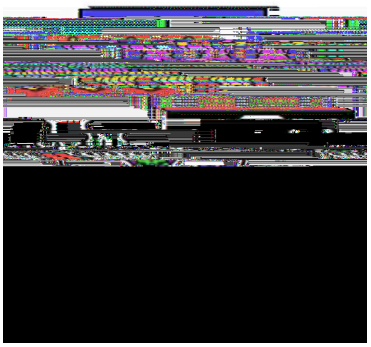


University of Reading

MSc Dissertation

Numerical Approximation of a Quenching Problem

Author:

Michael Conland

Supervisor:

Professor Michael J. Baines

August 21, 2010

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

Signed:

Date:

Abstract

This dissertation contains a study of the effectiveness of using an r-adaptive moving mesh method on the quenching equations $u_t = \frac{1}{a^2}u_{xx} + f(u)$ and $u_t = \frac{1}{a^2}r^2u + f(u)$, where $f(u) = \frac{1}{(1-u)}$, which under the right conditions are known to blow up in finite time in the centre of the domain on which they are being solved. Preliminary studies are carried out using fixed mesh methods, before using u_t as a monitor function for calculating nodal velocities in both a one and a two dimensional case. Numerical results are compared and conclusions drawn on the use of u_t as a monitor function.

Acknowledgements

The greatest of thanks must be directed towards my supervisor Mike Baines, whose door was never closed. Without his continued support, wealth of knowledge, ideas and that draw which never seems to run out of scrap paper, none of this would have been possible.

Contents

Chapter 1

An Introduction

the domain $0 <= x <= L$, then if $L > 2\sqrt{2}$, $u(L/2; t)$ will reach unity in finite time and if that is the case, $u_t(L/2; t)$ will become unbounded in finite time. Put more crudely, equation (1.1) will generate quenching/blow up in the centre of the domain, in finite time, assuming the domain's length is greater than $2\sqrt{2}$.

However, it is not always the case that quenching occurs at a single point in a domain. Chan and Ke [5] and Chan [4] found that, for equation (1.1), if $f(0) > 0$, $f' > 0$ and $f'' > 0$, the solution will eventually reach the quenching point throughout the domain. This is referred to as complete quenching. Studies have also been undertaken as to what occurs post quenching, as (according to [9]) the post quenching results can also have a physical meaning.

On another note, despite the earlier statement that the quenching referred to in this paper will be mathematical, mathematical quenching is often related to the modelling of physical processes. Budd et al [2] state that quenching, amongst other mathematical phenomena, is known to appear in problems concerning gas modelling (Liang et al base their paper on gas within a porous wall), combustion, detonation and mathematical biology amongst others.

This project will centre around the solving of a quenching problem using an adaptive method and thus we must consider why the use of an adaptive method is particularly relevant here, since it is possible to find a quenching solution using a fixed mesh. Obviously a numerical method is required because there is no analytic solution to equation (1.1) and despite it being possible that a complete quenching solution exists, this only occurs under specific conditions. Often, the quenching will occur at a single point and the issue with a fixed mesh method is that, without high resolution, the quenching point can be missed entirely. As such, unless the quenching point is known before the mesh is created, a very large number of nodes is required to guarantee the uncovering of a quenching point.

This is where the adaptive methods become ideal, especially the h and r-methods (which will be explained shortly). By starting with a fixed grid and either adding extra nodes in or shifting them based on the solution, one can create a high resolution of nodes around the point where the greatest rate of change (and hence the quenching point) occurs. Therefore the advantage of using an adaptive method for a quenching problem is that the

mesh resolution about the quenching point is increased, thus increasing the chance of actually locating the quenching point (assuming we do not know its location). Moving mesh methods also have the possibility of being more e cient than their mesh re nement counterparts, as in theory they should use fewer nodes and therefore less computational e ort.

1.2 Adaptive Methods

Generally there are three forms of adaptive method, these being the mesh re nement (h), order enrichment (p) and mesh movement (r). This dissertation will centre around the use of a mesh movement method with a xed number of nodes. Rather than using the more well known mesh re nement methods, which add extra points into the mesh where they are required, the number of nodes will be set at the start of the method and these will be shifted to the areas of the domain which require them.

There are both positives and negatives to the use of the mesh movement methods rather than the re nement methods. Budd et al [2] state that h-methods have been studied in great depth over many years, so much so that these methods now exist in commercial codes. r-methods on the other hand have seen far less interest and as such there are fewer studies to consider. Adding that adaptive methods can su er greatly from a requirement for high levels of computation. Generally, the nodes in the mesh are often shifted using another PDE, as well as the PDE which is actually being solved upon the mesh. As such the computational cost can be rather large, since two di erent systems are being considered. This introduction of another PDE can also lead to a need to solve a sti

solution. It is also stated in [2] that r-methods lend themselves well to problems in which the spatial and temporal length scale are very di erent, suggesting that this should result in an r-adaptive method being suitable in the solution of a quenching problem.

1.3 Monitor Functions and Methods For Solution

Having brie y explained both quenching and the type of method being considered here, the means of calculating the changes to the mesh should be considered. There are two aspects of the r-adaptive method which dramatically change the solution of the PDE. These two aspects are the monitor function and subsequently the class of the method.

If we consider two examples for this, we can see the scope for di erent methods for the solution. The rst of these is the aforementioned paper by Liang et al [9]. This is technically an h-method, but the monitor function used serves as a good example for the di erent choices which can be made. In [9] a monitor function is used which is

$$(u_t) = \sqrt{1 + u_{tt}^2};$$

an arc length monitor function. It should be noted that in [9], equation (1.1) is being solved by considering the domain to have some number of equally spaced nodes in the spatial direction. However, it is the time stepping in this paper which is adaptive and this is what the monitor function is used for. The monitor function governs the time step at each of the spatial nodes and alters it based on values of u_t.

There are many di erent options when it comes to using monitor functions in re nement methods. One which is particularly relevant here is that used by Baines et al [1], as well as the various others noted in [2]. This monitor function is the mass (or area) under the function u in the domain and the study in [1] is particularly relevant to this paper as the method used here is very similar. The method used in [1] of conserving the mass under the curve in time equates to preserving the area under the curve between two nodes to be its initial values. The nodes are moved by forcing each of these areas to retain their area (relative to the total mass under the curve) at each time step.

The subsequent part of such schemes is again summarised by Budd et al in [2]. Two options are referred to as velocity and location based methods. Velocity based methods

or produce more e ective results around the quenching point than that of their standard nite di erence counterparts.

The two adaptive methods shall then be considered in sections 3 and 4 respectively. They will use the same initial conditions as the preliminary tests and in the case of the one dimensional problem, exactly the same equation. The two dimensional case will

Chapter 2

Preliminary Tests

The reasoning behind performing these basic tests is to give benchmark values using schemes with known error estimates. The equation proposed in [8] has no analytic solution and therefore there is no de nite solution to test the adaptive methods against. As such, two preliminary tests will be undertaken, one for the one dimensional case and a second to give an approximation to a two dimensional solution, since the the problem posed is

Now that the initial and boundary conditions have been set, the solution can begin. If the mesh indices are $j;n$ and one lets $u(t;x) = u(n\ \ t;j\ \ x)\ \ \ u_j^n$ and then expands the u_{xx} terms using a central di erence scheme and the u_t term as a forward di erence scheme, then (2.1) becomes

$$\frac{u_j^{n+1}\ \ u_j^n}{t} = \frac{u_j^n{}_{\ 1}\ \ 2u_j^n + u_{j+1}^n}{a^2\ \ x^2} + f(u_j^n): \tag{2.3}$$

which can be rearranged to show that

$$u_j^{n+1} = u_j^n + \ \ t\ \ \frac{u_j^n{}_{\ 1}\ \ 2u_j^n + u_{j+1}^n}{a^2\ \ x^2} + f(u_j^n)\ \ : \tag{2.4}$$

By considering (2.1) and using the fact that an approximation for u has been found using (2.4), one can also very quickly derive a approximation for u_t. If the right hand side of equation (2.1) is expanded in the same way as above, then one has

$$(u_t)_i = \frac{u_j^n{}_{\ 1}\ \ 2u_j^n + u_{j+1}^n}{a^2\ \ x^2} + f(u_j^n): \tag{2.5}$$

Having an approximation for u_t as well as u is particularly useful, as it is the u_t value's growth which causes the quenching and as such, an approximation of it with a known truncation error provides a useful tool for analysing the adaptive one dimensional case.

Using this fairly straightforward method of numerically solving the problem also allows for the truncation error of the scheme to be found fairly simply. If we consider equation (2.3), then the truncation error estimate can be calculated by rst considering

$$_j^n = \frac{u_j^{n+1}\ \ u_j^n}{t}\ \ \ \frac{u_j^n{}_{\ 1}\ \ 2u_j^n + u_{j+1}^n}{a^2\ \ x^2}\ \ \ f(u_j^n); \tag{2.6}$$

with u_j^n replaced by $u(j\ \ x;n\ \ t)$. These terms can all be expanded using the *Taylor Series* about $u(j\ \ x;n\ \ t)$. Applying this expansion to equation (2.6) results in

$$_j^n = \frac{1}{t}\Big[u + \ \ tu_t + \frac{t^2}{2}u_{tt} + \qquad u$$

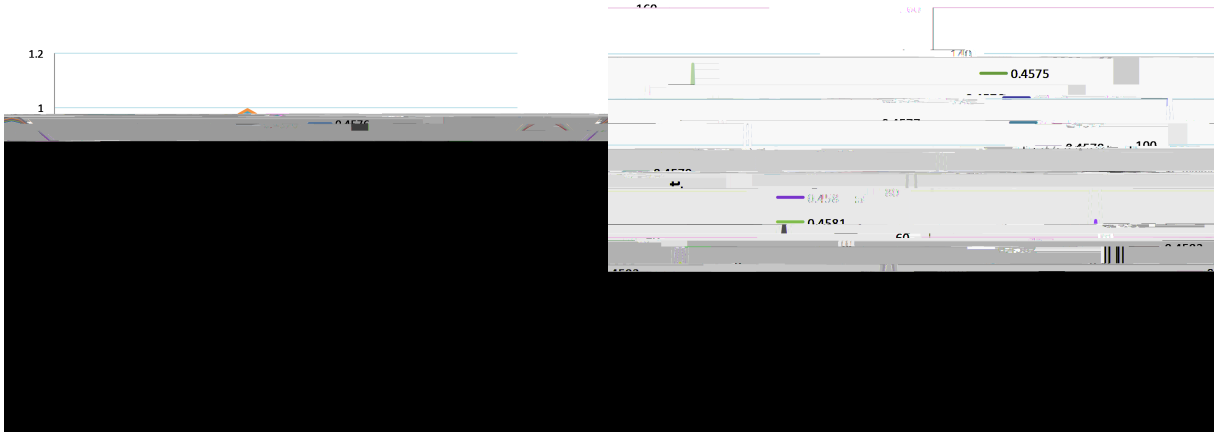
$$\frac{1}{a^2\ \ x^2}\Big[u\qquad xu_x + \frac{x^2}{2}u_{xx}\ \ \ \frac{x^3}{3!}u_{xxx} + \frac{x^4}{4!}u_{xxxx} + :::$$

$$2u + u + \ \ xu_x + \frac{x^2}{2}u_{xx} + \frac{x^3}{3!}u_{xxx} + \frac{x^4}{4!}u_{xxxx} + :::\Big]\qquad f(u);$$

where u represents $u(j\ \ x;n\ \ t$ h0 . 48 w 0 0 m 15 . 975 0 l S Q Tf 7.057 0 Td 01

(a) u plotted against x (b) u_t plotted against x

Figure 2.2: u and u_t plotted in time between $t = 0.451$ and $t = 0.458$ with intervals of 0.001.



(a) u plotted against x (b) u_t plotted against x

Figure 2.3: u and u_t plotted in time between $t = 0.4575$ and $t = 0.4587$ with intervals of 0.0001.

We can see from the results in gures 2.1-2.3 results that, given the parameters $t = 0.0001$ and $x = 0.02$, the problem quenches at approximately $t = 0.4587$. Figure 2.3a shows how u is just touching 1 at this time and by this point u_t has certainly blown up, as evidenced by gure 2.3b. By the next time step the solution is post quenching and the results no longer provide any useful information.

These results correspond to those of [9] and the solution behaves as stated by [7]. As

such, these results should give a reasonable estimation of what to expect from the one dimensional adaptive case, as this will use the same initial and boundary conditions.

2.2 A Test in Two Dimensions

In this two dimensional case, equation (2.1) will need to be modi ed so that it applies to a two dimensional case. The problem is known to be symmetrical and therefore, the problem will also have radial symmetry. This lends it to being solved as a radial di erentially equation, rather than using a $\underline{r}^2 u$ term. Therefore, as a radial problem, equation (2.1) becomes

$$u_t = \frac{1}{a^2 r}\frac{@}{@r}\quad r\frac{@u}{@r}\quad + f(u); \tag{2.8}$$

where $f(u)$ is still de ned as in equation (2.2), with remaining set to one.

If $u(t; r) = u(n\ t; j h)$ u_i^n (where h is the spacing between nodes along the radius)

@ (where)52 T 0 m 16.478 0 160w869 -27(f)2

=een

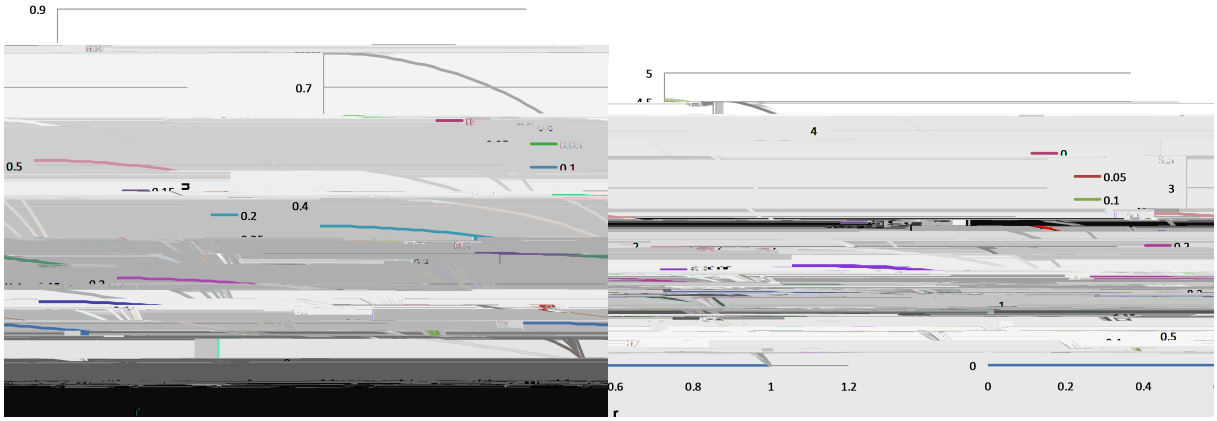
Applying a central di erence schemes to this results in

$$\frac{u_j^{n+1} \quad u_j^n}{t} = \frac{2}{h^2 a^2} \quad u_j^n \ _1 \quad 2u_j^n + u_{j+1}^n \quad + f(u_j^n);$$

which can be further rearranged to show that

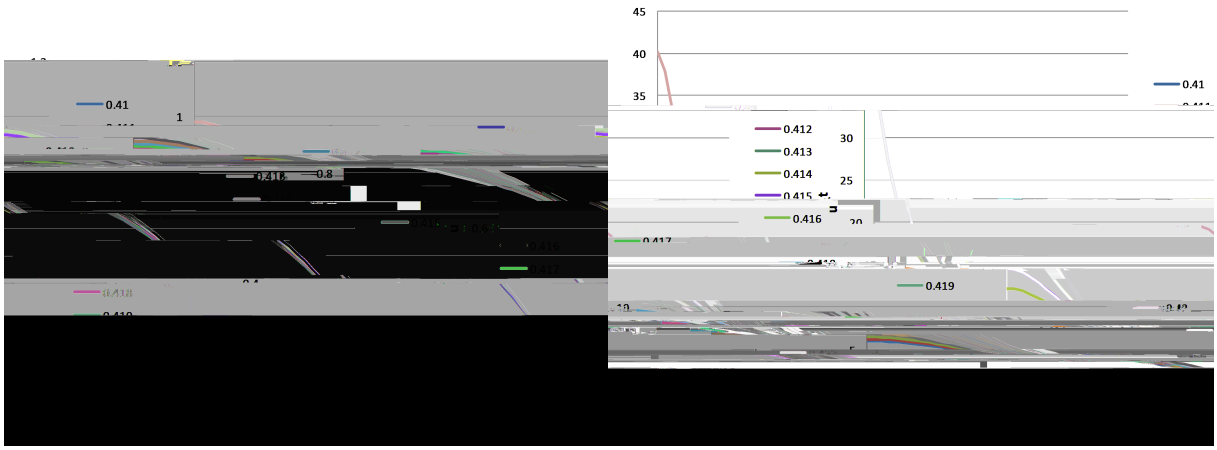
$$u_j^{n+1} = \frac{2 \ t}{h^2 a^2} \quad u_j^n \ _1 \quad 2u_j^n + u_{j+1}^n \quad + \quad tf(u_j^n) + u_j^n: \tag{2.11}$$

With this condition imposed and the initial condition $u(0; r) = \frac{1}{10} \cos \ _2 r$ and the boundary condition $u(t; 1) = 0$, the solution can begin. The following gures show the solution using $t = 0{:}0001$ and $x = 0{:}02$ for several di erent time intervals.



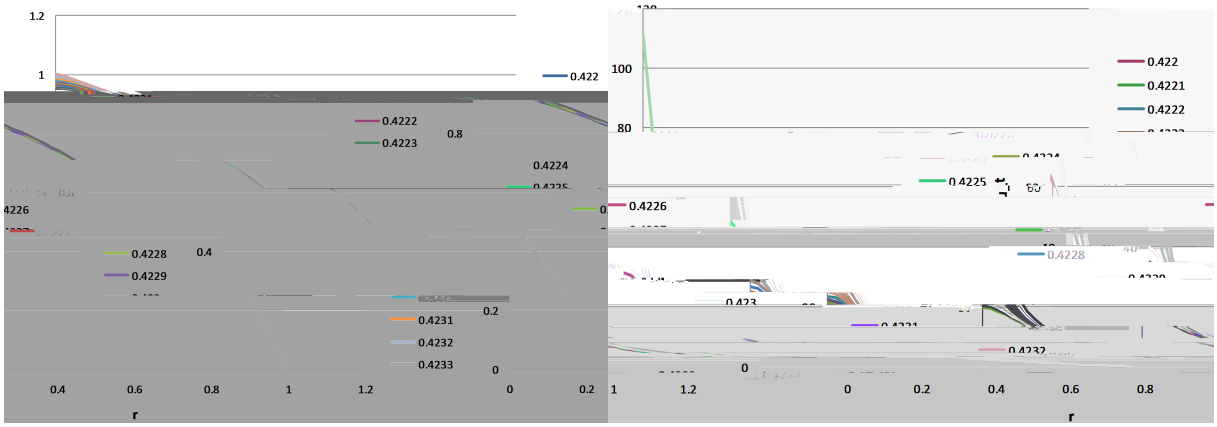
(a) u plotted against r (b) u_t plotted against r

Figure 2.4: u and u_t, plotted in time between $t = 0$ and $t = 0{:}4$ with intervals of 0.05.



(a) u plotted against r (b) u_t plotted against r

Figure 2.5: u and u_t, plotted in time between $t = 0{:}41$ and $t = 0{:}423$ with intervals of 0.001.



(a) u plotted against r (b) u_t plotted against r

Figure 2.6: u and u_t, plotted in time between t = 0:422 and t = 0:4233 with intervals of 0.0001.

This version of the problem quenches at t = 0:4234, which is slightly faster the one dimensional case, although it follows the expected pattern as u and u_t evolve, and nally it quenches on the left hand boundary, which represents the origin here. As with the one dimensional case, the t and x chosen ensure that the problem is converging. Using smaller values would show a more acc7(v)54(a.)-624(Using)]TJ -178.278 -23.114 cc7(v)54(a.)-624(Usinn

Chapter 3

One Dimensional Adaptive Method

Starting from the scaled equation from [9], we have

$$u_t = \frac{1}{a^2} u_{xx} + f(u) \tag{3.1}$$

where $f(u) = \frac{1}{(1 \quad u)}$ and $\quad = 1$. The boundary conditions are given by $u(0; t) = u(1; t) = 0$ and the initial condition chosen is $u(x; 0) = \frac{1}{10} sin(\ x)$. This choice of initial condition will be explained later.

There are numerous ways in which to apply an adaptive method. In this case and using this equation, it is known that the u_t term in the solution of the PDE is the part of the equation causing the quenching or blow-up. In [1] u is used as the monitor function, but in this dissertation u_t will be used to govern the re nement of the mesh in a very similar fashion to that of [1]. If one lets A_i be the absolute area between two points x_i and x_{i+1} then we have

$$\int_{x_i}^{x_{i+1}} u_t \, dx = A_i. \tag{3.2}$$

However, since the total integral from 0 to 1 is not conserved, conservation of the absolute area is of little use in this case. A better measure is that of a percentage or relative area with respect to the total area under u_t. If we de ne a new variable $\quad$ to represent the total area and C_i to represent the relative area under the curve between x_i and x_{i+1} then we have

$$= \int_0^1 u_t \, dx \tag{3.3}$$

and the relative area is

$$C_i = \frac{1}{-}A_i = \frac{1}{-}\int_{x_i}^{x_{i+1}} u_t \, dx: \tag{3.4}$$

It is this quantity that will be conserved in time. At this point the choice of initial conditions must be explained. From [9], we know that the initial u values must be in the region 0 u_0 < 1, with $u(0) = u(1) = 0$). Also, equation (3.3) stipulates that the initial condition must also be twice di erentiable in time, due to the use of the initial u_{xx} term. The sine function $\frac{1}{10}\sin(\ x)$ is used, since it satis es these criteria.

We need to know how changes with respect to time, as this will a ect the areas C_i in equation (3.4). As such, we di erentiate equation (3.3) with respect to time to give

$$\frac{d}{dt} = \frac{d}{dt}\int_0^1 u_t \, dx = \int_0^1 u_{tt} \, dx:$$

We can now substitute equation (3.1) into this to give

$$\frac{d}{dt} = \int_0^1 (\frac{1}{2}u_{xx} + f(u))_t \, dx:$$

$$\overline{a6s2}$$

If we rstly considers the second term, then we can see that the *Leibniz Integral Rule* can be applied to show that

$$\frac{d}{dt}\int_{x_i}^{x_{i+1}} u_t\,dx = \int_{x_i}^{x_{i+1}} u_{tt}\,dx + \left[u_t\frac{dx}{dt}\right]_{x_i}^{x_{i+1}};$$

which implies that

$$\frac{d}{dt}\left[\frac{1}{-}\int_{x_i}^{x_{i+1}} u_t\,dx + \frac{1}{-}\int_{x_i}^{x_{i+1}} u_{tt}\,dx + \left[u_t\frac{dx}{dt}\right]_{x_i}^{x_{i+1}}\right] = 0.$$

Once again, u_t can be substituted into the u_{tt} term from equation (3.1), as well as using $\frac{d}{dt}(\frac{1}{-}) = \frac{-}{-^2}$. This gives

$$\frac{-}{-^2}\int_{x_i}^{x_{i+1}} u_t\,dx + \frac{1}{-}\int_{x_i}^{x_{i+1}} \left[\frac{1}{a^2}u_{xx} + f(u)\right]_t dx + \left[u_t\frac{dx}{dt}\right]_{x_i}^{x_{i+1}} = 0$$

Clearly this can be multiplied through by , but also note that earlier C_i was de ned as equation (3.4). This can be substituted into the rst term, along with the multiplication by to give

$$\_C_i + \int_{x_i}^{x_{i+1}} \left[\frac{1}{a^2}u_{xx} + f(u)\right]_t dx + \left[u_t\frac{dx}{dt}\right]_{x_i}^{x_{i+1}} = 0.$$

Substituting equation (3.1) in again and rearranging produces

$$\left[u_t\frac{dx}{dt}\right]_{x_i}^{x_{i+1}} = \_C_i - \frac{1}{a^2}\int_{x_i}^{x_{i+1}} \left[\frac{1}{a^2}u_{xx} + f(u)\right]_{u_{xx}} dx$$

where $_i$

Substituting $v = u_{xx}$ into this gives

$$C_i = \frac{1}{-} \int_{x_i}^{x_{i+1}} f(u) \quad \frac{v}{a^2}\, dx; \tag{3.8}$$

which can be approximated using the trapezium rule. The C_i values are only calculated once, directly after the initial v^0s have been calculated and are constant for all time.

If we rst consider the problem at $t = 0$, we know the exact values of u at all points. Once the initial values have been applied, the v values can be calculated and thus an approximation to u_t can also be calculated. This is given by

$$u_t = f(u) \quad \frac{v}{a^2}: \tag{3.9}$$

Once the time loop begins, u_t is considered in two di erent ways. Firstly an approximation of u_t is calculated from equation (3.9), using v at each node. All of the other calculations mentioned so far use this approximation. u_t is approximated at the end of the loop, this time using a midpoint rule applied to equation (3.4). The v values are also recalculated at the end of the loop.

Now that u_t has a value at all points, the time loop can begin. Bearing in mind that u_t, , the C_i values and $\underline{v}$ have all been calculated before the time loop begins, the rst thing to be calculated in the loop is _.

_ has already been de ned in equation (3.5). However, in equation (3.5), all of the u_t terms have been expanded using equation (3.1), but now that u_t has been approximated using v, it can be reintroduced, since this is far simpler to calculate than the two terms which are otherwise created. As such, equation (3.5) becomes

$$- = \frac{u_{tx}}{a^2} \Big|_0^1 + \qquad \text{k}$$

with a suitable choice of t. Next comes the movement of the nodes. This is calculated using equation (3.6). In equation (3.6), every term is known apart from the two $\frac{dx}{dt}$ terms. However, two of the $\frac{dx}{dt}$ values are known. To keep the domain the same size for all time, the two outermost nodes (x_0 and x_N) are fixed in position and therefore $\frac{dx}{dt}$ at these nodes is 0. In equation (3.6) the $\frac{dx}{dt}$ terms are evaluated at x_i and x_{i+1}. Starting from $i = 0$, the evaluation at x_i (since there is no node movement at this point) is known and therefore equation (3.6) can be rearranged to find the unknown $\frac{dx}{dt}$ term and all the subsequent $\underline{x}$ terms explicitly.

As with equation (3.5), one can reform any expanded u_t terms using (3.1), as u_t is now known. As such, equation (3.6) becomes

$$u_t \frac{dx}{dt}\Big|_{x_i}^{x_{i+1}} = \_C_i \quad \frac{u_{tx}}{a^2}\Big|_{x_i}^{x_{i+1}} \quad \int_{x_i}^{x_{i+1}} \frac{1}{a^2} f^{\emptyset}(u) u_t \, dx:$$

This can then be rearranged into the form

$$\underline{x}_{i+1} = \frac{}{u_t j_{i+1}}; \tag{3.11}$$

where

$$\underline{x}_{i+1} = \frac{dx}{dt}\Big|_{x_{i+1}}$$

$$= \_C_i + u_t \underline{x} j_{x_i} \tag{3.12}$$

$$= \frac{u_{tx}}{a^2}\Big|_{x_i}^{x_{i+1}} \tag{3.13}$$

and

$$= \int_{x_i}^{x_{i+1}} f^{\emptyset}(u) u_t \, dx: \tag{3.14}$$

can generally be approximated using finite differences, with the derivatives being approximated using a central difference scheme. However, the scheme for calculating does need to be slightly modified if the velocity being calculated is at the first internal node x_1.

Therefore in general

$$\frac{1}{a^2} \quad \frac{(u_t)_{i+2} \quad (u_t)_i}{x_{i+2} \quad x_i} \quad \frac{(u_t)_{i+1} \quad (u_t)_{i\ 1}}{x_{i+1} \quad x_{i\ 1}} \quad : \tag{3.15}$$

But, at $i = 0$

$$\frac{1}{a^2} \quad \frac{(u_t)_{i+2} \quad (u_t)_i}{x_{i+2} \quad x_i} \quad \frac{(u_t)_{i+1} \quad (u_t)_i}{x_{i+1} \quad x_i} \quad :$$

can be approximated by simply using the trapezium rule. However, wherever the trapezium rule is being used to approximate an integral between x_i and x_{i+1}, the approximation accuracy is very dependent on the number of nodes being used. The values between nodes are unknown and thus it is not possible to increase the number of trapeziums being used to approximate any given integral. Therefore to maintain the accuracy of these approximations, a reasonable number of nodes must be used throughout.

Once these velocities have been calculated, the positions of the nodes can be updated using the same forward Euler method,

$$x_i^{n+1} = x_i^n + \quad t\underline{x}_i^n: \tag{3.16}$$

Once the nodes have been repositioned, the new value of u_t at the nodes can be approximated by applying the midpoint rule to equation (3.4) in the form

$$\frac{1}{\quad} \int_{i\ 1}^{i+1} u\,dx = C_{i\ 1} + C_i:$$

The standard midpoint rule applied here produces

$$(x_{i+1} \quad x_{i\ 1})\, u_t j_{x_i} \qquad (C_{i\ 1} + C_i);$$

which can then be rearranged to show that

$$u_t j_{x_i} \qquad \frac{C_{i\ 1} + C_i}{x_{i+1} \quad x_{i\ 1}}: \tag{3.17}$$

The extra is present to make up for the fact that the C's are actually a relative measurement of the areas under u_t, rather than the absolute values. It would be possible to use the absolute areas A_i, but this would mean recalculating A_i at every time step using the trapezium rule, rather than applying the above, simpler step.

Normally (if a xed grid is used), since u_t is now known, one could use

$$u_i^{n+1} = u_i^n + \quad tu_t$$

to nd the new u_i values. However, a di erent approach must be taken here. We start by de ning a new variable to be

$$_i = \int_{x_{i\ 1}}^{x_{i+1}} u\,dx: \tag{3.18}$$

$_i$ can be approximated using a trapezium rule before the time loop begins. We can then di erentiate $_i$ and use this derivative to calculate the new $_i$ values. As such, di erentiating (3.18) with respect to time produces

$$-_i = \frac{d}{dt} \int_{x_{i\ 1}}^{x_{x+1}}$$

3.4 Adaptations

The first adaptation used is a simple one. If we consider the calculation of in equation (3.15), we can see that generally can be calculated using a central difference scheme. However, when considering when calculating the velocity of the first internal point, a forward difference scheme must be considered, since the boundary conditions do not allow for any form of ghost point to be considered. This is a problem which can be mediated using the symmetry of the problem. It is known from both [8] and [9] that equation (3.1) is symmetric around the centre of the domain. Therefore we can calculate the the velocity of the nodes from the centre to $u = 1$ (where $\underline{x} = 0$ and then use those values to state the velocities of their opposite nodes on the other side of the centre node.

This however, does enforce another condition on the problem, which is that as well as $\underline{x}$ being 0 at both $x = 0$ and $x = 1$, $\underline{x}$ must also be 0 at $x = 0.5$. This further implies that there must be an odd number of nodes so that one node lies directly on $x = 0.5$.

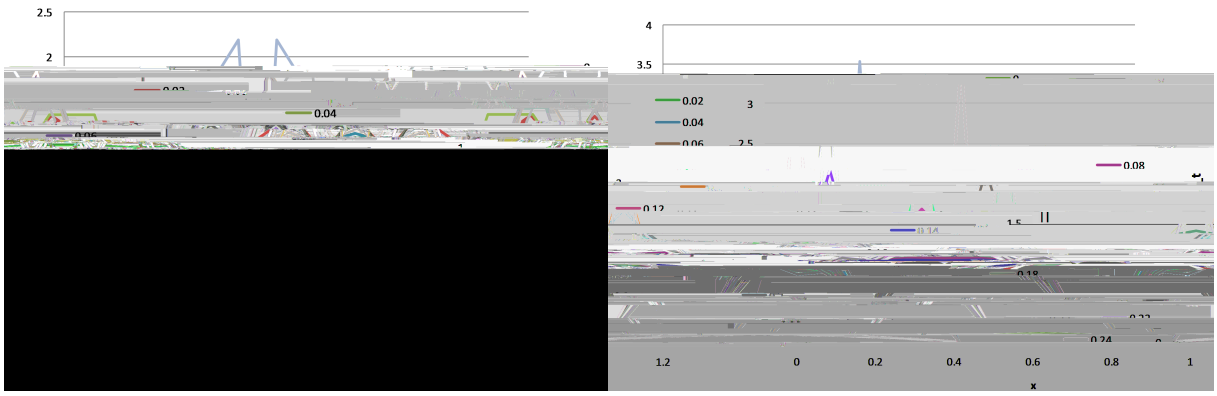
Applying these conditions does allow the method to produce more sensible results, although they are still not entirely expected. The method is far more unstable than the fixed mesh method, but it will run for slightly longer than the original adaptive method using $N = 10$ (N represents the number of areas created, so the number nodes is in fact $N + 1$) and a t value of 0.001, but still fails very quickly. However, using a smaller time step or a greater number of nodes results in the method failing more quickly, so the plots below use $t = 0.001$ and $x = 0.1$.

(a) The solution from $t = 0$ to $t = 0.24$ at intervals of 0.02.

(b) The solution from $t = 0.241$ to $t = 0.25$ at intervals of 0.001.

Figure 3.1: u plotted against x using $t = 0.001$ and $x = 0.1$

So, using this slight modi cation (forcing the central node to remain stationary), a quenching solution can be found. However, it is clearly not forming a solution overly similar to that of the preliminary tests. It lasts a little over half of the time that the preliminary method could run for under the same conditions before quenching. Although whether this can even be described as quenching is debatable. It should also be noted that adding more nodes and reducing t did not improve this solution. Reducing the time step makes little to no di erence, while adding more nodes forces the method to fail even faster. Lowering the number of nodes does appear to aid stability, but the solution becomes unusable in the process as it (Id8a7m17a4245ter.)-466(Lo)2Td [(did)-407(not)-472(soluo.16 m that Seducingthi.1145267so16 0l46l2..114 (the9.4.67-466(LITJ -220.809 -23.113 Td [(ti2gle)220no)-2t Tf



(a) The v estimate of u_t.

(b) The midpoint rule version of u_t

Figure 3.2: The two di erent methods of calculating u_t in the time period of $t = 0$ to $t = 0{:}24$, with intervals of $x = 0{:}1$ and a t value of 0:001.

Other adaptations were made, such as forcing the v end point values to be equal to a^2 (by rearranging (3.1)) and attempting to solve for v using a central nite di erence scheme instead of a nite element scheme. The reasoning behind trying to use these two methods is that in the formulation of the nite elements solution to v, the hat functions ( ) and therefore u_x at each end of v remain present. These hat functions themselves are not particularly an issue, it is the multiplication by the u_x term evaluated on the boundary which is the problem.

The only boundary conditions given are that u (and therefore u_t) must be equal to zero at the boundaries, which means that u_x is not speci ed. This poses a problem as an estimation of u_x needs to be made. A simple way to try to nd these terms was to apply a forward nite di erence scheme to the u_x term at $x = 0$ and a backwards di erence at $x = 1$. However, this is very crude and likely to be a major cause of problems during the solution of v. Unfortunately, without a boundary condition, these terms are very di cult to evaluate.

Based on this, another adaptation considered was assuming that $u_x = 0$ weakly on the boundaries alongside $u = 0$, the aim being to remove the extra terms created in the calculations of v. However, this proved even more unstable than having the approximation to u_x in the solution.

On a positive note, forcing the central node to remain in the same position throughout

and thus removing the tricky term does appear successful. The term when calculating the velocity of the first internal point also needed to be considered using a forward difference scheme. When solved using this, the method barely worked at all. Removing that term by using the symmetry of the problem seemingly improved it. This in turn suggests that perhaps the forward and backward difference schemes used in the calculation of v is another element of the solution causing an issue.

3.5 Critical Analysis

Clearly this method has proved somewhat unsuccessful in this study. Admittedly, it could be something as simple as a programming error, although this seems unlikely. Using the method as originally derived proved completely unusable. The introduction of the forced central node seems to help, but the solution produced is still very poor. Even with the addition of the stationary central node, the scheme is highly unstable, especially when altering the number of nodes. Raising N above 10 results in the solution method failing very quickly indeed, due to the unstable nature of forward Euler time stepping.

It is possible that the issue is inherent in the boundary conditions. By forcing u to be zero at the boundaries, it is possible that u_t suffers at the boundaries since u_t contains a u_{xx} term, which at the boundary is going to differ greatly from that of the internal points because u is being forced to 0. Now, since the monitor function here is u_t, it stands to reason that perhaps it is the boundary conditions causing the issue, since they are forcing u_t to have erratic values near the boundaries. One can see in figure 3.2b that there are three areas with large gradients in the v approximation to u_t. These are most likely caused by the large u_{xx} values being found nearest the boundaries.

One could also consider that the choice of monitor function is not ideal. It can be viewed in two ways. [9] stated that it is u_t causing the blow up and thus, while hoping to improve the resolution of the mesh about the quenching point, it could be argued that moving the mesh with regards to the term forcing the blow up may work. Conversely, since it is u_t

Chapter 4

A Two Dimensional Adaptive Method

Like the one dimensional case this method begins with the equation from [9], but in this case it clearly needs to be modi ed slightly. The equation has already been considered once as a radial problem, but here it will be considered in two dimensions and thus becomes

$$u_t = \frac{1}{a^2} r^2 u + f(u): \tag{4.1}$$

For this case, $f(u)$ remains as before ($f(u) = \frac{1}{(1\ u)}$, with set to 1), but the initial conditions are altered slightly. In this case the domain is circular and although the problem will be thoughp(the)-d(e)-34usingealtho=

where represents the domain on which the equation (equation (4.1)) is being solved, and consider the relative area for this method, thus de ning constants c_i which are given by

$$c_i = \frac{1}{} \int_{i} u_t \, d \quad : \tag{4.3}$$

The integral over $_i$ here represents the integral over the area created between two circular rings of radius r_i and r_{i+1}. If equation (4.3) is rearranged, we get

$$c_i = \int_{i} u_t \, d \quad : \tag{4.4}$$

However, once the mesh begins its movement the values will be calculated using a nite element method. This entails introducing a test (a member of a partition of unity) function to equation (4.3) to produce

$$c_i = \frac{1}{} \int w_i u_t \, d \quad ; \tag{4.5}$$

where w moves with $\underline{v}$ and therefore must satisfy the advection equation

$$\frac{@w_i}{@t} + \underline{v} \quad \underline{r} w_i = 0: \tag{4.6}$$

Equation (4.5) can then be rearranged to show that

$$c_i = \int w_i u_t \, d \quad : \tag{4.7}$$

For the initial values, all the u terms are known initially and the rings are evenly spaced. Now, we need to know how changes with time and therefore equation (4.7) must be di erentiated with respect to time. This gives

$$\_c_i = \frac{d}{dt} \int w_i u_t \, d \quad ;$$

where $\_ = \frac{d}{dt}$. Using the *Reynolds Transport Theorem* this becomes

$$\_c_i = \int (w_i u_t)_t \, d \quad + \quad \oint_{@} (w_i u_t \underline{v}):$$

Then, by using the *Divergence Theorem*, this becomes

$$\int_{\partial} u\underline{v} \cdot \underline{n}\, ds = \int \underline{\nabla} \cdot (u\underline{v})\, d\ :$$

One can quite easily show that equation (4.7) can be further rearranged to give

$$\dot{c}_i = \int w_i u_{tt} + u_t (w_i)_t + \underline{\nabla} \cdot (w_i u_t \underline{v})\, d\ :$$

The third term can then be split using the fact that $\underline{\nabla} \cdot (w_i u_t \underline{v}) = w_i u_t \underline{\nabla} \cdot \underline{v} + \underline{v} \cdot \underline{\nabla} w_i u_t$ to show that

$$\dot{c}_i = \int w_i u_{tt} + u_t (w_i)_t + u_t \underline{v} \cdot \underline{\nabla} w_i + w_i \underline{\nabla} \cdot (u_t \underline{v})\, d\ :$$

Multiplying equation (4.6) by u_t, we can quite clearly see that two of the above terms are cancelled out and thus the equation becomes

$$\dot{c}_i = \int w_i \left(u_{tt} + \underline{\nabla} \cdot (u_t \underline{v})\right) d\ : \tag{4.8}$$

Now one can begin substituting equation (4.1) into the first term of (4.8), which shows that

$$\dot{c}_i = \int w_i \left(\frac{1}{a^2} \nabla^2 u + f(u)\right)_t + \underline{\nabla} \cdot (u_t \underline{v})\ d\ :$$

This can then be expanded to give

$$\dot{c}_i = \int w_i \left(\frac{1}{a^2} \nabla^2 u_t + f'(u) u_t + \underline{\nabla} \cdot (u_t \underline{v})\right) d\ :$$

A further substitution of equation (4.1) can be made, but before this a similar approach as in the one dimensional case is used to remove the complications produced by having to try to evaluate a $\nabla^4 u$ term. Once again a substitution is made, this time allowing $p = \nabla^2 u$. The solution of p will be explained later. Substituting equation (4.1) in again produces

$$\dot{c}_i = \int w_i \left(\frac{1}{a^2} \nabla^2 \left(\frac{1}{a^2} \nabla^2 u + f(u)\right) + f'(u) \left(\frac{1}{a^2} \nabla^2 u + f(u)\right) + \underline{\nabla} \cdot (u_t \underline{v})\right) d\ :$$

If the substitution of p for $\nabla^2 u$ is now used the above equation becomes

$$\dot{c}_i = \int w_i \left(\frac{1}{a^2} \nabla^2 \left(f(u) - \frac{p}{a^2}\right) + f'(u) \left(f(u) - \frac{p}{a^2}\right) + \underline{\nabla} \cdot (u_t \underline{v})\right) d\ : \tag{4.9}$$

So far, the final term of the equation has been left untouched because it needs to be considered in a different way to the other terms. If we consider the term $\underline{\nabla} \cdot (u_t \underline{v})$, then it

As with one dimension, the change in needs to be considered and thus equation (4.16) must be di erentiated with respect to time to give

$$\dot{-}_i = \frac{d}{dt} \int w_i u \, d \; : \tag{4.13}$$

From the previous workings involving and _, it is clear that

$$\dot{-}_i = \int w_i \left(u_t + \underline{r} \quad (u\underline{v})\right) d$$

Once again one of Green's theorems can be applied to the second term to show that

$$\dot{-}_i = \int w_i u_t \quad u \underline{r} \quad (w_i \quad \underline{v}) \, d \; : \tag{4.14}$$

4.1 The Finite Element Part in 2 Dimensions

Currently only the test function w_i has been introduced. To complete a nite elements solution to the problem a function must actually be chosen here and in this case it will be the standard two dimensional hat function . Several equations ((4.11), (4.16) and (4.17)) need to be slightly altered to take into account this hat function. Equation (4.11) becomes

$$\_c_i = \int_{\quad i} \frac{1}{a^2} r^2 \quad f(u) \quad \frac{p}{a^2} \quad + f^{\prime}(u) \quad f(u) \quad \frac{p}{a^2} \quad d \quad \quad K_i(u_t) \_ : \tag{4.15}$$

Equation (4.12) becomes

$$_i = \int_{\quad i} u \, d \tag{4.16}$$

and (by expanding as $_j \; _j$) equation (4.14) becomes

$$\dot{-}_i = \int_{\quad i} u_t \, d \quad \quad K(u)_i \quad _i;$$

which can also (by expanding u_t as $(u_t)_j \; _j$) be written as

$$\_ \dot{-} \_ = M \underline{u}_t \quad K(u) \_ \tag{4.17}$$

$K(u)$ and M will be de ned later in this dissertation and _ will be known after solving equation (4.15), thus reducing equation (4.17) to an explicit means of calculating $\dot{-}_i$.

Now, if u is approximated using $u = \sum_{i=0}^{N} {}_j w_j$, then equation (4.16) reduces to

$$_i = (M \underline{u})_i :$$

This further reduces to

$$\_ = M\underline{u}; \tag{4.18}$$

where M is the standard finite elements mass matrix.

As mentioned in an earlier section, the matrices K and M need to be defined, as well as the matrix functions $K(u_t)$ and $K(u)$. However, before the assembly of the matrices can take place, the triangulation of the region must be considered.

4.2 The Triangulation

The two dimensional problem is being considered upon a circle with radius of 1. As with the one dimensional case, the problem has been scaled and thus, regardless of the actual radius, the solution is always shown on a circle with a radius of one.

To begin the triangulation, there must be a means of choosing nodes, upon which the triangles can be based. In this case, a number (N) of 'rings' will be placed about the centre of the circle. Initially, they will all be equally separated, but once the adaptive method begins this will clearly change. Upon each of these rings, an even number (M) of nodes will be placed. Using an even number of nodes (which remains fixed) retains the symmetry of the triangulation and forces every triangle to be isosceles. These nodes will be alternately placed depending on which ring they lie upon. So for example: if four nodes are chosen, then the first ring out from the centre of the circle will have its nodes positions at 0, 90, 180 and 270 degrees. On the next ring, the nodes will be placed at 45, 135, 225 and 315 degrees. This pattern is then repeated throughout, as illustrated by figure 4.1.

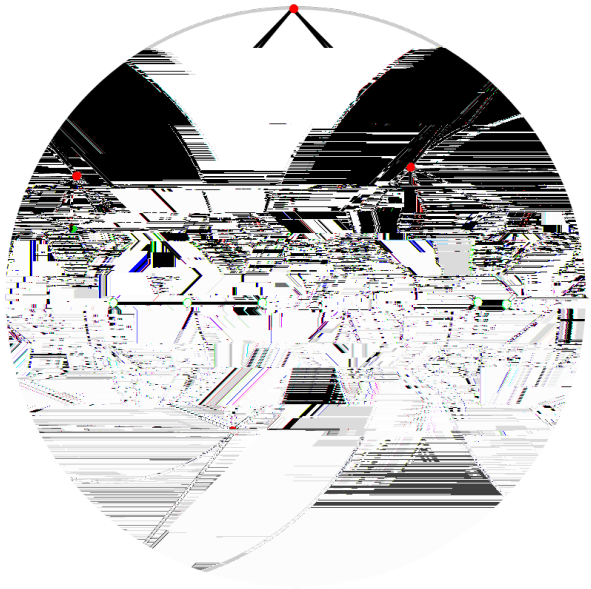
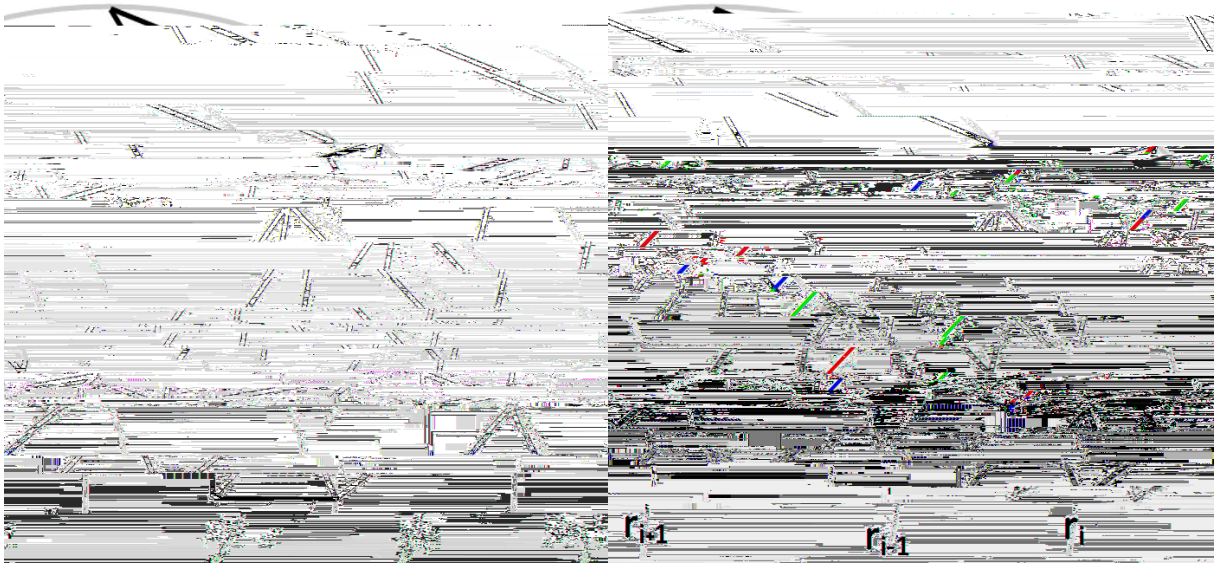


Figure 4.1: For this case, one can see that both N and M are 4. The centre point is counted as one of the rings despite not actually being a ring itself. Also note that this is a very simple example which could not be used in practice, as the angles in the larger triangles are beginning to turn obtuse, at which point the method will fail.

By considering the problem like this, all of the triangles become isosceles, thus lending a great deal of extra symmetry to the problem. However, it does mean that two di erent types of triangle must be considered: inward and outward pointing. Clearly, the circle created by the rst ring and the centre point of the circle will only have inward pointing triangles, but from then on each area created between the rings will be made up of both inward and outward pointing triangles.



(a) The angles used in the triangulation [Note that this gure assume that it is either $K(u_t)$ or $K(u)$ being solved.

(b) Lengths used in the triangulation

Figure 4.2: Labelling of the triangulation

It will become clearer when assembling the aforementioned K and M matrices that every angle, side length and height for all of the triangles must be found. It was stated above that the rst set of triangles (the set between the centre point and the rst ring) are all inward pointing. They are also the most simple to calculate, as the lengths of the pair of equal sides in each triangle are already known, as is the single angle in each triangle.

The rest of the triangles are a little more di cult to calculate. If one considers gure 4.2, then one can see how it is possible to calculate all of the values required.

As stated above, the rst set of triangles are all inward pointing and straightforward to calculate. As such, the rst triangle to be considered after this is an outward pointing one. If one considers the height ($H_{i\ 1}^{I}$) of the inward pointing triangle which has two of its corners touching the *ith* ring, then one can use this to calculate the height (H_i^O) of the outward pointing triangle which shares a base with it. If one lets r_i represent the position of the ith ring, then one can see that the sum of the two heights must be equal to

$$H_{i\ 1}^{I} + H_i^O = r_{i+1} \quad r_{i\ 1}$$

At this point, the two r values are known and H_i^l can be quite easily calculated using *Pythagoras' Theorem* to show that

$$H_{i\ 1}^{l} = \sqrt{A_{i\ 1}^{2} - \frac{R_{i\ 1}^{2}}{4}}: \tag{4.19}$$

Note that both the R and

Finally, the loop used to calculate the values required can only begin if the lengths and angles in the original triangles are known. If there are M nodes on each ring, then

$$_0 = \frac{2}{M} \tag{4.28}$$

$$_0 = \frac{^0}{2} \tag{4.29}$$

$$A_0 = r_1 \tag{4.30}$$

$$R_0 = A_0 \frac{sin\ _0}{sin\ _0}: \tag{4.31}$$

Now that the triangulation is complete, the nite elements formulation and assembly can begin.

4.3 Assembling the Matrices

As stated in an earlier section, there are several di erent matrices which need to be assembled. Three of these matrices (K, $K(u_t)$ and $K(u)$) are very similar and assembling just one of them gives the general form the the remaining two. M will also need to be assembled. However, we shall start by considering $K(u_t)$ rst, as this will also give K and $K(u)$.

It was mentioned during the triangulation that both inward and outward pointing triangles must be considered and therefore there are two elemental matrices which need to be considered for assembly. If one starts by considering the standard nite elements K matrix inei2(in).ontrices which neede52bot3dered [3.343f 148.978 0 Ti882 Td [(sin)-167(311442.726

Where $(K(u_t)^I_e)_i$ represents the elemental matrix for the *ith* inward pointing triangle, L and O represent the midpoint values of u_t of the element. So L_i is equal to the area of the inward pointing triangle, multiplied by the average of the values of u_t on the tips of the triangle. As such

$$L_i = \frac{R_i H^I_i}{2} \quad \frac{(u_t)_i + 2(u_t)_{i+1}}{3} \tag{4.32}$$

and therefore O can be calculated using

$$O_i = \frac{R_{i\ 1} H^O_i}{2} \quad \frac{2(u_t)_i + (u_t)_{i+1}}{3} \tag{4.33}$$

Concentrating again on the matrices representing the elements shows that they can be simpli ed due to the values of on every given ring having the same value. Using this symmetry reduces the two matrix systems to

$$(K(u_t)^I_e)_{i\_} = L_i \begin{pmatrix} 0 & & 1 & 0 \\ \cot_{\ i} & \cot_{\ i} & \text{A} & \text{@} \\ \cot_{\ i} & \cot_{\ i} & & \end{pmatrix}$$

Adding the next element to this produces

$$\begin{pmatrix} L_0 \cot\ _0 & L_0 \cot\ _0 & 0 \\ L_0 \cot\ _0 & L_0 \cot\ _0 + g_1 & g_1 \\ 0 & g_1 & g_1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} = \begin{pmatrix} f_0^I \\ 2f_0^I + f_1^I + 2f^{O(next)-326(elemen)27(t)-326} \end{pmatrix}$$

To begin solving $K(u_t)$

$K(u)$ given by

$$K(u)\_ = \begin{pmatrix} I_0 \cot\ _0 & I_0 \cot\ _0 & 0 & & & 0 \\ I_0 \cot\ _0 & I_0 \cot\ _0 + h_1 & h_1 & 0 & & 0 \\ 0 & h_1 & h_1 + h_2 & h_2 & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & h_{i\ 1} & h_{i\ 1} + h_i & h_i \quad 0 & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots \quad \ddots & h_{N\ 1} \\ 0 & & & & h_{N\ 1} \quad h_{N\ 1} & \end{pmatrix}; \tag{4.45}$$

u: where

$$h_i = I_i \cot\ _i + O_i \cot\ _i; \tag{4.46}$$

fethnedimensiohl2 11.;hateduced,ethis2 1126(b)-7(ncom

$$I_i = \frac{R_i H_i^I}{2} \quad \frac{u_i + 2u_{i+1}}{3} \tag{4.47}$$

and

$$O_i = \frac{R_{i\ 1} H_i^O}{2} \quad \frac{2u_i + u_{i+1}}{3} \quad : \tag{4.48}$$

4.4 Considering p

It was stated earlier that $p = \ r^2 u$. As with the one dimensional case (in which $p = v$),

p will be solved using 2 11. i63c4it(e) 326eleimevd(.) 45(So2 11.;fah)1r2 11.;(one)-36(h(as)]TJ/F26 11.9

anath (eh) 93.;ievonen (as) (J/F32 11.9552 Tf 10 11.981 8.996 Td [20

(2)]TJ/F29 11.9552 Tf 13.113 0 Td [()]TJ/F32 11.9552 Tf 319551 60.226 Td [lu

u

Since the hat function is not 0 around the boundary, the second term on the right hand side is removed by weakly imposing $\underline{r}u = 0$ on the boundary, thus reducing the problem to

$$\int_{} {}_i p \, d = \int_{} \underline{r}_i \, \underline{r} u \, d \tag{4.50}$$

This reduces to the matrix form

$$M\underline{p} = K\underline{u};$$

where K is the standard nite elements sti ness matrix and M is a standard mass matrix. Therefore K is given by

$$K = \begin{pmatrix} \cot {}_0 & \cot {}_0 & 0 & & & & 0 \\ \cot {}_0 & \cot {}_0 + C_1 & C_1 & 0 & & & 0 \\ 0 & C_1 & C_1 + C_2 & C_2 & 0 & & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & & 0 \\ 0 & 0 & C_{i\ 1} & C_{i\ 1} + C_i & C_i & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & C_{N\ 1} \\ 0 & & & & C_{N\ 1} & C_{N\ 1} & \end{pmatrix}; \tag{4.51}$$

where

$$C_i = \cot {}_i + \cot {}_i \tag{4.52}$$

4.5 Considering The Mass Matrix

So far three di erent K matrices have been considered and assembled, but to complete the nite elements solution the M matrix must be assembled, as this is required for the solutions of , p and subsequently u. To assemble M, we shall consider it using the equation (4.16). From equation (4.16), we know that

$$_i = \int_{} {}_i u \, d$$

and therefore

$$\_ = M\underline{u}:$$

This is a standard finite element result, however the matrix produced does need to be assembled, since there are multiple different sizes of triangles to be considered. Generally an elemental matrix is given by

$$M = \frac{triangle\,area}{12}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}. \tag{4.53}$$

However, there are two different types of triangle and so a different area is considered for each one. F_i^I represents the area of the ith inward pointing triangle and F_i^O the ith outward. This produces two elemental matrices, such that

$$(M_i^I)_e u = \frac{F_i^I}{12}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}\begin{pmatrix} u_i \\ u_{i+1} \\ u_{i+1} \end{pmatrix};$$

which simplifies to

$$(M_i^I)_e u = \frac{F_i^I}{6}\begin{pmatrix} 1 & 1 \\ 1 & 3 \end{pmatrix}\begin{pmatrix} u_i \\ u_{i+1} \end{pmatrix}. \tag{4.54}$$

Similarly the inward element is given by

$$(M_i^O)_e u = \frac{F_i^O}{12}$$

Adding the next full element to this gives

$$Mu = \frac{1}{6}\begin{pmatrix} F_0^I & F_0^I & 0 \\ F_0^I & 3F_0^I + F_i^I + 3F_1^O & F_1 \\ 0 & F_1 & 3F_1^I + F_1^O \end{pmatrix}\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix}:$$

Adding a further element produces

$$Mu = \frac{1}{6}\begin{pmatrix} F_0^I & F_0^I & 0 & 0 \\ F_0^I & 3F_0^I + F_i^I + 3F_1^O & F_1 & 0 \\ 0 & F_1 & 3F_1^I + F_1^O + F_2^I + 3F_2^O & F_2 \\ 0 & 0 & F_2 & 3F_2^I + F_2^O \end{pmatrix}\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix}:$$

From this, one can see the general form of the matrix is

$$M = \frac{1}{6}\begin{pmatrix} F_0^I & F_0^I & 0 & & & & \\ F_0^I & 3F_0^I + F_1^I + 3F_1^O & F_1 & 0 & & & \\ 0 & F_1 & 3F_1^I + F_1^O + F_2^I + 3F_2^O & F_2 & 0 & & \\ & \ddots & \ddots & \ddots & & & \\ & 0 & F_{i\ 1} & 3F_{i\ 1}^I + F_{i\ 1}^O + F_i^I + 3F_i^O & F_i & 0 & \\ & & \ddots & \ddots & \ddots & & \\ & & & & F_{N\ 1} & 3F_{N\ 1}^I + F & \end{pmatrix}$$

(4.57)

From this, can be solved using a tri-diagonal solver applied to the system

$$\_ = M\underline{u};$$

where

$$u = \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_i \\ \vdots \\ u_N \end{pmatrix}$$

4.6 The Two Dimensional Solution

Now that all of the required variables have been de ned, a method for the solution of the problem can begin. As with the one dimensional case, a twice di erentiable function has been chosen and thus the u values can be immediately set. However, the u_t values are not so simple to set initially. In the one dimensional case the sine function being used could simply be di erentiated twice with respect to x. The initial function for u being used here is not so easy to di erentiate, as it contains both an x and a y term. Fortunately, the symmetry of this problem allows it to be considered radially and thus the initial condition can also be stated as $\frac{1}{10}\cos\ \ \frac{}{2}r$.

It was stated in the preliminary section of this paper that the r^2u term can be replaced with with

$$\frac{1}{r}\frac{@}{@r}\quad r\frac{@u}{@r}\quad : \tag{4.58}$$

Substituting in the radial initial condition for u gives

$$\frac{1}{r}\frac{@}{@r}\qquad \frac{r}{20}\sin\ \ \frac{}{2}r \tag{4.59}$$

Applying the product rule to this produces.

$$\frac{1}{r}\quad \frac{}{20}\sin\ \ \frac{}{2}r\quad +\ \frac{r^{\ 2}}{40}\sin\ \ \frac{}{2}r\quad : \tag{4.60}$$

Once in this form, it can then be seen that the initial u_t can be written as

$$u_t = f(u)\quad \frac{1}{a^2}\quad \frac{}{r20}\sin\ \ \frac{}{2}r\quad +\ \frac{^2}{40}\sin\ \ \frac{}{2}r\quad : \tag{4.61}$$

Once the initial conditions are set, both the initial , and c_i values can be calculated.

The boundary conditions however, are slightly di erent. For the one dimensional case both $x = 0$ and $x = 1$ had the condition $u = 0$ imposed upon them and therefore u_t was also equal to zero at the boundary. In this, two dimensional case, only the condition at the outer boundary of the circle (*r389 (a)]TJ/F26 11.9552 Tfni7d [(316the) - (ale) -4wi0 (*

estimate in u_t. Once this is known, _ can be calculated. Equation (4.15) only refers to each area between the rings on the circle. If equation (4.15) is summed from 0 to N, then the $K(u_t)$ term is removed since all the rows sum up to 0. Similarly, the $_i$ terms all sum to 1, thus reducing the equation to

$$_c = \int \frac{1}{a^2} r^2 \quad f(u) \quad \frac{p}{a^2} \quad + f^0(u) \quad f(u) \quad \frac{p}{a^2} \quad d \; ; \tag{4.62}$$

where c is simply the sum of all the c_i values.

Once _ is known, equation (4.40) can be solved and therefore the system $K(u_t)\_ = \underline{f}$ can be solved using a tri-diagonal solver.

This solution will nd and thus the solution of _ can begin found by considering equation (4.17). It has already been stated that this solution is explicit, since all of the right hand terms are known.

Once is known, v can be found since $\underline{v} = \underline{r} \_$. Once again the symmetry of the problem can be employed to simplify this calculation. Obviously $\underline{r} \_$ is made up of the two components $\frac{@}{@x}$ and $\frac{@}{@y}$, but exploiting the symmetry means that one of these terms is 0. If for example, we assume the radius we are considering is in fact the x axis, then $\frac{@}{@y} = 0$ and $\frac{@}{@x} = \frac{@}{@r}$. Therefore v can be found by simply considering a central nite di erence scheme on .

It should also be noted when calculating that K is singular and therefore cannot be inverted as it is. It is known that the outer ring and the origin will not shift and thus $\underline{r} \_ = 0$ at these points. However, this tells us little about the actual value of at these points. As such $_N$ is assumed to be zero so that the K matrix can be inverted.

With all of the above calculated, , $\underline{r}$ and $\underline{\_}$ can all be updated. All three are updated using a rst order Eulerian method. As such,

$$^{n+1} = \quad ^n + \quad t \_^n; \tag{4.63}$$

$$\underline{r}^{n+1} = \underline{r}^n + \quad t\underline{v}^n \tag{4.64}$$

and

$$\_^{n+1} = \_^n + \quad t \underline{\_}^n: \tag{4.65}$$

Now $\underline{u}$ can be calculated. Unlike the one dimensional method, this does not need to use any form of midpoint rule, as the nite elements formulation produced equation (4.18),

which suggests that

$$M\underline{u} = \_: \tag{4.66}$$

_ has already been updated and therefore this can be solved using a tri-diagonal solver, with the nal row and column of the M matrix ignored, since u is known to be 0 on the boundary.

With this time step e ectively over u_t can be updated. In some ways the solution for this more elegant than that of the one dimensional case. If one considers equation (4.7) and adds in the hat function then one has

$$c_i = \int {}_i u_t \, d \ :$$

When summed from $i = 0$ to N, the integral becomes the standard nite elements mass matrix and therefore

$$\underline{c} = M\underline{u}_t:$$

Clearly this is another system which can be solved using a tri-diagonal solver. However, the system is slightly di erent from the M given in an earlier section since u_t is known to be 0 on the boundary and therefore the last row and column of the M matrix can be ignored.

Following u_t is the recalculation of p (and subsequently the estimate for u_t using p) and then the re-triangulation of the domain using the new positions of the rings.

4.8 Critical Analysis

In the case of the two dimensional method it does appear that it is the program at fault, or at least this is the easiest assumption to make. When running the program, the _ value becomes very large and negative. Further exploration into this revealed that it is not the coding causing this outright. _ is calculated by applying the trapezium rule across the radius and then using the radial symmetry of the problem to generate the volume under the curve. So for example, the method starts by calculating the area under the curve between r_0 and r_1. This value is then multiplied by r_1^2. Therefore the volume of the region between the origin and the first ring is given. The second ring is then calculated by considering the area under the curve (along the radius) between r_0 and r_2 and then multiplying this by r^2

Chapter 5

Conclusions and Futher Work

5.1 Conclusions

This dissertation has covered the use of an r-adaptive method for solving a particular quenching problem for both one and two dimensions. Initial tests and papers by [7, 9] gave an indication as to where in the domain the problem would quench and at what point in time the quenching point should be reached.

Both one and two dimensional methods were attempted. The two dimensional method added little to the study other than to illustrate that the boundary conditions play a huge part in trying to calculate a solution if u_t is taken as the monitor function.

The one dimensional method on the other hand, did provide some results and thus

always largest at the centre of the domain. As such, this is the term contributing a great deal to the blow up, as while it gets larger itself, it also forces u_{xx} to grow larger at the centre of the domain.

However, the forcing of u and u_t to be zero at the boundaries also creates two other large values of u_{xx}. In terms of the solution, this is not actually an issue because if u is forced to zero at the boundaries, then u_{xx} and thus u_t will be reasonably large at the first few internal points. However, in terms of the moving mesh, this is an issue, because the method is attempting to cluster the points about three areas. Although it is not the u_t itself causing this issue, but the u_{tt} values which are required by the method. The gradient of u_t is very large near the boundaries, because of the forcing of u and u_t to zero, thus forcing the u_{xx} and _ values to become large and therefore forces the method to fail.

As such, the main conclusion which can be drawn from this study is that using u_t as a monitor function is incredibly dependent on the boundary conditions of the problem. Using $u = 0$ at the boundaries creates such large changes in u_t that the method soon becomes unstable. What this study does allow though, is to consider further work in this area.

5.2 Further Work

method, could improve the way the mesh moves. A scheme considered during this study would have centred around applying a universal change in time step should the change under u_t become too great. This would be crude, but being so simple it would only take a small amount of work to implement it. Equally, it would be possible to apply a much more complex method, similar to that of [9] to the adaptive time stepping.

Also, using u_t as monitor contains two terms causing growth. u_{xx} is large nearest the boundaries, due to the drop o to 0, but $f(u)$ is largest in the centre of the domain. This suggests that u_t is quite large at three di erent points, which could be forcing the mesh to shift poorly. If the problem being solved is one of the form of (1.1), perhaps using $f(u)$ as the monitor function would prove more e ective.

It was also noted during this dissertation that the use of two di erent methods for calculating u_t

Bibliography

[1] M. J. Baines, M. E. Hubbard and P. K. Jimack, *A Moving Mesh Finite Element Algorithm for the Adaptive Solution of Time-Dependent Partial Di erential Equations with Moving Boundaries*, Applied Numerical Mathematics (2005), 54 (3-4). pp. 450-469.

[2] C. J. Budd, W. Huang and R. D. Russell, *Adaptivity with moving grids*, Acta Numerica (2009), pp. 1-131.

[3] W. Cao, W. Huang and R. D. Russell, *A Moving Mesh Method Based On The Geometric Conservation Law*, SIAM J. Sci. Comput., Vol. 24, No. 1 pp. 118-142.

[4] C. Y. Chan, *New results in quenching*, Proc. 1st World Congress Nonlinear Anal., de Gruyeter, Berlin, 1996, 427-434.

[5] C. Y. Chan and Lan Ke, *Beyond quenching for singular reaction-di usion problem*, Mathematical Methods in the Applied Sciences, 17(1994), 1-9.

[6] S. L. Cole, *Blow-up in a Chemotaxis Model Using a Moving Mesh Method*, Reading University, Dissertation, 2009.

[7] K. Deng and H. Levine,

[10] G.D. Smith, *Numerical Solution of Partial Di erential Equations: Finite Di erence Methods*, Third Edition, Oxford University Press, ISBN 0-19-859650-2

Chapter 6

Appendices

6.1 Appendix 1

6.1.1 Pseudo Code for the One Dimensional Adaptive Method

Parameters are set (N, x, t).

The x positions are set.

Initial u values ($u = \frac{1}{10}sin(\ x)$) are set.

Initial v values ($v = \frac{2}{10}sin(\ x)$) are set.

Initial u_t values ($u_t = f(u)\ \ \frac{v}{a^2}$) are set.

A_i values are calculated using a combination of the exact integral of u_{xx} and the trapezium rule.

is calculated by summing the A_i values.

C_i values are calculated by dividing each A_i value by .

The initial values are calculated.

The time step is moved forward to t and the time loop begins.

{ _ is calculated.

{ is updated.

- The $\underline{x}$ values are calculated.
- The x positions are updated.
- The $\bar{}$ values are calculated.
- The values are updated.
- The u values are updated.
- The u_t values are calculated using the midpoint method.
- The v values are calculated.
- The v estimate of u_t is calculated using $vu_t = f(u) \quad \frac{v}{a^2}$.
- The time step is advanced and the loop begins again.

6.2 Appendix 2

6.2.1 Pseudo Code for the Two Dimensional Adaptive Method

Parameters are set (N, x, t).

The r positions are set.

The initial u values ($u = \frac{1}{10}\cos \frac{}{2}r$ are set.

The initial p values are set by approximating r^2u using a radial version of the derivative.

The initial u_t values are calculated using $u_t = f(u)$ $\frac{p}{a^2}$.

The initial c values are calculated.

The triangulation of the domain takes place.

The initial values are calculated.

The time loop begins.

- { _ is calculated.
- { The values and subsequently the v values are calculated.
- { and the r positions are updated.
- { The _ values are calculated.
- { The values are updated.
- { The u values are updated.
- { The p values are updated.
- { The approximation of u_t using p ($pu_t = f(u)$ $\frac{p}{a^2}$) is calculated.
- { The u_t values are Tf 9.777 0 bf{327(uping)-426(the)].9552 Tf