# Implementation of a Moving Collocation Method for Solving Time Dependent Partial Differential Equations

by

Debasree Raychaudhuri

B.Sc.(Hons.), University of Calcutta, 1985

M.Sc., Indian Institute of Technology, Kanpur, 1990

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the Department
of
Mathematics and Statistics


SIMON FRASER UNIVERSITY
July 1995

# APPROVAL

**Name:** Debasree Raychaudhuri

**Degree:** Master of Science

**Title of thesis:** Implementation of a moving collocation method for solving time dependent partial differential equations

**Examining Committee:** Dr. G.A.C. Graham
Chair

Dr. R.D. Russell
Senior Supervisor

Dr. T. Tang

Dr. M. Trummer

Dr. W. Huang, External Examiner
Assistant Professor, Dept. of Math.
The University of Kansas,
Lawrence, KS 66045 U.S.A.

**Date Approved:** July 27, 1995

PARTIAL COPYRIGHT LICENSE

I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

Implementation of a Moving Collocation Method for Solving Time Dependent Partial Differential Equations

Author: _____
(signature)

Debasree Raychaudhuri
(name)

July 27, 1995
(date)

# Abstract

In the last decade and half, a number of numerical methods have been developed, based on the Method of Lines (MOL) approach and mesh moving strategies. They are specially suitable to cater to the needs of a particular range of partial differential equations (PDE). These are time-dependent PDEs having large solution variations, such as shock waves, boundary layers or contact surfaces. Significant improvements in accuracy and efficiency are realized by adapting the mesh points concentration about these areas of large variation. In this thesis, we look into one such method and implement it to solve various time-dependent PDEs in one space dimension. In chapter 1, we begin by explaining the idea behind the MOL approach and the need for adaptive meshing. We then describe two contemporary numerical techniques, which apply these ideas towards solving time-dependent PDEs in one space dimension. Chapter 2 starts with a detailed description of MOVCOL, the Moving Collocation method under consideration. We then implement it for three different test problems, chosen from the existing numerical literature. We present and discuss the results to demonstrate the performance of the method, and move on to chapter 3. Here, a phase separation model is presented and our approach towards solving it using MOVCOL is discussed in detail. The computational results are presented and analyzed. We conclude the thesis in chapter 4, with a summary of our overall findings, and some ideas towards further research.

# Acknowledgments

I would like to thank my senior supervisor Dr. R.D. Russell, for the direction, encouragement and support that he gave me during the period of my research.

I would also like to thank Dr. Weizhang Huang for his prompt and very helpful suggestions to my questions.

Lastly I thank all my friends and well-wishers for making the process of writing this thesis a worthwhile experience.

# Dedication

**To my** *parents*

**and**

**my daughter** *Nirokhi*

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Numerical Methods in Solving Time-dependent Partial Differential Equations

## 1.1 Introduction

Differential equations and in particular, partial differential equations (PDEs), have always played an important role in modeling physical phenomena. The exact solution of a PDE is often hard to obtain. The difficulties arise partly from the governing PDE and partly from the complexities[1] of the physical problem. Since only a small number of PDEs have known analytical solutions, numerical methods of solving PDEs have become extremely useful, and in some cases the only available tool for solving partial differential equations. In this thesis, we concern ourselves with solving a particular range of time-dependent PDEs, in one space dimension. Using a numerical technique, we attempt to find the solution(s) which evolves in time from a given initial

[1] e.g. irregular boundaries etc.

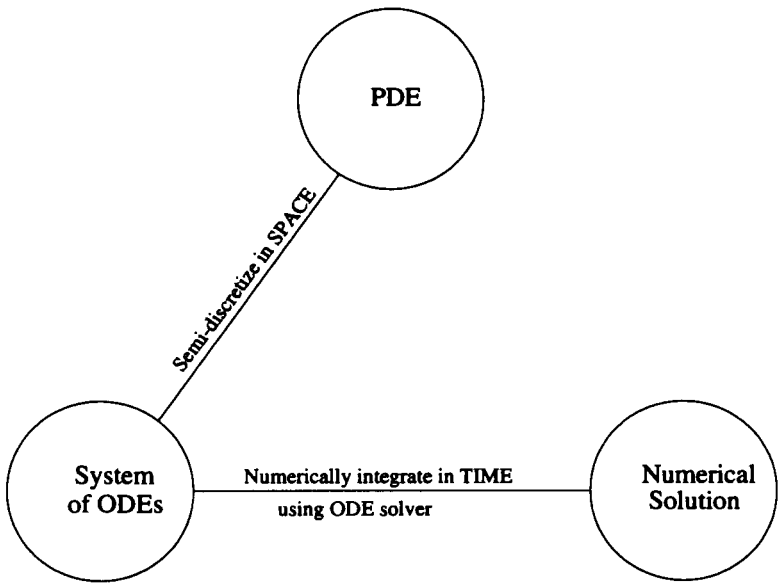


Figure 1.1: Method of lines approach

configuration. In this chapter, we discuss the tools of the method; the idea of the MOL (method of lines) approach, and the adaptive meshing strategy. We illustrate these ideas by taking a look at some of the important and highly used numerical solution techniques that use them, and move on to the next chapter for a detailed explanation of the method of our consideration.

## 1.2 Method of lines and adaptive meshing

The essence of MOL is to replace the original PDE with a system of ODEs and then to solve the system using any standard ODE integrator. This is done by replacing the spatial derivative terms in the PDE by their algebraic approximations which leads to a system of ODEs (usually stiff) in $t$, and then numerically integrating in time to generate the desired numerical solution. The MOL approach has given rise to many sophisticated PDE solvers in the recent past, especially for one-space-dimensional problems. With the improvement of stiff ODE solvers, these methods have become

reliable yet easy to implement tools in numerical analysis. Among the ODE solvers, one type has been particularly successful, in the terms of efficiency, robustness and reliability. Those are implicit Gear type BDF (backward differentiation formula) solvers. One particular ODE solver of this type has been used in the methods that we describe in the later part of this chapter.

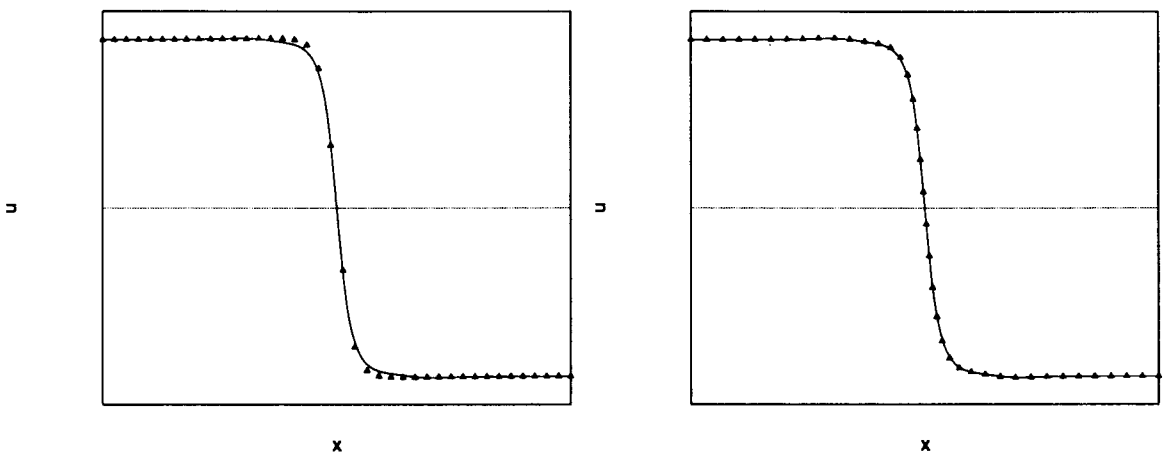


Figure 1.2: An example of *fixed* versus *moving* grid distribution for a solution curve with steep gradient.

### 1.2.1 Why adaptive mesh

In the cases of problems involving large solution variations, viz. shock waves, boundary layers and contact surfaces, the traditional fixed mesh approach is found to have some disadvantages. In order to adequately resolve the steep localized gradients which can occur in these types of solutions, the grid has to be very fine. However, if the grid is uniformly spaced, this implies that a very large number of grid points are required and that most of these are located in the regions where the solution is very smooth.

The obvious choice here is to use a non-uniform distribution of grid points so that the high spatial resolution is provided only in those regions where the non-linear

nature of the problem requires it. In other words, if the solution is steep, and it varies rapidly over a small region(s) in space, the grid points will be distributed so that the region(s) has enough points to resolve the steep gradient accurately (fig. 1.2). This is the fundamental concept of adaptive meshing. Two types of adaptive methods are currently in demand:

- **Moving mesh methods**, where a mesh equation which involves node speeds is employed to move a mesh having a fixed number of nodes so as to follow local non-uniformities of the solution.

- **Local refinement methods**, where uniform fine grids are added to coarser grids in regions where the solution is not adequately resolved.

In this chapter and the next, we are going to look into various adaptive methods of the first type, which take the method of lines approach. We start with two contemporary moving grid methods; one uses a finite difference technique and the other takes the finite element approach. The third and the final method, is of the moving collocation type, which we explain and implement in the later chapters.

From the user's point of view, an additional advantage of these moving grid methods is that they can be implemented in most of the MOL software packages based on sophisticated implicit stiff ODE/DAE solvers. We mention, for example, the BDF solvers developed by Gear, Hindmarsh, Petzold and others. Thus, the user only has to set some numerical control parameters, and formulate the mathematical problem in terms of the computing language. Both the spatial discretization and the temporal integration can then be left to the package[2]. Some examples of the control parameters are a local tolerance parameter for the numerical integration in time, the number of points for the spatial discretization, and some parameters controlling the grid movement.

[2]Here, by package we imply the moving grid PDE solver along with the ODE integrator.

## 1.3 Techniques currently in use

### 1.3.1 A moving finite difference method

The first method we are going to take a look at, implements the idea of MOL and solves a system of time-dependent PDEs in one space dimension. It is of Lagrangian type and introduces the semi-discretization in a moving reference frame. This method was discussed in detail in [29] by Verwer, Blom, Furzeland and Zageling. The grid movement for this case is based on the principle of spatial equidistribution of the nodes, the spatial discretization is done using standard central differencing technique, and a sophisticated BDF code is employed to carry out the numerical time integration.

Consider the following system of PDEs in one space dimension,

$$u_t = f(u,x,t), \qquad x_L < x < x_R, \quad t > 0 \tag{1.1a}$$

with the initial and boundary conditions

$$u(x,0) = u^0(x), \quad x_L < x < x_R \quad \text{and} \quad b(u,x,t) = 0, \quad x = x_L, x_R, \quad t > 0 \tag{1.1b}$$

Here $f$ and $b$ are spatial differential operators and the problem is assumed to have an unique solution and also to be well-posed. The derivation is started with an MOL approach as follows:

Consider smooth, continuous time trajectories for the mesh points as

$$x_L = X_0 < ... < X_i(t) < X_{i+1}(t) < .... < X_{N+1} = x_R, \quad t > 0.$$

Introducing the total derivative

$$u' = x' u_x + u_t = X_i' u_x + f(u, X_i(t), t), \qquad 1 \le i \le N. \tag{1.2}$$

along $x(t) = X_i(t)$, and spatially discretizing the space operators $\frac{\partial}{\partial x}$ and $f$, for each

fixed $t$, we obtain the semi-discrete scheme, given by

$$U_i' = X_i' \frac{(U_{i+1} - U_{i-1})}{(X_{i+1} - X_{i-1})} + F_i, \qquad t > 0, \quad 1 \leq i \leq N. \tag{1.3}$$

Here, $U_i(t)$ represents the semi-discrete approximation to the exact PDE solution $u$ at the point $(x,t) = (X_i(t), t)$ and $F_i$ is the finite difference replacement for $f(u,x,t)$ at this point.

It is then changed to the more compact form

$$U' = X'D + F, \qquad t > 0, \quad U(0) \text{ given} \tag{1.4}$$

where

$$X = [X_1, ...X_N]^T, \quad U = [U_1^T, ...U_N^T]^T, \quad F = [F_1^T, ...F_N^T]^T,$$
$$D_i = (U_{i+1} - U_{i-1})/(X_{i+1} - X_{i-1}), \qquad D = [D_1^T, ...D_N^T]^T.$$

The equation (1.4) represents the semi-discrete system to be numerically integrated in time to obtain the solution $U$.

Next an equation is constructed defining the time-dependent grid $X$ implicitly in terms of the continuous-time solution $U$. This equation is based on the idea of distributing the spatial variation in the solution equally over the interval $[x_L, x_R]$ and is thus called the *spatial equidistribution equation.* It is used to move the nodes and is given by

$$n_{i-1}/M_{i-1} = n_i/M_i, \qquad 1 \leq i \leq N. \tag{1.5a}$$

where $n_i = \Delta X_i^{-1}$, and $\Delta X_i = X_{i+1} - X_i$ for $0 \leq i \leq N$. Here $M_i \geq \alpha > 0$ represents a monitor value that reflects the spatial variation over the i-th subinterval $[X_i, X_{i+1}]$. $M_i$ is a semi-discrete form of a solution functional $m(u)$, containing one or more spatial derivatives of $u$. A commonly used (scalar form) of it is

$$m(u) = \sqrt{\alpha + (u_x)^2} \tag{1.5b}$$

Note that (1.5b) gives the *arclength monitor* for $\alpha = 1$.

Grid-smoothing is done to ensure a spatially smooth grid and to avoid oscillations for evolving time. The spatial grid smoothing is introduced to guarantee that the ratios of adjacent grid intervals satisfy the inequality

$$\kappa/(\kappa+1) \leq n_{i-1}/n_i \leq (\kappa+1)/\kappa \tag{1.6}$$

where $\kappa > 0$ is the spatial smoothing parameter, which controls the minimum and maximum interval length. In applications, $\kappa$ is normally set equal to 1 or 2. The temporal grid smoothing introduces the derivative of the point concentration $n_i$ in the grid equation (1.5a). This serves to prevent the grid movement from adjusting solely to the new monitor values. Instead, the introduction of $\tau n'$ forces the grid to adjust over a time interval of length $\tau$ (the time smoothing parameter) from old to new monitor values, i.e., the parameter $\tau$ acts as a delay factor. The aim here is to avoid temporal oscillations and hence to obtain a smoother progression of $X(t)$. Usually, the value of $\tau$ is chosen to be close to 0.

The incorporation[3] of the grid smoothing in the mesh moving equation (1.5a) gives rise to the following ODE system

$$\tau B X' = g, \qquad t > 0, \quad X(0) \text{ given}, \tag{1.7}$$

where $B$ is a penta-diagonal matrix and $g$ a solution-dependent vector. The above equation along with

$$U' - X'D = F, \tag{1.8}$$

are then numerically integrated in time, using the stiff ODE solver DASSL [25]. It is a BDF code which solves systems of implicit differential and algebraic equations having the general form

$$G(T, Y, \dot{Y}) = 0, \qquad t > 0 \tag{1.9}$$

[3]For details of how it is done, we refer the interested reader to [29]

and subject to appropriate[4] initial conditions. It can solve stiff systems with variable order of accuracy and temporal step size, and also problems where singular matrices occur. It is capable of exploiting the sparse and banded structure of a matrix, thus saving space and time.

In spite of the efficiency and robustness of this method, a few problems were reported in [29]. The integrations were interrupted due to Newton convergence test failures, for some of the test cases; especially so, when using extremely fine grids. According to the authors of [29], this could either be due to the fact that the local error and Newton convergence test was applied to $X_i$ and not $\Delta X_i$, or poor prediction of $X_i$ generated in the preparation of actual BDF step, causing convergence problem. The authors admit that both these problems need further attention.

### 1.3.2 A moving finite element method

The second method we explore, also solves system of time-dependent PDEs in one space dimension using the MOL approach. But it differs in the manner the spatial discretization and the grid movement are done. A Galerkin finite element approximation is used for discretizing in space and the mesh is moved so as to equidistribute the spatial component of the discretization error in $H^1$. This method was developed by Adjerid and Flaherty [1] at the *Rensselaer Polytechnic Institute.*

Consider the system of PDEs having the form

$$(1.10a)\quad Lu = M(x,t)u_t + f(x,t,u,u_x) - [D(x,t,u)u_x]_x = 0, \qquad 0 < x < 1, \quad t > 0$$

subject to the initial and boundary conditions

$$(1.10b)\qquad u(x,0) = u^0(x), \qquad 0 \le x \le 1$$

$$(1.10c)\qquad \text{either } u_i(x,t) = a_i(t) \quad \text{or} \quad \sum_{j=1}^{m} D_{ij}u_{jx}(x,t) = a_i(t)$$

[4]The initial approximations of $Y$ and $\dot{Y}$ must be consistent, i.e. they must satisfy (1.9).

$$\text{at } x = 0,1, \quad t > 0, \quad i = 1,2...m.$$

The problem (1.10) is assumed to have an isolated solution.

Next with the assumption $u \in H_E^1$, where the subscript E denotes that $u$ essentially satisfies (1.10c), a test function $v \in H_0^1$ is selected to construct a weak form of (1.10). Equation (1.10) is multiplied by $v$, integrated over $0 \leq x \leq 1$, and then integrated again by parts to obtain

$$(v, Mu_t) + (v, f) + a(v, u) = 0, \qquad \forall v \in H_0^1, \quad t > 0 \tag{1.11a}$$

where

$$(v, u) = \int_0^1 v(x,t)^T u(x,t) dx \tag{1.11b}$$

$$a(v, u) = \int_0^1 v_x^T D(x,t,u) u_x dx - v^T D(x,t,u) u_x \Big|_0^1 \tag{1.11c}$$

which upon discretization gives

$$(V, MU_t) + (V, f) + a(V, U) = 0, \qquad \forall V \in S_0^N, \quad t > 0 \tag{1.12a}$$

$$(V, U) = (V, u^0), \qquad t = 0 \tag{1.12b}$$

where $U \in S_E^N$ and $V \in S_0^N$ are finite-dimensional approximations to $u$ and $v$ respectively, and $U \in S_E^N$ is the solution sought. Here, $S_E^N$ and $S_0^N$ are finite-dimensional subspaces of $H_E^1$ and $H_0^1$ respectively. A partition

$$\Pi(t, N) = \{0 = x_0(t) < x_1(t) < ... < x_N(t) = 1\} \tag{1.13}$$

is introduced, in order to subdivide $(0,1)$ into $N$ subintervals $(x_{i-1}(t), x_i(t))$, $i = 1,2,..N$, $t \geq 0$. $U$ and $V$ are then selected so as to be piecewise linear polynomials with respect to (1.13). One example of which is the following equation

$$U(x,t) = \sum_{i=0}^{N} U_i(t) \phi_i(x, \Pi(t, N)) \tag{1.14a}$$

where

$$\phi_i(x,\Pi(t,N)) = \begin{cases} \dfrac{x - x_{i-1}(t)}{x_i(t) - x_{i-1}(t)}, & x_{i-1}(t) \le x \le x_i(t) \\ \dfrac{x - x_{i+1}(t)}{x_i(t) - x_{i+1}(t)}, & x_i(t) \le x \le x_{i+1}(t) \\ 0 \quad \text{otherwise.} \end{cases} \tag{1.14b}$$

Next a quadratic error estimate is done by substituting $u(x,t) = U(x,t) + e(x,t)$ into (1.10a) to get

$$(v, M(U_t + e_t)) + (v, f(.,t,U+e,U_x+e_x)) + a(v,U+e) = 0 \quad \forall v \in H_0^1, \quad t > 0 \tag{1.15}$$

and then replacing $e$ by a finite-dimensional approximation $E$ consisting of piecewise quadratic functions, i.e.,

$$E(x,t) = \sum_{i=0}^{N} E_i(t)\Psi_i(x,\Pi(t,N)) \tag{1.16}$$

where

$$\Psi_i(x,\Pi(t,N)) = \begin{cases} \dfrac{4(x - x_{i-1}(t))(x_i(t) - x))}{(x_i(t) - x_{i-1}(t))^2}, & x_{i-1}(t) \le x \le x_i(t) \\ 0 \qquad\qquad \text{otherwise.} \end{cases} \tag{1.17}$$

The error estimate is calculated by computing the approximation

(1.18a)

$$(V, M(U_t + E_t)) + (V, f(.,t,U+E,U_x+E_x)) + a(V,U+E) = 0, \quad \forall V \in \hat{S}_0^N, \quad t > 0$$

along with the initial conditions

$$(V,E) = (V, u^0 - U), \quad \forall V \in \hat{S}_0^N, \quad t = 0. \tag{1.18b}$$

The error estimate is then used to control the motion of the mesh through the equation

$$\dot{x}_{i+1} - 2\dot{x}_i + \dot{x}_{i+1} = -\lambda(\|E_{i+1}\Psi_{i+1}\|_1 - \|E_i\Psi_i\|_1), \quad t > 0 \tag{1.19}$$

where $\lambda$ is positive.

Finally the three sets of equations (1.11), (1.18), and (1.19) are solved respectively, for the piecewise linear approximation, the quadratic error estimate and the mesh positions using the backward difference code DDASSL which is the double precision version of DASSL.

One difficulty encountered here is finding an optimal value of $\lambda$. Where a large enough $\lambda$ keeps the mesh close to equidistributed (see [1]) it also causes the stiffness in (1.19) making the system expensive to integrate. Developing an automatic procedure to balance the competing effects of stiffness and equidistribution has been considered in [2], where a local refinement is done, while keeping $\lambda$ of modest size. This will however, complicate the data structure, representing the solution, error, and the mesh but may produce more efficient results.

## 1.4 Summary

In this chapter we discussed the relevance of PDEs and their numerical solutions, and explained the idea behind the *Method of lines* approach; the semi-discretization in space, and then numerical integration of the resultant ODE system. We then introduced the idea of adaptive meshing and justified its need in cases of rapid solution variations. Two prominent methods, both of which implement MOL and adaptive mesh techniques but differ in the discretization and mesh moving procedures are explored and their difficulties are presented. In the next chapter, we discuss the third and our final method MOVCOL, a moving collocation method in its experimental stage. It is then implemented to solve a few time-dependent one space dimension PDEs, and the results are presented.

# Chapter 2

# An Introduction to MOVCOL

## 2.1 Introduction

Here we discuss the moving mesh method under consideration, wherein the physical PDE and the mesh equation are solved simultaneously for the physical solution and the mesh, using the MOL approach. It is a moving collocation method, thus named MOVCOL, developed by Huang and Russell [23] at *Simon Fraser University.* In the next section, we explain the method in detail. In section 2.3 we illustrate and analyze the performance using some test problems. In the last section we summarize the findings and move on to the next chapter to investigate a phase separation model.

## 2.2 Details of the method

As mentioned above, MOVCOL takes the MOL approach and applies that to solve the physical PDE (s), together with an additional PDE that moves the mesh. The current MOVCOL model solves a system of second order parabolic PDEs of the general divergence form. It uses a collocation discretization for the physical PDE (s) and the three point finite difference discretization for the moving mesh PDE (hereafter referred

to as an MMPDE). The details of the method are as follows:

Consider a system of second order parabolic PDEs of the general divergence form

$$F(t,x,u,u_x,u_t,u_{xt}) = \frac{\partial}{\partial x}G(t,x,u,u_x,u_t,u_{xt}) \tag{2.1a}$$

for $x^L(t) < x < x^R(t)$ and $t_a < t < t_b$, supplemented with the boundary conditions

$$B^L(t,x^L,{x_t}^L,u(x^L,t),u_{xx}(x^L,t),u_t(x^L,t),u_{xt}(x^L,t)) = 0 \tag{2.1b}$$

$$B^R(t,x^R,{x_t}^R,u(x^R,t),u_{xx}(x^R,t),u_t(x^R,t),u_{xt}(x^R,t)) = 0$$

and the initial conditions

$$u(x,t_a) = U(x), \qquad x^L(t_a) < x < x^R(t_b) \tag{2.1c}$$

where $F, G, B^L, B^R, u$ and $U$ are vector-valued functions of length $NPDE$ (number of PDEs). Throughout, it is assumed that the system thus defined is always well posed.

As mentioned before, the physical PDE is discretized in space, using *Cubic Hermite Collocation*, where MOVCOL approximates the physical solution u(x,t) on the mesh

$$x_1(t) := x^L(t) < \ldots < x^R(t) := x_N(t) \tag{2.2}$$

by $v(x,t)$, where

(2.3)

$$v(x,t) = v_i(t)\Phi_1(s^{(i)}) + v_{x,i}(t)H_i(t)\Phi_2(s^{(i)}) + v_{i+1}(t)\Phi_3(s^{(i)}) + v_{x,i+1}(t)H_i(t)\Phi_4(s^{(i)}),$$

is a piecewise cubic Hermite interpolating polynomial approximation. Here, $x \in [x_i(t), x_{i+1}(t)]$, $i = 1 \ldots N-1$, and $v_i(t)$, $v_{x,i}(t)$ denote the approximations to $u(x_i(t),t)$ and $u_x(x_i(t),t)$ respectively. The local coordinate $s^{(i)}$ is defined as

$$s^{(i)} := \frac{(x - x_i(t))}{H_i(t)}, \qquad H_i(t) := x_{i+1}(t) - x_i(t) \tag{2.4}$$

and $\Phi_i$'s are the *Shape Functions*, defined by

$$\Phi_1(s) = (1+2s)(1-s^2), \qquad \Phi_2(s) := s(1-s)^2 \tag{2.5}$$

$$\Phi_3(s) := (3-2s)s^2, \qquad \Phi_4(s) := (s-1)s^2.$$

Note that $v_x$,$v_{xx}$,$v_t$ and $v_{xt}$ are the unknowns to be calculated. The substitution of equation (2.3) in the PDE (s) yield a residual, which is set equal to zero at the *collocation points*, in order to solve for these unknowns. The collocation points for each interval are given by

(2.6*a*) $$X_{ij} := X_i + s_j H_i \qquad j = 1,2$$

where $s_1$ and $s_2$ are the two *Gauss points* (zeros of the second degree Legendre polynomial) defined as

(2.6*b*) $$s_1 = 0.5(1 - 1/\sqrt{3}) \qquad s_2 = 0.5(1 + 1/\sqrt{3})$$

Note that the collocation method has several important advantages over the finite difference methods. It provides a higher order of convergence, gives a continuous approximate solution and easily handles general boundary conditions, yet being easy to code.

The MMPDEs, which control the node speeds are derived from an equidistribution principle [21]. The basic strategy here, is to choose a reliable *monitor function* M(x,t) which provides some measure of the computational difficulties in the solution of the physical PDEs, and to equidistribute it over the mesh. The most common choice is the *arclength monitor function.*

(2.7) $$M(x,t) = \sqrt{1 + |u_x|^2}, \qquad \text{where 'x' is the spatial variable .}$$

Mathematically speaking, the goal is to find mesh functions $\{x_i(t)\}$, i = 2....N-1, such that

(2.8) $$\int_{x_i(t)}^{x_{i+1}(t)} M(x,t)dx = \frac{1}{N-1}\int_{x_L}^{x_R} M(x,t)dx, \qquad \text{i=1...N-1}$$

For the purpose of mesh adaptation, the physical and computational spatial coordinates x and $\xi$ are related by a differentiable monotone coordinate transformation

$$x = x(\xi, t) \qquad \text{where } \xi \in [0, 1] \tag{2.9}$$

subject to the boundary conditions

$$x(0,t) = x_L(t), \qquad x(1,t) = x_R(t)$$

Notice that equation (2.8) is the continuous counterpart of its discrete form, which determines $x(\xi, t)$. This coordinate transformation translates the rapid transition in $x$ into a much more gradual one in $\xi$, thus making possible the use of equal spacing in $\xi$. Various MMPDEs have been developed (see [20]), by differentiating the continuous equidistribution equations. One example of the basic MMPDEs is

$$\tau \frac{\partial}{\partial \xi}\left(M \frac{\partial \dot{x}}{\partial \xi}\right) = -\frac{\partial}{\partial \xi}\left(M \frac{\partial x}{\partial \xi}\right) \tag{2.10}$$

where $\dot{x} = \frac{\partial x}{\partial t}$ , $\tau$ a nonnegative temporal smoothing parameter, and $\xi$ is the computational coordinate defined above.

When large solution variations occur, the monitor function can be fairly non-smooth in space, and some kind of smoothing is necessary, to prevent the mesh from changing too rapidly. Following is one example of a smoothed MMPDE from [22]

$$\frac{\partial}{\partial \xi}\left(\frac{1}{M}(1 - \lambda^{-2}\frac{\partial^2}{\partial \xi^2})(\tau \dot{n} + n)\right) = 0 \tag{2.11}$$

where $n = \left(\frac{\partial x}{\partial \xi}\right)^{-1}$ is so-called concentration function, $\lambda$ is a positive number. This is also the MMPDE that we use in our implementation of MOVCOL for the problems. MOVCOL currently has the option for four different MMPDEs, two of which are the smoothed ones.

In summary, the moving collocation method consists of numerically solving the ODE system involving the collocation approximation (2.3) for the physical PDE,

the discrete mesh equation, and the corresponding boundary and initial conditions. This system is integrated in time using the stiff ODE solver DDASSL, which we briefly described in the last chapter. Most of the parameters that DDASSL requires are supplied by MOVCOL, except those such as error tolerances and the smoothing parameters $\tau$ (temporal) and $\gamma$ (spatial), thus keeping the user-intervention at a comfortable level.

In the next section, we discuss the results of numerical testing on a set of three different problems, two of which are frequently used test models in existing moving grid literature (e.g. see [16]). The set includes the well-known convection-diffusion equation of Burgers, a reaction-diffusion equation from combustion theory, and Allen-Cahn equation which models the process of grain boundary migration. It is worth noting that these three problems show different solution behaviour, and this makes, our tests with (only) three problems reasonably challenging.

## 2.3 A few numerical examples using MOVCOL

**Example 1 : Burgers' equation**

Our first example is Burgers' equation

$$u_t = \epsilon u_{xx} - u u_x, \quad 0 < x < 1, \quad t > 0 \tag{3.1a}$$

with $\epsilon = 10^{-4}$ and boundary conditions and initial values,

$$u(0,t) = u(1,t) = 0, \qquad t > 0 \tag{3.1b}$$

$$u(x,0) = 10x(x-1)(x-3/4) \tag{3.1c}$$

For small times and $\epsilon$ the solution is a pulse moving in a positive $x$ direction while steepening. At about $t = 0.6$ a shock layer forms near $x = 0.8$ and after a time of $O(1/\epsilon)$ the solution dissipates to zero. We consider the time interval $[0, 1]$ and output our solutions at $t = 0, 0.3, 0.6, 1.0, 1.4$, following [1]. For the time integration, we use fixed tolerances $atol = 10^{-3}$ and $rtol = 10^{-5}$ for all cases unless otherwise stated. The solutions for the given output times are all displayed on the one graph, in which the leftmost peak is the initial curve, and the rightmost peak is the solution at the last output time $t = 1.4$, with intermediate solutions in between, moving from left to right.

Fig. 2.1 shows the solution obtained by the moving mesh (MM) method with 25 nodes[1], with the reference solution, computed with 200 MM nodes (shown in solid lines) with $\tau = 10^{-3}$. Note that the computed solution shows reasonably good agreement with the reference solution even with 25 nodes. The mesh trajectories however, show a slight oscillation between the times $t = 0.4$ and 1.

Increasing the number of nodes to 40 takes care of this problem. Detailed results

[1]The ODE integrator failed the error test repeatedly for 20 nodes.

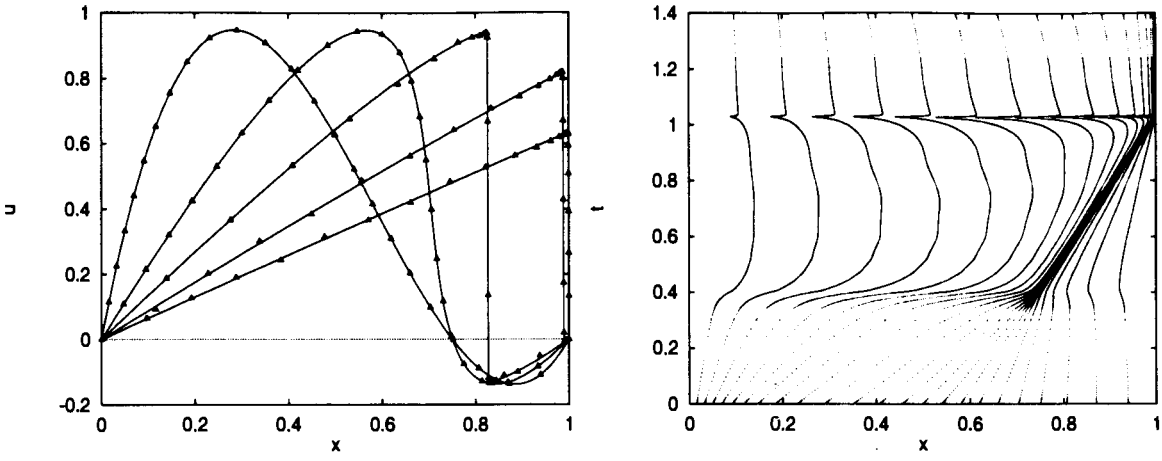


Figure 2.1: Solutions at $t = 0, 0.3, 0.6, 1, 1.4$ and the mesh trajectories for MM method with $n = 25$

for the MM method with 40 nodes are given in table 2.1. In figures 2.2 we display our solutions at different time points and the mesh trajectories for this case. Indeed, the computed solution here is visually indistinguishable from the reference solution.

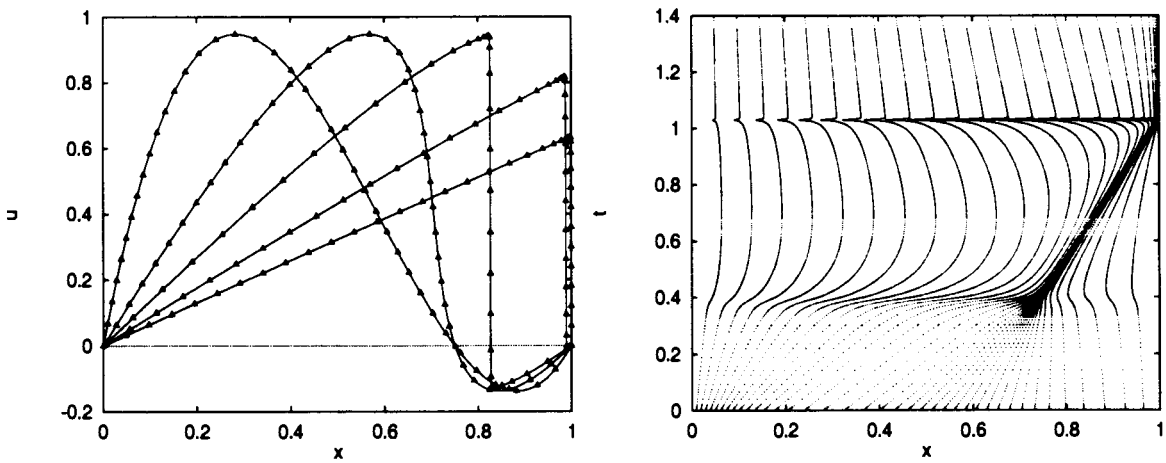


Figure 2.2: Solutions at $t = 0, 0.3, 0.6, 1, 1.4$ and the mesh trajectories for MM method with $n = 40$

Table 2.1 shows a comparison between the number of time steps (NTS) required by DDASSL to complete the time integration, the number of jacobian evaluations (JAC) computed by DDASSL, the number of error test failures (ETF) occurring in DDASSL, and the computer processing time (CPU) in second (s), for various runs with moving and fixed mesh methods. The computations, here as well as for the later problems were performed on a SPARC-server 20 work-station.

| n | | $\tau$ | $\gamma$ | NTS | JAC | ETF | CPU |
|---|---|---|---|---|---|---|---|
| 25 | | $10^{-3}$ | 1 | 810 | 88 | 38 | 33.70 |
| | | $10^{-2}$ | | 758 | 82 | 22 | 47.26 |
| 40 | | $10^{-3}$ | 1 | 636 | 75 | 21 | 46.45 |
| | | $10^{-5}$ | | 631 | 76 | 17 | 46.99 |
| | MM | | 2 | 624 | 81 | 25 | 49.41 |
| | | | 3 | - | - | - | - |
| 40 | | | | 446 | 15 | 2 | 14.48 |
| 80 | FM | | | 795 | 17 | 1 | 53.07 |
| 200 | | | | 1845 | 24 | 2 | 245.81 |

Table 2.1: Computational summary for Burgers' equation

We did some more testing to evaluate the performance of MOVCOL for different values of the temporal and spatial smoothing parameters $\tau$ and $\gamma$. As might be expected, for $\tau = 10^{-2}$ and more, the mesh movement is somewhat slow and oscillatory (graph not shown). Due to the small amount of diffusion, the semi-discrete solutions have a tendency to oscillate as soon as the grid becomes a little too coarse in the layer regions. This makes the problem difficult to solve, and in fact is the main cause for the relatively larger number of Jacobian updates. This also explains the slow and oscillatory movements of the mesh for a relatively large $\tau$. For smaller values of $\tau$, the method works better and more efficiently. For the spatial smoothing, however, $\gamma = 1$ is found to have given the best results in terms of reliability and efficiency. Increasing it only diminishes the efficiency. The integrator breaks down due to repeated error test failures for $\gamma = 3$.

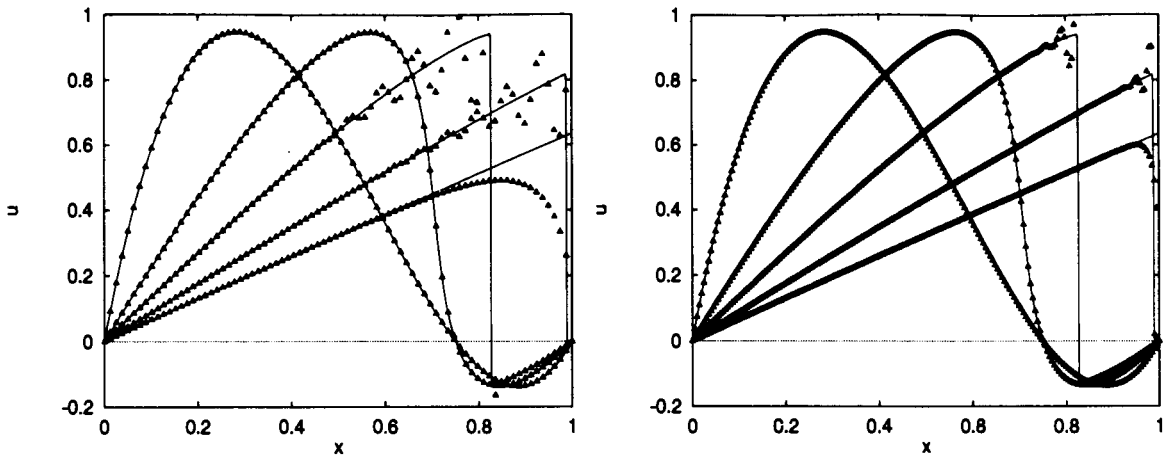


Figure 2.3: FM Solutions at $t = 0, 0.3, 0.6, 1, 1.4$ with $n = 80$ and $n = 200$

For comparison purposes, we try to compute the solution with a fixed mesh (FM) by freezing[2] the mesh movement in MOVCOL, and present the results. The solutions obtained even with $n = 200$, (almost five times that of the MM) fail to yield the same level of accuracy as the MM solutions. It places almost no points in the regions where the steep gradient occurs, and thus gives an inaccurate picture while consuming a huge amount of CPU time.

[2]MOVCOL has the option of solving the PDE (s) with a fixed collocation discretization.

**Example 2 : Reaction-diffusion equation**

Our next problem is a reaction-diffusion model in one space dimension, viz.

$$u_t = u_{xx} + \frac{Re^{\delta}}{a\delta}(1 + a - u)e^{-\frac{\delta}{u}} \qquad 0 < x < 1, \quad t > 0 \tag{3.2a}$$

$$u_x(0,t) = 0, \qquad u(1,t) = 1 \qquad t > 0 \tag{3.2b}$$

$$u(x,0) = 1, \qquad 0 \leq x \leq 1 \tag{3.2c}$$

where $R, a$, and $\delta$ are physical constants. The solution represents the temperature of a reactant in a chemical system. For small times, the temperature gradually increases from unity with a *hot spot* forming at $x = 0$. At a finite time ignition occurs and the temperature at $x = 0$ jumps suddenly from near unity to $1 + a$. A sharp flame front then forms and propagates towards $x = 1$ with exponential speed. The problem reaches a steady state once the flame propagates to $x = 1$. The degree of difficulty of the problem is determined by the value of $\delta$. Our first choice of problem parameters is $a = 1, R = 5$ and $\delta = 20$. For the current choice of parameters, the steady state is reached ([1], [16], [20]) slightly after time $t = 0.29$ and we take that as our end point for time integration and output our solutions at $t = 0.26, 0.27, 0.28, 0.29$.

In the plots for this case, the reference solution (solid lines) is obtained with $n = 200, \gamma = 1, \tau = 10^{-3}$ and $atol = rtol = 10^{-6}$. One main concern for this case is to detect the start of the ignition accurately as small errors at this stage can result in significantly larger global error later on. After some experiment we choose the tolerances $atol = rtol = 10^{-6}$ . Another parameter of concern is $\tau$.

Even though the start of ignition is detected accurately by a range of values for $\tau$ [see table 2.2], for a large enough $\tau$ (e.g.1), a nearly nonmoving mesh results, and the propagation of the flame front becomes very slow. After some experiments, we choose

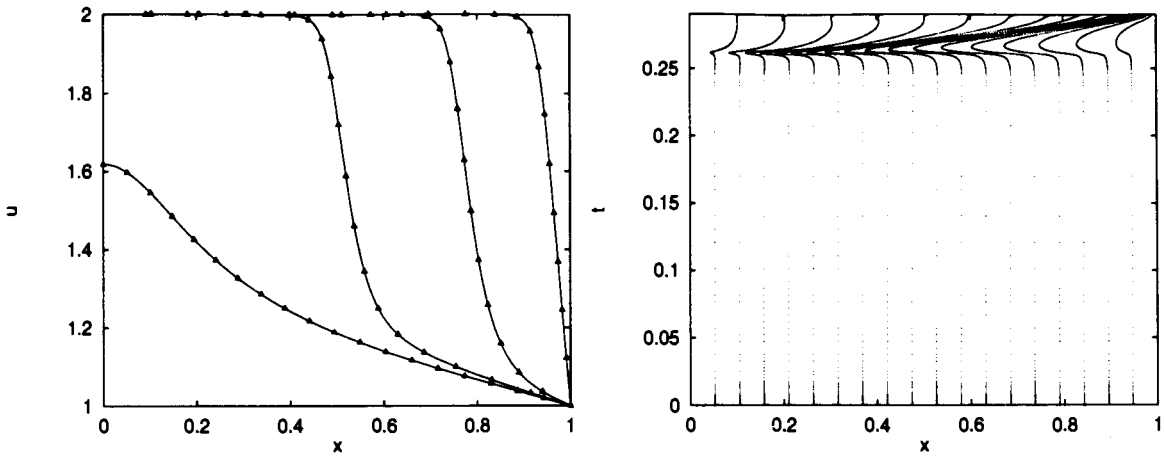


Figure 2.4: MM Solutions at $t = 0.26, 0.27, 0.28, 0.29$ and the mesh trajectories with $n = 20$

it to be $\tau = 10^{-5}$. The spatial smoothing is not too critical as the flame layer is not too thin, and we take $\gamma = 1$.

Figure 2.4 shows the solutions for the above case with $n = 20, \gamma = 1, \tau = 10^{-5}$ which accurately portrays the reference solution. As shown in the plot, the flame layer is not very thin and a fixed mesh with $n = 40$ gives nearly the same accuracy.

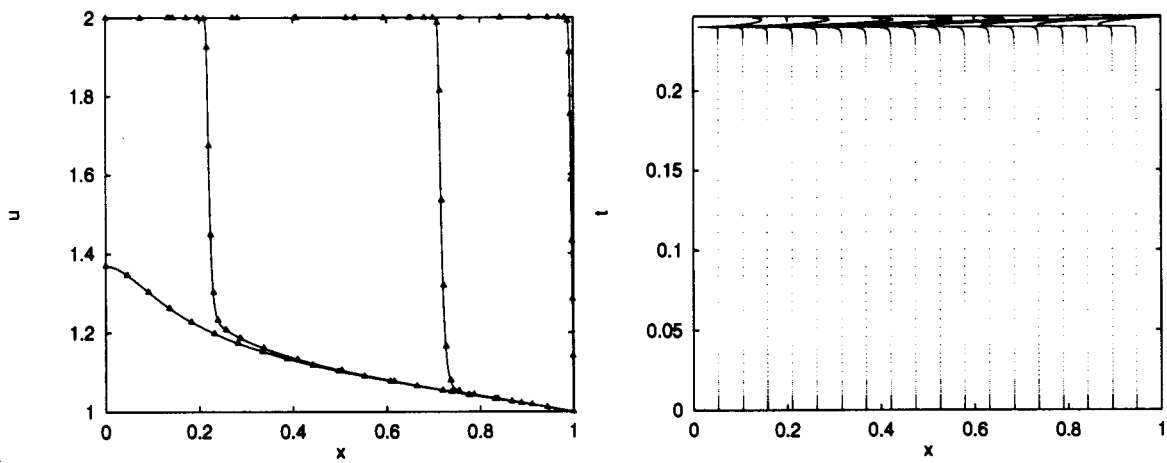


Figure 2.5: MM Solutions at $t = 0.240, 0.241, 0.244, 0.247$ with $n = 20$ and the corresponding mesh trajectories.

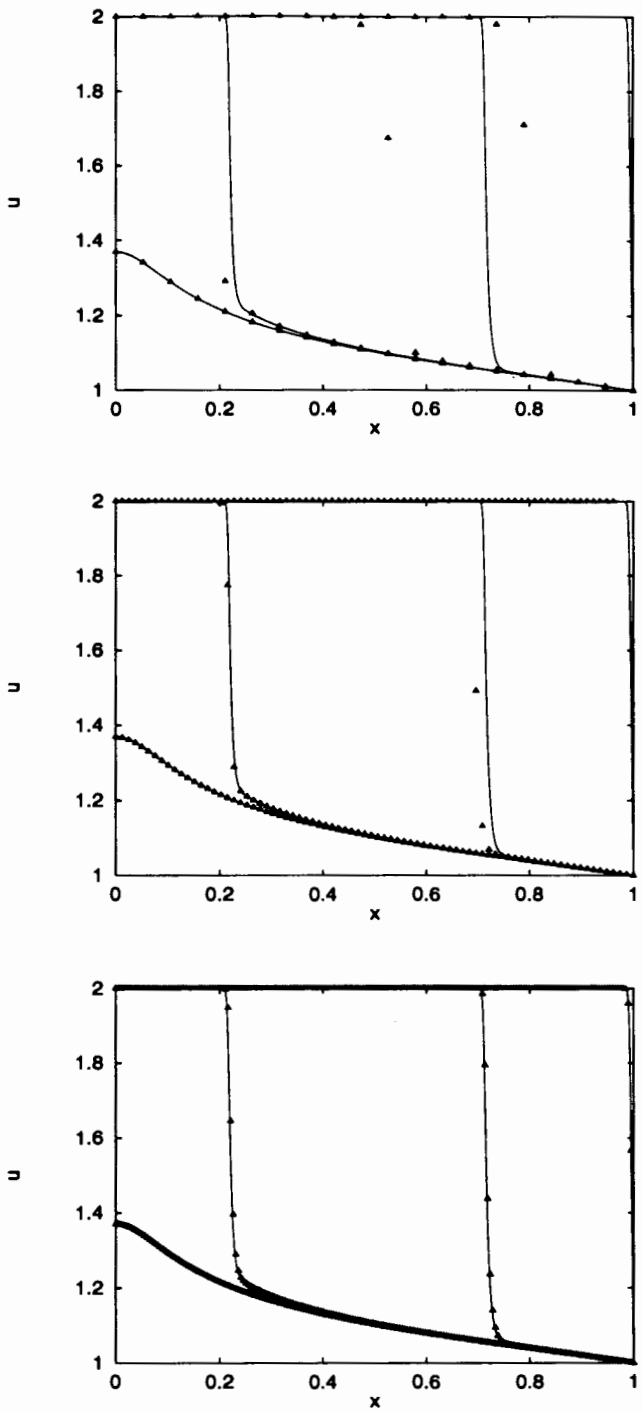


Figure 2.6: FM Solutions at $t = 0.240, 0.241, 0.244, 0.247$ with $n = 20, 80$ and $n = 200$ respectively.

The computational results are compiled in table 2.2.

A more interesting case is when $\delta = 30$; In this case, the flame front is much thinner and higher spatial resolution is required. Here the steady state is reached slightly before $t = 0.247$ which is the end point for our time integration.

The solution is plotted at times $t = 0.240, 0.241, 0.244, 0.247$ following [20]. We choose $\tau = 10^{-5}$, as a higher value results in inaccuracy. The reference solution is obtained in the same manner as in the case with $\delta = 20$. The time tolerances are crucial again, and fail to detect the start of ignition even for $atol = 10^{-5}$. However, lowering it to $atol = 10^{-6}$ accurate results are obtained. We plot our solution with $n = 20, atol = 10^{-6}$ and $\tau = 10^{-5}$ in fig. 2.5.

The fixed mesh with the same tolerance, however, fails (see fig.2.6) to follow the flame propagation, even though it is able to detect the start of ignition accurately. Even an increase of nodes by four times that of moving nodes shows deviation from the reference solution (see fig. 2.6). In order to produce comparable (to that of moving mesh) accuracy with fixed mesh we needed 200 nodes and proportionately more CPU time (see table 2.2).

| $\delta$ | n | | $atol$ | $\tau$ | NTS | JAC | ETF | CPU | $u(0, 0.26)$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | $10^{-4}$ | $10^{-5}$ | 204 | 33 | 21 | 9.41 | 1.7115 |
| | | | $10^{-6}$ | $10^{-1}$ | 770 | 81 | 28 | 25.78 | 1.6177 |
| | | | | $10^{-3}$ | 450 | 45 | 17 | 14.86 | 1.6177 |
| 20 | 20 | MM | | $10^{-5}$ | 449 | 43 | 16 | 15.55 | 1.6178 |
| | | | | $10^{-7}$ | 537 | 47 | 20 | 17.44 | 1.6178 |
| | 20 | FM | $10^{-6}$ | | 750 | 33 | 17 | 15.44 | 1.6177 |
| | 40 | | | | 756 | 31 | 17 | 28.31 | 1.6179 |
| | 200 | MM | $10^{-6}$ | $10^{-3}$ | 902 | 60 | 20 | 216.80 | 1.6182 |
| | 20 | MM | $10^{-6}$ | $10^{-3}$ | 1272 | 267 | 46 | 55.69 | 1.369 |
| | 20 | FM | | | 4288 | 518 | 256 | 99.04 | 1.409 |
| 30 | 80 | | | | 12089 | 1516 | 249 | 1134.8 | 1.370 |
| | 200 | | | | 10185 | 867 | 132 | 2127.5 | 1.371 |

Table 2.2: Computational summary for the Reaction-Diffusion equation

**Example 3 : Allen-Cahn equation**

The last problem that we consider in this chapter is called one-dimensional Ginzburg-Landau or Allen-Cahn equation, given by

$$u_t = \epsilon^2 u_{xx} - f(u), \quad 0 < x < 1, \quad t > 0 \tag{3.3a}$$

where the function $f$ is smooth and satisfies $f(\pm 1) = 0, f'(\pm 1) > 0, \int_{-1}^{1} f(u)du = 0$. A typical example is $f(u) = u^3 - u$, and this is also the function chosen here.

We consider the following boundary and initial conditions:

*Case 1*

$$u = 0, \quad x = 0, 1, \qquad u(x, 0) = sin(7\pi x). \tag{3.3b}$$

*Case 2*

$$u_x = 0, \quad x = 0, 1, \qquad u(x, 0) = cos(7\pi x). \tag{3.3c}$$

The solution for this bistable equation is known to have demonstrated interesting behaviour, when captured by the *attractor*[3] of the dynamical system. The only possible stable equilibrium solution of (3.3a) are constant in space, and are minimizers of the energy functional

$$E(u) = \int_0^1 (F(u) + \frac{1}{2}\epsilon^2 u_x^2)dx$$

where $f(u) = F'(u)$. Thus, for large $t$, typical solutions will be approximately constant in space. However, for $\epsilon$ small, the time taken to reach these patternless asymptotic states can be extremely long. Although a pattern of layers form in a relatively short time, having reached this state, the solution changes exponentially slowly. The rate depends on the physical parameter $\epsilon$, and is approximately of $O(e^{-1/\epsilon})$(See [12]).

[3]The attractor of a dynamical system is generically the union of the set of equlibria and their unstable manifolds (cf. [8] and [18]).

**Case 1:** We choose $\epsilon = 0.05$, and the initial condition as in (3.3b), following [12], for the purpose of comparison. The above equation models the process of grain boundary migration, by which two differently aligned crystal lattices in a solid evolve with time. For $\epsilon$ small, diffusion is negligible on a short time scale, and the initial data evolves towards the stable zeros of $f$, {+1,-1}. Development of interface separating regions (in which $u > 0$, and $u < 0$) occurs, and they propagate to the boundary $x = 0$, at an exponentially slow rate. For this problem, we concern ourselves with steady state solutions only, and stop the time-stepping once a solution reaches a state that appears to be stable. The reference solution is computed using 200 MM nodes, and $atol = rtol = 10^{-8}$, $\tau = 10^{-3}$, $\gamma = 2$, and as before it is plotted with solid lines.

A point worth noting here is that, this slow propagation of the interface sometimes gives rise to spurious steady state solution(s) as stated earlier (see also[12]). The resolution of the numerical method being insufficient to capture the tiny propagation speeds for the transition layers makes these metastable[4] states the steady solutions of the numerical method. With 20 fixed spatial nodes, and 800 grid points in time, their method (an implicit Euler scheme) was held at the spurious steady state, and the true result was obtained by increasing the spatial nodes to 100. Keeping this in mind, we proceed to discuss our test results.

Figure 2.7 shows the solutions at $t = 0$ and FM and MM solutions after they reached a steady state (which was at $t = 40$, for MM), where the results were obtained using $n = 20$, $atol = 10^{-5}$, $rtol = 10^{-6}$. Notice the agreement between the MM and the reference solution even at a low number of grid points $n = 20$.

The FM solutions on the other hand, produces a spurious steady state, and stays there for a very long time. We did further experiments to evaluate the performances of the parameters of MOVCOL. We noticed that for fewer grid points, e.g. $n = 20$ spatial smoothing proved to be at a disadvantage, as it led to a spurious stable solution

[4]The states which evolve at a very slow time scale are known as *metastable states.*

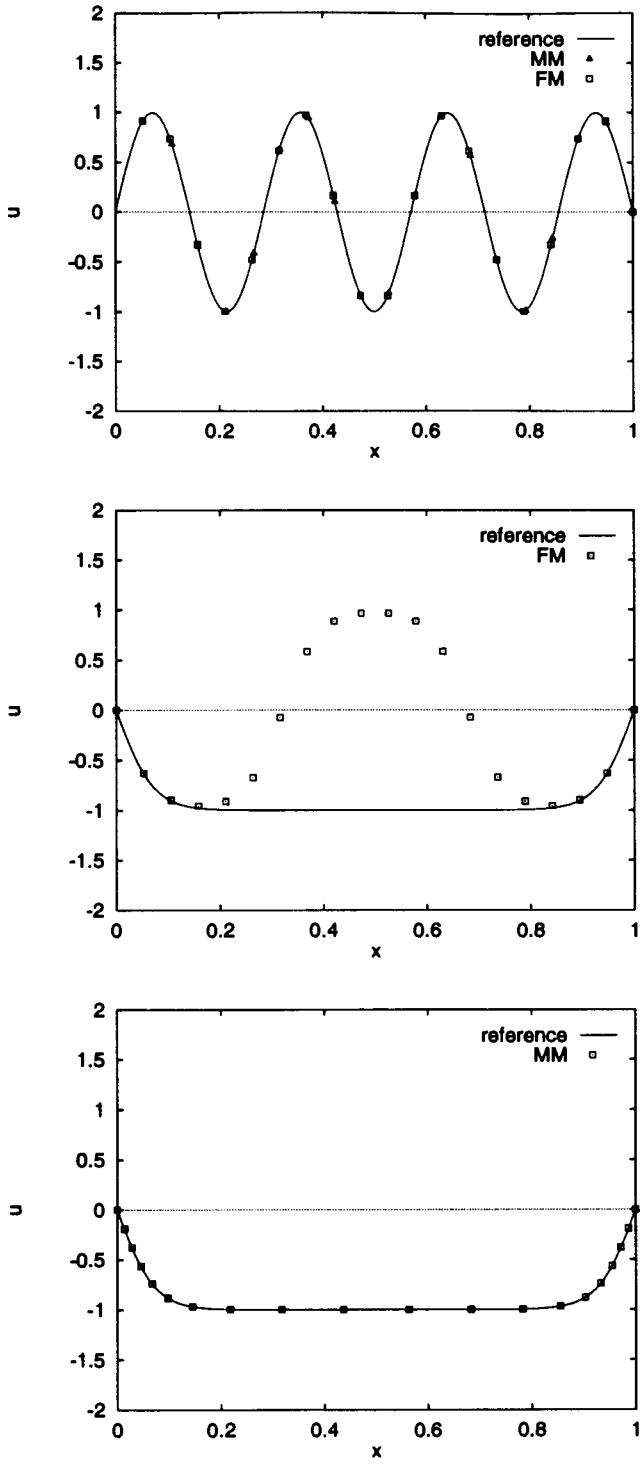


Figure 2.7: Initial Solution; FM solution; MM solution at $t = 40$

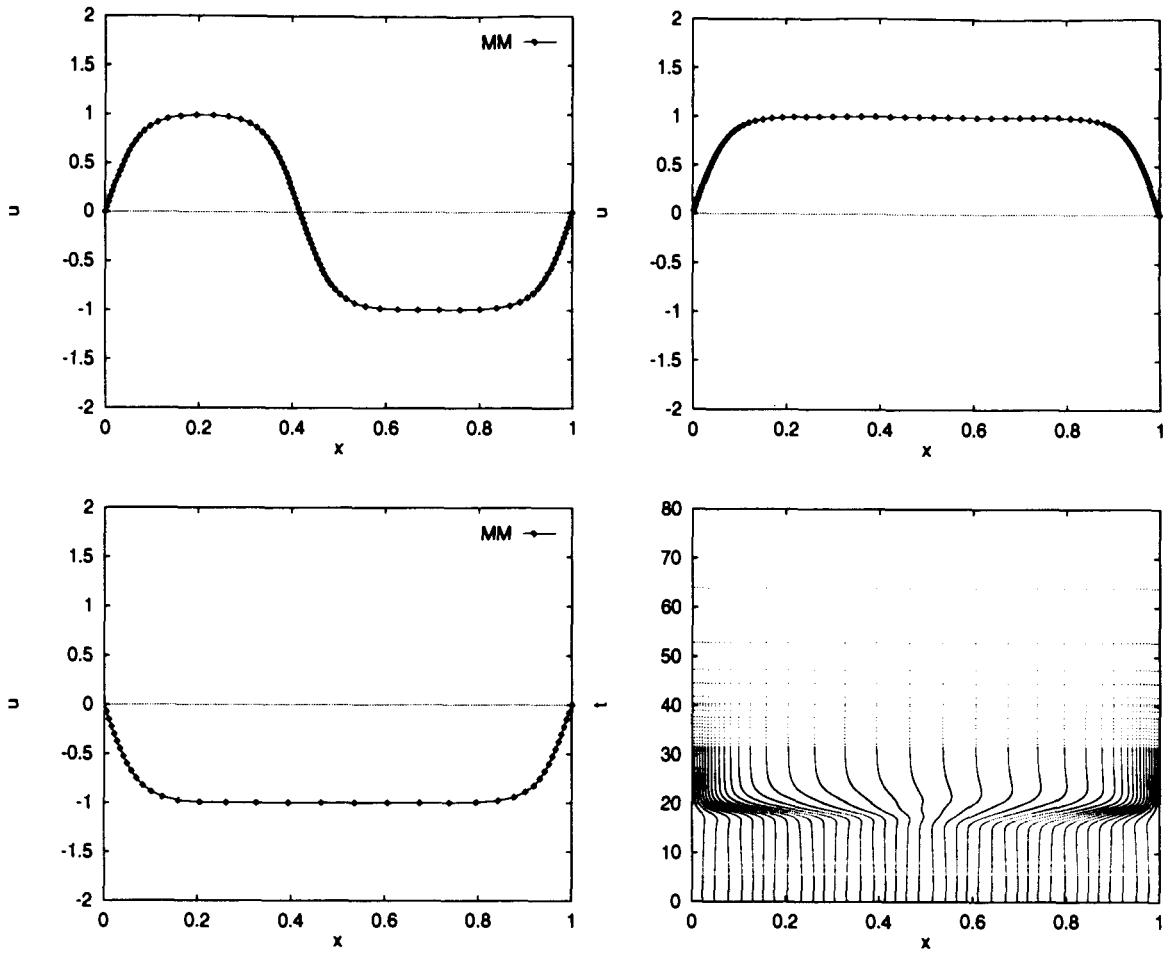


Figure 2.8: MM Solutions for $n = 90$ with $\gamma = 1$ and $\gamma = 2$; for $n = 40$ with $\tau = 1$ and $\gamma = 1$ and the mesh trajectories.

(table 2.3). However, when number of grid points was increased, e.g. to $n = 90$, smoothing helped the solution to get out of the spurious state (fig. 2.8). Although no apparent improvement was found by decreasing $\tau$ from $10^{-3}$ to $10^{-5}$, the solution behaviour deteriorated following an increase to $\tau = 10$. Interestingly enough though a moderately large[5] $\tau(=1)$ did not seem to disrupt the solution behaviour (fig. 2.8). It is worth noting that the solution behaviour was very sensitive to the numerical parameters. The solutions (both MM and FM) switched back and forth between

[5]MOVCOL's recommended range for $\tau$ is $10^{-6} < \tau < 10^{-3}$.

the two true stable solutions $U_1$ and $U_2$, following even the slightest change in the parameters. An example is shown in fig. 2.9, where a change in $\gamma$ from 1 to 2 changed the solution from $U_2$ to $U_1$. This, however, is not unusual for numerical solutions of bistable problems (see [18]).

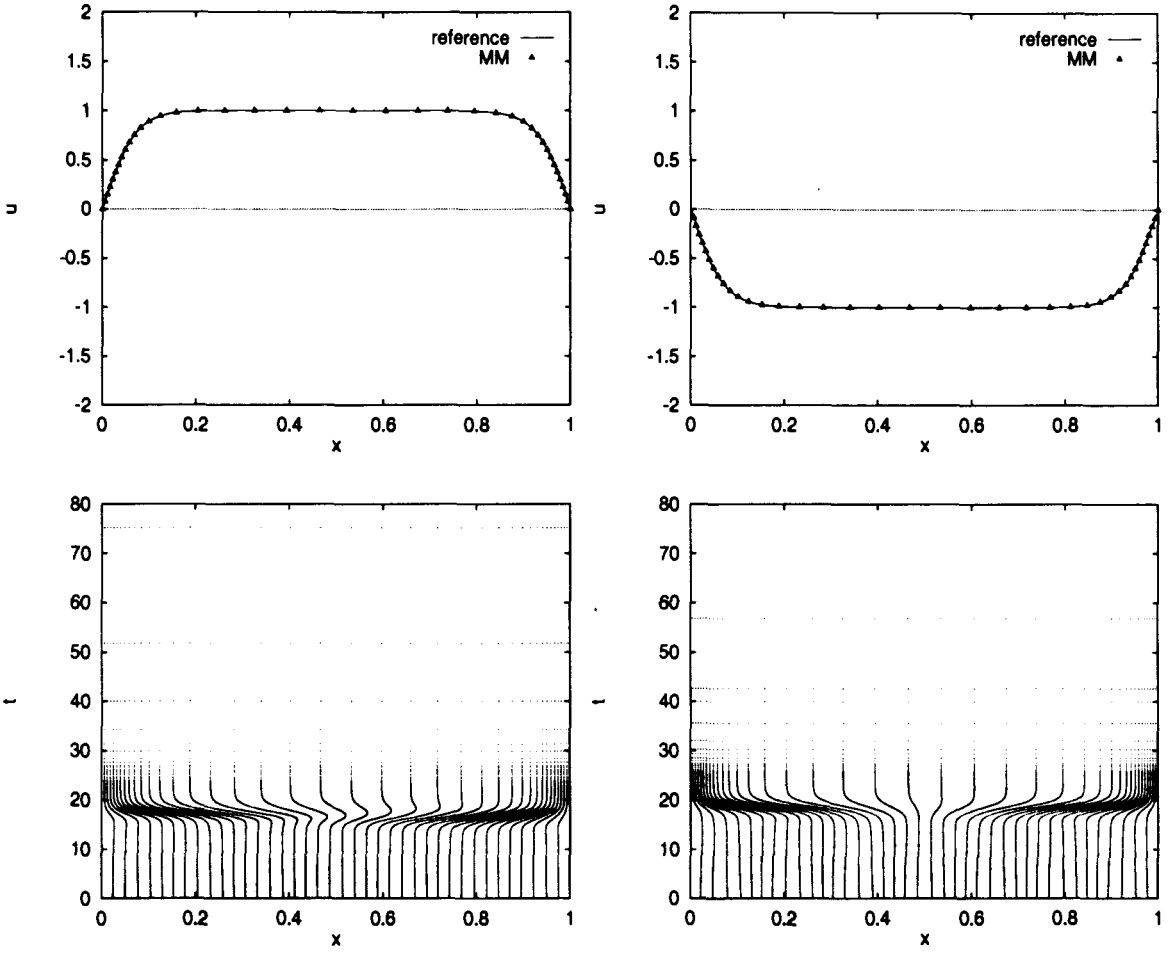


Figure 2.9: MM Solutions at $t = 40$ with $\gamma = 1$ and $\gamma = 2$ and the mesh trajectories.

Although FM produced comparable results to MM after increasing the number of nodes to $n = 40$, it gave spurious solutions for certain values of $n$ (see table 2.3). For MM this problem was remedied with the help of smoothing; FM on the other hand, had no choice but to increase the number of nodes in order to solve this problem.

| n | | $\gamma$ | $\tau$ | NTS | JAC | ETF | CPU | $U$ |
|---|---|---|---|---|---|---|---|---|
| 20 | MM | 1 | $10^{-3}$ | 459 | 50 | 13 | 14.96 | $U_1$ |
| | | 2 | | 591 | 49 | 2 | 16.65 | $U_S$ |
| | | 1 | $10^0$ | 404 | 44 | 10 | 12.01 | $U_1$ |
| | | | $10^1$ | 245 | 30 | 2 | 7.86 | $U_S$ |
| 40 | | | | 388 | 44 | 8 | 25.32 | $U_2$ |
| | | 2 | | 424 | 47 | 10 | 26.48 | $U_1$ |
| 90 | | 1 | $10^{-3}$ | 499 | 54 | 12 | 68.52 | $U_S$ |
| | | 2 | $10^{-3}$ | 398 | 45 | 11 | 59.67 | $U_2$ |
| 20 | FM | | | 207 | 25 | 2 | 4.95 | $U_S$ |
| 40 | | | | 222 | 34 | 3 | 11.34 | $U_1$ |
| 45 | | | | 241 | 28 | 3 | 11.24 | $U_S$ |
| 55 | | | | 226 | 31 | 5 | 14.31 | $U_S$ |

Table 2.3: Computational summary for the Allen-Cahn equation for the Dirichlet boundary condition; with $U_1$, $U_2$ and $U_S$ respectively giving the two true stable and the spurious solutions.

**Case 2:** The Allen-Cahn equation with the homogeneous Neumann boundary condition is our second test problem here. The initial solution is a cosine function in $[0, 1]$, and $\epsilon$ is chosen to be 0.05 as before. Note that the stationary solutions are given by the equation

$$\Phi_{xx} - f(\Phi) = 0$$

where $\Phi(0) = 0$, and $\Phi(x) \to \pm 1$ as $x \to \pm\infty$. The solution to the above equation is $\Phi(x) = \tanh(\frac{x}{\sqrt{2\epsilon}})$ for $f(u) = u^3 - u$. According to [8], the metastable states are approximated by appropriate translates or reflections of $\Phi$, the unstable stationary solution. That is, for one layer state with layer length $h$, the metastable states are given by $\Phi(x - h)$ or $\Phi(h - x)$.

The first solution that we compute is with $n = 20$, and all other parameters as in the first case except for $\tau$, which we choose to be $10^{-5}$ here. Both the FM and MM methods produces spurious steady state solutions at this point.

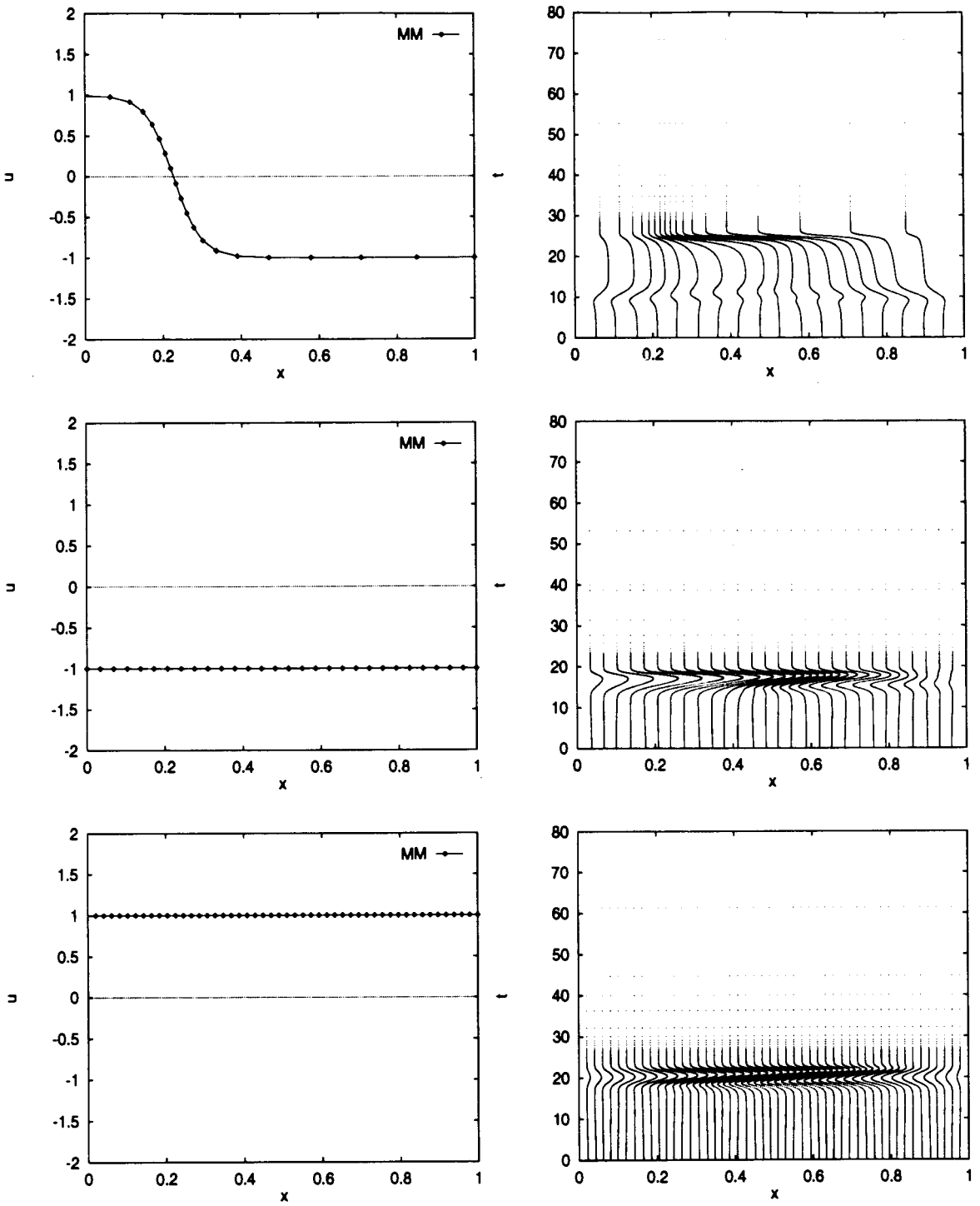


Figure 2.10: MM solutions with $n = 20$, $n = 30$, $n = 50$, and the corresponding mesh trajectories.

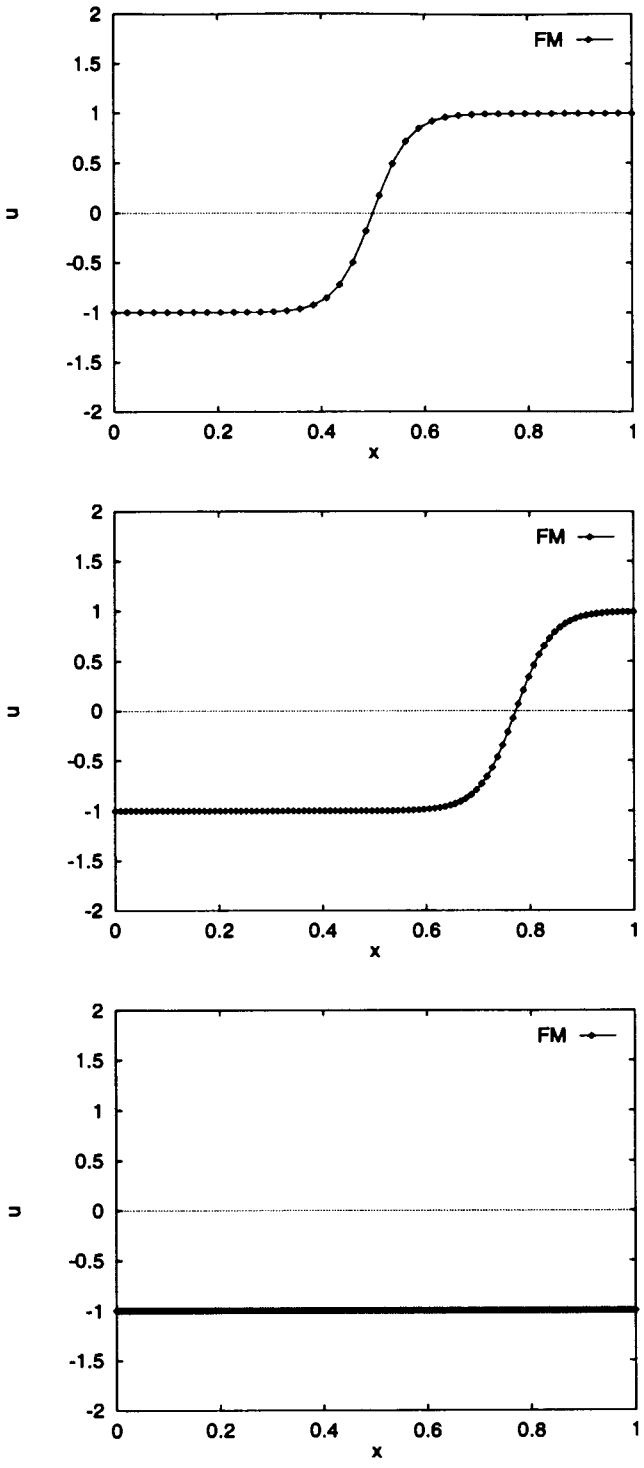


Figure 2.11: FM Solutions with $n = 40$, $n = 100$, $n = 200$ respectively.

Notice that the spurious MM solution here is a translated reflection of the stationary *tanh* solution $\Phi(x)$ as claimed in [8]. However, an increase in the nodepoints for MM gives the accurate solution, whereas FM even with five times that many points (fig. 2.11) fails to produce the desired result. Figure 2.10 shows MM solutions and the corresponding mesh trajectories for $n = 20$, $n = 30$ and $n = 50$ respectively. Our experiments with the MOVCOL parameters shows the same sensitive nature of the solution as before. A point worth noting is that spatial smoothing creates an adverse effect even at $n = 40$, unlike before, and makes no apparent improvement for larger $n$.

We tested case 2 further, with a smaller $\epsilon$(=0.04) in order to observe its effect on the propagation rate of the solution layers. As mentioned earlier, when $\epsilon$ decreases, so does the rate of propagation, being on a time scale of $O(e^{-1/\epsilon})$ (see [3], [8], and [12]). Our experiments agreed, with their claims, as the solution took considerably more time to reach the steady-state. Also, DDASSL took a lot more timesteps to converge (see table 2.5) than it did in the earlier case with $\epsilon = 0.05$. Other results were however similar to those for larger $\epsilon$.

| n | | $\gamma$ | $\tau$ | NTS | JAC | ETF | CPU | $U$ |
|---|---|---|---|---|---|---|---|---|
| 20 | MM | 1 | $10^{-5}$ | 860 | 75 | 13 | 25.96 | $U_S$ |
| | | 2 | | 424 | 40 | 5 | 10.7 | $U_S$ |
| 30 | | 1 | | 496 | 56 | 7 | 23.71 | $U_1$ |
| 50 | | | | 490 | 49 | 5 | 35.67 | $U_1$ |
| | | 2 | | 506 | 47 | 8 | 36.37 | $U_2$ |
| 20 | FM | | | 212 | 22 | 3 | 4.65 | $U_S$ |
| 40 | | | | 221 | 34 | 3 | 10.6 | $U_S$ |
| 80 | | | | 259 | 34 | 9 | 24.27 | $U_S$ |
| 100 | | | | 274 | 36 | 11 | 30.88 | $U_S$ |
| 200 | | | | 235 | 32 | 7 | 55.15 | $U_1$ |

Table 2.4: Computational summary for the Allen-Cahn equation with the homogeneous Neumann boundary condition and $\epsilon = 0.05$

| n | | $\gamma$ | $\tau$ | NTS | JAC | ETF | CPU | $U$ |
|---|---|---|---|---|---|---|---|---|
| 30 | MM | 2 | $10^{-5}$ | 1036 | 86 | 27 | 38.75 | $U_1$ |
| 50 | | 1 | | 1798 | 151 | 44 | 110.34 | $U_1$ |

Table 2.5: Integration history: A few results for the case when $\epsilon = 0.04$

## 2.4 Summary

In this chapter, we introduced MOVCOL, the method of implementation here, and discussed it in detail. We then simulated three problems with very different solution behaviour, with the help of MOVCOL, and presented our results, along with any difficulties that we encountered. For the first two problems, when the solutions reached their steady-state rather fast, the adaptive meshing proved its need beyond a shred of doubt, whereas for the Allen-Cahn problem, with an exponentially slow rate of solution propagation, the need of a moving mesh did not seem to have justified itself well enough in the first case, where a carefully chosen fixed mesh could produce results equally accurately and efficiently even with its drawbacks (e.g. spurious solution for certain nodepoints). However, MM exhibited superior performance in the case with the Neumann boundary condition, with the fixed mesh taking more than five times as many nodepoints to produce the same effect as that of the moving mesh. Hence we conclude this chapter with rather mixed views about the performance of the code for such problems, and move on to the next chapter to investigate a phase separation model.

# Chapter 3

# A Phase Separation Model

## 3.1 Introduction

In the process known as spinodal decomposition, a homogeneous alloy spontaneously separates into a fine grained mixture of nearly pure phases of metals. When the alloy is separated, some regions of space are rich in one phase and correspondingly poor in the others. The Cahn-Hilliard equation has been proposed as a model of spinodal decomposition for a binary alloy. In this chapter, we study this model in detail and simulate it with the help of MOVCOL. The numerical results are presented and discussed.

## 3.2 Phase separation in a binary alloy and the Cahn-Hilliard equation

Consider a binary alloy, comprising of species $X_A$ and $X_B$, existing in a state of isothermal equilibrium at a temperature $T_m$, greater than the critical temperature $T_c$. The alloy's composition is spatially uniform with the concentration $u(x,t)$ of

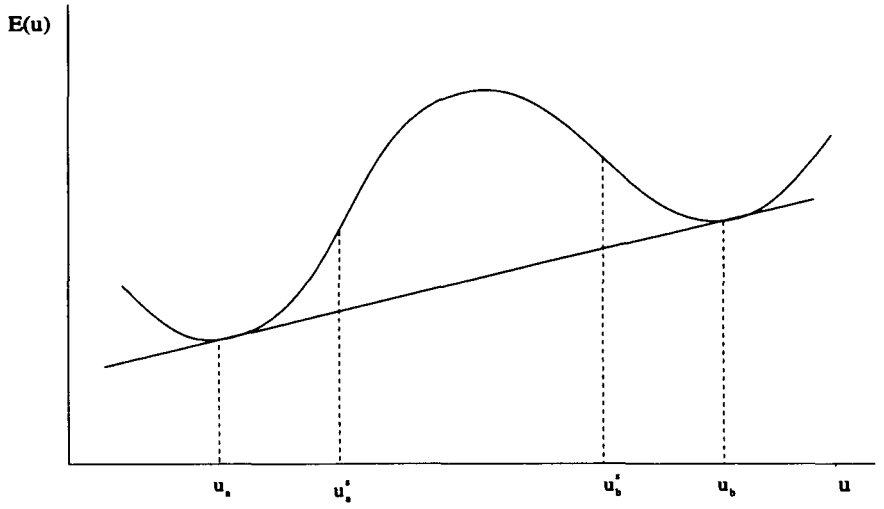


Figure 3.1: Free Energy

$X_B$, which takes the constant value $u_m$. Suppose that the alloy is now suddenly quenched to a uniform temperature $T_m$ below $T_c$. Phase separation takes place in which the composition of the alloy changes from the uniform mixed state to that of a spatially separated two phase structure, each phase being characterized by a different concentration value which is either $u_a$ or $u_b$.

A theory describing this phenomenon comes from considering the Gibb's free energy $\psi(x,t)$, which has the following properties

$$\psi_{uu}(u,t) > 0, \quad T > T_c \text{ and } \psi_{uu}(u,t) < 0, \quad T < T_c \tag{3.1}$$

for $u \in [u_a^s, u_b^s]$. This energy has the double-well form for $T < T_c$, as shown in Fig. 3.1.

Here $u_a^s$ and $u_b^s$ are the spinodal points. Close to the two local minima are the binodal values $u_a$ and $u_b$, which are the two unique points where the supporting tangent touches the curve. The spinodal region is unstable, states to the left and right of the binodal points are stable, and the remaining intervals are metastable.

Suppose that the mixture has been prepared to have an initial state with a spatial

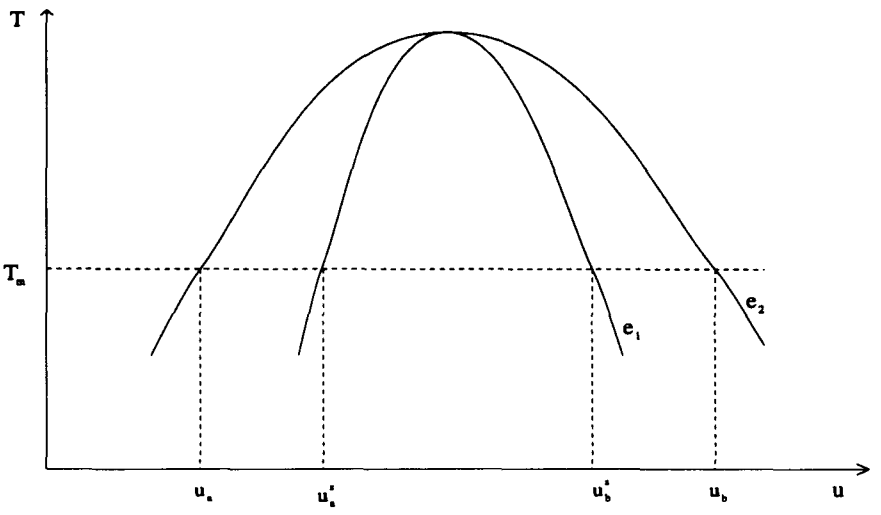


Figure 3.2: Phase diagram

composition taking values in the spinodal interval. The mixture will evolve from this unstable nonequilibrium state to an equilibrium configuration consisting of two coexisting phases with a spatial pattern composed of *grains* rich in either $X_A$ or in $X_B$. Such an evolution process is called *phase separation*, and when it takes place in spinodal region, it is called spinodal decomposition. Associated with this description is the phase diagram depicted in figure 3.2. Here, $e_1$ and $e_2$ represent the spinodal[1] and the coexistence[2] curves respectively. Below $e_1$, the state $(u_m, T_m)$ is unstable and the alloy separates into two phases characterized by the values $u_a$ and $u_b$ where the line $T = T_m$ crosses $e_2$.

It can be shown that the phase diagram in figure 3.2 can be explained by thermodynamics in terms of the minimization of the free energy $\psi(u)$ [11]. We now want to model the evolution of an alloy, initially in equilibrium in the uniform state $(u_m, T_m)$ with $T_m > T_c$ which is then quenched to a state where $T_m < T_c$ and $u_m$ lies in the spinodal region. The new state is not in equilibrium. Here we assume that the system

[1]Gives the locus of points where $\psi_{uu}(u, T) = 0$.

[2]Above this curve any uniform concentration $(u_m, T_m)$ is stable.

is isothermal. The mass flux is given by

$$J = -M\nabla\mu \tag{3.2}$$

where $M > 0$ denotes the *mobility* and $\mu = \psi'(u)$ is the *chemical potential* difference between species A and B. Upon substitution of $\mu = \psi'(u)$ in (3.2) we get

$$J = -M\psi''(u)\nabla u \tag{3.3}$$

which along with the mass balance law

$$\frac{d}{dt}\int_{\Re} u = -\int_{\partial\Re} \mathbf{J}.\mathbf{n} \text{ for } \Re \subset \Omega \tag{3.4}$$

yields the diffusion equation

$$\frac{\partial u}{\partial t} = \nabla(K(u)\nabla u) \tag{3.5}$$

where the *diffusivity* $K(u) = M\psi''(u)$.

In order to model the surface energy of the interface separating the phases Cahn and Hilliard [6] modified the free energy $\psi(u)$ by adding the gradient term $\frac{\gamma}{2}|\nabla u|^2$ so that the free energy becomes

$$\Psi = \psi(u) + \frac{\gamma}{2}|\nabla u|^2. \tag{3.6}$$

Here, $\gamma(> 0)$ is the parameter determining the *interaction length* for the gradient energy term, and $\psi(.)$ is the homogeneous free energy. The Cahn-Hilliard model for the equilibrium description of phase separation is thus to

$$\text{minimize} \quad \int_\Omega \{\psi(u(x)) + \frac{\gamma}{2}|\nabla u|^2\}dx, \text{ subject to } \int_\Omega u(x)dx = u_m|\Omega|. \tag{3.7}$$

The generalized chemical potential $\sigma = \psi'(u) - \gamma\Delta u$ is introduced such that $\sigma$ is the functional derivative of the energy

$$E(u) = \int_\Omega \{\psi(u) + \frac{\gamma}{2}|\nabla u|^2\}dx, \tag{3.8}$$

i.e.,

$$< \sigma, v >=< E'(u), v >= (\psi'(u), v) + \gamma(\nabla u, \nabla v). \tag{3.9}$$

Here $(.,.)$ denotes the $L^2(\Omega)$ inner-product. But the mass flux $J$ also satisfies

$$J = -\nabla\sigma. \tag{3.10}$$

Combining the above equation with (3.5) we get the generalized diffusion equation for this non-equilibrium gradient theory of phase separation

$$u_t = -\Delta(\gamma\Delta u - \phi(u)) \tag{3.11a}$$

where $\phi(u) = \psi'(u)$.

This fourth order, nonlinear evolution equation is called the Cahn-Hilliard equation. For a closed system it is supplemented by the following boundary conditions, the first of which is the zero mass flux condition and the latter is the natural boundary condition associated with the energy functional:

$$\nabla(\psi'(u) - \gamma\nabla^2(u)) = 0, \quad \gamma\nabla u = 0, \quad x = 0, L. \tag{3.11b}$$

The initial condition is given by

$$u(x,0) = u_0(x), \qquad \int_0^L u_0(x)dx = ML. \tag{3.11c}$$

where $M$ gives the *mean value* of the initial concentration $u_0(x)$.

### 3.2.1 Some Continuum Properties

This initial boundary value problem was studied by Elliot and Zheng [15], and the existence and uniqueness of a solution $u(t)$ of (3.11) were proved under certain conditions. They also showed that under these conditions, for any initial data, the solution $u(t)$ converges as $t \to \infty$ to a solution of the steady-state problem

$$\gamma u_{xx} = \phi(u) - \sigma, \quad 0 < x < L \qquad \text{and } \sigma \in \Re, \tag{3.12a}$$

(3.12b) $$u_x = 0, \quad x = 0, L \quad \text{and} \quad \int_0^L u_0(x)dx = ML,$$

solutions of which are associated with the local minimizers of the energy functional $E(u)$. These solutions are periodic[3] and monotone[4] for sufficiently small $\gamma$, and all non-monotone solutions are saddle points of $E(u)$. The last statement ensures that for every non-monotone solution $\bar{u}$ of (3.12), there is an initial value $u_0$ lying in any $H^1$ neighbourhood of $\bar{u}$, for which the solution of the dynamic problem (3.11) will stay bounded away from $\bar{u}$. It is also shown, that phase separation does not take place if the initial concentration $u_0(x)$ lies in the stable region, i.e., outside the interval $[u_a, u_b]$ (see fig. 3.1). Moreover, in such cases, the solution $u_0(x)$ tends to the uniform state $u = M$, where the mean value $M$ of the initial solution is given by (3.11c).

The simplest form of $\psi$ having the double-well potential is

(3.13) $$\psi(u) = \frac{1}{4}\gamma_2 u^4 + \frac{1}{3}\gamma_1 u^3 + \frac{1}{2}\gamma_0 u^2,$$

where $\gamma_0$, $\gamma_1$, and $\gamma_2$ are constants with $\gamma_2 > 0$. Note that $\psi''(u)$ has two real roots, namely the spinodal values $u_a^s$ and $u_b^s$.

Following Elliot and French [13], we choose the interval $[0, 6]$ and the energy $\psi(u)$ to be

(3.14) $$\psi(u) = \frac{1}{12}u^4 - \frac{1}{2}u^2.$$

Note that it implies $\phi(u) = \frac{1}{3}u^3 - u$. For (3.14), the spinodal points are

$$u_a^s = -1, \qquad u_b^s = 1$$

and the binodal points are

$$u_a = -\sqrt{3}, \qquad u_a = \sqrt{3},$$

[3]It means that all lengths along the x-axis of transition, from peak to trough or vice versa, are equal (see [7]).

[4]If $u_\gamma(x)$ is a unique minimizer of the energy functional, so is $u_\gamma(-x)$ (see [7]).

It has been shown in [13] that these values define piecewise constant functions that minimize the energy functional $E(u)$. Note that $\psi$ is symmetric about $u = 0$, and this implies that the binodal points correspond to the absolute minima of $\psi$, i.e. $\psi_m = \psi(u_a) = \psi(u_b)$.

### 3.2.2 Implementation Strategy

In order to implement MOVCOL for this problem, we transform this fourth order in space PDE into a system of two PDEs as follows:

Call $v := \gamma\Delta u - \phi(u)$. Then (3.11) can be written as

$$\text{(3.15}a\text{)} \qquad \begin{cases} v_t + (u^2 - 1)u_t = \gamma u_{xxt} \\ u_t = -v_{xx} \end{cases}$$

with boundary conditions

$$\text{(3.15}b\text{)} \qquad v_x = u_x = 0, \text{ at } \quad x = 0, 6.$$

We then simulate (3.15) with various initial conditions following [13], each giving rise to a different solution behaviour, and present our results in the next section. The numerical method used in [13] is a combination of the Galerkin method for discretization in space and an implicit midpoint rule for time discretization. One concern that the authors Elliot and French expressed was about the accuracy of the numerical solution. In spite of the optimal convergence (see [13]) of their method, it does not guarantee accuracy unless the spatial stepsize is very small. The loss in accuracy comes from a constant determining the error bounds which is rather large due to its dependence on $\gamma$[5] and the derivative of the solution $u$. Note that, $u$ is expected to have large derivatives in $x$, as the separation patterns form sharp interfaces.

[5]This constant is directly proportional to $\gamma^{-\frac{1}{2}}$, and $\gamma$ is small.

The smoothed MMPDE[6] that we use, is nonlinear in $\dot{x}$, whereas successful computation of $Y$ and $\dot{Y}$ by DDASSL[7] requires that the monitor function (or the initial physical solution) be smoother in the computational coordinates. We have encountered cases where DDASSL altogether failed to even start the time integration for a steep initial solution. An initial mesh equidistributed with respect to the initial solution is created before starting the time integration to reduce this difficulty (see [22]). This is done by solving internally the MMPDE and the PDE $u_t = U(x), \quad 0 < t < 1$, where $U(x)$ is the user prescribed initial approximation of $u(x)$. The reference solutions for all the cases are computed with MM using 200 nodes and $atol = rtol = 10^{-8}$ unless mentioned otherwise, and as before, are shown with solid lines.

## 3.3 Numerical Results

**Case 1:** For the first case, we choose a ninth-degree polynomial as our initial solution, given by

$$u_0(x) = Ap(x) + M, \quad \text{where} \quad p(x) = x^4(x-6)^4(x-2.2) + M_0$$

Here, $M = 3$, $M_0$ is picked such that $\int_0^6 p(x)dx = 0$, and $A$ is chosen so as to make the result $\| u_0 - M \|_\infty \simeq 2.7$ true. The parameter $\gamma$ is set equal to 0.005, following [13].

Notice that the mean value of the initial concentration $u_0(x)$ lies in the stable region $(-\infty, -\sqrt{3}) \cup (\sqrt{3}, \infty)$, and we expect that the solution will tend to its mean value $M$. We plot the solutions at times $t = 0, .5, 1.5, 2$, and 2.7, using MM with 20 nodes, $atol = 10^{-4}$, $rtol = 10^{-5}$, $\gamma = 1$ and $\tau = 10^{-5}$. Figure 3.3 show the concentration $u$ and the mesh trajectories for this experiment. As expected, the solution $u(x,t)$ converges to the constant function $M$. The solution for this case is not

[6]The MMPDE given by eqn. (2.11) in chapter 1.
[7]As defined in the eqn. (1.15) in chapter 1.

steep and a FM works as well with 20 node points. In fact with $\gamma = 1$, FM shows less error than MM. This is not surprising, as MM puts more points in the transitional area of the steep initial solution, and therefore fails to produce an uniform mesh when needed, whereas FM, having started with an uniform mesh, produces better resolution for this uniform solution. This is also clear from the movement of the mesh. The transition from steep initial solution to almost uniform constant solution takes place at around $t = 0.5$, and after that the need for mesh adaptation dies. No phase separation takes place with this initial condition as predicted, which justifies the comparatively less effective performance of an MM here.

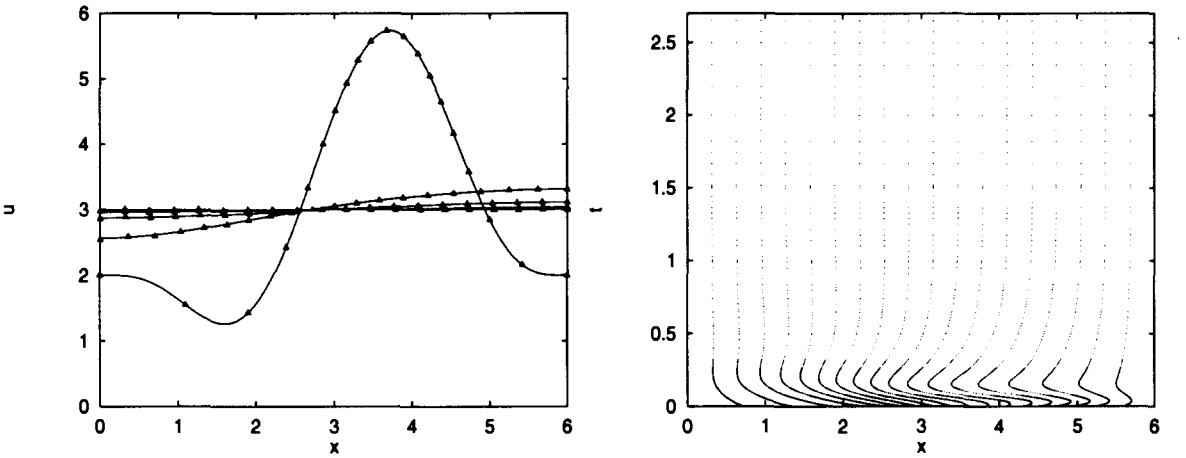


Figure 3.3: Solutions at $t = 0, 0.5, 1, 1.5, 2, 2.7$, and Mesh trajectories for MM method with $n = 20$ for case 1.

**Case 2:** We choose for our second experiment an initial concentration with mean value in the spinodal region, to simulate a phase separation. It is a cosine function and is given by

$$u_0(x) = cos(\pi x/6)$$

in the interval $[0, 6]$. Notice that the mean value here is 0, which lies in the spinodal region $(-1, 1)$. The parameter $\gamma$ is chosen to be 0.02. The MM solution is computed

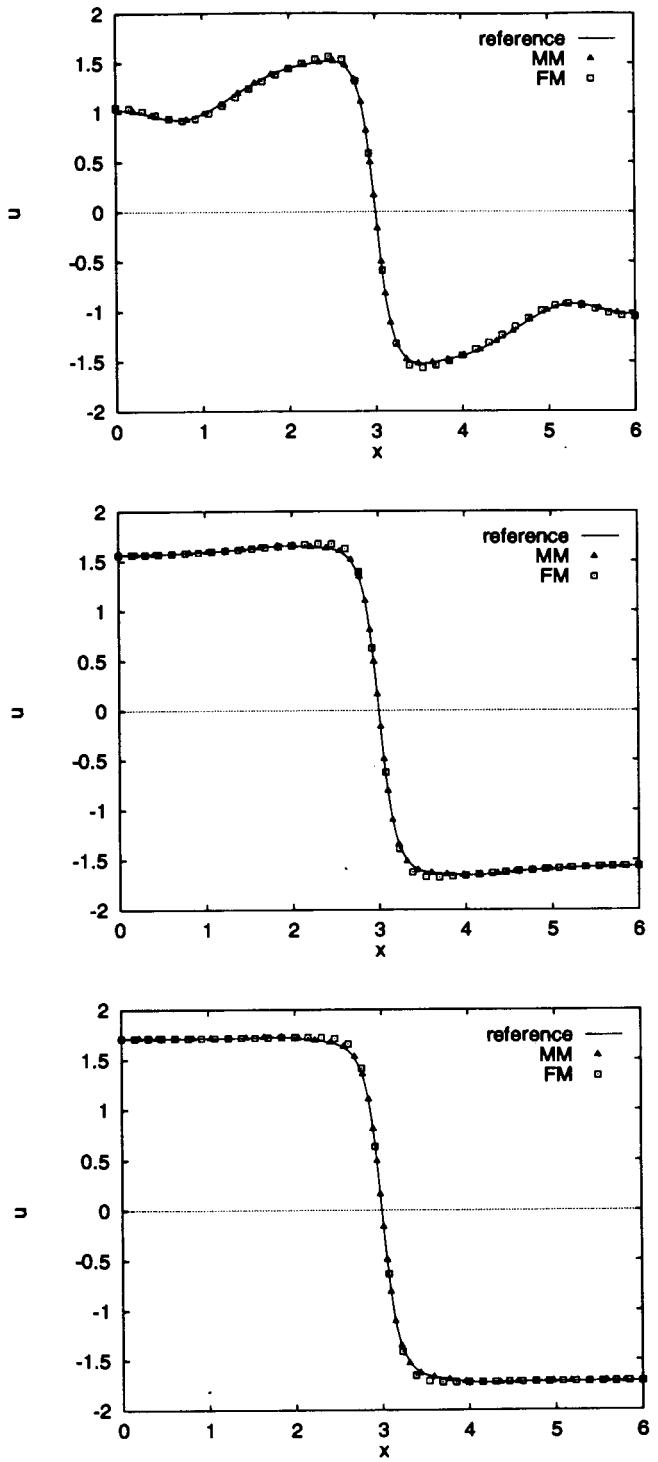


Figure 3.4: Solutions at t=2, t=6, t=10 for case 2.

with 40 node points, and is shown at $t = 2$, $t = 6$, and $t = 10$, in fig. 3.4, along with the reference solution and a FM comparison.

Notice that at $t = 2$ the FM solution shows deviation from the reference solution at the peak as well as at the dip, which it retained even at $t = 10$. Also, FM failed to resolve the steepness in the solution accurately. This is clearly reflected in the graph as well as in the error analysis (shown in table 3.1). The error for FM is maximum at $t = 2$, as the transition has already started to take place by that time, whereas for MM it is the minimum. The error for FM reduces as the meshpoints are doubled, but it still lags behind the MM solution as is clear from table 3.1. We have plotted the minimal spacing $H(t)$ against the output times in fig. (3.5), where

$$H(t) = min(x_{i+1}(t) - x_i(t)), \qquad i = 0, ..., n-1,$$

for the MM approximation. This shows the jump in the stepsize by the moving mesh, to adequately resolve the steep gradient. Figure 3.5 also shows the mesh trajectories for the MM method.

We ran some experiments, to find out the solution sensitivity towards the numer-

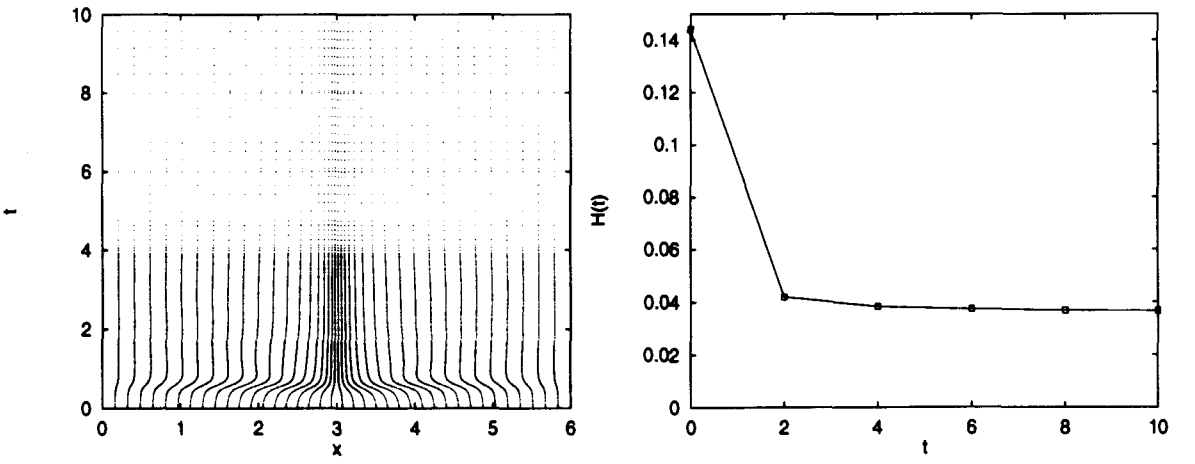


Figure 3.5: Mesh trajectories and the function $H(t)$ for MM method with $n = 40$

| | | MM | | FM | |
|---|---|---|---|---|---|
| n | t | $\|\cdot\|_2$ | $\|\cdot\|_\infty$ | $\|\cdot\|_2$ | $\|\cdot\|_\infty$ |
| 20 | 2 | 2.5E-2 | 3.5E-2 | 7.0E-2 | 5.9E-2 |
| | 4 | 2.7E-2 | 4.1E-2 | 6.4E-2 | 6.7E-2 |
| | 6 | 2.8E-2 | 4.4E-2 | 6.7E-2 | 7.0E-2 |
| | 8 | 2.9E-2 | 4.4E-2 | 6.8E-2 | 7.2E-2 |
| | 10 | 2.9E-2 | 4.5E-2 | 6.8E-2 | 7.3E-2 |
| 40 | 2 | 1.0E-2 | 1.4E-2 | 6.2E-2 | 5.1E-2 |
| | 4 | 1.1E-2 | 1.7E-2 | 5.5E-2 | 6.1E-2 |
| | 6 | 1.2E-2 | 1.9E-2 | 5.7E-2 | 6.5E-2 |
| | 8 | 1.3E-2 | 2.1E-2 | 5.8E-2 | 6.6E-2 |
| | 10 | 1.4E-2 | 2.2E-2 | 5.8E-2 | 6.7E-2 |

Table 3.1: Error history for case 2

-ical parameters. Both temporal and spatial smoothing proved to be less helpful than they were for some of our other test problems. In fact, the solution hardly showed any change when the temporal smoothing parameter $\tau$ was increased to 1. Although this resembles the results found for the Allen-Cahn problem, it is very unlike the case of our reaction-diffusion problem, where, following a similar increase in $\tau$, the mesh movement almost stopped causing a very slow solution propagation.

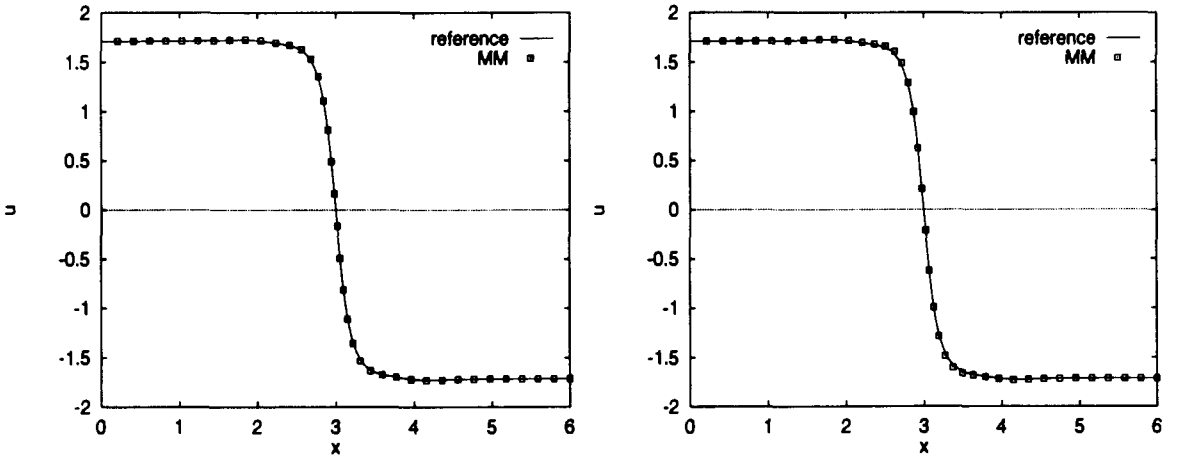


Figure 3.6: MM solutions at $t = 10$ with $n = 40$ and $\gamma = 1$ and $\gamma = 2$ respectively.

We attribute this result to the particular nature of this problem. Lack of sudden change in solution behaviour, and presence of only one interface layer decreases the need for temporal smoothing. Spatial smoothing did not seem to enhance the accuracy of the solution, although it converged faster and consumed less CPU time. At $n = 40$, the solution with $\gamma = 2$ deviated (at the peak as well as at the dip) from the reference solution more, than it did with $\gamma = 1$(fig. 3.6). Table 3.2 shows the computational summary for this case.

| n | | $\gamma$ | $\tau$ | NTS | JAC | ETF | CPU |
|---|---|---|---|---|---|---|---|
| 20 | | 1 | $10^{-5}$ | 189 | 22 | 0 | 13.46 |
| | | 2 | | 177 | 21 | 0 | 12.77 |
| | | 1 | $10^{-3}$ | 194 | 22 | 0 | 13.04 |
| | | | 1 | 182 | 22 | 0 | 13.17 |
| 40 | MM | | | 173 | 22 | 0 | 24.93 |
| | | | 10 | 159 | 19 | 0 | 23.93 |
| | | 1 | $10^{-5}$ | 202 | 23 | 0 | 27.87 |
| | | 2 | | 176 | 24 | 0 | 26.92 |
| 100 | | 1 | | 216 | 28 | 1 | 78.28 |
| | | 2 | | 189 | 26 | 1 | 74.85 |
| 20 | | | | 154 | 18 | 0 | 7.89 |
| 40 | FM | | | 144 | 19 | 0 | 16.02 |
| 100 | | | | 161 | 20 | 0 | 42.81 |
| 200 | | | | 160 | 19 | 0 | 84.34 |

Table 3.2: Computational Summary for case 2

**Case 3:** For our third and the last test case we choose an initial solution which seemed to have difficulty reaching the true steady-state (see [13]). It is of the same form $u_0(x) = Ap(x) + M$ as in case 3, but with $M = 0$ and $A$ set so that $\| u_0 \|_\infty \simeq 0.6$. The value of the physical parameter $\gamma$ was chosen to be 0.07.

Our first computation for this problem was done using a FM with $n = 200$, $atol = 10^{-2}$ and $rtol = 10^{-3}$. The results we thus obtained are shown in fig. 3.7 at times $t = 0, 4, 40, 200, 300,$ and 400, respectively. The solution evolved from layer formation following the phase separation, and then through eventual layer collapses

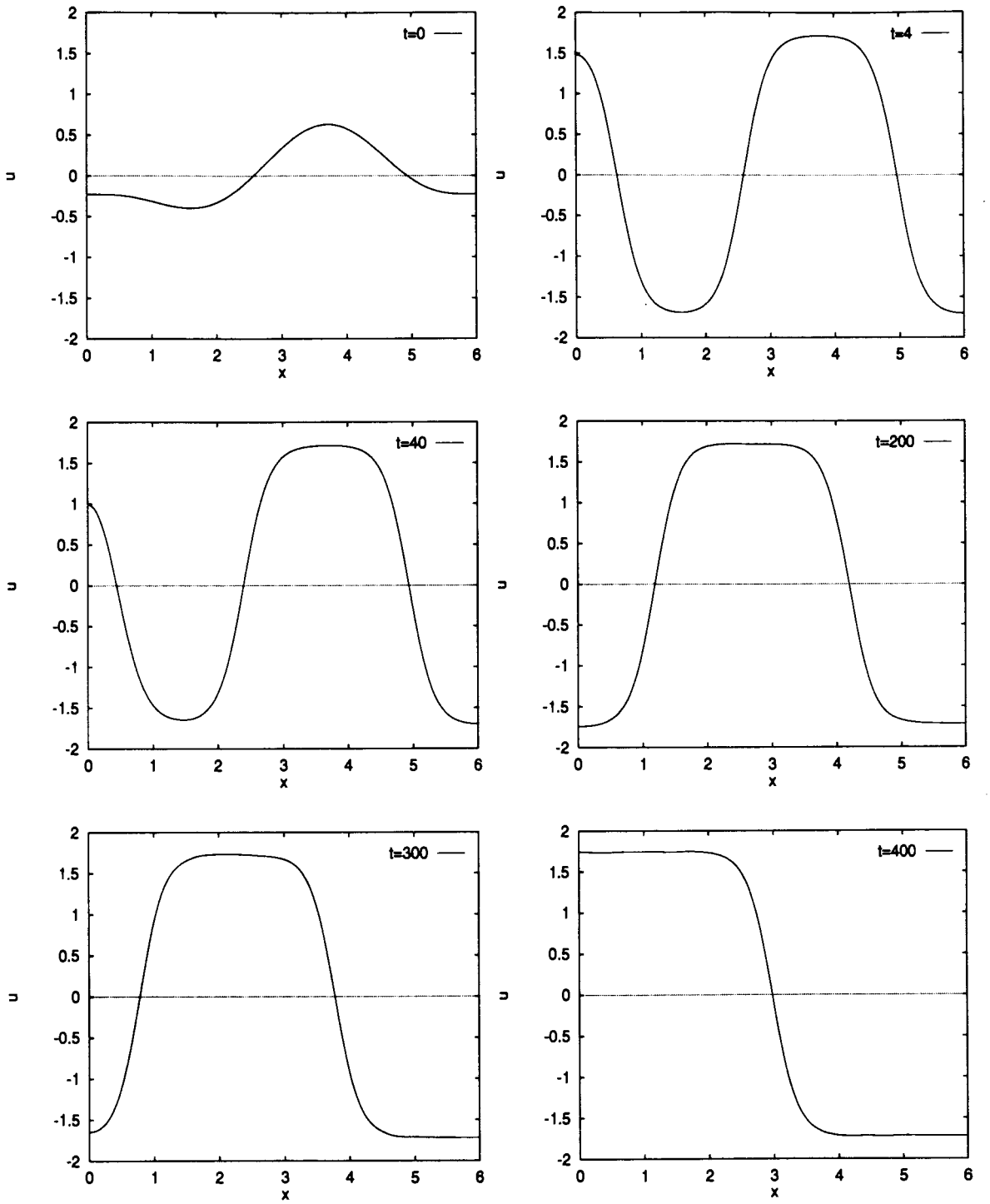


Figure 3.7: Spurious FM solutions showing layer evolving from $t = 0$ to $t = 400$

to one-layer monotone structure. Although the solution evolution in this case resembled that of a true stable solution, it proved otherwise when we decreased the error tolerances for time integration. With $atol = 10^{-6}$ and $rtol = 10^{-6}$, the solution did not change at all from $t = 40$ to $t = 400$, and stayed on a two-layer metastable state as shown in fig. 3.8.

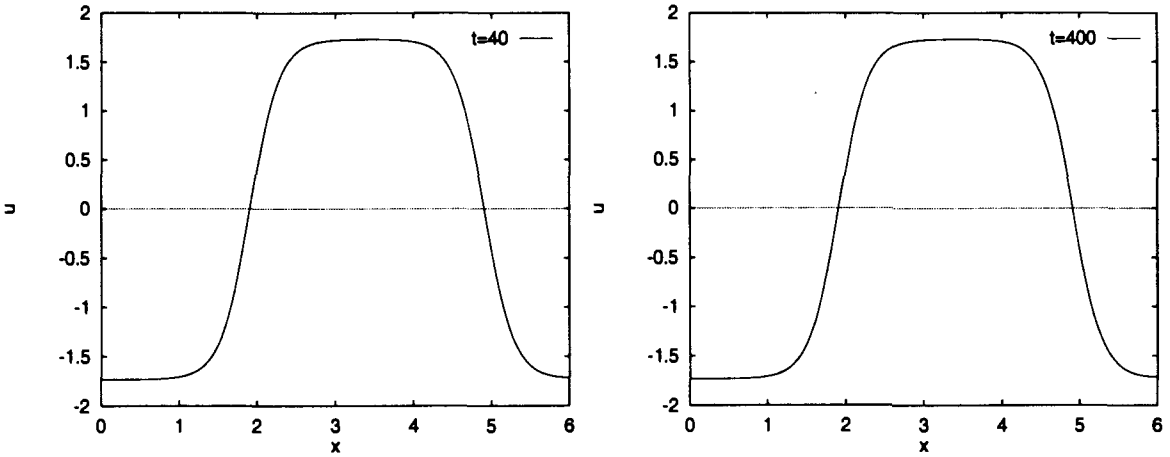


Figure 3.8: FM solutions at $t = 40$ and $t = 400$ with $atol = 10^{-6}$, $rtol = 10^{-6}$

Neither decreasing the time-stepsize nor increasing the number of spatial node-points helped in moving the metastable solution to its stable state. This slow motion of the solution for Cahn-Hilliard model has been widely observed[8]. There exist several explanations pertaining to this behaviour, and they are attributed to *a small interaction length* $\epsilon$ ( $\epsilon = \sqrt{\gamma}$) [3], *slow dissipation of energy* [19], and the evidence of spurious metastable solution introduced by *truncated boundaries* [27].

One known way to tackle this problem of slow-moving phase boundaries is to choose a very high number of spatial nodes and to continue the time-stepping for an enormous length of time (see [27]). We did several experiments with $n = 400$ and differing time lengths, but decided to stop the calculations at $t = 2000$ as neither $u$

[8]In fact, it is a very popular area of numerical research (see [3], [19], [27] etc.)

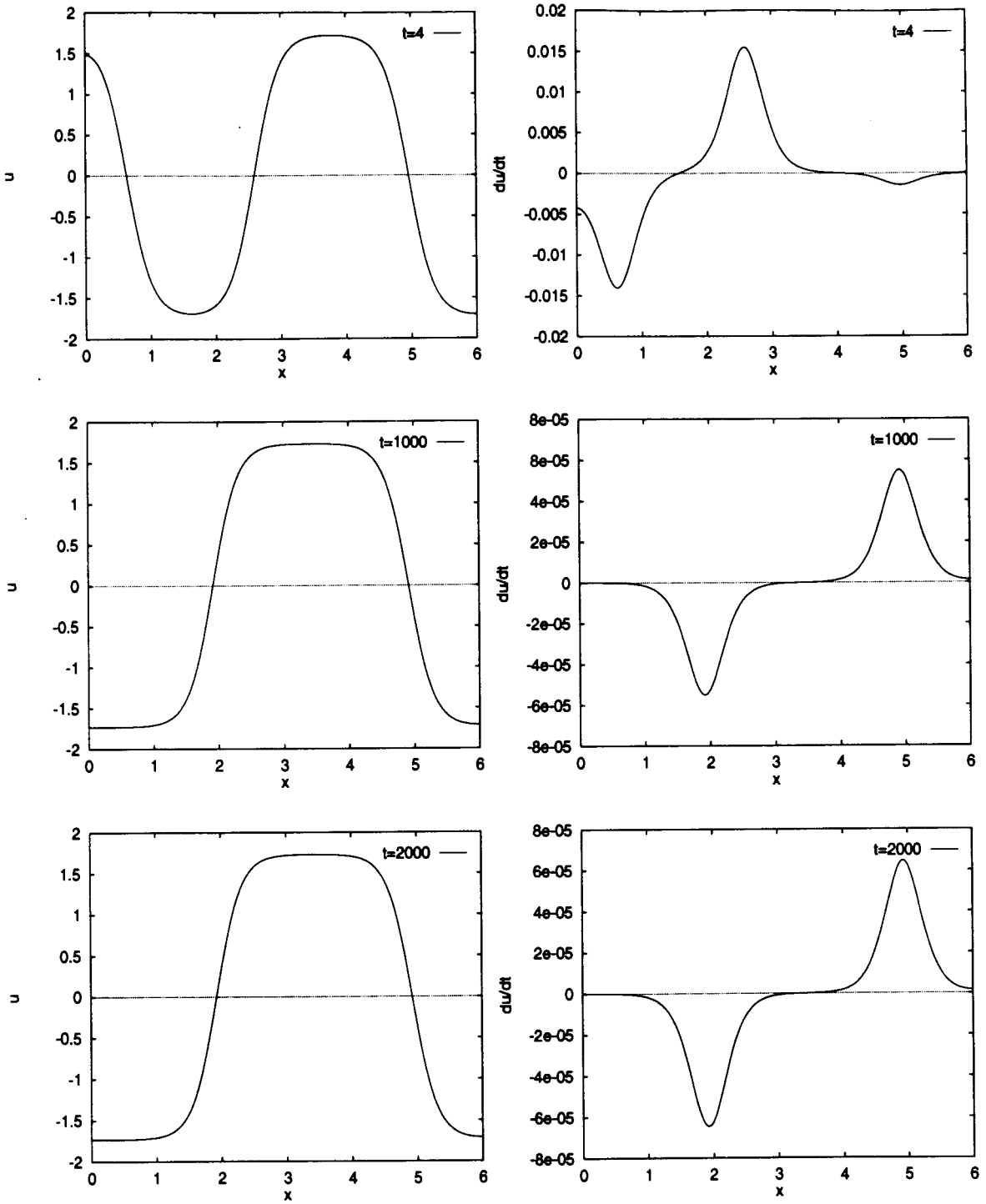


Figure 3.9: FM solutions at $t = 4$, $t = 1000$ and $t = 2000$ with $n = 400$, $atol = 10^{-6}$, $rtol = 10^{-6}$; and their time derivatives

nor $u^t$ showed any considerable change from $t = 1000$ to $t = 2000$ (see fig. 3.9).

It is a well known fact that for small $\epsilon$ these solution layers move at an exponentially small speed of $O(e^{-C/\epsilon})$, where $C$ depends only on the distance between layers. The stable state can be reached faster with a higher value of $\epsilon$. In fact, this was demonstrated when the solution reached the steady-state at $t = 500$ upon increasing the value of $\gamma$ $(= \epsilon^2)$ to 0.17 from 0.07 (fig. 3.10).

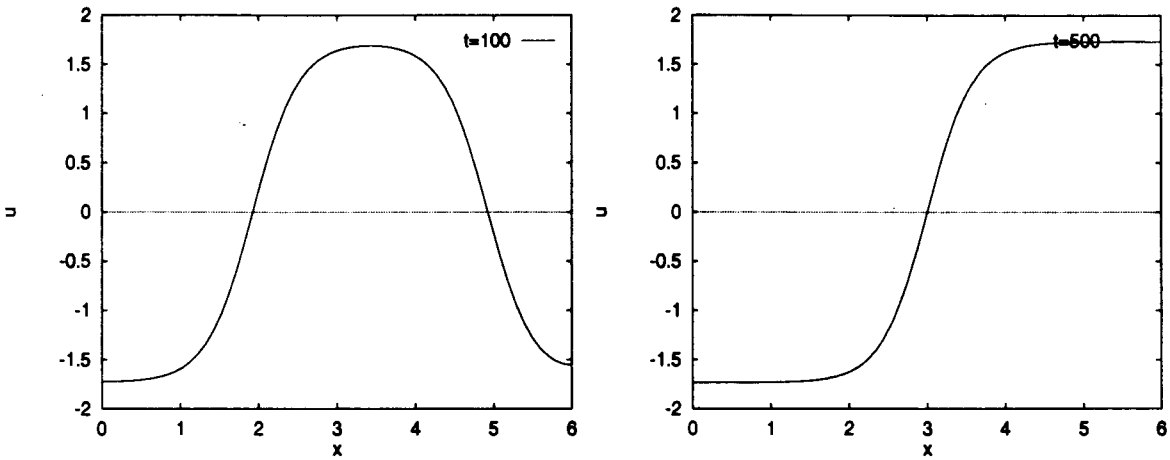


Figure 3.10: Steady-state solution with FM using $\gamma = 0.17$

The results of our experiments with the moving mesh were as follows. With $n = 40$ and $atol = rtol = 10^{-6}$ the MM showed behaviour almost similar to that of a fixed mesh. However, it produced an oscillation near the left layer unlike the fixed mesh (fig. 3.11); also, the mesh adaptation did not reflect the solution behaviour correctly. Smoothing created an adverse effect and made the solution movement slower; an effort to increase the number of nodes resulted in a break-down of DDASSL following repeated error test failures. After a number of experiments with different combinations of MOVCOL parameters, the results remained unchanged which led us to believe that this could be caused by the *monitor function.*

Numerical experiments with the phase separation problems has shown [10] that gradient monitor functions do not place enough emphasis on concentrating the mesh

in the phase change regions. They are more sensitive to larger transitions, and the mesh concentration is strongly dependent on the size of the transition. This can result in very poor resolution of the smaller transitions, thus stopping the movement altogether. Although choosing the right monitor function for these kind of problems could be a time-consuming and difficult task, it could prove to be very worthwhile.

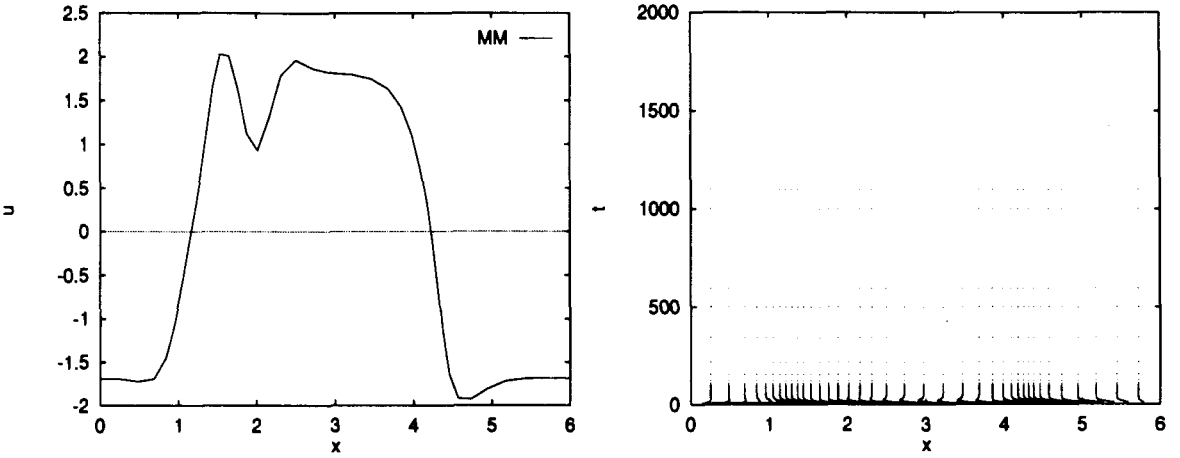


Figure 3.11: MM solution at $t = 4$ and also at $t = 2000$ and the mesh trajectories

We did some experiments with different monitor functions $M(x,t)$; upon changing $M(x,t)$ to

$$M(x,t) = \sqrt{1+\sigma_x^2}, \qquad \text{where} \qquad \sigma = \psi'(u) - \gamma\Delta u,$$

the oscillation disappeared, and also the layers moved. Increasing $n$ to 80 we reached a solution which showed the desired steady monotone structure at $t = 1000$. The results obtained with $n = 200$, was very similar to that with $n = 80$ and exhibited the final phase transition at $t = 1000$. We show our results at $n = 200$ in figure 3.12. at times $t = 4, 40, 100, 500, 1000$, and 2000, respectively.

Notice that this new and effective $M(x,t)$ is in fact the gradient monitor for the *generalised chemical potential* $\sigma$. It is the tiny jumps in potential across the layers that drives this (exponentially) slow motion. Hence it is not surprising that monitoring it

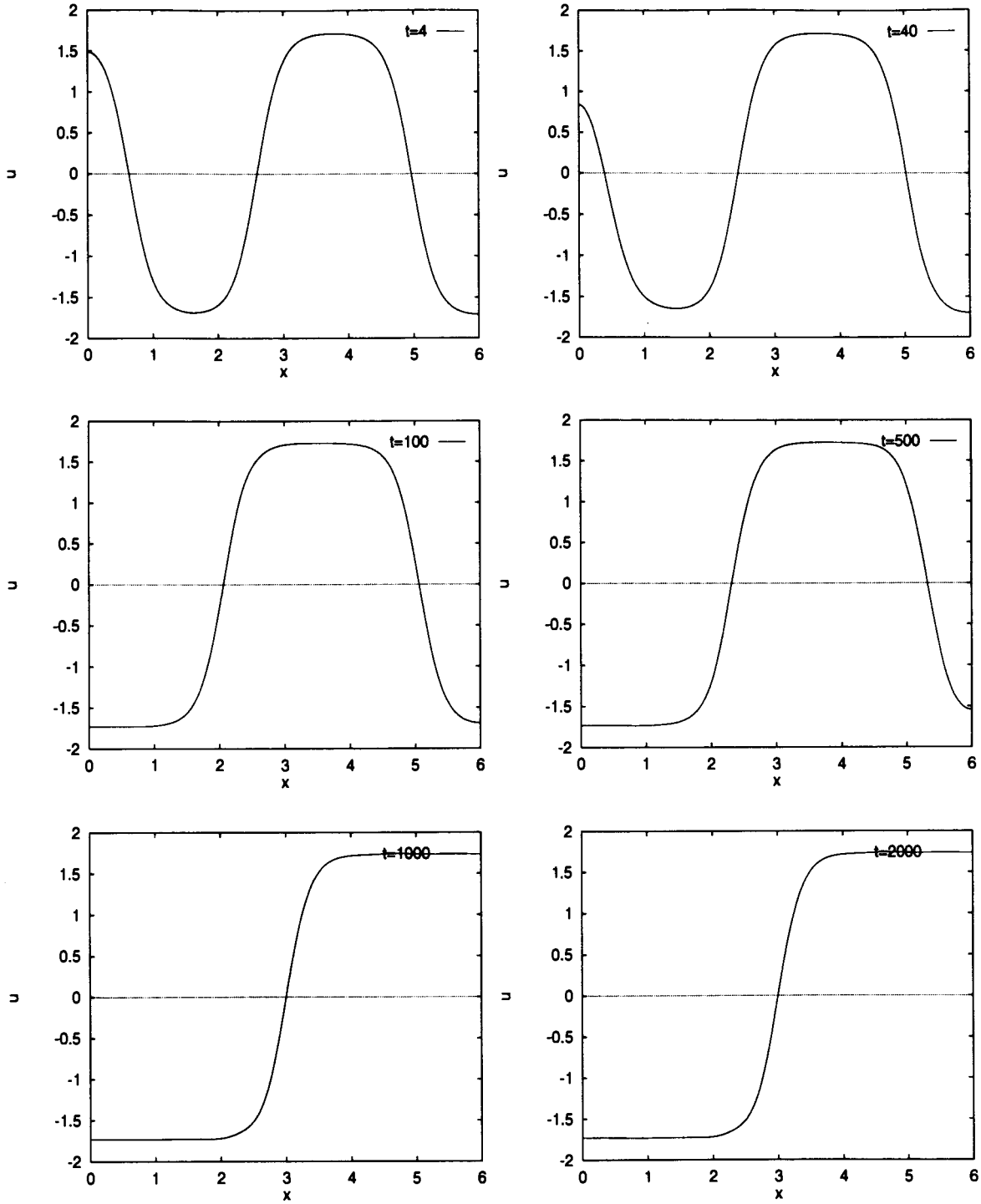


Figure 3.12: MM solutions with $n = 200$ and $atol = 10^{-6}$, $rtol = 10^{-6}$ demonstrating the evolution to the steady-state solution

would enable the mesh to track down the layer movements efficiently to produce the final transition.

Although the numerical results obtained through this monitor function converge to a solution resembling the true solution as we increase grid points in space, the mesh trajectories start behaving like a uniform mesh, as time increases. This is due to the fact that $\sigma_x$ is small, and that leads to $M(x,t) \simeq 1$ giving rise to a grid behaviour (in space) similar to that of a fixed one.

## 3.4 Summary

We started this chapter with the description of the phase separation in a binary alloy, and explained the theory that relates it to the phenomenological Cahn-Hilliard equation. We discussed some of its continuum properties before implementing MOVCOL for solving this problem, with initial conditions leading towards different solution behaviour. Three different cases were considered. The first case starting with a stable initial solution led to a uniform constant state; the second gave rise to a phase separation in a comparatively short time. When the performance of FM and MM were compared, the difference proved to be marginal. The third case, however, was very difficult to tackle; at first, both fixed and moving mesh gave spurious metastable solutions. When nothing seemed to help, a change in the monitor function for MM gave dramatic results, giving rise to the true stable monotone solution. FM with almost five times as many nodes remained on the two-layer metastable state. This proved two things beyond doubt. While a carelessly chosen monitor function can create extremely adverse effect, it is in fact a very powerful tool in successfully implementing a moving mesh method. And secondly, with an efficient and reflective monitor function, MM definitely has a scope of producing clear resolution of sharp transition regions, and is able to cope with gradual and rapid changes in the number of transition regions.

# Chapter 4

# Conclusions

In this chapter we summarize our overall findings obtained during the course of this work. We present some of the difficulties encountered, along with suggested remedies, and some directions towards further research.

In this thesis, we tested the performance of a moving collocation method for four different test problems, all time-dependent PDEs in one space dimension. The paths that the solutions take to reach their respective stable steady state are, however, interestingly different for second set of problems than the first. While Burgers' equation with its small diffusion term forms a wave with a steep gradient moving towards $x = 1$, the reaction-diffusion problem produces a flame front propagating at an exponentially high speed towards $x = 1$ following an ignition occurring at an early state. Despite their apparent dissimilarities, they approach the steady state in a rather fast manner. MOVCOL's performance was found to be excellent for these problems, especially for the more difficult versions, Burgers' equation with a small viscosity parameter $\epsilon$ causing a steeper gradient, or the reaction-diffusion equation with a larger $\delta$ ruling the thinness of the front.

The phase separation problems (Allen-Cahn and Cahn-Hilliard) on the other hand

are found to possess entirely different solution behaviour dominated by exponentially slow propagation rate. For small time, the solutions form a pattern with transition layers and thenceforth propagate at a time scale of $O(e^{-C/\epsilon})$ for $\epsilon$ small, with $\epsilon$ determining the *interaction length* and $C$ depending on the distance between layers.

While the arc-length monitor function that we used for the first two problems demonstrated reliability in moving the mesh-points where needed most, it failed to show the same accuracy in regard to the phase separation problems. It showed a tendency of being more sensitive towards larger transitions, or shock formation, than it was towards smaller ones caused by a tiny jump in potential across the layers found very commonly, e.g. in the Allen-Cahn or Cahn-Hilliard problem. This reconfirmed the already existing view that one single choice of monitor function cannot serve too large of a range of problems satisfactorily, and that choosing a proper monitor function is as important (if not more so) and as imperative as choosing a proper set of parameters.

Although it is $\epsilon$ which dictates the rate of propagation, for a given $\epsilon$, an estimate of the energy decay can be an excellent way to track down the comparatively smaller movements. If the motion is driven by energy dissipation, and the solutions move a large distance without losing much energy, it implies that they must move very slowly. Hence if we can monitor the energy, we can monitor the layer movements, however tiny they might be. Although it needs a rigorous approach and a lot more numerical experiments, but the end results would well justify doing this.

Besides searching for a proper monitor function for these kind of problems, a natural extension for this work would be to study Cahn-Morral systems, which are multicomponent analogues of the Cahn-Hilliard model. Such a system describes the phase separation and coarsening in multicomponent mixtures, and it possess the slow motion exhibited by its binary counterpart.

# Bibliography


[1] S. Adjerid S. and Flaherty J.E., *A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations,* SIAM J. Numer.Anal., 23 (1986), pp.778-795.

[2] S. Adjerid S. and Flaherty J.E., *A moving-mesh finite element method with local refinement for parabolic partial differential equations,* Comput. methods Appl. Mech. Engrg., 55 (1986), pp.3-26.

[3] Alikakos N. D. Bates and Fusco, *Slow motion for the Cahn-Hilliard equation in one space dimension,* J. of Differential Equations, 90 (1991), pp. 81-135.

[4] Blom J.E. and Verwer J.G., *On the use of the Arclength and Curvature monitor in a moving-grid method which is based on the method of lines,* Report NM-N8902, CWI, Amsterdam, 1989 .

[5] Cahn W. E., *On Spinodal decomposition,* Acta. Metall., 9 (1961), pp. 795-801.

[6] Cahn J. W. and Hilliard J. E., *Free energy of a non-uniform system I. Interfacial free energy,* J. of Chem. Phys. 28, (1958), pp. 258-267.

[7] Carr J. Gurtin and Slemrod M., *Structured Phase transitions on a finite interval,* Arch. Rat. Mech. Anal. 86, (1984), pp. 317-351.

[8] Carr J. and Pego R., *Metastable patterns in solutions of* $u_t = \epsilon^2 u_{xx} - f(u)$, Comm. Pure Appl. Math., 42 (1989), pp. 523-576.

[9] Carroll J. and Stewart S., *A comparison of some adaptive space mesh solvers for the numerical solution of parabolic partial differential equations,* manuscript.

[10] Duncan D.B., *A Simple and Effective Moving Mesh for Enthalpy Formulations of Phase Change Problems,* IMA J. Numer. Anal., 11,(1991), pp. 55-78.

[11] Elliot C.M., *The Cahn-Hilliard model for The Kinetics of Phase Separation,* Mathematical Models for Phase Change Problems, Int. Series of Numer Math., Vol 88.

[12] Elliot C.M. and Stuart A.M., *The Global Dynamics of Discrete Semilinear Parabolic Equations,* SIAM J. Numer. Anal. 6, (1993), pp. 1622-1663.

[13] Elliot C.M. and French D.A., *Numerical Studies of the Cahn-Hilliard Equation for Phase Separation,* IMA J. Appl. Math., 38,(1987) , pp. 97-128.

[14] Eyre D.J., *Systems of Cahn-Hilliard Equations,* To Appear.

[15] Elliot C. M., and Zheng S., *On the Cahn-Hilliard Equation,* Arch. Rat. Mech. Anal., 96 (1986), pp. 339-357.

[16] Furzeland, Verwer, and Zageling, *A numerical study of three moving grid methods for one-dimensional PDEs based on the method of lines,* J. Comput. Physics, 89,(1990), pp. 349-388.

[17] Grant C.P., *Slow motion in one dimensional Cahn-Morral systems,* Report CDSNS92-78, Centre for Dynamical System and nonlinear Studies.

[18] Griffiths D. F. and Mitchell A. R., *Stable periodic bifurcations of an explicit discretization of a nonlinear partial differential equation in reaction diffusion,* IMA J. Numer. Anal., (1988), pp. 435-454.

[19] Grant C.P. and Van Vleck E.S., *Slowly-migrating Transition Layers for the Discrete Allen-Cahn and Cahn-Hilliard Equation,* CDSNS94-163, Centre for Dynamical System and Nonlinear Studies.

[20] Huang W., Ren Y., and Russell R. D., *Moving mesh methods based on moving mesh partial differential equations,* J. Comput. Phys., 113, (1994), pp. 279-290.

[21] Huang W., Ren Y., and Russell R. D., *Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle,* SIAM J. Numer. Anal., 31 (1994), pp. 709-730.

[22] Huang W. and Russell R.D., *Analysis of moving mesh PDEs with spatial smoothing,* SIAM J. Numer. Anal. (to appear, 1995).

[23] Huang W. and Russell R.D., *A Moving Collocation Method for Solving Time Dependent PDEs,* submitted for publication.

[24] Lambert J.D., *Numerical Methods for Ordinary Differential Equations,* Wiley, London, 1973.

[25] Petzold L.R., *A description of DASSL: A differential/algebraic system solver,* SAND82-8637, Sandia Labs, Livermore, CA, 1982.

[26] Ren Y. and Russell R.D., *Moving Mesh Techniques Based upon Equidistribution, and their Stability,* SIAM J. Sci. Stat. Comput. 13,(1992),pp. 1285-1286.

[27] Reyna L. G. and Ward M. J., *Metastable Internal Layer Dynamics for the Viscous Cahn-Hilliard Equation,* manuscript.

[28] Schiesser W.E., *The Numerical Method of Lines; Integration of Partial Differential Equations,* Academic Press, San Diego, California, 1991.

[29] Verwer, Blom, Furzeland, and Zageling, *A moving grid method for one-dimensional PDEs based on the method of lines*, Report NM-R8818, Centre for Mathematics and Computer Science.