MOVCOL4: A Moving Mesh Code for Fourth-Order Time-Dependent Partial Differential Equations

R. D. Russell*, JF Williams† and X. Xu‡

October 19, 2005

Abstract

In this paper we develop and analyze a moving mesh code for the simulation of fourth-order PDEs based on collocation. The scheme is shown to enforce discrete conservation for problems written in a generalized conservation form. To demonstrate the breadth of applicability we present examples from both Cahn-Hilliard and thin-film type equations exhibiting metastable behaviour, finite-time solution blow-up, finite-time extinction and moving interfaces.

1 Introduction

Well-chosen model equations play an indispensable role in the study of complex patterns in physical, mechanical, chemical and biological systems. Classical model equations have typically been partial differential equations (PDEs) which are second-order in space because even though they are often much simpler than the full equations describing physical processes, they capture the essential features of certain specific problems. Familiar examples are Laplace's equation, the general heat equation, and the wave equation, designed to shed light on physical processes such as diffusion, dispersion, absorption or wave propagation, and their mutual interaction. Nevertheless, they are ultimately limited in their ability to describe certain complex features. To gain insight into the dynamics of rich spatial and temporal patterns in a wide range of physical and mechanical problems, fourth-order model equations and systems of fourth-order equations have been proposed.

Examples of physical applications of these include problems in thin film theory, lubrication theory, convection-explosion theory, flame and wave propagation, phase transition at critical Lipschitz points, and bi-stable systems. In recent years, fourth-order problems have also become important in image processing and for modelling diffusion processes in physics and material sciences. Thus, fourth-order terms are increasingly recognized as being significant in many physical models, and this has led to a burgeoning literature, including the recent book [37] which lists a number of models and references to the development of analytical techniques for the study of certain fourth-order problems.

Unlike for second-order equations, the theoretical study of higher-order equations, even semilinear ones, is far from complete. The main reason is that many effective methods used in treating second order equations, such as those based on maximum principles, variational structure and self-adjointness properties are not applicable for

*Department of Mathematics, Simon Fraser University, Burnaby, BC V5A 1S6, CANADA rdr@math.sfu.ca The author was supported in part by NSERC (Canada) through Grant OGP-0008781.

†Work completed while at the Mathematics Institute, University of Leiden, NL. Authors present position: Department of Mathematics, Simon Fraser University, Burnaby, BC V5A 1S6, CANADA jfw@math.sfu.ca

‡Department of Mathematics, Simon Fraser University, Burnaby, BC V5A 1S6, CANADA xxub@sfu.ca The author was supported in part by NSERC (Canada) through Grant OGP-0008781.

higher-order equations. As in other areas where the analytical theory is underdeveloped, many crucial features have been first explored by careful numerical experimentation. Examples include non-uniqueness in thin films [3], finite-time extinction in thin-film equations [10, 6], exact self-similar finite-time blow-up in higher-order semilinear parabolic equations [14], and solitary waves in nonlinear beam equations [22]. Generally, adaptive numerical methods must of necessity be brought to bear for the solution of these problems since important structures for many typical fourth-order problems occur at scales which change over the duration of the time period of interest, making a fixed grid inadequate to the task of efficiently obtaining an accurate final solution. While a number of numerical approaches have been used — e.g., see [7, 20, 15, 41, 4] — in our opinion the need to develop general numerical methods and software specifically for fourth-order problems is clear.

Generally speaking, there are three adaptive approaches for most numerical methods: h-refinement, whereby nodes are added to or removed from an existing mesh in such a way as to improve local grid resolution; p-refinement, whereby higher order schemes are used to improve local accuracy when the solution is sufficiently smooth; and r-refinement, whereby a fixed number of nodes are moved along with the solution over the domain. These strategies may of course be used singly or in combination.

We consider r-refinement methods [29], which have recently been successfully used for solving a number of differential equations that involve large solution variations like the ones arising in fourth-order problems — e.g., extinction, boundary layers, and blow-up. In particular, in this paper we discuss the construction and implementation of a high resolution moving collocation scheme for the adaptive approximation of fourth-order (in space) evolutionary partial differential equations. The numerical method used is a generalization of the one developed in [33]. The adaptive strategy is largely based upon the basic moving mesh methods in [31, 32] and found to be particularly effective for scaling invariant problems of a similar type to many important fourth-order problems. With our approach, the numerical schemes are applied directly to the fourth-order differential equations instead of converting the equations to lower order systems. This direct approach has proven useful in terms of efficiency and effectiveness for solving ordinary differential equations [1]. We believe it to be appropriate in the context of solving time dependent PDEs in one spatial dimension (1D) as well, for much the same reasons, and because added difficulties arise using lower order systems for moving mesh methods (cf. Section 6).

The collocation-type method we use is based upon a 7th degree Hermite spline basis since this is a natural order of spline approximation for fourth-order problems [2], as is a 3rd degree Hermite basis for second order problems [33]. The method is analyzed and the error analysis in [33] extended. The layout for this paper is as follows. In Sections 2 - 4 we introduce the moving mesh method and its spatial Gauss point collocation discretization. In Section 5 we describe the conservation properties of this method and analyze a related fully conservative strategy. Some properties of the resulting code MOVCOL4 as well as alternative methods for solving fourth-order PDEs are briefly discussed in Section 6. Finally, in Section 7 several problems are solved with MOVCOL4 which demonstrate the breadth of its applicability and the advantages of this approach.

2 1D r-adaptivity for PDEs

2.1 Equidistribution and optimal meshes

The main tool for optimal 1D adaptivity is to require that a local estimate of the error (e.g., the discretization error) be equalized over each computational cell [23], or *equidistributed*. If we define a *computational* variable ξ and a monitor function $M(x) > 0$, then the equidistributed mapping satisfies [31]

$$\int_0^{x(\xi)} M(s)ds = \xi \int_0^1 M(s)ds. \tag{2.1}$$

Although not written explicitly here, $M(x)$ is also a function of t and the solution $u(x,t)$ of the underlying physical PDE (and/or its derivatives). Without loss of generality, if M is normalized such that $\int M = 1$, then differentiating with respect to ξ gives the differential form of the equidistribution principle

$$Mx_\xi = 1. \tag{2.2}$$

Note that this specifies a condition on the Jacobian of the mapping $x(\xi)$. For an appropriately chosen monitor function, the equidistributed grid is optimal with respect to a given error norm [23].

Solving (2.2) directly in concert with a PDE leads to a coupled highly nonlinear differential algebraic equation (DAE) system with the mesh equations being algebraic. To solve this difficult numerical problem, various regularizations have been proposed. The one we use here involves solving so-called Moving Mesh PDEs (MMPDEs), so that (2.2) is only solved approximately in time [32]. While there are many different MMPDEs [31], here we discuss only two. The first, MMPDE6, is

$$x_{t\xi\xi} = -\frac{1}{\tau}(Mx_\xi)_\xi. \tag{2.3}$$

The positive parameter $\tau \ll 1$ is a relaxation time determining the time-scale over which a mesh converges to steady-state (2.2). The second, MMPDE4, is

$$(Mx_{t\xi})_\xi = -\frac{1}{\tau}(Mx_\xi)_\xi, \tag{2.4}$$

which has different scaling properties than MMPDE6, and suitability of one over another can be problem dependent. One can show that when solved exactly, neither produces mesh crossing, i.e., the mapping is well defined for all time [32].

A key to the success of an MMPDE method is the choice of monitor function. Monitor functions based on error control have been used in [4, 21], as have ones which cluster nodes based on qualitative solution properties, e.g., in [15, 41, 35]. A general methodology for constructing monitor functions to minimize errors in specified norms is presented in [30]. Scaling structure proves essential to many of the example problems listed here and elsewhere. One motivation for using a scale-invariant monitor function is the possibility of computing solutions with uniform error estimates, even for singular problems. For a general discussion of scale-invariant monitor functions and their role when using MMPDEs, see [15, 16, 18].

3 Moving collocation method

In the remainder of this paper, U and X will denote the numerical approximations for the spline solution parameters and the moving grid. The independent variables are the time t and the computational spatial variable ξ.

To solve time-dependent problems, we use a moving method of lines approach, first discretizing the physical PDE in space using Hermite collocation and the MMPDE using finite differences. Then the resulting system of ODEs in time is solved with an ODE solver. Our adaptive method is motivated by the desire to handle a variety of singular solution features for the problems under consideration. Because we are only considering problems in 1D, solving the MMPDE and physical PDE simultaneously for both the transformation $x(\xi,t)$ and physical solution $u(x(\xi,t),t)$ is computationally realistic. This avoids the need to interpolate the solution between different grids, and the method of lines discretization for the entire system may be integrated with standard software. An alternate strategy is mesh decoupling for which the physical and mesh PDEs are solved alternately — e.g., see [4]. Using Hermite collocation for the physical

PDE and only central finite differences for the mesh is also the approach taken in [33] with MOVCOL, a spline collocation code designed for second order problems. Its success is predicated on the fact that, as experience indicates, the mesh usually need not be computed as accurately as the solution to the physical PDE. This can be motivated by noting that the physical solution can, in theory, be solved on any grid. As long as the mesh nodes do not cross, errors in the computation of the grid do not directly correspond to errors in the approximation of the physical PDE.

The code MOVCOL has proven flexible and robust in solving a large class of second-order problems, including ones with finite time blow-up [13, 17, 19]. Our objective is to develop a method which is an extension of that approach for fourth-order problems. In fact, many of the strengths of the coupling of collocation and MMPDEs are greatly enhanced for higher-order problems.

Spline collocation has several advantages over a standard finite difference approximation: it affords a continuous representation of the solution and its first $(m-1)$ spatial derivatives, provides a higher order of convergence, is simple and flexible to program, easily handles boundary conditions involving high derivatives, and gives errors independent of local mesh grading. This last point is one of the most important points for a moving mesh method, particularly for higher-order problems. By using collocation, we are able to avoid the problem of approximating high order derivatives via differences over a wide non-uniform stencil (cf. [40]).

When considering a discretization with a coordinate transformation one can either write the PDE in terms of the physical variable x and discretize on a non-uniform mesh, or solve a transformed PDE in terms of the computational variable ξ on a uniform mesh. The latter approach is typical of moving finite difference approximations; however, with the flexibility of the collocation method it is simpler in 1D to discretize the PDE directly with respect to the physical variable x.

Secifically, suppose that at a time t the mesh

$$x_L(t) = X_1(t) < X_2(t) < ... < X_{N+1}(t) = x_R(t)$$

is the solution of the discretized MMPDE. The physical solution $u(x,t)$ is approximated by the piecewise 7th-order Hermite polynomial

$$\begin{aligned} U(x,t) = {} & U_i(t)L_{0,0}(s) + U_{x,i}(t)H_i(t)L_{0,1}(s) + U_{xx,i}(t)H_i^2(t)L_{0,2}(s) \\ & + U_{xxx,i}(t)H_i^3(t)L_{0,3}(s) + U_{i+1}(t)L_{1,0}(s) + U_{x,i+1}(t)H_i(t)L_{1,1}(s) \\ & + U_{xx,i+1}(t)H_i^2(t)L_{1,2}(s) + U_{xxx,i+1}(t)H_i^3(t)L_{1,3}(s), \end{aligned} \tag{3.1}$$

for $x \in [X_i(t), X_{i+1}(t)]$, $i = 1, ..., N$, where $U_i(t), U_{x,i}(t), U_{xx,i}(t)$ and $U_{xxx,i}(t)$ denote $U(X_i(t),t), U_x(X_i(t),t), U_{xx}(X_i(t),t)$ and $U_{xxx}(X_i(t),t)$, respectively. The local coordinate s is defined by

$$s = \frac{x - X_i(t)}{H_i(t)} \in [0,1], \qquad H_i(t) = X_{i+1}(t) - X_i(t).$$

The functions L_{ik} satisfy the relations

$$\frac{d^p}{dx^p} L_{ik}(x_j) = \begin{cases} 1 & \text{if } i = j \text{ and } k = p, \\ 0 & \text{otherwise,} \end{cases} \tag{3.2}$$

where for completeness we give

$$\begin{aligned} L_{0,0}(s) &= \left(20s^3 + 10s^2 + 4s + 1\right)(s-1)^4, & L_{0,1}(s) &= s\left(10s^2 + 4s + 1\right)(s-1)^4, \\ L_{0,2}(s) &= \frac{s^2}{2}(4s+1)(s-1)^4, & L_{0,3}(s) &= \frac{s^3}{6}(s-1)^4, \\ L_{1,0}(s) &= -\left(20s^3 - 70s^2 + 84s - 35\right)s^4, & L_{1,1}(s) &= s^4(s-1)\left(10s^2 - 24s + 15\right), \\ L_{1,2}(s) &= -\frac{s^4}{2}(s-1)^2(4s-5), & L_{1,3}(s) &= \frac{s^4}{6}(s-1)^3. \end{aligned} \tag{3.3}$$

On $x \in [X_i(t), X_{i+1}(t)]$, derivatives of U are determined by direct differentiation of (3.1), e.g.,

$$\begin{aligned} U_x(x,t) = \frac{1}{H_i}\Bigg(& U_i(t)\frac{dL_{0,0}}{ds} + U_{x,i}(t)H_i(t)\frac{dL_{0,1}}{ds} + U_{xx,i}(t)H_i^2(t)\frac{dL_{0,2}}{ds} \\ & + U_{xxx,i}(t)H_i^3(t)\frac{dL_{0,3}}{ds} + U_{i+1}(t)\frac{dL_{1,0}}{ds} + U_{x,i+1}(t)H_i(t)\frac{dL_{1,1}}{ds} \\ & + U_{xx,i+1}(t)H_i^2(t)\frac{dL_{1,2}}{ds} + U_{xxx,i+1}(t)H_i^3(t)\frac{dL_{1,3}}{ds}\Bigg), \end{aligned} \tag{3.4}$$

and

$$\begin{aligned} U_t(x,t) = & \frac{dU_i}{dt}L_{0,0} + \left(\frac{dU_{x,i}}{dt}H_i + U_{x,i}\frac{dH_i}{dt}\right)L_{0,1} + \left(\frac{dU_{xx,i}}{dt}H_i^2 + 2U_{xx,i}H_i\frac{dH_i}{dt}\right)L_{0,2} \\ & + \left(\frac{dU_{xxx,i}}{dt}H_i^3 + 3U_{xxx,i}H_i^2\frac{dH_i}{dt}\right)L_{0,3} + \frac{dU_{i+1}}{dt}L_{1,0} \\ & + \left(\frac{dU_{x,i+1}}{dt}H_i + U_{x,i+1}\frac{dH_i}{dt}\right)L_{1,1} + \left(\frac{dU_{xx,i+1}}{dt}H_i^2 + 2U_{xx,i+1}H_i\frac{dH_i}{dt}\right)L_{1,2} \\ & + \left(\frac{dU_{xxx,i+1}}{dt}H_i^3 + 3U_{xxx,i+1}H_i^2\frac{dH_i}{dt}\right)L_{1,3} - U_x(x,t)\left(\frac{dX_i}{dt} + s^{(i)}\frac{dH_i}{dt}\right). \end{aligned} \tag{3.5}$$

There are several natural ways to derive a system of ODEs for determining the unknowns $U_i(t)$, $U_{x,i}(t)$, $U_{xx,i}(t)$ and $U_{xxx,i}(t)$. The first we consider is collocation at the Gauss points. The second scheme is a conservative method, which is a natural alternative choice for many physical problems.

4 Gauss point collocation

Consider the PDE

$$u_t = \mathcal{N}(t, x, u, u_x, u_{xx}, u_{xxx}, u_{xxxx}), \quad t > 0,\ x \in I = (a, b), \tag{4.1}$$

supplemented with the initial condition

$$u(x, 0) = u_0(x), \quad x \in (a, b),$$

and separated boundary conditions

$$B_{a,1} = 0,\ B_{a,2} = 0,\ B_{b,1} = 0,\ B_{b,2} = 0, \tag{4.2}$$

where $B_{y,i} = B_{y,i}(t, y, U_t(y), U(y), U_x(y), U_{xx}(y), U_{xxx}(y))$. The standard collocation approach would be to solve for the $(4+1)(N+1)$ unknowns consisting of the solution U and its first three spatial derivatives at the mesh points and the $(N+1)$ mesh points $X_i(t)$ themselves. One enforces the collocation condition

$$U_t - \mathcal{N}(t, x, U, U_x, U_{xx}, U_{xxx}, U_{xxxx}) = 0 \tag{4.3}$$

at the 4 Gauss points in each interval. The MMPDE is approximated with central differences at the $(N-1)$ interior points, e.g. for $X_{t,\xi\xi} = -\frac{1}{\tau}\left(MX_\xi\right)_\xi$ use

$$X_{t,i-1} - 2X_{t,i} + X_{t,i+1} = -\frac{1}{\tau}\left(M_{i+1/2}(X_{i+1} - X_i) - M_{i-1/2}(X_i - X_{i-1})\right).$$

Here $M_{i+1/2}$ and $M_{i-1/2}$ are approximations to the average of the monitor function over cells $I_i = (x_i, x_{i+1})$ and I_{i-1}, respectively. Satisfying the boundary conditions (4.2) and requiring $X_1 = a$, $X_{N+1} = b$, this gives a system of $5(N+1)$ equations in $5(N+1)$ unknowns. From the standard ODE error analysis for a spline collocation solution U to (4.1) at a fixed time level t [2], the spatial error satisfies

$$|u(x) - U(x)| = \mathcal{O}(h^8) \quad \text{for all } x \in I, \quad \text{where } h = \max_i h_i.$$

Frequently in practice a PDE occurs in the form

$$u_t = \left(g(t, x, u, u_x, u_{xx}, u_{xxx})\right)_x , \tag{4.4}$$

and on a physical basis one is often interested in preserving the conservation property

$$\frac{d}{dt}\int_I u = g(b) - g(a). \tag{4.5}$$

The above Gauss collocation method does not do so in general, but it does in special cases such as the following.

Lemma 4.1. *Gauss point collocation preserves the conservation property of (4.5) for the equation (4.4) for linear functions*

$$g(u, u_x, u_{xx}, u_{xxx}) = \alpha_1 u + \alpha_2 u_x + \alpha_3 u_{xx} + \alpha_4 u_{xxx}, \tag{4.6}$$

where α_i, $i = 1, ..., 4$ *are constants.*

Proof. Denoting $U_i^{(0)} = U_i$, $U_i^{(1)} = U_{x,i}$, $U_i^{(2)} = U_{xx,i}$, $U_i^{(3)} = U_{xxx,i}$, we have that

$$\begin{aligned}
\frac{d}{dt}\int_I U dx &= \int_I \alpha_1 U_x + \alpha_2 U_{xx} + \alpha_3 U_{xxx} + \alpha_4 U_{xxxx} dx \\
&= \sum_i \sum_j \alpha_j \left(\sum_{k=0}^{3} (H_i)^{k-j} \left(U_i^{(k)} \int_0^1 \frac{d^j}{ds^j} L_{0,k}(s) ds + U_{i+1}^{(k)} \int_0^1 \frac{d^j}{ds^j} L_{1,k}(s) ds \right) \right) \\
&= \sum_i \sum_j \alpha_j (U_{i+1}^{(j-1)} - U_i^{(j-1)}) \qquad \text{(from (3.2))} \\
&= \sum_j \alpha_j (U_{N+1}^{(j-1)} - U_1^{(j-1)}) \\
&= G(X_{N+1}) - G(X_1).
\end{aligned}$$

□

5 Conservative method

While Gauss point collocation performs well for many problems, here we construct another method which has the advantage of preserving general conservation properties. For this, we consider a more general PDE of the form

$$f(t, x, u_t, u, u_x, u_{xx}, u_{xxx}) = \left(g(t, x, u, u_x, u_{xx}, u_{xxx})\right)_x \tag{5.1}$$

and derive an approximation method for it which satisfies the generalized conservation property

$$\int_I f dx = g|_{x=b} - g|_{x=a}. \tag{5.2}$$

An additional advantage of treating the PDE (5.1) directly is that when solutions develop singularities, each successive spatial derivative becomes larger, and we can avoid the explicit computation of u_{xxxx} by using a cell averaged form. As a result, one can often solve the problem more efficiently by reducing the stiffness of the ODE system compared to the standard Gauss scheme which requires fourth derivatives. Note, however, that a PDE (4.1) can be written in the form (5.1) in a number of ways.

To generate an approximation to u, we again approximate u by its Hermite interpolant U as defined in (3.1). To eliminate any dependence on U_{xxxx} in our system of ODEs and to enforce the conservation property (5.2) we use a cell-averaged approach for solving (5.1). We begin by approximating f by its Lagrange interpolant

$$F_i = \sum_{j=0}^{3} f_{ij} \tilde{L}_j, \tag{5.3}$$

where the canonical points are the four Gauss points on the unit interval

$$\rho_1 = \frac{1}{2} - \frac{\sqrt{525+70\sqrt{30}}}{70},\ \rho_2 = \frac{1}{2} - \frac{\sqrt{525-70\sqrt{30}}}{70},\ \rho_3 = 1-\rho_1,\ \rho_4 = 1-\rho_2 \quad (5.4)$$

and $f_{ij} = f(x_i + H_i\rho_j) = f(x_{ij})$. Equation (5.1) is integrated approximately using the five Lobatto points, which on the unit interval are

$$\tilde{\rho}_1 = 0,\ \ \tilde{\rho}_2 = \frac{1}{2} - \frac{\sqrt{21}}{14},\ \ \tilde{\rho}_3 = \frac{1}{2},\ \ \tilde{\rho}_4 = \frac{1}{2} + \frac{\sqrt{21}}{14},\ \ \tilde{\rho}_5 = 1. \quad (5.5)$$

Using the approximation (5.3) to f gives the system

$$A\boldsymbol{F}_i = B\boldsymbol{G}_i/H_i \quad (5.6)$$

in each interval I_i, where the matrices A, B are defined by

$$A_{jk} = \int_{\tilde{\rho}_j}^{\tilde{\rho}_j+1} \tilde{L}_k dt \quad j,\, k = 1,2,3,4,$$

$$B_{jk} = \begin{cases} 1 \text{ if } j = k, \\ -1 \text{ if } k = j+1, \\ 0 \text{ else,} \end{cases} \qquad \text{for } j = 1,...,4 \text{ and } k = 1,...,5,$$

and the vectors $\boldsymbol{F}_i$, $\boldsymbol{G}_i$ are

$$(\boldsymbol{F}_i)_j = F_i(x_i + H_i\rho_j) = F_i(x_{ij}), \quad j = 1,2,3,4,$$

$$(\boldsymbol{G}_i)_j = G_i(x_i + H_i\tilde{\rho}_j) = G_i(\tilde{x}_{ij}), \quad j = 1,2,3,4,5\ .$$

By direct construction we have enforcement of cell-averaging, i.e.,

$$\int_{I_i} F_i = G(X_{i+1}) - G(X_i),$$

and we have the following lemma.

Lemma 5.1. *A scheme satisfying* (5.6) *enforces discrete conservation.*

Proof.

$$\int_I F dx = \sum_i \int_{I_i} F_i dx = \sum_i G(X_{i+1}) - G(X_i) = G(X_{N+1}) - G(X_1).$$

□

The Hermite interpolant U is defined in (3.1) and is specified by choosing the coefficients $U_i(t), U_{x,i}(t), U_{xx,i}(t), U_{xxx,i}(t)$ such that in each interval

$$H_i\boldsymbol{F}_i = W\boldsymbol{G}_i,$$

where the differentiation matrix $W = A^{-1}B$. This gives four equations in each cell, like the standard Gauss collocation method, but they do not involve U_{xxxx}. Using the fact that the two extreme Lobatto points $\tilde{\rho}_1 = 0$ and $\tilde{\rho}_5 = 1$ coincide at successive intervals, an efficient implementation requires $8N+1$ function evaluations, twice that of the Gauss point formulation with $2N+1$ ODE function evaluations. However, in many cases the structure of g is much simpler than g_x, leading to fewer total floating point operations.

The spline collocation error analysis in [2] can when suitably modified be extended to give error bounds for the conservative scheme. We can find the local truncation error for our equations by considering the problem

$$g_x = f \quad (5.7)$$

as an equation for g. This approach extends the error estimate derived in [33] which is valid only at the Gauss points to a spatially global one. Henceforth, we shall refer to this discretization of (5.7) as a "conservative collocation" scheme, although technically speaking it is not a collocation scheme, as we discuss below.

Theorem 5.2. *The conservative collocation scheme* (5.6) *is globally fourth-order accurate with superconvergence of order five at the Gauss points.*

Proof. The results of Theorem 5.147 from [2] can be used to estimate the leading order term in $G_x - g_x$. There are $k = 5$ Lobatto points, with precision $p = 2k - 2 = 8$ to solve the first-order ($m = 1$) problem (5.7)

$$G_x - g_x \simeq H_i^5 \frac{g^{(6)}}{6!} \prod_{j=1}^{5} (s - \tilde{\rho}_j).$$

The error in the Lagrange approximation to f is given by

$$F - f \simeq \frac{f^{(4)}}{4!} \prod_{j=1}^{4} (s - \rho_j)$$

and thus the error

$$\begin{aligned} e = |(G_x - F)| &= |(G_x - g_x) - (F - f)| \\ &\simeq \left| H_i^5 \frac{g^{(6)}}{6!} \prod_{j=1}^{5} (s - \tilde{\rho}_j) - H_i^4 \frac{g^{(5)}}{4!} \prod_{j=1}^{4} (s - \rho_j) \right| \end{aligned} \tag{5.8}$$

for all points $x_i \le x \le x_i$, and $s = (x - x_i)/H_i$. □

Here we have used the fact that the exact solution satisfies $g_x = f$. At first glance this estimate may seem slightly strange — it is an estimate of the residual, something normally set to be identically zero in a collocation method. However, recall that there are *two* sources of error in the conservative method, one is an interpolation error for f, and the other is from the approximate solution to $g_x = f$. These errors vanish at different points in the domain, so there is no point where the residual is necessarily zero. This estimate may also be recovered by a standard Taylor series expansion, but that is far more cumbersome. This same construction can be used to extend the error estimate in [33] from the Gauss points to the entire solution domain.

For the standard Gauss collocation scheme we have explicit error estimates for $e_G = |u - U|$, whereas for the conservative scheme we only have estimates for $e_c = |g - G|$ which, in the general case, may not easily give information about $|u - U|$. We expect a similar estimate for $e = |u - U|$ to hold for both but leave a proof for future work and content ourselves now with a numerical test of the methods.

6 Implementation

MOVCOL4 is a Fortran code with the capability of using either the Gauss collocation or the conservative scheme described in the two previous sections. The user is provided the option of selecting either one and of solving the MMPDE in the form of either MMPDE4 or MMPDE6. The resulting equations are integrated with the software DASSL [38], a backward differentiation code designed to solve stiff DAEs. Other inputs provided by the user are the absolute error tolerance *atol* and relative error tolerance *rtol* which are needed for DASSL, the number of mesh points *npts* and initial mesh $X_i : i = 1, ..., npts$, and the definition of the monitor function to be used in the MMPDE. The onus is on the user to ensure that the PDE is well posed (e.g., not $u_t = u_{xxxx} + r(u)$, which is unstable like the backward heat equation). The code is designed to deal with very general systems of PDEs of the form

$$\boldsymbol{f}(t, x, \boldsymbol{u}, \boldsymbol{u}_t, \boldsymbol{u}_x, \boldsymbol{u}_{xx}, \boldsymbol{u}_{xxx}, \boldsymbol{u}_{xxxx}) = (\boldsymbol{g}(t, x, \boldsymbol{u}, \boldsymbol{u}_x, \boldsymbol{u}_{xx}, \boldsymbol{u}_{xxx}))_x, \tag{6.1}$$

although the form chosen can significantly affect the code's efficiency. One of the most important input parameters is the initial mesh. For many problems, it must be close to equidistributing in order for the integrator DASSL to start. To help alleviate this common difficulty, MOVCOL4, like MOVCOL, has an option to compute an equidistributing mesh corresponding to the initial data $u_0(x)$ before solving the PDE.

The user also has the option to choose different tolerances for the solution components. This is particularly effective for the higher derivatives which, because they appear in the solution definition multiplied by powers of the local grid spacing, have a lesser effect on the error in reconstructing the entire solution. Typically, one can take the most stringent tolerances for the solution only and demand less accuracy on the derivatives with no noticeable loss in the accuracy of the final solution.

A common approach for solving high order PDEs is to first convert them into lower order systems and use a code designed for these lower order problems. For example, the fourth order PDE

$$v_t = f(t, x, v, v_x, v_{xx}, v_{xxx}, v_{xxxx})$$

can be written as a second order system in the variables $\boldsymbol{u} = (u_1, u_2)^T = (v, v_{xx})^T$. The equivalent system

$$(u_1)_t = f(t, x, u_1, (u_1)_x, u_2, (u_2)_x, (u_2)_{xx}),\ (u_1)_{xx} = u_2$$

is then solved. Discretization on a fixed grid gives a DAE system of index 1, where time differentiation of the algebraic equation once gives an ODE system with the second equation replaced by $(u_2)_t = (u_1)_{xxt}$. One could then employ a method of lines code designed for second order problems which uses a time integration code such as DASSL, which is designed to solve index 1 DAEs under reasonable restrictions on the rate of change of stepsize [27]. However, when this approach is used for a moving grid we obtain

$$(u_2)_t = (u_1)_{xxt} + [(u_1)_{xxx} - (u_2)_x]\dot{x}(t).$$

Unfortunately, $(u_1)_{xxx} = (u_2)_x$ does not follow from $(u_1)_{xx} = u_2$ since u_2 and u_1 are approximated with different spline functions, and experience has shown that the error can easily grow if this term is dropped. Consequently, both the third derivative term $(u_1)_{xxx}$ and the mesh speed $\dot{x}(t)$ must be dealt with, and a code for second order problems like MOVCOL cannot be used directly. One can convert the PDE into the second order system

$$\begin{aligned}
(u_1)_t &= f(t, x, u_1, (u_1)_x, u_2, (u_2)_x, (u_3)_x, (u_3)_{xx}) \\
(u_2)_t &= (u_1)_{xt} + [(u_1)_{xx} - (u_2)_x]\dot{x}(t) \\
(u_3)_t &= (u_2)_{xt} + [(u_2)_{xx} - (u_3)_x]\dot{x}(t),
\end{aligned}$$

although this form is clearly more awkward. In addition, this can be quite inefficient due to the need to handle the mesh speed term and the large number of additional unknowns as compared to the method applied directly to the high order equation. In our experience, applying the moving mesh code MOVCOL for lower order systems has not generally warranted the associated cost nor achieved comparable accuracy to that with MOVCOL4 for fourth order problems. Nevertheless, it remains an interesting question, both in the fixed grid and moving grid contexts, as to whether or not there are reformulations of high order problems as systems of lower order ones which are sufficiently stable (and well-posed) to warrant their use with codes designed for such low order problems.

7 Numerical examples

In this section we demonstrate the efficiency, accuracy and robustness of MOVCOL4 by solving a variety of problems. Example 1 has been chosen to illustrate the rate of convergence for a simple test case. Examples 2 and 3 have been chosen from applications

to show the breadth of applicability of our approach. The first is the semilinear Cahn-Hilliard equation and the second concerns some quasilinear thin-film equations. For both examples two different solution regimes are investigated.

One of the most difficult choices for the novice when using moving mesh methods is that of the monitor function. There are several competing strategies, and the different problems in this section motivate some of those different choices. The first problem is trivial and not particularly sensitive to the precise choice of monitor function.

In the examples we use monitor functions determined either by properties of the physical equations or by information about the particular solution features. Two cases exhibit finite-time blow-up, and for such problems preserving scale invariance in the moving mesh method works particularly well [15, 18]. One case has very delicate finite-time extinction, and the monitor function is constructed based loosely on asymptotic properties of the solution. For this case MMPDE6 is most appropriate as it allows adaptivity on the same timescale as the monitor function, which is key for finite-time behaviour. The remaining problem exhibits metastable fronts. Because solution features emerge on an $O(1)$ timescale we use MMPDE4 and a monitor function prepared to place more points in any emergent layers. In this scenario, the time stepping is determined solely by the physical PDE and not by changes in magnitude of the monitor function.

7.1 A simple test problem

We begin with a simple test problem

$$\text{Ex. 1} \quad \begin{cases} u_t = -\dfrac{\sin t}{\cos t + 3} u_{xxxx} & x \in (0,\pi), 0 < t < \pi \\ u(x,0) = 4c\cos(x) & \\ u_x(0,t) = u_{xxx}(0,t) = 0 & \\ u_x(\pi,t) = u_{xxx}(\pi,t) = 0, & \end{cases} \tag{7.1}$$

which has the exact solution: $u(x,t) = c(\cos(t)+3)\cos(x)$. We investigate the convergence properties of MOVCOL4 for this example.

The high order of the spatial discretization necessitates that the problem be designed very carefully in order for one to check the actual convergence rate. An advantage of the problem in this form is that there is only the fourth derivative term, which simplifies the form of the discretization error over mathematically equivalent more complicated forms, although some care must be taken since this problem is ill-posed for $\sin t < 0$. Also, it is very hard to discern the rate of convergence before the time integration error and machine error become involved and play a significant role, so the absolute error tolerance and relative error tolerance for the time integration must be chosen to be extremely small — we choose $atol = 10^{-15}$, $rtol = 10^{-13}$. We solve the problem with 2 subintervals (3 mesh points), 4 subintervals (5 mesh points), and finally, 8 subintervals (9 mesh points). In order to see the global rate of convergence, we first compute the solution on $[0,\pi]$ using 3, 5 or 9 mesh points and then construct the solution on the whole interval using Hermite spline interpolation.

It is easiest to start the integration with a fixed grid using MMPDE6 and $\tau = 1.d{-}3$, where the smoothness of the solution and the monitor function $M(u) = 1 + |u|$ prevent the mesh from becoming too skew and the temporal discretization error from strongly affecting the total error. In our tests we choose $c = 0.3$.

In the log-log plot of the L^∞ norm of the error versus the mesh size given in Figure 1, the slope of the straight line fit, the rate of convergence, is approximately 8. In Table 1, we show computed rates of convergence

$$\text{ROC}N_1N_2 = \frac{\log\left(\frac{\|E_{N_1}\|_\infty}{\|E_{N_2}\|_\infty}\right)}{\log\left(\frac{\pi/N_1}{\pi/N_2}\right)}.$$

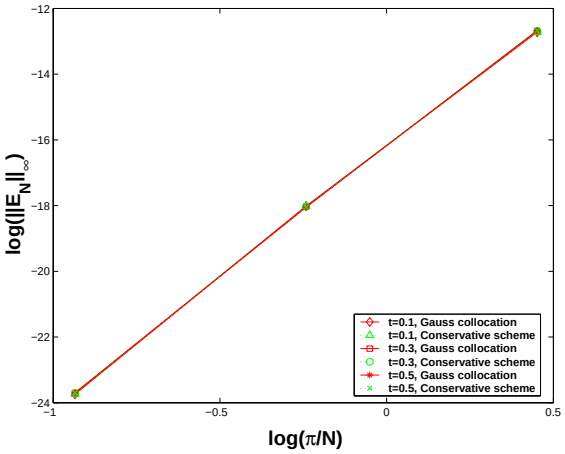

Figure 1: Rate of convergence, Gauss collocation method and conservative collocation method. The errors for both methods are indistinguishable.

Table 1: Rate of Convergence

Time	M	$\|E_2\|_\infty$	$\|E_4\|_\infty$	$\|E_8\|_\infty$	ROC42	ROC84	ROC82
$t = 0.1$	G	2.982353E-06	1.501874E-08	4.862577E-11	7.6335	8.2708	7.9522
	C	2.982353E-06	1.501874E-08	4.870682E-11	7.6335	8.2684	7.9510
$t = 0.3$	G	3.099078E-06	1.475483E-08	5.055389E-11	7.7145	8.1891	7.9518
	C	3.099078E-06	1.475483E-08	5.062472E-11	7.7145	8.1871	7.9508
$t = 0.5$	G	3.150803E-06	1.446814E-08	5.111500E-11	7.7667	8.1445	7.9558
	C	3.150803E-06	1.446855E-08	5.113077E-11	7.7667	8.1445	7.9556

For example, ROC42 $= \frac{\log\left(\frac{\|E_2\|_\infty}{\|E_4\|_\infty}\right)}{\log\left(\frac{\pi/2}{\pi/4}\right)}$ where $\|E_2\|_\infty = \max_{i=1}^{101} |u_{\text{comp}} - u_{\text{exact}}|$ is measured by sampling the reconstructed and the exact solutions at 101 equally spaced points on $[0, \pi]$.

7.2 The Cahn-Hilliard equation

The Cahn-Hilliard equation is a classical semilinear fourth-order equation modelling phase separation. Many diffusive processes with a natural length scale, such as phase separation in binary alloys, growth and dispersal in population, or spreading of an oil film over a solid surface, can be described by this equation. The special form for the Cahn-Hilliard equation which we investigate here is

$$u_t = -(\gamma_1 u_{xx} - \gamma_2 u^3 + \gamma_3 u^2 + \gamma_4 u)_{xx}. \tag{7.2}$$

We consider two cases with strikingly different behaviour, both arising in various application areas.

7.2.1 Finite-time blow-up in the unstable regime

For the case of $\gamma_1 = 1, \gamma_2 = -1$ and $\gamma_3 = \gamma_4 = 0$ [25], the equation has a combination of a fourth-order diffusion operator and the classical second-order porous medium operator but posed *backwards* in time. This competition of effects can lead to finite time blow-up, where there exists $T < \infty$ such that

$$\lim_{t \to T} \|u(\cdot, t)\|_\infty = \infty \quad \text{and} \quad \|u(\cdot, t)\|_\infty < \infty \ \ \forall \ \ 0 \le t < T.$$

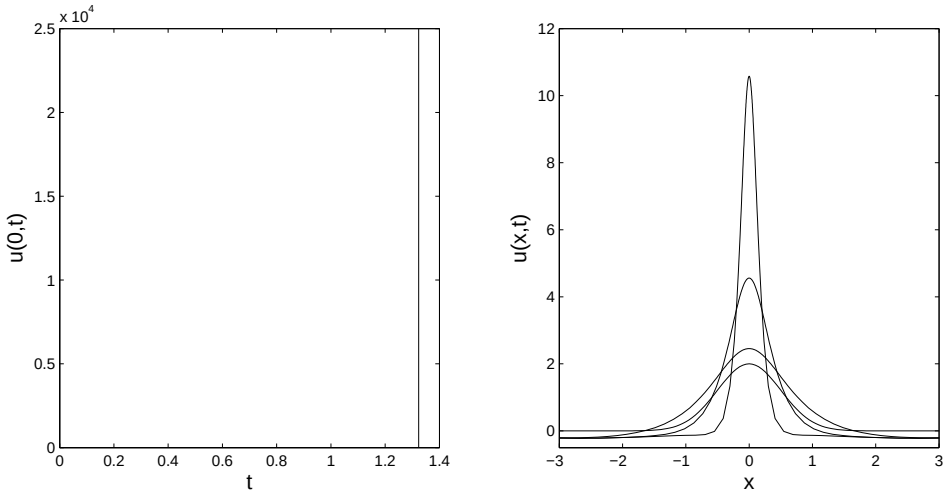

Figure 2: Blow-up in the unstable Cahn-Hilliard equation (7.3). (Left) The solution magnitude. (Right) Sample solutions.

The complete problem is

$$\text{Ex. 2.1} \quad \begin{cases} u_t = -(u_{xx} + u^3)_{xx} & x \in (0,6), t > 0 \\ u(x,0) = 2e^{-\frac{x^2}{2}} & \\ u_x(0,t) = u_{xxx}(0,t) = 0 & \\ u_x(6,t) = u_{xxx}(6,t) = 0. & \end{cases} \tag{7.3}$$

The problem is conservative, and conservation with blow-up implies that the blow-up can only occur at an isolated point (or set of isolated points). In this case the blow-up is self-similar in that there is a balance between the magnitude of and natural length-scale of the solution. The use of scaling invariant adaptivity is natural for problems such as this one, since as we see below the intrinsic length-scale of the solution is then automatically determined.

We take $npts = 41$, $rtol = atol = 10^{-5}$ and $\tau = 1e-4$. To resolve the blow-up solution we choose the monitor function

$$M(x,t) = u(x,t)^4 + \frac{1}{12}\int_0^6 u(x,t)^4\,dx,$$

because it is scale invariant in the blow-up region, i.e., the MMPDE is invariant under the same scaling group as the physical PDE. The factor 1/12 is chosen to place approximately two thirds of the grid points inside the blow-up region and one third outside — for further discussion of scaling in constructing monitor functions, see [15] and [16]. The role of the second term for the monitor function is to ensure that there is an even distribution of mesh points inside and outside of the blow-up region as blow-up is approached. This idea, first introduced by Mackenzie [5], is analyzed theoretically for PDEs with finite-time singularities in [18].

Typical blow-up solutions are presented in Figure 2. (A comprehensive treatment of this problem is given in [25].) We compare the effectiveness of the two spatial discretizations in Table 2. Note that the conservative discretization is more efficient and can integrate marginally further into the blow-up regime. A comparison of the accuracy of conservation for the two approaches is presented for a more difficult problem in Section 7.3.2.

	$u(0,t)$	Steps	Fcn.	Jac.
Conservative Collocation	$2.51e4$	1748	2934	217
	1000	1103	1998	167
	500	999	1729	161
	100	821	1279	131
	50	677	1199	121
Gauss Point Collocation	$2.31e4$	2731	4491	751
	1000	1499	3149	489
	500	1401	2691	390
	100	1121	1713	243
	50	720	1299	117

Table 2: Comparison of spatial discretizations for (7.2). Here Steps is the number of time steps required, Fcn. the number of function evaluations, and Jac. the number of Jacobian evaluations required by DASSL.

7.2.2 Metastable fronts in the stable regime

While the Cahn-Hilliard equation can also give exponential decay (like the heat equation) or blow-up, we now concentrate on the metastable case [43]. The typical underlying physical context involves a melted binary alloy with a given concentration of components that is quenched to a temperature at which exactly two different concentration phases can coexist. For $\gamma_1 \to 0$, the qualitative features of the dynamics associated with (7.2) are as follows: Through a very intricate transient process, a pattern of internal layers forms from the initial data over an $O(1)$ time interval. Once this pattern is formed, the subsequent motion of the internal layers is exponentially slow and hence becomes metastable. For a study of two layers pattern in [39], the internal layers collapse at a time of order 10^{11} and the interesting dynamics then occur very fast.

The form of the Cahn-Hilliard equation we consider is

$$\text{Ex. 2.2} \quad \begin{cases} u_t = -(0.001u_{xx} + u - u^3)_{xx}, & x \in (-1,1), t > 0 \\ u(x,0) = 0.1\cos(2\pi x) + 0.02\cos(10\pi x) \\ u_x(-1,t) = u_{xxx}(-1,t) = 0 \\ u_x(1,t) = u_{xxx}(1,t) = 0. \end{cases} \tag{7.4}$$

Since the nonlinear term in (7.2) together with the fourth derivative term make the Cahn-Hilliard equation very stiff, most finite difference schemes need to use time steps of many orders of magnitude smaller than Δx. As a consequence, it is numerically expensive to reach the time scales where the interesting dynamics occur. In Figure 3 we show our numerical results for a finite difference scheme provided in [36]. With this approach, the resulting ODE system is linear and easily solved directly in MATLAB. We have observed that if the ODE from the finite difference scheme is not carefully resolved by using a large stencil and very small time step size, a two-layer pattern will be formed much faster than actually occurs. As we increase our stencil and decrease the time step size, the numerical solution obtained by the finite difference scheme converges to the numerical solution obtained by using the moving collocation scheme with only 21 points (but a system of 105 unknowns). If the number of points is increased from 21 to 101, the numerical solution for the moving collocation scheme remains almost the same, suggesting that a small number of mesh points can be used to investigate the metastable behavior of the solution efficiently. Generally speaking, a very accurate discretization is required for the resolution of metastable behaviour, else the dynamics will be driven by numerical error rather than the very slow analytical instability.

Figure 4 shows the solutions of (7.4) starting with a uniform mesh of 51 points.

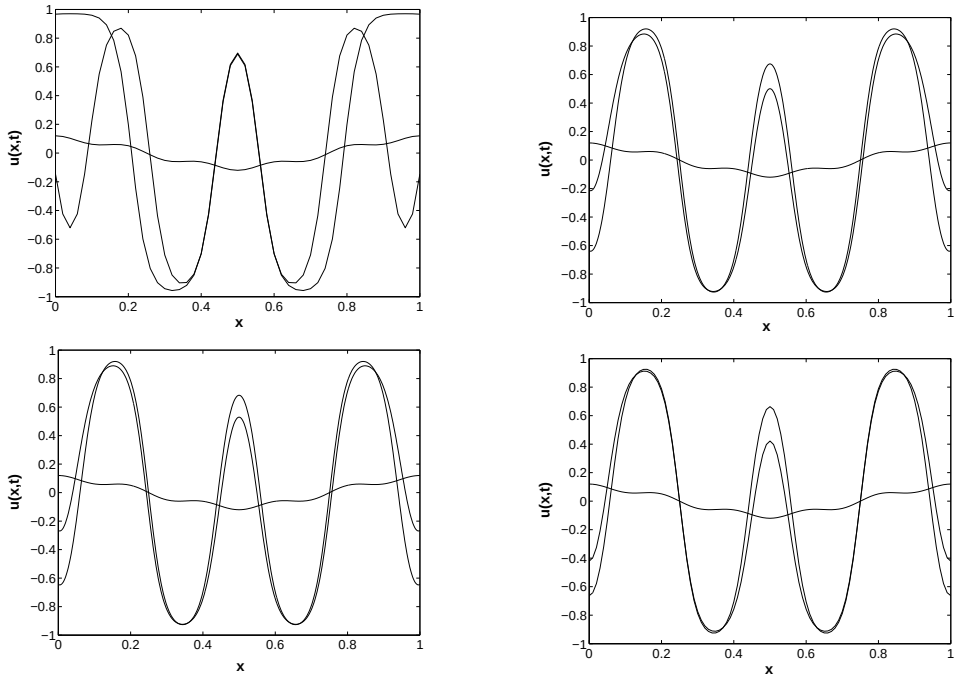

Figure 3: Solutions at t=0, 0.05, 0.1. Finite difference scheme vs moving collocation scheme. (Upper Left) Finite difference scheme, $\Delta x = 0.02, \Delta t = 1.d-6$; (Upper Right) Finite difference scheme, $\Delta x = 0.002, \Delta t = 1.d-4$; (Lower Left) Finite difference scheme, $\Delta x = 0.002, \Delta t = 1.d-8$; (Lower Right) Moving collocation scheme, $\Delta x = 0.05$, variable time step size by DDASSL.

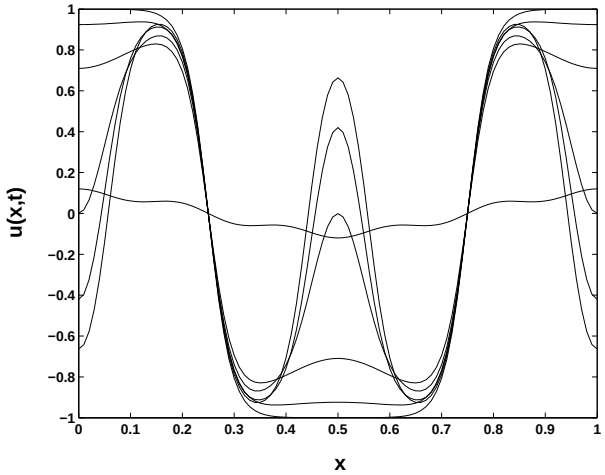

Figure 4: Solutions to (7.4) at different times: $t = 0, 0.05, 0.1, 0.3, 0.5, 0.7, 1, 1 \times 10^{10}$

The solution is computed until $t = 1 \times 10^{10}$ with the monitor function chosen as $M = \sqrt{1 + u_x^2}$ to place more mesh points in the internal layer and $atol = rtol = 1 \times 10^{-6}$. The solution at $t = 1 \times 10^{10}$, shown in Figure 4, is indistinguishable from the one at $t = 1$.

Our computational tests have shown that the time at which the two metastable interfaces collapse is highly affected by the error tolerances, making it difficult to capture the precise time of collapse, particularly since it is usually of order 10^{11}. Furthermore, the role of adaptivity in computing metastable solutions is at best problematic, since a suitable monitor would need to incorporate information about the solution's sensitivity to the boundary conditions. Further investigation of this issue is needed before a code such as MOVCOL4 could be recommended over other adaptive (or non-adaptive) ones for a general study of metastability for such problems.

7.3 Equations of thin-film type

A fourth-order equation of wide interest is the thin-film equation

$$u_t = -(f(u)u_{xxx} + g(u)u_x)_x. \tag{7.5}$$

In most applications u represents the thickness of a thin fluid on a substrate. This equation has garnered much analytical and computational interest over the past fifteen years (e.g., see [3, 6, 8, 10, 9, 12]), although many open problems remain. Typically, the nonlinearities are taken to be

$$f(u) = u^n, \quad \text{and} \quad g(u) = u^m$$

for positive integers n, m. We consider the cases

$$\begin{aligned} f(u) &= u^{1/2} \quad \text{with} \quad g(u) = 0 \quad \text{and} \\ f(u) &= u \quad \text{with} \quad g(u) = 4u^3. \end{aligned}$$

For suitable initial, boundary and free boundary conditions, (7.5) has compactly supported solutions which may spread or contract over time. In both of our examples we use (for both analysis and computation) a standard regularization of f,

$$f_\varepsilon(u_\varepsilon) = \frac{u_\varepsilon^4}{\varepsilon + u_\varepsilon^{4-n}}. \tag{7.6}$$

Because of this regularization one expects the solution to be bounded away from zero with

$$\min u_\varepsilon \sim O(\varepsilon^{1/(4-n)})$$

where $f_\varepsilon(u) \sim O(u^n)$ as $u \to 0$ for $n < 4$. For $n \geq 4$ no such regularization is required. Moreover, in this regime the dynamics are dominated by the regularization, since if $u_\varepsilon \ll u_\varepsilon^{4-n}$ then

$$f_\varepsilon \sim \frac{u_\varepsilon^4}{\varepsilon}.$$

With the appropriate boundary conditions, the solution to (7.5) satisfies the condition (5.2), or

$$\frac{d}{dt} \int_I u(x,t)\, dx = 0.$$

7.3.1 Moving contact lines in thin liquid films

For the form of the thin-film equation first studied,

$$u_t = -(u^n u_{xxx})_x,$$

the solutions do not blow-up, but initially uniformly positive solutions can become extinct at some finite-time. We consider this particular case because it is one where

there is conjectured to be approximate self-similar behaviour and because its numerical solution has been computed elsewhere, making it a reliable test problem.

For $n = 1/2$, the problem

$$\text{Ex. 3.1} \quad \begin{cases} u_t + (u^{1/2}u_{xxx})_x = 0, & x \in (-1,1), t > 0 \\ u(x,0) = 0.8 - \cos(\pi x) + 0.25\cos(2\pi x) \\ u_x(-1,t) = u_x(1,t) = 0 \\ u_{xxx}(-1,t) = u_{xxx}(1,t) = 0\ , \end{cases} \tag{7.7}$$

is considered in [10, 9, 44]. It is shown computationally in [9] that the solution exhibits finite time extinction. Specifically, the solution becomes zero at an isolated point for some finite time, and there is simultaneous blowup of higher derivatives there. Near the singular point the solution has a leading order asymptotic form [10]

$$u(x,t) \approx c(t_c - t) + \frac{p(x - x_c)^2}{2} \tag{7.8}$$

where t_c is the time of extinction, x_c is the "pinch-point", and the constant p is the curvature of the interface at this time. The solution and its first two spatial derivatives are smooth here, while blow-up in higher derivatives is locally described by

$$u_{xxx} \approx \frac{cx}{\sqrt{c(t_c - t) + p(x - x_c)^2/2}}. \tag{7.9}$$

It is proven in [11, 9] that after the initial singularity time, there exists a nonnegative weak solution for (7.7) in the sense of distributions, and there exists a second critical time after which the solution is again uniformly positive and strong, decaying to its mean in the infinite time limit. Introducing the regularization (7.6), the regularized problem has global, positive and smooth solutions [9]. The weak solution for (7.7) can then be obtained by finding the solution of the regularized problem and taking the regularization parameter ε to zero [12, 6, 8]. Many thin film problems have been solved with finite element discretizations [3], but in [9] a careful finite difference static regridding computation starting with a fine initial grid of $2^{10} = 1024$ uniform intervals on [0,1] is used. Since the initial data is symmetric about $x = 0$ and the equation preserves the symmetry, the solution is only computed on $[0, 1]$, using reflection symmetry at the boundary. In order to resolve the smallest length scale and keep the numerical solution positive, regridding is done by adding 64 new mesh points whenever $u_{\min}^{1/2} \le 64/1024$. At the end of the simulation, 55 levels of regridding have been done, with 3520 new points produced and the smallest grid spacing being 2^{-65}. We use M to automatically monitor quantities of interest and relocate the mesh points, making it a good test problem for the adaptive features of our high-order moving mesh code MOVCOL4.

Our computation consists of two parts: computing the solution of the non-regularized problem very close to the singularity time to examine the formation of the third derivative jump discontinuity, and computing the solution until it decays to its mean as dictated by the theory [9]. In either case, we start with an initially uniform grid with $2^7 = 128$ intervals on $[0, 1]$, hence a system of 640 equations for all time. For the monitor function, we basically choose $M(x,t) = 1/u$. It has been chosen to resolve the singularity as $\min_x u(x,t)$ approaches zero. The exact structure is based on the asymptotic form (7.8) which shows that the solution touches down proportionally to $(t - t_c)$. Hence, our monitor function scales in time the same way as the singularity. In other cases different powers of u may be required — generically,

$$M = \frac{1}{u^{1/p}} \qquad \text{if} \qquad u \simeq (t - t_c)^p.$$

By comparing (2.3) and (2.4), we can see that if the monitor function varies dramatically on the whole interval of interest, then the mesh evolves faster when MMPDE4 is used but MMPDE6 is better at forcing the mesh to concentrate in the area where the monitor function has larger values. For the non-regularized problem, in order to resolve the spatial structure and solve the problem very close to the singularity time we

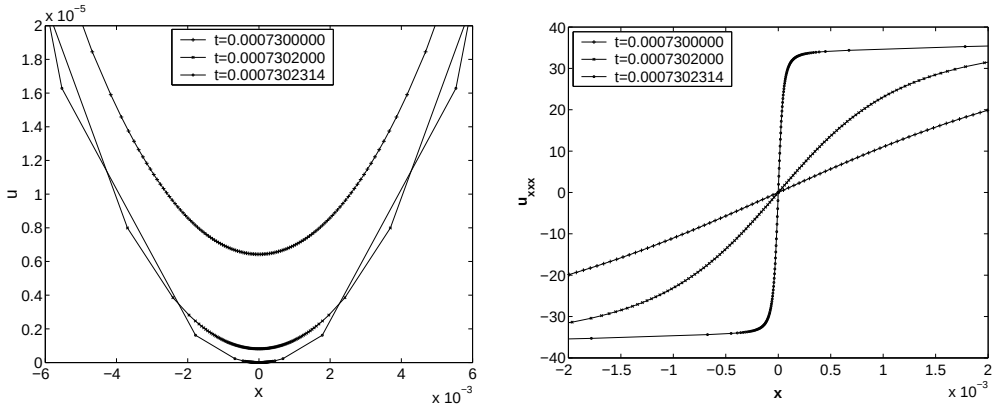

Figure 5: Solutions at times very close to the singularity time: (Left) Solutions near the pinch-point. (Right) Onset of initial singularity, formation of a jump discontinuity in the third derivative.

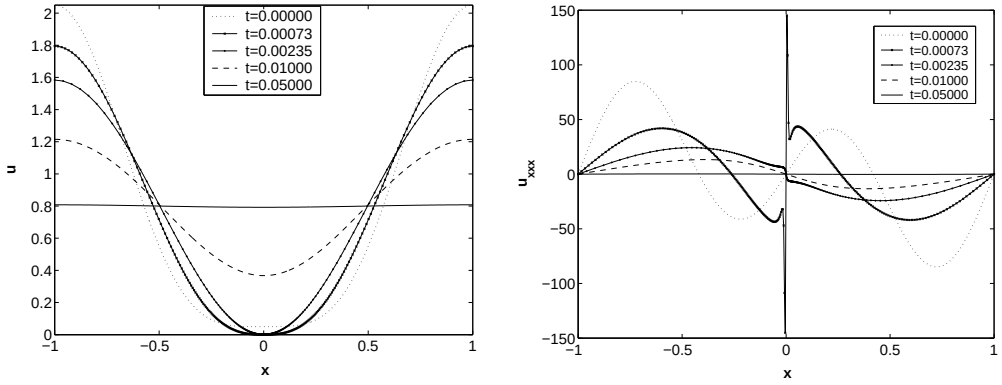

Figure 6: Solutions of the regularized system at typical times: t=0, the initial profile, t=0.00073, roughly the time when the pinch-off singularity occurs; t=0.002345, roughly the time when the weak solution becomes strong and positive again; t=0.01, strong solution decays to its mean; t=0.05, solution almost decays to its mean $\bar{u} = 0.8$. (Left) Typical solutions (Right) The third derivatives.

use MMPDE6 with $\tau = 1 \times 10^{-3}$. To make it easier for the mesh to evolve, we modify the monitor function to be $M(x,t) = c/(u+d)$. Here, $c < 1$ rescales the basic monitor function and helps the mesh to evolve faster while d determines the 'radius' of the area centred at the one-point singularity $u = 0$ where the mesh points concentrate. In the results shown in Figure 5, we use $c = 0.1$ and $d = 1 \times 10^{-8}$. The error tolerances for the time integrator are $atol = rtol = 1 \times 10^{-6}$.

Figure 5 displays the corresponding solutions and third derivatives at specified times close to the singularity time. These results closely agree with those found in [9, 10]. Figure 5 also shows that, by carefully choosing a suitable monitor function and an MMPDE, the mesh can in turn evolve along with the singularity and hence capture or resolve the spatial structure.

For the regularized system, in order to sufficiently resolve the spatial structure at all times, we use MMPDE4 with $\tau = 1 \times 10^{-3}$ and the basic monitor function $M(x,t) = 1/u$. The error tolerances for the time integrator are $atol = rtol = 1 \times 10^{-8}$ for the solution and $atol = rtol = 1 \times 10^{-4}$ for all its derivatives.

Figure 6 shows the corresponding solutions at specified times which are typical

for this problem. These results are also in close agreement with those found in [9, 10]. With our methods, only 128 points on $[0, 1]$ are used. Furthermore, we are able to easily evaluate other potential quantities of interest. In particular, higher derivatives may be easily and much more reliably evaluated than with a moving finite difference approach. Also, because we are using the inverse of the solution as the monitor function, it is easy to track the movement of the minimum of the solution. This makes it very simple to test asymptotic hypotheses.

7.3.2 Finite-time blow-up

Given the specific case $f(u) = u$ and $g(u) = 4u^3$, equation (7.5) takes the form

$$u_t = -(uu_{xxx} + 4u^3 u_x)_x.$$

As for the unstable Cahn-Hilliard equation, the balance between a stable fourth-order operator and an unstable second-order one can lead to finite-time blow-up. This is particularly delicate here, as it is a critical case for blow-up [12], [42], [26]. Given $f(u) = u$ and $g(u) = u^p$ we have no blow-up for $p < 3$, always blow-up for $p > 3$, and sometimes blow-up for $p = 3$. Moreover, the long time dynamics of solutions to this equation are very strongly determined by the mass of the initial data; for sufficient mass there is always blow-up, while for insufficient mass there is never blow-up. Not only that, but the set of allowable final time blow-up profiles is also restricted by the initial mass [42, 26].

Because of the combination of conservation of mass and finite-time blow-up we immediately see that the singularity must form at a point. This problem is another excellent test for our adaptive method, and in particular the adaptive *conservative* method.

We use both the conservative and Gauss collation methods when solving the problem

$$\text{Ex. 3.2} \quad \begin{cases} u_t = -(uu_{xxx} + 4u^3 u_x)_x, \quad x \in (0, 6), t > 0 \\ u_x(0, t) = u_{xxx}(0, t) = u_x(6, t) = u_{xxx}(6, t) = 0, \\ u(x, 0) = \frac{m}{\sqrt{2\pi}} e^{-.5x^2}. \end{cases} \tag{7.10}$$

It is solved with $N = 41$ grid points, $atol = rtol = 1e - 3$ and the monitor function

$$\tilde{M}(x, t) = u(x, t)^5 + |x^2 u_{xx}(x, t)|^5, \qquad M(x, t) = \tilde{M}(x, t) + \frac{1}{12} \int_0^6 \tilde{M}(s, t) dx.$$

Again, this particular monitor function has been chosen to be scale invariant in the blow-up region. One of the difficulties of this problem is that the solution approaches a blow-up profile which is compactly supported in the similarity variables, i.e.,

$$\lim_{t \to T} (T - t)^{1/5} u(x, t) = f(y), \quad \text{with} \quad y = \frac{x}{(T - t)^{1/5}}$$

where $f(y)$ is supported on a compact interval [42, 26]. Moreover, there are only certain masses allowable for the similarity solution, so as in the previous example a portion of the initial data must pinch-off if the initial data is sufficiently large. Depending on the initial data, this pinch-off may be before the blow-up time or else possibly precisely at the blow-up time. We have chosen the parameter $m = 2.65$ to illustrate this. Because of the pinch-off behaviour near the boundary of the blow-up region, we cannot simply use the solution u for the monitor function but need the second-derivative as well to ensure that the numerical solution remains positive in the region where the curvature is very high. We have added the scaled curvature to place additional points near the edge of the support of the emerging similarity solution.

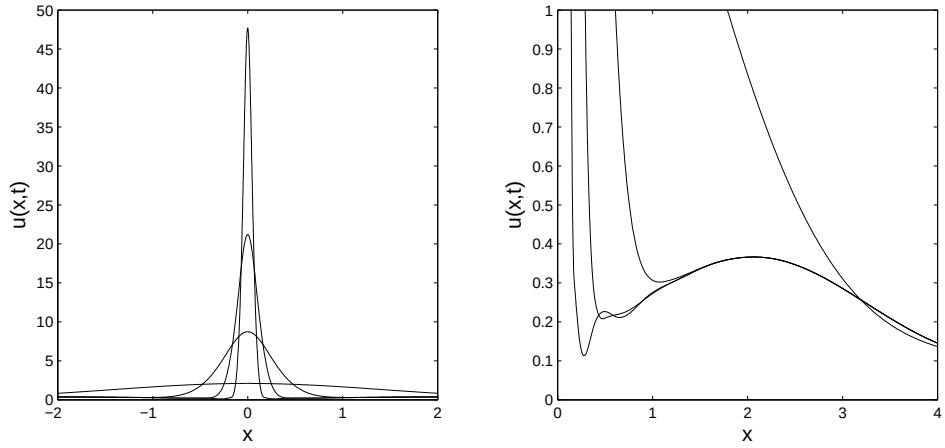

Figure 7: Blow-up solutions of an unstable thin-film equation. (Left) Sample solutions. (Right) Details near the edge of the support of the blow-up solution. As the magnitude of the solution increases at the origin, there is also a second singularity similar to the pinch-off singularity of the previous example. This point requires careful resolution.

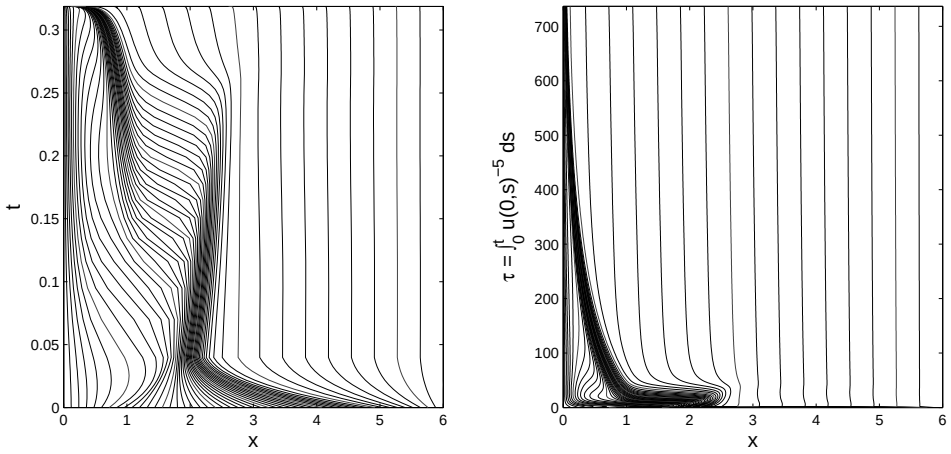

Figure 8: (Left) Grid in the physical co-ordinates. Initially the points cluster where the second derivative of the initial data is large but eventually get pulled in to the blow-up region. (Right) Grid in computational co-ordinates and rescaled time. Notice in both cases the essentially uniform grid in the outer region away from the blow-up point. Also observe the smooth mesh trajectories in the rescaled time co-ordinate as blow-up is approached.

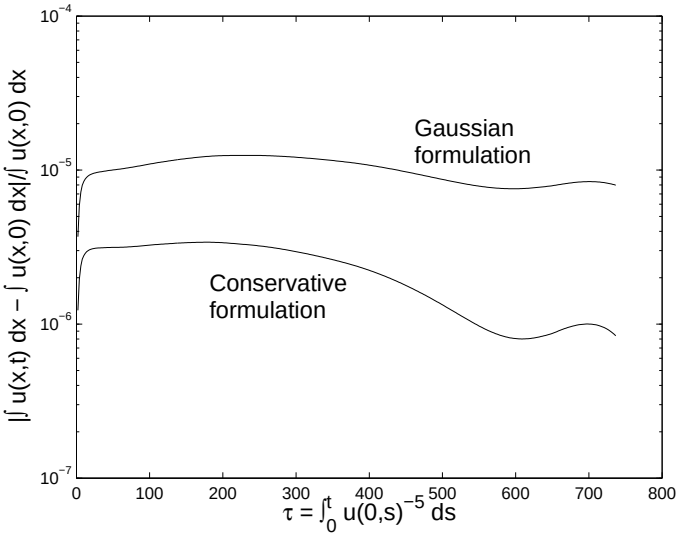

Figure 9: Comparison of error for mass in solving problem (7.10). The Gauss method error is an order of magnitude worse at the final time. This difference is more pronounced for less stringent tolerances and less pronounced for more stringent tolerances and more grid nodes.

8 Conclusions

We have discussed the implementation of a high-resolution collocation method to integrate nonlinear fourth-order evolutionary equations on adaptive grids. By using a cell-averaged discretization we are able to satisfy a general discrete conservation property and also reduce stiffness in the associated ODE problem. By carefully choosing the monitor function we are also able to resolve very delicate solution features.

The resulting software, MOVCOL4, has been designed with a simple easily expandable interface making it appropriate to solve a general class of fourth-order problems which are of interest to an increasingly wide range of researchers working on PDEs involving high-order equations. Advantages of MOVCOL4 include its ability to satisfy conservation equations, its high order accuracy, the local stencil (versus finite differences, which can cause difficulties for high-order equations [40]), its relative ease of use, the ability to handle high derivatives (e.g., in boundary conditions) directly, its robust adaptivity, and the facility to build scale-invariance into the MMPDE by a suitable choice of monitor function. We have demonstrated here that this code can be used to solve a variety of challenging problems with distinct features. We believe that it can be a useful tool for researchers from many different areas.

There are a number of questions remaining about which formulations of fourth-order problems prove the most important and useful computationally. Overall solution complexity can increase when going from second to fourth order, so using the best formulation becomes correspondingly more important. Understanding of this issue will increase as experience is gained solving ever more challenging problems using codes such as this one.

It should be noted that MOVCOL4 and a related code have been used to compute solutions to many other problems, e.g., see [17, 13, 19, 14, 28, 25, 26]. Many of these demonstrate further flexibility in the codes, particularly for solving blow-up problems. In our limited experience, directly solving fourth-order problems with MOVCOL4 generally works at least as well as, and often substantially better than, a method applied to a converted system of second order problems. In addition, for each of the problems, with solution features which vary over many different scales, having the adaptive features

which require no a priori knowledge about the particular solution can be essential.

In conclusion, numerical methods are indispensable for solving the many challenging fourth-order problems having complex solution behaviour, largely because the traditional theoretical tools are inapplicable. We expect that software like MOVCOL4 can be extremely useful both to underpin many of the numerical studies and, in other situations, to provide necessary verification of results for scientists who develop their own analytical and numerical methods with features tailored to their particular applications. Conversely, increased analysis of fourth-order problems will play an indispensable role in designing and refining codes such as MOVCOL4, as it would be naive to view current codes as a panacea for understanding the complex structures of these challenging nonlinear problems.

Acknowledgements: The authors are most grateful to Weizhang Huang for his insights and encouragement, and to Andrea Bertozzi for helpful comments on an early version of the paper.

References

[1] U. M. Ascher, J. Christiansen, and R. D. Russell, *A collocation solver for mixed order systems of boundary value problems*, Math. Comp. **33** (1979), 659–679.

[2] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Prentice Hall, New Jersey, 1988.

[3] J.W. Barrett, J.F. Blowey, and H. Garke, *Finite element approximation of a fourth-order nonlinear degenerate parabolic equation*, Numer. Math. **80** (1998), 525–556.

[4] G. Beckett, J. A. Mackenzie, A. Ramage, and D.M. Sloan, *On the numerical solution of one-dimensional PDEs using adaptive methods based on equidistribution.*, J. Comp. Phys. **167** (2001), no. 2, 372–392.

[5] G. Beckett, J. A. Mackenzie *On a uniformly accurate finite difference approximation of a singularly perturbed reaction-diffusion problem using grid equidistribution.*, J. Comp. App. Math. **131** (2001), no. 1-2, 381–405.

[6] E. Beretta, M. Bertsch, and R. Dal Passo, *Nonnegative solutions of a fourth-order nonlinear degenerate parabolic equation*, Arch. Rat. Mech. Anal. **129** (1995), 175–200.

[7] M. Berger and R. Kohn, *A rescaling algorithm for the numerical calculation of blowing-up solutions*, Comm. Pure Appl. Math. **41** (1988), no. 6, 841–863.

[8] F. Bernis and A. Friedman, *Higher-order nonlinear degenerate parabolic equations*, J. Diff. Eq. **83** (1990), 179–206.

[9] A.L. Bertozzi, *Loss and gain of regularity in a lubrication equation for thin viscous films*, Free Boundary Problems: Theory and Applications, Proceedings of the International Colloquium on Free Boundary Problems, Toledo, Spain, June 1993 (J. I. Díaz, M. A. Herrero, A. Liñán, and J. L. Vázquez, eds.), Pitman Research Notes in Mathematics Series, vol. 323, 1995, pp. 72–85.

[10] A. L. Bertozzi, *The mathematics of moving contact lines in thin liquid films*, Notice of the AMS **45** (1998), 689–697.

[11] A.L. Bertozzi and M.C. Pugh, *The Lubrication Approximation for Thin Viscous Films: Regularity and Long Time Behavior of Weak Solutions*, Comm. Pure. Appl. Math., **49** (1996), no. 2, 85–123.

[12] A.L. Bertozzi and M.C. Pugh, *Long-wave instabilities and saturation in thin film equations*, Comm. Pur. Appl. Math. **LI** (1998), 625–651.

[13] C. J. Budd, S. Chen, and R. D. Russell, *New self-similar solutions of the nonlinear Schrödinger equation with moving mesh computations*, J. Comp. Phys. **152** (1999), no. 2, 756–789.

[14] C. J. Budd, V. A. Galaktionov, and J. F. Williams, *Self-similar blow-up in higher-order semilinear parabolic equations*, SIAM J. App. Math., **64** (2004), no. 5, 1775–1809.

2003.

[15] C. J. Budd, W. Huang, and R. D. Russell, *Moving mesh methods for problems with blow-up*, SIAM J. Sci. Comput. **17** (1996), no. 2, 305–327.

[16] C. J. Budd and M. D. Piggott, *The geometric integration of scale-invariant ordinary and partial differential equations*, J. Comput. Appl. Math. **1-2** (2001), no. 128, 399–422.

[17] C.J. Budd, R. Caraterro, and R.D. Russell, *Precise computations of chemotactic collapse using moving mesh methods*, J. Comp. Phys.,**202** 2005, 462-487.

[18] C.J. Budd and J.F. Williams, *Uniform error estimates for PDEs with singularities*, In preparation, 2005.

[19] C.J. Budd, V. Rottschäffer, and J.F. Williams, *Blow-up multi-bump self-similar solutions to the complex Ginzburg-Landau equation*, SIAM App. Dyn. Sys., **4** (2005), 649–678.

[20] K. Clark, J. E. Flaherty, and M. S. Shephard, *Special Issue on Adaptive Methods for Partial Differential Equations*, Applied Numerical Mathematics, **14**, 1994.

[21] W. Cao, W. Huang, and R. D. Russell, *An error indicator monitor function for an r-adaptive finite-element method.*, J. Comput. Phys. **170** (2001), no. 2, 871–892.

[22] A.R. Champneys, P.J. McKenna, and P.A. Zegeling, *Solitary waves in nonlinear beam equations; stability, fission and fusion*, Non. Dyn. **21** (2000), 31–53.

[23] C. de Boor, *Good approximation by splines with variable knots. II*, Springer Lecture Note Series, vol. 363, Springer-Verlag, Berlin, 1973.

[24] C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, Berlin, New York, 1978.

[25] J. D. Evans, V. A. Galaktionov, and J. F. Williams, *Blow-up and global asymptotics of the unstable Cahn-Hilliard equation with a homogeneous nonlinearity*, submitted to SIAM J. Math. Anal.(`http://www.maths.bath.ac.uk/MATHEMATICS/preprints.html`), 2004.

[26] J. D. Evans, V. A. Galaktionov, J. R. King and J. F. Williams, *Blow-up and global asymptotics of an unstable thin-film equation with homogeneous nonlinearity*, in preparation, 2005.

[27] C. W. Gear, L. Petzold, *ODE methods for the solution of differential/algebraic systems*, SIAM Journal on Numerical Analysis, **21**, (1984), 716–728.

[28] V. A. Galaktionov and J. F. Williams, *Blow-up in a fourth-order semilinear parabolc equation from convection explosion theory*, Euro. J. Appl. Math, **1**4 (2003), 745–764.

[29] D. F. Hawken, J. J. Gottlieb and J. S. Hansen, *Review of some adaptive node-movement techniques in finite element and finite difference solutions of PDEs*, Journal of Computational Physics bf 95, 254-302, 1991.

[30] W. Huang, *Convergence analysis of finite element solution of one-dimensional singularly perturbed differential equations on equidistributing meshes*, Int. J. Numer. Anal. Modeling 2 (2005), 57 – 74.

[31] W. Huang, Y. Ren, and R. D. Russell, *Moving mesh methods based on moving mesh partial differential equations*, J. of Comp. Phys. **113** (1994), no. 2, 279–290.

[32] ———, *Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle*, SIAM J. Numer. Anal. **31** (1994), no. 3, 709–730.

[33] W. Huang and R. D. Russell, *A moving collocation method for solving time dependent partial differential equations*, Appl. Num. Math. **20** (1996), no. 1-2, 101–116.

[34] ———, *Analysis of moving mesh partial differential equations with spatial smoothing*, SIAM J. Numer. Anal. **34** (1997), no. 3, 1106–1126.

[35] J. A. Mackenzie and M. L. Robertson, *A moving mesh method for the solution of the one-dimensional phase-field equations.*, J. Comput. Phys. **181** (2002), no. 2, 526–544.

[36] E. V. L. de Mello, O. T. da Silveira Filho, *Numerical study of the Cahn-Hilliard equation in one, two and three dimensions*, Physica A **347** (2005), 429–443.

[37] L. Peletier and W. Troy, *Spatial Patterns: Higher-order Models in Physics and Mechanics*, Birkhäuser, New York, 2001.

[38] L. Petzold, *A Description of DASSL: A Differential /Algebraic System Solver*, SAND82-8637, Sandia Labs, Livermore, Cal., 1982.

[39] L. G. Reyna and M. J. Ward, *Metastable internal layer dynamics for the viscous Cahn-Hilliard equation*, Meth. Appl. Anal. **2** (1995), 285–306.

[40] P. Saucez, A. Vande Wouwer, W. E. Schiesser, and P. Zegeling, *Method of lines study of nonlinear dispersive waves*, submitted for publication.

[41] J. Stockie, J. A. MacKenzie, and R. D. Russell, *A moving mesh method for one-dimensional hyperbolic conservation laws.*, SIAM J. Sci. Comput. **22** (2000), no. 5, 1791–1813.

[42] T. P. Witelski, A. J. Bernoff, and A. L. Bertozzi, *Blow-up and dissipation in a critical-case unstable thin film equation*, Accepted to Euro. J. Appl. Math., 2003.

[43] Z. Wu, J. Zhao, J. Yin, and H. Li, *Nonlinear Diffusion Equations*, World Scientific, New Jersey, 2001.

[44] L. Zhornitskaya and A. L. Bertozzi, *Positivity preserving numerical schemes for lubrication type equations*, SIAM JNA **37** (2000), 523–555.