AN ADAPTIVE MESH REDISTRIBUTION ALGORITHM FOR CONVECTION-DOMINATED PROBLEMS

ZHENGRU ZHANG † AND TAO TANG †

Abstract. Convection-dominated problems are of practical applications and in general may require extremely fine meshes over a small portion of the physical domain. In this work an efficient adaptive mesh redistribution (AMR) algorithm will be developed for solving one- and two-dimensional convection-dominated problems. Several test problems are computed by using the proposed algorithm. The adaptive mesh results are compared with those obtained with uniform meshes to demonstrate the effectiveness and robustness of the proposed algorithm.

1. **Introduction**

Adaptive mesh redistribution (AMR) methods have important applications in a variety of physical and engineering areas such as solid and fluid dynamics, combustion, heat transfer, material science etc. The physical phenomena in these areas develop dynamically singular or nearly singular solutions in fairly localized regions, such as shock waves, boundary layers, detonation waves etc. The numerical investigation of these physical problems may require extremely fine meshes over a small portion of the physical domain to resolve the large solution variations. One class of such problems is the convection-dominated problems, including viscous shocks and large Reynolds number incompressible flows [5, 7, 8]. In this work, we will develop an efficient and robust AMR algorithm to solve convection-diffusion problems with small viscosity.

It is a challenging problem to generate an efficient AMR algorithm for two or more dimensional problems, especially when the underlying solution develops complicated structures and becomes singular or nearly singular. The earliest work on adaptive methods, based on moving finite element approach (MFEM) was done by Millers [13]. There are many applications and extensions of Miller's moving finite element methods, see e.g. Baines [1], Cao et al. [3], and Moore and Flaherty [14]. On the other hand, several moving mesh techniques have been introduced based on solving elliptic PDEs first proposed by Winslow [19]. There are also many applications and extensions of Winslow's method, see e.g. Brackbill and Saltzman [2], Ceniceros and Hou [4], Thompson et al. [18], and Ren and Wang [15]. Winslow's formulation requires the solution of a nonlinear, Poisson-like equation to generate a mapping from a regular domain in a parameter space to an irregularly shaped domain in physical space. By connecting points in the physical space corresponding to discrete points in the parameter space, the physical domain can be covered with a mesh suitable for the solution of finite difference/element equations.

Date: March 4, 2002.

‡ Department of Mathematics, The Hong Kong Baptist University, Kowloon Tong, Hong Kong. Email: `ttang@math.hkbu.edu.hk`, `zrzhang@math.hkbu.edu.hk`.

Typically, the map transforms a uniform mesh in the logical domain to cluster grid points at the regions of the physical domain where the solution has the largest gradients.

A class of efficient AMR methods are based on the so-called harmonic mappings. Dvinsky [6] suggests the possibility that harmonic function theory may provide a general framework for developing useful mesh generators. His method can be viewed as a generalization and extension of Winslow's method. However, unlike most other generalizations which add terms or functionals to the basic Winslow grid generator, his approach uses a single functional to accomplish the adaptive mapping. Recently, Li et al. [11] introduced a general AMR scheme based on the harmonic mapping. The numerical algorithm contains two independent parts, namely a solution algorithm and a mesh-redistribution algorithm. Using this framework, adaptive mesh solutions for singular problems in two and *three* space dimensions have been obtained successfully [12].

The main objective of this work is to develop an efficient adaptive mesh algorithm for convection-diffusion equations with very small viscosity. Our numerical method is based on an extension of a recent work of Tang and Tang [17], where an adaptive mesh redistribution algorithm was developed for solving multi-dimension hyperbolic problems. Their algorithm is again formed by two independent parts: a PDE evolution part and a mesh-redistribution part. The first part can be any appropriate high resolution schemes, and the second part is to solve a nonlinear elliptic equation by the Gauss-Seidel (GS) iterations. The key idea in [17] is to employ a conservative-interpolation such that mass-conservation of the underlying numerical solution is preserved at each re-distribution step. The difference between the present problems and the one considered in [17] is that a diffusion term is now involved, which (although with a small coefficient) creates some additional numerical difficulty. This work is to provide simple but efficient AMR algorithms for solving 1D and 2D convection dominated problems.

2. Adaptive mesh Algorithm in 1D

To begin with, let us consider a simple one-dimensional convection-diffusion equation:

$$u_t + f(u)_x = \epsilon(\sigma(u)u_x)_x \tag{2.1}$$

where $0 < \epsilon \ll 1$ is a (small) viscosity coefficient, $\sigma > 0$. Our AMR scheme consists of two independent parts: a PDE evolution and a mesh-redistribution, and a detailed solution flowchart was presented in [17]. An overview of the sequence of computations is given in Table 1.

We first describe a PDE evolution algorithm based on a finite volume approach, which will be used in Step 2 of Table 1. Assume we already have a partition of the physical domain and denote

$$x_{i+\frac{1}{2}} = \frac{1}{2}(x_i + x_{i+1}) \qquad I_{i+\frac{1}{2}} = [x_i, x_{i+1}], \qquad u_{i+\frac{1}{2}} = \frac{1}{|I_{i+\frac{1}{2}}|}\int_{I_i} u(x)dx\,.$$

Integrating the above equation over the cell $I_{i+\frac{1}{2}} \times [t^n, t^{n+1}]$ leads to the following numerical scheme

$$u^{n+1}_{i+\frac{1}{2}} = u^n_{i+\frac{1}{2}} - \frac{\Delta t^n}{|I_{i+\frac{1}{2}}|}(\bar{f}^n_{i+1} - \bar{f}^n_i) + \epsilon((\sigma(u)u_x)^n_{i+1} - (\sigma(u)u_x)^n_i) \tag{2.2}$$

Table 1. Outline of the numerical algorithm

Initial variables

0. Determine the initial mesh based on the initial function.

1. Determine Δt based on CFL-type condition so that $t^{n+1} = t^n + \Delta t$.

2. Advance the solution one time step based on an appropriate numerical scheme.

3. Grid Restructuring

a. Solve the mesh redistributing equation (a generalized Laplacian equation) by one Gauss-Seidel iteration, to get $\mathbf{x}^{(k),n}$

b. Interpolating the approximate solutions on the new grid $\mathbf{x}^{(k),n}$

c. A weighted average of the locally calculated monitor at each computational cell and the surrounding monitor values.

d. The iteration procedure (a.)-(c.) on grid-motion and solution-interpolation is continued until there is no significant change in calculated new grids from one iteration to the next.

Start new time step (go to **1** above).

---

where $\Delta t^n = t^{n+1} - t^n$, $\bar{f}_i = \bar{f}(u_i^-, u_i^+)$ is the numerical flux. In this paper, we adopt the simple and inexpensive Lax-Friedrichs flux:

$$\bar{f}(a,b) = \frac{1}{2}[f(a) + f(b) - \max|f_u| \cdot (b-a)] \tag{2.3}$$

where the maximum is taken between a and b. Godunov flux and Engquist-Osher flux can also be applied here. In order to approximate the flux, we reconstruct a linear approximation in each cell:

$$u_i^- = u_{i-\frac{1}{2}} + \frac{1}{2}s_{i-\frac{1}{2}}(x_i - x_{i-\frac{1}{2}}), \qquad u_i^+ = u_{i+\frac{1}{2}} + \frac{1}{2}s_{i+\frac{1}{2}}(x_i - x_{i+\frac{1}{2}}) \tag{2.4}$$

where

$$s^+_{i+\frac{1}{2}} = \frac{u_{i+\frac{3}{2}} - u_{i+\frac{1}{2}}}{x_{i+\frac{3}{2}} - x_{i+\frac{1}{2}}}, \qquad s^-_{i+\frac{1}{2}} = \frac{u_{i+\frac{1}{2}} - u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}}$$

$$s_{i+\frac{1}{2}} = \Big(\mathrm{sign}(s^-_{i+\frac{1}{2}}) + \mathrm{sign}(s^+_{i+\frac{1}{2}})\Big)\frac{|s^-_{i+\frac{1}{2}} \cdot s^+_{i+\frac{1}{2}}|}{|s^-_{i+\frac{1}{2}}| + |s^+_{i+\frac{1}{2}}|}\,.$$

We apply the following standard scheme to handle the diffusion term

$$u_i \approx \frac{(x_i - x_{i-\frac{1}{2}})u_{i+\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} + \frac{(x_{i+\frac{1}{2}} - x_i)u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}}, \qquad (u_x)_i \approx \frac{u_{i+\frac{1}{2}} - u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}}\,. \tag{2.5}$$

Remark 2.1. It is noted that the approximation formula for the first derivative, (2.5), is of first order only. In the interior layer region, this approach is not accurate enough. However, we found in our numerical experiments that this first-order approach is sufficient for 1D problems, although its extension to 2D is unsatisfactory.

The fully discretized numerical scheme (2.2)-(2.4) yields a semi-discretized difference equations (i.e. method of line) which will be solved by a 3-stage Runge-Kutta method proposed by Shu and Osher [16]. A two-dimensional 3-stage RK method will be given in Section 4.

We further describe the mesh redistribution at each time step. The mesh generation equation, based on the standard equidistribution principle, is

$$(\omega x_\xi)_\xi = 0, \qquad \xi \in [0,1], \tag{2.6}$$

where the function ω is called monitor function which in general depends on the underlying solution to be adapted and is an indicator of the degree of singularity. The above equation is solved in the computational domain $[0,1]$ with an uniform mesh.

The final part of this section is to discuss the solution-updating on the new mesh, i.e. Step 3(b) in Table 1. After obtaining the new mesh $\{\tilde{x}_j\}$ from (2.6), we need to update the numerical solution on the new points $\tilde{x}_{j+\frac{1}{2}} = (\tilde{x}_j + \tilde{x}_{j+1})/2$. In Tang and Tang [17], a second-order conservative interpolation formula is introduced. This interpolation formula does not increase the total variation, and as a result the resulting adaptive mesh solutions satisfy several fundamental properties for the hyperbolic conservation laws. For the convection-dominated problems, we will use the same conservative interpolation formula:

$$\Delta\tilde{x}_{j+\frac{1}{2}}\tilde{u}_{j+\frac{1}{2}} = \Delta x_{j+\frac{1}{2}} u_{j+\frac{1}{2}} - ((cu)_{j+1} - (cu)_j) \tag{2.7}$$

where $\Delta\tilde{x}_{j+\frac{1}{2}} = \tilde{x}_{j+1} - \tilde{x}_j, c_j = x_j - \tilde{x}_j$. The information on the right hand-side, i.e. $\tilde{x}_{j+\frac{1}{2}}$ (new mesh), $x_{j+\frac{1}{2}}$ (old mesh), and $u_{j+\frac{1}{2}}$ (numerical solution on the old mesh), are all available information. In the actual computation, the linear flux cu is approximated by a up-winding scheme, see [17]:

$$(\widehat{cu})_j = \frac{c_j}{2}(u_{j+\frac{1}{2}} + u_{j-\frac{1}{2}}) - \frac{|c_j|}{2}(u_{j+\frac{1}{2}} - u_{j-\frac{1}{2}}). \tag{2.8}$$

3. NUMERICAL EXPERIMENTS IN 1D

In this section, numerical experiments will be carried out to demonstrate the effectiveness of the AMR algorithm proposed in the last section. Some 1D convection-dominated problems will be considered. Most test problems here are taken from Kurganov and Tadmor [10].

Example 3.1. (One-dimensional viscous Burgers' equation) Our first problem is a simple Burgers' equation

$$u_t + \left(\frac{u^2}{2}\right)_x = \epsilon u_{xx} \qquad x \in (-2,2) \tag{3.1}$$

with boundary conditions $u(2,t) = 0, u(-2,t) = 1$ and initial condition

$$u(x,0) = \begin{cases} 1 & x \le 0 \\ 0 & x > 0. \end{cases}$$

Adaptive solutions are obtained for the above Burgers' problem with $\epsilon = 0.005$ and 0.001, up to $t = 2.5$. The standard CFL constant is chosen as 0.4, and the monitor function used is $\sqrt{1 + 80u_\xi^2}$. In the solution domain $[-2,2]$, 25 grid points are used, which in the uniform mesh case corresponds to quite large mesh size. In Fig. 1, the mesh trajectory for $0 \le t \le 2.5$ is plotted. As desired, considerable portion of grid points has been moved to steep gradient regions, which increases the resolution of the numerical

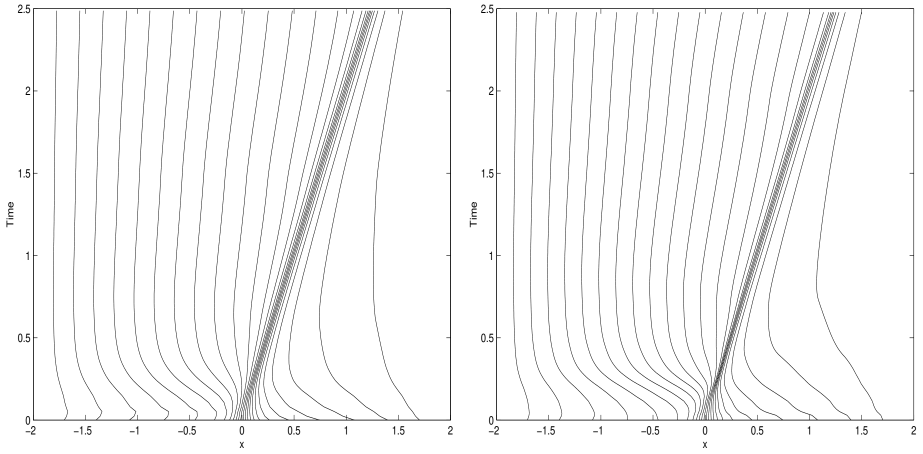

FIGURE 1. Example 3.1: adaptive mesh trajectory with 25 grid points. Left for $\epsilon = 0.005$ and right for $\epsilon = 0.001$.

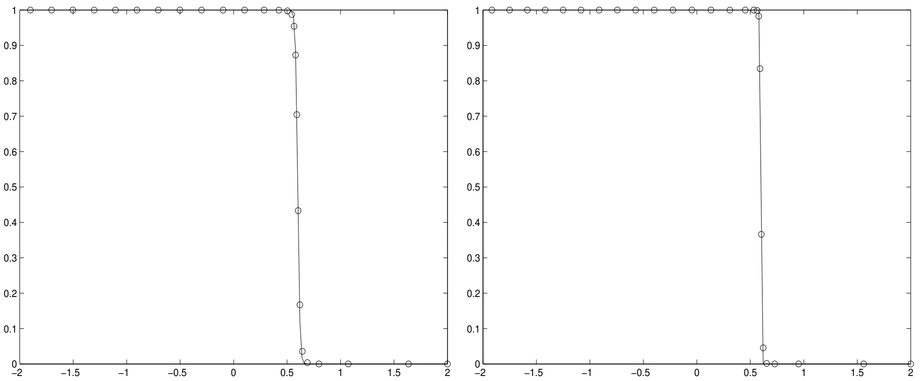

FIGURE 2. Example 3.1: adaptive mesh solution (o) at $t = 1.2$ and the exact solution (solid line). Left for $\epsilon = 0.005$ and right for $\epsilon = 0.001$.

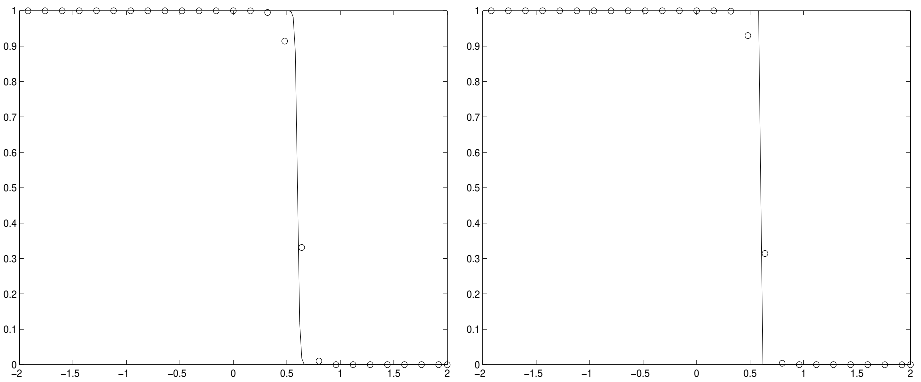

FIGURE 3. Same as Fig. 2, except for uniform mesh solution.

solutions. In Fig. 2, numerical solutions are also compared with the exact solution at $t = 1.2$, and very good agreement is observed. For comparison, we also plot in Fig. 3 the uniform mesh results. The accuracy improvement for the AMR algorithm can be clearly seen from these two figures.

Remark 3.1. It should be pointed out that the trajectory points in Fig. 1 refer to as grid points x_j, while the cell-average values plotted in Fig. 2 are at $\tilde{x}_j := (x_{j-1} + x_j)/2, 1 \leq j \leq N$. Same notations are used in other figures in the remaining of this section.

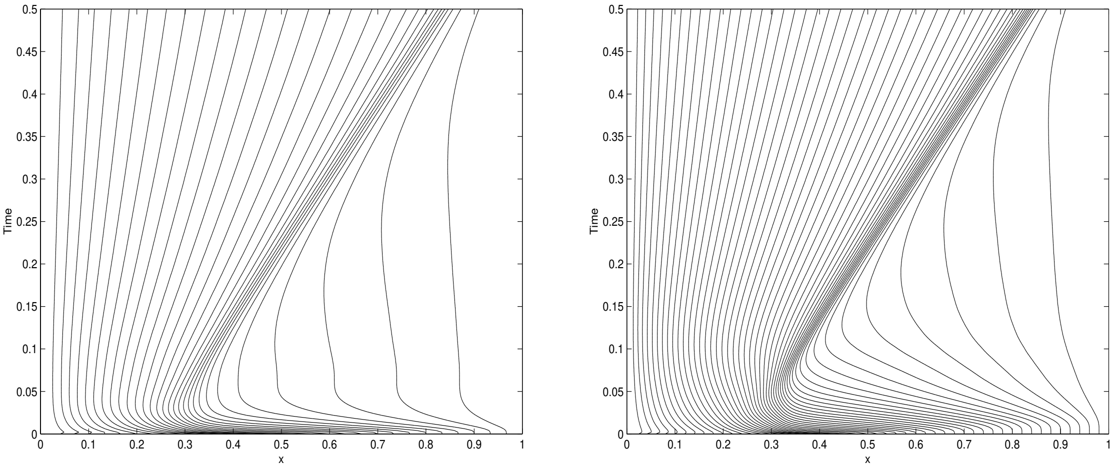

FIGURE 4. Example 3.2: adaptive mesh trajectory with $N = 30$ (left) and 50 (right).

Example 3.2. (One-dimensional Buckley-Leverett equation) Consider a prototype model for oil reservoir simulations (two-phase) flow:

$$u_t + f(u)_x = \epsilon(\sigma(u)u_x)_x \,. \tag{3.2}$$

Typically, $\sigma(u)$ vanished at some values of u, and (3.2) is a degenerate parabolic equation. In our computation, we take ϵ to be 0.01, $f(u)$ to have an s-shaped form

$$f(u) = \frac{u^2}{u^2 + (1-u)^2} \,, \tag{3.3}$$

and $\sigma(u)$ vanish at $u = 0$ and 1:

$$\sigma(u) = 4u(1-u). \tag{3.4}$$

The initial function is

$$u(x,0) = \begin{cases} 1 - 3x & 0 \leq x \leq \frac{1}{3} \\ 0 & \frac{1}{3} < x \leq 1, \end{cases}$$

and the boundary value of $u(0,t) = 1$ is kept fixed.

This problem was solved numerically with several numerical techniques, including the central schemes of Kurganov and Tadmor [10] and the operator splitting methods of Karlsen et al. [9]. In Fig. 4, the trajectories of the grid points up to $t = 0.5$ are presented, obtained with $N = 30$ and 50. The monitor function used is $\sqrt{1 + 50u_\xi^2}$. The ability of the AMR method to capture and follow the moving large gradients is clearly demonstrated in this figure. The numerical solutions at $t = 0.2$ with 30 and 50 grid points are shown in Fig. 5. It is seen from these results that this stiff problem can be well resolved by using

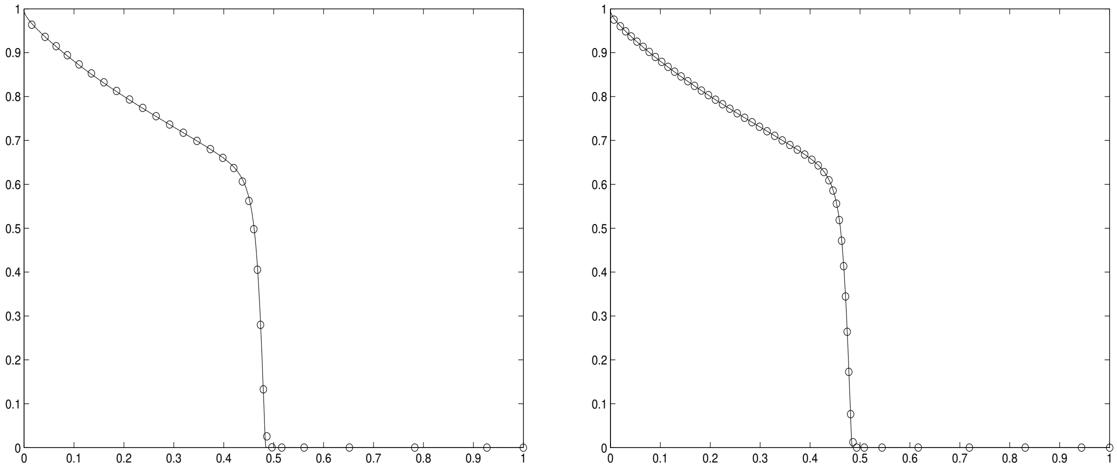

FIGURE 5. Example 3.2: adaptive solution (o) at $t = 0.2$ and the exact solution (–). $N = 30$ (left) and 50 (right).

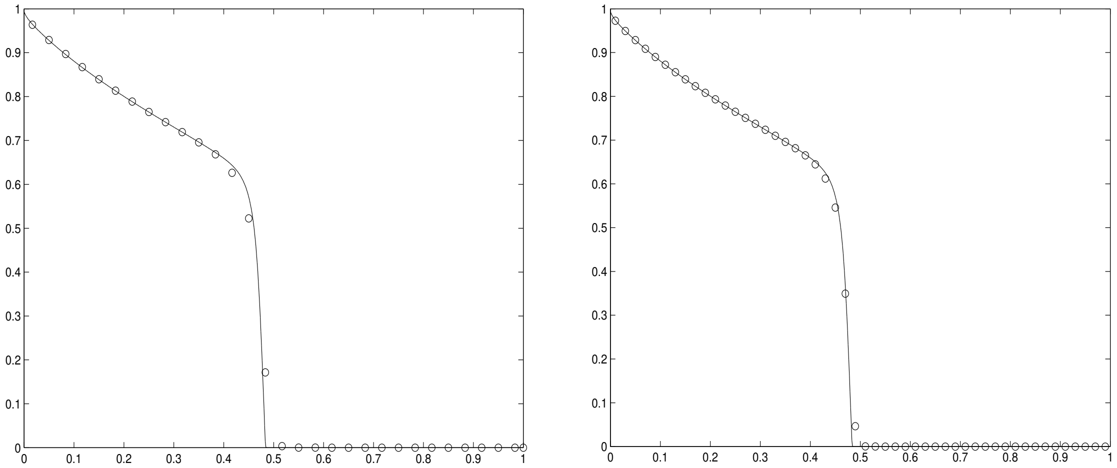

FIGURE 6. Same as Fig. 5, except for uniform mesh solution.

30 grid points. For comparison, Fig. 6 presents the uniform mesh results, which again demonstrate the great improvement in accuracy for the AMR method.

Example 3.3. (Gravitational Effects). Consider the Buckley-Leveret equation (3.2), with the same ϵ and the diffusion coefficient as in Example 3.2. However, the flux function now includes gravitational effects:

$$f(u) = \frac{u^2}{u^2 + (1-u)^2}(1 - 5(1-u)^2). \tag{3.5}$$

The initial condition is given by the Riemann data

$$u(x,0) = \begin{cases} 0, & 0 \le x < 1 - 1/\sqrt{2}\,, \\ 1, & 1 - 1/\sqrt{2} \le x \le 1. \end{cases}$$

As pointed in [10], this problem is more complicated than the previous one since we have to handle the flux (3.5) where $f'(u)$ changes sign. Indeed, it is found that more grid points are required to resolve this problem. Figs. 7 and 8 present numerical solutions

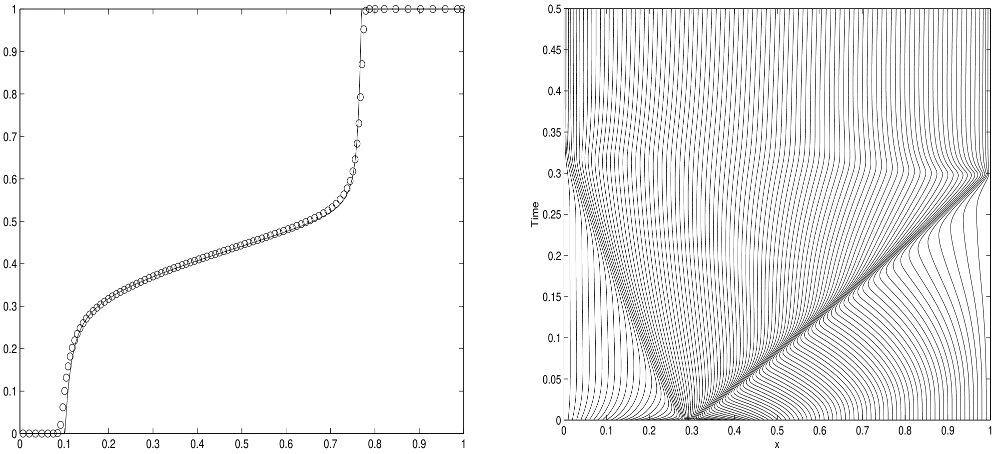

FIGURE 7. Example 3.3 with gravitation: adaptive solution (left) and the mesh trajectory (right) with $N = 100$. $T = 0.2$.

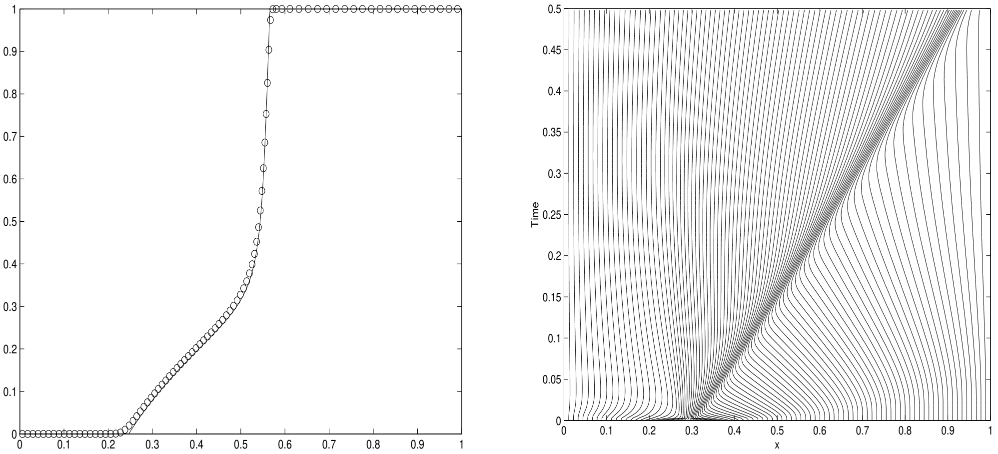

FIGURE 8. Example 3.3 without gravitation: adaptive solution (left) and the mesh trajectory (right) with $N = 90$. $T = 0.2$.

for this problem with (i.e. flux (3.5) and without (i.e. flux (3.3) the gravitational effects, respectively, at $t = 0.2$. The corresponding mesh trajectories are also plotted for time up to $t = 0.5$. The solid lines are numerical solutions with a uniform mesh of 800 grid points. The monitor function used is $\sqrt{1 + u_\xi^2}$ for this example.

4. AMR METHOD IN 2D

Without lose of generality, we consider in this section the following 2D convection dominated equation

$$u_t + f(u)_x + g(u)_y = \epsilon \Delta u \qquad (x, y) \in \Omega_p \tag{4.1}$$

where $0 < \epsilon \ll 1$ is the (small) viscosity coefficient, Δ is the standard Laplacian operator, Ω_p is the physical domain. The AMR method in 2D also follows the same procedure as given in Table 1. As mentioned in Remark 2.1, it is difficult to provide a second-order approximation for the first order derivatives in a compact stencil setting so we will use a transformed equation for 2D. In contrast to the hyperbolic system of conservation laws, the solutions for the PDE (4.1) are smooth so this approach seems reasonable. Using the following transformation formulas:

$$\begin{aligned}
u_x &= \frac{1}{J}\left[(y_\eta u)_\xi - (y_\xi u)_\eta\right],\\
u_y &= \frac{1}{J}\left[-(x_\eta u)_\xi + (x_\xi u)_\eta\right],\\
u_{xx} &= \frac{1}{J}\left[(J^{-1}y_\eta^2 u_\xi)_\xi - (J^{-1}y_\xi y_\eta u_\eta)_\xi - (J^{-1}y_\xi y_\eta u_\xi)_\eta + (J^{-1}y_\xi^2 u_\eta)_\eta\right],\\
u_{xy} &= \frac{1}{J}\left[-(J^{-1}x_\eta y_\eta u_\xi)_\xi + (J^{-1}x_\xi y_\eta u_\eta)_\xi + (J^{-1}x_\eta y_\xi u_\xi)_\eta - (J^{-1}x_\xi y_\xi u_\eta)_\eta\right],\\
u_{yy} &= \frac{1}{J}\left[(J^{-1}x_\eta^2 u_\xi)_\xi - (J^{-1}x_\xi x_\eta u_\eta)_\xi - (J^{-1}x_\xi x_\eta u_\xi)_\eta + (J^{-1}x_\xi^2 u_\eta)_\eta\right]
\end{aligned}$$

where $J = x_\xi y_\eta - x_\eta y_\xi$ is the Jacobian of the coordinate transformation. Using the above formulas, the underlying equation (4.1) becomes:

$$\begin{aligned}
& u_t + \frac{1}{J}\Big(y_\eta f(u) - x_\eta g(u)\Big)_\xi + \frac{1}{J}\Big(x_\xi g(u) - y_\xi f(u)\Big)_\eta \\
= \; & \frac{\epsilon}{J}\Big[\Big(J^{-1}(y_\eta^2 u_\xi + x_\eta^2 u_\xi - y_\xi y_\eta u_\eta - x_\xi x_\eta u_\eta)\Big)_\xi \\
& + \Big(J^{-1}(y_\xi^2 u_\eta + x_\xi^2 u_\eta - y_\xi y_\eta u_\xi - x_\xi x_\eta u_\xi)\Big)_\eta\Big] \qquad (\xi,\eta) \in \Omega_c
\end{aligned} \tag{4.2}$$

where Ω_c is the computational domain with fixed (uniform) mesh (ξ_j, η_k). For convenience, we write the above equation in a simpler form:

$$u_t + \frac{1}{J}F(u)_\xi + \frac{1}{J}G(u)_\eta = \frac{\epsilon}{J}[\mathsf{R}_\xi + \mathsf{S}_\eta] \tag{4.3}$$

We will solve the above equation again by a finite-volume approach. Denote the control cell $[\xi_j\,, \xi_{j+1}] \times [\eta_k\,, \eta_{k+1}]$ by $A_{j+\frac{1}{2},k+\frac{1}{2}}$ and the cell average value by

$$\bar{u}^n_{j+\frac{1}{2},k+\frac{1}{2}} = \frac{1}{\Delta\xi\Delta\eta}\int_{A_{j+\frac{1}{2},k+\frac{1}{2}}} u(\xi,\eta,t^n)\,d\xi d\eta\,.$$

We use a cell-center finite volume method to discretize (4.3), and the following trick will be used to obtain a simple finite volume scheme:

$$\begin{aligned}
& \frac{1}{\Delta\xi\Delta\eta}\int_{\xi_j}^{\xi_{j+1}}\int_{\eta_k}^{\eta_{k+1}} \frac{1}{J} w_\xi \, d\xi d\eta \\
= \; & \frac{1}{\Delta\xi\Delta\eta}\frac{1}{J_{j+\frac{1}{2},k+\frac{1}{2}}}\int_{\xi_j}^{\xi_{j+1}}\int_{\eta_k}^{\eta_{k+1}} w_\xi \, d\xi d\eta + \mathcal{O}(\Delta\xi^2) \\
= \; & \frac{1}{J_{j+\frac{1}{2},k+\frac{1}{2}}}\Big(w_{j+1,k+\frac{1}{2}} - w_{j,k+\frac{1}{2}}\Big) + \mathcal{O}(\Delta\xi^2)
\end{aligned}$$

where a mid-point role is used in the first step. Similar approach can treat terms involving $J^{-1}w_\eta$. Integrating equation (4.3) over the cell $[t^n, t^{n+1}] \times A_{j+\frac{1}{2},k+\frac{1}{2}}$ in the computational domain leads to

$$\begin{aligned}\bar{u}^{n+1}_{j+\frac{1}{2},k+\frac{1}{2}} &= \bar{u}^{n}_{j+\frac{1}{2},k+\frac{1}{2}} - \frac{\Delta t^n}{\Delta\xi\Delta\eta}\frac{1}{J_{j+\frac{1}{2},k+\frac{1}{2}}}\left(\bar{F}^n_{j+1,k+\frac{1}{2}} - \bar{F}^n_{j,k+\frac{1}{2}} + \bar{G}^n_{j+\frac{1}{2},k+1} - \bar{G}^n_{j+\frac{1}{2},k}\right)\\ &\quad + \frac{\epsilon}{J_{j+\frac{1}{2},k+\frac{1}{2}}}\left(\mathsf{R}^n_{j+1,k+\frac{1}{2}} - \mathsf{R}^n_{j,k+\frac{1}{2}} + \mathsf{S}^n_{j+\frac{1}{2},k+1} - \mathsf{S}^n_{j+\frac{1}{2},k}\right) \qquad (4.4)\end{aligned}$$

The one-dimensional Lax-Friedrichs numerical flux will be applied to $\bar{F}$, $\bar{G}$ in ξ-, η-direction, respectively:

$$\bar{F}_{j,k+\frac{1}{2}} = \bar{F}(u^-_{j,k+\frac{1}{2}}, u^+_{j,k+\frac{1}{2}}), \qquad \bar{G}_{j,k+\frac{1}{2}} = \bar{G}(u^-_{j,k+\frac{1}{2}}, u^+_{j,k+\frac{1}{2}}) \qquad (4.5)$$

The 1D Lax-Friedrichs flux for $\bar{F}$ and $\bar{G}$ is the same as (2.3). In (4.5), we use

$$\begin{aligned}u^-_{j,k+\frac{1}{2}} &= \bar{u}_{j-\frac{1}{2},k+\frac{1}{2}} + \frac{\Delta\xi}{2}s_{j-\frac{1}{2},k+\frac{1}{2}}\,, \qquad u^+_{j,k+\frac{1}{2}} = \bar{u}_{j+\frac{1}{2},k+\frac{1}{2}} - \frac{\Delta\xi}{2}s_{j+\frac{1}{2},k+\frac{1}{2}}\\ s_{j+\frac{1}{2},k+\frac{1}{2}} &= \left(\mathrm{sign}(s^-_{j+\frac{1}{2},k+\frac{1}{2}}) + \mathrm{sign}(s^+_{j+\frac{1}{2},k+\frac{1}{2}})\right)\frac{|s^+_{j+\frac{1}{2},k+\frac{1}{2}}s^-_{j+\frac{1}{2},k+\frac{1}{2}}|}{|s^+_{j+\frac{1}{2},k+\frac{1}{2}}| + |s^-_{j+\frac{1}{2},k+\frac{1}{2}}|}\\ s^-_{j+\frac{1}{2},k+\frac{1}{2}} &= \frac{\bar{u}_{j+\frac{1}{2},k+\frac{1}{2}} - \bar{u}_{j-\frac{1}{2},k+\frac{1}{2}}}{\Delta\xi} \qquad s^+_{j+\frac{1}{2},k+\frac{1}{2}} = \frac{\bar{u}_{j+\frac{3}{2},k+\frac{1}{2}} - \bar{u}_{j+\frac{1}{2},k+\frac{1}{2}}}{\Delta\xi}\,.\end{aligned}$$

For R and S terms in (4.3), the coordinate derivatives are also involved. We use the standard central differencing to discretize them, such as

$$\begin{aligned}(x_\xi)_{j,k+\frac{1}{2}} &= \frac{1}{2}\Big[\frac{x_{j+1,k} - x_{j-1,k}}{2\Delta\xi} + \frac{x_{j+1,k+1} - x_{j-1,k+1}}{2\Delta\xi}\Big]\\ (x_\eta)_{j,k+\frac{1}{2}} &= \frac{x_{j,k+1} - x_{j,k}}{\Delta\eta}\\ (x_\xi)_{j+\frac{1}{2},k+\frac{1}{2}} &= \frac{1}{2}\Big[\frac{x_{j+1,k} - x_{j,k}}{\Delta\xi} + \frac{x_{j+1,k+1} - x_{j,k+1}}{\Delta\xi}\Big]\\ (x_\eta)_{j+\frac{1}{2},k+\frac{1}{2}} &= \frac{1}{2}\Big[\frac{x_{j,k+1} - x_{j,k}}{\Delta\eta} + \frac{x_{j+1,k+1} - x_{j+1,k}}{\Delta\eta}\Big]\,.\end{aligned}$$

For the diffusion terms in the right hand side of (4.3), central differencing is also used, for example:

$$\begin{aligned}(u_\xi)_{j,k+\frac{1}{2}} &= \frac{\bar{u}_{j+\frac{1}{2},k+\frac{1}{2}} - \bar{u}_{j-\frac{1}{2},k+\frac{1}{2}}}{\Delta\xi}\\ (u_\eta)_{j,k+\frac{1}{2}} &= \frac{1}{2}\Big[\frac{\bar{u}_{j+\frac{1}{2},k+\frac{3}{2}} - \bar{u}_{j+\frac{1}{2},k-\frac{1}{2}}}{2\Delta\eta} + \frac{\bar{u}_{j-\frac{1}{2},k+\frac{3}{2}} - \bar{u}_{j-\frac{1}{2},k-\frac{1}{2}}}{2\Delta\eta}\Big]\,.\end{aligned}$$

The fully discretized numerical scheme (4.4) yields a semi-discretized difference equations (i.e. method of line) which will be solved by a 3-stage Runge-Kutta method proposed by Shu and Osher [16]. For the ODE system $u'(t) = L(u)$, we use

$$\begin{aligned}u^{(1)}_{jk} &= u^n_{jk} + \Delta t L(u^n_{jk})\\ u^{(2)}_{jk} &= \frac{3}{4}u^n_{jk} + \frac{1}{4}\left[u^{(1)}_{jk} + \Delta t L(u^{(1)}_{jk})\right]\\ u^{n+1}_{jk} &= \frac{1}{3}u^n_{jk} + \frac{2}{3}\left[u^{(2)}_{jk} + \Delta t L(u^{(2)}_{jk})\right]\,.\end{aligned}$$

We now describe the mesh redistribution in two space dimension. It is assumed that a fixed (square uniform) mesh is given on the computational domain. The mesh generation equation is of the form

$$\begin{aligned} \partial_\xi(G_1\partial_\xi x) + \partial_\eta(G_1\partial_\eta x) = 0, \\ \partial_\xi(G_2\partial_\xi y) + \partial_\eta(G_2\partial_\eta y) = 0. \end{aligned}$$

where G_1, G_2 are monitor functions. In our computations, we choose very simple monitor functions so that the mesh generation equation becomes

$$\tilde{\nabla} \cdot (\omega\tilde{\nabla}x) = 0, \qquad \tilde{\nabla} \cdot (\omega\tilde{\nabla}y) = 0 \tag{4.6}$$

where $\tilde{\nabla} = (\partial_\xi, \partial_\eta)^T$. In our 2D computations, we again use the conservative interpolation proposed by Tang and Tang [17]

$$\begin{aligned} |\tilde{A}_{j+\frac{1}{2},k+\frac{1}{2}}|\tilde{u}_{j+\frac{1}{2},k+\frac{1}{2}} &= |A_{j+\frac{1}{2},k+\frac{1}{2}}|u_{j+\frac{1}{2},k+\frac{1}{2}} \\ &- \left[(c^x u)_{j+1,k+\frac{1}{2}} - (c^x u)_{j,k+\frac{1}{2}}\right] - \left[(c^y u)_{j+\frac{1}{2},k+1} - (c^y u)_{j+\frac{1}{2},k}\right] \end{aligned} \tag{4.7}$$

where $c^x_{j,k} = x_{j,k} - \tilde{x}_{j,k}$, $c^y_{j,k} = y_{j,k} - \tilde{y}_{j,k}$.

5. Numerical experiments in 2D

In this section, numerical experiments in 2D will be carried out to demonstrate the effectiveness of the AMR algorithm proposed in this work. Throughout this section, the time step used satisfies

$$\Delta t = \lambda \min_{j,k} \left(\frac{x_{j+1,k} - x_{j,k}}{|f'(u_{j+\frac{1}{2},k+\frac{1}{2}})|}, \frac{y_{j,k+1} - y_{j,k}}{|f'(u_{j+\frac{1}{2},k+\frac{1}{2}})|} \right), \tag{5.1}$$

where λ is called a CFL constant.

Example 5.1. (Two-dimensional Burgers equations). Our first 2D problem is concerned with the two-dimensional Burgers' equation in the unit square:

$$u_t + uu_x + uu_y = \epsilon\Delta u \qquad 0 \le x, y \le 1. \tag{5.2}$$

Initial and boundary conditions are chosen so that the exact solution is given by

$$u(x, y; t) = \left(1 + e^{(x+y-t)/2\epsilon}\right)^{-1}.$$

It should be pointed out that (5.2) is just a special case of Burgers' equation. The standard Burgers' equations in 2D are a system of two PDEs for the velocity components derived from the Navier-Stokes equations. In our computation the viscosity coefficient is chosen as $\epsilon = 0.005$. In this problem, large gradient solutions are developed to the boundaries for $t > 0$. As a consequence, boundary point redistribution should be made in order to improve the quality of the adaptive mesh. A simple redistribution strategy is described as follows. The basic idea is to move the boundary points by solving 1-D moving mesh equations. Without lose of generality, we consider a simple boundary $[a, b]$ in the x-direction. Solving the two-point boundary value problem for $(\omega x_\xi)_\xi = 0, \omega(0) = a$ and $\omega(1) = b$, with uniform mesh in ξ will lead to a new boundary redistribution. Here the monitor function is restricted in the boundary $[a, b]$. This approach of the boundary point redistribution has been used in [3, 11, 17].

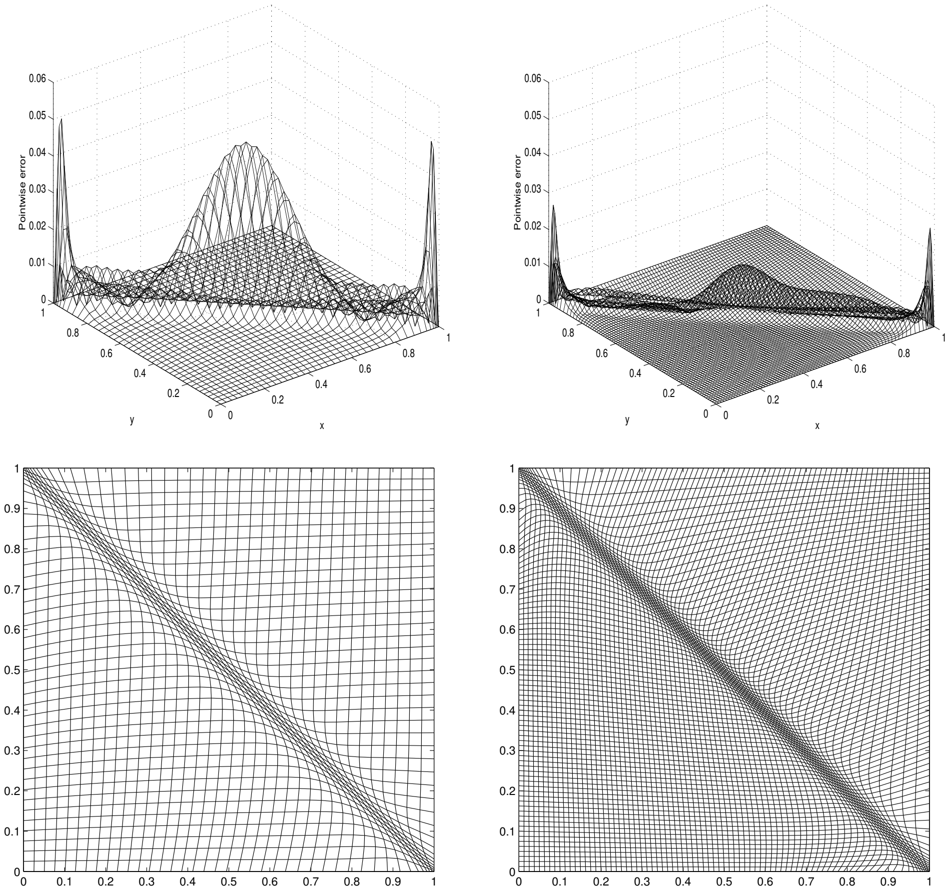

FIGURE 9. Example 5.1 at $t = 1$. Left: pointwise error and adaptive mesh with 40^2 grid points; right: those with 80^2.

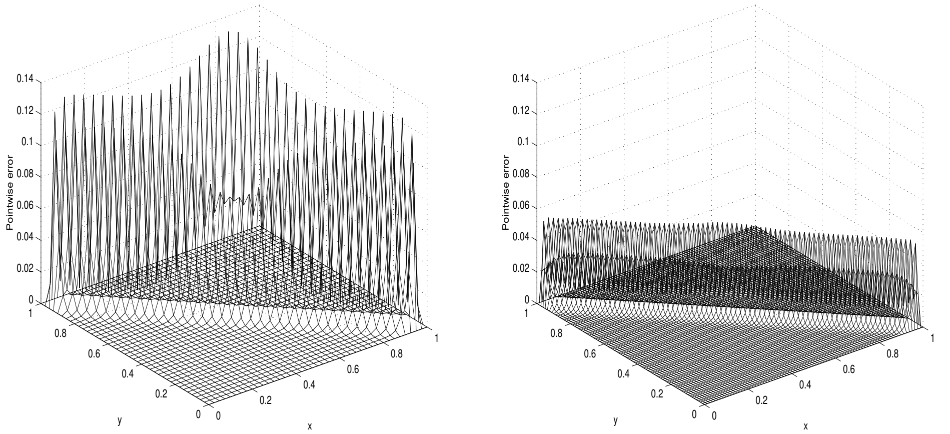

FIGURE 10. Example 5.1 at $t = 1$: pointwise errors for uniform mesh solution with 40^2 (left) and 80^2 (right) grid points.

Adaptive meshes and the corresponding solution errors at $t = 1$ are presented in Fig. 9. In these computations, 40 and 80 grid points are used, respectively. It is seen that quite large portion of the grid points have been moved to the regions with large solution

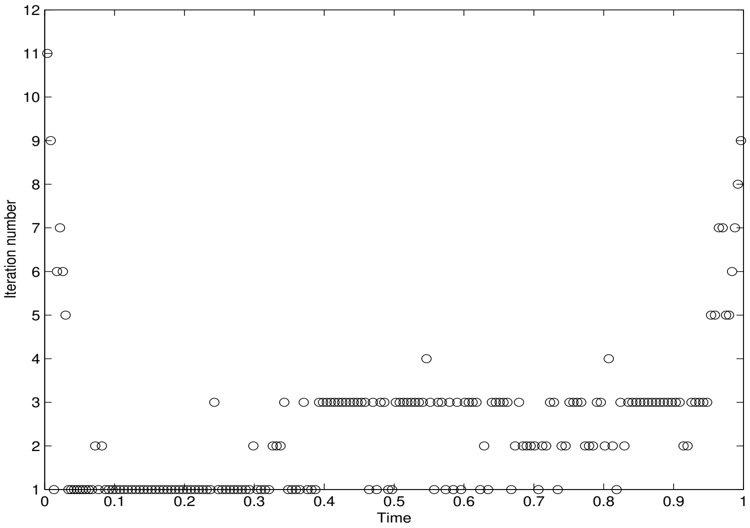

FIGURE 11. Example 5.1: the number of Gauss-Seidel cycles against time. 40^2 grid points are used.

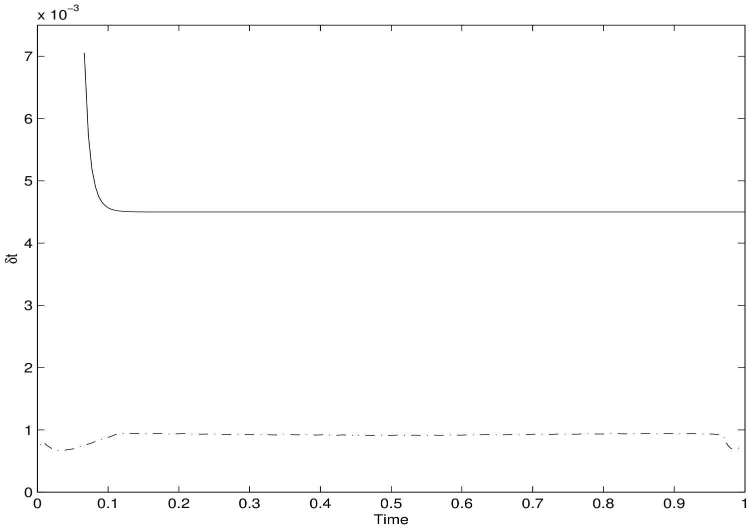

FIGURE 12. Example 5.1: time step against time. The solid line is the time step with the adaptive mesh and the broken line is that with the uniform mesh. 40^2 grid points are used.

variations. The CFL constant in (5.1) used is 0.18, and the monitor function used is $\sqrt{1+|\tilde{\nabla}u|^2}$, where $\tilde{\nabla} = (\partial_\xi, \partial_\eta)$. For comparison, Fig. 13 shows the solution errors with uniform meshes, showing clearly the advantage of moving mesh algorithm for the convection-dominated problem in 2D.

It is seen from Table 1 that at each time level, several cycles of the Gauss-Seidel iterations have to be performed. The iteration procedure on grid-motion and solution-interpolation is continued until there is no significant change in calculated new grids from one iteration to the next. It is essential that the (average) number of iteration cycle should not be large, otherwise the efficiency gained by the AMR methods will be seriously affected. In Fig. 11, the variation of the iteration numbers against time is plotted. It is seen that for this 2D problem the average number of iterations used at each time level is about 2.

It is observed that the AMR algorithm has to use smaller time steps than those for the uniform mesh computation. In Fig. 12m ttime steps used for both adaptive meshes and uniform mesh with 40^2 grid points are plotted. It is seen that the time step for the AMR algorithm is 5 times smaller, which is acceptable by considering significant improvement for the accuracy. In fact, to achieve the same L^1-error of the 40^2-adaptive results, more than 120^2-uniform mesh grids are required, which indicate about 10 times saving in the spatial grids. This is particularly useful when dealing with 3D computations – much smaller storage is required in using an AMR algorithm.

Example 5.2. (Two-Dimensional Buckley-Leverett Equation). Consider

$$u_t + f(u)_x + g(u)_y = \epsilon \Delta u \,, \tag{5.3}$$

with $\epsilon = 0.01$. The flux functions are of the form

$$f(u) = \frac{u^2}{u^2 + (1-u^2)},$$
$$g(u) = f(u)(1 - 5(1-u^2))$$

and the initial data is

$$u(x,y,0) = \begin{cases} 1 & x^2 + y^2 < 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Note that the above model includes gravitational effects in the y-direction. This example is taken from Kurganov and Tadmor [10], whose exact solution is unknown. Fig. 13 shows the numerical solutions with uniform and adaptive mesh solutions, computed in the square domain $[-1.5, 1.5]^2$ with 50^2 and 80^2 grid points. It is seen that the adaptive mesh solutions are more accurate than the uniform mesh ones. The effects of the mesh adaptation are also demonstrated in Fig. 14. For this example, the CFL constant in (5.1) used is 0.4, and the monitor function used is $\sqrt{1 + |\tilde{\nabla} u|^2}$, where $\tilde{\nabla} = (\partial_\xi, \partial_\eta)$.

6. CONCLUDING REMARKS

In this work, an 1D adaptive mesh redistribution methods, based on solving PDEs in the physical domain and obtaining the moving meshes in the computational domain, are developed. In the 1D approach, a first-order approximation is used to approximate the first order derivative associated with the small viscosity terms. In general, this is a first-order method, but the adaptation effects are clearly observed from the numerical experiments. However, this approach seems inappropriate when being extended to 2D problems, i.e. if a first-order approximation is used to handle the diffusion term (although the viscosity coefficient is very small) then the power of the adaptation will not be obvious. In the 2D case, the given PDEs are transformed into the computational domain, so that both the PDEs and the mesh generation equations are solved in the computational domain. A formal second-order accuracy for the AMR algorithms can be easily obtained, by using a mid-point integration rule and a central-differencing approach to handle the first-order

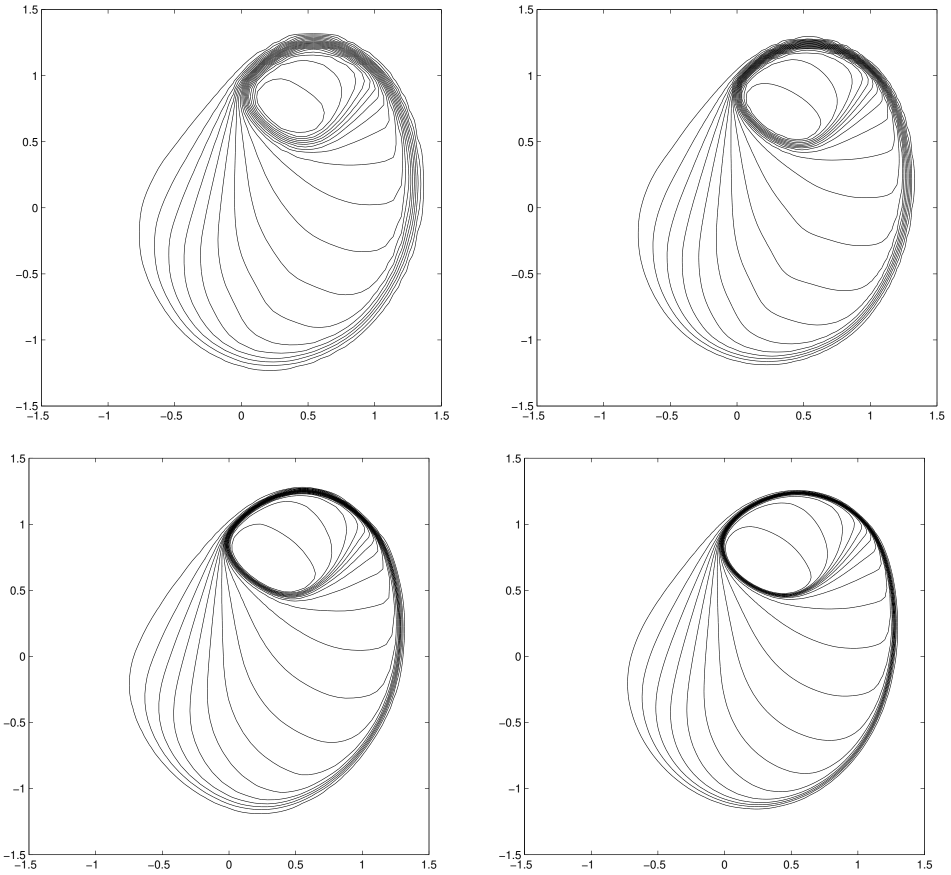

FIGURE 13. Example 5.2 at $t = 0.5$: uniform mesh solutions (top) and adaptive mesh solutions (bottom). 50^2 (left) and 80^2 (right) grid points.