INTRUSIVE UNCERTAINTY QUANTIFICATION METHOD FOR SIMULATIONS OF GAS-LIQUID MULTIPHASE FLOWS

by

Brian Robert Turnquist

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

of

Doctor of Philosophy

in

Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

June 2020

DEDICATION

This dissertation is dedicated to my family. Perhaps the order matters, but I'd like to first mention my dad Robert William Turnquist, and my sister Roxann Mae Turnquist, who have both passed on and I miss all the time. I think they would find it cool that I'm finally graduating for the last time.

I'd also like to mention my mother and step father, Randi and Bill Lloyd, as well as my brother Bret Turnquist. I love you guys.

Lastly, I want to dedicate this to my children, Trenton and Lily Turnquist. I hope that you're proud of me, and both find something you're passionate about and chase it down. There's no doubt in my mind that you're capable of anything you put your minds to. Use your powers for good!

ACKNOWLEDGEMENTS

I'd first like to acknowledge my advisor, Dr. Mark Owkes, for his guidance and mentorship over the last five years. I couldn't have asked for a better advisor, and it's going to be strange not to have him around all the time. I'd also like to acknowledge Dr. Erick Johnson, who has been a huge help to me along the way. Thanks also to Dr. Sarah Codd and Dr. Doug Cairns for serving on my committee and for providing feedback and support on my proposals and thesis. I'd also like to thank Dr. Dan Miller for providing me with the opportunity to teach and for support in finishing up. Thanks to Dr. Alan George and Dr. Shannon Reckinger for teaching me awesome things about fluid dynamics.

I've had several friends in the lab over the years, many who have moved on to new and exciting work and some who continue in the lab in pursuit of their own degrees. I'd like to acknowledge Kris Olshefski, Erin Hafla, Venkat Krishna, Adam Michaelson, Brandon Paschke, Stephen Kularis, and Eric Cauble. Thanks for listening to me talk about my project for years.

I also want to acknowledge Craig Freese, Josh & Mallory Powell, Erick Farnham, Martin Norunner, Jay Johnson, Skyler Strassman, and Chad Palmer. I couldn't have done this without any of them, and may be dead instead.

I'd also like to thank the National Science Foundation (grant numbers 1749779 & 1511325) for providing the grant funding to support my research.

VITA

Brian Robert Turnquist was born in Fargo, ND on May 22, 1985 to Robert and Randi Turnquist. The first born, he later became a big brother to Bret, then Roxann Turnquist in 1988 and 1989. They lived on a farm by the Wild Rice River, and spent many days in the woods building forts.

Spending the first several years of life in western Minnesota, he went to Norman County West Elementary in Hendrum, MN. After his father passed away in 1996, he and his family moved to Montana, skipping around some before settling in Judith Basin County in 1998. Brian graduated valedictorian of Stanford High School in 2003.

Brian received a Bachelor's degree in Forest Resources Management from the University of Montana in 2008, followed by a Master's degree in Resource Conservation in 2012.

TABLE OF CONTENTS

TABLE OF CONTENTS – CONTINUED

TABLE OF CONTENTS – CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES – CONTINUED

LIST OF FIGURES – CONTINUED

Figure Page

LIST OF FIGURES – CONTINUED

Figure Page

LIST OF FIGURES – CONTINUED

Figure Page

LIST OF FIGURES – CONTINUED

LIST OF FIGURES – CONTINUED

Figure Page

LIST OF FIGURES – CONTINUED

Figure Page

LIST OF FIGURES – CONTINUED

Figure Page

ABSTRACT

Simulations of fluid dynamics play an increasingly important role in the development of new technology. For example, engineers may need to simulate an atomizing jet to create a better direct injection system for improving fuel economy in a vehicle, or to more efficiently spray water for building fire mitigation systems. The increased use of computational fluid dynamics requires improvements in methodology to improve simulation efficiency and accuracy. We can extract a great deal from these models, including uncertainty information. Although simulation of gas-liquid multiphase flow scenarios are common, most are deterministic in nature. Model parameters, like fluid density or viscosity, are assumed to be known and fixed. But this is not usually the case, and a research gap exists for uncertainty analysis in these simulations. For efficient performance, an intrusive approach is used to create a multiphase solver capable of uncertainty analysis. Variables of interest, such as velocity and pressure, are converted into stochastic variables which are allowed to vary in an added uncertainty dimension. Variability is then added to fluid parameters or initial/boundary conditions and a simulation is run which produces stochastic results. To verify the solver, several cases are presented which compare the ability of the solver against analytic solutions. Once satisfied with the ability of the solver, we can answer questions about more complex scenarios. For instance, we may question how uncertainty about the surface tension force may affect the atomization of a jet and find that fluids with a lower surface tension coefficient breakup sooner (as expected). We could also consider scenarios that may not have such an obvious outcome, such as imposing uncertainty about the density ratio for an atomizing jet to determine the effect of running simulations at low vs high density ratios. multiUQ is capable of producing accurate results of real world situations. As a tool it can provide additional insight into understanding complicated multiphase flow systems.

1

INTRODUCTION

The physics of fluid behavior and motion have been of interest to scientists and engineers for millennia. Dating back to the ancient Greeks, a very famous idea is that of Archimedes' Principle (circa 250 BC), which states that the buoyancy force acting upon an object is equal to the mass of the fluid it displaces. While a basic concept that does not seek to quantify fluid motion, Archimedes' Principle helped to revolutionize the field of ship building, and is at play in the workings of lighter-than-air craft.

For several centuries after, much of the growth in fluid understanding focused in the realm of hydraulics, where it became useful and advantageous to be able to move water around for the service of cities. For example, a description of the aqueducts of Rome, including their operation, flow rates, and sizes are described in the report *De aquaeductu* by Sextus Julius Frontinus around the year 100AD.

More substantial developments in the description of fluid motion began during the 17^{th} century. Two students of Galileo, Benedetto Castelli and Evangelista Torricelli began some early work in the realm of multiphase flows. In 1628 Castelli described several dynamics on the motion of water in rivers, while in 1643 Torricelli published a treatise on the idea of the free jet, describing the exit velocity of the jet to be related to the square of the head, reduced by the friction in the orifice and the pressure of the air. Sir Isaac Newton discussed the effects of friction and viscosity on flowing water in his celebrated work *Principia* (*Mathematical Principles of Natural Philosophy*, originally published in Latin). Additionally, the development of calculus by both Newton and Gottfried Wilhelm Leibniz created the mathematics needed to more fully describe the physics of fluid motion.

During the 18$^{\text{th}}$ century fluid dynamics began to receive much more attention. Daniel Bernoulli published *Hydrodynamica seu de viribus et motibus fluidorum commentarii*, describing, among other things, what is now known as Bernoulli's Principle. In 1844 Jean le Rond d'Alembert published *Traité des fluides* outlining, in what is now known as a Lagrangian approach to fluid dynamics, principles relating to the equilibrium and motion of fluids, including the concept of incompressibility. His work was expanded upon by Leonard Euler, who in 1757 published *Principes généraux du mouvement des fluides*, which derived a general set of inviscid equations (specifically continuity and momentum equations) which describe the motion of fluids in the absence of internal friction between fluid molecules.

In the 19$^{\text{th}}$ century we eventually developed the basic framework for the field of fluid science as we know it today. Perhaps most notably is the scholarship of Claude-Louis Navier and George Gabriel Stokes, if only because they derived the equations which are considered to be humanity's best guess at describing the motion of fluids. While these men were separated by a generation and worked independently of one another, together their contributions resulted in what are today the Navier-Stokes equations. Navier, in a second paper published *Sur Les Lois du Mouvement des Fluides* in 1822, formally described friction in the equations of fluid motion for the first time [6]. Stokes published several papers in the realm of fluid dynamics, the first of which discussed incompressible flows [73], while he later discussed friction and viscosity. Another contribution was added by Hermann von Helmholtz in 1856, who published *Über Integrale der hydrodynamischen Gleichungen, welche den Wirbelbewegungen entsprechen*, discussing the theory of vortex motion and vorticity, early work in the research of turbulence.

Theoretically speaking, the equations of fluid motion have remained largely unchanged since the 19$^{\text{th}}$ century. Major developments since have come in the

form of methodologies for solving these complex partial differential equations. The invention of the computer, and eventually breakthroughs in discrete mathematics and the application of discrete math to fluid dynamics revolutionized the field of fluid dynamics, creating computational fluid dynamics (CFD). An early innovator and founder of CFD, Dudley Brian Spalding, developed several techniques for solving Navier-Stokes, including the SIMPLE algorithm, which is a very common solution method.

Uncertainty quantification in fluid dynamics

With the creation of CFD, and the expansion of methods to solve the various forms of Navier-Stokes, research moved in many directions. However, until recently most of the research to improve the computation of fluid systems have focused on deterministic solutions. This means that we assume to have perfect knowledge of the fluid parameters and the boundary or initial conditions at play. In reality, given imperfect knowledge of most scenarios, understanding how uncertainty propagates through a fluid system can be beneficial. To this aim, several studies have applied uncertainty quantification (UQ) techniques to a variety of problems, including incompressible and compressible single phase flows and flow through porous media [52, 65].

These UQ techniques may take on a variety of forms, and various approaches have been used with the goal of creating a stochastic solution set from a deterministic model. A wide array of numerical methods for a number of applications are discussed by Ghanem *et al.* [26], including rare event simulation, Bayesian approaches, and compressive sampling. Another text by Le Maître & Knio [38] outlines specifically spectral approaches to uncertainty quantification for a variety of physical applications, including fluid dynamics.

We can consider expansion of deterministic variables into an added uncertainty domain for substitution into some system of interest. For example, we may solve a stochastic system by utilizing a Green's function to solve for multiple possibilities. For a stochastic implementation, a Green's function of a differential operator acts on distributions over a subset of space. For a more explicit treatment, early work in this area was published by Adomian [1], who discussed the issue of deterministic parameters having measurement variation, and then derived a stochastic Green's formulation as a way to quantify the uncertainties due to that variation. While Green's functions are typically applied to systems with linear differentials, some work has been done to utilize a Green's formulation in non-linear equations, such as that by Frasca & Khurshudyan [22].

Though it is possible to apply a Green's function to create stochastic solutions of fluid flow problems, there are more well researched methods that have shown a great deal of promise. These existing formulations to impose UQ in simulations of fluid systems can be placed into two categories: intrusive and non-intrusive. The difference between these categories is the requirement to impose the scheme. Intrusive methods require modification of the model's governing equations, while non-intrusive methods do not.

Non-intrusive methods have a much longer history, and include the classic Monte Carlo approach [50]. With a Monte Carlo method, given a pre-built CFD solver, many simulations are run with a distribution of inputs, and statistics are computed from the many solutions. Other non-intrusive methods exist which seek to reduce the number of simulations run and thus improve computational cost. One example is the probabilistic collocation method as described by Tatang *et al.* [76], which models output uncertainty using *a priori* information of the input variables to weight the value of basis functions on which the results are projected.

Work has also been done to impose intrusive UQ methods on fluid systems for generating stochastic solutions. Perhaps most notably, Le maître *et al.* [39, 40] developed an idea for applying polynomial chaos (PC) [83] to the Navier-Stokes equations for incompressible single phase flow. Polynomial chaos allows for the creation of stochastic variables as a function of uncertainty by projection onto a series of N orthogonal polynomials. Other research has focused on UQ in multiphase flows. For example, Gel *et al.* [25] applied a non-intrusive PC method to a pre-built software package for flow through porous media. In another study an intrusive method is applied to the Euler equations for discontinuous 1D multiphase flow [62].

When deciding upon a method for solving a problem numerically, computational efficiency is important for large simulations. Various studies have compared the computational cost of intrusive and non-intrusive designs in simulations of fluid dynamics. For example, using a modification to polynomial chaos Xiu & Karniadakis [84] compare the results of their intrusive method to Monte Carlo for a channel flow with stochastic boundary layers. Through comparison of mean velocity along the centerline of the channel, they find that up to 40,000 deterministic realizations are needed for Monte Carlo to converge to the results found by their stochastic model run with an order of $N = 15$ basis functions. While this comparison is for a basic brute force method applied to a steady state problem, the performance speedup reported is ≈ 2500.

A study by El-Beltagy & Wafa [19] expanded the stream function vorticity formulation of the two-dimensional Navier-Stokes equations by way of a polynomial chaos expansion. Final results of the Monte Carlo and stochastic solver were in agreement, though no computational cost comparisons were made. Sudret [74] compared the global sensitivity analysis for a polynomial chaos approach to that of a Monte Carlo for several simple models and found the PC approach to be 2-3 orders

of magnitude faster. In another example, Jardak *et al.* [34] apply a PC approach to create a stochastic advection equation, also finding the method to be orders of magnitude faster than a Monte Carlo.

While we have not explicitly made a computational time comparison of our intrusive model to a non-intrusive scheme, we can make some estimates. For example, the case of the oscillating droplet is used to test multiphase flow solvers against an analytic result. For a coarse mesh of 64×64, a deterministic case run for 200 time steps will take 1.4 minutes on 4 processors. This provides enough information to determine if the numerical solution is comparable to the analytic solution. If uncertainty is imposed about one of the parameters, we then need to run a number of simulations for comparison of results. With the intrusive method presented in this dissertation running the same mesh for 200 times steps and a basis order of $N = 15$ takes 475 minutes, or roughly 339 deterministic runs. Depending on the distribution of the input data and the number of simulations needed to converge a Monte Carlo approach, this can result in a drastic improvement in computational time (i.e. 339 <40,000).

Methods for multiphase flows

The realm of multiphase flows consists of the interaction and motion of fluids of several phases, perhaps gas and liquid. With multiple phases present, there is need to describe the interface between phases, as well as the force implied by fluids attempting to remain fixed together (i.e. the surface tension force holding water molecules together). This can be difficult, as large jumps in pressure and density can exist at the interface. To deal with these jumps it is necessary to have some way to define the boundary between the interface.

The phase boundary can be defined through either interface capturing or tracking schemes. Among the first of such works to describe modeling of a phase interface

include a volume of fluid (VOF) approach [31] and the level set method [57], both of which capture the interface. In the volume of fluid method each cell contains a certain proportion of the reference fluid, and jumps from 0 to 1 over the course of a few cells. This proportion is key because it forces mass conservation, though numerical methods are low order and thus slow to reduce errors. In the original implementation of the level set, a scalar signed distance function is used to determine distance from the interface. While the level set method is continuous and high order formulations exist, it is not mass conservative. Another method to define the interface is a tracking scheme, as described by Tryggvason *et al.* [78], which uses Lagrangian particles to determine whether a location resides in a given fluid phase.

Since the first description of these methods, many modifications have been made. In the case of VOF methods, a reconstruction of the interface is required to determine the surface tension force. Owkes *et al.* [58] describes a weighted least squares method for recreation of the interface which improves the calculation of curvature needed to evaluate the surface tension force. In an effort to improve conservation about the level set method, Olsson & Kreiss [54] describe a transformation and transport for a conservative level set.

Additionally, a calculation of the surface tension force which is applied only at the phase interface is necessary to complete a multiphase solver. A continuum surface force method is described by Brackbill *et al.* [7], which smooths the discontinuous force over a few computational cells surrounding the interface. These contributions have created very well refined simulations for deterministic systems of multiphase flows. Unfortunately, a large knowledge gap exists for UQ of multiphase flows. Much of this gap exists due to difficulties for experimental work, as well as the computational cost of performing UQ analysis in simulations.

To address this gap, we have sought to develop a numerical model which can cost effectively produce simulations of gas-liquid multiphase flow scenarios with added uncertainty analysis. Given the goal of creating an efficient UQ solver multiphase flows, it is necessary to apply the best methodology. While many accurate and efficient deterministic multiphase solvers are already in existence, deploying them in a non-intrusive UQ format for compilation of statistics could be prohibitively expensive. For complicated scenarios such as an atomizing jet, simulations can take several days on thousands of clusters or more. As discussed previously, there is potential for major computational cost improvement through the use of intrusive methods.

Beginning with the framework described by Le Maître *et al.* [40], we already have the footprint for an incompressible stochastic single phase Navier-Stokes solver. To create a stochastic multiphase solver, we need to develop a scheme for defining a stochastic interface and calculating a stochastic surface tension force. Given that the PC method for defining stochastic variables relies on a string of continuous polynomials, we chose to extend the conservative level set method defined by Desjardins *et al.* [15] for straightforward substitution into a continuous differential. We then expand the continuum surface force method [7] into a stochastic domain to create our stochastic surface tension. Additionally to improve computational cost and avoid a system of N coupled pressure Poisson equations, we impose a density decoupled pressure correction method following the work of Dodd & Ferrante [17]. With these pieces we have created a novel stochastic multiphase flow solver - multiUQ.

Dissertation outline

This paper is a compilation of efforts focused on the development and proof-of-concept in the application of intrusive uncertainty quantification (UQ) techniques to gas-liquid multiphase flow simulations. Three chapters are composed of research articles which have either been published in a peer reviewed journal or are submitted to a journal and undergoing peer review, as outlined below.

Chapter 2 describes the first effort at proof-of-concept, and discusses the model framework in its early form. multiUQ began as a 2D model, and capturing a stochastic interface necessitated the development of a stochastic level set and required reinitialization, as well as a stochastic surface tension. This chapter has been published in the Journal of Computational Physics (JCP).

Chapter 3 discusses a major efficiency upgrade in the form of a density decoupled pressure correction method. The upgrade dramatically speeds up computational time and allowed for the expansion of the model into 3D. This chapter has been submitted for publication in JCP as well.

Chapter 4 was written for submission to a journal which focuses on publicizing the multiUQ open source code for use by the larger scientific community. Our desire is to offer the base code to other research groups that may be interested in expanding upon and developing more sophisticated methodologies for multiphase UQ with a shorter startup time.

A final chapter discusses ongoing efforts to implement a different set of basis functions for better describing the profile of the conservative level set in the uncertainty domain.

10

MULTIUQ: AN INTRUSIVE UNCERTAINTY QUANTIFICATION TOOL FOR GAS-LIQUID MULTIPHASE FLOWS

Contribution of Authors and Co-Authors

Manuscript in following chapter

Author: Brian Turnquist

Contributions: primary author, performed research, code development

Author: Mark Owkes

Contributions: secondary author, review and feedback, troubleshooting

Manuscript Information

Brian Turnquist and Mark Owkes

Journal of Computational Physics

Status of Manuscript:
____ Prepared for submission to a peer-reviewed journal
____ Officially submitted to a peer-reviewed journal
____ Accepted by a peer-reviewed journal
__X__ Published in a peer-reviewed journal

Elsevier, Inc.
Submitted November 2, 2018
Accepted September 9, 2019
Available online September 13, 2019

Abstract

Uncertainty quantification (UQ) of fluid flows offers the ability to understand the impact of variation in fluid properties, boundary conditions, and initial conditions on simulation results. In this work, an open-source program called multiUQ is developed which performs UQ using an intrusive approach applied to gas-liquid multiphase flows. Intrusive methods require modifying the governing equations by incorporating stochastic (uncertain) variables. This adds complexity but reduces computational cost compared to non-intrusive methods (e.g. Monte Carlo). To date, much of the work on intrusive UQ has focused on single phase flows. We extend this work by adding capabilities for gas-liquid flows which include a stochastic conservative level set method to capture the location of the phase interface, computing a stochastic curvature, and development of a stochastic surface tension force. Several test cases are presented which illustrate the strength of the framework. Both deterministic and stochastic channel flow cases converge to analytic results and demonstrate the accuracy of the level set transport. Zalesak's disk and the deformation test cases further highlight the abilities of the transport method as well as the robustness of the reinitialization equation, which maintains the level set profile. Deterministic and stochastic oscillating droplet test cases paired with analytic results, solve a true multiphase flow problem, and highlight the abilities of the UQ framework. Finally, results from a stochastic atomizing jet show droplet breakup and merging for cases of uncertainty about the surface tension coefficient and incoming velocity.

Introduction

Studies of gas-liquid flows typically include inexact knowledge of boundary and initial conditions, as well as fluid properties. However, simulations typically assume

these conditions are known, producing a single deterministic result that ignores variability. Uncertainty quantification (UQ) propagates inexact knowledge through the flow and measures how the uncertainty manifests in measurable outputs [38]. For example, in simulations of fuel atomization, the surface tension coefficient plays a critical role in determining when and how the coherent fuel structure that is injected breaks into ligaments and droplets [41]. Therefore, uncertainty about surface tension leads to uncertainty about spray angle, drop size, and velocity distributions, among other measurable quantities. Consequently, UQ is a useful tool for improving fuel injector designs, and has innumerable applications outside the atomization world.

UQ can be divided into two categories: non-intrusive and intrusive. A classic and well-known non-intrusive method is the Monte-Carlo sampling strategy, where multiple model runs are completed with varying input parameters and statistics are computed from the many results [50]. Non-intrusive methods do not require any changes to the solver used in the study, and can be viewed as a wrapper around the solver, making non-intrusive UQ rather simple to implement. However, it can be difficult to know that convergence has occurred, can require a great number of simulations to converge, and as such becomes cost prohibitive [13]. Additionally samples are generally correlated, slowing convergence further and complicating the calculation of variances [13].

Other methods of non-intrusive UQ exist that seek to improve convergence to a population distribution and reduce computational expense. One example is the probabilistic collocation method, suggested by Tatang et al. [76], which models output uncertainty using a priori information of the input variables to weight the value of basis functions on the polynomial chaos representation of outputs. This method is frequently employed due to the generally reduced number of simulation runs [43, 44, 48].

The alternative category is intrusive methods, which involve a transformation of the governing equations being solved to incorporate the stochastic (uncertain) variables. Including stochastic variables requires significant modification to algorithms and codes. There are several methods for implementing an intrusive UQ scheme, which largely focus on a spectral expansion of stochastic variables. Le Maître and Knio [38] discussed a number of these methods with respect to application in computational fluid dynamics (CFD) and particularly a polynomial chaos representation of stochastic variables.

Several studies have been conducted that use polynomial chaos (PC) to represent stochastic variables in CFD applications. El-Beltagy and Wafa [19] expanded the stream function vorticity formulation of the two-dimensional Navier-Stokes equations by way of a polynomial chaos expansion. The simplicity of this formulation allowed easy application of the intrusive design. Final results of the Monte Carlo and stochastic solver were in agreement, though no computational cost comparisons were made. In another work, two types of stochastic approaches were applied to the quasi-one-dimensional supersonic nozzle problem by Mathelin et al. [48]. Finding comparable results with both a stochastic collocation method and polynomial chaos approach, they also noted the lower computational cost of the stochastic collocation method. The work of Xiu and Karniadakis [84] developed an application of generalized polynomial chaos pressure driven flow and flow past a cylinder. Compared to Monte Carlo, the results were comparable, but with a huge computational cost improvement (speed-up factor of 2500 in one case). In a geoscience related application, Kröker et al. [37] applied a hybrid stochastic Galerkin approach to the so-called quarter five-spot problem, a standard test case which simulates the flow of a wetting fluid into a porous medium initially saturated by a non-wetting fluid. Kröker et al. show convergence with increasing basis order. In each case the authors note

the inefficiency and computational cost of the Monte-Carlo method, and provide justification for the development of an intrusive framework for multiphase flow dynamics.

While some work has been presented on UQ methods in computational fluid dynamics, much of it has taken a non-intrusive approach. Additionally, previous works have discussed the application of the intrusive method only to single phase flows [36, 40]. This work seeks to fill a research gap that exists for the creation of an intrusive stochastic multiphase flow solver.

A common technique to solve for multiphase flows is the one-fluid formulation [79], which solves the Navier-Stokes in the flow domain and embeds the effects of the interface on the single solution. Some method is needed for determining the interface location as the phase interface represents a discontinuity in fluid parameters where pressure, density, and viscosity are sometimes dramatically different on either side. The presence of a surface tension force at the interface further complicates the discretization of the Navier-Stokes equations.

Methodologies to locate the gas-liquid interface can be classified as interface tracking or interface capturing. Interface tracking methods explicitly locate the interface position with either a deforming mesh [16, 30, 33] or marker particles [67]. Integrating the tracking method within an intrusive UQ framework is difficult because the gas-liquid interface becomes a distribution of interfaces leading to a distribution of meshes or marker particles. The second class of methods are interface capturing schemes, which implicitly locate the interface through an intermediate function and include both volume of fluid (VOF) and level set methods. VOF frameworks [31, 69] are conservative by capturing the liquid volume fraction at each grid cell but require a reconstruction of the interface using the cell volume fraction. In the reconstruction step a single interface is typically created within each computational cell. However,

with stochastic variables a distribution of interfaces would be needed within each cell. Level set methods [57] operate by advecting a scalar quantity of which some predefined isosurface is the interface. Level set methods suffer from non-exact conservation of mass. More accurate conservative level set techniques exist [8, 15, 54, 55, 59] which greatly improve conservation properties, though contrary to what the name suggests, these methods are not entirely mass conservative either. Furthermore, the continuous definition of the level set function allows it to be represented with polynomial chaos expansions in an intrusive UQ framework. For these reasons, the conservative level set was chosen as the method to represent the gas-liquid interface in this work.

Mathematical Model

Creation of a stochastic multiphase flow solver requires adapting traditional deterministic approaches to an intrusive stochastic framework. We'll begin with a discussion of the PC expansion used to represent stochastic variables. Next, the conservative level set interface capturing method including a stochastic surface tension, and implementation of a stochastic initial geometry will be developed. Finally, a transformation of the Navier-Stokes equations, describing the formulation of the stochastic equation set, where the UQ methodology will be applied to the incompressible form of the Navier-Stokes equations, used for gas-liquid flow in a one-fluid approach.

Uncertainty Quantification

The polynomial chaos approach was originally presented by Wiener [83], where stochastic variable $y = y(\boldsymbol{x}, t, \boldsymbol{\zeta})$, is a function of the spatial dimensions $\boldsymbol{x}$, time t, and any number of uncertainty dimensions $\boldsymbol{\zeta}$. The polynomial chaos representation

of this stochastic variable is performed with the projection

$$y(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \sum_{k=0}^{N} y_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{\zeta}) = y_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{\zeta}), \tag{2.1}$$

which uses a finite sum of spatially and temporally varying weights $y_k(\boldsymbol{x}, t)$ and orthogonal polynomial basis functions $\phi_k(\boldsymbol{\zeta})$ defined in the uncertainty dimensions. Note that Einstein summation notation is used to simplify the expression and will be employed throughout this manuscript. The accuracy of the projection is controlled by the order, N, which determines the number of weights and basis functions used. An infinite number of basis functions ($N = \infty$) results in a perfect projection that describes all possible values of $y(\boldsymbol{x}, t, \boldsymbol{\zeta})$. In practice, the number of basis functions is set to a finite number by choosing a maximum order of the polynomials, introducing a truncation error about the projection.

It is possible to use any set of orthogonal basis functions, but certain sets are well suited for representing common stochastic distributions. For example, Wiener showed [83] the Hermite polynomials represent Gaussian distributions with a small set of basis functions. Other sets of basis functions include Legendre, Laguerre, and Chebychev polynomials. However, an interesting aspect in the choice is the fact that the function being described is not always a common probability density function. For instance, projection of the level set interface capturing scheme onto basis functions describes the value of the level set as a function of uncertainty. As such, the string of polynomials will have to approximate a hyperbolic tangent function which will shift in the uncertainty domain. Because there will be variability in the shape of the functions created (i.e.velocity vs. level set), we chose to utilize the common Legendre polynomials which are orthogonal over a small range of $[-1, 1]$. Using orthogonal basis functions simplify the equations and for the Legendre polynomials the property

of orthogonality can be written as

$$\int_{-1}^{1} \phi_k \phi_b \delta\zeta = \begin{cases} \langle \phi_k \phi_b \rangle & k = b \\ 0 & k \neq b \end{cases}. \tag{2.2}$$

In this framework, uncertainty is assumed about all variables present in the governing and level set equations, including velocity, surface tension, curvature, pressure, density, and viscosity for the Navier-Stokes equations, and level set and normal vectors for the interface capturing scheme. For multiphase flows, uncertainty about the velocity field requires uncertainty about the interface location. As such, uncertainty must then be present about all other variables that are a function of space and phase.

Stochastic Level Set Transport

A number of methods for describing and locating the interface between phases exist, such as the volume of fluid (VOF) method [69], the front-tracking method [78], and the level set method [57]. Among these, the level set method is perhaps the most easily integrated into a stochastic scheme. Because level set methods use continuous partial differential equations to describe interface dynamics, they can be extended using standard polynomial chaos expansions.

While a number of level set approaches now exist [54, 55, 56], this work suggests a modified version of the accurate conservative level set method detailed by Desjardins et al. [15], where transport is achieved, assuming an incompressible flow, with

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\boldsymbol{u}\psi) = 0, \tag{2.3}$$

where ψ is the level set, t is time, and $\boldsymbol{u}$ is velocity. Next, uncertainty is assumed

about the level set scalar ψ and velocity $\boldsymbol{u}$, such that $\psi(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \psi_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{\zeta})$ and $\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \boldsymbol{u}_l(\boldsymbol{x}, t)\phi_l(\boldsymbol{\zeta})$. Using index notation and substituting stochastic variables into Eq. 2.3 leads to

$$\frac{\partial \psi_k \phi_k}{\partial t} + \nabla \cdot (\boldsymbol{u}_k \phi_k \psi_l \phi_l) = 0. \tag{2.4}$$

Using the orthogonality property of the basis functions allows solving for each basis weight ψ_k that forms the stochastic variable. To do this, Eq. 2.4 is multiplied by a single basis function ϕ_b, such that

$$\frac{\partial \psi_k \phi_k}{\partial t}\phi_b + \nabla \cdot (\boldsymbol{u}_k \phi_k \psi_l \phi_l)\, \phi_b = 0. \tag{2.5}$$

Then the equation is integrated over the region of orthogonality ($-1 \leq \zeta \leq 1$ for Legendre polynomials), leading to

$$\int_{-1}^{1} \left[\frac{\partial \psi_k \phi_k}{\partial t}\phi_b + \nabla \cdot (\boldsymbol{u}_k \phi_k \psi_l \phi_l)\, \phi_b \right] d\zeta = 0. \tag{2.6}$$

Utilizing the property of orthogonality shown in Eq. 3.6, this reduces to the stochastic transport equation

$$\frac{\partial \psi_b}{\partial t} + \nabla \cdot (\boldsymbol{u}_k \psi_l)\, C_{klb} = 0, \tag{2.7}$$

where the multiplication tensor is defined as

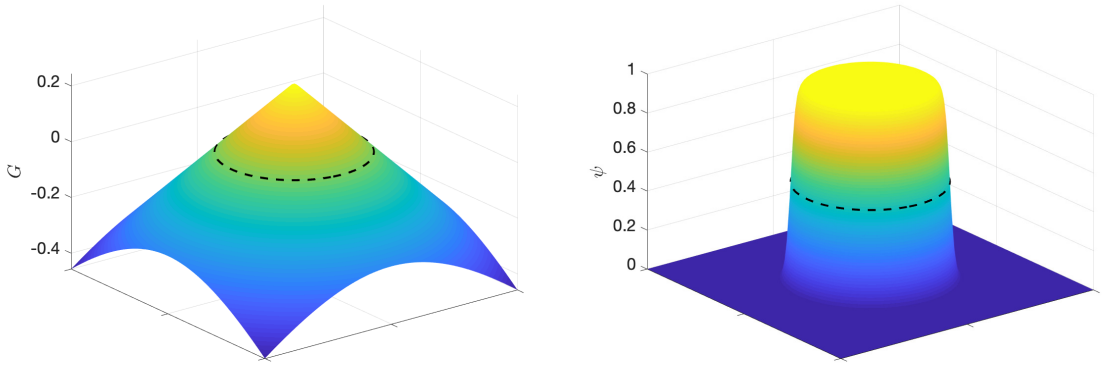
$$C_{klb} = \frac{\int_{-1}^{1} \phi_k \phi_l \phi_b \, d\zeta}{\int_{-1}^{1} \phi_b \phi_b \, d\zeta} = \frac{\langle \phi_k \phi_l \phi_b \rangle}{\langle \phi_b \phi_b \rangle}. \tag{2.8}$$

Equation 2.7 describes the transport of the level set weights ψ_b due to the fluid flow. The equation is very similar to the original transport equation (Eq. 2.3), but includes the interactions of the basis functions and weights through the multiplication tensor C_{klb}. These tensors occur throughout the governing equations and consist of integrals

of polynomials that do not change with time. Therefore, they can be calculated once at the beginning of a simulation, often leveraging analytic solutions to the integrals and requiring little computational expense. Further, many of these tensor values equal zero, allowing them to be removed from the final equation, speeding up simulation times.

Stochastic Level Set Definition

In the original level set formulation by Osher and Sethian [57], the scalar was defined as a signed distance function $g(\boldsymbol{x}, t)$, where the zero isosurface represents the interface boundary. Positive values of g indicated being within the interface while negative values were outside the interface. A two-dimensional, circular interface described by a signed distance level set is shown in Fig. 2.1a.



(a) Signed distance level set (b) Conservative level set

Figure 2.1: Comparison of signed distance and conservative level set profiles for a two-dimensional circular interface. Lines indicate the interface location.

Similar to the accurate conservative level set approach (ACLS) [15], a sharp profile about the interface is created by transforming a signed distance function $g(\boldsymbol{x}, t)$

with a sigmoid function

$$\psi(\boldsymbol{x},t) = \left[1 + \exp\left(\frac{-2g(\boldsymbol{x},t)}{\varepsilon_1 + \varepsilon_2}\right)\right]^{-1}. \tag{2.9}$$

The transformed level set scalar ψ ranges in value from $[0,1]$, with ε_1 and ε_2 controlling the slope about the interface now defined at $\psi(\boldsymbol{x},t) = 0.5$ and shown in Fig. 2.1b. Note, ε_1 and ε_2 are related to a necessary reinitialization procedure discussed below. The conservative level set improves mass conservation as it mimics the indicator function, which is a conserved quantity. This is in contrast to the signed distance level set that has no physical correspondence.

Using the conserved level set, expansion of the scalar into a stochastic framework requires substitution of uncertain variables into the above equation. Uncertainty about the initial signed distance function $g(\boldsymbol{x},t)$ may come from variability in the initial interface location, size, etc. Assuming uncertainty about both the initial signed distance function and the transformed level set scalar, substitution of stochastic variables leads to

$$\psi_k(\boldsymbol{x},t)\phi_k(\boldsymbol{\zeta}) = \left[1 + \exp\left(\frac{-2g_k(\boldsymbol{x},t)\phi_k(\boldsymbol{\zeta})}{\varepsilon_1 + \varepsilon_2}\right)\right]^{-1}. \tag{2.10}$$

Again making use of the orthogonality of the basis functions (Eq. 3.6), we can multiply Eqn. 2.10 by a basis function ϕ_b and integration over ζ leads to

$$\psi_b = \frac{1}{\langle\phi_b\phi_b\rangle}\int_{-1}^{1}\left[1 + \exp\left(\frac{-2g_k\phi_k}{\varepsilon_1 + \varepsilon_2}\right)\right]^{-1}\phi_b d\zeta, \tag{2.11}$$

which is used to compute the stochastic conservative level set weights for the initial condition. This equation has no analytic solution, but is only solved once at the beginning of the simulation to initialize the level set. Since there is only

one computation, Romberg's method is used, applying Richardson extrapolation to accelerate the convergence of the trapezoidal method [64, 66], and allowing integration to machine precision.

Stochastic Reinitialization

There is no reason, in a general velocity field, that the chosen sigmoid (smoothed Heaviside) shape of the level set profile will be maintained. As such, a reinitialization procedure is deployed which maintains the sigmoid profile of the level set. Harten [28] first described the use of a reinitialization procedure for the maintenance of a discontinuity. Sussman et al. [75] utilized a reinitialization method for maintaining a signed-distance function with application to incompressible two-phase flow. With respect to level set transport, Chopp [9] described a reinitialization procedure for maintaining a profile away from the zero level set for curvature flow, while for conservative level set transport the work of Olsson and Kreiss [54] and Olsson et al. [55] utilize reinitialization to aid in mass conservation. Many of these works proposed different forms of reinitialization with the goal of maintaining the shape of the level set without unnecessarily moving the gas-liquid interface. A commonly used reinitialization equation [15] is

$$\frac{\partial \psi}{\partial \tau} + \nabla \cdot (\psi(1-\psi)\boldsymbol{n}) = \nabla \cdot (\varepsilon(\nabla\psi \cdot \boldsymbol{n})\boldsymbol{n}), \tag{2.12}$$

where τ is pseudo-time, ε is the constant in Eq. 2.9 that controls the slope of the level set near the interface (where only one constant exists in this formulation), and $\boldsymbol{n} = \nabla\psi/|\nabla\psi|$ is the unit normal vector.

Performing reinitialization with this PDE for a stochastic level set seems promising. However, a significant challenge arises in the construction of stochastic unit normal vectors because unit normal vectors are discontinuous where the direction

changes, such as the center of the circle in Fig. 2.2a. Projecting the discontinuities onto smooth basis functions does not lead to useful results. Additionally, each basis weight $\boldsymbol{n}_b$ is found by evaluation of the integral

$$\boldsymbol{n}_b = \frac{1}{\langle \phi_b \phi_b \rangle} \int_{-1}^{1} \left[\frac{\nabla \psi_k \phi_k}{\sqrt{\left(\frac{\partial \psi_l \phi_l}{\partial x}\right)^2 + \left(\frac{\partial \psi_m \phi_m}{\partial y}\right)^2}} \right] \phi_b d\zeta, \tag{2.13}$$

which does not have an analytic solution and does not guarantee the magnitude of the unit normal will be one for all ζ.

Therefore, in this work a modified reinitialization equation is proposed,

$$\frac{\partial \psi}{\partial \tau} + \nabla \cdot (\psi(1-\psi)\boldsymbol{r}) = \nabla \cdot (\varepsilon_1 (\nabla \psi \cdot \boldsymbol{r})\boldsymbol{r}) + \nabla \cdot (\varepsilon_2 \nabla \psi), \tag{2.14}$$

which has a couple of differences to point out. The first and most significant is the use of a continuous (non-unit) normal vector,

$$\boldsymbol{r} = \frac{\nabla \psi}{|\nabla \psi|_{\max}}, \tag{2.15}$$

which describes the outward normal direction about the level set, as shown in Fig. 2.2b. This vector is scaled by $|\nabla \psi|_{\max}$, i.e., the magnitude of the maximum gradient of the level set. This value is a constant for the duration of the simulation and can be found by considering a reduced one-dimensional steady-state reinitialization equation, where

$$\frac{d}{dx}(\psi(1-\psi)\boldsymbol{r}) = \frac{d}{dx}(\varepsilon_1 (\nabla \psi \cdot \boldsymbol{r})\boldsymbol{r}) + \frac{d}{dx}\left(\varepsilon_2 \nabla \psi\right). \tag{2.16}$$

This equation can be integrated over x, leaving

$$(\psi(1-\psi)\boldsymbol{r}) = (\varepsilon_1(\nabla\psi\cdot\boldsymbol{r})\boldsymbol{r}) + (\varepsilon_2\nabla\psi)\,. \tag{2.17}$$

At the interface $\psi = 1/2$, $\boldsymbol{r} = \nabla\psi/|\nabla\psi|_{\max} = 1$, and $d\psi/dx = |\nabla\psi|_{\max}$. Plugging these in we have

$$\frac{1}{2}\left(1-\frac{1}{2}\right) = \varepsilon_1|\nabla\psi|_{\max} + \varepsilon_2|\nabla\psi|_{\max}. \tag{2.18}$$

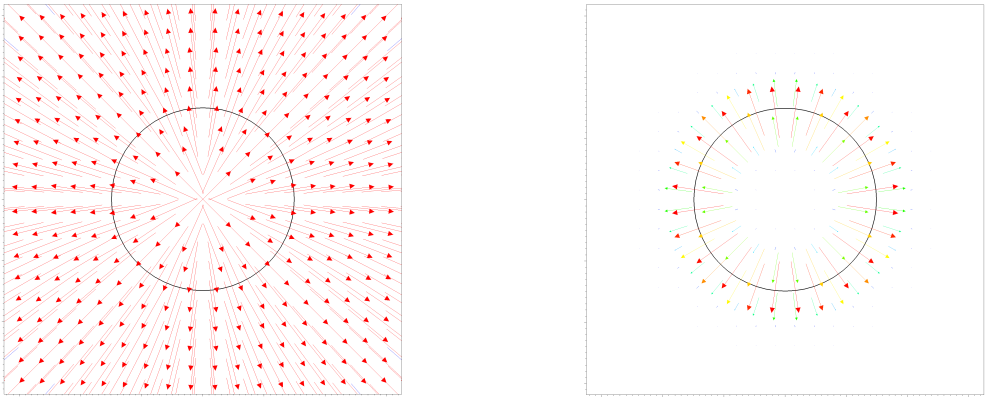
Finally, a bit of algebra leads us to our analytic solution of the maximum gradient,

$$|\nabla\psi|_{\max} = \frac{1}{4\left(\varepsilon_1+\varepsilon_2\right)}, \tag{2.19}$$

where the maximum gradient is defined by the ε values that control the speed with which the reinitialization function diffuses. It can also be pointed out that the epsilon values may be defined by a chosen maximum gradient scalar e.g., $d\psi/d\boldsymbol{x} = 1/3\Delta x$.

Implementation of the continuous normal vector (Eq. 2.15), offers substantial benefits. Since $\boldsymbol{r}$ is scaled by the maximum magnitude of the initial condition (a constant), these vectors have a theoretical magnitude of $[0, 1]$. As such, the vectors reach a maximum at the interface, and approach zero away from the interface, as shown in Fig. 2.2. Use of previously defined unit normal vectors required a procedure in which reinitialization only occurred within a few cells of the interface, where $\psi(\boldsymbol{x}, t) \approx 0.5$. Removing this procedure decreased computational cost. Additionally, the continuous nature of $\boldsymbol{r}$ is more suited to transformation into a stochastic random variable described by a string of continuous orthogonal polynomials. It is useful to note these continuous normal vectors are also of use in a classical, deterministic model. Highlighting the efficiency increase by removing the need for a banding procedure, as well as the ability of the reinitialization step to converge using continuous normal

vectors, their application exists outside the intrusive UQ realm, and will be the subject of future work. For example, they may be used to more efficiently reinitialize a deterministic level set, or may be useful in the calculation of the interface curvature.



(a) $\boldsymbol{n} = \frac{\nabla\psi}{|\nabla\psi|}$ (b) $\boldsymbol{r} = \frac{\nabla\psi}{|\nabla\psi|_{\max}}$

Figure 2.2: Comparison of normal vectors used in the reinitialization procedure about a simple circle.

Another difference of this reinitialization scheme is the addition of a general diffusion term $(\nabla \cdot (\varepsilon_2 \nabla\psi))$. This term is added because the continuous normal vectors modify the diffusion term $\nabla \cdot (\varepsilon_1(\nabla\psi \cdot \boldsymbol{r})\boldsymbol{r})$ such that it is only non-zero near the interface. Diffusion still needs to take place away from the interface to avoid accumulating errors; another diffusion term that is non-zero away from the interface is needed. The proposed term $\nabla\cdot(\varepsilon_2\nabla\psi)$ is the simplest option and is straightforward to add into the stochastic PC expansions. Note that more complicated terms could be added that avoid the extra diffusion near the interface caused by this term. For example, the term $(1-|\boldsymbol{r}|)\nabla\cdot(\varepsilon_2\nabla\psi)$ is only non-zero away from the interface and will be considered in future work. We do not consider this form here because including the magnitude operator causes challenges in the polynomial chaos expansion.

To transform the modified reinitialization equation (Eq. 4.16) into the stochastic

framework, stochastic variables are substituted such that

$$\begin{aligned}\frac{\partial \psi_k \phi_k}{\partial \tau} + \nabla \cdot \left(\psi_k \phi_k \left(1 - \psi_l \phi_l\right) \boldsymbol{r}_m \phi_m\right) = \\ \nabla \cdot \left(\varepsilon_1 \left(\nabla \psi_k \phi_k \cdot \boldsymbol{r}_l \phi_l\right) \boldsymbol{r}_m \phi_m\right) + \nabla \cdot \left(\varepsilon_2 \nabla \psi_k \phi_k\right).\end{aligned} \tag{2.20}$$

With a bit of algebra, and using the same methodology as with the transport equation (multiplying through by a test function, integrating over ζ, and utilizing the property of orthogonality), a final stochastic reinitialization equation for the calculation of each basis weight ψ_b with respect to pseudo-time is

$$\begin{aligned}\frac{\partial \psi_b}{\partial \tau} = &- \nabla \cdot \left(\psi_k \boldsymbol{r}_l C_{klb} - \psi_k \psi_m \boldsymbol{r}_l C_{klmb}\right) \\ &+ \nabla \cdot \left(\varepsilon_1 \left(\nabla \psi_k \cdot \boldsymbol{r}_l\right) \boldsymbol{r}_m\right) C_{klmb} + \nabla \cdot \left(\varepsilon_2 \nabla \psi_b\right),\end{aligned} \tag{2.21}$$

where

$$C_{klmb} = \frac{\langle \phi_k \phi_l \phi_m \phi_b \rangle}{\langle \phi_b \phi_b \rangle}. \tag{2.22}$$

Stochastic Multiphase Navier-Stokes

As previously mentioned, some work has already been done in the area of *single-phase*, incompressible flows with intrusive uncertainty quantification. Perhaps the earliest such work was presented by Le Maître et al. [39] and [40]. These works were further expanded upon in a textbook by Le Maître and Knio [38], including multiple applications and comparisons of various spectral expansion types for the establishment of stochastic random variables. These works provide a rigorous development of stochastic Navier-Stokes for single-phase flow, and are referred to for a more expansive mathematical treatment. Application of these works to single-phase flow mechanics serve as the starting point and guide for the multiUQ project.

The incompressible Navier-Stokes equations, including a force term for the

surface tension between phases, is

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{1}{\rho}\nabla P + \nu \nabla^2 \boldsymbol{u} + \frac{1}{\rho}\boldsymbol{f}_\gamma \delta_s, \tag{2.23}$$

where P is pressure, ρ is fluid density, ν is viscosity, and $\boldsymbol{f}_\gamma \delta_s$ is the surface tension force described in detail in Section 2. Conservation of mass, with the incompressible assumption, leads to

$$\nabla \cdot \boldsymbol{u} = 0. \tag{2.24}$$

In order to make integration of the stochastic variables simpler and avoid a division by basis functions, the division by density is handled using a dummy variable $\rho^{\mathrm{I}} = 1/\rho$. We then have uncertainty possible about the inverse density where appropriate as well as density where necessary. The resulting equation is

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\rho^{\mathrm{I}}\nabla P + \nu \nabla^2 \boldsymbol{u} + \rho^{\mathrm{I}}\boldsymbol{f}_\gamma \delta_s. \tag{2.25}$$

As shown previously, application of polynomial chaos requires importing stochastic random variables to transform these equations into a weak form that allows solving for the basis weights of velocity. Assuming uncertainty about all variables (velocity $\boldsymbol{u}_k\phi_k$, density $\rho_k\phi_k$, viscosity $\nu_k\phi_k$, pressure $P_k\phi_k$, and surface tension $\boldsymbol{f}_{\gamma:k}\phi_k$) and combining the stochastic variables with the Navier-Stokes equations leads to one form of the stochastic Navier-Stokes equations,

$$\frac{\partial \boldsymbol{u}_k\phi_k}{\partial t} + \boldsymbol{u}_k\phi_k \cdot \nabla \boldsymbol{u}_l\phi_l = -\rho_k^{\mathrm{I}}\phi_k\nabla P_l\phi_l + \nu_k\phi_k\nabla^2\boldsymbol{u}_l\phi_l + \rho_k^{\mathrm{I}}\phi_k\boldsymbol{f}_{\gamma:l}\phi_l\delta_s \tag{2.26}$$

and stochastic continuity,

$$\nabla \cdot \boldsymbol{u}_k\phi_k = 0 \tag{2.27}$$

Multiplication by a single basis function ϕ_b, use of the orthogonality property, and a bit of algebra leads to a useful form of the stochastic Navier-Stokes equations,

$$\frac{\partial \boldsymbol{u}_b}{\partial t} = -\boldsymbol{u}_k \cdot \nabla \boldsymbol{u}_l C_{klb} - \rho_k^{\mathrm{I}} \nabla P_l C_{klb} + \nu_k \nabla^2 \boldsymbol{u}_l C_{klb} + \rho_k^{\mathrm{I}} \boldsymbol{f}_{\gamma:l} \delta_s C_{klb} \tag{2.28}$$

and

$$\nabla \cdot \boldsymbol{u}_b = 0. \tag{2.29}$$

Stochastic Curvature and Surface Tension

Having described the level set methodology to capture the interface between fluid phases and the creation of the stochastic Navier-Stokes, the last step is to calculate the surface tension force. Surface tension can be written as

$$\boldsymbol{f}_\gamma \delta_s = \gamma \kappa \boldsymbol{n} \delta_s, \tag{2.30}$$

where γ is the surface tension coefficient, κ is the interface curvature, $\boldsymbol{n}$ is the outward facing interface normal vector, and δ_s is a surface Dirac Delta function that is non-zero only at the interface. The curvature is computed with

$$\kappa = -\nabla \cdot \boldsymbol{n}, \tag{2.31}$$

where the unit normal vector is calculated from the level set with

$$\boldsymbol{n} = \frac{\nabla \psi}{|\nabla \psi|}. \tag{2.32}$$

Calculation of the curvature using this unit normal is a common approach. However, it causes issues with the PC expansion, where projection of $\boldsymbol{n}$ onto continuous polynomials is ill-suited. Substitution of Eq. 2.32 into Eq. 2.31, importing stochastic

variables, and multiplying by a test function and integrating, a stochastic curvature is

$$\kappa_b = \frac{-1}{\langle \phi_b \phi_b \rangle} \int_{-1}^{1} \nabla \cdot \frac{\nabla \psi_k \phi_k}{|\nabla \psi_l \phi_l|} d\zeta. \tag{2.33}$$

While this integration is possible through numerical means, it lacks an analytic solution. More useful would be the application of continuous normal vector $\boldsymbol{r}$ for calculation of a stochastic curvature, which avoids a numerical integration. This is intended to be the subject of future work.

Having the surface tension coefficient, γ, and the curvature, κ, the last piece needed is to approximate the normal vector and Dirac delta function, which determine the direction and location of the force. A common approach is the continuum surface force method [7], in which the surface tension force is smeared over a few computational cells near the interface boundary. Tryggvason et al. [79] describe the use of the Heaviside function, H, to clearly define the separation between the two fluid phases such that

$$\nabla H = \boldsymbol{n} \delta_s. \tag{2.34}$$

The conservative level set scalar can be used as a smoothed approximation of the Heaviside function, where

$$\nabla \psi \approx \nabla H, \tag{2.35}$$

which is used to calculate the direction of the surface tension force and smooth it over a few cells. Thus, the gradient of ψ is used to approximate $\boldsymbol{n}\delta_s$. With this definition, calculation of the *smoothed* surface tension force is

$$\boldsymbol{f}_\gamma = \gamma \kappa \nabla \psi. \tag{2.36}$$

Next, this approximation of the surface tension force must be expanded into a

stochastic regime. Each variable will be allowed to vary in uncertainty dimension ζ. Substitution of stochastic variables into Eq. 2.36 leads to

$$\boldsymbol{f}_{\gamma:k}\phi_k = \gamma_k\phi_k\kappa_l\phi_l\nabla\psi_m\phi_m, \tag{2.37}$$

where variability may be present due to uncertainty about the surface tension coefficient, curvature, or location of the interface, creating a distribution of surface tension forces at each computational cell. Again leveraging the principal of orthogonality, this equation is multiplied by a test function and integrated over ζ. Solving for the b^{th} basis weight, a stochastic surface tension force can be found with

$$\boldsymbol{f}_{\gamma:b} = \gamma_k\kappa_l\left(\nabla\psi_m\right)C_{klmb}. \tag{2.38}$$

Numerical Methods

The proposed multiUQ framework is an open-source project under the GNU General Public License that is available for download via Bitbucket. As this code is currently under development, updates occur regularly. Current discretizations and implementation of the mathematics presented above are detailed in this section.

multiUQ is implemented on a structured two-dimensional staggered grid. Density ($\rho_{i,j}$), viscosity ($\nu_{i,j}$), pressure ($P_{i,j}$), and the level set values ($\psi_{i,j}$) are stored at the cell center, as shown in Fig. 4.3. Velocity ($u_{i-1/2,j}$, $v_{i,j-1/2}$) and surface tension force coefficients ($f_{\gamma,x:i-1/2,j}$, $f_{\gamma,y:i,j-1/2}$) are located at the left and bottom cell faces for the x and y components, respectively. Time marching is accomplished with an iterative Crank-Nicolson approach wrapped around the pressure correction method.

Following the order of the implementation, this section will first cover the time-marching and discretizations used to solve the Navier-Stokes equations on a structured

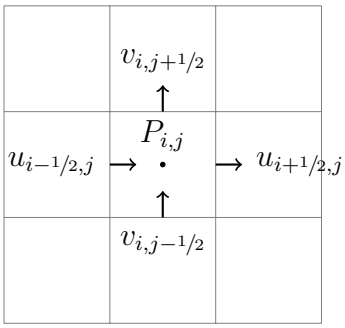


Figure 2.3: Schematic of staggered grid used for the stochastic multiphase solver.

grid, followed by the numerical scheme used to compute the curvature and surface tension about the phase interface. With these calculations, the level set transport scheme is discretized and advanced in time with the same methods used for the Navier-Stokes equations. Finally, the numerical scheme for finding the continuous normal vectors and completing the reinitialization procedures are discussed.

Navier-Stokes

The Navier-Stokes equations are dependent on fluid density and viscosity. In a deterministic model, these fluid properties are set as a jump condition at the $\psi = 0.5$ interface, i.e., $\rho = \rho_1 H(\psi - 0.5) + \rho_2 \left(1 - H(\psi - 0.5)\right)$. Instead, density and viscosity for a stochastic multiphase framework is calculated using the level set, causing them to vary smoothly with ψ. This is needed to avoid the discontinuity at the interface and take advantage of the continuous nature of the basis functions. Since the level set ranges between 0 and 1, the density can be found using

$$\rho = \rho_1 \psi + (1 - \psi)\rho_2, \tag{2.39}$$

where ρ_1 is the density within the interface (e.g. liquid) and ρ_2 is the density of the surrounding fluid (e.g. gas). Equation 2.39 is made stochastic by substitution of

random variables, multiplication by a test function, integration over ζ, and a bit of algebra leaving

$$\rho_b = \rho_{2,b} + \rho_{1,k}\psi_l C_{klb} - \rho_{2,k}\psi_l C_{klb}. \tag{2.40}$$

This equation assumes the possibility for uncertainty about both ρ_1 and ρ_2. The calculation of stochastic viscosity is handled similarly, where

$$\nu_b = \nu_{2,b} + \nu_{1,k}\psi_l C_{klb} - \nu_{2,k}\psi_l C_{klb}. \tag{2.41}$$

Time discretization for the Navier-Stokes equations involves a pressure-correction method inside an iterative Crank-Nicolson approach [14]. In a stochastic solver, the pressure correction method works very similarly to a deterministic model, with the addition of several loops to account for multiplication over the basis functions. First looking at the simple case of a deterministic explicit Euler discretization, the pressure-correction model is contained by Eqns. 3.16 and 3.17, where the Navier Stokes equations are broken into a predictor and a corrector step, respectively. Addition of these two equations leaves a typical explicit Euler discretization of the time derivative.

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} + \boldsymbol{u}^n \cdot \nabla \boldsymbol{u}^n = \nu \nabla^2 \boldsymbol{u}^n + \frac{1}{\rho}\boldsymbol{f}_\gamma^n \delta_s \tag{2.42}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\frac{1}{\rho}\nabla P^n \tag{2.43}$$

After predicting the value of the velocity field ($\boldsymbol{u}^*$) by leaving out the pressure differential, the next step is to correct with the pressure field. The pressure is calculated by solving an elliptic equation that is found by taking the divergence of the corrector step in Eq. 3.17 leading to

$$\nabla \cdot \frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = \nabla \cdot \frac{-1}{\rho}\nabla P^n, \tag{2.44}$$

which is the standard variable density pressure Poisson equation. Plugging in stochastic variables we have

$$\nabla \cdot \frac{\boldsymbol{u}_k^{n+1}\phi_k - \boldsymbol{u}_k^*\phi_k}{\Delta t} = \nabla \cdot \left(\rho_k^{\mathrm{I}}\phi_k \nabla P_l^n \phi_l\right), \tag{2.45}$$

where the right hand side is non-linear about ζ. Since the gradient of ρ^{I} is not zero, and is a function of the smoothed Heaviside function ψ, it is useful to expand the right hand side. Also, the velocity field of the next time step is to remain divergence free ($\nabla \cdot \boldsymbol{u}_k^{n+1}\phi_k = 0$). Eq. 2.45 then becomes

$$\frac{\nabla \cdot \boldsymbol{u}_k^*\phi_k}{\Delta t} = \nabla\rho_k^{\mathrm{I}}\phi_k \cdot \nabla P_l^n\phi_l + \rho_k^{\mathrm{I}}\phi_k\nabla^2 P_l^n\phi_l. \tag{2.46}$$

Again, due to the relationship between ρ and P, the solution of pressure is not decoupled from density. To isolate the Laplacian of pressure and create the pressure Poisson equation, equation 2.46 is multiplied by density and projected onto the polynomial chaos basis functions leading to

$$\nabla^2 P_b^n = \rho_k \frac{\nabla \cdot \boldsymbol{u}_l^*}{\Delta t} C_{klb} - \rho_k \nabla\rho_l^{\mathrm{I}} \cdot \nabla P_m^n C_{klmb} \tag{2.47}$$

that is solved for each of the stochastic pressure weights. Due to the non-linearity between density and pressure, these equations are coupled and are solved using the Gauss-Seidel method where

$$\begin{aligned} P_{b:i,j}^n = &\frac{(\Delta x)^2(\Delta y)^2\left(\rho_k\frac{\nabla\cdot\boldsymbol{u}_l^*}{\Delta t}C_{klb} - \rho_k\nabla\rho_l^{\mathrm{I}}\cdot\nabla P_m^n C_{klmb}\right)}{-2\left[(\Delta x)^2 + (\Delta y)^2\right]} \\ &+ \frac{(\Delta y)^2\left(P_{b:i-1,j}^n + P_{b:i+1,j}^n\right) + (\Delta x)^2\left(P_{b:i,j-1}^n + P_{b:i,j+1}^n\right)}{-2\left[(\Delta x)^2 + (\Delta y)^2\right]}. \end{aligned} \tag{2.48}$$

This equation is solved iteratively until the residual reaches a tolerance level.

Computational expense is reduced by recognizing that the $\nabla \cdot \boldsymbol{u}^*$ term doesn't change and is calculated once per time step. Unfortunately, each pressure iteration requires looping over the multiplication tensor C_{klmb}, which grows exponentially with the number of basis functions. Looping over C_{klmt} for 25 basis functions requires 25^4 iterations. However, many of these values are zero. For the case of 25 basis functions, only 4,147 and 162,969 non-zero values exist for the C_{klb} and C_{klmb} tensors, respectively. This reduces the C_{klmb} tensor by more than half ($\approx 41.7\%$), and the C_{klb} tensor by almost three-quarters ($\approx 26.5\%$). Relieving the program of so many iterations provides a huge computational cost benefit. Again, since these integrals need only be calculated once at the beginning of the simulation, all non-zero integrals and their corresponding index values (k, l, m, b) can be stored in a reduced tensor, which is then looped over for the remainder of the simulation. More efficient elliptic solvers could be used and will be explored in future work.

Once the pressure field has been solved for, the corrector step can be taken, where

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{\Delta t}{\rho} \nabla P^n \tag{2.49}$$

for an explicit Euler update. Utilizing the iterative Crank-Nicolson scheme employed in multiUQ, the predictor, pressure, and corrector calculations are all performed twice for each time step to achieve second-order accuracy. As such, the stochastic predictor step is

$$\boldsymbol{u}^*_{b:i,j} = \boldsymbol{u}^n_{b:i,j} + \Delta t \left(-\boldsymbol{u}^{n+1/2}_{k:i,j} \cdot \nabla \boldsymbol{u}^{n+1/2}_{l:i,j} + \nu_k \nabla^2 \boldsymbol{u}^{n+1/2}_{l:i,j} + \rho^{\mathrm{I}}_{k:i,j} \boldsymbol{f}^{n+1/2}_{\gamma l:i,j} \right) C_{klb}, \tag{2.50}$$

while the corrector step and final calculation of velocity is

$$\boldsymbol{u}^{n+1/2}_{b:i,j} = \boldsymbol{u}^*_{b:i,j} + \Delta t \left(-\rho^{\mathrm{I}}_{k:i,j} \nabla P^{n+1/2}_{l:i,j} \right) C_{klb}. \tag{2.51}$$

Mid-point time values for velocity are found with

$$\boldsymbol{u}_{b:i,j}^{n+1/2} = \frac{1}{2}\left(\boldsymbol{u}_{b:i,j}^{n} + \boldsymbol{u}_{b:i,j}^{n+1}\right), \tag{2.52}$$

and level set values ψ are found in a similar procedure. Additionally, density, viscosity, and pressure are found at the mid-time steps.

Stability of the time marching method is important. With the second-order accurate iterative Crank-Nicolson scheme, the time step is limited by either the viscous, advection, or surface tension force, as suggested by Gutierrez et. al [27]. The time step is found by

$$\Delta t = \text{CFL} \min\left[\frac{h_{\min}}{|\boldsymbol{u}|_{\max}}, \frac{h_{\min}^2}{2\nu}, \left(\frac{h_{\min}^3(\rho_1+\rho_2)}{4\pi\sigma}\right)^{1/2}\right]. \tag{2.53}$$

Curvature and Surface Tension

As mentioned in Section 2, projection of a unit normal vector onto a string of basis functions is problematic. Since the curvature is calculated at every time step and Romberg's method is somewhat expensive, a quadrature is used to project $\boldsymbol{n}$ onto basis functions. Rather than force a discontinuous vector onto continuous basis functions, a Gaussian quadrature scheme is applied in which unit normal vectors are calculated at each quadrature point, then used to determine the curvature. To numerically solve Eq. 2.33 for curvature basis weight κ_l, we use

$$\kappa_l = \frac{1}{\langle\phi_l\phi_l\rangle}\sum_{a=1}^{N_q} W_a\left(-\nabla\cdot\boldsymbol{n}_a\right) = \frac{1}{\langle\phi_l\phi_l\rangle}\sum_{a=1}^{N_q} W_a\kappa_a \tag{2.54}$$

for abscissa C_a, weight W_a, and outward normal vector $\boldsymbol{n}_a$ at each point, with a total of N_q quadrature points. By calculating a normal vector at each quadrature point we

can avoid any issues in discontinuity and satisfy the unit normal condition necessary to properly evaluate $-\nabla \cdot \boldsymbol{n}$. For these simulations $N_q = 50$ quadrature points were used, which seemed a reasonable balance of accuracy and efficiency. There is need to determine if the number of quadrature points needed is greater with more basis functions.

Prior to performing the quadrature, basis weights $\nabla\psi_b$ are calculated with a second-order finite difference approach. Within the quadrature, the unit normal vectors $\boldsymbol{n}_a$ are evaluated at each abscissa, where

$$n_{(x,a)} = \frac{\left(\partial\psi/\partial x\right)_k \phi_k(C_a)}{\sqrt{\left[\left(\partial\psi/\partial x\right)_l^2 + \left(\partial\psi/\partial y\right)_l^2\right] \phi_l\left(C_a\right)}} \tag{2.55}$$

for the x component of the normal vector. The y component is calculated similarly. The curvature at each abscissa is also found with a second-order finite difference approach, such that

$$\kappa_a = -\nabla \cdot \boldsymbol{n}_a = -\left(\frac{n_{(x,a)i+1,j} - n_{(x,a)i-1,j}}{2\Delta x} + \frac{n_{(y,a)i,j+1} - n_{(y,a)i,j-1}}{2\Delta y}\right), \tag{2.56}$$

which is located at the cell center. Finally, this curvature, which is associated with the a^{th} abscissa is used in Eq. 2.54 to compute the stochastic curvature weights κ_l.

The continuous surface force method, described by Brackbill et al. [7], is used to discretize the surface tension force. The method is also employed in other works [54, 55, 79] with reasonable results. Having calculated the curvature with Eq. 2.54, the next step is to integrate the surface tension into the intrusive UQ scheme and discretize. Adapting the method described by Brackbill [7], a discretized,

stochastic surface tension can be calculated over a smoothed interface with

$$\boldsymbol{f}_{\gamma b} = \gamma_k \kappa_l \left(\nabla \psi_m\right) C_{klmb}, \tag{2.57}$$

which is Eq. 2.38 repeated. The gradient of the level set is calculated using a second-order central difference approximation where a gradient for basis weight m at location i, j is

$$\nabla \psi_{m:i,j} = \left(\frac{\psi_{m:i+1,j} - \psi_{m:i-1,j}}{2\Delta x}\right) \hat{\boldsymbol{\imath}} + \left(\frac{\psi_{m:i,j+1} - \psi_{m:i,j-1}}{2\Delta y}\right) \hat{\boldsymbol{\jmath}}. \tag{2.58}$$

With all necessary information, the surface tension force can be calculated directly using

$$\boldsymbol{f}_{b:i,j} = \gamma_k \kappa_{l:i,j} \nabla \psi_{m:i,j} C_{klmb}, \tag{2.59}$$

which is located at the cell center. Since this force acts as a source term and needs to be added to the momentum equation, it is necessary to interpolate these values to the cell faces, where the velocity vectors are stored on the staggered grid. A simple linear interpolation is done, where

$$\begin{aligned} f_{x,b:i-1/2,j} &= \frac{1}{2}\left(f_{x,b:i,j} + f_{x,b:i-1,j}\right) \\ f_{y,b:i,j-1/2} &= \frac{1}{2}\left(f_{y,b:i,j} + f_{y,b:i,j-1}\right) \end{aligned}, \tag{2.60}$$

moving the x and y components to the left and bottom cell walls, respectively.

Level Set Transport

Time discretization for level set transport is also done using the iterative Crank-Nicolson scheme, where the velocity and level set updates occur in the same loop. In

a deterministic case, the scalar field ψ is advected with

$$\psi_{i,j}^{n+1} = \psi_{i,j}^{n} - \frac{\Delta t}{2}\left[\nabla\cdot\left(\boldsymbol{u}_{i,j}^{n}\psi_{i,j}^{n}\right) + \nabla\cdot\left(\boldsymbol{u}_{i,j}^{n+1}\psi_{i,j}^{n+1}\right)\right]. \tag{2.61}$$

Extending this for the polynomial chaos methodology leads to the discretized stochastic update equation

$$\psi_{b:i,j}^{n+1} = \psi_{b:i,j}^{n} - \frac{\Delta t}{2}\left[\nabla\cdot\left(\boldsymbol{u}_{k:i,j}^{n}\psi_{l:i,j}^{n}\right)C_{klb} + \nabla\cdot\left(\boldsymbol{u}_{k:i,j}^{n+1}\psi_{l:i,j}^{n+1}\right)C_{klb}\right]. \tag{2.62}$$

Since the level set ψ is stored at the cell center, a fifth-order upwind central (HOUC-5) scheme [53], interpolates ψ to the cell face where velocity is stored. The divergence operator is discretized and acts over the cell in a finite volume approach such that

$$\nabla\cdot\boldsymbol{u}\psi = \frac{u_{i+1/2,j}\psi_{i+1/2,j} - u_{i-1/2,j}\psi_{i-1/2,j}}{\Delta x} + \frac{v_{i,j+1/2}\psi_{i,j+1/2} - v_{i,j-1/2}\psi_{i,j-1/2}}{\Delta y}, \tag{2.63}$$

which provides the update ψ at the cell center. In the stochastic implementation, this becomes

$$\begin{aligned}\nabla\cdot\left(\boldsymbol{u}_{k:i,j}\psi_{l:i,j}\right)C_{klb} = \Big(&\frac{u_{k:i+1/2,j}\psi_{l:i+1/2,j} - u_{k:i-1/2,j}\psi_{l:i-1/2,j}}{\Delta x} + \\ &\frac{v_{k:i,j+1/2}\psi_{l:i,j+1/2} - v_{k:i,j-1/2}\psi_{l:i,j-1/2}}{\Delta y}\Big)C_{klb}.\end{aligned} \tag{2.64}$$

Continuous Normal Vectors

A key difference from the implementation of the conservative level set scheme presented by Desjardins et al. [15] is the introduction of continuous (non-unit) interface normal vectors. Projecting unit normal vectors onto smooth basis functions is ill-defined as they are discontinuous where the direction changes (i.e. a jump occurs

in component values, as seen in the center of Fig. 2.2). Additionally, projection onto basis functions requires solving an integral without an analytic solution due to the division by the magnitude $|\nabla\psi_k\phi_k|$. Rather than implementing a numerical integration scheme, the continuous normal vector is used (Eq. 2.15), which is scaled by $|\nabla\psi|_{\max} = 1/4(\varepsilon_1 + \varepsilon_2)$. In this work $\varepsilon_1 = \max(\Delta x, \Delta y)$ and $\varepsilon_2 = \max(\Delta x, \Delta y)/10$, which together control the smearing of the smoothed Heaviside function. It can also be considered to control the number of cells it takes to capture the interface.

Near the interface, the gradient of the level set function is large, while away from the interface the gradient is approximately zero. To calculate continuous normal vectors, a planar least squares method is used. This helps to smooth ripples caused by the reinitialization procedure, and has the added benefit of working nicely in the polynomial chaos setting. For a two-dimensional framework, the level set at a grid cell is approximated with

$$\psi = Ax + By + C. \tag{2.65}$$

This is a standard least-squares approach. However, in a stochastic framework, we must include the variation about ζ for ψ, such that $A = A_k\phi_k$, $B = B_k\phi_k$, and $C = C_k\phi_k$. Making this substitution, Eq. 2.65 becomes

$$\psi_k\phi_k = A_k\phi_k x + B_k\phi_k y + C_k\phi_k. \tag{2.66}$$

Subtracting the averages for each variable $(\overline{\psi}, \overline{x}, \overline{y})$ moves the spatial origin to the point of interest leaving

$$\psi_k\phi_k - \overline{\psi_k\phi_k} = A_k\phi_k\left(x - \overline{x}\right) + B_k\phi_k\left(y - \overline{y}\right) + C_k\phi_k. \tag{2.67}$$

The constant C will cancel with $\overline{\psi_k\phi_k}$. Multiplication by a test function ϕ_b, integration

over ζ, and using the orthogonality property leads to

$$\psi_b = A_b(x - \overline{x}) + B_b(y - \overline{y}). \tag{2.68}$$

Differentiating this equation with respect to x and y it is easy to show $A_b = \partial\psi_b/\partial x$ and $B_b = \partial\psi_b/\partial y$, which are the terms used to define the continuous normal vector $\boldsymbol{r}$. Using these coefficients the pieces of the continuous normal vector are

$$r_{x,b} = \frac{A_b}{|\nabla\psi|_{\max}}, \tag{2.69}$$

$$r_{y,b} = \frac{B_b}{|\nabla\psi|_{\max}}. \tag{2.70}$$

This planar least squares fit is completed for each basis weight with a square grid centered on each cell. With a two-dimensional system, the smallest number of points used to calculate the normal at any given location $\psi_{i,j}$ is 3×3. Depending on the mesh size, it can be beneficial to increase the number of points used. However, it is important to note that if the normal grid size is too large, the sharpness of possible corners is reduced. In the results presented in this manuscript a stencil of 3×3 is used.

Reinitialization

A finite volume approach to the discretization of the reinitialization function was used. Conservation of the level set ψ, and thus mass (given that the level set ψ is used to calculated density), is the primary concern. For example, in this approach the change in the value of the scalar ψ across each cell is a measure of flux through the center of the cell, such that

$$\frac{\partial\psi_i}{\partial x} = \frac{\psi_{i+1/2} - \psi_{i-1/2}}{\Delta x}. \tag{2.71}$$

The value at $\psi_{i+1/2}$ and the value at $\psi_{i-1/2}$ are the values at the right and left side of the cell, respectively, and must be interpolated to the cell walls. Derivatives in the y direction can be found similarly.

Breaking down Eq. 4.16 into the compression, diffusion, and general diffusion terms, we first distribute the subtraction of the compression term, then evaluate the divergence such that

$$\nabla \cdot (\psi(1-\psi)\boldsymbol{r}) = \nabla \cdot (\psi \boldsymbol{r} - \psi\psi\boldsymbol{r}) = \frac{d}{dx}\left(\psi r_x - \psi\psi r_x\right) + \frac{d}{dy}\left(\psi r_y - \psi\psi r_y\right). \quad (2.72)$$

We then calculate the x-derivative using values at the cell walls,

$$\begin{aligned}\frac{d}{dx}\left(\psi r_x - \psi\psi r_x\right) =& \frac{\psi_{i+1/2} r_{x,i+1/2} - \psi_{i-1/2} r_{x,i-1/2}}{\Delta x} - \\ & \frac{\psi_{i+1/2}\psi_{i+1/2} r_{x,i+1/2} - \psi_{i-1/2}\psi_{i-1/2} r_{x,i-1/2}}{\Delta x},\end{aligned} \quad (2.73)$$

where the cell center located variables ψ and $\boldsymbol{r}$ are interpolated to the cell walls with $\psi_{i+1/2} = 0.5\left(\psi_{i+1} + \psi_i\right)$ and $r_{x,i+1/2} = 0.5\left(r_{x,i+1} + r_{x,i}\right)$. Calculation of the y-derivative can be found in a similar manner. Distribution of the divergence operator on the diffusion term is

$$\nabla \cdot (\varepsilon_1(\nabla\psi \cdot \boldsymbol{r})\boldsymbol{r}) = \frac{d}{dx}\left(\varepsilon_1 \frac{d\psi}{dx} r_x r_x\right) + \frac{d}{dy}\left(\varepsilon_1 \frac{d\psi}{dy} r_y r_y\right). \quad (2.74)$$

Next, given $\left(d\psi/dx\right)_{i+1/2} = \left(\psi_{i+1} - \psi_i\right)/\Delta x$, discretization of the x-derivative for the interface diffusion term is then

$$\nabla \cdot (\varepsilon_1(\nabla\psi \cdot \boldsymbol{r})\boldsymbol{r}) = \varepsilon_1 \left[\frac{\left(\psi_{i+1} - \psi_i\right) r_{x,i+1/2} r_{x,i+1/2} - \left(\psi_i - \psi_{i-1}\right) r_{x,i-1/2} r_{x,i-1/2}}{\left(\Delta x\right)^2}\right]. \quad (2.75)$$

Finally, the discretized calculation of the general diffusion term is accomplished with

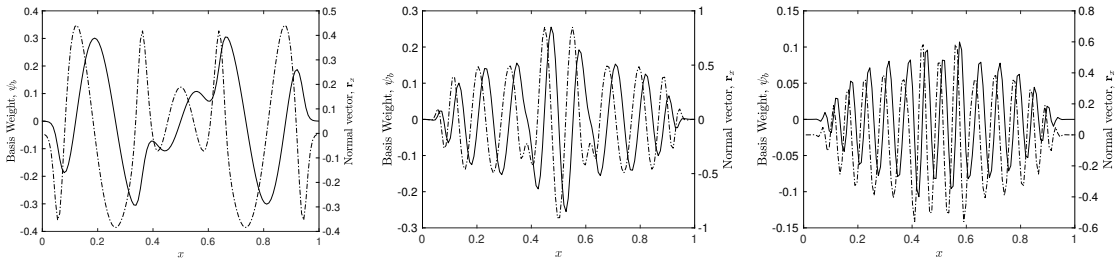
$$\nabla \cdot (\varepsilon_2 \nabla \psi) = \varepsilon_2 \left(\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{(\Delta x)^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{(\Delta y)^2} \right). \tag{2.76}$$

Each step of the reinitialization equation takes place in pseudo-time, a fictitious time used to solve the reinitialization PDE and maintain the shape of the level set profile which helps conserve mass. Since the reinitialization equation is of a different form than that of the transport equation, the time step, or pseudo-time step, in the reinitialization process is calculated somewhat differently. Shown in Eq. 2.77 below, the time step condition is a modification of that used by Owkes and Desjardins [59],

$$\Delta\tau = \min \left[h_{\min}, \frac{h_{\min}^2}{4(\varepsilon_1 + \varepsilon_2)}, \frac{h_{\min}}{3\max(|r_{x:b}|, |r_{y:b}|)} \right] \mathrm{CFL}_{\mathrm{reinit}} \tag{2.77}$$

where $h_{\min} = \min(\Delta x, \Delta y)$. The modification here lies in the addition of the $h_{\min}/3\max(|r_{x:b}|, |r_{y:b}|)$ term, which is used to incorporate uncertainty into the calculation of $\Delta\tau$. This term was determined by numerical experiments and is based on the idea that the continuous normal vector limits the 'speed' with which the level set can be reinitialized, and is a function of uncertainty. As the basis weight b increases, so too does the frequency of oscillation, as shown in Fig. 2.4. Too large of a pseudo-time step can cause instability and thus destroy the reinitialization process. This time step is necessarily updated at each iteration, as $\boldsymbol{r}_b$ is changing.

The amount of reinitialization can have a significant effect on accuracy of the level set [8, 49]. Too much reinitalization will excessively distort the level set, while too little will lead to mass conservation errors. A reasonable technique is to couple the amount of reinitialization to the physical time step used in the simulation and the interface motion [59]. With this approach, the total amount of pseudo-time which passes for each time iteration is $\tau_{\mathrm{step}} = F\max(|\boldsymbol{u} \cdot \boldsymbol{r}|)\Delta t$, where F is a constant. This total pseudo-time is taken in steps of $\Delta\tau$, as calculated in Eq. 2.77, for model stability.



(a) b = 5 (b) b = 15 (c) b = 25

Figure 2.4: Level set weights ψ_b (solid) and continuous normal vector weights $\boldsymbol{r}_{x:b}$ (dashed) for different orders of the basis functions b along a central line for a stochastic circle. Note the frequency of oscillation increases with basis order.

One can see that as the value of F is increased, the number of reinitialization steps per time step increases with it. As a result, more non-physical interface motion is taking place in an effort to conserve mass. As such, the value of F should be minimized as much as possible.

Summary

Numerical implementation of a stochastic multiphase solver requires a few major differences over traditional schemes utilized for deterministic models. First is the utilization of the conservative level set ψ as a smoothed Heaviside function for identifying the location of the interface and calculating a smoothed density and viscosity in space, which differs from the discontinuous jump set in typical schemes at $\psi = 0.5$. With this comes the coupled nature of density and pressure in the elliptic pressure Poisson equation needed for the pressure correction method of Navier-Stokes. Finally, to reinitialize the level set after each time step, a continous normal vector has been created which works nicely in a stochastic framework and reduces the need for a banding procedure that is expensive and difficult to conceptualize for a spatial grid with a distribution of stochastic interfaces.

Level Set Test Cases

In this section three common and well known test cases are used to assess the implementation of the level set transport and reinitialization equations with an imposed velocity field. The first case is a droplet moving in a channel flow with both a deterministic and stochastic velocity field. The other test cases include a deformation test and Zalesak's disk, both with modifications to add uncertainty.

Channel Flow

A channel flow test case is used to verify the transport scheme is conservative and to verify the time marching method. This case also tests the ability of the reinitialization procedure to maintain the sigmoid profile of ψ. For a deterministic simulation, the flow velocity is assumed to be known, and given the velocity is one-dimensional ($v = 0$), the transport equation reduces to

$$\frac{\partial \psi}{\partial t} = -u\frac{\partial \psi}{\partial x}, \tag{2.78}$$

of which there is the simple analytic solution $\psi(x, y, t) = \psi(x + ut, y, t = 0)$.

For a stochastic simulation, the velocity magnitude in the channel is allowed to vary, i.e. there is uncertainty about how fast the fluid is moving. Figure 2.5 illustrates the boundary and initial conditions for the channel test case.

Deterministic Channel Flow This basic case of deterministic channel flow is used to test the stability and performance of the model against the analytic solution. In this simulation, the velocity field was set as $u = 2.0$ and $v = 0.0$ with an initial starting position of $(x_o, y_o) = (0.5, 0.5)$ for the circle center, and a radius of $R = 0.3$. Figure 2.6 illustrates the solution after a time of $t = 1.0$ s.

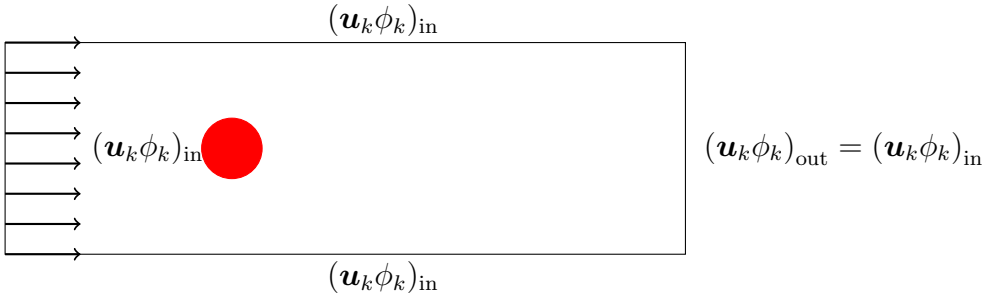


Figure 2.5: Schematic of a droplet moving within a channel flow, with uncertainty about the velocity magnitude, and outflow equal to that of inflow.

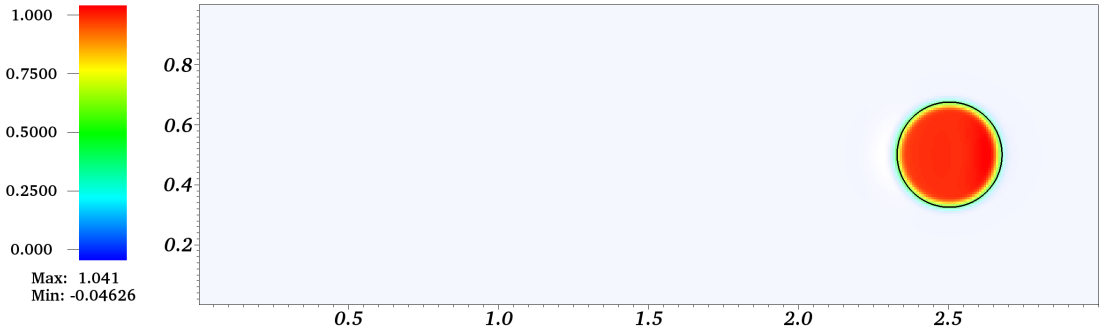


Figure 2.6: Deterministic channel flow at time = 1.0 s for a mesh grid of 375×125. The color bar represents the conservative level set value, ψ, and the black circle represents the analytic solution for the interface location ($\psi = 0.5$).

As shown in Fig. 2.6, the analytic solution for the gas-liquid interface location (shown as a black line) matches that of the numerical result. Additionally, the sigmoid profile of the level set has been maintained, with little deviation from the initial profile extending from $[0, 1]$.

Stochastic Channel Flow It is also useful to compare the results of a stochastic channel flow to the analytic solution. In this scenario, a uniform distribution of u on the interval $[1.75, 2.25]$ is used to describe the flow velocity, which is writen with the

Legendre basis functions as $u(x,\zeta) = 2.0\phi_0 + 0.25\phi_1 = 2.0 + 0.25\zeta$ for $-1 \leq \zeta \leq 1$. To capture this variability, the basis order was set to $N = 25$.

Solving the analytic solution using the lower and upper bounds of u allows for comparison of the solutions at $\zeta = -1$ and $\zeta = 1$, which are the two most outlying solutions of the infinite potential solutions. Looking at Fig. 4.6, comparison of position of the circle from the numerical and analytic solutions reveals the level set transport and reinitalization is accurate. The color shows the level set variance defined with

$$\sigma_\psi^2 = \sum_{k=1}^{N} \frac{\psi_k^2}{\langle \phi_k \phi_k \rangle}. \tag{2.79}$$

The variance quantifies the degree to which the level set value varies from average. A high degree of variance in the level set indicates regions where the location of the droplet is more uncertain.

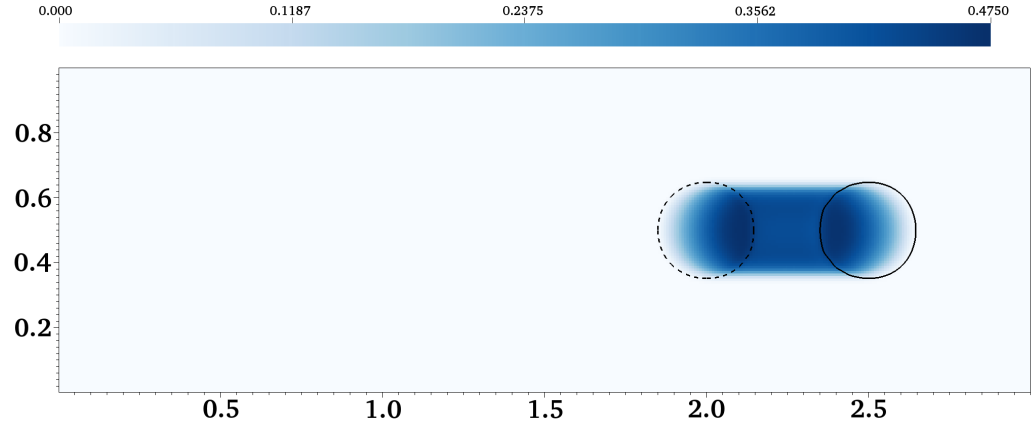


Figure 2.7: Position of the stochastic circle with $\zeta = -1$ (dashed) and $\zeta = +1$ (solid). Level set variance is represented by the color bar, while interface locations correspond to analytic solutions at $t = 1$s.

Perhaps more interesting is the probability of the droplet existing at any given location. This probability is calculated using a simple systematic sampling procedure

across ζ, where

$$P(\psi > 0.5) = \frac{1}{N_s} \sum_{i=1}^{N_s} H\left[\psi_k \phi_k\left(S_i\right) - 0.5\right], \tag{2.80}$$

N_s is the number of sample points used, and S_i is the sample ζ value calculated linearly from $[-1, 1]$. Figure 2.8 illustrates the probability of being within the droplet for any given location. Of course this is the probability of any part of the droplet being at a discrete location. The maximum value of $P(\psi > 0.5) = 0.6$ agrees with the analytic solution.

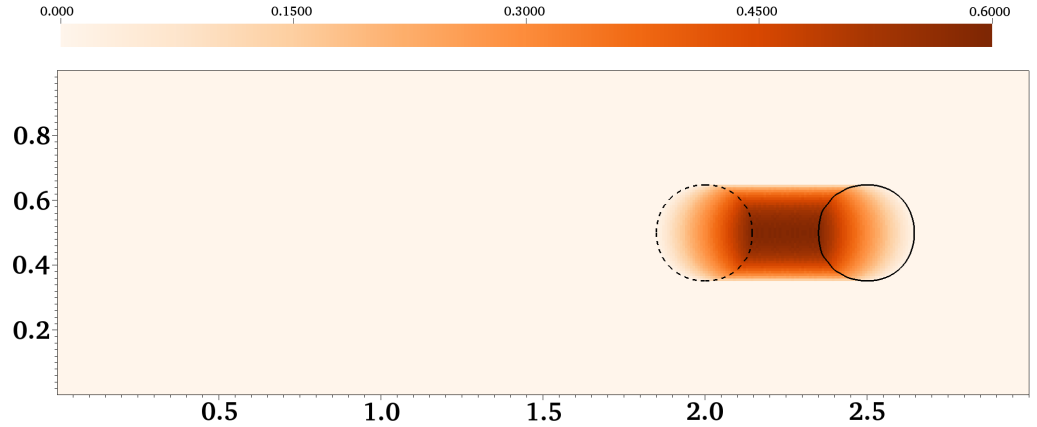


Figure 2.8: The color bar represents the probability of being within the droplet, while dashed and solid lines indicate the interface of the slowest and fastest solutions, respectively.

Deformation Field

Assessment of the conservative ability of the level transport in a more complicated velocity field for both the deterministic and stochastic frameworks is done with the deformation test case, as first presented by Bell et al. [2] and used frequently by others, e.g., [15, 20, 59, 60]. In this test, a circular droplet of radius $R = 0.15$ in a unit domain centered at initial location $(x_o, y_o) = (0.5, 0.75)$ is stretched and then

unstretched with the velocity field

$$\begin{aligned} u &= -2\sin^2(\pi x)\sin(\pi y)\cos(\pi y)\cos(\pi t/8) \\ v &= +2\sin^2(\pi y)\sin(\pi x)\cos(\pi x)\cos(\pi t/8) \end{aligned} . \tag{2.81}$$

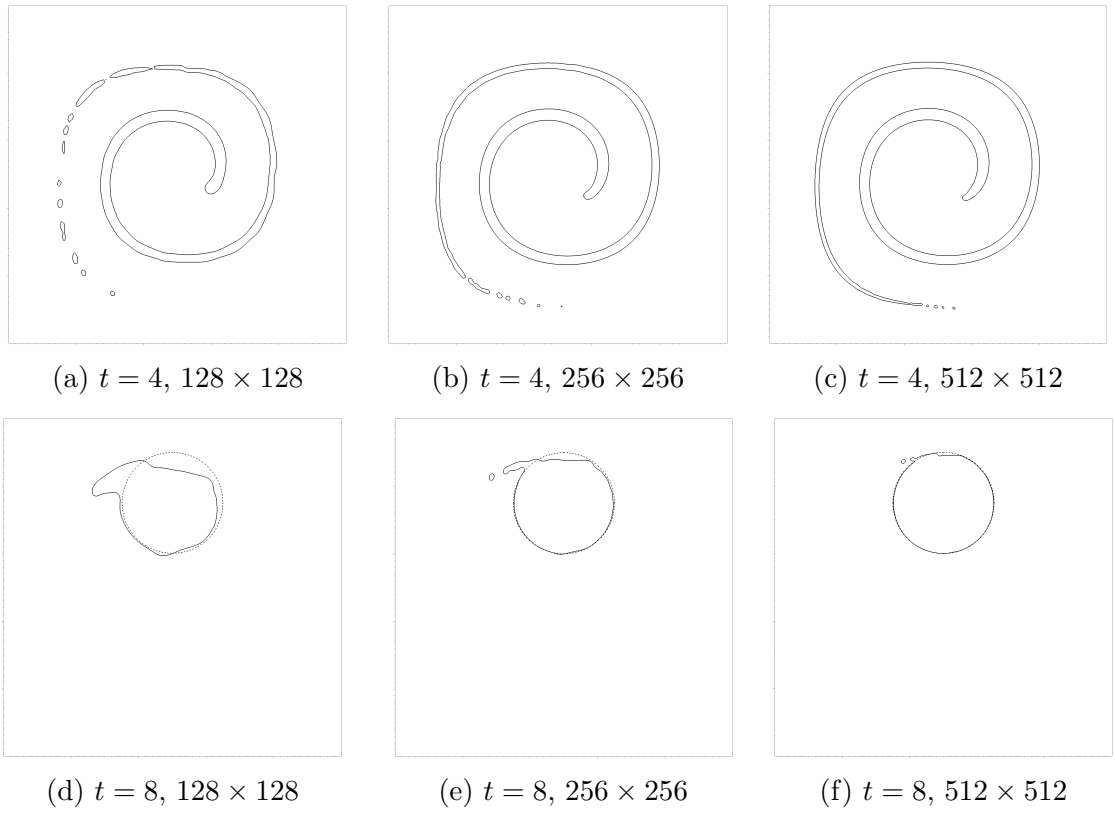
This velocity field stretches and un-stretches the two-dimensional disk to a maximum at $t = 4$, and then reverses such that the droplet moves back to its original position at $t = 8$.

Deterministic Deformation Case In order to test the level set transport scheme and reinitialization process, including the proposed continuous normal vectors, we begin by performing a deterministic simulation of the deformation field. These results can then be compared to other published results to provide a baseline with which to assess the stochastic results.

Perhaps the most interesting aspect to the deterministic deformation test case is in displaying the validity of the continuous normal vectors, $\boldsymbol{r}$, which utilize a relaxed magnitude condition compared to that described by Desjardins et al. [15]. Looking at Fig. 4.7, we see convergence of the final time step to the initial condition as the mesh is refined. The interface location at the final time is comparable to the results found in other published research [15, 20, 59].

Utilization of this technique does not degrade the conservative nature of the level set transport scheme, as evidenced by Fig. 2.10, which illustrates the area ratio $\mathcal{L}(t)/\mathcal{L}(t_o)$, where $\mathcal{L}(t) = \int H(\psi(t) - 0.5)dA$ is the area of liquid within the domain. As shown in Fig. 2.10, the area within the interface is converging as the mesh is refined. The area within the interface boundary has been calculated using a subgrid method as used by Herrmann [29].

As noted by Herrmann, even if perfect transport were possible, the solution of



(a) $t = 4$, 128×128 (b) $t = 4$, 256×256 (c) $t = 4$, 512×512

(d) $t = 8$, 128×128 (e) $t = 8$, 256×256 (f) $t = 8$, 512×512

Figure 2.9: Illustration of the $\psi = 0.5$ interface for the deformation test case at maximum stretching (top) and final condition (bottom) for mesh grid sizes of 128×128, 256×256, and 512×512. The reinitialization constant is set to $F = 2.0$. At the final time the interface is compared to the starting location, displayed as a dotted line.

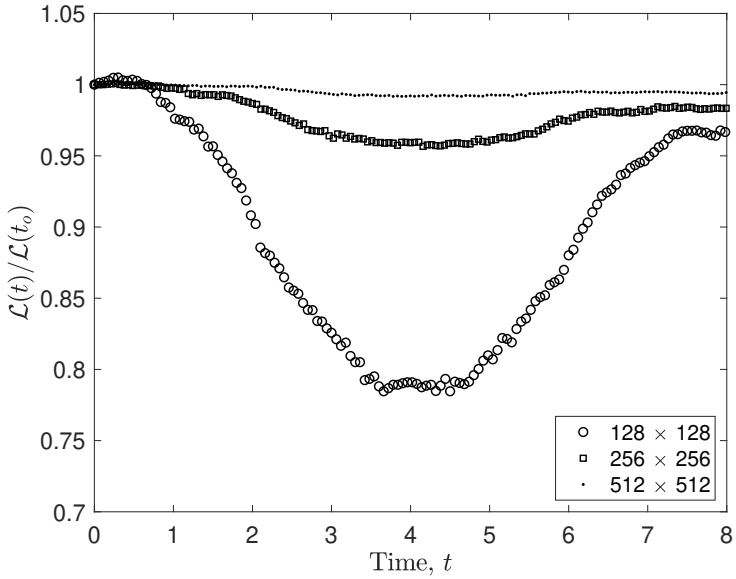
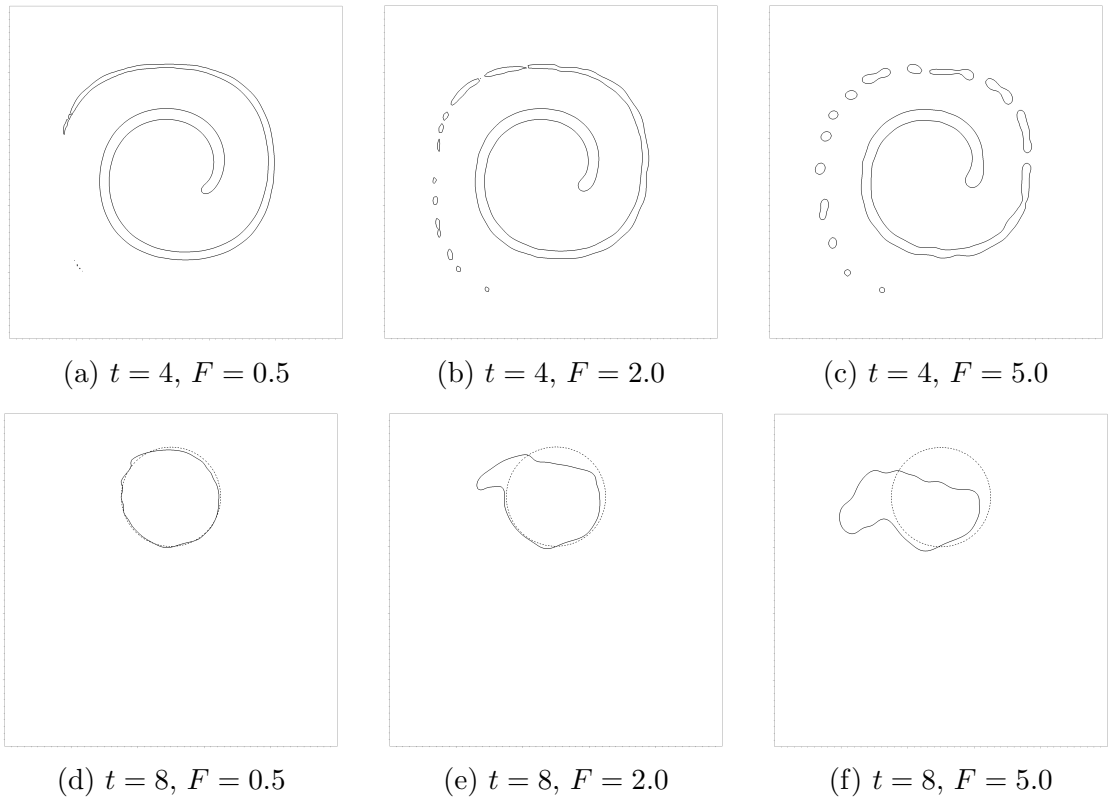


Figure 2.10: Ratio of liquid area $\mathcal{L}(t)$ to the initial liquid area $\mathcal{L}(t_o)$ over time for several mesh sizes of the deterministic deformation test case.

a 128^2 grid would lose significant area because the trailing edge thickness falls below the grid resolution. This is seen in the maximum % area loss of the initial shape, where for 128^2, 256^2, and 512^2 grids the area loss is 21.6%, 4.42%, and 0.899%, respectively. Additionally, the value of F used to calculate the reinitialization time has an impact on the results of simulation outputs, as illustrated in Fig. 2.11 and discussed in Section 2. Smaller values of F result in fewer reinitialization steps taken during the simulation and more mass loss but less artificial interface motion. Larger values of F conserves mass better but causes non-physical interface motions.

Stochastic Deformation Case In this example, consider the same deterministic velocity field presented in the previous section. The difference of this scenario lies in the added assumption that some uncertainty exists about the initial position of the interface. Two separate uniform distributions will be used to discuss the impact



(a) $t = 4$, $F = 0.5$ (b) $t = 4$, $F = 2.0$ (c) $t = 4$, $F = 5.0$

(d) $t = 8$, $F = 0.5$ (e) $t = 8$, $F = 2.0$ (f) $t = 8$, $F = 5.0$

Figure 2.11: Illustration of the $\psi = 0.5$ interface for the deformation test case at maximum stretching (top) and final condition (bottom) for a mesh grid of 128×128 over a range of F values. At the final time the interface is compared to the starting location, displayed as a dotted line.

of locational uncertainty and illustrate how this uncertainty exists in the simulation. An extreme example is used to analyze the ability of the PC method to converge to a difficult solution, compare analytic probabilities to numerical probabilities, and more clearly illustrate the solution dynamics in play. A more realistic example is used to run a full simulation which avoids any interaction with the boundary walls which is not typical of this test case.

Consider first the extreme example of uncertainty about the starting position of the interface, shown in Fig. 2.12. Uncertainty is only present about the x location of the center of the droplet, x_o. For a uniform distribution, a linear relationship defines x_o about circle's center, where $x_o(\zeta) = 0.5+0.3\zeta$. The result is a case where the center of the initial droplet exists somewhere between $(0.2, 0.75) \leq (x_o, y_o) \leq (0.8, 0.75)$, each having the same probability. By inspection, this is a case where exactly half of the possible starting locations place the grid point $(0.5, 0.75)$ inside the droplet and half of the cases place the grid point outside the droplet.

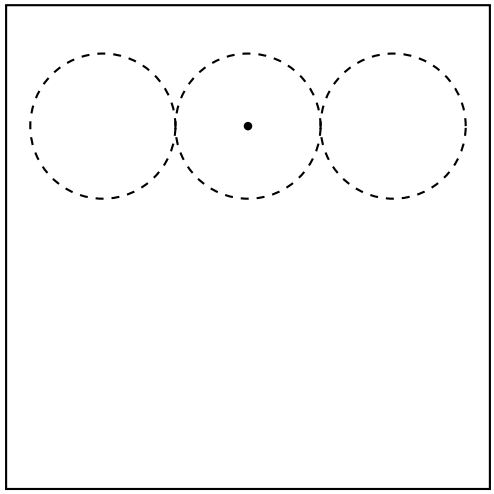
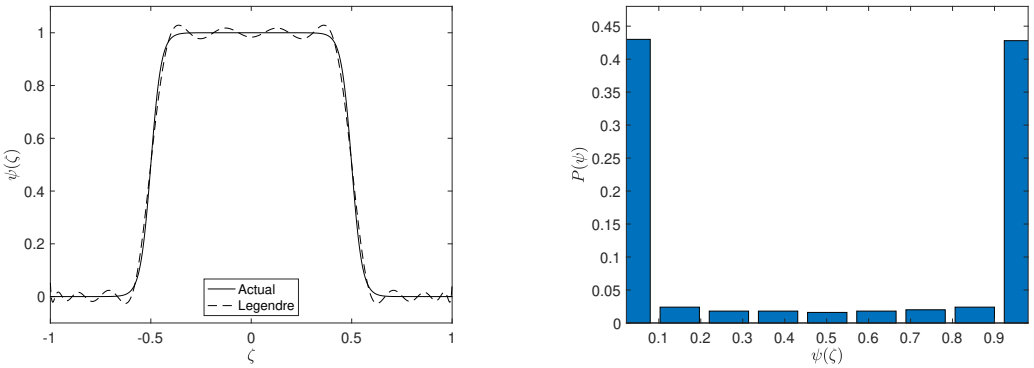


Figure 2.12: Schematic of uncertainty about the starting location. Dashed lines indicate equally probable interface locations for the initial condition. The black point is examined about the uncertainty dimension, ζ.

Studied at a grid size of 128×128, Fig. 2.13 illustrates the value of $\psi(\zeta)$ at the central point $(0.5, 0.75)$. The dashed line in Fig. 2.13a, representing the projected solution with an order of $N = 25$, shows reasonable convergence to the analytic result drawn with a solid line. The sigmoid profile of Fig. 2.13a displays the locations of the interface along the $x-$ axis of the central circle shown in Fig. 2.12. Figure 2.13b displays 10 probability slices $P(\psi)$, or the probability of the level set existing within values shown. Analytically the probability of being within the droplet is $P(\psi > 0.5) = 0.5$ and the probability of being outside the droplet is $P(\psi < 0.5) = 0.5$.



(a) Level set profile.

(b) Probability slices.

Figure 2.13: Profile of the sigmoid level set, $\psi(\zeta)$, at $(x, y) = (0.5, 0.75)$ for the initial condition about the two-dimensional deformation case. The value of the level set $\psi(\zeta)$ is shown in Fig. 2.13a, where $N = 25$ orthogonal basis functions are used. The representation of the function with Legendre polynomials is compared with the exact profile. Figure 2.13b shows a probability distribution of different values of $\psi(\zeta)$

Next, we implement a more realistic example of uncertainty. In this case, a smaller amount of uncertainty about the starting location in the x dimension is assumed so as not to set the starting locations too far outside the vortex field and provoke wall interactions. Looking at the implication of an uncertain starting location

about the deformation test case, a similar linear relationship is imposed, where

$$x_o(\zeta) = 0.5\phi_0 + 0.05\phi_1 = 0.5 + 0.05\zeta, \tag{2.82}$$

where the first two Legendre polynomials, $\phi_0 = 1$ and $\phi_1 = \zeta$ are used. Given the nature of the deformation test case, which stretches the interface boundary in a vortex field to a maximum then returns it to its original form, the expectation is that the initial variance profile will also be the ending variance profile. This tests the ability of the scheme to maintain the stochastic level set undergoing large deformations.

The outer bounds of possible interface locations and the variance about the level set ψ at 3 time instances is shown in Fig. 2.14. Comparison of these figures shows convergence about the final time for both the interface location as well as the variance profile with mesh refinement. This simulation was run using 9 orthogonal basis functions for reasonable convergence to the level set profile.

Looking at the affect the number of basis functions used has on the solution of the deformation case, we see in Fig. 2.15 that with few basis functions, it isn't possible to converge the initial condition to perfect circles. As the number of basis functions are increased, the initial condition, where an infinite number of circles are present from $0.4 \leq x_o \leq 0.5$, becomes more resolved. Additionally, more mass is lost from outlying solutions as small droplets are not captured by the function $\psi_k\phi_k$ with fewer degrees of freedom.

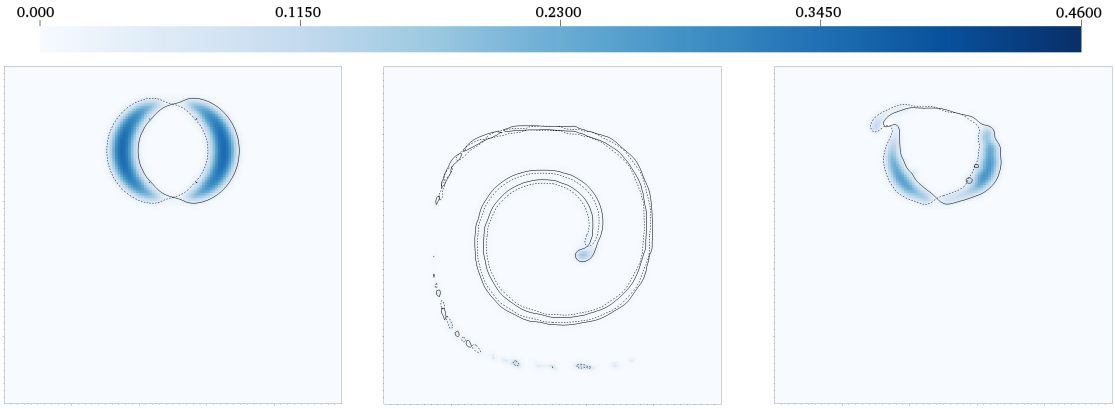
Zalesak's Disk

This test case assesses the ability of an interface tracking or capturing scheme to move sharp corners, and was originally presented by Zalesak in 1979 [85]. The disk involved is a circle of radius $R = 0.2$ centered at $(x_o, y_o) = (0, 0.25)$ in a unit domain of range $[-0.5, 0.5]$. A notch is placed in the center, with a width of $W_{\text{notch}} = 0.2R$ and

(a) 128×128, $t = 0$ (b) 128×128, $t = 4$ (c) 128×128, $t = 8$

(d) 256×256, $t = 0$ (e) 256×256, $t = 4$ (f) 256×256, $t = 8$

Figure 2.14: Variance about level set scalar, ψ, for three different time slices of the uncertain deformation case for 128×128 (top) and 256×256 (bottom) grids. A total of $N = 9$ degrees of freedom are used. Dashed and solid lines indicate the interface for the solution bounds.



(a) 128×128, $t = 0$, $N = 3$ (b) 128×128, $t = 4$, $N = 3$ (c) 128×128, $t = 8$, $N = 3$

(d) 128×128, $t = 0$, $N = 6$ (e) 128×128, $t = 4$, $N = 6$ (f) 128×128, $t = 8$, $N = 6$

(g) 128×128, $t = 0$, $N = 9$ (h) 128×128, $t = 4$, $N = 9$ (i) 128×128, $t = 8$, $N = 9$

Figure 2.15: Variance about level set scalar, ψ, for three different time slices of the uncertain deformation case with a mesh grid of 128×128 over a range of basis functions. Dashed and solid lines indicate the interface boundary for the outer most possible initial positions.

a height of $H_{\text{notch}} = 0.65R$. Four distinct corners are present, as shown in Fig. 2.16 below, where two sharp inside and outside corners exist. A simple velocity field is applied to the disk, where

$$\begin{aligned} u &= -C\pi y \qquad \text{and} \\ v &= +C\pi x. \end{aligned} \tag{2.83}$$

Results are examined after a full rotation occurs, when the disk returns to its starting orientation. The constant C has been added to allow uncertainty to be added to the velocity field. For the deterministic case $C = 2$. The stochastic test adds a variable velocity magnitude by varying $C(\zeta) = C_k\phi_k(\zeta)$, resulting in a divergence-free uncertain velocity field $\boldsymbol{u}_k\phi_k$.

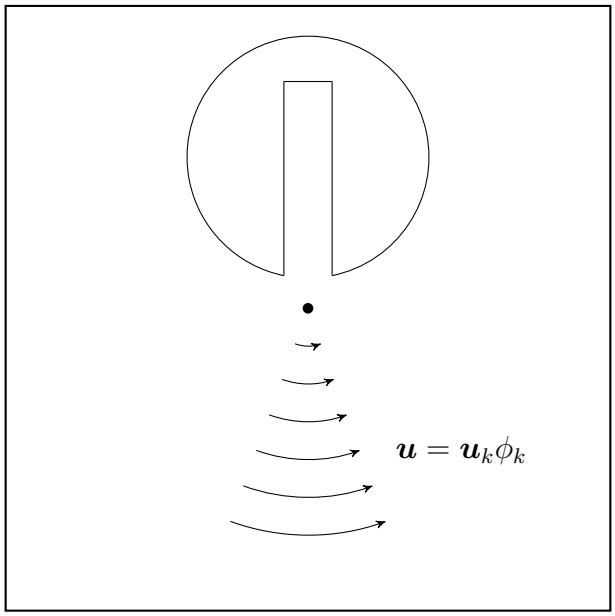
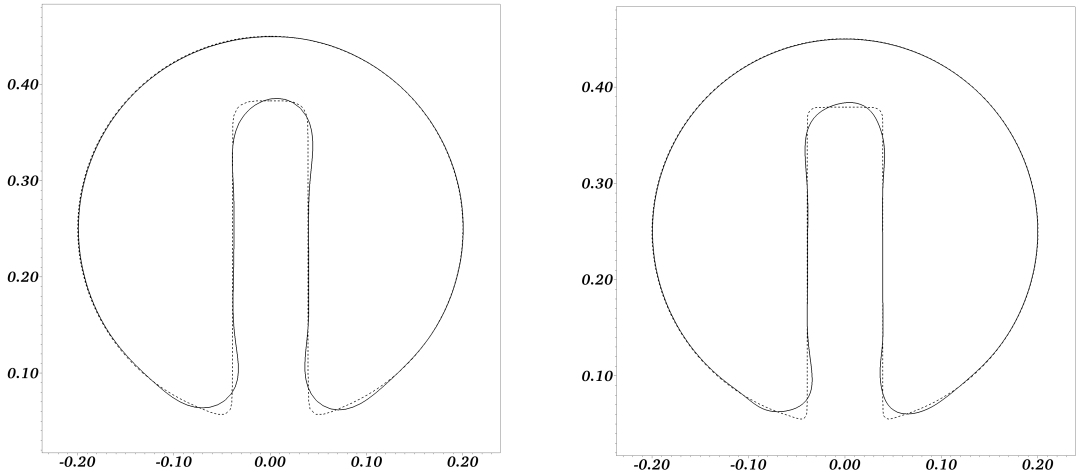


Figure 2.16: Schematic of Zalesak's disk, which goes through a single solid body rotation about the center point.

Deterministic Zalesak's Disk The deterministic disk is a common way of testing the abilities of an interface method by focusing on convergence as mesh is refined. Additionally, it is useful to compare these results to previously published results for comparison. Figure 2.17 shows convergence as the mesh is refined, where deviations from the initial solution after a full body rotation are much smaller with the finer mesh.

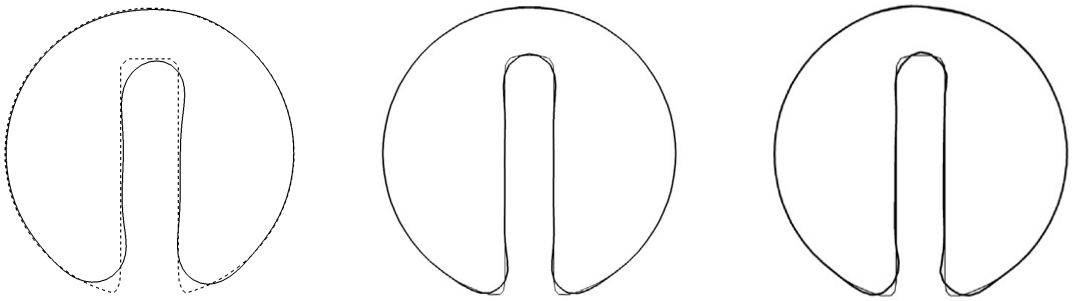


(a) 128×128 (b) 256×256

Figure 2.17: Interface location of the deterministic Zalesak's disk case after a single rotation for two mesh resolutions. The exact result is depicted with a dotted line and the simulation results are shown with solid lines.

Additionally, these results are similar to those found in published works [15, 29, 59]. Figure 2.18 compares the results from our deterministic Zalesak's disk to published results, with only slight deviations at a relatively coarse mesh.

Zalesak's Disk in an Stochastic Velocity Field Uncertainty about the value C in Eq. 2.83 manifests as variability in the magnitude of the velocity. Similar to the uncertain velocity magnitude of the channel flow problem, having started with a deterministic initial position, after existing in the velocity field for a set amount of



(a) Current framework (b) Owkes and Desjardins [59] (c) Desjardins et al. [15]

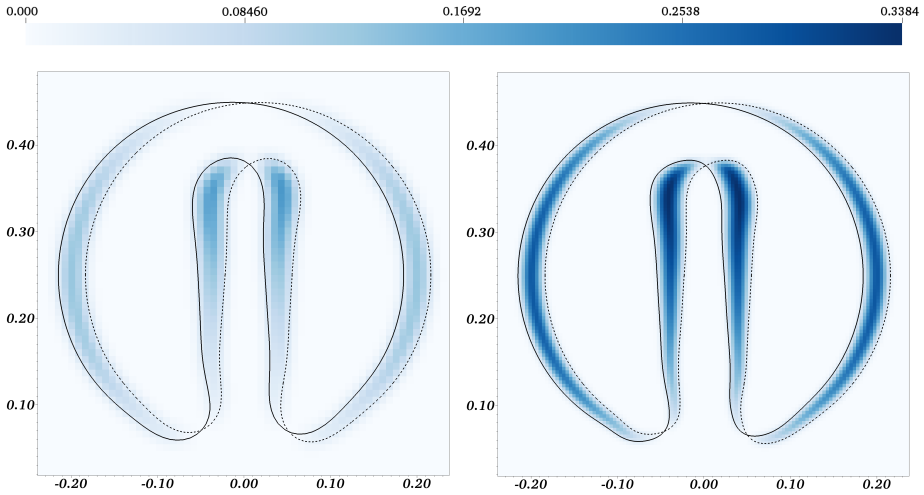
Figure 2.18: Comparison of deterministic Zalesak's disk to other published results for a 100×100 grid. The exact result is depicted with a dotted line and the simulation results are shown with solid lines.

time, there is uncertainty about where the disk is located. The expectation is that the disk exists between two outlying possibilities, where each final location is equally possible. Additionally, it would be appropriate to see the corners hold their form (converging to sharper corners for finer meshes) in a fashion similar to that seen in the deterministic case for each of the possible solutions.

As seen in Fig. 2.19, the shape of the interface is similar to the deterministic case for each of the presented mesh resolutions. Again, we see that as the mesh is refined, the corners are better preserved during the rotation for both outlying solutions, although for a stochastic solution, more basis functions are also needed to better preserve the corners. Additionally, the variance profile is sharper for the refined mesh, with the greatest magnitude existing around the inside of the slot.

Multiphase Test Cases

Having determined the stochastic level set is performing well and converging in the channel flow, deformation, and Zalesak's disk cases, the stochastic level set and Navier-Stokes equations are coupled to create a stochastic multiphase flow solver.



(a) 128×128 (b) 256×256

Figure 2.19: Variance of the level set for Zalesak's disk in an uncertain velocity field after a single rotation for two separate mesh grid sizes. The interface of the outlying solutions for the fastest and slowest velocity fields are shown by the solid and dotted lines, respectively. Simulation was run with a basis order of $N = 10$.

This solver was applied to the cases of a two-dimensional oscillating droplet and an atomizing jet.

Deterministic Oscillating Droplet

The phenomenon of a droplet oscillating due to surface tension force was first noted by Lord Rayleigh [46], and has been used widely as a method for both validating the abilities of a multiphase flow solver as well as for determining the surface tension coefficient of various fluids. The benefit lies in the analytical result for the frequency of oscillation, where the results of the numerical simulation can be directly compared for accuracy.

This case is frequently used; Olsson et. al [55] tested their conservative level set method with the oscillating droplet, among others [24, 29]. Analytical solutions to the oscillating droplet for both large and small deviations are provided by Fyfe et. al [23] and Shin et. al [71], including the decay of the amplitude due to viscosity. For small amplitude oscillations, the frequency ω_n for mode number n is given by

$$\omega_n^2 = \frac{\gamma(n^3 - n)}{\rho R^3}, \tag{2.84}$$

with surface tension coefficient γ, density ρ, and unperturbed radius R. The surface of the droplet is given for amplitude η in polar coordinates by

$$r = R + \eta \cos(n\theta). \tag{2.85}$$

Extending Rayleigh's idea into a situation where two phases exist, the frequency of oscillation is then

$$\omega_n^2 = \frac{\gamma(n^3 - n)}{(\rho_i + \rho_e)R^3}, \tag{2.86}$$

where ρ_i is the fluid density inside the droplet and ρ_e is the external fluid density (outside the droplet).

Following the procedure used by Fyfe et al. [23], for a two-dimensional system and mode two ($n = 2$) oscillation, the oscillation period τ is

$$\tau = 2\pi\sqrt{\frac{(\rho_i + \rho_e)R^3}{6\gamma}}. \tag{2.87}$$

To test the multiUQ framework with the oscillating droplet, the initial condition is set to an ellipse defined with

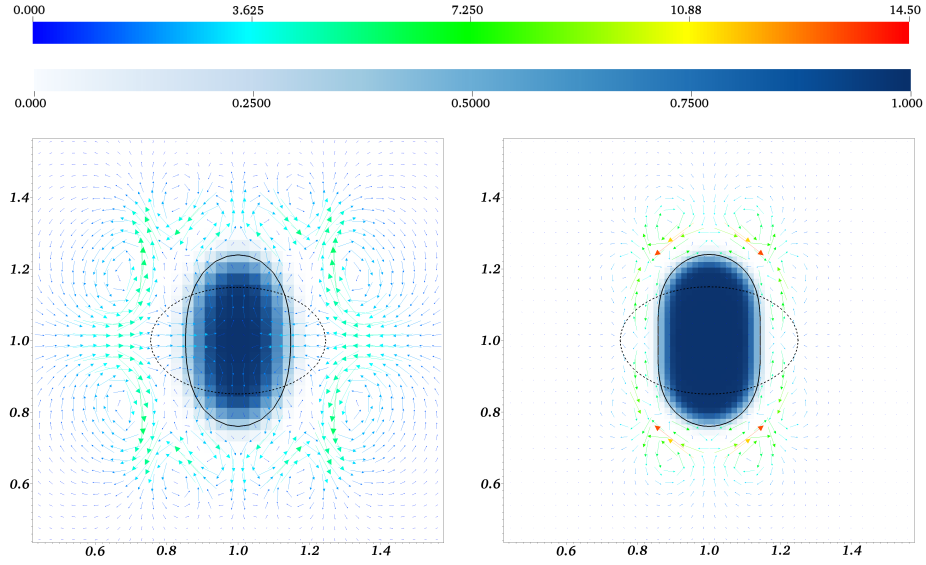
$$\frac{x^2}{A^2} + \frac{y^2}{B^2} = 1, \tag{2.88}$$

where the semimajor and semiminor axes are $A = 0.25$ cm and $B = 0.15$ cm, respectively. For an ellipse the unperturbed radius is found with $R = \sqrt{ab}$. The fluid properties are set to $\gamma = 72.8$ $^{\text{dynes}}/_{\text{cm}}$, $\rho_i = 1.0$ $^{\text{g}}/_{\text{cc}}$, $\rho_e = 0.001$ $^{\text{g}}/_{\text{cc}}$, $\nu_i = 0.01$ $^{\text{cm}^2}/_{\text{s}}$ and $\nu_e = 0.005$ $^{\text{cm}^2}/_{\text{s}}$, which is essentially a water and air system.

Undergoing a full period of oscillation results in the elliptic droplet, beginning at its initial state, moving inward along the semimajor axis and flexing upward along the semiminor axis, then eventually moving back to its nearly initial state, with some degree of decay about the radius of the semimajor axis. Figure 2.20 shows the droplet shape halfway through a period of oscillation.

Tracking the oscillation was done by calculating the kinetic energy. Using a two-dimensional space for volume and calculating mass inside it using density, the global kinetic energy can be found with

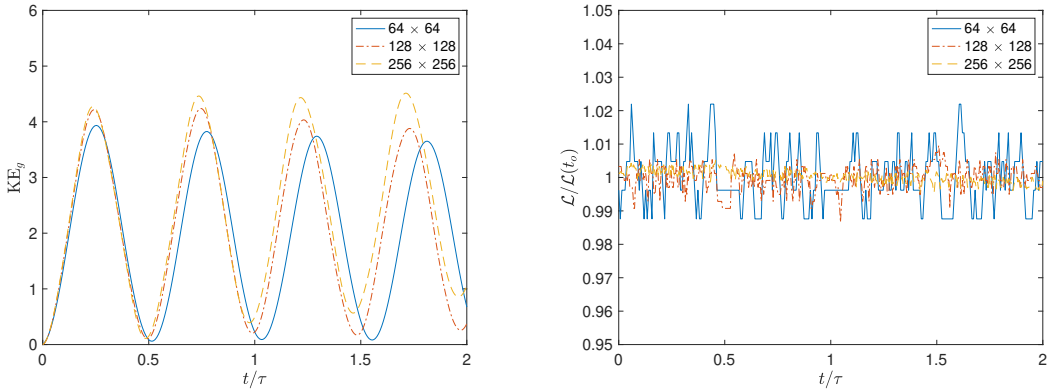
$$\text{KE}_g = \frac{1}{2}\int \rho(\boldsymbol{u}\cdot\boldsymbol{u})\ \text{dA}, \tag{2.89}$$



(a) 64×64 (b) 128×128

Figure 2.20: Deterministic oscillating droplet at time $t = 0.0138$ s, which is half of the period of oscillation. Solid line indicates the interface boundary, dashed line indicates interface boundary of initial position. Top colorbar represents velocity magnitude, while the bottom colorbar shows density ρ.

where velocity values must be interpolated to the cell center where density is held. Looking at the global kinetic energy shown in Fig. 2.21, the period of oscillation corresponds with the expected analytic result. Also shown is the more rapid decay of kinetic energy for coarser mesh sizes, as well as an initially growing kinetic energy peak with the finer 256×256 mesh.



(a) Global kinetic energy (b) Droplet area ratio

Figure 2.21: Results of a deterministic oscillating droplet at 3 different mesh sizes. At left is the global kinetic energy as a function of time. At right is the area ratio of the droplet over the course of the simulations.

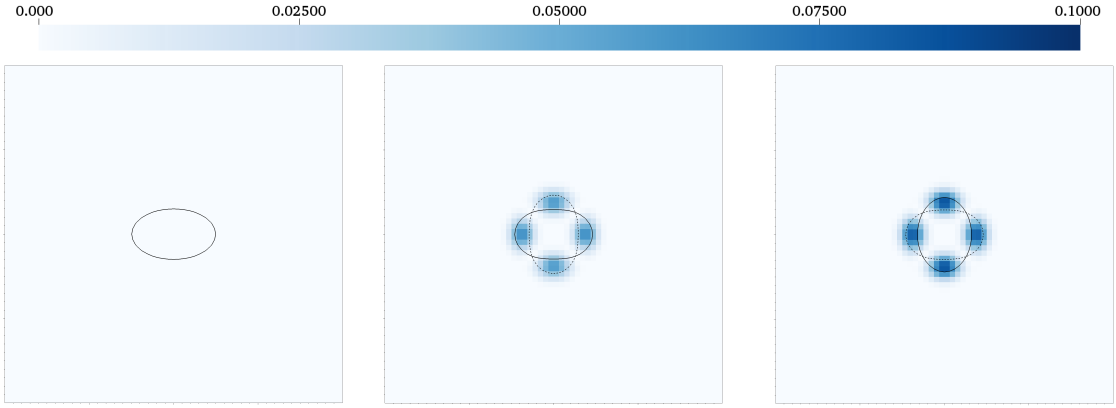
Stochastic Oscillating Droplet - Uncertain Surface Tension Coefficient

Assuming uncertainty about the surface tension coefficient, it is again useful to impose that uncertainty with a uniform distribution, where the probability is equal for each possible surface tension. As a function of uncertainty, the surface tension is defined as

$$\gamma(\zeta) = \gamma_0\phi_0 + \gamma_1\phi_1 = \gamma_0 + \gamma_1\zeta = 72.8 + 36.4\zeta, \tag{2.90}$$

where a deviation of $\pm 50\%$ from the average is used to assess the ability of the solver with large amounts of uncertainty.

Figure 2.22, depicts the extreme, or bounding, solutions for a UQ oscillating droplet, with the maximum and minimum surface tension force shown for 3 different time steps. The initial condition is set such that the position is known. As time passes, the surface tension sets the fluids into motion, at different rates for the infinite number of solutions that exist. The initial condition is shown, as well as times where the solution of the largest and smallest coefficient cases have undergone a full oscillation.



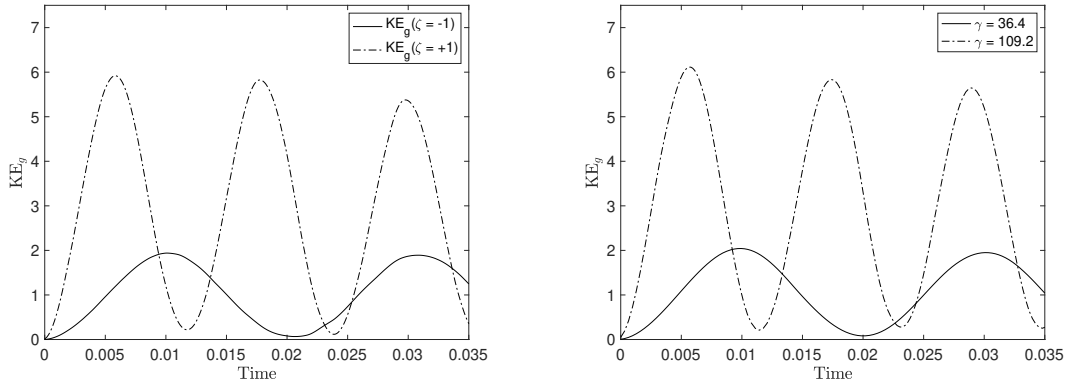
(a) 64×64, $t = 0.0000$ s (b) 64×64, $t = 0.0226$ s (c) 64×64, $t = 0.0391$ s

Figure 2.22: Three different time slices of an oscillating droplet with uncertainty about the surface tension coefficient for a mesh grid of 64×64. Dashed and solid lines indicate cases with the smallest and largest surface tension coefficients, respectively. The color map indicates the variance of the level set function.

Figure 2.23 displays the global kinetic energy values for the two extreme solutions of the oscillating droplet case. For comparison, the frequency of oscillation for these solutions found with a single run of the stochastic solver is presented with the results of two separate deterministic runs for the bounding solutions. The results are very similar.

Deterministic Jet

The previous example had a simple interface shape and was used to validate the methodology. In this test, the method is applied to the more challenging problem



(a) Stochastic solution (b) Deterministic solutions

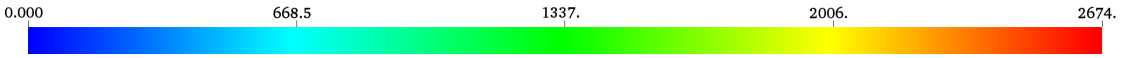
Figure 2.23: Global kinetic energy comparison of the result of a stochastic simulation to two deterministic bounding solutions for a 64×64 grid. The stochastic solution was calculated with a basis order $N = 10$.

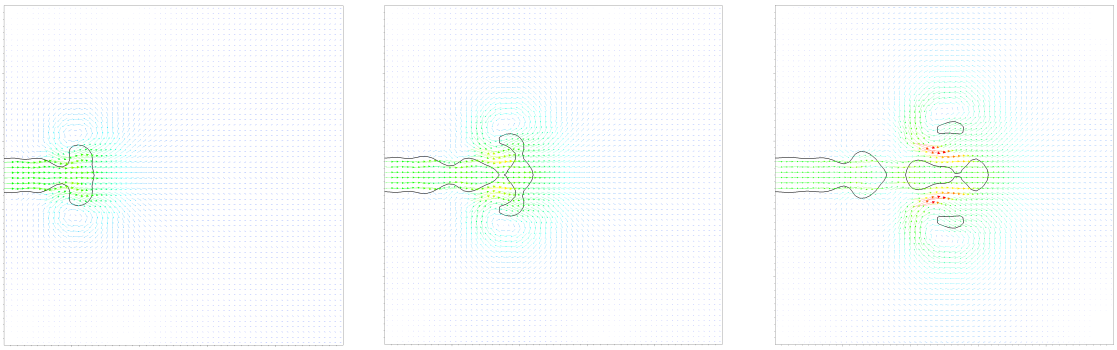
of an atomizing jet in two dimensions. Starting with a deterministic case, a fluid of density $\rho_i = 1.0$ $^{\mathrm{g}}/_{\mathrm{cm}^3}$ and viscosity $\nu_i = 0.01$ $^{\mathrm{cm}^2}/_{\mathrm{s}}$ is injected into a fluid with density $\rho_e = 0.001$ $^{\mathrm{g}}/_{\mathrm{cm}^3}$ and viscosity $\nu_e = 0.00018$ $^{\mathrm{cm}^2}/_{\mathrm{s}}$ at a rate of 1000 $^{\mathrm{cm}}/_{\mathrm{s}}$. The surface tension force is $\gamma = 72.8$ $^{\mathrm{g}}/_{\mathrm{s}^2}$.

Boundary conditions are no-slip at the top and bottom walls, no-slip on the left wall aside from the incoming jet, and an outlet on the right wall that maintains mass conservation so that mass in equals mass out. The initial velocity field inside the domain is zero. Figure 4.9 displays the result of the atomizing jet for two separate time steps for two different mesh sizes. By comparison, it can be seen that more breakup occurs for the finer mesh.

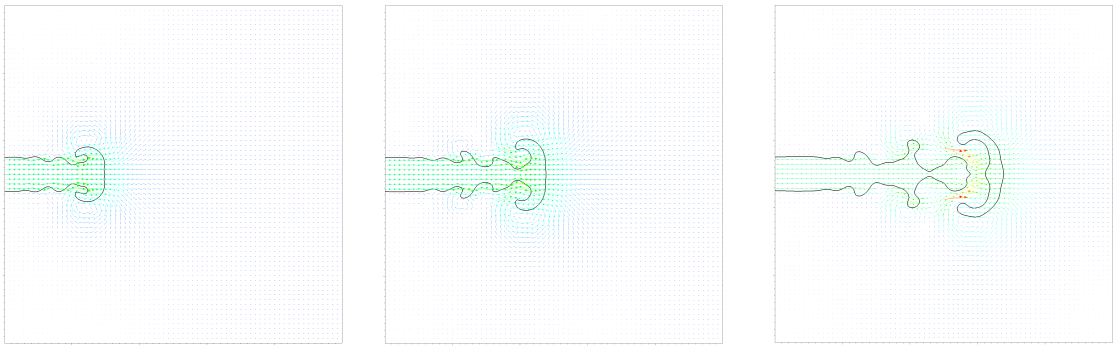
Stochastic Jet - Uncertain Surface Tension Coefficient

For the case of the stochastic jet, the fluid parameters are the same as for the deterministic case with the exception of added uncertainty about the surface tension coefficient, set to the uniform distribution as shown in Eq. 2.90, with a basis order of

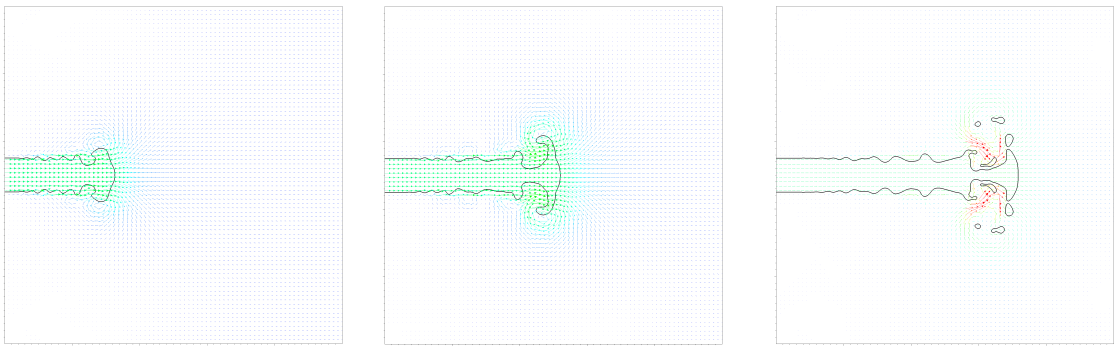




(a) 64×64, $t = 0.0003$ s (b) 64×64, $t = 0.0005$ s (c) 64×64, $t = 0.0007$ s



(d) 128×128, $t = 0.0003$ s (e) 128×128, $t = 0.0005$ s (f) 128×128, $t = 0.0007$ s



(g) 256×256, $t = 0.0003$ s (h) 256×256, $t = 0.0005$ s (i) 256×256, $t = 0.0007$ s

Figure 2.24: Results of a deterministic atomizing jet on 64×64 (top), 128×128 (middle), and 256×256 (bottom) mesh grids for three separate time steps. The shape profile at early time is more pronounced and more droplet breakup occurs later at the finer mesh (bottom).

$N = 5$.

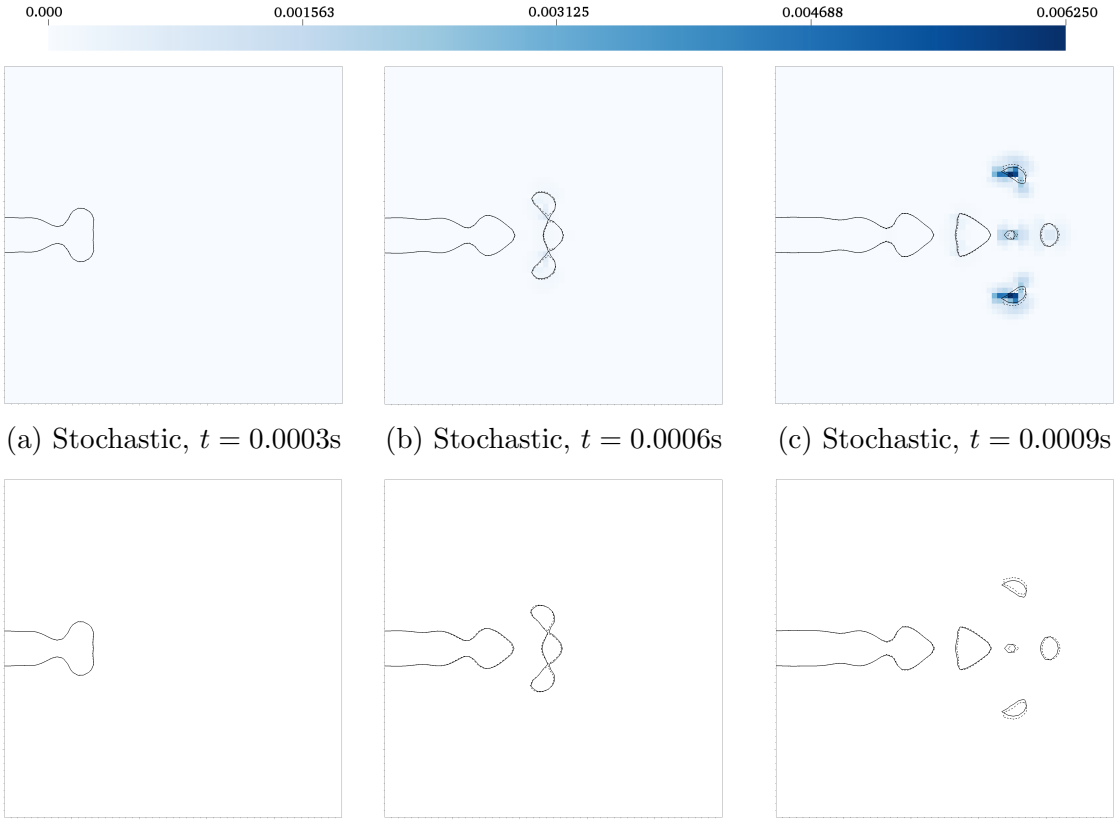
The stochastic solutions seen in Fig. 2.25 bound the deterministic solution shown in Fig. 4.9, while the variance profile shows regions where the value of the level set ψ is uncertain. Since the level set is situated such that $\psi > 0.5$ is within the droplet, variability in ψ indicates regions where the droplet location is uncertain. Comparison of deterministic simulations of bounding solutions of the stochastic model are also shown in Fig. 2.25, which presents stochastic results at $\zeta = -1$ and $+1$ that are nearly identical to the two deterministic simulations.

As seen by both the difference in the deterministic simulations as well as the stochastic solution, the affect of variability in the surface tension coefficient is upon the speed with which breakup occurs. Figure 2.25b illustrates droplet breakup occurring sooner for the lower surface tension coefficient case, with variation in the level set beginning to grow as a result. Refining the mesh, as shown in Fig. 2.26, we see this trend is more pronounced with the ability to resolve smaller droplets.

Stochastic Jet - Uncertain Incoming Velocity

To demonstrate a more difficult case, uncertainty was also applied to the incoming jet velocity, which more dramatically impacts the phase interface location and jet breakup. In this case, fluid properties for the incoming fluid are the same as for the deterministic jet case. The external fluid density is set to $\rho_e = 0.8$ $^{\mathrm{g}}/_{\mathrm{cm}^3}$, while the external fluid viscosity is $\nu_e = 0.18$ $^{\mathrm{cm}^2}/_{\mathrm{s}}$. Again, the incoming velocity field is assumed to have a uniform distribution, where $u_{\mathrm{in}}(\zeta) = 1000 + 500\zeta$.

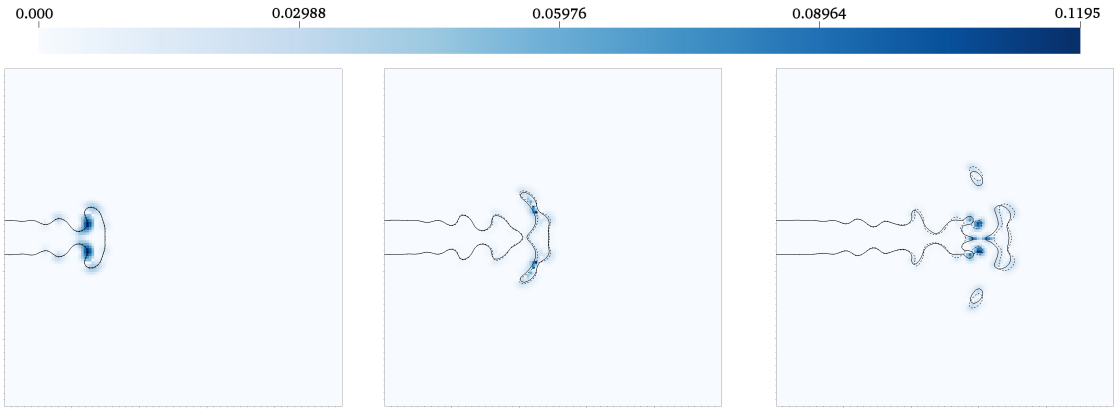
Figure 2.27 illustrates the ability of the method to resolve solutions that are vastly different. The difference is due to large variability in the incoming velocity where the largest velocity is 3 times greater than the lowest. To test the results, two deterministic simulations were performed which show very similar interface locations



(a) Stochastic, $t = 0.0003$s (b) Stochastic, $t = 0.0006$s (c) Stochastic, $t = 0.0009$s

(d) Deterministic, $t = 0.0003$s (e) Deterministic, $t = 0.0006$s (f) Deterministic, $t = 0.0009$s

Figure 2.25: Results of a stochastic atomizing jet on a 64×64 grid (top) using a basis order of $N = 5$ compared to bounding solutions of two deterministic runs (bottom). The dashed line indicates the interface boundary for the solution with the smallest surface tension coefficient (i.e. $\gamma(\zeta = -1) = 36.4$), while the solid line indicates solution with the largest surface tension coefficient (i.e. $\gamma(\zeta = +1) = 109.2$). The color bar represents variance about the level set scalar.



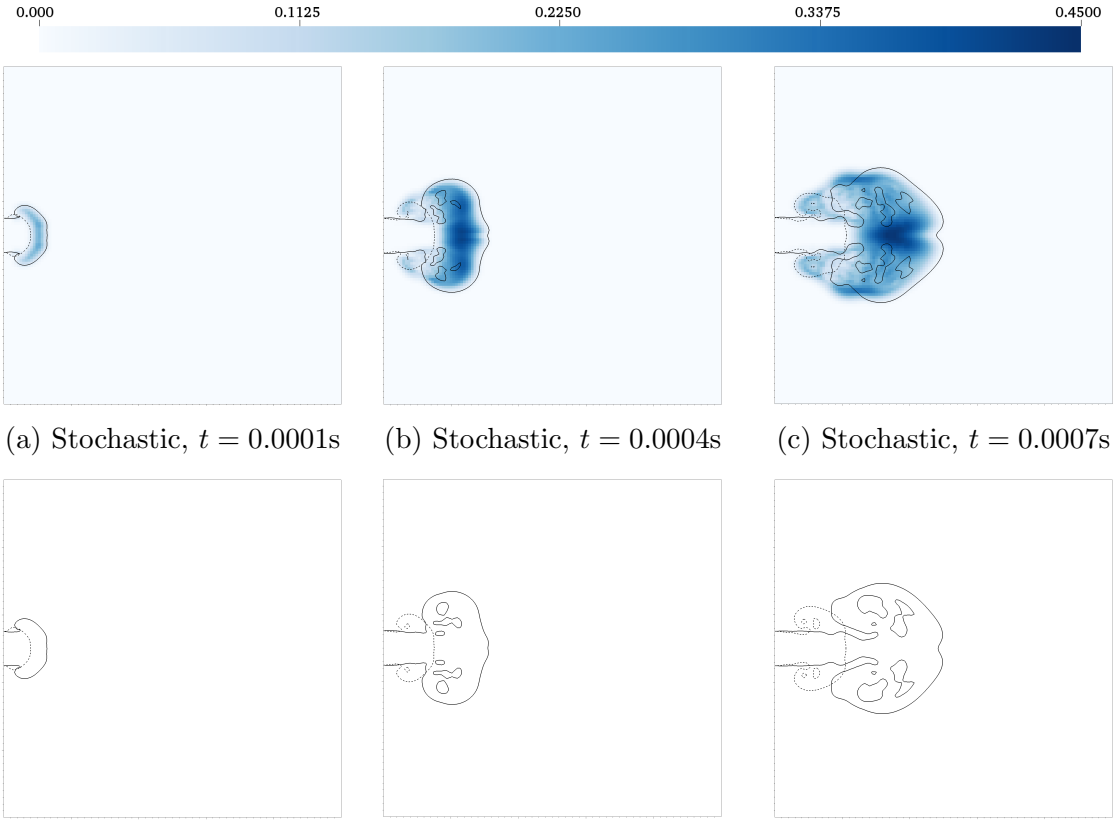
(a) Stochastic, $t = 0.0003$s (b) Stochastic, $t = 0.0005$s (c) Stochastic, $t = 0.0007$s

Figure 2.26: Results of a stochastic atomizing jet on 128×128 grid using a basis order of $N = 15$. The dashed line indicates the interface boundary for the solution with the smallest surface tension coefficient (i.e. $\gamma(\zeta = -1) = 36.4$), while the solid line indicates solution with the largest surface tension coefficient (i.e. $\gamma(\zeta = +1) = 109.2$). The color bar represents variance about the level set scalar.

when compared with the stochastic solutions at $\zeta = -1$ and $\zeta = +1$. As shown in previous cases, differences between deterministic and stochastic results are present when not enough basis functions are used to fully resolve the solution space. When more variation is present, more basis functions are needed for convergence, but with only 15 basis functions the method is able to capture this large amount of variability.

Conclusions

Intrusive uncertainty quantification methods can be applied to the level set interface capturing method, offering the ability to predict how input uncertainties manifest in outputs. The scheme presented is convergent to analytic results, and illustrates the ability of intrusive UQ to solve an infinite number of simulations at once in a multiphase system. Test cases of the two-dimensional oscillating droplet agree with analytical solutions, both for deterministic simulations, as well as UQ



(a) Stochastic, $t = 0.0001$s (b) Stochastic, $t = 0.0004$s (c) Stochastic, $t = 0.0007$s

(d) Deterministic, $t = 0.0001$s (e) Deterministic, $t = 0.0004$s (f) Deterministic, $t = 0.0007$s

Figure 2.27: Results of a stochastic atomizing jet on 128×128 grid (top) using a basis order of $N = 15$ compared to bounding solutions of two deterministic runs (bottom). The dashed line indicates the interface boundary for the solution with the smallest incoming velocity (i.e. $u_{\text{in}}(\zeta = -1) = 500$), while the solid line indicates solution with the largest incoming velocity (i.e. $u_{\text{in}}(\zeta = +1) = 1500$). The color bar represents variance about the level set scalar.

simulations. The quality of UQ simulation results are shown to converge as the number of basis functions increased.

This framework was also tested by application to the more difficult problem of an atomizing jet. Deterministic simulations showed convergence with smaller droplet breakup and conservation of mass. Simulations involving uncertainty corresponded to comparisons with deterministic model runs. Again, UQ results converged when enough basis functions were present.

The results in this paper demonstrate that performing intrusive UQ of multiphase flows is possible and the proposed method provides useful results. Several areas could be improved to enhance the quality of the framework and produce more accurate results. Removing the quadratures used to perform several integrals would reduce numerical errors and computational cost. For instance, creation of a quadrature-free calculation of curvature needed to find the surface tension force would produce a better result. Time step studies for both real and pseudo time are necessary to determine the best way to advance time when uncertainty is present, and potentially create a time step that is ζ dependent. It would also be useful to determine the relationship between the ε values associated with the reinitialization equation and limit the general diffusion term to only act away from the phase interface. Finally, applying the method to more realistic problems will allow the computational cost to be assessed and compared with non-intrusive results such as Monte-Carlo.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1511325. Computational efforts were performed on the Hyalite High-Performance Computing System, operated and supported by University Information Technology Research Cyberinfrastructure at Montana State University.

A FAST, DENSITY DECOUPLED PRESSURE SOLVER FOR AN INTRUSIVE STOCHASTIC MULTIPHASE FLOW SOLVER

Contribution of Authors and Co-Authors

Manuscript in following chapter

Author: Brian Turnquist

Contributions: primary author, performed research, code development

Author: Mark Owkes

Contributions: secondary author, review and feedback, troubleshooting

Manuscript Information

Brian Turnquist and Mark Owkes

Journal of Computational Physics

Status of Manuscript:
____ Prepared for submission to a peer-reviewed journal
_X_ Officially submitted to a peer-reviewed journal
____ Accepted by a peer-reviewed journal
____ Published in a peer-reviewed journal

Elsevier, Inc.
Submitted December 10, 2019

Abstract

Often, the most expensive aspect of solving the incompressible form of Navier-Stokes is the solution of the pressure Poisson equation. For a single phase deterministic model the pressure calculation is costly. Expanded to an intrusive stochastic multiphase framework, the simulation expense grows dramatically due to coupling between the stochastic pressure field and stochastic density. To address this issue in a deterministic framework, Dodd and Ferrante ("A fast pressure-correction method for incompressible two-fluid flows" Journal of Computational Physics, 273, 416-434, 2014) created a density decoupled method which utilizes a projected pressure field and constant density to modify the standard pressure correction method. The resulting method is useful for improving computational cost for one-fluid methods of multiphase flow calculations. In this paper, we extend the method of Dodd and Ferrante to intrusive uncertainty quantification of multiphase flows. The work improves upon the original formulation by modifying the projected field. The new method is assessed in terms of accuracy and reduction in computational cost with oscillating droplet, damped surface wave, and atomizing jet test cases where we find convergence of results with the proposed method to those of a traditional pressure correction method, as well as concurrence to analytic solutions, where appropriate.

Introduction

Methods of uncertainty quantification (UQ) can be placed into two categories: intrusive and non-intrusive. The non-intrusive category includes approaches such as Monte Carlo [50], collocation methods [47], and non-intrusive polynomial chaos (PC) [32]. The latter two essentially improve on a Monte Carlo by presenting a better way to select the input values so not as many simulations need to be run. In

all cases a standard solver can be utilized, which is run many times with parameters selected from a distribution of inputs to compile a database of simulation results. This database is then used to calculate useful statistics of the system in question.

Unlike non-intrusive flow solvers, intrusive UQ methods require a change in the fundamental structure of the solver resulting from modified equations created by the inclusion of stochastic (random) variables. An intrusive solver is created with stochastic variables which store information as a function of added uncertainty dimensions. Stochastic variables developed as a function of uncertainty may take several forms, including (PC) [83] and Karhunen-Loeve expansions [35, 45]. Each of these methods offer their own advantages. The advantage of PC lies in the ability to utilize any number of uncertainty dimensions, the availability of a number of orthogonal basis function families, and the straightforward integration of continuous PC variables into many systems of differential equations. Assuming stable intrusive and non-intrusive UQ schemes, computational cost comparisons are based on the time it takes to generate a reliable source of statistical information on the system being modeled.

Considering UQ applications to multiphase flow dynamics, application of intrusive UQ methods to gas-liquid flows is a developing field. Le Maître *et al.* [39, 40] first developed the stochastic Navier-Stokes equations for single-phase incompressible flows utilizing a PC expansion. Since these works, several studies have implemented a PC-based approach to single-phase flows for a variety of test cases ([19, 72, 84]). Previous work by Turnquist & Owkes [81] provided the first intrusive UQ method for gas-liquid multiphase flows named the multiUQ framework. The current work builds on this previous work by reducing the computational cost.

In either intrusive or non-intrusive UQ methodology, much computational expense is devoted to solving the pressure Poisson equation. To numerically solve

the incompressible Navier-Stokes equations, the standard pressure correction method (SPCM), first introduced by Chorin [10], is a commonly used approach. With the SPCM, time is discretized so that at every time step the convective, viscosity, and any source terms are evaluated and used to predict the velocity without the pressure term. Continuity (or mass conservation) is then used to enforce a divergence free condition at the next time step, while also creating an elliptic Poisson equation to solve for pressure. The approach makes it possible to solve the Navier-Stokes equations with imposed boundary conditions at reasonable computational expense. Further work has been done to expand the method, including improvements to order [82], and application to unstructured grids [77]. When using the SPCM in a multiphase scenario, the pressure Poisson equation becomes coupled to density, which adds computational cost and limits the possible algorithms used to solve. In an effort to counter this cost and following the work of Dong and Chen [18], Dodd and Ferrante [17] proposed a density decoupled pressure correction method (DDPCM) which would allow for using a fast Fourier transform (FFT) based solver. While numerical errors are added to the model, computational cost is reduced; certainly the trade off is worth consideration.

Given the computational cost improvements of a DDPCM in the deterministic setting, it seemed reasonable to apply this methodology to the multiUQ framework [81]. Because of the coupled nature of non-linear terms in the stochastic Navier-Stokes equations, due to the use of PC variables, the simulation expense grows at an exponential rate. This so-called curse of dimensionality increases the computational cost very rapidly for intrusive UQ. However, the same curse also affects non-intrusive methods. For example, the use of a Monte-Carlo [50] approach with two or more uncertain variables requires a way to compare the effect of one uncertain variable on another, compounding the number of simulations run to get convergent statistics.

Due to this problem, understanding the interaction in uncertainty between multiple variables in a multiphase system is extremely expensive. Use of accurate and cost effective numerical techniques will bring these analyses within reach.

This narrative seeks to develop a more efficient pressure correction approach for stochastic multiphase flows by applying the DDPCM to the multiUQ framework outlined in Turnquist & Owkes [81]. First, a mathematical development of the stochastic DDPCM is be developed, followed by a derivation of the numerical methods. Third, we present test cases which illustrate the computational improvement over previously published methods and the error associated with the density decoupled approach. Finally, we close with a summary of the results and a discussion of where this work will fit in moving forward.

Mathematical Development

Since the focus of this work is to develop an efficient pressure solver for stochastic multiphase flows, we begin with a development of the stochastic equations for fluid motion. Assuming the fluids are incompressible, this motion can be explained by the Navier-Stokes equations, where

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\eta \nabla P + \eta \nabla \cdot \left[\mu \left(\nabla \boldsymbol{u} + \nabla^{\mathrm{T}} \boldsymbol{u} \right) \right] + \eta \boldsymbol{f}_{\sigma} \delta_s \tag{3.1}$$

for velocity $\boldsymbol{u}$, time t, specific volume $\eta = 1/\rho$ (for density ρ), pressure P, dynamic viscosity μ, and surface tension force $\boldsymbol{f}_{\sigma} = \sigma \kappa \boldsymbol{n}$, where $\boldsymbol{n}$ is the interface normal vector, σ is the surface tension coefficient, and κ is the interface curvature. The surface tension force is only applied at the interface boundary, which is denoted by the Dirac delta function δ_s. Additionally, conservation of mass for the incompressible

form of Navier-Stokes is accomplished with the continuity equation,

$$\nabla \cdot \boldsymbol{u} = 0. \tag{3.2}$$

To account for the stochastic variables, this work utilizes the PC expansion as developed by Wiener [83], where some variable ψ, which varies in space $\boldsymbol{x}$ and time, may be allowed to vary in any number of uncertainty dimensions $\boldsymbol{\zeta}$ such that

$$\psi(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \sum_{k=0}^{N} \psi_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{\zeta}) = \psi_k(\boldsymbol{x}, t)\phi_k(\boldsymbol{\zeta}), \tag{3.3}$$

for basis weights $\psi_k(\boldsymbol{x}, t)$. $\phi_k(\boldsymbol{\zeta})$ for $k = 0, \ldots, N$ is a set of $N + 1$ orthogonal basis functions upon which the variable is projected. Any of several sets of orthogonal polynomials may be used, including Legendre, Hermite, Laguerre, and Chebychev, though it has been shown each works well for certain function behaviors. For example, Wiener [83] showed the Hermite polynomials can represent Gaussian distributions with a small number of basis functions.

Allowing uncertainty to exist about all variables (except time) in equation 3.1, we substitute the stochastic velocity $\boldsymbol{u}_k\phi_k$, specific volume $\eta_k\phi_k$, pressure $P_k\phi_k$, viscosity $\mu_k\phi_k$, and surface tension $\boldsymbol{f}_{\sigma:k}\phi_k$. The result is

$$\begin{aligned} \frac{\partial \boldsymbol{u}_k\phi_k}{\partial t} + \boldsymbol{u}_k\phi_k \cdot \nabla \boldsymbol{u}_l\phi_l &= -\eta_k\phi_k \nabla P_l\phi_l \\ &+ \eta_k\phi_k \nabla \cdot \left[\mu_l\phi_l \left(\nabla \boldsymbol{u}_m\phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m\phi_m\right)\right] + \eta_k\phi_k \boldsymbol{f}_{\sigma:l}\phi_l, \end{aligned} \tag{3.4}$$

and is one form of the stochastic Navier-Stokes equations. We also have the stochastic continuity equation

$$\nabla \cdot \boldsymbol{u}_k\phi_k = 0. \tag{3.5}$$

Because this form is difficult to work with and the real values of interest are the basis

weights, we utilize the property of orthogonality inherent in the basis functions. For the Legendre polynomials

$$\int_{-1}^{1} \phi_k \phi_b \delta\zeta = \begin{cases} \langle \phi_k \phi_b \rangle & k = b \\ 0 & k \neq b \end{cases}. \tag{3.6}$$

To this end, we first multiply equation 3.4 by a test function ϕ_b, resulting in

$$\begin{aligned} \frac{\partial \boldsymbol{u}_k \phi_k}{\partial t} \phi_b + \boldsymbol{u}_k \phi_k \cdot \nabla \boldsymbol{u}_l \phi_l \phi_b = -\eta_k \phi_k \nabla P_l \phi_l \phi_b \\ + \eta_k \phi_k \nabla \cdot \left[\mu_l \phi_l \left(\nabla \boldsymbol{u}_m \phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \phi_m \right) \right] \phi_b + \eta_k \phi_k \boldsymbol{f}_{\sigma:l} \phi_l \phi_b. \end{aligned} \tag{3.7}$$

To leverage the property of orthogonality, we then integrate over the region of orthogonality, $[-1, 1]$,

$$\begin{aligned} \int_{-1}^{1} \frac{\partial \boldsymbol{u}_k \phi_k}{\partial t} \phi_b + \boldsymbol{u}_k \phi_k \cdot \nabla \boldsymbol{u}_l \phi_l \phi_b d\zeta = \int_{-1}^{1} -\eta_k \phi_k \nabla P_l \phi_l \phi_b \\ + \eta_k \phi_k \nabla \cdot \left[\mu_l \phi_l \left(\nabla \boldsymbol{u}_m \phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \phi_m \right) \right] \phi_b + \eta_k \phi_k \boldsymbol{f}_{\sigma:l} \phi_l \phi_b d\zeta, \end{aligned} \tag{3.8}$$

and divide through by the integral $\langle \phi_b \phi_b \rangle$ to get a more useful form of the stochastic Navier-Stokes equations,

$$\frac{\partial \boldsymbol{u}_b}{\partial t} + \boldsymbol{u}_k \cdot \nabla \boldsymbol{u}_l C_{klb} = -\eta_k \nabla P_l C_{klb} + \eta_k \nabla \cdot \left[\mu_l \left(\nabla \boldsymbol{u}_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \right) \right] C_{klmb} + \eta_k \boldsymbol{f}_{\sigma:l} C_{klb}, \tag{3.9}$$

for $b = 0, \ldots, N$ where

$$C_{klb} = \frac{\int_{-1}^{1} \phi_k \phi_l \phi_b d\zeta}{\int_{-1}^{1} \phi_b \phi_b d\zeta} = \frac{\langle \phi_k \phi_l \phi_b \rangle}{\langle \phi_b \phi_b \rangle} \tag{3.10}$$

and

$$C_{klmb} = \frac{\langle \phi_k \phi_l \phi_m \phi_b \rangle}{\langle \phi_b \phi_b \rangle}. \tag{3.11}$$

Additionally, the stochastic continuity equations are

$$\nabla \cdot \boldsymbol{u}_b = 0 \tag{3.12}$$

for $b = 0, \ldots, N$.

Calculation of the surface tension force is not trivial. Equations 3.9 and 4.11 are implemented in an incompressible two-phase system utilizing a conservative level set interface capturing scheme as outlined in Turnquist & Owkes [81]. The surface tension is then found by way of a continuum surface force method, as first described by Brackbill *et al.* [7]. Tryggvason *et al.* [79] discuss smoothing the interface over a color function, in this case the conservative level set ψ, such that $\nabla\psi \approx \nabla H = \boldsymbol{n}\delta_s$. This methodology operates as a smoothed Heaviside function, $H(\boldsymbol{x})$, in which the stochastic implementation also deviates in the uncertainty domain, $\psi(\boldsymbol{x}, \boldsymbol{\zeta}) \approx H(\boldsymbol{x}, \boldsymbol{\zeta})$. As such, a deterministic curvature is calculated utilizing

$$\kappa = -\nabla \cdot \boldsymbol{n}, \tag{3.13}$$

for a unit normal about the interface, $\boldsymbol{n} = \nabla\psi/|\nabla\psi|$. Allowing uncertainty to exist about the level set, and thus the unit normal vectors and curvature, we then have a stochastic curvature

$$\kappa_b = \frac{1}{\langle \phi_b \phi_b \rangle} \int_{-1}^{1} \frac{\nabla \psi_k \phi_k}{|\nabla \psi_l \phi_l|} \phi_b d\zeta, \tag{3.14}$$

which is calculated with a Gaussian quadrature. To avoid projecting discontinuous unit normal vectors onto continuous basis functions, a unit normal is calculated at each quadrature point, thus projecting curvature κ onto the selected basis functions

ϕ_k. We then calculate a stochastic surface tension force as

$$\boldsymbol{f}_{\sigma:b} = \sigma_k \kappa_l \left(\nabla \psi_m\right) C_{klmb}. \tag{3.15}$$

Numerical Methodology

Stochastic Standard Pressure Correction

In the standard pressure correction method, the Navier-Stokes equations are time discretized (denoted by superscript n) such that

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -\boldsymbol{u}^n \cdot \nabla \boldsymbol{u}^n + \eta^n \nabla \cdot \left[\mu^n \left(\nabla \boldsymbol{u}^n + \nabla^{\mathrm{T}} \boldsymbol{u}^n\right)\right] + \eta^n \boldsymbol{f}_\sigma^n \delta_s \tag{3.16}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\eta^{n+1} \nabla P^{n+1}. \tag{3.17}$$

As shown, a predicted velocity field $\boldsymbol{u}^*$ is calculated without the pressure field. We then take the divergence of equation 3.17 to find pressure P^{n+1}, holding that $\nabla \cdot \boldsymbol{u}^{n+1} = 0$. In a multiphase system the density field varies, requiring expansion of the divergence operator and thus a coupling of density to pressure in the elliptic pressure Poisson equation

$$\nabla^2 P^{n+1} = \frac{\rho^{n+1} \nabla \cdot \boldsymbol{u}^*}{\Delta t} - \rho^{n+1} \nabla \eta^{n+1} \cdot \nabla P^{n+1}. \tag{3.18}$$

For a deterministic model, this equation isn't particularly problematic, though it is one of the most computationally expensive pieces of the solver. When building an intrusive UQ system, computational expense is compounded. Expanding equation 3.18 into a stochastic regime we have

$$\nabla^2 P_b^{n+1} = \frac{\rho_k^{n+1} \nabla \cdot \boldsymbol{u}_l^*}{\Delta t} C_{klb} - \rho_k^{n+1} \nabla \eta_l^{n+1} \cdot \nabla P_m^{n+1} C_{klmb} \tag{3.19}$$

for $b = 0, \ldots, N$, resulting in $N + 1$ coupled pressure Poisson equations. Due to the need for 4^{th} order multiplication tensor C_{klmb}, this results in a very expensive pressure step. While the multiplication tensor can be reduced by dropping all zero values, the equation becomes increasingly more expensive as more basis functions are required (i.e. increasing N).

Stochastic Density Decoupled Pressure Correction

To reduce the compounding computational cost of calculating a stochastic pressure field, it is useful to decouple the density from pressure. What is beneficial for a deterministic multiphase system is increasingly more beneficial in a stochastic system as the number of basis functions and uncertain variables used is increased. As mentioned previously, Dodd & Ferrante [17] discussed a modification of the SPCM where the pressure-density term of equation 3.17 is modified such that

$$\eta^{n+1}\nabla P^{n+1} \approx \eta_0 \nabla P^{n+1} + \left(\eta^{n+1} - \eta_0\right)\nabla \hat{P} \tag{3.20}$$

for some projected pressure field $\hat{P}$ and constant specific volume $\eta_0 = 1/\rho_0$. As shown, this substitution couples the needed pressure field P^{n+1} to a constant density term ρ_0, of which the gradient is zero. More specifically, as the projected pressure field $\hat{P}$ approaches the new pressure field P^{n+1}, the constant density term is canceled out, such that

$$\lim_{\hat{P} \to P^{n+1}} \left[\eta_0 \nabla P^{n+1} + \left(\eta^{n+1} - \eta_0\right)\nabla \hat{P}\right] = \eta^{n+1}\nabla P^{n+1}. \tag{3.21}$$

While the constant density term is essentially arbitrary, for numerical stability [17] report utilizing $\rho_0 = \min(\rho_1, \rho_2)$. In a stochastic implementation, we use the average for both ρ_1 and ρ_2 to find the minimum. Substitution of equation 3.20 into the

resulting system of equations for the modified pressure correction method is then

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -\boldsymbol{u}^n \cdot \nabla \boldsymbol{u}^n + \eta^n \nabla \cdot \left[\mu^n \left(\nabla \boldsymbol{u}^n + \nabla^{\mathrm{T}} \boldsymbol{u}^n\right)\right] + \eta^n \boldsymbol{f}_\sigma^n \delta_s \tag{3.22}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\eta_0 \nabla P^{n+1} - \left(\eta^{n+1} - \eta_0\right) \nabla \hat{P}. \tag{3.23}$$

A pressure Poisson equation is found by taking the divergence of equation 3.23 and enforcing $\nabla \cdot \boldsymbol{u}^{n+1} = 0$, leaving

$$\nabla^2 P^{n+1} = \rho_0 \frac{\nabla \cdot \boldsymbol{u}^*}{\Delta t} - \rho_0 \nabla \cdot \left(\eta^{n+1} - \eta_0\right) \nabla \hat{P}. \tag{3.24}$$

This equation is linear and can be easily calculated by a number of linear solution algorithms that exist. Additionally, expanding equation 4.31 for stochastic use by substitution of PC variables and integration over ζ we have

$$\nabla^2 P_b^{n+1} = \rho_0 \frac{\nabla \cdot \boldsymbol{u}_b^*}{\Delta t} - \rho_0 \nabla \cdot \left(\eta_k^{n+1} - \eta_0\right) \nabla \hat{P}_l C_{klb}, \tag{3.25}$$

which are $N + 1$ decoupled pressure Poisson equations. The right hand side of this equation is constant for all pressure iterations. Each basis weight P_b^{n+1} can be calculated by use of any solution algorithm, such as those contained within the HYPRE package maintained by Lawrence Livermore National Lab, which was utilized for this study. This is significantly less costly than the stochastic SPCM, where the coupling of pressure and density required looping over the multiplication tensor C_{klmb} for each iteration of the pressure solver.

To complete the stochastic Navier-Stokes equations with the DDPCM, we also

include stochastic variables in equations 3.22 and 3.23 leading to

$$\frac{\boldsymbol{u}_b^* - \boldsymbol{u}_b^n}{\Delta t} = -\boldsymbol{u}_k^n \cdot \nabla \boldsymbol{u}_l^n C_{klb} + \eta_k^n \nabla \cdot \left[\mu_l^n \left(\nabla \boldsymbol{u}_m^n + \nabla^{\mathrm{T}} \boldsymbol{u}_m^n\right)\right] C_{klmb} + \eta_k^n \boldsymbol{f}_{\sigma:l}^n C_{klb} \tag{3.26}$$

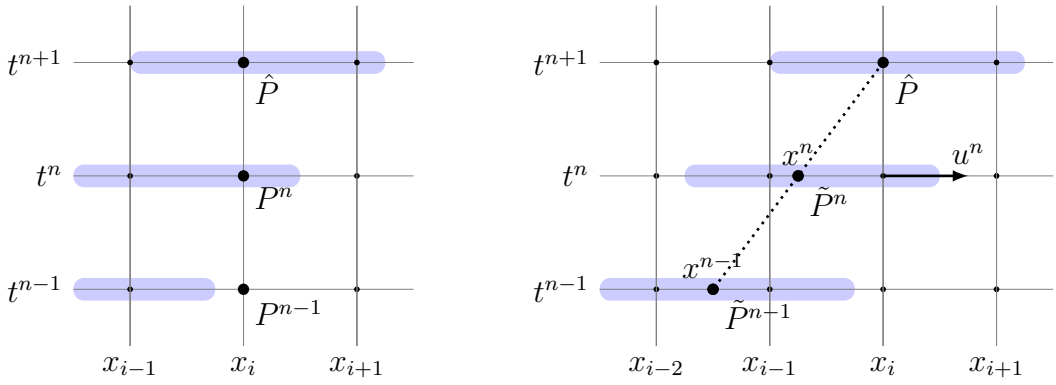
$$\frac{\boldsymbol{u}_b^{n+1} - \boldsymbol{u}_b^*}{\Delta t} = -\eta_0 \nabla P_b^{n+1} - \left(\eta_k^{n+1} - \eta_0\right) \nabla \hat{P}_l C_{klb}. \tag{3.27}$$

Estimation of $\hat{P}$ Clearly, the accuracy of the final pressure field is a function of $\left(\hat{P} - P^{n+1}\right)$. The best estimate of $\hat{P}$ mentioned by Dodd and Ferrante [17] is a linear projection, where

$$\hat{P} = 2P^n - P^{n-1}. \tag{3.28}$$

However, this simple projection is subject to problems, as discussed by Cifani [12], who made efforts to improve it. In a multiphase flow the passage of the interface over a given location results in a pressure jump. Utilizing equation 3.28, at a grid point where the interface has just arrived, the estimate of $\hat{P}$ for the next time step would be for another pressure jump, as illustrated in Fig. 3.1a. Inversely, at points where the interface has just left, the estimate of $\hat{P}$ would be for another pressure drop. A double jump exists at the front and back of a moving droplet for $\hat{P}$. This is extremely problematic, and requires multiple time steps to clear away from any given cell, but existing about the interface throughout the simulation. While the DDPCM does maintain a divergence free velocity field, this difference $\left(\hat{P} - P^{n+1}\right)$ near the interface creates pressure fluctuations which are non-physical and introduce a great deal of error to the simulation.

Alternatively, to avoid the issue caused by a simple linear projection, a semi-Lagrangian interpolation method is proposed where particles at each grid point are traced back to their previous location, and the pressure $\tilde{P}$ at that point is found with



(a) Projection (Eq. 3.28) (b) Semi-Lagrangian (Eq. 3.34)

Figure 3.1: Illustration of the differences in the projection and semi-Lagrangian approaches to estimation of $\hat{P}$. Blue lines indicate the position of a one-dimensional liquid droplet. At left we see that the projection approach utilizes pressures P^n and P^{n-1} that are within the liquid and gas phases, respectively, to estimate $\hat{P}$. At right we see that the semi-Lagrangian approach utilizes pressures P^n and P^{n-1} that are both in the liquid phase to estimate $\hat{P}$.

bi-linear interpolation (for a 2-D system). The cell center located particle $\boldsymbol{x}_i$ can be traced back to its previous location using the partial differential equation

$$\frac{\partial \boldsymbol{x}}{\partial t} = \boldsymbol{u}, \tag{3.29}$$

discretized with an explicit Euler approach, e.g., $\boldsymbol{x}^n = \boldsymbol{x}_i - \boldsymbol{u}^n \Delta t$ and $\boldsymbol{x}^{n-1} = \boldsymbol{x}_i - 2\boldsymbol{u}^n \Delta t$. Once located, the pressure is found via bi-linear interpolation of the nearest grid points. Due to the stochastic velocity field, the previous locations $\boldsymbol{x}^n$ and $\boldsymbol{x}^{n-1}$ are uncertain. We defined a particle $\boldsymbol{x}_i$ at the cell center and track it back through an uncertain velocity field. Following substitution of stochastic velocity $\boldsymbol{u}_k \phi_k$, multiplication by a test function, and integration over ζ, we discretize equation 3.29

to find the previous particle locations

$$\begin{aligned} \boldsymbol{x}_b^n &= \frac{\langle \phi_b \rangle}{\langle \phi_b \phi_b \rangle} \boldsymbol{x}_i - \boldsymbol{u}_b^n \Delta t \quad \text{and} \\ \boldsymbol{x}_b^{n-1} &= \frac{\langle \phi_b \rangle}{\langle \phi_b \phi_b \rangle} \boldsymbol{x}_i - 2\boldsymbol{u}_b^n \Delta t. \end{aligned} \tag{3.30}$$

Even in a deterministic setting, depending on the location of this particle, the nearest neighbors may be shifted to one of the four quadrants surrounding the central pressure $P_{i,j}$. For explanation, assuming the particle to be in the third quadrant, the semi-Lagrangian pressure $\tilde{P}^n$ at (x, y) may be found with

$$\tilde{P}_{i,j}^n = \frac{1}{\Delta x \Delta y} \begin{bmatrix} x_{i+1} - x^n & x^n - x_i \end{bmatrix} \begin{bmatrix} P_{i+1,j}^{n-1} & P_{i+1,j-1}^{n-1} \\ P_{i,j}^{n-1} & P_{i,j-1}^{n-1} \end{bmatrix} \begin{bmatrix} y_j - y^n \\ y^n - y_{j-1} \end{bmatrix}. \tag{3.31}$$

Rather than utilize a quadrature, the average particle location $\boldsymbol{x}_0^n$ is used given $\boldsymbol{x}_k^n \phi_k$ (i.e. $\phi_0 = 1$). Thus, the four nearest neighbors to $\boldsymbol{x}_0^n$ (based on the quadrant it falls in) are used to calculate the pressure field $\tilde{P}^n$ utilizing bi-linear interpolation. Importing uncertain variables, multiplying by a test function and integrating, we then find a stochastic bi-linear interpolation

$$\begin{aligned} \tilde{P}_{b:i,j}^n = & \frac{1}{\langle \phi_b \phi_b \rangle} \int_{-1}^{1} \frac{\phi_b}{\Delta x \Delta y} \cdot \\ & \begin{bmatrix} x_{i+1} - x_0^n \phi_0 & x_0^n \phi_0 - x_i \end{bmatrix} \begin{bmatrix} P_{l:i+1,j}^n \phi_l & P_{l:i+1,j-1}^n \phi_l \\ P_{l:i,j}^n \phi_l & P_{l:i,j-1}^n \phi_l \end{bmatrix} \begin{bmatrix} y_j - y_0^n \phi_0 \\ y_0^n \phi_0 - y_{j-1} \end{bmatrix} d\zeta. \end{aligned} \tag{3.32}$$

Performing the integration over ζ, and noting the simplification by utilizing the

average previous location, we have

$$\tilde{P}^n_{b:i,j} = \frac{1}{\Delta x \Delta y} \begin{bmatrix} x_{i+1} - x_0^{n-1} & x_0^{n-1} - x_i \end{bmatrix} \begin{bmatrix} P^n_{b:i+1,j} & P^n_{b:i+1,j-1} \\ P^n_{b:i,j} & P^n_{b:i,j-1} \end{bmatrix} \begin{bmatrix} y_j - y_0^{n-1} \\ y_0^{n-1} - y_{j-1} \end{bmatrix}. \tag{3.33}$$

Once this interpolated pressure $\tilde{P}^n$ has been calculated, and utilizing the interpolated pressure from the previous time step $\tilde{P}^{n-1}$, we can again extrapolate to the next time for our estimate

$$\hat{P} = 2\tilde{P}^n - \tilde{P}^{n-1}. \tag{3.34}$$

The difference here is that we are tracking the interface as it arrives and moves through any given grid point, as illustrated in Fig. 3.1b. While this provides a reasonable and bounded estimate of $\hat{P}$, which avoids a double jump, this estimate can be further improved within the context of the numerical scheme. Using an iterative approach, each successive iteration can utilize the previous calculation of pressure as $\hat{P}$ as described in the next section.

Iterative Crank-Nicolson

Given that the updated velocity field $\boldsymbol{u}^{n+1}$ is calculated using both the calculated pressure field P^{n+1} and $\hat{P}$, the best calculation of the next time iteration occurs when $\hat{P} = P^{n+1}$. To implement this, an iterative Crank-Nicolson scheme is used. For the first iteration, the estimated pressure field is calculated using either the linear projection method (equation 3.28) or the semi-Lagrangian approach of equation 3.34. Subsequent iterations use the previous calculation of P^{n+1} as $\hat{P}$, which converges to the best estimate of P^{n+1}. Furthermore, within each Crank-Nicolson step, while the right hand side of Navier-Stokes is constant, it is possible to further iterate over the pressure solver, nudging P^{n+1} to a steady state by improving the estimate of $\hat{P}$.

```
 1 ! Loop over time
 2 do n = 1,niter
 3
 4   ! Estimate pressure field
 5   call semiLagrangian        ! Eq. 34
 6   ! Phat = 2*P(n) - P(n-1)   ! Eq. 28
 7
 8   ! Perform Crank-Nicolson iterations
 9   cn = 0
10   do while ((cn.lt.Nc).and.(res.lt.converge))
11     ! Calculate ustar
12     call velocityPredict
13
14     ! Loop over pressure solver
15     pi = 0
16     do while ((pi.lt.Np).and.(resP.lt.converge))
17       ! Update pressure with elliptic solver
18       call ellipticSolver
19       Phat = P(n+1)
20     end do
21
22     ! Perform pressure correction
23     call velocityCorrect
24
25     ! Transport the interface
26     call interfaceTransport
27
28   end do
29 end do
```

Figure 3.2: Pseudo code describing the procedural order for an iterative approach to the fast pressure solver.

As shown in Fig. 3.2, the estimate of the pressure field, $\hat{P}$ is continually updated throughout the iterative process, converging to P^{n+1} and reducing the error inherent in the method. The maximum number of Crank-Nicolson (N_c) and pressure iterations (N_P) can be reduced if some convergence criterion is reached (variable converge). Note, there is a compounding effect of updating $\hat{P}$ inside the Crank-Nicolson loop. For low density ratios (≤ 100) and/or slow velocity fields it is sufficient to loop over the pressure solver once (i.e. $N_p = 1$). However, at high density ratios and/or rapidly evolving systems it is necessary to improve our estimate of the next pressure field, increasing our value of N_p. This can be done in lieu of reducing the time step size, or CFL value, as suggested by [17].

To obtain second-order accuracy in the time marching scheme, it is only necessary to perform 2 Crank-Nicolson iterations (i.e. $N_c = 2$). Given that updating the Crank-Nicolson loop runs through all calculations of the level set, velocity, and pressure solver, it makes sense to reduce this number and iterate over the pressure solver to converge the estimate of $\hat{P}$ to P^{n+1}. However, in practice it is not this straightforward, since the first step of a Crank-Nicolson is essentially an explicit Euler prediction, which is then improved some by another iteration. Thus, our first estimates of $\hat{P}$ are based on a first order accurate prediction. Through testing we have found that increasing N_c helps to converge $\hat{P}$, resulting in fewer overall iterations of the pressure Poisson equation per time step, as discussed in more detail in section 3.

It also makes sense to discuss the oddity of looping several times over the most expensive part of the simulation in an attempt to improve computational efficiency. The improvement of efficiency comes from a combination of the ability to use more efficient solution methods and the uncoupled nature of the pressure Poisson equation. Utilizing the SPCM, convergence of P^{n+1} is slow, and due to the coupling of density and pressure each basis weight P_b^{n+1} is linked to all others. By being able to solve for

each basis weight separately, the reduced number of calculations due to not looping over the entire 4^{th} order multiplication tensor C_{klmb} at each time step significantly reduces the computational cost growth as basis functions are added. Thus, the computational cost savings grows with N. Additionally, implementing an iterative approach allows for easy improvements to the first estimate of $\hat{P}$, which then decreases computational cost further.

Test Cases and Computational Assessment

Two test cases are used to evaluate the accuracy and efficiency of the proposed method, while a third is used to test the method on a more complicated scenario. First, an oscillating droplet case is used as there exists an analytic solution providing the oscillation period [46]. This case tests the ability of the surface tension force to drive flow. Second, a damped surface wave, which also has an analytic solution [63], is used to judge the accuracy of the interplay between viscosity and surface tension. Finally, the third test case is an atomizing jet, which demonstrates the ability of the method to resolve difficult physical situations. For each case, the semi-Langrangian projection method of equation 3.34 is utilized for best results, with comparison to equation 3.28 in a high density ratio scenario. These three tests focus on how the modified pressure projection method effects the solution and computational cost.

Oscillating Droplet

The oscillating droplet is a common benchmark test for validating the accuracy and abilities of a multiphase solver [3, 15, 24, 59]. Rather than inducing flow from the boundary, this flow is driven by the surface tension force, which allows for an indirect test of the accuracy of the numerical method to calculate the surface tension by comparison of the analytical oscillation period to that determined from the

simulation. As mentioned, the period of oscillation was defined by Lord Rayleigh [46] and described in 2-D by Fyfe *et al.* [23], where

$$\tau = 2\pi\sqrt{\frac{(\rho_1 + \rho_2)\, R^3}{6\sigma}} \tag{3.35}$$

for period τ and unperturbed radius $R = \sqrt{AB}$ of an ellipse described by $x^2/A^2 + y^2/B^2 = 1$, with semi-major and semi-minor axes A and B, respectively.

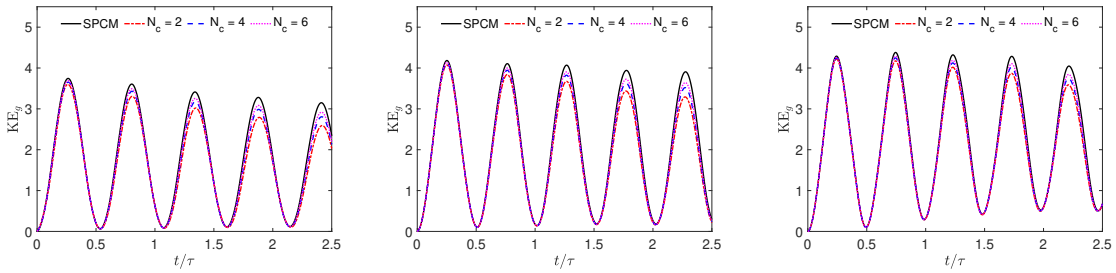
There are peculiarities to this case, some discussed by Salih and Ghosh Moulic [68]. First, the period of oscillation as defined by Lord Rayleigh assumed an inviscid system. We can see in equation 3.35 that fluid viscosity does not affect the period of oscillation. As discussed in [68], it is found that viscosity serves to dampen the amplitude of oscillations. Second, the period of oscillation of the simulation is found indirectly by computing the total kinetic energy of the system,

$$KE_{\mathrm{g}} = \frac{1}{2}\int_V \rho \boldsymbol{u} \cdot \boldsymbol{u} dV. \tag{3.36}$$

This works in part because the system begins as a static ellipse with only the potential energy of surface tension imposed. Kinetic energy then should grow and peak, but slowly diminish as the surface tension force then counteracts the movement of the fluid, at some point reaching the maximum ellipsoid shape about the 2^{nd} dimension, when a minima of movement is taking place.

We look at variations of the oscillating droplet case with an initial ellipse of $A = 0.25$ and $B = 0.15$ cm centered in a $[0, 2]^2$ domain. Liquid properties are first set to $\nu_1 = \nu_2 = 0$ for viscosity (in keeping with the original analytic result) while $\rho_1 = 1.0\ \frac{\mathrm{g}}{\mathrm{cm}^3}$ and $\rho_2 = 0.01\ \frac{\mathrm{g}}{\mathrm{cm}^3}$ are set for density. Surface tension coefficient is set at $\sigma = 72.8\ \frac{\mathrm{g}}{\mathrm{s}^2}$ for all simulations.

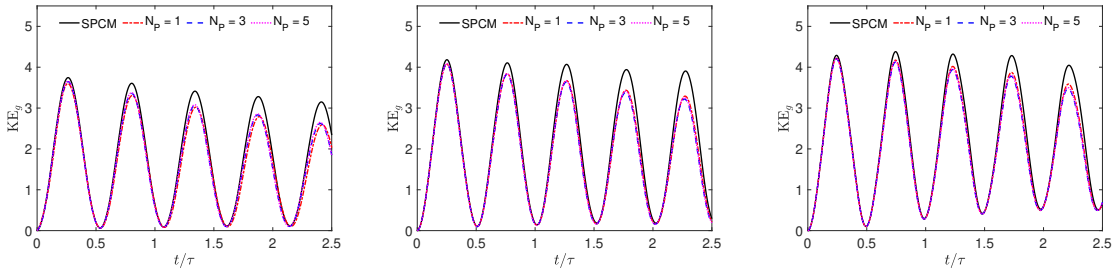
Comparison to Traditional Pressure Correction In the implementation of the DDPCM, there are two parameters, N_c and N_P, that can be used to improve the predicted pressure $\hat{P}$. The effect of N_c, the number of Crank-Nicolson iterations, is first investigated to determine if simply increasing N_c to improve the velocity, pressure, and level set transport will be enough to converge $\hat{P}$. With this approach to the numerical scheme, the magnitude of kinetic energy was found to be largely affected by the number of Crank-Nicolson iterations, while the period is shifted slightly. Fig. 3.3 illustrates this effect for a range of N_c and mesh sizes. This is due to the continually improved estimate of $\hat{P}$ with each iteration. Also shown is the more rapid decrease in kinetic energy with smaller N_c.



(a) 64×64 (b) 128×128 (c) 256×256

Figure 3.3: Global kinetic energy for a range of Crank-Nicolson iterations for three mesh sizes. Pressure is only calculated once per iteration, i.e. $N_P = 1$.

In implementing an iterative approach we might also consider if it is sufficient to run simply two Crank-Nicolson iterations ($N_c = 2$) to achieve 2^{nd} order accuracy in time, but loop over the pressure solver multiple times. Fig. 3.4 illustrates the solution tendency to this approach over a range of N_P with $N_c = 2$. We find that while this approach also appears to be convergent, there is slower improvement of kinetic energy magnitude. Again, as the estimate of $\hat{P}$ converges to P^{n+1}, the difference in the velocity field between the two correction methods shrinks.



(a) 64×64 (b) 128×128 (c) 256×256

Figure 3.4: Kinetic energy oscillations using the DDPCM for a range of pressure iterations N_P over three separate mesh sizes. For all cases $N_c = 2$. For reference, the kinetic energy plot with the SPCM is shown with the solid line.

We can see from Figs. 3.3 and 3.4 that finding a proper $\hat{P}$ is necessary, and that it is likely better to iterate over both the Crank-Nicolson scheme and the pressure solver. It is thus useful to find the most efficient combination of Crank-Nicolson and pressure iterations in hopes of achieving an acceptable level of error while keeping computational expense relatively low. For comparison, we see in Table 3.1 the simulation times for each of the test cases mentioned above, along with the average difference in kinetic energy between the two methods, $\bar{E}_{\text{diff}}$. The bottom row of the table displays the computational time for a simulation with the SPCM, which is orders of magnitude longer than any simulation with the DDPCM. Clearly there is a huge computational advantage to the DDPCM, even over several iterations of the pressure solver. This is due to the linearization of the Poisson equation, which allows standard and fast solvers such as the PFMG solver in the HYPRE library [11] from Lawrence Livermore National Lab that is used in this work.

To determine the most efficient possible simulation layout, multiple simulations were run with the oscillating droplet method, using the SPCM results as a baseline. These tests were run with the goal of achieving a satisfactory solution while also minimizing the total number of iterations needed to solve the pressure Poisson

N_c	N_P	Simulation Time (min)		Error ($\bar{E}_{\text{diff}}$)	
		64×64	128×128	64×64	128×128
6	1	0.930	5.693	0.074	0.080
4	1	0.759	4.151	0.110	0.118
2	1	0.591	2.852	0.182	0.190
2	3	0.799	3.813	0.167	0.206
2	5	0.859	4.959	0.157	0.194
2	SPCM	3.038	68.012	-	-

Table 3.1: Simulation run times of several combinations for DDPCM. Reported error is the average difference between DDPCM and SPCM over three oscillations.

equation to a given convergence level. Tests were run with a range of $N_c = [2, 6]$ and $N_P = [1, 7]$. These comparisons were done on a single course mesh of 64×64, and the results are summarized in Table 3.2 of the Appendix for this chapter.

Looking at Table 3.2, it appears two combinations make the most sense. In one case $N_c = 4$ and $N_P = 5$, while in the other $N_c = 5$ and $N_P = 4$. These two have very similar simulation run times, pressure time (both include 20 pressure iterations), and total pressure iterations for our divergence criterion. Since it is more useful to have a better estimate of $\hat{P}$ for the next guess at velocity, it makes the most sense to use the combination $N_c = 4$ and $N_P = 5$. Having made this determination, we now observe the convergence of the method due to mesh refinement, noting the improvement in computational efficiency over the previously reported stochastic Gauss-Seidel approach of Turnquist & Owkes [81]. Comparison to the traditional pressure correction method is accomplished by determining the differences between deterministic model runs of the proposed pressure solution method and the traditional method. Both models are allowed to run to a given divergence level

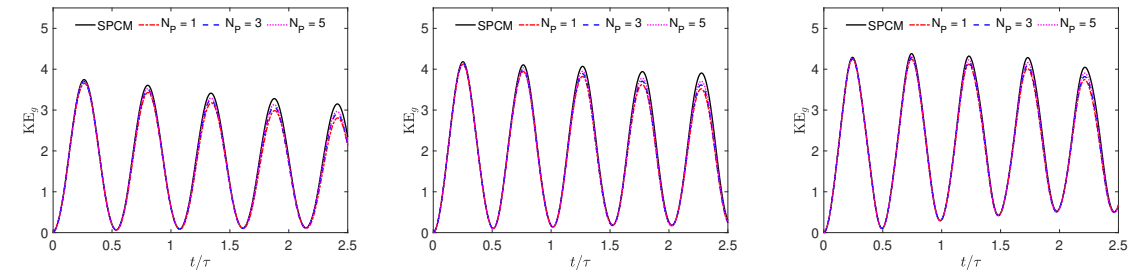
(i.e. $\nabla \cdot \boldsymbol{u} < 1 \times 10^{-14}$) with a set time step. Fig. 3.5 displays the difference in kinetic energy between the two methods,

$$\Delta \mathrm{KE}_g = \mathrm{KE}_{g:\mathrm{SPCM}} - \mathrm{KE}_{g:\mathrm{DDPCM}} \tag{3.37}$$

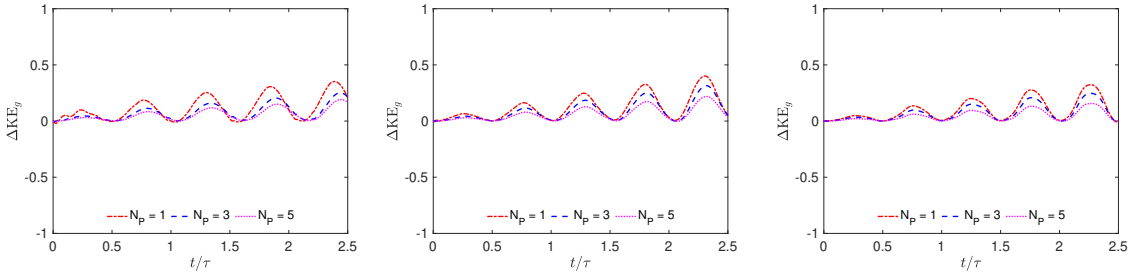
over several oscillations. There is also a slight phase shift that accounts for some of the difference in kinetic energy, likely caused by some momentum change due to the discrepancy between $\hat{P}$ and P^{n+1}. Also in Fig. 3.5, we can see continued convergence of the DDPCM to the SPCM as the number of pressure iterations is increased. While there is a relatively large difference in the solution from $N_P = 1$ to $N_P = 3$, the change is much less for the next jump. We see the trade-off here between computational efficiency and precision. However, in each case we do see similar behavior for both pressure correction methods. Additionally, the computational time for the SPCM on a 64^2 mesh was 6.31 minutes, while for the case of $N_P = 5$ the run time was 1.49 minutes, allowing for a good speed up with some additional error. Fig. 3.5 also displays convergence of DDPCM results to SPCM as the mesh is refined, with improvement of the kinetic energy calculations.

Finally, we compare the difference in the total simulation run time (using four Intel Core i7 2.5 Ghz processors) between standard and density decoupled pressure correction methods, as shown in Fig. 3.6. For this figure, simulations with the DDPCM are run with $N_c = 4$ and $N_P = 5$. In all simulations, the proportional cost of solving the pressure Poisson equation is $\approx 75\%$.

High Density Ratio Case An oscillating droplet with a high density ratio (i.e. an air and water system) is more numerically difficult to resolve. In the DDPCM framework, it is even more important to have a reasonable estimate of $\hat{P}$ to reduce computational errors. This is because the constant density term ρ_0 is then orders of

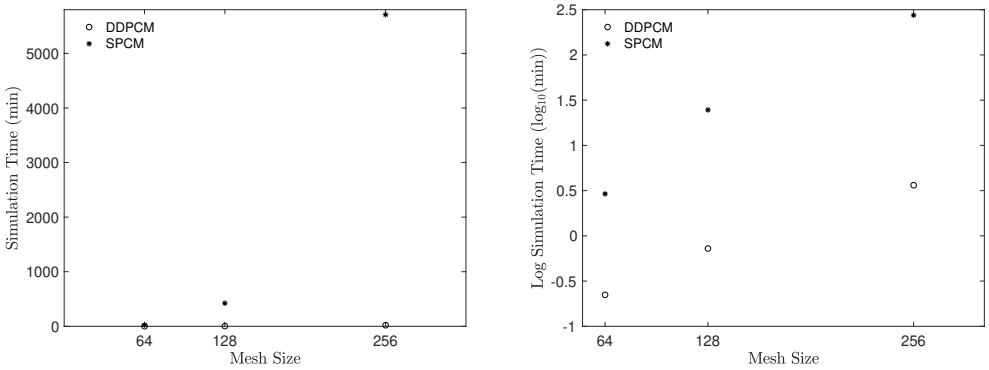


(a) 64×64 (b) 128×128 (c) 256×256



(d) Error, 64×64 (e) Error, 128×128 (f) Error, 256×256

Figure 3.5: Difference in kinetic energy plots between SPCM and DDPCM for a range of pressure iterations N_P over three separate mesh sizes. In each simulation $N_c = 4$. Simulation run times for the 64^2 mesh were 0.53, 1.00, 1.49, and 6.31 minutes for $N_P = \{1, 3, 5\}$ and SPCM, respectively.



(a) Time scale (b) Log time scale

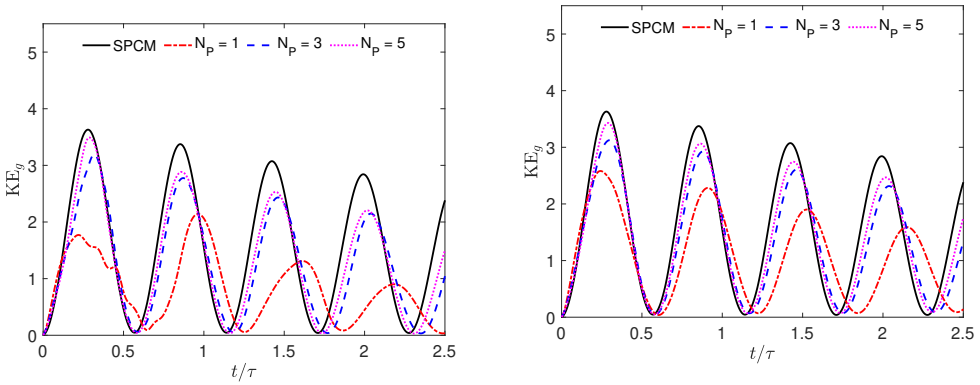
Figure 3.6: Computational time of oscillating droplet simulations over a range of mesh sizes for a set number of iterations.

magnitude different from the highest density fluid. This exacerbates the errors caused by the difference between $\hat{P}$ and P^{n+1}, and thus the resulting velocity field. Given this issue, we also consider the effect of a simple linear projection (equation 3.28), which creates unrealistic and non-physical jumps near the interface. This causes huge variations in the velocity field near the interface, and can cause simulation instabilities leading to failure. We compare this with the proposed semi-Lagrangian extrapolation which bounds $\hat{P}$ by the local maxima and minima of the current pressure field while tracking movement of the interface. This provides a stable first estimate which does not allow a double jump to occur and maintains a more stable estimate of $\hat{P}$.

For comparison of the methods, the issue caused by non-physical jumps is especially obvious with respect to high density ratios. Fig. 3.7 directly compares the two schemes. As shown, simply utilizing the basic projection mentioned in equation 3.28 is a very poor estimate without iterative help (i.e. the case of $N_P = 1$). By comparison, we see the proposed method has a more stable oscillation before improvement by iteration. Additionally, we see more rapid convergence to the SPCM utilizing the semi-Lagrangian projection.

As seen in Fig. 3.7, improvement of the estimate of $\hat{P}$ is necessary for high density ratios. Iteration over the pressure solver can quickly improve the estimate, while some additional Crank-Nicolson iterations add to stability. Additionally, the phase difference seen in Fig. 3.5 is improved by the use of the semi-Lagrangian extrapolation. Fig. 3.8 displays a range of results for combinations of N_c and N_P using the proposed semi-Lagrangian extrapolation for finding $\hat{P}$. In keeping with the results of our timing tests, we can see the diminishing returns on the reduction of error with the increase in both N_c and N_P, and settle with $N_c = 4$.

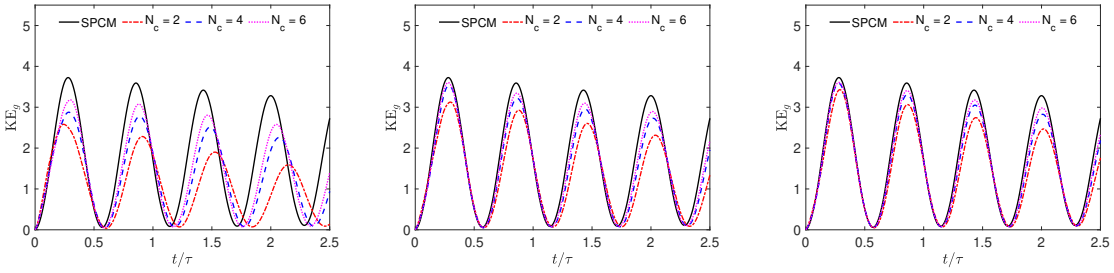
Displaying these results another way in Fig. **??**, as we increase the number of pressure iterations $N_P = [1, 5]$, utilizing $N_c = 4$, we can see that convergence of results



(a) Extrapolation method

(b) Combined semi-Lagrangian extrapolation

Figure 3.7: Effect of initial estimate of $\hat{P}$ on solution of high density ratio oscillating droplet. In each case $N_c = 2$.



(a) $N_P = 1$ (b) $N_P = 3$ (c) $N_P = 5$

Figure 3.8: Convergence to expected results as the number of Crank-Nicolson iterations and pressure estimation steps vary on a 64×64 mesh using the semi-Lagrangian extrapolation. Simulation results for the SPCM were run with $N_c = 4$ for accuracy.

is similar to that of the lower density ratio case presented previously. Additionally, we see convergence to the SPCM for the amplitude of kinetic energy, while the period of oscillation is in reasonable agreement for each.

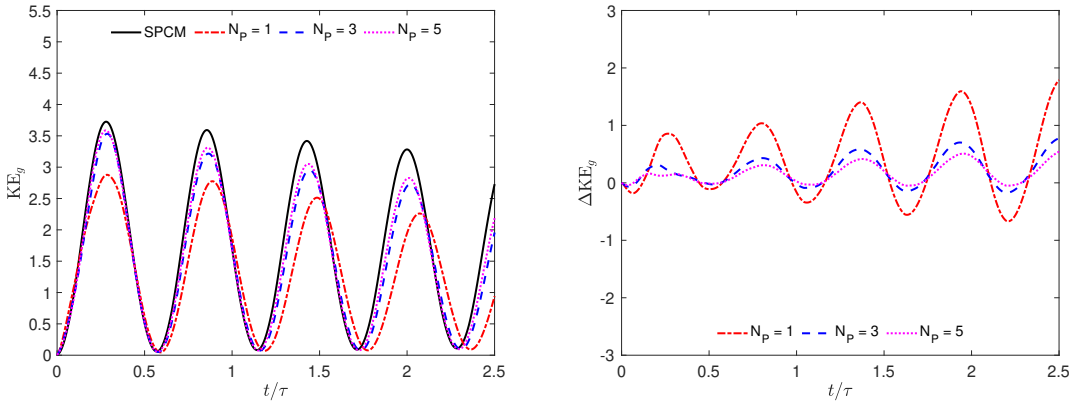


Figure 3.9: Difference in kinetic energy between standard pressure correction and several pressure iterations with DDPCM on a 64×64 mesh. In each simulation $N_c = 4$. Simulation run times were 0.371, 0.715, 1.026, and 4.477 minutes for $N_P = \{1, 3, 5\}$ and SPCM, respectively.

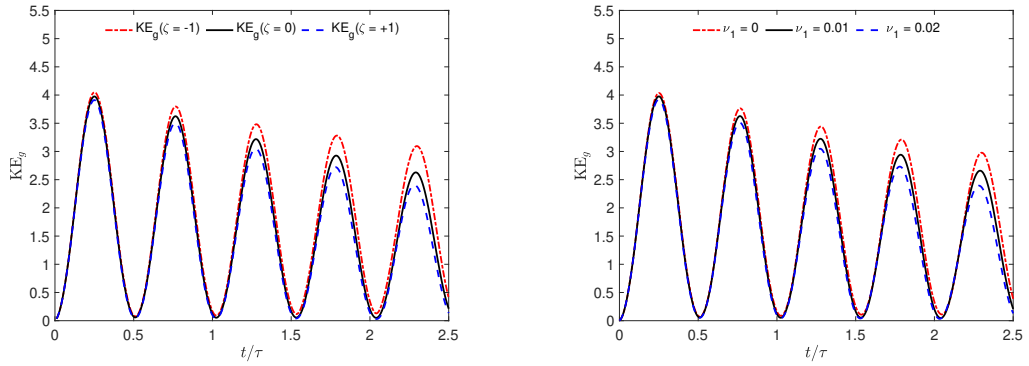
Effect of Viscosity on Frequency Given a stochastic multiphase solver, it is possible to look at the effect of viscosity on the oscillation of the droplet. For comparison in the stochastic model runs, the same statistics are used as the deterministic model runs, with the addition of comparing the model run times as the amount of uncertainty and basis functions is increased.

In this case we can look at the effect of viscosity on the frequency of oscillation by allowing for uncertainty about the viscosity of the fluid. To simplify the scenario,

we will utilize a uniform distribution about the fluids, where

$$\begin{aligned} \nu_1 &= 0.01 + 0.01\zeta \quad \text{and} \\ \nu_2 &= 0.15 + 0.15\zeta \end{aligned} \tag{3.38}$$

allow us to test a range of cases from an inviscid situation to one where viscosity is double that of air and water. Fig. 3.10 displays the results of this test case. We can see the viscosity damps the kinetic energy in the system, while the inviscid case maintains kinetic energy. We also see that the results of the UQ simulation closely match that of the deterministic simulations. Additionally, we see the viscosity has a negligible impact on the period of oscillation but does affect the dissipation rate as expected.

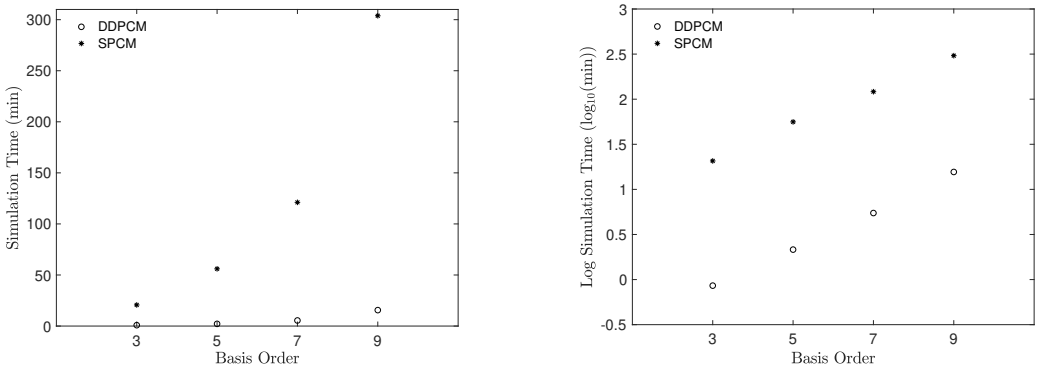


(a) Stochastic Simulation (b) Three Deterministic Simulations

Figure 3.10: Results of a stochastic oscillating droplet with variability about viscosity on a 64×64 mesh. At left are results from a single stochastic simulation, while at right are results from three deterministic simulations for comparison.

Accepting the level of error between the SPCM and DDPCM with $N_c = 4$ and $N_P = 5$, we look at a computational cost comparison of the stochastic simulations. For this comparison, we are interested in determining the effect of basis order N on the computational time, and the efficiency gained by implementing the DDPCM.

Timing tests were run using 100 time iterations with a convergence level of 10^{-8} for both the traditional and proposed pressure correction methods. All computations are performed on four 2.5 GHz Intel Core i7 processors with a 64^2 mesh. Fig. 3.11 illustrates the time taken to run these iterations over a range of basis functions. Given the vast difference in computational time, a log based plot is also presented. As shown, the cost of the SPCM grows much more rapidly than that of the DDPCM as basis order is increased.



(a) Time comparison (b) Log time comparison

Figure 3.11: Computational time for a set number of iterations over a range of basis functions using two types of pressure correction methods in a stochastic simulation.

Damped Surface Wave

We now look at the interaction between viscosity and surface tension as outlined by Prosperetti [63] and utilized by Herrmann [29]. This case superposes two viscous fluids with a slight disturbance imposed at the initial condition of signed distance function $g(\boldsymbol{x}, t)$ used to initialize the level set, where

$$g(\boldsymbol{x}, t = 0) = y - y_o + A_o \cos\left(x - \frac{h_g}{2}\right), \tag{3.39}$$

with $y_o = \pi$, $A_o = 0.01\lambda$, wavelength $\lambda = 2\pi$, and $h_g = \lambda/N_x$ for the number of mesh points in the x-direction N_x. Simulations are performed in a $[0, 2\pi] \times [0, 2\pi]$ domain. Numerical results of the wave amplitude are compared to the expected theoretic amplitude suggested by Prosperetti [63], in the case of two fluids with equal kinematic viscosity, where

$$\begin{aligned} A_{\exp}(t) = & \frac{4\left(1-4\beta\right)\nu^2 A_o}{8\left(1-4\beta\right)\nu^2+\omega_o^2}\operatorname{erfc}\left(\sqrt{\nu t}\right) \\ & + \sum_{i=1}^{4} \frac{z_i \omega_o^2 A_o}{Z_i\left(z_i^2-\nu\right)} \exp\left[\left(z_i^2-\nu\right)t\right]\operatorname{erfc}\left(z_i\sqrt{t}\right) \end{aligned} \tag{3.40}$$

and z_i are the roots of

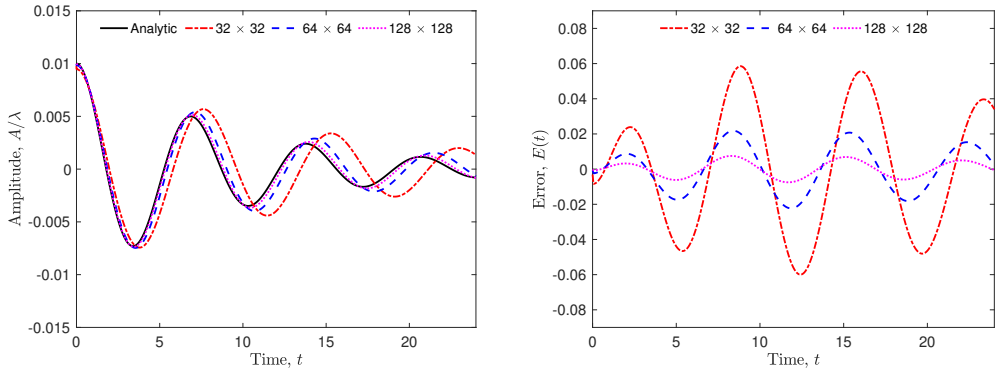
$$z^4 - 4\beta\sqrt{\nu}z^3 + 2\left(1-6\beta\right)\nu z^2 + 4\left(1-3\beta\right)\nu^{3/2}z + \left(1-4\beta\right)\nu^2 + \omega_o^2 = 0. \tag{3.41}$$

The inviscid oscillation frequency is given by $\omega_o = \sqrt{\sigma/\left(\rho_1+\rho_2\right)}$, with parameter $\beta = \rho_1\rho_2/\left(\rho_1+\rho_2\right)^2$, and $Z_i = \prod_{j=1, j\neq i}^{4}\left(z_j - z_i\right)$. The amplitude of the numerical model is found by monitoring the height of the wave which is located at the center of the domain.

Deterministic Simulations As discussed by Hermann [29], two deterministic cases are explored for a surface tension of $\sigma = 2$. In the first case we use densities of $\rho_1 = \rho_2 = 1$ with a viscosity of $\nu = 0.064720863$. In the second case a density ratio of 1000 is used, with $\rho_1 = 1000$ and $\rho_2 = 1$, both fluids have a viscosity of $\nu = 0.0064720863$. In all cases the DDPCM method is used with $N_c = 4$ and $N_P = 5$.

Results of the case of density ratio one are shown in Fig. 3.12, which displays a convergence of results with mesh refinement. Additionally, the error of the simulation is shown, where $E(t) = (A(t) - A_{\exp})/A_o$, with good agreement. It is useful to point

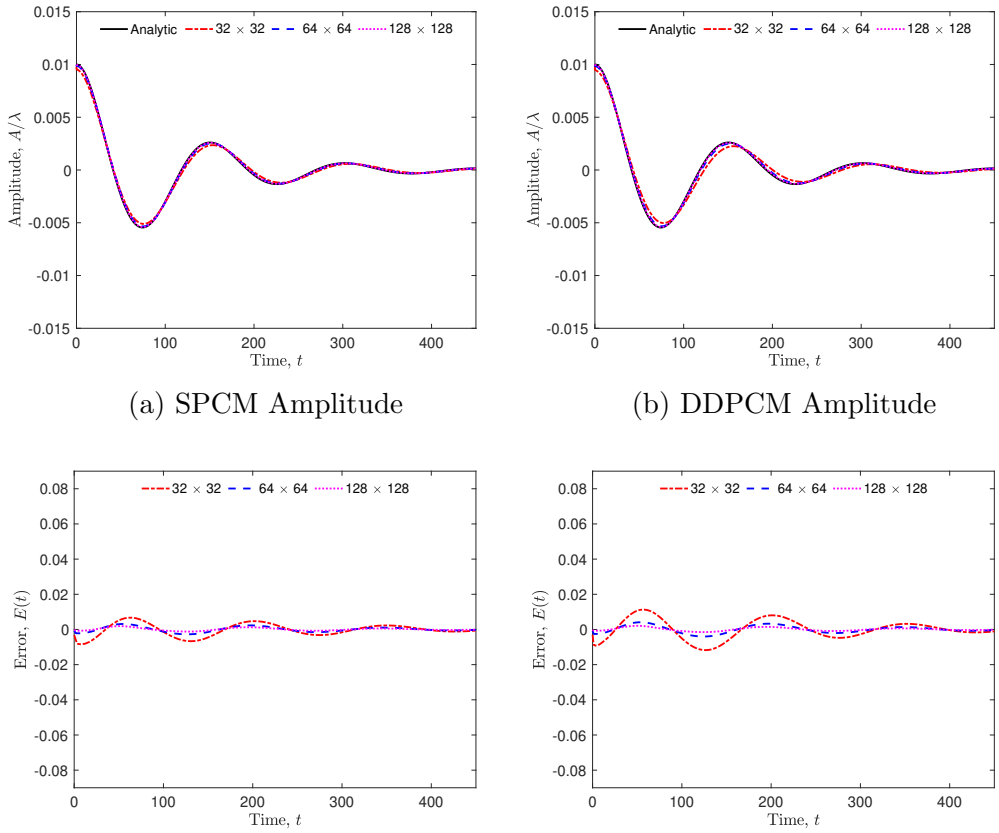
out that in this simulation, where the two fluids have equal density, the use of the DDPCM introduces very minor errors into the model as $\rho_o = \rho_1 = \rho_2$. Because of this we can quickly find $\hat{P} = P^{n+1}$, through iterative convergence. In fact, it was found that during the simulation, convergence was reached after 2 pressure iterations.



(a) Amplitude (b) Numerical Error

Figure 3.12: Amplitude of a damped surface wave with a density ratio of $\rho_1/\rho_2 = 1$ for three different mesh sizes. Analytic solution described by Prosperetti [63] shown with the solid black line.

In the case of a high density ratio, where $\rho_1/\rho_2 = 1000$, results are shown in Fig. 3.13. In this simulation we see even better agreement with analytic results than in the previous case. With mesh refinement, the solution rapidly converges to that of equation 3.40. Looking also at the comparison to the SPCM, we see they are somewhat different as the SPCM has a maximum error of $E(t) \approx 0.0085$, while for the DDPCM $E(t) \approx 0.0117$ for the 32×32 mesh. Of course, this solution difference could be minimized further by increasing the number of iterations. But again, the trade-off is at a greater computational expense. It is likely more useful to refine mesh for finer resolution given the convergence offerings (i.e. $E(t) \approx 0.0021$ for the 128×128 mesh with DDPCM).



(a) SPCM Amplitude

(b) DDPCM Amplitude

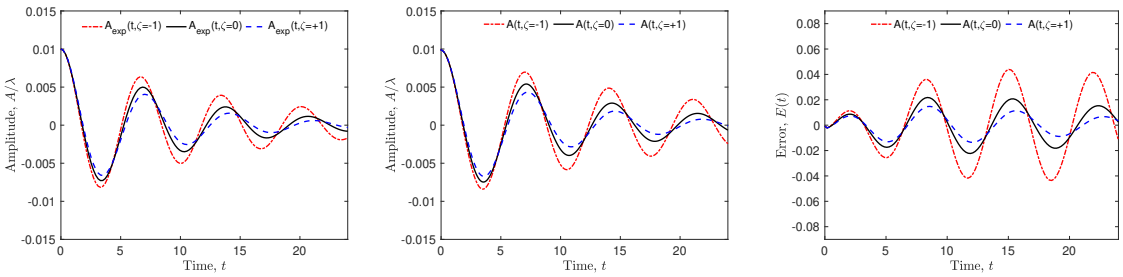
(c) SPCM Error

(d) DDPCM Error

Figure 3.13: Amplitude of a damped surface wave with a density ratio of $\rho_1/\rho_2 = 1000$ for three different mesh sizes with both pressure correction methods. Iteration counts for DDPCM solutions are $N_c = 4$ and $N_P = 5$. Analytic solution described by Prosperetti [63] shown with the solid black line.

Stochastic Surface Wave The purpose of the damped surface wave case is to test the ability of the solver to accurately predict situations where viscosity and surface tension forces interact. To test the ability of the stochastic solver to handle this interaction with uncertainty present, we will now assume there is some uncertainty about the viscosity of the fluids. As the analytic solution assumes equal viscosity fluids, we will maintain that assumption, but impose a uniform distribution about these fluids. Again we will look at both an equal and high density ratio test case to determine the ability of the solver. For the equal density case we now have kinematic viscosity $\nu = 0.064720863 + 0.0323604315\zeta$. In the case of the high density ratio we have $\nu = 0.0064720863 + 0.00323604315\zeta$.

Fig. 3.14 displays the expected and numerical results of the stochastic equal density case. The simulation was run with order $N = 5$ given the low amount of movement about the interface. For reference, the case of $A(t, \zeta = 0)$ is the same case presented in the deterministic section, where viscosity $\nu(\zeta = 0) = 0.064720863$. For this particular solution, we see very similar results to those found by the deterministic model. Note, all amplitude values are found during the course of a single simulation.

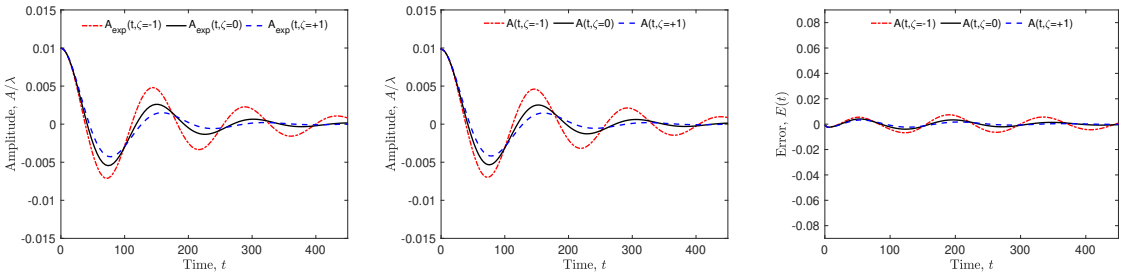


(a) Analytic Amplitude (b) Numerical Amplitude (c) Error

Figure 3.14: Amplitude of a damped surface wave with a density ratio of $\rho_1/\rho_2 = 1$ for a 64×64 mesh with $N_c = 4$, $N_P = 5$, and $N = 5$. Analytic solutions for 3 possible viscosities as described by Prosperetti [63] shown at left.

Applying an uncertain viscosity to the case of the high density ratio ($\rho_1/\rho_2 =$

1000) we find similar behavior to that shown in the deterministic model runs. Fig. 3.15 displays expected and numerical results for a UQ case with $N_c = 4$, $N_P = 5$, and $N = 5$. As with the deterministic solutions, the DDPCM methodology appears to be more closely aligned to analytic results than the unity density ratio case. Additionally, it appears the case of the largest viscosity has the smallest deviation from analytic expectations.



(a) Analytic Amplitude (b) Numerical Amplitude (c) Error

Figure 3.15: Amplitude of a damped surface wave with a density ratio of $\rho_1/\rho_2 = 1000$ for a 64×64 mesh with $N_c = 4$, $N_P = 5$, and $N = 5$. Analytic solutions for 3 possible viscosities as described by Prosperetti [63] shown at left.

Atomizing Jet

To highlight the ability of the proposed solver to solve complex and potentially real-world situations, we present the results of 2-D deterministic and stochastic atomizing jets. We compare the velocity field found with the DDPCM to that of the standard method, as well as the difference between the pressure field found by each. Given the desire to create a more efficient multiphase UQ approach, it is also important to determine the computational cost difference between the two methods.

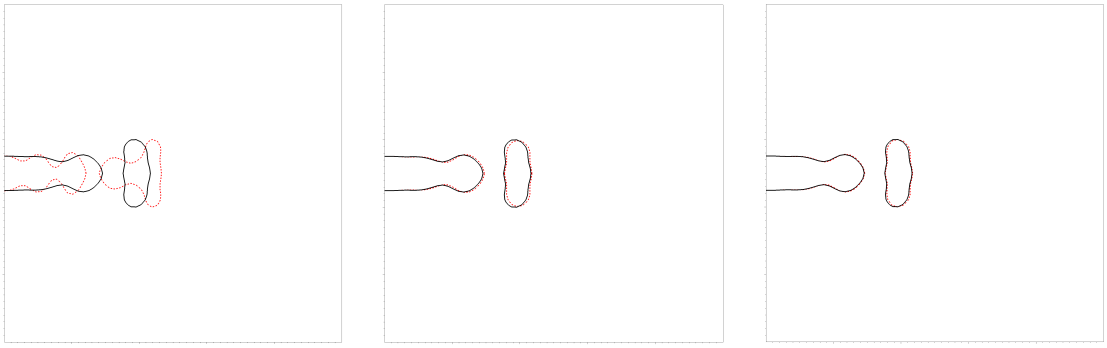
For the deterministic jet, the fluid characteristics include kinematic viscosities of $\nu_1 = 0.01$ $^{\text{cm}^2}/_{\text{s}}$ and $\nu_2 = 0.15$ $^{\text{cm}^2}/_{\text{s}}$, densities of $\rho_1 = 1.0$ $^{\text{g}}/_{\text{cm}^3}$ and $\rho_2 = 0.001$ $^{\text{g}}/_{\text{cm}^3}$, a surface tension coefficient of $\sigma = 72.8$ $^{\text{g}}/_{\text{s}^2}$, and an incoming velocity of 1000 $^{\text{cm}}/_{\text{s}}$

occurring in a 1 cm^2 domain. Given the incoming diameter of the inlet tube is $D_H = 0.1$ cm, the incoming Reynold's number is $\text{Re}_D = 1 \times 10^4$.

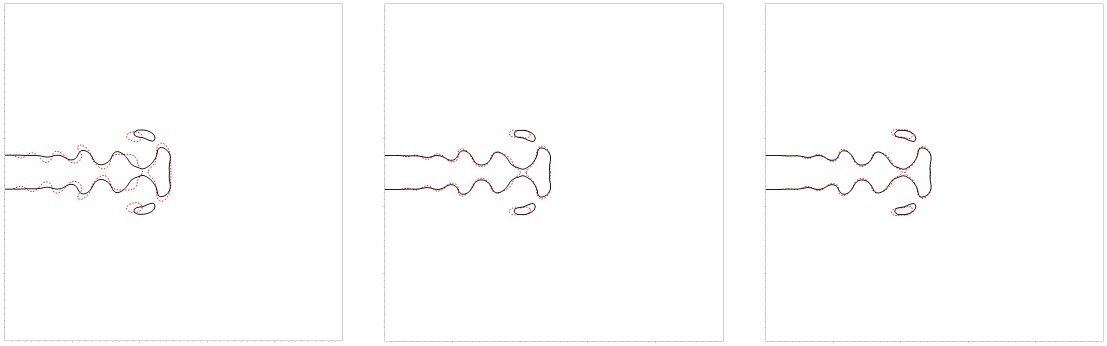
Comparison to Standard Pressure Correction We first compare the interface locations of the two methods through time for a 2-D atomizing jet. As previously mentioned, a reasonable method for improving the estimate of the pressure field $\hat{P}$ is to iterate over the pressure Poisson equation. For this scenario, we have set CFL$= 0.75$ to maintain stability. As shown in Fig. 3.16, the location of the interface can be significantly different when $\hat{P} \neq P^{n+1}$. However, as this quantity is improved by iteration, the interface location converges to the traditional pressure correction method. We can see that not only does improving the quantity $\hat{P}$ converge the solutions, but also refinement of the mesh.

Next, utilizing the most efficient combination of $N_c = 4$ and $N_P = 5$ suggested previously, we look at a direct comparison of the difference between the velocity fields of the two pressure solution methods, where $\boldsymbol{u}_{\text{diff}} = \boldsymbol{u}_{\text{SPCM}} - \boldsymbol{u}_{\text{DDPCM}}$ in a deterministic simulation. Figure 3.17 displays the difference in the velocity fields between SPCM and DDPCM. Peak values occur in the coarsest mesh at a few cell locations, with a difference of $\approx 1\%$. This error diminishes as the mesh is refined, as shown in the figure.

Stochastic Jet Of course, since the goal of utilizing the DDPCM is to improve the efficiency of stochastic models, we now look at a case involving uncertainty. To showcase the ability of the method to resolve a system with a great deal of uncertainty, we present a case with uncertainty about the velocity of an incoming jet. For simplicity, we utilize again a uniform distribution, where $u_{\text{in}} = 1000 + 100\zeta$ cm/s, or a variation of $\pm 10\%$. All other fluid parameters remain the same.

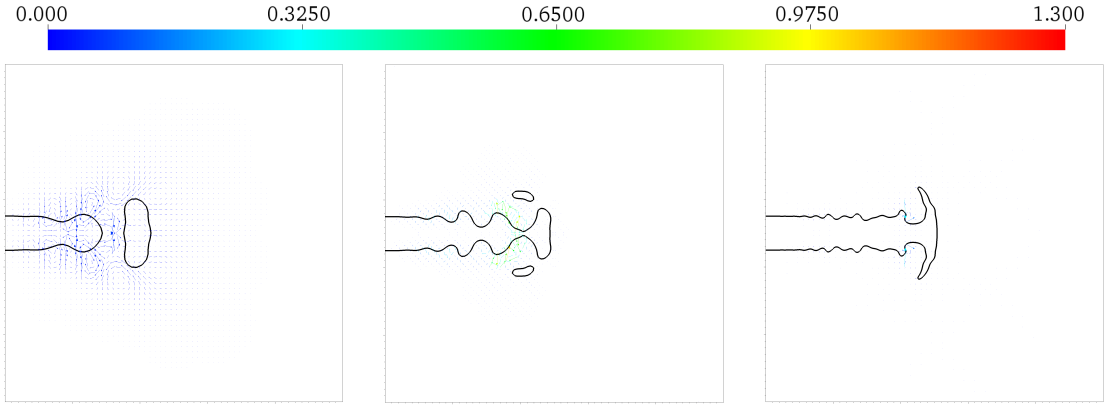


(a) $64 \times 64, N_P = 1$ (b) $64 \times 64, N_P = 3$ (c) $64 \times 64, N_P = 5$



(d) $128 \times 128, N_P = 1$ (e) $128 \times 128, N_P = 3$ (f) $128 \times 128, N_P = 5$

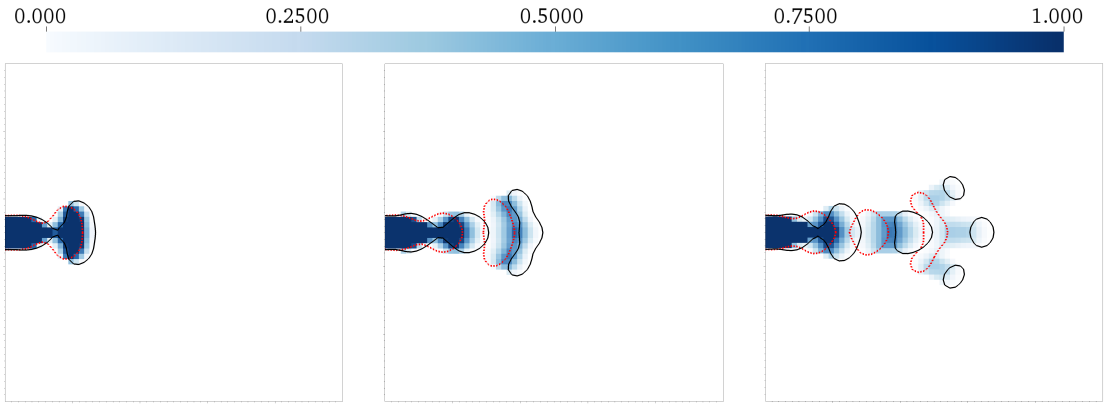
Figure 3.16: Interface location as a function of the number of pressure iterations for 2 different mesh sizes of a 2-D atomizing jet at time $t = 0.0005$ s. Black and red lines indicate the interface location for the standard and density decoupled correction methods, respectively. All simulations are run with $N_c = 2$.



(a) 64×64 (b) 128×128 (c) 256×256

Figure 3.17: Velocity field difference, $\boldsymbol{u}_{\text{diff}}$, between traditional and proposed pressure solver for 3 different mesh sizes of a 2-D atomizing jet at time $t = 0.0005$ s.

Fig. 3.18 depicts the range of solutions given this degree of uncertainty about the incoming fluid. We also see the results of two deterministic solutions for comparison. As shown, the effect of only a 10% variation in fluid velocity results in a rather dramatic difference in the possible solutions. Also shown is the probability of being in the liquid phase. With this map we can see regions where the fluid phase is known, and others where the interface is uncertain.



(a) $t = 0.0003$ s (b) $t = 0.0005$ s (c) $t = 0.0007$ s

(d) $t = 0.0003$ s (e) $t = 0.0005$ s (f) $t = 0.0007$ s

Figure 3.18: At top is the solution of a stochastic atomizing jet with uncertainty about incoming velocity on a 64^2 grid over time. Bounding solutions are shown for fastest (black line) and slowest (red dashed) incoming velocities. Color map indicates the probability of being in the liquid phase. At bottom are results of two deterministic simulations for comparison.

Conclusions

We have presented a modified pressure correction method, extending the density decoupled approach of Dodd & Ferrante [17] to an intrusive stochastic multiphase solver. Deviations from the standard pressure correction method arise due to differences in projected pressure $\hat{P}$ and the actual pressure P^{n+1}. We attempted to counter and reduce these deviations by imposing a semi-Lagrangian extrapolation method for a better initial estimate of $\hat{P}$, which is further improved by iterating over the pressure solver. Results from an oscillating droplet, surface wave, and atomizing jet show convergence of the density decoupled approach to that of the standard pressure correction method.

Additionally, given convergence of the DDPCM, an efficient combination of pressure and Crank-Nicolson iterations was found which significantly reduces computational cost over the SPCM, with little added error. Due to the iterative approach to convergence, some deviation threshold must be decided to limit the number of iterations taken. We then balance accuracy and computational cost, which is typical of numerical solutions in any system of differential equations. Possibilities are offered which significantly speed up simulation times while converging deviations from the traditional pressure correction as mesh is refined.

Certainly the biggest reward of this methodology is for application to stochastic systems. Where the coupled nature of density in the pressure Poisson equation caused $N + 1$ coupled equations for solution in stochastic problems, the decoupling allows for the use of more sophisticated linear solvers. Since most of the numerical research exists for application to deterministic systems, the ability to utilize these methods for intrusive UQ is extremely beneficial. As a result, we see significant reductions in simulation cost as shown in Figs. 3.6 and 3.11 as the number of basis functions and

uncertainty increases, with a speed up factor of ≈ 75 for a 256^2 mesh and ≈ 20 with a basis order of $N = 9$.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Nos. 1511325 and 1749779. Computational efforts were performed on the Hyalite High-Performance Computing System, operated and supported by University Information Technology Research Cyberinfrastructure at Montana State University.

Appendix

Efficiency Timing

Efficiency timing was run with the oscillating droplet test on a 64^2 mesh over a range of Crank-Nicolson and pressure iterations. A single run of the SPCM was used to test against for average kinetic energy error, $\bar{E}_{\text{diff}}$. All tests were run out to 3 oscillations for tracking of numerical dissipation. Total pressure steps taken throughout the simulation were also tracked, as well as the average number of iterations per time step taken to solve the pressure Poisson equation. The results of the efficiency study are outlined in Fig. 3.2 below.

Additionally, the best possible simulations, as identified in Fig. 3.2 are plotted in Fig. 3.19. To avoid confusion, three are shown in each of the two sub-figures. As shown, numerical dissipation is greater in simulations with lower N_c, leading to the choice of $N_c = 4$ and $N_P = 5$ as the "best" combination.

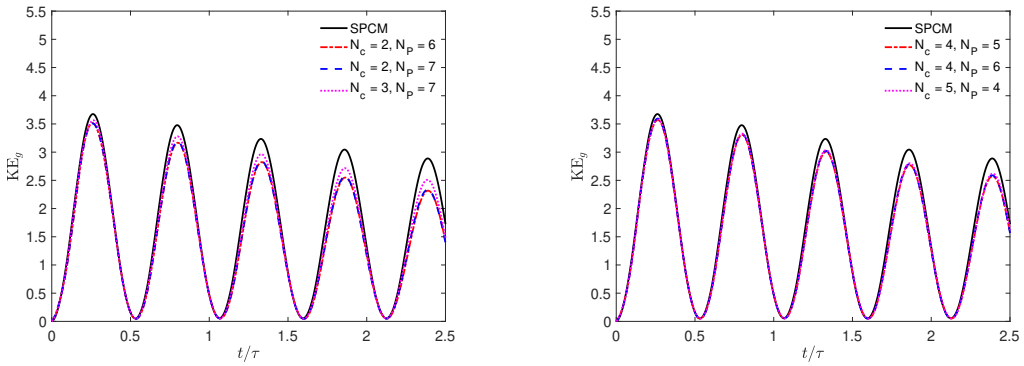


Figure 3.19: Comparison of solutions for several possible combinations of N_c and N_P of the oscillating droplet test. Simulations run with a density ratio of 1000 on a 64^2 mesh.

N_c	N_P	Time (min)	Pressure Iter.	Avg. Iter./Step	Error ($\bar{E}_{\text{diff}}$)
2	1	0.299	45856	61	0.393
3	1	0.397	69007	92	0.268
4	1	0.525	92069	123	0.225
5	1	0.654	115125	154	0.203
6	1	0.752	138192	184	0.191
2	2	0.418	92102	123	0.158
3	2	0.563	138255	184	0.143
4	2	0.711	184537	246	0.133
5	2	0.872	230883	308	0.127
6	2	1.049	277201	370	0.121
2	3	0.570	138331	184	0.102
3	3	0.772	207746	277	0.095
4	3	1.001	277336	370	0.090
5	3	1.271	346942	463	0.178
6	3	1.505	416297	555	0.119
2	4	0.639	184693	246	0.072
3	4	0.913	277431	370	0.150
4	4	1.237	370274	494	0.102
5	4	1.545	462736	617	0.083
6	4	1.775	555209	740	0.073
2	5	0.794	231088	308	0.088
3	5	1.130	347151	463	0.072
4	5	1.488	462981	617	0.063
5	5	1.818	578461	771	0.057
6	5	2.138	694219	926	0.052
2	6	0.886	277585	370	0.055
3	6	1.266	416662	556	0.049
4	6	1.663	555514	741	0.044
5	6	2.002	694295	926	0.040
6	6	2.421	832914	1111	0.103
2	7	1.175	324086	432	0.039
3	7	1.643	486258	648	0.035
4	7	1.903	648188	864	0.094
5	7	2.296	809901	1080	0.062
6	7	2.682	971399	1295	0.050

Table 3.2: Simulation run times and associated metrics for finding the best combination of N_c and N_P. Highlighted rows indicate best possibilities.

MULTIUQ: A SOFTWARE PACKAGE FOR UNCERTAINTY QUANTIFICATION OF MULTIPHASE FLOWS

Contribution of Authors and Co-Authors

Manuscript in following chapter

Author: Brian Turnquist

Contributions: primary author, performed research, code development

Author: Mark Owkes

Contributions: secondary author, review and feedback, troubleshooting

Manuscript Information

Brian Turnquist and Mark Owkes

Computer Physics Communications

Status of Manuscript:
_X_ Prepared for submission to a peer-reviewed journal
____ Officially submitted to a peer-reviewed journal
____ Accepted by a peer-reviewed journal
____ Published in a peer-reviewed journal

Elsevier, Inc.

Abstract

multiUQ is a novel tool that simulates gas-liquid multiphase flows and quantifies uncertainty in results due to variability about fluid properties and initial/boundary conditions. The benefit over a typical deterministic solver is that inexact information, such as variability in fluid properties or flow rates, can be included to determine the affect on simulation solutions. The proposed tool, multiUQ, uses an intrusive method wherein variables of interest are functions of space, time, and some uncertainty dimension. This differs from non-intrusive methods which utilize many solutions from a deterministic solver to generate a distribution of possible results. Alternatively, the intrusive solver is run once, giving an infinite number of solutions as an output, as well as desired statistics. Given the many applications of multiphase flows, including open flows, hydraulics, fuel injection systems, and atomizing jets, there is a massive potential benefit to reducing the computational cost of uncertainty quantification of these flows.

Introduction

Given the growth in capacity and efficiency of computational resources over the past several decades, physics based models of fluid dynamics and other systems have become commonplace. Worldwide, many groups are creating their own in house software, developing new and more robust techniques to accurately solve Navier-Stokes for a variety of applications (e.g. [3, 8, 21, 70]). With improved computational resources and software, high-fidelity simulations help us better understand variable-density multiphase flows. This has aided in a myriad of engineering improvements, including direct injection systems for combustion engines [5], rotary bell cup atomization for paint application [61], and fire sprinkler spray efficiency [51].

While methods for multiphase flow simulations have been steadily improving, a method gap has existed in uncertainty quantification (UQ). Until recently, little effort has been placed on UQ of fluid dynamics in general, with less on UQ of multiphase flows. In an attempt to fill this gap and also to encourage further development in this realm, multiUQ has been created as an open source application. multiUQ is a tool which allows for predicting fluid systems with uncertainty about each fluid parameter (i.e. density, viscosity, surface tension coefficient), initial and boundary conditions for the flow field, and the location of the interface between phases. It is then possible to output extreme solutions, average solutions, or variances about the average for each variable.

For a computationally efficient approach to UQ of multiphase flows, several approaches may be considered. Broadly speaking, UQ can be categorized as either non-intrusive or intrusive schemes. Non-intrusive schemes include Monte-Carlo [50], collocation approaches [76], or non-intrusive polynomial chaos (PC). For these methods, a deterministic solver is run many times with a distribution of inputs, with statistics computed from the many saved solutions.

In a different approach, intrusive methods incorporate stochastic variables that are a function of uncertainty. These variables typically take the form of a polynomial chaos [83] or Karhunen-Loeve [35, 45] expansion. These expanded variables operate as a function of some uncertainty domain, $\boldsymbol{\zeta}$, as well as time and space. They are substituted into a set of governing equations, requiring major code development from the ground up. Once this software has been developed, uncertainty is added as a model input, and an infinite number of solutions are generated with a single simulation. An example application is shown in Fig. 4.1, where there is uncertainty about the magnitude of an incoming jet (i.e. a boundary condition) resulting in uncertainty about the solution of the jet at a given time (i.e. stochastic output).

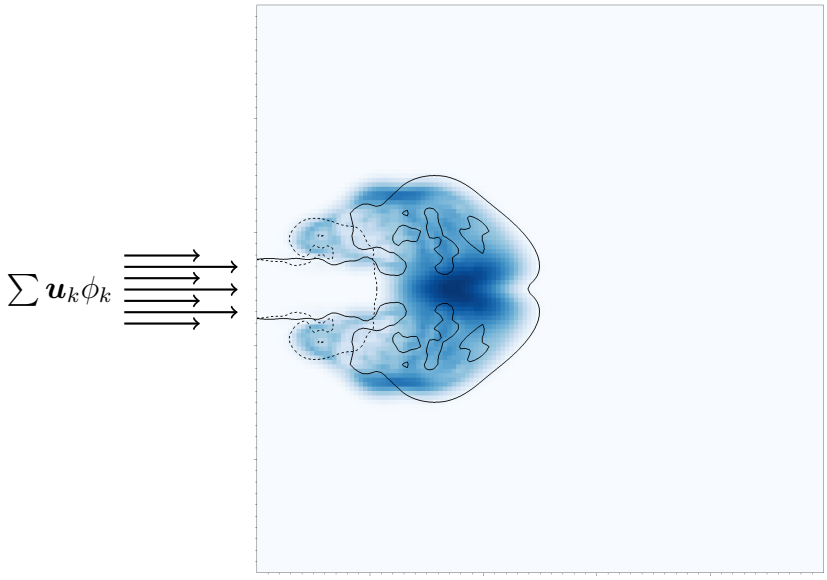


Figure 4.1: Example simulation from multiUQ, an intrusive stochastic multiphase flow solver. A single simulation has been performed of a jet with uncertainty about the incoming velocity magnitude. The dotted line indicates the interface boundary of one solution at a low velocity, while the solid line indicates the solution of another at a high velocity. The color map indicates the variance about the interface location.

multiUQ is the first intrusive stochastic multiphase flow solver capable of handling uncertainty about systems with high density ratios in three dimensions. As a tool, it provides the ability to understand the effect of uncertainty about initial and boundary conditions as well as fluid parameters on simulation solutions. Being able to quantify the uncertainty caused by these variations may allow for better understanding of a modeled system. This article seeks to provide a general overview of how multiUQ operates. As shown in Fig. 4.2, multiUQ takes stochastic inputs, runs them through a procedure which discretely steps forward in time, and outputs stochastic results that can be visualized. To get a more detailed idea of how it works we'll first provide the mathematical basis of the software, then describe the software mesh, discretization, and where to find the code and compile it. We then cover

simulation capabilities and test cases that are included, how to run the code, and finally discuss the performance and scalability with some final closing remarks.

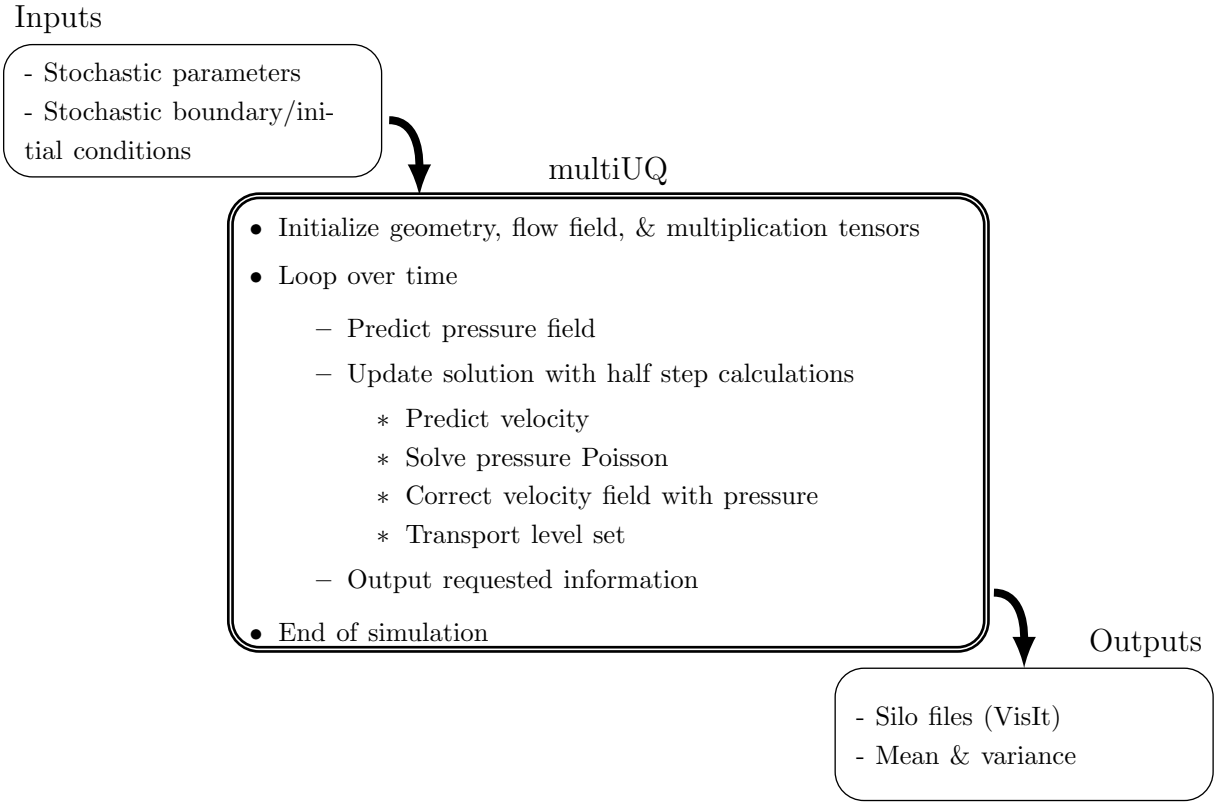


Figure 4.2: Visual flow of multiUQ software.

Mathematical development

This section provides a summary of the governing equations and representation of stochastic variables. Additional details are available in [81].

Governing equations

multiUQ uses the one-fluid approach for solution of the incompressible Navier-Stokes equations, where fluid motion is governed by

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\eta \nabla P + \eta \nabla \cdot \left[\mu \left(\nabla \boldsymbol{u} + \nabla^{\mathrm{T}} \boldsymbol{u} \right) \right] + \eta \boldsymbol{f}_{\sigma} \delta_s \tag{4.1}$$

for velocity $\boldsymbol{u}$, specific volume $\eta = 1/\rho$ (for density ρ), pressure P, and surface tension force $\boldsymbol{f}_{\sigma}$ denoted only at the interface as indicated by Dirac delta δ_s. Additionally, in an incompressible framework, mass is conserved with the continuity equation

$$\nabla \cdot \boldsymbol{u} = 0. \tag{4.2}$$

These are the equations of motion for a deterministic, incompressible system of fluids. To allow for uncertainty to exist about these equations, stochastic variables are used, where for instance some variable ψ is

$$\psi(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \sum_{k=0}^{N} \psi_k(\boldsymbol{x}, t) \phi_k(\boldsymbol{\zeta}) = \psi_k \phi_k, \tag{4.3}$$

which is a polynomial chaos expansion of a variable that is a function of spatial dimensions $\boldsymbol{x}$ and time t, projected onto a series of $N+1$ polynomial basis functions ϕ_k that are a function of uncertainty dimensions $\boldsymbol{\zeta}$. As shown and moving forward, we will utilize Einstein notation, emphasizing the summation over repeated indices.

Allowing for uncertainty about all variables in equation 4.1 (except time), we substitute stochastic velocity $\boldsymbol{u}_k \phi_k$, specific volume $\eta_k \phi_k$, pressure $P_k \phi_k$, viscosity $\mu_k \phi_k$ and surface tension force $\boldsymbol{f}_{\sigma:k} \phi_k$, to arrive at a form of stochastic Navier Stokes

equations

$$\frac{\partial \boldsymbol{u}_k \phi_k}{\partial t} + \boldsymbol{u}_k \phi_k \cdot \nabla \boldsymbol{u}_l \phi_l = \\ -\eta_k \phi_k \nabla P_l \phi_l + \eta_k \phi_k \nabla \cdot \left[\mu_l \phi_l \left(\nabla \boldsymbol{u}_m \phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \phi_m \right) \right] + \eta_k \phi_k \boldsymbol{f}_{\sigma:l} \phi_l \delta_s. \tag{4.4}$$

Because we wish to find the basis weights $\boldsymbol{u}_k$, we now utilize the property of orthogonality. This property depends on which basis functions are used. In multiUQ, we are currently utilizing Legendre polynomials, where the property of orthogonality is

$$\int_{-1}^{1} \phi_k \phi_b \delta\zeta = \begin{cases} \langle \phi_k \phi_b \rangle & k = b \\ 0 & k \neq b \end{cases}. \tag{4.5}$$

To utilize the property of orthogonality in equation 4.5, we first multiply equation 4.4 by a test function ϕ_b, giving us

$$\frac{\partial \boldsymbol{u}_k \phi_k}{\partial t} \phi_b + \boldsymbol{u}_k \phi_k \cdot \nabla \boldsymbol{u}_l \phi_l \phi_b = -\eta_k \phi_k \nabla P_l \phi_l \phi_b \\ + \eta_k \phi_k \nabla \cdot \left[\mu_l \phi_l \left(\nabla \boldsymbol{u}_m \phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \phi_m \right) \right] \phi_b + \eta_k \phi_k \boldsymbol{f}_{\sigma:l} \phi_l \delta_s \phi_b. \tag{4.6}$$

We then integrate over the region of orthogonality (i.e. $\boldsymbol{\zeta} \in [-1, 1]$ for Legendre polynomials)

$$\int_{-1}^{1} \frac{\partial \boldsymbol{u}_k \phi_k}{\partial t} \phi_b + \boldsymbol{u}_k \phi_k \cdot \nabla \boldsymbol{u}_l \phi_l \phi_b d\boldsymbol{\zeta} = \int_{-1}^{1} -\eta_k \phi_k \nabla P_l \phi_l \phi_b \\ + \eta_k \phi_k \nabla \cdot \left[\mu_l \phi_l \left(\nabla \boldsymbol{u}_m \phi_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m \phi_m \right) \right] \phi_b + \eta_k \phi_k \boldsymbol{f}_{\sigma:l} \phi_l \delta_s \phi_b d\boldsymbol{\zeta}. \tag{4.7}$$

Leveraging the orthogonality property, $\int_{-1}^{1} \phi_b \phi_b d\zeta$ can be factored out of the first term and divided out. This leads to the stochastic Navier-Stokes equations in a form

that can be solved to find the stochastic velocity weights $\boldsymbol{u}_b$

$$\begin{aligned}\frac{\partial \boldsymbol{u}_b}{\partial t} + \boldsymbol{u}_k \cdot \nabla \boldsymbol{u}_l C_{klb} &= -\eta_k \nabla P_l C_{klb} \\ &+ \eta_k \nabla \cdot \left[\mu_l \left(\nabla \boldsymbol{u}_m + \nabla^{\mathrm{T}} \boldsymbol{u}_m\right)\right] C_{klmb} + \eta_k \boldsymbol{f}_{\sigma:l} \delta_s C_{klb}\end{aligned} \tag{4.8}$$

for $b = 0, \ldots, N$, where

$$C_{klb} = \frac{\int_{-1}^{1} \phi_k \phi_l \phi_b d\zeta}{\int_{-1}^{1} \phi_b \phi_b d\zeta} = \frac{\langle \phi_k \phi_l \phi_b \rangle}{\langle \phi_b \phi_b \rangle} \tag{4.9}$$

and

$$C_{klmb} = \frac{\langle \phi_k \phi_l \phi_m \phi_b \rangle}{\langle \phi_b \phi_b \rangle}. \tag{4.10}$$

Additionally, the stochastic continuity equations are

$$\nabla \cdot \boldsymbol{u}_b = 0 \tag{4.11}$$

for $b = 0, \ldots, N$, which closes the incompressible system. Next, we define the location of the interface between phases.

Interface capturing

Given multiUQ is capable of both incompressible single- and multi-phase flows, we need to define the method for determining the interface. While there exist both interface tracking and interface capturing schemes, an interface capturing scheme like the level set approach [57] most easily interfaces with the stochastic PC design. More specifically, to capture the interface between phases a conservative level set approach is utilized, as outlined in [81] and following the work of [15, 54]. Transport of the

interface is accomplished by

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\psi \boldsymbol{u}) = 0, \tag{4.12}$$

for level set ψ and time t. The level set profile takes on the form of a hyperbolic tangent, which is initialized by a signed-distance function $g(\boldsymbol{x}, t)$ by

$$\psi(\boldsymbol{x}, t) = \left[1 + \exp\left(\frac{-2g(\boldsymbol{x}, t)}{\varepsilon_1 + \varepsilon_2}\right)\right]^{-1} \tag{4.13}$$

Values ε_1 and ε_2 control the sharpness of the interface. We set the values to $\varepsilon_1 = 9\text{max(dx,dy,dz)}/8$ and $\varepsilon_2 = \text{max(dx,dy,dz)}/8$, which provides seven grid cells to represent the hyperbolic tangent profile. These values may be reduced or increased to sharpen or spread the interface profile.

Expanding the level set transport equation to include uncertain variables is done by substituting $\psi_k \phi_k$ and $\boldsymbol{u}_k \phi_k$ into equation 4.12. We then multiply through by a test function ϕ_b and integrate over $\boldsymbol{\zeta} \in [-1, 1]$ to get

$$\frac{\partial \psi_b}{\partial \tau} + \nabla \cdot (\psi_k \boldsymbol{u}_l)\, C_{klb} = 0. \tag{4.14}$$

Similarly, equation 4.13 is generalized to include a stochastic interface position through a stochastic signed-distance function $g_k \phi_k$. Substituting in the stochastic signed-distance function, multiplying by a test function ϕ_b, integrating over $\boldsymbol{\zeta} \in [-1, 1]$, and solving for ψ_b leads to

$$\psi_b = \frac{1}{\langle \phi_b \phi_b \rangle} \int_{-1}^{1} \left[1 + \exp\left(\frac{-2 g_k \phi_k}{\varepsilon_1 + \varepsilon_2}\right)\right]^{-1} \phi_b d\boldsymbol{\zeta}. \tag{4.15}$$

Unfortunately, this equation has no analytic solution, and thus requires the use of

some numerical integration for calculation. multiUQ has both a Gaussian quadrature and a Romberg integration with Richardson extrapolation for the initialization of such profiles. While Romberg integration converges to machine precision, given the potentially high cost, it can be useful to compute an estimate with a Gaussian quadrature.

When transporting the conservative level set there is no guarantee the hyperbolic interface profile will be maintained. Due to this issue, a reinitialization procedure must be performed. As presented in [81], reinitialization is accomplished by

$$\frac{\partial \psi}{\partial \tau} + \nabla \cdot \left[\psi \left(1 - \psi \right) \boldsymbol{r} \right] = \nabla \cdot \left[\varepsilon_1 \left(\nabla \psi \cdot \boldsymbol{r} \right) \boldsymbol{r} \right] + \nabla \cdot \left(\varepsilon_2 \nabla \psi \right) \tag{4.16}$$

where τ is pseudo-time and $\boldsymbol{r}$ is a continuous (non-unit) interface normal vector calculated by

$$\boldsymbol{r} = \frac{\nabla \psi}{|\nabla \psi|_{\max}}. \tag{4.17}$$

As shown in [81], $|\nabla \psi|_{\max} = 1/4(\varepsilon_1 + \varepsilon_2)$. The use of continuous normal vectors $\boldsymbol{r}$ is for easy implementation in the PC framework, where $\boldsymbol{r}$ is nicely projected onto orthogonal basis functions. This is different from other methodologies, such as [15, 54, 55], where unit normal vectors (i.e. $\boldsymbol{n} = \nabla \psi / |\nabla \psi|$) are used.

To calculate the basis weights ψ_b and perform a stochastic reinitialization, we substitute stochastic variables, multiply by a test function ϕ_b and integrate over $\boldsymbol{\zeta} \in [-1, 1]$, finding

$$\frac{\partial \psi_b}{\partial \tau} + \nabla \cdot (\psi_k \boldsymbol{r}_l C_{klb} - \psi_k \psi_l \boldsymbol{r}_m C_{klmb}) = \nabla \cdot [\varepsilon_1 \left(\nabla \psi_k \cdot \boldsymbol{r}_l \right) \boldsymbol{r}_m] \, C_{klmb} + \nabla \cdot (\varepsilon_2 \nabla \psi_b) \, . \tag{4.18}$$

With a satisfactory interface capturing method and method of reinitialization, we utilize the level set to calculate fluid properties μ and η. In a deterministic setting,

this is accomplished with

$$\mu = \mu_1\psi + (1-\psi)\mu_2 = \mu_2 + (\mu_1 - \mu_2)\psi \tag{4.19}$$

$$\eta = \eta_1\psi + (1-\psi)\eta_2 = \eta_2 + (\eta_1 - \eta_2)\psi. \tag{4.20}$$

Allowing for uncertainty about both fluid phases (i.e. $\mu_1(\boldsymbol{\zeta}) = \mu_{1:k}\phi_k$ and $\mu_2(\boldsymbol{\zeta}) = \mu_{2:k}\phi_k$), we calculate stochastic quantities via

$$\mu_b = \mu_{2:b} + (\mu_{1:k}\psi_l - \mu_{2:k}\psi_l)\, C_{klb} \tag{4.21}$$

$$\eta_b = \eta_{2:b} + (\eta_{1:k}\psi_l - \eta_{2:k}\psi_l)\, C_{klb}. \tag{4.22}$$

Surface tension force

Having described the mathematics of the stochastic level set implementation, we can now calculate the surface tension force near the interface. Surface tension can be written as

$$\boldsymbol{f}_\sigma \delta_s = \sigma\kappa\boldsymbol{n}\delta_s \tag{4.23}$$

for surface tension coefficient σ, curvature κ, and unit normal $\boldsymbol{n}$ about the interface. The unit normal is dependent on the interface tracking or capturing method used, and is somewhat ill-defined. A commonly used approach is the continuum surface force method, first described by Brackbill *et al.* [7], where with the conservative level set

$$\boldsymbol{f}_\sigma \delta_s \approx \sigma\kappa\nabla\psi. \tag{4.24}$$

As we are using a hyperbolic tangent profile to describe the jump in fluid properties that occurs at the interface, we also use this profile to smooth the surface tension force. The hyperbolic tangent operates as a smoothed Heaviside function (i.e. $H(\boldsymbol{x})$

$\approx \psi(\boldsymbol{x})$). As outlined by [79], we can then calculate the unit normal at the interface with $\boldsymbol{n}\delta_s = \nabla H \approx \nabla\psi$. The surface tension force is thus smoothed over the few cells that surround the $\psi = 0.5$ isosurface, where the interface is defined. The number of cells defining this width depends on the previously defined ε_1 and ε_2, or inversely $|\nabla\psi|_{\max}$.

Curvature κ is calculated with

$$\kappa = -\nabla \cdot \boldsymbol{n} \tag{4.25}$$

for unit normal $\boldsymbol{n} = \nabla\psi/|\nabla\psi|$ about the interface. In a deterministic model, we then have

$$\boldsymbol{f}_\sigma = -\sigma\left(\nabla \cdot \boldsymbol{n}\right)\nabla\psi. \tag{4.26}$$

Defining the surface tension in a stochastic PC regime, we then substitute stochastic variables, multiply by a test function ϕ_b, and integrate over $\boldsymbol{\zeta} \in [-1, 1]$ to find

$$\boldsymbol{f}_{\sigma:b} = \sigma_k \kappa_l \nabla\psi_m C_{klmb}. \tag{4.27}$$

There is some consideration here required for the calculation of stochastic curvature. Given we require a unit normal at the interface, it is necessary for the magnitude requirement to hold. Rather than project $\boldsymbol{n}$ onto a system of basis functions with a quadrature (as no analytic solution exists), we instead calculate a unit normal at each quadrature point. To project a stochastic curvature onto a system of basis functions we use

$$\kappa_b = \frac{1}{\langle\phi_b\phi_b\rangle}\sum_{n=1}^{N_q} -w_n \nabla \cdot \frac{\nabla\psi_k(\zeta_n)\phi_k(\zeta_n)}{|\nabla\psi_l(\zeta_n)\phi_l(\zeta_n)|} \tag{4.28}$$

for N_q quadrature points with weights w_n and abscissas ζ_n for $n = 1, \ldots, N_q$.

Software description

multiUQ is written in modern FORTRAN, and can be compiled by GNU or Intel compilers. Parallelization of the code for computation on multiple processors is accomplished by use of the Message Passing Interface (MPI) library. A Linux or Unix terminal is currently required for compiling and running and the code.

Computational mesh and parallelization

multiUQ is represented on a rectangular computational domain with a structured Cartesian mesh. Scalar values S, such as pressure P, level set ψ, density ρ, and viscosity μ are held at the cell center. Vector components of velocity $\boldsymbol{u}$, surface tension $\boldsymbol{f}_\sigma$, and continuous normal vector $\boldsymbol{r}$ are held at the cell walls, as shown in Figure 4.3 for two dimensions. Where values of vector or scalar components are needed at half step locations, these values are linearly averaged for necessary computations.

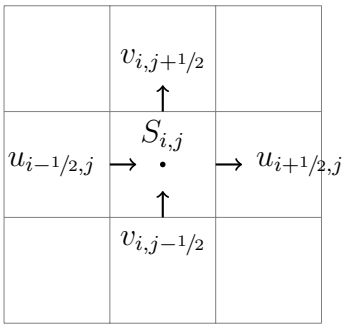


Figure 4.3: Schematic of staggered grid used for the stochastic multiphase solver.

Parallelization is achieved by decomposing the computational domain. Processor workload is divided spatially, utilizing the structured Cartesian staggered grid on which the computational domain is situated. Users may specify the processor division in the x, y, and z directions, depending on the extent of the domain in each. The

stochastic dimension(s) $\boldsymbol{\zeta}$ are not decomposed in the current implementation. Inter-processor communication is achieved by defining ghost cells at the edge of each processor. The ghost cells provide memory to store information communicated using MPI from the neighboring processor. At the edge of the spatial domain, these ghost cells are used to discretize the boundary conditions.

Numerical implementation

Discretization of the governing equations (Eqs. 4.8, 4.11, 4.14, and 4.18) on the Cartesian mesh is done with the finite difference method. For the purpose of simplicity in this work, a deterministic notation is used. In practice, a derivative is found for each basis weight b.

Time marching Time marching is accomplished through an iterative Crank-Nicolson approach coupled with a pressure correction method which breaks up Navier-Stokes into two steps. The first step is initially taken with the explicit Euler method, calculating a partially discretized predicted velocity field as

$$\begin{aligned}\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} &= -\boldsymbol{u}^{n+1/2} \cdot \nabla \boldsymbol{u}^{n+1/2} \\ &\quad + \eta^{n+1/2} \nabla \cdot \left[\mu^{n+1/2} \left(\nabla \boldsymbol{u}^{n+1/2} + \nabla^{\mathrm{T}} \boldsymbol{u}^{n+1/2}\right)\right] + \eta^{n+1/2} \boldsymbol{f}_\sigma^{n+1/2}\end{aligned} \tag{4.29}$$

We calculate a midpoint velocity $\boldsymbol{u}^{n+1/2} = \frac{1}{2}(\boldsymbol{u}^n + \boldsymbol{u}^{n+1})$, using an estimate of $\boldsymbol{u}^n$ for the first step. The predicted field $\boldsymbol{u}^*$ is then corrected with a density decoupled pressure correction method as presented by [17] and [80], where

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\eta_0 \nabla P^{n+1} - \left(\eta^{n+1} - \eta_0\right) \nabla \hat{P} \tag{4.30}$$

for maximum and constant specific volume η_0 and estimated pressure field $\hat{P}$. The estimation of $\hat{P}$ is accomplished by a semi-Langrangian projection for the initial iteration, but is updated several times at each time step to reduce numerical error. This approach improves computational efficiency for increasing orders of uncertainty. Calculation of P^{n+1} is accomplished by taking the divergence of equation 4.30, resulting in

$$\nabla^2 P^{n+1} = \rho_0 \frac{\nabla \cdot \boldsymbol{u}^*}{\Delta t} - \rho_0 \nabla \cdot \left(\eta^{n+1} - \eta_0\right) \nabla \hat{P}. \tag{4.31}$$

Using a second order accurate finite difference operator, calculation of velocity derivatives is performed as

$$\left(\frac{du}{dx}\right)_{i,j,k} = \frac{u_{i+1/2,j,k} - u_{i-1/2,j,k}}{\Delta x}, \tag{4.32}$$

where i, j, k indices are located at the cell center, and half steps are located at the cell walls, i.e. $i + 1/2$ is located at the left x wall. Similar calculations are performed for v and w components. This forces a divergence free condition at each cell for calculation of the pressure $P^{n+1}_{i,j,k}$ at the cell center. Several methods for solving the pressure Poisson equation are available within multiUQ, however, utilizing the PFMG solver within the HYPRE package from Lawrence Livermore National Lab offers the quickest solution. Additionally, the updated pressure gradient at cell walls needed to correct the velocity field is

$$\left(\frac{dP}{dx}\right)_{i+1/2,j,k} = \frac{P_{i+1,j,k} - P_{i,j,k}}{\Delta x}, \tag{4.33}$$

which assures alignment of calculations where velocity components are held.

Having calculated an updated velocity, we average $\boldsymbol{u}^{n+1}$ with the previous velocity $\boldsymbol{u}^n$ (refer to Fig. 4.2 and equation 4.29). Additionally, we similarly average

the level set to calculate mid-time specific volume $\eta^{n+1/2}$ and viscosity $\mu^{n+1/2}$. We again correct with a pressure correction step, advance the level set, and end with a 2^{nd} order accurate estimate of the solution at t^{n+1}.

Additionally, as presented in [80], we can iterate over the Crank-Nicolson scheme more than once to improve our calculation of the estimated pressure field $\hat{P}$ and thus converge to a standard pressure correction method. Not only does this reduce the error imposed by the density decoupled approach, it also improves simulation stability and allows for more accurate simulations of high density ratio systems. Our findings in [80] show that without iteration or very small time steps, a great degree of error can result from poor estimates of $\hat{P}$. Through testing, we've found that 4 Crank-Nicolson and 5 pressure iterations reduce this error, while the computational cost improves as uncertainty and mesh are scaled up.

Level set transport As mentioned previously, transport of the interface is governed by equation 4.12. We attempt to globally conserve fluid mass by utilizing a finite volume operator across the cell. For example, transport of the level set is discretized for the x direction as

$$\nabla \cdot (\psi u) = \frac{\psi_{i+1/2,j,k} u_{i+1/2,j,k} - \psi_{i-1/2,j,k}, u_{i-1/2,j,k}}{\Delta x} \tag{4.34}$$

resulting in a calculation of ψ at the cell center. Similar gradients are calculated for the y, and z dimensions. Additionally, upwinding is necessary for accurate scalar transport. We utilize a high order upwind central (HUOC-5) scheme, as described in [15], to interpolate values of ψ to the cell walls.

Stepping forward in time for the level set transport is done in the same manner as the velocity predictions. We utilize a half step location $\psi^{n+1/2} = \frac{1}{2}(\psi^n + \psi^{n+1})$ for calculation of the next time, estimating with ψ^n for the first iteration. Subsequent

iterations improve the estimate. Calculating the gradients at the half-step, we then update the level set with

$$\psi^{n+1} = \psi^n + \Delta t \left[\nabla \cdot \left(\psi^{n+1/2} \boldsymbol{u}^{n+1/2} \right) \right], \tag{4.35}$$

having calculated the half step information as a simple average of the current and next steps. This occurs for some user defined number of maximum iterations (or potentially to some convergence tolerance).

Software functionality

The goal of this section is to minimally describe the necessary components needed to get multiUQ up and running. The source code of multiUQ is available for download through Bitbucket, the details of which are presented in Fig. 4.4. This source code contains all the required source (*.f90) files as well as an example makefile for compilation and inputs file for executing a simulation. The general thought process for code layout and function is meant to be obvious. For example, all code necessary to solving for the velocity field exists in the velocity.f90 file, while transport of the level set and reinitialization codes exist in the levelset.f90 file.

Once obtaining the source code, the next step is to compile it. Additional libraries are necessary to successfully run the code, which are needed for computations run during simulations and the output of visual data files. We next discuss the needed libraries, then how to compile the code, and finally consider setting up a simulation and executing the program.

Program Title	multiUQ
Code Availability	https://bitbucket.org/markowkes/multiuq
Licensing Provisions	GPLv3
Programming Language	Fortran 95/2003
Parallelization	OpenMPI >3.xx
Assumptions	Incompressible, low Mach, DNS
Interface Scheme	Conservative level set, continuum surface force (CSF)
Dependencies	Szip, HDF5, Silo, HYPRE, FFTW, VisIt

Figure 4.4: General information about the code.

Required libraries

Several other libraries are required to compile and run multiUQ. Minimum software versions include Szip 2.1 and HDF5 1.10 provided by The HDF Group, Silo 4.10.2 and HYPRE 2.11.2 provided by Lawrence Livermore National Lab (LLNL), FFTW 3.3.8 developed at Massachusetts Institute of Technology, and OpenMPI 3 or above.

Calculation of the pressure field by solution of the pressure Poisson equation is accomplished with either the HYPRE package or one of the built-in solvers. Several different linear solvers are available within the HYPRE package, though not all have been implemented for use by multiUQ. Presently the SMG, PFMG, BICGSTAB, and GMRES methods are available within multiUQ. Of these, the PFMG method offers the most computationally efficient and stable solution for the various test cases present in the code. Currently in development is the use of a fast Fourier transform,

using FFTW for quick solution of the pressure field. However, this method is not currently active, but is required for code compilation.

Compilation of the Silo library depends on Szip and HDF5 which together provide the method for storing the output simulation data efficiently. Silo files are output by multiUQ for visualization of results, which is done with the VisIt package from LLNL after running a simulation with multiUQ. Additionally, a wiki is available on Bitbucket with details on compiling the code as well as additional libraries.

Simulation setup

Several inputs are required to run a successful simulation. Included with the source code is an inputs file, which is a text file containing all needed information. Among the variables most commonly used are mesh size, processor allocation (in space), simulation (initial velocity field and multiphase geometry), boundary conditions, CFL number for time step size determination, application of uncertainty and the number of basis functions used, tolerance levels for velocity field divergence and pressure fields, as well as the desired pressure solver.

A number of predefined cases have been implemented in the code including Poiseuille and Couette flows, a deformation case, lid driven cavity flow, oscillating droplet, standing wave, atomizing jet, and a jet in crossflow. When one of these cases is selected as the velocity simulation and geometry profile, the boundary conditions and level set geometries are created with the predefined conditions (found in simulation.f90 and geometry.f90). It is still necessary to choose fluid parameters and a pressure solver, as well as the precision of the divergence free condition and frequency of data output. Additionally, the output files required for visualization of the system depending on quantities of interest must be chosen. These outputs include velocity, pressure, density, and variance about these quantities, among others.

Having set the desired inputs, multiUQ is executed through the command line terminal in either Linux or Unix environments using

```
1 >> mpiexec -np 4 ./multiUQ inputs
```

for the number of proccesors requested (-np). Once the code is running, some basic simulation information will output to the command line terminal, while most will be saved to the folder where the run command was executed.

Outputs

For the most part, outputs come in the form of a Silo file, which is loaded into LLNL's VisIt program for visualization. A variety of arrays are output by default, including average values of velocity, density, viscosity, pressure, and level set, as well as variances about these variables and probability slices for some.

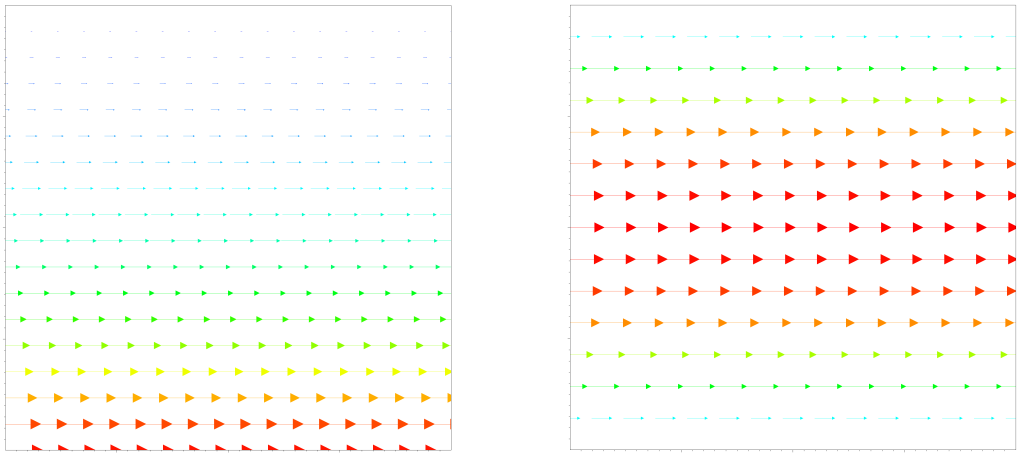
Statistical variables, including variance and probability, are useful for determining areas of a system which have the most uncertainty. When we find regions of uncertainty in our system, we can expect a variety of situations to arise, most of which should be contained within the stochastic functions we use to store the variables.

Simulation capabilities

Several test cases have been built into the multiUQ package which allow for testing the accuracy of the code. These tests exist to determine the accuracy of each component of the multiphase system, including the incompressible velocity field and the level set transport and conservation, as well as the fully coupled system. Given reasonable results in each of these tests, further simulations can be performed by modification of the geometry and velocity initialization and boundary conditions. Presently each of these tests must be run manually after installation.

Navier-Stokes tests

Basic tests of single phase flows are necessary to test the velocity and pressure solver. These can be run once the code has been compiled to be sure the program is running properly and without error. Several test cases are available within multiUQ, including Couette, Poiseuille, and lid-driven cavity flows, an example of which is shown in Fig. 4.5.



(a) Couette flow (b) Poiseuille flow

Figure 4.5: Single phase testing for comparison to analytic results for single phase flow scenarios.

These tests can be run by choosing a test of interest, i.e. 'Couette', in the velocity simulation subject field of the inputs file. Additionally, there is a choice to either use or not use multiphase, which will turn off the level set transport and set all fluid parameters to that of fluid 1. In these tests we are looking for convergence to analytic results, as well as a divergence free velocity field. We make sure boundary conditions are consistent with the flow scenario, and that the pressure field is as expected.

Level set tests

Several level set tests are built into multiUQ for verification of the numerical framework through comparison to analytic results. A simple test case is that of a channel flow, where imposing a uniform velocity field results in transport of the level set a certain distance for a given time. The analytic solution for this test is a displacement of the initial condition by $\boldsymbol{u}t$. Additionally, we can test the ability of multiUQ to transport an uncertain interface, such as that of an uncertain uniform velocity field as shown in Fig. 4.6. Here, we see that uncertainty in the magnitude of the flow velocity causes uncertainty in the displacement of the initial condition.

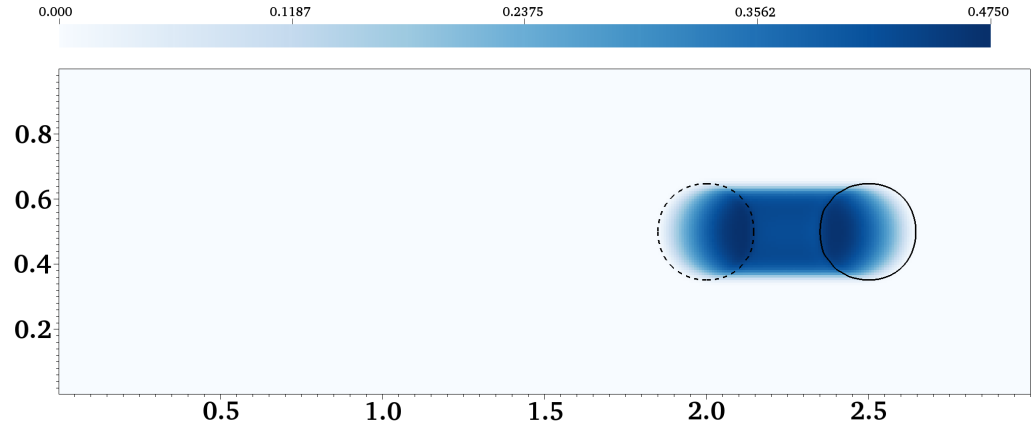


Figure 4.6: Position of the stochastic circle with $\zeta = -1$ (dashed) and $\zeta = +1$ (solid). Level set variance is represented by the color bar, while interface locations correspond to analytic solutions.

Another common benchmark test case is that of the deformation test, as shown in Fig. 4.7. In this scenario a 2D level set is placed in a rotating velocity field. The level set is stretched to a point, then rotated back to its original position. In a perfect scenario, the final solution would match the initial condition. Inspection of the level set at its final location provides a means to assess the accuracy of the level set transport and the ability of the reinitialization procedure to maintain the level

set. Furthermore, this test allows the mass conservation properties to be studied.

(a) $t = 4$ (b) $t = 8$

Figure 4.7: Illustration of the $\psi = 0.5$ interface for the deformation test case at maximum stretching left and final condition right for a 512×512 mesh. At the final time the interface is compared to the starting location, displayed as a dotted line.

Also available is the case of Zalesak's disk, which also tests the transport of the level set in 2D. Beginning with a circle, a slot is removed from the bottom, creating two sharp inside and two sharp outside corners. The disk is then rotated in a rigid-body vortex velocity field for one full cycle. At the conclusion of the test, deformation of the disk is inspected. Given the movement is rigid, a perfect test would result in the final state matching the initial state.

Multiphase tests

There are a few multiphase tests which can be used to test not only the transport of the level set in the calculated velocity field, but also the accuracy of the surface tension force and the precision of the numerical methods employed. A classic test case is that of the oscillating droplet, which has an analytic solution to the period of oscillation, as discussed by [46]. This case, which is typically performed in two dimensions, directly tests the ability of the surface tension force to drive flow, given

no imposition of velocity from the boundaries. A global kinetic energy is calculated at each time step, and the results are output for comparison to periodic expectation. An example is shown in Fig. 4.8, where an uncertain droplet is displayed for a given time, as well as the kinetic energy output for two specific solutions of the case.

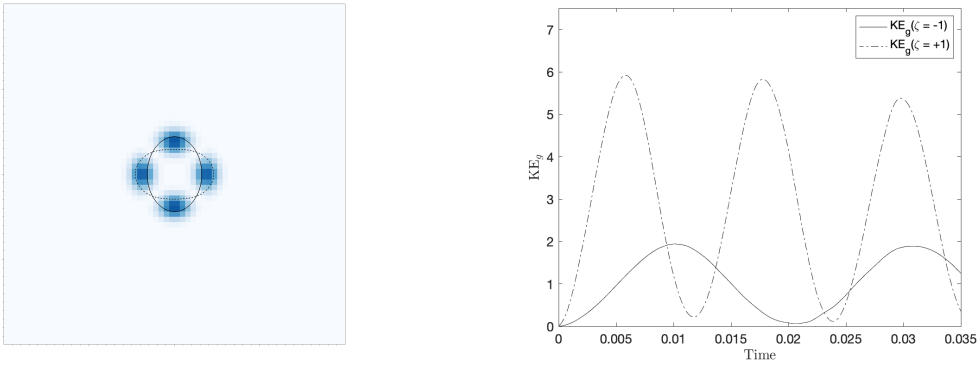
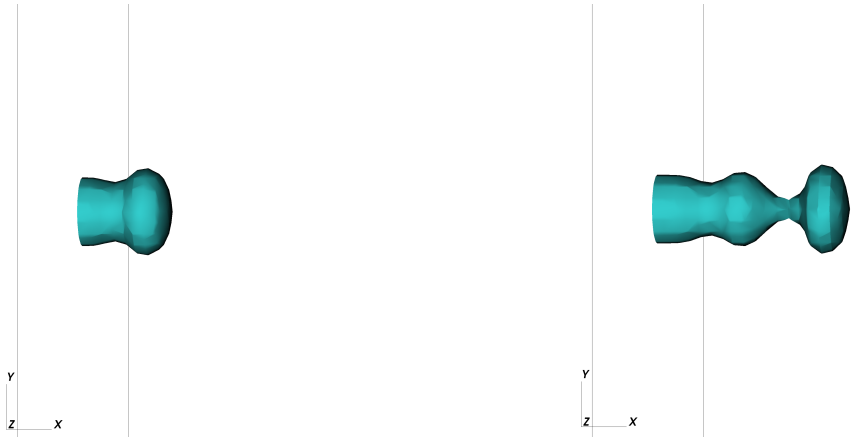


Figure 4.8: An uncertain oscillating droplet is presented, with uncertainty imposed about the surface tension coefficient. At left is a image of the solution at a given time, where the solid line indicates the solution to the droplet with the smallest surface tension coefficient, and the dotted line indicates the solution to the droplet with the largest surface tension coefficient. The color map indicates variance of the level set. At right we see the kinetic energy of the system over time.

Another test is that of the damped surface wave. This test, again run in 2D, gauges the interplay between surface tension and viscous damping of the fluid. At initialization, a flat surface is disturbed slightly. With no gravity imposed, the surface tension term seeks to balance and result in a flat surface. Two main analytic solutions are discussed by [29], which are built into the solver for comparison to numeric results.

These tests and comparisons between numerical and analytic solutions are used to test the implementation and build confidence the solver is working properly. With this knowledge, we can utilize the solver for other research problems. One focus of the multiUQ project was for application to the field of atomizing jets. For example, Fig. 4.9 displays a simulation of an atomizing jet at 2 separate time steps.



(a) Time = 0.0002 s (b) Time = 0.0004 s

Figure 4.9: Coarse mesh (64^3) simulation of an atomizing jet at two progressive time steps.

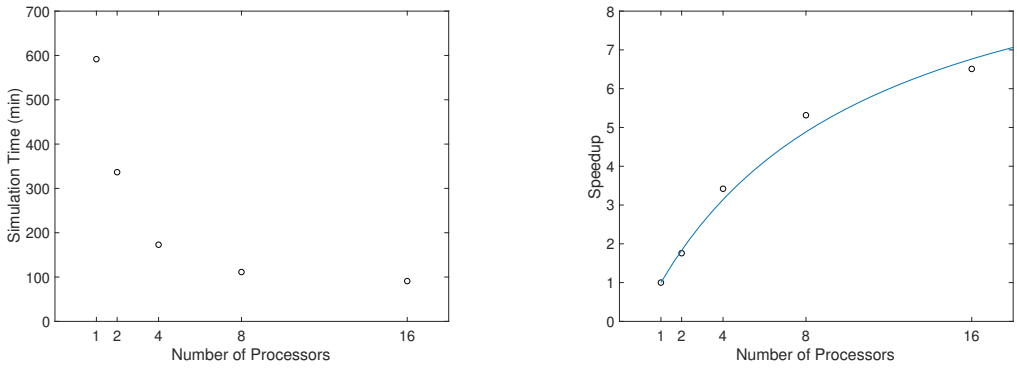
Performance and scaling

multiUQ is parallelizable to create a model capable of high fidelity multiphase flow simulations with uncertainty about several variables. Given the expense of running deterministic models on large clusters, it is also necessary for UQ modeling to be scalable. Amdahl's law for strong scaling states

$$S_s = \frac{1}{s + p/N_{\text{proc}}}, \tag{4.36}$$

where S_s is speedup for strong scaling, s is the proportion of the time spent on serial operations, $p = 1 - s$ is the proportion of time spent on parallel operations, and N_{proc} is the number of processors. We expect to see a speedup of simulations based on the proportion of serial to parallelized calculations in the code. As shown in Fig. 4.10, we have good scaling, with $s = 0.091$ found by way of a non-linear least squares fit.

A different approach to scaling is that of weak scaling, which suggests that as the problem size grows, so to should the number of processors used to perform the



(a) Simulation timing (b) Speedup from 1 processor

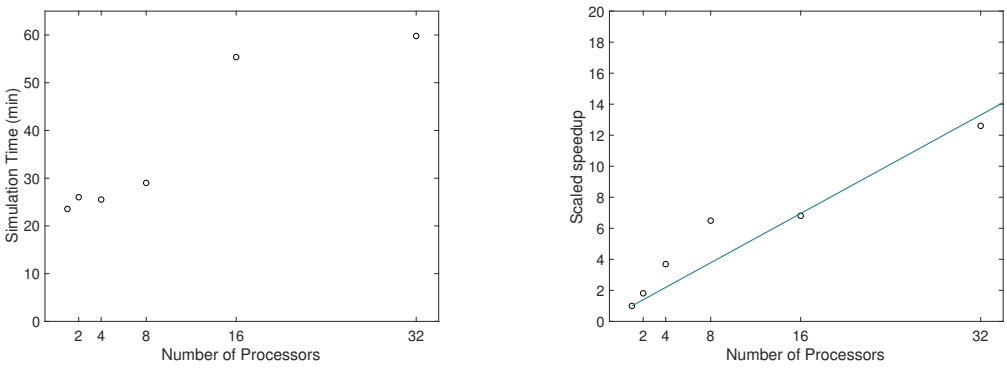
Figure 4.10: Simulation scaling of a coarse (100^3 mesh) running over a range of processors on a single node. At right, the line drawn illustrates strong scaling with Amdahl's law, with $s = 0.091$.

simulation. Gustafson's law states

$$S_w = s + pN_{\text{proc}}, \tag{4.37}$$

where S_w is the speedup for weak scaling, while other variables are as that described for equation 4.36. Fig. 4.11 displays the results of timing as the mesh is refined and the processor count increases, beginning with a 64^3 mesh on a single processor. Using least squares regression, we find that for weak scaling we have a value of $s = 0.603$, which is significantly different from the results of our strong scaling tests.

Additionally, given the parallelization scheme is set up to arrange processors along the spatial domain, we see that the uncertainty domain is not processor dependent. Because of this, we may also need to increase the number of processors in use for simulations requiring more basis functions, i.e. a weak scaling problem.



(a) Simulation timing (b) Speedup from 1 processor

Figure 4.11: Simulation efficiency and scaling for a range of mesh sizes, increasing the processor count as mesh is refined. At right, the line drawn illustrates weak scaling with Gustafson's law, with $s = 0.603$.

Conclusion

We have presented a general overview of the methodology and implementation of multiUQ. At present, this code is available for download as discussed. multiUQ is the first of its kind, and represents a niche sector of uncertainty quantification in multiphase flow dynamics. Certainly, given it is a development level software, modifications are ongoing and necessary to improve the quality of results and further the knowledge of UQ in multiphase systems.

In future work, perhaps the most pressing issues are related to the interface capturing scheme. Ideally a more conservative and robust method, such as a stochastic volume of fluid method, could be employed. Additionally, modifications to the code for improvements in run-time efficiency and scalability are needed, and would aid those wanting to using the software to research a fluid system.

EXPLORATION OF BASIS FUNCTIONS FOR PROJECTING A STOCHASTIC LEVEL SET IN A MULTIPHASE FLOW SOLVER

Contribution of Authors and Co-Authors

Manuscript in following chapter

Author: Brian Turnquist

Contributions: primary author, performed research, code development

Author: Mark Owkes

Contributions: review and feedback, troubleshooting

Manuscript Information

Brian Turnquist and Mark Owkes

Atomization and Sprays

Status of Manuscript:
_X_ Prepared for submission to a peer-reviewed journal
____ Officially submitted to a peer-reviewed journal
____ Accepted by a peer-reviewed journal
____ Published in a peer-reviewed journal

Institute for Liquid Atomization and Spray Systems

Abstract

Application of intrusive uncertainty quantification to simulations of gas-liquid multiphase flows requires a method for capturing or tracking the interface between phases. Previously, polynomial chaos has been paired with a conservative level set method to capture a stochastic interface. While this works, it was found that issues arise when using Legendre polynomials to run simulations with a great deal of uncertainty. To better address this issue, we propose two different sets of orthogonal basis functions that may better describe the shape of the conservative level set used to capture the interface. We consider how well each set of basis functions describes a hyperbolic tangent function, compare the results of each set of basis functions, and determine which set is best for application to high density ratio gas-liquid flows.

Introduction

Improvements in the development of intrusive schemes for the uncertainty quantification (UQ) of multiphase flow dynamics are on-going. At present, only a stochastic conservative level set has been created which is capable of capturing a distribution of interfaces between fluid phases [81]. While the work described so far is a valuable leap into intrusive UQ of multiphase flow dynamics, it has its shortcomings. Most glaring is the difficulty in projecting a hyperbolic tangent function onto a system of polynomials.

Background

The methodology for creating an intrusive stochastic multiphase flow solver is described in detail in [81], and is an extension of work done for single phase fluid

systems by [40]. The extension comes in the form of a stochastic conservative level set method with accompanying reinitialization scheme for maintenance of the level set profile. We'll begin by briefly discussing some details necessary to understand the issue at play.

Uncertainty is added to the Navier-Stokes and level set equations by utilization of stochastic variables, which allow for a variable of interest to be a function of space $\boldsymbol{x}$, time t, and some uncertainty dimension $\boldsymbol{\zeta}$. The stochastic variables utilize polynomial chaos as described by [83]. The basic idea of polynomial chaos is that given an infinite number of basis functions, any function can be recreated, as

$$\psi(\boldsymbol{x}, t, \boldsymbol{\zeta}) = \sum_{n=0}^{\infty} \psi_n(\boldsymbol{x}, t)\phi_n(\boldsymbol{\zeta}). \tag{5.1}$$

In reality, we must use a finite number of basis terms.

Our model for imposing uncertainty into a system focuses on the use of polynomial chaos for expansion of variables into the uncertainty dimension $\boldsymbol{\zeta}$. For creation of a stochastic level set transport, we begin with a deterministic level set transport

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\psi \boldsymbol{u}) = 0 \tag{5.2}$$

for level set ψ and velocity $\boldsymbol{u}$. We then substitute stochastic velocity $\boldsymbol{u}_k\phi_k$ and stochastic level set $\psi_k\phi_k$, noting the use of Einstein notation for summation over repeated indices. We then have

$$\frac{\partial \psi_k\phi_k}{\partial t} + \nabla \cdot (\psi_k\phi_k\boldsymbol{u}_l\phi_l) = 0, \tag{5.3}$$

with basis function ϕ_k of order k. To solve for each basis weight, and create an equation which can be more easily solved, we next invoke the property of

orthogonality. This property is dependent on the basis functions used, but for the Legendre polynomials is

$$\int_{-1}^{1} \phi_k \phi_b \delta\zeta = \begin{cases} \langle \phi_k \phi_b \rangle & k = b \\ 0 & k \neq b \end{cases}. \tag{5.4}$$

Utilizing orthogonality, we then multiply equation 5.3 by a test function ϕ_b, and integrate over the region of orthogonality

$$\int_{-1}^{1} \frac{\partial \psi_k \phi_k \phi_b}{\partial t} + \nabla \cdot (\psi_k \phi_k \boldsymbol{u}_l \phi_l) \phi_b = 0. \tag{5.5}$$

Having integrated, we then divide through by the resulting constant and end up with a stochastic level set transport

$$\frac{\partial \psi_b}{\partial t} + \nabla \cdot (\psi_k \boldsymbol{u}_l)\, C_{klb} = 0. \tag{5.6}$$

for $b = 0, \ldots, N$, where

$$C_{klb} = \frac{\int_{-1}^{1} \phi_k \phi_l \phi_b d\zeta}{\int_{-1}^{1} \phi_b \phi_b d\zeta} = \frac{\langle \phi_k \phi_l \phi_b \rangle}{\langle \phi_b \phi_b \rangle}. \tag{5.7}$$

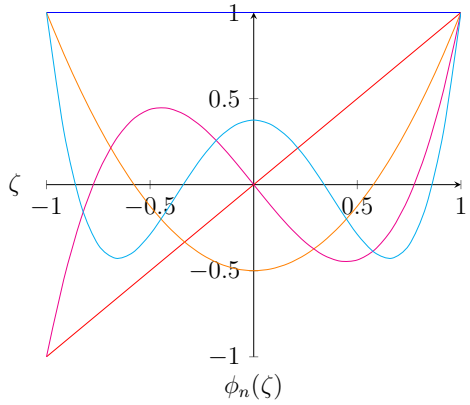
While we have found that this works well for a number of scenarios, as outlined in [81], and [80], we have also found scenarios where this does not. A key issue is the set of basis functions ϕ upon which to project the variable of interest. In previous work all simulations run have used Legendre polynomials, as they are very easy to calculate and don't require the use of an integration factor for orthogonality.

However, Legendre polynomials suffer from a couple of problems when employed in this framework. Determination of Legendre polynomials can be accomplished by

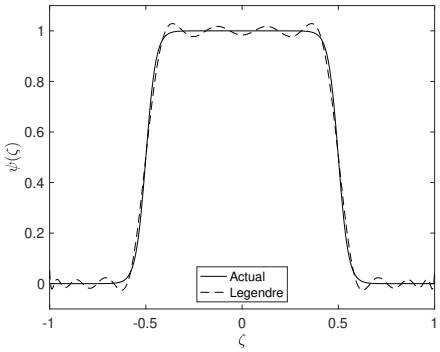
Rodrigues' formula

$$\phi_n(\zeta) = \frac{1}{2^n n!}\frac{d^n}{d\zeta^n}(\zeta^2 - 1)^n \tag{5.8}$$

for a polynomial ϕ of order n. As shown, as the order n increases, the constant values in the polynomial increase very quickly, which can easily overwhelm a numerical computation and reach machine precision limits. This can be somewhat remedied by using double, quadruple, or higher order machine precision, but at high computational cost. Another issue is caused by the shape of the Legendre polynomials, especially with respect to the recreation of a jump condition, such as a Heaviside function. Fig. 5.1 displays the shape of the first five Legendre polynomials, which not quickly translate to that of a Heaviside or hyperbolic tangent function.



(a) First 5 Legendre polynomials.



(b) Convergence with $N = 25$ basis functions.

Figure 5.1: Illustrating the issue of convergence to a hyperbolic tangent for projection onto a system of Legendre polynomials.

We can see in Fig. 5.1b that given the poor mismatch of shape and the problem of machine precision, it is difficult to generate enough terms to adequately recreate a step function with Legendre polynomials. The question then becomes, what is a reasonable alternative? Given the usefulness of describing a variable in terms of uncertainty, as well as the previously described application of polynomial chaos in

fluid dynamics simulations ([40, 81]), it is worth attempting to utilize a set of basis functions that better describes a hat function than the Legendre polynomials.

Mathematical development

With the conservative level set approach taken by [15] and [54], the hyperbolic tangent profile of the level set useful for defining the interface where properties of either fluid phase are then set. As discussed in [81] for the stochastic level set method this hyperbolic tangent profile is used as a smoothed Heaviside function, and is used differently for calculation of the density and viscosity within each fluid phase. In a deterministic setting, this is accomplished with

$$\mu = \mu_1\psi + (1-\psi)\mu_2 = \mu_2 + (\mu_1 - \mu_2)\psi \tag{5.9}$$

$$\eta = \eta_1\psi + (1-\psi)\eta_2 = \eta_2 + (\eta_1 - \eta_2)\psi \tag{5.10}$$

for viscosity μ and specific volume $\eta = 1/\rho$ for density ρ. Allowing for uncertainty about both fluid phases (i.e. $\mu_1(\boldsymbol{\zeta}) = \mu_{1:k}\phi_k$ and $\mu_2(\boldsymbol{\zeta}) = \mu_{2:k}\phi_k$), we calculate stochastic quantities via

$$\mu_b = \mu_{2:b} + (\mu_{1:k}\psi_l - \mu_{2:k}\psi_l)\, C_{klb} \tag{5.11}$$

$$\eta_b = \eta_{2:b} + (\eta_{1:k}\psi_l - \eta_{2:k}\psi_l)\, C_{klb}. \tag{5.12}$$

With this method for calculation of fluid properties, any values outside the range of $[0, 1]$ cause calculations of density and viscosity that can be non-physical (potentially negative). The following section discusses the use of two different sets of orthogonal basis functions that appear able to avoid these types of calculations. We develop the calculation of the basis functions, as well as implications to the

substitution of stochastic variables into the level set equation 4.14.

Orthogonal step functions

Given the need to avoid the Gibbs phenomenon, or the ringing effect caused near discontinuities as shown in Fig. 5.1b, it makes sense to utilize a set of orthogonal step functions. A system of orthogonal step functions was described by Li & Torney [42] which utilize the periodic sign change of the sine and cosine functions to jump between positive and negative values.

These step functions exist in pairs, similar to that of a Fourier series. To calculate the system we say, for $j \in \mathbb{N}$ let $\underline{j}$ denote its odd part, which is the quotient of j by its largest power of two factor. Additionally, the summations range over the divisors of $\underline{j}$, i.e. $l|\underline{j}$. We then have

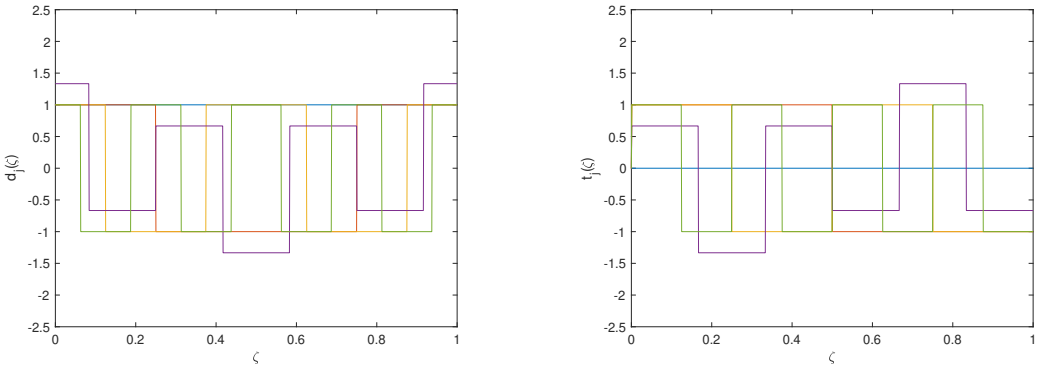
$$\begin{aligned} d_0(\zeta) &= 1 \\ d_j(\zeta) &= \sum_{l|\underline{j}} (-1)^{(l-1)/2} l^{-1} M(l) c_{j/l}(\zeta) \\ t_j(\zeta) &= \sum_{l|\underline{j}} l^{-1} M(l) s_{j/l}(\zeta) \end{aligned} \tag{5.13}$$

where $M(l)$ is the classical Mobius function and

$$\begin{aligned} c_j(\zeta) &= \operatorname{sgn}(\cos 2\pi j\zeta) \quad \text{for} \quad j \in 0 + \mathbb{N} \\ s_j(\zeta) &= \operatorname{sgn}(\sin 2\pi j\zeta) \quad \text{for} \quad j \in \mathbb{N} \end{aligned} \tag{5.14}$$

with $\operatorname{sgn}(\zeta)$ is the signum function, taking values of -1, 0, and 1 for negative, vanishing, and positive arguments, respectively. These basis function are shown in Fig. 5.2.

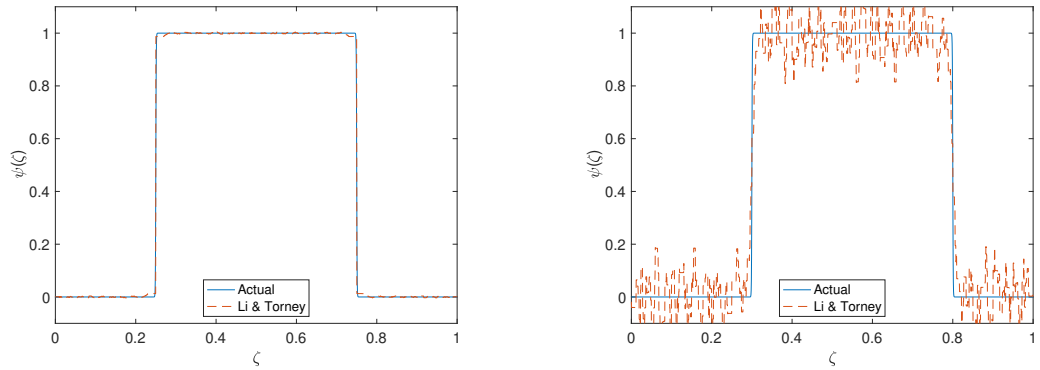
If we project a hyperbolic tangent profile ψ on to these functions, as shown in Fig. 5.3, we have an interesting situation. At left we see a very good projection of



(a) $d_j(\zeta)$ (b) $t_j(\zeta)$

Figure 5.2: First five orthogonal step functions described by [42].

an even function onto the orthogonal step functions. However, if we shift ψ slightly and create an odd function, we again have the issue of the Gibbs phenomenon as with the Legendre polynomials. This raises questions of whether there is fault in the calculation of the odd functions $t_j(\zeta)$, or perhaps the description of these functions.



(a) Centered (even) $\psi(\zeta)$ (b) Offset (odd) $\psi(\zeta)$

Figure 5.3: Projection of hyperbolic tangent profile $\psi(\zeta)$ onto a system of orthogonal step functions.

Orthonormal Bernstein polynomials

Another possibility is an orthogonal set of Bernstein polynomials, as outlined by [4]. We again seek to avoid the Gibbs phenomenon and thus to cancel any chance of calculating non-physical or negative values of density and viscosity. To calculate the standard Bernstein polynomials we use

$$B_{j,n}(\zeta) = \binom{n}{j} \frac{(\zeta - a)^j (b - \zeta)^{n-j}}{(b - a)^n} \tag{5.15}$$

for $j = 0, 1, \ldots, n$ over the arbitrary interval $[a, b]$.

Calculation of orthogonal Bernstein polynomials can be accomplished with

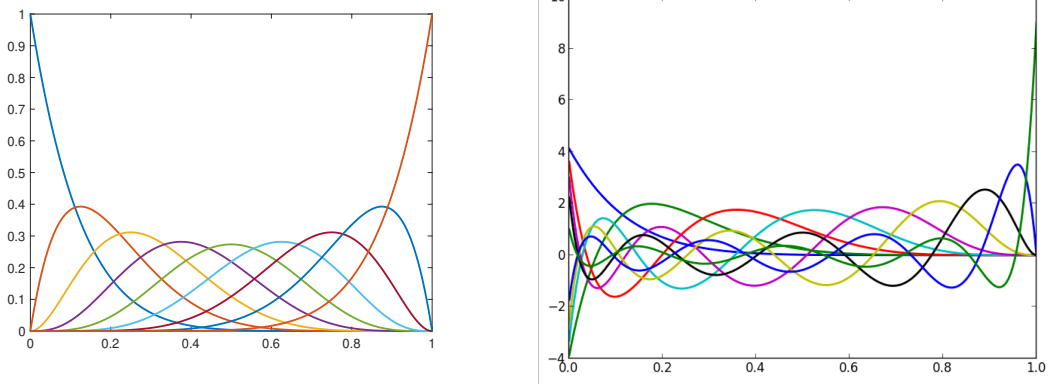
$$\phi_{j,n}(\zeta) = \left(\sqrt{2(n-j)+1}\right)(1-\zeta)^{n-j} \sum_{k=0}^{j} (1)^k \binom{2n+1-k}{j-k} \binom{j}{k} \zeta^{j-k} \tag{5.16}$$

for $j = 0, 1, \ldots, n$ over the interval $[0, 1]$. Additionally, the orthogonal Bernstein polynomials can be calculated from the standard Bernstein polynomials with

$$\phi_{j,n}(\zeta) = \left(\sqrt{2(n-j)+1}\right) \sum_{k=0}^{j} (1)^k \frac{\binom{2n+1-k}{j-k}\binom{j}{k}}{\binom{n-k}{j-k}} B_{j-k,n-k}(\zeta) \tag{5.17}$$

for $j = 0, 1, \ldots, n$ over any interval $[a, b]$. Looking at Fig. 5.4 we see the shape of Bernstein polynomials. The standard set exist as a breakdown of 1, where adding up the polynomials at any point in the interval results in a value of 1.

Utilizing the Bernstein polynomials, we now project hyperbolic tangent $\psi(\zeta)$ onto $B_{j,n}(\zeta)$. In Fig. 5.5 we see that we have found a set of polynomials capable of avoiding the Gibbs phenomenon. However, we now have a new issue. Bernstein polynomials require a high basis order n to converge to a hyperbolic tangent. This results in a smearing of the sharp interface in the uncertainty domain ζ. This smearing

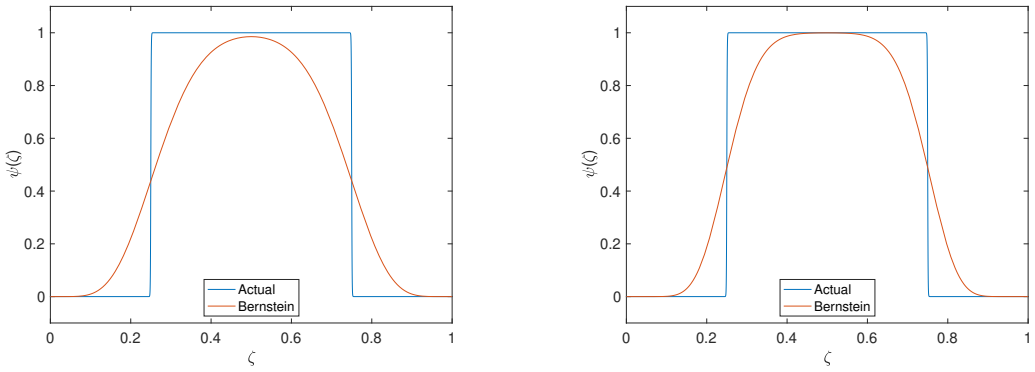


(a) Standard Bernstein polynomials

(b) Orthogonal Bernstein polynomials

Figure 5.4: Bernstein polynomials of order $n = 8$. Orthonormal Bernstein polynomials at right as shown and described by [4].

of the interface could result in reduced precision for fine mesh simulations.



(a) Bernstein order $n = 25$.

(b) Bernstein order $n = 50$.

Figure 5.5: Projection of hyperbolic tangent profile $\psi(\zeta)$ onto a system of Bernstein polynomials.

Conclusion

We have outlined an issue relating to the set of basis functions used to create a stochastic conservative level set. Where in the original implementation, use of

Legendre polynomials worked to display the proof of concept, these polynomials cause issues when dealing with systems where a great deal of variability about the location of the interface takes place. One solution is to find a set of basis functions that more easily defines a Heaviside or hat function. We then describe the use of two types of orthogonal basis functions for the description of a stochastic level set. While these sets of basis functions show some promise, they each bring with them their own issues that need to be addressed prior to implementation in the multiUQ package.

CONCLUSION

We have outlined the culmination of efforts to create and improve upon intrusive uncertainty quantification schemes for gas-liquid multiphase flows. First, a proof-of-concept was created, with satisfactory results that have been published in the Journal of Computational Physics [81]. In this paper we outlined the thought process and model for extending previous work of UQ in single phase flows to the multiphase regime, and highlighted the ability of the methodology to work in a conservative level set scheme for capturing an interface between phases.

Next, we discussed the computational cost of the pressure Poisson equation, which grew excessively expensive as the number of basis functions used to project a variable increased. To counteract the growing computational cost of increasing the number of basis functions used, a density decoupled pressure correction method was employed to decouple the pressure Poisson equation of each basis weight P_b. The result was a significant speedup in computational time, and a method that is convergent to the traditional pressure correction method. Additionally, speedup is shown to increase as the number of basis functions is increased and the mesh size is refined.

With reduced computational cost due to the implementation of the density-decoupled pressure correction, it became possible to implement a 3-D version of multiUQ. Though expansion of multiUQ into 3-D was not the primary focus, it does allow for a more useful software package. We thus wrote a third manuscript to spread the word about the abilities of multiUQ, and offer a brief overview of the code implementation, as well as how to download a copy and begin running simulations. This can help to foster continuing research in UQ of multiphase systems.

A last chapter discussed ongoing efforts to implement a set of basis functions that

more closely resembles the shape profile of the conservative level set. Identification of issues related to poor projections of the level set onto Legendre polynomials prompted a study into more adequate basis functions. With more accurate representations of the stochastic variables we can produce more reliable stochastic solutions of multiphase flows.

Looking ahead, there is certainly more work to be done. In addition to better representing the conservative level set, there is the possibility of creating a stochastic volume of fluid method that will be mass conservative, as is the case with volume of fluid methods, and better represent the interface[s] as a result. Additionally, to further improve computational efficiency, it would be useful to either consider a load balancing approach, where we could only perform level set operations near a stochastic interface (where weights $\psi_b = 0$) and distribute the load throughout all processors. Perhaps a more elegant approach is to identify a method for reducing the load caused by increasing the number of basis functions, where we identify an error cutoff which stops the calculation of further basis weights which do not add anything to the projection.

Given the utility of reducing computational cost, it is also important to compare the cost of intrusive and non-intrusive methods. Having a quantitative comparison of the cost of computation, as well as the potential cost of adding intrusive UQ to existing packages would likely improve adoption of these sort of UQ methods.

REFERENCES CITED

[1] G Adomian. Stochastic Green's Functions. In *Stochastic Processes in Mathematical Physics and Engineering*, volume 16, pages 1–39. American Mathematical Society, Providence, RI, 1964. Publisher: Amer. Math. Soc., Providence, RI.

[2] John B Bell, Phillip Colella, and Harland M Glaz. A second-order projection method for the incompressible navier-stokes equations. *Journal of Computational Physics*, 85(2):257–283, December 1989.

[3] Thomas Bellotti and Maxime Theillard. A coupled level-set and reference map method for interface representation with applications to two-phase flows simulation. *Journal of Computational Physics*, 392:266–290, September 2019.

[4] Michael A Bellucci. On the explicit representation of orthonormal Bernstein polynomials. *arXiv preprint arXiv:1404.2293*, 2014.

[5] Jesus Benajes, Ricardo Novella, Jose Manuel Pastor, Alberto Hernández-López, Manabu Hasegawa, Naohide Tsuji, Masahiko Emi, Isshoh Uehara, Jordi Martorell, and Marcos Alonso. Optimization of the combustion system of a medium duty direct injection diesel engine by combining CFD modeling with experimental validation. *Energy Conversion and Management*, 110:212–229, February 2016.

[6] Sylvio R Bistafa. On the development of the Navier-Stokes equation by Navier. *Revista Brasileira de Ensino de Física*, 40(2), 2018. Publisher: SciELO Brasil.

[7] J.U Brackbill, D.B Kothe, and C Zemach. A continuum method for modeling surface tension. *Journal of Computational Physics*, 100(2):335–354, June 1992.

[8] Robert Chiodi and Olivier Desjardins. A reformulation of the conservative level set reinitialization equation for accurate and robust simulation of complex multiphase flows. *Journal of Computational Physics*, 343:186–200, August 2017.

[9] David Layne Chopp. Computing minimal surfaces via level set curvature flow. Technical report, Elsevier, 1991.

[10] Alexandre Joel Chorin. Numerical solution of the Navier-Stokes equations. *Mathematics of computation*, 22(104):745–762, 1968.

[11] Edmond Chow, A Cleary, and R Falgout. Design of the hypre preconditioner library. Technical report, Lawrence Livermore National Lab., CA (US), 1998.

[12] Paolo Cifani. Analysis of a constant-coefficient pressure equation method for fast computations of two-phase flows at high density ratios. *Journal of Computational Physics*, 398:108904, December 2019.

[13] Mary Kathryn Cowles and Bradley P. Carlin. Markov chain Monte Carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434):883–904, 1996.

[14] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(1):50–67, 1947.

[15] Olivier Desjardins, Vincent Moureau, and Heinz Pitsch. An accurate conservative level set/ghost fluid method for simulating turbulent atomization. *Journal of Computational Physics*, 227(18):8395–8416, September 2008.

[16] Dettmer W., Saksono P. H., and Perić D. On a finite element formulation for incompressible Newtonian fluid flows on moving domains in the presence of surface tension. *Communications in Numerical Methods in Engineering*, 19(9):659–668, August 2003.

[17] Michael S. Dodd and Antonino Ferrante. A fast pressure-correction method for incompressible two-fluid flows. *Journal of Computational Physics*, 273:416–434, September 2014.

[18] S. Dong and J. Shen. A time-stepping scheme involving constant coefficient matrices for phase-field simulations of two-phase incompressible flows with large density ratios. *Journal of Computational Physics*, 231(17):5788–5804, July 2012.

[19] Mohamed A. El-Beltagy and Mohamed I. Wafa. Stochastic 2D Incompressible Navier-Stokes Solver Using the Vorticity-Stream Function Formulation. *Journal of Applied Mathematics*, pages 1–14, January 2013.

[20] Douglas Enright, Ronald Fedkiw, Joel Ferziger, and Ian Mitchell. A Hybrid Particle Level Set Method for Improved Interface Capturing. *Journal of Computational Physics*, 183(1):83–116, November 2002.

[21] Thomas Frachon and Sara Zahedi. A cut finite element method for incompressible two-phase Navier–Stokes flows. *Journal of Computational Physics*, 384:77–98, May 2019.

[22] Marco Frasca and Asatur Khurshudyan. General Representation of Nonlinear Green's Function for Second Order Differential Equations Nonlinear in the First Derivative. *arXiv preprint arXiv:1806.00274*, 2018.

[23] D.E Fyfe, E.S Oran, and M.J Fritts. Surface tension and viscosity with lagrangian hydrodynamics on a triangular mesh. *Journal of Computational Physics*, 76(2):349–384, June 1988.

[24] Daniel P. Garrick, Mark Owkes, and Jonathan D. Regele. A finite-volume HLLC-based scheme for compressible interfacial flows with surface tension. *Journal of Computational Physics*, 339:46–67, June 2017.

[25] A. Gel, R. Garg, C. Tong, M. Shahnam, and C. Guenther. Applying uncertainty quantification to multiphase flow computational fluid dynamics. *Powder Technology*, 242:27–39, July 2013.

[26] Roger Ghanem, David Higdon, and Houman Owhadi. *Handbook of uncertainty quantification*, volume 6. Springer, 2017.

[27] E. Gutiérrez, F. Favre, N. Balcázar, A. Amani, and J. Rigola. Numerical approach to study bubbles and drops evolving through complex geometries by using a level set – Moving mesh – Immersed boundary method. *Chemical Engineering Journal*, 349:662–682, October 2018.

[28] Harten Amiram. The artificial compression method for computation of shocks and contact discontinuities. I. Single conservation laws. *Communications on Pure and Applied Mathematics*, 30(5):611–638, September 1977.

[29] M. Herrmann. A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids. *Journal of Computational Physics*, 227(4):2674–2706, February 2008.

[30] C.W. Hirt, A.A. Amsden, and J.L. Cook. An Arbitrary Lagrangian–Eulerian Computing Method for All Flow Speeds. *Journal of Computational Physics*, 135(2):203–216, August 1997.

[31] 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, January 1981.

[32] Serhat Hosder, Robert Walters, and Rafael Perez. A Non-Intrusive Polynomial Chaos Method For Uncertainty Propagation in CFD Simulations. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 2006.

[33] Thomas J.R. Hughes, Wing Kam Liu, and Thomas K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29(3):329–349, December 1981.

[34] M Jardak, C-H Su, and George E Karniadakis. Spectral polynomial chaos solutions of the stochastic advection equation. *Journal of Scientific Computing*, 17(1-4):319–338, 2002. Publisher: Springer.

[35] Kari Karhunen. *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*, volume 37. Sana, 1947.

[36] O.M. Knio and O.P. Le Maître. Uncertainty propagation in CFD using polynomial chaos decomposition. *Recent Topics in Computational Fluid Dynamics*, 38(9):616–640, September 2006.

[37] Ilja Kröker, Wolfgang Nowak, and Christian Rohde. A stochastically and spatially adaptive parallel scheme for uncertain and nonlinear two-phase flow problems. *Computational Geosciences*, 19(2):269–284, 2015.

[38] O. P. Le Maître and Omar M. Knio. *Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics*. With Applications to Computational Fluid Dynamics. Springer Netherlands, Dordrecht, Dordrecht, 2010.

[39] O.P. Le Maître, Omar M Knio, Habib N Najm, and Roger G Ghanem. A Stochastic Projection Method for Fluid Flow. *Journal of Computational Physics*, 173(2):481–511, November 2001.

[40] O.P. Le Maître, Matthew T. Reagan, Habib N. Najm, Roger G. Ghanem, and Omar M. Knio. A Stochastic Projection Method for Fluid Flow: II. Random Process. *Journal of Computational Physics*, 181(1):9, September 2002.

[41] Chang Sik Lee and Rolf D. Reitz. EFFECT OF LIQUID PROPERTIES ON THE BREAKUP MECHANISM OF HIGH-SPEED LIQUID DROPS. *Atomization and Sprays*, 11(1):1–19, February 2001.

[42] Huaien Li and David Torney. A complete system of orthogonal step functions. *Proceedings of the American Mathematical Society*, 132(12):3491–3502, 2004.

[43] Qinzhuo Liao and Dongxiao Zhang. Constrained probabilistic collocation method for uncertainty quantification of geophysical models. *Computational Geosciences*, 19(2):311–326, April 2015.

[44] ZhiYi Liu, XiaoDong Wang, and Shun Kang. Stochastic performance evaluation of horizontal axis wind turbine blades using non-deterministic CFD simulations. *Energy*, 73:126–136, August 2014.

[45] Michel Loeve. Probability theory: foundations, random sequences. 1955.

[46] F. R. S. Lord Rayleigh. VI. On the capillary phenomena of jets. *Proceedings of the Royal Society of London*, 29(196-199):71–97, January 1879.

[47] M.R Malik, T.A Zang, and M.Y Hussaini. A spectral collocation method for the Navier-Stokes equations. *Journal of Computational Physics*, 61(1):64–88, October 1985.

[48] Lionel Mathelin, M. Yousuff Hussaini, and Thomas A. Zang. Stochastic approaches to uncertainty quantification in CFD simulations. *Numerical Algorithms*, 38(1):209–236, 2005.

[49] Jeremy O. McCaslin and Olivier Desjardins. A localized re-initialization equation for the conservative level set method. *Journal of Computational Physics*, 262:408–426, April 2014.

[50] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[51] T.M. Myers and A.W. Marshall. A description of the initial fire sprinkler spray. *Fire Safety Journal*, 84:1–7, August 2016.

[52] Habib N. Najm. Uncertainty Quantification and Polynomial Chaos Techniques in Computational Fluid Dynamics. *Annual Review of Fluid Mechanics*, 41(1):35–52, December 2008.

[53] R.R. Nourgaliev and T.G. Theofanous. High-fidelity interface tracking in compressible flows: Unlimited anchored adaptive level set. *Journal of Computational Physics*, 224(2):836–866, June 2007.

[54] Elin Olsson and Gunilla Kreiss. A conservative level set method for two phase flow. *Journal of Computational Physics*, 210(1):225–246, November 2005.

[55] Elin Olsson, Gunilla Kreiss, and Sara Zahedi. A conservative level set method for two phase flow II. *Journal of Computational Physics*, 225(1):785–807, July 2007.

[56] Stanley Osher and Ronald P Fedkiw. Level set methods: an overview and some recent results. *Journal of Computational physics*, 169(2):463–502, 2001.

[57] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[58] Mark Owkes, Eric Cauble, Jacob Senecal, and Robert A. Currie. Importance of curvature evaluation scale for predictive simulations of dynamic gas–liquid interfaces. *Journal of Computational Physics*, 365:37–55, July 2018.

[59] Mark Owkes and Olivier Desjardins. A discontinuous Galerkin conservative level set scheme for interface capturing in multiphase flows. *Journal of Computational Physics*, 249:275–302, September 2013.

[60] Mark Owkes and Olivier 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, August 2014.

[61] Mohammad-Reza Pendar and José Carlos Páscoa. Numerical modeling of electrostatic spray painting transfer processes in rotary bell cup for automotive painting. *International Journal of Heat and Fluid Flow*, 80:108499, December 2019.

[62] Per Pettersson, Gianluca Iaccarino, and Jan Nordström. An intrusive hybrid method for discontinuous two-phase flow under uncertainty. *Computers & Fluids*, 86:228–239, November 2013.

[63] Andrea Prosperetti. Motion of two superposed viscous fluids. *The Physics of Fluids*, 24(7):1217–1223, July 1981.

[64] Richardson, L.F. IX. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 210(459-470):307, January 1911.

[65] P. J. Roache. QUANTIFICATION OF UNCERTAINTY IN COMPUTATIONAL FLUID DYNAMICS. *Annual Review of Fluid Mechanics*, 29(1):123–160, January 1997.

[66] Romberg, Werner. Vereinfachte numerische Integration. *Det Kongelige Norske Videnskabers Selskab Forhandlinger*, 28(7):30–36, 1955.

[67] Rudman Murray. VOLUME-TRACKING METHODS FOR INTERFACIAL FLOW CALCULATIONS. *International Journal for Numerical Methods in Fluids*, 24(7):671–691, December 1998.

[68] A. Salih and S. Ghosh Moulic. Oscillation of a Liquid Drop in a Zero-Gravity Environment - A Benchmark Problem for Two-Phase Flow Computations. Roorkee, India, December 2002.

[69] Ruben Scardovelli and Stéphane Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annual review of fluid mechanics*, 31(1):567–603, 1999.

[70] M. Schick, V. Heuveline, and O. P. Le Maître. A Newton-Galerkin Method for Fluid Flow Exhibiting Uncertain Periodic Dynamics. *SIAM Review*, 58(1):119–140, January 2016.

[71] Seungwon Shin and Damir Juric. Modeling Three-Dimensional Multiphase Flow Using a Level Contour Reconstruction Method for Front Tracking without Connectivity. *Journal of Computational Physics*, 180(2):427–470, August 2002.

[72] P. Sochala and O.P. Le Maître. Polynomial Chaos expansion for subsurface flows with uncertain soil parameters. *Advances in Water Resources*, 62:139–154, December 2013.

[73] George Gabriel Stokes. On the Steady Motion of Incompressible Fluides. *Transactions of the Cambridge Philosophical Society*, 1880. Publisher: Cambridge at the University Press.

[74] Bruno Sudret. Global sensitivity analysis using polynomial chaos expansions. *Bayesian Networks in Dependability*, 93(7):964–979, July 2008.

[75] Mark Sussman, Peter Smereka, and Stanley Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114(1):146–159, September 1994.

[76] Menner A. Tatang, Wenwei Pan, Ronald G. Prinn, and Gregory J. McRae. An efficient method for parametric uncertainty analysis of numerical geophysical models. *Journal of Geophysical Research: Atmospheres*, 102(D18):21925–21932, September 1997.

[77] MICHAEL THOMADAKIS and MICHAEL LESCHZINER. A PRESSURE-CORRECTION METHOD FOR THE SOLUTION OF INCOMPRESSIBLE VISCOUS FLOWS ON UNSTRUCTURED GRIDS. *International Journal for Numerical Methods in Fluids*, 22(7):581–601, April 1996.

[78] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A Front-Tracking Method for the Computations of Multiphase Flow. *Journal of Computational Physics*, 169(2):708–759, May 2001.

[79] G. Tryggvason, R. Scardovelli, and S. Zaleski. *Direct Numerical Simulations of Gas–Liquid Multiphase Flows*. Cambridge University Press, 2011.

[80] Brian Turnquist and Mark Owkes. A fast, density decoupled pressure solver for an intrusive stochastic multiphase flow solver. *Journal of Computational Physics*, Under Review.

[81] Brian Turnquist and Mark Owkes. multiUQ: An intrusive uncertainty quantification tool for gas-liquid multiphase flows. *Journal of Computational Physics*, 399:108951, December 2019.

[82] JJIM Van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM Journal on Scientific and Statistical Computing*, 7(3):870–891, 1986.

[83] Norbert Wiener. The Homogeneous Chaos. *American Journal of Mathematics*, 60(4):897–936, 1938.

[84] Dongbin Xiu and George Em Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of Computational Physics*, 187(1):137–167, May 2003.

[85] Steven T Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31(3):335–362, June 1979.