A parameter-free moving mesh method applied to conservation laws

A. van Dam
Dept. of Mathematics, Utrecht University,
P.O.Box 80.010, 3508 TA Utrecht, The Netherlands
dam@math.uu.nl

Abstract

Adaptive techniques for solving partial differential equations (PDEs) have existed for decades, but are often finetuned for handling very specific problems. We have implemented a moving mesh finite volume solver that is suitable for nonlinear systems of hyperbolic PDEs in general. It employs a self-regulating monitor function to steer mesh movement, so that no parameter finetuning by hand is necessary. Application to problems from (magneto-)hydrodynamics show the robustness of this method.

Keywords: Moving mesh, monitor function, conservation laws

1. Introduction

In this work we use a moving mesh method to solve nonlinear systems of hyperbolic partial differential equations (PDEs). We show how an existing method can be made more powerful by combining it with a sophisticated monitor function. Our method is automatically fitted for any new input problem. Sometimes, new phenomena can then be discovered, as the experiments will show.

2. Systems of conservation laws

Non-viscous compressible fluid or gas flows are modeled by the Euler equations. Plasma flows obey the same laws, with additional terms that quantify Lorentz forces, and include an additional PDE describing magnetic induction. In general, we consider nonlinear systems of hyperbolic PDEs following from conservation laws:

$$\frac{\partial}{\partial t}q + \nabla f(q) = 0, \qquad (1)$$

where $q \in \mathbb{R}^m$ contains m model variables, and $f(q)$ is the flux function.

3. A self-regulating moving mesh solver

Our moving mesh solver combines mesh adaptation based on equidistribution and a high resolution finite volume solver as introduced by Tang et al. [2]. For increased performance, we use a more sophisticated monitor function. The method has two main advantages. Firstly, the mesh adaptation is always balanced for the current solution automatically, without manually set parameters. Secondly, the mesh movement is decoupled from solving the physical PDEs, which makes reuse with a different PDE solver easy.

The numerical algorithm is shown below. The symbol $Q_{j+1/2}$ represents the numerical solution for q averaged over a mesh cell, following from the finite volume approach in Section 3.2. Each time step consists of a mesh moving step and a physical PDE solving step. The following sections describe these separate steps. In a recent publication [1], we provide full details on the techniques used.

Algorithm 1 MMFVSOLVE – 1D moving mesh finite volume PDE solver.

```
Generate an initial uniform mesh: $x_j^0 = x_L + j \cdot \frac{x_R - x_L}{N}$, $j = 0, \ldots, N$.
Compute initial values $Q_{j+1/2}^0$ based on cell average of $q(x, 0)$.
**while** $t_n < T$ **do**
  **repeat**
    $\nu = 0$; $x_j^{[0]} = x_j^n$; $Q_{j+1/2}^{[0]} = Q_{j+1/2}^n$, $j = 0, \ldots, N$.
    Move grid $\left\{x_j^{[\nu]}\right\}$ to $\left\{x_j^{[\nu+1]}\right\}$, using a Gauss-Seidel step.
    Compute the solution $\left\{Q_{j+1/2}^{[\nu+1]}\right\}$ on the new mesh.
  **until** $\nu \geq \nu_{\max}$ or $\left\|x^{[\nu+1]} - x^{[\nu]}\right\| \leq \epsilon$
  Compute $Q^{n+1}$ using high resolution finite volumes.
**end while**
```

3.1. Mesh adaptation from a variational approach

The solution of the physical PDEs, denoted by $q \in \mathbb{R}^m$, is defined on the physical domain $\Omega_p \equiv [x_L, x_R] \subset \mathbb{R}$ with coordinate x. Introducing a fixed computational domain $\Omega_c \equiv [0, 1] \subset \mathbb{R}$, with coordinate ξ, a coordinate transformation, or one-to-one mesh map, is defined by $x = x(\xi), \quad \xi \in \Omega_c$.

In a variational approach, finding the most appropriate mesh map $x(\xi)$ for some solution profile is equivalent to finding a ξ that minimizes a mesh energy functional. Here, this leads to the often-used equidistribution relation:

$$(\omega x_\xi)_\xi = 0, \tag{2}$$

where $\omega(q)$ is monitor function that specifies the 'importance' of certain parts of the solution. This is further discussed in Section 3.3.

Moving the mesh The equidistribution relation (2) is discretized using central differences, and solved iteratively, e.g., by some Gauss-Seidel steps. More advanced iterative solvers exist, but the convergence should not be too fast. The large difference between old and new mesh points would then cause large errors in the solution interpolation step.

Solution interpolation Although interpolating the solution onto the new mesh can not be entirely free of errors, at least it should conserve solution quantities, as that is the most important physical law underlying the physical PDEs.

We use a finite volume-like formulation as 'solution flow' driven by moving mesh points:

$$Q_{j+1/2}^{[\nu+1]} = \frac{\left(x_{j+1}^{[\nu]} - x_j^{[\nu]}\right) Q_{j+1/2}^{[\nu]} - \left((cQ)_{j+1}^{[\nu]} - (cQ)_j^{[\nu]}\right)}{x_{j+1}^{[\nu+1]} - x_j^{[\nu+1]}}, \tag{3}$$

where $c_j = x_j^{[\nu+1]} - x_j^{[\nu]}$ denotes the mesh move speed for each point and ν counts the Gauss-Seidel/interpolation steps. This interpolation always conserves all solution components up to machine precision in the following way (cf.[2]):

$$\sum_j \Delta x_{j+\frac{1}{2}}^{[\nu+1]} Q_{j+\frac{1}{2}}^{[\nu+1]} = \sum_j \Delta x_{j+\frac{1}{2}}^{[\nu]} Q_{j+\frac{1}{2}}^{[\nu]}.$$

3.2. High resolution finite volume solver

As Algorithm 1 shows, the mesh update and PDE solver are two separate steps. We keep the adaptive mesh equation (2) and the physical PDEs (1) uncoupled, to keep the solver flexible, and to avoid problematic differences in times scales of the mesh and physical system.

The finite volume solver employs local Lax Friedrichs fluxes to keep numerical dissipation low. Solution values are defined at cell centers, the fluxes at the cell edges are computed by a MUSCL-type interpolation.

The time integration is performed by a second-order Runge-Kutta scheme, with the time stepsize limited by a locally determined Courant number.

3.3. A self-regulating monitor function

The main contribution of our work is the successful combination of the previously described mesh adaptation with a self-regulating monitor function. The monitor function $\omega(q)$ specifies the importance of parts of the solution, so that equidistribution (2) will pull more mesh points towards it. Monitor functions mostly contain solution gradients, but for different models and problems, the size of the gradients may differ enormously. The arc length-type monitor function

$$\omega(q) = \sqrt{1 + \alpha(\partial q/\partial x)^2}, \tag{4}$$

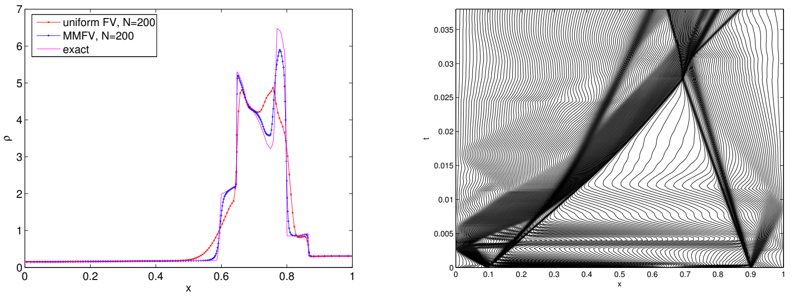

Figure 1: Colliding blast waves. Left: density profile by uniform and moving mesh method. Right: mesh history over $t \in [0, 0.038]$.

allows the user to 'correct' these differences by a scaling factor α for each problem. The problem with this monitor is its time-independence. Solutions that start smoothly may produce shock waves at some later time, and a different value for α would be needed then. Instead of introducing a scaling factor α, we replace the floor value 1 by a solution-dependent (hence time-dependent) value $\alpha(q)$. This floor value is set at the average of the current solution gradient on the entire domain:

$$\omega(q) = \sum_{p=1}^{m} \left[(1-\beta)\alpha_p(q) + \beta \left| \frac{\partial q_p}{\partial \xi} \right|^{1/2} \right], \text{ with } \alpha_p(q) = \int_{\Omega_c} \left| \frac{\partial q_p}{\partial \xi} \right|^{1/2} d\xi. \quad (5)$$

Note how all solution components q_p, $(q \in \mathbb{R}^m)$ are included. The parameter β specifies the ratio of points in important parts of the domain. The user does not have to set this parameter, we always keep it fixed at $\beta = 0.8$, i.e., approximately 80% of the points in important parts.

4. Experiments

In a recent paper [1] we show how well our moving mesh solver performs, for several, diverse problems from MHD, including comparisons with third-party packages. The next section shows new results.

4.1. Colliding blast waves

Basic shock tube examples are often used for systems of conservation laws. Woodward [4] proposed a more challenging problem, where *two* blast waves collide. It is based on the hydrodynamical laws, i.e., the 1D Euler equations.

Initially, the fluid is at rest on the domain $x \in [0, 1]$ between two reflective walls. Density is unity everywhere, and the fluid is hot in two small regions of length 0.1 at both sides. At the left the pressure is 1000, at the right it is 100, elsewhere the fluid is cold with, pressure 0.01.

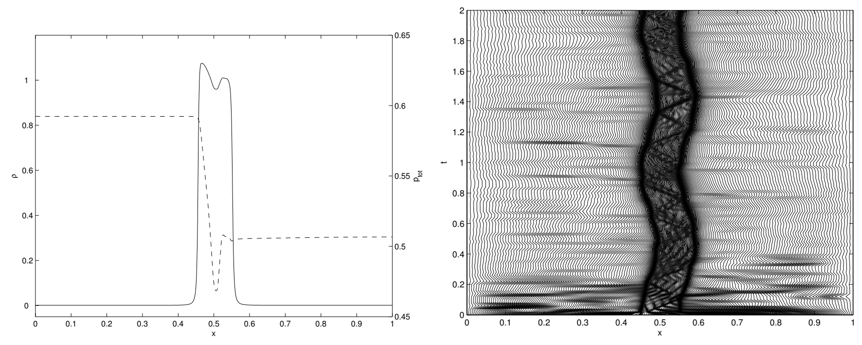

Figure 2: Oscillating plasma sheet. Left: density (solid line) and total pressure (dashed line) at $t = 1$. Right: mesh history over $t \in [0, 2]$.

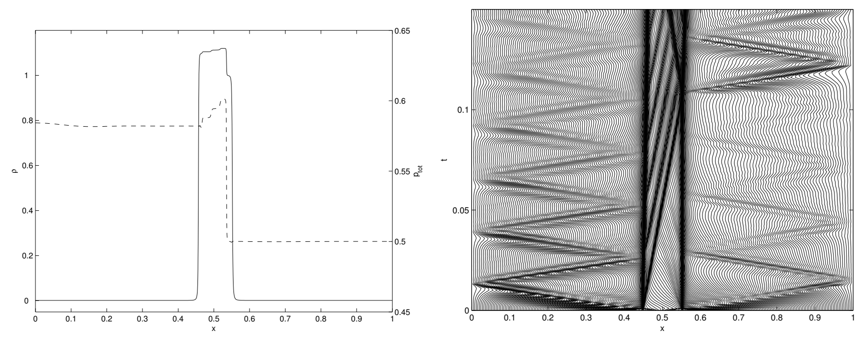

Figure 3: Oscillating plasma sheet, initial details. Left: density (solid line) and total pressure (dashed line) at $t = 0.1$. Right: mesh history over $t \in [0, 0.15]$.

Figure 1 shows uniform and moving mesh results, compared to the exact solution. The right diagram shows the mesh history, which shows that all shock waves and rarefactions have been properly detected (cf. [4, Fig. 1]).

4.2. Physical staircasing in plasma sheet

This example is a deeper investigation of the 1.5D MHD oscillating plasma sheet problem introduced by Tóth et al. [3]. A high density, high pressure plasma sheet is surrounded by a vacuum and reflective walls on both sides. An initial pressure imbalance brings the plasma sheet in motion. Because of conservation of magnetic flux in the left and right vacuum, this will result in an ongoing oscillation of the sheet.

Figure 2 shows how the oscillation indeed sets in, and the mesh refinement clearly tracks the moving sides of the plasma sheet. The amplitude and period of the oscillation in the results are in good agreement with theoretically predicted values.

Physical staircasing Apart from the global oscillation that sets in, the mesh history (right diagram in Figure 2) shows additional oscillations within the plasma sheet itself. Each side of the plasma sheet has formed a local Riemann problem with the neighbouring vacuum. A fast magnetosonic shock wave results in both vacuum parts, which keeps on being reflected between the wall and the plasma side. Each time the fast wave hits the plasma sheet, a new local Riemann problem results in a slight decrease in density in the sheet. This recurring mechanism forms a profile resembling a staircase within the plasma sheet, see Figure 3. The right diagram clearly shows the bouncing fast wave causing new shocks in the plasma sheet each time.

Experiments by others have not shown this behaviour before, which is partly due to the use of implicit solvers: the large time steps can not resolve the small time scales at which the staircase formation occurs. Besides, our method automatically tracks changes in the solution, even the small local ones. No prior knowledge from the user is required to steer mesh adaptation.

5. Conclusion

The use of a self-regulating monitor function in a moving mesh method makes it much more powerful, as the method now detects phenomena that the researcher may not have been aware of. We are currently combining this with a two-dimensional adaptive solver. Here, much more gain compared to uniform methods should be possible.

References

[1] A. van Dam, P.A. Zegeling, *A Robust Moving Mesh Finite Volume Method applied to 1D Hyperbolic Conservation Laws from Magnetohydrodynamics*, to appear in J. Comput. Phys., 2006. doi:10.1016/j.jcp.2005.12.014

[2] Huazhong Tang and Tao Tang. Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws. *SIAM J. Numer. Anal.*, 41(2):487–515 (electronic), 2003.

[3] G. Tóth, R. Keppens, and M. A. Botchev. Implicit and semi-implicit schemes in the Versatile Advection Code: numerical tests. *Astron. & Astroph.*, 332:1159–1170, April 1998.

[4] Paul R. Woodward. Trade-offs in designing explicit hydrodynamics schemes for vector computers. In G. Rodrigue, editor, *Parallel Computations*, volume 1, pages 153–171. Academic Press, New York, 1982.