CONSERVATIVE SIMULATIONS OF ATOMIZATION

APPLYING THE HEIGHT FUNCTION METHOD TO RUDMAN DUAL GRIDS

by

Kristopher Thomas Olshefski

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Mechanical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

November 2019

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

LIST OF FIGURES – CONTINUED

NOMENCLATURE

μ	Dynamic viscosity
$\mathbf{n}$	Normal vector
$\mathbf{u}$	Velocity vector

ABSTRACT

The atomization process is significantly affected by the surface tension force, which controls the size and distribution of droplets. The surface tension force is directly proportional to the interface curvature and an accurate calculation of curvature is essential for predictive simulations of atomization. Furthermore, methods that consistently and conservatively transport momentum, which is discontinuous at the gas-liquid interface, are necessary for robust and accurate simulations. The height function method is a common technique to compute an accurate curvature as it is straightforward to implement and provides a second-order calculation. Additionally, using a Rudman dual mesh (Int. J. Numer. Meth. Fluids, 1998), which discretizes density on a twice as fine mesh, provides consistent and conservative discretizations of mass and momentum.

This paragraph goes from curvature is important, to consistent/conservation, to height function, and then back to consistent/conservation.

This work extends the standard height function method to include information from the Rudman dual mesh. When a dual grid is used, the standard height function method fails to capture fine grid interface perturbations and these perturbations can grow. The proposed method leverages a fine-grid height function method to compute the fine-gird interface perturbations and uses a fine-grid velocity field to oppose the fine-grid perturbations. This approach maintains consistent mass and momentum transport while also providing accurate interface transport that avoids non-physical dynamics. The method is tested using an oscillating droplet test case and compared to a standard height function.

What are the big picture implications of this work.

INTRODUCTION

The study of fluid dynamics covers a wide array of phenomena across large spans of scales. Current research can be found exploring everything from biomedical simulations of capillary flows [?] to astrophysical approximations of star formations [?]. Researchers are studying bio-inspired dynamics of animals to gain insight into efficiency shortcomings of man made vessels [?]. Other research focuses on oceanic current modeling to gain a deeper understanding of the natural world [?]. The focus of this research however, is on multiphase flows. In the context of this research, interest is restricted to flows where both a liquid and gas are present. These flows are referred to as gas-liquid flows, multiphase flows, or free surface flows in literature [?, ?, ?]. Examples of gas-liquid flows can be seen throughout the natural world as well as industrial application and may be some of the most wide spread and common flow types on Earth. Bubbles rising in a carbonated drink, rain falling through the air, paint exiting a can of spray paint, a wave on the surface of the ocean; these are all examples of gas-liquid multiphase flows. These types of flows are of particular academic interest due to the ubiquity with which they emerge in industrial and engineering applications. For example, the increasing efficiency of internal combustion engines over the past several decades can largely be attributedto an increased understanding of the physics at work during fuel combustion. For efficient combustion to occur in a gasoline engine the fuel must first be atomized. Atomization is the process by which the fuel is broken from a stream into a fine mist or distribution of smaller droplets.

need a reference for a statement like this

For engineering applications there are two main pathways by which multiphase flows are studied. Experimentation is a common method of study inside many scientific disciplines; the field of fluid dynamics is no different. Many researchers design experimental studies to gain deeper insight into the behavior of multiphase flows. Some examples of experimental techniques include shadowgraphy, x-ray imaging, or Particle image velocimetry (PIV) . Computational fluid dynamics (CFD) is the second avenue used in multiphase flow study and is the focus this research. CFD employs the computational power of modern computing to digitally investigate fluid flow problems.

Include references for these techniques or reference a review paper

Within the field of CFD, simulation models are often classified by the scale to which they resolve turbulent structures. However, these methods all share the commonality that in some form or another, they all solve the Navier-Stokes equations which govern fluid flow. While many approaches exist, there are three main schemes: Reynolds Averaged Navier-Stokes (RANS), Large Eddy Simulation (LES), and Direct Numerical Simulation (DNS). These methods scale in fidelity respectively. RANS methods solve the time averaged Navier-Stokes equations. This is the least computationally intense method of the three but requires a modeling scheme for all turbulent structures. Due to its low computational cost, RANS models are most commonly seen in industry application where a rough approximation of the turbulence scale is sufficient for design purposes. LES models are more computationally expensive than RANS models but are still fairly common in industrial engineering applications as well as academic research. LES methods solve filtered Navier-Stokes equations. This means that a filter is applied to the governing equations and then a closure model allows for the numerical simulation of the equations. The filtering operation and subsequent closure models give LES methods the ability to resolve some larger turbulent structures within a flow. While DNS methods are the most expensive of the three, they have the advantage of providing the most complete solution of the mathematical model. This is because direct simulations resolve the entire range of turbulent scales with no modeling necessary. Due to the computational expense however, DNS methods are usually reserved for academic interests in which a critical understanding of the realistic flow characteristics is required. An example of a simulated atomizing jet can be seen in Figure 1.1. The image in this figure was produced from a DNS simulation. DNS methodology is the focus of this research.

It would be useful to have a paragraph on how gas-liquid flows can be modeled with RANS, LES, or DNS. With RANS, break-up models are used (see Clarks' paper for a list of references), there some work on LES but there are challenges (see work of Herrmann and Wojciech (Wojtek) ANISZEWSKI.)

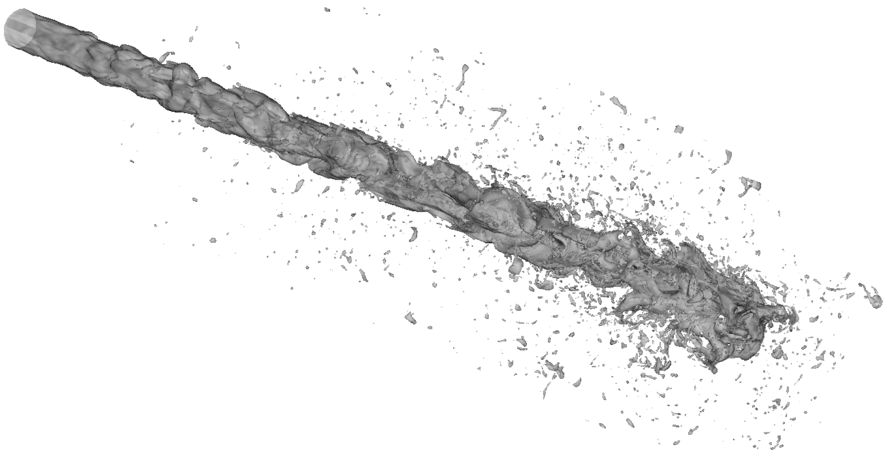


Figure 1.1: DNS Simulation of Atomizing Jet

Those involved in research using CFD can be segregated into two groupsbased on their research focus. Simulation based research utilizes in-house or commercial software packages to answer specific questions about fluid flow. An example of this might be a researcher trying to optimize an airfoil shape for specific flight conditions. With any number of software packages, various shapes and flow conditions can be simulated to determine an optimal geometry. Numerical methods for fluid dynamics focus on the accurate and efficient implementation of the Navier-Stokes equations, the governing equations of fluid mechanics, into numerical algorithms. An example

unclear what the two groups are in this paragraph of this might be the development of a new, more computationally efficient, method for solving partial differential equations. The focus of the research presented in this paper falls into this category.

It would be useful to give some examples of industrial applications as well as some recent algorithm development

.

Numerous challenges exist concerning numerical methods of multiphase flows. One challenge of particular interest is the accurate representation of the intersectionof the two fluids which exists for all gas-liquid multiphase flows. This intersection, known as the gas-liquid interface, presents notable challenge because the transition from one phase to another is represented as a mathematical discontinuity. That is, the change in density of the two fluids at the interface, is decidedly difficult to approximate mathematically. Surface tension is the governing force controlling the shape and behavior of the interface. For constant surface tension coefficient , the surface force ($\mathbf{f}_\sigma$) is expressed as $\mathbf{f}_\sigma = \sigma\kappa\mathbf{n}$ [?]. Here, σ is given as the surface tension coefficient, a value empirically determined forfluids. The vector perpendicular to the interface is $\mathbf{n}$. The curvature of the interface is given by κ. Various curvature estimation schemes exist. Review of the mechanics which are required for this estimation along with a novel curvature scheme are the focus of the research to be discussed in the remainder of this work.

I would just call this an interface and skip the intersection step.

somethings are continuous and other's are not

what changes if σ is not constant?

A paragraph (or two) focused on why the surface tension force is important, it's effect of fluid, where it comes from etc. would be interesting.

THEORY

Computational solutions for fluid flow rely on several fundamental ideas surrounding how fluids are simulated.The following section is an attempt to summarize several of these methods and lay the necessary framework for understanding the subsequent method, which is the primary focus of this work. Each of the methods discussed are present within NGA, the computational platform used in this research. Additionally, while these methods are prevalent in the CFD community, they are not the only options available. The interested reader is directed to **??** for a more comprehensive assessment.

Not sure what this says

Computational Platform

The proposed curvature estimation method exists as a module within a larger computational framework. While this scheme can be incorporated into any number of solvers, the platform used for this work was NGA [4, 6]. NGA solves low-Mach number, variable density formulations of mass and momentum conservation laws

$$\frac{\partial \rho_\phi}{\partial t} + \nabla \cdot \left(\rho_\phi \boldsymbol{u}_\phi\right) = 0 \quad \text{and} \tag{2.1}$$

$$\frac{\partial \rho_\phi \boldsymbol{u}_\phi}{\partial t} + \nabla \cdot \left(\rho_\phi \boldsymbol{u}_\phi \otimes \boldsymbol{u}_\phi\right) = -\nabla p_\phi + \nabla \cdot \left(\mu_\phi \left[\nabla \boldsymbol{u}_\phi + \nabla \boldsymbol{u}_\phi^{\mathsf{T}}\right]\right) + \rho_\phi \boldsymbol{g} \tag{2.2}$$

This section sounds “very” familiar, should probably paraphrase into your own words (read it, take notes, and then write it.

where ρ_ϕ is the density, $\boldsymbol{u}_\phi = [u, v, w]_\phi$ is the velocity field vector, t is time, p_ϕ is the hydrodynamic pressure, μ_ϕ is the dynamic viscosity, and $\boldsymbol{g}$ is the gravitational acceleration. The subscript ϕ indicates the phase and takes values of $\phi = g$ or $\phi = l$ in the gas or liquid phase, respectively.

These equations have been written in both the gas and liquid phases and are connected through jump conditions at the phase interface. For example, the jumps in density and viscosity at the interface Γ are written as

$$[\rho]_\Gamma = \rho_l - \rho_g \quad \text{and} \tag{2.3}$$

$$[\mu]_\Gamma = \mu_l - \mu_g. \tag{2.4}$$

In the absence of a phase change, the velocity field is continuous, i.e.,

$$[\boldsymbol{u}]_\Gamma = 0. \tag{2.5}$$

The pressure is discontinuous due to contributions from surface tension and the normal component of the viscous stress, i.e.,

$$[p]_\Gamma = \sigma\kappa + 2\,[\mu]_\Gamma\,\boldsymbol{n}^\mathsf{T} \cdot \nabla\boldsymbol{u} \cdot \boldsymbol{n}, \tag{2.6}$$

where σ is the surface tension coefficient and κ is the interface curvature. This is the curvature that is computed with the height function method.

These equations are discretized using a Cartesian mesh with pressure and other scalars located at cell centers and velocity components located at cell faces. Time is discretized using an iterative second order Crank-Nicolson formulation with a semi-implicit correction on each subiteration [2]. The interface is represented with a geometric volume-of-fluid (VoF) method [12,14]. The NGA code is highly parallelized, allowing for scalable simulations and fast run times and has been applied to many atomization applications [5,13,16].

Mesh Generation

In Numerical analysis, the grid or mesh, often refers to the manner in which a domain being simulated is subdivided into smaller sections and points. Traditionally these points are denoted using an i,j scheme and neighboring points will be referenced from an initial point as seen in Figure 2.1. This method is extended similarly in the z-direction for 3D applications [3].

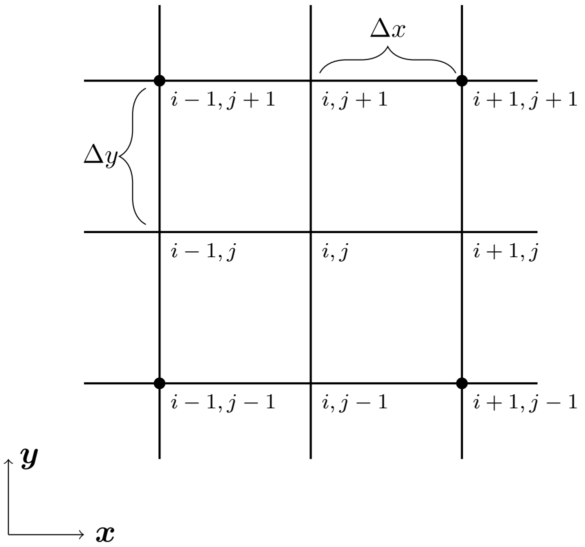


Figure 2.1: Typical notation of structured grid cells.

try foreach loop! this would be more informative if you showed the staggered grid with the location of P, u, v, $VOF = \alpha$.

The type of grid used can be classified as being in one of two categories, structured or unstructured [1]. The choice of whether to use a structured or unstructured grid should be considered on a case by case basis. However, there are some important characteristics of each that are important to recognize prior to implementation. A structured grid in 2D can be thought of a series of quadrilateral elements (bricks) placed side by side in a uniform fashion [3]. Here, neighboring elements are referenced by adding or subtracting from the base cell indices [1]. Figure **??** is an example of a structured grid. An unstructured grid does not retain uniformity and is normally (although, not always)comprised of triangular elements [18]. To reference neighboring cells in an unstructured grid, storage of cell-to-cell pointers are

really?

required [3]. Unstructured grids normally require a greater amount of memory storage and can result in slower computation times than that of a structured grid [?]. A simplified example of an unstructured grid can be referenced in Figure 2.2. The NGA code, which is this focus of this research, uses a structured grid approach for simplicity and computational efficiency. However, it should be noted that unstructured grids are increasing in popularity as the accuracy obtained from modeling complex flow geometry may be higher than that of a structured grid [8].

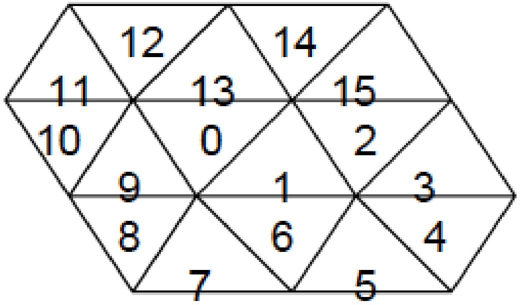


Figure 2.2: Simple Unstructured Grid [3]

Volume of Fluid Formulation

In order to accurately predict the behavior of a multiphase system, the location of the interface of the two fluids must be determined. One method for completing this is known as the Volume of Fluid (VOF) method . This method was first introduced by Hirt & Nichols (1981) and has been expanded upon by several others [?, ?, ?, ?, 8]. The defining contribution of the VOF method proposed by Hirt & Nichols is the introduction of a scalar marker function assigned to each mesh cell which is indicative of the volume of a given fluid within that cell [8]. This allows for interface to be tracked by the value of the marker cell in surrounding cells. For example, if $VOF = 1.0$ indicates a region of liquid and $VOF = 0.0$ indicates a region of gas, then a cell with

Should have a broader section on interface tracking and capturing schemes and explain why VOF is often used.

give mathematical definition of volume of fluid a value $0.0 \leq VOF \leq 1.0$ indicates a cell which contains interface. It is important to recognize that volume of a fluid may only be advected into a new cell once the current cell is full [17] . A one-dimensional example of this advection can be seen in Figure 2.3.

this is true for 1D problems but not in general

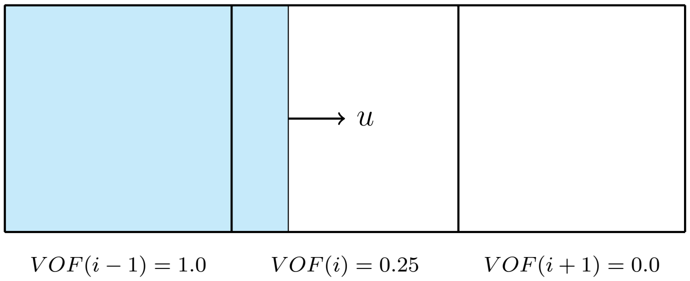
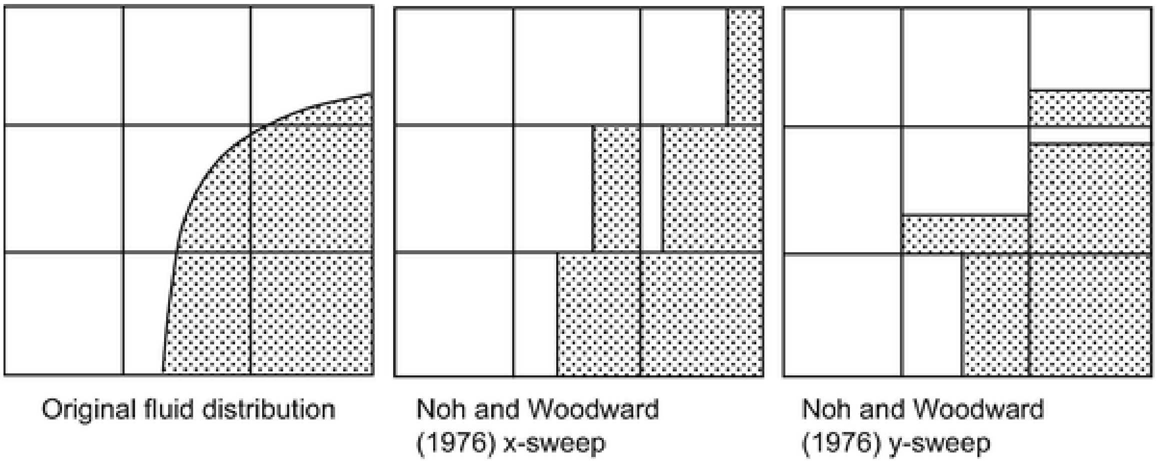


Figure 2.3: Advection of a one-dimensional fluid interface

The advection procedure for a VOF method is completed in two primary steps. First, the interface needs to be geometrically constructed. Second, the constructed interface is advected with the current velocity field [17]. Figure 2.3 depicts one-dimensional interface advection. In this case, geometric reconstruction is accomplished with a single vertical line. Adoption of this method is straightforward. As simulations increase to two or three dimensional analysis however, this problem quickly becomes non-trivial. Early attempts to resolve this difficulty include the Simple Line Interface Calculation (SLIC) method of Noh and Woodward (1976) [?]. Here, the reconstructed interface is made up of lines which align parallel to the mesh in both x and y directions.

Improvement to the SLIC method was presented by Youngs (1982) known as the Piecewise Linear Interface Calculation (PLIC) [?]. In the PLIC method, instead of aligning interfacial reconstruction lines with the mesh, a line (2D) or plane (3D) is oriented with a normal vector which is evaluated from the volume fraction gradient [?]. An example of PLIC can be seen in Figure 2.5. This method is a popular geometric

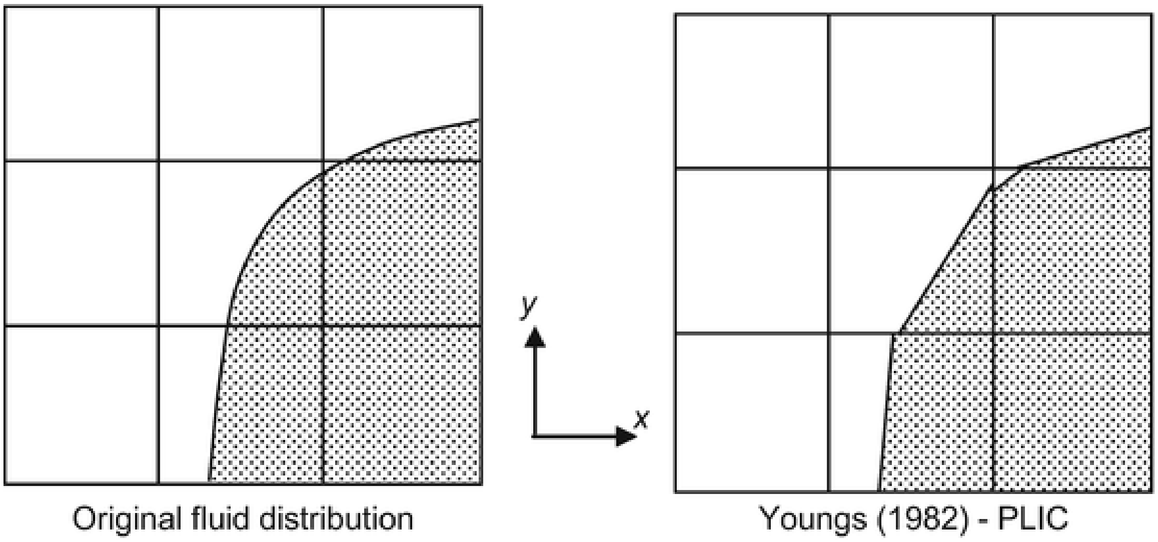

Original fluid distribution

Noh and Woodward (1976) x-sweep

Noh and Woodward (1976) y-sweep

Figure 2.4: Simple Line Interface Calculation of Noh and Woodward adapted from [?]

reconstruction scheme still today and is used in NGA for this research.



Original fluid distribution

Youngs (1982) - PLIC

Figure 2.5: Piecewise Linear Interface Calculation of Youngs adapted from [?]

should mention other options such as PROST

Rudman Dual Grid Formulation

Solution of the incompressible Navier-Stokes equations traditionally occurs on whats known as a staggered grid. On a staggered grid, pressure is typically stored at cell centers and velocity components are stored at cell faces [17]. Building from the structured grid example given in Figure 2.1, this implementation is illustrated in Figure 2.6.

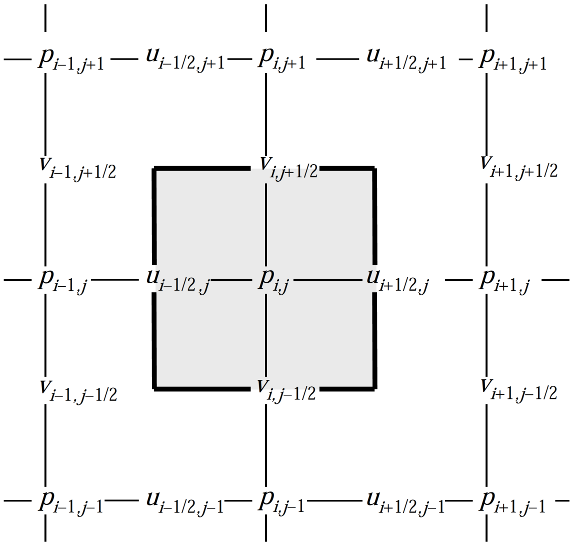


Figure 2.6: Typical staggered grid, adopted from [17].

This approach was first introduced by Harlow and Welch (1965) and is now considered the standard approach in structured mesh CFD applications today [?]. For incompressible flows, staggered grids offer the advantage of tightly coupling fluid property variables as well as eliminating pressure-velocity checkerboarding [?]. The Rudman Dual grid approach, first presented by Rudman (1998), is a technique developed for high density ratio, multiphase flows, which accurately conserves both mass and momentum. This is achieved by calculating momentum-flux values on a twice as fine mesh near the fluid interface as illustrated in Figure 2.7 [?]. The Rudman Dual mesh is incorporated into NGA and is essential to the formulation of the method presented in this research.

why no indent? if you want to put a figure call inside a paragraph, that is fine, just don't put empty lines before or after the figure.

Need more here. how does a dual grid provide conservation. do other options exist?

Within NGA there is a specific convention for moving within a cell to various subcells. This convention is illustrated in Figure 2.8

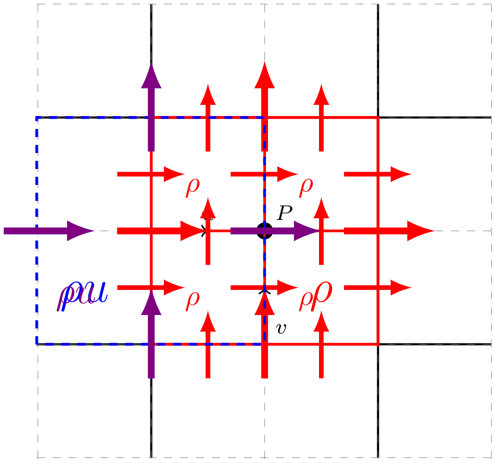
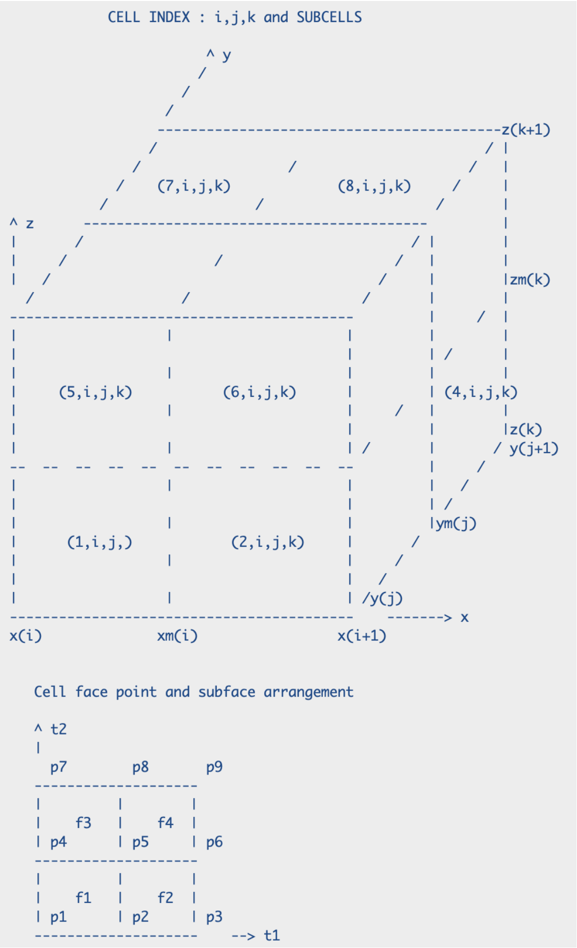


Figure 2.7: FIX THIS FIGURE



```
            CELL INDEX : i,j,k and SUBCELLS

                        ^ y
                       /
                     /
                   /
                  ------------------------------------------z(k+1)
                 /                                        / |
               /                  /                     /   |
             /    (7,i,j,k)             (8,i,j,k)     /     |
           /                  /                     /       |
^ z      ------------------------------------------         |
|       /                                        / |        |
|     /                  /                     /   |        |
|   /                                        /     |        |zm(k)
  /                  /                     /       |        |
------------------------------------------         |     /  |
|                  |                     |         |        |
|                                        |         | /      |
|                  |                     |         |        |
|     (5,i,j,k)           (6,i,j,k)      |         | (4,i,j,k)
|                  |                     |     /   |        |
|                                        |         |        |z(k)
|                  |                     | /       |       / y(j+1)
--  --  --  --  --  --  --  --  --  --  --         |     /
|                  |                     |         |   /
|                                        |         | /
|                  |                     |         |ym(j)
|      (1,i,j,)            (2,i,j,k)     |       /
|                  |                     |     /
|                                        |   /
|                  |                     | /y(j)
------------------------------------------    -------> x
x(i)              xm(i)                 x(i+1)


   Cell face point and subface arrangement

   ^ t2
   |
     p7        p8       p9
   --------------------
   |         |        |
   |    f3   |    f4  |
   | p4      | p5     | p6
   --------------------
   |         |        |
   |    f1   |    f2  |
   | p1      | p2     | p3
   --------------------    --> t1
```

Figure 2.8: finegrid representation

Curvature Estimation - Height Function Methods

Curvature can be discretely described as the reciprocal of radius of a given surface as illustrated in Figure 2.9 [?].

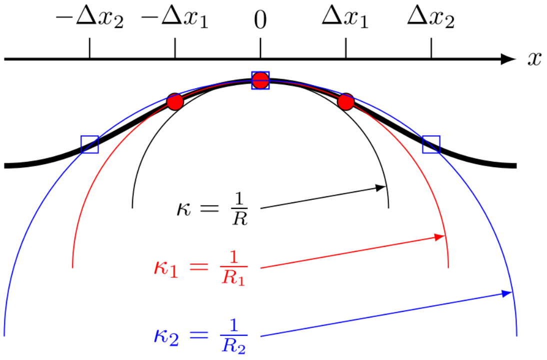


Figure 2.9: Typical staggered grid, adopted from [17].

Accurate simulations of multiphase flows require an accurate estimation of interface curvature. Figure 2.10 illustrates this importance by presenting two atomizing jet simulations. These simulations vary only by their Weber number, which is directly proportional to the surface tension term, and thereby, the curvature at the interface. It is clear that the jet seen in Figure 2.10(b) is experiencing far greater breakup than that seen in Figure 2.10(a). The correct representation of reality is highly dependent on the estimation of curvature made by the numerical model.

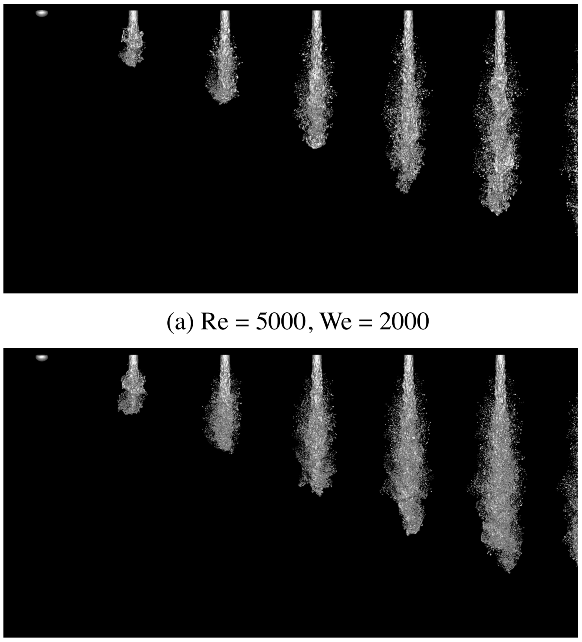
noindent

Many techniques exist for calculating values of curvature, some of these include: level set methods, height functions, coupled level set volume of fluid (CLSVOF) methods, and others [?, ?, ?, ?]. The focus of this work specifically, is on modified height function methods.

not sure about this list, these seem like mostly interface capturing methods and not ways to compute the curvature

Height functions work by integrating volume fractions (α) within columns of cells as in equation

choose a consistent way to reference equations and have a tilde



(a) Re = 5000, We = 2000

(b) Re = 5000, We = 5000

Figure 2.10: Atomization simulations with varying Weber number, adopted from [?].figure this spacing out

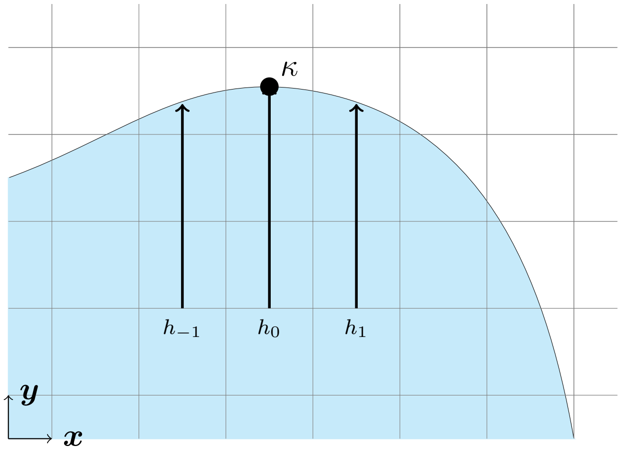


Figure 2.11: Traditional height function

2.7 to form heights. A similar approach is taken using widths where an interface is more vertical than horizontal.

add text so the equations flow with the reading.

$$h_i = \sum_{i-1}^{i+1} \alpha_{i,j} \Delta y \tag{2.7}$$

Figure 2.11 gives an example of heights at a liquid-gas interface. In the simplest execution of a height function, approximation of the curvature (κ) at the point on the grid is achieved by using a simple finite difference of the heights to approximate first and second derivatives as seen in equations 2.8 and 2.9 respectively.

Write the equations so they flow

$$H_x = \frac{h_{i-1,j} - h_{i+1,j}}{2\Delta x} \tag{2.8}$$

$$H_{xx} = \frac{h_{i+1,j} - 2h_{i,j} + h_{i-1,j}}{\Delta x^2} \tag{2.9}$$

Finally, curvature is calculated using:

$$\kappa = \frac{-H_{xx}}{(1 + H_x^2)^{\frac{3}{2}}}. \tag{2.10}$$

Clearly the above explanation is relevant for two-dimensional flows. Extension to three-dimensional flows is trivial.

What's Eq. 2.10 in 3D?

Height functions remain a prevalent method for estimating curvature because of their relative ease of implementation.However, several adjustments to the standard model have been proposed. Adjustments include changing the stencil size over which the heights are gathered [?], separating the columns from the computational mesh [?], combining heights and widths for approximation [?], and applying the approach to level sets [?](this sentence is a paraphrase from Mark's 18 JCP consider revising).

and their 2nd order convergence

Using Height Functions in Conjunction with Rudman Dual Mesh

When a dual grid is used, the standard height function method fails to capture the dynamics occurring on the fine grid. Left unmitigated, these dynamics can result in fine grid interfacial perturbations, small discontinuities in interface structures. These perturbations can grow uncontrollably and result in non-physical dynamics materializing in simulations. An example of this uncontrolled growth resulting in non-physical dynamics can be seen in Figure 2.12. The focus of this research is to develop an extension of the standard height function to include information from the Rudman dual mesh. This method results in consistent mass and momentum transport while also providing accurate interface transport that avoids non-physical dynamics.

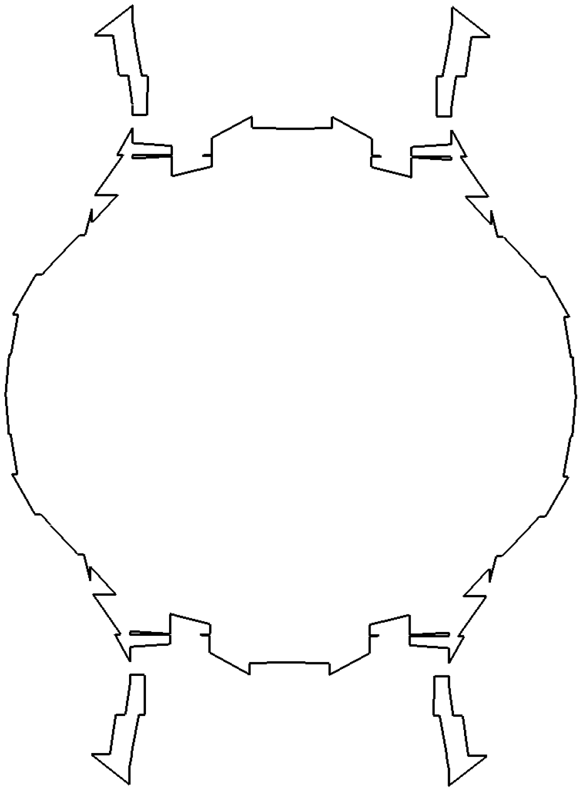


Figure 2.12: Standard method interface breakup.

Oscillating Droplet Test Case

To further quantify the problem that is occurring, a baseline test case which highlights the shortcomings of current methods is necessary. To this end, an oscillating two-dimensional droplet is used to assess the height function method and the proposed solution methods. This test case was chosen as it is considered a standard benchmark problem for testing the accurate prediction of multiphase flow behavior [**?**]. Additionally, for the height function method, the oscillating droplet offers an extensive testing of interface orientations which is important for measuring the robustness of the method. The interface is initialized with an ellipsoid. Surface tension drives the droplet's semi-major axis to fluctuate between alignment with the x and y axes. The period of oscillation T_e, is a function of surface tension coefficient (σ), density (ρ_l and ρ_g), and equivalent circular radius(R), and can be computed analytically as [15]

$$T_e = 2\pi\sqrt{\frac{(\rho_l + \rho_g)R^3}{6\sigma}}. \tag{2.11}$$

To establish successful algorithm performance of NGA, an alternate curvature scheme was selected and an oscillating droplet test case ran. Simulation parameters include a density ratio of 1000, viscosity in both the liquid and gas phases are 0, and no walls are present in the computational domain. For the initial baseline, a mesh of 64x64 grid cells make up the domain. A semi-major radius of 0.24 and a semi-minor axes radius of 0.20 define the initial displacement of the droplet. The total domain length is set to 1.0. These parameters are chosen as they align with the analytic solution assumptions made. Figure 2.13 shows kinetic energy conservation through the progression of the simulation. Simulation success is quantified by normal periods of oscillation and diminishing kinetic energy with time. Close adherence to this model can be used as a measure of success with cases from here forward and this case will be plotted with all further test cases.

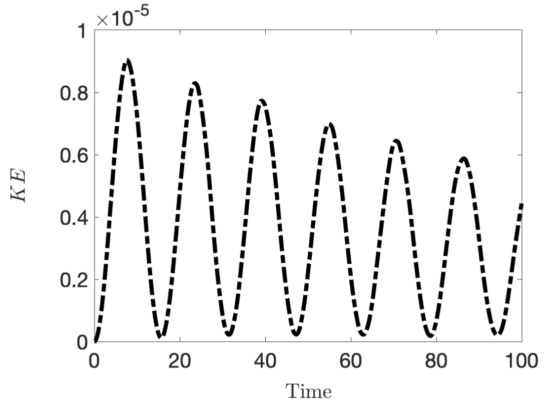


Figure 2.13: Kinetic Energy with Time - ACES Method **??**

Alternatively, Figure 2.14 shows the result of the same test case ran with a standard, coarse grid, curvature estimation scheme. The uncontrolled growth seen in the kinetic energy of the standard method is a result of the aformentioned interfacial perturbations. The curvature estimation error and resulting non-physical dynamics are the basis for this research.

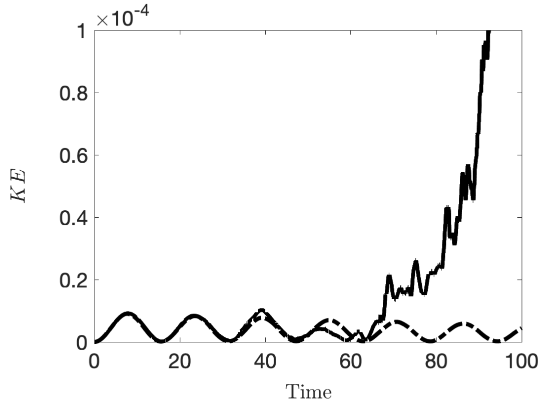


Figure 2.14: Kinetic Energy with Time - Standard Height Function Method

METHODOLOGY

The aim of this research is to allow the coarse grid to be aware of information from the fine grid. The following section will describe several methods which have been proposed and discuss the strengths and shortcomings of each.

When considering how to inform the coarse grid from the finer mesh, a natural direction is to adopt a height function method directly onto the fine grid. Implementing a height function is straightforward as previously described and, the same curvature stencil should result in a more accurate curvature estimation as there is more information available. Figure 3.1 gives an example of what this could look like. Notice that while the stencil the curvature is computed over remains the same as in Figure 2.11, there are now twice as many heights since information is pulled from columns built from fine grid cells.

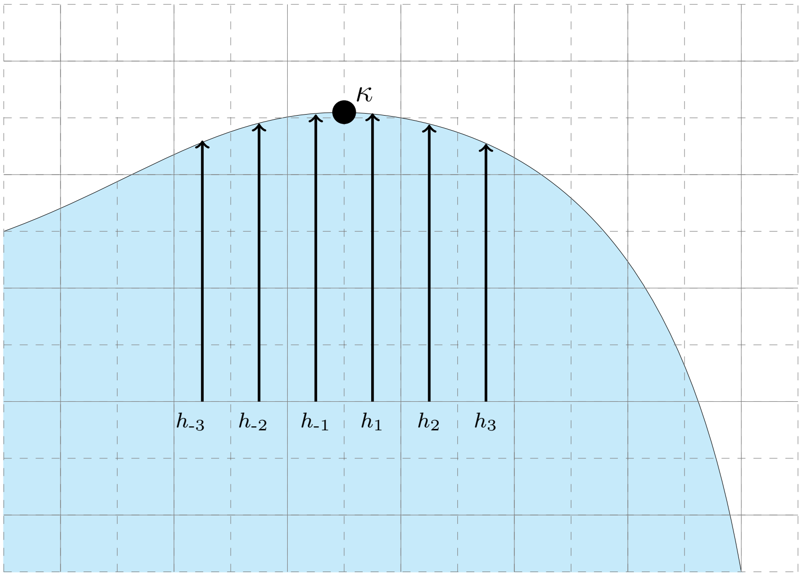


Figure 3.1: Fine grid height function

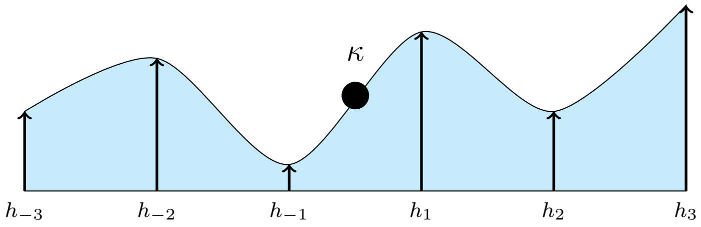


Figure 3.2: 5^{th} Order Fit

Fifth Order Height Function Method

As seen in Figure 3.1, there are now six columns over which information is being provided. With this information, it is possible to derive fifth order approximations for the first and second derivatives needed for the curvature calculation. Figure 3.2 gives an approximate representation of what a fit might look like for a given point. A general formula for deriving a finite difference approximation given a set of points is given by equations 3.1 & 3.2.

$$f'_j + \sum_{k=0}^{2} a_k f_{j+k} = O(?) \tag{3.1}$$

$$\frac{dh}{dx} = a_{-3}h_{-3} + a_{-2}h_{-2} + a_{-1}h_{-1} + a_{+1}h_{+1} + a_{+2}h_{+2} + a_{+3}h_{+3} \tag{3.2}$$

Here, a_k are the coefficients associated with the linear Taylor series which need to be solved for [9]. For the six heights given by our fine grid stencil Table 3.1 shows how the linear equations can be formed to find the first derivative.

Traditionally, from the columns in Table 3.1, six equations can be used to solve for the six unknown variables. With these coefficients, approximations can be derived for a fitting function. However, this method is not sufficient for application of the height function. While the above method provides the Taylor series approximation needed for a point-wise fitting function, column heights used in the height function

Table 3.1: Taylor Series Table

	f	f'	f''	f'''	f^{iv}	f^{v}
f'	0	1	0	0	0	0
$a_{\text{-}3}f_{\text{-}3}$	$\frac{a_{\text{-}3}}{0!}$	$a_{\text{-}3}\frac{(\text{-}3h)}{1!}$	$a_{\text{-}3}\frac{(\text{-}3h)^2}{2!}$	$a_{\text{-}3}\frac{(\text{-}3h)^3}{3!}$	$a_{\text{-}3}\frac{(\text{-}3h)^4}{4!}$	$a_{\text{-}3}\frac{(\text{-}3h)^5}{5!}$
$a_{\text{-}2}f_{\text{-}2}$	$\frac{a_{\text{-}2}}{0!}$	$a_{\text{-}2}\frac{(\text{-}2h)}{1!}$	$a_{\text{-}2}\frac{(\text{-}2h)^2}{2!}$	$a_{\text{-}2}\frac{(\text{-}2h)^3}{3!}$	$a_{\text{-}2}\frac{(\text{-}2h)^4}{4!}$	$a_{\text{-}2}\frac{(\text{-}2h)^5}{5!}$
$a_{\text{-}1}f_{\text{-}1}$	$\frac{a_{\text{-}1}}{0!}$	$a_{\text{-}1}\frac{(\text{-}h)}{1!}$	$a_{\text{-}1}\frac{(\text{-}h)^2}{2!}$	$a_{\text{-}1}\frac{(\text{-}h)^3}{3!}$	$a_{\text{-}1}\frac{(\text{-}h)^4}{4!}$	$a_{\text{-}1}\frac{(\text{-}h)^5}{5!}$
$a_{+1}f_{+1}$	$\frac{a_{+1}}{0!}$	$a_{+1}\frac{(h)}{1!}$	$a_{+1}\frac{(h)^2}{2!}$	$a_{+1}\frac{(\frac{h}{2})^3}{3!}$	$a_{+1}\frac{(\frac{h}{2})^4}{4!}$	$a_{+1}\frac{(\frac{h}{2})^5}{5!}$
$a_{+2}f_{+2}$	$\frac{a_{+2}}{0!}$	$a_{+2}\frac{(2h)}{1!}$	$a_{+2}\frac{(2h)^2}{2!}$	$a_{+2}\frac{(\frac{2h}{2})^3}{3!}$	$a_{+2}\frac{(\frac{2h}{2})^4}{4!}$	$a_{+2}\frac{(\frac{2h}{2})^5}{5!}$
$a_{+3}f_{+3}$	$\frac{a_{+3}}{0!}$	$a_{+3}\frac{(3h)}{1!}$	$a_{+3}\frac{(3h)^2}{2!}$	$a_{+3}\frac{(\frac{3h}{2})^3}{3!}$	$a_{+3}\frac{(\frac{3h}{2})^4}{4!}$	$a_{+3}\frac{(\frac{3h}{2})^5}{5!}$

method require integral quantities of the Taylor series. This is because VOF values are also integral quantities, not linear quantities. For this adjustment, the same basic principal applies. A Taylor series expansion is performed to find function values over which to integrate (h_x). Heights (H_i) are determined by integrating values of h_x over the are of the subcell as seen in equation 3.3. With heights determined, coefficients needed for derivative approximation are found using a constrained least squares approach which enforces second order accuracy. Derivatives are approximated using the heights and coefficients. Figure 3.3 provides an approximate visualization of the scheme as it differs from the representation in Figures 3.1 & **??**.

$$H_i = \int_{x_i}^{x_{i+1}} h(x)dx \tag{3.3}$$

Figure 3.4 shows the solution of the resulting fifth order finite difference first derivative used to approximate a solution of e^x plotted against an analytic solution. Figure 3.5 provide quantifying evidence that the method holds fifth order accuracy to machine precision. Figures 3.6 and 3.7 show similar behavior and similar stability

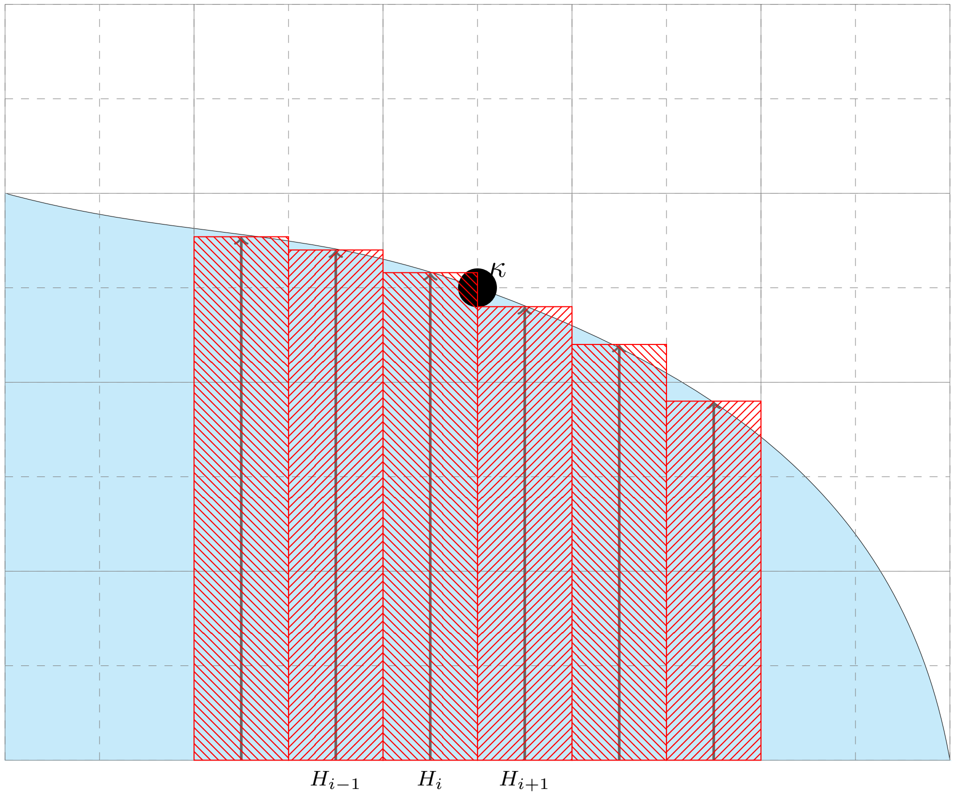


Figure 3.3: Fine grid height function

for the solution of the second derivative.

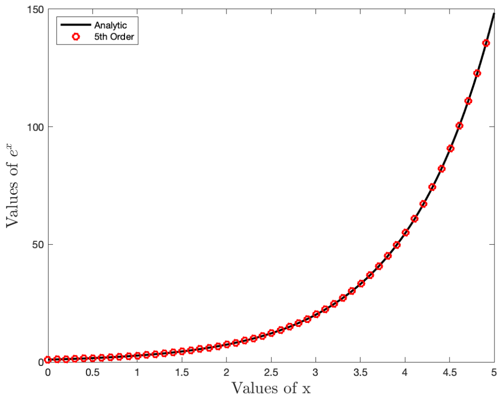


Figure 3.4: fix figure

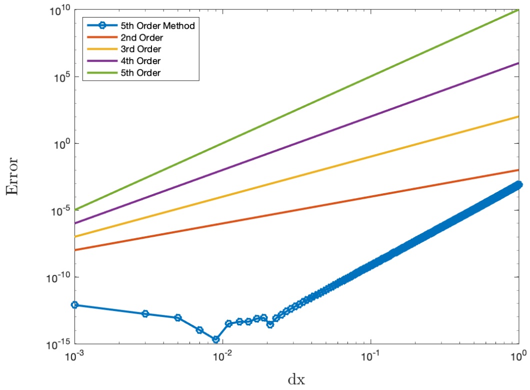


Figure 3.5: fix figure

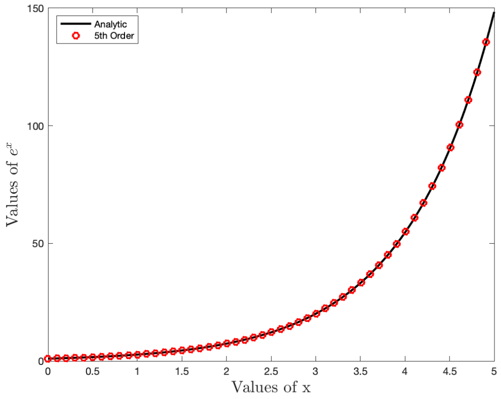


Figure 3.6: fix figure

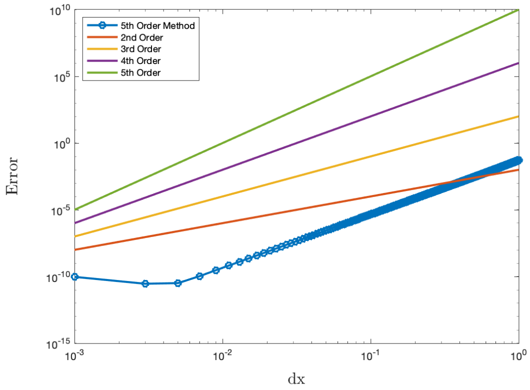


Figure 3.7: fix figure

The simplest test case for testing a curvature scheme is to compute the curvature of an exact VOF field. The curvature of a circle for example, is exactly equal to the reciprocal of the radius as seen in Figure 2.9. We establish a simple case where a circle is given a radius of 0.2. Figure 3.8 shows the resulting curvature field for this geometry. Results indicating that the method is performing as anticipated and accuracy increases with mesh refinement. With the successful implementation and initial calculation, the method was again tested using the oscillating droplet test case. Figure 3.9 shows the total kinetic energy with time throughout the course of the simulation. Here, instead of the dissipation of kinetic energy we see behavior similar to that of the standard height function method. The scheme is stable initially and then kinetic energy grows uncontrollably until eventual simulation failure.

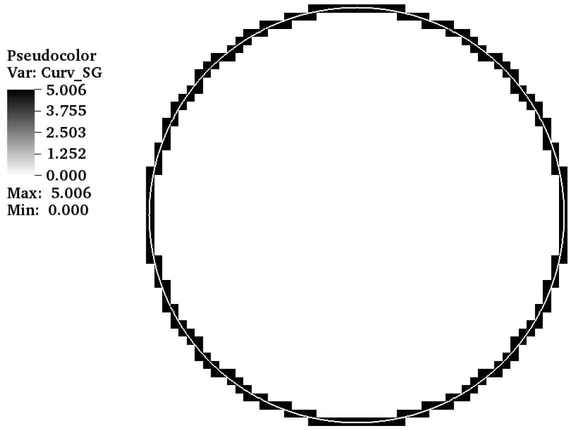


Figure 3.8: fix figure

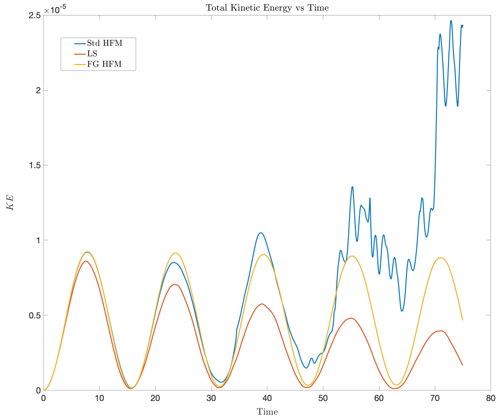


Figure 3.9: fix figure

Importance of Scale

Fifth order polynomials as used here, can have large oscillations in the curvature calculation leading to the observed errors in figure 3.9.This error is likely due to the scheme being influenced by areas of both highly positive and highly negative curvature within the fit. In a 2018 article, Owkes et al. found that the scale with which curvature is computed on heavily influences the growth of interface perturbations [11].This article also suggested that at certain scales, a second order method was more agreeable with accurate interface dynamics than a fourth order method on the same scale [11]. With this, we considered the possibility that a fifth order function on the fine grid scale may be over-fitting the points and reconstructing an interface with more perturbations than actually exist, essentially exacerbating our problem as opposed to alleviating it.

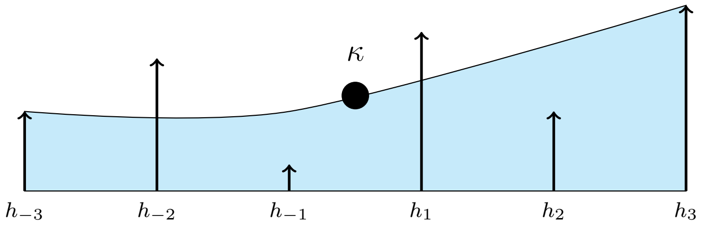


Figure 3.10: 2^{nd} Order Fit

Second Order Fine Grid Height Function Method

Because the error order of a method comes second to accurately representing physical dynamics, we decided to use the same stencil but pass a 2nd order polynomial through the points. The general idea is represented in Figure 3.10. Calculating a 2nd order method was done using the same general finite difference technique as described previously but to 2nd order accuracy. The same symmetry assumption as made previously lead to undetermined operators that could be used as a tuning parameters. The aim of this method would be to use the same fine grid information to inform the coarse model but to smear the function to reduce unwanted perturbations. Again with mesh refinement, calculation of an exact curvature field produced encouraging results as seen in Figure **??**. Parameterization of the free variables allows for an optimized solution which produces more favorable results than the fifth order method as seen in Figure **??**. However, the method still allows for the uncontrolled growth of fine grid fluctuations which results in simulation failure.

Gaussian Filtering Second Order Method

In the 5th order method, curvature is evaluated at the center of the stencil. However, a more reasonable curvature can be obtained by computing an average curvature over the stencil. This is done by filtering equation 3.2 with a Gaussian distribution as in equation 3.4. This smooths the polynomial and allows for a more

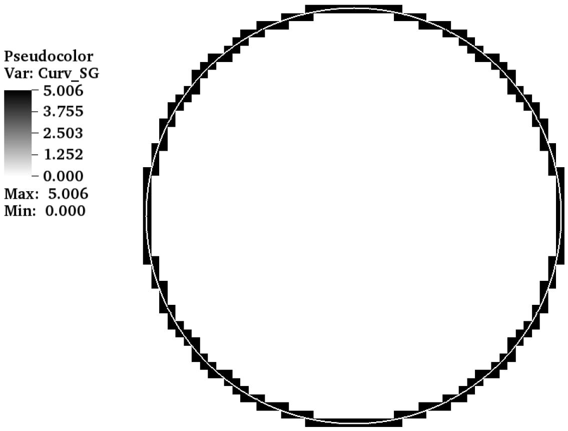


Figure 3.11: fix figure

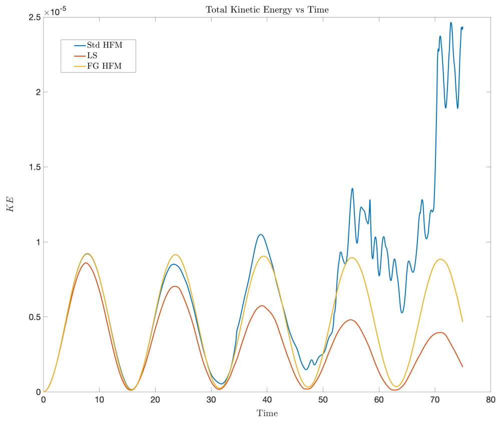


Figure 3.12: fix figure

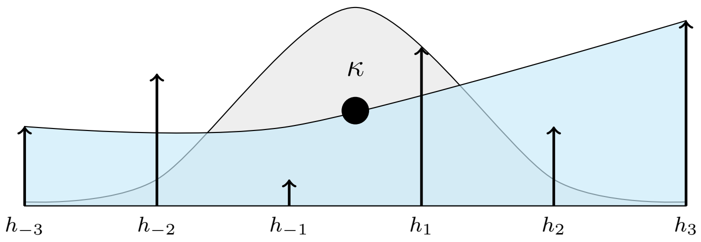


Figure 3.13: Gaussian Fit

realistic estimation of the curvature. Averaging is achieved using a convolution with a weighting kernel. Transition to ζ space simplifies this scheme and is defined as $\zeta = \frac{2}{\Delta x}(x - x_i)$. This scheme requires only one floating variable (σ) as seen in equation 3.5. Varying this parameter modifies the scale and filtering kernel of the scheme, allowing optimal operating parameters to be determined. Here, scale refers to the size of the computational stencil with which the curvature is computed as previously described by Owkes et al. [11]. As seen in figure 3.15, the result is better than previous iterations and is significantly more accurate than the standard height function method. However, this model also provides undesirable results at later time steps.

$$\kappa = \int_{-3}^{3} G(\zeta)\kappa(\zeta)d\zeta \tag{3.4}$$

$$G(\zeta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-\zeta^2}{2\sigma^2}} \tag{3.5}$$

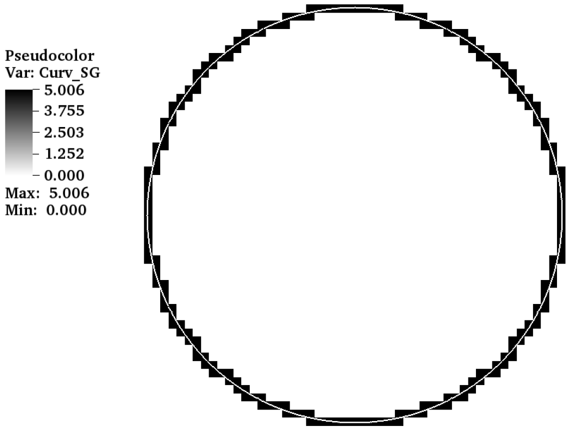


Figure 3.14: fix figure

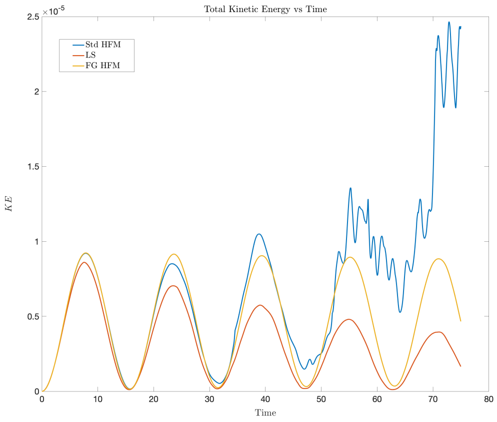


Figure 3.15: fix figure

Implementation of Fine Grid Velocities

With all three of the previous implementations we found the remaining existence of fine grid interfacial perturbations. While the fine grid height function method does provide a more rigorous representation of curvature, it does not reduce or remove these perturbations which are nonphysical and lead to the eventual failure of the simulation. To actively reduce the influence of these perturbations a fine grid velocity is implemented based on the work of Herrmann [7]. This method utilizes a spring-damper analogy to approximate interface motion and is derived originally from the Taylor analogy breakup model (TAB) of O'Rouke and Amsden [10]. The update equation proposed by Herrmann is

$$\frac{\partial \boldsymbol{u}_{\mathrm{fg}}}{\partial t} + (\bar{\boldsymbol{u}} + \boldsymbol{u}_{\mathrm{fg}}) \cdot \nabla \boldsymbol{u}_{\mathrm{fg}} = c_{\sigma} \frac{\sigma}{\rho} \bar{\kappa} (\kappa_{\mathrm{fg}} - \bar{\kappa}) - c_{\mu} \frac{\mu}{\rho L^2} \boldsymbol{u}_{\mathrm{fg}}$$

where, $\bar{\boldsymbol{u}}$ is the resolved velocity, $\boldsymbol{u}_{\mathrm{fg}}$ is the fine grid velocity, σ is the surface tension coefficient, ρ is density, $\bar{\kappa}$ is the resolved curvature, κ_{fg} is the fine grid curvature, μ is the dynamic viscosity, L is a modeling length scale and C_{σ} and C_{μ} are surface tension and viscous scaling coefficients, respectively [7]. The left side of the equation includes a velocity time rate of change as well as a convective term, while the right side includes the surface tension term which acts as a spring force and the viscous term which is analogous to a damping force. By this analogy, system behavior can be moderated by scaling the value on the surface tension or viscous coefficients. One important feature of this method is that it incorporates a difference of coarse grid curvature ($\bar{\kappa}$) and fine grid curvature (κ_{fg}). While there are several ways to approximate this difference, assume the coarse grid curvature is computed using a standard height function method and the fine grid curvature is computed using a second order height function approximation. Other options will be detailed in later sections. The generalized update algorithm can be summarized as

1. Solve for coarse grid curvature, κ
2. Solve for fine grid curvature , κ_{fg}
3. Interpolate coarse grid velocity components to subcell center
4. Compute the gradient of velocity
5. Compute normal vector
6. Compute viscous term
7. Compute density term
8. Determine accompanying fine grid velocity terms (for $\boldsymbol{x}$ direction, $\boldsymbol{v}$ & $\boldsymbol{w}$ terms are needed)
9. Update fine grid velocity value at cell face
10. Iterate through time

The update equation using equation 3.6, simplified to one-dimension can be written as

$$\boldsymbol{u_{fg,update}} = -(\bar{\boldsymbol{u}} + \boldsymbol{u}_{\mathrm{fg}}) \cdot \nabla \boldsymbol{u}_{\mathrm{fg}} - c_\sigma \frac{\sigma}{\rho} \bar{\kappa}(\kappa_{\mathrm{fg}} - \bar{\kappa}) - c_\mu \frac{\mu}{\rho L^2} \boldsymbol{u}_{\mathrm{fg}}$$

which discretizes as

$$\begin{aligned}
\boldsymbol{u}_{\text{fg,update}} = \\
&- (\boldsymbol{u}_{\text{cg}}(x) + \boldsymbol{u}_{\text{fg}}(\text{s,i,j,k})) \cdot \nabla \boldsymbol{u}_{\text{fg,x}} \\
&- (\boldsymbol{u}_{\text{cg}}(y) + \boldsymbol{v}_{\text{fg}}) \cdot \nabla \boldsymbol{u}_{\text{fg,y}} \\
&- (\boldsymbol{u}_{\text{cg}}(z) + \boldsymbol{w}_{\text{fg}}) \cdot \nabla \boldsymbol{u}_{\text{fg,z}} \\
&- C_\sigma \frac{\sigma}{\rho} \kappa (\kappa_{\text{fg}} - \kappa) \boldsymbol{n}_{\text{x}} \\
&- C_\mu \frac{\mu \boldsymbol{u}_{\text{fg}}}{\rho L^2} \boldsymbol{u}_{\text{fg}}(\text{s,i,j,k}) \qquad (3.6)
\end{aligned}$$

and the gradient is discretized using a central finite difference as

$$\nabla \boldsymbol{u}_{\text{fg,x}} \rightarrow \frac{\boldsymbol{u}_{\text{fg}}(\text{s}^+,\text{i}^+,\text{j,k}) - \boldsymbol{u}_{\text{fg}}(\text{s}^-,\text{i}^-,\text{j,k})}{\Delta x_{\text{fg}}}. \qquad (3.7)$$

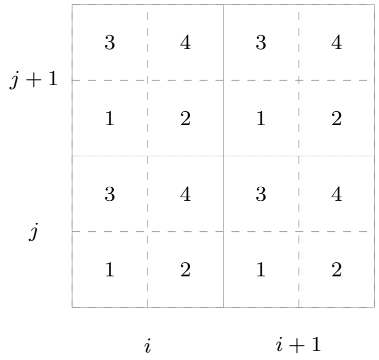


Figure 3.16: Fine grid orientation

Density and viscous terms are linear averages of the respective phase components, and $\boldsymbol{u}_{\text{cg}}$ are interpolated coarse grid velocity components from each respective direction. Note that the s index refers to the fine grid cell and the operator s^+ refers to the direction of the adjoining subcell. Illustration of subcell placement can be seen in Figure 3.16. Because movement within the fine grid can become tedious, NGA incorporates lookup tables which help to minimize confusion. These lookup tables take advantage of the repetitive sequencing of subcell identifiers to ensure accurate referencing of neighboring cells and subcells. go deeper

Correction Incorporation

Away from the phase interface, NGA uses high-order finite difference operators that conservatively transport mass, momentum, and any other scalars [6]. Near the phase interface however, the finite difference operators are inappropriate due to the discontinuous interface region. To accurately and conservatively transport near the interface, an unsplit geometric semi-Lagrangian flux method is used [12, 14]. This method relies on signed streaktubes, which contain the region of the domain that moves through a computational cell face during the timestep [14]. The streaktube is represented with a collection of tetrahedra as shown in Figures 3.17 & 3.18 and computational geometry is used to compute the flux of liquid, mass, momentum, and any scalars [14]. A correction is incorporated which forces a solenoidal condition for the cell. Adding the fine-grid velocity correction is done by modifying the additional flux due to the fine-grid velocities. For the finite difference scheme this entails adding the fine-grid velocities associated with a cell face onto the convection velocity at that face. Including the fine-grid velocity requires modifying the streaktubes and in this work two additional tetrahedra are added to each subface such that the volume of the additional tetrahedra is equal to $V_{\text{tets}} = \Delta t \mathcal{A}_{\text{face}} \boldsymbol{u}_{\text{fg}} \cdot \boldsymbol{n}$. This addition provides an explicit numerical representation of the fine grid velocity correction and maintains conservation laws.

Streaktube Mathematical Formulation Streaktube development relies on conservation laws applied to a fixed control volume. The following derivation presents

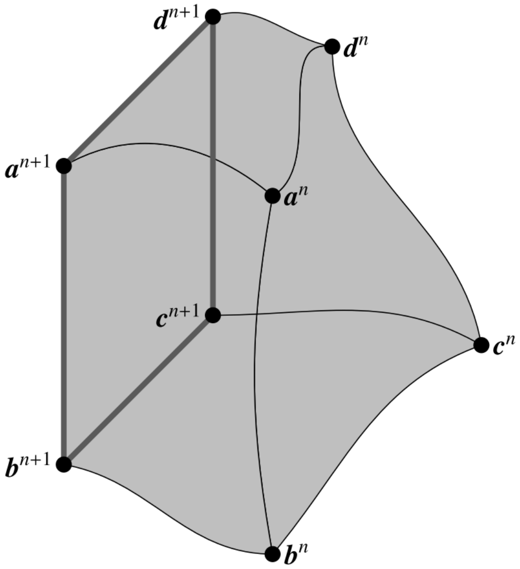


Figure 3.17: fix figure

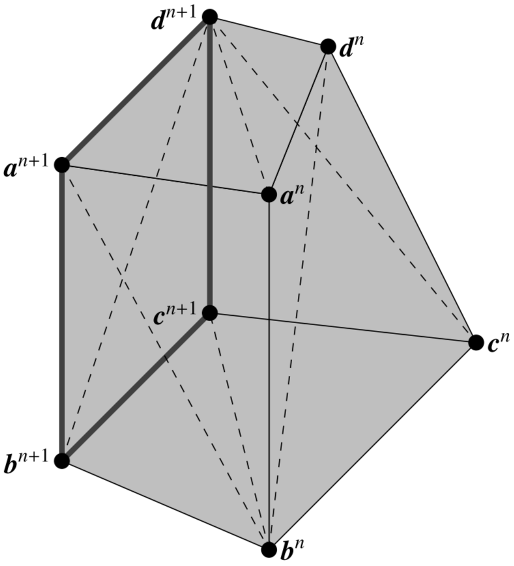


Figure 3.18: fix figure

advection of a scalar partial differential equation (PDE) into a scalar quantity as it progresses with time due to geometric flux. While the interested reader can be directed to **??** for similar derivations of equation **??**, this derivation is valuable in that it shows exact relations between equation **??** for a liquid volume fraction (α), and equation **??** for a general advected function, $f(x,t)$ **??**. This derivation shows that fluxes used within NGA to advect a function through time can be calculated by formulating streaktubes at each cell face of a control volume and evaluating that function at the current timestep. More simply, the derivation provides rigorous validation that any conserved scalar quantity that fluxes through a control volume face at a given timestep can be calculated in the previous timestep as a volumetric quantity and represented geometrically as such as displayed by Figure 3.17.

Derivation A conserved scalar $f(\boldsymbol{x}, t)$ evolves in a solenoidal velocity field as

$$\frac{\partial f}{\partial t} + \nabla \cdot (\boldsymbol{u} f) = 0 \tag{3.8}$$

where $\boldsymbol{x}$ is the spatial coordinate value, t is time, and $\boldsymbol{u}$ is the velocity field which is known **??**. Integrating this equation over a timestep and the area of a computational cell while using Gauss' theorem on the advection term, we find

$$\int_{CV} (f(\boldsymbol{x}, t^{n+1}) - f(\boldsymbol{x}, t^{n}))dV + \int_{t^n}^{t^{n+1}} \oint_{CV} f\boldsymbol{u} \cdot \boldsymbol{n}_{CV} dSdt = 0. \tag{3.9}$$

The right term is the surface flux through the control volume and is dependent on $f(\boldsymbol{x}, t^n)$, which is not normally a known quantity. However, a discrete representation of $f(\boldsymbol{x}, t^n)$ is typically known and and update equation can be used to determine $f(\boldsymbol{x}, t^{n+1})$. So, the flux is reformulated so that it is solely dependent on $f(\boldsymbol{x}, t^n)$. This is achieved by subdividing the surface of the control volume CS into sub-surfaces ∂CS_i such that

$$CS = \bigcup_{i=1}^{N_S} \partial CS_i \text{ and } \partial CS_i \cap \partial CS_j = 0 \text{ for } i \text{ and } j \in \{1, ..., N_S\} \text{ and } i \neq j \tag{3.10}$$

On each sub-surface ∂CS_i we can link the flux volume $\Omega_i(t)$ to the bounding surface $\omega_i(t)$. $\Omega_i(t)$ is just the signed volume which flows through the sub-surface ∂CS_i over a given timestep Δt. The sign of the volume is dependent on the orientation into or out of the control volume and is positive if flowing out of, and negative if flowing into, the control volume **??**.

Integrating Equation 3.8 over the flux volume Ω_i and again using Gauss' theorem on the second term we find

$$\int_{\Omega_i(t)} \frac{\partial f}{\partial t} dV + \oint_{\omega_i(t)} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS = 0. \tag{3.11}$$

Here, $\boldsymbol{n}_{\Omega_i}$ is the outward facing normal to the flux volume $\Omega_i(t)$. The bounding surface ω_i is partitioned as $\omega_{i,F} = \omega_i \cap \partial CS_i = \partial CS_i$, which is fixed and $\omega_{i,M} = \omega_i \backslash \partial CS_i$ which is a material surface **??**. This is valid by defining part of ω_i as having zero flux of f and therefore must move with the flow. The other portion of ω_i is defined such that it coincides with ∂CS_i, which is fixed in time. Intergrating Eq. 3.11 over time and using the previous partition we find that

$$\int_{t^n}^{t^{n+1}} \int_{\Omega_i(t)} \frac{\partial f}{\partial t} dV dt + \int_{t^n}^{t^{n+1}} \int_{\omega_{i,M}(t)} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS dt + \int_{t^n}^{t^{n+1}} \int_{\partial CS_i} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS dt = 0. \tag{3.12}$$

Because of its similarity to the flux term in Eq. 3.9, the right term of the above equation will provide a bridge between the two equations. The remaining terms can be made more clear using Leibniz's method which states

$$\frac{d}{dt} \int_{\Omega_i(t)} f dV = \int_{\Omega_i(t)} \frac{\partial f}{\partial t} dV + \int_{\omega_{i,M}(t)} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS. \tag{3.13}$$

With this, we can see that a $\Omega_i(t)$ is a streaktube projected backward in time from the surface ∂CS_i over the timestep Δt. Integrating Eq. 3.13 with time results in

$$\int_{\Omega_i(t^{n+1})} f(\boldsymbol{x}, t^{n+1}) dV - \int_{\Omega_i(t^n)} f(\boldsymbol{x}, t^n) dV = \int_{t^n}^{t^{n+1}} \int_{\Omega_i(t)} \frac{\partial f}{\partial t} dV dt + \int_{t^n}^{t^{n+1}} \int_{\omega_{i,M}(t)} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS dt. \tag{3.14}$$

By definition, $\Omega_i(t^{n+1})$ is zero, making the first term equal to zero. We can adopt the notation that $\Omega_i = \Omega_i(t^n)$ and call this term the flux volume. Subtracting Eq. 3.14

from Eq. 3.12 we find that

$$\int_{t^n}^{t^{n+1}} \int_{\partial CS_i} f\boldsymbol{u} \cdot \boldsymbol{n}_{\Omega_i} dS dt = \int_{\Omega_i} f(\boldsymbol{x}, t^n) dV, \tag{3.15}$$

which gives us a simple relation between the flux through the sub-surface ∂CS_i and the volume integral over Ω_i.

To obtain a useful time advancement equation we would like to combine Eq. 3.9 and Eq. **??**. However, doing so requires a relation between the normal terms, $\boldsymbol{n}_{CV}$ and $\boldsymbol{\Omega}$ as these normals are both defined using the same surface. This means that either, the two normals are identical or face opposite directions. We assume a signed notation for portions of the control surface and associate them with the flux volumes similarly. With this relationship we can finally obtain

$$\int_{CV} (f(\boldsymbol{x}, t^{n+1}) - f(\boldsymbol{x}, t^n)) dV + \sum_{i=1}^{N_S} \int_{\Omega_S} f(\boldsymbol{x}, t^n)) dV = 0 \tag{3.16}$$

where Ω_S is now the signed streaktube. The final form provides that the change in the scalar f within the control volume CV over the timestep Δt must be equal to the volume integral of the signed flux volumes.

Modification of Fine Grid Velocity

Initial results of the fine grid velocity method coupled with height function calculation of curvature suggested a fundamental problem. While explicit curvature calculation remained consistent with previous methods, the oscillating droplet test case again proved difficult as seen by Figures 3.20 & 3.19. Close inspection of Eq. 3.6 reveals that the equation is not well-poised when the coarse-grid curvature, $\bar{\kappa}$, is zero as the entire spring force goes to zero even if fine-grid interface perturbations exist. An alternative is to base the source term on the difference between coarse and fine-grid curvatures and add a delta function that restricts the source term to only be non-zero at the interface. The approximation of the Dirac delta function is calculated as the absolute difference in liquid volume fractions between cells divided by the mesh size as in equation 3.17.

$$\delta = \frac{|\alpha(\text{s,i,j,k}) - \alpha(\text{s,i-sc,j,k})|}{0.5\Delta x} \tag{3.17}$$

Additionally, a pressure term is added to ensure fine grid velocity remains divergence free, further enforcing momentum conservation. With these modifications, the proposed equation to create the fine-grid velocity can be written as

$$\frac{\partial \boldsymbol{u}_{\text{fg}}}{\partial t} + (\bar{\boldsymbol{u}} + \boldsymbol{u}_{\text{fg}}) \cdot \nabla \boldsymbol{u}_{\text{fg}} = c_\sigma \frac{\sigma}{\rho} \delta(\kappa_{\text{fg}} - \bar{\kappa}) - c_\mu \frac{\mu}{\rho L^2} \boldsymbol{u}_{\text{fg}} - \nabla P_{\text{fg}}$$

along with the continuity equation

$$\nabla \cdot \boldsymbol{u}_{\text{fg}} = 0. \tag{3.18}$$

The modified update algorithm is

1. Solve for coarse grid curvature, κ

2. Solve for fine grid curvature , κ_{fg}

3. Interpolate coarse grid velocity components to subcell center

4. Compute the gradient of velocity

5. Compute normal vector

6. Compute viscous term

7. Compute density term

8. Determine accompanying fine grid velocity terms (for $\boldsymbol{x}$ direction, $\boldsymbol{v}$ & $\boldsymbol{w}$ terms are needed)
9. Calculate delta function using equation 3.17
10. Update fine grid velocity value at cell face
11. Solve pressure Poisson equation
12. Correct velocity value using updated pressure field
13. Iterate through time

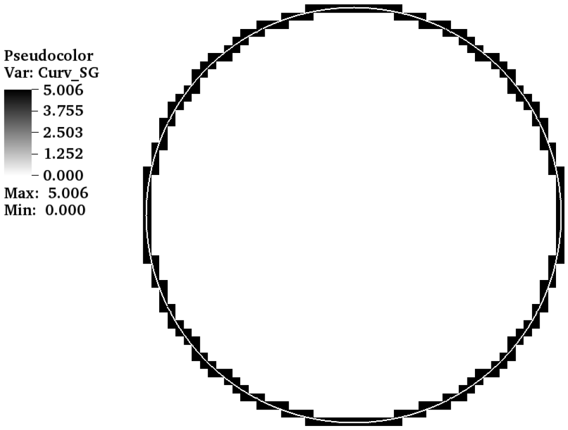


Figure 3.19: fix figure

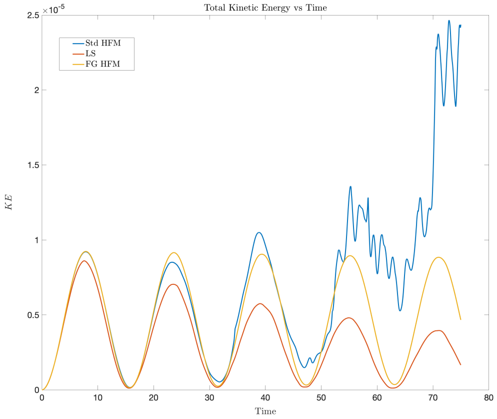


Figure 3.20: fix figure

Equation 3.18 is representative of the scheme implemented at the time of this publication. As before, the same explicit curvature calculation and oscillating droplet test cases were conducted. Results are shown in Figures 3.21 & 3.22 and show considerable improvement from the standard height function method. However, test cases still resulted in nonphysical dynamics when moving to more complex geometries. All test cases examined which include three dimensional geometry resulted in errors which eventually caused simulation failure. Additional efforts were

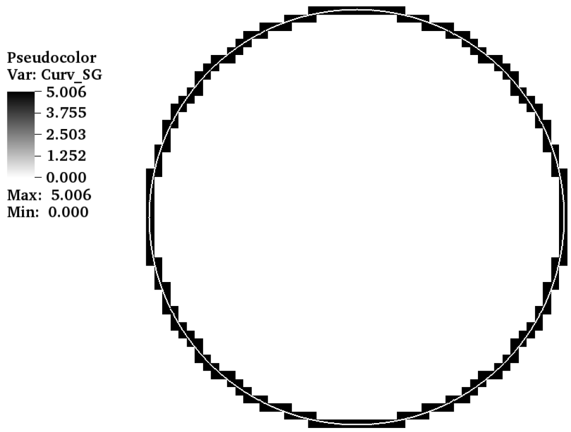


Figure 3.21: fix figure

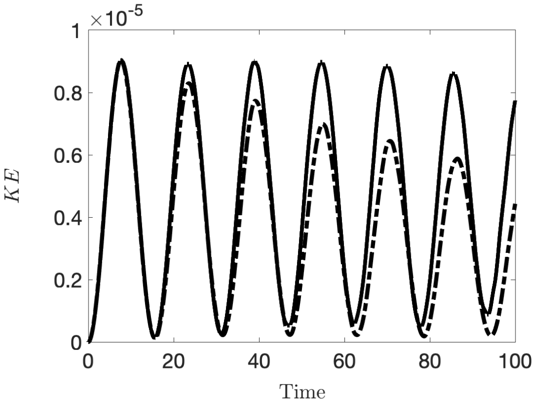


Figure 3.22: fix figure

taken to find an optimal parameterization strategy with little success. Figure 3.23 shows a parameterization study where C_σ and C_μ varied from $1e^{-}6$ to $1e^{6}$ and an oscillating droplet test case was ran. While a portion of the runs seem initially successful, tuned parameter values were not found to be consistent across various mesh sizes and no conclusive optimal configuration was found. Because of this, it seemed appropriate to take a deeper look into determining the mechanisms within the method which are the cause of these failures.

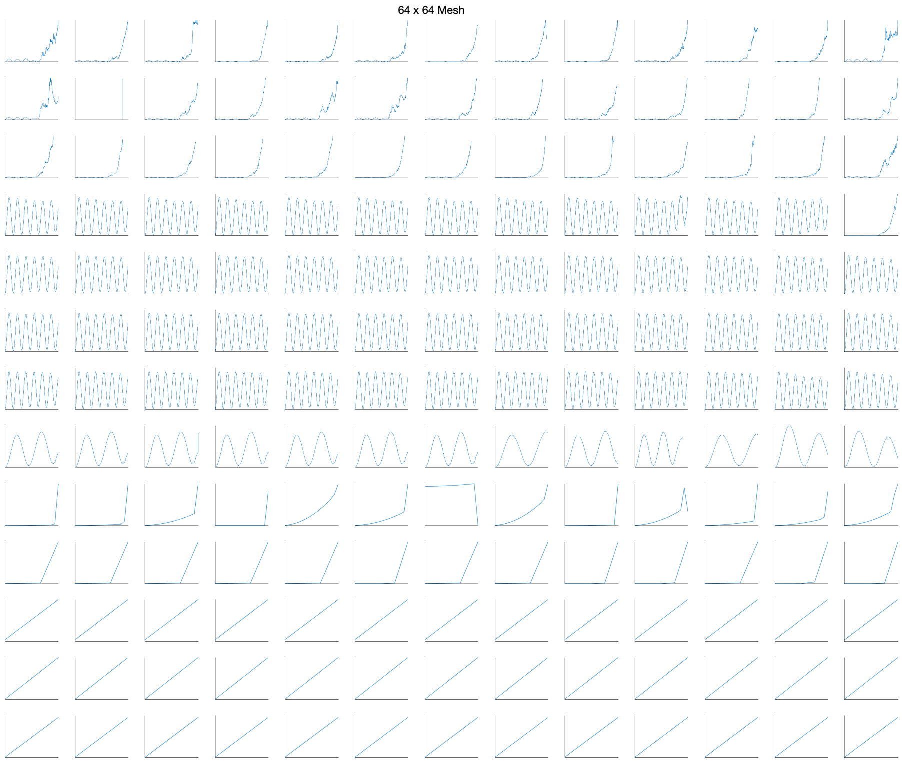


Figure 3.23: fix figure

DISCUSSION

Simplified Test Cases

With each of the schemes previously discussed, the method of evaluation consisted of an explicit curvature calculation of an exact VOF field followed by an oscillating droplet test case. Consistently, the exact curvature estimation was successful and the oscillating droplet problematic. The transition between the test cases began to seem like too great of a jump in complexity to have an intimate understanding of the progression of errors occurring. To gain a deeper understanding of the source of failure, it seemed necessary to have more simplified test cases with which to study the method. To this end, four simplified test cases were developed which we hoped would expose the limitations of the schemes.

Sine Wave Test Case A sinusoidal interface was constructed as seen in Figure 4.1. The test case was created specifically to test several key aspects of the method. First, by modifying the geometry of the sinusoid its possible to test internal and external face fluxes independently or jointly. Secondly, because the interface is highly symmetric, variation in velocity and curvature values across the domain helps to highlight deficiencies in boundary conditions as well as inter-cell transport schemes. The goal of this test case is for the fine grid scheme to force a collapse of the interface so that a final curvature of zero is achieved.

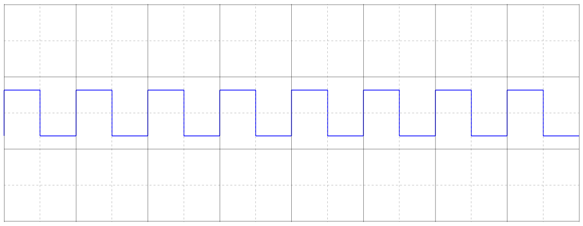


Figure 4.1: fix figure

Interfacial Protrusion Test Case The test cases shown in Figures 4.2, 4.3, & 4.4 are all of a simple linear interface at various orientations. At the center of each domain, a protrusion was placed on the interface. The aim of this test case was to recreate specific, isolated, scenarios to see if the fine grid method could accomplish collapsing the interface to a line and obtain a zero curvature across the domain. The protrusion on each interface was built to be approximately one cell width in both height and width.

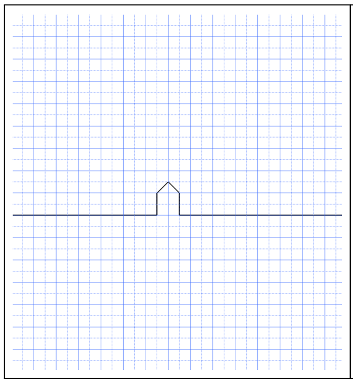


Figure 4.2: fix figure

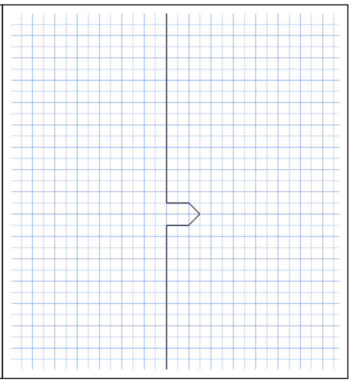


Figure 4.3: fix figure

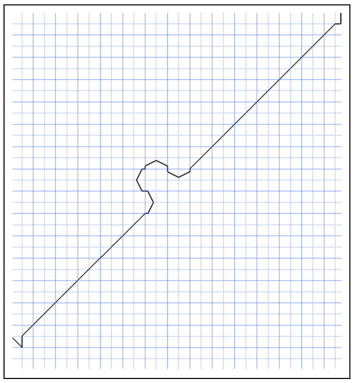


Figure 4.4: fix figure

For each of the above test cases, with the exception of the 45° test case, the method was successful in collapsing the interface to a line. This was an interesting discovery and seemed to coincide with results seen in failing oscillating droplet test cases. During the oscillating droplet test, often nonphysical geometries would begin to appear at each of the quadrant regions of the droplet as shown in Figure 4.5. This finding exposed a key limitation of the model. Well defined heights are not present in this case and cause problems for the height function method. When heights are not well defined, the approximation of derivatives is not accurate and results in nonphysical geometries.

At the current state, the fine grid method can successfully predict curvature for a select set of geometries. No successful execution of three dimensional flows have

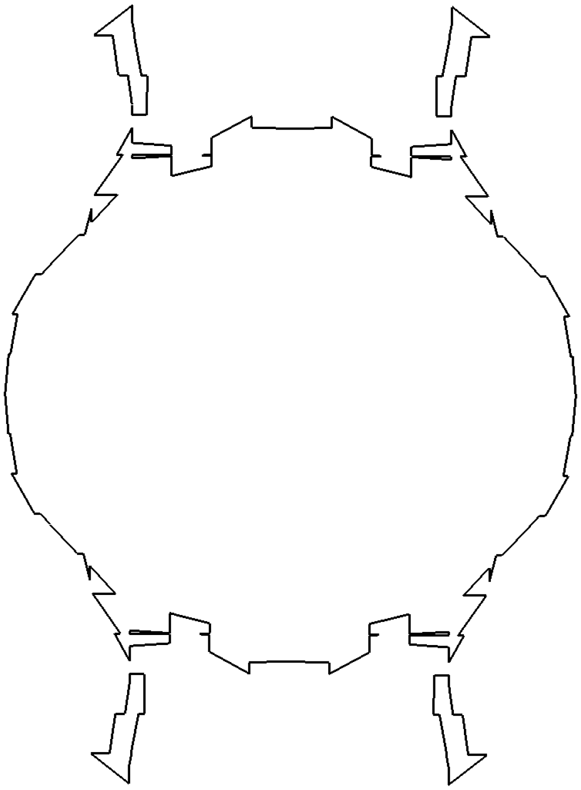


Figure 4.5: fix figure

been performed or analyzed. Rigorous investigation of the methods outlined above have left the researchers with several fundamental concerns and drawbacks of this method can be summarized as

1. It is clear that this method, in all iterations, requires well defined heights for successful execution. However, achieving well defined heights may require mesh sizes which are unrealistic for direct numerical simulation. While there may possibly be ways to work around this issue, research thus far has focused on performing successful simulations with geometries that provide ideal mesh size ratios. Investigation into more complex schemes should only be attempted after this has been achieved.

2. Incorporation of fine grid velocities seemed to have a positive impact on success of the system but required several modifications from the originally proposed method. The addition of a second pressure equation makes this method computationally expensive. So much so that simulation run time can increase by 40%. This increased expense contradicts original motivation of the investigation. Modification of the height function method was chosen as implementing the method is straightforward and the cost is low compared to more complex methods. To that point, while the fine grid height function is straightforward to implement, fine grid velocity incorporation and the subsequent incorporation of semi-lagrangian fluxes are not.

3. While little investigation has been done with three dimensional flow fields, accurate and realistic curvature becomes significantly harder to achieve as fitting is now done using planes instead of lines.

4. To present, efforts have been considering a simulation which doesn't ”blow up” to be a success. However, no efforts have gone into investigating the accuracy

of this model. This is vital to understanding the merit of the method.

Future Work

DONT DO IT

REFERENCES CITED

[1] Anderson, John David. *Computational fluid dynamics: the basics with applications*. McGraw-Hill, 2010.

[2] H. Choi and P. Moin. Effects of the computational time step on numerical solutions of turbulent flow. *Journal of Computational Physics*, 113(1):1 – 4, 1994.

[3] D. Darmofal. Structured vs. Unstructured Grids.

[4] O. Desjardins, G. Blanquart, G. Balarac, and H. Pitsch. High order conservative finite difference scheme for variable density low Mach number turbulent flows. 227:7125–7159, 2008.

[5] O. Desjardins, J. O. Mccaslin, and M. Owkes. Direct numerical and large-eddy simulation of primary atomization in complex geometries DIRECT NUMERICAL AND LARGE-EDDY SIMULATION OF PRIMARY ATOMIZATION IN COMPLEX GEOMETRIES. (December 2016), 2013.

[6] O. Desjardins, V. Moureau, and H. Pitsch. An accurate conservative level set/ghost fluid method for simulating turbulent atomization. *Journal of Computational Physics*, 227(18):8395–8416, 2008.

[7] M. Herrmann. A sub-grid surface dynamics model for sub-filter surface tension induced interface dynamics. *Computers and Fluids*, 87:92–101, 2013.

[8] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–225, 1981.

[9] P. Moin. *Fundamentals of Engineering Numerical Analysis.* Cambridge University Press, 2 edition, 2010.

[10] P. J. O'Rourke and A. A. Amsden. The tab method for numerical calculation of spray droplet breakup. In *SAE Technical Paper*. SAE International, 11 1987.

[11] M. Owkes, E. Cauble, J. Senecal, and R. A. Currie. Importance of curvature evaluation scale for predictive simulations of dynamic gasliquid interfaces. *Journal of Computational Physics*, 365:37–55, 2018.

[12] M. Owkes and O. Desjardins. A computational framework for conservative, three-dimensional, unsplit, geometric transport with application to the volume-of-fluid (VOF) method. *Journal of Computational Physics*, 270:587–612, 2014.

[13] M. Owkes and O. Desjardins. Large-eddy simulation study of injector geometry on liquid jet in cross-flow and validation with experiments. (December), 2014.

[14] M. Owkes and O. Desjardins. A mass and momentum conserving unsplit semi-Lagrangian framework for simulating multiphase flows. *Journal of Computational Physics*, 332:21–46, 2017.

[15] L. Rayleigh. On the capillary phenomena of jets. *Proc. R. Soc. London*, 29:71–97, 1879.

[16] P. Sheehy and M. Owkes. Numerical Study of Electric Reynolds Number on Electrohydrodynamic (Ehd) Assisted ... *Atomization and Sprays*, (December), 2016.

[17] G. Tryggvason, R. Scardovelli, and S. Zaleski. *Direct Numerical Simulations of Gas-Liquid Multiphase Flow.* 01 2011.

[18] Tu, Jiyuan and Yeoh, Guan Heng and Liu, Chaoqun. *Computational fluid dynamics: a practical approach.* Butterworth-Heinemann, 2013.