Manuscript Number: CAM-D-17-01442R1

Title: An adaptive moving mesh method for two-dimensional thin film flow equations with surface tension

Article Type: Research Paper

Section/Category: 65Nxx (Partial differential equations, boundary value problems)

Keywords: Thin film flows; Surface tension; Fingering instability; Adaptive moving mesh; r-adaptive method; Moving Mesh PDEs (MMPDEs)

Corresponding Author: Dr. Shailesh Naire, Ph.D

Corresponding Author's Institution: Keele University

First Author: Abdulghani R Alharbi, PhD

Order of Authors: Abdulghani R Alharbi, PhD; Shailesh Naire, Ph.D

Abstract: In this paper, we extend our previous work [A. Alharbi and S. Naire, An adaptive moving mesh method for thin film flow equations with surface tension, J. Computational and Applied Mathematics, 319 (2017), pp. 365-384.] on a one-dimensional r-adaptive moving mesh technique based on a mesh density function and moving mesh partial differential equations (MMPDEs) to two dimensions. As a test problem, we consider the gravity-driven thin film flow down an inclined and pre-wetted plane including surface tension and a moving contact line. This technique accurately captures and resolves the moving contact line and associated fingering instability. Moreover, the computational effort is hugely reduced in comparison to a fixed uniform mesh.

1. We have successfully applied a r-adaptive moving mesh method based on MMPDEs and mesh density functions to a two-dimensional fourth order order parabolic PDE. To our knowledge this is the first attempt to implement r-adaptive scheme to such a PDE.
2. Numerical experiment on a prototype thin film flow-related PDE with surface tension and a moving contact line shows this technique to accurately resolve the moving contact line and associated fingering instability.
3. This technique hugely reduces the computational effort in comparison to the numerical solution on a fixed uniform mesh.

An adaptive moving mesh method for two-dimensional thin film flow equations with surface tension

Abdulghani Alharbi[a,b], Shailesh Naire[b,*]

[a] *Mathematics Department, Taibah University, Medina, Kingdom of Saudi Arabia, Universities Road, PO Box: 344*

[b] *School of Computing and Mathematics, Keele University, Keele ST5 5BG, United Kingdom*

Abstract

In this paper, we extend our previous work [A. Alharbi and S. Naire, *An adaptive moving mesh method for thin film flow equations with surface tension*, J. Computational and Applied Mathematics, 319 (2017), pp. 365-384.] on a one-dimensional r-adaptive moving mesh technique based on a mesh density function and moving mesh partial differential equations (MMPDEs) to two dimensions. As a test problem, we consider the gravity-driven thin film flow down an inclined and pre-wetted plane including surface tension and a moving contact line. This technique accurately captures and resolves the moving contact line and associated *fingering* instability. Moreover, the computational effort is hugely reduced in comparison to a fixed uniform mesh.

Keywords: Thin film flows; Surface tension; Fingering instability; Adaptive moving mesh; r-adaptive method; Moving Mesh PDEs (MMPDEs)

1. Introduction

Thin liquid film flows driven by external forces are relevant in a wide range of applications. They display interesting dynamics, such as wave propagation

*Corresponding author
Email address: `s.naire@keele.ac.uk` (Shailesh Naire)

and steepening, finite time singularities leading to film rupture and spatial "fingering" instabilities. The interested reader is referred to the review articles by Oron *et al.* [1] and Craster & Matar [2].

Of particular interest, both in the physical and mathematical context, are thin film flow problems which include surface tension [3] and involve moving contact lines [4]. Typically, surface tension is only important in regions of very short length scales, particularly, near the contact lines, where the film's free surface exhibits internal layers. There is a large spatial variation in the film's free surface curvature in these internal layers and away from them surface tension is relatively unimportant and the curvature is almost negligible. In gravity and surface tension gradient-driven flows, the dynamic evolution of these internal layers have been associated with the onset of a transverse (in-plane and perpendicular to the flow) spatial fingering instability near the moving contact line [5–11]. A common example of this is when a sheet of rain spreading due to gravity on a window pane or car windscreen breaks up into long fingers. Their accurate resolution is important to understand the mechanisms behind this instability. From a computational viewpoint, one can then use a locally refined mesh in the regions of large spatial variation (near the contact line(s)) and a coarser mesh elsewhere. In contrast, a uniform mesh would use an unacceptably large number of mesh points especially due to the relatively large spatial scale and long time scale typical in the formation of the fingers. The main motivation for this paper is to employ a moving mesh that locally adapts itself to accurately resolve the internal layers and the associated fingering instability in a computationally efficient way compared to a fixed and uniform mesh.

A long wavelength or lubrication approximation based on the smallness of the film's aspect ratio is commonly employed to derive the thin film flow equations [1, 2]. When surface tension is included, this reduces the governing fluid flow equations and boundary conditions to a fourth order nonlinear parabolic PDE representing the evolution of the film's free surface [12]. In most problems, this may be coupled to a parabolic PDE (usually of second order), for example, representing the concentration of a chemical, such as surfactant or a temperature field. In the context of thin film spreading flows, there have been numerous numerical experiments using the finite difference method on a fixed uniform or nonuniform mesh ([6–11, 13–21], to name a few), the finite element method [22, 23] and spectral methods [24] in both one and two dimensions.

In comparison, very few numerical studies have considered an adaptive mesh [8, 9, 16–18, 21]. These studies have used general purpose publicly available solvers for parabolic PDEs which have built-in adaptive mesh capabilities (e.g., PDECOL [25] and TOMS731 [26–29]). Sun *et al.* [30] use a h-adaptive finite element mesh refinement method based on an optimal interpolation error estimate for a two dimensional thin film equation of gravity driven flow down an inclined plane. Li *et al.* [31] have also developed a h-adaptive finite difference method for this equation using a fully discrete and nonlinear multigrid scheme and adaptive mesh refinement method. The above adaptive mesh schemes were shown to capture and resolve the moving contact line and the associated fingering instability accurately and computationally efficiently compared to a fixed uniform grid scheme. In a recent paper, Alharbi & Naire [32] successfully implemented the r-adaptive moving mesh technique [33, 34] for a one-dimensional thin film equation with surface tension. This technique utilises a mesh density function and moving mesh partial differential equations (MMPDEs) to adapt and move the mesh coupled to the PDE(s) describing the thin film flow problem. Numerical experiments on two one-dimensional test problems showed that this technique accurately resolves the multiple internal layers observed in these problems. Moreover, it reduces the computational effort in comparison to a fixed uniform mesh. Encouraged by the success of our earlier work, we now extend it to two dimensions.

The rest of the paper is organized as follows. In section §2, we provide a brief overview of MMPDEs and state the two-dimensional MMPDEs and mesh density functions used in our numerical simulations. In §3, we briefly describe the governing PDE and boundary conditions for two dimensional gravity-driven thin liquid film flow using a lubrication theory model. In §4, the spatial discretisation of the governing equation and the MMPDEs using the finite difference method is presented. In §5, we present the numerical results. Conclusions are given in §6.

2. Moving Mesh Partial Differential Equations (MMPDEs) and mesh density functions

The underlying principle behind r-adaptive moving mesh methods is the the equidistribution principal in which a continuous function defined over an interval is evenly distributed between the subintervals determined by the mesh points. Using this a number of moving mesh equations (MMPDEs) can be developed for time dependent problems which are continuous forms of mesh movement strategies formulated in terms of coordinate transformations. The mesh adapts itself based on a mesh density function which is related to a specific solution characteristic. The interested reader is referred to the book, *Adaptive Moving Mesh Methods*, by Huang & Russell [33] and the review paper by Budd *et al.* [34], who have made seminal contributions in this area over the past twenty years.

In our earlier paper [32], we had listed four of the commonly used one dimensional MMPDEs, the so-called MMPDEs 4, 5 and 6 and modified MMPDE5, and two mesh density functions based on arc-length and curvature. In this work, we extend MMPDEs 4 and 5 to two dimensions. The MMPDEs can be represented as a coordinate transformation:

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t) \colon \boldsymbol{\xi} \in \Omega_c \times \Omega_c \equiv [0,1] \times [0,1] \to \boldsymbol{x} \in \Omega_p \equiv [a,b] \times [c,d],\ t > 0,$$

where $\boldsymbol{x} = (x, y)$ and $\boldsymbol{\xi} = (\xi, \eta)$ are the spatial variables, t is time, Ω_c and Ω_p are referred to as the computational and physical domains, respectively, and a fixed uniform mesh is used to represent any discretisation of Ω_c. Using this, the two dimensional version of MMPDEs 4 and 5 can be written as

$$\text{MMPDE4}: \quad \tau \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\hat{\rho}(\boldsymbol{x}, t) \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}_t) = -\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\hat{\rho}(\boldsymbol{x}, t) \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}), \tag{1}$$

$$\text{MMPDE5}: \quad \tau \boldsymbol{x}_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\hat{\rho}(\boldsymbol{x}, t) \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}). \tag{2}$$

Here, $\boldsymbol{\nabla}_{\boldsymbol{\xi}} = \left(\frac{\partial}{\partial \xi}, \frac{\partial}{\partial \eta}\right)$, $\hat{\rho}(\boldsymbol{x}, t)$ is a mesh density function (defined below) and $\tau > 0$ is a user-specified parameter. τ adjusts the response time of mesh movement to changes in the monitor function $\hat{\rho}(\boldsymbol{x}, t)$ [33]. The smaller τ, the more quickly the mesh responds to changes in $\hat{\rho}(\boldsymbol{x}, t)$. In the limit of $\tau \to 0$, the mesh relaxes to its quasi-steady state given by the solution of a system of elliptic partial differential equations: $\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\hat{\rho}(\boldsymbol{x}, t) \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}) = 0$. Likewise, the mesh moves slowly when a large value of τ is used. MMPDE5 given in Eq. (2) is generally quite stiff and a regularised form is used in practice,

$$\text{regularised MMPDE5}: \quad \tau (1 - \gamma_1 \nabla^2_{\boldsymbol{\xi}}) \boldsymbol{x}_t = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot (\hat{\rho}(\boldsymbol{x}, t) \boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{x}). \tag{3}$$

Here, the parameter $\gamma_1 > 0$ is related to the mesh density function $\hat{\rho}$ (see [34] and references therein). The boundary conditions for the above second order PDEs are

$$x(0,\eta,t)=a, \quad x(1,\eta,t)=b, \quad x_\eta(\xi,0,t)=x_\eta(\xi,1,t)=0, \tag{4}$$

$$y(\xi,0,t)=c, \quad y(\xi,1,t)=d, \quad y_\xi(0,\eta,t)=y_\xi(1,\eta,t)=0. \tag{5}$$

The initial conditions are

$$x(\xi,\eta,0)=(b-a)\xi+a, \quad y(\xi,\eta,0)=(d-c)\eta+c, \tag{6}$$

which represents a uniform initial mesh on the physical domain $\Omega_p \equiv [a,b]\times[c,d]$.

One disadvantage in using the above MMPDEs is that we need to solve two additional PDEs for the adaptive moving mesh coupled to the underlying PDE(s) which could make the computational task more intensive. Another disadvantage in two dimensions is that the coordinate transformation underlying the above MMPDEs cannot ensure existence and uniqueness of the solution only based on the equidistribution principle. This could result in mesh tangling and loss of mesh regularity (i.e., mesh elements that have very small aspect ratio). These can be overcome by mesh generation methods based on optimal transportation, for example, the so-called optimal transport equations, such as the Monge-Ampère and Parabolic Monge-Ampère (PMA) equations (see Budd *et al.* [35, 36]). The advantage of the PMA equation over the MMPDEs is that there is one equation less to solve in two dimensions to obtain the mesh and the meshes are generally regular and there is no mesh tangling. For the test problem considered here the MMPDEs have always generated regular meshes and there is no tangling. We do not consider optimal transport equations in this work.

The choice of the mesh density function $\hat{\rho}$ is essential for the success of adaptive moving mesh methods. They can be chosen based on error estimates (for example, polynomial interpolation or truncation error) or on the solution characteristics of the underlying PDE (for example, arc length or curvature). In the latter case, the mesh density function can be defined by the solution $u(\boldsymbol{x},t)$ (say) of the underlying PDE and possibly its derivatives. The choices of $\hat{\rho}$ that are commonly used are:

$$\text{arc length}: \ \hat{\rho}(\boldsymbol{x},t) = \sqrt{1+\alpha|\boldsymbol{\nabla} u(\boldsymbol{x},t)|^2}, \tag{7}$$

$$\text{curvature}: \quad \hat{\rho}(\boldsymbol{x},t) = \left(\alpha + \beta|\boldsymbol{\nabla}^2 u(\boldsymbol{x},t)|^2\right)^{1/n}, \ \text{where} \ n = 2 \text{ or } 4. \tag{8}$$

Here, α and β are adaptivity parameters (or weight parameters) of the mesh density function [33, 37, 38]. These are usually taken to be constant but could be dependent on the spatial variables if there are multiple regions over which the solution characteristics vary rapidly (see example in Alharbi & Naire [32]). The above monitor functions can also be extended to include multiple solution components (see example in Alharbi & Naire [32]). In addition, it is common practice in the context of moving mesh methods to smooth the monitor function in order to obtain a smoother mesh and also to make the MMPDEs easier to integrate. This is discussed in §4.

3. Two-dimensional thin film equations for gravity-driven spreading down an inclined and pre-wetted plane

We consider the two-dimensional thin liquid film flow of a droplet spreading down an inclined and pre-wetted substrate due to gravity (see Fig. 1). The

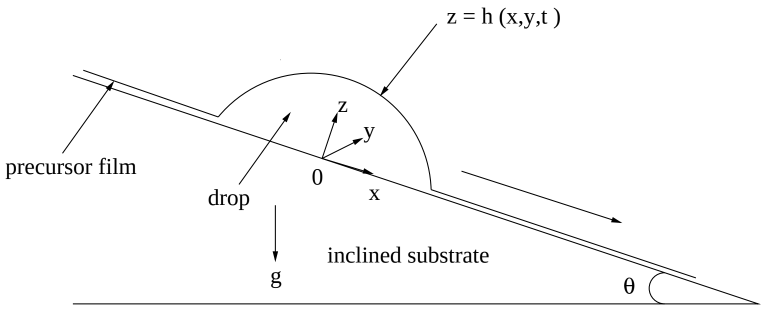

Figure 1: Schmatic of a two-dimensional drop or sheet spreading down an inclined and pre-wetted substrate.

bulk flow is governed by the Navier-Stokes equations. We assume that the substrate is pre-wetted with a thin precursor liquid film. Lubrication (or long wavelength) theory can be applied to reduce the governing equations and boundary conditions at the free surface to give the evolution equations for the film's free surface. The interested reader can refer to Kondic [7] for the derivation. This can be written in non-dimensional form as:

$$h_t + \boldsymbol{\nabla} \cdot \left[Ca\ h^3 \boldsymbol{\nabla} \nabla^2 h - D(\theta) h^3 \boldsymbol{\nabla} h \right] + \left[h^3 \right]_x = 0. \tag{9}$$

Here, $h = h(x, y, t)$, is the film height, x and y are the in-plane spatial variables along and perpendicular to the flow direction, respectively, and t is time. The dimensionless parameters $D(\theta)$, is the ratio of the size of the vertical and the horizontal components of gravity, Ca, is an inverse capillary number (compares surface tension to viscous forces) and θ is the inclination angle of the substrate. Eq. (9) is a nonlinear parabolic PDE of fourth order in space. The fourth order term (second term in Eq. (9)) is related to the curvature of the film's free surface and is due to surface tension. Typically $Ca \ll 1$, so there exist internal layers where curvature is important; elsewhere curvature is almost negligible and the film evolution is controlled by the horizontal component of gravity (fourth term in Eq. (9)). The second order diffusion term (third term in Eq. (9)) is related to the vertical component of gravity and is also only important in the internal layers where it has a smoothing influence on the film evolution there.

The boundary conditions (BCs) for the above PDE are prescribed as follows.

$$\begin{aligned} &h(\pm L_x, y, t) = b, \quad h_x(\pm L_x, y, t) = h_{xxx}(\pm L_x, y, t) = 0, \\ &h(x, -L_y/2, t) = h(x, L_y/2, t), \end{aligned} \tag{10}$$

where L_x and L_y are the lengths of the physical domain in the x and y directions, respectively. The first three BCs assume that the plane is pre-wetted with a precursor film of thickness $b \ll 1$ (represents ratio of precursor film thickness to initial drop height) and the drop connects onto a flat precursor film. We prescribe periodic boundary conditions in the y-direction.

The initial condition for h represents a y-independent droplet with sinusoidal perturbations imposed on it and is given by:

$$\begin{aligned} &h(x, y, 0) = h_0(x) + h_1(x, y), \\ &h_0(x) = (1 - x^2)[H(1 - x) - H(-1 - x)] + b[H(x - 1) + H(-1 - x)]. \end{aligned} \tag{11}$$

The expression for $h_0(x)$ (following [9, 17, 39, 40]) represents a parabolic-shaped drop and $x = \pm 1$, is the initial location where it connects to the precursor film both upstream and downstream, and $H(x)$ is the Heaviside function. The form of the sinusoidal perturbation $h_1(x, y)$ will be described in §5.

4. Finite difference semi-discretisation scheme on a moving adaptive mesh

The adaptive moving mesh method uses a coordinate transformation from the computational domain with coordinate $\boldsymbol{\xi} = (\xi, \eta)$, to the physical domain with coordinate $\boldsymbol{x} = (x, y)$: $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}, t) \colon \Omega_c \equiv [0, 1] \times [0, 1] \to \Omega_p \equiv [-L_x, L_x] \times [-L_y/2, L_y/2]$, $t > 0$. Then the solution can be written as $h(\boldsymbol{x}, t) = h(x(\boldsymbol{\xi}, t), t)$. A uniform mesh on the computational domain is described as: $\mathcal{J}_h^c(t) \colon \xi_j = (j-1)\Delta\xi,\ \eta_k = (k-1)\Delta\eta$, for $j = 1, \ldots, M_x+1,\ k = 1, \ldots, N_y + 1$, and a moving mesh on the physical domain associated with the solution $h(x(\boldsymbol{\xi}, t), t)$ is described as: $\mathcal{J}_h^p(t) \colon \boldsymbol{x}_{j,k}(\boldsymbol{\xi}) = \boldsymbol{x}(\xi_j, \eta_k, t), j = 1, \ldots, M_x + 1, k = 1, \ldots, N_y + 1$, where the boundary nodes are given by: $x_{1,k} = -L_x,\ x_{M_x+1,k} = L_x, k = 1, \ldots N_y + 1$ and $y_{j,1} = -L_y/2,\ y_{j,M_y+1} = L_y/2, j = 1, \ldots M_x + 1$. Here, $\Delta\xi = \dfrac{1}{M_x}$ and $\Delta\eta = \dfrac{1}{N_y}$ denote the uniform grid size in the computational domain and M_x, N_y are given positive integers denoting the number of mesh points in the x and y direction, respectively.

Eq. (9) is reformulated in Lagrangian form and can be written in terms of the computational coordinates (ξ, η) as:

$$h_t - (h_x x_t + h_y y_t) + \boldsymbol{\nabla} \cdot \left[Ca\ h^3 \boldsymbol{\nabla}\boldsymbol{\nabla}^2 h - D(\theta)\ h^3 \boldsymbol{\nabla} h\right] + \left[h^3\right]_x = 0, \quad (12)$$

where

$$h_x = \frac{1}{J}\left[(J\xi_x)h_\xi - (J\eta_x)h_\eta\right], h_y = \frac{1}{J}\left[-(J\xi_y)h_\xi + (J\eta_y)h_\eta\right], \quad (13)$$

$$\left[h^3\right]_x = \frac{1}{J}\left[(J\xi_x)(h^3)_\xi - (J\eta_x)(h^3)_\eta\right], J = x_\xi y_\eta - x_\eta y_\xi, \quad (14)$$

$$\boldsymbol{\nabla} \cdot \left[h^3 \boldsymbol{\nabla} h\right] = \frac{1}{J}\left[\left(h^3 R\right)_\xi + \left(h^3 S\right)_\eta\right], \quad (15)$$

$$G = \boldsymbol{\nabla}^2 h = \frac{1}{J}\left[R_\xi + S_\eta\right], \tag{16}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}\boldsymbol{\nabla}^2 h\right] = \boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right] = \frac{1}{J}\left[\left(h^3 R_1\right)_\xi + \left(h^3 S_1\right)_\eta\right]. \tag{17}$$

The functions R, S, R_1 and S_1 are defined as:

$$\begin{aligned} &R = \left[a\ h_\xi + c\ h_\eta\right],\ \ S = \left[q\ h_\eta + c\ h_\xi\right],\ \ R_1 = \left[a\ G_\xi + c\ G_\eta\right],\\ &S_1 = \left[q\ G_\eta + c\ G_\xi\right], \end{aligned} \tag{18}$$

where

$$\begin{aligned} &a = \frac{1}{J}\left[(J\xi_x)^2 + (J\xi_y)^2\right],\ \ q = \frac{1}{J}\left[(J\eta_x)^2 + (J\eta_y)^2\right],\\ &c = \frac{1}{J}\left[(J\xi_x)(J\eta_x) + (J\xi_y)(J\eta_y)\right],\\ &\xi_x = \frac{1}{J}y_\eta, \eta_x = -\frac{1}{J}y_\xi, \xi_y = \frac{1}{J}x_\eta, \eta_y = \frac{1}{J}x_\xi. \end{aligned} \tag{19}$$

A *conservative* finite difference semi-discretisation scheme for the spatial derivatives in Eq. (12) on the uniform mesh $\mathcal{J}_h^c$ using centred finite differences can be written as, keeping the time derivative continuous,

$$\begin{aligned} &h_{t,j,k} - \left(h_x x_t + h_y y_t\right)_{j,k} + \boldsymbol{\nabla}\cdot\left[Ca\ h^3\boldsymbol{\nabla}\boldsymbol{\nabla}^2 h - D(\theta)\ h^3\boldsymbol{\nabla}h\right]_{j,k}\\ &+ (h^3)_{x,j,k} = 0,\ \forall\ j = 2,\ldots,M_x,\ \ k = 1,\ldots,N_y+1. \end{aligned} \tag{20}$$

The finite-difference discretisation scheme used for each of the terms appearing above are given in Eqs. (A.1)-(A.18) in Appendix A. The boundary conditions $h(0,y,t) = 1$ and $h(L_x,y,t) = b$ are replaced by their ODE form: $h_{t,1,k} = h_{t,M_x+1,k} = 0,\ \forall\ k = 1,\ldots,N_y+1$.

A semi-discretisation scheme to discretise the spatial derivatives in MMPDE4 given in Eq. (1) is as follows, keeping the time derivative continuous:

$$\text{MMPDE4}: \quad \begin{aligned} &\tau\left[\tfrac{1}{\Delta\xi^2}\left(\hat{\rho}_{j+\frac{1}{2},k}\Delta_j \boldsymbol{x}_{t,j,k} - \hat{\rho}_{j-\frac{1}{2},k}\Delta_j \boldsymbol{x}_{t,j-1,k}\right)\right.\\ &\left.-\tfrac{1}{\Delta\eta^2}\left(\hat{\rho}_{j,k+\frac{1}{2}}\Delta_k \boldsymbol{x}_{t,j,k} - \hat{\rho}_{j,k-\frac{1}{2}}\Delta_j \boldsymbol{x}_{t,j,k-1}\right)\right] = -E_{j,k},\\ &\forall\ j = 2,\ldots,M_x,\ \ k = 2,\ldots,N_y, \end{aligned} \tag{21}$$

where

$$E_{j,k} = \tau\left[\frac{1}{\Delta\xi^2}\left(\hat{\rho}_{j+\frac{1}{2},k}\Delta_j\boldsymbol{x}_{j,k} - \hat{\rho}_{j-\frac{1}{2},k}\Delta_j\boldsymbol{x}_{j-1,k}\right) - \frac{1}{\Delta\eta^2}\left(\hat{\rho}_{j,k+\frac{1}{2}}\Delta_k\boldsymbol{x}_{j,k} - \hat{\rho}_{j,k-\frac{1}{2}}\Delta_k\boldsymbol{x}_{j,k-1}\right)\right]. \tag{22}$$

In the above, $\Delta_j\boldsymbol{x}_{t,j,k} = \boldsymbol{x}_{t,j+1,k} - \boldsymbol{x}_{t,j,k}$, $\Delta_k\boldsymbol{x}_{t,j,k} = \boldsymbol{x}_{t,j,k+1} - \boldsymbol{x}_{t,j,k}$, $\Delta_j\boldsymbol{x}_{j,k} = \boldsymbol{x}_{j+1,k} - \boldsymbol{x}_{j,k}$ and $\Delta_k\boldsymbol{x}_{j,k} = \boldsymbol{x}_{j,k+1} - \boldsymbol{x}_{j,k}$. Also, evaluations at half mesh points are obtained as an average of the neighbouring mesh points. The semi-discretisation of MMPDE5 given in Eq. (3) is similar to the above and is provided by Eq. (A.20) in Appendix A. The boundary conditions $x(0,\eta,t) = 0$, $x(1,\eta,t) = L_x$, $y(\xi,0,t) = -L_y/2$ and $y(\xi,1,t) = L_y/2$ are replaced by their ODE form: $x_{t,1,k} = x_{t,M_x+1,k} = 0, \forall\ k = 1,\ldots,N_y+1$; $y_{t,j,1} = y_{t,j,N_y+1} = 0, \forall\ j = 1,\ldots,M_x+1$.

The curvature mesh density function $\hat{\rho}(\boldsymbol{x},t)$ given in Eq. (8) is discretised using finite differences as follows:

$$\hat{\rho}_{j,k} = (1 + \alpha|(\boldsymbol{\nabla}^2 h)_{j,k}|^2)^{\frac{1}{n}},\ \forall\, j = 1,\ldots,M_x+1,\ k = 1,\ldots,N_y+1, \tag{23}$$

where $(\boldsymbol{\nabla}^2 h)_{j,k}$ is approximated by Eq. (A.5). A smoothing scheme suggested by Huang [33, 41] and based on weighted averaging is used to smooth the mesh density function. This is provided in Appendix A.

Eq. (20) and Eqs. (21) or (A.20), form a coupled system of $3(M_x+1)(N_y+1)$ ordinary differential equations (ODEs) for the solution $h_{1,1},\ldots,h_{M_x+1,N_y+1}$ and the mesh $\boldsymbol{x}_{1,1},\ldots,\boldsymbol{x}_{M_x+1,N_y+1}$, with initial conditions for $\boldsymbol{x}$ and h given by Eqs. (6),(11). These are solved by the Method of Lines using the stiff ODE solver DASPK [42] which uses Backward Differentiation Formulas (BDF) to approximate the time derivative. This solver uses an iterative method (based on preconditioned Krylov subspace method) to solve the linearised system of equations. DASPK also allows approximating the Jacobian using an Incomplete LU factorisation. This has a significant influence on the performance. However, we need to choose a sufficiently large fill-in for the LU factorisation of the Jacobian, otherwise the convergence of the iterative solver is very slow. We use a staggered system for numbering the unknowns, $h_{1,1}$, $x_{1,1}$, $y_{1,1}$, $h_{1,2}$, $x_{1,2}$, $y_{1,2}$,…,h_{M_x+1,N_y+1}, x_{M_x+1,N_y+1}, y_{M_x+1,N_y+1}, which provides a smaller bandwidth for the Jacobian matrix. This is in comparison to, for example, the numbering $h_{1,1}$, $h_{1,2}$,…, h_{M_x+1,N_y+1}, $x_{1,1}$, $x_{1,2}$, …, x_{M_x+1,N_y+1},

$y_{1,1}$, $y_{1,2}, \ldots$, y_{M_x+1,N_y+1}, which although sparse has a much bigger bandwidth. This significantly improves the performance of the ODE solver. We note here that a non-uniform initial mesh had to be used instead of the initial mesh in Eq. (6) which had an influence on the solution and the performance of the solver in comparison to the uniform initial mesh. This nonuniform initial mesh was obtained by solving in pseudo-time the chosen MMPDE (with the uniform mesh as the initial condition) with h fixed (hence, the mesh density function $\hat{\rho}$ is also fixed) at its initial condition given by Eq. (11). This mesh was then used to solve the MMPDEs in real time. For the Regularised MMPDE5 equation, we had to choose the parameter values $\tau = 1$ and $\gamma_1 = \sqrt{\max(\rho)}$ for the pseudo-time calculation after which $\tau = 10^{-2}$ was chosen for the solution in real time.

5. Numerical results

In this section, we perform numerical experiments on the two-dimensional thin film spreading flow problem for the free surface thickness h given by Eq. (9). In all the results presented below, the system parameter values are: $Ca = 10^{-3}$ (indicating smaller surface tension relative to viscous forces), $\theta = 90^o$ (representing a vertical substrate; so $D(\theta) = 0$), $b = 10^{-1}$ (the precursor film thickness is 1/10 times smaller than the initial drop or sheet thickness), $L_{x_1} = -2$, $L_{x_2} = 18$ (the upstream and downstream length of the computational domain, respectively) and $L_y = 2$. We only show the results using MMPDE4; the results using MMPDE5 are similar and hence not reported here. We are interested in the development of the fingering instability starting from a y-independent initial condition (given by $h_0(x)$ in Eq. (11)) with sinusoidal perturbations imposed on it (given by $h_1(x, y)$ in Eq. (11)). The form of $h_1(x, y)$ is chosen as

$$h_1(x, y, 0) = \sum_{l=1,k=1}^{m,n} a_k \cos(k\pi y)\ e^{\left[-K_l\ (x-x_l)^2\right]}, \tag{24}$$

where k is the wavenumber of each sinusoidal mode with period $= 2/k$ and amplitude a_k, n is the total number of modes imposed, m is the total number of locations $x = x_l$ in the x direction across which the sinusoidal perturbations are applied and K_l controls the width of the localised perturbations at

x_l. In the numerical simulations shown below, a single transverse perturbation ($k = n = 1$) is applied at two locations: $x_1 = 0$ with amplitude $a_1 = 0.1$ and $x_2 = 1$ (the leading edge of the initial droplet) with amplitude $a_1 = 0.01$. We choose $K_1 = 20$ and $K_2 = 100$.

Fig. $2(a, b, c)$ illustrate the surface plots of $h(x, y, t)$ (side view) showing its evolution in time (the times shown are $t = 40$, $t = 80$ and $t = 100$) using the adaptive moving mesh scheme with $M_x = 400$ and $N_y = 40$ (so, the initial $\Delta x = \Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature monitor function with $\alpha = 1$ and $n = 2$. At early time, a finger is observed to slowly start forming as shown in Fig. $2(a)$. As time t increases, the finger appears to lengthen with a preferred width as observed in Figs. $2(b, c)$. This is highlighted in the surface plot of $h(x, y, t)$ (top view) shown in Fig. $3(c)$ at $t = 80$. The width of the finger is approximately one (between $y = -0.5$ and $y = 0.5$). These results are visually identical to those obtained using a uniform mesh with a higher resolution. For example, Fig. $3(b)$ shows the surface plot of $h(x, y, t)$ (top view) at $t = 80$ using a uniform mesh with $M_x = 2000$ and $N_y = 200$ ($\Delta x = \Delta y = 0.01$). Visual comparison with the adaptive mesh simulation shown in Fig. $3(c)$) shows them to be nearly indistinguishable. Fig. $3(a)$ shows the surface plot of $h(x, y, t)$ (top view) at $t = 80$ using a uniform mesh with $M_x = 400$ and $N_y = 40$ ($\Delta x = \Delta y = 0.05$). This has the same mesh size as the initial mesh used in the adaptive mesh simulation shown in Fig. $3(c)$, however, the solution clearly has not converged. This clearly indicates that the adaptive mesh solution achieves a much higher accuracy using a coarser initial mesh in comparison to the corresponding fixed uniform mesh solution. Figs. $4(a, b, c)$ show the adaptive moving mesh, $x(\xi, \eta, t)$ and $y(\xi, \eta, t)$, at times $t = 40$, $t = 80$ and $t = 100$, respectively. We clearly see how the mesh adapts itself in both the x and y directions as the propagating finger gradually develops with localised clustering of mesh points in the y-direction along the finger.

To quantify the measure of accuracy of the adaptive mesh solution over the fixed uniform mesh solution we provide in Table 1 some metrics for accuracy which focus on the important aspects of the solution. These are: the L_2 norm error (characterises the mean error over the entire domain), h_{max}, the maximum height of the advancing front (the so-called capillary ridge) at the mid-line $y = 0$, x_{front}, the location of the "effective" contact line near the advancing front at the mid-line $y = 0$ (where the leading edge of the front connects onto the precursor film), L_{finger}=$x_{front}(y = 0)$−$x_{front}(y =$

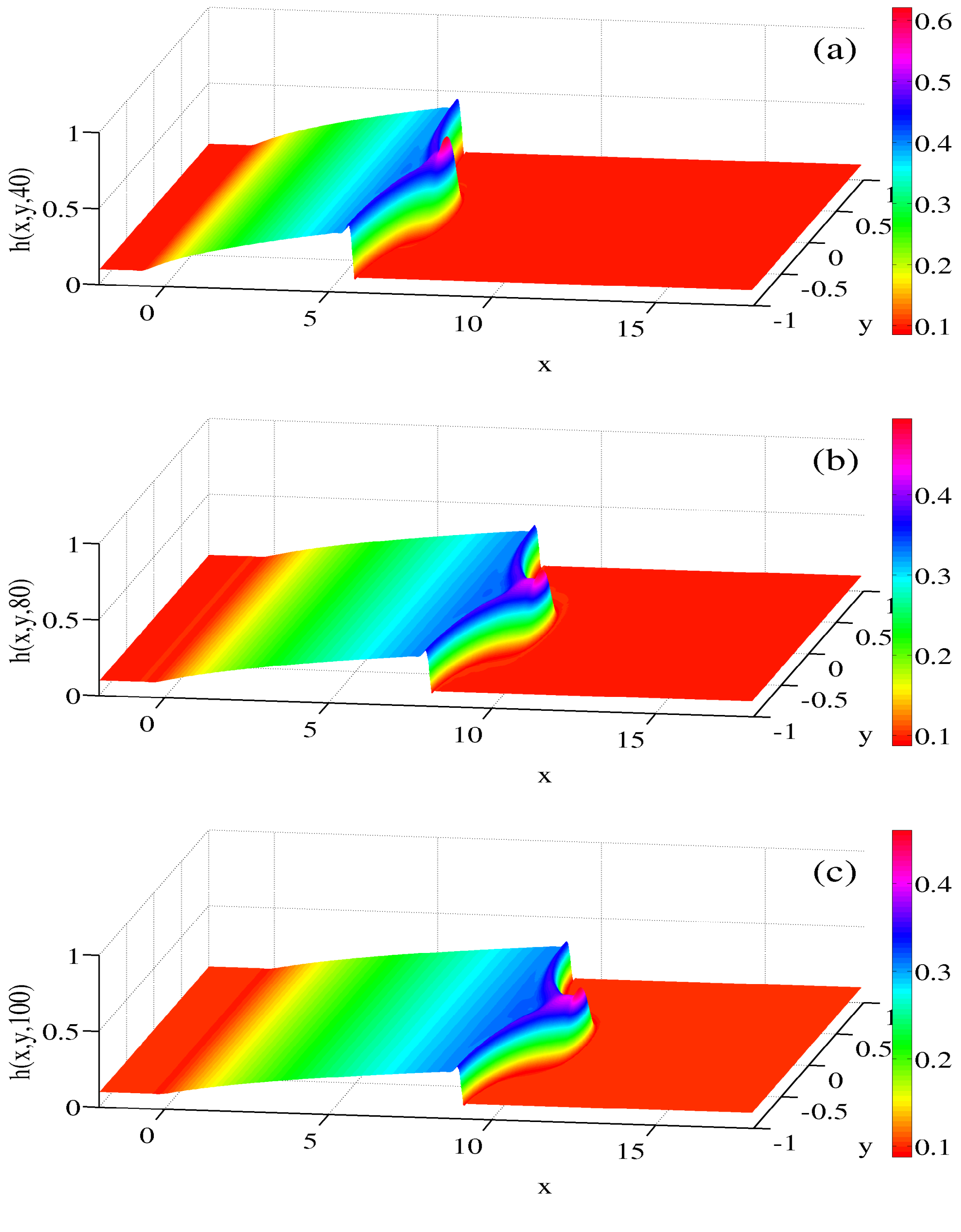

Figure 2: Surface plots of $h(x, y, t)$ (side view) showing its evolution in time (a) $t = 40$, (b) $t = 80$ and (c) $t = 100$ using the adaptive moving mesh scheme with $M_x = 400$, $N_y = 40$ (initial $\Delta x = \Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$ and $n = 2$.

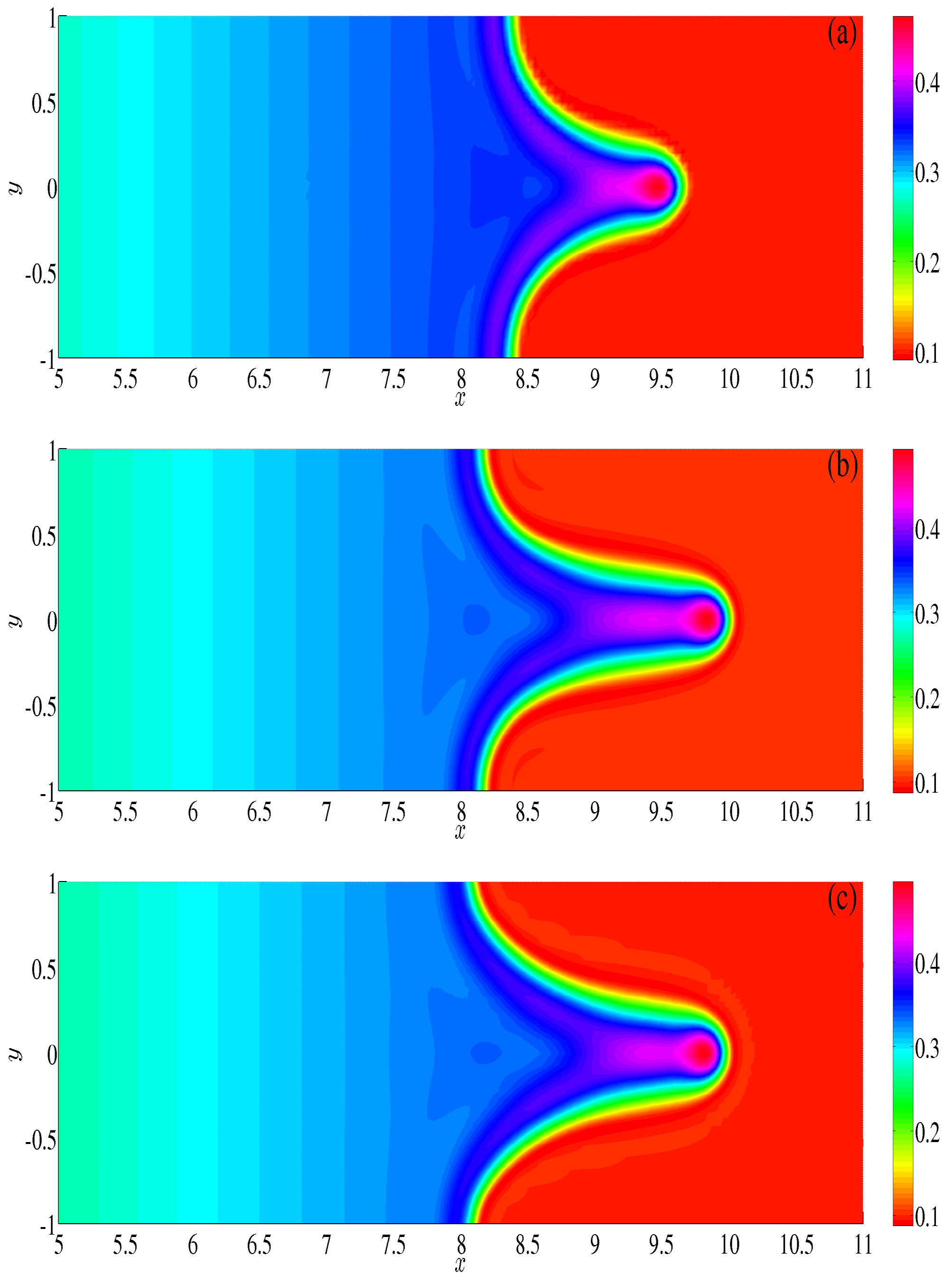

Figure 3: Surface plots of $h(x, y, t)$ (top view) at time $t = 80$ using (a) a uniform mesh with $M_x = 400$, $N_y = 40$ ($\Delta x = \Delta y = 0.05$), (b) a uniform mesh with $M_x = 2000$, $N_y = 200$ ($\Delta x = \Delta y = 0.01$) and (c) an adaptive moving mesh with $M_x = 400$, $N_y = 40$ (initial $\Delta x = \Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$ and $n = 2$.

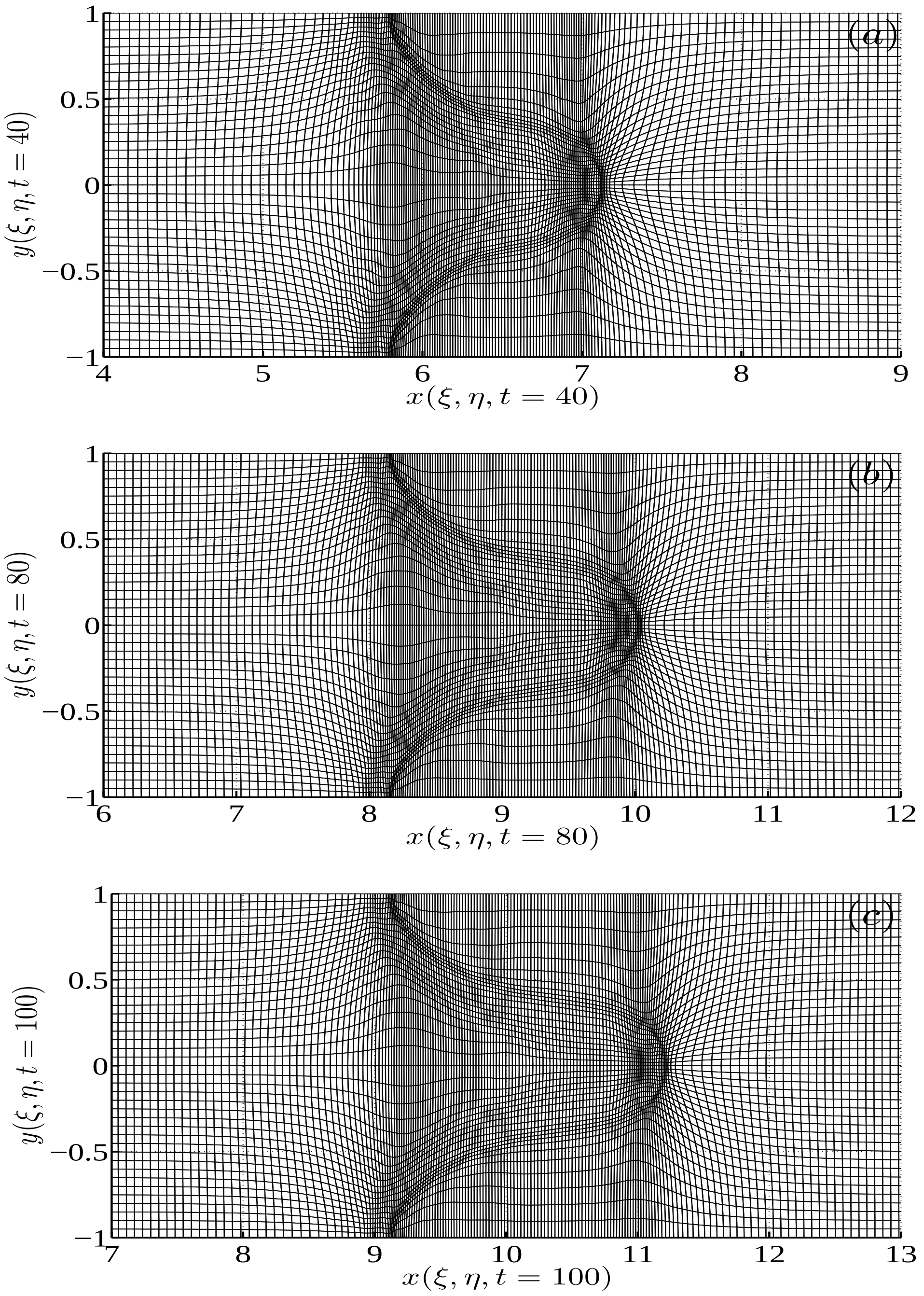

Figure 4: The adaptive moving mesh, $x(\xi, \eta, t)$ and $y(\xi, \eta, t)$, at time (a) $t = 40$, (b) $t = 80$ and (c) $t = 100$ using the adaptive moving mesh scheme with $M_x = 400$, $N_y = 40$ (initial $\Delta x = \Delta y = 0.05$), MMPDE4 with $\tau = 10^{-2}$ and curvature-based monitor function with $\alpha = 1$ and $n = 2$.

± 0.5), the length of the finger and W_{finger}, the width of the finger at a given location (arbitrarily chosen as $\xi = 8.5$). We calculate these quantities from the solution at a given time, $t = 80$ (when the finger has already developed). These quantities are calculated for four adaptive meshes (denoted by A in Table 1) with $M_x = 400$ and $N_y = 40$ (the initial $\Delta x = \Delta y = 0.05$), $M_x = 400$ and $N_y = 50$ (the initial $\Delta x = 0.05$ and $\Delta y = 0.04$), $M_x = 400$ and $N_y = 80$ (the initial $\Delta x = 0.05$ and $\Delta y = 0.025$) and $M_x = 400$ and $N_y = 100$ (the initial $\Delta x = 0.05$ and $\Delta y = 0.02$). These are compared against the corresponding values obtained using two fixed uniform mesh solutions (denoted by U in Table 1) with $M_x = 2000$ and $N_y = 200$ ($\Delta x = \Delta y = 0.01$) and $M_x = 400$ and $N_y = 40$ ($\Delta x = \Delta y = 0.05$). The L_2 norm error is calculated relative to the fixed uniform mesh solution with $M_x = 2000$ and $N_y = 200$ ($\Delta x = \Delta y = 0.01$). In calculating this error we interpolate the solution onto this fixed uniform mesh using linear interpolation. We observe that the mean error (relative to the uniform mesh with $M_x = 2000$ and $N_y = 200$, $\Delta x = \Delta y = 0.01$) for the adaptive mesh solutions using between $(16 - 40) \times 10^3$ elements is within 4.5% accuracy to that of this fixed uniform mesh which uses 4×10^5 elements. In contrast, the mean error for the solution obtained using a coarser fixed uniform mesh ($M_x = 400$ and $N_y = 40$, $\Delta x = \Delta y = 0.05$) with similar number of elements as the adaptive meshes is approximately 9%. The main contributor to this error

Mesh	M_x	N_y	% L_2 error	$h_{\max}$	x_{front}	L_{finger}	W_{finger}	CPU time
U	400	40	8.99	0.4720	9.75	1.25	2	1 day
	2000	200	-	0.4936	10.08	1.83	1.16	3 days
A	400	40	1.94	0.4941	10.05	1.81	1.04	2 hrs
	400	50	2.32	0.4918	10.02	1.76	1.02	2 hrs
	400	80	3.64	0.4892	10.01	1.68	1.01	7 hrs
	400	100	4.4	0.4891	10.01	1.66	1.01	16 hrs

Table 1: Metrics based on important characteristics of the solution obtained at time, $t = 80$ in order to determine the accuracy of the adaptive mesh solutions (denoted by A in the column titled "Mesh") against the corresponding fixed uniform mesh solution (denoted by U). See text for the description of each quantity.

is around the finger; the adaptive mesh adapts itself in this region both in the x and y directions (see Fig. 4) which reduces this error. In fact, the minimum $\Delta(x, y) = O(10^{-3})$ in this region for the adaptive meshes which is smaller than that of the uniform mesh ($\Delta(x, y) = 0.01$). We also note that the mean error relative to the uniform mesh increases as the adaptive mesh is refined which could suggest that the adaptive mesh solutions are more accurate than the fixed uniform mesh solutions. This is again due to the adaptivity of the meshes in reducing $\Delta(x, y)$ which makes the solution more accurate. We see a similar trend in the other metrics with the relative error being much smaller for these in comparison to the mean error. We also observe convergence in these quantities as the adaptive mesh is refined. However, the CPU time increases as the adaptive mesh is refined but is still much quicker compared to the fixed uniform mesh solution. For example, it took almost 3 days to compute the fixed uniform mesh (with $M_x = 2000$ and $N_y = 200$, $\Delta x = \Delta y = 0.01$) solution to time, $t = 80$ using MATLAB (Release 2013a, The MathWorks, Inc., Natick, Massachusetts, United States) on a computer with a 2.8GHz processor while it took between 2-16 hours to do the same calculation on the adaptive meshes.

6. Conclusions

In this paper, we have successfully applied the r-adaptive moving mesh method based on MMPDEs and mesh density functions to a prototype two-dimensional thin film flow problem with surface tension and involving a moving contact line. Our results show how the mesh adapts in both coordinate directions in order to align itself with the propagating finger with local refinement along the finger width (see Fig. 4). This adaptive moving mesh scheme is shown to be within 4.5% accuracy to that using a fixed uniform grid but with much fewer mesh points. In fact, by monitoring important characteristics of the solution behaviour, we observe the convergence to a more accurate solution compared to a fixed uniform mesh solution as the adaptive mesh is refined. One could increase the initial number of mesh points to increase the accuracy of the adaptive mesh solution but this would take more CPU time due to the additional mesh PDEs that need to be solved along with the underlying PDE. This difference between CPU times is not that large if the desired error is not too small and it is also much quicker compared to a fixed uniform mesh. For example, Table 1 shows a doubling of CPU time when increasing the initial number of points in the y direction from 80 to 100 (initial $\Delta y = 0.025$ and 0.02, respectively) although the solution appears to have converged. For this case, the solution corresponding to the adaptive mesh with $M_x = 400$ and $N_y = 80$ (the initial $\Delta x = 0.05$ and $\Delta y = 0.025$) could be considered reasonable in both accuracy as well as in computational time.

The structure of the locally refined mesh is also consistent with those obtained previously for the same test problem using h or hp-adaptive mesh refinement schemes for unstructured meshes [30, 31]. Although we have not made a quantitative comparison with these studies with respect to the accuracy of the solution, a visual inspection of their solution and meshes generated shows very similar qualitative features as ours. Moreover, the MMPDEs used here provide a simple framework to dynamically adapt and refine the mesh. This is in comparison to the seemingly more algorithmically complicated methods used by Sun *et al.* [30], who refine their mesh based on minimising an *a posteriori* error estimate, and Li *et al.* [31], who generate a hierarchy of multigrid meshes based on the gradient of the solution. The MMPDEs can be implemented in a straightforward way and solved simultaneously with the thin film equation quite efficiently using standard techniques for solving a system of parabolic partial differential equations by either the finite difference or the finite element method. In this respect, the r-adaptive method introduced here would hugely benefit the thin film flow research community. However, the CPU time taken by this method could be higher compared to the above two studies owing to the requirement of solving two additional equations for the mesh. This can be overcome by mesh generation methods within the r-adaptive framework such as the Monge-Ampère and Parabolic Monge-Ampère (PMA) equations (see Budd *et al.* [35, 36]) where there is only one equation for the mesh. These are currently being studied and will be reported in the future.

In conclusion, our results indicate great promise in terms of simplicity in its implementation and efficiency (in comparison to fixed uniform mesh schemes and possibly with h or hp-adaptive methods) for MMPDEs-based moving adaptive mesh methods to be applied on a regular basis in thin film flow problems. We note that the MMPDEs used at least in our test problem here did not result in mesh tangling or losss of mesh regularity. This does not, however, guarantee that these undesirable features will not appear in other thin film flow problems in general. We would need to further explore other mesh generation methods within the r-adaptive framework, for example, the so-called optimal transport equations, such as as the Monge-Ampère and Parabolic Monge-Ampère (PMA) equations (Budd *et al.* [35, 36]) which generate regular meshes and there is no mesh tangling. These would need to be tested on more challenging two-dimensional thin film problems that are prone to more dramatic dendritic fingering instabilities, for example, the fingering instabilities observed when a drop or sheet laden with surfactant spreads on a horizontal or inclined plane [9, 17, 39, 40], before its success can be guaranteed.

Acknowledgement

This work was a part of Abdulghani Alharbi's PhD research at Keele University. Abdulghani gratefully acknowledges financial support by the government of the Kingdom of Saudi Arabia and Taibah University.

References

[1] A. Oron, S. H. Davis, S. G. Bankoff, Long-scale evolution of thin liquid films, Rev. Mod. Phy. 69 (1997) 931–980.

[2] R. Craster, O. Matar, Dynamics and stability of thin liquid films, Rev. Mod. Phys. 81 (2009) 1131–1198.

[3] T. G. Myers, Thin films with high surface tension, SIAM Rev. 40 (3) (1998) 441–462.

[4] A. L. Bertozzi, The mathematics of moving contact lines in thin liquid films, Notices Amer. Math. Soc. 45 (6) (1998) 689 – 697.

[5] S. Troian, E. Herbolzheimer, S. Safran, Model for the fingering instability of the spreading surfactant drops, Phys. Rev. Lett. 65 (1990) 333–336.

[6] A. L. Bertozzi, M. P. Brenner, Linear stability and transient growth in driven contact lines, Phys. Fluids 9 (1997) 530–539.

[7] L. Kondic, Instabilities in gravity driven flow of thin fluid films, SIAM Rev. 45 (1) (2003) 95–115.

[8] M. R. E. Warner, R. V. Craster, O. K. Matar, Fingering phenomena associated with insoluble surfactant spreading on thin liquid films, Fluid Mech. 510 (2004) 169–200.

[9] B. Edmonstone, O. Matar, R. Craster, Surfactant-induced fingering phenomena in thin film flow down an inclined plane, Physica D: Nonlinear Phenomena 209 (2005) 62–79.

[10] O. E. Jensen, S. Naire, The spreading and stability of a surfactant-laden drop on a prewetted substrate, J. Fluid Mech. 554 (2006) 5–24.

[11] J. V. Goddard, S. Naire, The spreading and stability of a surfactant-laden drop on an inclined prewetted substrate, J. Fluid Mech. 772 (2015) 535–568.

[12] F. Bernis, Viscous flows, fourth order nonlinear degenerate parabolic equations and singular elliptic problems, in: Free boundary problems: theory and application, Vol. 323, Pitman Research Notes in Mathematics, 1995, pp. 40–56.

[13] L. Kondic, J. Diez, Pattern formation in the flow of thin films down an incline: Constant flux configuration, J. Comp. Phys. 13 (11) (2001) 3168–3184.

[14] J. A. Diez, L. Kondic, Computing three-dimensional thin film flows including contact lines, Journal of Computational Physics 183 (2002) 274–306.

[15] T. Witelski, M. Bowen, ADI schemes for higher-order nonlinear diffusion equations, Applied Numerical Mathematics 45 (2003) 331351.

[16] M. R. E. Warner, R. V. Craster, O. K. Matar, Fingering phenomena created by a soluble surfactant deposition on a thin liquid film, Phys. Fluids 16 (2004) 2933–2951.

[17] B. D. Edmonstone, O. K. Matar, R. V. Craster, Flow of surfactant-laden thin films down an inclined plane, J. Engrg. Math. 50 (2004) 141–156.

[18] B. Edmonstone, R. Craster, O. Matar, Surfactant-induced fingering phenomena beyond the critical micelle concentration, Fluid Mech. 564 (2006) 105–138.

[19] R. Levy, M. Shearer, The motion of a thin liquid film driven by surfactant and gravity., SIAM J. Appl. Math. 66 (5) (2006) 1588–1609.

[20] R. Levy, M. Shearer, T. P. Witelski, Gravity-driven thin liquid films with insoluble surfactant: smooth traveling waves, Eur. J. Appl. Math. 18 (2007) 679–708.

[21] A. Mavromoustaki, O. Matar, R. Craster, Dynamics of a climbing surfactant-laden film II: Stability, J. Colloid Interface Sci. 371 (2012) 121–135.

[22] J. Barrett, J. Blowey, H. Garcke, Finite element approximation of a fourth order degenerate parabolic equation, Numer. Math. 80 (1998) 525–556.

[23] G. Grün, M. Rumpf, Nonnegativity preserving convergent schemes for the thin film equation, Numer. Math. 87 (2000) 113–152.

[24] A. Heryudono, R. Braun, T. Driscoll, K. Maki, L. Cook, P. King-Smith, Single-equation models for the tear film in a blink cycle: realistic lid motion, Math. Med. Biol. 4 (2007) 347–377.

[25] P. Keast, P. H. Muir, Algorithm 688: EPDCOL: A more efficient PDECOL code, ACM Trans. Math. Softw. 17 (2) (1991) 153–166.

[26] J. Verwer, J. Blom, J. M. Sanz-Serna, An adaptive moving grid method for one-dimensional systems of partial differential equations, J. Comp. Phys. 82 (1989) 454–486.

[27] J. G. Verwer, J. G. Blom, R. M. Furzeland, P. A. Zegeling, A moving grid method for one-dimensional PDEs based on the method of lines, SIAM Philadelphia, 1989, Ch. 12, pp. 160–175.

[28] R. Furzeland, J. Verwer, P. A. Zegeling, A numerical study of three moving grid methods for one-dimensional partial differential equations which are based on the method of lines, J. Comp. Phys. 89 (1990) 349–388.

[29] J. Blom, P. Zegeling, Algorithm 731: A moving-grid interface for systems of one-dimensional partial differential equations, ACM Trans. Math. Softw. 20 (1994) 194–214.

[30] P. Sun, R. D. Russell, J. Xu, A new adaptive local mesh refinement algorithm and its application on fourth order thin film flow problem, J. Comp. Phys. 224 (2007) 1021–1048.

[31] Y. Li, D. Jeong, J. Kim, Adaptive mesh refinement for simulation of thin film flows, Meccanica 49 (2013) 239–252.

[32] A. Alharbi, S. Naire, An adaptive moving mesh method for thin film flow equations with surface tension, J. Comp. Appl. Maths. 319 (2017) 365–384.

[33] W. Huang, R. D. Russell, *Adaptive Moving Mesh Methods*, Springer, 2011.

[34] C. J. Budd, W. Huang, R. D. Russell, Adaptivity with moving grids, Acta Numerica 18 (2009) 111–241.

[35] C. J. Budd, J. F. Williams, Moving mesh generation using the parabolic mongeampre equation, SIAM journal on scientic computing 31 (2009) 3438–3465.

[36] C. Budd, M. Cullen, E. Walsh, Monge-ampère based moving mesh methods for numerical weather prediction, with applications to the Eady problem, J. of Comp. Phys. 236 (2013) 247–270.

[37] G. Beckett, J. Mackenzie, Convergence analysis of finite difference approximations on equidistributed grids to a singularly perturbed boundary value problem, Appl. Num. Math. 35 (2000) 87–109.

[38] W. Huang, R. D. Russell, Analysis of moving mesh partial differential equations with spatial smoothing, J. Comp. Phys. 34 (1997) 1106–1126.

[39] B. Edmonstone, O. Matar, R. Craster, Coating of an inclined plane in the presence of insoluble surfactant, J. Colloid Interface Sci. 287 (2005) 261–272.

[40] B. Edmonstone, O. Matar, R. Craster, A note on the coating of an inclined plane in the presence of soluble surfactant, J. Colloid Interface Sci. 293 (2006) 222–229.

[41] W. Huang, Y. Ren, R. D. Russell, Moving mesh methods based on moving mesh partial differential equations, Comp. Phys. 113 (1994) 279–290.

[42] P. Brown, A. Hindmarsh, L. Petzold, Using Krylov methods in the solution of large-scale differential- algebraic systems, SIAM J. Sci. Comp. 15 (1994) 1467–1488.

Appendix A. Finite difference discretisation and smoothing scheme

The finite-difference discretisation scheme used for each of the terms appearing in Eqs. (13)-(19) are:

$$[h_x]_{j,k} = \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_x)_{j+\frac{1}{2},k}\, h_{j+\frac{1}{2},k} - (J\xi_x)_{j-\frac{1}{2},k}\, h_{j-\frac{1}{2},k}\right) + \frac{1}{\Delta\eta}\left((J\eta_x)_{j,k+\frac{1}{2}}\, h_{j,k+\frac{1}{2}} - (J\eta_x)_{j,k-\frac{1}{2}}\, h_{j,k-\frac{1}{2}}\right)\right], \tag{A.1}$$

$$[h_y]_{j,k} = \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_y)_{j+\frac{1}{2},k}\, h_{j+\frac{1}{2},k} - (J\xi_y)_{j-\frac{1}{2},k}\, h_{j-\frac{1}{2},k}\right) + \frac{1}{\Delta\eta}\left((J\eta_y)_{j,k+\frac{1}{2}}\, h_{j,k+\frac{1}{2}} - (J\eta_y)_{j,k-\frac{1}{2}}\, h_{j,k-\frac{1}{2}}\right)\right], \tag{A.2}$$

$$\left[h^3_x\right]_{j,k} = \frac{1}{J}\left[\frac{1}{\Delta\xi}\left((J\xi_x)_{j+\frac{1}{2},k}\left(h^3\right)_{j+\frac{1}{2},k} - (J\xi_x)_{j-\frac{1}{2},k}\left(h^3\right)_{j-\frac{1}{2},k}\right) + \frac{1}{\Delta\eta}\left((J\eta_x)_{j,k+\frac{1}{2}}\left(h^3\right)_{j,k+\frac{1}{2}} - (J\eta_x)_{j,k-\frac{1}{2}}\left(h^3\right)_{j,k-\frac{1}{2}}\right)\right], \tag{A.3}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}h\right]_{j,k} = \frac{1}{J}\left[\frac{(h^3R)_{j,k} - (h^3R)_{j-1,k}}{\Delta\xi} + \frac{(h^3S)_{j,k} - (h^3S)_{j,k-1}}{\Delta\eta}\right], \tag{A.4}$$

$$G_{j,k} = (\boldsymbol{\nabla}^2 h)_{j,k} = \frac{1}{J}\left[\frac{R_{j,k} - R_{j-1,k}}{\Delta\xi} + \frac{S_{j,k} - S_{j,k-1}}{\Delta\eta}\right], \tag{A.5}$$

$$\boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}\boldsymbol{\nabla}^2 h\right] = \boldsymbol{\nabla}\cdot\left[h^3\boldsymbol{\nabla}G\right]_{j,k} = \frac{1}{J}\left[\frac{(h^3R_1)_{j,k} - (h^3R_1)_{j-1,k}}{\Delta\xi} + \frac{(h^3S_1)_{j,k} - (h^3S_1)_{j,k-1}}{\Delta\eta}\right], \tag{A.6}$$

$$R_{j,k} = \left[a_{j+\frac{1}{2},k}\frac{h_{j+1,k} - h_{j,k}}{\Delta\xi} + c_{j+\frac{1}{2},k}\frac{h_{j,k+1} - h_{j,k}}{\Delta\eta}\right], \tag{A.7}$$

$$S_{j,k} = \left[q_{j,k+\frac{1}{2}}\frac{h_{j,k+1} - h_{j,k}}{\Delta\eta} + c_{j,k+\frac{1}{2}}\frac{h_{j+1,k} - h_{j,k}}{\Delta\xi}\right], \tag{A.8}$$

$$R_{1j,k} = \left[a_{j+\frac{1}{2},k}\frac{G_{j+1,k} - G_{j,k}}{\Delta\xi} + c_{j+\frac{1}{2},k}\frac{G_{j,k+1} - G_{j,k}}{\Delta\eta}\right], \tag{A.9}$$

$$S_{1j,k} = \left[q_{j,k+\frac{1}{2}}\frac{G_{j,k+1} - G_{j,k}}{\Delta\eta} + c_{j,k+\frac{1}{2}}\frac{G_{j+1,k} - G_{j,k}}{\Delta\xi}\right], \tag{A.10}$$

$$a_{j,k} := \begin{cases} \frac{1}{J_{j,k}} \left((J\xi_x)^2 + (J\xi_y)^2 \right)_{j,k}, & \text{for } j = 2, \dots, M_x, \\ \frac{1}{J_{j,k}} (J\xi_x)^2_{j,k}, & \text{for } j = 1, \\ \frac{1}{J_{j,k}} (J\xi_x)^2_{j,k}, & \text{for } j = M_x + 1, \end{cases} \tag{A.11}$$

$$q_{j,k} := \begin{cases} \frac{1}{J_{j,k}} \left((J\eta_y)^2 + (J\eta_x)^2 \right)_{j,k}, & \text{for } k = 2, \dots, N_y, \\ \frac{1}{J_{j,k}} (J\eta_y)^2_{j,k}, & \text{for } k = 1, \\ \frac{1}{J_{j,k}} (J\eta_y)^2_{j,k}, & \text{for } k = N_y + 1, \end{cases} \tag{A.12}$$

$$c_{j,k} = \frac{1}{J_{j,k}} \left((J\eta_y)(J\xi_y) + (J\eta_x)(J\xi_x) \right)_{j,k}, \text{for } j = 2, \dots, M_x, k = 2, \dots, N_y, \tag{A.13}$$

$$J_{j,k} := \begin{cases} \left((J\xi_x)(J\eta_y) - (J\xi_y)(J\eta_x) \right)_{j,k}, & \text{for } j = 2, \dots, M_x, k = 2, \dots, N_y, \\ (J\xi_x)_{j,k} (J\eta_y)_{j,k}, & \text{for } j, k = 1 \\ (J\xi_x)_{j,k} (J\eta_y)_{j,k}, & \text{for } j = M_x + 1, k = N_y + 1. \end{cases} \tag{A.14}$$

$$(J\xi_x)_{j,k} = (y_\eta)_{j,k} = \frac{1}{2\Delta\eta} (y_{j,k+1} - y_{j,k-1}), \tag{A.15}$$

$$(J\xi_y)_{j,k} = -(x_\eta)_{j,k} = -\frac{1}{2\Delta\eta} (x_{j,k+1} - x_{j,k-1}), \tag{A.16}$$

$$(J\eta_x)_{j,k} = -(y_\xi)_{j,k} = -\frac{1}{2\Delta\xi} (y_{j+1,k} - y_{j-1,k}), \tag{A.17}$$

$$(J\eta_y)_{j,k} = (x_\xi)_{j,k} = +\frac{1}{2\Delta\xi} (x_{j+1,k} - x_{j-1,k}), \tag{A.18}$$

Some of the above approximations involve evaluations at half mesh points which are obtained as an average of the neighbouring mesh points. The expressions in Eqs. (A.11-A.14) are simplified along the boundaries using the mesh derivative boundary conditions given in Eq. (4). Evaluating the above at the boundaries require fictitious points which are obtained by discretising the boundary conditions in Eq. (10). In some cases, we had to use one-sided finite differences to evaluate a particular quantity at the boundary which was obtained using a Taylor's series approximation there.

The semi-discretisation scheme for MMPDE5 given in Eq. (3) is given by

$$\begin{aligned}\text{MMPDE5}: \ \tau\Bigg[\boldsymbol{x}_t - \gamma_1 \Bigg(&\frac{1}{\Delta\xi^2}\left(\Delta_j \boldsymbol{x}_{t,j,k} - \Delta_j \boldsymbol{x}_{t,j-1,k}\right) \\ &- \frac{1}{\Delta\eta^2}\left(\Delta_k \boldsymbol{x}_{t,j,k} - \Delta_k \boldsymbol{x}_{t,j,k-1}\right)\Bigg)\Bigg] = E_{j,k}, \qquad \text{(A.19)} \\ &\forall\, j = 2, \ldots, M_x, \quad k = 2, \ldots, N_y, \qquad \text{(A.20)}\end{aligned}$$

where the quantities involved are defined in the main text.

To obtain a smoother mesh and also make the MMPDEs easier to integrate, it is common practice in the context of moving mesh methods to smooth the mesh density function. A simple but effective smoothing scheme for a two dimensional rectangular computational mesh suggested by Huang [33, 41] is based on weighted averaging,

$$\begin{aligned}\hat{\rho}_{j,k} := &\frac{\sum_{i=\max(1,j-p)}^{\min(M_x+1,j+p)} \sum_{l=\max(1,k-p)}^{\min(N_y+1,k+p)} \hat{\rho}_{i,l}\gamma^{|j-i|+|k-l|}}{\sum_{i=\max(1,j-p)}^{\min(M_x+1,j+p} \sum_{l=\max(1,k-p)}^{\min(N_y+1,k+p)} \gamma^{|j-i|+|k-l|}}, \\ &j = 1, \ldots, M_x + 1,\ k = 1, \ldots, N_y + 1, \qquad \text{(A.21)}\end{aligned}$$

where p is a non-negative integer called the smoothing index and $\gamma \in (0, 1)$ is a smoothing parameter. Several sweeps of the scheme may be applied at each integration step.

Revision notes

We thank both referees for their careful reading of the manuscript and very useful comments. We have addressed all the comments/issues raised by the referees and incorporated their suggested corrections. This has resulted in revisions to the manuscript which are detailed below. Any page numbers below are with reference to the revised manuscript.

We hope that the referees are satisfied by these revisions and recommend the paper for publication.

Referee 1

Issue 1: *Statements made such as "much easier to implement" and "more accurate than h or hp refinement methods" need to be more specific and cannot be made without any evidence supporting this claim.*

Response: We fully agree with this. Any mention of this has now been removed from the manuscript except in the second paragraph in the section on Conclusions (Pg. 18). In this paragraph we make a qualitative comparison specifically for parabolic PDEs (such as the thin film equation studied here) between ours and the 2 related thin film flow studies of Sun et al. and Li et al. The comparison is with respect to the solution accuracy, ease of implementation and CPU time taken.

Issue 2: *Accuracy of method and computational time versus mesh size (number of points).*

Response: We have now provided more in-depth analysis on the accuracy and CPU time for varying mesh sizes. These include 4 adaptive meshes with varying number of points in the y direction (or varying Delta y). These changes are incorporated in Pgs. 16 and 17 and in the first paragraph in the section on Conclusions (Pgs. 17 and 18).

Issue 3: *How this method behaves on other test cases?*

Response: The focus of this study was meant to be on testing the numerical method on one problem which is a standard prototype in thin film spreading flows. We deliberately did not consider other test problems (as done in our earlier 1-D paper) since this would lose focus on the main problem and make the paper unnecessarily long. Work is currently underway on testing this method on other challenging problems (mentioned at the end of the Conclusions section) and will be reported elsewhere.

Issue 4: *Other metrics for accuracy and computational time versus mesh size (or number of points).*

Response: We thank the referee for this very helpful suggestion. We have now considered other metrics connected to the solution's characteristics such as the maximum height at the front, the location of the front, the length and width of the finger and the computational time for varying mesh sizes. These changes are incorporated in Pgs. 16 and 17 (including Table 1) and in the first paragraph in the section on Conclusions (Pgs. 17 and 18).

Issue 5: *Shorten section 4.*

Response: Section 4 has now been shortened and some of the details of the discretisation have been moved to Appendix A.

Referee 2

Issue 1: *Just state equations for MMPDE5 but not details of discretisation*.

Response: We have now deleted the discretisation of MMPDE5.

Issue 2: *It would help to define what steady states the MMPDEs relax to if rho is independent of time.*

Response: This is now defined on Pg. 4.

Issue 3: *What do you mean by regular meshes?*

Response: By regular meshes we mean those that do not have elements with very small aspect ratio. We have added a sentence stating this in the first paragraph on Pg. 5.

Issue 4: *The time discretisation is not stated.*

Response: We follow the Method of Lines to solve the equations. Only the spatial variables are discretised and the time derivative is left continuous resulting in a system of ODEs. The solver DASPK then solves the system of ODEs using built-in Backward Differentiation Formulas (BDF) to approximate the time derivative. A sentence stating this is now included on Pg. 10.