Numerical investigations on self-similar solutions of the nonlinear diffusion equation

Yibao Li, Junseok Kim

Department of Mathematics, Korea University, Seoul, 136-713, Republic of Korea

Abstract

In this paper, we present the numerical investigations of self-similar solutions for the nonlinear diffusion equation \( \frac{h_t}{h} = -(h^3 h_{xxx})_x \), which arises in the context of surface-tension-driven flow of a thin viscous liquid film. Here, \( h = h(x, t) \) is the liquid film height. A self-similar solution is \( h(x, t) = h(a(t)(x - x_0) + x_0, t_0) = f((a(t)(x - x_0)) \) and \( a(t) = [1 - 4A(t - t_0)^{1/4}] \), where \( A \) and \( x_0 \) are constants and \( t_0 \) is a reference time. To discretize the governing equation, we use the Crank–Nicolson finite difference method, which is second-order accurate in time and space. The resulting discrete system of equations is solved by a nonlinear multigrid method. We also present efficient and accurate numerical algorithms for calculating the constants, \( A, x_0 \), and \( t_0 \). To find a self-similar solution for the equation, we numerically solve the partial differential equation with a simple step-function-like initial condition until the solution reaches the reference time \( t_0 \). Then, we take \( h(x, t_0) \) as the self-similar solution \( f(x) \). Various numerical experiments are performed to show that \( f(x) \) is indeed a self-similar solution.

© 2013 Elsevier Masson SAS. All rights reserved.

1. Introduction

The objective of this paper is to numerically investigate self-similar solutions for the nonlinear diffusion equation

\[
\frac{h_t}{h} = -(h^3 h_{xxx})_x, \tag{1}
\]

which arises in the context of thin liquid film flow. Here \( h = h(x, t) \) denotes the liquid film height, \( x \), a spatial coordinate, and \( t \), time (see Fig. 1). \( h_{xxx} \) is the constant upstream height and \( b \) is the precursor film thickness. Eq. (1) can be considered as a zero gravity and no surface tension gradient limit of the following equation:

\[
h_t + (h^2 - h^3)_x = -(h^3 h_{xxx})_x \tag{2}
\]

which governs a thin layer of liquid on an inclined substrate driven by thermally created surface tension gradients and influenced by gravity. This equation has been extensively studied experimentally, analytically, and numerically \([1–24]\). A liquid film driven by a thermal gradient with a counteracting gravitational force has been studied experimentally \([15–17]\).

Jump initial data, from a moderately thick film to a thin precursor layer, is shown to give rise to a double wave structure that includes an undercompressive wave \([15]\). The wave structure of solutions observed in numerical simulations with Eq. (2) is related to the hyperbolic theory of the underlying scalar conservation law, \( h_t + (h^2 - h^3)_x = 0 \) \([18]\). See \([19,20]\) and the references therein for related mathematical problems concerning the dynamics of thin films.

Alternating direction implicit schemes are constructed for the solution of the fourth-order thin film equation for surface-tension-driven fluid flows \([21]\). Adaptive mesh refinement for thin film equations is developed in \([22]\). A detailed implementation of an adaptive finite element method was presented in \([23]\). In \([24]\), the authors numerically investigated the effect of the convection term treatment using the Godunov scheme, the WENO scheme, and an upwind-type scheme of a driven thin film equation.

Bernoff and Witelski \([25]\) studied the compactly-supported self-similar solutions of \( h_t = -(h^3 h_{xxx})_x \) for \( 0 < n < 3 \). Further, using linear stability analysis, they showed that the source-type solutions are stable. For further details about the self-similar
solutions of fourth-order nonlinear diffusion equations, we refer the reader to [25–34] and the references therein.

To find a self-similar solution of Eq. (1), we numerically solve the partial differential equation with a simple step-function like initial condition until the solution reaches a reference time. Then, we take the self-similar solution as the one at the reference time. Note that, this numerical self-similar solution is an approximation to the analytic one.

The remainder of this paper is organized as follows. In Section 2, we briefly review the governing equation and introduce a self-similar solution for the nonlinear diffusion equation. In Section 3, we present the Crank–Nicholson finite difference discretization of the governing equation and its nonlinear full-approximation storage (FAS) multigrid solver. Efficient and accurate numerical algorithms for calculating the constants, $A$, $x_0$, and $t_0$ are also described. In Section 4, we present various numerical results. Finally, we state our conclusions in Section 5.

2. Governing equation

We consider the dynamics of a thin layer of liquid of thickness $h = h(x, t)$ on a substrate, driven by surface tension. The configuration is shown schematically in Fig. 1. The spatial variables $x$ and $z$ denote the direction of flow and the film height, respectively. We model the dynamics of the thin film using lubrication approximation with a “depth averaged” velocity:

$$
\dot{u}(x, t) = \frac{1}{h(x, t)} \int_0^{h(x, t)} u(x, z, t) dz = \frac{\gamma h(x, t) \dot{h}(x, t) - \partial_x h^3(x, t)}{3h},
$$

where $\gamma$ denotes the surface tension coefficient and $\eta$ the viscosity of the fluid [35]. Coupling Eq. (3) with the conservation of mass [15], we obtain

$$
h_t(x, t) + [h(x, t) \dot{u}(x, t)]_x = 0.
$$

To non-dimensionalize Eq. (4), we employ the non-dimensional variables denoted by hats, $\hat{h} = h/H$, $\hat{x} = x/L$, and $\hat{t} = t/T$; thus we obtain

$$
\frac{H}{T} \frac{\hat{h}}{\hat{t}} + \left( \frac{H^2 \gamma \hat{h}^3 \hat{h}_{xxx}}{3L^3 \eta} \right) = 0,
$$

where $H$, $L$, and $T$ are the characteristic height, length, and time scales, respectively. Now choose the time scale $T = 3L^3 \eta/(H^2 \gamma)$ so that we have $\hat{h}_t + \left( \hat{h}^3 \hat{h}_{xxx} \right) = 0$. Drop the ‘$$\cdot$$’ to obtain the dimensionless thin film equation:

$$
h_t = -h^3 h_{xxx}
$$

with the boundary conditions, lim$_{x \to -\infty} h(x, t) = h_\infty$ and lim$_{x \to \infty} h(x, t) = 0$.

In this paper, we present a self-similar solution of Eq. (6) of the form

$$
h(x, t) = f(\phi) \quad \text{and} \quad \phi = \alpha(t)(x - x_0), \quad x \in \mathbb{R}, \quad t \geq t_0,
$$

where $x_0$ and $t_0$ are the reference points that satisfy $h_t(x_0, t_0) = 0$ and $\alpha(t_0) = 1$, respectively. Substitution of the similarity ansatz (7) into Eq. (6) yields the ordinary differential equations:

$$
\frac{\alpha'(t)}{\alpha^2(t)} = -\frac{\left(f^3(\phi)f''(\phi)\right)'}{\phi f'(\phi)} = A,
$$

where the prime symbol denotes differentiation with respect to the argument variable of each function and $A$ is a constant. From Eq. (8), $\alpha(t)$ is given as

$$
\alpha(t) = [1 - 4A(t - t_0)]^{-1/4}.
$$

Here we have used the initial condition $\alpha(t_0) = 1$. The similarity solution $f$ should satisfy the equation

$$
f' \left( f''(\phi) \right) = -A f' \left( f''(\phi) \right),
$$

subject to the boundary conditions: lim$_{\phi \to -\infty} f(\phi) = h_\infty$ and lim$_{\phi \to \infty} f(\phi) = b$.

In this study, we use dual approaches to calculate self-similar solutions. One is to calculate $f$ directly from Eq. (10) by using the bvp5c program [36–39], which is an ordinary differential equation solver. A sample MATLAB code is given in Appendix A. The other approach is to solve the evolution equation (6) with an initial condition such as a step-function, and we take $f$ as an intermediate solution at a certain time $t = t_0$.

3. Numerical method

We split the fourth-order equation (6) into a system of second order equations

$$
h_t(x, t) = [M(h(x, t))\mu(x, t)]_x, \quad (11)
$$

$$
\mu(x, t) = -h(x, t), \quad x \in \Omega = (0, L), \quad t \geq 0,
$$

where $M(h) = h^3$. Boundary conditions are given by

$$
h(0, t) = h_\infty, \quad h(L, t) = b,
$$

$$
\mu(0, t) = \mu(L, t) = 0,
$$

where $h_\infty$ is the constant upstream height and $b$ is the precursor film thickness. The first two boundary conditions are Dirichlet boundary conditions and the last two boundary conditions are homogeneous Neumann boundary conditions, which are no-flux boundary conditions.

3.1. Discretization and numerical solver

Now, we present fully discrete schemes for Eqs. (11) and (12) in one dimensional space $\Omega = (0, L)$. Let $N$ be a positive even integer, $\Delta x = L/N$, the uniform mesh size, and $x_i = (i - 0.5)\Delta x$, $1 \leq i \leq N$, the cell-center node point. Let $h^n_i$ and $\mu^n_i$ be approximations of $h(x_i, n\Delta t)$ and $\mu(x_i, n\Delta t)$, respectively. Then, a Crank–Nicholson finite difference discretization of Eqs. (11) and (12) is given by

$$
h^{n+1}_i - h^n_i = \Delta t \left( \frac{M(h^{n+1/2}_i) (\mu^{n+1}_i - \mu^{n+1}_{i-1}) - M(h^{n+1/2}_{i-1}) (\mu^{n+1}_i - \mu^{n+1}_{i-2})}{2(\Delta x)^2} \right),
$$

$$
\frac{n+1}{2} \left( \frac{M(h^{n+1/2}_i) (\mu^{n+1}_i - \mu^{n+1}_{i-1}) - M(h^{n+1/2}_{i-1}) (\mu^{n+1}_i - \mu^{n+1}_{i-2})}{2(\Delta x)^2} \right) + \frac{\mu^n_i}{2} - \frac{2h^n_{i+1} + h^n_{i+1}}{(\Delta x)^2},
$$

where $h^{n+1/2}_i = (h^n_i + h^{n+1}_{i+1})/2$. The boundary conditions are defined as

$$
h_0 = 2h_\infty - h_1, \quad h_{N+1} = 2b - h_N,
$$

$$
\mu_0 = \mu_1, \quad \mu_{N+1} = \mu_N.
$$

In this paper, we use a multigrid method [40] to solve the nonlinear discrete system (15) and (16) at the implicit time level. A detailed
3.2. Calculation of a self-similar solution

In this subsection, we describe our proposed numerical algorithm for calculating a self-similar solution by solving the evolution equation (6) with an initial condition such as a step-function.

A suitable self-similar solution $f(\phi) = h(x, t_0)$ should satisfy

(i) $\max_{\phi \in \Omega} h(x, t) = \max_{\phi \in \Omega} f(\phi)$ and $\min_{\phi \in \Omega} h(x, t) = \min_{\phi \in \Omega} f(\phi)$,

(ii) $A(t) = (\alpha^A(t) - 1)/[4(t - t_0)\alpha^A(t)]$ is constant for $t > t_0$,

where $t_0$ is the reference time when the self-similar solution begins.

Let $h_{\text{max}}^n$ and $h_{\text{min}}^n$ denote the maximum and minimum values of $h(x, t)$ at $t = n\Delta t$. First, we evolve the equation with a given initial condition until the relative errors $|h^{n+1} - h^n|/h^n$ and $|h_{\text{max}}^{n+1} - h_{\text{min}}^{n+1}|/h_{\text{max}}^n$ are smaller than a given tolerance, tol = 1e-6. Initially, we set $t_0 = n\Delta t$ and carry out the following two steps until we get the reference time $t_0$.

Step 1. Compute $A(t) = (\alpha^A(t) - 1)/[4(t - t_0)\alpha^A(t)]$ at three different times $t = t_0 + \beta \Delta t$, $t_0 + 2\beta \Delta t$, and $t_0 + 3\beta \Delta t$. Here $\beta$ is some integer and we use $\beta = 10$ in this study.

Let $x_0(t)$ be the position at time $t$ such that $h(x_0(t), t) = \max_{x \in \Omega} h(x, t)$. Since the maximum value of $h(x, t)$ should be the same in the self-similar solution, i.e., $h(x_0(t), t) = f(\phi_0) = h(x_0(t), t_0)$, from Eq. (7) we have

$$\alpha(t) = (x_0(t) - x_0)/x_0(t) - x_0) \quad \text{and} \quad \phi_0 = (x_0(t) - x_0) / x_0(t) - x_0.$$  \hspace{1cm} (19)

Let $h_0^n = \max_{x \in \Omega} h(x_0(t))$ and we define the quadratic polynomial approximation passing through the three points, $(x_{k-1}, h_{k-1}^n)$, $(x_k, h_k^n)$, and $(x_{k+1}, h_{k+1}^n)$. Then define $x_0(t_0)$ as the critical point of the polynomial and $x_0(t)$ is defined similarly.

Next, we calculate $x_0$ which satisfies $h(x_0, t_0) = 0$ and $h(x_0, t) \neq 0$ for all $t_0$ and $t_0 < t_0 < t_0 + \Delta t$; then, $h(x_0, t) = h(x_0, t_0)$ (see Fig. 2). We can find the unique index $k$ such that $(h_{k+1}^n - h_k^n)/(h_{k+1}^n - h_k^n) \leq 0$ and $b + 0.1(h_{\infty} - b) < h_k^n < b + 0.9(h_{\infty} - b)$. We define the point $x_0$ as the $x$-coordinate of the intersection point of two line segments. One line connects the points $(x_k, h_k^n)$ and $(x_{k+1}, h_{k+1}^n)$, whereas the other passes through the points $(x_k, h_k^n)$ and $(x_{k-1}, h_{k-1}^n)$. Then, $x_0$ is defined as

$$x_0 = x_k + (x_{k+1} - x_k)(h_{k+1}^n - h_k^n)/(h_{k+1}^n - h_{k-1}^n - h_k^n + h_{k+1}^n).$$

Using Eq. (19), we can calculate $A(t) = (\alpha^A(t) - 1)/[4(t - t_0)\alpha^A(t)]$.

Step 2. If $|A(t + 2\Delta t) - A(t + \Delta t)|/A(t + \Delta t + A(t + 2\Delta t)) < $ tol, we set $t_0 = t_0 + \Delta t$. Otherwise, we go back to Step 1 with $t_0 = t_0 + \beta \Delta t$.

After this algorithm, we get $t_0$, $x_0$, $A$, and $f(\phi)$.

We note that unless the initial condition is the similarity solution and the domain is infinite, the numerical solution will always have small deviations from the similarity solution. The similarity solution will be approached as time increases (but not converged in finite time) on the infinite domain. If the width of the domain is finite, then the boundary conditions will eventually override this approach to the similarity solution and the numerical solution converges to the steady-state solution (also only in the limit of infinite time).

4. Numerical experiments

In this section, we describe various numerical experiments, such as a convergence test, as well as other experiments to demonstrate the finite computational domain effect, long time evolution, a numerical self-similar approach to the similarity solution and the effect of parameters and initial profiles on self-similarity. Unless otherwise specified, we use the computational domain $\Omega = (0, 100)$ with an $N = 1024$ mesh grid and a time step $\Delta t = 1$.

4.1. Convergence test

We start with spatial and temporal convergence tests of the numerical scheme. In order to obtain the spatial convergence rate, we perform a number of simulations with increasingly finer grids $\Delta x = 100/2^n$ for $n = 6, 7, 8, 9, 10$. The initial condition is $h(x, 0) = 0.5[b + (h_{\infty} - b) \tanh(3(x - 50))]$.

The rate of convergence is defined as the ratio of successive errors: $e_{\Delta x/2^n} := h_{\Delta x/2^n} - (h_{\Delta x/2^{n-1}} + h_{\Delta x/2^{n+1}})/2$. Here $e_{\Delta x/2^n}$ is a discrete $l_2$-norm and is defined as $e_{\Delta x/2^n} := \sum_{i=1}^{M^n} e_i^2 / N$. Fig. 3 shows a log-log plot of $l_2$-norm of errors (circle) against various mesh grids at $T = 100$ with a linear fitting (solid line). The second-order accuracy with respect to space and time is observed as expected from the discretization.

Furthermore, we consider the CPU time in seconds for the convergence test. Tests were performed on a system with a 3-GHz Intel Pentium CPU and 3-GB RAM, loaded with C++. If we refine the spatial and temporal grids by a factor of 2, the CPU time should increase by a factor of 4. As can be observed from Table 1, using the multigrid method, the CPU time increases by a factor of approximately four. This result indicates that the computational complexity of the multigrid method is indeed $O(N)$.
4.2. Evolution of thin liquid film

In this subsection, we show the evolution of the thin liquid film and its self-similar solution. The initial condition is the same as in Eq. (20). Solutions are computed up to time $T = 5000$. Fig. 4(a) shows the evolutions of the thin film height $h$. We can observe that there exits a point $x_0 = 50.02$, which satisfies $h_t(x_0, t) = 0$ and $h_x(x_0, t) \neq 0$ for all $t \geq t_0 = 891$ with $A = -1.81 e^{-4}$. In Fig. 4(b), the starred and circled lines represent the evolutions of $h_{\text{max}} - h_\infty$ and $b - h_{\text{min}}$, respectively. They start from zero and converge quickly to constant values. Fig. 4(c) shows snapshots of the thin film shapes at $t_0$ and $t = 5000$. A comparison of the numerical solutions at $t = 5000$ and from a self-similar solution at $t_0$ is shown in Fig. 4(d). This result shows that our numerical algorithm can generate a self-similar solution.

4.3. Effect of the finite computational domain size

The original equation is in the infinite domain; however, to get numerical solutions by using a finite difference method, the domain must be truncated and suitable boundary conditions should be applied. In order to show the effect of the computational domain size on the numerical solutions, we take a set of different domains $\Omega = (-25, 25), (-50, 50), \text{ and } (-100, 100)$, with a fixed space step size $\Delta x = 50/512$. The initial condition is $h(x, 0) = 0.5[h_\infty + b - (h_\infty - b) \text{tanh}(3x)]$, where $h_\infty = 0.3$ and $b = 0.1$. We run the computation up to $T = 2000$. Fig. 5 shows a comparison of the thin liquid film profiles in three different computational domains. This result suggests that the effect of the domain size on the numerical result is negligible as long as the computational domain is sufficiently large and the temporal evolution is not too long.

Next, we perform a long time evolution. The initial condition is $h(x, 0) = h_\infty$, if $x < 0$ and $h(x, 0) = b$, otherwise, where $h_\infty = 0.3$ and $b = 0.1$. Fig. 6(a) shows the evolution of the thin film in the domain $\Omega = (-25, 25)$ up to time $T = 8000000$. As can be seen, the numerical solution becomes a steady state (thick solid line) that is nearly linear profile owing to the boundary conditions. The symbol 'o' indicates the self-similar solution, $f(\phi)$, from the ODE solver. We can observe a good agreement between $f(\phi)$ and one of evolution profiles. In Fig. 6(b), we show the discrete $l_2$-norm of the difference between the numerical solution and the self-similar solution (circled line in Fig. 6(a)) obtained by solving Eq. (10) with

![Fig. 3. Log-log plot of $l_2$-norm of errors (circle) against various mesh grids at $T = 100$ with a linear fitting (solid line).](image)

Table 1

<table>
<thead>
<tr>
<th>Grid</th>
<th>256</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU time</td>
<td>5.787</td>
<td>28.360</td>
<td>114.563</td>
<td>442.828</td>
</tr>
</tbody>
</table>

Factor

| Factor | 4.90 | 4.04 | 3.87 |

![Fig. 4. (a) Evolution of the thin film height $h$ with $h_\infty = 0.3$ and $b = 0.1$. (b) The starred and circled lines represent evolutions of $h_{\text{max}} - h_\infty$ and $b - h_{\text{min}}$, respectively. (c) Snapshots of the thin film shapes at $t_0 = 891$ and $t = 5000$. (d) Profiles of $h(x, 5000)$ and $h(\phi(5000)(x - x_0) + x_0, t_0)$.](image)
4.4. Effect of parameters, \( h_\infty \), \( b \), and \( t_0 \)

In this subsection, we study the effect of the parameters, \( h_\infty \), \( b \), and \( t_0 \) on self-similarity. The initial condition is the same as in Eq. (20) with different \( h_\infty \) and \( b \) values. Here, we take \( t_0 = 2000 \). With a set of numerical solutions of the thin film at \( t = 2200, 2400, \ldots, 4000 \), the averaged values of \( A \) are calculated and listed in Table 2. From the results, we observe that although different values of \( h_\infty \) and \( b \) are used, the value of \( A \) is nearly the same. However, with different \( t_0 \) values, the value of \( A \) is different (see Table 3). Here \( h_\infty = 0.3 \), \( b = 0.1 \), \( N = 2048 \), and \( \Delta t = 0.5 \) are used.

\[
A = -1.81e^{-4}.
\]

The result shows that the error starts at a moderate initial value, decreases for a while, then increases for longer times.

4.5. Simulation with another initial condition

To demonstrate the independence of the initial profiles on generating a self-similar solution, we perform a numerical test with another initial condition

\[
h(x, 0) = \begin{cases} 
0.5[h_\infty + h_f - (h_\infty - h_f) \tanh(3(x - 45))] \\
0.5[h_\infty + b - (h_\infty - b) \tanh(3(x - 50))] & \text{otherwise,}
\end{cases}
\]

Here \( h_\infty = 0.3 \), \( h_f = 0.35 \), and \( b = 0.1 \) are used. Solutions are computed up to time \( T = 20000 \). Fig. 7(a) shows the temporal evolution of the numerical solution. Fig. 7(b) shows snapshots of the thin film shapes at \( t_f = 9542 \) and \( t_f = 20000 \). Fig. 7(c) shows profiles of \( h(x, 20000) \) and \( h(\alpha(20000)(x - x_0) + x_0, t_0) \), where \( \alpha(20000) = 1.201 \) and \( x_0 = 50.13 \). The results indicate that the numerical self-similar solution is in good agreement. Further, the self-similar solution is independent of initial profiles.

5. Conclusions

In this article, we numerically investigated the self-similar solution for the nonlinear diffusion equation \( h_t = -(h^2 h_{xxx})_x \), which arises in the context of surface-tension driven flow of thin viscous liquid film. The suggested self-similar solution is

\[
h(x, t) = h(\alpha(t)(x - x_0) + x_0, t_0) = f(\alpha(t)(x - x_0)) \text{ and } \alpha(t) = [1 - 4A(t - t_0)]^{-1/4},
\]

where \( A \), \( x_0 \), and \( t_0 \) are constants and \( \alpha(t) \) is a reference time. To numerically solve the governing equation, we used the Crank–Nicolson finite difference method with a nonlinear multigrid solver. We provided numerical algorithms for calculating the constants, \( A \), \( x_0 \), and \( t_0 \) in detail. To find a self-similar solution of the equation, we numerically solved the partial differential equation with a simple step-function like initial condition until the solution reached the reference time \( t_0 \). Then, we took \( h(x, t_0) \) as the self-similar solution \( f(x) \). Various numerical experiments were performed to demonstrate that \( f(x) \) is indeed a self-similar solution. In particular, we found a self-similar solution could be obtained from arbitrary initial profiles.

Acknowledgments

The corresponding author (J. Kim) was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0023794). The authors thank the help of Dr. Youngsoo Ha and Professor Tim Myers in using the ODE solver, bvp5c. The authors greatly appreciate the reviewers for their constructive and insightful comments on this article.
Eqs. (15) and (16) as full approximation storage (FAS) multigrid method. Let us rewrite Appendix B

Simulation with another initial condition. (a) Temporal evolution of the numerical solution. (b) Snapshots of the thin film shapes at $t_0 = 9542$ and $t = 20000$. (c) Profiles of $h(x, 20000)$ and $h(x(20000)(x - x_0) + x_0, t_0)$.

**Appendix A**

A MATLAB m-file, which calculates the self-similar solution by solving Eq. (10).

```matlab
function selfODE % This solves $f'''' = -(Af'' + 3f'f''')f^3$
    global heq b
    h = 0.1; heq = 0.3; A = -1.81e-4; x = linspace(-25, 25, 3000);
    sol = bvp5c(@twoode, @twobc, bvpinit(x, @mat4init)); y = deval(sol, x);
    plot(x, y(1,:), 'k-');
    % Compute an approximate solution of the discrete system (15) and (16),
    % which implies performing $v$ smoothing steps with the initial
    % approximations $h^n_k$, $\mu^n_k$, source terms $\phi^n_k$, $\psi^n_k$, and
    % SMOOTH relaxation operator to get the approximations $\bar{h}^m_k$, $\bar{\mu}^m_k$. One SMOOTH
    % relaxation operator step consists of solving the system (21) and
    % (22) given below by $2 \times 2$ matrix inversion for each $i$.
    % (1) Presmoothing
    % Compute $[\bar{h}^m_k, \bar{\mu}^m_k]$ by applying $v$ smoothing steps to $[h^n_k, \mu^n_k]$
    % $\bar{h}^m_k - \bar{\mu}^m_k = \text{SMOOTH}_v(h^n_k, \mu^n_k, N_k, \phi^n_k, \psi^n_k)$,
    % where $v$ is the number of smoothing relaxation sweeps. $h^n_k$ and
    % $\bar{h}^{m+1}_k$ are the approximations of $h^{n+1}_k$ before and after an FAS cycle.
    % (2) Compute the defect: $(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) = (\phi^n_k, \psi^n_k) - N_k(\bar{h}^m_k, \bar{\mu}^m_k)$.
    % (3) Restrict the defect and $(\bar{h}^m_k, \bar{\mu}^m_k)$
    % $(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) = \bar{L}^{-1}(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k)$,
    % $N_k(\bar{h}^m_k, \bar{\mu}^m_k) = \bar{L}^{-1}(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k)$.
    % (4) Compute the right-hand side
    % $(\phi^n_k, \psi^n_k) = (\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) + N_{k-1}(\bar{h}^m_{k-1}, \bar{\mu}^m_{k-1})$.
    % (5) Compute an approximate solution $[\tilde{h}^m_{k-1}, \tilde{\mu}^m_{k-1}]$ of the coarse grid
equation on $\Omega_{k-1}$, i.e.,
    % $N_{k-1}(\tilde{h}^m_{k-1}, \tilde{\mu}^m_{k-1}) = (\phi^n_{k-1}, \psi^n_{k-1})$.
```

**Appendix B**

To solve the discrete system (15) and (16), we use a nonlinear full approximation storage (FAS) multigrid method. Let us rewrite Eqs. (15) and (16) as

$$N(h^{n+1}_k, \mu^{n+1}_k) = (\phi^n_k, \psi^n_k),$$

where the left- and right-hand side equations contain $n + 1$ and $n$ time level terms, respectively. Let us assume that a sequence of grids $\Omega_k (\Omega_{k-1}$ is coarser than $\Omega_k$ by a factor of 2).

**FAS multigrid cycle**

$$h^{m+1}_k, \mu^{m+1}_k = \text{FASCycle}(k, h^m_k, \mu^m_k, N_k, \phi^n_k, \psi^n_k, v),$$

where $v$ is the number of smoothing relaxation sweeps. $h^n_k$ and $h^{m+1}_k$ are the approximations of $h^{n+1}_k$ before and after an FAS cycle.

**Presmoothing**

Compute $[\bar{h}^m_k, \bar{\mu}^m_k]$ by applying $v$ smoothing steps to $[h^n_k, \mu^n_k]$

$$[\bar{h}^m_k, \bar{\mu}^m_k] = \text{SMOOTH}_v(h^n_k, \mu^n_k, N_k, \phi^n_k, \psi^n_k),$$

which implies performing $v$ smoothing steps with the initial approximations $h^n_k$, $\mu^n_k$, source terms $\phi^n_k$, $\psi^n_k$, and SMOOTH relaxation operator to get the approximations $\bar{h}^m_k$, $\bar{\mu}^m_k$. One SMOOTH relaxation operator step consists of solving the system (21) and (22) given below by $2 \times 2$ matrix inversion for each $i$.

**(1) Presmoothing**

Compute $[\bar{h}^m_k, \bar{\mu}^m_k]$ by applying $v$ smoothing steps to $[h^n_k, \mu^n_k]$

$$[\bar{h}^m_k, \bar{\mu}^m_k] = \text{SMOOTH}_v(h^n_k, \mu^n_k, N_k, \phi^n_k, \psi^n_k),$$

which implies performing $v$ smoothing steps with the initial approximations $h^n_k$, $\mu^n_k$, source terms $\phi^n_k$, $\psi^n_k$, and SMOOTH relaxation operator to get the approximations $\bar{h}^m_k$, $\bar{\mu}^m_k$. One SMOOTH relaxation operator step consists of solving the system (21) and (22) given below by $2 \times 2$ matrix inversion for each $i$.

**(2) Compute the defect:** $(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) = (\phi^n_k, \psi^n_k) - N_k(\bar{h}^m_k, \bar{\mu}^m_k)$.

**(3) Restrict the defect and $(\bar{h}^m_k, \bar{\mu}^m_k)$**

$$(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) = \bar{L}^{-1}(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k),$$

$$N_k(\bar{h}^m_k, \bar{\mu}^m_k) = \bar{L}^{-1}(\bar{d}^{m+1}_k, \bar{d}^{m+1}_k).$$

**(4) Compute the right-hand side**

$$(\phi^n_k, \psi^n_k) = (\bar{d}^{m+1}_k, \bar{d}^{m+1}_k) + N_{k-1}(\bar{h}^m_{k-1}, \bar{\mu}^m_{k-1}).$$

**(5) Compute an approximate solution $[\tilde{h}^m_{k-1}, \tilde{\mu}^m_{k-1}]$ of the coarse grid equation on $\Omega_{k-1}$, i.e.,**

$$N_{k-1}(\tilde{h}^m_{k-1}, \tilde{\mu}^m_{k-1}) = (\phi^n_{k-1}, \psi^n_{k-1}).$$
If $k = 1$, we explicitly invert a $2 \times 2$ matrix to obtain the solution. If $k > 1$, we solve (23) by performing an FAS $k$-grid cycle using $\{h_{k-1}^m, \mu_{k-1}^m\}$ as an initial approximation:

$$\left(\hat{h}_{k-1}^m, \hat{\mu}_{k-1}^m\right) = \text{FAScycle}(k - 1, h_{k-1}^m, \mu_{k-1}^m, N_{k-1}, \phi_{k-1}, \psi_{k-1}, \nu).$$

(6) Compute the coarse grid correction (CGC):

$$\hat{e}_{k-1}^m = h_{k-1}^m - \hat{h}_{k-1}^m, \quad \hat{e}_{k-2}^m = \hat{h}_{k-1}^m - \hat{\mu}_{k-1}^m.$$

(7) Interpolate the correction:

$$\hat{e}_{k-1}^m = \hat{e}_{k-1}^m + \hat{e}_{k-2}^m,$$

$$\hat{\mu}_{k-1}^m = \hat{\mu}_{k-1}^m + \hat{e}_{k-2}^m.$$

(8) Postsmoothing:

$$\{h_{k-1}^{m+1}, \mu_{k-1}^{m+1}\} = \text{SMOOTH}\left(h_{k-1}^m, \hat{\mu}_{k-1}^m, N_k, \phi_k, \psi_k\right).$$

This completes the description of a nonlinear FAS cycle. See the reference text [40] for additional details and background.

References