A Moving Mesh Finite Element Method And Its Application To Population Dynamics

Anna Watkins

Department of Mathematics

Reading University

This dissertation is submitted for the degree of

Doctor of Philosophy

June 2017

Declaration

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

Anna Watkins
June 2017

Acknowledgements

There are many who deserve acknowledgement for their part in the completion of this thesis. The work here must be one of the longest running PhD theses of all time at 11 years, and has been completed in part-time bursts whenever life and time allowed. The beginning was as a new graduate in a rented flat, coding on my first laptop. During the course of the work, there have been two house moves, a wedding, a house renovation, a career as an athlete spanning two Olympic games, a start-up business, a new career in analytics, and two children. The work was done on planes, buses and commuter trains, at home with a baby in a sling, in any number of hotel rooms whilst on training camp and and even greater number of coffee shops, and even occasionally in the department at Reading. Therefore the support of those around me has been fundamental.

I'm very grateful to Paul Thompson, my rowing coach, who believed that space needs to be made for things like this in an athletic career. This attitude meant that I have been able to cope with retirement from sport much better than I would have otherwise. I'm also grateful to Reading University Boat Club, who gave me a sports scholarship, and to UK Sport, who gave me an education grant, between which I was able to make ends meet in the early days when money was tight. I'm grateful to my parents for the encouragement they gave. I'd also like to thank Paul Glaister and Peter Grindrod for their generous time and friendly support, in supervising me and sharing ideas and research.

There are certain people I'd like to thank simply for putting up with me. In this group are my rowing partners, particularly Annie Vernon, Elise Sherwell and Katherine Grainger, who put up with me ignoring them as we shared rooms whilst training, and even as I zoned out to think about maths in the middle of a training session in the boat. The other two important people who have put up with me are my sons William and Richard, aged 2 and 3 now. I'm sorry I was tired and busy and that you weren't allowed to mash the keys on the keyboard or watch train videos on the screen.

However I do have two very major acknowledgements to share. I'm incredibly grateful to my main supervisor Mike Baines. Mike and I have always spent our time together with equal time given to maths and life in general. Mike has heard and advised on the trials and tribulations of my rowing career and motherhood as well as the ups and downs of research. His relentless support, friendly hello and interest in discussing any and every challenge in life means that I will count him as a lifelong friend. I always looked forward to our time, even when things weren't going well. I'm sorry Mike that I took so long and was often absent for long periods.

Finally I would like to thank my wonderful husband Oliver. His contributions to this work are too numerous to detail, but stretch comprehensively across all three of the financial, practical and emotional ranges. Thanks for paying for our flat when we first started out. Thank you for answering with good humour yet another late night LaTeXquestion delivered in a bad mood. Thanks for teaching me coding standards and debugging. Thanks for working out ways to make it all happen. Thanks for being my rock every day, a brilliant dad and and my sounding board for life. I could never have done any of it at all without you.

Anna Watkins

Abstract

The moving mesh finite element method (MMFEM) is a highly useful tool for the numerical solution of partial differential equations. In particular, for reaction-diffusion equations and multi-phase equations, the method provides the ability to track features of interest such as blow-up, the ability to track a free boundary, and the ability to model a dynamic interface between phases. This is achieved through a geometric conservation approach, whereby the integral of a suitable quantity is constant within a given patch of elements, but the footprint and location of those elements are dynamic. We apply the MMFEM to a variety of systems, including for the first time to various forms of the Lotka-Volterra competition equations. We derive a Lotka-Volterra based reaction-diffusion-aggregation system with two phases, representing spatially segregated species separated by a competitive interface. We model this system using the MMFEM, conserving an integral of population density within each patch of elements. We demonstrate its feasibility as a tool for ecological studies.

Table of contents

Chapter 1

Introduction

In a great many areas of study, partial differential equations (PDEs) are used to describe models, laws and systems. From the simplest of examples, the equations governing heat transfer, through to trading models for global financial markets, the PDE gives us an approach that can tackle a vast and ever-growing range of real-world problems. We may understand and make predictions about the behaviour of complex mechanical systems, we may study the weather, or we may gain insights into biological systems. The scope of PDEs and their relevance to our lives is beyond doubt. In many of these systems we have very complex interactions for which analytical solutions are not practicable or even possible. Direct experimentation and measurement may likewise not be practical and is generally expensive. Numerical modelling is therefore the key tool to unlock our understanding of how these systems are working or how they might evolve in time. Techniques for doing so are well established and are subject to continual refinement and improvement. One particular modelling technique, the use of finite elements, has plenty to recommend it. It involves dividing the domain into small discrete elements, and calculating the effect of each part upon its neighbours. In doing so an approximation to the whole system is produced. The size and spacing of these elements can be chosen to particularly suit the shape or dynamics of the domain, and is specified by a grid or mesh. The mesh may be uniformly distributed or otherwise. In the particular case of time dependent PDEs, there may be advantages to having a mesh that moves with time, so that features of interest may be tracked with accuracy without the computational expense of increasing the resolution everywhere. For certain phenomena such as boundary layers, interior moving interfaces and blow-up problems this can be especially true. This is the field of moving mesh finite element modelling, and this field is the subject of this thesis.

1.1 Mesh adaptation

In the body of work concerning mesh adaptation, there are three basic approaches which are usually given the following names:

h-refinement is the insertion of extra mesh points around an area of interest;

p-refinement is the use of a higher-order polynomial in each interval between mesh points, so that values between mesh points are better approximated;

r-refinement is the dynamic movement of existing mesh points to track a feature of interest.

Most commonly, h-refinement and p-refinement techniques are used and are often combined together. Their strength is that the algorithms produced are versatile; they do not need to utilise any particular dynamic properties of the underlying solution. This is also a weakness, since the dynamic properties of the solution can be an excellent guide to the most efficient mesh adaptations.

In r-refinement, the mesh nodes are assigned a velocity at each time step. This approach naturally lends itself to the solving of time dependent systems, as the time integration for the mesh movement and the solution evolution can be performed alongside one another, using any chosen integration scheme. Also, the node velocity can be chosen to work with useful properties of the system; for example one might wish to conserve mass within each element. Taking advantage of this sort of property means that, if our scheme is well chosen, the mesh evolves to reflect the solution in an efficient and elegant way. The nodes move smoothly along with the solution. We do not need to add or remove nodes, and we do not need to interpolate the solution between nodes. The node positions and the solution are completely linked. An excellent summary of the theory and practice of r-refinement techniques can be found in Huang and Russell's book [49].

1.2 Scope of work

In this thesis, we consider in particular the application of one r-refinement technique. The technique of interest is termed the moving mesh finite element method (MMFEM). This method was developed in 2005 by Baines, Hubbard and Jimack [5], and uses a geometric conservation approach to generate mesh adaptation. A finite element construction provides the framework. We apply this method to a variety of reaction-diffusion PDE systems. We have a particular focus on multi-phase systems, where a dynamic interface exists between phases. The MMFEM has previously been applied to the Stefan problem [8] where the dynamic interface represents the melting of ice into water. We extend this work with a simplified method. We then consider the application of the MMFEM to models of population dynamics. We take a version of the Lotka-Volterra competition model that, like Stefan, describes a two-phase reaction-diffusion system, and implement the MMFEM for this system. We then consider the application of the MMFEM to systems of intraspecies and interspecies interactions with aggregating dynamics. Finally, we present a new model for interspecies reactions that permits a dynamic interface combined with aggregating dynamics, as well as the more familiar reaction-diffusion dynamics. We implement the MMFEM for this model in chapter 7, and demonstrate its utility.

1.3 Novel material

This thesis contains the following novel material

- An application of the equidistribution method to a vertical water column under wind shear;
- A two dimensional MMFEM implementation for the Fisher's equation for the first time;
- A two dimensional MMFEM implementation for the Keller-Segel model for the first time;
- The first numerical model of the two phase Lotka Volterra competition system derived by Hilhorst *et al.* [31]. We use the MMFEM to achieve this;
- A novel approach to generating an interface velocity between phases for the competitive Lotka-Volterra system.
- A MMFEM model for single species population aggregation in one dimension;
- A MMFEM model for single species population aggregation in two dimensions, which is also the first 2-D numerical model of the aggregation proposed by Grindrod [29];
- A finite element model of the Lotka-Volterra competition equations in 2-D;
- A new multi-phase aggregation-reaction-diffusion model for Lotka-Volterra competition, and its implementation using the MMFEM in both 1 and 2-D.

Chapter 2

Technical background

In this thesis we apply a moving mesh finite element method to a variety of systems, with a particular focus on population dynamics. Here we set out the historical evolution of moving mesh methods, and also a history of PDE systems for population modelling.

2.1 Moving mesh methods

In moving the mesh, we have two fundamentally different approaches. We may use a system that provides a mapping to move the nodes at each time step in a fixed, Eulerian frame, or we may construct the entire system in a Lagrangian, or moving, co-ordinate system. Following [18], we will call these location-based, and velocity-based methods, respectively. An overview of these methods is given here. For a more detailed summary, the 2009 paper by Budd, Huang and Russell [15] is recommended.

2.1.1 Location-based methods

The common feature of this class of methods is that the location of the mesh nodes at a particular time step is directly controlled by a mapping function. The principle most often used to achieve this is equidistribution. Equidistribution is a term used to describe the locating of points such that a particular monitor function, for example arc length, is the same for all intervals between nodes. This is achieved either directly, or by defining the mapping as the minimiser of a functional. In one dimension, consider the case of an adaptive mapping $x(\xi,t)$ from a computational domain Ω_c to a physical domain Ω. If we are using a uniform computational mesh then $\frac{\partial \xi}{\partial x}$ is the density of the mesh on Ω. We then choose a monitor function $M(x) > 0$ and require the mesh density to be proportional to it,

$$\frac{\partial \xi}{\partial x} = c\, M(x). \tag{2.1}$$

The equivalence to a functional approach is apparent if we take the quadratic functional:

$$I[\xi] = \int_{\Omega} [M(x)]^{-1} \left(\frac{\partial \xi}{\partial x} \right)^2 dx \tag{2.2}$$

for which the corresponding Euler-Lagrange equation is:

$$\frac{\partial}{\partial x} \left([M(x)]^{-1} \frac{\partial \xi}{\partial x} \right) = 0 \tag{2.3}$$

which is the same as dividing (2.1) by $M(x)$ and differentiating, and can be solved with a given M to give ξ in terms of x. The functional approach is useful as it is comparatively easily extended to higher dimensions.

An early example of the use of equidistribution is given by White [52]. He uses the integral version of the equidistribution principle (2.1) which is, in continuous form:

$$\int_0^{x(\xi,t)} M(x(\xi,t),t)dx = \xi \int_0^1 M(x(\xi,t),t)dx \qquad \forall t. \tag{2.4}$$

If this is differentiated with respect to ξ we obtain

$$M(x(\xi,t),t)\frac{\partial}{\partial \xi} x(\xi,t) = \theta(t) \tag{2.5}$$

where

$$\theta(t) = \int_0^1 M(x(\xi,t),t)dx$$

and differentiating with respect to ξ again gives

$$\frac{\partial}{\partial \xi} \left(M(x(\xi,t),t) \frac{\partial}{\partial \xi} x(\xi,t) \right) = 0. \tag{2.6}$$

This will generally be nonlinear and so has been solved using an iterative approach by Baines [3]. We use this approach in Chapter 4, where we use an arc length monitor function to update the node spacing for a water column model with coriolis forces.

2.1.2 Moving mesh partial differential equations (MMPDEs)

It is recommended by Huang, Ren and Russell in their 1994 paper [32] to choose a method that generates moving mesh equations in a continuous form. A simple algorithm is also very desirable. This is achieved in their work by constructing moving mesh partial differential equations (MMPDEs) directly from an equidistribution principle. This is a neat and elegant construction that avoids having to consider user-defined input parameters in the mesh mapping. In taking this approach a more stable and more general algorithm can be produced. A simple example is given here. Huang *et al.* derive a MMPDE by differentiating (2.6) with respect to time to give

$$\frac{d}{dt}\left(\frac{\partial}{\partial\xi}\left(M(x(\xi,t),t)\frac{\partial}{\partial\xi}x(\xi,t)\right)\right)=0 \tag{2.7}$$

which can be rearranged to give the MMPDE

$$\frac{\partial}{\partial\xi}\left(M\frac{\partial\dot{x}}{\partial\xi}\right)+\frac{\partial}{\partial\xi}\left(\frac{\partial M}{\partial\xi}\dot{x}\right)=-\frac{\partial}{\partial\xi}\left(\frac{\partial M}{\partial t}\frac{\partial x}{\partial\xi}\right) \tag{2.8}$$

where $\dot{x}(\xi,t)$ is the mesh velocity. A great variety of MMPDEs exist, which vary in their approach to temporal and spatial smoothing and regularisation. The power of selecting the right one was demonstrated by Budd *et al.* in 1996 [14]. They took an MMPDE from a 1986 paper [1] and applied it to a blow up problem. The MMPDE they used was derived from (2.7) using temporal smoothing and is

$$\frac{\partial^2\dot{x}}{\partial\xi^2}=-\frac{1}{r}\frac{\partial}{\partial\xi}\left(M\frac{\partial x}{\partial\xi}\right) \tag{2.9}$$

where r is a small relaxation time after which the mesh is to reach equidistribution. This form has scale invariance properties. Here it is demonstrated that the use of monitor functions which incorporate such key properties of the original PDE can be particularly useful, as they allow features such as scaling invariance to be preserved. Natural spatial features of the PDE are inherited by the MMPDE. In this paper, self-similar or approximately self-similar solutions of blow-up equations are shown to be successful.

Another key concept was introduced by Budd and Williams in their 2006 paper [16]. They solve a relaxed form of the Monge–Ampere equation to compute a transformation from a regular (computational) to the desired spatially non-uniform mesh. The method involves the creation of a mesh potential which determines the location of the mesh points. Using the Legendre transformation, the equidistribution principle is transformed into the

Monge–Ampere equation giving the mesh potential.

2.1.3 Velocity-based methods

The following velocity based methods make use of the Arbitrary Langrangian Eulerian (ALE) form of the PDE; that is to say that a moving co-ordinate system is used to directly provide the mesh velocity. The form provides a mapping from the fixed to the moving frame. Consider the time dependent PDE

$$\frac{\partial u}{\partial t} = \mathscr{L}u, \tag{2.10}$$

where $u = u(\mathbf{x},t)$ is defined in a fixed (Eulerian) reference frame, and $\mathscr{L}$ is a differential operator involving only space derivatives. To rewrite this in a moving (Lagrangian) frame, we allow $\mathbf{x}$ to be a moving co-ordinate $\mathbf{x}(t)$, which is related to a set of reference co-ordinates $\mathbf{a}$ by the invertible mapping

$$\mathbf{x} = \hat{\mathbf{x}}(\mathbf{a},t) \tag{2.11}$$

where the hat denotes a mapping from the Eulerian frame to the moving frame. We can then define the solution $u(\mathbf{x},t)$ in the moving frame:

$$u(\mathbf{x},t) = u(\hat{\mathbf{x}}(\mathbf{a},t),t) = \hat{u}(\mathbf{a},t) \tag{2.12}$$

and then by the chain rule

$$\frac{\partial \hat{u}}{\partial t} = \frac{\partial \hat{\mathbf{x}}}{\partial t} \cdot \nabla u + \frac{\partial u}{\partial t} \tag{2.13}$$

where we clarify that

$$\dot{u} = \frac{\partial \hat{u}}{\partial t}, \dot{\mathbf{x}} = \frac{\partial \hat{\mathbf{x}}}{\partial t}. \tag{2.14}$$

The ALE form of the PDE is then

$$\dot{u} - \dot{\mathbf{x}} \cdot \nabla u = \mathscr{L}u. \tag{2.15}$$

There are now two unknowns, $\dot{u}$ and $\dot{\mathbf{x}}$, so we must know the mesh velocities before we are able to find the solution. The specific method for constructing these velocities varies from using a real physical motion that provides a natural reference frame, through to defining the motion with the sole aim of optimizing geometric properties of the mesh. The velocity can be defined in any way that is helpful or suitable to the system. The various approaches are outlined here.

Moving finite elements

The MFE method of Miller and Miller, [38] and [39], involves taking the PDE (2.10) and determining the solution and the mesh simultaneously. This is achieved by minimising a discrete residual of the ALE form of the PDE (2.15) in a moving frame. Miller and Miller made the first attempts at a moving mesh of finite elements to deal with a model involving a sharp transition layer. These attempts made use of Burgers' equation as a test equation and had some success in having nodes automatically concentrate in the critical regions. This pioneering work inevitably led to the discovery of several common difficulties with the moving mesh approach, such as instabilities arising from nodes colliding or crossing, and high sensitivities to particular user-defined parameters and unknowns. Having moving piecewise linear functions approximating smooth functions produced an inherent singularity in certain cases, where a node velocity on a smooth section would be unconstrained. The improvements that were attempted included introducing viscous forces to encourage nodes to stay separate, and then in [39] replacing this with a variety of short and long range internodal repellant forces. Whilst stability remained a problem and computation efficiency was not yet demonstrated, these early models did at least have a mesh with nodes that followed a moving shock and allowed an improved resolution of the form of the shock when compared with a static grid. A comprehensive summary of the work in this area can be found in Baines' book 'Moving Finite Elements' [2]. Carlson and Miller in [19] and [20] provide some further suggestions for improvements using gradient weighted finite elements, which reduce the sensitivity of the system. The use of smoothing for second order terms and the introduction of small diffusion terms are suggested as well as the addition and removal of nodes at intervals of several ordinary time steps.

The Geometric Conservation Law

In the 2002 paper [17], Cao, Huang and Russell introduce the concept of the Geometric Conservation Law (GCL) to MMPDEs. A variational principle is used to construct a minimisation problem that must be solved to find the mesh velocities. This is a method that seeks to preserve properties associated with the volume of each element, so that, for example, a moving fluid could not lose or generate mass by mesh movement alone. This is achieved through enforcing the conservation of the integral of a suitable monitor function across a mesh interval. The advantages associated with this new method are that the degree of mesh adaptation is easy to control, and there are theoretical stabilities inherent in the method that prevent a singular or non-existent co-ordinate transform. A particularly helpful innovation in the paper is the use of a mesh velocity potential in the calculation of grid velocities; this can make the finite element formulation better conditioned as certain asymmetric matrices can be substituted out. Furthermore, velocities in two or more space dimensions can be uniquely calculated from it. The mesh velocity potential idea is extensively used in this field after this publication.

The conservation method

The 2005 paper by Baines, Hubbard and Jimack [5] takes the GCL concept and firmly establishes it from a finite element perspective. This method shares common roots with the GCL, but instead of using the variational principle to find the mesh velocities they are directly calculated from the integral form of the PDE. This is achieved by taking a weak form of the PDE that includes a set of weight functions that move with the mesh. Then the Reynolds Transport Theorem is used to provide a link between the Eulerian and Lagrangian perspectives. A system is constructed where the mesh velocity is given in terms of a potential at a particular location (Eulerian view), but the elements themselves track the movement of mass (Lagrangian view). This gives rise to the Arbitrary Lagrangian Eulerian (ALE) equation, where a single equation ties together the relationship between the moving and static reference frames. The examples demonstrated each conserve a proportion of a quantity within each patch of elements. This may be mass itself for systems where mass is conserved overall, in which case the simplest form of the theory can be used. This is demonstrated for the porous medium equation and a fourth-order nonlinear diffusion equation. For non-conservative systems, the theory uses the concept of relative mass, the proportion of total mass associated with each element patch, and this is applied to a Stefan problem and a diffusion problem with a negative source term. The method is extended in their 2006 paper [7] to include the solution of scale invariant PDEs. This exploits the inherent independence of physical systems from any given unit system. Again using moving mesh finite element systems, the time stepping is coupled to the mesh resolution, resulting in a scheme that provides uniform local accuracy in time. This exploitation of scale invariance is not an option for fixed mesh models since they are time-independent and therefore cannot exploit the coupling of dependent and independent variables in time.

2.1.4 Monitor functions

The choice of a suitable monitor function is of course key. The choice will be influenced by the underlying physics of the system as well as the moving mesh method itself. There are three classes of construction:

- An estimate of a quantity related to the solution such as arc length or mass, that can be made at the prior time step;
- An estimate of the error at each node or across each element, which can then be corrected by a suitable mesh adjustment. This is the approach used in moving finite element methods, where the mesh movement is determined by the velocity term in an ALE equation;
- A feature of underlying physics which is advantageous, often because of scale invariant properties. An example would be potential vorticity in a meteorological problem.

2.2 Population Dynamics

The time-dependent interactions between species are of great interest to ecologists. The best known set of equations, the predator-prey Lotka-Volterra equations, were first derived in the 1920s. Alfred Lotka [37] and Vito Volterra [47] independently derived a pair of equations which describe the interaction and self-interaction of a predator-prey pairing. These equations were famously used to model cyclical interactions between Canadian lynx and snowshoe hares [26]. For a prey species u_1 and a predator u_2,

$$\begin{aligned} \frac{du_1}{dt} &= \alpha u_1 - \beta u_1 u_2 \\ \frac{du_2}{dt} &= \delta u_1 u_2 - \gamma u_2 \end{aligned} \tag{2.16}$$

where all parameters are positive and real.

The form of the Lotka–Volterra equations for interspecies competition [48] is similar to the form for predation in that the equation for each species has one term for self-interaction and one term for the interaction with other species. However, in the equations for predation, the base population model is exponential, whilst for the competition equations, both species have a logistic equation as the base. The competition equations are

$$\begin{aligned} \frac{du_1}{dt} &= r_1 u_1 \left(1 - \left(\frac{u_1 + K_1 u_2}{k_1}\right)\right) \\ \frac{du_2}{dt} &= r_2 u_2 \left(1 - \left(\frac{u_2 + K_2 u_1}{k_2}\right)\right) \end{aligned} \tag{2.17}$$

where k_1 and k_2 are the carrying capacities of species 1 and 2 respectively, K_1 is a measure of the effect that species 1 has on species 2, and K_2 is a measure of the effect species 2 has on species 1. The parameters r_1 and r_2 are a measure of the timescales upon which births and deaths operate.

These early sets of equations did not consider spatial effects, so an important development was made by Conway and Smoller in 1977 [22], where a diffusion term was included along with spatial dependence. This allowed the study of a vastly increased range of phenomena, such as the geographic spread of invasive species, or of disease, or the effect of non-homogenous resource distribution. When random motion of the individuals is considered in the form of a diffusion term, the Lotka-Volterra equations are of reaction-diffusion form. We have

$$\begin{aligned}\frac{\partial u_1}{\partial t} &= \delta_1 \nabla^2 u_1 + f(u_1,u_2)u_1 \\ \frac{\partial u_2}{\partial t} &= \delta_1 \nabla^2 u_2 + g(u_1,u_2)u_2 \end{aligned} \tag{2.18}$$

where δ_1, δ_2 are constant diffusion coefficients, and with $f(u_1,u_2)$ and $g(u_1,u_2)$ given by the logistic equations

$$\begin{aligned} f(u_1,u_2) &= r_1\left(1-\frac{u_1-K_1u_2}{k_1}\right) \\ g(u_1,u_2) &= r_2\left(1-\frac{u_2-K_2u_1}{k_2}\right). \end{aligned} \tag{2.19}$$

It is this set of equations which is of interest to us here.

Chapter 3

The MMFEM and existing applications

3.1 The moving mesh finite element method

The conservation method of Baines, Hubbard and Jimack [5] can be implemented from either a finite difference or finite element perspective. Using the finite element method can be more computationally expensive than the finite difference method, but can be more easily extended to higher dimensions, and depending on the system, more stable. Furthermore, finite elements lend themselves well to being applied to complex geometries, although that is beyond the scope of this work. In this thesis we will take a finite element approach. This approach is termed the Moving Mesh Finite Element Method (MMFEM), and is the foundation of the methods in this thesis. The mass conservation concept involves assigning a local proportion of mass to a patch of elements surrounding a particular mesh node. The mass is assigned from the initial data and conditions, and then remains constant with respect to time as the solution evolves. In this way, parts of the solution with increasing density will also have increasing mesh resolution. It is not necessary to have a system with constant total mass as a property, since there is a version of the method that instead uses a system of relative proportions of mass or density. Furthermore, it is not necessary to be working with mass or density in a physical sense. Other quantities, such as volume, concentration or temperature, are appropriate alternative choices. The MMFEM, in common with standard static finite element methods, requires that the PDE of interest is written in a weak, integral form, along with the other relationships necessary for solution. At that point finite element substitutions can be made and the system can then be solved.

The finite element method can be traced back to approaches to civil engineering problems, with an early form of the method published by Courant in 1943 [23]. The method was developed from an engineering perspective during the 1950s within the aerospace industry, and although many contributors could be acknowledged a key individual was certainly M.J. Turner at Boeing. His 1956 paper [46] contains the basis of the method. A rigorous mathematical basis was provided in 1973 by Strang and Fix [45]. Modern reference textbooks include those of J.N.Reddy [41], and Brenner and Scott, [12] which both outline the modern forms and techniques for the standard, static finite element method. For semilinear forms, Larsson's 1994 paper [35] introduced the approach, and a second textbook by J.N.Reddy [42] outlines the modern methodology. The key steps of the moving (MMFEM) approach are outlined here.

3.1.1 Generating the weak forms of the PDE and associated equations

Our example will be a time dependent PDE of the form

$$\frac{\partial u}{\partial t} = Lu \qquad \mathbf{x} \in \Omega(t) \tag{3.1}$$

where $u = u(\mathbf{x},t)$ is defined in a fixed (Eulerian) reference frame, and L is a differential operator involving only space derivatives. The boundary conditions must be consistent over time, and the initial conditions must be known. The variable we seek to solve for, $u = u(\mathbf{x},t)$, may represent any of a number of physical quantities. For the PDEs examined in this thesis, u is most commonly a measurement of density. When the integral of u is conserved over a particular domain for a particular PDE, we will say that it is a 'mass conserving problem'. We will use the term 'mass' to refer to u throughout this thesis to allow for comparability between PDEs, even though for certain PDEs a different physical quantity is actually the subject of the equation. The boundary conditions will typically include given normal fluxes $u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}$, where $\hat{\mathbf{n}}$ is the outward pointing normal to the boundary.

Mass conserving problems

Define a reference test domain $\Omega(0)$ at $t = 0$ and a moving test volume $\Omega(t)$ in the moving frame with co-ordinate $\mathbf{x}(t)$ and boundary $S(t)$. Mass is conserved so we can write

$$\frac{d}{dt} \int_{\Omega(t)} u \, d\Omega = 0. \tag{3.2}$$

The Reynolds Transport Theorem, derived in the original 1903 book by Reynolds [43] is

$$\frac{d}{dt} \int_{\Omega(t)} f d\Omega = \int_{\Omega(t)} \frac{\partial f}{\partial t} d\Omega + \int_{S(t)} (\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}) f \, dS \tag{3.3}$$

which, when applied to our mass conserving system, becomes

$$\int_{\Omega(t)} \frac{\partial u}{\partial t} d\Omega + \int_{S(t)} u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \; dS = 0 \tag{3.4}$$

which can be written using the Divergence Theorem as

$$\int_{\Omega(t)} \left(\frac{\partial u}{\partial t} + \nabla \cdot (u\dot{\mathbf{x}}) \right) d\Omega = 0 \tag{3.5}$$

where $\dot{\mathbf{x}}$ is any velocity field consistent with the boundary velocities. In the moving frame then, our original PDE (3.1) can be written in integral form as

$$\int_{\Omega(t)} Lu \; d\Omega = -\int_{\Omega(t)} \nabla \cdot (u\dot{\mathbf{x}}) d\Omega. \tag{3.6}$$

The PDE can be generalised to a suitable weak form. A weight function w_i is introduced, which moves with velocity $\dot{\mathbf{x}}$. The function w_i is part of a set of functions that form a partition of unity, *i.e.* $\sum_i w_i = 1$. This will satisfy the advection equation

$$\frac{\partial w_i}{\partial t} + \dot{\mathbf{x}} \cdot \nabla w_i = 0. \tag{3.7}$$

We can write a generalised form of equation (3.2) as

$$\frac{d}{dt} \int_{\Omega(t)} w_i u \; d\Omega = 0 \tag{3.8}$$

which, using the Reynolds Transport Theorem (3.3) again, with $f = w_i u$, implies the generalised form of (3.5),

$$\int_{\Omega(t)} \frac{\partial}{\partial t}(w_i u) d\Omega + \int_{S(t)} w_i u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \; dS = 0 \tag{3.9}$$

leading to

$$\int_{\Omega(t)} \left(w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + \nabla \cdot (w_i u\dot{\mathbf{x}}) \right) d\Omega = 0 \tag{3.10}$$

or

$$\int_{\Omega(t)} \left(w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + w_i \nabla \cdot (u\dot{\mathbf{x}}) + u\dot{\mathbf{x}} \cdot \nabla w_i \right) d\Omega = 0. \tag{3.11}$$

Here, $\dot{\mathbf{x}}$ is any velocity field consistent with the boundary velocity. Using the advection equation (3.7) we can cancel out terms

$$\int_{\Omega(t)} \left(w_i \frac{\partial u}{\partial t} + w_i \nabla \cdot (u\dot{\mathbf{x}}) \right) d\Omega = 0 \tag{3.12}$$

giving us the weak form of a PDE in the moving frame,

$$-\int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}}) d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega \tag{3.13}$$

or, after integration by parts,

$$-\int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS + \int_{\Omega(t)} u \dot{\mathbf{x}} \cdot \nabla w_i d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega \tag{3.14}$$

where $\hat{\mathbf{n}}$ is the outward pointing unit normal. We assume that the boundary flux $u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}$ is given by boundary conditions that either (1) give both $\dot{\mathbf{x}}$ and u directly or (2) permit a free boundary with $\dot{\mathbf{x}}$ undefined, but have the condition that $u = 0$. We now have an equation for $\dot{\mathbf{x}}$ in terms of u.

A velocity potential is introduced. When we come to calculate the values of $\dot{\mathbf{x}}$ from equation (3.14), we will be working within a finite element framework. We can make a substitution here that turns (3.14) into a form that lends itself more naturally to a viable numerical method. The substitution will give a unique solution, as well as enabling us to work with symmetric matrices which make for a more straightforward and well conditioned algorithm. The substitution we will use is to define a velocity potential, ϕ,

$$\dot{\mathbf{x}} = \nabla \phi. \tag{3.15}$$

We take care to consider the implications of the Helmholtz decomposition [27] in the solution of (3.15). Because a vector field in three dimensions can be resolved into the sum of an irrotational (curl-free) vector field and a solenoidal (divergence-free) vector field, providing that either ϕ or $\nabla\phi.\hat{\mathbf{n}}$ is specified at each point along the boundary, a unique solution is defined. The choice of which to specify is determined by the boundary conditions of the system under consideration. Equation (3.14) can be rewritten as

$$\int_{\Omega(t)} u \nabla \phi \cdot \nabla w_i d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega + \int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \mathbf{n} \, dS \tag{3.16}$$

which can then be used to determine ϕ. The recovery of $\dot{\mathbf{x}}$ can then be made using the weak form of the definition (3.15) of ϕ,

$$\int_{\Omega(t)} w_i \dot{\mathbf{x}}\, d\Omega = \int_{\Omega(t)} w_i \nabla\phi d\Omega. \tag{3.17}$$

The Eulerian velocity $\dot{\mathbf{x}}$ is now known, and a moving reference frame can be generated. This can be considered as a deformation $\mathbf{x} \to \hat{\mathbf{x}}$ in time, derived from the ODE system

$$\frac{d\hat{\mathbf{x}}}{dt} = \dot{\mathbf{x}}(\hat{\mathbf{x}}, t) \tag{3.18}$$

with initial condition $\hat{\mathbf{x}} = \mathbf{x}$. Once $\hat{\mathbf{x}}$ has been found we can recover the solution from the mass conservation principle (3.8) in the form

$$\int_{\Omega(t)} w_i(\hat{\mathbf{x}}(t), t) u(\hat{\mathbf{x}}(t), t) d\Omega = \int_{\Omega(0)} w_i(\hat{\mathbf{x}}(0), 0) u(\hat{\mathbf{x}}(0), 0) d\Omega \tag{3.19}$$

at any later time t.

Algorithm 1

The solution of the mass conserving equation (3.1) on the moving mesh therefore consists of the following steps.
Given functions u and $\mathbf{x}$ initially, for each time t:

1. Find the velocity potential by solving equation (3.16) for $\phi(\mathbf{x}, t)$;
2. Find the node velocity by solving equation (3.17) for $\dot{\mathbf{x}}(t)$;
3. Generate the moving co-ordinate system at the next time-step by integrating (3.18) for $\hat{\mathbf{x}}(t + dt)$;
4. Find the solution $u(\hat{\mathbf{x}}(t + dt), t + dt)$ by solving equation (3.19).

Non mass-conserving problems

For the case where total mass is not conserved, the situation requires an additional variable.

A relative conservation principle is introduced. In order to derive the moving co-ordinate system, a principle is introduced whereby a normalised, or relative local mass is conserved. The total mass of the system is defined as

$$\theta(t) = \int_{\Omega(t)} u\, d\Omega \tag{3.20}$$

and the test volume $\Omega(t)$ is defined to be the total spatial domain of the model at time t, moving with velocity $\dot{\mathbf{x}}$. We introduce again our weight function w_i, also moving with velocity $\dot{\mathbf{x}}$, as in (3.7). Again we require that w_i is part of a set of functions that together form a partition of unity. We now define the moving co-ordinate system by requiring that the integral of u multiplied by that moving weight function is a constant proportion of the total mass in the system, *i.e.*

$$\int_{\Omega(t)} w_i u \, d\Omega = c_i \theta(t) \tag{3.21}$$

where the constant c_i is determined by the initial w_i and the initial data. Since $\sum_i w_i = 1$, it follows that $\sum_i c_i = 1$ also. Differentiating with respect to time gives

$$\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega = c_i \frac{d\theta}{dt} = c_i \dot{\theta}(t). \tag{3.22}$$

As in the case of conserved total mass, we define a reference test domain $\Omega(0)$ at $t = 0$ and a moving test volume $\Omega(t)$ in the moving frame $\mathbf{x}$. Applying the Reynolds Transport Theorem to $w_i u$, we obtain

$$\begin{aligned}\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega &= \int_{\Omega(t)} \frac{\partial}{\partial t}(w_i u) d\Omega + \int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS \\ &= \int_{\Omega(t)} \left(w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + \nabla \cdot (w_i u \dot{\mathbf{x}}) \right) d\Omega \end{aligned} \tag{3.23}$$

for the generalised weak form of the PDE. Using the advection equation (3.7) we can cancel out terms as before, giving us the weak form of the PDE in the moving frame,

$$\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}}) d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega. \tag{3.24}$$

We now use the relative conservation principle (3.21) to make a substitution. We use the weak form (3.22) to give

$$c_i \dot{\theta}(t) - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}}) d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega. \tag{3.25}$$

After integration by parts we obtain

$$c_i \dot{\theta}(t) + \int_{\Omega(t)} u\dot{\mathbf{x}} \cdot \nabla w_i d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega + \int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS. \tag{3.26}$$

The boundary flux $u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}$ is again assumed to be given by the boundary conditions. We now have an expression for $\dot{\mathbf{x}}$ in terms of u and $\dot{\theta}$. So long as we select weight functions w_i that form a partition of unity, $\sum_i w_i = 1$, we can calculate $\dot{\theta}$ by summing this expression over all weight functions in the model and using the boundary conditions. Recalling that $\sum_i c_i = 1$, we sum equation (3.26) over all i, to give

$$\sum_i c_i\dot{\theta}(t) + \sum_i \left(\int_{\Omega(t)} u\dot{\mathbf{x}} \cdot \nabla w_i d\Omega \right) = \sum_i \left(\int_{\Omega(t)} w_i L u \, d\Omega \right) + \sum_i \left(\int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS \right). \tag{3.27}$$

Noting that $\nabla \sum w_i = 0$,

$$\dot{\theta}(t) = \int_{\Omega(t)} Lu \, d\Omega + \int_{S(t)} u\dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS \tag{3.28}$$

will determine $\dot{\theta}$, providing that the boundary flux is indeed known.

A velocity potential is then introduced in the same way as for the conservative case. A velocity potential, ϕ is defined,

$$\dot{\mathbf{x}} = \nabla \phi \tag{3.29}$$

so that equation (3.26) can be rewritten as

$$c_i\dot{\theta}(t) + \int_{\Omega(t)} u \nabla \phi \cdot \nabla w_i d\Omega = \int_{\Omega(t)} w_i L u \, d\Omega + \int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS \tag{3.30}$$

and we are now able to uniquely determine ϕ in terms of u and $\dot{\theta}$, as long as ϕ is given at one point at least. As before, the recovery of $\dot{\mathbf{x}}$ is made using the weak form of the definition (3.29) of ϕ

$$\int_{\Omega(t)} w_i \dot{\mathbf{x}} \, d\Omega = \int_{\Omega(t)} w_i \nabla \phi d\Omega. \tag{3.31}$$

Having updated $\hat{\mathbf{x}}(t)$ from $\dot{\mathbf{x}}$ and $\theta(t)$ from $\dot{\theta}$ using a suitable integration procedure, we can now recover u from the relative conservation principle,

$$\frac{1}{\theta(t)} \int_{\Omega(t)} w_i(\hat{\mathbf{x}}(t), t) u(\hat{\mathbf{x}}(t), t) d\Omega = \frac{1}{\theta(0)} \int_{\Omega(0)} w_i(\hat{\mathbf{x}}(0), 0) u(\hat{\mathbf{x}}(0), 0) d\Omega. \tag{3.32}$$

Algorithm 2

The solution of the non mass conserving equation (3.1) on the moving domain therefore consists of the following steps. Given u and $\mathbf{x}$ at $t = 0$ and θ calculated from (3.20), for each time t:

1. Find $\dot{\theta}(t)$ from (3.28)

2. Find the velocity potential by solving equation (3.30) for $\phi(\mathbf{x},t)$;

3. Find the deformation velocity by solving equation (3.31) for $\dot{\mathbf{x}}(t)$;

4. Generate the moving co-ordinate $\hat{\mathbf{x}}$ at the next time-step $t+dt$ by integrating (3.18). Similarly, update $\theta(t+dt)$ from $\dot{\theta}(t)$;

5. Find the solution $u(\hat{\mathbf{x}}(t+dt),t+dt)$ by solving the relative conservation equation (3.32).

We now have all the key relationships expressed in weak forms, suitable for a finite element method of solution.

1-D forms

Before looking at the two dimensional finite element method, we will examine the one dimensional version. In order to do this we will need the analogous system of weak PDE equations in 1-D. The domain will be $x \in [a(t),b(t)]$, and we use $u = u(x,t)$. We begin with the 1-D definition for θ, analogous to (3.20)

$$\theta(t) = \int_{a(t)}^{b(t)} u\, dx. \tag{3.33}$$

Having introduced the weight function w_i, we require that a proportion c_i of total mass associated with that weight function is conserved. We recall equation (3.21), and write the 1-D analogy

$$\int_{a(t)}^{b(t)} w_i u\, dx = c_i \theta(t) \tag{3.34}$$

where the constant c_i is determined by the initial w_i and the initial data.

The section spanning equations (3.21) to (3.30) outlines the steps required to arrive at the 2-D weak form of the PDE (3.30). Taking the 1-D definitions (3.33) and (3.34) we follow the same steps to arrive at the analogous 1-D weak form of (3.30),

$$c_i\dot{\theta}(t) + \int_{a(t)}^{b(t)} u \frac{\partial \phi}{\partial x}\frac{\partial w_i}{\partial x} dx = \int_{a(t)}^{b(t)} w_i L u\, dx + [w_i u \dot{x}]_{a(t)}^{b(t)} \tag{3.35}$$

where the boundary flux $u\dot{x}$ is given by the boundary conditions and $\dot{\theta}$ is given by the 1-D form of (3.28),

$$\dot{\theta} = \int_{a(t)}^{b(t)} Lu\, dx + [u\dot{x}]_{a(t)}^{b(t)}. \tag{3.36}$$

The weak integral form defining the one dimensional velocity potential ϕ is

$$\int_{a(t)}^{b(t)} w_i \dot{x}\, dx = \int_{a(t)}^{b(t)} w_i \frac{\partial \phi}{\partial x} dx. \tag{3.37}$$

3.1.2 Introducing the MMFEM framework in one dimension

A good choice for the weight function w_i is a piecewise linear basis function, having the property that all the weight functions in the domain together form a partition of unity. For the second term of (3.35) we require that the weight function is once differentiable. However our weak formulation will mean that we only need to obtain the derivative between mesh points rather than at those points, and so a piecewise linear function fits this requirement. We can write the weak form (3.30) for each function in the domain and can then assemble the whole system in matrix form. All the instances of equation (3.35) can then be solved at the same time.

In one dimension, the finite element framework we use is as follows. We discretise the domain of the PDE on which we wish to solve into intervals, which may be regular or irregular depending on the initial conditions and/or prior knowledge of the solution. We call this closed domain $[a(t), b(t)]$ where $a(t)$ and $b(t)$ are moving boundaries in general. The interval partitions $x_i(t)$ will be known as nodes. For $i = 1, ..., N$,

$$[a(t) = x_0(t) < x_1(t) < ... < x_{i-1}(t) < x_i(t) < x_{i+1}(t) < ... < x_{N+1}(t) = b(t)] \tag{3.38}$$

We define the one-dimensional weight functions as piecewise linear functions:

$$W_i(x) = \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}} & x_{i-1} \le x \le x_i \\ \frac{x_{i+1} - x}{x_{i+1} - x_i} & x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

which are graphically shown in figure 3.1. At the domain edge boundary nodes, $x_0(t)$ and $x_{N+1}(t)$, half functions are used as shown in the diagram. Our weak form (3.35) now specif-

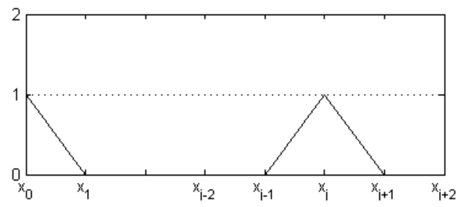


Fig. 3.1 Weight or basis functions for nodes x_0 and x_i

ically uses the weight functions $W_i(x)$ rather than the generic weight function w_i, so

$$c_i\dot{\theta}(t) - \left[W_i u \frac{\partial \phi}{\partial t}\right]_{a(t)}^{b(t)} + \int_{a(t)}^{b(t)} u \frac{\partial \phi}{\partial x}\frac{\partial W_i}{\partial x} dx = \int_{a(t)}^{b(t)} W_i Lu \, dx \qquad (i = 0, ..., N+1). \tag{3.39}$$

Now we can substitute finite element approximations Φ, $\dot{X}$ and U for ϕ, $\dot{x}$ and u respectively. These are also piecewise linear in form, and are linear combinations of basis functions $W_j(t)$. The basis functions $W_j(t)$ are often chosen to be the same set of functions as the weight functions $W_i(t)$, although this does not have to be the case. Here we will use the same definitions for $W_j(t)$ and $W_i(t)$ unless we specify otherwise. We will use the subscript i for weight functions, and the subscript j for basis functions. For example, the function $\Phi(x,t)$ is defined as

$$\Phi(x,t) = \sum_{j=0}^{N+1} \Phi_j(t) W_j(x,t). \tag{3.40}$$

In this formulation each of the $N+2$ nodes will have a coefficient Φ_j associated with it and (3.40) will form a linear spline. Similarly we also define

$$\dot{X}(x,t) = \sum_{j=0}^{N+1} \dot{X}_j(t) W_j(x,t) \tag{3.41}$$

and

$$U(x,t) = \sum_{j=0}^{N+1} U_j(t) W_j(x,t). \tag{3.42}$$

We can also use the result that

$$\frac{\partial \Phi}{\partial x} = \sum_{j=0}^{N+1} \Phi_j \frac{\partial W_j}{\partial x}. \tag{3.43}$$

These can now be substituted into equation (3.39) to give

$$c_i\dot{\theta}(t)+\sum_{j=0}^{N+1}\left[\int_{a(t)}^{b(t)} U\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]\Phi_j=\int_{a(t)}^{b(t)} W_iLU\ dx+[W_iU\dot{x}]_{a(t)}^{b(t)}. \tag{3.44}$$

Note that the term $\int_{a(t)}^{b(t)} W_iLU\ dx$ is not subject to an approximation or substitution for this general case. Its treatment will depend on the nature of the operator L. If the term cannot be converted into a linear form then it will be evaluated using quadrature rules.

We will also use the weight functions to evaluate the c_i. Each of the c_i is associated with a weight function W_i at $t=0$, $cf.$ equation (3.21)

$$c_i=\frac{1}{\theta}\int_a^b W_iU\ dx \tag{3.45}$$

at $t=0$. Then, since the basis functions are all compact functions, we can write (3.44) for each internal node $i=[1,...,N]$.

$$c_i\dot{\theta}(t)+\sum_{j=i-1}^{j=i+1}\left[\int_{x_{i-1}(t)}^{x_{i+1}(t)} U\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]\Phi_j=\int_{x_{i-1}(t)}^{x_{i+1}(t)} W_iLU\ dx \tag{3.46}$$

and for the boundary nodes $i=0$ and $i=N+1$

$$c_0\dot{\theta}(t)+\sum_{j=0}^{j=1}\left[\int_{x_0(t)}^{x_1(t)} U\frac{\partial W_j}{\partial x}\frac{\partial W_0}{\partial x}dx\right]\Phi_j=\int_{x_0(t)}^{x_1(t)} W_iLU\ dx-W_0U_0\dot{x}|_{a(t)} \tag{3.47}$$

$$c_{N+1}\dot{\theta}(t)+\sum_{j=0}^{j=1}\left[\int_{x_N(t)}^{x_{N+1}(t)} U\frac{\partial W_j}{\partial x}\frac{\partial W_{N+1}}{\partial x}dx\right]\Phi_j=\int_{x_N(t)}^{x_{N+1}(t)} W_iLU\ dx+W_{N+1}U_{N+1}\dot{x}|_{b(t)}. \tag{3.48}$$

We can summarize this system as a single matrix equation

$$\underline{C}\dot{\theta}+K(\underline{U})\underline{\Phi}=\underline{\mathbf{L}} \tag{3.49}$$

where $\underline{\Phi}$ is a vector of the unknowns Φ_j, $\underline{C}$ is a vector of the values c_i, $\underline{\mathbf{L}}$ is a vector of the right hand sides of (3.46) to (3.48) with the boundary term included, and the matrix $K(\underline{U})$ is a weighted stiffness matrix, consisting of entries

$$K(\underline{U})_{ij}=\int_{x_{i-1}}^{x_{i+1}} U(x)\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx. \tag{3.50}$$

The process for assembly of $K(\underline{U})$ is outlined in section 3.1.2. The matrix is singular but a value of ϕ is imposed at one point.

The right hand side $\underline{\mathbf{L}}$ of equation (3.49) can take many forms depending on the nature of the operator L and the boundary conditions. It may be necessary to make substitutions and/or perform integration by parts to obtain a computable form: a weak form requiring functions once-differentiable between nodes only. Note that if we sum over all rows of (3.49) the rows of the stiffness term $K(U)$ of (3.49) will sum to zero, and the c_i values will sum to unity. Providing that $\int_{a(t)}^{b(t)} LU dx$ is known, this makes it possible to recover $\dot{\theta}$ as the only remaining unknown by summing the rows of (3.49).

Recovering $\dot{X}$

Equation (3.37) is

$$\int_{a(t)}^{b(t)} w_i \dot{x}\, dx = \int_{a(t)}^{b(t)} w_i \frac{\partial \phi}{\partial x} dx. \tag{3.51}$$

Selecting the piecewise linear weight functions $w_i = W_i$ and using the piecewise linear approximations (3.41) and (3.43) with basis functions W_j we rewrite (3.51) as

$$\sum_{j=0}^{N+1} \left[\int_{a(t)}^{b(t)} W_i W_j \quad dx \right] \dot{X}_j = \sum_{j=0}^{N+1} \left[\int_{a(t)}^{b(t)} W_i \frac{\partial W_j}{\partial x} dx \right] \Phi_j. \tag{3.52}$$

In matrix form this is

$$M\underline{\dot{X}} = B\underline{\Phi}_j \tag{3.53}$$

where $\underline{\dot{X}}$ and $\underline{\Phi}_j$ are the vectors containing the unknown weightings $\dot{x}_j$ and the known Φ_j. The mass matrix M is symmetric positive definite and has entries

$$M_{ij} = \int_{a(t)}^{b(t)} W_i W_j dx \tag{3.54}$$

and B is an asymmetric matrix consisting of entries

$$B_{ij} = \int_{a(t)}^{b(t)} \frac{\partial W_j}{\partial x} W_i dx. \tag{3.55}$$

In this way, (3.53) can be solved to obtain the $\dot{X}_j$ values. The time integration is approximated by any chosen scheme, for example the explicit Euler scheme.

Recovering U

Equation (3.34) is

$$\int_{a(t)}^{b(t)} w_i u \, dx = c_i \theta(t). \tag{3.56}$$

Since the c_i are constant, we may write

$$\frac{1}{\theta(t)} \int_{a(t)}^{b(t)} w_i(x(t),t) u(x(t),t) dx = \frac{1}{\theta(0)} \int_{a(0)}^{b(0)} w_i(x(0),0) u(x(0),0) dx \tag{3.57}$$

for any time t. Using the weight functions W_i, the basis functions W_j, and making the substitution $U(x,t) = \sum_j W_j(x,t) U_j(t)$ we obtain

$$\sum_{j=0}^{N+1} \left[\int_{a(t)}^{b(t)} W_i W_j dx \right] U_j(t) = c_i \theta(t) \tag{3.58}$$

where c_i is given by

$$c_i = \frac{1}{\theta(0)} \sum_{j=0}^{N+1} \left[\int_{a(t_0)}^{b(t_0)} W_i W_j dx \right] U_j(t_0) \tag{3.59}$$

for the initial data $\underline{U}(0)$. Then (3.58) is equivalent to the mass matrix system

$$M\underline{U} = \underline{C}\theta(t) \tag{3.60}$$

with $\underline{C}$ as the vector containing entries c_i, and M the mass matrix as in (3.54). We may then solve (3.60) for $\underline{U}$.

Algorithm 3

The finite element solution of the non mass conserving equation (3.1) on the moving mesh therefore consists of the following steps. Given initial $\underline{U}$ and $\underline{X}$, and having calculated θ and $\underline{C}$ from (3.59), then for each time t:

1. Find $\dot{\theta}(t)$ by summing over all rows of the matrix equation (3.49);

2. Find the velocity potential by solving equation (3.49) for the $\Phi_j(t)$ values;

3. Find the node velocity by solving equation (3.53) for the $\dot{X}_j(t)$ values;

4. Generate the moving co-ordinate system from (3.18) using the forward Euler approximation. Update $\theta(t+dt)$ from $\dot{\theta}(t)$ similarly;

5. Find the solution values $U_j(t+dt)$ by solving the relative conservation equation (3.60).

Constructing the weighted stiffness matrix

Whilst it is necessary to use both mass and stiffness matrices in the calculations described in this chapter, the standard forms are well known. The weighted stiffness matrix $K(U)$ of equation (3.49) is of more interest.

Recall that the elements of $K(U)$ are given by equation (3.50) as

$$K(U)_{ij} = \int_{x_{i-1}}^{x_{i+1}} U(x) \frac{\partial W_i}{\partial x} \frac{\partial W_j}{\partial x} dx.$$

Over each element, the value of $U(x)$ is approximated in a piecewise linear manner. This leads to an element matrix for the interval $e = [i-1, i]$ of:

$$K(U)_e = \begin{pmatrix} \frac{U_i+U_{i-1}}{2(x_i-x_{i-1})} & \frac{-(U_i+U_{i-1})}{2(x_i-x_{i-1})} \\ \frac{-(U_i+U_{i-1})}{2(x_i-x_{i-1})} & \frac{U_i+U_{i-1}}{2(x_i-x_{i-1})} \end{pmatrix}. \tag{3.61}$$

Since U is continuous, and the functions W_i can be summed from two halves (one lying in each interval) we may superimpose the element matrices to form an assembled matrix for all (i, j),

$$K(U) =$$

$$\begin{bmatrix} \frac{U_1+U_0}{2(x_1-x_0)} & \frac{-(U_1+U_0)}{2(x_1-x_0)} & 0 & \cdots & \cdots & \cdots & 0 \\ \frac{-(U_1+U_0)}{2(x_1-x_0)} & \left(\frac{U_1+U_0}{2(x_1-x_0)} + \frac{U_2+U_1}{2(x_2-x_1)}\right) & \frac{(U_2+U_1)}{2(x_2-x_1)} & \cdots & \cdots & \cdots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \cdots & \frac{-(U_{i+1}+U_i)}{2(x_{i+1}-x_i)} & \left(\frac{U_{i+1}+U_i}{2(x_{i+1}-x_i)} + \frac{U_{i+2}+U_{i+1}}{2(x_{i+2}-x_{i+1})}\right) & \frac{-(U_{i+2}+U_{i+1})}{2(x_{i+2}-x_{i+1})} & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \frac{-(U_{N+1}+U_N)}{2(x_{N+1}-x_N)} & \frac{U_{N+1}+U_N}{2(x_{N+1}-x_N)} \end{bmatrix}.$$

3.1.3 Introducing the MMFEM framework in two dimensions

In two dimensions we also use piecewise linear weight and basis functions, again chosen so that all the weight functions in the domain together form a partition of unity. There are a

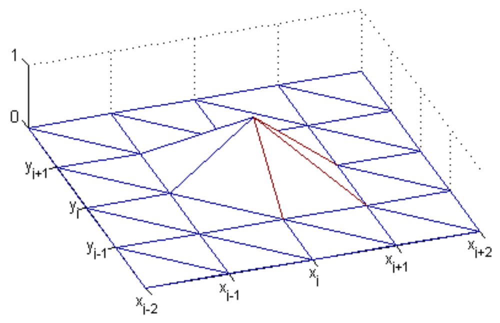


Fig. 3.2 Weight function or basis function centred at the node W_i

variety of suitable functions available but in this thesis we will use the simplest option, that of piecewise linear functions on a triangulated domain.
We triangulate the domain $\Omega(t)$ of the PDE we wish to solve. The nodes of the triangulation will be $\{\mathbf{X}_i\},(i=1,...,N)$.

We define the two-dimensional weight function $W_i(\mathbf{x})$ as the piecewise linear function that takes the value 1 at node i and the value 0 at all other nodes. Figure 3.2 contains a diagram for clarity. Our weak form of the PDE now specifically uses the piecewise linear weight functions W_i rather than the generic weight function w_i. From (3.30) we have

$$c_i\dot{\theta}(t)+\int_{\Omega(t)} u\nabla\phi\cdot\nabla W_i d\Omega=\int_{\Omega(t)} W_i Lu\, d\Omega+\int_{S(t)} W_i u\dot{\mathbf{x}}\cdot\hat{\mathbf{n}}\, dS. \tag{3.62}$$

Importantly, $\int_{S(t)} W_i u\dot{\mathbf{x}}\cdot\hat{\mathbf{n}}\, dS$ is assumed known from the boundary conditions.

The function $\phi(x)$ is again defined in terms of a piecewise linear approximation $\Phi(x)$. As in the 1-D case, each of the N nodes will have a coefficient Φ_j associated with it and these will specify a linear combination of the basis functions W_j. A convenient choice for these is the same set of functions as we use for the weight functions W_i,

$$\Phi(\mathbf{x},t)=\sum_{j=1}^{N}\Phi_j(t)W_j(\mathbf{x},t). \tag{3.63}$$

We also have the analogous definition

$$U(\mathbf{x},t)=\sum_{j=1}^{N}U_j(t)W_j(\mathbf{x},t). \tag{3.64}$$

We use the result that

$$\nabla\Phi = \sum_{j=1}^{N} \Phi_j \nabla W_j \tag{3.65}$$

and make a substitution for $\nabla\phi$ and u into equation (3.62), giving

$$c_i\dot{\theta}(t) + \sum_{j=1}^{N} \left[\int_{\Omega(t)} U\nabla W_j \cdot \nabla W_i d\Omega \right] \Phi_j = \int_{\Omega(t)} W_i LU \, d\Omega + \int_{S(t)} W_i U \dot{\mathbf{x}} \cdot \hat{\mathbf{n}} \, dS. \tag{3.66}$$

We also use the weight functions to evaluate the values for c_i, recalling equation (3.21)

$$c_i = \frac{1}{\theta(t)} \int_{\Omega(t)} W_i U \, d\mathbf{x}. \tag{3.67}$$

We may construct (3.66) for every triangle and node combination, and thus obtain a linear system of equations. When doing so we must take special care to include the boundary term for domain edge boundary triangles. The boundary terms for internal triangle edges will cancel out, since each edge connects two triangles which will have opposite outwards pointing normals $\hat{\mathbf{n}}$, and U is continuous. Each triangle with an edge lying along the boundary does make a contribution to the boundary term, so that in the sum of these contributions the whole boundary has been considered.
The assembled matrix equation has exactly the same form as the 1D case,

$$\underline{C}\dot{\theta}(t) + K(\underline{U})\underline{\Phi} = \underline{\mathbf{L}}. \tag{3.68}$$

However, the weighted stiffness matrix in 2D is given by

$$K(\underline{U})_{ij} = \int_{\Omega(t)} U(\mathbf{x})\nabla W_i(\mathbf{x}) \cdot \nabla W_j(\mathbf{x}) d\Omega. \tag{3.69}$$

As in the 1-D case, we can use (3.68) to obtain $\dot{\theta}$ by summing over all rows. The stiffness term will sum to zero and the c_i values will sum to unity, leaving the boundary terms assumed known and $\dot{\theta}$ as the only unknown. We can then use (3.68) in full form to determine the vector $\underline{\Phi}$, imposing a value of Φ_j at one point.

Recovering $\dot{\mathbf{X}}$

To find $\dot{\mathbf{x}}$, we work from the definition of ϕ (3.16)

$$\dot{\mathbf{x}} = \nabla\phi \tag{3.70}$$

for which a weak form is

$$\int_{\Omega(t)} w_i \dot{\mathbf{x}} d\Omega = \int_{\Omega(t)} w_i \nabla \phi d\Omega. \tag{3.71}$$

Using again the linear weight functions $w_i = W_i(\mathbf{x},t)$, and (using basis functions W_j) the piecewise linear approximations $\dot{\mathbf{X}} = \sum_j \dot{\mathbf{X}}_j(t) W_j(\mathbf{x},t)$ and $\nabla\Phi = \sum_j \Phi_j(t) \nabla W_j(\mathbf{x},t)$ we obtain

$$\sum_j \left[\int_{\Omega(t)} W_i W_j \, d\Omega \right] \dot{\mathbf{X}}_j = \sum_j \left[\int_{\Omega(t)} W_i \nabla W_j d\Omega \right] \Phi_j. \tag{3.72}$$

Hence in matrix form, (3.72) can be solved for $\dot{\mathbf{X}}$ using

$$M \underline{\dot{\mathbf{X}}} = B \underline{\Phi} \tag{3.73}$$

where $\underline{\dot{\mathbf{X}}} = \{\dot{\mathbf{X}}_j\}$, M is the symmetric mass matrix with elements $M_{ij} = \int_{\Omega(t)} W_i W_j d\Omega$, and B is an asymmetric matrix with elements $B_{ij} = \int_{\Omega(t)} W_i \nabla W_j d\Omega$.

Having found $\dot{\mathbf{X}}$, the nodes are repositioned using the forward Euler scheme, and $\dot{\theta}$ is used to update θ for the new time step in the same way.

Recovering U

We recover u using our distributed mass conservation principle (3.67). For each node i,

$$c_i = \frac{1}{\theta(t)} \int_{\Omega(t)} W_i u \, d\mathbf{x}.$$

Using the piecewise linear approximation $U = \sum_j U_j(t) W_j(\mathbf{x},t)$ we obtain

$$\sum_j \left[\int_{\Omega(t)} W_i W_j d\mathbf{x} \right] U_j = c_i \theta(t) \tag{3.74}$$

which is equivalent to the mass matrix system

$$M \underline{U} = \theta(t) \underline{C}. \tag{3.75}$$

This equation is used to calculate the initial (and constant) values of c_i, using the initial values of U_j and $\mathbf{X}_j$. After repositioning the nodes we may recover $U_j(t)$ from the mass matrix system (3.75).

Algorithm 4

The finite element solution of the non mass conserving equation (3.1) on the moving mesh therefore consists of the following steps. Given the initial U and $\mathbf{X}$, and having calculated $\underline{C}$ and the initial θ from the definition (3.20), then for each time t:

1. Find $\dot{\theta}(t)$ by summing over all rows of the matrix equation (3.68);
2. Find the velocity potential by solving equation (3.68) for the $\Phi_j(t)$ values;
3. Find the node velocity by solving equation (3.73) for the $\dot{\mathbf{X}}_j(t)$ values;
4. Generate the moving nodes $\mathbf{X}_j(t+dt)$ at the next time-step by solving (3.18) using the forward Euler approximation. Update $\theta(t+dt)$ from $\dot{\theta}(t)$ in the same way;
5. Find the solution $U(t+dt)$ by solving the relative conservation equation in the form (3.75).

Constructing the 2-D weighted stiffness matrix

The entries of the weighted stiffness matrix (3.69) are

$$K(\underline{U})_{ij} = \int_{\Omega(t)} U(\mathbf{x},t)\nabla W_i(\mathbf{x},t)\cdot\nabla W_j(\mathbf{x},t)d\Omega.$$

In order to determine the entries for each element matrix we examine a weight or basis function W_A on triangle ω_e represented by the co-ordinates $\mathbf{x}_{e_k}$ labelled (A,B,C) (figure 3.3). The triangle has angles α,β,γ as shown in figure 3.3. Such a triangle actually contains three local linear functions W_A,W_B,W_C, one associated with each node. The gradient of each weight or basis function can be calculated from the properties of the triangle. If height$_A$ is the height of triangle ω_e in the direction of the normal to side BC,

$$|\nabla W_A| = \frac{1}{\text{height}_A} = \frac{1}{b\sin\gamma}. \tag{3.76}$$

Likewise,

$$|\nabla W_B| = \frac{1}{\text{height}_B} = \frac{1}{c\sin\alpha} \tag{3.77}$$

and

$$|\nabla W_C| = \frac{1}{\text{height}_C} = \frac{1}{a\sin\beta}. \tag{3.78}$$

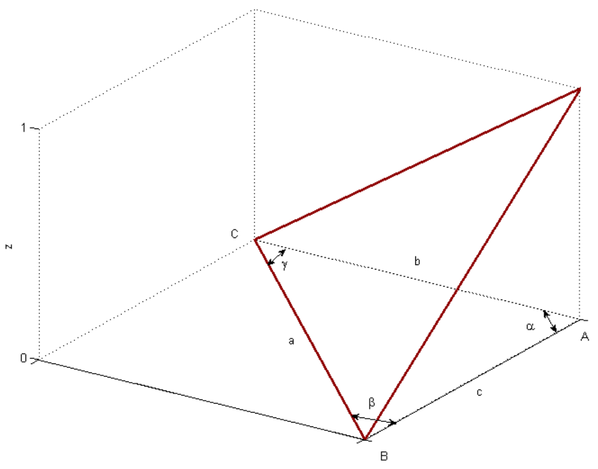


Fig. 3.3 Weight or basis function W_A centred at the node A

The area of the triangle can be calculated from any of these heights as

$$\begin{aligned}\text{area}_e &= \frac{1}{2}a(\text{height}_A) = \frac{1}{2}b(\text{height}_B) = \frac{1}{2}c(\text{height}_C)\\ &= \frac{1}{2}bc\sin\alpha = \frac{1}{2}ca\sin\beta = \frac{1}{2}ab\sin\gamma.\end{aligned} \tag{3.79}$$

The piecewise linear construction of function U means that U is linear in each triangle. We can therefore use the mean to give U in any triangle. Taking each combination of two nodes at a time, the entries for the element stiffness matrix can be determined. For example,

$$\begin{aligned}K_{BC} &= \int_{\omega_e} U\nabla W_B\cdot\nabla W_C d\mathbf{x}\\ &= \left(\frac{U_A+U_B+U_C}{3}\right)\frac{\text{area}_e}{(\text{height}_B)(\text{height}_C)}(-\cos\alpha)\\ &= \left(\frac{U_A+U_B+U_C}{3}\right)\frac{\frac{1}{2}bc\sin\alpha}{bc\sin^2\alpha}(-\cos\alpha)\\ &= \left(\frac{U_A+U_B+U_C}{3}\right)\frac{1}{2\sin\alpha}(-\cos\alpha)\\ &= \frac{(U_A+U_B+U_C)}{3}\frac{(-\cot)}{2}\alpha\end{aligned} \tag{3.80}$$

and since $\nabla(W_A + W_B + W_C) = 0$ it follows that

$$\begin{aligned} K_{AA} &= \int_{\omega_e} U \nabla W_A \cdot \nabla W_A d\mathbf{x} \\ &= -\int_{\omega_e} \left(\frac{U_A + U_B + U_C}{3} \right) \nabla W_A \cdot (\nabla W_B + \nabla W_C) d\mathbf{x} \\ &= \left(\frac{U_A + U_B + U_C}{6} \right) (\cot\beta + \cot\gamma). \end{aligned} \tag{3.81}$$

The other combinations follow in the same way so that the element stiffness matrix is:

$$K(U)_e = \left(\frac{U_A + U_B + U_C}{6} \right) \begin{pmatrix} \cot\gamma + \cot\beta & -\cot\gamma & -\cot\beta \\ -\cot\gamma & \cot\alpha + \cot\gamma & -\cot\alpha \\ -\cot\beta & -\cot\alpha & \cot\beta + \cot\alpha. \end{pmatrix}. \tag{3.82}$$

Entries calculated from each triangle can be superposed since we have a continuous U, and a W_j that can be summed from its component triangles. Because of the node interconnectivity, the full matrix is an obvious candidate for being assembled as part of the algorithm. Having described the technique we now turn to some existing examples.

3.2 Existing applications of the MMFEM

3.2.1 The porous medium equation

In their 2005 paper [5], Baines *et al.* apply algorithm 4 to the porous medium equation, for which the fixed reference form of the PDE is

$$\frac{\partial u}{\partial t} = \nabla \cdot (u^m \nabla u) \tag{3.83}$$

with $u = 0$ at the boundary. This is a mass conserving problem, so $\dot{\theta} = 0$ and only the simpler form of the method is required, given in section 3.1.1. When the steps outlined in this chapter are applied, the weak, integral form for Φ produced is

$$\int_{\Omega(t)} U \nabla \Phi \cdot \nabla W_i \, d\Omega = -\int_{\Omega(t)} U^m \nabla U \cdot \nabla W_i \, d\Omega \tag{3.84}$$

after integration by parts, and imposing the boundary condition $u = 0$. In [5] Φ was computed using a finite element approximation in both one and two dimensions. The model was found to be second-order accurate for $m = 1$, although of lower accuracy for $m = 3$. An improvement was made in having the initial grid non-uniform. The starting nodal positions were optimised using a least squares best fit to the initial conditions which improved accuracy to second order in one dimension. A further development was made in the 2006 paper [7], for which scale invariance is incorporated. In this case the time stepping and the mesh spacing are coupled together with the solution, and this was found to improve accuracy.

3.2.2 A fourth order problem

The fourth order diffusion equation seen in [5] and covered in detail in the 2014 PhD thesis by N.Bird [10], is

$$\frac{\partial u}{\partial t} = \nabla \cdot (u^m \nabla p) \tag{3.85}$$

where p, the pressure, is given by

$$p = -\nabla^2 u, \tag{3.86}$$

and

$$u = \nabla u \cdot \hat{\mathbf{n}} = 0 \tag{3.87}$$

on the boundary. This system models the capillary effects in the coating of a solid surface by a thin liquid film.

This problem requires an additional step compared to the Porous Medium Equation. It is necessary to obtain p from u, and this can also be done using a finite element approach. The weak form (3.86) for p, using integration by parts, is

$$\int_{\Omega(t)} w_i p \, d\Omega = \int_{\Omega(t)} \nabla w_i \cdot \nabla u \, d\Omega \tag{3.88}$$

which is suitable for finite element substitutions to be made. Solution of the resulting system of equations when $m = 1$ is compared to an exact solution and found to be fourth-order accurate in one dimension, and second-order accurate in two dimensions.

3.2.3 A Stefan problem

A non mass conserving example is given in the original 2005 Baines paper [5]. The single phase Stefan problem describes heat diffusion in two dimensions given by the PDE

$$\frac{\partial u}{\partial t} = k\nabla^2 u \tag{3.89}$$

with different values for k and an interface boundary condition

$$\left.\frac{\partial u}{\partial n}\right|_{\Gamma_1} = C_L \dot{\mathbf{x}} \cdot \mathbf{n} \qquad\qquad u_{\Gamma_1} = u_B \tag{3.90}$$

where C_L is the heat of phase change per unit volume, and the temperature u_{Γ_1} at the interface is the constant u_B. Here Γ_1 represents a moving boundary. This is the basis for the later two-phase method in [8].

3.2.4 Finite difference implementations

As has been mentioned, the underlying conservation method behind the MMFEM can also be implemented numerically from a finite difference perspective. The conservation method with a finite difference approach has been applied to a wide variety of problems in one and two dimensions. These demonstrate the generality and adaptability of the method, and also provide a guide to useful implementation directions for the MMFEM. In one and two dimensions, a blow up problem in a version of Fisher's Equation has been studied by S.Cole [21]. She studies systems with $p = 2$ in the reaction-diffusion PDE

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^p. \tag{3.91}$$

A model of a volcano has been studied by N. Robertson [44]. In this model, h is the height of the profile and U is a plastic flow property that is a function of h and the slope gradient. The PDE is:

$$\frac{\partial h}{\partial t} + \frac{\partial U}{\partial x} = w_s(x). \tag{3.92}$$

The system of Euler equations of compressible flow were studied in the PhD thesis by Wells [50] and the paper [51] by Wells, Glaister and Baines. This requires the solution of a system of three equations to be solved together at each time step as they have complicated interdependency and this was achieved using the finite difference ALE (Arbitrary Langrangian Eulerian) method.

A different combined system of equations was studied by S.Cole in [21]. Models of chemotaxis using the Keller-Segel equations were solved in one and two (radial) dimensions using a finite difference conservation method. The system involves a substrate and a reactant, and the PDEs are

$$u_t = \nabla.(k_1(u,v)\nabla u - k_2(u,v)u\nabla v) + k_3(u,v) \tag{3.93}$$

and

$$v_t = D_v\nabla^2 v + k_4(u,v) - k_5(u,v)v \tag{3.94}$$

where u is cell density, v is concentration of substrate, k_1 is diffusivity, k_2 is chemotactic sensitivity, k_3 is cell growth and death, k_4 is production of substrate and k_5 is degradation of substrate.

Free boundary problems

In their 2015 paper [36], Lee, Baines and Langdon use the finite difference implementation of the method to examine free boundary problems in one dimension. These included the Porous Medium equation, Richards equation and the Crank Gupta problem. A moving boundary is introduced with a flux boundary condition. For a boundary at $x = b(t)$, the boundary conditions are

$$u(b(t),t) = 0, \qquad u(b(t),t)\frac{db}{dt} = 0 \tag{3.95}$$

This is found to provide solutions accurate to second order.

In this thesis we will take a selection of these finite difference implementations and derive, implement and study the corresponding MMFEM.

3.3 Extensions to the MMFEM

Since the Baines, Hubbard and Jimack papers [5] and [7], a variety of interesting applications and extensions to the method have been investigated. Excellent overviews are given in the review papers [6] and [36]. It has been demonstrated that forms of the method can successfully be applied to linear and nonlinear systems, and systems with more than one phase. Highlights are described in this section.

High order nonlinear diffusion

In his PhD thesis [10], N.Bird considers nonlinear diffusion of second, fourth and sixth order. The MMFEM is applied, and interestingly an alternative type of higher order basis function is also tried. In one dimension Lagrange polynomials of linear, quadratic and fourth-order forms are used to provide a basis for the finite element approximation. The MMFEM is compared with a finite difference method. Some practical difficulties in applying the finite difference method are considered. These arise when the boundaries are permitted to move, resulting in certain functions becoming unbounded and singularities being introduced. It is found that the MMFEM alleviates this problem partially, although undesirable oscillations are still observed.

Two phase Stefan problem

In 2009, Baines, Hubbard and Jimack together with Mahmood [8] present a version of the algorithm from [5] in the form of an Arbitrary Lagrangian-Eulerian equation (ALE) that is sufficiently general to be able to model a two phase problem with a moving interface. Each phase is a diffusion system with driving PDEs

$$K_S \frac{\partial u}{\partial t} = \nabla \cdot (k_S \nabla u) \qquad \text{in solid regions} \tag{3.96}$$

$$K_L \frac{\partial u}{\partial t} = \nabla \cdot (k_L \nabla u) \qquad \text{in liquid regions} \tag{3.97}$$

where u is the temperature, the K are the volumetric heat capacities of each phase, and the k are the thermal conductivity of each phase. There is also an interface condition described by the Stefan equation

$$k_S \frac{\partial u_S}{\partial t} - k_L \frac{\partial u_L}{\partial t} = \lambda v \tag{3.98}$$

in the normal direction to the interface, with constant temperatures $u_S = u_L = u_M$ at the interface M. Here u_M is the temperature at which melting takes place, λ is the heat of phase change per unit volume, and v is the velocity of the interface.

Again a conservation principle is used to generate nodal velocities. This is then applied to a two-phase model, in both one and two (radially symmetric) dimensions. The model considers a domain containing both a solid and a liquid phase, with a moving interface between the two. This moving interface is of course internal to the domain. The transition between solid and liquid is modelled by applying the Stefan condition (3.98) at the interface. A system is constructed for each phase and also the interface, and the three are linked and solved as a single system to provide the mesh velocities. The masses are recovered for each phase separately since they are decoupled by the interface; these are again obtained from the conservation properties. This work is extended and developed for a new system in Chapter 5 of this thesis.

Ice sheets

In the 2013 PhD thesis [40] by Partridge and the subsequent paper in collaboration with Bonan *et al.* [11], a 1-D MMFEM is applied to dynamic ice flow equations to model the evolution of a glacier. The method is able to accurately capture and track the glacial front using a moving boundary framework, and the model is extended to two dimensions. In addition real world data is assimilated using the 3d-var scheme. This is found to work well in one dimension and to improve the accuracy of the profile of the ice front. In two dimensions the moving mesh alone works well, but the data assimilation aspect of the problem remains open-ended.

Explicit and implicit time-stepping schemes

In the methods described above, the time-stepping schemes are usually simple choices such as the explicit Euler method. These can place considerable constraints on the size of the time-step that can be made, because mesh tangling can occur. This is caused by nodes overtaking one another, and imposes a limitation on the speed of computation such that it becomes impractical to run models for long time horizons. A particular semi implicit or implicit method is proposed by Baines and Lee in the 2014 paper [9] that can make it impossible for nodes to tangle in one dimension. This allows us to choose a larger time-step. The method involves manipulating the structure of the velocity equation so that it makes use of its similarity to a variable co-efficient heat equation. A maximum/minimum principle can then be employed which makes it impossible for nodes to overtake. An alternative explicit method is given by Baines in his 2015 paper [4]. This method focuses on the node spacings or edge lengths (in 2D) and employs an amplification factor to calculate the distances. This factor is always positive and prevents overtaking. This is implemented in a finite difference framework in one dimension and the extension to two dimensions is outlined. It is noted that smoothness problems in ancillary variables may occur in certain circumstances.

Phase field models

Another approach to the study of phase transitions and interfaces is to attempt the modelling of a small but finite transition layer between two uniform bulk phases. These are known as phase field models, and a moving mesh finite element approach is discussed by Zhang and Du [53]. In these models the field varies smoothly but with a steep gradient in the transition layer. The example used is the Allen-Cahn equation, and the challenge of suitably resolving the thin interface layer is discussed with reference to appropriate time-stepping schemes and numerical stability. The paper also examines cases where such layers move over time such that dynamically evolving fronts can be tracked with an appropriately adapting mesh.

Chapter 4

New applications for MMFEMs

We shall begin this chapter by illustrating methods that form a part of a development pathway for MMFEMs. This will allow us to become familiar with useful techniques as well as to assess the incremental benefits offered by each evolutionary step in the development of the MMFEM.

4.1 An Illustration of the Equidistribution Method: a vertical velocity profile

The Ekman spiral [25] is a structure of currents near the ocean surface in which the flow direction rotates as one moves away from the surface. It was first noted by Swedish oceanographer Fridtjof Nansen, who observed an ice floe drifting at a tangent to the wind direction, and whose observations allowed Ekman to develop his model. The rotation is driven by the Coriolis effect. A feature of this structure of currents is the development of a shallow layer (Ekman layer) with behaviour that differs from the water below. The development of this layer in an initially stationary water column subject to wind stress is an interesting candidate for a moving mesh model, because we might wish to adapt the mesh to better resolve the emerging layer. Here we illustrate the equidistribution method and assess its utility for resolving the Ekman layer. A column of water is modelled under wind stress and with a Coriolis effect taken into account. A 1-D finite element method is used, and both the fixed mesh and an adaptive scheme for the mesh are considered. In this example, the adaptive scheme will be the equidistribution method, using arc length as a monitor function. We also consider alternative monitor functions. Time integration is performed using an Adams Bashforth method of third order.

The PDE of interest is

$$\frac{\partial \mathbf{u}}{\partial t} + F\mathbf{e}_z \times \mathbf{u} = \frac{\partial}{\partial z}\left(k_u \frac{\partial \mathbf{u}}{\partial z}\right) \tag{4.1}$$

where $\mathbf{u}(z,t)$ is the velocity, a function of the depth z. The velocity has two horizontal components, u_x and u_y. The physical constants are $F = 10^{-4}s^{-1}$ (the Coriolis force) and $K_u = 10^{-2}m^{-2}s^{-1}$ (eddy viscosity, a function of density, here assumed to be constant). The boundary condition at the deepest extent of the water column $z = -h$ is the Dirichlet condition $\mathbf{u}(z = -h,t) = 0$. On the surface, $z = 0$, we have a wind shear providing a flux boundary condition $\left.\frac{\partial \mathbf{u}}{\partial z}\right|_0 = \boldsymbol{\beta}$ with components $\beta_x = 10^{-2}s^{-1}$ and $\beta_y = 0$. The initial conditions are $\mathbf{u}(z,t=0) = 0$.
We separate (4.1) into x and y components. This generates two interdependent equations in 1-D. The PDEs for each component are

$$\frac{\partial u_x}{\partial t} = \frac{\partial}{\partial z}\left(k_u \frac{\partial u_x}{\partial z}\right) + F u_y \tag{4.2}$$

$$\frac{\partial u_y}{\partial t} = \frac{\partial}{\partial z}\left(k_u \frac{\partial u_y}{\partial z}\right) - F u_x. \tag{4.3}$$

4.1.1 Weak forms

To enable substitution of piecewise linear forms suitable for the finite element method, we must obtain the weak forms of the PDEs. The first step is to multiply the PDEs by a weight function w_i,

$$w_i \frac{\partial u_x}{\partial t} = w_i \frac{\partial}{\partial z}\left(k_u \frac{\partial u_x}{\partial z}\right) + w_i F u_y \tag{4.4}$$

$$w_i \frac{\partial u_y}{\partial t} = w_i \frac{\partial}{\partial z}\left(k_u \frac{\partial u_y}{\partial z}\right) - w_i F u_x \tag{4.5}$$

and integrate from $-h$ to 0. Then integration by parts gives the weak forms. These are, for each component respectively,

$$\begin{aligned}\int_{-h}^{0} w_i \frac{\partial u_x}{\partial t} dz &= \int_{-h}^{0} w_i \frac{\partial}{\partial z}\left(k_u \frac{\partial u_x}{\partial z}\right) dz + \int_{-h}^{0} w_i F u_y \, dz \\ &= \left[k_u w_i \frac{\partial u_x}{\partial z}\right]_{-h}^{0} - \int_{-h}^{0} k_u \frac{\partial w_i}{\partial z}\frac{\partial u_x}{\partial z} dz + \int_{-h}^{0} w_i F u_y \, dz \end{aligned} \tag{4.6}$$

and

$$\begin{aligned}\int_{-h}^{0} w_i \frac{\partial u_y}{\partial t} dz &= \int_{-h}^{0} w_i \frac{\partial}{\partial z}\left(k_u \frac{\partial u_y}{\partial z}\right) dz - \int_{-h}^{0} w_i F u_x \, dz \\ &= \left[k_u w_i \frac{\partial u_y}{\partial z}\right]_{-h}^{0} - \int_{-h}^{0} k_u \frac{\partial w_i}{\partial z}\frac{\partial u_y}{\partial z} dz - \int_{-h}^{0} w_i F u_x \, dz. \end{aligned} \tag{4.7}$$

We can now substitute piecewise linear functions into the weak forms. These functions are a weighted sum of a set of basis functions W_i,

$$U_x(z,t) = \sum_{j=0}^{N+1} U_{x_j}(t) W_j(z) \tag{4.8}$$

$$U_y(z,t) = \sum_{j=0}^{N+1} U_{y_j}(t) W_j(z). \tag{4.9}$$

The derivatives with respect to z are

$$\frac{\partial U_x}{\partial z} = \sum_{j=0}^{N+1} U_{x_j} \frac{\partial W_j}{\partial z} \tag{4.10}$$

$$\frac{\partial U_y}{\partial z} = \sum_{j=0}^{N+1} U_{y_j} \frac{\partial W_j}{\partial z} \tag{4.11}$$

and the derivatives with respect to time are

$$\frac{\partial U_x}{\partial t} = \sum_{j=0}^{N+1} \frac{\partial U_{x_j}}{\partial t} W_j \tag{4.12}$$

$$\frac{\partial U_y}{\partial t} = \sum_{j=0}^{N+1} \frac{\partial U_{y_j}}{\partial t} W_j. \tag{4.13}$$

We also choose the same functions W_i for the weight function w_i in (4.6) and (4.7), so that $w_i = W_i$.

The weak forms (4.6) and (4.7) after substitution are, for any i,

$$\sum_{j=0}^{N+1} \frac{\partial U_{x_j}}{\partial t} \int_{-h}^{0} W_i W_j \, dz = \left[k_u W_i \frac{\partial u_x}{\partial z}\right]_{-h}^{0} - \sum_{j=0}^{N+1} U_{x_j} \int_{-h}^{0} k_u \frac{\partial W_i}{\partial z}\frac{\partial W_j}{\partial z} dz + \sum_{j=0}^{N+1} U_{y_j} \int_{-h}^{0} F W_i W_j dz \tag{4.14}$$

and

$$\sum_{j=0}^{N+1} \frac{\partial U_{y_j}}{\partial t} \int_{-h}^{0} W_i W_j \, dz = \left[k_u W_i \frac{\partial u_y}{\partial z} \right]_{-h}^{0} - \sum_{j=0}^{N+1} U_{y_j} \int_{-h}^{0} k_u \frac{\partial W_i}{\partial z} \frac{\partial W_j}{\partial z} dz - \sum_{j=0}^{N+1} U_{x_j} \int_{-h}^{0} F W_i W_j dz. \tag{4.15}$$

The Dirichlet boundary condition $\mathbf{u}(z = -h, t) = 0$, is strongly imposed; therefore we will not need to calculate (4.14) and (4.15) for the zeroth node at $z = -h$. However, at $z = 0$ we will need to incorporate the boundary condition $\frac{\partial \mathbf{u}}{\partial z} = \beta$. For the weight functions w_i, a convenient choice is the collection of piecewise linear W_i functions from Chapter 3. These functions are also used as the basis functions for the piecewise linear approximations (4.8) and (4.9), so that in this case basis functions and weight functions are the same. The weak forms are now

$$\sum_{j=1}^{N+1} \frac{\partial U_{x_j}}{\partial t} \int_{-h}^{0} W_i W_j \, dz = k_u W_i \beta_x|_0 - \sum_{j=1}^{N+1} U_{x_j} \int_{-h}^{0} k_u \frac{\partial W_i}{\partial z} \frac{\partial W_j}{\partial z} dz + \sum_{j=1}^{N+1} U_{y_j} \int_{-h}^{0} F W_i W_j dz \tag{4.16}$$

and

$$\sum_{j=1}^{N+1} \frac{\partial U_{y_j}}{\partial t} \int_{-h}^{0} W_i W_j \, dz = - \sum_{j=1}^{N+1} U_{y_j} \int_{-h}^{0} k_u \frac{\partial W_i}{\partial z} \frac{\partial W_j}{\partial z} dz - \sum_{j=1}^{N+1} U_{x_j} \int_{-h}^{0} W_i F W_j dz. \tag{4.17}$$

This pair of equations can be written for any choice of i, so that $N + 1$ pairs of equations must be considered (recall that the zeroth node due to the Dirichlet condition need not be considered). The equations hold for all internal nodes, with the boundary term $k_u W_i \beta_x|_0$ being relevant to the $j = N + 1$ case only. This set of equations lends itself therefore to be written in matrix form. We define matrices $M = \{M_{ij}\}$, $K = \{K_{ij}\}$ and $F = \{F_{ij}\}$ with the following entries

$$M_{ij} = \int_{z_{i-1}}^{z_{i+1}} W_i(z) W_j(z) dz \tag{4.18}$$

$$K_{ij} = \int_{z_{i-1}}^{z_{i+1}} \frac{\partial W_i}{\partial z} \frac{\partial W_j}{\partial z} \, dz \tag{4.19}$$

$$F_{ij} = \int_{z_{i-1}}^{z_{i+1}} F W_i(z) W_j(z) dz. \tag{4.20}$$

We may now write (4.16) and (4.17) in matrix form. For $i = [1...N]$, this will be:

$$M\underline{\dot{U}}_x + K\underline{U}_x = F\underline{U}_y \tag{4.21}$$

$$M\underline{\dot{U}}_y + K\underline{U}_y = -F\underline{U}_x \tag{4.22}$$

with $\underline{\dot{U}}_x$ denoting a vector of entries $\{\dot{U}_{x_i}\}$ and so on. The boundary term is evaluated separately and added on to the $N+1$th row of the computation.

We can now determine $\dot{U}_x$ and $\dot{U}_y$,

$$\underline{\dot{U}}_x = M^{-1}(F\underline{U}_y - K\underline{U}_x) \tag{4.23}$$

$$\underline{\dot{U}}_y = M^{-1}(-F\underline{U}_x - KU\underline{U}_y). \tag{4.24}$$

Equation (4.1) can then be solved on the static mesh using the following algorithm.

Algorithm 5

1. Taking the initial condition $\mathbf{u}(z, t = 0) = 0$, equations (4.23) and (4.24) are solved to give $\underline{\dot{U}}_x$ and $\underline{\dot{U}}_y$, incorporating the boundary term on the $i = N+1$th row, and ignoring the $i = 0$ row, overwriting the value of U_0 as necessary;

2. Time integration to obtain $\underline{U}_x(t+dt)$ and $\underline{U}_y(t+dt)$ is performed using an Adams-Bashforth scheme of 3rd order.

A convenient visualisation of the Ekman spiral produced and its evolution in time is given in Figure 4.1.

We shall now compare the results obtained from this static mesh to those from a remapping mesh method, in this case an equidistribution method.

4.1.2 Equidistribution by arc length

The principle behind equidistribution is to select a function $M(z)$ that monitors a suitable property of the solution and then to equidistribute the nodes according to the integral of that function.
The monitor function used for this example is the scaled arc length monitor $M(z) = ds(z)/dz$

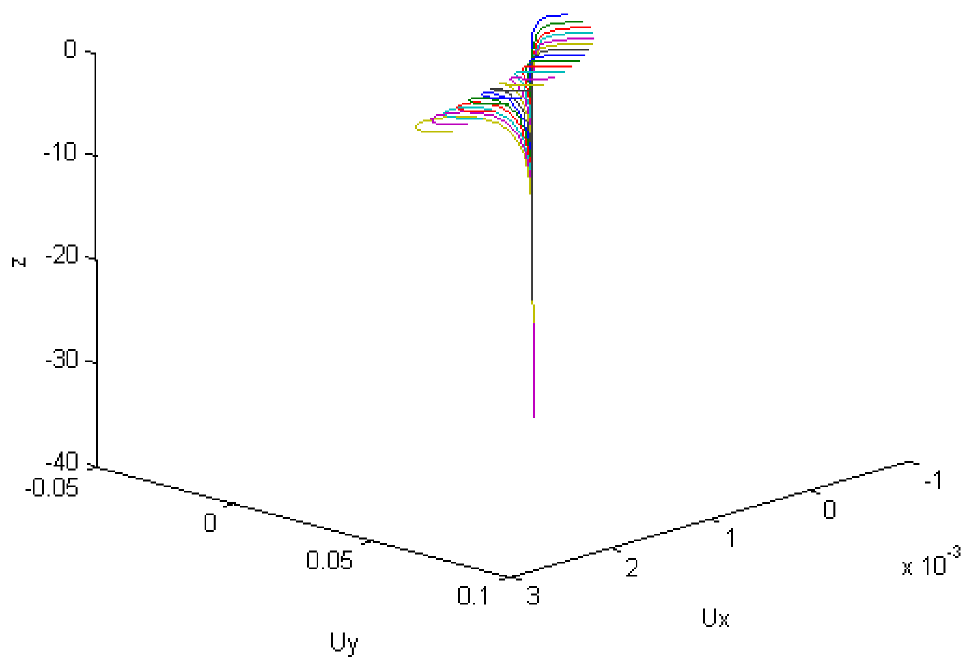


Fig. 4.1 A solution of (4.1) using a fixed mesh. Each line represents the velocity U in the x and y directions of a column of water and the variation of that velocity by depth. The navy blue line at the top is the velocity profile at $t = 5s$, and the subsequent lines show the evolution of the velocity profile at 5s intervals. The parts of the profiles that appear as horizontal lines on the chart all correspond to the surface of the water ($z = 0$). At the surface of the water, all lines have a similar y component to the velocity which is driven by the wind shear. The x component to the velocity at the surface is driven by the water movement happening below the surface. Below the surface, the velocity vector is rotated due to the Coriolis effect, and the extent of this rotation is a function of time. As the velocity vector below the surface is rotated, the surface velocity gains an increasingly large component perpendicular to the wind shear. The evolution in time up to $t = 100s$ is shown. The calculation assumed a depth of 40m, and used 100 elements with time steps of 0.01s.

where

$$s(z) = \int \left(1+\gamma\left(\frac{\partial \mathbf{u}}{\partial z}\right)^2\right)^{1/2} dz. \tag{4.25}$$

Here γ is a scaling factor appropriate to the scaling of the system being studied. For this example, $\gamma = 1000$ is used since the lateral velocity variations are three or four orders of magnitude smaller than the vertical scale. For our system this function can be expressed as

$$s(z) = \int_{-h}^{z} \left(1+\gamma\left(\frac{\partial u_x}{\partial z}\right)^2 + \gamma\left(\frac{\partial u_y}{\partial z}\right)^2\right)^{\frac{1}{2}} dz \tag{4.26}$$

where u_x and u_y are the horizontal components of the velocity $\mathbf{u}$. New grid points z_i are selected such that

$$\eta(z) = \frac{\int_{-h}^{z} M(z)dz}{\int_{-h}^{0} M(z)dz} \tag{4.27}$$

for a set of regularly spaced grid points $0 \leq \eta_i \leq 1$. By differentiating (4.27) with respect to η twice, we arrive at the differential equation

$$\frac{\partial}{\partial \eta}\left(M(z)\frac{\partial z}{\partial \eta}\right) = 0. \tag{4.28}$$

This is a nonlinear PDE, so we will solve it iteratively and therefore write

$$\frac{\partial}{\partial \eta}\left(M(\underline{z}^p)\frac{\partial \underline{z}^{p+1}}{\partial \eta}\right) = 0 \qquad (p = 0, 1, \ldots) \tag{4.29}$$

with an initial guess for a vector of discrete points $\underline{z}^0$. As we have only discrete values of $\underline{z}$ and therefore $M(\underline{z})$ to work with, we aim for approximate equidistribution and approximate (4.29) as

$$M(z^p_{j+1/2})(z^{p+1}_{j+1} - z^{p+1}_{j}) - M(z^p_{j-1/2})(z^{p+1}_{j} - z^{p+1}_{j-1}) = 0 \tag{4.30}$$

with our discretised monitor function M as

$$M(z_{j+1/2}) = \left(1+\gamma\left(\frac{U_{x_{j+1}} - U_{x_j}}{z_{j+1} - z_j}\right)^2 + \gamma\left(\frac{U_{y_{j+1}} - U_{y_j}}{z_{j+1} - z_j}\right)^2\right)^{1/2}. \tag{4.31}$$

We assemble (4.30) into a matrix system,

$$T(\underline{z}^p)\underline{z}^{p+1} = \underline{b} \tag{4.32}$$

where T is a tridiagonal matrix of the form

$$T = \begin{bmatrix} \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & M(z_{j-1/2}) & -M(z_{j-1/2}) - M(z_{j+1/2}) & M(z_{j+1/2}) & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{4.33}$$

and

$$\underline{b} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \end{bmatrix}. \tag{4.34}$$

The boundary conditions on z are incorporated into T and $\underline{b}$ so that the top and bottom rows of each are as in

$$T = \begin{bmatrix} 1 & 0 & 0 & \dots \\ . & . & . & . \\ \dots & 0 & 0 & 1 \end{bmatrix} \qquad \underline{b} = \begin{bmatrix} 0 \\ . \\ -h \end{bmatrix}. \tag{4.35}$$

To implement the equidistribution process into the finite element scheme, we solve (4.32) to find the new mesh spacings $z(t)$. We then interpolate the solution onto the new grid using a cubic spline. The solution of (4.32) is computationally expensive, particularly as it involves an iterative process, so we won't remesh the domain at every time step. Instead we choose a remeshing rate, which will be set to be once every 10 time steps for these models.

Algorithm 6

1. Taking the initial condition $\mathbf{u}(z, t = 0) = 0$, equations (4.23) and (4.24) are solved to give $\underline{\dot{U}}_x$ and $\underline{\dot{U}}_y$, incorporating the boundary term on the $i = N + 1$th row and ignoring the $i = 0$ row, overwriting the values of $U_x(0,t)$ and $U_y(0,t)$ as necessary;

2. Time integration to obtain $\underline{U}_x(t + dt)$ and $\underline{U}_y(t + dt)$ is performed using an Adams-Bashforth scheme of 3rd order;

3. Steps 1 and 2 are repeated for 10 iterations;

4. Using equation (4.32), calculate $\underline{z}^{p+1}$ from $\underline{z}^{p}$ and repeat until satisfactory convergence ($|\underline{z}^{p+1} - \underline{z}^{p}| < 10^{-5}$) is achieved between the two. This is then $\underline{z}(t+dt)$;

5. Interpolate the solutions $\underline{U}_x(t+dt)$ and $\underline{U}_y(t+dt)$ onto the new grid $\underline{z}(t)$.

Figure 4.2 shows a comparison of the fixed grid and an adapted grid at t=100s. It is easy to see the clear improvements to the model that come with increased resolution. The fixed grid with 160 elements is the highest resolution that the current MATLAB implementation can reasonably compute. If we take this as our reference solution, we see that the moving mesh equidistribution models are more accurate than the fixed mesh models with the same number of elements, without a corresponding leap in the computational cost. The improvement is primarily in the gradient of the top portion of the line. Figure 4.3 shows where the grid adaptation has taken place. We see most adaptation around node 8, where the accelerations of the fluid integrated over time have been the greatest. Node 8 is at a depth of around 7m. However, there is a physically important transition from a shearing of the fluid to a stationary fluid (Ekman layer) at around 10-20m depth at time $t = 100$ and this is poorly resolved. We investigate an alternative monitor function in search of a method that can better resolve this transition.

Equidistribution by curvature

As the region we wish to better resolve is a region of high curvature, we shall attempt a modification of (4.26) to provide a measure of curvature. This experimental monitor function, which we will call $C(z)$, is based on the form of the arc length monitor, but using a second order derivative to quantify the rate of change of slope. We define

$$C(z) = \int_{-h}^{z} (1 + (\gamma \nabla^2 \mathbf{u})^2)^{\frac{1}{2}} \, dz. \tag{4.36}$$

For our system $C(z)$ is then

$$C(z) = \int_{-h}^{z} \left(1 + \gamma \left(\frac{\partial^2 U_x}{\partial z^2} \right)^2 + \gamma \left(\frac{\partial^2 U_y}{\partial z^2} \right)^2 \right)^{\frac{1}{2}} dz. \tag{4.37}$$

We find the following problems with using curvature as a monitor function:

- Curvature is estimated at a point rather than measured absolutely, so is dependent

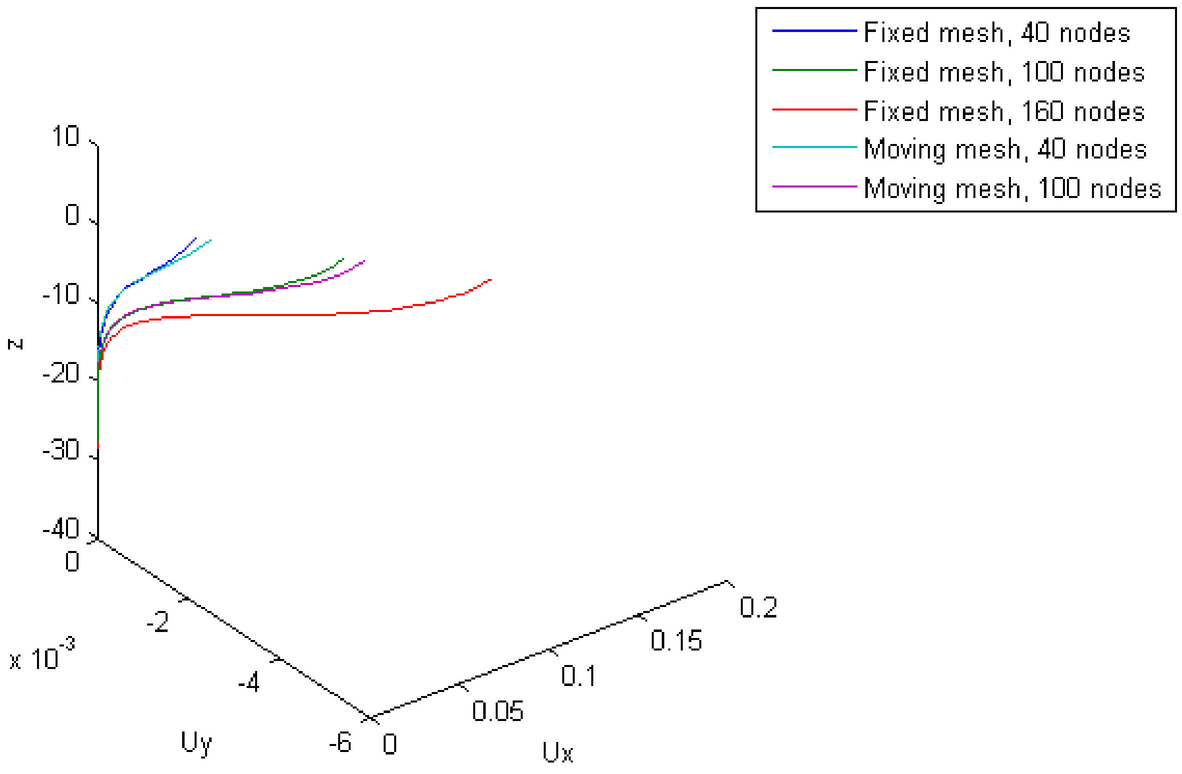


Fig. 4.2 A comparison of the solution given by fixed and equidistributed grids for 40, 100 and 160 elements at t=100. See figure 4.1 for a detailed explanation of how this chart represents a velocity profile. The reference solution (red) is computed on a fixed grid with 160 elements. Assuming this higher resolution computation to be the most accurate, we compare the green and purple solutions, computed on 100 elements. We see that we can improve accuracy through the moving grid whilst still only requiring the inversion of 100 by 100 matrices. This advantage is of course partly offset by requiring the iterative step to remesh the domain.

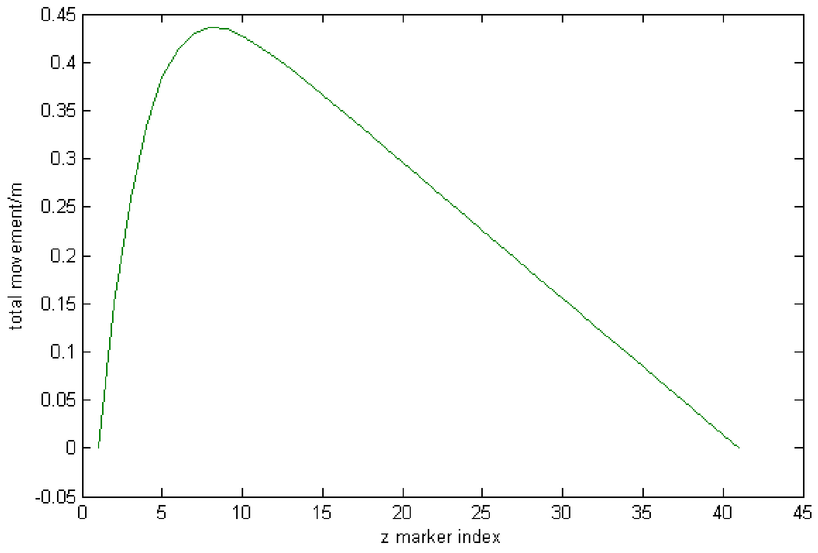


Fig. 4.3 Overall mesh movement for 40 nodes. Total movement for node i is given by $z_i(t) = z_i(0)$.

on the size of the interval in z for accuracy. A sensible solution to this would be to integrate $C(z)$ along the curve;

- Using the integral of curvature as a monitor function we find, at the iterative stage (4.29), many cases that do not converge;
- We achieve sufficient stability to run the model in limited cases (t<30, $\gamma = 1000$). However, the node movement cannot take place in advance of the feature of interest forming, so the transition zone is not better resolved. Instead, the increased resolution is observed where the transition zone had previously been located.

From these examples we can see that the equidistribution method has improved the resolution of the Ekman layer for the arc length monitor function, but does not compare well with a grid with globally higher resolution. The curvature monitor function performs rather badly. A weakness of the method is the latency in the system; because we look at the solution to determine mesh spacing we can only respond to features of interest that have already formed. We have no way of observing the formation of features of interest. Further weaknesses include the need to interpolate the solution after the grid has been remeshed, and the need for an iterative solver and the resulting stability/convergence challenges that this may present. The conservation method can improve upon each of these drawbacks. As a velocity based method, interesting features in the flow should be tracked as they develop, and therefore be tracked before they are observed in the features of the solution. The grid will not need remeshing periodically so we will not need to interpolate the solution at any point, instead the solution is tied to the grid at all times by the moving basis functions. Furthermore, the need for an iterative step is eliminated. We will now illustrate the conservation method using Fisher's equation.

4.2 An Illustration of the Conservation Method: Fisher's Equation

Fisher's equation is a reaction diffusion system that describes a balance between linear diffusion and nonlinear reaction. It arises in ecology where it is known as a population growth model, but it can also be used to describe biological invasion, or a simple combustion model for flame propagation, amongst others. In contrast to the alternative population models described later, it involves only a reactant, *i.e.* any substrate is not relevant. The Fisher's equation is known for exhibiting blow-up, which makes it a particularly interesting target for an adaptive mesh method. We consider here an illustration of Fisher's equation using the conservation method. The aim will be to derive a moving mesh that increases resolution around the blow-up.

4.2.1 Fisher's Equation in 1D

Fisher's equation has a variety of common forms but following the Budd *et al.* paper [14], we look at the particular form describing the temperature u of a reacting or combusting medium. The Masters theses by Edgington, 2011 [24], and Cole, 2009 [21], both examine this same version of Fisher's equation on moving meshes from a finite difference perspective, but here we look at a finite element perspective. As discussed in Chapter 3, for ease of comparison between studies, we will refer to u as mass rather than temperature. Fisher's equation is

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^p. \tag{4.38}$$

We consider the case with $p = 2$,

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^2 \qquad\qquad (a \le x \le b) \tag{4.39}$$

which has the weak form

$$\int_a^b w_i \frac{\partial u}{\partial t} dx = \int_a^b w_i \left(\frac{\partial^2 u}{\partial x^2} + u^2 \right) \, dx. \tag{4.40}$$

We will work with the Dirichlet boundary conditions $u(a,t) = u(b,t) = 0$ for $a = -0.5$, $b = 0.5$, and $u(x,0) \geq 0$ in order to produce results that we can compare with Budd *et al.* [14]. We will call this case 1. However, we will also consider an alternative set of boundary conditions where the end points of the range will not be fixed. In this case the boundary conditions are $u(a(t),t) = u(b(t),t) = 0$ where the points $a(t)$ and $b(t)$ may have a non zero velocity. We call this case 2. This allows all nodes including boundary nodes to respond to the mass dynamics. This is a useful alternative system to model because it will allow us to develop the approach that can later be used for a two phase system with a moving interface.

Conservation of relative mass

The approach to moving the nodes is now driven entirely by a conservation of mass in each patch of elements; we have no specialised monitor function with this approach. As the domain moves, the elements must shrink or grow to keep the proportions of mass constant in each. However, for this particular problem we do not have the advantage of a conservative total mass. Instead, as in the generic example in Chapter 3, we will introduce the concept of a relative total mass. This will be defined as the *proportion* of mass in each patch of elements. These proportions will remain constant with respect to time. This principle is set out as follows for the Fisher's equation. Define $\theta(t)$ to be the area (mass) under the entire solution curve at time t,

$$\theta(t) = \int_{a(t)}^{b(t)} u(x,t)\ dx. \tag{4.41}$$

We may use (4.41) to calculate θ for any known u at the initial time.

We now show how the concept of relative conservation of mass is applied in a distributed fashion across the domain. We may write such a relative conservation principle as

$$\frac{1}{\theta(t)} \int_{a(t)}^{b(t)} w_i u dx = c \tag{4.42}$$

where w_i is a weight function. If we select the weight functions from a partition of unity, so that $\sum_i w_i = 1$, we may define a distributed conservation of relative mass principle. This is consistent with the conservation of total relative mass, and additionally requires that relative mass is conserved within each patch of elements. We define

$$\frac{1}{\theta(t)} \int_{a(t)}^{b(t)} w_i u dx = c_i \tag{4.43}$$

where the c_i are constant in time. This is our modified conservation principle. Then for all i,

$$\frac{d}{dt}\left[\frac{1}{\theta(t)} \int_{a(t)}^{b(t)} w_i u dx\right] = 0. \tag{4.44}$$

Using Leibnitz' integral rule we have

$$-\frac{\dot{\theta}}{\theta}\int_{a(t)}^{b(t)} w_i u dx + \int_{a(t)}^{b(t)} \frac{\partial}{\partial t}(w_i u) dx + (w_i u)|_b \frac{db}{dt} - (w_i u)|_a \frac{da}{dt} = 0 \tag{4.45}$$

or

$$-\frac{\dot{\theta}}{\theta}\int_{a(t)}^{b(t)} w_i u dx + \int_{a(t)}^{b(t)} \frac{\partial}{\partial t}(w_i u) dx + \int_{a(t)}^{b(t)} \frac{\partial}{\partial x}(\dot{x} w_i u) dx = 0 \tag{4.46}$$

where $\dot{x}$ is any velocity consistent with db/dt and da/dt, and then

$$-\frac{\dot{\theta}}{\theta}\int_{a(t)}^{b(t)} w_i u dx + \int_{a(t)}^{b(t)} \left[w_i \frac{\partial u}{\partial t} + u\frac{\partial w_i}{\partial t} + w_i \frac{\partial}{\partial x}(u\dot{x}) + u\dot{x}\frac{\partial w_i}{\partial x}\right] dx = 0. \tag{4.47}$$

After substitution of (4.43)

$$-c_i\dot{\theta} + \int_{a(t)}^{b(t)} \left[w_i \frac{\partial u}{\partial t} + u\frac{\partial w_i}{\partial t} + w_i \frac{\partial}{\partial x}(u\dot{x}) + u\dot{x}\frac{\partial w_i}{\partial x}\right] dx = 0. \tag{4.48}$$

We fix our weight functions w_i to the domain that moves with velocity $\dot{x}$. Therefore we can argue, by analogy to a convecting system, that

$$\frac{\partial w_i}{\partial t} + \dot{x}\frac{\partial w_i}{\partial x} = 0. \tag{4.49}$$

We can multiply (4.49) by u and take out this term from equation (4.48). Rearrangement yields

$$\int_{a(t)}^{b(t)} w_i \frac{\partial}{\partial x}(\dot{x}u) dx = -\int_{a(t)}^{b(t)} w_i \frac{\partial u}{\partial t} dx + c_i\dot{\theta}. \tag{4.50}$$

Substituting from the weak form of Fisher's equation (4.40),

$$\int_{a(t)}^{b(t)} w_i \frac{\partial}{\partial x}(\dot{x}u) dx = -\int_{a(t)}^{b(t)} w_i \left(\frac{\partial^2 u}{\partial x^2} + u^2\right) dx + c_i\dot{\theta}. \tag{4.51}$$

Integrating the first term on the left hand side by parts (assuming w_i is sufficiently smooth),

$$[w_i \dot{x} u]_{a(t)}^{b(t)} - \int_{a(t)}^{b(t)} \frac{\partial w_i}{\partial x}\dot{x} u dx = -\int_{a(t)}^{b(t)} w_i \left(\frac{\partial^2 u}{\partial x^2} + u^2\right) dx + c_i\dot{\theta}. \tag{4.52}$$

For both case 1 and case 2, we note that the boundary term on the left hand side of (4.52) is zero due to the Dirichlet conditions $u(a(t),t) = u(b(t),t) = 0$. We now have a weak form of the velocity equation (4.52), so that, given $u(x,t)$, $\theta(t)$ and $\dot{\theta}(t)$, we can solve for $\dot{x}$. We can transform the second order derivative term in (4.52) by integrating by parts again on the right hand side

$$\int_{a(t)}^{b(t)} \frac{\partial w_i}{\partial x}\dot{x}udx = \left[w_i\frac{\partial u}{\partial x}\right]_{a(t)}^{b(t)} - \int_{a(t)}^{b(t)} \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\ dx + \int_{a(t)}^{b(t)} w_i u^2 dx - c_i\dot{\theta}. \tag{4.53}$$

We refer to the w_i as weight functions. Equation (4.53) is now in a suitable form for finite element functions to be substituted.

Finite elements

We choose the set of functions W_i for our weight functions. Consider the boundary term $\left[w_i\frac{\partial u}{\partial x}\right]_{a(t)}^{b(t)}$ in (4.53). In a finite element framework with Dirichlet conditions, the usual approach is to solve (4.53) for internal nodes only, and in those cases the boundary term would be equal to zero. Therefore the boundary term disappears. The given solution on the boundary can then be strongly imposed. However, in a conservation based system, ignoring boundary terms would destroy conservation in general. In this circumstance, following [33] we switch to a modified set of weight functions, which we will call $\tilde{W_i}$. These weight functions include a combined weight function for the boundary node and its nearest neighbour. This will allow us to strongly impose the Dirichlet conditions without destroying mass conservation. Our approach from here varies depending on the presence or otherwise of a free boundary.

Case 1: Fixed boundaries: Boundary conditions are $u = 0, \dot{x} = 0$

For the static boundary, case 1, we work in modified weight functions throughout. The modified weight functions $\tilde{W_i}$ are constructed from the original weight functions W_i as follows,

$$\tilde{W_1}(t,x) = W_0(t,x) + W_1(t,x) \tag{4.54}$$

and

$$\tilde{W_N}(t,x) = W_N(t,x) + W_{N+1}(t,x) \tag{4.55}$$

with the remaining W_i unaltered. These modified weight functions are illustrated in figure 4.4. Note the dimension of the subspace in which these functions reside is reduced from $N+2$ to N. The c_i values of equation (4.43) must be adjusted accordingly,

$$\tilde{c_1} = c_0 + c_1 = \int_a^b \frac{1}{\theta}(W_0 + W_1)u\ dx \tag{4.56}$$

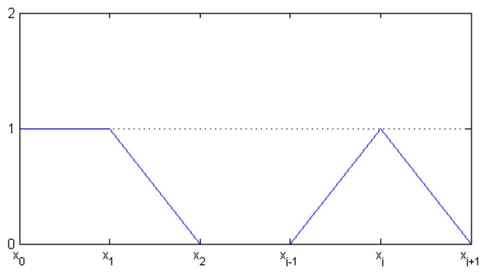


Fig. 4.4 Modified weight functions in 1-D for boundary node x_0 and internal node x_i. These W_i form a partition of unity and are compatible with strongly imposed Dirichlet conditions.

and

$$\tilde{c_N} = c_N + c_{N+1} = \int_a^b \frac{1}{\theta}(W_N + W_{N+1})u\ dx. \tag{4.57}$$

We then have no c_0 or c_{N+1} values. The remaining c_i are unaltered. The use of $\tilde{W}_i$ and $\tilde{c}_i$ ensure that global conservation is not violated in (4.43) by Dirichlet boundary conditions. We define the piecewise linear approximations $\dot{X}$ and U in terms of the (unmodified) basis functions

$$\dot{X}(x,t) = \sum_{j=0}^{N+1} \dot{X}_j(t)W_j(x,t) \tag{4.58}$$

$$U(x,t) = \sum_{j=0}^{N+1} U_j(t)W_j(x,t). \tag{4.59}$$

We may substitute these in (4.53) to give

$$\sum_{j=0}^{N+1}\left[\int_a^b U\frac{\partial\tilde{W}_i}{\partial x}W_j dx\right]\dot{X}_j = \left[\tilde{W}_i\frac{\partial u}{\partial x}\right]_a^b - \sum_{j=0}^{N+1}\left[\int_a^b \frac{\partial\tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_a^b \tilde{W}_i U^2 dx - \tilde{c}_i\dot{\theta}. \tag{4.60}$$

Note also the fixed domain boundaries a and b for case 1. The term $\left[\tilde{W}_i\frac{\partial u}{\partial x}\right]_a^b$ is non-zero for case 1 and must be included. For all nodes we can therefore write this as a system of equations. For $i = 1,2,3,..,N$

$$\sum_{j=0}^{N+1}\left[\int_a^b U\frac{\partial\tilde{W}_i}{\partial x}W_j dx\right]\dot{X}_j = -\sum_{j=0}^{N+1}\left[\int_a^b \frac{\partial\tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_a^b \tilde{W}_i U^2 dx - \tilde{c}_i\dot{\theta} + \left[\tilde{W}_i\frac{\partial u}{\partial x}\right]_a^b \tag{4.61}$$

or in matrix form

$$\tilde{B}(U)\underline{\dot{X}} = \underline{\tilde{f}} \tag{4.62}$$

where $\dot{\underline{X}} = \{X_i\}$, and $\underline{\tilde{f}} = \{\tilde{f}_i\}$ given by

$$\tilde{f}_i = -\sum_{j=0}^{N+1} \left[\int_a^b \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} dx \right] U_j + \int_a^b \tilde{W}_i U^2 dx - \tilde{c}_i \dot{\theta} + \left[\tilde{W}_i \frac{\partial u}{\partial x} \right]_a^b . \tag{4.63}$$

At a and b, the weighting $\tilde{W}_i = 1$, so in order to estimate the term $\left[\tilde{W}_i \frac{\partial u}{\partial x}\right]_a^b$ we simply calculate $\frac{\partial u}{\partial x}$ with a finite difference approximation. The nonlinear term, $\int_a^b \tilde{W}_i U^2 dx$, is evaluated exactly using quadrature rules (described in detail in [42]). In general, Gaussian quadrature rules are the appropriate choice, especially when higher dimensions come into play, but in this case the simplest choice for implementation is Simpson's rule. Simpson's rule is exact for cubics but its advantage here is that minimal interpolation between nodes is required. Simpson's rule (exact for cubics) is

$$\int_a^b f(x)\ dx = \frac{b-a}{6} \left(f(a) + 4\frac{f(a+b)}{2} + f(b) \right). \tag{4.64}$$

The matrix $\tilde{B}(\underline{U})$ is asymmetric and has entries of the form

$$\tilde{B}(\underline{U})_{ij} = \int_a^b U \frac{\partial \tilde{W}_i}{\partial x} W_j dx. \tag{4.65}$$

We now have the equation (4.62) for $\dot{\underline{X}}$ and we can also use (4.62) as a convenient way to recover $\dot{\theta}$. If we sum over all rows i of the matrix system (4.62), the terms containing $\frac{\partial \tilde{W}_i}{\partial x}$ will sum to zero. The $\tilde{c}_i$ will sum to 1 from their definitions (4.43) and (4.56), which leaves the sum over all i as

$$\dot{\theta} = \sum_i \left[\int_a^b \tilde{W}_i U^2 dx \right] + \sum_i \left[\tilde{W}_i \frac{\partial u}{\partial x} \right]_a^b = \int_a^b U^2\ dx + \left[\frac{\partial u}{\partial x} \right]_a^b \tag{4.66}$$

which allows us to determine $\dot{\theta}$ for either the fixed or moving boundary case, since the dependence on the choice of w_i is removed in the summation process. The expression (4.62) does determine $\dot{\underline{X}}$ but with two caveats. Firstly, matrix $\tilde{B}(\underline{U})$ is singular, so a boundary condition on $\dot{\underline{X}}$ is required to bring the infinity of solutions down to a unique solution. This is most easily achieved by requiring $\dot{X}_i = 0$ for one node, which then serves as an anchor point. In our system, given a set of initial conditions symmetrical about the origin, the natural symmetry means that the origin is the obvious choice. We may then reduce the matrix system under the transformation $\tilde{B}(\underline{U}) \to \bar{B}(\underline{U})$, removing the column corresponding to the stationary central node and obtaining a potentially non-singular $\bar{B}(\underline{U})$. Even then the matrix

$\bar{B}(\underline{U})$ could be singular if U were constant and it had an odd number of rows and columns. However, the second caveat is that when we consider the system in two dimensions, the velocity is not unique because we could add an arbitrary curl vector to $u\dot{\mathbf{x}}$ (see equation (3.5)). By introducing a velocity potential Φ, we can avoid this problem since the velocity potential *is* unique. We then specify a curl of zero when we recover $\dot{X}$ from Φ. In order to keep the method consistent between one and two dimensions then, we will also work with a velocity potential in one dimension. We proceed therefore by introducing the velocity potential Φ, defined by

$$\dot{X} = \frac{\partial \Phi}{\partial x} \tag{4.67}$$

where

$$\Phi(x,t) = \sum_{j=0}^{N+1} W_j(x,t)\Phi_j(t) \tag{4.68}$$

so that

$$\frac{\partial \Phi}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x}\Phi_j. \tag{4.69}$$

Substituting this into equation (4.61), equation (4.62) becomes

$$\tilde{K}(\underline{U})\underline{\Phi} = \underline{\tilde{g}} \tag{4.70}$$

where $\underline{\Phi}$ is the vector containing $\{\Phi_j\}$. In (4.69) $\tilde{K}(\underline{U})$ is a symmetric positive definite matrix with entries, for $i,j = [1,...,N]$, of

$$\tilde{K}(\underline{U})_{ij} = \int_a^b U \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x} dx, \tag{4.71}$$

and $\underline{\tilde{g}}$ is a vector with entries, for $i,j = [1,...,N]$, of

$$\tilde{g}_i = -\sum_{j=0}^{N+1} \left[\int_a^b \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x} dx\right] U_j + \int_a^b \tilde{W}_i U^2 dx - \tilde{c}_i\dot{\theta} + \left[\tilde{W}_i \frac{\partial u}{\partial x}\right]_a^b. \tag{4.72}$$

The nonlinear term is evaluated using Simpson's rule (4.64). Again, $\tilde{K}(\underline{U})$ is singular and the system (4.70) requires a boundary condition. We impose $\Phi = 0$ at one node, by removing the corresponding column in $\tilde{K}(\underline{U})$. Then the reduced system

$$\bar{K}(\underline{U})\underline{\Phi} = \underline{\tilde{g}} \tag{4.73}$$

is uniquely solvable for Φ.

Getting $\dot{X}$

We then obtain $\dot{X}$ from a finite element approximation of (4.67) at each node. From the definition (4.67) we write the weak form,

$$\int_a^b w_i \dot{X}\ dx = \int_a^b w_i \frac{\partial \Phi}{\partial x} dx. \tag{4.74}$$

For case 1, the fixed boundaries are equivalent to imposing the boundary conditions $\dot{x}|_a = 0$ and $\dot{x}|_b = 0$. To impose these without violating relative mass conservation in (4.43), modified weight functions are again required. We select the modified weight functions $w_i = \tilde{W}_i$ of (4.54), (4.55) and use the piecewise linear approximations (4.58) and (4.68). We obtain

$$\sum_{j=0}^{N+1} \left[\int_a^b \tilde{W}_i W_j\ dx \right] \dot{X}_j = \sum_{j=0}^{N+1} \left[\int_a^b \tilde{W}_i \frac{\partial W_j}{\partial x} dx \right] \Phi_j. \tag{4.75}$$

In matrix form this is

$$\tilde{M} \underline{\dot{X}} = \tilde{B} \underline{\phi}_j. \tag{4.76}$$

The matrix $\tilde{M}$ is a positive definite and well-conditioned mass matrix with entries

$$\tilde{M}_{ij} = \int_a^b \tilde{W}_i W_j dx \tag{4.77}$$

and $\tilde{B}$ is an asymmetric matrix similar to (4.65), with entries

$$\tilde{B}_{ij} = \int_a^b \tilde{W}_i \frac{\partial W_j}{\partial x} dx. \tag{4.78}$$

Note that we will only need to invert $\tilde{M}$ in order to recover $\underline{\dot{X}}$.

Finding X

A time integration approximation such as forward Euler is used to generate the grid at the next time step from the values of $\dot{X}$.

Recovering U

To generate the new solution for U at the new time step we return to our relative conservation principle (4.43), with our modified basis functions $w_i = \tilde{W}_i$,

$$\frac{1}{\theta(t)}\int_a^b \tilde{W}_i u dx = \tilde{c}_i.$$

In discretised form this becomes, with $U(x,t) = \sum_j W_j(x,t)U_j(t)$,

$$\sum_{j=0}^{N+1}\left[\int_a^b \tilde{W}_i W_j dx\right] U_j = \tilde{c}_i \theta(t) \tag{4.79}$$

where the $\tilde{c}_i$ are given by the modified values

$$\tilde{c}_i = \frac{1}{\theta(t_0)}\int_a^b \tilde{W}_i U_0 dx = \frac{1}{\theta(t_0)}\int_a^b \tilde{W}_i u_0 \ dx \tag{4.80}$$

for initial data u_0, if the U_0 is the L^2 best fit to u_0. Then (4.79) is equivalent to the mass matrix system

$$\tilde{M}\underline{U} = \theta(t)\underline{\tilde{c}} \tag{4.81}$$

with $\underline{\tilde{c}}$ as the vector containing entries $\tilde{c}_i$, and $\tilde{M}$ the mass matrix calculated for the new nodal positions. We may then solve for U with the boundary condition $\left[\tilde{W}_i \dot{x} U\right]_a^b = 0$ strongly imposed on U, without violating relative mass conservation.

Algorithm 7

For case 1 with fixed boundaries.
Having initial u_0 and x_0, and having calculated the piecewise linear function U_0 at the nodes X_0, as well as θ from (4.41), the finite element solution of Fisher's equation (4.39) on the moving mesh in 1-D consists of the following steps at each time t:

1. Find $\dot{\theta}(t)$ by evaluating (4.66);
2. Find the velocity potential by solving equation (4.73) for the $\Phi_j(t)$ values, with Φ specified at the central node;
3. Find the node velocity by solving (4.76) for the $\dot{X}$ values at internal nodes, strongly imposing the boundary conditions;

4. Generate the co-ordinate system $\underline{X}(t+dt)$ at the next time-step by evaluating (3.18) using the forward Euler approximation. Similarly, update $\dot{\theta}$ from $\dot{\theta}(t)$;

5. Find the solution $U(t+dt)$ by solving the relative conservation equation (4.81) using the strong form of the boundary conditions.

Case 2: Moving boundaries: Boundary conditions are $u=0$, $u\dot{x}=0$

For the free boundary, case 2, whilst we have Dirichlet conditions for u we do not have them for $\dot{x}$. In fact there are no boundary conditions to impose on $\dot{x}$. We will not need modified weight functions to obtain $\dot{x}$ and indeed, using them would prevent us from obtaining a solution for $\dot{x}$ at the boundaries $x=a$ and $x=b$. For this reason we proceed initially with unmodified weight functions. Substituting the standard piecewise linear approximations (3.41) and (3.42) into (4.53) we obtain

$$\sum_{j=0}^{N+1}\left[\int_{a(t)}^{b(t)} U\frac{\partial W_i}{\partial x}W_j dx\right]\dot{X}_j = \left[W_i\frac{\partial u}{\partial x}\right]_{a(t)}^{b(t)} - \sum_{j=0}^{N+1}\left[\int_{a(t)}^{b(t)}\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_{a(t)}^{b(t)} W_i U^2 dx - c_i\dot{\theta}. \tag{4.82}$$

The term $\left[W_i\frac{\partial u}{\partial x}\right]_{a(t)}^{b(t)}$ again remains non-zero for case 2 and must be included. Proceeding as for case 1, we obtain analogous equations for $\underline{\dot{X}}$ and $\underline{\Phi}$ which differ only from case 1 in that there is now no tilde denoting modified basis functions. We have the matrix equation to give $\underline{\dot{X}}$,

$$B(\underline{U})\underline{\dot{X}} = \underline{f} \tag{4.83}$$

where $\underline{\dot{X}} = \{\dot{X}_i\}$, and $\underline{f} = \{f_i\}$ given by

$$f_i = -\sum_{j=i-1}^{j=i+1}\left[\int_{a(t)}^{b(t)}\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_{a(t)}^{b(t)} W_i U^2 dx - \tilde{c}_i\dot{\theta} \tag{4.84}$$

for $i=[1,...,N]$. The nonlinear term is evaluated using Simpson's rule (4.64). For $i=0$,

$$f_0 = -\sum_{j=0}^{1}\left[\int_{a(t)}^{b(t)}\frac{\partial W_0}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_{a(t)}^{b(t)} W_0 U^2 dx - \tilde{c}_i\dot{\theta} - \left[W_0\frac{\partial u}{\partial x}\right]_{a(t)} \tag{4.85}$$

and for $i=N+1$,

$$f_{N+1} = -\sum_{j=N}^{N+1}\left[\int_{a(t)}^{b(t)}\frac{\partial W_{N+1}}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_j + \int_{a(t)}^{b(t)} W_{N+1} U^2 dx - \tilde{c}_i\dot{\theta} + \left[W_{N+1}\frac{\partial u}{\partial x}\right]_{b(t)}. \tag{4.86}$$

Matrix $B(\underline{U})$ is asymmetric and has entries

$$B(\underline{U})_{ij} = \int_{a(t)}^{b(t)} U \frac{\partial W_i}{\partial x} W_j dx. \tag{4.87}$$

For the same reasons as in case 1, we introduce a velocity potential Φ. The piecewise linear approximation for Φ is composed of unmodified basis functions so that $\Phi = \sum_j W_j \Phi_j$, and hence we obtain the matrix equation

$$K(\underline{U})\underline{\Phi} = \underline{g} \tag{4.88}$$

where $\underline{\Phi}$ is the vector containing $\{\Phi_j\}$, for $j = [0, ..., N+1]$. $K(\underline{U})$ is a symmetric positive definite matrix with entries

$$K(\underline{U})_{ij} = \int_{a(t)}^{b(t)} U \frac{\partial W_i}{\partial x} \frac{\partial W_j}{\partial x} dx \tag{4.89}$$

and $\underline{g}$ is a vector with entries

$$g_i = -\sum_{j=i-1}^{j=i+1} \left[\int_{a(t)}^{b(t)} \frac{\partial W_i}{\partial x} \frac{\partial W_j}{\partial x} dx \right] U_j + \int_{a(t)}^{b(t)} W_i U^2 dx - \tilde{c}_i \dot{\theta} + \left[W_i \frac{\partial u}{\partial x} \right]_{a(t)}^{b(t)}. \tag{4.90}$$

As before the nonlinear term is evaluated using Simpson's rule (4.64).

Recovering $\dot{X}$

We continue to use standard weight functions. Beginning with the weak form (4.74), and following the process described for case 1, we obtain the matrix form that will allow us to recover $\dot{X}$

$$M\underline{\dot{X}} = B\underline{\Phi}_j. \tag{4.91}$$

This differs from (4.76) for case 1 only in that basis functions are now standard, so that we have separate basis functions for each of the boundary nodes making two more in total. The matrix M is a mass matrix with entries

$$M_{ij} = \int_{a(t)}^{b(t)} W_i W_j dx \tag{4.92}$$

and B is the asymmetric matrix of (4.78):

$$B_{ij} = \int_{a(t)}^{b(t)} W_i \frac{\partial W_j}{\partial x} dx. \tag{4.93}$$

Finding X

A time integration approximation such as forward Euler is used to generate the grid at the new time step according the values of $\dot{X}$.

Transferring between weight function systems

Now we must recover U, but we have a complication. If we are to strongly impose the Dirichlet boundary conditions on U, we would require modified basis functions in order to preserve mass conservation. However, we have not been able to use modified weight functions thus far because of the need to obtain a value for $\dot{X}$ on the free boundary.
We note that we have the choice of two sets of weight functions (modified or unmodified) that move with the nodal velocities $\dot{X}_j$. We may write our finite element system in terms of either set, and all of the component equations of that system would then be mutually consistent. It is possible to derive a method by which we can transfer from one set of weight functions to another whilst preserving conservation of mass, as is required in this case. Having first determined the values of the $\dot{X}_j$ using the standard weight functions (so that we have results for all nodes), we then transfer to the modified weight function system to recover U. We proceed as follows in the modified basis function system. Rather than using the concise definition of the c_i, (4.43), we use equation (4.61) which is an expanded form. In terms of the modified $\tilde{c}_i$ this is, for $i = [1,2,3,...,N]$

$$\tilde{c}_i = \frac{1}{\dot{\theta}} \left(- \sum_{j=i-1}^{j=i+1} \left[\int_{x_{i-1}(t)}^{x_{i+1}(t)} U \frac{\partial \tilde{W}_i}{\partial x} W_j dx \right] \dot{X}_j - \sum_{j=i-1}^{j=i+1} \left[\int_{x_{i-1}(t)}^{x_{i+1}(t)} \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} dx \right] U_j \right.$$
$$\left. + \int_{x_{i-1}(t)}^{x_{i+1}(t)} \tilde{W}_i U^2 dx + \left[\tilde{W}_i \frac{\partial u}{\partial x} \right]_a^b \right). \tag{4.94}$$

Again the nonlinear term is evaluated using Simpson's rule (4.64). This gives the relationship between the $\tilde{c}_i$ and $\dot{X}$, for a given t. We may insert any chosen values of $\dot{X}$ into this and thus obtain the values of $\tilde{c}_i$ necessary to conserve relative proportions of total mass. We insert the values of $\dot{X}$ obtained from the unmodified system (4.91). The resulting $\tilde{c}_i$ values are defined using the modified weight functions and so are suitable for the recovery of U

using the modified conservation principle.

Recovering U

U can now be recovered in exactly the same way as in case 1, as described in section 4.2.1.

Algorithm 8

For case 2 with free boundaries.
Having initial u_0 and x_0, and having calculated the piecewise linear function U_0 at the nodes X_0, as well as θ from (4.41), the finite element solution of Fisher's equation (4.39) on the moving mesh in 1-D consists of the following steps at each time t:

1. Find $\dot{\theta}(t)$ by evaluating (4.66);
2. Find the velocity potential by solving equation (4.88) for the $\Phi_j(t)$ values, with Φ specified at the central node;
3. Find the node velocity by calculating (4.91) for the $\dot{X}$ values at all nodes including boundary nodes;
4. Generate the co-ordinates $X(t+dt)$ at the next time-step from (3.18) using the forward Euler approximation. Similarly, update $\dot{\theta}$ from $\dot{\theta}(t)$;
5. Find the solution $U(t+dt)$ by solving the relative conservation equation (4.81) using the strong form of the boundary conditions.

Results for the fixed boundary problem: Case 1

The finite element system was implemented using MATLAB. For computational efficiency we need to compute only half the domain ($x = [0, 0.5]$) since it is symmetric. We can use the node adjacent to $x = 0$ to provide variable values for the node that would have been on the other ($x < 0$) side of $x = 0$ if we had modelled the whole domain. Following [14], the initial condition used is $u(x,0) = 20\cos\pi x$.

The vertical lines on figure 4.5 represent the positions of the nodes with iteration number on the vertical axis. We can see that as we approach blow up the nodes are moving towards the centre as expected.

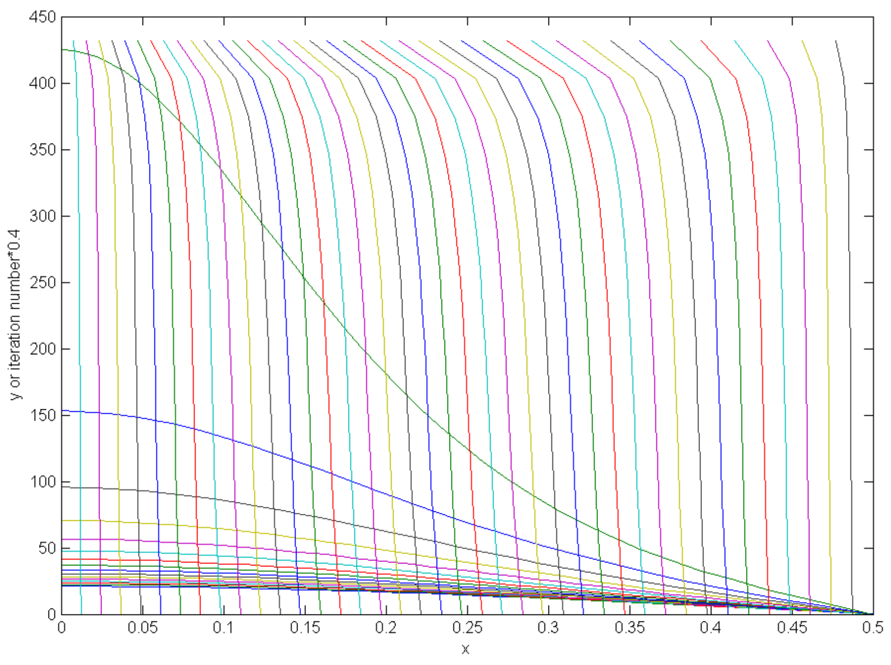


Fig. 4.5 A solution of the 1D Fisher's equation using a moving mesh with the fixed boundary of case 1. Here we use 40 elements and a time step of 0.00005. As the model approaches blow-up the nodes become co-located and the solution becomes unstable.

We test the model with a variety of time steps Δt and also a variety of initial mesh spacings Δx. The values for these were chosen to enable comparison with previously modelled methods for the Fisher's equation, notably Budd's 2005 paper [13], Edgington's M.Sc.[24] and Sarah Cole's M.Sc. [21]. The choice of time steps Δt is bounded by the stability condi-

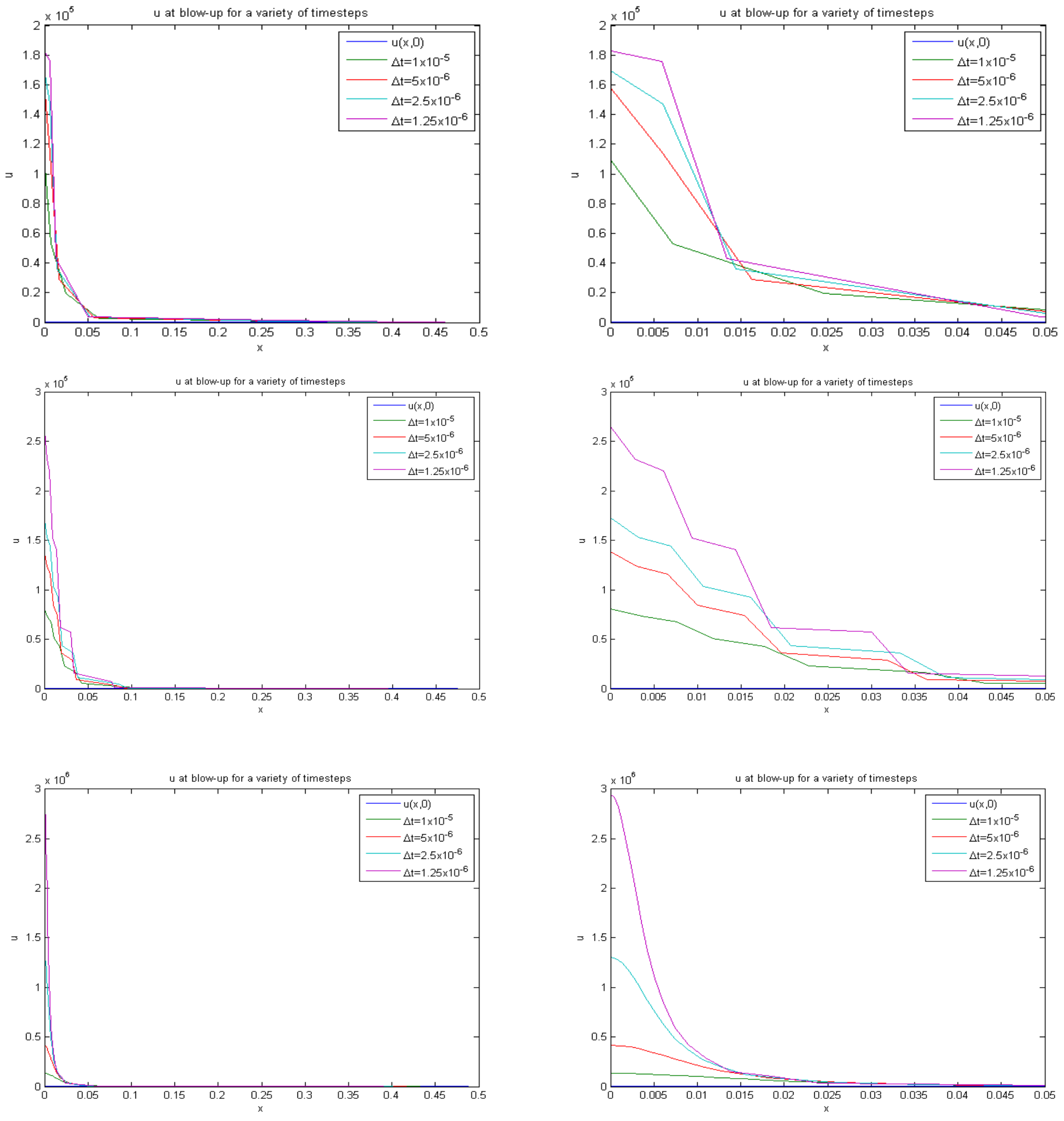


Fig. 4.6 Blow-up of solution $u(x,t)$ of Fisher's equation (4.39), with case 1 boundary conditions (fixed boundaries). The model is run to t=0.0825, beyond which solutions begin to suffer from node crossing and other instabilities. The precise time that this occurs for each choice of Δt and Δx is given in table 4.1. The grid resolution is 6 nodes (top), 11 nodes (centre) and 21 nodes (bottom) in the half domain shown. Initial spacing Δx is regular. Various Δt choices are tested for each initial grid resolution. The figures on the right show the detail at x close to 0. As Δx is reduced a small improvement in the definition of the peak is noted.

tion, that

$$\Delta t < \Delta x^2 = \left(\frac{0.5}{N}\right)^2 \tag{4.95}$$

which limits Δt to a maximum of $1.25x10^{-3}$. In both [21], and [24], Cole and Edgington attempt a moving mesh solution of the same problem. They use an implicit finite difference method to compute a conservation-based approach to moving the mesh. The results from our finite element method are consistent with the approximate blow up time $T \approx 0.082372$ given in [13], as table 4.1 shows. We define the blow-up time of the model as the point of failure of the model to further resolve a solution, *i.e.* nodes are crossing or some other catastrophic instability. Looking in more detail, we are able to resolve a higher peak for u at blow-up with values of the order of $u = 10^5$ (figure 4.6) rather than the $u = 10^4$ in [21]. We also observe a narrower, more defined peak in u at all values of Δx and Δt than in the Cole dissertation. Furthermore, we note from the results in [24] that the 11 node model performs better than the 6 node or 21 node models (defined as the number of nodes in the half-domain $0 \leq x \leq 0.5$). Presumably the 6 node model is limited by lack of resolution and the 21 node model is limited by node tangling. We do not see the same node tangling limit in the finite element implementation. In figure 4.6 we see that the 21 node finite element model gives a maximum u of the order of 10^6, whereas in the 21 node finite difference model the maximum u is of the order of 10^3 .

Variable Timesteps

We note that we use many more time steps than Cole because she increases the size of her time step as she approaches blow-up for reasons of numerical stability, whereas our time step is constant. We do not have this stability problem but for a fair comparison we will repeat the experiments with the variable time step from [21]. The time step used is

$$\Delta t = \frac{\Delta t_0}{T - t} \tag{4.96}$$

which has Δt increasing as $t \to T$, where T is the blow-up time. As before our reference T is taken from [13], and has the value $T \approx 0.082372$. Since [21] has some blow-up times greater than this value, and some that are impossible to generate using (4.96), we are not confident that the method used there is precisely as stated.

Figure 4.8 shows the results from the finite element method with the variable time step. With this constraint it is apparent that the finite element method does not converge to as narrow or high a peak in u as the finite difference method, for given initial values of Δt and

Δx, when the variable time steps are used.

Results for the moving boundary problem: Case 2

Edgington extends the work in [21] by examining the effect of allowing the boundary nodes to move. He finds that in the finite difference model, the maximum u achieved is reduced when the boundaries are allowed to move, except in the coarsest 11 node model.

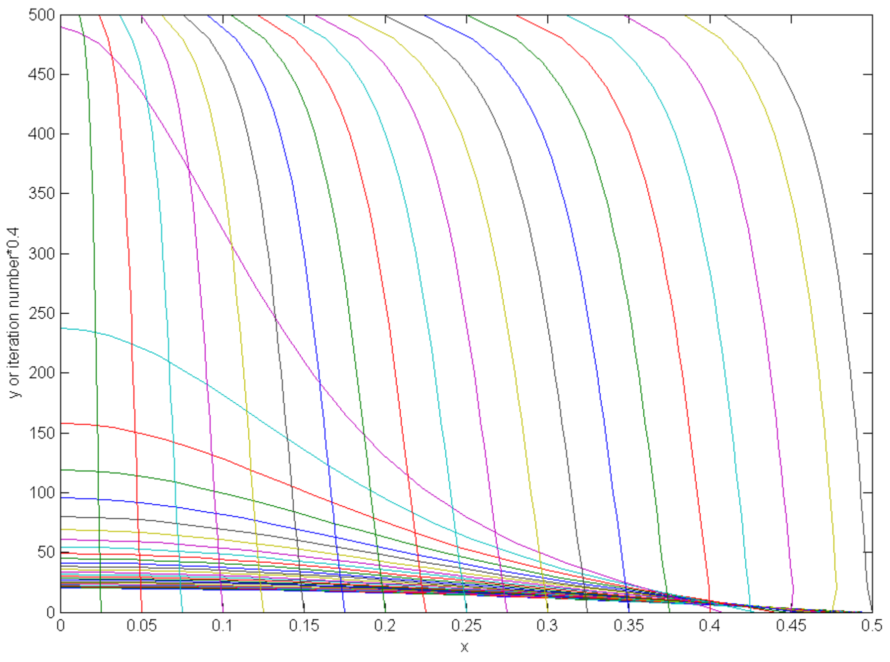


Fig. 4.7 A solution of the 1D Fisher's equation using a moving mesh with the free boundary of case 2. Here we use 20 elements and a time step of 0.00005.

We find that our results are somewhat mixed. The maximum u achieved is equalled or improved when the boundaries are allowed to move, when compared to the fixed boundary case. However it must be recognised that the problem is defined differently for each case. For the 21 node model, the maximum resolvable u_{max} is broadly unchanged at (near) blowup when the moving (case 2) and fixed boundary (case 1) versions are compared. For the 11 node model, allowing the moving boundary increases the resolvable u_{max} at blow-up by about a factor of 5. For the 6 node model, allowing the moving boundary increases the resolvable u_{max} at blow-up by about a factor of 10. However, the time taken to blow up is much less accurate with a moving boundary than with a static boundary. The model stops running due to nodes crossing at around $T = 0.065$ in the moving boundary case, whereas we would aim for the model to run to $T = 0.0823$. It is probable that this is due to the greater nodal velocities observed when a free boundary is present.

Table 4.1 Blow-up times from MMFEM implementation of Fisher's equation, case 1 with fixed time step

N	Δt	steps	$T_{blow-up}$
6	1x10^{-5}	8411	0.08411
6	5x10^{-6}	16815	0.08408
6	2.5x10^{-6}	33622	0.08406
6	1.25x10^{-6}	67237	0.08405
11	1x10^{-5}	8294	0.08294
11	5x10^{-6}	16582	0.08291
11	2.5x10^{-6}	33156	0.08289
11	1.25x10^{-6}	66304	0.08288
21	1x10^{-5}	8262	0.08262
21	5x10^{-6}	16517	0.08259
21	2.5x10^{-6}	33025	0.08256
21	1.25x10^{-6}	66042	0.08255
41	1x10^{-5}	8253	0.08253
41	5x10^{-6}	16500	0.08250
41	2.5x10^{-6}	32993	0.08250
41	1.25x10^{-6}	65977	0.08247

Table 4.2 Side by side comparison of blow-up times from moving finite difference method (left) and moving finite element method (right). The finite difference results are taken from variable time step method of [21]. Fixed boundaries are used in [21], so the equivalence is with case 1. We use the same variable time step method for the MMFEM to allow comparison. Dependence on number of nodes N in the half domain, and initial time step Δt_0 is shown.

Finite Difference				Finite Element			
N	Δt_0	steps	$T_{blow-up}$	N	Δt_0	steps	$T_{blow-up}$
6	$1.71\text{x}10^{-5}$	200	0.0774	6	$1\text{x}10^{-5}$	342	0.0815
6	$8.55\text{x}10^{-6}$	399	0.0797	6	$5\text{x}10^{-6}$	681	0.0808
6	$4.275\text{x}10^{-6}$	796	0.0806	6	$2.5\text{x}10^{-6}$	1360	0.0817
6	$2.1375\text{x}10^{-6}$	1591	0.0836	6	$1.25\text{x}10^{-6}$	2717	0.0817
11	$1\text{x}10^{-5}$	341	0.0786	11	$1\text{x}10^{-5}$	342	0.0815
11	$4.9\text{x}10^{-6}$	694	0.0793	11	$5\text{x}10^{-6}$	681	0.0808
11	$2.45\text{x}10^{-6}$	1387	0.0807	11	$2.5\text{x}10^{-6}$	1360	0.0817
11	$1.225\text{x}10^{-6}$	2773	0.0824	11	$1.25\text{x}10^{-6}$	2717	0.0817
21	$1\text{x}10^{-5}$	342	0.0812	21	$1\text{x}10^{-5}$	339	0.0755
21	$5\text{x}10^{-6}$	682	0.0835	21	$5\text{x}10^{-6}$	681	0.0808
21	$2.5\text{x}10^{-6}$	1361	0.0847	21	$2.5\text{x}10^{-6}$	1360	0.0817
21	$1.24\text{x}10^{-6}$	2740	0.0836	21	$1.25\text{x}10^{-6}$	2717	0.0817

Table 4.3 Blow-up times from MMFEM implementation of Fisher's equation, case 1 with smoothing, and fixed time steps. Blow-up happens later with smoothing.

N	Δt	steps	$T_{blow-up}$
6	1x10^{-5}	9821	0.0982
6	5x10^{-6}	19653	0.0983
6	2.5x10^{-6}	39317	0.0983
6	1.25x10^{-6}	78645	0.0983
11	1x10^{-5}	9892	0.0989
11	5x10^{-6}	19865	0.0993
11	2.5x10^{-6}	39817	0.0995
11	1.25x10^{-6}	79727	0.0997
21	1x10^{-5}	9691	0.0969
21	5x10^{-6}	19664	0.0983
21	2.5x10^{-6}	39663	0.0992
21	1.25x10^{-6}	79669	0.0996

Smoothing

We also note some saw-tooth instability in both the constant time step case and, to a lesser extent, the variable time step case finite element models. This is a common problem with finite element methods because of the central differences involved combined with explicit time stepping. We will attempt to smooth this out by introducing a viscosity term (Laplacian smoothing),

$$x_i^{new} = x_i + \frac{1}{4}\delta^2 x_i, \qquad \delta^2 = x_{i+1} - 2x_i + x_{i-1}. \tag{4.97}$$

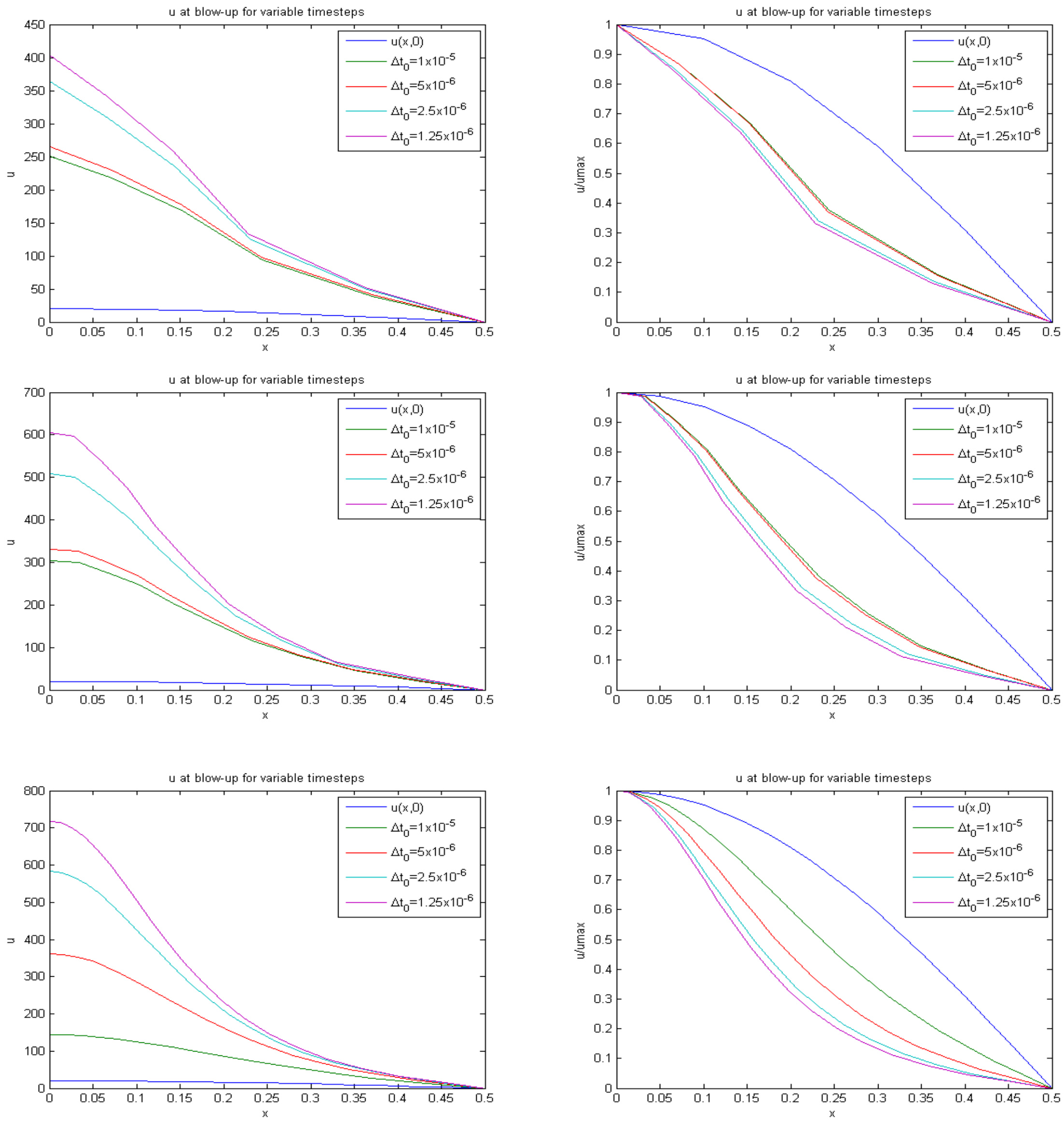


Fig. 4.8 Blow-up of the solution $u(x,t)$ of Fisher's equation (4.39) with fixed boundaries (case 1) and variable time steps. The models are run until t=0.0825. Grid resolutions are from top to bottom, 6 nodes, 11 nodes and 21 nodes. Variable time steps for comparison with [21] are used. The figures on the right show the results with a normalised u.

When the viscosity is applied, we can run the model for longer, past the expected blow-up time, because of changes in the model behaviour introduced by the smoothing. Table 4.3 shows the blow-up times predicted by this method (using a fixed time step). We are able to resolve much higher peaks in u, but it is still somewhat unsatisfactory because of the obvious inaccuracies in the blow-up time. This inaccuracy arises because the smoothing term tends to flatten out the natural concave shape of the solution, and in such a way moves the solution further away from a blow-up state. In effect we change the behaviour of the Fisher equation to one with viscosity. The effect of the viscosity term is to add to the rate of diffusion, and may cause a requirement for a correspondingly smaller Δt. We also note that the conservation principle is violated by the addition of (4.97), because we are adjusting nodal positions with no regard for the effect upon the relative distribution of mass. The intention is that the taking of such a liberty would at least provide the advantages of a stable implementation. However, with the effects on the shape of the solution that we see here, this approach is clearly not justifiable. The results for this method are shown in figure 4.9.

Order of convergence

Since we have a value for the blow up time from [13], we may examine the orders of convergence p or q with respect to time or space respectively. When Δt is varied with Δx held constant, we expect a fixed non-zero component of the spatial error, so we may estimate p and q by looking at the rate at which the differences between successive errors decrease. We assume

$$E_n = C(\Delta x)^q + D(\Delta t)^p \tag{4.98}$$

where the E_n are the errors in blow-up time, T, *i.e.*, $T - 0.082372$, and C and D are constants. For a fixed Δx, with Δt halving as n increases by 1, we have

$$E_{n+1} - E_n = D((\Delta t/2)^p - (\Delta t)^p) \tag{4.99}$$

$$= D(\Delta t)^p (1/2^p - 1) \tag{4.100}$$

and

$$(E_n - E_{n-1})/(E_{n+1} - E_n) = D(2\Delta t)^p (1/2^p - 1)/D(\Delta t)^p (1/2^p - 1) \tag{4.101}$$

$$= 2^p \tag{4.102}$$

Similarly if Δt is fixed and Δx is halved as n increases by 1, we have

$$(E_n - E_{n-1})/(E_{n+1} - E_n) = C(2\Delta x)^q (1/2^p - 1)/C(\Delta x)^q (1/2^q - 1) \tag{4.103}$$

$$= 2^q \tag{4.104}$$

We have examined the change in the error when either the time step or the node spacing is varied (table 4.4).

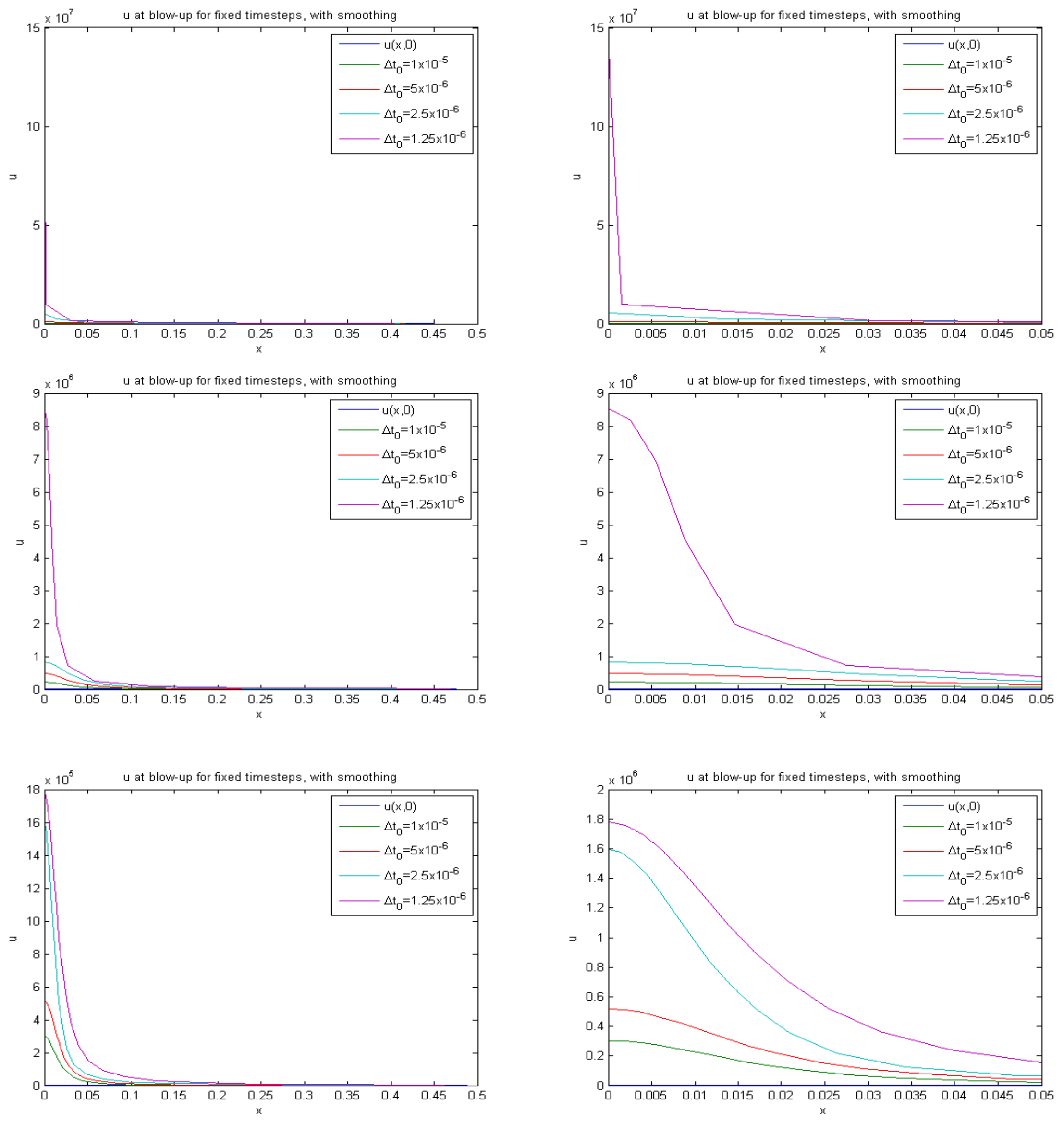


Fig. 4.9 The smoothed, fixed time step results at t=0.0825 for blow-up of the solution $u(x,t)$ of Fisher's equation (4.39) with fixed boundaries (case 1). Grid resolutions are from top to bottom, 6 nodes, 11 nodes and 21 nodes in the half domain. The figures on the right show the same results as the figures on the left but with a change of scale on the x axis. As Δx is reduced the peak actually gets wider as the smoothing becomes more effective.

Table 4.4 Errors in blow-up time from MMFEM implementation of Fisher's equation, case 1 with fixed time step, and their variation by time step and node spacing

N	Δx	Δt	E_n	$E_n - E_{n-1}$	$\frac{E_n - E_{n-1}}{E_{n+1} - E_n}$
21	0.025	1x10^{-5}	0.000248		
21	0.025	5x10^{-6}	0.000213	-0.000035	1.6
21	0.025	2.5x10^{-6}	0.000191	-0.000022	2.0
21	0.025	1.25x10^{-6}	0.000180	-0.00011	2.2
21	0.025	6.25x10^{-7}	0.000175	-0.00005	
6	0.1	1.25x10^{-6}	0.001674		
11	0.05	1.25x10^{-6}	0.000508	-0.001166	3.6
21	0.025	1.25x10^{-6}	0.000180	-0.000328	4.0
41	0.0125	1.25x10^{-6}	0.000099	-0.000081	4.8
81	0.00625	1.25x10^{-6}	0.000082	-0.000082	

We see in table 4.4 that successive differences between errors as you halve Δt go down by a factor of about 2, suggesting $p \approx 1$ or first order in time. When you halve Δx these differences go down by a factor of about 4, suggesting $q \approx 2$ or second-order accuracy in space. This is as expected from forward Euler in time and linear finite elements in space.

4.2.2 Fisher's Equation in 2D

The two dimensional solution of the Fisher's equation has not previously been attempted with a moving mesh, as far as we are aware, in either radial or fully 2-D form. The conservation-based MMFEM is presented here for the fully two dimensional case. The 2-D form of Fisher's equation with $p = 2$ is

$$\frac{\partial u}{\partial t} = \nabla^2 u + u^2. \tag{4.105}$$

We consider the domain $\Omega(t)$ with free boundary $S(t)$ and the Dirichlet boundary conditions $u(S,t) = 0$. This is analogous to case 2 in the 1-D version of the Fisher's equation. The weak form of (4.105) is

$$\int_{\Omega(t)} w_i \frac{\partial u}{\partial t}\, d\Omega = \int_{\Omega(t)} (w_i \nabla^2 u + w_i u^2)\, d\Omega. \tag{4.106}$$

Define $\theta(t)$ as the volume (mass) under the solution u, given by

$$\int_{\Omega(t)} u\, d\Omega = \theta(t). \tag{4.107}$$

Because

$$\frac{1}{\theta(t)} \int_{\Omega(t)} u\, d\Omega = 1 \tag{4.108}$$

we can define a distributed conservation principle in terms of weight functions w_i as

$$\int_{\Omega(t)} w_i u\, d\Omega = c_i \theta(t). \tag{4.109}$$

Providing that the set of weight functions w_i forms a partition of unity, $\sum_i w_i = 1$, equation (4.109) will allow us to obtain a set of values c_i also forming a partition of unity, $\sum_i c_i = 1$. The constants c_i are independent of t and are determined by the initial spacing and the initial data. Differentiating (4.109) with respect to time gives

$$\frac{d}{dt} \int_{\Omega(t)} w_i u\, d\Omega = c_i \frac{d\theta}{dt} = c_i \dot{\theta}. \tag{4.110}$$

We define a reference test domain $\Omega(0)$ at $t=0$ and a moving test domain $\Omega(t)$. Applying the Reynolds Transport Theorem we obtain

$$\begin{aligned}\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega &= \int_{\Omega(t)} \frac{\partial}{\partial t}(w_i u)\; d\Omega + \int_{\Omega(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}}\; dS \\ &= \int_{\Omega(t)} \left(w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + \nabla \cdot (w_i u \dot{\mathbf{x}}) \right) \, d\Omega \end{aligned} \tag{4.111}$$

for the generalised weak form, where $\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}$ is any normal velocity consistent with the normal boundary velocity. Using the advection equation (3.7) we can cancel out terms giving us the weak form of the Reynolds Transport Theorem in the moving frame,

$$\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}})\; d\Omega = \int_{\Omega(t)} w_i \frac{\partial u}{\partial t}\; d\Omega. \tag{4.112}$$

We now consider the specific system described by Fisher's equation. We substitute the weak form of Fisher's equation (4.106), and obtain

$$\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}})\; d\Omega = \int_{\Omega(t)} (w_i \nabla^2 u + w_i u^2)\; d\Omega. \tag{4.113}$$

Integrating by parts on the right gives

$$\frac{d}{dt}\int_{\Omega(t)} w_i u \, d\Omega - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}})\; d\Omega = \int_{S(t)} w_i \nabla u \cdot \hat{\mathbf{n}}\; dS - \int_{\Omega(t)} \nabla w_i \cdot \nabla u\; d\Omega + \int_{\Omega(t)} w_i u^2\; d\Omega. \tag{4.114}$$

We now use the relative conservation principle (4.109). From (4.110),

$$c_i \dot{\theta} - \int_{\Omega(t)} w_i \nabla \cdot (u\dot{\mathbf{x}})\; d\Omega = \int_{S(t)} w_i \nabla u \cdot \hat{\mathbf{n}}\; dS - \int_{\Omega(t)} \nabla w_i \cdot \nabla u\; d\Omega + \int_{\Omega(t)} w_i u^2\; d\Omega. \tag{4.115}$$

After integration by parts we obtain

$$\begin{aligned} &c_i \dot{\theta} - \int_{S(t)} w_i u \dot{\mathbf{x}} \cdot \hat{\mathbf{n}}\; dS + \int_{\Omega(t)} u\dot{\mathbf{x}} \cdot \nabla w_i\; d\Omega = \\ &\quad \int_{S(t)} w_i \nabla u \cdot \hat{\mathbf{n}}\; dS - \int_{\Omega(t)} \nabla w_i \cdot \nabla u\; d\Omega + \int_{\Omega(t)} w_i u^2\; d\Omega \end{aligned} \tag{4.116}$$

where $\hat{\mathbf{n}}$ is the outward pointing unit normal. The boundary flux $u\dot{\mathbf{x}}\cdot\hat{\mathbf{n}}$ is zero due to the Dirichlet condition on u. We now have an equation for $\dot{\mathbf{x}}$ in terms of u and $\dot{\theta}$,

$$c_i\dot{\theta} + \int_{\Omega(t)} u\dot{\mathbf{x}}\cdot\nabla w_i \; d\Omega = \int_{S(t)} w_i\nabla u\cdot\hat{\mathbf{n}} \; dS - \int_{\Omega(t)} \nabla w_i\cdot\nabla u \; d\Omega + \int_{\Omega(t)} w_i u^2 \; d\Omega. \quad (4.117)$$

Providing that we select weight functions w_i that form a partition of unity, $\sum w_i = 1$, we can calculate $\dot{\theta}$ by summing this expression over all weight functions in the model and using the boundary conditions. From (4.109), we define c_i as the proportion of mass associated with a particular weight function w_i. The sum is

$$\begin{aligned}\sum_i c_i\dot{\theta}(t) - \left(\int_{S(t)} \sum_i w_i u\dot{\mathbf{x}}\cdot\hat{\mathbf{n}} \; dS\right) + \left(\int_{\Omega(t)} u\dot{\mathbf{x}}\cdot\nabla\left(\sum_i w_i\right) \; d\Omega\right) \\ = \left(\int_{S(t)} \sum_i w_i\nabla u\cdot\hat{\mathbf{n}} \; dS\right) - \left(\int_{\Omega(t)} \nabla\left(\sum_i w_i\right)\cdot\nabla u \; d\Omega\right) + \left(\int_{\Omega(t)} \sum_i w_i u^2 \; d\Omega\right).\end{aligned} \quad (4.118)$$

Since $u\dot{\mathbf{x}}\cdot\hat{\mathbf{n}}$ is zero on the boundary,

$$\dot{\theta}(t) = \int_{S(t)} w_i\nabla u\cdot\hat{\mathbf{n}} \; dS + \int_{\Omega(t)} u^2 \; d\Omega \quad (4.119)$$

will give us $\dot{\theta}$.

In the same way as in the 1-D case, a velocity potential ϕ is defined by

$$\dot{\mathbf{x}} = \nabla\phi \quad (4.120)$$

and equation (4.116) can then be rewritten as

$$\begin{aligned}c_i\dot{\theta}(t) - \int_{S(t)} w_i u\nabla\phi\cdot\hat{\mathbf{n}} \; dS + \int_{\Omega(t)} u\nabla\phi\cdot\nabla w_i \; d\Omega \\ = \int_{S(t)} w_i\nabla u\cdot\hat{\mathbf{n}} \; dS - \int_{\Omega(t)} \nabla w_i\cdot\nabla u \; d\Omega + \int_{\Omega(t)} w_i u^2 \; d\Omega\end{aligned} \quad (4.121)$$

or, since we have zero Dirichlet conditions on u,

$$c_i\dot{\theta}(t) + \int_{\Omega(t)} u\nabla\phi\cdot\nabla w_i \; d\Omega = \int_{S(t)} w_i\nabla u\cdot\hat{\mathbf{n}} \; dS - \int_{\Omega(t)} \nabla w_i\cdot\nabla u \; d\Omega + \int_{\Omega(t)} w_i u^2 \; d\Omega. \quad (4.122)$$

This expression gives ϕ for given u and $\dot{\theta}$, which can be obtained from (4.119). The implications of the Helmholtz decomposition [27] tell us that (4.122) will have a unique solution if either ϕ or $\frac{\partial \phi}{\partial \hat{\mathbf{n}}}$ are given on the boundary S. We choose $\phi = 0$ on S. The remaining part of the method does not differ from the earlier examples in 1-D. Having obtained ϕ, we are able to recover $\dot{x}$ from the definition (4.120). We move the nodes and calculate θ at the new time step using the chosen integration scheme, and then obtain u at the new time step from the distributed mass conservation principle (4.109). Again, the chosen numerical method for these steps is the finite element method, so we now write these steps in finite element form.

Finite element form

In the same way as for the 1-D case, we may write our system in terms of modified or standard 2-D basis functions. First presented in [33], the use of the modified basis functions allows us to strongly impose the Dirichlet boundary conditions on u without violating relative conservation of mass. We do not then, however, have an equation for the boundary velocities, and to solve for u we first require knowledge of all the values of $\dot{\mathbf{x}}$ including boundary values. We now follow the same process as for the 1-D case. We first write the system in terms of standard basis functions. This allows us to obtain values for $\dot{\mathbf{x}}$ including at the boundaries. We then rewrite the system in terms of modified basis functions. We write our conservation principle (4.109) for the modified system in the expanded ALE form. We impose the values of $\dot{\mathbf{x}}$ obtained from the standard system into the modified system, and obtain the values of c_i that are consistent with the $\dot{\mathbf{x}}$. After time integration, we recover U from the modified conservation principle, allowing strong imposition of U on the boundaries.

We proceed as described, with standard basis functions W_i. We make the piecewise linear approximations

$$U(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)U_j(t) \tag{4.123}$$

$$\dot{\mathbf{X}}(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)\dot{\mathbf{X}}_j(t) \tag{4.124}$$

and

$$\Phi(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)\Phi_j(t) \tag{4.125}$$

giving

$$\nabla\Phi(\mathbf{x},t) = \sum_{j=1}^{N} \nabla W_j(\mathbf{x},t)\Phi_j(t). \tag{4.126}$$

We can now write (4.122) in the form

$$\begin{aligned}\sum_{j=1}^{N}\left[\int_{\Omega(t)} U\nabla W_i\cdot\nabla W_j\; d\Omega\right]\Phi_j &= \int_{S(t)} W_i\nabla U\cdot\hat{\mathbf{n}}\; dS \\ &-\sum_{j=1}^{N}\left[\int_{\Omega(t)} \nabla W_i\cdot\nabla W_j\; d\Omega\right]U_j + \int_{\Omega(t)} W_i U^2\; d\Omega - c_i\dot{\theta}(t)\end{aligned} \tag{4.127}$$

or in matrix form

$$K(\underline{U})\underline{\Phi} = \underline{f} \tag{4.128}$$

with the vector $\underline{\Phi}$ containing the values Φ_i, and the vector $\underline{f}$ containing the values f_i given by

$$f_i = \int_{S(t)} W_i\nabla U\cdot\hat{\mathbf{n}}\; dS - \sum_{j=1}^{N}\left[\int_{\Omega(t)} \nabla W_i\cdot\nabla W_j\; d\Omega\right]U_j + \int_{\Omega(t)} W_i U^2\; d\Omega - c_i\dot{\theta}(t). \tag{4.129}$$

The nonlinear term, $\int_{\Omega(t)} W_i U^2\; d\Omega$, is evaluated using Gaussian quadrature (see Appendix B). Whilst not exact, the order of accuracy is high enough not to affect the order of accuracy of the complete algorithm.

$K(\underline{U})$ is the weighted stiffness matrix with elements

$$K(\underline{U})_{ij} = \int_{\Omega(t)} U\nabla W_i\cdot\nabla W_j\; d\Omega. \tag{4.130}$$

We obtain $\dot{\mathbf{X}}$ from the finite element approximation of (4.120), for which the process is described in detail in Chapter 3, section 3.1.3. This gives the matrix form

$$M\underline{\dot{\mathbf{X}}} = B\underline{\Phi} \tag{4.131}$$

where $\underline{\dot{\mathbf{X}}} = \{\dot{\mathbf{x}}_i\}$, M is the standard mass matrix and B is an asymmetric matrix with elements

$$B_{ij} = \int_{\Omega(t)} W_i\nabla W_j\; d\Omega. \tag{4.132}$$

Modified weight functions in 2-D

Having obtained $\dot{\mathbf{X}}$ we now rewrite the system in terms of modified weight functions, so that the Dirichlet condition on U can be imposed. Modified weight functions in this context are any suitable set of piecewise linear weight functions where the weighting normally associated with a boundary node has been transferred to an internal node, and where a partition of unity is preserved. We turn our attention firstly to describing our system in terms of modified weight functions, and afterwards will take a closer look at the form of these functions and how they may be used in calculating matrices. We use the tilde to denote the use of the modified weight functions, *i.e.* $w_i = \tilde{W}_i(x,y)$.

For the approximations to variables, we continue to make piecewise linear approximations in terms of standard (unmodified) basis functions,

$$U(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)U_j(t) \tag{4.133}$$

$$\dot{\mathbf{X}}(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)\dot{\mathbf{X}}_j(t). \tag{4.134}$$

The ALE equation (4.117) can now be written, with a little rearrangement, in terms of modified weight functions $\tilde{W}_i$ and unmodified basis functions W_j as

$$\begin{aligned}\tilde{c}_i = \frac{1}{\dot{\theta}(t)}\Bigg(&-\sum_{j=1}^{N}\left[\int_{\Omega(t)} U\tilde{W}_i \cdot \nabla W_j \; d\Omega\right]\dot{\mathbf{X}}_j + \int_{S(t)} \tilde{W}_i \nabla U \cdot \hat{\mathbf{n}} \; dS + \int_{S(t)} \tilde{W}_i U \dot{\mathbf{X}} \cdot \hat{\mathbf{n}} \; dS \\ &-\sum_{j=1}^{N}\left[\int_{\Omega(t)} \nabla\tilde{W}_i \cdot \nabla W_j \; d\Omega\right] U_j + \int_{\Omega(t)} \tilde{W}_i U^2 \; d\Omega\Bigg).\end{aligned} \tag{4.135}$$

We impose our $\dot{\mathbf{X}}$ obtained from the unmodified system into this modified system, and thus obtain the correct values of $\tilde{c}_i$ for the modified system. The nonlinear term is calculated using Gaussian quadrature (see Appendix B). After time integration, we recover U using the finite element version of (4.109) with modified weight functions

$$\sum_{j=1, j\notin S}^{N}\left[\int_{\Omega(t)} \tilde{W}_i W_j \; d\Omega\right] U_j = \tilde{c}_i \theta(t) \tag{4.136}$$

which references internal nodes only. In matrix form this is

$$\tilde{M}\underline{U} = \underline{\tilde{c}}\theta \tag{4.137}$$

for mass matrix $\tilde{M}$, and vectors $\underline{U}$ containing the U_j values and $\underline{\tilde{c}}$ containing the $\tilde{c}_i$ values.

We now turn our attention to selecting a form for the modified weight functions and consider the implications for matrix construction. Following Hubbard, Baines and Jimack, 2009 [33] we are presented with the choice between two approaches for modifying the weight functions. These are termed the 'averaged modified approach' and the 'compact modified approach'. The two approaches are both derived and discussed only in the context of the mass matrix. The modified mass matrix alone is sufficient to solve the conservation equation (4.137), but here we require a more extensive implementation of the modified weight functions. In order to solve (4.135) we will require an evaluation of both a stiffness matrix and an asymmetric matrix. We must therefore extend one of the approaches from [33] in order to provide a way to construct any matrix from the modified weight functions. The averaged modified approach of [33] lends itself best to this, since it is defined in terms of the weight functions themselves. In [33] the modified weight functions are constructed in a similar way to the 1-D case, but with the added complication of increased connectivity. It is stated that the weight functions associated with boundary nodes are redistributed equally between their adjacent internal nodes. Therefore all basis functions defined on fully internal elements remain unaffected. With regard to the construction of the mass matrix, [33] sets out the following process. For triangles with two nodes on the boundary, all the weight associated with that triangle has only one internal node to go to, and the calculation is simple. For a given internal node j on a triangle with vertices $[j,J,K]$ where J and K are boundary nodes, the modified weight function $\tilde{W}_j\big|_{[j,J,K]}$ for triangle $[j,J,K]$ is given by

$$\tilde{W}_j\big|_{[j,J,K]} = W_j + W_J + W_K. \qquad (4.138)$$

An example of such a triangle is number 3 of figure 4.10.

For triangles with one node on the boundary, the weight associated with that node is split equally between the two internal nodes. For a given internal node j on a triangle with vertices $[i,j,J]$ where only J is on the boundary, the modified weight function $\tilde{W}_j\big|_{[i,j,J]}$ for triangle $[i,j,J]$ is given by

$$\tilde{W}_j\big|_{[i,j,J]} = W_j + \frac{1}{2}W_J. \qquad (4.139)$$

An example of such a triangle is number 2 of figure 4.10.

These sums are presented visually in figure 4.11. Recalling the standard 2-D basis functions W_i of figure 3.2, we obtain from equations (4.138) and (4.139) the coloured prisms $\tilde{W}_i$ of figure (4.11). The red volume represents $\tilde{W}_j\big|_{[j,J,K]}$, the contribution from triangle 3 to the modified basis function at internal node j. All the mass from triangle 3 has been as-

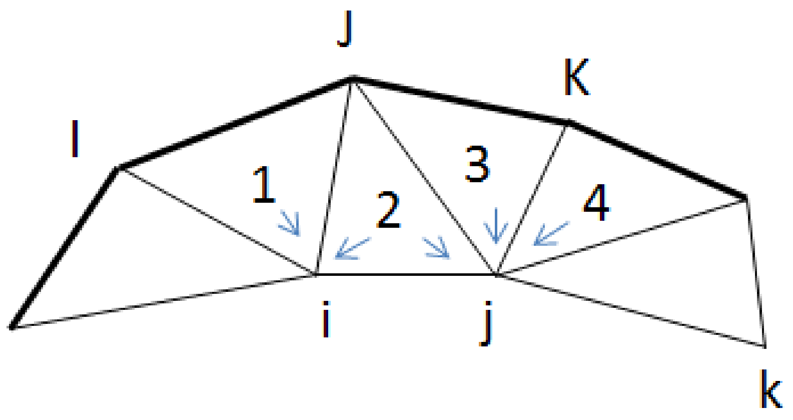


Fig. 4.10 Connectivity between boundary nodes (I,J and K) and internal nodes (i,j, and k). The arrows show where the weight function from each triangle will be transferred to under the modified system

signed to node j, so the red modified basis function is a triangular prism with height 1 at all three corners. The purple volume represents $\tilde{W}_j\big|_{[i,j,J]}$, the contribution from triangle 2 to the modified basis function at internal node j. Half of the mass normally assigned to boundary node J is transferred to internal node j, with the remaining half being transferred to internal node i. The purple modified basis function is therefore a modified prism with height 1 at j, height 0 at i and height 0.5 at J.

The practical implementation of this modification process takes place at the level of matrix assembly. The 2-D matrices are assembled as part of the algorithm by summing the element contributions from each triangle. When we require a matrix calculated from modified weight functions such as the $\tilde{M}$ of (4.137), the contributions from boundary triangles are adjusted before assembly according to (4.138) and (4.139). Contributions from triangles with no boundary nodes are unaffected.

The matrix assembly using these modified functions must consider the interactions between modified weight functions and unmodified basis functions. A generalised matrix A defined in terms of functions F and G with standard weight functions W_i and basis functions W_j has entries

$$A_{ij} = \int_{\Omega} F(W_i)G(W_j)\ d\Omega \tag{4.140}$$

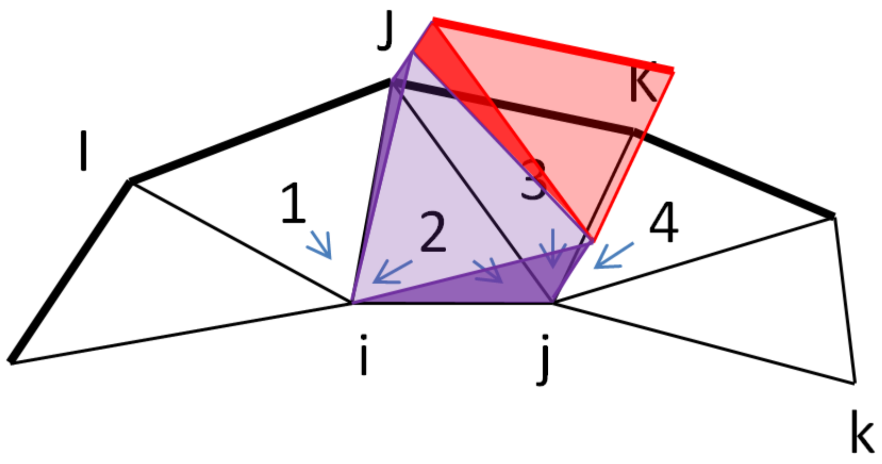


Fig. 4.11 Modified basis functions for internal nodes. The red modified weight function represents the mass contribution from triangle 3 to internal node j, and is a triangular prism with height 1 at all three corners. The purple modified basis function represents the mass contribution from triangle 2 to internal node j, and is a modified prism with height 1 at j, height 0 at i and height 0.5 at J.

with a corresponding element matrix given by

$$A_e = \begin{pmatrix} e_{ii} & e_{ij} & e_{ik} \\ e_{ji} & e_{jj} & e_{jk} \\ e_nki & e_{kj} & e_{kk} \end{pmatrix}. \tag{4.141}$$

We now consider the example of triangle 2 of figure (4.11), with one boundary node J replacing k and two internal nodes i and j. For an element with one boundary node such as triangle 2, we instead require a modified matrix with entries

$$\tilde{A}_{ij} = \int_{\omega_{e_2}} F(\tilde{W}_i)G(W_j)\ d\Omega \tag{4.142}$$

where ω_{e_2} is triangle 2 of figure (4.11). The unmodified basis functions W_j for this triangle are

$$W_i|_{[i,j,J]} \tag{4.143}$$

$$W_j\big|_{[i,j,J]} \tag{4.144}$$

and

$$W_J\big|_{[i,j,J]}\,. \tag{4.145}$$

The modified weight functions $\tilde{W}_i$ for triangle 2 are, in terms of those unmodified basis functions,

$$\tilde{W}_i\big|_{[i,j,J]} = W_i\big|_{[i,j,J]} + \frac{1}{2}\,W_J\big|_{[i,j,J]} \tag{4.146}$$

$$\tilde{W}_j\big|_{[i,j,J]} = W_j\big|_{[i,j,J]} + \frac{1}{2}\,W_J\big|_{[i,j,J]} \tag{4.147}$$

and

$$\tilde{W}_J\big|_{[i,j,J]} = 0. \tag{4.148}$$

The entries for the modified element matrix as defined by (4.142) can be calculated for triangle 2 from the local W_j and $\tilde{W}_i$ functions , (4.143) to (4.148). By reference to (4.140) and (4.141), the entries can be given in terms of the unmodified elements of (4.141) as

$$\tilde{A}_e = \begin{pmatrix} e_{ii} + \frac{1}{2}e_{iJ} & e_{ij} + \frac{1}{2}e_{jJ} & e_niJ + \frac{1}{2}e_nnJJ \\ e_{ji} + \frac{1}{2}e_nniJ & e_nnjj + \frac{1}{2}e_njJ & e_nnjJ + \frac{1}{2}e_nnJJ \\ 0 & 0 & 0 \end{pmatrix}. \tag{4.149}$$

The matrix is partitioned into an upper left 2×2 matrix, a bottom row of all zeros, and a right hand column which refers to a known value obtained from the Dirichlet condition. For example to calculate $A\underline{U}$, we can see that we have

$$\left(\begin{array}{cc|c} e_nii + \frac{1}{2}e_nniJ & e_{ij} + \frac{1}{2}e_{jJ} & e_{iJ} + \frac{1}{2}e_nJJ \\ e_{ji} + \frac{1}{2}e_niJ & e_{jj} + \frac{1}{2}e_{jJ} & e_{jJ} + \frac{1}{2}e_{JJ} \\ \hline 0 & 0 & 0 \end{array}\right) = \left(\begin{array}{c} U_i \\ U_j \\ \hline U_J \end{array}\right). \tag{4.150}$$

where U_i and U_j are free and U_J is fixed. The known terms generated by the right hand column of the matrix can be added directly into the rows of the calculation, allowing us to use only the square matrix of the upper left in the matrix operation. This has the advantage of being invertible.

We can use this approach to generate the specific matrices we will use. The unmodified mass matrix given by

$$M = \int_\Omega W_i W_j \; d\Omega \tag{4.151}$$

has the element mass matrix

$$M_e = area_\triangle \begin{pmatrix} \frac{1}{6} & \frac{1}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{12} & \frac{1}{6} \end{pmatrix} \tag{4.152}$$

and the modified mass matrix given by

$$\tilde{M} = \int_\Omega \tilde{W}_i W_j \; d\Omega \tag{4.153}$$

has the element mass matrix (for a triangle such as ω_{e_2} with two internal nodes and one boundary node) given by

$$M_{e_2} = area_\triangle \begin{pmatrix} \frac{5}{24} & \frac{3}{24} & \frac{1}{6} \\ \frac{3}{24} & \frac{5}{24} & \frac{1}{6} \\ 0 & 0 & 0 \end{pmatrix}. \tag{4.154}$$

We can calculate modified stiffness matrices in the same way. The standard element stiffness matrix is

$$K_e = \frac{1}{2} \begin{pmatrix} \cot\gamma + \cot\beta & -\cot\gamma & -\cot\beta \\ -\cot\gamma & \cot\alpha + \cot\gamma & -\cot\alpha \\ -\cot\beta & -\cot\alpha & \cot\beta + \cot\alpha \end{pmatrix} \tag{4.155}$$

and the modified element stiffness matrix for a triangle such as ω_{e_2} with two internal nodes and one boundary node is given by

$$\tilde{K}_{e_2} = \frac{1}{2}\begin{pmatrix} \cot\gamma + \frac{1}{2}\cot\beta & -\cot\gamma - \frac{1}{2}\cot\alpha & \frac{1}{2}\cot\alpha - \frac{1}{2}\cot\beta \\ -\cot\gamma - \frac{1}{2}\cot\beta & \frac{1}{2}\cot\alpha + \cot\gamma & \frac{1}{2}\cot\beta - \frac{1}{2}\cot\alpha \\ 0 & 0 & 0 \end{pmatrix}. \tag{4.156}$$

In the same way, the modified weighted stiffness matrix given by

$$\tilde{K}(\underline{U})_{ij} = \int_{\Omega_i} U\nabla W_i W_j \; d\Omega. \tag{4.157}$$

for which the standard element stiffness matrix is given by (3.82), has a modified element stiffness matrix (for a triangle such as ω_{e_2} with two internal nodes and one boundary node) given by

$$K(\tilde{\underline{U}})_{e_2} = \left(\frac{U_A + U_B + U_C}{6}\right)\begin{pmatrix} \cot\gamma + \frac{1}{2}\cot\beta & -\cot\gamma - \frac{1}{2}\cot\alpha & \frac{1}{2}\cot\alpha - \frac{1}{2}\cot\beta \\ -\cot\gamma - \frac{1}{2}\cot\beta & \frac{1}{2}\cot\alpha + \cot\gamma & \frac{1}{2}\cot\beta - \frac{1}{2}\cot\alpha \\ 0 & 0 & 0 \end{pmatrix}. \tag{4.158}$$

Algorithm 9

The finite element solution of Fisher's equation (4.105) on the moving mesh in 2-D therefore consists of the following steps:

1. Find $\dot{\theta}(t)$ by summing over all rows of the matrix equation (4.127);
2. Find the velocity potential by solving equation (4.127) for the $\Phi_j(t)$ values;
3. Find the node velocity by solving equation (4.131) for the $\dot{\mathbf{X}}_j(t)$ values;
4. Generate the co-ordinate system at the next time-step $t + dt$ by solving (3.18) using the forward Euler approximation. Similarly, update $\dot{\theta}$ from $\dot{\theta}(t)$;
5. Find the updated $\tilde{c}_i$ values using the calculated node velocities $\dot{\mathbf{X}}_j(t)$ in the ALE equation (4.135);

6. Find the solution $\underline{U}(t+dt)$ by solving the relative conservation equation (4.137) with the updated $\tilde{c}_i$ values.

Results

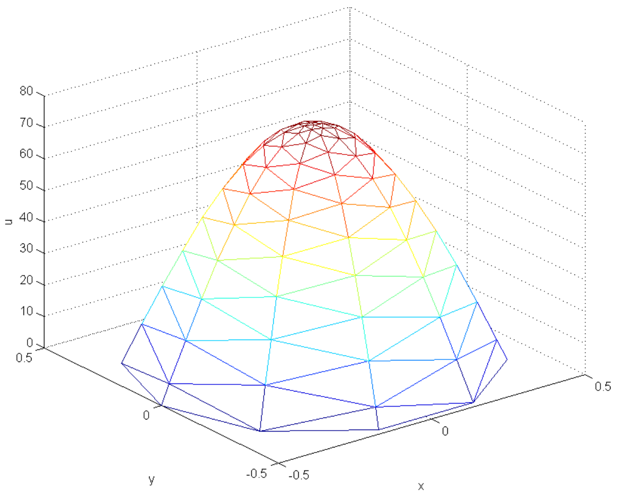


Fig. 4.12 Initial data (4.159) taken from [13] for the solution of the 2D Fisher's equation using a moving mesh. Here we use 5 nodes on 20 concentric circles, and a time step of $dt = 10^{-5}$. The mesh is assembled so that node positions alternate on adjacent concentric circles, see figure (4.17) for clarity.

We first test the model in a radial geometry with initial conditions taken from the 1-D case [13]. The initial domain is a circle of radius 0.5 centred at the origin, and initial u is given by, for radius r,

$$u(0) = 20\sin(\pi(0.5-r)). \tag{4.159}$$

We find that with these initial conditions in 2-D, the reaction does not build. The diffusion term overwhelms the reaction term and the system cools; the integral of u reduces over time. This can be seen in the calculation of (4.127), which gives a negative value for $\dot{\theta}$. The result makes intuitive sense, given that heat diffusion, or cooling, is taking place around the entire circumference of a circle rather that simply at a point on a line. We therefore provide a set of alternative initial conditions, with the same form but a higher amplitude. We calculate this amplitude so that the total rate of reaction and the total rate of diffusion (given by the terms in (4.127)) are in the same ratio as in the 1-D case. These initial conditions are given by

$$u(0) = 75\sin(\pi(0.5 - r). \tag{4.160}$$

In this case, the reaction does build, and we observe blow-up in a similar manner to the 1-D case. We observe node movement towards the centre as u becomes large there. The solution tends towards a Dirac delta function before the model collapses due to node tangling. These results are presented in figures 4.12 to 4.16. We use 5 nodes on 20 concentric circles. The initial grid is presented in figure 4.17. Note that the outermost circle is different in having 10 nodes. This is to avoid the situation where if only 5 nodes were used, nodes from the next circle inward from the boundary would form part of the boundary, as a consequence of the alternating positioning of the nodes on adjacent circles. This would complicate the implementation of the boundary conditions, so additional nodes are added on the outer circle only. We use a time step of $dt = 10^{-5}$. Figure 4.13 shows the solution at $t = 0.01$, and figure 4.14 shows the solution at $t = 0.02$. We observe the shape of the solution becoming narrower and taller. After $t = 0.0219$ (figure 4.15), we rapidly approach blow up. The final solution before node tangling occurs is that of figure 4.16, which approximates a Dirac delta function with amplitude $u = 3.6 * 10^6$. The degree of node movement achieved is apparent by comparing figure 4.17 with 4.18. Figure 4.17 shows the initial node positions and figure 4.18 shows the positions at $t = 0.0219$ as we approach blow-up. We note that the nodes are indeed clustering around the area of interest, in this case the centre. Having moved the nodes allows a much better resolution of the shape of the blow up peak, compared to what is achievable if the nodes had stayed as in figure 4.17.

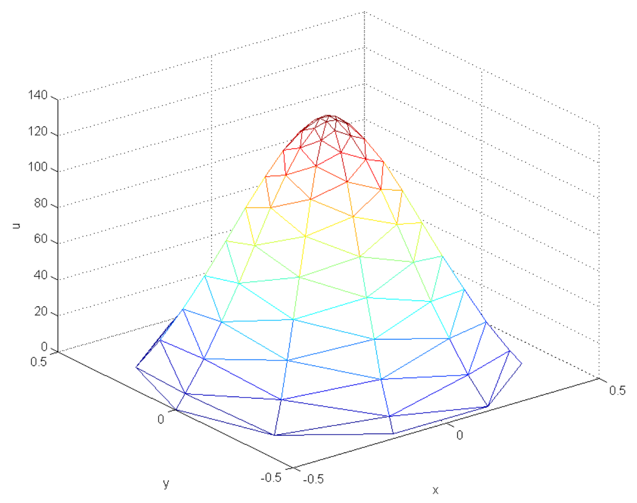


Fig. 4.13 Solution of the 2D Fisher's equation at $t = 0.01$. Note change of scale on the vertical axis.

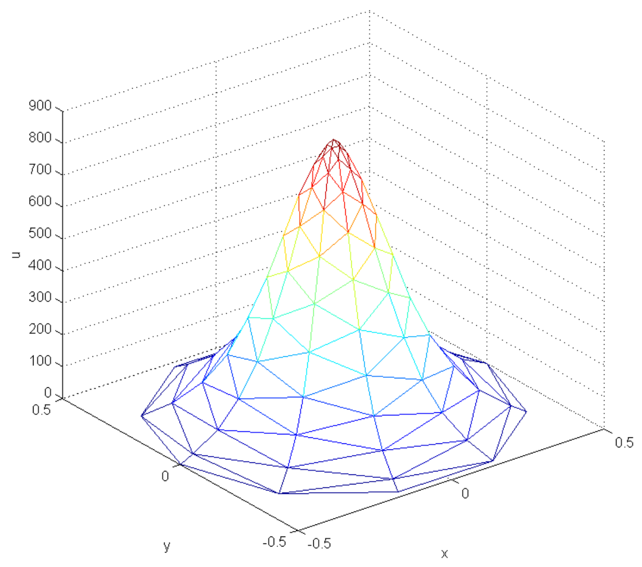


Fig. 4.14 Solution of the 2D Fisher's equation at $t = 0.02$. Note change of scale on the vertical axis.

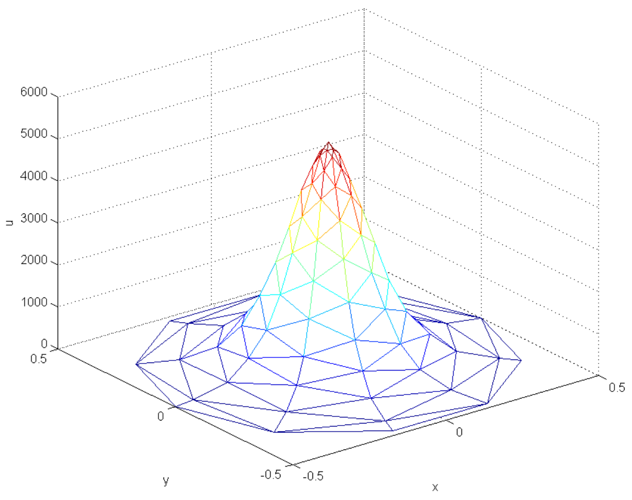


Fig. 4.15 Solution of the 2D Fisher's equation at $t = 0.0219$. Approaching blow-up. Note change of scale on the vertical axis.

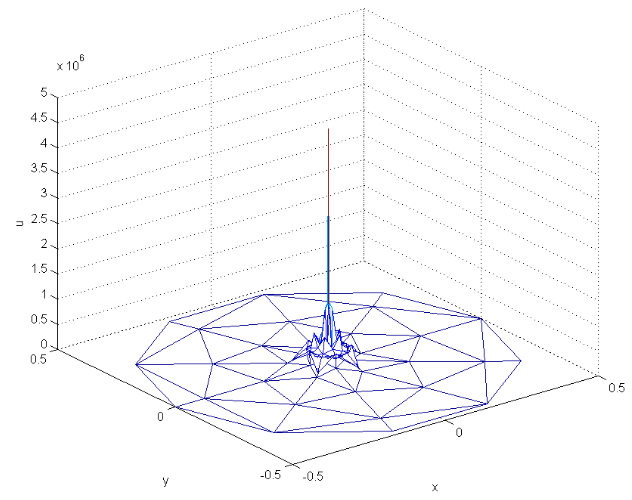


Fig. 4.16 Final solution of the 2D Fisher's equation. Here $t = 0.0225$. The solution approximates a Dirac delta function, and shortly after this time step the nodes become co-located and the model becomes unstable.

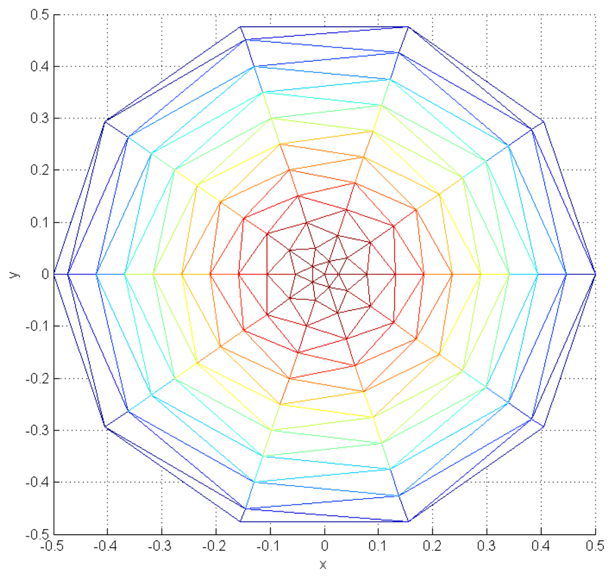


Fig. 4.17 Initial node positions for 2-D Fisher's equation at $t = 0$. We have 5 nodes on 20 equally spaced concentric circles.

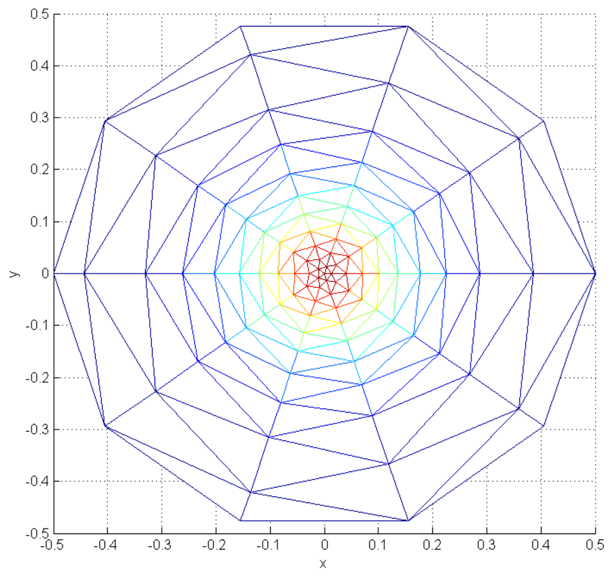


Fig. 4.18 Node positions for 2-D Fisher's equation at $t = 0.0219$ as we approach blow up. When compared to the initial grid, the movement towards the centre is clearly apparent.

4.3 Keller-Segel model in 2D

The Keller-Segel model [34] is a reaction-diffusion system related to the Fisher's equation. It differs from the Fisher's equation in that it involves both a substrate and a reactant, whereas the Fisher's equation is concerned with only the reactant. Both Cole [21] and Budd [13] consider the Keller-Segel system in two-dimensional, but radially symmetric, terms, on a moving mesh. Budd's paper [13] contains an equidistribution approach to moving the mesh, whereas Cole [21] demonstrates a conservation based method with a finite differences implementation. Here we move to a fully two dimensional approach, with a conservation based finite element method of solution (MMFEM).

This model, for chemotaxis of cells, takes the form of a pair of interdependent PDEs,

$$\frac{\partial u}{\partial t} = \nabla.(k_1(u,v)\nabla u - k_2(u,v)u\nabla v) + k_3(u,v) \tag{4.161}$$

$$\frac{\partial v}{\partial t} = D_v\nabla^2 v + k_4(u,v) - k_5(u,v)v \tag{4.162}$$

where
u=cell density
v=concentration of substrate
k_1=diffusivity
k_2=chemotactic sensitivity
k_3=cell growth and death
k_4=production of substrate
k_5=degradation of substrate.

We model a system on a fixed domain Ω with boundary S. We take the Neumann boundary conditions used in [13], given by

$$\nabla u \cdot \hat{\mathbf{n}}|_S = 0 \tag{4.163}$$

and

$$\nabla v \cdot \hat{\mathbf{n}}|_S = 0. \tag{4.164}$$

We also take the initial values for u and v from [13], given by

$$u(r,0) = 1000e^{(-500r^2)} \tag{4.165}$$

$$v(r,0) = 10e^{(-500r^2)} \tag{4.166}$$

where $r \in \Omega = \{r : \|r\| \leq R\}$, and $R = 1$. A free boundary is unimportant here, since the initial conditions give a wide margin where $u, v \approx 0$ between the central reacting zone and the boundary. In the event that we were to allow the boundary to move, this set of initial conditions would drive no movement in any case.

We consider a minimal model where the rate parameters k_i have linear form, and in particular the case where (4.161) and (4.162) are simplified to

$$\frac{\partial u}{\partial t} = \nabla^2 u - \chi\nabla.(u\nabla v) \tag{4.167}$$

$$\frac{\partial v}{\partial t} = \nabla^2 v + u - v \tag{4.168}$$

where χ is the chemotactic coefficient, for which a value χ=8 is suggested in [13]. In weak form, equations (4.167) and (4.168) become respectively

$$\int_\Omega w_i \frac{\partial u}{\partial t}\ d\Omega = \int_\Omega w_i \nabla^2 u\ d\Omega - \int_\Omega w_i \chi\nabla.(u\nabla v)\ d\Omega \tag{4.169}$$

$$\int_\Omega w_i \frac{\partial v}{\partial t}\ d\Omega = \int_\Omega w_i(\nabla^2 v + u - v)\ d\Omega \tag{4.170}$$

for a domain Ω.

We proceed similarly to the Fisher's model, although here matters are simplified by having a true conservation of total mass so that $\dot{\theta} = 0$. The Leibnitz integral gives that

$$\frac{d}{dt}\int_\Omega u\ d\Omega = \int_S \dot{\mathbf{x}}u.\hat{\mathbf{n}}\ dS + \int_\Omega \frac{\partial u}{\partial t}\ d\Omega. \tag{4.171}$$

Substitution from (4.167) gives

$$\frac{d}{dt}\int_\Omega u\ d\Omega = \int_S \dot{\mathbf{x}}u.\hat{\mathbf{n}}\ dS + \int_\Omega \left(\nabla^2 u - \chi\nabla.(u\nabla v)\right)\ d\Omega = 0 \tag{4.172}$$

which is equal to zero due to (i) the Neumann boundary conditions (4.163) and (4.164), and (ii) the fixed boundary so that $\mathbf{x}.\hat{\mathbf{n}} = 0$. Hence

$$\int_\Omega u\ d\Omega = c \tag{4.173}$$

and

$$\frac{d}{dt}\int_{\Omega} u \; d\Omega = 0 = \dot{\theta}, \tag{4.174}$$

i.e., mass is conserved. We define a distributed conservation principle using the weight functions w_i.

$$\int_{\Omega} w_i u \; d\Omega = c_i \tag{4.175}$$

or

$$\frac{d}{dt}\int_{\Omega} w_i u \; d\Omega = 0. \tag{4.176}$$

We differentiate using Leibnitz' rule and obtain

$$\int_{\Omega} \frac{\partial}{\partial t}(w_i u) \; d\Omega - \int_{S} w_i u \dot{\mathbf{x}}.\hat{\mathbf{n}} \; dS = 0 \tag{4.177}$$

or

$$\int_{\Omega} \left[w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + w_i \nabla.(u\dot{\mathbf{x}}) + u\dot{\mathbf{x}}.\nabla w_i \right] \; d\Omega = 0. \tag{4.178}$$

If w_i moves with velocity $\dot{\mathbf{x}}$, then by analogy with a convecting system

$$\frac{\partial w_i}{\partial t} + \dot{\mathbf{x}}.\nabla w_i = 0 \tag{4.179}$$

then

$$\int_{\Omega} w_i \nabla.(\dot{\mathbf{x}}u) \;\; d\Omega = -\int_{\Omega} w_i \frac{\partial u}{\partial t} \;\; d\Omega. \tag{4.180}$$

By Green's theorem on the left hand side, and substituting from the weak form (4.169),

$$-\int_{S} w_i \dot{\mathbf{x}} u \cdot \hat{\mathbf{n}} \; dS + \int_{\Omega} \nabla w_i.\dot{\mathbf{x}} u \;\; d\Omega = \int_{\Omega} w_i (\nabla^2 u - \chi \nabla.(u\nabla v)) \; d\Omega. \tag{4.181}$$

The first term on the left hand side is equal to zero, as $\dot{\mathbf{x}} \cdot \hat{\mathbf{n}}$ is zero on the boundary. Expanding the right hand side using integration by parts, we obtain

$$\int_{\Omega} \nabla w_i.\dot{\mathbf{x}} u \;\; d\Omega = \int_{S} w_i \nabla u.\hat{\mathbf{n}} \; dS - \int_{\Omega} \nabla w_i.\nabla u \;\; d\Omega - \int_{\Omega} w_i \chi \nabla.(u\nabla v) \; d\Omega \tag{4.182}$$

and after integrating by parts again, we arrive at

$$\int_{\Omega} \nabla w_i.\dot{\mathbf{x}} u \;\; d\Omega = \int_{S} w_i \nabla u.\hat{\mathbf{n}} \; dS - \int_{\Omega} \nabla w_i.\nabla u \;\; d\Omega - \int_{S} w_i \chi u \nabla v.\hat{\mathbf{n}} \; dS + \int_{\Omega} \chi \nabla w_i.u\nabla v \; d\Omega. \tag{4.183}$$

The boundary terms are zero due to the zero flux Neumann conditions (4.163) and (4.164).

We arrive at

$$\int_\Omega \nabla w_i.\dot{\mathbf{x}}u \;\; d\Omega = -\int_\Omega \nabla w_i.\nabla u \;\; d\Omega + \int_\Omega \chi \nabla w_i.u\nabla v \; d\Omega. \tag{4.184}$$

This is our weak form for $\dot{\mathbf{x}}$ in terms of u and v. We will move the nodes using a time integration scheme, and recover u using a conservation approach. We do however, require a weak form for $\dot{v}$. We calculate $\dot{v}$ from the definition of $\frac{\partial v}{\partial t}$, (4.168), the known nodal velocity $\dot{\mathbf{x}}$ and the material derivative

$$\frac{dv}{dt} = \frac{\partial v}{\partial t} + \nabla v \cdot \dot{\mathbf{x}}. \tag{4.185}$$

The $\dot{\mathbf{x}}$ value is now known. Noting that the weak form (4.170) of the definition of $\frac{\partial v}{\partial t}$ is

$$\int_\Omega w_i \frac{\partial v}{\partial t} \; d\Omega = \int_\Omega \left[w_i \nabla^2 v + w_i u - w_i v \right] \; d\Omega \tag{4.186}$$

we apply Green's theorem to obtain

$$\int_\Omega w_i \frac{\partial v}{\partial t} \; d\Omega = \int_S w_i \nabla v.\hat{\mathbf{n}} dS - \int_\Omega \nabla w_i.\nabla v \; d\Omega + \int_\Omega w_i(u-v) \; d\Omega. \tag{4.187}$$

Since we have zero Neumann conditions (4.164) the boundary term is equal to zero. We have

$$\int_\Omega w_i \frac{\partial v}{\partial t} \; d\Omega = -\int_\Omega \nabla w_i.\nabla v \; d\Omega + \int_\Omega w_i(u-v) \; d\Omega \tag{4.188}$$

which is the weak form we require in order to obtain v at the new time step through (4.185).

Finite elements

Having constructed the necessary weak form, we now make the finite element substitutions. We have no Dirichlet conditions to impose, hence we can use unmodified basis functions as in case 2. To solve (4.184) using the finite element method, we introduce the velocity potential ϕ defined by

$$\dot{\mathbf{x}} = \nabla \phi. \tag{4.189}$$

Equation (4.184) becomes

$$\int_\Omega \nabla w_i.\nabla \phi u \;\; d\Omega = -\int_\Omega \nabla w_i.\nabla u \;\; d\Omega + \int_\Omega \chi \nabla w_i.u\nabla v \; d\Omega. \tag{4.190}$$

We use the basis functions $w_i = \{W_i(x,y)\}$ and the piecewise linear approximations

$$\Phi(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)\Phi_j(t) \tag{4.191}$$

$$U(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)U_j(t) \tag{4.192}$$

$$V(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t)V_j(t). \tag{4.193}$$

We can now write equation (4.184) in a finite element form.

$$\sum_{j=1}^{N}\left[\int_{\Omega} U\nabla W_i.\nabla W_j \; d\Omega\right]\Phi_j = -\sum_{j=1}^{N}\left[\int_{\Omega} \nabla W_i.\nabla W_j \; d\Omega\right]U_j + \chi\sum_{j=1}^{N}\left[\int_{\Omega}(U\nabla W_i.\nabla W_j)d\Omega\right]V_j. \tag{4.194}$$

In matrix form this is

$$K(\underline{U})\underline{\Phi} = \underline{f} \tag{4.195}$$

with $K(\underline{U})$ the weighted stiffness matrix of chapter 3, section 3.1.3. The vector $\underline{\Phi}$ contains the Φ_j values and the vector $\underline{f}$ is the vector containing the f_i values given by

$$f_i = -\sum_{j=1}^{N}\left[\int_{\Omega} \nabla W_i.\nabla W_j \; d\Omega\right]U_j + \sum_{j=1}^{N}\left[\int_{\Omega}(\chi U\nabla W_i.\nabla W_j)d\Omega\right]V_j \tag{4.196}$$

or

$$\underline{f} = -K\underline{U} + \chi K(\underline{U})\underline{V} \tag{4.197}$$

with $\underline{V}$ the vector containing the V_j values. We solve (4.195) for Φ, defining $\Phi = 0$ at one node to overcome the infinity of solutions that would otherwise be obtainable from the singular matrix $K(\underline{U})$. The velocities can then be recovered exactly as for the Fisher's equation, using (4.131).

Time step

We generate the co-ordinate system at the next time-step from (3.18) using the forward Euler approximation.

Recovering U

We recover U using our defined mass conservation property, (4.175). The process is similar to that outlined in chapter 3, section 3.1.3, but is simplified by having true conservation of mass.

Equation (4.175), using the approximation $U \approx u$ and having selected the set of basis functions $w_i = W_i$ is, for all i

$$\int_{\Omega} W_i U \; d\Omega = c_i \tag{4.198}$$

where the c_i are the constant local masses associated with the corresponding W_i. Using the piecewise linear form of U (4.192) we obtain

$$\sum_{j=1}^{N} \left[\int_{\Omega} W_i W_j d\mathbf{x} \right] U_j = c_i \tag{4.199}$$

which is equivalent to the mass matrix system

$$M\underline{U} = \underline{c} \tag{4.200}$$

where $\underline{U}$ is the vector containing the U_j values and $\underline{c}$ is the vector containing the constant c_i values. This equation is used to calculate the initial (and constant) values of c_i, using the initial values of U_j. After repositioning the nodes using a time integral of $\mathbf{x}_j$, we update the mass matrix and then may recover $U_j(t)$ from the mass matrix system (4.200).

The calculation of $\frac{\partial v}{\partial t}$

Taking the weak form (4.188), we use the piecewise linear basis functions $w_i = W_i$ and the piecewise linear approximations

$$V(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) V_j(t) \tag{4.201}$$

and

$$\frac{\partial V}{\partial t} = Q(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) Q_j(t) \tag{4.202}$$

together with the piecewise linear approximations for U (4.192) and V (4.193), we obtain from (4.188),

$$\sum_{j=1}^{N}\left[\int_{\Omega_i} W_i W_j \; d\Omega\right] Q_j = -\sum_{j=1}^{N}\left[\int_{\Omega} \nabla W_i . \nabla W_j \; d\Omega\right] \; V_j + \sum_{j=1}^{N}\left[\int_{\Omega} W_i W_j \; d\Omega\right] \; (U_j - V_j) \tag{4.203}$$

or in matrix form

$$M\underline{Q} = M(\underline{U} - \underline{V}) - K\underline{V} \tag{4.204}$$

in terms of the mass and stiffness matrices. A vector $\underline{\dot{V}}$ is thus obtained from known U and V and then inserted into equation (4.185) to obtain $\frac{dV}{dt}$. The forward Euler method is then used to approximate V at the next time step.

Algorithm 10

The finite element solution of the Keller-Segel equations (4.167) and (4.168) on the moving mesh in 2-D therefore consists of the following steps. Having obtained the values of c_i from (4.200):

1. Find the velocity potential by solving equation (4.195) for the $\Phi_j(t)$ values;
2. Find the node velocity by solving equation (4.131) for the $\dot{\mathbf{X}}_j(t)$ values;
3. Find Q by solving equation (4.204);
4. Generate the co-ordinate system at the next time-step $t + dt$ from (3.18) using the forward Euler approximation;
5. Find $V(t + dt)$ at the next time step by integrating the material derivative(4.185) using the forward Euler approximation;
6. Find the solution $U(t + dt)$ by solving the conservation equation (4.200).

Results

We implement algorithm 10 using MATLAB. We set up an initial grid consisting of m nodes on n concentric circles. We find that the model is stable and robust under a wide range of choices of m, n and time step Δt. For the reference blow up time, we refer to [13], where a very high resolution model determined a blow up time of $T \approx 5.15 \times 10^{-5}$. For every set of parameters for which we run our model, the limiting factor for the resolution of the blow-up peak is node tangling. We find that the performance is affected not just by the independent choices of m and n, but by the shapes of the triangles that m and n produce in combination. However, for the better performing combinations of m and n, the time at blow-up is much more accurate than was achieved by the radially symmetric conservation method finite difference models of Cole [21]. For the poorer performing combinations of m and n, the radially symmetric finite difference model is the better performer. In mitigation of this it should be noted that the fully 2D model is much more flexible than the radial model in terms of the range of initial conditions it can simulate. We find that the model result does not exhibit a close sensitivity to the size of Δt. We are able to use a larger Δt than [21], which is clearly advantageous for the speed of computation. We can then reduce Δt as we approach blow up, purely to better resolve the time of blow up. We find that the nodes move towards the centre as expected. As was observed in [21] and [13], the concentration of the substrate v does not vary much when the initial conditions are as stated. The concentration of the reactant forms an approximate Dirac delta function at blow up. We find that the blow up time is, however, sensitive to the number of concentric circles of nodes n. With reduced n, we avoid node tangling for longer and so a higher peak in u can be resolved at a later time. This same trend is observed in the radial model of [21]. We also note that the blow up time is sensitive to the number of nodes m on each concentric circle. As m increases, node tangling occurs sooner. It is likely that the long, thin shape of the central triangles in these high m models is causing an ill-conditioned stiffness matrix to be produced in the computation. We conclude that it is necessary to design the initial grid with careful consideration given to avoiding long, thin triangles if possible. Tables 4.5, 4.6 and 4.7 give blow up times for a variety of m, n and Δt, chosen so that direct comparisons with [21] can be made.

Table 4.5 Blow up time for 2D Keller-Segel model with $m = 10$ showing variation by Δt, and n

m	n	Δt	$T_{blow-up}$
10	5	4x10^{-7}	2.00x10^{-5}
10	5	2x10^{-7}	1.90x10^{-5}
10	5	1x10^{-7}	1.90x10^{-5}
10	5	5x10^{-8}	2.00x10^{-5}
10	10	4x10^{-7}	1.48x10^{-5}
10	10	2x10^{-7}	1.24x10^{-5}
10	10	1x10^{-7}	1.10x10^{-5}
10	10	5x10^{-8}	1.00x10^{-5}

Table 4.6 Blow up time for 2D Keller-Segel model with $m = 20$ showing variation by Δt, and n

m	n	Δt	$T_{blow-up}$
20	5	4x10^{-7}	5.52x10^{-5}
20	5	2x10^{-7}	5.54x10^{-5}
20	5	1x10^{-7}	5.54x10^{-5}
20	5	5x10^{-8}	5.55x10^{-5}
20	10	4x10^{-7}	1.48x10^{-5}
20	10	2x10^{-7}	1.52x10^{-5}
20	10	1x10^{-7}	1.52x10^{-5}
20	10	5x10^{-8}	1.52x10^{-5}
20	20	4x10^{-7}	8.80x10^{-6}
20	20	2x10^{-7}	9.00x10^{-6}
20	20	1x10^{-7}	9.10x10^{-6}
20	20	5x10^{-8}	9.10x10^{-6}

Table 4.7 Blow up time for 2D Keller-Segel model with $m = 40$ showing variation by Δt, and n

m	n	Δt	$T_{blow-up}$
40	5	4x10^{-7}	5.52x10^{-5}
40	5	2x10^{-7}	5.54x10^{-5}
40	5	1x10^{-7}	5.56x10^{-5}
40	5	5x10^{-8}	5.56x10^{-5}
40	10	4x10^{-7}	1.52x10^{-5}
40	10	2x10^{-7}	1.52x10^{-5}
40	10	1x10^{-7}	1.52x10^{-5}
40	10	5x10^{-8}	1.52x10^{-5}
40	20	4x10^{-7}	8.80x10^{-6}
40	20	2x10^{-7}	8.80x10^{-6}
40	20	1x10^{-7}	8.80x10^{-6}
40	20	5x10^{-8}	8.85x10^{-6}

Examples of the graphical results obtained are given in figures 4.19, 4.20 and 4.21. We observe the increasing height of the peak in u and the much lesser reduction in the height of v. Figure 4.22 shows the node positions at $t = 0$ (dotted line) and at blow up (solid line). We observe adaptation in the centre of the domain; figure 4.23 shows a closer view. Interestingly, because the outer elements of the domain hold no mass (*i.e.* $u = 0$), they are unable to adapt their configuration at all in this mass conservation driven approach. We have node movement only where we have mass movement, and the outer elements are unable to 'donate' any of their resolution to the centre. This is a drawback to the mass conservation method and it should be noted that for domains which involve large regions with zero mass or constant mass, node movement cannot occur in those regions.

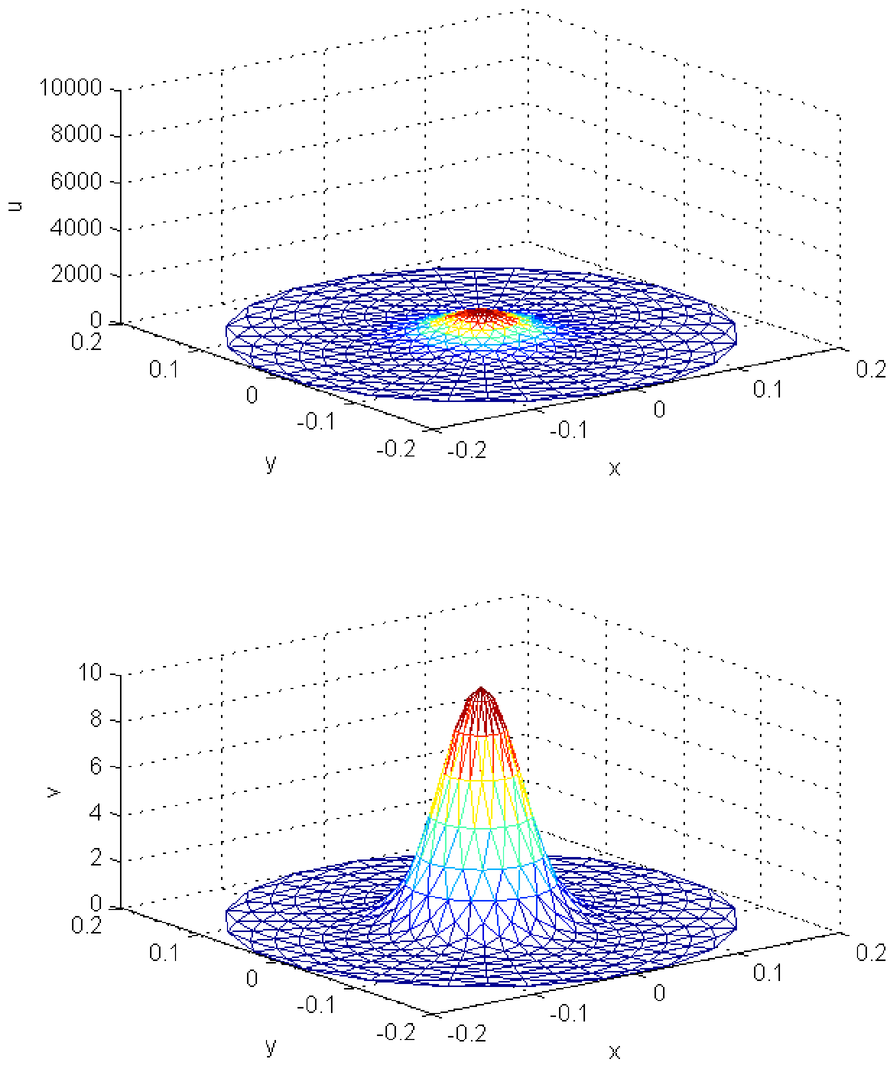
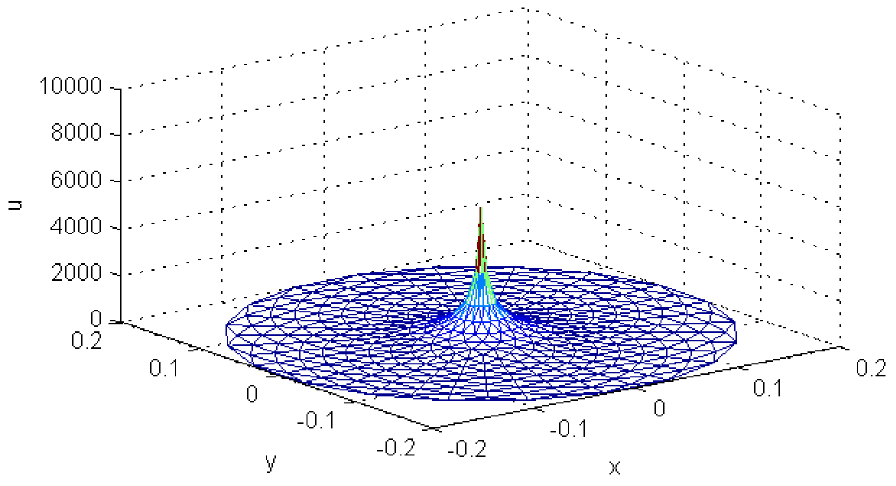


Fig. 4.19 Initial conditions for the Keller Segel model.



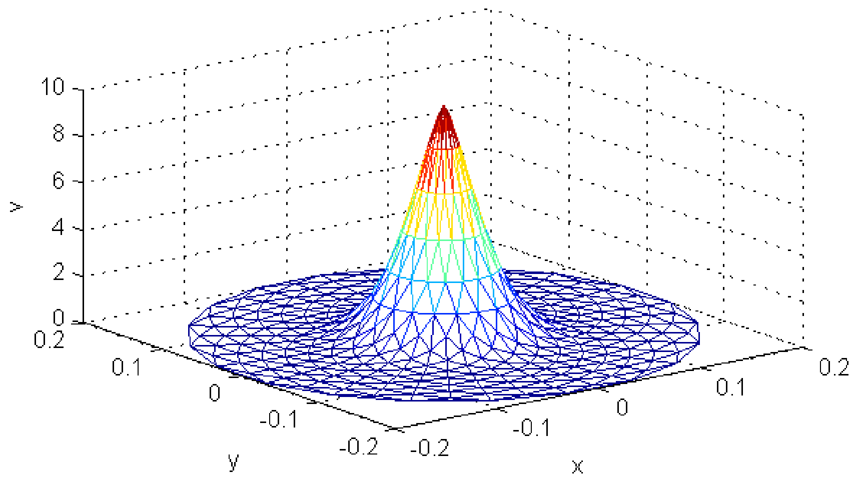
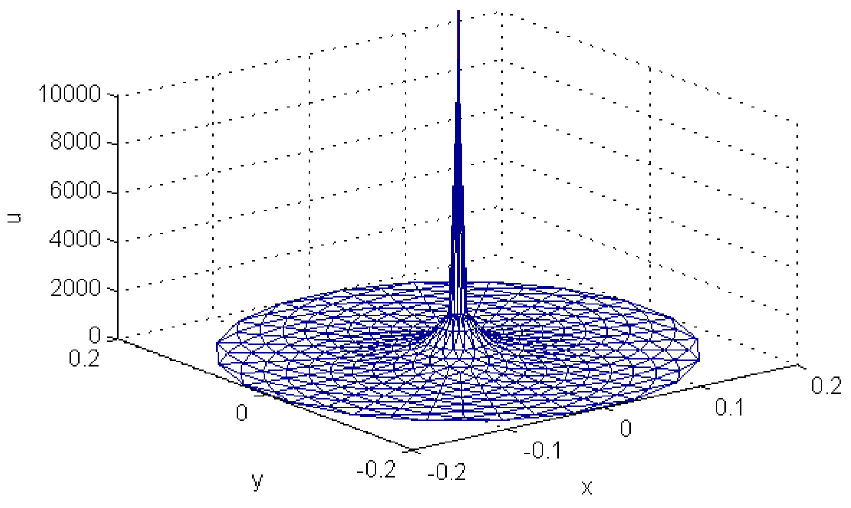


Fig. 4.20 Solution of the Keller Segel model on a grid with 20 nodes on 20 concentric circles at $t = 5x10^{-6}$.



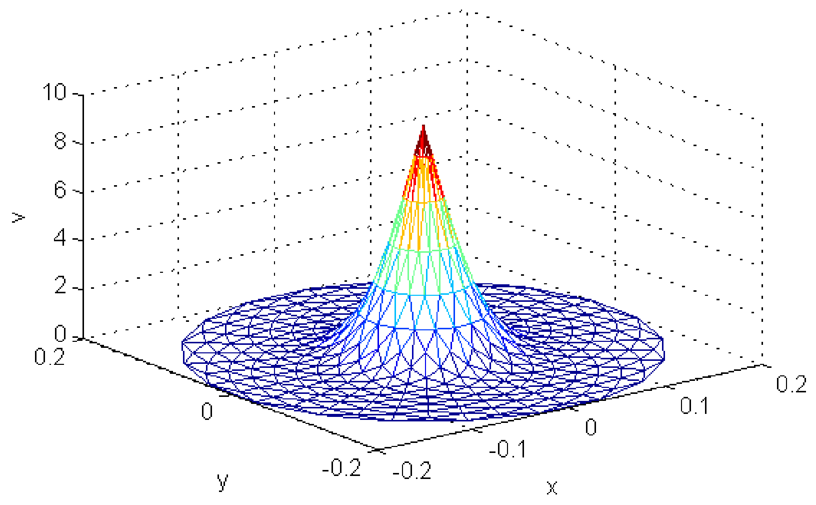


Fig. 4.21 Solution of the Keller Segel model on a grid with 20 nodes on 20 concentric circles as we approach blow-up.

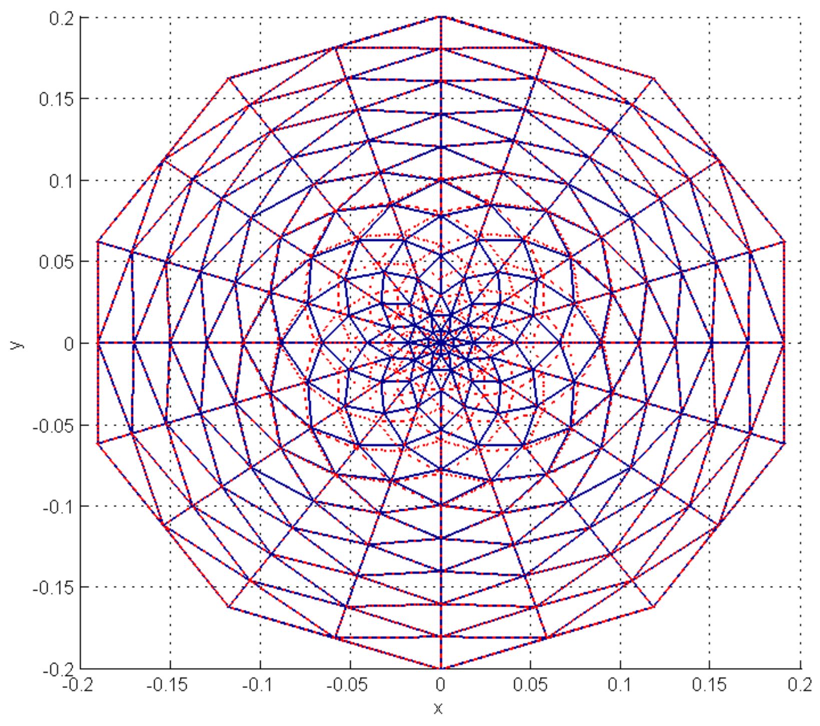


Fig. 4.22 Comparison of mesh movement between initial distribution (red dotted line) and approaching blow-up (blue solid line).

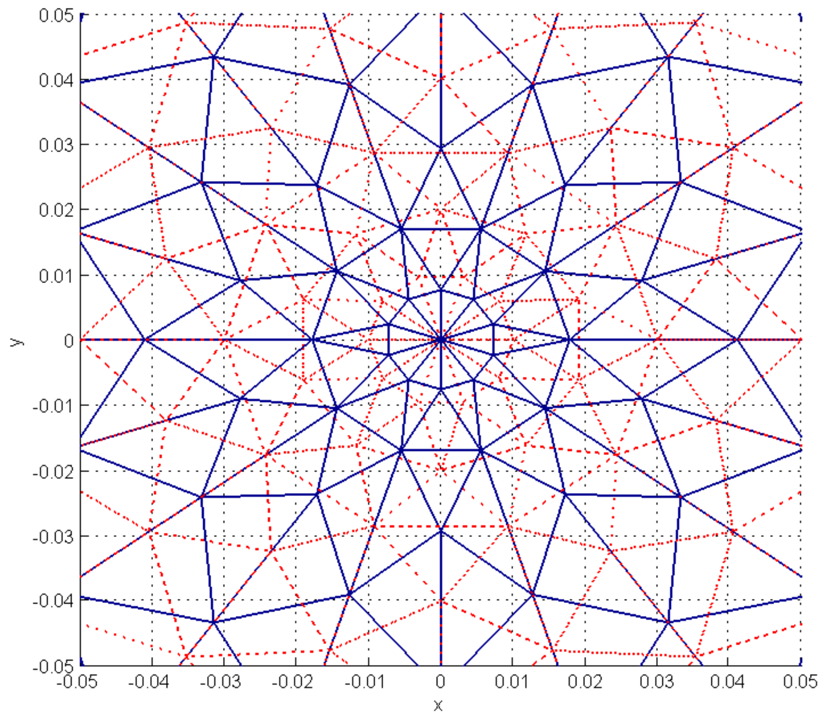


Fig. 4.23 Comparison of mesh movement between initial distribution (red dotted line) and approaching blow-up (blue solid line), a closer view.

Chapter 5

Moving interface models

5.1 The two phase Stefan problem in 1D

We now consider models with a moving interface between two phases. These models are a natural development from the free boundary variants of Chapter 4, for example the Fisher's model of section 4.2 with case (2) boundary conditions. We begin with a model of the two phase Stefan problem, constructed in a similar manner to that described in the Baines, Hubbard, Jimack and Mahmood (2009) paper [8]. The model describes the melting of ice into water. This model differs from those seen in this thesis so far in that the nodes at the phase boundary are themselves moving, as well as node movement within each phase. The model explicitly calculates the velocity of the interface between phases as the ice melts. This velocity comes from an interface condition, and this information is then incorporated into the model as a Dirichlet condition at the moving boundary. The model is constructed as a moving mesh finite element model. We present a modification to the paper [8]. In this problem we have Dirichlet boundary conditions on the boundary velocities as well as on the temperature of the ice or water. This makes it possible to construct the entire finite element model from start to finish in terms of the modified basis functions described in Chapter 4, section 4.2.1. We therefore do not need to switch basis systems via the ALE equation, as we did for the free boundary Fisher's problem (4.2) and as is derived in the paper [8]. We derive this alternative process and demonstrate that results equivalent to [8] can be obtained by it. The system is driven by the diffusion of heat. We consider the 1-D diffusion PDEs

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

$$K_L \frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(k_L \frac{\partial u}{\partial x}\right). \tag{5.1}$$

The parameters used are K_S and K_L, the volumetric heat capacities of the solid and liquid phases; k_S and k_L, the thermal conductivities; and u, the temperature.

At the interface, $u = u_m$, the temperature at which melting takes place. There is an energy balance across the phase-change boundary $\Gamma_m(t)$. This is described by the Stefan equation

$$k_S \frac{\partial u_S}{\partial x} - k_L \frac{\partial u_L}{\partial x} = \lambda \dot{x}_m \tag{5.2}$$

with λ, the heat of phase change per unit volume; and $\dot{x}_m$, the velocity of the interface. We assume that all parameters are constant within their respective phases. In this system the derivative $\frac{\partial u}{\partial x}$ is not continuous across the moving interface so we will need to be explicit about in which phase we are evaluating that gradient.

The particular case we will consider uses fixed outer boundaries $x \in [0,1]$ with zero Dirichlet conditions on the velocity for external boundary nodes, and initial conditions taken from a system with an exact solution,

$$u_S = u^*\left(1 - \frac{erf(x/(2\sqrt{\kappa_S\ t}))}{erf\psi}\right)$$

$$u_L = u_0\left(1 - \frac{erfc(x/(2\sqrt{\kappa_L\ t}))}{erfc(\psi\sqrt{\kappa_S/\kappa_L})}\right). \tag{5.3}$$

The following values for the parameters are used:

$$\begin{aligned}
&u^* = -20\\
&u_0 = 10\\
&k_S = 2.22\\
&k_L = 0.556\\
&K_S = 1.762\\
&K_L = 4.226\\
&\lambda = 338\\
&\psi = 0.2054\\
&t_{\text{initial}} = 0.0012 \text{ (in order to avoid a singularity in (5.3))}\\
&\kappa = k/K.
\end{aligned}$$

We consider a domain $R(t)$ with fixed external boundaries $\partial R = [0,1]$ and an interface m. The boundary conditions are then formally

$$u_S(0) = -20 \tag{5.4}$$

$$u_L(1) = 10 \tag{5.5}$$

$$u_m(m) = 0 \tag{5.6}$$

for the temperatures, and

$$\dot{x}(0) = 0 \tag{5.7}$$

$$\dot{x}(1) = 0 \tag{5.8}$$

for the velocities. We also have $\dot{x}_m$, the velocity at the interface, given by the interface condition (5.2). We write the diffusion PDEs (5.1) in weak form

$$\int_R w_i K_S \frac{\partial u}{\partial t}\, dx = \int_R w_i \frac{\partial}{\partial x}\left(k_S \frac{\partial u}{\partial x}\right)\, dx \tag{5.9}$$

$$\int_R w_i K_L \frac{\partial u}{\partial t}\, dx = \int_R w_i \frac{\partial}{\partial x}\left(k_L \frac{\partial u}{\partial x}\right)\, dx. \tag{5.10}$$

We begin by defining θ. Although in physical terms this is the integral over temperature u, it may be helpful to think of this as 'mass' and we will use that shorthand here.

$$\theta(t) = \int_{R(t)} u \; dx. \tag{5.11}$$

We may then write a relative conservation principle in terms of θ,

$$\frac{1}{\theta(t)} \int_{R(t)} u \; dx = 1. \tag{5.12}$$

We may consistently introduce a weighted form

$$\frac{1}{\theta(t)} \int_{R(t)} w_i u \; dx = c_i \tag{5.13}$$

where w_i is the weight, equivalent to

$$\int_{R(t)} w_i u \; dx = c_i \theta(t) = c_i \int_{R(t)} u \; dx. \tag{5.14}$$

Here the constant c_i is determined by the choice of w_i. If we choose a set of w_i that together form a partition of unity, then the set of c_i will likewise form a partition of unity. Equation (5.14) is now a principle governing the distributed conservation of mass. We differentiate (5.14) with respect to time using the Leibnitz integral rule. We use a moving reference frame, so the velocity of the node movements, $\dot{x}(t,x)$, must be considered in our calculations,

$$\frac{d}{dt}\left[\int_{R(t)} w_i u \; dx\right] = \int_{R(t)} \left(\frac{\partial (w_i u)}{\partial t} + \frac{\partial}{\partial x}(w_i u \dot{x})\right) \; dx. \tag{5.15}$$

This is the Reynolds Transport Theorem described in Chapter 3, section 3.1.1. Assuming that the basis functions w_i move with the domain we know that they have velocity $\dot{x}$ and therefore

$$\frac{\partial w_i}{\partial t} + \dot{x}\frac{\partial w_i}{\partial x} = 0 \tag{5.16}$$

hence

$$\frac{d}{dt}\left[\int_{R(t)} w_i u \; dx\right] = \int_{R(t)} w_i \left(\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(u\dot{x})\right) \; dx \tag{5.17}$$

or

$$\frac{d}{dt}\left[\int_{R(t)} w_i u \; dx\right] - \int_{R(t)} w_i \frac{\partial}{\partial x}(u\dot{x}) \; dx = \int_{R(t)} w_i \frac{\partial u}{\partial t} \; dx. \tag{5.18}$$

In terms of $\dot{\theta}$ and the constants c_i this is

$$c_i\dot{\theta} - \int_{R(t)} w_i \frac{\partial}{\partial x}(u\dot{x})\ dx = \int_{R(t)} w_i \frac{\partial u}{\partial t}\ dx. \tag{5.19}$$

For consistency of method with the 2-D version, we introduce the velocity potential ϕ defined by

$$\dot{x} = \frac{\partial \phi}{\partial x} \tag{5.20}$$

so that

$$c_i\dot{\theta} - \int_{R(t)} w_i \frac{\partial}{\partial x}\left(u\frac{\partial \phi}{\partial x}\right)\ dx = \int_{R(t)} w_i \frac{\partial u}{\partial t}\ dx \tag{5.21}$$

or, after integration by parts

$$c_i\dot{\theta} + \int_{R(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R(t)} = \int_{R(t)} w_i \frac{\partial u}{\partial t}\ dx. \tag{5.22}$$

We substitute in a weak form of the driving PDE, either (5.9) or (5.10), depending on the phase under consideration. For either phase $p \in [S,L]$

$$c_i\dot{\theta} + \int_{R(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R(t)} = \int_{R(t)} \frac{w_i}{K_p}\left(\frac{\partial}{\partial x}\left(k_p\frac{\partial u}{\partial x}\right)\right)\ dx. \tag{5.23}$$

Again integrating by parts, this time on the right hand side

$$c_i\dot{\theta} + \int_{R(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R(t)} = -\int_{R(t)} \kappa_p\frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\ dx + \left[w_i\kappa_p\frac{\partial u}{\partial x}\right]_{\partial R(t)} \tag{5.24}$$

where $\kappa_p = k_p/K_p$. From this point onwards we consider the two phases separately, on their domains R_S with boundary $\partial R_S = [0,m]$, and R_L with boundary $\partial R_L = [m,1]$.
The 'masses' of equation (5.11) become, for the solid phase and liquid phase respectively,

$$\theta_S(t) = \int_{R_S(t)} u\ dx \tag{5.25}$$

$$\theta_L(t) = \int_{R_L(t)} u\ dx \tag{5.26}$$

with distributed (weak) forms

$$c_{S_i}\theta_S(t) = \int_{R_S(t)} w_i u \; dx \tag{5.27}$$

$$c_{L_i}\theta_L(t) = \int_{R_L(t)} w_i u \; dx. \tag{5.28}$$

We rewrite (5.24) for each phase separately. Since $\dot{x} = 0$ at the external boundaries and $u = u_m$ on the interface boundary, the velocity potential for the solid phase is given by

$$c_{S_i}\dot{\theta}_S + \int_{R_S(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} \; dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R_S(t)} = -\int_{R_S(t)} \kappa_S\frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx + \left[w_i\kappa_S\frac{\partial u}{\partial x}\right]_{\partial R_S(t)} \tag{5.29}$$

so that

$$\begin{aligned} c_{S_i}\dot{\theta}_S + \int_{R_S(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} \; dx - u_m w_i\frac{\partial \phi}{\partial x}\bigg|_{R_m(t)} \\ = -\int_{R_S(t)} \kappa_S\frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx + w_i\kappa_S\frac{\partial u}{\partial x}\bigg|_{R_m(t)} - w_i\kappa_S\frac{\partial u}{\partial x}\bigg|_{R_f} \end{aligned} \tag{5.30}$$

and the velocity potential for the liquid phase is given by

$$c_{L_i}\dot{\theta}_L + \int_{R_L(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} \; dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R_L(t)} = -\int_{R_L(t)} \kappa_L\frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx + \left[w_i\kappa_L\frac{\partial u}{\partial x}\right]_{\partial R_L(t)} \tag{5.31}$$

so that

$$\begin{aligned} c_{L_i}\dot{\theta}_L + \int_{R_L(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} \; dx + u_m w_i\frac{\partial \phi}{\partial x}\bigg|_{R_m(t)} = \\ -\int_{R_L(t)} \kappa_L\frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx + w_i\kappa_L\frac{\partial u}{\partial x}\bigg|_{R_f} - w_i\kappa_L\frac{\partial u}{\partial x}\bigg|_{R_m(t)} \end{aligned} \tag{5.32}$$

where the subscripts f and m denote the fixed and moving boundaries respectively. The Stefan condition in distributed form is

$$k_S w_i\frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L w_i\frac{\partial u_L}{\partial x}\bigg|_{R_m(t)} = \lambda \; w_i\frac{\partial \phi}{\partial x}\bigg|_{R_m(t)}. \tag{5.33}$$

At the moving interface, the Stefan condition can replace the terms in equations involving ϕ. We may run into difficulties with this if $u_m = 0$ or changes sign at any point, so as in [8] we will add a constant to the whole domain when constructing the algorithm. Equations (5.30) and (5.32) become

$$c_{S_i}\dot{\theta}_S + \int_{R_S(t)} u \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx = -\int_{R_S(t)} \kappa_S \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\, dx + w_i \kappa_S \frac{\partial u}{\partial x}\bigg|_{R_m(t)} - w_i \kappa_S \frac{\partial u}{\partial x}\bigg|_{R_f} + \frac{u_m}{\lambda}\left(k_S w_i \frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L w_i \frac{\partial u_L}{\partial x}\bigg|_{R_m(t)} \right) \tag{5.34}$$

$$c_{L_i}\dot{\theta}_L + \int_{R_m(t)} u \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx = -\int_{R_L(t)} \kappa_L \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\, dx + w_i \kappa_L \frac{\partial u}{\partial x}\bigg|_{R_f} - w_i \kappa_L \frac{\partial u}{\partial x}\bigg|_{R_m(t)} - \frac{u_m}{\lambda}\left(k_S w_i \frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L w_i \frac{\partial u_L}{\partial x}\bigg|_{R_m(t)} \right). \tag{5.35}$$

We can then sum equations (5.34) and (5.35) over $R_S(t)$ and $R_L(t)$ respectively, to give us the rate of change of total 'mass', θ_S and θ_L, in each phase. Providing that we have chosen a set of basis functions w_i that form a partition of unity, the full integral terms will sum to zero and the values of c_i will sum to 1. We obtain

$$\dot{\theta}_S = \kappa_S \frac{\partial u}{\partial x}\bigg|_{R_m(t)} - \kappa_S \frac{\partial u}{\partial x}\bigg|_{R_f} + \frac{u_m}{\lambda}\left(k_S \frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L \frac{\partial u_L}{\partial x}\bigg|_{R_m(t)} \right) \tag{5.36}$$

$$\dot{\theta}_L = \kappa_L \frac{\partial u}{\partial x}\bigg|_{R_f} - \kappa_L \frac{\partial u}{\partial x}\bigg|_{R_m(t)} - \frac{u_m}{\lambda}\left(k_S \frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L \frac{\partial u_L}{\partial x}\bigg|_{R_m(t)} \right). \tag{5.37}$$

To recover the nodal velocities we first solve the four equations (5.34), (5.35), (5.36) and (5.37) to give $\dot{\theta}$ and ϕ in each phase. The Dirichlet boundary conditions (5.7) and (5.8) on the velocities correspond to zero Neumann boundary conditions on ϕ. We then return to our definition of ϕ (5.20), now written in weak form,

$$\int_{R(t)} w_i \dot{x}\, dx = \int_{R(t)} w_i \frac{\partial \phi}{\partial x}\, dx \tag{5.38}$$

with weight function w_i. The system of equations (5.38) can be solved for $\dot{x}$. For the interface itself, it is more accurate to replace this derived system with the Stefan condition directly, *i.e.*, for the interface, the velocity $\dot{x}_m$ is given by

$$\lambda\ w_i \dot{x}_m |_{R_m(t)} = k_S w_i \frac{\partial u_S}{\partial x}\bigg|_{R_m(t)} - k_L w_i \frac{\partial u_L}{\partial x}\bigg|_{R_m(t)}. \tag{5.39}$$

The boundary conditions (5.7) and (5.8) that give $\dot{x} = 0$ on the external boundary, together with the Stefan condition (5.39) are strongly imposed on (5.38). In doing so we note that there is a compatibility condition on the sum over equation (5.38), given by

$$\int_{R(t)} \dot{x}\ dx = \int_{R(t)} \frac{\partial \phi}{\partial x}\ dx = [\phi]_{R(t)}. \tag{5.40}$$

Having obtained $\dot{x}$, we move the interior points of the domain. We also update θ_S and θ_L from $\dot{\theta}_S$ (5.36) and $\dot{\theta}_L$ (5.37).

We can then recover u. Equations (5.27) and (5.28) allow us to determine the (constant) mass fractions c_{S_i} and c_{L_i} using the initial conditions. We can rewrite (5.27) and (5.28) for $t = 0$ to give

$$c_{S_i} = \frac{1}{\theta_S(0)} \int_{R_S(0)} w_i(x,0) u(x,0)\ dx \tag{5.41}$$

$$c_{L_i} = \frac{1}{\theta_L(0)} \int_{R_L(0)} w_i(x,0) u(x,0)\ dx. \tag{5.42}$$

We may then use the same equations to recover u, given θ_S and θ_L at the new time step and with the weight functions having moved with the domain. For the solid phase, u can be recovered from (5.14) in the form

$$\int_{R_S(t)} w_i(x,t) u(x,t)\ dx = c_{S_i} \theta_S(t) \tag{5.43}$$

and for the liquid phase, u can be recovered from

$$\int_{R_L(t)} w_i(x,t) u(x,t)\ dx = c_{L_i} \theta_L(t). \tag{5.44}$$

In each case the Dirichlet boundary conditions including the temperature at the interface are strongly imposed.

5.1.1 Construction of the finite element form

We solve the derived system using a finite element method. Since we have Dirichlet boundary conditions on equation (5.38) for the velocity, and also on (5.43) and (5.44) for the temperature, we use the modified piecewise linear weight functions $w_i = \tilde{W}_i$ of 4.2.1. We define an approximation to each of our variables in terms of a weighted linear combination of the W_j. These are given in Appendix A.

We also define the weightings c_i of (5.14) in terms of the same $\tilde{W}_i$. These are

$$\sum_{j \in Z_S} \left[\int_{R(t)} \tilde{W}_i W_j \; dx \right] U_{S_j} = \tilde{c}_{i_S} \theta_S(t) \tag{5.45}$$

$$\sum_{j \in Z_L} \left[\int_{R(t)} \tilde{W}_i W_j \; dx \right] U_{L_j} = \tilde{c}_{i_L} \theta_L(t). \tag{5.46}$$

Here Z_S and Z_L are the sets of nodes in the solid and liquid phases respectively. We can now express the system in finite element form. We make substitutions as necessary from equations (A.5) to (A.14) into equations (5.34) and (5.35) so that all our variables are expressed in terms of their piecewise linear approximations. We obtain

$$\begin{aligned} \tilde{c}_{i_S} \dot{\theta}_S + \sum_{j \in Z_S} \left[\int_{R_S(t)} U_S \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} \; dx \right] \Phi_j = &- \sum_{j \in Z_S} \left[\int_{R_S(t)} \kappa_S \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} \; dx \right] U_{S_j} + \tilde{W}_i \kappa_S \frac{\partial U_S}{\partial x} \bigg|_{R_m(t)} \\ &- \tilde{W}_i \kappa_S \frac{\partial U_S}{\partial x} \bigg|_{R_f} + \frac{U_m}{\lambda} \left(k_S \tilde{W}_i \frac{\partial U_S}{\partial x} \bigg|_{R_m(t)} - k_L \tilde{W}_i \frac{\partial U_L}{\partial x} \bigg|_{R_m(t)} \right) \end{aligned} \tag{5.47}$$

$$\begin{aligned} \tilde{c}_{i_L} \dot{\theta}_L + \sum_{j \in Z_L} \left[\int_{R_m(t)} U_L \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} \; dx \right] \Phi_j = &- \sum_{j \in Z_L} \left[\int_{R_L(t)} \kappa_L \frac{\partial \tilde{W}_i}{\partial x} \frac{\partial W_j}{\partial x} \; dx \right] U_{L_j} + \tilde{W}_i \kappa_L \frac{\partial U_L}{\partial x} \bigg|_{R_f} \\ &- \tilde{W}_i \kappa_L \frac{\partial U_L}{\partial x} \bigg|_{R_m(t)} - \frac{U_m}{\lambda} \left(k_S \tilde{W}_i \frac{\partial U_S}{\partial x} \bigg|_{R_m(t)} - k_L \tilde{W}_i \frac{\partial U_L}{\partial x} \bigg|_{R_m(t)} \right). \end{aligned} \tag{5.48}$$

In matrix form (5.47) is

$$\tilde{K}(\underline{U}_S) \;\; \underline{\Phi}_S = \underline{\tilde{f}}_S \tag{5.49}$$

where $\tilde{K}(\underline{U}_S)$ is the weighted stiffness matrix of 3.1.2 constructed with modified basis functions, and $\underline{\Phi}_S$ is the vector containing the values of Φ_{S_j}, and $\underline{\tilde{f}}_S$ is a vector with entries $\tilde{f}_{S_i}$

given by

$$\tilde{f}_{S_i} = -\tilde{c}_{i_S}\dot{\theta}_S - \sum_{j\in Z_S}\left[\int_{R_S(t)}\kappa_S\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\,dx\right]U_{S_j} + \tilde{W}_i\kappa_S\frac{\partial U_S}{\partial x}\bigg|_{R_m(t)}$$
$$-\tilde{W}_i\kappa_S\frac{\partial U_S}{\partial x}\bigg|_{R_f} + \frac{U_m}{\lambda}\left(k_S\tilde{W}_i\frac{\partial U_S}{\partial x}\bigg|_{R_m(t)} - k_L\tilde{W}_i\frac{\partial U_L}{\partial x}\bigg|_{R_m(t)}\right). \tag{5.50}$$

Similarly, (5.48) can be expressed as

$$\tilde{K}(\underline{U})_L\ \underline{\Phi}_L = \underline{\tilde{f}}_L \tag{5.51}$$

with the vector $\underline{\tilde{f}}_L$ containing entries $\tilde{f}_{L_i}$ given by

$$\tilde{f}_{L_i} = -\tilde{c}_{i_L}\dot{\theta}_L - \sum_{j\in Z_L}\left[\int_{R_L(t)}\kappa_L\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\,dx\right]U_{L_j} + \tilde{W}_i\kappa_L\frac{\partial U_L}{\partial x}\bigg|_{R_f}$$
$$-\tilde{W}_i\kappa_L\frac{\partial U_L}{\partial x}\bigg|_{R_{m(t)}} - \frac{U_m}{\lambda}\left(k_S\tilde{W}_i\frac{\partial U_S}{\partial x}\bigg|_{R_m(t)} - k_L\tilde{W}_i\frac{\partial U_L}{\partial x}\bigg|_{R_m(t)}\right). \tag{5.52}$$

The matrix systems can be solved to obtain Φ_L and Φ_R. Since the weighted stiffness matrices $\tilde{K}(\underline{U})_S$ and $\tilde{K}(\underline{U})_L$ are singular, we have an infinity of solutions available and we set $\Phi = 0$ at the interface node to reduce the system in order to give a unique solution. Note that the expressions for $\dot{\theta}_S$ (5.36) and $\dot{\theta}_L$ (5.37) can be obtained and solved in a straightforward manner by simply summing over the rows of (5.49) and (5.51).
To recover $\dot{x}$, we use the approximation

$$\dot{X} = \sum_j \dot{X}_j W_j. \tag{5.53}$$

We substitute this into equation (5.38) to obtain the finite element form

$$\sum_{j\in Z_S\cup Z_L}\left[\int_{R(t)}\tilde{W}_i W_j\,dx\right]\dot{X}_j = \sum_{j\in Z_S\cup Z_L}\left[\int_{R(t)}\tilde{W}_i\frac{\partial W_j}{\partial x}\,dx\right]\Phi_j \tag{5.54}$$

or in matrix form,

$$M\underline{\dot{X}} = B\underline{\Phi}. \tag{5.55}$$

We impose the velocity on the interface obtained from (5.39), and we impose $\dot{X} = 0$ on the external boundaries. Since we are using modified weight functions we will not interfere with the compatibility condition (5.40) by doing so. We solve (5.55) for the remaining velocities.

As in [33], we now move the nodes using Heun's scheme [30]. Using the same scheme, we update the values of θ_S and θ_L from the values of $\dot{\theta}_S$ (5.36) and $\dot{\theta}_L$ (5.37). The final step is the recovery of U. We can obtain U on the updated grid from the relative conservation of mass equations (5.45) and (5.46). These can be expressed in matrix form as

$$\tilde{M}\underline{U}_S = \underline{\tilde{c}}_{S_i}\theta_S(t) \tag{5.56}$$

and

$$\tilde{M}\underline{U}_L = \underline{\tilde{c}}_{L_i}\theta_L(t). \tag{5.57}$$

In the initial set up, we set $t = 0$ and use (5.56) and (5.57) to find the constant vectors $\underline{\tilde{c}}_{S_i}$ and $\underline{\tilde{c}}_{L_i}$. Then for each subsequent time step we proceed as follows. We take θ_S and θ_L at the new time step. We calculate the mass matrix $\tilde{M}$ for the updated grid. We can then obtain the updated $\underline{U}_S$ and $\underline{U}_L$ from inversions of (5.56) and (5.57) respectively.

Algorithm 11

The finite element solution of the Stefan problem given by equations (5.1) and with an interface condition given by (5.2) on the moving mesh in 1-D therefore consists of the following steps. We first add a constant to the domain so that we avoid any zero or negative values for U. Having obtained the values of $\tilde{c}_{S_i}$ and $\tilde{c}_{L_i}$ from (5.56) and (5.57):

1. Find the velocity potential by solving equation (5.49) and (5.51) for the $\Phi_j(t)$ values;
2. Find the node velocity by solving equation (5.55) for the $\dot{X}_j(t)$ values;
3. Generate the co-ordinate system at the next time-step by solving (3.18) using Heun's approximation;
4. Update the values of θ_S and θ_L from the values of $\dot{\theta}_S$ (5.36) and $\dot{\theta}_L$ (5.37);
5. Find the solution $U(t+dt)$ by solving the conservation equations (5.56) and (5.57).

5.1.2 Results

We are able to replicate the results given visually in [8], with suitable node movement and interface movement. This lends confidence that the use of modified basis functions throughout is appropriate. Since an exact solution is available, we are able to calculate the order of convergence for both solution accuracy and interface position. Figure 5.1 shows

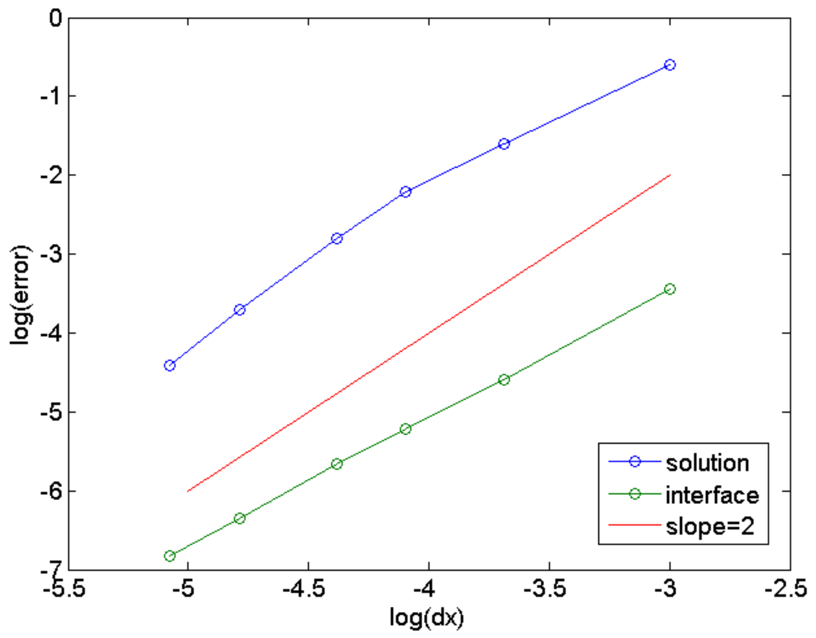


Fig. 5.1 Comparison of L^2 errors in the solution and the magnitudes of the errors in the interface node position for the two-phase Stefan problem in one space dimension, at $T = 0.5$. We observe an order of convergence of $p \approx 2$.

the convergence of the L^2 errors at $T = 0.5$ as the mesh resolution is increased. Both the normalised solution error and the interface position error have an order of convergence of approximately $p = 2$. This is consistent with the order of convergence given in [8] and demonstrates that this method is an acceptable alternative.

5.2 The two phase model of competition-diffusion

We now turn our attention to competition-diffusion models, in particular the Lotka-Volterra systems of theoretical ecology. As described in Chapter 2, there are many variations. After considering the Stefan problem, we find the 2003 paper by Hilhorst *et al.* [31] of particular interest. In [31] it is demonstrated that where competition is strong enough to spatially segregate two populations, the Lotka-Volterra PDE system can be reduced to a form that is similar to the Stefan problem. Two major differences remain. Firstly, in the competition system, we have logistic growth terms. Secondly, in the competition system there is a parameter set equal to zero that is the equivalent of the latent heat coefficient of the Stefan problem. In physical terms, one species does not transform into another. This means that the competition system has an interface condition that does not directly give an interface velocity. This presents a challenge when attempting the same approach as we used for the Stefan problem, because in the Stefan problem the interface velocity is taken directly from the interface condition. However, we feel that the approach in [8] is an excellent way to model the competition system, because not only will it allow us to fully track the evolution of the interface between species, it provides a framework for keeping particular mesh nodes attached to particular species. This means that the internal dynamics of a species can be assigned to particular nodes rather than a particular space, and the dynamics for any given location will automatically be those of the correct species.

We therefore proceed to model the system set out by Hilhorst *et al.* [31], using a method developed from [8], in one dimension. We present a solution for determining the interface velocity, and furthermore demonstrate that the method in [8] can be extended to include logistic growth terms.

We have a two component reaction-diffusion system given by the Lotka-Volterra system,

$$\frac{\partial u_1}{\partial t} = \delta_1 \frac{\partial^2 u_1}{\partial x^2} + f(u_1, u_2) u_1 \qquad t > 0, x \in [a, b] \tag{5.58}$$

$$\frac{\partial u_2}{\partial t} = \delta_2 \frac{\partial^2 u_2}{\partial x^2} + g(u_1, u_2) u_2 \qquad t > 0, x \in [a, b] \tag{5.59}$$

where δ_1, δ_2 are constant diffusion coefficients, and with, in general

$$f(u_1,u_2) = r_1\left(1 - \frac{u_1 + K_1 u_2}{k_1}\right) \tag{5.60}$$

$$g(u_1,u_2) = r_2\left(1 - \frac{u_2 + K_2 u_1}{k_2}\right). \tag{5.61}$$

Here u_1 and u_2 are the population densities of two competing species, the k are the respective carrying capacities of the species, the K are the species specific competition rates, and r is a reproductive rate parameter. The Hilhorst paper [31] demonstrates that this system can be reduced, if we have two species completely segregated, to

$$f(u_1,u_2) = r_1(1 - u_1/k_1) \tag{5.62}$$

$$g(u_1,u_2) = r_2(1 - u_2/k_2). \tag{5.63}$$

The resulting system represents the limit where the K values are very large; the competition rate is high enough that the two species cannot coexist in space. In the area populated by species 1, $u_2 = 0$, and in the area populated by species 2, $u_1 = 0$. At the interface, we have a condition that gives the relationship between the fluxes of the two species. In essence, the species both flow into the interface and annihilate each other in a ratio according to the competition coefficient, μ. This condition is given by [31] as

$$\mu\delta_1\frac{\partial u_1}{\partial x} = -\delta_2\frac{\partial u_2}{\partial x} \tag{5.64}$$

where $\mu = K_2/K_1$. We will call μ the interspecies competition rate. We work with Neumann boundary conditions on the external boundaries, which will be fixed,

$$\left.\frac{\partial u_1}{\partial x}\right|_a = 0 \qquad t > 0 \tag{5.65}$$

$$\left.\frac{\partial u_2}{\partial x}\right|_b = 0 \qquad t > 0. \tag{5.66}$$

Because the annihilation is complete, we also have a Dirichlet condition at the interface, m;

$$u|_m = 0 \qquad t > 0. \tag{5.67}$$

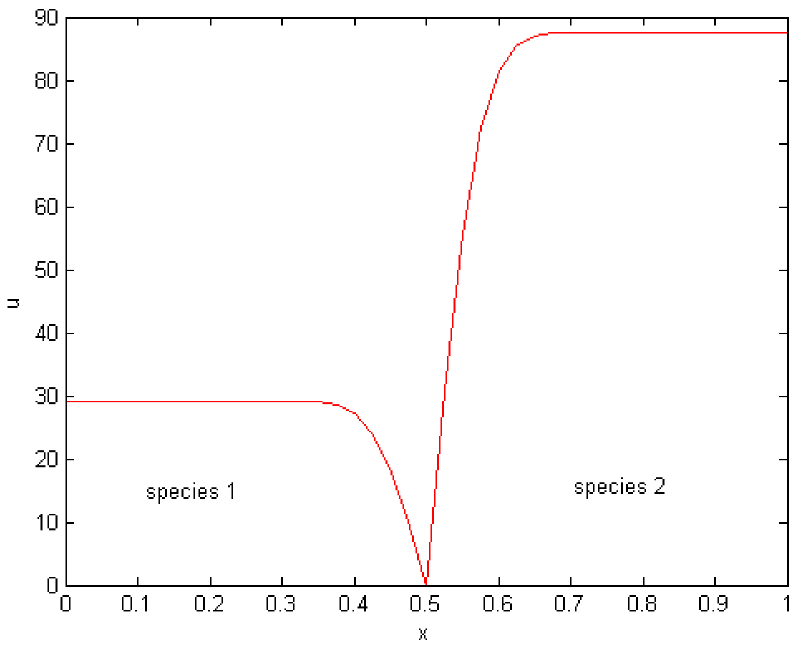


Fig. 5.2 Initial conditions for the competition system, with population density U_1 of species 1 (on the left) and U_2 of species 2 (on the right). The interface node has zero population and must always satisfy the interface condition

The initial conditions are not given in [31], but we select suitable initial conditions and physical parameters such that the interface condition is satisfied at $t = 0$, and so that we have one species in growth and one in decline. The initial conditions are shown in figure (5.2). We begin by writing the driving Lotka-Volterra equations (5.58) and (5.59) in the weak forms,

$$\int_{R(t)} w_i \frac{\partial u_1}{\partial t}\ dx = \int_{R(t)} \delta_1 w_i \frac{\partial^2 u_1}{\partial x^2}\ dx + \int_{R(t)} w_i r_1 u_1 \left(1 - \frac{u_1}{k_1}\right)\ dx \tag{5.68}$$

$$\int_{R(t)} \frac{\partial u_2}{\partial t}\ dx = \int_{R(t)} \delta_2 w_i \frac{\partial^2 u_2}{\partial x^2}\ dx + \int_{R(t)} w_i r_2 u_2 \left(1 - \frac{u_2}{k_2}\right)\ dx. \tag{5.69}$$

We then define the total population of a species as θ, given by

$$\theta(t) = \int_{R(t)} u\ dx \tag{5.70}$$

where $R(t)$ is the moving domain inhabited by that species. We write a relative conservation principle in terms of θ, as

$$\frac{1}{\theta(t)}\int_{R(t)} u\ dx = 1. \tag{5.71}$$

We write this in a weighted form, introducing the weight function w_i,

$$\frac{1}{\theta(t)}\int_{R(t)} w_i u\ dx = c_i \tag{5.72}$$

or

$$\int_{R(t)} w_i u\ dx = c_i\theta(t) = c_i\int_{R(t)} u\ dx \tag{5.73}$$

where c_i is independent of time. The constant c_i is determined by the choice of weighting w_i. All of the weightings together should be chosen to provide a partition of unity. We differentiate (5.73) with respect to time using the Leibnitz integral rule on our moving frame $R(t)$,

$$\frac{d}{dt}\left[\int_{R(t)} w_i u\ dx\right] = \int_{R(t)}\left(\frac{\partial(w_i u)}{\partial t} + \frac{\partial}{\partial x}(w_i u\dot{x})\right)\ dx. \tag{5.74}$$

We impose the condition that the basis functions w_i move with the domain. Hence the basis functions also have velocity $\dot{x}$ and therefore

$$\frac{\partial w_i}{\partial t} + \dot{x}\frac{\partial w_i}{\partial x} = 0 \tag{5.75}$$

hence

$$\frac{d}{dt}\left[\int_{R(t)} w_i u\ dx\right] = \int_{R(t)} w_i\left(\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(u\dot{x})\right)\ dx \tag{5.76}$$

or

$$\frac{d}{dt}\left[\int_{R(t)} w_i u\ dx\right] - \int_{R(t)} w_i\frac{\partial}{\partial x}(u\dot{x})\ dx = \int_{R(t)} w_i\frac{\partial u}{\partial t}\ dx. \tag{5.77}$$

We write this in terms of $\dot{\theta}$ and the constants c_i to give

$$c_i\dot{\theta} - \int_{R(t)} w_i\frac{\partial}{\partial x}(u\dot{x})\ dx = \int_{R(t)} w_i\frac{\partial u}{\partial t}\ dx. \tag{5.78}$$

We introduce the velocity potential ϕ, defined by

$$\dot{x} = \frac{\partial\phi}{\partial x} \tag{5.79}$$

so that

$$c_i\dot{\theta} - \int_{R(t)} w_i\frac{\partial}{\partial x}\left(u\frac{\partial\phi}{\partial x}\right)\ dx = \int_{R(t)} w_i\frac{\partial u}{\partial t}\ dx \tag{5.80}$$

or, after integration by parts,

$$c_i\dot{\theta} + \int_{R(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[uw_i\frac{\partial \phi}{\partial x}\right]_{\partial R(t)} = \int_{R(t)} w_i\frac{\partial u}{\partial t}\ dx. \tag{5.81}$$

We substitute in a weak form of the driving PDE, either (5.68) or (5.69), depending on the phase under consideration. For either phase $p \in [1,2]$

$$\begin{aligned} c_p\dot{\theta}_p + \int_{R_p(t)} u_p\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[u_pw_i\frac{\partial \phi}{\partial x}\right]_{\partial R_p(t)} &= \int_{R_p(t)} w_i\delta_p\frac{\partial^2 u_p}{\partial x^2}\ dx \\ &+ \int_{R_p(t)} w_iu_pr_p\left(1-\frac{u_p}{k_p}\right)\ dx. \end{aligned} \tag{5.82}$$

Again integrating by parts, this time on the right hand side

$$\begin{aligned} c_p\dot{\theta} + \int_{R_p(t)} u_p\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\ dx - \left[u_pw_i\frac{\partial \phi}{\partial x}\right]_{\partial R_p(t)} &= -\int_{R_p(t)} \delta_p\frac{\partial w_i}{\partial x}\frac{\partial u_p}{\partial x}\ dx + \left[w_i\delta_p\frac{\partial u_p}{\partial x}\right]_{\partial R_p(t)} \\ &+ \int_{R_p(t)} w_iu_pr_p\left(1-\frac{u_p}{k_p}\right)\ dx. \end{aligned} \tag{5.83}$$

We now consider each species separately, on their respective domains.We have R_1 with boundary $\partial R_1 = [0,m]$, and R_2 with boundary $\partial R_2 = [m,1]$. The masses of equation (5.70) are defined in terms of each species as

$$\theta_1(t) = \int_{R_1(t)} u_1\ dx \tag{5.84}$$

$$\theta_2(t) = \int_{R_2(t)} u_2\ dx. \tag{5.85}$$

The weak forms are then

$$c_{1_i}\theta_1(t) = \int_{R_1(t)} w_iu_1\ dx \tag{5.86}$$

$$c_{2_i}\theta_2(t) = \int_{R_2(t)} w_iu_2\ dx. \tag{5.87}$$

Equation (5.83) can be written now for each phase separately. For species 1 it becomes

$$c_{1_i}\dot{\theta}_1 + \int_{R_1(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx - \left[u_1 w_i \frac{\partial \phi}{\partial x}\right]_{\partial R_1(t)} = -\int_{R_1(t)} \delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_1}{\partial x}\, dx + \left[w_i \delta_1 \frac{\partial u_1}{\partial x}\right]_{\partial R_1(t)} + \int_{R_1(t)} w_i u_1 r_1 \left(1 - \frac{u_1}{k_1}\right) dx. \tag{5.88}$$

At the external boundaries $\frac{\partial u}{\partial x} = 0$ (5.65), and also $\frac{\partial \phi}{\partial x} = 0$ because the boundaries are fixed. Together with the condition that $u = 0$ (5.67) on the interface boundary, we see that certain terms will be equal to zero. The resulting equation for the velocity potentials for species 1 is given by

$$c_{1_i}\dot{\theta}_1 + \int_{R_1(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx = -\int_{R_1(t)} \delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_1}{\partial x}\, dx + w_i \delta_1 \frac{\partial u_1}{\partial x}\bigg|_{m(t)} + \int_{R_1(t)} w_i u_1 r_1 \left(1 - \frac{u_1}{k_1}\right) dx \tag{5.89}$$

and the velocity potentials for species 2 are given by

$$c_{2_i}\dot{\theta}_2 + \int_{R_2(t)} u_2 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx - \left[u_2 w_i \frac{\partial \phi}{\partial x}\right]_{\partial R_2(t)} = -\int_{R_2(t)} \delta_2 \frac{\partial w_i}{\partial x}\frac{\partial u_2}{\partial x}\, dx + \left[w_i \delta_2 \frac{\partial u_2}{\partial x}\right]_{\partial R_2(t)} + \int_{R_2(t)} w_i u_2 r_2 \left(1 - \frac{u_2}{k_2}\right) dx. \tag{5.90}$$

Again we may use the boundary conditions and set certain terms to zero. The equation for species 2 is then

$$c_{2_i}\dot{\theta}_2 + \int_{R_2(t)} u_2 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\, dx = -\int_{R_2(t)} \delta_2 \frac{\partial w_i}{\partial x}\frac{\partial u_2}{\partial x}\, dx - w_i \delta_2 \frac{\partial u_2}{\partial x}\bigg|_{m(t)} + \int_{R_2(t)} w_i u_2 r_2 \left(1 - \frac{u_2}{k_2}\right) dx. \tag{5.91}$$

We have now reached the stage where, if this were the Stefan problem, we would use the Stefan condition at the interface to give the velocity of the interface node. We recall that the Stefan interface condition is given by

$$k_S \frac{\partial u_S}{\partial x} - k_L \frac{\partial u_L}{\partial x} = \lambda \dot{x} \tag{5.92}$$

and hence gives $\dot{x}$, the velocity of the interface directly. In contrast, the interface condition

(5.64) for the competition system is

$$\mu\delta_1 \frac{\partial u_1}{\partial x} = -\delta_2 \frac{\partial u_2}{\partial x} \tag{5.93}$$

which is equivalent to the Stefan condition with $\lambda = 0$ and does not contain the velocity $\dot{x}$. Note also that the Stefan condition is relevant to a situation where the gradients of u either side of the interface are of the same sign in general. In contrast, equation (5.64) is relevant to an interface where the gradients either side are of opposite polarity. Since $u = 0$ on the interface and we can't have a negative mass, we are in effect considering 'v' shaped interfaces. We note that whilst the interface velocity is not given by (5.64), the expression does implicitly contain information about the location of the interface. In particular, if we know the position of the mesh points adjacent to the interface and also the values of u at those points, we may use the fact that $u = 0$ at the interface to infer an interface position that satisfies (5.64). We select an interface position such that the values of $\frac{\partial u}{\partial x}$ either side of the interface are in the ratio $-\mu$. We proceed as follows. At a given time step t_N we write the interface condition (5.64) in a finite difference form

$$\mu\delta_1 \frac{u^N_{1_m} - u^N_{1_{m-1}}}{x_m - x^N_{m-1}} = -\delta_2 \frac{u^N_{2_{m+1}} - u^N_{2_m}}{x^N_{m+1} - x_m} \tag{5.94}$$

where the subscript m denotes the interface node, and the x_i are the spatial co-ordinates of the nodes. We have that $u_m = 0$, and so we can obtain an expression for the position of the interface node, x_m,

$$x^{N+1}_m = \frac{(\mu\delta_1 u^N_{1_{m-1}} x^N_{m+1} + \delta_2 u^N_{2_{m+1}} x^N_{m-1})}{(\mu\delta_1 u^N_{1_{m-1}} + \delta_2 u^N_{2_{m+1}})}. \tag{5.95}$$

If we had used Euler integration for the time integration then we could have simply imposed this new interface position directly, but here it is more helpful to calculate a velocity. This grants more flexibility in the chosen time integration scheme. Here we use the finite difference approximation

$$\dot{x}^{N+1}_m = \frac{\left(\frac{(\mu\delta_1 u^N_{1_{m-1}} x^N_{m+1} + \delta_2 u^N_{2_{m+1}} x^N_{m-1})}{(\mu\delta_1 u^N_{1_{m-1}} + \delta_2 u^N_{2_{m+1}})} - x^N_m\right)}{dt}. \tag{5.96}$$

Note that through this method it is only possible to generate an adapted interface position once a distortion of the interface condition already exists. This interface condition cannot predict where the interface ought to move to in anticipation of a forthcoming violation of the interface condition. We must allow the solution around the interface to evolve first, and then adapt the interface position in response to that. We cannot generate the position of the interface that satisfies (5.64) at the same time as we find the node velocities elsewhere, because we must solve the system for u on the updated grid before we can see where the interface ought to be positioned. After we have solved for u, we can obtain the interface position resulting from those u values, but we cannot impose it on the system straightaway. We would violate conservation of mass by doing so. Instead, we determine the new position at the next time step. A concern this raises is whether the interface position is effectively imposed one time step behind where it should be. The condition (5.64) is always slightly violated, since it is this violation that drives the interface movement. Philosophically, we can reconcile this difficulty by considering that there ought to be a force driving a movement of the interface *before* the interface starts to respond. In the real world, would our species retreat in anticipation of competition, or else compete and then accept the resulting boundary change? The subtleties of this interaction, and its timing or lag, are not considered in the Lotka-Volterra equations. We can therefore be confident that the explicit nature of our system does not violate any conditions of the system, and indeed it may better reflect reality than a predictive approach. Should we determine that a problem does exist in this regard, a suitable solution would be to use an implicit time integration method, which would accord the ability to reassign the interface movement to the prior time step if so desired.

With this treatment of the interface node in mind, we do not calculate (5.89) and (5.91) for the interface node itself. Instead, once in the finite element framework we will use modified basis functions either side of the interface that will allow strong imposition of the interface position. In taking this approach we will obtain versions of (5.89) and (5.91) for all nodes except the interface node. The mass from the interface node is assigned to the adjacent nodes through the modified basis functions, so mass conservation is preserved.

We note that we require the rate of change of mass, $\dot{\theta}$, in each phase. This can be obtained by summing over all i in equations (5.89) and (5.91) as appropriate. With a choice of w_i forming a partition of unity, and recalling that $\sum_i c_{p_i} = 1$, we obtain for the sum over equation (5.89)

$$\dot{\theta}_1 = \delta_1 \frac{\partial u_1}{\partial x}\bigg|_{m(t)} + \int_{R(t)} u_1 r_1 \left(1 - \frac{u_1}{k_1}\right) dx \tag{5.97}$$

and for the sum over equation (5.91)

$$\dot{\theta}_2 = -\,\delta_2 \frac{\partial u_2}{\partial x}\bigg|_{m(t)} + \int_{R(t)} u_2 r_2 \left(1 - \frac{u_2}{k_2}\right) dx. \tag{5.98}$$

To recover the nodal velocities we first solve (5.97) and (5.98) for $\dot{\theta}$ in each phase. We then solve (5.89) and (5.91) to give ϕ in each phase, but not on the interface node due to the modified basis functions we will use. We then return to our definition of ϕ (5.79), now written in distributed form,

$$\int_{R(t)} w_i \dot{x} \; dx = \int_{R(t)} w_i \frac{\partial \phi}{\partial x} \; dx \tag{5.99}$$

which can be solved for $\dot{x}$. Having obtained $\dot{x}$, we move the domain using the explicit Euler integration scheme. We also update θ_1 and θ_2 from $\dot{\theta}_1$ (5.97) and $\dot{\theta}_2$ (5.98) using the same time integration procedure. For the interface itself, we calculate the new position by correcting the interface condition at the prior time step. We obtain the resultant interface velocity by solving equation (5.96) with $u = 0$ in the interface node.

We may now recover u. We determine the constant partial masses c_{1_i} and c_{2_i} from (5.86) and (5.87) and the initial conditions. We obtain, for $t = 0$

$$c_{1_i} = \frac{1}{\theta_1(0)} \int_{R_1(0)} w_i(x,0) u(x,0) \; dx \tag{5.100}$$

$$c_{2_i} = \frac{1}{\theta_2(0)} \int_{R_2(0)} w_i(x,0) u(x,0) \; dx. \tag{5.101}$$

We then use (5.86) and (5.87) again, to recover u_1 and u_2. We require θ_1 and θ_2 at the new time step. We move the weight functions with the domain. For species 1, u_1 can be recovered from

$$\int_{R_1(t)} w_i(x,t) u_1(x,t) \; dx = c_{1_i}(x) \theta_1(t) \tag{5.102}$$

and for species 2, u_2 can be recovered from

$$\int_{R_2(t)} w_i(x,t) u_2(x,t) \; dx = c_{2_i}(x) \theta_2(t). \tag{5.103}$$

In each case the Dirichlet condition that $u = 0$ at the interface is strongly imposed, and the Neumann condition at the external boundaries is also strongly imposed.

We solve the derived system using a finite element method. We have Dirichlet boundary conditions on equation (5.99) for the velocity, at both the interface and external boundaries. For the values of u_1 and u_2, given by equations (5.102) and (5.103), we have a Dirichlet condition at the interface only. At the external boundaries we have Neumann boundary conditions instead. However, all these conditions are compatible with using the modified piecewise linear weight functions $w_i = \tilde{W}_i$ of 4.2.1, with a modified weight function at each external boundary, and also at each side of the interface. We may then strongly impose the values of the velocity and u_1 and u_2 at the interfaces and external boundaries. The values of u_1 and u_2 at the external boundaries can be transferred from their adjacent nodes because we have the Neumann conditions.

5.2.1 Construction of the finite element form

We begin the finite element method implementation by defining an approximation to each of our variables in terms of a weighted linear combination of the W_j. These are given in Appendix A. The weightings c_{p_i} of (5.102) and (5.103) are likewise defined in terms of $\tilde{W}_i$. We obtain

$$\sum_{j\in Z_1}\left[\int_{R(t)} \tilde{W}_i W_j \; dx\right] U_{1_j} = \tilde{c}_{1_i}\theta_1(t) \tag{5.104}$$

$$\sum_{j\in Z_2}\left[\int_{R(t)} \tilde{W}_i W_j \; dx\right] U_{2_j} = \tilde{c}_{2_i}\theta_2(t) \tag{5.105}$$

where Z_i is the set of nodes in phase i. We may rewrite the system in finite element form. We take the approximations (A.1) to (A.9) as necessary, and also (5.104) to (5.105), and make substitutions as necessary into equations (5.89) and (5.91). We obtain the following, with all variables now expressed in terms of their piecewise linear approximations. Equation (5.89) becomes

$$\begin{aligned}\tilde{c}_{1_i}\dot{\theta}_1 + \sum_{j\in Z_1}\left[\int_{R_1(t)} U_1 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\; dx\right]\Phi_j = &-\sum_{j\in Z_1}\left[\int_{R_1(t)} \delta_1 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\; dx\right] U_{1_j} \\ &+ \tilde{W}_i\delta_1 \frac{\partial U_1}{\partial x}\bigg|_{R_m(t)} + \sum_{j\in Z_1}\left[\int_{R_1(t)} \tilde{W}_i W_j r_1 \; dx\right] U_{1_j} - \int_{R_1(t)} \frac{r_1}{k_1}\tilde{W}_i U_1^2 \; dx \end{aligned} \tag{5.106}$$

and equation (5.91) becomes

$$\begin{aligned}\tilde{c}_{2_i}\dot{\theta}_2 + \sum_{j\in Z_2}\left[\int_{R_2(t)} U_2 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\; dx\right]\Phi_j = &-\sum_{j\in Z_2}\left[\int_{R_2(t)} \delta_2 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\; dx\right] U_{2_j} \\ &- \tilde{W}_i\delta_2 \frac{\partial U_2}{\partial x}\bigg|_{R_m(t)} + \sum_{j\in Z_2}\left[\int_{R_2(t)} \tilde{W}_i W_j r_2 \; dx\right] U_{2_j} - \int_{R_2(t)} \frac{r_2}{k_2}\tilde{W}_i U_2^2 \; dx. \end{aligned} \tag{5.107}$$

In matrix form (5.106) is expressed as

$$\tilde{K}(\underline{U}_1)\ \underline{\Phi}_1 = \underline{\tilde{f}}_1 \tag{5.108}$$

where $\tilde{K}(\underline{U}_1)$ is the weighted stiffness matrix of Chapter 3, section 3.1.2, constructed with the modified basis functions $\tilde{W}_i$, and $\underline{\Phi}_1$ is the vector containing the values of Φ_{1_j}, and $\underline{\tilde{f}}_1$ is a vector with entries $\tilde{f}_{1_i}$ given by

$$\begin{aligned}\tilde{f}_{1_i} = &-\tilde{c}_{1_i}\dot{\theta}_1 - \sum_{j\in Z_1}\left[\int_{R_1(t)} \delta_1 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\ dx\right] U_{1_j} \\ &+ \tilde{W}_i\delta_1 \frac{\partial U_1}{\partial x}\bigg|_{R_m(t)} + \sum_{j\in Z_1}\left[\int_{R_1(t)} \tilde{W}_i W_j r_1\ dx\right] U_{1_j} - \int_{R_1(t)} \frac{r_1}{k_1}\tilde{W}_i U_1^2\ dx. \end{aligned} \tag{5.109}$$

Similarly, (5.107) can be expressed as

$$\tilde{K}(\underline{U}_2)\underline{\Phi}_2 = \underline{\tilde{f}}_2 \tag{5.110}$$

with the vector $\underline{\tilde{f}}_2$ containing entries $\tilde{f}_{2_i}$ given by

$$\begin{aligned}\tilde{f}_{2_i} = &-\tilde{c}_{2_i}\dot{\theta}_2 - \sum_{j\in Z_2}\left[\int_{R_2(t)} \delta_2 \frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}\ dx\right] U_{2_j} \\ &- \tilde{W}_i\delta_2 \frac{\partial U_2}{\partial x}\bigg|_{R_m(t)} + \sum_{j\in Z_2}\left[\int_{R_2(t)} \tilde{W}_i W_j r_2\ dx\right] U_{2_j} - \int_{R_2(t)} \frac{r_2}{k_2}\tilde{W}_i U_2^2\ dx. \end{aligned} \tag{5.111}$$

The nonlinear terms in (5.109) and (5.111) are evaluated exactly using Simpson's rule (4.64). The matrix systems can be solved to obtain Φ_1 and Φ_2. Since the weighted stiffness matrices $\tilde{K}(\underline{U}_1)$ and $\tilde{K}(\underline{U}_2)$ are singular, we have an infinity of solutions available and we set $\Phi = 0$ at a selected node to collapse the system to give a single solution. Note that the expressions for $\dot{\theta}_1$ (5.97) and $\dot{\theta}_2$ (5.98) can be obtained and solved in a straightforward manner by simply summing over the rows of (5.108) and (5.110). Equation (5.97) becomes

$$\dot{\theta}_1 = \delta_1 \frac{\partial U_1}{\partial x}\bigg|_{R_m} + \int_{R(t)} u_1 r_1 \left(1 - \frac{u_1}{k_1}\right)\ dx \tag{5.112}$$

and equation (5.98) becomes

$$\dot{\theta}_2 = -\delta_2 \frac{\partial U_2}{\partial x}\bigg|_{R_m} + \int_{R(t)} u_2 r_2 \left(1 - \frac{u_2}{k_2}\right)\ dx, \tag{5.113}$$

for which the nonlinear terms may be computed exactly using Simpson's rule (4.64). To recover $\dot{X}$, we use the approximation

$$\dot{X} = \sum_{j \in Z_1 \cup Z_2} \dot{X}_j W_j. \tag{5.114}$$

We substitute this into equation (5.99) to obtain the finite element form

$$\sum_{j \in Z_1 \cup Z_2} \left[\int_{R(t)} \tilde{W}_i W_j \, dx \right] \dot{X}_j = \sum_{j \in Z_1 \cup Z_2} \left[\int_{R(t)} \tilde{W}_i \frac{\partial W_j}{\partial x} \, dx \right] \Phi_j \tag{5.115}$$

or in matrix form

$$\tilde{M} \underline{\dot{X}} = \tilde{B}(u) \underline{\Phi}. \tag{5.116}$$

We impose $v = 0$ on the external boundaries. We impose the interface velocity obtained from (5.96). Since we are using modified weight functions we will not interfere with the compatibility condition (5.40) by doing so. We solve (5.116) for the remaining velocities. We move the nodes using Euler's scheme. Using the same scheme, we update the values of θ_1 and θ_2 from the values of $\dot{\theta}_1$ (5.112) and $\dot{\theta}_2$ (5.113). We may now recover the values of U_1 and U_2. We can obtain U on the updated grid from the relative conservation of mass equations (5.104) and (5.105). In matrix form these are

$$\tilde{M}_1 \underline{U}_1 = \underline{\tilde{c}}_{1_i} \theta_1(t) \tag{5.117}$$

and

$$\tilde{M}_2 \underline{U}_2 = \underline{\tilde{c}}_{2_i} \theta_2(t). \tag{5.118}$$

At the initial set up, we set $t = 0$ and use (5.117) and (5.118) to find the constants $\underline{\tilde{c}}_{1_i}$ and $\underline{\tilde{c}}_{2_i}$. Then for each subsequent time step we proceed as follows. We take θ_1 and θ_2 at the new time step. We calculate the mass matrix $\tilde{M}$ for the updated grid. We can then obtain the updated $\underline{U}_1$ and $\underline{U}_2$ from inversions of (5.117) and (5.118) respectively.

Algorithm 12

The finite element solution of the competition problem given by equations (5.58) and (5.59) and with an interface condition given by (5.64) on the moving mesh in 1-D therefore consists of the following steps. We obtain the constant values of $\tilde{c}_{1_i}$ and $\tilde{c}_{2_i}$ from (5.117) and (5.118), and for each time step:

1. Find the velocity potential by solving equation (5.108) and (5.110) for the $\Phi_j(t)$ val-

ues;

2. Find the internal node velocity by solving equation (5.116) for the $\dot{X}_j(t)$ values;
3. Find the interface node velocity by solving equation (5.96) for the $\dot{X}_m(t)$ value;
4. Generate the co-ordinate system at the next time-step $t+dt$ by solving (3.18) using Euler's approximation;
5. Update the values of θ_1 and θ_2 from the values of $\dot{\theta}_1$ (5.112) and $\dot{\theta}_2$ (5.113);
6. Find the solutions $U_1(t+dt)$ and $U_2(t+dt)$ by solving the conservation equations (5.117) and (5.118).

5.2.2 Results

We find that the model is stable and robust. Even using the simplest Euler integration scheme, we observe minimal oscillations affecting the smoothness of results. Figure 5.3 shows convergence in the solution of second or third order as $\Delta x \to 0$. This estimate is obtained by comparison of the result generated by each grid spacing with a high-resolution (641 node) result, since no absolute result is available. This order of convergence is at least as high as that reported for the very similar method in [8].

In the body of work concerning Lotka-Volterra equations, there are a vast range of parameter values in use, because there are so many varied but suitable examples of the type of competition that is described. We therefore select a conservatively representative set of parameters, chosen to demonstrate some of the interesting behaviours that this model is able to describe. We may choose a set of parameters that favour species 1, as shown in figure 5.4. In this case we see an increasing interface velocity in the initial stages, followed by a long steady phase where the interface velocity is approximately constant (figure 5.5). As we approach the annihilation of species 2, the interface velocity increases again (figure 5.6). This is due to the low mass of species 2 affecting its ability to grow. The movement of the interface is given in figure 5.7.

We then investigate alternative parameter choices. We may restrict the growth of species 1 by lowering its carrying capacity, k_1. We observe that in this scenario neither species is dominant, even though all the competition and diffusion characteristics are unchanged. This scenario is shown in figure 5.8.

Alternatively, we may adjust the diffusion characteristics of the system. By allowing species 2 to diffuse at a higher rate, we observe that species 2 is able to make territorial gains due to this alone (figure 5.9). However, as time goes on, the growth and competition characteristics become increasingly important. We see species 1 becoming more dominant over time, so that the interface velocity actually reverses direction. This is fascinating interface behaviour! Figure 5.10 shows the evolution of the system at $t = 12.3$, and figure 5.11 shows the movement of the interface with the direction reversal. These results give confidence that this model is likely to be able to satisfy the requirements of modelling a wide variety of competition systems. It is stable to a large choice of set-up parameters and is able to produce complex behaviours without problems.

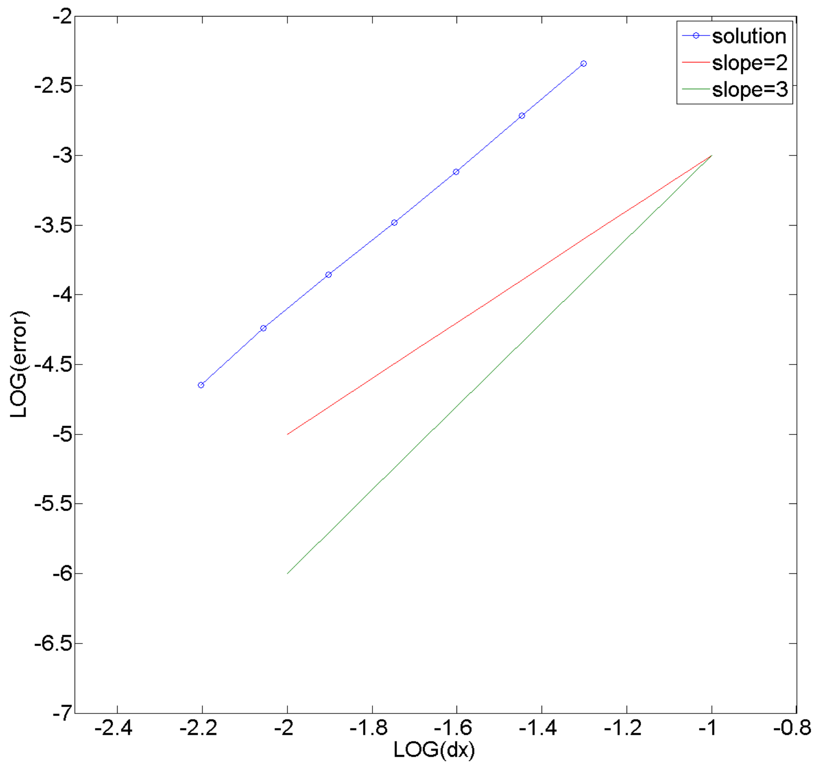


Fig. 5.3 Comparison of L^2 errors in the solution of algorithm 12. We observe an order of convergence of $p \approx 3$.

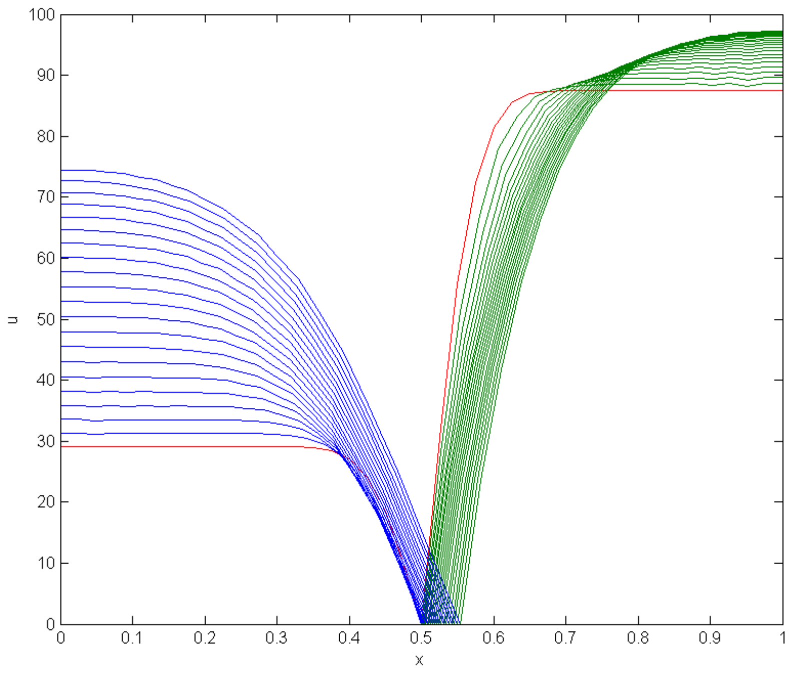


Fig. 5.4 Result of competition model at $t = 1.7$. Here we use $\delta_1 = \delta_2 = 0.01$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 17000 iterations and plot the results every $dt = 0.1$. We see the internal dynamics of the species driving population density and interface fluxes, and the position of the interface responding to those fluxes. The initial conditions are shown in red, with species 1 in blue and species 2 in green.

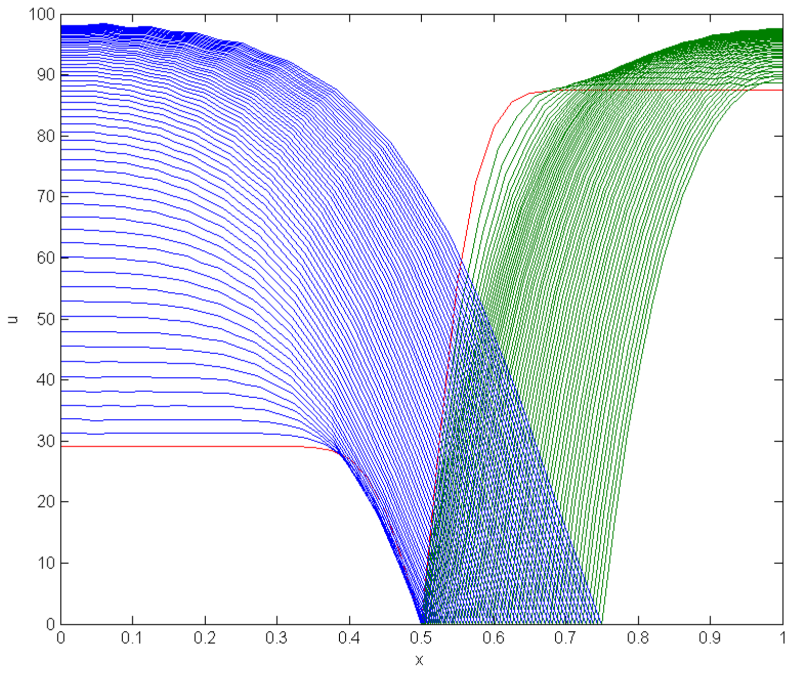


Fig. 5.5 Result of competition model at $t = 6.0$. Here we use $\delta_1 = \delta_2 = 0.01$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 60000 iterations and plot the results every $\Delta t = 0.1$. The interface continues to evolve and the masses of the species are now limited by the respective carrying capacities. The initial conditions are shown in red, with species 1 in blue and species 2 in green

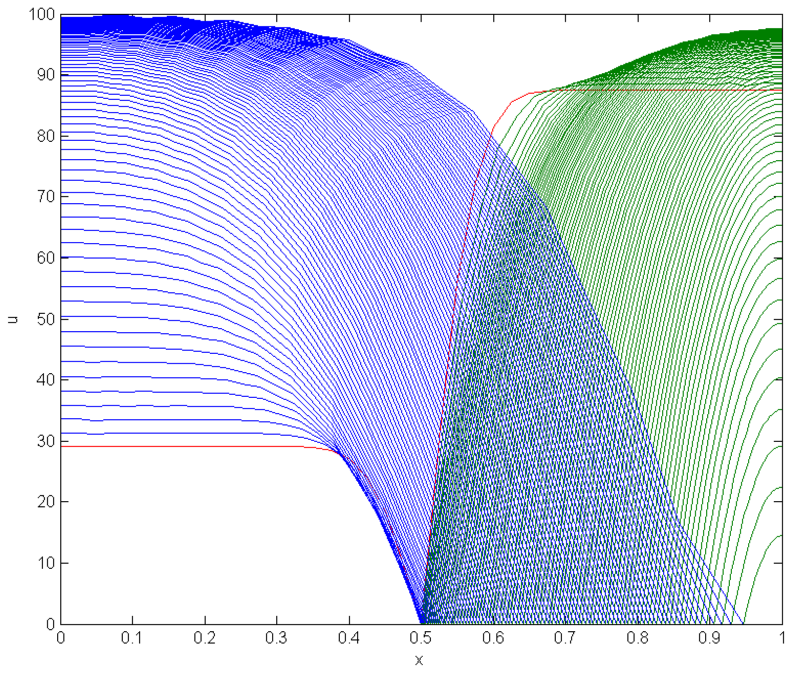


Fig. 5.6 Result of competition model at $t = 8.8$. Here we use $\delta_1 = \delta_2 = 0.01$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 122000 iterations and plot the results every $dt = 0.1$. Final step before node crossing occurs. We observe that whilst species 2 initially grew in mass, it will now be wiped out by competition with species 1.

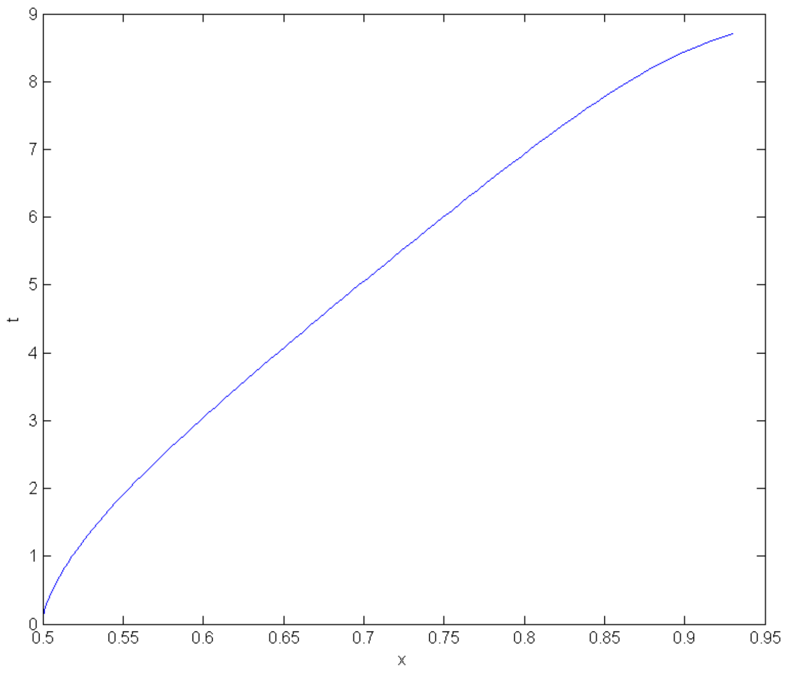


Fig. 5.7 Movement of interface position x_m for competition model with parameters $\delta_1 = \delta_2 = 0.01$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$. We see the interface increase in velocity after a slower initial phase where both species are experiencing population growth. We see the interface velocity accelerate as we approach an annihilation event.

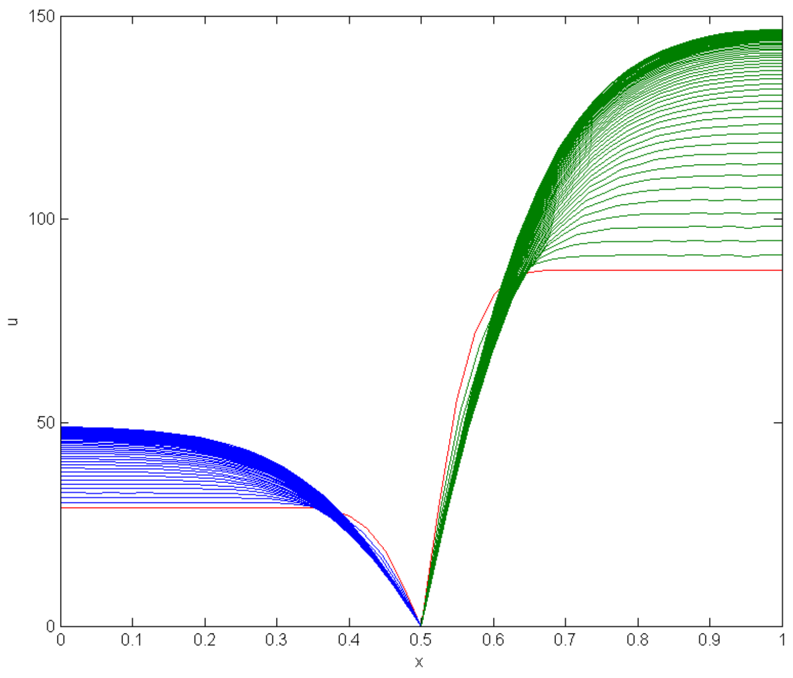


Fig. 5.8 Result of competition model at $t = 8$, considering the effect of altered carrying capacities. Here we use $\delta_1 = \delta_2 = 0.01$, $k_1 = 50, k_2 = 150$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 80000 iterations and plot the results every $dt = 0.1$. We see that with differently chosen carrying capacities we find the interface position is approximately steady and these two species are in balance.

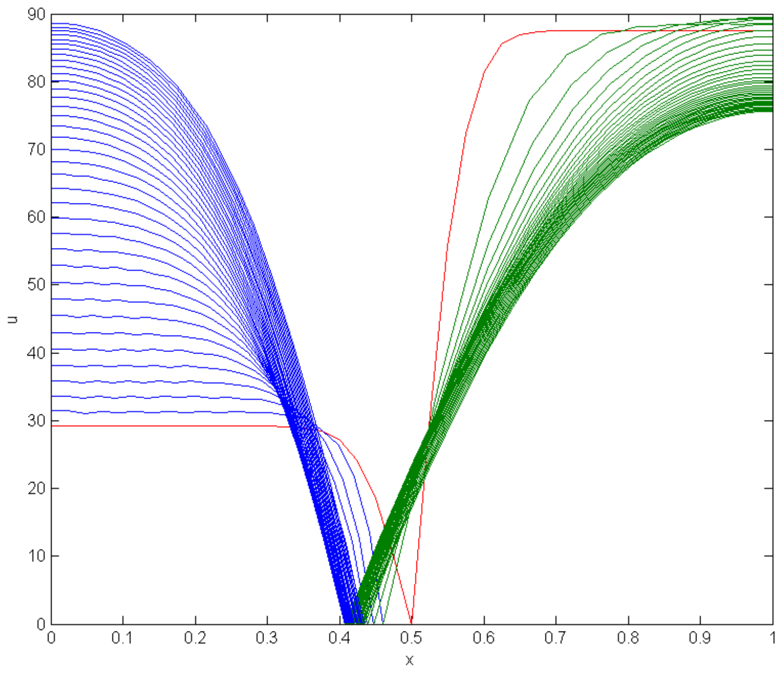


Fig. 5.9 Result of competition model at $t = 3.5$, considering the effect of an increased diffusion rate for species 2. Here we use $\delta_1 = 0.01, \delta_2 = 0.05$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 35000 iterations, and plot the results every $dt = 0.1$. We observe that species 2 is able to make initial territory gains due to its high diffusion rate, even though the competition rate is unaltered.

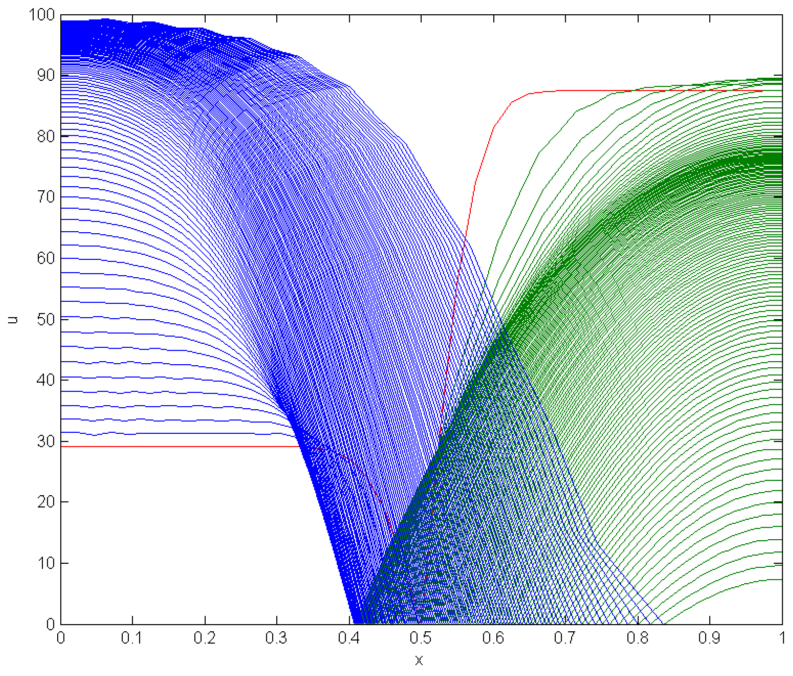


Fig. 5.10 Result of competition model at $t = 12.3$, considering the effect of an increased diffusion rate for species 2. Here we use $\delta_1 = 0.01, \delta_2 = 0.05$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 123000 iterations, and plot the results every $dt = 0.1$. We see that the initial diffusion-driven gains by species 2 are reversed, and that the overall growth characteristics are dominating so that species 1 is gaining territory.

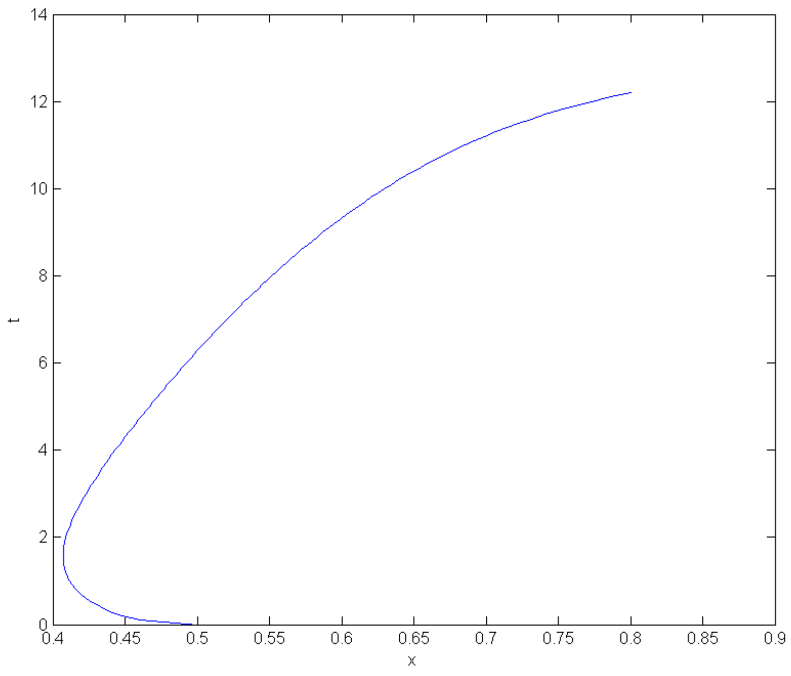


Fig. 5.11 Position of interface, x_m, showing interface movement for the competition model at up to $t = 12.3$, considering the effect of an increased diffusion rate for species 2 (*cf.* figure 5.7). Here we use $\delta_1 = 0.01, \delta_2 = 0.05$, $k_1 = k_2 = 100$, $r_1 = r_2 = 1$ and $\lambda = 3$. We run the model with a time step of $dt = 0.0001$ for 123000 iterations, and plot the results every $dt = 0.1$. Due to the growth characteristics we can see interesting temporal effects. Here the interface velocity has actually reversed directions as the system changes from diffusion dominated to growth dominated.

Chapter 6

Aggregation models

In [28] and [29], Grindrod presents a new consideration for population modelling. He points out that the derivation of the Lotka-Volterra competition models and similar single-species dispersion models rests on the assumption that the dispersal of individuals is due to random diffusive motion. This assumption is difficult to justify, since it is readily apparent that in the real world, individuals group together to improve their chances of survival, do not voluntarily overcrowd themselves to death, and deliberately avoid predators. Grindrod therefore introduces an element of deterministic behaviour to his model. In the Grindrod models, we assume that the random motion of individuals is biased by an optimal velocity v. This velocity is selected so as to increase an individual's expected rate of reproduction. On average, the population is dispersing in the ideal direction. Grindrod produces results obtained from this model as derived for a single species, and demonstrates that from an initially random seeding of individuals, clusters are formed. This work is of interest to us for three reasons. Firstly, the aggregation model has not previously been constructed in finite element form, on either a static or moving grid. Secondly, the model has not previously been implemented for a two species competitive environment. Thirdly, the assumptions made by Hilhorst in [31] require a zero population condition on the interface that is entirely driven by high competition rates. Whilst we would need a high competition interface in any multiphase scenario, having intelligent aggregation as a component of the model would seem to add somewhat more justification to the imposition of a zero population interface condition. We derive the MMFEM algorithms for single species in one and two dimensions. We also derive a fixed mesh finite element model for two species in two dimensions, for a conservative population case and a non-conservative population case. We then implement a selection of the algorithms and examine the results for the 2-D models.

6.1 Population clustering models for a single species

Following [28] and [29], we investigate a model for aggregation of individuals in a competitive environment. We have a new term E that describes the projected net rate of reproduction per individual at x, at time t. This is a constructed term comparable to the logistic term in a Lotka-Volterra equation. The crucial difference here compared to a standard population model is that we assume that individuals seek to maximise their chances of survival, and so will seek to move towards maximum E. We use a form for E that incorporates the assumption that survival chances depend only on population density, $u(\mathbf{x},t)$. Overcrowding or loneliness means a death rate higher than birth rate, and in between there is an optimum population density. In order to look particularly at clustering effects, we will not take births and deaths into account, *i.e.* we assume that births and deaths take place on a much longer time scale than clustering of individuals, so

$$E(u) = (u-a)(1-u), \quad a \in (0,1). \tag{6.1}$$

Our other parameters are,

δ=a nonnegative constant representing the random dispersion of individuals;
ε=a small positive constant.

We assume that individuals will move in a random walk biased by an optimal velocity, ν. A relationship between $E(u)$ and ν is constructed with ν as a local average of $\nabla E(u)$. For convenience we use a substitution $\nabla q = \nu$. The chosen form has the advantage of allowing us to impose zero flux boundary conditions on ν to prevent individuals moving across the boundary S of the domain Ω. As in [29] we define the relationship between $E(u)$ and q to be

$$E(u) = -\varepsilon\nabla^2 q + q. \tag{6.2}$$

Conceptually, q is a measure of the attractiveness of a location for an individual, taking into account not just survival chances at that point but also in the local area. The size of the area we define as local is important and is controlled by the parameter ε. The average velocity, $\mathbf{v}$, is the sum of the optimal velocity and a diffusive term,

$$\mathbf{v} = -\delta\frac{\nabla u}{u} + \nu. \tag{6.3}$$

We may now derive the PDE giving population density time dependence. We have

$$\begin{aligned}\frac{\partial u}{\partial t} &= -\nabla.(u\mathbf{v}) \\ &= -\nabla.(-\delta\nabla u + u\nu) \\ &= \delta\nabla^2 u - \nabla\cdot(u\nu) \\ &= \delta\nabla^2 u - \nabla\cdot(u\nabla q) \qquad (6.4)\end{aligned}$$

with boundary conditions

$$\hat{\mathbf{n}}.\nabla u = 0 \qquad \mathbf{x}\in\partial\Omega, \quad t\geq 0, \tag{6.5}$$

$$\hat{\mathbf{n}}.\nu = 0 \qquad \mathbf{x}\in\partial\Omega, \quad t\geq 0, \quad \nu = \nabla q. \tag{6.6}$$

6.1.1 1D population clustering model for a single species

We examine the 1D analogues of the equations described in section 6.1.

$$\frac{\partial u}{\partial t} = \delta\frac{\partial^2 u}{\partial x^2} - \frac{\partial}{\partial x}\left(u\frac{\partial q}{\partial x}\right) \tag{6.7}$$

$$E(u) = -\varepsilon\frac{\partial^2 q}{\partial x^2} + q \tag{6.8}$$

$$E(u) = (u-a)(1-u). \tag{6.9}$$

We have the boundary conditions

$$\frac{\partial u}{\partial x} = 0 \qquad x = A, B, \qquad t\geq 0, \tag{6.10}$$

$$\frac{\partial q}{\partial x} = 0 \qquad x = A, B, \qquad t\geq 0 \tag{6.11}$$

where A and B are fixed. We derive the moving-mesh, finite element model for this system. We consider the system with no births or deaths, so we have a true conservation of mass. Over the domain $x\in[A,B]$,

$$\int_A^B u dx = constant. \tag{6.12}$$

We define the distributed conservation principle, for a weight function w_i,

$$\int_A^B w_i u dx = constant \tag{6.13}$$

hence

$$\frac{d}{dt}\int_A^B w_i u \; dx = 0. \tag{6.14}$$

By the Reynolds Transport Theorem, we can say that

$$\int_A^B \frac{\partial}{\partial t}(w_i u)\, dx + \int_A^B \frac{\partial}{\partial x}(\dot{x} w_i u)\, dx = 0 \tag{6.15}$$

$$\int_A^B \left[w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + w_i \frac{\partial}{\partial x}(u\dot{x}) + u\dot{x}\frac{\partial w_i}{\partial x} \right] dx = 0. \tag{6.16}$$

Assuming the weight functions w_i move with the domain

$$\frac{\partial w_i}{\partial t} + \dot{x}\frac{\partial w_i}{\partial x} = 0 \tag{6.17}$$

hence

$$\int_A^B w_i \frac{\partial}{\partial x}(u\dot{x}) \; dx = -\int_A^B w_i \frac{\partial u}{\partial t} dx. \tag{6.18}$$

Substituting from equation (6.7),

$$\int_A^B w_i \frac{\partial}{\partial x}(u\dot{x}) \; dx = -\int_A^B w_i \left[\delta \frac{\partial^2 u}{\partial x^2} - \frac{\partial}{\partial x}\left(u \frac{\partial q}{\partial x} \right) \right] dx \tag{6.19}$$

and after integration by parts we obtain

$$[w_i \dot{x} u]_A^B - \int_A^B \frac{\partial w_i}{\partial x}(u\dot{x}) \; dx = \int_A^B \delta \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx - \int_A^B \frac{\partial w_i}{\partial x} u \frac{\partial q}{\partial x} dx + \left[w_i u \frac{\partial q}{\partial x} \right]_A^B - \left[\delta w_i \frac{\partial u}{\partial x} \right]_A^B \tag{6.20}$$

or, because we have fixed boundaries,

$$-\int_A^B \frac{\partial w_i}{\partial x}(u\dot{x}) \; dx = \int_A^B \delta \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x} \; dx - \int_A^B \frac{\partial w_i}{\partial x} u \frac{\partial q}{\partial x} dx + \left[w_i u \frac{\partial q}{\partial x} \right]_A^B - \left[\delta w_i \frac{\partial u}{\partial x} \right]_A^B . \tag{6.21}$$

We make the velocity potential substitution $\dot{x} = \frac{\partial \phi}{\partial x}$. We obtain

$$-\int_A^B u \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx = -\left[\delta w_i \frac{\partial u}{\partial x}\right]_A^B + \int_A^B \delta \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\, dx + \left[w_i u \frac{\partial q}{\partial x}\right]_A^B - \int_A^B u \frac{\partial q}{\partial x}\frac{\partial w_i}{\partial x} dx. \tag{6.22}$$

We note the presence of the zero flux boundary conditions (6.10), hence the two non-integral terms on the right hand side will be equal to zero. The equation to be solved for ϕ is then

$$-\int_A^B u \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx = \int_A^B \delta \frac{\partial w_i}{\partial x}\frac{\partial u}{\partial x}\, dx - \int_A^B u \frac{\partial q}{\partial x}\frac{\partial w_i}{\partial x} dx. \tag{6.23}$$

This expression requires known q. We return to the definition (6.8). We write this in weak form,

$$\int_A^B w_i E(u)\, dx = -\varepsilon \int_A^B w_i \frac{\partial^2 q}{\partial x^2} dx + \int_A^B w_i q\, dx. \tag{6.24}$$

Integrating by parts on the right hand side, we obtain

$$\int_A^B w_i E(u)\, dx = -\varepsilon \left[w_i \frac{\partial q}{\partial x}\right]_A^B + \varepsilon \int_A^B \frac{\partial w_i}{\partial x}\frac{\partial q}{\partial x} dx + \int_A^B w_i q\, dx. \tag{6.25}$$

Noting again the zero flux boundary condition on q, we may simplify this to

$$\int_A^B w_i E(u)\, dx = \varepsilon \int_A^B \frac{\partial w_i}{\partial x}\frac{\partial q}{\partial x} dx + \int_A^B w_i q\, dx \tag{6.26}$$

which may be solved for q.

6.1.2 Construction of the finite element form

We use the unmodified piecewise linear basis functions $w_i = W_i$, since we have only Neumann conditions to consider. We define our finite element variables U, Q and E in terms of the piecewise linear approximations $U = \sum_j W_j U_j$, $Q = \sum_j W_j Q_j$, $E = \sum_j W_j E_j$. Note that although E is itself a nonlinear function of U, we simply calculate discrete values of $E(U)$ at nodes only and accept a linear approximation between nodes. We make this remark since elsewhere in the thesis, nonlinear terms are treated more robustly, but we do not expect this simpler approximation to be of significance here. After substituting in these approximations, equation (6.26) becomes

$$\sum_{j=0}^{N+1} \left[\int_A^B W_i W_j dx\right] E_j = \sum_{j=0}^{N+1} \left[\varepsilon \int_A^B \frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x} dx\right] Q_j + \sum_{j=0}^{N+1} \left[\int_A^B W_i W_j dx\right] Q_j. \tag{6.27}$$

In terms of the standard mass and stiffness matrices this is

$$\varepsilon K\underline{Q} + M\underline{Q} = M\underline{E} \tag{6.28}$$

for vectors $\underline{Q}$ containing the values of Q_j, and $\underline{E}$ containing the values of E_j. To solve this we first obtain the values of $E(u)$ from equation (6.9). We can now obtain $\underline{Q}$ from the steady state system

$$\underline{Q} = (\varepsilon K + M)^{-1} M\underline{E}. \tag{6.29}$$

We now require ϕ, which can be obtain from equation (6.23). We make the same piecewise linear approximations and, after substitution, obtain

$$-\sum_{j=0}^{N+1}\left[\int_A^B U\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]\Phi_j = \sum_{j=0}^{N+1}\left[\int_A^B \delta\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}\,dx\right]U_j - \sum_{j=0}^{N+1}\left[\int_A^B U\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]Q_j. \tag{6.30}$$

We solve for the vector $\Phi = \{\Phi_j\}$ using the matrix form

$$K(\underline{U})\underline{\Phi} = \delta K\underline{U} - K(\underline{U})\underline{Q} \tag{6.31}$$

with $K(\underline{U})$ analogous to the stiffness matrix, and given by

$$K(\underline{U})_{ij} = \int_{x_{i-1}}^{x_{i+1}} U\frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x}\,dx. \tag{6.32}$$

Once $\underline{\Phi}$ is recovered, we obtain $\dot{X}$ and U in the manner now standard in this thesis. Briefly, we use the weak form of the definition for the velocity potential $\dot{x} = \partial\phi/\partial x$, which is

$$\int_A^B w_i\dot{x}\,dx = \int_A^B w_i\frac{\partial\phi}{\partial x}dx. \tag{6.33}$$

We select the weight functions $w_i = W_i$ and using the piecewise linear approximations $\dot{X} = \sum_j W_j\dot{X}_j$ and $\Phi = \sum_j W_j\Phi_j$ we make substitutions to obtain

$$\sum_{j=0}^{N+1}\left[\int_A^B W_iW_j\quad dx\right]\dot{X}_j = \sum_{j=0}^{N+1}\left[\int_A^B W_i\frac{\partial W_j}{\partial x}dx\right]\Phi_j. \tag{6.34}$$

In matrix form this is

$$M\underline{\dot{X}} = B\underline{\Phi}_j \tag{6.35}$$

where $\underline{\dot{X}}$ and $\underline{\Phi}_j$ are the vectors containing the unknown velocities $\dot{X}_j$ and the known Φ_j, and M and B are the symmetric mass matrix and an asymmetric matrix respectively, as defined in section 3.1.2.

In this way, (6.35) can be solved to obtain the $\dot{X}_j$ values. We then perform the time integration step using any chosen scheme. Once the grid position has been recalculated, the basis functions are likewise moved and the matrices defined by them are recalculated.

We recover $\underline{U}$ from the conservation principle (6.13)

$$\int_A^B w_i u \; dx = constant. \tag{6.36}$$

Making the substitution for the approximation $U = \sum_j W_j U_j$ we obtain

$$\sum_{j=0}^{N+1} \left[\int_A^B W_i W_j dx \right] U_j = c_i \tag{6.37}$$

where c_i is given by

$$c_i(x) = \int_A^B W_i(x,0) U(x,0) \; dx \tag{6.38}$$

for the initial data $U(x,0)$. Then (6.37) is equivalent to the mass matrix system

$$M\underline{U} = \underline{c} \tag{6.39}$$

where $\underline{c}$ is the vector containing the c_i, and M is the mass matrix for the new time step. We may then solve (6.39) for U using the updated M. Whilst we do not implement this model in this form, preferring to focus on the two dimensional model, the derivation is useful for a later two phase version with an interface condition.

6.1.3 2D population clustering model for a single species

For the 2D model, we remind ourselves of the driving PDE system. This is

$$\frac{\partial u}{\partial t} = \delta \nabla^2 u - \nabla . (u \nabla q) \tag{6.40}$$

$$E(u) = -\varepsilon \nabla^2 q + q \tag{6.41}$$

$$E(u) = (u-a)(1-u) \tag{6.42}$$

with boundary conditions on the fixed boundary S

$$\hat{\mathbf{n}}.\nabla u = 0 \qquad \mathbf{x} \in S, t \geq 0, \tag{6.43}$$

$$\hat{\mathbf{n}}.v = 0 \qquad \mathbf{x} \in S, t \geq 0. \tag{6.44}$$

In this single species system we consider the case where we have no births or deaths, so that clustering effects are most apparent even if transient. We therefore have a true conservation of mass. Over the domain $\mathbf{x} \in \Omega$ the conservation principle is

$$\int_\Omega u d\Omega = constant. \tag{6.45}$$

We define the distributed form, for a weight function w_i,

$$\int_\Omega w_i u \; d\Omega = c_i \tag{6.46}$$

where the constant c_i is determined by the choice of w_i. Hence

$$\frac{d}{dt}\int_\Omega w_i u \;\; d\Omega = 0. \tag{6.47}$$

Using the Reynolds Transport Theorem, we can write

$$\int_\Omega \frac{\partial}{\partial t}(w_i u) \;\; d\Omega + \int_\Omega \nabla x \cdot (\dot{\mathbf{x}} w_i u) \;\; d\Omega = 0 \tag{6.48}$$

leading to

$$\int_\Omega \left[w_i \frac{\partial u}{\partial t} + u \frac{\partial w_i}{\partial t} + w_i \nabla \cdot (u\dot{\mathbf{x}}) + u\dot{\mathbf{x}} \cdot \nabla w_i \right] d\Omega = 0. \tag{6.49}$$

We assume that the weight functions w_i move with the domain, which gives

$$\frac{\partial w_i}{\partial t} + \dot{\mathbf{x}} \cdot \nabla w_i = 0 \tag{6.50}$$

hence (6.49) is

$$\int_\Omega w_i \nabla \cdot (u\dot{\mathbf{x}}) \;\; d\Omega = -\int_\Omega w_i \frac{\partial u}{\partial t} \; d\Omega. \tag{6.51}$$

We make a substitution from the driving PDE (6.40) to obtain

$$\int_\Omega w_i \nabla \cdot (u\dot{\mathbf{x}}) \; d\Omega = -\int_\Omega w_i(\delta \nabla^2 u - \nabla \cdot (u\nabla q)) \; d\Omega \tag{6.52}$$

and after integration by parts we obtain

$$\begin{aligned}\int_S w_i \dot{\mathbf{x}} u \cdot \hat{\mathbf{n}} \; dS - \int_\Omega \nabla w_i \cdot (u\dot{\mathbf{x}}) \; d\Omega = \int_\Omega \delta \nabla w_i \cdot \nabla u \;\; d\Omega - \int_\Omega \nabla w_i \cdot u \nabla q \; d\Omega \\ + \int_S w_i u \nabla q \cdot \hat{\mathbf{n}} \; dS - \int_S \delta w_i \nabla u \cdot \hat{\mathbf{n}} \; dS\end{aligned} \tag{6.53}$$

or, because we have fixed boundaries,

$$\begin{aligned}-\int_\Omega \nabla w_i \cdot (u\dot{\mathbf{x}}) \;\; d\Omega = \int_\Omega \delta \nabla w_i \cdot \nabla u \;\; d\Omega - \int_\omega \nabla w_i \cdot u \nabla q \; d\Omega \\ + \int_S w_i u \nabla q \cdot \hat{\mathbf{n}} \; dS - \int_S \delta w_i \nabla u \cdot \hat{\mathbf{n}} \; dS.\end{aligned} \tag{6.54}$$

We make the velocity potential substitution $\dot{\mathbf{x}} = \nabla \phi$. We obtain

$$\begin{aligned}-\int_\Omega u \nabla w_i \cdot \nabla \phi \; d\Omega = -\int_S \delta w_i \nabla u \cdot \hat{\mathbf{n}} \; dS + \int_\Omega \delta \nabla w_i \cdot \nabla u \;\; d\Omega \\ + \int_S w_i u \nabla q \cdot \hat{\mathbf{n}} \; dS - \int_\Omega u \nabla q \cdot \nabla w_i \; d\Omega\end{aligned} \tag{6.55}$$

subject to zero flux conditions on the boundary (6.43), so the two boundary terms on the right hand side will be equal to zero. We obtain the equation to be solved for ϕ,

$$-\int_\Omega u \nabla w_i \cdot \nabla \phi \; d\Omega = \int_\Omega \delta \nabla w_i \cdot \nabla u \;\; d\Omega - \int_\Omega u \nabla q \cdot \nabla w_i \; d\Omega. \tag{6.56}$$

Before this can be solved we need a value for q and this can be obtained, in a similar way to the 1D case, from the steady state equation (6.41),

$$E(u) = -\varepsilon \nabla^2 q + q. \tag{6.57}$$

Writing this in weak form, we have

$$\int_\Omega w_i E(u)\ d\Omega = -\varepsilon \int_\Omega w_i \nabla^2 q\ d\Omega + \int_\Omega w_i q\ d\Omega. \tag{6.58}$$

We integrate the right hand side by parts to obtain

$$\int_\Omega w_i E(u)\ d\Omega = -\varepsilon \int_S w_i \nabla q \cdot \hat{\mathbf{n}}\ dS + \varepsilon \int_\Omega \nabla w_i \cdot \nabla q\ d\Omega + \int_\Omega w_i q\ d\Omega \tag{6.59}$$

since $\nabla q \cdot \hat{\mathbf{n}} = 0$ on S, the boundary term is equal to zero. We therefore have

$$\int_\Omega w_i E(u)\ d\Omega = \varepsilon \int_\Omega \nabla w_i \cdot \nabla q\ d\Omega + \int_\Omega w_i q\ d\Omega. \tag{6.60}$$

This allows us to obtain q once $E(u)$ is known. We may obtain the values of $E(u)$ from equation (6.42).

6.1.4 Construction of the finite element form

In order to solve equations (6.56) and (6.60), we use the finite element method. We use the unmodified two dimensional triangular weight functions $w_i = W_i$ described in Chapter 3 (3.1.3), since we have no Dirichlet conditions to impose. We define our finite element variables Q and E in terms of the same basis functions, using the approximations $Q(\mathbf{x},t) = \sum_j W_j(\mathbf{x})Q_j(t)$ and $E(\mathbf{x},t) = \sum_j W_j(\mathbf{x})E_j(t)$. After making the substitutions for these approximations, (6.60) becomes

$$\sum_{j=1}^{N} \left[\int_\Omega W_i W_j\ d\Omega\right] E_j = \sum_{j=1}^{N} \left[\varepsilon \int_\Omega \nabla W_i \cdot \nabla W_j\ d\Omega\right] Q_j + \sum_{j=1}^{N} \left[\int_{\Omega_i} W_i W_j\ d\Omega\right] Q_j. \tag{6.61}$$

In terms of mass and stiffness matrices M and K this is written as

$$\varepsilon K \underline{Q} + M \underline{Q} = M \underline{E} \tag{6.62}$$

for vectors $\underline{Q}$ containing Q_i and $\underline{E}$ containing E_i. Rearranging, we obtain $\underline{Q}$ from the steady state system

$$\underline{Q} = (\varepsilon K + M)^{-1} M \underline{E} \tag{6.63}$$

where $\underline{E}$ is given by the definition (6.42). Similarly, we make substitutions into equation (6.56), defining our variables in terms of piecewise linear approximations based on the W_i. In addition to the approximation $Q(\mathbf{x},t) = \sum_j W_j(\mathbf{x})Q_j(t)$ already given, we require

$U(\mathbf{x},t) = \sum_j W_j(\mathbf{x})U_j(t)$ as the approximation for u, and $\Phi(\mathbf{x},t) = \sum_j W_j(\mathbf{x})\Phi_j(t)$ as the approximation for ϕ. Equation (6.56) becomes

$$-\sum_{j=1}^{N}\left[\int_\Omega \nabla W_i \cdot \nabla W_j U \; d\Omega\right]\Phi_j = \sum_{j=1}^{N}\left[\delta\int_\Omega \nabla W_i . \nabla W_j \; d\Omega\right]U_j + \sum_{j=1}^{N}\left[\int_\Omega \nabla W_i \cdot (U\nabla W_j) d\Omega\right]Q_j \tag{6.64}$$

or in matrix form

$$K(\underline{U})\underline{\Phi} = \delta K\underline{U} + K(\underline{U})\underline{Q} \tag{6.65}$$

where $K(\underline{U})$ is the weighted stiffness matrix given by (3.69). Equation (6.65) is now sufficient to recover Φ. We then calculate $\dot{\mathbf{X}}$, perform the time integration and lastly recover $\underline{U}$ from the conservation of mass equation. This process is described fully in Chapter 3 but briefly, the definition of ϕ is

$$\dot{\mathbf{x}} = \nabla\phi \tag{6.66}$$

for which a weak form is

$$\int_\Omega w_i \dot{\mathbf{x}} \; d\Omega = \int_\Omega w_i \nabla\phi \; d\Omega. \tag{6.67}$$

Using again the piecewise linear approximations $w_i = W_i(\mathbf{x})$, $\dot{\mathbf{X}}(\mathbf{x},t) = \sum_{j=1}^N \dot{\mathbf{X}}_j(t)W_j(\mathbf{x})$ and $\nabla\Phi(\mathbf{x},t) = \sum_{j=1}^N \Phi_j(t)\nabla W_j(\mathbf{x})$ we obtain

$$\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \;\; d\Omega\right]\dot{\mathbf{X}}_j = \sum_{j=1}^{N}\left[\int_\Omega W_i \nabla W_j \; d\Omega\right]\Phi_j. \tag{6.68}$$

Hence in matrix form, (6.68) can be solved for $\dot{\mathbf{X}}$ using

$$M\underline{\dot{\mathbf{X}}} = B\underline{\Phi} \tag{6.69}$$

where $\underline{\dot{\mathbf{X}}} = \{\dot{\mathbf{X}}_i\}$, M is the symmetric mass matrix, and B is an asymmetric matrix with elements $B_{ij} = \int_\Omega W_i \nabla W_j \; d\Omega$. Having found $\dot{\mathbf{X}}$, the nodes are repositioned using the forward Euler scheme. We recover $\underline{U}$ using our distributed mass conservation principle (6.13). Using the piecewise linear W_i that together form a partition of unity, equation (6.13) is, for each node i,

$$c_i = \int_\Omega W_i U \; d\Omega.$$

Using the piecewise linear approximation $U(\mathbf{x},t) = \sum_j U_j(t)W_j(\mathbf{x})$ we obtain

$$\sum_{j=1}^{N} \left[\int_{\Omega} W_i W_j \; d\Omega \right] U_j = c_i \tag{6.70}$$

which is equivalent to the mass matrix system

$$M\underline{U} = \underline{c}_i. \tag{6.71}$$

This equation is used to calculate the initial (and constant) values of c_i, using the initial values of U_j and $\mathbf{X}_j$. After repositioning the nodes we may recover $U_j(t)$ from the mass matrix system (6.71).

Algorithm 13

The finite element solution of the single species aggregation model defined by equations (6.40), (6.41) and (6.42) on the moving mesh in 2-D therefore consists of the following steps. We obtain the constant values of c_i from (6.71) calculated at $t = 0$, and then for each time step:

1. Calculate the reproductive potential by solving equation (6.42) for $E(u)$;
2. Find the values of Q by solving equation (6.63);
3. Find the velocity potential by solving equation (6.65) for the $\Phi_j(t)$ values;
4. Find the node velocity by solving equation (6.69) for the $\dot{\mathbf{X}}(t)$ values;
5. Generate the co-ordinate system at the next time-step $t + dt$ by solving (3.18) using Euler's approximation;
6. Find the solution $U(t + dt)$ by solving the conservation equation (6.71).

6.1.5 Results

In common with [29], we use a random seeding to provide the initial conditions for the model. The random seeding is selected from a normal distribution with a mean of 0.3 and a standard deviation of 0.01. The model is stable and robust. We are able to run the model sometimes to a blow up and sometimes to a solution where population growth and decline become approximately balanced. The outcome depends on the initial values of u, and also on the parameters δ and ε. We are familiar from the diffusion models with the parameter δ and its effects. As the parameter controlling the rate of diffusion, it has a smoothing effect when large. The parameter ε is less familiar. From the definition contained within (6.41), it is apparent that the intention for ε is to define the scale of the clusters that need to form. We find that we can see this scaling effect in the results, with the number and size of clusters reduced as ε increases. We notice that the patterns formed in u (where approximately balanced states are reached) tend to approximate eigenmodes of the Laplacian ∇^2, and that these are consistent for given parameters. This is of relevance and interest to the work of Grindrod [29], in which he approaches his models from a patterns and waves perspective. However, with our focus elsewhere we make this remark in passing only.

An example solution is given in figure 6.1, for parameters $\varepsilon = 0.005$ and $\delta = 0.01$. This setup produces four clusters from the initially random seeding. The clusters are under development in this snapshot and we have not yet reached an approximately balanced population. We can see the difference that an alternative choice of ε produces in figure (6.2). With $\varepsilon = 0.001$ and $\delta = 0.01$ we observe six clusters forming. If the model is allowed to continue, we reach an approximately steady-state solution (figure 6.3). We observe that the reproductive potential $E(u)$ is very low in the centre of the clusters, due to overcrowding. This low $E(u)$ tends to disperse individuals away from the centre of the cluster. However, the population densities at the edges of the cluster are low enough to draw individuals in, and so eventually the two effects become balanced and the approximately balanced solution is observed.

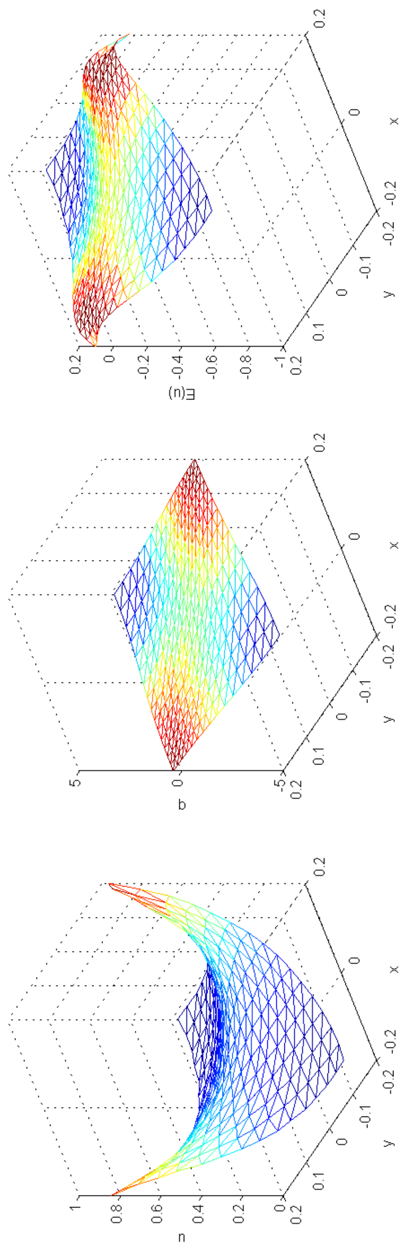


Fig. 6.1 A solution after 350 iterations at $t = 0.35$ of the 2D population equations, with $\varepsilon = 0.005$ and $\delta = 0.01$. This solution has not yet reached a balance, but is approximating the 4th eigenmode of the Laplacian.

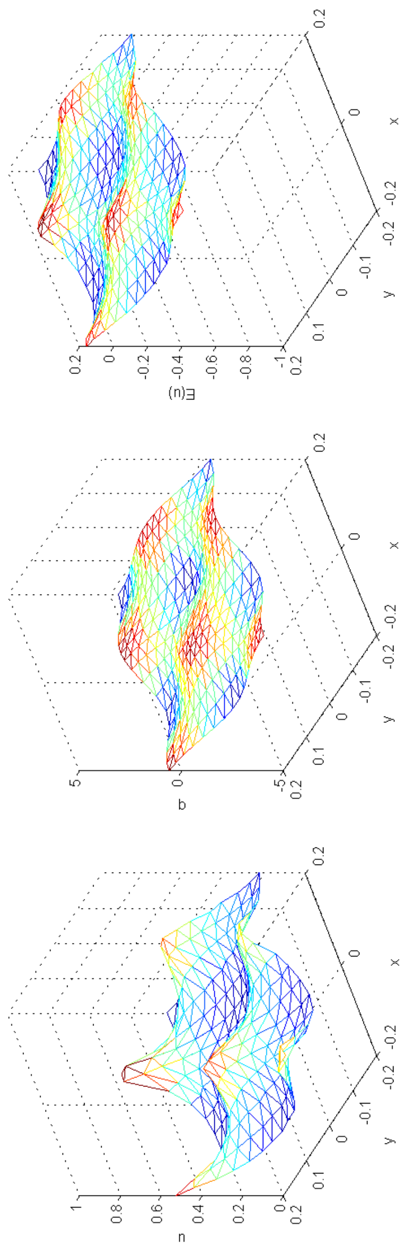


Fig. 6.2 A solution after 10 iterations at $t = 0.01$ of the 2D population equations, with $\varepsilon = 0.001$ and $\delta = 0.01$. This solution has not yet reached a balance, but is approximating the 20th eigenmode of the Laplacian.

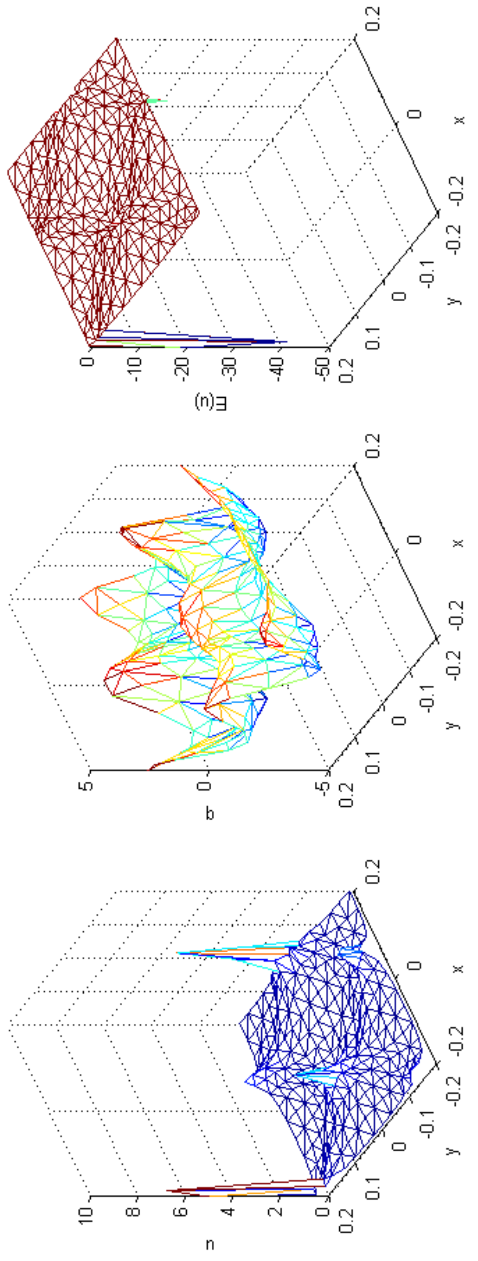


Fig. 6.3 An approximately balanced solution of the 2D population equations, with $\varepsilon = 0.001$ and $\delta = 0.01$. plotting (from left to right) u, q and $E(u)$. Whilst there is overcrowding in the centres of the clusters, giving a dramatically negative $E(u)$, the rate of population decline resulting from that is balanced by the attraction of the cluster to individuals nearby. These two effects mean that the shape of the solution does not evolve further, with only minor local effects observed.

6.2 Population clustering models for two competitive species

We now consider reaction-aggregation-diffusion models with two species. We consider the case where the species share a domain, so that we may examine clustering into species specific groups and the resulting claiming of territory. This may be of use in informing suitable starting conditions for a two phase model of competition.

A finite element formulation for the fixed mesh case

We begin with the Lotka-Volterra competition equations of Chapter 2, section 2.2, for two competing species with population densities u_1 and u_2. Following [29], we also have an aggregation term so that the driving PDEs are

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla \cdot (u_1 v_1) + r u_1 E_1 \tag{6.72}$$

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu \nabla \cdot (u_2 v_2) + r u_2 E_2 \tag{6.73}$$

where E_1 and E_2 are net reproduction rates for each species. The r is a coefficient controlling the timescale upon which births and deaths operate, so that we may examine potentially transient clustering effects by setting $r = 0$ or alternatively look at the longer term evolution of the system by setting $r = 1$. The v_1 and v_2 are optimal velocities for individuals' movements, defined (following [29]) in terms of E. For simplicity of analysis we define quantities q_1 and q_2 such that

$$\nabla q_1 = v_1 \tag{6.74}$$

$$\nabla q_2 = v_2. \tag{6.75}$$

The relationship between velocity and survivability is defined indirectly in terms of q_1 and q_2 as

$$\varepsilon_1 \nabla^2 q_1 + q_1 = E_1 \tag{6.76}$$

$$\varepsilon_2 \nabla^2 q_2 + q_2 = E_2. \tag{6.77}$$

Overall movement is therefore controlled by a combination of a diffusive or random walk element, with a maximum survivability bias. Following [29] we will examine a system where inter and intra-species competition are important, but where low population densities do not in themselves reduce survivability, in contrast with the single-species models. We define the logistic equations

$$E_1 = A - au_1 - bu_2 \tag{6.78}$$

$$E_2 = B - a^*u_1 - b^*u_2. \tag{6.79}$$

The parameters a, b, a^* and b^* replace the rate parameters K, k and κ of chapter 5. There is no functional difference; the convention of [29] is merely to select a form for the parameters that gives optimal mathematical clarity rather then optimal ecological relatability.

The driving PDEs we work with are then, with the velocity defined in terms of q,

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla \cdot (u_1 \nabla q_1) + ru_1E_1 \tag{6.80}$$

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu \nabla \cdot (u_2 \nabla q_2) + ru_2E_2 \tag{6.81}$$

with boundary conditions

$$\hat{\mathbf{n}}.\nabla u = 0 \qquad \mathbf{x} \in \partial\Omega, t \geq 0, \tag{6.82}$$

$$\hat{\mathbf{n}}.\nabla q = 0 \qquad \mathbf{x} \in \partial\Omega, t \geq 0. \tag{6.83}$$

6.2.1 The conservative population case

In order to understand the nature of the behaviour we first derive a finite element model on a static mesh. In order to understand the clustering effects, we consider the case where births and deaths happen on a much longer timescale than individual movements. For this conservation of population we simply set $r = 0$, so that the driving PDEs (6.80) and (6.81) become

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla \cdot (u_1 \nabla q_1) \tag{6.84}$$

and

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu \nabla \cdot (u_2 \nabla q_2). \tag{6.85}$$

In common with all our population models we have zero flux boundary conditions, *i.e.* $\hat{\mathbf{n}} \cdot \nabla u = 0$ and $\hat{\mathbf{n}} \cdot \nabla q = 0$. We consider the fixed domain Ω with boundary S. Combining

(6.78) and (6.76) we obtain

$$A - au_1 - bu_2 = \varepsilon_1 \nabla^2 q_1 + q_1. \tag{6.86}$$

We write this in weak form, using a weight function w_i to give

$$\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega = \int_\Omega w_i \varepsilon_1 \nabla^2 q_1 \; d\Omega + \int_\Omega w_i q_1 \; d\Omega. \tag{6.87}$$

After integration by parts on the right-hand side we obtain

$$\begin{aligned} &\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega \\ &\qquad = \varepsilon_1 \int_S w_i \nabla q_1 \cdot \hat{\mathbf{n}} \; dS - \varepsilon_1 \int_\Omega \nabla w_i \cdot \nabla q_1 \; d\Omega + \int_\Omega w_i q_1 \; d\Omega. \end{aligned} \tag{6.88}$$

The first term on the right hand side is equal to zero at the domain boundaries due to the zero flux boundary conditions, and so can be removed here to give

$$\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega = -\varepsilon_1 \int_\Omega \nabla w_i \cdot \nabla q_1 \; d\Omega + \int_\Omega w_i q_1 \; d\Omega. \tag{6.89}$$

Equation (6.89) will give us q_1 in terms of u_1 and u_2. In exactly the same way, from (6.79) and (6.77) we obtain

$$\int_\Omega w_i B \; d\Omega - \int_\Omega w_i a^* u_1 \; d\Omega - \int_\Omega w_i b^* u_2 \; d\Omega = -\varepsilon_2 \mu \int_\Omega \nabla w_i \cdot \nabla q_2 \; d\Omega + \mu \int_\Omega w_i q_2 \; d\Omega \tag{6.90}$$

which gives us q_2 in terms of u_1 and u_2. We may now turn our attention to finding a suitable form of the PDEs defining $\partial u_1/\partial t$ (6.84) and $\partial u_2/\partial t$ (6.85). Equation (6.84) is

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla \cdot (u_1 \nabla q_1)$$

or

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - u_1 \nabla^2 q_1 - \nabla q_1 \cdot \nabla u_1. \tag{6.91}$$

We rewrite this in weak form to obtain

$$\int_\Omega w_i \frac{\partial u_1}{\partial t} \; d\Omega = \delta_1 \int_\Omega w_i \nabla^2 u_1 \; d\Omega - \int_\Omega w_i u_1 \nabla^2 q_1 \; d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1 \; d\Omega \tag{6.92}$$

and after integration by parts we have

$$\begin{aligned}\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = \delta_1 \int_S w_i \nabla u_1 \cdot \hat{\mathbf{n}} dS - \delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega \\ - \int_\Omega w_i u_1 \nabla^2 q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega\end{aligned} \tag{6.93}$$

the first term on the right hand side of which is equal to zero at the domain boundaries due to the zero flux boundary conditions (6.82), leaving

$$\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega - \int_\Omega w_i u_1 \nabla^2 q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega. \tag{6.94}$$

Turning our attention to the second term on the right hand side, again we integrate by parts and obtain

$$\begin{aligned}\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega - \int_S w_i u_1 \nabla q_1 \cdot \hat{\mathbf{n}}\ dS \\ + \int_\Omega \nabla(w_i u_1) \cdot \nabla q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega.\end{aligned} \tag{6.95}$$

Again the boundary integral is zero from the boundary condition (6.82). Therefore we can reduce this to

$$\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega + \int_\Omega \nabla(w_i u_1) \cdot \nabla q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega. \tag{6.96}$$

By the product rule, this becomes

$$\begin{aligned}\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega + \int_\Omega u_1 \nabla w_i \cdot \nabla q\ d\Omega \\ + \int_\Omega w_i \nabla u_1 \cdot \nabla q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega\end{aligned} \tag{6.97}$$

which simplifies to

$$\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega + \int_\Omega u_1 \nabla w_i \cdot \nabla q_1\ d\Omega. \tag{6.98}$$

Equation (6.98) determines $\frac{\partial u_1}{\partial t}$ in terms of q_1 and u_1, and is ready for finite element substitutions to be made. We follow the same process for $\partial u_2/\partial t$ as we did for $\partial u_1/\partial t$. Equation

(6.85), the driving PDE for $\partial u_2/\partial t$, is

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu \nabla \cdot (u_2 \nabla q_2) \tag{6.99}$$

or

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu u_2 \nabla^2 q_2 - \mu \nabla q_2 \cdot \nabla u_2. \tag{6.100}$$

In weak form we rewrite this as

$$\int_\Omega w_i \frac{\partial u_2}{\partial t}\, d\Omega = \delta_2 \int_\Omega w_i \nabla^2 u_2 \, d\Omega - \mu \int_\Omega w_i u_2 \nabla^2 q_2 \, d\Omega - \mu \int_\Omega w_i \nabla q_2 \cdot \nabla u_2 \, d\Omega. \tag{6.101}$$

The process that follows is identical to the u_1 case, as set out from equations (6.92) to (6.98), with only the choice of parameters by way of a difference. We arrive at the weak form

$$\int_\Omega w_i \frac{\partial u_2}{\partial t}\, d\Omega = -\delta_2 \int_\Omega \nabla w_i \cdot \nabla u_2 \, d\Omega + \mu \int_\Omega u_2 \nabla w_i \cdot \nabla q_2 \, d\Omega. \tag{6.102}$$

Equation (6.102) determines $\frac{\partial u_2}{\partial t}$ in terms of q_1 and u_1, and is ready for finite element substitutions to be made.

6.2.2 Construction of the finite element form

Since we have Neumann conditions on the boundaries, we select the standard two dimension triangular weight functions W_i of (3.1.3). We use these functions as our weight functions and also define approximations to our variables using these functions as basis functions. Our approximations are

$$U_1(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) U_{1_j}(t) \tag{6.103}$$

$$U_2(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) U_{2_j}(t) \tag{6.104}$$

$$Q_1(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) Q_{1_j}(t) \tag{6.105}$$

and

$$Q_2(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) Q_{2_j}(t). \tag{6.106}$$

We substitute these approximations into (6.89) and obtain

$$\int_\Omega W_i A \; d\Omega - a\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]U_{1_j} - b\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]U_{2_j}$$
$$= -\varepsilon_1\sum_{j=1}^{N}\left[\int_\Omega \nabla W_i\cdot\nabla W_j \; d\Omega\right]Q_{1_j} + \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]Q_{1_j}. \qquad (6.107)$$

We may write (6.107) in terms of our mass and stiffness matrices M and K to obtain

$$M\underline{A} - aM\underline{U}_1 - bM\underline{U}_2 = -\varepsilon_1 K\underline{Q}_1 + M\underline{Q}_1. \qquad (6.108)$$

Here $\underline{A}$ is a vector with all entries equal to A. We may rewrite this in terms of $\underline{Q}_1$

$$\underline{Q}_1 = (-\varepsilon_1 K + M)^{-1}M(\underline{A} - a\underline{U}_1 - b\underline{U}_2). \qquad (6.109)$$

In exactly the same manner, we substitute the approximations (6.103) to (6.106) into (6.90) and obtain

$$\int_\Omega W_i B \; d\Omega - a^*\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]U_{1_j} - b^*\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]U_{2_j}$$
$$= -\varepsilon_2\sum_{j=1}^{N}\left[\int_\Omega \nabla W_i\cdot\nabla W_j \; d\Omega\right]Q_{2_j} + \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]Q_{2_j} \qquad (6.110)$$

which is, in matrix form

$$\underline{Q}_2 = (-\varepsilon_2 K + M)^{-1}M(\underline{B} - a^*\underline{U}_1 - b^*\underline{U}_2). \qquad (6.111)$$

Here $\underline{B}$ is a vector with all entries equal to B. Equations (6.109) and (6.111) can be solved to obtain $\underline{Q}_1$ and $\underline{Q}_2$.

We now tackle the solution of $\frac{\partial U_1}{\partial t}$ by constructing equation (6.98) in finite element form. We again choose $w_i = W_i$ and make substitutions for the approximations (6.103) and (6.104), together with the derivatives

$$\frac{\partial U_1}{\partial t} = \dot{U}_1 = \sum_{j=1}^{N} W_j \dot{U}_{1_j} \qquad (6.112)$$

$$\frac{\partial U_2}{\partial t} = \dot{U}_2 = \sum_{j=1}^{N} W_j \dot{U}_{2_j}. \qquad (6.113)$$

Thus (6.98) can be rewritten in the form

$$\sum_{j=1}^{N}\left[\int_{\Omega} W_i W_j \; d\Omega\right]\dot{U}_{1_j} = -\delta_1 \sum_{j=1}^{N}\left[\int_{\Omega} \nabla W_i \cdot \nabla W_j \; d\Omega\right] U_{1_j} + \sum_{j=1}^{N}\left[\int_{\Omega} U_1 \nabla W_i \cdot \nabla W_j \; d\Omega\right] Q_{1_j}. \tag{6.114}$$

In matrix form this is

$$M\underline{\dot{U}}_1 = -\delta_1 K\underline{U}_1 + K(\underline{U}_1)\underline{Q}_1 \tag{6.115}$$

where the entries of matrix $K(\underline{U}_1)$ are those of the weighted stiffness matrix and are given by

$$K(\underline{U}_1)_{ij} = \int_{\Omega} U_1 \nabla W_i \cdot \nabla W_j \; d\Omega. \tag{6.116}$$

The process for assembly of the weighted stiffness matrix is outlined in section 3.1.2. We make finite element approximation substitutions into equation (6.98) to obtain the expression for $\frac{\partial U_2}{\partial t}$ in exactly the same manner. We arrive at the matrix form

$$M\underline{\dot{U}}_2 = -\delta_2 K\underline{U}_2 + \mu K(U_2)\underline{Q}_2. \tag{6.117}$$

Equations (6.115) and (6.117) allow us to solve for $\underline{\dot{U}}_1$ and $\underline{\dot{U}}_2$. A suitable time integration procedure can then be chosen to obtain $\underline{U}_1$ and $\underline{U}_2$.

Algorithm 14

Randomly seed a distribution of u_1 and u_2 near to a pre-chosen population density, usually 0.4 for U_1 and 0.3 for U_2. An example of such a starting point is shown in figure 6.10. Then proceed as follows until desired time is reached:

1. Obtain $\underline{Q}_1(t)$ and $\underline{Q}_2(t)$ from (6.109) and (6.111);
2. Find $\underline{\dot{U}}_1(t)$ and $\underline{\dot{U}}_2(t)$ using equations (6.115) and (6.117);
3. Generate values of $\underline{U}_1(t+dt)$ and $\underline{U}_2(t+dt)$ at the next time step using the values $\underline{\dot{U}}_1(t)$ and $\underline{\dot{U}}_2(t)$ in forward Euler time integration.

6.2.3 Results

The model has been encoded on the square domain defined by $-0.2 \leq x \leq 0.2$, $-0.2 \leq y \leq 0.2$. We set 21 by 21 regularly spaced nodes, creating a mesh containing 512 triangular elements. The default variables used for the simulations are (from [29]):

$$\begin{aligned}
&A = 1\\
&B = 1.5\\
&a = 1\\
&b = 2\\
&a^* = 3\\
&b^* = 1\\
&\varepsilon_1 = 0.025\\
&\varepsilon_2 = 0.025\\
&\delta_1 = 0.1\\
&\delta_2 = 0.1\\
&\mu = 1.
\end{aligned}$$

The initial population is generated randomly. Boundary nodes are set at $u_1 = 0.4$, $u_2 = 0.3$ for $t = 0, \mathbf{x} \in \partial\Omega$. Internal nodes are assigned a random value for u_1 from a set with a mean of 0.4 and a standard deviation of 0.01. For u_2 the random values are assigned from a set with a mean of 0.3 and a standard deviation of 0.01. These values are chosen so that E_1 and E_2 are zero (neutral survivability) at the boundaries, and have small perturbations from neutral elsewhere. These random perturbations seed the evolution of the population densities towards preferred locations. One such set of preferred locations is shown in figure 6.4. At $t = 0.7$, the two species have separated almost completely in space and four clusters are formed, each species inhabiting two corners of the domain with one favoured corner each. In figure 6.5, the same simulation is run to $t = 0.7$ with a different random initial seeding and this time only two larger clusters are formed. For the parameters used in this initial simulation, all the outputs fall broadly into one of these two categories. By experimenting with parameters, we are able to affect the number and size of clusters that are formed. Figure 6.6 shows a simulation at $t = 0.7$ with $\varepsilon_1 = \varepsilon_2 = 0.01$. The clusters produced are more compact and the four corners of the domain are more evenly populated. Running further simulations with these parameters always produces this four-corner pattern. However, the question of which species inhabit which diagonal pair of corners is determined by the random seeding.

The simulations run smoothly with an initial diffusion dominated phase lasting to approximately $t = 0.05$, whilst groupings are established and peak population densities are reduced, then a much longer group growth stage where populations tend towards their groups containing maximum sustainable density, *i.e.* $u_1 = 1$ and $u_2 = 1.5$. Between groups populations are approximately zero, and the habitat appropriated by each species is clearly defined. Steady state is reached at around $t = 0.7$. These simulations are robust to significant experimentation with parameters and so provide a useful tool for understanding the behaviour described by the model.

6.2.4 The non-conservative population case

We now consider the evolution of a system that allows births and deaths to take place. This is non mass conserving so the treatment is slightly different. The equations for q_1, q_2, E_1 and E_2, and the boundary conditions are unchanged from the conservative case. These are given as

$$\varepsilon_1 \nabla^2 q_1 + q_1 = E_1 \tag{6.118}$$

$$\varepsilon_2 \nabla^2 q_2 + q_2 = E_2 \tag{6.119}$$

$$E_1 = A - au_1 - bu_2 \tag{6.120}$$

$$E_2 = B - a^* u_1 - b^* u_2 \tag{6.121}$$

with boundary conditions

$$\nabla u \cdot \hat{\mathbf{n}} = 0 \qquad \mathbf{x} \in \partial\Omega, t \geq 0 \tag{6.122}$$

$$\nabla q \cdot \hat{\mathbf{n}} = 0 \qquad \mathbf{x} \in \partial\Omega, t \geq 0. \tag{6.123}$$

However, for the time dependent PDEs we have a different system. We set the reproduction parameter $r = 1$, so that (6.80) and (6.81) become

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla \cdot (u_1 \nabla q_1) + u_1 E_1 \tag{6.124}$$

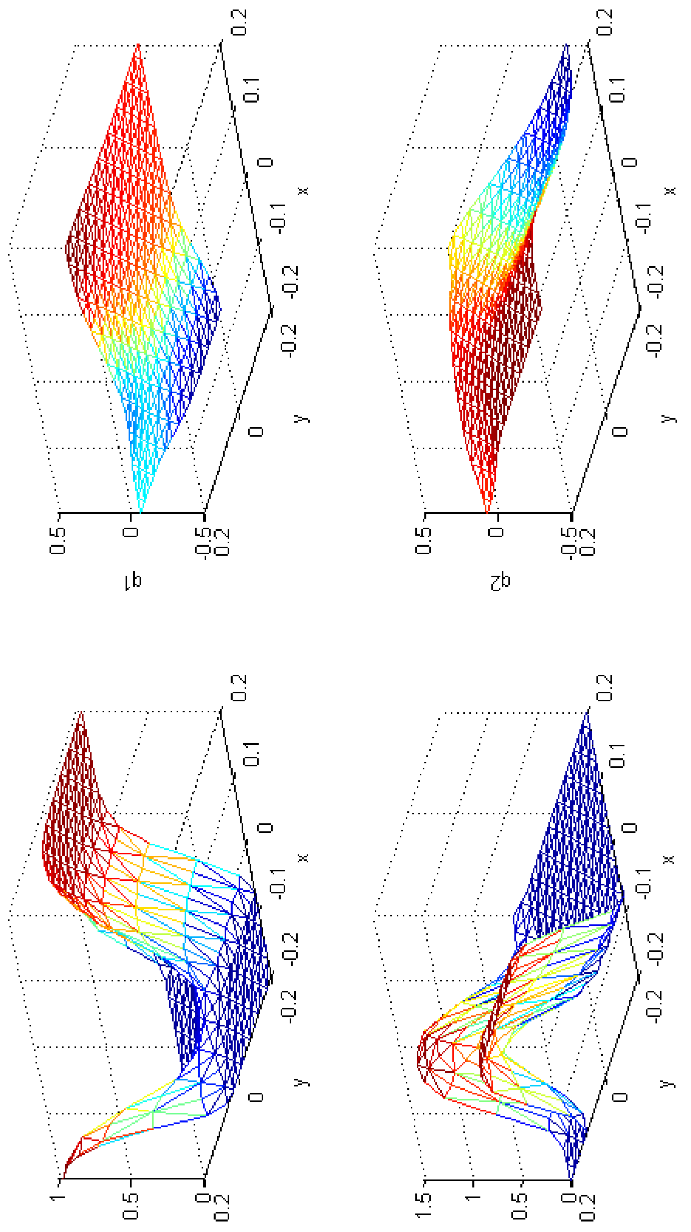


Fig. 6.4 A conservative, static mesh, two species simulation at $t = 0.7$ with $\varepsilon_1 = \varepsilon_2 = 0.025$ and $\delta_1 = \delta_2 = 0.1$. Initial seeding is random, so no two results are identical.

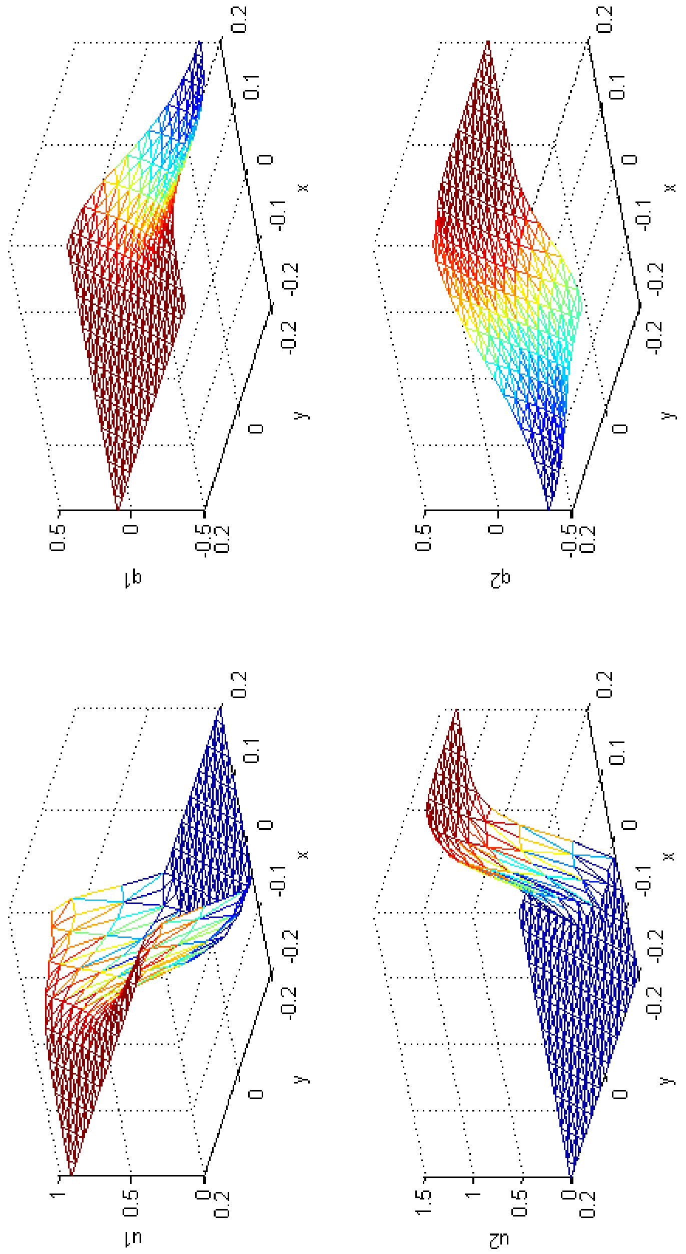


Fig. 6.5 An alternative result from the conservative, static mesh, two species simulation $t = 0.7$, the only difference being in the initial random population seeding. The parameters are identical to those for figure 6.4, $\varepsilon_1 = \varepsilon_2 = 0.025$ and $\delta_1 = \delta_2 = 0.1$.

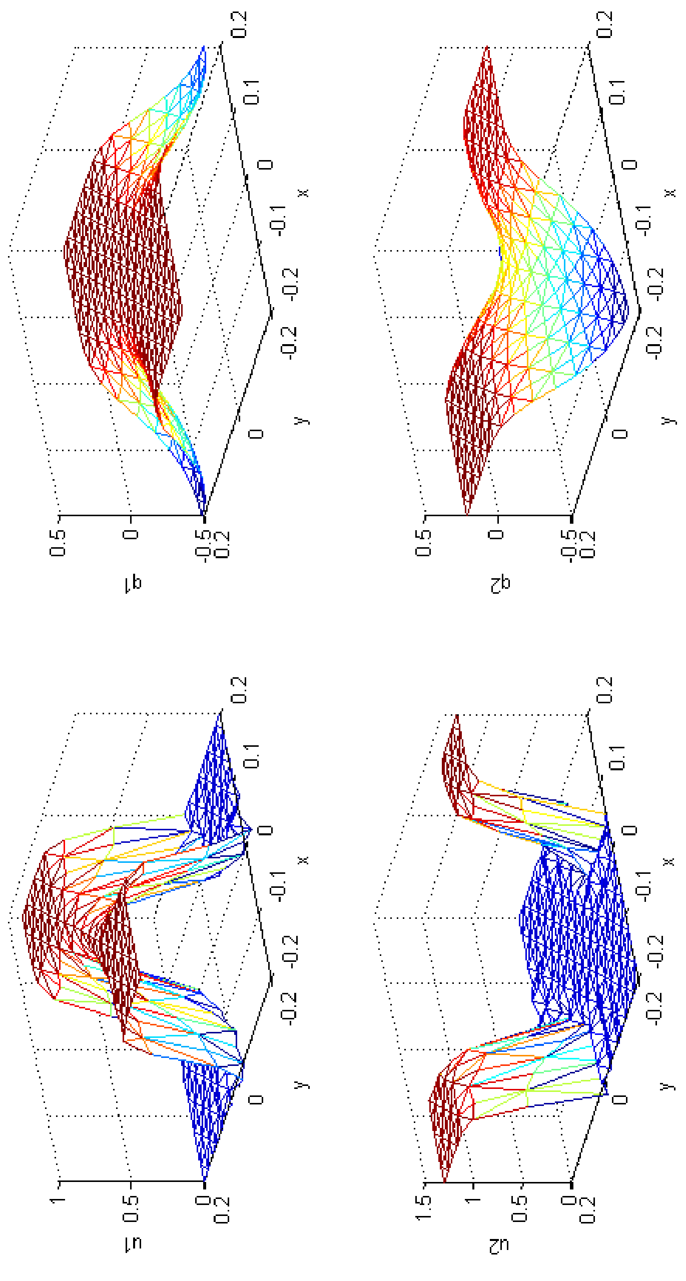


Fig. 6.6 A result from the conservative, static mesh, two species simulation $t = 0.7$ with an alternative choice for ε. Here $\varepsilon_1 = \varepsilon_2 = 0.01$ and $\delta_1 = \delta_2 = 0.1$ as before. Note the difference in q_1 and q_2 now that the scale over which diffusion is important is reduced.

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \mu \nabla \cdot (u_2 \nabla q_2) + u_2 E_2. \tag{6.125}$$

This means that we have an extra term to consider in constructing the appropriate weak form for finite element substitutions. The weak form of (6.124) is

$$\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = \delta_1 \int_\Omega w_i \nabla^2 u_1\ d\Omega - \int_\Omega w_i u_1 \nabla^2 q_1\ d\Omega - \int_\Omega w_i \nabla q_1 \cdot \nabla u_1\ d\Omega + \int_\Omega w_i u_1 E_1\ d\Omega \tag{6.126}$$

where w_i is part of a set of weight functions that together form a partition of unity. We treat the first three terms on the right hand side in the same manner as in the conservative case, given by equations (6.92) to (6.98). We obtain the simpler weak form

$$\int_\Omega w_i \frac{\partial u_1}{\partial t}\ d\Omega = -\delta_1 \int_\Omega \nabla w_i \cdot \nabla u_1\ d\Omega + \int_\Omega u_1 \nabla w_i \cdot \nabla q_1\ d\Omega + \int_\Omega w_i u_1 E_1\ d\Omega \tag{6.127}$$

and similarly, from (6.125) we obtain the weak form

$$\int_\Omega w_i \frac{\partial u_2}{\partial t}\ d\Omega = -\delta_2 \int_\Omega \nabla w_i \cdot \nabla u_2\ d\Omega + \int_\Omega \mu u_2 \nabla w_i \cdot \nabla q_2\ d\Omega + \int_\Omega w_i u_2 E_2\ d\Omega. \tag{6.128}$$

6.2.5 Construction of the finite element form

We choose the piecewise linear weight functions $w_i = W_i$. We use the piecewise linear approximations (6.103) to (6.106). We also use a similar choice for E_1 and E_2, defined as

$$E_1(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) E_{1_j}(t) \tag{6.129}$$

$$E_2(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x}) E_{2_j}(t). \tag{6.130}$$

Substituting these choices into (6.127) we obtain

$$\begin{aligned}\sum_{j=1}^{N} \left[\int_\Omega W_i W_j\ d\Omega\right] \dot{U}_{1_j} &= -\delta_1 \sum_{j=1}^{N} \left[\int_\Omega \nabla W_i \cdot \nabla W_j\ d\Omega\right] U_{1_j} \\ &+ \sum_{j=1}^{N} \left[\int_\Omega U_1 \nabla W_i \cdot \nabla W_j\ d\Omega\right] Q_{1_j} + \int_\Omega W_i U_1 E_1\ d\Omega.\end{aligned} \tag{6.131}$$

The matrix form is then

$$M\underline{\dot{U}}_1 = -\delta_1 K\underline{U}_1 + K(U_1)\underline{Q}_1 + \underline{N}_1 \tag{6.132}$$

where $\underline{N}_1 = \int_\Omega W_i U_1 E_1 \; d\Omega$, a nonlinear term which can be evaluated using Gaussian quadrature (see Appendix B). This matrix form may be rearranged to give

$$\underline{\dot{U}}_1 = M^{-1}(-\delta_1 K\underline{U}_1 + K(U_1)\underline{Q}_1 + \underline{N}_1 \tag{6.133}$$

which is the non-conservative finite element form giving $\underline{\dot{U}}_1$.

Similarly, substituting the piecewise linear approximations into (6.128) we obtain the equivalent form for $\underline{\dot{U}}_2$,

$$\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right]\dot{U}_{2_j} = -\delta_2 \sum_{j=1}^{N}\left[\int_\Omega \nabla W_i \cdot \nabla W_j \; d\Omega\right] U_{2_j}$$
$$+ \sum_{j=1}^{N}\left[\int_\Omega \mu U_2 \nabla W_i \cdot \nabla W_j \; d\Omega\right] Q_{2_j} + \int_\Omega W_i U_2 E_2 \; d\Omega. \tag{6.134}$$

The matrix form is then

$$M\underline{\dot{U}}_2 = -\delta_2 K\underline{U}_2 + \mu K(U_2)\underline{Q}_2 + \underline{N}_2. \tag{6.135}$$

The nonlinear term $\underline{N}_2$ is given by $\underline{N}_2 = \int_\Omega W_i U_2 E_2 \; d\Omega$. This can be calculated in the same exact way as for $\underline{N}_1$. We may rearrange this matrix form to give

$$\underline{\dot{U}}_2 = M^{-1}(-\delta_2 K\underline{U}_2 + \mu K(U_2)\underline{Q}_2 + \underline{N}_2. \tag{6.136}$$

Algorithm 15

Randomly seed a distribution of u_1 and u_2 near to a pre-chosen population density, usually 0.4 for u_1 and 0.3 for u_2. An example of such a starting point is shown in figure 6.10. Then proceed as follows until desired time is reached:

1. Obtain $\underline{Q}_1(t)$ and $\underline{Q}_2(t)$ from (6.109) and (6.111);

2. Find $\underline{\dot{U}}_1(t)$ and $\underline{\dot{U}}_2(t)$ using equations (6.133) and (6.136);

3. Generate values of $\underline{U}_1(t+dt)$ and $\underline{U}_2(t+dt)$ at the next time step using the values

$\underline{\dot{U}}_1(t)$ and $\underline{\dot{U}}_2(t)$ in forward Euler time integration.

6.2.6 Results

We use the same grid and default variables as in the conservative case, given in section 6.2.3. The initialisation is also unchanged from the conservative case. Boundary nodes are set at $u_1 = 0.4$, $u_2 = 0.3$ for $t = 0, \mathbf{x} \in \partial\Omega$. Internal nodes are assigned a random value for u_1 at $t = 0$ from a set with a mean of 0.4 and a standard deviation of 0.01. For u_2 at $t = 0$ the random values are assigned from a set with a mean of 0.3 and a standard deviation of 0.01. Again the simulations run smoothly with the short initial diffusion dominated phase then the much longer group growth stage. Steady state, or at least a phase of very slow change, is reached at approximately $t = 0.7$ with no significant change thereafter to at least $t = 25$. The results shown here show a single simulation at different stages. We show progress of clusters forming at $t = 0.5$ (figure 6.7), smaller clusters becoming extinct at $t = 1.0$ (figure 6.8) and a straighter interface forming at $t = 1.5$ (figure 6.9). Compared to the conservative case, we see that only the larger groupings survive, which is to be expected if threatened populations are now allowed to suffer deaths. We also see the formation of a clear and increasingly straight interface between the two populations. As regards our aim of generating a system that truly tends towards a zero population species interface suitable for a spatially segregated multi-phase model, this is a success.

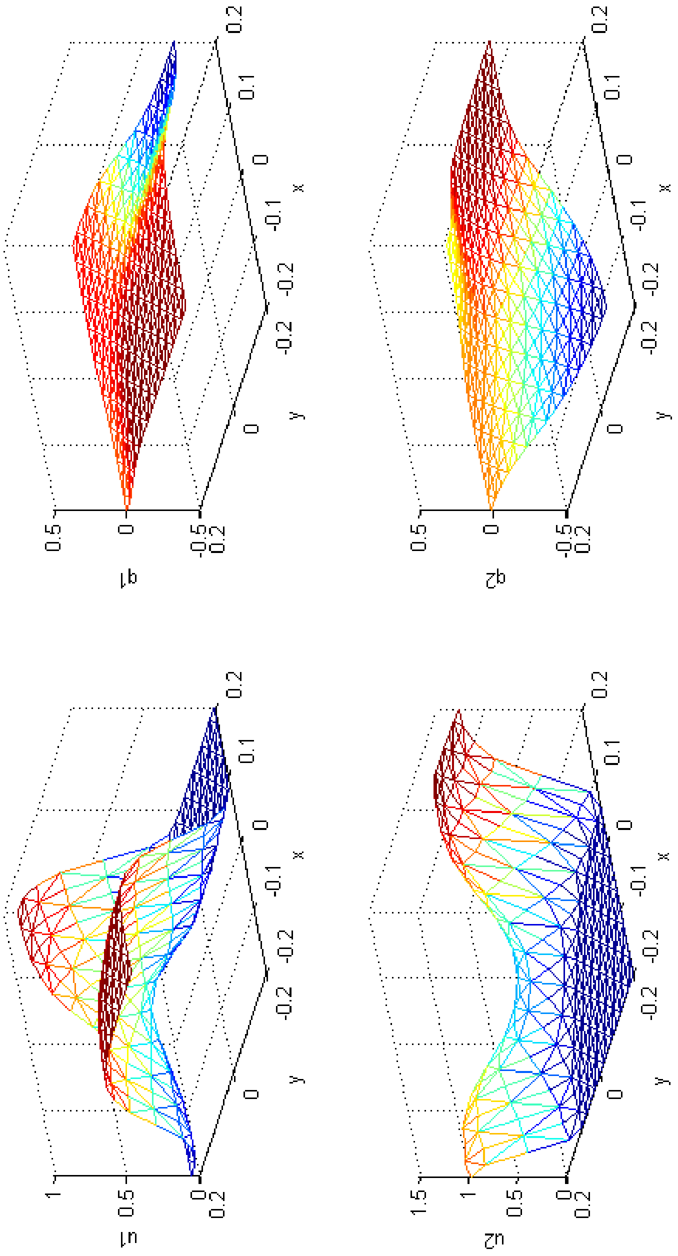


Fig. 6.7 An example result from the non-conservative static mesh at $t = 0.5$. Random seeding is used to produce the initial conditions. In this case two clusters of each species are formed.

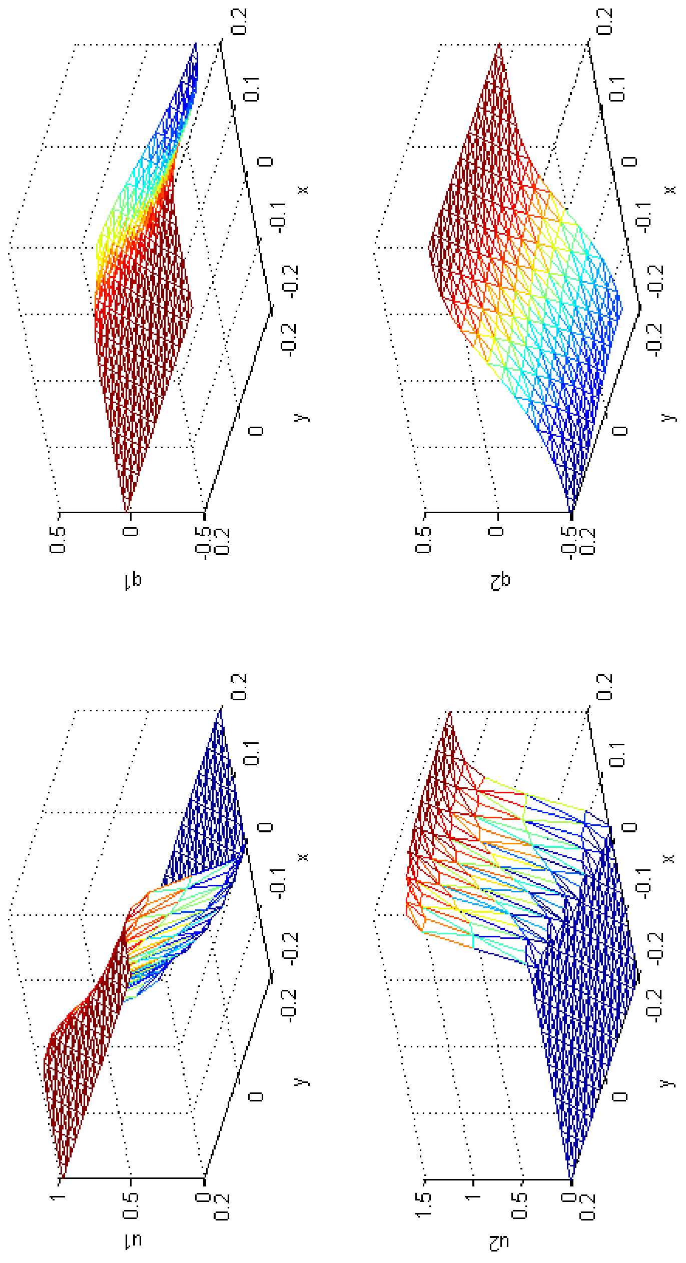


Fig. 6.8 An example result from the non-conservative static mesh at $t = 1.0$. Random seeding is used to produce the initial conditions. As the reproductive terms make impact, the number of clusters is reduced to one per species.

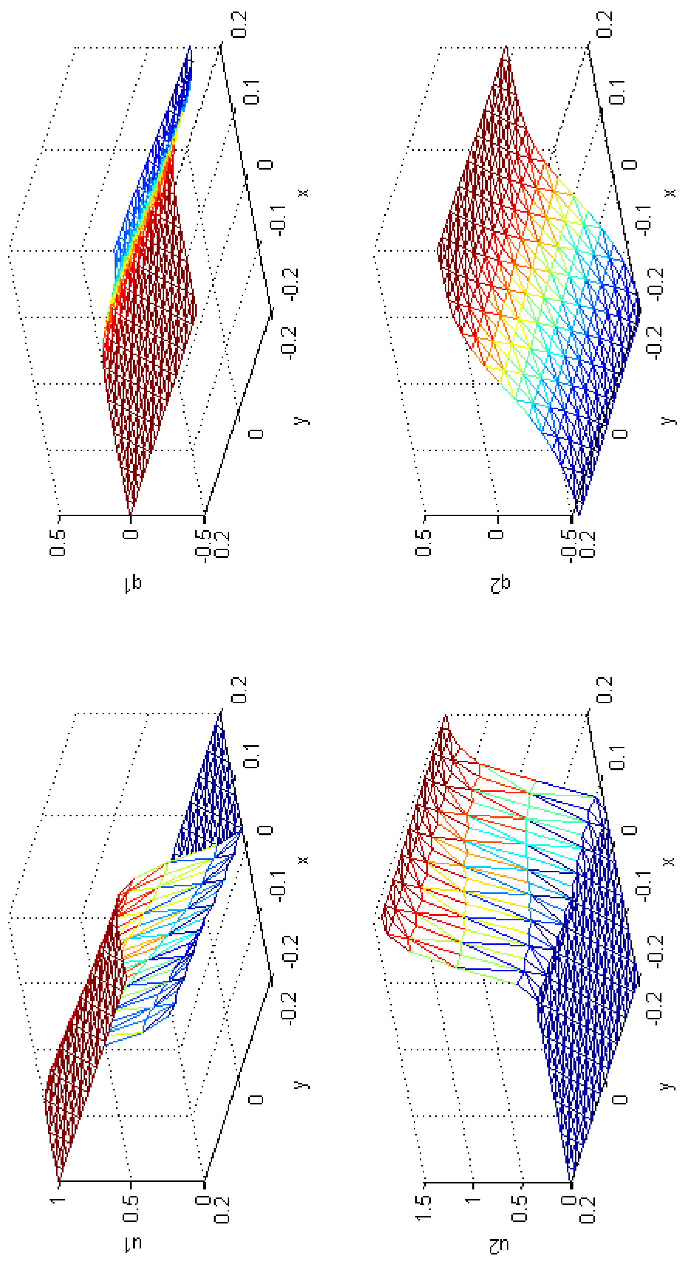


Fig. 6.9 An example result from the non-conservative static mesh at $t = 1.5$. Random seeding is used to produce the initial conditions. The clusters become more established and the interface becomes straighter.

6.2.7 A change in the resource space

An interesting consideration is how changing the resource space affects the dynamics of the group. This is particularly relevant when we move on to restricting each species to its own domain. We can see how the shape of the interface will come into play, as well as our later look at the effect of the interface as it is moving. The variables A and B are the carrying capacities for species 1 and species 2 respectively, and can be considered to represent the maximum resource a species can access. In this simulation, we look at the effect of removing the resource from a part of the domain after a period of time during which groupings have become established. We allow the simulation to run as normal to $t = 1.0$ (figures 6.10 and 6.11), with the usual random population seeding, and then we reduce A and B to zero in one quadrant of the domain (figures 6.12 and 6.13). After the removal of resource, overcrowding in the inhabited areas is apparent. Note the change of scale on the u_1 and u_2 axes in figures 6.12 and 6.13. This overcrowding takes a long time to resolve because although the survivability is reduced at the centre of the crowd, the population at the edge of the crowd is both growing and moving towards the centre. The observed trend suggests that a steady state balanced population would be reached at something of the order of $t = 200$.

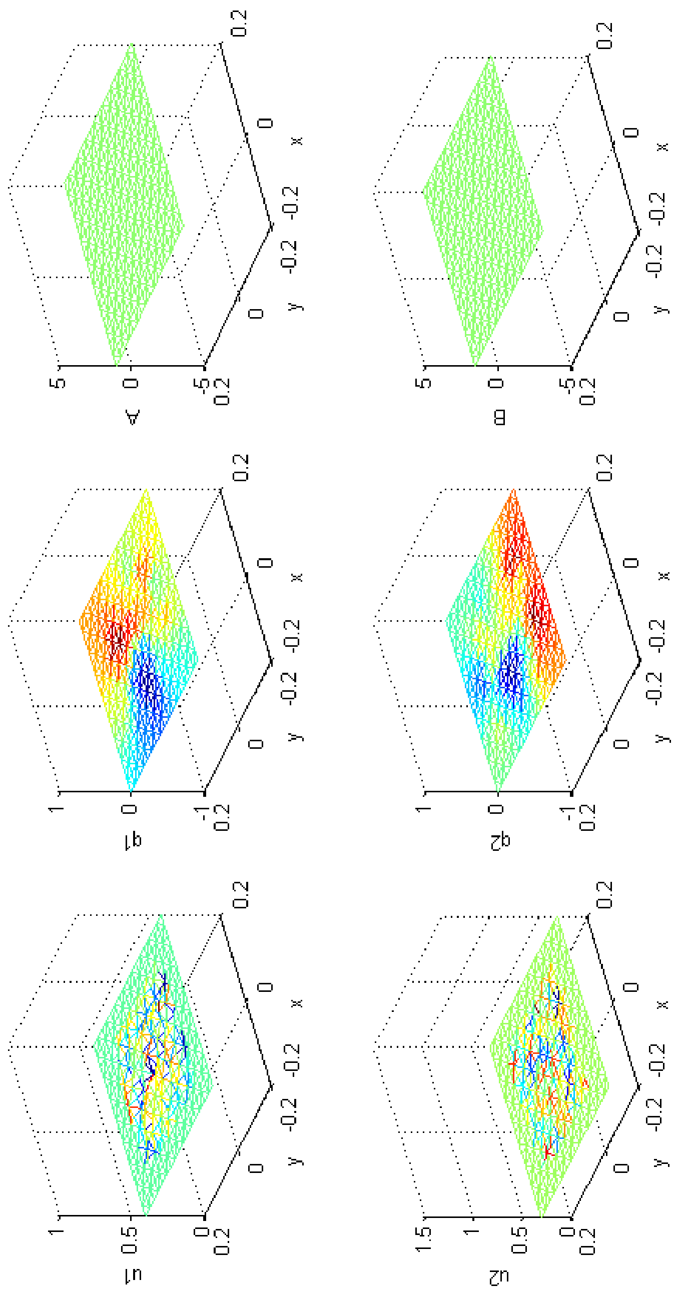


Fig. 6.10 An example with changing resource space, showing random population seeding at $t = 0.0$. At this stage, resource distribution is homogenous.

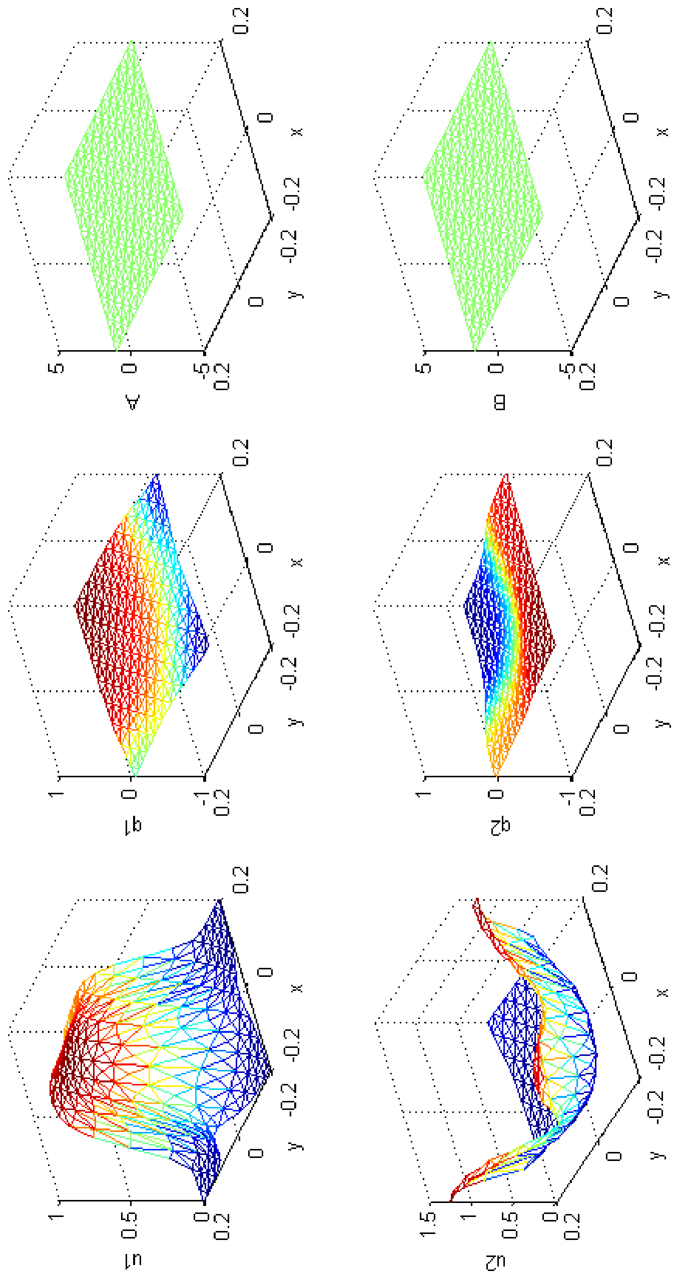


Fig. 6.11 An example with changing resource space, showing random population seeding at $t = 1.0$. At this stage, resource distribution is still homogenous.

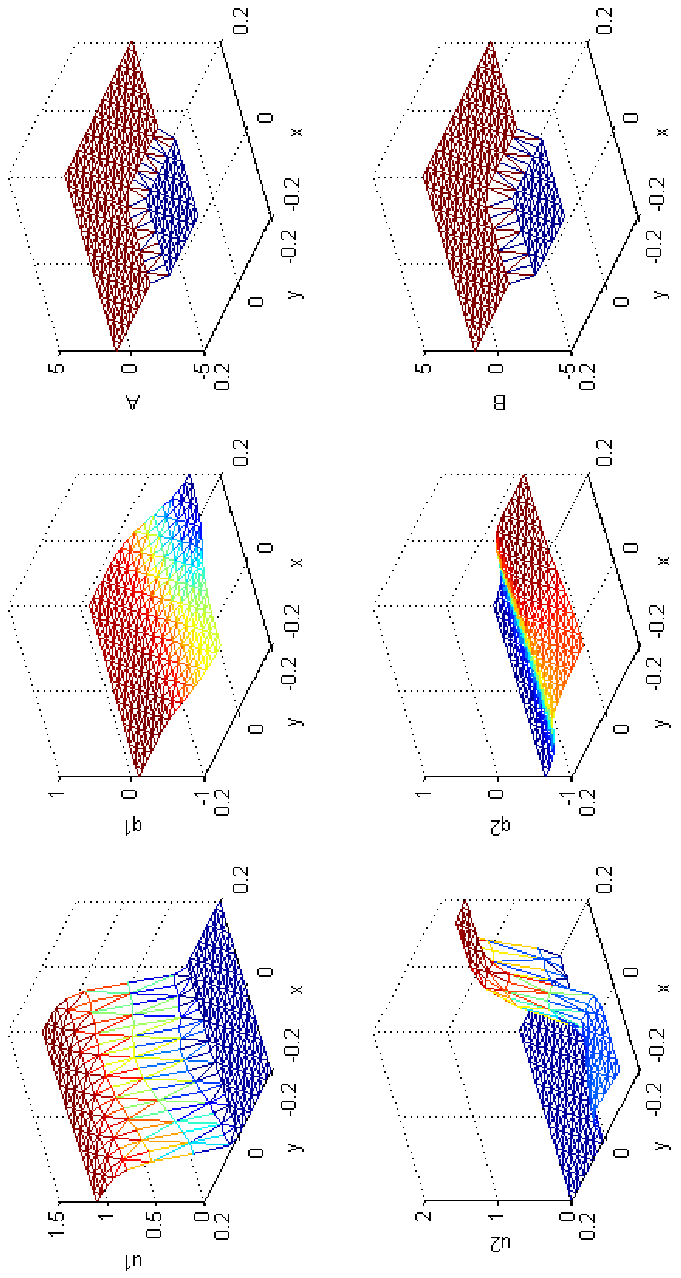


Fig. 6.12 An example with changing resource space, showing random population seeding at $t = 1.5$. At this stage, resource distribution is non-homogenous and species 2 is subject to a falling population.

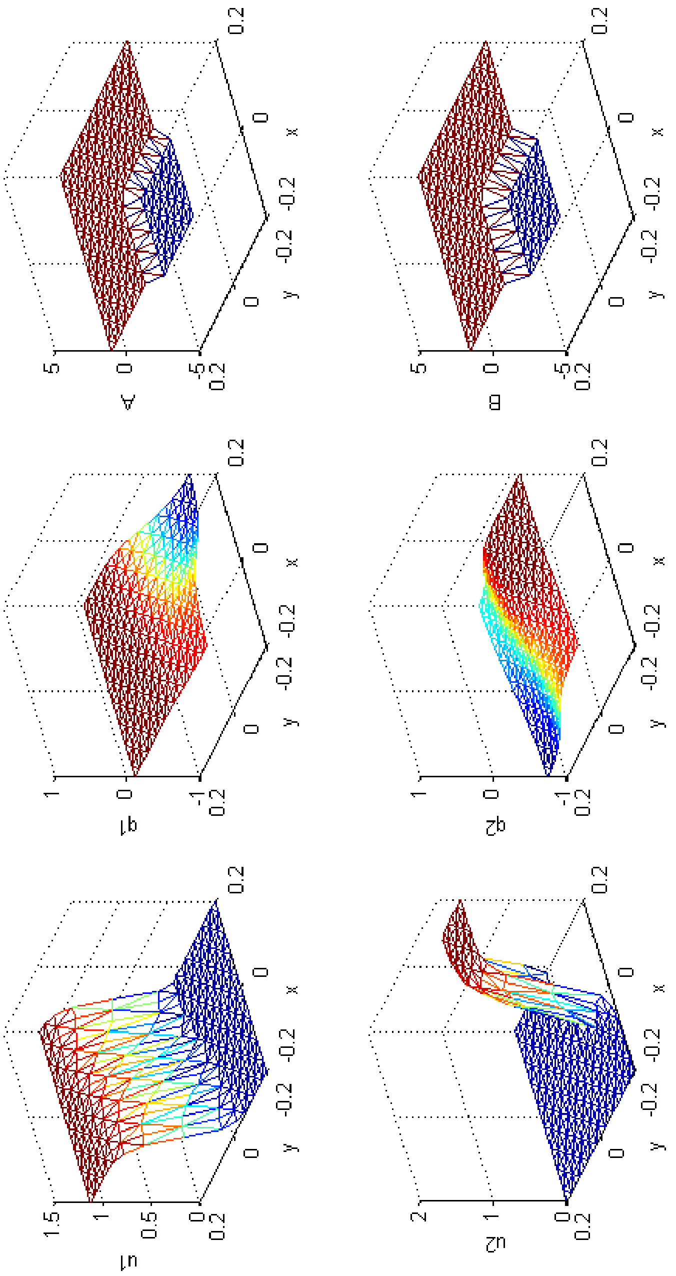


Fig. 6.13 An example with changing resource space, showing random population seeding at $t = 2.0$. At this stage, resource distribution is non-homogenous, but species 2 has adapted to a new domain and is forming a smaller cluster.

Chapter 7

A combined model with a moving interface

7.1 The two phase model of competition-diffusion-aggregation

We propose a model for a two component reaction-diffusion-aggregation system based on the Lotka-Volterra competition system, which will additionally incorporate the aggregation characteristics proposed by Grindrod [29] and the interface condition proposed by Hilhorst [31]. We construct the model in such a way that we will be able to utilise the two phase MMFEM of Baines, Hubbard *et al.* [8], with an adapting mesh based on a relative conservation principle. The PDE system that defines the basis of the model is given by the reaction-diffusion-aggregation PDEs from [29]. See Chapter 6, section 6.2 for a more detailed background. In Chapter 6 we derived a model based upon the same PDEs for two species sharing a domain, but here we are concerned with a truly two phase model. The driving PDEs are

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla\left(u_1 \nabla q_1\right) + r u_1 E_1 \qquad t > 0, x \in \Omega_1(t) \tag{7.1}$$

and

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \rho \nabla\left(u_2 \nabla q_2\right) + r u_2 E_2 \qquad t > 0, x \in \Omega_2(t). \tag{7.2}$$

We use a fixed domain Ω bounded externally by S_e, but Ω is divided into two subdomain classes Ω_1 and Ω_2 which are separated by the moving interface(s) $S_m(t)$. The 1-D analogies are given by

$$\frac{\partial u_1}{\partial t} = \delta_1 \frac{\partial^2 u_1}{\partial x^2} - \frac{\partial}{\partial x}\left(u_1 \frac{\partial q_1}{\partial x}\right) + r u_1 E_1 \qquad t > 0, x \in (\alpha, m(t)) \tag{7.3}$$

and

$$\frac{\partial u_2}{\partial t} = \delta_2 \frac{\partial^2 u_2}{\partial x^2} - \rho \frac{\partial}{\partial x}\left(u_2 \frac{\partial q_2}{\partial x}\right) + r u_2 E_2 \qquad t > 0, x \in (m(t), \beta) \tag{7.4}$$

for a domain with fixed boundaries α and β but with a moving interface between species $m(t)$. The parameters E_1 and E_2 are the net reproduction rates for each species, given by the logistic equations

$$E_1 = A - au_1 - bu_2 \tag{7.5}$$

$$E_2 = B - a^* u_1 - b^* u_2. \tag{7.6}$$

We can see that this system also has parallels with the competition-diffusion model of Chapter 5. This system differs from that in Chapter 5 in the additional consideration of an aggregation component (the term containing ρ). We note that the parameters used in the expressions (7.5) and (7.6) for the reproduction rate E are named differently to the competition diffusion model, but we can see that no material difference exists. For simplicity we adopt the naming conventions used by Hilhorst where we extend her work, and have followed the naming conventions used by Grindrod where we extend his work. This model is a combination model drawing together both of those threads, but we will use the Grindrod naming conventions as the driving PDEs come from his work.

The aggregating behaviour is defined by the PDEs

$$\varepsilon_1 \nabla^2 q_1 + q_1 = E_1 \tag{7.7}$$

$$\varepsilon_2 \nabla^2 q_2 + q_2 = E_2 \tag{7.8}$$

for which the 1-D analogies are

$$\varepsilon_1 \frac{\partial^2 q_1}{\partial x^2} + q_1 = E_1 \tag{7.9}$$

$$\varepsilon_2 \frac{\partial^2 q_2}{\partial x^2} + q_2 = E_2. \tag{7.10}$$

7.2 1-D competition-aggregation-diffusion in a two phase model

We have boundary conditions given by

$$\begin{aligned} \frac{\partial u}{\partial x} &= 0 & x &= \alpha, \beta \\ \frac{\partial q}{\partial x} &= 0 & & \\ u &= 0 & x &= m(t) \end{aligned} \tag{7.11}$$

and we work in the high competition limit defined by Hilhorst [31], so that the species cannot exist in the opposite species' domain. Formally,

$$\begin{aligned} u_1 &= 0 & x &\in [m(t), \beta] \\ u_2 &= 0 & x &\in [\alpha, m(t)]. \end{aligned} \tag{7.12}$$

The interface condition is taken from [31], and is

$$\mu \delta_1 \frac{\partial u_1}{\partial x}\bigg|_{m(t)} = -\delta_2 \frac{\partial u_2}{\partial x}\bigg|_{m(t)} \tag{7.13}$$

where, once parameter naming conventions are compared between [31] and [29], $\mu = aa^*/bb^*$. We will call μ the interspecies competition rate. We work with Neumann boundary conditions on the external boundaries, which will be fixed. We use parameter choices from [29] which are given in Chapter 6, section 6.2.3. In order to set suitable initial conditions, we consider the results of the shared-domain clustering models of Chapter 6. We note the steady state solutions that arise from the Chapter 6 models, and construct initial conditions that approximate those steady state results. These are given by figure 7.1. We begin by redefining the driving Lotka-Volterra based equations (7.3) and (7.4) in weak form, incorporating the weight function w_i,

$$\int_\alpha^\beta w_i \frac{\partial u_1}{\partial t} dx = \int_\alpha^\beta \delta_1 w_i \frac{\partial^2 u_1}{\partial x^2} dx - \int_\alpha^\beta w_i \frac{\partial}{\partial x}\left(u_1 \frac{\partial q_1}{\partial x}\right) dx + \int_\alpha^\beta w_i r_1 u_1 E_1 \; dx \tag{7.14}$$

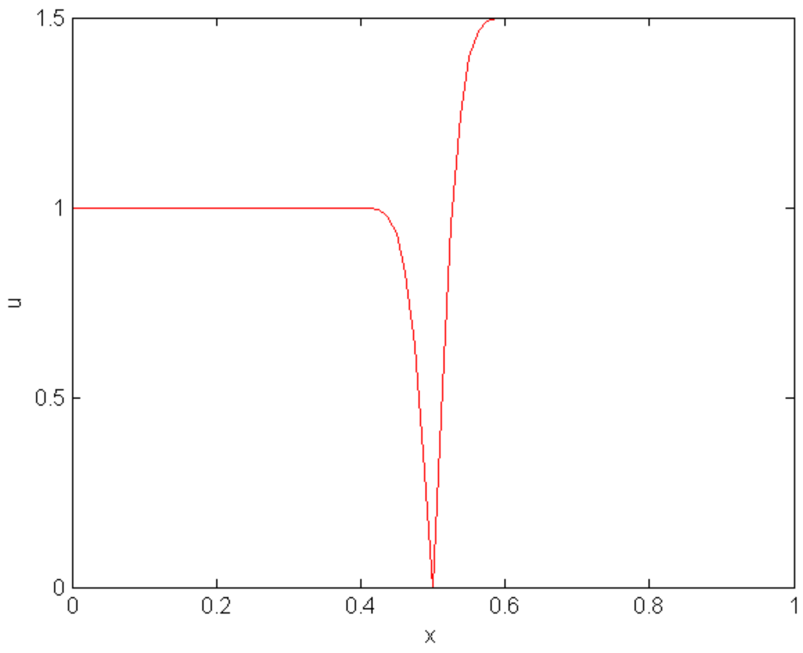


Fig. 7.1 Initial conditions for the two-phase reaction-diffusion-aggregation model. The amplitudes are taken from the steady state results arising from the shared domain model of Chapter 6. Species 1 in on the left and species 2 is on the right.

$$\int_{\alpha}^{\beta} w_i \frac{\partial u_2}{\partial t} dx = \int_{\alpha}^{\beta} \delta_1 w_i \frac{\partial^2 u_2}{\partial x^2} dx - \int_{\alpha}^{\beta} w_i \rho \frac{\partial}{\partial x}\left(u_2 \frac{\partial q_2}{\partial x}\right) dx + \int_{\alpha}^{\beta} w_i r_2 u_2 E_2 \; dx. \quad (7.15)$$

We substitute the definitions for E_1 (7.5) and E_2 (7.6), noting that because we have the high competition limit, the terms containing b and a^* are equal to zero. We obtain

$$\int_{\alpha}^{m(t)} w_i \frac{\partial u_1}{\partial t} dx = \int_{\alpha}^{m(t)} \delta_1 w_i \frac{\partial^2 u_1}{\partial x^2} dx - \int_{\alpha}^{m(t)} w_i \frac{\partial}{\partial x}\left(u_1 \frac{\partial q_1}{\partial x}\right) dx$$
$$+ \int_{\alpha}^{m(t)} w_i r_1 u_1 (A - a u_1) \; dx \quad (7.16)$$

and

$$\int_{m(t)}^{\beta} w_i \frac{\partial u_2}{\partial t} dx = \int_{m(t)}^{\beta} \delta_1 w_i \frac{\partial^2 u_2}{\partial x^2} dx - \int_{m(t)}^{\beta} w_i \rho \frac{\partial}{\partial x}\left(u_2 \frac{\partial q_2}{\partial x}\right) dx$$
$$+ \int_{m(t)}^{\beta} w_i r_2 u_2 (B - b^* u_2) \; dx. \quad (7.17)$$

We do not have a mass conserving system, so instead we use the idea of conserving relative mass. We define the total population of a species as θ, given by

$$\theta(t) = \int_{R(t)} u dx \quad (7.18)$$

where $R(t)$ is the moving domain inhabited by that species. A relative conservation principle can now be defined in terms of θ. We introduce the weight function w_i,

$$\frac{1}{\theta(t)} \int_{R(t)} w_i u dx = c_i \quad (7.19)$$

or

$$\int_{R(t)} w_i u dx = c_i \theta(t) = c_i \int_{R(t)} u dx. \quad (7.20)$$

The constant c_i is determined by the choice of w_i. We require that w_i is part of a set which forms a partition of unity. We now have, in equation (7.20), our distributed conservation of mass principle. Using the Leibnitz integral rule, we differentiate (7.20) with respect to time on our moving frame $R(t)$,

$$\frac{d}{dt}\left[\int_{R(t)} w_i u dx\right] = \int_{R(t)} \left(\frac{\partial (w_i u)}{\partial t} + \frac{\partial}{\partial x}(w_i u \dot{x})\right) dx. \quad (7.21)$$

We impose the condition that the basis functions w_i move with the domain. Hence the basis functions also have velocity $\dot{x}$. By analogy with advection, we write,

$$\frac{\partial w_i}{\partial t} + \dot{x}\frac{\partial w_i}{\partial x} = 0 \tag{7.22}$$

hence

$$\frac{d}{dt}\left[\int_{R(t)} w_i u dx\right] = \int_{R(t)} w_i \left(\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(u\dot{x})\right) dx \tag{7.23}$$

or

$$\frac{d}{dt}\left[\int_{R(t)} w_i u dx\right] - \int_{R(t)} w_i \frac{\partial}{\partial x}(u\dot{x})dx = \int_{R(t)} w_i \frac{\partial u}{\partial t}dx. \tag{7.24}$$

In terms of $\dot{\theta}$ and the constants c_i, equation (7.24) becomes

$$c_i\dot{\theta} - \int_{R(t)} w_i \frac{\partial}{\partial x}(u\dot{x})dx = \int_{R(t)} w_i \frac{\partial u}{\partial t}dx. \tag{7.25}$$

We introduce the velocity potential ϕ defined by

$$\dot{x} = \frac{\partial \phi}{\partial x} \tag{7.26}$$

so that

$$c_i\dot{\theta} - \int_{R(t)} w_i \frac{\partial}{\partial x}\left(u\frac{\partial \phi}{\partial x}\right) dx = \int_{R(t)} w_i \frac{\partial u}{\partial t}dx \tag{7.27}$$

or, after integration by parts

$$c_i\dot{\theta} + \int_{R(t)} u\frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}dx - \left[u w_i \frac{\partial \phi}{\partial x}\right]_{\partial R(t)} = \int_{R(t)} w_i \frac{\partial u}{\partial t}dx. \tag{7.28}$$

This equation holds for either of our species. We now construct a form unique to each species. We make the species specific definitions of total mass (*cf.* equation (7.18)),

$$\theta_1(t) = \int_{\alpha}^{m(t)} u_1 \; dx \tag{7.29}$$

$$\theta_2(t) = \int_{m(t)}^{\beta} u_2 \; dx. \tag{7.30}$$

The weak forms are then

$$c_{1_i}\theta_1(t) = \int_{\alpha}^{m(t)} w_i u \; dx \tag{7.31}$$

$$c_{2_i}\theta_2(t) = \int_{m(t)}^{\beta} w_i u \; dx. \tag{7.32}$$

Then equation (7.28) becomes, for species 1,

$$c_{1_i}\dot{\theta}_1 + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_1 w_i \frac{\partial \phi}{\partial x} \right]_{\alpha}^{m(t)} = \int_{\alpha}^{m(t)} w_i \frac{\partial u_1}{\partial t} dx. \tag{7.33}$$

We now substitute (7.16). We obtain

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_1 w_i \frac{\partial \phi}{\partial x} \right]_{\alpha}^{m(t)} &= \int_{\alpha}^{m(t)} \delta_1 w_i \frac{\partial^2 u_1}{\partial x^2} dx \\ - \int_{\alpha}^{m(t)} w_i \frac{\partial}{\partial x}\left(u_1 \frac{\partial q_1}{\partial x} \right) dx &+ \int_{\alpha}^{m(t)} w_i r_1 u_1 (A - a u_1) \; dx. \end{aligned} \tag{7.34}$$

Integration by parts on the right leads to

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_1 w_i \frac{\partial \phi}{\partial x} \right]_{\alpha}^{m(t)} &= - \int_{\alpha}^{m(t)} \delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_1}{\partial x} dx + \left[w_i \delta_1 \frac{\partial u_1}{\partial x} \right]_{\alpha}^{m(t)} \\ + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial q_1}{\partial x} dx - \left[w_i u_1 \frac{\partial q_1}{\partial x} \right]_{\alpha}^{m(t)} &+ \int_{\alpha}^{m(t)} w_i r_1 u_1 (A - a u_1) \; dx. \end{aligned} \tag{7.35}$$

We note the zero Neumann boundary conditions (7.11) on q_1 and u_1 at the external boundary, and the Dirichlet boundary condition (7.12) on u_1 at the interface. We also note the fixed external boundaries which mean that $\frac{\partial \phi}{\partial x} = 0$ on α. These conditions mean that most of the boundary terms in (7.35) are equal to zero. The remaining expression is

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx = - \int_{\alpha}^{m(t)} \delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_1}{\partial x} dx + w_i \delta_1 \frac{\partial u_1}{\partial x} \bigg|_{m(t)} \\ + \int_{\alpha}^{m(t)} u_1 \frac{\partial w_i}{\partial x}\frac{\partial q_1}{\partial x} dx + \int_{\alpha}^{m(t)} w_i r_1 u_1 (A - a u_1) \; dx. \end{aligned} \tag{7.36}$$

Likewise, equation (7.33) becomes, for species 2,

$$c_{2_i}\dot{\theta}_2 + \int_{m(t)}^{\beta} u_2 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_2 w_i \frac{\partial \phi}{\partial x} \right]_{m(t)}^{\beta} = \int_{m(t)}^{\beta} w_i \frac{\partial u_2}{\partial t} dx. \tag{7.37}$$

After substitution of (7.17), equation (7.37) becomes

$$c_{2_i}\dot{\theta}_2 + \int_{m(t)}^{\beta} u_2 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_2 w_i \frac{\partial \phi}{\partial x} \right]_{m(t)}^{\beta} = \int_{m(t)}^{\beta} \delta_1 w_i \frac{\partial^2 u_2}{\partial x^2} dx$$
$$- \int_{m(t)}^{\beta} w_i \rho \frac{\partial}{\partial x}\left(u_2 \frac{\partial q_2}{\partial x} \right) dx + \int_{m(t)}^{\beta} w_i r_2 u_2 (B - b^* u_1)\ dx. \tag{7.38}$$

Integration by parts on the right leads to

$$c_{2_i}\dot{\theta}_2 + \int_{m(t)}^{\beta} u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx - \left[u_2 w_i \frac{\partial \phi}{\partial x} \right]_{m(t)}^{\beta} = - \int_{m(t)}^{\beta} \delta_2 \frac{\partial w_i}{\partial x}\frac{\partial u_2}{\partial x} dx + \left[w_i \delta_2 \frac{\partial u_2}{\partial x} \right]_{m(t)}^{\beta}$$
$$+ \int_{m(t)}^{\beta} \rho u_2 \frac{\partial w_i}{\partial x}\frac{\partial q_2}{\partial x} dx - \left[w_i \rho u_2 \frac{\partial q_2}{\partial x} \right]_{m(t)}^{\beta} + \int_{m(t)}^{\beta} w_i r_2 u_2 (B - b^* u_2)\ dx. \tag{7.39}$$

After considering the boundary conditions (7.11) and (7.12) the remaining expression is

$$c_{2_i}\dot{\theta}_2 + \int_{m(t)}^{\beta} u_2 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x} dx = - \int_{m(t)}^{\beta} \delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_2}{\partial x} dx - w_i \delta_2 \frac{\partial u_2}{\partial x}\bigg|_{m(t)}$$
$$+ \int_{m(t)}^{\beta} \rho u_2 \frac{\partial w_i}{\partial x}\frac{\partial q_2}{\partial x} dx + \int_{m(t)}^{\beta} w_i r_2 u_2 (B - b^* u_2)\ dx. \tag{7.40}$$

We may solve (7.36) and (7.40) for ϕ given q_1, q_2, θ_1 and θ_2. This will allow us to subsequently recover the nodal velocities. To obtain q_1 and q_2, we refer to equations (7.5) and (7.6), (7.9) and (7.10). Combining (7.9) and (7.5) we obtain

$$A - a u_1 - b u_2 = \varepsilon_1 \frac{\partial^2 q_1}{\partial x^2} + q_1. \tag{7.41}$$

We write this in weak form, using a weight function w_i to give

$$\int_{\beta}^{\alpha} w_i A dx - \int_{\beta}^{\alpha} w_i a u_1\ dx - \int_{\beta}^{\alpha} w_i b u_2\ dx = \int_{\beta}^{\alpha} w_i \varepsilon_1 \frac{\partial^2 q_1}{\partial x^2}\ dx + \int_{\beta}^{\alpha} w_i q_1\ dx. \tag{7.42}$$

After integration by parts on the right-hand side we obtain

$$\int_{\beta}^{\alpha} w_i A\ dx - \int_{\beta}^{\alpha} w_i a u_1\ dx - \int_{\beta}^{\alpha} w_i b u_2\ dx = \varepsilon_1 \left[w_i \frac{\partial q_1}{\partial x} \right]_{\beta}^{\alpha} - \varepsilon_1 \int_{\beta}^{\alpha} \frac{\partial w_i}{\partial x}\frac{\partial q_1}{\partial x} dx + \int_{\beta}^{\alpha} w_i q_1\ dx. \tag{7.43}$$

We have zero flux external boundary conditions (7.11), so the first term on the right is equal to zero, leaving

$$\int_\beta^\alpha w_i A\ dx - \int_\beta^\alpha w_i a u_1\ dx - \int_\beta^\alpha w_i b u_2\ dx = -\varepsilon_1 \int_\beta^\alpha \frac{\partial w_i}{\partial x}\frac{\partial q_1}{\partial x} dx + \int_\beta^\alpha w_i q_1\ dx. \quad (7.44)$$

Equation (7.44) will give us q_1 in terms of u_1 and u_2. In exactly the same way, from (7.10) and (7.6) we obtain

$$\int_\beta^\alpha w_i B\ dx - \int_\beta^\alpha w_i a^* u_1\ dx - \int_\beta^\alpha w_i b^* u_2\ dx = -\varepsilon_2 \int_\beta^\alpha \frac{\partial w_i}{\partial x}\frac{\partial q_2}{\partial x} dx + \int_\beta^\alpha w_i q_2\ dx \quad (7.45)$$

which gives us q_2 in terms of u_1 and u_2. In order to solve (7.36) and (7.40) we will also require the rate of change of mass, $\dot{\theta}$, for each species. Equations (7.36) and (7.40) will provide these. Recalling that $\sum_i c_{p_i} = 1$, we sum over all i in equation (7.36) and obtain

$$\begin{aligned}\sum_i c_{1_i}\dot{\theta}_1 + \int_\alpha^{m(t)} \sum_i \left[u_1 \frac{\partial w_i}{\partial x}\frac{\partial \phi}{\partial x}\right] dx = - \int_\alpha^{m(t)} \sum_i \left[\delta_1 \frac{\partial w_i}{\partial x}\frac{\partial u_1}{\partial x}\right] dx + \sum_i w_i \delta_1 \left.\frac{\partial u_1}{\partial x}\right|_{m(t)} \\ + \int_\alpha^{m(t)} \sum_i \left[\frac{\partial w_i}{\partial x}\frac{\partial q_1}{\partial x}\right] dx + \int_\alpha^{m(t)} \sum_i [w_i r_1 u_1 (A - a u_1)]\ dx \quad (7.46)\end{aligned}$$

or

$$\dot{\theta}_1 = \delta_1 \left.\frac{\partial u_1}{\partial x}\right|_{m(t)} + \int_\alpha^{m(t)} r_1 u_1 (A - a u_1)\ dx \quad (7.47)$$

and for the sum over equation (7.36)

$$\dot{\theta}_2 = -\ \delta_2 \left.\frac{\partial u_2}{\partial x}\right|_{m(t)} + \int_{m(t)}^{\beta} r_2 u_2 (B - b^* u_2)\ dx. \quad (7.48)$$

We are now in a position to solve (7.36) and (7.40) for ϕ. We do not have an expression for the interface velocity that is neatly compatible with being an integral part of (7.36) and (7.40), as was possible with the Stefan model. However, we may obtain the interface velocity and then impose it as a Dirichlet condition on the velocity solution $\dot{x}$. At a given time step t_N we write the interface condition (7.13) in a finite difference form

$$\mu \delta_1 \frac{u^N_{1_{m(t)}} - u^N_{1_{m-1}}}{x_{m(t)} - x^N_{m-1}} = -\delta_2 \frac{u^N_{2_{m+1}} - u^N_{2_{m(t)}}}{x^N_{m+1} - x_{m(t)}} \quad (7.49)$$

where the subscript m denotes the interface node, and the x_i are the spatial co-ordinates of the nodes. We have that $u_{m(t)} = 0$, and so we can obtain an expression for the position of the interface node, $x_{m(t)}$,

$$x_{m(t)}^{N+1} = \frac{(\mu \delta_1 u_{1_{m-1}}^N x_{m+1}^N + \delta_2 u_{2_{m+1}}^N x_{m-1}^N)}{(\mu \delta_1 u_{1_{m-1}}^N + \delta_2 u_{2_{m+1}}^N)}. \tag{7.50}$$

We use the finite differences approximation to calculate the interface velocity

$$\dot{x}_{m(t)}^{N+1} = \frac{\left(\frac{(\mu \delta_1 u_{1_{m-1}}^N x_{m+1}^N + \delta_2 u_{2_{m+1}}^N x_{m-1}^N)}{(\mu \delta_1 u_{1_{m-1}}^N + \delta_2 u_{2_{m+1}}^N)} - x_{m(t)}^N\right)}{dt}. \tag{7.51}$$

This velocity can then be imposed on the interface when the velocity is recovered from ϕ. We return to our definition of ϕ (7.26), now written in distributed form,

$$\int_{R(t)} w_i \dot{x} dx = \int_{R(t)} w_i \frac{\partial \phi}{\partial x} dx. \tag{7.52}$$

This system of equations can be solved for $\dot{x}$. For the interface itself, we calculate the new position by correcting the interface condition at the prior time step. We obtain the resultant interface velocity by solving equation (7.51) with $u = 0$ imposed at the interface node. Having obtained $\dot{x}$, we move the domain using Euler integration. We also update θ_1 and θ_2 from $\dot{\theta}_1$ (7.47) and $\dot{\theta}_2$ (7.48) using the same time integration procedure. We may now recover u. We determine the constant partial masses c_{1_i} and c_{2_i} from (7.31) and (7.32) and the initial conditions. We obtain, for $t = 0$

$$c_{1_i} = \frac{1}{\theta_1(0)} \int_{\alpha}^{m(t)} w_i(x,0) u_1(x,0)\ dx \tag{7.53}$$

$$c_{2_i} = \frac{1}{\theta_2(0)} \int_{m(t)}^{\beta} w_i(x,0) u_2(x,0)\ dx. \tag{7.54}$$

Having obtained c_{1_i} and c_{2_i} and having moved the weight functions w_i with the domain, we also use (7.31) and (7.32) to recover u_1 and u_2. To do so we require θ_1 and θ_2 (from (7.47) and (7.48)) at the new time step. For species 1, u_1 can be recovered from

$$\int_{\alpha}^{m(t)} w_i(x,t) u_1(x,t)\ dx = c_{1_i} \theta_1(t) \tag{7.55}$$

and u_2 can be recovered from

$$\int_{m(t)}^{\beta} w_i(x,t)u_2(x,t)\ dx = c_{2_i}\theta_2(t). \tag{7.56}$$

In each case the Dirichlet condition that $u = 0$ at the interface is strongly imposed, and the Neumann condition at the external boundaries is also strongly imposed.

7.2.1 Construction of the finite element form

We solve this system for u using a finite element method. The boundary conditions are varied, but are all compatible with using the modified piecewise linear weight functions $w_i = \tilde{W}_i$ of 4.2.1, with a modified weight function at each external boundary, and also at each side of the interface. We have Dirichlet boundary conditions on the velocity (equation (7.52)) at both the interface and external boundaries. For the values of u_1 and u_2, given by equations (7.55) and (7.56), we have a Dirichlet condition at the interface only. At the external boundaries we have Neumann boundary conditions instead. However, all these conditions are compatible with strongly imposing the boundary values in the manner that results from the use of modified weight functions. This avoids the need to transfer between basis function definitions via the ALE equation, as we did for the Fisher's equation in Chapter 4. This is helpful because that particular step introduces an extra source of numerical error. We therefore strongly impose the values of the velocity and u_1 and u_2 at the interfaces and external boundaries. The values of u_1 and u_2 at the external boundaries can be transferred from their adjacent nodes because we have the Neumann condition.

We must begin by solving for q_1 and q_2. To construct the finite element form for this step, we select appropriate weight and basis functions. We do not require modified weight functions at this stage for several reasons. Firstly, we do not have Dirichlet conditions to impose on q_1 and q_2. Secondly, the solution of q_1 and q_2 is not part of the conservation-based ALE equation, so there is no necessity to use the same basis functions across both systems. Thirdly, in solving for q_1 and q_2 we treat the domain as a single system rather than separating into species specific domains, so the interface requires no special consideration. We define a weighted linear combination of the standard basis functions W_i for each of our variables. We do not repeat those definitions here but instead refer to Appendix A, equations

(A.1) to (A.4). We substitute those approximations into equation (7.44) and obtain

$$\begin{aligned}\int_{\alpha}^{\beta} W_i A\ dx - a\sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j\ dx\right] U_{1_j} - b\sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j\ dx\right] U_{2_j} \\ = -\varepsilon_1 \sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} \frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x} dx\right] Q_{1_j} + \sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j dx\right] Q_{1_j}.\end{aligned} \qquad (7.57)$$

In terms of our mass and stiffness matrices M and K, equation (7.57) can be rewritten as

$$M\underline{A} - aM\underline{U}_1 - bM\underline{U}_2 = -\varepsilon_1 K\underline{Q}_1 + M\underline{Q}_1. \qquad (7.58)$$

Here $\underline{A}$ is a vector with all entries equal to the resource parameter A. This construction is helpful should we later wish to incorporate spatial variability in resources. We may rewrite (7.58) in terms of $\underline{Q}_{1_j}$

$$\underline{Q}_1 = (-\varepsilon_1 K + M)^{-1} M(\underline{A} - a\underline{U}_1 - b\underline{U}_2). \qquad (7.59)$$

For species 2 we make the same piecewise linear approximations. We substitute the approximations (A.1) to (A.4) defined in Appendix A into (7.45) and obtain

$$\begin{aligned}\int_{\alpha}^{\beta} W_i B\ dx - a^*\sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j\ dx\right] U_{1_j} - b^*\sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j\ dx\right] U_{2_j} \\ = -\varepsilon_2 \sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} \frac{\partial W_i}{\partial x}\frac{\partial W_j}{\partial x} dx\right] Q_{2_j} + \sum_{j=0}^{N+1}\left[\int_{\alpha}^{\beta} W_i W_j dx\right] Q_{2_j}\end{aligned} \qquad (7.60)$$

which is, in matrix form

$$\underline{Q}_2 = (-\varepsilon_2 K + M)^{-1} M(\underline{B} - a^*\underline{U}_1 - b^*\underline{U}_2). \qquad (7.61)$$

Where $\underline{B}$ is a vector with all entries equal to the resource parameter B. Equations (7.59) and (7.61) can be solved to obtain $\underline{Q}_1$ and $\underline{Q}_2$. We now tackle the solution of u_1 and u_2, and must use the modified weight functions $\tilde{W}_i$ of Chapter 4 (figure 4.4) to do so. This will allow us to impose a velocity on the interface and on the external boundaries without violating the principle of relative conservation of mass. We take the relevant approximations from (A.1) to (A.14) and make substitutions as necessary into equations (7.36) and (7.40). We obtain the following form for equation (7.36),

$$\begin{aligned}\tilde{c}_{1_i}\dot{\theta}_1 + \sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)} U_1\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]\Phi_j = &-\sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)}\delta_1\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_{1_j}\\ &+\tilde{W}_i\delta_1\frac{\partial U_1}{\partial x}\bigg|_{m(t)} + \sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)} U_1\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]Q_{1_j}\\ &+\sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)} r_1A\tilde{W}_iW_j\ dx\right]U_{1_j} - \int_{\alpha}^{m(t)} r_1a\tilde{W}_iU_1^2\ dx\end{aligned} \tag{7.62}$$

where Z_i is the set of nodes populated by species i. Likewise equation (7.40) becomes

$$\begin{aligned}\tilde{c}_{2_i}\dot{\theta}_2 + \sum_{j\in Z_2}\left[\int_{m(t)}^{\beta} U_2\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]\Phi_j = &-\sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\delta_2\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_{2_j}\\ &-\tilde{W}_i\delta_2\frac{\partial U_2}{\partial x}\bigg|_{m(t)} + \sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\rho U_2\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]Q_{2_j}\\ &+\sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\tilde{W}_iW_jr_2Bdx\right]U_{2_j} - \int_{m(t)}^{\beta} r_2b^*\tilde{W}_iU_2^2dx.\end{aligned} \tag{7.63}$$

In matrix form (7.62) can be expressed as

$$\tilde{K}(\underline{U}_1)\ \ \underline{\Phi}_1 = \underline{\tilde{f}}_1. \tag{7.64}$$

Here $\tilde{K}(\underline{U}_1)$ is the weighted stiffness matrix of Chapter 3, section 3.1.2 constructed with the modified weight functions $\tilde{W}_i$, and $\underline{\Phi}_1$ is the vector containing the values of Φ_{1_j}, and $\underline{\tilde{f}}_1$ is a vector with entries $\tilde{f}_{1_i}$ given by

$$\begin{aligned}\tilde{f}_{1_i} = -\tilde{c}_{1_i}\dot{\theta}_1 &- \sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)}\delta_1\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_{1_j} + \tilde{W}_i\delta_1\frac{\partial U_1}{\partial x}\bigg|_{m(t)}\\ &+\sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)} U_1\frac{\partial \tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]Q_{1_j} + \sum_{j\in Z_1}\left[\int_{\alpha}^{m(t)} r_1A\tilde{W}_iW_j\ dx\right]U_{1_j} \qquad (7.65)\\ &-\int_{\alpha}^{m(t)} r_1a\tilde{W}_iU_1^2\ dx. \qquad (7.66)\end{aligned}$$

For species 2, (7.63) can be expressed as

$$\tilde{K}(\underline{U}_2)\underline{\Phi}_2 = \underline{\tilde{f}}_2 \tag{7.67}$$

with the vector $\underline{\tilde{f}}_2$ containing entries $\tilde{f}_{2_i}$ given by

$$\tilde{f}_{2_i} = -\tilde{c}_{2_i}\dot{\theta}_2 - \sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\delta_2\frac{\partial\tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]U_{2_j} - \tilde{W}_i\delta_2\frac{\partial U_2}{\partial x}\bigg|_{m(t)}$$
$$+\sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\rho U_2\frac{\partial\tilde{W}_i}{\partial x}\frac{\partial W_j}{\partial x}dx\right]Q_{2_j} + \sum_{j\in Z_2}\left[\int_{m(t)}^{\beta}\tilde{W}_iW_jr_2Bdx\right]U_{2_j} \tag{7.68}$$
$$-\int_{m(t)}^{\beta}r_2b^*\tilde{W}_iU_2^2dx. \tag{7.69}$$

The nonlinear terms $\int_{\alpha}^{m(t)} r_1a\tilde{W}_iU_1^2\ dx$ and $\int_{m(t)}^{\beta} r_2b^*\tilde{W}_iU_2^2dx$ can be calculated exactly using Simpson's rule (4.64). The matrix systems can be solved to obtain Φ_1 and Φ_2. Noting that the weighted stiffness matrices $\tilde{K}(\underline{U}_1)$ and $\tilde{K}(\underline{U}_2)$ are singular, we have an infinity of solutions available. We therefore set $\Phi = 0$ at any one node to obtain a single solution. Note that the expressions for $\dot{\theta}_1$ (7.47) and $\dot{\theta}_2$ (7.48) can be obtained and solved in a straightforward manner by simply summing over the rows of (7.64) and (7.67).
To recover $\dot{X}$, we use the approximation

$$\dot{X}(x,t) = \sum_{j=1}^{N}\dot{X}_j(t)W_j(x,t). \tag{7.70}$$

We substitute this into equation (7.52) to obtain the finite element form

$$\sum_{j\in Z_1\cup Z_2}\left[\int_{R(t)}\tilde{W}_iW_jdx\right]\dot{X}_j = \sum_{j=1}^{N}\left[\int_{R(t)}\tilde{W}_i\frac{\partial W_j}{\partial x}dx\right]\Phi_j. \tag{7.71}$$

In matrix form this is

$$\tilde{M}\underline{\dot{X}} = \tilde{B}\underline{\Phi}. \tag{7.72}$$

We impose $\dot{x} = 0$ on the external boundaries, and we impose the interface velocity obtained from (7.51). The use of modified basis functions means that we will not interfere with the conservation of mass by doing so. We may then solve (7.72) for the remaining velocities.

Time integration

We move the nodes using Euler's scheme. Using the same scheme, we update the values of θ_1 and θ_2 from the values of $\dot{\theta}_1$ (7.47) and $\dot{\theta}_2$ (7.48).

Obtaining the solution U_1 and U_2

We may now recover the values of U_1 and U_2. We can obtain U on the updated grid from the relative conservation of mass equations (7.55) and (7.56). After substitution for the finite element approximation, in matrix form these are

$$\tilde{M}_1 \underline{U}_1 = \underline{\tilde{c}}_{1_i} \theta_1(t) \tag{7.73}$$

and

$$\tilde{M}_2 \underline{U}_2 = \underline{\tilde{c}}_{2_i} \theta_2(t). \tag{7.74}$$

At the initial set up, we set $t = 0$ and use (7.73) and (7.74) to find the vectors $\underline{\tilde{c}}_{1_i}$ and $\underline{\tilde{c}}_{2_i}$. Then for each subsequent time step we move the mesh and recalculate $\theta_1(t)$ and $\theta_2(t)$. On the updated grid, we calculate the mass matrix $\tilde{M}(t)$. We can then obtain the updated $\underline{U}_1$ and $\underline{U}_2$ from inversions of (7.73) and (7.74) respectively.

Algorithm 15

The finite element solution of the competition problem given by equations (7.3) and (7.4) and with an interface condition given by (7.13) on the moving mesh in 1-D therefore consists of the following steps. We obtain the constant values of $\underline{\tilde{c}}_{1_i}$ and $\underline{\tilde{c}}_{2_i}$ from (7.73) and (7.74), and for each time step:

1. Find the velocity biases $\underline{Q}_1$ and $\underline{Q}_2$ from (7.59) and (7.61);
2. Find the velocity potential by solving equations (7.64) and (7.67) for the $\Phi_j(t)$ values;
3. Find the internal node velocity by solving equation (7.72) for the $\dot{X}_j(t)$ values;
4. Find the interface node velocity by solving equation (7.51) for the $\dot{X}_m(t)$ value;
5. Generate the co-ordinate system at the next time-step $t + dt$ by solving (3.18) using Euler's approximation
6. Update the values of θ_1 and θ_2 from the values of $\dot{\theta}_1$ (7.47) and $\dot{\theta}_2$ (7.48);
7. Find the solutions $U_1(t + dt)$ and $U_2(t + dt)$ by solving the conservation equations (7.73) and (7.74).

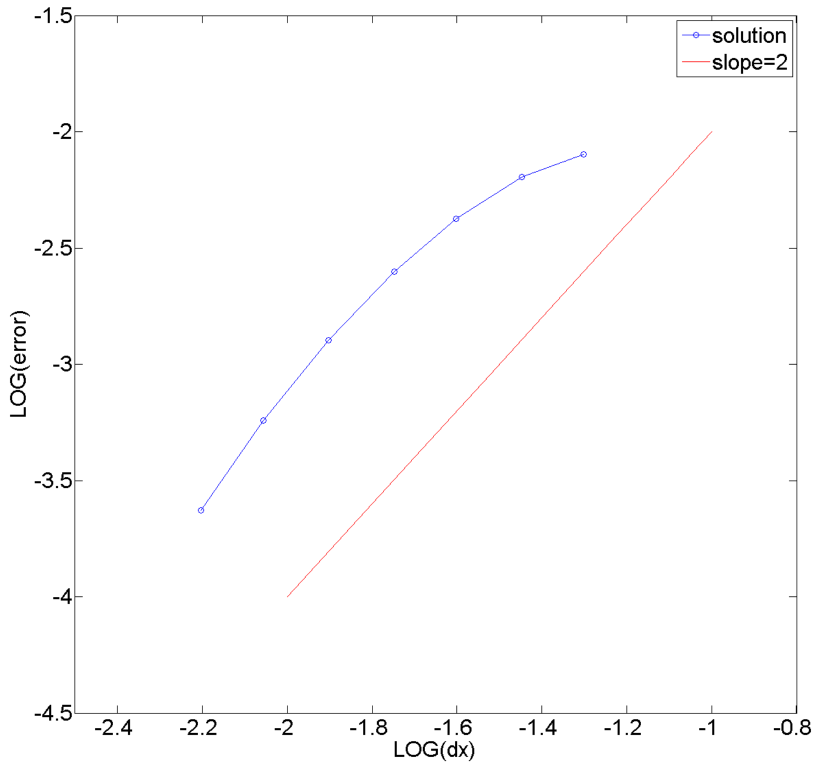


Fig. 7.2 Comparison of L^2 errors in the solution of algorithm 15. We observe an order of convergence of $p \approx 2$ in space, with time steps held constant at $\Delta t = 10^{-7}$.

7.2.2 Results

We find that the model is robust and the oscillations commonly found in finite element implementations, which are caused by the central differences approach, are minimal. Figure 7.2 shows convergence in the solution of approximately second order in space, as $\Delta x \to 0$ and with time steps held constant at $\Delta t = 10^{-7}$. This estimate is obtained by comparison of the result generated by each grid spacing with a high-resolution (641 node) result, since no absolute result is available. This order of convergence is as reported for the similar method in [8].

We are able to observe all the varied effects of diffusion, logistic growth or decline and aggregation, and we are also able to generate sensible interface movement. We use the parameters from [29] in order to be confident that the choices are sensible. We are able to make comparisons between the aggregating and non aggregating two-phase models. Figure 7.3 shows a non-aggregating model (with the q values set to zero); this is exactly equivalent to the competition diffusion model in Chapter 5 (5.2). With this choice of parameters, we observe no interface movement in the non aggregating model. The only development observed is in the shape of the solution near the interface, which is driven by diffusion. However, when we introduce aggregation, both species attempt to move away from the interface, resulting in a differently shaped solution (figure 7.6). We can see from figure 7.4 that the survivability index E_1 for species 1 is raised near the interface due to low population density, but then is very low in the domain occupied by species 2. We see in figure 7.5 how the q value takes a longer range average, so that despite the low population density near the interface, species 1 has an ideal velocity away from the interface. In figure 7.6 we observe that as both species vacate the area close to the interface, the changed interface dynamics favour species 2 and the interface moves to the left. Interestingly, in this particular scenario the increased 'intelligence' of the individuals does not help their longer term survival, because these additional movements cause mild overcrowding which offsets the reduced rate of competition at the interface. This suggests that the parameters given by [29] are potentially not the most representative, when this full model with the interface is constructed. With the large number of parameters at our disposal, the range of dynamics we could produce is limitless and very varied. We argue therefore that this model could be of real use to biologists in the field studying any spatially segregated competition system.

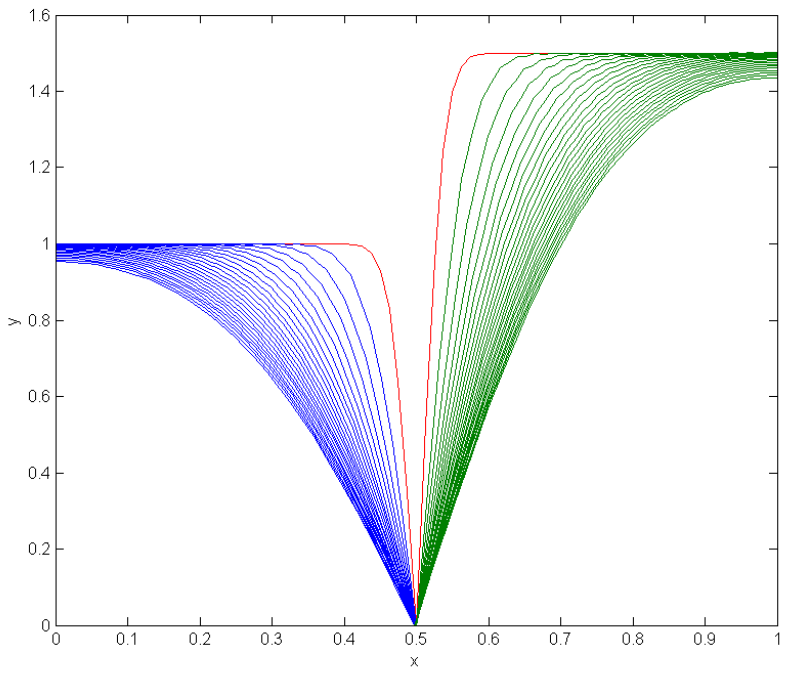


Fig. 7.3 The two phase competition model without aggregation at $t = 0.24$, using the parameters from Grindrod. Time steps are every 0.01s. We see stable population densities as the external boundaries, and an evolving shape to the interface.

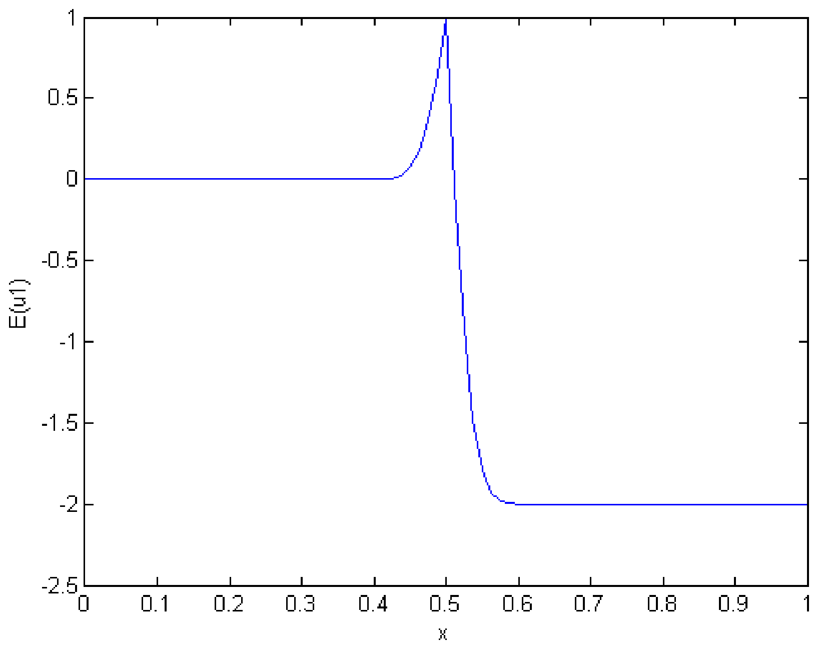


Fig. 7.4 The survivability index $E_1(u_1, u_2)$ for the two phase model with aggregation. The survivability index $E_1(u_1, u_2)$ is constructed so that the low populations around the interface are making it an attractive location for species 1.

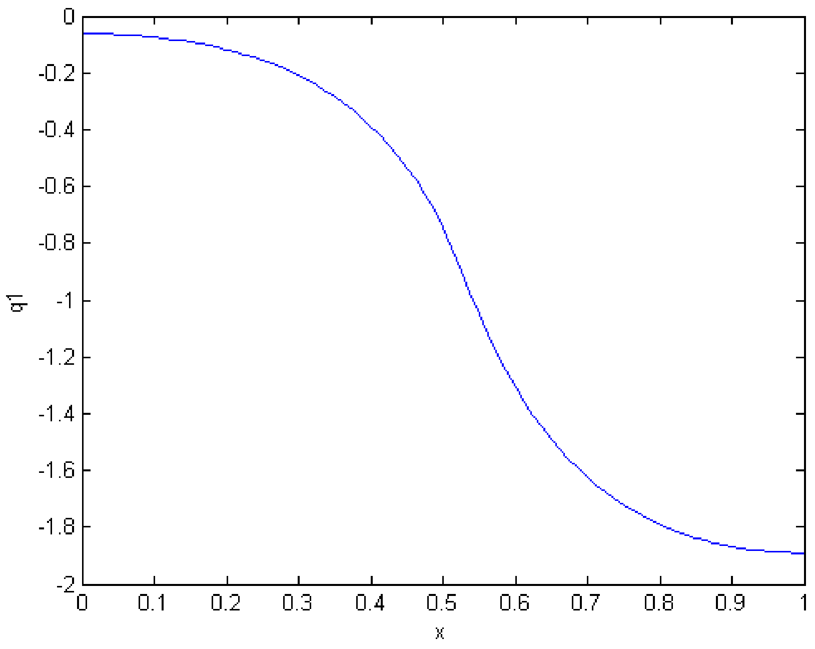


Fig. 7.5 The velocity bias q_1 for the two phase model with aggregation. The survivability index $E_1(u_1, u_2)$ is constructed so that the low populations around the interface are making it an attractive location for species 1, but the local averaging effect in q_1 allows the individuals to feel longer range effects. The domain populated by species 2 is unattractive enough to override the low population draw, and so species one will move to the left. We see this here in the negative gradient in q_1.

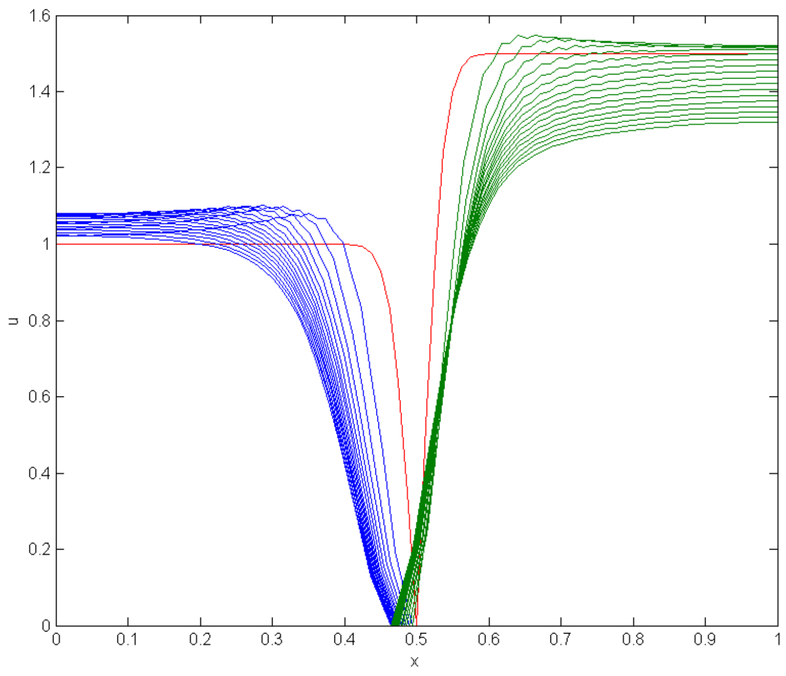


Fig. 7.6 Population decline in the two phase model with aggregation at $t = 0.16$. Time steps are every $0.01s$. We observe decreased movement towards the interface compared to the non-aggregating model. We see initially higher population densities a short distance away from the interface as the individuals resist moving towards it. The resulting overcrowding reduces overall survival rates, for this scenario.

7.3 2-D competition-aggregation-diffusion in a two phase model

We now consider the two dimensional version of the combination model in two phases, which is of additional interest because of the aggregating behaviour possible in 2-D. Reminding ourselves of the driving PDEs, we have

$$\frac{\partial u_1}{\partial t} = \delta_1 \nabla^2 u_1 - \nabla (u_1 \nabla q_1) + r u_1 E_1 \qquad t > 0, \mathbf{x} \in \Omega_1(t) \tag{7.75}$$

and

$$\frac{\partial u_2}{\partial t} = \delta_2 \nabla^2 u_2 - \rho \nabla (u_2 \nabla q_2) + r u_2 E_2 \qquad t > 0, \mathbf{x} \in \Omega_2(t). \tag{7.76}$$

We consider a fixed domain Ω bounded externally by S_e, but with Ω divided into two subdomain classes Ω_1 and Ω_2 which are separated by the moving interface(s) S_m. The boundaries S_1 and S_2 for each subdomain are therefore formed from $S_m(t)$ together with a time-variable section of S_e. We also have the survival behaviour given by

$$E_1 = A - au_1 - bu_2 \tag{7.77}$$

$$E_2 = B - a^* u_1 - b^* u_2. \tag{7.78}$$

The aggregating behaviour is defined by the PDEs

$$\varepsilon_1 \nabla^2 q_1 + q_1 = E_1 \tag{7.79}$$

$$\varepsilon_2 \nabla^2 q_2 + q_2 = E_2. \tag{7.80}$$

We have the Neumann boundary conditions given by

$$\begin{aligned} \nabla u \cdot \hat{\mathbf{n}} &= 0 & \mathbf{x} &\in S_e \\ \nabla q \cdot \hat{\mathbf{n}} &= 0 & \mathbf{x} &\in S_e \\ u &= 0 & \mathbf{x} &\in S_m. \end{aligned} \tag{7.81}$$

The high competition limit defined by Hilhorst [31] is given in 2-D as

$$\begin{aligned} u_1 &= 0 & \mathbf{x} &\in \Omega_2 \\ u_2 &= 0 & \mathbf{x} &\in \Omega_1. \end{aligned} \tag{7.82}$$

The interface condition is taken from [31], and is

$$\mu \delta_1 \nabla u_1 \cdot \hat{\mathbf{n}}_1 = \delta_2 \nabla u_2 \cdot \hat{\mathbf{n}}_2 \qquad x \in S_m \tag{7.83}$$

where $\hat{\mathbf{n}}_k$ is defined to be the outward pointing normal for the related species k, so that at the interface between species $\hat{\mathbf{n}}_1 = -\hat{\mathbf{n}}_2$. As in the 1-D case, μ, the interspecies competition rate is given by $\mu = aa^*/bb^*$. We use parameter choices from [29] which are given in Chapter 6, section 6.2.3. For the initial condition, we once more turn to the results of Chapter 6. We take the steady state solution of a shared domain model, and then introduce an asymmetry in the distribution of species 1. This asymmetry gives us the opportunity to explore the two dimensional nature of the model. These initial conditions are shown in figure 7.7.

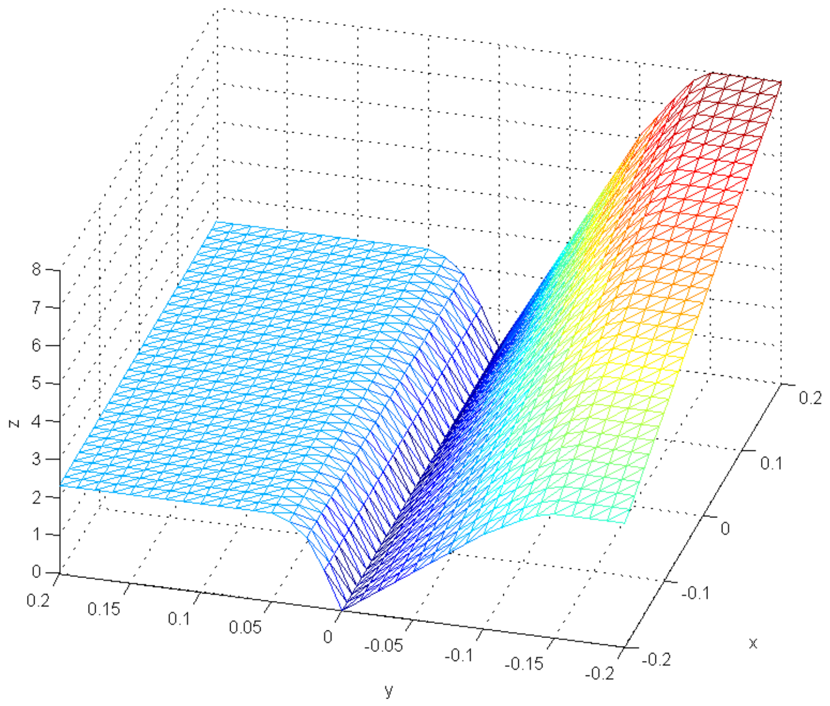


Fig. 7.7 Initial conditions for the 2-D competition-aggregation-diffusion model with two phases and a moving interface. We have introduced an asymmetry in the distribution of species 1 in order to fully explore the 2-D space.

The driving Lotka-Volterra based equations (7.75) and (7.76) are first rewritten in weak form, incorporating the weight function w_i,

$$\int_{\Omega_1(t)} w_i \frac{\partial u_1}{\partial t}\ d\Omega = \int_{\Omega_1(t)} \delta_1 w_i \nabla^2 u_1\ d\Omega - \int_{\Omega_1(t)} w_i \nabla\left(u_1 \nabla q_1\right)\ d\Omega + \int_{\Omega_1(t)} w_i r_1 u_1 E_1\ d\Omega \tag{7.84}$$

$$\int_{\Omega_2(t)} w_i \frac{\partial u_2}{\partial t}\ d\Omega = \int_{\Omega_2(t)} \delta_2 w_i \nabla^2 u_2\ d\Omega - \int_{\Omega_2(t)} w_i \rho \nabla\left(u_2 \nabla q_2\right)\ d\Omega + \int_{\Omega_2(t)} w_i r_2 u_2 E_2\ d\Omega. \tag{7.85}$$

We take the definitions for E_1 (7.77) and E_2 (7.78), and substitute them into (7.85). The terms containing b and a^* are equal to zero because we have no overlap between species.

We obtain

$$\int_{\Omega_1(t)} w_i \frac{\partial u_1}{\partial t}\ d\Omega = \int_{\Omega_1(t)} \delta_1 w_i \nabla^2 u_1 d\Omega - \int_{\Omega_1(t)} w_i \nabla \cdot (u_1 \nabla q_1)\ d\Omega \\ + \int_{\Omega_1(t)} w_i r_1 u_1 (A - a u_1)\ d\Omega \tag{7.86}$$

and

$$\int_{\Omega_2(t)} w_i \frac{\partial u_2}{\partial t}\ d\Omega = \int_{\Omega_2(t)} \delta_2 w_i \nabla^2 u_2\ d\Omega - \int_{\Omega_2(t)} w_i \rho \nabla \cdot (u_2 \nabla q_2)\ d\Omega \\ + \int_{\Omega_2(t)} w_i r_2 u_2 (B - b^* u_2)\ d\Omega. \tag{7.87}$$

We define the total population of a species k as θ_k, given by

$$\theta_k(t) = \int_{\Omega_k(t)} u_k\ d\Omega \tag{7.88}$$

where $\Omega_k(t)$ is the moving domain inhabited by that species. Since mass is not conserved in general, we will use the concept conserving relative mass. We write a relative conservation principle in terms of θ, introducing the weight function w_i,

$$\frac{1}{\theta_k(t)} \int_{\Omega_k(t)} w_i u_k\ d\Omega = c_{k_i} \tag{7.89}$$

or

$$\int_{\Omega_k(t)} w_i u_k\ d\Omega = c_{k_i} \theta(t) = c_{k_i} \int_{\Omega_k(t)} u_k\ d\Omega. \tag{7.90}$$

Here the constant c_{k_i} is determined by the choice of w_i, which should be chosen to provide a partition of unity. A distributed conservation of mass principle is now given by equation (7.90). Note that

$$\frac{d}{dt}\left[\int_{\Omega_k(t)} w_i u_k\ d\Omega\right] = c_{k_i} \dot{\theta}_k. \tag{7.91}$$

We differentiate (7.90) with respect to time using the Leibnitz integral rule on our moving frame $\Omega_k(t)$ to give

$$\frac{d}{dt}\left[\int_{\Omega_k(t)} w_i u_k\ d\Omega\right] = \int_{\Omega_k(t)} \left(\frac{\partial (w_i u_k)}{\partial t} + \nabla \cdot (w_i u_k \dot{\mathbf{x}})\right)\ d\Omega. \tag{7.92}$$

We require that the basis functions w_i move with the domain. Hence the basis functions also have velocity $\dot{\mathbf{x}}$ and therefore

$$\frac{\partial w_i}{\partial t} + \dot{\mathbf{x}} \cdot \nabla w_i = 0. \tag{7.93}$$

We obtain

$$\frac{d}{dt}\left[\int_{\Omega_k(t)} w_i u_k \; d\Omega\right] = \int_{\Omega_k(t)} w_i \left(\frac{\partial u_k}{\partial t} + \nabla \cdot (u_k \dot{\mathbf{x}})\right) \; d\Omega \tag{7.94}$$

or

$$\frac{d}{dt}\left[\int_{\Omega_k(t)} w_i u_k \; d\Omega\right] - \int_{\Omega_k(t)} w_i \nabla \cdot (u_k \dot{\mathbf{x}}) \; d\Omega = \int_{\Omega_k(t)} w_i \frac{\partial u_k}{\partial t} \; d\Omega. \tag{7.95}$$

We write this in terms of $\dot{\theta}_k$ and the constants c_{k_i} to give

$$c_{k_i}\dot{\theta}_k - \int_{\Omega_k(t)} w_i \nabla \cdot (u_k \dot{\mathbf{x}}) \; d\Omega = \int_{\Omega_k(t)} w_i \frac{\partial u_k}{\partial t} \; d\Omega. \tag{7.96}$$

We introduce the velocity potential ϕ, defined by

$$\dot{\mathbf{x}} = \nabla \phi \tag{7.97}$$

so that

$$c_{k_i}\dot{\theta}_k - \int_{\Omega_k(t)} w_i \nabla \cdot (u_k \nabla \phi) \; d\Omega = \int_{\Omega_k(t)} w_i \frac{\partial u_k}{\partial t} d\Omega \tag{7.98}$$

or, after integration by parts

$$c_{k_i}\dot{\theta}_k + \int_{\Omega_k(t)} u_k \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_k(t)} u_k w_i \nabla \phi \cdot \hat{\mathbf{n}}_k \; dS = \int_{\Omega_k(t)} w_i \frac{\partial u_k}{\partial t} \; d\Omega. \tag{7.99}$$

We turn our attention to each species individually. For species 1 equation (7.99) becomes

$$c_{1_i}\dot{\theta}_1 + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_1(t)} u_1 w_i \nabla \phi \cdot \hat{\mathbf{n}}_1 \; dS = \int_{\Omega_1(t)} w_i \frac{\partial u_1}{\partial t} d\Omega. \tag{7.100}$$

We substitute (7.86) to obtain

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_1(t)} u_1 w_i \nabla \phi \cdot \hat{\mathbf{n}}_1 \; dS &= \int_{\Omega_1(t)} \delta_1 w_i \nabla^2 u_1 \; d\Omega \\ - \int_{\Omega_1(t)} w_i \nabla \cdot (u_1 \nabla q_1) d\Omega &+ \int_{\Omega_1(t)} w_i r_1 u_1 (A - a u_1) \; d\Omega. \end{aligned} \tag{7.101}$$

Integration by parts on the right leads to

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_1(t)} u_1 w_i \nabla \phi \cdot \hat{\mathbf{n}}_1 \; dS = \\ - \int_{\Omega_1(t)} \delta_1 \nabla w_i \cdot \nabla u_1 d\Omega + \int_{S_1(t)} \delta_1 w_i \nabla u_1 \cdot \hat{\mathbf{n}}_1 \; dS + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla q_1 \; d\Omega \\ - \int_{S_1(t)} w_i u_1 \nabla q_1 \cdot \hat{\mathbf{n}}_1 \; dS + \int_{\Omega_1(t)} w_i r_1 u_1 (A - a u_1) \; d\Omega. \end{aligned} \tag{7.102}$$

We now consider the boundary conditions. We have zero Neumann boundary conditions (7.81) on q_1 and u_1 at the external boundary, and the Dirichlet boundary condition (7.82) on u_1 at the interface S_m. We also have fixed external boundaries, which mean that $\nabla \phi = 0$ on S_e. Certain of the boundary terms in (7.102) are therefore equal to zero along the external boundaries, and other boundary terms are equal to zero everywhere. We can reduce (7.102) to

$$\begin{aligned} c_{1_i}\dot{\theta}_1 + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla \phi d\Omega = - \int_{\Omega_1(t)} \delta_1 \nabla w_i \cdot \nabla u_1 \; d\Omega + \int_{S_m(t)} w_i \delta_1 \nabla u_1 \cdot \hat{\mathbf{n}}_1 \; dS \\ + \int_{\Omega_1(t)} u_1 \nabla w_i \cdot \nabla q_1 \; d\Omega + \int_{\Omega_1(t)} w_i r_1 u_1 (A - a u_1) \; d\Omega. \end{aligned} \tag{7.103}$$

For species 2, we begin once more with equation (7.99). This becomes

$$c_{2_i}\dot{\theta}_2 + \int_{\Omega_2(t)} u_2 \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_2(t)} u_2 w_i \nabla \phi \cdot \hat{\mathbf{n}}_2 \; dS = \int_{\Omega_2(t)} w_i \frac{\partial u_2}{\partial t} \; d\Omega. \tag{7.104}$$

We substitute (7.87) to obtain

$$\begin{aligned} c_{2_i}\dot{\theta}_2 + \int_{\Omega_2(t)} u_2 \nabla w_i \cdot \nabla \phi d\Omega - \int_{S_2(t)} u_2 w_i \nabla \phi \cdot \hat{\mathbf{n}}_2 \; dS = \int_{\Omega_2(t)} \delta_2 w_i \nabla^2 u_2 d\Omega \\ - \int_{\Omega_2(t)} w_i \rho \nabla \cdot (u_2 \nabla q_2) \; d\Omega + \int_{\Omega_2(t)} w_i r_2 u_2 (B - b^* u_2) \; d\Omega. \end{aligned} \tag{7.105}$$

Integration by parts on the right leads to

$$\begin{aligned} c_{2_i}\dot{\theta}_2 + \int_{\Omega_2(t)} u_2 \nabla w_i \cdot \nabla \phi \; d\Omega - \int_{S_2(t)} u_2 w_i \nabla \phi \cdot \hat{\mathbf{n}}_2 \; dS = \\ - \int_{\Omega_2(t)} \delta_2 \nabla w_i \cdot \nabla u_2 d\Omega + \int_{S_2(t)} w_i \delta_2 \nabla u_2 \cdot \hat{\mathbf{n}}_2 \; dS + \int_{\Omega_2(t)} \rho u_2 \nabla w_i \cdot \nabla q_2 \; d\Omega \\ + \int_{S_2(t)} w_i \rho u_2 \nabla q_2 \cdot \hat{\mathbf{n}}_2 \; dS + \int_{\Omega_2(t)} w_i r_2 u_2 (B - b^* u_2) \; d\Omega. \end{aligned} \tag{7.106}$$

After considering the boundary conditions (7.81) and (7.82) the remaining expression is

$$c_{2_i}\dot{\theta}_2 + \int_{\Omega_2(t)} u_1 \nabla w_i \cdot \nabla \phi \; d\Omega = -\int_{\Omega_2(t)} \delta_2 \nabla w_i \cdot \nabla u_2 d\Omega + \int_{S_m(t)} w_i \delta_2 \nabla u_2 \cdot \hat{\mathbf{n}}_2 \; dS$$
$$+ \int_{\Omega_2(t)} \rho u_2 \nabla w_i \cdot \nabla q_2 \; d\Omega + \int_{\Omega_2(t)} w_i r_2 u_2 (B - b^* u_2) \; d\Omega. \qquad (7.107)$$

If we have a given q_1, q_2, θ_1 and θ_2, we may solve (7.103) and (7.107) for ϕ . This will allow us to subsequently recover the nodal velocities. In order to obtain q_1 and q_2, we refer to equations (7.79) and (7.80), (7.77) and (7.78). Combining (7.77) and (7.79) we obtain

$$A - au_1 - bu_2 = \varepsilon_1 \nabla^2 q_1 + q_1. \qquad (7.108)$$

We write this in weak form, using a weight function w_i to give

$$\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega = \int_\Omega w_i \varepsilon_1 \nabla^2 q_1 \; d\Omega + \int_\Omega w_i q_1 \; d\Omega. \qquad (7.109)$$

After integration by parts on the right-hand side we obtain

$$\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega$$
$$= \varepsilon_1 \int_{S_e} w_i \nabla q_1 \cdot \hat{\mathbf{n}}_1 \; dS - \varepsilon_1 \int_\Omega \nabla w_i \cdot \nabla q_1 \; d\Omega + \int_\Omega w_i q_1 \; d\Omega. \qquad (7.110)$$

The first term on the right hand side is equal to zero due to the zero flux external boundary conditions (7.81), resulting in

$$\int_\Omega w_i A \; d\Omega - \int_\Omega w_i a u_1 \; d\Omega - \int_\Omega w_i b u_2 \; d\Omega = -\varepsilon_1 \int_\Omega \nabla w_i \cdot \nabla q_1 d\Omega + \int_\Omega w_i q_1 \; d\Omega. \quad (7.111)$$

Equation (7.111) will give us q_1 in terms of u_1 and u_2. In exactly the same way, from (7.78) and (7.80) we obtain

$$\int_\Omega w_i B \; d\Omega - \int_\Omega w_i a^* u_1 \; d\Omega - \int_\Omega w_i b^* u_2 \; d\Omega = -\varepsilon_2 \int_\Omega \nabla w_i \cdot \nabla q_2 \; d\Omega + \int_\Omega w_i q_2 \; d\Omega \qquad (7.112)$$

which gives us q_2 in terms of u_1 and u_2. We note that in order to solve (7.103) and (7.107) we also require the rate of change of mass, $\dot{\theta}$, for each species. This can be obtained by summing over all w_i equations (7.103) and (7.107) as appropriate. With a choice of w_i forming a partition of unity, and recalling that $\sum_i c_{p_i} = 1$, we obtain for the sum over all w_i

in equation (7.103),

$$\begin{aligned}\sum_i c_{1_i}\dot{\theta}_1 + \int_{\Omega_1(t)} \sum_i [u_1 \nabla w_i \cdot \nabla \phi]\ d\Omega = \\ -\int_{\Omega_1(t)} \sum_i [\delta_1 \nabla w_i \cdot \nabla u_1]\ d\Omega + \int_{S_m(t)} \sum_i w_i \delta_1 \nabla u_1 \cdot \hat{\mathbf{n}}_1\ dS \\ + \int_{\Omega_1(t)} \sum_i [\nabla w_i \cdot \nabla q_1]\ d\Omega + \int_{\Omega_1(t)} \sum_i [w_i r_1 u_1 (A - a u_1)]\ d\Omega \end{aligned} \tag{7.113}$$

or

$$\dot{\theta}_1 = \int_{S_m(t)} \delta_1 \nabla u_1 \cdot \hat{\mathbf{n}}_1\ dS + \int_{\Omega_1(t)} r_1 u_1 (A - a u_1)\ d\Omega \tag{7.114}$$

and for the sum over equation (7.107)

$$\dot{\theta}_2 = \int_{S_m(t)} \delta_2 \nabla u_2 \cdot \hat{\mathbf{n}}_2\ dS + \int_{\Omega_2(t)} r_2 u_2 (B - b^* u_2)\ d\Omega. \tag{7.115}$$

We are now in a position to solve (7.103) and (7.107) for ϕ, and hence recover the nodal velocities $\dot{\mathbf{x}}$. However, we will also require a velocity for the interface which will need to be recovered from (7.83) using a separate numerical approximation. The method is similar to the 1-D version. For each point on the interface m for which we require a velocity, we take a 1-D cross-section across the interface. The plane of the cross section is the plane defined by the interface normal $\hat{\mathbf{n}}_1$ (or its opposite, $\hat{\mathbf{n}}_2$) at the point of interest, and the vertical axis $\hat{\mathbf{z}}$ (see figure 7.9). We may then obtain the interface velocity from a finite differences approximation of (7.83). We begin by rewriting (7.83) with a common normal $\hat{\mathbf{n}}_1$, defined as the normal to the interface that points from species 1 to species 2. Equation (7.83) becomes

$$\mu \delta_1 \nabla u_1 \cdot \hat{\mathbf{n}}_1 = -\delta_2 \nabla u_2 \cdot \hat{\mathbf{n}}_1. \tag{7.116}$$

Along the line $\hat{\mathbf{n}}_1$ we write, for a given time step t_N, the finite difference form

$$\mu \delta_1 \left(\frac{u_{1_m}^N - u_{1_{m-1}}^N}{\mathbf{x}_m - \mathbf{x}_{m-1}^N} \right) \cdot \hat{\mathbf{n}}_1 = -\delta_2 \left(\frac{u_{2_{m+1}^N} - u_{2_m}^N}{\mathbf{x}_{m+1}^N - \mathbf{x}_m} \right) \cdot \hat{\mathbf{n}}_1 \tag{7.117}$$

where the subscript m denotes the interface, and the $\mathbf{x}_i$ are the spatial co-ordinates of points i along m. We have that $u_m = 0$, and so we can obtain an expression for the position of the interface node, $\mathbf{x}_m$,

$$\mathbf{x}_m^{N+1} \cdot \hat{\mathbf{n}}_1 = \left(\frac{(\mu \delta_1 u_{1_{m-1}}^N \mathbf{x}_{m+1}^N + \delta_2 u_{2_{m+1}}^N \mathbf{x}_{m-1}^N)}{(\mu \delta_1 u_{1_{m-1}}^N + \delta_2 u_{2_{m+1}}^N)} \right) \cdot \hat{\mathbf{n}}_1. \tag{7.118}$$

Note that the point under consideration only has a velocity determined in the direction $\hat{\mathbf{n}}_1$, and so we will constrain its movement to that vector only. We use the finite differences approximation to calculate the interface velocity

$$\mathbf{x}_m^{N+1} \cdot \hat{\mathbf{n}}_1 = \frac{\left(\frac{(\mu \delta_1 u^N_{1_{m-1}} \mathbf{x}^N_{m+1} + \delta_2 u^N_{2_{m+1}} \mathbf{x}^N_{m-1})}{(\mu \delta_1 u^N_{1_{m-1}} + \delta_2 u^N_{2_{m+1}})} - \mathbf{x}^N_m \right) \cdot \hat{\mathbf{n}}_1}{dt}. \tag{7.119}$$

Note that for the solution of (7.119) we require a known solution u at a known position $\mathbf{x}$ a short distance either side of the interface (positions $m+1$ and $m-1$). These are obtained by calculating a spline interpolation through u at a set distance away from the interface. This distance is chosen to be less than the smallest expected node spacing. For the purposes of this experimental implementation we have chosen $1/10$ of the initial (regular) grid spacing. The spline interpolation is cubic and is performed in three stages (see figure 7.8) .

1. The lines of nodes running parallel to the interface are considered. A cubic spline is used to interpolate values of u along the adjacent three rows of nodes on either side of the interface.

2. The normal to the interface at a selected node is considered. Discrete values of u along this normal are assigned from the intersections with the interpolated parallel rows.

3. A second cubic spline is used to interpolate values of u along this normal to the interface. The value of u at the set distance from the interface is thus approximated.

Figure 7.9 shows the lines (red) for which values of u need to be calculated. The points requiring u to be approximated through this method are defined by the intersection of these red lines with the normal to the interface at every interface node.

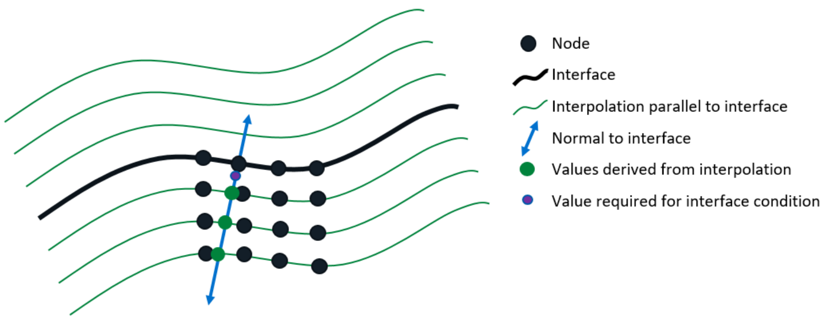

Node
Interface
Interpolation parallel to interface
Normal to interface
Values derived from interpolation
Value required for interface condition

Fig. 7.8 Diagram showing positions of cubic splines. Green lines are splines through nodes adjacent to the interface. Green spots are values extracted using those splines for positions along the normal to the interface. A second spline through the green spots allows the required value (purple spot) to be approximated.

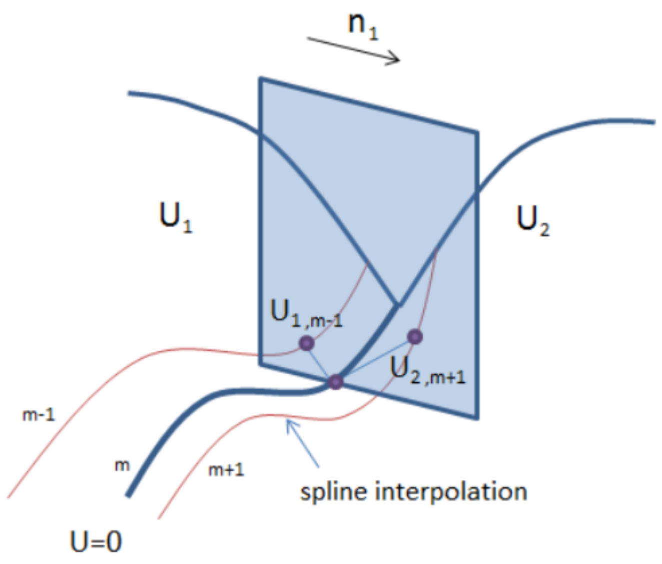


Fig. 7.9 Diagram showing how the interface condition is calculated in 2-D. The blue plane is the plane in which the calculation is performed. The purple spots are interpolated positions and values of u. The central spot is a point on the interface which may only move in the normal direction given by the vector $\hat{\mathbf{n}}_1$.

We calculate the velocity at any point on the interface using (7.119). The velocities elsewhere (and not on the interface) are recovered from ϕ. We obtain this velocity field from the distributed form of the definition of ϕ (7.97), which is

$$\int_{\Omega_k(t)} w_i \dot{\mathbf{x}} d\Omega = \int_{\Omega_k(t)} w_i \nabla\phi \ d\Omega \tag{7.120}$$

which we can solve for $\dot{\mathbf{x}}$ with the interface velocity strongly imposed. Having obtained $\dot{\mathbf{x}}$, we move the domain and together with it, all the weight functions w_i. We also update θ_1 and θ_2 from $\dot{\theta}_1$ (7.114) and $\dot{\theta}_2$ (7.115). We may now recover u on the modified domain. Using the initial conditions, we calculate the constant partial mass fractions c_{1_i} and c_{2_i} from (7.89). We obtain, for $t = 0$

$$c_{1_i} = \frac{1}{\theta_1(0)} \int_{\Omega_1} w_i(x,0) u_1(x,0) \ d\Omega \tag{7.121}$$

$$c_{2_i} = \frac{1}{\theta_2(0)} \int_{\Omega_2} w_i(x,0) u_2(x,0) \ d\Omega. \tag{7.122}$$

We also use (7.89) to recover u_1 and u_2 on the new domain. To do so we require θ_1 and θ_2 from (7.114) and (7.115) at the new time step. For species 1, u_1 can be recovered from

$$\int_{\Omega_1(t)} w_i(x,t) u_1(x,t) \ d\Omega = c_{1_i} \theta_1(t) \tag{7.123}$$

and u_2 can be recovered from

$$\int_{\Omega_2(t)} w_i(x,t) u_2(x,t) \ d\Omega = c_{2_i} \theta_2(t). \tag{7.124}$$

Exactly as in the 1-D case, the Dirichlet conditions prescribing that $u = 0$ at the interface are strongly imposed, and the Neumann conditions at the external boundaries are also strongly imposed.

7.3.1 Construction of the finite element form

We solve this system for u using a finite element method. We choose modified basis functions at both internal and external boundaries; the argument for that choice is identical to the argument for the same choice in the 1-D case so we will not repeat it here. Instead we refer to section 7.2.1 for the detailed reasoning. As in the 1-D case we are not required to transfer between basis systems. We use the 2-D modified piecewise linear weight functions $w_i = \tilde{W}_i$ described in Chapter 4 (figure 4.11), and we strongly impose the values of both the velocity and of u_1 and u_2 at the interfaces and external boundaries. The values of u_1 and u_2

at the external boundaries can be approximated from their near neighbours because we have Neumann conditions in place.

Before we are able to solve for u_1 and u_2, we require a finite element approximation to obtain a solution for q_1 and q_2 as the first step. We define an approximation to each of our variables in terms of the standard basis functions W_i. We do not require modified basis functions at this stage for the same reasons given for the 1-D case. We do not repeat those definitions here but instead refer to Appendix A, equations (A.1) to (A.4). We substitute those approximations into equation (7.111) and obtain

$$\begin{aligned}\int_\Omega W_i A \; d\Omega - a\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] U_{1_j} - b\sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] U_{2_j} \\ = -\varepsilon_1 \sum_{j=1}^{N}\left[\int_\Omega \nabla W_i \cdot \nabla W_j d\Omega\right] Q_{1_j} + \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] Q_{1_j}. \qquad (7.125)\end{aligned}$$

We may write (7.125) in terms of our mass and stiffness matrices M and K to obtain

$$M\underline{A} - aM\underline{U}_{1_j} - bM\underline{U}_{2_j} = \varepsilon_1 K\underline{Q}_{1_j} + M\underline{Q}_{1_j}. \qquad (7.126)$$

which we may rewrite in terms of $\underline{Q}_{1_j}$

$$\underline{Q}_1 = (-\varepsilon_1 K + M)^{-1} M(\underline{A} - a\underline{U}_1 - b\underline{U}_2). \qquad (7.127)$$

Here $\underline{A}$ is a vector with all entries equal to A. In exactly the same manner, we substitute the approximations (A.1) to (A.4) defined in Appendix A into (7.112) and obtain

$$\int_\Omega W_i B \; d\Omega - a^* \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] U_{1_j} - b^* \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] U_{2_j} \qquad (7.128)$$

$$= -\varepsilon_2 \sum_{j=1}^{N}\left[\int_\Omega \nabla W_i \cdot \nabla W_j \; d\Omega\right] Q_{2_j} + \sum_{j=1}^{N}\left[\int_\Omega W_i W_j \; d\Omega\right] Q_{2_j} \qquad (7.129)$$

which is, in matrix form

$$\underline{Q}_2 = (-\varepsilon_2 K + M)^{-1} M(\underline{B} - a^*\underline{U}_1 - b^*\underline{U}_2), \qquad (7.130)$$

where $\underline{B}$ is a vector with all entries equal to B. We can now recover $\underline{Q}_1$ and $\underline{Q}_2$ by solving equations (7.127) and (7.130). We now consider the ALE system which will allow us to obtain U_1 and U_2. For this system we must use the modified weight functions $\tilde{W}_i$ of Chapter 4 (figure 4.11). This will allow us to obey the principle of relative conservation of mass and yet impose a velocity on the interface and on the external boundaries. We take equations (7.103) and (7.107) and substitute into them the approximations (A.1) to (A.14) as necessary. We obtain the following, with all variables now expressed in terms of their piecewise linear approximations. Equation (7.103) becomes

$$\begin{aligned}\tilde{c}_{1_i}\dot{\theta}_1 + \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} U_1\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]\Phi_j &= -\sum_{j\in Z_1}\left[\int_{\Omega_1(t)} \delta_1\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]U_{1_j} \\ &+ \int_{S_m(t)} \tilde{W}_i\delta_1\nabla U_1\cdot\hat{\mathbf{n}}_1 \; dS + \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} U_1\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]Q_{1_j} \\ &+ \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} r_1A\tilde{W}_iW_j \; d\Omega\right]U_{1_j} - \int_{\Omega_1(t)} r_1a\tilde{W}_iU_1^2 \; d\Omega \end{aligned} \tag{7.131}$$

and equation (7.107) becomes

$$\begin{aligned}\tilde{c}_{2_i}\dot{\theta}_2 + \sum_{j\in Z_2}\left[\int_{\Omega_2(t)} U_2\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]\Phi_j &= -\sum_{j\in Z_2}\left[\int_{\Omega_2(t)} \delta_2\nabla\tilde{W}_i\nabla W_j \; d\Omega\right]U_{2_j} \\ &+ \int_{S_m(t)} \tilde{W}_i\delta_2\nabla U_2\cdot\hat{\mathbf{n}}_2 \; dS + \sum_{j\in Z_2}\left[\int_{\Omega_2(t)} \rho U_2\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]Q_{2_j} \\ &+ \sum_{j\in Z_2}\left[\int_{\Omega_2(t)} \tilde{W}_iW_jr_2B \; d\Omega\right]U_{2_j} - \int_{\Omega_2(t)} r_2b^*\tilde{W}_iU_2^2 \; d\Omega \end{aligned} \tag{7.132}$$

where Z_i is the set of nodes populated by species i. In matrix form (7.131) can be expressed as

$$\tilde{K}(\underline{U}_1)\;\underline{\Phi}_1 = \underline{\tilde{f}}_1 \tag{7.133}$$

where $\tilde{K}(\underline{U}_1)$ is the weighted stiffness matrix of Chapter 3, section 3.1.3, constructed with the modified basis functions $\tilde{W}_i$, and $\underline{\Phi}_1$ is the vector containing the values of Φ_{1_j}, and $\underline{\tilde{f}}_1$ is a vector with entries $\tilde{f}_{1_i}$ given by

$$\begin{aligned}\tilde{f}_{1_i} = -\tilde{c}_{1_i}\dot{\theta}_1 &- \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} \delta_1\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]U_{1_j} \\ &+ \int_{S_m(t)} \tilde{W}_i\delta_1\nabla U_1\cdot\hat{\mathbf{n}}_1 \; dS + \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} U_1\nabla\tilde{W}_i\cdot\nabla W_j \; d\Omega\right]Q_{1_j} \\ &+ \sum_{j\in Z_1}\left[\int_{\Omega_1(t)} r_1A\tilde{W}_iW_j \; d\Omega\right]U_{1_j} - \int_{\Omega_1(t)} r_1a\tilde{W}_iU_1^2 \; d\Omega. \end{aligned} \tag{7.134}$$

Similarly, (7.132) can be expressed as

$$\tilde{K}(\underline{U}_2)\underline{\Phi}_2 = \underline{\tilde{f}}_2 \tag{7.135}$$

with the vector $\underline{\tilde{f}}_2$ containing entries $\tilde{f}_{2_i}$ given by

$$\begin{aligned}\tilde{f}_{2_i} = {} & -\tilde{c}_{2_i}\dot{\theta}_2 - \sum_{j\in Z_2}\left[\int_{\Omega_2(t)}\delta_2\nabla\tilde{W}_i\nabla W_j\ d\Omega\right]U_{2_j} \\ & + \int_{S_m(t)}\tilde{W}_i\delta_2\nabla U_2\cdot\hat{\mathbf{n}}_2\ dS + \sum_{j\in Z_2}\left[\int_{\Omega_2(t)}\rho U_2\nabla\tilde{W}_i\cdot\nabla W_j\ d\Omega\right]Q_{2_j} \\ & + \sum_{j\in Z_2}\left[\int_{\Omega_2(t)}\tilde{W}_iW_jr_2B\ d\Omega\right]U_{2_j} - \int_{\Omega_2(t)}r_2b^*\tilde{W}_iU_2^2\ d\Omega. \end{aligned} \tag{7.136}$$

The nonlinear terms in (7.134) and (7.136) can be calculated using Gaussian quadrature (see Appendix B). We can now obtain Φ_1 and Φ_2 by solving these matrix systems. Since the weighted stiffness matrices $\tilde{K}(\underline{U}_1)$ and $\tilde{K}(\underline{U}_2)$ are singular, we have an infinity of solutions available and we set $\nabla\Phi\cdot\hat{\mathbf{n}}_k = 0$ at all external boundary nodes to obtain a single solution, where $\hat{\mathbf{n}}_k$ is the normal to the boundary for either species $k\in[1,2]$. Note that summing over the rows of (7.133) and (7.135) will give the expressions for $\dot{\theta}_1$ (7.114) and $\dot{\theta}_2$ (7.115).
To recover $\dot{\mathbf{X}}$, we use the approximation

$$\dot{\mathbf{X}}(\mathbf{x},t) = \sum_{j\in Z_1\cup Z_2}\dot{\mathbf{X}}_j(t)W_j(\mathbf{x},t). \tag{7.137}$$

To obtain the finite element form, we substitute this into equation (7.120),

$$\sum_{j\in Z_1\cup Z_2}\left[\int_{\Omega_k(t)}\tilde{W}_iW_j\ d\Omega\right]\dot{\mathbf{X}}_j = \sum_{j\in Z_1\cup Z_2}\left[\int_{\Omega_k(t)}\tilde{W}_i\nabla W_j\ d\Omega\right]\Phi_j. \tag{7.138}$$

In matrix form this is

$$\tilde{M}\underline{\dot{\mathbf{X}}} = \tilde{B}\underline{\Phi}. \tag{7.139}$$

We impose $\dot{\mathbf{x}}.\hat{\mathbf{n}}_k = 0$ on the external boundaries. We impose the interface velocity obtained from (7.119). Since we are using modified basis functions we will not interfere with the conservation of relative mass by doing so. We solve (7.139) for the remaining velocities.

Time integration

We move the nodes using Euler's scheme. Using the same scheme, we update the values of θ_1 and θ_2 from the values of $\dot{\theta}_1$ (7.114) and $\dot{\theta}_2$ (7.115).

Obtaining the solution U_1 and U_2

We may now recover the values of U_1 and U_2. The relative conservation of mass equations (7.123) and (7.124) allow us to obtain U on the updated grid. We substitute the familiar piecewise linear approximations (A.1) and (A.2) into (7.123) and (7.124), and obtain the matrix forms

$$\tilde{M}_1 \underline{U}_1 = \underline{\tilde{c}}_{1_i} \theta_1(t) \tag{7.140}$$

and

$$\tilde{M}_2 \underline{U}_2 = \underline{\tilde{c}}_{2_i} \theta_2(t). \tag{7.141}$$

We begin by setting $t = 0$ and using (7.140) and (7.141) to find the constants $\tilde{c}_{1_i}$ and $\tilde{c}_{2_i}$. Then for each subsequent time step we proceed as follows. Having moved the grid, we take θ_1 and θ_2 at the new time step. We calculate the mass matrix $\tilde{M}$ for the updated grid. We can then obtain the updated $\underline{U}_1$ and $\underline{U}_2$ from inversions of (7.140) and (7.141) respectively.

Algorithm 16

The finite element solution of the competition problem given by equations (7.75) and (7.76) and with an interface condition given by (7.83) on the moving mesh in 2-D therefore consists of the following steps. We obtain the constant values of c_{1_i} and c_{2_i} from (7.140) and (7.141), and for each time step:

1. Find the velocity biases $\underline{Q}_1(t)$ and $\underline{Q}_2(t)$ from (7.127) and (7.130);
2. Find the velocity potential by solving equations (7.133) and (7.135) for the $\Phi_j(t)$ values;
3. Find the internal node velocity by solving equation (7.139) for the $\dot{\mathbf{X}}_j(t)$;
4. Find the interface node velocity by solving equation (7.119) for the $\dot{\mathbf{X}}_m(t)$ value;
5. Generate the co-ordinate system at the next time-step $t + dt$ by solving (3.18) using Euler's approximation;

6. Update the values of $\theta_1(t+dt)$ and $\theta_2(t+dt)$ from the values of $\dot{\theta}_1(t)$ (7.114) and $\dot{\theta}_2(t)$ (7.115);

7. Find the solutions $U_1(t+dt)$ and $U_2(t+dt)$ by solving the conservation equations (7.140) and (7.141).

7.3.2 Results

The model is implemented in MATLAB on a square domain with 33 nodes along each side. We are able to produce plausible behaviour. We are able to observe the moving interface exhibiting different behaviour at different points along its length, according to the population dynamics either side (see figure 7.10). The interface moves according to condition derived from the high competition limit, and the population densities adjacent to the interface are subject to significant increases or decreases because of this motion. Unfortunately, we run into the problem of internal node tangling at the point when more interesting behaviour begins to emerge. Figures 7.11 and 7.12 shown the state of the system shortly before this occurs. This is likely to be a fundamental weakness of this complex implementation of the MMFEM. The MMFEM keeps the node order and connectivity intact, no matter how much movement is occurring, and so cannot easily cope with highly distorted grids. In this particular MMFEM, we have an interface condition which is only indirectly related to the dynamics of the majority of the system. The interface is free to make large and sudden movnements because the calculation of its velocity takes place separately to that of the velocities elsewhere. This freedom has the potential to have a negative impact on the stability of the rest of the domain. It may be possible to find a set of parameters which are more stable. It is certainly possible to run a steady state system but it is of little interest. However, to make useful progress from the point we have reached, the sensible approach would be to further research the interface condition from both an ecological and a mathematical perspective. In this way it may be possible to find a construction that could be completely integrated into the MMFEM, in the manner demonstrated for the Stefan model of Chapter 5.

7.3.3 Further work

For the 2-D, two phase, MMFEM model of Lotka-Volterra competition with diffusion and aggregation, the implementation suffers from node tangling. This may be addressed by some of the existing methods of easing tangling, such as adding a viscosity term or introducing repellant forces between nodes. Alternatively, a smarter way of incorporating the interface condition may help. From an ecological perspective, there may be a different, simpler interface condition that we could use, which may ease this difficulty.

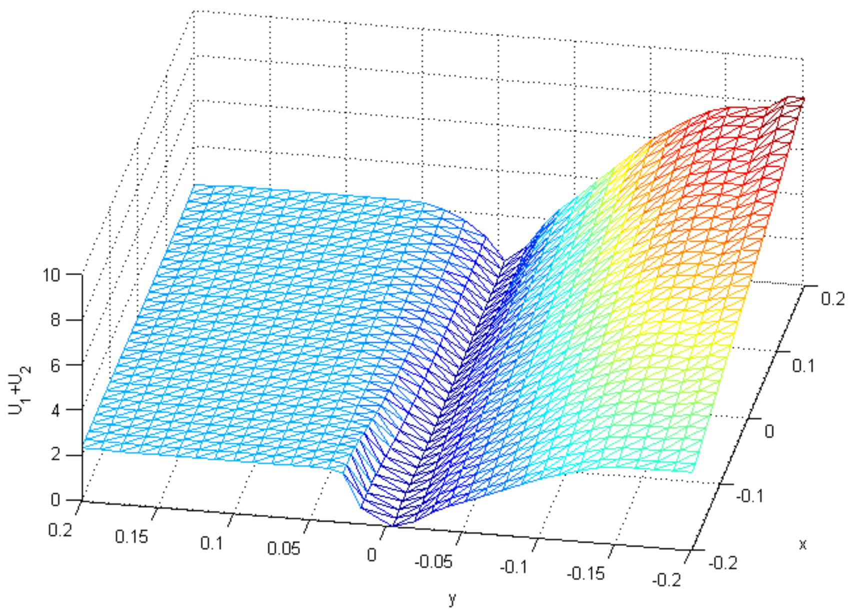


Fig. 7.10 The solution of the 2-D competition-aggregation-diffusion model in two phases with a moving interface at $t = 2 \times 10^{-5}$. The sum of both species is plotted, although they are segregated completely with the population consisting of only species 1 to the left of the interface and only species 2 to the right of the interface. We observe heterogenous movement of the interface, which no longer aligns with $y = 0$. We observe a small building of population density adjacent to it (near $y = 0.05, x = -0.2$) as a result. The parameters used are $\delta_1 = \delta_2 = 0.01$, $k_1 = 1$, $k_2 = 1/3$, $r_1 = r_2 = 1$, $A = 1.5$, $B = 2$, $a = 1$, $b = 2$, $a^* = 3$ $b^* = 1$, and $\varepsilon_1 = \varepsilon_2 = 0.001$.

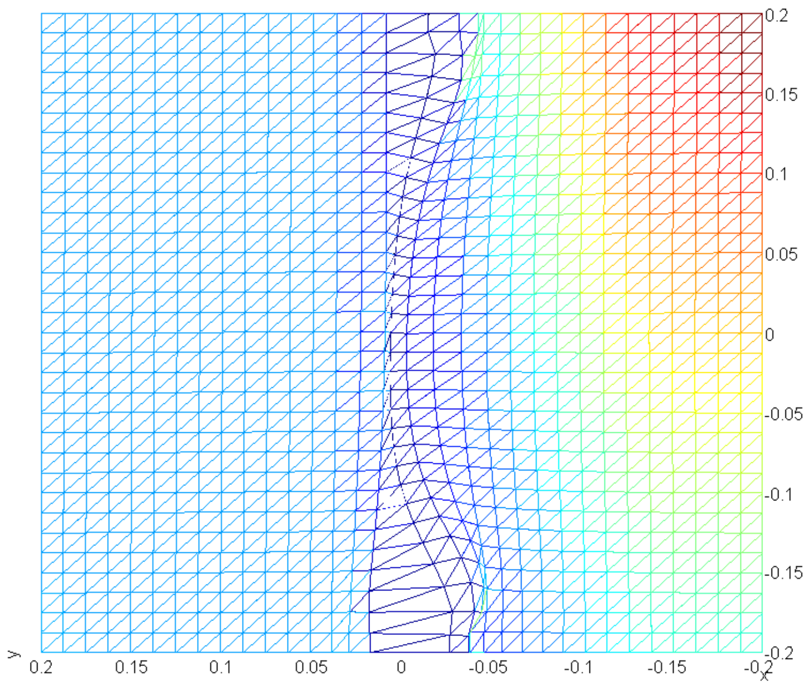


Fig. 7.11 The solution of the 2-D competition-aggregation-diffusion model in two phases with a moving interface at $t = 2.3 \times 10^{-5}$. The heterogenous movement of the interface can be clearly seen in this plan view. The parameters used are $\delta_1 = \delta_2 = 0.01$, $k_1 = 1$, $k_2 = 1/3$, $r_1 = r_2 = 1$, $A = 1.5$, $B = 2$, $a = 1$, $b = 2$, $a^* = 3$ $b^* = 1$, and $\varepsilon_1 = \varepsilon_2 = 0.001$.

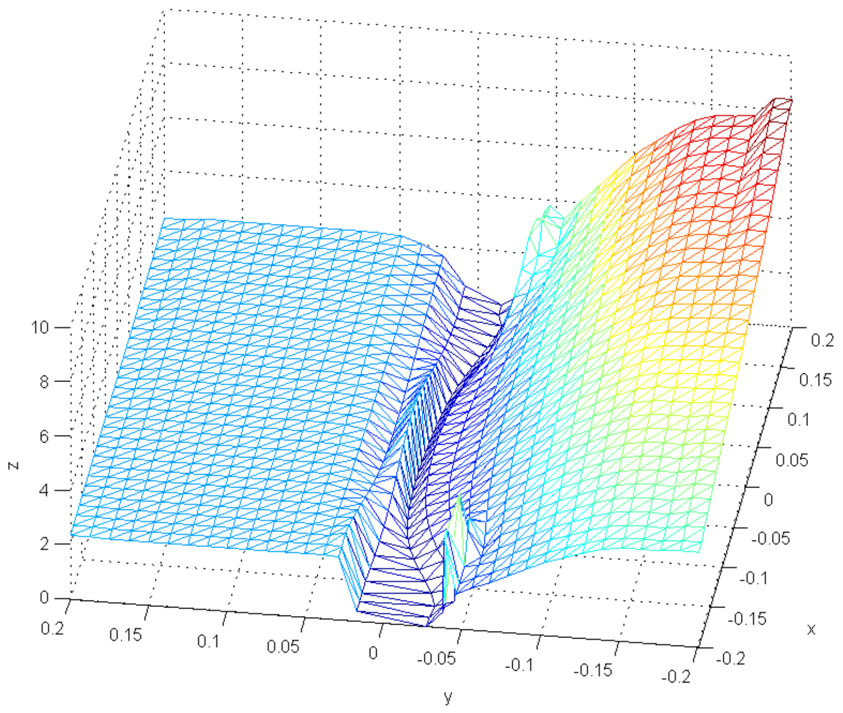


Fig. 7.12 The solution of the 2-D competition-aggregation-diffusion model in two phases with a moving interface at $t = 2.3 \times 10^{-5}$. The heterogenous movement of the interface has produced three distinct areas of high population density. Node tangling occurs soon after. The parameters used are $\delta_1 = \delta_2 = 0.01$, $k_1 = 1$, $k_2 = 1/3$, $r_1 = r_2 = 1$, $A = 1.5$, $B = 2$, $a = 1$, $b = 2$, $a^* = 3$ $b^* = 1$, and $\varepsilon_1 = \varepsilon_2 = 0.001$.

Chapter 8

Summary

We will now summarize the material covered in this thesis and discuss the next steps for this research.

In Chapters 1 and 2, we introduced the concept of moving mesh methods and outlined the various approaches. We discussed the history and development of a variety of velocity-based methods which formed a pathway towards the moving mesh finite element method. We then introduced the Lotka-Volterra competition equations.

In Chapter 3, we outlined, in general terms, the process for applying the MMFEM in either 1 or 2 dimensions. We examined the existing body of work performed using the MMFEM, looking at both classic examples and others that require modifications to the method.

In Chapter 4, we began to demonstrate new applications for moving mesh methods. We illustrated the equidistribution method with a model of a column of water undergoing wind sheer at the surface, and which is also subject to Coriolis forces. We then illustrated the MMFEM, applying it to the Fisher's equation of blow-up or combustion. We built this model in both 1 and 2 dimensions, and with both a free and fixed boundary. We compared the fixed boundary model to a finite difference implementation, and found that we were able to resolve the blow up peak at a higher, narrower stage. For the free boundary case we used modified basis functions, and considered how best to construct a stiffness matrix in terms of the modified basis functions. For this model, we also made a switch between basis systems using the ALE form. We found that we resolved a higher peak, but we lost accuracy in the time at which the blow-up occurs. We also applied the MMFEM to the Keller-Segel model, which has both a substrate and a reactant, building this model in 2-D. We found that the accuracy of this model is dependent on the shapes of the triangles in the mesh, which is determined by the initial node distribution. For the better node distributions, the MMFEM outperforms a radially symmetric finite difference model, but for poorly distributed initial grids the finite difference model is the better choice.

In Chapter 5, we examined moving interface models, or two phase models. We reconstructed the MMFEM for the Stefan problem, and we introduced a simplification of the method compared to the previous work. We were able to replicate the results given visually in previous work. We then applied this method for the first time to a Lotka-Volterra competition system with a high competition limit, so that the species are completely spatially segregated. As far as we know this was the first attempt at a numerical model of this particular system. We used an interface condition based on this high competition limit, which required a novel implementation. This is one of the most successful models in the thesis, producing interesting and realistic behaviour in a very stable model. We were able to implement the model with a wide variety of creative parameter combinations, and observed various effects dominating in turn as the populations evolve through time.

In Chapter 6, we turned again to population models, and in particular we examined the introduction of an aggregating term to the Lotka-Volterra competition equations. We began with a single species interacting with only itself, and built a MMFEM in 1 and 2 dimensions. For the first time we were able to model the effects of the aggregation term in 2-D, and we did so with a moving mesh. We then turned to systems with two competing species. We built a non-moving, shared domain finite-element model in 2-D in order to understand the behaviour of the system. The first model was a mass-conserving example. We then varied the model to look at cases where the population is non-conservative, and also where we have a change in the resource space. We were able to observe the development of the system from a random dispersion of individuals, through to separate clusters containing a single species each. In the conservative case, we saw that large and small groupings survived through to an approximate steady state solution. By contrast in the non-conservative case, only the larger groupings survived. At steady state these models produced segregated species separated by an interface with approximately zero population density, which supports the conceptual basis for developing a two-phase model for this system.

In Chapter 7, we combined the approaches and subject matter in chapters 5 and 6 into a single population model. We used the two phase methodology from Chapter 5, including the interface condition based on the high competition limit. Over the whole domain, we modelled the aggregating behaviour from Chapter 6, which allows some intelligent determinism by the individuals. We combined this together with the more traditional diffusion and logistic behaviour of the standard Lotka-Volterra models within each phase. The 1-D implementation of this was successful. The 2-D implementation produced plausible behaviour, but suffers from node tangling whenever more dynamic mesh movements are produced.

The potential exists for useful further work to extend the research in this thesis. In particular, the population models and simulations in chapters 6 and 7 are novel, and are also suitable for application to real-world situations. There are three useful dimensions for further work.

- Validate existing models. It would be extremely interesting to compare the behaviour of the models against an empirical data set. The models easily lend themselves to adaptions in the sizes and shapes of the domains, alterations to the logistic terms and of course changes to parameters, without the need for any further calculations. This adaptability means there are a wide range of potential biological and ecological systems which we could now test the models on, and a first priority would be to find one or more data sets suitable for validating the model behaviour. A data set for a species which shows competition-diffusion-aggregation behaviour (to validate the new model in chapter 7) should be a particular aim of the search.

- Deploy existing models. The models in chapter 6 are also already suitable for tackling certain questions, such as how changes in the resource space might alter behaviour. For example, the impact could be modelled of a road or railway that divides a domain.

- Collaborate with domain experts. The further research should focus on collaboration to understand the particular modelling requirements of real-world systems which can be described in a similar manner to our model. The aim should be to understand these requirements from both a mathematical and value perspective. The subsequent development work should be in the direction of the research requirements of those ecological systems which would most benefit from a study which has access to this modelling capability.

Appendix A

Piecewise linear approximations

We make the following piecewise linear approximations in terms of the standard basis functions W_i, which may be defined in 1 or 2 dimensions.

$$U_1(x,t) = \sum_{j=0}^{N+1} W_j(x,t) U_{1_j}(t) \text{ (in 1D)} \qquad U_1(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) U_{1_j}(t) \text{ (in 2D)} \tag{A.1}$$

$$U_2(x,t) = \sum_{j=0}^{N+1} W_j(x,t) U_{2_j}(t) \text{ (in 1D)} \qquad U_2(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) U_{2_j}(t) \text{ (in 2D)} \tag{A.2}$$

$$Q_1(x,t) = \sum_{j=0}^{N+1} W_j(x,t) Q_{1_j}(t) \text{ (in 1D)} \qquad Q_1(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) Q_{1_j}(t) \text{ (in 2D)} \tag{A.3}$$

$$Q_2(x,t) = \sum_{j=0}^{N+1} W_j(x,t) Q_{2_j}(t) \text{ (in 1D)} \qquad Q_2(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) Q_{2_j}(t) \text{ (in 2D)} \tag{A.4}$$

$$U_S(x,t) = \sum_{j=0}^{N+1} W_j(x,t) U_{S_j}(t) \text{ (in 1D)} \qquad U_S(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) U_{S_{j(t)}} \text{ (in 2D)} \tag{A.5}$$

$$U_L(x,t) = \sum_{j=0}^{N+1} W_j(x,t) U_{L_j}(t) \text{ (in 1D)} \qquad U_L(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) U_{L_j}(t) \text{ (in 2D)} \tag{A.6}$$

$$\Phi(x,t) = \sum_{j=0}^{N+1} W_j(x,t) \Phi_j(t) \text{ (in 1D)} \qquad \Phi(\mathbf{x},t) = \sum_{j=1}^{N} W_j(\mathbf{x},t) \Phi_j(t) \text{ (in 2D)} \tag{A.7}$$

with derivatives

$$\frac{\partial U_1}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} U_{1_j} \text{ (in 1D)} \qquad \nabla U_1 = \sum_{j=1}^{N} \nabla W_j U_{1_j} \text{ (in 2D)} \tag{A.8}$$

$$\frac{\partial U_2}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} U_{2_j} \text{ (in 1D)} \qquad \nabla U_2 = \sum_{j=1}^{N} \nabla W_j U_{2_j} \text{ (in 2D)} \tag{A.9}$$

$$\frac{\partial Q_1}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} Q_{1_j} \text{ (in 1D)} \qquad \nabla Q_1 = \sum_{j=1}^{N} \nabla W_j Q_{1_j} \text{ (in 2D)} \tag{A.10}$$

$$\frac{\partial Q_2}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} Q_{2_j} \text{ (in 1D)} \qquad \nabla Q_2 = \sum_{j=1}^{N} \nabla W_j Q_{2_j} \text{ (in 2D)} \tag{A.11}$$

$$\frac{\partial U_S}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} U_{S_j} \text{ (in 1D)} \qquad \nabla U_S = \sum_{j=1}^{N} \nabla W_j U_{S_j} \text{ (in 2D)} \tag{A.12}$$

$$\frac{\partial U_L}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} U_{L_j} \text{ (in 1D)} \qquad \nabla U_L = \sum_{j=1}^{N} \nabla W_j U_{L_j} \text{ (in 2D)} \tag{A.13}$$

$$\frac{\partial \Phi}{\partial x} = \sum_{j=0}^{N+1} \frac{\partial W_j}{\partial x} \Phi_j \text{ (in 1D)} \qquad \nabla \Phi = \sum_{j=1}^{N} \nabla W_j \Phi_j \text{ (in 2D)} \tag{A.14}$$

Appendix B

Gaussian quadrature

For a nonlinear term such as

$$f_i = \int_{\omega_e} W_i U^2 \; d\Omega \tag{B.1}$$

we may write the contribution to f_i from a triangular element ω_e corresponding to the weight function W_i as

$$\int_{\omega_e} W_i U^2 \; d\Omega = \int_{\omega_e} W_i \left(\sum_{j=1}^{3} U_j W_j \right)^2 \; d\Omega. \tag{B.2}$$

The right hand side may be evaluated using three point Gaussian quadrature. This is exact for quadratics and has a higher order of accuracy than the approximation of any of the PDEs in this thesis, so will not affect the numerical accuracy obtainable. Suitable sets of weights and integration points are widely published, for example in [42]. A neat choice uses the same piecewise linear weight functions W_i as are used throughout this work, those of figure 3.3. The locations of the integration points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ which correspond to these weights are given in figure B.1. The values of U at these points can be calculated from the values of U at the vertices $\mathbf{x}_A$, $\mathbf{x}_B$ and $\mathbf{x}_C$ as follows,

$$U_1 = U(\mathbf{x}_1) = \frac{2U(\mathbf{x}_A)}{3} + \frac{U(\mathbf{x}_B)}{6} + \frac{U(\mathbf{x}_C)}{6} \tag{B.3}$$

$$U_2 = U(\mathbf{x}_2) = \frac{U(\mathbf{x}_A)}{6} + \frac{2U(\mathbf{x}_B)}{3} + \frac{U(\mathbf{x}_C)}{6} \tag{B.4}$$

$$U_3 = U(\mathbf{x}_3) = \frac{U(\mathbf{x}_A)}{6} + \frac{U(\mathbf{x}_B)}{6} + \frac{2U(\mathbf{x}_C)}{3}. \tag{B.5}$$

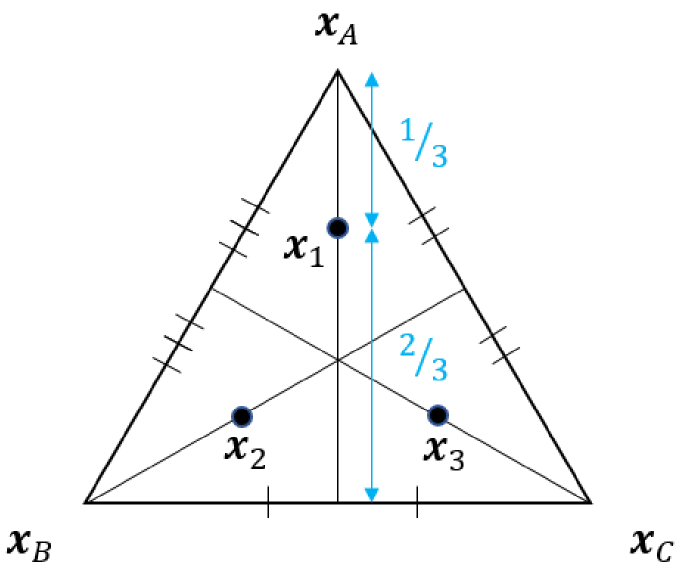


Fig. B.1 Location of integration points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ for three point Gaussian quadrature for triangle with vertices at $\mathbf{x}_A$, $\mathbf{x}_B$ and $\mathbf{x}_C$. Each integration point lies $1/3$ of the way along the line connecting a vertex to the midpoint of the opposite edge.

Selecting piecewise linear weight function $W_i = W_1$ centered at node A, we can now calculate (B.2) as follows

$$\int_{\omega_e} W_1 \left(\sum_{j=1}^{3} U_j W_j \right)^2 \, d\Omega = \frac{Area_e}{3} \left(\frac{2U_1^2}{3} + \frac{U_2^2}{6} + \frac{U_3^2}{6} \right). \tag{B.6}$$

Likewise, if $W_i = W_2$ centered at node B, (B.2) becomes

$$\int_{\omega_e} W_2 \left(\sum_{j=1}^{3} U_j W_j \right)^2 \, d\Omega = \frac{Area_e}{3} \left(\frac{U_1^2}{6} + \frac{2U_2^2}{3} + \frac{U_3^2}{6} \right) \tag{B.7}$$

and if $W_i = W_3$ centered at node C, (B.2) becomes

$$\int_{\omega_e} W_3 \left(\sum_{j=1}^{3} U_j W_j \right)^2 \, d\Omega = \frac{Area_e}{3} \left(\frac{U_1^2}{6} + \frac{U_2^2}{6} + \frac{2U_3^2}{3} \right). \tag{B.8}$$

References

[1] Adjerid, S. and Flaherty, J. (1986). A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 23(4):778–796.

[2] Baines, M. (1994). *Moving finite elements*. Oxford University Press, Inc.

[3] Baines, M. (1998). Grid adaptation via node movement. *Applied Numerical Mathematics*, 26(1):77–96.

[4] Baines, M. (2015). Explicit time-stepping for moving meshes. *Journal of Mathematical Study*, 48(2):93–105.

[5] Baines, M., Hubbard, M., and Jimack, P. (2005). A moving mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries. *Applied Numerical Mathematics*, 54(3):450–469.

[6] Baines, M., Hubbard, M., and Jimack, P. (2011). Velocity-based moving mesh methods for nonlinear partial differential equations. *Communications in Computational Physics*, 10(3):509–576.

[7] Baines, M., Hubbard, M., Jimack, P., and Jones, A. (2006). Scale-invariant moving finite elements for nonlinear partial differential equations in two dimensions. *Applied Numerical Mathematics*, 56(2):230–252.

[8] Baines, M., Hubbard, M., Jimack, P., and Mahmood, R. (2009). A moving-mesh finite element method and its application to the numerical solution of phase-change problems. *Communications in Computational Physics*, 6(3):595–624.

[9] Baines, M. and Lee, T. (2014). A large time-step implicit moving mesh scheme for moving boundary problems. *Numerical Methods for Partial Differential Equations*, 30(1):321–338.

[10] Bird, N. (2014). *A moving-mesh method for high order nonlinear diffusion*. PhD thesis, University of Reading, Department of Mathematics.

[11] Bonan, B., Baines, M., Nichols, N., and Partridge, D. (2016). A moving-point approach to model shallow ice sheets: a study case with radially symmetrical ice sheets. *The Cryosphere*, 10(1):1–14.

[12] Brenner, S. and Scott, R. (2007). *The mathematical theory of finite element methods*, volume 15. Springer Science & Business Media.

[13] Budd, C., Carretero-González, R., and Russell, R. (2005). Precise computations of chemotactic collapse using moving mesh methods. *Journal of Computational Physics*, 202(2):463–487.

[14] Budd, C., Chen, J., Huang, W., and Russell, R. (1996). Moving mesh methods with applications to blow-up problems for pdes. *Pitman Research Notes in Mathematics Series (1996)*, pages 1–18.

[15] Budd, C., Huang, W., and Russell, R. (2009). Adaptivity with moving grids. *Acta Numerica*, 18:111–241.

[16] Budd, C. and Williams, J. (2006). Parabolic Monge–Ampère methods for blow-up problems in several spatial dimensions. *Journal of Physics A: Mathematical and General*, 39(19):5425.

[17] Cao, W., Huang, W., and Russell, R. (2002). A moving mesh method based on the geometric conservation law. *SIAM Journal on Scientific Computing*, 24(1):118–142.

[18] Cao, W., Huang, W., and Russell, R. (2003). Approaches for generating moving adaptive meshes: location versus velocity. *Applied Numerical Mathematics*, 47(2):121–138.

[19] Carlson, N. and Miller, K. (1998a). Design and application of a gradient-weighted moving finite element code i: In one dimension. *SIAM Journal on Scientific Computing*, 19(3):728–765.

[20] Carlson, N. and Miller, K. (1998b). Design and application of a gradient-weighted moving finite element code ii: In two dimensions. *SIAM Journal on Scientific Computing*, 19(3):766–798.

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

[22] Conway, E. and Smoller, J. (1977). Diffusion and the predator-prey interaction. *SIAM Journal on Applied Mathematics*, 33(4):673–686.

[23] Courant, R. (1994). Variational methods for the solution of problems of equilibrium and vibrations. *Lecture Notes in Pure and Applied Mathematics*, pages 1–1.

[24] Edgington, M. (2011). Moving mesh methods for semi-linear problems. Master's thesis, University of Reading, Department of Mathematics.

[25] Ekman, V. W. (1905). On the influence of the earth's rotation on ocean currents. *Ark. Mat. Astron. Fys.*, 2:1–53.

[26] Gilpin, M. (1973). Do hares eat lynx? *The American Naturalist*, 107(957):727–730.

[27] Griffiths, D. and Reed College (1999). *Introduction to electrodynamics*, volume 3. prentice Hall Upper Saddle River, NJ.

[28] Grindrod, P. (1988). Models of individual aggregation or clustering in single and multi-species communities. *Journal of Mathematical Biology*, 26(6):651–660.

[29] Grindrod, P. (1991). *Patterns and waves: The theory and applications of reaction-diffusion equations*. Oxford University Press, USA.

[30] Heun, K. (1900). Neue methoden zur approximativen integration der differentialgleichungen einer unabhängigen veränderlichen. *Z. Math. Phys*, 45:23–38.

[31] Hilhorst, D., Mimura, M., and Schätzle, R. (2003). Vanishing latent heat limit in a stefan-like problem arising in biology. *Nonlinear analysis: real world applications*, 4(2):261–285.

[32] Huang, W., Ren, Y., and Russell, R. (1994). Moving mesh partial differential equations (mmpdes) based on the equidistribution principle. *SIAM Journal on Numerical Analysis*, 31(3):pp. 709–730.

[33] Hubbard, M., Baines, M., and Jimack, P. (2009). Consistent Dirichlet boundary conditions for numerical solution of moving boundary problems. *Applied Numerical Mathematics*, 59(6):1337–1353.

[34] Keller, E. and Segel, L. (1971). Model for chemotaxis. *Journal of Theoretical Biology*, 30(2):225–234.

[35] Larsson, S. and Sanz-Serna, J.-M. (1994). The behavior of finite element solutions of semilinear parabolic problems near stationary points. *SIAM journal on numerical analysis*, 31(4):1000–1018.

[36] Lee, T., Baines, M., and Langdon, S. (2015). A finite difference moving mesh method based on conservation for moving boundary problems. *Journal of Computational and Applied Mathematics*, 288:1–17.

[37] Lotka, A. (1920). Analytical note on certain rhythmic relations in organic systems. *Proceedings of the National Academy of Sciences*, 6(7):410–415.

[38] Miller, K. and Miller, R. (1981). Moving finite elements. i. *SIAM Journal on Numerical Analysis*, 18(6):pp. 1019–1032.

[39] Miller, K. (1981). Moving finite elements. ii. *SIAM Journal on Numerical Analysis*, 18(6):pp. 1033–1057.

[40] Partridge, D. (2013). *Numerical modelling of glaciers: Moving meshes and data assimilation*. PhD thesis, University of Reading, Department of Mathematics.

[41] Reddy, J. (1993). *An introduction to the finite element method*, volume 2. McGraw-Hill New York.

[42] Reddy, J. (2014). *An Introduction to Nonlinear Finite Element Analysis: with applications to heat transfer, fluid mechanics, and solid mechanics*. OUP Oxford.

[43] Reynolds, O., Brightmore, A., and Moorby, W. (1903). *The sub-mechanics of the universe*, volume 3. University Press.

[44] Robertson, N. (2006). A moving Lagrangian mesh model of a lava dome volcano and talus slope. Master's thesis, University of Reading, Department of Mathematics.

[45] Strang, G. and Fix, G. (1973). *An analysis of the finite element method*, volume 212. Prentice-hall Englewood Cliffs, NJ.

[46] Turner, M., Clough, R., Martin, H., and Topp, L. (1956). A technique for the precise measurement of small fluid velocities. *J. Fluid Mech*, 23(9):805–823.

[47] Volterra, V. (1926). Fluctuations in the abundance of a species considered mathematically. *Nature*, 118:558–560.

[48] Volterra, V. (1928). Variations and fluctuations of the number of individuals in animal species living together. *J. Cons. Int. Explor. Mer*, 3(1):3–51.

[49] Weizhang, H. and Russell, R. (2010). *Adaptive moving mesh methods*, volume 174. Springer Science & Business Media.

[50] Wells, B. (2004). *A moving mesh finite element method for the numerical solution of partial differential equations and systems*. PhD thesis, University of Reading, Department of Mathematics.

[51] Wells, B., Baines, M., and Glaister, P. (2005). Generation of arbitrary lagrangian–eulerian (ale) velocities, based on monitor functions, for the solution of compressible fluid equations. *International Journal for Numerical Methods in Fluids*, 47(10-11):1375–1381.

[52] White, Jr, A. B. (1979). On selection of equidistributing meshes for two-point boundary-value problems. *SIAM Journal on Numerical Analysis*, 16(3):472–502.

[53] Zhang, J. and Du, Q. (2009). Numerical studies of discrete approximations to the allen-cahn equation in the sharp interface limit. *SIAM Journal on Scientific Computing*, 31(4):3042–3063.