,w} = 6 \times 10^8$ Pa [5,18,26,27,45,81], and the ideal-gas parameter for the air inside the bubble is $\gamma_b = 1.4$. The mesh initially presents, in each direction, 150 cells between 0 mm and 1.5 mm and it is stretched over 40 cells, with an aspect ratio of 1.1 between each cell, until it reaches 6 mm. AMR is used with 3 levels of refinement and thus allows to obtain an effective resolution of 400 cells per initial bubble radius. In addition to the AMR, the THINC interface-sharpening method is used.

Fig. 15 shows the evolution of the bubble radius R for the solution given by the simulation and for the semi-analytic solution given by the Keller–Miksis equation for spherical bubble collapse with compressibility effect taken into account [80]. The radius of the simulation is computed from the gas volume by assuming the shape is spherical. The simulation solution is in very good agreement with the semi-analytic solution for the collapse and slightly less for the rebound. However, one should note that the Keller–Miksis equation holds under the assumption of low-Mach interface speed [82] while in the present test case, a Mach number of 1.189 is measured at the instant just before reaching the minimum bubble radius. The rebound part is thus not fully shown in Fig. 15.

6.2. Validation against experimental results

In ECOGEN's validation process, it has also been compared to experimental results, where good agreements were observed, for flows such as:

- Shock wave traveling through a bifurcation in an attenuation -study context [84]. The experiment used a shock tube and a branch with a trap, while the modeling used an unstructured mesh for this single-phase configuration (Euler equations). The incident Mach number was 1.44;
- dynamics of microbubbles at the surface of urinary stones [10]. Ultra-high-speed video microscopy was used and the lipid-shell microbubbles designed to accumulate on stone surfaces were driven by bursts of ultrasound in the sub-MHz range with pressure amplitudes on the order of 1 MPa. Microbubbles were observed to undergo repeated cycles of expansion and violent collapse. At maximum expansion, the microbubbles' cross-section resembled an ellipse truncated by the stone. Approximating the bubble shape as an oblate spheroid, this study modeled the collapse by solving System (6) within a 2D-axisymmetric configuration and with AMR [26] for fine resolution of the gas–liquid interface. Modeled bubble collapse and high-speed video microscopy showed a distinctive circumferential pinching during the collapse. Modeled pressure spikes had amplitudes two-to-three orders of magnitude greater than that of the driving wave. Micro-computed tomography was used to study surface erosion and formation of microcracks from the action of microbubbles. This study suggests that engineered microbubbles enable stone-treatment modalities with driving pressures significantly lower than those required without the microbubbles;
- interaction of a shock wave with a water column and aerobreakup of a water column and of a water droplet by a high-speed flow [27,79]. The latter being initiated by shock

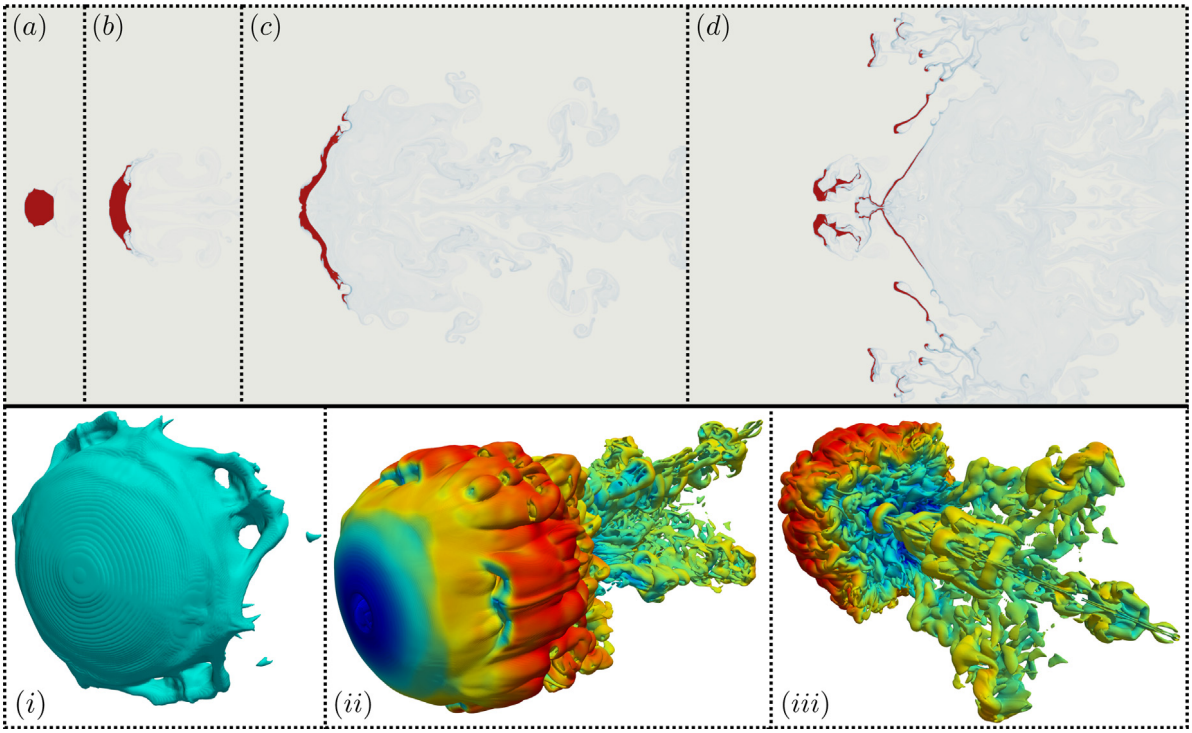

Fig. 16. 2D water-column atomization (*a*)–(*d*) and 3D water-droplet breakup (*i*)–(*iii*) [79]. (*a*) 200 µs, (*b*) 600 µs, (*c*) 1400 µs, (*d*) 2200 µs and (*i*)–(*iii*) 281 µs after the first interaction between the shock wave and the column or the droplet. In (*a*)–(*d*) is shown in red a threshold on the volume fraction of water for values equal or superior to 0.1 ($\alpha \geqslant 0.1$) and in blue what is considered as a mist of micrometer water columns (2D droplets). In (*i*) is shown contours of $\alpha = 0.01$ (cyan), while in (*ii*)–(*iii*) vorticity contours are represented and colored by the velocity magnitude. (*a*)–(*d*) © 2017, Aix-Marseille Université. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

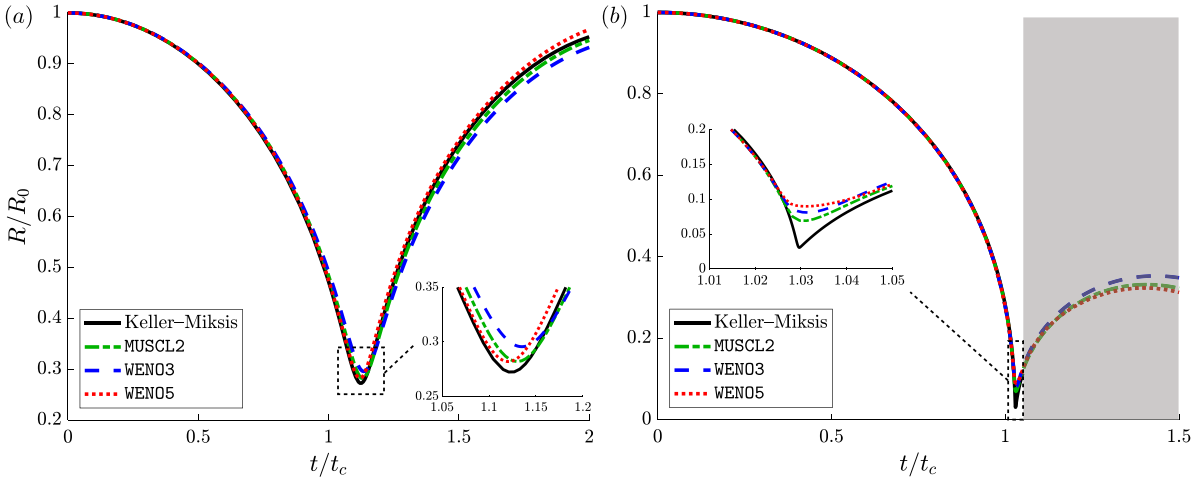

Fig. 17. Radial bubble-wall evolution for (*a*) $p_\infty/p_b = 10$ and (*b*) $p_\infty/p_b = 1427$ [55]. Solutions are computed using `MUSCL2` (`ECOGEN`), `WENO3` and `WENO5` (`MFC` [54]) and the Keller–Miksis equation [80].

wave. The waves propagation and the first stages of a water-column breakup for a Mach number of 1.3 were compared to the experiments of Igra and Takayama [32]. In the experiments, two transparent optical windows were placed in a shock tube. A separation of 4 mm in height was present and the authors claim that the shock wave propagating inside the 4 mm high channel was only very slightly perturbed by the windows. The water column was initially produced and maintained between those two windows. The height of the water column was thus equal to the height of the channel. On the side of the modeling, surface tension [27] and AMR [26] were taken into account. The column displacement, the flow-direction and the lateral deformations were assessed and showed good agreement. Few of the 2D and 3D results produced by `ECOGEN` are presented in Fig. 16 for illustration purposes. For the 2D water-column atomization case, the initial column diameter is $D_0 = 6.4$ mm and it is exposed to a shock wave of Mach number 1.3 in atmospheric air. For the 3D water-droplet breakup case, $D_0 = 1.3$ mm and the flow conditions are similar to the ones of the 2D case. The corresponding initial Weber numbers in these conditions are 3690 and 750, respectively.

6.3. Comparison of ECOGEN with MFC

There is a few open-source codes able to solve compressible multiphase flow equations. MFC [54] is one of them and some comparisons with ECOGEN have been performed on a very challenging test case in Schmidmayer et al. [55] about dynamics of a spherical gas bubble.

The numerical results must properly treat the compressibility effects in the mixture region, the discrete conservation must be enforced while the sphericity of the bubble should be maintained in the presence of large density and pressure ratios. Both codes use diffuse-interface methods and, in this study, solved the multicomponent model presented in Section 6.1: System (6). ECOGEN uses the second-order-accurate MUSCL scheme presented in Section 5.4 (labeled here as MUSCL2), while MFC uses either a third- or a fifth-order-accurate WENO scheme (labeled here as WENO3 and WENO5, respectively).

A collapsing and rebounding air bubble at pressure p_b in water at 10 or 1427 times higher pressure p_∞ was considered. Simulation specifications were presented in [55]. Fig. 17 shows the evolution of the bubble radius; it reaches a minimum near the nominal Rayleigh collapse time t_c [83], then rebounds, as expected. This closely matches the solution expected following the Keller–Miksis equation [80]. Note that for the $p_\infty/p_b = 1427$ case, the Keller–Miksis solution is not sufficiently trustworthy after such a strong collapse and is thus not represented (gray area). We also observe that MFC is performing slightly better for the relatively slow bubble dynamics ($p_\infty/p_b = 10$ case) when using the WENO5 numerical method. This is due to the more substantial numerical diffusion intrinsic to the lower-order schemes, even though the WENO5 scheme required an initially smeared interface to maintain simulation stability. One can note ECOGEN has a slight lead for the fast bubble dynamics ($p_\infty/p_b = 1427$ case). Similar trend was also observable for the bubble sphericity [55]. Further, they noticed that the relatively small bubble size at the collapse time resulted in significantly distorted interface shapes. However, these shapes were shown to be more spherical for finer spatial meshes. Thus, an adaptive-mesh-refinement technique, such as the one developed in ECOGEN [26], would help maintain bubble interface sphericity at the same computational cost as a uniform mesh near the bubble. Finally, this validates again ECOGEN's simulation quality.

7. Illustrative examples

To complement the simulation validation presented in the previous section, three illustrative examples to show typical abilities of ECOGEN are presented:

1. 2D droplet impact on a wall to show its ability to manage surface tension and gravity.
2. 2D-axisymmetric, shock-induced bubble collapse near a wall to show its ability to reproduce complex physics while using AMR method and mesh stretching.
3. 3D pump rotor to show its ability to deal with complex geometries using an unstructured mesh and rotating frames.

7.1. 2D droplet impact on a wall

A 2D droplet of liquid is introduced in a box filled with air. At the initialization, thermodynamical equilibrium between the liquid and the gas is present and the droplet is moving with an oblique direction to the upper wall (Fig. 18). Wall boundary conditions are used. Surface tension is taken into account which explains the higher pressure inside the droplet. Gravity effects are also present to force the droplet to drop down after the impact

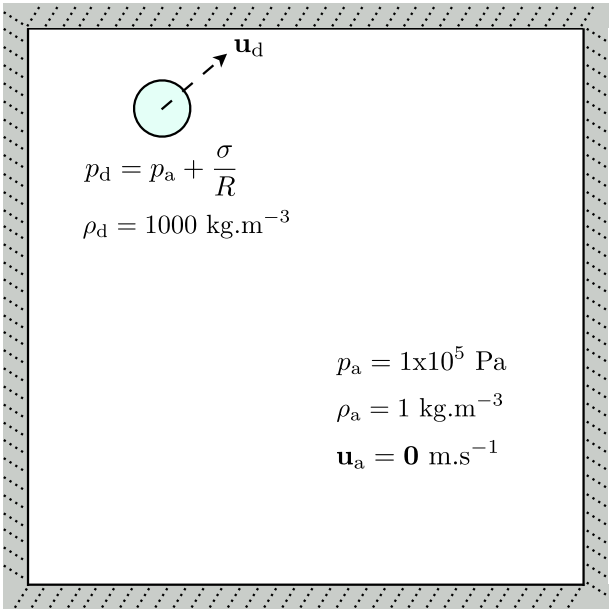

Fig. 18. Initial configuration of the test case of the 2D droplet impact on a wall.

on the upper wall. Initially, the Weber and Bond numbers are the following:

$$We = \frac{\rho_\mathrm{d}\|\mathbf{u}_\mathrm{d}\|D}{\sigma} = 31.78, \qquad Bo = \frac{(\rho_\mathrm{d} - \rho_\mathrm{a})\, aD^2}{\sigma} = 2.737,$$

where ρ_d, $\mathbf{u}_\mathrm{d}$, D, ρ_a, σ and a are the density, the velocity and the diameter of the liquid droplet, the density of the air, the surface-tension coefficient and the magnitude of the gravity acceleration, respectively.

The hyperbolic model of Schmidmayer et al. [27], taking into account surface tension, is used with additional gravity effects:

$$\begin{cases} \dfrac{\partial \alpha_1}{\partial t} + \mathbf{u}\cdot\nabla\alpha_1 & = K\nabla\cdot\mathbf{u}, \\ \dfrac{\partial \alpha_1\rho_1}{\partial t} + \nabla\cdot(\alpha_1\rho_1\mathbf{u}) & = 0, \\ \dfrac{\partial \alpha_2\rho_2}{\partial t} + \nabla\cdot(\alpha_2\rho_2\mathbf{u}) & = 0, \\ \dfrac{\partial \rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u} + p\mathbf{I} + \boldsymbol{\Omega}) & = \rho\mathbf{a}, \\ \dfrac{\partial \rho E}{\partial t} + \nabla\cdot((\rho E + p)\,\mathbf{u} + \boldsymbol{\Omega}\cdot\mathbf{u}) & = \rho\mathbf{a}\cdot\mathbf{u}, \\ \dfrac{\partial c}{\partial t} + \mathbf{u}\cdot\nabla c & = 0, \end{cases} \tag{7}$$

Here, $E = e + \frac{1}{2}\|\mathbf{u}\|^2 + \frac{\varepsilon_\sigma}{\rho}$ and $\boldsymbol{\Omega}$ is the capillary tensor given by:

$$\boldsymbol{\Omega} = -\sigma\left(\|\nabla c\|\mathbf{I} - \frac{\nabla c\otimes\nabla c}{\|\nabla c\|}\right).$$

The capillary energy is equal to $\varepsilon_\sigma = \sigma\|\nabla c\|$ and c is the color function. $\mathbf{a}$ is the gravitational acceleration.

The surface tension is treated with a splitting methodology and is thus considered as additional physics, while the gravity is considered as source terms. Note that a relaxation procedure is also used to solve the model. Complete details are available in [27]. The stiffened-gas parameters for the liquid droplet are $\gamma_\mathrm{d} = 2.1$ and $\pi_{\infty,\mathrm{w}} = 10^6$ Pa [26,27], and the ideal-gas parameter for the air is $\gamma_\mathrm{b} = 1.4$. AMR is used and the effective resolution

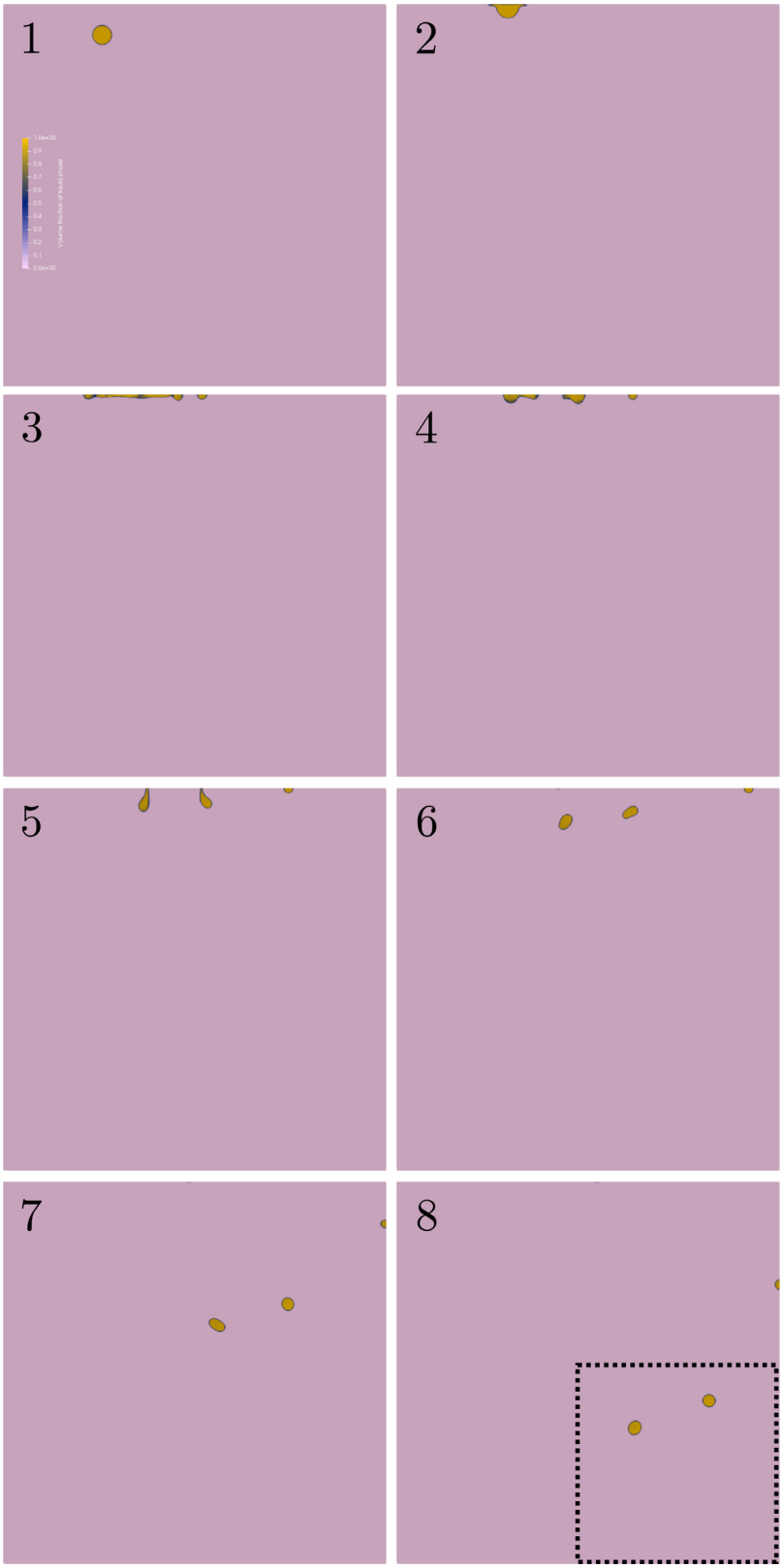

Fig. 19. Results at different typical times for the test case of the 2D droplet impact on a wall. The impact on the upper wall and the drop of the subsequent droplets are shown.

at the vicinity of the interface is of 32 cells per initial droplet diameter.

Results at different typical times are presented in Figs. 19 and 20. In the first figure is shown the complete domain with the droplet impacting the upper wall, spreading over it, forming three smaller, spread droplets and then with the effect of surface tension, the spread droplets are gathered and finally have enough mass and inertia to drop down from the upper wall under the

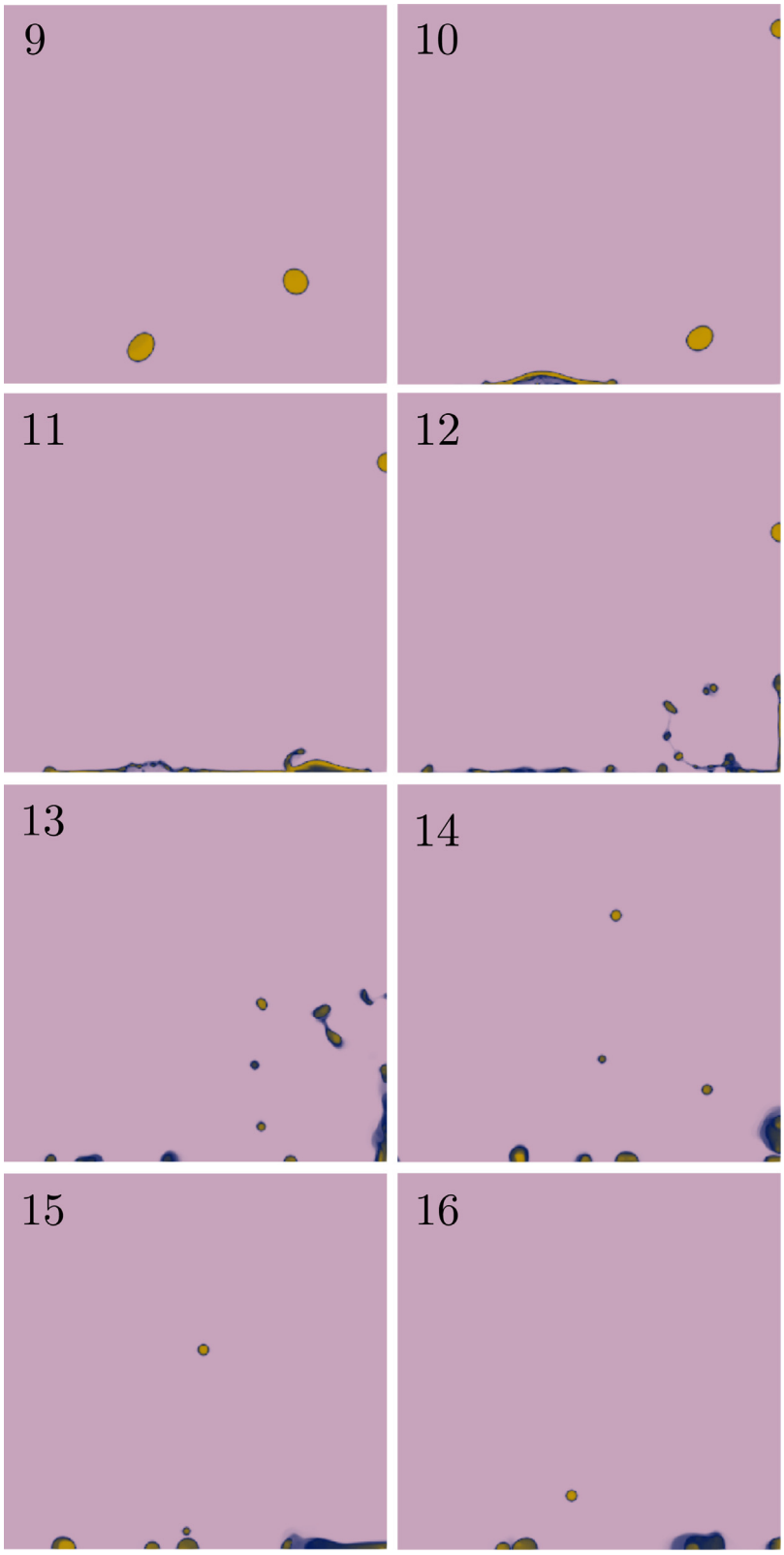

Fig. 20. Results at different typical times for the test case of the 2D droplet impact on a wall. A magnified view, corresponding to the region of the black, dotted box in the last picture of Fig. 19, is shown with the impact of the droplets on the lower wall.

effect of gravity. The second figure shows a magnified part of the complete domain (bottom, right quarter of the domain) and its first frame (frame 9) shows the two biggest droplets just before they touch the lower wall. This time, the stronger impacts enhance the formation of multiple, even smaller droplets. In frame 10, one can observe the third, smallest droplet flowing downward along the right wall appearing into the magnified domain, as well as a cavity pocket formed under the first droplet impacting the lower wall. In frame 12, detached droplets from the second droplet impacting the lower wall are flowing upward along the

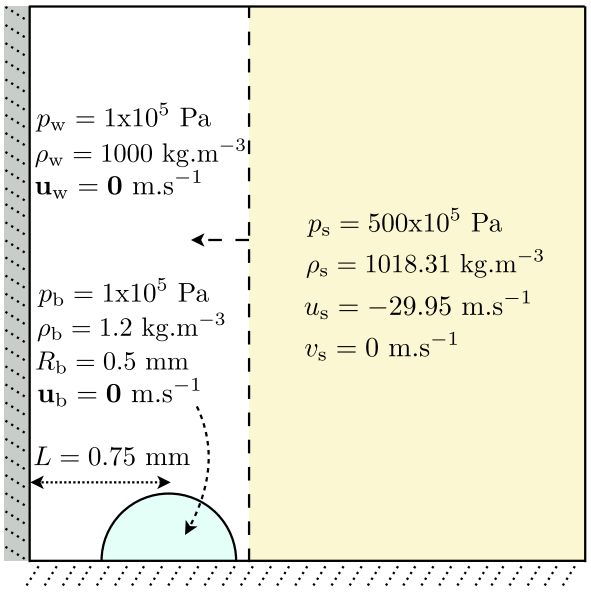

Fig. 21. Initial configuration of the test case of the shock-induced bubble collapse near a wall.

right wall. Between frames 12 and 13, the droplets on the right wall are impacting each other. In frame 14, the highest droplet is at its highest elevation and is thus falling down in the subsequent frames. The smallest droplet in frame 15 is going downward and is going to be absorbed by the droplet directly under it (frame 16).

7.2. 2D-axisymmetric, shock-induced bubble collapse near a wall

A spherical bubble collapse in a 2D-axisymmetric configuration and under strong pressure ratio was presented in Section 6.1 for comparison against the semi-analytic solution of Keller–Miksis [80]. Here, a similar configuration is considered but where the bubble collapses near a wall in a non-spherical manner.

The spherical air bubble is still initialized at atmospheric state but it is placed at a standoff distance (ratio between the distance from the center of the bubble to the wall and the bubble radius) of 1.5 from the wall. The rest of the 2D-axisymmetric domain is separated in two parts: One is filled with water also at atmospheric pressure allowing thermodynamical equilibrium with the initial bubble, and the other one is filled with shocked water at high pressure of $p_s = 500 \times 10^5$ Pa, density of $\rho_s = 1018.31$ kg m^{-3} and velocities of $u_s = -29.95$ m s^{-1} and $v_s = 0$ m s^{-1}, representing the conditions of a shock propagating parallelly to the wall (Fig. 21). The domain size is 6 mm x 6 mm. The initial radius of the bubble is $R_b = 0.5$ mm. The bottom boundary implies symmetry condition while the left one represents a wall condition. Non-reflecting boundary conditions are used elsewhere.

The model used is the same than for the spherical-bubble-collapse test case of Section 6.1. The mesh initially presents, in each direction, 100 cells between 0 mm and 2 mm and then it is stretched over 32 cells, with an aspect ratio of 1.1 between each cell, until it reaches 6 mm. AMR is used with 4 levels of refinement and thus allows to obtain an effective resolution of 400 cells per initial bubble radius. An example of the mesh at the initialization and at time $t = 0.5125$ μs is shown in Fig. 22 for a simulation using 24 cores. We recall that in addition to the AMR, the THINC interface-sharpening method is used and that within the version 1.0 of ECOGEN, AMR parallel balancing is not present. The latter is going to be available within the next public version.

In Fig. 23 are shown results of the simulation for different times. On the upper half of each picture is shown the pressure field, while the lower half shows the velocity magnitude. Logarithmic color bars are used to allow visualization of the different physical phenomena appearing at different scales. The pressure and velocity fields are overlaid by the volume fraction of gas shown in black and white, with an opacity function to render translucent surfaces. The black shows regions of high gas content. The opaqueness decreases with volume fraction until the gas volume fraction is zero (100% liquid) depicted as 100% transparent.

One can observe the initial shock propagating from the right to the left and crossing the bubble. The shock is then reflected on the wall to cross another time the bubble. The bubble progressively collapses in a non-spherical manner. A jet to the wall is produced by the bubble, accompanied by a shock wave of locally stronger amplitude than the initial one.

7.3. 3D pump rotor with cavitation

A 3D pump rotor is simulated using the simplified geometry presented in Fig. 24. A liquid is entering from an annular inflow zone represented by a red color in (b) and considered as a tank boundary condition in the simulation. The pump rotor is equipped with 16 blades and is rotating at 1500 rpm around Z-axis. The fluid is exiting the pump rotor from an annular outflow zone at imposed pressure represented by a white color. The pressure ratio between outlet and tank is $r_p = 0.25$.

The model solved here is similar to those solved in Petitpas et al. [19] in the limit of a two-phase flow model taking into account the effect of phase change. As a rotating flow is simulated, the model is written in the rotating reference frame with a constant angular velocity $\boldsymbol{\omega}$ corresponding to the rotor angular velocity. Thus, the model contains the centrifugal and Coriolis accelerations:

$$\begin{cases} \dfrac{\partial \alpha_1}{\partial t} + \mathbf{u}\cdot\nabla\alpha_1 & = K\nabla\cdot\mathbf{u} + K_{th}H(T_2 - T_1) \\ & \quad + K_{ch}\nu(\varepsilon_2 - \varepsilon_1), \\ \dfrac{\partial \alpha_1\rho_1}{\partial t} + \nabla\cdot(\alpha_1\rho_1\mathbf{u}) & = \rho\nu(\varepsilon_2 - \varepsilon_1), \\ \dfrac{\partial \alpha_2\rho_2}{\partial t} + \nabla\cdot(\alpha_2\rho_2\mathbf{u}) & = -\rho\nu(\varepsilon_2 - \varepsilon_1), \\ \dfrac{\partial \rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u} + p\mathbf{I}) & = -2\rho\boldsymbol{\omega}\wedge\mathbf{u} - \rho\boldsymbol{\omega}\wedge\boldsymbol{\omega}\wedge\mathbf{x}, \\ \dfrac{\partial \rho E}{\partial t} + \nabla\cdot((\rho E + p)\,\mathbf{u}) & = -\mathbf{u}\cdot(\rho\boldsymbol{\omega}\wedge\boldsymbol{\omega}\wedge\mathbf{x}), \end{cases} \tag{8}$$

where

$$K_{th} = \frac{\alpha_1\alpha_2}{\alpha_1\rho_1 s_1^2 + \alpha_2\rho_2 s_2^2}\left(\frac{\Gamma_1}{\alpha_1} + \frac{\Gamma_2}{\alpha_2}\right),$$

$$K_{ch} = \rho\frac{\frac{\rho_1 s_1^2}{\alpha_1} + \frac{\rho_2 s_2^2}{\alpha_2}}{\frac{s_1^2}{\alpha_1} + \frac{s_2^2}{\alpha_2}}.$$

T_k, Γ_k and ε_k are the temperature, the Grüneisen coefficient and the Gibbs free energy of phase k, respectively. H and ν are relaxation coefficients for heat and mass transfers. This complex model involves solving equations under conservative form and then complementing them with non-conservative, source and relaxation terms as described in Section 5. The parameters of the stiffened-gas liquid and ideal-gas vapor are presented in Table 1 [19].

Simulation results are presented in Fig. 25 when a pseudo steady state is obtained. In (a) and (b) are shown velocity vectors

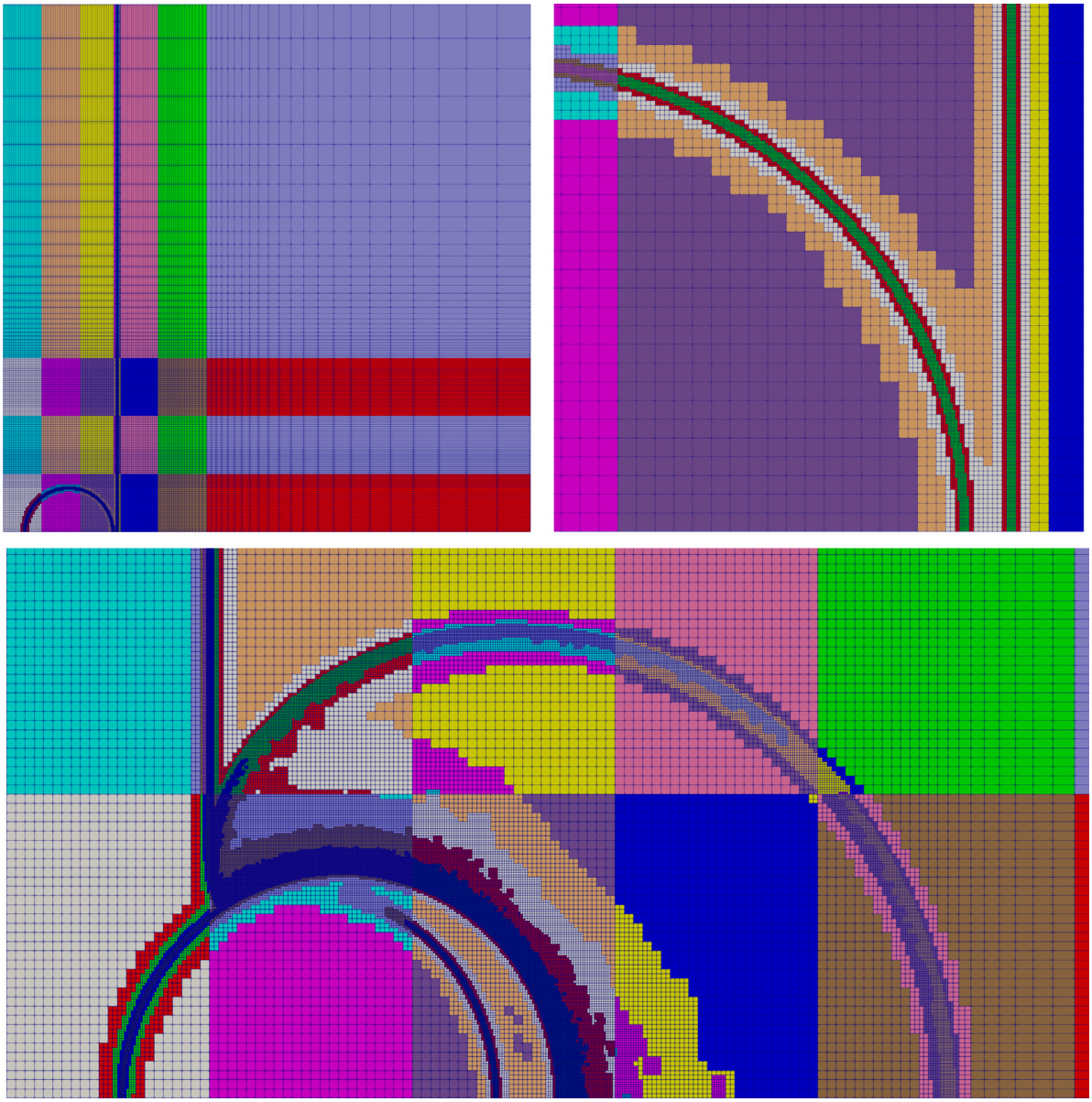

Fig. 22. Meshes for the test case of the shock-induced bubble collapse near a wall. Top, left picture shows the full domain at the initialization with the apparent stretched mesh. Top, right picture shows a magnified view of the initial mesh at the location of the bubble and of the shock. Bottom picture shows a magnified view around the bubble at time $t = 0.5125$ µs. In the last two pictures, different levels of refinement are observable. Colors either change for cores or levels of refinement. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Equation-of-state parameters for the 3D pump rotor test.

EOS parameters	γ	π_∞ (Pa)	c_v (J/(kg K))	e_{ref} (J/kg)	S_{ref} (J/kg)
Liquid	2.45	1.06×10^8	695	-2.8×10^3	0
Vapor	1.47	0	530	6.9×10^3	-9.2×10^3

colored by the magnitude of the mixture velocity to observe the rotational phenomenon and colored by the pressure, respectively. While the volume fraction of vapor is represented in (c) where one observes that decreasing pressure produces cavitation pockets filled with vapor along blades.

8. Code performance

ECOGEN's performance is tested regarding parallel scaling for AMR and non-AMR methods and single-node, time-to-solution compared to another recent open-source code, here MFC [54].

8.1. Parallel performance

The parallel scaling is presented herein through the strong and weak speedups:

$$\text{Speedup}_{\text{strong}} = \frac{T_1}{T_k} C_1, \tag{9}$$

$$\text{Speedup}_{\text{weak}} = \frac{T_1}{T_k} C_k, \tag{10}$$

where T is the computational wall time of the total simulation and C corresponds to the number of cores. Subscripts 1 and k refer to the base and the current simulations, respectively. Here, the base simulation uses a single node of 24 cores ($C_1 = 24$). The strong scaling uses the same total number of cells for each case, while the weak scaling adapts this number to correspond to an ideal constant time-to-solution, *e.g.* 1000 cells for 1 node and 2000 for 2. One can note that performing a weak scaling

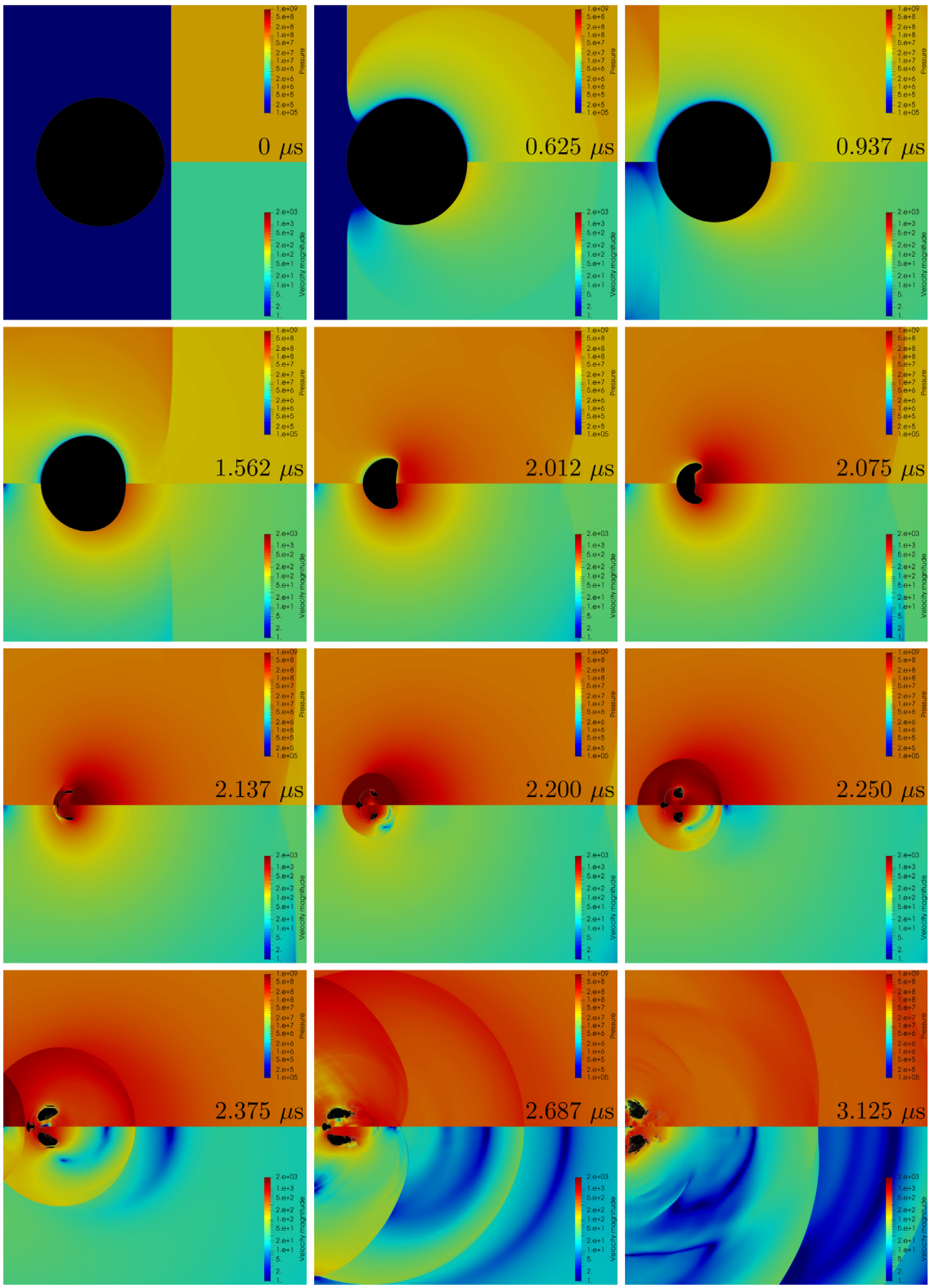

Fig. 23. Results at different times for the test case of the shock-induced bubble collapse near a wall.

configuration when using AMR is challenging. Therefore, only the strong configuration for AMR is presented in the following.

Fig. 26 shows ECOGEN's scaling in four different configurations, all starting with a single node. The non-AMR scaling (Cartesian) was performed on the Comet supercomputer from the San Diego Supercomputer Center for both the weak and strong configurations and on a small portion of the 3D, water-droplet-breakup test case of Section 6.2. The AMR scaling was also performed on Comet but on a 3D, bubble-dynamics test case involving a shock-wave interaction with three successive gas bubbles [26]. The latter was done for two AMR configurations: One with a maximum refinement level of $l_{\max} = 2$ and another with $l_{\max} = 3$. The non-AMR method shows for both configurations good scaling performances: Close to perfectly linear until about a 1000 cores

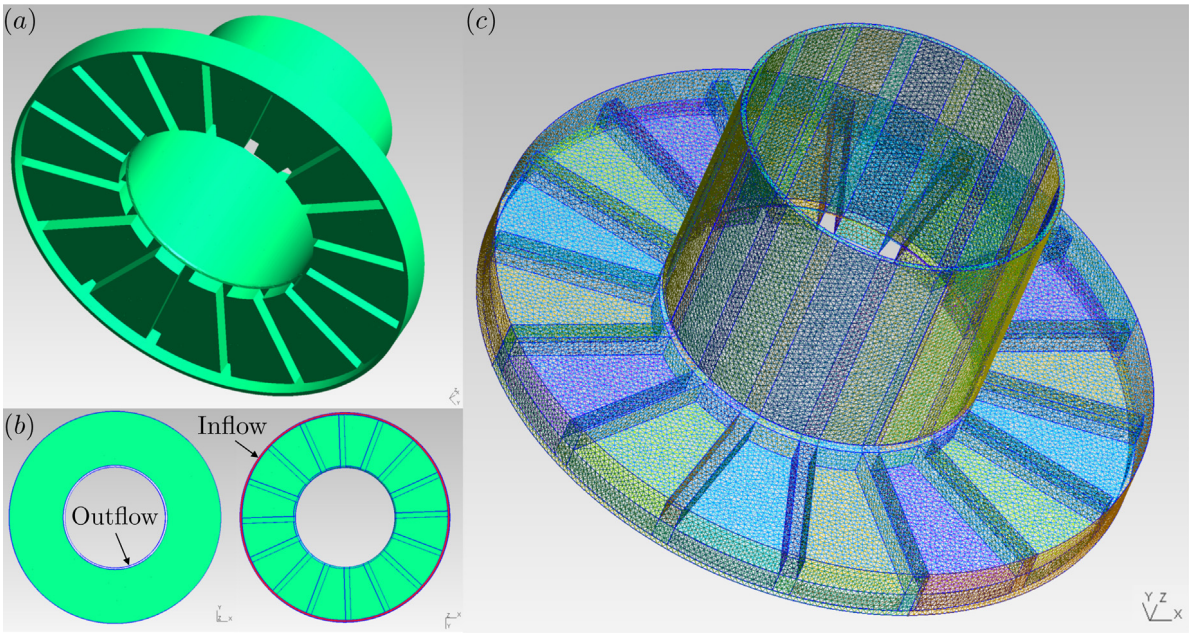

Fig. 24. Views of the 3D pump rotor geometry. White and red annular zones in (b) represent fluid outflow and inflow, respectively. The mesh surfaces are visible in (c). It is composed with 227,062 tetrahedrons and 96,172 triangles. Mesh colors represent the core decomposition (24 cores here). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

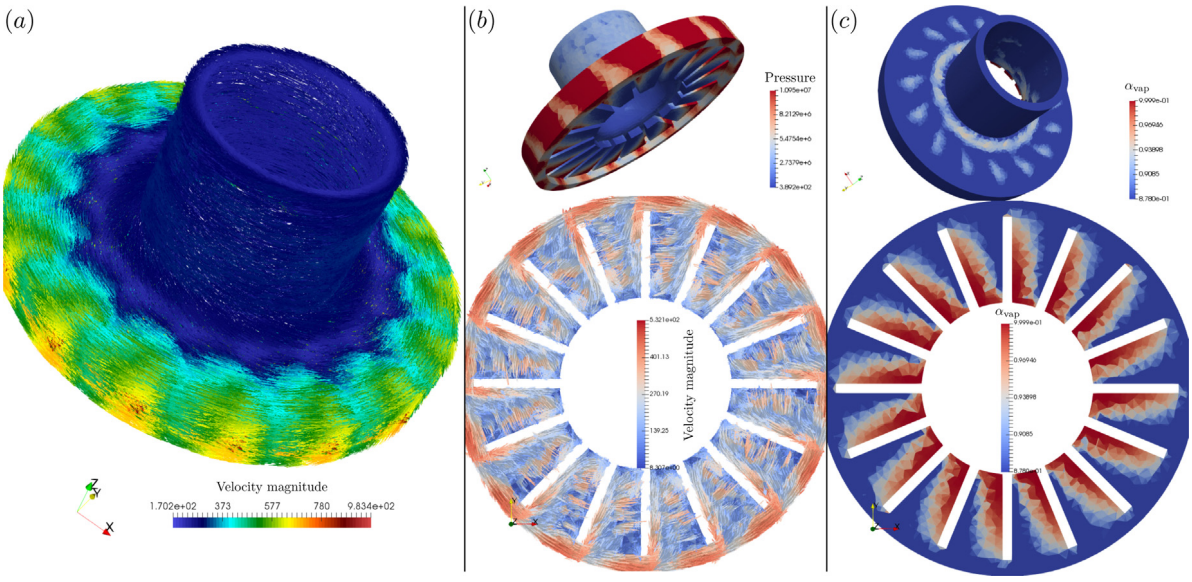

Fig. 25. Simulation results at pseudo steady state for the 3D pump rotor geometry (1500 rpm). Velocity vectors colored by velocity magnitude (a) and pressure (b), and field of vapor volume fraction (c) are represented. In the latter, cavitation pockets are observed.

where the performance decreases of a few percent from the ideal solution. Concerning the AMR method, one can note that even if adaptive parallel load balancing is not present in the code yet, the algorithm is performing reasonably well and the trend seems to stay approximately constant until about a few hundred cores. The speedup gives slightly better results for the cases with $l_{\max} = 2$ than the ones with $l_{\max} = 3$. On average, the speedup factor when increasing the number of cores by two is 1.76 and 1.72 for $l_{\max} = 2$ and 3, respectively, instead of an ideal factor of 2. The latter and the fact that the present AMR version of the code does not perform well for a high number of cores are explained by the lack of the previously mentioned adaptive parallel load balancing. Improved speedup for AMR is thus expecting in future versions of the code.

8.2. Single-node performance

ECOGEN's time-to-solutions on a single node for 3D, spherical-bubble-collapse test cases with a pressure ratio of $p_\infty/p_b = 10$ and 1427 are compared to the ones performed by MFC with its third- and fifth-order WENO methods. Initial conditions and simulation details are specified in [55]. Non-AMR (Cartesian) method is used for ECOGEN. For this comparison, the simulations are performed on a personal cluster where a single node also has 24 cores.

Table 2 shows the different time-to-solutions and we observe for the lower pressure ratio a factor 2.10 and 6.19 of better performance in favor of ECOGEN in comparison to MFC's third- and fifth-order WENO, respectively. In the case of the higher pressure

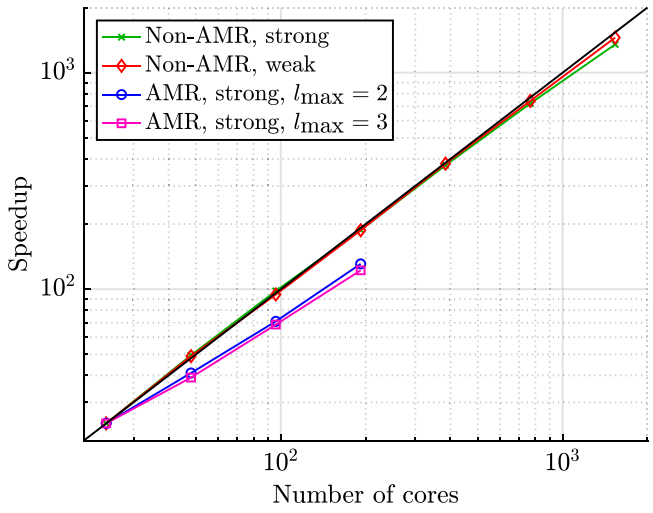

Fig. 26. Speedup function of the number of cores. Four configurations are presented: Two non-AMR (strong and weak scaling) and two AMR ($l_{\max} = 2$ and $l_{\max} = 3$, both strong scaling).

Table 2
ECOGEN's and MFC's single-node time-to-solutions for two test cases.

Pressure ratio	Simulation's time-to-solution (h)		
	2nd-order ECOGEN	3rd-order MFC	5th-order MFC
$p_\infty/p_b = 10$	11.93	25.09	73.84
$p_\infty/p_b = 1427$	11.36	61.70	147.53

ratio, ECOGEN's relative performance gives even better results with a factor 5.43 and 12.99, respectively. Those differences are mainly explained by the fact that the codes are using different methods (MUSCL and WENO) using different stencils and different CFL restrictions to maintain stability. As additional information, one should note that the order of accuracy of the method may not matter as much as expected for problems involving interfaces where first-order is recovered [55].

ECOGEN's good performance are then demonstrated. One should note that these performance tests are test-case dependent. However, the trend would be respected. Furthermore, considering last developments, one should expect noteworthy performance improvements in the next release version of the code.

9. Conclusion

The new open-source, computational-fluid-dynamics code ECOGEN has been presented. It is a multi-model tool devoted to the simulation of multiphase, compressible, multiphysics flows and the code is suited for strongly unsteady flows. The numerical solver of ECOGEN is implemented in a flexible structure making the code able to compute such complex flows on different kinds of discretization grids (AMR, stretched and unstructured). Its numerical implementation has been presented in detail to help the enthusiastic developer to contribute to this open-source project. Validation, parallel scalability and representative test cases have been presented to show the tool abilities and to open the gate to future developments.

10. Metadata

The essential metadata of ECOGEN are summarized in Table 3.

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.

Acknowledgments

The authors would like to thank Prof. Tim Colonius for fruitful discussions. This work has been partially supported by the French spatial agency CNES (Centre National d'Etudes Spatiales) under Grant Number MNS-CT-1459000 and by the Office of Naval Research under Grant Number N0014-18-1-2625. Computations associated to parallel performance have partially utilized the Extreme Science and Engineering Discovery Environment, which is supported by the National Science Foundation grant number CTS120005.

Appendix. Physical-relaxation methods

Pressure relaxation

The procedure to constrain the pressures in a n-pressure model follows the lines of Petitpas et al. [20] and can be schematized as in Fig. 27.

During the relaxation procedure, the following system is considered:

$$\begin{aligned} &\forall k, \ \frac{\partial \alpha_k}{\partial t} &&= \omega(p_k - p_*), \\ &\forall k, \ \frac{\partial \alpha_k \rho_k}{\partial t} &&= 0, \\ &\forall k, \ \frac{\partial \alpha_k \rho_k e_k}{\partial t} &&= -p_I \omega(p_k - p_*), \\ &\frac{\partial \rho e}{\partial t} &&= 0, \end{aligned} \tag{11}$$

where e_k represents the internal energy of phase k. The mixture variables ρ and e are both constant during the relaxation procedure. ω is a relaxation parameter and p_I represents an interfacial pressure. The pressure p^* should be seen as a common target pressure. Those terms do not need any explicit closure relations since the present relaxation procedure will lead to an equilibrium pressure state for all phases. More sophisticated relaxation models can be found in literature (*e.g.* [77]), but lead to solve the same relaxation procedure. Solving System (11) in the limit $\omega \to \infty$ is equivalent to look for the solution of the following algebraic system where superscript "0" stands for the initial disequilibrium state and "f" for the final equilibrium state:

$$\begin{aligned} &\forall k, \ e_k^f(p^f, \rho_k^f) - e_k^0(p_k^0, \rho_k^0) + p^f(\rho_k^f - \rho_k^0) = 0, \\ &\sum_k \frac{(\alpha\rho)_k^0}{\rho_k^f} = 1. \end{aligned} \tag{12}$$

Solving this system to find the final equilibrium state "f" is strongly linked to the choice of the equations of state. In ECOGEN, System (12) is solved in the general case using an iterative Newton–Raphson method. Achievement of this relaxation procedure relies on the possibility to reverse the first equation of System (12). The corresponding function $\rho_k^{\text{relax}_p}(p)$ has to be provided in the corresponding EOS classes. In the released version of ECOGEN, these functions are provided for ideal-gas and stiffened-gas equations of state. Full details of this particular pressure-relaxation procedure are available in [18,20].

Table 3
ECOGEN metadata.

Nr.	ECOGEN metadata description	
C1	Current code version	V1.0
C2	Permanent link to code/repository used of this code version	https://github.com/code-mphi/ECOGEN
C3	Legal Code License	GNU General Public License 3 (http://www.gnu.org/licenses/gpl.txt)
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	C++ and XML
C6	Compilation requirements, operating environments & dependencies	C++ compiler with MPI library
C7	Link to developer documentation/manual	https://code-mphi.github.io/ECOGEN/docs/sphinx_docs/index.html
C8	Support email for questions	ecogen@code-mphi.fr

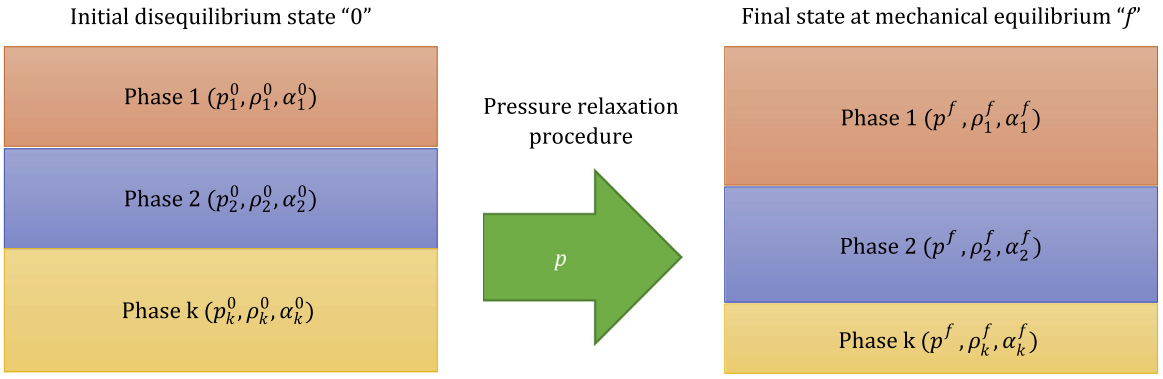

Fig. 27. During the pressure relaxation process, volume occupied by phases in the cell may change.

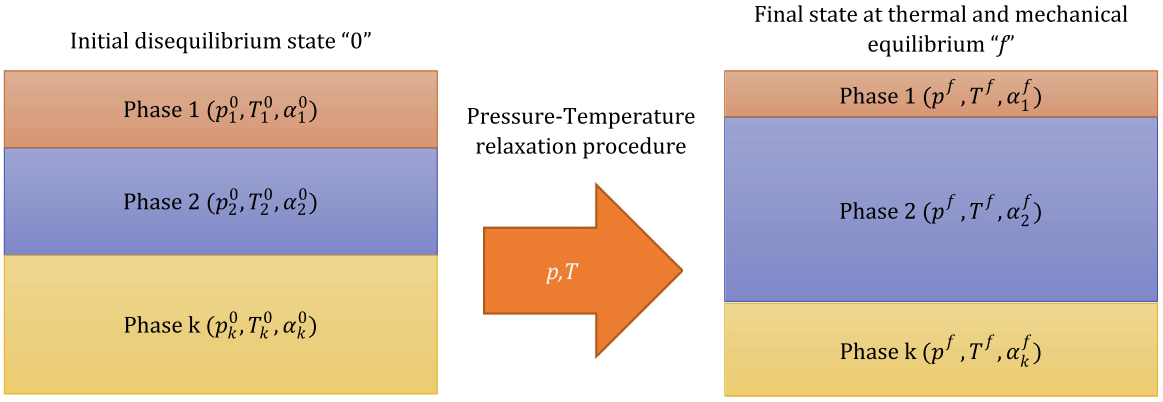

Fig. 28. During the Pressure–Temperature relaxation process, the final state can only be obtain by equilibrium assumptions.

Pressure–Temperature relaxation

In a similar manner than for the pressure relaxation procedure, from an initial disequilibrium state, one can obtain the corresponding Pressure–Temperature equilibrium state as schematized in Fig. 28.

Introducing mass fraction Y_k and specific volume v_k of phase k, conservation law during the relaxation procedure for the isolated system imposes that $v = \sum_k Y_k v_k$ and $e = \sum_k Y_k e_k$ remain constant. The following algebraic system of equations is thus solved:

$$\begin{aligned} &\sum_k Y_k^0 v_k^f(p^f, T^f) = v^0, \\ &\sum_k Y_k^0 e_k^f(p^f, T^f) = e^0, \end{aligned} \tag{13}$$

with unknowns p^f and T^f. Solving System (13) is also strongly linked to the choice of the equations of state.

Pressure–Temperature–Chemical potential relaxation

Relaxation to Pressure–Temperature–Chemical potential holds for a liquid–vapor mixture in equilibrium. From an initial disequilibrium state, one can obtain the corresponding Pressure–Temperature–Chemical potential equilibrium state as schematized in Fig. 29.

Equilibrium assumptions coupled with conservation constraints of mass and energy for the isolated system (cell) lead to the following algebraic system:

$$\begin{aligned} &\varepsilon_l^f(p^f, T^f) = \varepsilon_v^0(p^0, T^0), \\ &Y_l^f v_l^f(p^f, T^f) + (1 - Y_l^f) v_v^f(p^f, T^f) = v^0, \\ &Y_l^f e_l^f(p^f, T^f) + (1 - Y_l^f) e_v^f(p^f, T^f) = e^0, \end{aligned} \tag{14}$$

with unknowns p^f, T^f and Y_l^f. The first equation of System (14) together with expressions of the corresponding equations of state for liquid and vapor phases leads to the link between saturated pressure and temperature $T(p) = T_{sat}(p_{sat})$. Solving System (14) is also strongly linked to the choice of the equations of state. In

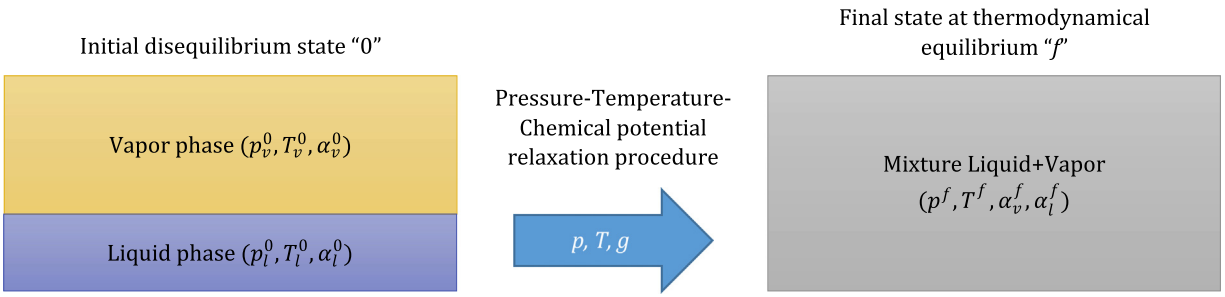

Fig. 29. During the Pressure–Temperature–Chemical potential relaxation process, the final state is a mixture of liquid and its vapor at thermodynamical equilibrium.

the released version of ECOGEN, the method is implemented for a liquid and its vapor governed by the stiffened-gas EOS and the ideal-gas EOS, respectively. A full description of the method can be found in Le Métayer et al. [77].

References

[1] T.J. Matula, P.R. Hilmo, B.D. Storey, A.J. Szeri, Phys. Fluids 14 (3) (2002) 913–921.
[2] W. Kreider, L.A. Crum, M.R. Bailey, O.A. Sapozhnikov, J. Acoust. Soc. Am. 130 (5) (2011) 3531–3540.
[3] E. Johnsen, T. Colonius, J. Comput. Phys. 219 (2) (2006) 715–732.
[4] E. Johnsen, T. Colonius, J. Fluid Mech. 629 (2009) 231–262.
[5] A. Tiwari, J. Freund, C. Pantano, J. Comput. Phys. 252 (2013) 290–309.
[6] V. Coralic, T. Colonius, Eur. J. Mech. B 40 (2013) 64–74.
[7] V. Coralic, T. Colonius, J. Comput. Phys. 274 (2014) 95–121.
[8] K. Maeda, Simulation, Experiments, and Modeling of Cloud Cavitation with Application to Burst Wave Lithotripsy (Ph.D. thesis), California Institute of Technology, 2018.
[9] S.A. Beig, E. Johnsen, J. Comput. Phys. 302 (2015) 548–566.
[10] Y.A. Pishchalnikov, W.M. Behnke-Parks, K. Schmidmayer, K. Maeda, T. Colonius, T.W. Kenny, D.J. Laser, J. Acoust. Soc. Am. 146 (2019) 516–531.
[11] C.E. Brennen, Interface Focus 5 (5) (2015) 20150022.
[12] K. Laksari, S. Assari, B. Seibold, K. Sadeghipour, K. Darvish, Biomech. Model. Mechanobiol. 14 (3) (2015) 459–472.
[13] W. Proud, T.-T. Nguyen, C. Bo, B. Butler, R. Boddy, A. Williams, S. Masouros, K. Brown, Metall. Mater. Trans. A 46 (10) (2015) 4559–4566.
[14] J.B. Estrada, C. Barajas, D.L. Henann, E. Johnsen, C. Franck, J. Mech. Phys. Solids 112 (2018) 291–317.
[15] J.-C. Veilleux, K. Maeda, T. Colonius, J.E. Shepherd, Int. Symp. Cav. (2018) 10th.
[16] J.-C. Veilleux, Pressure and Stress Transients in Autoinjector Devices (Ph.D. thesis), California Institute of Technology, 2019.
[17] R. Saurel, F. Petitpas, R. Abgrall, J. Fluid Mech. 607 (2008) 313–350.
[18] R. Saurel, F. Petitpas, R. Berry, J. Comput. Phys. 228 (5) (2009) 1678–1712.
[19] F. Petitpas, J. Massoni, R. Saurel, E. Lapebie, L. Munier, Int. J. Multiph. Flow. 35 (8) (2009) 747–759.
[20] F. Petitpas, R. Saurel, B. Ahn, S. Ko, Int. J. Nav. Arch. Ocean Eng. 3 (2011) 263–273.
[21] M. Pelanti, K.-M. Shyue, J. Comput. Phys. 259 (2014) 331–357.
[22] M. Bode, F. vom Lehn, H. Pitsch, Int. Symp. Cav. (2018) 10th.
[23] P. Welch, P. Boyle, GRC Trans. 33 (2009) 765–772.
[24] J. Meng, T. Colonius, Shock Waves (2014) 1–16.
[25] J. Meng, T. Colonius, J. Fluid Mech. 835 (2018) 1108–1135.
[26] K. Schmidmayer, F. Petitpas, E. Daniel, J. Comput. Phys. 388 (2019) 252–278.
[27] K. Schmidmayer, F. Petitpas, E. Daniel, N. Favrie, S. Gavrilyuk, J. Comput. Phys. 334 (2017) 468–496.
[28] B. Devassy, C. Habchi, E. Daniel, Atomization Sprays 25 (1) (2015) 47–80.
[29] O. Engel, J. Res. Natl. Bur. Stand. 60 (3) (1958) 245–280.
[30] D. Joseph, J. Belanger, G. Beavers, Int. J. Multiph. Flow. 25 (6) (1999) 1263–1303.
[31] D. Igra, K. Takayama, Atomization Sprays 11 (2) (2001) 167–185.
[32] D. Igra, K. Takayama, Shock Waves 11 (3) (2001) 219–228.
[33] A. Chauvin, E. Daniel, A. Chinnayya, J. Massoni, G. Jourdan, Shock Waves (2015) 1–13.
[34] A. Chauvin, G. Jourdan, E. Daniel, L. Houas, R. Tosello, Phys. Fluids (1994-present) 23 (11) (2011) 113301.
[35] R. Reitz, F. Bracco, Phys. Fluids 25 (10) (1982) 1730–1742.
[36] Q. Zeng, S.R. Gonzalez-Avila, S. Ten Voorde, C.-D. Ohl, J. Fluid Mech. 846 (2018) 916–943.
[37] Y. Tagawa, N. Oudalov, A. El Ghalbzouri, C. Sun, D. Lohse, Lab Chip 13 (7) (2013) 1357–1363.
[38] Y. Tagawa, N. Oudalov, C.W. Visser, I.R. Peters, D. van der Meer, C. Sun, A. Prosperetti, D. Lohse, Phys. Rev. X 2 (3) (2012) 031002.
[39] S. Gavrilyuk, N. Favrie, R. Saurel, J. Comput. Phys. 227 (2008) 2941–2969.
[40] N. Favrie, S.L. Gavrilyuk, R. Saurel, J. Comput. Phys. 228 (16) (2009) 6037–6077.
[41] S. Ndanou, N. Favrie, S. Gavrilyuk, J. Elast. 115 (1) (2014) 1–25.
[42] S. Ndanou, N. Favrie, S. Gavrilyuk, J. Comput. Phys. 295 (2015) 523–555.
[43] S. Hank, N. Favrie, J. Massoni, J. Comput. Phys. 330 (2017) 65–91.
[44] S. Hank, S. Gavrilyuk, N. Favrie, J. Massoni, Int. J. Impact Eng. 109 (2017) 104–111.
[45] S. Le Martelot, B. Nkonga, R. Saurel, J. Comput. Phys. 255 (2013) 53–82.
[46] S. Le Martelot, R. Saurel, B. Nkonga, Int. J. Multiph. Flow. 66 (2014) 62–78.
[47] A. Chiapolino, P. Boivin, R. Saurel, . Fluids 83 (7) (2017) 583–605.
[48] A. Kapila, R. Menikoff, J. Bdzil, S. Son, D. Stewart, Phys. Fluids 13 (2001) 3002–3024.
[49] F. Petitpas, R. Saurel, E. Franquet, A. Chinnayya, Shock Waves 19 (5) (2009) 377–401.
[50] E. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer Verlag, Berlin, 1997.
[51] M. Dumbser, A. Hidalgo, O. Zanotti, Comput. Methods Appl. Mech. Engrg. 268 (2014) 359–387.
[52] K. Shyue, F. Xiao, J. Comput. Phys. 268 (2014) 326–354.
[53] H. Jasak, A. Jemcov, Z. Tukovic, Int. Workshop Coupled Methods Num. Dyn., Vol. 1000, IUC Dubrovnik Croatia, 2007, pp. 1–20.
[54] S.H. Bryngelson, K. Schmidmayer, V. Coralic, J.C. Meng, K. Maeda, T. Colonius, submitted for publication,
[55] K. Schmidmayer, S.H. Bryngelson, T. Colonius, J. Comput. Phys. 402 (2020).
[56] S. Godunov, Math. Sb. 47 (1959) 357–393.
[57] C. Geuzaine, J.-F. Remacle, Internat. J. Numer. Methods Engrg. 79 (11) (2009) 1309–1331.
[58] R. Saurel, F. Petitpas, Introduction to Diffuse Interfaces and Transformation Fronts Modelling in Compressible Media, in: ESAIM: Proc., vol. 40, EDP Sciences, 2013, pp. 124–143.
[59] N. Thévand, E. Daniel, J. Loraud, Int. J. Numer. Meth. Fluids 31 (4) (1999) 681–702.
[60] G. Périgaud, R. Saurel, J. Comput. Phys. 209 (2005) 139–178.
[61] A.H. Squillacote, J. Ahrens, C. Law, B. Geveci, K. Moreland, B. King, The Paraview Gide, Vol. 366, Kitware Clifton Park, NY, 2007.
[62] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, G.H. Weber, H. Krishnan, et al., High Perf. Visua.: Enabling Ext.-Scale Sc. Insight, 2012, pp. 357–372.
[63] F. Petitpas, E. Franquet, R. Saurel, O. Le Métayer, J. Comput. Phys. 225 (2) (2007) 2214–2248.
[64] F.H. Harlow, A.A. Amsden, J. Comput. Phys. 3 (1) (1968) 80–93.
[65] O. Le Métayer, J. Massoni, R. Saurel, Int. J. Therm. Sci. 43 (2004) 265–276.
[66] A. Khokhlov, J. Comput. Phys. 143 (2) (1998) 519–543.
[67] P.K. Sweby, SIAM J. Numer. Anal. 21 (5) (1984) 995–1011.
[68] B. Van Leer, J. Comput. Phys. 14 (4) (1974) 361–370.
[69] G. Van Albada, B. Van Leer, W. Roberts, Upwind and High-Res. Schemes, Springer, 1997, pp. 95–103.
[70] C.F. Gammie, J.C. McKinney, G. Tóth, Astrophys. J. 589 (1) (2003) 444.
[71] P.L. Roe, Annu. Rev. Fluid Mech. 18 (1) (1986) 337–365.
[72] Z. Sun, S. Inaba, F. Xiao, J. Comput. Phys. 322 (2016) 309–325.
[73] S. Gavrilyuk, K. Ivanova, N. Favrie, J. Comput. Phys. 366 (2018) 252–280.
[74] K. Schmidmayer, F. Petitpas, E. Daniel, 46th AIAA Fluid Dyn. Conf., 2016, p. 4269.
[75] E. Han, M. Hantke, S. Müller, J. Comput. Phys. 338 (2017) 217–239.
[76] M.-H. Lallemand, A. Chinnayya, O. Le Métayer, Int. J. Numer. Meth. Fluids 49 (1) (2005) 1–56.
[77] O. Le Métayer, J. Massoni, R. Saurel, Dynamic Relaxation Processes in Compressible Multiphase Flows. Application to Evaporation Phenomena, in: ESAIM: Proc., vol. 40, EDP Sciences, 2013, pp. 103–123.

[78] A. Zein, M. Hantke, G. Warnecke, J. Comput. Phys. 229 (8) (2010) 2964–2998.
[79] K. Schmidmayer, Simulation De L'atomisation D'une Goutte Par Un Écoulement À Grande Vitesse (Ph.D. thesis), Aix-Marseille, 2017.
[80] J.B. Keller, M. Miksis, J. Acoust. Soc. Am. 68 (2) (1980) 628–633.
[81] A. Murrone, H. Guillard, J. Comput. Phys. 202 (2005) 664–698.
[82] D. Fuster, C. Dopazo, G. Hauke, J. Acoust. Soc. Am. 129 (1) (2011) 122–131.
[83] C.E. Brennen, Cavitation and Bubble Dynamics, Oxford University Press, 1995.
[84] K. Schmidmayer, A. Marty, F. Petitpas, E. Daniel, Int. Conf. Appl. Aero. (2018) 1.