A Numerical Scheme for Particle-Laden Thin Film Flow in Two Dimensions

Matthew R. Mata[a,*], Andrea L. Bertozzi[a]

[a]*Department of Mathematics, University of California Los Angeles, 520 Portola Plaza, Los Angeles, California, 90095-1555*

Abstract

The physics of particle-laden thin film flow is not fully understood, and recent experiments have raised questions with current theory. There is a need for fully two-dimensional simulations to compare with experimental data. To this end, a numerical scheme is presented for a lubrication model derived for particle-laden thin film flow in two dimensions with surface tension. The scheme relies on an ADI process to handle the higher-order terms, and an iterative procedure to improve the solution at each timestep. This is the first paper to simulate the two-dimensional particle-laden thin film lubrication model. Several aspects of the scheme are examined for a test problem, such as the timestep, runtime, and number of iterations. The results from the simulation are compared to experimental data. The simulation shows good qualitative agreement. It also suggests further lines of inquiry for the physical model.

Keywords: adaptive timestepping, alternating direction implicit, coupled system, particle-laden, surface tension, thin film

1. Introduction

In recent years, the problem of numerically solving gravity-driven thin film flow for clear fluids has had ample work done in both one and two dimensions. However, the case when the film contains particles suspended within it has received less attention, especially in two dimensions. The evolution of a clear fluid down an inclined plane is modeled using a single partial differential equation and numerical schemes have been derived using finite differences [9, 16] and finite elements [32]. For similar equations, such as spreading thin films, there are methods for finite elements in one dimension [10, 11, 37] and for finite differences in two dimensions [35]. The incorporation of particles into such a flow leads to another variable in the model, namely the particle concentration, and an accompanying equation related to the evolution of the particles. The result is a system of equations that requires a different approach from the clear fluid case to formulate a practical numerical scheme, due to the coupling of the equations.

An active area of research in the last decade has been the development of numerical methods for higher-order thin film equations including complex fluids described by systems of equations. Related problems include methods for coupled systems of nonlinear parabolic equations [22, 26].

[*] Principal corresponding author

Email addresses: matthewmata@math.ucla.edu (Matthew R. Mata), bertozzi@math.ucla.edu (Andrea L. Bertozzi)

The scheme presented here is, in part, inspired by recent models for surfactants [34] and thin films [35]. We choose an Alternating Direction Implicit (ADI) scheme as a tractable method for implicit timesteps, because surface tension introduces a severe restriction on the timestep in the case of explicit schemes. This ADI approach also allows for an implicit scheme while avoiding to have to solve the large sparse linear algebra problems by an iterative method, such as GMRES, that result from linearizing the two-dimensional operators in Newton's method [35]. ADI is also amenable to parallelization. While ADI schemes for numerically solving parabolic equations date back to the 1950's [27], their use in higher-order problems is rather new, e.g., [35], and not all that well-studied. However, the ease of parallelization makes such schemes a viable choice for multiprocessor platforms. Since their inception, ADI schemes have been extended to handle parabolic problems with mixed derivative terms [2, 8, 24, 30], variable coefficients [15, 35], and high-order terms [35].

The ideas present in these schemes can be combined to create an efficient way to numerically solve the particle-laden thin film flow equations. The nonlinearity and higher-order terms are handled in a similar manner to Witelski and Bowen [35], which dealt with thin film equations, and the remaining terms are treated as in Warner et al. [34], which devised a semi-implicit scheme for surfactants. This combined approach is fine-tuned to draw out better efficiency, via adaptive timestepping and an iterative procedure within each timestep. At the cost of the extra calculations due to the iterative nature of the scheme, the timestep needed for stability can be improved over recent methods. The result is an efficient method to simulate the continuum model in two dimensions.

The full physics of particle-laden thin film flow is not well understood. Recent experiments, and their comparison to the model, have raised questions. We present such a comparison in this paper, where the results show qualitative agreement. In particular, by performing two-dimensional simulations, we are able to observe finger formation and compare directly with experiments. The development of quantitatively correct models for these systems is an ongoing active area of research. Thus, there is a need for accurate, fully two-dimensional simulations of the model, such as in the case of mudslides and oil spills.

The paper is organized as follows: Section 2 presents the system of evolution equations for the flow. In Section 3, the numerical scheme for this system is derived. Section 4 covers the adaptive timestepping scheme implemented in the code. A complete explanation of the spatial discretization is given in Section 5. The practicality and implementation of a moving reference frame in the simulations are discussed in Section 6. Numerical simulations are presented in Section 7. We compare the results generated from the numerical scheme to an experiment using silicone oil and glass beads in Section 8. Finally, in Section 9, we provide a discussion of the results and future work.

2. Model

The results from experiments indicate that particle-laden thin film flows exhibit three distinct regimes, based on the initial particle concentration and angle of inclination [36]. For low concentrations and angles, the particles settle to the substrate with clear fluid flowing over the top. The behavior after sedimentation is similar to clear fluid experiments, such as those performed by Huppert [14]. High concentrations and angles cause a particle-rich ridge to emerge at the front of the flow. Medium concentrations and angles lead to a particle concentration which appears to stay well-mixed throughout the duration of the experiment. Based on Cook [5], this behavior likely belongs to one of the two previously mentioned regimes, but may not have evolved to the point where this distinction can be made.

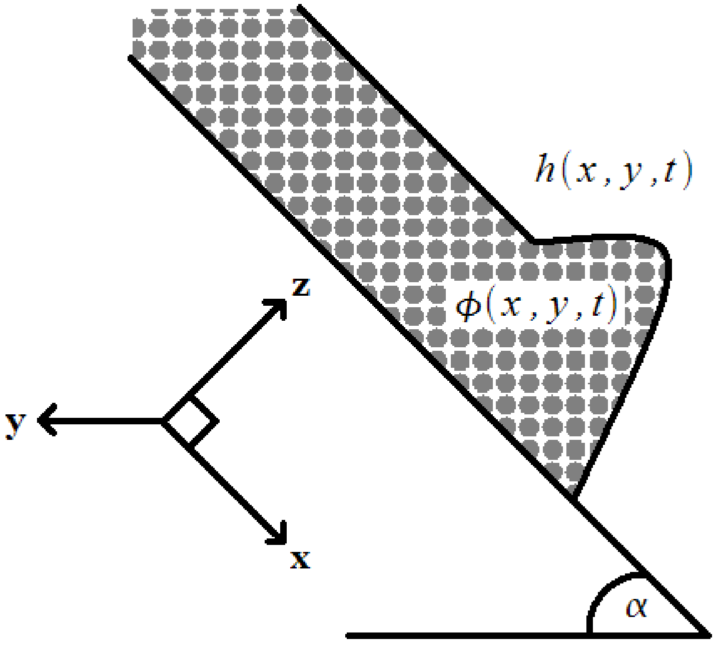

Figure 1: The coordinate system and variables considered in this problem. **x** is in the plane, in the direction of the flow; **y** is in the plane, perpendicular to **x**; and **z** is normal to the plane. h is the film thickness and ϕ is the particle concentration.

The evolution equations for the flow are based on the regime where the inclination angle and particle concentration are both high enough to induce the formation of a particle-rich ridge. The equations are formulated in terms of the thickness of the film, h, and the particle concentration by volume, ϕ (see Figure 1). The equations for modeling this regime were first derived in Zhou et al. [36]; re-derived in Cook et al. [6], using conservation of volume rather than mass; and modified in Cook et al. [7], adding in a shear-induced diffusion term to correct for an instability affecting ϕ. The dimensionless system [7] is

$$h_t + \nabla \cdot (h\mathbf{v}_{\text{av}}) = 0, \tag{1}$$

$$(\phi h)_t + \nabla \cdot [\phi h (\mathbf{v}_{\text{av}} + (1-\phi)\mathbf{v}_{\text{rel}}) - \mathbf{F}_{\text{diff}}] = 0. \tag{2}$$

The orientation for (1)-(2) is such that **x** lies in the plane and is parallel to the direction of the flow, **y** is across the plane and perpendicular to **x**, and **z** is normal to the plane. A one-dimensional form of the problem considers only the x-direction, while two dimensions includes both x and y. The two velocity terms, $\mathbf{v}_{\text{av}}$ and $\mathbf{v}_{\text{rel}}$, are the volume-averaged velocity of the fluid and the velocity of the particles relative to the liquid, respectively. We use the term liquid to refer to the substance that the particles are suspended in and fluid to refer to the mixture as a whole. In Equation (2), $\mathbf{v}_{\text{av}} + (1-\phi)\mathbf{v}_{\text{rel}}$ is the individual velocity of the particles [6] and $\mathbf{F}_{\text{diff}}$ is shear-induced diffusion of the particles.

The volume-averaged velocity of the liquid and the particles together is

$$\mathbf{v}_{\text{av}} = \frac{h^2}{\mu(\phi)}\nabla\nabla^2 h - D(\alpha)\left[\frac{h^2}{\mu(\phi)}\nabla\left(\rho(\phi)h\right) - \frac{5}{8}\frac{h^3}{\mu(\phi)}\nabla\left(\rho(\phi)\right)\right] + \frac{\rho(\phi)}{\mu(\phi)}h^2\hat{\mathbf{x}}, \tag{3}$$

where the terms in (3) come from surface tension, the effects of gravity normal to the inclined plane, and the effects of gravity parallel to the inclined plane.

The density of the fluid as a whole is $\rho(\phi) = 1 + \rho_f\phi$; $\rho_f = \frac{\rho_p - \rho_l}{\rho_l}$ is the difference in the densities between the particles and the liquid. The function $\mu(\phi) = (1 - \phi/\phi_{\max})^{-2}$ [18, 31] is the effective fluid viscosity, where $\phi_{\max}$ is the maximum packing fraction of particles, assuming the particles are spheres. For this problem, the maximum packing fraction has been empirically determined to be 0.58, while the theoretical value is 0.64 [33]. $D(\alpha) = (3Ca)^{1/3}\cot\alpha$ [3] is a modified capillary number, where Ca is the capillary number of the liquid and α is the angle of inclination of the plane on which the fluid is flowing ($\alpha = 0$ corresponds to the plane being horizontal while $\alpha = \pi/2$ to vertical).

The settling velocity of the particles, relative to the velocity of the liquid, is a combination of three factors, assumed to be multiplicative,

$$\mathbf{v}_{\text{rel}} = V_s f(\phi) w(h)\hat{\mathbf{x}}. \tag{4}$$

The coefficient $V_s = \frac{2}{3}a^2\rho_f$ in (4) is the Stokes settling velocity of a single sphere settling in a viscous liquid, where a is the dimensionless particle radius. A hindered settling function, in this case the Richardson-Zaki function $f(\phi) = (1-\phi)^5$ [29], accounts for the effect of sedimentation. The particles settling parallel to the substrate is modeled using a wall effects function, $w(h) = A(h/a)^2/\sqrt{1 + (A(h/a)^2)^2}$ with $A = 1/18$. This function is an approximation to a method of images solution to a single sphere falling parallel to a vertical wall [13]. This has the property that it is near 0 for h small and near 1 for h large.

Since the system (1)-(2) is fourth-order and (3) contains higher-order terms but (4) does not, $\mathbf{v}_{\text{rel}}$ is not regularized. This leads to an instability affecting the particle concentration in numerical simulations [7]. To correct for this, a shear-induced diffusion term (5) was added in,

$$\mathbf{F}_{\text{diff}} = \frac{3}{2}a^2(3Ca)^{1/3}\hat{D}(\phi)\frac{h^2\rho(\phi)}{\mu(\phi)}\nabla\phi. \tag{5}$$

This behavior can be seen in a one-dimensional example on the domain $x : 0{-}50$ with $\Delta x = 0.05$. The initial film thickness is a jump, from 1 to 0.05 at $x = 25$, smoothed by hyperbolic tangent. The initial particle concentration is taken to be $\phi = 0.3$. This simulation is similar to those described in Section 7, and a moving reference frame is used, as discussed in Section 6. By time $t = 1000$, the solution without the extra diffusion term has developed an instability near $x = 10$ (Figure 2) while the one with it is still stable (Figure 3). Note that the oscillations trailing the particle-rich ridge, between $x = 0$ and $x = 10$ are a result of the discretization of the moving reference frame and are discussed in Section 6.

Equation (5) accounts for horizontal diffusion of particles in the fluid caused by horizontal gradients of ϕ and was derived based on results from Leighton [20] and Leighton and Acrivos [21]. The term $\hat{D}(\phi) = (1/3)\phi^2\left(1 + (1/2)e^{8.8\phi}\right)$ is a dimensionless diffusion coefficient.

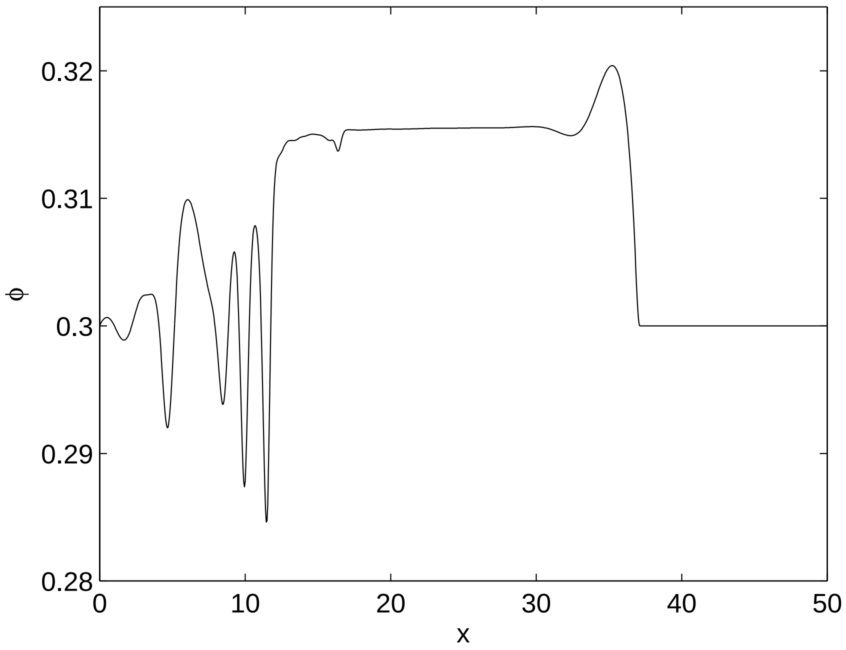

Figure 2: The numerical solution of ϕ at time $t = 1000$ without shear-induced diffusion. By this time, an instability has developed near $x = 10$.

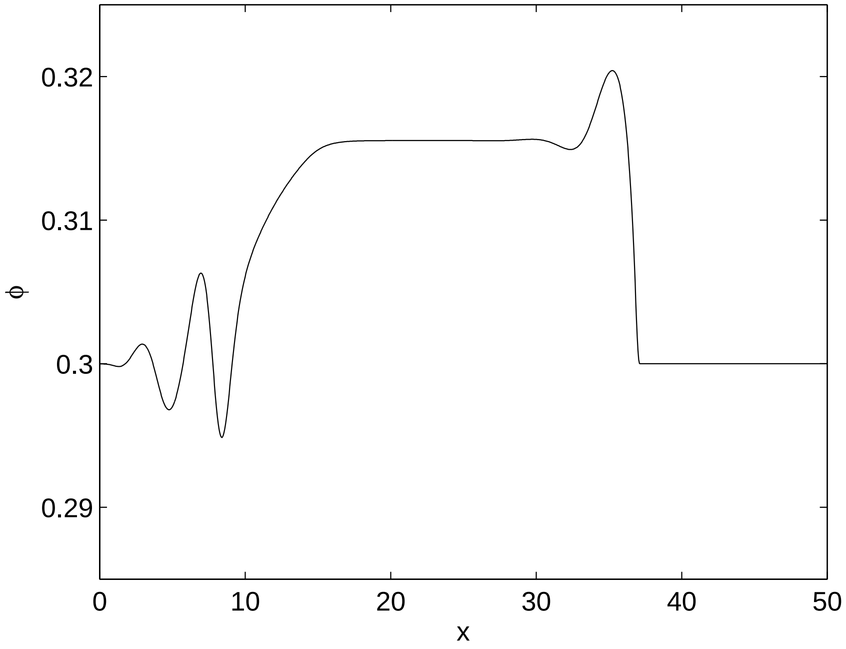

Figure 3: The numerical solution ϕ at time $t = 1000$ with shear-induced diffusion (5). The solution is still stable due to the extra term.

3. Numerical Scheme

In the case of a gravity-driven clear fluid flow, the model reduces to a single equation [3] for the film thickness, h,

$$h_t + (h^3)_x + \nabla \cdot \big(h^3 \nabla \nabla^2 h - D(\alpha) h^3 \nabla h\big) = 0. \tag{6}$$

Solving (6), and similar problems, numerically in one and two dimensions has been performed using several different methods [1, 9, 16, 23, 32, 35]. Including particles in the physics not only adds a second equation, but couples it to the equation for the film thickness. The particle-laden case has been solved numerically in one dimension with methods such as forward Euler with upwind differencing [36] and the Lax-Friedrichs method [6] when the high-order terms are omitted, and backward Euler with centered differencing [36] when the terms are included.

This system of PDEs in two dimensions poses numerical difficulties beyond those present in the clear fluid problem. For both the clear and particle-laden cases, fully explicit schemes typically have the problem that an $O(\Delta x^4)$ timestep, assuming $\Delta x = \Delta y$, is needed for stability. One solution is to use an implicit scheme. For the clear fluid and similar problems, the nonlinearity combined with an implicit scheme amounts to solving the problem at each timestep using an iterative process, such as Newton's method, to converge to the solution [35]. For the particle-laden case, using an implicit scheme typically requires that both equations be solved simultaneously, using an iterative process to account for the nonlinearity. This results in a linear algebra problem with twice the number of unknowns and a matrix that is twice as large in each dimension, compared to the clear fluid problem. Therefore, solving the particle-laden case leads to larger linear algebra problems to solve at each timestep and the matrix from Newton's method will have a more complex structure than for clear fluids.

The goal of the scheme presented here is to circumvent some of the aforementioned difficulties. The advantages of this approach, over a purely explicit scheme or implicit with Newton's method, is that the timestep is more lenient than for a fully explicit scheme and the linear algebra problem that results from the implicit part of the scheme is reduced to a series of smaller banded matrix solves, which can be done efficiently and independently for each equation.

The numerical scheme that we employ for the particle-laden thin film flow problem is inspired by the schemes presented in Witelski and Bowen [35] for higher-order parabolic PDEs and Warner et al. [34] for surfactants. In Witelski and Bowen, several ADI schemes, based on backward Euler, second-order backward difference formulas, as well as Newton-like schemes, are derived for solving the nonlinear PDE known as the thin film equation,

$$h_t + \nabla \cdot \big(f(h) \nabla \nabla^2 h\big) = 0. \tag{7}$$

The backward Euler-based ADI scheme for (7) uses approximate values of h in the nonlinear and mixed-derivative implicit terms. It is suggested to start with approximations, such as time-lagged values, for evaluating these terms and calculating the numerical solution at the timestep. Then use this solution for the new approximate values within the same timestep and recalculate. This results in an iterative scheme at each timestep. However, for solving the thin film equation, it was noted that the iterations did not provide a noticeable improvement. Warner et al. use this method for a coupled system of nonlinear PDEs relating to surfactants. They handle the higher-order terms implicitly using Crank-Nicolson, and apply ADI to this. The remaining terms, which are at least second-order in space, are treated explicitly. For the nonlinear and mixed-derivative terms, the values are time-lagged and the problem is solved only once per timestep. In the simulations, $\Delta x = \Delta y = \pi/100 \approx 0.0314$ required a timestep of $O(10^{-5})$.

Our approach is to handle applicable terms implicitly, using ADI, and treat the remaining terms explicitly, as we show below. The terms handled implicitly are those with spatial derivatives on the same variable as the time derivative. For example, Equation (1) has the time derivative on h, so the terms treated implicitly should have spatial derivatives on h. Making this choice allows for the splitting of the two-dimensional operators into to the product of two one-dimensional operators in the derivation of the ADI scheme. Iterations within each timestep allow for a larger Δt to be taken at the cost of some extra calculations. In general, the increase in the size of the timestep outweighs the extra computational work, as shown in Section 7.

For Equation (1), the terms

$$\nabla \cdot \left(\frac{h^3}{\mu(\phi)} \nabla \nabla^2 h + \frac{\rho(\phi)}{\mu(\phi)} h^3 \hat{\mathbf{x}} \right) \tag{8}$$

can be handled implicitly. This is because the spatial derivatives on these terms are applied to h. Of these terms, some parts of them will be handled by approximation, as in Witelski and Bowen [35]. Including the first-order terms in the implicit treatment allows them to be discretized spatially using centered differencing to maintain stability. Solving this equation numerically assumes that ϕ is known, or can be approximated, and we are solving for h. First discretize the terms in (8) in time with backward Euler, including the time derivative,

$$h^{n+1} + \Delta t \nabla \cdot \left(\frac{h^3}{\mu(\phi)} \nabla \nabla^2 h + \frac{\rho(\phi)}{\mu(\phi)} h^3 \hat{\mathbf{x}} \right)^{n+1} = h^n. \tag{9}$$

Write out the operators in (9) fully,

$$\begin{aligned} h^{n+1} + \Delta t \left[\partial_x \left(\frac{h^3}{\mu(\phi)} h_{xxx} \right) + \partial_y \left(\frac{h^3}{\mu(\phi)} h_{yyy} \right) \right. \\ \left. + \partial_x \left(\frac{\rho(\phi)}{\mu(\phi)} h^3 \right) \right]^{n+1} + \Delta t \left[\partial_x \left(\frac{h^3}{\mu(\phi)} h_{yyx} \right) + \partial_y \left(\frac{h^3}{\mu(\phi)} h_{xxy} \right) \right]^{n+1} = h^n. \end{aligned} \tag{10}$$

The idea behind the ADI approach is to reduce the implicit part of (10), with derivatives in both x and y, to a product of two operators, each with only derivatives in either x or y. To achieve this, the terms involving only x-derivatives and only y-derivatives are grouped together. Define the operators

$$D_x = \partial_x \left(\frac{h^3}{\mu(\phi)} \partial_{xxx} + \frac{\rho(\phi)}{\mu(\phi)} h^2 I \right)^{n+1}, \; D_y = \partial_y \left(\frac{h^3}{\mu(\phi)} \partial_{yyy} \right)^{n+1}. \tag{11}$$

Then replacing the terms in (10) with the definitions in (11), we have

$$\begin{aligned} h^{n+1} + \Delta t (D_x + D_y) h^{n+1} \\ + \Delta t \left[\partial_x \left(\frac{h^3}{\mu(\phi)} h_{yyx} \right) + \partial_y \left(\frac{h^3}{\mu(\phi)} h_{xxy} \right) \right]^{n+1} = h^n. \end{aligned} \tag{12}$$

In order to obtain an ADI scheme from (12), note that $I + \Delta t D_x + \Delta t D_y = (I + \Delta t D_x)(I + \Delta t D_y) - (\Delta t)^2 D_x D_y$ and so the left-hand side, with the addition of an $O(\Delta t^2)$ term, can be written as a product of two one-dimensional operators.

$$\begin{aligned}&(I+\Delta t D_x)(I+\Delta t D_y)h^{n+1}-(\Delta t)^2 D_x D_y h^{n+1}\\&+\Delta t\left[\partial_x\left(\frac{h^3}{\mu(\phi)}h_{yyx}\right)+\partial_y\left(\frac{h^3}{\mu(\phi)}h_{xxy}\right)\right]^{n+1}=h^n.\end{aligned}\tag{13}$$

To handle the nonlinear terms, which occur in front of derivatives, and mixed-derivative terms in (13), define them as approximate, denoted by a tilde (e.g., $\tilde{h}^{n+1}$). The approximate terms can be chosen in some reasonable manner, such as time-lagged or extrapolated. This will be discussed in more detail later. Subtract the mixed-derivative terms from and add the $O(\Delta t^2)$ term to both sides. This leaves a scheme in which all the terms operating on h^{n+1} are known, as is the entire right-hand side.

$$\begin{aligned}&(I+\Delta t\tilde{D}_x)(I+\Delta t\tilde{D}_y)h^{n+1}=h^n\\&+\left\{(\Delta t)^2\tilde{D}_x\tilde{D}_y-\Delta t\left[\partial_x\left(\frac{\tilde{h}^3}{\mu(\tilde{\phi})}\partial_{yyx}\right)+\partial_y\left(\frac{\tilde{h}^3}{\mu(\tilde{\phi})}\partial_{xxy}\right)\right]\right\}^{n+1}\tilde{h}^{n+1}.\end{aligned}\tag{14}$$

For simplicity, define the operators in (14) as

$$\tilde{L}_x=I+\Delta t\tilde{D}_x,\ \tilde{L}_y=I+\Delta t\tilde{D}_y.$$

Subtracting $\tilde{L}_x\tilde{L}_y\tilde{h}^{n+1}$ from both sides of (14), which cancels the $O(\Delta t^2)$ term, yields

$$\tilde{L}_x\tilde{L}_y\left(h^{n+1}-\tilde{h}^{n+1}\right)=-\left(\tilde{h}^{n+1}-h^n\right)-\Delta t\nabla\cdot\left(\frac{\tilde{h}^3}{\mu(\tilde{\phi})}\nabla\nabla^2\tilde{h}+\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^3\hat{\mathbf{x}}\right)^{n+1}.\tag{15}$$

At this point, the implicit part of the scheme is complete and the explicit terms can be added back into (15) using forward Euler.

$$\begin{aligned}\tilde{L}_x\tilde{L}_y\left(h^{n+1}-\tilde{h}^{n+1}\right)&=-\left(\tilde{h}^{n+1}-h^n\right)-\Delta t\nabla\cdot\left(\frac{\tilde{h}^3}{\mu(\tilde{\phi})}\nabla\nabla^2\tilde{h}+\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^3\hat{\mathbf{x}}\right)^{n+1}\\&+\Delta t\nabla\cdot\left\{D(\alpha)\left[\frac{h^3}{\mu(\phi)}\nabla\left(\rho(\phi)h\right)-\frac{5}{8}\frac{h^4}{\mu(\phi)}\nabla\left(\rho(\phi)\right)\right]\right\}^n.\end{aligned}\tag{16}$$

Define

$$u=h^{n+1}-\tilde{h}^{n+1},$$

which can be thought of as a correction term to the approximation of h^{n+1}, and (16) can be written as a three-step process: two one-directional solves (17)-(18) and an update step (19).

$$\begin{aligned}\tilde{L}_x v&=-\left(\tilde{h}^{n+1}-h^n\right)-\Delta t\nabla\cdot\left(\frac{\tilde{h}^3}{\mu(\tilde{\phi})}\nabla\nabla^2\tilde{h}+\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^3\hat{\mathbf{x}}\right)^{n+1}\\&+\Delta t\nabla\cdot\left\{D(\alpha)\left[\frac{h^3}{\mu(\phi)}\nabla\left(\rho(\phi)h\right)-\frac{5}{8}\frac{h^4}{\mu(\phi)}\nabla\left(\rho(\phi)\right)\right]\right\}^n,\end{aligned}\tag{17}$$

$$\tilde{L}_y u = v, \tag{18}$$

$$h^{n+1} \approx \tilde{h}^{n+1} + u. \tag{19}$$

Since the operators $\tilde{L}_x$ and $\tilde{L}_y$ involve at most fourth-order terms, the spatial discretization of them will lead to a five-point stencil in the x- and y-direction, respectively. This discretization is discussed fully in Section 5. Along each row/column of the discretized domain, this results in a pentadiagonal linear algebra problem. This can be solved using a pentadiagonal solver, or a more generic banded matrix solver.

To help with the inaccuracy in the nonlinear and mixed-derivative terms resulting from approximation, an iterative procedure can be used at each timestep to improve the solution and size of the timestep. This was first suggested for the ADI scheme in the context of thin film equations [35]. This procedure amounts to repeating the three-step process associated with solving each equation at each timestep and updating the approximate solution with the most recent solution, until the new and approximate solutions sufficiently converge. For example, one would solve (17)-(19), solve (29)-(31), and examine how much the approximate solution differs from this computed solution. If this difference is significant, one can replace the old approximate terms with the computed solution and solve the same timestep again. This process can be continued until the approximate and computed solutions are close. This is similar to fixed-point iteration.

For Equation (1), when entering the timestep, a choice must be made as to the value of $\tilde{h}^{n+1}$ and $(\tilde{\phi}\tilde{h})^{n+1}$. Using h as an example, two reasonable choices would be a time-lagged approximation, h^n, which is a first-order accurate approximation in time, or an extrapolated approximation, $2h^n - h^{n-1}$, which is second-order in time. For adaptive timestepping, this extrapolation is given by $h^n + (\Delta t/\Delta t_{\text{old}})(h^n - h^{n-1})$, where Δt is the prospective timestep between t^n and t^{n+1} and Δt_{old} is the timestep between t^{n-1} and t^n. While the second choice of an approximation is second-order, it also requires storing an extra set of data, namely h^{n-1}. Other choices for estimating h^{n+1} and $(\phi h)^{n+1}$ based on previous data could be used as well. With this choice made, the three-step process for each equation can be implemented, obtaining a solution for h^{n+1} and $(\phi h)^{n+1}$. We refer to the case when the solution obtained here is accepted as performing *One Iteration.* However, at this point, the approximation can be redefined, $\tilde{h}^{n+1} = h^{n+1}$, and the process repeated. This can be continued until convergence between the approximate and new solution, or equivalently when the correction term u is small in a chosen norm. We refer to this case as *Iterations* since the problem is solved iteratively for each timestep.

For (2), the ADI method is applied to ϕh as a whole, since the time derivative is on this term. The applicable terms in the equation are

$$\nabla \cdot \left[-D(\alpha) \left(\rho_f \frac{(\phi h) h^2}{\mu(\phi)} \nabla(\phi h) \right) + \phi h \left(\frac{\rho(\phi)}{\mu(\phi)} h^2 + (1-\phi) V_s f(\phi) w(h) \right) \hat{\mathbf{x}} \right]. \tag{20}$$

As with (1), the time discretization of (20) is based on a backward Euler method

$$\begin{aligned} (\phi h)^{n+1} + \Delta t \nabla \cdot \Bigg[-D(\alpha) \left(\rho_f \frac{(\phi h) h^2}{\mu(\phi)} \nabla(\phi h) \right) \\ + \phi h \left(\frac{\rho(\phi)}{\mu(\phi)} h^2 + (1-\phi) V_s f(\phi) w(h) \right) \hat{\mathbf{x}} \Bigg]^{n+1} = (\phi h)^n. \end{aligned} \tag{21}$$

Writing out the operators in (21) explicitly,

$$\begin{aligned}(\phi h)^{n+1} - \Delta t D(\alpha)\rho_f \left[\partial_x\left(\frac{(\phi h)h^2}{\mu(\phi)}\partial_x(\phi h)\right) + \partial_y\left(\frac{(\phi h)h^2}{\mu(\phi)}\partial_y(\phi h)\right)\right]^{n+1} \\ +\Delta t\partial_x\left[\phi h\left(\frac{\rho(\phi)}{\mu(\phi)}h^2 + (1-\phi)V_s f(\phi)w(h)\right)\right]^{n+1} = (\phi h)^n.\end{aligned} \tag{22}$$

Define the operators in (22) involving only x-derivatives and only y-derivatives as $\mathcal{D}_x$ and $\mathcal{D}_y$, respectively.

$$\begin{aligned}\mathcal{D}_x = -D(\alpha)\rho_f\partial_x\left(\frac{(\phi h)h^2}{\mu(\phi}\partial_x\right)^{n+1} + \partial_x\left(\left[\frac{\rho(\phi)}{\mu(\phi)}h^2(1-\phi)V_s f(\phi)w(h)\right]I\right)^{n+1}, \\ \mathcal{D}_y = -D(\alpha)\rho_f\partial_y\left(\frac{(\phi h)}{\mu(\phi)}h^2\partial_y\right)^{n+1}.\end{aligned} \tag{23}$$

Using (23), the equation can be compactly written as

$$(\phi h)^{n+1} + \Delta t\left(\mathcal{D}_x + \mathcal{D}_y\right)(\phi h)^{n+1} = (\phi h)^n. \tag{24}$$

Note that there are no mixed-derivative terms to handle in (24). The left-hand side can be written as the product of two one-dimensional operators, incurring an $O(\Delta t^2)$ term in the process.

$$\left(I + \Delta t\mathcal{D}_x\right)\left(I + \Delta t\mathcal{D}_y\right)(\phi h)^{n+1} - (\Delta t)^2\mathcal{D}_x\mathcal{D}_y(\phi h)^{n+1} = (\phi h)^n. \tag{25}$$

Add the $O(\Delta t^2)$ term to both sides of (25), and make all terms that occur nonlinearly at time t^{n+1} approximate, as before.

$$\left(I + \Delta t\tilde{\mathcal{D}}_x\right)\left(I + \Delta t\tilde{\mathcal{D}}_y\right)(\phi h)^{n+1} = (\phi h)^n + (\Delta t)^2\tilde{\mathcal{D}}_x\tilde{\mathcal{D}}_y(\tilde{\phi}\tilde{h})^{n+1}. \tag{26}$$

Define

$$\tilde{\mathcal{L}}_x = I + \Delta t\tilde{\mathcal{D}}_x,\ \tilde{\mathcal{L}}_y = I + \Delta t\tilde{\mathcal{D}}_y$$

and subtract $\tilde{\mathcal{L}}_x\tilde{\mathcal{L}}_y(\tilde{\phi}\tilde{h})^{n+1}$ from both sides of (26) to obtain

$$\begin{aligned}\tilde{\mathcal{L}}_x\tilde{\mathcal{L}}_y\left((\phi h)^{n+1} - (\tilde{\phi}\tilde{h})^{n+1}\right) = -\left((\tilde{\phi}\tilde{h})^{n+1} - (\phi h)^n\right) \\ -\Delta t\nabla\cdot\left[-D(\alpha)\left(\rho_f\frac{(\tilde{\phi}\tilde{h})\tilde{h}^2}{\mu(\tilde{\phi})}\nabla(\tilde{\phi}\tilde{h})\right) + \tilde{\phi}\tilde{h}\left(\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^2 + (1-\tilde{\phi})V_s f(\tilde{\phi})w(\tilde{h})\right)\hat{\mathbf{x}}\right]^{n+1}.\end{aligned} \tag{27}$$

The remaining terms can be incorporated into (27) via forward Euler.

$$\begin{aligned}\tilde{\mathcal{L}}_x\tilde{\mathcal{L}}_y\left((\phi h)^{n+1} - (\tilde{\phi}\tilde{h})^{n+1}\right) = -\left((\tilde{\phi}\tilde{h})^{n+1} - (\phi h)^n\right) \\ -\Delta t\nabla\cdot\left[-D(\alpha)\left(\rho_f\frac{(\tilde{\phi}\tilde{h})\tilde{h}^2}{\mu(\tilde{\phi})}\nabla(\tilde{\phi}\tilde{h})\right) + \tilde{\phi}\tilde{h}\left(\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^2 + (1-\tilde{\phi})V_s f(\tilde{\phi})w(\tilde{h})\right)\hat{\mathbf{x}}\right]^{n+1} \\ -\Delta t\nabla\cdot\left[\phi h\left(\frac{h^2}{\mu(\phi)}\nabla\nabla^2 h - D(\alpha)\left(\frac{h^2}{\mu(\phi)}\nabla h - \frac{5}{8}\frac{h^3}{\mu(\phi)}\nabla\left(\rho(\phi)\right)\right)\right) - \mathbf{F}_{\text{diff}}\right]^n.\end{aligned} \tag{28}$$

Define

$$w = (\phi h)^{n+1} - (\tilde{\phi}\tilde{h})^{n+1}.$$

Then (28) can be written out as the three-step process (29)-(31):

$$\begin{aligned}\tilde{\mathcal{L}}_x v = &-\left((\tilde{\phi}\tilde{h})^{n+1} - (\phi h)^n\right)\\ &-\Delta t \nabla \cdot \left[-D(\alpha)\left(\rho_f \frac{(\tilde{\phi}\tilde{h})\tilde{h}^2}{\mu(\tilde{\phi})}\nabla(\tilde{\phi}\tilde{h})\right) + \tilde{\phi}\tilde{h}\left(\frac{\rho(\tilde{\phi})}{\mu(\tilde{\phi})}\tilde{h}^2 + (1-\tilde{\phi})V_s f(\tilde{\phi}) w(\tilde{h})\right)\hat{\mathbf{x}}\right]^{n+1}\\ &-\Delta t \nabla \cdot \left[\phi h \left(\frac{h^2}{\mu(\phi)}\nabla\nabla^2 h - D(\alpha)\left(\frac{h^2}{\mu(\phi)}\nabla h - \frac{5}{8}\frac{h^3}{\mu(\phi)}\nabla\left(\rho(\phi)\right)\right)\right) - \mathbf{F}_{\text{diff}}\right]^n,\end{aligned} \tag{29}$$

$$\tilde{\mathcal{L}}_y w = v, \tag{30}$$

$$(\phi h)^{n+1} \approx (\tilde{\phi}\tilde{h})^{n+1} + w. \tag{31}$$

The spatial operators in the $\tilde{\mathcal{L}}_x$ and $\tilde{\mathcal{L}}_y$ terms are at most second-order, and spatial discretization leads to a three-point stencil in each direction. Similar to (17) and (18), a tridiagonal solver or banded matrix solver can be used to solve along each row/column.

Solving the system, as a whole, at each timestep can be then achieved by solving (1) using (17)-(19) for h^{n+1}, solving (2) using (29)-(31) for $(\phi h)^{n+1}$, then recovering the particle concentration as $\phi^{n+1} = (\phi h)^{n+1}/h^{n+1}$. Note that each solve only uses values h^n, $\tilde{h}^{n+1}$, ϕ^n, and $\tilde{\phi}^{n+1}$, all of which are known. This scheme can be solved in other possible ways. One might choose to use, after solving (1), h^{n+1} in lieu of an approximation for $\tilde{h}^{n+1}$ for solving (2). Alternatively, the equations could be solved in the opposite order.

4. Adaptive Timestepping

We use an adaptive timestepping scheme to advance the solution. The scheme utilizes the solution at consecutive timesteps t^{n-1}, t^n, t^{n+1}. Based on a measure of error, it decides whether or not to accept the new solution, and if it is reasonable to increase the size of the timestep. This is a modification of the scheme used in Bertozzi et al. [4], in which it serves as an estimate of a dimensionless local truncation error in time. Consider the solution of the film thickness, h, at times t^{n-1}, t^n, and t^{n+1}. Calculate $e^{n+1} = (h^{n+1} - h^n)/h^n$ and $e^n = (h^n - h^{n-1})/h^n$. The modification from the original method is to divide by the value h^n at each point rather than $h^n_{\max} = \max_{i,j}\{h^n_{i,j}\}$, since it produces a better-working adaptive scheme for this problem. Denote the timestep going from time t^n to t^{n+1} as Δt and from t^{n-1} to t^n as Δt_{old}. Then define

$$\text{Error} = \left|\left|e^{n+1} - \frac{\Delta t}{\Delta t_{\text{old}}}e^n\right|\right|. \tag{32}$$

This provides a dimensionless estimate of the local truncation error in time, accumulated over the grid. The solution will be accepted if this error is less than some tolerance, denoted Tol_1. If the error is less than a smaller tolerance, $\text{Tol}_2 < \text{Tol}_1$, for a fixed number of steps, the timestep is increased by a scale factor. If the error is larger than Tol_1, the maximum number of iterations within a timestep is surpassed, or the solution becomes negative, the timestep is reduced by a factor of 2. An example for Tol_1 and Tol_2 would be $10^{-7} \times$ (Area of Domain) and $10^{-9} \times$ (Area of Domain) respectively, where the difference in the tolerances are at least an order of magnitude apart to prevent the error from alternating between too large to accept and small enough to increase the timestep. The form of these tolerances were chosen to make it convenient for various size domains without having to change the tolerances manually for each domain.

Since (32) only takes into account one of the two variables, this error can be computed for ϕh, or merely ϕ, as well. These two errors can be combined into an overall measure of the error by taking the maximum of the two, or by some other reasonable combination such as adding the two errors together or choosing a separate set of tolerances for each.

5. Spatial Discretization

We use centered finite differences for all spatial discretizations. Using the notation, $h_{i+1/2,j} \approx (h_{i,j} + h_{i+1,j})/2$, the fourth-order term in (1) is

$$\begin{aligned}
&\nabla \cdot \left(\frac{h^3}{\mu(\phi)} \nabla \nabla^2 h \right)_{i,j} \\
&\approx \left(\frac{h^3_{i+1/2,j}}{\mu(\phi_{i+1/2,j})} h_{xxx,i+1/2,j} - \frac{h^3_{i-1/2,j}}{\mu(\phi_{i-1/2,j})} h_{xxx,i-1/2,j} \right) / \Delta x \\
&+ \left(\frac{h^3_{i+1/2,j}}{\mu(\phi_{i+1/2,j})} h_{yyx,i+1/2,j} - \frac{h^3_{i-1/2,j}}{\mu(\phi_{i-1/2,j})} h_{yyx,i-1/2,j} \right) / \Delta x \\
&+ \left(\frac{h^3_{i,j+1/2}}{\mu(\phi_{i,j+1/2})} h_{xxy,i,j+1/2} - \frac{h^3_{i,j-1/2}}{\mu(\phi_{i,j-1/2})} h_{xxy,i,j-1/2} \right) / \Delta y \\
&+ \left(\frac{h^3_{i,j+1/2}}{\mu(\phi_{i,j+1/2})} h_{yyy,i,j+1/2} - \frac{h^3_{i,j-1/2}}{\mu(\phi_{i,j-1/2})} h_{yyy,i,j-1/2} \right) / \Delta y.
\end{aligned} \tag{33}$$

Here, the third derivatives are calculated at half-grid points by differencing consecutive standard second-order approximations. Two representative examples are

$$h_{xxx,i+1/2,j} \approx \left(h_{i+2,j} - 3h_{i+1,j} + 3h_{i,j} - h_{i-1,j} \right) / \Delta x^3, \tag{34}$$

$$\begin{aligned} h_{xxy,i,j+1/2} \approx &\left((h_{i+1,j+1} - 2h_{i,j+1} + h_{i-1,j+1})/\Delta x^2 \right. \\ &\left. -(h_{i+1,j} - 2h_{i,j} + h_{i-1,j})/\Delta x^2 \right) / \Delta y. \end{aligned} \tag{35}$$

The two second-order terms are discretized as

$$\begin{aligned}
&\nabla \cdot \left(\frac{h^3}{\mu(\phi)} \nabla (\rho(\phi) h) \right)_{i,j} \\
&\approx \left(\frac{h^3_{i+1/2,j}}{\mu(\phi_{i+1/2,j})} (\rho(\phi_{i+1,j}) h_{i+1,j} - \rho(\phi_{i,j}) h_{i,j}) - \frac{h^3_{i-1/2,j}}{\mu(\phi_{i-1/2,j})} (\rho(\phi_{i,j}) h_{i,j} - \rho(\phi_{i-1,j}) h_{i-1,j}) \right) / \Delta x^2 \\
&+ \left(\frac{h^3_{i,j+1/2}}{\mu(\phi_{i,j+1/2})} (\rho(\phi_{i,j+1}) h_{i,j+1} - \rho(\phi_{i,j}) h_{i,j}) - \frac{h^3_{i,j-1/2}}{\mu(\phi_{i,j-1/2})} (\rho(\phi_{i,j}) h_{i,j} - \rho(\phi_{i,j-1}) h_{i,j-1}) \right) / \Delta y^2,
\end{aligned} \tag{36}$$

$$
\begin{aligned}
&\nabla \cdot \left(\frac{h^4}{\mu(\phi)} \nabla(\rho(\phi)) \right)_{i,j} \\
&\approx \left(\frac{h^4_{i+1/2,j}}{\mu(\phi_{i+1/2,j})} (\rho(\phi_{i+1,j}) - \rho(\phi_{i,j})) - \frac{h^4_{i-1/2,j}}{\mu(\phi_{i-1/2,j})} (\rho(\phi_{i,j}) - \rho(\phi_{i-1,j})) \right) / \Delta x^2 \\
&+ \left(\frac{h^4_{i,j+1/2}}{\mu(\phi_{i,j+1/2})} (\rho(\phi_{i,j+1}) - \rho(\phi_{i,j})) - \frac{h^4_{i,j-1/2}}{\mu(\phi_{i,j-1/2})} (\rho(\phi_{i,j}) - \rho(\phi_{i,j-1})) \right) / \Delta y^2.
\end{aligned} \tag{37}
$$

The advective term is discretized using a standard centered-differencing scheme.

The terms in (2) are discretized in the same manner since many of them are similar to those in (1). The fourth- and second-order terms that come from $\mathbf{v}_{\text{av}}$ are discretized as in (33)-(37), with h replaced by ϕh. Both advective terms are discretized via standard centered differencing.

The shear-induced diffusion term is discretized the same way as (36)-(37).

$$
\begin{aligned}
&\nabla \cdot \left(\hat{D}(\phi) \frac{h^2 \rho(\phi)}{\mu(\phi)} \nabla \phi \right)_{i,j} \\
&\approx \left(\hat{D}(\phi_{i+1/2,j}) \frac{h^2_{i+1/2,j} \rho(\phi_{i+1/2,j})}{\mu(\phi_{i+1/2,j})} (\phi_{i+1,j} - \phi_{i,j}) - \hat{D}(\phi_{i-1/2,j}) \frac{h^2_{i-1/2,j} \rho(\phi_{i-1/2,j})}{\mu(\phi_{i-1/2,j})} (\phi_{i,j} - \phi_{i-1,j}) \right) / \Delta x^2 \\
&+ \left(\hat{D}(\phi_{i,j+1/2}) \frac{h^2_{i,j+1/2} \rho(\phi_{i,j+1/2})}{\mu(\phi_{i,j+1/2})} (\phi_{i,j+1} - \phi_{i,j}) - \hat{D}(\phi_{i,j-1/2}) \frac{h^2_{i,j-1/2} \rho(\phi_{i,j-1/2})}{\mu(\phi_{i,j-1/2})} (\phi_{i,j} - \phi_{i,j-1}) \right) / \Delta y^2.
\end{aligned}
$$

Centered differencing is not used for the moving reference frame, if one is employed. Instead, a second-order upwind differencing scheme is used, which will be discussed in the next section.

6. Reference Frame

The area of interest in the simulations is near the front of the flow, where effects like the capillary and particle-rich ridges occur. With a fixed reference frame, the spatial domain would need to be taken as the entire area over which the flow would evolve, leading to large portions of the domain where no change is occurring. This issue can be easily addressed by using a moving reference frame.

To implement a moving reference frame, we add an extra term to each equation, $-sh_x$ on the left-hand side of (1) and $-s(\phi h)_x$ on (2). Here, $s > 0$ is the speed at which the moving reference frame travels. Zhou et al. [36] approximate the front speed by removing all terms from the equations which are higher than first order, leaving only the advective terms. They observe that these terms capture the large scale dynamics, including the speed of the shocks, and the ridges that develop in h and ϕ. This leaves a 2×2 system of conservation laws of the form

$$
h_t + [F(h, \phi h)]_x = 0, \tag{38}
$$

$$
(\phi h)_t + [G(h, \phi h)]_x = 0, \tag{39}
$$

$$
F(h, \phi h) = \frac{\rho(\phi)}{\mu(\phi)} h^3,
$$

$$
G(h, \phi h) = \frac{\rho(\phi)}{\mu(\phi)} (\phi h) h^2 + (\phi h)(1 - \phi) V_s f(\phi) w(h).
$$

The initial conditions for (38)-(39) are

$$h(x,0) = \begin{cases} h_l, & x \leq 0, \\ h_r, & x > 0, \end{cases} \tag{40}$$

$$(\phi h)(x,0) = \begin{cases} \phi_0 h_l, & x \leq 0, \\ \phi_0 h_r, & x > 0. \end{cases} \tag{41}$$

where h_l and h_r in (40) and (41) are the initial film thickness and the height of the precursor b, respectively, and ϕ_0 in (41) is the initial particle concentration of the fluid. These initial conditions specify a Riemann problem [19]. From the initial shock in both equations, an intermediate state emerges, $(h_i, (\phi h)_i)$. The weak form of this system produces two Rankine-Hugoniot jump conditions, which define the shock speeds, ahead and behind the intermediate states. For s_1, the speed of the shock behind the intermediate state, and s_2, the speed ahead, these conditions are given by

$$\begin{aligned} s_1 &= \frac{F(h_i,(\phi h)_i) - F(h_l,(\phi h)_l)}{h_i - h_l} = \frac{G(h_i,(\phi h)_i) - G(h_l,(\phi h)_l)}{(\phi h)_i - (\phi h)_l}, \\ s_2 &= \frac{F(h_r,(\phi h)_r) - F(h_i,(\phi h)_i)}{h_r - h_i} = \frac{G(h_r,(\phi h)_r) - G(h_i,(\phi h)_i)}{(\phi h)_r - (\phi h)_i}. \end{aligned} \tag{42}$$

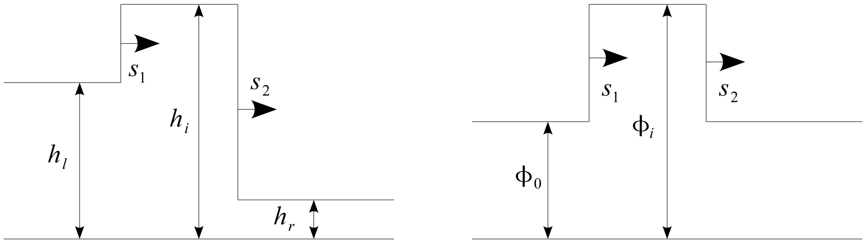

Figure 4: The intermediate states that develop in the film thickness (left) and particle concentration (right) for the first-order system of equations.

The intermediate states and shocks can be seen in Figure 4. This nonlinear system (42) of four equations and four unknowns, $h_i, (\phi h)_i, s_1$, and s_2, can be solved via Newton's method. For the simulations shown in Section 7, our reference frame speed is an average of the two speeds, $s = (s_1 + s_2)/2$.

The discretization of the terms for the moving reference frame is done explicitly using forward Euler combined with second-order upwind-differencing,

$$-sh_x \approx -s\frac{-h_{i+2,j} + 4h_{i+1,j} - 3h_{i,j}}{2\Delta x}.$$

This was chosen over explicit first-order upwind and implicit centered differencing. For a test run to time $t = 10$ with no variation in the y-direction, implicit centered differencing produced the highest particle-rich ridge, but introduced small oscillations ahead of the flow that were approximately 2%

of the height of the ridge. First-order upwind was dissipative and lead to the ridge being 28% smaller than implicit centered differencing. The effects of choosing second-order upwind appear to be some minor dissipation, about 17% as compared to implicit centered differencing, and dispersion, which was not seen in this test case, behind the particle-rich ridge.

The moving reference frame can be used for both the one- and two-dimensional cases (see Figures 5 and 6). To demonstrate this, simulations were run under the same conditions as those in Section 7. The theory-based solution for the problem without higher-order terms (38)-(41) aligns well with the one-dimensional numerical solution for the full problem. The two-dimensional solution for the full problem with a perturbation to the initial film thickness leads to a finger that moves faster than the one-dimensional case and the troughs, to the sides of the finger, move slower.

This can be viewed more succinctly in Figures 7 and 10, where the contours of the perturbed two-dimensional case are shown. The position of the finger runs ahead of the one-dimensional front, which is approximately at $x = 15$, while the troughs lag behind. The averaging of the front position was first investigated by Huppert [14] for experiments involving clear fluids. Both simulations start with the same volume and, after an initial transient, the average front positions for the one- and perturbed two-dimensional case (measured at $h = 0.5$) stay close to each other (Figure 8). Figure 9 shows the position of the finger and the trough in the two-dimensional case over time.

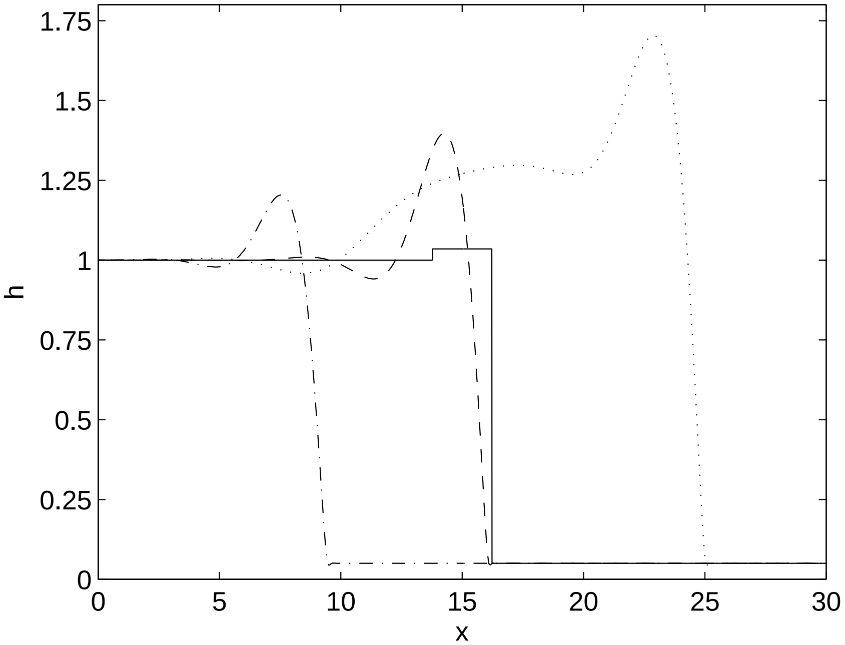

Figure 5: Comparison of theory and simulations at time $t = 100$ for the film thickness, h: theory without higher-order terms (solid line), one-dimensional solution to the full problem (dashed line), perturbed two-dimensional finger (dotted line), and perturbed two-dimensional trough (dot-dashed line). The domain in the y-direction is 15 units long, with the finger slice taken at $y = 7.5$ and the trough slice taken at $y = 1$ (see Figure 7).

7. Simulations

A rectangular domain is used with the x-direction oriented down the inclined plane and the y-direction across the inclined plane. In all cases, the particle concentration is initially taken to be

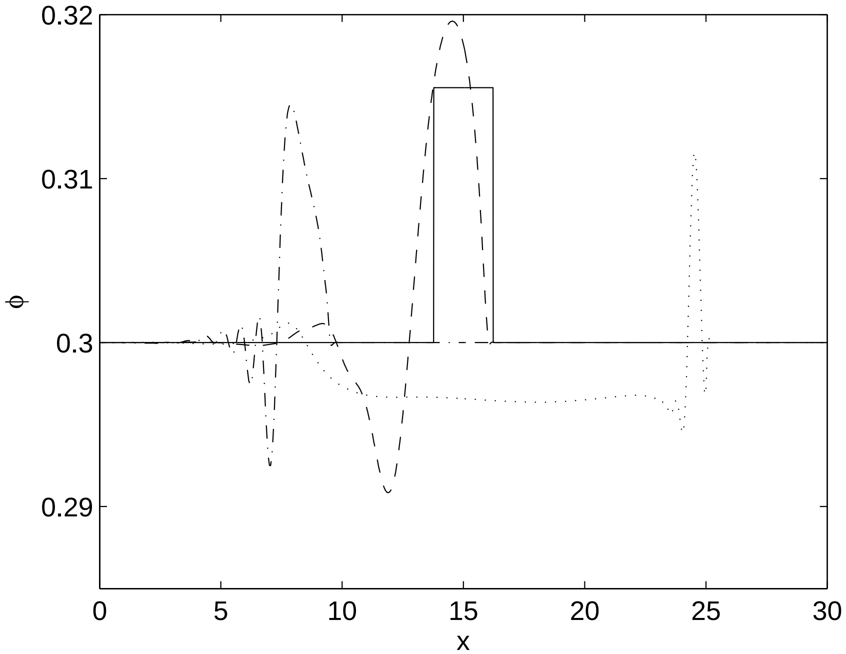

Figure 6: Comparison of theory and simulations at time $t = 100$ for the particle concentration, ϕ. The labels are the same as in Figure 5.

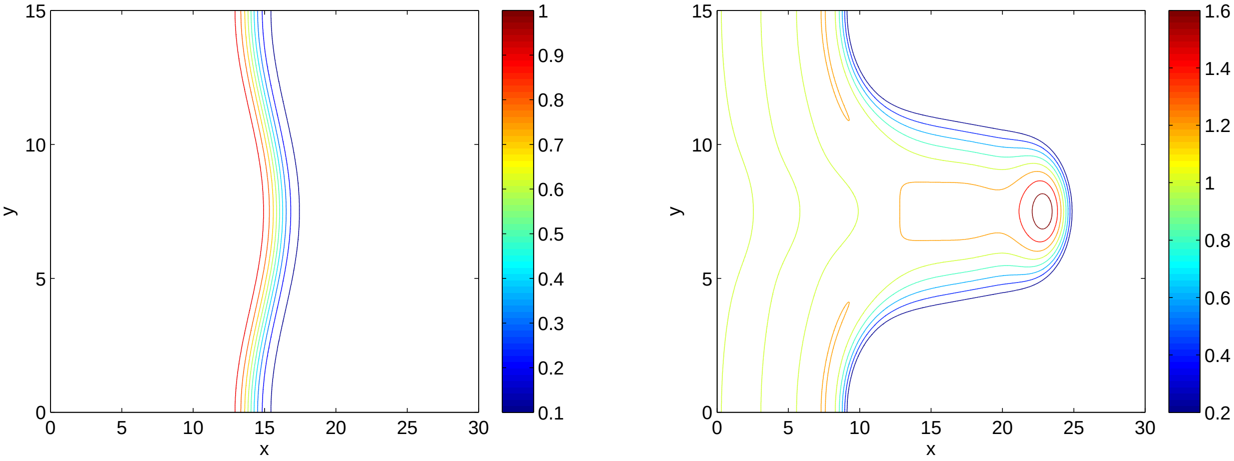

Figure 7: A contour plot of the simulation at times $t = 0$ (left) and $t = 100$ (right) for the film thickness, h, in the perturbed two-dimensional case. The perturbation in two dimensions leads to a fingering instability not seen in the one-dimensional case.

$\phi(x, y, 0) = \phi_0$, where $0 \leq \phi_0 \leq \phi_{\max}$. This corresponds to having a well-mixed initial fluid. The film thickness far behind the contact line is set at $h(x, y, 0) = 1$ and ahead of the flow, a precursor of height $h(x, y, 0) = b$ is assumed. At the contact line, a perturbation to a linear front can be applied to induce behavior such as a fingering instability. The parameters in the model are taken to be: $a = 0.1, \rho_f = 1.7, Ca = 10^{-3}, \alpha = \pi/4$. The constant $\phi_{\max}$ is taken to be 0.67, in line with the simulations in Cook et al. [7]. The initial timestep is set to $\Delta t = 10^{-6}$ and the mesh width is

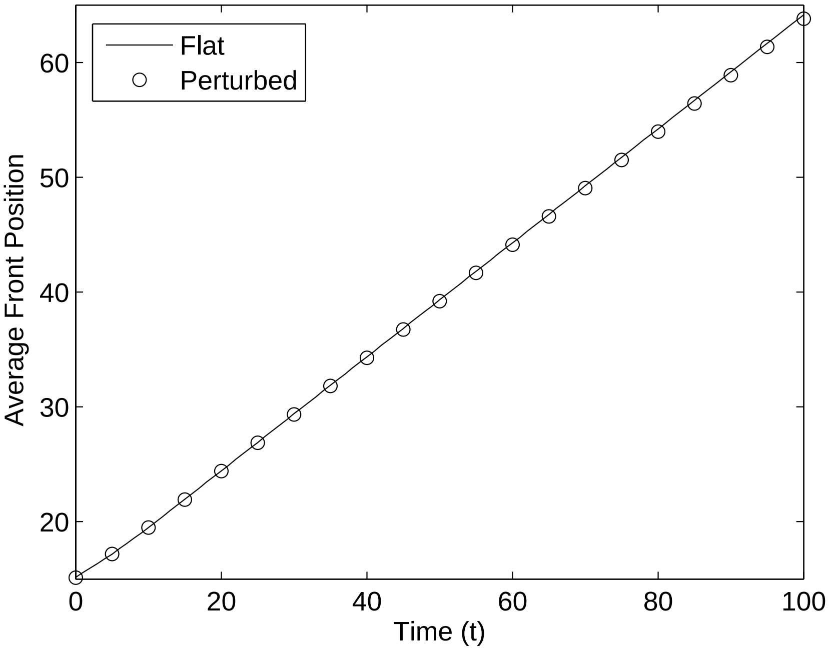

Figure 8: The average front position of the film thickness, h, of the one-dimensional and perturbed two-dimensional case up to time $t = 100$. After an initial transient, the average front positions stay close to each other.

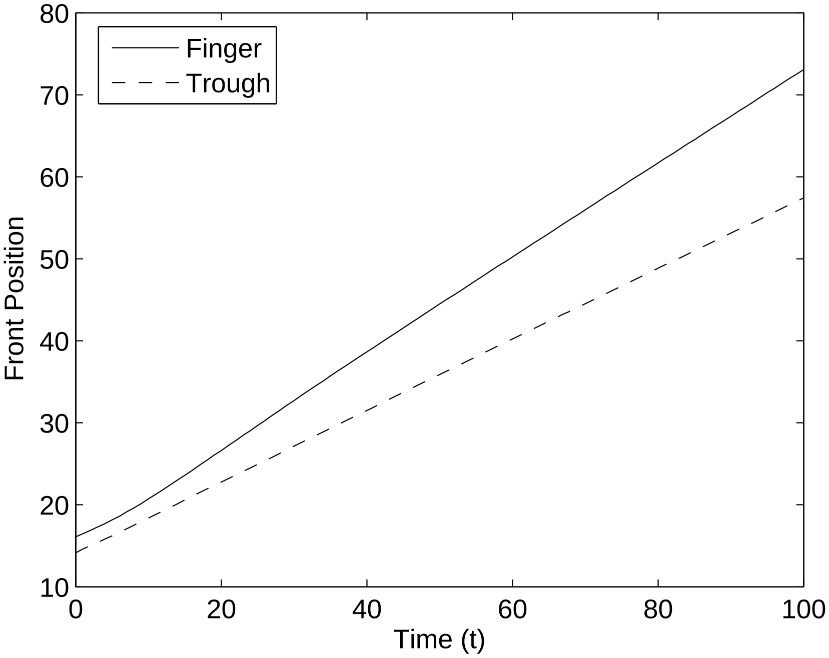

Figure 9: The front position of the film thickness, h, of the perturbed two-dimensional case up to time $t = 100$ along the finger and trough.

$\Delta x = \Delta y = 0.05$.

For the model, two sources contribute to the height of the film thickness and particle concentra-

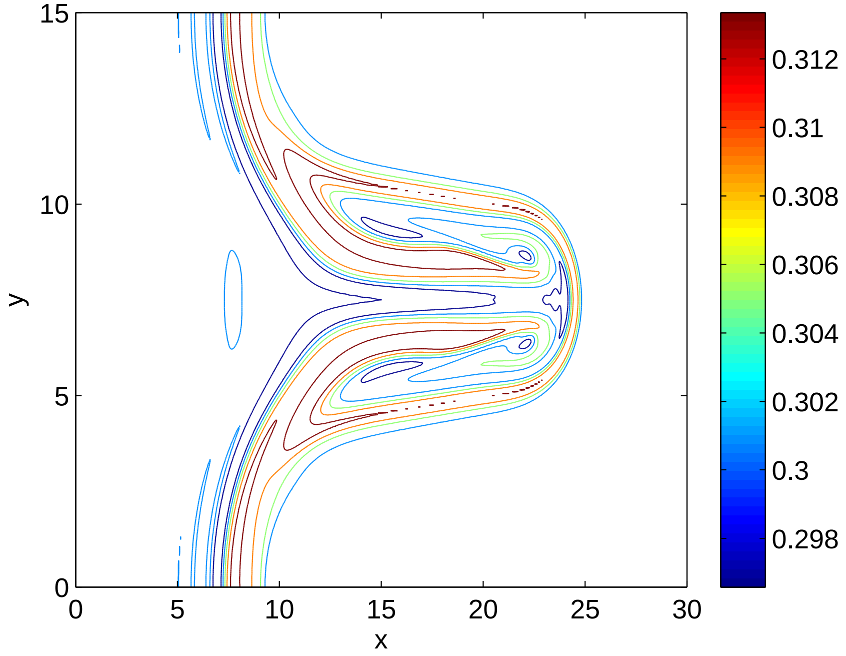

Figure 10: A contour plot of the simulation data at time $t = 100$ for the particle concentration, ϕ, in the perturbed two-dimensional case. The perturbation leads to a particle-rich ridge that outlines and begins to fill in the finger.

tion near the front of the flow. The first is the higher-order terms, such as surface tension, which produce smooth ridges in both h and ϕ. Second, even without these terms, an intermediate state at the front emerges for both variables, higher than either of their respective left or right states. These heights are dependent on the precursor b.

The height of the precursor in the following simulations is chosen to be the same as Δx. In general, the choice of precursor has a small effect on the speed of the flow, but a large effect on both the film thickness and particle concentration. To illustrate this, Table 1 shows the height of the intermediate states for both h and ϕ as well as the speeds of the trailing and leading shocks obtained from the theory-based solution to the system of conservation laws (38)-(41) (see Section 6 for a more in-depth discussion). The intermediate film thickness h_i and particle concentration ϕ_i increase as the height of the precursor b decreases. For the shock speeds, a smaller precursor leads to the trailing shock speed s_1 staying relatively the same, but the leading shock speed s_2 slows down and approaches s_1. These results agree with the previous ones related to solving the system of conservation laws [6, 36]. For this model, the smallest precursor for which a solution exists is $b \approx 9 \times 10^{-4}$ [6]. A precursor close to this case, $b = 0.001$, produces shocks speeds which are close together and an intermediate particle concentration near the maximum packing fraction. An alternative settling function that permits solutions with smaller precursors, $f_B(\phi) = (1 - \phi/\phi_{\max})^5$, is examined in Cook et al. [6].

The boundary conditions for h are Dirichlet in front and behind, in the x-direction, the flow and Neumann on the sides, in the y-direction. The same is employed for ϕ. In addition, all third derivatives in h, normal to the boundary, are set to 0. More specifically, for a rectangular domain with length X_0 and width Y_0, the boundary conditions are

b	h_i	ϕ_i	s_1	s_2
0.1	1.01653	0.307566	0.459323	0.510221
0.05	1.03478	0.315538	0.459314	0.483782
0.025	1.07107	0.330331	0.459301	0.471418
0.0125	1.1427	0.356006	0.459289	0.465441
0.00625	1.28276	0.396078	0.459294	0.462488
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0.001	9.14247	0.635545	0.459788	0.459916

Table 1: The intermediate states and shock speed solutions from Equation (42) based on the precursor thickness b. As the precursor decreases, both h_i and ϕ_i increase and the shock speeds converge.

$$\begin{gathered}h(0,y)=1,\ h_{xxx}(0,y)=0,\ h(X_0,y)=b,\ h_{xxx}(X_0,y)=0,\\ h_y(x,0)=0,\ h_{yyy}(x,0)=0,\ h_y(x,Y_0)=0,\ h_{yyy}(x,Y_0)=0,\\ \phi(0,y)=\phi_0,\ \phi(X_0,y)=\phi_0,\ \phi_y(x,0)=0,\ \phi_y(x,Y_0)=0.\end{gathered}$$

The simulations are all run using moving reference frames, with the speed of the frame determined as in Section 6.

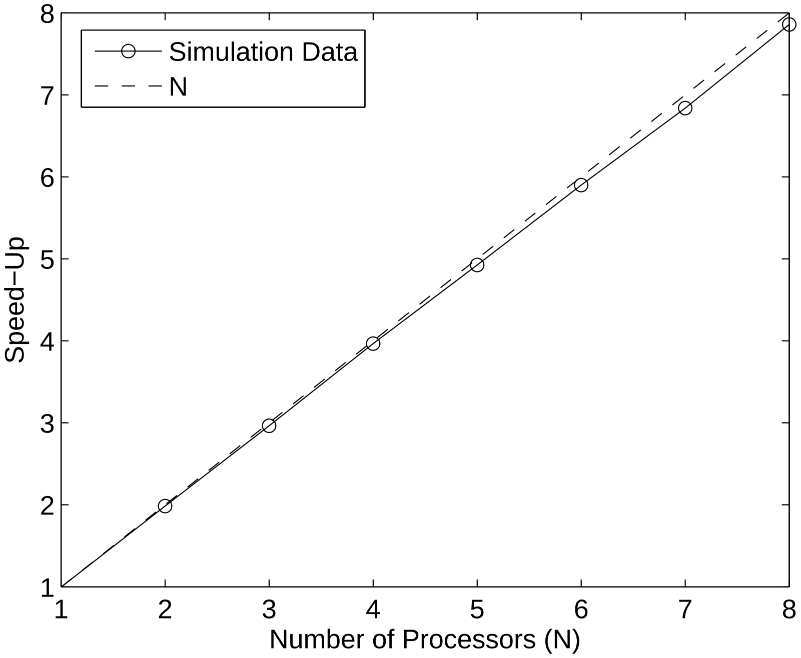

Figure 11: The speed-up gained by going from 1 to N processors using OpenMP. The line $y = N$ is shown as a point of reference.

The code is written in parallel using the C++ OpenMP package. This choice of parallelization was made since the majority of calculations are done via *for* loops and OpenMP works well with loop-heavy code. This includes the calculation of all finite differences and the solves along rows and columns associated with the ADI part of the scheme. This is especially useful since rows/columns can be solved independently of each other for each equation. In addition, writing special solvers for linear systems of equations across multiple processors [25, 28] is avoided by this approach. The speed-up attained using N processors is calculated by dividing the runtime for one processor by the runtime for N processors (Speed-Up = Time(1 Processor)/Time(N Processors)).

Based on Figure 11, the scaling is close to linear up to 4 processors, with a small drop-off in performance as the number increases. This almost-linear behavior is a result of all of the code, outside of a few minor calculations and the recording of the data, being amenable to parallelization.

To test some preferences that need be chosen a priori in the simulation, we conducted short-time tests to gauge the effectiveness of each approach. The ones considered here are (a) whether to time-lag or extrapolate the approximate terms and (b) whether or not to perform iterations past a single solve to improve the approximate terms, and therefore the solution at each timestep (see Table 2).

(a) Approximate Terms	Time-Lagged	Extrapolation
(b) Number of Iterations	One Iteration	Iterations

Table 2: The two choices to be made when implementing the numerical scheme. One must choose whether to (a) time-lag or extrapolate the approximate terms and (b) whether or not to perform additional iterations past the initial solve.

Consider an initial condition of $\phi_0 = 0.3$ and a front perturbed from Riemann initial data, $h(x, y, 0) = 1$ far behind the front, $h(x, y, 0) = 0.05$ far ahead of the front. At the jump from fluid to precursor, the shape of the front is given as $x_{\text{front}} = X_0/2 - \cos(2\pi y/Y_0)$. This initial data is then smoothed using hyperbolic tangent and matched to the boundary condition (see Figure 14). This has the effect that the initial timestep can be taken more leniently.

We ran this initial simulation for each of the four combinations in Table 2 to time $t = 1$ and the maximum timestep allowed, average number of iterations per timestep, and the total runtime, in seconds, are listed in the table below (Table 3). This and Table 4 provide some global measures to compare the different schemes rather than illustrating convergence studies for any particular method. The choice of $t = 1$ was made as the timestep changes dramatically over this time interval and can provide insight as to what methods seem practical for long-time runs. Since adaptive timestepping is utilized here, the tolerances are tuned so as to ensure that the simulation stays stable, not only to time $t = 1$ but for some time afterwards as well (it is taken up to $t = 100$ in this case, which is the length of the long-run simulations).

	$\Delta t_{\max}$	Avg. Iter.	Runtime
Time-Lagged and One Iteration	0.000568341	1.0	518.2
Time-Lagged and Iterations	0.00183296	2.20997	601.468
Extrapolation and One Iteration	4.07743×10^{-5}	1.0	19596.1
Extrapolation and Iterations	0.00486338	1.29668	376.603

Table 3: Results for time $t = 1$ based on various choices for implementation.

Using *Iterations* performs well for both choices of approximate terms in that the total runtimes are low, the maximum timesteps are large, and the number of iterations stays close to 1. Between these two, *Extrapolation and Iterations* does best, with nearly one fewer iteration required per timestep, on average, and a runtime that is 37% shorter. Performing *One Iteration*, the runtime for *Time-Lagged* is in between the two cases with *Iterations*, but for *Extrapolation*, it performs poorly, producing a runtime that is 33 to 52 times worse than the other three options. This is due to the small maximum timestep that is associated with this approach, which is 14 to 119 times smaller than the other three. At this point, it makes sense to discard the *Extrapolation and One Iteration* approach due to its excessive runtime and explore the remaining ones.

Under the same conditions, we ran a longer simulation, this time to $t = 100$. Using the best remaining options, we can glean some idea as to which one(s) will work best for a longer simulation.

	$\Delta t_{\max}$	Avg. Iter.	Runtime
Time-Lagged and One Iteration	0.00107169	1.0	17811.3
Time-Lagged and Iterations	0.00329173	2.95498	13153.8
Extrapolation and Iterations	0.0106161	2.01204	3364.93

Table 4: Results for time $t = 100$.

Comparing Tables 3 and 4, the maximum timestep for each approach has increased. Using *Iterations*, the average number has gone up in for both *Time-Lagged* and *Extrapolation*. However, the average number of iterations per timestep for *Extrapolation* is approximately one iteration fewer than for *Time-Lagged*. Also the runtime takes about 2.9 times longer for *Time-Lagged* compared to *Extrapolation*. One can see the benefit of performing iterations instead of using a smaller timestep in comparing the results for *Time-Lagged and One Iteration* and *Time-Lagged and Iterations*. *Time-Lagged and One Iteration* advances the solution approximately the same time forward with three timesteps as *Time-Lagged and Iterations* does with one timestep and three iterations. However, doing two extra timesteps costs more than two extra iterations, as seen in their respective runtimes. This is because the explicit terms do not need to be re-calculated for each iteration while they do for each timestep. Therefore, the only two options which make sense to use are the ones involving *Iterations*. Of these, *Extrapolation* is the clear favorite.

In Figure 12, we see that by time $t = 8$, all three approaches have settled into a respective timestep. The timestep for *Extrapolation and Iterations* does best, followed by *Time-Lagged and Iterations* and *Time-Lagged and One Iteration*. The timestep for *Extrapolation and Iterations* is 3.2 times better than *Time-Lagged and Iterations* and 9.9 times better than *Time-Lagged and One Iteration*. The benefit of the larger timestep for both approaches with *Iterations* is partially offset by the need for extra calculations related to the iterations.

Figure 13 shows the number of iterations required throughout the simulation. For *Extrapolation and Iterations*, the increase in iterations approximately between times $t = 20$ and $t = 30$ corresponds to the finger forming and stretching out ahead of the flow in the film thickness and the particle-rich ridge growing higher and outlining the finger. While the number of iterations jumps once to 3 and then back down to 2 for *Extrapolation and Iterations*, it remains constant at 3 for *Time-Lagged and Iterations*. The cost of storing extra data and performing a small computation to find the extrapolated approximations seems a small price to pay to save one iteration per timestep, which includes recalculating values involving the approximate terms and performing the ADI solves.

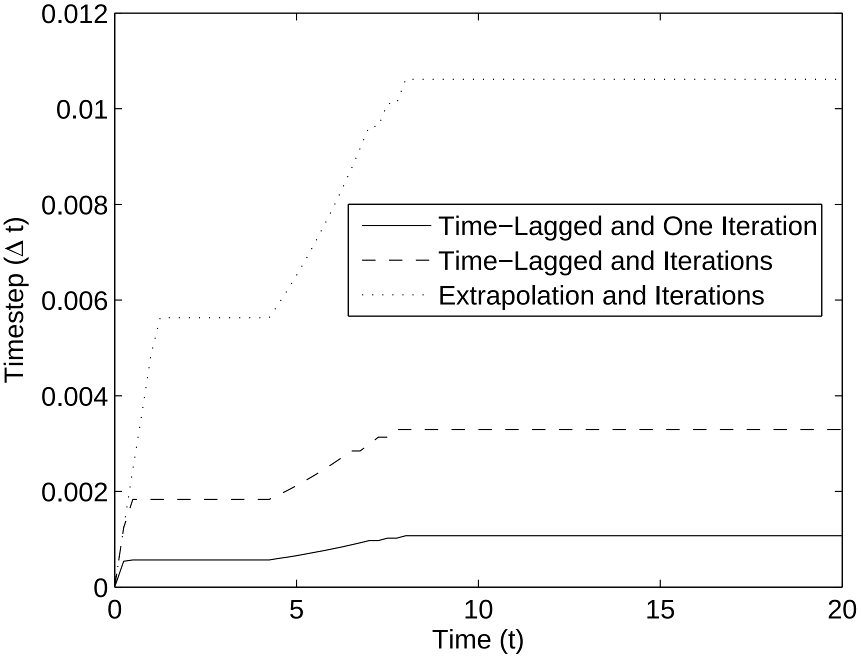

Figure 12: The adaptive timestep up to time $t = 20$. The timestep, Δt, is recorded in intervals of 0.25 for the three cases. *Extrapolation and Iterations* has a significantly larger timestep than either *Time-Lagged and One Iteration* or *Time-Lagged and Iterations*.

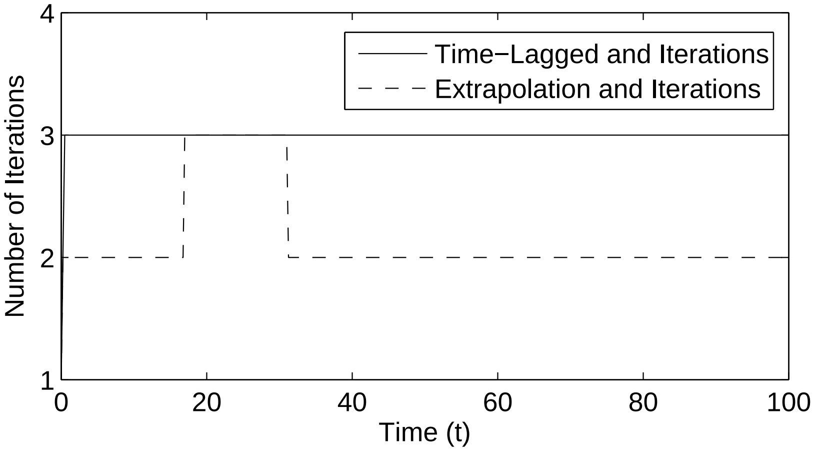

Figure 13: The number of iterations up to time $t = 100$. The iterations are recorded in intervals of 0.25 for the two cases. Using *Extrapolation and Iterations* does better than *Time-Lagged and Iterations* in terms of fewest number of iterations.

Using the simulation data up to $t = 100$, we can examine the effects of the initial perturbation graphically. For the film thickness, a small capillary ridge forms in the center of the perturbation (Figure 15) and begins to stretch out ahead of the bulk flow (Figures 16 and 17). This is the well-known fingering instability present in thin film flows. For the particle concentration, a particle-rich ridge initially forms at the contact line (Figure 15) and, as the fingering instability evolves, outlines the shape of the finger (Figures 16 and 17). Directly behind the ridge, a pocket of lower

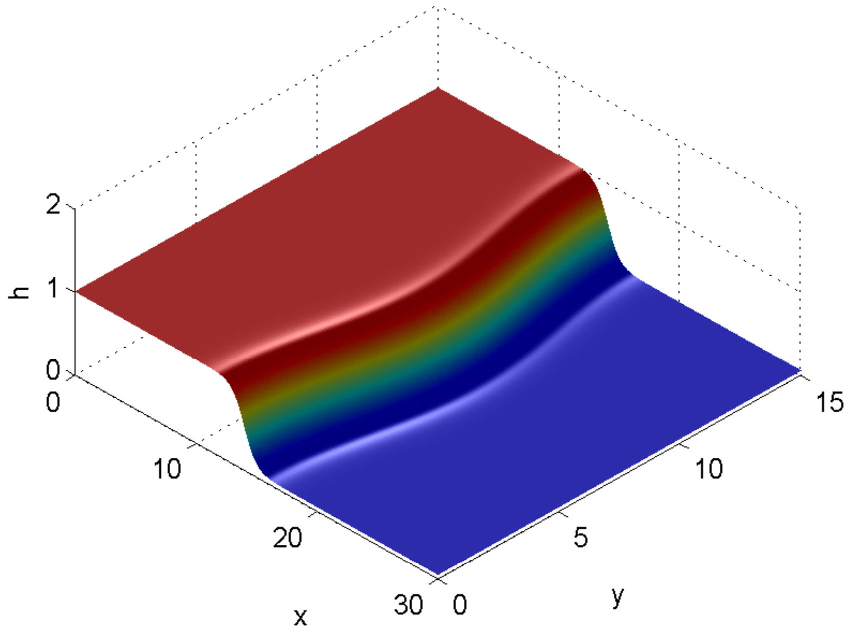

Figure 14: The initial film thickness. It is perturbed by a cosine wave along y and smoothed along x by hyperbolic tangent.

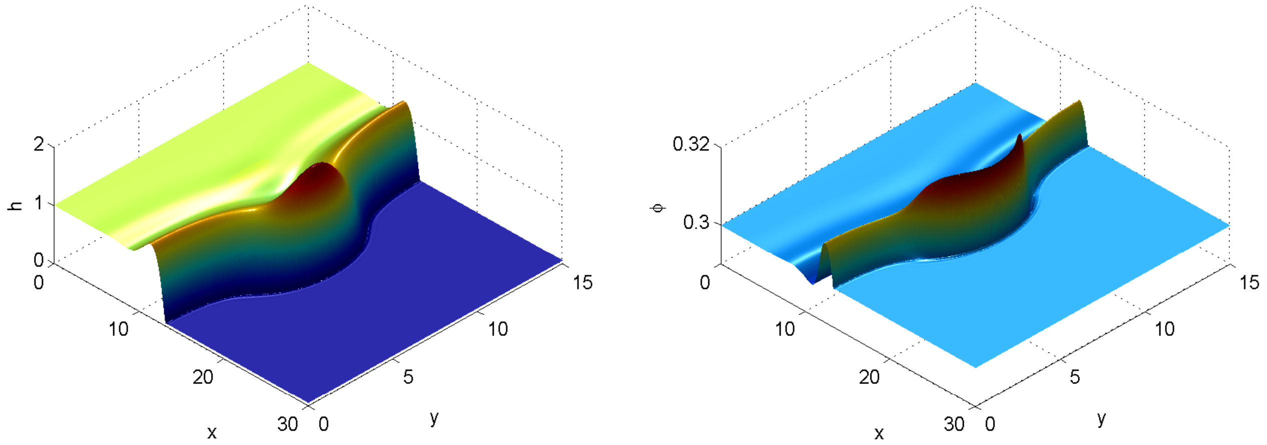

Figure 15: Film thickness (left) and particle concentration (right) at time $t = 25$. A small ridge forms in both, with the highest point in the perturbation.

concentration forms. The interior of the finger is slowly encroached upon by the particles that have accumulated near the back and sides of the finger. This can be seen in Figure 17 as an interior layer along the inside of the particle-rich ridge. It is possible that this phenomenon is not physical, meaning that it occurs only in the simulations and not in the experiments, and may be a result of the current model not containing all of the necessary physics.

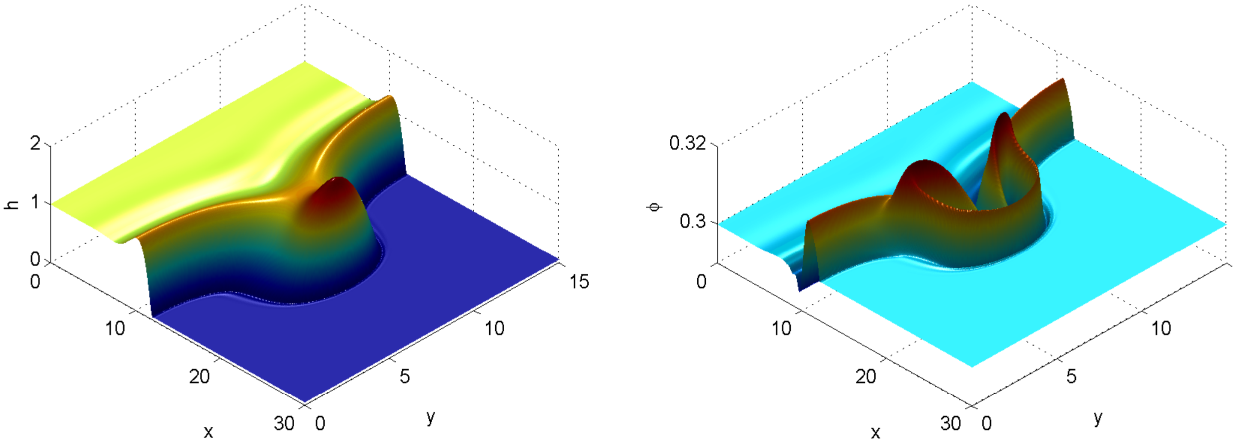

Figure 16: Film thickness (left) and particle concentration (right) at time $t = 50$. A fingering instability and particle-rich ridge form.

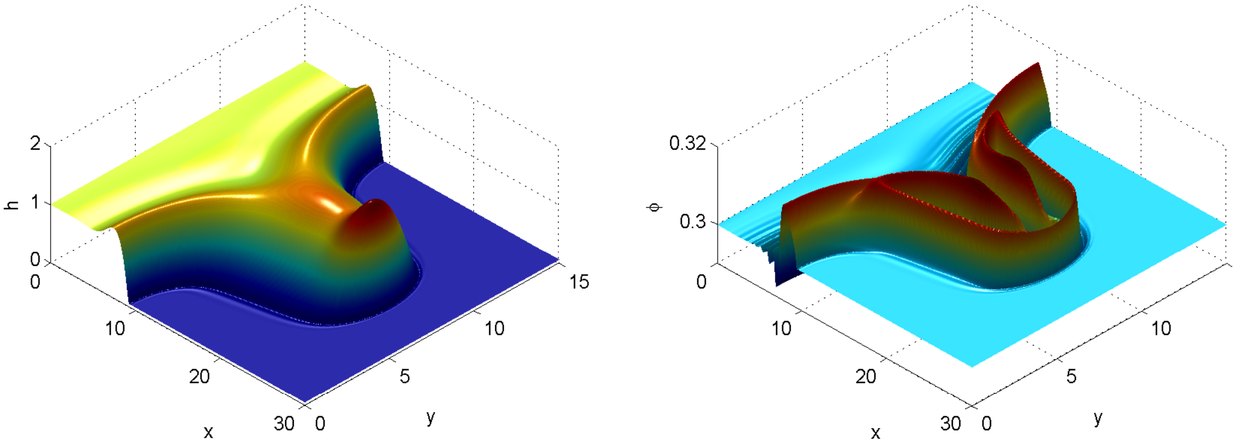

Figure 17: Film thickness (left) and particle concentration (right) at time $t = 100$. The fluid finger stretches out ahead of the bulk flow. The particle-rich ridge increases in concentration and has a higher concentration in and around the fingering instability.

8. Comparison to Experiments

Experiments for particle-laden thin film flows have been compared in one dimension to the solution, both analytically and numerically, for constant-volume clear fluid flows. The average front position for clear fluids is given by a power law, where the location of the front scales like $Ct^{1/3}$, where C is a scaling constant [14]. Ward et al. [33] compare the average front position of the flow to this scaling and find agreement for particle concentrations below $\phi = 0.45$ and deviations at later times for higher concentrations. Grunewald et al. [12] compare the average front position to a re-derived one-dimensional model, based on results from Huppert [14] with a precursor, and to experiments and numerical solutions of the one-dimensional problem. The $Ct^{1/3}$ scaling appears valid for concentrations of 0.25 to 0.45, and the scaling constant for experiments and numerics are within 20% of the theoretical constant. We seek to compare the numerical solution in two dimensions to images of experiments, taking into account that variations occur across the front of the flow.

We use 1000 cSt polydimethylsiloxane (PDMS), a silicone oil, for the liquid component of the fluid. For the particles, glass beads with diameters in the range of $250 - 425$ μm are used. The two components are then well-mixed and released down an inclined plane from a reservoir. This corresponds to a constant-volume experiment, whereas our simulations are constant-flux. The approximation of a constant-volume problem by a constant-flux one may be invalid at early times, as the height of the fluid will be changing rapidly. However, the height of the flow changes slower at later times, at which point a constant-flux approximation may be valid.

The experiment, which we will compare to simulation, is a fluid of approximately 90 cm^3 containing a volume which is 35.9% particles. The plane is inclined at a 32-degree angle. The fluid is allowed to flow down the plane, which is 14 cm across and 90 cm down. In the experiments, the flow starts out close to uniform across the front, away from the edges, and over time develops instabilities, in the form of fingers stretching out ahead of the bulk flow. Since, for simulations, starting with a uniform front along the y-direction leads to a uniform solution, we start the simulation some time after the start-time to add a perturbation to the initial data, which induces the type of behavior seen in the latter stages of the experiments.

In order to avoid simulating the problem over the entire domain, we truncate the solution near the front and treat the problem locally as being constant-flux. We are interested in the dynamics of finger formation during which time the film thickness only changes by at most 20% , so a local approximation by constant-flux is reasonable.

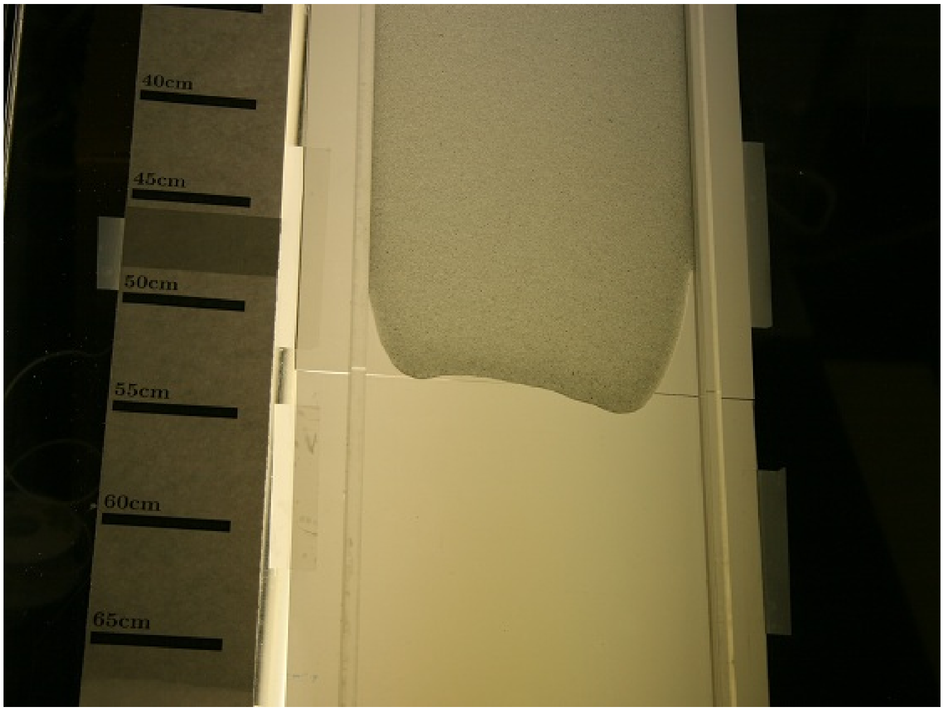

Figure 18: The initial condition of the experiment, used for comparing with the simulation. At this point, the front of the flow has begun to develop perturbations, which will lead to fingering instabilities.

We use two images, taken three minutes apart, to compare with the simulation (Figures 18 and 20). The first image is taken when the front of the flow has reached approximately 53 cm down the plane. The shape of the front is parabolic-like with two large perturbations at either end of the front. In between, smaller perturbations exist which lead to fingering instabilities. The two outer perturbations lead to longer and thicker fingers than the smaller inner perturbations. We take a front similar to this in our simulation.

The scales for a constant-flux problem can be taken from Cook et al. [6], which are the same as for the clear fluid case. The height scale is taken to be $h_0 = 1$ mm. The length scale is $x_0 = (l^2 h_0)^{1/3}$, where the capillary length, l, is $l = \sqrt{\gamma/\rho_l g_\parallel}$. The constants are γ, the coefficient of surface tension; ρ_l, the liquid density; and $g_\parallel$, the component of gravity parallel to the inclined plane. The time scale is $t_0 = (3\mu_l/\gamma)x_0 l^2/h_0^2$, where μ_l is the dynamic liquid viscosity. The capillary number is given by $Ca = \mu_l x_0/\gamma t_0 = h_0^2/3l^2$.

The scales, given these parameters, are $h_0 = 0.001$ m, $x_0 = 0.00161396$ m, $l = 0.00205041$ m, $t_0 = 0.93235$ s, and $Ca = 0.0792863$. Using this, we can construct an initial condition which resembles the experiment and will produce similar results. This is done by measuring the features of the initial image and creating a similar condition. While the flow in the experiment is asymmetric, we take a symmetric initial condition in the simulation which has features that are approximately, in both location and size, the same as in the experiment. The track is taken to be 86.75 units wide (rounded up to the nearest 0.05 increment, which is the value of $\Delta x, \Delta y$), which corresponds to the 14 cm wide track. The precursor in the simulation is set to $b = 0.05$, as in the previous simulations.

A moving reference frame is used since this is taken to be a constant-flux problem locally. The speed of the moving reference frame is approximately $s = 0.343198$, calculated as in Section 6. Running a simulation over the course of three minutes leads to a distance traveled for the frame of approximately 10.69 cm, where the actual displacement, based on experiments, is around 12 cm, so using the constant-flux assumption seems to produce a decent approximation of the distance the fluid will flow.

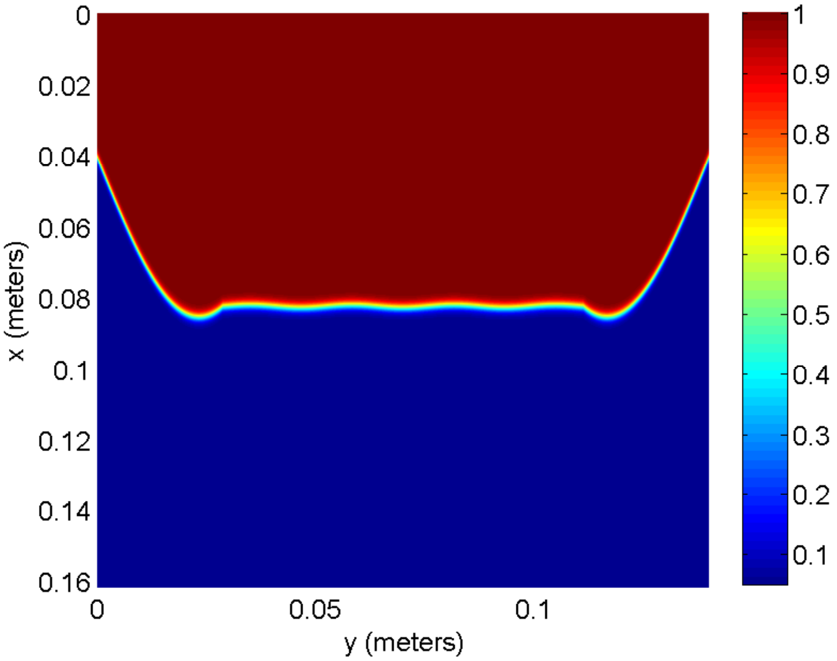

Figure 19: The initial condition for the film thickness, h, used in the simulation. This is an artificially-created starting condition to be representative of the state shown for the experiment. The height is in mm.

The initial data is generated using a sine wave to form the two large perturbations and the space away from the edges. The three fingers that develop between these two perturbations are simulated with a cosine wave of small amplitude, 0.25 in dimensionless units (Figure 19). The simulation is run to $t = 193.06$, the equivalent of three minutes of real-time.

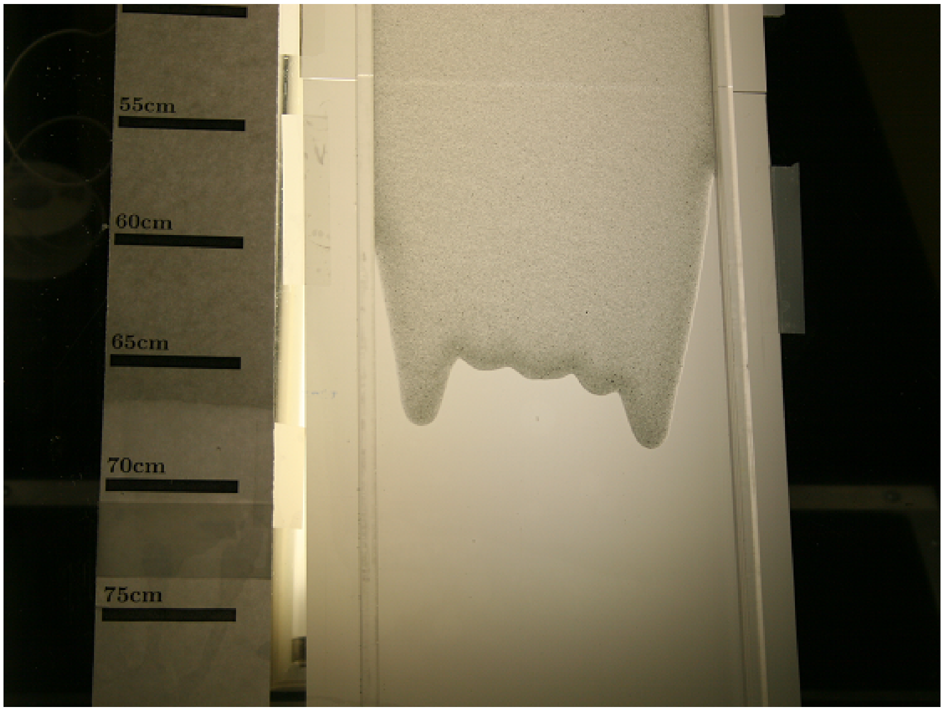

Figure 20: The evolution of the experiment after three minutes. The fingering instability starts to form at the front.

Over the course of the three minutes, the exterior of the outer fingers in the experiments go from 4 cm and 6.5 cm on the left and right, respectively, to 7.5 cm and 12 cm. The interior of these fingers go from less than 1 cm on each side to about 3 cm. The interior fingers are not discernable in the initial image. The flow as a whole, measured from where the fluid touches the walls, has moved about 11 cm down the plane. The interior fingers in the experiment, extend about 0.5 cm ahead of the flow.

In the simulation (Figure 21), the moving reference frame accounts for 10.69 cm of movement, so the position where the fluid touches the edges has moved approximately 11.4 cm. The evolution of the fingers in the simulation is slightly less pronounced than in the experiments. This is likely due to the simulation initially undergoing a transient state where the fluid travels slower than at later times, while the transient in the experiment has occurred prior to this three-minute interval. The exterior of the outer fingers is approximately 4.2 cm and interior 1.2 cm. The interior fingers extend ahead of the flow about 0.8 cm. The tip of the longest finger in the experiments has moved 15 cm while in the simulations, it has advanced approximately 11.4 cm. The tips of the fingers, in the z-direction, reach up to 1.37 mm.

The particle concentration cannot be determined accurately at the particle-rich ridge in the experiment, but the increased opacity at the leading edge of the flow indicates an increase in the concentration, relative to the ambient concentration. This change in shade is approximately 1 cm long in the direction of the flow. In the simulation (Figure 22), the thickness of the ridge ranges from 0.6 to 1.1 cm, which is consistent with the experiments.

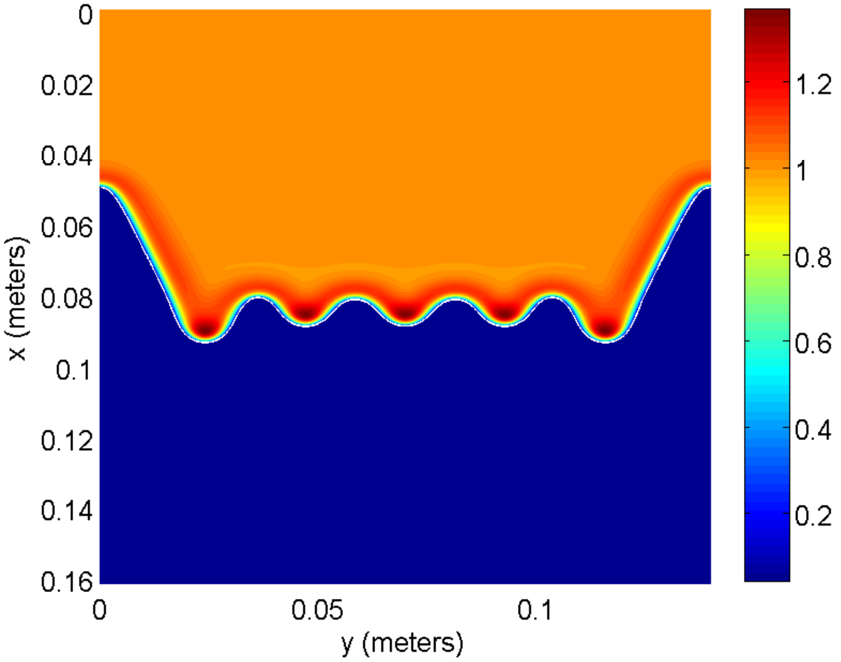

Figure 21: The evolution of the film thickness, h, in the simulation after three minutes. Both the experiment and simulation exhibit a fingering instability, but the instability in the simulation is less pronounced. The height is in mm.

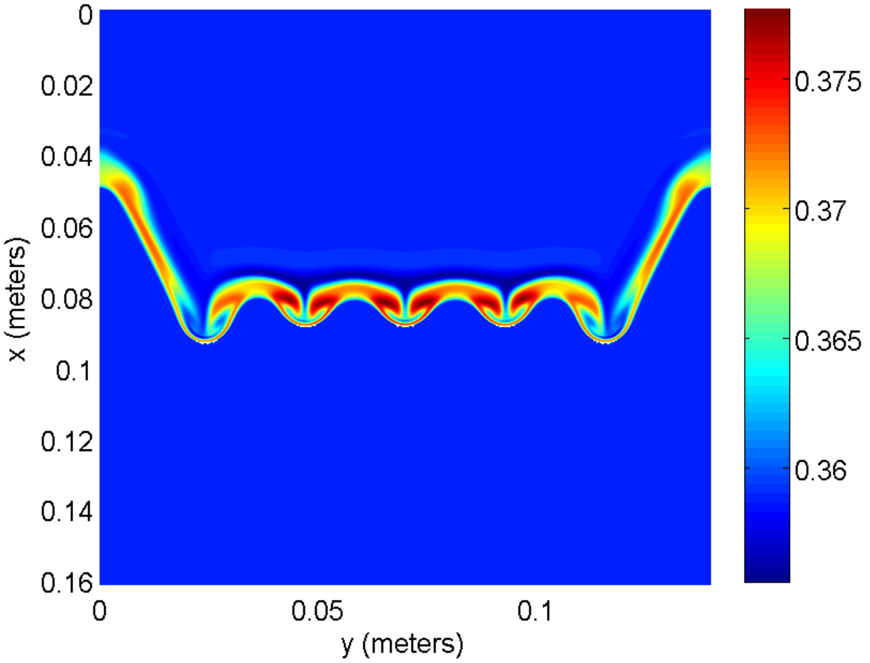

Figure 22: Particle concentration, ϕ, for the film thickness in Figure 21.

9. Discussion

Schemes originally derived for numerically solving high-order parabolic problems have recently been extended to high-order systems, such as the case of surfactants and particle-laden thin films.

Handling the higher-order terms in a practical way is necessary for fast and efficient computation. The scheme we have presented here for particle-laden thin film flow provides an easy-to-program and effective way to solve this high-order coupled system. This scheme can provide a blueprint for approaches to solving similar problems.

The numerical scheme developed for particle-laden thin film flow has several nice attributes. The timestep required for this scheme is in the range of $O(\Delta x^2)$, which is much better than the $O(\Delta x^4)$ for a fully explicit scheme. The structure of the scheme allows for the possibility of solving each equation with its own unique timestep for better efficiency, as the particle concentration is typically the equation that fails the timestep restriction criteria. The linear algebra problem that results from an implicit time discretization along with the nonlinearity is reduced to a series of tri- and pentadiagonal solves, which can be done in parallel along the rows/columns of the grid.

The parallelization of the code is straightforward using OpenMP. The loops for computing the explicit and approximate terms as well as the solves along rows and columns can be done in parallel, leading to a code that scales close to linearly for up to 8 processors, getting close to 8 times speedup. Adding OpenMP implementation to C++ code on any multicore machine is easy to implement, as it only requires adding a few lines of code to existing *for* loops and needs no managing of the movement of data on the programmer's part. Since the code is predominantly such loops, it is easy to parallelize and is highly effective in getting better runtimes.

Implementing *Iterations* within each timestep, which is first presented in Witelski and Bowen [35], but not used in Warner et al. [34], seems to work best for this problem, in terms of allowing for a larger timestep and producing an accurate solution. Among the choices for the approximate terms when performing *Iterations*, *Extrapolation* seems to produce the best runtime and fewest iterations. Implementation requires only storing an extra set of data used in extrapolating the approximate terms but, using the adaptive timestepping discussed here, this data is stored anyway.

The choice of *Extrapolation and Iterations* may work best for this problem, but for other problems or initial conditions, another choice may fare better. It is recommended, as in this case, that a short-term simulation be performed for the different choices of approximate terms and whether or not to perform extra iterations. The small cost of these short runs may allow for a more efficient run for actual simulations. It is also recommended that one examines the results to make sure that the scheme is not only fast with the choice, but sufficiently accurate.

The numerical solution agrees reasonably well with the behavior seen in experiments. This is in part because the model was derived for the case when a particle-rich ridge forms. This is seen in the experiments for high angles of inclination and high concentrations, but will occur in the model for all concentrations and angles. The particle-rich ridge in the simulations is two thin layers of particles, one which originates at the front of the flow and the other from the troughs of the emerging fingers, which may not be physical.

The current model assumes a constant, or average, particle concentration throughout the fluid layer in the z-direction. The same is true for the velocity, which is averaged in the z-direction. Theory exists for the vertical movement of the particles [5], whether they will settle to the inclined plane or form a ridge, and incorporating this behavior into a new model is the current research of the authors. It is hoped that the current numerical scheme will be adaptable to this new model.

Acknowledgements

The authors would like to thank Chris Anderson, Nebojsa Murisic, Joseph Teran, and Tom Witelski for their helpful comments and discussions. The images from experiments are courtesy of

Joyce Ho, Vincent Hu, Paul Latterman, Trystan Koch, and Kanhui Lin, and were performed during the Research Experience for Undergraduates (REU) program at the University of California, Los Angeles during the summer of 2009. This research was supported by NSF grants DMS-0601395 and DMS-1048840 and a grant from the UC Lab Fees Research fund 09-LR-04-116471-BERA.

[1] J.W. Barrett, J.F. Blowey, J. Garcke, Finite element approximation of a fourth order nonlinear degenerate problem, Numer. Math. 80 (1998) 525-556.

[2] R.M. Beam, R.F. Warming, Alternating direction implicit methods for parabolic equations with a mixed derivative, SIAM J. Sci. Stat. Comp. 1 (1980) 131-159.

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

[4] A.L. Bertozzi, M.P. Brenner, T.F. Dupont, L.P. Kadanoff, Singularities and similarities in interface flows, Trends and Perspectives in Applied Mathematics, L. Sirovich, ed., Springer-Verlag Applied Mathematical Sciences, New York, 1994, pp. 155-208.

[5] B. Cook, A theory for particle settling and shear-induced migration in thin film flow, Phys. Rev. E Stat. Nonlin. Soft Matter Phys. 78 (2008) 045303 1-4.

[6] B.P. Cook, A.L. Bertozzi, A.E. Hosoi, Shock solutions for particle-laden thin films, SIAM J. Appl. Math. 68 (2008) 760-783.

[7] B. Cook, O. Alexandrov, A. Bertozzi, Linear stability of particle-laden thin films, Eur. Phys. J. Spec. Top. 166 (2009) 77-81.

[8] I.J.D. Craig, A.D. Sneyd, An alternating-direction implicit scheme for parabolic equations with mixed derivatives, Comput. Math. Applic. 16 (1988) 341-350.

[9] J.A. Diez, L. Kondic, Computing three-dimensional thin film flows including contact lines, J. Comput. Phys. 183 (2002) 274-306.

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

[11] G. Grun, M. Rumpf, Simulations of singularities and instabilities arising in thin film flow, Euro. J. Appl. Math. 12 (2001) 293-320.

[12] N. Grunewald, R. Levy, M. Mata, T. Ward, A.L. Bertozzi, Self-similarity in particle-laden flows at constant volume, J. Eng. Math. 66 (2010) 53-63.

[13] H. Happel, H. Brenner, Low Reynolds number hydrodynamics with special applications to particulate media, Prentice-Hall, Englewood Cliffs, NJ, 1965.

[14] H. Huppert, Flow and instability of a viscous current down a slope, Nature 300 (1982) 427-429.

[15] S. Karaa, A high-order ADI method for parabolic problems with variable coefficients, Int. J. Comput. Math. 86 (2009) 109-120.

[16] L. Kondic, Instabilities in gravity driven flow of thin fluid films, SIAM Rev. Soc. Ind. Appl. Math. 45 (2003) 95-115.

[17] J.K. Kowalski, A finite-difference method for parabolic differential equations with mixed derivatives, Math. Comput. 25 (1971) 675-698.

[18] I.M. Krieger, Rheology of monodisperse lattices, Adv. Colloid Interface Sci. 3 (1972) 111-136.

[19] P.D. Lax, Hyperbolic systems of conservation laws and mathematical theory of shock waves, CBMS-NSF Regional Conference Series in Applied Mathematics 11 (1973) 1-48.

[20] D. Leighton, Ph.D. thesis, Stanford Univ., Stanford, California, 1985.

[21] D. Leighton, A. Acrivos, The shear-induced migration of particles in concentrated suspensions, J. Fluid Mech. 181 (1987) 415-439.

[22] X-Z. Liu, X. Cui, J-G. Sun, FDM for multi-dimensional nonlinear coupled system of parabolic and hyperbolic equations, J. Comput. Appl. Math. 186 (2006) 432-449.

[23] A.V. Lyushnin, A.A. Golovin, L.M. Pismen, Fingering instability of thin evaporating liquid films, Phys. Rev. E Stat. Nonlin. Soft Matter Phys. 65 (2002) 021602 1-7.

[24] S. McKee, Alternating direction methods for a system of parabolic equations in two space dimensions with a mixed derivative, J. Inst. Math. Appl. 8 (1971) 376-385.

[25] M. Naumov, A.H. Sameh, A tearing-based hybrid parallel banded linear system solver, J. Comput. Appl. Math. 226 (2009) 306-318.

[26] C.V. Pao, Numerical analysis of coupled systems of nonlinear parabolic equations, SIAM J. Numer. Anal. 36 (1999) 393-416.

[27] D.W. Peaceman, H.H. Rachford, Jr., The numerical solution of parabolic and elliptic differential equations, J. Soc. Ind. Appl. Math. 3 (1955) 28-41.

[28] E. Polizzi, A.H. Sameh, A parallel hybrid banded system solver: The SPIKE algorithm, Parallel Comput. 32 (2006) 177-194.

[29] J.F. Richardson, W.N. Zaki, Sedimentation and fluidization: Part I, Trans. Inst. Chem. Eng. 32 (1954) 35-53.

[30] Satteluri, R.K. Iyengar, M.K. Jain, Comparative study of two and three level ADI methods for parabolic equations with a mixed derivative, Int. J. Numer. Methods Eng. 10 (1976) 1309-1315.

[31] J.J. Stickel, R.L. Powell, Fluid mechanics and rheology of dense suspensions, Annu. Rev. Fluid Mech. 37 (2005) 129-149.

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

[33] T. Ward, C. Wey, R. Glidden, A.E. Hosoi, A.L. Bertozzi, Experimental study of gravitation effects in the flow of a particle-laden thin film on an inclined plane, Phys. Fluids 21 (2009) 083305 1-7.

[34] M.R.E. Warner, R.V. Craster, O.K. Matar, Fingering phenomena associated with insoluble surfactant spreading on thin liquid films, J. Fluid Mech. 510 (2005) 169-200.

[35] T. Witelski, M. Bowen, ADI schemes for higher-order nonlinear diffusion equations, Appl. Numer. Math. 45 (2003) 331-351.

[36] J. Zhou, B. Dupuy, A.L. Bertozzi, A.E. Hosoi, Theory for shock dynamics in particle-laden thin films, Phys. Rev. Lett. 94 (2005) 117803 1-4.

[37] L. Zhornitskaya, A.L. Bertozzi, Positivity-preserving numerical schemes for lubrication-type equations, SIAM J. Numer. Anal. 37 (2000) 523-555.