The University of Reading

School of Mathematics, Meteorology & Physics

Moving Mesh Methods for Semi-Linear Problems

by

Matthew Paul Edgington

August 2011

This dissertation is a joint MSc in the Departments of Mathematics & Meteorology and is submitted in partial fulfilment of the requirements for the degree of Master of Science

Abstract

In this dissertation we examine the application of moving mesh methods to a number of semi-linear partial differential equations (PDEs). In particular we apply moving mesh methods that are based on the principle of conservation of certain quantities. The PDEs we consider are the Fisher's equation, Non-linear Schrödinger equation and the Cahn-Allen equation. We begin with some examples of PDEs that exhibit blow-up behaviour, that is, they have a solution that becomes infinite within a finite time, and then investigate some applications to other problems not displaying blow-up behaviour. The main aim of this dissertation is to examine the effects of using these conservative moving mesh methods on the capture of the solutions obtained for the specific PDEs considered and then to discuss the results obtained.

Acknowledgements

I would like to begin by acknowledging Professor Mike Baines for his help and support during the course of this dissertation. In particular I would like to thank him for always being available to help and for being an endless source of information and guidance. All of this on top of being a genuinely nice person.

I would also like to thank all of the academic staff in both the mathematics and meteorology departments who make this course what it is. A special mention must also go to Dr. Peter Sweby for his excellent organisation of the Mathematical and Numerical Modelling of the Atmosphere and Oceans MSc course.

A huge debt of gratitude is also owed to everyone who encouraged me to return to university and pursue my interests.

Finally, I must thank the Natural Environment Research Council (NERC) for their financial support, without which I would not have been able to further my education in this way.

Declaration

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

Signed ..... Date .....

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Why do we use moving mesh methods?

According to Budd et al. [1], many systems of time-dependent partial differential equations (PDEs) have structures which change significantly over time. Budd et al. [1] go on to explain that these changes may be due to interfaces, shocks, singularities, changes of phase, high vorticity or other regions of complexity. Within this dissertation we shall focus on problems with structures which change due to singularities and interfaces.

Mesh adaptivity takes three main forms which are:

- **h-refinement:** This method adds refinement to the mesh by adding in extra nodes in the area where blow-up occurs. The main weakness of this is the growth of the computational expense as we add in extra nodes.

- **p-refinement:** This method works by using higher-order polynomials to give a representation of the solution. By using this method a more accurate approximation of the solution is obtained in each cell than h-refinement, although this method will still not be able to model the blow-up behaviour if the blow-up point is between the nodes of the mesh.

- **r-refinement:** This is a moving mesh method that uses a fixed number

of nodes which are redistributed at each time step in order to track the blow-up behaviour as it develops. The main advantages of this type of method are that it allows computations to be carried out all the way up to the blow-up time and also it is not computationally expensive to do so. There is however a weakness in that as the nodes track the blow-up behaviour (i.e. they move towards the blow-up point) the nodes away from the blow-up point become more sparsely distributed meaning that the solution in these areas may be poorly represented.

In this project we shall focus our attention on r-refinement which, according to Westwood [2], is a Lagrangian-based approach which may be divided into mapping-based and velocity-based approaches. Westwood [2] states that the mapping-based method is the more commonly applied of these however in this case we investigate a velocity-based moving mesh method.

A velocity based method uses the information available in order to assign a velocity to each node with which it will move towards an area of interest, the location of which will depend on the PDE being considered. The velocity of each node will be assigned in such a way that a particular quantity will remain invariant in each interval as time passes.

According to Budd et al. [3], one of the main benefits associated with this form of r-refinement is that the mesh can track certain properties of the PDEs solution. This allows us to obtain a greater resolution in the solution close to certain points of interest. Budd et al. [3] also state that in many cases one of the most important features of this type of moving mesh method is that we may use fewer nodes in order to obtain a solution with a suitable resolution. This feature can result in a significant reduction in the computational cost associated with the solution of various PDEs.

1.2 What is a blow-up problem?

Blow-up problems occur in many different branches of science such as combustion in chemicals and chemotaxis in mathematical biology, according to Budd et al. [4]. The same paper [4] goes on to explain that in a blow-up problem a singularity forms and, as time goes by, changes begin to occur on increasingly small length scales. It is these changes on increasingly small length scales which lead us to examine the implementation of moving mesh methods, i.e. a moving mesh will allow the small length scale changes to be resolved whilst retaining a certain level of computational efficiency.

Many papers including Budd et al. [3] and [4], give an explanation of a 'typical' blow-up problem. In the 'typical' blow-up problem, the solution becomes infinite within a finite time, called the blow-up time and is denoted by T. This blow-up typically (but not always) occurs at a single location which is often denoted x^* and is called the blow-up point. That is, we have

$$u(x^*, t) \to \infty \quad \text{as} \quad t \to T,$$

and

$$u(x, t) \to u(x, T) \quad \text{if} \quad x \neq x^*,$$

where u represents the solution value and $T < \infty$. Budd et al. [3] [4] also explain that close to the point x^*, the solution $u(x, t)$ develops an isolated peak which becomes narrower, tending to zero width, as $t \to T$. An example of this type of solution profile is shown in Figure (1.1).

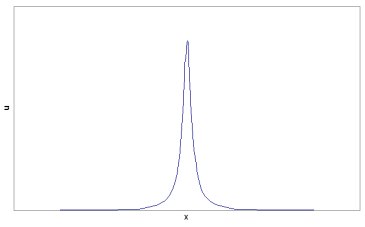


Figure 1.1: An example of a typical blow-up solution profile

1.3 What is a phase field problem?

According to Westwood [2], phase field models are used to represent situations whereby a sharp interface is represented by very thin transition layers so that the phase field varies continuously over these transition layers, and yet remains uniform over the bulk phases.

Zhang and Du [5] state that one of the most important challenges of modeling this type of problem is to suitably resolve the thin interfacial layer. In this small layer the solution will remain smooth, but develop a large spatial gradient. Zhang and Du [5] go on to explain that cases where such layers move over time may be used to model dynamically evolving fronts.

The literature on this type of problem clearly shows somewhat of a typical solution which has a very specific form. This is that in a case with two phases P_1 and P_2, there will be a smooth curve from each phase into a linear slope between them. An example of such a solution can be seen in Figure (1.2).

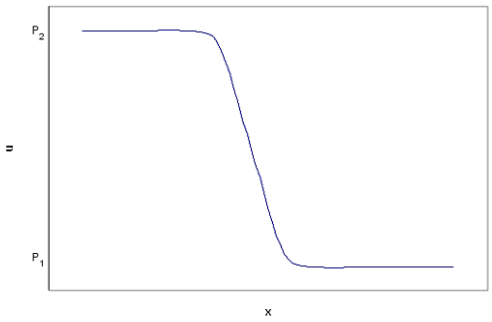


Figure 1.2: An example of a typical phase field problem solution

Chapter 2

Blow-Up in Fisher's Equation

Our investigation into the application of moving mesh methods in the solution of various PDEs begins with the examination of Fisher's equation and this is a common starting point when considering blow-up problems.

2.1 Introduction to Fisher's Equation

Fisher's equation is the standard one-dimensional heat equation with an extra source term and is given by

$$u_t = u_{xx} + u^p, \qquad (p > 1). \tag{2.1}$$

Fisher's equation has applications in many areas of science as described in [3], [6] and [7]. Budd et al. [3] state that the equation is used as a representation of the temperature in a reacting or combusting medium, whilst Braun and Kluwick [6] and Kluwick et al. [7] explain a use of Fisher's equation in the representation of various processes involved in laminar boundary layer separation, which is a key research area within meteorological science.

Fisher's equation is an example of a typical blow-up problem in so much as the solution can exhibit blow-up at a single blow-up point, denoted x^*, and this will occur within a finite blow-up time, T. Budd et al. [3] describe this by saying that for some blow-up time $T < \infty$, as $t \to T$ we have

$$u(x^*, t) \to \infty \text{ and } u(x, t) \to u(x, T) < \infty, \text{ if } x \neq x^*.$$

2.2 Problem Formation

In examining Fisher's equation we will consider two different cases which are related to the power of the blow-up term. We begin by considering the case in [3] where $p = 2$ and we will then go on to look at the case where $p = 3$, i.e. we will examine

$$u_t = u_{xx} + u^2 \tag{2.2}$$

and

$$u_t = u_{xx} + u^3 \tag{2.3}$$

in $x \in [0, 1]$. Throughout our investigation into Fisher's equation we will use Dirichlet boundary conditions of the form

$$u(0, t) = u(1, t) = 0.$$

According to Budd et al. [3] our choice of initial condition must be large enough to ensure that blow-up will occur and also must be chosen such that our problem will display blow-up at a single blow-up point (in this case $x^* = 0.5$). With these considerations in mind we take our initial condition to be that given in [8], i.e.

$$u(x, 0) = 20 \sin(\pi x).$$

2.3 Fisher's Equation on a Fixed Mesh

In order to demonstrate the need for moving mesh methods we first consider Fisher's equation as in Equation (2.2), as discussed previously and examine two different solution methods which both utilise a fixed mesh. The two fixed mesh methods we consider are very simple explicit and semi-implicit solution methods. We will then discuss the results obtained and demonstrate the need for a moving mesh method for solving blow-up problems.

2.3.1 Explicit Method

One of the simplest methods which can be used to solve Fisher's equation numerically is to utilise finite difference methods applied on a fixed mesh (i.e. a mesh made of static nodes). This will give us some insight into the blow-up solution of Fisher's equation and act as a benchmark against which to compare results.

Using a forward difference in time and a central difference in space we may discretise Fisher's equation to obtain

$$\frac{\left(u_j^{n+1} - u_j^n\right)}{\Delta t} = \frac{\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right)}{\Delta x^2} + \left(u_j^n\right)^2,$$

which may be rearranged to give

$$u_j^{n+1} = u_j^n + \mu\left[u_{j+1}^n - 2u_j^n + u_{j-1}^n\right] + \Delta t\left(u_j^n\right)^2,$$

where $\mu = \frac{\Delta t}{\Delta x^2}$ and must be chosen so as to satisfy conditions for numerical stability.

In the particular example shown in Figure (2.1) we have chosen to utilise a grid of 21 mesh points (i.e. $\Delta x = 0.05$ and with numerical stability in mind we have chosen to take $\Delta t = 0.0005$. This value of Δt has been chosen so as to ensure that $\frac{\Delta t}{\Delta x^2} < \frac{1}{2}$ which is required for stability when the u^2 term is absent. The results shown in Figure (2.1) show the first 165 time steps.

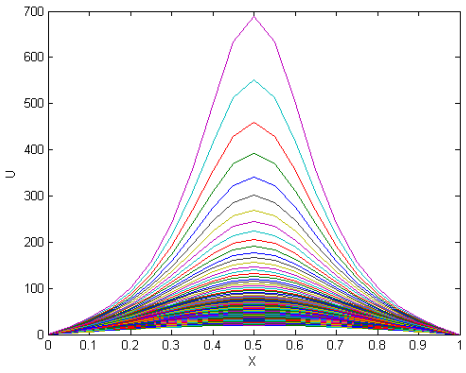


Figure 2.1: Solution of Fisher's equation for 165 time steps using an explicit method

We can clearly see that in this case we have a blow-up point located at $x^* = 0.5$. It is also the case that as our blow-up peak grows taller and narrower, we can see that our fixed mesh method fails to resolve the solution and with this being the case, under a fixed mesh method we would need to add extra nodes into our mesh in order to add extra resolution to the solution.

Another common issue with using a fixed mesh to tackle a blow-up problem is that the blow up point, x^*, may not actually be located at one of the nodes leading to an even greater failure of the fixed mesh method to resolve the solution of the blow-up problem.

2.3.2 Semi-Implicit Method

The stability condition referred to in Section (2.3.1) is the main weakness associated with using an explicit method in order to solve problems and so we now extend our investigation into fixed mesh solutions of blow-up problems by considering a semi-implicit solution method. As in Section (2.3.1) we begin by discretising Fisher's equation, however in this case we discretise our spatial derivative at the forward time which gives

$$\frac{\left(u_j^{n+1} - u_j^n\right)}{\Delta t} = \frac{\left(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}\right)}{\Delta x^2} + \left(u_j^n\right)^2,$$

and this rearranges to

$$u_j^{n+1} - \mu\left[u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}\right] = u_j^n + \Delta t\left(u_j^n\right)^2,$$

where again $\mu = \frac{\Delta t}{\Delta x^2}$. We may now solve this as a standard matrix problem of the form $A\underline{u}_j^{n+1} = \underline{u}_j^n + \Delta t(\underline{u}_j^n)^2$, where

$$A = \begin{pmatrix} 1+2\mu & -\mu & 0 & \dots & \dots & \dots & 0 \\ -\mu & 1+2\mu & -\mu & 0 & \dots & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & -\mu & 1+2\mu & -\mu & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & -\mu & 1+2\mu & -\mu \\ 0 & \ddots & \ddots & \ddots & \ddots & -\mu & 1+2\mu \end{pmatrix}.$$

We find that, in using the semi-implicit solution method we are able to relax out restriction on the size of Δt to obtain the solution in Figure (2.2). This gives a blow-up of similar magnitude of that in Section (2.3.1) however in this case we may use $\Delta x = 0.025$ and a time step of $\Delta t = 0.1$ which allows us to reach the solution in Figure (2.2) in just 5 time steps.

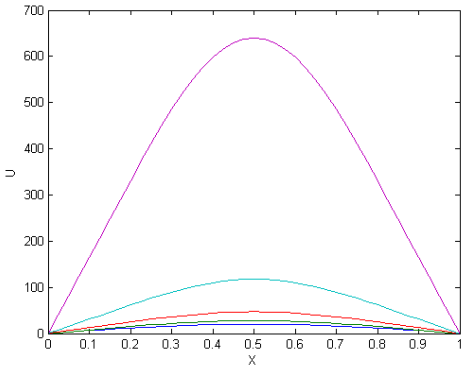


Figure 2.2: Solution of Fisher's equation for 5 time steps using a semi-implicit method

As in the previous case we can see that the blow-up point is located at $x^* = 0.5$. Using the semi-implicit solution method relaxes the restriction on our time step so that we may use a larger Δt meaning this method is computationally less expensive than the explicit method although again the fixed mesh will at some point fail to resolve the solution as blow-up occurs.

2.3.3 Evaluation of Fixed Mesh Methods

We have seen in both the explicit and implicit solution methods for Fisher's equation on a fixed mesh that as we begin to see blow-up developing, the fixed mesh will at some point in time will stop being able to resolve the blow-up, i.e. these methods will be unable to represent the narrowing of the blow-up peak as time goes by.

The other main issue we have when using a fixed mesh method occurs when the blow-up point lies between the nodes making up the mesh. This is an issue because we may end up cutting the largest part of the blow-up peak out of our approximate solution altogether or alternatively we may cause a situation where the location of the blow-up peak in mis-represented.

Clearly there are some quite serious issues associated with utilising a fixed mesh in order to examine a blow-up problem, and it is these issues which lead us to the development of moving mesh methods.

2.4 Method of Conservation for Fisher's Equation (with $p = 2$)

As explored in the previous sections, a method based upon a fixed mesh will fail to resolve the blow-up behaviour of our PDE. This makes it necessary to utilise an adaptive mesh method which will allow the blow-up behaviour to be resolved more accurately. As mentioned earlier there are three main adaptive methods which are used in the literature and the strategy which will be used

here is the form of r-refinement whereby we aim to conserve the fractional area under the curve as our solution evolves.

2.4.1 Generating Node Velocities

In order to guide the movement of the nodes making up our mesh we consider a velocity-based approach whereby each node is assigned a velocity with which it will move towards the blow-up point. This is recalculated at each time step and a new mesh is created. The velocity with which the nodes are allowed to move is calculated in such a way that the fractional area under the solution curve is conserved at each time step.

We begin by splitting our domain $x \in [0, 1]$ into two halves since we know the solution is symmetric. The domain now becomes $x \in [0.5, 1]$, which is divided into N equally sized intervals which shall be denoted by $(x_{j-1}(t), x_j(t))$, for $j = 1, 2, ..., J$. Using the domain of $x \in [0.5, 1]$ means we must have a boundary condition at $x = 0.5$ and this shall be given by $u_x = 0$. The area under the solution curve for each interval is given by

$$a_j = \int_{x_{j-1}(t)}^{x_j(t)} u(x, t)dx. \tag{2.4}$$

We may then combine the sum of each of these areas to give us the area under the solution curve for the entire domain, which shall be called θ. This gives

$$\theta(t) = \int_{0.5}^{1} u(x, t)dx. \tag{2.5}$$

The rate of change of the area under the solution curve may be obtained by differentiating (2.5) with respect to time in order to obtain

$$\dot{\theta} = \int_{0.5}^{1} u_t dx, \tag{2.6}$$

into which we may substitute (2.2) in place of u_t, giving

$$\begin{aligned}\dot{\theta} &= \int_{0.5}^{1} u_{xx} + u^2 dx \\ &= [u_x]_{0.5}^{1} + \int_{0.5}^{1} u^2 dx. \end{aligned} \tag{2.7}$$

In the method of conservation we hold the fractional area of the regions under the solution curve constant over time, i.e.

$$\frac{1}{\theta}\int_{x_{j-1}(t)}^{x_j(t)} u(x,t)dx = \text{constant}. \tag{2.8}$$

Differentiation of (2.8) with respect to time yields

$$\frac{d}{dt}\left[\frac{1}{\theta}\int_{x_{j-1}(t)}^{x_j(t)} u(x,t)dx\right] = 0, \tag{2.9}$$

which, since we are differentiating under the integral sign we may utilise the Leibniz integral rule in order to obtain

$$0 = -\frac{1}{\theta^2}\dot{\theta}\int_{x_{j-1}(t)}^{x_j(t)} u(x,t)dx + \frac{1}{\theta}\left[\int_{x_{j-1}(t)}^{x_j(t)} \frac{\partial u}{\partial t}dx + [uv]_{x_{j-1}}^{x_j}\right].$$

Using (2.2) and (2.4), this may be re-written as

$$\begin{aligned} 0 &= -\frac{\dot{\theta}}{\theta}a_j + \int_{x_{j-1(t)}}^{x_j(t)} u_{xx} + u^2 dx + [uv]_{x_j-1}^{x_j} \\ &= -\frac{\dot{\theta}}{\theta}a_j + [u_x]_{x_{j-1}}^{x_j} + \int_{x_{j-1(t)}}^{x_j(t)} u^2 dx + u_j v_j - u_{j-1}v_{j-1}. \end{aligned}$$

Finally, we may rearrange this to obtain an expression for the velocity of each node, which is given by

$$v_j = -\frac{1}{u_j}\left[-\frac{\dot{\theta}}{\theta}a_j + [u_x]_{x_{j-1}}^{x_j} + \int_{x_{j-1(t)}}^{x_j(t)} u^2 dx - u_{j-1}v_{j-1}\right], \tag{2.10}$$

so long as $u_j \neq 0$, and this may be solved sequentially since we know that $v_0 = 0$ and this means that the velocity of each node is uniquely defined for each time

step. At $j = N$, v is not actually defined by (2.10) however we take $v_N = 0$.

2.4.2 Generation of New Meshes and Total Area

In order to generate the new mesh and total area under the solution curve to be used at the next time level we use a simple explicit Euler time stepping method.

We begin by making the observation that

$$v_j = \frac{dx_j}{dt},$$

which allows us to generate our new grid using the expression

$$x_j^{n+1} = x_j^n + \Delta t \; v_j^n.$$

We then generate the new area under the solution curve (θ) in a very similar way using

$$\theta_j^{n+1} = \theta_j^n + \Delta t \; \dot{\theta}_j^n.$$

2.4.3 Recovering the Solution

As mentioned earlier we take (2.8), which also implies

$$\frac{1}{\theta} \int_{x_{j-1}(t)}^{x_{j+1}(t)} u(x,t)dx$$

to be constant in time. Since we take this to be constant in time we may deduce that

$$\frac{1}{\theta(t)} \int_{x_{j-1}(t)}^{x_{j+1}(t)} u(x,t)dx = \frac{1}{\theta(0)} \int_{x_{j-1}(0)}^{x_{j+1}(0)} u(x,0)dx,$$

to which we may apply the mid-point rule as an approximation in order to give

$$\begin{aligned} \frac{1}{\theta(t)} u_j(t)[x_{j+1}(t) - x_{j-1}(t)] &= \frac{1}{\theta(0)} u_j(0)[x_{j+1}(0) - x_{j-1}(0)] \\ \Rightarrow u_j(t) &= u_j(0) \frac{\theta(t)[x_{j+1}(0) - x_{j-1}(0)]}{\theta(0)[x_{j+1}(t) - x_{j-1}(t)]}. \end{aligned}$$

2.4.4 Adaptive Time-Step

In order to allow our method to track the bahaviour of the solution appropriately we may use the scaling argument given in Budd et al. [3] to create a condition for varying the length of the time step. Budd et al. [3] state that Fisher's equation is invariant under the rescaling

$$(T - \bar{t}) = \lambda(T - t), \tag{2.11}$$

$$\bar{u} = \lambda^{\frac{-1}{(p-1)}} u, \tag{2.12}$$

$$(\bar{x} - x^*) = \lambda^{\frac{1}{2}} (x - x^*), \tag{2.13}$$

This paper then extends this to say that the blow-up peak of our solution is approximately self-similar under (2.11 - 2.13) with

$$u(x,t) = \left(\frac{1}{p-1}\right)^{\frac{1}{p-1}} (T-t)^{\frac{-1}{p-1}} \left(1 + \left(\frac{p-1}{4p}\right)\mu^2\right)^{\frac{-1}{p-1}}, \tag{2.14}$$

where

$$\mu(x,t) = (x - x^*)(T-t)^{\frac{-1}{2}} |\log(T-t)|^{\frac{-1}{2}},$$

and as stated earlier we have chosen $p = 2$.

With this in mind we are able to choose a variable time step which will fit with the characteristics of the solution and in order to do this we scale the time step by $\frac{1}{(T-t)}$. This means that as $t \to T$ our time step will vary according to

$$\Delta t = \frac{\Delta t_0}{T - t}. \tag{2.15}$$

In order to utilise this scheme we must use an approximation T which is given by Budd et al. [3] as $T \approx 0.082372$

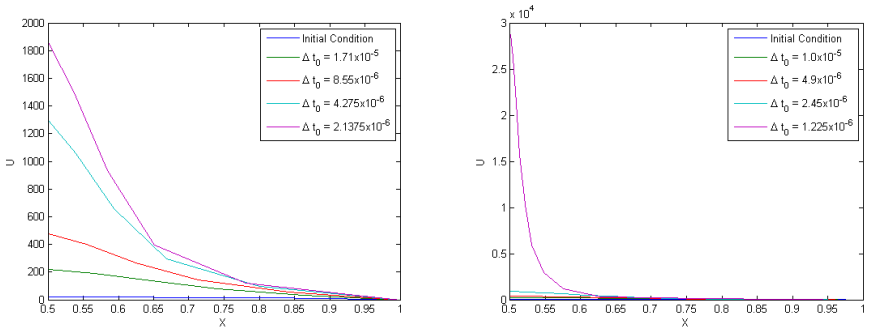
In order to explore the behaviour of the solution using the method outlined in the previous sections it is useful to look at a number of different cases both in terms of the number of nodes used to make up the mesh and the length of the initial time-step. The various cases used for these are displayed in Table (2.1).

Nodes	Δt_0	Nodes	Δt_0	Nodes	Δt_0
11	1.71×10^{-5}	21	1×10^{-5}	41	1×10^{-5}
11	8.55×10^{-6}	21	4.9×10^{-6}	41	15×10^{-6}
11	4.275×10^{-6}	21	2.45×10^{-6}	41	2.5×10^{-6}
11	2.1375×10^{-6}	21	1.225×10^{-6}	41	1.24×10^{-6}

Table 2.1: Cases considered for the number of nodes in $x \in [0, 1]$ and Δt_0 in Fisher's equation

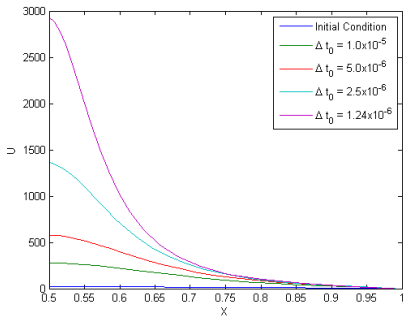
2.5 Results for Fisher's equation ($p = 2$)

In order to demonstrate the results obtained from the above method we begin by showing the solution $u(x,t)$ at the final time step for every combination of Δt_0 and number of nodes as given in Table (2.1), and these are shown in Figure (2.3). Each of these sets of result is obtained utilising the initial condition $u(x,0) = 20\sin(\pi x)$. In Figure (2.3) we can see that even using relatively few



(a) 11 Nodes

(b) 21 Nodes



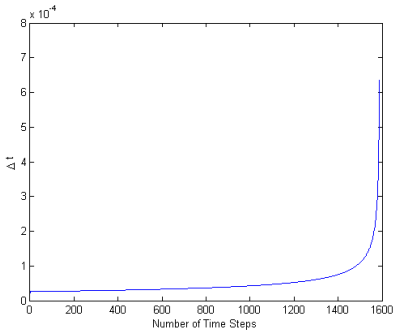
(c) 41 Nodes

Figure 2.3: Solution of Fisher's equation (with $p = 2$) at the final time step

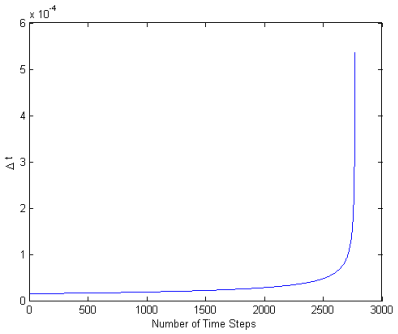
nodes will still allow the blow-up solution to be resolved as time passes. The difference in the largest values of u in the three subfigures is due to the size of the initial time-steps as well as the subsequent time-steps. In order to obtain a larger maximum value for u it is necessary to choose a Δt_0 such that we obtain a final time level which is very close to our blow-up time, T. As we can see in Figure (2.3 (b)) it is the initial condition of $\Delta t_0 = 1.225 \times 10^{-6}$ which gives us

a final time level closest to T. In this case $\Delta t = 1.225 \times 10^{-6}$ is the smallest initial time step considered however it is not necessarily the case that a smaller time step will allow us to get closer to the blow-up time, T. This will depend on the evolution of the time step as described in Section (2.4.4).

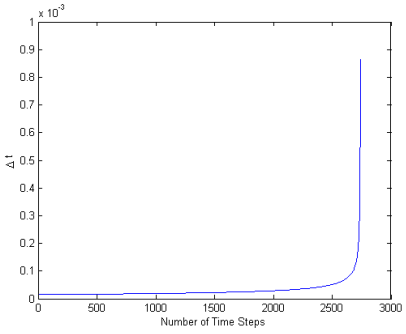
In attempting to understand the behaviour of our model it is important to examine the manner in which the time-steps evolve over time and it is the knowledge of this time-step development which allows us to approach the final value of T required in order to obtain a very large maximum value of u. In Figure (2.4) we plot the size of the time-step against the number of time-steps taken in order to see this development of the time-step in the cases of the three smallest Δt_0 values in Table (2.1).



(a) $\Delta t_0 = 2.1375 \times 10^{-6}$



(b) $\Delta t_0 = 1.225 \times 10^{-6}$



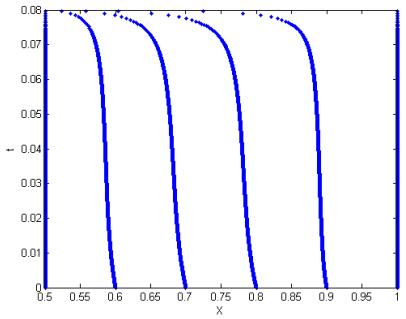
(c) $\Delta t_0 = 1.24 \times 10^{-6}$

Figure 2.4: Development of time-steps

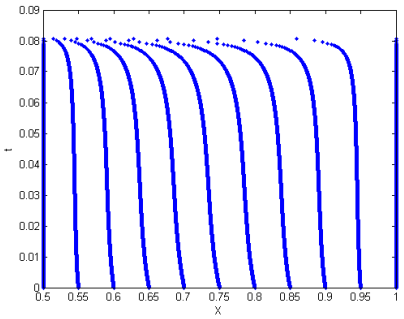
From Figure (2.4) we are able to see that for any initial time step chosen, the length of the time step will begin to increase very slowly at first, but as t

becomes close to T the length of the time step begins to increase very rapidly.

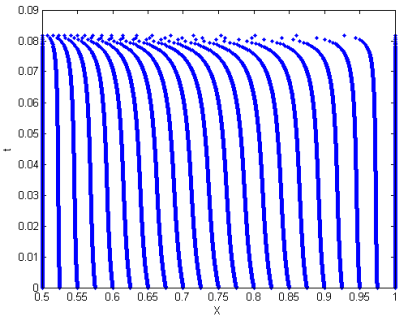
It is also useful to examine the movement of the nodes as time passes as this gives us a good idea of how the method of conservation works in terms of redistributing mesh points to help with the resolution of the blow-up behaviour. Figure (2.5) shows how the mesh evolves over time.



(a) 11 nodes, $\Delta t_0 = 2.1375 \times 10^{-6}$



(b) 21 nodes, $\Delta t_0 = 1.225 \times 10^{-6}$



(c) 41 nodes, $\Delta t_0 = 1.24 \times 10^{-6}$

Figure 2.5: Mesh evolution

We can see from Figure(2.5) that, as time passes, each of the nodes which do not have a fixed location will move in towards the blow-up point, x^*. It is also possible to see that as we get closer to our final time level, the nodes begin to move in towards x^* much faster due to the value of u increasing faster at x^* as we get closer to T.

2.6 Can We Allow Moving Boundary Nodes?

Since we know that $v_N = 0$ is chosen arbitrarily in order to fix the boundary node and hence maintain the length of our domain, it may be of use to consider the possibility of allowing the boundary node to have a velocity other than $v_N = 0$.

With this is in mind we investigate a possible solution to this problem which is to allow the boundary node to move with a certain velocity whilst maintaining its fixed value of $u = 0$. This method has been applied to Fisher's equation here in order to examine what effect this has on our solution when compared to those obtained in the previous sections where we had a fixed boundary node.

In order to assign a velocity to the boundary node it is necessary to consider a different method to that laid out in Equation (2.10) since the value of u_j for the boundary node will be equal to zero and we may therefore not divide by it. To overcome this issue we consider a linear extrapolation of the velocities assigned to the two nearest nodes as shown in Figure (2.6), where the dotted line represents the linear extrapolation.

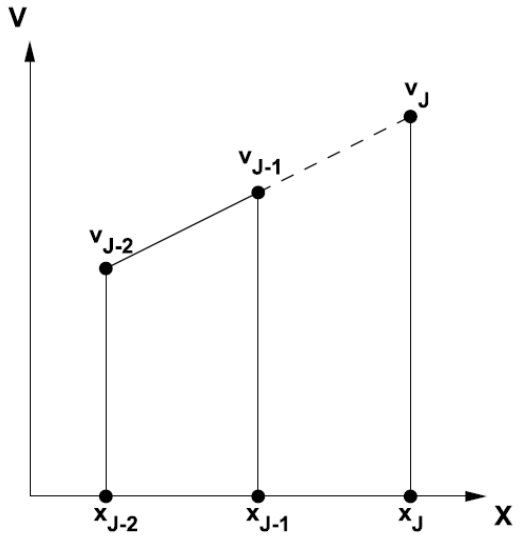


Figure 2.6: Demonstration of the method of linear extrapolation of velocities

We may express this linear extrapolation of the velocity using the expression

$$v_J = v_{J-1} + (x_J - x_{J-1})\frac{(v_{J-1} - x_{J-2})}{(x_{J-1} - x_{J-2})}, \tag{2.16}$$

where the J subscript represents the boundary node.

The examples considered here are as laid out in Table (2.2) where Δt_0 represents the initial time step and all subsequent time steps are determined by Equation (2.15).

Nodes in $x \in [0,1]$	11	21	41
Δt_0	2.1375×10^{-6}	1.225×10^{-6}	1.24×10^{-6}

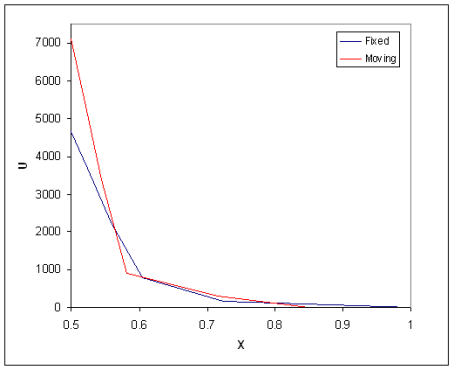
Table 2.2: Number of nodes and initial time steps used for comparison of fixed and moving boundary node methods

A comparison of the results obtained using these two different methods are shown in Figure (2.7). Each of the lines in this figure represents the solution obtained at the final time-step.
We also examine the difference in the maximum values of u obtained at the final time-step in order to look at the effect on our solution of allowing our boundary node to move. This is shown in Table (2.3) where the percentage refers to the percentage of the value of u_{MAX} obtained using a fixed boundary node method which can be achieved when using a moving boundary node method.

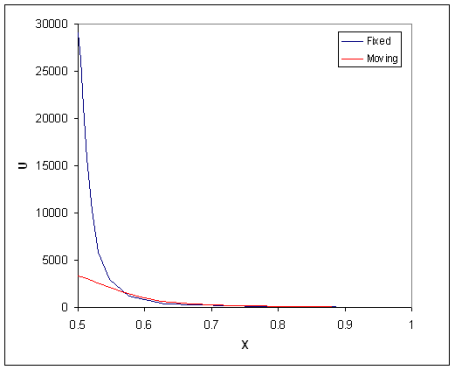
Nodes	Fixed Boundary Node	Moving Boundary Node	Percentage
11	4664.85	7100.06	152.20%
21	29144.84	3292.37	11.30%
41	2933.36	1828.48	62.33%

Table 2.3: Comparison of results obtained using a fixed and a moving boundary node method

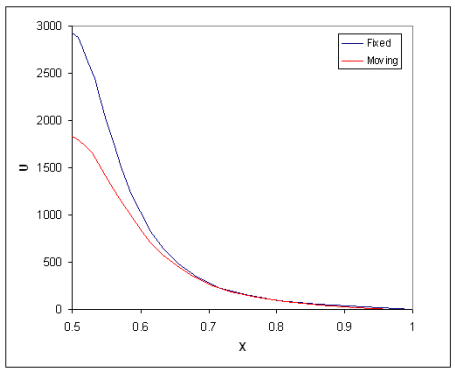
We can see from Figure (2.7) that allowing a method whereby the boundary node is allowed to move does not have a huge effect in terms of the general shape of the solution which would suggest that there may be some potential in this method, however upon examination of Table (2.3) we see that there does not appear to be any reliable pattern in how close the two methods final time



(a) 11 nodes, $\Delta t_0 = 2.1375 \times 10^{-6}$



(b) 21 nodes, $\Delta t_0 = 1.225 \times 10^{-6}$



(c) 41 nodes, $\Delta t_0 = 1.24 \times 10^{-6}$

Figure 2.7: Comparison of results obtained using a fixed and a moving boundary node method

solutions are associated with the number of nodes or the size of the time step.

The fact that this method of moving boundary nodes appears to be rather unpredictable in terms of how close to the fixed boundary node solution the results we obtain are would suggest that this method should not be applied within the conservative moving mesh methods discussed here.

2.7 Method of Conservation for Fisher's Equation (with $p = 3$)

We now move on to considering the Fisher's equation with a different power, p, which in this case will be chosen as $p = 3$ which gives us Equation (2.3), i.e.

$$u_t = u_{xx} + u^3.$$

Budd et al. [1] state that for Fisher's equation taking the form in Equation (2.1), we must conserve the quantity

$$\int u^{p-1} dx,$$

and so in this case we conserve

$$\int u^2 dx. \tag{2.17}$$

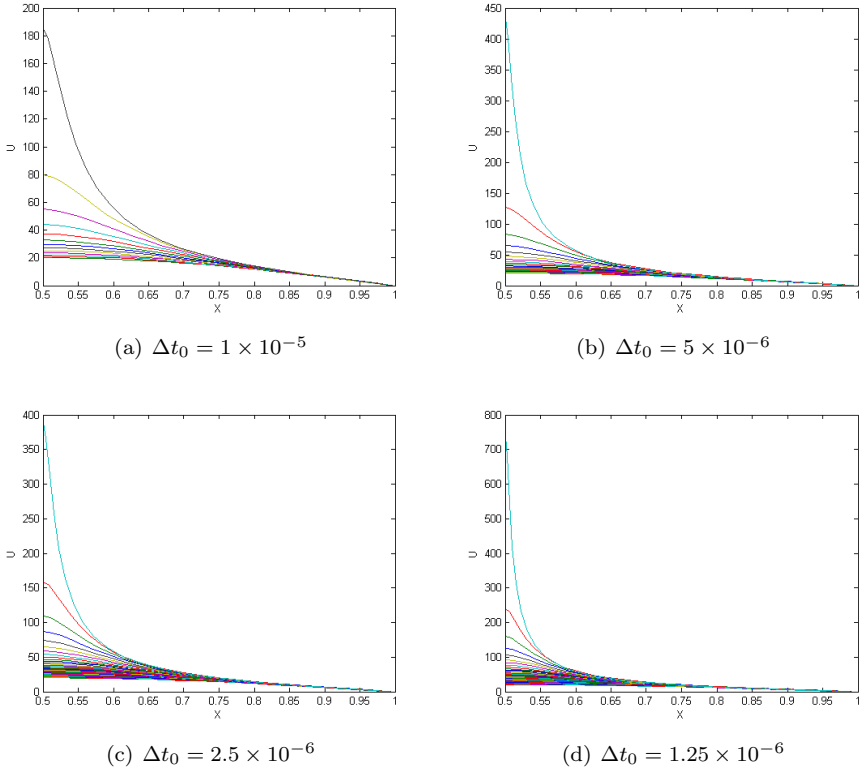
The method we will use in this section is exactly as laid out in Section (2.4) except for the fact that we will conserve the value in Equation (2.17) instead of the conserved value of

$$\int u dx$$

which was conserved in the case where $p = 2$. The cases which shall be considered here all use 41 nodes in $x \in [0, 1]$ and we utilise the Δt_0 values from the $p = 2$ case of Fisher's equation.

2.8 Results for Fisher's equation ($p = 3$)

We begin by illustrating the behaviour of our solution as time passes. As mentioned previously, here we use 41 mesh points over the domain $x \in [0, 1]$ and we use the same initial time-steps, Δt_0 as we used in the case with $p = 2$. In each of these cases we have shown the development of the solution in Figure (2.8).



(a) $\Delta t_0 = 1 \times 10^{-5}$ (b) $\Delta t_0 = 5 \times 10^{-6}$

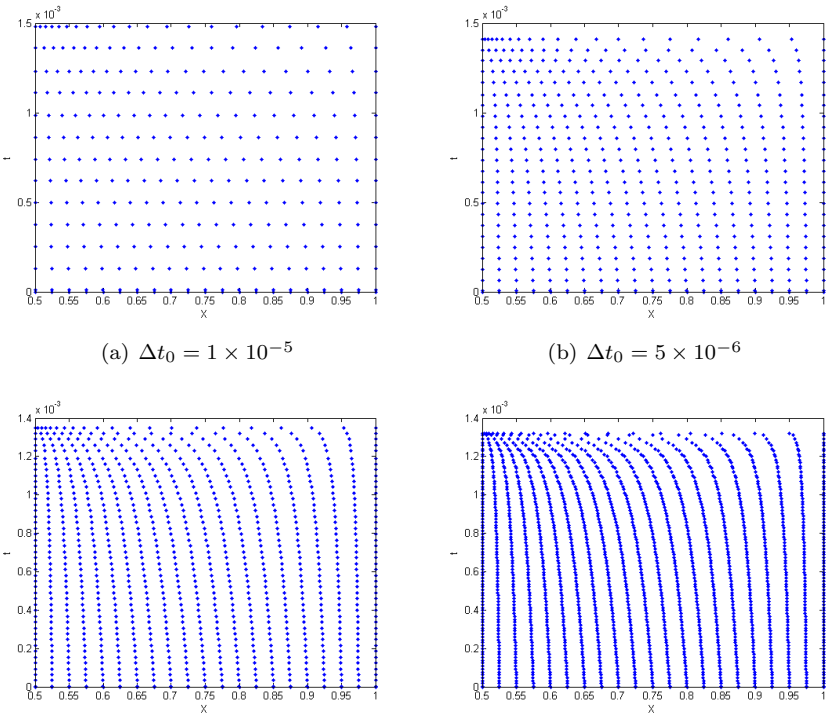
(c) $\Delta t_0 = 2.5 \times 10^{-6}$ (d) $\Delta t_0 = 1.25 \times 10^{-6}$

Figure 2.8: Solution of Fisher's equation (with $p = 3$) at the final time step

As before we are able to see that the largest value of u that we are able to obtain is dependent on both the initial time-step and its development over time. We can also see that in this case (where $p = 3$) we reach our largest solution in a much smaller number of time-steps than we were able to in the $p = 2$ case. The reason for this is that the u^p term is the blow-up term, and in the case where $p = 3$ this blow-up will occur faster due to the greater power of u.

We also examine the evolution of our mesh as time passes in order to see how the method works in terms of moving the mesh points in order to allow the resolution of the solution to be maintained.

Similar to the previous case we can see that as time passes, the nodes without a fixed position move in towards the blow-up point. The main difference between this case and the one considered previously is that it takes many fewer time-steps in order to reach the blow-up time, T. This can be seen by the fact that



(a) $\Delta t_0 = 1 \times 10^{-5}$ (b) $\Delta t_0 = 5 \times 10^{-6}$

(c) $\Delta t_0 = 2.5 \times 10^{-6}$ (d) $\Delta t_0 = 1.25 \times 10^{-6}$

Figure 2.9: Mesh evolution in Fisher's equation with $p = 3$

the time at which the mesh points really accelerate towards the blow-up point occurs much earlier than in the $p = 2$ case.

Chapter 3

Alternative 'Fisher Type' Equations

We now move the investigation into moving mesh methods on to look into their application for some further examples of 'Fisher type' equations.

3.1 Introduction to 'Fisher Type' Equations

According to Ockendon et al. [9], 'Fisher type' equations are examples of semi-linear reaction-diffusion equations and they take the form

$$u_t = u_{xx} + f(u, x, t). \tag{3.1}$$

Ockendon et al. [9] go on to state that these equations often appear in models of population dynamics, where the function $f(u, x, t)$ can be either positive or negative depending upon the process which is being modelled.

The two examples of 'Fisher type' equations which shall now be considered, in addition to those in Section (2), are the 'traditional' Fisher's equation and the Cahn-Allen equation.

The 'traditional' Fisher's equation is given by Equation (3.1), where $f =$

$u(1-u)$, i.e.

$$u_t = u_{xx} + u(1-u). \tag{3.2}$$

This equation may be used in the modelling of many scientific phenomena. In particular Qiu and Sloan [10] use Equation (3.2) to represent the simultaneous growth an spread of a dominant gene. Ockendon et al. [9] give $f = u - u^2$, and state that Equation (3.2) models a species' population where the u term models a linear birth rate and the $1-u$ factor adds the limiting effect of a finite food supply.

We also investigate the Cahn-Allen equation (often referred to as Allen-Cahn) which is given by Equation (3.1) with $f = u(1-u^2)$, i.e.

$$u_t = u_{xx} + u(1-u^2). \tag{3.3}$$

Ockendon et al. [9] give $f = u - u^3$ and state that u represents the fraction of a material undergoing a phase change from a stable phase at $u = -1$ to another at $u = 1$. Zhang and Du [5] give a number of uses of the Cahn-Allen equation which include phase transitions and interface dynamics in materials science, nonlinear waves, vortex dynamics, superconductivity, superfluidity, liquid crystals and strings in field theory.

3.2 Choosing an Appropriate Monitor Function

In the previous sections we have seen the importance of choosing a suitable monitor function for a particular problem. With this in mind we look to the literature for suggestions as to what may be a suitable monitor function for use in the 'traditional' Fisher's and the Cahn-Allen equations.

Due to the different nature of these 'Fisher type' equations compared to the blow-up PDEs considered in the previous sections, it would appear unwise to attempt to use the monitor functions considered previously. Qiu and Sloan [10] give a number of possible monitor functions which may be used in the

'traditional' Fisher's equation.

The first of these monitor functions given by Qiu and Sloan [10] is a standard arc-length monitor function. This takes the form

$$M(x,t) = \sqrt{1 + \alpha^2 \left(\frac{\partial u}{\partial x}\right)^2}, \tag{3.4}$$

in which α is a user-specified parameter that is given a value of $\alpha = 2$ in the paper by Qiu and Sloan [10]. Another monitor function considered in this same paper is that of a curvature monitor function of the form

$$M(x,t) = \left(1 + \alpha^2 \left(\frac{\partial^2 u}{\partial x^2}\right)^2\right)^{\frac{1}{4}}, \tag{3.5}$$

where α is again a user-specified parameter taking the value $\alpha = 2$ in the literature. Qiu and Sloan [10] then go on to define a modified monitor function which they believe captures the characteristics of the model effectively and this is given in [10]. Another monitor function considered in this same paper is that of an extended curvature monitor function of the form

$$M(x,t) = \left[1 + \alpha^2(1-u)^2 + \beta^2(a-u)^2 \left(\frac{\partial^2 u}{\partial x^2}\right)^2\right]^{\frac{1}{2}}, \tag{3.6}$$

and in this case we have three user-defined parameters α, β and a which are given the values $\alpha = 1.5$, $\beta = 0.1$ and $a = 1.015$ in the paper by Qiu and Sloan [10].

In this investigation, we are concerned with implementing velocity-based moving mesh methods and with this in mind it will be the arc-length monitor function,

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

which shall be applied in subsequent sections. This is chosen as it should be suitable for use with both the 'traditional' Fisher's equation and the Cahn-Allen equation due to the fact that they share many similar characteristics.

3.3 'Traditional' Fisher's Equation

3.3.1 Problem Formation

During the investigation into the 'traditional' Fisher's equation we shall consider the problem as laid out by Qiu and Sloan [10]. This gives the equation as

$$u_t = u_{xx} + u(1-u), \tag{3.7}$$

and shall be considered in the domain $x \in [0,1]$.

It is also necessary to define both initial and boundary conditions in order to be able to solve this problem. The boundary conditions given in [10] are

$$\lim_{x\to-\infty} u(x,t) = 1 \quad \text{and} \quad \lim_{x\to\infty} u(x,t) = 0,$$

and so we consider an approximate version of these which shall be given by

$$u(0,t) = 1 \quad \text{and} \quad u(1,t) = 0,$$

which allow us to utilise the domain given above. We must also choose an initial condition which satisfies the boundary conditions. With this in mind we look again to Qiu and Sloan [10], who take an initial condition of the form

$$u(x,0) \sim e^{-\beta x},$$

where β is a user defined parameter. In order to utilise the domain and boundary conditions given above, we must first choose a suitable value for β. The boundary condition at $x = 0$ will be satisfied for any value of β however the other boundary condition may not be precisely satisfied. With this in mind we choose a value of $\beta = 5$ and truncate the function at $x = 1$, where we shall set a value of $u(1,0) = 0$ in order to meet the boundary condition. This gives our

initial condition as

$$u(x,0) = \begin{cases} e^{-5x} & \text{for } 0 \leq x < 1 \\ 0 & \text{for } x = 1. \end{cases}$$

3.4 Method for the 'Traditional' Fisher's Equation

As discussed in Section (3.2), we create a velocity based method whereby an arc-length monitor function is used in order to guide the movement of the nodes. The method which shall be used for this problem is essentially a hybrid of two separate monitors since we use an area monitor to avoid a complicated inversion process which would be necessary were we to use an arc-length monitor in the retrieval of new approximations of the solution.

3.4.1 Generation of Nodal Velocities

We begin by splitting the domain $x \in [0,1]$ into N equally sized intervals $(x_{j-1}(t), x_j(t))$, for $j = 1, 2, ..., J$. The arc-length of the solution curve for each of these intervals is given by

$$l_j = \int_{x_{j-1}(t)}^{x_j(t)} M dx, \tag{3.8}$$

where $M = \sqrt{1 + u_x^2}$.

It is then possible to combine the arc-lengths for each of these intervals to give the arc-length of the entire domain, which shall be denoted γ. This gives

$$\gamma(t) = \int_0^1 M dx. \tag{3.9}$$

We may now differentiate γ with respect to time in order to give the rate of change of the entire arc-length. This is given by

$$\dot{\gamma} = \frac{d}{dt} \int_0^1 M dx, \tag{3.10}$$

to which we may apply the Leibniz integral rule to obtain

$$\dot{\gamma} = \int_0^1 \frac{\partial M}{\partial t} dx = \int_0^1 \frac{u_x u_{tx}}{\sqrt{1+u_x^2}} dx.$$

We are now able to substitute Equation (3.7) into this in order to obtain

$$\dot{\gamma} = \int_0^1 \frac{u_x}{\sqrt{1+u_x^2}} \frac{\partial}{\partial x}(u_{xx} + u(1-u)) dx.$$

In each interval we hold $u_x/\sqrt{1+u_x^2}$ constant and so we may take this outside of the integral to obtain

$$\dot{\gamma} = \sum_{j=1}^{J} \left. \frac{u_x}{\sqrt{1+u_x^2}} \right|_{x_{j-\frac{1}{2}}(t)} [u_{xx} + u(1-u)]_{x_{j-1}(t)}^{x_j(t)}.$$

Within this method we hold the fractional arc-length in each interval to be constant over time, i.e.

$$\frac{1}{\gamma} \int_{x_{j-1}(t)}^{x_j(t)} M dx = \text{constant}.$$

Differentiation with respect to time now yields

$$\frac{d}{dt}\left[\frac{1}{\gamma} \int_{x_{j-1}(t)}^{x_j(t)} M dx\right] = 0,$$

to which the Leibniz integral rule may be applied in order to obtain

$$0 = \frac{-1}{\gamma^2}\dot{\gamma} \int_{x_{j-1}(t)}^{x_j(t)} M dx + \frac{1}{\gamma}\left[\int_{x_{j-1}(t)}^{x_j(t)} \frac{\partial M}{\partial t} dx + [Mv]_{x_{j-1}(t)}^{x_j(t)}\right].$$

We may now substitute Equation (3.8) into this to give

$$\begin{aligned} 0 &= \frac{-\dot{\gamma}}{\gamma} l_j + \int_{x_{j-1}}^{x_j} \frac{\partial M}{\partial t} dx + M_j v_j - M_{j-1} v_{j-1} \\ &= \frac{-\dot{\gamma}}{\gamma} l_j + \left. \frac{u_x}{\sqrt{1+u_x^2}} \right|_{x_{j-\frac{1}{2}}} \cdot [u_{xx} + u(1-u)]_{x_{j-1}}^{x_j} + M_j v_j - M_{j-1} v_{j-1}. \end{aligned}$$

This may now be rearranged to give an expression for the nodal velocity which is given by

$$v_j = \frac{-1}{M_j}\left[\frac{\dot{\gamma}}{\gamma}l_j - M_{j-1}v_{j-1} + [u_{xx} + u(1-u)]_{x_{j-1}}^{x_j} \cdot \frac{u_x}{M}\Big|_{x_{j-\frac{1}{2}}}\right].$$

3.4.2 New Mesh and Arc-Length Creation

In order to create the new meshes and total arc-length of the solution curve, we use a simple explicit Euler time stepping method.

This allows us to generate a new grid using the expression

$$x_j^{n+1} = x_j^n + \Delta t\ v_j^n.$$

The same method is then used to generate a new arc-length for the entire domain which is given by

$$\gamma_j^{n+1} = \gamma_j^n + \Delta t\ \dot{\gamma}_j^n.$$

3.4.3 Recovery of New Approximations

We now resort to the use of the previous monitor function in order to obtain new approximations of the solution.

Firstly, we calculate the area under the solution curve for the entire domain, which shall be denoted by θ, and is given by

$$\theta(t) = \int_0^1 u(x,t)dx.$$

We now differentiate with respect to time, which gives

$$\dot{\theta} = \frac{d}{dt}\int_{x_{j-1}}^{x_j} u dx,$$

and we may now apply the Leibniz integral rule to this, which yields

$$\dot{\theta} = \int_{x_{j-1}}^{x_j} u_t dx + [uv]_{x_{j-1}}^{x_j}.$$

Equation (3.8) may now be substituted into this to give

$$\begin{aligned} \dot{\theta} &= \int_{x_{j-1}}^{x_j} (u_{xx} + u(1-u))dx + [uv]_{x_{j-1}}^{x_j} \\ &= [u_x]_{x_{j-1}}^{x_j} + \int_{x_{j-1}}^{x_j} u(1-u)dx + [uv]_{x_{j-1}}^{x_j}. \end{aligned}$$

It is now possible to obtain all of these values from previous stages of the method.

A simple explicit Euler time stepping method is used to generate a new total area under the solution curve which is given by

$$\theta^{n+1} = \theta^n + \Delta t \, \dot{\theta}^n.$$

We then use this new θ value to give

$$\int_{x_{j-1}}^{x_{j+1}} u dx = \theta^{n+1},$$

to which we may apply the mid-point rule, giving

$$u_j^{n+1}[x_{j+1}^{n+1} - x_{j-1}^{n+1}] = \theta^{n+1},$$

and this may be rearranged to give an expression for the approximate solution which is

$$u_j^{n+1} = \frac{\theta^{n+1}}{[x_{j+1}^{n+1} - x_{j-1}^{n+1}]}.$$

3.5 Cahn-Allen Equation

3.5.1 Problem Formation

Before we may investigate the application of a velocity based moving mesh method to the Cahn-Allen equation, we must begin by laying out the problem as it will be considered here. Lyons et al. [11] carry out numerical simulations of this equation which is given by

$$u_t = u_{xx} + u - u^3, \tag{3.11}$$

and this shall be investigated on the domain $x \in [-1, 1]$.

It is also necessary to define both initial and boundary conditions for this problem. The boundary conditions given by Lyons et al. [11] are

$$u_x(-1, t) = 0 \quad \text{and} \quad u_x(1, t) = 0,$$

which hold at all time levels. We must now give an initial condition for consideration here, and this is again given by Lyons et al. [11] who give the initial condition as

$$u(x, 0) = \sin\left(\frac{5\pi x}{2}\right),$$

and this condition clearly satisfies the boundary conditions given above.

It is easily verified that this problem has two stable equilibria which are at $u(x) = \pm 1$ and also an unstable equilibrium at $u(x) = 0$.

3.6 Method for the Cahn-Allen Equation

The method used for the Cahn-Allen equation is exactly the same as that laid out in Section (3.4) for the 'traditional' Fisher's equation.

There are some differences in the expressions related to this method which result from the different problem that it is applied to. The expressions which

differ in the method for the Cahn-Allen equation are

$$\dot{\gamma} = \sum_{j=1}^{J} \frac{u_x}{\sqrt{1+u_x^2}}\bigg|_{x_{j-\frac{1}{2}}} \cdot \left[u_{xx} + u(1-u^2)\right]_{x_{j-1}}^{x_j},$$

$$\dot{\theta} = [u_x]_{x_{j-1}}^{x^{j+1}} + \int_{x_{j-1}}^{x_{j+1}} u(1-u^2)dx + [uv]_{x_{j-1}}^{x_{j+1}},$$

and

$$v_j = \frac{-1}{M_j}\left[\frac{-\dot{\gamma}}{\gamma} l_j - M_{j-1}v_{j-1} + \left[u_{xx} + u(1-u^2)\right]_{x_{j-1}}^{x_j} \cdot \frac{u_x}{M}\bigg|_{x_{j-\frac{1}{2}}}\right].$$

3.7 Numerical Results

In examining the results obtained for the 'traditional' Fisher's and Cahn-Allen equations we shall look at the evolution of the solution curves in addition to the movement of the nodes within each mesh.

The examples considered here both utilise a time step of $\Delta t = 1 \times 10^{-6}$ and a mesh with an initial nodal spacing of $\Delta x = 0.05$. We shall also consider the initial and boundary conditions as given in Sections (3.3.1) and (3.5.1).

3.7.1 Results for the 'Traditional' Fisher's Equation

As mentioned in Section (3.3.1), for the 'traditional' Fisher's equation, we consider the problem upon the domain $x \in [0, 1]$ and so we take a mesh consisting of 21 nodes.

We begin by looking at the results obtained for the solution curve, u. These are shown in Figure (3.1), in which the blue line shows the initial condition and the red represents the solution after 2999 time steps of length Δt.

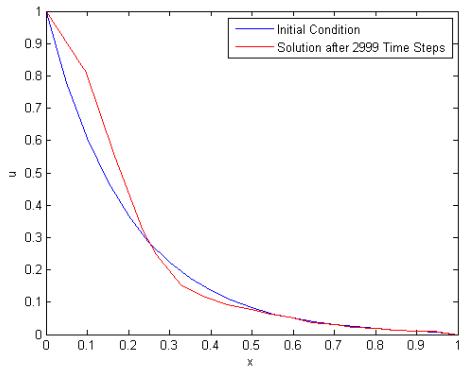


Figure 3.1: Solution obtained for the 'Traditional' Fisher's equation

Figure (3.1) displays the result after 2999 time steps, which is the final time step before we encounter an issue whereby nodes overtake one another. Upon examining this final solution curve we can see that the curve appears to have formed the interface which we would expect to see according to Qiu and Sloan [10].

The next stage is to look at the evolution of the mesh over time, which is displayed in Figure (3.2).

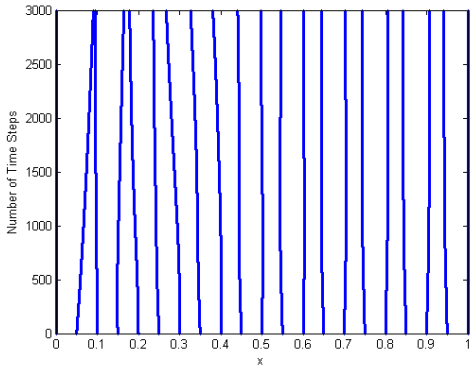


Figure 3.2: Mesh Evolution for the 'Traditional' Fisher's equation

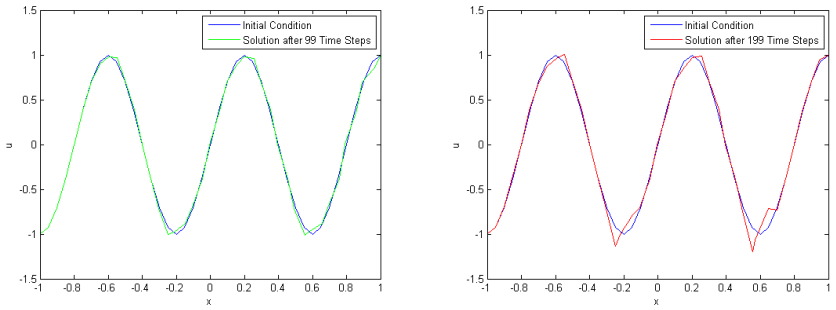
From Figure (3.2) we are able to see that in the left hand half of the domain, the nodes appear to be moving in towards the location at which the top of the interface forms. This is the behaviour we would wish to see when using this type of method as it would help to accurately resolve the solution at the ends

of the interface. Whilst there is a general pattern of nodal movement towards the desired location, the nodes move in at very different rates which may cause issues of nodal crossing before a final solution has been obtained.

3.7.2 Results for the Cahn-Allen Equation

In Section (3.5.1), we explain that we shall use the domain $x \in [-1, 1]$ for our investigations into the Cahn-Allen equation. With this in mind we consider a mesh consisting of 41 nodes which initially have an equal spacing.

We begin by examining the results which were obtained for the solution curve, u. The results obtained after 99 and 199 time steps are displayed in Figure (3.3).



(a) After 99 Time Steps (b) After 199 Time Steps

Figure 3.3: Solution obtained for the Cahn-Allen equation

We can see from Figure (3.3a) that after 99 time steps, the peaks and troughs in the solution curve appear to be moving towards eachother. Figure (3.3b) shows that after 199 time steps, the troughs of the solution curve appear to be developing some unphysical behaviour.

Next, we move on to examine the evolution of the mesh over the first 199 time steps, and this is shown in Figure (3.4).

It is clear to see that across most of the domain there is very little nodal movement. There is however some movement of nodes at locations around $x = -0.2$ and $x = 0.6$ which are the locations of the troughs of the solution curve. As was

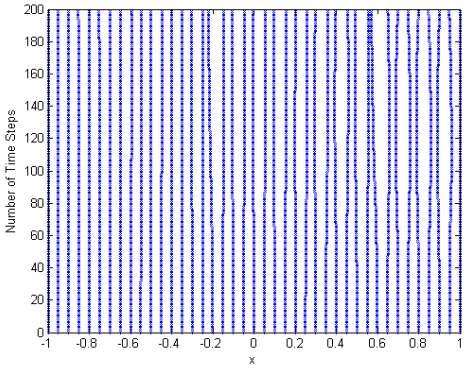


Figure 3.4: Mesh Evolution for the Cahn-Allen equation

the case with the 'Traditional' Fisher's equation, the nodes move at very different rates leading to the problem of nodal overtaking which causes a breakdown of the solution. This nodal overtaking causes the solution after 199 time steps to display serious amounts of unphysical behaviour.

3.8 Summary of Other 'Fisher Type' Equations

In this chapter we have laid out a method based on a hybrid of an arc-length monitor and an area monitor function.

When examining the application of this method to the 'Traditional' Fisher's equation we obtained some positive results both in terms of the solution curve and the mesh evolution. We then applied the same method to the Cahn-Allen equation. From the results obtained we are able to demonstrate very limited potential in so much as there was nodal movement in the correct locations, however the breakdown of physical behaviour within a small number of time steps may suggest a need to seek a more appropriate method.

Chapter 4

The Nonlinear Schrödinger Equation

We conclude the investigation into applications of moving mesh methods by examining the nonlinear Schrödinger equation.

4.1 Introduction to the Nonlinear Schrödinger Equation

In the literature the nonlinear Schrödinger equation (NLS) is expressed in many forms with the most common being

$$i\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial r^2} + \frac{d-1}{r}\frac{\partial u}{\partial r} + |u|^2 u = 0, \tag{4.1}$$

and

$$i\frac{\partial u}{\partial t} + \frac{1}{r^{d-1}}\frac{\partial}{\partial r}\left(r^{d-1}\frac{\partial u}{\partial r}\right) + |u|^2 u = 0, \tag{4.2}$$

where u is complex valued, r represents the radial coordinates and we must also have $d \geq 1$.

The NLS equation has applications within many and varied fields of science

as described in [12] and [13]. Budd et al. [12] and [13] state that this equation may be used to represent phenomena in both plasma physics and nonlinear optics.

As was the case with Fisher's equation, we may consider the NLS equation as a typical blow-up problem since, given suitable initial conditions, the solutions will display blow-up at a single blow-up point and this occurs within a finite blow-up time, T.

4.2 Problem Formation

In this investigation into the nonlinear Schrödinger equation we will consider the conservative form of the equation, and will look only at the case where $d = 2$ since there is known to be an 'exact' solution for the $d = 1$ case. This gives the NLS equation as

$$i\frac{\partial u}{\partial t} + \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + |u|^2 u = 0, \tag{4.3}$$

and it shall be considered on the domain $r \in [0, 1]$.

In order to solve this problem we must define both initial and boundary conditions. The boundary conditions considered for this particular problem are

$$\begin{aligned} u_r(0, t) &= 0 \\ u(1, t) &= 0, \end{aligned}$$

and we must choose an initial condition which satisfies the boundary conditions whilst also being large enough to ensure blow-up will occur. With this in mind we choose an initial condition similar to that chosen for Fisher's equation however we will adapt it such that the part in the domain $x \in [0.5, 1]$ is stretched to cover the domain of $r \in [0, 1]$ used for the NLS equation. This gives our initial condition as

$$u(r, 0) = 20 \sin\left(\pi\left(\frac{1 + r}{2}\right)\right).$$

According to Budd et al. [12] there are two quantities which are invariant over time which are the 'mass'

$$P = \int_0^\infty |u(r,t)|^2 r^{d-1} dr$$

and the energy

$$E = \int_0^\infty \left(\left| \frac{\partial u(r,t)}{\partial r} \right|^2 - \frac{1}{2}|u(r,t)|^4 \right) r^{d-1} dr.$$

In this case we shall seek to conserve the 'mass' of the solution over each interval.

Throughout the investigation into the NLS equation we take u to be

$$u = \phi + i\psi,$$

and this means our boundary conditions must hold for both the real and imaginary parts of the solution u.

4.3 'Mass' Conservative Method for the Nonlinear Schrödinger Equation

As with the examples considered for Fisher's equation, we consider a velocity based approach whereby a velocity is calculated for each node, with which it will move towards the blow-up point as time passes. The velocity assigned to each node is calculated in such a way that the mass of the solution held within each interval will remain constant at each time step.

4.3.1 Analysis of Properties

Before we attempt to produce solutions to any equation using a conservative moving mesh method it is useful to prove various features associated with the method to ensure that it should work for the method being considered.

Proof of Invariant Conserved Quantity

We begin our analytic investigation into the NLS equation by proving that the total integral of the quantity we aim to conserve will not vary over time. In other words we look to prove that

$$\frac{d}{dt}\int_0^R |u|^2 r dr = 0,$$

where R is the end of the domain.

In order to prove that this is true we take the NLS equation as given in Equation (4.2), and into this we may substitute $u = \phi + i\psi$ to obtain

$$i\frac{\partial}{\partial t}(\phi + i\psi) + \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial}{\partial r}(\phi + i\psi)\right) + (\phi^2 + \psi^2)(\phi + i\psi) = 0,$$

which may be rearranged and split into its real and imaginary parts. This gives expressions for $\dot{\phi}$ and $\dot{\psi}$, which are

$$\dot{\phi} = \frac{\partial \phi}{\partial t} = \frac{-1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \psi}{\partial r}\right) - \psi(\phi^2 + \psi^2). \tag{4.4}$$

$$\dot{\psi} = \frac{\partial \psi}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \phi}{\partial r}\right) + \phi(\phi^2 + \psi^2) \tag{4.5}$$

We now look at the time derivative of our conserved quantity, which is given by

$$\frac{d}{dt}\int_0^R (\phi^2 + \psi^2) r dr, \tag{4.6}$$

since $|u|^2 = \phi^2 + \psi^2$, and this may be re-written as

$$\int_0^R 2\phi\frac{\partial \phi}{\partial t} r dr + \int_0^R 2\psi\frac{\partial \psi}{\partial t} r dr. \tag{4.7}$$

It is now possible to substitute for $\dot{\phi}$ and $\dot{\psi}$ using Equations (4.4) and (4.5),

which gives

$$-\int_0^R 2\phi\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right)dr - \int_0^R 2\phi r\psi(\psi^2)dr - \int_0^R 2\phi r\psi(\phi^2)dr \tag{4.8}$$
$$+\int_0^R 2\psi\frac{\partial}{\partial r}\left(r\frac{\partial\phi}{\partial r}\right)dr + \int_0^R 2\psi r\phi(\phi^2)dr + \int_0^R 2\psi r\phi(\psi^2)dr.$$

We may then simplify Equation (4.8) to leave

$$\int_0^R 2\psi\frac{\partial}{\partial r}\left(r\frac{\partial\phi}{\partial r}\right)dr - \int_0^R 2\phi\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right)dr. \tag{4.9}$$

The next stage is to integrate each of the terms in Equation (4.9) by parts which will leave us with

$$\left[2\psi r\frac{\partial\phi}{\partial r}\right]_0^R - 2\int_0^R \frac{\partial\psi}{\partial r}r\frac{\partial\phi}{\partial r}dr - \left[2\phi r\frac{\partial\psi}{\partial r}\right]_0^R + 2\int_0^R \frac{\partial\phi}{\partial r}r\frac{\partial\psi}{\partial r}dr, \tag{4.10}$$

from which the integral terms cancel leaving us with

$$\frac{d}{dt}\int_0^R(\phi^2+\psi^2)rdr = \left[2\psi r\frac{\partial\phi}{\partial r}\right]_0^R - \left[2\phi r\frac{\partial\psi}{\partial r}\right]_0^R \tag{4.11}$$

$$= 2\psi(R)R\frac{\partial\phi(R)}{\partial r} - 2\psi(0)0\frac{\partial\phi(0)}{\partial r} - 2\phi(R)R\frac{\partial\psi(R)}{\partial r} + 2\phi(0)0\frac{\partial\psi(0)}{\partial r},$$

which, using the boundary conditions $u_r(0,t) = 0$ and $u(1,t) = 0$ is equal to zero, and hence we have proven that

$$\frac{d}{dt}\int_0^R |u|^2 rdr = \frac{d}{dt}\int_0^R(\phi^2+\psi^2)rdr = 0.$$

Derivation of Velocity Formula

As mentioned earlier we aim to conserve the partial 'mass' of our solution over each interval. In order to do this we must define a monitor function, which according to Twigger [14] is given by

$$M = |u|^2 = \phi^2 + \psi^2.$$

In order to derive a formula for the velocity of the mesh nodes, we begin by considering the expression

$$Mv = -\int_0^{r_j} \frac{\partial M}{\partial t} r dr,$$

which may be rearranged to give

$$v = \frac{-1}{(\phi^2 + \psi^2)} \int_0^{r_j} \left(2\phi \frac{\partial \phi}{\partial t} + 2\psi \frac{\partial \psi}{\partial t} \right) r dr.$$

We may then substitute for $\frac{\partial \phi}{\partial t}$ and $\frac{\partial \psi}{\partial t}$ using Equations (4.4) and (4.5), giving

$$\begin{aligned} v = \frac{-1}{(\phi^2 + \psi^2)} \Bigg(& \int_0^{r_j} \left(2\phi \frac{\partial}{\partial r} \left(r \frac{\partial \psi}{\partial r} \right) \right) dr + \int_0^{r_j} \left(2\phi r \psi (\phi^2 + \psi^2) \right) dr \\ & - \int_0^{r_j} \left(2\psi \frac{\partial}{\partial r} \left(r \frac{\partial \phi}{\partial r} \right) \right) dr - \int_0^{r_j} \left(2\psi r \phi (\phi^2 + \psi^2) \right) dr \Bigg), \end{aligned}$$

and this simplifies to

$$v = \frac{-1}{(\phi^2 + \psi^2)} \left(\int_0^{r_j} 2\phi \frac{\partial}{\partial r} \left(r \frac{\partial \psi}{\partial r} \right) dr - \int_0^{r_j} 2\psi \frac{\partial}{\partial r} \left(r \frac{\partial \phi}{\partial r} \right) dr \right).$$

Now we may integrate by parts as in Section (4.3.1), which will give

$$v = \frac{-1}{(\phi^2 + \psi^2)} \left[2r\psi \frac{\partial \phi}{\partial r} - 2r\phi \frac{\partial \psi}{\partial r} \right]_0^{r_j}. \tag{4.12}$$

The quotient rule applied to $\frac{\psi}{\phi}$ will give

$$\frac{\partial}{\partial r} \left(\frac{\psi}{\phi} \right) = \frac{\phi \frac{\partial \psi}{\partial r} - \psi \frac{\partial \phi}{\partial r}}{\phi^2}. \tag{4.13}$$

We are now able to substitute Equation (4.13) into Equation (4.12) to obtain

$$v = \frac{-1}{(\phi^2 + \psi^2)} \left[-2r\phi^2 \frac{\partial}{\partial r} \left(\frac{\psi}{\phi} \right) \right]_0^{r_j},$$

and the case where $r = 0$ returns only zero and so we now have an expression

for the nodal velocity which is

$$v = \frac{2r\phi^2}{(\phi^2+\psi^2)}\frac{\partial}{\partial r}\left(\frac{\psi}{\phi}\right)\Big|_{r_j}. \tag{4.14}$$

4.3.2 Calculating Conserved Quantities

In order to calculate the partial 'mass' values which we will hold constant, we begin by splitting our domain $r \in [0,1]$ into N equally sized intervals with nodes denoted by $r_j(t)$ for $j = 0, 1, ..., J$. As mentioned earlier, we aim to conserve the 'mass' in each interval which is given by

$$m_j = \int_{r_{j-1}(t)}^{r_{j+1}(t)} (\phi^2+\psi^2) r dr. \tag{4.15}$$

4.3.3 Generation of New Meshes

The generation of a new mesh relies on us being able to define a suitable velocity for each node. Once this velocity is known, we may use a simple explicit Euler time stepping method.

An analytic derivation of the formula used to give the velocity of each node is laid out in Section (4.3.1) and the velocities of the nodes are given by

$$v_j = \frac{2r_j\phi_j^2}{\phi_j^2+\psi_j^2}\frac{\partial}{\partial r}\left(\frac{\psi_j}{\phi_j}\right).$$

We then make the observation that

$$v_j = \frac{dr_j}{dt},$$

and this allows us to create a new grid using the standard explicit Euler time stepping method as given by the expression

$$r_j^{n+1} = r_j^n + \Delta t\, v_j,$$

where Δt represents the length of the time step. This new mesh may then be

used in the calculation of the values of ϕ and ψ at the next time level.

4.3.4 Recovering New Approximations of ϕ and ψ

New ϕ Approximations

In order to calculate a new approximation of the real part of our solution, ϕ, we begin by calculating

$$\theta_j^n = \int_{r_{j-1}(t)}^{r_{j+1}(t)} \phi r dr, \tag{4.16}$$

for each node. We then continue by examining how the value θ evolves over time and this is done by considering

$$\begin{aligned} \dot{\theta}_j^n &= \frac{d}{dt} \int_{r_{j-1}(t)}^{r_{j+1}(t)} \phi r dr \\ &= \int_{r_{j-1}(t)}^{r_{j+1}(t)} \frac{\partial \phi}{\partial t} r dr + [\phi r v]_{r_{j-1}(t)}^{r_{j+1}(t)}, \end{aligned}$$

into which we may substitute Equation (4.4).

Once each of these values has been calculated for each node, we may then use a simple explicit Euler time stepping method to calculate a value of θ at the next time level using

$$\theta_j^{n+1} = \theta_j^n + \Delta t \, \dot{\theta_j^n}.$$

Using Equation (4.16) we know that

$$\theta_j^{n+1} = \int_{r_{j-1}^{n+1}}^{r_{j+1}^{n+1}} \phi r dr,$$

to which we may apply the mid-point rule to obtain

$$\theta_j^{n+1} = \phi_j^{n+1} \left(\frac{(r_{j+1}^{n+1})^2 - (r_{j-1}^{n+1})^2}{2} \right). \tag{4.17}$$

Equation (4.17) may then be rearranged to give our approximation of the real

part of our solution, ϕ, at the next time level and this is given by

$$\phi_j^{n+1} = \frac{2\theta_j^{n+1}}{\left((r_{j+1}^{n+1})^2 - (r_{j-1}^{n+1})^2\right)}. \tag{4.18}$$

New ψ Approximations

We may now use a simple combination of Equations (4.15) and (4.18) to obtain an approximation for the imaginary part of the solution, ψ. This begins by applying the mid-point rule to Equation (4.15) to give

$$m_j = \int_{r_{j-1}^{n+1}}^{r_{j+1}^{n+1}} (\phi^2 + \psi^2) r dr = \left((\phi_j^{n+1})^2 + (\psi_j^{n+1})^2\right) \left(\frac{(r_{j+1}^{n+1})^2 - (r_{j-1}^{n+1})^2}{2} \right),$$

which may be rearranged to obtain

$$\psi_j^{n+1} = \sqrt{\frac{2m_j}{(r_{j+1}^{n+1})^2 - (r_{j-1}^{n+1})^2} - (\phi_j^{n+1})^2}.$$

This may be calculated since we may substitute Equation (4.18) in place of the ϕ^2 term. The value of m_j need not be recalculated at each time step since this quantity is time invariant as demonstrated analytically in Section (4.3.1).

This method may only be used to calculate values of ϕ and ψ for $j = 1, ..., J-1$, meaning we have a requirement for an alternative method to calculate approximations at the nodes r_0 and r_J. For the values of ϕ and ψ at r_J we use the boundary conditions to set

$$\phi_J = \psi_J = 0.$$

The approximations at the node $r_0 = 0$ must be extrapolated from the values at the other nodes and the method used to do this is laid out in Section (4.3.6).

4.3.5 Lagrange Polynomial Exrapolation

Since we are unable to produce a solution value at the point $r = 0$ using the method laid out above it is necessary to utilise some other method to generate a

solution at $r = 0$. The method which is used here is that of Lagrange polynomial extrapolation whereby we create a polynomial which runs through a set of points and then we are able to use this to extend our solution curve to the point $r = 0$.

In this case we use the three points next to the point $r = 0$ to create a quadratic polynomial which may be used to extend our solution curve.

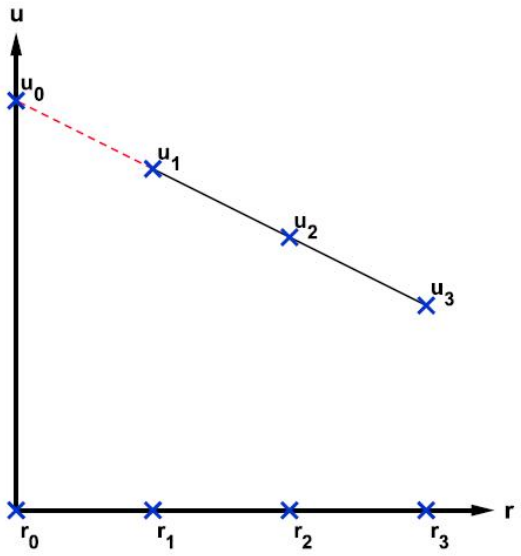


Figure 4.1: Demonstration of the method of Lagrange polynomials to extrapolate solutions

Figure (4.1) shows a simple example of this method, where the r_j are the radial coordinates and the u_j are the solution values. This example shows the case of a straight line solution however this method is equally valid when applied to curves such as those obtained in the solution of the NLS equation.

The formula which is used to create a quadratic running through the three points next to $r = 0$ and extrapolate our solution curve along to the point $r = 0$ is given by

$$\begin{aligned} u_0 \quad &\approx \quad u_1 \cdot \frac{r_0 - r_2}{r_1 - r_2} \cdot \frac{r_0 - r_3}{r_1 - r_3} \\ &+ \quad u_2 \cdot \frac{r_0 - r_1}{r_2 - r_1} \cdot \frac{r_0 - r_3}{r_2 - r_3} \\ &+ \quad u_3 \cdot \frac{r_0 - r_1}{r_3 - r_1} \cdot \frac{r_0 - r_2}{r_3 - r_2}, \end{aligned}$$

and this is applied at each time step to produce both a ϕ and ψ value at the point $r = 0$.

This method yields some reasonable results however it is the case that this method will violate one of the boundary conditions. The reason for this is that our solution will not necessarily have the property that $u_r(0,t) = 0$, and this leads us to seek a method of extrapolation which will ensure we do not violate the boundary conditions.

4.3.6 Improved Polynomial Exrapolation

In order to ensure we do not violate the boundary condition $u_r(0,t) = 0$, we begin by creating a quadratic polynomial of the form

$$u_j = ar_j^2 + br_j + c$$

and then use the solutions and the radial coordinates at two other points in order to specify the values of each coefficient.

Using the boundary condition $u_r(0,t) = 0$ gives

$$\frac{du_j}{dr_j} = 2ar_j + b = 0,$$

from which we may deduce that $b = 0$ leaving our quadratic polynomial as

$$u_j = ar_j^2 + c.$$

We now use the two closest values of u and r to assign values to the other coefficients which will be given by

$$a = \frac{u_1 - u_2}{r_1^2 - r_2^2}$$

and

$$c = u_2 - ar_2^2.$$

This now gives us a polynomial of the form given above which may be used to calculate the solution value at the point $r = 0$, and using this method clearly ensures that our solution will satisfy the boundary condition $u_r(0, t) = 0$ which takes effect at the point $r = 0$.

4.4 Results for Mass Conservative Method for Nonlinear Schrödinger Equation

In order to examine the results obtained for the above method whereby a mass monitor function was used, we begin by showing how the real and imaginary parts of the solution develop over time and then look at the evolution of the mesh over the same time period.

The example considered here uses a time step of $\Delta t = 1 \times 10^{-6}$, a mesh consisting of 11 nodes with an initial spacing of $\Delta r = 0.1$ and uses the initial condition $u(r, 0) = 20 \sin\left(\pi\left(\frac{1+r}{2}\right)\right)$ with boundary conditions as given in Section (4.2).

We begin by looking at the results obtained for the real part of the solution as displayed in Figure (4.2).

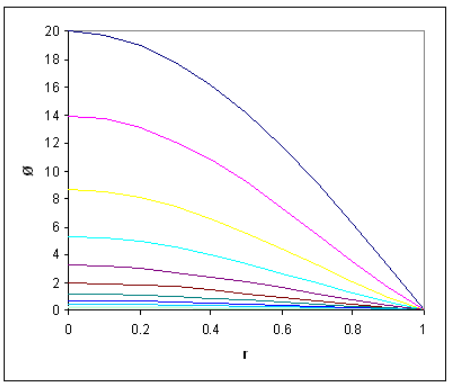


Figure 4.2: Real part of the 'mass' conservative solution of the Nonlinear Schrödinger Equation

Figure (4.2) clearly shows that using the method laid out in the previous sections results in the real part of the solution ϕ, decaying as time passes which is

most certainly not consistent with what we would expect to see from a blow-up problem.

The results which were obtained for the imaginary part of the solution are shown in Figure (4.3).

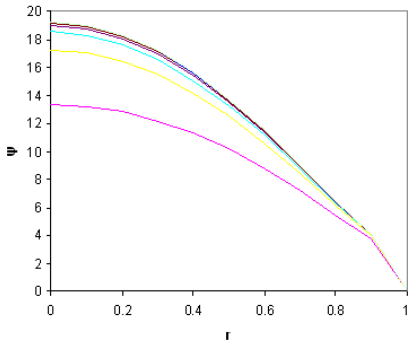


Figure 4.3: Imaginary part of the 'mass' conservative solution of the Nonlinear Schrödinger Equation

We can clearly see from Figure (4.3) that this method will result in the imaginary part of the solution, ψ, will increase over time. This does not necessarily go against what we should expect to see from a known blow-up problem.

We also examine the evolution of the mesh over time, and this is shown in Figure (4.4).

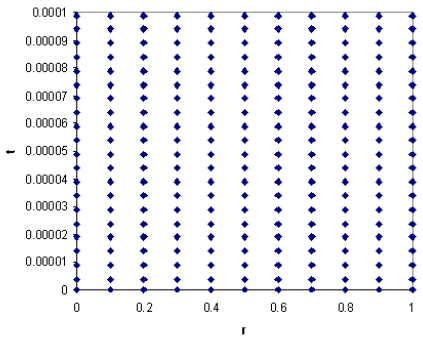


Figure 4.4: Mesh Evolution of the 'mass' conservative solution of the Nonlinear Schrödinger Equation

From Figure (4.4) we are able to see that the method laid out in the previous

sections will cause the nodes to move very slightly. This change is however likely to be too small to give much benefit in terms of accurate representation of the solutions obtained.

Consideration of all of the results obtained together would suggest that this method does not suitably model the behaviour which should be produced from the NLS equation. The most likely reason for the failure of this method is that whilst the 'mass' conservation property is necessary, it is not sufficient to capture the correct behaviour of our solution.

4.5 Area Conservative Method for Non-linear Schrödinger Equation

We now consider a velocity-based method whereby the velocity assigned to each node in order to ensure that the area under the solution curve held in each interval remains constant over time.

As explained previously, solutions of the NLS equation consist of a real part, ϕ and an imaginary part, ψ. With this in mind we define two separate meshes. One of these is for use in calculating ϕ solutions and the other for the ψ solutions and these shall be denoted r_ϕ and r_ψ respectively.

The method used here is very similar for calculations of both the real and imaginary parts of the solution. With this being the case we shall lay out the method used in detail for the imaginary part of the solution and merely state the most important equations associated with the real part of the solution.

4.5.1 Generation of Node Velocities

In this section the r which is given in any equation relates to either r_ϕ or r_ϕ as appropriate to the variable in question.

We begin by calculating the area under the graph of the imaginary part of

the solution for each interval as given by

$$a_{\psi j} = \int_{r_{j-1}(t)}^{r(t)} \psi(r,t) r dr. \tag{4.19}$$

The next step is to sum the $a_{\psi j}$ value from each interval to give the area under the solution curve for the entire domain, θ. This is given by

$$\theta(t) = \int_0^1 \psi(r,t) r dr. \tag{4.20}$$

Differentiation of Equation (4.20) with respect to time gives

$$\dot{\theta}(t) = \frac{d}{dt}\int_0^1 \psi r dr = \int_0^1 \psi_t r dr + [\psi r v]_0^1,$$

into which we may substitute Equation (4.5) in order to obtain

$$\dot{\theta} = \int_0^1 \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \phi}{\partial r}\right) r + \phi r(\phi^2 + \psi^2) dr + [\psi r v]_0^1,$$

and we may use integration by parts to give

$$\dot{\theta} = \left[r\frac{\partial \phi}{\partial r}\right]_0^1 + \int_0^1 \phi r\left(\phi^2 + \psi^2\right) dr + [\psi r v]_0^1.$$

Since this is a conservative method, we hold the fractional area under the solution curve constant in time for each interval, i.e.

$$\frac{1}{\theta}\int_{r_{j-1}(t)}^{r_j(t)} \psi(r,t) r dr = \text{constant}. \tag{4.21}$$

Differentiation of (4.21) with respect to time yields

$$\frac{d}{dt}\left[\frac{1}{\theta}\int_{r_{j-1}(t)}^{r_j(t)} \psi(r,t) r dr\right] = 0,$$

which, since we have a situation with differentiation under the integral, we may

use the Leibniz integral rule to obtain

$$0 = \frac{-1}{\theta^2}\dot{\theta}\int_{r_{j-1}(t)}^{r_j(t)} \psi(r,t) r dr + \frac{1}{\theta}\left[\int_{r_{j-1}(t)}^{r_j(t)} \frac{\partial \psi}{\partial t} r dr + \left[\psi r v_\psi\right]_{r_{j-1}(t)}^{r_j(t)}\right].$$

Using Equations (4.5) and (4.19), allows us to re-write this as

$$0 = \frac{-\dot{\theta}}{\theta} a_{\psi j} + \left[r\frac{\partial \phi}{\partial r}\right]_{r_{j-1}}^{r_j} + \int_{r_{j-1}}^{r_j} \phi r\left(\phi^2 + \psi^2\right) dr + \psi_j r_j v_{\psi j} - \psi_{j-1} r_{j-1} v_{\psi j-1},$$

where v_ψ represents the velocity of the mesh used in ψ calculations. Finally, rearranging this will give an expression for the velocity of each node in the mesh for the imaginary part, which is given by

$$v_{\psi j} = \frac{-1}{r_j \psi_j}\left[\frac{-\dot{\theta}}{\theta} a_{\psi j} + \left[r\frac{\partial \phi}{\partial r}\right]_{r_{j-1}}^{r_j} + \int_{r_{j-1}}^{r_j} \phi r\left(\phi^2 + \psi^2\right) dr - \psi_{j-1} r_{j-1} v_{\psi j-1}\right],$$

and this may be solved sequentially since we set $v_{\psi 0} = 0$, meaning that the velocity of each node is uniquely defined. At $r_N = 1$ (i.e. the right-hand boundary), v_ψ is not defined and so we choose $v_{\psi N=0}$.

In the case of the real part of the solution we consider the area under the real solution curve in each interval, as given by

$$a_{\phi j} = \int_{r_{j-1}}^{r_j} \phi(r,t) r dr.$$

As in the imaginary case we now sum these to obtain the area under the entire real solution curve which is given by

$$\chi(t) = \int_0^1 \phi(r,t) r dr.$$

We may then follow the same process as in the imaginary part of the solution, and this will give the velocity of nodes in the real mesh as

$$v_{\phi j} = \frac{-1}{r_j \phi_j}\left[\frac{-\dot{\chi}}{\chi} a_{\phi j} - \left[r\frac{\partial \psi}{\partial r}\right]_{r_{j-1}}^{r_j} - \int_{r_{j-1}}^{r_j} \psi r\left(\phi^2 + \psi^2\right) dr - \phi_{j-1} r_{j-1} v_{\phi j-1}\right].$$

4.5.2 Generating New Meshes and Total Areas

The generation of new meshes and total areas under the solution curve for both the real and imaginary parts of the solution is carried out using a simple explicit Euler time stepping method.

Since the velocities of the nodes are

$$v_{\phi j} = \frac{dr_{\phi j}}{dt} \quad \text{and} \quad v_{\psi j} = \frac{dr_{\psi j}}{dt},$$

we may generate new meshes for the real and imaginary parts of the solution using the expressions

$$r_{\phi j}^{n+1} = r_{\phi j}^{n} + \Delta t \; v_{\phi j}^{n},$$

and

$$r_{\psi j}^{n+1} = r_{\psi j}^{n} + \Delta t \; v_{\psi j}^{n}.$$

It is then necessary to generate the new areas under the solution curves (θ and χ) in the same way with

$$\chi_{j}^{n+1} = \chi_{j}^{n} + \Delta t \; \dot{\chi}_{j}^{n},$$

and

$$\theta_{j}^{n+1} = \theta_{j}^{n} + \Delta t \; \dot{\theta}_{j}^{n}.$$

4.5.3 Recovery of New Approximations

We have already mentioned that we hold the expression

$$\frac{1}{\theta} \int_{r_{j-1}(t)}^{r_{j+1}(t)} \psi(r,t) r dr,$$

to be constant in time. This means we may deduce that

$$\frac{1}{\theta(t)} \int_{r_{j-1}(t)}^{r_{j+1}(t)} \psi(r,t) r dr = \frac{1}{\theta(0)} \int_{r_{j-1}(0)}^{r_{j+1}(0)} \psi(r,0) r dr,$$

and the mid-point rule my be applied to this in order to obtain

$$\frac{1}{\theta(t)}\psi_j(t)\frac{\left[r_{j+1}^2(t)-r_{j-1}^2(t)\right]}{2}=\frac{1}{\theta(0)}\psi_j(0)\frac{\left[r_{j+1}^2(0)-r_{j-1}^2(0)\right]}{2}.$$

This may be rearranged to obtain

$$\psi_j(t)=\psi_j(0)\frac{\theta(t)\left[r_{j+1}^2(0)-r_{j-1}^2(0)\right]}{\theta(0)\left[r_{j+1}^2(t)-r_{j-1}^2(t)\right]},$$

and this is used to recover a new approximation of the imaginary part of the solution.

Utilising the same method applied to

$$\frac{1}{\chi}\int_{r_{j-1}(t)}^{r_{j+1}(t)}\phi(r,t)rdr,$$

will give the expression used to recover the real part of the solution which is

$$\phi_j(t)=\phi_j(0)\frac{\chi(t)\left[r_{j+1}^2(0)-r_{j-1}^2(0)\right]}{\chi(0)\left[r_{j+1}^2(t)-r_{j-1}^2(t)\right]}.$$

4.6 Results for the Area Conservation Method for the Nonlinear Schrödinger Equation

In examining the results obtained for the area conservative method applied to the NLS equation, we begin by looking at the velocities of the nodes in both the ϕ and ψ meshes. Based on the theory of this type of problem we should expect all of the nodes with the ability to move, to have a velocity such that they will move towards the blow-up point located at $r=0$. This should be the case especially for the part of the solution in which the blow-up occurs.

The examples considered here use a time step of $\Delta t=1\times10^{-6}$, a mesh made up of 11 nodes with initially equal spacings and utilise the boundary conditions given in Section (4.2).

We begin by examining the nodal velocities obtained when using an initial

condition of the form

$$6\sqrt{2}e^{-r^2},$$

which is considered on the domain $r \in [0, 5]$. The nodal velocities obtained for the first time step are shown in Figure (4.5).

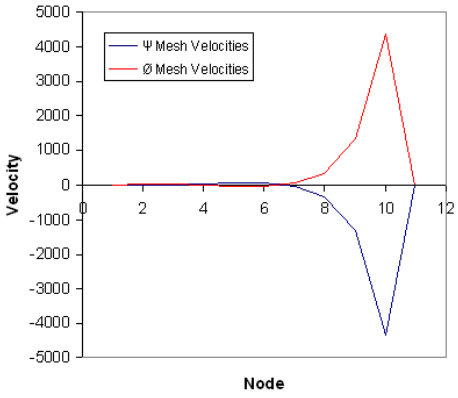


Figure 4.5: Nodal velocities obtained for the NLS equation with an initial condition of $6\sqrt{2}e^{-r^2}$

Figure (4.5) clearly shows that some of the nodes in each mesh will have a velocity such that they move to the left and some will have velocities such that they move to the right. Movements of this type will obviously not aid in accurately resolving the detail of the blow-up in the solution as time passes.

Another initial condition was considered in order to test whether the unusual velocities obtained for the other initial condition were some type of anomaly. The condition that was next to be considered was of the form

$$20 \sin\left(\pi\left(\frac{1+r}{2}\right)\right),$$

and this was considered on the domain $r \in [0, 1]$. As before we present a plot of the nodal velocities obtained for the first time step of this method in Figure (4.6).

We are clearly able to see that, as was the case for the other initial condition, some of the nodes will move to the left and some will move to the right.

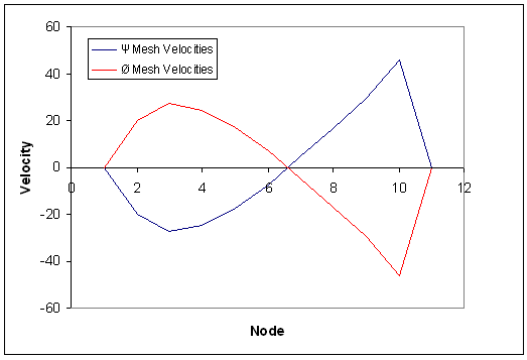


Figure 4.6: Nodal velocities obtained for the NLS equation with an initial condition of $20\sin\left(\pi\left(\frac{1+r}{2}\right)\right)$

Again, we are able to say that mesh evolution of this form will not be of use in attempting to aid the resolution of the blow-up in the solution of the NLS equation.

The consideration of both of these initial conditions and the velocities obtained for the nodes within each mesh, allows us to conclude that this method is not suitable for use in the solution of the NLS equation.

4.7 Nonlinear Schrödinger Equation Summary

Within this chapter there has been two methods set out to solve the NLS equation on a moving mesh.

The first method was based on the use of a 'mass' monitor function and this resulted in a mesh which did not evolve in a suitable manner to help resolve the blow-up that our solution should exhibit.

We then went on to consider a method based around the use of an area monitor function. This method did create a mesh within which the nodes moved, however the direction of nodal movements was not in a manner which we may deem as being suitable for helping to resolve the blow-up in the solution of the NLS equation.

Chapter 5

Discussion of Project

5.1 Summary

In this dissertation we have carried out an investigation into the application of velocity-based moving mesh methods based on various monitor functions. The methods investigated in this project were applied to a number of semi-linear time dependant partial differential equations.

Chapter 1 began by explaining the motivation for this investigation into velocity-based moving mesh methods as well as the types of problems to which these methods were applied. In Chapter 2 we demonstrated the effectiveness of a method based upon an area conservation monitor function for two different powers of blow-up term. Some other 'Fisher type' equations, namely the 'traditional' Fisher's equation and the Cahn-Allen equation were investigated in Chapter 3. The method applied to these equations was built upon a hybrid of an arc-length monitor function and an area monitor function. Finally, Chapter 4 investigated two separate approaches to solving the Nonlinear Schrödinger equation. The first of these was based on a 'mass' monitor function and the other built upon an area monitor function.

5.2 Conclusions

Throughout this dissertation we have outlined a number of different methods which have been applied to a variety of semi-linear PDEs and have achieved varying degrees of success.

The method which was applied to Fisher's equation, which was based on an area monitor function generated very positive results. This chapter demonstrated the ability of the method to produce results with a good level of resolution in the blow-up of the solution whilst using a small number of nodes. It is clear from the results obtained that using an area monitor function allowed the mesh to effectively cluster nodes around the blow-up point.

A method based upon a hybrid of an arc-length monitor function and an area monitor function was applied to the 'traditional' Fisher's and Cahn-Allen equations. This method displayed some limited potential in terms of the movement of nodes towards the areas of interest, however, there were issues surrounding the crossing of nodes relatively early on in the mesh evolution. The potential of this method was particularly evident in the solution of the 'traditional' Fisher's equation where we obtained the desired interface type of solution which we would expect to see in a phase-field problem such as this. An issue of unphysical behaviour appeared in the solution of the Cahn-Allen equation which would suggest that an alternative method should be sought for the solution of this equation.

Two separate methods were considered for the Nonlinear Schrödinger equation. The first of these methods was based upon a mass monitor function was not successful due to a lack of nodal movement. Another method was then proposed and this was based on an area monitor function, which created very unpredictable nodal velocities and certainly did not cluster nodes towards the blow-up point. The failure of these two methods may suggest that velocity-based moving mesh methods should not be considered as an appropriate method for the solution of this problem.

In general it is clear from the work carried out in this dissertation that these velocity-based moving mesh methods have some potential in terms of reallocating nodes in such a way that the resolution in certain areas of the solution is increased. This also aids with the efficiency of the numerical methods used to solve these problems since we may accurately represent the solution with fewer nodes than would be necessary when using a standard fixed mesh method. It is also clear that the choice of an appropriate monitor function is essential for these methods to be successful. The final conclusion to draw from this dissertation is that whilst we have demonstrated the potential of velocity-based moving mesh methods, they are not always successful and as such they should be considered as one possible solution method but not always the most effective one.

5.3 Further Work

As a result of this investigation into velocity-based moving mesh methods, there are a number of areas into which we may direct future research.

The first area into which future research may be carried out is relevant to each of the problems considered in this dissertation. This is that a study into the effects of using different monitor functions may help to improve the results obtained. In relation to Fisher's equation (Chapter 2) and the 'traditional' Fisher's equation (Chapter 3), this would simply be a question of whether or not alternative monitor functions would improve upon the resolution of the solutions obtained here. When relating this research area to the Nonlinear Schrödinger equation and the Cahn-Allen equation, a study of alternative monitor functions may lead to the discovery of a monitor which gives the behaviour we would expect to see from these types of problems.

We may also choose to delve deeper into the computational efficiency of the methods used in this dissertation. An obvious starting point for this would be to examine Fisher's equation due to the relative success of the method consid-

ered. There are two main ways in which we may seek to improve our method's efficiency. Firstly, we may look at the numerical code used to obtain the results in order to ensure it holds a minimum amount of information at any time. We may also wish to ensure that all calculations are carried out using the smallest number of operations possible. The other way in which the computational efficiency of our method may be improved would be to look into alternative time stepping techniques. This could potentially allow us to use fewer time steps in order to reach our final solution and this could significantly reduce the amount of operations necessary to obtain final results.

The investigation into the 'traditional' Fisher's and Cahn-Allen equations was severely limited by the vastly different rates at which adjacent nodes moved as this caused nodes to overtake one another. This leads to the possibility of an investigation into methods which may be used to prevent nodes from overtaking each other. This represents a particularly interesting future research area as there are a great number of possible routes which the investigation may follow.

Finally, we may seek to utilise one of many more sophisticated methods which may be able to be applied to a wider array of different types of problems. This research area covers a wide array of different methods however it would require a deviation from velocity-based moving mesh methods and so it shall not be discussed here.

Bibliography

[1] C. Budd, W. Huang, and R. Russell, "Adaptivity with moving grids," *Acta Numerica*, 2009.

[2] P. Westwood, "A moving mesh finite element approach for the cahn-hilliard equation," Master's thesis, University of Reading, Department of Mathematics, 2010.

[3] C. Budd, J. Chen, W. Huang, and R. Russell, "Moving mesh methods with applications to blow-up problems for pdes," *Numerical Analysis*, 1995.

[4] C. Budd, W. Huang, and R. Russell, "Moving mesh methods for problems with blow-up," *SIAM Journal of Scientific Computing*, 1996.

[5] J. Zhang and Q. Du, "Numerical studies of discrete approximations to the allen-cahn equation in the sharp interface limit," *SIAM Journal of Scientific Computing*, 2009.

[6] S. Braun and A. Kluwick, "Blow-up and control of marginally separated boundary layers," *Philosophical Transactions of The Royal Society A: Mathematical, Physical and Engineering Sciences*, 2005.

[7] A. Kluwick, S. Braun, and E. Cox, "Near critical phenomena in laminar boundary layers," *Journal of Fluids and Structures*, 2008.

[8] S. Cole, "Blow-up in a chemotaxis model using a moving mesh method," Master's thesis, University of Reading, Department of Mathematics, 2009.

[9] J. Ockendon, S. Howison, A. Lacey, and A. Movchan, *Applied partial differential equations*. Oxford University Press, 1999.

[10] Y. Qiu and D. Sloan, "Numerical solution of fisher's equation using a moving mesh method," *Journal of Computational Physics*, 1998.

[11] W. Lyons, H. Ceniceros, S. Chandresekaran, and M. Gu, "Fast algorithms for spectral collocation with non-periodic boundary conditions," *Journal of Computational Physics*, 2004.

[12] C. Budd, S. Chen, and R. Russell, "New self-similar solutions of the nonlinear schrodinger equation with moving mesh computations," *Journal of Computational Physics*, 1999.

[13] C. Budd and V. Dorodnitsyn, "Symmetry-adapted moving mesh schemes for the nonlinear schrodinger equation," *Journal of Physics A: Mathematical and General*, 2001.

[14] A. Twigger, "Blow-up in the nonlinear schrödinger equation using an adaptive mesh method," Master's thesis, University of Reading, Department of Mathematics, 2008.