Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 214 (2006) 618–632

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A hybrid Cartesian grid and gridless method for compressible flows

Hong Luo [a,*], Joseph D. Baum [a], Rainald Löhner [b]

[a] *Center for Applied Computational Sciences, Science Applications International Corporation, 1710 SAIC Drive, MS 2-6-9, McLean, VA 22102, USA*

[b] *Institute for Computational Sciences and Informatics, George Mason University, Fairfax, VA 22030, USA*

Received 7 January 2005; received in revised form 3 October 2005; accepted 6 October 2005
Available online 22 November 2005

Abstract

A hybrid Cartesian grid and gridless method is presented to compute unsteady compressible flows for complex geometries. In this method, a Cartesian grid is used as baseline mesh to cover the computational domain, while the boundary surfaces are addressed using a gridless method. This hybrid method combines the efficiency of a Cartesian grid method and the flexibility of a gridless method for the complex geometries. The developed method is used to compute a number of test cases to validate the accuracy and efficiency of the method. The numerical results obtained indicate that the use of this hybrid method leads to a significant improvement in performance over its unstructured grid counterpart for the time-accurate solution of the compressible Euler equations. An overall speed-up factor of about eight and a saving in storage requirements about one order of magnitude for a typical three-dimensional problem in comparison with the unstructured grid method are demonstrated.

Keywords: Compressible flow; Euler equations; Cartesian grid method; Gridless methods

1. Introduction

Over the course of last decade, significant progress has been made on developing numerical methods for the solution of the compressible Euler and Navier–Stokes equations. In general, these numerical methods can be classified by the mesh they use as structured grid methods, unstructured grid methods, Cartesian grid methods, and gridless methods. Each of these methods is advocated, promoted, developed, and used by their respective supporters. Since each method has its own advantages and disadvantages, the answer to which method is preferable depends on the problem to be solved. The structured grid methods [1–3] have a disadvantage in mesh generation for complex geometries. The main advantage of the unstructured grid methods [4–6] is the ease of grid generation for complex configurations. However, the computational costs

[*] Corresponding author. Tel.: +1 703 676 5957; fax: +1 703 676 5323.
E-mail address: hong.luo@saic.com (H. Luo).

0021-9991/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2005.10.002

and memory requirements are generally higher than their structured grid counterparts. The advantages of the Cartesian grid methods [7–9] include ease of grid generation, lower computational storage requirements, and significantly less operational count per cell. However, the main challenge in using Cartesian methods is how to deal with arbitrary boundaries, as the grids are not body-aligned. The cells of a Cartesian mesh near the body can extend through surfaces of boundaries. Accurate means of representing boundary conditions in cells that intersect surfaces are essential for successful Cartesian methods. Lately, gridless methods [10–15] came into focus. This class of methods is essentially propelled by the fact that there exists a perceived difficulty in generating volume filling grids for complex geometries in spite of significant progress in the theory and practice of mesh generation over the last decade. However, global conservation of mass, momentum, and energy for these methods is not necessarily assured. Furthermore, these gridless methods are generally slower than their mesh-based counterparts.

The objective of the efforts presented in this paper is to develop a fast and efficient numerical method for time-accurate computation of the compressible Euler equations using a hybrid Cartesian grid and gridless approach. The development of such a numerical method is strongly motivated by the need to be able to simulate blast and shock waves for complex geometries on a personal computer platform in a reasonable time and accuracy. Simulation of high explosive detonation, blast propagation, and shock wave diffraction plays an important role in determining and assessing target vulnerability and weapon lethality. This type of hybrid methods [16–18] was first developed by Liu et al. mainly to simplify grid generation for complex geometries and has been successfully applied to compute steady aerodynamics flows for 2D configurations. In this paper, this hybrid method is extended and modified to time-accurate simulation of shock waves for 3D configurations in order to realize its benefit and potential in terms of computational efficiency. The idea behind this hybrid method is to combine the advantages of both gridless and Cartesian grid methods in an attempt to develop a fast, low storage method for complex geometries. The method uses gridless method to address the boundary or interface while a standard Cartesian grid method is used elsewhere. Since a majority of computational cells are solved using the Cartesian grid method, the resulting method is very efficient in terms of both computational costs and storage requirements. As the gridless method is used to obtain the solution for solid boundary points, the developed method can easily be used for complex geometries without a need to generate a body-filled mesh. A variety of test cases is performed to demonstrate the accuracy, efficiency, robustness and versatility of the developed hybrid method. The computed results are in good agreement with those obtained using a cell-vertex finite volume method and with available analytical data. A quite demonstrative test case indicates that the use of this hybrid method leads to a significant improvement in performance over its unstructured grid counterpart for shock interaction problems in terms of both computational costs and memory requirements.

2. Governing equations

The Euler equations governing unsteady compressible inviscid flows can be expressed in conservative form as

$$\frac{\partial \mathbf{U}}{\partial t}+\frac{\partial \mathbf{F}}{\partial x}+\frac{\partial \mathbf{G}}{\partial y}+\frac{\partial \mathbf{H}}{\partial z}=0, \tag{2.1}$$

where the conservative state vector **U** and the inviscid flux vectors **F**, **G**, **H** are defined by

$$\mathbf{U}=\begin{pmatrix}\rho\\ \rho v_x\\ \rho v_y\\ \rho v_z\\ \rho e\end{pmatrix},\quad \mathbf{F}=\begin{pmatrix}\rho v_x\\ \rho v_x^2+p\\ v_x\rho v_y\\ v_x\rho v_z\\ v_x(\rho e+p)\end{pmatrix},\quad \mathbf{G}=\begin{pmatrix}\rho v_y\\ \rho v_y v_x\\ \rho v_y^2+p\\ \rho v_y v_z\\ v_y(\rho e+p)\end{pmatrix},\quad \mathbf{H}=\begin{pmatrix}\rho v_z\\ \rho v_z v_x\\ \rho v_z v_y\\ \rho v_z v_z+p\\ v_z(\rho e+p)\end{pmatrix}, \tag{2.2}$$

where ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively, and v_x, v_y, and v_z are the velocity components of the flow in the coordinate direction x, y, z, respectively. This set of equations is completed by the addition of the equation of state

$$p = (\gamma - 1)\rho\left(e - \frac{1}{2}(v_x^2 + v_y^2 + v_z^2)\right), \tag{2.3}$$

which is valid for perfect gas, where γ is the ratio of the specific heats.

3. Numerical method

The basic idea behind the present method is to apply a gridless method to cells in the vicinity of a solid boundary and a conventional Cartesian grid method to all other cells in an attempt to develop a fast, low storage method for time-accurate computation of the unsteady Euler equations for complex geometries. In this method, body geometries are first defined and represented by a set of boundary points. These boundary points are chosen as the center points of triangles, which can be, for example, obtained by triangulation of the body geometries. Note that such chosen points have a well-defined normal direction, therefore, avoiding ambiguity of normal definitions for surface singularities such as the tip of an airfoil trailing edge [16–18]. A Cartesian grid can then be generated whose cell size is consistent with the size for the triangulation of the body geometries, which ensures the success of the method as a whole.

3.1. Determination of gridless points

As the solid boundary points are immersed in the baseline Cartesian grid, three types of nearby cells can be identified as shown in Fig. 1. Cartesian cells are cells whose solutions can be computed using a Cartesian grid method, since the values of all their neighbors are known and the stencil for the Cartesian grid method is complete. Cut-off cells are cells that contain one or more boundary points or embed within a body. These cut-off cells are omitted from the computation. Transitional cells are cells that have cut-off cells as its neighbors. The solution for these cells and the boundary points is obtained using a gridless method. An unstructured data structure is used to manage the solution of these transitional cells and boundary points, which will be termed gridless points hereafter. The procedure for the determination of cut-off, Cartesian, and transitional cells is simple and straightforward, which is listed as follows:

1. All the cells are first marked as Cartesian cells.
2. The cells which have at least one node out of computational domain are marked as cut-off cells. This can be readily achieved using a ray casting algorithm based on either alternating digital tree or bin data structures.

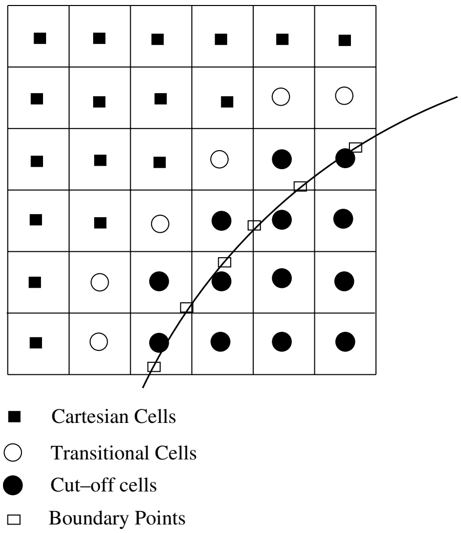


Fig. 1. Classification of grid cells.

3. The neighboring cells to the cut-off cells are then marked as transitional cells.

3.2. Determination of cloud points for gridless points

A gridless method requires a local cloud of neighboring points. For each gridless point, a host Cartesian cell, which contains this point, is first determined. A cloud of 27 cells for this point are then chosen using $3 \times 3 \times 3$ stencil. Among these 27 cells, those cut-off cells will be removed and replaced by the gridless points they contain, if these gridless points do exist. For a boundary point, those cells located in the opposite direction of this point will also removed in order to avoid choosing cells from the wrong side of the profile at thin surfaces. Furthermore, a post-check is performed to ensure that there are enough stencils for any gridless points. In case that there are insufficient stencils to guarantee existence of the solution to the least-squares approximations, more neighboring Cartesian cells are added to the local clouds.

3.3. Gridless method

For all the gridless points, the governing equation (2.1) is discretized using a gridless method. In recent years, significant progress has been made in developing gridless methods for the compressible Euler and Navier–Stokes equations [10–15]. Gridless methods are attractive due to the fact that they are free of generating a computational mesh, which is still a daunting challenge for complex geometries. Almost all of gridless methods make use of a least-square formulation and they differ from one another in the way they introduce artificial dissipations which are essential and necessary for the governing hyperbolic equations. In the present work, the dual least-squares approximation, a variant of the least-squares approximation, is used to compute the inviscid fluxes. This dual least-squares approximation provides a natural way to use upwind-type discretization for the inviscid fluxes of the Euler equations. Assume that C_i is the set of cloud points for a given point i. Let f_{ij} denote the value of any function f at the mid-point of the edge $\vec{ij}$, where $j \in C_i$. Assuming that this function varies linearly along an edge $\vec{ij}$ and using Taylor's formula about the point i, one obtains

$$\left.\frac{\partial f}{\partial x}\right|_i (x_{ij} - x_i) + \left.\frac{\partial f}{\partial y}\right|_i (y_{ij} - y_i) + \left.\frac{\partial f}{\partial z}\right|_i (z_{ij} - z_i) = f_{ij} - f_i. \tag{3.1}$$

Similar equations could be written for all cloud points associated with point i, subject to an arbitrary weighting factor w_i. This yields the following non-square matrix:

$$\begin{pmatrix} w_1(x_{i1} - x_i) & w_1(y_{i1} - y_i) & w_1(z_{i1} - z_i) \\ \vdots & \vdots & \vdots \\ w_n(x_{in} - x_i) & w_1(y_{in} - y_i) & w_1(z_{in} - z_i) \end{pmatrix} \begin{pmatrix} \frac{\partial f}{\partial x}|_i \\ \frac{\partial f}{\partial y}|_i \\ \frac{\partial f}{\partial z}|_i \end{pmatrix} = \begin{pmatrix} w_1(f_{i1} - f_i) \\ \vdots \\ w_n(f_{in} - f_i) \end{pmatrix}, \tag{3.2}$$

where n is the number of cloud points for the point i. This formulation provides a freedom in the choice of weighting coefficients w_j. These weighting coefficients can be selected as a function of the geometry and/or solution. Classical approximations in one dimension can be recovered by choosing geometrical weights of the form $w_j = 1.0/|\mathbf{r_{ij}}-\mathbf{r_i}|^t$ for values of $t = 0$, 1, 2. The numerical computations shown in the following section are performed using $t = 0$.

The spatial derivatives of the function f can then be obtained by solving Eq. (3.2) using the least-squares method as

$$\left.\frac{\partial f}{\partial x}\right|_i = \sum_{j \in C_i} a_{ij}(f_{ij} - f_i), \quad \left.\frac{\partial f}{\partial y}\right|_i = \sum_{j \in C_i} b_{ij}(f_{ij} - f_i), \quad \left.\frac{\partial f}{\partial z}\right|_i = \sum_{j \in C_i} c_{ij}(f_{ij} - f_i). \tag{3.3}$$

Note that if the function values at the mid-points are simply taken to be the average values of its two end points $f_{ij} = (f_i + f_j)/2$, then the standard least-squares approximation of the derivatives is recovered

$$\left.\frac{\partial f}{\partial x}\right|_i = \frac{1}{2}\sum_{j\in C_i} a_{ij}(f_j - f_i), \quad \left.\frac{\partial f}{\partial y}\right|_i = \frac{1}{2}\sum_{j\in C_i} b_{ij}(f_j - f_i), \quad \left.\frac{\partial f}{\partial z}\right|_i = \frac{1}{2}\sum_{j\in C_i} c_{ij}(f_j - f_i). \tag{3.4}$$

Applying the dual least-squares approximations given by Eq. (3.3) to each component of flux functions in Eq. (2.1), one obtains a semi-discrete form of the Euler equations at point i

$$\frac{\partial \mathbf{U}_i}{\partial t} + \sum_{j\in C_i}(a_{ij}(\mathbf{F}_{ij} - \mathbf{F}_i) + b_{ij}(\mathbf{G}_{ij} - \mathbf{G}_i) + c_{ij}(\mathbf{H}_{ij} - \mathbf{H}_i)) = 0. \tag{3.5}$$

In this dual least-squares formulation, fluxes are computed at the mid-points, which are not known a priori and have to be reconstructed in some way. It is by defining these numerical fluxes in a consistent and upwind manner that upwinding can be naturally introduced into the gridless methods and specified accuracy can be obtained. If the numerical fluxes at the mid-points are evaluated using the simple arithmetic averages of conservative variables at the two end points, the resulting numerical scheme is equivalent to the central differencing method. It is well known that such discretizations lead to unstable schemes, and must be augmented by stabilizing terms. This can be achieved either by adding directly second-, forth- or higher-order damping, or by using upwinding schemes. Over the last two decades characteristic-based upwind methods have established themselves as the methods of choice for prescribing the numerical fluxes for compressible Euler equations. The upwinding schemes seem especially attractive in the present context, as they do not require any intrinsic measure of length. In the present work, the numerical fluxes are approximated using the HLLC approximate Riemann solver [19], which has been successfully used to compute compressible viscous and turbulent flows on both structured grids [2] and unstructured grids [6]. HLLC scheme is known to be very robust in terms of positivity, entropy consistency, and convergence history. If the Riemann fluxes are evaluated using the flow variables $\mathbf{U}_i$ and $\mathbf{U}_j$, this scheme is equivalent to the first order upwind scheme. Many different ways exist to achieve higher order accuracy. In the present study, a scheme of higher order accuracy is obtained by using upwind-biased interpolations of the solution $\mathbf{U}$, via MUSCL approach [20]. The upwind-biased interpolations for $\mathbf{U}_i^+$ and $\mathbf{U}_j^-$ are defined by:

$$\mathbf{U}_i^+ = \mathbf{U}_i + \frac{1}{4}[(1-k)\Delta_i^- + (1+k)(\mathbf{U}_j - \mathbf{U}_i)], \tag{3.6}$$

$$\mathbf{U}_j^- = \mathbf{U}_j - \frac{1}{4}[(1-k)\Delta_j^+ + (1+k)(\mathbf{U}_j - \mathbf{U}_i)], \tag{3.7}$$

where the forward and backward difference operators are given by:

$$\Delta_i^- = \mathbf{U}_i - \mathbf{U}_{i-1} = 2(\nabla\mathbf{U})_i \cdot \mathbf{l}_{ij} - (\mathbf{U}_j - \mathbf{U}_i), \tag{3.8}$$

$$\Delta_j^+ = \mathbf{U}_{j+1} - \mathbf{U}_j = 2(\nabla\mathbf{U})_j \cdot \mathbf{l}_{ij} - (\mathbf{U}_j - \mathbf{U}_i), \tag{3.9}$$

where $\mathbf{l}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ is the length vector between the points i and j. The gradients $(\nabla\mathbf{U})_i$ and $(\nabla\mathbf{U})_j$ are computed using the standard least-squares approximation (3.4).

The parameter k can be chosen to control a family of difference schemes in the interpolation. On structured meshes it is easy to show that $k = -1$ yields a fully upwind scheme, $k = 0$ yields semi-upwind approximation (Fromm's scheme), and $k = 1$ yields central differencing. The value $k = 1/3$ leads to a third-order-accurate upwind-biased scheme, although third-order-accuracy is strictly correct only for one-dimensional calculations. Nevertheless, $k = 1/3$ was used in the calculations presented herein. With higher order spatial accuracy, spurious oscillations in the vicinity of shock waves are expected to occur. Some form of limiting is usually required to eliminate these numerical oscillations of the solution and to provide some kind of monotonicity property. The slope limiter modifies the upwind-biased interpolation $\mathbf{U}_i$ and $\mathbf{U}_j$, replacing them by:

$$\mathbf{U}_i^+ = \mathbf{U}_i + \frac{s_i}{4}[(1-ks_i)\Delta_i^- + (1+ks_i)(\mathbf{U}_j - \mathbf{U}_i)], \tag{3.10}$$

$$\mathbf{U}_j^- = \mathbf{U}_j - \frac{s_j}{4}[(1-ks_j)\Delta_j^+ + (1+ks_j)(\mathbf{U}_j - \mathbf{U}_i)], \tag{3.11}$$

where s is the limiter. van Albada limiter is employed in this study. It acts in a continuously differentiable manner and is defined by:

$$s_i = \max\left\{0, \frac{2\Delta_i^-(\mathbf{U}_j - \mathbf{U}_i) + \epsilon}{(\Delta_i^-)^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \epsilon}\right\}, \tag{3.12}$$

$$s_j = \max\left\{0, \frac{2\Delta_j^+(\mathbf{U}_j - \mathbf{U}_i) + \epsilon}{(\Delta_j^+)^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \epsilon}\right\}, \tag{3.13}$$

where ϵ is a very small number to prevent division by zero in smooth regions of the flow. Three options exist concerning the choice of interpolation variables: conservative variables, primitive variables, and characteristic variables. Using limiters on characteristic variables seems to give the best results. However, the primitive variables are used in this study for the sake of computational efficiency.

3.4. Cartesian grid method

For all the Cartesian cells, the governing equation (2.1) is discretized using a standard cell-centered finite volume formulation, where the control volume is taken as the cell of the Cartesian grid itself, and the cell-averaged variables are stored at the center of the cell. The finite volume approximation of the governing equations, applied to the cell (i,j,k) becomes

$$V\frac{\mathrm{d}\mathbf{U}_{i,j,k}}{\mathrm{d}t} + \int_{\Gamma_{i,j,k}} (\mathbf{F}n_x + \mathbf{G}n_y + \mathbf{H}n_z)\,\mathrm{d}\Gamma = 0, \tag{3.14}$$

where V is the cell volume, $\Gamma_{i,j,k}$ is the boundary of the cell (i,j,k), and $\mathbf{n} = (n_x, n_y, n_z)$ denotes the unit outward normal vector to the boundary of the cell. The flux integral in Eq. (3.14) is evaluated by summing all the contributions over the cell interfaces between the cell (i,j,k) and its neighboring cells. Eq. (3.14) can then be rewritten in a compact form as

$$\frac{\mathrm{d}\mathbf{U}_{i,j,k}}{\mathrm{d}t} = \mathbf{R}_{i,j,k}, \tag{3.15}$$

where $\mathbf{R}_{i,j,k}$ is the right hand side residual,

$$\mathbf{R}_{i,j,k} = -\left(\frac{\mathbf{F}_{i+\frac{1}{2},j,k} - \mathbf{F}_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{\mathbf{G}_{i,j+\frac{1}{2},k} - \mathbf{G}_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{\mathbf{H}_{i,j,k+\frac{1}{2}} - \mathbf{H}_{i,j,k-\frac{1}{2}}}{\Delta z}\right), \tag{3.16}$$

where Δx, Δy, and Δz are the cell size in the x-, y-, z-directions, respectively.

As for the gridless points, the numerical fluxes at the interface in Eq. (3.16) are computed using the HLLC approximate Riemann solver. A second order accuracy is achieved using the reconstruction scheme. van Albada limiter is used to suppress oscillations in the vicinity of discontinuities.

3.5. Temporal discretization

Eqs. (3.5) and (3.15) represent the time evolution of the unknown vector, which can be written as the following semi-discrete form for both Cartesian cells and gridless points:

$$\frac{\mathrm{d}\mathbf{U}_i}{\mathrm{d}t} = \mathbf{R}_i, \tag{3.17}$$

where $\mathbf{R}_i$ is the residual vector. Assuming that the unknown vector $\mathbf{U}_i^n$ is known at time t_n, the solution is advanced over a time step Δt, to time t_{n+1} by an explicit multi-stage Runge–Kutta time-stepping scheme given by:

$$\begin{aligned}
&\mathbf{U}_i^{(0)} = \mathbf{U}_i^n,\\
&\vdots\\
&\mathbf{U}_i^{(p)} = \mathbf{U}_i^{(0)} - \alpha_p \Delta t \mathbf{R}_i(\mathbf{U}_i^{(p-1)}), \quad p = 1, 2, \ldots, m,\\
&\vdots\\
&\mathbf{U}_i^{n+1} = \mathbf{U}_i^{(m)}
\end{aligned} \tag{3.18}$$

with the parameters α_p assigned appropriate values. For steady-state computations, where time accuracy is not required, local time-stepping is used to accelerate convergence to steady-state. The local time-stepping uses separately a maximum allowable step size for each node according to the local stability analysis.

4. Computational results

Computational results for a number of test cases are presented in this section. In order to demonstrate the numerical accuracy and computational efficiency of the present hybrid Cartesian grid and gridless method, these results are compared with those obtained using a cell vertex finite volume code [5,6] and with analytical solutions wherever possible. The first four test cases are chosen to validate and verify the baseline Cartesian grid method, which is the foundation of the proposed hybrid method. Test cases 5–9 are selected to validate the hybrid Cartesian and gridless method and to test the ability of this hybrid method for a variety of flow problems. Finally, the efficiency of the hybrid method in comparison with an unstructured grid method is shown for a typical three-dimensional flow problem in test 10. Note that one of advantages of the present hybrid method is its ability to compute 1D, 2D, and 3D problems using the very same code, which greatly alleviates the need and pain for code maintenance and upgrade. Results for one-dimensional flows can be readily obtained by setting the number of cells in both y- and z-directions to be 1. For two-dimensional problems, the number of cells in the z-direction is simply set to be 1. All computations use an explicit four-stage Runge–Kutta time-stepping scheme to advance the solution in time. A local time-stepping is used to accelerate convergence to steady-state computations.

Test case 1: Sod shock tube problem. The classic Sod shock tube problem is computed in this test case using 100 cells. The initial conditions in the present computation are the following:

$$\begin{aligned}&\rho = 1.000, \quad u = 0, \quad p = 1.0, \quad 0 \leqslant x < 0.5,\\ &\rho = 0.125, \quad u = 0, \quad p = 0.1, \quad 0.5 \leqslant x \leqslant 1.\end{aligned}$$

Fig. 2 compares the computed density, velocity, and pressure profile with the exact solution at $t = 0.2$.

Test case 2: Lax–Harden shock tube problem. This is another well-known test case for shock tube problem. The initial conditions in the present computation are the following:

$$\begin{aligned}&\rho = 0.445, \quad u = 0.698876404, \quad p = 3.52773, \quad 0 \leqslant x < 0.5,\\ &\rho = 0.50, \quad u = 0, \quad p = 0.571, \quad 0.5 \leqslant x \leqslant 1.\end{aligned}$$

The computation was performed using 100 cells. Fig. 3 compares the computed density, velocity, and pressure profile with the exact solution at $t = 0.15$.

Test case 3: Woodward–Colella blast wave problem. In this example, two interacting blast waves given by Woodward and Colella [3] are computed. The initial conditions for the Woodward–Colella blast wave are the following:

$$\begin{aligned}&\rho = 1.000, \quad u = 0.0, \quad p = 1000.0, \quad 0 \leqslant x < 0.1,\\ &\rho = 1.00, \quad u = 0, \quad p = 0.01, \quad 0.1 \leqslant x \leqslant 0.9,\\ &\rho = 1.00, \quad u = 0, \quad p = 100, \quad 0.9 < x \leqslant 1.0,\end{aligned}$$

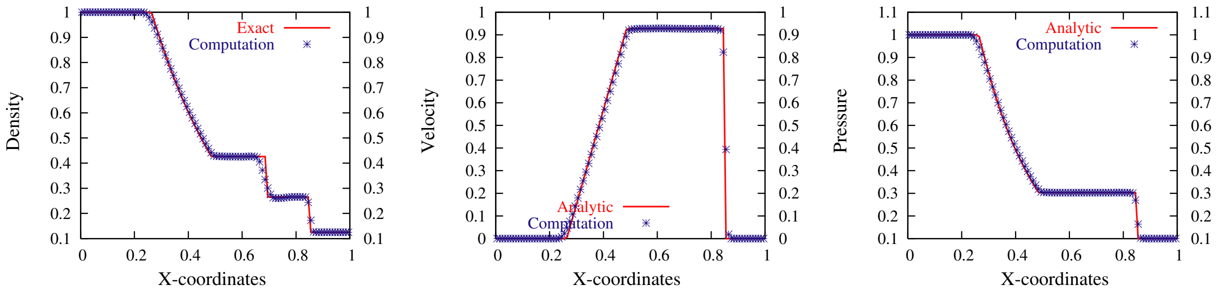


Fig. 2. Comparison of the computed solution with the analytic solution for Sod shock tube problem.

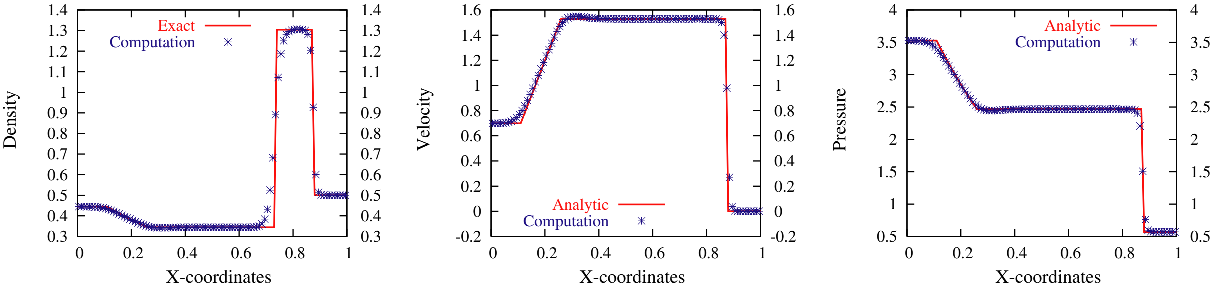


Fig. 3. Comparison of the computed solution with the analytic solution for Lax–Harden shock tube problem.

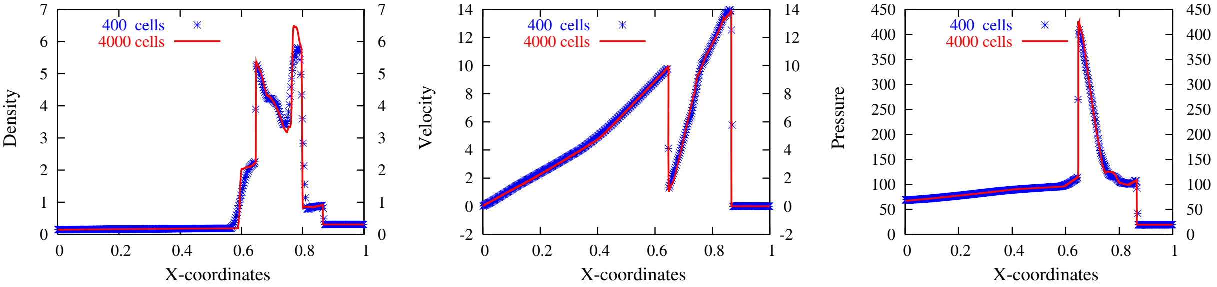


Fig. 4. Comparison of the computed solutions between coarse mesh and fine mesh for Woodward–Colella blast wave problem.

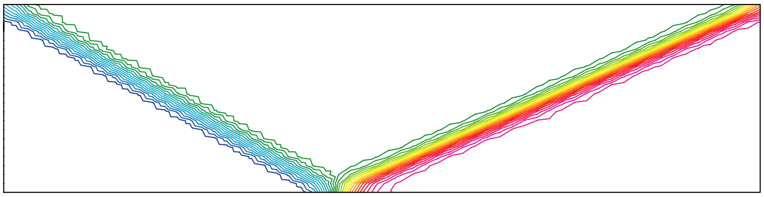


Fig. 5. Computed density contours in the flow field for stationary shock reflection problem.

400 cells are used in this case, and the computed results are shown in Fig. 4 for the density, velocity, and pressure distributions at $t = 0.038$, where the solid lines are obtained from the same calculation with 4000 cells.

Test case 4: Steady two-dimensional oblique shock wave. A two-dimensional shock wave reflecting from a rigid surface is considered in this test case. The computational domain is a rectangle of length 4.1 and height

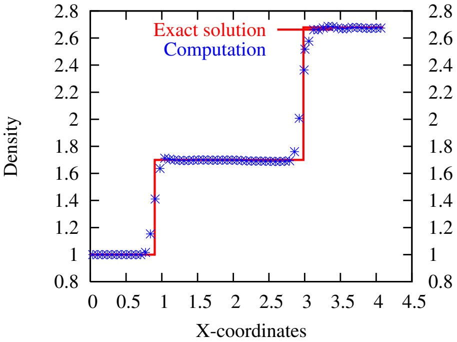


Fig. 6. Comparison of the computed density profile with the exact solution at $y = 0.5$ for stationary shock reflection problem.

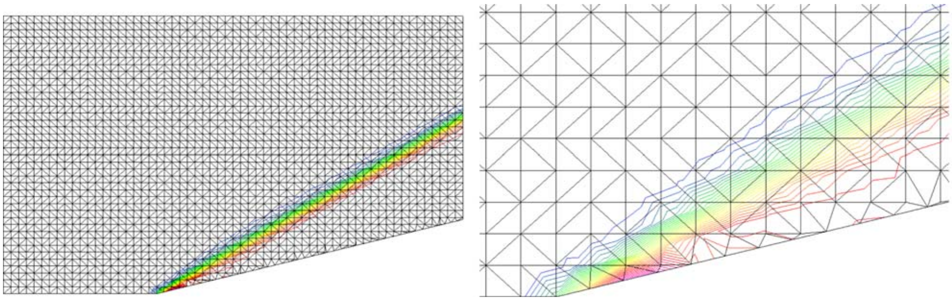


Fig. 7. Computed pressure contours in the flow field for supersonic flow past a 15° wedge at a Mach number of 3.

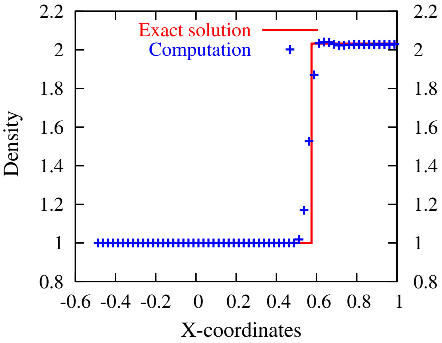


Fig. 8. Comparison of the computed density profile with the exact solution at $y = 0.3625$ for supersonic flow past a 15° wedge.

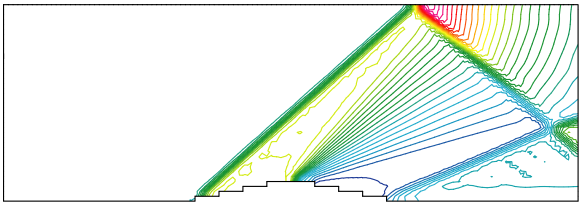


Fig. 9. Computed density contours for supersonic flow past a wedge at a Mach number of 2.

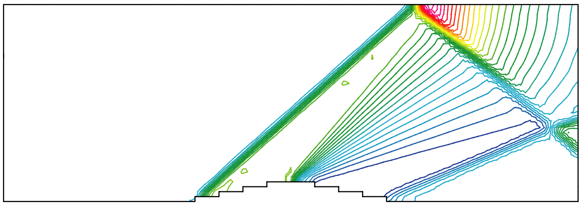


Fig. 10. Computed pressure contours for supersonic flow past a wedge at a Mach number of 2.

1, with a uniform grid size of 61 × 21 × 1. The boundary conditions are that of a reflecting surface along the bottom surface, supersonic outflow along the right surface, and prescribed fixed values on the other two sides, which are:

$$(\rho, u, v, p)_{(0,y,t)} = (1, 2.9, 0, 1/1.4),$$
$$(\rho, u, v, p)_{(x,1,t)} = (1.69997, 2.61934, -0.50632, 1.52819).$$

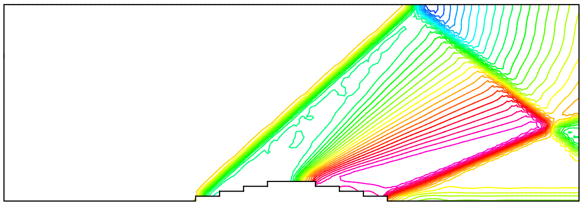


Fig. 11. Computed Mach number contours for supersonic flow past a wedge at a Mach number of 2.

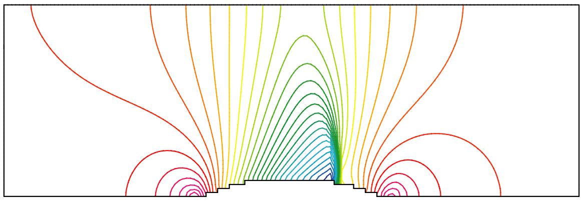


Fig. 12. Computed pressure contours for transonic flow past a bump at a Mach number of 0.675.

The boundary conditions produce an incident shock angle of 29° and the free stream Mach number M_∞ is 2.9. Fig. 5 shows the computed density contours in the flow field for the shock reflection problem. The computed density distribution is compared with the exact solution at $y = 0.5$ in Fig. 6.

Test case 5: Supersonic flow past a wedge. A recurring question often asked about gridless methods is whether they are able to maintain conservation at the discrete level, a property considered vital for proper shock capturing. For this reason, supersonic flow at $M_\infty = 3$ past a 15° wedge is considered in this example.

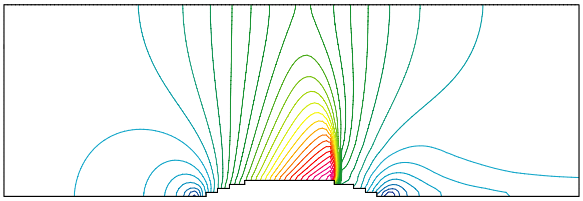


Fig. 13. Computed Mach number contours for transonic flow past a bump at a Mach number of 0.675.

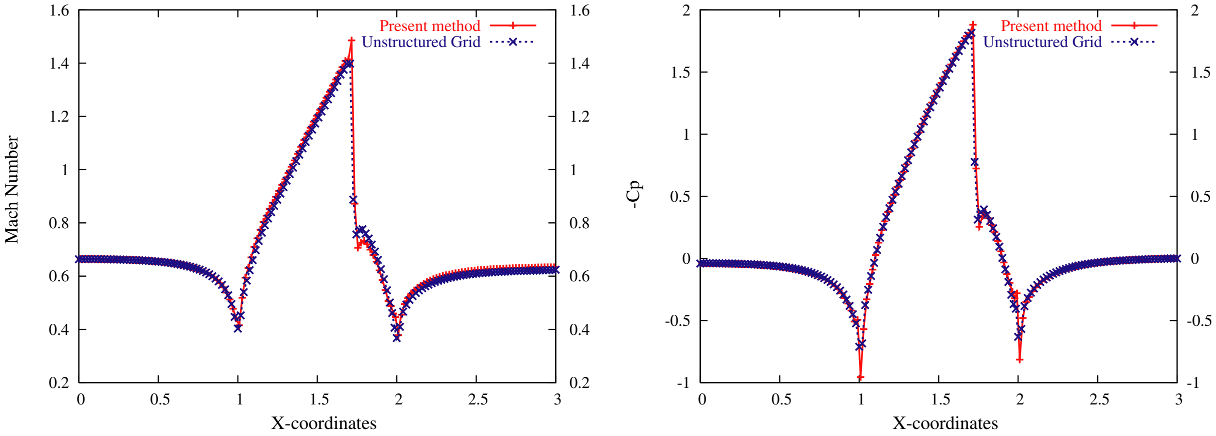


Fig. 14. Computed Mach number and pressure coefficient profile on the lower wall of the channel.

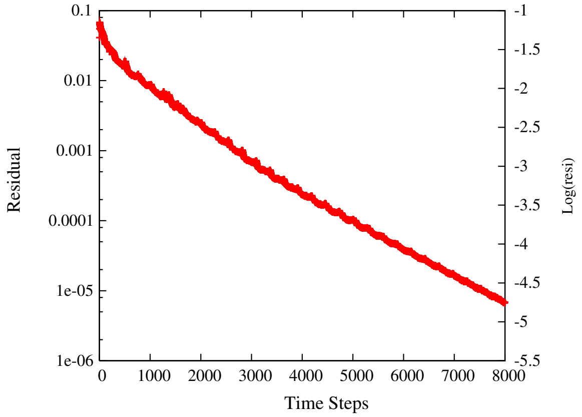


Fig. 15. Residual convergence history for transonic flow in a channel with a circular bump on the lower wall.

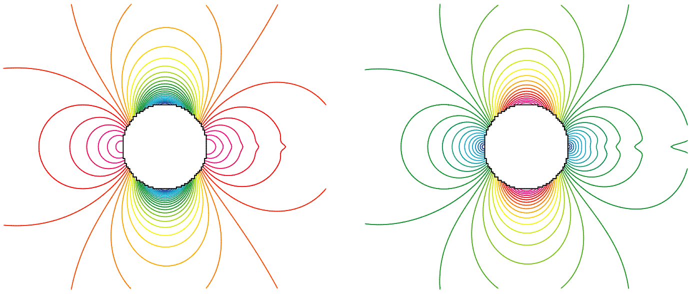


Fig. 16. Computed density (left) and Mach number (right) contours for subsonic flow past a sphere at a Mach number of 0.3, and an angle of attack of 0.

The Cartesian grid domain is a rectangle of length 1.5 and height 1, with a uniform grid size of 60 × 40 × 1. The computed pressure contours in the flow field are shown in Fig. 7, where one can observe a smooth transition in the region joining the gridless points and Cartesian cells. Note that a triangulation has been performed for both gridless points and Cartesian cells in order to plot the solution in the flow field. One can clearly see the shock, which is obtained at an angle of $\alpha_s = 32.33°$, in excellent correlation with analytical solution

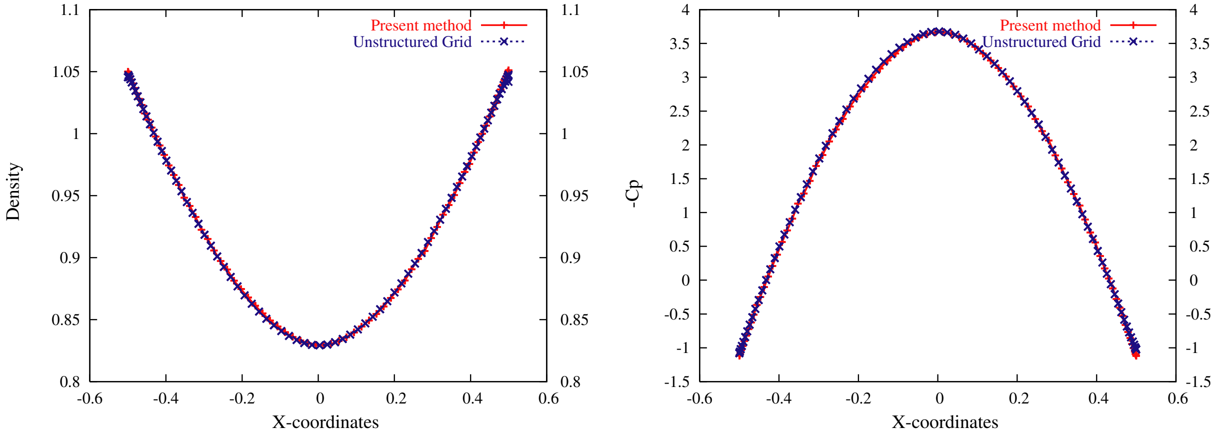


Fig. 17. Comparison of density profile (left) and pressure profile (right) on the surface of the sphere between the hybrid Cartesian grid and gridless method and the unstructured grid method.

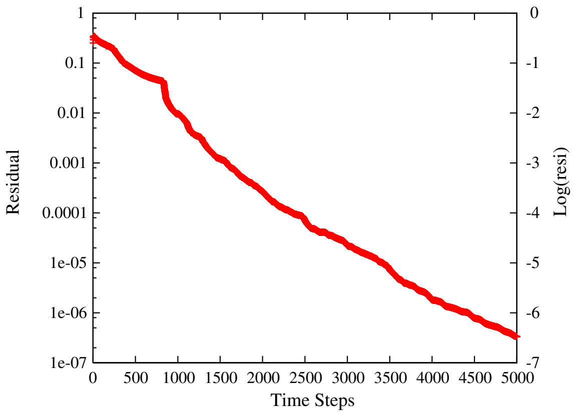
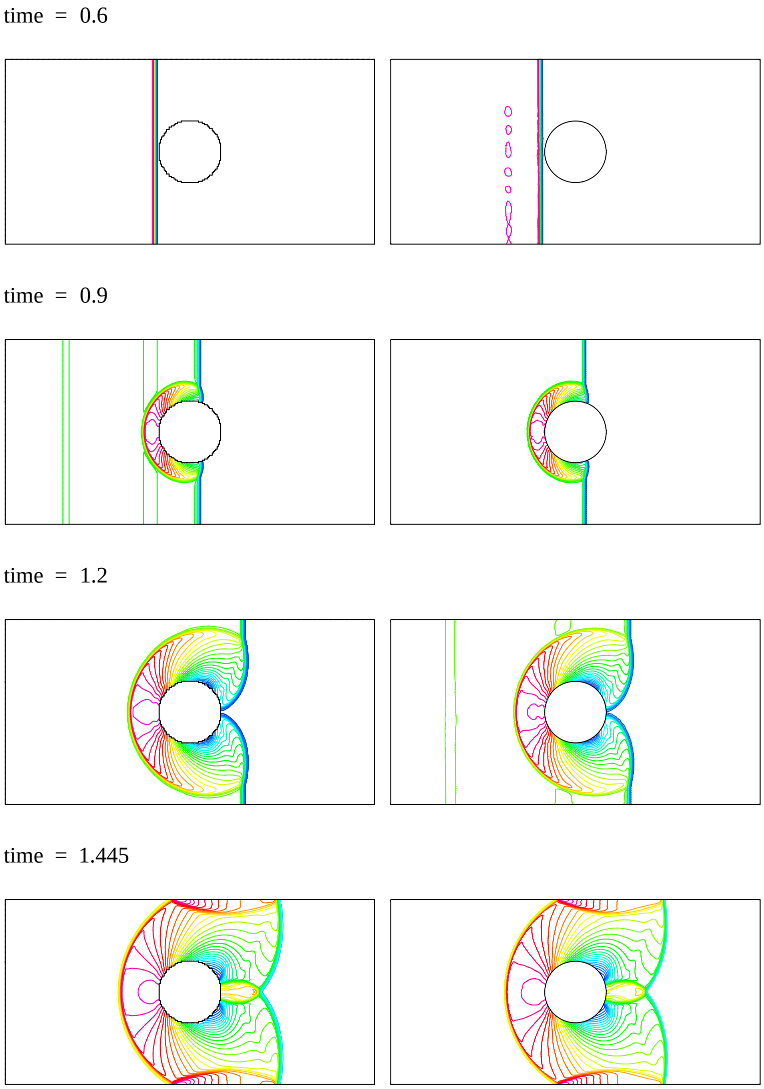


Fig. 18. Residual convergence history for subsonic flow past a sphere at a Mach number of 0.3, and an angle of attack of 0.



time = 0.6

time = 0.9

time = 1.2

time = 1.445

Fig. 19. Comparison of computed density contours for a incident shock $M_s = 2$ past a cylinder at different times obtained using the present hybrid method and the unstructured grid method.

$\alpha_s = 32.24°$. To see the resolution of the oblique shock, density distribution along a horizontal line at a distance of 0.3625 from the bottom is compared with the analytical solution in Fig. 8, where the oblique shock is captured within four or five cells: a typical resolution obtained using a second order scheme.

Test case 6: Supersonic flow past a diamond wedge. The case considered here is the supersonic flow past a wedge in a channel at $M_\infty = 3$. The height of the wedge is 0.1. The Cartesian grid domain is a rectangle of length 3.0 and height 1, with a uniform grid size of $120 \times 40 \times 1$. The height of the wedge is 0.1 and the length is 1.0. The computed density, pressure, and Mach number contours in the flow field are shown in Figs. 9–11, respectively.

Test case 7: Transonic flow in a channel with a circular bump on the lower wall. The example is the well-known Ni's test case: a transonic flow in a channel with a 10% thick circular bump on the bottom. Inlet Mach number is 0.675. The Cartesian grid domain is a rectangle of length 3.0 and height 1, with a uniform grid size of $150 \times 50 \times 1$. The computed pressure and Mach number contours in the flow field are presented in Figs. 12 and 13, respectively. The Mach number and pressure coefficient distributions on the lower wall obtained by the present hybrid method are compared with those obtained using a cell-vertex finite volume scheme in Fig. 14. Fig. 15 shows the convergence history versus time steps for this computation. The computed lift coefficient of −2.492 and drag coefficient of 0.01534 for the bump obtained by the hybrid method are compared closely with the values of −2.484 and 0.01528 by the unstructured grid method, respectively.

Test case 8: Subsonic flow in a channel over a cylinder. Flow past a cylinder for a free-stream Mach number of 0.3 is simulated in this case. This is a potential flow case for which an exact solution is available. The potential solution is completely symmetric, and any spurious entropy creation will be reflected in the numerical solution. The Cartesian grid domain is a square of length 6 and height 6, with a uniform grid size of $300 \times 300 \times 1$. The diameter of the cylinder is 1. The computed pressure and Mach number contours in the flow field are presented in Fig. 16. The Mach number and pressure coefficient distributions on the lower wall obtained by the present hybrid method are compared with those obtained using a cell-vertex finite volume scheme in Fig. 17(d). Fig. 18 shows the convergence history versus time steps for this computation.

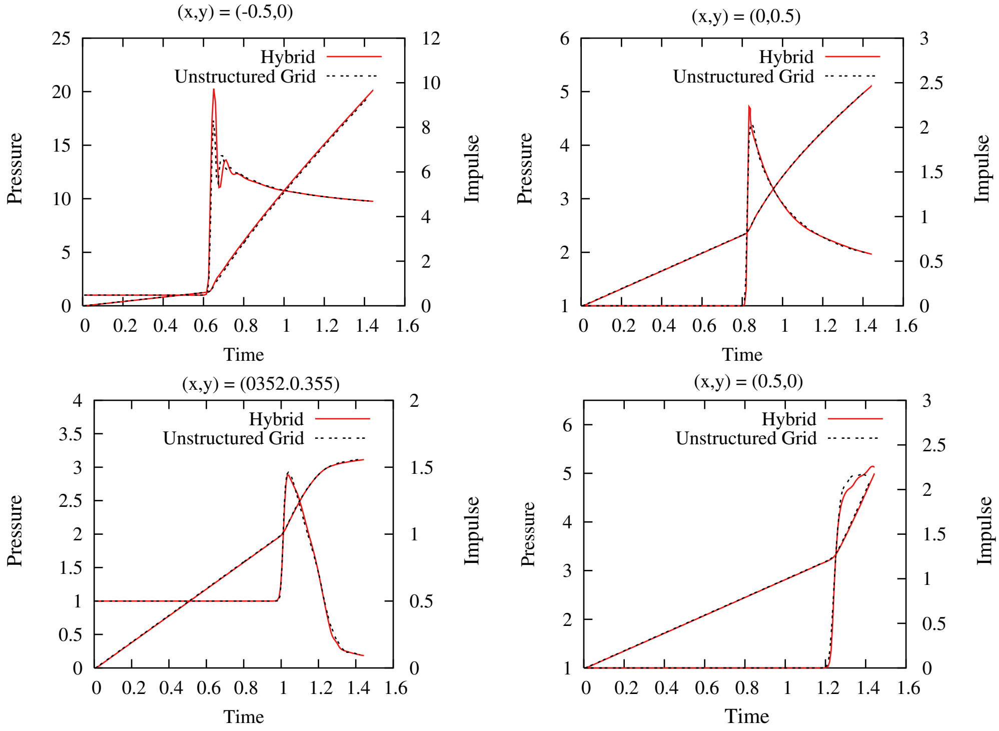


Fig. 20. Comparison of pressure and impulse time history at different locations obtained using the present hybrid method and the unstructured grid method.

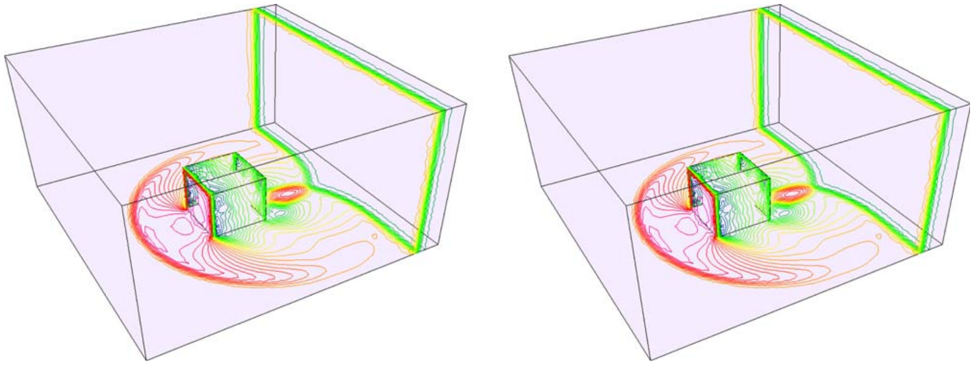


Fig. 21. Comparison of computed density contours for a incident shock $M_s = 2$ past a cube obtained using the present hybrid method (left) and the unstructured grid method (right).

Test case 9: Incident shock past a cylinder. In this case, an incident shock $M_s = 2$ past a cylinder in a channel is computed. The Cartesian grid domain is a rectangle of length 6 and height 3, with a uniform grid size of $300 \times 150 \times 1$. The diameter of the cylinder is 1 and its center is located at $(0,0)$. The computed density contours in the flow field at different times obtained using the present hybrid method are compared with those obtained using the unstructured grid method in Fig. 19. Fig. 20 shows a comparison of pressure and impulse time histories at the different locations between the present hybrid method and the unstructured grid method.

Test case 10: Incident shock past a cube. In this case, an incident shock $M_s = 2$ past a cube in a channel is computed. The computed density contours in the flow field obtained using the present hybrid method are compared with those obtained using the unstructured grid method in Fig. 21. One can see that both methods produce virtually identical results. This example is chosen to demonstrate the superior performance of the present hybrid Cartesian-grid and gridless method over the unstructured grid method for a typical computation of shock wave propagation and diffraction problem. The Cartesian grid domain is a cube of $5 \times 3 \times 5$, with a uniform grid size of $63 \times 38 \times 63$. The boundary points, representing the cube, have 720 points. The unstructured tetrahedral grid with the same mesh size ($h = 0.08$), used for our unstructured grid method, consists of 1,268,020 elements, 224,833 points, and 20,219 boundary points. An overall speed-up factor about eight and a saving in storage requirements about one order of magnitude in comparison with the unstructured grid method are observed for this computation.

5. Conclusions

A hybrid Cartesian grid and gridless method has been developed for solving the unsteady compressible Euler equations. The developed method combines the efficiency of a Cartesian method and the flexibility of a gridless method for the complex geometries. Extensive numerical tests have been performed to demonstrate the accuracy, efficiency, robustness, and versatility of the proposed method. The numerical results obtained indicate that the use of this hybrid method leads to a significant improvement in performance over its unstructured grid counterpart without compromising solution accuracy, demonstrating the great potential and benefits of this hybrid method for the simulation of transient shock interaction problems. An overall speed-up factor from six to ten and a saving in storage requirements about one order of magnitude for a typical 3d simulation in comparison with the unstructured grid method are obtained. Ongoing work is to extend this hybrid method for non-uniform Cartesian grids using the building-cube method [21].

Acknowledgments

This research was partially sponsored by the Defense Threat Reduction Agency. Drs. Darren Rice and Michael E. Giltrud served as the technical program monitor.

References

[1] A. Jameson, S. Yoon, Lower–upper implicit schemes with multiple grids for the Euler equations, AIAA Journal 25 (1987) 929–935.

[2] P. Batten, M.A. Leschziner, U.C. Goldberg, Average-state Jacobians and implicit methods for compressible viscous and turbulent flows, Journal of Computational Physics 137 (1997) 38–78.

[3] P.R. Woodward, P. Colella, The numerical simulation of two-dimensional fluid flow with strong shocks, Journal of Computational Physics 54 (1984) 115–173.

[4] A. Jameson, D. Mavriplis, Finite volume solution of the two-dimensional Euler equations on a regular triangular mesh, AIAA Journal 24 (1986) 611–618.

[5] H. Luo, J.D. Baum, R. Löhner, A fast, matrix-free implicit method for compressible flows on unstructured grids, Journal of Computational Physics 146 (2) (1998) 664–690.

[6] H. Luo, J.D. Baum, R. Löhner, High-Reynolds Number Viscous Flow Computations Using an Unstructured-Grid Method, AIAA Paper, AIAA-2004-1103, 2004, Journal of Aircraft 42 (2) (2005) 483–492.

[7] W.J. Coirier, K.G. Powell, An accuracy assessment of Cartesian mesh approaches for the Euler equations, Journal of Computational Physics 117 (1995) 121–131.

[8] M.J. Berger, R.J. LeVeque, An Adaptive Cartesian Mesh Algorithm for the Euler Equations in Arbitrary Geometries, AIAA Paper 1989-1930, 1989.

[9] D.K. Clarke, M.D. Salas, H.A. Hassan, Euler calculations for multielement airfoils using Cartesian grids, AIAA Journal 24 (1986) 353–358.

[10] J.T. Batina, Gridless Euler/Navier–Stokes Solution Algorithm for Complex Aircraft Applications, AIAA Paper 1993-0333, 1993.

[11] K. Morinishi, An Implicit Gridless Type Solver for the Navier–Stokes Equations, in: International Symposium on CFD, Bremen, 1999.

[12] K. Morinishi, Effective Accuracy and Conservation Consistency of Gridless Type Solver, in: Proceedings of the First International Conference on Computational Fluid Dynamics, Kyoto, Japan, 10–14, July, 2000.

[13] P. Chandrashekar, S.M. Deshpande, A New Grid-Free Method for Conservation Laws, in: Proceedings of the Second International Conference on Computational Fluid Dynamics, Sydney, Australia, 15–19, July, 2002.

[14] R. Löhner, S. Sacco, E. Onate, S. Idelsohn, An finite point method for compressible flow, International Journal for Numerical Methods in Engineering 53 (2002) 1765–1779.

[15] D. Sridar, N. Balakrishnan, An upwind finite difference scheme for meshless solvers, Journal of Computational Physics 189 (2003) 1–29.

[16] E.P. Koh, H.M. Tsai, F. Liu, Euler solution using Cartesian grid with least squares technique, AIAA Journal 43 (2) (2005) 246–255.

[17] D.J. Kirshman, F. Liu, Gridless boundary condition treatment for a non-body-conforming mesh, Journal of Computational Physics 201 (1) (2004) 119–147.

[18] D.J. Kirshman, F. Liu, Cartesian Grid Solution of the Euler Equations Using A Gridless Boundary Condition Treatment, AIAA Paper 2003-3974, 2003.

[19] E.F. Toro, M. Spruce, W. Speares, Restoration of the contact surface in the HLL-Riemann solver, Shock Waves 4 (1994) 25–34.

[20] B. van Leer, Towards the ultimate conservative difference scheme, II. Monotonicity and conservation combined in a second order scheme, Journal of Computational Physics 14 (1974) 361–370.

[21] K. Nakahashi, L.S. Kim, Building-Cube Method for Large-Scale, High Resolution Flow Computations, AIAA Paper 2004-0423, 2004.