UNIVERSITY OF READING
DEPARTMENT OF MATHEMATICS AND STATISTICS

A Moving-Mesh Method for High Order Nonlinear Diffusion

Nicholas Bird

Thesis submitted for the degree of
Doctor of Philosophy
August 2014

Abstract

In this thesis we study the use of a velocity-based moving-mesh numerical method, driven by conservation, for obtaining approximate solutions to nonlinear diffusion equations of second, fourth and sixth-order in one dimension. These problems often have moving boundaries and possess many properties that we are able to conserve using the method, such as scale-invariance and conservation of mass.

A key feature of the method is that it possesses the ability to propagate similarity solutions forward in time to within rounding error (termed the *S Property*). This property, occurring when a scale-invariant time stepping scheme is used and when the method is suitably tuned, produces highly accurate solutions.

Concentrating mainly on the fourth-order case, the method is implemented using finite differences and finite elements, with numerical results verifying the *S Property* for cases where explicit similarity solutions exist. For the finite difference implementation we describe the schemes used and show that the method has zero local truncation error over a single time step. In the finite element implementation we discuss the required weak forms, before highlighting the choice of approximation spaces needed for the method to possess the *S Property*.

Where no explicit similarity solutions exist we investigate the capability of the methods to reproduce known initial behaviour of the boundaries. We show that for the fourth-order PDE written as two second-order equations, singularities in the intermediate variable prevent the methods from behaving correctly. We discuss alternative approaches, one of which is broadly successful in reproducing initial behaviours.

Declaration

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

Nicholas Bird

Acknowledgements

Firstly, my sincerest thanks go to my supervisors for their excellent guidance and support throughout this project. Thank you Dr. Steve Langdon for treating every piece of writing as part of my thesis and honing my precision as a result. It definitely helps in the long run. Thanks also to Prof. Mike Baines for the near-unlimited ideas and enthusiasm for the work that I have done. Together you have kept me positive when things were not going as well as they could and encouraged me at all other times, which has made the project one I will never forget.

I would also like to acknowledge the funding of both the University of Reading and the Engineering and Physical Sciences Research Council (ESPRC) for enabling this journey to take place.

Particular thanks go to the Mathematics department at Reading for providing a great environment to work in over the past few years. All the various office-mates I have had made the process much more interesting, as have all the other PhD students and lecturers that I have been able to talk with (even if just over a cup of tea or two).

Thanks also to my friends and family for being there, asking me how my work was going even if you had no idea what I was talking about most of the time. I hope I've made you proud.

Finally, but most importantly, I need to thank my wife Suzanne for her love and support. Thank you for keeping me sane and putting up with me through the good times and the not-so-good times. We got to the other side of it together and without her I would not even be here. Gratitudes!

Contents

List of Notation

Notation related to analytical functions

Ω_T Domain upon which solutions are sought, both spatial and time dependent.

$\Omega(t)$ Spatial domain, typically a bounded subset of $\mathbb{R}$.

$\hat{\Omega}(t)$ Subregion of $\Omega(t)$ with boundary $\partial\hat{\Omega}(t)$.

$a(t)$ Left-hand moving boundary of $\Omega(t)$.

$b(t)$ Right-hand moving boundary of $\Omega(t)$.

$u(x,t)$ Solution of a nonlinear PDE.

$q(x,t)$ Auxiliary function, related to $u(x,t)$ depending upon order of PDE considered.

$p(x,t)$ Auxiliary function, related to $u(x,t)$ and $q(x,t)$ depending upon order of PDE considered.

$v(x,t)$ Deformation velocity of $\Omega(t)$.

v_b Boundary velocity, typically at $x = b(t)$.

$z(x,t)$ Velocity potential.

t Time variable.

x Spatial variable.

$\hat{x}(t)$ Arbitrary point in $\Omega(t)$.

s Scaled time variable, related to t.

m Constant determining the order of a PDE.

n Diffusion coefficient.

$\bar{f}$ Scaled variable, where f denotes a generic variable.

λ Scaling parameter.

β, γ, σ Scaling constants.

ξ, η, ζ Similarity Variables.

$f^S(x,t)$ Similarity function, where f denotes a generic function. In particular, $u^S(x,t)$ denotes the similarity solution to a nonlinear PDE.

κ, ω Constants in similarity solution.

ρ Mass of solution.

$c(\hat{x})$ Partial mass such that $c(\hat{x}) \subset \rho$.

α Constant used in initial conditions in chapter 6.

Notation related to numerical solutions

$F(x,t)$ Approximation to the analytical function $f(x,t)$.

t^k Discrete time point in the discretisation of time domain.

Δt Constant time step related to t.

s^k Discrete time point in the discretisation of scaled time domain.

Δs Constant time step related to s.

X_i^k Mesh point in spatial discretisation of $\Omega(t)$, at time t^k.

$\mathbf{X}^k$ Set of mesh points at time t^k.

N Number of nodes in mesh $\mathbf{X}^k$.

K Total number of time steps.

F_i^k Approximation to function $f(x,t)$ at node i and time point t^k.

$\mathbf{F}^k$ Set of points F_i^k.

ρ_i Partial mass of solution centred at node i of mesh.

$\boldsymbol{\rho}$ Set of partial masses.

$\widehat{V}^k$ Approximation to $v(x,s)$ at s^k in terms of scaled time variable s.

$w(x,t)$ Test function in weak formulation or finite element method.

τ_i^k Truncation error of finite difference scheme at node i, time t_k (also $\tilde{\tau}_i^k$, $\hat{\tau}_i^k$).

$\psi(x,t)$ Sextic Lagrange polynomial basis function.

$\chi(x,t)$ Quartic Lagrange polynomial basis function.

$\varphi(x,t)$ Quadratic Lagrange polynomial basis function.

$\phi(x,t)$ Linear Lagrange polynomial basis function.

E_N^F l^∞ error of $\mathbf{F}^k$ on a mesh of N nodes.

E_N^B Relative error in boundary node position on a mesh of N nodes (typically right-hand boundary $x = b(t)$).

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

There are a number of physical processes which correspond to nonlinear diffusion problems when modelled mathematically (see, e.g., Barrett et al. (2004)). These problems can be of various orders, but those of particular interest in this thesis are those of high order (fourth and sixth-order), although we also consider the second-order Porous Medium Equation (PME). Mathematically, problems of this type consist of a partial differential equation (PDE), together with an initial condition and suitable boundary conditions.

The high-order nonlinear diffusion problems considered in this thesis have a number of interesting features, the most important of which is that the boundaries of the solution can evolve over time. These moving boundaries add an extra complexity to the problems, since (from a numerical viewpoint) some mechanism is required which can track the position of the boundary.

There exists a large body of literature describing the behaviour of solutions to high-order nonlinear diffusion problems. In particular, the degree of nonlinearity of the problem can have an important effect on the behaviour of the solution close to the moving boundary (see, e.g., Beretta et al. (1995), Bowen and King (2001)). The preservation of nonnegative solutions has also been proven for a range of problems relevant to the work

in this thesis (see Bernis and Friedman (1990) among others).

Another important feature is that there exist some similarity solutions to the problems, which may act as long-time limits of the problem regardless of the initial condition (Smyth and Hill (1988)). The existence of explicit similarity solutions to high-order nonlinear diffusion problems is limited to a small number of cases, which motivates the development of numerical methods for obtaining approximations to the solutions of the problems, but explicit similarity solutions can provide a comparison in cases where they exist.

The use of numerical methods for solving nonlinear diffusion problems can be split broadly into fixed-mesh and moving-mesh methods, with the discretisation of the domain being considered differently in each case. In a fixed-mesh method the discretisation of the domain into a number of fixed points is carried out and the problem solved on this fixed domain, with mesh points possibly being added or removed as required to assist resolution of features of the solution (see, e.g., Budd et al. (2009)). A moving-mesh method differs in that the points of the mesh are allowed to vary in time, which necessitates the mesh and the solution being solved for together (Budd et al. (2009)). There exist mapping-based and velocity-based moving-mesh methods in which the approximation of the mapping or the mesh velocity is of as much importance as the method of approximating the solution of the problem.

A number of moving-mesh methods exist for obtaining approximate solutions to nonlinear diffusion problems, with the finite element method of Baines, Hubbard and Jimack (BHJ) of particular influence in this thesis (see Baines et al. (2005, 2006, 2011)). This method is driven by local conservation and in its simplest form uses the mass of the solution to obtain the mesh velocity, before recovering the solution algebraically on the updated mesh. The BHJ method is a moving-mesh finite element method which has been shown numerically to be approximately second order in space for second-order and fourth-order problems for which similarity solutions exists (Baines et al. (2005)).

A further relevant piece of work is the MSc thesis of (Parker (2010)), who showed for a second-order problem with a specific power law that mesh movement and solution

recovery can be achieved essentially to within rounding error using a finite difference scheme, in the case where the initial condition coincides with a similarity solution and when a scale invariant time stepping scheme is used.

Both the BHJ method and that of Parker are velocity-based moving-mesh schemes built upon the same conservation strategy. We shall refer to these as conservation-based schemes (or simply the conservation method). This thesis generalises the conservation method and provides an implementation in both finite differences and finite elements which is able to propagate similarity solutions forward in time to essentially within rounding error. A significant feature of the implementations in this thesis is the use of invariant time stepping schemes which preserve scale invariance.

Also of interest is whether such a method is able to accurately model the small-time behaviours of solutions which do not initially correspond to a similarity solution. Particular attention is given to (and comparison made with) the work in Blowey et al. (2007) in this regard, as it produces a comparison between known small-time behaviours (King (2001)) using a fixed-mesh method.

1.2 Aims and Key Results

The basic aim of this thesis is to explore the use of moving-mesh numerical methods based on conservation in solving high-order nonlinear diffusion problems. The main focus is on fourth-order problems, although attention is also given to second and sixth-order problems. Specifically, we aim to:

- Extend the finite difference work of Parker (2010) for a particular second-order problem to higher orders (fourth and sixth) as well as to more general second-order problems using a finite difference implementation of the conservation method. The implementations, which use a scale-invariant time stepping scheme, are theoretically able to propagate similarity solutions forward in time to essentially within rounding error of the exact solution over a single time step. We term this property of the numerical method the *S Property*.

- Produce an implementation of the BHJ method using higher order basis functions and scale-invariant time stepping in order to propagate similarity solutions forward in time to within rounding error. This finite element implementation should be able to approximate solutions to second, fourth and sixth-order problems.

- Use these implementations to provide a comparison with the work in Blowey et al. (2007) in order to verify whether the moving-mesh method is able to replicate the small-time behaviours of solutions (computed in that paper using a fixed-mesh method). It is hoped that the method can not only model such behaviour, but do so at a reduced computational expense.

The key results of this thesis are as follows:

- The results of Parker (2010) are successfully extended to a more general class of second-order nonlinear diffusion problems using a conservation-based finite difference method. The method (termed the Moving Point Conservation Method, or MPCM) can be shown to be possess the *S Property*, with approximate solutions matching the exact solutions essentially to within rounding error.

- The MPCM is extended to approximate fourth-order (written as two second-order equations) and sixth-order (written as three second-order equations) nonlinear diffusion problems. By careful construction of the numerical schemes used in the method, the MPCM is shown to possess the *S Property* for the fourth and sixth-order problems.

- Numerical results verifying the possession of the *S Property* reveal a slow build up of global error in the method (shown to be bounded) when run over multiple time steps.

- The BHJ method is implemented using piecewise linear finite elements, with some minor differences with regards to the solution recovery step.

- By selection of appropriate basis functions, a finite element implementation of the MPCM (termed the FEMPCM) for second, fourth and sixth-order problems is

also shown to possess the *S Property*, with numerical results verifying this for the fourth-order problem.

- Study of the fourth-order problem in cases where an explicit similarity solution does not exist highlights the issues which the MPCM can face related to the boundary velocity of the domain (when the fourth-order PDE is written as a pair of second-order equations). The presence of singularities at the moving boundary causes the method to be unable to model a large range of cases where the boundary experiences a finite or zero velocity due to certain functions becoming unbounded at the boundary. The method is also unable to model cases where the boundary initially moves with an instantaneously unbounded velocity.

- An investigation into whether the FEMPCM is able to alleviate some of the issues experienced by the finite difference implementation (for the fourth-order PDE when written as two second-order equations). It is shown that the boundary velocity issues are mollified in the finite element case, but not eliminated completely.

- An alternative expression for the velocity of the domain is proposed (in which the fourth-order PDE is not written as a pair of coupled equations) which alleviates the issues experienced in the instances where the boundary experiences a finite or zero velocity. This is achieved by rewriting the velocity in terms which are smooth and finite at the boundary. Numerical results show that this alternative expression produces approximations to the velocity which are accurate, albeit with some oscillations in the velocity still appearing.

1.3 Thesis Outline

In Chapter 2 we introduce the nonlinear diffusion PDEs which will be studied in this thesis. Coupled with appropriate boundary conditions and an initial condition, these PDEs make up the fourth, second and sixth-order problems, of which the fourth-order problem (written as a pair of coupled equations) will form the main focus of the thesis.

A discussion of the existing literature surrounding these problems is presented, detailing known properties and behaviours of the nonlinear PDEs. The properties which are of the most relevance for this thesis are then expanded upon, including mass conservation and similarity solutions.

The chapter ends with a description of existing numerical methods for solving the problems outlined at the beginning of the chapter, drawing upon the large body of literature which exists around this topic. As all work in this thesis in concerned with one spatial dimension we focus on this, but a short description of methods in higher dimensions is provided.

Chapter 3 explores a particular type of moving-mesh numerical method, which forms the basis of the numerical method developed for this thesis. A velocity-based moving-mesh method based on conservation seeks to obtain the velocity of the discretised domain along with the approximate solution to the PDE. This mesh velocity is then used to update the domain over time. Of the different velocity-based methods which exist the conservation method is emphasised in the thesis.

The numerical method used for the work in the thesis, the MPCM, is then described, with finite difference implementations for the various nonlinear diffusion problems outlined. An important property which the MPCM may possess is then defined, the *S Property*, with scale-invariant time stepping an essential ingredient in the MPCM for this property to be present. If an implementation of the MPCM possesses the *S Property*, then we are able to obtain extremely accurate solution approximations under certain conditions.

An implementation of the MPCM is described in Chapter 4 for the fourth-order problem. This implementation is shown to possess the *S Property* upon some modification to the basic MPCM. Details on how the MPCM implementation can be modified to enable the method to possess the *S Property* for second and sixth-order problems are also given.

Chapter 5 sees the introduction of the FEMPCM (building upon the BHJ method), starting with the necessary weak forms. It is shown that by carefully selecting the finite-

dimensional subspaces and the basis functions which lie in those spaces, the FEMPCM can possess the *S Property*.

The cases where there is no explicit similarity solution available (so that the MPCM implementations do not possess the *S Property*) are explored in Chapter 6. In particular, we show that there are multiple possible initial behaviours of the boundary of the domain. We demonstrate these different behaviours using both the MPCM and FEMPCM and highlight issues stemming from singularities at the boundary which can cause the MPCM and FEMPCM to fail. We also investigate convergence of the method through tracking of the boundary position of the domain and solution comparisons.

Two attempts to alleviate the issues with boundary singularities for the MPCM are proposed. The first of these is a hybrid numerical method making use of both fixed-mesh and moving-mesh methods which allows for modelling of an initially unbounded velocity. The hybrid numerical method is explored in a descriptive sense only without validation.

The second proposal is to introduce an alternative expression for the velocity, bypassing writing the fourth-order problem as a pair of coupled equations. Promising numerical results are given to validate the alternative velocity method.

Finally, in Chapter 7 we summarise the main conclusions of the thesis and outline possible areas of future work.

Chapter 2

Nonlinear Diffusion

2.1 Aims of this Chapter

In this chapter we introduce the various nonlinear diffusion equations which shall be studied over the course of this thesis, with specific examples of second, fourth and sixth-order equations considered. We also describe the boundary conditions used in the work, as well as a discussion on alternative boundary conditions that have been used in the literature.

We shall also outline various properties which these equations possess such as scale invariance and conservation of mass. The preservation of these properties is a key aim in the numerical methods to be discussed in this and later chapters.

We then provide a discussion on numerical methods for finding approximate solutions to nonlinear diffusion problems and give examples of their use in the literature.

2.2 Nonlinear Diffusion Equations

2.2.1 General Form

The equations considered in this thesis are one-dimensional (1D) nonlinear diffusion PDEs which can be written in the general form: Find $u = u(x,t)$ such that

$$\frac{\partial u}{\partial t} = (-1)^m \frac{\partial}{\partial x}\left(u^n \frac{\partial^{2m+1} u}{\partial x^{2m+1}}\right), \qquad \text{in } \Omega_T := \Omega(t) \times (t^0, T), \tag{2.1}$$

where $\Omega(t) \equiv (a(t), b(t)) \subset \mathbb{R}$ is a moving bounded domain, n is a positive constant and the integer $m = 0, 1, 2, \ldots$ determines the order of the equation. We supplement (2.1) with an initial condition $u(x, t^0) = u^0(x)$ and suitable boundary conditions (see §2.2.2–§2.2.4 for a description of the boundary conditions to be used in the work in this thesis).

We can rewrite the general form (2.1) as

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), && \text{in } \Omega_T, \\ q &= (-1)^m \frac{\partial^{2m} u}{\partial x^{2m}}, && \text{in } \Omega_T, \end{aligned} \tag{2.2}$$

where $q(u)$ is dependent on the order of the problem under consideration. Equation (2.2) is the generalised Reynolds equation (Flitton and King (2004)) and the problems considered in this thesis will be derived from (2.1) or (2.2) for second, fourth and sixth-order problems with different $q(u)$ for each case.

The majority of the work covered in this thesis pertains to the fourth-order problem, so this will form the main focus of this chapter.

2.2.2 A Fourth-Order PDE

A fourth-order PDE is obtained by taking $m = 1$ in (2.1) or by setting $q = -\frac{\partial^2 u}{\partial x^2}$ in (2.2)

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), \qquad \text{in } \Omega_T, \tag{2.3a}$$

$$q = -\frac{\partial^2 u}{\partial x^2}, \qquad \text{in } \Omega_T. \tag{2.3b}$$

In order for this problem to be well-posed, it is necessary to have the correct number of boundary conditions at each of the moving boundaries. For a fixed boundary problem only two boundary conditions would be required at each boundary, but the presence of moving boundaries necessitates application of an additional condition at each boundary.

We shall seek solutions to (2.3), subject to an initial condition at time $t = t^0$ given by

$$u(x, t^0) = u^0(x), \qquad \text{for } x \in \Omega(t^0), \tag{2.4}$$

and the boundary conditions

$$u = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.5a}$$

$$\frac{\partial u}{\partial x} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.5b}$$

$$uv + u^n \frac{\partial q}{\partial x} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.5c}$$

where $v(x, t)$ is the velocity $\frac{dx}{dt}$ of the boundary.

We have chosen a zero contact angle condition (2.5b) for this work since strong solutions (i.e., those satisfying the definition of a weak solution given in Bernis and Friedman (1990) as well as the entropy estimates detailed in Beretta et al. (1995), Bertozzi (1996)) of the fourth-order problem for $n \in (0, 3)$ satisfy this condition (see, e.g., Bernis et al. (2000), Bowen et al. (2001)). The $u = 0$ boundary condition (2.5a) defines the position of the moving boundary, while the third boundary condition (2.5c) corresponds to zero net flux across the boundary. An important point to note is that the problem is degenerate at points where $u = 0$ (as explained in, e.g., Barrett et al. (1998)). This degeneracy signifies that the boundary points in particular need to be treated with care.

From this point in the thesis we shall refer to (2.3)–(2.5) as the **fourth-order problem**, where (2.3) is the fourth-order PDE.

Applications and Existing Results in Literature

The fourth-order PDE (2.3) arises in several models of physical processes. For example, in fluid dynamics it can be derived from a lubrication approximation in order to model

the motion of thin films and droplets spreading over a surface (see Greenspan (1978), Bertozzi and Pugh (1996), Bernis and Friedman (1990) and many others). For this reason the fourth-order PDE is often referred to as the thin film equation.

When $n = 1$ the PDE models the flow of fluid between two plates in a Hele-Shaw cell (Beretta et al. (1995)), while for $n = 3$ a droplet with no-slip conditions is modelled (see, e.g., Greenspan (1978)). In this context, the function q may represent pressure in the droplet (Bertozzi and Pugh (1996)).

The value of $n = 3$ corresponds to a no-slip condition at the moving boundary, which implies an infinite force at the contact line (Dussan and Davis (1974)). This value of n appears to be of particular importance, since similarity solutions with finite mass exist only in the case when $n \in (0, 3)$ (Bernis et al. (1992)). This has led to a focus in research on this range of n, particularly in relation to numerical solutions to the fourth-order PDE (see, e.g., Blowey et al. (2007)).

There has been a large amount of discussion of solutions to the fourth-order PDE. For values of $n \geq 1$ it is known that weak solutions to the fourth-order problem (2.3)–(2.5) exist and that such solutions remain nonnegative for all time provided that the initial condition is nonnegative (Bernis and Friedman (1990)). Bernis et al. (1992) prove the existence of source-type similarity solutions for different values of n, as well as discussion of the local behaviour of solutions near the moving boundary.

The properties discussed above are valid for the problems considered in this thesis, i.e. problems with zero net flux conditions at the moving boundary as in (2.5c). There is also a large amount of literature focussing on problems which do not have this boundary condition. The case of the fourth-order problem on a fixed domain, in which the fluid being modelled is allowed to flow from the edge of the domain is discussed in Bowen and King (2001) and Bouwe Van Den Berg et al. (2004).

In Bertozzi and Pugh (1996), a weak nonnegative solution, which becomes strictly positive after some finite time, is proved to exist for all time for two fourth-order problems with a variety of different boundary conditions. Bowen et al. (2001) look at the fourth-order problem on the half-line, with zero pressure (i.e. $\frac{\partial^2 u}{\partial x^2} = 0$) at the moving

boundary. This work is concentrated on dipole solutions to the problem, where the point at $x = 0$ is not a moving boundary point. This half-line problem is also considered in Bernis et al. (2000).

It is also possible to consider problems which do not possess a zero contact angle boundary condition. Otto (1998) examines the existence of weak solutions for large-time in the $n = 1$ case. In this case a fixed contact angle of $\frac{\pi}{4}$ is used.

The issue of 'dry spots' (alt. 'dead cores'), in which the solution develops a region of zero-values inside the moving boundaries has also been studied. King and Taranets (2013) provide existence and uniqueness proofs for travelling wave solutions of this nature, as well as upper and lower bounds on such solutions.

In Giacomelli et al. (2008) the fourth-order problem is transformed from a moving to a fixed domain for the $n = 1$ case. This paper notes the limiting expression for the boundary velocity which is important for the work in this thesis (see equation (6.3)). Existence, uniqueness, smoothness, decay of high derivatives and convergence to a steady-state are all proved for the transformed problem.

There has also been a lot of work on generalisations of the thin-film equation with extra terms in the governing equation. In King (2001) a study of two generalisations of the thin-film equation is given, with a focus on nonnegative mass-conserving solutions. An asymptotic analysis of the solution close to the boundary is provided. Jiang and Wei-Ming (2007) also study a generalisation of the thin-film equation in the case of a van der Waals force driven thin film.

The unstable fourth-order equation

$$\frac{\partial u}{\partial t} = -\nabla \cdot (|u|^n \nabla \Delta u) - \Delta(|u|^{p-1}u),$$

where $n > 0$, $p > 1$ and $\Delta(|u|^{p-1}u)$ is an unstable second-order term, is considered in Evans et al. (2007a), with regard to the existence of similarity solutions.

Moving Boundaries for the Fourth-Order Problem

When considering the fourth-order problem (2.3)–(2.5), the value of n is very important for determining the local behaviour of the boundary. In Beretta et al. (1995), it is

shown that the value of n will determine the nature of the support of the solution. In particular, if $n \geq 3/2$ then the support of u will not shrink over time, while for values of $n \geq 4$ the support is constant.

Similar results are shown in King and Bowen (2001), where the initial condition is of the form

$$u^0(x) = I(x) + \epsilon,$$

where $I(x) = 0$ for $|x| \geq a$, $I(x) > 0$ for $|x| < a$ for initial interface position $x = \pm a$ and $\epsilon > 0$. The behaviour in the limit $\epsilon \to 0^+$ is considered and it is shown that for $n < 3/2$ the interface velocity may be of either sign, while for $n > 3/2$ the velocity is strictly nonnegative. This therefore identifies $n = 3/2$ as a critical case. The support of the solution is shown in this paper to be constant for values of $n \geq 3$, which identifies the region $n \in (0, 3)$ as being of importance for the moving boundary problem.

Bernis (1996b) demonstrates a finite speed of propagation property for nonnegative strong solutions which exist for two fourth-order problems when $0 < n < 2$. The first problem considered uses homogeneous boundary conditions $\frac{\partial u}{\partial x} = \frac{\partial^3 u}{\partial x^3} = 0$ at each boundary, while the second problem uses periodic boundary conditions. The results are then extended to the case $2 \leq n < 3$ in Bernis (1996a). This finite speed of propagation property ensures that we may expect a bounded velocity for the moving boundaries for all $t > t^0$. The finite speed of propagation property is also shown in Hulshof and Shishkov (1998) for a fourth-order problem with slightly different boundary conditions. The main result of this paper focusses on the case when $2 \leq n < 3$, but some consideration is given to the $0 < n < 2$ case.

The local behaviour of solutions close to the right-hand boundary $x = b(t)$ is considered in King and Bowen (2001), and for $n \in (0, 3)$ is shown to be of the form

$$\begin{aligned}
u &\sim \left(\frac{n^3 \dot{b}}{3(3-n)(2n-3)} (b(t) - x)^3 \right)^{\frac{1}{n}}, \quad \text{for } \frac{3}{2} < n < 3, \\
u &\sim \left(\frac{3}{4} \dot{b} (b(t) - x)^3 \ln \left[\frac{1}{(b(t) - x)} \right] \right)^{\frac{2}{3}}, \qquad \text{for } n = \frac{3}{2}, \\
u &\sim B(t)(b(t) - x)^2, \qquad \text{for } n < \frac{3}{2},
\end{aligned} \tag{2.6}$$

as $x \to b(t)^-$, where $\dot{b} = \frac{\mathrm{d}b}{\mathrm{d}t}$ and where $B(t)$ needs to be determined as a part of the solution. For the case when $3/2 < n < 3$, the velocity of the boundary is strictly positive, while for $n < 3/2$ the velocity may take either sign.

Blowey et al. (2007) consider the results of King and Bowen (2001) and apply them to initial conditions of the form

$$u^0(x) \sim A_0(b_0 - x)^\alpha + C_0(b_0 - x)^\beta,$$

where α, β are constants such that $\beta > \alpha$, and explore the small-time behaviour of the right-hand boundary. They demonstrate that, for various choices of n and α, the boundary behaves as detailed in King and Bowen (2001) and provide numerical results to support the asymptotic analysis. In particular, they highlight a region in (n, α) space in which a variety of waiting-time scenarios are exhibited.

2.2.3 A Second-Order PDE

We now describe the second-order problem considered in this thesis. The second-order problem will not be covered in as much detail, but as the first in the hierarchy of the generalised Reynolds equations (Flitton and King (2004)) it is worth detailing here.

A second-order PDE, obtained by setting $m = 0$ in (2.1) or by taking $q = u$ in (2.2) is

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial u}{\partial x}\right), \qquad \text{in } \Omega_T. \tag{2.7}$$

We shall seek solutions to (2.7) subject to the initial condition

$$u(x, t^0) = u^0(x), \qquad \text{for } x \in \Omega(t^0), \tag{2.8}$$

and subject to suitable boundary conditions imposed at $x = a(t)$ and $x = b(t)$. For the second-order moving boundary problem we require two boundary conditions at each boundary in order for the problem to be well-posed. For the work in this thesis we shall consider boundary conditions

$$u = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.9a}$$

$$uv + u^n \frac{\partial u}{\partial x} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.9b}$$

The first condition is specifies the position of the moving boundary, while the second condition denotes a zero net flux across the boundary and ensures conservation of the total mass (total integral) of the solution (see §2.4.1). These conditions match (2.5a) and (2.5c) used in the fourth-order problem (2.3)–(2.5), but do not include the zero contact angle boundary condition (2.5b).

From this point, we shall refer to (2.7)–(2.9) as the **second-order problem** (with (2.7) referred to as the second-order PDE).

Applications and Known Results from Literature

The second-order equation is well-known as the porous medium equation (see, e.g., Vasquez (2007)). It can be used to model various physical processes. For $n \geq 1$ the equation models the flow of a gas through a porous medium (Muskat (1937)), while for $n = 3$ the equation can be used to model the flow of a thin liquid film which spreads under the effects of gravity (Buckmaster (1977)). The case $n = 1$ can correspond to unconfined groundwater flow (Peletier (1981)), with the physical meaning of $u(x, t)$ dependent upon the process being modelled.

The porous medium equation is widely discussed in the literature (see Aronson (1986), Peletier (1981), Vasquez (2007) among many others) and many results about the behaviour of solutions to this equation are known. Smyth and Hill (1988) provide a detailed summary of the various behaviours of the solutions to (2.7).

In particular, it is known that waiting-time solutions exist, where the initial support of the solution remains fixed for a time, for this equation supplemented by the boundary condition (2.9a). The solution may change in the interior of the support before the boundary begins to move. Knerr (1977) provides similarity solutions possessing such behaviour, which cease to be valid once the boundary begins to move. Solutions valid after boundary movement has begun are given in Lacey et al. (1982), which are not however expressible in closed-form. Kath and Cohen (1982) construct approximate solutions for small n ($0 < n \ll 1$) which exhibit waiting time behaviour.

Bounds on the waiting time are given in Perazzo and Gratton (2004). In Lacey

(1983), a discussion on waiting time behaviour for a second-order equation is given, with a focus on the initial movement of the boundary once the waiting time is over.

We note that the second-order problem (2.7)–(2.9) has one important property missing from the fourth-order problem (2.3)–(2.5), in that a maximum principle exists for the second-order problem but not for higher-order problems (as explained in Barrett et al. (1998)). This maximum principle can be used to obtain many of the important results for the second-order problem (2.7)–(2.9), particularly uniqueness of solutions.

2.2.4 A Sixth-Order PDE

We now describe the sixth-order problem which will also be considered in this thesis, which is the third in the hierarchy of the generalised Reynolds equations. It shares many properties with the fourth-order problem (2.3)–(2.5), and as such we shall consider this problem an extension of the fourth-order work. In particular, we note the lack of a maximum principle for this problem.

The sixth-order PDE can be obtained by setting $m = 2$ in (2.1) or by setting $q = -\frac{\partial^2 p}{\partial x^2}$, $p = -\frac{\partial^2 u}{\partial x^2}$, in (2.2) and writing it as

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), \qquad \text{in } \Omega_T, \tag{2.10a}$$

$$q = -\frac{\partial^2 p}{\partial x^2}, \qquad \text{in } \Omega_T, \tag{2.10b}$$

$$p = -\frac{\partial^2 u}{\partial x^2}, \qquad \text{in } \Omega_T. \tag{2.10c}$$

We seek solutions to (2.10) subject to the initial condition

$$u(x, t^0) = u^0(x), \qquad \text{for } x \in \Omega(t^0), \tag{2.11}$$

and subject to boundary conditions

$$u = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.12a}$$

$$\frac{\partial u}{\partial x} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.12b}$$

$$\frac{\partial^3 u}{\partial x^3} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \tag{2.12c}$$

$$uv + u^n \frac{\partial q}{\partial x} = 0, \qquad \text{at } x = a(t), \quad x = b(t), \quad t > t^0. \tag{2.12d}$$

The boundary conditions (2.12a), (2.12b) and (2.12d) match those used in the fourth-order problem (2.3)–(2.5), with boundary condition (2.12c) new, in comparison to the fourth-order problem.

From this point on, we shall refer to (2.10)–(2.12) as the **sixth-order problem** (with (2.10) the sixth-order PDE).

Applications and Results from Literature

The sixth-order PDE with $n = 3$ arises in the modelling of silicon oxidisation in superconductor devices (see King (1986), King (1989a), King (1989b)). The $n = 3$ case can also be used to model the motion of a droplet under an elastic plate driving force (see, e.g., Barrett et al. (2004)).

While the fourth-order problem (2.3)–(2.5) has been widely discussed in the literature (as has the second-order problem (2.7)–(2.9)), there appears to have been much less work done on sixth-order problems. There are however still some important results than can be gleaned for such problems.

Bernis and Friedman (1990) proved the existence of Hölder continuous nonnegative solutions to (2.10) in the 1D case. In this case however, the boundary condition (2.12d) is replaced by $\frac{\partial q}{\partial x} = 0$. Barrett et al. (2004) note that the sixth-order problem (2.10)–(2.12) may not have a unique solution, indeed that there are some choices of n for which at least two solutions to the problem exist. The authors demonstrate this by computing spreading and non-spreading solutions for various choices of n.

Huang et al. (2013) perform a Lie point symmetry group classification of the sixth-order PDE (2.10a) and show that for $n < -6$ the equation admits sink solutions while for $n > -6$ source solutions are admitted. The $n = -6$ case admits both source and sink solutions. The paper also shows travelling-wave solutions for $n = 1$ as well as waiting-time solutions for general n.

There has also been work on sixth-order problems different to that being considered in this thesis. In Evans et al. (2007b) and Evans et al. (2007c) an unstable sixth-order problem is considered in which an unstable second-order term has been added to the

equation given in (2.10a). The existence of blow-up similarity solutions occurring in finite time is covered in the first of these papers, while global similarity solutions are explored in the second.

In Smith et al. (1996) the flow under a nitride cap is modelled using the sixth-order equation

$$u_t = \frac{C}{12}\frac{\partial}{\partial x}\left(u^3\frac{\partial^5 u}{\partial x^5}\right),$$

where C is a constant denoting the dimensionless flexural rigidity. This problem is solved subject to the conditions that $u > 1$ inside the moving boundary and $u = 1$ elsewhere, with the moving boundary not known *a-priori* and therefore found as part of the solution.

Moving Boundaries in the Sixth-Order Problem

It has also been noted that for the sixth-order problem (2.10)–(2.12) a variety of behaviours of the moving boundary can be observed, dependent upon the value of n.

In Flitton and King (2004), a number of initial boundary value problems relating to the sixth-order problem (2.10)–(2.12) are discussed. They note that the support of the initial condition changes only for $n < 5/2$, with the boundary velocity able to take either sign for $n < 5/3$. For values of $n \geq 5/2$ the boundary moves with a zero speed for all problems considered in the paper.

2.3 PDEs in a Moving Framework

The PDEs described in §2.2 are written in a fixed frame of reference with v appearing only in the boundary conditions. It is possible to rewrite the problems in a framework moving with a general velocity $v(x,t)$ by considering some arbitrary sub-region $\hat{\Omega}(t)$ of our domain $\Omega(t)$, which has a boundary $\partial\hat{\Omega}(t)$ moving with this velocity. We shall illustrate this framework for the alternate form (2.2) of the nonlinear PDE (2.1), but the process applies to any of the problems considered in §2.2.

Differentiating the integral

$$\int_{\hat{\Omega}(t)} u(x,t)\,\mathrm{d}x,$$

with respect to t using Leibniz's Integral Rule with variable limits, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\hat{\Omega}(t)} u(x,t)\,\mathrm{d}x = \int_{\hat{\Omega}(t)} \frac{\partial u}{\partial t}\,\mathrm{d}x + \Big[uv\Big]_{\partial\hat{\Omega}(t)}. \tag{2.13}$$

Then, using the divergence theorem, along with an integral form of the PDE (2.2), we may rewrite (2.13) as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{\hat{\Omega}(t)} u(x,t)\,\mathrm{d}x &= \int_{\hat{\Omega}(t)} \frac{\partial}{\partial x}\left(u^n\frac{\partial q}{\partial x}\right)\,\mathrm{d}x + \Big[uv\Big]_{\partial\hat{\Omega}(t)},\\ &= \Big[uv + u^n\frac{\partial q}{\partial x}\Big]_{\partial\hat{\Omega}(t)},\end{aligned} \tag{2.14}$$

which is now written in a framework moving with velocity $v(x,t)$. Appropriate boundary conditions, (when $\hat{\Omega}(t) \equiv \Omega(t)$) are the same as those discussed in §2.2 for the relevant problems (where v is the boundary velocity). Equation (2.14) will be of use in obtaining an expression for the velocity in mass conserving problems (see §2.4.1).

2.4 Some Properties of Nonlinear PDEs

The problems outlined in §2.2 possess a number of properties which we seek to maintain in the numerical methods developed here for obtaining approximate solutions. In this section we shall outline the properties which we wish to preserve in our numerical methods.

2.4.1 Mass Conservation

At any given time we can calculate the area (alternatively referred to as the mass) underneath the curve representing the solution $u(x,t)$ of the equation (2.1). We denote the total mass of the solution by $\rho(t)$. The mass is defined by the integral of the solution over the domain,

$$\rho(t) := \int_{a(t)}^{b(t)} u(x,t)\,\mathrm{d}x. \tag{2.15}$$

Differentiation of (2.15) with respect to time shows how the area under the support of $u(x,t)$ changes through time. We use Leibniz's Integral Rule with variable limits again, resulting in

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} u(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial u}{\partial t}\,\mathrm{d}x + \frac{\mathrm{d}b}{\mathrm{d}t}u(b(t),t) - \frac{\mathrm{d}a}{\mathrm{d}t}u(a(t),t). \tag{2.16}$$

Substituting the expression for $\frac{\partial u}{\partial t}$ from equation (2.2) into (2.16) and writing

$$\frac{\mathrm{d}a}{\mathrm{d}t} = v(a(t),t), \qquad \frac{\mathrm{d}b}{\mathrm{d}t} = v(b(t),t),$$

gives

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} u(x,t)\,\mathrm{d}x &= \int_{a(t)}^{b(t)} \frac{\partial}{\partial x}\left(u^n\frac{\partial q}{\partial x}\right)\mathrm{d}x + \left[uv\right]_{a(t)}^{b(t)},\\ &= \left[u^n\frac{\partial q}{\partial x} + uv\right]_{a(t)}^{b(t)}, &(2.17a)\\ &= 0, &(2.17b)\end{aligned}$$

where the step from (2.17a) to (2.17b) holds due to the boundary condition $uv + u^n\frac{\partial q}{\partial x} = 0$ at the boundaries $a(t), b(t)$.

Integrating (2.17b) with respect to t we see that $\rho(t)$ as given by (2.15) is constant with respect to t (and hence we may write $\rho(t) = \rho$ for mass-conserving problems).

There is a link between the steps shown here to demonstrate mass conservation and the steps required to rewrite the problems of §2.2 in a moving framework. Comparing (2.14) to (2.17a) in the case where $\hat{\Omega}(t) = \Omega(t)$ demonstrates this link for the full domain.

2.4.2 Scale Invariance

An important property of PDEs of the form (2.1) is that they are scale invariant. A PDE is said to be scale invariant if there exists a group transformation of the underlying variables which satisfy the same equations (see, e.g., Budd and Piggott (2001)). In the case of (2.1) we can define a scaling transformation

$$\bar{t} = \lambda t, \qquad \bar{x} = \lambda^\beta x, \qquad \bar{u} = \lambda^\gamma u, \tag{2.18}$$

where λ is an arbitrary positive quantity, which leaves the underlying PDE (2.1) unchanged provided that

$$\gamma - 1 = (n+1)\gamma - 2(m+1)\beta. \tag{2.19}$$

An additional condition (such as that arising from the mass conservation property, (2.15)) allows the indices γ and β to be uniquely defined. Applying (2.18) to (2.15), where ρ is constant in time, we see that for scale invariance

$$\gamma + \beta = 0. \tag{2.20}$$

The unique indices in (2.18) to ensure scale invariance of the PDE (2.1) are therefore

$$\beta = \frac{1}{n+2m+2}, \qquad \gamma = -\frac{1}{n+2m+2}. \tag{2.21}$$

The alternate form (2.2) of the PDE can also be shown to be scale invariant by introducing the scaling transformation

$$\bar{q} = \lambda^{\sigma} q,$$

in addition to the scalings (2.18). Under these scalings, (2.2) remains unchanged provided that

$$\gamma - 1 = n\gamma + \sigma - 2\beta,$$

where the condition defining σ uniquely will depend on the order of problem being considered:

- For the second-order PDE (2.7) where $q = u$, we have that $\sigma = \gamma$.
- For the fourth-order PDE (2.3a) we have that $q = -\frac{\partial^2 u}{\partial x^2}$ and $\sigma = \gamma - 2\beta$.
- For the sixth-order PDE (2.10a) we have $q = -\frac{\partial^2 p}{\partial x^2}$, $p = -\frac{\partial^2 u}{\partial x^2}$, giving $\sigma = \gamma - 4\beta$.

Scale invariance of the PDE does not ensure scale invariance of the solutions, although there may exist a class of scale invariant similarity solutions which we shall now discuss.

2.4.3 Similarity Solutions

For general initial conditions we are unable to find an analytical solution to the problems discussed in §2.2, but there may exist closed-form similarity solutions for certain choices of n and m in (2.1) which we are able to write down explicitly.

Smyth and Hill (1988) note that an explicit closed-form source-type similarity solution exists for equations of the form (2.1) in the case when $n = 1$ for any m, and for any choice of n when $m = 0$. We shall use these similarity solutions as a means of estimating the accuracy of approximate solutions arising from the numerical methods described in this thesis.

Similarity solutions can be constructed through the use of similarity variables

$$\xi = \frac{x}{t^\beta}, \qquad \eta = \frac{u}{t^\gamma}, \tag{2.22}$$

where β and γ are as given in the scaling transformation (2.18).

Similarity solutions $u^S(x,t)$ then take the form $\eta = f(\xi)$, or

$$u^S(x,t) = t^\gamma f(x/t^\beta), \tag{2.23}$$

with f sought from a reduced order equation (with $\beta + \gamma = 0$ for mass conserving problems).

Substituting (2.23) into the PDE (2.1) and using (2.20), we can rewrite (2.1) in terms of the similarity variables (2.22) as the ordinary differential equation (ODE)

$$-\beta t^{\gamma-1} \frac{\mathrm{d}}{\mathrm{d}\xi}(\xi\eta) = (-1)^m t^{-(n+1)\beta-(2m+1)\beta} \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\eta^n \frac{\mathrm{d}^{2m+1}\eta}{\mathrm{d}\xi^{2m+1}}\right). \tag{2.24}$$

With β, γ given by (2.21) the terms involving t disappear under the scaling condition (2.19), and (2.24) reduces to the ODE

$$-\beta \frac{\mathrm{d}}{\mathrm{d}\xi}(\xi\eta) = (-1)^m \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\eta^n \frac{\mathrm{d}^{2m+1}\eta}{\mathrm{d}\xi^{2m+1}}\right). \tag{2.25}$$

Solving the ODE given by (2.25) then provides η as a function of ξ and hence (from (2.23)) the similarity solution $u^S(x,t)$ for a given value of m. We shall deal with the second, fourth and sixth-order cases separately.

A Fourth-Order Similarity Solution

In the case of the fourth-order problem (2.3)–(2.5) equation (2.25) (with $m = 1$) leads to

$$\beta \frac{\mathrm{d}}{\mathrm{d}\xi}(\xi\eta) = \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\eta^n \frac{\mathrm{d}^3\eta}{\mathrm{d}\xi^3}\right), \tag{2.26}$$

where we have removed the negative sign from each side of (2.25) arising from the choice $m = 1$.

Closed-form solutions to (2.26) exist for the choice of $n = 1$. In this case, the ODE integrates simply to give (subject to boundary conditions $\eta = \frac{\mathrm{d}\eta}{\mathrm{d}\xi} = 0$ at the $\xi = \xi_b$ and the symmetry condition $\frac{\mathrm{d}\eta}{\mathrm{d}\xi} = 0$ at $\xi = 0$)

$$\eta = \frac{1}{120}\left[\xi_b^2 - \xi^2\right]_+^2. \tag{2.27}$$

In equation (2.27) the subscript + indicates the positive part of $(\xi_b^2 - \xi^2)$. It follows that η (and consequently $u^S(x,t)$) is symmetric about $\xi = 0$ and has a moving compact support (in terms of x).

The similarity solution is therefore a quartic which can be written in the form

$$u^S(x,t) = \frac{1}{120t^\beta}\left[\omega^2 - \frac{x^2}{t^{2\beta}}\right]_+^2,$$

where ω is positive constant related to ξ_b depending upon the initial condition (see Smyth and Hill (1988), Diez et al. (2000)). We note from (2.21) that $\beta = 1/5$ for the $m = 1$, $n = 1$ case and that $u^S(x,t)$ is symmetric about $x = 0$.

An example of such a fourth-order similarity solution (presented in Barrett et al. (1998) among others) is given by

$$u^S(x,t) = \frac{1}{120(t+\kappa)^{1/5}}\left[\omega^2 - \xi^2\right]_+^2, \qquad \xi = \frac{x}{(t+\kappa)^{1/5}}, \tag{2.28}$$

where κ is a nonnegative constant. The position of the free boundaries at time $t = t^0$ is given by $x = \pm\omega(t^0 + \kappa)^{1/5}$.

A Second-Order Similarity Solution

Setting $m = 0$ in (2.25) produces the ODE

$$-\beta \frac{\mathrm{d}}{\mathrm{d}\xi}(\xi\eta) = \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\eta^n \frac{\mathrm{d}\eta}{\mathrm{d}\xi}\right), \tag{2.29}$$

which can be solved for any n, as noted in Smyth and Hill (1988), subject to the boundary condition $\eta = 0$ at $\xi = \xi_b$ and the symmetry condition $\frac{\mathrm{d}\eta}{\mathrm{d}\xi} = 0$ at $\xi = 0$. The solution to this ODE can be written as

$$\eta = \left(\frac{n}{2(n+2)}\right)^{1/n} \left[\xi_b^2 - \xi^2\right]_+^{1/n},$$

and hence (from (2.23)) the similarity solution is given as

$$u^S(x,t) = \frac{1}{t^\beta}\left(\frac{n}{2(n+2)}\right)^{1/n} \left[\omega^2 - \frac{x^2}{t^{2\beta}}\right]_+^{1/n},$$

where ω is related to ξ_b by the initial condition and $\beta = 1/(n+2)$ from (2.21) (since $m = 0$). The similarity solution is again symmetric about $\xi = 0$ with moving compact support (in terms of x). An example of a similarity solution for the second-order problem (2.7)–(2.9) is given by (Pattle (1959), also Barenblatt (1952))

$$u^S(x,t) = \frac{1}{t^\beta}\left(\frac{n\beta}{2}\right)^{1/n} \left[\omega^2 - \frac{x^2}{t^{2\beta}}\right]_+^{1/n}, \tag{2.30}$$

where $\beta = 1/(n+2)$.

A Sixth-Order Similarity Solution

Setting $m = 2$ in (2.25) produces the ODE

$$-\beta \frac{\mathrm{d}}{\mathrm{d}\xi}(\xi\eta) = \frac{\mathrm{d}}{\mathrm{d}\xi}\left(\eta^n \frac{\mathrm{d}^5\eta}{\mathrm{d}\xi^5}\right). \tag{2.31}$$

As in the fourth-order case, closed-form solutions exist only in the case where $n = 1$, in which case the ODE (2.31) integrates, subject to the boundary conditions $\eta = \frac{\mathrm{d}\eta}{\mathrm{d}\xi} = \frac{\mathrm{d}^3\eta}{\mathrm{d}\xi^3} = 0$ at $\xi = \pm\xi_b$ and the symmetry condition $\frac{\mathrm{d}\eta}{\mathrm{d}\xi} = 0$ at $\xi = 0$, to give

$$\eta = \frac{1}{5040}\left[\xi_b^2 - \xi^2\right]_+^3,$$

The corresponding similarity solution to the sixth-order problem (2.10)–(2.12) is of the form

$$u^S(x,t) = \frac{1}{5040t^\beta}\left[\omega^2 - \frac{x^2}{t^{2\beta}}\right]_+^3.$$

where ω is related to ξ_b by the initial condition and where $\beta = 1/7$ from (2.21) in the case where $n = 1$. This solution is symmetric about $x = 0$ and has moving compact support.

An example of a sixth-order similarity solution (presented in Barrett et al. (2004)) is given by

$$u^S(x,t) = \frac{1}{5040(t+\kappa)^{1/7}}\left[\omega^2 - \frac{x^2}{(t+\kappa)^{2/7}}\right]_+^3, \tag{2.32}$$

for constants κ, ω.

2.5 Numerical Methods for Nonlinear Diffusion

As mentioned in §2.4.3, for most choices of m or n in the PDEs (see (2.1), (2.2)) describing nonlinear diffusion in this thesis there are no closed form solutions available. Indeed, for those cases such that closed form solutions exist they are useful only if the initial condition is of the correct form. This lack of explicit solutions motivates the need for numerical methods for use in obtaining approximate solutions to such nonlinear diffusion equations.

We now present a review of some of the different types of numerical method which feature in the literature, with the main focus on numerical methods for fourth-order problems. There is a large body of literature to draw on, and a wide variety of methods which can be used to produce approximate solutions to the nonlinear PDEs, from which a selection is made here.

We shall mention both moving-mesh methods and fixed-mesh methods in 1D, as well as a brief mention on methods in higher dimensions. A particular class of moving-mesh methods, velocity-based methods based on conservation, is of particular relevance to the work presented in this thesis, and as such is covered separately in Chapter 3.

2.5.1 Moving-Mesh Methods for Nonlinear Diffusion

A moving-mesh method allows the mesh of points to evolve over time, which adds an additional complexity to the solution process but has the benefit of allowing moving features of the solution to be modelled more accurately (Budd et al. (2009)). Such methods are also known as *r-refinement* methods, as opposed to *h-refinement* methods in which mesh points are added or removed from the mesh where required. This latter form of adaptivity is typically applied to fixed-mesh methods, hence we make the distinction between the two types of refinement here.

Of the moving-mesh methods which do not belong in the velocity-based class of method, we shall only discuss the Moving Mesh PDE (MMPDE) method.

In the MMPDE method, a mapping is constructed from a fixed reference space in order to satisfy an equidistribution principle, which depends upon a given monitor function $M(x,t)$ (see Huang et al. (1994), Budd et al. (1999), Huang and Russell (2011)).

The core of the method involves obtaining and solving a moving-mesh PDE (MMPDE) which is solved alongside the original (modified) PDE, which together update the solution values and mesh positions. This MMPDE originates from a mapping $x = x(\xi, t)$ from a fixed reference domain Ω_C with coordinates $\xi \in [0,1]$ to the physical domain Ω_P with coordinates $x \in [a(t), b(t)]$.

This mapping, which is based on equidistribution of the monitor function $M(x,t)$ satisfies

$$M(x,t)\frac{\partial x}{\partial \xi} = \theta(t), \qquad x(0,t) = a, \quad x(1,t) = b, \qquad \theta(t) = \int_a^b M(x,t)\,\mathrm{d}x. \tag{2.33}$$

Differentiating (2.33) with respect to ξ,

$$\frac{\partial}{\partial \xi}\left(M\frac{\partial x}{\partial \xi}\right) = 0, \qquad x(0,t) = a, \quad x(1,t) = b. \tag{2.34}$$

A number of variations of (2.34) are used which lead to a range of different MMPDEs. Equation (2.34) ensures that the mesh is equidistributed according to the monitor function chosen.

Once the mapping has been found the rate of change of x is constructed numerically for insertion into a pseudo-Lagrangian form of the original PDE (obtained from a chain

rule and written here in terms of (2.2))

$$\frac{\partial u}{\partial t} - \frac{\mathrm{d}u}{\mathrm{d}x}\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), \tag{2.35}$$

where $\frac{\mathrm{d}u}{\mathrm{d}x}$ denotes the rate of change of u in the moving frame.

We refer the reader to Huang et al. (1994), Budd et al. (2009), Huang and Russell (2011) and the references within for a detailed review of the MMPDE method. In particular, Budd et al. (1999) present an application of the MMPDE method to the porous medium equation for generating self-similar numerical solutions.

2.5.2 Fixed-Mesh Methods for Nonlinear Diffusion

A fixed-mesh method differs from a moving-mesh method (such as that discussed in §2.5.1) in that the discretisation of the domain is such that the mesh points are no longer moving over time. This necessitates a distinction between the boundary of the computational domain and the interfaces which define the support of the solution $u(x,t)$, as the interfaces may not lie exactly on the computational boundaries.

For the general PDE given by (2.1), solutions $u(x,t)$ are sought to

$$\frac{\partial u}{\partial t} = (-1)^m \frac{\partial}{\partial x}\left(u^n \frac{\partial^{2m+1} u}{\partial x^{2m+1}}\right), \qquad \text{in } \Omega \times (t^0, T),$$

where $\Omega := (-B, B)$ is a truncation of the real line such that $x = \pm B$ are fixed points chosen far enough away from the moving interfaces at $x = a(t)$, $x = b(t)$ (see Figure 2.1).

The approximate solution is then sought at each point on a mesh whose points remain fixed over time. Standard numerical methods for boundary value problems (BVPs) can then be used in obtaining the approximation (although there may be issues with representing sharp features of the solution if the density of the mesh points is low in the vicinity of the feature).

Fixed-mesh methods have difficulties modelling the position of a moving boundary, since in general the moving boundary is not located at a mesh point. This generally necessitates the inclusion of some mechanism for determining the position of the interface between two mesh points, such as interpolation or extrapolation. In particular a

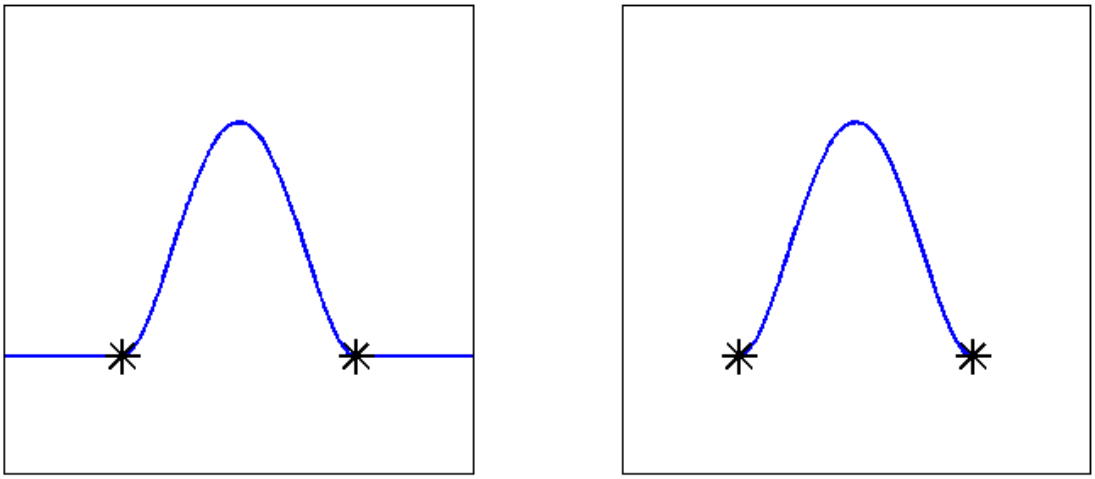

Figure 2.1: Image highlighting the difference between fixed and moving domains Ω_T. The left-hand image shows a fixed-mesh domain, with interfaces given by the black asterisk. The solution outside the interfaces exists but is equal to the value at the interface. The right-hand plot shows a moving domain, where the black asterisk denotes the boundary of the Ω_T. The solution is not defined outside of the boundary.

fixed-mesh method may have difficulty when approximating singularities in a solution, with the qualitative behaviour of the solution differing from the true behaviour (see for example Budd et al. (1996)).

An advantage of fixed-mesh methods is that there exists much analysis associated with such methods, with known results on error estimates and convergence of numerical solutions. In particular, a large body of literature has arisen looking at constructing numerical methods which preserve nonnegativity (or alternatively positivity) of numerical solutions for fixed-mesh methods (see Grün and Rumpf (2000, 2001), Grün (2003) and Jüngel and Pinnau (2001) for examples of such schemes for a range of fourth-order problems).

Finite difference approaches to the problems considered in this thesis often include a regularisation of the diffusion coefficient u^n in order to avoid degeneracy at the moving boundaries. This regularisation allows for the practical construction of zero contact angles solutions (Diez et al. (2000)). An example of such a regularisation, presented in Bernis and Friedman (1990), takes the form

$$u_\epsilon^n = \frac{u^n u^s}{\epsilon u^n + u^s},$$

where $u_\epsilon \to u$ as $\epsilon \to 0$.

Diez et al. (2000) use such a regularisation in their finite difference method, noting that for $n \geq 2$ positivity is preserved. For $n < 2$, an entropy dissipative scheme needs to be combined with such regularisation to preserve positivity. Such methods are described in Zhornitskaya and Bertozzi (2000), which preserve positivity, nonlinear entropy dissipation and surface energy dissipation. In this paper a finite element scheme is employed, using higher order elements.

Regularisation methods, while positivity-preserving, provoke many questions of the solution. For example, as the solution is essentially 'lifted' from $u = 0$ by the small quantity ϵ, it remains unclear how to correctly specify the position of the moving boundaries. It is also unclear how to ensure that the mass of the solution is conserved, since the lifting also increases the mass of the initial condition.

Barrett et al. (1998) present a finite element scheme for solving the fourth-order problem (2.3)–(2.5) without the above regularisation. Convergence of numerical solutions to a weak solution of the PDE in 1D is shown. Nonnegativity of the solution is imposed as a constraint, since numerical solutions cannot be guaranteed to satisfy this property otherwise.

Blowey et al. (2007) use the finite element scheme of Barrett et al. (1998) with some modifications to demonstrate the various small-time behaviours of the free boundary outlined in King and Bowen (2001).

A mixed finite element method is proposed in Burger et al. (2010) for a class of second-order nonlinear PDEs, while Duque et al. (2013) use a finite element method to obtain solutions to a second-order problem with variable exponent n, which differs from the problems considered in this thesis (which have constant n).

The literature for sixth-order problems is much less abundant than for the second and fourth-order problems, which is likely to be due to the fact that less is known about the behaviour of solutions to such problems. In Smith et al. (1996) a numerical method for solving a sixth-order problem related to flow under a nitride cap is given. The method uses finite differences and Newton's method is employed at each time step to solve the resulting system. Evans et al. (2000) provide a finite difference scheme for a sixth-order problem, which is restricted to one space dimension. In both cases, no analysis of the finite difference schemes is given.

One of the first attempts of producing a finite element implementation to the sixth-order problem (2.10)–(2.12) is given in Barrett et al. (2004). The method is an extension of the method of Barrett et al. (1998) for the fourth-order problem (2.3)–(2.5), and the paper notes the lack of research for this order of problem. The method presented is shown to converge for dimensions less than three (although some of the analysis is valid for $d \geq 1$). Flitton and King (2004) provide a brief description of a finite difference scheme for solving a sixth-order problem, but as an appendix to the analytical results given in the paper.

2.5.3 Moving-to-Fixed Domain Mappings

It is common to create numerical methods for solving nonlinear diffusion problems with moving boundaries, such as those considered here, by introducing a mapping from the moving domain to a fixed domain. Once this mapping has been constructed, the transformed problem is then solved on the fixed domain using standard BVP techniques.

An example of this type of method in the literature arises in the field of modelling tumour growth. Breward et al. (2002) transform the moving domain $x \in [0, b(t)]$ to a fixed domain $\xi \in [0, 1]$ by using a coordinate transform based on a simple scaling and then solve the problem on the fixed domain. This method is also described in Lee (2011) (also Lee et al. (2013)), who uses a moving finite difference method to find approximate solutions to the tumour growth problem. It is mentioned in Lee (2011) that, although standard techniques can be used in the transformed problem, care must be taken to avoid disrupting the balance between diffusion and reaction laws present in the original problem.

Another example of this type of method for a fourth-order problem is given in Giacomelli et al. (2008), where after the moving-to-fixed domain mapping has been performed the problem is further formulated by introducing a new variable which splits the operator into linear and nonlinear parts. The authors then prove existence, uniqueness and smoothness of the transformed problem. The problem considered in this paper is the fourth-order PDE (2.3a), subject to zero solution, zero contact angle and

$$V = \lim_{u>0 \ni x \to \partial\{u>0\}} \frac{\partial^3 u}{\partial x^3}, \quad \text{on } \partial\{u > 0\},$$

at the boundaries, where V denotes the speed at the free boundary. This is a slightly different problem to those considered in this thesis in that conservation of mass is not enforced, at the expense of enforcing the limiting process for the boundary velocity. The problem also differs in that the authors initially assume that $u_0 > 0$ in the initial interval (s_0, ∞), where s_0 is the initial position of the free boundary, which is defined by

$$s(t) = \inf\{y \in \mathbb{R} : u(t, \cdot) > 0 \text{ in } (y, \infty)\}.$$

Figure 2.2 provides an illustrative example of the type of initial condition considered in Giacomelli et al. (2008) for $s_0 = 2$.

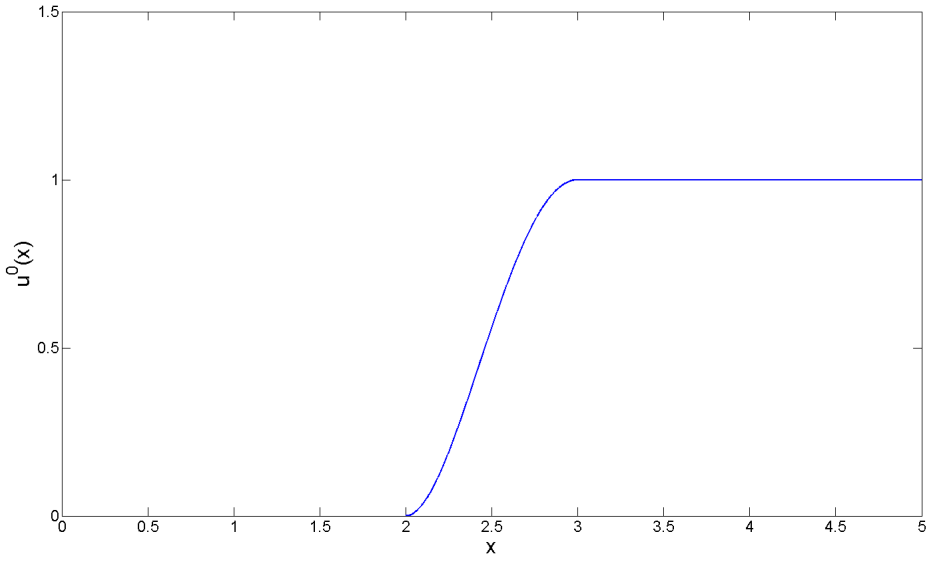

Figure 2.2: Example of the type of initial condition considered in Giacomelli et al. (2008) with $s_0 = 2$.

A further example of this type of method is given in Socolovsky (1988). The issue of how to track moving boundaries using numerical methods is covered, with regard to the porous medium equation and a generalised heat equation. Once transformed onto the fixed domain, the problems in Socolovsky (1988) are solved using finite difference and finite element discretisations, with numerical results presented for the finite difference schemes for the various problems considered.

2.6 Numerical Methods in Higher Dimensions

All the methods mentioned so far have been in 1D, as these are relevant to the work presented in later chapters of this thesis. There also exist numerical methods in higher dimensions which provide approximate solutions. We now provide a brief insight into some of these methods.

2.6.1 Meshless Methods

The numerical methods described in §2.5.1–§2.5.2 all involve obtaining the solution on a given mesh of points in 1D. If considered in 2D, this mesh of points would include a strong form of connectivity, which links the different points together.

An alternative type of numerical method is known as a meshless, or mesh-free, method. In this type of method a number of points are distributed throughout the continuous domain, without concern for the connectivity of the points. The solution is then sought on this cloud of points.

By ignoring connectivity between the points, the issue of mesh tangling is avoided, which can cause problems for moving-mesh methods (see, e.g., Budd et al. (2009)) in that instabilities can arise in the solution which cause an undesired blow-up. Avoiding connectivity concerns also reduces the need for mesh generation procedures, which can be complex and/or computationally expensive (Ma et al. (2008)).

The meshless method has issues with poorly approximating the solution if the points are coarsely distributed through the domain (Belytschko et al. (1996)), or when points are not positioned to accurately capture the dynamics of the solution. This can also occur in fixed/moving-mesh methods, particularly if the mesh points are not positioned to capture singularities (Nguyen et al. (2008)), so this issue is not limited to meshless methods.

For a review of meshless methods, we refer to Belytschko et al. (1996), or the more recent review paper of Nguyen et al. (2008). Ma et al. (2008) also present a study of an adaptive algorithm for solving the Euler equation of gas dynamics.

2.6.2 MMPDE Methods in 2D

The MMPDE method described in §2.5.1 can be extended into higher dimensions. A major difference between 1D and higher dimensional formulations of the MMPDE method is that the mapping obtained through equidistribution is no longer unique (Budd et al. (2009)). In order to obtain a unique mapping additional concerns such as the skewness of the mesh and overlapping must be taken into consideration (Huang and Russell

(1997)).

There are a number of different ways of ensuring equidistribution in higher dimensions, but the method presented in Huang and Russell (1997) and Huang and Russell (1999) makes use of variational procedures and gradient flow equations to obtain the MMPDE required. These variational-based methods are based on minimising a functional $I(\boldsymbol{\xi})$, which in d dimensions has the general form (Budd et al. (2009))

$$I(\boldsymbol{\xi}) = \int_{\Omega_P} G(\mathbf{M}, \boldsymbol{\xi}, \boldsymbol{\nabla}\xi_i)\mathrm{d}\mathbf{x}, \quad i = 1, \ldots, d,$$

for some continuous function G, monitor function $\mathbf{M}$ (which could be scalar or matrix-valued) and where $\boldsymbol{\nabla}$ is the gradient operator in terms of physical coordinates $\mathbf{x}$.

The advantage of such formulations is that in a non-convex domain node tangling is less likely than with other formulations. The resulting systems are highly nonlinear however, which is a disadvantage of the method due to the additional computational costs.

We refer the reader to Huang and Russell (2011) and Budd et al. (2009) for more details on this method.

2.6.3 The Parabolic Monge-Ampère Method

If the mapping is represented by a gradient, then equidistribution in 2D leads to a Monge-Ampère equation. This process is related to minimisation of a functional

$$I = \int_{\Omega_C} |\mathbf{F}(\boldsymbol{\xi}, t) - \boldsymbol{\xi}|^2 \mathrm{d}\boldsymbol{\xi},$$

where $\mathbf{F}$ is the mapping between computational and physical space. If the mapping is written as the gradient $\boldsymbol{\nabla}_{\boldsymbol{\xi}}$ of a mesh potential $P(\boldsymbol{\xi}, t)$, the Hessian $H(P)$ of such a potential gives rise to the Monge-Ampère equation

$$M(\boldsymbol{\nabla}_{\boldsymbol{\xi}} P, t)H(P) = \theta(t), \tag{2.36}$$

where $\theta(t)$ is defined as the 2D equivalent of (2.33).

Budd and Williams (2009) provide a description of the Parabolic Monge-Ampère method as a means of generating an equidistributed mesh in higher dimensions, noting that for some applications an equidistributed mesh which is as close as possible to a uniform mesh is desirable. The Parabolic Monge-Ampère solver then arises from a parabolic relaxation of (2.36), involving a time-evolving function $Q(\boldsymbol{\xi}, t)$ with a corresponding mesh, giving

$$\epsilon(I - \gamma\Delta_{\boldsymbol{\xi}})\frac{\partial Q}{\partial t} = (H(Q)M(\boldsymbol{\nabla}_{\boldsymbol{\xi}}Q))^{1/d}, \tag{2.37}$$

where ϵ denotes the relaxation rate and γ controls the amount of spatial smoothing.

Equation (2.37) is solved alongside the underlying PDE in order to find a mesh which evolves towards the gradient of solutions to (2.36) over a short time scale (Budd et al. (2013)).

For further details, see Budd and Williams (2009) and Budd et al. (2013)

2.7 Summary of this Chapter

In this chapter we have introduced the different orders of nonlinear diffusion PDE considered in this thesis. By discussing existing results in the literature we have identified some of the issues that can arise in obtaining solutions to these PDEs as well as presenting existing results for these problems. We then discussed various types of numerical methods which exist for obtaining approximate solutions to the problems.

In the next chapter we shall focus on velocity-based moving-mesh methods, before presenting a particular version of such a method, based on conservation, which we shall use throughout the remainder of this thesis in finding approximate solutions to second, fourth and sixth-order problems.

Chapter 3

Velocity-Based Moving Mesh Methods

3.1 Aims of this Chapter

In this chapter we discuss some velocity-based moving-mesh methods, one of which forms the focal point of the numerical methods discussed in this thesis. We also provide some insight into the literature that exists for this type of method.

We shall then present in detail a particular implementation of the approach for use in solving the fourth-order problem (2.3)–(2.5). We also describe how this implementation can be modified for finding approximate solutions to the second-order ((2.7)–(2.9)) and sixth-order (2.10)–(2.12) problems. In this chapter the method is implemented using finite difference approximations.

3.2 Velocity Based Methods

A velocity-based method is a type of moving-mesh method in which the nodes X_i^k of a mesh $\mathbf{X}^k = \{X_i^k\}$, $i = 1, \ldots, N$, are moved by a velocity V_i^k at each node. The superscript $k = 0, \ldots, K$ indicates the kth of K time steps. The velocity V_i^k is used to update node X_i^k by time stepping forward to find the node X_i^{k+1} at the new time step.

This is achieved through time stepping the ODE

$$\frac{\mathrm{d}X_i}{\mathrm{d}t} = V_i,$$

at each node of the mesh.

There are various methods of determining the point velocities in a velocity-based method and we shall highlight some of these in this section. One of these methods (the conservation based method) forms the basis of the Moving Point Conservation Method (MPCM) which is used in this thesis to approximate high order nonlinear diffusion.

We now survey some of the key velocity-based methods in the literature.

3.2.1 Moving Finite Element Method

The Moving Finite Element (MFE) method is an early example of a velocity-based method developed by Miller and Miller (1981) and Miller (1981). While a potentially powerful tool for solving nonlinear PDEs it requires considerable care in the implementation in order to avoid inherent singularities.

In the discrete MFE method, the approximate solution $U(x,t)$ is written as a linear combination of time dependent basis functions $w_j(x,t)$ with moving nodes, as

$$U(x,t) = \sum_j U_j(t) w_j(x,t), \tag{3.1}$$

with coefficients $U_j(t)$.

The MFE method is a general method for PDEs of the form $\frac{\partial u}{\partial t} = \mathcal{L}u$, where $\mathcal{L}$ is a purely spatial operator. The method involves the minimisation over $\frac{\mathrm{d}U_i}{\mathrm{d}t}$ (the rate of change of U_i in the moving frame) and $\frac{\mathrm{d}X_i}{\mathrm{d}t}$ of the L^2 norm of the residual of the Pseudo-Lagrangian form of the PDE (see (2.35)) in the form

$$\int_{a(t)}^{b(t)} \left(\sum_j \frac{\mathrm{d}U_j}{\mathrm{d}t} w_j - \sum_j \frac{\mathrm{d}U}{\mathrm{d}x}\frac{\mathrm{d}X}{\mathrm{d}t} w_j - \mathcal{L}U \right)^2 \mathrm{d}x, \tag{3.2}$$

at each node of the mesh. Carrying out the minimisation over $\frac{\mathrm{d}U_i}{\mathrm{d}t}$ and $\frac{\mathrm{d}X_i}{\mathrm{d}t}$ of (3.2) leads to the following pair of equations for each node i:

$$\int_{a(t)}^{b(t)} w_i \left(\frac{\mathrm{d}U}{\mathrm{d}t} - V\frac{\partial U}{\partial x} \right) W \,\mathrm{d}x = \int_{a(t)}^{b(t)} w_i \mathcal{L}U \,\mathrm{d}x, \tag{3.3}$$

$$\int_{a(t)}^{b(t)} \left(\frac{\partial U}{\partial x} w_i\right) \left\{\frac{\mathrm{d}U}{\mathrm{d}t} - V\frac{\partial U}{\partial x}\right\} \mathrm{d}x = \int_{a(t)}^{b(t)} \left(\frac{\partial U}{\partial x} w_i\right) \mathcal{L}U \,\mathrm{d}x. \tag{3.4}$$

In later developments a more general weight function was used as standard, with w_i replaced by $W_i = \frac{1}{\sqrt{1+\nabla U_i}}$ (see Carlson and Miller (1998a,b)).

The solution of (3.3)–(3.4) determines the mesh velocities $\frac{\mathrm{d}X_i}{\mathrm{d}t}$ and solution rates of change $\frac{\mathrm{d}U_i}{\mathrm{d}t}$ at each node in the mesh. Care must be taken however, in that the set of equations may become singular and as such cannot be solved uniquely (Miller (1981)). The two situations in which singularities occur are when a basis function is redundant in (3.1) or when nodes overtake. The possibility of a singular set of equations appearing leads to the need to introduce penalty functions into the minimisation of (3.2) (Miller and Miller (1981)).

The MFE method has been used successfully in Carlson and Miller (1998a) and Carlson and Miller (1998b), but the need for fine-tuning to avoid singularities is a big disadvantage for the method (Budd et al. (2009)).

3.2.2 The GCL Method

The Geometric Conservation Law (GCL) method (Cao et al. (2002)) arose as a means of deriving an equation for the mesh velocity through differentiation of the equidistribution equation with respect to time (see §2.5.1 for information on this equation, and also Budd et al. (2009)). A simple explanation of the method in 1D is presented here (taken from Cao et al. (2002)), but the reader is referred to this paper plus that of Budd et al. (2009) and Baines et al. (2011) for more details, in particular the higher dimensional case.

Consider a coordinate transformation $x = x(\xi, t)$ between a computational domain Ω_C and a physical domain $\Omega_P(t)$. Let A_C be an arbitrary subset of Ω_C with boundary ∂A_C. Under the coordinate transformation there exists a corresponding subset of the physical domain $A_P(t) \subseteq \Omega_P(t)$.

The integral form of the GCL arises from noting that the change in volume of $A_P(t)$ is equal to the total inward flux over the boundary $\partial A_P(t)$, i.e.

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{A_P(t)} \mathrm{d}x = \Big[v\Big]_{\partial A_P(t)}, \tag{3.5}$$

with v the velocity $\frac{\mathrm{d}x}{\mathrm{d}t}$ as previously defined.

Under the coordinate transformation we may write

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{A_P(t)} \mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}t}\int_{A_C} J(\xi,t)\,\mathrm{d}\xi = \int_{A_C} \frac{DJ}{Dt}\,\mathrm{d}\xi, \tag{3.6}$$

where J is the Jacobian of the coordinate transformation and $\frac{DJ}{Dt}$ denotes the total derivative of J.

We may rewrite the right-hand side of (3.5) using the divergence theorem as

$$\Big[v\Big]_{\partial A_P(t)} = \int_{A_P(t)} \frac{\partial v}{\partial x}\,\mathrm{d}x = \int_{A_C} \left(\frac{\partial v}{\partial x}\right) J(\xi,t)\,\mathrm{d}\xi = \int_{A_C} \frac{DJ}{Dt}\,\mathrm{d}\xi,$$

from (3.6). Since A_C is arbitrary, this leads to the differential form of the GCL

$$\frac{\partial v}{\partial x} = \frac{1}{J}\frac{DJ}{Dt}. \tag{3.7}$$

This differential form can be used as the basis for moving-mesh methods. The choice

$$J(\xi,t) = \frac{c(\xi)}{M(x,t)}, \tag{3.8}$$

as highlighted in Cao et al. (2002), for some positive monitor function M corresponds to equidistribution, where $c(\xi)$ is a constant independent of time determined from the initial coordinate transform.

Substituting (3.8) into (3.7) we see that

$$\begin{aligned} \frac{\partial v}{\partial x} &= -\frac{1}{M}\frac{DM}{Dt}, \\ \Rightarrow \quad \frac{\partial M}{\partial t} + v\frac{\partial M}{\partial x} &= -M\frac{\partial v}{\partial x}, \\ \Rightarrow \quad \frac{\partial (Mv)}{\partial x} + \frac{\partial M}{\partial t} &= 0. \end{aligned} \tag{3.9a}$$

For a given M, equation (3.9a) can be used to define v, with suitable boundary conditions on v. In 1D, this problem can be solved uniquely, but in higher dimensions additional conditions are required in order to solve uniquely for v (Budd et al. (2009)).

As an example, the monitor function $M = u$ can be used to conserve the mass of the solution. Then (3.9a) becomes

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(uv) = 0, \tag{3.10}$$

which is the Eulerian form of mass conservation. We shall use this property in the MPCM described in §3.3. Combined with $\frac{\partial u}{\partial t} = \mathcal{L}u$, (3.10) can be used to obtain an expression for v.

The solution of the GCL mesh equations takes place in the physical domain $\Omega(t)$ as opposed to the fixed computational domain, which negates any speed advantage present in solving for the mapping in the computational domain (such as that obtained through use of spectral methods for example, see Budd et al. (2009)). On the other hand there is no need to derive the velocity from the mapping.

3.2.3 The Deformation Map Method

The Deformation Map method originated from the work of Moser (1965) and Dacorogna and Moser (1990) in proving the existence of a C^1 diffeomorphism between two domains. The mapping was then used to generate moving meshes in Liao and Anderson (1992) and Bochev et al. (1996) (among others). The velocities are generated in a similar way to the GCL method with modified dependent variables We refer the reader to the given references for more information.

3.2.4 A Conservation-Based Method

In a conservation-based method, the velocity is found directly through a combination of the conservation equation (3.10) (in Eulerian form) and the PDE. This correspondence between the two is used to define the velocity, which can then be used to update the mesh points. The emphasis is therefore on the velocity, with the solution derived afterwards from the Lagrangian form of the conservation law.

The conservation method developed in Baines et al. (2005, 2006, 2011) is such an example, where the mass (alternatively area under the solution) is conserved over time. The implementation is built in a similar way as the GCL method described in §3.2.2, but in this case the velocities are found directly from the conservation equation as opposed to solving a minimisation problem as in Cao et al. (2002).

In Baines et al. (2005, 2006, 2011) the method is described for both mass-conserving

and non mass-conserving problems in terms of finite elements in the moving frame. Finite difference implementations have also been used in Lee (2011), Blake (2001) and Partridge (2013) for a variety of different problems.

In Chapter 2 we demonstrated that the fourth-order ((2.3)–(2.5)) , second-order ((2.7)–(2.9)) and sixth-order ((2.10)–(2.12)) problems were mass-conserving. We now use this property to define a consistent local conservation of mass principle for obtaining the velocity at all interior points of the domain $\Omega(t)$.

3.2.5 Local Conservation of Mass

We first define a local conservation of mass principle, consistent with conservation of total mass, in which the mass under the curve $u(x,t)$ from $a(t)$ up to an arbitrary point $\hat{x}(t) \in \Omega(t)$ is conserved in time, i.e.

$$\int_{a(t)}^{\hat{x}(t)} u(x,t)\,\mathrm{d}x = c(\hat{x}), \tag{3.11}$$

where $c(\hat{x})$ is independent of time. equivalently

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{\hat{x}(t)} u(x,t)\,\mathrm{d}x = 0. \tag{3.12}$$

We shall describe the derivation of the velocity in the case of the PDE (2.2), although the process applies for any of the problems found in §2.2 since they are all mass-conserving, being subject to zero-flux boundary conditions. The steps involved in defining the local conservation of mass principle are similar to those taken in §2.4.1 for demonstrating that the total mass of the PDE is mass conserving.

We begin by differentiating the left-hand side of (3.12) using Leibniz's Integral Rule, giving

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{\hat{x}(t)} u(x,t)\,\mathrm{d}x = \int_{a(t)}^{\hat{x}(t)} \frac{\partial u}{\partial t}\,\mathrm{d}x + \frac{\mathrm{d}\hat{x}}{\mathrm{d}t}u(\hat{x}(t),t) - \frac{\mathrm{d}a}{\mathrm{d}t}u(a(t),t).$$

From (3.12) and (2.2),

$$\int_{a(t)}^{\hat{x}(t)} \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right)\,\mathrm{d}x + \frac{\mathrm{d}\hat{x}}{\mathrm{d}t}u(\hat{x}(t),t) - \frac{\mathrm{d}a}{\mathrm{d}t}u(a(t),t) = 0,$$

Integrating and writing

$$\frac{\mathrm{d}a}{\mathrm{d}t} = v(a(t), t), \qquad \frac{\mathrm{d}\hat{x}}{\mathrm{d}t} = v(\hat{x}(t), t),$$

we arrive at

$$\left[u^n \frac{\partial q}{\partial x} + uv\right]_{a(t)}^{\hat{x}(t)} = 0, \tag{3.13}$$

which implies (due to the zero flux boundary condition) that

$$v = -u^{n-1} \left.\frac{\partial q}{\partial x}\right|_{x=\hat{x}(t)} \tag{3.14}$$

for all $\hat{x}(t) \in \Omega(t)$ where $u \neq 0$ (i.e. excluding the boundary points).

At the boundary points where $u = 0$ we are unable to solve (3.13) for $v(x,t)$, but provided $u(x,t)$ is continuous up to the boundary it is possible to take the limit of (3.14) as $u \to 0$.

The method is summarised as follows: For a given function $q(x,t)$, we can obtain $v(x,t)$ using (3.14). Once $v(x,t)$ has been found points of the domain can be moved using

$$\frac{\mathrm{d}\hat{x}}{\mathrm{d}t} = v, \tag{3.15}$$

where $\hat{x}(t)$ coincides instantaneously with the Eulerian coordinate x. Having obtained the new $\hat{x}$ positions, the solution $u(x,t)$ can be deduced from (3.11).

We next describe the main implementation of this approach as used in this thesis, called the Moving Point Conservation Method (MPCM), which is implemented using finite differences and is used to approximate solutions to the nonlinear diffusion equations described in §2.2. The MPCM operates by assigning a velocity (approximated using (3.14)) to each node of a mesh of points $\mathbf{X}^k$ which discretise the domain $\Omega(t)$ at a specific time level t^k. The mesh points $\mathbf{X}^k$ are then updated by a time stepping scheme using this velocity, before the solution is recovered on the updated mesh using the Lagrangian conservation form (3.11).

In the next section we describe the process involved in solving the fourth-order problem (2.3)–(2.5) using the MPCM. We focus on the fourth-order problem (2.3)–(2.5)

in the next section, but shall provide details on how the implementation can be modified for second and sixth-order problems in §3.4 and §3.5.

3.3 A Finite Difference Implementation of the MPCM for the Fourth-Order Problem

We now calculate approximate solutions to the fourth-order problem (2.3)–(2.5) using a finite difference implementation of the MPCM. In the case where a self-similar solution exists (i.e. for $n = 1$) $u(x,t)$ is a quartic function (Smyth and Hill (1988)), with $q(x,t)$ therefore a quadratic function from (2.3b). From (3.14), the velocity $v(x,t)$ is therefore a linear function. We shall use this information in the implementation of the finite difference method, although the method itself applies to non self-similar cases.

3.3.1 Discretisations

At each time t^k, the interval $(a(t^k), b(t^k))$ is discretised by an ordered mesh of N nodes $a(t^k) = x_1(t^k) < x_2(t^k) < \ldots < x_N(t^k) = b(t^k)$.

We shall denote by F_i^k the approximation of the function $f(x,t)$ at time $t = t^k$, node X_i^k, i.e. $F_i^k \approx f(X_i^k, t^k)$. We shall seek approximations

$$\mathbf{U}^k := \{U_i^k\}, \qquad \text{for } i = 2, \ldots, N-1, \tag{3.16a}$$

$$\mathbf{Q}^k := \{Q_i^k\}, \qquad \text{for } i = 1, \ldots, N, \tag{3.16b}$$

$$\mathbf{V}^k := \{V_i^k\}, \qquad \text{for } i = 1, \ldots, N, \tag{3.16c}$$

to $u(x,t)$, $q(x,t)$ and $v(x,t)$ on the moving mesh, with $U_1^k = U_N^k = 0$ for all k from (2.5a).

3.3.2 Initial Conditions

On the initial mesh $\mathbf{X}^0$ we sample the initial condition $u^0(x)$ at each node to get an initial approximation $\mathbf{U}^0$ by setting

$$U_i^0 = u^0(X_i^0),$$

for $i = 2, \ldots, N-1$. We enforce the boundary condition $u = 0$ strongly and therefore set

$$U_1^0 = U_N^0 = 0.$$

3.3.3 Approximating $q(x,t)$

To approximate the velocity we require an approximation to the function $q(x,t)$ at each node in the mesh. As there are no boundary conditions prescribed for $q(x,t)$, we shall require the approximation to be valid for all $i = 1, \ldots, N$.

Given the values U_i^k, $i = 1, \ldots, N$, (which we denote in this section by U_i for ease of notation) we compute Q_i^k, $i = 1, \ldots, N$, (denoted Q_i for ease of notation) as follows (the calculation of interior and boundary points is considered separately since the method used for deriving these values differs in each case).

Interior Points

As a first approximation, at each time step consider the three-point central differencing scheme on a variable mesh,

$$Q_i \approx -\frac{\frac{U_{i+1}-U_i}{X_{i+1}-X_i} - \frac{U_i-U_{i-1}}{X_i-X_{i-1}}}{\frac{1}{2}(X_{i+1}-X_{i-1})}, \tag{3.17}$$

for $i = 2, \ldots, N-1$.The local truncation error (LTE), τ_i, of this approximation can be shown (by Taylor Series expansions and a Mean Value Theorem) to be equal to

$$\tau_i = -\frac{(X_{i+1} - 2X_i + X_{i-1})}{3} q_i' - \frac{1}{12} \frac{(X_{i+1}-X_i)^3 + (X_i - X_{i-1})^3}{X_{i+1} - X_{i-1}} q_i'' \Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$, where the prime $'$ denotes a derivative with respect to x.

Boundary Points

The scheme (3.17) holds only for the interior points of the mesh since they involve central differences. At the boundary we require a different approach with which to calculate the equations for the first and last points of the mesh.

The approach used here is that of extrapolation using a second-order Lagrange polynomial. Since $q(x,t)$ is a quadratic in the self-similar case this is the appropriate degree of polynomial required here.

Performing an extrapolation using second-order Lagrange polynomials we arrive at an equation for determining the value Q_1 at a given time step:

$$L_1(X_4)Q_1 + L_2(X_4)Q_2 + L_3(X_4)Q_3 - Q_4 = 0, \tag{3.18}$$

where the $L_1, \ldots, L_4$ are the second-order Lagrange polynomials.

Similarly, an equation for determining the value of Q_N at a given time step is

$$-Q_{N-3} + L_{N-2}(X_{N-3})Q_{N-2} + L_{N-1}(X_{N-3})Q_{N-1} + L_N(X_{N-3})Q_N = 0. \tag{3.19}$$

The Matrix System

To obtain the values of $\mathbf{Q}$ at a given time step at each of the nodal positions we solve the matrix system

$$S\mathbf{Q} = T\mathbf{U}, \tag{3.20}$$

where $\mathbf{Q} = \{Q_i\}$ $(i = 1, \ldots, N)$, $\mathbf{U} = \{U_i\}$ $(i = 1, \ldots, N)$ and S, T are matrices of size $N \times N$. Rows $2, \ldots, N-1$ of these matrices come from equation (3.17), while the first and final rows are given by equations (3.18) and (3.19) respectively. The vector $\mathbf{U}$ includes the boundary values $U_1 = 0$ and $U_N = 0$ as well as the interior values.

By combining these equations, we see that the matrix S is of the form

$$S = \begin{pmatrix} L_1(X_4) & L_2(X_4) & L_3(X_4) & -1 & 0 & \ldots & 0 \\ 0 & 1 & 0 & & & & \vdots \\ & 0 & 1 & 0 & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & & 0 & 1 & 0 & \\ \vdots & & & & 0 & 1 & 0 \\ 0 & \ldots & 0 & -1 & L_{N-2}(X_{N-3}) & L_{N-1}(X_{N-3}) & L_N(X_{N-3}) \end{pmatrix}. \tag{3.21}$$

The matrix T is of the form

$$T = \begin{pmatrix} 0 & \dots & & & & \dots & 0 \\ T_2^- & T_2 & T_2^+ & 0 & & & \vdots \\ 0 & T_3^- & T_3 & T_3^+ & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & 0 & T_{N-2}^- & T_{N-2} & T_{N-2}^+ & 0 \\ \vdots & & & 0 & T_{N-1}^- & T_{N-1} & T_{N-1}^+ \\ 0 & \dots & & & & \dots & 0 \end{pmatrix}, \tag{3.22}$$

where for $i = 2, \dots, N-1$

$$\begin{aligned} T_i^- &= -\frac{1}{\frac{1}{2}(X_i - X_{i-1})(X_{i+1} - X_{i-1})}, \\ T_i &= \frac{1}{\frac{1}{2}(X_{i+1} - X_{i-1})} \left(\frac{1}{(X_{i+1} - X_i)} + \frac{1}{(X_i - X_{i-1})} \right), \\ T_i^+ &= -\frac{1}{\frac{1}{2}(X_{i+1} - X_i)(X_{i+1} - X_{i-1})}. \end{aligned}$$

The matrix S is pentadiagonal, although only due to the first and last rows. The remaining rows of S are diagonal.

The necessity for the interpolating polynomials at the boundary points rather than one-sided differences comes from the lack of a boundary condition on $q(x,t)$. Use of one-sided differences at the boundary points again results in a pentadiagonal matrix S, but the system is not solvable since S in this case is singular. Without a suitable condition on $q(x,t)$ it is not possible to avoid S being singular and hence a different method is required.

Having solved (3.20) for $\mathbf{Q}$, we can now use it to determine the nodal velocity.

3.3.4 The Nodal Velocity

Using the local conservation of mass principle (3.14) we have that the deformation velocity is given by $v(x,t) = -u^{n-1}\frac{\partial q}{\partial x}$ at all points in the domain $\Omega(t)$. We now discretise this equation to obtain an approximation to $v(x,t)$ at each node in the mesh.

Given U_i^k and Q_i^k, both for $i = 1, \ldots, N$, (denoted by U_i and Q_i respectively for ease of notation) we compute V_i^k, $i = 1, \ldots, N$, (denoted V_i for ease of notation) as follows.

The gradient q' of $q(x,t)$ in (X_{i-1}, X_i) is approximated by

$$Q'_{i-1/2} = \frac{Q_i - Q_{i-1}}{X_i - X_{i-1}},$$

say, and is of first order accuracy. A second order approximation can be obtained by a linear interpolation of these gradients through the points $(X_{i-1/2}, Q'_{i-1/2})$ and $(X_{i+1/2}, Q'_{i+1/2})$ such that

$$V_i = -(U_i^{n-1}) \left(\frac{\dfrac{(Q_{i+1} - Q_i)}{(X_{i+1} - X_i)^2} + \dfrac{(Q_i - Q_{i-1})}{(X_i - X_{i-1})^2}}{\dfrac{1}{(X_{i+1} - X_i)} + \dfrac{1}{(X_i - X_{i-1})}} \right), \tag{3.24}$$

for $i = 2, \ldots, N-1$.

The local truncation error of the velocity calculated by (3.24) at node i, denoted by $\tilde{\tau}_i$, can be shown (using Taylor Series and a Mean Value Theorem) to be equal to

$$\tilde{\tau}_i = \frac{(X_{i+1} - X_i)(X_i - X_{i-1})}{6} q_i'''\Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$.

At the boundaries, we use a quadratic extrapolation through adjacent values of V_i in order to obtain the values of V_1 and V_N. An extrapolation is used since at the boundary (where $u = 0$) the approximation (3.24) cannot be applied.

The nodal velocities $\mathbf{V}$ can then be used to update the nodal positions at the next time step.

3.3.5 Updating the Nodal Positions

With the velocity of each node calculated, the updated positions of the nodes $\mathbf{X}^{k+1}$ are found by approximately solving the ODEs (3.15) in the form

$$\frac{\mathrm{d}X_i}{\mathrm{d}t} = V_i, \tag{3.25}$$

for $i = 1, \ldots, N$, by a time stepping scheme.

We approximate the ODEs (3.25) by the explicit Forward Euler time stepping method

$$X_i^{k+1} = X_i^k + \Delta t V_i^k, \tag{3.26}$$

where Δt is the size of the time step used, for $i = 1, \ldots, N$. The scheme (3.26) is not scale invariant with a fixed Δt, which we shall discuss further in Chapter 4.

The size of Δt in (3.26) is of great importance to the successful updating of mesh positions from one time step to the next. In a moving-mesh method the value of Δt must be chosen in order to avoid node tangling occurring (see §3.3.7 for a description of node tangling and the consequences of such an occurrence). The size of Δt also influences the accuracy of the resulting mesh positions. The local truncation error of (3.26), which we denote by τ_i, is

$$\tau_i = \frac{\Delta t}{2} \frac{\mathrm{d}^2 x}{\mathrm{d}x^2}\bigg|_{t^*},$$

for some $t^* \in (t, t + \Delta t)$. This truncation error indicates that a smaller value of Δt will lead to a more accurate resulting mesh position at the updated time step.

3.3.6 Updating the Solution Values

We now recover the solution on the updated mesh $\mathbf{X}^{k+1}$ using the mass constants of (3.11). The discretised mass can be defined using a discrete form of the total mass ρ of the solution (2.15). We discretise (2.15) using the composite trapezoidal rule

$$\rho \approx \frac{1}{2} \sum_{j=2}^{N} (X_j^{k+1} - X_{j-1}^{k+1})(U_j^{k+1} - U_{j-1}^{k+1}), \tag{3.27}$$

which is constant for all k due to mass conservation.

Expanding and rearranging the terms in (3.27) we see that, using the boundary conditions $U_1^{k+1} = U_N^{k+1} = 0$,

$$\frac{1}{2} \sum_{j=2}^{N} (X_j^{k+1} - X_{j-1}^{k+1})(U_j^{k+1} - U_{j-1}^{k+1}) \equiv \frac{1}{2} \sum_{j=2}^{N} (X_{j+1}^{k+1} - X_{j-1}^{k+1}) U_j^{k+1}.$$

Denoting by ρ_i the mass constants

$$\rho_i := \frac{1}{2}(X_{i+1}^{k+1} - X_{i-1}^{k+1}) U_i^{k+1}, \tag{3.28}$$

centred at node i of the mesh, we see that $\sum_i \rho_i = \rho$.

As the total mass is conserved, it is consistent to make the ρ_i constant for all time and this is the approximation to (3.11) that we use. Due to conservation of mass, we can use the initial values X_i^0 and U_i^0 at time $t = t^0$ to calculate the values of the mass constants

$$\rho_i = (X_{i+1}^0 - X_{i-1}^0)U_i^0, \qquad \text{for } i = 2, \ldots, N-1. \tag{3.29}$$

We can use the ρ_i to recover the solution at any given time. From (3.28) and (3.29), we use the new nodal positions to update the solution values for our approximation $\mathbf{U}^{k+1}$ to $u(x,t)$. The updated solution values are therefore calculated from the constants ρ_i and the new mesh positions using

$$\begin{aligned} U_i^{k+1} &= \frac{\rho_i}{X_{i+1}^{k+1} - X_{i-1}^{k+1}} \\ &= \frac{(X_{i+1}^0 - X_{i-1}^0)U_i^0}{X_{i+1}^{k+1} - X_{i-1}^{k+1}}, \end{aligned} \tag{3.30}$$

for $i = 2, \ldots, N-1$.

3.3.7 Node Tangling

As mentioned in §3.3.5, if the size of the time step Δt is taken to be too large when updating the mesh positions from time t^k to t^{k+1}, node tangling can occur.

At time t^k, an untangled mesh is a mesh in which the values of X_i^k (for $i = 1, \ldots, N$) are monotonic, i.e.

$$X_i^k < X_{i+1}^k, \qquad \text{for } i = 1, \ldots, N-1.$$

If this monotonicity is violated as a result of Δt being too large, then *node tangling* is said to have occurred.

In the MPCM, if node tangling occurs, then (3.30) will result in the recovered solution becoming negative in the vicinity of the tangled nodes. This negativity happens when the $X_{i+1}^{k+1} - X_{i-1}^{k+1}$ factor switches from positive to negative as a result of the node tangling.

One of the consequences of performing solution recovery in the manner described in §3.3.6 is that a nonnegative initial condition will remain nonnegative for all later times (provided monotonicity of the mesh is upheld). This consequence is desirable since it mimics the properties of the underlying problem (see §2.2.2 and Bernis and Friedman (1990)). Node tangling is therefore extremely important to avoid when choosing the size of Δt in (3.26).

We next detail how the implementation of the MPCM described above can be modified for use in the second and sixth-order problems detailed in §2.2.

3.4 Implementation of the MPCM for Second-Order Problems

We now outline the steps required for implementing the MPCM for the second-order problem given by (2.7)–(2.9).

The implementation of the MPCM described in §3.3 requires some modifications in order to be valid for the second-order problem (2.7)–(2.9), with the most important change being that $q(x,t) = u(x,t)$, so there is no need to solve for $q(x,t)$ in the second-order problem (2.7)–(2.9). We shall seek approximations $\mathbf{U}^k$ and $\mathbf{V}^k$ from (3.16a) and (3.16c) respectively.

The work described in this Section is an extension of that found in Parker (2010), which considers the second-order problem (2.7)–(2.9) for the specific value of $n = 3$.

3.4.1 Approximating $v(x,t)$

Obtaining the velocity $v(x,t)$ at each node in the mesh is also slightly different in the second-order problem (2.7)–(2.9), since there exists an expression for the velocity for any value of n (Smyth and Hill (1988)).

From (3.14) and noting that $q = u$, the velocity (at all points where $u > 0$) is given by

$$v = -u^{n-1}\frac{\partial u}{\partial x},$$

which we may rewrite as

$$v = -\frac{1}{n}\frac{\partial(u^n)}{\partial x}. \tag{3.31}$$

Given U_i^k, $i = 1, \ldots, N$, (denoted U_i for ease of notation) we compute V_i^k, $i = 1, \ldots, N$, (denoted V_i for ease of notation) as follows. The velocity is approximated from (3.31) at interior points by (3.24) with Q_i replaced by U_i^n

$$V_i = -\frac{1}{n}\left(\frac{\dfrac{U_{i+1}^n - U_i^n}{(X_{i+1} - X_i)^2} + \dfrac{U_i^n - U_{i-1}^n}{(X_i - X_{i-1})^2}}{\dfrac{1}{X_{i+1} - X_i} + \dfrac{1}{X_i - X_{i-1}}}\right), \tag{3.32}$$

which is a linear interpolation through the points $(X_{i-1/2}, (U'_{i-1/2})^n)$ and $(X_{i+1/2}, (U'_{i+1/2})^n)$ and is of second order on the irregular grid.

The approximation (3.32) is used for interior points of the mesh, with v at the boundary approximated using a quadratic extrapolation as described in §3.3.4.

3.4.2 Mesh Movement and Solution Recovery

Updating the mesh and solution recovery is performed exactly as described in §3.3.5 and §3.3.6.

3.5 Implementation of the MPCM for the Sixth-Order Problem

The implementation of the MPCM for the sixth-order problem (2.10)–(2.12) is similar to that for the fourth-order implementation given in §3.3, with the added requirement of an approximation $\mathbf{P}^k$ to the $p(x,t)$ function.

We shall seek approximations $\mathbf{U}^k$, $\mathbf{Q}^k$ and $\mathbf{V}^k$ from (3.16a), (3.16b) and (3.16c) respectively, in addition to

$$\mathbf{P}^k := \{P_i^k\}, \text{ for } i = 1, \ldots, N. \tag{3.33}$$

3.5.1 Approximating $p(x,t)$

We seek an approximation to $p(x,t)$ at each node in the mesh. Given U_i^k, $i = 1, \ldots, N$, (denoted by U_i for ease of notation) we compute P_i^k, $i = 1, \ldots, N$, (denoted P_i for ease of notation) as follows.

On a variable mesh, a three-point central difference approximation P_i is

$$P_i = -\left(\frac{\frac{U_{i+1}-U_i}{X_{i+1}-X_i} - \frac{U_i-U_{i-1}}{X_i-X_{i-1}}}{\frac{1}{2}(X_{i+1}-X_{i-1})} \right), \tag{3.34}$$

which has a local truncation error, τ_i, that can be shown to be equal to

$$\tau_i = -\frac{(X_{i+1}-2X_i+X_{i-1})}{3}p_i' - \frac{1}{12}\frac{(X_{i+1}-X_i)^3+(X_i-X_{i-1})^3}{X_{i+1}-X_{i-1}}p_i''\Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$.

At the boundary points, we use second-degree Lagrange polynomials (as described in §3.3.3), giving the following equations for obtaining P_1 and P_N:

$$L_1(X_4)P_1 + L_2(X_4)P_2 + L_3(X_4)P_3 - P_4 = 0, \tag{3.35}$$

and

$$-P_{N-3} + L_{N-2}(X_{N-3})P_{N-2} + L_{N-1}(X_{N-3})P_{N-1} + L_N(X_{N-3})P_N = 0. \tag{3.36}$$

To obtain the values of $\mathbf{P}$ (from (3.33)) at a given time step we solve the matrix system

$$S\mathbf{P} = T\mathbf{U},$$

arising from (3.34), (3.35) and (3.36), where $\mathbf{P} = \{P_i\}$ $(i = 1, \ldots, N)$, $\mathbf{U} = \{U_i\}$ $(i = 1, \ldots, N$, including boundary values $U_1 = U_N = 0)$ and where the matrices S and T are as given in (3.21) and (3.22) respectively.

3.5.2 Approximating $q(x,t)$

Given the P_i, $i = 1, \ldots, N$, obtained from §3.5.1, we now seek Q_i^k, $i = 1, \ldots, N$, (denoted by Q_i for ease of notation) approximating $q(x,t)$ from (2.10b) on each node of the mesh.

On a variable mesh, a three-point central difference approximation Q_i at node i is

$$Q_i = -\left(\frac{\frac{P_{i+1}-P_i}{X_{i+1}-X_i} - \frac{P_i-P_{i-1}}{X_i-X_{i-1}}}{\frac{1}{2}(X_{i+1}-X_{i-1})} \right). \tag{3.37}$$

The local truncation error, τ_i, of (3.37) can be shown to be equal to

$$\tau_i = -\frac{(X_{i+1}-2X_i+X_{i-1})}{3} q_i' - \frac{1}{12} \frac{(X_{i+1}-X_i)^3 + (X_i-X_{i-1})^3}{X_{i+1}-X_{i-1}} q_i'' \Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$.

At the boundary points, we use second-degree Lagrange polynomials (as described in §3.3.3), giving the following equations for obtaining P_1 and P_N:

$$L_1(X_4)Q_1 + L_2(X_4)Q_2 + L_3(X_4)Q_3 - Q_4 = 0, \tag{3.38}$$

and

$$-Q_{N-3} + L_{N-2}(X_{N-3})Q_{N-2} + L_{N-1}(X_{N-3})Q_{N-1} + L_N(X_{N-3})Q_N = 0. \tag{3.39}$$

To obtain the values of $\mathbf{Q}$ at a given time step we solve the matrix system

$$S\mathbf{Q} = T\mathbf{P},$$

where $\mathbf{Q} = \{Q_i\}$ $(i = 1, \ldots, N)$, $\mathbf{U} = \{U_i\}$ $(i = 1, \ldots, N$, including boundary values $U_1 = U_N = 0)$ arising from (3.37), (3.38) and (3.39), where the matrices S and T are as given in (3.21) and (3.22) respectively.

3.5.3 Approximating $v(x,t)$

Given the P_i and Q_i, $i = 1, \ldots, N$, obtained from §3.5.1 and §3.5.2, we now seek V_i^k, $i = 1, \ldots, N$. This is achieved in the same manner as described for the fourth-order problem in §3.3.4.

3.5.4 Mesh Movement and Solution Recovery

Updating the mesh and solution recovery is performed exactly as described in §3.3.5 and §3.3.6.

3.6 The S Property

In the next chapter, we shall show that by modifying some of the steps in the MPCM (i.e. by modifying the appropriate schemes such that they are of the required order to match the exact solution for the relevant problem) and by use of a scale-invariant time stepping scheme, it is possible to propagate initial conditions coinciding with a similarity solution forward in time (over a single time step) to within rounding error. We shall define this property as the *S Property*:

Definition: The S Property

A numerical method is said to possess the *S Property* in some norm if, given a self-similar scaling solution to a time dependent problem, the norm of the error in the numerical solution after a single time step is zero. In other words the exact solution is preserved in that norm.

3.7 Summary of this Chapter

In this chapter we have provided a brief review of velocity-based moving-mesh methods, including some of the more common approaches seen in the literature. We then introduced the MPCM, which is a finite difference implementation of a conservation-based method. The method has been discussed in terms of the fourth-order problem detailed in §2.2, with further details on how the method is adapted for second and sixth-order problems provided.

In the next chapter we shall provide a discussion on how the MPCM as described in this chapter can be modified such that it then possesses the *S Property*. We shall detail the required modifications to the basic MPCM presented in this chapter as well as the other requirements for the MPCM (such as initial condition and time stepping method).

Chapter 4

An Implementation of the MPCM Possessing the S Property

4.1 Aims of this Chapter

In this chapter we shall demonstrate the ability of the MPCM described in Chapter 3 (with some modifications) to possess the *S Property* (described in §3.6) in the l^∞ norm. This ability shall be proven for the fourth-order problem (2.3)–(2.5) (with $n = 1$), with numerical results shown to validate the claim of the method possessing the *S Property*.

We shall also outline and provide numerical results which demonstrate the *S Property* for implementations of the MPCM in the second-order problem (2.7)–(2.9) (for any n) and the sixth-order problem (2.10)–(2.12) with $n = 1$.

4.2 The Fourth-Order Problem with $n = 1$

In this chapter we shall mainly focus on the fourth-order problem (2.3)–(2.5) with the specific choice of $n = 1$. We recall that for this choice of n, closed-form similarity solutions of the fourth-order PDE (2.3) exist. We shall demonstrate that if such a similarity solution is used to provide the initial approximation $\mathbf{U}^0$ at the nodes of the initial mesh, the MPCM given in §3.3 can be modified such that it possesses the *S*

Property (see §3.6) in the l^∞ norm.

Throughout this chapter we shall make use of the scaled time variable s rather than t, since a scale-invariant time stepping scheme will be used (see §4.3.3 for details on the scale-invariant time stepping scheme used).

We restate the problem being considered for clarity. We seek a solution $u(x,s)$ to the 1D fourth-order nonlinear diffusion equation with $n=1$,

$$\frac{\partial u}{\partial s} = \frac{\partial}{\partial x}\left(u\frac{\partial q}{\partial x}\right), \tag{4.1a}$$

$$q(x,s) = -\frac{\partial^2 u}{\partial x^2}, \tag{4.1b}$$

for $x \in \Omega_S$, subject to the initial condition at time $s = s^0$ given by

$$u(x,s^0) = u^0(x), \qquad \text{for } x \in \Omega(s^0),$$

and boundary conditions

$$\begin{aligned} u &= 0 \\ \frac{\partial u}{\partial x} &= 0 \\ uv + u\frac{\partial q}{\partial x} &= 0 \end{aligned}$$

at the moving boundaries $x = a(s)$ and $x = b(s)$, for $s > s^0$.

Similarity solutions $u(x,s)$ to the fourth-order problem (2.3)–(2.5) with $n=1$ exist which are quartic functions (Smyth and Hill (1988)), with the function $q(x,s)$ therefore a quadratic function.

4.3 The S Property in the MPCM for the Fourth-Order Problem

We shall now demonstrate that the implementation of the MPCM given in §3.3 can be modified in order to possess the *S Property* over a single time step for any initial mesh.

We are given an initial mesh $\mathbf{X}^0$ discretisating the domain $\Omega(s^0)$ and an approximation $\mathbf{U}^0$ to the initial condition $u^0(x)$ such that $u^0(x) = u^S(x, s^0)$ at each node in the mesh. We therefore have that

$$U_i^0 = u^S(X_i^0, s^0), \qquad \text{for } i = 1, \ldots, N.$$

4.3.1 Approximating $q(x, s)$

As noted in §3.3.3, we first seek an approximation $\mathbf{Q}^k$ at the nodes. The scheme given in §3.3.3 has a local truncation error proportional to $\frac{\partial q}{\partial x}$ and as such is not zero for quadratic $q(x, s)$, quartic $u(x, s)$. We therefore seek to modify the finite difference scheme used so that it is exact for quadratic $q(x, s)$, quartic $u(x, s)$. Given U_i^k, $i = 1, \ldots, N$, (denoted by U_i for ease of notation) we compute Q_i^k, $i = 1, \ldots, N$, (denoted by Q_i for ease of notation) as follows.

We obtain $\mathbf{Q}$ at interior points of the mesh using a modified three-point central-difference approximation of (4.1b) of the form

$$Q_i \approx -\frac{\frac{U_{i+1}-U_i}{X_{i+1}-X_i} - \frac{U_i-U_{i-1}}{X_i-X_{i-1}}}{\frac{1}{2}(X_{i+1}-X_{i-1})} + C_1\frac{Q_{i+1}-Q_{i-1}}{X_{i+1}-X_{i-1}} + C_2\frac{\frac{Q_{i+1}-Q_i}{X_{i+1}-X_i} - \frac{Q_i-Q_{i-1}}{X_i-X_{i-1}}}{\frac{1}{2}(X_{i+1}-X_{i-1})}, \tag{4.3}$$

where we choose C_1 and C_2 such that terms relating to the first and second derivatives of q in the local truncation error will vanish. The method will then be exact for quartic $u(x, s)$, quadratic $q(x, s)$.

The local truncation error, τ_i, of the modified approximation (4.3) can be shown to be equal to

$$\begin{aligned}\tau_i = q_i + u_i'' &- \left(\frac{1}{3}(X_{i+1} - 2X_i - X_{i-1}) + C_1\right) q_i' \\ &- \left[\frac{1}{12}\left(\frac{(X_{i+1}-X_i)^3 + (X_i - X_{i-1})^3}{X_{i+1}-X_{i-1}}\right) + \frac{1}{2}(X_{i+1} - 2X_i - X_{i-1})C_1 + C_2\right] q_i''\Big|_\theta,\end{aligned} \tag{4.4}$$

for some $\theta \in (X_{i-1}, X_{i+1})$, where $q_i = q(X_i)$, $q_i' = \frac{\partial q}{\partial x}\Big|_{X_i}$, $q_i'' = \frac{\partial^2 q}{\partial x^2}\Big|_{X_i}$.

From equation (4.4) we can see that using $q = -u''$ at X_i and choosing

$$C_1 = -\frac{(X_{i+1} - 2X_i - X_{i-1})}{3}, \tag{4.5}$$

removes all derivatives up to q_i' from the local truncation error, which then becomes equal to

$$\tau_i = -\left(\frac{C_1}{2}(X_{i+1} - 2X_i - X_{i-1}) + C_2 + \frac{1}{12}\frac{(X_{i+1} - X_i)^3 + (X_i - X_{i-1})^3}{X_{i+1} - X_{i-1}}\right) q_i''\Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$.

Therefore, choosing

$$\begin{aligned} C_2 &= -\frac{C_1}{2}(X_{i+1} - 2X_i - X_{i-1}) - \frac{1}{12}\frac{(X_{i+1} - X_i)^3 + (X_i - X_{i-1})^3}{X_{i+1} - X_{i-1}}, \\ &= \frac{(X_{i+1} - 2X_i - X_{i-1})^2}{6} - \frac{1}{12}\frac{(X_{i+1} - X_i)^3 + (X_i - X_{i-1})^3}{X_{i+1} - X_{i-1}}, \\ &= \frac{(X_{i+1} - X_i)^2 - 3(X_{i+1} - X_i)(X_i - X_{i-1}) + (X_i - X_{i-1})^2}{12}, \end{aligned} \tag{4.6}$$

removes all q_i'' terms from the local truncation error, which then vanishes for quadratic $q(x, s)$.

We can therefore use equation (4.3) to construct a system of equations, with the values of C_1 and C_2 from equations (4.5) and (4.6) respectively. Once solved, the system will produce an approximation $\mathbf{Q}$ at each interior point of the mesh having zero local truncation error for quartic $u(x, s)$, quadratic $q(x, s)$.

At the boundary points we use the Lagrange interpolating polynomials outlined in §3.3.3. The details are omitted here, but we note that since second-degree Lagrange polynomials are used the approximation will be exact for quadratic $q(x, s)$.

To obtain the values of $\mathbf{Q}$ at each of the nodal positions we therefore solve the matrix system

$$S\mathbf{Q} = T\mathbf{U},$$

where $\mathbf{Q} = \{Q_i\}$ $(i = 1, \ldots, N)$, $\mathbf{U} = \{U_i\}$ $(i = 1, \ldots, N$, including $U_1 = U_N = 0)$ and S, T are sparse matrices of size $N \times N$. Rows $2, \ldots, N-1$ of these matrices come from equation (4.3), while the first and final rows are given by equations (3.18) and (3.19) respectively.

By combining these equations, we see that the matrix S is of the form

$$S = \begin{pmatrix} L_1(X_4) & L_2(X_4) & L_3(X_4) & -1 & 0 & \dots & 0 \\ S_2^- & S_2 & S_2^+ & 0 & & & \vdots \\ 0 & S_3^- & S_3 & S_3^+ & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & 0 & S_{N-2}^- & S_{N-2} & S_{N-2}^+ & 0 \\ \vdots & & & 0 & S_{N-1}^- & S_{N-1} & S_{N-1}^+ \\ 0 & \dots & 0 & -1 & L_{N-2}(X_{N-3}) & L_{N-1}(X_{N-3}) & L_N(X_{N-3}) \end{pmatrix}, \tag{4.7}$$

where for $i = 2, \dots, N-1$

$$\begin{aligned} S_i^- &= \left(\frac{C_1}{X_{i+1} - X_{i-1}} - \frac{C_2}{\frac{1}{2}(X_i - X_{i-1})(X_{i+1} - X_{i-1})} \right), \\ S_i &= \left(1 + \frac{C_2}{\frac{1}{2}(X_{i+1} - X_{i-1})} \left(\frac{1}{X_{i+1} - X_i} + \frac{1}{X_i - X_{i-1}} \right) \right), \\ S_i^+ &= \left(-\frac{C_1}{X_{i+1} - X_{i-1}} - \frac{C_2}{\frac{1}{2}(X_{i+1} - X_i)(X_{i+1} - X_{i-1})} \right). \end{aligned}$$

The matrix T is of the form

$$T = \begin{pmatrix} 0 & \dots & & & & \dots & 0 \\ T_2^- & T_2 & T_2^+ & 0 & & & \vdots \\ 0 & T_3^- & T_3 & T_3^+ & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & 0 & T_{N-2}^- & T_{N-2} & T_{N-2}^+ & 0 \\ \vdots & & & 0 & T_{N-1}^- & T_{N-1} & T_{N-1}^+ \\ 0 & \dots & & & & \dots & 0 \end{pmatrix}, \tag{4.9}$$

where for $i = 2, \dots, N-1$

$$\begin{aligned} T_i^- &= -\frac{1}{\frac{1}{2}(X_i - X_{i-1})(X_{i+1} - X_{i-1})}, \\ T_i &= \frac{1}{\frac{1}{2}(X_{i+1} - X_{i-1})} \left(\frac{1}{X_{i+1} - X_i} + \frac{1}{X_i - X_{i-1}} \right), \\ T_i^+ &= -\frac{1}{\frac{1}{2}(X_{i+1} - X_i)(X_{i+1} - X_{i-1})}. \end{aligned}$$

The matrix S is pentadiagonal due to the first and last rows, with the remaining rows tridiagonal. Through the application of two Gaussian elimination steps the whole system becomes tridiagonal.

4.3.2 The Nodal Velocity

The nodal velocities $\mathbf{V}$ are found using the steps detailed in §3.3.4 with no modifications. We repeat the schemes here for completeness.

In the case where the initial condition corresponds to a similarity solution we note that since the similarity solution for (4.1a) is a quartic, $q(x,s)$ will be quadratic and hence from (3.14) the velocity $v(x,s)$ will be linear.

Given U_i^k and Q_i^k, both $i = 1,\ldots,N$, (denoted by U_i and Q_i respectively for ease of notation), we compute V_i^k, $i = 1,\ldots,N$, (denoted by V_i for ease of notation) as follows.

The nodal velocities V_i at interior points of the mesh (i.e. for $i = 2,\ldots,N-1$) are calculated using (3.24), i.e.

$$V_i = -\left(\frac{\dfrac{(Q_{i+1}-Q_i)}{(X_{i+1}-X_i)^2} + \dfrac{(Q_i-Q_{i-1})}{(X_i-X_{i-1})^2}}{\dfrac{1}{X_{i+1}-X_i} + \dfrac{1}{X_i-X_{i-1}}} \right). \tag{4.11}$$

The local truncation error of the formula (4.11) at node i, denoted by $\tilde{\tau}_i$, can be shown to be equal to

$$\tilde{\tau}_i = \frac{(X_{i+1}-X_i)(X_i-X_{i-1})}{6} q_i'''\Big|_\theta,$$

for some $\theta \in (X_{i-1}, X_{i+1})$, where $q_i''' = \left.\frac{\partial^3 q}{\partial x^3}\right|_{X_i}$.

In the case where $q(x,s)$ is a quadratic function, the third derivative of $q(x,s)$ and hence the local truncation error will be equal to zero and so the velocity will be approximated by (4.11) to within rounding error of the exact value.

At the boundary points, the velocity is approximated using a quadratic extrapolation from the three closest interior points to the boundary. Such an extrapolation is exact for a linear $v(x,s)$.

4.3.3 Updating the Nodal Positions

We noted in §3.3.5 that the explicit Forward Euler time stepping scheme was not scale-invariant. It is possible to make the time stepping scheme scale invariant however, through the introduction of a scaled time step with a new time variable s, where $s = t^\beta$ (Budd and Piggott (2001), Budd et al. (2001), Baines et al. (2006)). Writing (3.15) in terms of the known function $v(x,t)$,

$$\frac{\mathrm{d}x}{\mathrm{d}s} = \frac{\mathrm{d}x}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}s} = \frac{\mathrm{d}x}{\mathrm{d}t}\frac{1}{\beta t^{\beta-1}} = \frac{1}{\beta}s^{1/\beta-1}v(x,t), \tag{4.12}$$

the explicit Euler time stepping scheme for (4.12),

$$X_i^{k+1} = X_i^k + \frac{1}{\beta}(s^k)^{1/\beta-1}\Delta s V_i^k, \tag{4.13}$$

is then scale invariant with constant Δs. We shall refer to (4.13) throughout this thesis as a scale invariant time stepping scheme.

For the fourth-order problem (2.3)–(2.5) with $n = 1$ we have $\beta = 1/5$ from (2.21) (since we have made the specific choices of $m = n = 1$). The scale invariant time stepping scheme (4.13) is then,

$$\mathbf{X}^{k+1} = \mathbf{X}^k + 5(s^k)^4\Delta s\mathbf{V}^k, \tag{4.14}$$

The LTE incurred by the time stepping scheme (4.14) at node i, denoted by $\hat{\tau}_i$, can be shown to be equal to

$$\begin{aligned}
\hat{\tau}_i &= \frac{x_i(s+\Delta s) - x_i(s)}{\Delta s} - \frac{\mathrm{d}x_i}{\mathrm{d}s}, \\
&= \frac{x_i + \Delta s\frac{\mathrm{d}x_i}{\mathrm{d}s} + \frac{1}{2}\Delta s^2\frac{\mathrm{d}^2x_i}{\mathrm{d}s^2} + \ldots - x_i}{\Delta s} - 5s^4v_i, \\
&= \frac{\Delta s}{2}\frac{\mathrm{d}^2x}{\mathrm{d}s^2}\bigg|_{s^*}
\end{aligned}$$

for some $s^* \in (s, s+\Delta s)$.

In terms of s, the similarity variable ξ in equation (2.22) is related to x by

$$\xi = \frac{x}{s},$$

from which we have, at any fixed time,

$$\frac{\mathrm{d}^2 x}{\mathrm{d}s^2} = 0. \tag{4.16}$$

From (4.14) and (4.16), we see that in the case where a scale invariant time stepping scheme is used, the LTE relating to the time discretisation of the mesh movement is equal to zero. This implies that the mesh points are updated to within rounding error of their exact values in one time step.

4.3.4 Updating the Solution Values

The solution approximation $\mathbf{U}^{k+1}$ is found from the updated nodal positions as described in §3.3.6, using (3.30):

$$U_i^{k+1} = \frac{X_{i+1}^0 - X_{i-1}^0}{X_{i+1}^{k+1} - X_{i-1}^{k+1}} U_i^0,$$

with $U_1^{k+1} = U_N^{k+1} = 0$ from the boundary conditions.

We now confirm that if the error in $\mathbf{X}$ is zero in a single time step according to (4.14), then (3.30) is able to recover the similarity solution to within rounding error at each node in the mesh in the same time step. We shall therefore confirm that

$$U_i^{k+1} - u^S(X_i^{k+1}, s^{k+1}) = 0, \tag{4.17}$$

for $i = 2, \ldots, N-1$.

From (4.20), the similarity solution $u^S(x,s)$ can be written in the general form

$$u^S(x,s) = \frac{C}{s}\left[D^2 - \left(\frac{x}{s}\right)^2\right]_+^2, \tag{4.18}$$

where C and D are constants. In terms of the similarity variables $\xi = \frac{x}{s}$ and $\eta = us$ from (2.22), (4.18) is written as

$$\eta = C\left[D^2 - \xi^2\right]_+^2,$$

We write (4.17) in terms of ξ and η as

$$\begin{aligned} U_i^{k+1} - u^S(X_i^{k+1}, s^{k+1}) &= \frac{(\xi_{i+1}^0 - \xi_{i-1}^0)}{(\xi_{i+1}^{k+1} - \xi_{i-1}^{k+1})} \frac{\eta_i^0}{s^{k+1}} - \frac{\eta_i^{k+1}}{s^{k+1}}, \\ &= \frac{(\xi_{i+1}^0 - \xi_{i-1}^0)}{(\xi_{i+1}^{k+1} - \xi_{i-1}^{k+1})} \left(\frac{C}{s^{k+1}} \left[D^2 - (\xi_i^0)^2 \right]_+^2 \right) - \frac{C}{s^{k+1}} \left[D^2 - (\xi_i^{k+1})^2 \right]_+^2, \\ &= \frac{C}{s^{k+1}} \left(\frac{(\xi_{i+1}^0 - \xi_{i-1}^0)}{(\xi_{i+1}^{k+1} - \xi_{i-1}^{k+1})} \left[D^2 - (\xi_i^0)^2 \right]_+^2 - \left[D^2 - (\xi_i^{k+1})^2 \right]_+^2 \right). \end{aligned} \tag{4.19}$$

Since ξ and η are invariant, they are independent of time and therefore independent of k. From this invariance, $\xi_i^0 = \xi_i^{k+1}$, for $i = 2, \ldots, N-1$, provided that the mesh points (and hence ξ_i values) have been moved exactly. The right hand side of (4.19) is equal to zero and hence the error at node i is equal to zero. The solution is therefore recovered exactly on each node of the mesh in the case where the error in the mesh positions is equal to zero.

4.3.5 Implications

We have shown that in the case where the initial condition coincides with a similarity solution at the nodes of an initial mesh, the modified MPCM implemented in this section will approximate $q(x,s)$ and $v(x,s)$ with zero local truncation error. These approximations are then equal to the exact values to within rounding error at the nodes over a single time step. The local truncation error incurred by the method in moving mesh points is also equal to zero, and hence mesh points are moved to within rounding error, if a scale-invariant time stepping scheme is used. The approximation to $u(x,s)$ is then also recovered exactly since the mesh points are moved exactly.

We therefore argue that the modified MPCM described here possesses the *S Property* in the l^∞ norm (although in practice rounding errors will be present in all approximations).

4.3.6 Numerical Results

We now present numerical results for the implementation of the MPCM given in §4.3. We shall demonstrate that over a single time step the method is able to propagate a similarity solution to within rounding error in the l^∞ norm. We then present results over a larger time window consisting of multiple time steps in order to illustrate how the MPCM performs over multiple steps.

Our choice of the l^∞ norm for the numerical results in this section stems from the point-wise nature of the finite difference method. If our claim that the MPCM possesses the *S Property* is true, then we would expect the absolute error in the approximation at each mesh point to be within rounding error. This would then equate to an error in the l^∞ norm in the region of rounding error. We shall also make use of a small number of mesh points in our numerical experiments to demonstrate that the method is accurate on a coarse mesh.

In order to demonstrate the behaviour of the MPCM from §4.3, we make use of the similarity solution (2.28) given in Barrett et al. (1998)

$$u^S(x,s) = \frac{1}{120s}\left[4 - \frac{x^2}{s^2}\right]_+^2, \tag{4.20}$$

with the scaled time variable s replacing the time variable t^β used in that paper. This similarity solution will be used to set the initial approximation $\mathbf{U}^0$ and also as a means of testing the accuracy of the MPCM.

Using (4.20), we can calculate the similarity functions $q^S(x,s)$ and $v^S(x,s)$, using (4.1b) and (3.14) respectively:

$$q^S(x,s) = \frac{2}{15s^3} - \frac{x^2}{10s^5}, \qquad v^S(x,t) = \frac{x}{5s^5}. \tag{4.21}$$

Single Time Step

We first run the implementation of the MPCM for a single time step in order to demonstrate that a similarity solution can be propagated within rounding error.

We run the method on an initially equally spaced mesh of $N = 21$ nodes. We use the similarity solution (4.20) to determine the initial approximation $\mathbf{U}^0$ at the initial

time $s^0 = 1$, such that

$$U_i^0 = \frac{1}{120}\left[4 - (X_i^0)^2\right]_+^2,$$

for $i = 1, \ldots, N$. This initial approximation is centred on $x = 0$ with initial boundary positions at $x = \pm 2$.

A single time step of size $\Delta s = 2.5 \times 10^{-5}$ is performed using the scale-invariant Forward Euler scheme (4.13).

After the single time step has been performed, we calculate the absolute error of the approximations $\mathbf{U}^1$, $\mathbf{Q}^0$ and $\mathbf{V}^0$, as well as the absolute error in the position of the updated mesh nodes $\mathbf{X}^1$. The error in the mesh positions is calculated using the similarity variable ξ from (2.22), using the initial values

$$\xi = \frac{x(s^k)}{s^k} \qquad \forall k,$$
$$\Rightarrow \quad |\mathbf{X}^k - x(s^k)| = \left|\mathbf{X}^k - s^k \frac{x^0}{s^0}\right|, \qquad \forall k.$$

Figure 4.1 details the absolute errors of these approximations. We see that the approximations are in the region of rounding error, with the error in $\mathbf{U}^1$ being of $O(10^{-17})$, the error in $\mathbf{Q}^0$ is of $O(10^{-15})$ and the error in $\mathbf{V}^0$ is $O(10^{-14})$. Finally, we observe the absolute error in the mesh positions is of $O(10^{-16})$.

Multiple Time Steps

We then run the implementation of the MPCM over multiple time steps in order to determine the accuracy of the method for more than one time step.

The MPCM is run on an initially equally-spaced mesh of $N = 21$ nodes over $60,000$ time steps using the scale-invariant Forward Euler time stepping scheme (4.13). The time step Δs is chosen such that $\Delta s = O(1/N^4)$, specifically $\Delta s = 2.5 \times 10^{-5}$. The value of Δs is chosen such that $\Delta s = O(1/N^4)$ in order to reduce the risk of node tangling occurring (see §3.3.7). Using an initial time of $s^0 = 1$ this relates to a time window of $s \in (1, 2.5)$.

The similarity solution (4.20) is used to determine the initial approximation $\mathbf{U}^0$ and will be used in determining the accuracy of the method. We shall seek to determine the

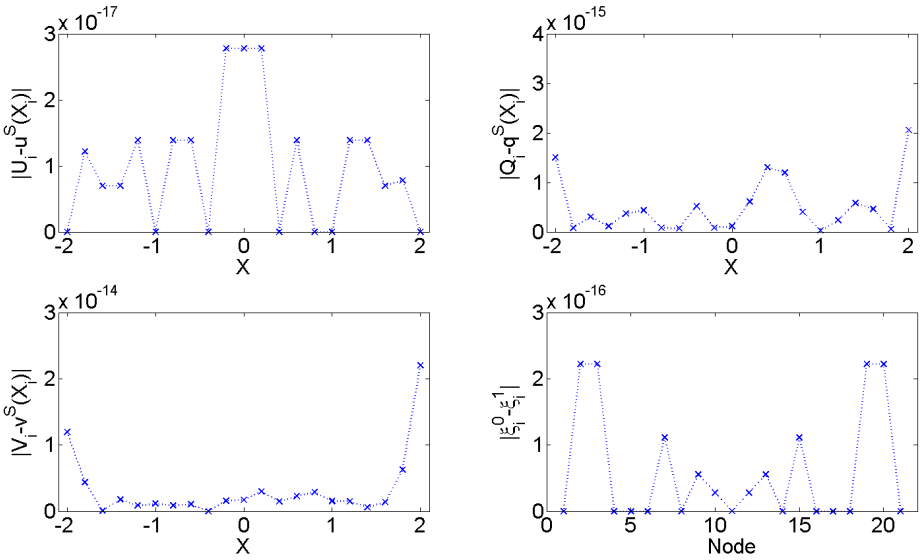

Figure 4.1: Absolute error in the MPCM over a single time step. The top left window contains the absolute error in $\mathbf{U}^1$ ($O(10^{-17})$), the top right window the error in $\mathbf{Q}^0$ ($O(10^{-15})$), bottom left the error in $\mathbf{V}^0$ ($O(10^{-14})$) and bottom right the error in the mesh positions ($O(10^{-16})$).

error in the approximations and the boundary nodes.

At the end of each time step, we determine the accuracy of the method by calculating the l^∞ norm of the error, given by (in the case of the error in the solution $\mathbf{U}^k$)

$$E_N^U = \max_{i=1,\ldots,N} |U_i^k - u^S(X_i^k, s^k)| \qquad \text{for } k = 1, \ldots, K,$$

where K is the total number of time steps performed. Similar quantities for the error in $\mathbf{Q}^k$ (denoted by E_N^Q) and $\mathbf{V}^k$ (denoted by E_N^V) are calculated, using (4.21) for the exact values of q^S and v^S.

The error in the boundary node positions is denoted by E_N^B (for the right-hand boundary node) and is given by

$$E_N^B = \frac{|X_N^k - b(s^k)|}{|b(s^k)|},$$

where $b(s^k)$ denotes the exact position of the right-hand boundary at time s^k and from (4.20) is given by $b(s) = 2s$.

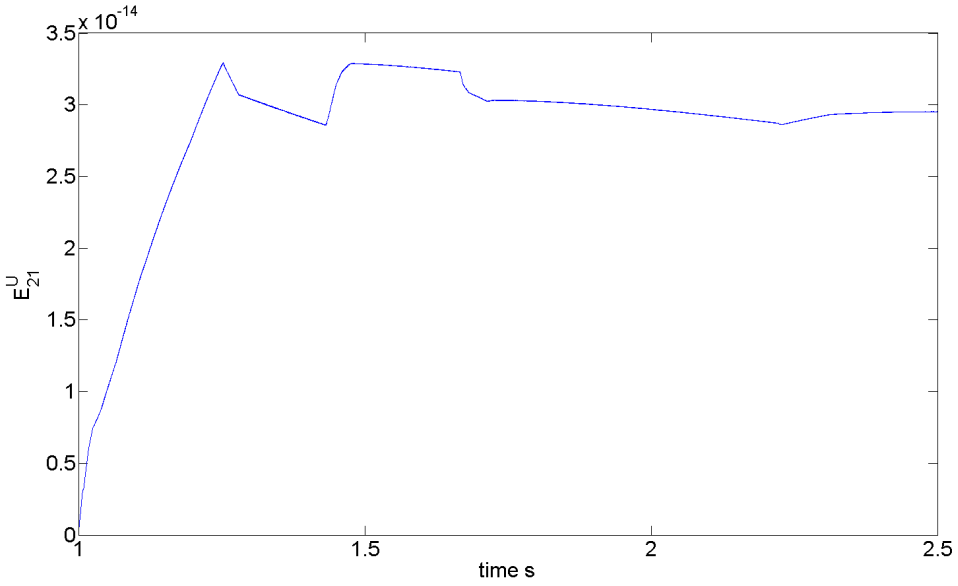

Figure 4.2: l^∞ error in the approximation $\mathbf{U}^k$ from the MPCM over the time window $s \in [1, 2.5]$. Scale of y-axis is $O(10^{-14})$.

Figure 4.2 shows the error E_{21}^U in the approximation $\mathbf{U}^k$ for this run of the MPCM.

We see that the error increases at the start of the time window, reaching a peak at time $s \approx 1.25$ which is approximately 3.25×10^{-14}. The error then remains of this magnitude throughout the remainder of the time window, with slight fluctuations observed.

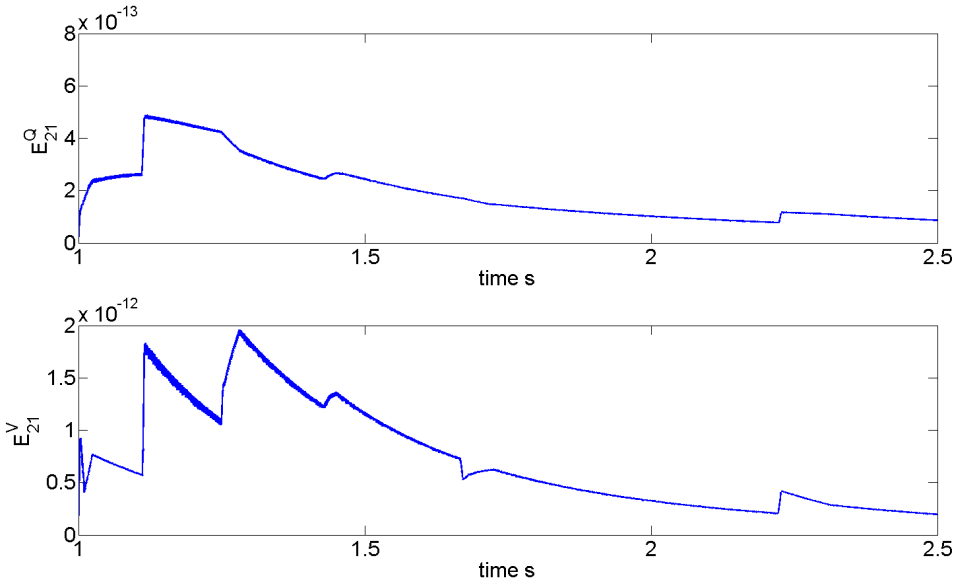

Figure 4.3: l^∞ error in $\mathbf{Q}^k$ (top) and $\mathbf{V}^k$ (bottom) over the time window $s \in [1, 2.5]$. Vertical scale of the top plot is $O(10^{-13})$, while the vertical scale for the bottom plot is $O(10^{-12})$

The errors E_{21}^Q and E_{21}^V for this run are shown in Figure 4.3. In these cases the error appears to be much more random, with an increase in the error at the beginning of the time window which has fallen by the end of the window. This suggests that the approximations $\mathbf{Q}^k$ and $\mathbf{V}^k$ are being calculated to within rounding error in each time step.

Figure 4.4 shows the error in the right-hand boundary position E_{21}^B for this run of the MPCM. We observe that the error increases throughout the time window, but initially remains very small, with the increase only visible at $s \approx 1.1$. The difference between the computed and exact value of the boundary is therefore increasing as time progresses, but still remains within 10^{-11} at the end of the time window. The maximum difference

between two successive time steps is approximately 2×10^{-15}, which suggests that the *S Property* is being upheld in each individual time step.

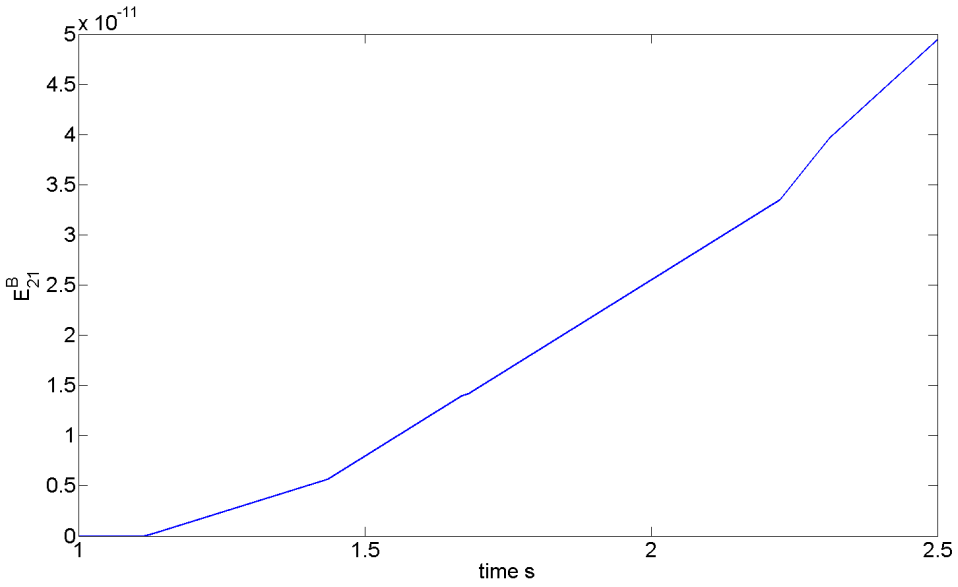

Figure 4.4: Boundary node error E_{21}^B over the time window $s \in [1, 2.5]$. Vertical scale of plot is $O(10^{-11})$.

The errors observed in Figures 4.2–4.4 are a result of the large number of steps being taken (60,000 in the results from this subsection). The difference in errors between successive time steps can be attributed to rounding error, which indicates that a propagation of rounding error is taking place over multiple time steps.

In the case where we are propagating an initial condition coinciding with a similarity solution, the results displayed in this section demonstrate that a single time step of the required size is more beneficial than running several smaller time steps and is in fact a major advantage of the MPCM. Not only is the risk of node tangling eliminated, but the build up of rounding error highlighted in Figures 4.2–4.4 is also avoided.

We shall now attempt to explain the source of this accumulation of rounding error in the MPCM.

4.4 Propagation of Rounding Error

We observed in §4.3.6 that the l^∞ error in the solution does not always remain in the region of rounding error, despite each individual time step of the MPCM arguably possessing the *S Property*. The reason for this appears to be a propagation of rounding error in the time step of the method and is discussed in this section.

4.4.1 Explaining the Additional Multiple Time Step Error

We shall repeat the experiment conducted in §4.3.6, with some alterations. At each time step:

- The approximation of $\mathbf{Q}^k$ described in §4.3.1 is replaced by setting

$$Q_i^k = \frac{2}{15(s^k)^3} - \frac{(X_i^k)^2}{10(s^k)^5}, \qquad \text{for } i = 1, \ldots, N.$$

  i.e. setting Q_i^k to be the exact value for all time.

- The nodal velocity is also set equal to the exact value,

$$V_i^k = \frac{X_i^k}{5(s^k)^5}, \qquad \text{for } i = 1, \ldots, N.$$

- Instead of using the scale-invariant time stepping scheme (4.14) to update the nodal positions, we set

$$X_i^{k+1} = s^{k+1}\frac{X_i^k}{s^k}, \qquad \text{for } i = 1, \ldots, N,$$

  which is the exact position of the nodes at time s^{k+1} using the similarity variable ξ from (2.22).

- The recovered solution on the updated mesh is then set equal to the similarity solution at time s^{k+1}:

$$U_i^{k+1} = \frac{1}{120s^{k+1}}\left[4 - \left(\frac{X_i^{k+1}}{s^{k+1}}\right)^2\right]_+^2, \qquad \text{for } i = 2, \ldots, N-1,$$

  with $U_1^{k+1} = U_N^{k+1} = 0$ to strongly enforce the $u = 0$ boundary condition.

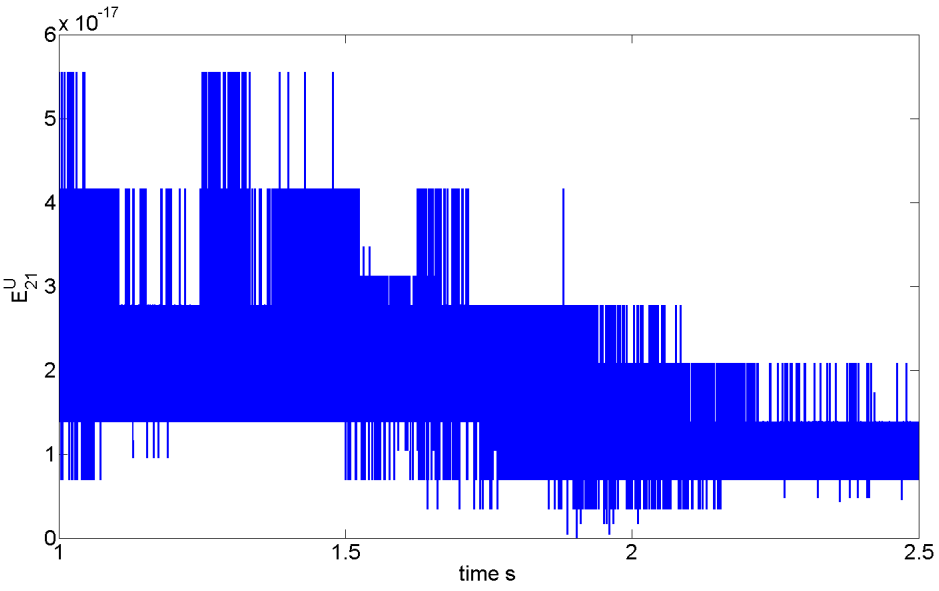

Figure 4.5: l^∞ error in the solution when $\mathbf{Q}$, $\mathbf{V}$ and $\mathbf{X}$ in the method are set equal to the exact value, but the solution is recovered approximately. Scale of y-axis is 10^{-17}.

Performing these alterations produces zero error over the whole time window (not shown), since no quantity is being approximated in the method.

We then repeat the experiment, but recover the solution using the approximation (3.30) while setting Q_i^k, V_i^k, X_i^k equal to the exact values. This solution recovery is expected to be within rounding error of the similarity solution (4.20). Figure 4.5 shows that this is the case, with E_{21}^U remaining proportional to 10^{-17} throughout the time window. Note that in Figure 4.5 the large number of time steps taken has forced the noise-like structure of the errors to be compressed and hence less distinguishable.

Finally, we repeat the experiment one further time. In this instance we update the nodal positions using the time stepping scheme (4.14) and recover the solution using the approximation (3.30), while keeping all other functions equal to their exact values. Figure 4.6 shows the value of E_{21}^U for this experiment. We see that in this case the error increases throughout the time window and has reached a larger value than observed in Figure 4.2 by one order of magnitude. Since the only change made in the method

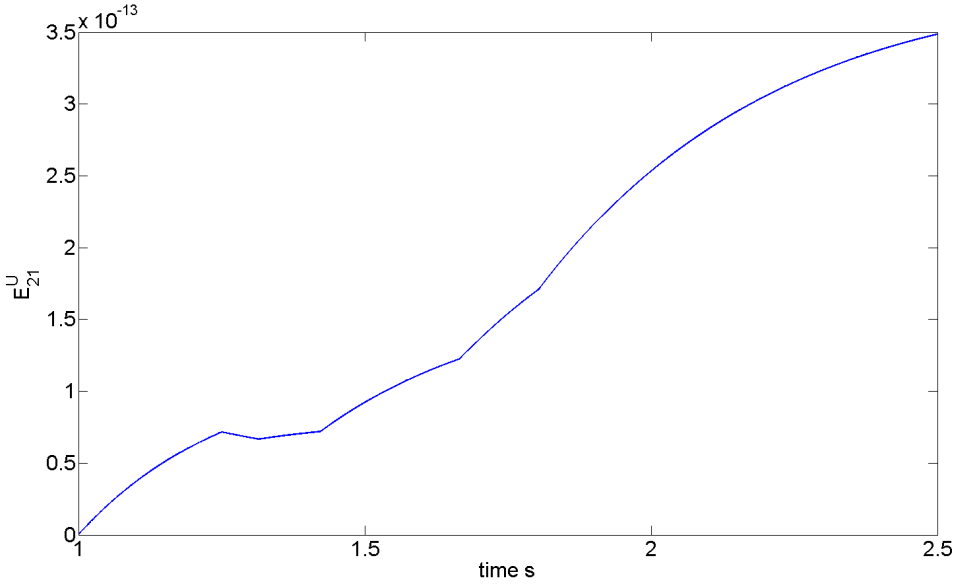

Figure 4.6: l^∞ error in the solution when $\mathbf{Q}$ and $\mathbf{V}$ in the method are set equal to the exact value, but the nodes are updated using a scale-invariant time stepping scheme and the solution is recovered approximately. Scale of y-axis is 10^{-13}.

between this and the previous experiment was to perform time stepping of the nodal positions, we conclude that this is the step responsible for the increase of error in the solution.

Interestingly, the original experiment (in which every function was approximated) performs more accurately than this last experiment. This implies that the process of approximating $q(x,s)$ and $v(x,s)$ may help to 'correct' some of the error caused by the time stepping scheme.

4.4.2 Examining the Time Stepping Scheme

The scale invariant time stepping scheme (4.13) is

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \Delta s \widehat{\boldsymbol{V}}_k,$$

where Δs is the size of the time step using the scaled time variable s and $\widehat{\boldsymbol{V}}_k$ is the mesh velocity in terms of the s time variable. The subscript k denotes the time level (written as a subscript as opposed to the previous superscript time level notation).

In the case where the similarity solution is known, the exact value of the nodes is

$$\mathbf{x}_{k+1}^S = \mathbf{x}_k^S + \Delta s \widehat{\boldsymbol{v}}_k^S, \tag{4.23}$$

where the similarity velocity $\widehat{\boldsymbol{v}}_k^S$ is

$$\widehat{\boldsymbol{v}}_k^S = \frac{\beta \mathbf{x}_k^S}{s_k^{1/\beta}}. \tag{4.24}$$

Substituting (4.24) into (4.23) gives

$$\mathbf{x}_{k+1}^S = \left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right) \mathbf{x}_k^S.$$

Hence the global error, which we denote by ϵ_k^S, is defined by

$$\begin{aligned} \epsilon_k^S &= x_k^S - X_k^S, \\ \epsilon_{k+1}^S &= \left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right) \epsilon_k^S, \\ \Rightarrow \qquad \epsilon_p^S &= \prod_{k=0}^{p-1} \left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right) \epsilon_0^S, \end{aligned} \tag{4.25}$$

at a particular time step p.

We see that there is an amplification factor multiplying ϵ_0^S of

$$\left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right),$$

which we know is always greater than 1 since all quantities are positive.

We know that ϵ_0^S should equal zero, so $\epsilon_p^S = 0$ for all p. In a computational sense however, we will have ϵ_0^S in the region of rounding error ($O(10^{-16})$, say) and we see that this global error ϵ_p^S is magnified over time.

We now seek to obtain an upper bound for ϵ_p^S in order to demonstrate that this error will not grow unboundedly over time.

4.4.3 Bounding the Error

We begin by noting that

$$\left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right) \leq \exp\left(\frac{\beta \Delta s}{s_k^{1/\beta}}\right),$$

for all k, from which we may rewrite (4.25) as

$$\begin{aligned}
\epsilon_p^S &= \prod_{k=0}^{p-1} \left(1 + \frac{\beta \Delta s}{s_k^{1/\beta}}\right) \epsilon_0^S, \\
&\leq \prod_{k=0}^{p-1} \exp\left(\frac{\beta \Delta s}{s_k^{1/\beta}}\right) \epsilon_0^S, \\
&= \exp\left[\sum_{k=0}^{p-1} \left(\frac{\beta \Delta s}{s_k^{1/\beta}}\right)\right] \epsilon_0^S,
\end{aligned}$$

We now take $\beta = 1/5$ and write $s_k = s_0 + k\Delta s$,

$$\epsilon_p^S \leq \exp\left[\frac{\Delta s}{5} \sum_{k=0}^{p-1} \left(\frac{1}{(s_0 + k\Delta s)^5}\right)\right] \epsilon_0^S. \tag{4.27}$$

Noting that the function $s_0 + x\Delta s$ is an increasing function of x, we can bound the summand $\frac{1}{(s_0+k\Delta s)^5}$ using

$$\frac{1}{(s_0 + k\Delta s)^5} \leq \int_{k-1}^{k} \frac{1}{(s_0 + x\Delta s)^5} \, \mathrm{d}x,$$

which implies that

$$\sum_{k=1}^{p-1} \frac{1}{(s_0 + k\Delta s)^5} \leq \int_0^{p-1} \frac{1}{(s_0 + x\Delta s)^5}\,\mathrm{d}x,$$
$$= \frac{1}{4\Delta s}\left[\frac{1}{s_0^4} - \frac{1}{(s_0 + (p-1)\Delta s)^4}\right].$$

Therefore,

$$\frac{\Delta s}{5}\sum_{k=0}^{p-1} \frac{1}{(s_0 + k\Delta s)^5} \leq \frac{\Delta s}{5s_0^5} + \frac{1}{20}\left[\frac{1}{s_0^4} - \frac{1}{s_{p-1}^4}\right] \tag{4.29}$$

Using (4.27) and (4.29) we see that

$$\begin{aligned} \epsilon_p^S &\leq \exp\left(\frac{\Delta s}{5s_0^5} + \frac{1}{20}\left[\frac{1}{s_0^4} - \frac{1}{s_{p-1}^4}\right]\right)\epsilon_0^S, \\ &\leq \exp\left(\frac{\Delta s}{5s_0^5} + \frac{1}{20s_0^4}\right)\epsilon_0^S, \end{aligned} \tag{4.30}$$

since $s_{p-1}^{-4} > 0$.

The bound is dependent upon the values of s_0 and Δs (both fixed). Let us consider the values used to generate the numerical results presented in §4.3.6. In the numerical results run over multiple time steps, values of $s_0 = 1$ and $\Delta s \approx 2.5 \times 10^{-5}$ were used. The bound (4.30) is then

$$\begin{aligned} \epsilon_p^S &\leq \exp\left(\frac{1.6 \times 10^{-6}}{5} + \frac{1}{20}\right)\epsilon_0^S, \\ &\leq 1.0513\epsilon_0^S. \end{aligned}$$

If ϵ_0^S is in the region of rounding error, then the bound indicates that the error at the pth time step will be of the same magnitude. If Δs is increased to $\Delta s = 10^{-3}$, (4.30) is still approximately equal to $1.05\epsilon_0^S$. These results indicate that there is an additional source of error contributing to the build up of error in the MPCM, beyond that which can be explained by the arguments in this section. This additional error likely appears from the previous steps of the method (those arising from the approximation of q and v), which have not been taken into account in this error bound.

4.5 The S Property in the MPCM for the Second-Order Problem

We shall now demonstrate that the implementation of the MPCM for the second-order problem (2.7)–(2.9) possesses the *S Property* (see §3.6) in the l^∞ norm for **any** value of n.

We refer to §3.4 for details of how the MPCM is implemented for the second-order problem (2.7)–(2.9), with no modifications to the implementation necessary to be able to propagate similarity solutions to within rounding error at the nodes over a single time step, other than the use of the scale-invariant time stepping scheme (4.13) in place of the explicit Forward Euler scheme (3.26).

Similarity solutions to the second-order problem (2.7)–(2.9) are of degree $2/n$ (Smyth and Hill (1988)). The velocity $v(x,s)$ given by (3.31) is therefore a linear function.

The local truncation error of the velocity approximation at node i of the mesh, τ_i, is

$$\tau_i = \frac{(X_{i+1}-X_i)(X_i-X_{i-1})}{6n}\left.\frac{\partial^3 u^n}{\partial x^3}\right|_\theta,$$

for some $\theta \in (X_{i+1}, X_{i-1})$, which is proportional to $\frac{\partial^3 u^n}{\partial x^3}$ and hence equal to zero in the similarity solution case. We would therefore expect the approximation $\mathbf{V}^k$ of the velocity calculated from (3.32) to be within rounding error of the similarity velocity at the nodes over a single time step.

Updating the mesh is performed exactly as described in §3.3.5, using the scale invariant time stepping scheme (4.13). This scheme has the same truncation error properties shown for the fourth-order problem in §4.3.3 and so we would expect the mesh points to be moved to within rounding error of the exact values in a single time step.

The solution is recovered on the updated mesh using (3.30), just as for the fourth-order problem in §4.3.4.

As in the fourth-order problem (2.3)–(2.5) we are able to show that the solution is recovered exactly on the updated mesh, provided that the mesh has been moved exactly. The method therefore possesses the *S Property* in the l^∞ norm over a single time step.

4.5.1 Numerical Results

We now present numerical results for the implementation of the MPCM for the second-order problem (2.7)–(2.9) given in §4.5 for values of $n = 1, 1.5, 2$.

We shall perform experiments over a single time step in order to demonstrate that the implementation of the MPCM possesses the *S Property* in the l^∞ norm. We shall then present results for the MPCM run over multiple time steps in order to illustrate how the method performs over multiple steps.

We shall make use of the similarity solution (2.30) from Pattle (1959) and rewritten for general n:

$$u^S(x,s) = \frac{1}{s}\left(\frac{n\beta}{2}\right)^{1/n}\left[\omega^2 - \frac{x^2}{s^2}\right]_+^{1/n}, \tag{4.32}$$

where $\beta = (n+2)^{-1}$ is a constant and where the scaled time variable s is used in place of t^β from the given paper. We shall use this similarity solution to set the initial approximation and to test the accuracy of the numerical solutions calculated by the MPCM.

We can use (4.32) to calculate the similarity velocity $v^S(x,s)$ using (3.14):

$$v^S(x,s) = \frac{\beta x}{s^{1/\beta}}.$$

Single Time Step

We now run the implementation of the MPCM for the second-order problem (2.7)–(2.9) over a single time step. A mesh of $N = 21$ nodes is used, with the time step Δs in the scale invariant time stepping scheme taken to be $\Delta s = 10^{-4}$.

The initial approximation is taken at time $s^0 = 1$, such that

$$U_i^0 = \left(\frac{n\beta}{2}\right)^{1/n}\left[1 - (X_i^0)^2\right]_+^{1/n},$$

for $i = 1, \ldots, N$. This initial approximation is centred about $x = 0$ with initial boundary positions at $x = \pm 1$.

Once the single time step has been performed we calculate the absolute error in the approximation $\mathbf{V}^0$ and the updated solution $\mathbf{U}^1$. These errors are shown in Figures 4.7 and 4.8 respectively, for $n = 1, 1.5, 2$.

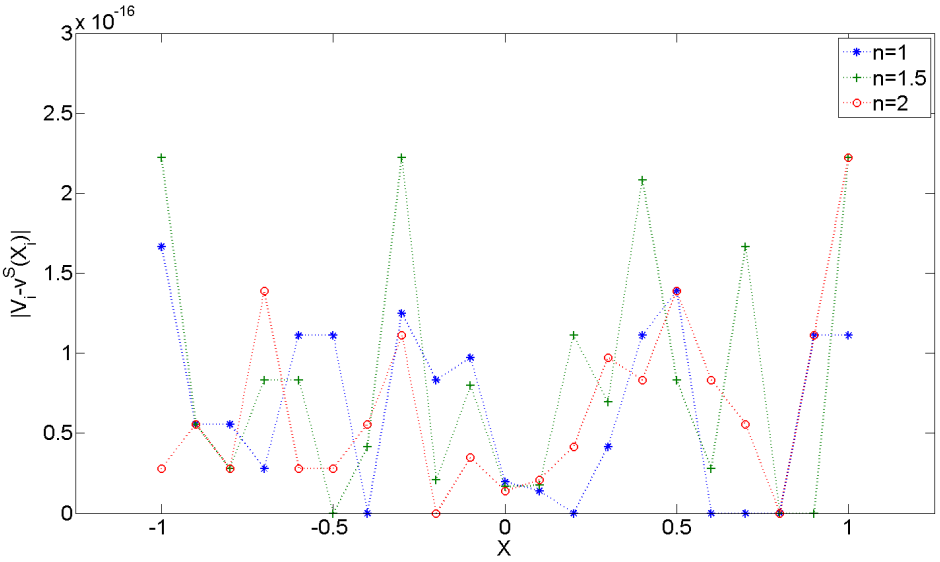

Figure 4.7: Absolute error in the velocity in the MPCM over a single time step. The blue line with asterisk markers denotes $n = 1$, the green line with + markers denotes $n = 2$ and the red line with o markers denotes $n = 3$. Vertical scale on the plot is $O(10^{-16})$.

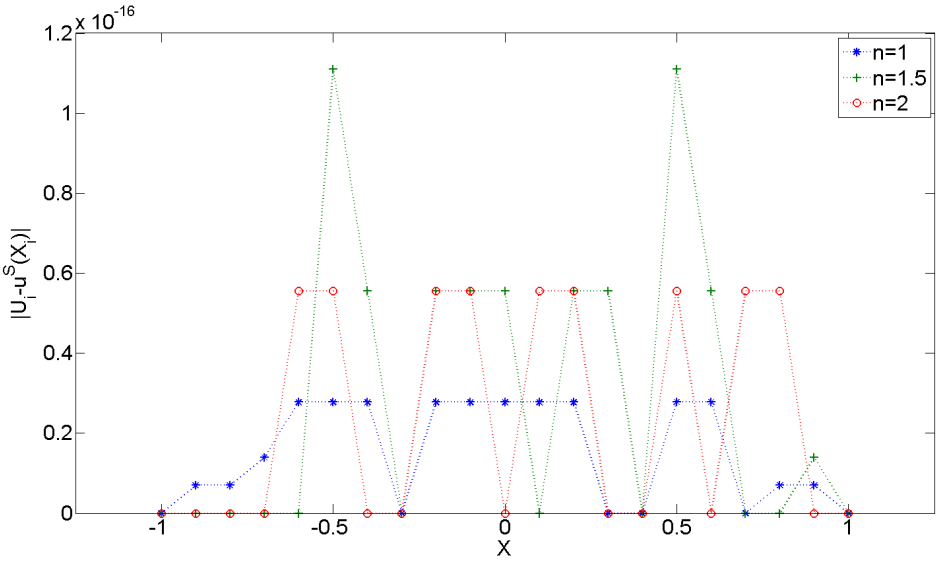

Figure 4.8: Absolute error in the approximate solution in the MPCM over a single time step. The blue line with asterisk markers denotes $n = 1$, the green line with + markers denotes $n = 1.5$ and the red line with o markers denotes $n = 2$. Vertical scale on the plot is $O(10^{-16})$.

We can see that the errors shown in Figures 4.7 and 4.8 are in the region of rounding error, as expected, for each value of n. This indicates that the MPCM is operating as expected over a single time step.

Multiple Time Steps

If we repeat the experiment performed above for a single time step, but instead allow the method to run for multiple time steps, we would anticipate that an accumulation of rounding error would cause the error in the approximate solution to increase over the time window (as observed for the fourth-order implementation in §4.3.6).

We therefore run the MPCM implementation using the same value of Δs as in the single time step case ($\Delta s = 10^{-4}$), over a time window of $s \in (1, 2.5)$, which equates to 15000 steps. This choice of Δs is made such that $\Delta s = O(1/N^2)$ in order to reduce the possibility of node tangling occurring (see §3.3.7).

At the end of each time step, we calculate the l^∞ norm of the error, given by (in the case of the approximate solution $\mathbf{U}^k$)

$$E_N^U = \max_{i=1,\ldots<N} |U_i^k - u^S(X_i^k, s^k)|, \qquad \text{for } k = 1, \ldots, K,$$

where K is the total number of time steps performed. The resulting error is plotted in Figure 4.9, for $n = 1, 1.5, 2$.

From Figure 4.9 we observe that the error E_{21}^U does increase over the time window, reaching $O(10^{-13})$ by the final time point for all three choices of n. This increase is slow however, with a difference in the error between successive time steps no larger than 10^{-16} in any instance.

A similar expression is also calculated for E_{21}^V. These results are given in Figure 4.10 for the three choices of n. In these cases we see that the errors are erratic, increasing sharply at the beginning of the time window and then increasing and decreasing multiple times. The error between successive time steps is no larger than rounding error in all cases.

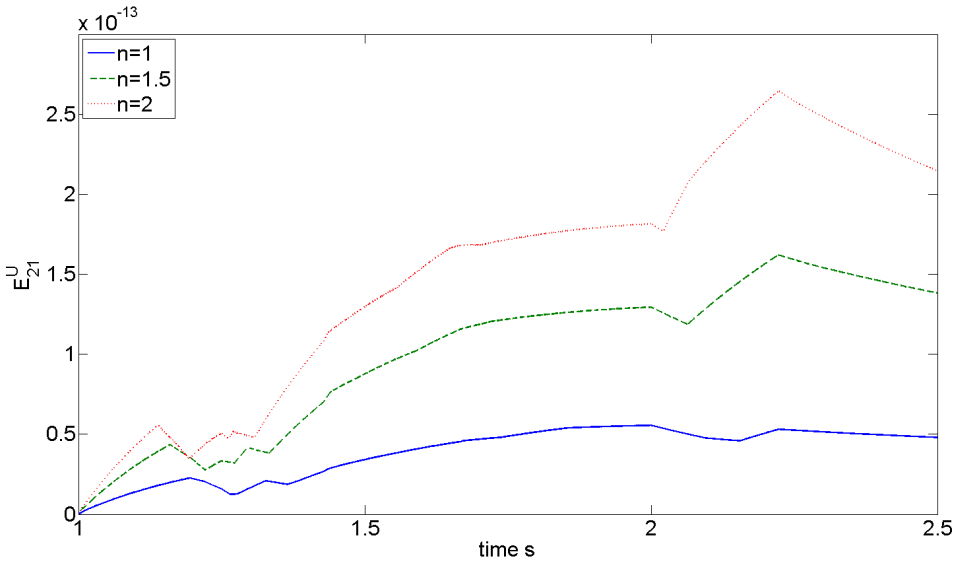

Figure 4.9: l^∞ error in the approximate solution $\mathbf{U}^k$ from the MPCM over multiple time steps. The solid blue line is for $n = 1$, the dashed green line for $n = 1.5$ and the dash-dotted red line for $n = 2$. Vertical scale of the plot is $O(10^{-13})$.

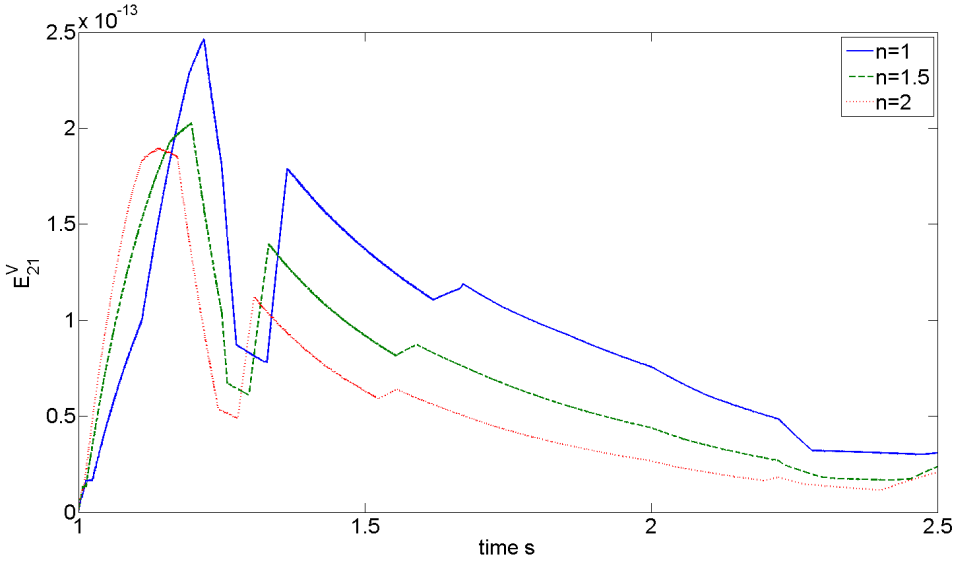

Figure 4.10: l^∞ error in the velocity $\mathbf{V}^k$ from the MPCM over multiple time steps. The solid blue line is for $n = 1$, the dashed green line for $n = 1.5$ and the dash-dotted red line for $n = 2$. Vertical scale of the plot is $O(10^{-13})$.

We also calculate E_N^B, the error in the right-hand boundary position, given by

$$E_N^B = \frac{|X_N^k - b(s^k)|}{|b(s^k)|}, \qquad \text{for } k = 1, \ldots, K,$$

where $b(s^k) = s^k$ denotes the exact position of the right-hand boundary as time s^k (from (4.32)). This boundary error is plotted in Figure 4.11.

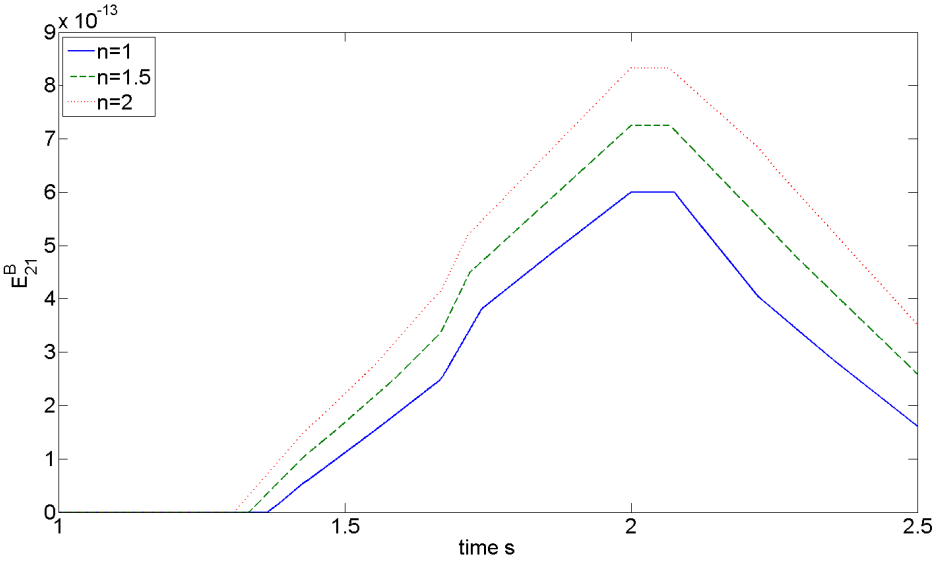

Figure 4.11: l^∞ error in the right-hand boundary position from the MPCM over multiple time steps. The solid blue line is for $n = 1$, the dashed green line for $n = 1.5$ and the dash-dotted red line for $n = 2$. Vertical scale of plot is $O(10^{-13})$.

From Figure 4.11 we see that the error remains near zero for a short time, before beginning to rise and then fall towards the end of the time window. The small value of this error indicates that the method is performing as expected, with results remaining in the vicinity of rounding error throughout the time window.

4.6 The S Property in the MPCM for the Sixth-Order Problem

We shall now demonstrate that the implementation of the MPCM given in §3.5 for the sixth-order problem (2.10)–(2.12) can be modified in order to possess the *S Property* (see §3.6) in the l^∞ norm. As closed-form similarity solutions exist only in the case when $n = 1$, we shall focus on this choice of n in this section. We shall also describe the implementation in terms of the scaled time variable s as we use a scale-invariant time stepping scheme in this implementation.

For clarity, we summarise the problem being considered, for the choice of $n = 1$. We seek solutions $u(x, s)$ to the sixth-order PDE

$$\begin{aligned} \frac{\partial u}{\partial s} &= \frac{\partial}{\partial x}\left(u\frac{\partial q}{\partial x}\right), \\ q(x,s) &= -\frac{\partial^2 p}{\partial x^2}, \\ p(x,s) &= -\frac{\partial^2 u}{\partial x^2}, \end{aligned}$$

for $x \in \Omega_S$, subject to the boundary conditions

$$\begin{aligned} u &= 0, && \text{at } x = a(s), \quad x = b(s), \quad s > s^0, \\ \frac{\partial u}{\partial x} &= 0, && \text{at } x = a(s), \quad x = b(s), \quad s > s^0, \\ \frac{\partial^3 u}{\partial x^3} &= 0, && \text{at } x = a(s), \quad x = b(s), \quad s > s^0, \\ uv + u\frac{\partial q}{\partial x} &= 0, && \text{at } x = a(s), \quad x = b(s), \quad s > s^0. \end{aligned}$$

We seek approximations

$$\begin{aligned} \mathbf{U}^k &:= \{U_i^k\}, \quad \text{for } i = 2, \ldots, N-1, \\ \mathbf{P}^k &:= \{P_i^k\}, \quad \text{for } i = 1, \ldots, N, \\ \mathbf{Q}^k &:= \{Q_i^k\}, \quad \text{for } i = 1, \ldots, N, \\ \mathbf{V}^k &:= \{V_i^k\}, \quad \text{for } i = 1, \ldots, N, \end{aligned}$$

on a set of mesh points $\mathbf{X}^k$ with $U_1^k = U_N^k = 0$ for all k.

Similarity solutions to the sixth-order problem (2.10)–(2.12) exist in the case when $n = 1$ which are sextic functions (Smyth and Hill (1988)). The function $p(x, s)$ is therefore a quartic function, while $q(x, s)$ will be a quadratic. We modify our implementation of the MPCM given in §3.5 such that it is exact for sextic $u(x, s)$, quartic $p(x, s)$ and quadratic $q(x, s)$ in the case when $n = 1$.

We are given an initial mesh $\mathbf{X}^0$ discretising the domain $\Omega(s^0)$ and an approximation $\mathbf{U}^0$ to the initial condition such that $u^0(x) = u^S(x, s^0)$ at each node of the mesh.

4.6.1 Approximating $p(x, s)$

We seek to find an approximation $\mathbf{P}^k$ to $p(x, s)$. The scheme (3.34) given in §3.5.1 has a local truncation error proportional to $\frac{\partial p}{\partial x}$ at node i of the mesh and as such is not exact for sextic $u(x, s)$, quartic $p(x, s)$. We therefore wish to modify the first approximation (3.34) such that it becomes exact for quartic $p(x, s)$.

Given U_i^k, $i = 1, \ldots, N$, (denoted by U_i for ease of notation), we compute P_i^k, $i = 1, \ldots, N$ (denoted by P_i for ease of notation) as follows.

The modified scheme for obtaining $\mathbf{P}$ at interior points in the mesh is given by a seven-point central differencing scheme of the form

$$
\begin{aligned}
P_i = &- \left(\frac{\frac{U_{i+1}-U_i}{h_i} - \frac{U_i-U_{i-1}}{h_{i-1}}}{\frac{1}{2}(h_i + h_{i-1})} \right) + C_1 \left(\frac{P_{i+1} - P_{i-1}}{h_i + h_{i-1}} \right) + C_2 \left(\frac{\frac{P_{i+1}-P_i}{h_i} - \frac{P_i-P_{i-1}}{h_{i-1}}}{\frac{1}{2}(h_i + h_{i-1})} \right) \\
&+ \frac{C_3}{(h_i + h_{i-1})} \left(\left(\frac{\frac{P_{i+2}-P_{i+1}}{h_{i+1}} - \frac{P_{i+1}-P_i}{h_i}}{\frac{1}{2}(h_{i+1} + h_i)} \right) - \left(\frac{\frac{P_i-P_{i-1}}{h_{i-1}} - \frac{P_{i-1}-P_{i-2}}{h_{i-2}}}{\frac{1}{2}(h_{i-1} + h_{i-2})} \right) \right) \\
&+ \frac{C_4}{\frac{1}{2}(h_{i+1} + h_i + h_{i-1} + h_{i-2})} \left(\left(\frac{\frac{P_{i+3}-P_{i+2}}{h_{i+2}} - \frac{P_{i+2}-P_{i+1}}{h_{i+1}}}{\frac{1}{2}h_{i+1}(h_{i+2} + h_{i+1})} \right) - \left(\frac{\frac{P_{i+2}-P_{i+1}}{h_{i+1}} - \frac{P_{i+1}-P_i}{h_i}}{\frac{1}{2}h_{i+1}(h_{i+1} + h_i)} \right) \right. \\
&\left. - \left(\frac{\frac{P_i-P_{i-1}}{h_{i-1}} - \frac{P_{i-1}-P_{i-2}}{h_{i-2}}}{\frac{1}{2}h_{i-2}(h_{i-1} + h_{i-2})} \right) - \left(\frac{\frac{P_{i-1}-P_{i-2}}{h_{i-2}} - \frac{P_{i-2}-P_{i-3}}{h_{i-3}}}{\frac{1}{2}h_{i-2}(h_{i-2} + h_{i-3})} \right) \right)
\end{aligned}
\tag{4.36}
$$

where we aim to choose the constants C_1, C_2, C_3 and C_4 such that all terms up to and including p_i'''' in the local truncation error are equal to zero. We have introduced

$h_{i+2} = X_{i+3} - X_{i+2}$, $h_{i+1} = X_{i+2} - X_{i+1}$, $h_i = X_{i+1} - X_i$, $h_{i-1} = X_i - X_{i-1}$, $h_{i-2} = X_{i-1} - X_{i-2}$ and $h_{i-3} = X_{i-2} - X_{i-3}$ for ease of notation.

The local truncation error τ_i in the modified method (4.36) can be shown to be equal to

$$\begin{aligned}\tau_i = p_i + u_i'' &- \left[\frac{h_i - h_{i-1}}{3} + C_1\right] p_i' - \left[\frac{h_i^3 + h_{i-1}^3}{12(h_i + h_{i-1})} + \frac{C_1}{2}(h_i - h_{i-1}) + C_2\right] p_i'' \\ &- \left[\frac{h_i^4 - h_{i-1}^4}{60(h_i + h_{i-1})} + \frac{C_1}{6}(h_i^2 - h_i h_{i-1} + h_{i-1}^2) + \frac{C_2}{3}(h_i - h_{i-1}) + C_3\right] p_i''' \\ &- \left[\frac{h_i^5 + h_{i-1}^5}{360(h_i + h_{i-1})} + \frac{C_1}{24}(h_i^3 - h_i^2 h_{i-1} + h_i h_{i-1}^2 - h_{i-1}^3)\right. \\ &\left. + \frac{C_2}{12}(h_i^2 - h_i h_{i-1} + h_{i-1}^2) + \frac{C_3}{2}(h_i - h_{i-1}) + C_4\right] p_i''''\Big|_\theta,\end{aligned} \tag{4.37}$$

for some $\theta \in (X_{i-3}, X_{i+3})$, where $p_i = p(X_i)$, $p_i' = \frac{\partial p}{\partial x}\Big|_{X_i}$, $p_i'' = \frac{\partial^2 p}{\partial x^2}\Big|_{X_i}$, $p_i''' = \frac{\partial^3 p}{\partial x^3}\Big|_{X_i}$, $p_i'''' = \frac{\partial^4 p}{\partial x^4}\Big|_{X_i}$.

In order to remove the p_i' terms from equation (4.37), we choose our constant C_1 such that

$$C_1 = -\frac{(h_i - h_{i-1})}{3}. \tag{4.38}$$

Similarly, the p_i'' terms can be removed from equation (4.37) through utilising the constant C_1 in equation (4.38), and setting

$$\begin{aligned}C_2 &= -\frac{C_1}{2}(h_i - h_{i-1}) - \frac{h_i^3 + h_{i-1}^3}{12(h_i + h_{i-1})}, \\ &= \frac{(h_i - h_{i-1})^2}{6} - \frac{h_i^3 + h_{i-1}^3}{12(h_i + h_{i-1})}.\end{aligned}$$

Removing the p_i''' terms in (4.37) involves setting the constant C_3 such that

$$\begin{aligned}C_3 &= -\frac{h_i^4 - h_{i-1}^4}{60(h_i + h_{i-1})} - \frac{C_1}{6}(h_i^2 - h_i h_{i-1} + h_{i-1}^2) - \frac{C_2}{3}(h_i - h_{i-1}), \\ &= \frac{(2h_i^4 + 8h_{i-1}^4 + 5h_i^3 h_{i-1} - 15 h_i h_{i-1}^3)}{180(h_i + h_{i-1})}.\end{aligned}$$

Finally, setting

$$\begin{aligned} C_4 = &-\frac{h_i^5 + h_{i-1}^5}{360(h_i + h_{i-1})} - \frac{C_1}{24}(h_i^3 - h_i^2 h_{i-1} + h_i h_{i-1}^2 - h_{i-1}^3) \\ &- \frac{C_2}{12}(h_i^2 - h_i h_{i-1} + h_{i-1}^2) - \frac{C_3}{2}(h_i - h_{i-1}), \\ = &-\frac{(h_i^5 - 13h_{i-1}^5 + h_i^4 h_{i-1} + 31 h_i h_{i-1}^4 - 5h_i^3 h_{i-1}^2 - 15 h_i^2 h_{i-1} 3)}{720(h_i + h_{i-1})}, \end{aligned}$$

removes all p_i'''' terms from (4.37).

Equation (4.36) is then used for $i = 4 \ldots N-3$. For the values of $i = 2, 3, N-2, N-1$ (i.e. the remaining interior points of the mesh) we make use of a reflective boundary condition to determine the values of P_i.

By making use of the boundary condition

$$\frac{\partial^3 u}{\partial x^3} = -\frac{\partial p}{\partial x} = 0,$$

at the moving boundaries $x = a(s)$ and $x = b(s)$, we are able to use a reflective boundary condition in order to extend the method described by equation (4.36) to the required points near the boundary.

By writing

$$P_0 = P_2, \qquad P_{-1} = P_3, \qquad P_{N+1} = P_{N-1}, \qquad P_{N+2} = P_{N-2},$$

and

$$X_1 - X_0 = X_2 - X_1, \qquad X_0 - X_{-1} = X_3 - X_2,$$

$$X_{N+1} - X_N = X_N - X_{N-1}, \qquad X_{N+2} - X_{N+1} = X_{N-2} - X_{N-1},$$

we can use (4.36) at the points $i = 2, 3, N-2, N-1$ as required.

At the boundary points we use the fourth-degree Lagrange interpolating polynomials to provide the values of P_1 and P_N:

$$L_1(X_6)P_1 + L_2(X_6)P_2 + L_3(X_6)P_3 + L_4(X_6)P_4 + L_5(X_6)P_5 - P_6 = 0,$$

$$\begin{aligned} -P_{N-5} + &L_{N-4}(X_{N-5})P_{N-4} + L_{N-3}(X_{N-5})P_{N-3} + L_{N-2}(X_{N-5})P_{N-2} \\ &+ L_{N-1}(X_{N-5})P_{N-1} + L_N(X_{N-5})P_N = 0, \end{aligned}$$

The values of $\mathbf{P}$ are then obtained by solving the matrix system

$$S\mathbf{P} = T\mathbf{U},$$

where the matrices S, T are of size $N \times N$. The vector $\mathbf{U}$ contains the boundary conditions $U_1 = U_N = 0$ as well as the interior values.

4.6.2 Approximating $q(x, s)$

Given P_i^k, $i = 1, \ldots, N$ (denoted P_i for ease of notation) detailed in §4.6.1, we now seek an approximation Q_i^k, $i = 1, \ldots, N$, (denoted by Q_i for ease of notation) which is exact for quadratic $q(x, s)$, quartic $p(x, s)$.

We have previously outlined the steps required in obtaining a quadratic $q(x, s)$ from a quartic $u(x, s)$ in the fourth-order problem (2.3)–(2.5) (see §4.3.1), so we use the same steps in obtaining a quadratic $q(x, s)$ from quartic $p(x, s)$ by substituting any references to $\mathbf{U}$ by $\mathbf{P}$. We shall restate the appropriate schemes in terms of $\mathbf{P}$ for completeness.

For interior points we obtain $\mathbf{Q}$ using

$$Q_i \approx -\frac{\frac{P_{i+1}-P_i}{h_i} - \frac{P_i-P_{i-1}}{h_{i-1}}}{\frac{1}{2}(h_i + h_{i-1})} + C_1 \frac{Q_{i+1} - Q_{i-1}}{h_i + h_{i-1}} + C_2 \frac{\frac{Q_{i+1}-Q_i}{h_i} - \frac{Q_i-Q_{i-1}}{h_{i-1}}}{\frac{1}{2}(h_i + h_{i-1})},$$

with choices of

$$C_1 = -\frac{(h_i + h_{i-1})}{3}, \qquad C_2 = \frac{h_i^2 - 3h_i h_{i-1} + h_{i-1}^2}{12},$$

from (4.5) and (4.6).

The local truncation error of this approximation is then proportional to q_i''' and will therefore be exact for a quadratic $q(x, s)$.

At the boundary points we use the second-degree Lagrange interpolating polynomials described in §3.5.2.

$\mathbf{Q}$ is then found by solving the matrix system

$$S\mathbf{Q} = T\mathbf{P},$$

where the matrices S and T are from (4.7) and (4.9) respectively.

4.6.3 Mesh Movement

Updating the mesh is performed exactly as described in §3.3.5, using the scale invariant time stepping scheme (4.13).

This scheme has the same truncation error properties shown for the fourth-order problem (2.3)–(2.5) in §4.3.3 and so we would expect the mesh points to be moved to within rounding error of the exact values.

4.6.4 Solution Recovery

The solution is recovered on the updated mesh using (3.30), just as for the fourth-order problem (2.3)–(2.5) shown in §4.3.4.

We are able to show as in the fourth-order problem (2.3)–(2.5) that the solution to the sixth-order problem (2.10)–(2.12) is recovered exactly on the updated mesh, provided that the mesh has been moved exactly.

4.7 Summary of this Chapter

In this chapter we introduced modifications to the basic MPCM which enable the method to possess the *S Property* when solving the fourth-order problem (2.3)–(2.5) with $n = 1$. The required modifications to the method were discussed and numerical results verified the possession of the *S Property* in the MPCM. A build up of rounding error in the method was identified to be emerging from the time stepping stage of the method and shown to be bounded.

The MPCM was also shown to possess the *S Property* for the second-order problem (2.7)–(2.9) for any n, which was verified by numerical results. The MPCM can also be modified for the sixth-order problem (2.10)–(2.12) with $n = 1$ in such a way that it is expected to possess the *S Property*. A description of the modifications required is presented, without numerical verification (due to a lack of a working program code).

In the next chapter we outline a finite element implementation of the MPCM. We initially implement the BHJ method of Baines, Hubbard and Jimack (Baines et al.

(2005, 2006, 2011)), before showing that through careful selection of basis functions it is possible to construct a finite element method which possesses the *S Property* in the same cases as were demonstrated in this chapter (i.e. $n = 1$ for fourth/sixth order and any n for second-order).

Chapter 5

A Finite Element Implementation of the MPCM

5.1 Aims of this Chapter

In this chapter we introduce a finite element implementation of the MPCM (which we refer to as the FEMPCM) for use in solving the fourth-order problem (2.3)–(2.5). By selecting the basis functions in the method appropriately, we show that for $n = 1$ the FEMPCM can possess the *S Property* in the L^2 norm.

An outline of the weak forms of the various equations is provided, before describing the implementation of the FEMPCM for the fourth order problem (2.3)–(2.5) for any choice of n in the case where the basis functions are piecewise-linear 'hat' functions. This implementation is similar to the BHJ method (Baines et al. (2005, 2006, 2011)). Numerical results for this implementation are provided.

An alternative implementation of the FEMPCM for the fourth-order problem is then provided for $n = 1$, with the modifications resulting in the method possessing the *S Property* in the L^2 norm. Numerical results demonstrating this alternative implementation are then given, validating the *S Property* claim.

We also describe how the FEMPCM can be implemented for the second-order problem (2.7)–(2.9) and sixth-order (2.10)–(2.12) problem, without numerical results.

5.2 Comparison between the MPCM and FEMPCM

In this section we shall provide a brief description of the main differences between the MPCM and the FEMPCM. In doing so, we hope to provide some justification as to why a finite element implementation of the MPCM is necessary, since we have already demonstrated that the MPCM possesses the *S Property* in certain cases (see Chapter 4).

The main difference between the finite difference implementation of the MPCM and the finite element implementation of the FEMPCM is the validity of the resulting solution in the domain $\Omega(t)$. In the MPCM the solution obtained by the method is valid only at the mesh points forming the discretisation, with no information available at other points of $\Omega(t)$ not lying on the mesh.

In comparison, the solution obtained by the FEMPCM is valid throughout the whole domain $\Omega(t)$ and not only at mesh points. This allows for a greater knowledge of such a solution, since unless additional information is available concerning a finite difference solution (e.g. that the solution is a polynomial of a certain degree), the available knowledge is restricted to the mesh.

This difference can be alleviated by the use of an increased number of points in the mesh, at an increased computational expense in performing the finite difference method. For a given number of mesh points, the FEMPCM is more computationally expensive due to the need to solve at more than just the mesh points. Increasing the number of mesh points would then negate the reduced computational expense advantage of the MPCM.

This comparison is rounded off here by mentioning that when considering the implementation of either the MPCM or FEMPCM in higher dimensions, the finite element method is more naturally extended from 1D to 2D due to the simplicity of discretising the 2D domain using triangulation. Implementation of the MPCM in higher dimensions would be a much more difficult task than in 1D.

5.3 Weak Forms

Motivated by a desire to implement a conservation-based method using finite elements, the required weak forms of equations in the fourth-order problem (2.3)–(2.5) are constructed.

Weak forms are generally constructed by multiplying the equation by a time-dependent test function $w(x,t) \in W$, where W is a suitable test space of functions to be defined in due course. The resulting equation is then integrated with respect to x. In the approach used here weak forms are constructed in a moving framework.

Remark. *Throughout this chapter we shall assume that functions in the test space W are suitably smooth. In particular, if $\frac{\partial u}{\partial t}$ and $u^n \frac{\partial q}{\partial x}$ are both L^2 in space then $\mathcal{H}^1$ is a suitable choice for W. This chapter is motivated by the implementation of the FEMPCM, as opposed to a thorough analysis of the required spaces or the spaces in which functions lie, so these details are left deliberately vague.*

5.3.1 Weighted Conservation of Mass

In order to determine a weak form for the velocity $v(x,t)$ using conservation, we require the implementation of a weighted conservation of mass principle, as opposed to the local conservation of mass principle described in §3.2.5.

Let $w(x,t)$ be a weight function in W. We then differentiate the weighted mass $\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x$ with respect to t using Leibniz's Integral Rule, giving

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial}{\partial t}\left(w(x,t)u(x,t)\right)\,\mathrm{d}x + \Big[w(x,t)u(x,t)v(x,t)\Big]_{a(t)}^{b(t)}$$

where

$$v(b(t),t) = \frac{\mathrm{d}b(t)}{\mathrm{d}t}, \qquad v(a(t),t) = \frac{\mathrm{d}a(t)}{\mathrm{d}t}. \tag{5.1}$$

This implies that

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x &= \int_{a(t)}^{b(t)} w(x,t)\frac{\partial u}{\partial t}+\frac{\partial w}{\partial t}u(x,t)+\frac{\partial}{\partial x}\left(wuv\right)\,\mathrm{d}x,\\ &= \int_{a(t)}^{b(t)}\left[w\frac{\partial u}{\partial t}+\frac{\partial w}{\partial t}u+\frac{\partial u}{\partial x}wv+\frac{\partial w}{\partial x}uv+\frac{\partial v}{\partial x}wu\right]\,\mathrm{d}x, \quad (5.2)\\ &= \int_{a(t)}^{b(t)}\left[u\left(\frac{\partial w}{\partial t}+v\frac{\partial w}{\partial x}\right)+w\left(\frac{\partial u}{\partial t}+\frac{\partial}{\partial x}(uv)\right)\right]\,\mathrm{d}x,\end{aligned}$$

where $v(x,t)$ is any velocity consistent with the boundary velocities (5.1). We now specify the evolution of functions $w(x,t)$ in the space W which vary in time such that $w(x,t)$ is convected with the velocity $v(x,t)$ in the weak sense that

$$\int_{a(t)}^{b(t)} u(x,t)\left(\frac{\partial w}{\partial t}+v(x,t)\frac{\partial w}{\partial x}\right)\,\mathrm{d}x=0 \qquad \forall w\in W, \tag{5.3}$$

which reduces equation (5.2) to

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x=\int_{a(t)}^{b(t)} w(x,t)\left(\frac{\partial}{\partial x}\left(uv\right)+\frac{\partial u}{\partial t}\right)\,\mathrm{d}x. \tag{5.4}$$

for all $w\in W$ satisfying (5.3).

Using equation (2.3a) we may write equation (5.4) in the moving integral form

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} w(x,t)u\,\mathrm{d}x-\int_{a(t)}^{b(t)} w(x,t)\frac{\partial}{\partial x}\left(uv\right)\,\mathrm{d}x=-\int_{a(t)}^{b(t)}\frac{\partial w}{\partial x}\left(u^n\frac{\partial q}{\partial x}\right)\,\mathrm{d}x, \tag{5.5}$$

where $v(x,t)$ is yet to be chosen but must be consistent with (5.1) and (5.3).

We now define the velocity $v(x,t)$ implicitly by the weighted conservation of mass principle,

$$\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x=\rho(w), \qquad \forall w\in W, \tag{5.6}$$

where the weighted masses $\rho(w)$ remain constant in time. In order to be consistent with the global conservation of mass (2.15) the functions $w(x,t)$ must constitute a partition of unity (this is tacitly assumed in what follows). Equation (5.6) implies that

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{a(t)}^{b(t)} w(x,t)u(x,t)\,\mathrm{d}x=0, \qquad \forall w\in W, \tag{5.7}$$

and hence from (5.4) that

$$\int_{a(t)}^{b(t)} w(x,t)\left(\frac{\partial}{\partial x}\left(uv\right)+\frac{\partial u}{\partial t}\right)\,\mathrm{d}x=0, \qquad \forall w\in W, \tag{5.8}$$

which are respectively weak forms of Lagrangian and Eulerian conservation principles for the function $u(x,t)$.

Using equations (5.5) and (5.7) we see that $v(x,t)$ is defined by

$$\int_{a(t)}^{b(t)} w(x,t)\frac{\partial}{\partial x}\left(uv + u^n\frac{\partial q}{\partial x}\right)\,\mathrm{d}x = 0,$$

for all $w \in W$ (consistent with (5.8)), which can be integrated by parts in order to produce

$$\left[w\left(uv + u^n\frac{\partial q}{\partial x}\right)\right]_{a(t)}^{b(t)} - \int_{a(t)}^{b(t)} \frac{\partial w}{\partial x}\left(u^n\frac{\partial q}{\partial x} + uv\right)\,\mathrm{d}x = 0, \qquad \forall w \in W. \tag{5.9}$$

Due to the zero net flux boundary conditions (2.5c) and (5.1) the first term in (5.9) vanishes, which results in equation (5.9) being rewritten as

$$\int_{a(t)}^{b(t)} u(x,t)\frac{\partial w}{\partial x}\left(v + u^{n-1}\frac{\partial q}{\partial x}\right)\,\mathrm{d}x = 0, \qquad \forall w \in W, \tag{5.10}$$

which is the weak form of the equation (2.17a) for the velocity $v(x,t)$.

5.3.2 Additional Weak Forms

Depending upon the order of the problem being considered, there may be a requirement to construct weak forms of equations for $q(x,t)$ ((2.3b) or (2.10b) for the fourth-order problem (2.3)–(2.5) or sixth-order problem (2.10)–(2.12) respectively) and/or $p(x,t)$ ((2.10c) for the sixth-order problem (2.10)–(2.12)).

In the fourth-order problem (2.3)–(2.5) we require a weak form of (2.3b), which is

$$\int_{a(t)}^{b(t)} w(x,t)q(x,t)\,\mathrm{d}x = -\int_{a(t)}^{b(t)} w(x,t)\frac{\partial^2 u}{\partial x^2}\,\mathrm{d}x, \qquad \forall w \in W.$$

Integrating by parts and applying the boundary condition (2.5b) gives the weak form

$$\int_{a(t)}^{b(t)} w(x,t)q(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial w}{\partial x}\frac{\partial u}{\partial x}\,\mathrm{d}x, \qquad \forall w \in W, \tag{5.11}$$

for $q(x,t)$.

In the sixth-order problem (2.10)–(2.12) we require weak forms of (2.10b) and (2.10c). Following the same steps as were performed above, we can find weak forms

$$\int_{a(t)}^{b(t)} w(x,t)q(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial w}{\partial x}\frac{\partial p}{\partial x}\,\mathrm{d}x, \qquad \forall w \in W. \tag{5.12}$$

for $q(x,t)$ and

$$\int_{a(t)}^{b(t)} w(x,t)p(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial w}{\partial x}\frac{\partial u}{\partial x}\,\mathrm{d}x, \qquad \forall w \in W. \tag{5.13}$$

for $p(x,t)$.

5.4 An Implementation of the FEMPCM for the Fourth-Order Problem

We now describe an implementation of the FEMPCM for the fourth-order problem (2.3)–(2.5). The implementation is based on the weak forms (5.6), (5.10) and (5.11) constructed in §5.3.

5.4.1 Finite Dimensional Subspaces

From the weak forms described in §5.3 we can construct a moving-mesh finite element method. We select finite-dimensional subspaces $\mathcal{S}_E^1(t), \mathcal{S}^1(t), \widehat{\mathcal{S}}^1(t), \widetilde{\mathcal{S}}^1(t) \in W$, where the subscript E indicates that functions in that subspace satisfy zero Dirichlet boundary conditions. We then approximate $u(x,t)$ by a function $U(x,t) \in \mathcal{S}_E^1(t)$. We also approximate $q(x,t)$ by $Q(x,t) \in \mathcal{S}^1(t)$ and $v(x,t)$ by $V(x,t) \in \widehat{\mathcal{S}}^1(t)$.

5.4.2 The Finite Element Approximation

We now begin the task of constructing the FEMPCM.

We firstly seek the finite element approximation $Q(x,t)$ of $q(x,t)$ for a given $U(x,t) \in \mathcal{S}_E^1(t)$ in the form: Given $U(x,t) > 0$, find $Q \in \mathcal{S}^1(t)$ such that

$$\int_{a(t)}^{b(t)} w(x,t)Q(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} \frac{\partial w}{\partial x}\frac{\partial U}{\partial x}\,\mathrm{d}x \qquad \forall w \in \mathcal{S}^1(t). \tag{5.14}$$

From (5.10) we then seek a Finite Element approximation $V(x,t)$ of $v(x,t)$ for a given $U(x,t) \in \mathcal{S}_E^1(t)$, $Q(x,t) \in \mathcal{S}^1(t)$ in the form: Given $U(x,t)$, $Q(x,t)$, find $V \in \widehat{\mathcal{S}}^1(t)$ such that

$$\int_{a(t)}^{b(t)} U(x,t)\frac{\partial w}{\partial x}\left[V + U^{n-1}\frac{\partial Q}{\partial x}\right]\,\mathrm{d}x = 0 \qquad \forall w \in \widehat{\mathcal{S}}^1(t). \tag{5.15}$$

5.4.3 Updating the Nodal Positions

Once $Q(x,t)$ and $V(x,t)$ have been obtained, the evolution of each moving point with coordinate $X(t)$ is determined by solving the ODE

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = V(X(t),t),$$

using a suitable time stepping method.

5.4.4 Recovering the Solution $U(x,t)$

Since we have stipulated that the distributed masses ρ_i remain constant over time, we are able to use the evolved moving points $X(t)$ in order to update our numerical solution $U(x,t)$. This is achieved by inverting the distributed weighted form of the Conservation of Mass Principle (5.6) within the appropriate subspace.

From equation (5.6) (on the updated domain), we seek $U(x,t) \in \mathcal{S}_E^1(t)$ such that

$$\int_{a(t)}^{b(t)} w(x,t)U(x,t)\,\mathrm{d}x = \rho(w), \qquad \forall w \in \widetilde{\mathcal{S}}^1(t), \tag{5.16}$$

where the $\rho(w)$ are constant in time and hence known from initial data.

By making specific choices of the test functions $w(x,t)$ we are able to construct a moving finite element method. We shall next outline two different implementations for the fourth-order problem (2.3)–(2.5). The first implementation, using piecewise linear basis functions, can be used for any choice of n in (2.3a) and is based on the method of Baines, Hubbard and Jimack (the BHJ method, see Baines et al. (2005, 2006, 2011)). The second implementation is limited to the $n = 1$ case and makes use of higher order spaces in order to produce a method which possesses the *S Property* (§3.6) in the L^2 norm.

5.5 The FEMPCM with Piecewise Linear Basis Functions

In Baines et al. (2005, 2006) a moving-mesh finite element method is described which uses piecewise linear basis functions in its formulation. In this section we describe a

one dimensional implementation of this method, following the steps in §5.4. We shall provide an overview of the implementation, since many of the details are found in the given papers (with the main difference being in the solution recovery step, see §5.5.3). Additionally, numerical results on this implementation for $n = 2$ shall be provided.

The first step is to discretise the domain defined by the support of the solution $U(x,t)$ at a given time level t^k into elements using a number of nodes. If there are N nodes in the domain this implies that there are $N-1$ elements to consider. The nodal positions are given by the vector $\mathbf{X}^k = \{X_i^k\}$, where X_i^k is the position of node $i, (i = 1, \ldots, N)$ at time level t^k.

We shall require that $X_i^k < X_{i+1}^k$, for $i = 1, \ldots, N-1$, at any given time t^k otherwise the nodes will have tangled. If the nodes tangle in any time step, the solution recovery step (see §5.5.3) will cause the solution to become negative, which is inadmissible as the the local conservation of mass principle (3.30) will no longer be consistent with conservation of the total mass of the solution. The solution becoming negative also causes the method to fail to preserve nonnegativity of the solution.

Remark. *The results given in Section 4.2 (in particular Figure 4) of Baines et al. (2005) demonstrate that in a finite element method for obtaining approximate solutions to the fourth-order problem* (2.3)–(2.5)*, the choice of piecewise linear basis functions is suitable in terms of accuracy of the resulting solution. The authors demonstrate results with fourth order accuracy when* $n = 1$ *for both the solution and mesh error, when the initial condition is sampled from a similarity solution.*

5.5.1 Basis Functions

We next define the basis functions to be used in this implementation of the moving mesh method. We choose the basis functions $w(x,t)$ to be the linear Lagrange polynomials $\phi_i(x,t)$, $i = 1, \ldots, N$, lying in the space of piecewise linear functions. An example of such a function is given in Figure 5.1.

The test functions have a time dependence due to the fact that the nodal positions move with the velocity $v(x,t)$.

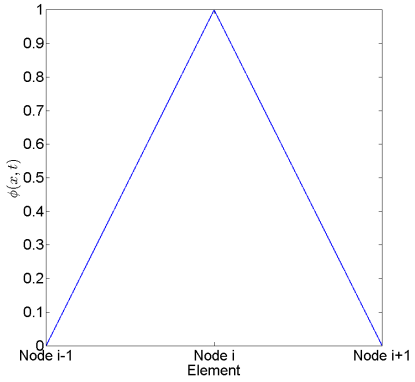

Figure 5.1: An example $\phi_i(x,t)$ function used in this implementation of the FEMPCM, plotted over a two consecutive elements.

The approximations $U(x,t)$, $Q(x,t)$ and $V(x,t)$ in the method are defined as piecewise linear functions, which we write as linear combinations of the $\phi_i(x,t)$:

$$U(x,t) = \sum_{j=2}^{N-1} \phi_j(x,t) U_j(t), \tag{5.17a}$$

$$Q(x,t) = \sum_{j=1}^{N} \phi_j(x,t) Q_j(t), \tag{5.17b}$$

$$V(x,t) = \sum_{j=1}^{N} \phi_j(x,t) V_j(t), \tag{5.17c}$$

with the reduced number of $\phi_i(x,t)$ in (5.17a) due to the $u = 0$ boundary conditions.

5.5.2 Finite Dimensional Subspaces

Throughout §5.5 our function approximations are chosen to lie in finite dimensional subspaces of the test space given by

$$\mathcal{S}_E^1(t) = \text{span}\{\phi_j(x,t)\}, \qquad \text{for } j = 2, \ldots, N-1,$$

$$\mathcal{S}^1(t),\ \widehat{\mathcal{S}}^1(t)\ \widetilde{\mathcal{S}}(t) = \text{span}\{\phi_j(x,t)\}, \qquad \text{for } j = 1, \ldots, N,$$

The reduced span of $S_E^1(t)$ is due to the boundary conditions $u = 0$ and therefore there are no test functions attributed to the boundary points in this case.

5.5.3 An Algorithm for the FEMPCM

With the above information, we next describe the algorithm used to obtain the approximations $Q(x,t)$, $V(x,t)$ and $U(x,t)$:

Obtaining the $Q(x,t)$ Approximation

Given $U(x,t)$ (either from the initial approximation or the previous time step), we obtain $Q(x,t)$. The coefficients $Q_j(t)$ of $Q(x,t)$ in (5.17b) can be obtained by solving the matrix system

$$M\mathbf{Q} = K\mathbf{U}, \tag{5.19}$$

where the mass and stiffness matrices M and K are of dimension $N \times N$ and $N \times (N-2)$, respectively.

M is a tridiagonal 'Mass' matrix with time-dependent entries

$$M_{ij}(t^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \phi_1(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \phi_i(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} \phi_N(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = N, \end{cases}$$

for $j = 1,\ldots,N$, while the matrix K is a rectangular 'Stiffness' matrix with time-dependent entries

$$K_{ij}(t^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \phi_1'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \phi_i'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = 2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} \phi_N'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = N, \end{cases}$$

for $j = 2,\ldots,N-1$, where the prime $'$ denotes partial differentiation with respect to x.

Obtaining the Velocity $V(x,t)$

Using the $Q(x,t)$ obtained in the previous step, we can now obtain the approximation $V(x,t)$ to the velocity $v(x,t)$. The coefficients $V_j(t)$ of $V(x,t)$ in (5.17c) can be obtained by solving the matrix system

$$B(U)\mathbf{V} = -\widetilde{K}(U^n)\mathbf{Q}, \tag{5.20}$$

where $B(U)$ is a tridiagonal weighted stiffness matrix of dimension $N \times N$ with time-dependent entries given by

$$B(U)_{ij}(t^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} U(x,t)\phi_1'(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} U(x,t)\phi_i'(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} U(x,t)\phi_N'(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = N. \end{cases} \tag{5.21}$$

Entries of $B(U)$ can be evaluated using a suitable quadrature method (such as three-point Gaussian quadrature which should evaluate entries of $B(U)$ to within rounding error).

The matrix $\widetilde{K}(U^n)$ is a tridiagonal weighted stiffness matrix of dimension $N \times N$ with time-dependent entries given by

$$\widetilde{K}(U^n)_{ij}(t^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} U^n(x,t)\phi_1'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} U^n(x,t)\phi_i'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = 2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} U^n(x,t)\phi_N'(x,t)\phi_j'(x,t)\,\mathrm{d}x, & \text{for } i = N. \end{cases} \tag{5.22}$$

Entries of $\widetilde{K}(U^n)$ can be evaluated using a suitable quadrature method (such as three or five-point Gaussian Quadrature). If n is an integer then these integrals can be calculated exactly since the integrand is a polynomial (e.g. if $n = 1$ then three-point Gaussian quadrature is suitable), but for non-integer n this will not be the case.

Introducing a Velocity Potential $z(x,t)$

The skew-symmetric matrix $B(U)$ may be ill-conditioned or singular in some cases (when U is uniform and N is odd, $B(U)$ is singular for example) and hence we may be unable to accurately determine $\mathbf{V}$ using (5.20) (see Baines et al. (2005, 2011)). We can overcome this issue through the introduction of a velocity potential $z(x,t)$ such that

$$v(x,t) = \frac{\partial z}{\partial x}.$$

The weak form (5.10) now becomes

$$\int_{a(t)}^{b(t)} u \frac{\partial w}{\partial x} \left(\frac{\partial z}{\partial x} + u^{n-1} \frac{\partial q}{\partial x} \right) \mathrm{d}x = 0, \qquad \forall w \in W.$$

We then seek the finite element approximation $Z(x,t)$ of $z(x,t)$ for a given $U(x,t) \in \mathcal{S}_E^1(t)$, $Q(x,t) \in \mathcal{S}^1(t)$ in the form: Find $Z(x,t) \in \mathcal{S}^1(t)$ such that

$$\int_{a(t)}^{b(t)} U(x,t) \frac{\partial w}{\partial x} \left[\frac{\partial Z}{\partial x} + U^{n-1} \frac{\partial Q}{\partial x} \right] \mathrm{d}x = 0, \qquad \forall w \in \widehat{\mathcal{S}}^1(t), \tag{5.23}$$

in place of (5.15).

Choosing w in (5.23) to be one of the $\phi_i(x,t)$ functions, writing $Q(x,t)$ and $U(x,t)$ using (5.17b) and (5.17a) respectively, and writing $Z(x,t)$ as the linear combination

$$Z(x,t) = \sum_{j=1}^{N} \phi_j(x,t) Z_j(t),$$

we can obtain the coefficients $Z_j(t)$ by solving the matrix system

$$\widetilde{K}(U)\mathbf{Z} = -\widetilde{K}(U^n)\mathbf{Q},$$

where $\widetilde{K}(U)$ and $\widetilde{K}(U^n)$ have entries given by (5.22) (using $n = 1$ in (5.22) in the case of the former).

The $\widetilde{K}(U)$ matrix is singular (see Remark 5.1 below) and hence non-invertible, but we may obtain a unique $\mathbf{Z}$ through the imposition of a suitable constraint, such as

$$\sum_{j=1}^{N} Z_j(t) = 0, \tag{5.24}$$

thereby also putting a constraint on $\mathcal{S}^1(t)$. There is no loss of generality since $z(x,t)$ is a potential and has a free parameter.

Remark 5.1. *The matrix $\widetilde{K}(U)$ is singular as a result of the choice of basis functions made in this implementation of the FEMPCM. Over a single element, the basis functions have been chosen in order that they sum to one (i.e. that they form a partition of unity). The derivative of such functions will therefore sum to zero over a single element.*

This results in the determinant of the $\widetilde{K}(U)$ matrix being equal to zero, and hence that the matrix is singular.

Using $Z(x,t)$ to obtain $V(x,t)$

An additional step is required to obtain the velocity $V(x,t)$ from $Z(x,t)$. We use the weak form

$$\int_{a(t)}^{b(t)} w(x,t)\left(v(x,t) - \frac{\partial z}{\partial x}\right) \mathrm{d}x = 0, \qquad \forall w \in W,$$

from which we seek the finite element approximation $V(x,t)$ of $v(x,t)$ given $Z(x,t) \in \mathcal{S}^1(t)$ in the form: Find $V \in \widehat{\mathcal{S}}^1(t)$ such that

$$\int_{a(t)}^{b(t)} w(x,t)V(x,t)\,\mathrm{d}x = \int_{a(t)}^{b(t)} w(x,t)\frac{\partial Z}{\partial x}\,\mathrm{d}x, \qquad \forall w \in \widehat{\mathcal{S}}^1(t).$$

The coefficients of $V(x,t)$ can be obtained by solving the matrix system

$$M\mathbf{V} = B\mathbf{Z},$$

where the M matrix is as given in (5.19) and is well conditioned. The B matrix is of dimension $N \times N$ and has entries given by (5.21) without the factor $U(x,t)$. Due to the nature of the $\phi_i(x,t)$ functions, entries of B are constant valued (and hence can be calculated once and used for all time).

Time Stepping

Using the approximation to the velocity at the nodes we can update the positions of the mesh points in $\mathbf{X}^k$ to those at the next time step $\mathbf{X}^{k+1}$. For example, the Forward Euler method (3.26) constructs the $\mathbf{X}^{k+1}$ vector at the next time $t^{k+1} = t^k + \Delta t$ such that

$$\mathbf{X}^{k+1} = \mathbf{X}^k + \Delta t \mathbf{V}^k,$$

where the vector $\mathbf{V}^k$ contains the coefficients $V_j(t^k)$ of the velocity from (5.17c) calculated from (5.20) and where Δt denotes the size of the time step.

Recovery of Solution $U(x,t)$ on the Updated Mesh

With the new nodal positions calculated, we can recover the solution $U(x,t)$ using the updated mesh values $\mathbf{X}^{k+1}$. This step is the main difference between the implementation presented in Baines et al. (2005, 2006, 2011) and the method presented here. The coefficients $U_j(t)$ in (5.17a) can be found by solving the matrix system

$$A\mathbf{U} = \boldsymbol{\rho}, \tag{5.25}$$

where A is of dimension $N \times N-2$ and has time-dependent entries given by

$$A_{ij}(t^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \phi_1(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \phi_i(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = 2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} \phi_N(x,t)\phi_j(x,t)\,\mathrm{d}x, & \text{for } i = N, \end{cases}$$

for $j = 2,\ldots,N-1$. The lack of values for $j = 1$ and N is due to the reduced number of $\phi_j(x,t)$ in $U(x,t)$ (see (5.17a)). The values of $\boldsymbol{\rho} = \{\rho_i\}$, $i = 1,\ldots,N$, can be obtained using the initial condition $u^0(x)$, where

$$\rho_i = \int_{a(t^0)}^{b(t^0)} \phi_i(x,t^0)u^0(x)\,\mathrm{d}x, \qquad \text{for } i = 1,\ldots,N. \tag{5.26}$$

The integrals in (5.26) can be obtained using a suitable numerical integration scheme (such as Gaussian quadrature), or in some cases can be evaluated exactly depending upon the choice of $u^0(x)$.

The system given in (5.25) is overdetermined at present, since the matrix A has dimension $N \times N-2$. In order to alleviate this issue we may introduce modified basis

functions $\tilde{\phi}_i(x,t)$, such that

$$\begin{aligned}
\tilde{\phi}_1(x,t) &= \phi_1(x,t) + \phi_2(x,t), && \text{for } i = 1, \\
\tilde{\phi}_i(x,t) &= \phi_{i+1}(x,t), && \text{for } i = 2, \ldots, N-3, \\
\tilde{\phi}_{N-2}(x,t) &= \phi_{N-1}(x,t) + \phi_N(x,t), && \text{for } i = N-2,
\end{aligned}$$

which still form a partition of unity. Note that there are only $N-2$ of these basis functions.

Figure 5.2 gives an example of the $\tilde{\phi}_1(x,t)$ basis function. We see that the function has a value of one over the first element and is linear over the second element. The $\tilde{\phi}_{N-2}(x,t)$ basis function is similar, but over the final two elements of the mesh.

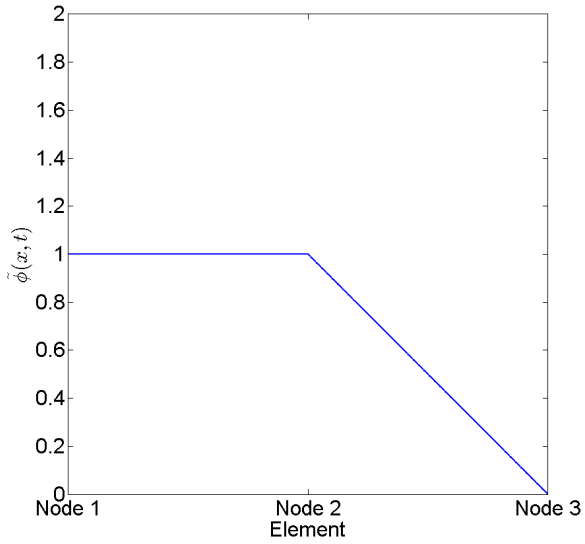

Figure 5.2: The $\tilde{\phi}_1(x,t)$ basis function, plotted over the first two elements of the mesh.

The use of these modified basis functions also changes the constants ρ_i so that new constants $\tilde{\rho}_i$ are produced, i.e.,

$$\begin{aligned}
\tilde{\rho}_1(x,t) &= \rho_1(x,t) + \rho_2(x,t), && \text{for } i = 1, \\
\tilde{\rho}_i(x,t) &= \rho_{i+1}(x,t), && \text{for } i = 2, \ldots, N-3, \\
\tilde{\rho}_{N-2}(x,t) &= \rho_{N-1}(x,t) + \rho_N(x,t), && \text{for } i = N-2.
\end{aligned}$$

Then,

$$\widetilde{A}\mathbf{U} = \tilde{\boldsymbol{\rho}}, \tag{5.29}$$

where $\widetilde{A}$ is a mass matrix of dimension $N-2 \times N-2$ with entries given by

$$\widetilde{A}_{ij}(t^k) = \int_{a(t)}^{b(t)} \tilde{\phi}_i(x,t)\phi_j(x,t)\,\mathrm{d}x, \qquad \text{for } i,j = 1,\ldots,N-2.$$

The coefficients $U_j(t)$ obtained through solving the matrix system (5.29) give $U(x,t)$ at the interior mesh points. If we had solved for $U(x,t)$ at all N nodes of the mesh, the zero boundary condition (2.5a) would only have been enforced weakly and hence U_1 and U_N would have been unknowns (Baines et al. (2005)). By introducing the modified basis functions with their partition of unity property we are able to solve for unknown $U_2,\ldots,U_{N-1}$ and combine this information with the boundary condition $U_1 = U_N = 0$ to ensure that the zero boundary condition (2.5a) is strongly enforced. Further details of the strong enforcement of boundary conditions can be found in Hubbard et al. (2009).

5.5.4 Numerical Results

We shall now present numerical results for this implementation of the FEMPCM described in §5.5, for the choice of $n = 2$.

We shall perform experiments on meshes of different numbers of nodes. In each successive experiment we increase the number of nodes such that the initial mesh spacing is halved (on an initially uniformly-spaced mesh), resulting in meshes of $N = 21$, 41, 81 and 161 nodes. In each experiment the method is run over the time window $t \in [1, 1.25]$, with the time step size Δt kept proportional to $1/N^4$ in order to reduce the risk of node tangling occurring. Specifically, we choose Δt such that

$$\Delta t = \frac{10^{-4}}{16^j} \qquad \text{for } j = 1, 2, \ldots.$$

The initial condition used in the experiments is

$$u^0(x) = \frac{1}{120}\left[4 - x^2\right]_+^2,$$

which is incidentally a similarity solution in the $n = 1$ case at time $t^0 = 1$ but has no such relevance to $n = 2$. This initial condition is symmetric about $x = 0$ with initial

boundary positions at $x = \pm 2$. This initial condition is used to calculate the ρ_i values and the initial approximation $U^0(x)$ following the steps in §5.5.3.

At the end of the time window, we record the nodal values of the approximations $U(x, 1.25)$, $Q(x, 1.25)$ and $V(x, 1.25)$. We also record the position of the right-hand boundary point at each time step of the method.

Figures 5.3, 5.4 and 5.5 show the nodal values for the various meshes. In the case of the approximations $U(x, 1.25)$ and $Q(x, 1.25)$ there appears to be little visible difference between the various meshes. In the case of the approximate velocity $V(x, 1.25)$ however, there is a noticeable difference in the nodal values near to the boundary points.

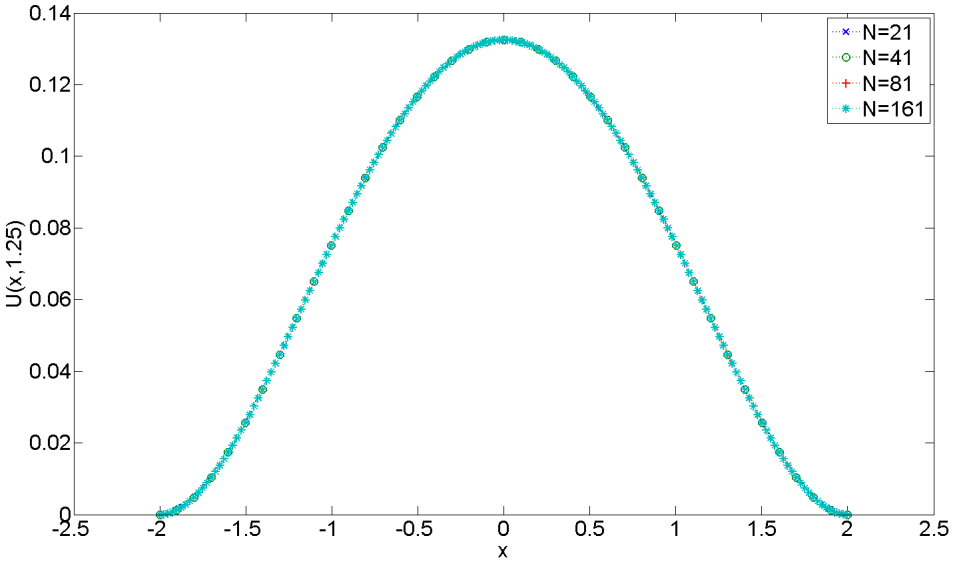

Figure 5.3: Nodal values of $U(x, 1.25)$ on meshes of $N = 21$ (blue xs), 41 (green os), 81 (red $+$s) and 161 (cyan $*$s) nodes.

This difference in the nodal values of the velocity $V(x, 1.25)$ causes the nodes to move differently in each of the different meshes. This is illustrated clearly in the case of the right-hand boundary, which is shown in figure 5.6. As the number of nodes in the mesh increases, the boundary moves a smaller distance over the time window. Examining the approximate velocities in figure 5.5, the boundary velocity is approaching zero as the

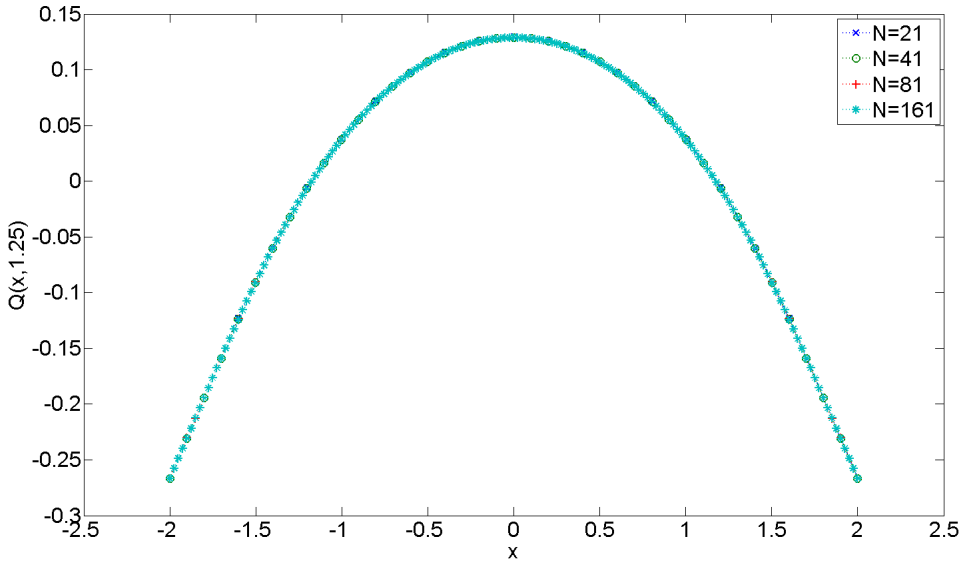

Figure 5.4: Nodal values of $Q(x, 1.25)$ on meshes of $N = 21$ (blue xs), 41 (green os), 81 (red +s) and 161 (cyan $*$s) nodes.

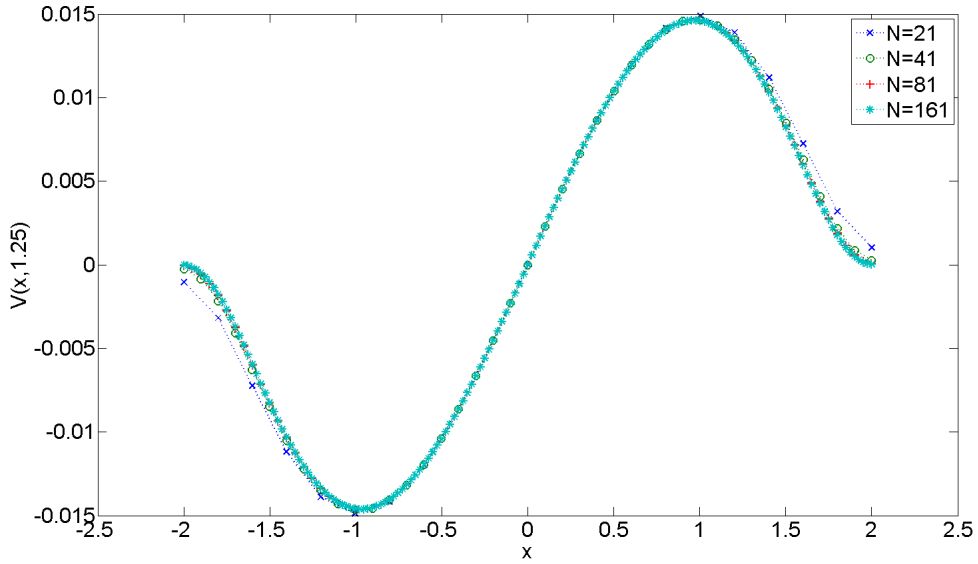

Figure 5.5: Nodal values of $V(x, 1.25)$ on meshes of $N = 21$ (blue xs), 41 (green os), 81 (red +s) and 161 (cyan $*$s) nodes.

number of nodes increases. This will then cause the boundary to move outwards less (and is consistent with the expected velocity as we shall discuss in Chapter 6).

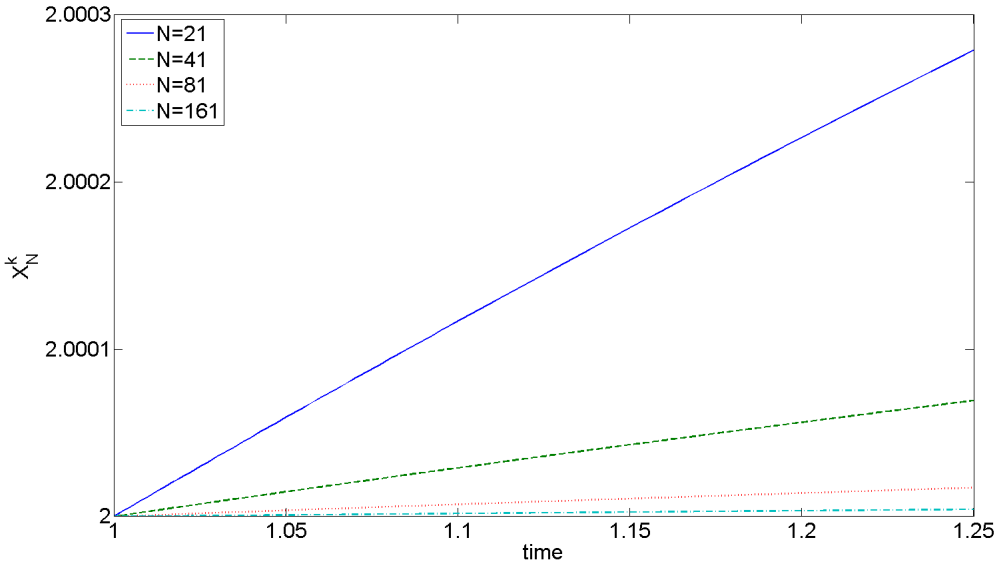

Figure 5.6: Position of the right-hand boundary node on meshes of $N = 21$ (blue solid line), 41 (green dashed line), 81 (red dotted line) and 161 (cyan dash-dotted line) nodes.

Figure 5.6 also provides some information as to the convergence of the linear finite element method in that the boundary trajectories for the examples shown display a lack of convergence. The velocities given in Figure 5.5 show that the boundary velocity appears to be tending towards zero as N increases, so further investigation into the boundary trajectories for larger N may provide evidence of convergence.

5.6 An Implementation of the FEMPCM with the S Property

The implementation of the FEMPCM described in §5.5 is a general method for obtaining approximate solutions to the fourth-order problem (2.3)–(2.5) for any value of n. In this section we shall describe an alternative implementation of the FEMPCM which, for

$n = 1$ only, possesses the *S Property* in the L^2 norm.

The steps involved in the implementation are similar to those described in §5.5, which leads to some repetition between the information in this section and §5.5. The basis functions used in this section are of higher order however, and as such many of the details are more involved. We shall therefore describe the implementation in more detail than that given in §5.5. We shall also be using the scale-invariant time stepping scheme (5.47), so we replace any reference to the time variable t with the new scaled time variable s in this section.

The similarity solution for the fourth-order problem (2.3)–(2.5) with $n = 1$ is a quartic function (Smyth and Hill (1988)). This results in the exact values of $q(x, s)$ and $v(x, s)$ being quadratic and linear respectively. We shall therefore choose our finite-dimensional subspaces $\mathcal{S}_E^1(s)$, $\mathcal{S}^1(s)$ $\widehat{\mathcal{S}}^1(s)$ in this particular implementation such that $U(x, s)$ is piecewise quartic, $Q(x, s)$ piecewise quadratic and $V(x, s)$ piecewise linear. Combined with a suitable choice of piecewise quartic, quadratic and linear basis function in each weak form and scale-invariant time stepping, the method can be shown to possess the *S Property* in the L^2 norm.

5.6.1 Discretisation

The first step is to discretise the domain defined by the support of the solution $U(x, s)$ at a given time level s^k into elements, using a number of nodes. If there are N nodes in the domain this implies that there are $N - 1$ elements to consider. The nodal positions are given by the vector $\mathbf{X}^k = \{X_i^k\}$, where X_i^k is the position of node $i, (i = 1, \ldots, N)$ at time level s^k.

As stated in §5.5, we shall require that $X_i^k < X_{i+1}^k$, at any given time s^k, for $i = 1, \ldots, N-1$, otherwise the nodes are considered to have tangled (with the consequences regarding resulting loss of consistency between local and global mass conservation and loss of positivity of the solution applying).

5.6.2 Basis Functions

We next define the basis functions to be used throughout this implementation of the moving mesh method for the fourth order problem (2.3)–(2.5) with $n = 1$. We choose the basis functions $w(x, s)$ to be Lagrange polynomials of either first, second or fourth order, as follows:

- For functions lying in the space of piecewise quartics we set $w(x, s)$ to be a fourth order Lagrange polynomial denoted by $\chi_i(x, s)$, for $i = 1, \ldots, 4N - 3$. In a given element we require five different $\chi_i(x, s)$ to specify a unique quartic function, resulting in a total of $4N - 3$ basis functions $\chi_i(x, s)$ over the mesh.

- For the space of piecewise quadratics we choose $w(x, s)$ to be a quadratic Lagrange polynomial denoted by $\varphi_i(x, s)$ for $i = 1, \ldots, 2N - 1$. There are three different $\varphi_i(x, s)$ required to specify a unique quadratic function over a given element, resulting in $2N - 1$ basis functions $\varphi_i(x, s)$ over the mesh.

- Finally, we define $w(x, s)$ to be a linear Lagrange polynomial, denoted by $\phi_i(x, s)$, $i = 1, \ldots, N$, chosen for functions lying in the space of piecewise linears. There are two different $\phi_i(x, s)$ required to define a unique linear function over a given element, resulting in a total of N basis functions $\phi_i(x, s)$ over the mesh.

In Figure 5.7 we plot examples of these functions over a single element, with $\chi_i(x, s)$ given in the left plot, $\varphi_i(x, s)$ in the central plot and $\phi_i(x, s)$ in the right plot.

The test functions have a time dependence due to the fact that the nodal positions move with the velocity $v(x, s)$.

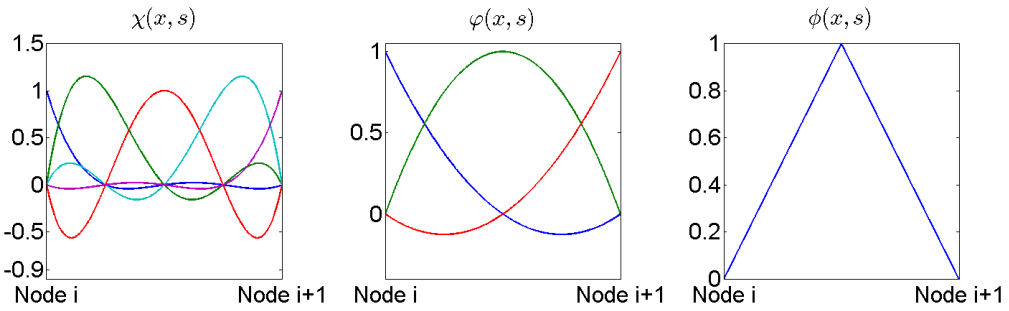

Figure 5.7: Plot of the various choices of Lagrange polynomial used in this implementation of the Finite Element Method, plotted over a single element.

5.6.3 Finite Dimensional Subspaces

Throughout §5.6, our function approximations are chosen to lie in finite dimensional subspaces of the test space given by

$$\begin{aligned}
\mathcal{S}_E^1(s) &= \text{span}\{\chi_j(x,s)\}, && \text{for } j = 2,\ldots,\hat{N}-1,\\
\mathcal{S}^1(s) &= \text{span}\{\varphi_j(x,s)\}, && \text{for } j = 1,\ldots,\tilde{N},\\
\widehat{\mathcal{S}}^1(s) &= \text{span}\{\phi_j(x,s)\}, && \text{for } j = 1,\ldots N,\\
\widetilde{\mathcal{S}}^1(s) &= \text{span}\{\chi_j(x,s)\}, && \text{for } j = 1,\ldots,\hat{N}.
\end{aligned}$$

where we have introduced $\hat{N} = 4N-3$ and $\tilde{N} = 2N-1$ for ease of notation. The reduced span of $S_E^1(s)$ is due to the boundary conditions $u = 0$ and therefore there are no test functions attributed to the boundary points in this case.

5.6.4 Obtaining the $Q(x,s)$ Approximation

Given $U(x,s)$, we can then find the approximation $Q(x,s)$ to $q(x,s)$ from (5.14).

We choose our test function $w(x,s)$ in (5.14) to be one of the $\varphi_i(x,s)$ $(i = 1,\ldots,\tilde{N})$.

Due to the properties of the $\varphi_i(x,s)$ functions this produces the system of equations

$$\begin{aligned}
\int_{a(s)}^{b(s)} \varphi_1(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_1'(x,s)U'(x,s)\,\mathrm{d}x, && \text{for } i=1,\\
\int_{a(s)}^{b(s)} \varphi_i(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_i'(x,s)U'(x,s)\,\mathrm{d}x, && \text{for } i=2,\ldots,\tilde{N}-1,\\
\int_{a(s)}^{b(s)} \varphi_{\tilde{N}}(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_{\tilde{N}}'(x,s)U'(x,s)\,\mathrm{d}x, && \text{for } i=\tilde{N}.
\end{aligned}$$

Writing $Q(x,s)$ as a linear combination of the $\varphi_j(x,s)$,

$$Q(x,s) = \sum_{j=1}^{\tilde{N}} \varphi_j(x,s)Q_j(s), \tag{5.32}$$

and $U(x,s)$ as a linear combination of the $\chi_j(x,s)$,

$$U(x,s) = \sum_{j=2}^{\hat{N}-1} \chi_j(x,s)U_j(s), \tag{5.33}$$

we obtain the system of equations

$$\begin{aligned}
\sum_{j=1}^{\tilde{N}} Q_j(s) \int_{a(s)}^{b(s)} \varphi_1(x,s)\varphi_j(x,s)\,\mathrm{d}x &= \sum_{j=2}^{\hat{N}-1} U_j(s) \int_{a(s)}^{b(s)} \varphi_1'(x,s)\chi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i=1,\\
\sum_{j=1}^{\tilde{N}} Q_j(s) \int_{a(s)}^{b(s)} \varphi_j(x,s)\varphi_j(x,s)\,\mathrm{d}x &= \sum_{j=2}^{\hat{N}-1} U_j(s) \int_{a(s)}^{b(s)} \varphi_i'(x,s)\chi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i=2,\ldots,\tilde{N}-1,\\
\sum_{j=1}^{\tilde{N}} Q_j(s) \int_{a(s)}^{b(s)} \varphi_{\tilde{N}}(x,s)\varphi_j(x,s)\,\mathrm{d}x &= \sum_{j=2}^{\hat{N}-1} U_j(s) \int_{a(s)}^{b(s)} \varphi_{\tilde{N}}'(x,s)\chi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i=\tilde{N}.
\end{aligned}$$

which is equivalent to solving the matrix system

$$M\mathbf{Q} = K\mathbf{U}, \tag{5.35}$$

where the mass and stiffness matrices M and K defined below are of dimension $\tilde{N}\times\tilde{N}$ and $\tilde{N}\times(\hat{N}-2)$, respectively.

The time-dependent entries of M are given by

$$M_{ij}(s^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \varphi_1(x,s)\varphi_j(x,s)\,\mathrm{d}x & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \varphi_i(x,s)\varphi_j(x,s)\,\mathrm{d}x & \text{for } i = 2,\ldots,\tilde{N}-1, \\ \displaystyle\int_{X_{\tilde{N}-1}^k}^{X_{\tilde{N}}^k} \varphi_{\tilde{N}}(x,s)\varphi_j(x,s)\,\mathrm{d}x & \text{for } i = \tilde{N}, \end{cases}$$

for $j = 1,\ldots,\tilde{N}$, while time-dependent entries of K are given by

$$K_{ij}(s^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \varphi_1'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \varphi_i'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = 2,\ldots,\tilde{N}-1, \\ \displaystyle\int_{X_{\tilde{N}-1}^k}^{X_{\tilde{N}}^k} \varphi_{\tilde{N}}'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = \tilde{N}, \end{cases}$$

for $j = 1,\ldots,\tilde{N}$. Entries of M and K are evaluated using a suitable numerical integration scheme (such as Gaussian quadrature).

5.6.5 Obtaining the Velocity Potential $Z(x,s)$

Using the approximation $Q(x,s)$ calculated by solving (5.35) we can now obtain the approximation $Z(x,s)$ to the velocity potential $z(x,s)$. As in the linear finite element implementation §5.5, we introduce a velocity potential to avoid potential issues with the skew-symmetric matrix $B(U)$ being singular or ill-conditioned (such as if U is uniform and N odd).

We introduce the velocity potential $z(x,s)$ such that

$$v(x,s) = \frac{\partial z}{\partial x},$$

with the weak form (5.10) now becoming

$$\int_{a(s)}^{b(s)} u\frac{\partial w}{\partial x}\left(\frac{\partial z}{\partial x} + \frac{\partial q}{\partial x}\right)\,\mathrm{d}x = 0, \qquad \forall w \in W. \tag{5.36}$$

We then make use of the finite element approximation (5.23), which we repeat here for clarity (in terms of s): Find $Z(x,s) \in \mathcal{S}^1(s)$ such that

$$\int_{a(s)}^{b(s)} U(x,s)\frac{\partial w}{\partial x}\left[\frac{\partial Z}{\partial x}+\frac{\partial Q}{\partial x}\right]\,\mathrm{d}x = 0, \qquad \forall w \in \widehat{\mathcal{S}}^1(s), \tag{5.37}$$

which is used instead of (5.15).

Choosing w in (5.37) to be one of the $\phi_i(x,s)$ functions, writing $Q(x,s)$ using (5.32), and writing $Z(x,s)$ as the linear combination

$$Z(x,s) = \sum_{j=1}^{\tilde{N}} \varphi_j(x,s) Z_j(s), \tag{5.38}$$

we can obtain the coefficients $Z_j(s)$ by solving the matrix system

$$\widetilde{K}(U)\mathbf{Z} = -\widetilde{K}(U)\mathbf{Q}, \tag{5.39}$$

where $\widetilde{K}(U)$ is of dimension $\tilde{N}\times\tilde{N}$ and has time-dependent entries given by

$$\widetilde{K}(U)_{ij}(s^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} U(x,s)\phi_1'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i=1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} U(x,s)\phi_i'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i=2,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} U(x,s)\phi_N'(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i=N, \end{cases} \tag{5.40}$$

for $j=1,\ldots,\tilde{N}$.

The $\widetilde{K}(U)$ matrix is singular and hence non-invertible (see Remark 5.1). We may obtain a unique $\mathbf{Z}$ through the imposition of a suitable constraint, such as (5.24)

$$\sum_{j=1}^{\tilde{N}} Z_j(s) = 0,$$

thereby putting a constraint on $\mathcal{S}^1(s)$. There is no loss of generality since $z(x,t)$ is a potential with a free parameter.

5.6.6 Obtaining the Velocity $V(x,s)$

The velocity $V(x,s)$ can be obtained from $Z(x,s)$ through the weak form

$$\int_{a(s)}^{b(s)} w(x,s)\left(v(x,s)-\frac{\partial z}{\partial x}\right)\,\mathrm{d}x = 0, \qquad \forall w \in W, \tag{5.41}$$

from which we seek the finite element approximation $V(x,s)$ of $v(x,s)$ given $Z(x,s) \in \mathcal{S}^1(s)$ in the form: Find $V \in \widehat{\mathcal{S}}^1(s)$ such that

$$\int_{a(s)}^{b(s)} w(x,s)V(x,s)\,\mathrm{d}x = \int_{a(s)}^{b(s)} w(x,s)\frac{\partial Z}{\partial x}\,\mathrm{d}x, \qquad \forall w \in \widehat{\mathcal{S}}^1(s). \tag{5.42}$$

We select $w(x,s)$ in (5.42) to be one of the $\phi_i(x,s)$ functions ($i = 1,\ldots,N$), resulting in the system of equations

$$\begin{aligned}
\int_{a(s)}^{b(s)} \phi_1(x,s)V(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \phi_1(x,s)\frac{\partial Z}{\partial x}\,\mathrm{d}x, && \text{for } i = 1,\\
\int_{a(s)}^{b(s)} \phi_i(x,s)V(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \phi_i(x,s)\frac{\partial Z}{\partial x}\,\mathrm{d}x, && \text{for } i = 2,\ldots,N-1,\\
\int_{a(s)}^{b(s)} \phi_N(x,s)V(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \phi_N(x,s)\frac{\partial Z}{\partial x}\,\mathrm{d}x, && \text{for } i = N.
\end{aligned}$$

Expanding $V(x,s)$ as the linear combination

$$V(x,s) = \sum_{j=1}^{N} \phi_j(x,s)V_j(s), \tag{5.44}$$

and using the expansion for $Z(x,s)$ given by (5.38), we obtain the modified system of equations

$$\begin{aligned}
\sum_{j=1}^{N} V_j(s)\int_{a(s)}^{b(s)} \phi_1(x,s)\phi_j(x,s)\,\mathrm{d}x &= \sum_{j=1}^{\tilde{N}} Z_j(s)\int_{a(s)}^{b(s)} \phi_1(x,s)\varphi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i = 1,\\
\sum_{j=1}^{N} V_j(s)\int_{a(s)}^{b(s)} \phi_i(x,s)\phi_j(x,s)\,\mathrm{d}x &= \sum_{j=1}^{\tilde{N}} Z_j(s)\int_{a(s)}^{b(s)} \phi_i(x,s)\varphi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i = 2,\ldots,N-1,\\
\sum_{j=1}^{N} V_j(s)\int_{a(s)}^{b(s)} \phi_N(x,s)\phi_j(x,s)\,\mathrm{d}x &= \sum_{j=1}^{\tilde{N}} Z_j(s)\int_{a(s)}^{b(s)} \phi_i(x,s)\varphi_j'(x,s)\,\mathrm{d}x,\\
&\qquad \text{for } i = N.
\end{aligned}$$

The coefficients of $Z(x,t)$ can be obtained by solving the matrix system

$$M\mathbf{V} = B\mathbf{Z}, \tag{5.46}$$

where the M matrix is as given in (5.19), and the B matrix is of dimension $N \times \tilde{N}$ and has time-dependent entries given by

$$B_{ij}(s^k) = \begin{cases} \displaystyle\int_{X_1^k}^{X_2^k} \phi_1(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = 1, \\ \displaystyle\int_{X_{i-1}^k}^{X_{i+1}^k} \phi_i(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = 1,\ldots,N-1, \\ \displaystyle\int_{X_{N-1}^k}^{X_N^k} \phi_N(x,s)\varphi_j'(x,s)\,\mathrm{d}x & \text{for } i = N, \end{cases} ,$$

for $j = 1, \ldots, \tilde{N}$.

5.6.7 Time Stepping

Using the approximation to the velocity at the nodes we can update the positions of the nodal points in $\mathbf{X}$.

We shall make use of the scale-invariant Forward Euler time stepping scheme used in the finite difference implementation of the MPCM (given by (4.13)),

$$X_i^{k+1} = X_i^k + 5s^4 \Delta s V_i^k, \qquad \text{for } i = 1, \ldots, N, \tag{5.47}$$

to obtain the mesh positions at time level s^{k+1}. We have used the fact that $\beta = 1/5$ in (4.13) for the choice of $n = 1$ in (5.47).

5.6.8 Recovery of Solution $U(x,s)$ on the Updated Mesh

With the new nodal positions calculated, we can recover the solution $U(x,s)$ using the updated mesh values $\mathbf{X}^{k+1}$ from (5.47).

This is achieved by considering a partitioned form of the distributed conservation of mass principle (5.16). Choosing $w(x,s)$ to be one of the $\chi_i(x,s)$ basis functions, we obtain: Seek $U(x,s) \in \mathcal{S}_E^1(s)$ such that

$$\int_{a(s)}^{b(s)} \chi_i(x,s) U(x,s)\,\mathrm{d}x = \rho_i, \qquad i = 1, \ldots, \hat{N},$$

where the ρ_i are now fixed by our choice of $w(x,s) = \chi_i(x,s)$ at the initial time.

Since the ρ_i values remain constant for all time, we can use the initial conditions $u^0(x)$ to pre-calculate

$$\rho_i = \int_{a(s^0)}^{b(s^0)} \chi_i(x, s^0) u^0(x)\, \mathrm{d}x, \qquad \text{for } i = 1, \ldots, \hat{N}.$$

These integrals can be evaluated by any sufficiently accurate numerical integration scheme (such as five-point Gaussian Quadrature).

As discussed in §5.5.3, the solution recovery step leads to an overdetermined system of equations to solve in obtaining the values of $U_j(s)$ at the interior points. In a similar manner to that in §5.5.3, we can ensure that the system is not overdetermined by introducing modified basis functions $\tilde{\chi}_i(x, s)$ such that

$$\begin{aligned}
\tilde{\chi}_1(x, s) &= \chi_1(x, s) + \chi_2(x, s), && \text{for } i = 1, \\
\tilde{\chi}_i(x, s) &= \chi_{i+1}(x, s), && \text{for } i = 2, \ldots, \hat{N} - 3, \\
\tilde{\chi}_{\hat{N}-2}(x, s) &= \chi_{\hat{N}-1}(x, s) + \chi_{\hat{N}}(x, s), && \text{for } i = \hat{N} - 2,
\end{aligned}$$

which still form a partition of unity and allow the boundary conditions $u = 0$ to be enforced strongly. Note that there are only $\hat{N} - 2$ of these basis functions.

The use of these modified basis functions also changes the constants ρ_i in elements adjacent to the boundaries. These modified ρ_i are denoted by $\tilde{\rho}_i$ and defined such that

$$\begin{aligned}
\tilde{\rho}_1(x, s) &= \rho_1(x, s) + \rho_2(x, s), && \text{for } i = 1, \\
\tilde{\rho}_i(x, s) &= \rho_{i+1}(x, s), && \text{for } i = 2, \ldots, \hat{N} - 3, \\
\tilde{\rho}_{\hat{N}-2}(x, s) &= \rho_{\hat{N}-1}(x, s) + \rho_{\hat{N}}(x, s), && \text{for } i = \hat{N} - 2.
\end{aligned}$$

Initial Approximation $U^0(x)$

Once the $\tilde{\rho}_i$ values have been determined they can be used to obtain an approximation $U^0(x)$ to the initial condition $u^0(x)$ in the space $\mathcal{S}_E^1(s^0)$ as the linear combination

$$U^0(x) = \sum_{j=2}^{\hat{N}-1} \chi_j(x, s^0) U_j^0,$$

with $U_1^0 = U_{\hat{N}}^0 = 0$ from the zero boundary conditions. Once substituted, the partitioned form (5.16) becomes equivalent to solving the system of equations

$$\begin{aligned}
\sum_{j=1}^{\hat{N}-2} U_j^0 \int_{a(s^0)}^{b(s^0)} \tilde{\chi}_1(x,s^0)\chi_j(x,s^0)\,\mathrm{d}x &= \tilde{\rho}_1, && \text{for } i=1,\\
\sum_{j=1}^{\hat{N}-2} U_j^0 \int_{a(s^0)}^{b(s^0)} \tilde{\chi}_i(x,s^0)\chi_j(x,s^0)\,\mathrm{d}x &= \tilde{\rho}_i, && \text{for } i=2,\ldots,\hat{N}-3,\\
\sum_{j=1}^{\hat{N}-2} U_j^0 \int_{a(s^0)}^{b(s^0)} \tilde{\chi}_{\hat{N}-2}(x,s^0)\chi_j(x,s^0)\,\mathrm{d}x &= \tilde{\rho}_{\hat{N}-2}, && \text{for } i=\hat{N}-2,
\end{aligned}$$

which gives the best L^2 fit to the initial condition in the space $\mathcal{S}_E^1(s^0)$. These equations can be written in the matrix form

$$\widetilde{A}^0\mathbf{U}^0 = \tilde{\boldsymbol{\rho}}, \tag{5.51}$$

where $\widetilde{A}$ is a mass matrix of dimension $\hat{N}-2\times\hat{N}-2$ with time-dependent entries given by

$$\widetilde{A}_{ij}(s^k) = \int_{a(s)}^{b(s)} \tilde{\chi}_i(x,s^k)\chi_j(x,s^k)\,\mathrm{d}x, \qquad \text{for } i,j=1,\ldots,\hat{N}-2.$$

and $\boldsymbol{\rho} = \{\rho_i\}$. In (5.51) $\widetilde{A}^0$ is taken at the time $s = s^0$.

Solving equation (5.51) provides the coefficients for the approximation $U^0(x)$ to the initial conditions $u^0(x)$.

Solution Recovery for $s > s^0$

The solution $U(x,s)$ is then recovered by at each time from the partitioned form (5.16) of the distributed conservation of mass principle using the expansion (5.33) of $U(x,s)$. The process in which this occurs is similar to that presented in §5.5.3, but with higher order basis functions. We repeat the steps here for completeness.

By choosing one of the $\tilde{\chi}(x,s)$ as our $w(x,s)$ in the partitioned conservation of mass

principle (5.16), the system of equations becomes

$$\begin{aligned}
\sum_{j=1}^{\hat{N}-2} U_j(s) \int_{a(s)}^{b(s)} \tilde{\chi}_1(x,s)\chi_j(x,s)\,\mathrm{d}x &= \tilde{\rho}_1, && \text{for } i = 1,\\
\sum_{j=1}^{\hat{N}-2} U_j(s) \int_{a(s)}^{b(s)} \tilde{\chi}_i(x,s)\chi_j(x,s)\,\mathrm{d}x &= \tilde{\rho}_i, && \text{for } i = 2,\ldots,\hat{N}-3\\
\sum_{j=1}^{\hat{N}-2} U_j(s) \int_{a(s)}^{b(s)} \tilde{\chi}_{\hat{N}-2}(x,s)\chi_j(x,s)\,\mathrm{d}x &= \tilde{\rho}_{\hat{N}-2}, && \text{for } i = \hat{N}-2,
\end{aligned}$$

which can be written as the matrix system

$$\widetilde{A}\mathbf{U} = \tilde{\boldsymbol{\rho}}. \tag{5.53}$$

We then solve (5.53) to obtain the coefficient $U_j(s)$ of (5.33) at the updated time, while strongly enforcing the zero boundary condition (2.5a). We again refer the reader to Hubbard et al. (2009) for more details of this process.

5.6.9 The FEMPCM over a Single Time Step

We now demonstrate how the particular implementation of the FEMPCM given in §5.6 possesses the *S Property* in the L^2 norm. In cases where the initial condition is taken to be a similarity solution at the initial time s^0 and under a scale-invariant time stepping scheme, the specific choices of basis function throughout §5.6 should cause the approximations $Q(x,s)$, $V(x,s)$ and $U(x,s)$ to be within rounding error of the exact values over a single time step.

We now describe the steps of the FEMPCM over the first time step from s^0 to s^1, detailing how each approximation is calculated to within rounding error of the exact values.

Initial Approximation

Given a set of constant masses $\boldsymbol{\rho}$ from the initial condition, we can use equation (5.53) to provide an initial approximation $U^0(x)$ such that

$$\widetilde{A}^0\mathbf{U}^0 = \tilde{\boldsymbol{\rho}}, \tag{5.54}$$

where the superscript 0 indicates the initial values of the quantity considered.

Remark. *Since* (5.54) *is derived from*

$$\int_{a(s^0)}^{b(s^0)} \tilde{\chi}_i(U^0(x) - u^0(x))\,\mathrm{d}x = 0, \qquad \text{for } i = 1, \ldots, \hat{N} - 2,$$

using the $\tilde{\rho}_i$ *values, it follows that the initial approximation* $U^0(x)$ *is identical to the similarity solution at time* s^0 *to within rounding error since the self similar solution* $u^S(x,s) \in \mathcal{S}_E^1(s)$*, the space of quartic functions.*

Obtaining $Q(x, s^0)$

Given the initial approximation $U^0(x)$, we obtain the coefficients Q_j^0 in (5.32) using equation (5.35):

$$M^0\mathbf{Q}^0 = K^0\mathbf{U}^0.$$

Remark. *The exact* $q(x,s)$ *function is a quadratic in the case when* $n = 1$*. The approximation* $Q^0(x)$ *to* $q^0(x)$ *calculated in §5.6.4 is also quadratic. Since the initial function* $U^0(x)$ *used in calculating* $Q^0(x)$ *is identical to the similarity solution to within rounding error,* $Q^0(x)$ *is also exact within rounding error.*

Obtaining $Z(x, s^0)$

Given $Q^0(x)$ and $U^0(x)$, we obtain $Z^0(x)$ by solving the matrix system (5.39):

$$\widetilde{K}(U^0)\mathbf{Z}^0 = -\widetilde{K}(U^0)\mathbf{Q}^0,$$

giving the coefficients Z_j^0 in (5.38).

Remark. *The exact velocity potential* $z(x,s)$ *is quadratic and the approximation* $Z^0(x)$ *calculated in §5.6.5 is piecewise quadratic. Hence the approximate velocity potential* $Z^0(x, s^0)$ *will be identical to the exact velocity potential to within rounding error at time* s^0*.*

Calculating the Nodal Velocities

The velocity $V^0(x)$ is calculated using the previously calculated $Z^0(x)$ by solving the matrix system (5.46),

$$M\mathbf{V}^0 = B^0\mathbf{Z}^0,$$

giving the coefficients V_j^0 in (5.44).

Remark. *In the case when $n = 1$ the exact velocity function $v^0(x)$ is linear and the approximation $V^0(x)$ calculated in §5.6.6 is piecewise linear. Hence the approximate velocity $V^0(x, s^0)$ will be identical to the linear similarity velocity to within rounding error at time s^0.*

Time Stepping

Time stepping is carried out using (5.47),

$$\mathbf{X}^1 = \mathbf{X}^0 + 5s^4\Delta s\mathbf{V}^0,$$

Remark. *The scheme* (5.47) *has a truncation error proportional to $\frac{\mathrm{d}^2x}{\mathrm{d}s^2}$. Hence in the case when $n = 1$ (for which $v(x, s)$ is linear in s) $\mathbf{X}^1$ is calculated exactly.*

Recovery of the Solution

The solution $U(x, s)$ is recovered at time s^1 using (5.53),

$$\widetilde{A}\mathbf{U} = \tilde{\boldsymbol{\rho}}.$$

Since $u^S(x, s) \in \mathcal{S}_E^1(s)$, we have that

$$\tilde{\rho}_i = \int_{a(s)}^{b(s)} \tilde{\chi}_i(x, s)u^S(x, s)\,\mathrm{d}x, \qquad \text{for } i = 1, \ldots, \hat{N} - 2, \tag{5.55}$$

holds for any time s by conservation of mass.

At time s^1, from the recovery step

$$\int_{a(s^1)}^{b(s^1)} \tilde{\chi}_i(x, s^1)U(x, s^1)\,\mathrm{d}x = \tilde{\rho}_i, \qquad \text{for } i = 1, \ldots, \hat{N} - 2,$$

which implies from (5.55) that

$$\int_{a(s^1)}^{b(s^1)} \tilde{\chi}_i(x, s^1)(U(x, s^1) - u^S(x, s^1))\,\mathrm{d}x = 0, \qquad \text{for } i = 1, \ldots, \hat{N} - 2,$$

and hence that $U(x, s^1)$ is the L^2 best fit to the similarity solution over one time step.

Remark. *In practice the best fit of the solution is preserved only to within rounding error. The arguments in this section are not a proof that this implementation of the FEMPCM possesses the* S Property*, but by choosing our approximations to lie in the appropriate space matching the order of the function we wish to approximate, the method should compute an approximation to within rounding error over a single time step.*

Subsequent time steps

Once this result is established at $s = s^1$ the argument can be repeated for any subsequent time step.

5.7 Numerical Results

We now present numerical results obtained from the FEMPCM described in §5.3–§5.6, which support the arguments made in §5.6.9. The FEMPCM was implemented based on the steps in this chapter.

The initial condition $u^0(x)$ for the method is chosen to be the similarity solution from (2.28),

$$u^S(x, s) = \frac{1}{120s}\left[4 - \xi^2\right]_+^2, \qquad \xi = \frac{x}{s}, \tag{5.56}$$

written in terms of the scaled time variable $s = (t + \kappa)^{1/5}$. From (5.56), we know the exact values of $q^S(x, s)$ and $v^S(x, s)$ (repeated from (4.21)):

$$q^S(x, s) = \frac{2}{15s^3} - \frac{x^2}{10s^5}, \qquad v^S(x, s) = \frac{x}{5s^5}.$$

5.7.1 Single Time Step

We run the implementation of the FEMPCM for a single time step in order to demonstrate that the method possesses the *S Property* in the L^2 norm.

We run the method on an initially equally spaced mesh of $N = 21$ nodes. We use the similarity solution (4.20) to determine the initial approximation $\mathbf{U}^0$ at the initial time $s^0 = 1$, such that

$$u^0(x) = \frac{1}{120}\left[4 - x^2\right]_+^2,$$

for $i = 1, \ldots, N$. This initial approximation is centred about $x = 0$ with initial boundary positions at $x = \pm 2$. The initial approximation $\mathbf{U}^0$ is found by solving (5.51).

A single time step of size $\Delta s = 2.5 \times 10^{-5}$ is performed using the scale-invariant Forward Euler scheme (4.13). The value of Δs is chosen such that $\Delta s = O(1/N^4)$ in order to reduce the risk of node tangling occurring.

After the single time step has been performed, we calculate the absolute error of the coefficients in the approximations $U(x, s^1)$, $Q(x, s^0)$ and $V(x, s^0)$. We evaluate the error in the coefficients as this determines whether the method is able to approximate the nodal values to within rounding error as expected.

We also calculate the absolute error in the position of the updated mesh nodes $\mathbf{X}^1$. The error in the mesh positions is calculated using the similarity variable ξ from (2.22), using the initial values

$$\xi = \frac{x(s^k)}{s^k} \qquad \forall k,$$
$$\Rightarrow \quad |\mathbf{X}^k - x(s^k)| = \left|\mathbf{X}^k - s^k \frac{x^0}{s^0}\right|, \qquad \forall k.$$

Figure 5.8 details the absolute errors. We see that the approximations are in the region of rounding error, with the error in $\mathbf{U}^1$ being of $O(10^{-16})$, the error in $\mathbf{Q}^0$ is of $O(10^{-13})$ and the error in $\mathbf{V}^0$ is $O(10^{-13})$. Finally, we observe the absolute error in the mesh positions is of $O(10^{-16})$. These errors are in the expected regions, so we are confident that the method possesses the *S Property* over this single time step.

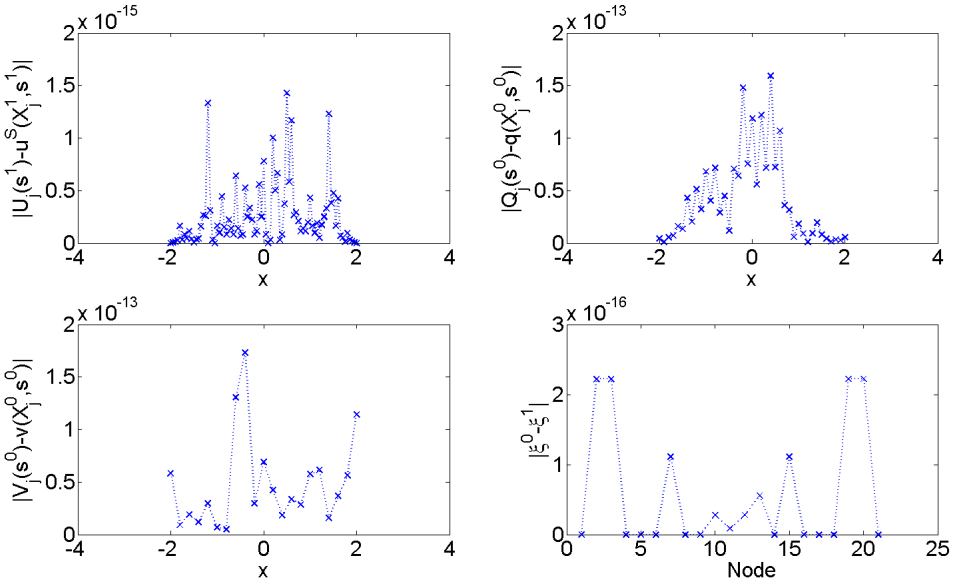

Figure 5.8: Absolute error in the FEMPCM over a single time step. The top left window contains the absolute error in $\mathbf{U}^1$ ($O(10^{-15})$), the top right window the error in $\mathbf{Q}^0$ ($O(10^{-13})$), bottom left the error in $\mathbf{V}^0$ ($O(10^{-13})$) and bottom right the error in the mesh positions ($O(10^{-16})$).

5.7.2 Multiple Time Steps

At the end of each time step the approximate solution $U(x, s)$ obtained by the method is compared with the similarity solution $u^S(x, s)$. We calculate the L^2 error using the solution values at each of the points of the domain. From §5.6.9 we would expect the error in the solution to be close to rounding error.

We run the method on a mesh of $N = 21$ nodes over the time window $s \in [1, 1.25]$ in order to demonstrate how the similarity solution evolves. The time step size is $\Delta s = 2.5 \times 10^{-5}$, as discussed for the single time step results above.

Figure 5.9 details this evolution of the solution calculated from the initial condition at s^0 through to the final solution profile at time $s = 1.25$. We see that the free boundaries expand outwards throughout the time window. The velocity of the boundary is highest near the beginning of the time window, and becomes slower as time increases. The maximum value of the solution decreases in line with these velocity change, due to the enforcement of conservation of mass.

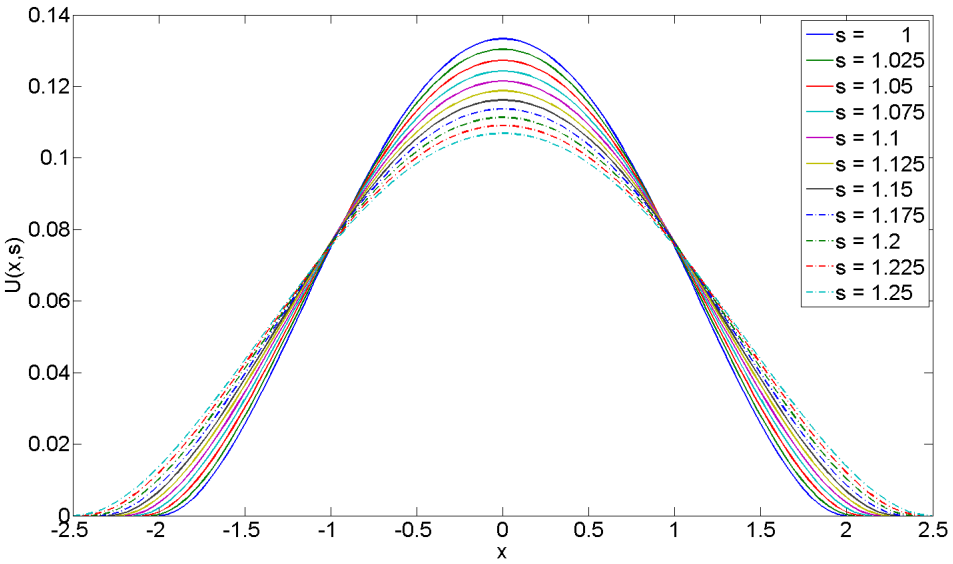

Figure 5.9: Evolution of the solution for the FEMPCM described in §5.6 for $s \in [1, 1.25]$ using a mesh of $N = 21$ nodes.

Note that if we were to continue running the FEMPCM over a longer time window we would expect this behaviour to continue. The solution profile would become shallower as the boundary expands outwards, in keeping with the conservation of mass principle.

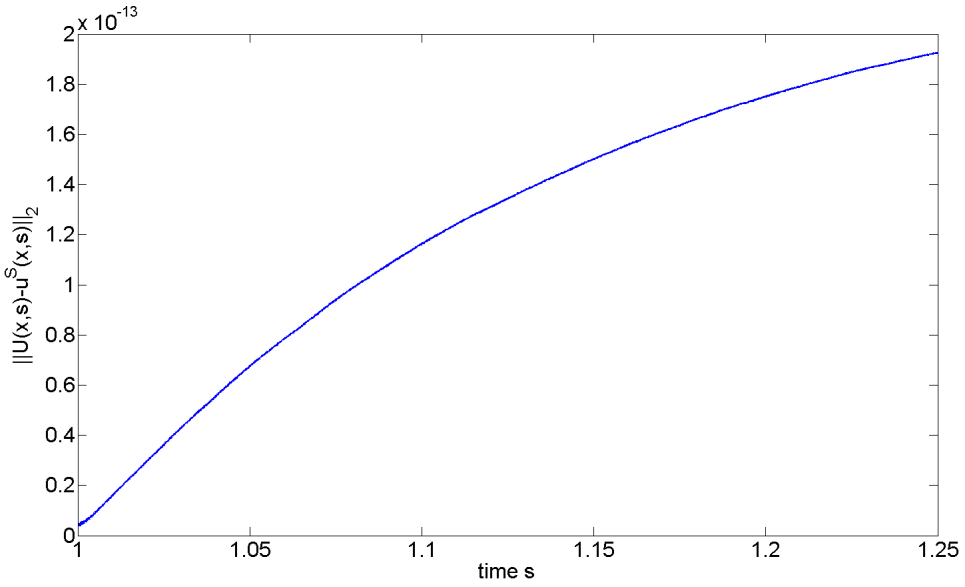

Figure 5.10: L^2 error in the solution for the quartic implementation of the FEMPCM, plotting over the time window $s \in [1, 1.25]$ on a mesh of 21 Nodes. Scale of y-axis on plot is 10^{-13}.

We next observe the error incurred by the method in calculating the solution $U(x, s)$ shown in Figure 5.9. Figure 5.10 shows how the L^2 error in the solution evolves over the time period we have used. It is observed that while the error is initially in the region of rounding error, it increases over the time window, reaching approximately 1.9×10^{-13} by the end of the time window. This behaviour matches that observed in Chapter 4 (see §4.3.6) for the implementation of the MPCM using finite differences for the same problem. This is likely an accumulation of numerical rounding error, as discussed in §4.4.

The L^2 error in the approximations $Q(x, s)$ and $V(x, s)$ for this experiment are shown in Figure 5.11. From the single time step results (Figure 5.8), the initial error in these

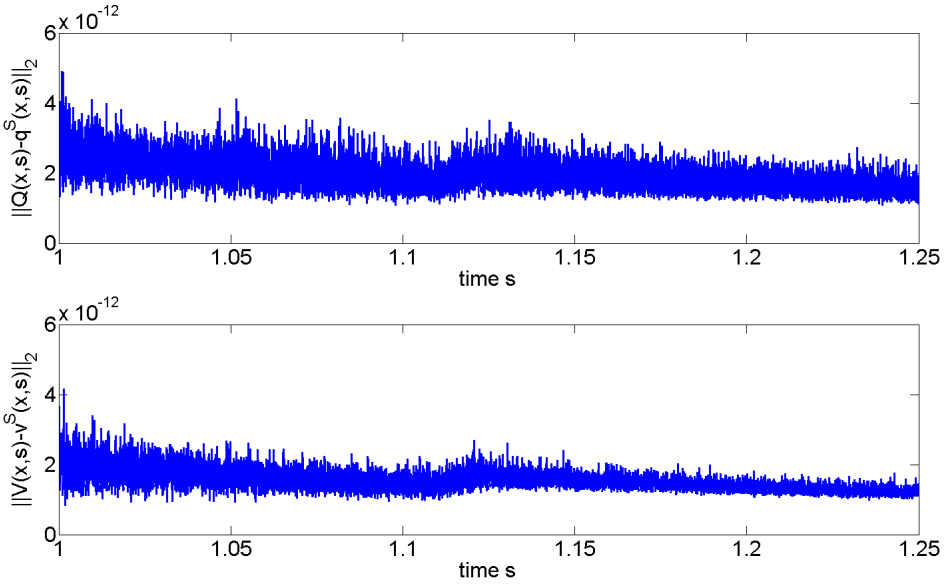

Figure 5.11: L^2 error in $Q(x,s)$ (top plot) and $V(x,s)$ (bottom plot) for the quartic implementation of the FEMPCM, plotting over the time window $s \in [1, 1.25]$ on a mesh of 21 Nodes. Scale of y-axis on both plots is 10^{-12}.

approximations is higher than that of $U(x,s)$. Over the course of the time window however, this error does not increase in the same manner as observed in Figure 5.10. We see that the error is approximately 10^{-12} throughout the time window, with the variance decreasing as time progresses. The high oscillation observed in these errors, along with their magnitude suggests that this error is likely numerical and not from the method, which suggests that the *S Property* is approximately maintained over multiple time steps.

5.8 The FEMPCM in Second/Sixth-Order Problems

In chapter 4 the MPCM was shown to be able to possess the *S Property* in the l^∞ norm, not only for the fourth-order problem (2.3)–(2.5) but also for the second (2.7)–(2.9) and sixth-order (2.10)–(2.12) problems. It is also possible to produce an implementation of the FEMPCM for second-order and sixth-order problems which possess the *S Property* in the L^2 norm for similarity solutions as for the fourth-order problem (2.3)–(2.5).

In order to try and avoid unnecessary repetition in this section, the steps used in the implementation for each problem will be omitted as the majority of the steps are outlined in detail in §5.6. We shall instead outline the differences between the implementation given in §5.6 and those required for the second or sixth order problems. The main differences occur in the choice of basis function used in each step of the method and the matrix systems which must be solved to obtain the function approximations. As in §5.6, all references to the time variable shall be made in terms of s rather than t due to the use of a scale-invariant time stepping scheme.

5.8.1 An Implementation of the FEMPCM for the Second-Order Problem for any n

The implementation for the second-order problem (2.7)–(2.9) for any n makes use of the piecewise quadratic $\varphi_i(x,s)$ $(i = 1,\ldots,\tilde{N})$ and piecewise linear $\phi_i(x,s)$ $(i = 1,\ldots,N)$ basis functions given in §5.6.2 (written here in terms of s).

The biggest difference between the second and fourth-order implementations is that there is no requirement for the weak form (5.11), since $q = u$ everywhere in the second order case. We therefore do not require the finite dimensional subspace $\mathcal{S}^1(s)$ in this implementation.

We choose our function approximations to lie in the test spaces given by

$$\begin{aligned}
\mathcal{S}_E^1(s) &= \text{span}\{\varphi_j(x,s)\}, \qquad \text{for } j = 2,\ldots,\tilde{N}-1,\\
\widetilde{\mathcal{S}}^1(s) &= \text{span}\{\varphi_j(x,s)\}, \qquad \text{for } j = 1,\ldots,\tilde{N},\\
\widehat{\mathcal{S}}^1(s) &= \text{span}\{\phi_j(x,s)\}, \qquad \text{for } j = 1,\ldots,N,
\end{aligned}$$

and expand $Z(x,s)$, $V(x,s)$ and $U(x,s)$ as the following linear combinations:

$$\begin{aligned}
Z(x,s) &= \sum_{j=1}^{\tilde{N}} \varphi_j(x,s) Z_j(s),\\
V(x,s) &= \sum_{j=1}^{N} \phi_j(x,s) V_j(s),\\
U(x,s) &= \sum_{j=2}^{\tilde{N}-1} \varphi(x,s) U_j(s).
\end{aligned}$$

Obtaining the Velocity Potential $Z(x,s)$

The coefficients $Z_j(s)$ of the velocity potential $Z(x,s)$ are obtained by solving the matrix system

$$\widetilde{K}(U)\mathbf{Z} = -\widetilde{K}(U^n)\mathbf{Q},$$

where $\widetilde{K}(U)$ is of dimension $\tilde{N}\times\tilde{N}$ and has entries given by (5.40). The matrix $\widetilde{K}(U^n)$ has entries given by (5.40), with U replaced by U^n. A constraint such as (5.24) is required to uniquely solve the matrix system, as $\widetilde{K}(U)$ is a singular matrix (see Remark 5.1).

Obtaining the Velocity $V(x,s)$

The coefficients of $Z(x,t)$ can be obtained by solving the matrix system

$$M\mathbf{V} = B\mathbf{Z},$$

which is the same system as described in §5.6 for obtaining $\mathbf{V}$ from $\mathbf{Z}$.

Time Stepping

Time stepping is performed using the scale-invariant time stepping scheme (5.47),

$$X_i^{k+1} = X_i^k + \frac{1}{\beta} s^{1/\beta - 1} \Delta s V_i^k, \qquad \text{for } i = 1, \ldots, N,$$

where $\beta = (n+2)^{-1}$ in the second-order problem (2.7)–(2.9).

Recovery of $U(x,s)$ on the Updated Mesh

Solution recovery for the second-order problem (2.7)–(2.9) involves the construction of modified basis functions $\tilde{\varphi}_i(x,s)$ and $\tilde{\rho}_i(x,s)$, both for $i = 1, \ldots, \tilde{N}-2$. These quantities are constructed in the same manner as the $\tilde{\chi}_i$ and $\tilde{\rho}_i$ described in §5.6.8.

The coefficients $U_j(s)$ for $j = 2, \ldots, \tilde{N}-1$ are then found by solving the matrix system

$$\widetilde{A}\mathbf{U} = \tilde{\boldsymbol{\rho}},$$

where $\widetilde{A}$ is a mass matrix of dimension $\tilde{N}-2 \times \tilde{N}-2$ with time-dependent entries given by

$$\widetilde{A}_{ij}(s^k) = \int_{a(s)}^{b(s)} \tilde{\varphi}_i(x,s)\tilde{\varphi}_j(x,s)\,\mathrm{d}x, \qquad \text{for } i,j = 1, \ldots, \tilde{N}-2.$$

and $\tilde{\boldsymbol{\rho}} = \{\tilde{\rho}_i\}$, for $i = 1, \ldots, \tilde{N}-2$.

Implications

By implementing the method described above, the method can be shown to possess the *S Property* in the L^2 norm for **any** value of n.

5.8.2 An Implementation of the FEMPCM for the Sixth-Order Problem with $n = 1$

The process of producing an implementation of the FEMPCM for the sixth-order problem (2.10)–(2.12) is more complex than that of the fourth-order problem (2.3)–(2.5) due

to the addition of the $p(x,t)$ function and the need to use higher order spaces in the method. Many of the steps are similar to those detailed in §5.6, but using different basis functions in the various finite element approximations.

Weak Forms

We shall make use of the weak forms (5.13), (5.12), (5.36), (5.41) and the weighted conservation principle (5.6), all of which we state here for clarity:

$$\int_{a(s)}^{b(s)} w(x,s)p(x,s)\,\mathrm{d}x = \int_{a(s)}^{b(s)} \frac{\partial w}{\partial x}\frac{\partial u}{\partial x}\,\mathrm{d}x, \qquad \forall w \in W,$$

$$\int_{a(s)}^{b(s)} w(x,s)q(x,s)\,\mathrm{d}x = \int_{a(s)}^{b(s)} \frac{\partial w}{\partial x}\frac{\partial p}{\partial x}\,\mathrm{d}x, \qquad \forall w \in W,$$

$$\int_{a(s)}^{b(s)} u\frac{\partial w}{\partial x}\left(\frac{\partial z}{\partial x} + \frac{\partial q}{\partial x}\right)\,\mathrm{d}x = 0, \qquad \forall w \in W,$$

$$\int_{a(s)}^{b(s)} w(x,s)\left(v(x,s) - \frac{\partial z}{\partial x}\right)\,\mathrm{d}x = 0, \qquad \forall w \in W,$$

$$\int_{a(s)}^{b(s)} w(x,s)u(x,s)\,\mathrm{d}x = \rho(w), \qquad \forall w \in W.$$

Finite Dimensional Subspaces

We next select the finite dimensional subspaces $\mathcal{S}_E^1(s), \acute{\mathcal{S}}^1(s), \widehat{\mathcal{S}}^1(s), \mathcal{S}^1(s), \widetilde{\mathcal{S}}^1(s) \in W$. The basis functions for use in this implementation will be the piecewise linear $\phi_i(x,s)$ $(i = 1,\ldots,N)$, piecewise quadratic $\varphi_i(x,s)$ $(i = 1,\ldots,\tilde{N})$, piecewise quartic $\chi_i(x,s)$ $(i = 1,\ldots,\widehat{N})$ and we introduce piecewise sextic $\psi_i(x,s)$ $(i = 1,\ldots,\acute{N})$ functions for the sixth-order problem (2.10)–(2.12), where we have introduced $\acute{N} = 6N-5$, $\widehat{N} = 4N-3$ and $\tilde{N} = 2N-1$ for ease of notation.

We then set

$$\begin{aligned}
\mathcal{S}_E^1(s) &= \text{span}\{\psi_j(x,s)\}, && \text{for } j = 2,\ldots,\acute{N}-1,\\
\acute{\mathcal{S}}^1(s) &= \text{span}\{\psi_j(x,s)\}, && \text{for } j = 1,\ldots,\acute{N},\\
\widetilde{\mathcal{S}}^1(s) &= \text{span}\{\chi_j(x,s)\}, && \text{for } j = 1,\ldots,\widehat{N},\\
\mathcal{S}^1(s) &= \text{span}\{\varphi_j(x,s)\}, && \text{for } j = 2,\ldots,\tilde{N}-1,\\
\widehat{\mathcal{S}}^1(s) &= \text{span}\{\phi_j(x,s)\}, && \text{for } j = 1,\ldots,N.
\end{aligned}$$

Finite Element Approximations

From the weak forms above, we obtain the following finite element approximations:

For a given $U(x,s) \in \acute{\mathcal{S}}^1(s)$ we seek the finite element approximation $P(x,s)$ in the form: Find $P \in \widetilde{\mathcal{S}}^1(s)$ such that

$$\int_{a(s)}^{b(s)} w(x,s)P(x,s)\,\mathrm{d}x = \int_{a(s)}^{b(s)} \frac{\partial w}{\partial x}\frac{\partial U}{\partial x}\,\mathrm{d}x, \qquad \forall w \in \widetilde{\mathcal{S}}^1(s). \tag{5.61}$$

We then seek the finite element approximation $Q(x,s)$ for a given $P(x,s) \in \widetilde{\mathcal{S}}^1(s)$ in the form: Find $Q(x,s) \in \mathcal{S}^1(s)$ such that

$$\int_{a(s)}^{b(s)} w(x,s)Q(x,s)\,\mathrm{d}x = \int_{a(s)}^{b(s)} \frac{\partial w}{\partial x}\frac{\partial P}{\partial x}\,\mathrm{d}x \qquad \forall w \in \mathcal{S}^1(s). \tag{5.62}$$

The finite element approximation $Z(x,s)$ for a given $Q(x,s) \in \mathcal{S}^1(s)$, $U(x,s) \in \acute{\mathcal{S}}^1(s)$ is: Find $Z(x,s) \in \mathcal{S}^1(s)$ such that

$$\int_{a(s)}^{b(s)} U(x,s)\frac{\partial w}{\partial x}\left[Z(x,s) + \frac{\partial Q}{\partial x}\right]\,\mathrm{d}x = 0, \qquad \forall w \in \mathcal{S}^1(s).$$

The finite element approximation $V(x,s)$ for a given $Z(x,s) \in \mathcal{S}^1(s)$ is of the form: Find $V(x,s) \in \widehat{\mathcal{S}}^1(s)$ such that

$$\int_{a(s)}^{b(s)} w(x,s)\left[V(x,s) - \frac{\partial Z}{\partial x}\right]\,\mathrm{d}x = 0, \qquad \forall w \in \widehat{\mathcal{S}}^1(s).$$

Once the mesh points have been updated using the scale-invariant time stepping scheme (4.13), we recover $U(x,s)$ on the updated mesh by seeking $U \in \mathcal{S}_E^1(s)$ such that

$$\int_{a(s)}^{b(s)} w(x,s)U(x,s)\,\mathrm{d}x = \rho(w) \qquad \forall w \in \acute{\mathcal{S}}^1(s).$$

Obtaining $P(x,s)$

For a given $U(x,s)$, we obtain the approximation $P(x,s)$. We choose the $w(x,s)$ in (5.61) to be one of the $\chi_i(x,s)$ functions ($i = 1,\ldots,\widehat{N}$). This leads to a system of $\widehat{N}$ equations

$$\begin{aligned}
\int_{a(s)}^{b(s)} \chi_1(x,s)P(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \chi_1'(x,s)U'(x,s)\,\mathrm{d}x, \qquad \text{for } i = 1,\\
\int_{a(s)}^{b(s)} \chi_i(x,s)P(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \chi_i'(x,s)U'(x,s)\,\mathrm{d}x, \qquad \text{for } i = 2,\ldots,\widehat{N}-1,\\
\int_{a(s)}^{b(s)} \chi_{\widehat{N}}(x,s)P(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \chi_{\widehat{N}}'(x,s)U'(x,s)\,\mathrm{d}x, \qquad \text{for } i = \widehat{N}.
\end{aligned}$$

Writing $P(x,s)$ as the linear combination of the $\chi_j(x,s)$,

$$P(x,s) = \sum_{j=1}^{\widehat{N}} \chi_j(x,s)P_j(s),$$

and $U(x,s)$ as the linear combination of the $\psi_j(x,s)$,

$$U(x,s) = \sum_{j=2}^{\acute{N}-1} \psi_j(x,s)U_j(s),$$

we obtain a system of equations from which the coefficients $P_j(s)$ can be obtained through solving the matrix system

$$M\mathbf{P} = K\mathbf{U},$$

where M is a tridiagonal matrix of dimension $\widehat{N}\times\widehat{N}$ and K is a matrix of dimension $\widehat{N}\times\acute{N}-2$.

Obtaining $Q(x,s)$

Once we have obtained $P(x,s)$, we next seek the approximation $Q(x,s)$. We choose the $w(x,s)$ in (5.62) to be one of the $\varphi_i(x,s)$ functions ($i = 1,\ldots,\tilde{N}$). This leads to a

system of $\tilde{N}$ equations

$$\begin{aligned}
\int_{a(s)}^{b(s)} \varphi_1(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_1'(x,s)P'(x,s)\,\mathrm{d}x, \qquad \text{for } i=1,\\
\int_{a(s)}^{b(s)} \varphi_i(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_i'(x,s)P'(x,s)\,\mathrm{d}x, \qquad \text{for } i=2,\ldots,\tilde{N}-1,\\
\int_{a(s)}^{b(s)} \varphi_{\tilde{N}}(x,s)Q(x,s)\,\mathrm{d}x &= \int_{a(s)}^{b(s)} \varphi_{\tilde{N}}'(x,s)P'(x,s)\,\mathrm{d}x, \qquad \text{for } i=\tilde{N}.
\end{aligned}$$

Writing $Q(x,s)$ as the linear combination of the $\varphi_j(x,s)$,

$$Q(x,s) = \sum_{j=1}^{\tilde{N}} \varphi_j(x,s)Q_j(s),$$

we obtain a system of equations from which the coefficients $Q_j(s)$ can be obtained through solving the matrix system

$$M\mathbf{Q} = K\mathbf{P},$$

where M is a tridiagonal matrix of dimension $\tilde{N}\times\tilde{N}$ and K is a matrix of dimension $\tilde{N}\times\widehat{N}$.

Obtaining $Z(x,s)$ and $V(x,s)$

The steps used to obtain the approximation $Z(x,s)$ to $z(x,s)$ and consequently $V(x,s)$ are exactly the same as described in §5.6. The details are not repeated here in full, but we may obtain the coefficients of $Z(x,s)$ by solving the matrix system

$$\widetilde{K}(U)\mathbf{Z} = \widetilde{K}(U)\mathbf{Q},$$

with $\widetilde{K}(U)$ having entries given by (5.40), and then obtain the coefficients of $V(x,s)$ by solving

$$M\mathbf{V} = B\mathbf{Z},$$

as described in §5.6.

Updating Mesh Positions

The mesh positions are then updated the new positions $\mathbf{X}^{k+1}$ using the scale-invariant time stepping scheme,

$$X_i^{k+1} = X_i^k + 7s^6 \Delta s V_i^k, \qquad \text{for } i = 1, \ldots, N,$$

which differs from (5.47) since $\beta = 1/7$ in the sixth-order problem (2.10)–(2.12) for $n = 1$.

Recovery of $U(x, s)$

We next recover $U(x, s)$ on the updated mesh positions using the distributed conservation of mass principle.

We introduce modified basis functions $\tilde{\psi}_i(x, s)$ and modified constants $\tilde{\rho}_i$ (both for $i = 1, \ldots, \acute{N} - 2$), obtained using the same method as in §5.6.8. The coefficients $U_j(s)$ may then be obtained by solving the matrix system

$$\widetilde{A}\mathbf{U} = \tilde{\boldsymbol{\rho}},$$

where $\widetilde{A}$ is a matrix of dimension $\acute{N} - 2 \times \acute{N} - 2$ and $\tilde{\boldsymbol{\rho}} = \{\tilde{\rho}_i\}$ $(i = 1, \ldots, \acute{N} - 2)$.

5.9 Summary of this Chapter

In this chapter we have introduced the FEMPCM for the fourth-order problem (2.3)–(2.5). A general implementation for the fourth-order problem (2.3)–(2.5) has been described which uses piecewise linear basis functions throughout.

It has then been shown that by careful choice of basis functions in the implementation it is possible to create an implementation of the FEMPCM for the fourth-order problem (2.3)–(2.5) with $n = 1$ which possesses the *S Property* in the L^2 norm.

Finally, we have outlined the choice of basis functions required in the implementation of the FEMPCM for second-order (2.7)–(2.9) (any n) and sixth-order (2.10)–(2.12) ($n = 1$) problems such that the *S Property* is possessed by the method.

Numerical results for the second-order and sixth-order cases have not been produced as a working code for general n has not at present been produced. The outlines of the FEMPCM for these problems is therefore included as a guide on how such methods would be implemented.

In the next chapter we shall explore cases where the MPCM or FEMPCM is not expected to possess the *S Property*. We focus on the fourth-order problem (2.3)–(2.5) with $n > 1$ and discuss possible initial boundary behaviours and whether the MPCM is able to accurately model the different types of behaviour.

Chapter 6

The MPCM in the Fourth-Order Problem with $n > 1$

6.1 Aims of this Chapter

In this chapter we shall examine the fourth-order problem in cases where $n > 1$. In particular we shall examine the initial and small-time behaviour of the boundary of the domain for various initial conditions and choices of n, since there is a wide range of behaviours which can be exhibited.

We shall investigate whether the MPCM as described in Chapters 3 and 4 is able to model these various behaviours and in cases where it fails explore alternatives. The presence of singularities at the boundary of the domain will be shown to have a drastic effect on the success of the MPCM as described in previous chapters. We also explore whether the difficulties relating to the implementation of the MPCM are present in the FEMPCM.

A hybrid numerical method is then proposed, consisting of a fixed-mesh method combined with the MPCM, which is able to model in principle the movement of the boundary of the problem in cases which the standard MPCM is not. The steps required to implement the method are discussed in detail, along with a critique of the method in the absence of satisfactory numerical results. We then propose an alternative way

of expressing the velocity, in which the fourth-order PDE is not written as two second-order equations, for use in a modified version of the MPCM. This alternative velocity avoids the need to approximate the $q(x,t)$ function, as this function is a major cause of the numerical issues in the MPCM.

6.2 The 4th Order Problem with $n > 1$

In this chapter we shall mainly focus on the fourth-order problem with $n > 1$ in (2.3a), written as two second-order equations, which we restate here for clarity. We seek a solution $u(x,t)$ to the 1D fourth-order nonlinear diffusion equation with $n > 1$,

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), \\ q &= -\frac{\partial^2 u}{\partial x^2}, \end{aligned} \tag{6.1}$$

for $x \in \Omega_T$, subject to an initial condition at time $t = t^0$ given by

$$u(x,t_0) = u^0(x), \qquad \text{for } x \in \Omega(t^0),$$

and boundary conditions

$$\begin{aligned} u &= 0, \\ \frac{\partial u}{\partial x} &= 0, \\ uv + u^n \frac{\partial q}{\partial x} &= 0, \end{aligned}$$

at the moving boundaries $x = a(t)$ and $x = b(t)$, for $t > t^0$.

As we are investigating the fourth-order problem with $n > 1$ and with no explicit similarity solutions available, we shall focus on the qualitative behaviour of the approximate solutions generated by the MPCM. In chapter 3 we observed that in the case where $n = 1$ and the initial condition corresponds to a similarity solution, the velocity $v(x,t)$ is a linear function in x given by

$$v(x,t) = \frac{\beta x}{t},$$

with $\beta = 1/5$. This linear velocity causes the domain to expand outwards uniformly in space as time progresses, as observed in the numerical results from the MPCM (see Figure 5.9, for example). In the case when $n > 1$ however (or for $n = 1$ when the initial conditions do not correspond to a similarity solution), this outwards behaviour may not occur and a range of behaviours may be observed depending upon the values of n and the initial condition used (see §2.2.2).

Of special interest is the velocity of the boundary points at $x = a(t)$ and $x = b(t)$, since this velocity directly determines the behaviour of the boundary.

6.3 Boundary Velocities when $n > 1$

Using the local conservation of mass principle defined in §3.2.5, we have an expression (3.14) for use in obtaining the velocity at points $\hat{x}(t) \in \Omega(t)$ for which $u \neq 0$, which we restate here for completeness:

$$v = -\left(u^{n-1}\frac{\partial q}{\partial x}\right)\Bigg|_{x=\hat{x}(t)}. \tag{6.2}$$

To determine the velocity at points where $u = 0$, we assume that $v(x,t)$ is continuous and treat (6.2) as a limiting expression. Taking the right-hand boundary $b(t)$ of $\Omega(t)$ as an example, we have that the velocity at the boundary, v_b, is

$$v_b = \lim_{x \to b(t)^-} \left(-u^{n-1}\frac{\partial q}{\partial x}\right), \tag{6.3}$$

provided that the limit exists.

Remark. *In the implementation of the MPCM described in §3.3, we approximated the limiting expression* (6.3) *through an extrapolation using nearby points. This was necessary due to the strong enforcement of* $u = 0$ *at the boundary. If* (6.2) *had been used in the finite difference implementation for* $n > 1$*, the boundary velocity would have been set to zero regardless of any finite value of* $\frac{\partial q}{\partial x}$ *at the boundary. By extrapolating from interior points we were able to obtain a non-zero velocity at the boundary.*

6.4 Initial Boundary Velocities

In this section we shall focus on the initial time $t = t^0$ and examine the initial boundary velocity for $n > 1$. We consider a particular family of initial conditions and determine under what circumstances the limit (6.3) exists.

We shall consider initial conditions of the form

$$u^0(x) = c_1(\omega^2 - x^2)^\alpha, \tag{6.4}$$

for some constant $\alpha > 0$, where c_1 and ω are arbitrary constants. In this case the initial boundary positions are located at the points $x = \pm\omega$, so $a(t^0) = -\omega$ and $b(t^0) = \omega$, giving (together with the symmetry of the PDE (6.1)) symmetric solutions about the point $x = 0$.

We wish to consider the different possible asymptotic behaviours of the moving boundaries for different choices of α in (6.4) and n in (6.1), as outlined in King (2001) and Blowey et al. (2007). From this point on we shall focus our discussion on the asymptotic behaviour at the right-hand boundary at $x = \omega$, although the same arguments apply for the left-hand boundary with a little adjustment.

For $x \approx \omega$ we have, from (6.4),

$$u^0(x) \approx c_1(2\omega)^\alpha(\omega - x)^\alpha. \tag{6.5}$$

Using (6.4) and (6.2) we can calculate the initial velocity $v^0(x)$ to be

$$\begin{aligned} v^0(x) &= 12c_1^n\alpha(\alpha-1)x(\omega^2-x^2)^{n\alpha-2} - 8c_1^n\alpha(\alpha-1)(\alpha-2)x^3(\omega^2-x^2)^{n\alpha-3}, \\ &\approx c_2\alpha(\alpha-1)(\omega-x)^{n\alpha-2} - c_3\alpha(\alpha-1)(\alpha-2)(\omega-x)^{n\alpha-3}, \end{aligned} \tag{6.6}$$

for $x \approx \omega$, where c_2 and c_3 are

$$c_2 \approx 12c_1^n\omega(2\omega)^{n\alpha-2}, \qquad c_3 \approx c_1^n(2\omega)^{n\alpha}. \tag{6.7}$$

The boundary velocity (6.3) at the initial time (v_b^0, say) is therefore

$$v_b^0 = c_2\alpha(\alpha-1)\lim_{x\to\omega}\left((\omega-x)^{n\alpha-2}\right) - c_3\alpha(\alpha-1)(\alpha-2)\lim_{x\to\omega}\left((\omega-x)^{n\alpha-3}\right), \tag{6.8}$$

and the moving boundary may experience different behaviours depending on the value of $n\alpha$, as follows.

6.4.1 $n\alpha = 3$

If n and α are such that $n\alpha = 3$, then the limit on the right-hand side of (6.8) is finite, with the limiting velocity

$$v_b^0 = -c_3\alpha(\alpha - 1)(\alpha - 2).$$

Hence the limiting boundary velocity (using c_3 from (6.7) at $x = \omega$) is

$$v_b^0 = -c_1^n(2\omega)^{n\alpha}\alpha(\alpha - 1)(\alpha - 2), \tag{6.9}$$

which is non-zero if $\alpha \neq 1, 2$. The direction of motion of the moving boundary is then dependent on the sign of c_1 and the value of α. For example, if $c_1 > 0$ in (6.4) then from (6.9) we have a positive velocity for $1 < \alpha < 2$ and a negative velocity for $\alpha < 1$, $\alpha > 2$.

It is worth noting that the values $\alpha = 1$ and $\alpha = 2$ correspond to points at which the finite velocity (6.9) changes sign from positive to negative or vice versa.

6.4.2 $n\alpha > 3$

In the case where n and α are such that $n\alpha > 3$, the limit on the right-hand side of (6.8) (and therefore v_b) is zero as $x \to \omega$. In this case the boundary is initially stationary.

6.4.3 $n\alpha < 3$

In this case the limit on the right-hand side of (6.8) does not exist. In this instance the boundary will move with an initially unbounded velocity. It is known however, that the velocity is only unbounded instantaneously at the initial time, after which the finite speed of propagation property possessed by the PDE results in a finite velocity (see Bernis (1996b) and Bernis (1996a)).

This result is supported by asymptotic results presented in Blowey et al. (2007) (following work in King (2001)), in which a region in (n, α) space was identified in which the initial velocity of the boundary point was unbounded. In that paper, for values of $n\alpha < 3$ the boundary was expected to advance (for $\alpha < 2$) or retreat (for $\alpha > 2$) with an initially unbounded velocity.

Since the MPCM is a finite difference method requiring a finite velocity in order to update the boundary points, an unbounded velocity (even one existing instantaneously) cannot be calculated at $t = t^0$. Indeed, any finite difference method that attempts to compute the boundary velocity in this pointwise manner will ultimately fail.

6.5 Small-Time Behaviours of the Boundary

The small-time behaviour of the fourth-order problem with initial conditions of the form (6.4) has been detailed in Blowey et al. (2007). Their results for the initial boundary velocity agree with the statements made in §6.4, so we shall provide a summary of the main results in Blowey et al. (2007) in this section to provide insight into the behaviour of the solution after the initial time t^0.

6.5.1 Initial Behaviour

For $n\alpha > 3$ the initial velocity of the boundary is equal to zero, with $n\alpha > 4$ providing global waiting time (with the local behaviour of the solution at points close to the boundary remaining unchanged) for some non-zero time t^w.

When $3 < n\alpha < 4$ the type of waiting-time behaviour is dependent upon α, with $\alpha = 2$ forming a delicate point in the analysis presented in the paper. When $\alpha < 2$ the velocity at the boundary tends to zero as $t \to 0^+$, while for $\alpha > 2$ a range of different waiting-time scenarios can be observed (we refer the reader to the paper for further details on these scenarios).

Values of $n\alpha < 3$ cause the boundary to move with an initially unbounded velocity, with the direction of motion determined by the value of α. The case when $\alpha < 2$ corresponds to an outward moving boundary, while $\alpha > 2$ corresponds to an inwardly moving boundary ($\alpha = 2$ is again a delicate case).

When $n\alpha = 3$ the velocity of the boundary is finite, as discussed in §6.4, with the direction of travel again determined by α as in the $n\alpha < 3$ regime above.

6.5.2 Evolution of the Solution Close to the Boundary

Once the initial behaviour has occurred, the local behaviour of the solution for $t > t^0$ is dependent upon the value of n and is discussed in King (2001). When $n\alpha \leq 3$ the local behaviour of the solution as $x \to \omega$ (see also (2.6)) takes the form

$$\begin{aligned} u &\sim \left(\frac{n^3 \dot{b}}{3(3-n)(2n-3)} (b(t)-x)^3 \right)^{\frac{1}{n}}, \quad \text{for } \frac{3}{2} < n < 3, \\ u &\sim \left(\frac{3}{4} \dot{b} (b(t)-x)^3 \ln \left[\frac{1}{(b(t)-x)} \right] \right)^{\frac{2}{3}}, \qquad \text{for } n = \frac{3}{2}, \\ u &\sim B(t)(b(t)-x)^2, \qquad \text{for } n < \frac{3}{2}, \end{aligned} \tag{6.10}$$

where $\dot{b}$ denotes the velocity of the boundary $b(t)$ and $B(t)$ must be determined as part of the solution. Here $b(t)$ is the position of the moving boundary, so at time $t = t^0$ we have $b(t^0) = \omega$.

If $n\alpha > 3$ the solution only takes the form (6.10) once any waiting-time has ceased. In some cases the waiting-time ceases due to a shock forming as the gradient of the solution becomes unbounded close to the boundary. The solution may also decrease such that

$$u \sim \left(\frac{n^3(\omega - x)^4}{8(4-n)(2-n)(n+4)t} \right)^{1/n}, \qquad \text{as } x \to \omega,\ t^0 < t < t^w,$$

experiencing various types of decay based upon n and α (see Blowey et al. (2007) for more details).

In the regime $3 < n\alpha < 4$, with $n > 3/2$ the boundary velocity tends to zero as $t \to t^0$, with the boundary waiting. We anticipate that between t^0 and the end of the waiting-time, α decreases from the initial value (greater than $3/n$) until such time as it becomes equal to $3/n$. At this point the solution will take the form (6.10) and the boundary begins to move.

6.6 Initial Boundary Velocities using the MPCM

From the results in §6.4 we expect the MPCM to be able to track the boundaries of the domain only in cases where $n\alpha \geq 3$, since the velocity in these cases is finite or zero.

The case where the boundary velocity is initially unbounded needs special consideration (and is therefore not considered further in this section).

6.6.1 Numerical Verification of Boundary Velocity Issues

We shall now perform experiments using the implementation of the MPCM from Chapter 4 to verify that the issues described in the previous section will occur.

We shall calculate the initial velocity $\mathbf{V}^0$ using (3.24) (see §3.3.4 for more details) using the MPCM described in Chapter 4 on meshes of an increasing number of nodes. We use the initial condition

$$u^0(x) = \frac{1}{10}\left(1 - x^2\right)^\alpha, \tag{6.11}$$

for various α in order to estimate the correct boundary velocities for the finite advance, finite retreat and waiting-time cases described in §6.4.

We perform experiments on meshes of $N = 21, 41, 81, \ldots, 1281$ nodes, using the following choices of n in (6.1) and α in (6.11):

Finite Advance

For the finite advance case we choose $n = 8/5$, $\alpha = 15/8$, giving $n\alpha = 3$ with $1 < \alpha < 2$. In this case we should observe a positive boundary velocity, corresponding to a finite advance of the boundary. The exact value of the initial boundary velocity in this case is equal to

$$v_b^0 = \frac{105}{64}\left(\frac{1}{10}\right)^{8/5} \approx 0.04.$$

We shall also choose $n = 2$, $\alpha = 3/2$, again corresponding to a positive boundary velocity. The exact value of the initial boundary velocity in this case is equal to

$$v_b^0 = 0.03.$$

Finite Retreat

In the finite retreat case we choose $n = 6/5$, $\alpha = 5/2$, giving $n\alpha = 3$ with $\alpha > 2$. In this case we anticipate a negative boundary velocity and a finite retreat of the boundary,

with the initial boundary velocity approximately equal to

$$v_b^0 = -15\left(\frac{1}{10}\right)^{6/5} \approx -0.95.$$

Our second choice for the finite retreat case is $n = 15/11$, $11/5$ also corresponding to a negative boundary velocity. The exact value of the initial boundary velocity in this case is equal to

$$v_b^0 = -\frac{528}{125}\left(\frac{1}{10}\right)^{15/11} \approx -0.18.$$

Waiting-Time

For the waiting-time case we choose $n = 5/2$, $\alpha = 5/2$, and $n = 2$, $\alpha = 2$ giving $n\alpha > 3$ and therefore an expected zero velocity of the boundary in for both choices.

Once $\mathbf{V}^0$ has been calculated we record the computed velocity at the right-hand boundary for the different expected behaviours (advance, retreat or waiting-time). We can then compare this value to the expected value to see whether this velocity is being approximated accurately. We would expect that the computed boundary velocity for the coarsest meshes would be less accurate than that of the finest meshes, but that it would become more accurate as the number of mesh points increases.

Remark 6.1. *We are unable to provide an exact value for the initial boundary velocity without a detailed asymptotic analysis. The computation of a boundary velocity which is approximately equal to the value of v_b^0 will be deemed to be an accurate approximation for the discussion that follows.*

The initial computed boundary velocities V_N^0 for the finite advance experiments are given in Table 6.1. The velocities for the finite retreat experiments are given in Table 6.2 while the waiting-time results are given in Table 6.3.

It is clear from Table 6.1 that in the finite advance experiments the initial velocity is quite inaccurate, with the $n = 8/5$, $\alpha = 15/8$ case having a larger velocity than expected for all values of N and the $n = 2$, $\alpha = 3/2$ case possessing a negative velocity for all values of N. Despite the difference between successive meshes decreasing in all three

N	$n = 8/5$, $\alpha = 15/8$	$n = 2$, $\alpha = 3/2$
21	0.0994	-0.0025
41	0.0930	-0.0036
81	0.0971	-0.0039
161	0.0999	-0.0041
321	0.1015	-0.0042
641	0.1023	-0.0042
1281	0.1027	-0.0042
v_b^0	≈ 0.04	0.03

Table 6.1: Initial boundary velocities for the finite advance experiments in §6.6.1. The expected boundary velocity is given in the final row for each case.

N	$n = 6/5$, $\alpha = 5/2$	$n = 15/11$, $\alpha = 11/5$
21	-0.5638	-0.0731
41	-0.6685	-0.0944
81	-0.7062	-0.1014
161	-0.7219	-0.1044
321	-0.7289	-0.1058
641	-0.7323	-0.1064
1281	-0.7339	-0.1068
v_b^0	≈ -0.95	≈ -0.18

Table 6.2: Initial boundary velocities for the finite retreat experiments in §6.6.1. The expected boundary velocity is given in the final row for each case.

N	$n = 5/2$, $\alpha = 5/2$	$n = 2$, $\alpha = 2$
21	-0.0076	-0.0076
41	-0.0012	-0.0012
81	-1.63×10^{-4}	-1.63×10^{-4}
161	-2.15×10^{-5}	-2.15×10^{-5}
321	-2.75×10^{-6}	-2.75×10^{-6}
641	-3.48×10^{-7}	-3.47×10^{-7}
1281	-4.36×10^{-8}	-4.37×10^{-8}
v_b^0	0	0

Table 6.3: Initial boundary velocities for the waiting-time experiments in §6.6.1. The expected boundary velocity is given in the final row for each case.

cases we have no conclusive evidence of convergence and we cannot be confident that the method is producing satisfactory results.

The finite retreat cases (Table 6.2) do have the correct sign, but there appears little evidence that the velocity is converging towards the expected value at an acceptable rate as N increases. In both cases the calculated boundary velocity is larger than the expected value for all values of N.

Of the three cases investigated the waiting-time case (Table 6.3) is most encouraging, as the computed velocity does appear to be decreasing towards zero as N increases. Since the MPCM is a numerical method we would not expect the velocity to be exactly zero, but a decrease of one order of magnitude as N is doubled suggests that it will eventually reach rounding error. The values shown in Table 6.3 are much higher than rounding error however, so the number of nodes required for the velocity to approach rounding error is likely to be prohibitively high.

In an attempt to understand why the initial velocities are not as expected, we investigate the computed boundary velocity for one of the experiments. Choosing the $n = 6/5$, $\alpha = 5/2$ finite retreat case, we plot the computed initial velocity for the final 20 mesh points of the mesh. This computed velocity is given in Figure 6.1.

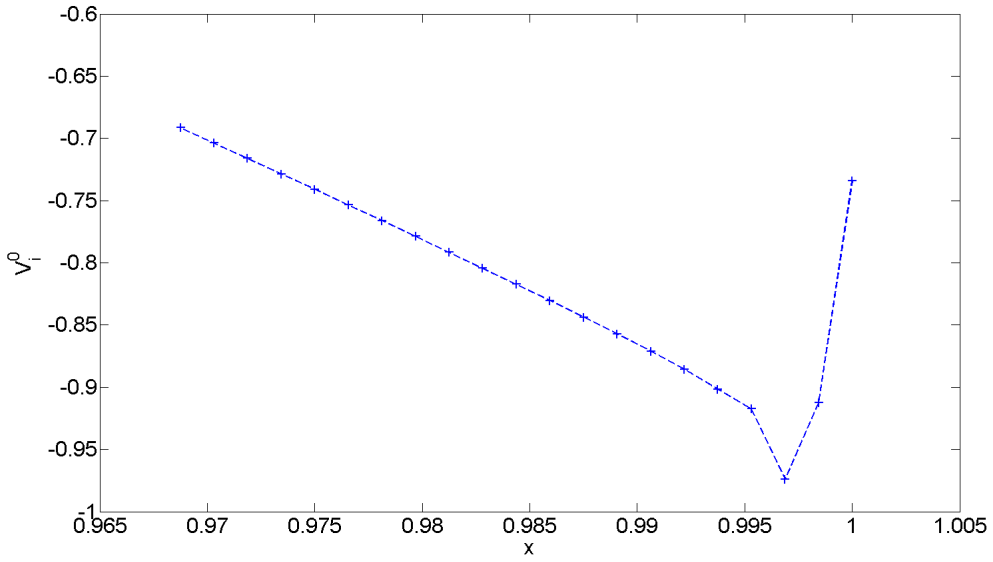

Figure 6.1: The computed velocity V_i^0 plotted over the final 20 mesh points for the $n = 6/5$, $\alpha = 5/2$ finite retreat case.

We see that oscillations have appeared at points close to the boundary, which then cause the quadratic extrapolation (used in the evaluation of the boundary velocity) to be performed inaccurately. Examining the velocity for the other cases (not shown), we also observe the appearance of oscillations close to the boundary which cause the computed boundary velocity to be different than the expected value.

In an attempt to explain these inaccurate boundary velocities, we must examine more than the product of terms in (6.8). The restriction on the values of n and α is not the only consideration that must be made when considering the limit in (6.8). The function $q(x,t)$ in (6.1) and its derivative $\frac{\partial q}{\partial x}$ may also be problematic, which we shall examine next.

6.6.2 Initial Boundary Values of $q(x,t)$

The implementation of the MPCM described in Chapters 3 and 4 involves an approximation of the $q(x,t)$ function as well as the velocity $v(x,t)$ (which from (6.2) consists of

a pointwise multiplication of u^{n-1} and $\frac{\partial q}{\partial x}$). We should therefore take into consideration potential issues arising from the method for the approximation of the boundary values of $q(x,t)$ and $\frac{\partial q}{\partial x}$, from both an asymptotic and computational standpoint.

If $u^0(x)$ is of the form (6.4), then at time $t = t^0$ we can study the initial value of $q(x,t)$ ($q^0(x)$, say) for points close to the boundary. From (6.4), $q^0 = -\frac{\partial^2 u^0}{\partial x^2}$ is given by

$$q^0(x) = 2c_1\alpha(\omega^2 - x^2)^{\alpha-1} - 4c_1\alpha(\alpha-1)x^2(\omega^2 - x^2)^{\alpha-2},$$

which, for $x \approx \omega$ is given by

$$q^0(x) \approx 2^\alpha c_1\alpha\omega^{\alpha-1}(\omega - x)^{\alpha-1} - 2^\alpha c_1\alpha(\alpha-1)\omega^\alpha(\omega - x)^{\alpha-2}. \tag{6.12}$$

We see from (6.12) that the limit as $x \to \omega$ is unbounded for values of $\alpha < 2$. Since (6.12) is not required in the limit of (6.8) however, we have no concerns with whether such a quantity exists in the calculation of (6.8). Nevertheless, the issue is that for values of $\alpha < 2$ the MPCM is attempting to approximate an unbounded quantity using a finite difference method, which is not feasible.

For $\alpha = 2$, $q^0(x)$ is finite at the boundary while, for $\alpha > 2$, $q^0(x)$ is equal to zero. In such cases, we are able to approximate the boundary value of $q^0(x)$ using finite differences. The MPCM in Chapters 3 and 4 may not produce an accurate approximation however, since as mentioned previously in §3.3.3, the method employs an extrapolation using second degree Lagrange polynomials to approximate a quadratic $q(x,t)$. For non-integer values of α such an approach may not produce an accurate approximation to the boundary value of $q(x,t)$ if it is not quadratic, although this may not necessarily be a substantial source of error in the MPCM.

6.6.3 Initial Value of $\frac{\partial q}{\partial x}$ in the Vicinity of the Boundary

The limit in (6.3) includes the term $\frac{\partial q}{\partial x}$, which we have not yet discussed in our asymptotic analysis of the velocity. The limit of $\frac{\partial q}{\partial x}$ at the boundary is of crucial importance to the resulting velocity, so we now discuss the initial asymptotic behaviour of $\frac{\partial q}{\partial x}$ for initial conditions of the form (6.4).

From (6.4), we can calculate the initial approximation to $\frac{\partial q}{\partial x}$ asymptotically at points close to the boundary. Using (6.4),

$$\frac{\partial q^0}{\partial x} = 8c_1\alpha(\alpha-1)(\alpha-2)x^3(\omega^2-x^2)^{\alpha-3} - 12c_1\alpha(\alpha-1)x(\omega^2-x^2)^{\alpha-2},$$

which, for $x \approx \omega$ is given by

$$\frac{\partial q^0}{\partial x} \approx 2^\alpha c_1\alpha(\alpha-1)(\alpha-2)\omega^\alpha(\omega-x)^{\alpha-3} - 3(2^\alpha)c_1\alpha(\alpha-1)\omega^{\alpha-1}(\omega-x)^{\alpha-2}. \quad (6.13)$$

It is clear from (6.13) that for $\alpha < 3$, $\frac{\partial q^0}{\partial x}$ becomes unbounded as $x \to \omega$. We cannot therefore expect the MPCM to be able to accurately approximate $\frac{\partial q}{\partial x}$ at the boundary points using extrapolation (or indeed any numerical method) for values of $\alpha < 3$.

This information provides further insight into the possible boundary behaviours from (6.8). For $\alpha < 3$ we have that $\frac{\partial q^0}{\partial x}$ is unbounded at the boundary of the domain and hence the limit (6.3) (if it exists) will involve a multiplication of the form "$0 \times \infty$" for values of $n > 1$ (and may therefore be indeterminate).

6.6.4 Consequences for the MPCM

From the asymptotic analysis for the velocity near the boundary presented in §6.4–§6.6.2, we see that the implementations of the MPCM described in Chapters 3 and 4 for $n > 1$ are not capable of accurately approximating either the function $q(x,t)$ or the velocity $v(x,t)$ at points close to the boundary. In particular, for initial conditions of the form (6.4) there is a range of values of n and α for which the boundary velocity is unbounded. These issues have been supported by numerical results which demonstrate the inadequacies of the MPCM.

6.7 The FEMPCM when $n > 1$

All discussion relating to numerical methods in this Chapter so far has been focussed on the MPCM for $n > 1$. The issues reported in the finite difference implementation occur because the representation requires all calculations to be performed point-wise on each node of the mesh (including boundary points). In the FEMPCM (see Chapter 5) the

weak forms and resulting finite element approximations are integral forms and it can be argued that the FEMPCM will be able to calculate the velocity at points close to the boundary without the issues previously experienced due to the singularity present at the boundary.

In this section we explore the initial behaviour and convergence of the piecewise linear implementation of the FEMPCM, described in §5.5 (which includes the calculation of the approximation $Q(x,t)$). We shall use this implementation, as opposed to the higher order implementation which possesses the *S Property* described in §5.6, as the higher order implementation is only appropriate for the $n = 1$ case.

We conduct numerical experiments in which we hope to observe the various behaviours of the boundaries (finite advance, finite retreat and waiting-time) outlined in §6.4 for values of n and α such that $n\alpha \geq 3$. It is hoped that since the functions are approximated throughout the domain (using L^2 projections) rather than just point-wise at the mesh points, the velocity will be non-oscillatory close to the boundary and therefore produce convergent boundary trajectories.

The implementation used involves the calculation of $Q(x,t)$, which we have shown to be problematic in the implementation of the MPCM for point-wise approximations. It is hoped that the integral forms of the finite element implementation will alleviate the issues associated with $Q(x,t)$ and $V(x,t)$ close to the boundary.

6.7.1 Finite Advance

We set $n = 2$ in (6.1), and use the initial condition

$$u^0(x) = \frac{1}{10}(1 - x^2)^{3/2},$$

in the finite element implementation, giving a value of $n\alpha = 3$ in which $n = 2$, $\alpha = 3/2$ so that $1 < \alpha < 2$. We therefore expect the right-hand boundary of the domain to move outwards with a finite velocity. In this experiment a time window of $t \in [1, 1.25]$ is used in order to provide information on the behaviour of the boundary for times beyond the initial time.

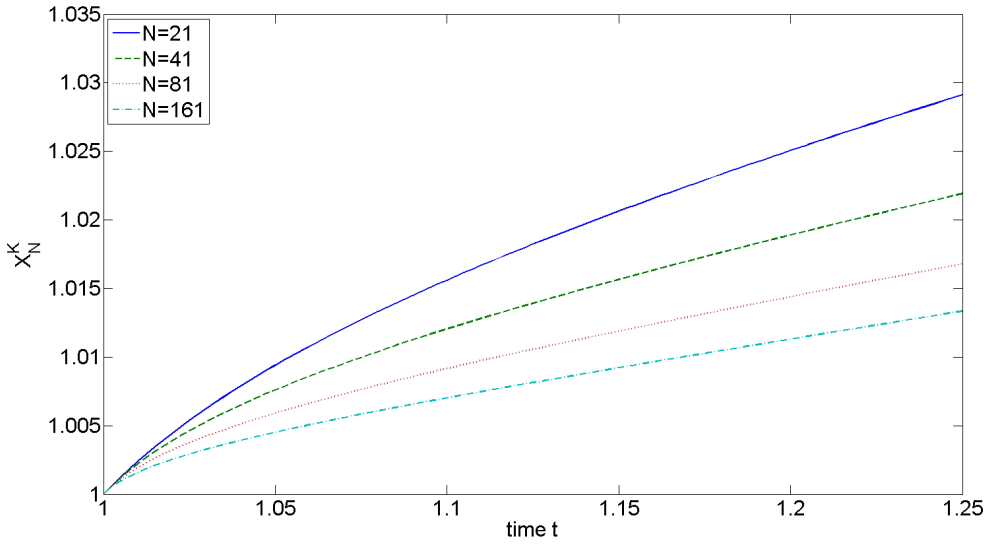

Figure 6.2: Boundary trajectories for the finite advance experiment. Shown here are trajectories for meshes of $N = 21$ (blue solid line), 41 (green dashed line), 81 (red dotted line) and 161 (cyan dash-dotted line) nodes.

Figure 6.2 details the boundary trajectories for the finite advance experiment. We see that the boundary has moved further for the $N = 21$ case than all others, with the maximum distance moved by the boundary decreasing each time N is increased. There is no conclusive evidence of convergence of the boundary trajectories.

We next examine the nodal values of the solution at the end of the time window on mesh points which we would expect to coincide over all of the meshes used. We test values at the central point of the mesh $x = 0$, the right-hand boundary X_N and an 'Interior Point' lying between the central point and the right hand boundary.

If we use the $N = 161$ solution as a reference solution (U_{ref}, say), we can determine the absolute difference between the solution values and mesh locations for the other meshes. The absolute differences are given in Table 6.4, with the solution values at the boundary omitted due to the zero boundary conditions making such calculations redundant. We also omit the difference in mesh position at the central point as the symmetric initial conditions cause the central point to remain fixed at $x = 0$ throughout

the time window.

	Central Point	Interior Point		Boundary Point
N	$\lvert U_{ref} - U \rvert$	$\lvert U_{ref} - U \rvert$	$\lvert X_{ref} - X \rvert$	$\lvert X_{ref} - X \rvert$
21	3.49×10^{-5}	3.59×10^{-6}	2.07×10^{-4}	0.0158
41	3.55×10^{-6}	1.05×10^{-5}	6.23×10^{-5}	0.0086
81	8.09×10^{-8}	4.22×10^{-6}	1.46×10^{-5}	0.0034

Table 6.4: Absolute difference of mesh points and solution values for the finite advance experiment at coinciding points

Table 6.4 shows some evidence of slow convergence between the solution values of various meshes. The rate at which this convergence is being achieved is not very quick however, indicating a possible need to increase the number of mesh points further. Some evidence of convergence has been observed for the interior and boundary mesh points, but the results are unconvincing.

6.7.2 Finite Retreat

In the finite retreat experiment we set $n = 1.2$ in (6.1) and use the initial condition

$$u^0(x) = \frac{1}{10}\left(1 - x^2\right)^{5/2},$$

in the finite element implementation. For these choices of n and α we therefore expect an initially retreating right-hand boundary with a finite velocity. We run the method over the time window $t \in [1, 1.025]$.

Figure 6.3 shows the boundary trajectories for meshes of $N = 21$, 41, 81 and 161 nodes. We see that in all cases the boundary moves inwards as expected, and there is some evidence of convergence for the $N = 41$, 81 and 161 trajectories. The $N = 21$ trajectory has not moved as far outward by the end of the time window as the other cases, which may indicate that the initial mesh was too coarse.

This result appears more encouraging, since there is some evidence of convergence of the boundary trajectories in this example. By increasing the number of nodes in the

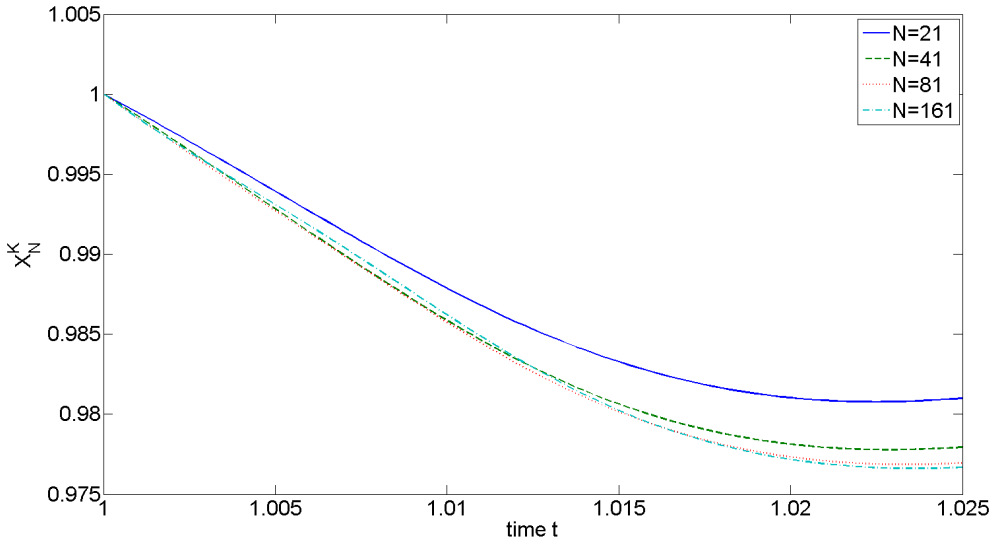

Figure 6.3: Boundary trajectories for the finite retreat experiment. Shown here are trajectories for meshes of $N = 21$ (blue solid line), 41 (green dashed line), 81 (red dotted line) and 161 (cyan dash-dotted line) nodes.

mesh further, we might expect the trajectories to become closer together. As discussed in the finite difference case however, the rate of convergence is not quick enough to propose that a reference trajectory has been reached.

If we use the $N = 161$ solution as a reference solution (U_{ref}, say), we can determine the absolute difference between the solution values and mesh locations for the other meshes. The absolute differences are given in Table 6.5.

Table 6.5 shows that there is some evidence of convergence between the various meshes. The rate of convergence is slow however, indicating a possible need to increase the number of mesh points further.

6.7.3 Waiting-Time

The waiting-time experiment uses value of $n = 2$ and $\alpha = 2$, giving an initial condition

$$u^0(x) = \frac{1}{10}(1 - x^2)^2$$

	Central Point	Interior Point		Boundary Point
N	$\lvert U_{ref} - U \rvert$	$\lvert U_{ref} - U \rvert$	$\lvert X_{ref} - X \rvert$	$\lvert X_{ref} - X \rvert$
21	8.70×10^{-6}	8.59×10^{-6}	1.99×10^{-4}	0.0043
41	2.62×10^{-6}	1.85×10^{-6}	5.06×10^{-5}	0.0013
81	5.45×10^{-7}	3.85×10^{-7}	9.89×10^{-6}	2.76×10^{-4}

Table 6.5: Absolute difference of mesh points and solution values for the finite retreat experiment at coinciding points

and an initial boundary velocity of zero. The time window used in this experiment is $t \in [1, 1.025]$, as in the finite retreat case.

Figure 6.4 shows the boundary trajectories for the various meshes, which we would expect to remain within rounding error of $x = 1$ if the boundary is experiencing waiting-time behaviour. We see that the $N = 21$ trajectory has moved outwards more than expected, indicating that the velocity at the boundary node is unacceptably larger than zero. As the number of nodes in the mesh increases however, the trajectories begin to converge towards the expected values. We would therefore expect that a higher number of nodes would produce a trajectory which is even closer to zero than the $N = 161$ trajectory.

Using the $N = 161$ solution as a reference solution, we can determine the absolute difference between the solution values and mesh locations for the other meshes. The absolute differences are given in Table 6.6. The results for the finite element waiting-time experiment are slightly more encouraging than those for the finite difference case, since better evidence of convergence is observed.

6.7.4 Initial Boundary Velocities

We can estimate the initial velocity of the right-hand boundary by calculating the initial velocity using the finite element implementation. By only calculating the initial velocity we are able to obtain computed boundary velocities for a much larger range of N. In Table 6.7 we produce the computed velocities for the three experiments conducted in

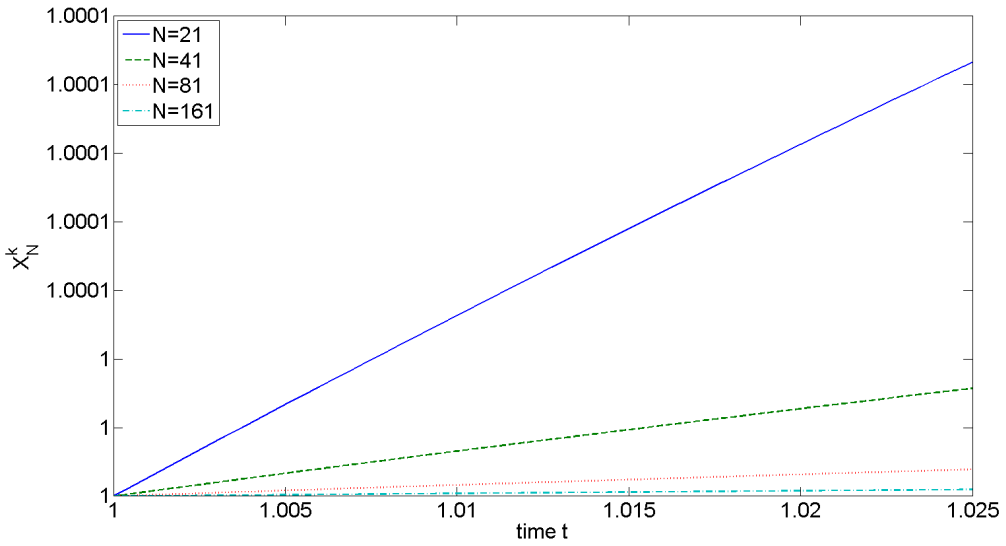

Figure 6.4: Boundary trajectories for the waiting-time experiment. Shown here are trajectories for meshes of $N = 21$ (blue solid line), 41 (green dashed line), 81 (red dotted line) and 161 (cyan dash-dotted line) nodes.

	Central Point	Interior Point		Boundary Point
N	$\lvert U_{ref} - U \rvert$	$\lvert U_{ref} - U \rvert$	$\lvert X_{ref} - X \rvert$	$\lvert X_{ref} - X \rvert$
21	5.03×10^{-7}	7.22×10^{-6}	3.21×10^{-5}	1.25×10^{-4}
41	7.83×10^{-8}	1.64×10^{-6}	7.92×10^{-5}	2.95×10^{-5}
81	1.38×10^{-8}	3.29×10^{-7}	1.59×10^{-6}	5.87×10^{-6}

Table 6.6: Absolute difference of mesh points and solution values for the waiting-time experiment at coinciding points

§6.7.1–§6.7.3 for values of $N = 21$ to $N = 1281$, with Remark 6.1 about v_b^0 from §6.6.1 valid here.

N	Advance	Retreat	Wait
21	0.2597	-1.1248	0.0054
41	0.2629	-1.3433	0.0014
81	0.2643	-1.4534	3.39×10^{-4}
161	0.2650	-1.5085	8.46×10^{-5}
321	0.2653	-1.5362	2.12×10^{-5}
641	0.2655	-1.5500	5.29×10^{-6}
1281	0.2655	-1.5569	1.33×10^{-6}
v_b^0	0.03	~ -0.95	0

Table 6.7: Initial Boundary Velocities for the experiments in §6.7.1–6.7.3. The expected boundary velocity is given in the final row for each case.

Examining the results given in Table 6.7 we see that there are some surprising results being obtained, indicating that the velocity calculation near the boundary is not being obtained as expected. The finite advance case sees computed boundary velocities an order of magnitude higher than expected throughout, which appear to be converging to the incorrect value. The finite retreat case is producing a negative velocity, but this velocity is not converging to the expected value as the number of mesh points increases.

The waiting time computed velocities are decreasing as the number of nodes increases, but are not in the region of rounding error by several orders of magnitude. It was hoped that a mesh of $N = 1281$ nodes would be producing a boundary velocity in the region of rounding error.

If we examine the computed velocity for one of these examples, we can determine whether the oscillations in the velocity close to the boundary (observed in the finite difference implementation) are also appearing in the finite element implementation. Figure 6.5 shows the computed velocity for the finite advance case with $N = 81$, for a selection of mesh points close to the right-hand boundary. We see that there is clear oscillation

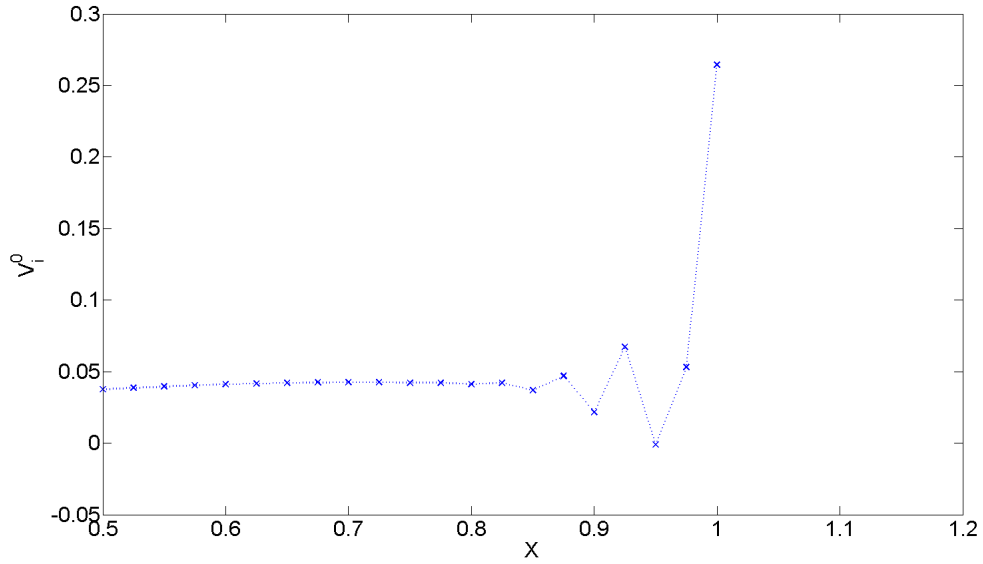

Figure 6.5: Computed velocity in the first time step for the finite advance experiment with $N = 81$, at points near the right-hand boundary.

close to the boundary, which has caused the boundary velocity to be larger than the expected value.

6.7.5 Critique of the FEMPCM with $n > 1$

Since the oscillations are observable in both the MPCM (see §6.6.1) and the FEMPCM (§6.7) , the issue with convergence that we are observing at the boundaries of the domain is not restricted to the implementation being used.

The need to calculate an approximation to the $q(x,t)$ function and its first derivative, even in weak form, appears to be the main cause of issues when considering the implementation of the FEMPCM (as discussed in §6.6.2–§6.6.3 for the MPCM). When considering initial conditions of the form (6.4), the range of α for which either $q(x,t)$ or $\frac{\partial q}{\partial x}$ becomes unbounded at the boundary is restrictively large.

Despite the integral forms present in the FEMPCM, allowing for a weak form representation present throughout the whole domain (as opposed to point-wise values in

the finite difference implementation), the implementation appears to be unable to adequately model the velocities expected for the various values of α tested in this section.

6.8 Mesh Point Distribution

In the discussion of the numerical results for the MPCM and FEMPCM presented in this thesis, the distribution of the mesh points has thus far been overlooked. In this section we shall explore the initial distribution of the mesh points and their evolution at later times, with a view to how the MPCM and FEMPCM could benefit from careful selection of the initial mesh.

6.8.1 Uniform Initial Mesh

All the numerical results presented in this thesis so far have been generated using a mesh which was chosen to be initially equally spaced. In the cases where the MPCM or FEMPCM possesses the *S Property* (i.e. the results given in Chapters 4 and 5), this uniform initial mesh remains uniform over time due to the linear velocity present in these cases (see equations (3.14) and (3.31) for $n = 1$).

As an example, Figure 6.6 illustrates the mesh evolution for the multiple time step results given in §4.3.6. In this case the initial mesh spacing was taken to be $\Delta X = 0.2$, which has increased to $\Delta X = 0.5$ at time $s = 2.5$.

For the examples presented in this chapter, an initially uniform mesh (driven by a velocity which is not necessarily linear in terms of x) may not remain uniform. A uniform mesh may also not provide adequate resolution at points of interest (such as near the boundary for the types of problems studied in this chapter).

To demonstrate this we examine the cases explored in §6.6.1, for the various choices of n and α used. In the results presented only the initial boundary velocity was discussed. Here we run the MPCM for the various n, α values over 10,000 time steps and plot the evolution of the final 11 mesh points from a mesh of $N = 81$ points. These plots are given in Figure 6.7.

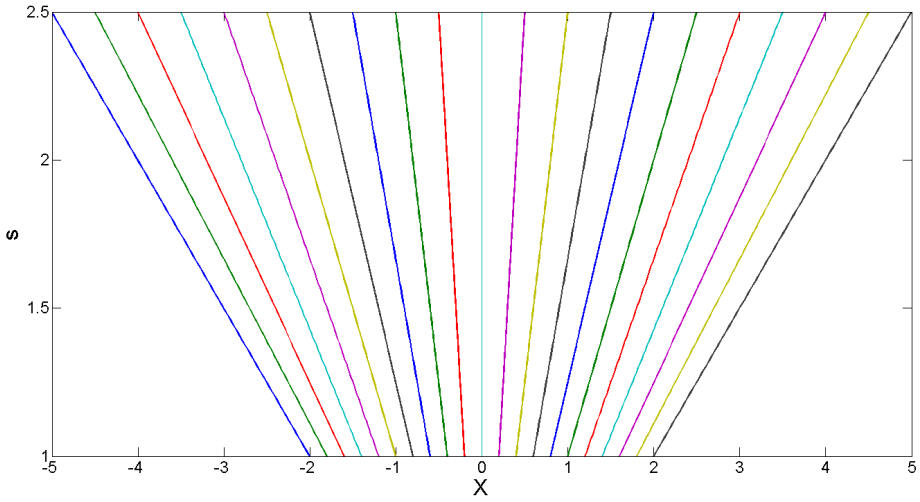

Figure 6.6: Evolution of the mesh points for the multiple time step results given in 4.3.6. Mesh used was initially equally spaced.

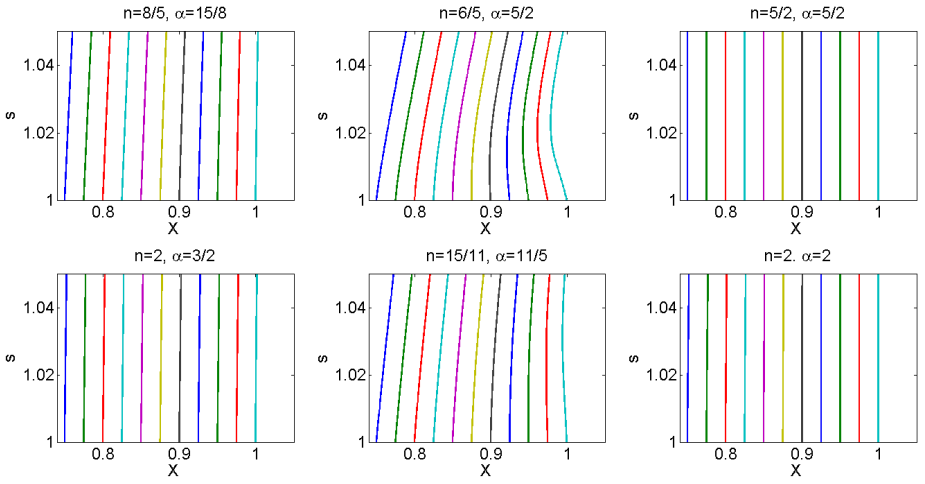

Figure 6.7: Mesh point evolution for the final 11 points of the $N = 81$ mesh for the various examples given in §6.6.1 over 10,000 time steps. Shown here are finite advance (top/bottom left), finite retreat (top/bottom centre) and waiting-time (top/bottom right) meshes.

We see that the most interesting behaviour is taking place in the finite retreat cases ($n = 6/5, \alpha = 5/2$ and $n = 15/11, \alpha = 11/5$), with the points closest to the boundary being of particular interest. Here the mesh spacing decreases as the boundary retreats, before remaining at this size once the boundary begins to move outwards again.

In the finite advance and waiting-time cases the behaviour of the mesh is expected, with the uniform mesh appearing to remain uniform over the multiple time steps.

Instead of a uniform mesh, we could instead distribute the points such that they are concentrated nearer to the boundary positions. This will allow for a better resolution of features close to the boundary.

We shall now explore a couple of different methods of initially distributing the mesh points for the MPCM/FEMPCM which are based on equally distributing some feature of the solution.

6.8.2 Equidistribution of Mass

The mesh can be initially set up such that the mass between two mesh points is equal for all mesh points, i.e.

$$\int_{X_i^0}^{X_{i+1}^0} u^0(x)\,\mathrm{d}x = c_i, \qquad \text{for } i = 1, \ldots, N-1.$$

The advantage of this method of initially distributing points for the MPCM or FEMPCM is that since the velocity and recovery steps of the method are driven by mass conservation, if the initial partial masses are equal then they will remain so for all time.

A disadvantage of equidistributing mass for the types of problem studied in this thesis is that since the solution is generally smallest near the boundary, the initial mesh distribution will be sparse close to the boundary, even more so than the uniform meshes used in previous chapters. This sparseness will then lead to a poor resolution of important features of the solution, particularly for the examples in this chapter.

6.8.3 Equidistribution of Arc Length

Alternatively, the initial mesh can be set up such that the arc length between any two mesh points is equally distributed throughout the mesh. In this case, the mesh is chosen such that

$$\int_{X_i^0}^{X_{i+1}^0} \sqrt{1+\left(\frac{\partial u^0}{\partial x}\right)^2}\,\mathrm{d}x = c_i, \qquad \text{for } i = 1,\ldots,N-1.$$

Equally distributing the arc length of the solution is a more suitable choice for the problems studied in this chapter, since it can result in an initial mesh which has a better distribution of points close to the boundary.

In the next section we shall explore the first of two possible strategies for alleviating the issues observed by both the MPCM and FEMPCM in the chapter thus far. The first of these strategies will attempt to allow the modelling of solutions for cases where $0 < n\alpha < 3$ and involves a hybrid numerical method.

6.9 A Hybrid Numerical Method

Depending upon the value of n chosen and the initial condition to the problem, it is possible that (analytically) the interfaces may move with an unbounded velocity (Bowen and King (2001)). This unbounded velocity is impossible to model using the MCPM (see §6.3) and requires an alternative approach in order to produce solutions in such cases.

In order to avoid the issue of having to track the movement of a point moving with an unbounded velocity, a hybrid numerical method is proposed. In the hybrid method a fixed mesh numerical method is used over a time window $t \in [t^0, T^0]$, where T^0 is chosen large enough such that the velocity of the interface has become finite, so that it is able to be approximated accurately by the MPCM.

Once the solution to the fixed-mesh part of the method has been found (and shown to be convergent), the approximate solution calculated at time $t = T^0$ is then used as the initial condition in the MPCM. Since the problem possesses a finite speed of propagation property for all $t > t^0$, the MPCM should then be able to track the movement of the

interfaces.

6.9.1 The Fixed-Mesh Method

The use of a fixed-mesh method requires us to recast the original fourth-order problem. We shall seek a solution $u(x,t)$ to the fourth-order PDE

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u^n \frac{\partial q}{\partial x}\right), \qquad \text{in } \Omega \times (t^0, T)$$
$$q = -\frac{\partial^2 u}{\partial x^2}, \qquad \text{in } \Omega \times (t^0, T)$$

for $x \in \Omega$, where Ω is a finite-space interval (X_a, X_b) and no longer time-dependent.

We seek solutions subject to an initial condition

$$u(x, t_0) = u^0(x), \qquad \text{at } t = t^0,$$

and boundary conditions

$$\frac{\partial u}{\partial x} = 0,$$
$$uv + u^n \frac{\partial q}{\partial x} = 0,$$

at the moving interfaces $x = a(t)$ and $x = b(t)$, for $t > t^0$.

Compared with the MPCM, the fixed-mesh part of the hybrid numerical method operates on a static mesh of points, with a fixed distance Δx between points. Since the mesh is now fixed, the domain Ω will include points outside of the support of $u(x,t)$, to allow the interfaces of the support to potentially expand over time. For all points in Ω outside of the interfaces at $x = a(t)$, $x = b(t)$ we have that $u = 0$. We shall choose the two points X_a, X_b far enough away from the support of $u(x,t)$ that the interfaces will not reach these points by the end of the time window.

By including mesh points outside of the support of the solution we are able to use a numerical scheme in the approximation of $u(x,t)$ which has an effect on the movement of the free boundary over a given time-step. This feature of the numerical scheme will allow the hybrid method to be able to model the case where the boundary moves instantaneously with an unbounded velocity, which was not previously possible in the MPCM.

Given a discretisation of the domain $\Omega \equiv x \in [X_a, X_b] \subset \mathbb{R}$ in the form of a mesh of N nodes, we can approximate the function $u(x,t)$ by nodal values U_i $(i = 1, \ldots, N)$ using a finite difference scheme.

Due to the presence of the u^n factor in (6.1) we choose the semi-implicit finite difference scheme

$$U_i^{k+1} = U_i^k + \frac{\Delta t}{\Delta x^2}\left((U_{i+1/2}^k)^n \left\{\frac{(-U_{i+2}^{k+1} + 2U_{i+1}^{k+1} - U_i^{k+1})}{\Delta x^2} - \frac{(-U_{i+1}^{k+1} + 2U_i^{k+1} - U_{i-1}^{k+1})}{\Delta x^2}\right\}\right.$$
$$\left. +(U_{i-1/2}^k)^n \left\{\frac{(-U_{i+1}^{k+1} + 2U_i^{k+1} - U_{i-1}^{k+1})}{\Delta x^2} - \frac{(-U_i^{k+1} + 2U_{i-1}^{k+1} - U_{i-2}^{k+1})}{\Delta x^2}\right\}\right), \tag{6.14}$$

for $i = 2, \ldots, N-2$.

The values $U_{i+1/2}^k$ and $U_{i-1/2}^k$ in (6.14) are obtained by the midpoint approximations

$$U_{i+1/2}^k = \frac{U_{i+1}^k + U_i^k}{2}, \qquad U_{i-1/2}^k = \frac{U_i^k + U_{i-1}^k}{2}, \tag{6.15}$$

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

We enforce boundary conditions by setting $U_0^k = U_1^k = U_{N-1}^k = U_N^k = 0$ for all values of k, as we have chosen the boundaries of the domain Ω such that the numerical free boundary does not reach the boundary of Ω over the course of the time window. Hence we do not expect the solution at the nodes outside the support to take any value other than zero.

We can rearrange (6.14) and the boundary conditions in order to obtain a linear system of N equations of the form

$$A\mathbf{U}^{k+1} = \mathbf{F}^k,$$

for the solution vector $\mathbf{U}^{k+1} = \{U_i^{k+1}\}$, where A is a $N \times N$ pentadiagonal matrix and $\mathbf{F}^k$ is a vector of length N.

The first two rows and last two rows of A have non-zero entries only on the main diagonal, with entries $A_{0,0} = A_{1,1} = A_{N-1,N-1} = A_{N,N} = 1$. The remaining rows have

the following form:

$$
\begin{aligned}
A_{i,i-2} &= \mu[(U^k_{i-1/2})^n], \\
A_{i,i-1} &= -\mu[(U^k_{i+1/2})^n + 3(U^k_{i-1/2})^n], \\
A_{i,i} &= 1 + 3\mu[(U^k_{i-1/2})^n + (U^k_{i+1/2})^n], \\
A_{i,i+1} &= -\mu[3(U^k_{i+1/2})^n + (U^k_{i-1/2})^n], \\
A_{i,i+2} &= \mu[(U^k_{i+1/2})^n],
\end{aligned}
$$

where $\mu = \Delta t/(\Delta x)^4$, for $i = 2, \ldots, N-2$. The vector $\mathbf{F}^k$ has entries

$$\mathbf{F}^k = (0, 0, U^k_2, U^k_3, \ldots, U^k_{N-3}, U^k_{N-2}, 0, 0)^T.$$

6.9.2 Defining the Numerical Interface

Since the approximate solution U^k_i is only defined at discrete points and the position of the moving interface (at which $u(x,t) = 0$) may lie between two mesh points we need a method of defining an approximation to the position of the moving interface at the end of the time window $t \in [t^0, T^0]$, since the position of the interfaces will need to be passed to the MPCM.

From (6.14) and (6.15), we can see that for points outside of the support of $u(x,t)$ we have

$$U^k_{i-1} = U^k_i = U^k_{i+1} = 0 \;\Rightarrow U^{k+1}_i = 0, \tag{6.17}$$

and as a result the free boundary can advance by at most one mesh point over a single time step. A consequence of (6.17) is that the time step size Δt must be chosen such that $\Delta t = O(\Delta x^2)$ (Barrett et al. (1998)).

Given this feature of the numerical scheme, it is possible to introduce a means of determining the position of the numerical boundary. Since the solution is expected to remain nonnegative if the initial condition is nonnegative (Bernis and Friedman (1990)), we may define the position of the numerical interface to coincide with the first mesh point at which the solution is zero outside of the support of the approximate solution. This approach is flawed if oscillations appear in the solutions as a result of the numerical

scheme however, which makes determining such a point extremely problematic. A non-negativity preserving scheme (such as that found in Blowey et al. (2007)) should ensure that the interface can be obtained in this manner, but this has not been included in the implementation presented here.

6.9.3 Convergence of the Fixed-Mesh Solution

It is important that the solution obtained from the fixed-mesh part of the hybrid method is shown to converge, since the solution at time T^0 is needed for use as the initial condition to the MPCM part of the method.

We test convergence of the fixed mesh method by reducing the mesh spacing Δx and comparing the approximate solution calculated in each case (for each experiment the value of Δt used in the time stepping method is chosen such that the ratio $\Delta t/\Delta x^2$ remains constant throughout). The first comparison test made is the difference between the mass of the initial condition and the mass of the approximate solution at time T^0, calculated using a trapezoidal rule. It is hoped that as Δt and $\Delta x \to 0$, the difference in these masses will also tend to zero.

We may also test convergence by observing the position of the numerical interface over the time window. For a convergent method, as the number of points in the mesh increases the trajectories formed by the interface positions will converge towards some form of reference trajectory. If the interface trajectories do not show convergence, the fixed-mesh solution may not be usable with the MPCM.

At present, the implementation of the fixed-mesh portion of the hybrid method produces oscillations at points in the vicinity of the interfaces, which causes a lack of convergence of the interface trajectories. This problem needs to be addressed in order to enable a satisfactory implementation of the hybrid method, but is left as future work due to time constraints.

6.9.4 Use of the MPCM

Once the solution obtained by the fixed-mesh method described in §6.9.1 has been shown to converge, the mesh is reduced to only mesh points containing the support of U, with at most one zero-valued mesh point at each interface.

From the fixed mesh solution, we select a representation of mesh points throughout the support of U, using the selected points as the initial approximation $\mathbf{U}^0$ in the MPCM. The MPCM is much more computationally expensive to run when the number of nodes in the mesh is high, which is likely to be the case when the fixed-mesh method has converged, hence the need to reduce the number of points used in this part of the hybrid method.

This solution is then used as the initial condition in the MPCM, at time $t = T^0$. The MPCM method is then run over the time window $t \in [T^0, T]$, where T is the final time the method is run to, as described in Chapters 3 or 4.

6.9.5 Critique of the Hybrid Method

The hybrid method described in this section has the potential to be able to model the case where the initial velocity of the boundary of the domain is instantaneously unbounded, as well as modelling the other expected behaviours discussed in §6.4. However, there are a number of factors to consider in order to enable the method to work correctly, which have so far prevented the method being implemented satisfactorily:

- The value of T^0 is important, since we are interested in using the MPCM part of the method as soon as possible. If the value of T^0 is taken to be too small however, the boundary velocities may be too large to be modelled accurately. Conversely, if the value of T^0 is taken to be too large, the fixed-mesh part of the method may be being used for an unnecessarily large number of time steps. The semi-implicit scheme used in the fixed-mesh part of the method is of a lower order of accuracy compared with the MPCM part, so minimising the number of time steps used in the fixed-mesh part is beneficial to the overall accuracy of the hybrid method.

- In the fixed-mesh portion of the hybrid method there is a further potential issue which may arise if the moving interface lies on or close to a mesh point, in cases where $q(x,t)$ or $\frac{\partial q}{\partial x}$ are unbounded at the interface. In this instance we may not produce accurate results as the finite difference scheme used would not be able to approximate these unbounded quantities.

- Early simulations of the hybrid numerical method have shown that the fixed-mesh portion may develop unwanted oscillations at points close to the boundary, which makes estimation of the position of the numerical interface problematic. Although much effort was made in attempting to reduce these oscillations or improve the means of estimating the interface position, the outcome was unsatisfactory.

6.10 An Alternative Expression for the Velocity

In this section we shall consider the fourth-order problem written as a single fourth-order PDE rather than two second-order equations (and therefore without reference to the $q(x,t)$ function). Setting $m = 1$ in (2.1), we therefore seek solutions $u(x,t)$ to

$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}\left(u^n \frac{\partial^3 u}{\partial x^3}\right), \qquad \text{in } \Omega_T,$$

subject to the initial condition at time $t = t^0$ given by

$$u(x,t^0) = u^0(x), \qquad \text{for } x \in \Omega(t^0),$$

and the boundary conditions

$$\begin{aligned} u &= 0, && \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \\ \frac{\partial u}{\partial x} &= 0, && \text{at } x = a(t), \quad x = b(t), \quad t > t^0, \\ uv - u^n \frac{\partial^3 u}{\partial x^3} &= 0, && \text{at } x = a(t), \quad x = b(t), \quad t > t^0. \end{aligned}$$

Through conservation of mass the velocity at points in the interior of the domain is

$$v = u^{n-1} \frac{\partial^3 u}{\partial x^3}, \tag{6.19}$$

avoiding any reference to $q(x,t)$.

We now introduce an identity which allows us to rewrite (6.19) as

$$v = u^{n-1}\frac{\partial^3 u}{\partial x^3} \equiv \frac{1}{n}\frac{\partial^3(u^n)}{\partial x^3} - \frac{6(n-1)}{n^2}\frac{\partial}{\partial x}\left(\left[\frac{\partial(u^{n/2})}{\partial x}\right]^2\right) + \frac{27(n-1)(n-2)}{2n^3}\left(\frac{\partial(u^{n/3})}{\partial x}\right)^3, \tag{6.20}$$

the proof of which is as follows.

Proof. We shall being by making use of the identity

$$\frac{\partial^3}{\partial x^3}(u^n) \equiv n\frac{\partial^2}{\partial x^2}\left(u^{n-1}\frac{\partial u}{\partial x}\right), \tag{6.21}$$

from which we may write

$$\begin{aligned} n\frac{\partial^2}{\partial x^2}\left(u^{n-1}\frac{\partial u}{\partial x}\right) &\equiv nu^{n-1}\frac{\partial^3 u}{\partial x^3} + 2n\frac{\partial(u^{n-1})}{\partial x}\frac{\partial^2 u}{\partial x^2} + n\frac{\partial u}{\partial x}\frac{\partial^2}{\partial x^2}(u^{n-1}), \\ &\equiv nu^{n-1}\frac{\partial^3 u}{\partial x^3} + 3n(n-1)u^{n-2}\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2} + n(n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3. \end{aligned} \tag{6.22}$$

Dividing (6.22) through by n, using (6.21) and rearranging we see that

$$u^{n-1}\frac{\partial^3 u}{\partial x^3} \equiv \frac{1}{n}\frac{\partial^3 u}{\partial x^3} - 3(n-1)u^{n-2}\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2} - (n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3, \tag{6.23}$$

where the LHS of (6.23) matches the RHS of (6.19). The second and third terms on the RHS of (6.23) are not as yet in their required form.

We note that

$$\begin{aligned} \frac{\partial}{\partial x}\left[\frac{\partial(u^{n-1})}{\partial x}\right] &\equiv \frac{\partial(u^{n-1})}{\partial x}\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2(u^{n-1})}{\partial x^2}\frac{\partial u}{\partial x}, \\ &\equiv \frac{\partial(u^{n-1})}{\partial x}\frac{\partial^2 u}{\partial x^2} + (n-1)\frac{\partial u}{\partial x}\frac{\partial}{\partial x}\left(u^{n-2}\frac{\partial u}{\partial x}\right), \\ &\equiv 2(n-1)u^{n-2}\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2} + (n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3, \end{aligned}$$

which we may rearrange to obtain

$$(n-1)u^{n-2}\frac{\partial u}{\partial x}\frac{\partial^2 u}{\partial x^2} \equiv \frac{1}{2}\frac{\partial}{\partial x}\left[\frac{\partial(u^{n-1})}{\partial x}\right] - \frac{1}{2}(n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3,$$
$$\equiv \frac{1}{2}(n-1)\frac{\partial}{\partial x}\left[u^{n-2}\left(\frac{\partial u}{\partial x}\right)^2\right] - \frac{1}{2}(n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3. \tag{6.25}$$

Substituting (6.25) into (6.23),

$$u^{n-1}\frac{\partial^3 u}{\partial x^3} \equiv \frac{1}{n}\frac{\partial^3 u}{\partial x^3} - \frac{3}{2}(n-1)\frac{\partial}{\partial x}\left[u^{n-2}\left(\frac{\partial u}{\partial x}\right)^2\right] - \frac{1}{2}(n-1)(n-2)u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3, \tag{6.26}$$

which is almost in the required form.

We may write (6.26) in the required form using the identities

$$u^{n-2}\left(\frac{\partial u}{\partial x}\right)^2 \equiv \left(u^{n/2-1}\frac{\partial u}{\partial x}\right)^2 \equiv \frac{4}{n^2}\left(\frac{\partial(u^{n/2})}{\partial x}\right)^2,$$

and

$$u^{n-3}\left(\frac{\partial u}{\partial x}\right)^3 \equiv \left(u^{n/3-1}\frac{\partial u}{\partial x}\right)^3 \equiv \frac{27}{n^3}\left(\frac{\partial(u^{n/3})}{\partial x}\right)^3,$$

which results in

$$u^{n-1}\frac{\partial^3 u}{\partial x^3} \equiv \frac{1}{n}\frac{\partial^3 u}{\partial x^3} - \frac{6(n-1)}{n^2}\left(\frac{\partial(u^{n/2})}{\partial x}\right)^2 + \frac{27}{2n^3}(n-1)(n-2)\left(\frac{\partial(u^{n/3})}{\partial x}\right)^3,$$

as required. □

Remark. *A corresponding identity holds for the second-order problem and in principle exists for the sixth-order problem when the sixth-order PDE is considered as a single equation as opposed to three second-order equations. The details of these corresponding identities are omitted here as the focus is on the fourth-order problem.*

In a similar manner to that discussed in §6.4, we can examine the behaviour of the initial alternative velocity at the boundary at points $x \approx \omega$, when the initial condition is of the form (6.4). For points close to $x = \omega$, we have that $u^0(x)$ is of the form (6.5) and using this we can examine the various terms in (6.20).

The first term on the RHS of (6.20) is given by

$$\frac{1}{n}\frac{\partial^3(u^n)}{\partial x^3} = 12c_1^n\alpha(n\alpha-1)x(\omega^2-x^2)^{n\alpha-2} - 8c_1^n\alpha(n\alpha-1)(n\alpha-2)x^3(\omega^2-x^2)^{n\alpha-3},$$

which, for $x \approx \omega$,

$$\frac{1}{n}\frac{\partial^3(u^n)}{\partial x^3} \approx c_1^n\alpha(n\alpha-1)\Big[3(2^{n\alpha}\omega^{n\alpha-1}(\omega-x)^{n\alpha-2} - (n\alpha-2)(2\omega)^{n\alpha}(\omega-x)^{n\alpha-3}\Big].$$

Similarly, the second term is given by

$$\frac{6(n-1)}{n^2}\frac{\partial}{\partial x}\left(\left[\frac{\partial(u^{n/2})}{\partial x}\right]^2\right) = 12c_1^n(n-1)\alpha^2\Big(x(\omega^2-x^2)^{n\alpha-2} - (n\alpha-2)x^3(\omega^2-x^2)^{n\alpha-3}\Big),$$

which for $x \approx \omega$ has the form

$$\frac{6(n-1)}{n^2}\frac{\partial}{\partial x}\left(\left[\frac{\partial(u^{n/2})}{\partial x}\right]^2\right) \approx c_1^n(n-1)\alpha^2\Big(3(2^{n\alpha})\omega^{n\alpha-1}(\omega-x)^{n\alpha-2} - \frac{3}{2}(n\alpha-2)(2\omega)^{n\alpha}(\omega-x)^{n\alpha-3}\Big).$$

Finally, the third term is given by

$$\frac{27(n-1)(n-2)}{2n^3}\left(\frac{\partial(u^{n/3})}{\partial x}\right)^3 = -\frac{8}{2}c_1^n\alpha^3(n-1)(n-2)x^3(\omega^2-x^2)^{n\alpha-3},$$

which for $x \approx \omega$,

$$\frac{27(n-1)(n-2)}{2n^3}\left(\frac{\partial(u^{n/3})}{\partial x}\right)^3 \approx -\frac{1}{2}c_1^n\alpha^3(n-1)(n-2)(2\omega)^{n\alpha}(\omega-x)^{n\alpha-3}.$$

The velocity (6.20) at points $x \approx \omega$ is therefore

$$\begin{aligned} v \approx\ & 3c_1^n(2^{n\alpha})\omega^{n\alpha-1}\Big(\alpha(n\alpha-1) - (n-1)\alpha^2\Big)(\omega-x)^{n\alpha-2} \\ & - (c_1)^n(2\omega)^{n\alpha}\left[\alpha(n\alpha-1)(n\alpha-2) - \frac{3}{2}\alpha^2(n-1)(n\alpha-2)\right. \\ & \left. +\frac{1}{2}\alpha^3(n-1)(n-2)\right](\omega-x)^{n\alpha-3}. \end{aligned} \tag{6.27}$$

Examining (6.27), we see that the velocity consists entirely of terms which are smooth for choices of $n\alpha \geq 3$, indicating that they should possess a limit at the boundary in these cases. For cases where $n\alpha < 3$ the velocity becomes unbounded as $x \to \omega$ as expected.

6.10.1 Approximating the Alternative Velocity

We can readily approximate the terms in (6.20) using finite differences to obtain an approximation for the velocity for use in the MPCM. This approximation can then be used to determine whether the boundary is behaving as expected from the analysis in §6.4. Crucially, the terms in (6.20) avoid any multiplications of the form "$0 \times \infty$" or the need to approximate the $q(x,t)$ function. This alleviates the issues observed previously with the MPCM.

We shall describe the process of approximating the terms of (6.20) on a mesh of N nodes. At a given time level t^k we seek an approximation V_i at each node of the mesh (with the time level k suppressed for clarity).

The first term on the right hand side of (6.20) is approximated in two steps. In the first step, we approximate $\frac{\partial^2}{\partial x^2}(u^n)$ at interior points using central differences. We denote the approximation $\frac{\partial^2}{\partial x^2}(u^n)$at node i (for $i = 1, \ldots, N-1$) by P_i, then set

$$P_i = \frac{\dfrac{U^n_{i+1} - U^n_i}{X_{i+1} - X_i} - \dfrac{U^n_i - U^n_{i-1}}{X_i - X_{i-1}}}{\frac{1}{2}(X_{i+1} - X_{i-1})}, \qquad \text{for } i = 2, \ldots, N-1. \tag{6.28}$$

This step is similar to the approximation of $q(x,t)$ given in (3.17) with U_i replaced by U^n_i and without the minus sign. At the boundary points of the mesh we set $P_1 = P_N = 0$, since for $n\alpha \geq 3$ we have that

$$\frac{\partial^2}{\partial x^2}(u^n) \sim n\alpha(n\alpha - 1)(\omega - x)^{n\alpha - 2} = 0, \qquad \text{as } x \to \omega,$$

at the boundary points.

We then complete the approximation of the first term of the right-hand side of (6.20) at node i (which we shall denote by T^1_i) in the alternative velocity (6.20) by setting

$$T^1_i = \frac{1}{n} \frac{\dfrac{P_{i+1} - P_i}{(X_{i+1} - X_i)^2} + \dfrac{P_i - P_{i-1}}{(X_i - X_{i-1})^2}}{\frac{1}{X_{i+1} - X_i} + \frac{1}{X_i - X_{i-1}}}, \tag{6.29}$$

for $i = 2, \ldots, N-1$.

The approximation of second term in (6.20), which we denote by T^2_i, is taken as

$$T^2_i = \frac{6(n-1)}{\frac{n^2}{2}(X_{i+1} - X_{i-1})} \left[\left(\frac{(U_{i+1})^{n/2} - (U_i)^{n/2}}{X_{i+1} - X_i} \right)^2 - \left(\frac{(U_i)^{n/2} - (U_{i-1})^{n/2}}{X_i - X_{i-1}} \right)^2 \right], \tag{6.30}$$

for $i = 2, \ldots, N-1$, while the approximation of the third term, denoted T_i^3, is given by

$$T_i^3 = \frac{27(n-1)(n-2)}{2n^3} \left(\frac{(U_{i+1})^{n/3} - U_{i-1}^{n/3}}{X_{i+1} - X_{i-1}} \right)^3, \tag{6.31}$$

for $i = 2, \ldots, N-1$.

The velocity at node i, is then the sum of the three terms (6.29), (6.30) and (6.31),

$$V_i = T_i^1 - T_i^2 + T_i^3,$$

for $i = 2, \ldots, N-1$.

We then approximate the boundary velocity V_1 and V_N using an extrapolation from nearby velocity approximations, which we would expect to be feasible as all terms in the alternative velocity are smooth and do not involve any indeterminate multiplications as in the previous expression of the velocity (3.14).

6.10.2 Provisional Results

We now provide provisional results which have been generated using an implementation of the MPCM featuring the alternative velocity described in §6.10.1. We shall repeat the experiments performed in §6.6.1, using the same choices of n and α and with the same number of nodes in the meshes.

The initial computed velocities V_N^0 using the alternative velocity are given in Tables 6.8–6.10 for the finite advance, finite retreat and waiting-time cases respectively,

It is clear from Tables 6.8–6.10 that there is a marked difference between the results generated using the alternative velocity method and the original results presented in Tables 6.1–6.3. In the finite advance and retreat cases in particular the computed boundary velocities are much closer to the expected values, while the waiting time case has produced results of a similar magnitude to the previous velocity implementation.

The finite retreat case (Table 6.9) highlights an interesting feature of the results. In both cases the $N = 21$ results are the least accurate of the meshes, as may be expected since the mesh is coarsest for this value of N. As the number of mesh points increases the boundary values approach the expected values, before slowly moving away as we reach $N = 1281$.

N	$n = 8/5$, $\alpha = 15/8$	$n = 2$, $\alpha = 3/2$
21	0.0502	0.0289
41	0.0445	0.0311
81	0.0460	0.0321
161	0.0468	0.0324
321	0.0470	0.0325
641	0.0471	0.0325
1281	0.0471	0.0325
v_b^0	≈ 0.04	0.03

Table 6.8: Initial boundary velocities for the finite advance experiments using the alternative velocity. The expected boundary velocity is given in the final row for each case.

N	$n = 6/5$, $\alpha = 5/2$	$n = 15/11$, $\alpha = 11/5$
21	-0.7990	-0.1323
41	-0.9314	-0.1755
81	-0.9397	-0.1762
161	-0.9386	-0.1750
321	-0.9380	-0.1746
641	-0.9378	-0.1745
1281	-0.9377	-0.1744
v_b^0	≈ -0.95	≈ -0.18

Table 6.9: Initial boundary velocities for the finite retreat experiments using the alternative velocity. The expected boundary velocity is given in the final row for each case.

N	$n = 5/2$, $\alpha = 5/2$	$n = 2$, $\alpha = 2$
21	-0.0207	-0.0079
41	5.53×10^{-4}	6.75×10^{-5}
81	1.84×10^{-4}	9.22×10^{-4}
161	7.77×10^{-6}	3.96×10^{-4}
321	-7.12×10^{-7}	1.23×10^{-4}
641	-1.87×10^{-7}	3.41×10^{-5}
1281	-2.65×10^{-8}	8.94×10^{-6}
v_b^0	0	0

Table 6.10: Initial boundary velocities for the waiting-time experiments using the alternative velocity. The expected boundary velocity is given in the final row for each case.

In an attempt to understand this difference, we investigate the computed boundary velocity for one of the experiments to determine whether oscillations are still appearing close to the boundary. As in §6.6.1, we select the finite retreat case (with $n = 6/5$, $\alpha = 15/8$ and $n\alpha = 3$) and plot the computed initial velocity for the final five mesh points in the $N = 81$ case and the final 20 mesh points of the mesh in the $N = 1281$ case. These computed velocities are given in Figure 6.8, along with the exact $v^0(x)$ calculated using (6.20).

From Figure 6.8 it is clear that there are still oscillations forming close to the boundary, which affects the final computed boundary velocity. These oscillations are much smaller in magnitude than those seen in Figure 6.1 however, which explains why the computed velocities are much closer to the expected values. We also observe that the oscillations are larger for the $N = 1281$ mesh, which can explain why the computed boundary velocity is slowly moving away from the expected values.

It is likely that the oscillations observed here may arise from the numerical approximation of the terms in (6.20) as opposed to the presence of the singularity in the original experiments in (§6.6.1). The finite difference approximation of the derivatives for mesh

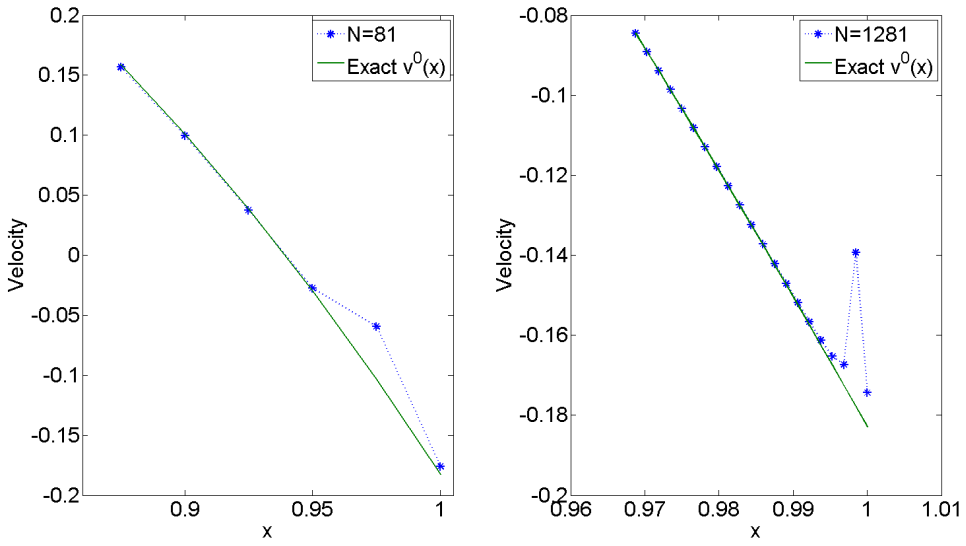

Figure 6.8: The computed alternative velocity V_i^0 for the $n = 6/5$, $\alpha = 15/8$ finite retreat case. The left plot shows the final five mesh points for the $N = 81$ mesh, while the right plot shows the $N = 1281$ mesh plotted over the final 20 mesh points. The green line in each plot is the exact $v^0(x)$ calculated from (6.20)

points close to the boundary could be a source of numerical error arising in the method. As the number of mesh points increases, these solution values will become smaller and could cause the oscillations observed in Figure 6.8.

Overall, the alternative velocity expression appears to be a great improvement over the previous velocity expression for choices of $n > 1$. By avoiding approximating the $q(x,t)$ function the issues with point-wise multiplications that caused difficulties in the previous implementation of the velocity are avoided. The method is still limited to the case where $n\alpha \geq 3$ however.

6.10.3 The Alternate Velocity near the Boundary as N increases

The provisional results given in Tables 6.8 and 6.9 illustrate a potential issue which can occur as the value of N increases. In both Tables 6.8 and 6.9 the computed boundary velocity approached the expected value as N remained small ($N = 21, 41$ for example), before slowly beginning to move away from the expected value as N became larger.

In order to explain this behaviour, we note that the finite difference schemes (6.28)–(6.31) involve terms containing factors $X_{i+1} - X_i$ or $X_i - X_{i-1}$. As the value of N increases, the size of these factors will decrease, which (when coupled with the value of U_i approaching zero as we approach the boundary) can cause rounding errors to have a greater impact on the accuracy of the resulting velocity. This effect can be seen in Figure 6.8, with the $N = 1281$ approximate velocity being more unstable than the $N = 81$ for points close to the boundary.

This information suggests that the MPCM utilising the alternate velocity is best suited to smaller values of N in order to provide a more accurate representation of the velocity close to the boundary.

6.10.4 The Alternative Velocity when $n = 1$

If the implementation of the alternative velocity can be performed such that the results for $n > 1$ are converging, then it opens up possibilities for an alternative MPCM (the MPCMb, say). In this alternative MPCM, the velocity (6.20) when $n = 1$ has only one

term. It may be possible to modify the scheme used to approximate this term such that the MPCMb possesses the *S Property* over a single time step (if scale-invariant time stepping is used). This then results in a method which may be able to overcome some of the limitations of the original MPCM, while also possessing the *S Property*.

6.11 Summary of this Chapter

In this chapter we have examined the ability of the MPCM to model various initial boundary behaviours for the fourth-order problem (2.3)–(2.5) when $n > 1$. By considering initial conditions of the form (6.4) we were able to identify various different initial behaviours of the boundary (supporting results in Blowey et al. (2007)). This investigation highlights a range of situations in which the MPCM is unable to track the moving boundary.

Further investigation of the behaviour of the $q(x,t)$ function suggested that this was due to the singularity present at the boundary. This investigation highlighted the issues present in writing the fourth-order PDE (2.3) as a pair of second-order equations. It was found that these issues occurred in both the MPCM and FEMPCM, in spite of the weak forms used in the latter.

A discussion on the initial distribution of mesh points in the MPCM/FEMPCM was presented. All numerical results presented in this thesis are obtained on an initially uniform mesh. This can cause the method to be unable to resolve significant features of the solution (for points close to the boundary). Alternative means of initially distributing the mesh are discussed, highlighting advantages and disadvantages of each.

Two possible strategies for overcoming the problems discovered in the MPCM were then proposed. The first of these was a hybrid numerical method which is potentially able to model an initially unbounded boundary velocity. An outline of the method was given, without numerical verification. The second strategy involved a direct expression for the velocity without recourse to the $q(x,t)$ function, using identities, and approximating this new expression numerically. The new implementation of the MPCM was then shown to produce correct qualitative boundary velocities when compared to the

original MPCM.

In the final chapter we shall provide a summary of the thesis, including conclusions drawn throughout. We shall also provide a discussion on potential future directions of the work.

Chapter 7

Conclusions and Further Work

In this final chapter we shall summarise the work in this thesis and present conclusions that can be drawn from the progress made. We then provide an outline of future directions in which the subject can be taken, either related directly to the work covered in the thesis or natural extensions to the work which have not been previously covered.

7.1 Summary

In Chapter 1 we outlined the main aims of the thesis. These were to:

- Extend the finite difference work of Parker (2010) for a particular second-order problem to higher orders (fourth and sixth) as well as to more general second-order problems using a finite difference implementation of the velocity-based conservation method. These implementations, which use a scale-invariant time stepping scheme, are theoretically able to propagate similarity solutions forward in time to essentially within rounding error of the exact solution over a single time step. We term this property of the numerical method the *S Property*.

- Produce an implementation of the BHJ method using higher order basis functions and scale-invariant time stepping in order to propagate similarity solutions forward in time to within rounding error. In principle, this finite element implementation

is able to approximate solutions to those second, fourth and sixth-order problems which admit such similarity solutions.

- Use these implementations to provide a comparison with the work in Blowey et al. (2007) in order to verify whether the moving-mesh method is able to replicate the small-time behaviours of solutions (computed in that paper using a fixed-mesh method). It was hoped that the method can not only model such behaviour, but do so at a reduced computational expense.

In Chapter 2 we introduced the nonlinear diffusion problems to be studied in the thesis, with the emphasis being on the fourth-order problem (2.3)–(2.5). A study of the literature relating to these problems was provided, which highlighted the known behaviour of solutions to the problems as well as the various studies that have been made. The chapter also described the properties possessed by the nonlinear diffusion problems, such as mass conservation and the existence of similarity solutions to the problems.

We then introduced the subject of using numerical methods for finding approximate solutions to nonlinear diffusion problems, using both moving-mesh methods and fixed-mesh methods. This was presented in the context of one-dimensional (in space) implementations, although some mention of methods in higher dimensions was made in order to highlight the body of work that exists in the wider literature.

Chapter 3 then focused on a particular class of moving-mesh numerical method; the velocity-based method. This class of method was central to the work done in the thesis and as such is given a larger role in the discussion of the various methods. We described a selection of existing velocity-based methods, including the conservation-based methods on which the MPCM is based. The MPCM is a finite difference implementation of a conservation method which generalises the work of Parker (2010) for a specific second-order problem and which has been extended to work on the problems considered in this thesis. Another influence on the MPCM is the BHJ method of Baines, Hubbard and Jimack (Baines et al. (2005, 2006, 2011)) which has been used successfully in solving some of the problems covered in this thesis.

The majority of Chapter 3 focused on introducing the MPCM for the various orders of problem considered. The conservation method uses local mass conservation in order to establish the velocities of interior mesh points (with extrapolation used to obtain the velocity at the boundary points), which is then used to move the mesh in time using a suitable time stepping method. The local mass conservation is then used to recover the solution algebraically on the updated mesh at the next time step. The chapter finishes with a definition of the *S Property*, which we demonstrate in later chapters that the MPCM possesses in certain circumstances.

In Chapter 4 the MPCM was modified for the fourth order problem (2.3)–(2.5) with $n = 1$ in order to demonstrate that the method possesses the *S Property* when the initial condition coincides with a similarity solution and a scale-invariant time stepping scheme is used. By modifying the schemes used, the truncation error of each step of the method was shown to be equal to zero over a single time step. This property was verified by numerical results, although multiple time steps highlighted a build up of rounding error incurred by the time stepping method. This build up of error was investigated and bounded.

Chapter 4 also provided insight into how the MPCM possesses the *S Property* for the second-order problem (2.7)–(2.9) for any choice of n. For the sixth-order problem (2.10)–(2.12) with $n = 1$, the required modifications to the schemes in the MPCM are presented. Numerical verification is not provided for the sixth-order problem due to a lack of a working program code.

A departure from finite difference implementations took place in Chapter 5, with the description of the FEMPCM. This required the introduction of weak forms of the various equations in the different problems considered. We described the BHJ method (from which the FEMPCM originated) with a few minor alterations for the fourth-order problem (2.3)–(2.5), using piecewise linear basis functions throughout. We then demonstrated that by choosing the basis functions in the FEMPCM to be of the same order as the exact solution, the method for the fourth-order problem (2.3)–(2.5) with $n = 1$ could be shown to possess the *S Property*. Numerical results were presented which

verify the possession of the *S Property*, with a build up of rounding error present as in the MPCM.

We then described the steps required to modify the FEMPCM for the second-order problem (2.7)–(2.9) with any n and the sixth-order problem (2.10)–(2.12) with $n = 1$ so that they possess the *S Property*.

In Chapter 6 we explored the fourth-order problem (2.3)–(2.5) for choices of $n > 1$, for which there are no explicit similarity solutions which can be used to verify results obtained from the MPCM. It was hoped that although the MPCM does not possess the *S Property* in this instance, the method would still be able to model various expected small-time behaviours of the solutions. In particular, a comparison between results presented in Blowey et al. (2007) and results from the MPCM was proposed.

Examining the initial boundary velocities for the fourth-order problem (2.3)–(2.5) with $n > 1$ and for initial conditions of the form (6.4) identified a regime in (n, α) space such that the boundary would be expected to move with an initially unbounded velocity. In this instance the MPCM would not be able to model such behaviour (as any numerical method attempting to approximate the unbounded boundary velocity would not). A range of α was also identified for which the function $q(x,t)$ or $\frac{\partial q}{\partial x}$ would become unbounded at the boundary. The current method of calculating the boundary velocity used in the MPCM would therefore not be able to accurately approximate such values due to the appearance of oscillations at points close to the boundary, which hinders the effectiveness of the method.

An investigation into whether the FEMPCM (incorporating the $q(x,t)$ function) experiences the same issues with calculating the boundary velocity was made. It was shown that the oscillations present in the MPCM also occur in the FEMPCM. Therefore, despite the FEMPCM featuring integral forms covering the whole domain (as opposed to merely point-wise representation as in the MPCM), the singularity at the moving boundary still causes problems in the FEMPCM.

A discussion on the initial mesh distribution for the MPCM or FEMPCM was then made. A uniformly distributed mesh was used for all numerical results in this thesis.

For problems in which the method would be expected to possess the *S Property*, this uniform mesh is suitable for producing results of the required accuracy. In the case of the problems examined in Chapter 6, the initial mesh may benefit from being focused closer to the boundary (with alternative means of initially distributing the mesh points are discussed).

A possible means of alleviating the issue of modelling an initially unbounded boundary velocity was proposed in the form of a hybrid numerical method, combining a fixed-mesh method with the MPCM. At present this method is unverified, but a description of how this could be implemented is given in §6.9, along with a critique of the issues to be addressed in the implementation.

Finally, a formulation with an alternative expression for the velocity was proposed which, when implemented numerically, alleviates the issues experienced with approximating the $q(x,t)$ function (for $n\alpha \geq 3$). This alternative velocity does not contain any of the products of terms which were shown to be problematic in the previous implementation. Numerical results show that this alternative velocity produces much improved initial boundary velocities and is therefore a potential solution to the problems experienced previously.

7.2 Conclusions

In conclusion, we have demonstrated both advantages and disadvantages in the use of the MPCM for solving the type of nonlinear diffusion problems discussed in this thesis. The ability of the MPCM to propagate similarity solutions forward in time to the accuracy demonstrated in Chapter 4 is noteworthy and the manner in which it can apply to different orders of problem is also of interest. The problems highlighted in the case where the similarity solution is not known *a-priori* can be considered a major disadvantage however, although some promising results in this area were observed.

Specifically, we conclude:

- *The MPCM possesses the* S Property *(and therefore propagates similarity solutions*

forward in time to within rounding error if a scale-invariant time stepping scheme is used) for problems of various orders (second/fourth/sixth) when the numerical schemes are carefully modified.

In chapter 4 the numerical schemes of the MPCM were modified from the schemes originally described in Chapter 3. These modified schemes were shown to have zero local truncation error over a single time step when a similarity solution is used as an initial condition and through the use of a scale-invariant time stepping scheme. In practice we observed a build up of rounding error which was shown to be due to the time stepping part of the method and this error was bounded. This level of accuracy is present even in a small number of nodes, which reduces the computational cost of the method.

- *The FEMPCM can be implemented through careful choice of approximation spaces so that it can also possess the* S Property*, producing a solution valid over the whole of the domain at an increased computational cost (when compared to the MPCM which is only valid at the mesh points).*

  Chapter 5 demonstrated that an implementation of the FEMPCM can also be constructed which possesses the *S Property* for the same problems as previously mentioned. As in the MPCM, a build up of rounding error is observed in the numerical results, which is attributable to the time stepping method.

- *There are a range of situations in which the MPCM (and FEMPCM) is unable to accurately model the behaviour of the moving boundary of the fourth-order problem* (2.3)–(2.5)*, either due to the* $q(x,t)$ *function or its derivative becoming unbounded at the boundary or due to the boundary being expected to move with an initially unbounded velocity.*

  In Chapter 6 we explored the applications of the MPCM to the fourth-order problem (2.3)–(2.5) for more general cases, since the ability of the method to produce solutions for general n and for more general initial conditions is of interest to the wider community. It was noted that for initial conditions of the form (6.4) there

are choices of initial condition in which either the velocity or the functions $q(x,t)$ or $\frac{\partial q}{\partial x}$ can become unbounded at the boundary.

In the case of the initial boundary velocity becoming unbounded any numerical method attempting to calculate this velocity will fail. In particular, the MPCM is unable to be applied since it requires a finite velocity to update the mesh points. If $\frac{\partial q}{\partial x}$ becomes unbounded at the boundary it will not be accurately approximated by the MPCM, and the point-wise multiplication of u^{n-1} and $\frac{\partial q}{\partial x}$ in the velocity near the boundary will not be accurately performed, even if the product of these two terms is finite. This then causes the boundary velocity to be inaccurately approximated and could cause the boundary to behave in a different manner to that expected (when compared with results from King (2001) and Blowey et al. (2007)). These issues were also demonstrated with the FEMPCM.

- *Proposed methods to combat the issues exhibited by the MPCM show promise; A hybrid numerical method may allow an initially unbounded velocity to be modelled but is unverified, while an alternative velocity expression could allow for more accurate boundary velocities to be calculated when implemented into an alternative MPCM.*

  Chapter 6 also introduced two proposed alternatives to the MPCM which may potentially alleviate the issues reported in the previous bullet point. Each alternative tackles a separate issue, with the first of these (a hybrid numerical method) using a combination of a fixed-mesh method and the MPCM to track an initially unbounded boundary velocity. The fixed-mesh portion of the hybrid method uses a scheme in which the boundary may advance at most one mesh point over a single time step, which allows the method to approximate the solution until such a time as the boundary velocity is able to be accurately approximated by the MPCM. The solution at the end of the fixed-mesh time window is then used as the initial condition for the MPCM. At present the hybrid method has not been able to be implemented successfully.

The second alternative involves avoiding the approximation of the $q(x,t)$ function entirely by rewriting the velocity expression using an identity. The MPCM approximates the velocity using a point-wise multiplication which we have demonstrated to be flawed in some cases. This alternative velocity expression avoids any multiplications which could cause such issues in the resulting velocity approximation and can then be numerically approximated. This alternative velocity has been shown to compute velocities with reduced oscillations when compared with those obtained by the MPCM and is a promising area of future research.

7.3 Future Work

The results achieved in this thesis, particularly those in Chapter 6, have highlighted the potential for further research on this topic in a number of areas. There are also some areas which have not been discussed in this thesis but could be pursued in the future. We shall discuss some of these areas in this section.

The alternative velocity expression (6.20) produces a different implementation of the MPCM (the MPCMb, say) which warrants further study, particularly in light of encouraging preliminary results. It would be of interest to see whether such an implementation is able to produce convergent boundary trajectories which can then be compared to those presented in Blowey et al. (2007). If this comparison can be made then the MPCMb could produce approximate solutions to the fourth-order problem (2.3)–(2.5) which match the small time behaviours expected from the existing literature, but at a fraction of the computational cost.

As an additional consideration, the current implementation of the MPCMb in §6.10 uses low-order schemes to approximate each terms. If the schemes were modified such that in the case when $n = 1$ the schemes used to calculate the velocity are of higher-order and are exact for quartic $u(x,t)$ and linear $v(x,t)$, then it could be shown that this implementation possesses the *S Property* in the l^∞ norm. The MPCMb would then be a method which possesses the main strength of the MPCM (possession of the *S Property*), with the ability to accurately model boundary behaviours for $n\alpha \geq 3$.

Implementation of the hybrid numerical method is also a possible future research avenue, as discussed in §6.9. The main issue with the method is the time point at which the method switches from a fixed-mesh to a moving-mesh method, as this is not obvious *a-priori*. This value may also be dependent upon the value of n and the initial condition used, since it is intended that the fixed-mesh part of the method be run until the boundary velocity is able to be approximated accurately by the moving-mesh portion.

There is also more work to be achieved on the current implementations of the MPCM and FEMPCM, particularly with regard to the sixth-order problem (2.10)–(2.12). Numerical results were omitted from this thesis for the MPCM as a working program code was not available. The method as described in Chapter 4 is expected to possess the *S Property*, but currently the program code is unable to approximate the quartic $p(x, s)$ function to the required accuracy.

Finally, a natural extension to the work in this thesis is to move into higher spatial dimensions. A logical extension into two spatial dimensions would be an implementation of the FEMPCM, since two-dimensional finite element methods have been used successfully in the past (such as the BHJ method of Baines et al. (2005, 2006, 2011)). Implementing the FEMPCM using the correct approximation spaces should enable the FEMPCM to possess the *S Property* in higher dimensions, albeit with increased complexity in the implementation of higher-order basis functions.

Bibliography

D.G. Aronson. The porous media equation. In A. Fasano and M. Primicerio, editors, *Nonlinear Diffusion Problems*, volume 1224 of *Lecture Notes in Mathematics*, pages 1–46. Springer-Verlag, Berlin, 1986.

M.J. Baines, M.E. Hubbard, and P.K. Jimack. A moving mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries. *Applied Numerical Mathematics*, 54:450–469, 2005.

M.J. Baines, M.E. Hubbard, P.K. Jimack, and A.C. Jones. Scale-invariant moving finite elements for nonlinear partial differential equations in two dimensions. *Applied Numerical Mathematics*, 56:230–252, 2006.

M.J. Baines, M.E. Hubbard, and P.K. Jimack. Velocity-Based Moving Mesh Methods for Nonlinear Partial Differential Equations. *Commun. Comput. Phys*, 10:509–576, 2011.

G.I. Barenblatt. On some unsteady motions of a liquid and gas in a porous medium. *Prikl. Mat. Mekh*, 16(1):67–78, 1952.

J.W. Barrett, J.F. Blowey, and H. Garcke. Finite element approximation of a fourth order nonlinear degenerate parabolic equation. *Numerische Mathematik*, 80:525–556, 1998.

J.W. Barrett, S. Langdon, and R. Nurnberg. Finite element approximation of a sixth

order nonlinear degenerate parabolic equation. *Numerische Mathematik*, 96:401–434, 2004.

T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: an overview and recent developments. *Computer methods in applied mechanics and engineering*, 139(1):3–47, 1996.

E. Beretta, M. Bertsch, and R. Dal Passo. Nonnegative solutions of a fourth-order nonlinear degenerate parabolic equation. *Arch. Rational Mech. Anal.*, 129:175–200, 1995.

F. Bernis. Finite speed of propagation for thin viscous flows when $2 \leq n \leq 3$. *C.R. Acad. Sci. Paris Sér. I Math*, 322:1169–1174, 1996a.

F. Bernis. Finite speed of propagation and continuity of interface for thin viscous flows. *Adv. Differential Equations*, 1:337–368, 1996b.

F. Bernis and A. Friedman. Higher order nonlinear degenerate parabolic equations. *Journal of Differential Equations*, 83:179–206, 1990.

F. Bernis, L.A. Peletier, and S.M. Williams. Source type solutions of a fourth order nonlinear degenerate parabolic equation. *Nonlinear Analysis, Theory, Methods and Applications*, 18:217–234, 1992.

F. Bernis, J. Hulshof, and J.R. King. Dipoles and similarity solutions of the thin film equation in the half-line. *Nonlinearity*, 13:413–439, 2000.

A.L. Bertozzi. Symmetric singularity formation in lubrication-type equations for interface motion. *SIAM J. Appl. Math.*, 56:681–714, 1996.

A.L. Bertozzi and M. Pugh. The lubrication approximation for thin viscous films: Regularity and long time behaviour of weak solutions. *Comm. on Pure and Applied Mathematics*, 49:85–123, 1996.

K.W. Blake. *Moving mesh methods for non-linear parabolic partial differential equations.* PhD thesis, University of Reading, UK., 2001.

J.F. Blowey, J.R. King, and S. Langdon. Small and waiting time behaviour of the thin film equation. *SIAM J. Appl. Math.*, 67:1776–1807, 2007.

P. Bochev, G. Liao, and G. dela Pena. Analysis and computation of adaptive moving grids by deformation. *Numerical Methods for Partial Differential Equations*, 12(4): 489–506, 1996.

J. Bouwe Van Den Berg, M. Bowen, J.R. King, and M.M.A. El-Sheikh. The self-similar solution for draining in the thin film equation. *Euro. Jnl of Applied Mathematics*, 15: 329–346, 2004.

M. Bowen and J.R. King. Asymptotic behaviour of the thin film equation in bounded domains. *Euro. J. Appl. Math.*, 12:135–157, 2001.

M. Bowen, J. Hulshof, and J.R. King. Anomalous exponents and dipole solutions for the thin film equation. *SIAM J. Appl. Math.*, 62:149–179, 2001.

C.J.W. Breward, H.M. Byrne, and C.E. Lewis. The role of cell-cell interactions in a two-phase model for avascular tumour growth. *J. Math. Biol.*, 45:125–152, 2002.

J. Buckmaster. Viscous sheets advancing over dry beds. *J. Fluid. Mech.*, 81:735–756, 1977.

C.J. Budd and M.D. Piggott. The geometric integration of scale-invariant ordinary and partial differential equations. *Journal of Computational and Applied Mathematics*, 128:399–422, 2001.

C.J. Budd and J.F. Williams. Moving mesh generation using the parabolic monge-ampere equation. *SIAM Journal on Scientific Computing*, 31(5):3438–3465, 2009.

C.J. Budd, W. Huang, and R.D. Russell. Moving mesh methods for problems with blow-up. *SIAM Journal on Scientific Computing*, 17(2):305–327, 1996.

C.J. Budd, G.J. Collins, W.Z. Huang, and R.D. Russell. Self-similar numerical solutions of the porous medium equation using moving mesh methods. *Philosophical*

Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 357:1047–1077, 1999.

C.J. Budd, B. Leimkuhler, and M.D. Piggott. Scaling invariance and adaptivity. *Applied numerical mathematics*, 39(3):261–288, 2001.

C.J. Budd, W. Huang, and R.D. Russell. Adaptivity with moving grids. *Acta Numerica*, 18:111–241, 2009.

C.J. Budd, M.J.P. Cullen, and E.J. Walsh. Monge–ampére based moving mesh methods for numerical weather prediction, with applications to the eady problem. *Journal of Computational Physics*, 236:247–270, 2013.

M. Burger, J.A. Carrillo, and M-T. Wolfram. A mixed finite element method for nonlinear diffusion equations. *Kinetic and Related Models*, 3:59–83, 2010.

W. Cao, W. Huang, and R.D. Russell. A moving mesh method based on the geometric conservation law. *SIAM J. Sci. Comput.*, 24:118–142, 2002.

N.N. Carlson and K. Miller. Design and application of a gradient-weighted moving finite element code i: in one dimension. *SIAM Journal on Scientific Computing*, 19 (3):728–765, 1998a.

N.N. Carlson and K. Miller. Design and application of a gradient-weighted moving finite element code ii: in two dimensions. *SIAM Journal on Scientific Computing*, 19(3): 766–798, 1998b.

B. Dacorogna and J. Moser. On a pde involving the jacobian determinant. *Ann. I. H. Poincaré C*, 7:1–26, 1990.

J.A. Diez, L. Kondic, and A. Bertozzi. Global models for moving contact lines. *Physical Review E*, 63:011208, 2000.

J.C.M. Duque, R.M.P. Almeida, and S.N. Antontsev. Convergence of the finite element method for the porous media equation with variable exponent. *SIAM Journal on Numerical Analysis*, 51(6):3483–3504, 2013.

E.B. Dussan and S.H. Davis. On the motion of a fluid-fluid interface along solid surface. *J. Fluid Mech.*, 65:71–95, 1974.

J.D. Evans, M. Vynnycky, and S.P. Ferro. Oxidation-induced stresses in the isolation oxidation of silicon. *Journal of engineering mathematics*, 38(2):191–218, 2000.

J.D. Evans, V.A. Galaktionov, and J.R. King. Source-type solutions of the fourth-order unstable thin film equation. *Euro. Jnl of Applied Mathematics*, 18:273–321, 2007a.

J.D. Evans, V.A. Galaktionov, and J.R. King. Unstable sixth-order thin film equation: I. blow-up similarity solutions. *Nonlinearity*, 20:1799–1841, 2007b.

J.D. Evans, V.A. Galaktionov, and J.R. King. Unstable sixth-order thin film equation: Ii. global similarity patterns. *Nonlinearity*, 20:1843–1881, 2007c.

J.C. Flitton and J.R. King. Moving-boundary and fixed-domain problems for a sixth-order thin-film equation. *Euro. Jnl of Applied Mathematics*, 15:713–754, 2004.

L. Giacomelli, H. Knüpfer, and F. Otto. Smooth zero-contact-angle solutions to a thin-film equation around the steady state. *J. Differential Equations*, 245:1454–1506, 2008.

H.P. Greenspan. On the motion of a small viscous droplet that wets a surface. *J. Fluid Mech.*, 84:125–143, 1978.

G. Grün. On the convergence of entropy consistent schemes for lubrication type equations in multiple space dimensions. *Mathematics of Computation*, 72:1251–1279, 2003.

G. Grün and M. Rumpf. Nonnegativity preserving convergent schemes for the thin film equation. *Numerische Mathematik*, 87:113–152, 2000.

G. Grün and M. Rumpf. Simulation of singularities and instabilities arising in thin film flow. *European J. Appl. Math.*, 12:293–320, 2001.

D. Huang, Q. Yang, and S. Zhou. Group properties and invariant solutions of a sixth-order thin film equation in viscous fluid. *Journal of Mathematical Physics*, 54:013510, 2013.

W. Huang and R.D. Russell. A high dimensional moving mesh strategy. *Applied Numerical Mathematics*, 26(1):63–76, 1997.

W. Huang and R.D. Russell. Moving mesh strategy based on a gradient flow equation for two-dimensional problems. *SIAM Journal on Scientific Computing*, 20(3):998–1015, 1999.

W. Huang and R.D. Russell. *Adaptive Moving Mesh Methods*. Springer, New York, 2011.

W. Huang, Y. Ren, and R.D. Russell. Moving Mesh Partial Differential Equations (MMPDES) Based on the Equidistribution Principle. *SIAM J. Numer. Anal.*, 31: 709–730, 1994.

M. Hubbard, M.J. Baines, and P.K. Jimack. Consistent dirichlet boundary conditions for numerical solution of moving boundary problems. *Applied Numerical Mathematics*, 59:1337–1353, 2009.

J. Hulshof and A.E. Shishkov. The thin film equation with $2 \leq n < 3$: Finite speed of propogation in terms of the l^1-norm. *Adv. Differential Equations*, 3:625–642, 1998.

H. Jiang and N. Wei-Ming. On steady states of van der waals force driven thin film equations. *Euro. J. Appl. Math.*, 18:153–180, 2007.

A. Jüngel and R. Pinnau. A positivity-preserving numerical scheme for a nonlinear fourth order parabolic system. *SIAM J. Numer. Anal.*, 39:385–406, 2001.

W.L. Kath and D.S. Cohen. Waiting-time behavior in a nonlinear diffusion equation. *Studies in Applied Mathematics*, 67:79–105, 1982.

J.R. King. *Mathematical aspects of semiconductor process modelling*. PhD thesis, University of Oxford, 1986.

J.R. King. The isolation oxidation of silicon. *SIAM J. Appl. Math.*, 49:264–280, 1989a.

J.R. King. The isolation oxidation of silicon: the reaction-controlled case. *SIAM J. Appl. Math.*, 49:1064–1080, 1989b.

J.R. King. Two generations of the thin film equation. *Mathematical and Computer Modelling*, 34:737–756, 2001.

J.R. King and M. Bowen. Moving boundary problems and non-uniqueness for the thin film equation. *Euro. J. Appl. Math.*, 12:321–356, 2001.

J.R. King and R.M. Taranets. Asymmetric travelling waves for the thin film equation. *J. Math. Anal. Appl.*, 404:399–419, 2013.

B.F. Knerr. The porous medium equation in one dimension. *Trans. Amer. Math. Soc.*, 234:381–415, 1977.

A.A. Lacey. Initial motion of the free boundary for a non-linear diffusion equation. *IMA Journal of Applied Mathematics*, 31:113–119, 1983.

A.A. Lacey, J.R. Ockendon, and A.B. Tayler. "Waiting-Time" Solutions of a Nonlinear Diffusion Equation. *SIAM J. Appl. Math.*, 42:1252–1264, 1982.

T.E. Lee. *Modelling time-dependent Partial differential equations using a moving mesh approach based on conservation.* PhD thesis, University of Reading, UK., 2011.

T.E. Lee, M.J. Baines, S. Langdon, and M.J. Tindall. A moving mesh approach for modelling avascular tumour growth. *Applied Numerical Mathematics*, 72:99–114, 2013.

G. Liao and D. Anderson. A new approach to grid generation. *Applicable analysis*, 44 (3-4):285–298, 1992.

Z. Ma, H. Chen, and C. Zhou. A study of point moving adaptivity in gridless methods. *Computer Methods in Applied Mechanics and Engineering*, 197:1926–1937, 2008.

K. Miller. Moving finite elements. ii. *SIAM Journal on Numerical Analysis*, 18(6): 1033–1057, 1981.

K. Miller and R.N. Miller. Moving finite elements. i. *SIAM Journal on Numerical Analysis*, 18(6):1019–1032, 1981.

J. Moser. Volume elements of a riemann manifold. *T. Am. Math. Soc.*, 120:286–294, 1965.

M. Muskat. *The Flow of Homogeneous Fluids Through Porous Media.* McGraw-Hill, New York, 1937.

V.P. Nguyen, T. Rabczuk, S. Bordas, and M. Duflot. Meshless methods: a review and computer implementation aspects. *Mathematics and Computers in Simulation*, 79(3): 763–813, 2008.

F. Otto. Lubrication approximation with prescribed non-zero contact angle. *Comm. Partial Differential Equations*, 23:2077–2164, 1998.

J. Parker. An invariant approach to moving-mesh methods for PDEs. Master's thesis, University of Oxford, 2010.

D. Partridge. *Numerical Modelling of Glaciers: Moving Meshes and Data Assimilation.* PhD thesis, University of Reading, UK., 2013.

R.E. Pattle. Diffusion from an instantaneous point source with a concentration-dependent coefficient. *Q.J. Mech. Appl. Math.*, 12:407–409, 1959.

L.A. Peletier. The porous media equation. In H. Amman et. al., editor, *Applications of nonlinear analysis in the physical sciences*, pages 229–241. Pitman, 1981.

C.A. Perazzo and J. Gratton. Bounds of waiting-time in nonlinear diffusion. *Applied Mathematics Letters*, 17:1253–1259, 2004.

W.R. Smith, S.D. Howison, and D.F. Mayers. Numerical and asymptotic solution of a sixth-order nonlinear diffusion equation and related coupled systems. *IMA Journal of Applied Mathematics*, 57:79–98, 1996.

N.F. Smyth and J.M. Hill. High-Order Nonlinear Diffusion. *IMA Journal of Applied Mathematics*, 40:73–86, 1988.

E.A. Socolovsky. On the numerical approximation of finite speed diffusion problems. *Numerische Mathematik*, 53:97–105, 1988.

J.L. Vasquez. *The Porous Medium Equation: Mathematical Theory*. Oxford University Press, 2007.

L. Zhornitskaya and A.L. Bertozzi. Positivity-preserving numerical schemes for lubrication-type equations. *SIAM J. Numer. Anal.*, 37:523–555, 2000.