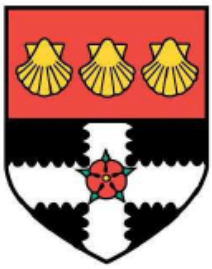
Blow-up in the Nonlinear Schrödinger Equation Using an Adaptive Mesh Method



UNIVERSITY OF READING
Department of Mathematics

Ashley Twigger

Supervisor: Professor Michael J. Baines

Contents

Abstract

In this dissertation we give a brief overview of moving mesh methods, including one based upon moving mesh PDEs and one based on relative conservation. Then we describe the blow-up problems that we are interested in applying the relative conservation method to. Later on we analyse our results comparing them to existing results.

Acknowledgements

Firstly I would like to thank the entire department for being so supportive and understanding. In particular my supervisor Mike Baines who has gone above and beyond what I would expect of a supervisor, and Sue Davis who throughout the year has been there when I needed her. I would also like to thank the EPSRC for allowing me the opportunity to be on this course, without the funding that they provided it would not of been possible for me to further my education.

Declaration

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

Ashley Twigger

Chapter 1

Introduction

Blow-up has increasingly become a major phenomena in the evolution of nonlinear equations. Physical problems when modelled may develop singularities in a finite amount of time T ($T < \infty$). Combustion in chemicals, chemotaxis in cellular aggregates, or the formation of shocks in the inviscid Burgers equation are examples of blow-up in the solution of a model [3]. These singularities can represent a change in the properties of the model such as ignition of a heated gas mixture.

A class of problems that displays this feature is the semilinear parabolic equations. These are used in the description of blow-up in the temperature of a reacting medium such as burning gas. These have the form

$$u_t = u_{xx} + f(u) \tag{1.1}$$

with boundary conditions

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

and initial conditions

$$u(x,0) = u_0(x)$$

where $f(u)$ is any convex function of u. As this is the part of the equation that is blowing up we require $f(u) \to \infty$ as $u \to \infty$. The restrictions of the initial condition are that it must be sufficiently large and have a single nondegenerate maximum [3].

1.1 The structure of blow-up solutions for PDEs

First we give an overview of the problem followed by an example of scaling.

1.1.1 Overview

Let x^* denote the single blow-up point, such that

$$u(x^*,t) \to \infty \qquad \text{as} \qquad t \to T \tag{1.2}$$

and

$$u(x,t) \to u(x,T) < \infty \qquad \text{if} \qquad x \neq x^* \tag{1.3}$$

This just means that as we approach the blow-up time T the only point that approaches infinity is the blow-up point x^*. Around x^* an isolated peak develops with the width tending towards zero as time approaches T. To compute a solution which represents the true solution accurately, an adap-

tive numerical method must be used that evolves the spatial mesh as the singularity develops. The singularity develops in a fairly simple manner, often independent of local structures in the initial conditions.It is conjectured in [5] that the growth of $u(x,t)$ near the blow-up time T is described by

$$\max|u(x,t)| \propto (T-t)^{-\alpha} \qquad \alpha > 0$$

We will be looking at the equation when $f(u) = u^p$ $(p > 1)$ which is Fisher's equation and $f(u) = e^u$ which is the Kassoy problem in addition to the nonlinear Schrödinger equation which we will discuss later. These problems are great for testing out numerical methods as the formation of the singularities is typical of a wide range of PDEs (partial differential equations). Also, a lot is known about the underlying analytic structure of the solutions for t close to T and x close to x^*. Thus they make excellent problems for testing performance and accuracy. If the numerical method faithfully follows the underlying asymptotic structure we can assume that it does the same for more complicated problems where we do not know the underlying structure.

Most of the work that precedes [3] is based on either clearly knowing the analytic structure of the singularity and exploiting it or using h-refinement based methods that increase the number of mesh points as $t \to T$. Contrary to this both Budd's and our methods do not rely on a priori knowledge of the solution or additional mesh points. This is achieved by using a monitor function $M(u)$ which gives information on how the mesh points should be distributed through space and time.

Analysing the scaling properties of the PDE leads to an optimal choice for M. A full proof of the asymptotic scaling of the singularity can be found in [1].

Let us consider the solutions of the Fisher's equation

$$u_t = u_{xx} + u^p, \qquad p > 1 \tag{1.4}$$

with

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

$$u(x,0) = u_0(x) > 0$$

If $u_0(x)$ is sufficiently large and has a single nondegenerate maximum then (1.2) and (1.3) hold. The point x^* at which the isolated spike occurs and the blow-up time T depend on $u_0(x)$, but the development of the blow-up itself is almost independent of the initial conditions, provided x and t are close to x^* and T respectively [2].

1.1.2 Scaling

We now consider scaling properties of (1.4).

Let

$$\begin{aligned} t &= \lambda t' \\ x &= \lambda^\theta x' \\ u &= \lambda^\beta u' \end{aligned}$$

Substituting into the left hand side of (1.4)

$$u_t = \frac{\partial u}{\partial t} = \frac{\partial \left(\lambda^\beta u'\right)}{\partial \left(\lambda t'\right)} = \lambda^{\beta-1}\frac{\partial u'}{\partial t'}$$

and the right hand side

$$u_{xx} = \frac{\partial^2 u}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right) = \frac{\partial}{\partial x}\left(\lambda^{\beta-\theta}\frac{\partial u'}{\partial x'}\right) = \lambda^{\beta-2\theta}\frac{\partial^2 u'}{\partial x'^2}$$

$$u^p = \lambda^{\beta p} u'^p$$

Changing variables the Fisher's equation becomes

$$\lambda^{\beta-1}u'_{t'} = \lambda^{\beta-2\theta}u'_{x'x'} + \lambda^{\beta p}u'^p$$

For natural scaling invariance of the PDE we need

$$\beta - 1 = \beta - 2\theta = \beta p$$

which implies $\theta = \frac{1}{2}$ and $\beta = -\frac{1}{p-1}$, and so a natural scaling of the solutions of (1.4) is

$$\begin{cases} (T-t) \to \lambda(T-t) \\ x \to \lambda^{\frac{1}{2}}x \\ u \to \lambda^{-\beta} \end{cases} , \qquad \lambda > 0 \tag{1.5}$$

A self-similar solution of (1.4) is any solution of the form

$$\frac{u}{(T-t)^{-\beta}} = f\left(\frac{x}{(T-t)^{-\frac{1}{2}}}\right)$$

which is invariant under this scaling. Such scaling invariance and corresponding self-similar solutions can be found in various equations describing blow-up, with very similar scalings used for the Kassoy problem and nonlinear Schroödinger equation.

Here are a few that we will be looking at in this dissertation.

Fisher's equation

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

The Kassoy problem

$$u_t = u_{xx} + e^u$$

The nonlinear Schrödinger equation

$$i\psi_t + \nabla^2\psi + |\psi|^{2\sigma}\,\psi = 0$$

The first two are semilinear parabolic equations and the final one is a hyperbolic PDE.

Chapter 2

Moving grids

When a singularity forms it gains height and loses width at increasingly smaller time scales when approaching time T. This isolated spike could be missed by a fixed mesh method over time, as the spike could fall between mesh points. So an adaptive mesh method should be used to overcome this. There are three main types of adaptivity:

(1) h-refinement is static and refines the mesh by adding nodes to make the mesh finer in places shown in figure 2.1, but practically this is not a viable method to use in this case as it becomes more and more computationally expensive as the problem develops and the singularity loses width.

(2) p-refinement is also static and uses higher order polynomials to represent the solution more accurately (figure 2.2). It has high rates of convergence and accuracy compared to h-refinement but a polynomial

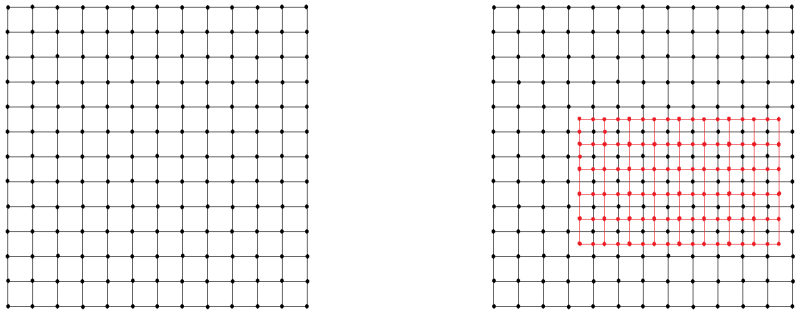


Figure 2.1: Left: original mesh. Right: post h-refinement, refined edges in red.

will never be able to fully model the blow-up if it falls between nodes.

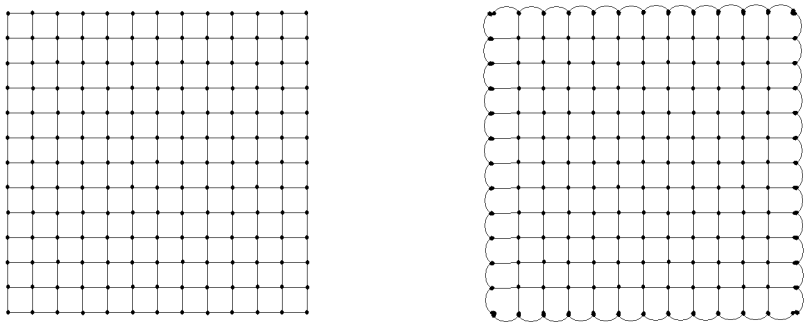


Figure 2.2: A sketch of p-refinement where higher order polynomials are used between nodes.

(3) The methods that both ourselves and Budd use are based on r-refinement. These are moving mesh methods that uses a fixed number of nodes and

redistribute them to keep track of main features according to a certain criteria which is set (figure 2.3). This has the advantage that it can keep track of the singularity right up to blow-up time T without being expensive to compute, but it has the drawback that away from the blow-up point the solution can be poorly resolved due to few nodes remaining close.

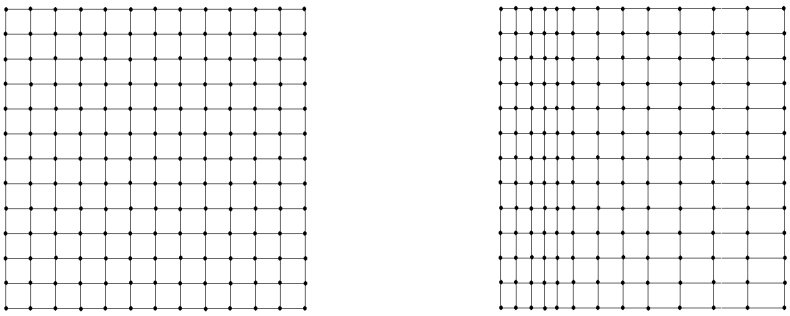


Figure 2.3: r-refinement: the nodes are redistributed give finer resolution in places.

Now we describe the r-refinement based moving mesh methods.

2.1 Moving mesh methods

2.1.1 Moving mesh PDEs

To solve PDEs such as (1.1) Budd proposes in [3] that by using a moving mesh PDE method in which $u(x,t)$ is discretised in the spatial variable to give the solution $u_i(t)$ on a moving mesh $x_i(t)$, $i = 0, ..., N$. The boundary conditions of (1.1) dictate that $u_0(t) = u_N(t) = 0$, $x_0 = 0$ and $x_N = 1$. The mesh $x_i(t)$ is defined by the mesh transformation

$$x(\xi, t) : [0, 1] \rightarrow [0, 1],$$

where x is the physical coordinate and ξ is the computational coordinates.

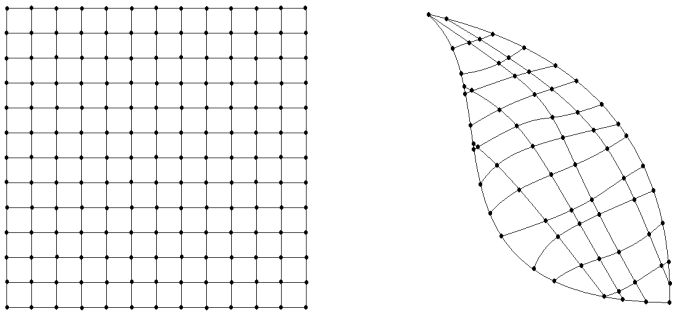


Figure 2.4: Sketch of transformation from computational space (left) to physical space (right)

As can be seen in figure 2.4 a simple mesh in computational space can be

used to describe a complicated physical space.

$$x_i(t) = x(\xi, t) = x\left(\frac{i}{N}, t\right). \tag{2.1}$$

The constraint $\frac{\partial x}{\partial \xi} > 0$ ensures that mesh crossing does not occur.

The moving mesh PDE (MMPDE) approach [2] requires a new PDE to solve $x(\xi, t)$ known as the moving mesh PDE which is solved simultaneously with the original PDE to find $u(x, t)$.

The process used to determine $x(\xi, t)$ is the equidistribution of a positive monitor function $M(u)$. The equidistribution principle takes some measure of something such as error, density or a function and places the nodes of a mesh so that the contributions between the nodes are distributed equally to give a smooth solution.

$$\int_0^{x_i(t)} M \ \mathrm{dx} = \frac{i}{N} \int_0^1 M \ \mathrm{dx} = \xi \int_0^1 M \ \mathrm{dx} \tag{2.2}$$

since M is distributed equally between nodes (2.2) holds. Differentiating (2.2) gives

$$\frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) = 0, \qquad x(0, t) = 0, \qquad x(1, t) = 1 \tag{2.3}$$

If (2.3) holds then a coordinate transformation is said to be equidistributed. Budd et al [2] found that for his method it is more convenient not to strictly enforce equidistribution but instead to solve for an MMPDE which tends towards an equidistributed solution. That way he states a simple initial mesh can be used, such as a uniform one. Also the process produces a more stable mesh with less chance of mesh crossing then if (2.2) were strictly enforced,

and a smoothing approach was used.

Out of the various MMPDEs proposed in [4], Budd uses the two labelled MMPDE4 and MMPDE6, which are respectively

$$\tau \frac{\partial}{\partial \xi}\left(M \frac{\partial \dot{x}}{\partial \xi}\right) = -\frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) \tag{2.4}$$

and

$$\tau \frac{\partial^2 \dot{x}}{\partial \xi^2} = -\frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) \tag{2.5}$$

where $\dot{x}$ denotes $\frac{\partial x}{\partial t}\big|_{\xi}$, and τ is a small parameter used to relax the mesh to increase resolution away from the blow-up.

τ tends to zero on the left hand side of both (2.4) and (2.5) as t tends to T therefore the MMPDEs head towards an equidistributed state (2.3). This relaxes the need to enforce an exact equidistribution at the start allowing the use of a simple initial mesh such as a uniform one. Also this relaxation increases resolution further away from blow-up giving a better approximarion to the exact solution in the region.

If the MMPDE method is used the MMPDE must be invariant under the scaling (1.5), which can be achieved by using a suitable parameter τ and monitor function $M(u)$ [2]. For Fisher's equation the MMPDEs remain invariant under the monitor function $M(u) = u^{p-1}$.

For the numerical computation the PDE (1.4) is transformed in terms of the computational coordinate ξ using the chain rule

$$u_t = \dot{u} + u_x \dot{x} \qquad u_{x_i} = u_x x_i$$

and discretised by a central finite difference into the quasi-Lagrangian form

$$\begin{cases} \dot{u} - \frac{u_\xi}{x_\xi}\dot{x} = \frac{1}{x_\xi}\left(\frac{u_\xi}{x_\xi}\right)_\xi + u^p \\ u(0,t) = u(1,t) = 0 \end{cases} \tag{2.6}$$

Discretising this equation in the spatial variable whilst remaining continuous in time gives

$$\dot{u_i} - \frac{u_{i+1} - u_{i-1}}{x_{i+1} - x_{i-1}}\dot{x_i} = \frac{2}{x_{i+1} - x_{i-1}}\left(\frac{u_{i+1} - u_i}{x_{i+1} - x_i} - \frac{u_i - u_{i-1}}{x_i - x_{i-1}}\right) + u_i^p \tag{2.7}$$

for $i = 1, ..., N-1$ which are the interior points, since the exterior points are fixed ($u_0 = u_N = 0$ for all t). MMPDE4 and MMPDE6 can be discretised in a similar fashion.

To obtain a reasonably accurate solution without oscillatory behaviour it is often necessary to use some sort of smoothing of the mesh. Using a smoothed monitor function $\tilde{M}$ Budd's final forms of the discretised MMPDEs are

$$\tau\left(\tilde{M}_{i+\frac{1}{2}}\left(\dot{x}_{i+1} - \dot{x}_i\right) - \tilde{M}_{i-\frac{1}{2}}\left(\dot{x}_i - \dot{x}_{i-1}\right)\right) = -\left(\tilde{M}_{i+\frac{1}{2}}\left(x_{i+1} - x_i\right) - \tilde{M}_{i-\frac{1}{2}}\left(x_i - x_{i-1}\right)\right) \tag{2.8}$$

for MMPDE4 and

$$\tau\left(\dot{x}_{i+1} - 2\dot{x}_i + \dot{x}_{i-1}\right) = -\left(\tilde{M}_{i+\frac{1}{2}}\left(x_{i+1} - x_i\right) - \tilde{M}_{i-\frac{1}{2}}\left(x_i - x_{i-1}\right)\right) \qquad (2.9)$$

for MMPDE6. $i = 1, ..., N-1$ with $x_0 = 0$ and $x_N = 1$, where $\tilde{M}_{i+\frac{1}{2}} := \frac{1}{2}\left(\tilde{M}_i + \tilde{M}_{i+1}\right)$. Note that the smoothing of the monitor function maintains dimension which is important when rescaling of the equations is considered.

This method has been used on blow-up problems successfully. We now describe a different approach.

2.1.2 Conservative method

The way our numerical method differs to Budd's is that instead of moving into a computational space to solve the PDE and then differentiating with respect to time to get $\dot{x}$ we remain in the physical space and compute $\dot{x}$ directly in terms of x. Also, Budd uses a monitor function to equidistribute the error of some property (e.g. density) whereas we are using a monitor function to maintain any distribution of the nodes so that for example any initial "mass" in a cell remains constant relative to the total mass.

$$\frac{\text{Mass in cell}}{\text{Total mass}} = \text{constant} \qquad \forall t < T$$

this gives each node a velocity so we know where it will be at the end of each

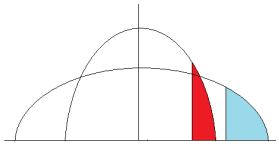


Figure 2.5: Sketch showing the same cell represented in blue then red and containing the same relative "mass" in both as the problem has evolved

time step.

Now we look at the general outline to the method that has been used.

Method

The conservative method begins with an initialisation process. Δt is the time step used throughout the computations and if fixed then it is chosen here, whereas if a variable Δt is being used then the constants Δs which is a small scalar, and T an estimate to the blow-up time are chosen now, later determining Δt at each time step.

$$x_i(0) = \frac{i}{N}L \qquad \text{for } i = 0, ..., N, \qquad L > 0.$$

This creates a uniformed mesh of $N + 1$ nodes in the region $[0, L]$. We then apply the initial condition

$$u(x,0) = f(x_i) \qquad \text{for } i = 0, ..., N$$

to the mesh points. If the mass of the region changes during evolution then we define θ in terms of the monitor function $M(u)$ as follows.

$$\theta = \int_{x_0}^{x_N} M(u)\ \mathrm{d}x.$$

This is a normalising variable used in the next part of the initialisation process

$$c_i = \frac{1}{\theta} \int_{x_0}^{x_i} M(u)\ \mathrm{d}x \qquad \text{for } i = 1, ..., N-1, \tag{2.10}$$

which remain constant for all t due to the $\frac{1}{\theta}$ in front of the integral.

This is the beginning of the loop, where the method starts in earnest. It determines the velocities of θ and each mesh point individually. From equation (2.10) $\dot{\theta}$ is given by

$$\dot{\theta} = \int_{x_0}^{x_N} \frac{\partial M(u)}{\partial t}\ \mathrm{d}x$$

also from (2.10) using Leibnitz' rule

$$\dot{x}_i = \frac{1}{u_i} \left[- \int_{x_0}^{x_i} \frac{\partial M(u)}{\partial t}\ \mathrm{d}x + c_i \dot{\theta} \right] \qquad \text{for } i = 1, ..., N-1. \tag{2.11}$$

The singularity at x_0 is an attractor, all the mesh points x_i that are not fixed

have a negative velocity heading towards x_0 at a quicker rate the closer they are. This monotonic decrease in velocity ($\dot{x}_{i+1} < \dot{x}_i$) insures that no node crossing occurs during the evolution of the problem. By using an Euler time stepping equation it is then possible to approximate θ and the mesh points at the next time step by

$$\theta(t + \Delta t) = \theta(t) + \Delta t\dot{\theta}$$

and

$$x_i(t + \Delta t) = x_i(t) + \Delta t\dot{x}_i \qquad \text{for } i = 1, ..., N - 1.$$

With the mesh points redistributed we can approximate $u(x, t)$ using our numerical method using the following equations.

$$u_0 = \frac{2\theta}{x_1}$$

$$u_i = \frac{2\theta}{x_{i+1} - x_{i-1}} (c_{i+1} - c_{i-1}) \qquad \text{for } i = 1, ..., N - 1.$$

Once we have reached this point we begin again at the beginning of the loop and approximate for the next time level until the blow-up time T is reached.

Chapter 3

Numerical Results

To obtain the results shown in this dissertation we used C++ to write a program which outputs to Matlab, allowing us to look at the results graphically and quickly be able to analyse them qualitatively.

3.1 Fisher's equation

To investigate whether our numerical method returns meaningful results we first look at the Fisher's equation for $p = 2$ for which there are relatively large amounts of text about the underlying asymptotic structure of the blow-up. In particular the equation appears in the Budd papers referenced in this dissertation.

Unlike Budd in [2] we only use the right side of the domain as the problem is symmetric about the centre. In doing this we have to slightly change the boundary conditions from

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

to

$$u_x(x,t) = 0 \text{ at } x = 0 \text{ and } u(x,t) = 0 \text{ at } x = 0.5.$$

In addition the initial condition has to be modified so that it is translated left from

$$u(x,0) = 20\sin(\pi x) \tag{3.1}$$

to

$$u(x,0) = 20\sin\left[\pi\left(x + \frac{1}{2}\right)\right] \tag{3.2}$$

For the equation we have used the monitor function

$$M(u) = u^{p-1} \tag{3.3}$$

which remains invariant under evolution, thus the rescaling (1.5) holds for all time before the blow-up time T.

As can be seen in figure 3.1 for the initial condition (3.2) the solution shows convergence, as the number of nodes is increased and the fixed time step is reduced the blow-up time approaches a time of $T \approx 0.08244$ which is close

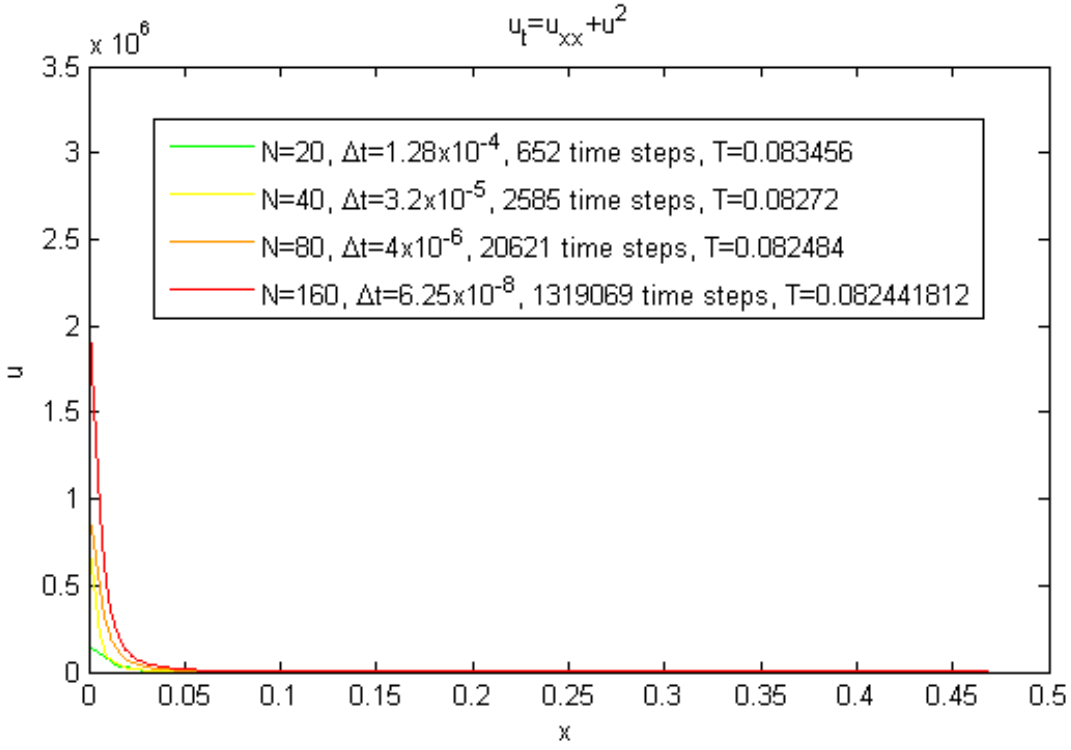


Figure 3.1: Convergence of the solution for increasing numbers of nodes.

to the blow-up time $T = 0.08237$ that Budd found in [2] with his method.

Since we know that the blow-up occurs at u_0 we can divide the entire region by this to get a solution that remains between zero and one for all time. Figure 3.2 shows the normalised evolution from the initial state for a few time steps close to blow-up. The normalised solution is converging towards a delta function this shows that it is only u_0 that is blowing up creating an isolated spike.

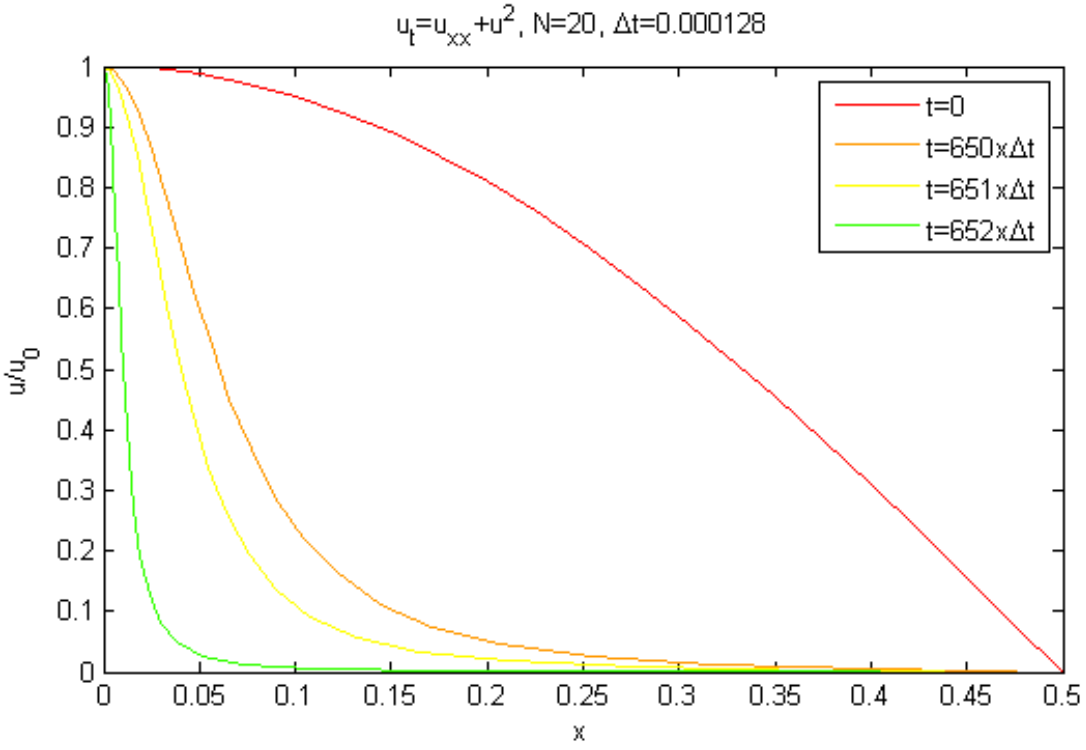


Figure 3.2: The scaled solution is approaching a delta function.

Again using the monitor function (3.3) we now look at the Fisher's equation for $p = 3$. As it is the u^p term that is blowing-up in the problem it is safe to assume that the blow-up time T will occur sooner then for the $p = 2$ case.

From our program blow-up time appears to occur at $T \approx 0.0012836$ which agrees with our assumption that blow-up will occur sooner then for $p = 2$. Even though the curve reaches a lower height then for the $p = 2$ case, there is a steeper incline to the spike which if normalised would be even closer to the delta function then the first case.

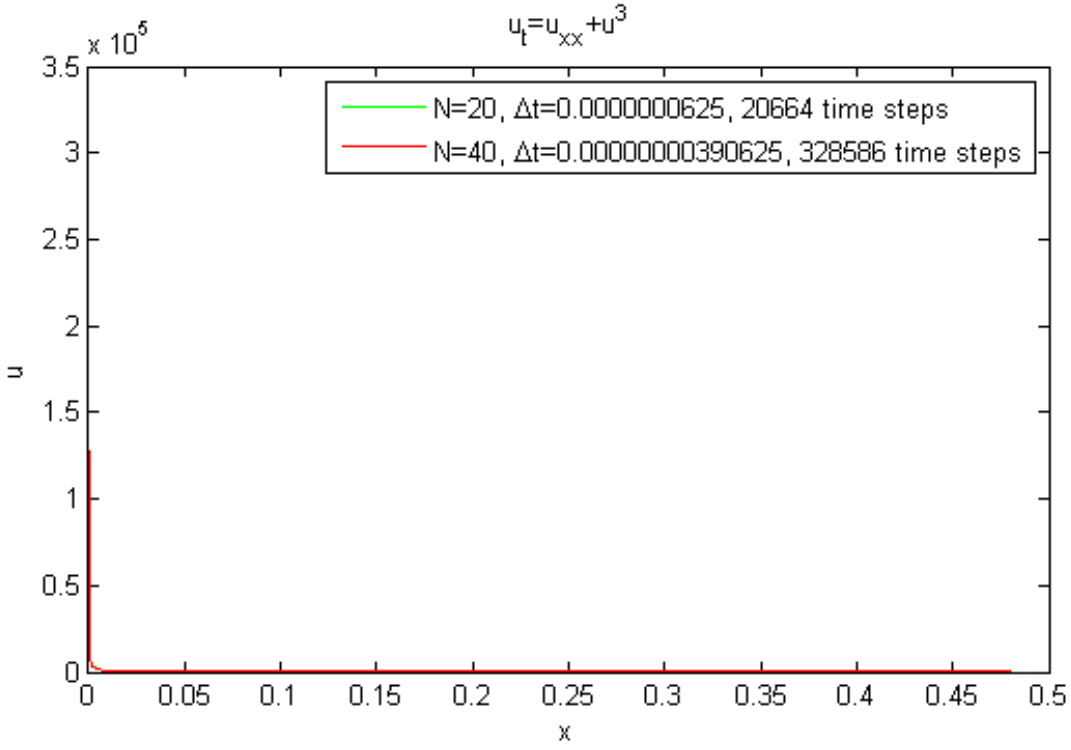


Figure 3.3: Blow-up in Fisher's equation for u^3.

3.2 The Kassoy problem

The Kassoy problem is much like the Fisher's equation except that the forcing function is now e^u instead of u^p.

The monitor function we use throughout the numerical results for the Kassoy problem is

$$M(u) = e^u \tag{3.4}$$

which remains invariant under the scaling (1.5).

After much experimentation with the same initial and boundary conditions as for the Fisher's equation it appears that a fixed time interval does not do an adequate job of handling the blow-up. If a relatively large time interval is chosen then after only a few time steps the blow-up will evolve too quickly to keep track of, as can be seen in figure 3.4. On the other hand if a relatively small time interval is chosen then it becomes impractical and expensive to compute.

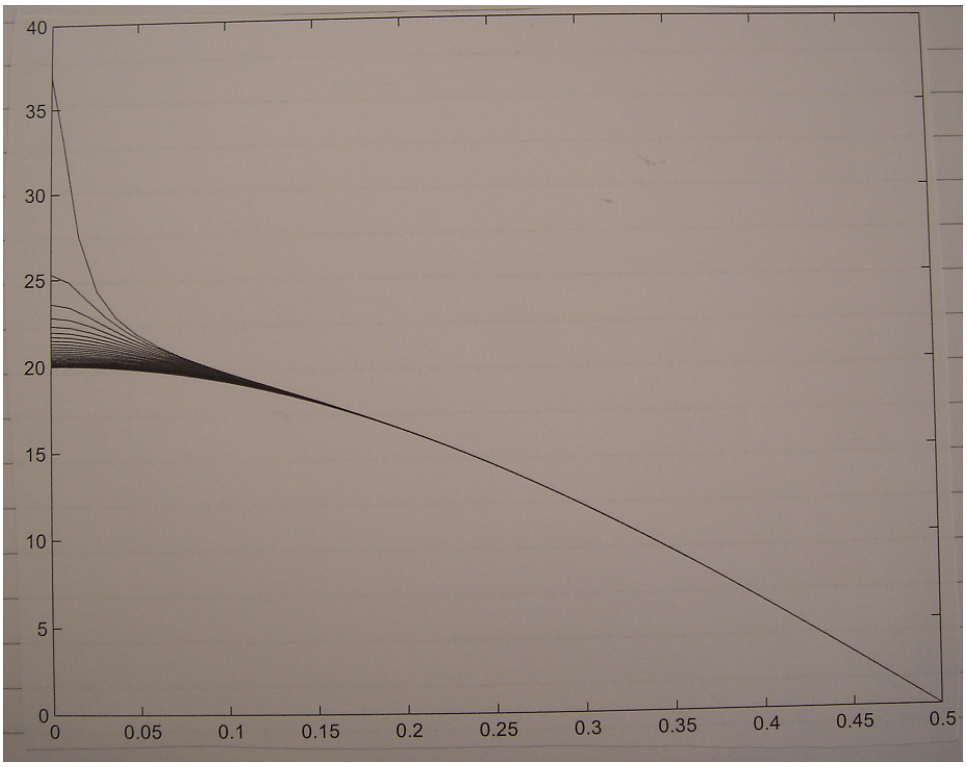


Figure 3.4: Blow-up in the Kassoy problem for $N = 40$ and $\Delta t = 10^{-10}$ for the first 24 time steps.

There is nothing keeping us from using a variable time step that adheres to a certain set criteria, thus we can use an increasingly small time step as we approach the blow-up time thereby always being able to keep track of the development of the problem.

If instead of fixing Δt in the Euler time stepping equation we fix another variable, Δs, that follows the relation

$$\Delta t = 2(T-t)^{\frac{1}{2}}\Delta s \tag{3.5}$$

which is independent of the scaling factor λ so that Δt becomes variable and tends towards zero as t tends towards T. Euler time stepping therefore becomes

$$x^{n+1} = x^n + \underbrace{2(T-t)^{\frac{1}{2}}\Delta s}_{=\Delta t}\dot{x} \tag{3.6}$$

and similarly for θ,

$$\theta^{n+1} = \theta^n + \underbrace{2(T-t)^{\frac{1}{2}}\Delta s}_{=\Delta t}\dot{\theta}. \tag{3.7}$$

A drawback of defining Δt in this way is that we need an estimation for the blow-up time T, which is not difficult if we know the evolution of the solution, but if the blow-up time is unknown then it can become an issue. Although it should be noted that it is better to over estimate T initially rather then under since if we under estimate T then Δt will reach zero before the blow-up time, and so the actual blow-up time will never be reached.

For a slightly modified initial condition ($u(x,0) = 2\sin\left[\pi\left(x+\frac{1}{2}\right)\right]$) we obtain figure 3.5 which follows the development of the singularity much better then when a fixed Δt was used giving an estimate of $T \approx 2.276 \times 10^{-7}$

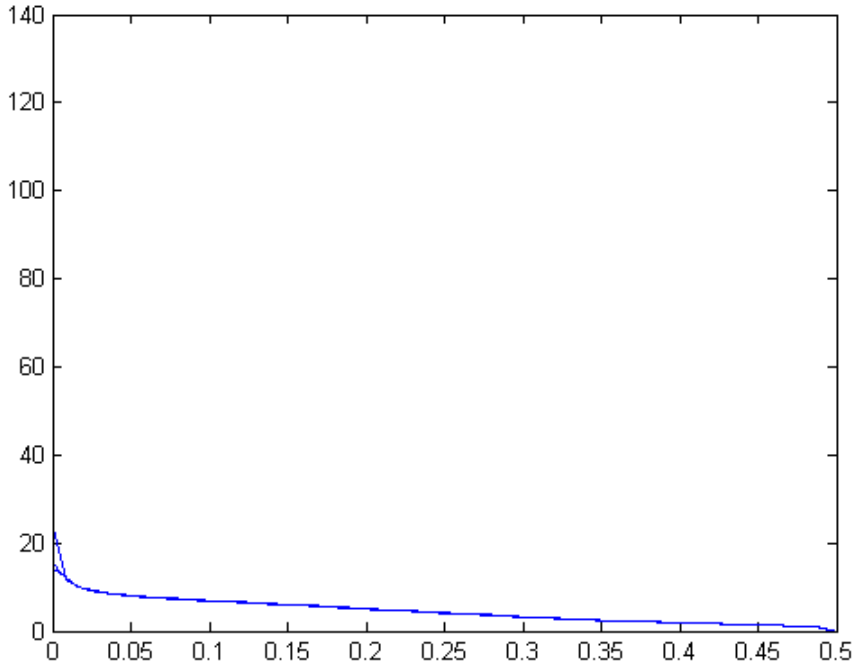


Figure 3.5: $\Delta s = 10^{-8}$, $N = 40$, $T = 2.3 \times 10^{-7}$ and $t \to 2.27642 \times 10^{-7}$

This seems to be what we would expect from the problem, it should be blowing up more rapidly then the Fisher's equation. It is also much quicker to compute then if a small fixed Δt were used. It appears that it still does not follow the development of the problem to the true blow-up time from assumption you would expect the incline towards the blow-up to be steeper then for anything the Fisher's equation could generate due to it growing exponentially. But comparing figure 3.5 with figure 3.3 the numerical results for the Kassoy problem has less of an extreme gradient at the blow-up time.

3.3 The nonlinear Schrödinger equation

Assuming radial symmetry the nonlinear Schrödinger equation is

$$i\frac{\partial\psi}{\partial t} + \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + |\psi|^2\psi = 0 \tag{3.8}$$

where

$$\psi = u + iv$$

and we use the initial condition

$$u(r,0) = \begin{cases} 6\sqrt{2}e^{-r^2} & \text{if } 0 \le r < 5 \\ 0 & \text{if } r \ge 5 \end{cases} \tag{3.9}$$

and boundary conditions

$$u_r(0,t) = 0 \text{ and } u(5,t) = 0 \tag{3.10}$$

using the monitor function

$$M(\psi) = |\psi|^2$$

$u(r,t)$ in fact tends towards zero as r tends to infinity but since it is not possible to compute an infinite region we can truncate it at $r = 5$ in the boundary conditions because the rate at which it tends towards zero is of $O(e^{-r^2})$. Furthermore it should be noted that there is no need for the normalisation factor θ in the nonlinear Schrödinger equation numerical method since the total "mass" of a cell is invariant throughout the evolution.

To solve we seperate the problem into a real part and an imaginary part, solving seperately we can then find a solution for ψ at each time step using much the same method as in the general case. The biggest change is that $\dot{\theta}$ is zero, erasing a term in (2.11), losing the need to calculate $\dot{\theta}$ and (2.10) entirely.

Approaching the nonlinear Schrödinger equation using this method created results that clearly did not describe the blow-up. This seemed to be because of the way the mesh was being redistributed, from (2.11)

$$M(\psi_i)\dot{r}_i = -\int_{r_0}^{r_i} \frac{\partial M(\psi)}{\partial t} r \, \mathrm{d}r + c_i \underbrace{\dot{\theta}}_{=0}. \tag{3.11}$$

Our monitor function in this case is

$$M(\psi_i) = |\psi_i|^2 = u_i^2 + v_i^2$$

thus (3.11) becomes

$$\dot{r}_i = -\frac{1}{|\psi_i|^2} \int_{r_0}^{r_i} \frac{\partial}{\partial t} |\psi_i|^2 \, r \, \mathrm{d}r$$

$$= \frac{2}{|\psi_i|^2} \left[r u^2 \frac{\partial}{\partial r} \left(\frac{v}{u} \right) \right]_{r_0}^{r_i}$$

$$= \frac{2 r_i u_i^2}{u_i^2 + v_i^2} \frac{\partial}{\partial r} \left(\frac{v}{u} \right) \Big|_{r=r_i}. \tag{3.12}$$

For the mesh points to move towards the blow-up (3.12) must be negative, meaning $\frac{\partial}{\partial r}\left(\frac{v}{u}\right)$ must be less then zero. Plotting $\frac{v}{u}$ against r it was easy to see the mesh points were not all being attracted towards the singularity.

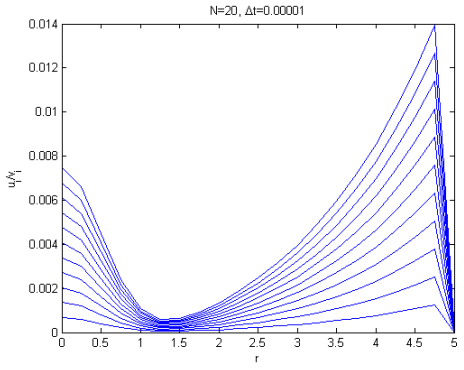


Figure 3.6: Plot of $\frac{v_i}{u_i}$ against r_i

From figure 3.6 it is clear that throughout time there is a local minimum within the domain at $r = 1.25$. $\dot{r}_i$ is determined by the relation

$$\dot{r}_i = \frac{2r_i u_i^2}{u_i^2 + v_i^2} \frac{\partial}{\partial r} \left(\frac{v}{u}\right)\Big|_{r=r_i} \tag{3.13}$$

and is positive except when $\frac{\partial}{\partial r} \left(\frac{v}{u}\right)\Big|_{r=r_i}$ is negative meaning that before this point the nodes are attracted to the singularity, at $r = 1.25$ the node remains static and any points after the minimum will actually be travelling away from the singularity. This will actually reduce resolution at the singularity from our initial mesh and therefore this method will not work for the nonlinear Schrödinger equation under the current conditions.

Chapter 4

Discussion

In this paper we have shown that for problems exhibiting a single blow-up it is possible to follow the evolution using an adaptive mesh method that has a finite number of mesh points and remains in physical space, yet still providing results that concur with the underlying asymptotic structure. In addition to giving an overview of preceding work done by Budd et al that does a similar job but by solving in computational space.

Comparing the two moving mesh methods, the moving mesh PDE method is very efficient needing very few nodes to give a good approximation to the blow-up, it's weakness is it's reliance on small parameters and fixes to create a robust program. In contrast, the conservative method although less efficient is also much simpler requiring no adjustments or fixes to the method.

Future work includes trying different monitor functions, for example arc length $M(u) = \sqrt{1+u_x^2}$ for the Fisher's equation and Kassoy problem or $M(\psi) = \left|\frac{\partial \psi}{\partial r}\right| - \frac{1}{2}|\psi|^4$ for the nonlinear Schrödinger equation as these properties also remain invariant ([3] and [5] respectively). This change in monitor function may improve our results which would be of particular interest for the

nonlinear Schrödinger equation as it was the monitor function that defined $\dot{r}$ and thus why the process failed.

In addition, it may also be worth looking into a variable Δt for more than just the Kassoy problem and whether a higher accuracy time stepping such as Heun's method makes a significant difference to the overall results. Similarly the C++ program currently uses the Trapezium rule for numerical integration if we had more time then maybe it would be good to investigate if alternative methods have an impact. The program also assume that a single blow-up occurs at x_0 future work could generalise this to look at blow-up within the domain or to take into account multiple blow-ups.

Bibliography

[1] J. Bebernes and S. Bricher, *Final time blowup profiles for semilinear parabolic equations via center manifold theory*, SIAM J. Math. Anal. **23** (1992), no. 4, 852–869.

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

[3] Chris J. Budd, Weizhang Huang, and Robert D. Russell, *Moving mesh methods for problems with blow-up*, SIAM Journal on Scientific Computing **17** (1996), no. 2, 305–327.

[4] Weizhang Huang, Ren, and D. Russell, *Moving mesh partial differential equations (mmpdes) based on the equidistribution principle*, SIAM J. Numer. Anal **31** (1994), 709–730.

[5] Y. Tourigny and J.M. Sanz-Serna, *The numerical study of blowup with application to a nonlinear schrodinger equation*, Journal of Computational Physics **102** (1992), 407–416.