APC523 Final Project: Spreading of a Viscous Mound

Clara Martín Blanco and Samuel Koblensky

May 6, 2024

1 Introduction

The goal of this project is to solve the equations which govern the motion of an axisymmetric, viscous drop spreading under gravity. A schematic of this problem is depicted in Fig. 1. The governing non-linear PDE for the spreading viscous drop is [1, 2]:

$$\delta^3\left[\frac{\partial^2\delta}{\partial r^2}+\frac{1}{r}\frac{\partial\delta}{\partial r}\right]+3\delta^2\left[\frac{\partial\delta}{\partial r}\right]^2-3\frac{\nu}{g}\frac{\partial\delta}{\partial t}=0 \quad (1)$$

Where $\delta(r,t)$ is the thickness of the drop, r is the radial distance, ν is the viscosity of the fluid, g is gravitational acceleration, and t is time. The boundary conditions for this problem are given by the following equations.

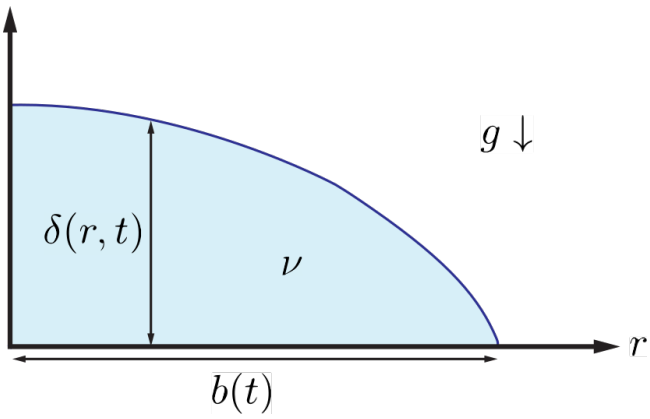

Figure 1. Schematic displaying the setup for viscous drop spreading.

$$\delta(r=b(t),t)=0 \quad (2)$$

$$\frac{\partial\delta}{\partial r}(r=0,t)=0 \quad (3)$$

Equation 2 defines the radius of the drop, $b(t)$, where the thickness $\delta(b,t)=0$. Equation 3 ensures the solution is smooth on the axis of symmetry, which is necessary to have a physical solution.

A final condition is given by the following integral expression, which enforces conservation of volume in the drop.

$$2\pi\int_0^{b(t)} r\delta(r,t)dr = constant \quad (4)$$

The problem defined by equations 1-4 consists of a non-linear, second order, parabolic PDE with an integral constraint and two boundary conditions. When studying this problem analytically, it is possible to find a similarity solution which gives the following two analytical results [2].

$$b(t)\sim t^{1/8} \quad (5)$$

$$\delta(r,t)=\left(\frac{3a\nu}{16\pi gt}\right)^{1/4} f\left(r\left(\frac{3\nu\pi^3}{a^3gt}\right)^{1/8}\right) \quad (6)$$

Where f is the similarity solution function and a is a constant. For this project, we focus on solving the PDE, and we evaluate our solutions by comparing the results against the theoretical equations (5-6).

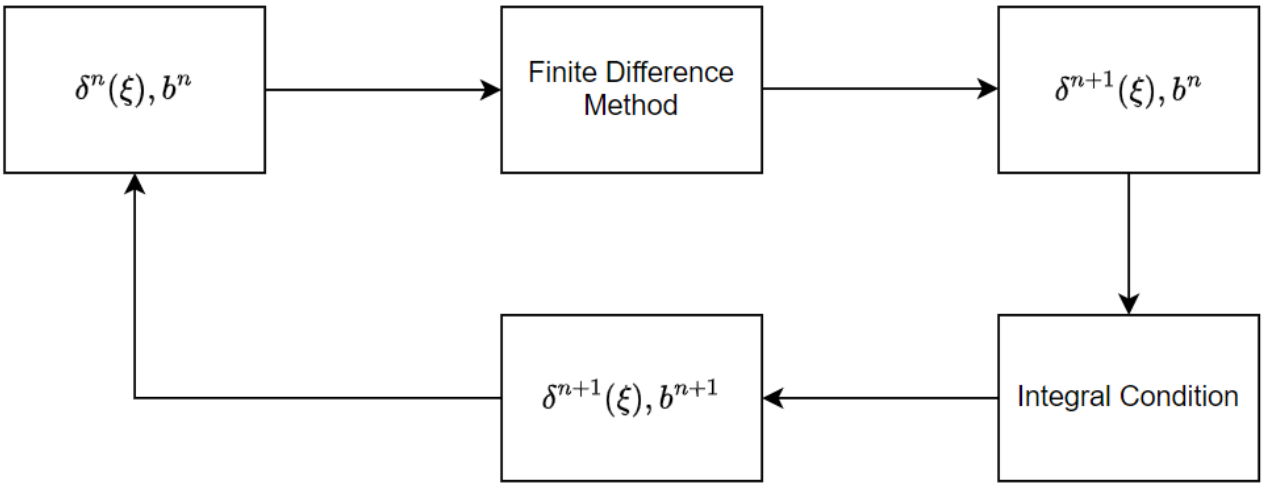

Figure 2. Diagram showing the solution procedure. A finite difference method is first used to advance the δ to the next time step. Then the integral condition advances the radius b. This process is repeated until all time steps are completed.

2 General Solution Procedure

The solution of the nonlinear system eqns. 1-4 can be formulated numerically by first recasting the governing equation to solve for $\delta(\xi, t)$, where $\xi = r/b$. This step simplifies the solution procedure since for each time step, $\xi^n \in [0, 1]$ is constant while $r^n \in [0, b^n]$ is not constant. Consequently, formulating the problem in ξ means the physical domain does not need to change in size with each time step. By performing a change in variables, the governing equation (1) can be re-expressed as

$$\frac{\delta^3}{b^2}\left[\frac{\partial^2\delta}{\partial\xi^2} + \frac{1}{\xi}\frac{\partial\delta}{\partial\xi}\right] + \frac{3\delta^2}{b^2}\left[\frac{\partial\delta}{\partial\xi}\right]^2 - 3\frac{\nu}{g}\frac{\partial\delta}{\partial t} = 0 \tag{7}$$

and the boundary conditions (2-3) become $\delta(\xi = 1, t) = 0$ and $\frac{\partial\delta}{\partial t}(\xi = 0, t) = 0$. Now each new time step can be fully solved in two steps. First, $\delta^{n+1}(\xi)$ is calculated. Then, b^{n+1} is found by satisfying the integral condition (4). The solution method is summarized graphically in Figure 2. These two steps can be performed using a variety of numerical methods, and the specific methods chosen for this project are discussed in §3.

3 Algorithms

We solve the PDE system defined by equation 7 and its corresponding boundary and integral conditions using two methods. First, an explicit scheme is used to solve for each successive time step, and a shooting method using a trapezoidal integration scheme is used to meet the integral condition.

Then, an implicit formulation of the finite difference scheme is applied. The stability and results from these two methods are analyzed and compared in §4.

3.1 Explicit Method with Adaptive Time-Stepping

A simple explicit method is first used to get a baseline solution to compare against the more accurate method. The governing equation is expressed explicitly with N points using 2^{nd} order finite difference approximations for the first and second spatial derivative as follows:

$$\frac{3\nu}{g}\frac{\delta_i^{n+1} - \delta_i^n}{\Delta t} = \frac{(\delta_i^n)^3}{(b^n)^2}\left[\frac{\delta_{i+1}^n - 2\delta_i^n + \delta_{i-1}^n}{\Delta\xi^2} + \frac{1}{\xi_i}\frac{\delta_{i+1}^n - \delta_{i-1}^n}{2\Delta\xi}\right] + \frac{3(\delta_i^n)^2}{(b^n)^2}\left[\frac{\delta_{i+1}^n - \delta_{i-1}^n}{2\Delta\xi}\right]^2 \tag{8}$$

With this formulation, the boundary conditions become $\delta_0^{n+1} = \delta_1^{n+1}$ and $\delta_N^{n+1} = 0$, where we use a 1^{st} order approximation for the derivative condition. After solving for δ^{n+1}, we find b^{n+1} by using a

shooting method and a trapezoidal integral approximation as follows:

$$2\pi \int_0^{b(t)} r\delta(r,t)dr \simeq \frac{\pi}{2}\sum_{i=1}^{N}(b^{n+1})^2(\xi_i^2 - \xi_{i-1}^2)(\delta_i^n + \delta_{i-1}^n) = constant \tag{9}$$

The base radius b^{n+1} for each time step is calculated using an iterative shooting method, aimed at conserving the volume of the fluid. Specifically, b^{n+1} is adjusted until the volume computed through equation (9) matches the initial volume with some precision $\epsilon \sim O(10^{-6})$. This approach involves iteratively guessing the value of b^{n+1}, recalculating the fluid's volume for each guess, and refining the guess based on whether the recalculated volume is above or below the initial volume. The process continues until the volume difference falls within the specified tolerance, ensuring that volume conservation is maintained throughout the simulation with high accuracy.

The expected stability of this method can be found by analyzing the method in the form of $\delta^{n+1} = \mathbb{A}^n\delta^n$, where $\mathbb{A}^n$ is the n^{th} time step of the matrix representing the explicit method used to find the $(n+1)^{st}$ step solution. $\mathbb{A}$ depends on δ and changes with each step due to the non-linearity of the problem. The stability criterion is $\rho(\mathbb{A}) \leq 1$. This cannot be solved analytically due to the non-linearity. Instead, an adaptive time-stepping approach is implemented to modify the time step Δt until the stability criterion is met. For each time step, a value for Δt is selected. If $\rho(\mathbb{A}) > 1$, then Δt is reduced until stability is met. This ensures the explicit method will reach a valid solution in an efficient amount of time.

3.2 Implicit Method

A second, implicit solution method is also used as a comparison against the simple explicit solver. The governing equation is formulated with N points using the following implicit finite difference scheme:

$$\frac{3\nu}{g}\frac{\delta_i^{n+1} - \delta_i^n}{\Delta t} = \frac{(\delta_i^{n+1})^3}{(b^n)^2}\left[\frac{\delta_{i+1}^{n+1} - 2\delta_i^{n+1} + \delta_{i-1}^{n+1}}{\Delta\xi^2} + \frac{1}{\xi_i}\frac{\delta_{i+1}^{n+1} - \delta_{i-1}^{n+1}}{2\Delta\xi}\right] + \frac{3(\delta_i^{n+1})^2}{(b^n)^2}\left[\frac{\delta_{i+1}^{n+1} - \delta_{i-1}^{n+1}}{2\Delta\xi}\right]^2 \tag{10}$$

With boundary conditions represented again as $\delta_0^{n+1} = \delta_1^{n+1}$ and $\delta_N^{n+1} = 0$. An integral method is again used to solve for b^{n+1}, this time a more accurate Gauss-Legendre scheme:

$$2\pi \int_0^{b(t)} r\delta(r,t)dr \simeq \sum_{i=1}^{k} w_i f(r_i) = constant \tag{11}$$

Where $r_i = \frac{b-a}{2}x_i + \frac{b+a}{2}$, are the nodes remapped from the interval [-1,1] to the interval $[0, b^{n+1}]$, w_i are the weights and $f(r_i) = 2\pi r_i\delta(r_i,t)$ is the function being integrated, which includes the 2π factor due to integration over a radial dimension.

A shooting method with a maximum error of $\epsilon = 10^{-6}$ is again used to find b^{n+1}.

To solve equation (10), the numerical method is first reformulated into matrix form $\mathbb{A}\delta^{n+1} = v^n$, where $v^n = -\delta^n$ for inner i and $v^n = 0$ for boundary points $i = 0$ and $i = N$. This reformulation produces a tri-diagonal matrix $\mathbb{A} = \mathbb{A}(\delta^{n+1})$ which changes with each time step due to non-linearities and takes the following form:

$$\mathbb{A} = \begin{bmatrix} 1 & -1 & & & & \\ A_2 & B_2 & C_2 & & & \\ & A_3 & B_3 & C_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & A_{N-1} & B_{N-1} & C_{N-1} \\ & & & & & 1 \end{bmatrix} \tag{12}$$

$$A_i = \alpha_i\kappa_i\left[\frac{1}{(\Delta\xi)^2} - \frac{1}{2\xi_i\Delta\xi}\right] + \frac{\beta_i\kappa_i}{4(\Delta\xi)^2}\left[\delta_{i-1}^{n+1} - \delta_{i+1}^{n+1}\right] \tag{13}$$

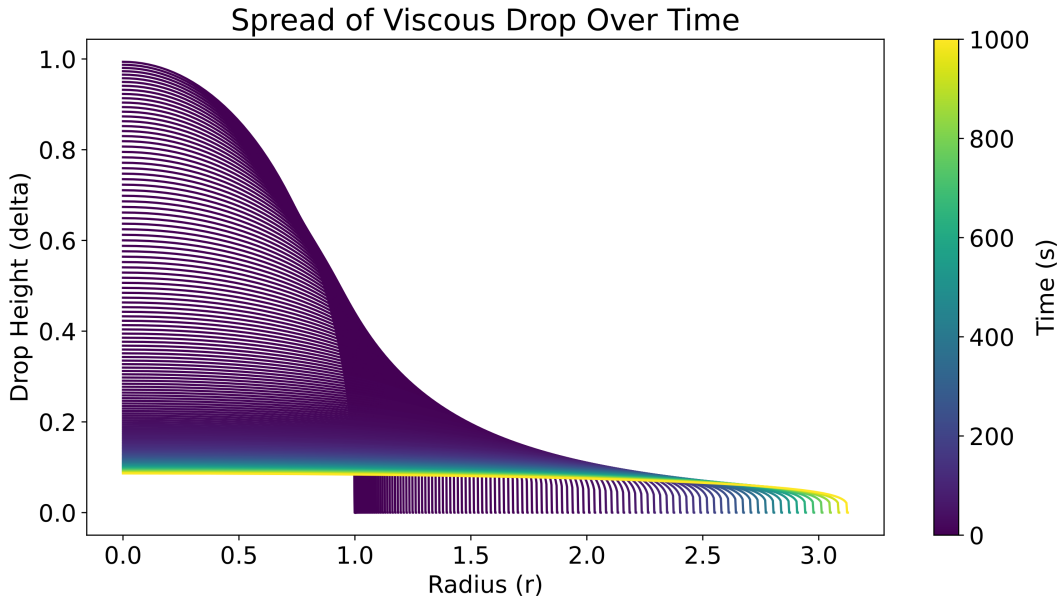

Figure 3. Plot of the physical drop profile vs time. Time is represented by the color. As time progresses, the drop spreads and becomes flat as expected.

$$B_i = -1 - \frac{2\alpha_i \kappa_i}{(\Delta\xi)^2} \tag{14}$$

$$C_i = \alpha_i \kappa_i \left[\frac{1}{(\Delta\xi)^2} + \frac{1}{2\xi_i \Delta\xi}\right] + \frac{\beta_i \kappa_i}{4(\Delta\xi)^2}\left[\delta_{i+1}^{n+1} - \delta_{i-1}^{n+1}\right] \tag{15}$$

and $\alpha_i = (\delta_i^{n+1})^3/(b^n)^2$, $\beta_i = (3\delta_i^{n+1})^2/(b^n)^2$, and $\kappa_i = g\Delta t/3\nu$. For each time step, the matrix $\mathbb{A}$ is calculated using (12-15), and then the new step $\delta^{n+1} = \mathbb{A}^{-1}v^n$. Since $\mathbb{A}$ is diagonally dominant, we expect it to be invertible. Since this is a 1D problem, $\mathbb{A} \in [N, N]$ is relatively small and can simply be solved using direct matrix inversion for small to moderately large N without needing an iterative method.

Since the governing equation is parabolic, we expect the change to be most rapid at the start, and the motion of the drop will slow with time. Therefore we can implement an increasing adaptive time step to accurately model time close to $t = 0$ while also rapidly calculating new steps for $t >> 0$. We use a simple 10% adaptive scheme, where $\Delta t^{n+1} = 1.1\Delta t^n$.

4 Results

4.1 Overall Solution

Both methods were able to successfully solve the governing equations (1-4) and model the spreading of a drop vs time. For these solutions, we selected $g = 9.81$ and $\nu = 1$, and initialized the problem with a spherical drop of radius $b(t = 0) = 1$. This can be thought of as a very large, and very viscous drop in Earth gravity, but the physical interpretation of the problem is beyond the scope of this report. Figure 3 shows the time history of the drop as is spreads from $t = 0$ to $t = 1000$. As expected, the drop spreads with time, becoming flatter and larger in radius.

The conservation of volume is validated in Figure 4, where the absolute error in volume ($|V(t) - V_0|$) is plotted vs time.

The solution can also be compared against the theoretical result that for long times, $b(t) \sim t^{1/8}$ [2], which is shown in Figure 5, and plotted in the similarity solution units given by equation 6 in Figure 6. From these plots, it is apparent that $b(t)$ matches the expected long-term behavior, and the shape rapidly approaches a similarity solution. These results confirm that the numerical method is accurately modeling the governing equation.

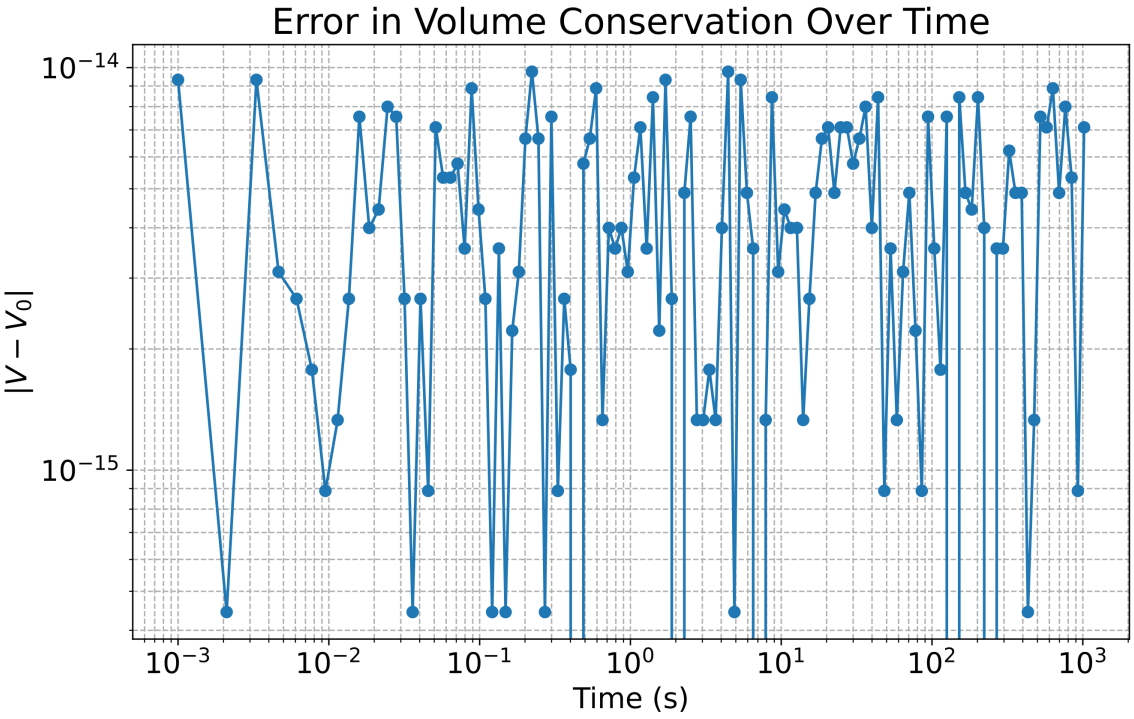

Figure 4. Plot of error in volume vs time. The error remains below the specified error threshold ϵ. For this plot, $\epsilon = 10^{-14}$.

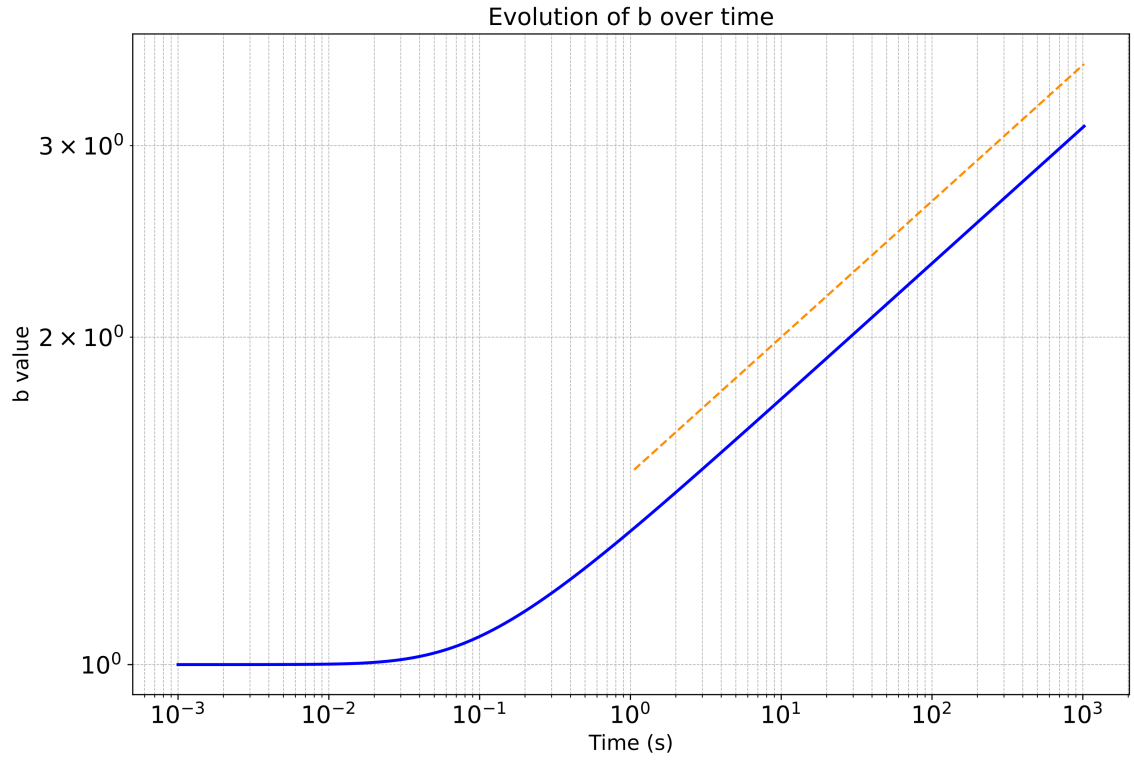

Figure 5. Plot of radius vs time (blue) against the $t^{1/8}$ scaling (orange). As t increases, the scaling becomes more accurate as expected since the solution approaches a similarity solution.

Figure 6. Plot of the drop profiles converted into similarity variables for $N = 1024$. For long times, the solution has converged to a self-similar solution, seen as a yellow line. Here, $\eta = r\left((3\nu\pi^3)/(a^3 gt)\right)^{1/8}$ and $f(\eta) = \delta(r,t)\left((16\pi gt)/(3a\nu)\right)^{1/4}$

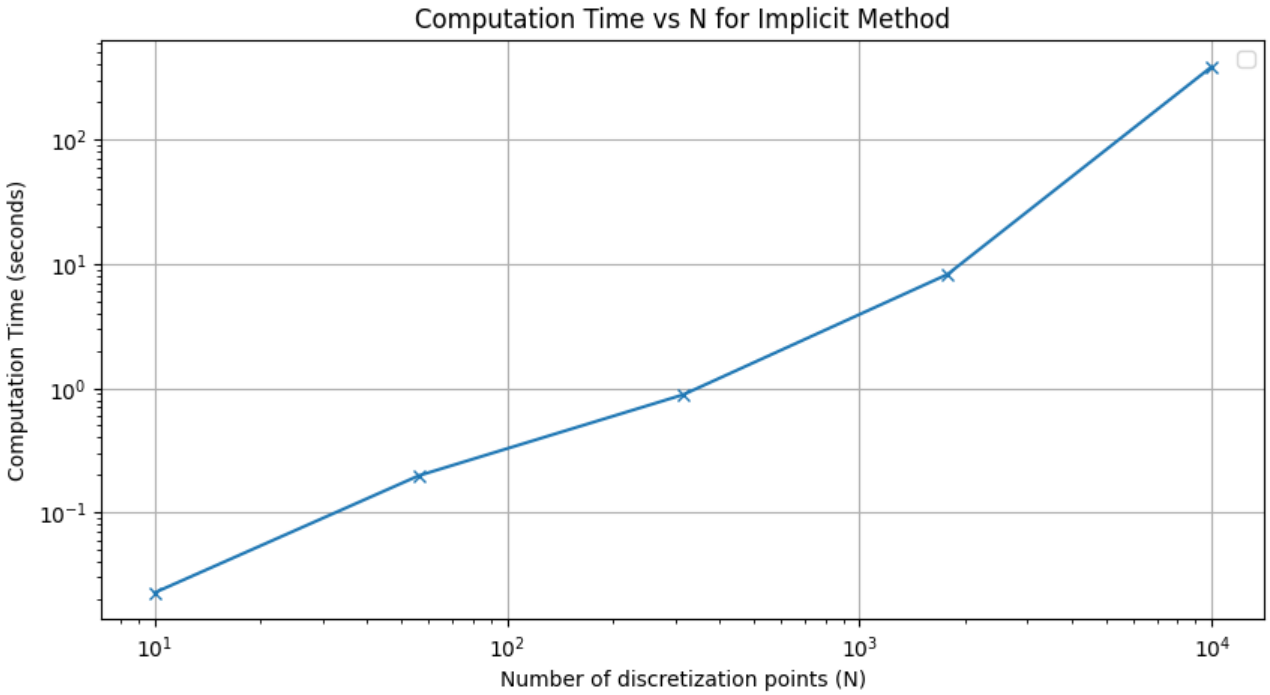

Figure 7. Plot of computation time vs N for both explicit and implicit methods.

4.2 Comparing Explicit and Implicit Methods

In the simulation work involving both explicit and implicit methods, each step of the implicit method is slower due to the required matrix inversion, but it offers greater stability and allows for a significantly larger time step. This stability is essential, especially when dealing with stiff equations and higher numerical diffusivity, which effectively reduces the risk of numerical oscillations and instabilities, common issues with the explicit method. The explicit method, although faster per step owing to straightforward computations that avoid complex matrix inversions, is constrained by the need for smaller stability time steps. According to the Courant–Friedrichs–Lewy (CFL), the time step in explicit simulations must be sufficiently small to ensure stability, which can lead to significant computational demands over long simulation periods or with fine spatial resolutions. Conversely, the implicit method's capability to handle larger time steps often compensates for its higher computational cost per step. This feature is particularly beneficial in simulations where dynamics evolve slowly, which is expected of a parabolic PDE. Additionally, the implicit approach tends to yield smoother solutions for transient problems, effectively minimizing the amplitude of any oscillations—a frequent challenge in explicit schemes.

In evaluating the time complexity of both schemes, we find the explicit scheme to be incredibly inefficient. When N exceeded 64, the solution would not complete, which indicates either it no longer has any stable time steps, or the time steps for stability were very small.

The implicit scheme, however, was able to efficiently run up to $N = 10^4$. The compution time vs N is plotted in Figure 7. For small $N \in [10^1, 10^3]$, the scheme is very efficient and scales approximately as $O(N)$. We see, however, for higher N the solution begins to slow down. This is expected, since direct matrix inversion has a time complexity of $O(N^3)$ at worst. The likely reason why our matrix inversion is more efficient is because the matrix is tri-diagonal. The built-in Python matrix inversion function is optimized for simple matricies like these, and is therefore effient while N is small.

5 Summary

In this report, we developed two numerical methods to solve the nonlinear, parabolic PDE governing the spread of a viscous drop under gravity. Using a finite difference method, this problem can be solved using two steps for each time step. Firstly, the governing equation is used to solve for the shape of the drop in the next time step. Then, the integral condition is used to solve for the radius of

the drop at the next time step.

We use both an explicit and implicit numerical formulation of the problem and compare the results. Both methods reproduce the expected long-term trends of drop radius growing as $b(t) \sim t^{1/8}$ and the drop satisfying a similarity solution following [1, 2]. The implicit solver, however, is far faster due to its higher stability, which allows for larger time steps. By applying an adaptive time stepping method, the implicit solver is accurate for all time scales and can solve for long times. As a result, this project demonstrated the importance of selecting an appropriate numerical scheme for a given problem. While the explicit scheme works, it is very inefficient and cannot reasonably be used for higher values of N. The implicit scheme is both efficient and accurate, although higher order schemes (both finite difference and time-stepping) may be applied to increase the accuracy of the method for future work.

6 Contributions

Both group members had equal contributions to the project. Sam was responsible for formulating the overall solution procedure and coding the initial explicit solution method. He also performed analytical stability analysis for both explicit and implicit methods. Clara was responsible for coding the implicit solver, making the code more modular and readable, improving the plot formats, and performing numerical stability analysis. Both members contributed equally to the report and presentation slides.

References

[1] N. Bergemann, A. Juel, and M. Heil, "Viscous drops on a layer of the same fluid: From sinking, wedging and spreading to their long-time evolution," *Journal of Fluid Mechanics*, vol. 843, pp. 1–28, 2018.

[2] L. Deike, "MAE501 problem set," 2023.