Wave propagation algorithms on quadrilateral and hexahedral meshes

by Donna Calhoun
Commissariat à l'Energie Atomique, DEN/DM2S/SFME
Saclay, France

Abstract

We propose a high resolution finite-volume scheme for the solution of hyperbolic problems on two- and three-dimensional mapped grids. The algorithms are based on the wave propagation algorithms available in CLAWPACK, a freely available software package developed by R. J. LeVeque. The method has also been extended to adaptively refined meshes. In this brief article, we present a novel grid mapping that can be used in place of the standard mapping based on cylindrical coordinates. We use this mapping to solve the Euler equations of gas dynamics in a closed circular cylinder.

1 Introduction

Many multi-dimensional problems of scientific and engineering interest are posed in complicated, non-rectangular domains. For example, computing the flow around an entire aircraft configuration, blood flow in arteries and flow over mountain topography requires solving partial differential equations in regions that to do not naturally align with Cartesian directions. For this reason, there is significant interest in developing numerical methods that can be used to solve equations in arbitrary geometry.

In fact, the success of the finite-element method is due in large part to its ability to easily handle arbitrarily shaped domains. However, for many modern three-dimensional problems, the unstructured mesh generation required for very complicated domains can require considerable computational effort. In particular, if the physical domain is changing in time, as it is in many biological problems, mesh generation becomes prohibitive. In addition, unstructured mesh data that is collocated physically may not be collocated in computer memory, thereby making it difficult to take advantage of memory caching.

For this reason, there has been renewed interest in developing algorithms which can handle realistic geometry, but do not require extensive mesh generation and which can also take advantage of a structured layout of the physical memory of the computer. One class of such algorithms is the class of Cartesian grid methods. These algorithms attempt to take maximum advantage of the structured layout of a Cartesian grid while also taking into account physical boundaries that do not naturally align with the uniform Cartesian mesh.

There are essentially two approaches to handling geometry on a Cartesian grid. In the first approach, one simply places the geometry on a uniform Cartesian grid and allows the geometry to cut the mesh cells in an arbitrary manner. Away from the boundary, one can apply the usual regular grid operator. Near the boundary, one applies special stencils that accurately account for the presence of the physical boundary or interface. The advantage of this approach is that one can handle essentially arbitrary geometry. These methods go by various names, including "embedded boundary methods", "immersed boundary methods", "cut-cell" methods, or simply "Cartesian grid methods" for irregular domains [3, 6, 7, 10, 12].

In the second approach, and the one that is the focus of the present research, one deforms the Cartesian mesh so that the grid aligns with the boundaries of the desired physical domain. In this approach, as in the previous approach, the computational grid remains logically Cartesian, but one is not restricted to a rectangular physical region. The classic example of a mapped logically Cartesian grid is the polar grid. Other examples include the grid mappings used to compute flow past airfoils and over topography.

The use of these deformed grids, better known as curvilinear or mapped grids, to solve problems on non-rectangular geometry in fact precedes the use of embedded boundary techniques. Nonetheless, curvilinear grids have gained renewed interest for several reasons. For one, the difficulty in implementing

a curvilinear grid algorithm roughly scales with the complexity of the physical geometry. Many physical domains of practical interest are easily described by a curvilinear grid, and implementing a mapped grid algorithm in these cases can be significantly easier than using an embedded boundary technique. Second, curvilinear grids may do a better job of resolving the solution near physical boundaries than embedded boundary techniques, which rely on complicated stencils near the boundary for accuracy. Finally, mapped grid algorithms can be easily adapted for use with adaptive mesh refinement (AMR) techniques [2, 11].

For the above reasons, we have focused our attention on developing mapped grid algorithms for hyperbolic problems. In what follows, we briefly describe a fairly general approach to solving hyperbolic problems using finite-volume methods on logically Cartesian, mapped grids in two and three dimensions. As an example, we solve the Euler equations of gas dynamics in a cylinder. For the computational mesh, we present a novel alternative to the standard mapping based on cylindrical coordinates.

2 Computational framework

We focus our attention on the class of hyperbolic problems in either conservative or non-conservative form given here in one dimension by

$$q_t + f(q)_x = \psi(q, x, t) \tag{1}$$

or in quasi-linear form, given by

$$q_t + A(q, x, t)q_x = \psi(q, x, t). \tag{2}$$

We assume that q is a state vector of conserved quantities in $R^{m\times 1}$ and that the flux function $f(q)$, also in $R^{m\times 1}$ is nonlinear. The flux Jacobian $f'(q)$ and the matrix $A(q, x, t)$, both in $R^{m\times m}$, are assumed to have full sets of eigenvectors with associated real eigenvalues. For the present work, we assume that the source term ψ depends only on q and not on spatial derivatives of q.

To solve the above equations numerically, we use the high-resolution wave propagation algorithms developed by Randall LeVeque and available in the software package CLAWPACK [8]. The algorithm is explicit and proceeds by "sweeping" over rows or columns of cells in each Cartesian direction on the computational mesh and solving one-dimensional Riemann problems at each cell interface. At each interface, waves and associated speeds are obtained and then used to update cell averages. These methods are formally second-order accurate on smooth grids. Wave limiters are used to damp spurious oscillations that can occur near discontinuities. The complete details of this particular approach are described in [9].

3 Wave propagation algorithms on mapped grids

The wave propagation algorithm on a uniform Cartesian mesh assumes that the cell interfaces across which Riemann problems are solved align with the Cartesian directions. On a mapped grid, this, in general, will not be true. When solving on mapped grids, one must take into account the particular orientation of each cell interface. Furthermore cell average updates must take into account the varying cell sizes of mapped computational cells.

Typically, one includes these necessary metric terms by transforming the given partial differential equation (PDE), usually specified in Cartesian components, into components in the desired computational coordinate system. The resulting PDE is then solved without further reference to the underlying coordinate system and the components of any tensor quantities in the resulting solution are interpreted as coefficients of the underlying coordinate vectors. However, this process of transforming the PDE using analytically derived metric terms can be error prone, especially for complicated grid mappings, and does not lend itself well to the development of general algorithms. For numerical methods based on solving Riemann problems, one is also likely to need a different Riemann solver for each type of grid transformation.

For this reason, we have taken an approach which relies on approximating the mapped grid cells by quadrilateral or hexahedral cells. In our approximation, edges of the two-dimensional mapped grid cell are assumed to be straight lines. Faces of the three-dimensional mapped cell are assumed to be ruled

surfaces. With these basic approximations, we can then compute approximations to all necessary metric quantities using only the physical locations of the four corners of the quadrilateral or the eight corners of the hexahedral grid cell.

To apply the wave propagation algorithm on a mapped quadrilateral or hexahedral grid, we can still sweep in the directions corresponding to those of the logically Cartesian grid and solve Riemann problems at cell edges or faces. We must however take into account the orientation of these interfaces relative to the Cartesian directions. To do so, we project any tensorial data (momentum, for example) into a local coordinate system aligned with the cell edges of the mapped grid. This rotated data is then passed to the one-dimensional Riemann solver to obtain waves and speeds. The resulting waves are then rotated back into the Cartesian space and the speeds are scaled by the length or area of the particular grid cell interface. The update of the cell average based on these waves and speeds then involves the physical area (or volume) of the mapped grid cell.

One advantage of our approach is that we can use the same Riemann solver for both mapped and non-mapped problems. Furthermore, we have written several routines which automate the process of computing and accessing the metric terms needed to implement our algorithms in two and three dimensions. We have performed several numerical tests of this approach and have found that the numerical accuracy of the scheme is essentially the same as that on a uniform Cartesian grid, at least for reasonably smooth mappings.

4 Solving the Euler equations on a circular cylinder

To demonstrate our mapped grid approach, we solve the three-dimensional compressible Euler equations in a closed cylinder with a circular cross-section of varying radius. The axis of the cylinder is aligned with the z-axis and the cross-section is parallel to the x-y-plane.

The obvious choice of mesh for the circular cross-section is the polar grid. This grid has the chief advantage that it is easy to describe, and therefore one can easily transform the PDE into polar coordinates. Figure 1 shows a typical polar grid. However, for explicit finite-volume schemes, the polar grid has at least one major drawback. The ratio of the size of the largest to smallest cell grows like $O(1/\Delta r)$. This places severe restrictions on the time step size needed for stability without significantly improving the global resolution of the solution. Furthermore, a general Cartesian grid method typically assumes that one can define values in a border layer of ghost cells surrounding the physical domain. The values one puts in these ghost cells depends on the desired physical boundary conditions at that boundary. For the polar grid, these ghost cells are not formally defined for $r < 0$, and so one must treat them in a special manner.

To overcome these disadvantages, we have developed alternatives to the standard polar grid [5]. One such mapping is shown in Figure 1. Our two-dimensional mapping transforms computational coordinates $(\xi, \eta) \in [-1, 1] \times [-1, 1]$ into a circle of radius $\sqrt{2}$. For $|\xi| \leq \eta$, this mapping is given by

$$\begin{aligned} X_2(\xi, \eta) &= \xi \\ Y_2(\xi, \eta) &= \sqrt{R(\eta)^2 - \xi^2} - \sqrt{R(\eta)^2 - \eta^2} + \eta, \qquad R(\eta) \quad = \sqrt{2}\left(\alpha\eta + \frac{1-\alpha}{\eta}\right) \end{aligned} \tag{3}$$

where α is a parameter in $[0, 1]$. The three other triangular regions of the computational grid are mapped in an analogous manner. We can easily extend this mapping for $|\xi| \leq \eta$ to a three-dimensional cylinder as

$$\begin{aligned} X(\xi, \eta, \zeta) &= \rho(\zeta)\xi/\sqrt{2} \\ Y(\xi, \eta, \zeta) &= \rho(\zeta)Y_2(\xi, \eta)/\sqrt{2} \\ Z(\xi, \eta, \zeta) &= \zeta. \end{aligned} \tag{4}$$

For our present example, we have used this cylindrical mapping, with the radius $\rho(\zeta)$ of the cross-section defined as $\rho(\zeta) = 0.4\zeta^2 + 0.6\zeta$. The resulting cylinder is shown in Figure 2.

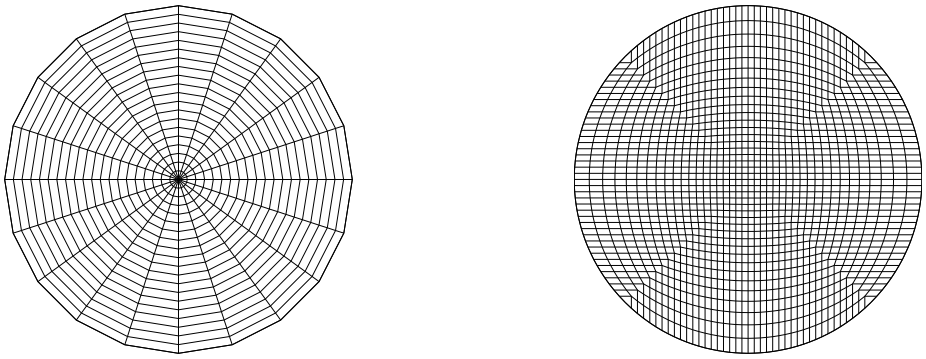

Figure 1: The left plot shows a 20×20 polar grid. The ratio of the areas of the largest to smallest grid cells for this grid is 43. The right plot shows a 40×40 grid of the new circular mesh described in (3) with $\alpha = 0$. The area ratio of largest to smallest cells for this mesh is about 2.43.

The equations we solve inside this cylinder are the standard Euler equations of gas dynamics. See for example [9]. To initialize the problem, we define a spherical region centered at a point on the lower edge of the cylinder and set the pressure inside this region to $p_{in} = 3$ and outside to $p_{out} = 0.03$. The density is set to one everywhere inside the cylinder, and the velocity is set to zero everywhere. On all surfaces of the cylinder, we impose solid wall boundary conditions.

The initial high pressure region expands into the cylinder and the resulting pressure waves are reflected off the surfaces of the cylinder. Figure 2 shows contours of pressure at two different times. The contour lines are relatively smooth and show little evidence of grid effects. They show clearly how the advancing pressure wave is reflected off the sides of the cylinder and collides with itself on the surface opposite the initial spherical region of high pressure.

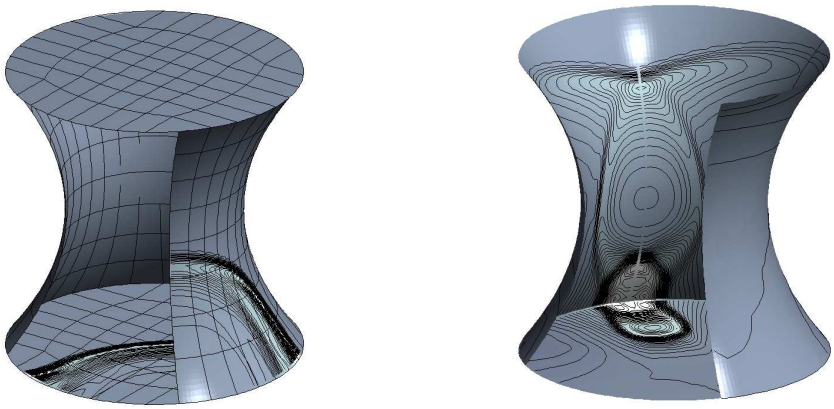

Figure 2: The evolution of a high pressure region in a closed cylinder at time $t = 1.5$ (left) and at time $t = 4.5$ (right). In both plots, we have cut away a portion of the cylinder to show the pressure contours on the interior surface. The contour levels shown are in the range $[0, 0.14]$ (left) and in $[0, 0.11]$ (right). The left plot shows the computational mesh, coarsened by a factor of 8.

The FORTRAN code and MATLAB visualization tools used to create this example are available at

5 Conclusions

We have briefly described methods for adapting the existing Cartesian grid algorithms available in CLAWPACK to mapped, logically Cartesian grids. One advantage of our approach is that one is not required to provide metric terms analytically. Quantities such as length or area of the grid cell faces and edges and the area of volume of the grid cells are all computed based on a quadrilateral or hexahedral approximation of the two- or three-dimensional mapped grid. Another advantage is that for many problems, one can use existing Riemann solvers written for Cartesian grids rather than writing a new one for each grid transformation.

These methods have all been extended to two- and three-dimensional adaptively refined grids (AMR), using the Berger-Oliger block-structured approach to refining grid patches [1, 2]. The AMR version of these algorithms has been implemented in the freely available CHOMBOCLAW software package [4]. CHOMBOCLAW runs on a variety of platforms, including parallel platforms supporting the Message Passing Interface (MPI). At the CEA, we plan to use CHOMBOCLAW to simulate peak over-pressures resulting from potential hydrogen explosions.

References

[1] M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

[2] M. J. Berger and R. J. LeVeque. Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems. *SIAM J. Num. Anal.*, 35(6):2298–2316, 1998.

[3] D. Calhoun. A Cartesian grid method for solving the two-dimensional streamfunction-vorticity equations in irregular regions. *J. Comput. Phys.*, 176:231–275, 2002.

[4] D. Calhoun. ChomboClaw User's Guide, Version 1.0. Technical report, Lawrence Berkeley Laboratory, 2005. In preparation.

[5] D. Calhoun, C. Helzel, and R. J. LeVeque. Wave Propagation on Logically rectangular Finite Volume Grids for "Circular" and "Spherical" Domains. *to appear*, 2006.

[6] D. Calhoun and R. J. LeVeque. A Cartesian grid finite-volume method for the advection-diffusion equation in irregular geometries. *J. Comput. Phys.*, 157:143–180, 1999.

[7] C. Helzel, M. Berger, and R. J. LeVeque. A high-resolution rotated grid method for conservation laws with embedded boundaries. *SIAM J. Num. Anal.*, 26(3):785–809, 2005.

[8] R. J. LeVeque. CLAWPACK software. Available on the Web at the URL `http://www.amath.washington.edu/~claw/`.

[9] R. J. LeVeque. *Finite volume methods for hyperbolic problems.* Cambridge University Press, 2002.

[10] X.D. Liu, R. P. Fedkiw, and M. Kang. A boundary condition capturing method for Poisson's equation on irregular domains. *J. Comput. Phys.*, 160(1):151–178, May 2000.

[11] E. Steinthorsson, D. Modiano, and P. Colella. Computations of unsteady viscous compressible flows using adaptive mesh refinement in curvilinear body-fitted grid systems. Technical Report NASA Technical Memorandum 106704, NASA, 1994.

[12] T. Ye, R. Mittal, H. S. Udaykumar, and W. Shyy. An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries. *J. Comput. Phys.*, 156:209–240, 1999.