THE UNIVERSITY OF READING

DEPARTMENT OF MATHEMATICS

A Moving Mesh Finite Element Method for the Numerical Solution of Partial Differential Equations and Systems.

B. V. Wells

Thesis submitted for the degree of

Doctor of Philosophy

October 2004

Abstract

In this thesis we consider the adaptive solution of time-dependent partial differential equations using a moving mesh technique. A moving mesh method is developed utilising monitor functions to drive the mesh motion and is based on equidistribution ideas. The method is derived in multidimensions from a conservation of monitor function principle and from this initial principle a conservation law for the monitor function is derived which generates the velocity of the mesh. In dimensions higher than one the mesh velocity is underdetermined for this conservation law, therefore the *curl* of the velocity field is prescribed to obtain a unique velocity field. The conservation law together with the *curl* condition are combined to produce an elliptic equation for a mesh velocity potential, from which the mesh velocity can be obtained. The moving mesh equations are solved for the mesh velocity using standard linear finite elements and then a new mesh is constructed by integrating the mesh velocity forward in time using finite differences.

The moving mesh method is applied to the adaptive solution of parabolic and hyperbolic equations. The parabolic porous medium equation (PME) with a moving boundary problem is solved taking advantage of scale invariant properties. For this problem the solution to the PME is obtained through the conservation of monitor function for two different monitor functions; a mass and a gradient monitor function. The method is then applied to the solution of hyperbolic conservation laws in one and two spatial dimensions. In one dimension the solution to the inviscid Burgers equation and the compressible Euler equations are solved. However, in these cases the numerical solution is obtained by solving the equation on the moving mesh using the Arbitrary Lagrangian Eulerian (ALE) form of the equation. In two dimensions hyperbolic conservation laws are solved using the moving mesh method in conjunction with the mass monitor function and also a monitor function based on the gradient of the solution. The numerical solutions obtain on the various moving meshes are found to be advantageous compared to a fixed mesh Eulerian method for some standard test problems.

Acknowledgements

Firstly, I would like to thank my supervisors Mike Baines and Paul Glaister for their wisdom, guidance and enthusiasm during the project. I'd especially like to thank Mike Baines for being such an inspiration and giving me so much of his retirement time.

I would also like to thank Pete Sweby for help and useful suggestions during the PhD and also to Peter Jimack and Matthew Hubbard at the school of computing, Leeds university for their involvement in the project.

I would like to thank fellow postgraduates during my three years; especially Chris, Claire, Dave, Ken, Kev and Wako, who have made my time at Reading university such an enjoyable experience. Thanks also to friends at home for constantly encouraging me to get a job and to stop being a student!

I would like to thank my Mum and Dad, Amy, Lucy, Hannah and Grandad for their love and support during my time at university. Finally, I would like to thank Kathryn for her love and encouragement.

I acknowledge the financial support of AWE Aldermaston and would like to thank both Andy Barlow and Graham Ball for their helpful suggestions throughout the course of the project.

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Contents

List of Figures

Chapter 1

Introduction

Adaptive methods have been used for many years now for the solution of differential equations (DEs) and have become an integral part of many numerical methods. One of the attractive properties of adaptive methods is that they can often be incorporated into existing numerical methods for solving DEs and in many cases have been found to give increased efficiency and accuracy when compared to non-adaptive methods.

This thesis is primarily concerned with the adaptive solution of scalar and systems of partial differential equations (PDEs) of the form

$$\underline{\mathrm{u}}_t = \mathcal{L}\underline{\mathrm{u}}$$

where $\underline{\mathrm{u}}$ is the solution to the PDE and may either be a scalar or an m-vector of solution variables. In this work the differential operator $\mathcal{L}$ will either be of parabolic or hyperbolic type and in general will be nonlinear. This type of equation is time-dependent and therefore solutions to such equations will have transient features which are often very localised and hard to approximate numerically. Also, at certain instants the location of the feature may lie within a computational cell and this makes matters worse. For this kind of equation and for this reason it can be advantageous to use an adaptive numerical method which automatically resolves features of the flow.

There are various types of adaption that can be used to increase the accuracy and efficiency of numerical methods. The main three types of adaption are called $h-$refinement, $p-$refinement and $r-$refinement and are generally associated with finite elements. An essential part of all these methods is a numerical solution indicator which is needed to

identify where adaption should take place and often determines the effectiveness of the resulting numerical method. The solution indicator should ideally usually come from error estimates; however in many circumstances they are not available and other more heuristic techniques may have to be used. We will now briefly describe these three adaption techniques.

The h-refinement method is a technique in which extra computational nodes are added to a mesh in spatial regions where the solution is badly approximated, as identified by an error indicator. The second type of adaption is the p-refinement technique which consists of increasing the order of the numerical approximation to the solution in certain regions of the domain, to improve the local accuracy of the numerical solution. Finally, r-refinement is used to dynamically move the position of the computational nodes, keeping the total number of nodes fixed in time, to improve the overall accuracy of the numerical solution. These techniques have been used by many authors to improve the accuracy of existing numerical methods and have also been combined to obtain a more robust numerical method. For example, h-refinement and p-refinement have been combined to produce an hp-refinement method.

In this thesis we will concentrate on the r-refinement method and present a method for moving the mesh nodes. In general, the r-refinement method may be applied by targeting either the position or the velocity of the mesh points. The mesh can be construed as a mapping from a background computational space to the physical space in which the PDE is being solved, and then the velocity of the mesh can be taken to be the rate of change in time of this mapping. In this manner each computational node will have associated with it a unique velocity, at which it moves, from which the mesh can be advanced forward in time. Since there is an implicit mapping between the computational and physical spaces this velocity can be regarded as a function of the physical space variable. We therefore require a method for generating this velocity and in this work it will be constructed through the use of a monitor function. This function will be chosen to control the relative density of the mesh points in the physical domain and hence the degree of mesh adaption. The method that we will use for generating these mesh velocities will be based on a conservation of monitor function principle in time, which can be utilised to induce a motion on the mesh. From this principle an

Eulerian conservation law for the monitor can be derived, from which the mesh velocity can be obtained. This conservation law governing the motion of the mesh will be used in conjunction with a *curl* condition, which prescribes the rotational properties of the mesh, to obtain a unique mesh velocity. The conservation law together with the *curl* condition is then shown to generate an elliptic equation for the mesh velocity potential, from which the mesh velocity can be obtained. This approach for generating a moving mesh will be used for the adaptive solution of partial differential equations and in this thesis we will consider two particular types of PDE; parabolic and hyperbolic. We now describe the layout of the thesis.

Chapter 2 begins with a survey of some commonly used moving mesh methods in one spatial dimension. It also introduces many of the ideas and techniques that are used in constructing a moving mesh method. In particular, the idea of such a method being viewed as a transformation from a background computational space to the physical space in which the problem is solved is highlighted. The often used equidistribution principle of de Boor [16] is also introduced, in which some measure of the solution is equalised over each computational cell. We then outline the multidimensional extensions of these moving mesh methods. The new research area of geometric integration [25] in which properties of the continuous problem being solved are used to drive the numerical solution process is also mentioned briefly.

In chapter 3 we use many of the ideas that were described in chapter 2 to derive a moving mesh method. Since many moving mesh methods based on equidistribution are only available in one spatial dimension and are not easily generalised to higher dimensions, one of our aims will be to produce a method which can be used in an arbitrary number of spatial dimensions. Our moving mesh method is based on the use of a monitor function in a conservation principle. From this conservation principle an equation for obtaining the velocities of the mesh points is derived. It is found that this equation is insufficient to determine the mesh velocity field uniquely in spatial dimensions higher than one; therefore an additional curl condition is prescribed to overcome this uniqueness problem providing a mesh velocity potential. A weak form of the moving mesh equation for the mesh velocity potential is then derived and discretised using standard linear finite elements. A weak form is also used to obtain the mesh velocities from the mesh velocity

potential. A simple time-stepping routine is used to advance the mesh forward in time and hence generate a new mesh. In certain cases it is possible to recover the solution to the problem through the conservation of monitor function principle. The resulting method is related to some other moving mesh methods, such as the deformation map method of Liao and Anderson [64], and this is also highlighted in this chapter.

We then apply this method, in one and two dimensions, to the solution of a variety of different partial differential equations, firstly of parabolic type and secondly of hyperbolic type. In chapter 4 we consider the solution of the porous medium equation (PME) using scale invariant techniques similar to those found in [21, 22, 24, 13, 25]. The solution method and mesh motion is firstly driven by the desirability of conserving the mass of the solution in a discrete sense; therefore a mass monitor is utilised. Secondly we describe how a gradient monitor function can be used in a similar way for the solution of the PME. The monitor function based on the solution gradient moves mesh points into regions of the spatial domain where the solution gradient is high, such as in moving discontinuities. The end of the chapter details how to extend the method using the mass monitor function to two spatial dimensions. The PME in two dimensions is solved using the moving mesh method in a finite element framework on a unstructured triangular mesh. It should be noted that for this scalar equation and for these choices of monitor function it is possible to recover the solution to the PME directly from the mesh through the initial conservation of monitor function principle itself.

In chapter 5 the adaptive solution of hyperbolic conservation laws in one spatial dimension will be considered. Both the inviscid Burgers equation and the compressible Euler equations of gas dynamics will be solved using the adaptive mesh method derived in chapter 3 for two different monitor functions, the mass monitor function and a monitor function based on the gradient of the solution. It will be found that the solution to the conservation law being solved cannot be recovered directly from the mesh, as was the case in chapter 4 for the solution of the PME, and hence a finite volume method, modified to take into account the motion of the mesh, will be used. This finite volume method is based on the work of Harten and Hyman [43] and solves the Arbitrary Lagrangian Eulerian (ALE) form of the conservation laws. The ALE finite volume method is then used in conjunction with an ALE version of the Roe approximate Riemann solver

to evaluate intercell fluxes. The spatial and temporal accuracy of the resulting finite volume method is only first order, therefore the order of the numerical method is increased, modified by the use of flux limiters. This moving method with an ALE solver is then used in one spatial dimension to solve a moving shock problem for the inviscid Burgers equation and the Sod shock tube for the compressible Euler equations.

In chapter 6 the compressible Euler equations in two spatial dimensions are solved using moving mesh techniques similar to the ones described in chapter 5. Again the moving mesh method will be used along with the mass monitor function and a monitor function based on the solution gradient. The Euler equations will again be solved on a moving mesh, so the ALE version of the conservation laws will be used with a finite volume method modified to take into account the motion of the mesh. The intercell fluxes are evaluated with the use of an ALE HLLC approximate Riemann solver [94]. This leads to a finite volume method which is first order in space and time, therefore the MUSCL technique of Van Leer is utilised to increase the accuracy of the method in space. The Euler equations are solved on an unstructured triangular mesh in two dimensions using the cell-centred ALE finite volume method and again the mesh is moved in a finite element framework. The resulting method is then used to solve two cylindrical shock problems.

The final chapter summarises the work that has been carried out in this thesis and considers some possible further work.

Chapter 2

Mesh Generation and Adaption

Mesh generation and adaption techniques have been used for many years now to produce irregular grids upon which both ordinary and partial differential equations can be solved. A wide variety of methods have been devised which in general fall into one of two main categories. These categories are referred to as velocity or location based methods. The velocity based methods are derived by introducing the velocities of the computational nodes in the mesh. Alternatively, the location based methods seek the position of the computational nodes in terms of a mapping from a reference mesh. We will start this chapter by introducing some of the commonly used ideas for generating non-uniform meshes and then go on to describe some of the popular velocity and location moving mesh methods.

2.1 Introduction

Most methods for generating an irregular mesh are constructed as a transformation from a background computational domain into the physical domain in which the particular problem is being solved. In this work we will let $\mathbf{x}$ and $\boldsymbol{\xi}$ denote physical and computational co-ordinates which are defined on the domains Ω and Ω_c, respectively. Then we suppose that there exists a one-to-one transformation denoted by

$$\mathbf{x} = \mathbf{x}(\boldsymbol{\xi}, t),$$

which takes points in the computational space at a certain time into the physical space. Such a mapping is shown in figure 2.1. The aim of an irregular mesh method is to

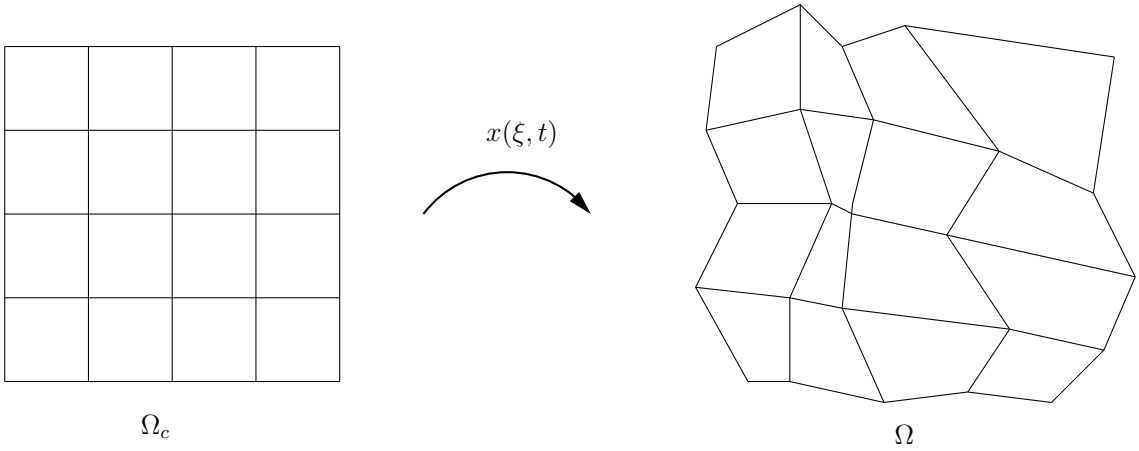

Figure 2.1: Figure showing the mapping $\mathbf{x}(\boldsymbol{\xi}, t)$ *from the background computational mesh* Ω_c *to the physical mesh* Ω*.*

control this mapping to obtain the desired clustering of mesh points and alignment of the computational cells.

We now consider ways of generating this mapping. One of the most widely used principles for generating a non-uniform mesh is that of equidistribution. The equidistribution principle was originally introduced by De Boor [16] for obtaining a discrete approximation to a function on a non-uniform mesh and involves choosing the mesh points such that the integral of some measure is equalised over each computational cell of the mesh. This measure is user-defined and referred to as the monitor function. The monitor function is chosen to reflect the characteristics of the numerical solution or function being approximated on the mesh. It is required to be positive and may depend on the solution and/or its derivatives. Therefore a general form of the monitor function in multidimensions is given by

$$M = M(\mathbf{x}, t, u, u_x, u_y, \dots).$$

where u is the function being approximated. This function u may be a given function or alternatively may be the solution to a differential equation.

We will now describe the equidistribution principle in one spatial dimension and then go on to look at multidimensional extensions of it. Without loss of generality we have taken the the domains Ω and Ω_c to be the unit interval $[0, 1]$ in real space, so $x, \xi \in [0, 1]$.

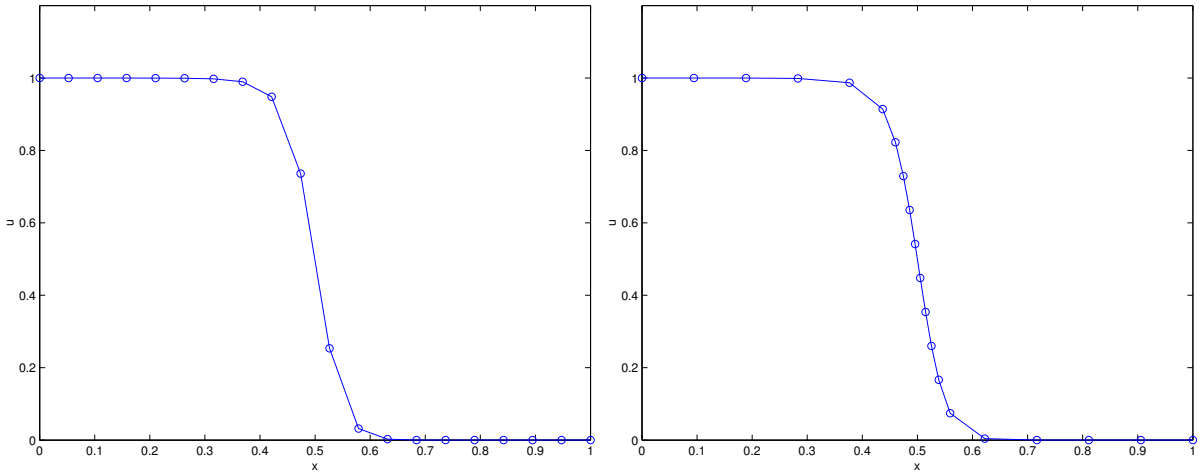

Figure 2.2: Figures illustrating the equidistribution principle. (Left) a function represented on equally spaced mesh (right) the same function represented using equal arc-length. Both graphs are computed with 20 computational nodes.

We also have the boundary conditions for the mapping that $x(0,t) = 0$ and $x(1,t) = 1$. Then the continuous form of the equidistribution principle in the integral form of White [99] is given as

$$\int_0^{x(\xi,t)} M\left(x\left(\xi,t\right),t\right)\,\mathrm{d}x = \xi \int_0^1 M\left(x\left(\xi,t\right),t\right)\,\mathrm{d}x, \quad \forall t. \tag{2.1}$$

We now give a simple illustration of the equidistribution principle. Consider the function

$$u(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(20\left(x^2 - \frac{1}{4}\right)\right)$$

defined on the interval $x \in [0,1]$. We wish to represent this function discretely on a mesh with only a finite number of mesh points given as

$$0 \equiv x_0 < x_1 < \cdots < x_{N-1} < x_N \equiv 1.$$

Figure 2.2 shows the function u represented on two different meshes. The left figure shows the function on an equally spaced mesh, i.e. such that $x_i - x_{i-1} = x_{i+1} - x_i$ for $i = 1, \ldots N-1$. Note that this mesh can be seen as generated from the equidistribution principle (2.1) with the monitor function taken to be $M = 1$. Alternatively the right

figure shows the same function, but this time represented on a mesh generated from the equidistribution principle with the arc-length monitor function

$$M = \sqrt{1 + \left(\frac{\mathrm{d}u}{\mathrm{d}x}\right)^2}.$$

This choice of monitor function causes mesh points to cluster more in and around regions of large variation in the function and appears to give a"better" discrete approximation of the function.

Following Huang, Ren and Russell [49], the equidistribution principle given by (2.1) can be differentiated with respect to ξ to obtain

$$M\left(x\left(\xi,t\right),t\right)\,\frac{\partial}{\partial\xi}x\left(\xi,t\right) = \theta(t), \tag{2.2}$$

where $\theta(t) = \int_0^1 M\left(x\left(\xi,t\right),t\right)\,\mathrm{d}x$. The time-independent version of equation (2.2) was solved numerically by White in [99] to produce an adaptive mesh for the solution of two-point boundary-value problems. Differentiating the equidistribution principle twice with respect to ξ gives

$$\frac{\partial}{\partial\xi}\left(M\left(x\left(\xi,t\right),t\right)\,\frac{\partial}{\partial\xi}x\left(\xi,t\right)\right) = 0. \tag{2.3}$$

Equations (2.2) and (2.3) are referred to as Quasi-static equidistribution principles (QSEPs) in [49] since they contain no information about how the computational nodes are moving. The time-independent version of equation (2.3) was solved numerically by Baines in [7] using an iterative procedure, although there are queries over stability of some solvers. Since the monitor function is in general a function of x, equation (2.3) will be a non-linear equation for the mesh position, so the equation is solved iteratively in the following manner,

$$\frac{\partial}{\partial\xi}\left(M(x^p)\frac{\partial x^{p+1}}{\partial\xi}\right) = 0,$$

where p is the iteration counter. Discretising this equation leads to a tridiagonal system which can be solved using a Jacobi iteration. When this process is used to produce an

equidistributed mesh to give a good approximation of a given function, the values of the monitor function are available and this iterative process is usually efficient. However, when this process is used to generate a mesh for the solution of a differential equation it is often more computationally efficient to use an interleaving approach where the mesh and solution are updated alternatively. When an updated mesh has been generated, interpolation can be used to transfer the solution from the old mesh to the new one.

In [3] Flaherty *et al* derive a moving mesh equation by differentiating the equidistribution principle (2.1) with respect to time to obtain

$$M(x(\xi,t),t)\,\dot{x}(\xi,t) + \int_0^{x(\xi,t)} \frac{\partial M}{\partial t}\,\mathrm{d}x = \xi\dot{\theta}(t), \tag{2.4}$$

where $\dot{x} = \dot{x}\,(\xi,t)$ is the rate of change of the mapping x with respect to time and is called the mesh velocity. Assuming that $\xi_i = \frac{i}{N}$ and differencing equation (2.4) from ξ_{i-1} to ξ_i the discrete ODE system

$$M_i(t)\,\dot{x}_i(t) - M_{i-1}(t)\,\dot{x}_{i-1}(t) + \int_{x_{i-1}(t)}^{x_i(t)} \frac{\partial M}{\partial t}\,\mathrm{d}x = \frac{1}{N}\,\dot{\theta}(t) \quad \text{for } i = 1,\ldots,N$$

is obtained. This discretisation was later analysed in [33] to determine whether the method is stable to linear perturbations. More will be said about this later in this section.

In [79] Ren and Russell obtain a conservative form of a moving mesh equation by differentiating (2.2) with respect to time to produce

$$\frac{\partial}{\partial\xi}\,(M\,\dot{x}) + \left.\frac{\partial M}{\partial t}\right|_{\xi\ fixed} \frac{\partial x}{\partial\xi} = \dot{\theta}, \tag{2.5}$$

This equation can be transformed into physical co-ordinates to give

$$\frac{\partial}{\partial x}\,(M\,\dot{x}) + \left.\frac{\partial M}{\partial t}\right|_{x\ fixed} = \frac{M\,\dot{\theta}}{\theta}. \tag{2.6}$$

In [79] Ren and Russell give a physical interpretation of these moving mesh equations in terminology that is familiar in fluid dynamics literature. If $\dot{\theta}(t) \equiv 0$ then equation

(2.6) becomes the Euler equation for the "fluid" with density function $M(x,t)$. It is also noticed that meshes generated using a static re-gridding procedure can be viewed as corresponding to the case of steady-flow where the density function $M(x,t)$ is independent of time and hence $\frac{\partial M}{\partial t} = 0$.

It is evident that the moving mesh equation of Flaherty *et al* (2.4) is mathematically equivalent to the equations of Ren and Russell (2.5) and (2.6). However, Ren and Russell [79] show that in actual fact the semi-discrete and stability properties of the moving mesh equations may be different, depending on how the equations are discretised.

In [33, 79, 49] the continuous and discrete properties of equations (2.4) and (2.6) are analysed. Linear perturbation analysis is applied by introducing the perturbation $x \to x + \delta x$, where the perturbation δx is assumed to be much smaller than unity. Then, provided no additional errors are introduced, equation (2.6) is found to be stable to linear perturbations provided that the condition

$$\max_{0\le\xi\le 1} \left| \frac{M(x(\xi,0),0)\,\theta(t)}{M(x(\xi,t),t)\,\theta(0)} \right| \le 1$$

holds. A condition on whether mesh crossing will occur is also derived. It is noted that if the Jacobian $\frac{\partial x}{\partial \xi}$ of the transformation from the computational to the physical co-ordinates is positive for all t, then no mesh crossing will occur. Equation (2.6) can be rearranged to give

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial x}{\partial \xi}\right) + \frac{1}{M}\left(\frac{\partial M}{\partial t} + \dot{x}\frac{\partial M}{\partial x}\right)\frac{\partial x}{\partial \xi} = \frac{\dot{\theta}}{M},$$

or

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial x}{\partial \xi}\right) + \frac{\dot{M}}{M}\frac{\partial x}{\partial \xi} = \frac{\dot{\theta}}{M}.$$

This is now a first order ordinary differential equation for the Jacobian $\frac{\partial x}{\partial \xi}$ and has the solution

$$\frac{\partial x}{\partial \xi}(\xi,t) = \frac{\theta(t)-\theta(0)}{M(x(\xi,t),t)} + \frac{M(x(\xi,t),0)}{M(x(\xi,t),t)}\frac{\partial x}{\partial \xi}(\xi,0).$$

It can then be shown that that if $\frac{\partial x}{\partial \xi}(\xi, 0) > 0$ the Jacobian $\frac{\partial x}{\partial \xi}(\xi, t) > 0$ provided that

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_0^1 M(x(\xi, t), t)\, \mathrm{d}x \geq 0.$$

Huang, Ren and Russell state in [49] that the quantity $\theta(t)$ in equation (2.6) is not convenient for actual numerical simulations. They therefore go on to derive a series of Moving Mesh Partial Differential Equations (MMPDEs) by eliminating this term. Firstly, differentiating equation (2.3) with respect to time gives the equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial}{\partial \xi}\left(M\left(x\left(\xi, t\right), t\right)\, \frac{\partial}{\partial \xi} x\left(\xi, t\right)\right)\right) = 0,$$

which upon rearrangement produces the MMPDE

$$\frac{\partial}{\partial \xi}\left(M\, \frac{\partial \dot{x}}{\partial \xi}\right) + \frac{\partial}{\partial \xi}\left(\frac{\partial M}{\partial \xi}\, \dot{x}\right) = -\frac{\partial}{\partial \xi}\left(\frac{\partial M}{\partial t}\, \frac{\partial x}{\partial \xi}\right). \qquad \text{(MMPDE1)}$$

It can easily be seen that moving mesh method equation MMPDE1 does not contain the troublesome quantity $\theta(t)$. Huang *et al* note that MMPDE1 has a zero speed solution when $\frac{\partial M}{\partial t} = 0$ and therefore if the monitor function $M(x, t)$ is independent of time the mesh will not move. In this sense this term can be seen as the source of the mesh movement for this MMPDE. They also state that $\frac{\partial M}{\partial t}$ is hard to approximate in real applications.

Huang *et al* also develop other MMPDEs which contain a correction term so that the mesh can be required to satisfy the equidistribution principle at a later time given by $t + \tau$ $(0 < \tau << 1)$, where τ is referred to as the relaxation parameter. Using this methodology MMPDEs 2-4 are derived. The mesh is chosen to satisfy the quasi-static equidistribution principle (2.2) at a later time, therefore

$$\frac{\partial}{\partial \xi}\left(M\left(x\left(\xi, t+\tau\right), t+\tau\right) \frac{\partial}{\partial \xi} x\left(\xi, t+\tau\right)\right) = 0. \qquad (2.7)$$

Then using the Taylor series expansions

$$\frac{\partial}{\partial \xi} x\left(\xi, t+\tau\right) = \frac{\partial}{\partial \xi} x\left(\xi, t\right) + \tau \frac{\partial}{\partial \xi} \dot{x}\left(\xi, t\right) + O\left(\tau^2\right)$$

and

$$M\left(x\left(\xi, t+\tau\right), t+\tau\right) = M\left(x\left(\xi, t\right), t\right) + \tau\, \dot{x} \frac{\partial}{\partial x} M\left(x\left(\xi, t\right), t\right) + \tau \frac{\partial}{\partial t} M\left(x\left(\xi, t\right), t\right) + O\left(\tau^2\right)$$

in equation (2.7) and neglecting the higher-order terms we obtain the MMPDE given by

$$\frac{\partial}{\partial \xi}\left(M \frac{\partial \dot{x}}{\partial \xi}\right) + \frac{\partial}{\partial \xi}\left(\frac{\partial M}{\partial \xi} \dot{x}\right) = -\frac{\partial}{\partial \xi}\left(\frac{\partial M}{\partial t} \frac{\partial x}{\partial \xi}\right) - \frac{1}{\tau} \frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right). \qquad \text{(MMPDE2)}$$

The last term in MMPDE2 measures how well the mesh satisfies the equidistribution principle. Also it is noted that even when the monitor function $M(x,t)$ is independent of t and the mesh is not equidistributed MMPDE2 still moves the mesh towards equidistribution. Huang, Ren and Russell argue that since the forcing term involving $\frac{\partial M}{\partial t}$ is less important in MMPDE2 it is reasonable to discard it. This leads to two simplified MMPDEs.

$$\frac{\partial^2}{\partial \xi^2}\left(M \dot{x}\right) = -\frac{1}{\tau} \frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) \qquad \text{(MMPDE3)}$$

$$\frac{\partial}{\partial \xi}\left(M \frac{\partial \dot{x}}{\partial \xi}\right) = -\frac{1}{\tau} \frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) \qquad \text{(MMPDE4)}$$

It is hoped that this regularisation of the moving mesh equations will produce smoother mesh trajectories and hence meshes which are more stable.

Moving mesh partial differential equations are also derived based on attraction and repulsion pseudo-forces [49]. In this work it is assumed that the computational nodes are attracted to others when a measure of the error is larger than average, while if the measure is smaller than average the surrounding nodes are repelled. The error measure, denoted by W, can be chosen to be related to the monitor function in the following way,

$$W = M\frac{\partial x}{\partial \xi}.$$

Anderson in [2] uses the moving mesh PDE

$$\dot{x} = \frac{1}{\tau}\frac{\partial}{\partial \xi}\left(M\frac{\partial x}{\partial \xi}\right) = \frac{1}{\tau}\frac{\partial W}{\partial \xi} \qquad \text{(MMPDE5)}$$

to compute the mesh velocity. Notice that this MMPDE moves mesh points towards regions where the error indicator W is large and when the mesh is equidistributed in the sense that

$$M\frac{\partial x}{\partial \xi} = \text{constant},$$

there will be no mesh movement. In MMPDE5 τ is again a positive relaxation constant.

Adjerid and Flaherty [1] use a method within a finite element framework which can be seen as a discretisation of

$$\frac{\partial^2 \dot{x}}{\partial \xi^2} = -\frac{1}{\tau}\frac{\partial}{\partial \xi}\left(M\frac{\partial x}{\partial \xi}\right) \qquad \text{(MMPDE6)}$$

in which the monitor function is taken to be an local error estimator. In this manner they seek to produce a mesh on which the local error is equidistributed. An $h-$refinement strategy is also used in conjunction with this moving mesh equation to solve parabolic partial differential equations.

Huang *et al* [49] also derive a seventh MMPDE, ignoring spatial smoothing, which can be shown to correspond to the discrete mesh equation of Dorfi and Drury in [37] given as

$$\frac{\partial}{\partial \xi}\left(M\,\frac{\partial \dot{x}}{\partial \xi}\right) - 2\,\frac{\partial}{\partial \xi}\left(M\,\frac{\partial x}{\partial \xi}\right)\,\frac{\partial \dot{x}}{\partial \xi}\Big/\frac{\partial x}{\partial \xi} = -\frac{1}{\tau}\,\frac{\partial}{\partial \xi}\left(M\,\frac{\partial x}{\partial \xi}\right). \qquad \text{(MMPDE7)}$$

All the MMPDEs derived in [49] were theoretically analysed to show stability of the resulting meshes if the MMPDE were solved exactly. This work followed on from earlier

stability analysis of Flaherty *et al* [33]. Some of these moving mesh methods were later analysed in greater detail by Li, Petzold and Ren in [63]. MMPDEs 1-7 have been used by many authors to produce adaptive meshes for a variety of applications.

So far in this chapter we have described the derivation and analysis of a variety of moving mesh methods, but up until now no choice for the monitor function has been made for actual applications. One of the most popular choices is the arc-length monitor function which was discussed earlier. This monitor function has been found to give good results for many parabolic problems. However, for real applications with very steep fronts this monitor function may move more mesh points than are necessary into the layer and hence over-resolve the feature. Due to this the arc-length monitor function is often modified and the gradient term may be weighted by a parameter to give

$$M = \sqrt{1 + \alpha \left(\frac{\partial u}{\partial x} \right)^2}. \tag{2.8}$$

The parameter α allows the magnitude of the monitor function to be reduced in regions where the gradient is large. In [86] Stockie, Mackenzie and Russell used this type of monitor function along with MMPDE4 for the adaptive solution of hyperbolic conservation laws. The conservation laws were solved using the high-resolution finite volume software package CLAWPACK [59] which had to be modified to take into account the motion of the mesh. The mesh equation and the physical PDE were then solved alternately and this process was iterated until the mesh equation had converged. This work was based on that of Harten and Hyman [43] who had originally shown how to use a Godunov scheme on an adaptively moving mesh. The monitor function (2.8) was used for the solution of the inviscid Burgers equation and the Buckley-Leverett equation using different choices of the parameter α. The Euler equations of gas dynamics were also solved on a moving mesh. Two monitor functions were constructed to capture certain features of the flow. These monitor functions are given by

$$M^S = \sqrt{1 + \alpha \left(\frac{|v_x|}{\max\limits_x |v_x|} \right)^2} \tag{2.9}$$

and

$$M^C = \sqrt{1 + \alpha \left(\frac{|S_x|}{\max\limits_x |S_x|} \right)^2} \tag{2.10}$$

where v is the velocity of the gas and S is the entropy of the gas. The monitor functions given by (2.9) and (2.10) were developed to resolve shock and contact discontinuities, respectively. A convex combination of these two monitor functions was also used to resolve both contact discontinuities and shock waves.

In early work by White [99] three monitor functions are derived for the solution of two-point boundary problems. These are the aforementioned arc-length monitor and two monitors based on error estimates. Monitor functions based on the single step error and the local truncation error are given by

$$M = \sqrt[6]{1 + ||u_{xxx}||_2^2}$$

and

$$M = \sqrt[4]{1 + ||u_{xxx}||_2^2}$$

respectively. Qiu and Sloan [77] use the monitor function

$$M = \left[1 + \alpha^2 (1 - u)^2 + \beta^2 (\gamma - u)^2 \left(\frac{\partial^2 u}{\partial x^2} \right)^2 \right]^{\frac{1}{2}}$$

where α, β and γ are parameters which have to be chosen for the problem. This monitor function was designed to be used with MMPDE6 for the solution of Fisher's equation, given as

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x} + u\,(1 - u)\,,$$

once both arc-length and curvature monitors had previously failed to give good results. The MMPDE approach for the solution of this reaction-diffusion equation was also compared to the Moving Mesh Differential Algebraic Equation (MMDAE) technique of

Mulholland, Qiu and Sloan [71]. In the MMDAE method the mesh is moved through the QSEP given by (2.3) which will approximately equidistribute the mesh at each time level. The equation for the mesh is then coupled with the Lagrangian form of the PDE being solved, which for the PDE $u_t = \mathcal{L}u$ is given by

$$\frac{\mathrm{D}u}{\mathrm{D}t} - \dot{x}\,\frac{\partial u}{\partial x} = \mathcal{L}u. \tag{2.11}$$

The system of Differential Algebraic Equations (DAEs) is then integrated forward in time using the stiff ODE/DAE solver DASSL [76]. It is found in this work that the MMDAE leads to a more reliable approach when compared to that of the MMPDE, this being due to the fact that the temporal smoothing parameter τ in MMPDE6 has to be chosen carefully to be consistent with the solution being computed. Further comparisons between the MMPDE and MMDAE approaches are made in [78] for the solution of Burgers' equation.

For many applications the monitor function is often modified by a smoothing process. Dorfi and Drury [37] and Furzeland *et al* [40] found that, in order to obtain a good approximation to a PDE on moving mesh, the mesh should be in some sense smoothed. They argue that abrupt variations in the mesh can cause deterioration in convergence and accuracy of the solution being approximated on the mesh and therefore mesh smoothness can become an important factor in moving mesh computations. Also, large variations in the size of the computational cells can cause the system to become stiff and hence harder to integrate forward in time. Verwer *et al* [96] proved that smoothing the mesh will be basically equivalent to smoothing the monitor function on the mesh. A common method for smoothing the monitor function over the mesh, given in [48], is

$$\tilde{M}_i = \sqrt{\frac{\sum_{k=i-p}^{i+p} (M_k)^2 \left(\frac{\gamma}{\gamma+1}\right)^{|k-i|}}{\sum_{k=i-p}^{i+p} \left(\frac{\gamma}{\gamma+1}\right)^{|k-i|}}}$$

where $\tilde{M}_i$ is the smoothed value of the monitor function at the i^{th} computational node, γ is the smoothing parameter and p is a positive integer referred to as the smoothing

index and determines how many nodes the monitor is smoothed over. This smoothed monitor function has been used extensively in [86, 77, 78]. Huang and Russell [50] later analysed MMPDEs when the monitor function had been smoothed by introducing the PDE

$$\tilde{M} + \frac{1}{\lambda^2} \frac{\partial^2 \tilde{M}}{\partial \xi^2} = M,$$

where λ is a positive constant. Boundary conditions are also required for this smoothing process. Using this type of continuous smoothing process the continuous properties of the MMPDEs derived earlier are examined and conditions are derived which will guarantee non-singularity of the resulting mesh.

In one spatial dimension many equations have been derived for moving the computational mesh. In the next section we will consider moving mesh methods in higher dimensions. As we will see, many moving mesh methods in higher dimensions started as static grid generation techniques, but were later extended to produce meshes for the solution of time-dependent problems. One of the most popular ways for generating an adaptive mesh in one-dimension was that of the equidistribution principle. However we will see that this principle does not easily extend into higher dimensions.

2.2 Moving Mesh Methods in Higher Dimensions

2.2.1 Grid Generation

We begin this section by outlining some popular methods for generating a static mesh in multidimensions. We will then go on to explain how these static mesh generation techniques may be extended to cope with time-dependent problems. Perhaps one of the earliest methods for generating a mesh in multi-dimensions is that of Winslow [101]. The ideas behind this method provided a setting for many mesh generation techniques that were to follow.

The main idea behind Winslow's method is to formulate the mesh generation problem as a potential problem where the mesh lines behave as equipotential lines. The equipotential lines can then be used to define the mesh. (The main body of [101] was devoted

to the solution of quasi-linear PDEs on triangular meshes, with only the appendix relating to the mesh generation problem.) The method is as follows. Let two sets of equipotentials defined as $\xi = \xi(x, y)$ and $\eta = \eta(x, y)$ satisfy the Laplace equations

$$\nabla^2 \xi = 0 \tag{2.12}$$

$$\nabla^2 \eta = 0 \tag{2.13}$$

in some region Ω. The solution to equations (2.12) and (2.13) gives the equipotentials $\xi = \textit{constant}$ and $\eta = \textit{constant}$ from which a mesh can be constructed by considering the intersection of these lines. The desired mesh can be obtained numerically from inverting the mappings to give $x = x(\xi, \eta)$ and $y = y(\xi, \eta)$. Using the Jacobian determinant $J = x_\xi \, y_\eta - x_\eta \, y_\xi$ equations (2.12) and (2.13) can be transformed to give

$$\alpha \, x_{\xi\xi} - 2\beta \, x_{\xi\eta} + \gamma \, x_{\eta\eta} = 0 \tag{2.14}$$

$$\alpha \, y_{\xi\xi} - 2\beta \, y_{\xi\eta} + \gamma \, y_{\eta\eta} = 0 \tag{2.15}$$

where

$$\begin{aligned} \alpha &= x_\eta^2 + y_\eta^2 \\ \beta &= x_\xi \, x_\eta + y_\xi \, y_\eta \\ \gamma &= x_\xi^2 + y_\xi^2. \end{aligned}$$

Therefore the solutions to the inverse Laplace equations (2.14) and (2.15) give the coordinates x, y of a given equipotential directly.

These equations were discretised using a finite-difference method and then solved using an iterative successive over-relaxation procedure. The details of this algorithm can be found in the main section of [101]. This method was originally developed to generate triangular meshes, but may also be used to generate meshes of quadrilateral cells.

A nice property of Winslow's method is that the resulting mesh can in some sense be seen as smooth since the mapping from the background domain into the physical one comes from Laplace's equation. However one problem with the method is that the structure

of the interior mesh depends entirely on the boundary of the domain. Therefore it is difficult to control the mesh point locations locally for arbitrarily shaped domains. It should also be noted that there is no explicit control over the mesh through the use of a function, although a modification was made to Winslow's method to produce the variable diffusion method [102]. In this method equations (2.12) and (2.13) are modified to include a weight function w which can be used to control the adaption. The modified equations are given as

$$\nabla \cdot \left(\frac{1}{w} \nabla \xi \right) = 0 \tag{2.16}$$

$$\nabla \cdot \left(\frac{1}{w} \nabla \eta \right) = 0. \tag{2.17}$$

The main use of Winslow's method was to produce meshes which were adapted to particular domains. Thompson, Thames and Mastin in [92] used Winslow's method to generate body-fitted meshes around multiple curvilinear boundaries. In particular they generated meshes around multiple airfoil shapes.

Also in [92] Thompson *et al* extended the method of Winslow to allow some control over the position of the interior mesh points. They introduced control functions as inhomogeneous terms in the Laplace equations to give control over the mesh. Therefore their mesh equations were given as

$$\nabla^2 \xi = P \tag{2.18}$$

$$\nabla^2 \eta = Q \tag{2.19}$$

where P and Q are called mesh control functions. Equation (2.18) and (2.19) are often referred to as the TTM mesh generator after its developers.

Brackbill and Saltzman [18] developed a mesh generator that attempted to control various properties of the mesh, such as smoothness, concentration and orthogonality. In [18] Brackbill and Saltzman show that Euler-Lagrange variational equations for the minimization of the functional

$$I_S = \int_\Omega \left(|\nabla \xi|^2 + |\nabla \eta|^2 \right) \, \mathrm{d}\Omega \tag{2.20}$$

are the Laplace equations given by (2.12) and (2.13). The concentration and orthogonality of the mesh lines was incorporated into the method by adding extra terms to the variational principle. Control on the orthogonality of the mesh can be included through the use of the functional

$$I_O = \int_\Omega (\nabla\xi \cdot \nabla\eta)^2 \, \mathrm{d}\Omega.$$

This functional is chosen since when the mesh is orthogonal $\nabla\xi \cdot \nabla\eta = 0$, the functional is minimised. A Jacobian weight function may also be included in functional I_O so that larger computational cells will be more strongly persuaded to be orthogonal. The concentration of the mesh is then controlled by introducing the functional

$$I_C = \int_\Omega w(\xi, \eta)\, J \, \mathrm{d}\Omega,$$

where J is the Jacobian determinant as before and $w(\xi, \eta)$ is a weight function which causes the mesh cells to be small where it is large. The mesh is then generated by taking a linear combination of the three functionals in the following manner.

$$I = \lambda_S \, I_S + \lambda_O \, I_O + \lambda_C \, I_C \tag{2.21}$$

The properties of the resulting mesh can now be controlled by the choice of the constants λ_S, λ_O and λ_C. The effect of each term in the functional (2.21) was also shown in [18] for certain numerical examples. The method was also investigated for the adaptive solution of a steady PDE.

Brackbill [17] later introduced a directional control functional to generate a mesh for the solution of unsteady magnetohydrodynamic flow in two spatial dimensions.

In [51, 52] Huang and Russell present a mesh adaption functional which can be seen as a general form of the Winslow functional (2.20). This functional, in two dimensions, is given as

$$I[\xi, \eta] = \int_\Omega \left(\nabla\xi^T G_1^{-1} \nabla\xi + \nabla\eta^T G_2^{-1} \nabla\eta \right) \mathrm{d}\Omega, \tag{2.22}$$

where G_1 and G_2 are some given symmetric positive definite matrices which are referred to as the monitor matrices. The functional gives the associated Euler-Lagrange equations

$$\nabla \cdot \left(G_1^{-1} \nabla \xi \right) = 0, \tag{2.23}$$

$$\nabla \cdot \left(G_2^{-1} \nabla \eta \right) = 0. \tag{2.24}$$

(cf. (2.16), (2.17)). When the monitor matrices $G_1 = G_2$ are equal to the identity matrix then it is straightforward to see that the functional of Huang and Russell (2.22) reduces to that of Winslow. Monitor matrices are also given which cause (2.22) to reduce to Brackbill's directional control functional.

The reason for the choice of the functional is due to its close relation to harmonic maps. The Euler-Lagrange equations (2.23) and (2.24) together with appropriate boundary conditions will define a harmonic map provided that the boundary of the computational domain Ω_c is convex. In [38] Dvinsky proves that the map will be guaranteed to exist and be unique if $G_1 = G_2 = G$, say. It is unclear whether this result extends to when $G_1 \neq G_2$. Dvinsky [38] also notes that since no constraint on the monitor matrix G is required to guarantee invertibility of the harmonic map, the monitor matrices may be used purely for mesh adaption.

Azarenok [5] uses the functional

$$I\left[x, y\right] = \int_{\Omega_c} \frac{\left(x_\xi^2 + x_\eta^2\right)\,\left(1 + f_x^2\right) + \left(y_\xi^2 + y_\eta^2\right)\,\left(1 + f_y^2\right) + 2 f_x f_y \left(x_\xi y_\xi + x_\eta y_\eta\right)}{\left(x_\xi y_\eta - x_\eta y_\xi\right)\,\sqrt{1 + f_x^2 + f_y^2}}\, \mathrm{d}\Omega_c,$$

which defines a harmonic map from the computational space Ω_c to the physical space Ω. This functional tends to cluster mesh points where the gradient of the given function $f \equiv f\left(x, y\right)$ is large. In [5] this functional is used for the solution of hyperbolic conservation laws in one and two spatial dimensions and the function f is taken to be one of the solution variables.

More recently, work by Huang [47] has used local error distributions to derive a functional for generating an adaptive mesh. He states that it is unclear whether any of the methods described so far for generating an adaptive mesh will produce an optimal mesh in the

sense of minimising the actual error in the solution. Huang observes that often the local error distribution for a given discretisation can be written in the form

$$E(\mathbf{x}) = \sqrt{\mathbf{d\xi}^T \mathcal{J}^T G \mathcal{J} \mathbf{d\xi}},$$

where G is an $n \times n$ symmetric positive definite matrix, $\mathbf{d\xi} = \boldsymbol{\xi} - \boldsymbol{\xi}_c$ where $\boldsymbol{\xi}_c$ is an arbitrary point, and $\mathcal{J}$ is a Jacobian matrix of the transformation from the background to physical domain. Here n denotes the spatial dimension of the problem. Huang then states that a perfect mesh would have this error distribution equidistributed over it and therefore would have

$$A \equiv \mathcal{J}^{-1} G^{-1} \mathcal{J}^{-T} = c\, I, \tag{2.25}$$

for some constant c. It is then stated that necessary conditions for (2.25) are

$$\lambda_1 = \lambda_2 = \cdots = \lambda_{n-1} = \lambda_n \qquad \text{(Isotropy)},$$

$$\sqrt{\prod_i \lambda_i} = constant \qquad \text{(Uniformity)},$$

where λ_i for $1 \leq i \leq n$ are the eigenvalues of the matrix A, are true. By satisfying these conditions Huang shows that the functional

$$I[\boldsymbol{\xi}] = \frac{1}{2} \int_\Omega \sqrt{g} \left(\sum_i (\nabla \xi_i)^T \, G^{-1} \nabla \xi_i \right)^{\frac{n}{2}} \mathrm{d}\Omega,$$

where g is the determinant of the matrix G, can be used to control the isotropy of the mesh and the functional

$$I[\boldsymbol{\xi}] = \int_\Omega \frac{\sqrt{g}}{\left(J\sqrt{g}\right)^q} \mathrm{d}\Omega,$$

where J is the determinant of the Jacobian $\mathcal{J}$, controls the uniformity of the mesh. Here q is a constant greater than unity. A combination of these two functionals is used in practice to give emphasis to either the mesh isotropy or uniformity. In [28] Cao *et al* compare Huang's method for generating an adaptive mesh with other commonly used methods such as the Harmonic map method and Winslow's variable diffusion method.

2.2.2 Links with Equidistribution

In one dimension we considered various moving mesh equations which were derived from the equidistribution principle. The equidistribution principle in one-dimension (2.1) can also be written as the Euler-Lagrange equation for minimising the functional

$$I[\boldsymbol{\xi}] = \int_{\Omega} \frac{1}{M(\mathbf{x})} \left(\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right)^2 \mathrm{d}x. \tag{2.26}$$

However the multidimensional extension of this minimisation principle in the form (2.26) is not well defined as it would require that we have

$$\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} = M(\mathbf{x}),$$

where $M(\mathbf{x})$ is now an n by n matrix which controls the various properties of the mapping. This system cannot be solved as it is, since it is over-determined. Therefore Knupp uses the least-squares principle to determine the mapping in the following way

$$I[\boldsymbol{\xi}] = \int_{\Omega} \left|\left| \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} - M(\mathbf{x}) \right|\right|_F^2 \mathrm{d}\Omega$$

where $||\cdot||_F$ is the Frobenius norm of the matrix.

Baines in [7] considered a natural generalisation of the one dimensional equidistribution principle. The equation

$$\nabla_\zeta \cdot (M(n)\ \nabla_\zeta n) = 0, \tag{2.27}$$

where n is a co-ordinate along the direction of the gradient of the solution, ∇u, and $\zeta = (\xi, \eta)$, is employed to move the mesh points. This equation for the mesh reduces to the one dimensional equidistribution principle perpendicular to ∇u. Baines also notes that if n in equation (2.27) is replaced by either x or y the useful mesh adaption technique

$$\nabla_\zeta \cdot (M \nabla_\zeta x) = \nabla_\zeta \cdot (M \nabla_\zeta y) = 0 \tag{2.28}$$

is obtained. These equations using the arc-length monitor function is used to generate meshes adapted to given functions using either Dirichlet or Neumann boundary conditions. As with the one-dimensional approach highlighted in [7], the mesh and solution are solved in an iterative manner and it is found that the resulting mesh does not strictly equidistribute the monitor function. However, the method is found to cluster points in regions of large M and therefore produces reasonably convincing meshes.

All the approaches detailed in two dimensions so far have only been described for generating a single adaptive mesh. However, when time-dependent problems are solved, an adaptive mesh is required at each time step. There are two main ways of doing this. The first is to simply use the static mesh generator at each time level and therefore produce a series of meshes. Unfortunately this is not an efficient procedure in general and often the mesh can change quite abruptly in time. Therefore, in an attempt to produce a smoother transition of the mesh in time, Huang and Russell [51, 52] conjecture that one way to minimise a given functional I is to follow the steepest descent direction given by the first derivative of the functional. Thus the gradient flow equations

$$\begin{aligned} \frac{\partial \xi}{\partial t} &= -\frac{\partial I}{\partial \xi} \\ \frac{\partial \eta}{\partial t} &= -\frac{\partial I}{\partial \eta} \end{aligned}$$

are used since the limit as $t \to \infty$ is the Euler-Lagrange equations for minimising the given functional. However in practice these parabolic gradient flow equations are regularised by introducing the relaxation parameter τ so that the Euler-Lagrange equations for the functional are satisfied when $\tau \to 0$. These modified gradient flow equations are given by

$$\begin{aligned} \frac{\partial \xi}{\partial t} &= -\frac{P}{\tau}\frac{\partial I}{\partial \xi} \\ \frac{\partial \eta}{\partial t} &= -\frac{P}{\tau}\frac{\partial I}{\partial \eta} \end{aligned}$$

where P is a preferred direction for the descent. It was highlighted in [52] that when the functional being minimised is the equidistribution principle given by (2.26), the method reduces to the one dimensional MMPDE approach. For particular choices of the function

P MMPDEs 3-6 can be obtained. For example MMPDE5 from the previous section can be obtained if we choose $P = M^2 \left(\frac{\partial \xi}{\partial x}\right)^{-2}$.

With suitable choices of the function P, which is given in terms of the determinants of the matrices G_1 and G_2, the multidimensional MMPDEs become

$$\begin{aligned} \frac{\partial \xi}{\partial t} &= -\frac{1}{\tau\sqrt{g_1}}\frac{\partial I}{\partial \xi}, \\ \frac{\partial \eta}{\partial t} &= -\frac{1}{\tau\sqrt{g_2}}\frac{\partial I}{\partial \eta}, \end{aligned}$$

where $g_1 = |G_1|$ and $g_2 = |G_2|$. If the functional I is given as (2.22) then the MMPDEs are given as

$$\begin{aligned} \frac{\partial \xi}{\partial t} &= -\frac{1}{\tau\sqrt{g_1}}\nabla \cdot \left(G_1^{-1}\nabla \xi\right), \\ \frac{\partial \eta}{\partial t} &= -\frac{1}{\tau\sqrt{g_2}}\nabla \cdot \left(G_2^{-1}\nabla \eta\right). \end{aligned} \tag{2.29}$$

In practice it often simpler to solve this MMPDE by interchanging the dependent and independent variables as was done with Winslow's method described earlier in this section. On doing this the PDEs given by (2.29) become

$$\begin{aligned} \frac{\partial \mathbf{x}}{\partial t} = &-\frac{\mathbf{x}_\xi}{\tau\sqrt{g_1}\,J}\left\{+\frac{\partial}{\partial \xi}\left[\frac{1}{J\,g_1}\left(\mathbf{x}_\eta^T G_1 \mathbf{x}_\eta\right)\right] - \frac{\partial}{\partial \eta}\left[\frac{1}{J\,g_1}\left(\mathbf{x}_\xi^T G_1 \mathbf{x}_\eta\right)\right]\right\} \\ &-\frac{\mathbf{x}_\eta}{\tau\sqrt{g_2}\,J}\left\{-\frac{\partial}{\partial \xi}\left[\frac{1}{J\,g_2}\left(\mathbf{x}_\eta^T G_2 \mathbf{x}_\xi\right)\right] + \frac{\partial}{\partial \eta}\left[\frac{1}{J\,g_2}\left(\mathbf{x}_\xi^T G_2 \mathbf{x}_\xi\right)\right]\right\}. \end{aligned} \tag{2.30}$$

Suitable boundary conditions for this MMPDE are also required to define the co-ordinate mapping. There are many ways in which the boundary conditions may be prescribed. Firstly the MMPDE may be supplemented with Dirichlet conditions where the mesh points on the boundary are held fixed. When all the mesh adaption is occurring inside the domain this technique can produce satisfactory results. However when the adaption is occurring at the boundary it is necessary for the mesh points to be allowed to move along it. Secondly, Neumann boundary conditions could be used, but this type of condition has been found to be not very robust and can lead to a non-smooth mesh [46]. Thirdly, a lower dimensional moving mesh equation could be solved along the

boundary to prescribe the required mesh boundary concentration. So for example, for the two-dimensional mesh adaption a one-dimensional moving mesh equation could be solved along the boundary. Several formulations of this boundary equation have been proposed by Huang [46] and Huang and Russell [53]. One commonly used equation for the boundary mesh distribution is

$$\frac{\partial s}{\partial t} = \frac{1}{\tau}\left(M\frac{\partial s}{\partial \chi}\right)^{-2}\frac{\partial}{\partial \chi}\left(M\frac{\partial s}{\partial \chi}\right),$$

where M is the matrix G projected on to the boundary of the domain, χ is the arc-length variable and s is the arc-length along the boundary segment.

In [52] Huang and Russell use this MMPDE approach to solve partial differential equations of the form

$$u_t = f\left(t, x, y, u, u_x, u_y, u_{xx}, u_{yy}\right). \tag{2.31}$$

The mesh PDE (2.30) and the physical PDE (2.31) have to be solved either simultaneously or alternately. In one dimension moving mesh methods often comprise of updating the solution and mesh simultaneously; however this process is less straightforward in two dimensions and therefore Huang and Russell use the Method Of Lines (MOL) approach of integrating the mesh and solution alternately in time. The procedure for updating the solution is now illustrated.

1. Given the solution u^n to the PDE, the mesh $\mathbf{x}^n$ and the time-step Δt^n at t^n, compute the monitor matrices $G_1 = G_1\left(t^n, \mathbf{x}^n, u^n\right)$ and $G_2 = G_2\left(t^n, \mathbf{x}^n, u^n\right)$.

2. Compute the new mesh $\mathbf{x}^{n+1}$ by integrating the MMPDE given by (2.30) forward in time from $t = t^n$ to $t = t^n + \Delta t^n$ using $\mathbf{x}^n$ as an initial mesh and keeping G_1^n and G_2^n constant in time.

3. Compute the physical solution u^{n+1} by integrating the physical PDE forward in time from $t = t^n$ to $t = t^n + \Delta t^n$ using

$$\mathbf{x} = \mathbf{x}^n + \frac{t - t^n}{\Delta t^n}\left(\mathbf{x}^{n+1} - \mathbf{x}^n\right)$$

and

$$\dot{\mathbf{x}} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t^n}.$$

4. Choose Δt^{n+1} as the next time-step size predicted during the physical PDE integration.

This solution procedure is then used to generate a mesh around a NACA0012 airfoil configuration and also to solve Burgers equation and a model problem in combustion theory.

Up until now nothing had been said about the choice of the monitor matrices. The use of a variety of monitor matrices were discussed in [26] by Cao *et al* for the functional (2.22). They consider the case when the monitor matrices are identical, i.e. $G = G_1 = G_2$ and use an eigendecomposition of G given as

$$G = \lambda_1 \, v_1 \, v_1^T + \lambda_2 \, v_2 \, v_2^T,$$

where v_1 and v_2 are the normalised eigenvectors corresponding to the eigenvalues λ_1 and λ_2. They find, using this eigendecomposition and a Greens function analysis for elliptic PDEs, that the eigenvectors and eigenvalues determine the mesh directions and strengths of the mesh concentration. In particular the mesh concentrates points in regions of the domain where the eigenvalues change rapidly. Once the eigenvector v_1 has been suitably chosen the second eigenvector is normally chosen to be orthogonal to v_1, i.e. $v_2 = v_1^{\perp}$. The eigenvalues λ_1 and λ_2 are then chosen to have variations in the v_1 and $v_1^{\perp}$ directions respectively. In this sense the mesh adaption in the directions of the eigenvectors is determined by the ratio of the two eigenvalues $\frac{\lambda_1}{\lambda_2}$. Generally speaking, the smaller this ratio the larger the adaption in the v_1 direction. Many solution dependent choices of the monitor matrix are discussed in [26]; for example one such choice is to have

$$v_1 = \frac{\nabla u}{|\nabla u|}, \quad v_2 = v_1^{\perp}, \quad \lambda_1 = \sqrt{1 + |\nabla u|^2}$$

which will cause mesh adaption where there are large variations in the solution u. A choice for the second eigenvalue also has to be made. One choice is to let $\lambda_1 = \lambda_2$ which

will cause $G = \lambda_1 I_2$, where I_2 is a 2×2 identity matrix. This choice is equivalent to using Winslow's variable diffusion functional with a particular choice of the weighting function. A second choice is $\lambda_2 = \frac{1}{\lambda_1}$ which will result in the Harmonic map method [38]. A third choice for the second eigenvalue is $\lambda_2 = 1$ and this makes the monitor matrix $G = \left(I_2 + \nabla u \, \nabla u^T\right)^{\frac{1}{2}}$ which can be seen as a multidimensional extension of the familiar arc-length monitor function.

In [12] Beckett, Mackenzie and Robertson use the monitor matrix

$$G = \left(1 + \frac{\mu_1}{\sqrt{\mu_2 \, (\mathrm{x} - \mathrm{x}_*)^2 + 1}}\right) I_2,$$

for the adaptive solution of Stefan problems, to cluster mesh points around an evolving phase change interface. Here x_* is a point on a phase front which is closest to x, and μ_1 and μ_2 are constants.

Tang and Tang [88] solve hyperbolic conservation laws in one and two spatial dimensions using a variational approach to generate the mesh. Their approach, based on work by Li, Tang and Zhang [62], is to solve the PDE on a static mesh using an existing PDE solver and then to redistribute the mesh points to obtain a better approximation. The mesh redistribution is performed iteratively and the solution is interpolated between meshes using a conservative method. Tang and Tang observe that the mesh equations (2.30) are more complicated than the Euler-Lagrange equations given by (2.29) and the solution of these equations requires more computational effort. Therefore they use the functional of Ceniceros and Hou [31]

$$I\left[x, y\right] = \frac{1}{2} \int_{\Omega_c} \left(\nabla_\zeta x^T \, G_1 \, \nabla_\zeta x + \nabla_\zeta y^T \, G_2 \, \nabla_\zeta y\right) \mathrm{d}\Omega_c$$

to generate an adaptive mesh where $\nabla_\zeta = \left(\frac{\partial}{\partial \xi}, \frac{\partial}{\partial \eta}\right)^T$. Notice that if $G = M \, I_2$ then the Euler-Lagrange equations for this functional are the same as the equations (2.28) derived by Baines [7]. Similar monitor functions to those used by Stockie, Mackenzie and Russell [86] are employed to produce adaptive meshes for the one dimensional Euler equations. However for two-dimensional calculations the monitor function $G = \omega \, I_2$ is used where

$$\omega = \sqrt{1 + \alpha \left(\rho_\xi^2 + \rho_\eta^2\right)}.$$

Here ρ is the density of the gas being modelled and α is a constant. It should be noted that when the domain Ω_c is non-convex the grid generation technique of Ceniceros and Hou [31] will not in general be able to produce a suitable mesh in the physical domain.

The method of Tang and Tang [88] is in contrast to Arbitrary Lagrangian-Eulerian (ALE) methods, where instead of interpolating the solution between adaptive meshes and then using a standard PDE solver, the PDE is solved in a moving reference frame. Before describing the ALE method we will firstly introduce the Eulerian and Lagrangian methods.

2.2.3 Links with Fluid Dynamics

The classical Lagrangian method is a technique for moving the computational mesh in fluid problems in which the velocity of the nodes is taken to be equal to the actual fluid velocity being modelled. Therefore we have that the mesh velocity is given by

$$\dot{\mathbf{x}} = \mathbf{v},$$

where $\mathbf{v}$ is the velocity of the fluid. This method has many advantages, but unfortunately has some serious downsides. The Lagrangian method maintains good resolution of compressions and expansions in the solution and also preserves multi-material interfaces in the fluid well. However, in higher than one spatial dimension the Lagrangian mesh can quickly become singular for computations of compressible flow. Meshes become more and more skewed when vorticity and shear is present in the flow and can become singular in finite time.

In the Eulerian method the mesh is held fixed in time and the fluid which is being modelled moves through the mesh. Thus the mesh velocity is taken to be $\dot{\mathbf{x}} = \mathbf{0}$. Due to this, Eulerian meshes do not tangle and become singular. But one problem with numerical solutions calculated on Eulerian meshes is that they suffer from excessive diffusion, and also material interfaces are hard to maintain.

The aim of the Arbitrary Lagrangian-Eulerian (ALE) method of Hirt, Amsden and Cook [45] is to try to combine the best components of the Eulerian and Lagrangian methods. The main idea behind the ALE methodology is that the mesh movement does not have to be constrained to be either Eulerian or Lagrangian and that it can be chosen arbitrarily to improve accuracy and robustness of the solution procedure. Once the mesh movement has been chosen the partial differential equation is solved in a moving frame of reference. For example the integral conservation of mass equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \rho \,\mathrm{d}\Omega + \int_{\Omega(t)} \nabla \cdot \rho\, \mathbf{v} \,\mathrm{d}\Omega = 0$$

in a general reference frame moving with velocity $\dot{\mathbf{x}}$ becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \rho \,\mathrm{d}\Omega + \int_{\Omega(t)} \nabla \cdot \rho\, (\mathbf{v} - \dot{\mathbf{x}}) \,\mathrm{d}\Omega = 0.$$

where ρ is the density of the fluid being solved. ALE methods have been successfully used to solve a variety of applications.

In ALE simulations the mesh velocity is often constrained to satisfy the Geometric Conservation Law (GCL). Thomas and Lombard [91] considered the solution of conservation laws on moving grids and conjectured that oscillations and instabilities in numerical solutions can occur if geometric quantities, such as cell volumes and edge velocities, are not approximated in a consistant manner. They argued that the the numerical discretisation of these geometric quantities should satisfy a relevant conservation law and that this conservation law comes from the consideration that a constant flow field should not be affected by the motion of the mesh. When a constant flow field is substituted into the ALE form of a conservation law the condition

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathrm{d}\Omega = \int_{\Omega(t)} \nabla \cdot \dot{\mathbf{x}} \,\mathrm{d}\Omega$$

is obtained. This is called the GCL since it deals purely with the geometric properties of the spatial region $\Omega(t)$ and is often referred to as the space conservation law [35]. The GCL has been shown to be a necessary but not sufficient condition to ensure good numerical solutions on dynamically moving meshes [39].

An interesting method has recently been derived by Cao *et al* [27] which can be regarded as a generalisation of the GCL. This method is therefore aptly named the GCL method and has a close relationship with the Deformation map method which will be described shortly. The GCL method will be discussed in greater detail in the next chapter.

The deformation map method of Liao and Anderson [64] is a technique for generating an adaptive numerical mesh which can be used for grid generation or for the adaptive solution of partial differential equations. The method is based on a theorem in differential geometry, due to Moser [70] and Dacorogna and Moser [34]. The method is derived from the study of volume elements of a compact Riemannian manifold to prove existence of $\mathcal{C}^1$ diffeomorphisms with specified Jacobian. The mesh velocity is calculated by solving the equation

$$\dot{\mathrm{x}} = \frac{1}{M}\nabla\phi \tag{2.32}$$

in some spatial domain. Here ϕ is a velocity potential coming from the solution of the equation

$$\nabla^2\phi = -\frac{\partial M}{\partial t} \tag{2.33}$$

and M is the usual monitor function. A suitable boundary condition for the problem is also required and is given as

$$\frac{\partial\phi}{\partial n} = 0, \tag{2.34}$$

where n is the outward normal to the boundary of the domain. In this method the monitor function may be chosen so as to control the Jacobian of the transformation. In [82, 67] the monitor function is chosen by defining a distance function d which measures the distance to some feature of the numerical solution. The monitor function is then explicitly given as a function of d. One drawback to this technique for choosing the monitor function is that some sort of *a priori* knowledge of the solution is required. This technique for defining the monitor function has more recently been used by Liao *et*

al [65] in conjunction with a technique based on level-sets [83] for the solution of Stefan problems.

In [66] the steady Euler equations were solved using a monitor function designed for the solution of steady state equations. This is done by defining a monitor function $M(\mathbf{x})$ which is dependent only on the spatial variable and targets how the mesh should be distributed at steady state. A pseudo-time-dependent monitor function is then given in terms of the spatially dependent monitor function $M(\mathbf{x})$ by

$$M(\mathbf{x}, t) = (1 - t) + t\, M(\mathbf{x})$$

for $0 < t < 1$. The moving mesh equations are then integrated forward in pseudo-time until $t = 1$ to give the steady state mesh. The spatial dependent monitor function in [66] was defined to be

$$M(\mathbf{x}) = 1 + |\nabla p|^2$$

where p is the pressure of the gas being modelled.

2.2.4 Moving Finite Elements

Another velocity-based method is that of Moving Finite Elements (MFE) which was developed by Miller and Miller [69, 68]. The MFE method was originally used to approximate solutions of time-dependent partial differential equations of the form

$$u_t = \mathcal{L}u$$

which had steep moving fronts. The main concept behind MFE was to minimise the L_2-norm of the residual of the differential form of the moving PDE (2.11). This can be written in the form

$$\min_{\dot{\mathrm{X}}, \frac{\mathrm{D}U}{\mathrm{D}t}} \int_\Omega \left(\frac{\mathrm{D}U}{\mathrm{D}t} - \dot{\mathrm{X}} \cdot \nabla U - \mathcal{L}U \right)^2 w \, \mathrm{d}\Omega, \tag{2.35}$$

where w is a weight function and U and $\dot{\mathrm{X}}$ are linear finite element approximations given as

$$U = \sum_j U_j \, w_j \qquad \dot{\mathrm{X}} = \sum_j \dot{\mathrm{X}}_j \, w_j$$

and here w_j are the standard nodal finite element basis functions. In the original version of MFE the weight function in (2.35) was taken to be $w \equiv 1$. However this weight function may also be chosen to be solution dependent, so it may depend on the solution u and or its derivatives. One common choice for the weight function is $w \equiv \frac{1}{\sqrt{(1+|\nabla U|^2)}}$. This choice of the weight function leads to Gradient Weighted Moving Finite Elements (GWMFE) [29, 30]. The MFE minimisation is done by deriving the normal equations

$$\int_\Omega \left(\frac{\mathrm{D}U}{\mathrm{D}t} - \dot{\mathrm{X}} \cdot \nabla U - \mathcal{L}U \right) w \, \mathrm{d}\Omega = 0 \tag{2.36}$$

$$\int_\Omega \left(\frac{\mathrm{D}U}{\mathrm{D}t} - \dot{\mathrm{X}} \cdot \nabla U - \mathcal{L}U \right) \nabla U \, w \, \mathrm{d}\Omega = 0 \tag{2.37}$$

for the minimisation over $\frac{\mathrm{D}U}{\mathrm{D}t}$ and $\dot{\mathrm{X}}$ respectively. It is easily seen that these are in fact weak forms of the PDE being solved. In certain circumstances the mesh velocity obtained from the MFE method can be shown to be approximately Lagrangian [6]. The main feature of the MFE method is that the movement of the mesh tries to minimise the weighted L_2-norm of the residual of the discrete PDE being solved. One of the problems with the MFE method is that the matrices which are obtained from (2.36) and (2.37) can become singular. Therefore for some problems careful regularisation of the method is required to obtain satisfactory results. This regularisation process normally consists of adding penalty terms onto the residual norm, and therefore the normal equations, to prevent them becoming singular.

For most problems the techniques described so far in this chapter can lead to good results for the numerical solution to partial differential equations on adaptively moving meshes. However, for some problems unless extra care is taken these methods can lead to poor results and often spurious solutions [23]. Therefore there has been recent interest

in geometric methods which are designed to inherit some or all of the structure of the system being solved.

2.3 Geometric Integration

The aim of geometric integration is to devise discrete approximations to continuous systems of differential equations which preserve key features of the system. An excellent review article discussing geometric integration and its applications can be found in [25].

There has been a lot of research interest in the use of moving mesh methods for the solution of partial differential equations which exhibit scale invariant behaviour and self-similar solutions. One particular PDE which has been solved using this type of geometric integration approach is the Porous Medium Equation (PME). In [21] Budd and Collins construct a moving mesh discretisation of the PME. The aim of the work is to produce discrete solutions to the PME which have the same asymptotic behaviour as solutions to the underlying continuous problem. This is done by ensuring that the numerical scheme inherits the properties of the continuous equation. In particular the scheme is derived to have the same conservation properties and also the same invariants. In their method the PME is semi-discretised using a finite difference MOL technique and then the computational mesh is moved to preserve discrete conservation of mass in the numerical solution. The resulting discrete system is then shown to have discrete invariants which are analogous to the invariants in the continuous problem. They are also able to show that the discretisation error is independent of time, which is in contrast to non-invariant moving mesh techniques for which the errors will grow as the spatial domain expands and the spatial step size increases. The PME is again considered in [22] by Budd *et al.* The notion of scale invariance and self-similar solutions will be introduced later in chapter 4 with specific reference to the porous medium equation.

The PME is also solved by Budd and Piggott [24, 25] again using a moving mesh technique. They argue that the monitor function used in the moving mesh equation MMPDE6 should be in some sense scale invariant when applied to PDEs exhibiting scale invariant behaviour. Therefore they use the so called mass monitor function

$$M = u.$$

This monitor function also ensures that the mesh moves to discretely conserve the mass of the solution.

Budd, Huang and Russell in [23, 25] consider the adaptive solution of PDEs of the form

$$u_t = u_{xx} + f(u), \tag{2.38}$$

with particular reference to the cases when $f(u) = u^p$ where $p > 1$ is a parameter and $f(u) = e^u$. It can be shown that when the initial condition $u_0(x)$ for these problems is "sufficiently large" the solution to the PDE can blow-up in finite time. This type of behaviour can be hard to approximate using traditional fixed grid methods since when the spatial size of the singularity decreases to below the size of the fixed spatial step size the accuracy will deteriorate significantly and in certain circumstances the blow-up feature of the solution may be completely missed by the method. Budd *et al* construct a moving mesh method using MMPDE6 with the monitor function

$$M(u) = u^{p-1}$$

for the case when $f(u) = u^p$. This monitor function is chosen because it makes the mesh equation scale invariant to the scale inherent in the underlying continuous problem. By using a moving mesh method for the solution of blow-up problems Budd *et al* cluster mesh points inside the blow-up region and hence capture this type of phenomena accurately.

Budd *et al* in [20, 25] apply similar techniques used for the blow-up problem to the solution of the non-linear Schrödinger equation. The non-linear Schrödinger equation is given as

$$i\frac{\partial u}{\partial t} + \nabla^2 u + |u|^2 u = 0.$$

This equation was solved using a moving mesh method in radial co-ordinates using scale invariant techniques to define a suitable monitor function. They consider the scale invariant monitor functions $M = |u|^2$ and $M = \sqrt{\alpha|u|^4 + \beta\,|u_r|^2}$ where r is the radial co-ordinate and α and β are constants. The mesh was then moved using MMPDE6. The blow-up problem for the non-linear Schödinger equation is considered and the moving mesh calculations are found to give accurate predictions of the blow-up region.

In [13] Blake used geometric techniques to solve a variety of parabolic partial differential equations. The moving mesh method used is primarily constructed for the solution of the PME to produce a method which conserves the mass of the solution locally in a discrete sense. The moving mesh equation is discretised using a finite difference method and integrated forward in time using a Backward Differentiation Formula (BDF) method. The first monitor function that was used was the mass monitor $M = u$ due to the scale invariance of this monitor. This monitor function was found to produce good results for the PME when the solution gradient was not too large. However, the method is found to give insufficient resolution of steep moving fronts, therefore the monitor function was altered in an attempt to increase the accuracy of the method. The gradient monitor function

$$M = |u_x|$$

was chosen and led to a more accurate solution. The moving mesh method was also coupled with an h-refinement method.

Until now all of the techniques described so far, for solving PDEs with scale invariant properties, have only been solved in one spatial dimension. In Blake [13] the solution of the PME is considered in more than one spatial dimension. However the moving mesh equation was found to be a under-determined system in higher spatial dimensions and the solution technique was ill-conditioned. Therefore the method gave limited success in multi-dimensions. Blake [13, 14] also sucessfully considered the blow-up equation (2.38) and a parabolic semi-conductor problem in one-dimension.

A variety of techniques have been illustrated in this chapter for the solution of partial differential equations in both one and two spatial dimensions. Many of the ideas

introduced will be used in the following chapters for the adaptive solution of PDEs.

Chapter 3

A Moving Mesh Method

In this chapter, following Baines, Hubbard and Jimack [8, 9], we will present a novel moving mesh method for the adaptive solution of partial differential equations. The method is based on many of the ideas that have been presented in the previous chapter and in particular the concept of a monitor function. In this chapter we will follow the development of the method defined by an initial so-called conservation-of-monitor-function principle. The method will be derived in a general framework, that is in multiple spatial dimensions and for any given monitor function. From this initial conservation principle we will construct a method in which the velocities of the computational nodes in the mesh are sought and from these velocities we will be able to generate a new mesh by using a simple time-stepping algorithm.

The link between the method derived here and the deformation map method of Liao and Anderson [64] will be highlighted in this chapter and also the relationship with the Geometric Conservation Law [91] will be illustrated.

Specific examples of the moving mesh method will follow in subsequent chapters for parabolic and hyperbolic partial differential equations and for two different monitor functions.

3.1 A Conservation Principle

We begin by introducing a monitor function which is taken to be positive definite ($M > 0$). This function will be dependent on the problem that we are solving, but in general will depend on the temporal variable t, the spatial variables $\mathbf{x}$ and the solution of the

partial differential equation u and its partial derivatives with respect to $\mathbf{x}$. Therefore a general form of the monitor function in multiple spatial dimensions is taken to be

$$M \equiv M(t, \mathbf{x}, u, u_x, u_y, \dots).$$

As stated before, the monitor function reflects the difficulty in approximating the solution to the underlying partial differential equation being solved and in particular is often designed to resolve key features of the solution.

The movement of a control volume $\Omega(t)$ is now defined to satisfy the conservation-of-monitor-function principle

$$\int_{\Omega(t)} M \, \mathrm{d}\Omega = \text{constant in time.} \tag{3.1}$$

The conservation principle (3.1) states that the total integral of the monitor function should be equal to a constant for all time, as a result of which the monitor function induces a motion on the control volume such that (3.1) holds.

In general there may not be a motion such that the integral of a general monitor function will satisfy (3.1) since (3.1) may be inconsistent with the solution of the problem being solved. Therefore, so as to not restrict our choice of the monitor we need to be able to modify a given monitor so that (3.1) is true. Hence we shall scale the given monitor function in such a manner that (3.1) always holds for a modified monitor function $\tilde{M}$.

There are two ways of generating this scaled monitor function $\tilde{M}$. Firstly we can use a normalisation process so that the new transformed monitor function is given by

$$\tilde{M} = \frac{M}{\theta(t)},$$

where

$$\theta(t) = \int_{\Omega(t)} M \, \mathrm{d}\Omega. \tag{3.2}$$

(cf. equidistribution (2.1)). We then have that

$$\int_{\Omega(t)} \tilde{M}\, \mathrm{d}\Omega = \int_{\Omega(t)} \frac{M}{\theta(t)}\, \mathrm{d}\Omega = 1$$

which is certainly constant in time, for any problem. However, it is then necessary to solve an extra equation for $\theta\,(t)$.

Alternatively, we may have some a priori information about how certain key quantities, such as the solution, scale in a given problem and from this we may be able to scale the monitor function M so that (3.1) holds. One possible source of this information comes from scale invariance arguments in geometric integration and it is this approach that will be used in the solution of the porous medium equation in the next chapter. For now we will assume that the monitor function M satisfies (3.1).

We now wish to generate nodal velocities for the moving mesh method defined by the conservation principle given by (3.1). This can be done by requiring the computational nodes to move in such a manner as to satisfy the conservation principle in each computational cell. Since the conservation principle contains no explicit information about how the control volume $\Omega(t)$ is moving, we differentiate (3.1) with respect to time to extract this information, giving

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega(t)} M\, \mathrm{d}\Omega = 0. \tag{3.3}$$

Equation (3.3) can be seen as a Lagrangian form of a conservation law for some fluid in which the integral of the monitor function M (density) is conserved. We now transform (3.3) to an Eulerian form which explicitly contains the velocity using the Reynolds Transport Theorem [72]. If M is a differentiable scalar field we have that

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega(t)} M\, \mathrm{d}\Omega &= \int_{\Omega(t)} \frac{\partial M}{\partial t}\, \mathrm{d}\Omega + \oint_{\partial\Omega} M\, \dot{\mathbf{x}} \cdot \mathrm{d}\mathbf{\Gamma} \\ &= \int_{\Omega(t)} \left(\frac{\partial M}{\partial t} + \nabla \cdot (M\, \dot{\mathbf{x}}) \right) \mathrm{d}\Omega = 0 \end{aligned} \tag{3.4}$$

holds, where $\dot{\mathbf{x}}$ is the velocity of the moving frame of reference $\Omega(t)$. Equations (3.3) and (3.4) can be interpreted as forms of conservation law for a pseudo-fluid which has

the monitor function as its pseudo-density function and is moving with pseudo-velocity $\dot{\mathbf{x}}$. Since $\Omega(t)$ is arbitrary we have the strong form of (3.4) given as

$$\frac{\partial M}{\partial t} + \nabla \cdot (M\,\dot{\mathbf{x}}) = 0. \tag{3.5}$$

The term $\frac{\partial M}{\partial t}$ in (3.4) can be written in terms of the partial differential equation that we are solving. For example, if we are solving the PDE given by $u_t = \mathcal{L}u$ and M is a function of both u and ∇u then

$$\frac{\partial M}{\partial t} = \left(\frac{\partial M}{\partial u} + \frac{\partial M}{\partial\,(\nabla u)} \cdot \nabla\right) \frac{\partial u}{\partial t} = \left(\frac{\partial M}{\partial u} + \frac{\partial M}{\partial\,(\nabla u)} \cdot \nabla\right) \mathcal{L}u \tag{3.6}$$

and is therefore a known function of the space variables. Therefore, for a given M, the only unknown function in equation (3.4) or (3.5) is the velocity field $\dot{\mathbf{x}}$ and it is this function that will eventually become the mesh velocity. We also note that the term $\frac{\partial M}{\partial t}$ and hence the right hand side of (3.6) can be regarded as the "source" of the velocity field $\dot{\mathbf{x}}$.

Now, given the monitor function M the aim is to solve equation (3.4) numerically for the velocity field $\dot{\mathbf{x}}$ of the pseudo fluid. This equation was solved in one spatial dimension by Blake in [13] using finite differences for a variety of applications and in particular the porous medium equation, but the approach was found to be ill-conditioned when the number of spatial dimensions was higher than one. This is due to the fact that equation (3.4) is insufficient to determine the velocity field $\dot{\mathbf{x}}$ uniquely, as it only stipulates the divergence of the vector field. Therefore this equation is not well-posed for $\dot{\mathbf{x}}$ in spatial dimensions higher than one, so we need some other condition for the solution to be unique. In this work we have followed Cao, Huang and Russell [27] in using the Helmholtz Decomposition Theorem to prescribe the *curl* of the vector field to obtain a unique solution. Therefore we will impose the condition that

$$\nabla \times (\varpi\,\dot{\mathbf{x}}) = \nabla \times (\varpi\,\mathbf{q})\,, \tag{3.7}$$

where $\mathbf{q}$ is some given vector field and ϖ is a weight function which introduces extra control on the vector field $\dot{\mathbf{x}}$. Thus we have prescribed the velocity field $\dot{\mathbf{x}}$ to have similar

rotational properties as the prescribed vector field $\mathbf{q}$. Equation (3.7) can be rearranged to give

$$\nabla \times (\varpi(\dot{\mathbf{x}} - \mathbf{q})) = \mathbf{0}. \tag{3.8}$$

Since the *curl* in (3.8) is equal to zero there exists a differentiable scalar field ϕ (velocity potential) such that

$$\varpi\,(\dot{\mathbf{x}} - \mathbf{q}) = \nabla\phi,$$

i.e.

$$\dot{\mathbf{x}} = \mathbf{q} + \frac{1}{\varpi}\nabla\phi. \tag{3.9}$$

Then substituting equation (3.9) into equation (3.5) gives an elliptic equation for the velocity potential ϕ,

$$\nabla \cdot \left(\frac{M}{\varpi}\nabla\phi\right) = -\frac{\partial M}{\partial t} - \nabla \cdot (M\mathbf{q}) \tag{3.10}$$

A boundary condition is also required for the velocity potential ϕ. Therefore, either ϕ or $\frac{\partial\phi}{\partial n}$ will need to be given on the boundary of the region $\Omega(t)$, where n is measured along the outward normal coordinate to the boundary of the domain.

There are a variety of boundary conditions that may be given for ϕ, but perhaps the most obvious one is to require that the velocity $\dot{\mathbf{x}}$ has no component in the outward normal direction to the boundary of $\Omega(t)$ and hence none of the pseudo-fluid leaves the (moving) domain. This can be written as

$$\dot{\mathbf{x}} \cdot \mathbf{n} = 0$$

on the boundary $\partial\Omega$, where $\mathbf{n}$ is the unit outward normal, which through using (3.9) gives the condition on ϕ that

$$\frac{\partial \phi}{\partial n} = -\varpi\, \mathbf{q} \cdot \mathbf{n}. \tag{3.11}$$

An alternative approach can be obtained if instead of seeking the velocity potential ϕ of the velocity field $\dot{\mathbf{x}}$ the streamlines of the velocity field are sought. In certain circumstances the stream function for the velocity field $\dot{\mathbf{x}}$, denoted by ψ, may be obtained. For example, consider the case when we are solving a PDE in conservative form given as

$$u_t = \nabla \cdot \mathbf{F} \tag{3.12}$$

and using the moving mesh method with the monitor function taken to be $M = u$. Then the strong form (3.5) of the velocity equation becomes

$$\nabla \cdot (u\, \dot{\mathbf{x}}) + u_t = 0.$$

We can substitute the PDE we are solving, given by (3.12), into this equation to obtain

$$\nabla \cdot (u\, \dot{\mathbf{x}} + \mathbf{F}) = 0.$$

Then a stream function ψ (in two spatial dimensions) exists such that

$$u\, \dot{\mathbf{x}} + \mathbf{F} = \left(-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x} \right).$$

This equation for the velocity field $\dot{\mathbf{x}}$ can be substituted into the curl condition (3.7) to obtain

$$\mathbf{k} \cdot \nabla \times \left(\frac{\varpi}{u} \left(-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x} \right) - \frac{\varpi}{u}\, \mathbf{F} \right) = \mathbf{k} \cdot \nabla \times (\varpi\, \mathbf{q})$$

where $\mathbf{k}$ is a unit vector in the direction perpendicular to the plane. The equation can be manipulated to give an elliptic equation for the stream function ψ given as

$$\nabla \cdot \left(\frac{\varpi}{u} \nabla \phi \right) = \mathbf{k} \cdot \nabla \times \left(\frac{\varpi}{u} \mathbf{F} + \varpi\, \mathbf{q} \right). \tag{3.13}$$

A boundary condition is also required on the stream function ψ and can easily be obtained in a similar way as for the velocity potential. Although this streamline formulation will not be used in this thesis it represents another useful way of obtaining the mesh velocity in two spatial dimensions. Note that the monitor function appears in the denominator in (3.13).

We have now derived an equation, given by (3.10), for the pseudo-fluid velocity potential from the conservation principle (3.1) and it is this equation that we will use to create a moving mesh method. The pseudo-fluid velocity $\dot{\mathbf{x}}$ will eventually become our mesh velocity which can be used to generate an adaptive mesh. So we need a numerical method to solve equation (3.10) and a way of recovering the velocity $\dot{\mathbf{x}}$. Blake in [13] used a finite difference/finite volume discretisation of (3.4) to obtain a discrete approximation to the velocity field $\dot{\mathbf{x}}$ which gave good results in one dimension. However, in two spatial dimensions the matrix systems resulting from the discretisations were found to be non-square and difficult to solve accurately without introducing instabilities. Therefore in this work we use a finite element discretisation of equation (3.10) as this will lead to well-behaved square matrix systems via assembly procedures.

However, to be able to apply a finite element formulation of the problem we will need to derive weak forms of the equations (3.1), (3.9) and (3.10) and this will be done in the following section.

3.2 Weak Formulations

We now want to derive weak formulations of the equations in section 3.1, as eventually finite elements will be used to solve these equations to obtain a moving numerical method. A weak form of the conservation principle (3.1) is introduced by definition as

$$\int_{\Omega(t)} w\, M \,\mathrm{d}\Omega = \text{constant in time}, \tag{3.14}$$

where w is a test function which is continuous and once differentiable. (It is assumed that M is suitably normalised to allow (3.14) to hold.) Following the same procedure as in the previous section we differentiate (3.14) with respect to time to extract information about how the control volume $\Omega(t)$ is moving, thus we have that

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} w\,M\,\mathrm{d}\Omega = 0.$$

Again we can use the Reynolds Transport Theorem to change from a Lagrangian description to an Eulerian one. In this way we obtain the equation (cf. (3.4))

$$\int_{\Omega(t)} \left(\frac{\partial}{\partial t}\left(w\,M\right) + \nabla\cdot\left(w\,M\,\dot{\mathbf{x}}\right)\right)\,\mathrm{d}\Omega = 0.$$

The test function w now sits inside the partial differentials in the above equation, but we can use the product rule of differentiation to produce

$$\int_{\Omega(t)} \left[w\left(\frac{\partial M}{\partial t} + \nabla\cdot\left(M\dot{\mathbf{x}}\right)\right) + M\left(\frac{\partial w}{\partial t} + \dot{\mathbf{x}}\cdot\nabla w\right)\right]\,\mathrm{d}\Omega = 0. \tag{3.15}$$

Now, assuming that the test function w is moving with the control volume $\Omega(t)$ and therefore travelling with velocity $\dot{\mathbf{x}}$, the test function will satisfy the advection equation

$$\frac{\mathrm{D}w}{\mathrm{D}t} = \frac{\partial w}{\partial t} + \dot{\mathbf{x}}\cdot\nabla w = 0. \tag{3.16}$$

Using (3.16) in equation (3.15) we obtain the weak form of equation (3.4) as

$$\int_{\Omega(t)} w\left(\frac{\partial M}{\partial t} + \nabla\cdot\left(M\dot{\mathbf{x}}\right)\right)\,\mathrm{d}\Omega = 0.$$

We can also use the curl condition (3.9) that was derived in the previous section to finally obtain the weak form analogue of equation (3.10),

$$\int_{\Omega(t)} w\,\nabla\cdot\left(\frac{M}{\varpi}\nabla\phi\right)\,\mathrm{d}\Omega = -\int_{\Omega(t)} w\,\left(\frac{\partial M}{\partial t} + \nabla\cdot\left(M\mathbf{q}\right)\right)\,\mathrm{d}\Omega. \tag{3.17}$$

We shall also require a weak way of recovering the velocity $\dot{\mathbf{x}}$ from the velocity potential ϕ and the velocity field $\mathbf{q}$ in (3.9). We choose to do this by enforcing the condition (3.9) in the obvious weak sense (equivalent to a least squares minimisation)

$$\int_{\Omega(t)} w\,\dot{\mathbf{x}}\,\mathrm{d}\Omega = \int_{\Omega(t)} w\,\left(\mathbf{q} + \frac{1}{\varpi}\nabla\phi\right)\,\mathrm{d}\Omega. \tag{3.18}$$

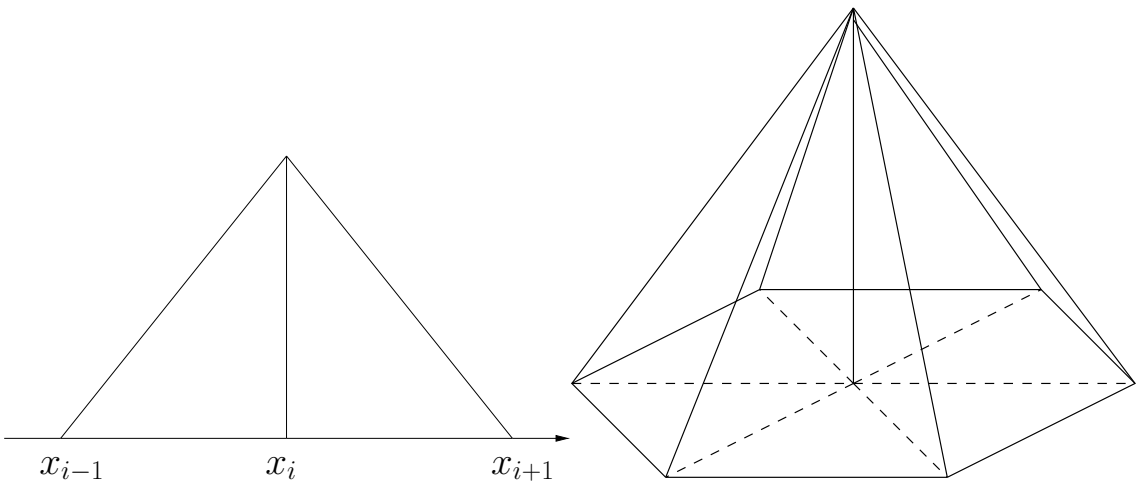

Figure 3.1: Linear finite element basis functions in 1D (left) and in 2D (right).

Now we have the weak forms (3.14), (3.17) and (3.18) of our proposed moving mesh method and we are able to introduce finite element discretisations of these equations to obtain a working numerical method. This discretisation procedure will be described in the following section.

3.3 Finite Element Approximations

Now that we have weak forms of equations (3.1), (3.10) and (3.9) given by (3.14), (3.17) and (3.18) respectively we are able to solve them using a finite element discretisation. We introduce finite element basis functions w_i for $i = 1, \ldots, N$ which in this work will be defined to be either linear hat functions defined on intervals in one dimension or linear pyramid functions defined on triangular cells in two dimensions, both forming a partition of unity, that is

$$\sum_{i=1}^{N} w_i = 1.$$

Typical basis functions are shown in figure 3.1.

We also introduce finite element approximations to the variables $\dot{\mathbf{x}}$, ϕ and u, given as $\dot{\mathbf{X}}$, Φ and U, respectively. These approximations are taken to be

$$\dot{\mathbf{X}} = \sum_{j=1}^{N} \dot{\mathbf{X}}_j \, w_j$$

$$\Phi = \sum_{j=1}^{N} \Phi_j \, w_j \tag{3.19}$$

$$U = \sum_{j=1}^{N} U_j \, w_j$$

where $\dot{\mathbf{X}}_j$, Φ_j and U_j are the coefficients corresponding to the same basis function w_j. Thus substituting these approximations into the weak form of the distributed conservation of monitor function equation (3.14) and letting the test function $w \approx w_i$ we obtain

$$\int_{\Omega_i(t)} w_i \, M \, \mathrm{d}\Omega = \theta_i, \tag{3.20}$$

where θ_i is a constant defined by (3.20) and is determined by the initial mesh and solution. Notice that since the basis functions are non-zero on a patch of computational cells, denoted by $\Omega_i(t)$, we need only integrate over this patch.

After introducing the same approximations and integrating by parts, the velocity potential equation (3.17) becomes

$$\oint_{\partial\Omega_i(t)} w_i \, \frac{M}{\varpi} \, \frac{\partial \Phi}{\partial \mathbf{n}} \cdot \, \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \nabla \Phi \, \frac{M}{\varpi} \, \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, \left(\frac{\partial M}{\partial t} + \nabla \cdot (M \mathbf{q}) \right) \, \mathrm{d}\Omega, \tag{3.21}$$

where $\partial\Omega_i(t)$ is the boundary of the patch of computational cells $\Omega_i(t)$. (For internal nodes the line integral vanishes.) The approximations given by (3.19) can now be substituted into the right hand side of this equation to give

$$\sum_{j=1}^{N} \Phi_i \, \left(\oint_{\partial\Omega_i(t)} w_i \, \frac{M}{\varpi} \, \frac{\partial w_j}{\partial \mathbf{n}} \cdot \, \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \nabla w_j \, \frac{M}{\varpi} \, \mathrm{d}\Omega \right).$$

The discretisation of (3.21) then leads to the matrix system

$$K \underset{\sim}{\Phi} = \underset{\sim}{f},$$

where the entries of K are given by

$$K_{ij} = \oint_{\partial\Omega_i(t)} w_i \, \frac{M}{\varpi} \, \frac{\partial w_j}{\partial \mathbf{n}} \cdot \, \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \nabla w_j \, \frac{M}{\varpi} \, \mathrm{d}\Omega$$

and

$$f_i = - \int_{\Omega_i(t)} w_i \, \left(\frac{\partial M}{\partial t} + \nabla \cdot (M\mathbf{q}) \right) \, \mathrm{d}\Omega.$$

Here K is a weighted finite element stiffness matrix. The equation (3.18) for the velocity becomes

$$\int_{\Omega_i(t)} w_i \, \dot{\mathbf{X}} \, \mathrm{d}\Omega = \int_{\Omega_i(t)} w_i \, \left(\mathbf{q} + \frac{1}{\varpi} \nabla \Phi \right) \, \mathrm{d}\Omega$$

and leads to the matrix system

$$A \, \underset{\sim}{\dot{\mathrm{X}}} = \underset{\sim}{b} \tag{3.22}$$

where

$$A_{ij} = \int_{\Omega_i(t)} w_i \, w_j \, \mathrm{d}\Omega$$

and

$$b_i = \int_{\Omega_i(t)} w_i \, \left(\mathbf{q} + \frac{1}{\varpi} \nabla \Phi \right) \, \mathrm{d}\Omega$$

Here A is the standard finite element mass matrix. Once the velocity of the mesh $\underset{\sim}{\dot{\mathrm{X}}}$ has been obtained the mesh can be advanced forward in time by using any particular time-stepping scheme. The system of equations for $\underset{\sim}{\mathrm{X}}$ has the form

$$\underset{\sim}{\dot{X}} = \underset{\sim}{F}(\underset{\sim}{X})$$

and is a set of n ordinary differential equations. In this work this system will be solved with the forward Euler time-stepping method, which has the form

$$\frac{\underset{\sim}{X}^{n+1} - \underset{\sim}{X}^{n}}{\Delta t} = \underset{\sim}{F}(\underset{\sim}{X}^{n}).$$

For certain applications the conservation-of-monitor-function principle (3.20) can be used to recover the solution U to the PDE that we are solving on the moving mesh. However this will be highlighted later for specific examples.

In the following chapters we will solve these equations using a variety of monitor functions and for different PDEs. However before we do so we will highlight some of the properties of the moving mesh method that has been derived and also illustrate the close connection of the method with the Deformation map method of Anderson and Liao.

3.4 Relationship with the Geometric Conservation Law

The geometric conservation law (GCL) was briefly introduced in chapter 2 in the context of moving meshes. The integral form of the Geometric Conservation Law (GCL) [91] of fluid dynamics, often referred to as the space conservation law [35], was given as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathrm{d}\Omega = \int_{\Omega(t)} \nabla \cdot \dot{\mathbf{x}}\, \mathrm{d}\Omega \tag{3.23}$$

and states that when considering a constant flow field the only change in the volume of the region $\Omega(t)$ should be due to the movement of its boundary. The GCL is easily derived from the Arbitrary-Lagrangian-Eulerian (ALE) form of the conservation of mass equation given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \rho\, \mathrm{d}\Omega = -\int_{\Omega(t)} \nabla \cdot \rho\,(\mathbf{v} - \dot{\mathbf{x}})\ \mathrm{d}\Omega. \tag{3.24}$$

It is straightforward to see that if a constant flow field is considered, that is, the density (ρ) and velocity ($\mathbf{v}$) of the fluid are constant, then the conservation of mass equation (3.24) reduces to the GCL given by (3.23).

When ALE numerical methods are used to solve problems on a dynamic mesh, which may be moving and deforming, it is useful to have the GCL as a constraint on the discretisation of geometric variables (such as cell volumes and edge velocities), although more recently Boffi and Gastaldi [15] have shown that the GCL is neither necessary or sufficient to guarantee stability of space-time ALE discretisations. They study the finite element approximation of the ALE form of a parabolic problem in a two-dimensional deforming domain and analyse the accuracy and stability of the approximation under several time marching schemes. The concept behind the GCL is that any ALE numerical scheme should preserve the trivial solution of a constant flow field under the motion of the mesh.

In [27] Cao, Huang and Russell derive a moving mesh method based on the geometric conservation law and is hence referred to as the GCL method. The GCL method and the method derived in the previous section are very similar, although they are derived in a very different manner.

To illustrate the connection between the geometric conservation law and the GCL method of Cao *et al* [27] we will now derive the GCL method. Consider the Jacobian determinant, J, of the transformation which takes the computational/static frame of reference Ω_c with co-ordinates $(\boldsymbol{\xi}, t)$ into the moving frame $\Omega(t)$ with co-ordinates $(\mathbf{x}, t)$. This transformation can be shown as the mapping

$$\Omega_c(\boldsymbol{\xi}, t) \rightarrow \Omega(t)(\mathbf{x}, t),$$

where the Jacobian determinant is explicitly given by

$$J = \left| \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \right|.$$

The integral form of the GCL can now be transformed into the static frame of reference Ω_c. Therefore the left hand side of (3.23) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega(t)} \mathrm{d}\Omega = \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_c} J^{-1} \, \mathrm{d}\Omega_c = \int_{\Omega_c} \frac{\mathrm{d}J^{-1}}{\mathrm{d}t} \, \mathrm{d}\Omega_c$$

where $\frac{\mathrm{d}}{\mathrm{d}t} = \frac{\partial}{\partial t} + \dot{\mathbf{x}} \cdot \nabla$, while the right hand side becomes

$$\int_{\Omega(t)} \nabla \cdot \dot{\mathbf{x}} \, \mathrm{d}\Omega = \int_{\Omega_c} (\nabla \cdot \dot{\mathbf{x}}) \, J^{-1} \, \mathrm{d}\Omega_c.$$

Therefore the GCL written in the static frame of reference Ω_c becomes

$$\int_{\Omega_c} \frac{\mathrm{d}J^{-1}}{\mathrm{d}t} \, \mathrm{d}\Omega_c = \int_{\Omega_c} (\nabla \cdot \dot{\mathbf{x}}) \, J^{-1} \, \mathrm{d}\Omega_c. \tag{3.25}$$

The GCL is sometimes written in the strong form as

$$\frac{1}{J^{-1}} \frac{\mathrm{d}J^{-1}}{\mathrm{d}t} = \nabla \cdot \dot{\mathbf{x}}. \tag{3.26}$$

Since the relationship

$$\frac{\mathrm{d}J^{-1}}{\mathrm{d}t} = -\frac{1}{J^2} \frac{\mathrm{d}J}{\mathrm{d}t}$$

holds, the GCL (3.25) can finally be written in physical co-ordinates as

$$\int_{\Omega(t)} J^{-1} \left(\frac{\partial J}{\partial t} + \nabla \cdot (J \dot{\mathbf{x}}) \right) \mathrm{d}\Omega = 0.$$

If the Jacobian determinant of the transformation from the background to physical co-ordinate is now chosen to be the monitor function M then the above equation is the same as equation (3.4) derived in the previous section, although the integrand is now weighted by the inverse of the Jacobian determinant. Conversely, if the transformation has a Jacobian equal to the monitor function M, then we have that the GCL is automatically satisfied. Also, provided that the Jacobian determinant comes from a non-singular co-ordinate transformation that is never zero we can use the differential form of the GCL method which is given as

$$\frac{\partial J}{\partial t} + \nabla \cdot (J \dot{\mathbf{x}}) = 0.$$

(cf. (3.26)). This derivation of the GCL method differs slightly from the derivation used in [27] since in their derivation they consider the inverse of the co-ordinate transformation used here. Cao, Huang and Russell also highlight the closeness of this method to

the deformation map method of Liao and Anderson [64], which was described in chapter 2 and is now reviewed here.

3.5 Relation to the Deformation Map Method

It can easily be seen that when we take $\varpi = M$ and $\mathbf{q} = \mathbf{0}$ in equations (3.9), (3.10) and (3.11) we obtain the mesh velocity equation

$$\dot{\mathbf{x}} = \frac{1}{M}\nabla\phi,$$

where the velocity potential is given by

$$\nabla^2\phi = -\frac{\partial M}{\partial t}$$

and is subject to the boundary condition

$$\frac{\partial\phi}{\partial\mathbf{n}} = 0.$$

These equations are the same as the deformation map equations given by (2.32), (2.33) and (2.34) in chapter 2. It is possible to show that the choice of ϖ and $\mathbf{q}$ to produce the deformation map method are probably not the best. This is due to the fact that the deformation map method will not produce an irrotational mesh in general and therefore meshes generated by this method may become highly skewed. This can be shown simply by considering the curl condition (3.7) with the choices $\varpi = M$ and $\mathbf{q} = \mathbf{0}$. Then we find that

$$\nabla \times (M\,\dot{\mathbf{x}}) = \mathbf{0},$$

which upon rearrangement implies that

$$\nabla \times \dot{\mathbf{x}} = -\frac{1}{M}\,\nabla M \times \dot{\mathbf{x}}.$$

Therefore the mesh velocity $\dot{\mathbf{x}}$ is not irrotational in general.

3.6 Summary

In this chapter we have derived a moving mesh method based on the use of monitor functions. Based on the previous experience of others we have decided to solve the resulting moving mesh equation with finite elements. Therefore we have derived weak forms of the moving mesh equations and then considered the matrix forms arising from a finite element discretisation of these equations. The connection of the method with both the Geometric Conservation Law and the Deformation Map method has also been highlighted in this chapter. In the next chapter we will consider the use of this method for the solution of the porous medium equation.

Chapter 4

The Porous Medium Equation

4.1 Introduction

We will begin this chapter by introducing the porous medium equation (PME) and go on to establish some of its properties. The PME is a non-linear partial differential equation which is of parabolic type and arises primarily in the study of ideal gas flowing through a porous medium. It also arises naturally as a model for many other physical phenomena such as the swarming of various insect species [73], the spreading of thin liquids under gravity [19] and radiative heat transfer [11, 56].

In multi-dimensions the porous medium equation is given as

$$u_t = \nabla \cdot (u^m \nabla u)\,, \tag{4.1}$$

where $u = u(\mathbf{x}, t)$ is a scalar function depending on the spatial and temporal variables $\mathbf{x}$ and t, respectively and m is a constant, which is usually taken to be a positive integer.

The porous medium equation may be derived from considering the diffusion of gas through a porous medium under the action of Darcy's law relating the velocity to the pressure gradient. The flow of the gas is characterised in terms of the variables density (u), pressure (p) and velocity ($\mathbf{v}$). It is assumed that the gas obeys the conservation of mass equation given by

$$\rho\, u_t + \nabla \cdot (u\, \mathbf{v}) = 0, \tag{4.2}$$

where ρ is the constant porosity of the medium, and Darcy's law which is given as

$$\mu\,\mathbf{v} = -\kappa\,\nabla p. \tag{4.3}$$

Darcy's law is an empirical law for the dynamics of the flow through a porous medium. Here μ is the viscosity of the gas and κ is the permeability of the medium, both assumed to be constant. The gas is also assumed to be ideal, so the pressure is related to the density by

$$p = p_0\,u^\gamma, \tag{4.4}$$

where p_0 is the reference pressure and γ is the ratio of specific heats for the gas. If equations (4.3) and (4.4) are substituted into equation (4.2) the equation

$$u_t = c\,\nabla\cdot\left(u^\gamma \nabla u\right),$$

is obtained, where c is a constant given as

$$c = \frac{\kappa\,p_0\,\gamma}{\mu\,\rho}.$$

The constant c can be scaled out of the problem and if this is done and m is taken to be equal to γ then we obtain the PME (4.1).

There has been an extensive study of the analytical properties of the PME, most of which can be found in [4, 55, 95]. Given an initial condition $u(\mathbf{x}, 0)$, solutions to the PME conserve two important quantities, namely, mass and centre of mass. Given a solution to the PME which satisfies the conditions that both $u \to 0$ and $u^m \nabla u \to 0$ as $\mathbf{x} \to \infty$, the total mass of the solution satisfies

$$\frac{\mathrm{d}}{\mathrm{d}t}\int u\,\mathrm{d}\Omega = \int u_t\,\mathrm{d}\Omega = \int \nabla\cdot\left(u^m\,\nabla u\right)\,\mathrm{d}\Omega = \oint u^m \nabla u\cdot\mathrm{d}\mathbf{\Gamma} \to 0 \quad \text{as } \mathbf{x}\to\infty. \tag{4.5}$$

Thus the PME conserves mass. Similarly, if the centre of mass is given by

$$\bar{\mathbf{x}} = \int \mathbf{x}\, u \, \mathrm{d}\Omega$$

then

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} \int \mathbf{x}\, u \, \mathrm{d}\Omega &= \int \mathbf{x}\, u_t \, \mathrm{d}\Omega = \int \mathbf{x} \nabla \cdot (u^m \nabla u) \, \mathrm{d}\Omega. \\
&= \oint \mathbf{x}\, u^m \nabla u \cdot \mathrm{d}\mathbf{\Gamma} - \frac{1}{m+1} \int (\nabla \cdot \mathbf{x}) \, \nabla \left(u^{m+1}\right) \, \mathrm{d}\Omega \\
&= \oint \mathbf{x}\, u^m \nabla u \cdot \mathrm{d}\mathbf{\Gamma} - \frac{1}{m+1} \nabla \cdot \mathbf{x} \oint \left(u^{m+1}\right) \, \mathrm{d}\mathbf{\Gamma} \to 0 \quad \text{as } \mathbf{x} \to \infty.
\end{aligned}$$

Thus the PME also conserves the centre of mass of a given solution.

Special solutions to the PME which conserve both mass and centre of mass arise from the study of scale invariance and are referred to as similarity solutions. In the next section we will define scale invariance and how, from this, similarity solutions to the PME can be constructed.

4.2 Scale Invariance and Similarity Solutions

Scale invariance can be described as follows. Given the set of variables $(u, \mathbf{x}, t)$ which satisfy the PDE under consideration, introduce a mapping to a new system $(\hat{u}, \hat{\mathbf{x}}, \hat{t})$ given by the scaling transformation

$$\hat{u} \to \lambda^\alpha \, u, \quad \hat{\mathbf{x}} \to \lambda^\beta \, \mathbf{x} \quad \hat{t} \to \lambda \, t \tag{4.6}$$

where α and β are exponents to be found and the constant λ is arbitrary. Then the system is said to be scale invariant if the mapping from $(u, \mathbf{x}, t)$ to $(\hat{u}, \hat{\mathbf{x}}, \hat{t})$ leaves the PDE unchanged.

We will now seek the transformation of the form (4.6) which leaves the PME, in radial co-ordinates, given as

$$u_t = \frac{1}{r^{d-1}} \left(r^{d-1} \, u^m \, u_r\right)_r, \tag{4.7}$$

where r is the radial co-ordinate and d is the number of spatial dimensions, invariant. Carrying out a change of variables of the form (4.6) the radial porous medium equation (4.7) becomes

$$\lambda^{1-\alpha}\,\hat{u}_{\hat{t}} = \lambda^{2\beta-(m+1)\,\alpha}\,\frac{1}{\hat{r}^{d-1}}\left(\hat{r}^{d-1}\hat{u}^{m}\,\hat{u}_{\hat{r}}\right)_{\hat{r}}.$$

Therefore the radial PME will be scale invariant under the transformation $(u, r, t) \to (\hat{u}, \hat{r}, \hat{t})$ provided that

$$1 - \alpha = 2\beta - (m+1)\,\alpha,$$

or

$$m\,\alpha - 2\beta + 1 = 0 \tag{4.8}$$

holds. We shall also require that the conservation of mass principle should hold in the transformed co-ordinates. The conservation of mass principle in radial co-ordinates in d dimensions is given by

$$\int_0^\infty u\,r^{d-1}\,\mathrm{d}r = \text{constant} \quad \forall t.$$

Substituting the change of variables into the above conservation principle we find that

$$\lambda^{-\alpha-d\,\beta}\int_0^\infty \hat{u}\,\hat{r}^{d-1}\,\mathrm{d}\hat{r} = \text{constant} \quad \forall t$$

provided that

$$\alpha + d\,\beta = 0. \tag{4.9}$$

Therefore using the algebraic equations (4.8) and (4.9) we find that the radial porous medium equation and its conservation property are invariant under the transformation

$$\hat{u} \to \lambda^{\alpha}\,u, \quad \hat{r} \to \lambda^{\beta}\,r \quad \hat{t} \to \lambda\,t \tag{4.10}$$

if

$$\alpha = -\frac{d}{d\,m+2} \quad \text{and} \quad \beta = \frac{1}{d\,m+2}. \tag{4.11}$$

We notice that the conservation of centre of mass property is not scale invariant under these scalings, however.

We now describe self-similar solutions to the PME. Any solution of the PME must also be invariant under the transformation given by (4.10) and (4.11), and will therefore satisfy

$$u(r,t) = \lambda^{\alpha}\, u(\lambda^{\beta}\, r, \lambda\, t). \tag{4.12}$$

To find the self-similar solutions which satisfy (4.12) we look for co-ordinates which are invariant, and from the scale invariance arguments it can be seen that there are two of these quantities in our problem, which are

$$f = \frac{u}{t^{\alpha}} \quad \text{and} \quad z = \frac{r}{t^{\beta}}.$$

Therefore for self-similarity we seek a solution of the radial PME of the form

$$\frac{u(r,t)}{t^{\alpha}} = f\left(\frac{r}{t^{\beta}}\right).$$

Substituting this expression into (4.7) we obtain a second order ordinary differential equation for f which can be solved by using techniques found in [11] and yields the set of self-similarity solutions

$$u(r,t) = \begin{cases} \frac{1}{\lambda(t)^d}\left(1 - \left(\frac{r}{r_0\,\lambda(t)}\right)^2\right)^{\frac{1}{m}} & \text{if } |r| \le r_0\,\lambda(t), \\ 0 & \text{if } |r| > r_0\,\lambda(t), \end{cases} \tag{4.13}$$

where

$$r_0^d = \frac{Q\,\Gamma(\frac{1}{m}+\frac{d}{2}+1)}{\Gamma(\frac{d}{2})\,\Gamma(\frac{1}{m}+1)}, \quad t_0 = \frac{m\,r_0^2}{2(dm+2)} \quad \text{and} \quad \lambda(t) = \left(\frac{t}{t_0}\right)^{\frac{1}{dm+2}}.$$

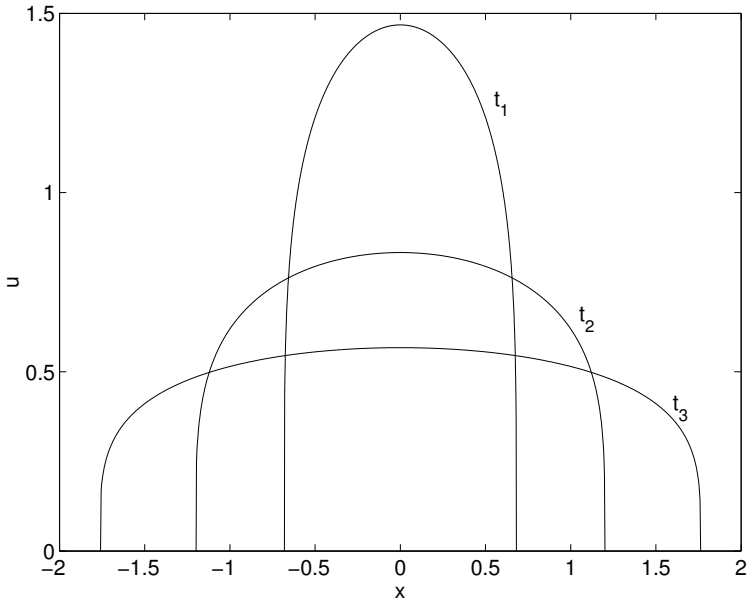

Figure 4.1: Figure depicting the radial similarity solution given by (4.13) in one spatial dimension $(d = 1)$ *at three different times* $t_1 < t_2 < t_3$ *with* $m = 4$.

Here Q denotes the total mass of the solution and Γ is the usual gamma function defined to be

$$\Gamma(x) = \int_0^\infty t^{x-1}\, e^{-t}\, \mathrm{d}t.$$

This similarity solution was originally found by Barenblatt [10] and Zel'dovich and Kompaneets [103] and was subsequently found by Pattle [75]. It is often referred to as the Barenblatt solution or the source-type solution, because as $t \to 0$ the solution tends towards a delta function with mass Q. The similarity solution can be seen at three different time instances in figure 4.1 for $d = 1$ and $m = 4$.

There are certain key properties of this solution. Firstly, the similarity solution has compact support in space and this compact support grows with time. Due to this the solution has a moving boundary which has a finite speed of propagation. The speed of the moving boundary can be found easily by considering the conservation of mass principle in radial coordinates

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_0^{r_0\,\lambda(t)} u\, r^{d-1}\, \mathrm{d}r = 0. \tag{4.14}$$

Differentiating, (4.14) becomes

$$\int_0^{r_0\,\lambda(t)} \left[u_t\, r^{d-1} + \left(u\, r^{d-1}\, \dot{r}\right)_r\right]\, \mathrm{d}r = 0$$

and using the radial PME (4.7) we obtain

$$\int_0^{r_0\,\lambda(t)} \left[\left(r^{d-1}\, u^m\, u_r\right)_r + \left(u\, r^{d-1}\dot{r}\right)_r\right]\, \mathrm{d}r = 0.$$

This equation can be integrated to give

$$\left[r^{d-1}\, u^m\, u_r + u\, r^{d-1}\, \dot{r}\right]_0^{r_0\,\lambda(t)} = 0.$$

Now using the conditions that $u_r = 0$ and $\dot{r} = 0$ at $r = 0$ we obtain

$$u^m\, u_r\Big|_{r_0\,\lambda(t)} + u\, \dot{r}_f\Big|_{r_0\,\lambda(t)} = 0,$$

which upon using a limiting procedure (since $u = 0$ at $r = r_0\,\lambda(t)$) gives an explicit expression for the speed of the moving boundary $\dot{r}_f$ given by

$$\dot{r}_f = -u^{m-1}\, u_r\Big|_{r_0\,\lambda(t)}.$$

The second property of the solution to notice is that the solution gradient u_r is not continuous at the moving boundary and therefore u does not explicitly satisfy the radial porous medium equation (4.7), i.e. it is not a classical solution. Therefore the solution is seen to satisfy the PME in some generalised sense and in fact only satisfies it in a weak sense. The similarity solution is said to be a weak solution to the porous medium equation.

The set of similarity solutions (4.13) will help us in our numerical work by allowing us to compare our numerical results with an exact solution, and the scale invariance properties obtained will also be used to derive a scale invariant numerical method.

Another property of similarity solutions in general is that they are global attractors for other general solutions of the given PDE. So if we take an arbitrary solution v of the

PME which has a given mass and centre of mass, then the similarity solution u which has the same mass and centre of mass as v will be a global attractor for v in the sense that

$$t^{\frac{d}{dm+2}} ||u - v||_{L_2} \to 0 \quad \text{as} \quad t \to \infty.$$

The details of this proof can be found in [4, 95]. This attraction property of similarity solutions allows us to determine the long time asymptotic behaviour of a solution. So although we may not know what the analytic solution to the PME will be for a general given initial condition, we will be able to say something about the long time behaviour of that solution. We would therefore also like any numerical method used to solve this equation to have the property that the numerical solution tends towards the correct similarity solution.

The scale invariance property has been used by Budd *et al* [21, 22] as the driving force behind the construction of numerical methods. Budd states that by the principles of geometric integration any numerical method should accurately reproduce the qualitative dynamics of the PDE it is solving and therefore it will only be likely to do this if it is constructed in such a way that it has discrete features which are analogous to the continuous PDE being solved. By constructing a numerical method in this manner it is hoped that it would be less likely to generate spurious solutions.

In particular, if the partial differential equation being solved has inherent scales then any numerical method should also respect them. In this way, if a numerical mesh is fixed in time it will be imposing an artificially fixed spatial structure on the problem and therefore have no hope of obtaining the dynamics of the continuous problem in the numerical discretisation. Thus it will be advantageous to have a mesh which respects the length scales in the problem and one way to do this is to use an adaptive mesh which moves to preserve the spatial length scales.

In the next section we consider the application of the moving mesh method derived in chapter 3 to the solution of the porous medium equation.

4.3 A Mass Conserving Monitor Function

We now consider the numerical solution of the porous medium equation in one spatial dimension ($d = 1$) and solve it using the moving mesh method that was derived in chapter 3. The aim will be to derive a method which produces numerical solutions which have the same asymptotic behaviour as the continuous solutions. We hope that this can be done by constructing a scheme which has the same conservation and scaling properties as the PME.

The method outlined in chapter 3 gave us three equations, which were the conservation principle (3.1), the velocity potential equation (3.10) and the velocity recovery equation (3.9). Weak forms of these equations were also derived, given by (3.14), (3.17) and (3.18). Therefore in one dimension the distributed conservation of monitor function principle (3.14) becomes

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i M \, \mathrm{d}x = \theta_i \tag{4.15}$$

and the mesh velocity potential equation (3.17) is given as

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \, \frac{\partial}{\partial x}\left(M \frac{\partial \phi}{\partial x}\right) \, \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \, \frac{\partial M}{\partial t} \, \mathrm{d}x \tag{4.16}$$

for $i = 1, \ldots, N-1$.

We now need to chose a suitable monitor function for our problem. The initial choice will be $M = u$ which is motivated by the fact that the mass of the solution to the PME is conserved for all time and also by the fact that (4.15) is then scale invariant. This monitor function is also used in the work of Budd [24] and Blake [13]. From this monitor function we hope to develop a method which conserves mass in a discrete manner.

Therefore, with this choice of monitor function the equation (4.15) becomes for interior nodes

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \, U \, \mathrm{d}x = \theta_i,$$

where θ_i is determined by the initial mesh and solution. If we now expand the finite element approximation U in terms of the basis functions w_j we obtain the matrix system

$$A(\underset{\sim}{X})\,\underset{\sim}{U} = \underset{\sim}{\theta}, \tag{4.17}$$

where A is the usual finite element mass matrix with entries given by

$$A_{ij} = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\, w_j \,\mathrm{d}x.$$

We have also enforced the boundary condition that $U = 0$ on the boundary of the domain. With the same choice of monitor function the velocity potential equation (4.16) becomes

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial}{\partial x}\left(u\frac{\partial \phi}{\partial x}\right)\,\mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial u}{\partial t}\,\mathrm{d}x.$$

A weak form of the PME can be substituted into the right hand side of the velocity potential equation above to give

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial}{\partial x}\left(u\frac{\partial \phi}{\partial x}\right)\,\mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial}{\partial x}\left(u^m\frac{\partial u}{\partial x}\right)\,\mathrm{d}x.$$

To obtain a weak form of the velocity potential equation we integrate by parts to give

$$\left[w_i\, u\,\frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\, u\,\frac{\partial \phi}{\partial x}\,\mathrm{d}x = -\left[w_i\, u^m\,\frac{\partial u}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} + \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\, u^m\frac{\partial u}{\partial x}\,\mathrm{d}x.$$

The first term vanishes for interior nodes since the basis function w_i is equal to zero at $x_{i-1}(t)$ and $x_{i+1}(t)$. We will use the boundary conditions that $u = 0$ and $\phi = 0$ at the boundary of the domain to produce

$$-\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\, u\frac{\partial \phi}{\partial x}\,\mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\, u^m\frac{\partial u}{\partial x}\,\mathrm{d}x \tag{4.18}$$

for all nodes. Once the finite element approximations have been substituted in this equation and expanded in terms of the basis functions we obtain a weighted stiffness matrix system for the mesh velocity potential Φ and has the form

$$K \underset{\sim}{\Phi} = \underset{\sim}{f}$$

where the entries of the weighted stiffness matrix are given by

$$K_{ij} = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x} U \frac{\partial w_j}{\partial x} \, \mathrm{d}x.$$

The mesh velocity can then be recovered by solving the mass matrix system which results from enforcing

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \dot{X} \, \mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial \Phi}{\partial x} \, \mathrm{d}x. \tag{4.19}$$

leading to

$$A \dot{\underset{\sim}{X}} = \underset{\sim}{b}.$$

The matrices that are generated using the monitor function $M = u$ are tridiagonal and may be solved by using a direct tridiagonal solver.

The mesh will be integrated forward in time using a forward Euler time stepping method. The method is conditionally stable, but otherwise robust. However, if a standard algorithm for solving the ODE system

$$\dot{\underset{\sim}{X}} = \underset{\sim}{F}(\underset{\sim}{X})$$

given by

$$\frac{\underset{\sim}{X}^{n+1} - \underset{\sim}{X}^{n}}{\Delta t} = \underset{\sim}{F}(\underset{\sim}{X}^{n})$$

is used we will not obtain a scale invariant method in general. This is due to the fact that the local truncation error for the numerical method will have an intrinsic length scale and therefore will not be independent in time. One way of circumventing this problem and recovering a scale invariant method is to instead solve the ODE system

$$\frac{\mathrm{d}\underset{\sim}{X}}{\mathrm{d}t^{\beta}} = \beta\, t^{\beta-1}\dot{\underset{\sim}{X}} = \beta\, t^{\beta-1}\underset{\sim}{F}(\underset{\sim}{X}),$$

which leads to the forward Euler discretisation given by

$$\frac{\underset{\sim}{X}^{n+1} - \underset{\sim}{X}^{n}}{(t^{n+1})^{\beta} - (t^{n})^{\beta}} = \beta\,(t^{n})^{\beta-1}\underset{\sim}{F}(\underset{\sim}{X}^{n}). \tag{4.20}$$

The local truncation error of this discretisation is independent of time and hence scale-invariant if the new variable t^{β} satisfies

$$\left(t^{n+1}\right)^{\beta} - (t^{n})^{\beta} = c,$$

where c is a constant which is chosen through a stability condition at the initial time. Thus the size of the time-step $\Delta t^{n} = t^{n+1} - t^{n}$ is given

$$\Delta t^{n} = \left(c + (t^{n})^{\beta}\right)^{\frac{1}{\beta}} - t^{n} \tag{4.21}$$

and is a monotonically increasing function of time. Scale invariance shows that larger time-steps are allowed as time increases and the diffusion becomes weaker. However we still need to chose an appropriate initial time-step for the numerical solution. This is chosen from stability analysis and is given as

$$\Delta t^{0} \leq \frac{1}{2}\frac{\min\,(\Delta x)^{2}}{\max\, u^{m}}.$$

The subsequent time-steps are then calculated from (4.21) with $c = (\Delta t^{0} + t^{0})^{\beta} - (t^{0})^{\beta}$.

The outline of the numerical method is:

- Given an initial mesh, solution and time-step, solve the stiffness matrix system (4.18) for the mesh velocity potential Φ.
- Recover the mesh velocity $\dot{X}$ using (4.19) and time-step the mesh using the scale invariant forward Euler time stepping procedure given by (4.20).

- Obtain the solution U on the new mesh by solving the conservation of monitor principle equation (4.17).

- Repeat until the desired output time is reached.

In the next section we will numerically illustrate this adaptive method for the solution of the porous medium equation.

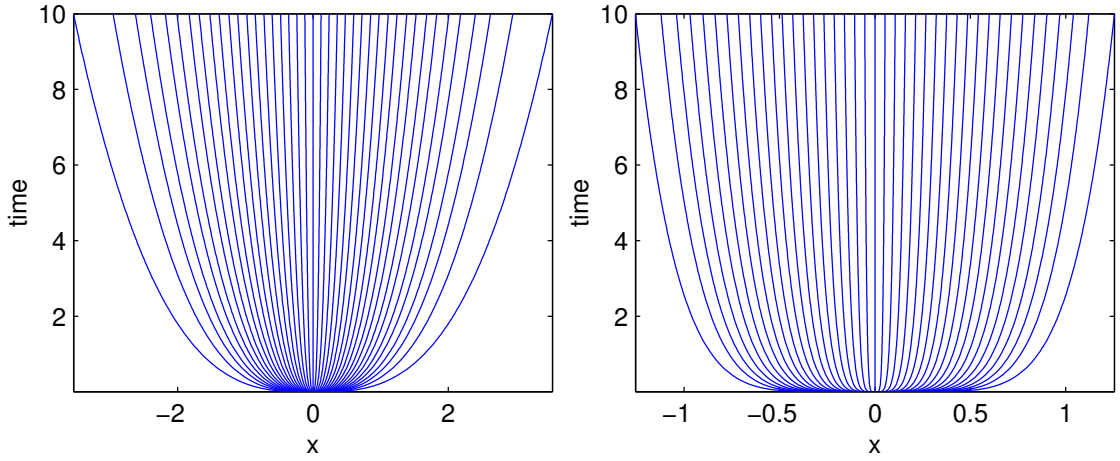

Figure 4.2: Node trajectories for $m = 1$ (left) and $m = 4$ (right).

4.4 Numerical Results I

The following section contains results obtained using the moving mesh method with the mass monitor $M = u$. Solutions to the PME have been obtained for three different meshes having 11, 21 and 41 computational nodes, respectively, and for a variety of values of the power m in (4.1). The similarity solution was used as an initial condition for the computation with $Q = 1$ and $t_0 = 0.01$. Initially the mass in each computational cell was equidistributed using the algorithm which can be found in Baines [7], although the method may also be used with different amounts of mass in each cell. The PME was then integrated forward in time to a final time of $t = 10$. Figures 4.5 and 4.6 show the solutions obtained for $m = 1$ and $m = 4$ and are compared with the exact analytical solutions. The results obtained show a good approximation to the true solution given by (4.13).

The trajectories of the mesh points are shown in figure 4.2 for $m = 1$ and for $m = 4$. These figures were produced on a mesh with 41 computational nodes. It can be seen in these figures that the mesh expands as time increases to cover the compact support of the solution to the problem.

Figures 4.3 and 4.8 show the invariant behaviour of the numerical solution. From the self-similar results obtained in section 4.2 we know that the similarity solution has two invariant quantities, namely $t^{\frac{1}{m+2}}u$ and $t^{-\frac{1}{m+2}}x$. Therefore we would hope that any numerical method approximating this similarity solution should also have a discrete

version of these invariant quantities. In figures 4.3 and 4.8 we have plotted these invariant quantities using the discrete values from our numerical approximation and they clearly suggest that the numerical method is indeed scale invariant.

The error in the height of the numerical solution at the origin ($x = 0$) and the position of the moving boundary have been calculated for the solutions and are shown in figures 4.7 and 4.9. These figures also show plots of the quantities

$$t^{\frac{1}{m+2}}||u - U|| \quad \text{and} \quad t^{-\frac{1}{m+2}}||x - X||.$$

If our numerical solution is scale invariant we would hope that the solution should respect the scalings given by (4.10) and (4.11) and therefore the errors should satisfy the equalities

$$\hat{t}^{\frac{1}{m+2}}||\hat{u} - \hat{U}|| = t^{\frac{1}{m+2}}||u - U||$$

and

$$\hat{t}^{-\frac{1}{m+2}}||\hat{x} - \hat{X}|| = t^{-\frac{1}{m+2}}||x - X||,$$

so therefore be constant in time. These quantities do indeed appear to tend to a constant value as the time of the numerical simulation increases.

The nominal error measure

$$||u - U||_{L_2}^2 = \sum_i \int_{x_{i-1}(t)}^{x_i(t)} (u - U)^2 \, \mathrm{d}x \tag{4.22}$$

has also been calculated for the numerical solutions obtained on the three different meshes and for a variety of the values m. The results are summarised in figure 4.19. It can be seen from this figure that the numerical solution is first order accurate in this norm for $m = 1$, but for higher values of m the order of the numerical method seems to be degraded. Also the precision of the numerical scheme, in the sense of (4.22), seems to lower as m increases. This is probably due to the fact that the solution has an infinite

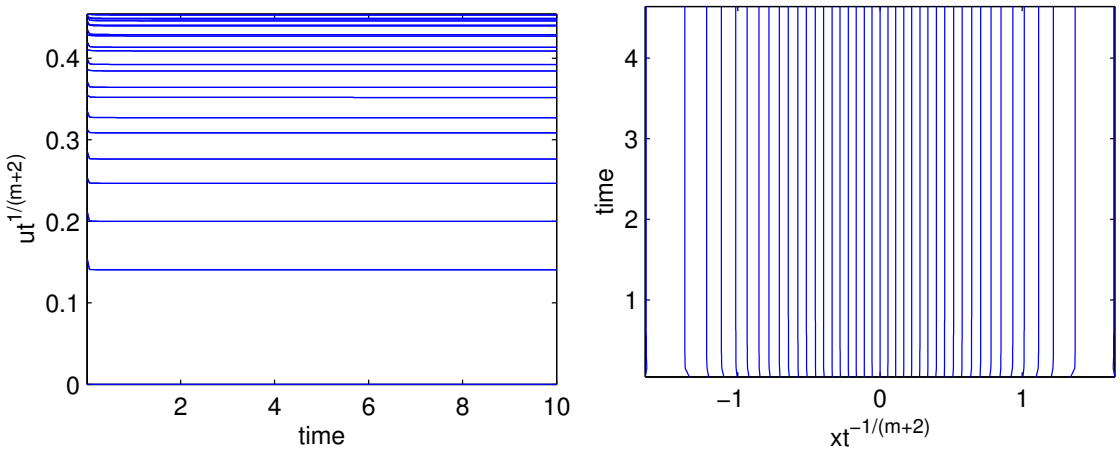

Figure 4.3: Invariant behaviour of solution (left) and mesh (right) for $m = 1$ *computed on a mesh with* 41 *nodes.*

gradient at the moving boundary for $m > 1$ and is therefore harder to resolve accurately, particularly with the monitor function $M = u$ which does not overly seek to place nodes near the moving boundary. It should also be noted that we would expect the method to be second order accurate since linear finite elements have been used. The order of accuracy can be increased by using a more suitable projection of the initial conditions. For example a least-squares fit with adjustable nodes of the initial conditions results in a method which is second order accurate, at least for $m = 1$ (see [9]).

It is quite straightforward to show that the method that has been used is a scale invariant method. If the scalings given by (4.10) and (4.11) are substituted into the mesh velocity equation we obtain

$$\int w \frac{\partial}{\partial \hat{x}} \left(\hat{u} \dot{\hat{x}} \right) \mathrm{d}\hat{x} = \int w \, \hat{u}_{\hat{t}} \, \mathrm{d}\hat{x}.$$

Notice that the basis functions w are independent of scale. Thus the mesh velocity equation is invariant to scalings of the form (4.10) and (4.11). Similarly the conservation of monitor function principle is scale invariant, since

$$\int w \, u \mathrm{d}x = \lambda^{-\alpha-\beta} \int w \, \hat{u} \mathrm{d}\hat{x}$$

and from the scaling arguments at the start of the chapter we know that $\alpha + \beta = 0$, so therefore the conservation of monitor function is scale-invariant for this choice of

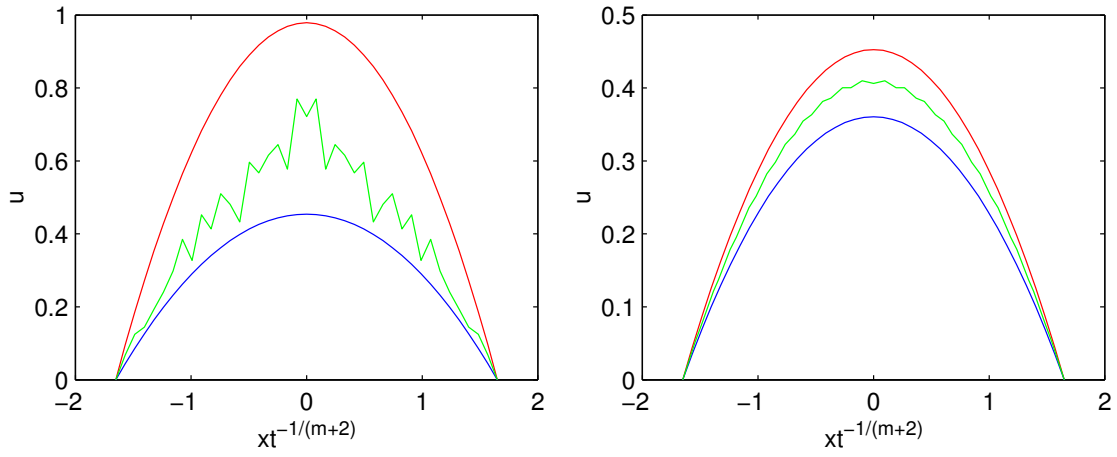

Figure 4.4: Figures showing the discrete comparison theorem of the method. Three initial conditions at $t = 0$ (left) and the three final solutions at $t = 1$ (right), both shown on scaled meshes for $m = 1$.

monitor. The time-stepping algorithm that is used to integrate the mesh points forward in time is also scale-invariant by construction.

Although no asymptotic properties of the moving mesh method have been proved analytically, there is strong numerical evidence to show that the moving mesh method that has been developed does indeed tend towards the correct solution as $t \to \infty$ and has therefore inherited the underlying properties of the porous medium equation. One example of this is the fact that the continuous PME satisfies a comparison principle in the scaled variables u and $x\,t^{-\frac{1}{m+2}}$, i.e. given three initial conditions u_1, u_2 and u_3 at t_{init}, which all have the same mass, such that

$$u_1 \leq u_2 \leq u_3$$

in the scaled variables then these inequalities hold for all time [25, 8]. Results appear to show that the numerical solutions also satisfy a discrete comparison theorem. Results are shown in figure 4.4 to this effect for three solutions which all have the same mass. Two of the initial solutions are given by exact self-similar solutions and the third, which is 'sandwiched' inbetween the other two, is a random (though symmetrical) perturbation. These initial conditions are then integrated forward in time and it is found that the random initial condition remains bounded by the two self-similar solutions and hence

asymptotically tends towards the self-similar soluton as time increases. A similar test was performed in [24].

The next section deals with the extension of the method to other monitor functions.

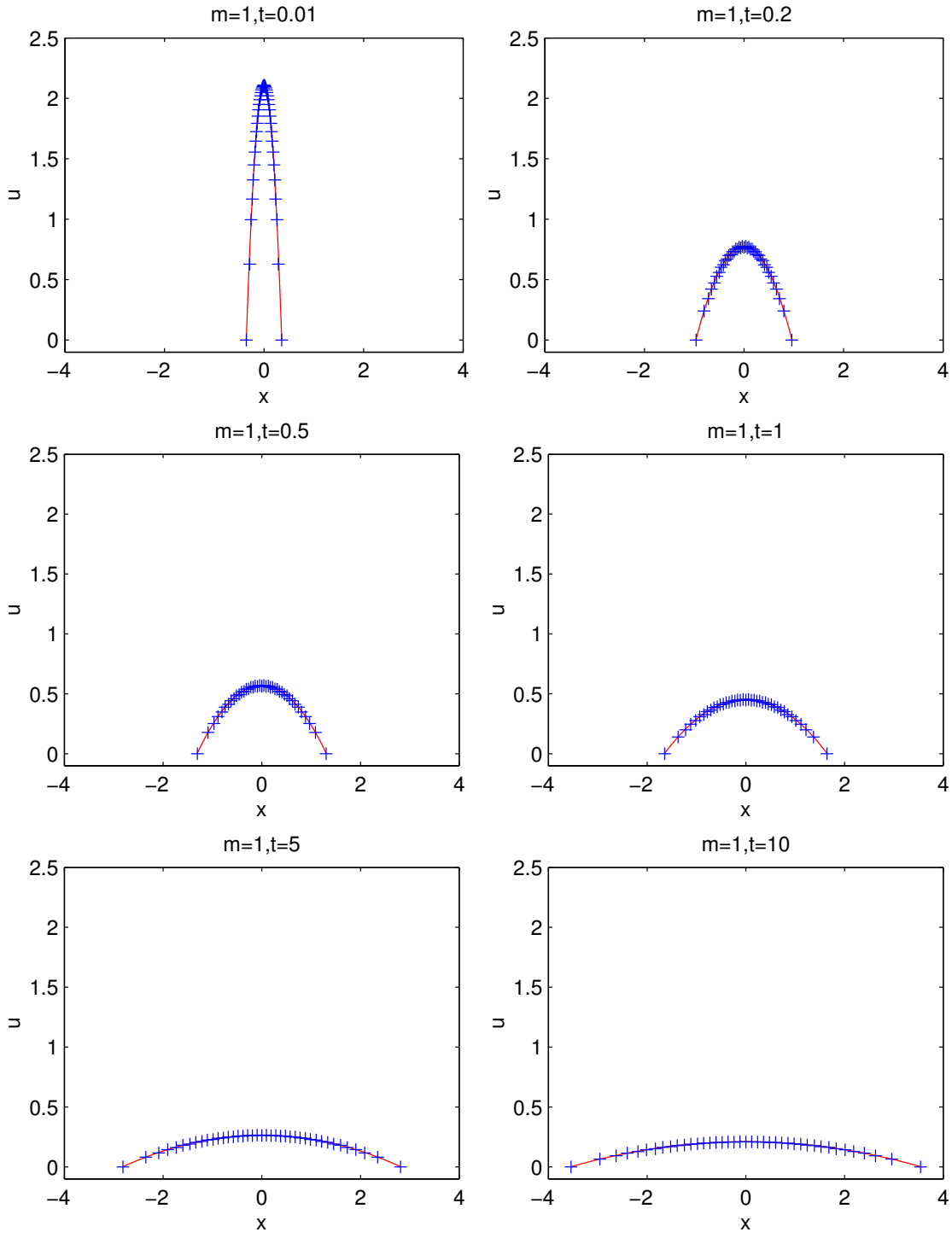

Figure 4.5: Numerical and exact solutions to the PME with $m = 1$ at six different times. Numerical solution computed with 41 nodes using the mass monitor function.

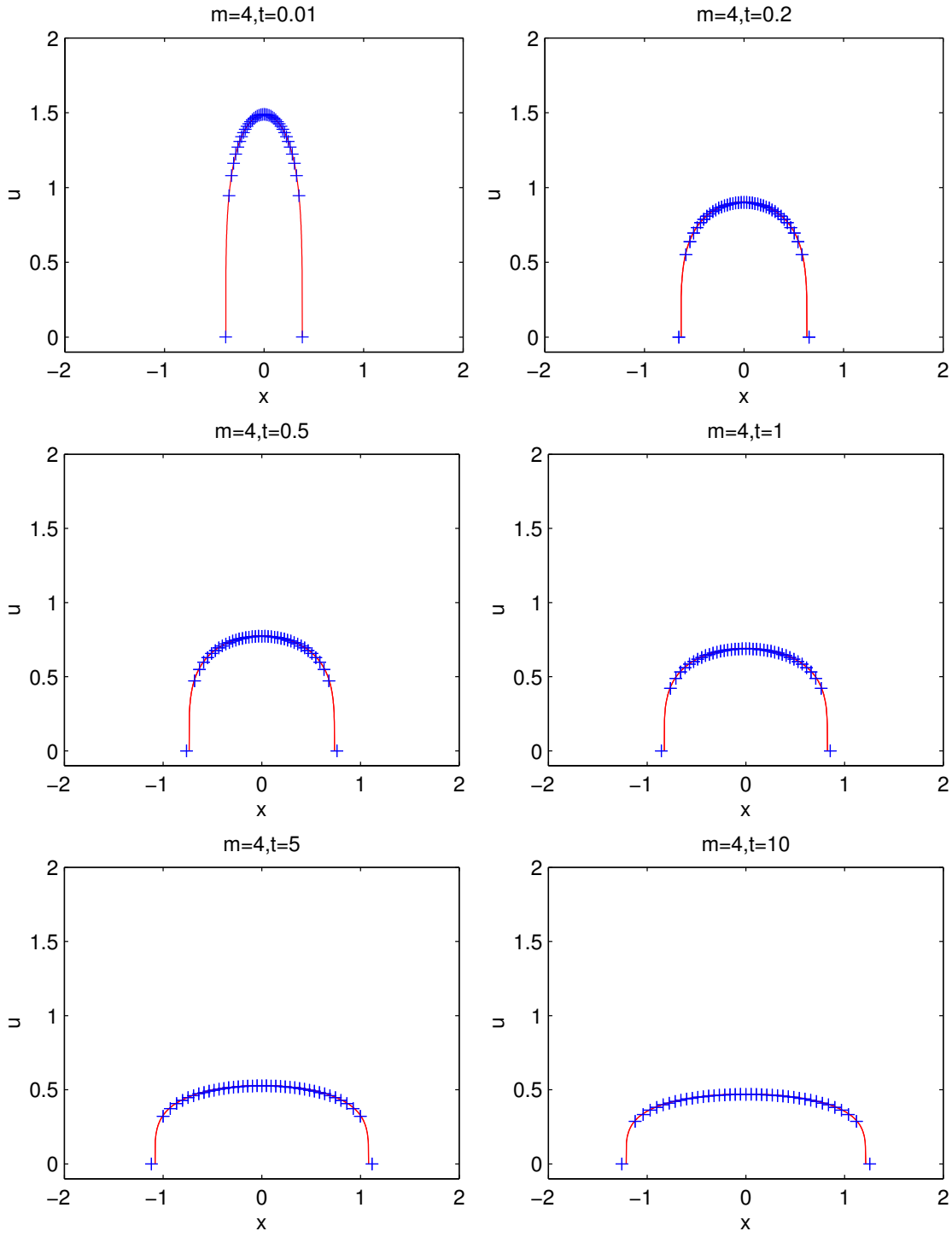

Figure 4.6: Numerical and exact solutions to the PME with $m = 4$ at six different times. Numerical solution computed with 41 nodes using the mass monitor function.

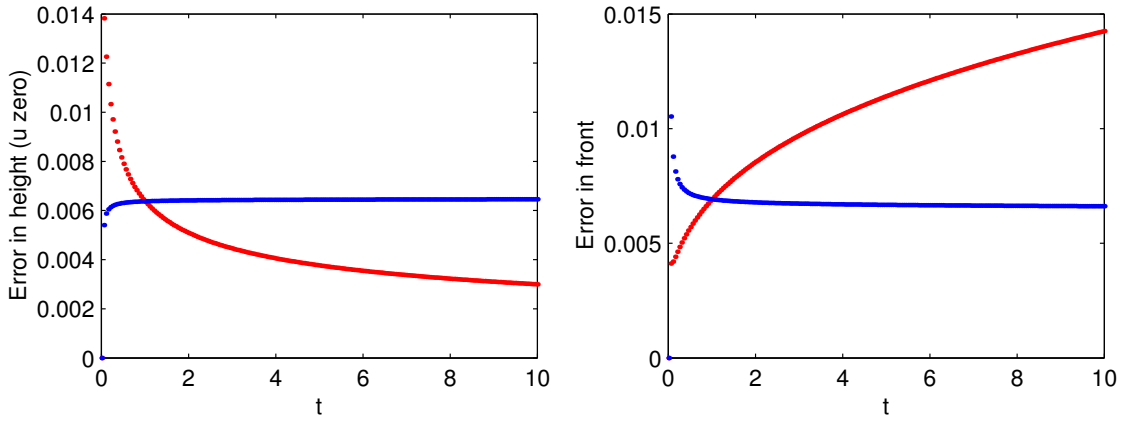

Figure 4.7: Left figure showing the error in the height of the solution (red) and the scaled error (blue). Right figure showing the error in the position of the moving boundary (red) and the scaled error (blue), both for $m = 1$.

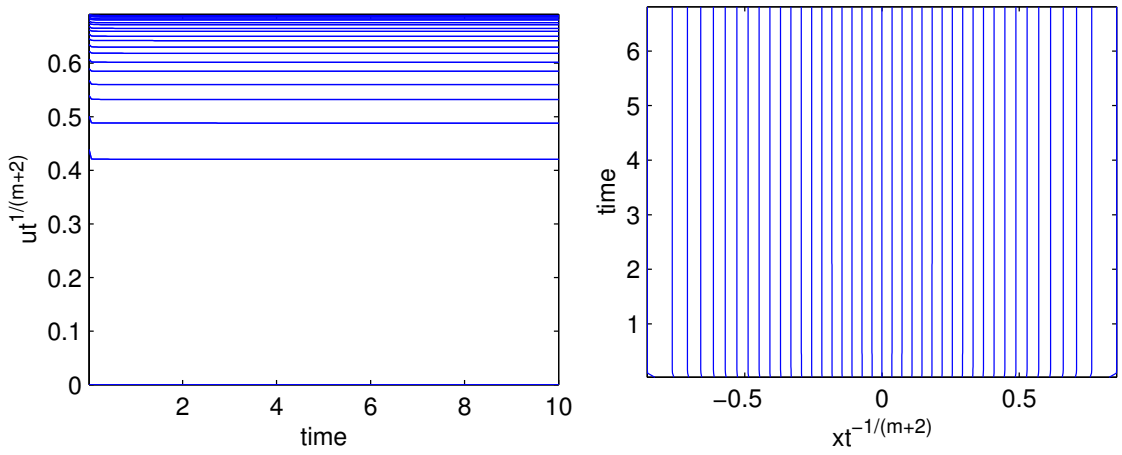

Figure 4.8: Invariant behaviour of the computational solution (left) and mesh (right) for $m = 4$ and 41 nodes.

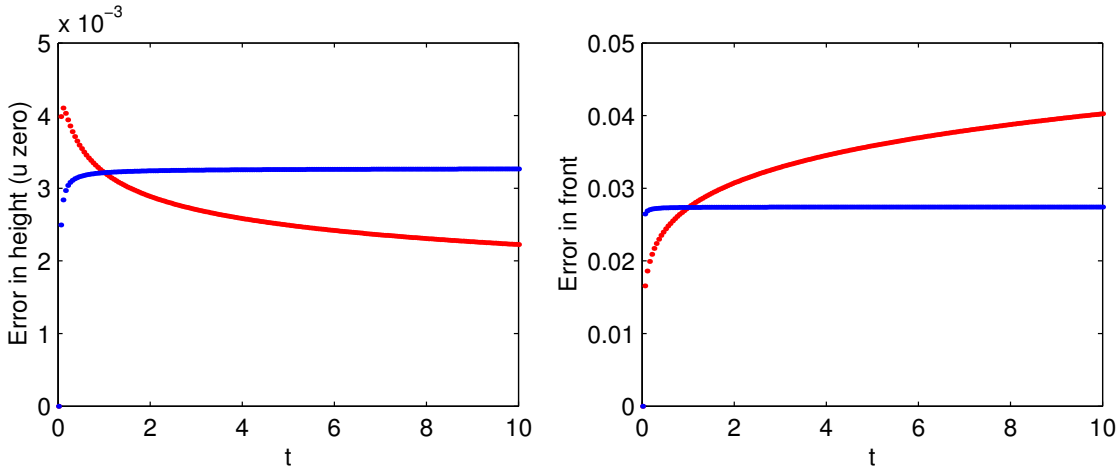

Figure 4.9: Left figure showing the error in the height of the solution (red) and the scaled error (blue). Right figure showing the error in the position of the moving boundary (red) and the scaled error (blue), both for $m = 4$.

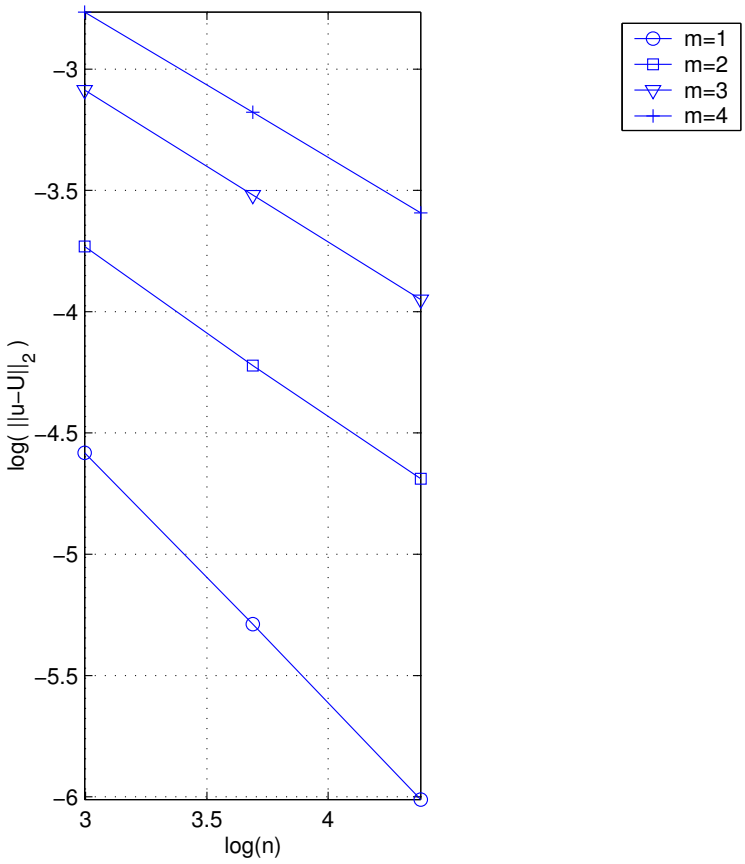

Figure 4.10: Log-Log plot of the error (Err) vs. the number of computational nodes for a variety of values of the power m.

4.5 General Monitor Functions

We now want to extend the ideas developed in the previous section to obtain a numerical solution to the PME using a general form of monitor function. When we used the mass monitor function in the previous section we took advantage of the fact that the global integral of the monitor function was constant in time and therefore we didn't need to calculate the change in the mass of the solution (see (4.5)). However if we want to use a different monitor function, its global integral will not be constant in time in general and therefore a normalisation needs to be taken into account. One way to do this is to generate a scale invariant monitor function.

Firstly, however, we will look at generating a scale invariant gradient monitor. It would be quite advantageous to use a gradient monitor function for the solution of this problem as it would cluster the computational nodes in and around the moving boundary to give better resolution. Unfortunately the integral of the gradient monitor is not scale invariant under the scalings (4.10) and therefore it is not constant in time. This can be seen if we substitute the scalings (4.10) and (4.11) into the gradient monitor to obtain

$$\int |u_x| \, \mathrm{d}x = \lambda^{-\beta} \int \lambda^{\beta-\alpha} |\hat{u}_{\hat{x}}| \, \mathrm{d}\hat{x} = \lambda^{-\alpha} \int |\hat{u}_{\hat{x}}| \, \mathrm{d}\hat{x}. \tag{4.23}$$

Since λ still appears on the right hand side of the above equation we know that the gradient monitor as it stands is not scale invariant, and hence the integral is not constant in time. However, if we use the fact that

$$\lambda = \frac{\hat{t}}{t},$$

in (4.23), we find that

$$t^{-\alpha} \int |u_x| \, \mathrm{d}x = \hat{t}^{-\alpha} \int |\hat{u}_{\hat{x}}| \, \mathrm{d}\hat{x}. \tag{4.24}$$

So the integral of the scaled monitor function $M = t^{-\alpha} |u_x|$ is scale invariant since (4.24) is independent of λ. Motivated by (4.24) we define the scale invariant conservation principle

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} t^{\frac{1}{m+2}} \left|u_x\left(x,t\right)\right| \mathrm{d}x = \theta_i.$$

Incidentally, (4.24) tells us the dependence of $\theta\left(t\right)$ on t in (3.2).

This argument can be generalised to determine other scale invariant monitor functions that can be used in our moving mesh method. Given a monitor function M, in d dimensions, which is not scale invariant, we can produce a scale invariant monitor function $\tilde{M}$ given by

$$\tilde{M}(t,x,u,u_r,\dots) = t^{-d\,\beta} M(1, t^{-\beta} r, t^{-\alpha} u, t^{\beta-\alpha} u_r, \dots).$$

In the case of the PME $\alpha = -\frac{d}{d\,m+2}$ and $\beta = \frac{1}{d\,m+2}$.

The rest of this section will be used to detail the use of the scale invariant gradient monitor function $M = t^{\frac{1}{m+2}}\, u_x$. The distributed conservation of monitor function principle (3.14) with this choice of monitor function becomes

$$t^{\frac{1}{m+2}} \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial U}{\partial x} \mathrm{d}x = \theta_i \tag{4.25}$$

for interior nodes. We have assumed without loss of generality that $\frac{\partial U}{\partial x} > 0$ since in each half of the spatial domain the solution of the PME is either monotone increasing or monotone decreasing (see figure 4.1). Expanding the finite element approximation U in terms of the basis functions we obtain

$$t^{\frac{1}{m+2}} \sum_{j=1}^{N} U_j \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial w_j}{\partial x} \mathrm{d}x = \theta_i,$$

which leads to the matrix form

$$B(\underset{\sim}{X})\,\underset{\sim}{U} = \underset{\sim}{\theta} \tag{4.26}$$

where the individual elements of the matrix B are given as

$$B_{ij} = t^{\frac{1}{m+2}} \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \,\frac{\partial w_j}{\partial x} \mathrm{d}x.$$

Notice that when the linear basis functions w_i are substituted into (4.25) the resulting matrix form has zero diagonal entries and therefore can no longer be solved by a standard tridiagonal solver. The resulting matrix system is therefore solved using a direct matrix solver with pivoting. It should also be noted that the $N \times N$ bi-diagonal matrix B is singular when the number of nodes N is taken to an even number.

The mesh velocity potential equation (3.17) with the gradient monitor gives

$$t^{\frac{1}{m+2}} \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\right) \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial t}\left(t^{\frac{1}{m+2}}\frac{\partial u}{\partial x}\right) \mathrm{d}x$$

The product rule of differentiation can now be used to expand the right hand side of the mesh velocity potential equation to give

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\right) \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right) \mathrm{d}x - \frac{1}{(m+2)t}\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial u}{\partial x}\,\mathrm{d}x$$

where we have also divided by the factor $t^{\frac{1}{m+2}}$. Using the fact that u is smooth we can interchange the order of differentiation in the following way

$$\frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial t}\right),$$

and so, enforcing this in a weak sense and using the PME, we obtain

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\right) \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x^2}\left(u^m \frac{\partial u}{\partial x}\right) \mathrm{d}x - \frac{1}{(m+2)t}\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial u}{\partial x}\,\mathrm{d}x. \tag{4.27}$$

The left hand side of the mesh velocity potential equation can now be integrated by parts to give

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\right) \mathrm{d}x = \left[w_i \frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\frac{\partial \phi}{\partial x}\,\mathrm{d}x \tag{4.28}$$

The first term on the left hand side of the velocity potential equation (4.27) may also be integrated by parts to give

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x^2}\left(u^m \frac{\partial u}{\partial x}\right) \mathrm{d}x = \left[w_i \frac{\partial}{\partial x^2}\left(u^m \frac{\partial u}{\partial x}\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(u^m \frac{\partial u}{\partial x}\right) \mathrm{d}x \tag{4.29}$$

Once the finite element approximations to the velocity potential ϕ and the solution u have been introduced and expanded in terms of the linear basis functions equations (4.28) and (4.29) become a stiffness matrix system for the approximation to the velocity potential Φ. However, we can simplify matters somewhat since the finite element basis functions are piecewise linear functions. Thus $\frac{\partial w_i}{\partial x}$ is constant in each computational cell, so we can remove these terms from inside the integral to obtain

$$\begin{aligned}
\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(U^m \frac{\partial U}{\partial x}\right) \mathrm{d}x = & \left(\frac{\partial w_i}{\partial x}\right)_{i+\frac{1}{2}} \int_{x_i(t)}^{x_{i+1}(t)} \frac{\partial}{\partial x}\left(U^m \frac{\partial U}{\partial x}\right) \mathrm{d}x \\
+ & \left(\frac{\partial w_i}{\partial x}\right)_{i-\frac{1}{2}} \int_{x_{i-1}(t)}^{x_i(t)} \frac{\partial}{\partial x}\left(U^m \frac{\partial U}{\partial x}\right) \mathrm{d}x, \\
= & \left(\frac{\partial w_i}{\partial x}\right)_{i+\frac{1}{2}} \left[U^m \frac{\partial U}{\partial x}\right]_{x_i(t)}^{x_{i+1}(t)} \\
& + \left(\frac{\partial w_i}{\partial x}\right)_{i-\frac{1}{2}} \left[U^m \frac{\partial U}{\partial x}\right]_{x_{i-1}(t)}^{x_i(t)}
\end{aligned}$$

where $\left(\frac{\partial w_i}{\partial x}\right)_{i+\frac{1}{2}}$ denotes the value of the derivative of the basis function w_i in the cell $[x_i, x_{i+1}]$. The cell boundary terms will be approximated by central differences, so

$$U^m \frac{\partial U}{\partial x}\Big|_{x_i(t)} \approx \frac{1}{2} U_i^m \left(\frac{U_{i+1} - U_i}{x_{i+1} - x_i} + \frac{U_i - U_{i-1}}{x_i - x_{i-1}}\right).$$

Our velocity potential equation for Φ is now given by

$$\left[w_i \frac{\partial U}{\partial x}\frac{\partial \Phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial U}{\partial x}\frac{\partial \Phi}{\partial x} \mathrm{d}x = -\left[w_i \frac{\partial}{\partial x^2}\left(U^m \frac{\partial U}{\partial x}\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} + \left(\frac{\partial w_i}{\partial x}\right)_{i+\frac{1}{2}} \left[U^m \frac{\partial U}{\partial x}\right]_{x_i(t)}^{x_{i+1}(t)} + \left(\frac{\partial w_i}{\partial x}\right)_{i-\frac{1}{2}} \left[U^m \frac{\partial U}{\partial x}\right]_{x_{i-1}(t)}^{x_i(t)} - \frac{1}{(m+2)t} \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial U}{\partial x} \mathrm{d}x,$$

for interior nodes. We also enforce the boundary condition that $\Phi = 0$ on the boundary of the domain. The mesh velocity is then recovered from Φ in the usual manner (4.19).

Again the mesh will be advanced in time by the scale invariant forward Euler time-stepping method given by (4.20). However, the solution to the PME will be recovered by inverting the system (4.25). We now go on to present some results obtained by using the scale invariant gradient monitor function.

4.6 Numerical Results II

This section summarises the results obtained using the scale invariant monitor function $M = t^{\frac{1}{m+2}} u_x$. We solved the same problem as in the case when $M = u$. So, again the initial condition for the problem was given by the similarity solution with $Q = 1$ and $t_0 = 0.01$. The initial solution was also equidistributed in the monitor function. The initial condition was then integrated forward in time until $t = 10$ using the method outlined in the previous section.

Results are shown in figure 4.11 for the case when $m = 4$. The results show that this choice of monitor function leads to a method which is less accurate in comparison with the results obtained with $M = u$, although the method tended to cluster most of the computational nodes in and around the moving boundary as can be seen in figure 4.12. Due to this, the solution around the peak tended to be less well resolved. It was also found that the error in the position of the moving front was much higher for the gradient monitor. This is probably caused by the fact that there is no constraint on the numerical solution to conserve the mass of the solution in a discrete manner, and since the position of the front is characterised by this conservation the numerical solution is attracted to a solution with a slightly different mass.

This conservation of mass problem can be avoided if instead of recovering the solution to the PME by inverting the matrix system (4.26) we were to solve the ALE equation

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\int u\,\mathrm{d}x &= \int \left(\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(u\,\dot{x}\right)\right)\mathrm{d}x \\ &= \int \left(\frac{\partial}{\partial x}\left(u^m\,\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial x}\left(u\,\dot{x}\right)\right)\mathrm{d}x \end{aligned}$$

on the moving mesh. In this way the solution to the PME is updated by using an integral form which would conserve mass.

The scale invariance properties of the method are shown are figure 4.13.

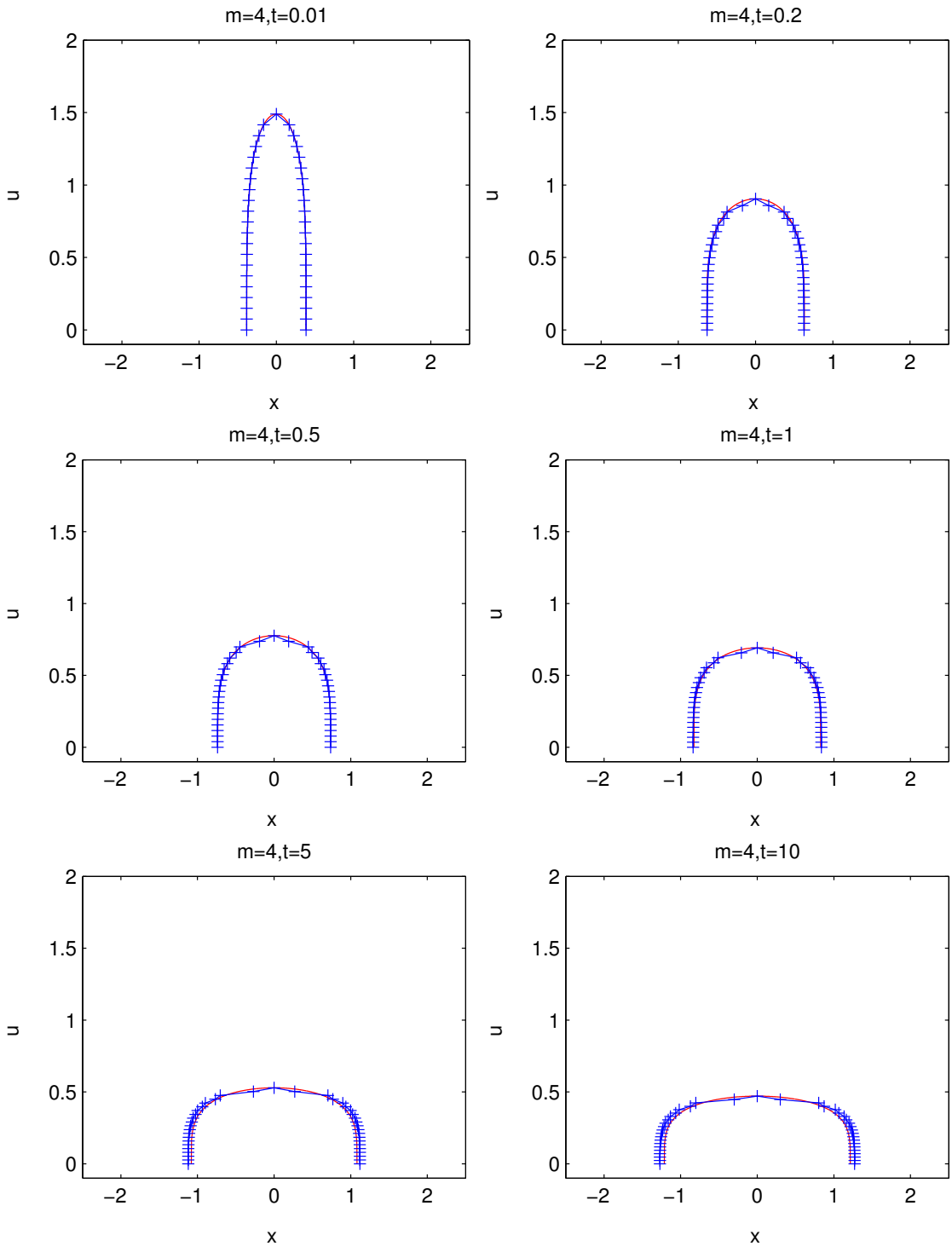

Figure 4.11: Numerical and exact solutions to the PME with $m = 4$ at six different times. Numerical solution computed with 41 nodes using the gradient monitor function.

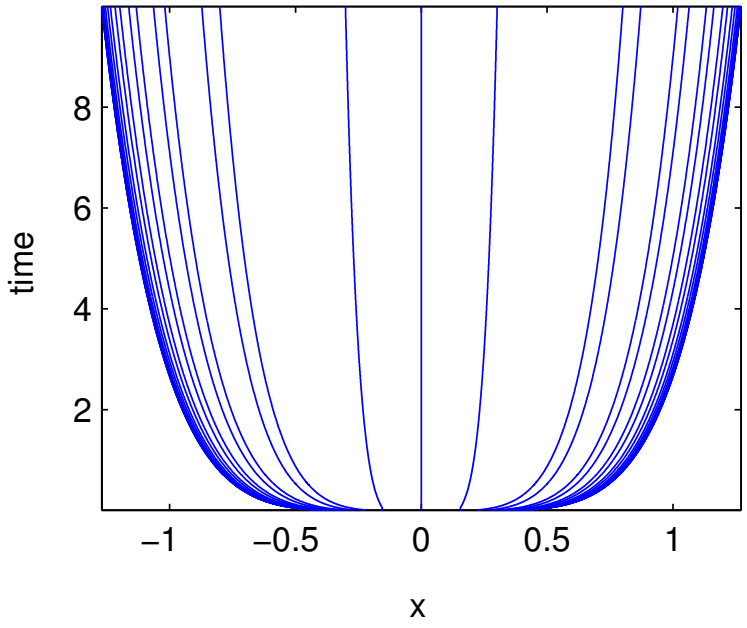

Figure 4.12: Node trajectories for $m = 4$.

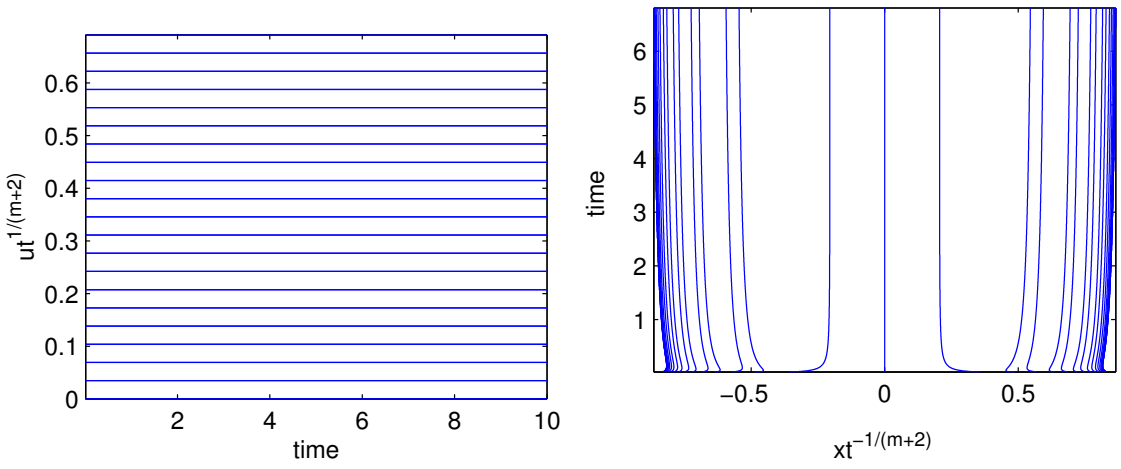

Figure 4.13: Invariant behaviour of solution (left) and mesh (right) for $m = 4$.

4.7 The PME in Two Spatial Dimensions

The aim of this section is to extend the method developed in one spatial dimension to the solution of the porous medium equation in two spatial dimensions. As the method outlined in chapter 3 was derived in multiple dimensions we expect that the extension of the method into two spatial dimensions should be reasonably straightforward. We have restricted attention to the mass monitor function given in section 4.3 as it gave more accurate results in one dimension in the nominal error measure when compared to the gradient monitor. We have also chosen to solve the PME on an unstructured triangular mesh as it will represent non-trival geometries more easily.

In section 4.2 we considered the self-similar solution to the porous medium equation in radial co-ordinates. We were motivated to use the mass monitor function in one-dimension since mass is conserved for this equation. The distributed conservation principle in two spatial dimensions for $M = u$ is

$$\int_{\Omega_i(t)} w_i \, U \, \mathrm{d}\Omega = \theta_i$$

where $\Omega_i(t)$ is now a patch of triangular cells on which the basis function w_i is non-zero. Following the same procedure as in one spatial dimension, we can expand our finite element approximation U in terms of the basis functions w_j as

$$U = \sum_{j=1}^{N} U_j \, w_j,$$

where w_j are now pyramid functions. So the distributed conservation of monitor function principle becomes

$$\sum_{j=1}^{N} U_j \int_{\Omega_i(t)} w_i \, w_j \, \mathrm{d}\Omega = \theta_i,$$

for $i = 1, \ldots, N$, which produces the mass matrix system

$$A(\underset{\sim}{\mathbf{X}}) \, \underset{\sim}{U} = \underset{\sim}{\theta}$$

where the entries of the mass matrix A are given by

$$A_{ij} = \int_{\Omega_i(t)} w_i \, w_j \, \mathrm{d}\Omega.$$

The condition that $U = 0$ on the boundary of the domain is also used. The velocity potential eqution in two dimensions with $\mathbf{q} = 0$ and $\varpi = 1$ becomes

$$\int_{\Omega_i(t)} w_i \, \nabla \cdot (u \, \nabla \phi) \, \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, u_t \, \mathrm{d}\Omega.$$

Again we can follow the same methodology as in one dimension and substitute a weak form of the PME into the velocity potential equation to give

$$\int_{\Omega_i(t)} w_i \, \nabla \cdot (u \, \nabla \phi) \, \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, \nabla \cdot (u^m \, \nabla u) \, \mathrm{d}\Omega.$$

To obtain a weak form of the velocity potential equation we can use Green's Theorem to give

$$\oint_{\partial\Omega_i(t)} w_i \, u \, \frac{\partial \phi}{\partial \mathbf{n}} \cdot \, \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot (u \, \nabla \phi) \, \mathrm{d}\Omega = - \oint_{\partial\Omega_i(t)} w_i \, (u^m \, \nabla u) \cdot \, \mathrm{d}\mathbf{\Gamma} + \int_{\Omega_i(t)} \nabla w_i \, \cdot (u^m \, \nabla u) \, \mathrm{d}\Omega,$$

which upon using the conditions that $w_i = 0$ around the edge of the patch $\Omega_i(t)$ and $u = 0$ on the boundary of the domain, gives

$$- \int_{\Omega_i(t)} \nabla w_i \, \cdot (u \, \nabla \phi) \, \mathrm{d}\Omega = \int_{\Omega_i(t)} \nabla w_i \, \cdot (u^m \, \nabla u) \, \mathrm{d}\Omega$$

for $i = 1, \ldots, N$. The finite element approximations for the mesh velocity potential and the solution to the PME can now be introduced and expanded in terms of the linear basis functions to obtain a discrete system to be solved. The mesh velocity potential again becomes a weighted stiffness matrix system which has the form

$$K \underset{\sim}{\Phi} = \underset{\sim}{f},$$

where the entries of the stiffness matrix are given by

$$K_{i\,j} = -\int_{\Omega_i(t)} \nabla w_i \cdot (U\,\nabla w_j)\,\mathrm{d}\Omega$$

and the entries of the vector $\underset{\sim}{f}$ are given by

$$f_i = \int_{\Omega_i(t)} \nabla w_i \cdot (U^m\,\nabla U)\,\mathrm{d}\Omega.$$

The condition that $\Phi = 0$ on the boundary of the domain is also used. The mass and stiffness matrices in two spatial dimensions are sparse and have a banded structure that depends on the connectivity of the mesh that is being used, but remain symmetric.

The mesh velocity $\dot{X}$ is again recovered in a weak sense from the mesh velocity potential Φ. Therefore we use the weak form

$$\int_{\Omega_i(t)} w_i\,\dot{X}\,\mathrm{d}\Omega = \int_{\Omega_i(t)} w_i\,\nabla\Phi\,\mathrm{d}\Omega$$

to recover the mesh velocity. Discretisation of this recovery equation with finite elements results in a series of mass matrix systems. Notice that now we have to solve two mass matrix systems, one for each component of the mesh velocity. Then the scale invariant time-stepping method described in section 4.3 is used to advance the mesh forward in time. Results for this moving mesh method for the solution of the two-dimensional PME are shown in the following section.

4.8 Numerical Results III

This section contains the numerical results for the solution of the PME in two spatial dimensions using the monitor function $M = u$. The PME was solved on three meshes, of varying size, consisting of triangular computational cells. These initial meshes are shown in figure 4.14 and were produced by the anisotrophic grid generator ANGENER [36]. Given an initial coarse mesh and boundary information ANGENER generates a mesh with a prescribed number of computational nodes.

The matrix systems resulting from the finite element discretisation were solved by a preconditioned conjugate gradient solver. The mass matrix systems were solved using a Jacobi preconditioner and the stiffness matrix system was not preconditioned. Numerical quadrature was used to evaluate many of the integrals, in particular Gaussian quadrature on triangles of order 13, which integrates bivariate polynominals of degree 7 exactly, was used with corresponding weights and abscissae found in [32]. This high degree numerical quadrature was used so that it was possible to solve the PME with high values of the exponent m, although lower numerical quadrature could be used for simulations with lower values of m.

Again the initial condition for the problem came from the similarity solution with the parameters $Q = 1$ and $t_0 = 0.01$. The solutions were then calculated from an initial time of $t = 0.01$ to a final time of $t = 10.0$ for various choices of the parameter m. Solutions for values of $m = 1$, $m = 2$ and $m = 4$ are shown in figures 4.15, 4.16 and 4.17. Also the solutions have been plotted against the radial similarity solution to give an indication of their accuracy in figure 4.18.

As with the results in 1D, the error measure

$$||u - U||_{L_2}^2 = \sum_i \int_{\Delta_i} (u - U)^2 \, \mathrm{d}\Omega$$

where Δ_i denotes the ith triangle has been calculated for the numerical solutions obtained on the three different meshes and for a variety of the values m. These results are summarised in figure 4.19. It can be seen from this figure that the numerical solution is second order accurate in space for $m = 1$, but for higher values of m the order of

the numerical method seems to be degraded. The error is now second order in space in contrast to results obtained in one spatial dimension where we found that the error was only first order accurate. This difference in the order of the numerical method may be attributed to how the initial mesh is designed and shows that it is important to obtain a good initial mesh for computations.

The precision of the numerical scheme also seems to lower as m increases. Again, this is probably due to the fact that the solution has an infinite gradient at the moving boundary for $m > 1$. This loss of accuracy and precision may be overcome by using different initial meshes which have more mesh points clustered in and around the boundary of the computational domain.

The moving mesh method was also applied to the PME with an initial condition which is different from the self-similar solution. The boundary of the support for the initial condition was designed to be non-circular and is described by the radial distance

$$r = \frac{a + b\cos(4\,\theta)}{a + b},$$

where $a = 1.0$ and $b = 0.2$ and the variable $\theta = \text{atan}\left(\frac{y}{x}\right)$. The initial condition for the PME on this initial mesh was then taken to be $u = 1 - \frac{r}{r_{max}}$. The PME with $m = 1$ and this inital condition was then solved with the moving mesh method. Solutions and meshes are shown for this test case in figures 4.20 and 4.21 at four different output times. It can be seen from these figures that both the solution and mesh are attracted towards the self-similar solution defined by the amount of mass contained in the initial condition.

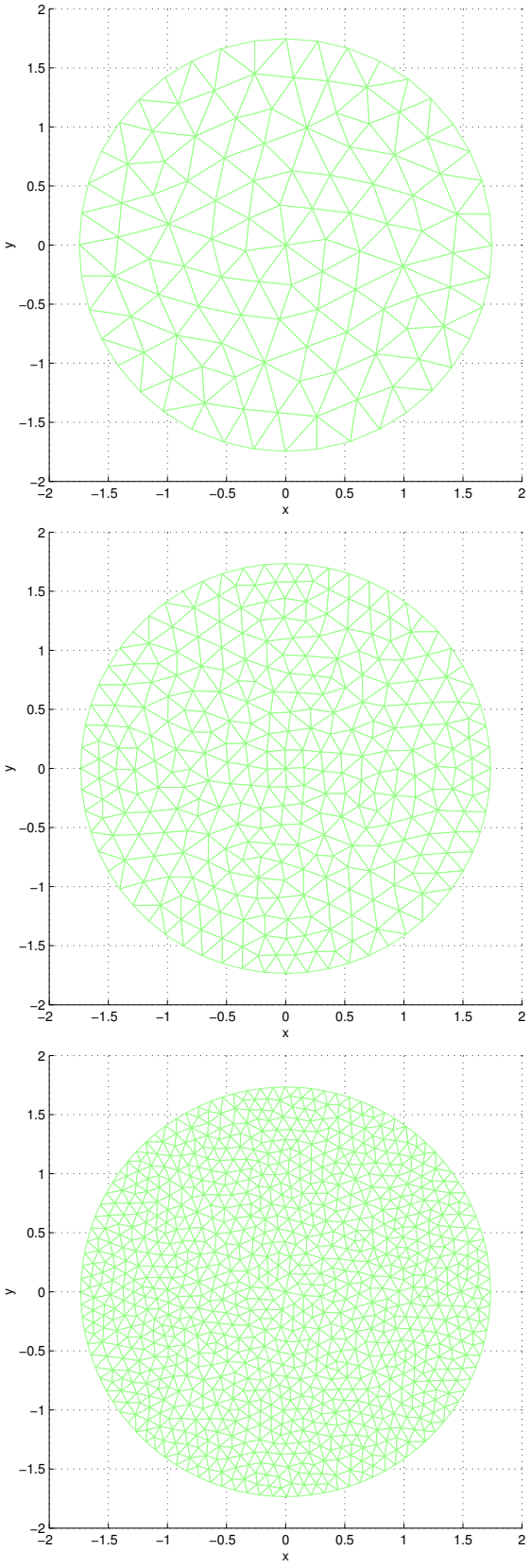

Figure 4.14: Initial meshes for the PME. Mesh 1 has 125 nodes and 208 computational cells. Mesh 2 has 315 nodes and 568 computational cells. Mesh 3 has 941 nodes and 1780 computational cells.

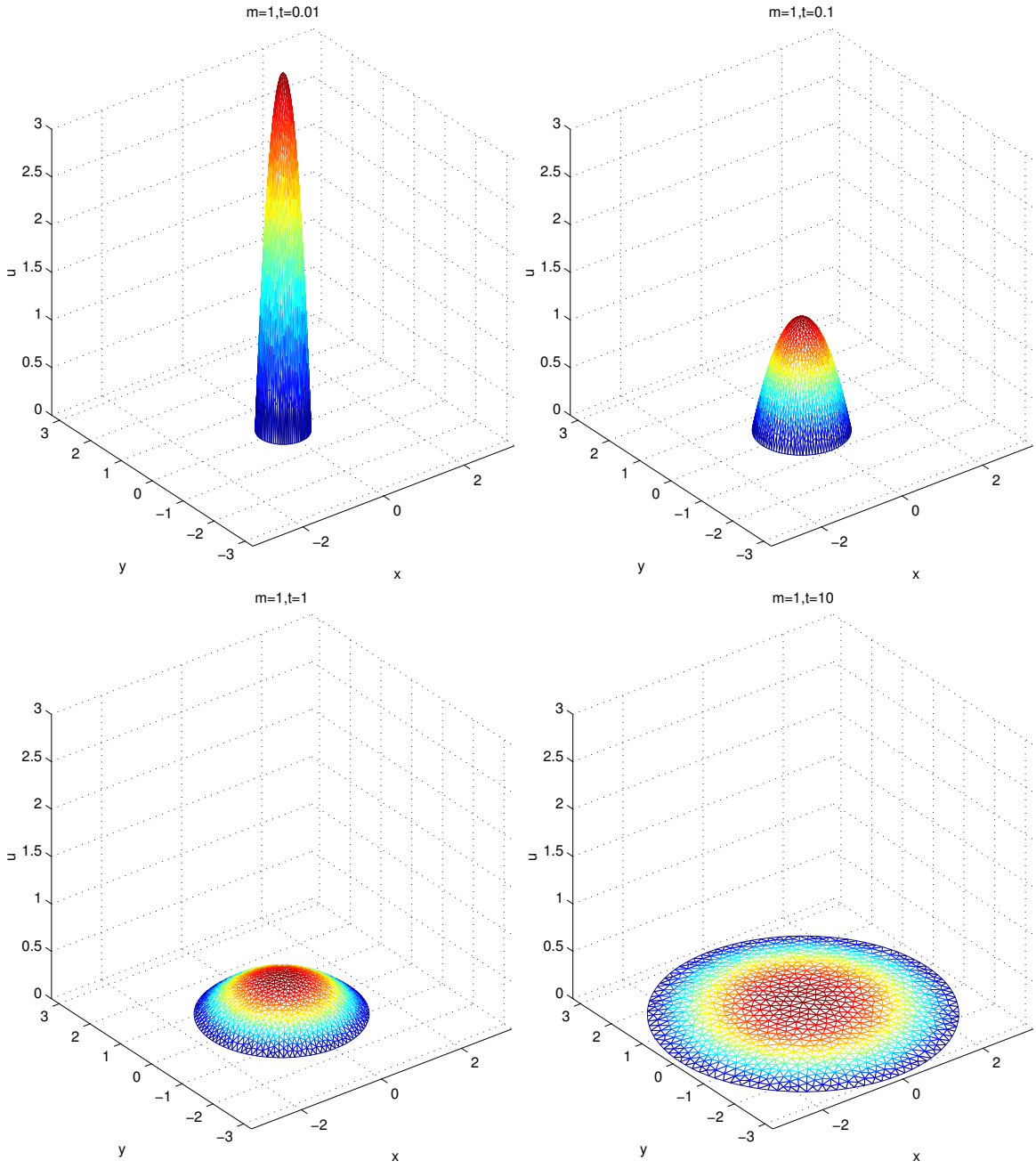

Figure 4.15: Numerical solutions to the PME with $m = 1$ at four different times. Numerical solution computed with 941 nodes.

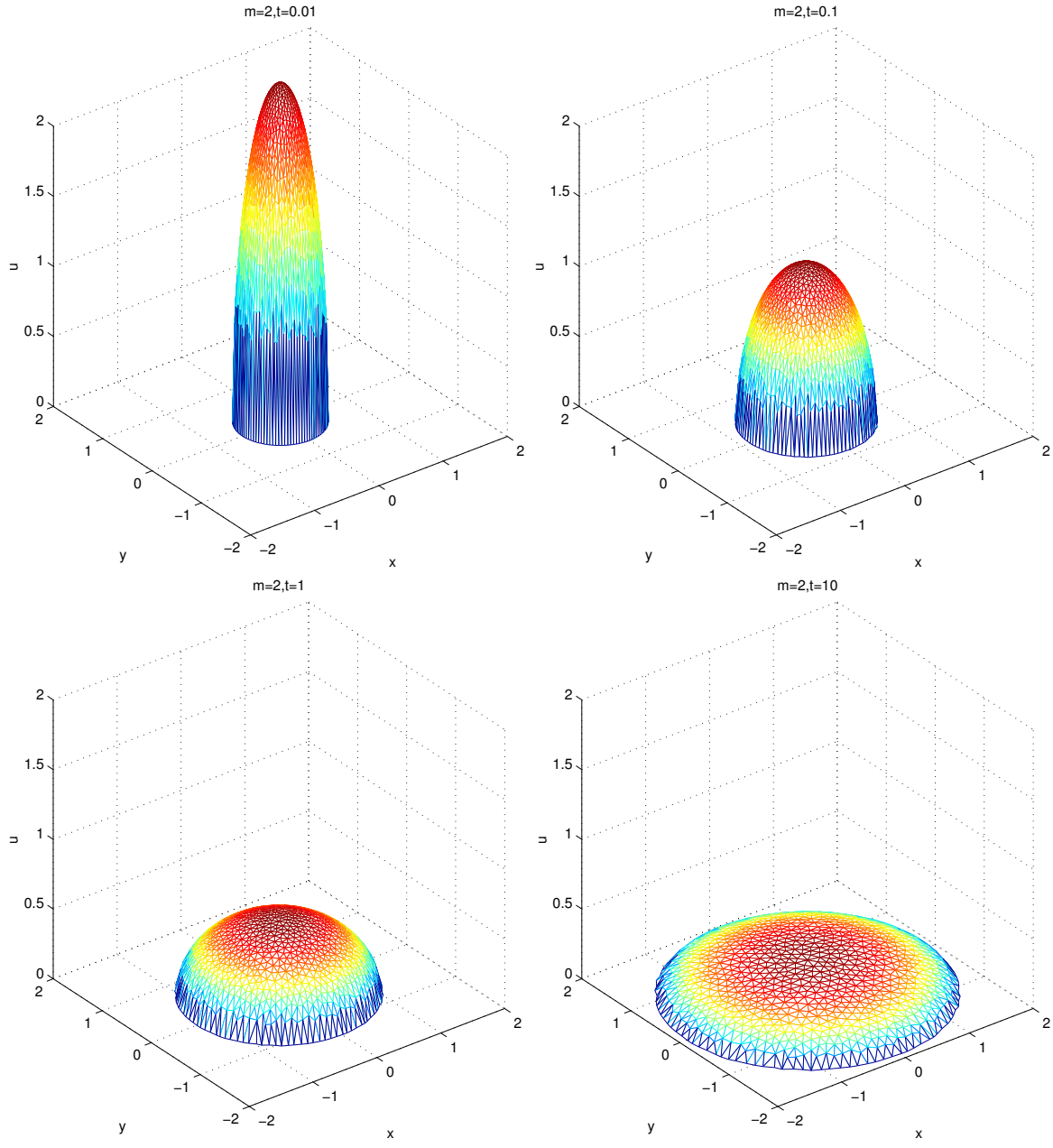

Figure 4.16: Numerical solutions to the PME with $m = 2$ at four different times. Numerical solution computed with 941 nodes.

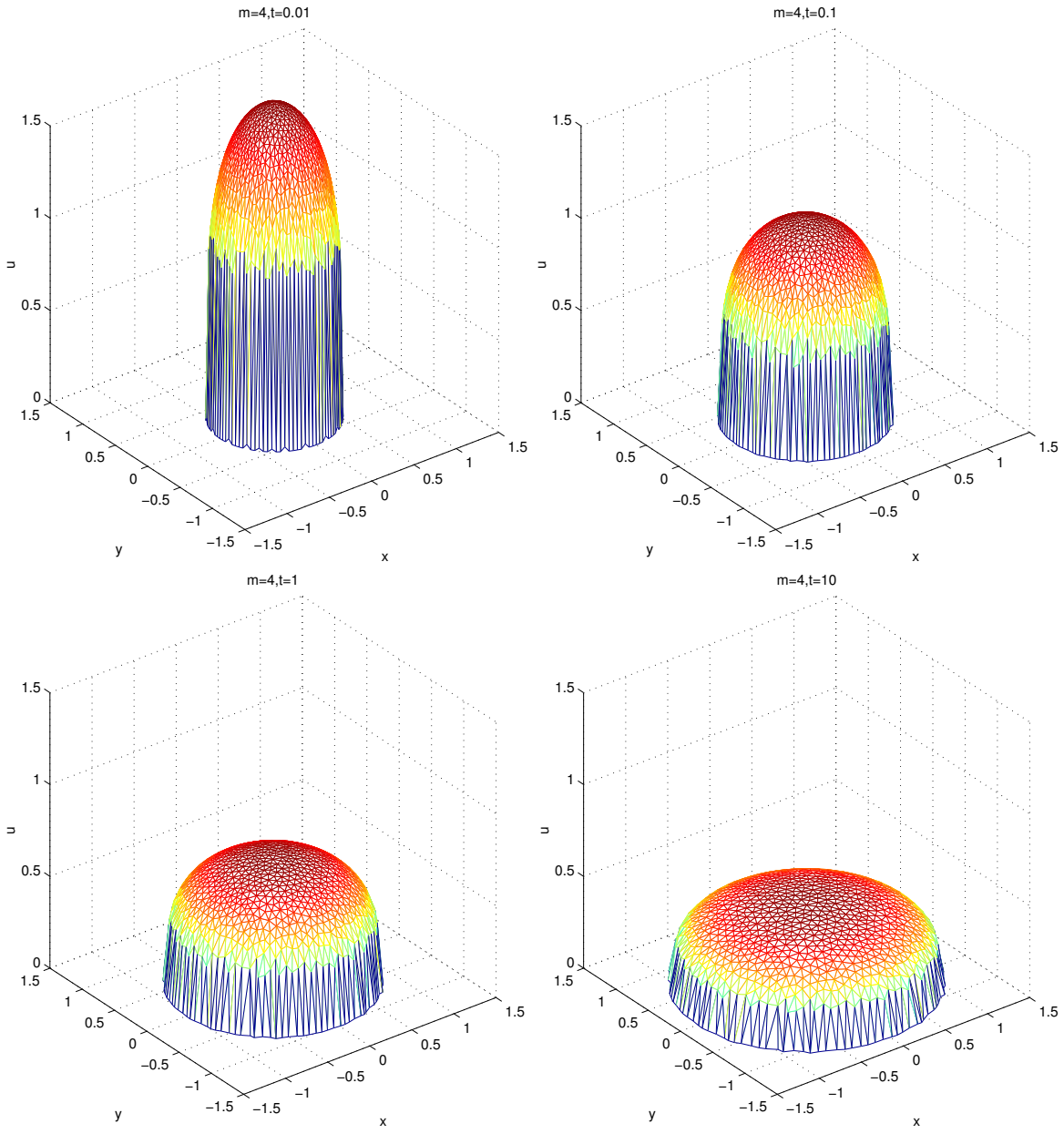

Figure 4.17: Numerical solutions to the PME with $m = 4$ at four different times. Numerical solution computed with 941 nodes.

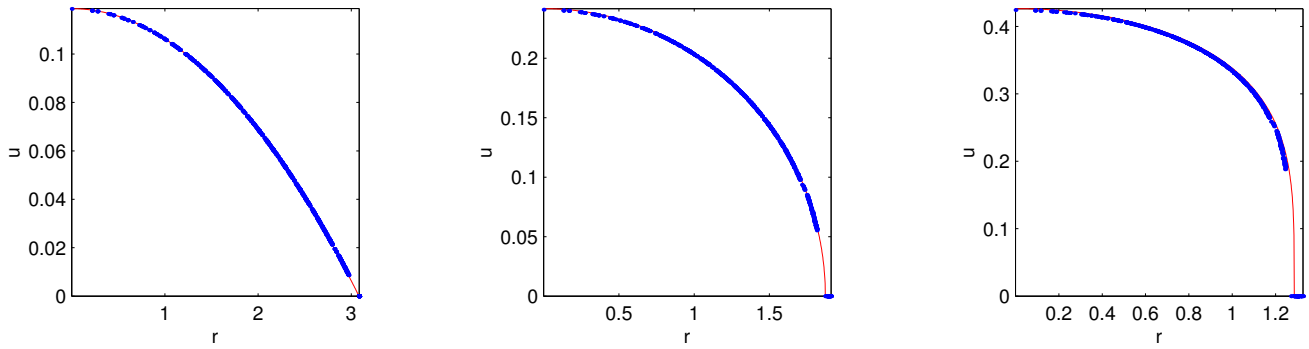

Figure 4.18: Solutions to the PME plotted against the radial distance for $m = 1$ (left), $m = 2$ (middle) and $m = 4$ (right). The solutions were computed on the finest mesh and are shown at $t = 10$.

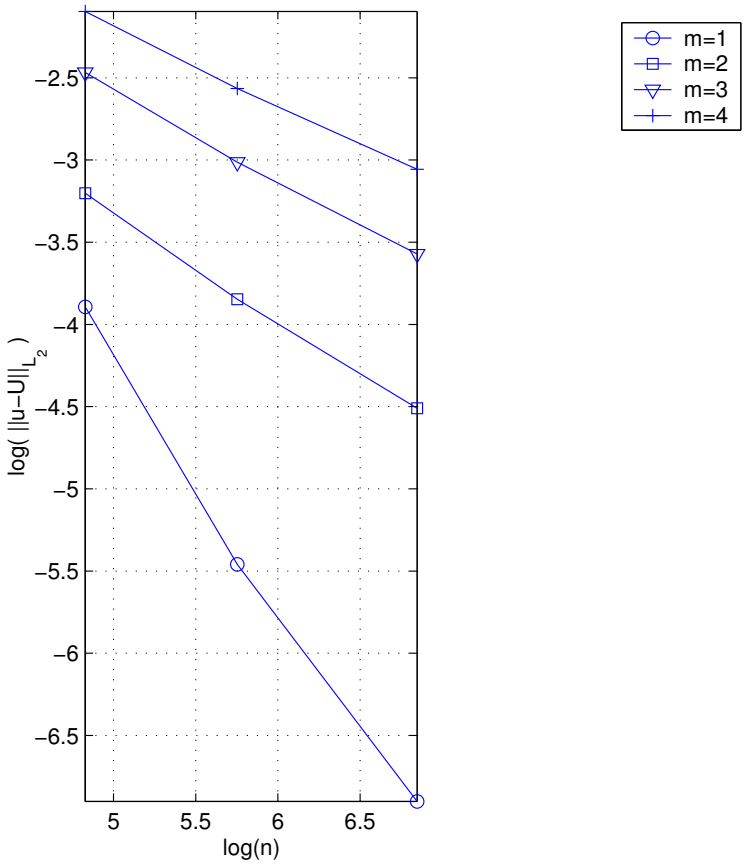

Figure 4.19: Log-Log plot of the error (Err) vs. the number of computational nodes for a variety of values of the power m.

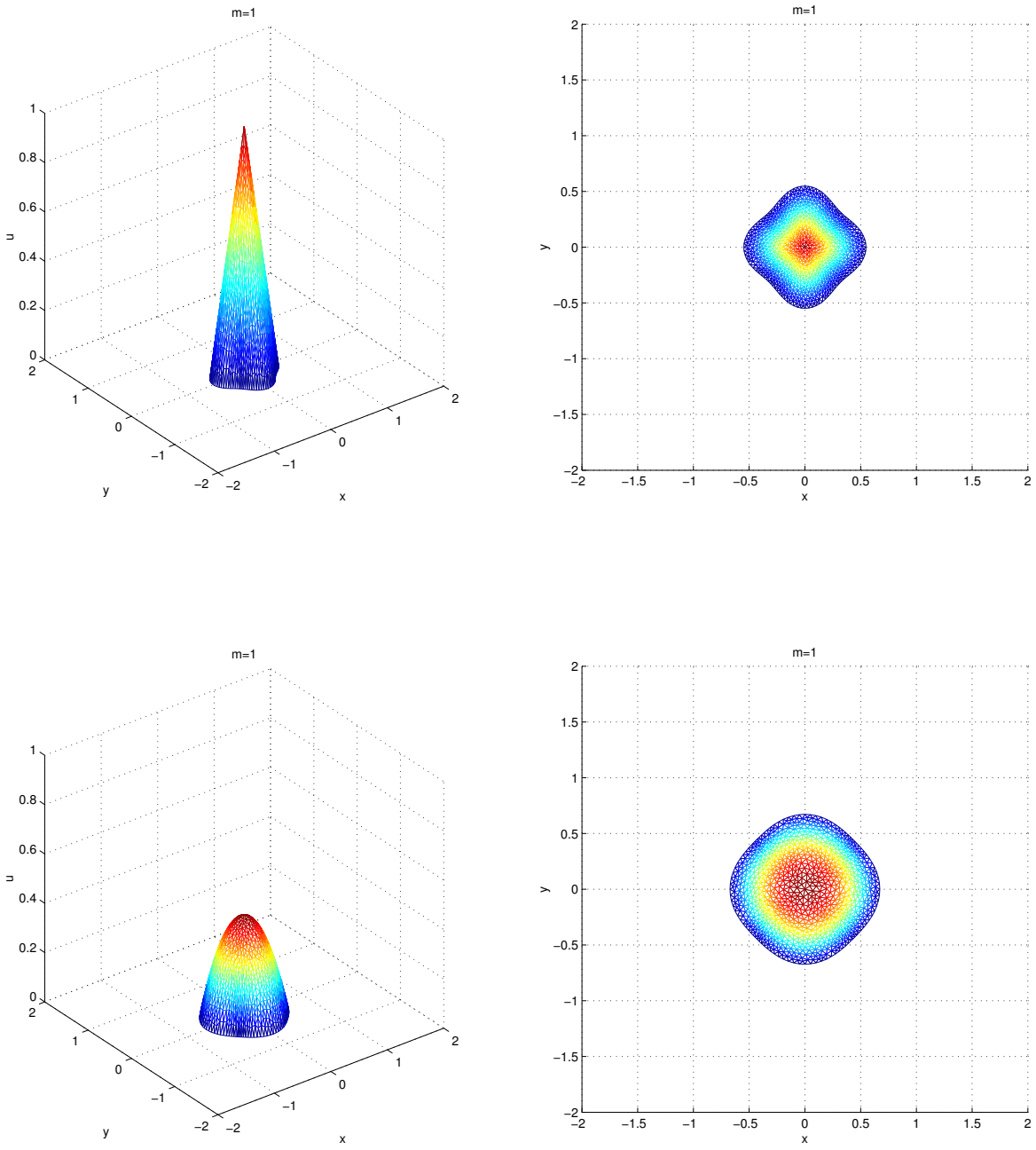

Figure 4.20: Numerical solutions (left) and meshes (right) to the PME with a non-self-similar initial condition and $m = 1$ times $t = 0.01$ and $t = 0.1$. Numerical solution computed with 941 nodes.

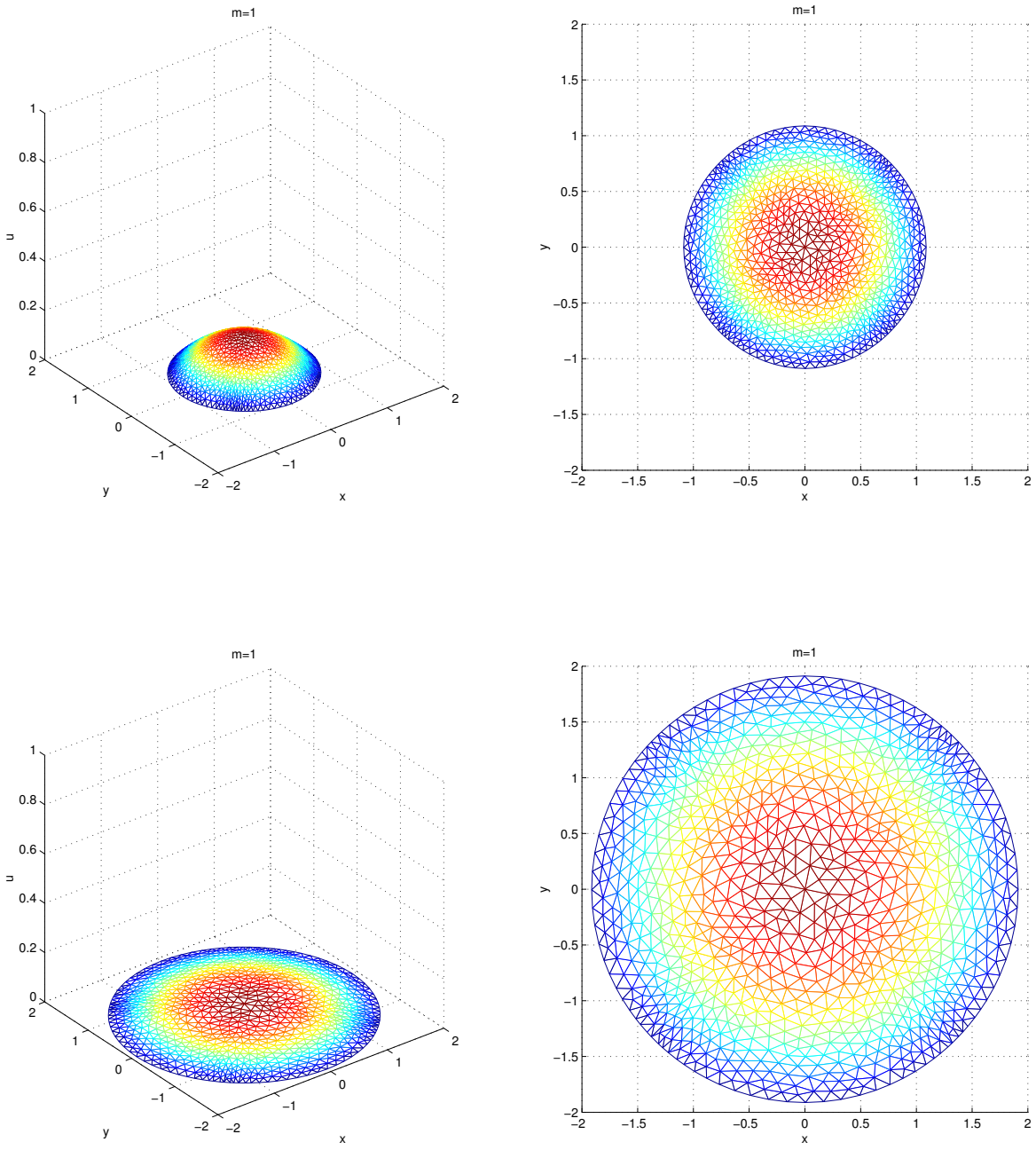

Figure 4.21: Numerical solutions (left) and meshes (right) to the PME with a non-self-similar initial condition and $m = 1$ *times* $t = 1$ *and* $t = 10$*. Numerical solution computed with 941 nodes.*

4.9 Summary

In this section the results that have been obtained for the solution of the porous medium equation using the moving mesh method derived in chapter 3 will be summarised.

We began this chapter by illustrating some of the analytical properties of the PME, in particular the scale invariant nature of the equation was shown. Also self-similar solutions to the PME in radial co-ordinates were given, and the analytical self-similar solutions were used throughout as a yardstick for the numerical solutions obtained. The properties of the PME were then used to construct suitable monitor functions which could be used in the moving mesh method that was derived in the previous chapter.

Firstly, the moving mesh method was used in conjunction with the monitor function $M = u$. This monitor function had been used by various other authors as it is consistent with the scaling properties of the PME. The resulting equation could then be solved for the mesh velocity potential and from this the mesh velocity. The system of ODEs was then integrated forward in time using a scale invariant forward Euler time-stepping procedure to obtain the position of the updated mesh. We were then able to recover the solution to the PME at the next time-level by inverting the conservation of monitor function principle.

Numerical solutions were then obtained from this method with an initially equidistributed mesh and initial solution. On comparison of the numerical solutions with the exact solutions the method was found to be accurate in both the solution value and the mesh position. The solutions were also found to exhibit many of the properties of the continuous PME.

To obtain better resolution of the moving front in the problem we then chose to solve the PME using the same method, but this time using a gradient monitor function. Since the integral of the gradient monitor function is not conserved in time globally for this problem, a scaled version of the gradient monitor function was derived which was conserved. This modified monitor function was derived from considering the scaling properties of the variables in the problem. We also considered how to adapt a general monitor function into a scale invariant one so that it could be used within this method. The numerical results obtained from the scaled gradient monitor function showed that

the method clustered most of the nodes tightly in the moving front and hence other regions of the solution were not as well resolved. Also due to the fact that mass was not discretely conserved in this solution the front was found to be in the incorrect position and therefore numerical errors increased with this choice of monitor function.

It should also be noted that for the mass and scaled gradient monitor function we were able to recover the solution to the PME directly through the conservation of monitor function principle, but for a general monitor function this may not be feasible due to difficulties with inversion. In that case the solution can still be recovered by using the ALE equation

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\int u\,\mathrm{d}\Omega &= \int\left(\frac{\partial u}{\partial t}+\nabla\cdot(u\,\dot{\mathbf{x}})\right)\,\mathrm{d}\Omega \\ &= \int\left(\nabla\cdot(u^m\nabla u)+\nabla\cdot(u\,\dot{\mathbf{x}})\right)\,\mathrm{d}\Omega = 0 \end{aligned} \tag{4.30}$$

to obtain $\int_{\Omega(t)} u\,\mathrm{d}\Omega$ on the moving mesh and then recovering u as in the case of the mass monitor. This choice has the advantage of conserving mass.

The remainder of the chapter was concerned with the solution of the PME in two spatial dimensions. In two dimensions we chose to use the mass monitor function to generate our moving mesh, due to the fact that it gave better results in one dimension when compared with the scale invariant gradient monitor function. This mass monitor function was found to give satisfactory results in two dimensions. However it was found that the method became less successful as the power in the porous medium equation increased in terms of the error measure, although the front is clearly well resolved.

As a final note we outline what happens when there is a lack of scale invariance in the problem being solved. For example, if we were solving the equation

$$u_t = \nabla\cdot(u^m\,\nabla u)+f\,(u)\,,$$

where f is a general function, then it would be unclear whether this problem is scale invariant and for many choices of the function f it would not be possible to scale the equation in a manner which leaves it invariant. In this case we still want to be able

to apply the moving mesh method for a general monitor function and unless we could identify a natural conserved quantity in the problem which could be used as a monitor function to drive the mesh movement, we would have to normalise the monitor function as described in chapter 3. The distributed conservation of monitor function principle (3.20) then becomes

$$\int_{\Omega_i(t)} w_i \, M \, \mathrm{d}\Omega = \kappa_i \, \theta(t), \tag{4.31}$$

where $\theta(t)$ is now a function of time and hence the monitor function is not conserved in time (cf. (3.2)). The constants κ_i are chosen so that they sum up to unity and may be chosen for example by an equidistribution strategy. Then, since no scale invariance exists so that we can scale the monitor function in such a way so that the monitor function is conserved in time, we must carry the extra variable $\theta\,(t)$. From the conservation principle (4.31), using a normalisation process, we can obtain the modified velocity potential equation for the moving mesh method given as

$$\int_{\Omega_i(t)} w_i \, \nabla \cdot (M \, \nabla \phi) \; \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, \frac{\partial M}{\partial t} \, \mathrm{d}\Omega + \kappa_i \, \dot{\theta},$$

where the variable $\dot{\theta}(t) = \frac{\mathrm{d}\theta}{\mathrm{d}t}$. It can be seen that if $\dot{\theta} = 0$, and hence the monitor function is conserved, we obtain the previous equation for the velocity potential (3.4). To be able to solve this equation for the velocity $\dot{\mathbf{x}}$ we need to determine $\dot{\theta}(t)$. An equation for the rate of change of the integral of the monitor function may come from various assumptions. If the condition $\phi = 0$ is used on the boundary of the domain then an independent equation for $\dot{\theta}$ is given as

$$\int_{\Omega(t)} \nabla \cdot (M \, \nabla \phi) \; \mathrm{d}\Omega = - \int_{\Omega(t)} \frac{\partial M}{\partial t} \, \mathrm{d}\Omega + \dot{\theta},$$

if boundary conditions for this equation, coming from the PDE being solved, are prescribed. Then we are able to solve this system for the velocity potential for internal nodes and also $\dot{\theta}(t)$ sumultaneously [9].

In the next chapters we will consider the solution of both scalar and systems of hyperbolic conservation laws using the moving mesh method derived in chapter 3. It will be found

that, since the solutions to these problems may be discontinuous in space (and the fact that we may be solving a system of equations), the solution to the PDE being solved will not be able to be recovered through the use of the conservation of monitor function principle as for the solution of the PME in this chapter. Therefore the solution will have to be obtained through solving the PDE on the moving mesh, as in (4.30). In the following chapters we will outline the use of a finite volume solver for the solution of the Arbitrary Lagrangian Eulerian (ALE) form of the conservation law being solved. Also, in the absense of scale invariance properties of the equation being solved we will have to base our choice for the monitor function on properties of the solutions.

Chapter 5

Hyperbolic Conservation Laws

5.1 Introduction

In this chapter the moving mesh method derived in chapter 3 is applied to the solution of first order partial differential equations including hyperbolic conservation laws. Some of the work may also be found in [98]. Hyperbolic conservation laws are time-dependent partial differential equations which often arise from the mathematical modelling of physical processes where certain quantities are conserved. For example, in many fluid dynamics problems mass, momentum and energy are conserved and this leads to a system of conservation laws. This type of equation also arises when viscous and dissipative effects are ignored in higher order equations. Hyperbolic conservation laws take the differential form

$$\frac{\partial \underline{\mathrm{u}}}{\partial t} + \nabla \cdot \underline{\mathbf{f}}\left(\underline{\mathrm{u}}\right) = \underline{0}, \tag{5.1}$$

where $\underline{\mathrm{u}}\left(\mathbf{x}, t\right) = \left(u_1, \ldots, u_m\right)^T$ is an m-vector of conserved variables which depends on the spatial variable $\mathbf{x}$ and the temporal variable t. The vector-valued function $\underline{\mathbf{f}}$ is called the flux function for the system and determines how the conserved variables change in time. In general, the flux function is non-linear in the conserved variables $\underline{\mathrm{u}}$. The underlying principle behind hyperbolic conservation laws is that, given the amount of each conserved variable $\underline{\mathrm{u}}\left(\mathbf{x}, t\right)$ at a given place and instant, we are able to determine the rate of flow, or flux $\underline{\mathbf{f}}\left(\underline{\mathrm{u}}\left(\mathbf{x}, t\right)\right)$, of this variable across any surface. Hence the change in the conserved variables within a closed surface can be found. The variables u_i for $i = 1, \ldots, m$ can be shown to be conserved in the following sense. The rate of change in

time of the total integral of each of the variables, in the domain Ω with boundary $\partial\Omega$, is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega} u_i \,\mathrm{d}\Omega = -\oint_{\partial\Omega} \mathbf{f}_i\left(\underline{\mathrm{u}}\right)\cdot \mathrm{d}\mathbf{\Gamma} \quad \text{for } i = 1,\ldots,m \tag{5.2}$$

and therefore if boundary fluxes are ignored then the quantities

$$\int_{\Omega} u_i \,\mathrm{d}\Omega \quad \text{for } i = 1,\ldots,m$$

are constant in time and hence conserved.

The system of conservation laws given by (5.1) can be written in the quasi-linear form

$$\frac{\partial \underline{\mathrm{u}}}{\partial t} + \frac{\partial \underline{\mathbf{f}}}{\partial \underline{\mathrm{u}}}\cdot\nabla\underline{\mathrm{u}} = \underline{0}$$

and is referred to as hyperbolic if any real linear combination of the Jacobian matrices $\frac{\partial \underline{\mathbf{f}}}{\partial \underline{\mathrm{u}}}$ have m real valued eigenvalues and a complete set of right eigenvectors. All the conservation laws considered in this work will have this hyperbolic property.

To be able to solve the system of conservation laws given by (5.1), on the whole real line, an initial condition, given by

$$\underline{\mathrm{u}}(\mathbf{x},0) = \underline{\mathrm{u}}_0(\mathbf{x}), \tag{5.3}$$

is needed. The problem of solving the system (5.1) together with the initial condition (5.3) is referred to as a Cauchy problem. In general the Cauchy problem cannot be solved analytically to obtain an exact solution, thus there is a need for accurate numerical methods to investigate the solutions to such problems. An example of a Cauchy problem for a conservation law is that of the Riemann problem. The Riemann problem has initial data of two constant states $\underline{\mathrm{u}}_L$ and $\underline{\mathrm{u}}_R$ separated by a single discontinuity. As we shall see later, many numerical methods for the solution of hyperbolic conservation laws rely on the solution of the Riemann problem, e.g. Godunov's method [42].

One characteristic feature of hyperbolic conservation laws is that solutions can evolve to have unbounded derivatives even with smooth initial data. These solutions do not satisfy

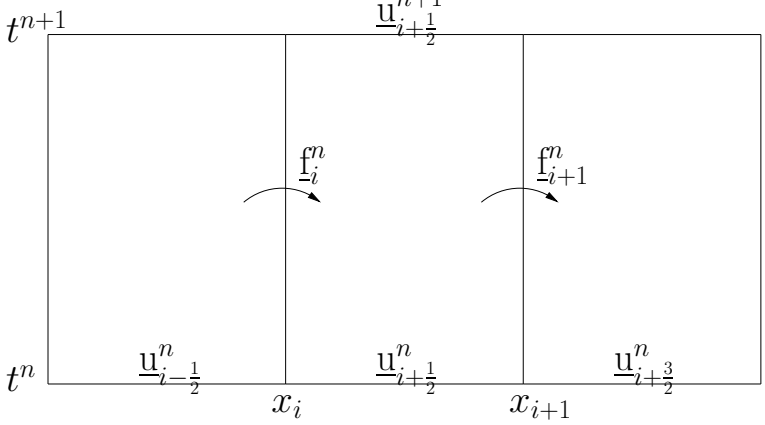

Figure 5.1: Figure showing how cell averages are updated by the fluxes through the cell boundaries.

the differential form (5.1) in the classical sense at all spatial points since derivatives are not defined at discontinuities in the solution gradient. Therefore the conservation law has to be understood in some generalised sense and the notion of a weak solution is introduced. A weak solution of the conservation law is defined as one that satisfies the integral form of the conservation law given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \phi\,\underline{\mathrm{u}}\,\mathrm{d}\Omega + \oint_{\partial\Omega} \phi\,\underline{\mathbf{f}}\cdot\mathrm{d}\mathbf{\Gamma} - \int_\Omega \nabla\phi\cdot\underline{\mathbf{f}}\,(\underline{\mathrm{u}})\;\mathrm{d}\Omega = \underline{0}$$

for all once differentiable functions ϕ belonging to a suitable space. It can be shown that the weak form of the conservation laws admits discontinuous solutions, such as shock waves.

Some of the theory of hyperbolic conservation laws can be found in [57, 84, 100] with good reviews of numerical methods for solving them in [60, 61, 93].

We now introduce some details of how hyperbolic conservation laws can be solved numerically. There are special difficulties associated with the numerical solution of hyperbolic conservation laws, many of them arising because solutions may be discontinuous. One problem which arises for hyperbolic conservation laws is that the numerical solution may not converge to the correct weak solution of the problem. A condition on the method for it to converge to the correct weak solution can be obtained and will now be outlined.

A sufficient requirement on the numerical method for it to converge to the correct solution is that the discretisation is written in conservative form. We will now consider

the conservative form of the hyperbolic conservation law in one spatial dimension in finite volume form. If we consider the conservation law (5.1) in one spatial dimension and integrate over the control volume $[x_i, x_{i+1}] \times [t^n, t^{n+1}]$, shown in figure 5.1, we obtain

$$\int_{x_i}^{x_{i+1}} \underline{\mathrm{u}}\left(x, t^{n+1}\right)\, \mathrm{d}x = \int_{x_i}^{x_{i+1}} \underline{\mathrm{u}}\left(x, t^{n}\right)\, \mathrm{d}x - \left[\int_{t^n}^{t^{n+1}} \underline{\mathrm{f}}\left(\underline{\mathrm{u}}\left(x_{i+1}, t\right)\right)\, \mathrm{d}t - \int_{t^n}^{t^{n+1}} \underline{\mathrm{f}}\left(\underline{\mathrm{u}}\left(x_{i}, t\right)\right)\, \mathrm{d}t\right]. \tag{5.4}$$

Now defining $\underline{\mathrm{u}}^n_{i+\frac{1}{2}}$ to be the cell average of the solution within the control volume at t^n, given by

$$\underline{\mathrm{u}}^n_{i+\frac{1}{2}} = \frac{1}{\Delta x} \int_{x_i}^{x_{i+1}} \underline{\mathrm{u}}\left(x, t^{n}\right)\, \mathrm{d}x \tag{5.5}$$

where $\Delta x = x_{i+1} - x_i$, then (5.4) becomes

$$\underline{\mathrm{u}}^{n+1}_{i+\frac{1}{2}} = \underline{\mathrm{u}}^{n}_{i+\frac{1}{2}} - \frac{1}{\Delta x}\left[\int_{t^n}^{t^{n+1}} \underline{\mathrm{f}}\left(\underline{\mathrm{u}}\left(x_{i+1}, t\right)\right)\, \mathrm{d}t - \int_{t^n}^{t^{n+1}} \underline{\mathrm{f}}\left(\underline{\mathrm{u}}\left(x_{i}, t\right)\right)\, \mathrm{d}t\right]. \tag{5.6}$$

If we now approximate the flux function by a time averaged one, given by

$$\underline{\mathrm{F}}_i\left(\underline{\mathrm{u}}_{i-\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{1}{2}}\right) \approx \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} \underline{\mathrm{f}}\left(\underline{\mathrm{u}}\left(x_{i}, t\right)\right)\, \mathrm{d}t \tag{5.7}$$

where $\Delta t = t^{n+1} - t^n$, then (5.6) becomes

$$\underline{\mathrm{u}}^{n+1}_{i+\frac{1}{2}} = \underline{\mathrm{u}}^{n}_{i+\frac{1}{2}} - \frac{\Delta t}{\Delta x}\left[\underline{\mathrm{F}}_{i+1}\left(\underline{\mathrm{u}}_{i+\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{3}{2}}\right) - \underline{\mathrm{F}}_i\left(\underline{\mathrm{u}}_{i-\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{1}{2}}\right)\right]. \tag{5.8}$$

This equation is a discrete conservative form of the conservation law in the sense that

$$\begin{aligned}
\Delta x \sum_{i=0}^{N-1} \underline{\mathrm{u}}^{n+1}_{i+\frac{1}{2}} &= \Delta x \sum_{i=0}^{N-1} \underline{\mathrm{u}}^{n}_{i+\frac{1}{2}} - \Delta t \sum_{i=0}^{N-1} \left[\underline{\mathrm{F}}_{i+1}\left(\underline{\mathrm{u}}_{i+\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{3}{2}}\right) - \underline{\mathrm{F}}_i\left(\underline{\mathrm{u}}_{i-\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{1}{2}}\right)\right], \\
&= \Delta x \sum_{i=0}^{N-1} \underline{\mathrm{u}}^{n}_{i+\frac{1}{2}} - \Delta t \left[\underline{\mathrm{F}}_{N}\left(\underline{\mathrm{u}}_{N-\frac{1}{2}}, \underline{\mathrm{u}}_{N+\frac{1}{2}}\right) - \underline{\mathrm{F}}_0\left(\underline{\mathrm{u}}_{-\frac{1}{2}}, \underline{\mathrm{u}}_{\frac{1}{2}}\right)\right], \\
&= \Delta x \sum_{i=0}^{N-1} \underline{\mathrm{u}}^{n}_{i+\frac{1}{2}},
\end{aligned}$$

apart from boundary fluxes and is analogous to the continuous equation (5.2). Therefore neglecting the boundary flux terms, the total amount of the conserved variables does not change in time. This discrete form can be used to define a numerical method to evolve the cell average values of the conserved variables. The time-averaged flux function $\underline{\mathrm{F}}_i$ at the i^{th} cell boundary is often referred to as the numerical flux function, as it is an approximation to the true flux at this boundary. Equation (5.8) can now be used as a method for updating the cell average of the solution in a given control volume and a method resulting from this equation is referred to as a finite volume method. However, if (5.8) is to be used as a numerical method we need to make a choice for the numerical flux function. There are many ways in which this might be done, but perhaps one of the most celebrated choices is that of Godunov. In [42] Godunov proposed a method for solving the Euler equations of gas dynamics by solving local Riemann problems between computational cells. Since the solution of the Riemann problem for the Euler equations can be found analytically, Godunov suggested that given piecewise constant data in each computational cell, local Riemann problems can be solved exactly or approximately between cells using the wave structure of the solution and then these local Riemann solutions can be pieced together to give a numerical solution to the conservation law. This can be seen in figure 5.2. The size of the time-step $\Delta t^n = t^{n+1} - t^n$ is assumed to be small enough so that the waves from neighbouring Riemann problems do not interact. This method has been generalised by constructing polynomial approximations in each computational cell at each time level and then solving local Riemann problems with polynomial initial data [58].

Denoting the number of computational cells by N, Godunov's method uses these N constant states as the initial data for a series of $N-1$ Riemann problems which can be solved and pieced together to give a numerical approximation to the true solution. In principle these non-linear Riemann problems could be solved exactly at each cell interface, but this process can be expensive numerically since it often involves solving a (complicated) set of non-linear equations using an iterative procedure. Therefore, in practice an approximate Riemann solver is usually employed since this is much cheaper to implement, and in any case the "piecing together" step is of low order.

Godunov's method was extended by Harten and Hyman [43] to solve hyperbolic conser-

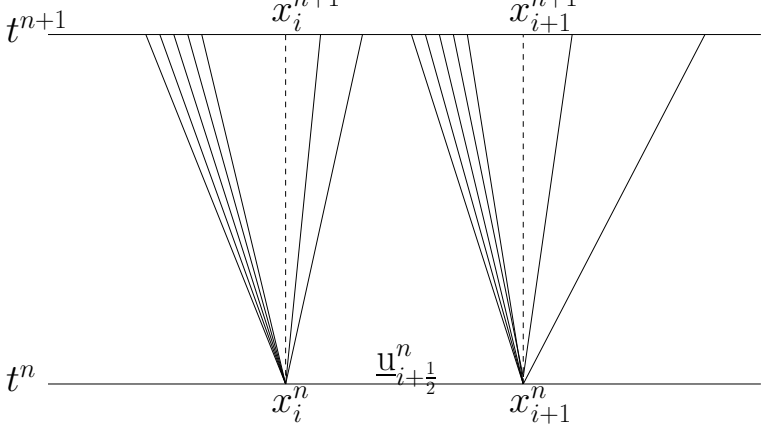

Figure 5.2: Figure showing the local wave structure of the Riemann problem in $x-t$ space.

vation laws on self-adjusting adaptive meshes. They used simple conservation principles to derive the generalisation of (5.8) in a moving reference frame and found that the flux function in the conservation law had to be shifted to take into account the motion of the mesh. This led to the Arbitrary Lagrangian Eulerian (ALE) form of the conservation law. This method has been used for example by Stockie, Mackenzie and Russell in [86] for the solution of one-dimensional hyperbolic conservation laws. The Arbitrary Lagrangian Eulerian (ALE) form of the conservation law will be introduced in the next section.

5.2 Arbitrary Lagrangian Eulerian (ALE) Form of the Conservation Law

The idea behind the Arbitrary Lagrangian Eulerian (ALE) form of a conservation law, originally introduced by Hirt, Amsden and Cook [45], was to improve the accuracy of methods for solving the equations of fluid motion. As stated in chapter 2 Hirt *et al* observe that the motion of the mesh does not have to be exclusively Eulerian or Lagrangian; that is it does not have to be either in a frame fixed in time or moving with the local fluid velocity. Therefore they transform the fluid equations into a general frame of reference which is moving with an arbitrary velocity which could give Eulerian, Lagrangian or some intermediate type of reference frame.

We now consider what happens to a conservation law when it is derived in a moving

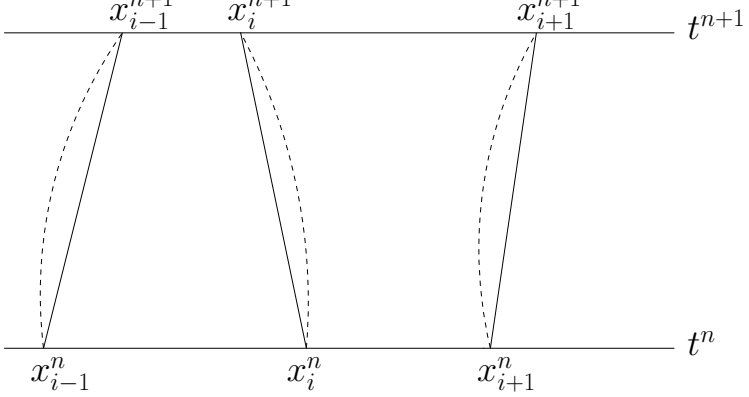

Figure 5.3: Figure showing the motion of computational cells in (x, t) space. The dotted and bold lines show the true and approximate trajectories of the computational nodes.

reference frame $\Omega(t)$. We will assume that this reference frame is moving with velocity $\dot{\mathbf{x}}$ which will be referred to as the ALE velocity. By the Reynolds Transport Theorem [72]

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \underline{\mathrm{u}}\,\mathrm{d}\Omega = \int_{\Omega(t)} \left(\underline{\mathrm{u}}_t + \nabla\cdot(\underline{\mathrm{u}}\,\dot{\mathbf{x}})\right)\,\mathrm{d}\Omega$$

and, using the differential form of the conservation law $\underline{\mathrm{u}}_t = -\nabla\cdot\underline{\mathbf{f}}$ and rearranging, we obtain the ALE form of the conservation law in the frame moving with $\dot{\mathbf{x}}$ as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \underline{\mathrm{u}}\,\mathrm{d}\Omega + \int_{\Omega(t)} \nabla\cdot(\underline{\mathbf{f}} - \underline{\mathrm{u}}\,\dot{\mathbf{x}})\,\mathrm{d}\Omega = \underline{0}.$$

Given $\dot{\mathbf{x}}$ this equation can be solved to obtain a solution to the conservation law in a moving reference frame. Notice that in the ALE form of the conservation law the flux function has been shifted to take into account the motion of the reference frame. Also when $\dot{\mathbf{x}} = 0$ we obtain the Eulerian form of the conservation law given by (5.1) and when $\dot{\mathbf{x}}$ is the same as the fluid velocity $\mathbf{v}$, i.e. $\dot{\mathbf{x}} = \mathbf{v}$, we obtain the Lagrangian form of the conservation law.

5.3 Godunov's Method on a Moving Mesh

We will now derive an ALE version of Godunov's method in one spatial dimension. First we define a set of finite volumes for the method. The spatial domain $[a, b]$ is split into

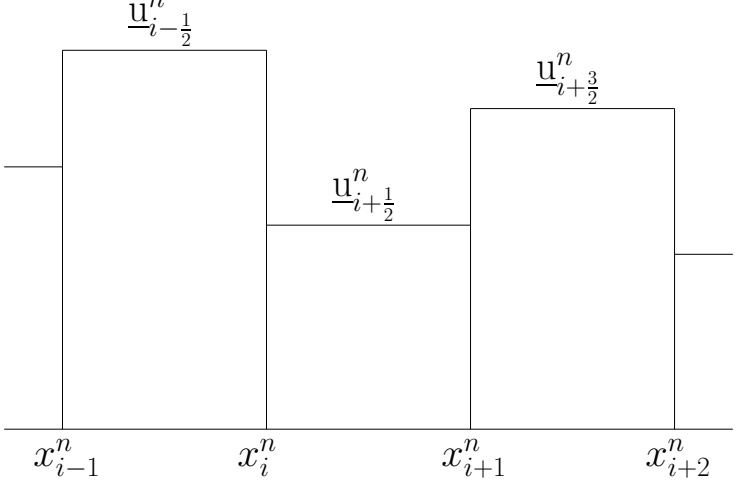

Figure 5.4: Figure showing the piecewise constant approximations in each computational cell.

N non-overlapping elements, $[x_i(t), x_{i+1}(t)]$ for $i = 0, \ldots, N-1$, which move in time. The mesh points are also ordered as

$$a \equiv x_0(t) < x_1(t) < \cdots < x_{N-1}(t) < x_N(t) \equiv b$$

and time is discretised into a series of (unequally) spaced times t^n with the size of the time-step given by $\Delta t^n = t^{n+1} - t^n$. Having a varying time-step allows the use of an adaptive time-stepping procedure, more details of which will be mentioned later. Typical computational cells are shown in figure 5.3 in (x, t) space.

Godunov's method assumes a piecewise constant approximation to the cell average in each computational cell $[x_i(t), x_{i+1}(t)]$ and this is shown in figure 5.4. Therefore the approximate value (the generalisation of (5.5) on a non-uniform moving mesh) is given by

$$\underline{\mathrm{u}}^n_{i+\frac{1}{2}} = \frac{1}{\Delta x^n_{i+\frac{1}{2}}} \int_{x^n_i(t)}^{x^n_{i+1}(t)} \underline{\mathrm{u}}\,(x, t^n)\ \mathrm{d}x$$

where $\Delta x^n_{i+\frac{1}{2}} = x^n_{i+1} - x^n_i$. A discrete approximation to the conservation law written in conservative form can then be derived based on conservation principles and is given as (cf. (5.8))

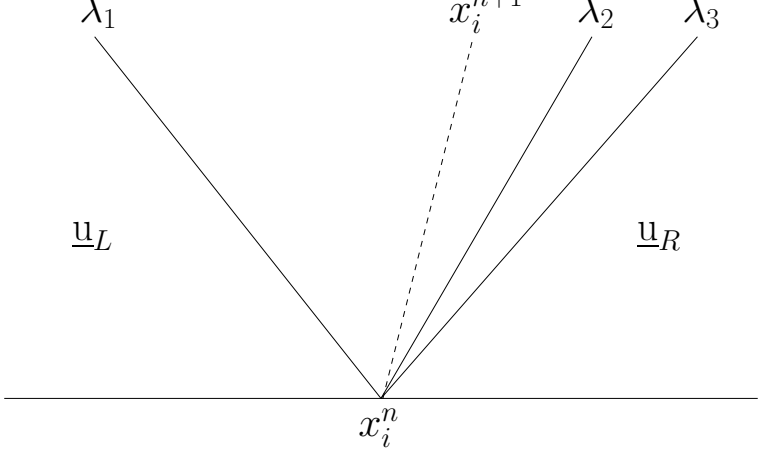

Figure 5.5: Figure showing the Riemann problem for the linearised system at node x_i with the initial states $\underline{u}_L$ and $\underline{u}_R$ for a 3-characteristic family of waves. The dashed line shows the approximate motion of the computational node between t^n and t^{n+1}.

$$\underline{\mathrm{u}}_{i+\frac{1}{2}}^{n+1} = \left(\frac{\Delta x_{i+\frac{1}{2}}^n}{\Delta x_{i+\frac{1}{2}}^{n+1}}\right) \underline{\mathrm{u}}_{i+\frac{1}{2}}^n - \frac{\Delta t^n}{\Delta x_{i+\frac{1}{2}}^{n+1}} \left[\bar{\underline{\mathrm{F}}}_{i+1}\left(\underline{\mathrm{u}}_{i+\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{3}{2}}\right) - \bar{\underline{\mathrm{F}}}_i\left(\underline{\mathrm{u}}_{i-\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{1}{2}}\right)\right]. \tag{5.9}$$

In Godunov's method on a moving mesh, the time-averaged flux function is defined to be

$$\bar{\underline{\mathrm{F}}}_i\left(\underline{\mathrm{u}}_{i-\frac{1}{2}}, \underline{\mathrm{u}}_{i+\frac{1}{2}}\right) \approx \frac{1}{\Delta t^n} \int_{t^n}^{t^{n+1}} \bar{\underline{\mathrm{f}}}\left(\underline{\mathrm{u}}\left(x_i, t\right)\right) \, \mathrm{d}t$$

(cf. (5.7)) where $\bar{\underline{\mathrm{f}}}$ is now the ALE flux function taken to be $\bar{\underline{\mathrm{f}}} = \underline{\mathrm{f}} - \underline{\mathrm{u}}\,\dot{\mathrm{x}}$.

In this work Roe's approximate Riemann solver [80] will be used to evaluate the numerical flux function in 1D. The main concept behind Roe's approximate Riemann solver is that rather than solving the original non-linear system of conservation laws, a locally linearised version is solved. The linearised system of conservation laws is given by

$$\underline{\mathrm{u}}_t + \tilde{A}\,\underline{\mathrm{u}}_x = \underline{0},$$

where $\tilde{A}$ is a locally linearised form of the Jacobian matrix $\frac{\partial \underline{\mathrm{f}}}{\partial \underline{\mathrm{u}}}$. Within each cell this matrix is taken to be a function of the left and right data $\underline{\mathrm{u}}_L$, $\underline{\mathrm{u}}_R$, i.e. $\tilde{A} = \tilde{A}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R)$ and these states are regarded as the left and right states of a Riemann problem. The waves associated with the solution of this Riemann problem are shown in figure 5.5. The

linearised system, together with the initial data for the Riemann problem, is much easier to solve than the original non-linear problem since it is a constant advection problem and can be solved exactly.

In [80] Roe suggested that at each interface between the states $\underline{\mathrm{u}}_L$ and $\underline{\mathrm{u}}_R$ the matrix $\tilde{A}$ should have the following three properties in order to keep some of the properties of the original system:

1. The linearised system should keep the hyperbolic nature of the original non-linear system.

$$\begin{aligned} &\tilde{A}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) \text{ is diagonalisable with real eigenvalues} \\ &\qquad \text{and linearly independent right eigenvectors.} \end{aligned} \tag{5.10}$$

2. The linearised system should be consistent with the exact Jacobian.

$$\tilde{A}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) \rightarrow \frac{\partial \underline{\mathrm{f}}}{\partial \underline{\mathrm{u}}}(\underline{\mathrm{u}}) \text{ smoothly as } \underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R \rightarrow \underline{\mathrm{u}} \tag{5.11}$$

3. The linearised system should be conservative across discontinuities.

$$\tilde{A}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R)(\underline{\mathrm{u}}_R - \underline{\mathrm{u}}_L) = f(\underline{\mathrm{u}}_R) - f(\underline{\mathrm{u}}_L) \tag{5.12}$$

Properties 1 and 2 ensure that the linearised system is consistent with the original problem and also that the hyperbolic nature of the non- linear system is preserved by the linearisation. Property 3 ensures that the linearised system is conservative and treats shocks correctly.

Since the linearised system is hyperbolic the eigenvalues and the right eigenvectors of the matrix $\tilde{A}$ can be found. The eigenvalues and eigenvectors of $\tilde{A}$ will be denoted by λ_i and $\underline{\mathrm{r}}_i$ respectively for $i = 1, \ldots, m$. These represent wavespeeds and wave profiles of the system being solved.

The jumps $\Delta\underline{\mathrm{u}}$ in the conservative variables $\underline{\mathrm{u}}$ can then be decomposed across the cells onto the eigenvectors $\underline{\mathrm{r}}_p$ as

$$\Delta \underline{\mathrm{u}} = \underline{\mathrm{u}}_R - \underline{\mathrm{u}}_L = \sum_{p=1}^{m} \alpha_p \, \underline{\mathrm{r}}_p, \tag{5.13}$$

where α_p are called the wave strengths. Equation (5.13) is a set of m algebraic equations which can be solved to obtain the wave-strengths α_p for $p = 1, \ldots, m$. It is shown in [93] and [60] that the upwinded numerical flux for Roe's method can be formulated in three equivalent ways. These are given as

$$\begin{aligned}
\underline{\mathrm{F}}(\mathrm{u}_L, \mathrm{u}_R) &= \underline{\mathrm{f}}(\underline{\mathrm{u}}_L) + \sum_{p=1}^{m} \lambda_p^- \, \alpha_p \, \underline{\mathrm{r}}_p, \\
\underline{\mathrm{F}}(\mathrm{u}_L, \mathrm{u}_R) &= \underline{\mathrm{f}}(\underline{\mathrm{u}}_R) - \sum_{p=1}^{m} \lambda_p^+ \, \alpha_p \, \underline{\mathrm{r}}_p, \\
\underline{\mathrm{F}}(\mathrm{u}_L, \mathrm{u}_R) &= \frac{1}{2} \left(\underline{\mathrm{f}}(\underline{\mathrm{u}}_L) + \underline{\mathrm{f}}(\underline{\mathrm{u}}_R) \right) - \frac{1}{2} \sum_{p=1}^{m} |\lambda_p| \, \alpha_p \, \underline{\mathrm{r}}_p,
\end{aligned}$$

where $\lambda_p^- = \min(0, \lambda_p)$ and $\lambda_p^+ = \max(0, \lambda_p)$.

In the moving case the numerical flux function needs to be modified to take into account the movement of the mesh. This is done by shifting the eigenvalues of the original linearised system in an Eulerian frame by the mesh velocity. The flux function $\underline{\mathrm{f}}$ also changes to the ALE flux function $\bar{\underline{\mathrm{f}}} = \underline{\mathrm{f}} - \dot{x} \, \underline{\mathrm{u}}$. Therefore the numerical flux function at the i^{th} intercell boundary with left and right data states $\underline{\mathrm{u}}_L$ and $\underline{\mathrm{u}}_R$ on a moving mesh is given by

$$\underline{\mathrm{F}}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) = \frac{1}{2} \left(\bar{\underline{\mathrm{f}}}(\underline{\mathrm{u}}_L) + \bar{\underline{\mathrm{f}}}(\underline{\mathrm{u}}_R) \right) - \frac{1}{2} \sum_{p=1}^{m} |\tilde{\lambda}_p| \, \alpha_p \, \underline{\mathrm{r}}_p \tag{5.14}$$

where $\tilde{\lambda}_p = \lambda_p - \dot{x}_i^n$ is the shifted wave speed. Once we have calculated the eigenvalues, the right eigenvectors and the wave strengths for the system, we can then evaluate the numerical flux function given by equation (5.14). (Of course we still need the ALE velocities $\dot{x}_i^n$ for $i = 0, \ldots, N$.) This numerical flux function can then be used in the conservative update formula to evolve the cell average values forward in time.

The finite volume method, together with this choice for the numerical flux function, leads to a method that is only first order accurate. The method will therefore introduce

diffusion into the numerical solution and hence tend to smear out discontinuities. It is possible to include second order corrections to the finite volume method to produce a second order accurate method. However, doing so may also introduce spurious oscillations near discontinuities. One way to obtain higher order accuracy without oscillations is to use flux limiters [60, 93] which operate near discontinuities. In this way the flux function is limited so that in regions where the solution is smooth we have second order accuracy, but in the vicinity of discontinuities (where order of accuracy is less important than monotonicity) we only have first order accuracy. We can write the limited form of the flux function as

$$\underline{\mathrm{F}}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) = \underline{\mathrm{F}}_{Low}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) + \Phi(\theta)\left[\underline{\mathrm{F}}_{High}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) - \underline{\mathrm{F}}_{Low}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R)\right], \tag{5.15}$$

where $\underline{\mathrm{F}}_{Low}$ and $\underline{\mathrm{F}}_{High}$ denote low (first) and high (second) order flux functions respectively. The function $\Phi(\theta)$ is called the flux limiter and is a function of a variable θ which identifies the local behaviour of the solution. The variable θ is conveniently taken to be the ratio of consecutive wave strengths. It can easily be seen from (5.15) that when $\Phi(\theta) = 0$ we obtain a first order flux function and when $\Phi(\theta) = 1$ we obtain a second order flux function. When $\Phi(\theta)$ has a value somewhere in between zero and one we obtain some composite flux function. We now need to make a choice for the flux limiter $\Phi(\theta)$.

It was shown by Sweby [87] that the flux limiter $\Phi(\theta)$ should have certain properties to ensure that the resulting method is total variation diminishing, that is the total variation in the numerical solution is not increasing in time. One choice of $\Phi(\theta)$ that we will use in our numerical calculations is the superbee flux limiter of Roe [81] which is given by

$$\Phi(\theta) = \max(0, \min(1, 2\theta), \min(\theta, 2))$$

and takes the greatest possible amount of flux up to the point where the ratio of consecutive wave strengths is $\frac{1}{2}$ or 2. If we use the Lax-Wendroff flux as the high-order flux function in (5.15), given as

$$\underline{\mathrm{F}}_{High}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) = \frac{1}{2}\left(\bar{\underline{\mathrm{f}}}(\underline{\mathrm{u}}_L) + \bar{\underline{\mathrm{f}}}(\underline{\mathrm{u}}_R)\right) - \frac{1}{2}\frac{\Delta t^n}{\Delta x_i^n}\sum_{p=1}^{m}\tilde{\lambda}_p^2\,\alpha_p\,\underline{\mathrm{r}}_p,$$

then can write our limited flux function as

$$\underline{\mathrm{F}}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) = \underline{\mathrm{F}}_{Low}(\underline{\mathrm{u}}_L, \underline{\mathrm{u}}_R) + \frac{1}{2}\sum_{p=1}^{m} |\tilde{\lambda}_p| \left(1 - \frac{\Delta t^n}{\Delta x_i^n}|\tilde{\lambda}_p|\right) \Phi(\theta_p)\, \alpha_p\, \underline{\mathrm{r}}_p, \tag{5.16}$$

where $\underline{\mathrm{F}}_{Low}$ is the first order Roe flux function given by (5.14) and $\Delta x_i = \frac{1}{2}(\Delta x_{i+\frac{1}{2}} + \Delta x_{i-\frac{1}{2}})$. The variable θ_p is given by the ratio of consecutive wave strengths to the left if $\tilde{\lambda}_p$ is greater than zero and to the right if $\tilde{\lambda}_p$ is less than zero.

Since the Roe scheme is explicit in time we also require that the mesh and time-step satisfy a CFL condition. This condition is given by

$$CFL = \Delta t \max_{i,p} \left\{ \left| \frac{\left(\tilde{\lambda}_p^+\right)_i}{\Delta x_{i+\frac{1}{2}}} \right|, \left| \frac{\left(\tilde{\lambda}_p^-\right)_{i+1}}{\Delta x_{i+\frac{1}{2}}} \right| \right\} \leq 1 \tag{5.17}$$

and constrains the time-step to be small enough that neighbouring waves do not intersect.

It has been found that Roe's Riemann solver does not satisfy an entropy condition and so can produce non-physical solutions. This entropy violating solution may arise if a sonic rarefaction occurs in the solution relative to the motion of the mesh. In this case we will have that $\tilde{\lambda}_p < 0$ to the left of the wave and $\tilde{\lambda_p} > 0$ to the right of the wave. It is therefore essential to modify the approximate Riemann solver when we have this case. One method for doing this on a moving mesh is outlined by Harten and Hyman in [43] and consists of modifying the wave speeds to obtain an entropy satisfying solution.

We will now consider a particular example of a hyperbolic conservation law that the moving mesh method can be applied to. The equation that will be solved is a first order scalar partial differential equation.

5.4 Inviscid Burgers' Equation

The inviscid Burgers equation in one spatial dimension has the form

$$u_t + \left(\frac{1}{2}\,u^2\right)_x = 0. \tag{5.18}$$

This equation may also be written in quasi-linear form as

$$u_t + u\,u_x = 0$$

The inviscid Burgers equation is perhaps the simplest conservation law which exhibits non-linear effects and is often studied to illustrate the structure of solutions. The equation is derived from the advection-diffusion equation

$$u_t + u\,u_x = \epsilon\,u_{xx}, \tag{5.19}$$

when $\epsilon \to 0$. Equation (5.19) is the simplest equation to include both non-linear and viscous effects (and can be reduced to the linear heat equation by a Cole-Hopf transformation as shown in Whitham in [100]). The case of vanishing viscosity ($\epsilon \to 0$) is considered for the solution of (5.18) in [100].

We now consider the scale invariance of the inviscid Burgers equation. We will seek a set of transformations of the form

$$\hat{u} \to \lambda^\alpha\,u \qquad \hat{x} \to \lambda^\beta\,x \qquad \hat{t} \to \lambda\,t$$

which leave the inviscid Burgers equation, given by (5.18), invariant. Substituting these transformations into the inviscid Burgers equation given by (5.18) we obtain

$$\lambda^{1-\alpha}\,\hat{u}_{\hat{t}} + \lambda^{\beta-2\,\alpha}\,\left(\frac{1}{2}\,\hat{u}^2\right)_{\hat{x}} = 0.$$

Therefore the inviscid Burgers equation will be scale invariant under the transformation $(u, x, t) \to \left(\hat{u}, \hat{x}, \hat{t}\right)$ provided that

$$1 - \alpha = \beta - 2\,\alpha,$$

or

$$\alpha - \beta + 1 = 0$$

holds. A further condition is also required to uniquely determine the transformation; one suitable condition comes from imposing the Dirichlet boundary conditions $u(a,t) = A$ and $u(b,t) = B$ where a and b are the end points of the spatial domain and A and B are constants. These boundary conditions can be written as

$$\int_a^b u_x \,\mathrm{d}x = u\,(b) - u\,(a) = B - A = \textit{constant}.$$

Therefore we shall require that the total jump in the variable u should be constant in time and hold in the transformed co-ordinates $\left(\hat{u}, \hat{x}, \hat{t}\right)$. Substituting the change of variables into this condition we obtain

$$\int_a^b u_x \,\mathrm{d}x = \lambda^{-\alpha} \int_a^b \hat{u}_{\hat{x}} \,\mathrm{d}\hat{x},$$

which is scale invariant provided that $\alpha = 0$. Therefore we have found that the inviscid Burgers equation given by (5.18), together with Dirichlet boundary conditions, is scale invariant under the transformation

$$\hat{u} \to u \qquad \hat{x} \to \lambda\, x \qquad \hat{t} \to \lambda\, t.$$

From these scale invariant arguments it can be seen that there are two invariant quantities in this problem, given as u and $\frac{x}{t}$ and hence self-similar solutions may be sought of the form $u = f\left(\frac{x}{t}\right)$. Also, these arguments suggest that the monitor function $M = u_x$ may be used for the solution of the inviscid Burgers equation. However, the gradient of the solution is likely to be equal to zero in parts of the domain and therefore is unsuitable for use as a monitor function. Hence the modified monitor function $M = 1 + \alpha\,|u_x|$ is used here so that in regions where the gradient of the solution is zero, the mesh will remain Eulerian.

5.4.1 Details of an ALE Solver

In this subsection we briefly describe the application of Roe's approximate Riemann solver for the solution of the ALE form of the inviscid Burgers equation, which is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} u\,\mathrm{d}x + \int_{\Omega(t)} \frac{\partial}{\partial x}\left(\frac{1}{2}\,u^2 - u\,\dot{x}\right)\,\mathrm{d}x = 0$$

To be able to use the Roe approximate Riemann solver outlined in section 5.3 we need to obtain the eigenvalues and eigenvectors of the Jacobian matrix for the equations flux function. However, since the equation being solved is a scalar system, the single eigenvalue of the equation is $\lambda = u$. In the case of the Riemann problem the eigenvalue is calculated by considering the Rankine-Hugoniot jump condition for the equation. Therefore

$$\lambda = \frac{[f\,(u)]}{[u]},$$

where $[\cdot] = \cdot_R - \cdot_L$ denotes the jump in the variable. Then since the flux function for the inviscid Burgers equation is $f\,(u) = \frac{1}{2}\,u^2$ we have that

$$\lambda = \frac{\left[\frac{1}{2}\,u_R^2 - \frac{1}{2}\,u_L^2\right]}{[u_R - u_L]} = \frac{1}{2}\,(u_R + u_L)\,.$$

From equation (5.13) we have that the jump in the variable u is given simply by

$$\Delta u = u_R - u_L$$

and replaces the product of the wave strength and eigenvector in the numerical flux function given by (5.16). Therefore the ALE finite volume equation derived in section 5.3 has a much simpler form for the inviscid Burgers equation and is given by

$$u_{i+\frac{1}{2}}^{n+1} = \left(\frac{\Delta x_{i+\frac{1}{2}}^{n}}{\Delta x_{i+\frac{1}{2}}^{n+1}}\right) u_{i+\frac{1}{2}}^{n} - \frac{\Delta t^n}{\Delta x_{i+\frac{1}{2}}^{n+1}}\left[\bar{F}_{i+1}\left(u_{i+\frac{1}{2}}, u_{i+\frac{3}{2}}\right) - \bar{F}_i\left(u_{i-\frac{1}{2}}, u_{i+\frac{1}{2}}\right)\right],$$

where the limited flux function (5.16) is given by

$$\bar{F}(u_L, u_R) = \frac{1}{2}\left(\bar{f}(u_L) + \bar{f}(u_R)\right) - \frac{1}{2}|\tilde{\lambda}|\left(1 - \left(1 - \frac{\Delta t^n}{\Delta x_i^n}|\tilde{\lambda}|\right)\Phi(\theta)\right)\Delta u.$$

Here $\bar{f} = \frac{1}{2}u^2 - u\dot{x}$ is the ALE flux function and $\tilde{\lambda} = \lambda - \dot{x}_i$ is the shifted eigenvalue for the inviscid Burgers equation.

In subsections 5.4.2 and 5.4.3 we will outline the use of two different monitor functions to produce an ALE velocity for the solution of the one-dimensional inviscid Burgers equation by the method above. The results obtained using the different monitor functions will then be compared with each other and with the traditional Eulerian method, which can be seen as arising from the monitor function $M = 1$. The first monitor function that we will consider for the adaptive solution of the inviscid Burgers equation is the mass monitor function.

5.4.2 A Mass Monitor Function

In this section the moving mesh method derived in chapter 3, along with the finite volume solver outlined in the previous section, will be used to solve the inviscid Burgers equation in one spatial dimension. The first choice for the monitor function is the "mass" monitor which is defined as $M = u$. This choice leads to the mesh movement being approximately Lagrangian. Therefore the nodes will move approximately with the local fluid velocity $\frac{1}{2}u$. This can be shown by considering the continuous version of the moving mesh method derived in chapter 3 with the monitor function $M = u$. Thus, from (3.4),

$$\int \frac{\partial}{\partial x}\left(u\,\dot{x}\right)\,\mathrm{d}x = -\int u_t\,\mathrm{d}x,$$

which upon using the inviscid Burgers equation (5.18) and rearranging becomes

$$\int \frac{\partial}{\partial x}\left(u\,\dot{x} - \frac{1}{2}\,u^2\right)\,\mathrm{d}x = 0.$$

Therefore, simply integrating and using the condition that $u = 0$ when $\dot{x} = 0$, we obtain

$$u\,\dot{x} - \frac{1}{2}\,u^2 = 0,$$

which, using a limiting argument, finally gives us that $\dot{x} = \frac{1}{2}\,u$. We will now outline the use of the mass monitor function in the moving mesh method. Substituting this choice for the monitor function into equation (3.17) gives

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(u\,\frac{\partial \phi}{\partial x}\right)\,\mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,u_t\,\mathrm{d}x.$$

The right hand side of this equation can be written in terms of the PDE being solved, which in this case is Burgers' equation. Therefore, substituting equation (5.18), we obtain

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\left(u\,\frac{\partial \phi}{\partial x}\right)\,\mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\left(\frac{1}{2}\,u^2\right)\,\mathrm{d}x$$

To obtain a weak form of the velocity potential equation we can integrate the left-hand side by parts to remove one of the differential operators from ϕ to give

$$\left[w_i\,u\frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\,u\,\frac{\partial \phi}{\partial x}\,\mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\left(\frac{1}{2}\,u^2\right)\,\mathrm{d}x.$$

The basis functions w_i are equal to zero at the extremities of the integral and at the edge of the domain $\phi = 0$, so the first term disappears to give

$$-\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\,u\,\frac{\partial \phi}{\partial x}\,\mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\left(\frac{1}{2}\,u^2\right)\,\mathrm{d}x$$

for internal nodes. The finite element approximations can now be substituted and expanded in terms of the linear basis functions to obtain a discrete numerical method. The result of this is a weighted stiffness matrix system given by

$$K\,\underset{\sim}{\Phi} = \underset{\sim}{f}, \tag{5.20}$$

where

$$K_{i\,j} = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\,U\,\frac{\partial w_j}{\partial x}\,\mathrm{d}x$$

and

$$f_i = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\frac{1}{2}U^2\right) \mathrm{d}x.$$

Once the numerical approximation to the mesh velocity potential has been obtained, the mesh velocity can then be recovered weakly, as in chapter 4 for the PME, through using equation (4.19). The mesh positions are then integrated forward in time using the standard forward Euler time-stepping routine given as

$$\underset{\sim}{X}^{n+1} = \underset{\sim}{X}^{n} + \Delta t^n \, \dot{\underset{\sim}{X}}^{n}, \tag{5.21}$$

where a suitable time-step Δt^n is calculated through (5.17). The velocity obtained from this choice of the monitor function can then be used in the ALE integral form of Burgers' equation to update the solution variable. The solution procedure will be highlighted at the end of the next subsection and results obtained using this monitor function will be shown in the subsequent section.

5.4.3 A Geometric Monitor Function

The next monitor function that will be considered is given by $M = 1 + \alpha\,|\frac{\partial u}{\partial x}|$, where α is a constant. This monitor function is chosen from more geometrical considerations of the solution structure. Since non-linear hyperbolic equations exhibit solutions which contain regions where the solution gradient is very high, such as close to shock waves, we would hope that a monitor function based on the gradient of the solution would move mesh points into these regions and hence give better resolution of these features. We will assume from now on that $\frac{\partial u}{\partial x} > 0$ (or $\frac{\partial u}{\partial x} < 0$) everywhere. The presence of the 1 is because in regions of the domain the gradient u_x will be equal to zero and hence the matrices being solved will become singular. Therefore we require that when this occurs the mesh movement should be Eulerian in these regions. The constant α is included so that the relative amount of mesh adaption can be increased/decreased by increasing/decreasing this constant. Also it should be noted that in practice the monitor function will be smoothed to obtain a more regularised mesh, however this process will be described in the numerical results section.

Substituting this geometric monitor function into equation (3.17) gives

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\left(1+\alpha\frac{\partial u}{\partial x}\right)\frac{\partial \phi}{\partial x}\right)\mathrm{d}x = -\alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right)\mathrm{d}x. \tag{5.22}$$

We now need to deal with the right hand side of the above integral and in particular the term $\frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right)$. If the derivatives exist we can interchange the order of space and time differentiation in the right hand side integrand of this equation. Then we may use the inviscid Burgers equation given by (5.18) to obtain

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right)\mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial^2}{\partial x^2}\left(\frac{1}{2}u^2\right)\mathrm{d}x,$$

giving

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\left(1+\alpha\frac{\partial u}{\partial x}\right)\frac{\partial \phi}{\partial x}\right)\mathrm{d}x = \alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial^2}{\partial x^2}\left(\frac{1}{2}u^2\right)\mathrm{d}x.$$

To obtain a suitable weak form of this velocity potential equation we need to integrate the right and left hand sides of the above integral equation by parts to obtain

$$\begin{aligned}\left[w_i\left(1+\alpha\frac{\partial u}{\partial x}\right)\frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} &- \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\left(1+\alpha\frac{\partial u}{\partial x}\right)\frac{\partial \phi}{\partial x}\,\mathrm{d}x \\ &= \left[\alpha\, w_i \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right)\mathrm{d}x.\end{aligned}$$

Again, standard linear finite element approximations are substituted and expanded in terms of the linear basis functions to obtain a discrete numerical method. The condition that $\phi = 0$ on the boundary of the domain is also used. The result of this discretisation is a weighted stiffness matrix system given by

$$K\,\underset{\sim}{\Phi} = \underset{\sim}{f}, \tag{5.23}$$

where

$$K_{i\,j} = \left[w_i\left(1+\alpha\frac{\partial U}{\partial x}\right)\frac{\partial w_j}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\left(1+\alpha\frac{\partial U}{\partial x}\right)\frac{\partial w_j}{\partial x}\,\mathrm{d}x$$

and

$$f_i = \left[\alpha\, w_i \frac{\partial}{\partial x}\left(\frac{1}{2}U^2\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \alpha \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(\frac{1}{2}U^2\right) \mathrm{d}x.$$

The mesh velocity is recovered weakly using (4.19). The mesh is then integrated forward in time with the standard forward Euler time-stepping routine (5.21) and the solution to the inviscid Burgers equation is updated with the Roe scheme on a moving mesh.

The outline of the numerical method is therefore:

- Given an initial mesh and solution, calculate a suitable time-step using equation (5.17).
- Solve the stiffness matrix system (5.20) or (5.23), depending on which monitor function is being used, for the mesh velocity potential Φ using the boundary condition $\Phi = 0$.
- Recover the mesh velocity $\dot{X}$ using (4.19) and time-step the mesh using the forward Euler time stepping procedure given by (5.21).
- Obtain the solution U on the new mesh by solving the ALE form of the inviscid Burgers equation[1].
- Repeat until the desired output time is reached.

Results will be shown for the adaptive solution of Burgers' equation in the next section.

[1]If the solution to the problem is smooth then the solution may be recovered using the conservation of monitor function principle as in chapter 4 for the solution of the porous medium equation.

5.5 Numerical Results IV

In this section we consider the solution to the inviscid Burgers equation on an adaptively moving mesh generated from the moving mesh method derived in chapter 3. The solution has been obtained for two different test cases which have been chosen to demonstrate the features of the equation. The first problem that we will solve has the initial condition

$$u_0(x) = \begin{cases} 1 & \text{for } 0 \leq x \leq \frac{1}{10}, \\ \frac{1}{6}\,(7 - 10\,x) & \text{for } \frac{1}{10} \leq x \leq \frac{2}{5}, \\ \frac{1}{2} & \text{for } \frac{2}{5} < x < 1, \end{cases} \tag{5.24}$$

which is a continuous ramp-type function and demonstrates the phenomenon that shocks can arise from smooth initial data. This problem has the solution

$$u(x,t) = \begin{cases} 1 & \text{for } 0 \leq x \leq \frac{1}{10} + t, \\ \frac{7-10\,x}{6-10\,t} & \text{for } \frac{1}{10} + t \leq x \leq \frac{2}{5} + \frac{1}{2}\,t, \\ \frac{1}{2} & \text{for } \frac{2}{5} + \frac{1}{2}\,t < x < 1, \end{cases} \tag{5.25}$$

for $0 < t < t^*$, where $t^* = \frac{3}{5}$. At $t = t^*$ a shock forms and the solution to the weak form of the problem is a shock wave given by

$$u(x,t) = \begin{cases} 1 & \text{for } x \leq \frac{7}{10} + \frac{3}{4}\,(t - t^*), \\ \frac{1}{2} & \text{for } x > \frac{7}{10} + \frac{3}{4}\,(t - t^*), \end{cases} \tag{5.26}$$

for $t \geq t^*$. In all the numerical simulations for this test case we will integrate the problem forward in time until $t = 0.7$ so that the shock has formed. The second test case consists of initial data which is already discontinuous, with initial condition

$$u_0(x) = \begin{cases} 1 & \text{for } 0 \leq x \leq \frac{1}{2} \\ \frac{1}{2} & \text{for } \frac{1}{2} < x < 1 \end{cases} \tag{5.27}$$

The solution to the inviscid Burgers equation with this initial condition can be found simply using the Rankine-Hugoniot jump condition and is given as

$$u(x,t) = \begin{cases} 1 & \text{for } 0 \leq x \leq \frac{1}{2} + \frac{3}{4}\,t \\ \frac{1}{2} & \text{for } \frac{1}{2} + \frac{3}{4}\,t < x < 1 \end{cases} \quad . \tag{5.28}$$

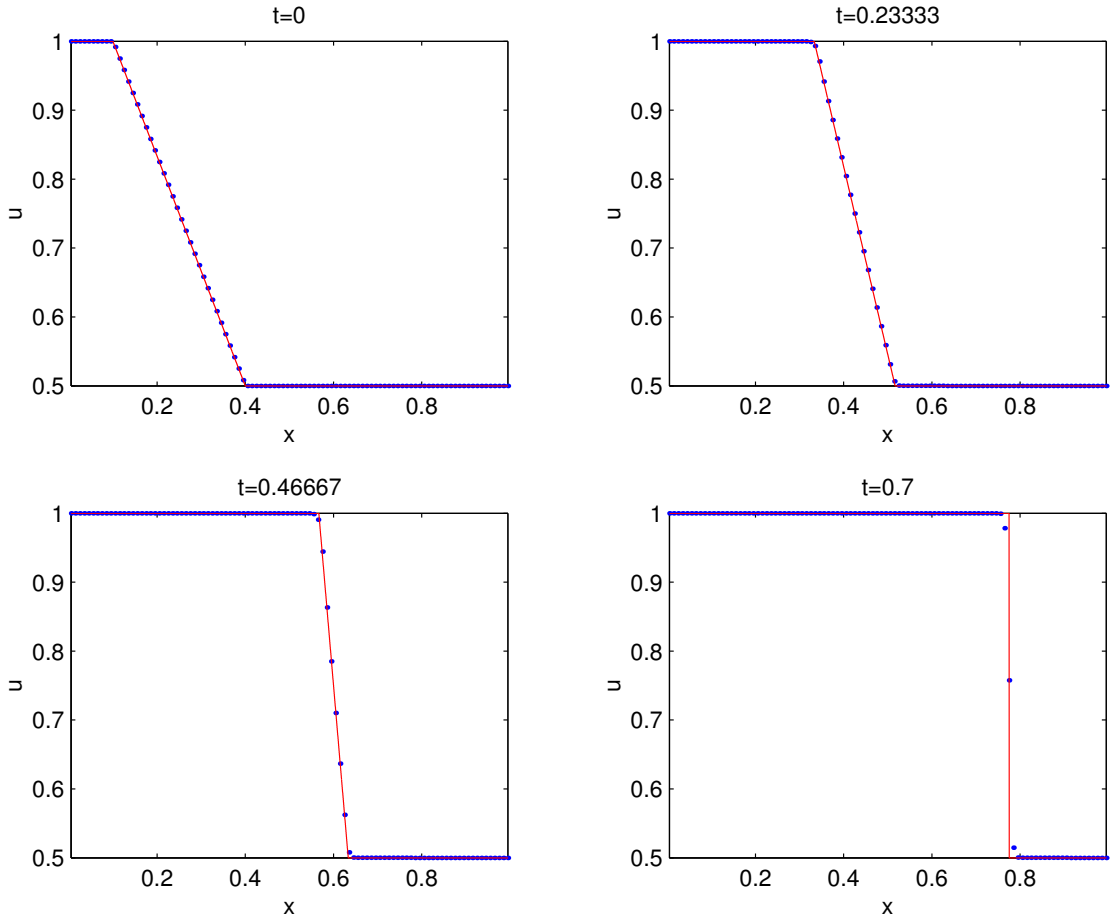

Figure 5.6: Eulerian solution to Burgers' equation at $t = 0.7$ with initial condition (5.24), computed on a mesh with 100 mesh points and $CFL = 0.5$. The bold red line and the blue dotted line represent exact and numerical solutions respectively.

In the numerical simulations for this test case we will integrate the problem forward in time until $t = 0.6$ and transmissive boundary conditions will be used for the ALE solver.

These two test problems were solved on an adaptive mesh generated by the moving mesh method using each monitor function. However, the test problems were first solved on a fixed equispaced Eulerian mesh (as generated from the monitor function $M = 1$). The problems were solved using four different sized meshes having 50, 100, 200 and 400 mesh points. The solution obtained using 100 mesh points for the first test problem can be seen in figure 5.6 and results for the second in figure 5.7. These solutions were computed with a CFL limit of 0.5 and used the superbee flux limiter.

The two test problems were then solved on a moving mesh generated from the monitor function $M = u$, which led to approximately Lagrangian mesh movement. The problem

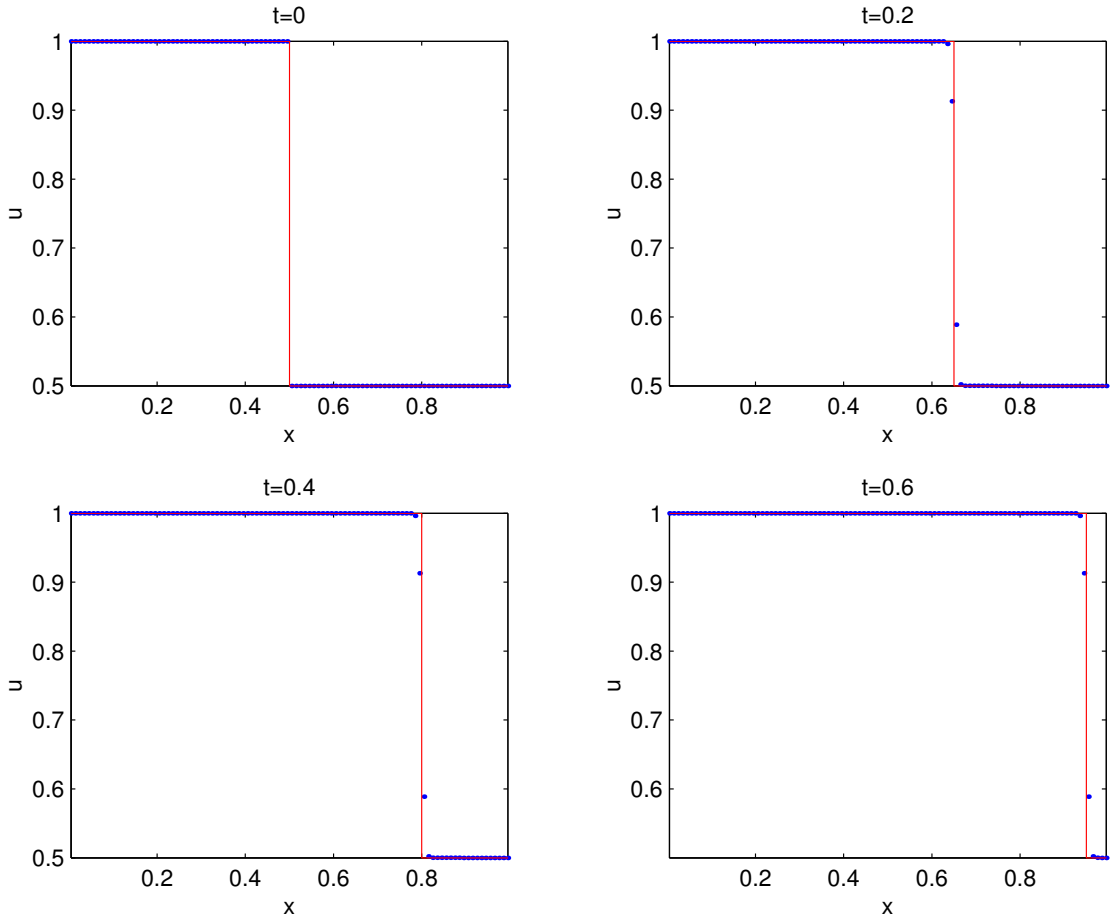

Figure 5.7: Eulerian solution to Burgers' equation at $t = 0.6$ with initial condition (5.27), computed on a mesh with 100 mesh points and $CFL = 0.5$. The bold red line and the blue dotted line represent exact and numerical solutions respectively.

was solved on initially equispaced meshes comprising of 50, 100 and 200 mesh points. The solutions obtained on the mesh having 100 points can be seen in figures 5.10 and 5.8 for the first and second test problems respectively. Again the solutions were computed with a CFL limit of 0.5 and the superbee flux limiter was employed. The mesh velocity and the trajectories of the mesh points can be seen in figures 5.9 and 5.11 again for the two test problems. It can easily be seen that the mesh movement is approximately Lagrangian since the mesh velocity at the relevant output times is approximately $\dot{x} = \frac{1}{2}u$. The Lagrangian mesh movement tends to compress the mesh points into the region of the moving discontinuity and hence gives better resolution of it, which can be seen in the mesh point trajectories in figures 5.11 and 5.9.

Lastly, the problems were solved using the monitor function $M = 1 + \alpha|u_x|$, where α is a constant. The constant α was normalised by the maximum gradient by setting

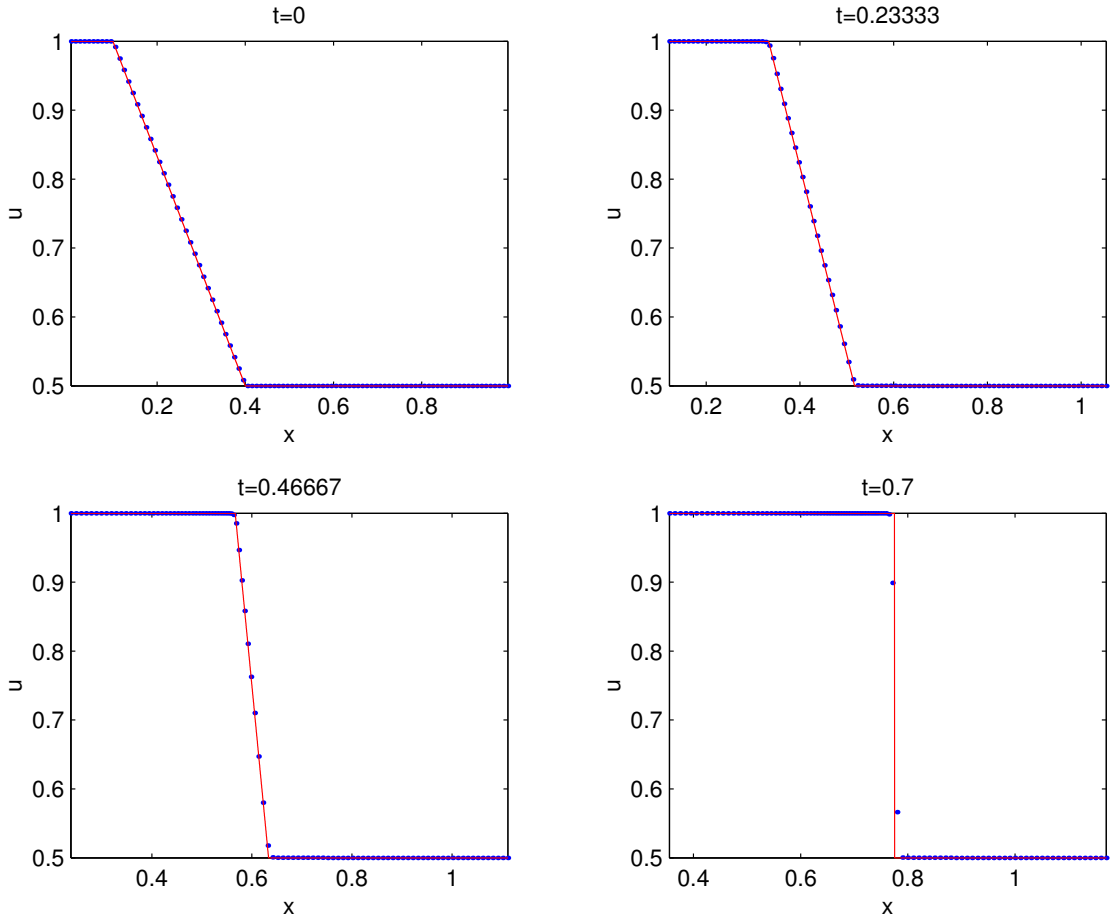

Figure 5.8: Solution to Burgers' equation at $t = 0.7$ *with initial condition (5.24) obtained with the moving mesh method with* $M = u$*, computed on a mesh with 100 mesh points and* $CFL = 0.5$*. The bold red line and the blue dotted line represent exact and numerical solutions respectively.*

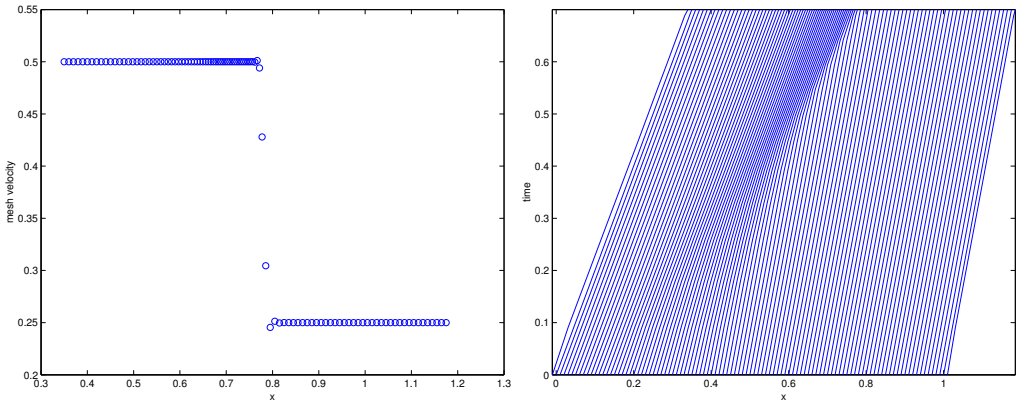

Figure 5.9: Figures showing the velocity of the mesh points at the output time $t = 0.7$ *(left) and the trajectories of the mesh points (right) arising from the use of the monitor function* $M = u$*.*

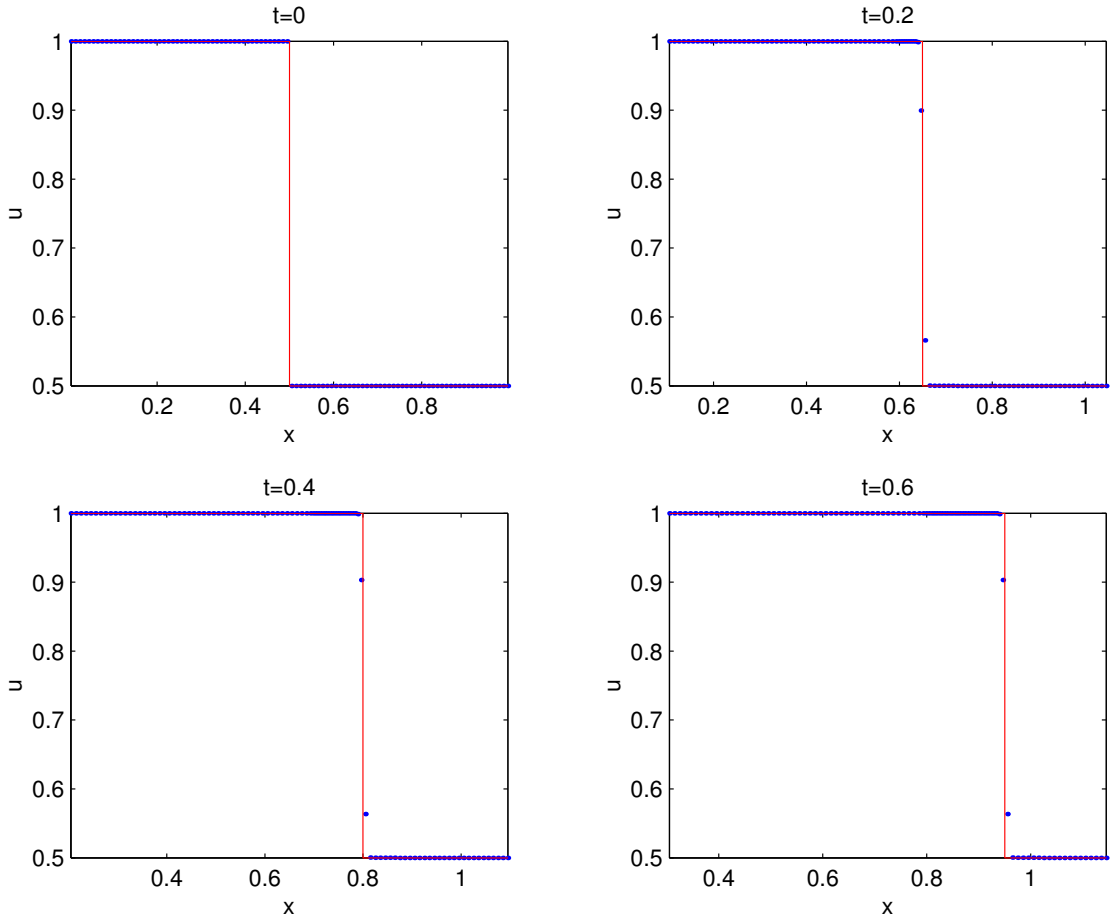

Figure 5.10: Solution to Burgers' equation at $t = 0.6$ with initial condition (5.27) obtained with the moving mesh method with $M = u$, computed on a mesh with 100 mesh points and $CFL = 0.5$. The bold red line and the blue dotted line represent exact and numerical solutions respectively.

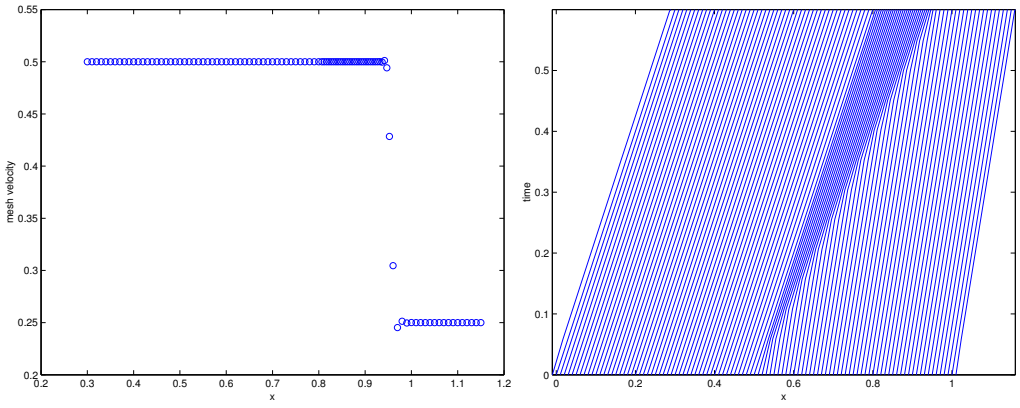

Figure 5.11: Figures showing the velocity of the mesh points at the output time $t = 0.6$ (left) and the trajectories of the mesh points (right) arising from the use of the monitor function $M = u$.

$$\alpha = \frac{\beta}{\max\limits_{x} |u_x|},$$

where β is a constant chosen to determine the degree of mesh adaption. The problems were solved on meshes comprising of 50, 100 and 200 mesh points. An initial mesh was generated for the first test problem with the initial ramp function by equidistributing the monitor function over the mesh and the initial mesh for the second test problem with the initial discontinuous data given by (5.27) was produced by replacing it by a hyperbolic tangent function. This function was then used to equidistribute the monitor function over the mesh. Both the initial meshes were equidistributed using the algorithm found in Baines [7]. It should be noted that the initial data given by (5.27), not the hyperbolic tangent function, was then used on the adapted mesh as the initial setup for the second test problem. To maintain accuracy of the numerical solution on the adaptive mesh it is usually necessary to smooth either the mesh or the monitor function. However, since the monitor function is solution dependent it is possible to merely smooth the solution variable u. This was done by diffusing the solution in the following manner,

$$(u_i)_{smoothed} = u_i + \frac{\left(\frac{u_{i+1}-u_i}{x_{i+1}-x_i} - \frac{u_i-u_{i-1}}{x_i-x_{i-1}}\right)}{(x_{i+1}-x_{i-1})}.$$

This smoothing process can be applied as many times as required to obtain a smoothed monitor to be used to generate the moving mesh. The solutions obtained on the mesh having 100 points can be seen in figures 5.14 and 5.12. Again the solution was computed with a CFL limit of 0.5 and the superbee flux limiter was employed. The mesh velocity and the mesh trajectories can be seen in figure 5.15 for this choice of monitor function. It can be seen from these figures that this monitor function causes mesh points to cluster around the shock region and hence give better resolution of the solution.

The errors in the resulting numerical solution were computed for all three methods using a weighted L_1-norm given as

$$||u - U||_{L_1} = \sum_{i=0}^{N} \left| u(x_{i+\frac{1}{2}}, t) - U(x_{i+\frac{1}{2}}, t) \right| \Delta x_{i+\frac{1}{2}}.$$

The errors in the solutions are summarised in table 5.1 and 5.2 for the two test problems and give a quantitative measure of the accuracy of the method. The errors are shown for

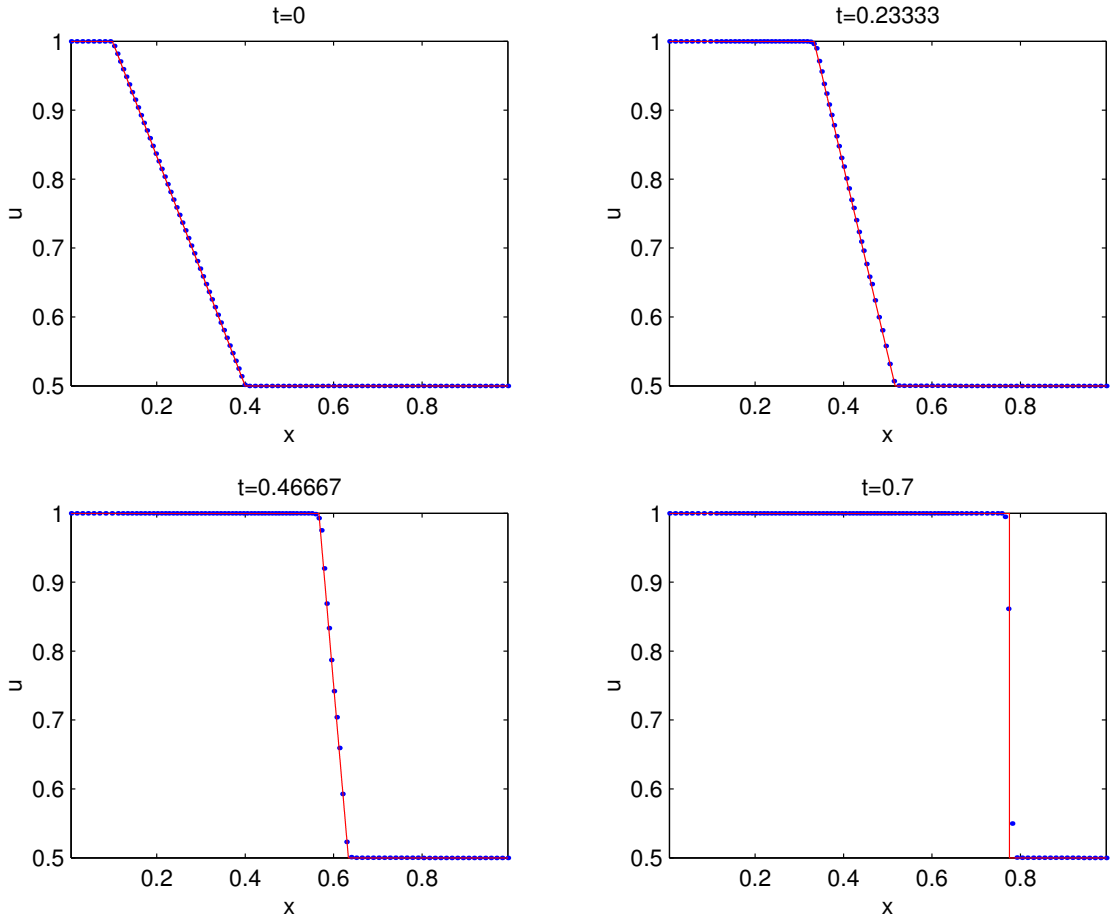

Figure 5.12: Solution to Burgers' equation at $t = 0.7$ with initial condition (5.24) obtained with the moving mesh method with $M = 1 + \alpha\ |u_x|$, computed on a mesh with 100 mesh points and $CFL = 0.5$. The bold red line and the blue dotted line represent exact and numerical solutions respectively.

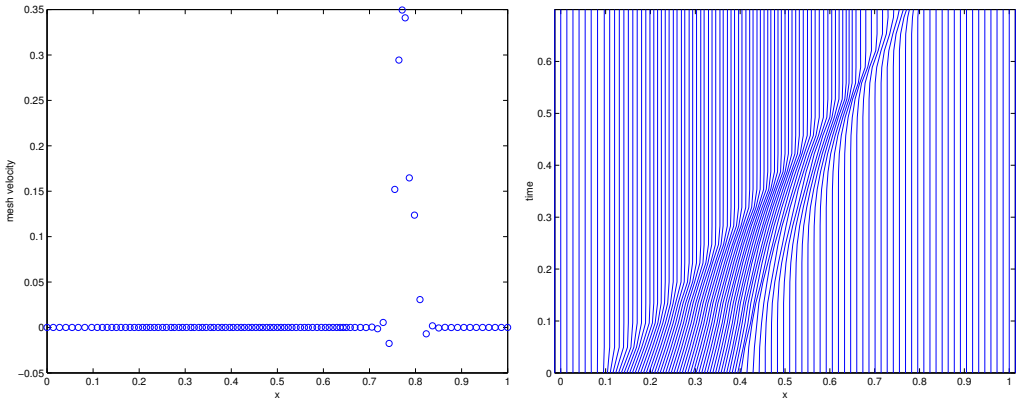

Figure 5.13: Figures showing the velocity of the mesh points at the output time $t = 0.7$ (left) and the trajectories of the mesh points (right) arising from the use of the monitor function $M = 1 + \alpha\ |u_x|$.

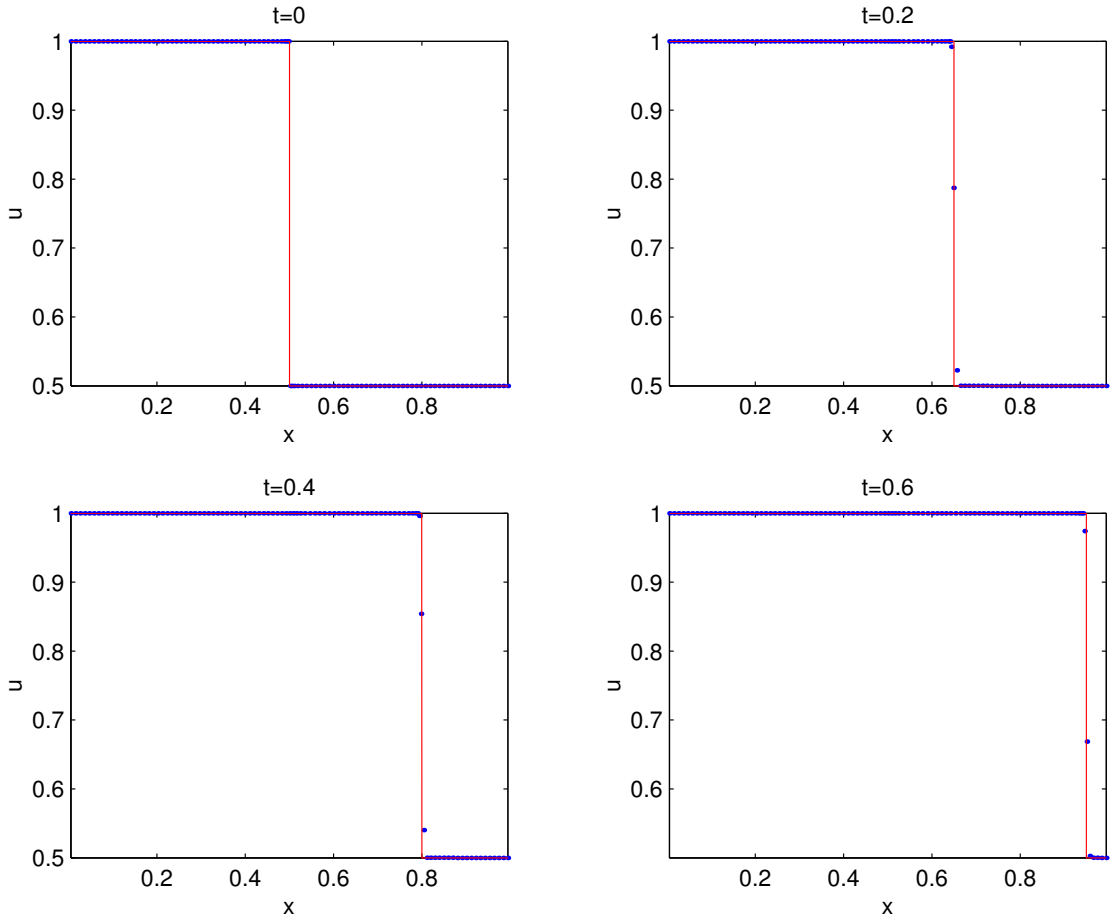

Figure 5.14: Solution to Burgers' equation at $t = 0.6$ with initial condition (5.27) obtained with the moving mesh method with $M = 1+\alpha\ |u_x|$, computed on a mesh with 100 mesh points and $CFL = 0.5$. The bold red line and the blue dotted line represent exact and numerical solutions respectively.

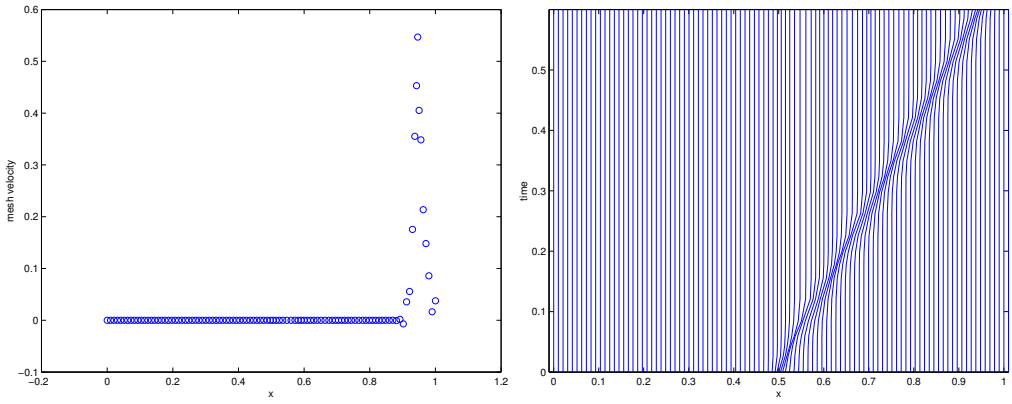

Figure 5.15: Figures showing the velocity of the mesh points at the output time $t = 0.6$ (left) and the trajectories of the mesh points (right) arising from the use of the monitor function $M = 1 + \alpha\ |u_x|$.

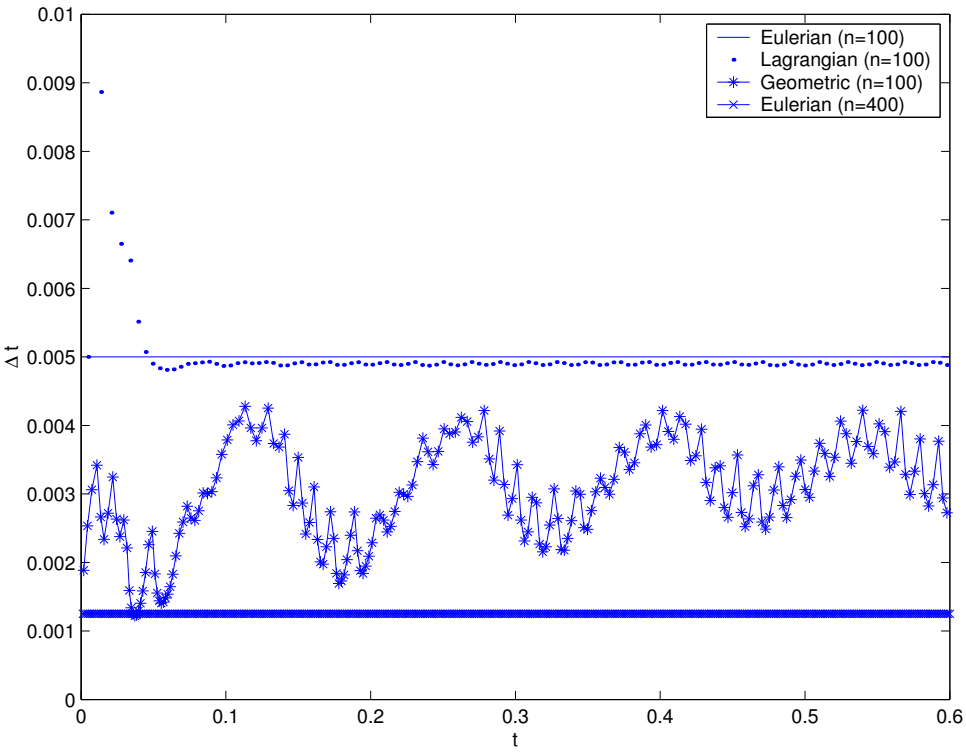

Figure 5.16: Figure showing plots of the time-steps for the Eulerian and moving mesh methods.

both the first and second order in space versions of the Roe method. It can be seen from these tables that the moving mesh methods give rise to much more accurate numerical solutions in comparison with the Eulerian fixed grid method. For example, for the second test problem the moving mesh method using the monitor function $M = 1 + \alpha\,|u_x|$ on a mesh with 200 points can give a more accurate solution in comparison with the Eulerian method calculated on a mesh with 400 mesh points.

Figure 5.16 shows the change in size of the time-step as the simulation with initial data given by (5.27) is carried out. It can be seen that the magnitude of the time-steps, for the moving mesh method with the monitor function $M = 1 + \alpha\,|u_x|$, oscillate in time. This phenomena is caused by the fact that the mesh points cannot be restrained to lie within the shock wave region and therefore periodically enter and leave the shock region. This feature of the mesh movement is usually referred to as mesh racing.

Method	Error (1st order)	Error (2nd order)
Eulerian (Fixed)		
N=50	0.03471	0.00987
N=100	0.02183	0.00551
N=200	0.01199	0.00254
N=400	0.00631	0.00127
Lagrangian $M = u$		
N=50	0.02030	0.00753
N=100	0.01074	0.00377
N=200	0.00538	0.00188
Geometrical $M = 1 + \alpha \lvert u_x \rvert$		
N=50 ($\beta = 0.5$)	0.02403	0.00416
N=100 ($\beta = 0.5$)	0.01294	0.00216
N=200 ($\beta = 0.5$)	0.00692	0.00107

Table 5.1: L_1 errors for fixed and moving mesh methods for inviscid Burgers' equation with initial data (5.24).

Method	Error (1st order)	Error (2nd order)
Eulerian (Fixed)		
N=50	0.02377	0.01391
N=100	0.01644	0.00508
N=200	0.00887	0.00254
N=400	0.00449	0.00127
Lagrangian $M = u$		
N=50	0.01780	0.00755
N=100	0.00893	0.00377
N=200	0.00449	0.00189
Geometrical $M = 1 + \alpha \lvert u_x \rvert$		
N=50 ($\beta = 1$)	0.01592	0.00222
N=100 ($\beta = 1$)	0.00711	0.00185
N=200 ($\beta = 1$)	0.00387	0.00098

Table 5.2: L_1 errors for fixed and moving mesh methods for inviscid Burgers' equation with initial data (5.27).

5.6 The Compressible Euler Equations Of Gas Dynamics

In this section we will consider the adaptive solution of the compressible Euler equations of gas dynamics in one spatial dimension. The Euler equations in one spatial dimension are a set of three time-dependent hyperbolic conservation laws which govern the motion of a compressible, inviscid fluid. The Euler equations in differential form are given as

$$\begin{pmatrix} \rho \\ \rho v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho v \\ \rho v^2 + p \\ v\ (E+p) \end{pmatrix}_x = \underline{0}, \tag{5.29}$$

where ρ is the density, v is the velocity, p is the pressure and E is the total energy of the fluid being modelled. These three equations represent the conservation of mass, momentum and energy. The Euler equations are a simplification of the more general compressible Navier-Stokes equations of fluid dynamics, which model fluids with viscous effects. The Euler equations are obtained by ignoring these effects, rather as $\epsilon \to 0$ in the viscous Burgers equation of the previous section.

Since the Euler equations are three equations in four unknowns we need another equation to close the system. The system is usually closed by prescribing an empirical law called the equation of state (EOS) which attempts to describe the thermodynamic relationship between the variables. In this work we will use the ideal gas equation of state so that the total energy can be written as

$$E = \frac{p}{(\gamma - 1)} + \frac{1}{2}\rho u^2,$$

where γ is the ratio of specific heats for the gas, and in this work will be taken to be $\gamma = 1.4$.

To be able to use the Roe scheme outlined in section 5.3 for the solution of the compressible Euler equations, given by (5.29), we need to find the eigenvectors and eigenvalues of the Jacobian matrix for the flux function in the Euler equations. The Euler equations can be written in quasi-linear form as

$$\underline{\mathrm{u}}_t + A\,\underline{\mathrm{u}}_x = 0,$$

where $\underline{\mathrm{u}} = (\rho, \rho\,v, E)^T$ and the Jacobian matrix A is given as

$$A = \begin{pmatrix} 0 & 1 & 0 \\ (\gamma-1) - v^2 - c^2 & (3-\gamma)\;v & \gamma - 1 \\ \frac{1}{2}\;[(\gamma-3)\;H - c^2] & H - (\gamma-1)\;v^2 & \gamma\,v \end{pmatrix}.$$

Here H is defined as the enthalpy of the gas and is given as

$$H = \frac{E+p}{\rho} = \frac{c^2}{\gamma-1} + \frac{1}{2}\,v^2$$

and $c = \sqrt{\frac{\gamma\,p}{\rho}}$ is the speed of sound in the gas. It can be shown that the eigenvalues λ_p for the Jacobian matrix A are

$$\lambda_1 = v - c \qquad \lambda_2 = v \qquad \lambda_3 = v + c$$

and the right eigenvectors for the system are

$$\underline{\mathrm{r}}_1 = \begin{pmatrix} 1 \\ v-c \\ H - v\,c \end{pmatrix} \qquad \underline{\mathrm{r}}_2 = \begin{pmatrix} 1 \\ v \\ \frac{1}{2}\,v^2 \end{pmatrix} \qquad \underline{\mathrm{r}}_3 = \begin{pmatrix} 1 \\ v+c \\ H + v\,c \end{pmatrix}.$$

The wave-strengths α_p are also needed for the solution of the Euler equations using the Roe approximate Riemann solver and are obtained by solving the set of algebraic equations that result from (5.13). Once these equations have been solved it is found that the wave-strengths in a cell are given by

$$\begin{aligned} \alpha_1 &= \frac{1}{2\,c}\left(\Delta\rho\,(v+c) - \Delta\,(\rho\,v) - c\,\alpha_2\right), \\ \alpha_2 &= \frac{\gamma-1}{c^2}\left(\Delta\rho\left(H - v^2\right) - v\,\Delta\,(\rho\,v) - \Delta E\right), \\ \alpha_3 &= \Delta\rho - (\alpha_1 + \alpha_2)\,. \end{aligned}$$

where $\Delta(\cdot)$ denotes the jump in the corresponding conserved variable. To obtain a conservative discrete numerical method, the eigenvalues, eigenvectors and the wave-strengths for the system need to be evaluated at a consistent cell-averaged state. In [80] Roe found that a sufficient condition to satisfy the three properties (5.10), (5.11) and (5.12) was that the Roe matrix should be evaluated at the Roe-averaged states given by

$$\hat{\rho} = \sqrt{\rho_L\,\rho_R}, \qquad \hat{v} = \frac{\sqrt{\rho_L}\,v_L + \sqrt{\rho_R}\,v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \qquad \hat{H} = \frac{\sqrt{\rho_L}\,H_L + \sqrt{\rho_R}\,H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}.$$

Here the subscripts L and R denote the left and right states at the cell boundary. The sound speed is also evaluated at this Roe-averaged state and is given by

$$\hat{c}^2 = (\gamma - 1)\left(\hat{H} - \frac{1}{2}\,\hat{v}^2\right).$$

We now have all the information that is required to apply the Roe Riemann solver for the solution of the Euler equations. In the following subsections we will outline the application of the moving mesh method derived in chapter 3 to the Euler equations for specific choices of the monitor function M. We will begin by considering the mass monitor function.

5.6.1 A Mass Monitor Function

In this section we will consider the adaptive solution of the compressible Euler equations given by (5.29). Using similar monitor functions to those which were used for the solution of the inviscid Burgers equation we will solve the Euler equations using the moving mesh method outlined in chapter 3. The first choice for the monitor function is the mass monitor $M = \rho$ and, as with Burgers' equation, this will lead to a moving mesh method which is approximately Lagrangian. Therefore we expect that the computational nodes move with the local fluid velocity, i.e. $\dot{x} = v$. This is easily shown using the same technique as in section 5.4.2. Let us consider the continuous version of the moving mesh method derived in chapter 3 together with the monitor function $M = \rho$, so that

$$\int \frac{\partial}{\partial x}\left(\rho\,\dot{x}\right)\,\mathrm{d}x = -\int \rho_t\,\mathrm{d}x.$$

Now using the conservation of mass equation $\rho_t + (\rho\, v)_x = 0$, given by the first Euler equation in (5.29), in the above equation we obtain

$$\int \frac{\partial}{\partial x} \left(\rho\, (\dot{x} - v)\right)\, \mathrm{d}x = 0,$$

which can be integrated, using the condition that $v = 0$ when $\dot{x} = 0$, to give

$$\rho\, (\dot{x} - v) = 0.$$

Since the density ρ is positive we have that $\dot{x} = v$. Therefore we conclude that this choice for the monitor function will cause the mesh points to move approximately with the fluid velocity.

We now describe the use of the monitor function $M = \rho$ in the moving mesh method. If we substitute this choice for the monitor function into the moving mesh equation (3.17) we obtain the equation

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x} \left(\rho \frac{\partial \phi}{\partial x}\right)\, \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\, \rho_t\, \mathrm{d}x.$$

The right-hand side of this equation can be written in terms of the Euler equations that we are solving by using the conservation of mass equation $\rho_t + (\rho\, v)_x = 0$ to give

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x} \left(\rho \frac{\partial \phi}{\partial x}\right)\, \mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x} \left(\rho\, v\right)\, \mathrm{d}x.$$

We integrate the left-hand side of the above expression by parts to remove one of the differentials from ϕ, giving

$$\left[w_i\, \rho \frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\, \rho \frac{\partial \phi}{\partial x}\, \mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x} \left(\rho\, v\right)\, \mathrm{d}x.$$

Since the basis functions are equal to zero at the end points of the region of integration and also we have the that $\phi = 0$ at the ends of the domain the boundary term in the above integral disappears. Therefore we obtain the velocity potential equation

$$-\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\,\rho\,\frac{\partial \phi}{\partial x}\,\mathrm{d}x = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\,(\rho\,v)\;\mathrm{d}x,$$

for all internal nodes. Once the finite element approximations have been introduced we obtain the weighted stiffness equation

$$K\,\underset{\sim}{\Phi} = \underset{\sim}{f},$$

where

$$K_{i\,j} = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\,\rho\,\frac{\partial w_j}{\partial x}\,\mathrm{d}x$$

and

$$f_i = \int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\,(\rho\,v)\;\mathrm{d}x.$$

To avoid proliferation of notation the finite element approximations to the density and velocity of the gas are still denoted by ρ and v. Again, once the numerical approximation to the mesh velocity potential has been obtained, the mesh velocity can be recovered weakly using equation (4.19). The mesh is then integrated forward in time using a standard forward Euler time-stepping routine with a suitable time-step given by (5.17). The velocity obtained from this choice of the monitor function can then be used in the ALE Godunov method (5.9) to update the solution variables. Results generated from this choice of monitor function will be shown in section 5.7.

5.6.2 A Geometric Monitor Function

In this subsection we will outline the use of the geometric monitor function $M = 1 + \alpha\,|\rho_x|$, where α is a constant to be chosen. We will assume that $\frac{\partial \rho}{\partial x} > 0$ (or $\frac{\partial \rho}{\partial x} < 0$) in the remainder of the section. If we substitute this choice for the monitor function into the moving mesh equation (3.17) we obtain the equation

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial x}\left(\left(1+\alpha\,\frac{\partial \rho}{\partial x}\right)\,\frac{\partial \phi}{\partial x}\right)\,\mathrm{d}x = -\alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i\,\frac{\partial}{\partial t}\left(\frac{\partial \rho}{\partial x}\right)\,\mathrm{d}x. \tag{5.30}$$

We now need to deal with the right hand side of the above integral and in particular the term $\frac{\partial}{\partial t}\left(\frac{\partial \rho}{\partial x}\right)$. This term is dealt with in similar manner as in subsection 5.4.3 for Burgers' equation. Therefore, following a similar procedure, we assume we can interchange the order of differentiation on the right hand side. We now need to obtain an equation for ρ_t. This can be found from the conservation of mass equation given by $\rho_t = -(\rho\, v)_x$. Therefore

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial t}\left(\frac{\partial \rho}{\partial x}\right) \mathrm{d}x = -\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial^2}{\partial x^2}\left(\rho\, v\right) \mathrm{d}x.$$

Hence equation (5.30) becomes

$$\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial}{\partial x}\left(\left(1+\alpha\frac{\partial \rho}{\partial x}\right)\frac{\partial \phi}{\partial x}\right) \mathrm{d}x = \alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} w_i \frac{\partial^2}{\partial x^2}\left(\rho\, v\right) \mathrm{d}x.$$

Integrating by parts to derive a weak form of the velocity potential equation, we obtain

$$\begin{aligned}\left[w_i\left(1+\alpha\frac{\partial \rho}{\partial x}\right)\frac{\partial \phi}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} &- \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\left(1+\alpha\frac{\partial \rho}{\partial x}\right)\frac{\partial \phi}{\partial x}\,\mathrm{d}x \\ &= \left[\alpha\, w_i \frac{\partial}{\partial x}\left(\rho\, v\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(\rho\, v\right) \mathrm{d}x.\end{aligned}$$

Again, finite element approximations for ρ and v can be substituted and expanded in terms of the linear basis functions w_i to obtain a discrete numerical method. The condition that $\phi = 0$ on the boundary of the domain is also used. The result of this is a weighted stiffness matrix system given by

$$K\,\underset{\sim}{\Phi} = \underset{\sim}{f},$$

where the individual entries of the stiffness matrix are given as

$$K_{i\,j} = \left[w_i\left(1+\alpha\frac{\partial \rho}{\partial x}\right)\frac{\partial w_j}{\partial x}\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\left(1+\alpha\frac{\partial \rho}{\partial x}\right)\frac{\partial w_j}{\partial x}\,\mathrm{d}x$$

and the entries of $\underset{\sim}{f}$ are

$$f_i = \left[\alpha\, w_i \frac{\partial}{\partial x}\left(\rho\, v\right)\right]_{x_{i-1}(t)}^{x_{i+1}(t)} - \alpha\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial w_i}{\partial x}\frac{\partial}{\partial x}\left(\rho\, v\right) \mathrm{d}x.$$

The finite element approximations to the density and velocity of the gas are still denoted by ρ and v. We now have a matrix system that can be solved to obtain the velocity potential Φ and from this we can recover the mesh velocity $\dot{X}$ through equation (4.19). The mesh can then be time-stepped using the same standard Euler time-stepping algorithm that was used for the inviscid Burgers equation. The solution to the compressible Euler equations is then advanced in time approximately by using the ALE Godunov scheme with the approximate Roe Riemann solver.

Results for the Euler equations will be presented in the next section, solved on both static and adaptive meshes generated using the monitor functions $M = \rho$ and $M = 1 + \alpha \left| \rho_x \right|$.

5.7 Numerical Results V

In this section we will show results obtained for the compressible Euler equations of gas dynamics using the Roe approximate Riemann solver on an adaptive mesh generated by the method derived in chapter 3. For our test case we have taken the shock tube problem of Sod [85]. This problem has a material interface which separates two regions of gas initially at rest with constant, but different densities, pressures and energies. At $t = 0$ the interface is broken and the gases are allowed to move into one another. The resulting solution of the problem consists of a left travelling rarefaction wave and right travelling contact and shock waves. The initial conditions for this problem are given as

$$(\rho,\, v,\, p) = \begin{cases} (1.0, 0.0, 1.0) & \text{for } 0 \leq x \leq \frac{1}{2} \\ (0.125, 0.0, 0.1) & \text{for } \frac{1}{2} < x < 1 \end{cases} \tag{5.31}$$

In all the numerical simulations we will output the solutions at $t = 0.2$.

The Sod shock tube problem is firstly solved on an Eulerian mesh, that is one which is static in time and equispaced. The problem was solved on four different meshes having different numbers of computational nodes. These meshes had 50, 100, 200 and 400 nodes. The solution to the problem computed on a mesh with 100 nodes can be seen in figure 5.17 and was computed with a CFL limit of 0.5 and the superbee flux limiter.

The same problem was then solved using the moving mesh method using the mass monitor function $M = \rho$. This problem was solved on three meshes comprising of 50, 100 and 200 mesh points. The solution to the problem on the adaptive mesh with 100 nodes is shown in figure 5.18 and was again computed with a CFL limit of 0.5 and used the superbee flux limiter. The velocity of the mesh points at the output time $t = 0.2$ and the trajectories of the mesh points are shown in figure 5.19. It can be seen from these figures that the mesh velocity is very similar to the true fluid velocity and therefore the mesh is approximately moving with the fluid velocity. This type of mesh movement tends to cluster points in regions where the gas is being compressed and expands the mesh in regions where the gas is being rarefied.

The problem was then solved by the moving mesh method using the geometric monitor function $M = 1 + \alpha\,|\rho_x|$ where the constant α was taken to be

$$\alpha = \frac{\beta}{\max\limits_{x} |\rho_x|},$$

where β is a second constant which is selected to obtain the best possible solution error. The moving mesh method with this choice of monitor function was used to solve the Sod shock tube problem on three different meshes comprising of 50, 100 and 200 mesh points. Again, as with the work presented for the solution of the inviscid Burgers' equation, we began the numerical simulation by producing an initial adapted mesh. Since the initial conditions to the Sod shock tube problem are discontinuous we replaced it with a hyperbolic tangent function and then equidistribute the geometric monitor function over it to produce an adapted mesh using the algorithm found in Baines [7]. The problem was then solved using this initial mesh and the initial conditions for the Euler equations given by (5.31). The solution variables were also smoothed as in the inviscid Burgers' section 5.5 to produce a smoothed monitor function. The solution to this problem on the mesh with 100 nodes is shown in figure 5.20 and was again computed with a CFL limit of 0.5 using the superbee flux limiter. The velocity of the mesh points at the output time $t = 0.2$ and the trajectories of the mesh points are shown in figure 5.21. It can be seen from the figure that the velocity of the mesh is very different when compared to the real fluid velocity and tends to move mesh points into regions of the domain where the gradient of the density is large. It can also be seen from these figures that this choice for the monitor function tends to cause the mesh movement to approximately inherit some of the wave structure of the solution.

Again, as with the results that were produced for the inviscid Burgers' equation, we calculated the error in the resulting numerical solution for all three methods using the weighted L_1-norm given as

$$||\underline{\mathrm{u}} - \underline{\mathrm{U}}||_{L_1} = \sum_{p=1}^{m} \sum_{i=0}^{N} \left| u_p(x_{i+\frac{1}{2}}, t) - U_p(x_{i+\frac{1}{2}}, t) \right| \Delta x_{i+\frac{1}{2}},$$

where m is the number of conserved variables.

The exact solution for the Sod shock tube problem does not exist in closed form, however the solution may be obtained iteratively to any degree of precision by using an exact Riemann solver. The exact solution used here was computed using the exact Riemann solver found in [93]. The errors for the three different methods can be seen in table 5.3.

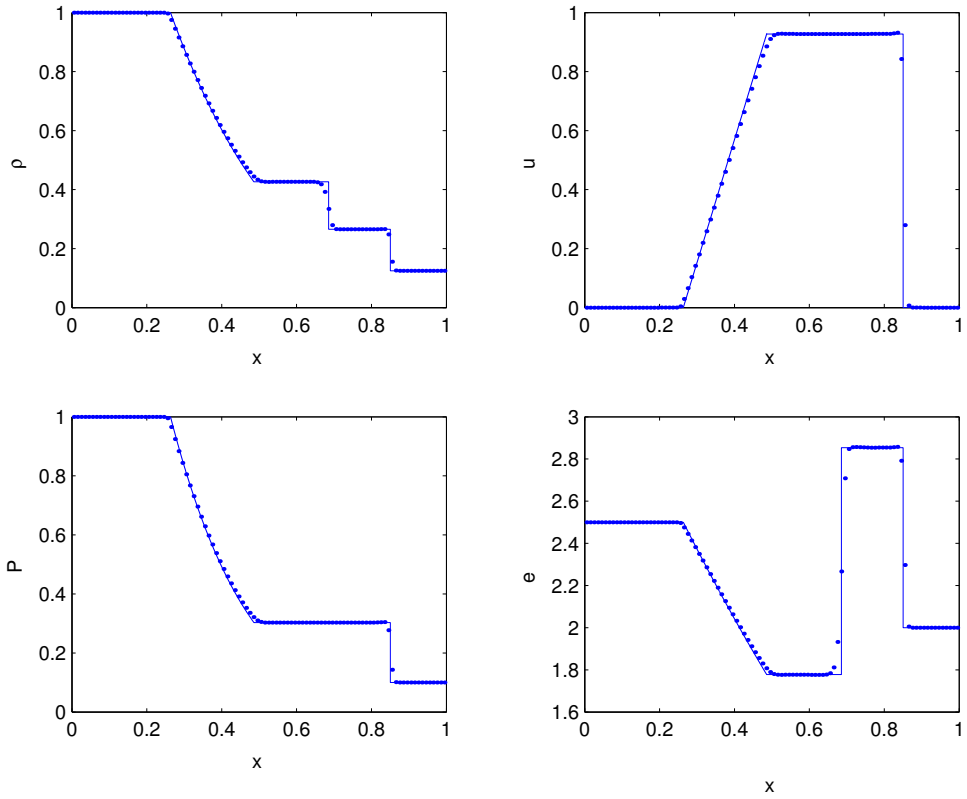

Figure 5.17: Eulerian solution to the Euler equations. Figure shows plots of the density, velocity, pressure and internal energy of the gas at $t = 0.2$. The bold and dotted lines represent the exact and numerical solutions respectively. The solution was calculated with 100 mesh points.

The error table 5.3 shows that again the moving mesh method along with the geometrical monitor function $M = 1 + \alpha |\rho_x|$ gives rise to numerical solutions at least twice as accurate in comparison with both the Eulerian method and the moving mesh method with the monitor function $M = \rho$. It can be seen from this table that the results obtained from the moving mesh method with the mass monitor function produces results which are less accurate than those computed with the Eulerian method. This is partly due to the fact that the mesh movement causes slight oscillations in the solution around the contact surface. The results using this monitor function are improved if smoothing is employed.

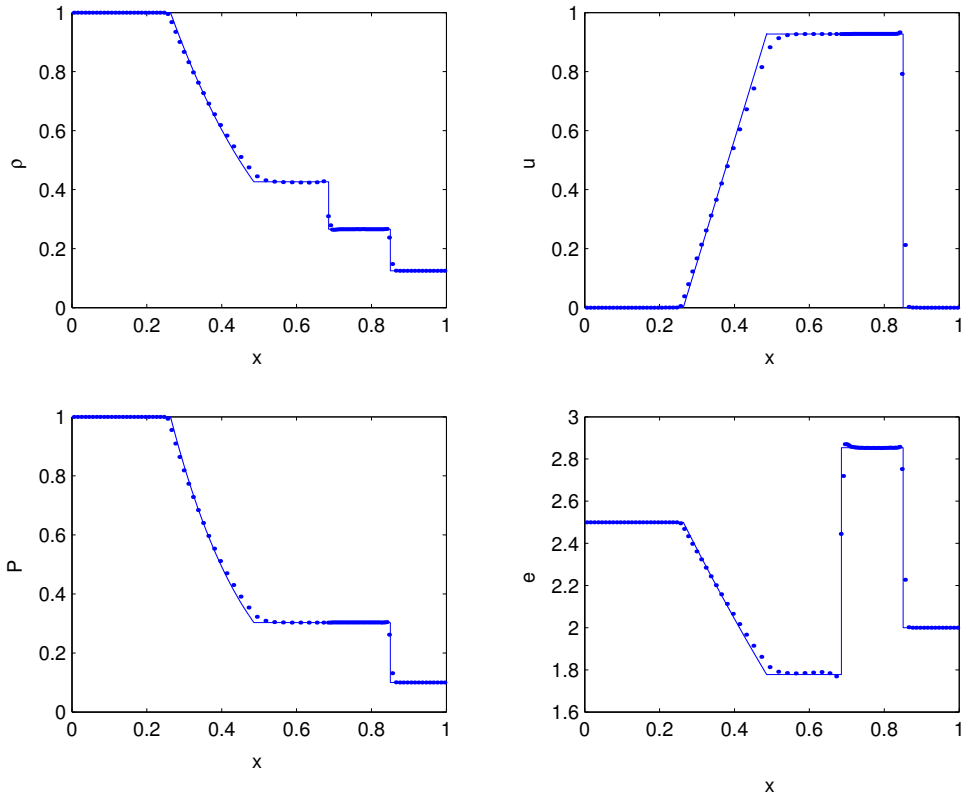

Figure 5.18: Solution calculated by the moving mesh method with $M = \rho$. The figure shows plots of the density, velocity, pressure and internal energy of the gas at $t = 0.2$. The bold and dotted lines represent the exact and numerical solutions respectively. The solution was calculated with 100 mesh points.

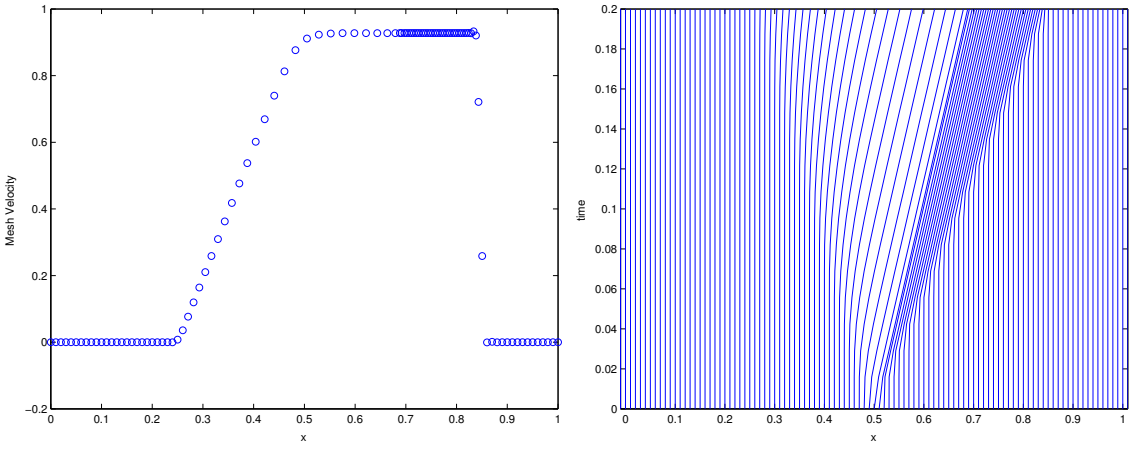

Figure 5.19: Figures showing the velocity of the mesh points at the output time $t = 0.2$ (left) and the trajectories of the mesh points (right) arising from the use of the monitor function $M = \rho$.

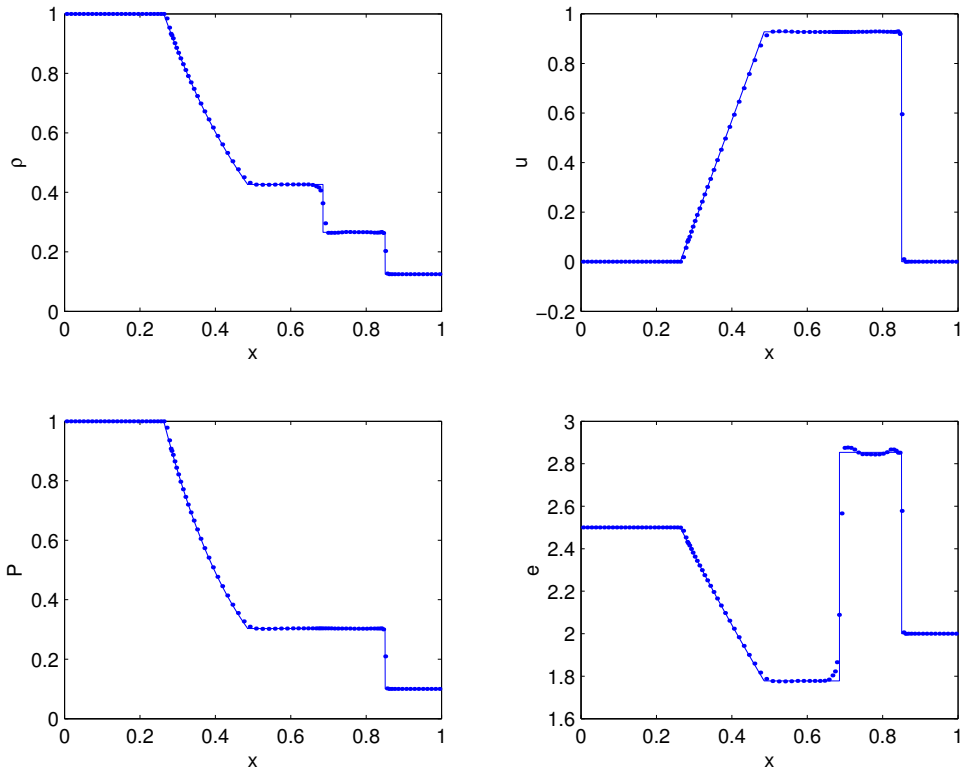

Figure 5.20: Solution calculated with moving mesh method with $M = 1 + \alpha |\rho_x|$. *Figure shows plots of the density, velocity, pressure and internal energy of the gas at* $t = 0.2$. *The bold and dotted lines represent the exact and numerical solutions respectively. The solution was calculated with 100 mesh points.*

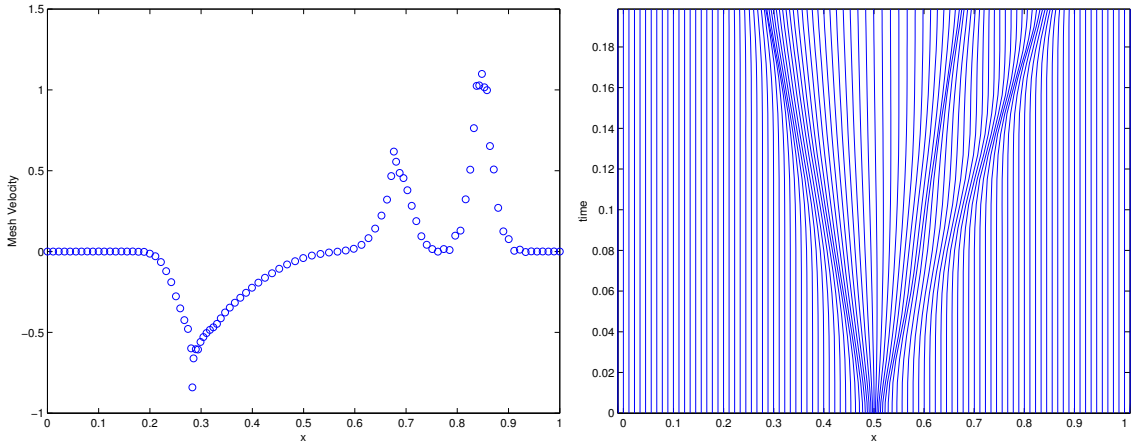

Figure 5.21: Figures showing the velocity of the mesh points at the output time $t = 0.2$ *(left) and the trajectories of the mesh points (right) arising from the use of the monitor function* $M = 1 + \alpha |\rho_x|$.

Method	Error (1st order)	Error (2nd order)
Eulerian (Fixed)		
N=50	0.11282	0.03061
N=100	0.07234	0.01422
N=200	0.04461	0.00695
N=400	0.02473	0.00349
Lagrangian $M = \rho$		
N=50	0.13313	0.04034
N=100	0.08523	0.02012
N=200	0.05315	0.01069
Geometrical $M = 1 + \alpha \lvert\rho_x\rvert$		
N=50 ($\beta = 2.5$)	0.06369	0.01003
N=100 ($\beta = 2.5$)	0.03579	0.00576
N=200 ($\beta = 2.5$)	0.02532	0.00396

Table 5.3: L_1 errors for fixed and moving mesh methods for the compressible Euler equations.

5.8 Summary

In this chapter we have used the moving mesh method derived in chapter 3 to generate adaptive meshes on which hyperbolic conservation laws have been solved. The chapter began by considering how to adapt fixed grid finite volume methods onto moving grids. Based on the work of Harten and Hyman [43], a finite volume method for the solution of hyperbolic conservation laws was derived. It was found that to take into account the motion of the mesh the flux function had to be shifted. The subsequent numerical method employed the approximate Roe Riemann solver to solve local Riemann problems between computational cells and was utilised to solve both the inviscid Burgers equation and the compressible Euler equations on both moving and fixed grids.

Two monitor functions were used to drive the moving mesh method. One was the mass monitor and was found to lead to the mesh motion being approximately Lagrangian. The other monitor function was based on geometrical properties of the solution and was designed to move mesh points into spatial regions where the gradient of the solution was large. In this manner it was seen that the resulting adaptive mesh gave better resolution of moving discontinuities.

For the inviscid Burgers equation we considered the test problem of a single moving discontinuity. This problem was solved with the two types of mesh movement and was compared to the numerical solution obtained on a fixed grid. It was found that the moving mesh calculations gave more accurate numerical solutions in an aggregated L_1-norm.

The method was then extended to generate an adaptive mesh for the compressible Euler equations. Using similar monitor functions to those which had been used for the solution of the inviscid Burgers equation we obtained two different moving meshes. Again a mass monitor function led to the mesh motion being approximately Lagrangian and a monitor function based on the gradient of the fluid density led to a mesh movement that was neither Eulerian or Lagrangian, but inherited some of the wave-structure of the solution and hence clustered mesh points in shock, contact and to a lesser extent in rarefaction waves. Again, the combined L_1-norm of the solution error was computed for this test problem and it was found that the results for the geometric monitor function were much

more accurate when compared with the Eulerian method and the moving mesh method with the mass monitor function $M = \rho$.

One drawback of the geometric monitor function $M = 1 + \alpha|\rho_x|$ was that the constant α is undetermined and hence in numerical calculations we had to tune this parameter. Although the solution seems not to be particularly sensitive to the choice of this parameter, we would, in practice, not like to tune parameters. In the numerical experiments it was found that the parameter α could be used to control the amount of grid deformation and hence the amount of adaptivity. If the constant was taken to be too small it was found that the resulting mesh adaption was slight. Alternatively, if it was taken to be too large there would be large deformations in the mesh which would cause the time-step of the numerical calculation to decrease significantly in time. Therefore it would be advantageous to have some technique for choosing the parameter α depending on the solution to the problem.

Another field of extended study could be the design of other monitor functions for this test problem such as the ones found in [86]. In the next chapter we will use ideas developed in this chapter to solve the two dimensional Euler equations on an adaptive mesh.

Chapter 6

The Euler Equations In Two Dimensions

6.1 Introduction

In this chapter we will consider the adaptive solution of the compressible Euler equations of gas dynamics in two spatial dimensions. In two dimensions the Euler equations are given as

$$\begin{pmatrix} \rho \\ \rho\, u \\ \rho\, v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho\, u \\ \rho\, u^2 + p \\ \rho\, u\, v \\ u\ (E+p) \end{pmatrix}_x + \begin{pmatrix} \rho\, v \\ \rho\, u\, v \\ \rho\, v^2 + p \\ v\ (E+p) \end{pmatrix}_y = \underline{0} \tag{6.1}$$

where ρ is the density, $\mathbf{v} = (u, v)^T$ are the velocity components in the x and y directions respectively, p is the pressure and E is the total energy of the fluid. Now the compressible Euler equations in two spatial dimensions are a set of four hyperbolic conservation laws since the conservation of momentum equation comprises of one equation for each component of the momentum. Again, as with the one-dimensional system of Euler equations, an equation of state (EOS) is required to close the system since we have four equations in five unknowns. In this work we will again use the ideal gas equation of state which in two spatial dimensions, given as

$$E = \frac{p}{(\gamma - 1)} + \frac{1}{2}\rho\, |\mathbf{v}|^2 ,$$

where $|\mathbf{v}|^2 = u^2 + v^2$. We will solve the compressible Euler equations given by (6.1) using the moving mesh method that has been derived in chapter 3, and use a similar solution procedure to the one for the solution of the one dimensional conservation laws used in chapter 5. Therefore we will have to solve the ALE form of the Euler equations which are given as

$$\begin{pmatrix} \rho \\ \rho\, u \\ \rho\, v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho\ (u - \dot{x}) \\ \rho\, u\, (u - \dot{x}) + p \\ \rho\, v\, (u - \dot{x}) \\ E\ (u - \dot{x}) + u\, p \end{pmatrix}_x + \begin{pmatrix} \rho\ (v - \dot{y}) \\ \rho\, u\ (v - \dot{y}) \\ \rho\, v\, (v - \dot{y}) + p \\ E\ (v - \dot{y}) + v\, p \end{pmatrix}_y = \underline{0}, \tag{6.2}$$

where $\dot{\mathbf{x}} = (\dot{x}, \dot{y})$ are the ALE velocity components in the x and y directions, respectively.

In the next section we will consider the solution of the two-dimensional compressible Euler equations in ALE form given by (6.2).

6.2 A Finite Volume Solver in 2D

In this section we describe a method for solving hyperbolic conservation laws in two spatial dimensions. In chapter 5 we considered the solution of one dimensional hyperbolic conservation laws on adaptively moving meshes by using the Arbitrary-Lagrangian Eulerian (ALE) formulation of the equations. Using this formulation of the conservation laws in a moving spatial domain, we derived a finite volume method which could then be used to solve the equation being considered, when $\dot{x}$ was given. In two dimensions we will use a similar methodology and determine the two-dimensional generalisation of the ALE finite volume method.

Since we aim to solve the Euler equations on an unstructured triangular mesh we will derive the finite volume method using triangular volume elements. We begin by considering a general system of hyperbolic conservation laws on a moving triangular control volume Δ_i. This triangular control volume can be seen in figure 6.1 and comprises of three edges denoted by $\partial\Delta_1$, $\partial\Delta_2$, $\partial\Delta_3$ and has size $|\Delta_i|$. We now consider the ALE form of a two-dimensional conservation law in this moving control volume given as

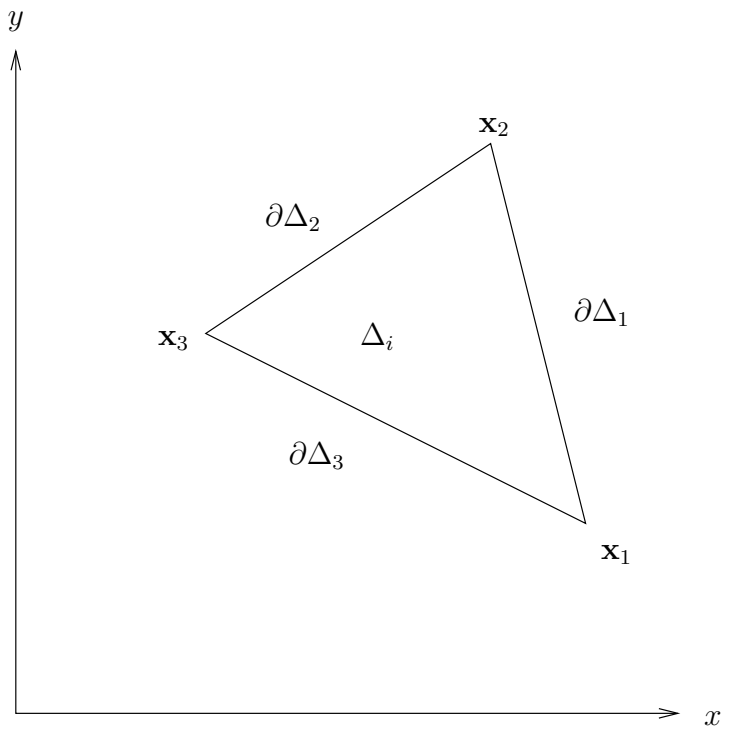

Figure 6.1: Figure showing the triangular control volume Δ_i.

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Delta_i} \underline{\mathrm{u}}\,\mathrm{d}\Omega + \int_{\Delta_i} \nabla\cdot(\underline{\mathbf{f}} - \underline{\mathrm{u}}\,\dot{\mathbf{x}})\;\mathrm{d}\Omega = \underline{0},$$

where $\underline{\mathbf{f}} = (\underline{\mathrm{f}}, \underline{\mathrm{g}})$. We will denote the ALE flux function as $\bar{\underline{\mathbf{f}}} = \underline{\mathbf{f}} - \underline{\mathrm{u}}\,\dot{\mathbf{x}}$. From this initial statement of the integral ALE form of the conservation law we construct a conservative numerical method analogous to the one that was produced in one spatial dimension. We can use the divergence theorem to write the flux terms of this conservation law in terms of a boundary integral to obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Delta_i} \underline{\mathrm{u}}\,\mathrm{d}\Omega + \oint_{\partial\Delta_i} \bar{\underline{\mathbf{f}}}\cdot\mathbf{n}\,\mathrm{d}\Gamma = \underline{0}. \tag{6.3}$$

We can write this path integral around the boundary of the control volume Δ_i as a sum of integrals around the boundary segments. Therefore

$$\oint_{\partial\Delta_i} \bar{\underline{\mathbf{f}}}\cdot\mathbf{n}\,\mathrm{d}\Gamma = \sum_{j=1}^{3}\oint_{\partial\Delta_j} \bar{\underline{\mathbf{f}}}\cdot\mathbf{n}_j\,\mathrm{d}\Gamma,$$

where $\mathbf{n}_j$ is the unit outward normal to the j^{th} edge of the triangular control volume as shown in figure 6.2. From figure 6.2 it is straightforward to see that the outward normal

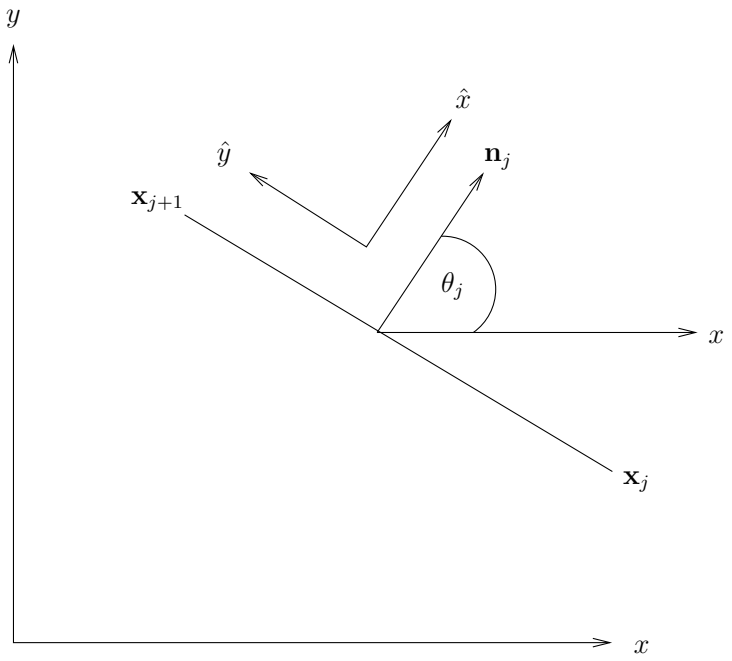

Figure 6.2: Figure showing the j^{th} edge $\partial\Delta_j$ of the control volume Δ_i and the unit outward normal $\mathbf{n}_j$ to this edge.

to the j^{th} side can be written in terms of the angle θ_j between the x-direction and the outward normal $\mathbf{n}_j$. Therefore we may write the unit outward normal as

$$\mathbf{n}_j = (\cos\,\theta_j, \sin\,\theta_j)\,.$$

Using this notation, the flux term for the conservation law can be written as

$$\sum_{j=1}^{3} \oint_{\partial\Delta_j} \bar{\underline{\mathrm{f}}} \cdot \mathbf{n}_j \,\mathrm{d}\Gamma = \sum_{j=1}^{3} \oint_{\partial\Delta_j} \left(\cos\,\theta_j\, \bar{\underline{\mathrm{f}}} + \sin\,\theta_j\, \bar{\underline{g}}\right) \mathrm{d}\Gamma,$$

where $\bar{\underline{\mathrm{f}}}$ and $\bar{\underline{g}}$ are the ALE flux functions in the x and y-direction respectively. Therefore, substituting this expression for the flux of the conservation law into equation (6.3) we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Delta_i} \underline{\mathrm{u}}\,\mathrm{d}\Omega + \sum_{j=1}^{3} \oint_{\partial\Delta_j} \left(\cos\,\theta_j\, \bar{\underline{\mathrm{f}}} + \sin\,\theta_j\, \bar{\underline{g}}\right) \mathrm{d}\Gamma = \underline{0}. \tag{6.4}$$

If we take the cell average value of the conserved variables in the control volume Δ_i to be

$$\underline{\mathrm{u}}_i = \frac{1}{|\Delta_i|} \int_{\Delta_i} \underline{\mathrm{u}} \, \mathrm{d}\Omega,$$

where $|\Delta_i|$ is the size of the control volume, then equation (6.4) becomes an equation for the evolution of the cell average value of the conserved variables and is given as

$$\frac{\mathrm{d}}{\mathrm{d}t} \left(|\Delta_i| \, \underline{\mathrm{u}}_i \right) + \sum_{j=1}^{3} \oint_{\partial \Delta_j} \left(\cos \theta_j \, \bar{\underline{\mathrm{f}}} + \sin \theta_j \, \bar{\underline{g}} \right) \, \mathrm{d}\Gamma = \underline{0}. \tag{6.5}$$

Up until now we have made no numerical approximations in the derivation of equation (6.5), but to obtain a numerical finite volume method we now need to make a choice for the method of approximating both the time derivative and the intercell flux. We will use the forward Euler approximation for the time derivative in equation (6.5) to obtain the semi-discrete system

$$\underline{\mathrm{u}}_i^{n+1} = \frac{|\Delta_i^n|}{\left|\Delta_i^{n+1}\right|} \underline{\mathrm{u}}_i^n - \frac{\Delta t^n}{\left|\Delta_i^{n+1}\right|} \sum_{j=1}^{3} \oint_{\partial \Delta_j} \left(\cos \theta_j \, \bar{\underline{\mathrm{f}}} + \sin \theta_j \, \bar{\underline{g}} \right) \, \mathrm{d}\Gamma. \tag{6.6}$$

We still need to make a choice for the approximation to the flux functions. This can be done by exploiting the rotational invariance of the hyperbolic conservation laws being solved. For example, the ALE flux terms for the two-dimensional Euler equations can be written as

$$\cos \theta_j \, \bar{\underline{\mathrm{f}}} + \sin \theta_j \, \bar{\underline{g}} = R_j^{-1} \bar{\underline{\mathrm{f}}} \left(R_j \, \underline{\mathrm{u}} \right),$$

where $R_j \equiv R\left(\theta_j\right)$ is a rotation matrix given as

$$R\left(\theta\right) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.7}$$

and $R_j^{-1} \equiv R^{-1}\left(\theta_j\right)$ is given as

$$R^{-1}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6.8}$$

With this simplification equation (6.6) becomes

$$\underline{\mathrm{u}}_i^{n+1} = \frac{|\Delta_i^n|}{\left|\Delta_i^{n+1}\right|}\,\underline{\mathrm{u}}_i^n - \frac{\Delta t^n}{\left|\Delta_i^{n+1}\right|}\sum_{j=1}^{3}\oint_{\partial\Delta_j} R_j^{-1}\bar{\underline{\mathrm{f}}}\left(R_j\,\underline{\mathrm{u}}\right)\,\mathrm{d}\Gamma. \tag{6.9}$$

Notice that the quantity $\hat{\underline{\mathrm{u}}} = R_j\,\underline{\mathrm{u}}$ is the value of the conserved variables rotated to align with the outward normal $\mathbf{n}_j$ in the new rotated cartesian co-ordinate system $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$. The conservation laws in this new co-ordinate system become a one-dimensional system given by

$$\hat{\underline{\mathrm{u}}}_t + \bar{\underline{\mathrm{f}}}\left(\hat{\underline{\mathrm{u}}}\right)_{\hat{x}} = \underline{0}.$$

Therefore, for the compressible Euler equations this system becomes

$$\begin{pmatrix} \rho \\ \rho\,\hat{u} \\ \rho\,\hat{v} \\ E \end{pmatrix}_t + \begin{pmatrix} \rho\left(\hat{u} - \hat{\dot{x}}\right) \\ \rho\,\hat{u}\left(\hat{u} - \hat{\dot{x}}\right) + p \\ \rho\,\hat{v}\left(\hat{u} - \hat{\dot{x}}\right) \\ E\left(\hat{u} - \hat{\dot{x}}\right) + \hat{u}\,p \end{pmatrix}_{\hat{x}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{6.10}$$

We can make the approximation for the integrals around the control volume

$$\oint_{\partial\Delta_j} R_j^{-1}\bar{\underline{\mathrm{f}}}\left(R_j\,\underline{\mathrm{u}}\right)\,\mathrm{d}\Gamma \approx L_j\,R_j^{-1}\,\bar{\underline{\mathrm{f}}}_j,$$

where L_j is the length of the j^{th} edge of the triangular control volume Δ_i. Our finite volume scheme (6.9) together with this approximation then becomes

$$\underline{\mathrm{u}}_i^{n+1} = \frac{|\Delta_i^n|}{\left|\Delta_i^{n+1}\right|}\,\underline{\mathrm{u}}_i^n - \frac{\Delta t^n}{\left|\Delta_i^{n+1}\right|}\sum_{j=1}^{3} L_j\,R_j^{-1}\,\bar{\underline{\mathrm{f}}}_j\left(R_j\,\underline{\mathrm{u}}\right). \tag{6.11}$$

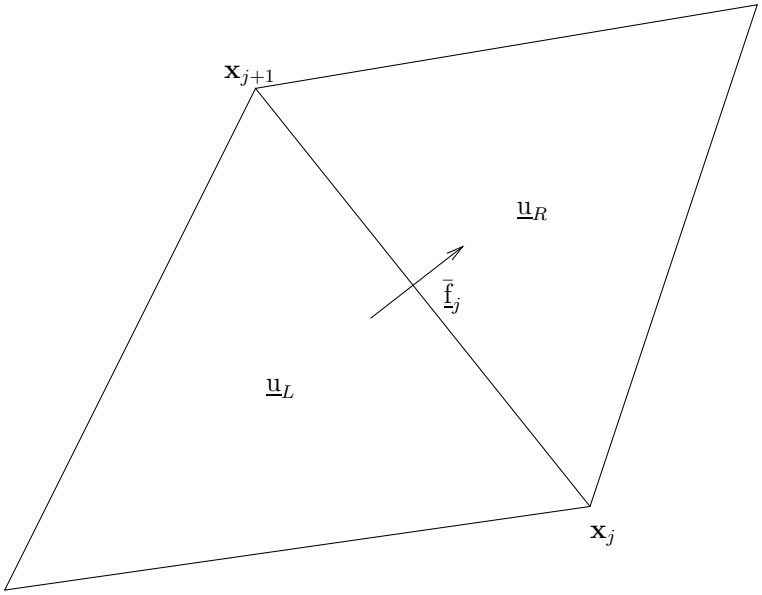

Figure 6.3: Figure showing the local Riemann problem between the two constant cell centered states $\underline{u}_L$ and $\underline{u}_R$.

We now have a cell-centred finite volume method which can be used to evolve the cell averages of the conserved variables. However, we still need to make a choice for determining the flux through the computational cell. As in one dimension we will use Godunov's method to determine the intercell flux by solving local Riemann problems between neighbouring cells using the cell centred values as initial conditions. This can be seen in figure 6.3. The Riemann problem for the Euler equations only needs to be solved in the direction normal to the edge that separates the neighbouring cells. So we may rotate the initial conditions for the Riemann problem using the rotation matrix (6.7) so that the variables align with the outward normal of the edge and then solve the x-split system of equations given by equation (6.10). The solution of the Riemann problem can then be obtained by rotating the solution back onto the original cartesian co-ordinates by using the matrix (6.8).

In two spatial dimensions we will use the HLLC approximate Riemann solver of Toro, Spruce and Speares [94] to determine the intercell flux. The details of this approximate Riemann solver will be given in the following section.

6.3 An ALE HLLC Riemann Solver

In this section we will outline the use of the HLLC Riemann solver of Toro *et al* [94]. Harten, Lax and Van Leer [44] originally proposed a method for obtaining intercell Godunov fluxes referred to as the HLL approximate Riemann solver after its authors. This Riemann solver assumes a wave configuration of the solution consisting of two waves separating three constant states, where the speeds of the waves are given by some suitable algorithm. However the HLL Riemann solver can lead, in some cases, to poor accuracy of contact and shear waves, as the solution technique does not explicitly include these in the wave configuration. Therefore, due to this deficiency in the HLL Riemann solver Toro, Spruce and Speares [94] developed the HLLC Riemann solver, where the C stands for contact. They restored the contact wave into the assumed wave configuration so that the solution to the Riemann problem consists of three waves separating four constant states. The speeds of these three waves will be be denoted by S_L, S^* and S_R and are assumed to separate the four constant states $\underline{\mathrm{u}}_L$, $\underline{\mathrm{u}}_L^*$, $\underline{\mathrm{u}}_R^*$ and $\underline{\mathrm{u}}_R$. Also $\dot{x}$ will denote the velocity of the intercell boundary. The five possible cases for the structure of the assumed ALE HLLC Riemann solution are shown in figure 6.4. Using conservation principles it is possible to obtain expressions for the numerical flux function. We will briefly outline how the flux function can be obtained for case (b) of figure 6.4.

We now apply the conservation law over the control volume between the left cell boundary and the motion of the intercell boundary at the start and end of the time-levels. At the start of the time-level the amount of the conserved variable between the left cell boundary and the intercell boundary is $\frac{1}{2}\,\Delta x\,\underline{\mathrm{u}}_L$. This amount then changes in time by the flux through the left and intercell boundary $\underline{\mathrm{F}}_L$ and $\underline{\mathrm{F}}_{HLLC}$. Therefore the amount of the conserved variable at time Δt is given as

$$\frac{1}{2}\,\Delta x\,\underline{\mathrm{u}}_L + \Delta t\,\left(\underline{\mathrm{F}}_L - \underline{\mathrm{F}}_{HLLC}\right). \tag{6.12}$$

Alternatively, the amount of the conserved variable can be calculated by considering the amount of the conserved variable at Δt. Thus we can write this as

$$\frac{1}{2}\,\Delta x\,\underline{\mathrm{u}}_L - (S_L - \dot{x})\;\Delta t\,\underline{\mathrm{u}}_L + (S_L - \dot{x})\;\Delta t\,\underline{\mathrm{u}}_L^*. \tag{6.13}$$

Therefore equating the expressions (6.12) and (6.13) and rearranging we obtain the HLLC flux function for case (b) given as

$$\underline{\mathrm{F}}_{HLLC} = \underline{\mathrm{F}}_L + (S_L - \dot{x})\ (\underline{\mathrm{u}}_L^* - \underline{\mathrm{u}}_L)\,.$$

Applying similar techniques for the rest of the possible wave configurations it is possible to write the ALE HLLC flux function as

$$\underline{\mathrm{F}}_{HLLC} = \begin{cases} \underline{\mathrm{F}}_L & \text{if } S_L - \dot{x} > 0, \\ \underline{\mathrm{F}}_L^* = \underline{\mathrm{F}}_L + (S_L - \dot{x})\,(\underline{\mathrm{u}}_L^* - \underline{\mathrm{u}}_L) & \text{if } S_L - \dot{x} \le 0 < S^* - \dot{x}, \\ \frac{1}{2}\ (\underline{\mathrm{F}}_L^* + \underline{\mathrm{F}}_R^*) & \text{if } S^* - \dot{x} = 0, \\ \underline{\mathrm{F}}_R^* = \underline{\mathrm{F}}_R + (S_R - \dot{x})\,(\underline{\mathrm{u}}_R^* - \underline{\mathrm{u}}_R) & \text{if } S^* - \dot{x} < 0 < S_R - \dot{x}, \\ \underline{\mathrm{F}}_R & \text{if } S_R - \dot{x} < 0. \end{cases} \tag{6.14}$$

Estimates for the wave speeds S_L, S^* and S_R and also the values $\underline{\mathrm{u}}_L^*$, $\underline{\mathrm{u}}_R^*$ of the conserved variables in the star region, i.e. the region between the waves S_L and S_R, now need to be obtained to fully determine the HLLC flux function given by (6.14).

The value of the conserved variables for the compressible Euler equations $\underline{\mathrm{u}}_L^*$ and $\underline{\mathrm{u}}_R^*$ can be obtained by applying the Rankine-Hugoniot jump conditions across the waves S_L, S^* and S_R. By doing this we obtain the conditions

$$\left.\begin{aligned} \underline{\mathrm{F}}_L^* &= \underline{\mathrm{F}}_L + S_L\ (\underline{\mathrm{u}}_L^* - \underline{\mathrm{u}}_L)\,, \\ \underline{\mathrm{F}}_R^* &= \underline{\mathrm{F}}_L^* + S^*\ (\underline{\mathrm{u}}_R^* - \underline{\mathrm{u}}_L^*)\,, \\ \underline{\mathrm{F}}_R^* &= \underline{\mathrm{F}}_R + S_R\ (\underline{\mathrm{u}}_R^* - \underline{\mathrm{u}}_R)\,. \end{aligned}\right\} \tag{6.15}$$

We also enforce the conditions that the velocity of the fluid in the x-direction and the pressure does not jump across the central wave S^*. The condition that the velocity component v in the y-direction does not to jump across the waves S_L and S_R is used. These conditions can be written as

$$\left.\begin{aligned} u_L^* = u_R^* = u^*, \\ p_L^* = p_R^* = p^*, \\ v_L^* = v_L \quad v_R^* = v_R. \end{aligned}\right\} \tag{6.16}$$

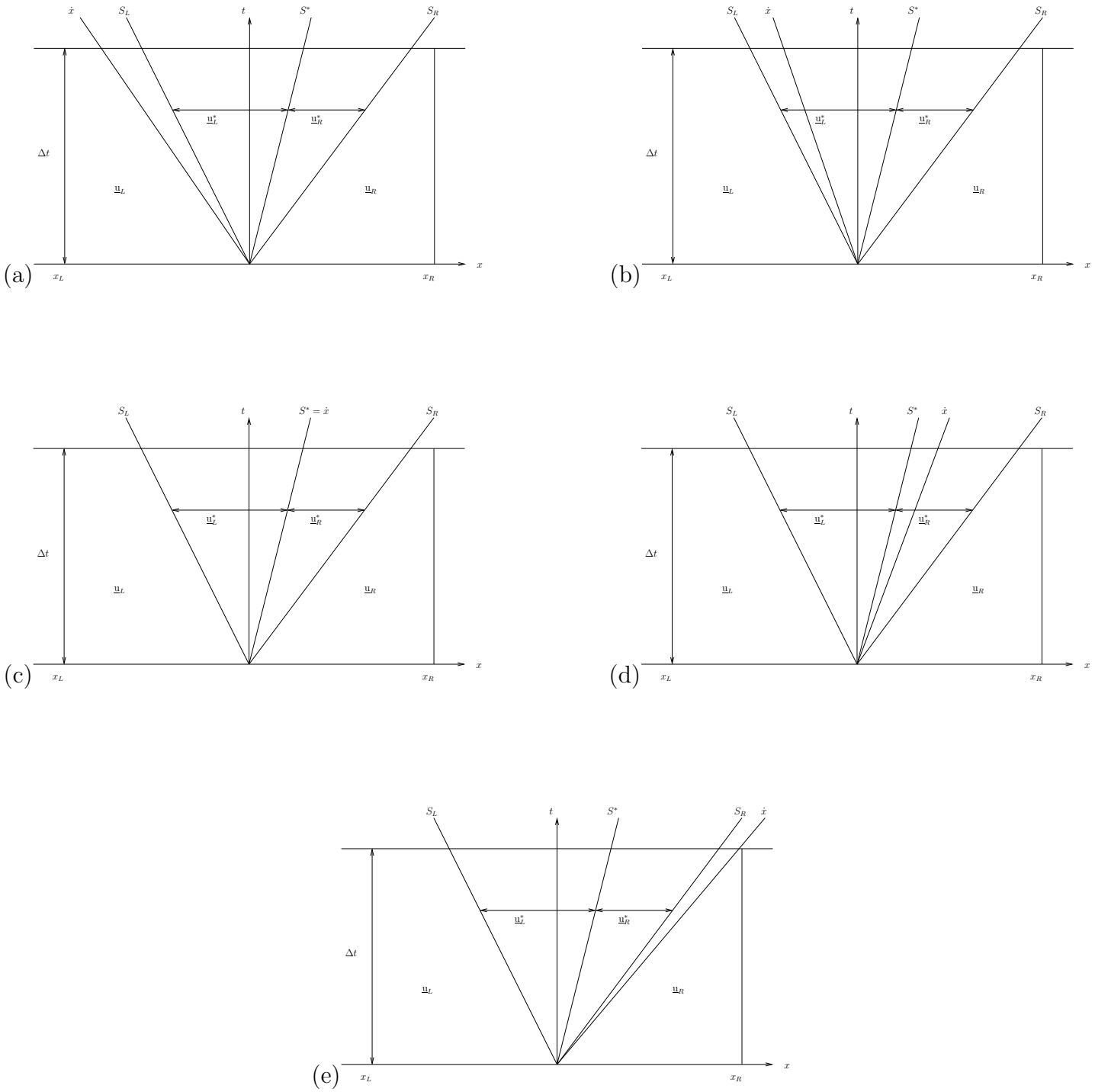

Figure 6.4: Figure showing the five possible cases for the ALE HLLC approximate Riemann solver.

We also set $S^* = u^*$. Using the Rankine-Hugoniot conditions given by (6.15) and the conditions (6.16) we obtain an expression for the values of the conserved variables in the star region given by

$$\underline{\mathrm{u}}_K^* = \rho_K \left(\frac{S_K - u_K}{S_K - S^*} \right) \begin{bmatrix} 1 \\ S^* \\ v_K \\ \frac{E_k}{\rho_K} + (S^* - u_K) \left[S^* + \frac{p_K}{\rho_K (S_K - u_K)} \right] \end{bmatrix}. \tag{6.17}$$

Here the subscript K denotes either the left state L or the right state R.

We will now state the choice of the wave speeds S_L, S^* and S_R. Given estimates for the pressure p^* and the velocity u^* in the star region then we may choose the wave speeds to be

$$S_L = u_L - c_L\, q_L \qquad S^* = u^* \qquad S_R = u_R + c_R\, q_R \tag{6.18}$$

where

$$q_K = \begin{cases} 1 & \text{if } p^* \leq p_K, \\ \left[1 + \frac{\gamma+1}{2\gamma} \left(\frac{p^*}{p_K - 1} \right) \right] & \text{if } p^* > p_K. \end{cases} \tag{6.19}$$

This quantity distinguishes between whether we have a shock or rarefaction wave. If the wave is a shock then the first case in (6.19) is chosen and the wave speed estimates are approximations to the shock speeds. Alternatively, if we have a rarefaction wave then the second case in (6.19) is chosen, and gives us an estimate for the speed of the head of the rarefaction wave.

We also need to obtain estimates for the velocity u^* and pressure p^* in the star region. In this work we will use the wave speed estimates from [93] which are based on the exact Riemann solver of Toro. The pressure and velocity estimates will now be described. Given an initial estimate of the pressure in the star region given as

$$p^*_{init} = \frac{1}{2}\,(p_L + p_R) - \frac{1}{2}\,(u_R - u_L)\;\bar{\rho}\,\bar{c}$$

where $\bar{\rho} = \frac{1}{2}\,(\rho_L + \rho_R)$ and $\bar{c} = \frac{1}{2}\,(c_L + c_R)$,

- If $p_{min} < p^*_{init} < p_{max}$ and $\frac{p_{max}}{p_{min}} < 2$ where $p_{min} = \min(p_L, p_R)$ and $p_{max} = \max(p_L, p_R)$ then

$$\begin{aligned} p^* &= \max(tol, p_{init}), \\ u^* &= \frac{1}{2}(u_L + u_R) - \frac{(p_R - p_L)}{2\,\bar{c}\,\bar{\rho}}, \end{aligned}$$

where $tol = 0.0001$.

- If $p^*_{init} < p_{min}$ then

$$\begin{aligned} p^* &= \left[\frac{c_L + c_R - \frac{\gamma-1}{2}(u_R - u_L)}{\frac{c_L}{(p_L)^z} + \frac{c_R}{(p_R)^z}}\right]^{\frac{1}{z}}, \\ u^* &= u_L - \frac{2\,c_L}{(\gamma-1)}\left[\left(\frac{p^*}{p_L}\right)^z - 1\right], \end{aligned}$$

where $z = \frac{\gamma-1}{2\,\gamma}$.

- Otherwise

$$\begin{aligned} p^* &= \frac{g_L(p_0)\,p_L + g_R(p_0)\,p_R - (u_R - u_L)}{g_L(p_0) + g_R(p_0)} \\ u^* &= \frac{1}{2}(u_L + u_R) + [(p^* - p_R)\,g_R(p_0) - (p^* - p_L)\,g_L(p_0)] \end{aligned}$$

where

$$g_K(p) = \sqrt{\frac{A}{p + B_K}},$$

$$A = \frac{2}{(\gamma+1)\,\rho_K}, \qquad B = \frac{\gamma-1}{(\gamma+1)}\,p_K$$

for $K = L$, R and $p_0 = max(p_{init}, 0)$.

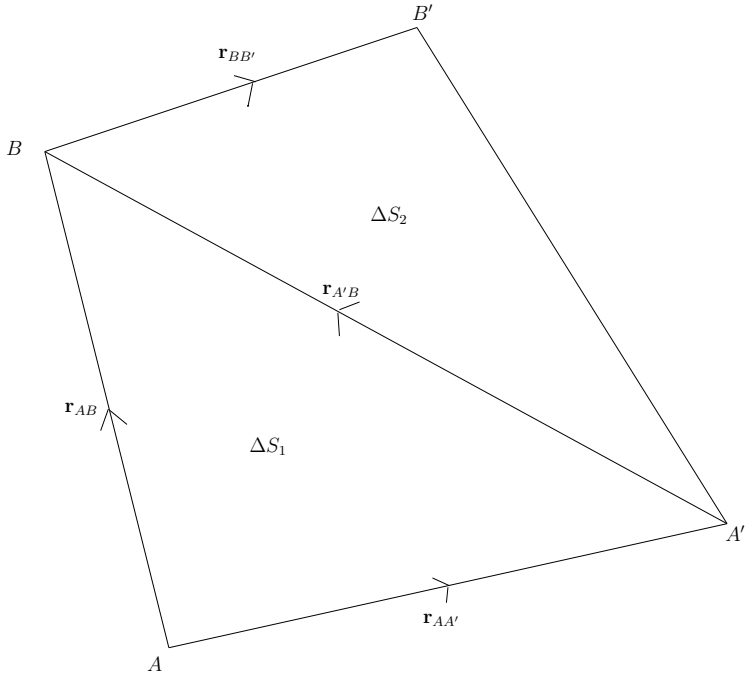

Figure 6.5: Figure showing the area swept out by the edge $A - B$ in time Δt.

We also need to obtain a value for the speed of the intercell boundary $\dot{x}$. Since we will be solving the Riemann problem across the common edge in the normal direction between two computational cells this intercell boundary speed $\dot{x}$ translates into the speed of the edge in the normal direction. The evaluation of the edge velocity is considered by Zhang *et al* in [104]. They obtain the speed of the edge by calculating the area ΔS swept out by the edge of the moving control volume and dividing this by the size of the time-step and the length of the edge L. Therefore the velocity is given by

$$\dot{x} = \frac{\Delta S}{L\,\Delta t}. \tag{6.20}$$

In [104] the area swept out by the edge is calculated in the following manner. Consider the edge connecting the points A and B which are located at the positions $\mathbf{x}_A$ and $\mathbf{x}_B$ which move in time to the points A' and B' which are located at the positions $\mathbf{x}_{A'}$ and $\mathbf{x}_{B'}$, respectively. This motion of the edge is shown in figure 6.5.

The total area swept out by the edge is then calculated by splitting it into two triangular regions where the size of these regions add up to the total area in the sense that

$$\Delta S = \Delta S_1 + \Delta S_2,$$

where

$$\begin{aligned}\Delta S_1 &= \frac{1}{2}\left|\mathbf{r}_{AA'} \times \mathbf{r}_{AB}\right| \\ &= \frac{1}{2}\left(x_{B'} - x_{A'} - \Delta t\left(\dot{x}_B - \dot{x}_A\right)\right)\Delta t\,\dot{y}_A - \frac{1}{2}\left(y_{B'} - y_{A'} - \Delta t\left(\dot{y}_B - \dot{y}_A\right)\right)\Delta t\,\dot{x}_A\end{aligned}$$

and

$$\begin{aligned}\Delta S_2 &= \frac{1}{2}\left|\mathbf{r}_{BB'} \times \mathbf{r}_{A'B}\right| \\ &= \frac{1}{2}\left(y_{B'} - y_{A'} - \Delta t\,\dot{y}_B\right)\Delta t\,\dot{x}_B - \frac{1}{2}\left(x_{B'} - x_{A'} - \Delta t\,\dot{x}_B\right)\Delta t\,\dot{y}_B.\end{aligned}$$

An outline for the use of the ALE HLLC approximate Riemann solver is therefore

- Given initial estimates for the velocity of the fluid in the x-direction u^* and the pressure p^* in the star region, compute the wave speeds given by (6.18).
- Then compute the states of the conserved variables in the star region given by (6.17) and also calculate the velocity of the intercell boundary given by (6.20).
- Calculate the ALE HLLC approximate Riemann flux function given by (6.14) and use it in the finite volume scheme given by (6.11).

The resulting finite volume method with the use of the HLLC approximate Riemann solver is only first order accurate in space and time, but in the following section we will describe how higher spatial accuracy can be achieved.

6.4 The MUSCL Technique of Van Leer

The finite volume method together with the HLLC approximate Riemann solver which has been described in sections 6.2 and 6.3, respectively, will result in a first order accurate method in space and time. Therefore to obtain higher order numerical approximations to the conservation laws being solved we need to use some other technique. The technique that will be used in this work to obtain a high resolution solution is the Monotone Upwind Schemes for Conservation Laws (MUSCL) technique of Van Leer [58]. This

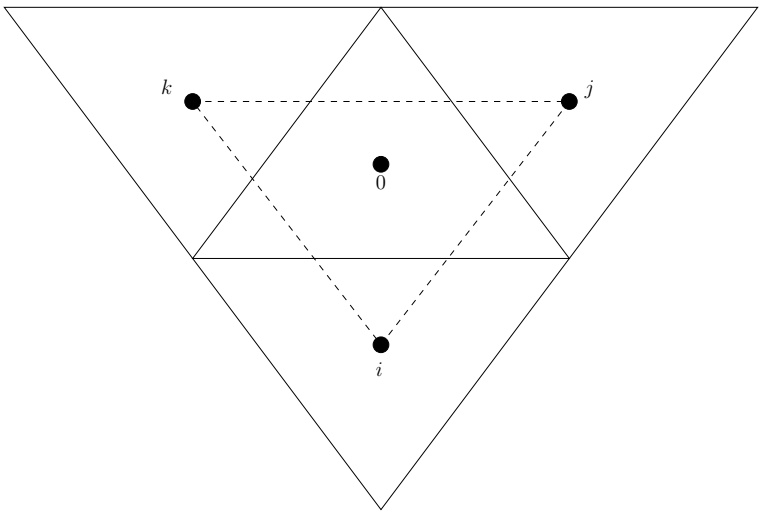

Figure 6.6: Figure showing the three surrounding computational cells used to construct a gradient plane.

technique is a multidimensional slope limiting method which constructs piecewise polynomial data from the initial piecewise constant data. The values of the reconstructed data are then calculated at the cell boundary of each side and these values are then used to solve the local Riemann problem. This procedure will now be outlined in more detail.

As stated above, we can attain a higher order of numerical accuracy for the solution of hyperbolic conservation laws by producing a higher order polynomial reconstruction of the solution in each computational cell. In this work we will produce a linear reconstruction of the conserved variables in each computational cell, given initial constant data. This reconstruction technique will lead to a numerical method which is second order accurate in space, that is a method for which the exact solution is obtained for linear data.

This process of reconstructing linear functions in each computational cell is outlined in Hubbard [54] for use on unstructured triangular and quadrilateral meshes. Let $\underline{\bar{u}}$ denote the constant cell average value of the solution in a computational cell, then we can produce a linear reconstruction within the computational cell in the following manner,

$$\underline{u} = \underline{\bar{u}} + \mathbf{r} \cdot \underline{\mathbf{L}}, \tag{6.21}$$

where $\mathbf{r}$ is the position vector relative to the centroid of the triangular computational cell and $\underline{\mathbf{L}}$ is a gradient operator. This gradient operator can be constructed by taking the three computational cells surrounding triangle 0 and forming a triangle with anti-clockwise ordering of its indexes. This triangle with vertices ijk formed from the centroids of the three surrounding computational cells is shown in figure 6.6. The gradient operator is then defined for each conserved variable u_l for $l = 1, \ldots m$ as

$$\nabla(\Delta ijk) = \begin{cases} \begin{pmatrix} -\frac{n_x}{n_{u_l}} \\ -\frac{n_y}{n_{u_l}} \end{pmatrix} & \text{for } n_{u_l} > \epsilon \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases} \tag{6.22}$$

where ϵ is a set tolerance taken to be 10^{-10}. The components of the vector $\mathbf{n} = (n_x, n_y, n_{u_l})$ come from

$$\mathbf{n} = (\mathbf{P}_i - \mathbf{P}_k) \times (\mathbf{P}_j - \mathbf{P}_k),$$

where

$$\mathbf{P}_z = (x_z, y_z, (u_l)_z) \qquad \text{for } z = i, j, k.$$

Choosing the linear operator $\underline{\mathbf{L}}$ to be the gradient operator (6.22) leads to a second-order accurate method. Given the linear reconstruction to the piecewise constant data in each computational cell we can then calculate the value of the conserved variables immediately to the left and right of a common edge; for example for the triangles denoted by 0 and j these values of the conserved variables are denoted by $\mathrm{u}_{0\,j}$ and $\mathrm{u}_{j\,0}$ and shown in figure 6.7. These values are calculated by taking the value of the position vector $\mathbf{r}$ in equation (6.21) to be the vector from the centroid of the cell to the midpoint of the edge. These values of the conserved variables at the common edge of the computational cells are then used as the initial values for solving the local Riemann problem in the HLLC approximate Riemann solver.

Although this procedure will lead to a method which is second order accurate in space it can cause the numerical method to create new extrema if the solution is rapidly varying.

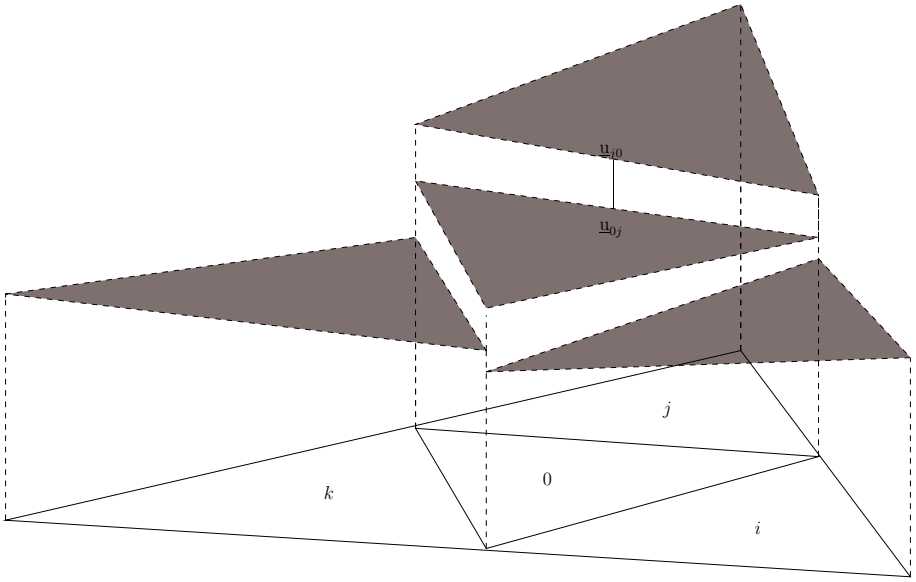

Figure 6.7: Figure showing the gradient planes constructed in a set of computational cells.

Therefore to eliminate the formation of unphysical oscillations we need to modify this gradient so that the scheme will satisfy a local maximum principle at the midpoints of the computational cells. The scheme can be made to satisfy a maximum principle if a limiting process is applied to the gradient planes which have been reconstructed.

This limiting process is carried out by applying a limiter function to the gradient operator to ensure that no local extrema are created at the intercell boundary. In this work we will use the Limited Central Differences (LCD) limiter described in [54]. The limited gradient plane is given as

$$\underline{\mathbf{L}}_{LCD} = \underline{\alpha}\,\underline{\mathbf{L}}$$

where $\underline{\alpha} = \min_{k=1,2,3} \underline{\alpha}_K$ and

$$\underline{\alpha}_K = \begin{cases} \frac{\max(\underline{u}_k - \underline{u}_0, 0)}{\mathbf{r}_{0\,k} \cdot \underline{\mathbf{L}}} & \text{if } \mathbf{r}_{0\,k} \cdot \underline{\mathbf{L}} > \max\left(\underline{u}_k - \underline{u}_0, 0\right), \\ \frac{\min(\underline{u}_k - \underline{u}_0, 0)}{\mathbf{r}_{0\,k} \cdot \underline{\mathbf{L}}} & \text{if } \mathbf{r}_{0\,k} \cdot \underline{\mathbf{L}} < \min\left(\underline{u}_k - \underline{u}_0, 0\right), \\ 1 & \text{otherwise.} \end{cases}$$

Since the resulting finite volume scheme with the HLLC and MUSCL linear reconstruction is explicit in time it must satisfy a CFL type condition. It has been shown that

schemes using this type of limiting process will not produce new local extrema at the cell edge mid-points if the time-step satisfies the condition that

$$\Delta t^n \leq \frac{|\Delta_i|}{3 \max\limits_{k=1,3} |\lambda_k \cdot \mathrm{n}_k|}, \tag{6.23}$$

where λ_k is the wave speed of the system being solved across the edge k of the computational cell Δ_i.

We have now outlined the use of a high resolution method for the solution of hyperbolic conservation laws on an unstructured adaptively moving mesh. In the following sections we will describe the use of the moving mesh method detailed in chapter 3, using both the mass monitor function and a geometric monitor function.

6.5 A Mass Monitor Function

In this section we describe the use of the mass monitor function in the moving mesh method that was derived in chapter 3 to generate an adaptively moving mesh for the solution of the two dimensional compressible Euler equations of gas dynamics. The mass monitor which was used for the solution of one dimensional hyperbolic conservation laws in the previous chapter is given as $M = \rho$. If this monitor function is substituted into the moving mesh velocity potential equation given by (3.21), taking $\mathbf{q} = 0$ and $\varpi = 1$, we obtain the equation around the patch of computational cells $\partial\Omega_i$

$$\oint_{\partial\Omega_i(t)} w_i \, \rho \, \frac{\partial \Phi}{\partial \mathbf{n}} \cdot \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \rho \, \nabla \Phi \, \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, \frac{\partial \rho}{\partial t} \, \mathrm{d}\Omega.$$

The boundary integral around the patch of computational cells is equal to zero since the basis function w_i is equal to zero on this boundary and on the boundary of the domain we have the boundary condition $\frac{\partial \Phi}{\partial \mathbf{n}} = 0$. Therefore the velocity potential equation for Φ becomes

$$\int_{\Omega_i(t)} \nabla w_i \cdot \rho \, \nabla \Phi \, \mathrm{d}\Omega = \int_{\Omega_i(t)} w_i \, \frac{\partial \rho}{\partial t} \, \mathrm{d}\Omega.$$

We now need to obtain an expression for ρ_t to substitute into the above equation. This relationship for the rate of change in time of the density function comes from the conservation of mass equation, which is the first equation in the system of Euler equations (6.1) and given as $\rho_t = -\nabla \cdot (\rho \, \mathbf{v})$. Therefore substituting this expression into the velocity potential equation we obtain

$$\int_{\Omega_i(t)} \nabla w_i \cdot \rho \, \nabla \Phi \, \mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \, \nabla \cdot (\rho \, \mathbf{v}) \, \mathrm{d}\Omega.$$

This equation for the approximate velocity potential Φ becomes a stiffness matrix system once the linear finite element approximations have been made. This stiffness matrix system has the form

$$K \underset{\sim}{\Phi} = \underset{\sim}{f},$$

where the elements of the stiffness matrix are given as

$$K_{i\,j} = \int_{\Omega_i(t)} \nabla w_i \cdot \nabla w_j \, \rho \, \mathrm{d}\Omega$$

and the elements of the load vector are

$$f_i = - \int_{\Omega_i(t)} w_i \, \nabla \cdot (\rho \, \mathbf{v}) \, \mathrm{d}\Omega.$$

Once the numerical approximation to the mesh velocity potential has been obtained, the mesh velocity can then be recovered weakly by solving the mass matrix system (3.22). The mesh is then integrated forward in time using a standard forward Euler time-stepping routine. The ALE velocity resulting from this choice of the monitor function can then be used in the 2D ALE finite volume method that has been outlined in sections 6.2, 6.3 and 6.4 to update the solution variables. Results generated from this choice of monitor function will be shown in section 6.7.

6.6 A Geometric Monitor Function

In this section we will outline the use of the geometric monitor function $M = 1 + \alpha \, |\nabla \rho|^2$ in the moving mesh method derived in chapter 3. This monitor function will be used to

generate an adaptive mesh on which the two-dimensional compressible Euler equations can be solved. If this monitor function is substituted into the moving mesh velocity potential equation given by (3.21), taking $\mathbf{q} = 0$ and $\varpi = 1$, we obtain the equation

$$\oint_{\partial\Omega_i(t)} w_i \left(1 + \alpha \left|\nabla\rho\right|^2\right) \frac{\partial\phi}{\partial\mathbf{n}} \cdot \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \left(1 + \alpha \left|\nabla\rho\right|^2\right) \nabla\phi \,\mathrm{d}\Omega = - \int_{\Omega_i(t)} w_i \left(1 + \alpha \left|\nabla\rho\right|^2\right)_t \mathrm{d}\Omega$$

on the patch of computational cells $\partial\Omega_i$. We now need to obtain an expression for the rate of change of the monitor function

$$\left(1 + \alpha \left|\nabla\rho\right|^2\right)_t = \alpha \left(\left|\nabla\rho\right|^2\right)_t .$$

Expanding the differentiation of time and reversing the order of spatial and temporal differentiation we obtain

$$\begin{aligned} \alpha \left(\left|\nabla\rho\right|^2\right)_t &= 2\alpha \,\nabla\rho \cdot (\nabla\rho)_t \\ &= 2\alpha \,\nabla\rho \cdot \nabla\rho_t \end{aligned} \tag{6.24}$$

We now need to obtain an expression for ρ_t to substitute into the above equation. This relationship for the rate of change in time of the density function comes from the conservation of mass equation, which is the first equation in the system of Euler equations (6.1) and given as $\rho_t = -\nabla \cdot (\rho\,\mathbf{v})$. Therefore substituting this expression into equation (6.24) we obtain

$$2\alpha \,\nabla\rho \cdot \nabla\rho_t = -2\alpha \,\nabla\rho \cdot \nabla \left(\nabla \cdot (\rho\,\mathbf{v})\right).$$

Thus we have obtained the relationship that

$$\left(1 + \alpha \left|\nabla\rho\right|^2\right)_t = -2\alpha \,\nabla\rho \cdot \nabla \left(\nabla \cdot (\rho\,\mathbf{v})\right)$$

which can be weakly enforced and substituted into equation (5.30) to give

$$\oint_{\partial\Omega_i(t)} w_i \left(1+\alpha\,|\nabla\rho|^2\right) \frac{\partial\phi}{\partial\mathbf{n}} \cdot \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \left(1+\alpha\,|\nabla\rho|^2\right) \nabla\phi \,\mathrm{d}\Omega = 2\alpha \int_{\Omega_i(t)} w_i \,\nabla\rho\cdot\nabla\left(\nabla\cdot(\rho\,\mathbf{v})\right)\,\mathrm{d}\Omega.$$

The variables ρ and $\mathbf{v}$ will both be approximated by piecewise linear functions, so the integral on the right-hand side of this equation has to be integrated by parts since the momentum $\rho\,\mathbf{v}$ will be a quadratic function and the second derivatives will not be well-defined. The right hand side of this equation can be dealt with by writing this integral as a sum of integrals over each cell in the patch. Therefore we have that

$$\int_{\Omega_i(t)} w_i \,\nabla\rho\cdot\nabla\left(\nabla\cdot(\rho\,\mathbf{v})\right)\,\mathrm{d}\Omega = \sum_{j}^{\Delta_i' s\, in\, \Omega_i(t)} \int_{\Delta_j(t)} w_i \,\nabla\rho\cdot\nabla\left(\nabla\cdot(\rho\,\mathbf{v})\right)\,\mathrm{d}\Omega$$

We will assume that $\nabla\rho$ is constant over the cell and therefore can be taken out of the integral to give

$$\sum_{j}^{\Delta_i' s\, in\, \Omega_i(t)} \int_{\Delta_j(t)} w_i \,\nabla\rho\cdot\nabla\left(\nabla\cdot(\rho\,\mathbf{v})\right)\,\mathrm{d}\Omega = \sum_{j}^{\Delta_i' s\, in\, \Omega_i(t)} \nabla\rho\cdot\int_{\Delta_j(t)} w_i \,\nabla\left(\nabla\cdot(\rho\,\mathbf{v})\right)\,\mathrm{d}\Omega.$$

This integral may now be integrated by parts to give

$$\sum_{j}^{\Delta_i' s\, in\, \Omega_i(t)} \nabla\rho\cdot\left\{\oint_{\partial\Delta_j(t)} w_i \,\nabla\cdot(\rho\,\mathbf{v})\,\mathrm{d}\mathbf{\Gamma} - \int_{\Delta_j(t)} \nabla w_i \,\nabla\cdot(\rho\,\mathbf{v})\,\mathrm{d}\Omega\right\}$$

Once the finite element approximations to the density, velocity and the velocity potential denoted by ρ, $\mathbf{v}$ and Φ respectively have been introduced and expanded in terms of linear basis functions the equation for the approximation to the mesh velocity potential becomes a stiffness matrix system given by

$$K\,\underset{\sim}{\Phi} = \underset{\sim}{f},$$

where the elements of the stiffness matrix are given as

$$K_{i\,j} = \oint_{\partial\Omega_i(t)} w_i \left(1+\alpha\,|\nabla\rho|^2\right) \frac{\partial w_j}{\partial\mathbf{n}} \cdot \mathrm{d}\mathbf{\Gamma} - \int_{\Omega_i(t)} \nabla w_i \cdot \left(1+\alpha\,|\nabla\rho|^2\right) \nabla w_j \,\mathrm{d}\Omega$$

and the elements of the load vector $\underset{\sim}{f}$ are

$$f_i = 2\alpha \sum_{j}^{\Delta_i' s \, in \, \Omega_i(t)} \nabla\rho \cdot \left\{ \oint_{\partial\Delta_j(t)} w_i \, \nabla \cdot (\rho \, \mathbf{v}) \, \mathrm{d}\mathbf{\Gamma} - \int_{\Delta_j(t)} \nabla w_i \, \nabla \cdot (\rho \, \mathbf{v}) \, \mathrm{d}\Omega \right\}.$$

Once the numerical approximation to the mesh velocity potential has been obtained, the mesh velocity can then be recovered weakly by solving the mass matrix system (3.22). The mesh is then integrated forward in time using a standard forward Euler time-stepping routine given by

$$\underset{\sim}{\mathbf{x}}^{n+1} = \underset{\sim}{\mathbf{x}}^{n} + \Delta t^n \, \dot{\underset{\sim}{\mathbf{x}}}^{n}.$$

The velocity generated from this choice of the monitor function can then be used in the 2D ALE finite volume method that has been outlined in sections 6.2, 6.3 and 6.4 to update the solution variables. Results generated from this choice of monitor function will be shown in section 6.7.

6.7 Numerical Results VI

In this section we will show results obtained for the two dimensional compressible Euler equations of gas dynamics using the HLLC approximate Riemann solver on a genuinely two-dimensional adaptive mesh generated by the method derived in chapter 3. As our test problem we have chosen to solve the cylindrical generalisation of Sod's shock tube problem. The problem consists of a circular region with radius $\frac{1}{2}$ containing gas at high pressure and density which is initially at rest. Outside of this circular region the gas is at low pressure and density and is also at rest. At $t = 0$ the membrane between the two regions of gas is removed and the two regions of gas are allowed to move into one another. The resulting solution to the problem has the same structure in the radial direction r as the one-dimensional Sod shock tube problem. Therefore the solution consists of a shock and contact wave which move away from the high pressure, density region and a rarefaction wave travelling into the high pressure, density region. The solution to the problem differs slightly from the one-dimensional Sod shock tube problem in that the solution has depressions caused by the cylindrical geometry of the problem. The initial conditions for this problem are given as

$$(\rho,\, u,\, v,\, p) = \begin{cases} (1.0, 0.0, 0.0, 1.0) & \text{for } 0 \leq r \leq \frac{1}{2} \\ (0.125, 0.0, 0.0, 0.1) & \text{for } \frac{1}{2} < r < 1 \end{cases} \tag{6.25}$$

In all the numerical simulations we will give the solutions at $t = 0.2$.

This problem was first solved using the HLLC method on an Eulerian mesh with 20000 computational cells and 10201 nodes and can be seen in figure 6.8. The numerical solution obtained for this problem is shown in figure 6.9 and shows plots of the density, velocity components in the x and y directions, respectively, pressure and the specific internal energy of the gas. There does not exist an exact solution to this problem in closed form, so as a reference solution we solved the problem as a one-dimensional radially symmetric problem. The compressible Euler equations written in radial co-ordinates r are given as

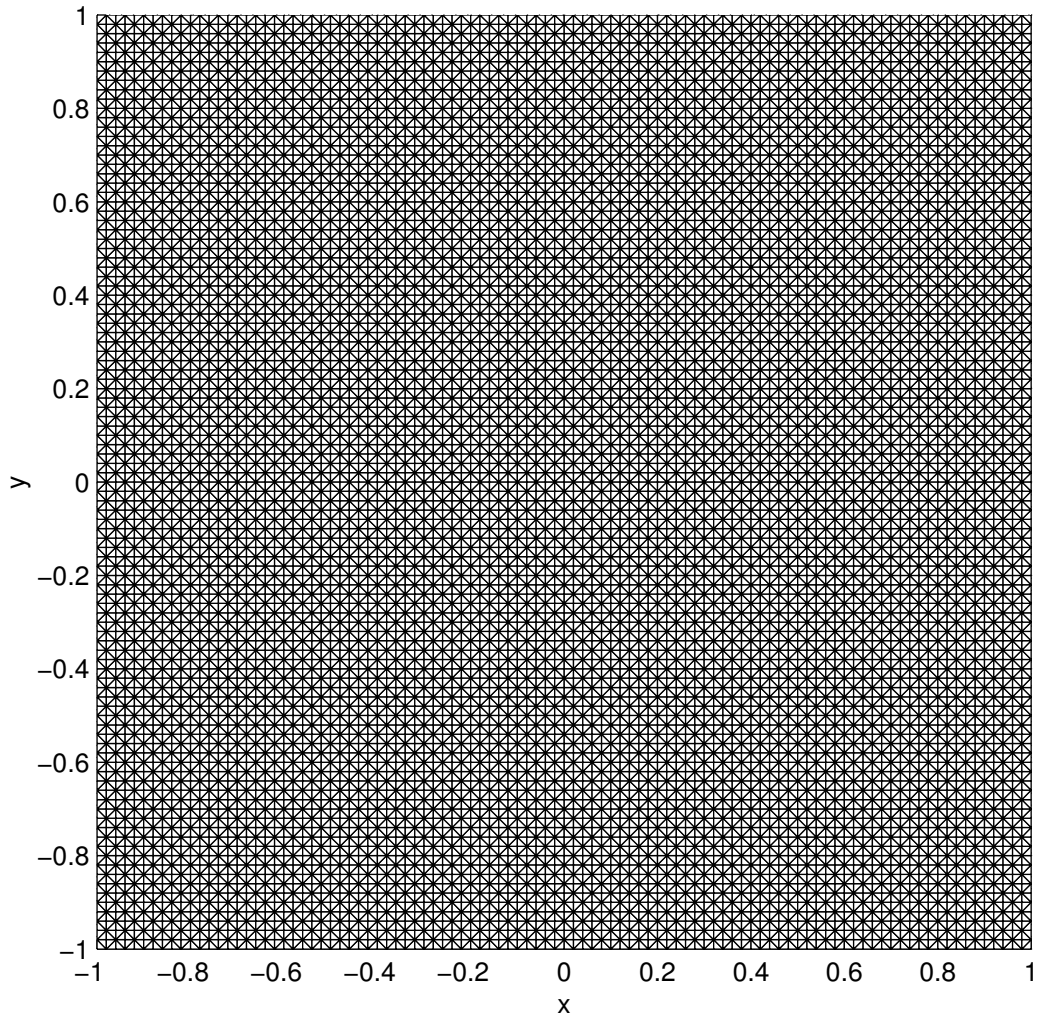

Figure 6.8: Initial mesh generated for the Eulerian computations.

$$\begin{pmatrix} \rho \\ \rho\, v_{rad} \\ E \end{pmatrix}_t + \begin{pmatrix} \rho\, v_{rad} \\ \rho\, v_{rad}^2 + p \\ v_{rad}\,(E+p) \end{pmatrix}_r = -\frac{1}{r} \begin{pmatrix} \rho\, v_{rad} \\ \rho\, v_{rad}^2 \\ v_{rad}\ (E+p) \end{pmatrix}$$

where v_{rad} is the radial velocity of the gas. This system of conservation laws was then solved using a one-dimensional Roe approximate Riemann solver with 5000 mesh points to produce a comparative "exact" solution. The two-dimensional solutions obtained were plotted against the radial distance and compared with the one-dimensional reference solution. These results can be seen in figure 6.10.

The same problem was then solved using the moving mesh method along with the mass monitor function $M = \rho$. The initial mesh for the computation was equally spaced and is shown in figure 6.8. The solution on the moving mesh is shown in figure 6.11

	$\min \lvert\Delta_i\rvert$	$\max \lvert\Delta_i\rvert$	$\frac{\max\lvert\Delta_i\rvert}{\min\lvert\Delta_i\rvert}$
Eulerian	2.0×10^{-4}	2.0×10^{-4}	1
Lagrangian	3.6343×10^{-5}	6.0584×10^{-4}	16.6699
Geometric	5.6867×10^{-5}	5.2327×10^{-4}	9.2017

Table 6.1: Table showing the maximum and minimum size of computational mesh cells in the final mesh at $t = 0.2$.

and again these solutions are plotted against the radial distance so that they could be compared with the reference solution. The final mesh at $t = 0.2$ for this computation is also shown in figure 6.13. It can be seen from this figure that the computational cells are compressed in regions of the solution where the gas is also compressed. Therefore many of the mesh points tend to cluster in the spatial region between the contact and shock waves.

The cylindrical shock problem was then solved on a moving mesh obtained from the use of the moving mesh method with the monitor function $M = 1 + \alpha \left|\nabla\rho\right|^2$. In the simulations the constant α was taken to be

$$\alpha = \frac{\beta}{\max\limits_{x} \left|\nabla\rho\right|^2},$$

where β is a second constant which determines the degree of mesh adaption. In numerical computations it was found that we obtained adequate numerical results with $\beta = 20$.

Solutions for the compressible Euler equations for this choice of monitor function can be seen in figure 6.14. The solution obtained has also been plotted against the radial distance and is shown in figure 6.15. The final mesh for the problem at $t = 0.2$ has also been displayed in figure 6.16 and shows mesh points being clustered in regions of high density gradient, such as in the shock and contact waves and also in the rarefaction wave.

In table 6.1 the maximum and minimum size of the computational cells for the three different methods is shown and gives us an indication of the stiffness of the mass matrices

being solved. The stiffness of the mass matrix systems is known to be proportional to the relative size of the largest to the smallest computational cell [97]. Therefore

$$\kappa = \frac{\max |\Delta_i|}{\min |\Delta_i|}$$

We find that the moving mesh method with the mass monitor function produces the stiffest matrices to be solved.

As a final test problem for the two-dimensional Euler equations we considered a converging cylindrical shock problem. Initially at $t = 0$ the problem comprises of a circular region with radius 100 containing gas at low pressure and density which is initially at rest. Outside of this circular region the gas is at high pressure and density and is also at rest. At $t = 0$ the membrane between the two regions of gas is removed and the two regions of gas are allowed to move into one another. The solution then consists of a converging shock wave followed by a converging contact wave and a diverging rarefaction wave. The shock then reflected at the origin of the domain and shock then interacts with the converging contact wave. The result of this interaction is a diverging shock wave, a converging contact wave and also a weak shock wave converging towards the origin. The initial conditions for this problem are taken to be

$$(\rho,\, u,\, v,\, p) = \begin{cases} (1.0, 0.0, 0.0, 1.0) & \text{for } 0 \leq r \leq 100 \\ (4.0, 0.0, 0.0, 4.0) & \text{for } 100 < r < 250 \end{cases} \tag{6.26}$$

The problem was solved by Glaister in [41] as a one-dimensional cylindrically symmetric problem. The results for this problem were computed until $t = 90$ when the shock and contact waves interact on the domain $[-250, 250] \times [-250, 250]$. This problem was simulated with the three methods used here and results are shown for the Eulerian method in figures 6.17 and 6.18. The results obtained for the moving mesh method together with the mass and the geometric monitor functions are shown in figures 6.19. 6.20 and 6.22, 6.22 respectively. The final meshes for the moving mesh computational are shown in figures 6.21 and 6.24 for the mass and geometric monitor functions.

On comparing the three solutions obtained for this problem we find that the moving mesh method together with the geometric monitor function leads to a more accurate

numerical solution. The region in which the shock and contact waves interact is better resolved and hence gives a better numerical solution in this region. Also with this type of mesh movement we have found that multidimensional effects are less pronounced, therefore the solution has retained its cylindrical symmetry better in comparison with both the Eulerian method and the moving mesh method with the mass monitor function. It can also be seen from the figures that none of the methods resolve the fine features of the solution. The problem was also solved with the Eulerian method on a finer mesh which has 40401 nodes and 80000 computational cells. The solutions obtained for this simulation are shown in figure 6.25 and show that, even when the number of computational cells had been quadrupled, the solution to the problem was still possibly not as accurate as the moving mesh method together with the geometric monitor function. This Eulerian solution is still very smeared in the region of the shock, contact wave interaction. The peak in the density in this region is much higher in comparison with the moving mesh method, but it seems to be in the wrong position. This error in the position and the smearing of the shock is probably due to a break in the symmetry of the problem as the initial shock wave is reflected off the origin of the domain. The moving mesh method with the geometric monitor function obtains a better solution due to the fact that is it able to cluster mesh points tightly at the origin when the shock is reflected and hence symmetry errors are kept to a minimum.

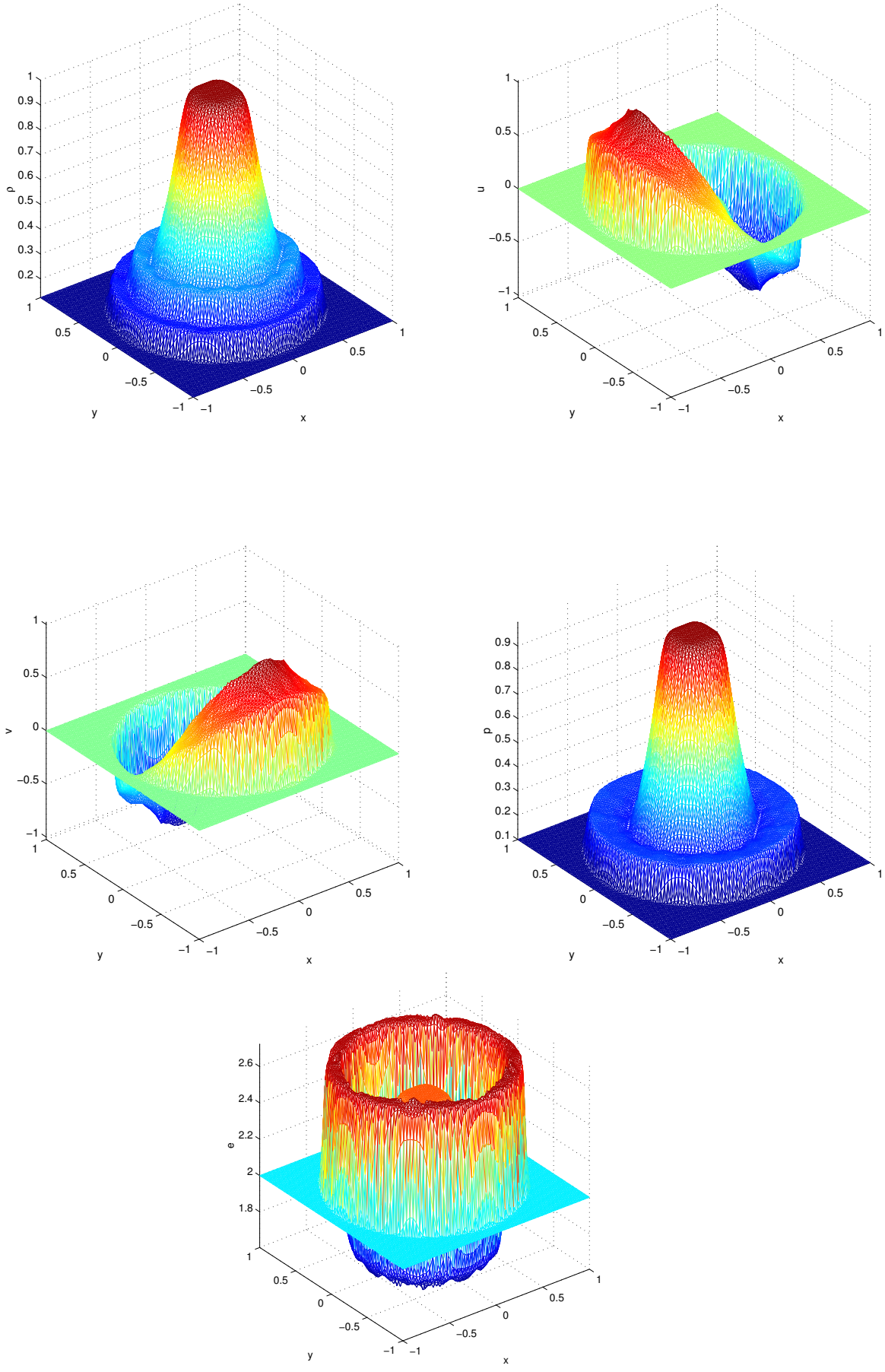

Figure 6.9: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the Eulerian method.

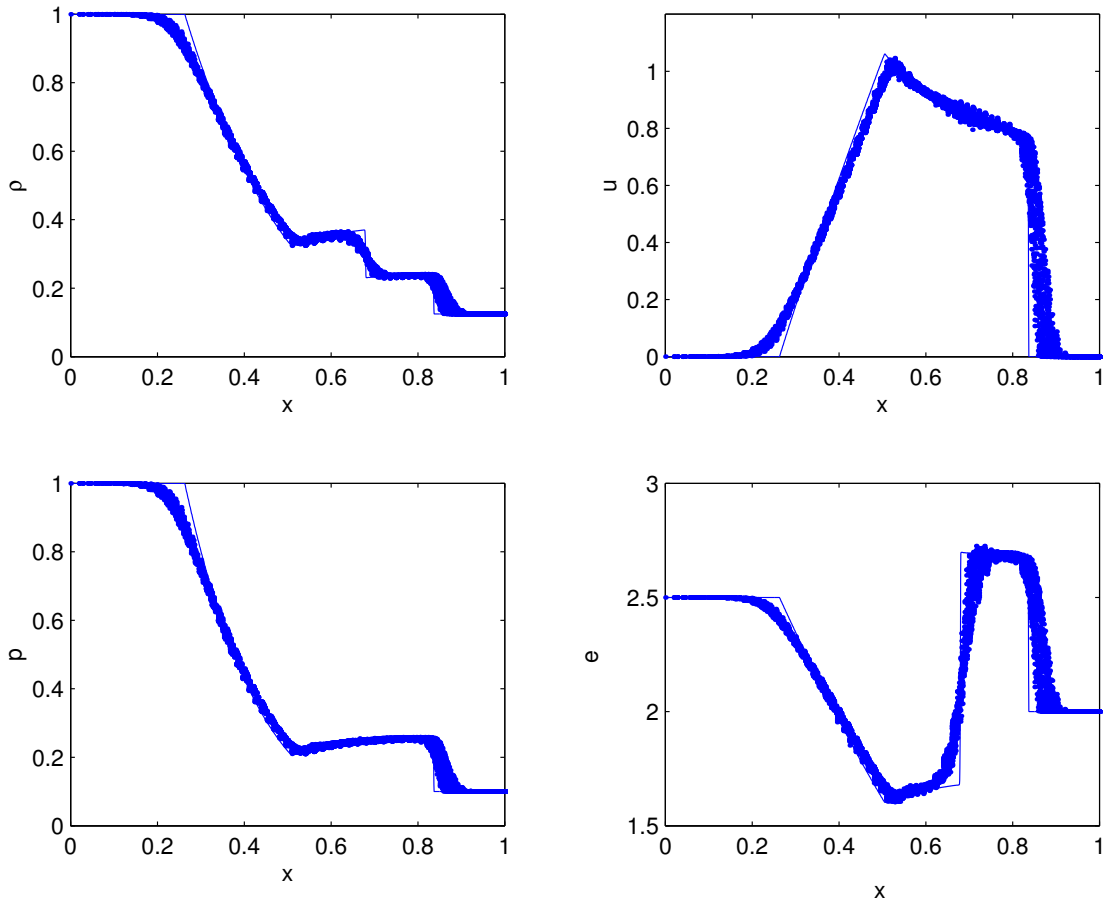

Figure 6.10: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$*, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)*

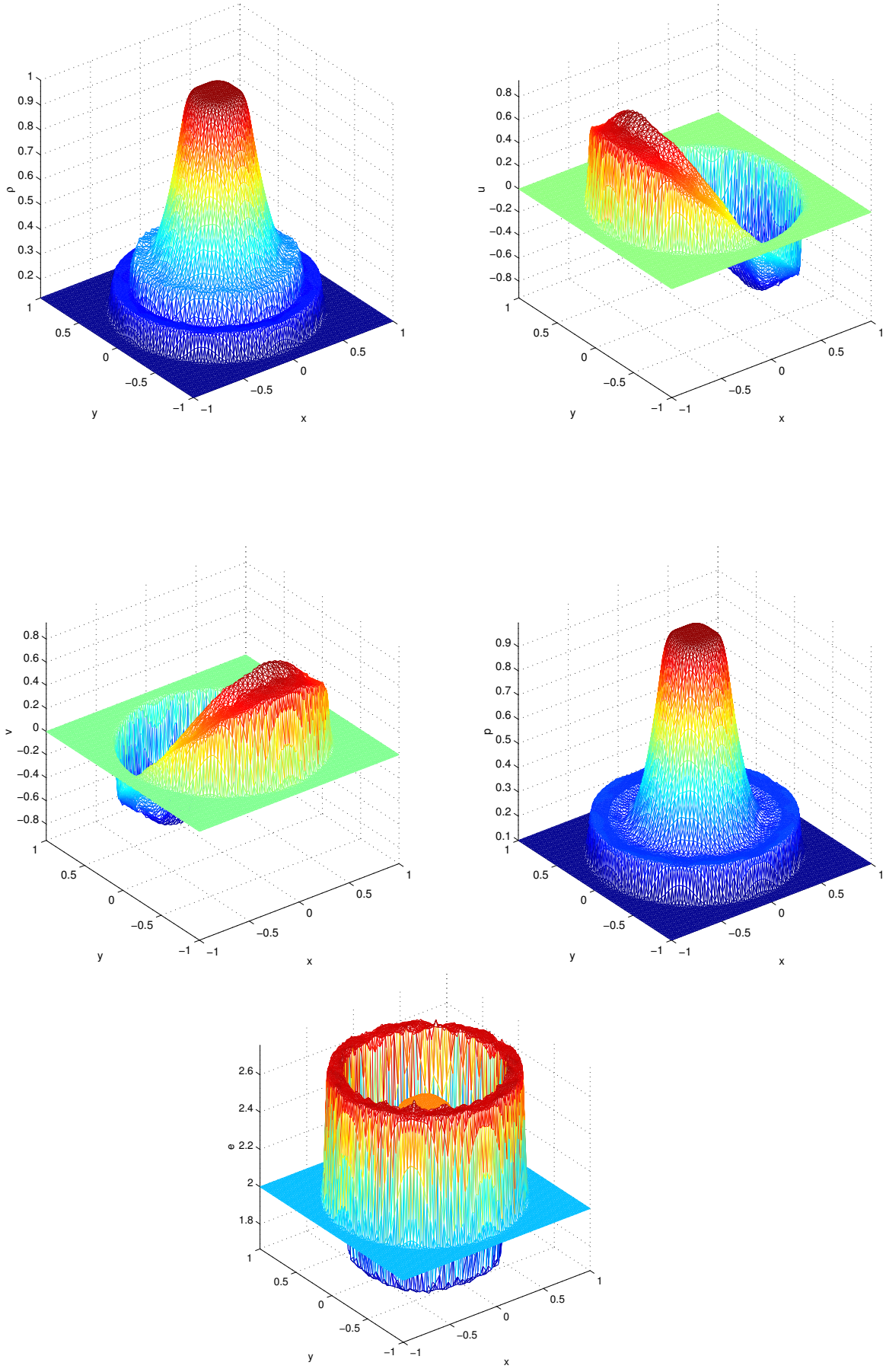

Figure 6.11: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the moving mesh method with $M = \rho$.

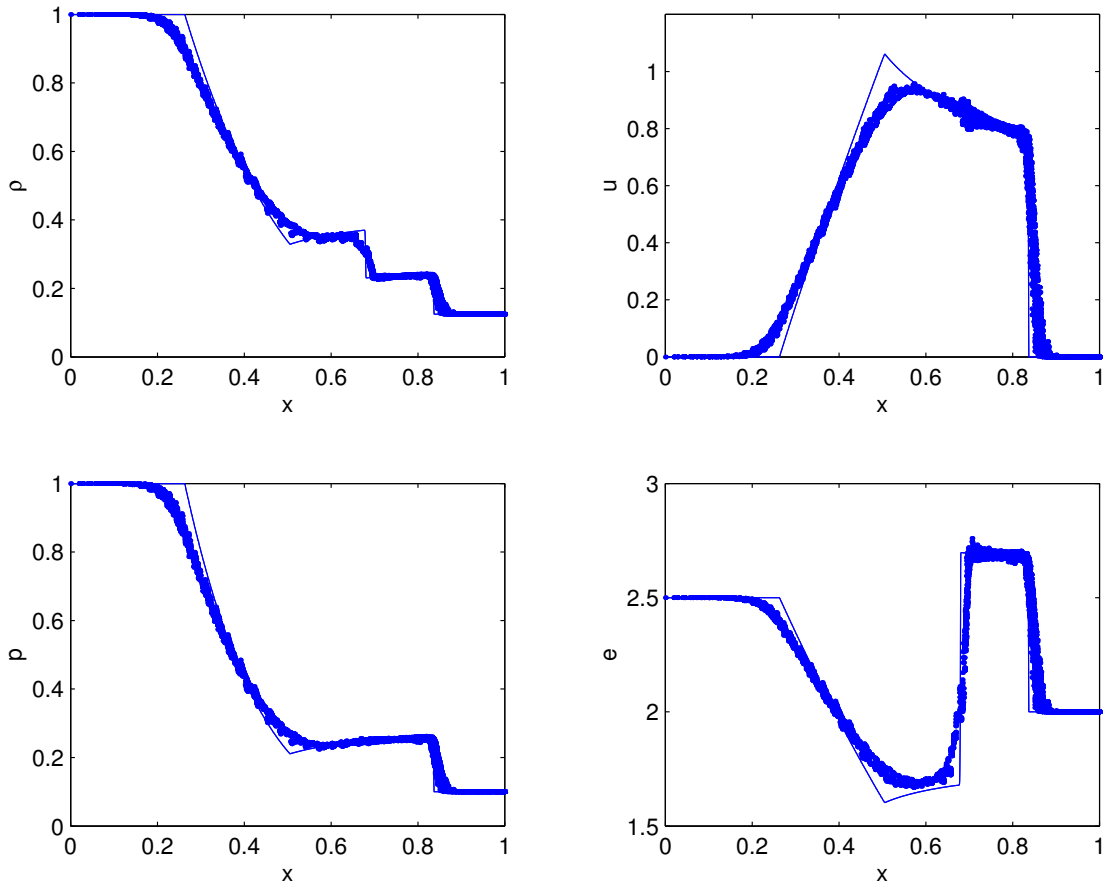

Figure 6.12: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)

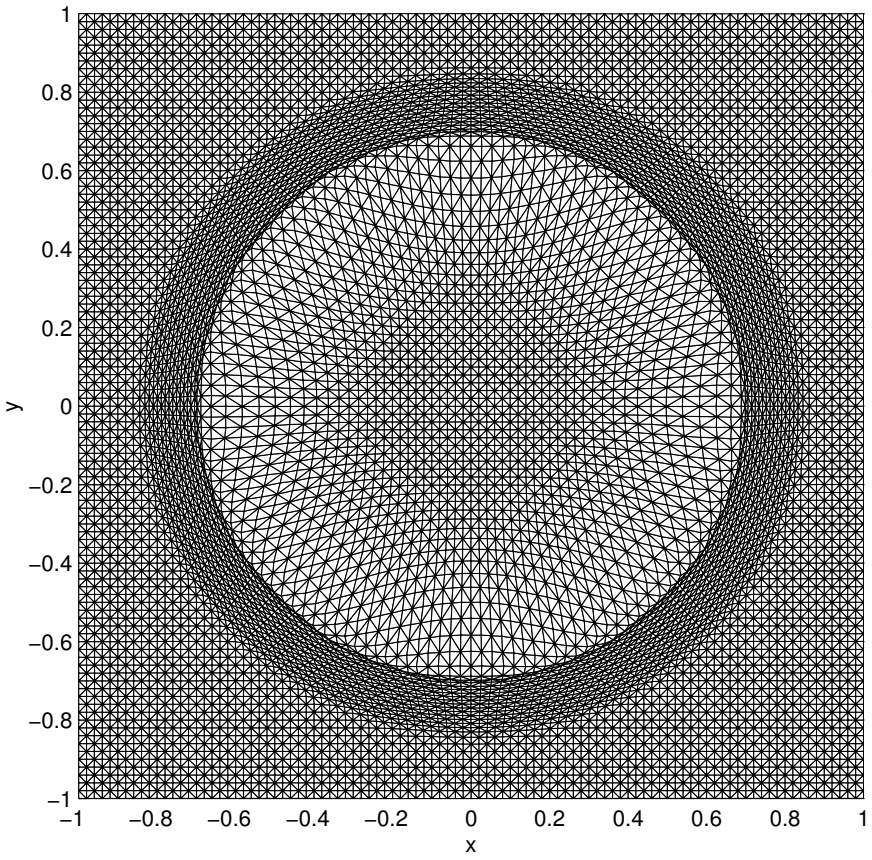

Figure 6.13: Plot of the final mesh at $t = 0.2$ obtained for the solution of the Euler equations with the moving mesh method with $M = \rho$.

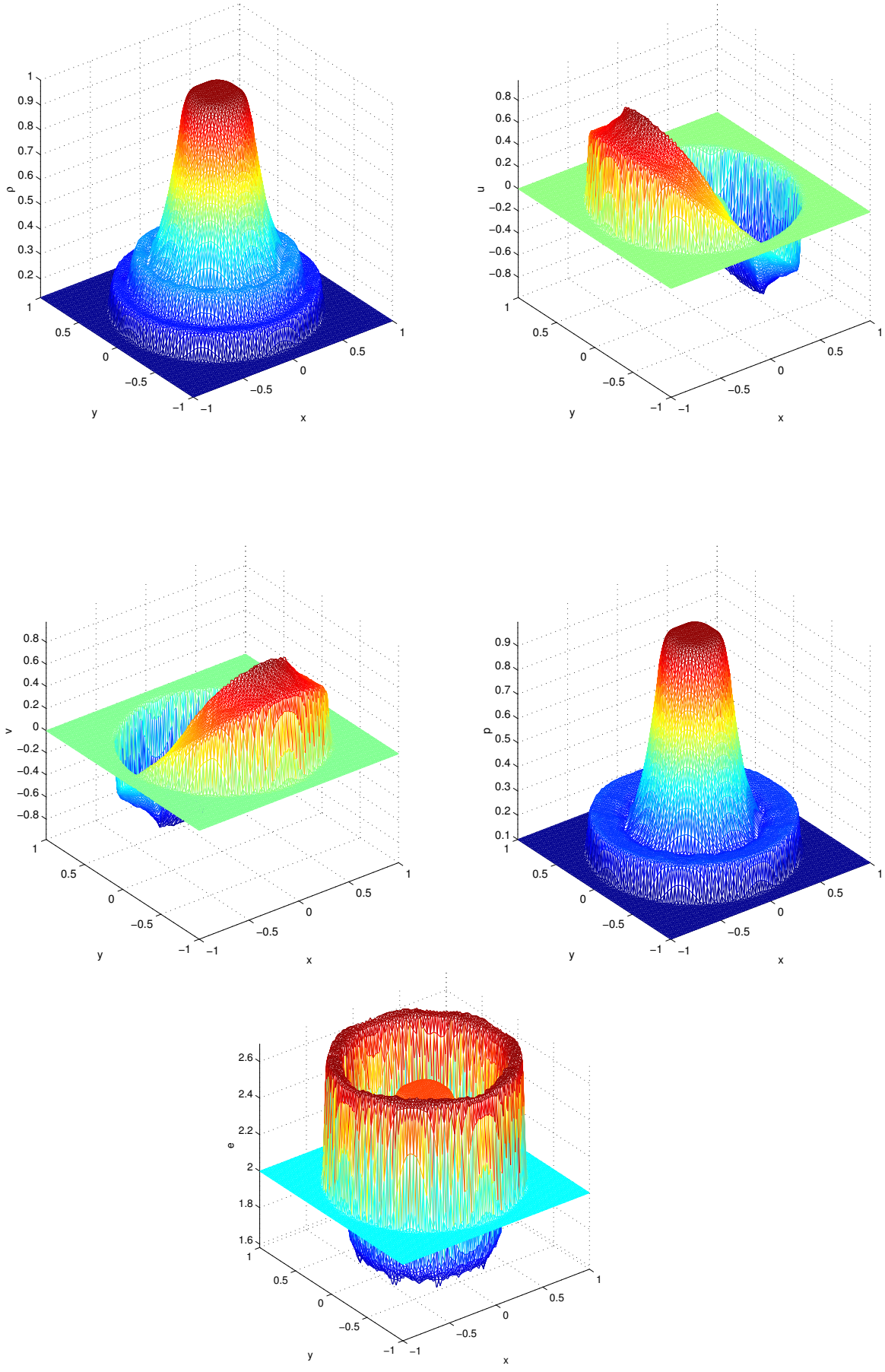

Figure 6.14: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the moving mesh method with $M = 1 + \alpha\ |\nabla\rho|^2$.

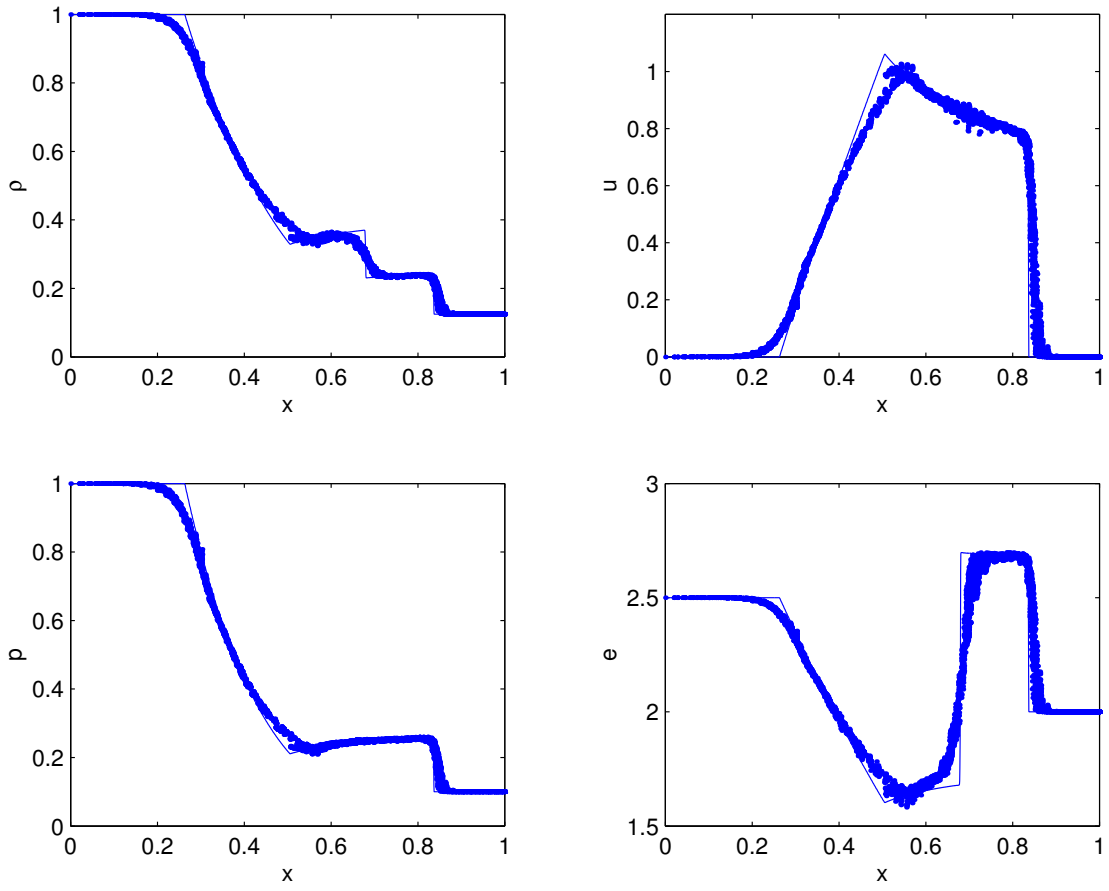

Figure 6.15: Solution of the Euler equations with initial data (6.25). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$*, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)*

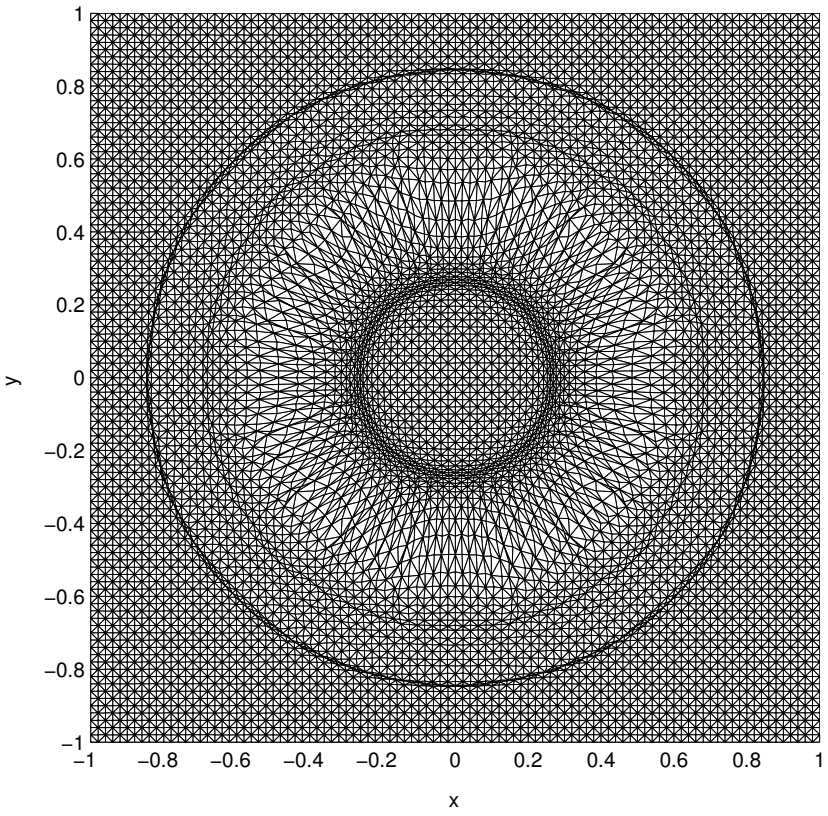

Figure 6.16: Plot of the final mesh at $t = 0.2$ obtained for the solution of the Euler equations with the moving mesh method with $M = 1 + \alpha \, |\nabla \rho|^2$.

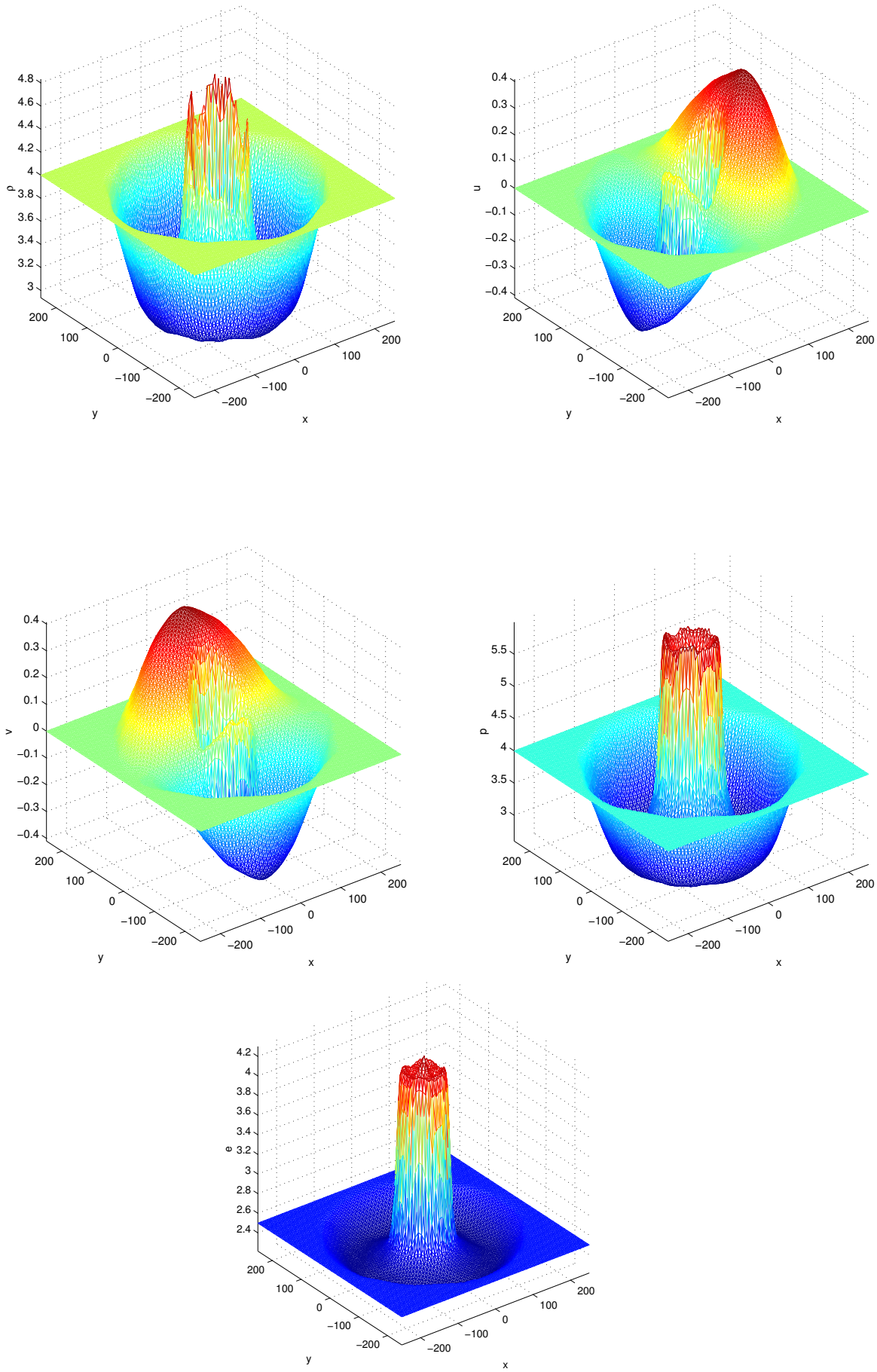

Figure 6.17: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the Eulerian method.

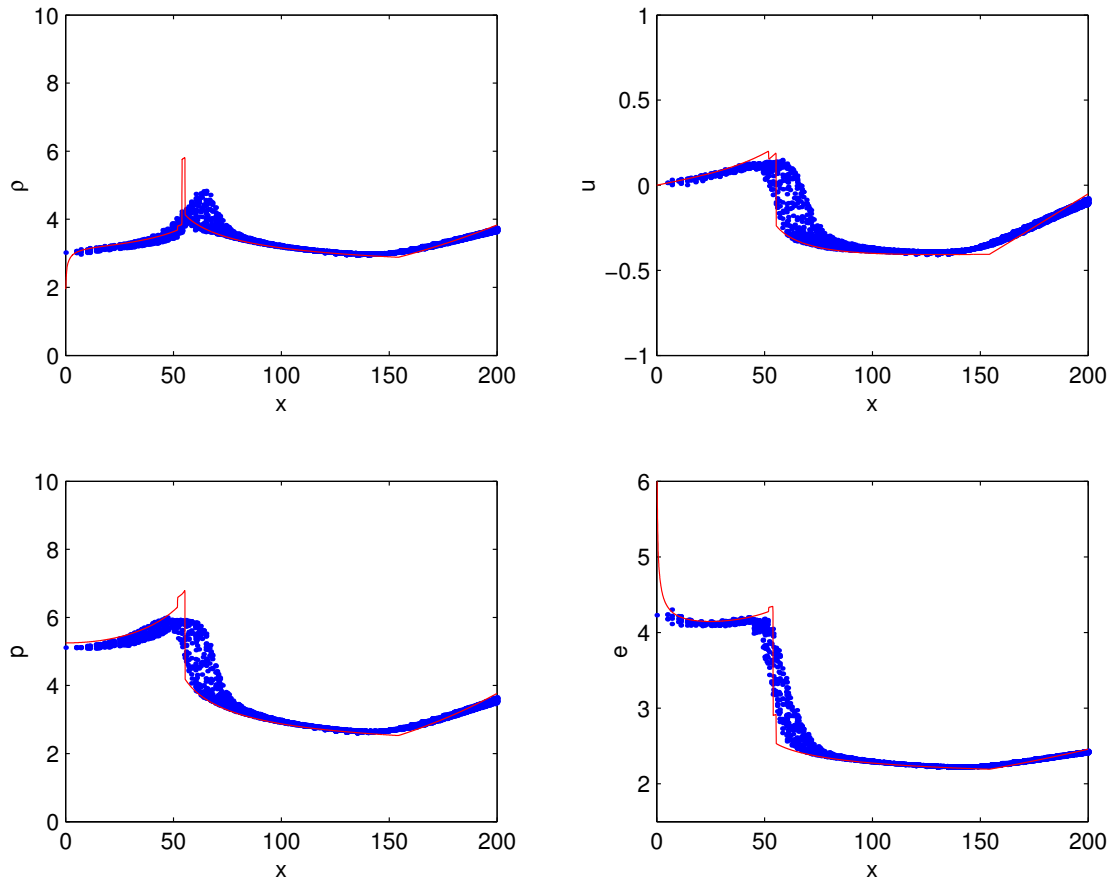

Figure 6.18: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$*, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)*

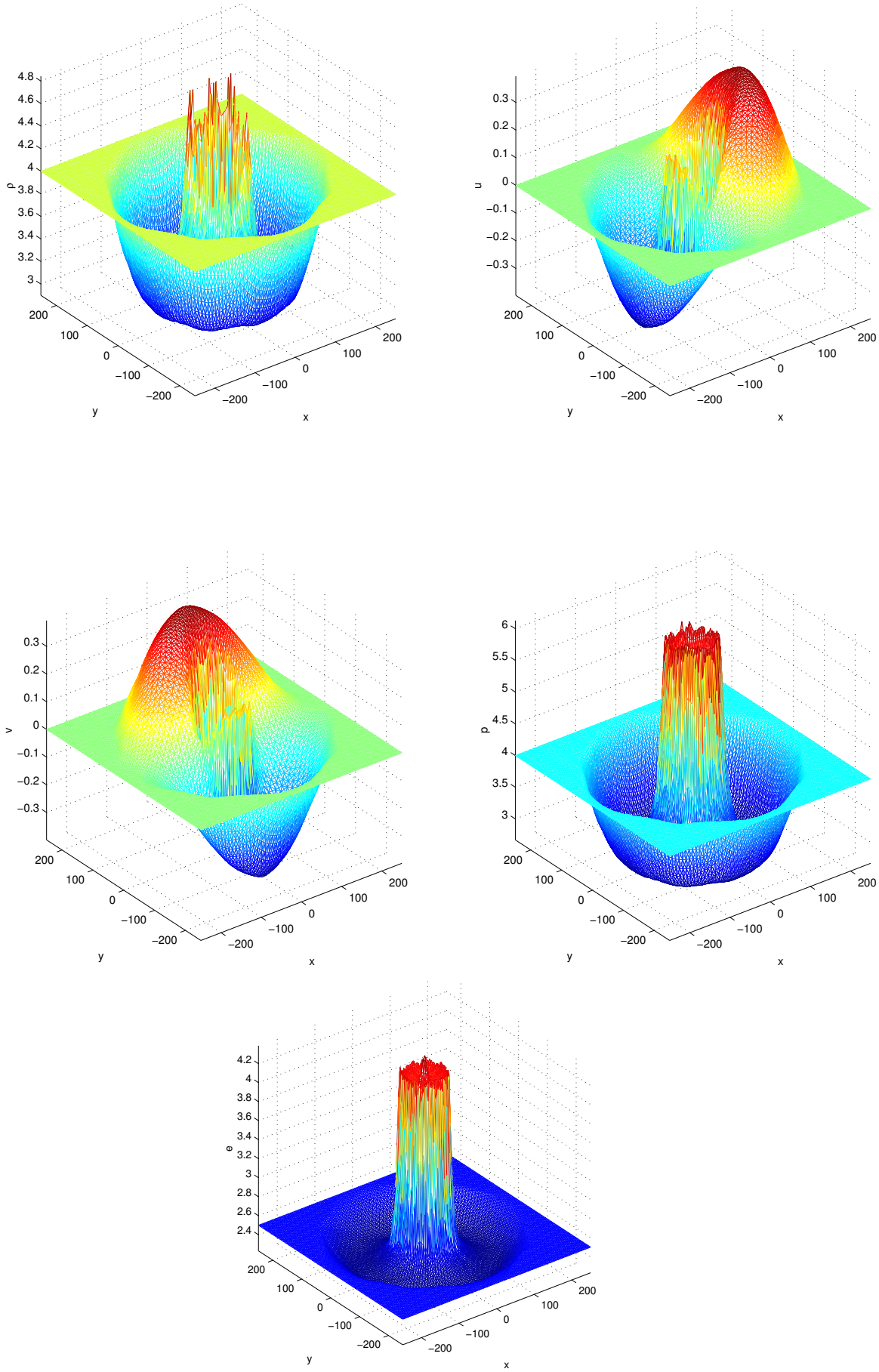

Figure 6.19: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the moving mesh method with $M = \rho$.

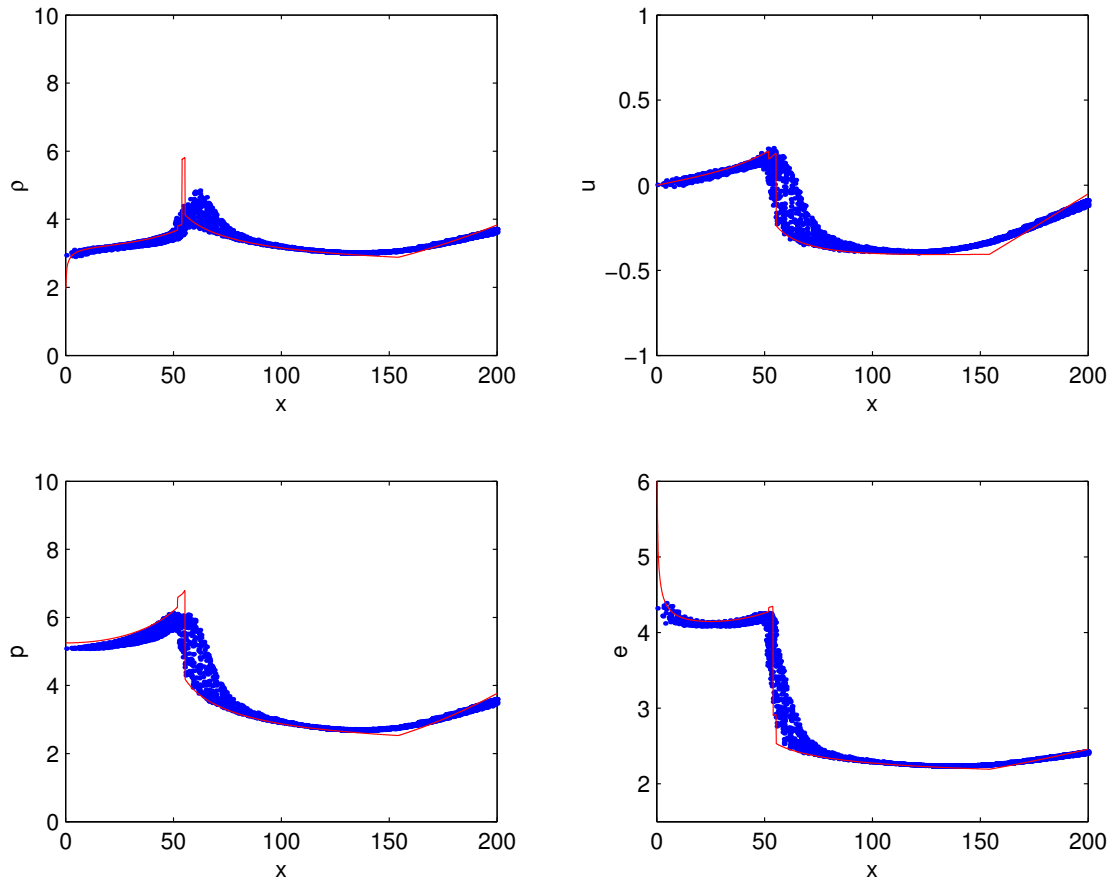

Figure 6.20: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$*, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)*

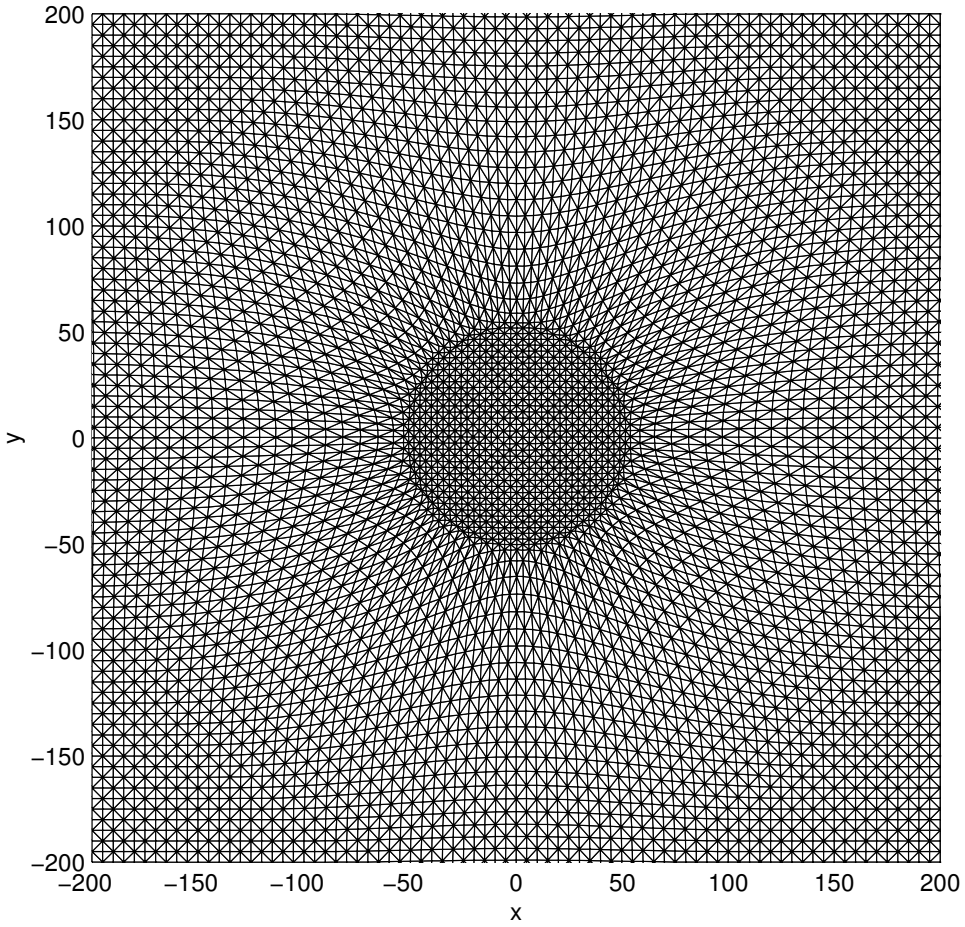

Figure 6.21: Plot of the final mesh at $t = 90$ obtained for the solution of the Euler equations with the moving mesh method with $M = \rho$.

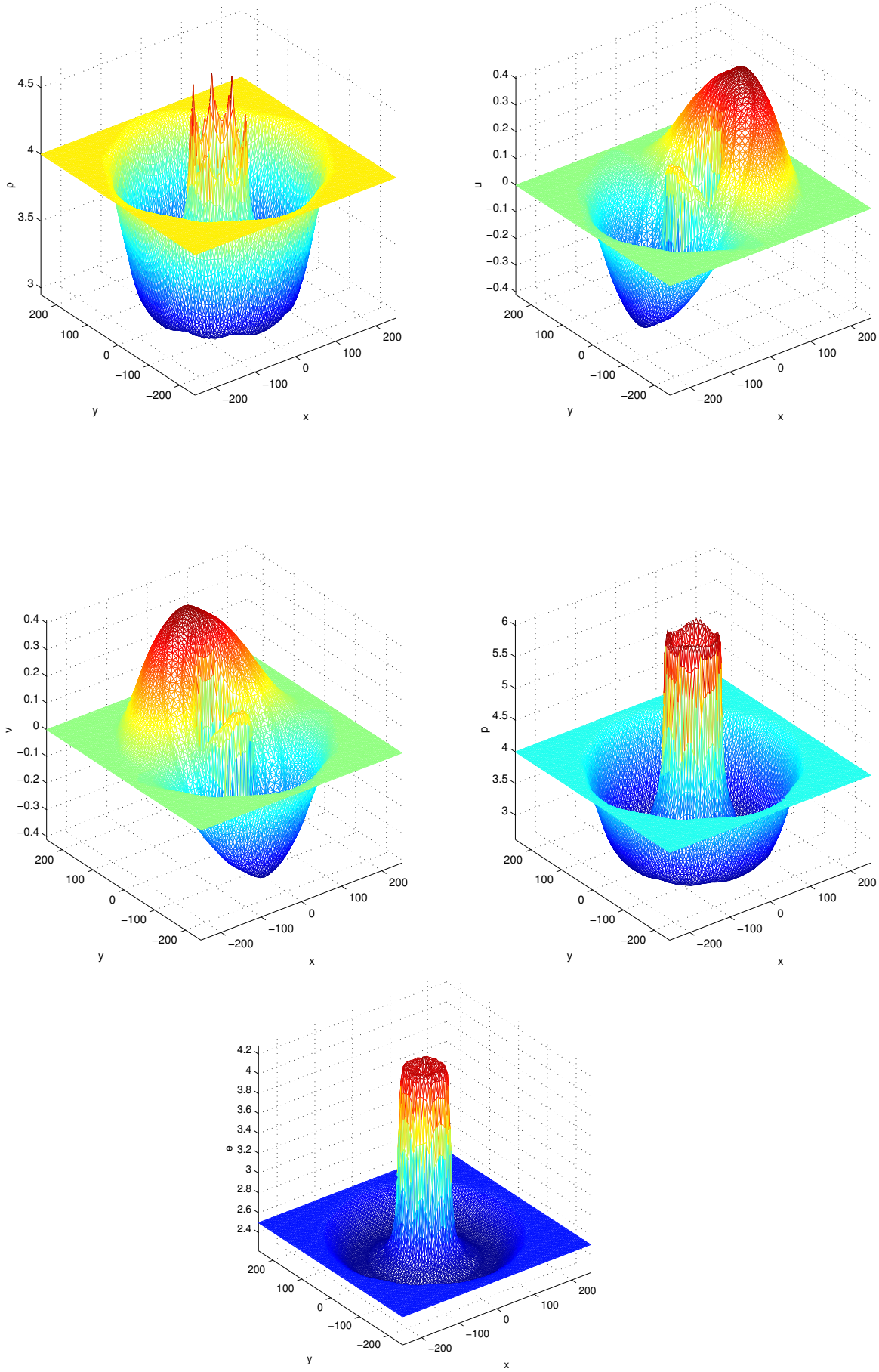

Figure 6.22: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, velocity components in x and y directions, pressure and specific internal energy computed with the moving mesh method with $M = 1 + \alpha \, |\nabla \rho|^2$.

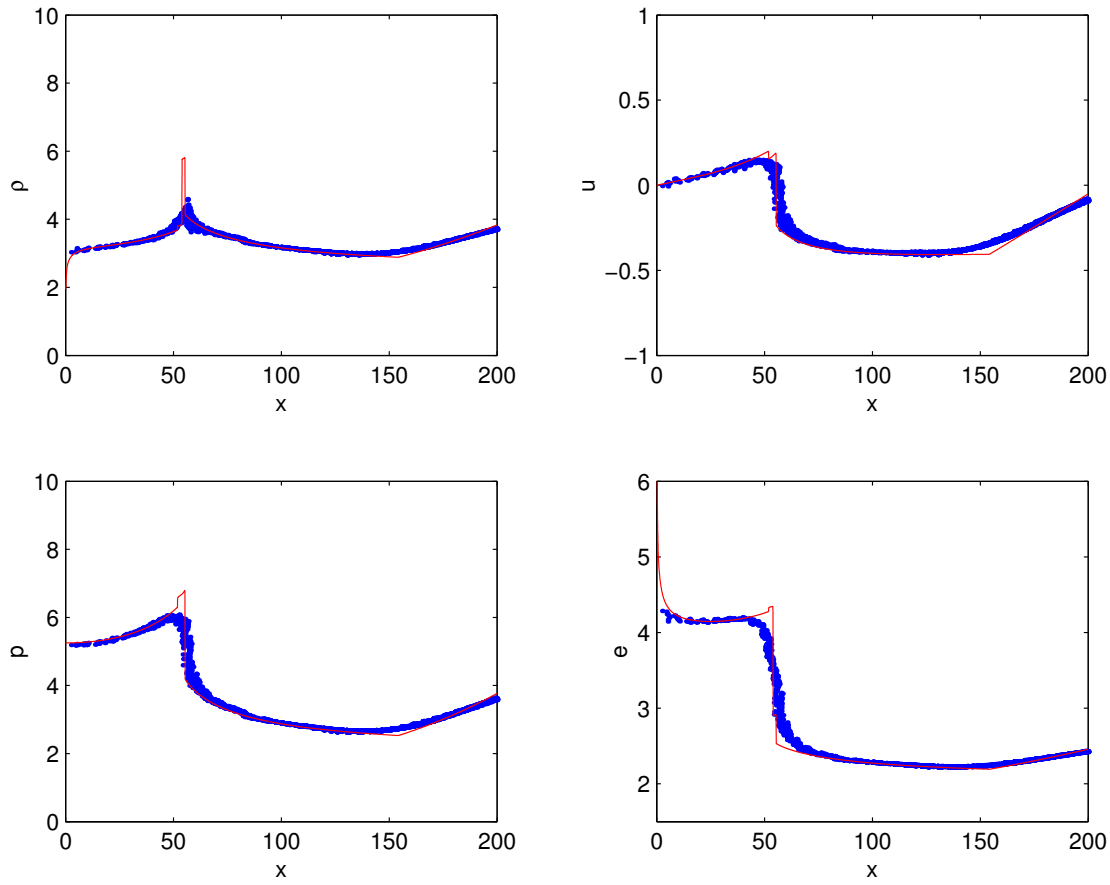

Figure 6.23: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$*, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)*

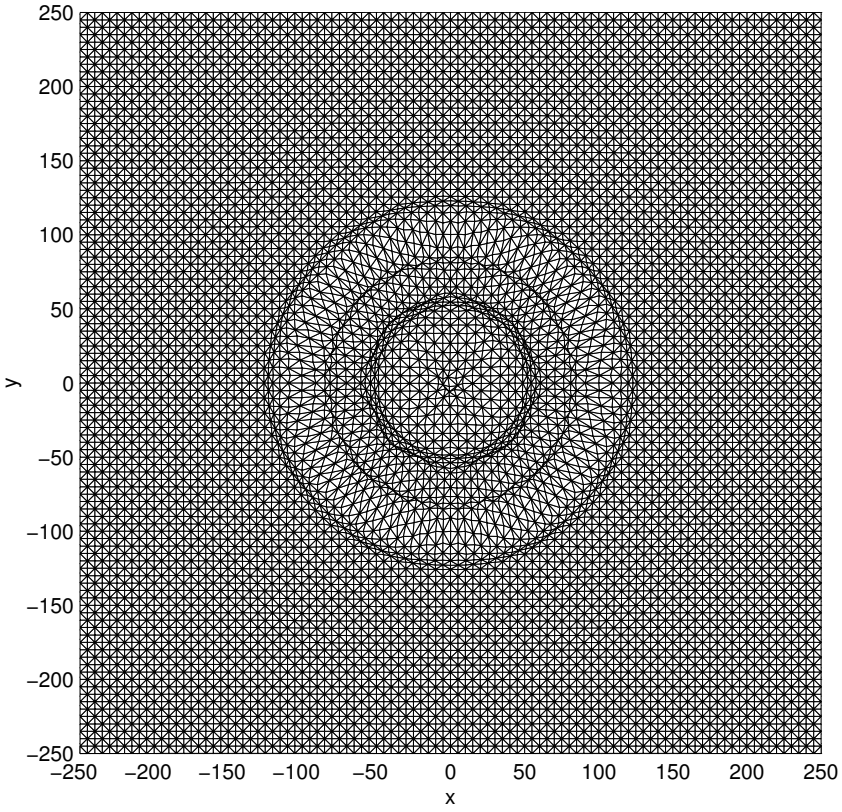

Figure 6.24: Plot of the final mesh at $t = 90$ *obtained for the solution of the Euler equations with the moving mesh method with* $M = 1 + \alpha \left|\nabla\rho\right|^2$.

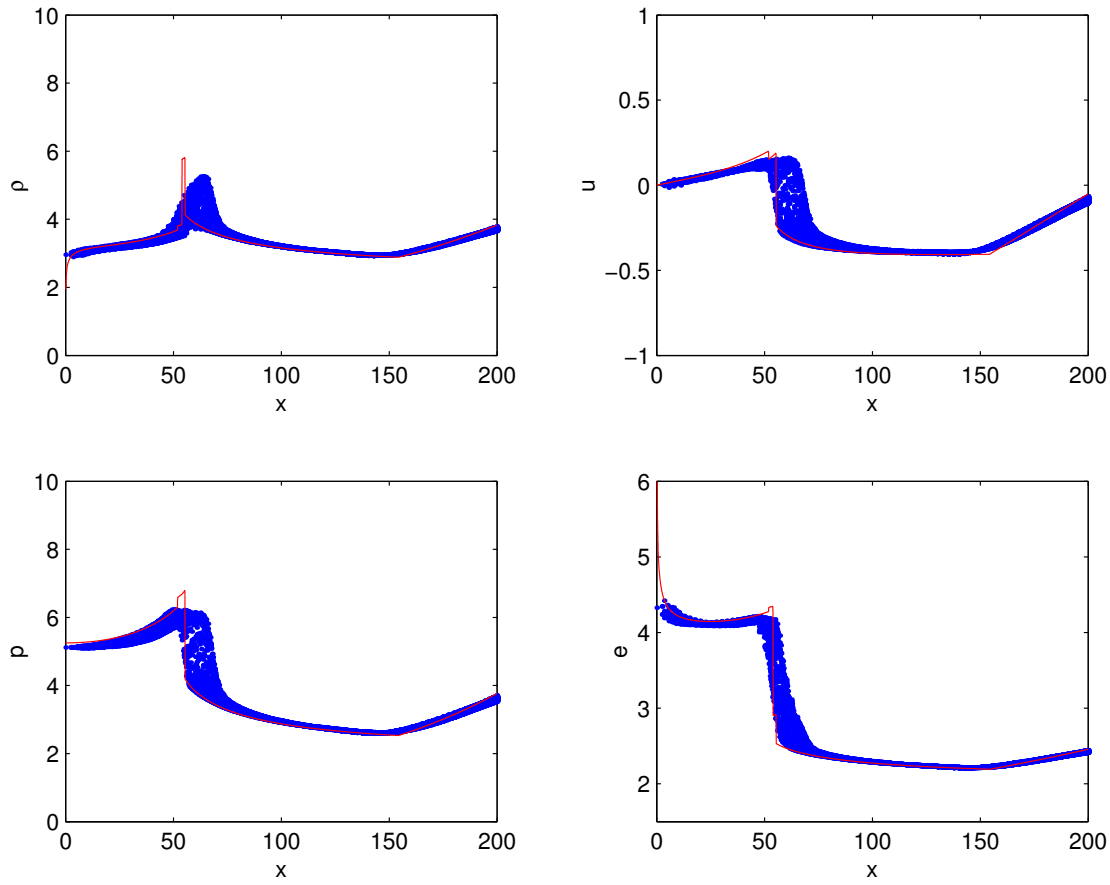

Figure 6.25: Solution of the Euler equations with initial data (6.26). Figure shows plots of density, radial velocity $v_{rad} = \sqrt{u^2 + v^2}$, pressure and specific internal energy plotted against the radial direction. (Note that the numerical solution obtained at all points has been plotted against the radial distance.)

6.8 Summary

In this chapter the moving mesh method derived in chapter 3 has been used to generate moving meshes upon which the two dimensional compressible Euler equations of gas dynamics have been solved. In two dimensions a finite volume method was obtained in section 6.2 on an unstructured triangular mesh for the solution of the arbitrary Lagrangian Eulerian (ALE) form of hyperbolic conservation laws. This ALE finite volume method was then used in conjunction with an ALE HLLC approximate Riemann solver which was outlined in section 6.3. Due to the fact that this numerical method led to a scheme that was only first order in both space and time we then used a MUSCL technique with the LCD limiter, described in section 6.4, to obtain higher order in space results.

The use of multidimensional generalisations of the monitor functions used in one spatial dimension for the compressible Euler equations was then outlined in sections 6.5 and 6.6. In two dimensions we used the mass monitor function $M = \rho$ and the geometric monitor $M = 1 + \alpha \ |\nabla\rho|^2$ to generate an adaptively moving mesh.

In section 6.7 we applied the moving mesh method using both the mass and geometric monitor functions. We produced results for a two dimensional generalisation of the one dimensional Sod shock tube problem, which has similar conditions to the Sod shock tube problem, but against the radial distance. The moving mesh methods were then compared which each other and the standard fixed grid Eulerian method. The computed solutions were all compared with the one dimensional reference solution.

It was found that the mass monitor led to an approximately Lagrangian method and tended to cluster mesh points in between the shock and contact wave where the gas was being compressed, as in the one-dimensional results. The moving mesh method together with the geometric monitor function produced a mesh in which points were clustered in regions of the solution where its gradient was high, such as in shock, contact and rarefaction waves.

Chapter 7

Discussion

7.1 Summary

In this chapter we summarise the findings of the work carried out in this thesis and suggest some further research that could be carried out. The extent of the original work carried out by the author is also detailed. The thesis begins in chapter 1 by outlining different types of adaptive methods for the solution of differential equations. Also in this chapter we described the main three types of numerical adaption, techniques which were referred to as h-, p- and r-adaption. It was also stated that in the thesis we would only consider the r-adaption technique, which consists of dynamically moving the positions of the computational mesh nodes in time in an attempt to increase the accuracy of the numerical method. The outline and aims of the thesis were also stated.

In chapter 2 some commonly used moving mesh methods and grid generation techniques were described. The notion of an adaptive mesh being produced as a one-to-one mapping from a background computational domain onto the physical domain in which the problem is being solved was introduced and it was shown how the various properties of the mesh in the physical domain can be controlled by this mapping. We then went on to introduce the notion of the of a monitor function in the context of the equidistribution principle, due to de Boor [16]. It was pointed out that in one spatial dimension many moving mesh methods are based on this equidistribution principle and from this principle the concept of moving mesh partial differential equations (MMPDEs) can be derived. We also gave examples of several monitor functions that have been used for the solution of both ordinary and partial differential equations.

In chapter 2 we also noted that in general moving mesh methods fall into one of two categories, velocity and location based methods, which differ in the fact that velocity based methods target the rate of change of the mapping from the computational to physical domain $\dot{x}(x(\xi,t),t)$ and location based methods target the mapping $x(\xi,t)$.

We then went on to consider multidimensional extensions of the ideas introduced in one spatial dimension. It was found that the equidistribution principle does not easily extend to higher dimensions, since the natural extension is an underdetermined system. Therefore in higher dimensions we described alternative routes which researchers have followed in generating an adaptive mesh. Many of the methods outlined in this chapter can be formulated as minimisation problems. In most cases, functionals were derived which could be used to control properties of the mesh. Also the idea of a monitor matrix to control the properties of the mesh was described.

Subsequently in chapter 2 we described the classical Lagrangian method in fluids in which the motion of the mesh is chosen to move with the local fluid velocity, and the generalisation of this known as the Arbitrary Lagrangian Eulerian (ALE) method.

Finally in chapter 2 we considered geometric integration techniques in which discrete approximations to systems are constructed in such a manner so that the discrete system inherits as many properties of the continuous problem as possible.

In chapter 3 we derived a moving mesh method using monitor functions. This moving mesh method was based on a conservation of monitor function principle in time. It was then shown that a mesh velocity is induced by prescribing that the mesh obeys this conservation principle. The resulting Eulerian form of the conservation law obtained for the velocity of the mesh is an equation for the velocity, but is known to be an underdetermined system for spatial dimensions higher than one. Therefore we chose to prescribe the *curl* of the mesh velocity to obtain a unique velocity field. Using this condition we then obtained an elliptic PDE for a mesh velocity potential, from which the velocity of the mesh could be recovered. An interpretation of the method was also given in terms of modelling the mesh as a pseudo-fluid in which the monitor function acts as the pseudo-density function. The mesh velocity was then found to be the velocity of the pseudo-fluid.

We chose to generate the velocity of the mesh through a finite element method, therefore in this chapter we derived weak forms of the equations for the mesh velocity potential and the equation for the recovery of the mesh velocity. These weak forms could then be used in a finite element framework to obtain a unique mesh velocity. The velocity is used to advance the mesh forward in time using an ODE time-stepping method.

In chapter 4 we considered the solution of the porous medium equation (PME) using the moving mesh method derived in chapter 3. We began this chapter by examining the scale invariance properties of the PME and detailing how a radial self-similar solution can be constructed. The scale invariance properties of the PME were then utilised to develop a scale invariant moving mesh method. We first considered the moving mesh method with the monitor function $M = u$, which is designed to conserve "mass" in the discrete sense. A scale invariant time-stepping routine was used to advance the mesh forward in time. In the case of the PME with $M = u$ we found that once the new adaptive mesh had been found, the solution to the problem could be recovered directly through the use of the conservation of monitor function principle. Results for this choice of monitor function were shown and it was found that the method produced accurate numerical solutions in a suitable norm. We also found that the solutions seemed to be correctly attracted to the self-similar solution as time increased. Also it was found that the scaled error in the solution was approximately invariant in time.

We then went on to consider the solution of the PME using the same moving mesh method but with other choices for the monitor function. It was described how to modify general monitor functions to produce scale invariant monitor functions, with particular reference to the gradient monitor function $M = |u_x|$. This monitor function was chosen as we expected that it would cluster many of the computational nodes within the steep moving front of the solution. Results were shown for the moving mesh method together with the gradient monitor function. It was found that recovering the solution directly with this choice of monitor function led to a less accurate method overall in the norm chosen in comparison with the mass monitor which was probably due to the fact that there was no constraint for the numerical solution to conserve mass in the discrete sense. Since this problem is characterised by the total mass in the solution this turned out to have a detrimental effect. It was however noted that the mass conserving property of

the solution could be maintained by using the Arbitrary Lagrangian Eulerian (ALE) form of the PME to solve the PME on the mesh generated from the gradient monitor function.

At the end of chapter 4 we extended the moving mesh method together with the mass monitor to the solution of the PME in two spatial dimensions. Results were shown for solutions to the PME with a variety of the powers m. The solution to a problem with non-self-similar initial conditions was also presented and it could be seen how the solution was attracted towards the self-similar solution as time increased.

Chapters 5 and 6 described the development of the moving mesh method to solve non-linear hyperbolic conservation laws, both in one and two spatial dimensions. In chapter 5 we considered the adaptive solution of the one-dimensional inviscid Burgers equation and the compressible Euler equations of gas dynamics. Since the solution to the equations being solved could not be recovered through the use of the conservation of monitor function principle the ALE approach was needed to recover the solution on the moving mesh. A finite volume scheme was used to solve the conservation law. It was found that to incorporate the velocity of the mesh in the method we had to solve the conservation law in a general frame of reference and hence we derived the ALE form of the conservation law.

In chapter 5 we also solved both the inviscid Burgers equation and the compressible Euler equations on an adaptively moving mesh generated from one of two monitor functions. The first monitor function we considered was the mass monitor and it was found that this choice led to the mesh motion being approximately Lagrangian. The second monitor function was designed by considering the structure of the solution to hyperbolic conservation laws. A geometric monitor function based on the gradient of the solution was utilised to move mesh points into regions of the domain where the gradient of the solution is high, such as in moving discontinuities. For both the inviscid Burgers equation and the Euler equations it was found for various test problems that the moving mesh method together with the geometric monitor function led to more accurate numerical solutions.

Chapter 6 considered the solution of the compressible Euler equations in two spatial

dimensions. In this chapter a generalisation of the one-dimensional ALE finite volume method was derived which was then used in conjunction with an ALE HLLC approximate Riemann solver. This method gives only first order accuracy in both space and time. Therefore to increase the accuracy of the numerical approximation we used the MUSCL method of Van Leer together with the LCD slope limiter. We then outlined the use of the mass monitor function and a generalisation of the one dimensional geometric monitor function which was used for the compressible Euler equations. These monitor functions were then used in the moving mesh method to generate a moving mesh in two spatial dimensions. We applied the method to a cylindrical shock problem, which can be seen as a generalisation of the one-dimensional Sod shock tube problem in the radial direction. Results were obtained for the two monitor functions and then compared to the standard Eulerian method.

The introductory chapters 1 and chapter 2 and the applications to hyperbolic problems in chapters 5 and 6 are entirely the work of the candidate. The method chapter 3 and the PME application chapter 4 parallel the work of Baines, Hubbard and Jimack [8, 9], but there are some original sections, namely sections 4.4 to 4.8.

7.2 Discussion of Similarity in Hyperbolic Systems

In this thesis we solved both the porous medium equation and the compressible Euler equation of gas dynamics using the moving mesh method. However, the solution procedure of these two equations was based on very different aims. The aim of the moving mesh method for the solution of the PME was to produce a numerical method which was scale invariant and hence inherited many of the continuous properties of the continuous equation. Using this property of the continuous equation we were able to construct solutions to the problem which were attracted to the self-similar solution as time advanced. In contrast, the solution technique for the Euler equations consisted of designing monitor functions which characterised the geometrical properties of the solution and in this manner we hoped that certain features of the flow would be well resolved by the mesh.

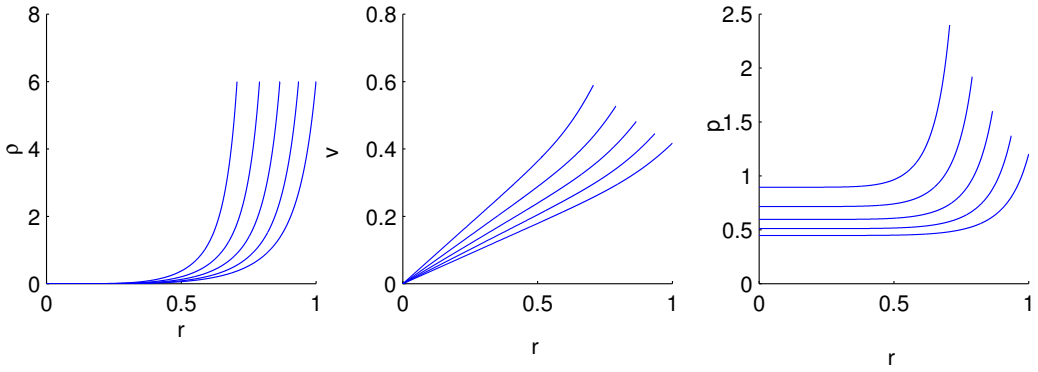

Figure 7.1: Self-similar solution to the compressible Euler equations in radial co-ordinates. Figure shows plots of density, velocity and pressure at four different times.

An interesting extension of this work could be to solve the Euler equations using similar scale invariant techniques that were used for the PME. For example one problem for the Euler equations which does exhibit scale invariance is that of the point source solution derived by Taylor [89, 90] and Von Neumann [74], which is also often referred to as the Sedov blast wave problem [11]. This problem consists of a very intense blast wave which propagates outwards from an initial high energy state. The early stages of this problem are described by the PME with the specific choice for the power m taken to be roughly 5 [11]. However later stages of the evolution are described by the compressible Euler equations in radial co-ordinates given as

$$\begin{pmatrix} \rho \\ \rho\, v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho\, v \\ \rho\, v^2 + p \\ v\ (E+p) \end{pmatrix}_r = -\frac{(d-1)}{r} \begin{pmatrix} \rho\, v \\ \rho\, v^2 \\ v\ (E+p) \end{pmatrix},$$

where d is the number of spatial dimensions. It can be shown [11] that the variables of the radial Euler equations, together with the condition that the total energy is conserved in time,

$$\int_0^\infty E\, r^{d-1}\, \mathrm{d}r = \int_0^\infty \left(\frac{p}{\gamma - 1} + \frac{1}{2}\, \rho\, v^2 \right) r^{d-1}\, \mathrm{d}r = \text{constant in time},$$

are invariant under the scalings

$$\hat{r} = \lambda^{\frac{2}{d+2}}\, r, \qquad \hat{\rho} = \rho, \qquad \hat{v} = \lambda^{-\frac{d}{d+2}}\, v, \qquad \hat{p} = \lambda^{-\frac{2\,d}{d+2}}\, p, \qquad \hat{t} = \lambda\, t,$$

where the variables ρ, v and p are the density, velocity and pressure of the gas and r and t are the spatial and temporal variables respectively. It can also be shown that under these scalings the Euler equations have a unique self-similar solution. The structure of the self-similar solution consists of a shock moving outward from the origin of the domain with speed s and a continuous transition of the solution behind the shock. Ahead of the shock the gas is at rest and has a constant density ρ_0. This region also has zero pressure. Applying the Rankine-Hugoniot jump conditions across this blast wave we find that

$$u_f = \frac{2}{\gamma+1}\, s, \qquad \rho_f = \frac{\gamma+1}{\gamma-1}\,\rho_0, \qquad p_f = \frac{2}{\gamma+1}\,\rho_0\, s^2$$

where the subscript f denotes the value of the variable immediately to the left of the shock. From scaling arguments we know that

$$\hat{t}^{-\frac{2}{d+2}}\,\hat{r} = t^{-\frac{2}{d+2}}\, r = a,$$

where a is a constant. Therefore the trajectory of the blast wave must have the form

$$r_f = a\, t^{\frac{2}{d+2}}$$

and hence the speed of the blast wave is given by

$$s = \frac{2}{d+2}\, a\, t^{\frac{2}{d+2}-1}.$$

Substituting this expression for the shock speed into the Rankine-Hugoniot jump conditions we obtain the relationships

$$u_f = \frac{2}{\gamma+1}\,\frac{2}{d+2}\, a\, t^{\frac{2}{d+2}-1}, \qquad \rho_f = \frac{\gamma+1}{\gamma-1}\,\rho_0, \qquad p_f = \frac{2}{\gamma+1}\,\rho_0\,\left(\frac{2}{d+2}\, a\, t^{\frac{2}{d+2}-1}\right)^2. \tag{7.1}$$

Also from this scale invariance argument it is easy to see that the quantity

$$\xi = \frac{r}{a\, t^{\frac{2}{d+2}}}$$

is invariant in time, and hence we may seek a self-similar solution to the problem in terms of a function of this invariant quantity. Thus we may assume that the solution to the radial Euler equations have the form

$$\rho = \rho_0 \, R\left(\xi\right) \qquad v = \frac{2}{d+2} \, \frac{r}{t} \, V\left(\xi\right) \qquad p = \frac{4}{\left(d+2\right)^2} \, \rho_0 \, \frac{r^2}{t^2} P\left(\xi\right) \tag{7.2}$$

which upon substituting these ansatzs into the radial Euler equations we obtain a set of three ODEs,

$$\left(V-1\right) \frac{\partial \left(\ln R\right)}{\partial \left(\ln \xi\right)} + \frac{\partial V}{\partial \left(\ln \xi\right)} + d \, V = 0$$

$$\left(V-1\right) \, R \frac{\partial V}{\partial \left(\ln \xi\right)} + \frac{\partial P}{\partial \left(\ln \xi\right)} + 2P + RV \left(V - \frac{d+2}{2}\right) = 0 \tag{7.3}$$

$$\frac{\partial}{\partial \left(\ln \xi\right)} \left(\ln \left(\frac{P}{R^\gamma}\right)\right) + \frac{\left(V - \left(d+2\right)\right)}{\left(V-1\right)} = 0$$

for the solution of V, P and R. Boundary conditions for this ODE system at $\xi = 1$ are obtained by equating the relations (7.1) and (7.2) and are given as

$$R\left(1\right) = \frac{\gamma+1}{\gamma-1}, \qquad V\left(1\right) = \frac{2}{\gamma+1}, \qquad P\left(1\right) = \frac{2}{\gamma+1}, \tag{7.4}$$

It is possible to solve the ODE system given by (7.3) together with the boundary conditions (7.4) for the functions R, V, P from which the self-similar solution can be obtained and is given by

$$\rho = \frac{\gamma+1}{\gamma-1} \rho_0 \theta^{\frac{d}{2(\gamma-1)+d}} \left(\frac{\theta+\gamma}{\gamma+1}\right)^{-\frac{2(d-1)}{d(\gamma-1)+2}} \left(\frac{d(2-\gamma)\theta + 2(\gamma-1)+d}{(2-d)\gamma + (3d-2)}\right)^{\frac{\gamma^2(d^2+4)-(d-2)(3d-2)\gamma+4d(d-2)}{(2-\gamma)(2(\gamma-1)+d)(d(\gamma-1)+2)}},$$

$$u = \frac{4}{(d+2)(\gamma+1)} a t^{-\frac{d}{d+2}} \theta^{\frac{\gamma-1}{2(\gamma-1)+d}} \left(\frac{1+\theta}{2}\right)^{\frac{d}{d+2}} \left(\frac{\theta+\gamma}{\gamma+1}\right)^{-\frac{(d-1)(\gamma-1)}{d(\gamma-1)+2}} \times \left(\frac{d(2-\gamma)\theta + 2(\gamma-1)+d}{(2-d)\gamma + (3d-2)}\right)^{-\frac{\gamma^2(d^2+4)-(d-2)(3d-2)\gamma+4d(d-2)}{(d+2)(2(\gamma-1)+d)(d(\gamma-1)+2)}},$$

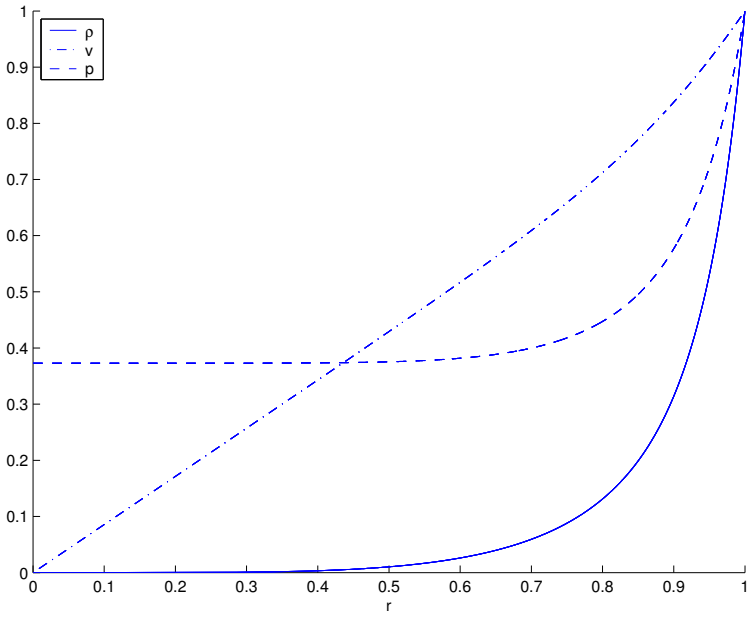

Figure 7.2: Scaled variables for self-similar solution to the compressible Euler equations in radial co-ordinates. Figure shows plots of scaled density, velocity and pressure.

$$p = \frac{8}{(d+2)^2(\gamma+1)}\rho_0 a^2 t^{-\frac{2d}{d+2}} \left(\frac{1+\theta}{2}\right)^{\frac{2d}{d+2}} \left(\frac{\theta+\gamma}{\gamma+1}\right)^{-\frac{2(d-1)\gamma}{d(\gamma-1)+2}} \times \left(\frac{d(2-\gamma)\theta + 2(\gamma-1)+d}{(2-d)\gamma+(3d-2)}\right)^{\frac{\gamma^2(d^2+4)-(d-2)(3d-2)\gamma+4d(d-2)}{(d+2)(2-\gamma)(d(\gamma-1)+2)}},$$

where θ is a new variable which ranges between zero and one. It should be noted that this solution is only valid for $1 < \gamma < 2$. This solution can be seen in figure 7.1 and the scale invariant quantities can be seen in figure 7.2.

We could therefore use ideas from chapter 4 to produce scale invariant monitor functions to produce an adaptive mesh for this problem. For example, it is straightforward to show that the monitor functions

$$\int_0^\infty t^{\frac{2(d-1)}{d+2}} \rho_r \, r^{d-1} \, \mathrm{d}r \qquad \text{and} \qquad \int_0^\infty E \, r^{d-1} \, \mathrm{d}r,$$

where E is the total energy, are scale invariant for this problem.

7.3 Further Research

We conclude this thesis by considering some other possible areas of further research. There are a variety of avenues for further research using the moving mesh method that has been used in this thesis. One could solve many other partial differential equations or apply the method to the PDEs already solved in other contexts, as well as devising more complicated test problems such as the Mach reflection test problem for the compressible Euler equations.

Alternatively, we could pursue the design of different monitor functions which could result in a more accurate and robust method [13]. Error estimates may be used as monitors if available, or monitor functions may be designed to bring out particular features of the solution. For example, it may be advantageous for the solution of the compressible Euler equations to have a monitor function based on the curvature of the density, or some other variable, to move mesh points into shock and contact wave regions. This type of monitor function may also be combined with the already utilised gradient monitor function. Therefore a more general form of a monitor function for the solution of the Euler equations could be

$$M = 1 + \alpha\,\rho_x^2 + \beta\,\rho_y^2 + \gamma\,\rho_{x\,x}^2 + \delta\,\rho_{x\,y}^2 + \epsilon\,\rho_{y\,y}^2,$$

where α and β are constants to be chosen. The monitor function may also be based on other variables such as entropy in the case of the Euler equations, as in [86].

Also the velocity $\mathbf{q}$ and the variable ϖ in the moving mesh have not been exploited in this work. In all the results produced in this thesis these two quantities have been chosen to be $\mathbf{q} = 0$ and $\varpi = 1$. Since these values lead to the mesh generated being irrotational. However, in some circumstances it may be advantageous to have a mesh which has some rotational properties. For instance, if a mesh is generated which is highly skewed due to the presence of large amounts of vorticity in the solution then we may want to choose the curl of the mesh velocity in such a way as to minimise the amount of skewness in the mesh.

We may also investigate how the solution of the problem being solved depends on the initial mesh chosen. For the two-dimensional results obtained for both the porous medium

equation and the Euler equations we used a relatively uniform initial mesh, however we may be able to obtain more accurate results by using an initially adapted mesh. For example, we could use a least-squares fit with adjustable nodes to produce an initial mesh [6]. Alternatively it may be possible to construct a non-uniform mesh from an uniform mesh by using the moving mesh method described in this thesis. This could be done by considering the initial grid generation as a steady state problem and advancing the mesh forward in pseudo-time to produce an adapted mesh.

Lastly, the computational efficiency of the moving mesh method may be investigated to discover whether the method is more or less efficient when compared to other commonly used methods.

Bibliography

[1] S. Adjerid and J.E. Flaherty. A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations. *SIAM Journal on Numerical Analysis. Vol. 23, Issue 4, 778-796*, 1986.

[2] D.A. Anderson. Adaptive mesh schemes based on grid speeds. *AIAA Paper 83-1931 311-318*, 1983.

[3] D.A. Anderson. Application of adaptive grids in transient problems. *In Adaptive Computational Methods for PDE's, SIAM, Philadelphia. 208-223*, 1983.

[4] D.G. Aronson. The porous medium equation. *In "Nonlinear Diffusion Problems", Lecture Notes in Mathematics, No 1224. Springer-Verlag*, 1986.

[5] B.N. Azarenok. Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics. *SIAM Journal of Numerical Analysis. 40, 651-682*, 2002.

[6] M.J. Baines. Moving finite elements. *Oxford Science Publications*, 1994.

[7] M.J. Baines. Grid adaption via node movement. *Applied Numerical Mathematics. 26 77-96*, 1998.

[8] M.J. Baines, M.E. Hubbard, and P.K. Jimack. A Lagrangian moving finite element method incorporating monitor functions. *In proceedings of the 3rd workshop on numerical methods for partial differential equations, Hong Kong*, 2003.

[9] M.J. Baines, M.E. Hubbard, and P.K. Jimack. A moving mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries. *Applied Numerical Mathematics*, 2004. To appear.

[10] G.I. Barenblatt. On some unsteady motions of a liquid or a gas in a porous medium. *Prikl. Mat. Mekh. 16, 67-78*, 1952.

[11] G.I. Barenblatt. Self-similarity and intermediate asymptotics. *Cambridge University Press*, 1996.

[12] G. Beckett, J.A. Mackenzie, and M.L. Robertson. A moving mesh finite element method for the solution of two-dimensional Stefan problems. *Journal of Computational Physics. 168, 500-518*, 2001.

[13] K.W. Blake. Moving mesh methods for non-linear parabolic partial differential equations. *Ph.D, Department of Mathematics, University of Reading*, 2001.

[14] K.W. Blake and M.J. Baines. A moving mesh method for non-linear parabolic equations. *Numerical Analysis Report 2/2002, Department of Mathematics, University of Reading*, 2002.

[15] D. Boffi and L. Gastaldi. Stability and geometric conservation laws for ALE formulations. *Computer Methods in Applied Mechanics and Engineering, Vol. 193, Issues 42-44, 4717-4739*, 2004.

[16] C. De Boor. Good approximation by splines with variable knots II. *Springer Lecture Notes Series 363*, 1973.

[17] J.U. Brackbill. An adaptive grid with directional control. *Journal of Computational Physics. 108, 38-50*, 1993.

[18] J.U. Brackbill and J.S. Saltzman. Adaptive zoning for singular problems in two dimensions. *Journal of Computational Physics. 46, 342-368*, 1982.

[19] J. Buckmaster. Viscous sheets advancing over dry beds. *Journal of Fluid Mechanics. 81, 735-756*, 1977.

[20] C.J. Budd, S. Chen, and R.D. Russell. New self-similar solutions of the nonlinear Schrödinger equation with moving mesh computations. *Journal of Computational Physics. 152, 756-789*, 1999.

[21] C.J. Budd and G. Collins. An invariant moving mesh scheme for the nonlinear diffusion equation. *Technical Report 19/08/96, School of Mathematics, University of Bath*, 1996.

[22] C.J. Budd, G.J. Collins, W.Z. Huang, and R.D. Russell. Self-similar discrete solutions of the porous medium equation. *Philos. Trans. Roy. Soc. London A. 357, 1047-1078*, 1999.

[23] C.J. Budd, W. Huang, and R.D. Russell. Moving mesh methods for problems with blow-up. *SIAM Journal on Scientific Computing. Vol. 17, 305*, 1996.

[24] C.J. Budd and M. Piggott. The geometric integration of scale-invariant ordinary and partial differential equations. *Journal of Computational and Applied Mathematics. 128, 399-422*, 2001.

[25] C.J. Budd and M. Piggott. Geometric integration and its applications. *Handbook of Numerical Analysis. Vol. 9, 35-139*, 2003.

[26] W. Cao, W. Huang, and R.D. Russell. A study of monitor functions for two dimensional adaptive mesh generation. *SIAM Journal on Scientific Computing. Vol. 20, 1978-1994*, 1999.

[27] W. Cao, W. Huang, and R.D. Russell. A moving mesh method based on the geometric conservation law. *SIAM Journal on Scientific Computing. Vol. 24, No. 1, 118-142*, 2002.

[28] W. Cao, W. Huang, and R.D. Russell. Approaches for generating moving adaptive meshes: location versus velocity. *Applied Numerical Mathematics 47 121-138*, 2003.

[29] N.N. Carlson and K. Miller. Design and application of a gradient-weighted moving finite element code, part I, in 1-D. *SIAM Journal on Scientific Computing. Vol. 19, 728-765.*, 1998.

[30] N.N. Carlson and K. Miller. Design and application of a gradient-weighted moving finite element code, part II, in 2-D. *SIAM Journal on Scientific Computing. Vol. 19, 766-798*, 1998.

[31] H.D. Ceniceros and T.Y. Hou. An effficient dynamically adaptive mesh for potentially singular solutions. *Journal of Computational Physics. 172, 609-639*, 2001.

[32] G.R. Cowper. Gaussian quadrature formulas for triangles. *International Journal for Numerical Methods in Engineering. 7, 405-408*, 1973.

[33] J.M. Coyle, J.E. Flaherty, and R. Ludwig. On the stability of mesh equidistribution strategies for time-dependent partial differential equations. *Journal of Computational Physics. 62 25-39*, 1986.

[34] B. Dacorogna and J. Moser. On a PDE involving the Jacobian determinant. *Ann. Inst. H. Poincare, 7*, 1990.

[35] I. Demirdzic and M. Peric. Space conservation law in finite volume calculations of fluid flow. *International Journal for Numerical Methods in Fluids, Vol. 8, 1037-1050*, 1988.

[36] Vit Dolejsi. Angener. *Users Guide*, 2001.

[37] E.A. Dorfi and L.O'C. Drury. Simple adaptive grids for 1-D initial value problems. *Journal of Computational Physics. 69 175-195*, 1987.

[38] A.S. Dvinsky. Adaptive grid generation from harmonic maps on Riemannian manifolds. *Journal of Computational Physics. 95 450-476*, 1991.

[39] C. Farhat, Philippe Geuzaine, and Celine Grandmont. The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids. *Journal of Computational Physics. 174 669-694*, 2001.

[40] R.M. Furzeland, J.G. Verwer, and P.A. Zegeling. A numerical study of three moving-grid methods for one-dimensional partial differential equations which are based on the method of lines. *Journal of Computational Physics. 89 349-388*, 1990.

[41] P. Glaister. Flux difference splitting for the Euler equations in one spatial coordinate with area variation. *International Journal for Numerical Methods in Fluids, Vol.8, 97-119*, 1988.

[42] S.K. Godunov. Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat. Sb. 47, 271*, 1959.

[43] A. Harten and J.M. Hyman. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *Journal of Computational Physics 50, 235-269*, 1983.

[44] A. Harten, P.D. Lax, and B. Van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review. Vol. 25, No, 1, 35-61*, 1983.

[45] C.W. Hirt, A.A. Amsden, and J. L. Cook. An Arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of Computational Physics 14, 227*, 1974.

[46] W. Huang. Practical aspects of formulation and solution of moving mesh partial differential equation. *Journal of Computational Physics. 171, 753-775*, 2001.

[47] W. Huang. Variational mesh adaption: isotropy and equidistribution. *Journal of Computational Physics. 174, 903-924*, 2001.

[48] W. Huang, Y. Ren, and R. Russell. Moving mesh methods based on moving mesh partial differential equations. *Journal of Computational Physics. Vol 113, pages 279-290*, 1994.

[49] W. Huang, Y. Ren, and R.D. Russell. Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle. *SIAM Journal of Numerical Analysis. Vol.31, No.3, 709-730*, 1994.

[50] W. Huang and R.D. Russell. Analysis of moving mesh partial differential equations with spacial smoothing. *SIAM Journal on Numerical Analysis. Vol. 34, Issue 3, 1106-1126*, 1997.

[51] W. Huang and R.D. Russell. A high dimensional moving mesh strategy. *Applied Numerical Mathematics. 26, 63-76*, 1997.

[52] W. Huang and R.D. Russell. Moving mesh strategy based on a gradient flow equation for two-dimensional problems. *SIAM Journal on Scientific Computing . Vol. 20, No. 3, 998-1015*, 1999.

[53] W. Huang and R.D. Russell. Adaptive mesh movement - the MMPDE approach and its applications. *Journal of Computational and Applied Mathematics. 128, 383-398*, 2001.

[54] M.E. Hubbard. Multidimensional slope limiters for MUSCL-type finite volume schemes on unstructured grids. *Journal of Computational Physics. Vol 155, pages 54-74*, 1999.

[55] A.S. Kalashnikov. Some problems of the qualitative theory of non-linear degenerate second-order parabolic equations. *Russian Mathematical Surveys. 42, 2, 169-222*, 1987.

[56] E.W. Larsen and G.C. Pomraning. Asymptotic analysis of nonlinear Marshak waves. *SIAM Journal on Applied Mathematics. 39, 201-212*, 1980.

[57] P.D. Lax. Hyperbolic systems of conservation laws and the mathematical theory of shock waves. *SIAM Regional Conference Series in Applied Mathematics. No. 11*, 1972.

[58] B. Van Leer. Towards the ultimate conservative difference scheme v. A second order sequel to Godunov's method. *Journal of Computational Physics. 32 101-136*, 1979.

[59] R.J. LeVeque. CLAWPACK software. **http://www.amath.washington.edu/~rjl/clawpack.html.**

[60] R.J. LeVeque. Numerical methods for conservation laws. *Lectures in Mathematics, ETH Zurich. Birkhauser Verlag*, 1992.

[61] R.J. LeVeque. Finite volume methods for hyperbolic problems. *Cambridge Texts in Applied Mathematics. Cambridge University Press.*, 2002.

[62] R. Li, T. Tang, and P. Zhang. Moving mesh methods in multiple dimensions based on harmonic maps. *Journal of Computational Physics. 170, 562-588*, 2001.

[63] S. Li, L. Petzold, and Y. Ren. Stability of moving mesh systems of partial differential equations. *SIAM Journal of Scientific Computing. Vol. 20, No. 2, 719-738*, 1999.

[64] G. Liao and D.A. Anderson. A new approach to grid generation. *Applicable Analysis, No. 44, 285*, 1992.

[65] G. Liao, F. Liu, G.C. De La Pena, D. Peng, and S. Osher. Level-set-based deformation methods for adaptive grids. *Journal of Computational Physics. 159, 103-122*, 2000.

[66] F. Liu, S. Ji, and G. Liao. An adaptive grid method and its applications to steady Euler flow calculations. *SIAM Journal on Scientific Computing. Vol. 20 No. 3 , 811-825*, 1998.

[67] F. Liu, S. Ji, and G. Liao. An adaptive grid method with cell volume control and its applications to Euler flow calculations. *SIAM Journal on Scientific Computing. Vol. 20*, 1999.

[68] K. Miller. Moving finite element. II. *SIAM Journal on Numerical Analysis. Vol. 18, Issue 6, 1033-1057*, 1981.

[69] K. Miller and R.N. Miller. Moving finite element. I. *SIAM Journal on Numerical Analysis. Vol. 18, Issue 6, 1019-1032*, 1981.

[70] J. Moser. On the volume elements on a manifold. *Transactions of the American Mathematical Society. Vol. 120, No. 2, 286-294*, 1965.

[71] L.S. Mulholland, Y. Qiu, and D.M. Sloan. Solution of evolutionary partial differential equations using adaptive finite differences with pseudospectral post-processing. *Journal of Computational Physics 131, 280-298*, 1997.

[72] B.R. Munson, D.F. Young, and T.H. Okiishi. Fundamentals of fluid mechanics. *John Wiley and Sons.*, 1990.

[73] J.D. Murray. Mathematical biology. *Pages 238-241. Berlin New York: Springer*, 1989.

[74] J. Von Neumann. The point source solution. *Collected Works. Vol.VI, 219-237*, 1947.

[75] R.E. Pattle. Diffusion from an instantaneous point source with concentrated dependent coefficient. *Quarterly Journal on Mechanical Applied Mathematics. 12, 407-409*, 1959.

[76] L.R. Petzold. A description of DASSL: A differential / algebraic system solver. *SAND 82-8637, Sandia Labs, Livermore, CA*, 1982.

[77] Y. Qiu and D.M. Sloan. Numerical solution of Fisher's equation using a moving mesh method. *Strathclyde University Mathematics Research Report. No. 97/22*, 1997.

[78] Y. Qiu and D.M. Sloan. On multiple solutions of a convection-diffusion boundary value problem produced by an adaptive mesh method. *Strathclyde University Mathematics Research Report. No. 97/42*, 1997.

[79] Y. Ren and R.D. Russell. Moving mesh techniques based upon equidistribution, and their stability. *SIAM Journal of Scientific Statistical Computing. Vol. 32, Issue 3*, 1992.

[80] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics 135, 250-258*, 1981.

[81] P.L. Roe. Some contributions of the modelling of discontinuous flows. *Lecture Notes in Applied Mathematics, 163-193*, 1985.

[82] B. Semper and G. Liao. A moving grid finite-element method using grid deformation. *Numerical Methods for Partial Differential Equations. No. 11, 603-615*, 1995.

[83] J.A. Sethian. Level set methods - evolving interfaces in geometry, fluid mechanics, computer vision and material science. *Cambridge University Press*, 1996.

[84] J. Smoller. Shock waves and reaction-diffusion equations. *Springer-Verlag*, 1983.

[85] G.A. Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics 27, 1-31*, 1978.

[86] J. Stockie, J. Mackenzie, and R. Russell. A moving mesh method for one-dimensional hyperbolic conservation laws. *SIAM Journal of Scientific Computing. Vol 22, No.5, pages 1791-1813*, 2001.

[87] P.K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis Vol. 21, No. 5, 995-1011*, 1984.

[88] H. Tang and T. Tang. Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws. *SIAM Journal of Numerical Analysis. Vol. 41, No. 2, 487-515.*, 2003.

[89] G. Taylor. The formation of a blast wave by a very intense explosion. I. theoretical discussion. *Proceedings of The Royal Society of London. Series A, Mathematical and Physical Sciences. Vol. 201, No. 1065, 159-174*, 1950.

[90] G. Taylor. The formation of a blast wave by a very intense explosion. II. the atomic explosion of 1945. *Proceedings of The Royal Society of London. Series A, Mathematical and Physical Sciences. Vol. 201, No. 1065, 175-186*, 1950.

[91] P.D. Thomas and C.K. Lombard. Geometric conservation law and its application to flow computations on moving grids. *AIAA Journal, Vol. 17, No. 10 1031-1037*, 1979.

[92] J.F. Thompson, F.C. Thames, and C.W. Mastin. Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies. *Journal of Computational Physics. 15, 299-319*, 1974.

[93] E.F. Toro. Riemann solvers and numerical methods for fluid dynamics. *Springer-Verlag*, 1997.

[94] E.F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves 4, 25-34*, 1994.

[95] J.L. Vasquez. An introduction to the mathematical theory of the porous medium equation. *Shape Optimisation and Free Boundaries. Kluwer Academic, pages 347-389*, 1992.

[96] J.G. Verwer, J.G. Blom, R.M. Furzeland, and P.A. Zegeling. A moving grid method for one-dimensional PDEs based on the method of lines. *In Adaptive Methods for Partial Differential Equations, 160-175, SIAM*, 1989.

[97] A.J. Wathen. Realistic eigenvalue bounds for the Galerkin mass matrix. *IMA Journal on Numerical Analysis, 7 449-457*, 1987.

[98] B.V. Wells, M.J. Baines, and P. Glaister. Generation of Arbitrary Lagrangian-Eulerian (ALE) velocities based on monitor functions, for the solution of the compressible fluid equations. *Proceedings of the ICFD Conference 2004, OUCL Oxford. Submitted to International Journal on Numerical Methods in Fluids*, 2004.

[99] A.B. White. On selection of equidistributing meshes for two-point boundary-value problems. *SIAM Journal on Numerical Analysis. Vol.16, No.3, 472-502*, 1979.

[100] G.B. Whitham. Linear and nonlinear waves. *New York: Wiley-Interscience*, 1974.

[101] A.M. Winslow. Numerical solution of the quasilinear Poisson equation in a nonuniform triangle mesh. *Journal of Computational Physics 2, 149-172*, 1978.

[102] A.M. Winslow. Adaptive mesh zoning by the equipotential method. *Lawrence Livermore Laboratory Report UCID-19062*, 1981.

[103] Y.B. Zel'dovich and A.S. Kompaneets. Theory of heat transfer with temperature dependent thermal conductivity. *Akad. Nauk SSSR, Moscow. 67-71*, 1950.

[104] H. Zhang, M. Reggio, J.Y. Trepanier, and R. Camarero. Discrete form of the GCL for moving meshes and its implementation in CFD schemes. *Computers Fluids. Vol. 22, No. 1, 9-23*, 1993.