THE UNIVERSITY OF READING

Department of Mathematics and Statistics

Modelling time-dependent partial di□erential equations

ase on c-0.0186694(o)0.0689466(n)0.118

Declaration

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

Tamsin Lee

Acknowledgments

I would like to thank EPRSC for funding my studies. I extend that gratitude to Reading University Maths Department who organised everything from the funding to endless monitoring meetings, especially Sue Davis, the heart of the department. I thank Peter Grindrod and Paul Glaister for making my monitoring meetings a comfortable environment to discuss my work and help see the direction of my thesis. An additional thanks to Peter Grindrod for his contributions to my thesis. I am also grateful to Marcus Tindall for the aid of his enthusiastic insight into tumour modelling that has been invaluable. In addition, I thank Matthew Hubbard for suggesting the tumour growth problem.

The most important gratitude is for my supervisors, Mike Baines and Steve Langdon, for their continuing patience, and ensuring that studying a Ph.D. with them was an enjoyable experience. However, a Ph.D. is not all about fun, so I am additionally grateful that Mike and Steve have resulted in me continually achieving work I am proud of. I am particularly grateful to Mike, but trying to express the enormity of my gratitude would be like trying to divide by zero.

A big thanks to Bonhi for being a `writing-up-veteran' that I could turn to frequently for advice, and for injecting some enthusiasm and pride into my work when it was waning. I also thank Sanita for the delight of sharing an office with her

Table of Variables

Symbol	Meaning
t	Time variable
x	Independent space variable
L	Spatial di erential operator that conserves mass
G	Spatial di erential operator that does not conserve mass
H	Spatial di erential operator that conserves mass
$S(x;t)$	Source term
$u(x;t)$	The solution to a patial di erential operator
$a(t), b(t)$	The boundary positions of the solution
x_j , $j = 0;1;:::;N$	The mesh with N nodes
t	Time o

List of Figures

List of Tables

and general classes of time-dependent problems [4, 48, 72].

There are three main types of grid adaptation, the most common type being h-refinement, which uses a static mesh and adds or removes nodes to or from the existing mesh, resulting in local refinement or coarsening of the mesh. Another is p-refinement, where a finite element discretisation of the PDE is used with local polynomials, in which the order of the polynomials is increased or decreased to adapt the method according to the smoothness of the solutions, often as measured by error estimates. It is common to combine these two methods to give hp-methods. However, hp-methods can be complex,

conservation, specifically upon conserving the local proportion of the total integral (mass) of the dependent variable across the domain. We concentrate

not satisfy the set boundary conditions. As with Richards' equation, this demonstrates the need for a numerical solution.

The Crank-Gupta problem has a negative source term, resulting in a solution that decreases in mass. The final problem is covered in Chapter 7 where we consider a model for avascular tumour growth which has an increasing mass. This is the most complicated of the four problems. We begin this chapter with a brief introduction to tumours, and the role that mathematics has played on tumour growth research. We note that the modelling of tumour growth is an area of much interest to mathematical b

2

Background on Moving Mesh Methods

Moving mesh methods belong to the class of adaptive mesh methods. They are often referred to as r-adaptivity (relocation) methods.

Such methods have a natural application to problems with close coupling between

of interest can often be identified by a rapid variation of either the solution, or one of its derivatives. The error from an r-adaptive method will depend not only on the solution itself, but also the number and position of the mesh nodes. Moving mesh methods often utilise two

Baines, Hubbard and Jimack [5, 9], since the methods we will use are essentially finite difference variations on their work. However, we begin with location-based methods.

2.1 Location-based methods

tion principle.

Huang et al. [55] recommend that a moving mesh method should not only move nodes to areas of interest, but require a simple algorithm that is easy to program, and be reasonably insensitive to the choice of user-defined parameters. In their work this is achieved by constructing moving mesh equations directly from the numerical equidistribution principle. Furthermore, they recommend that in order to allow ease of comparison and theoretical analysis of this moving mesh method, the moving mesh equations should have a continuous form, and in [55] these equations are devised and referred to as moving mesh partial differential equations (MMPDEs). MMPDE-based atio

mesh points to follow the flow itself, [28]. This intuitive movement of mesh points makes a velocity-based moving mesh approach particularly suited for fluid flow, but the approach is more general.

In this section a number of velocity-based methods are described. We begin in §2.2.1 with schemes related to fluid dynamics which use a purely Lagrangian approach. This leads on to methods which rely on the so-called ALE (Arbitrary Lagrangian-Eulerian) formula-

Chapter Two

This equation is widely used in the numerical solution of uid-structure interaction problems, for example in [54, 60, 71, 86]. The speci c mechanism of constructing the mesh velocities varies signi cantly, from treating the mesh as though it is a physical material with its own constitutive law [60] through to de ning the mes h motion purely with the goal of optimizing geometric qualities of the mesh [86].

Other ALE forms, such as the di erential form,

$$\frac{Du}{Dt} = \frac{@u}{@t} + v \quad r \; u; \tag{2.6}$$

have also been used for free-surface problems, based upon mai ning mesh equality [78, 79], Laplacian smoothing [89] or pseudo-solid deformation [2, 101]. Other applications which have bene ted from successful ALE algorithms include phase-change problems [70, 93], and the interaction of free surfaces with solid boundar

singular and signi cant regularisation is needed.

2.2.4 The Geometric Conservation Law (GCL)

The GCL is a tool which has been used for many years in the engineering community to develop cell-volume-preserving nite-volume schemes. An early example, formulated in [97], is the Space Conservation Law (SCL) which was approximated i

2.3 The Finite Element Conservation Method

Chapter Two

The nite element formulation and the full algorithm

In [5], a nite element mesh is set up and the initial condition is projected on to the mesh. Piecewise linear basis function are used, replacing w_i with $_i$, where $_i$ are linear nite element basis functions on the mesh of nodes $x_i(t)$, $i = 0;1;:::;N$, within a polygonal approximation (t) of (t).

The equations which determine $, v(x;t)$ and $u(x;t)$ (from equations (2.19), (2.20) and (2.21) respectively), are used with $x, , v,$ and u replaced with piecewise linear approximations. The algorithm e 248413(t)0.313314(h)0.328t

3

A Finite Difference Velocity-Based Moving Mesh

Our moving mesh method moves the nodes such that the partial masses of the solution are conserved, i.e. we determine $\tilde{x}_j(t)$ $(j = 0, \ldots, N)$ from

$$c_j = \int_{a(t)}^{\tilde{x}_j(t)} u(x,t)\,dx, \tag{3.6}$$

where the c_j $(j = 0, \ldots, N)$ remain constant in time. Equation (3.6) is consistent with equation (3.4). The c_j are positive since $u > 0$, and $c_N = c$.

For a given mesh $\tilde{x}_j(t)$ and solution $u_j(t) = u(\tilde{x}_j(t), t$

3.1.2 Advancing the mesh in time

We choose a time step $\Delta t > 0$ and define time-levels $t^m = m\,\Delta t$, $m = 0, 1, \ldots$, denoting $x_j(t^m)$ by x_j^m. We also use the approximations $u_j^m \approx u_j(t^m)$ and $v_j^m \approx v_j(t^m)$. The updated x_j^{m+1} are calculated using the mesh velocity w_j^m with a time-stepping scheme.

3.1.3 Recovering the solution

To recover the solution on the new mesh we use an incremental form of (3.6),

$$c_{j+1} - c_{j-1} = \int_{x_{j-1}^{m+1}}^{x_{j+1}^{m+1}} u(x,t)\,dx = \int_{x_{j-1}^{t^0}}^{x_{j+1}^{t^0}} u(x,t^0)\,dx, \qquad j = 1, \ldots, N-1, \tag{3.9}$$

(where each c_j is a constant known from initial conditions). We approximate the integrals of (3.9) using a simple quadrature rule, which allows us to recover u_j^{m+1}, $j = 1, \ldots, N-1$ on the new mesh. We examine two different quadrature rules, a mid-point approximation and an interpolating approximation on a non-uniform mesh.

1 in the intervals adjacent to x_j as

$$c_j = c_j - c_{j-1};$$
$$c_{j+} = c_{j+1} - c_j;$$

where each of these are determined from the initial conditions. We note that $c_{j+} + c_j = c_j$, from (3.9). We then use (3.11) to derive two equations for the sum $(c_{j+} + c_j)$ and difference $(c_{j+} - c_j)$. These equations are evaluated at $t = t^{m+1}$, and the unknown slope $g_j^{m+1} - g(x_j, t^{m+1})$ is eliminated, so that we achieve an expression for u_j^{m+1}

mass (which is required for determining the updated solution) we first compute $\dot{\lambda}(t)$. Then the mesh velocity $v_J(t)$ is computed. The mesh and total mass are updated simultaneously using a time-stepping scheme. This enables us to recover the updated solution on the new mesh. Details are given in the following subsections.

3.2.1 Determining the rate of change of totae..4 (.148055(.)0.0827066(1)-1125.32(D)0.2930

Chapter Three

u_j^{m+1} (which is exact for a linear solution on a non-uniform mesh) isthe same as (3.16), but with the addition of the ratio of total masses $\frac{m+1}{0}$, i.e.

$$u_j^{m+1} = \frac{m+1}{0} \frac{\frac{c_j = x_j^{m+1}}{x_j^{m+1}} + \frac{c_{j+} = x_{j+}^{m+1}}{x_{j+}^{m+1}}}{\frac{1}{\quad}}$$

Substituting (3.30) into (3.29), we have

$$\lambda_j(t) = \int_{a(t)}^{x_j(t)} S(x;t)\,dx. \tag{3.31}$$

For a specific IBVP, the integral in (3.31) can then be approximated using quadrature to determine a discrete form of (3.31).

3.3.2 Determining the mesh velocity

To obtain an expression for the mesh velocity $v_j(t)$ we differentiate (3.30) with respect to time using the Leibnitz integral rule again, to give

$$\lambda_j(t) = \frac{d}{dt}\int_{a(t)}^{x_j(t)} u(x;t)\,dx = \int_{a(t)}^{x_j(t)} \frac{\partial u}{\partial t}\,dx + \mu_j(t)\nu_j(t) - \mu_0(t)\nu_0(t).$$

Substituting $\frac{\partial u}{\partial t}$ from (3.27), and using the boundary conditions (3.28),

$$\lambda_j(t) = \int_{a(t)}^{x_j(t)} \{H u + S(x;t)\}\,dx + \mu_j(t)\nu_j(t). \tag{3.32}$$

Equating (3.32) and (3.31),

$$\int_{a(t)}^{x_j(t)} H u\,dx + \mu_j(t)\nu_j(t) = 0.$$

Thus, for $\mu_j(t) \neq 0$, the nodes move such that

$$\nu_j(t) = -\frac{1}{\mu_j(t)}\int_{a(t)}^{x_j(t)} H u\,dx. \tag{3.33}$$

Recalling Remark 3.3.1, equation (3.33) holds for interior points $j = 1,\ldots,N-1$. Again, the mesh velocities at the boundaries, $\nu_0(t), \nu_N(t)$, can be extrapolated from the interior mesh velocities. We observe that if there were no source term, equation (3.33) would be equivalent to (3.8) for $j = 1,2,\ldots,N-1$.

Once more, as in 3.2.1, for a specific IBVP the integral in (3.33) is approxima

mesh x_j^m, and solution u_j^m, $j = 0;::::;N$, we compute the rate of change of partial C

Compute the mesh velocity v_j^m from a discrete form of (3.33). Determine the velocity at the boundaries from an appropriate extrapolation scheme;

Compute the updated partial masses $_j^{m+1}$ and mesh x_j^{m+1};

Compute the updated solution u_j^{m+1} from (3.34) or (3.35). The solution at the boundaries, u_0^{m+1}; u_N^{m+1}, are given by the boundary conditions.

Examples are discuses in Chapters 6 and 7.

We have shown how we can solve a problem that does not conserve mass using a moving mesh approach that balances the partial mass fractions with a source term. We now look at some of the time-stepping schemes that we use with our moving mesh method.

3.4 Time-stepping schemes

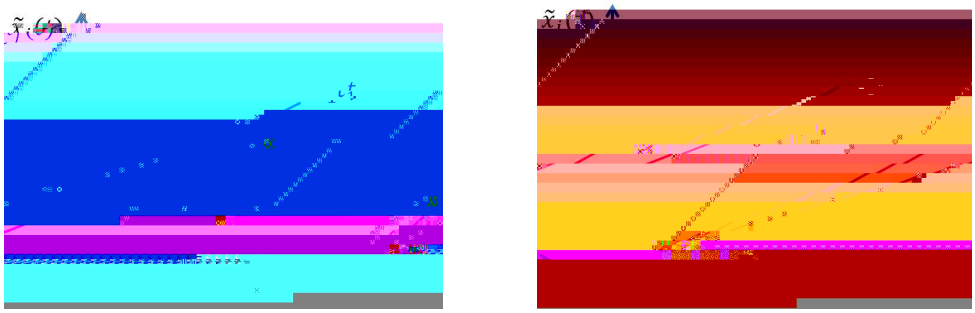


Fig. 3.1: Diagrams to illustrate the relation between $x_j(t)$ and j for a mesh that is tangled compared to one that is not tangled. The graph on the left shows a mesh that is tangled (not monotonic) at time t_1.

Both solvers invoke explicit Runge-Kutta methods to integrate the system of N 1 ordinary di erential equations. The solver ODE23 uses second an

Then, assuming that the mesh $x_j(t)$ changes smoothly in time, we alter the Euler scheme (3.36) to be semi-implicit, in the manner

$$\frac{x_j^{m+1} - x_j^m}{t} = \frac{1}{x_j^m - x_{j-}^m}\left(\alpha_{j+}^m \frac{x_j^{m+1} - x_{j+}^{m+1}}{x_j^m - x_{j+}^m} + \alpha_j^m \frac{x_j^{m+1} - x_{j-}^{m+1}}{x_j^m - x_{j-}^m}\right); \quad j = 1, \ldots, N-1; \tag{3.39}$$

where $x_{j+}^m = (x_{j+1}^m - x_j^m)$ and $x_j^m = (x_j^m - x_{j-1}^m)$. We choose x_j^{m+1} to be either x_{j+}^{m+1} or x_j^{m+1}, that is

$$\frac{x_j^{m+1} - x_j^m}{t} = \begin{cases} \frac{1}{x_j^m - x_{j-}^m}\left(\alpha_{j+}^m \frac{x_j^{m+1} - x_{j+}^{m+1}}{x_j^m - x_{j+}^m} + \alpha_j^m \frac{x_j^{m+1} - x_{j-}^{m+1}}{x_j^m - x_{j-}^m}\right) & \text{for } \alpha_{j+}^m, \alpha_j^m > 0; \\ \frac{1}{x_j^m - x_{j-}^m}\left(\alpha_{j+}^m \frac{x_{j+}^{m+1} - x_{j+}^{m+1}}{x_j^m - x_{j+}^m} + \alpha_j^m \frac{x_{j+}^{m+1} - x_{j-}^{m+1}}{x_{j+}^m - x_{j-}^m}\right) & \text{for } \alpha_{j+}^m > 0, \alpha_j^m < 0; \\ \frac{1}{x_j^m - x_{j-}^m}\left(\alpha_{j+}^m \frac{x_j^{m+1} - x_{j+}^{m+1}}{x_j^m - x_{j+}^m} + \alpha_j^m \frac{x_j^{m+1} - x_{j-}^{m+1}}{x_j^m - x_{j-}^m}\right) & \text{for } \alpha_{j+}^m, \alpha_j^m < 0; \\ \frac{1}{x_j^m - x_{j-}^m}\left(\alpha_{j+}^m \frac{x_j^{m+1} - x_{j+}^{m+1}}{x_j^m - x_{j+}^m} + \alpha_j^m \frac{x_{j+}^{m+1} - x_{j-}^{m+1}}{x_{j+}^m - x_{j-}^m}\right) & \text{for } \alpha_{j+}^m < 0, \alpha_j^m > 0; \end{cases} \tag{3.40}$$

for reasons that will be explained below. The boundary values, $x_0(t), x_N(t)$, are updated explicitly by a first order scheme. These are calculated before the internal nodes so that x_1^{m+1} and x_{N-1}^{m+1} can be determined. The whole scheme is then first order in time

Theorem 3.4.1 The semi-implicit scheme (3.39) ensures that the mesh does not tangle, i.e.

$$x_{j-1}^m < x_j^m < x_{j+1}^m; \tag{3.41}$$

for all $j = 0, 1, \ldots, N$ and all time t^m, $m = 1, 2, \ldots$, provided the x_j^{m+1} are chosen to be either x_{j+}^{m+1} or x_j^{m+1} according to the four parts of (3.40).

Proof Given that the mesh is not tangled at time-level t^m we show that (3.41) holds for all subsequent time-levels by proving that the maximum and minimum of the set $\{x_j^{m+1}\}$ occur at the boundaries, for all $j = 1, \ldots, N-1$.

We first prove the maximum principle, by contradiction. Suppose that an isolated maximum of x occurs at the interior point x_j^{m+1}. We consider the sign of each term in (3.39), and subsequently determine that the sign of right-hand side contradicts that of the

Now, if the interior point x_j^{m+1} is a maximum then

$$x_j^{m+1} - x_j^m > 0; \tag{3.42}$$

$$x_j^{m+1} = x_j^{m+1} - x_{j-1}^{m+1} > 0; \tag{3.43}$$

$$x_{j+}^{m+1} = x_{j+1}^{m+1} - x_j^{m+1} < 0. \tag{3.44}$$

The inequality (3.42) implies that the left-hand side of (3.39) is positive. To complete the proof by contradiction, we note that x_j^{m+1} gives a negative right-hand side if x_j^{m+1} is not carefully considered. For example, suppose $_{j+}^m$; $_j^m < 0$ and the x_j^{m+1} of (3.39) are determined such that

$$\frac{x_j^{m+1} - x_j^m}{t} = \frac{1}{x_j^m} \left({}_{j+}^m \frac{x_j^{m+1}}{x_j^m} - {}_j^m \frac{x_{j+}^{m+1}}{x_{j+}^m} \right);$$

This gives a negative right-hand side, contradicting the left-hand side. We have therefore shown that an isolated maximum cannot occur. Using the same reasoning given here, it can be shown that a set of equal maximum values cannot occur. Hence the maximum occurs at the boundary.

A minimum principle can be proved with the same reasoning to show that an isolated minimum can not occur at any interior point when x_j^{m+1} is specified according to (3.40). The proof shows that x_j^m, $j = 0; 1; :::; N$, is bounded by its neighbours. It also extends to non-isolated interior points and hence, equation (3.39) with appropriate x_j^{m+1} (determined by (3.40)), ensures that

4

The Porous Medium Equation

4.1 Introduction

The Porous Medium Equation (PME)

$$\frac{\partial u}{\partial t} = \nabla \cdot (u(x,t)^n \nabla u), \tag{4.1}$$

where $n > 1$ is one of the simplest nonlinear evolution equations of parabolic type. It can be used to describe physical situations such as fluid flow, heat transfer or diffusion. Most notably, it is used to describe the flow of a perfect gas in a homogeneous porous medium.

The main aim of this chapter is to solve the PME numerically using the moving mesh method described in §3.1.

We begin this chapter by deriving the PME from a general form of Darcy's Law in §4.2, as shown in [99]. Then, before solving the PME numerically, we discuss some of the properties of the PME which our numerical scheme aims to preserve, in §4.3. One of these properties is self-similarity, so in §4.4 we derive a self-similar solution, as originally presented in [12]. This self-similar solution is used to provide the initial conditions when we solve the PME numerically, and also to compare the numerical solution to an exact solution in the results section, §4.7. Finally, in §4.8 we present results using the moving mesh finite

element method discussed in §2.3.

4.2 Deriving the PME from Darcy's Law

We first derive the PME by considering three model equations which relate variables associated with gas flow through a porous medium.

(i) Mass balance

We assume that the flow of gas obeys the equation of continuity

$$\frac{\partial}{\partial t}\phi + \nabla \cdot (\rho V) = 0; \tag{4.2}$$

where $\phi \in (0,1)$ is the porosity of the medium, ρ is the density, and V is the velocity. Here $\nabla \cdot$ represents the divergence operator.

(ii) Darcy's Law

Darcy's Law was formulated by the French engineer H. Darcy in 1856 [99]. It models the flow of a fluid (or gas) through a porous medium. Maintaining the notation in (4.2),

$$V = -\frac{k}{\mu}\nabla p; \tag{4.3}$$

where ∇p is the pressure gradient vector, k is the permeability tensor (assumed to be a strictly positive constant in most applications), and $\mu > 0$ is the viscosity of the fluid (or gas).

(iii) Equation of state

The equation of state for a perfect gas is

$$p = p_0 \rho^{\gamma}; \tag{4.4}$$

where p is the pressure, p_0 is the reference pressure and $\gamma \geq 1$ is the ratio of specific heats for the gas.

Substituting Darcy's Law (4.3) and the equation of state (4.4) into the mass con-

servation equation (4.2) gives

$$\frac{@}{@t} = \frac{p_0}{} r :(r \quad): \tag{4.5}$$

Now

$$r \quad = \quad {}^{1}r \quad = \quad r \ ;$$

so we may write (4.5) as

$$\frac{@}{@t} = \frac{p_0}{} r :(\quad r \quad):$$

The constant $\frac{p_0}{}$ can be scaled out (de ne for instance a new time,$t^0 = \frac{p_0}{}t$), thus leaving us with the PME. We adapt this result to meet standard notation by writing $= n$, and $= u$, giving

$$\frac{@u}{@t} = r \quad (u^n r \ u);$$

where $n \quad 1$. In one-dimension this equation is

$$\frac{@u}{@t} = \frac{@}{@x} \ u^n \frac{@u}{@x} \ : \tag{4.6}$$

4.3 Properties of the PME in one dimension

We are interested in several properties of the PME for the one-dimensional Cartesian case (4.6) with its boundary defined by the edge of the support, i.e.

$$u = 0 \quad \text{at} \quad x = a(t), b(t), \quad t > 0. \tag{4.8}$$

In [99] many properties of the PME are given, and proved. We prove two well-established properties which our numerical approach relies upon: conservation of mass, and stationary centre of mass.

Lemma 4.3.1 The one-dimensional PME conserves mass in time.

Proof To prove that the total mass does not change over time we show that the derivative of the total mass (in time) is zero. Using the Leibnitz integral rule,

$$\frac{d}{dt}\int_{a(t)}^{b(t)} u(x,t)\,dx = \int_{a(t)}^{b(t)} \frac{\partial u}{\partial t}\,dx + u(b,t)\frac{db}{dt} - u(a,t)\frac{da}{dt},$$

Substituting $\frac{\partial u}{\partial t}$ from (4.6), and setting the last two terms to zero due to the boundary conditions (4.8),

$$\begin{aligned}\frac{d}{dt}\int_{a(t)}^{b(t)} u(x,t)\,dx &= \int_{a(t)}^{b(t)} \frac{\partial}{\partial x}\left(u(x,t)^n \frac{\partial u}{\partial x}\right) dx, \\ &= u(b,t)^n \frac{\partial b}{\partial x} - u(a,t)^n \frac{\partial a}{\partial x}.\end{aligned}$$

The right-hand side is zero, due to boundary conditions (4.8) again, hence

$$\frac{d}{dt}\int_{a(t)}^{b(t)} u(x,t)\,dx = 0,$$

as required. The result is easily extended to the radially symmetric case

$$\frac{d}{dt}\int_{0}^{R(t)} u(r,t) r^{\nu-1}\,dr = 0,$$

where $R(t)$ is the radius and $\nu = 2, 3$ is the number of dimensions. □

Lemma 4.3.2 The one-dimensional PME has a stationary centre of mass.

Proof The centre of mass $x(t)$ in one dimension is defined by the ratio

$$x(t) = \frac{\int_{a(t)}^{b(t)} u(x;t)x\,dx}{\int_{a(t)}^{b(t)} u(x;t)\,dx}. \tag{4.9}$$

To demonstrate that the centre of mass $x(t)$ does not move, it is sufficient to show that

$$\frac{d}{dt}\int_{a(t)}^{b(t)} u(x;t)x\,dx = 0;$$

since the denominator of (4.9) is constant in time. By the Leibnitz integral rule,

$$\frac{d}{dt}\int_{a(t)}^{b(t)} u(x;t)x\,dx = \int_{a(t)}^{b(t)} x\frac{\partial u}{\partial t}dx + u(b;t)b(t)\frac{db}{dt} - u(a;t)a(t)\frac{da}{dt}.$$

Substituting $\frac{\partial u}{}$

that the centre of mass remains stationary allows us to set the mesh velocity to zero at that point. Before numerically solving the PME, we recall the self-similar solution to the PME in the next section.

4.4 A self-similar solution

In this section we recall a class of exact solutions to the radially symmetric PME (4.7) that are invariant under a scaling group in the variables (t, r, u), and therefore take the so-called self-similar form.

A time dependent phenomenon is called self-similar if the spatial distributions of its variables at different times can be obtained from one another by a similarity transform [11], which is a transformation that maintains certain features of a function or curve. A particular similarity transformation is a scale-invariant transformation, where the variables are scaled by powers of a common factor.

A prerequisite to deriving a self-similar scaling solution is determining a scale-invariant transformation of the PME [11].

4.4.1 Scale invariance

Scale invariance, defined as the invariance of the PDE under scaling, originated from the analysis of the consequences of the change of units of measur

The first of these (4.12) is the left-hand side of the PME. Using (4.13) we transform the right-hand side to get

$$\frac{1}{r}$$

To summarise, the variables u, r and t can be rescaled as in (4.18) whilst still satisfying

A further substitution of $rt = $, from (4.20), in the rst term on the right-hand side gives

$$\frac{@u}{@t} = \quad t \quad ^{1}\frac{d}{d} + \quad ( ) t \quad ^{1}: \tag{4.21}$$

We have expressed the left-hand side of (4.7) in terms of and , and we wish to transform the right-hand side in the same way. Thus

$$\frac{1}{r \quad ^{1}}\frac{@}{@r} \quad r \quad ^{1}u(r;t)^{n}\frac{@u}{@r} = t \quad ^{(1 \quad)}\frac{1}{\quad ^{1}}\frac{@}{@r}\frac{d}{rd} \quad \quad ^{1}t \quad ^{(\quad 1)} \quad ( )^{n}t \quad ^{n}\frac{@}{@r}\frac{@u}{@}\frac{d}{d} \quad :$$

Substituting $\frac{@}{@r} = t \quad$ from (4.20), and $\frac{@u}{@} = t \quad$ from (4.19) into the right-hand side, gives

$$\begin{aligned}\frac{1}{r \quad ^{1}}\frac{@}{@r} \quad r \quad ^{1}u(r;t)^{n}\frac{@u}{@r} \quad &= \quad \frac{1}{\quad ^{1}}t \quad \quad \frac{d}{d} \quad \quad ^{1}t \quad ^{(\quad 2)+ \quad (n+1)} \quad ( )^{n}\frac{d}{d} \quad ; \\ &= \quad t \quad ^{(n+1) \quad 2} \quad \frac{1}{\quad ^{1}}\frac{d}{d}\frac{}{d}( ): \quad ^{1} \quad ( )^{n}\frac{d}{d} \quad :\end{aligned} \tag{4.22}$$

Putting together (4.21) and (4.22) gives the PME (4.7) in terms of and ,

$$t \quad ^{1}\frac{d}{d} + \quad ( ) t \quad ^{1} = t^{n} \quad ^{2} \quad \frac{1}{\quad ^{1}}\frac{d}{d} \quad \quad ^{1} \quad ( )^{n}\frac{d}{d} \quad :$$

Note that t disappears from the equation since $n \quad 2 \quad +1 = 0$ (from (4.14)). Substituting for and from (4.17), gives the ODE

$$\frac{1}{n \quad +2} \quad \frac{d}{d} \quad \quad ( ) \quad \frac{}{n \quad +2} = \frac{1}{\quad ^{1}}\frac{d}{d} \quad \quad ^{1} \quad ( )^{n}\frac{d}{d} \quad :$$

By moving all the terms to one side we have,

$$\frac{1}{\quad ^{1}}\frac{d}{d} \quad \quad ^{1} \quad ( )^{n}\frac{d}{d} \quad + \frac{}{n \quad +2}\frac{d}{d} + \frac{( )}{n \quad +2} = 0: \tag{4.23}$$

From the zero Dirichlet boundary conditions imposed on the PME we have corresponding zero Dirichlet boundary conditions on in (4.23). The solution of the two point boundary problem (4.23), along with the previous de nitions $u = t \quad$ and $r = t \quad$ provides the self-similar solution.

Using an integrating factor $\exp\left\{ \int \frac{}{n \quad +2} d \right\} = \quad$ enables us to group the last two terms of (4.23), giving

$$\frac{1}{\quad ^{1}} \quad d$$

We multiply through by ^{1} and integrate to get

$$^{1}\ (\ )^{n}\frac{d}{d}+\frac{(\ )}{(n\ +2)}=\ C;$$

where C is an integration constant. Taking C = 0, which it is at the boundary where = 0,

$$^{1}\ (\ )^{n\ \ 1}\frac{d}{d}+\frac{1}{n\ +2}=0:$$

Separating the variables,

$$\int\ (\ )^{n\ \ 1}d\ =\ \frac{1}{n\ +2}\int\ d\ ;$$

which gives

$$\frac{(\ )^{n}}{n}=\ \ \frac{1}{n\ +2}\ \ \frac{^{2}}{2}\ \ E\ \ ;$$

where E is a constant ntegratec

as initial conditions. We also discuss the e ect of not usinga self-similar solution as the initial conditions, where we nd that the PME boundaries do n ot move until the solution at the boundary resembles a self-similar solution. This behaviour is nicely captured by our moving mesh scheme.

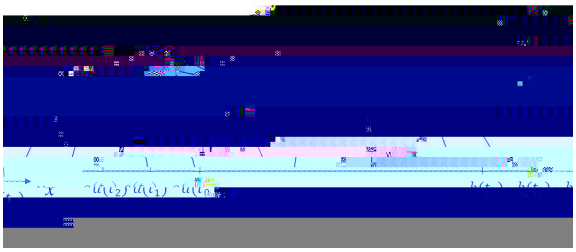


Fig. 4.1:

tial solution for our numerical work, as well as an exact solution to compare with our numerical results. In the next section we apply our moving mesh method to the Cartesian one-dimensional PME.

4.5 Moving meshes

The integral of a solution of the PME (the mass) is conserved in time, so we use the moving mesh method described in §3.1, with the same notation, i.e. $x_j(t^m) \equiv x_j^m$ denotes the jth node of the mesh with $N+1$ nodes, at time $m\,\Delta t$, $m = 0, 1, \ldots$, and $u_j^m \equiv u_j(t^m)$ and $v_j^m \equiv v_j(t^m)$ denote the solution and mesh velocity at these nodes.

We model the PDE (3.1) with

$$Lu \equiv \frac{\partial}{\partial x}\, u(x,t)^n \frac{\partial u}{\partial x}\,; \tag{4.29}$$

from (4.6). The moving mesh method in §3.1 can be applied to any geometrically non-symmetric problem. However, for convenience we assume that the one-dimensional solution $u(x,t)$ is symmetrical about its centre of mass (see §4.3). Then by symmetry we need only model half the region $x(t) \in [0, b(t$

four are

∂u

for interior points $j = 1, \ldots, N-1$. Since $\frac{\partial u}{\partial x} = 0$ at $x = 0$, the interior points move in time such that

$$\dot{x}_j(t) = -u_j(t)^{n-1} \left.\frac{\partial u}{\partial x}\right|_{x_j(t)}; \tag{4.32}$$

which can also be written as

$$\dot{x}_j(t) = -\frac{1}{n} \left.\frac{\partial (u^n)}{\partial x}\right|_{x_j(t)}. \tag{4.33}$$

Remark 4.5.1 We refer to §4.2 and observe that the mesh velocity (4.33) resembles the velocity V obtained from substituting the equation of state (4.4) into Darcy's Law (4.3),

$$V = -\frac{p_0}{\mu} \nabla \rho^{\gamma};$$

where $\rho = u$ and $\gamma = n$.

We use a discretised form of (4.33), at time $t = t^m$

$$v_j^m = -\frac{1}{n}\frac{(u^n)_{j+1}^m - (u^n)_{j-1}^m}{(x_{j+1}^m - x_{j-1}^m)}; \qquad j = 1, 2, \ldots, N-1; \tag{4.34}$$

which is a second order discretisation on a uniform mesh, but only a first order discretisation on a non-uniform mesh. By definition, at the inner boundary $v_0^m = 0$. The outer boundary velocity v_N^m is extrapolated by a polynomial approximation using $(v_{N-3}^m, v_{N-2}^m, v_{N-1}^m)$.

We note that when approximating $(u^n)_x\big|_{x_j(t)} = \left.\frac{\partial u^n}{\partial x}\right|_{x_j(t)}$

Expanding both of these using a Taylor series about $x_j(t)$ gives

$$(u^n)^m_{j+1} \quad u_j(t)^n + \quad x^m_{j+} \frac{@(u^n)}{@x}_{x_j(t)} + \frac{1}{2}(\quad x^m_{j+})^2 \frac{@^2(u^n)}{@x^2}_{x_j(t)} + O(\quad x^m_{j+})^3; \quad (4.35)$$

$$(u^n)^m_{j\ 1} \quad u_j(t)^n \quad x^m_j \frac{@(u^n)}{@x}_{x_j(t)} + \frac{1}{2}(\quad x^m_j)^2 \frac{@^2(u^n)}{@x^2}_{x_j(t)} + O(\quad x^m_j)^3: \quad (4.36)$$

We subtract (4.36) multiplied by $(\quad x^m_{j+})^2$ from (4.35) multiplied by $(\quad x^m_j)^2$ to give a higher order approximation to $(u^n)_{x\ x_j(t)}$ at time t^m as,

$$(u^n)_{x\ x_j(t)} = \frac{@(u^n)}{@x}_{x_j(t)} \quad \frac{(\quad x^m_j)^2 \ (u^n)^m_{j+1} \quad (u^n)^m_j + (\quad x^m_{j+})^2 \ (u^n)^m_j \quad (u^n)^m_{j\ 1}}{x^m_j \quad x^m_{j+} \quad x^m_{j+} + \quad x^m_j};$$

$$= \quad \frac{\frac{1}{x^m_{j+}} \quad \frac{(u^n)^m_{j+}}{x^m_{j+}} + \frac{1}{x^m_j} \quad \frac{(u^n)^m_j}{x^m_j}}{\frac{1}{x^m_{j+}} + \frac{1}{x^m_j}}; \quad (4.37)$$

for $j = 1; : : : ; N \quad 1$, where $(u^n)^m_{j+} = (u^n)^m_{j+1} \quad (u^n)^m_j$ and $(u^n)^m_j = (u^n)^m_j \quad (u^n)^m_{j\ 1}$ (corresponding to our earlier de nitions for x^m_j). The second order approximation (4.37) is an inversely weighted sum, or interpolation, of slopes and is frequently used in numerical work throughout this thesis.

Substituting (4.37) in (4.33) gives the second-order approximation to the mesh velocity,

$$v^m_j = \quad \frac{1}{n} \frac{\frac{1}{x^m_{j+}} \quad \frac{(u^n)^m_{j+}}{x^m_{j+}} + \frac{1}{x^m_j} \quad \frac{(u^n)^m_j}{x^m_j}}{\frac{1}{x^m_{j+}} + \frac{1}{x^m_j}}; \quad (4.38)$$

for $j = 1; : : : ; N \quad 1$ where the time-level m notation has been re-instated. As with (4.34), the velocity at the inner boundary is given by $v^m_0 = 0$, and at the outer boundary the velocity v^m_N is extrapolated by a polynomial approximation using $(v^m_{N\ 3}; v^m_N$ TJ T*[(j)-0.545258 0 Td [(()0.3088383(

from (3.10) and the boundary condition (4.30). At the outer boundary, $u_{N+1}^{m+1} = 0$ from (4.31).

4.5.3 The full algorithm

Given a mesh x_j^m, solution u_j^m, $j = 0, \ldots, N$, $m \geq 0$:

Compute the mesh velocity v_j^m from (4.34) or (4.38);

Using a time-stepping scheme, compute the updated mesh x_j^{m+1} by a time-stepping scheme;

Compute the updated solution u_j^{m+1} from (3.10) or (3.16).

4.5.4 Waiting times

The velocity of the boundary is given by (4.32). The boundary behaviour of the PME has been investigated in [62, 90]. We now present some of their findings.

At the boundary $u = 0$, which infers that the boundary velocity is zero unless u_x is infinite for $n > 1$. We examine the effect of the initial conditions on the boundary velocity by considering initial conditions at $t = 1$, of the form given in [18, 62],

$$u(x, 1) = \left[1 - \frac{x - s(1)}{b(1)} \right]^2 = 1 - \frac{x - s(1)}{b(1)} \left[1 + \frac{x - s(1)}{b(1)} \right], \qquad (4.39)$$

where $= \frac{1}{n}$ (the self-similar initial condition). We use (4.39) to determine u_x in (4.32), to give

$$\frac{ds}{dt} = \frac{2}{b(1)^2} \quad 1 \quad s(1)$$

The boundary behaviour for these three cases is shown in Figure 4.2.

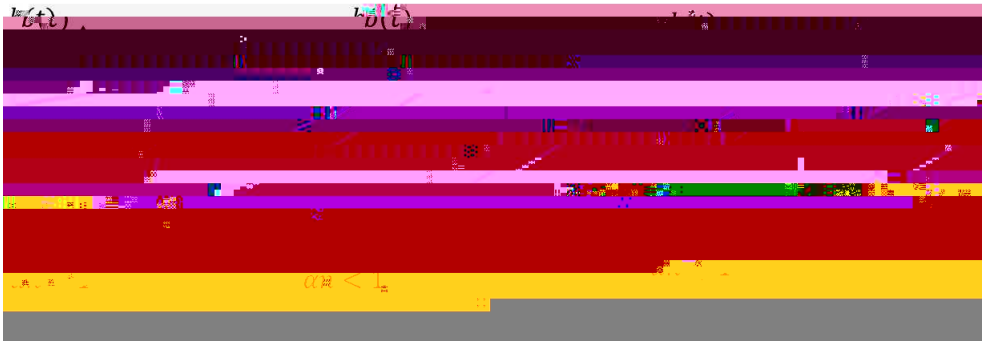


Fig. 4.2: The three different types of boundary behaviour for the PME.

A semi-implicit scheme

To determine a semi-implicit time-stepping scheme for solving the PME we consider the general semi-implicit time-stepping scheme (3.39) (for the internal nodes, $j = 1; 2; :::; N \quad 1$) with

$$ {}^{m}_{j+} = \frac{1}{n}(u^n)^m_{j+\frac{1}{2}}; \qquad {}^{m}_{j} = \frac{1}{n}(u^n)^m_{j\ \frac{1}{2}}; $$

from (4.33). To ensure that the mesh does not tangle using (3.40), we determine the x_j^{m+1} terms in (3.39) such that

$$ \frac{x_j^{m+1} \quad x_j^m}{t} = \frac{1}{n\ x_j^m}\ (u^n)^m_{j+\frac{1}{2}} \frac{x_j^{m+1}}{x_j^m} \quad (u^n)^m_{j\ \frac{1}{2}} \frac{x_{j+}^{m+1}}{x_{j+}^m}\ ! ; \tag{4.43} $$

where $x_j^m = x^m_{j+\frac{1}{2}} \quad x^m_{j\ \frac{1}{2}}$, $x_j^m = x_j^m \quad x_{j\ 1}^m$ and $x_{j+}^m = x^m_{j+1} \quad x_j^m$. Before calculating the internal nodes semi-implicitly by (4.43), the boundary node x_N^{m+1} is calculated by the explicit scheme (4.42), enabling $x^{m+1}_{N\ 1+} = x_N^{m+1} \quad x^{m+1}_{N\ 1}$ to be determined.

Rearranging (4.43) and expanding the x_j^{m+1} terms,

$$ \frac{x_j^{m+1} \quad x_j^m}{t} = \frac{(u^n)^m_{j+\frac{1}{2}}\ x_{j+}^m (x_j^{m+1} \quad x_{j\ 1}^{m+1}) \quad (u^n)^m_{j\ \frac{1}{2}}\ x_j^m (x_{j+1}^{m+1} \quad x_j^{m+1})}{n\ x_j^m\ x_{j+}^m\ x_j^m}: \tag{4.44} $$

Our moving mesh method moves the nodes such that partial masses of the solution are conserved, see equation (3.6) in §3.1. Bearing this in mind we define approximations to the mass in the subintervals which remain unchanged in time, as

$$ \begin{aligned} C_{j+} &= (u^n)^0_{j+\frac{1}{2}}\ x^0_{j+} = (u^n)^m_{j+\frac{1}{2}}\ x^m_{j+}; \\ C_j &= (u^n)^0_{j\ \frac{1}{2}}\ x^0_j = (u^n)^m_{j\ \frac{1}{2}}\ x^m_j; \end{aligned} $$

thus simplifying equation (4.44) to

$$ \frac{x_j^{m+1} \quad x_j^m}{t} = \frac{C_{j+}(x_j^{m+1} \quad x_{j\ 1}^{m+1}) \quad C_j\ (x_{j+1}^{m+1} \quad x_j^{m+1})}{n\ x_j^m\ x_{j+}^m\ x_j^m}: \tag{4.45} $$

To determine the new mesh by the semi-implicit scheme we solve the system

$$ Ax^{m+1} = x^m; \tag{4.46} $$

where $x^{m+1} = (x_1^{m+1}; \quad x_{N-1}^{m+1})^T$, $x^m = (x_1^m; \quad x_{N-1}^m)^T$, and A is a tridiagonal matrix with lower, main and upper diagonals Al_j, Ad_j and Au_j given by

$$Al_j = \qquad C_{j+}$$

Chapter Four

Chapter Four

and hence means that the updated solution is recovered on the new mesh by

$$u_j^{m+1} = \frac{(r^{\alpha-1})_j^0}{(r^{\alpha-1})_j^{m+1}} \left(\frac{r_{j+1}^0 - r_j^0}{r_{j+1}^{m+1} - r_j^{m+1}} u_j^0 \right) \tag{4.55}$$

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

However, as the mesh is not necessarily uniform, it is more accurate to use a quadrature rule for a non-uniform mesh. We note that the term in the brackets on the right-hand side of (4.55) is the same as the right-hand side of (3.10), where $x = r$. Subsequently, we can replace this term in (4.55) with the right-hand side of (3.16) (substituting $x = r$), which simplifies to (4.55) for a uniform mesh. Hence, a better approximation to update the solution is

$$u_j^{m+1} = \frac{(r^{\alpha-1})_j^0}{(r^{\alpha-1})_j^{m+1}} \left(\frac{c_j - r_j^{m+1}}{r_j^{m+1}} + \frac{c_{j+} - r_{j+}^{m+1}}{r_{j+}^{m+1}} \right) \Bigg/ \left(\frac{1}{r_j^{m+1}} + \frac{1}{r_{j+}^{m+1}} \right), \tag{4.56}$$

for $j = 1, \ldots, N-1$, where

$$c_j = \int_{r_{j-1}(t^0)}^{r_j(t^0)} u(r, t^0) r^{\alpha-1} \, dr = \int_{r_{j-1}(t)}^{r_j(t)} u(r, t) r^{\alpha-1} \, dr,$$

$$c_{j+} \quad 0$$

Since $\tilde{r}_1(t) = r_1(t)$ by symmetry,

$$\frac{r_1(t)}{}\psi_0(t) + \frac{r_1(t)^{\ +2}}{+2}k \quad = \quad \frac{r_1(0)}{}\psi_0(0) + \frac{r_1(0)^{\ +2}}{+2}k:$$

We assume that $r_1(t)k << 1$. Then

$$\psi_0(t) \qquad \frac{r_1(0)}{r_1(t)}\psi_0(0);$$

which gives

$$u_0^{m+1} = \quad (r\ )_1^0$$

x_N ,

$$E_N(u) = \sqrt{\frac{\sum_{i=0}^{10}(u_{2^{\hat{N}-1};i;\hat{N}} - u_{2^{\hat{N}-1};i;\hat{N}})^2}{\sum_{i=0}^{10}(u_{2^{\hat{N}-1};i;\hat{N}})^2}}; \qquad E_N(x_N) = \frac{(x_{N;\hat{N}} - x_{N;\hat{N}})}{(x_{N;\hat{N}})}; \qquad (4.58)$$

for $\hat{N} = 1;:::;6$ (i.e. $N = 10;20;40;80;160;320$). We investigate the hypothesis that

$$E_N(u) \quad \frac{1}{N^p} \quad \text{and}$$

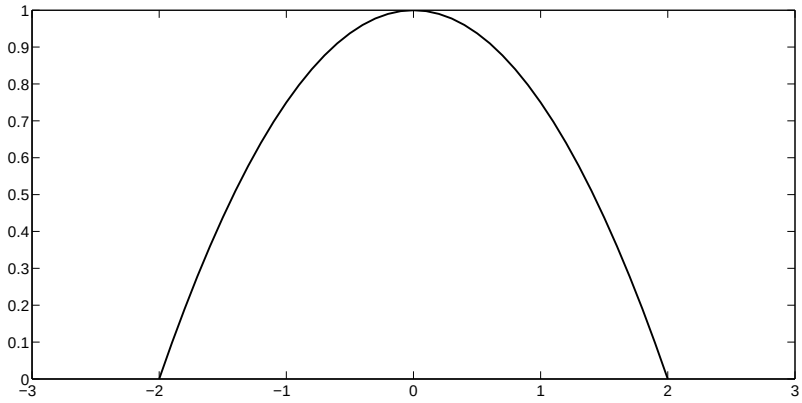
conditions given by (4.57) and then $n = 2; 3$ cases are given by

$$n = 2: \qquad \tilde{u}_j(1) = 1 - \frac{\tilde{x}_j(1)^2}{4}^{\frac{1}{2}}; \qquad \tilde{x}_{20}(1) = 2; \tag{4.60}$$

$$n = 3: \qquad \tilde{u}_j(1) = 1 - \frac{3\tilde{x}_j(1)^2}{10}^{\frac{1}{3}}; \qquad \tilde{x}_{20}(1) = \sqrt{\frac{10}{3}}: \tag{4.61}$$

has an infinite gradient at the boundaries. This is portrayed in Figures 4.4(a) and 4.5(a) which show the gradient of the solution near the moving boundary is very large for $n = 2, 3$, resulting in a less accurate approximation near the moving boundary than for the $n = 1$ case, shown in Figure 4.3. Hence, from Figure 4.8, it is reasonable to deduce that a good graphical indication of the accuracy of our method is to compare the exact and numerical outer boundary movement, as shown in the second plot of Figures 4.3–4.5. These plots show that the exact boundary position is slightly larger than the

Chapter Four


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
−3
−2
−1
0
1
2
3


−2.5
−2

Chapter Four

Chapter Four

cases are given by

$$n = 2: \qquad \dot{u}_j(1) = 1 - \frac{r_j(1)^2}{6}^{\frac{1}{2}}; \qquad x_{20}(1) = \sqrt{6}; \tag{4.67}$$

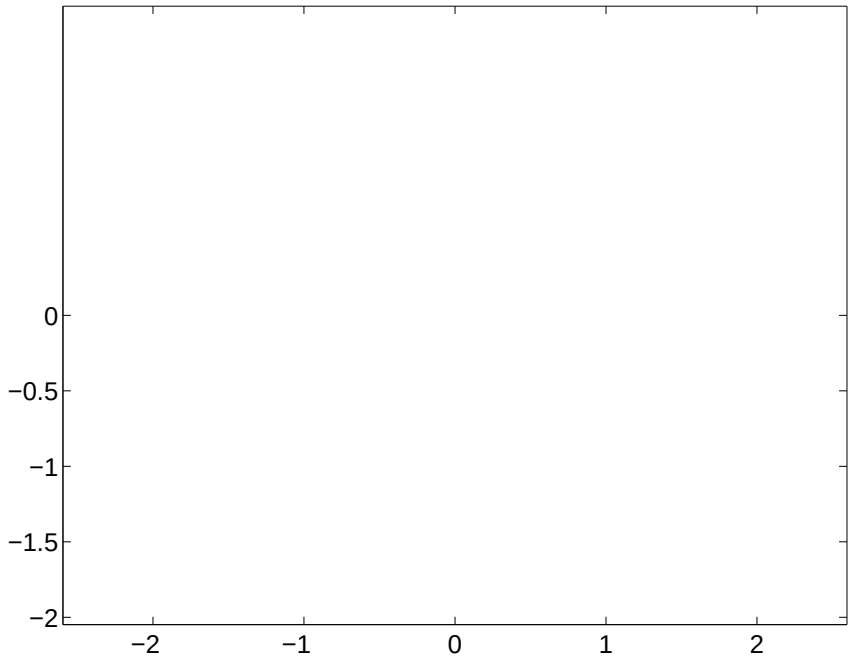
$$n = 3: \qquad \dot{u}_j(1) = 1 - \frac{3r_j(1)^2}{16}^{\frac{1}{3}}; \qquad x_{20}(1) = \frac{4}{\sqrt{3}}: \tag{4.68}$$

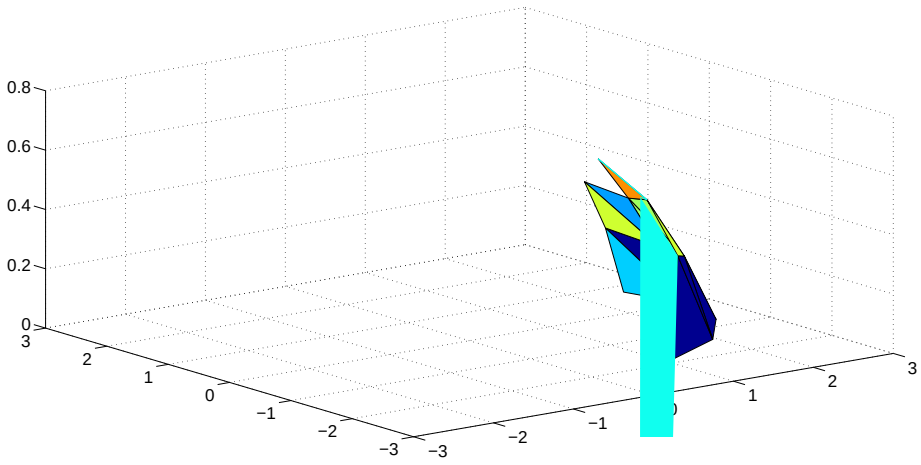
The Figure 4.12 confirm the findings from the one-dimensional case, that as n increases, the boundary slope is closer to infinity. Note that for $n = 3$ a smaller Δt was required. Although it appears that the mesh is coarser in the radial direction at the boundary, we observe from Figure 4.13 that this is not actually the case.

We have completed our application of our finite difference moving mesh method applied to the PME. In the next section we present our results from applying the finite element moving mesh method of Baines, Hubbard and Jimack [5] to the PME.

4.8 Finite elements

As mentioned in §2.3, the moving mesh method we use is a one-dimensional finite difference version of the multi-dimensional Conservation Method given in [5], which uses linear finite elements. For completeness we also solved the PME on a circle using the Conservation


0
−0.5
−1
−1.5
−2
−2
−1
0
1
2


0.8
0.6
0.4
0.2
0
3
2
1
0
−1
−2
−3
−3
−2
−1
1
2
3

5

Richards' Equation

5.1 Introduction

Richards' equation is a non-linear PDE which models the movement of moisture in an unsaturated porous medium. It was formulated in 1931 by Richards' [83] in the form

as given in [10]. This is the mixed form of Richards' equation. In our work we focus on the -based form

$$\frac{@}{@t} = \frac{@}{@z} D( ) \frac{@}{@z} + \frac{@K}{@z};$$

where the di usion coe cient

$$D( ) = K( ) \frac{d}{d} \quad \text{and} \quad / \frac{1}{-}:$$

This is the form of Richaion h Ih. Td ET Q55e tn2m ( )Tj 6.632492351(n16[(a)0.046.63233867(q)-0.20386

similar properties.

In the next section we show how Richards' [83] originally derived (5.1) from Darcy's Law to model the capillary conduction of liquids in a porous medium. We specialise the derivation presented in [83] to the one-dimensional case since we focus our work on the case where fluid is flowing in one direction only.

The second half of this chapter applies a moving mesh method. In §5.5 we use the mass conservation finite difference method, and update the mesh using both an explicit and semi-implicit time-stepping scheme. However, for Richards' equation, the mesh velocity from our mass conservation method cannot be arranged so that (3.39) holds, so the proof of monotonicity under semi-implicit time-stepping does not hold in this case. An alternative approach to define the mesh velocity (similar to that in §3.3) is explored so that (3.39) does hold. Numerical results from the finite difference methods are given in §5.7. In §5.8 the finite element method is applied to the one-dimensional case, corresponding to that presented in [90]. We give the results from the one-dimensional finite element method in §5.8.

5.2 Deriving Richards' equation

We describe the one-dimensional derivation of Richards' equation, derived from [83], where the equation of continuity and Darcy's Law are applied to the flow of liquid through a porous medium.

(i) Equation of continuity

Lemma 5.3.1 Richards' equation (5.3) conserves mass in time.

Proof To prove that the total mass does not change over time we show that the derivative of the total mass (in time) is zero. Using the Leibnitz integral rule,

$$\frac{d}{dt}\int_{a(t)}^{b(t)} u(x;t)\,dx = \int_{a(t)}^{b(t)} \frac{@u}{@t}dx + u(b;t)\frac{db}{dt} \quad u(a;t)\frac{da}{dt}:$$

Substituting $\frac{@u}{@t}$

The $u(a;t)$ and $u(b;t)$ terms vanish due to the boundary conditions (5.4). This leaves only the integral on the right-hand side, which can be rearranged such that

$$\begin{aligned}\frac{d}{dt}\int_{a(t)}^{b(t)} x\,u(x;t)\,dx &= \frac{1}{n+1}\int_{a(t)}^{b(t)} \frac{\partial}{\partial x}u(x;t)^{n+1}\,dx - \int_{a(t)}^{b(t)} u(x;t)^n\,dx;\\ &= \frac{1}{n+1}\left[u(b;t)^{n+1} - u(a;t)^{n+1}\right] - \int_{a(t)}^{b(t)} u(x;t)^n\,dx.\end{aligned}$$

Once more, the $u(a;t)$ and $u(b;t)$ terms vanish due to the boundary conditions (5.4), leaving

$$\frac{d}{dt}\int_{a(t)}^{b(t)} x\,u(x;t)\,dx = -\int_{a(t)}^{b(t)} u(x;t)^n\,dx < 0;$$

which is strictly negative for $u > 0$, indicating that the centre of mass moves in the direction of the negative x-axis. □

By showing that the mass is conserved, we know that updating the total mass at each time-level is not required when applying our moving mesh method. In addition, knowing that the centre of mass moves in one direction allows us to check that our numerical solution is exhibiting expected behaviour. Before numerically solving Richards' equation, we seek a self-similar solution to Richards' equation in the next section.

5.4 A self-similar solution

In this section we describe a class of exact solutions to Richards' equation (5.3) that are invariant under a scaling group in the variables $(t;x;u)$, and therefore take the so-called self-similar form. A definition of self-similarity, from [11], is given in §4.4 where we derived a self-similar solution for the PME. We use the same procedure from §4.4, there

Putting together (5.18) and (5.19) gives an equation for in terms of ,

When discretising (5.28) we consider two ways to approximate the derivative. The first gives

$$v_j^m = \frac{1}{n-2} \frac{(u^{n-2})_{j+1}^m - (u^{n-2})_j^{m-1}}{x^m}$$

5.5.4 Time-stepping schemes

Explicit schemes

The simplest method to time-step the mesh is the first order explicit Euler time-stepping scheme,

$$\frac{x_j^{m+1} - x_j^m}{t} = v_j^m.$$

We substitute for v_j^m

Our moving mesh method moves the nodes such that partial mass of the solution are conserved, see equation (3.6) in §3.1. Bearing this in mind we define initial masses, which remain unchanged in time, as

$$D_{j+} = (u^{n-2})^0_{j+\frac{1}{2}} \, x^0_{j+} = (u^{n-2})^m_{j+\frac{1}{2}} \, x^m_{j+}; \tag{5.33}$$

$$D_j = (u^{n-2})^0_{j-\frac{1}{2}} \, x^0_j = (u^{n-2})^m_{j-\frac{1}{2}} \, x^m_j; \tag{5.34}$$

thus simplifying equation (5.32) to

$$\frac{x^{m+1}_j - x^m_j}{t} = \frac{D_{j+}(x^{m+1}_j - x^{m+1}_{j-1}) - D_j (x^{m+1}_{j+1} - x^{m+1}_j)}{(n-2)\, x^m_j \, x^m_{j+} \, x_{j*}}$$

From (5.41) we can determine $_j(t^{m+1}$

scheme enabling $x_1^{m+1} = x_1^{m+1} \quad x_0^{m+1}$ and $x_{N\ 1+}^{m+1} = x_N^{m+1} \quad x_{N\ 1}^{m+1}$ to be determined.

Rearranging (5.46), and expanding the x_j^{m+1} terms gives

$$\frac{x_j^{m+1} \quad x_j^m}{t} = \frac{1}{(n \quad 2)(\ x)_j^m \quad x_{j+}^m \quad x_j^m} (u^{n\ 2})_{j+\frac{1}{2}}^m \quad x_{j+}^m (x_j^{m+1} \quad x_{j\ 1}^{m+1}) \quad (u^{n\ 2})_{j\ \frac{1}{2}}^m \quad x_j^m (x_{j+1}^{m+1} \quad x_j^{m+1}) \ : \tag{5.47}$$

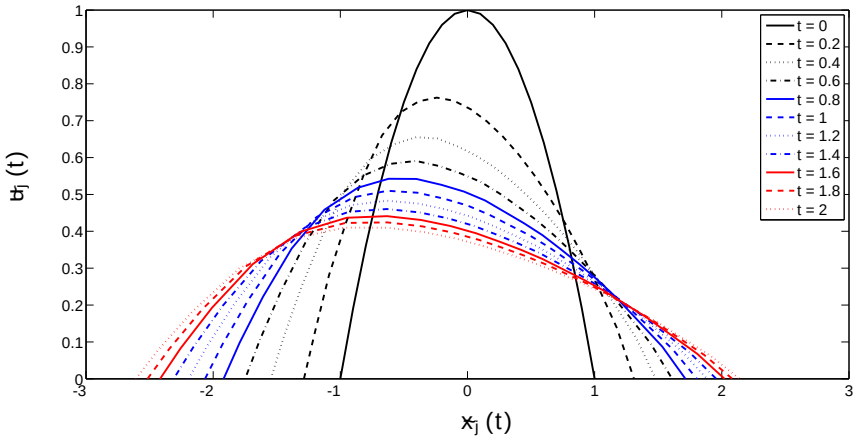
Equation (5.47) is nearly identical to our earlier semi-implicit scheme for Richards' equation (5.32), the difference being that (5.47) does not have the last term present in (5.32). Subsequently, the updated mesh x_j^{m+1} is derived by solving the matrix system

$$Ax^{m+1} = x^m;$$

where $x^{m+1} = (x_1^{m+1}; \quad x_{N\ 1}^{m+1})^T$, $x^m = (x_1^m; \quad x_{N\ 1}^m)^T$

the moving mesh method given in §5.6 with semi-implicit time-stepping;

as the number of nodes N increases and Δt decreases. We solve for $t \in [0, 0.5]$ and compute results for $N = 10 \cdot 2^{\hat{N}-1}$, $\hat{N} = 1, \ldots, 5$. We use the same notation given in §4.7 for the PME to compute both $x_{2^{\hat{N}-1}i,\hat{N}}$ and $u_{2^{\hat{N}-1}i,\hat{N}}$


t = 0
t = 0.2
t = 0.4
t = 0.6
t = 0.8
t = 1
t = 1.2
t = 1.4
t = 1.6
t = 1.8
t = 2
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3

(a) Using

5.8.1 Numerically solving Richards' equation using nite e lements

We solve the equations for the nite element velocity potential (5.50), the nite element velocity (5.51), and the nite element solution (5.52) as systems of equations using piecewise linear expansions, $= \sum_j \ _j$, $U = \sum U_j \ _j$, $V = \sum V_j \ _j$, which we substitute into each equation in turn, with $w_i = \ _i$.

The equation for the nite element velocity potential (5.50) becomes

$$\int_{a(t)}^{b(t)} U \left[\sum_{j=0}^{N} \ _j \frac{@ \ _j}{@x} \right] \frac{@ \ _i}{@x} dx = \int_{a(t)}^{b(t)} \frac{@ \ _i}{@x} \left[U^n \ ^2 \sum_{j=0}^{N} U_j \frac{@ \ _j}{@x} + U^n \right] dx; \qquad (5.53)$$

where U, and are piecewise linear forms of u, and w respectively. In (5.53) it is convenient to substitute the summation for U_x, but not U. Interchanging the summation and integral gives

$$\sum_{j=0}^{N} \ _j \left[\int_{a(t)}^{b(t)} \frac{@ \ _j}{@x} \frac{@ \ _i}{@x} U \, dx \right] = \sum_{j=0}^{N} U_j \left[\int_{a(t)}^{b(t)} \frac{@ \ _j}{@x} \frac{@ \ _i}{@x} U^n \ ^2 dx \right] - \int_{a(t)}^{b(t)} \frac{@ \ _i}{@x} U^n \, dx; \qquad (5.54)$$

for $i = 0; 1; : : : ; N$. In [5]

$$K_{ij}(U) = \int_{a(t)}^{b(t)} U \frac{@ \ _i}{@x} \frac{@ \ _j}{@x} dx; \qquad S_i(U) = \int_{a(t)}^{b(t)} \frac{@ \ _i}{@x} U^n \, dx;$$

are de ned. Using these de nitions (5.54) can be written

$$\sum_{j=0}^{N} K_{ij}(U) \ _j = \sum_{j=0}^{N} K_{ij}(U^n \ ^2) U_j \quad S_i(U): \qquad (5.55)$$

Hence, to determine the mesh velocity potential at time $t = t^m$ we solve the matrix system

$$K(U) \quad = \quad K(U^n \ ^2) U \quad S(U); \qquad (5.56)$$

where $K(U^n) = K(U^n)^m$ is the sti ness matrix de ned by (5.55), $S(U) = S(U)^m$ is the vector of $S_i(U)$ values, $= \ ^m$ is the vector $(\ ^m_1; : : : ; \ ^m_N))^T$, and $U = U^m$ is the vector $(U^m_1; : : : ; U^m_N))^T$. We solve (5.56) to nd the velocity potential vector .

By the same process, and fromT /R506 7.410.9016677046524(:)5(t)0.311076(i)-0.247294(o)0..2

where $M = M^m$

5.8.2 Numerical details

We provide speci cs to the general algorithm given in the thelast section to solve Richards' equation. We use nodes x_j^m, $j = 0; 1; : : : ; N$, and time-step t.

Preliminaries

To begin we determine the constant vector$g$ from (5.60) using the initial mesh and initial conditions. Using an initial equispaced mesh we calculate the initial mass matrix M^0

The velocity potential ^{m} is substituted into (5.57) to give the mesh velocity V^{m}

$$V^{m} = (M^{m})^{1} H^{m} \quad ^{m}; \tag{5.65}$$

where the vector $H^{m} \quad ^{m}$

conserves mass). Although, this is not the case when the mesh is moved by balancing the partial masses with the flux term.

We showed that the mass of the solution remains constant, and that the centre of mass moves in one direction. Knowing that the centre of mass moves in one direction allows us to check that our numerical solution is exhibiting the expected behaviour.

6

The Crank-Gupta Problem

6.1 Introduction

In [38] Crank and Gupta introduced the so-called oxygen-consumption problem for the evolution of oxygen concentration in a tissue, in which oxygen is absorbed at a prescribed

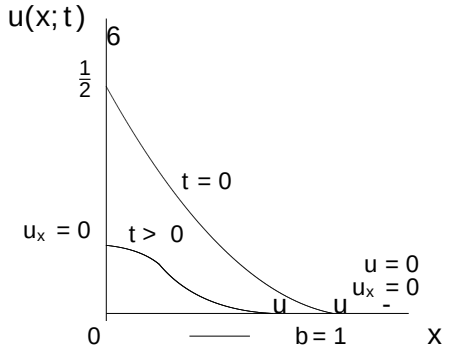


Fig. 6.1: Diagrammatic representation of the Crank-Gupta solution.

Oxygen in the region 0=9 61688 l S 2663.46 6168.574.(0)]TJ /R 612351]T368(a)0.0492sR 612353 2351

PDE (6.1) into the variables and to obtain an ODE. The left-hand side of (6.1) is the

where $\in [0;1]$.

To find a solution to (6.13) we consider a series solution of the form

$$s = \sum_{k=0}^{\infty} c_k \, ^k; \tag{6.14}$$

where c_k are constants to be determined. By substituting (6.14), and its derivatives, into (6.13) we achieve the equation

$$2c_2 \quad c_0 + \sum_{k=1}^{\infty} (k+2)(k+1)\, c_{k+2} + \frac{k}{2} c_k \quad c_k \quad ^k = 0: \tag{6.15}$$

For (6.15) to hold, each coefficient of ^{k}, $k = 0;1;:::$ must be zero, hence

$$\begin{aligned} &\text{for } k = 0; && 2c_2 \quad c_0 = 0; \\ &\text{for } k = 1;2;3:::; && (k+2)(k+1)\, c_{k+2} + c_k \quad \frac{k}{2} \quad 1 \quad = 0: \end{aligned}$$

Thus,

$$c_2 = \frac{c_0}{2}; \tag{6.16}$$

$$c_{k+2} = \frac{c_k \quad 1 \quad \frac{k}{2}}{(k+2)(k+1)}: \tag{6.17}$$

Substituting $k = 2$ into (6.17) gives

$$c_4 = \frac{c_2 \quad 1 \quad \frac{2}{2}}{3 \quad 4} = 0:$$

Thus, by induction, $c_k = 0$ for even $k \quad 4$. Substituting for odd values of k gives

$$k = 1: \qquad c_3 = \frac{c_1}{2 \quad 2}$$

2702 4(TJ /R146 10.9091 Tf 5.45391 0 Td [(2)0.0492351]TJ /R

c2

Let $k = 2m + 1$, $m = 1; 2; 3; : : :$, such that for $m = 1; 3; 4; 5; 6$ ($k = 3; 5; 7; 9; 11$) we observe that

$$c_{2m+1} = \frac{(\quad 1)^{m+1}}{2^{2m\quad 1}} \quad \frac{(2m\quad 2)!}{(2m+1)!(\ m\quad 1)!}\quad c_1; \tag{6.18}$$

and we write (6.17) as

$$c_{2m+3} = \qquad (1\quad 2m)$$

then a_m converges.

To prove (1) we take b_m,

$$b_m = \frac{(2m-2)!}{2^{2m-1}(2m+1)!(m-1)!};$$
$$= \frac{1}{2^{2m-1}(2m+1)(2m)(2m-1)(m-1)!}.$$

We immediately observe that as $m \to 1$, $b_m \to 0$.

To prove (2) we note that

$$\frac{b_{m+1}}{b_m} = \frac{2^{2m-1}(2m+1)!(2m)!(m-1)!}{2^{2m+1}(2m+3)!(2m-2)!m!} = \frac{2m(2m-1)}{4m(2m+3)(2m+2)}.$$

Since $4m(2m+3)(2m+2) > 2m(2m-1)$,

$$\frac{b_{m+1}}{b}$$

6.3.2 Determining the mesh velocity

The total mass is updated with the mesh, where the mesh velocity is given by substituting (6.23){(6.25) into (3.23), and evaluating the integral, so that

$$v_j(t) = \frac{1}{u_j(t)}\,\lambda(t)c_j - \int_0^{x_j(t)} \left(\frac{\partial^2 u}{\partial x^2} - 1\right) dx\,;$$

for interior points $j = 1;\ldots;N-1$. Substituting for $\lambda(t)$ from (6.28) gives

$$v_j(t) = \frac{1}{u_j(t)}\left(c_j b(t) + \frac{\partial u}{\partial x}_{x_j(t)} - x_j(t)\right). \tag{6.30}$$

We use a discretised form of (6.30),

$$v_j^m = \frac{1}{u_j^m}\left(c_j x_N^m + \frac{u_{j+1}^m - u_{j-1}^m}{x_{j+1}^m - x_{j-1}^m} - x_j^m\right); \qquad j = 1;2;:::;N-1. \tag{6.31}$$

We note that the term in the curved brackets can be replaced by (4.37) (where $n = 1$), to give a discretisation of (6.30) which is more accurate on a non-uniform mesh, namely

$$v_j^m = \frac{1}{u_j^m}\left(c_j x_N^m + \frac{\frac{1}{x_{j+}^m}\frac{u_{j+}^m}{x_{j+}^m} + \frac{1}{x_j^m}\frac{u_j^m}{x_j^m}}{\frac{1}{x_{j+}^m} + \frac{1}{x_j^m}} - x_j^m\right); \tag{6.32}$$

for interior nodes $j = 1;2;\ldots;N-1$, where $(\,)_j^m = (\,)_j^m - (\,)_{j-1}^m$ and $(\,)_{j+}^m = (\,)_{j+1}^m - (\,)_j^m$. The new mesh x_j^{m+1}, $j = 1;\ldots;N-1$, is obtained from v_j^m by a time-stepping scheme.

At the outer boundary $u(b;t) = 0$ from (6.25), so we seek an alternative method to determine x_N^m. One approach is to extrapolate the boundary velocity v_N^m from the internal velocities, and then update the position of the outer node along with the internal nodes. However, extrapolation sometimes produces a numerical solution with a boundary that moves out (the boundary should move in [38]). As an alternative, we consider the asymptotic behaviour of the solution near the outer boundary. At the outer boundary $u = 0$ so $u_t = 0$, reducing (6.1) to

$$\frac{\partial^2 u}{\partial x^2}_{x=b} = 1.$$

The Taylor expansion for $u(x)$ about $x = b$ is

$$u(x) = u(b) + (b-x)\frac{\partial u}{\partial x}_{x=b} + \frac{(b-x)^2}{2}\frac{\partial^2 u}{\partial x^2}_{x=b} + \ldots;$$

hence,

$$u(x;t) \quad \frac{1}{2}(x_j \quad b(t))^2; \tag{6.33}$$

close to $b(t)$. Therefore, for the discrete case where $j = N \quad 1$ and $t = t^{m+1}$, we make the approximation

$$u_{N \quad 1}^{m+1} \quad \frac{1}{2}(x_{N \quad 1}^{m+1} \quad x_N^{m+1})^2;$$

which gives the following formula for the outer node,

$$x_N^{m+1} = x_{N \quad 1}^{m+1} + \sqrt{2u_{N \quad 1}^{m+1}}; \tag{6.34}$$

taking the positive square root.

6.3.3 Recovering the solution

Once the updated mesh x_j^{m+1} has been determined, the updated solution u_j^{m+1}, $j = 1;\ldots;N \quad 1$, is given by either (3.25) for a uniform mesh, or (3.26) for anon-uniform mesh. The solution at the inner boundary u_0^{m+1} is calculated using

$$u_0^{m+1} = \frac{m+1}{0} \frac{x_1^0 u_0^0}{x_1^{m+1}};$$

f 0u 21 8J EET Q 311076(a)0.0492351(k)-0.222677(i1218(o)0.0492351(n5Td [(u)-0.096232(2)0.049 8J EET Q nd)0.330921 1436(c)0.1317 304824

from (3.25) and the boundary condition ((s)0.0906373(e)-343.448(7681]TJ T*[(0)0.5176851(n)-222.406((

6.3.5 Time-stepping schemes

Explicit schemes

The simplest method to time-step the mesh is the first order explicit Euler time-stepping scheme,

$$\frac{x_j^{m+1} - x_j^m}{t} = v_j^m;$$

for $j = 1; : : : ; N - 1$. We substitute for v_j^m from (6.31) or (6.32). At the inner boundary $v_0^m = 0$, and at the outer boundary we use (6.34). The explicit Euler time-stepping scheme requires small t so that the x_j^m remain stable, and to avoid mesh tangling. We also implemented the adaptive predictor-corrector Runge-Kutta methods in Matlab. We used the solver, ODE15s, which is designed to solve a stiff system. Implementing this solver produced results similar to results from ODE23 and ODE45 (which are not designed for stiff systems), inferring that the method does not lead to a stiff system.

A semi-implicit scheme

The semi-implicit scheme described in §3.4.2 updates the mesh x_j^m only. Thus, for a method that requires the total mass $\ $ to be updated as well, we cannot use it in the same way, since $\ $ would have to be updated separately (for example, using the explicit Euler time-stepping

to polar coordinates using $r^2 = x^2 + y^2$ giving

$$\frac{@u}{@t} = \frac{1}{r}\frac{@}{@r}\left(r\frac{@u}{@r}\right) \quad 1; \tag{6.35}$$

with boundary conditions

$$\frac{@u}{@r} = 0 \qquad \text{at} \qquad r = 0;\ t > 0; \tag{6.36}$$

$$u = 0; \quad \frac{@u}{@r} = 0 \qquad \text{at} \qquad R(t);\ t > 0; \tag{6.37}$$

and initial conditions

$$u = \frac{1}{2}(1 \quad r)^2; \qquad r \in [0;1];\ t = 0: \tag{6.38}$$

As with the one-dimensional case, we introduce the dependent variable $r_j(t)$, $j = 0;:::;N$, to represent the $N+1$ nodes on the radius of the mesh, which are dependent on t. The mesh is initially equally-spaced. We define the velocity of the j-th node $s(r_j;t)$ on the radius to be

$$s(r_j;t) = \dot{r}_j(t) = \frac{dr_j}{dt}:$$

We assume conservation of relative mass (as with the one-dimensional case) such that

$$d_j = \frac{1}{(t)}\int_0^{r_j(t)} u(r;t)r\,dr; \tag{6.39}$$

where d_j is a constant in time, (t) is the total mass

$$(t) = \int_0^{R(t)} u(r;t)r\,dr; \tag{6.40}$$

and $R(t)$ is the outer boundary. Specifically, for initial conditions (6.38),

$$(0) = \int_0^1 (1 \quad r)^2 r\,dr = \frac{7}{12};$$

and

$$d_j = \frac{12}{7}\int_0^{r_j(0)} (1 \quad r)^2 r\,dr = \frac{12}{7}r_j(0)^2 \quad \frac{8}{7}r_j(0)^3 + \frac{3}{7}r_j(0)^4:$$

Given a mesh $x_j(t)$ and solution $u_j(t) = u(x_j(t);t)$, we can evaluate the total mass (t) directly from (6.40). To evaluate an updated value of the total mass (which is required for determining the updated solution) we compute $\dot{}(t)$, and then approximate the total

Substituting for $\frac{@u}{@t}$ from (6.35), and cancelling the last term due to the boundary condition (6.36),

$$-(t)d_j = \int_0^{r_j(t)} \frac{@}{@r}\left(r\frac{@u}{@t}\right) r\,dr + u_j(t)r_j(t)s_j(t):$$

Evaluating the integral and using the boundary condition (6.36),

$$-(t)d_j = r_j(t)\left(\frac{@u}{@r}\right)_{r_j(t)} \frac{r_j(t)^2}{2} + u_j(t)r_j(t)s_j(t):$$

Hence, the radially symmetric Crank-Gupta mesh velocity is given by

$$s(r_j;t) = \frac{}{r}\left(\frac{@u}{}\right.$$

0 5259.90 h 138008 Td 0 Td [2]TJ /R509 154 -0.248413 (m) 9 10.9091 Tf -334.

via the Taylor series. This is the same as the Cartesian case (6.33), therefore we can de ne the outer node by

$$r_N^{m+1} = r_{N-1}^{m+1} + \sqrt{\frac{\quad}{2u_{N-1}^{m+1}}}; \qquad (6.44)$$

taking the positive square root.

6.4.3 Recovering the solution

To approximate the updated solution u_j^{m+1}, we equate (6.39) at times $t = t^{m+1}$ and $t = 0$ between the points r_{j+1} and r_{j-1},

$$\frac{1}{(t^{m+1})} \int_{r_{j-1}^{m+1}}^{r_{j+1}^{m+1}} u(r; t^{m+1}) r \, dr = \frac{1}{(0)} \int_{r_{j-1}^{0}}^{r_{j+1}^{0}} u(r; 0) r \, dr:$$

Approximating the integrals by the mid-point rule,

$$u_j^{m+1} = \frac{{}^{m+1}}{{}^{0}} \quad r_{j+1}^0 \quad r_j^0 \ldots$$

6.5.1 Determining the rate of change of total mass

The total mass is updated using an explicit time-stepping scheme, where is given by (6.1), (6.3) and (6.48) substituted into (3.21),

$$\dot{\alpha}(t) = \int_0^{b(t)} \frac{\partial^2 u}{\partial x^2} \, 1 \, dx = 1 \quad e^{t} \quad 1 \quad b(t):$$

Comparing this to (6.28) from the original problem, we observe that 1 e

of (6.52)

$$v_j^m = \frac{1}{u_j^m}\left(c_j - \dot{m}\left(\frac{\frac{1}{x_{j+}^m}\frac{u_{j+}^m}{x_{j+}^m} + \frac{1}{x_j^m}\frac{u_j^m}{x_j^m}}{\frac{1}{x_{j+}^m} + \frac{1}{x_j^m}}\right) + x_j^m + e^{t^m - t^{m-1}}\right), \qquad (6.54)$$

where $(\ )_j^m = (\ )_j^m - (\ )_{j-1}^m$ and $(\ )_{j+}^m = (\ )_{j+1}^m - (\ )_j^m$, and $v_0^m = 0$ at the inner boundary. At the right boundary, $u(b,t) = 0$, as for the original Crank-Gupta problem, so we seek an alternative method to determine x_N^m. Again, either v_N^m is extrapolated from the internal velocities using a polynomial, or equation (6.34) is employed since the outer boundary is the same as with the original problem.

The new mesh x_j^{m+1} is obtained from v_j^m by a time-stepping scheme.

6.5.3 Recovering the solution

The solution is recovered in the same manner as with the original Crank-Gupta problem in Chapter 6.3, i.e. by either (3.25), for a uniform mesh, or (3.26), for a non-uniform mesh. The solution at the inner boundary u_0^{m+1} is calculated using a one-sided approximation of (3.25)

$$u_0^{m+1} = \frac{m+1}{0}\frac{(x_1^0 - x_0^0)u_0^0}{x_1^{m+1} - x_0^{m+1}}.$$

At the outer boundary, $u_{N+1}^{m+1} = 0$ from the zero boundary condition (6.25).

6.5.4 The full algorithm

Given a total mass m, mesh r_j^m, and solution u_j^m, $j = 0,\ldots,N$, at t^m, $m \geq 0$:

Compute the rate of change of mass $\dot{m}$ from (6.51);

Compute the mesh velocity v_j^m from (6.53) or (6.54);

Compute the updated mesh x_j^{m+1} by a time-stepping scheme;

Compute the updated solution u_j^{m+1} from (3.25) or (3.26).

The last case we consider is the original Crank-Gupta problem where we move the nodes to preserve partial mass balances, as in §3.3.

6.6 A partial mass balance moving mesh method

The Crank-Gupta PDE has a source term so we can use the moving mesh method described in §3.3, with the same notation, i.e. $x_j(t^m) \equiv x_j^m$ denotes the jth node of the mesh with $N+1$ nodes, at time $m\,t$, $m = 0, 1, \ldots$, and $u_j^m \equiv u_j(t^m)$ and $v_j^m \equiv v_j(t^m)$ denote the solution and mesh velocity at these nodes. The partial masses of the solution at $m\,t$ are $\mu_j^m \equiv \mu_j(t^m)$.

We model the Crank-Gupta problem by the PDE (3.27) with

$$Hu \equiv \frac{\partial^2 u}{\partial x^2} \quad \text{and} \quad S(x,t) \equiv 1, \quad a(t) = 0 \le x \le b(t), \tag{6.55}$$

from (6.1). Hence, the mass balance relation (3.29) is

$$\frac{d}{dt}\int_0^{x_j(t)} u(x,t)\,dx = \int_0^{x_j(t)} 1\,dx.$$

This relation implies that given a mesh $x_j(t^m)$, with corresponding solution $u_j(t^m)$ and partial masses $\mu_j(t^m)$, we can calculate the updated partial masses $\mu_j(t^{m+1})$, mesh $x_j(t^{m+1})$ and solution $u_j(t^{m+1})$ by computing the rate of change of partial masses $\dot{\mu}_j(t^m)$ and mesh velocity $v_j(t^m)$.

We use the same initial solution from [38], namely

$$u(x,0) = \frac{1}{2}(1-x)^2, \tag{6.56}$$

for $x \in [0,$

6.6.3 Recovering the solution

The solution is updated using the approach given in §3.3, namely, by either (3.34) or (3.35), the latter being more accurate for a non-uniform mesh. At the outer boundary, $u_{N+1}^{m+1} = 0$ from (6.2). At the inner boundary we use the trapezoidal approximation

$$\alpha_0^{m+1} = \frac{1}{2}(u_1^{m+1} + u_0^{m+1})x_1^{m+1};$$

thus

$$u_0^{m+1} = \frac{2\,\alpha_0^{m+1}}{x_1^{m+1}} - u_1^{m+1}.$$

6.6.4 The full algorithm

Given the partial masses α^m, mesh x_j^m, and solution u_j^m, $j = 0,\ldots,N$, at t^m, $m \geq 0$:

Compute the rate of change of partial masses $\dot{\alpha}_j^m$ from (6.58);

Compute the mesh velocity v_j^m from (6.60) or (6.61);

Compute the updated mesh x_j^{m+1} by a time-stepping scheme;

Compute the updated solution u_j^{m+1} from (3.34) or (3.35).

In the last four sections we have given the details for applying the moving mesh method to the Crank-Gupta problem. In the next section we present the numerical results.

6.7 Numerical results

We have looked at solving the Crank-Gupta problem

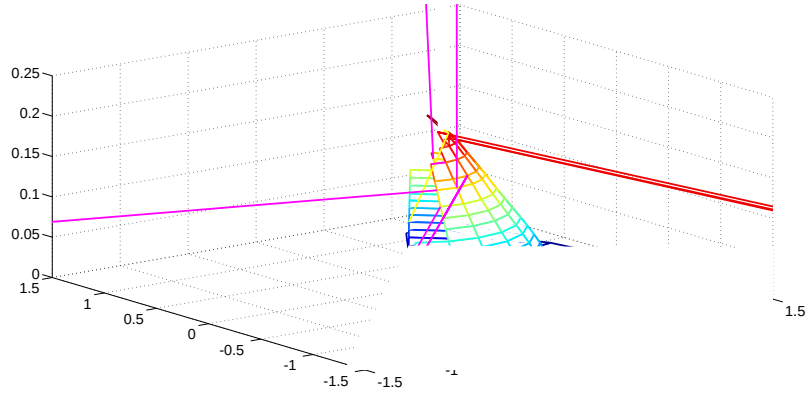
in the one-dimensional case, using the moving mesh method which preserves partial mass fractions (see §6.3);

in the radially symmetric case, using the moving mesh method which preserves partial mass fractions (see §6.4);

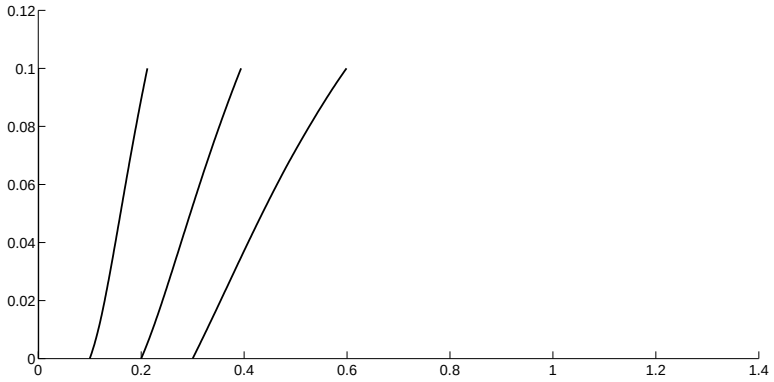
in the one-dimensional case, using alternative boundary conditions and the moving mesh method which preserves partial mass fractions (see §6.5);

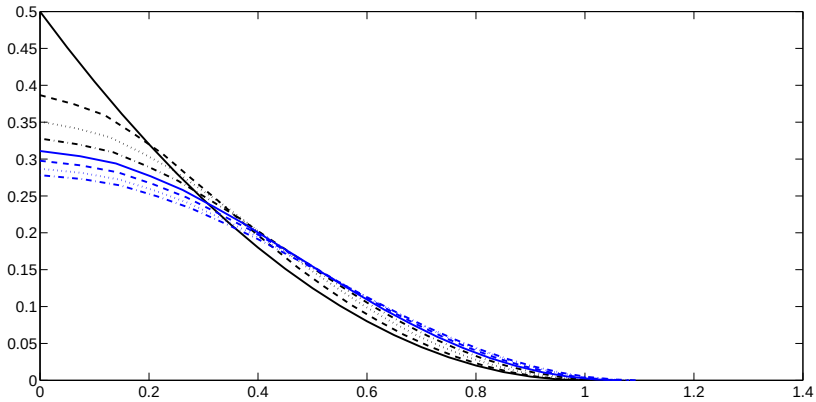
in the one-dimensional case, using the moving mesh method which preserves partial mass balance (see §6.6).

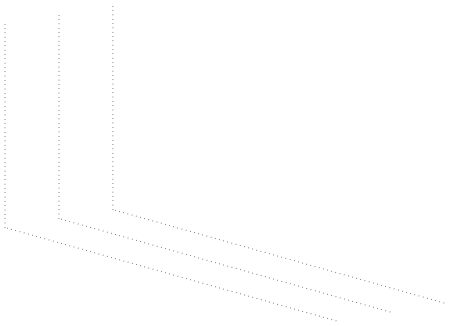
The Crank-Gupta Problem


0.25
0.2
0.15
0.1
0.05
0
1.5
1
0.5
0
-0.5
-1
-1.5
-1.5
1.5

decreases as N


0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4


0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4



an exact solution. Both comparisons indicated convergence and accuracy of our moving mesh method that conserves partial masses. It appears that the mesh and solution have second-order convergence. There was no analysis of the method that balances the partial masses with the source term since we found that the boundary moves out very rapidly, which is highly inaccurate, so we conclude that it is not a suitable method to solve the

7

A Tumour Growth Problem

Indeed, it has been noted that a conceptual framework within which all these new (and old) data can be fitted is lacking [49]. In [47] it states that `clinical oncologists and tumour biologists posses virtually no comprehensive theoretical model to serve as a framework for understanding, organising and applying these data'. By being educated as to which mechanisms are critical to the essence of tumour growth, these could possibly be manipulated to our advantage. As Byrne [30] remarks, `In order to gain such insight, it is usually necessary to perform large numbers of time-consuming and intricate experiments - but not always. Through the development and solution of mathematical models that describe different aspects of solid tumour growth, applied mathematics has the potential to prevent excessive experimentation'.

Ideally, experiments and modelling work hand-in-hand. The experiments can not only prove to be costly, but the subtleties of the many intricate processes can easily be

growth.

Whilst differences between such models exist, many exhibit the following features:

Equations describing the diffusion of nutrients or growth factors in and around the tumour region (generally parabolic in type);

Mass transfer equations describing the dynamic variation in tumour tissue (generally hyperbolic);

Mass balance equations describing the growth of the tumour (generally elliptic).

All of these equations are generally coupled via nonlinear interactions. For instance, the

as to compare results with our moving mesh strategies, of which there are three. The details of these strategies are given in §7.6, where we solve the tumour growth model numerically using each one in turn. The results from the fixed mesh method and the three moving mesh methods are discussed in §7.7. Finally, in §7.8 we conclude that a moving mesh method can prove to be an elegant and accurate numerical approach that updates the mesh smoothly with the solution of the orginal model, whilst preserving chosen features of the model such as local mass balance, or relative partial masses. However, since the mesh depends upon the model, care must be taken when choosing a feature of the model to preserve.

Model formulation

The model assumes the tumour consists of two phases, water and live cells, which are treated as incompressible fluids whose densities are equal to leading order. The model is derived by applying mass balance to the cell and water phases. Further assumptions made are that inertial effects are negligible, no external forces act on the system, and, on the timescale of interest, the cell and water phases can be treated as viscous and inviscid fluids respectively. The model is applied to a tumour whose growth is in parallel to the x-axis, and is symmetric about its midpoint. We have altered the notation to be consistent with previous chapters.

From [22] the non-dimensional model, in Cartesian form, for the volume fraction of cells $u(x,t) \in (0,1)$, with $t > 0$ and $x \in [0, b(t)]$, where $b(t)$ is the tumour radius, is comprised of

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(wu) = \frac{(1+s_1)u(1-u)C}{1+s_1C} - \frac{s_2+s_3C}{1+s_4C}u =: S(u,C), \quad (7.1)$$

$$\frac{\partial}{\partial x}\left(u\frac{\partial w}{\partial x} - u\frac{u-u}{(1-u)^2}H(u-u\right.$$

The normalised model has initial and boundary conditions

$$b = 1; \quad u = u^0(x) \quad \text{at} \quad t = 0; \tag{7.5}$$

$$w = \frac{@C}{@x} = 0 \quad \text{at} \quad x = 0; \; t > 0; \tag{7.6}$$

$$\frac{@w}{@x} \quad \frac{u \quad u}{(1 \quad u)^2} H(u \quad u_{min}) = 0; \quad C = 1; \quad \frac{@b}{@t} = w \quad \text{at} \quad x = b; \; t > 0: \tag{7.7}$$

Remark 7.4.1 We observe that for the case of zero viscosity $= 0$, equations (7.1) and (7.2) reduce to

$$\frac{@u}{@t} = \frac{@}{@x} \quad \frac{1 \quad u}{k} \frac{@}{@x} (u) \quad + S(u; C);$$

where $(u) = u^{u \quad u}$

To determine u_N^m, we discretise the boundary condition (7.13) by taking values $()_{N\ \frac{1}{2}}^m$ and $(\ )_{N+\frac{1}{2}}^m$ (the average about $()_N^m$) to obtain

$$\frac{1}{2}\left[\frac{w_{N+1}^m - u_N^m}{} - b^m \frac{u_{N+\frac{1}{2}}^m - u}{(1 - u_{N+\frac{1}{2}}^m)^2} H(u_{N+\frac{1}{2}}^m - u_{min})\right]$$
$$\frac{1}{2}\left[\frac{w_N^m - w_{N\ 1}^m}{} - b^m \frac{u_{N\ \frac{1}{2}}^m - u}{(1 - u_{N\ \frac{1}{2}}^m)^2} H(u_{N\ \frac{1}{2}}^m - u_{min})\right] = 0: \qquad (7.16)$$

We then adapt (7.15) for $j = N$, using u_N^m (7.16) to replace the rGr53 .621( )0.331218(r)-0.5 Td [

3

5 ^m_N u^m

where

$$T^l = \frac{2}{(x_j^m - x_{j-1}^m)(x_{j+1}^m - x_{j-1}^m)};$$
$$T^d = \frac{2}{(x_{j+1}^m - x_j^m)(x_j^m - x_{j-1}^m)};$$
$$T^u = \frac{2}{(x_{j+1}^m - x_j^m)(x_{j+1}^m - x_{j-1}^m)};$$

and where $x_{-1}^m = -x_1^m$, $C_{-1}^m = C_1^m$ and $C_N^m = 1$, from the boundary conditions (7.6) and (7.7).

Step 2: Find w_j^m by applying central finite differences to (7.2) on the non-uniform mesh $\{x_0^m, \ldots, x_N^m\}$ with boundary conditions given by (7.6) and (7.7). The resulting set of equations is similar to (7.15) and takes the form

$$\frac{1}{x_{j+\frac{1}{2}}^m - x_{j-\frac{1}{2}}^m}\left\{u_{j+\frac{1}{2}}^m\left[\frac{w_{j+1}^m - w_j^m}{x_{j+1}^m - x_j^m} - \frac{u_{j+\frac{1}{2}}^m - u}{(1-u_{j+\frac{1}{2}}^m)^2}H(u_{j+\frac{1}{2}}^m - u_{min})\right]\right.$$
$$\left. - u_{j-\frac{1}{2}}^m\left[\frac{w_j^m - w_{j-1}^m}{x_j^m - x_{j-1}^m} - \frac{u_{j-\frac{1}{2}}^m - u}{(1-u_{j-\frac{1}{2}}^m)^2}H(u_{j-\frac{1}{2}}^m - u_{min})\right]\right\} = \frac{ku_j^m}{1-u_j^m}w_j^m; \quad (7.18)$$

where $x_{j+\frac{1}{2}}^m - x_{j-\frac{1}{2}}^m = \frac{1}{2}(x_{j+1}^m - x_{j-1}^m)$, $j = 1, 2, \ldots, N-1$, and $w_0^m = 0$ (from (7.6)). As with the fixed numerical mesh method, to determine the boundary value u_N we discretise the boundary condition (7.7) in a similar way to (7.16) by taking the average at $(\,)_{N-\frac{1}{2}}^m$ and $(\,)_{N+\frac{1}{2}}^m$, giving

$$\frac{1}{2}\left[\frac{w_{N+1}^m - w_N^m}{x_{N+1}^m - x_N^m} - \frac{u_{N+\frac{1}{2}}^m - u}{(1-u_{N+\frac{1}{2}}^m)^2}H(u_{N+\frac{1}{2}}^m - u_{min})\right]$$
$$+ \frac{1}{2}\left[\frac{w_N^m - w_{N-1}^m}{x_N^m - x_{N-1}^m} - \frac{u_{N-\frac{1}{2}}^m - u}{(1-u_{N-\frac{1}{2}}^m)^2}H(u_{N-\frac{1}{2}}^m - u_{min})\right] = 0. \quad (7.19)$$

We then adapt (7.18) for $j = N$ using (7.19) to replace the first term in square

Step 3: Calculate the mesh velocity v_j^m. This step will differ for each of Methods A, B and C, and is detailed below.

Step 4: Update the mesh points by the explicit Euler scheme applied to (7.17)

$$x_j^{m+1} = x_j^m + \Delta t v_j^m; \qquad j = 0, 1, \ldots, N,$$

with v_j^m obtained from Step 3.

Step 5: Calculate u_j^{m+1}. The details of this step will again differ for each method used, and are given in §7.6.1, 7.6.2 and 7.6.3 respectively.

When comparing this scheme to the fixed numerical mesh algorithm in §7.828044(e)-0.-0.248413(n)25413

of §7.5 in the sense that the transformation is effected exactly by the boundary velocity. However, when u is calculated in §7.5 using a velocity derived from the transformation, a quasi-Lagrangian form of the mass balance equation is used in which the velocity is incorporated using a chain rule. The result is an extra term which cannot be written in divergence form. By contrast, in Method A we have preferred to use an integral approach which already incorporates local conservation.

7.6.2 Method B

Under this strategy, in Step 3 the velocity of each node is determined by the cell velocity

and $a(t) = 0$, $b(t) = b(t)$. Let the total mass be

$$(t) = \int_0^{b(t)} u(x;t)\,dx;$$

as given by (3.19). We de ne c_j to be the mass fraction given by (3.19), so that

$$c_j = \frac{1}{(t)} \int_0^{x_j(t)} u(x;t)\,dx; \tag{7.26}$$

and calculate $x_j(t)$ such that c_j remains constant with respect to time.

The total mass will be required in order to approximate u, so we rst determine by substituting (7.25) into (3.22) giving,

$$(t) = \int_0^{b(t)} S(u;C) \quad \frac{@}{@x}(wu)\,dx + u_b v_b \text{ BT /R165 10.9091 Tf 1 0 0 1 30976 0 Td}$$

for $u_j(t) = u(x_j, t) \geq 0$. We use the composite trapezoidal rule on the integral to obtain a discrete form of (7.29) at time $t = t^m$,

$$v_j^m = \frac{c_j - m}{u_j^m} \quad \frac{1}{u_j^m} \sum_{i=0}^{j-1} \frac{1}{2}(x_{i+1}^m \quad x_i^m)(S_{i+1}^m + S_i^m) + w_j^m: \tag{7.30}$$

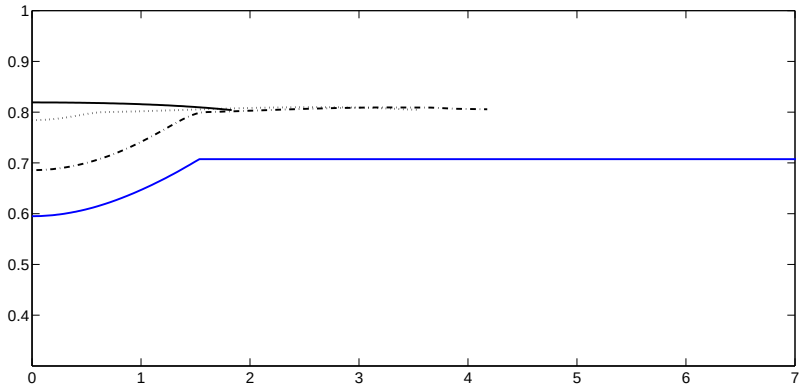
Using (7.30), the new mesh x_j^{m+1} is computed as in Step 4. To approximate the updated solution u_j^{m+1} in Step 5, we use (3.25),

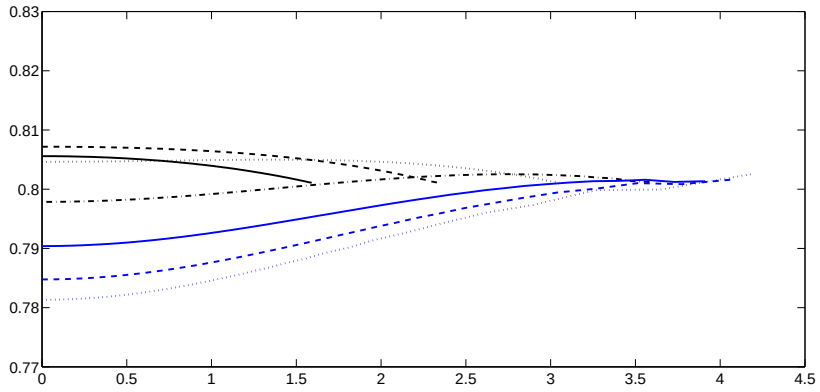
$$u_j^{m+1} = \frac{m+1}{0} \frac{x_{j+1}^0 \quad x_j^0 \quad 1}{x_{j+1}^{m+1} \quad x_j^{m+1} \quad 1} u_j^0:$$

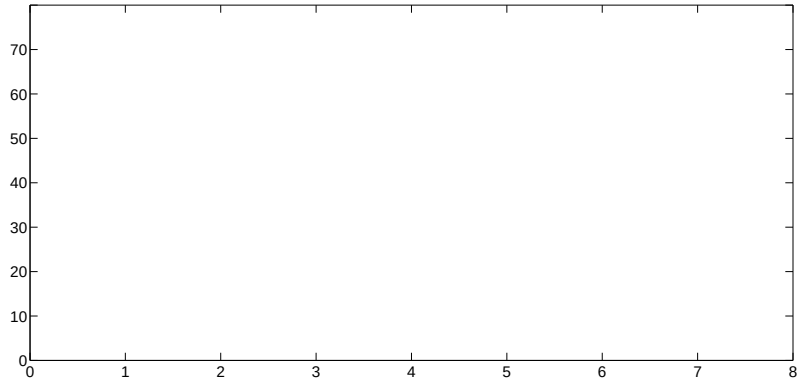
7.7 1

Method	N	$E_N(u)$	p_N	$E_N(*)$	q_N
A	10	$2{:}034 \quad 10^{-4}$			

the numerical method. The processes of Method A and Method B are very similar, and because Method A behaves as in Figure 7.5, it is reasonable to


1
0.9
0.8
0.7
0.6
0.5
0.4
0
1
2
3
4
5
6
7


0.83
0.82
0.81
0.8
0.79
0.78
0.77
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


70
60
50
40
30
20
10
0
0
1
2
3
4
5
6
7
8

8

Summary and Further Work

Work on moving meshes has evolved considerably over recent years, becoming a versatile tool to accurately simulate a wide range of problems. The key advantage of a moving mesh is its ability to adjust its distribution to focus on areas of interest, such as a moving boundary or blow-up. In this thesis we have discussed one such method, a finite difference

considered balancing the partial masses (8.1) with a source term, where appropriate.

We applied these methods to a number of moving boundary problems to investigate the effectiveness of this moving mesh approach. The problems we numerically solved to demonstrate how our moving mesh approach increased in complexity, initially looking at problems which conserve mass: the PME and Richards' equation, both of which are fluid flow problems. Then we looked at a problem with a variable mass: the Crank-Gupta problem, which is used to model oxygen-diffusion through tissue. Lastly, we considered an avascular tumour growth model, which is a system for which the mass increases over time. This has three PDEs (two quasi-steady) which need to be updated at each time-level. The quasi-steady PDEs were solved using finite difference on an irregular mesh, but this process did not compromise the moving mesh method. We summarise the application of each moving mesh approach in turn, and then discuss the time-stepping schemes used.

Preserving mass fractions

Preserving mass fractions was applied in all the problems. We examined the accuracy in all cases and found that the numerical solution converged with roughly second-order accuracy.

them to a constant. We saw that balancing the partial masses to the flux term of Richards' equation provided results that initially appear satisfactory, but when compared to the mass conserving approach we notice that the alternative method is less accurate. For the Crank-

Conclusions and further work

We conclude that the mass-conserving approach, with an expli

[10] Barari, A., Omidvar, M., Ghotbi, A.R. and Ganji, D.D. (20 09) Numerical analysis of Richards' problem for water penetration in unsaturated soils. Hydrol. Earth Syst. Sci. 6 6359-6385.

[11] Barenblatt, G.I. (1996) Scaling, self-similarity, and intermediate asymptotics. Cambridge University Press

[12] Barenblatt, G.I. (1952) On some unsteady motions of uids and gases in a porous medium. Prikladnaya Matematika i Mekhanika. (Translated in J. Appl. Math. Mech.) 6 67{78.

[13] Barlow, A. (2008) A compatible nite element multi-mate rial ALE hydrodynamics algorithm. Int. J. Numer. Meth. Fluids 56 953{964.

[38] Crank. J. and Gupta. R.S. (1972) A moving boundary problem arising from the di usion of oxygen in absorbing tissue. J. Inst. Maths. Applics. 10 19{33.

[39] Dahmardah, H.O. and Mayers, D.F. (1983) A Fourier-Series solution of the Crank-Gupta equation. IMA J. Numer. Anal. 3 81{85.

[40] DiMilla. P.A., Barbee, K. and Lau enburger, D.A. (1991) Mathematical model for the e ects of adhesion and mechanics on cell migration speed. Biophys J. 60 15{37.

[41] Doedel, E.J. (1981) AUTO, A program for the automatic bifurcation analysis of autonomous systems. Congr. Numer. 30 265{384.

[42] Dor , E.A. and Drury, L.O'C. (1987) Simple adaptive gri ds for 1-D initial value problems. J. Comput. Phys. 69 175{195.

[43] Demirdzic, I. and Peric, M. (1988) Space conservation law in nite volume calculations of uid ow. Int. J. Numer. Meth. Fl. 8 1037-1050.

[44] Demirdzic, I. and Peric, M. (1990) Finite volume method for prediction of uid ow in arbitrarily shaped domains with moving boundaries. Int. J. Numer. Meth. Fl. 10 771-790.

[45] Dvinsky, A.S. (1991) Adaptive grid generation from harmonic maps on Riemannian manifolds. J. Comput. Phys. 95 450-476.

[46] Gatenby, R.A. and Gawlinkski, E.T. (1996) A reaction-di usion model of cancer invasion. Cancer Res. 56 5745-5753.

[47] Gatenby, R.A. and Maini, P.K. (2003) Cancer summed up. Nature 421 321.

[48] Gelinas, R.J., Doss, S.K. and Miller, K. (1981) The moving nite element t t41 0 Td 2E7763(8(n7

[53] Harten, A. and Hyman, J.M. (1983) Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. J. Comput. Phys. 50 235–269.

[54] Heil, M. (2004) An efficient solver for the fully coupled solution of large displacement fluid-structure interaction problems. Comput. Methods Appl. M. 193 1-23.

[55] Huang, W., Ren, Y. and Russell, R.D. (1994) Moving mesh partial differential equations (MMPDEs) based on the equidistribution priciple. SIAM J. Sci. Comput. 31 709-730.

[56] Huang, W. and Russell, R.D. (1996) A moving collocation method for solving time dependent partial differential equations. Appl. Numer. Math 34 1106-1126.

[57] Huang, W. and Russell, R.D. (2011) Adaptive Moving Mesh Methods. Springer, New York.

[58] Knupp, P.M. (1995) Mesh generation using vector-fields. J. Comput. Phys. 119 142-148.

[59] Knupp, P.M. (1996) Jacobian-weighted elliptic grid generation. SIAM J. Sci. Comput. 17 1475-1490.

[60] Kuhl, E., Hulshoff, S. and Borst, R.de (2003) An arbitrary Lagrangian Eulerian finite-element approach for fluid-structure interaction phenomena. Int. J. Numer. Meth. Eng. 57 117-142.

[61] Kuraz, M. (2009) An Adaptive Time Discretization to the Numerical Solution of Richards' Equation. Czech University of Life Science http://klobouk.fsv.cvut.cz/ miguel/poster.pdf.

[62] Lacey, A.A., Ockendon, J.R. and Tayler, A.B. (1982) Waiting-time solutions of a nonlinear diffusion equation. SIAM J. of Appl. Maths. 42 1252–1264.

[63] Landman, K.A. and Please, C.P. (2001) Tumour dynamics and necrosis: Surface tension and stability. IMA J. Math. Appl. Medicine Biol. 18 131-158.

[64] Lee, H.K. (2001) For the behaviour of the solutions of a general porous media equations at large time scale. Information Center for Mathematical Sciences, Seoul University, 4 33–88.

[65] Lee, T.E., Baines, M.J., Langdon, S. and Tindall, M.J. (2011) A moving mesh approach for modelling avascular tumour growth. Math. Comput. Model Submitted.

[79] Piggott, M.D., Gorman, G.J., Pain, C.C., Allison, P.A. , Candy, A.S., Martin, B.T. and Wells, M.R. (2008) A new computational framework for multiscale ocean modelling based on adapting unstructured meshes. Int. J.Numer. Meth. Fl. 56 1003-1015.

[80] Please, C.P., Pettet, G.J. and McElwain, D.L.S. (1998) A new approach to modelling

[92] Tang, T. (2005) Moving mesh methods for computational fluid dynamics. Contemporary mathematics AMS 383 141–173.

[93] Tenchev, R.T., Mackenzie, J.A., Scanlon, T.J. and Stickland, M.T. (2005) Finite element moving mesh analysis of phase change problems with natural convection. Int J. Heat Fluid Fl. 26 597–612.

[94] Thomas, P.D. and Lombard, C.K. (1979) Geometric conservation law and its application to flow computations on moving grids. AIAA 17 1030–1037.

[95] Thompson, J.F, Warsi, Z.A. and Mastin, C.W. (1985) Numerical grid generation: foundations and applications. Elsevier North-Holland, Inc. New York, NY, USA.

[96] Tindall, M.J. and Please, C.P. (2007) Modelling the cell cycle and cell movement in multicellular tumour spheroids. Bull. Math. Biol. 69 1147–1165.

[97] Trulio, J.G. and Trigger, K.R. (1961) Numerical solution of the one-dimensional

[105] Winslow, A. (1967). Numerical solution of the quasi-linear Poisson equation in a nonuniform triangle mesh. J. Comput. Phys. 1 149-172.