Journal of Computational Physics 298 (2015) 661–677

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A moving mesh finite difference method for equilibrium radiation diffusion equations ☆

Xiaobo Yang [a], Weizhang Huang [b], Jianxian Qiu [c,*]

[a] Department of Mathematics, College of Science, China University of Mining and Technology, Xuzhou, Jiangsu 221116, China
[b] Department of Mathematics, University of Kansas, Lawrence, KS 66045, USA
[c] School of Mathematical Sciences and Fujian Provincial Key Laboratory of Mathematical Modeling and High-Performance Scientific Computing, Xiamen University, Xiamen, Fujian 361005, China

ARTICLE INFO

Article history:
Received 9 February 2015
Accepted 20 June 2015
Available online 29 June 2015

Keywords:
Moving mesh method
Equilibrium radiation diffusion equations
Prediction–correction
Freezing coefficient
Nonnegativity
Cutoff
Two-level mesh movement

ABSTRACT

An efficient moving mesh finite difference method is developed for the numerical solution of equilibrium radiation diffusion equations in two dimensions. The method is based on the moving mesh partial differential equation approach and moves the mesh continuously in time using a system of meshing partial differential equations. The mesh adaptation is controlled through a Hessian-based monitor function and the so-called equidistribution and alignment principles. Several challenging issues in the numerical solution are addressed. Particularly, the radiation diffusion coefficient depends on the energy density highly nonlinearly. This nonlinearity is treated using a predictor–corrector and lagged diffusion strategy. Moreover, the nonnegativity of the energy density is maintained using a cutoff method which has been known in literature to retain the accuracy and convergence order of finite difference approximation for parabolic equations. Numerical examples with multi-material, multiple spot concentration situations are presented. Numerical results show that the method works well for radiation diffusion equations and can produce numerical solutions of good accuracy. It is also shown that a two-level mesh movement strategy can significantly improve the efficiency of the computation.

1. Introduction

Radiation diffusion plays an important role in a variety of physical applications such as inertially confined fusion, combustion simulation, and atmospheric dynamics. When photon mean free paths are much shorter than characteristic length scales, a diffusion approximation can be used to describe the radiation penetrating from a hot source to a cold medium. This diffusion approximation forms a highly nonlinear diffusion coefficient and gives a sharp hot wave steep front (often referred to as a Marshak wave). Solutions near this steep front can vary dramatically in a very short distance. Such complex local solution structures make radiation diffusion an excellent example for using mesh adaptation methods because the number of mesh points can be prohibitively large when a uniform mesh is used.

☆ Research was supported in part by National Natural Science Foundation of China through grants 91230110, 11328104, 11426214, and 2014QNB35, and National Science Foundation (USA) through grant DMS-1115118.

* Corresponding author.
E-mail addresses: xwindyb@126.com (X. Yang), whuang@ku.edu (W. Huang), jxqiu@xmu.edu.cn (J. Qiu).

http://dx.doi.org/10.1016/j.jcp.2015.06.014
0021-9991/© 2015 Elsevier Inc. All rights reserved.

Generally speaking, there are two types of radiation diffusion equations, equilibrium and non-equilibrium systems. When the energy density E satisfies the relation $E = T^4$, where T is temperature, the system is called in an equilibrium state and otherwise in a non-equilibrium state. Radiation diffusion has attracted considerable attention from researchers in the past; e.g., see [3,16,21,23–25,28–30]. For example, foundations of radiation hydrodynamics can be found in the book [21] while numerical techniques for radiation diffusion and transport are addressed systematically in the book [3]. Rider et al. [25] study multi-material equilibrium radiation diffusion equations and propose a class of algorithms with the Newton–Krylov (GMERS) method preconditioned by a multi-grid method as a special example. Ovtchinnikov and Cai [23] study a parallel one-level Newton–Krylov–Schwarz algorithm for an unsteady nonlinear radiation diffusion problem.

On the other hand, there are only a few published works that have employed mesh adaptation for the numerical solution of radiation diffusion equations. Winkler et al. [26,27] use an adaptive moving mesh method to solve radiation diffusion and fluid equations in one dimension. Unfortunately, their method has difficulty with mesh crossing and cannot be extended in multi-dimensions. Lapenta and Chacón [16] study an equilibrium radiation diffusion equation in one dimension using an adaptive moving mesh method. Pernice and Philip [24] use the AMR method (a structured adaptive mesh refinement method) to solve two-dimensional equilibrium radiation diffusion equations. They employ a fully implicit scheme to integrate the partial differential equations (PDEs), JFNK (Jacobian-free Newton–Krylov) [14,15,22,25] to solve the resulting nonlinear algebraic equations, and the FAC (Fast Adaptive Composite) preconditioner [19,20] to precondition the implicit coefficient matrix. Their numerical results show that the method can capture the fronts of Marshak waves and have good accuracy for problems with smooth initial solutions.

The objective of this paper is to study the finite difference (FD) solution of two-dimensional equilibrium radiation diffusion equations using an adaptive moving mesh method. The method is based on the so-called moving mesh PDE (MMPDE) approach [10] with which the mesh is moved continuously in time using an MMPDE. The latter is defined as the gradient flow equation of a meshing functional formulated based on mesh equidistribution and alignment and taking into full consideration of the shape, size and orientation of mesh elements [7]. The method is combined with treatments of high nonlinearity and preservation of solution nonnegativity of the equations. The high nonlinearity comes from the diffusion coefficient. We use a coefficient-freezing predictor–corrector procedure to linearize the PDEs. More specifically, at the prediction stage the diffusion coefficient is calculated using the energy density at the previous time step while at the correction stage the coefficient is calculated using the energy density obtained at the prediction stage. This predictor–corrector procedure is known to be comparable to the Beam and Warming linearization method in terms of accuracy and stability [17]. It is also easy and efficient to implement since it contains only two steps of the lagged diffusion computation. Note that for each stage we only need to solve linear PDEs so there is no need for nonlinear iteration. Moreover, the procedure allows an easy and effective dealing of negative values occurring in the computed energy density. Radiation diffusion equations admit nonnegative energy densities. It is crucial for numerical approximation to preserve this property. Excessive negative values in the computed solution not only introduce unphysical oscillations but also can cause the computation to exit unexpectedly. We use a cutoff strategy with which negative values in the computed energy density are replaced with zero after each stage. It has been shown in [18] that the cutoff strategy retains the accuracy and convergence order of FD approximation for parabolic PDEs.

The moving mesh method, together with the above described treatments for nonlinearity and preservation of solution nonnegativity, is applied to a two-dimensional equilibrium radiation diffusion equation for two multi-material, multiple spot concentration scenarios. The numerical results show that the method is able to catch interfaces and onsets of new interfaces and concentrate mesh points near them. The results are comparable to those obtained by Pernice and Philip [24] using the AMR method and to those obtained with the uniform mesh of a much larger size. Moreover, it is shown that the computational efficiency can be significantly improved by a two-level mesh movement strategy [6] while maintaining a comparable level of accuracy.

An outline of the paper is as follows. The physical problem and the governing equations are described in Section 2. The moving mesh method and the treatments of nonlinearity and preservation of solution nonnegativity are discussed in Section 3. In Section 4 we present numerical results obtained for two multi-material, multiple spot concentration scenarios. Finally, Section 5 contains conclusions.

2. Problem description

Radiation diffusion occurs in a variety of astrophysical and laboratory settings and can be formulated in a number of forms; e.g., see Mihalas and Mihalas [21]. For a simple setting where the material temperature is in equilibrium with the radiation energy density, radiation diffusion can be modeled by a nonlinear parabolic PDE in dimensionless form as

$$\frac{\partial E}{\partial t} = \nabla \cdot (D_L(E)\nabla E), \tag{2.1}$$

where E is the dimensionless gray radiation energy density and $D_L(E)$ is Larsen's form of the flux-limited diffusion coefficient [1,13] defined as

$$D_L(E) = \left(\frac{1}{D(E)^2} + \frac{|\nabla E|^2}{E^2} \right)^{-\frac{1}{2}}. \tag{2.2}$$

Here, $D(E) = Z^{-3}E^{\frac{3}{4}}$ and Z is the atomic number of the medium which can be a discontinuous function of spatial coordinates when multi-materials co-exist in the physical domain. Two choices of Z are used in our computation; see Fig. 4.1.

We consider (2.1) in two dimensions on the unit square domain $\Omega = (0, 1) \times (0, 1)$. Neumann boundary conditions are used for boundary segments $y = 0$ and $y = 1$ and inflow and outflow boundary conditions are employed on $x = 0$ and $x = 1$, respectively. Specifically, the boundary conditions are

$$\begin{cases} \frac{\partial E}{\partial y} = 0, & \text{on } y = 0 \text{ or } y = 1 \\ \frac{1}{4}E - \frac{1}{2}D_L(E)\frac{\partial E}{\partial x} = E_{\text{in}}, & \text{on } x = 0 \\ \frac{1}{4}E + \frac{1}{2}D_L(E)\frac{\partial E}{\partial x} = E_{\text{out}}, & \text{on } x = 1 \end{cases} \tag{2.3}$$

where $E_{\text{in}} = 2500$ and $E_{\text{out}} = 0.25$ are taken in our computation. The initial condition is

$$E(x, y, 0) = (1 - \tanh(10x))(10^4 - 1) + 1, \quad (x, y) \in \Omega. \tag{2.4}$$

This initial solution is essentially equal to one except that it has a thin layer near the boundary $x = 0$. This transition is used to make the initial solution to be consistent with the boundary condition (2.3), avoiding a potentially difficult initial start in time integration.

3. Moving FD method

In this section we describe an adaptive moving mesh FD method for solving the initial–boundary value problem (IBVP) (2.1), (2.3), and (2.4). The problem is discretized in space using central finite differences and in time using a Singly Diagonally Implicit Runge–Kutta scheme (SDIRK) [2]. Issues for adaptive mesh movement, linearization of (2.1), and preservation of solution nonnegativity are also discussed.

3.1. FD discretization on moving meshes

We assume for the moment that a curvilinear moving mesh

$$(x_{j,k}(t), y_{j,k}(t)), \quad j = 1, \ldots, J, \; k = 1, \ldots, K \tag{3.1}$$

is given for Ω, where J and K are positive integers. The generation of adaptive moving meshes will be discussed in Section 3.3.

A common practice for constructing an FD discretization of PDEs on a moving mesh is to first view the mesh as an image of a fixed, rectangular mesh in the reference domain under a coordinate transformation, then transform the PDEs into the reference domain, and finally discretize the transformed PDEs on the fixed reference mesh. For the current situation, we take the reference domain to be the unit square and consider (3.1) as the image of a fixed rectangular mesh under a coordinate transformation $x = x(\xi, \eta, t)$, $y = y(\xi, \eta, t)$, i.e.,

$$x_{j,k}(t) = x(\xi_j, \eta_k, t), \quad y_{j,k}(t) = y(\xi_j, \eta_k, t), \quad j = 1, \ldots, J, \quad k = 1, \ldots, K \tag{3.2}$$

where the reference mesh is taken as

$$(\xi_j, \eta_k) = ((j-1)\Delta\xi, (k-1)\Delta\eta), \quad j = 1, \ldots, J, \quad k = 1, \ldots, K, \tag{3.3}$$

and $\Delta\xi = 1/(J - 1)$ and $\Delta\eta = 1/(K - 1)$. Notice that the boundary correspondence between the reference and physical domains is given by

$$x(0, \eta) = 0, \quad x(1, \eta) = 1, \quad y(\xi, 0) = 0, \quad y(\xi, 1) = 1. \tag{3.4}$$

Denote by

$$\hat{E}(\xi, \eta, t) = E(x(\xi, \eta, t), y(\xi, \eta, t), t).$$

Using the coordinate transformation, PDE (2.1) can be transformed (e.g., see [10, §3.1.4]) into the reference domain as

$$\hat{E}_t - \boldsymbol{b}(t) \cdot \hat{\nabla}\hat{E} = \frac{1}{J(t)}\hat{\nabla} \cdot \left(D_L(\hat{E})A(t)\hat{\nabla}\hat{E} \right), \tag{3.5}$$

where

$$J(t) = x_\xi y_\eta - x_\eta y_\xi, \quad \hat{\nabla} = \begin{bmatrix} \frac{\partial}{\partial\xi} \\ \frac{\partial}{\partial\eta} \end{bmatrix},$$

$$\boldsymbol{b}(t) = \frac{1}{J(t)} \begin{bmatrix} y_\eta x_t - x_\eta y_t \\ -y_\xi x_t + x_\xi y_t \end{bmatrix}, \quad A(t) = \frac{1}{J(t)} \begin{bmatrix} x_\eta^2 + y_\eta^2 & -(x_\xi x_\eta + y_\xi y_\eta) \\ -(x_\xi x_\eta + y_\xi y_\eta) & x_\xi^2 + y_\xi^2 \end{bmatrix}.$$

Similarly, the boundary condition (2.3) can be transformed into

$$\begin{cases} \frac{-x_\eta}{x_\xi y_\eta}\frac{\partial \hat{E}}{\partial \xi} + \frac{1}{y_\eta}\frac{\partial \hat{E}}{\partial \eta} = 0, & \text{on } \eta = 0 \text{ or } \eta = 1 \\ \frac{1}{4}\hat{E} - \frac{1}{2}D_L(\hat{E})\left(\frac{1}{x_\xi}\frac{\partial \hat{E}}{\partial \xi} + \frac{-y_\xi}{x_\xi y_\eta}\frac{\partial \hat{E}}{\partial \eta}\right) = E_{\text{in}}, & \text{on } \xi = 0 \\ \frac{1}{4}\hat{E} + \frac{1}{2}D_L(\hat{E})\left(\frac{1}{x_\xi}\frac{\partial \hat{E}}{\partial \xi} + \frac{-y_\xi}{x_\xi y_\eta}\frac{\partial \hat{E}}{\partial \eta}\right) = E_{\text{out}}, & \text{on } \xi = 1 \end{cases} \tag{3.6}$$

where we have used the fact that $x_\eta = 0$ on $\xi = 0$ and $\xi = 1$ and $y_\xi = 0$ on $\eta = 0$ and $\eta = 1$.

The discretization of (3.5) and (3.6) on rectangular mesh (3.3) using central finite differences is straightforward. To save space, we omit the detail of the derivation and formulation of the FD approximation here and refer the interested reader to [10, §3.2]. The FD approximation of (3.5) can be expressed as

$$\frac{dE_h}{dt} - \boldsymbol{b}_h(t) \cdot \hat{\nabla}_h E_h = \frac{1}{J_h(t)}\hat{\nabla}_h \cdot \left(D_L(E_h) A_h(t) \hat{\nabla}_h E_h \right), \tag{3.7}$$

where the FD approximations of quantities or gradient operator are denoted by the same symbols with subscript h.

3.2. Temporal discretization, linearization, and nonnegativity preservation

From (3.7) (and the FD approximation of the boundary condition (3.6)), we can see that the IBVP is nonlinear since the diffusion coefficient depends on the energy density. Time integration of such nonlinear diffusion equations has been studied extensively in the past; e.g. see [12,13,17,23–25]. Essentially, two types of method have been considered, fully implicit schemes and schemes based on linearization. For fully implicit methods, (3.7) is discretized with an implicit scheme and the resulting nonlinear algebraic equations are solved using either the Newton iteration together with a linear system solver or a Newton–Krylov-type solver. They provide an excellent stability and allow large time steps. However, they are also more expensive and have difficulty in handling the preservation of solution nonnegativity (see the discussion below).

On the other hand, linearization-based methods use some type of linearization. Commonly used strategies include lagged diffusion (where the diffusion coefficient is calculated with the energy density at the previous time step), the Beam and Warming linearization or linearly implicit approximation (where the diffusion coefficient is expanded up to the linear term about the energy density at the previous time step), and the predictor–corrector strategy (to be described below). These methods have been found to perform surprisingly well when compared to fully implicit methods although they may produce unphysical solutions for very large time steps [17]. We employ the predictor–corrector strategy in our computation since it is comparable to the Beam and Warming linearization in terms of accuracy and stability and to the lagged diffusion in terms of efficiency and ease in implementation. With the strategy, the linearized equation of (3.7) reads as

$$\begin{cases} \frac{dE_h}{dt} - \boldsymbol{b}_h(t) \cdot \hat{\nabla}_h E_h = \frac{1}{J_h(t)}\hat{\nabla}_h \cdot \left(D_L(E_h^*) A_h(t) \hat{\nabla}_h E_h \right), & t_n < t \le t_{n+1} \\ E_h(t_n) = E_h^n, \end{cases} \tag{3.8}$$

where E_h^n is the approximation of the energy density at $t = t_n$. During the prediction stage, E_h^* is taken as the energy density at $t = t_n$, i.e., $E_h^* = E_h^n$. This stage is the same as the lagged diffusion method. The solution obtained in this stage at $t = t_{n+1}$ is used as E_h^* during the correction stage. In both stages, the linear equation (3.8) is integrated with a two-stage SDIRK scheme (Cash [2]). The resulting linear systems are solved by the unsymmetric multifrontal sparse LU factorization package UMFPACK [4].

It is pointed out that the solution of IBVP (2.1), (2.3), and (2.4) stays nonnegative for all time. However, an FD approximation may not preserve this property. Excessive negative values in the solution approximation not only introduce nonphysical oscillations but also can cause problems such as not-a-number (NaN), divergence of nonlinear and/or linear iterations, too small time steps, and early stopping of computation [28]. To avoid these difficulties, we employ a cutoff method which replaces negative values with zero after the prediction and correction stages. Lu et al. [18] show that the cutoff procedure retains the convergence order and stability of the underlying FD scheme for linear parabolic PDEs. Their numerical results also show that the procedure works for nonlinear parabolic PDEs too.

3.3. Generation of adaptive moving meshes

We use the MMPDE approach [8–10] of moving mesh methods for the generation of adaptive moving meshes. The basic idea of the approach is to use a mesh PDE to move the mesh continuously and adaptively according to some feature of the error in approximating the physical solution. The mesh PDE, or called the MMPDE, is defined as the gradient flow equation of a functional. The latter is formulated in terms of the inverse coordinate transformation $\xi = \xi(x, y, t)$ and $\eta = \eta(x, y, t)$ to avoid possible mesh singularity (e.g., see Dvinsky [5]) and based on mesh equidistribution and alignment. The approach employs a monitor function – a symmetric and uniformly positive definite matrix valued function – to control mesh equidistribution and alignment. The reader is referred to [10,11] for choices of the monitor function based on interpolation error estimates.

Denote the Hessian of the energy density by

$$H=\begin{bmatrix} E_{xx} & E_{xy} \\ E_{xy} & E_{yy} \end{bmatrix}.$$

Let $|H|=\sqrt{H^2}$. We choose the monitor function as

$$M=\det(\alpha I+|H|)^{-\frac{1}{4}}\left[\alpha I+|H|\right], \tag{3.9}$$

which is known to be optimal for the H^1 norm of the error of linear interpolation [11]. Here, $\alpha>0$ is the regularity parameter defined through the equation

$$\int_\Omega \det(M(\alpha))^{\frac{1}{2}}dxdy=2\int_\Omega \det(M(0))^{\frac{1}{2}}dxdy,$$

where $M(0)$ denotes the monitor function (3.9) with $\alpha=0$. The above equation can be rewritten as

$$\int_\Omega \det(\alpha I+|H|)^{\frac{1}{2}}dxdy=2\int_\Omega \det(|H|)^{\frac{1}{2}}dxdy.$$

The left-hand side of the above equation is an increasing function of α, and thus the equation has a unique solution provided that the right-hand side does not vanish. It can be solved easily for α using the bisection method.

For the meshing functional, we use

$$\begin{aligned} I[\xi,\eta]=\theta\int_\Omega \det(M)^{\frac{1}{2}}\left(\nabla\xi^T M^{-1}\nabla\xi+\nabla\eta^T M^{-1}\nabla\eta\right)^2 dxdy \\ +(1-2\theta)2^2\int_\Omega \frac{\det(M)^{\frac{1}{2}}}{\left(J\det(M)^{\frac{1}{2}}\right)^2}dxdy, \end{aligned} \tag{3.10}$$

where $J=x_\xi y_\eta-x_\eta y_\xi=1/(\xi_x\eta_y-\xi_y\eta_x)$ is the Jacobian of the coordinate transformation and θ is a dimensionless parameter balancing the first and second terms. (We use $\theta=0.1$ in our computation.) This functional was proposed in [7] to control mesh equidistribution and alignment. Indeed, it is shown in [7] that the minimization of the second term leads to the equidistribution principle

$$J\det(M)^{\frac{1}{2}}=\sigma, \qquad (x,y)\in\Omega$$

where $\sigma=\int_\Omega \det(M)^{\frac{1}{2}}dxdy$. It implies that the mesh is denser in regions where $\det(M)^{\frac{1}{2}}$ is larger. On the other hand, the minimization of the first term will force the mesh to be aligned with M in the sense that the mesh elements tend to be as equilateral as possible when measured in the metric specified by M.

The MMPDE is defined as the gradient flow equation of the meshing functional, i.e.,

$$\xi_t=-\frac{1}{\tau}\frac{\delta I}{\delta\xi}, \quad \eta_t=-\frac{1}{\tau}\frac{\delta I}{\delta\eta}, \tag{3.11}$$

where τ is a parameter used to control the response of the mesh movement to the change in the monitor function (and therefore the change in the physical solution) and $\delta I/\delta\xi$ and $\delta I/\delta\eta$ are the functional derivatives of $I[\xi,\eta]$. It is easy to find that

$$\frac{\delta I}{\delta\xi}=-4\theta\nabla\cdot\left(\det(M)^{\frac{1}{2}}\beta M^{-1}\nabla\xi\right)-8(1-2\theta)\nabla\cdot\left(\frac{1}{J\det(M)^{\frac{1}{2}}}\begin{bmatrix}\eta_y\\ -\eta_x\end{bmatrix}\right), \tag{3.12}$$

$$\frac{\delta I}{\delta\eta}=-4\theta\nabla\cdot\left(\det(M)^{\frac{1}{2}}\beta M^{-1}\nabla\eta\right)-8(1-2\theta)\nabla\cdot\left(\frac{1}{J\det(M)^{\frac{1}{2}}}\begin{bmatrix}-\xi_y\\ \xi_x\end{bmatrix}\right), \tag{3.13}$$

where

$$\beta=\nabla\xi^T M^{-1}\nabla\xi+\nabla\eta^T M^{-1}\nabla\eta. \tag{3.14}$$

Moreover, by differentiating the identities

$$\xi=\xi(x(\xi,\eta,t),y(\xi,\eta,t),t), \quad \eta=\eta(x(\xi,\eta,t),y(\xi,\eta,t),t)$$

with respect to t while fixing ξ and η, we get

$$0=\xi_t+\xi_x x_t+\xi_y y_t, \quad 0=\eta_t+\eta_x x_t+\eta_y y_t,$$

or

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = -\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix}^{-1} \begin{bmatrix} \xi_t \\ \eta_t \end{bmatrix} = -\begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \begin{bmatrix} \xi_t \\ \eta_t \end{bmatrix}. \tag{3.15}$$

Combining (3.11) with (3.12), (3.13), and (3.15), we get

$$\begin{aligned} \begin{bmatrix} x_t \\ y_t \end{bmatrix} &= \frac{1}{\tau} \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} 4\theta \nabla \cdot \left(\det(M)^{\frac{1}{2}} \beta M^{-1} \nabla \xi \right) + 8(1-2\theta) \nabla \cdot \left(\frac{1}{J \det(M)^{\frac{1}{2}}} \begin{bmatrix} \eta_y \\ -\eta_x \end{bmatrix} \right) \\ 4\theta \nabla \cdot \left(\det(M)^{\frac{1}{2}} \beta M^{-1} \nabla \eta \right) + 8(1-2\theta) \nabla \cdot \left(\frac{1}{J \det(M)^{\frac{1}{2}}} \begin{bmatrix} -\xi_y \\ \xi_x \end{bmatrix} \right) \end{bmatrix}. \end{aligned} \tag{3.16}$$

By interchanging the roles of independent and dependent variables and after some lengthy but straightforward derivations (e.g., see [10, Chapter 6]), we can rewrite the above equation into

$$\frac{\partial}{\partial t} \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{\tau} \left(A_{11} \frac{\partial^2}{\partial \xi^2} + (A_{12} + A_{21}) \frac{\partial^2}{\partial \xi \partial \eta} + A_{22} \frac{\partial^2}{\partial \eta^2} + I\, b_1 \frac{\partial}{\partial \xi} + I\, b_2 \frac{\partial}{\partial \eta} \right) \begin{bmatrix} x \\ y \end{bmatrix}, \tag{3.17}$$

where

$$\begin{aligned} A_{ij} &= 8\theta \det(M)^{\frac{1}{2}} \widetilde{\boldsymbol{a}}^i (W \widetilde{\boldsymbol{a}}^j)^T + 4\theta \det(M)^{\frac{1}{2}} \beta (\boldsymbol{a}^i)^T \widetilde{\boldsymbol{a}}^j\, I + \frac{8(1-2\theta)}{J^2 \det(M)^{\frac{1}{2}}} \boldsymbol{a}_i (\boldsymbol{a}^j)^T, \quad i, j = 1, 2 \\ b_1 &= -(\boldsymbol{a}^1)^T \left[4\theta \det(M)^{\frac{1}{2}} (d_1 \widetilde{\boldsymbol{a}}^1 + d_2 \widetilde{\boldsymbol{a}}^2) + 4\theta\beta \left(\frac{\partial \det(M)^{\frac{1}{2}}}{\partial \xi} \widetilde{\boldsymbol{a}}^1 + \frac{\partial \det(M)^{\frac{1}{2}}}{\partial \eta} \widetilde{\boldsymbol{a}}^2 \right) \right. \\ &\qquad \left. + 4\theta \det(M)^{\frac{1}{2}} \beta \left(\frac{\partial M^{-1}}{\partial \xi} \boldsymbol{a}^1 + \frac{\partial M^{-1}}{\partial \eta} \boldsymbol{a}^2 \right) \right] + \frac{8(1-2\theta)}{(J \det(M)^{\frac{1}{2}})^2} \frac{\partial \det(M)^{\frac{1}{2}}}{\partial \xi}, \\ b_2 &= -(\boldsymbol{a}^2)^T \left[4\theta \det(M)^{\frac{1}{2}} (d_1 \widetilde{\boldsymbol{a}}^1 + d_2 \widetilde{\boldsymbol{a}}^2) + 4\theta\beta \left(\frac{\partial \det(M)^{\frac{1}{2}}}{\partial \xi} \widetilde{\boldsymbol{a}}^1 + \frac{\partial \det(M)^{\frac{1}{2}}}{\partial \eta} \widetilde{\boldsymbol{a}}^2 \right) \right. \\ &\qquad \left. + 4\theta \det(M)^{\frac{1}{2}} \beta \left(\frac{\partial M^{-1}}{\partial \xi} \boldsymbol{a}^1 + \frac{\partial M^{-1}}{\partial \eta} \boldsymbol{a}^2 \right) \right] + \frac{8(1-2\theta)}{(J \det(M)^{\frac{1}{2}})^2} \frac{\partial \det(M)^{\frac{1}{2}}}{\partial \eta}, \end{aligned}$$

and

$$\begin{aligned} &\boldsymbol{a}_1 = \begin{bmatrix} x_\xi \\ y_\xi \end{bmatrix}, \quad \boldsymbol{a}_2 = \begin{bmatrix} x_\eta \\ y_\eta \end{bmatrix}, \quad \boldsymbol{a}^1 = \begin{bmatrix} \xi_x \\ \xi_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta \\ -x_\eta \end{bmatrix}, \quad \boldsymbol{a}^2 = \begin{bmatrix} \eta_x \\ \eta_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} -y_\xi \\ x_\xi \end{bmatrix}, \\ &\widetilde{\boldsymbol{a}}^1 = M^{-1} \boldsymbol{a}^1, \quad \widetilde{\boldsymbol{a}}^2 = M^{-1} \boldsymbol{a}^2, \quad W = \boldsymbol{a}^1 (\boldsymbol{a}^1)^T + \boldsymbol{a}^2 (\boldsymbol{a}^2)^T, \\ &d_1 = (\boldsymbol{a}^1)^T \frac{\partial M^{-1}}{\partial \xi} \boldsymbol{a}^1 + (\boldsymbol{a}^2)^T \frac{\partial M^{-1}}{\partial \xi} \boldsymbol{a}^2, \quad d_2 = (\boldsymbol{a}^1)^T \frac{\partial M^{-1}}{\partial \eta} \boldsymbol{a}^1 + (\boldsymbol{a}^2)^T \frac{\partial M^{-1}}{\partial \eta} \boldsymbol{a}^2. \end{aligned}$$

The moving mesh equation (3.17) is supplemented with the one dimensional version of the MMPDE for the adaptation of boundary mesh points (cf. [7]). They are discretized in space using central finite differences and in time by the backward Euler method with coefficients A_{ij} and b_i calculated at the previous time step. The resulting algebraic systems are solved with the sparse matrix solver UMFPACK.

3.4. Solution procedure

We now are ready to discuss the solution procedure. We solve the mesh and physical PDEs in an alternating way for the mesh and the physical. Assuming that physical solution E^n, mesh (x^n, y^n), and time step size Δt_n are given at $t = t_n$, we elaborate the meshing and physical PDE solving steps in the following.

The meshing step. The monitor function (3.9) is computed using the current E^n and (x^n, y^n) and smoothed using several sweeps of a low-pass filter. The Hessian of the energy density used in (3.9) is replaced by an approximation obtained using least squares fitting. More specifically, at any mesh point a local quadratic polynomial is constructed by least squares fitting of the nodal values of E^n at neighboring mesh points. The approximate Hessian at the given mesh point is then obtained by differentiating the quadratic polynomial twice. After the monitor function has been obtained, the mesh equation (3.17) is integrated from t_n to $t_{n+1} = t_n + \Delta t_n$ for the new mesh (x^{n+1}, y^{n+1}).

The physical PDE solving step. The physical PDE (3.7) is integrated from t_n to t_{n+1} using the predictor–corrector scheme (3.8). During the integration, the mesh is considered to move linearly in time, viz.,

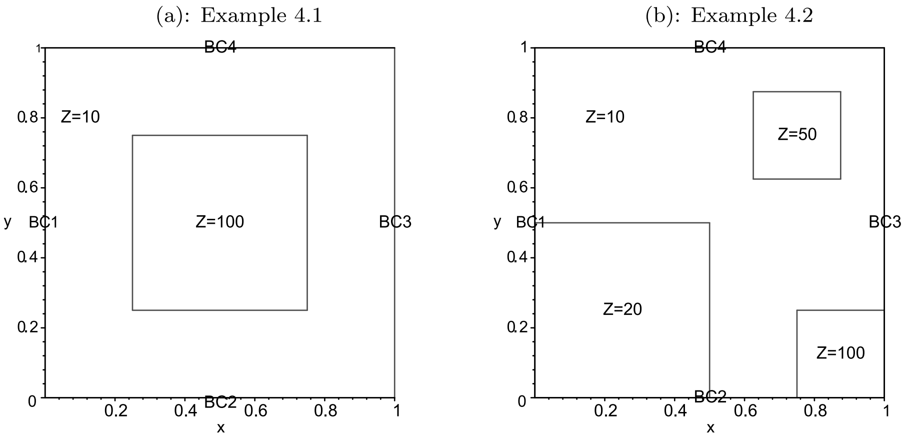


Fig. 4.1. Material configuration for the numerical examples.

$$x(t) = \frac{t - t_n}{\Delta t_n} x^{n+1} + \frac{t_n + \Delta t_n - t}{\Delta t_n} x^n, \qquad y(t) = \frac{t - t_n}{\Delta t_n} y^{n+1} + \frac{t_n + \Delta t_n - t}{\Delta t_n} y^n.$$

4. Numerical tests

We now present numerical results obtained with the moving mesh FD method described in the previous section for two multi-material scenarios of radiation diffusion. The material configuration is given in Fig. 4.1 for the examples. In both cases, the computational domain is taken as $\Omega = (0,1) \times (0,1)$ and the boundary condition (2.3) is used for the boundary segments BC1, BC2, BC3 and BC4.

Example 4.1. We first consider a situation where the atomic number is defined as (cf. Fig. 4.1(a))

$$Z(x,y) = \begin{cases} 100, & \text{for } (x,y) \in (\frac{1}{4}, \frac{3}{4}) \times (\frac{1}{4}, \frac{3}{4}) \\ 10, & \text{otherwise.} \end{cases} \tag{4.1}$$

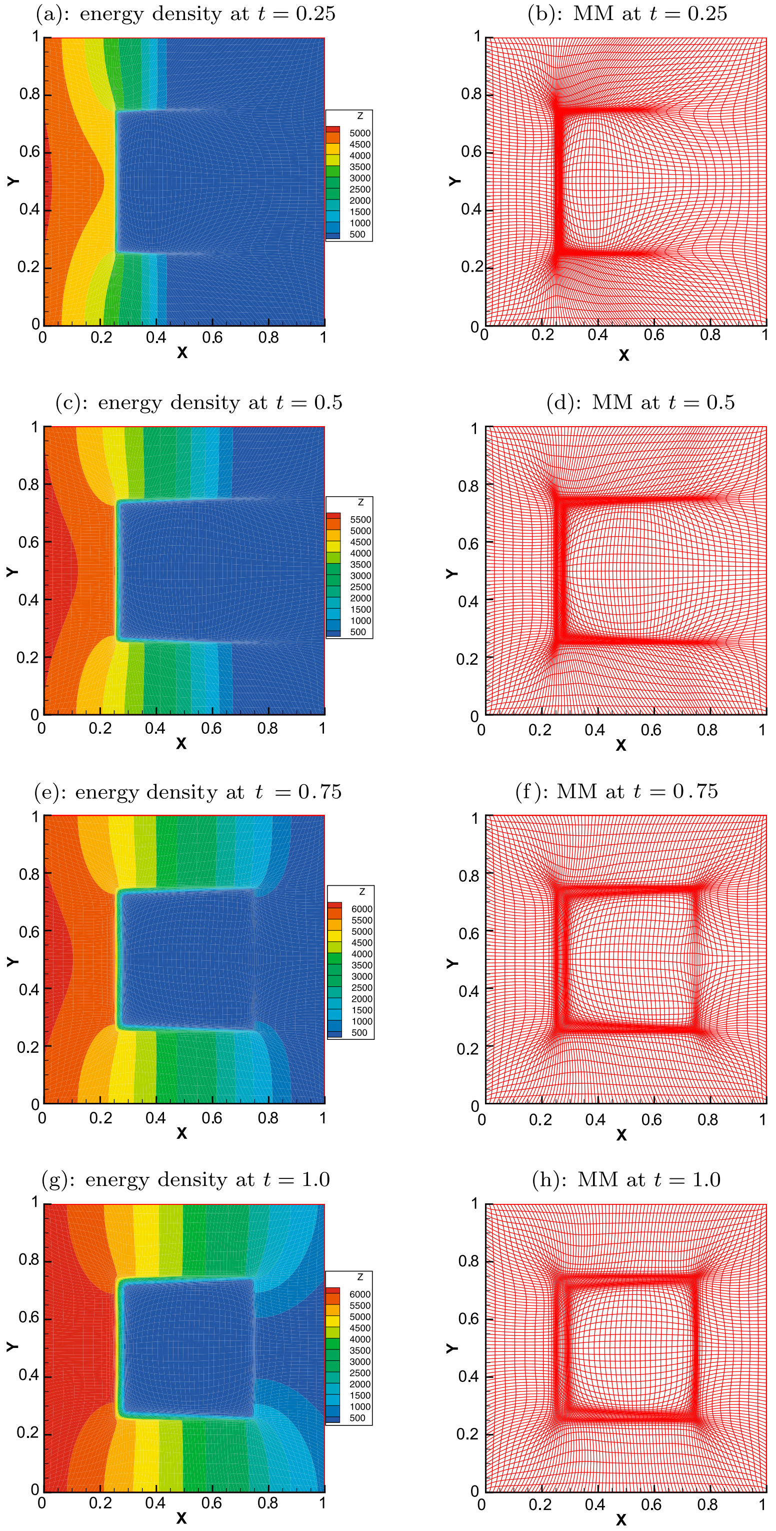
This example has been studied by Pernice and Philip [24]. The radiation diffusion equation is integrated up to $T = 1.0$ with a fixed time step $\Delta t = 10^{-3}$.

A typical adaptive moving mesh of size 81 × 81 and the contours of the corresponding energy density are plotted in Fig. 4.2. It can be seen that the moving mesh method captures precisely the interface between the hot diffusion wave and the cold medium and the mesh points are concentrated in the regions of the interface. Moreover, the solution obtained here is comparable to those in [24, Fig. 6.4] where the AMR method is used.

The CPU time is recorded in Table 4.1 for computation with one-level moving, two-level moving, and fixed meshes. For two-level moving mesh computation, a coarse mesh is computed by the MMPDE (3.17) and the radiation diffusion equation is solved on a fine mesh obtained by uniformly interpolating the coarse one. From the table, one can see that the one-level moving mesh method is more costly than the computation with a fixed mesh of the same size. This is expected since the moving mesh method solves more equations and its main advantage is that the adaptive mesh of a much smaller size can be used to achieve the same level of accuracy as a fixed mesh of large size. For instance, Fig. 4.3 shows that the contours of the energy density obtained with an adaptive moving mesh of size 81 × 81 are comparable to those obtained with a fixed mesh of size 512 × 512. Moreover, the efficiency can be improved using the two-level moving mesh strategy [6]. From the table we can see that for the case with mesh 81 × 81, the CPU time of the two-level moving mesh method is about 6% of that with the one-level moving mesh method. For the case of mesh 121 × 121, the two-level moving mesh is only about 0.5% of the cost of the one-level moving mesh method. Furthermore, we compare the two-level moving mesh method with the uniform mesh method. From Table 4.1, we see that with the increasing mesh size, the CPU difference between the two is getting smaller and smaller. Particularly, when the mesh size is 121 × 121, the CPU cost of two-level moving mesh is only about 28% more than that of the uniform mesh. In this sense, the two-level moving mesh method is competitive with the uniform mesh method even in CPU time.

To see the accuracy of the two-level moving mesh method, we plot the mesh and energy density in Figs. 4.4 and 4.5 for the one-level and two-level moving meshes of size 121 × 121. For the two-level computation, a coarse mesh of size 41 × 41 is moved by the moving mesh equation. From Fig. 4.5, one can see that the one-level method leads to an adaptive mesh with higher mesh concentration in a narrower region of the interface. Nevertheless, the results obtained with the one-level and two-level meshes seem very comparable (Fig. 4.4).

Example 4.2. The atomic number in the second example is defined as (cf. Fig. 4.1(b))


(a): energy density at $t = 0.25$

(b): MM at $t = 0.25$

(c): energy density at $t = 0.5$

(d): MM at $t = 0.5$

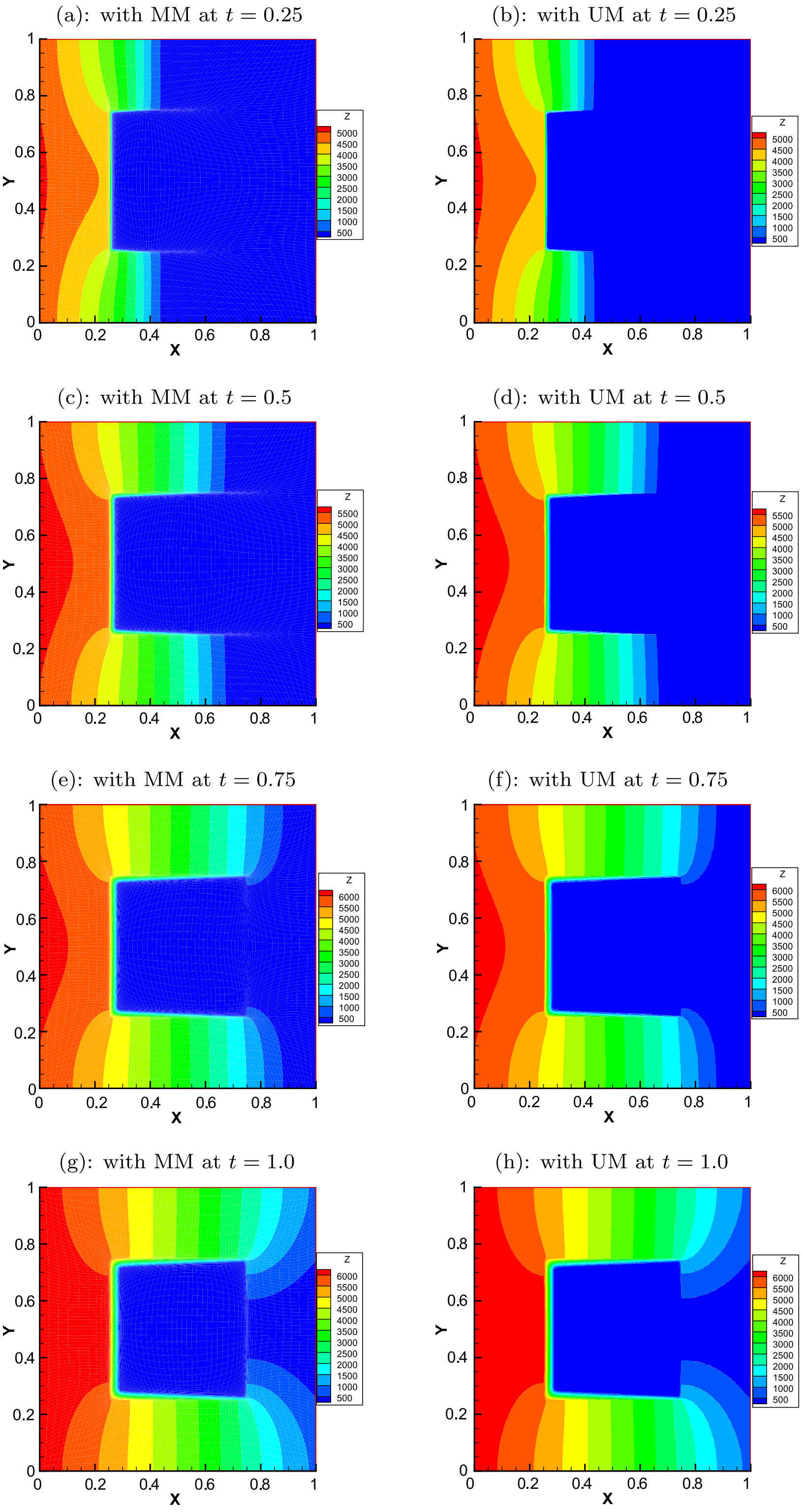
(e): energy density at $t = 0.75$

(f): MM at $t = 0.75$

(g): energy density at $t = 1.0$

(h): MM at $t = 1.0$

Fig. 4.2. Example 4.1. An adaptive moving mesh (MM) of size 81 × 81 and the contours of the computed energy density obtained thereon are plotted at $t = 0.25$, 0.5, 0.75, and 1.0.



(a): with MM at $t = 0.25$

(b): with UM at $t = 0.25$

(c): with MM at $t = 0.5$

(d): with UM at $t = 0.5$

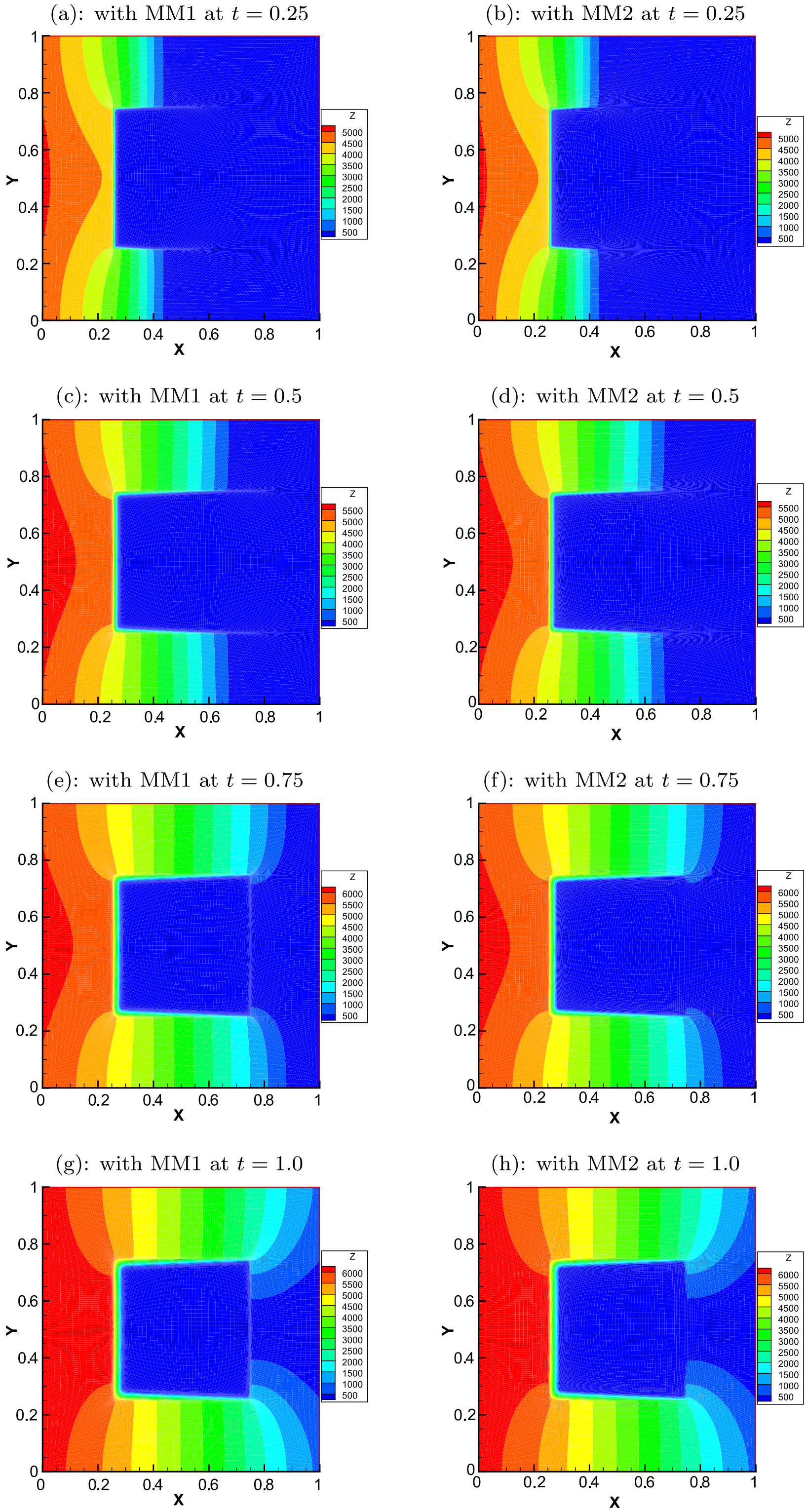
(e): with MM at $t = 0.75$

(f): with UM at $t = 0.75$

(g): with MM at $t = 1.0$

(h): with UM at $t = 1.0$

Fig. 4.3. Example 4.1. The contours of the energy density obtained with an adaptive moving mesh (MM) of size 81 × 81 are compared with those obtained with a uniform mesh (UM) of size 512 × 512.



(a): with MM1 at $t = 0.25$

(b): with MM2 at $t = 0.25$

(c): with MM1 at $t = 0.5$

(d): with MM2 at $t = 0.5$

(e): with MM1 at $t = 0.75$

(f): with MM2 at $t = 0.75$

(g): with MM1 at $t = 1.0$

(h): with MM2 at $t = 1.0$

Fig. 4.4. Example 4.1. The contours of the energy density obtained with a one-level adaptive moving mesh (MM1) of size 121 × 121 are compared with those obtained with a two-level adaptive moving mesh (MM2) of size 41 × 41 (with the physical PDE being solved on a mesh of size 121 × 121).

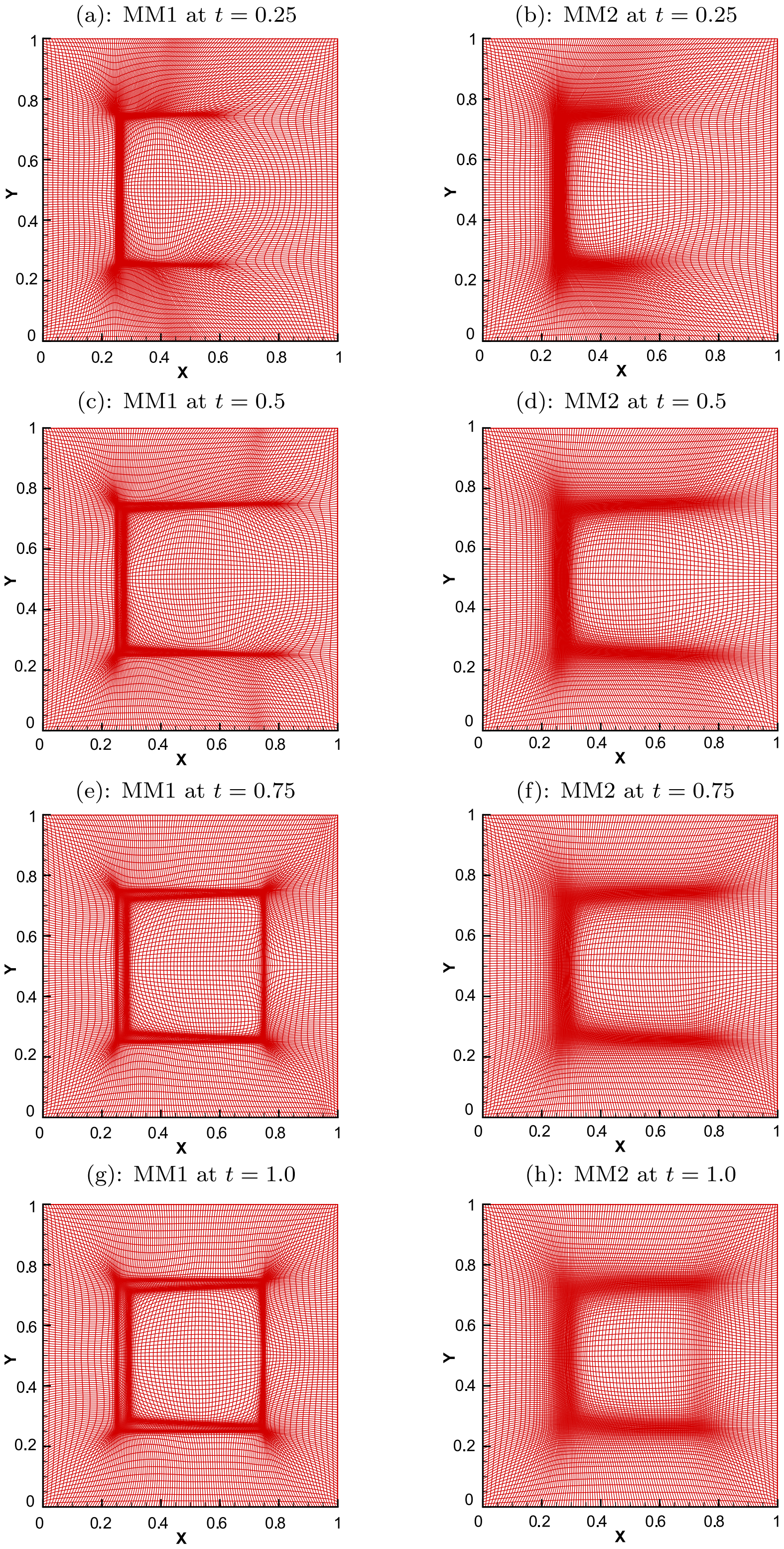


Fig. 4.5. Example 4.1. Adaptive moving meshes of size 121 × 121 obtained with one-level (MM1) and two-level (MM2) moving mesh strategies. MM2 is obtained by uniformly interpolating a 41 × 41 moving mesh.

Table 4.1
CPU time comparison among one-level, two-level moving mesh and uniform mesh methods for Example 4.1. The CPU time is measured in seconds. The last column is the ratio of the used CPU time to that used with a uniform mesh of the same size.

	Fine mesh	Coarse mesh	Total CPU time	Ratio
One-level MM	41 × 41	41 × 41	1545	2.8
	81 × 81	81 × 81	52 673	22.7
	121 × 121	121 × 121	1 467 826	254.3
Two-level MM	41 × 41	41 × 41	1545	2.8
	81 × 81	41 × 41	3329	1.43
	121 × 121	41 × 41	7430	1.28
Fixed mesh	41 × 41	n/a	553	1
	81 × 81	n/a	2317	1
	121 × 121	n/a	5771	1

Table 4.2
CPU time comparison among one-level, two-level moving mesh and uniform mesh methods for Example 4.2. The CPU time is measured in seconds. The last column is the ratio of the used CPU time to that used with a uniform mesh of the same size.

	Fine mesh	Coarse mesh	Total CPU time	Ratio
One-level MM	41 × 41	41 × 41	1563	2.22
	81 × 81	81 × 81	45 904	15.5
	121 × 121	121 × 121	1 204 614	169.7
Two-level MM	41 × 41	41 × 41	1563	2.22
	81 × 81	41 × 41	3875	1.31
	121 × 121	41 × 41	7526	1.06
Fixed mesh	41 × 41	n/a	704	1
	81 × 81	n/a	2957	1
	121 × 121	n/a	7096	1

$$Z(x,y)=\begin{cases}100, & \text{for } (x,y)\in(\frac{3}{4},\frac{1}{4})\times(0,\frac{1}{4})\\ 50, & \text{for } (x,y)\in(\frac{5}{8},\frac{7}{8})\times(\frac{5}{8},\frac{7}{8})\\ 20, & \text{for } (x,y)\in(0,\frac{1}{2})\times(0,\frac{1}{2})\\ 10, & \text{otherwise.}\end{cases} \tag{4.2}$$

The radiation diffusion equation is integrated up to $T = 1.25$ with a fixed time step $\Delta t = 10^{-3}$.

The result obtained with a moving mesh of size 81 × 81 is shown in Fig. 4.6. The contours of the computed energy density are comparable to those obtained in [24, Fig. 6.6]. Meanwhile, one can see that the moving mesh method captures the interfaces and the onsets of the new interfaces very well.

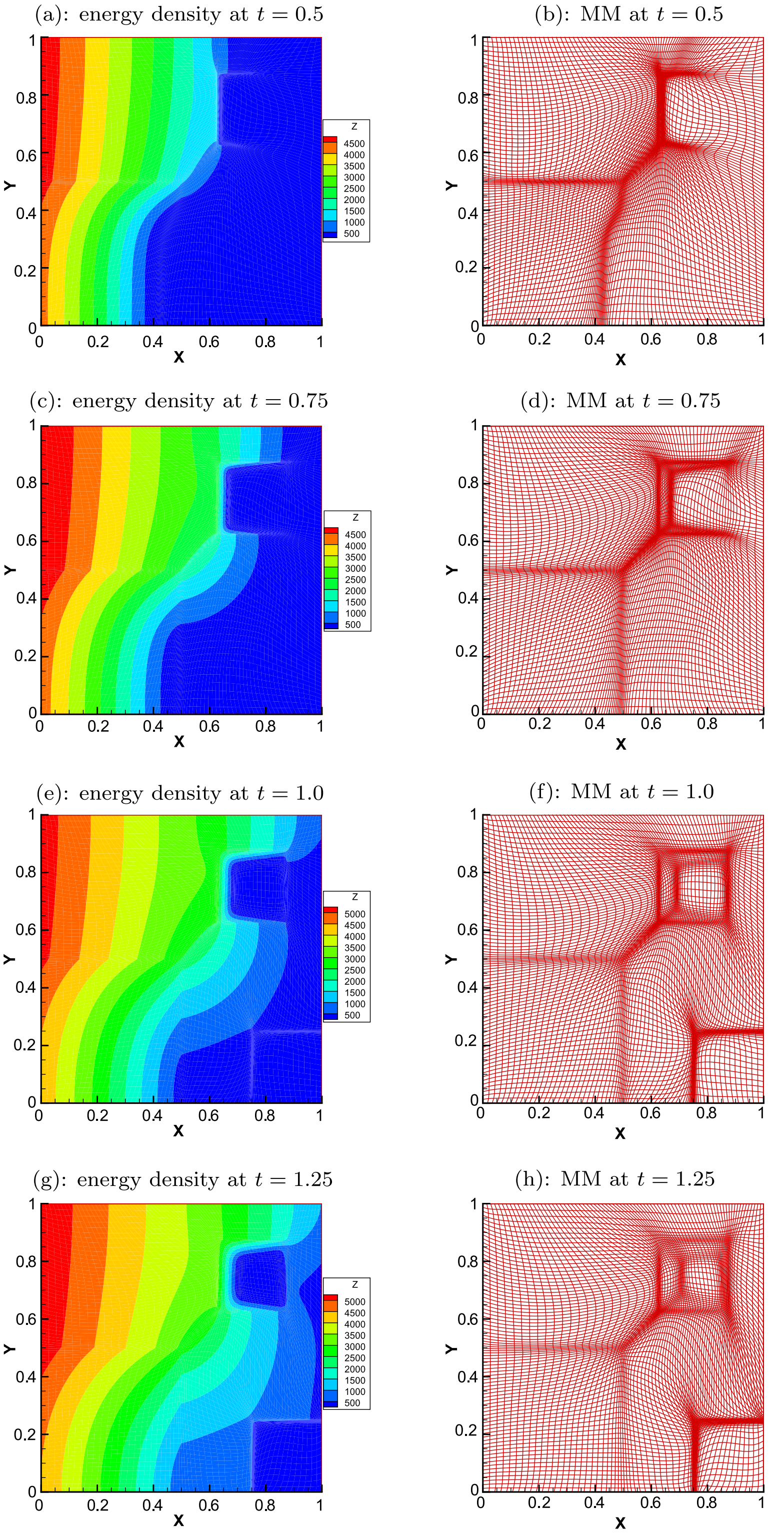
The CPU time is reported in Table 4.2. Once again, one can see that the one-level moving mesh method is costly while the cost of the two-level method stays within twice of that for fixed meshes. For instance, the CPU time of two-level moving mesh is only about 8% and 0.6% of that for the one-level moving mesh for mesh size 81×81 and 121×121, respectively. This is a significant gain of using the two-level mesh movement strategy. Moreover, the CPU difference between the two-level moving mesh and the uniform mesh is becoming smaller and smaller when the mesh is getting finer. In particular, only about 6% more CPU time is used with the two-level strategy than that for the uniform mesh for the case of mesh size 121 × 121.

Next, the moving mesh solution of mesh size 81 × 81 is compared with the uniform mesh solution with mesh size 512 × 512 in Fig. 4.7. The two sets of the solution are comparable.

Finally, the results obtained with the one-level and two-level moving meshes of size 121 × 121 are shown in Figs. 4.8 and 4.9. (A coarse moving mesh of size 41 × 41 is used in the latter case.) While both capture the interfaces and the onsets of the new interfaces, the one-level moving mesh has higher point concentration near the interfaces. On the other hand, the contours of the computed energy density are almost identical for both cases.

5. Conclusions

In the previous sections we have considered the numerical solution of two-dimensional equilibrium radiation diffusion equations with a moving mesh FD method. Radiation diffusion equations are challenging to solve numerically. The diffusion coefficient is highly nonlinear and often discontinuous over the spatial domain. This requires the use of an implicit time integration scheme and careful construction of linear and nonlinear iteration methods for the solution of the resulting nonlinear algebraic equations. Moreover, when radiation waves penetrate from hot sources to cold media, it will form a sharp front near the interface between the hot wave and the cold medium and the energy density will vary greatly in a


(a): energy density at $t = 0.5$

(b): MM at $t = 0.5$

(c): energy density at $t = 0.75$

(d): MM at $t = 0.75$

(e): energy density at $t = 1.0$

(f): MM at $t = 1.0$

(g): energy density at $t = 1.25$

(h): MM at $t = 1.25$

Fig. 4.6. Example 4.2. An adaptive moving mesh (MM) of size 81 × 81 and the contours of the computed energy density obtained thereon are plotted at $t = 0.5$, 0.75, 1.0, and 1.25.

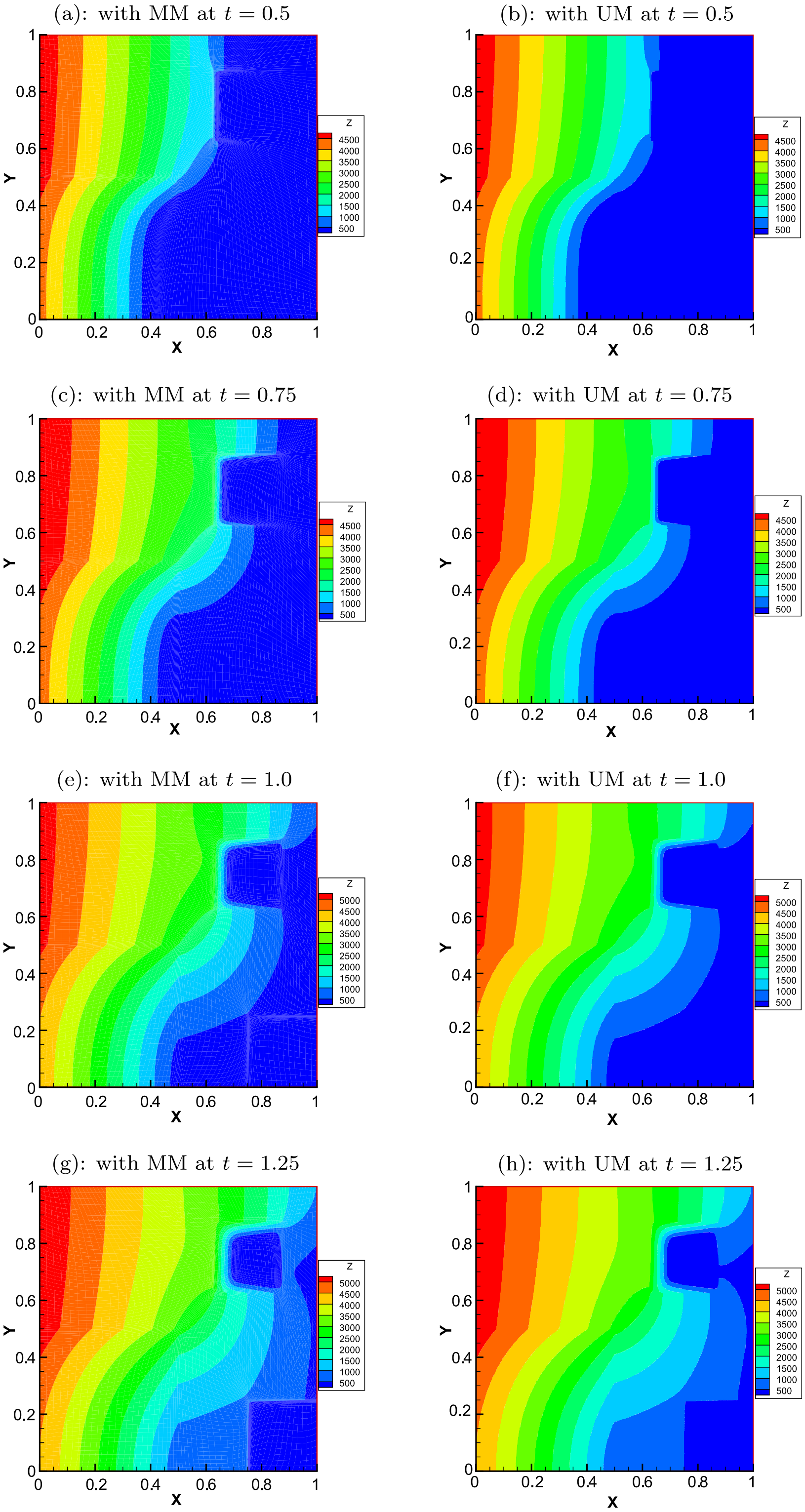
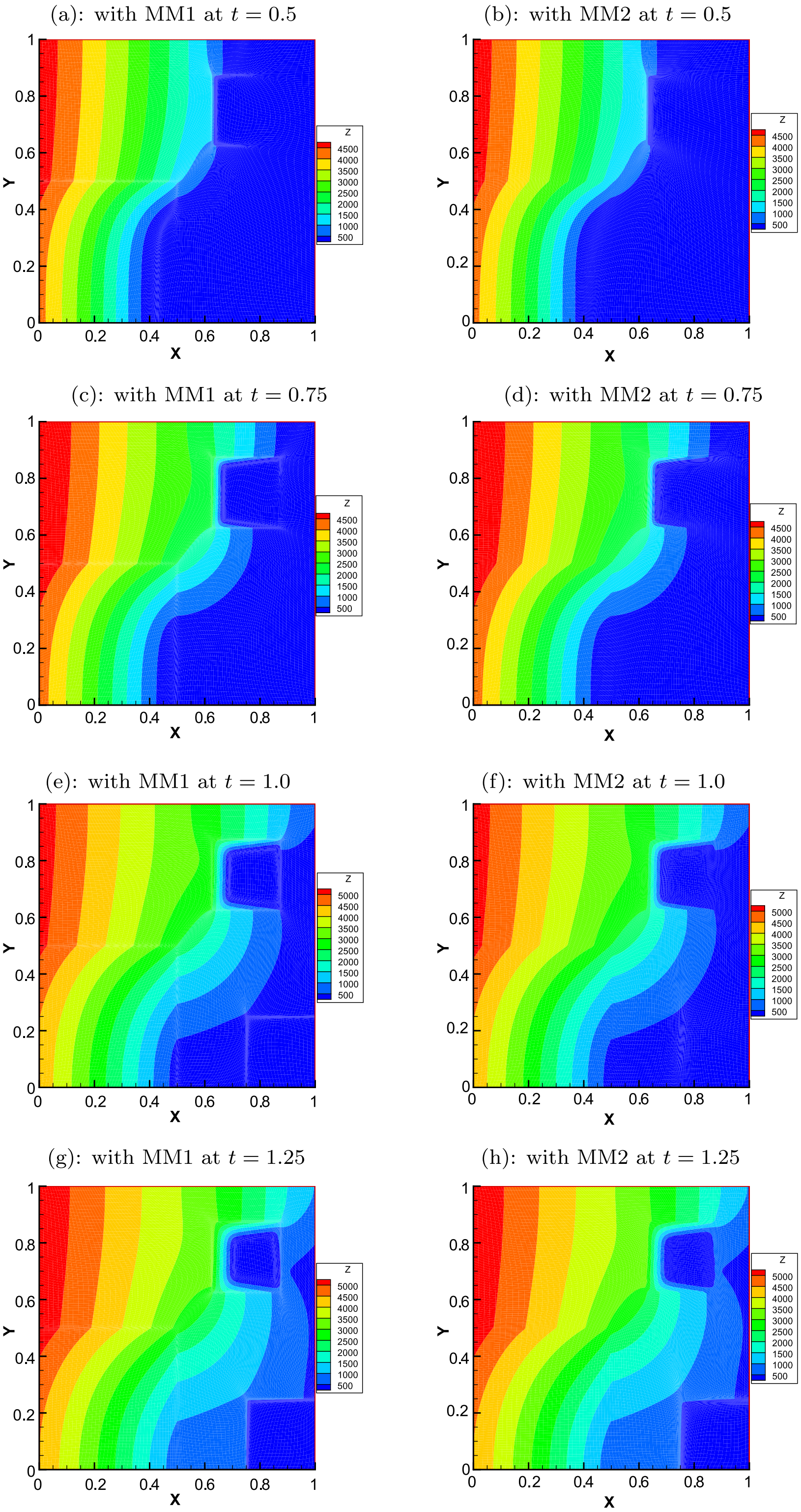


Fig. 4.7. Example 4.2. The contours of the energy density obtained with an adaptive moving mesh (MM) of size 81 × 81 are compared with those obtained with a uniform mesh (UM) of size 512 × 512.



(a): with MM1 at $t = 0.5$

(b): with MM2 at $t = 0.5$

(c): with MM1 at $t = 0.75$

(d): with MM2 at $t = 0.75$

(e): with MM1 at $t = 1.0$

(f): with MM2 at $t = 1.0$

(g): with MM1 at $t = 1.25$

(h): with MM2 at $t = 1.25$

Fig. 4.8. Example 4.2. The contours of the energy density obtained with a one-level adaptive moving mesh (MM1) of size 121 × 121 are compared with those obtained with a two-level adaptive moving mesh (MM2) of size 121 × 121 (obtained by uniformly interpolating a moving mesh of size 41 × 41).

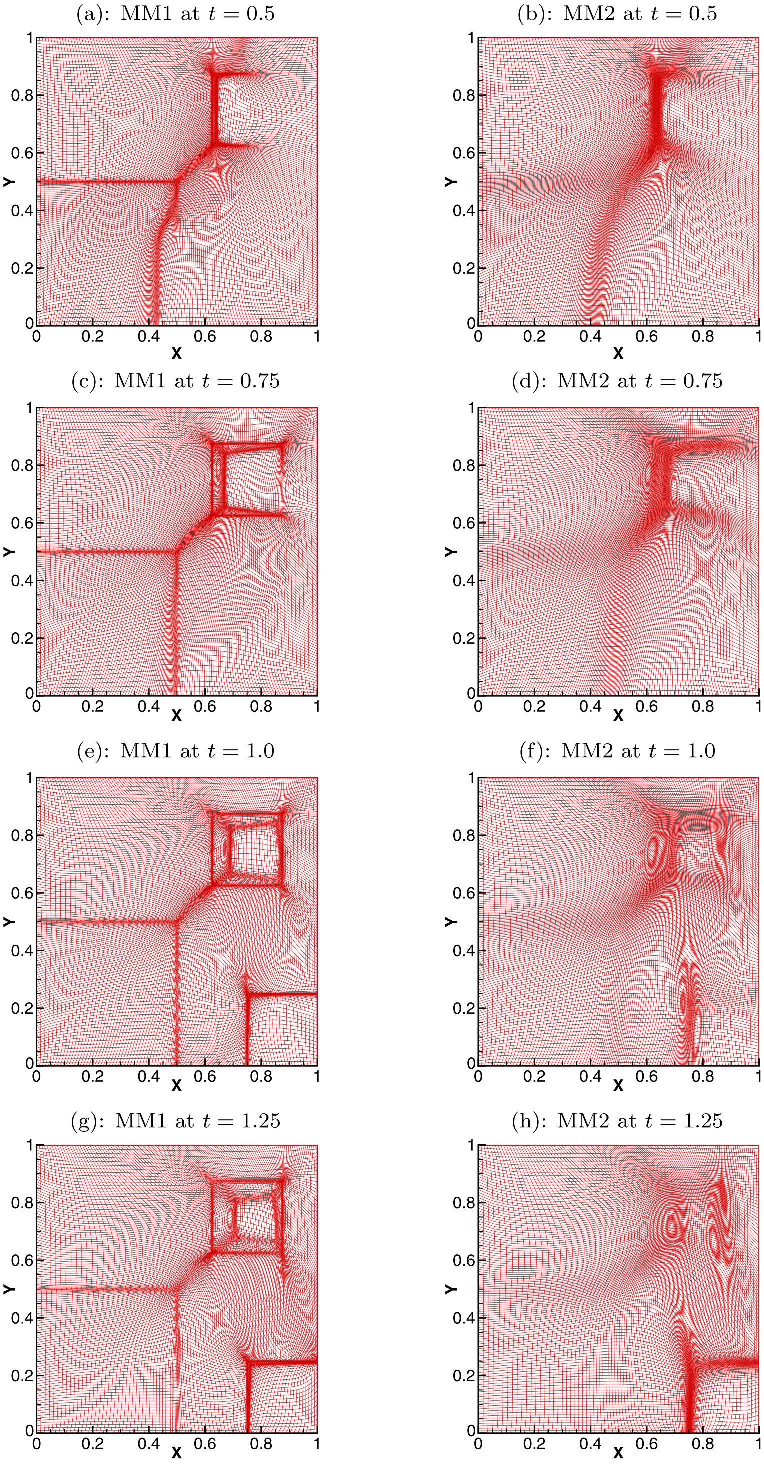


Fig. 4.9. Example 4.2. Adaptive moving meshes of size 121 × 121 obtained with one-level (MM1) and two-level (MM2) moving mesh strategies. MM2 is obtained by uniformly interpolating a 41 × 41 moving mesh.

very short distance. It is necessary to use mesh adaptation to resolve these interfaces efficiently in the numerical simulation. Furthermore, the solution of radiation diffusion equations typically stays nonnegative. Preserving this property or at least avoiding excessive negative values in the numerical solution is crucial to the success of the numerical simulation of radiation diffusion.

In this work we have employed an MMPDE moving mesh method for dynamical mesh adaptation. The method uses a meshing functional [7] based on mesh equidistribution and alignment and takes the size, shape, and orientation of mesh elements into consideration. A two-level mesh movement strategy is further utilized to improve efficiency. The nonlinearity of the equation is treated with the combination of lagged diffusion and the predictor–corrector procedure (cf. (3.8)). In the prediction stage, the diffusion coefficient is calculated using the energy density at the previous step. The energy density obtained at the prediction stage is then used for the calculation of the diffusion coefficient for the correction stage. Note that linear equations are solved at each stage. This treatment, which is equivalent to two steps of the lagged diffusion procedure, is comparable to the Beam and Warming linearization method in terms of accuracy and stability [17]. Moreover, this allows an easy and effective dealing of negative values in the computed energy density. We have used the cutoff method to replace the negative values by zero after the prediction and correction stages. This ensures the nonnegativity of numerical solution which is crucial to the continuation of the computation. It has been shown in [18] that the cutoff method retains the accuracy and convergence order of FD approximation for parabolic PDEs.

The moving mesh FD method with the above described treatments for nonlinearity and preservation of solution nonnegativity has been applied to the two-dimensional equilibrium radiation diffusion equation with multi-material, multiple spot concentration situations. The numerical experiment shows that the method is able to catch the interfaces and the onsets of the new interfaces and concentrate mesh points near them. It is also shown that the method can produce numerical results comparable to those obtained in [24] with the AMR method and those obtained with the uniform mesh of a much bigger size. Numerical examples demonstrate that the computational efficiency can be significantly improved by the two-level mesh movement strategy while maintaining a comparable accuracy. For instance, the overhead cost of the two-level strategy over that with a uniform mesh of the same size is about 28% and 6% for the first and second examples for the case of mesh size 121 × 121.

Several generalizations of the current work are under investigation. It is interesting to know how well the developed method works in three dimensions. It is also interesting to know how it works for non-equilibrium radiation diffusion systems.

References

[1] R.L. Bowes, J.R. Wilson, Numerical Modeling in Applied Physics and Astrophysics, Jones and Bartlett, Boston, 1991.
[2] J.R. Cash, Diagonally implicit Runge–Kutta formulate with error estimate, J. Inst. Math. Appl. 24 (1979) 293–301.
[3] J.I. Castor, Radiation Hydrodynamics, Cambridge University Press, 2004.
[4] T.A. Davis, Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method, ACM Trans. Math. Softw. 30 (2004) 196–199.
[5] A.S. Dvinsky, Adaptive grid generation from harmonic maps on Riemannian manifolds, J. Comput. Phys. 95 (1991) 450–476.
[6] W. Huang, Practical aspects of formulation and solution of moving mesh partial differential equations, J. Comput. Phys. 171 (2001) 753–775.
[7] W. Huang, Variational mesh adaptation: isotropy and equidistribution, J. Comput. Phys. 174 (2001) 903–924.
[8] W. Huang, Y. Ren, R.D. Russell, Moving mesh methods based on moving mesh partial differential equations, J. Comput. Phys. 113 (1994) 279–290.
[9] W. Huang, R.D. Russell, A high dimensional moving mesh strategy, Appl. Numer. Math. 26 (1998) 63–76.
[10] W. Huang, R.D. Russell, Adaptive Moving Mesh Methods, Applied Mathematical Sciences Series, vol. 174, Springer, New York, 2011.
[11] W. Huang, W. Sun, Variational mesh adaptation II: error estimates and monitor functions, J. Comput. Phys. 184 (2003) 619–648.
[12] D.A. Knoll, L. Chacon, L.G. Margolin, V.A. Mousseau, On balanced approximations for time integration of multiple time scale systems, J. Comput. Phys. 185 (2003) 583–611.
[13] D.A. Knoll, R.B. Lowrie, J.E. Morel, Numerical analysis of time integration errors for nonequilibrium radiation diffusion, J. Comput. Phys. 226 (2007) 1332–1347.
[14] D.A. Knoll, W.J. Rider, G.L. Olson, An efficient nonlinear solution method for non-equilibrium radiation diffusion, J. Quant. Spectrosc. Radiat. Transf. 63 (1999) 15–29.
[15] D.A. Knoll, W.J. Rider, G.L. Olson, Nonlinear convergence, accuracy, and time step control in nonequilibrium radiation diffusion, J. Quant. Spectrosc. Radiat. Transf. 70 (2001) 25–36.
[16] G. Lapenta, L. Chacón, Cost-effectiveness of fully implicit moving mesh adaptation: a practical investigation in 1D, J. Comput. Phys. 219 (2006) 86–103.
[17] R.B. Lowrie, A comparison of implicit time integration methods for nonlinear relaxation and diffusion, J. Comput. Phys. 196 (2004) 566–590.
[18] C. Lu, W. Huang, E.S. Van Vleck, The cutoff method for numerical computation of nonnegative solutions of parabolic PDEs with application to anisotropic diffusion and lubrication-type equations, J. Comput. Phys. 242 (2013) 24–36.
[19] S.F. McCormick, Multilevel Adaptive Methods for Partial Differential Equations, SIAM, Philadelphia, PA, 1989.
[20] S.F. McCormick, J.W. Thomas, The fast adaptive composite grid (FAC) method for elliptic equations, Math. Comput. 46 (1986) 439–456.
[21] D. Mihalas, B.W. Mihalas, Foundations of Radiation Hydrodynamics, Oxford University Press, New York, Oxford, 1984.
[22] V.A. Mousseau, D.A. Knoll, W.J. Rider, Physical-based preconditioning and the Newton–Krylov method for non-equilibrium radiation diffusion, J. Comput. Phys. 160 (2000) 743–765.
[23] S. Ovtchinnikov, X.C. Cai, One-level Newton–Krylov–Schwarz algorithm for unsteady nonlinear radiation diffusion problems, Numer. Linear Algebra Appl. 11 (2004) 867–881.
[24] M. Pernice, B. Philip, Solution of equilibrium radiation diffusion problems using implicit adaptive mesh refinement, SIAM J. Sci. Comput. 27 (2006) 1709–1726 (electronic).
[25] W.J. Rider, D.A. Knoll, G.L. Olson, A multigrid Newton–Krylov method for multimaterial equilibrium radiation diffusion, J. Comput. Phys. 152 (1999) 164–191.
[26] K.H. Winkler, M. Norman, D. Mihalas, Adaptive-mesh radiation hydrodynamics I: the radiation transport equation in a completely adaptive coordinate system, J. Quant. Spectrosc. Radiat. Transf. 31 (1984) 473.
[27] K.H. Winkler, M. Norman, D. Mihalas, Adaptive-mesh radiation hydrodynamics II: the radiation and fluid equations in relativistic flows, J. Quant. Spectrosc. Radiat. Transf. 31 (1984) 479.
[28] G. Yuan, X. Hang, Z. Sheng, J. Yue, Progress in numerical methods for radiation diffusion equations, Chin. J. Comput. Phys. 26 (2009) 475–500.
[29] J. Yue, G. Yuan, Picard–Newton iterative method with time step control for multimaterial non-equilibrium radiation diffusion problem, Commun. Comput. Phys. 10 (2011) 844–866.
[30] X. Zhao, Y. Chen, Y. Gao, C. Yu, Y. Li, Finite volume element methods for nonequilibrium radiation diffusion equations, Int. J. Numer. Methods Fluids 73 (2013) 1059–1080.