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

iii

LIST OF TABLES

iv

LIST OF FIGURES

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 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.

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 [26] to astrophysical approximations of star formations [1]. Researchers are studying bio-inspired dynamics of animals to gain insight into efficiency shortcomings of man made vessels [3]. Other research focuses on oceanic current modeling to gain a deeper understanding of the natural world [23]. 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 [16, 22, 25]. 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 attributed to 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.

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) [9, 18, 24]. 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.

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 fo 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.
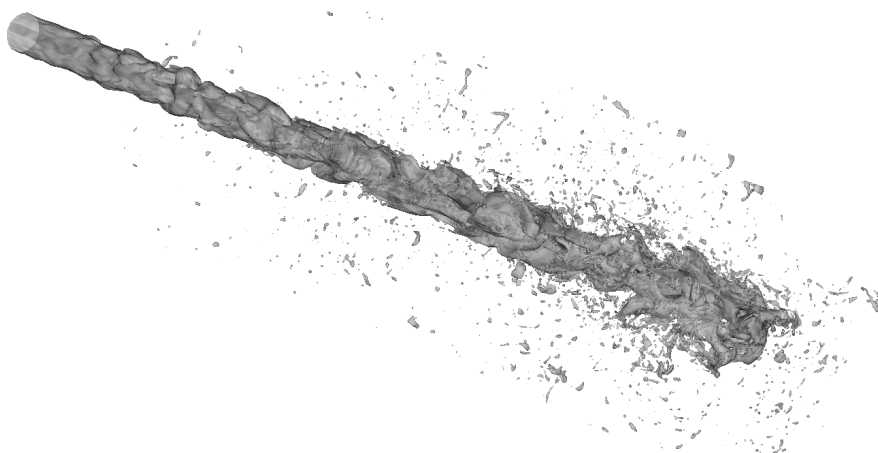
Figure 1.1: DNS Simulation of Atomizing Jet

Those involved in research using CFD can be segregated into two groups based 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 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.

Numerous challenges exist concerning numerical methods of multiphase flows. One challenge of particular interest is the accurate representation of the intersection of 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, the surface force($\mathbf{f}_\sigma$) is expressed as $\mathbf{f}_\sigma = \sigma\kappa\mathbf{n}$ [7]. Here, σ is given as the surface tension coefficient, a value empirically determined for all fluids. 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.

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 [20] for a more comprehensive assessment.

The Navier-Stokes Equations

They exist, here they are.... consider discussion

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 [6, 8]. NGA solves low-Mach number, variable density formulations of mass and momentum conservation laws

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

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

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.

6

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 [4]. The interface is represented with a geometric volume-of-fluid (VoF) method [13,15]. The NGA code is highly parallelized, allowing for scalable simulations and fast run times and has been applied to many atomization applications [7, 14, 19].

<u>Mesh Generation</u>

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 [5].
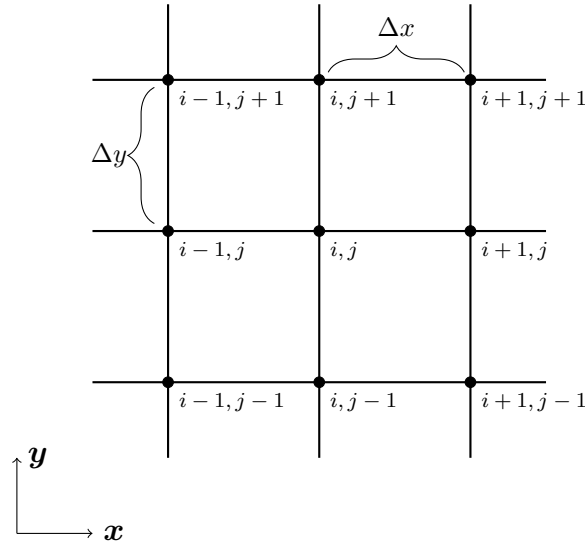

Figure 2.1: Typical notation of structured grid cells.

The type of grid used can be classified as being in one of two categories, structured or unstructured [2]. 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 [5]. Here, neighboring elements are referenced by adding or subtracting from the base cell indices [2]. 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 [21]. To reference neighboring cells in an unstructured grid, storage of cell-to-cell pointers are required [5]. 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 [11].
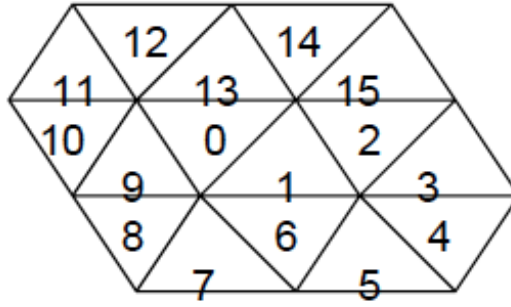

Figure 2.2: Simple Unstructured Grid [5]

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 [?,?,?,?,11]. 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 [11]. 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 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 [20]. A one-dimensional example of this advection can be seen in Figure 2.3.

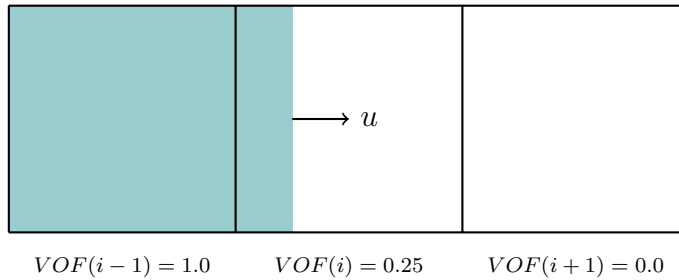$$VOF(i-1) = 1.0 \qquad VOF(i) = 0.25 \qquad VOF(i+1) = 0.0$$

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 [20]. 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
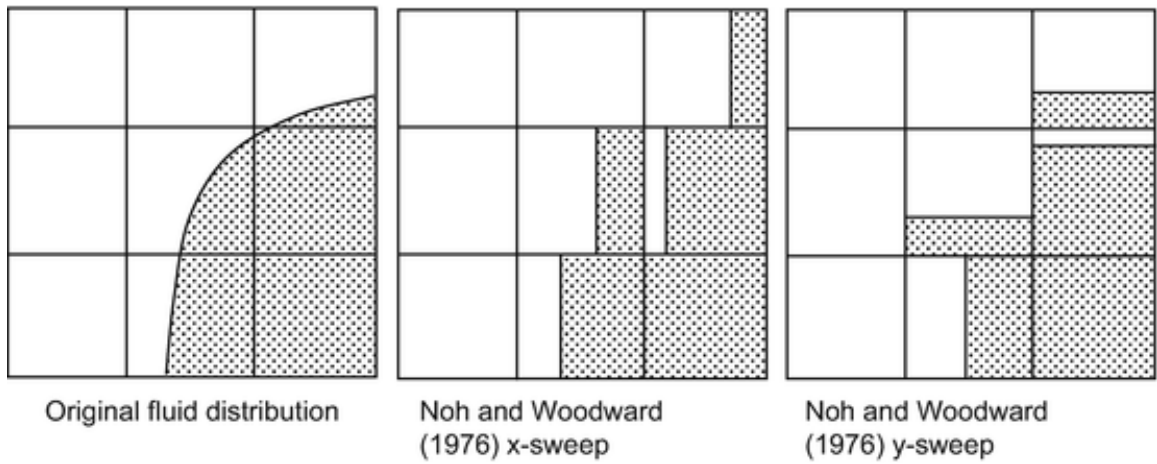
Figure 2.4: Simple Line Interface Calculation of Noh and Woodward adapted from [**?**]

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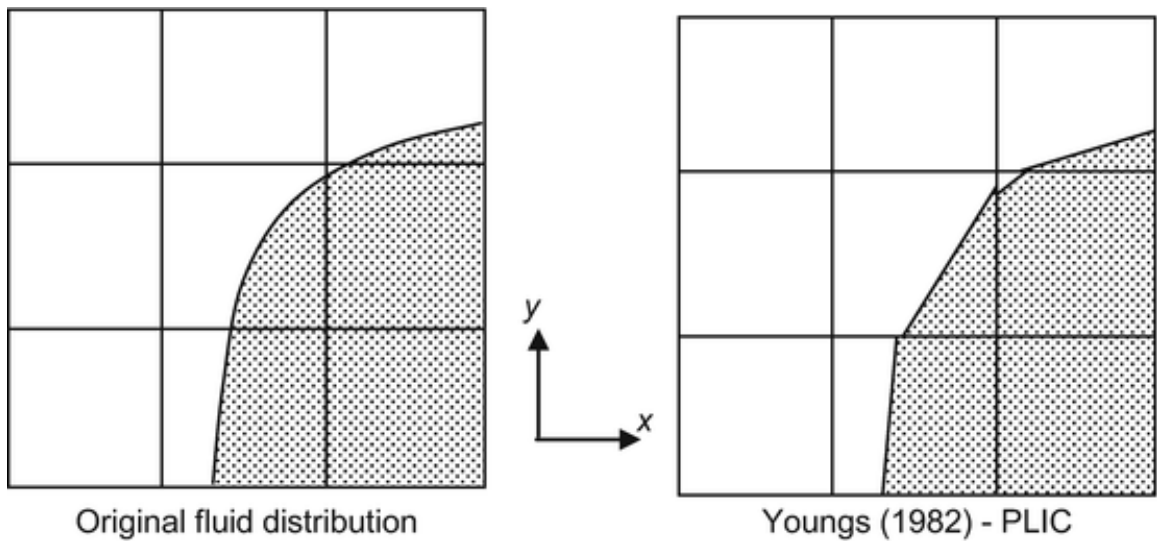

Figure 2.5: Piecewise Linear Interface Calculation of Youngs adapted from [**?**]

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 [20]. Building from the structured grid example given in Figure 2.1, this implementation is illustrated in Figure 2.6.
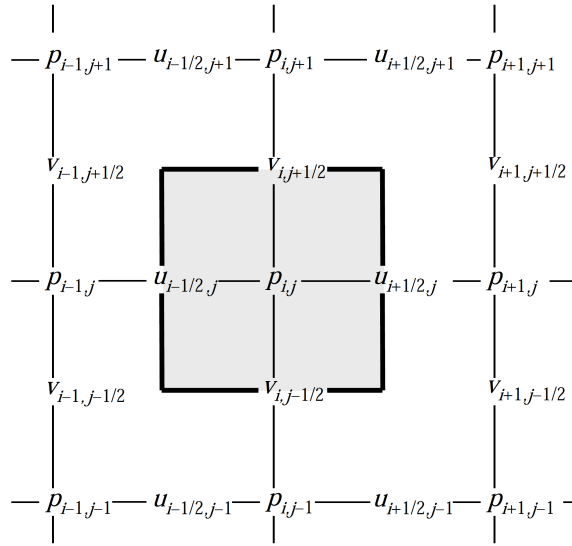

Figure 2.6: Typical staggered grid, adopted from [20].

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.
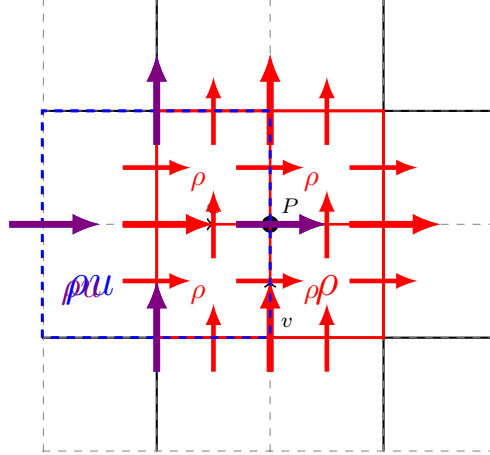
Figure 2.7: FIX THIS FIGURE

Curvature Estimation - Height Function Methods

Curvature can be discretely described as the reciprocal of radius of a given surface as illustrated in Figure 2.8 [?].

Figure 2.8: Typical staggered grid, adopted from [20].

Accurate simulations of multiphase flows require an accurate estimation of interface curvature. Figure 2.9 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 2.9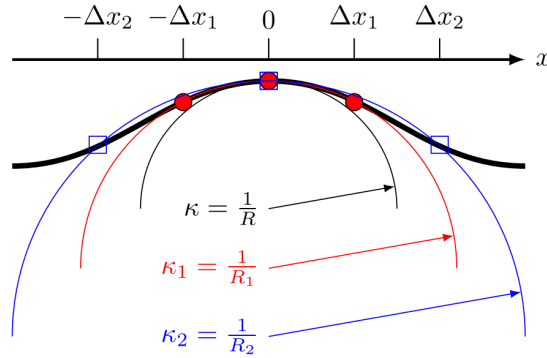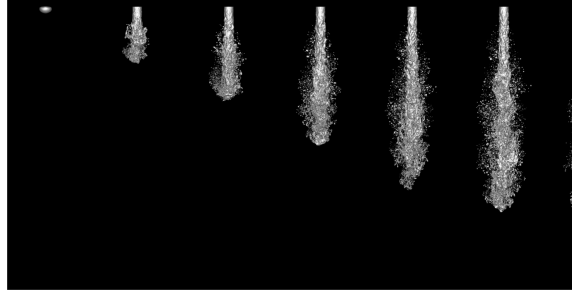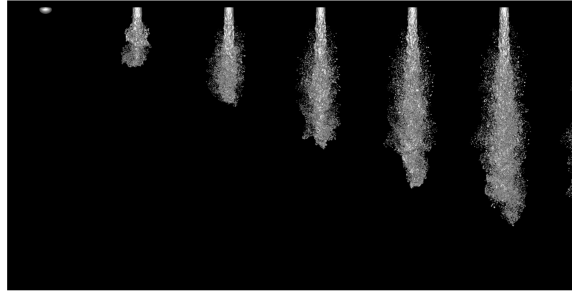(b) is experiencing far greater breakup than that seen in Figure 2.9(a). The correct representation of reality is highly dependent on the estimation of curvature made by the numerical model.

(a) Re = 5000, We = 2000

(b) Re = 5000, We = 5000

Figure 2.9: Atomization simulations with varying Weber number, adopted from [?].figure this spacing out

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.

Height functions work by integrating volume fractions (α) within columns of cells as in equation 2.7 to form heights. A similar approach is taken using widths where an
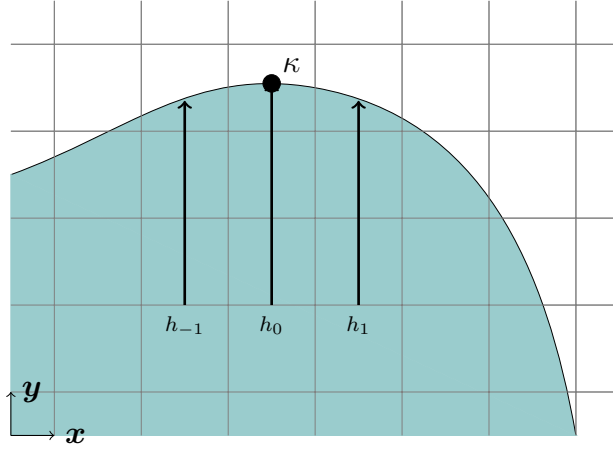
Figure 2.10: Traditional height function

interface is more vertical than horizontal.

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

Figure 2.10 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.

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

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

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.

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).

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 2.11. 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.

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
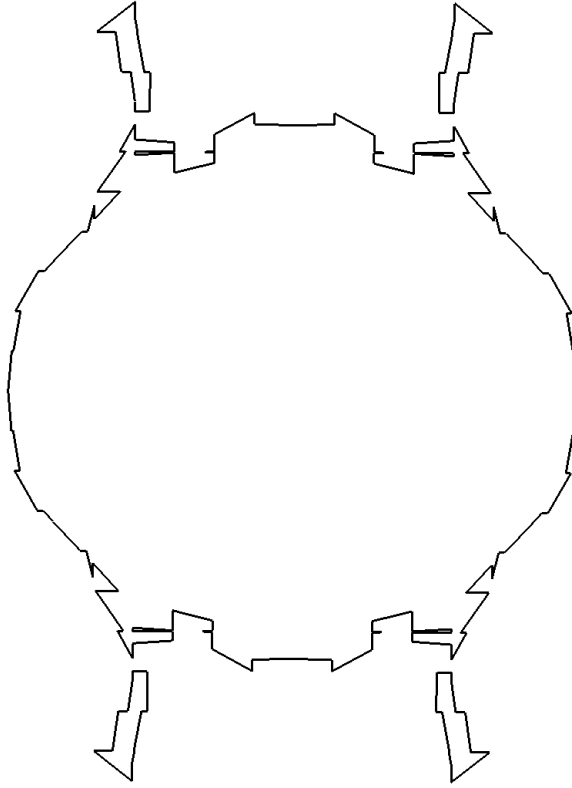
Figure 2.11: Standard method interface breakup.

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 [17]

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

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 2.12 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.
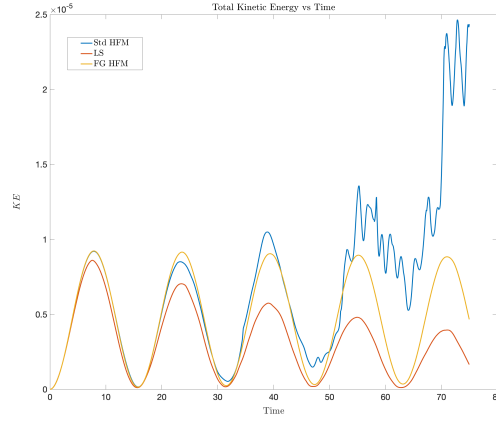
Figure 2.12: Kinetic Energy with Time THIS IS NOT THE CORRECT FIGURE

Alternatively, Figure 2.13 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 aforementioned interfacial
perturbations. The curvature estimation error and resulting non-physical dynamics
are the basis for this research.
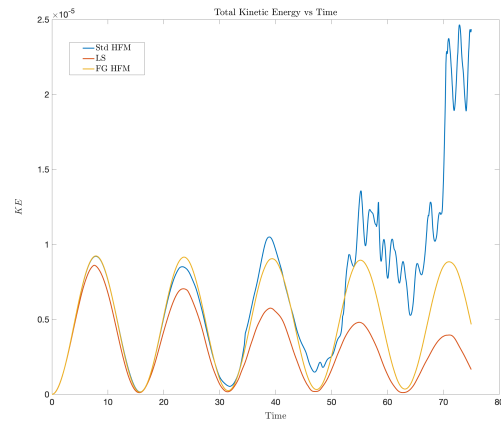
Figure 2.13: Kinetic Energy with Time THIS IS NOT THE CORRECT FIGURE

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.

Figure 3.1: Fine grid height function

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

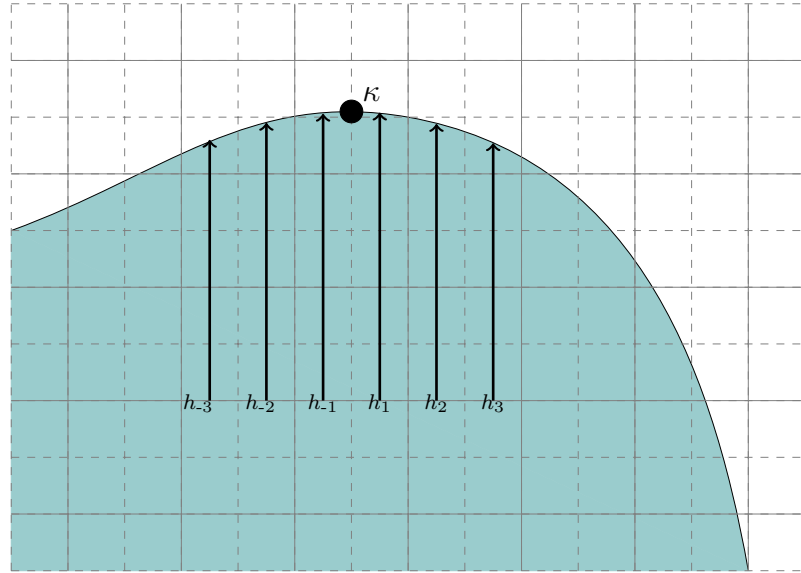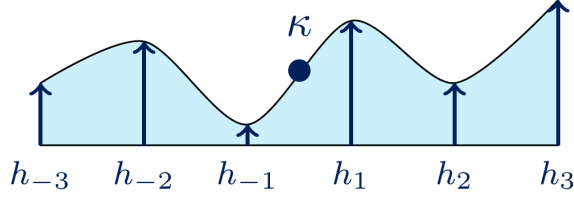Figure 3.2: 5th order scheme.

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 [?]. 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.

To force a symmetric solution, we assume

$$a_{-1} = -a_1 \tag{3.3}$$

$$a_{-2} = -a_2 \tag{3.4}$$

$$a_{-3} = -a_3. \tag{3.5}$$

Table 3.1: Taylor Series Table

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

Solving these equations, we find that the trivial solutions exists for the first, third, and fifth columns. The remaining equations are

$$a_1(2h) + a_2(4h) + a_3(6h) = -1 \tag{3.6}$$

$$a_1\frac{(h)^3}{3} + a_2\frac{(2h)^3}{3} + a_3\frac{(3h)^3}{3} = 0 \tag{3.7}$$

$$a_1\frac{(h)^5}{60} + a_2\frac{(2h)^5}{60} + a_3\frac{(3h)^5}{60} = 0. \tag{3.8}$$

We now have as many equations as unknown variables and can solve for the coefficient values. A similar process is used for the approximation of a second derivative and curvature is calculated as in equation 2.10.

Figure **??** 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 **??** provide quantifying evidence that the method holds fifth order accuracy to
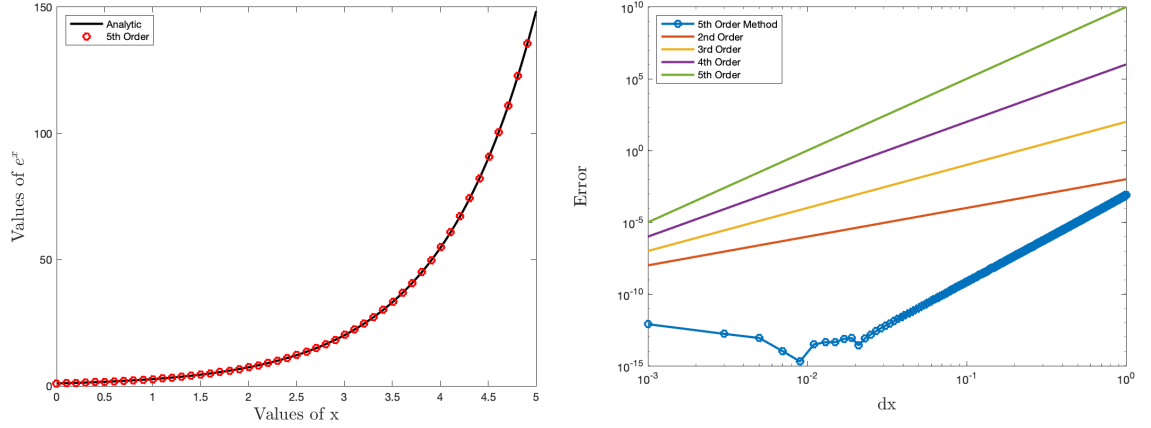
Figure 3.3: First derivative results. FIGURE OUT WHAT TO DO ABOUT THIS

machine precision. Figures **??** and **??** show similar behavior and similar stability for the solution of the second derivative.

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 2.8. We establish a simple case where a circle is given a radius of 0.2. Figure 3.5 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 3.6 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.
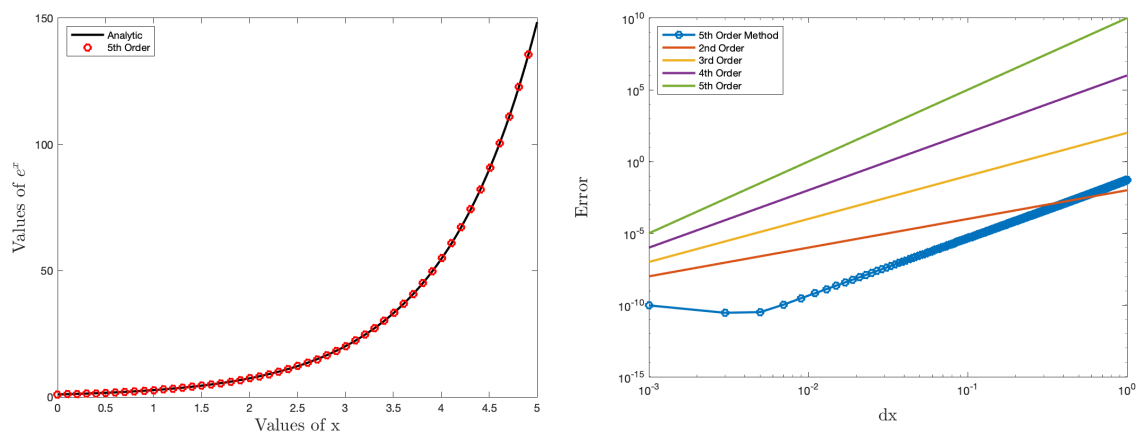
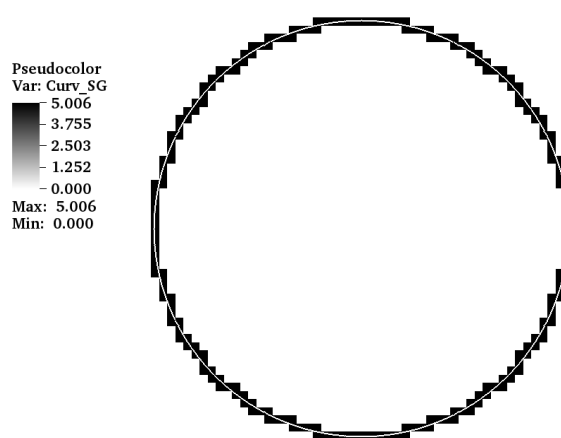Figure 3.4: Second derivative results. <mark>FIGURE OUT WHAT TO DO ABOUT THIS</mark>

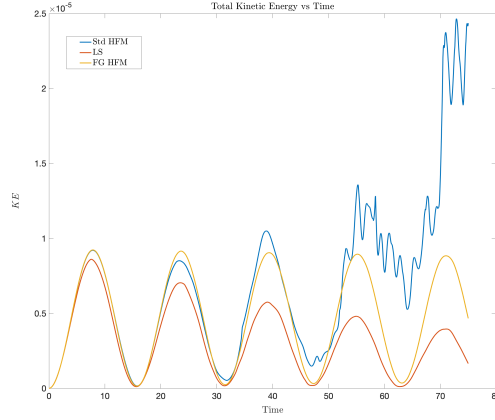Figure 3.5: 5th order scheme. <mark>THIS IS NOT THE CORRECT FIGURE</mark>

Figure 3.6: Kinetic Energy with Time THIS IS NOT THE CORRECT 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 3.6.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 [12].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 [12]. 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 3.7 shows an example of what this may look like.

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
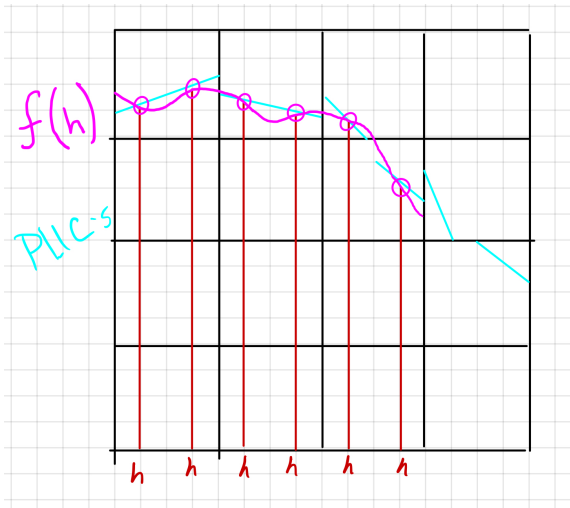
Figure 3.7: Draw something that gives the same message

through the points. The general idea is represented in Figure 3.9. 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 3.10. 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
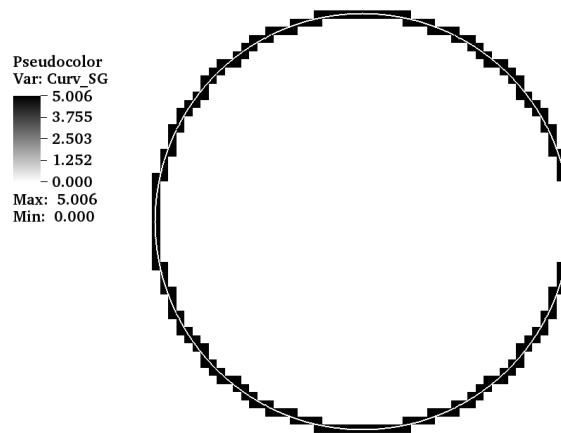
Figure 3.8: 2nd order scheme. THIS IS NOT THE CORRECT FIGURE
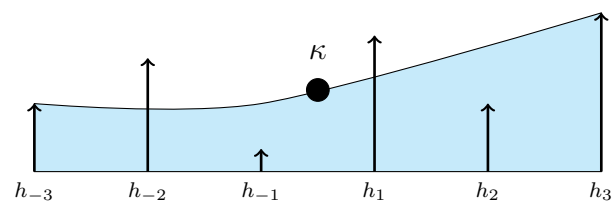
Figure 3.9: 2nd Order Fit
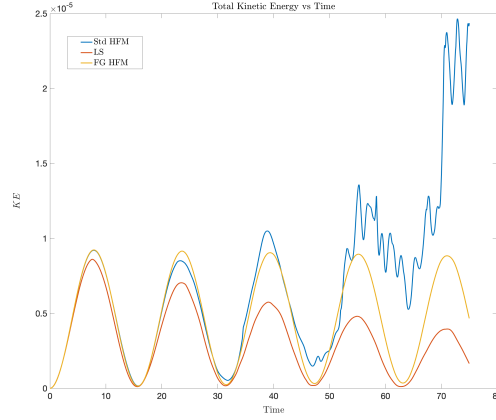
Figure 3.10: Kinetic Energy with Time THIS IS NOT THE CORRECT FIGURE

distribution as in equation 3.9. This smooths the polynomial and allows for a more 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 3.10. 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. [12]. As seen in figure ??, 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.

$$\kappa = \int_{-3}^{3} G(\zeta)\kappa(\zeta)d\zeta \tag{3.9}$$

$$G(\zeta) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{-\zeta^2}{2\sigma^2}} \tag{3.10}$$
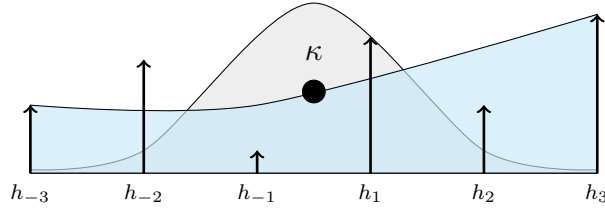
Implementation of Fine Grid Velocities

this is pulled from ILASS paper, consider rewriting this...talk more about how the implication is now that we are dealing with differences in curvature so we have tried several ways of representing coarse grid curv (avg of fg, weighted avg, std hfm) as well as adding another pressure eqn

As previously mentioned, the inclusion of a fine grid allows for the 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 the interfacial perturbations which exist on the fine mesh. To reduce the influence of these perturbations a fine grid velocity is implemented based on the work of Herrmann [10].

A spring-damper analogy is used to approximate interface motion, i.e.,

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

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 [10]. 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. Close inspection of Eq. 3.11 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. 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

$$\nabla \cdot \boldsymbol{u}_{\text{fg}} = 0. \tag{3.11}$$

Correction Incorporation

<mark>talk more about fluxes and streaktubes... have room to go into detail</mark>

Away from the phase interface, NGA uses arbitrarily high-order finite difference operators that conservatively transport mass, momentum, and any other scalars [8]. These operators are well suited for simulations of turbulent flows. Near the phase interface the finite difference operators are inappropriate due to discontinuities. Alternatively, an unsplit geometric semi-Lagrangian VoF method is leveraged [13,15].

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. The semi-Lagrangian fluxes are modified as described below.

In the semi-Lagrangian scheme, a streaktube is constructed that contains the region of the domain that moves through a computational cell face during the timestep [15]. The streaktube is represented with a collection of tetrahedra and computational geometry is used to compute the flux of liquid, mass, momentum, and any scalars [15]. 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.

RESULTS

Sine Wave test case → why is it appropriate? Show convergence.

Oscillating Droplet test case → why is it appropriate? Show convergence.

3D Atomization test case → why is it appropriate? Show convergence.

CONCLUSION

LaTeX produces documents that look great, automatically handles references and citations, and easily incorporates figures and tables. This is not a guide to LaTeX but rather an introduction to the MSU style. If you want more information about LaTeX many introductory guides can be found online.

FUTURE WORK

YOURE A LONG WAY OFF FROM WORRYING ABOUT THIS SECTION... GET TO WORK!

REFERENCES CITED

[1] M. Alexander. Simulation of binary-single star and binary-binary scattering. *Journal of Computational Physics*, 64(1):195 – 219, 1986.

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

[3] M. Bergmann and A. Iollo. Modeling and simulation of fish-like swimming. *Journal of Computational Physics*, 230(2):329 – 348, 2011.

[4] 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.

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

[6] 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.

[7] 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.

[8] 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.

[9] T. J. Heindel. A Review of X-Ray Flow Visualization With Applications to Multiphase Flows. *Journal of Fluids Engineering*, 133(7), 2011.

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

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

[16] J. Palmore and O. Desjardins. A volume of fluid framework for interface-resolved simulations of vaporizing liquid-gas flows. *Journal of Computational Physics*, 399:108954, 2019.

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

[18] G. S. Settles and M. Hargather. A review of recent developments in schlieren and shadowgraph techniques. 2017.

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

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

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

[22] B. Turnquist and M. Owkes. multiuq: An intrusive uncertainty quantification tool for gas-liquid multiphase flows. *Journal of Computational Physics*, 399:108951, 2019.

[23] J. Weare. Particle filtering with path sampling and an application to a bimodal ocean current model. *Journal of Computational Physics*, 228(12):4312 – 4331, 2009.

[24] J. Westerweel, G. E. Elsinga, and R. J. Adrian. Particle image velocimetry for complex and turbulent flows. *Annual Review of Fluid Mechanics*, 45(1):409–436, 2013.

[25] A. Xuan and L. Shen. A conservative scheme for simulation of free-surface turbulent and wave flows. *Journal of Computational Physics*, 378:18 – 43, 2019.

[26] T. YAMAGUCHI, T. Ishikawa, K.-i. Tsubota, Y. IMAI, M. Nakamura, and T. Fukui. Computational blood flow analysis new trends and methods. *Journal of Biomechanical Science and Engineering*, 1:29–50, 01 2006.

APPENDIX: EXAMPLE CODE

Include relevant codes → multiphase finegrid.f90, multiphase curv.f90, multiphase fluxes.f90 etc

```
clear all; close all; clc


%%
%------------------------%
%     dh/dx [6 Eqns]        %
%------------------------%


syms h a3 a_3 a2 a_2 a0 a1 a_1
eqn1= a_3                  +a_2                 +a_1              +a1             +a2
eqn2= a_3*(-5*h/2)        +a_2*(-3*h/2)        +a_1*(-h/2)       +a1*(h/2)       +a2*(3
eqn3= a_3*(-5*h/2)^2/2    +a_2*(-3*h/2)^2/2   +a_1*(-h/2)^2/2   +a1*(h/2)^2/2   +a2*(3
eqn4= a_3*(-5*h/2)^3/6    +a_2*(-3*h/2)^3/6   +a_1*(-h/2)^3/6   +a1*(h/2)^3/6   +a2*(3
eqn5= a_3*(-5*h/2)^4/24   +a_2*(-3*h/2)^4/24  +a_1*(-h/2)^4/24  +a1*(h/2)^4/24  +a2*(3
eqn6= a_3*(-5*h/2)^5/120 +a_2*(-3*h/2)^5/120+a_1*(-h/2)^5/120+a1*(h/2)^5/120+a2*(3


sol = solve( [eqn1, eqn2, eqn3, eqn4, eqn5, eqn6] , [a1, a2,a3,a_1,a_2,a_3]);
h   = 1;
format long
a_3x = eval(sol.a_3);
a_2x = eval(sol.a_2);
a_1x = eval(sol.a_1);
```

```
a1x  = eval(sol.a1);

a2x  = eval(sol.a2);

a3x  = eval(sol.a3);

hx5   = -[a_3x; a_2x; a_1x; a1x; a2x; a3x];



%%

%-------------------------%

%    d2h/dx2 [6 Eqns]        %

%-------------------------%



syms a3 a_3 a2 a_2 a0 a1 a_1 h

eqn1= a_3                 +a_2               +a_1             +a1           +a2

eqn2= a_3*(-5*h/2)        +a_2*(-3*h/2)      +a_1*(-h/2)      +a1*(h/2)      +a2*(3

eqn3= a_3*(+5*h/2)^2/2    +a_2*(+3*h/2)^2/2  +a_1*(+h/2)^2/2  +a1*(h/2)^2/2  +a2*(3

eqn4= a_3*(-5*h/2)^3/6    +a_2*(-3*h/2)^3/6  +a_1*(-h/2)^3/6  +a1*(h/2)^3/6  +a2*(3

eqn5= a_3*(+5*h/2)^4/24   +a_2*(+3*h/2)^4/24 +a_1*(+h/2)^4/24 +a1*(h/2)^4/24 +a2*(3

eqn6= a_3*(-5*h/2)^5/120 +a_2*(-3*h/2)^5/120+a_1*(-h/2)^5/120+a1*(h/2)^5/120+a2*(3



sol = solve([eqn1, eqn2, eqn3,eqn4, eqn5, eqn6], [a1, a2,a3,a_1,a_2,a_3]);

h   = 1;

format long

a_3xx = eval(sol.a_3);

a_2xx = eval(sol.a_2);

a_1xx = eval(sol.a_1);

a1xx  = eval(sol.a1);

a2xx  = eval(sol.a2);
```

```
a3xx  = eval(sol.a3);

hxx5  = -[a_3xx; a_2xx; a_1xx; a1xx; a2xx; a3xx];



x=linspace(0,5,500);

dx=linspace(1,1e-3, length(x));

ana = exp(x);

f   = [ exp(x-5.*dx/2);

exp(x-3.*dx/2);

exp(x-   dx/2);

exp(x+   dx/2);

exp(x+3.*dx/2);

exp(x+5.*dx/2) ];

hxxw = [-1/8;7/8;-3/4;-3/4;7/8;-1/8];


Hxx5 =  sum(hxx5.*f)./dx.^2;

Hxxw =  sum(hxxw.*f)./dx.^2;


er5xx = sqrt((Hxx5-ana).^2) ./ sqrt(ana.^2);



%%
% ===============
%     PLOTS
% ===============
```

```
h=figure(1); clf(1)

plot (x,ana,'k-','LineWidth',2)

hold on

plot (x(1:10:end),Hxx5(1:10:end),'ro','LineWidth',2)

xlabel('Values of x','interpreter','latex','fontsize',16)

ylabel('Values of $e^x$','interpreter','latex','fontsize',16)

legend('Analytic','5th Order','Location','NorthWest')

%title('Calculation of $\frac{d^2h}{dx^2}$ vs Analytic Solution','interpreter','la

basefile = 'figs';

saveas(h,fullfile(basefile,'2ndDer.png'));


Hx5 =  sum(hx5.*f)./dx;

er5 = sqrt((Hx5-ana).^2) ./ sqrt(ana.^2);


h=figure(2); clf(2)

plot (x,ana,'k-','LineWidth',2)

hold on

plot (x(1:10:end),Hx5(1:10:end),'ro','LineWidth',2)

xlabel('Values of x','interpreter','latex','fontsize',16)

ylabel('Values of $e^x$','interpreter','latex','fontsize',16)

legend('Analytic','5th Order','Location','NorthWest')

%title('Calculation of $\frac{dh}{dx}$ vs Analytic Solution','interpreter','latex'

basefile = 'figs';

saveas(h,fullfile(basefile,'1stDer.png'));


h=figure(3); clf(3)
```

```
loglog(dx,er5,'-o','LineWidth',2);

hold on ;

loglog(dx,dx.^(2)*1e-2,'LineWidth',2);

hold on ;

loglog(dx,dx.^(3)*1e2,'LineWidth',2);

hold on ;

loglog(dx,dx.^(4)*1e6,'LineWidth',2);

hold on ;

loglog(dx,dx.^(5)*1e10,'LineWidth',2);

xlabel_h=xlabel({'','dx',''},'interpreter','latex','fontsize',16);

ylabel_h=ylabel({'','Error',''},'interpreter','latex','fontsize',16);

%title('Convergence of $\frac{dh}{dx}$ Error With Mesh Refinement','interpreter','

legend('5th Order Method','2nd Order','3rd Order','4th Order','5th Order', 'Locati

set(gca, 'Color', 'none'); % Sets axes background

basefile = 'figs';

saveas(h,fullfile(basefile,'1stErr.png'));


h=figure(4); clf(4)

loglog(dx,er5xx,'-o','LineWidth',2);

hold on ;

loglog(dx,dx.^(2)*1e-2,'LineWidth',2);

hold on ;

loglog(dx,dx.^(3)*1e2,'LineWidth',2);

hold on ;

loglog(dx,dx.^(4)*1e6,'LineWidth',2);

hold on ;
```

```
loglog(dx,dx.^(5)*1e10,'LineWidth',2);

xlabel_h=xlabel({'','dx',''},'interpreter','latex','fontsize',16);

ylabel_h=ylabel({'','Error',''},'interpreter','latex','fontsize',16);

%title('Convergence of $\frac{d^2h}{dx^2}$ Error With Mesh Refinement','interprete

legend('5th Order Method','2nd Order','3rd Order','4th Order','5th Order', 'Locati

set(gca, 'Color', 'none'); % Sets axes background

basefile = 'figs';

saveas(h,fullfile(basefile,'2ndErr.png'));
```