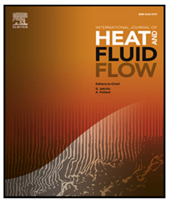
Contents lists available at ScienceDirect

International Journal of Heat and Fluid Flow

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



Conservative Immersed Boundary Methods on Cartesian grids for inviscid compressible flows simulation

El Hadji Abdou Aziz Ndiaye [a,*], Jean-Yves Trépanier [a], Renan De Holanda Sousa [b], Sébastien Leclaire [a]

[a] *Department of Mechanical Engineering, Polytechnique Montréal, 2500 Chemin de Polytechnique, Montréal, H3T 1J4, Québec, Canada*
[b] *Grid Solutions, GE Vernova, 21 Rue Cyprian, Villeurbanne, 69100, France*

ARTICLE INFO

Keywords:
Conservation
Immersed Boundary Method
Cut-cells
Cartesian grid
Compressible flow

ABSTRACT

This work introduces three conservative methods based on the Immersed Boundary Method. These methods make use of cut-cells to ensure the conservation properties in the numerical solution. However, since some cut-cells can be very small, they can significantly restrict the time step of an explicit time integration scheme. To circumvent this limitation, a semi-implicit treatment of the small cells is employed. The first method relies on a straightforward flux redistribution procedure that globally restores conservation on the cut-cells grid. The other two methods employ the local conservative discretization form of the finite volume method, along with optimization procedures, to ensure local conservation of the numerical solution within each cell. These methods have been tested on two-dimensional inviscid compressible flow problems, demonstrating results comparable to those obtained with the standard Cut-Cells method in terms of accuracy and conservation. Furthermore, the methods are stable and can be effectively used with an explicit time integration scheme without encountering any stability issues related to the small cut-cells.

1. Introduction

Multi-physics simulations of compressible flows involving complex geometries and moving bodies present significant challenges. A critical step in developing simulation codes for such applications is mesh generation. Over the years, several approaches have been proposed to address these challenges, including unstructured grids, overset grids, and Cartesian grids. Unstructured grids are highly flexible and can accommodate complex geometries. However, generating high-quality unstructured grids can be difficult, and they often require remeshing for problems involving moving boundaries. The overset grids approach can circumvent the need for remeshing in moving boundary problems, but it necessitates the generation of the basic grids that will be superimposed to form the overset grids, which can be time-consuming. Additionally, the data structures required to couple these superimposed grids can be complex. On the other hand, Cartesian grids are very easy to generate and can be used for moving boundaries. However, special treatment is necessary for the discretization of cells near the boundaries to account for their presence. In the context of moving boundaries, Cartesian grids can be directly used as a background grid. Furthermore, due to their simple structure, Cartesian grids can be easily coupled with other Cartesian grids to form a set of overset grids. This work will focus on the Cartesian grids approach due to their simplicity and greater potential for automation.

When using a Cartesian grid, one encounters an initial problem: the cells near the boundaries are typically not aligned with the boundaries. To address this issue, the boundary cells can be approximated using a staircase representation of the actual boundary. However, this approach is not very accurate and requires a very fine mesh to achieve a good approximation of the boundary. An alternative approach is the Cut-Cells (CC) method (Clarke et al., 1986; Berger, 2017), which involves clipping the cells intersected by the boundary, resulting in a piecewise linear approximation of the boundary. The discretization on the cut-cells grid is straightforward by using the usual finite volume method (FVM) for unstructured grids. The CC method has the advantage of being fully conservative, but it has two main disadvantages. The first disadvantage is that generating the cut-cells grid in three dimensions is challenging, with high computational costs and potential robustness issues. Additionally, for problems involving moving boundaries, the list of cut-cells must be recalculated at each time step, exacerbating these robustness issues. The second, and perhaps most significant, disadvantage is that the geometry in the CC method can arbitrarily cut the grid cells, leading to the formation of very small cells. These small cells

* Corresponding author.
E-mail address: aziz.ndiaye@polymtl.ca (E.H.A.A. Ndiaye).

https://doi.org/10.1016/j.ijheatfluidflow.2025.109775
Received 3 October 2024; Received in revised form 5 February 2025; Accepted 6 February 2025
Available online 27 February 2025

Nomenclature

A_f	Area of face f	γ	Ratio of specific heats
E	Total energy	ρ	Density
$\mathbf{F}$	Flux vector	ω	Weights for flux redistribution
$\mathbf{F}_f$	Numerical fluxes at face f	Ω	Computational domain
h	Average cell size		
$\mathbf{n}$	Normal vector	*Abbreviations*	
n_x, n_y	Components of the normal vector	CC	Cut-Cells
p	Pressure	FCM	Flux Correction Method
$S(i)$	Stencil of neighbors of cell i	FIM	Flux Interpolation Method
u, v	Velocity components	FRM	Flux Redistribution Method
$\mathbf{U}$	Vector of conservative variables	FVM	Finite Volume Method
$\mathbf{v}$	Velocity vector	IBM	Immersed Boundary Method
		RCCM	Reconstructed Cut-Cells Method
Greek letters		`BC-Cell`	Boundary cut-cell
α_i	Cut-cell volume fraction of cell i	`BI-Cell`	Boundary inside cell
$\partial\Omega$	Boundary of the computational domain	`C-Cell`	Conserved cell
∂V_i	Set of faces of cell i	`I-Cell`	Inside cell
ΔM_i	Conserved Mass Error at cell i	`N-Cell`	Neighbor (of a reconstructed) cell
Δt^n	Time step at the nth time iteration	`NR-Cell`	Non-reconstructed cell
ΔV_i	Volume of cell i	`OC-Cell`	Outer cut-cell
$\epsilon_{L_1}, \epsilon_{L_2}, \epsilon_{L_\infty}$	L_1, L_2, and L_∞ error norms	`R-Cell`	Reconstructed cell

require a very small time step to satisfy the CFL condition of explicit time integration schemes, significantly increasing the computational time of the simulation. This issue is known in the literature as the small cell problem, and various methods have been devised to address it.

One of the earliest techniques proposed to handle the small cell problem is the cell-merging method (Ye et al., 1999; Udaykumar et al., 2001), which involves merging small cells with their larger neighbors. While this method is conceptually simple, it is challenging to implement robustly in three dimensions. Specifically, the selection of neighboring cells for merging is often not systematic (Kirkpatrick et al., 2003). A variation of this approach is the cell-linking method (Kirkpatrick et al., 2003; Hartmann et al., 2008, 2011), which links small cells with their larger neighbors and treats them as a single cell in the discretization process without actually merging them. This method circumvents the complexity associated with the merging procedure. Another technique is the Flux Redistribution Method, which redistributes the fluxes between small cells and their larger neighbors to satisfy the stability condition of the small cells. Colella et al. (2006) employed this method for simulating compressible flows on cut-cells grids. Although the Flux Redistribution Method is relatively easy to implement, it can result in a loss of accuracy at the boundaries (May and Berger, 2017). Other methods, such as the h-box method (Helzel et al., 2005; Berger and Helzel, 2012) and the state redistribution method (Berger and Giuliani, 2021), have also been proposed in the literature to overcome the small cell problem. Each of these techniques has its own advantages and disadvantages, and their suitability can vary depending on the specific application.

Instead of employing the Cut-Cells (CC) approach, one can utilize the Immersed Boundary Method (IBM) (Peskin, 1972) to handle complex geometries. The IBM method discretizes the equations on the Cartesian grid by applying a special treatment to the cells near the boundaries, thereby accounting for the presence of the boundaries. Mittal and Iaccarino (Mittal and Iaccarino, 2005) have provided an extensive review of the IBM method and the various techniques used to treat boundaries within this framework. Among these techniques, the sharp interface method (Mohd-Yusof, 1997; Iaccarino and Verzicco, 2003; Sotiropoulos and Yang, 2014; Ghias et al., 2007; Mittal et al., 2008) is one of the most popular. This method employs an interpolation procedure to impose boundary conditions on the cells near the boundaries. By controlling the accuracy of the interpolation procedure, one can obtain the desired accuracy of the solution at the boundaries. While the imposition of boundary conditions is generally more complex for the IBM method compared to the CC method, the IBM method offers the significant advantage of not imposing any restrictions on the time step when using an explicit time integration scheme. Additionally, the IBM method is well-suited for problems involving moving boundaries. However, one of the main disadvantages of the IBM method is the difficulty in ensuring numerical conservation at the boundaries.

In this work, we propose three methods that combine the advantages of both the Cut-Cells (CC) and Immersed Boundary Method (IBM) in terms of conservation, accuracy, and stability while avoiding their respective disadvantages. To ensure the conservation of the numerical solution, the cut-cells grid is used to discretize the equations, but a reconstruction procedure similar to the one used in the IBM method is employed to impose the boundary conditions. This reconstruction procedure allows to obtain an accurate solution at the boundaries without imposing any restrictions on the time step. However, the IBM reconstruction introduces a loss of conservation, necessitating an additional step called conservative correction step to ensure the conservation of the various conservative variables. The main contribution of this work is the development of this conservative correction step that restores conservation on the cut-cells grid when using the IBM method for boundary condition treatment. The first method relies on a straightforward flux redistribution procedure that globally restores conservation on the cut-cells grid. The other two methods employ newly developed optimization procedures to ensure local conservation of the numerical solution within each cell. These methods have been tested on two-dimensional inviscid compressible flow problems, demonstrating results comparable to those obtained with the standard Cut-Cells method in terms of accuracy and conservation. Furthermore, the methods are stable and can be effectively used with an explicit time integration scheme without encountering any stability issues related to the small cut-cells.

The paper is organized as follows: Section 2 presents the governing equations for inviscid compressible flow and the finite volume discretization of these equations using the Cut-Cells (CC) method. It then details the Immersed Boundary Method (IBM) and the Reconstructed Cut-Cells Method (RCCM), which is a combination of the CC and IBM methods without the conservative correction step. Section 3 introduces the methods that employ a correction step that restore conservation on the cut-cells grid when using the IBM method for boundary condition treatment. Section 4 presents numerical results demonstrating the accuracy and conservation properties of the different methods using four test cases. Finally, Section 5 concludes the paper.

2. Numerical method

The goal of this section is to present the governing equations for inviscid compressible flow, described by the Euler equations and their finite volume discretization using the Cut-Cells (CC) method and the Immersed Boundary Method (IBM). It is divided into three parts. The first one briefly presents the Euler equations that govern the inviscid compressible flow. The second part details the finite volume discretization of these equations using the Cut-Cells (CC) method. Finally, the third part introduces the Immersed Boundary Method (IBM) and a combination of the CC and IBM methods called the Reconstructed Cut-Cells Method (RCCM). These three methods will serve as the basis for the development of the new conservative methods presented in Section 3.

2.1. Governing equations

The integral form of the Euler equations, which encapsulate the conservation laws of mass, momentum, and energy, is given by Eq. (1).

$$\frac{\partial}{\partial t}\int_{\Omega}\mathbf{U}d\Omega+\oint_{\partial\Omega}\mathbf{F}(\mathbf{U})dA=0 \tag{1}$$

Here, **U** denotes the vector of conservative variables, and **F** represents the flux vector along the outward unit normal **n** to the boundary. The conservative variables and the flux vector are defined in Eq. (2).

$$\mathbf{U}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{bmatrix}\qquad \mathbf{F}=\begin{bmatrix}\rho(\mathbf{v}\cdot\mathbf{n})\\ \rho u(\mathbf{v}\cdot\mathbf{n})+pn_x\\ \rho v(\mathbf{v}\cdot\mathbf{n})+pn_y\\ \rho E(\mathbf{v}\cdot\mathbf{n})+p(\mathbf{v}\cdot\mathbf{n})\end{bmatrix} \tag{2}$$

where ρ is the density, $\mathbf{v}=(u,v)$ is the velocity vector, p is the pressure and E is the total energy. The system of equations is closed using the perfect gas relation, given by Eq. (3).

$$p=(\gamma-1)\left(\rho E-\frac{1}{2}\rho\|\mathbf{v}\|^2\right) \tag{3}$$

where γ is the ratio of specific heats.

2.2. Finite volume discretization of the Euler equations

This section details the finite volume discretization of the Euler equations using the Cut-Cells (CC) method and discusses the conservation properties of the method.

From the finite volume method (FVM) viewpoint, the integral form of the Euler equations can be discretized as shown in Eq. (4).

$$\mathbf{U}_i^{n+1}=\mathbf{U}_i^n-\frac{\Delta t^n}{\Delta V_i}\sum_{f\in\partial V_i}\mathbf{F}_f A_f \tag{4}$$

where:

- $\mathbf{U}_i^n$ is the vector of conservative variables at the cell i at the nth time step.
- Δt^n and ΔV_i are the time step and the volume of the cell i, respectively.
- $\mathbf{F}_f$ is the numerical flux at the face f, shared by the two cells.
- $A_f=\int_f dA$ is the area of the face f.

The numerical flux vector $\mathbf{F}_f$ is an approximation of the flux vector **F** at the face f:

$$\mathbf{F}_f(\mathbf{U}_{i_1},\mathbf{U}_{i_2},\ldots)\approx\frac{1}{A_f}\int_f\mathbf{F}(\mathbf{U})dA \tag{5}$$

where $\mathbf{U}_{i_1},\mathbf{U}_{i_2},\ldots$ are the conservative variables at the cells surrounding face f.

The discretization form of Eq. (4) is known as the *conservative discretization form* of the Euler Eqs. (1). Any method that can be written in this form is considered *conservative*. Using Eq. (4), it can be shown that the sum of all differences between the conservative variables at time steps $n+1$ and n depends only on the fluxes at the domain boundaries $\partial\Omega$, as shown in Eq. (6). In particular, if there is no flux at the boundaries, the sum of all conservative variables remains constant over time. A method that respects the property given by Eq. (6) is said to be *globally conservative*.

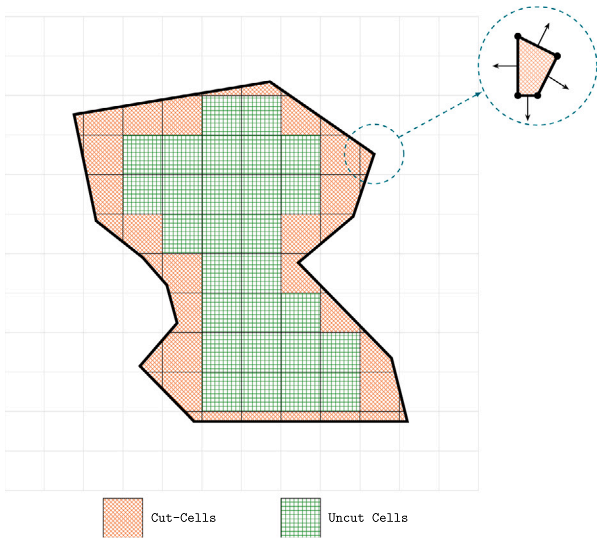


Fig. 1. Illustration of the cut-cells grid used by the CC solver. The dashed encirclement shows a detailed view of a cut-cell.

$$\sum_{i\in\Omega}\Delta V_i\mathbf{U}_i^{n+1}=\sum_{i\in\Omega}\Delta V_i\mathbf{U}_i^n-\Delta t^n\sum_{f\in\partial\Omega}\mathbf{F}_f A_f \tag{6}$$

Having a conservative method is crucial as it preserves the conservation properties derived from physical laws in the numerical solution. It also allows for the correct capture of shocks and other discontinuities in the solution of the Euler equations (Ndiaye et al., 2024; LeVeque, 1992).

In this work, the approximate Riemann solver of Roe (1981) is used to compute the numerical fluxes $\mathbf{F}_f$ at the faces f.

The CC solver is directly based on the FVM discretization described above. First, the cut-cells grid is generated by clipping the cells intersected by the boundary. Then, the FVM discretization is applied to the cut-cells grid. Fig. 1 shows an example of the cut-cells grid used by the CC solver. Ghost cells are added at the cut-cells to compute the numerical fluxes at the boundaries (Blazek, 2015). Since no special treatment such as cell-merging or flux redistribution is applied to the small cells, the time step Δt^n can be very small to satisfy the CFL condition. Nevertheless, the CC solver is accurate and conservative. The only other potential downside is the deterioration of mesh quality near the boundaries due to the arbitrary shape of the cut-cells, which can affect the accuracy of the FVM solver.

The next section will present the IBM method and the Reconstructed Cut-Cells Method (RCCM) method, which is an approach proposed by the authors in Ndiaye et al. (2024) that combines the advantages of the CC and IBM methods.

2.3. Immersed Boundary Method (IBM)

This section[1] introduces the IBM and RCCM methods, which are similar to the CC method but use different grids and a reconstruction procedure to impose the boundary conditions. It is divided into three parts. The first part presents the IBM solver with a focus on the general idea of the method along with the grid used to discretize the equations.

[1] The content of this section is based on the conference paper (Ndiaye et al., 2024) and is presented here for the sake of completeness.

The second part presents the details of the reconstruction procedure used by the IBM solver to impose the boundary conditions. Finally, the third part introduces the RCCM solver, which combines the advantages of the CC and IBM methods.

2.3.1. IBM solver

The IBM solver is also based on the FVM discretization described in Section 2.2. However, for the cells near the boundaries, a reconstruction procedure is used to impose the boundary conditions. The reconstruction procedure will be summarized in the following section. But first, the grid used by the IBM solver will be described.

The IBM solver only considers (solves) the cells that have their centers inside the domain Ω. The cells that have their center outside the domain Ω are not solved by the solver. These cells are categorized into two groups:

- The cells that are completely outside the domain Ω.
- The cells named outer cut-cells (`OC-Cells`) that are partially inside the domain Ω but have their centers outside the domain Ω.

Although the `OC-Cells` are not solved by the solver, they are used in the reconstruction procedure since they contain some information about the boundary conditions.

Regarding the solved cells, they are also categorized into two subgroups: the boundary cells (`B-Cells`) and the inside cells (`I-Cells`). The `I-Cells` are completely inside the domain Ω and do not have any `OC-Cells` as neighbors. The `B-Cells` are the remaining unidentified cells and fall into one of the following two classes:

- The cells that are intersected by the boundary $\partial\Omega$ and have their center inside the domain Ω. These cells can be called boundary cut-cells (`BC-Cells`).
- The cells that are completely inside the domain Ω and have at least one neighbor that is an `OC-Cell`. These cells can be called boundary inside-cells (`BI-Cells`).

The `B-Cells` form then a watertight approximation of the boundary $\partial\Omega$. Fig. 2 shows an example of the grid used by the IBM solver. The IBM grid can be efficiently generated without explicitly clipping the cells with the boundary. Indeed, the reconstruction procedure used by the IBM solver to impose the boundary conditions does not need the explicit geometric definition of the boundary cells.

The `I-Cells` are defined such that they have a complete stencil of neighbors solved by the IBM solver. Thus, the `I-Cells` are directly discretized using the FVM discretization described above. The `B-Cells`, on the other hand, are solved using the reconstruction procedure summarized below. The IBM solver functions then as follows:

- The `I-Cells` are advanced in time using the FVM discretization: $\mathbf{U}^{n}_{\texttt{I-Cells}} \rightarrow \mathbf{U}^{n+1}_{\texttt{I-Cells}}$.
- The `B-Cells` are reconstructed implicitly using the boundary conditions (BC) and the new values of the `I-Cells`:

$$\mathbf{U}^{n+1}_{\texttt{B-Cells}} = \texttt{Reconstruction}(\mathbf{U}^{n+1}_{\texttt{B-Cells}}, \mathbf{U}^{n+1}_{\texttt{I-Cells}}, \text{BC}) \tag{7}$$

2.3.2. Reconstruction procedure

The reconstruction procedure, as described in Eq. (7), performs an interpolation for the primitive variables $\mathbf{W} = (\rho, u, v, p)$ of the `B-Cells` using the updated values of the `I-Cells` and the boundary conditions. The interpolation is carried out using the following steps:

1. For each primitive variable ϕ ($\phi = \rho, u, v$ or p), it is assumed that the variable ϕ is approximated by a linear function around the cell center of the `B-Cell`:

$$\phi = a + b(x - x_0) + c(y - y_0) \tag{8}$$

where (x_0, y_0) is the center of the `B-Cell` to be reconstructed.

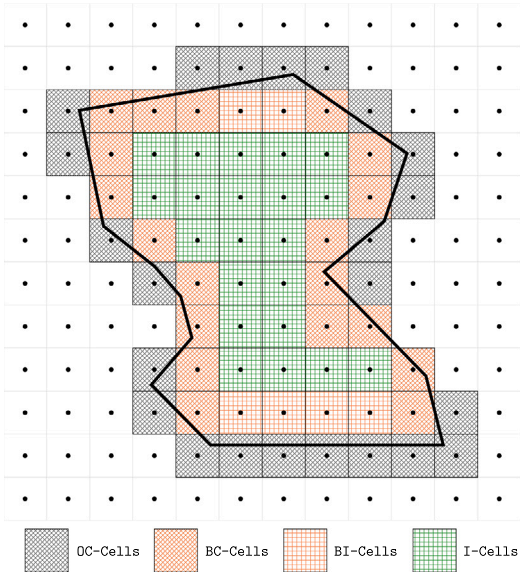


Fig. 2. Illustration of the different types of cells in the IBM grid.

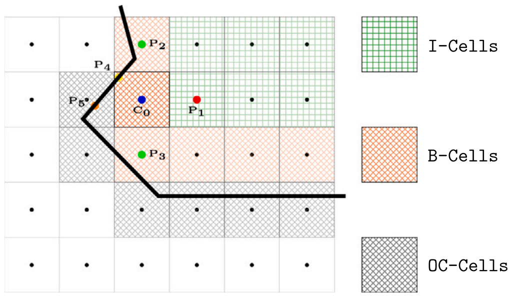


Fig. 3. Illustration of a reconstruction stencil $\{P_i\}$ of a boundary cell C_0.

2. Now, the coefficients (a, b, c) for each primitive variable and each boundary cell are to be determined. To do that, a set of points (called reconstruction stencil) around the cell center of each boundary cell, where some information about the primitive variables is known, is used. This information can be the value of the primitive variable at the center of an `I-Cell` or another `B-Cell`, or a Dirichlet boundary condition. For points on a boundary with a Neumann boundary condition (e.g., "no-slip" or "supersonic outlet"), the information is the derivative of the primitive variable along the normal direction to the boundary. Typically, the reconstruction stencil includes the cell centers of solved cells (`I-Cells` and `B-Cells`) that are neighbors of the current boundary cell, and a set of points on the boundary close to the cell center of the current boundary cell. Fig. 3 shows an example of a reconstruction stencil for a boundary cell, where the points on the boundary are the closest points to the centers of the current `B-Cell` and its `OC-Cells` neighbors.
3. Once the reconstruction stencil is defined, small linear systems are assembled for each `B-Cell` in order to find the coefficients of the polynomial.
4. Since the stencil of a `B-Cell` can contain others `B-Cells`, all the systems are assembled into one global system that is solved to find the coefficients of the polynomial for all the `B-Cells` simultaneously.

The reconstruction procedure is very flexible and can be used with any type of boundary conditions. Since it is not based on the conservative discretization Eq. (4), it will introduce therefore a loss of conservation. Hence, the IBM solver is not conservative.

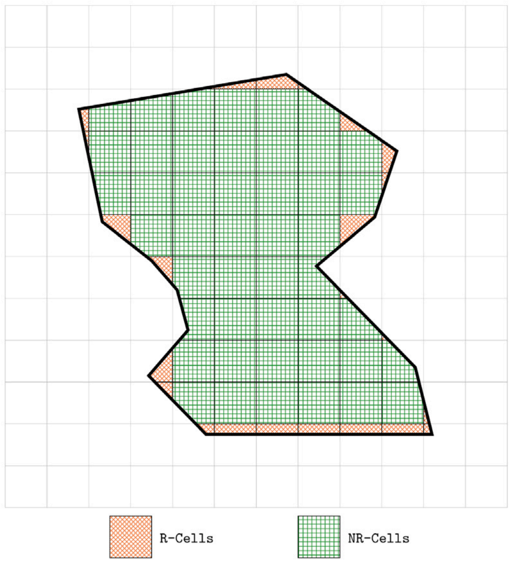


Fig. 4. Illustration of the RCCM grid.

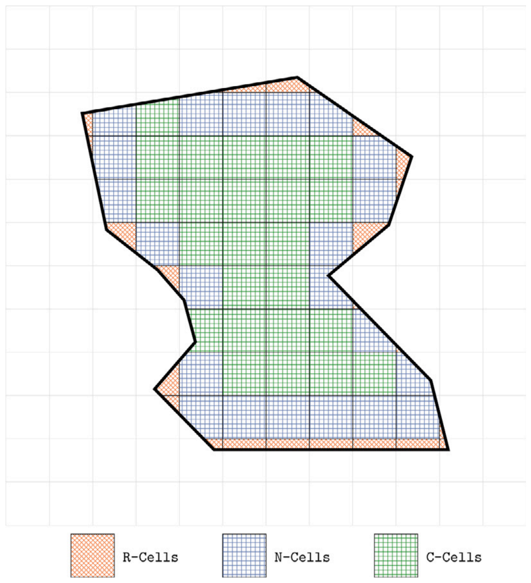


Fig. 5. Illustration of the different types of cells in the grid used by the conservative methods.

2.3.3. *Reconstructed Cut-Cells Method (RCCM) solver*

The Reconstructed Cut-Cells Method (RCCM) solver is a combination of the CC and IBM solvers. The idea consists of using the reconstruction procedure of the IBM solver on the cut-cells for the purpose of avoiding the small cell problem.

The RCCM method uses the cut-cells grid shown in Fig. 1 for discretization. The cells are, this time, categorized into two groups: the non-reconstructed cells (`NR-Cells`), which are discretized using the FVM discretization, and the reconstructed cells (`R-Cells`). The `R-Cells` are defined as the clipped version of the `OC-Cells` in the IBM grid shown in Fig. 2. This choice is motivated by the fact that the set of `OC-Cells` contains all the small cells and is, on average, half the size of the set of all cut-cells. Once the reconstructed cells are identified, the remaining cells are the `NR-Cells`, which are discretized using the FVM discretization. Fig. 4 shows the grid used by the RCCM solver.

In summary, the RCCM functions as follows:

1. The non-reconstructed cells are advanced in time using the FVM discretization:

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n - \frac{\Delta t^n}{\Delta V_i} \sum_{f \in \partial V_i} \mathbf{F}_f A_f \qquad \text{for } i \in \mathtt{NR-Cells} \tag{9}$$

2. The reconstructed cells are reconstructed using the boundary conditions (BC) and the new values of the non-reconstructed cells:

$$\mathbf{U}_i^{n+1} = \mathtt{Reconstruction}(\mathbf{U}^{n+1}_{\mathtt{R-Cells}}, \mathbf{U}^{n+1}_{\mathtt{NR-Cells}}, \mathrm{BC}) \qquad \text{for } i \in \mathtt{R-Cells} \tag{10}$$

The reconstruction step described in Eq. (10) is similar to the one used by the IBM solver. Here, the centroid of the `R-Cells` is used as the center of the reconstruction and the boundary points of the reconstruction stencil are chosen to be the centers of the boundary faces composing the cut-cells. In the RCCM solver, the reconstruction procedure can be viewed as a semi-implicit treatment of the reconstructed cells that will relieve the time step restriction due to the small cells. As in the IBM solver, the reconstruction procedure will also induce a loss of conservation in the RCCM solver. However, since the number and the volume occupied by the reconstructed cells of the RCCM solver are in general smaller than those in the IBM solver, the errors associated with the loss of conservation are expected to be smaller. This is indeed the case, as shown in the numerical results presented in Ndiaye et al. (2024) and in Section 4 of this work.

In this section, the governing equations for inviscid compressible flow and their finite volume discretization using the CC, IBM, and RCCM methods have been presented. All three methods divide the cells of the computational domain into two sets that are solved differently. The CC method uses the finite volume discretization directly on all the cells of the grid making it fully conservative but affected by the small cell problem. The IBM method does not consider cut-cells and uses a reconstruction procedure instead to impose the boundary conditions on the cells near the boundaries. The RCCM method combines the advantages of the CC and IBM methods by using the cut-cells grid for discretization and the reconstruction procedure of the IBM method to impose the boundary conditions. The IBM and RCCM methods are not conservative due to the reconstruction procedure but are not affected by the small cell problem. The next section will present various methods based on the RCCM solver that allow to fully restore the conservation properties on the numerical solutions.

3. Conservative methods

This section presents three methods designed to restore the conservation properties on the numerical solutions. It is divided into four parts. The first part presents the general description of the conservative methods along with some terminology and the grid used by these methods. The following three parts present the three conservative methods. All these methods are based on the RCCM solver with an additional conservative correction step applied after each iteration of the RCCM solver's time integration. The first method relies on a straightforward flux redistribution procedure that globally restores conservation on the cut-cells grid. The flux redistribution technique used in this method was proposed by Pember et al. (1995) and Colella et al. (2006). The last two methods correspond to new innovative approaches developed by the authors that ensure local conservation of the numerical solution within each cell by solving an optimization problem.

3.1. *Overview of the conservative methods*

Grid description

The conservative methods will use the same grid as the RCCM solver. Hence, the cells are initially divided into two sets: the non-reconstructed cells (`NR-Cells`) and the reconstructed cells (`R-Cells`). But, to apply the conservative correction step that restore the conservation, the `NR-Cells` will be further separated into two subsets:

1. The cells that have at least one `R-Cells` as neighbor. These cells are called the neighbor of the reconstructed cells (`N-Cells`).
2. The remaining cells that are not `NR-Cells` are called conserved cells (`C-Cells`).

Fig. 5 illustrates an example of the grid used by the conservative methods.

Since `N-Cells` share a face with at least one `R-Cell`, they are affected by the loss of conservation introduced by the reconstruction of the `R-Cells`. Therefore, the solutions for these cells typically needs correction alongside the solution for the `R-Cells`. Conversely, `C-Cells` are not impacted by the reconstruction of the `R-Cells`, and their solutions does not require any modification during the conservative correction step.

Notation

Table 1 summarizes the different notations used in the following sections to describe the different conservative methods.

General description of the conservative methods

The various conservative methods operate in a similar manner. The objective is to update the current solution $\mathbf{U}^n$ to obtain a new conservative solution $\mathbf{U}^{n+1}$.

First, the new time step Δt^n is computed without taking into account the `R-Cells` as shown in Eq. (11).

$$\Delta t^n = \text{CFL} \min_{i \notin \texttt{R-Cells}} \Delta t_i^n \tag{11}$$

Next, a conservative solution $\mathbf{U}_{cc}^{n+1}$ is calculated using the CC solver with Eq. (12). Since the `R-Cells` were ignored in the computation of the time step, the CFL condition is not satisfied for the CC solver. Thus, the update (12) is performed without any restriction on the time step, but making it unstable at the same time.

$$\mathbf{U}_{cc,i}^{n+1} = \mathbf{U}_i^n - \frac{\Delta t^n}{\Delta V_i} \sum_{f \in \partial V_i} \mathbf{F}_f A_f \tag{12}$$

The next step is to compute the non-conservative solution $\mathbf{U}_{nc}^{n+1}$ using the RCCM solver. Eq. (13) recalls the computation of the non-conservative solution $\mathbf{U}_{nc}^{n+1}$. This solution is stable due to the reconstruction on the `R-Cells`, but it is not conservative for the same reason.

$$\begin{cases} \mathbf{U}_{nc,i}^{n+1} = \mathbf{U}_{cc,i}^{n+1} & \text{for } i \in \texttt{C-Cells} \text{ and } \texttt{N-Cells} \\ \mathbf{U}_{nc,i}^{n+1} = \texttt{Reconstruction}(\mathbf{U}_{nc}^{n+1}, \text{BC}) & \text{for } i \in \texttt{R-Cells} \end{cases} \tag{13}$$

Finally, the last step is to construct the new conservative solution $\mathbf{U}_c^{n+1}$ using the solution $\mathbf{U}_{nc}^{n+1}$ and $\mathbf{U}_{cc}^{n+1}$.

The following sections describe three different methods that allow to compute the new conservative solution $\mathbf{U}_c^{n+1}$ while ensuring stability.

3.2. Flux Redistribution Method

The Flux Redistribution Method (FRM) is based on the flux redistribution technique (Chern and Colella, 1987; Bell et al., 1991; Pember et al., 1995; Colella et al., 2006). In the present work, the FRM is employed for the purpose of restoring conservation in the Immersed Boundary Method. A first optional step involves computing a new stable but still non-conservative solution $\mathbf{U}_{cont}^{n+1}$ using a linear combination of the solutions $\mathbf{U}_{cc}^{n+1}$ and $\mathbf{U}_{nc}^{n+1}$, as shown in Eq. (14). The non-conservative solution $\mathbf{U}_{nc}^{n+1}$ is obtained using the RCCM solver based on a reconstruction procedure.

Table 1
Summary of the different notations used to describe the conservative methods.

Notation	Description
$\mathbf{U}$	The vector of conservative variables corresponding to the current solution.
$\mathbf{U}_{cc}$	The conservative solution obtained with the CC solver.
$\mathbf{U}_{nc}$	The non-conservative solution obtained with the RCCM solver.
$\mathbf{U}_{c}$	The new conservative solution obtained with the conservative methods.
$\mathbf{F}_f$	The fluxes at face f obtained with the FVM discretization using the current solution $\mathbf{U}$.
$\mathbf{F}_{m,f}$	Modified fluxes at face f used to obtain the conservative solution $\mathbf{U}_c$.

$$\mathbf{U}_{cont,i}^{n+1} = \alpha_i \mathbf{U}_{cc,i}^{n+1} + (1-\alpha_i)\mathbf{U}_{nc,i}^{n+1} \tag{14}$$

The parameter α_i is defined as the ratio of the volume of cell i to the volume of the uncut cell i, as shown in Eq. (15).

$$\alpha_i = \frac{\Delta V_i}{\Delta V_{uncut,i}} \tag{15}$$

If the cell i is an uncut cell, then $\alpha_i = 1$ and the new solution $\mathbf{U}_{cont,i}^{n+1}$ is equal to the conservative solution $\mathbf{U}_{cc,i}^{n+1}$ given by the CC solver. If the cell i is a cut-cell, then $0 < \alpha_i < 1$ and by combining Eqs. (12), (14) and (15), it is evident that the division by the small cell volume ΔV_i is avoided, ensuring the stability of the solution $\mathbf{U}_{cont}^{n+1}$. For the remainder of this section, the solution $\mathbf{U}_{cont}^{n+1}$ will be used as the non-conservative solution $\mathbf{U}_{nc}^{n+1}$.

Next, the conservative solution $\mathbf{U}_c^{n+1}$ is computed using the solutions $\mathbf{U}_{nc}^{n+1}$ and $\mathbf{U}_{cc,i}^{n+1}$ with the flux redistribution technique. The redistribution is performed to satisfy the global conservation relation (16).

$$\sum_{i\in\Omega} \Delta V_i \mathbf{U}_{c,i}^{n+1} = \sum_{i\in\Omega} \Delta V_i \mathbf{U}_{cc,i}^{n+1} \tag{16}$$

Since the solution $\mathbf{U}_{cc}^{n+1}$ is conservative, satisfying the constraint (16) ensures that the solution $\mathbf{U}_c^{n+1}$ will also be globally conservative by respecting the conservation relation (6). The flux redistribution is carried out as follows:

1. Initialize the conservative solution $\mathbf{U}_c^{n+1}$ with the solution $\mathbf{U}_{nc}^{n+1}$.
$$\mathbf{U}_c^{n+1} = \mathbf{U}_{nc}^{n+1} \tag{17}$$
2. After that, compute the mass error ΔM_i for each cell i that allows to satisfy the conservation relation (16).
$$\Delta M_i = \Delta V_i(\mathbf{U}_{cc,i}^{n+1} - \mathbf{U}_{c,i}^{n+1}) \tag{18}$$
Note that the mass error ΔM_i is equal to zero for the uncut cells. Therefore, only the mass error ΔM_i of the cut-cells will be redistributed.
3. Finally, redistribute all the mass errors ΔM_i among the cells such that the relation (16) is satisfied.
   (a) Loop over the cut-cells i and generate a stencil $S(i)$ of neighbors of the cell i. In this work, the stencil $S(i)$ is taken to be the union of the cell i and the set of cells that share a face with the cell i or one of its direct neighbors.
   (b) Loop over the elements j of the stencil $S(i)$ and distribute a part of the mass error ΔM_i to the neighbor j.
   $$\Delta M_{i,j} = \omega_{i,j} \Delta M_i \tag{19}$$

The weights $\omega_{i,j}$ are chosen such that $\sum_{j\in S(i)} \omega_{i,j} = 1$ for each cell i and are computed as follows.

$$\omega_{i,j} = \frac{\Delta V_j}{\sum_{k\in S(i)} \Delta V_k} \tag{20}$$

(c) Update the conservative solution $\mathbf{U}_{c,j}^{n+1}$ of the neighbor j.

$$\mathbf{U}_{c,j}^{n+1} = \mathbf{U}_{c,j}^{n+1} + \frac{1}{\Delta V_j}\Delta M_{i,j} \tag{21}$$

The solution $\mathbf{U}_c^{n+1}$ obtained with the Flux Redistribution Method is both conservative and stable. However, the conservation is only global. The next section will present the Flux Interpolation Method, which aims to restore local conservation in the numerical solutions.

3.3. Flux Interpolation Method

To achieve locally conservative solutions, the local conservative discretization form (4) must be employed. This section introduces a new method developed by the authors and called the Flux Interpolation Method (FIM) to accomplish this goal. The method operates as follows:

For each cell i, the new conservative solution $\mathbf{U}_{c,i}^{n+1}$ is expressed in the form of Eq. (22).

$$\mathbf{U}_{c,i}^{n+1} = \mathbf{U}_i^n - \frac{\Delta t^n}{\Delta V_i}\sum_{f\in\partial V_i} \mathbf{F}_{m,f} A_f \tag{22}$$

Thus, it suffices to compute the fluxes $\mathbf{F}_{m,f}$ using the solution $\mathbf{U}_{nc}^{n+1}$ from the RCCM solver to obtain the conservative solution $\mathbf{U}_c^{n+1}$.

Since the conservation form (22) is already satisfied for the non-reconstructed cells (`C-Cells` and `N-Cells`), the fluxes $\mathbf{F}_{m,f}$ for all faces f shared by two non-reconstructed cells are set equal to the fluxes $\mathbf{F}_f$ obtained with the FVM discretization using the solution $\mathbf{U}^n$. Consequently, the solutions at the `C-Cells` remain unchanged, as shown in Eq. (23).

$$\mathbf{U}_{c,i}^{n+1} = \mathbf{U}_{nc,i}^{n+1} \qquad \text{for } i \in \mathtt{C-Cells} \tag{23}$$

For the faces f that are shared by two `R-Cells` or by an `R-Cell` and an `N-Cell`, referred to as interpolated faces (`I-Faces`), the fluxes $\mathbf{F}_{m,f}$ are interpolated using the solution $\mathbf{U}_{nc}^{n+1}$. The interpolation is performed through the following steps:

1. The solution $\mathbf{U}_c^{n+1}$ at the `R-Cells` is set to be equal to the solution $\mathbf{U}_{nc}^{n+1}$.

   $$\mathbf{U}_{c,i}^{n+1} = \mathbf{U}_{nc,i}^{n+1} \qquad \text{for } i \in \mathtt{R-Cells} \tag{24}$$

   Only the solution $\mathbf{U}_c^{n+1}$ at the `N-Cells` remains then to be computed.
2. A combination of Eqs. (22) and (24) is then made for the goal of obtaining a set of equations that constrain the fluxes $\mathbf{F}_{m,f}$ at the `I-Faces`:

   $$\sum_{\substack{f\in\partial V_i\\ f\in\mathtt{I-Faces}}} \mathbf{F}_{m,f} A_f = \frac{\Delta V_i}{\Delta t^n}\left(\mathbf{U}_i^n - \mathbf{U}_{nc,i}^{n+1}\right) - \sum_{\substack{f\in\partial V_i\\ f\notin\mathtt{I-Faces}}} \mathbf{F}_f A_f \quad \text{for } i \in \mathtt{R-Cells} \tag{25}$$

   Eq. (25) is a linear system of equations of the form $Kx = b$ where the unknown x represents the fluxes $\mathbf{F}_{m,f} A_f$ at the `I-Faces`. The right-hand side b depends on the solution $\mathbf{U}_{nc}^{n+1}$, and the matrix K depends only on the discretization grid. The dimensions of the matrix K are $n\times m$ where n is the number of reconstructed cells and m is the number of `I-Faces`.
3. If the number of `R-Cells` is less than the number of `I-Faces`, which is usually the case, then the system of Eqs. (25) can be solved using the least norm method:

   $$\begin{aligned}&\min_{\mathbf{F}_{m,f}} \sum_{f\in\mathtt{I-Faces}} \left(\mathbf{F}_{m,f}A_f - \mathbf{F}_f A_f\right)^2\\ &\text{s.t.} \quad \mathbf{U}_{c,i}^{n+1} = \mathbf{U}_{nc,i}^{n+1} \qquad \text{for } i\in\mathtt{R-Cells}\end{aligned} \tag{26}$$

   In matrix form, the optimization Eq. (26) can be written as:

   $$\begin{aligned}&\min_x \quad \|x - x_0\|^2\\ &\text{s.t.} \quad Kx = b\end{aligned} \tag{27}$$

   where x_0 is the vector of the standard FVM fluxes $\mathbf{F}_f A_f$ at the `I-Faces`. The least norm solution of (27) is given by Eq. (28).

   $$x = x_0 + K^T(KK^T)^{-1}(b - Kx_0) \tag{28}$$
4. If the number of `R-Cells` exceeds the number of `I-Faces`, the system of Eqs. (25) becomes over-determined, and not all conservation constraints can be satisfied. In this case, the system of Eqs. (25) is solved using the least squares method:

   $$\min_{\mathbf{F}_{m,f}} \sum_{i\in\mathtt{R-Cells}} \left(\mathbf{U}_{c,i}^{n+1} - \mathbf{U}_{nc,i}^{n+1}\right)^2 \tag{29}$$

   In matrix form, the minimization problem (29) can be written as:

   $$\min_x \quad \|Kx - b\|^2 \tag{30}$$

   The solution to the optimization problem (30) is given by Eq. (31).

   $$x = (K^TK)^{-1}K^Tb \tag{31}$$
5. Once all the fluxes $\mathbf{F}_{m,f}$ are computed, the conservative solution $\mathbf{U}_c^{n+1}$ at the `N-Cells` is obtained using Eq. (22).

The Flux Interpolation Method is stable since the reconstructed solutions at the `R-Cells` are kept unchanged. However, the method is not always conservative since the system of Eqs. (25) can be over-determined in some cases. Nevertheless, in most situations, the method is conservative since the system (25) is usually under-determined. The next section will introduce another new method developed by the authors and called Flux Correction Method, which is a variant of the Flux Interpolation Method that is always locally conservative.

3.4. Flux Correction Method

The Flux Correction Method (FCM) closely resembles the Flux Interpolation Method but with relaxed constraints from Eq. (25). Here, an alternative formulation of the method will be presented. The approach begins with the non-conservative solution $\mathbf{U}_{nc}^{n+1}$, obtained, for example, using the RCCM solver. The solutions at the `C-Cells` are left unchanged since they already satisfy the conservative discretization form (4). Conversely, the solutions at the `R-Cells` and `N-Cells` are perturbed to meet the conservative discretization form (4). The updated solution $\mathbf{U}_c^{n+1}$ for the `R-Cells` and `N-Cells` is expressed as in Eq. (32).

$$\mathbf{U}_{c,i}^{n+1} = \mathbf{U}_{cc,i}^{n+1} - \frac{\Delta t^n}{\Delta V_i}\sum_{\substack{f\in\partial V_i\\ f\in\mathtt{I-Faces}}} \left(\mathbf{F}_{m,f} - \mathbf{F}_f\right) A_f \quad \text{for } i \in \mathtt{R-Cells} \cup \mathtt{N-Cells} \tag{32}$$

The objective is then to compute the perturbation $\Delta\mathbf{F}_f = \left(\mathbf{F}_{m,f} - \mathbf{F}_f\right) A_f$ for all the faces $f \in \mathtt{I-Faces}$. Remark that the update in Eq. (32) corresponds to a fully conservative update of the form (4), since only the values of the fluxes at the interpolated faces f are modified compared to the Cut-Cells solver. If the perturbed fluxes $\Delta\mathbf{F}_f$ vanish, then the solution $\mathbf{U}_c^{n+1}$ is exactly identical to the solution $\mathbf{U}_{cc}^{n+1}$ obtained with the CC solver. However, since the solution $\mathbf{U}_{cc}^{n+1}$ is unstable, the perturbed fluxes $\Delta\mathbf{F}_f$ will be non-zero in general so that they can positively affect the stability of the solution $\mathbf{U}_c^{n+1}$.

The perturbed fluxes $\Delta\mathbf{F}_f$ are computed using the reconstructed solution $\mathbf{U}_{nc}^{n+1}$ from Eq. (13), following these steps:

Table 2
Summary of the different methods.

Method	Conservation	Small cells Δt Constraint	Comments
CC	YES	YES	Stable only with very small time steps.
IBM	NO	NO	Fastest method, but has the largest errors due to loss of conservation and reconstruction.
RCCM	NO	NO	Errors from loss of conservation and reconstruction are smaller than those in the IBM method (Ndiaye et al., 2024).
FRM	YES	NO	Conservation is guaranteed only globally.
FIM	YES/NO	NO	In rare cases, the method may not be fully conservative.
FCM	YES	NO	Less stable than the Flux Interpolation Method, but always locally conservative.

1. Using Eq. (32), the differences between the solutions $\mathbf{U}_c^{n+1}$ and $\mathbf{U}_{nc}^{n+1}$ at the `R-Cells` and `N-Cells` is minimized:

$$\min_{\Delta \mathbf{F}_f} \sum_{i \in \texttt{R-Cells} \cup \texttt{N-Cells}} \left(\mathbf{U}_{c,i}^{n+1} - \mathbf{U}_{nc,i}^{n+1}\right)^2 \tag{33}$$

2. As in the Flux Interpolation Method, the minimization Eq. (33) can be written in matrix form as:

$$\min_{x} \quad \|Kx - b\|^2 \tag{34}$$

where x is the vector of the perturbed fluxes $\Delta \mathbf{F}_f$ at the `I-Faces`. The right-hand side b depends on the solution $\mathbf{U}_{nc}^{n+1}$, and the matrix K depends only on the discretization grid. The dimensions of the matrix K is $n \times m$ where n is the total number of `R-Cells` and `N-Cells` and m is the number of `I-Faces`. The problem (33) is solved using the least squares method to obtain the perturbed fluxes $\Delta \mathbf{F}_f$.

3. Once the perturbed fluxes $\Delta \mathbf{F}_f$ are computed, the conservative solution $\mathbf{U}_c^{n+1}$ at the `R-Cells` and `N-Cells` is obtained using Eq. (32).

The Flux Correction Method is conservative in general since the conservative update (32) is applied to every cell. The method is also stable because the reconstructed solutions $\mathbf{U}_{nc}^{n+1}$ are used to constrain the perturbed fluxes $\Delta \mathbf{F}_f$. However, since the solutions at the small cells initially obtained with a reconstruction are now perturbed, However, since the solutions at the small cells initially obtained through reconstruction are now perturbed, the stability of the Flux Correction Method may not be as robust as that of the Flux Interpolation Method.

In this section, three conservative methods have been presented that allow to restore the conservation properties on the numerical solutions. The Flux Redistribution Method is the simplest method and ensures global conservation of the solution by redistributing the mass errors among the cells. The Flux Interpolation Method and the Flux Correction Method are more sophisticated methods that ensure local conservation of the solution within each cell by solving an optimization problem. The next section will present numerical results of the different methods using various test cases.

Summary of the different methods

The properties of the methods introduced so far are summarized in Table 2.

4. Numerical results

This section presents four test cases to evaluate the performance of the different methods. The first test case is the shock tube problem of Sod (1978), which is used to examine the conservation properties of the various methods. The second test case involves a transonic flow through a canal with a bump, aimed at qualitatively assessing the impact of conservation properties on capturing critical flow features. The third test case examines the confluence of two supersonic flows, providing a measure of the accuracy of the different methods for handling discontinuous solutions. The final test case is the supersonic vortex problem, which is used to study the convergence behavior of the different methods with a smooth solution.

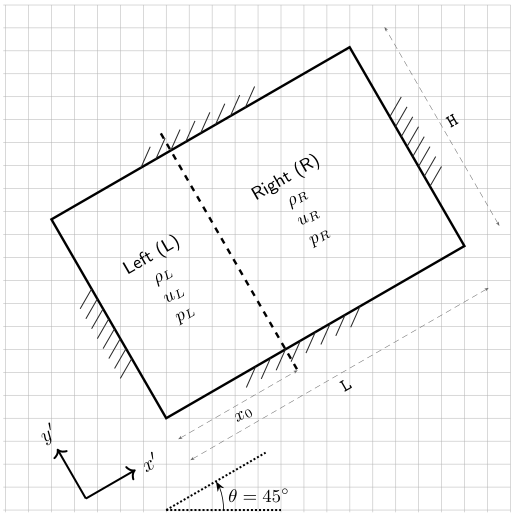


Fig. 6. Geometry of the rotated shock tube test case.

4.1. Shock tube

The test case extends the one presented in Ndiaye et al. (2024). Here, the shock tube is rotated by an angle of 45° to allow the presence of `R-Cells` at the boundary of the domain. The geometry is illustrated in Fig. 6, and the parameters of the test case are given in Table 3. The final time of the simulation is $t_{final} = 0.15$.

Solid Wall boundary conditions are imposed on all the boundaries, ensuring that the total mass within the system remains constant. Fig. 7 compares the numerical solutions for density and velocity profiles obtained with different methods against the analytical solution at the final time.

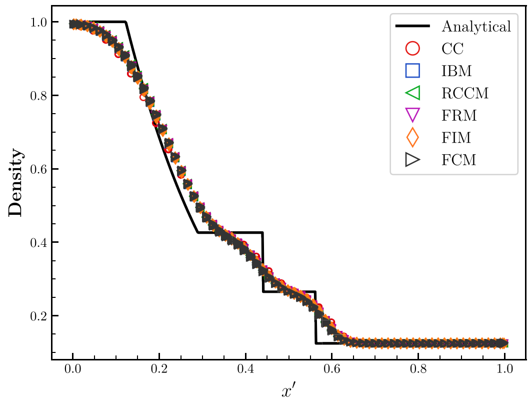
The differences between the methods are very negligible in terms of accuracy, as shown in Fig. 7. However, significant differences arise in terms of conservation, as illustrated in Fig. 8. Figs. 8(a) and 8(b) depict the evolution of the errors on the total mass and energy of the system for the different methods as a function of the average mesh size. The mass and energy conservation errors, along with the average mesh size, are defined by Eqs. (35), (36), and (37), respectively.

$$\text{Mass Conservation Error} = \frac{\left|\sum_{i\in\Omega} \Delta V_i \rho_i - \sum_{i\in\Omega} \Delta V_i \rho_{i,0}\right|}{\sum_{i\in\Omega} \Delta V_i \rho_{i,0}} \tag{35}$$

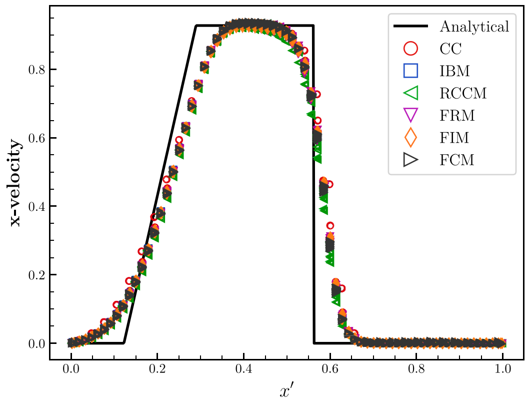
$$\text{Energy Conservation Error} = \frac{\left|\sum_{i\in\Omega} \Delta V_i \rho_i E_i - \sum_{i\in\Omega} \Delta V_i \rho_{i,0} E_{i,0}\right|}{\sum_{i\in\Omega} \Delta V_i \rho_{i,0} E_{i,0}} \tag{36}$$

Table 3
Initial conditions and domain parameters of the shock tube test case.

Left state $[\rho_L,\ u_L,\ p_L]$	Right state $[\rho_R,\ u_R,\ p_R]$	Membrane position x_0	Domain $[L, H]$	Grid cell size $[\Delta x, \Delta y]$
[1.0, 0.0, 1.0]	[0.125, 0.0, 0.1]	0.3	[1.0, 1.0]	[0.029, 0.029]

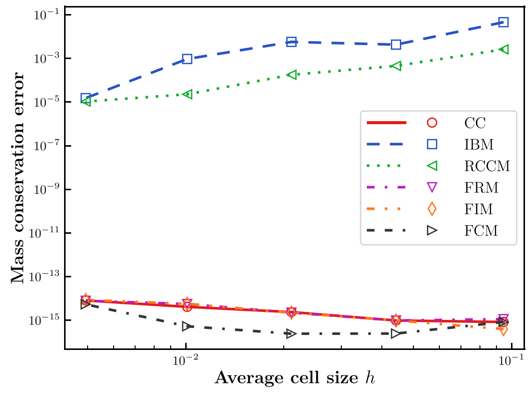


(a) Density profiles at the final time.

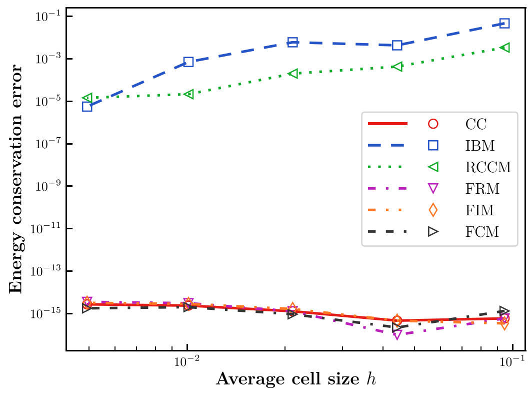


(b) Velocity profiles at the final time.

Fig. 7. Comparison of the numerical solutions for the shock tube test case.



(a) Mass conservation error.



(b) Energy conservation error.

Fig. 8. Comparison of the conservation errors for the shock tube test case.

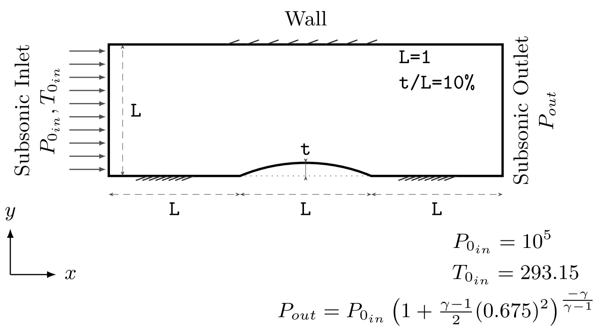

Wall
Subsonic Inlet $P_{0_{in}}, T_{0_{in}}$
Subsonic Outlet P_{out}
L=1
t/L=10%
L
t
L
L
L
y
x
$P_{0_{in}} = 10^5$
$T_{0_{in}} = 293.15$
$P_{out} = P_{0_{in}}\left(1+\frac{\gamma-1}{2}(0.675)^2\right)^{\frac{-\gamma}{\gamma-1}}$

Fig. 9. Geometry and boundary conditions for the transonic flow in a canal with a bump (Ndiaye et al., 2024; Ni, 1982).

$$\text{Average Mesh Size } h = \sqrt{\frac{\sum_{i\in\Omega}\Delta V_i}{\text{Total Number of Cells}}} \tag{37}$$

where $\rho_{i,0}$ and ρ_i are respectively the initial and final density of the cell i, and the same notation is used for the total energy E.

The mass and energy conservation errors for the CC and conservative methods are consistently at machine precision, as expected. The non-conservative methods (IBM and RCCM) exhibit non-zero conservation errors that diminish as the mesh is refined. Since the RCCM method takes into account all cut-cells, its conservation errors are smaller than those of the IBM method, particularly for coarse meshes. As the mesh is refined, the size of the cut-cells ignored by the IBM method becomes negligible, resulting in similar mass and energy conservation errors for both methods. However, the conservation errors of the IBM and RCCM methods do not reach machine precision, even for very fine meshes, due to the reconstruction at the boundaries introducing some conservation errors. For the conservative methods, despite relying on a reconstruction at the boundary, the conservative correction step ensures mass and energy conservation errors at machine precision for all mesh sizes.

The next test case will demonstrate the impact of conservation properties on capturing critical flow features.

4.2. Transonic flow inside a canal with a bump

The second test case involves a transonic flow through a canal featuring a bump with a thickness of 10%. This test case extends also the one presented in Ndiaye et al. (2024). The geometry along with the boundary conditions are illustrated in Fig. 9.

Fig. 10 shows the Mach number contours obtained with the CC method for a fine mesh (last mesh of Table 4). The contours of the

Table 4
Meshes size and properties for the transonic flow over a bump test case.

Mesh	Grid size	Number of uncut cells	Number of cut-cells	Ratio of the cut-cells area [%]	Ratio of the outer cut-cells area [%]	Factor Max/Min of cell area
Extra coarse	150 × 50	5680	364	0.950	2.853	22.807
Coarse	300 × 100	23 722	738	0.922	0.477	110.878
Medium	450 × 150	54 933	1128	0.177	0.678	80.298
Fine	600 × 200	93 883	1472	0.305	1.464	236.641

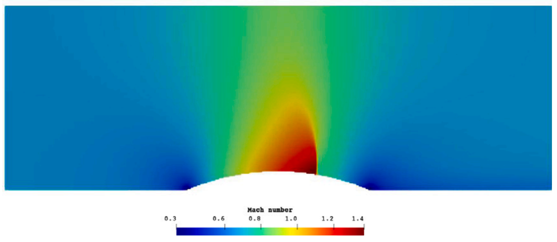


Fig. 10. Mach number contours obtained with the CC method for the transonic flow over bump.

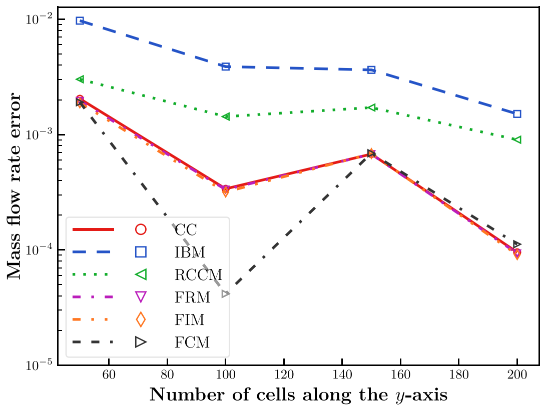


Fig. 11. Mass flow rate error for the transonic flow over a bump test case.

other methods are indistinguishable from those of the CC method and are therefore not shown. The shock on the bump and other important flow features, such as the supersonic pocket, are well captured by all solvers for this mesh.

To evaluate the impact of the conservation properties on the numerical solutions, the mass flow rate and the pressure coefficient along the lower surface of the canal are presented for successive mesh refinements. The properties of the four meshes used are detailed in Table 4. All the meshes have between 0.2% to 1.0% cut-cells, and the canal is positioned on the grid such that the ratio of the area between an uncut cell and the smallest cut-cell ranges from 20 to 1100.

At steady state, the mass flow rate through the canal is constant and is given by Eq. (38) for a surface $x = c$ where c is a constant.

$$\dot{m} = \int_0^L \rho u d y = \sum_{i|\ x_i = c} \rho_i u_i \Delta y_i \tag{38}$$

In Eq. (38), ρ_i and u_i are respectively the density and velocity of the cell i, thus numerically, the mass flow rate is not guaranteed to be constant for different surfaces $x = c$, even if the solver is conservative, since the conserved variables $\mathbf{U}$ are used to compute the mass flow rate not the fluxes $\mathbf{F}$. By evaluating the mass flow rate for two surfaces $x = c_1$ and $x = c_2$ where $c_1 < c_2$, an error on the mass flow rate can be computed as shown in Eq. (39).

$$\text{Mass Flow Rate Error} = \frac{\left|\dot{m}_{c_1} - \dot{m}_{c_2}\right|}{\dot{m}_{c_2}} \tag{39}$$

It is expected that the error on the mass flow rate will diminish as the mesh is refined and that the conservative methods will have a smaller error compared to the non-conservative methods. This is indeed observed in Fig. 11, where the error in the mass flow rate between two surfaces near the inlet and the outlet of the canal is plotted as a function of the number of cells in the y direction. Except for the second mesh, the mass flow rate errors of the conservative methods have the same order of magnitude, and they are smaller than the ones obtained with the non-conservative methods (IBM and RCCM). The error on the mass flow rate of the IBM method is larger than the RCCM method, confirming that the RCCM method is more conservative than the IBM method.

Another way to assess the impact of the conservation properties on the numerical solutions is to examine the pressure coefficient along the lower surface of the canal. Figs. 12(a) to 12(d) show the pressure coefficient along the lower surface of the canal for the four meshes. First, it can be observed that all the methods seem to converge to the same solution as the mesh is refined. Except for the IBM method, the pressure coefficient profiles of the CC method are very similar to those of the other methods, especially the conservative ones (FRM, FIM and FCM). The shock position on the lower surface of the canal obtained with the IBM method is a slightly shifted compared to the other methods, particularly for the coarse meshes. As the mesh is refined, the shock position of the IBM method converges to the shock position obtained with the other methods. The differences in the shock position can be attributed to the non-conservative nature of the IBM method (Brahmachary et al., 2018; Laney, 1998). The shock position of the RCCM method is only slightly shifted compared to the conservative methods. This is understandable since, although the RCCM method is non-conservative, it accounts for all cut-cells, resulting in smaller conservation errors than the IBM method. Consequently, the shock position of the RCCM method is closer to that of the conservative methods. The solutions of the three conservative methods are almost identical to those of the CC method for all meshes.

The next section will evaluate the accuracy of the different methods using a 2D supersonic flow containing discontinuities.

4.3. Confluence of two supersonic flows

Fig. 13 illustrates the configuration of the confluence test case. Two supersonic flows coming from opposite angles are colliding at the center point of the left boundary. The parameters for this problem are given in Table 5. The analytical solution is characterized by three discontinuities: two shocks and a contact discontinuity, defined by their respective angles β_1, β_2, and β_c. These three discontinuities delimit four regions of constant properties. Using the theory of compressible flows (Anderson, 2021), the positions of these discontinuities and the state of the flow in each region can be computed. The analytical solution corresponding to the parameters in Table 5 is provided in Table 6.

To visualize the numerical solutions, a mesh M_0 aligned with the geometry is used, and its properties are listed in Table 7. The vertical alignment of the mesh with the geometry ensures a better cut near the outlet plane. Fig. 14(a) shows the convergence of the density residuals

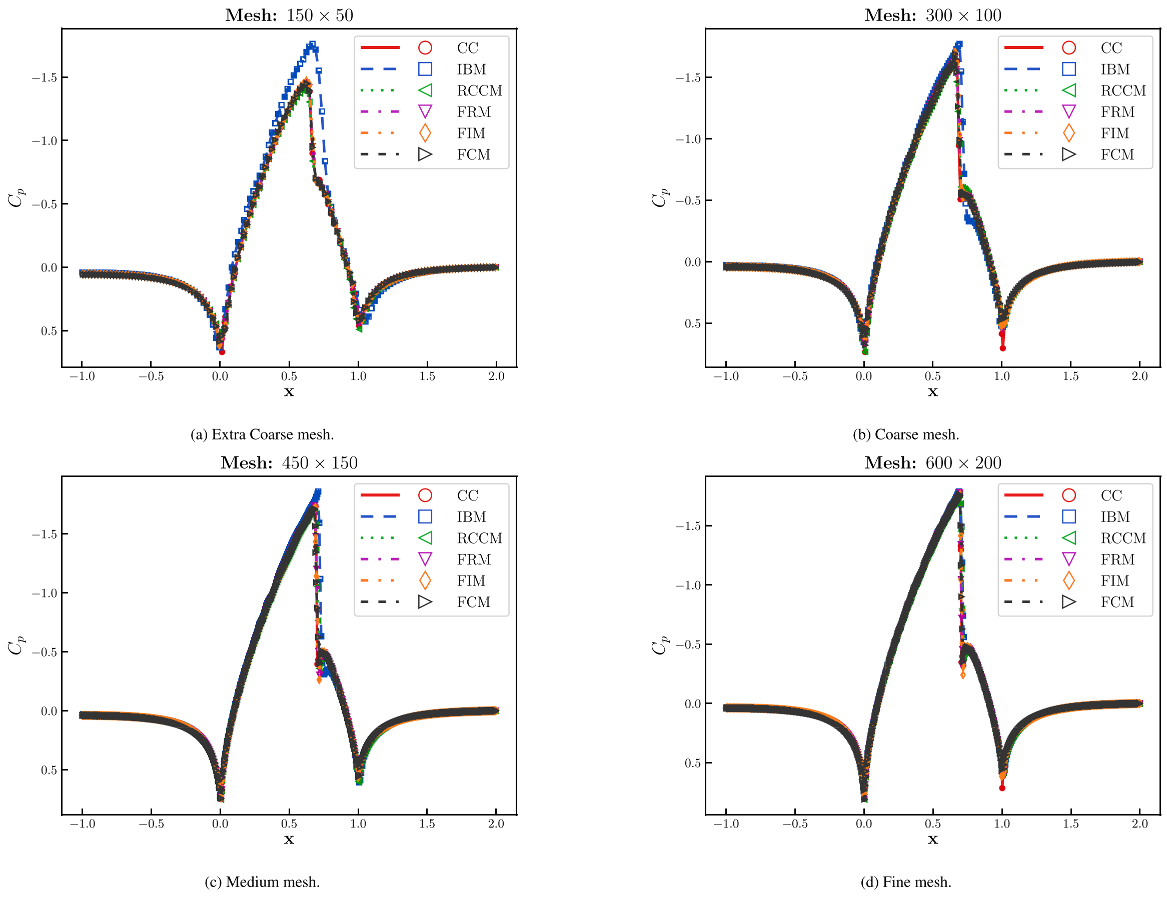


Fig. 12. Pressure coefficient C_p along the lower surface of the canal for the transonic flow over a bump test case.

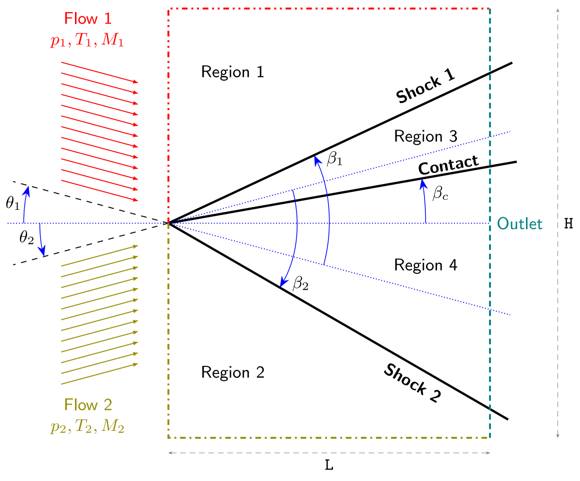


Fig. 13. Geometry and boundary conditions of the confluence test case.

Table 5
Domain and initial conditions of the confluence test case.

Flow 1 $[M_1, p_1, T_1, \theta_1]$	Flow 2 $[M_2, p_2, T_2, \theta_2]$	Domain $[L, H]$
$[1.8, 10^5, 300, 10°]$	$[2.3, 1.5\times10^5, 400, 10°]$	[1, 2]

of all solvers for the mesh M_0. It can be observed that the residuals of all the solvers have successfully converged to the machine precision. Fig. 14(b) displays the Mach number cuts near the outlet plane for the different methods and the analytical solution. Similar to the shock tube test case, the numerical solutions obtained with the different methods are very similar. The shocks and contact discontinuity are captured by all solvers, albeit with some diffusion, which is expected due to the use of a first-order scheme. Overall, the reconstruction procedure and the conservative correction step do not introduce significant errors in the numerical solutions.

To assess the accuracy of the different methods, five meshes are used. These meshes are generated using the geometry shown in Fig. 13, but rotated by 45° to introduce arbitrary reconstructed cells at the domain boundaries. The properties of these meshes are listed in Table 8.

Fig. 15 shows the convergence of the density with the mesh refinement for the different methods. The L_1 and L_2 error norms used to compute the convergence rate of a field ϕ are defined in Eqs. (40) and (41).

$$\epsilon_{L_1} = \sum_{i\in\Omega} \frac{\Delta V_i}{V} \left|\phi_i - \phi_{i,\text{exact}}\right| \tag{40}$$

$$\epsilon_{L_2} = \sqrt{\sum_{i\in\Omega} \frac{\Delta V_i}{V} \left(\phi_i - \phi_{i,\text{exact}}\right)^2} \tag{41}$$

where ϕ_i and $\phi_{i,\text{exact}}$ are respectively the numerical and analytical values of the field ϕ at the cell i, and $V = \sum_{i\in\Omega} \Delta V_i$ denotes the total volume of the domain.

The convergence rates are computed using Eq. (42).

$$\text{convergence rate} = \frac{\log\left(\frac{\epsilon_{h_2}}{\epsilon_{h_1}}\right)}{\log\left(\frac{h_2}{h_1}\right)} \tag{42}$$

where ϵ_{h_1} and ϵ_{h_2} are the error norms obtained with the meshes with sizes h_1 and h_2, respectively. The mesh size h is defined by Eq. (37).

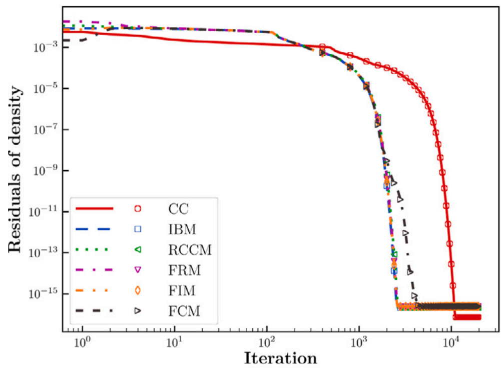
All methods yield similar results, with convergence rates around 0.3 and 0.6 for the L_2 and L_1 norms, respectively, as shown in Fig. 15.

Table 6
Analytical solution for the confluence test case.

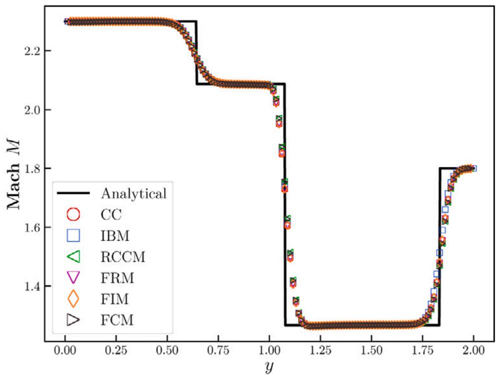
Discontinuities $[\beta_1, \beta_2, \beta_c]$	Region 3 $[M_3, p_3, T_3]$	Region 4 $[M_4, p_4, T_4]$
$[50.523°, 30.152°, 4.528°]$	$[1.266, 2.086 \times 10^5, 374.365]$	$[2.087, 2.086 \times 10^5, 439.958]$

Table 7
Properties of the mesh used for visualizing the numerical solutions of the confluence test case.

Mesh	Grid size	Number of uncut cells	Number of cut-cells	Ratio of the cut-cells area [%]	Ratio of the outer cut-cells area [%]	Factor max/min of cell area
M_0	80 × 160	10 368	436	1.3465	0.956	11.243



(a) Convergence of the density residuals.



(b) Cut of the mach number near the outlet.

Fig. 14. Comparison of the different methods for the confluence test case.

Table 8
Mesh sizes and properties for the convergence study of the confluence test case.

Mesh	Grid size	Number of uncut cells	Number of cut-cells	Ratio of the cut-cells area [%]	Ratio of the outer cut-cells area [%]	Factor max/min of cell area
M_1	25 × 25	175	88	3.392	16.726	23.469
M_2	50 × 50	812	178	1.102	8.387	10.302
M_3	100 × 100	3422	359	1.147	4.538	11.947
M_4	200 × 200	12 656	683	0.512	2.359	130.641
M_5	400 × 400	56 525	1436	0.395	1.197	70.201

Table 9
Convergence rates of the different methods for the confluence test case.

	ρ		M		p		T	
	L_1	L_2	L_1	L_2	L_1	L_2	L_1	L_2
CC	0.58	0.29	0.51	0.25	0.66	0.30	0.54	0.26
IBM	0.60	0.35	0.51	0.25	0.70	0.41	0.53	0.26
RCCM	0.58	0.33	0.52	0.25	0.71	0.45	0.52	0.23
FRM	0.58	0.28	0.51	0.25	0.66	0.30	0.54	0.26
FIM	0.58	0.28	0.51	0.25	0.66	0.30	0.54	0.26
FCM	0.62	0.35	0.54	0.25	0.76	0.47	0.56	0.24

Similar convergence rates are observed for other fields such as pressure and Mach number, as detailed in Table 9. The expected convergence rate for a first-order scheme in smooth flows is 1 for all norms. However, the presence of discontinuities in the solution explains why the convergence rate is not close to 1 (Delarue et al., 2017).

The next section will employ a smooth solution to more accurately compute the convergence of the different methods.

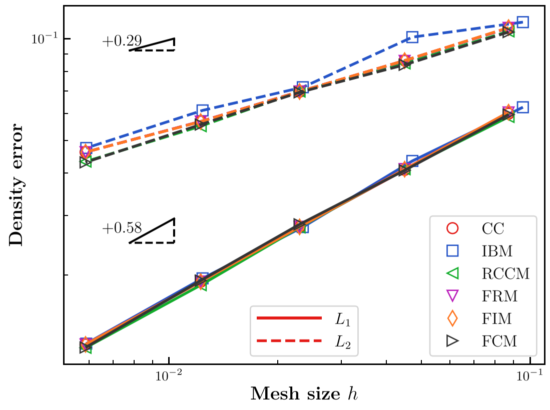


Fig. 15. Convergence of the density with the mesh refinement for the confluence test case.

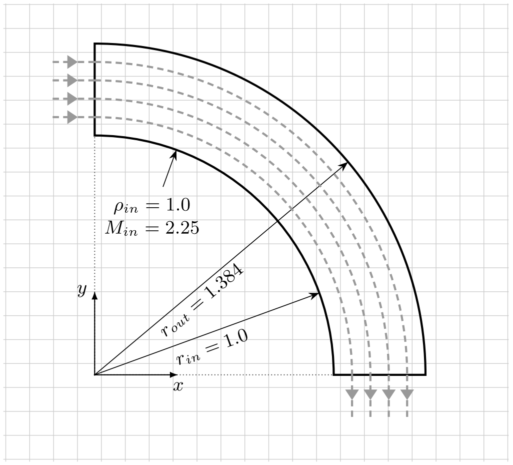


Fig. 16. Geometry of the vortex test case.

4.4. *Supersonic vortex flow between two concentric quarter circles*

The final test case examines a supersonic vortex flow between two concentric quarter circles, a test extensively used in the literature to evaluate the accuracy of numerical methods for inviscid compressible flows (Aftosmis et al., 1995; Krivodonova and Berger, 2006; Berger and Giuliani, 2021). The geometry is shown in Fig. 16, and the analytical solution is provided in Eq. (43).

$$\begin{aligned} &\rho = \rho_{in}\left[1 + \frac{\gamma-1}{2}M_{in}^2\left(1 - \frac{r_{in}^2}{r^2}\right)\right]^{\frac{1}{\gamma-1}} \\ &p = \frac{\rho^\gamma}{\gamma} \qquad u = |\mathbf{v}|\cos\theta \qquad v = |\mathbf{v}|\sin\theta \\ &r|\mathbf{v}| = r_{in}|\mathbf{v}_{in}| \quad \text{where } |\mathbf{v}_{in}| = M_{in}\left(\rho_{in}^{\frac{\gamma-1}{2}}\right) \end{aligned} \tag{43}$$

where ρ_{in} and M_{in} are respectively the density and mach number at the inner radius r_{in}, and are chosen to be 1.0 and 2.25, respectively.

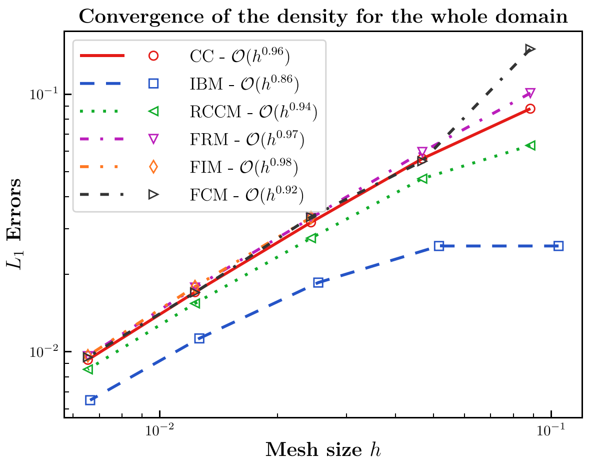
To obtain the numerical solutions, the analytical solution is imposed as Dirichlet boundary condition on all the boundaries and used as initial condition as well. Five uniform grids, ranging from 16×16 to 256×256 cells, are employed. After the convergence of the residuals of all the methods to the machine precision, the error norms (L_1, L_2 and L_∞) and convergence rates of the density and the pressure are computed using Eqs. (40), (41), (44) and (42), respectively.

$$\epsilon_{L_\infty} = \max_{i\in\Omega}\left|\phi_i - \phi_{i,\text{exact}}\right| \tag{44}$$

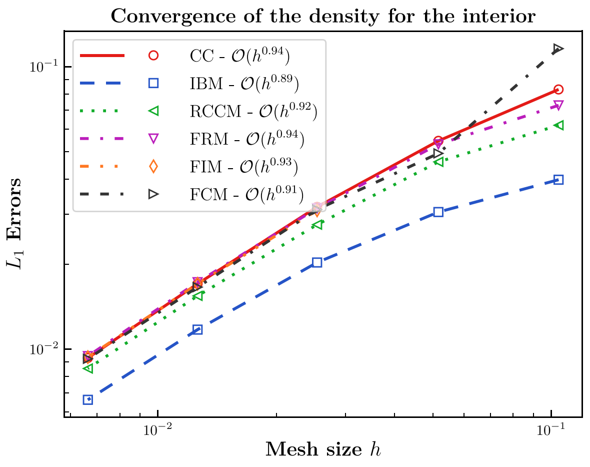
Also, to measure the impact of the reconstruction procedure and the conservative correction step on the convergence of the numerical solution, the convergence rates are computed on three set of cells: the entire domain (W), only the cells strictly inside the domain (I), and only the cells on the boundary of the domain (B).

Figs. 17(a) to 17(c) illustrate the convergence of the density using the L_1 norm for the whole domain, the interior, and the boundary, respectively. Tables 10 and 11 present the convergence rates of the density and the pressure for the different methods across the different regions.

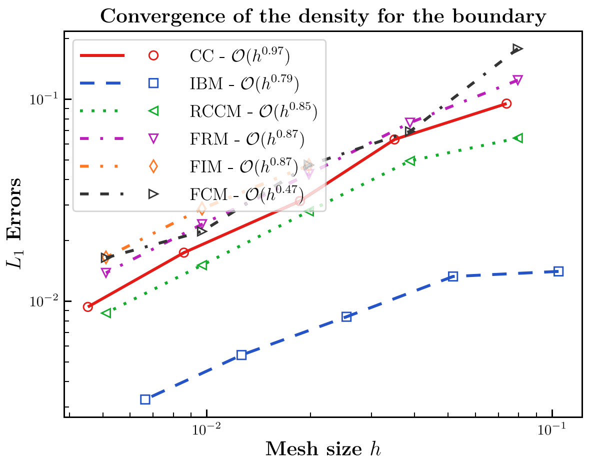
Aside from a few exceptions, all methods exhibit convergence rates close to the expected rate of 1 for all three regions, consistent with the first-order scheme employed in this study. For the L_1 and L_2 norms, the convergence rates of the CC, RCCM, and conservative methods are nearly identical across the entire domain and its interior. This similarity arises because the cut-cells are very small, resulting in minimal L_1 and L_2 norm errors associated with them. However, as shown in Tables 10 and 11, the convergence rates degrade slightly for all three



(a) Whole domain.



(b) Interior domain.



(c) Boundary domain.

Fig. 17. Convergence of the density for the vortex test case using the L_1 norm.

regions when considering the L_∞ norm. The boundary of the domain is particularly affected by this degradation, especially for the conservative method. This degradation is partly due to the reconstruction procedure (used in RCCM and conservative methods) and partly due to the conservative correction step (specific to the conservative method), which can introduce discrepancies in the solution compared to the CC method, particularly near the domain boundaries. The FCM method

Table 10
Convergence rates of the density for the different methods and regions.

	L_1			L_2			L_∞		
	W	I	B	W	I	B	W	I	B
CC	0.96	0.94	0.97	0.96	0.94	0.95	0.91	0.89	0.88
IBM	0.86	0.89	0.79	0.86	0.88	0.77	0.75	0.75	0.69
RCCM	0.94	0.92	0.85	0.94	0.92	0.84	0.86	0.84	0.85
FRM	0.97	0.94	0.87	0.98	0.94	0.83	0.73	0.85	0.72
FIM	0.98	0.93	0.87	1.00	0.93	0.85	0.67	0.89	0.66
FCM	0.92	0.91	0.47	0.83	0.91	0.02	−1.55	0.87	−1.53

Table 11
Convergence rates of the pressure for the different methods and regions.

	L_1			L_2			L_∞		
	W	I	B	W	I	B	W	I	B
CC	0.95	0.92	0.91	0.95	0.93	0.92	0.94	0.92	0.93
IBM	0.90	0.93	0.87	0.89	0.91	0.83	0.84	0.84	0.81
RCCM	0.94	0.91	0.90	0.94	0.92	0.90	0.96	0.94	0.93
FRM	0.98	0.93	0.88	1.01	0.94	0.89	0.91	0.91	0.90
FIM	0.98	0.92	0.83	0.99	0.93	0.80	0.64	0.92	0.63
FCM	0.95	0.88	0.83	0.99	0.90	0.88	0.61	0.90	0.60

appears to be the most affected near the domain boundary. For the IBM method, the convergence rates are somewhat lower compared to the other methods, although its errors at the boundary are relatively smaller. The small errors at the boundary arise from the fact that the boundary cells employed in the IBM method are of high quality and comparable in size to the interior cells.

5. Conclusion

This paper presents three distinct methods designed to restore the conservation properties of the Immersed Boundary Method using a cut-cell approach. All methods are based on a semi-implicit reconstruction procedure that mitigates the small cell problem inherent in classical cut-cell methods. The first method (FRM) employs a flux redistribution technique to ensure the global conservation of all conserved quantities (mass, momentum, and energy) across the entire domain. The second (FIM) and third (FCM) methods utilize optimization procedures to guarantee the local conservation of these quantities within each cell. These methods have been rigorously tested on various test cases, demonstrating results comparable to the cut-cell method in terms of accuracy, convergence, and conservation. The numerical results demonstrate that all three conservative methods (FRM, FIM, and FCM) effectively restore the conservation properties while maintaining accuracy and stability. The shock tube test case shows that the conservative methods achieve mass and energy conservation errors at machine precision, unlike the non-conservative methods (IBM and RCCM). In the transonic flow over a bump test case, the conservative methods exhibit smaller mass flow rate errors, and they capture the shock wave more accurately than the non-conservative methods. Finally, the confluence of two supersonic flows and the supersonic vortex flow test cases confirm that the conservative methods achieve convergence rates close to the expected rate for all norms, and the results are consistent with the non-conservative methods.

In summary, the three methods (FRM, FIM, and FCM) share the common goal of restoring conservation properties in the numerical solutions by modifying the fluxes at the cut-cell boundaries, which is why they all give results that are almost identical to the cut-cell method. However, they differ in their approach to achieving conservation. The FRM method is computationally the most efficient and the simplest to implement, but it only guarantees global conservation and does not ensure local conservation within each cell. The FIM method is generally conservative and stable, but in rare cases, it may not be fully conservative. It also requires a more complex optimization procedure, making it more computationally expensive than the FRM method. The FCM method is always locally conservative but may be less stable than the FIM method due to the perturbation of reconstructed solutions. For a general case, the FRM method is recommended due to its simplicity and efficiency in ensuring global conservation. However, for more complex cases where local conservation is critical, the FIM or FCM methods may be preferred despite their higher computational cost.

The three methods are all based on a flux formulation that ensures conservation properties in the numerical solutions, making them straightforward to implement in three-dimensional simulations. In the context of viscous flows, the proposed methods can also be extended to incorporate the additional terms related to the viscous fluxes. These viscous terms do not affect the implementation of the proposed methods, as all methods are based on a flux formulation that can be easily extended to include viscous terms. Also, the inclusion of viscous terms introduces some dissipation to the solution, which can enhance the stability of the numerical methods. However, the reconstruction procedure employed in the three methods may impact the resolution of the boundary layer, necessitating further studies to investigate its effects on viscous flow simulations. Additionally, the Cartesian grid used in the proposed methods may not be suitable for resolving the boundary layer in viscous flows. Alternative grid configurations, such as a hybrid grid combining a body-fitted grid near the immersed boundary with a Cartesian grid in the rest of the domain, may be required to accurately capture viscous effects. These considerations highlight the complexity of extending the proposed methods to viscous flows, necessitating further research and investigation.

Future work will focus on developing second-order schemes using these methods and testing them on more complex inviscid test cases to further assess their accuracy and robustness.

CRediT authorship contribution statement

El Hadji Abdou Aziz Ndiaye: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Jean-Yves Trépanier:** Writing – review & editing, Visualization, Supervision, Resources, Project administration, Funding acquisition, Conceptualization. **Renan De Holanda Sousa:** Writing – review & editing, Supervision, Conceptualization. **Sébastien Leclaire:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this manuscript, the authors utilized GitHub Copilot to assist in writing and generating figures. Following the use of this tool, the authors thoroughly reviewed and edited the content as necessary and assume full responsibility for the final publication.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: EL HADJI ABDOU AZIZ NDIAYE reports financial support was provided by GE Vernova. Renan De Holanda Sousa reports a relationship with GE Vernova that includes: employment. The other 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.

Acknowledgment

The authors would like to thank GE Vernova for the financial support provided for this research.

Data availability

Data will be made available on request.

References

Aftosmis, M., Gaitonde, D., Tavares, T.S., 1995. Behavior of linear reconstruction techniques on unstructured meshes. AIAA J. 33 (11), 2038–2049. http://dx.doi.org/10.2514/3.12945.

Anderson, J., 2021. Modern compressible flow: With historical perspective. McGraw-Hill Series in Aeronautical and Aerospace Engineering, McGraw-Hill Education.

Bell, J.B., Welcome, M.L., Colella, P., 1991. Conservative front-tracking for inviscid compressible flow. In: 10th Computational Fluid Dynamics Conference. http://dx.doi.org/10.2514/6.1991-1599, arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.1991-1599, URL: https://arc.aiaa.org/doi/abs/10.2514/6.1991-1599.

Berger, M., 2017. Chapter 1 - cut cells: Meshes and solvers. In: Abgrall, R., Shu, C.-W. (Eds.), Handbook of Numerical Methods for Hyperbolic Problems. In: Handbook of Numerical Analysis, vol. 18, Elsevier, pp. 1–22. http://dx.doi.org/10.1016/bs.hna.2016.10.008.

Berger, M., Giuliani, A., 2021. A state redistribution algorithm for finite volume schemes on cut cell meshes. J. Comput. Phys. 428, 109820. http://dx.doi.org/10.1016/j.jcp.2020.109820.

Berger, M., Helzel, C., 2012. A simplified h-box method for embedded boundary grids. SIAM J. Sci. Comput. 34 (2), A861–A888. http://dx.doi.org/10.1137/110829398.

Blazek, J., 2015. Computational fluid dynamics: principles and applications. Butterworth-Heinemann.

Brahmachary, S., Natarajan, G., Kulkarni, V., Sahoo, N., 2018. A sharp-interface immersed boundary framework for simulations of high-speed inviscid compressible flows. Internat. J. Numer. Methods Fluids 86 (12), 770–791. http://dx.doi.org/10.1002/fld.4479.

Chern, I.-L., Colella, P., 1987. A Conservative Front Tracking Method for Hyperbolic Conservation Laws. LLNL Rep. No. UCRL-97200,, Vol. 51, Lawrence Liver-more National Laboratory, pp. 83–110.

Clarke, D.K., Salas, M., Hassan, H., 1986. Euler calculations for multielement airfoils using cartesian grids. AIAA J. 24 (3), 353–358. http://dx.doi.org/10.2514/3.9273.

Colella, P., Graves, D.T., Keen, B.J., Modiano, D., 2006. A cartesian grid embedded boundary method for hyperbolic conservation laws. J. Comput. Phys. 211 (1), 347–366. http://dx.doi.org/10.1016/j.jcp.2005.05.026.

Delarue, F., Lagoutière, F., Vauchelet, N., 2017. Convergence order of upwind type schemes for transport equations with discontinuous coefficients. J. Mathématiques Pures Appliquées 108 (6), 918–951. http://dx.doi.org/10.1016/j.matpur.2017.05.012.

Ghias, R., Mittal, R., Dong, H., 2007. A sharp interface immersed boundary method for compressible viscous flows. J. Comput. Phys. 225 (1), 528–553. http://dx.doi.org/10.1016/j.jcp.2006.12.007.

Hartmann, D., Meinke, M., Schröder, W., 2008. An adaptive multilevel multigrid formulation for cartesian hierarchical grid methods. Comput. Fluids 37 (9), 1103–1125. http://dx.doi.org/10.1016/j.compfluid.2007.06.007.

Hartmann, D., Meinke, M., Schröder, W., 2011. A strictly conservative cartesian cut-cell method for compressible viscous flows on adaptive grids. Comput. Methods Appl. Mech. Engrg. 200 (9), 1038–1052. http://dx.doi.org/10.1016/j.cma.2010.05.015.

Helzel, C., Berger, M.J., Leveque, R.J., 2005. A high-resolution rotated grid method for conservation laws with embedded geometries. SIAM J. Sci. Comput. 26 (3), 785–809. http://dx.doi.org/10.1137/S106482750343028X.

Iaccarino, G., Verzicco, R., 2003. Immersed boundary technique for turbulent flow simulations. Appl. Mech. Rev. 56 (3), 331–347. http://dx.doi.org/10.1115/1.1563627.

Kirkpatrick, M.P., Armfield, S.W., Kent, J.H., 2003. A representation of curved boundaries for the solution of the Navier-Stokes equations on a staggered three-dimensional cartesian grid. J. Comput. Phys. 184 (1), 1–36. http://dx.doi.org/10.1016/S0021-9991(02)00013-X.

Krivodonova, L., Berger, M., 2006. High-order accurate implementation of solid wall boundary conditions in curved geometries. J. Comput. Phys. 211 (2), 492–512. http://dx.doi.org/10.1016/j.jcp.2005.05.029.

Laney, C.B., 1998. Computational Gasdynamics. Cambridge University Press.

LeVeque, R.J., 1992. Numerical methods for conservation laws. vol. 214, Springer, http://dx.doi.org/10.1007/978-3-0348-8629-1.

May, S., Berger, M., 2017. An explicit implicit scheme for cut cells in embedded boundary meshes. J. Sci. Comput. 71 (3), 919–943. http://dx.doi.org/10.1007/s10915-016-0326-2.

Mittal, R., Dong, H., Bozkurttas, M., Najjar, F., Vargas, A., von Loebbecke, A., 2008. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. J. Comput. Phys. 227 (10), 4825–4852. http://dx.doi.org/10.1016/j.jcp.2008.01.028.

Mittal, R., Iaccarino, G., 2005. Immersed boundary methods. Annu. Rev. Fluid Mech. 37 (1), 239–261. http://dx.doi.org/10.1146/annurev.fluid.37.061903.175743.

Mohd-Yusof, J., 1997. Combined immersed-boundary/B-spline methods for simulations of flow in complex geometries. Cent. Turbul. Res. Annu. Res. Briefs 161 (1), 317–327.

Ndiaye, E.H.A.A., Trépanier, J.-Y., Sousa, R.D.H., Leclaire, S., 2024. Improvement of the conservation properties of the immersed boundary method for inviscid compressible flows using cut-cells. In: CSME/CFDSC (Ed.), The Canadian Society of Mechanical Engineering and Computational Fluid Dynamics Canada International Congress 2024.

Ni, R.-H., 1982. A multiple-grid scheme for solving the Euler equations. AIAA J. 20 (11), 1565–1571. http://dx.doi.org/10.2514/6.1981-1025.

Pember, R.B., Bell, J.B., Colella, P., Curtchfield, W.Y., Welcome, M.L., 1995. An adaptive cartesian grid method for unsteady compressible flow in irregular regions. J. Comput. Phys. 120 (2), 278–304. http://dx.doi.org/10.1006/jcph.1995.1165.

Peskin, C.S., 1972. Flow patterns around heart valves: A numerical method. J. Comput. Phys. 10 (2), 252–271. http://dx.doi.org/10.1016/0021-9991(72)90065-4.

Roe, P.L., 1981. Approximate Riemann solvers, parameter vectors, and difference schemes. J. Comput. Phys. 43 (2), 357–372. http://dx.doi.org/10.1016/0021-9991(81)90128-5.

Sod, G.A., 1978. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. J. Comput. Phys. 27 (1), 1–31. http://dx.doi.org/10.1016/0021-9991(78)90023-2.

Sotiropoulos, F., Yang, X., 2014. Immersed boundary methods for simulating fluid-structure interaction. Prog. Aerosp. Sci. 65, 1–21. http://dx.doi.org/10.1016/j.paerosci.2013.09.003.

Udaykumar, H., Mittal, R., Rampunggoon, P., Khanna, A., 2001. A sharp interface cartesian grid method for simulating flows with complex moving boundaries. J. Comput. Phys. 174 (1), 345–380. http://dx.doi.org/10.1006/jcph.2001.6916.

Ye, T., Mittal, R., Udaykumar, H., Shyy, W., 1999. An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries. J. Comput. Phys. 156 (2), 209–240. http://dx.doi.org/10.1006/jcph.1999.6356.