Title

A Hybrid Finite Difference Level Set—Implicit Mesh Discontinuous Galerkin Method for Multi-Layer Coating Flow Problems

Author

Corcos, Luke

Publication Date

2023

Peer reviewed|Thesis/dissertation

A Hybrid Finite Difference Level Set–Implicit Mesh Discontinuous Galerkin Method for Multi-Layer Coating Flow Problems

by

Luke Pernal Corcos

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Applied Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor James A. Sethian, Chair
Professor Per-Olof Persson
Professor Phillip Colella

Fall 2023

A Hybrid Finite Difference Level Set–Implicit Mesh Discontinuous Galerkin Method for Multi-Layer Coating Flow Problems

Abstract

A Hybrid Finite Difference Level Set–Implicit Mesh Discontinuous Galerkin Method for Multi-Layer Coating Flow Problems

by

Luke Pernal Corcos

Doctor of Philosophy in Applied Mathematics

University of California, Berkeley

Professor James A. Sethian, Chair

Industrial painting operations consume significant amounts of energy, owing in part to the individual application and curing of multiple layers of paint. Energy-efficient manufacturing lines that co-cure (i.e., simultaneously bake) multiple film-layers have the potential to reduce energy consumption by 30%. However, achieving a smooth, defect-free film of paint is often the biggest technical hurdle to commercializing these energy-efficient coating systems. In this thesis, we develop high-fidelity mathematical and numerical frameworks to model the complex multi-physics underlying multi-layer coating flow dynamics, with applications to the leveling of multi-layer paint films, i.e., the coupled evaporation, solidification, fluid flow, and settling dynamics of multiple layers of liquid paint.

Our mathematical model captures a coupled set of multi-physics that includes multi-phase quasi-Newtonian fluid dynamics; the transport, diffusion, and mixing of multiple dissolved species; mass transfer and interface recession from solvent evaporation; intricate interfacial forces of surface tension and Marangoni stresses on paint-gas and paint-paint interfaces and their coupling; and substrate roughness and the pull of gravity. Using this model, we study the highly complex and intricate dynamics of "watching paint dry", capturing several experimental findings and studying the formation of Marangoni plumes and Bénard cells, the impact of long-wave deformational surface modes on immersed interfaces, and the emergence of the final multi-layer film profile.

This thesis presents a hybrid numerical framework for the multi-layer coating flow problem that consists of: finite difference level set methods and high-order accurate sharp-interface implicit mesh discontinuous Galerkin methods; newly developed local discontinuous Galerkin solvers for Poisson problems with Robin boundary and jump conditions on implicitly-defined domains, to capture solvent evaporation; state-of-the-art Stokes solvers that integrate concentration-dependent rheological parameters for quasi-Newtonian interface

dynamics; high-order accurate methods to couple the transport, diffusion, and evaporation of multiple dissolved species while also tracking interface recession; a tailored finite difference projection algorithm that calculates surface gradients, to robustly and accurately incorporate Marangoni stresses; and a coupled multi-physics time stepping approach that incorporates all the different solvers at play, among a host of additional numerical algorithms. Several components of our hybrid numerical framework are high-order accurate and the algorithm is applicable to an arbitrary number of layers and dissolved species. Our particular implementation of the fully coupled numerical algorithm for the multi-layer coating flow problem is 2^{nd} order accurate in space and 1^{st} order in time. A new high-order accurate local discontinuous Galerkin formulation for Stokes problems with Navier-slip boundary conditions on implicitly-defined domains is also presented in the appendix.

The framework is designed, in part, to predict the ultimate surface roughness of the multi-layer system; here, we apply it to a variety of settings, including multi-solvent evaporative paint dynamics, the flow and leveling of multi-layer automobile paint coatings in both 2D and 3D—presenting the results of a 2D parametric study performed at industrially-relevant conditions, and an examination of "interfacial turbulence" within a multi-layer matter cascade. This work revealed many of the driving mechanisms underlying multi-layer coating flow dynamics, including: the creation, characteristics, and impact of short- and long-wave Marangoni hydrodynamic instabilities; the impact of basecoat deformation and telegraphing of substrate roughness on the clearcoat surface profile; conjectures concerning the role of interfacial forces exhibited by the immersed paint-paint interface; and the overall dynamic's significant sensitivity to mass diffusion coefficients. The model and the developed numerical framework presented in this thesis provide opportunities to develop new coating formulas that can be co-cured with a single, lower-temperature bake and to identify specific features critical to achieving a smooth paint surface.

Contents

Acknowledgments

From my advisor James Sethian – I developed patience and fortitude; I grew to appreciate a difficult job done well. This PhD was my greatest challenge so far and I am a better man having completed it. Thank you Jamie.

From my mentor Robert Saye – I learned rigor and discipline; through your teaching, I discovered a rich and beautiful world of mathematics that I did not dream possible. I didn't know I could learn so much. Thank you Rob.

From the Berkeley professors Alexandre Chorin, Phillip Colella, Lin Lin, Michael Lindsey, Per-Olof Persson, and Jon Wilkening – I drew tremendous inspiration. Thank you for our discussions over the years.

From the Berkeley Lab scientists Jeffrey Donatelli, Zixi Hu, Dinesh Kumar, Kanupriya Pande, Will Pazner, and Matthew Zahr – I witnessed scientific excellence and achievement; we shared many laughs and I will miss our games. I'll win our fantasy football league one day, hopefully this year! Thank you all.

From my fellow graduate students Michael Franco, Yanhe Huang, Mikayla Kelley, Christopher Miller, Meredith Shea, Colin Wahl, and Meghan Turner – I found great strength and special friendships. Thank you for all the good times and all the laughs!

To my good friends Andrew Payne and Bryan Walsh – thank you for the support and the (many) hours of video games over the years! I hope the path is treating you well.

From my family, my father Ivan, mother Liz, brother Joey, sister Hannah, uncle Claude, and cousin Sam – I am blessed with love. You shaped me into the person I am today. I love you all.

A special thanks to my undergraduate advisor Luis Orozco – for setting me on my scientific journey; from you I learned compassion and kindness. I've yet to meet a better chef! Thank you Luis.

Thank you to our collaborators at PPG, Reza Rock, Xinyu Lu, and Brandon Petrouskie – for providing numerous insights into this challenging multi-physics problem.

Thank you to Vicky Lee and Kerri Peyovich for the administrative help over the years.

Thank you to everyone who's taught me a lesson.

— Luke P. Corcos

Chapter 1

Introduction

1.1 Coating flows

Thin liquid films covering a solid surface are ubiquitous in nature and industrial applications, ranging from nanofluidics and the macroscale flow of lava to lacquer spin coatings for compact discs and industrial paint lines. Understanding the coupled fluid flow and settling of multiple liquid film-layers is of particular interest for the control and design of modern coating systems.

For example, the high energy consumption of the automobile painting industry has garnered great interest in energy-efficient coating procedures. In typical automotive coating operations, a single film-layer (or coat) of paint is applied, the layer is then baked (or cured) in an oven, and the process is repeated for subsequent layers. The curing stages are particularly energy-intensive; automotive coating operations use 10,000s of gigawatt-hours each year in the U.S. alone [1], accounting for roughly 0.05% of the country's energy usage. Energy-efficient manufacturing lines that co-cure multiple layers of paint have the potential to eliminate curing stages and reduce energy consumption by 30% when compared to traditional coating methods [2]. However, the final film profile must adhere to the automobile industry's high standard for paint smoothness in order to provide the proper shine and color while also acting as a strong protective barrier. Figure 1.1 illustrates two seemingly similar paint coatings, but on closer inspection, the letters of the eye chart on the right are faded, the shine deadened, and the colors lack vibrancy when compared to the left. The left is acceptable while the right is not. Achieving a smooth, defect-free film of paint is often the biggest technical hurdle to commercializing energy-efficient multi-layer coating systems. In current practice, controlling the flow and leveling process is achieved through chemical means, and optimizing the process is done empirically for each new coating system. A high-fidelity mathematical and numerical framework that captures the complex physics driving fluid flow and leveling within the multi-layer system would provide opportunities to develop new coating formulas that can be co-cured with a single, lower temperature bake as well as identify specific features critical to achieving a smooth film surface, potentially leading towards the widespread adoption of these energy-efficient coating techniques.

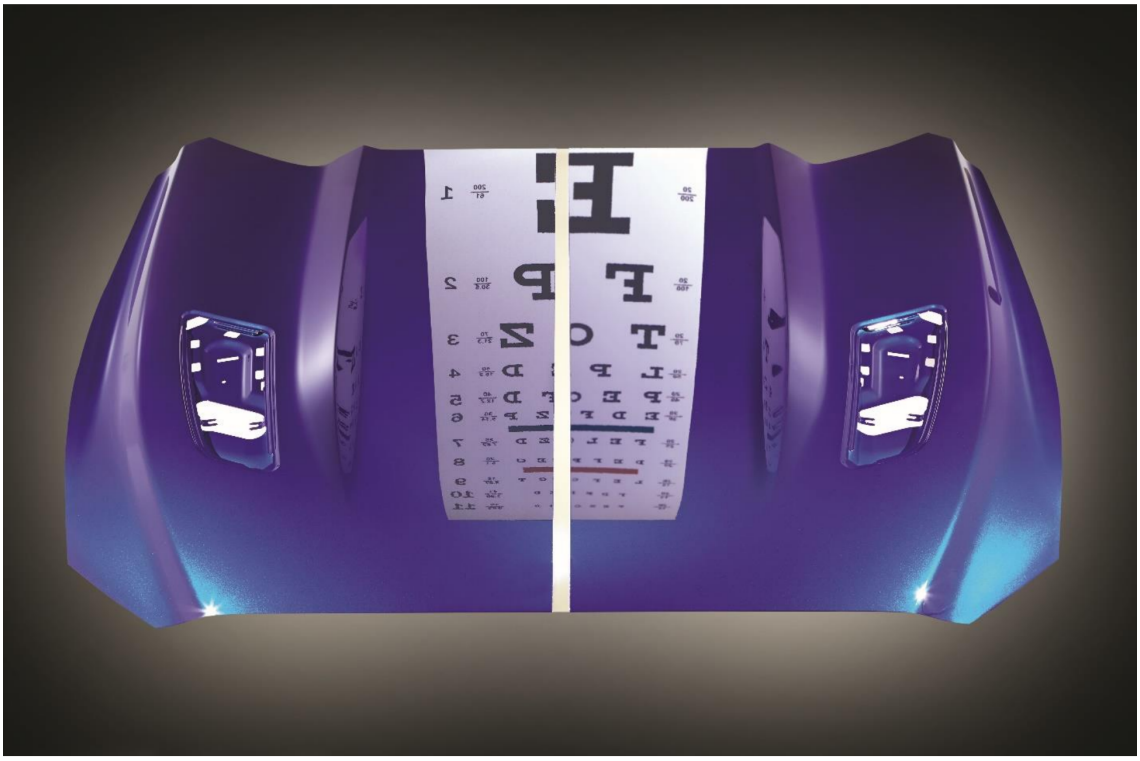


Figure 1.1: Achieving a smooth paint surface is critical to the automotive industry. The crispness of the left side is acceptable while the right is not.

The multi-layer coating flow problem has a range of coupled multi-physics, including multi-phase non-Newtonian fluid dynamics, mass transfer and interface recession from solvent evaporation, strong destabilizing surface forces such as Marangoni stresses, and intricate couplings between paint films. The final paint appearance is also impacted by many additional variables, including a constantly-evolving coating rheology as solvents evaporate and the coating cures, the orientation of pigment particles in the film, and the nature of the substrate and its orientation against the pull of gravity. The interplay between the layers and their multi-physical effects during the initial "flash" period has been shown experimentally by PPG Industries, Inc. to significantly impact the final appearance of the film after cure. The flash is the time, often at ambient temperature, between when the coating is applied and the time when the coating is heated and cured.

The focus of this thesis is the development of high-fidelity mathematical models and numerical methods to simulate the multi-layer coating flow problem, specifically in the micro-flow regime during the initial flash phase. A computational fluid dynamics (CFD) model that solves this problem must, among other aspects, accurately track interface motion and the flow of multiple phases of quasi-Newtonian fluid; the transport, diffusion, mixing, and evaporation of multiple solvents; and incorporate intricate boundary conditions and couplings

between paint layers. The complex multi-physics of the problem occur over vastly different length and time scales. Surface roughness has an amplitude of sub-microns to microns and a wavelength of millimeters, coating thickness is on the order of microns, and flash and cure are on the time scale of minutes to an hour. Additionally, the framework must adequately resolve the micro-scale solutal boundary layer dynamics crucial to capturing the Marangoni-driven flow of the multi-layer coating problem. To accurately solve the equations of motion necessitates the use of state-of-the-art high-order accurate numerical methods, and to fully resolve the various scales involved in the problem requires the use of high-performance computing and supercomputing resources.

The numerical methods presented in this thesis include hybrid finite difference level set methods and high-order accurate sharp-interface implicit mesh discontinuous Galerkin (DG) methods, newly developed local discontinuous Galerkin (LDG) solvers for Poisson problems with Robin boundary and jump conditions on implicitly-defined domains, and a finite difference surface gradient formulation for Marangoni stress calculations, as well as extensions of recently developed operator coarsening multigrid methods. Additionally, a new high-order accurate LDG formulation for Stokes problems with Navier-slip boundary conditions on implicitly-defined domains is presented in the appendix. Several components of our numerical framework are high-order accurate, while other components, whose dynamics do not require high-order methods, benefit from a simpler lower-order implementation. Our particular implementation choices lead to a fully coupled numerical algorithm for the multi-layer coating flow problem that is 2^{nd} order accurate in space and 1^{st} order in time.

The model is designed to predict the ultimate surface roughness of the multi-layer system. Results are presented in both 2D and 3D at industrially-relevant conditions, motivated in part by automotive paint coating applications. A high-fidelity 2D parametric study is performed to identify the key features impacting the surface profile and shows the formation of Marangoni vortices and plumes, long-wave deformational oscillatory surface modes, as well as structures resembling trees and flowers. Preliminary small-scale 3D results are presented, observing the formation of Marangoni sheets along with physically-consistent hexagonal surface patterns and Bènard cells. Several exploratory studies are also performed, examining multi-solvent evaporation, a Marangoni-driven drilling phenomenon within a multi-layer coating, and, lastly, a multi-layer interfacially turbulent matter cascade.

This work was performed in part with the Department of Energy's *HPC4Mfg* program and in collaboration with PPG Industries, Inc., who provided experimentally determined constitutive laws for how the film's rheological properties (viscosity, surface tension, etc.) evolve over time as solvents evaporate, as well as the roughness profiles of different substrates. The majority of the simulations were performed on the National Energy Research Scientific Computing Center's (NERSC's) *Cori* supercomputer. We continue this chapter by introducing the key physical processes present in the multi-layer coating flow problem.

1.2 Physical background

A multi-layer coating consists of several layers of liquid paint covering a solid surface. The motion of the paint films is intricately coupled and strong interfacial forces along the paint-paint and paint-gas surfaces drive fluid flow, during which solvents mix between paint layers and evaporate along the paint-gas interface, resulting in the solidification of all paint layers. Two of the key driving forces within the multi-layer coating flow problem are mass transfer from evaporation and Marangoni forces; these effects are discussed below. An illustration of the physics driving fluid flow within the multi-layer coating flow problem can be found in Figure 2.1.

1.2.1 The Marangoni effect and hydrodynamic instability

Marangoni forces are tangential surface forces brought on by surface tension gradients, most commonly caused by variations in surface temperature or species concentration. Put simply, fluid will flow from areas of low surface tension to areas of high surface tension since the higher surface tension pulls on the fluid more strongly. First reported by James Thomson while studying tears of wine [3] and later attributed to Carlo Marangoni [4], Marangoni forces are powerful interfacial phenomena prominent in many heat and mass transfer processes, including crystal growth [5][6], inkjet printing [7], the motion of bubbles [8], and thin film and paint coatings [9]–[12]. The dynamics of the tears of wine phenomena can be seen in Figure 1.2(b), where the evaporation of alcohol at the thin liquid layer along the glass results in a higher concentration of water in that region than in the bulk of the fluid. Water has a higher surface tension than alcohol, so wine is pulled up the glass by the Marangoni forces, and then driven down by gravity, resulting in the tears. Figure 1.2(c) illustrates the Marangoni effect in a similar setting, here a droplet of alcohol breaks itself apart due to the stronger surface tension forces at its perimeter [13].

The Marangoni effect was studied by Bénard in the context of thermally-driven flows heated from below [14], notably capturing the emergence of hexagonal-shaped circulation cells within the fluid. These cells can be seen in Figure 1.2(a). Later, the experiments of Block [15] and the mathematical analysis of Pearson [16] demonstrated that the flow patterns arise due to gradients in surface tension caused by variations in surface temperature. Pearson's analysis shows a fundamental stationary hydrodynamic instability in thermal Marangoni flows that produces cellular convection when the system's dimensionless Marangoni number exceeds a critical threshold. This became known as Bénard-Marangoni (or thermocapillary) convection. These are short-wave Marangoni instability modes that, for thin films, have a wavelength on the order of the film thickness [10]. Scriven and Sternling provided analysis for the case of these instabilities arising from a species concentration-dependent surface tension [17] and attributed the Marangoni effect as a mechanism for the spontaneous agitation of an interface between two fluids undergoing mass transfer, known as "interfacial turbulence" [18]. Scriven and Sternling showed the various regimes of stability for different rheological

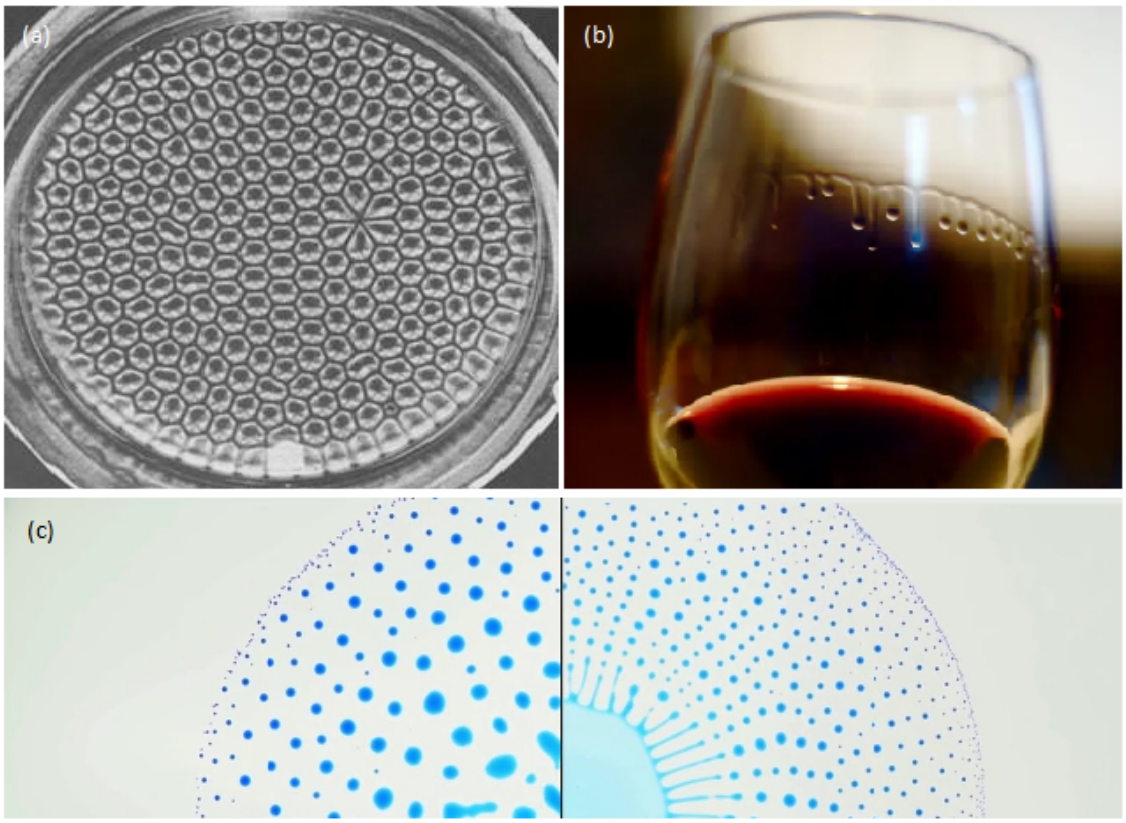


Figure 1.2: Examples of the Marangoni effect. **(a)** Hexagonal-shaped thermal Rayleigh-Bénard cells. **(b)** Tears of wine. **(c)** Alcohol droplet scattering, figure modified from [13].

parameters, such as viscosity and mass diffusivity, and for the direction of mass transfer, with some systems being unstable in one direction of mass transfer and stable in the reverse direction. These short-wave Marangoni instabilities are highlighted in the experimental results of Sherwood and Wei [19], where the transfer of hydrochloric acid from a solution of isobutyl alcohol across an interface into water containing ammonia produces spontaneous emulsification of the two solutions and droplets develop without the presence of any chemical reactions. Here the Marangoni effect also accelerates the rate of mass transfer when compared to pure diffusion, an effect first observed by Langmuir and Langmuir [20].

A second type of Marangoni hydrodynamic instability is the long-wave instability described by Scriven and Sternling in [21]; this phenomena appears when a free surface undergoing Marangoni-driven flow is allowed to deform. While Pearson's analysis in [16] assumes a flat free surface, Scriven and Sternling found that the presence of surface deformations permits oscillatory long-wave instabilities for which there is no critical Marangoni number, the strength of which is dependent on the mean value of surface tension. These modes were examined experimentally and numerically by van Hook in the context of thermal Marangoni

forces [22] and were found to induce significant deformation of the free surface, resulting in areas of local depression and elevation. These long-wave oscillatory deformational instabilities also exist in evaporating flows; the oscillatory nature of paint drying is described by Overdiep in [23] and is studied numerically in the context of lubrication theory in [24][25].

The presence of both instability types is well-established in the drying of thin films of paint [9][10][26]–[29], affecting the final paint smoothness and potentially leading to film defects, holes, tears, and surface corrugations. For reviews on the thermal Marangoni effect, see [6][30][31] and for reviews of the Marangoni effect specific to thin films, see [11][12][32]. The Marangoni effect is captured within our numerical algorithm for the multi-layer coating flow problem through the use of a tailored finite difference projection algorithm that calculates surface gradients along a height function to high-order accuracy, presented in section 4.3.2.

1.2.2 Evaporation and mass transfer

Evaporation is an endothermic vaporization process wherein the surface molecules of a liquid break from their bonds and transform into a gas. Evaporation causes mass to leave the liquid phase, corresponding to a loss of volume and inducing a downward recession of the liquid-gas interface. Within the multi-layer coating flow problem, the evaporation of solvent results in a high solute-concentration boundary layer along the paint-gas interface, the thickness of which depends on the solvents' mass diffusion coefficients. The thin boundary layer, in turn, leads to strong solutal concentration gradients along the surface, which produce strong Marangoni forces within the system. As previously mentioned, Marangoni forces may produce hydrodynamic instabilities [16][21], generate circulating Bénard cells [14], transport material, and act to replenish the evaporation process [20]. Therefore within the multi-layer coating flow problem, the degree of mass diffusion, the rate of solvent evaporation, and the strength of the Marangoni forces are all intricately coupled. The numerical framework for the multi-layer coating flow problem accurately handles the evaporation of multiple solvents through the use of a newly developed high-order accurate local discontinuous Galerkin solver for Poisson problems with Robin boundary conditions, presented in section 3.3, and captures the corresponding interface recession via the level set method.

1.3 Previous work

A large body of work has been performed to numerically model evaporating flows, as well as solve the associated Robin boundary problem along a moving interface. These include arbitrary Euler-Lagrange (ALE) finite element methods [33]–[35], where the computational mesh is deformed to align with the interface and the Robin boundary condition is naturally handled by the finite element method's weak formulation. Finite volume methods combined with level set methods [36][37] and ghost fluid methods [38] have been applied to vaporizing two-phase flows [39][40]. In this setting, boundary conditions are applied along an implicitly-defined interface via extrapolation onto fictitious cells or nodes, with the cut-cell Robin solver

of Papac [41] being one of the first methods for applying Robin boundary conditions within the level set framework. Ghost-cell finite difference methods [42]–[44] have recently been developed and applied to similar flow problems. Here extrapolation onto ghost nodes is combined with a cell-wise calculation of the normal derivative to impose Robin boundary conditions along interfaces implicitly defined by the level set method.

In recent years, a number of numerical studies on the Marangoni effect have been performed, particularly in the context of thermocapillary convection. Interface tracking finite element methods have widely been used to model the Marangoni effect within evaporating micro-droplets [45]–[47], with these and other boundary-fitting methods [48] allowing for accurate implementation of the Marangoni effect along surfaces with limited deformation. An alternative approach is to capture the interface implicitly in a fully-Eulerian framework. For example, a common technique used in the volume of fluid (VOF) [49][50] and level set frameworks [51]–[53] is to apply a diffuse-interface approach, whereby interfacial jump conditions are replaced by locally-smoothed Dirac delta forcing terms added to the right-hand-side of the Navier-Stokes momentum equations. Another notable work is that of Köllner et al. [54], which examines the behavior of short-wave solutal Marangoni instabilities and the onset of interfacial turbulence using pseudo-spectral methods along a fixed flat interface. Their work, in collaboration with experiments [55], classifies the various complex patterns arising from short-wave Marangoni instabilities into different hierarchical spatial structures, including associated merging/coarsening processes. These results were replicated in the work of Yiantsios [35], in which an ALE finite element method captures the formation and merger of solutal Marangoni plumes/roll cells during the drying of polymer solutions.

1.4 New contributions

This thesis presents a new multi-physics mathematical framework tailored to multi-layer coating flows, in combination with a hybrid numerical framework consisting of finite difference level set methods and high-order accurate sharp-interface implicit mesh discontinuous Galerkin methods. Our mathematical model captures a coupled set of multi-physics that includes multi-phase quasi-Newtonian fluid dynamics; mass transfer and interface recession from solvent evaporation; and intricate interfacial forces of surface tension and Marangoni stresses. The numerical methods developed in this work include local discontinuous Galerkin solvers for Poisson problems with Robin boundary and jump conditions on implicitly-defined domains, to capture solvent evaporation; a tailored finite difference projection algorithm that calculates surface gradients, to robustly and accurately incorporate Marangoni stresses; and a coupled multi-physics time stepping approach that incorporates all the different solvers at play. Additionally, the numerical framework uses state-of-the-art Stokes solvers that integrate concentration-dependent rheological parameters for quasi-Newtonian interface dynamics. Using this model and numerical framework, we capture several experimental findings and study the formation of Marangoni plumes and Bénard cells, the impact of long-wave

deformational surface modes on immersed interfaces, and the emergence of the final multi-layer film profile. Parts of this thesis have been adapted from the following jointly-authored article, currently under review:

- L. P. Corcos, R. I. Saye, & J. A. Sethian, A hybrid finite difference level set–implicit mesh discontinuous Galerkin method for multi-layer coating flows, *Journal of Computational Physics*

The presentation here includes an expanded discussion of the physics of multi-layer coatings, our hybrid numerical framework, and of the LDG Robin solvers, as well as additional numerical studies. Included are studies on multi-solvent evaporative paint dynamics, the flow and leveling of multi-layer automobile paint coatings in both 2D and 3D—presenting the results of a parametric study performed at industrially-relevant conditions, the formation of tree-like Marangoni plumes, and an examination of interfacial turbulence within a multi-layer matter cascade. Through the developed mathematical model and accompanying numerical framework, this work revealed many of the driving mechanisms underlying multi-layer coating flow dynamics. We also present a novel high-order accurate LDG method for Stokes problems with Navier-slip boundary conditions on implicitly-defined domains in the appendix.

1.5 Outline

The outline of this thesis is as follows: In Chapter 2, we introduce the relevant background information for the multi-layer coating flow problem, including the domain of interest, the equations of motion, and a brief description of the level set method. In Chapter 3, we describe discontinuous Galerkin methods (DG) and operator coarsening multigrid methods for Poisson problems with Robin boundary and jump conditions. In Chapter 4, the hybrid numerical methods for solving the multi-layer coating flow problem are presented, including the results of a convergence study. The results of our simulations of multi-layer coatings are shown in Chapter 5. In Chapter 6, we conclude the discussion of the multi-layer coating flow problem and examine future directions. In Appendix A, an LDG method for Stokes problems with Navier-slip boundary conditions is presented. The rheological parameters used in the numerical simulations can be found in Appendix B. Additional images of evaporative Marangoni flow are shown in Appendix C.

Chapter 2

Background

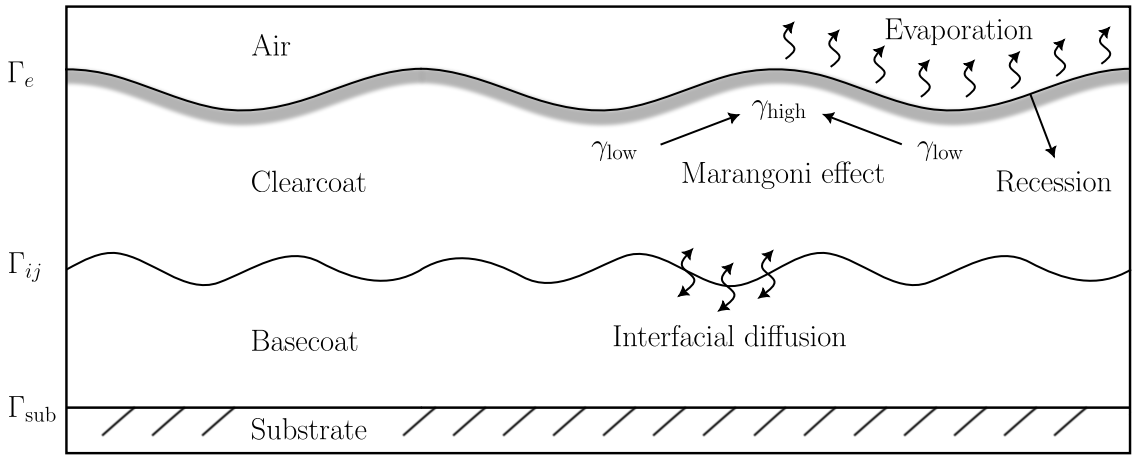


Figure 2.1: The two-layer coating flow problem in 2D, consisting of four subdomains separated by three interfaces. Highlighted are the key physical effects driving the dynamics of the multi-layer coating flow problem.

In the setting of multi-layer coating flow, one or more film-layers of basecoat paint are applied onto a plastic or metallic substrate. The basecoats provide the color, pigment particles, and shine of the automobile. Afterwards, a final transparent clearcoat layer is sprayed on top of the basecoats to provide a protective coating. Our focus in this work is the case when all film-layers are cured simultaneously. The multi-layer coating flow problem therefore has a domain of interest consisting of four subdomains—the substrate, the basecoats, the clearcoat, and the air—separated by the following interfaces:

- Γ_{sub} The top of the substrate.
- Γ_{ij} The middle embedded paint-paint surfaces separating the coats of paint.
- Γ_e The evaporative surface between the clearcoat and the air.

The partial differential equations (PDEs) describing the multi-layer coating flow problem are relevant within the fluid phases. In this model, the dynamics of the air are assumed

to have a negligible effect. Therefore, the computational domain is restricted to the layers of liquid paint and a free-surface model is used for the evaporative surface Γ_e. We define the d-dimensional multi-phase liquid domain $\Omega \subset \mathbb{R}^d$ as $\Omega = \bigcup_i \Omega_i$, where Ω_i represents the region of paint layer i, with $\Gamma_{ij} = \Omega_i \bigcap \Omega_j$. The domain is taken to be periodic along all horizontal dimensions (x in 2D; x, y in 3D) and we make the simplifying assumption that the interfaces never cross. Together these two assumptions alleviate the numerical difficulties of contact line and triple point dynamics; the extension of this framework to more general situations is discussed in the conclusion. An example domain for the case of two paint layers is shown in Figure 2.1.

The coating flow problem is in the micro-flow regime, i.e., the coats of paint are very thin. The initial thicknesses of the basecoats range from $30-50\mu$m and that of the clearcoat ranges from $50 - 100\mu$m, about the width of two human hairs. The length scale in the horizontal direction spans the entirety of the automobile and the typical "roughness" wavelengths along the surface of a dried paint film range from $1 - 10$mm. To properly capture these long-wave modes, we take the maximum horizontal length of our computational domain in 2D to be 25.6mm, which results in simulations on long, skinny domains with aspect ratios up to 256×1.

2.1 Equations of motion

The model for the multi-layer coating flow problem must incorporate numerous physical effects, including quasi-Newtonian fluid dynamics; transport, diffusion, and mixing of multiple dissolved species; mass transfer and interface recession from solvent evaporation; a constantly-evolving coating rheology; intricate interfacial forces of surface tension and Marangoni stresses on paint-gas and paint-paint interfaces and their coupling; and substrate roughness and the pull of gravity. The model developed in this work is purely isothermal, however, the effects of temperature could be incorporated. Additionally, the pigment flakes/particles within the basecoat paints are ignored as the large size of the flakes with respect to the film thickness would necessitate the use of fluid-structure interaction models, which were determined to be outside the scope of this work.

To determine the equations of motion, first consider a single film-layer of liquid paint composed of multiple solvents dissolving an underlying resin (also known as the solid or solute). As the paint dries, the solvents evaporate into the air, thereby causing the liquid volume to shrink, the liquid-gas interface to recess, and eventually the paint to solidify. When the paint is fully dry, only the resin remains and any imprints or irregularities caused by, or occurring during, the drying process will become permanent. The dynamics of the single-layer coating flow problem therefore requires solving for:

(i) The motion of each of the solvents within the paint, as well as their evaporation at the paint-gas interface. This describes the conservation of species within the system.

(ii) The motion and downward recession of the paint-gas interface. The degree of recession is dictated by conservation of volume.

(iii) The fluid dynamics of the liquid paint, capturing both conservation of mass and momentum.

(iv) The solidification of the paint as a function of resin concentration. This introduces non-Newtonian dynamics to the fluid.

The effect of buoyancy is small in the micro-flow regime and, therefore, a constant paint density is assumed in this model. The fluids are considered quasi-Newtonian, governed by the incompressible Navier-Stokes equations (2.1,2.2) with the viscosity varying with respect to resin concentration. Should the densities of the components comprising the paints not be equal, the paints could be modeled as quasi-incompressible fluids, however, the boundary conditions associated with conservation of mass and volume become more intricate in such a setting. In our model, the solidification process is captured by the exponential nature of the viscosity profile with respect to resin concentration, described in Appendix B. We consider PPG-provided viscosity curves, typically setting the higher viscosities to the basecoats and the lower to the clearcoat. Species and energy transfer are modeled by the convection-diffusion equations (2.3,2.4), where the convection is driven by the underlying fluid. The initial flash phase of drying occurs at ambient room temperature and our model is purely isothermal. The equations of motion, boundary, and jump conditions describing conservation of energy are included for presentation and the effects of temperature could be incorporated. The following conservation laws hold in the bulk of the liquid paint (in Ω) for both the single-layer and multi-layer coating flow problems:

$$\rho_t + \nabla \cdot (\rho \boldsymbol{u}) = 0 \qquad \text{Mass} \qquad (2.1)$$

$$(\rho \boldsymbol{u})_t + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla p + \nabla \cdot (\mu(c_R)(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)) + \rho \boldsymbol{g} \qquad \text{Momentum} \qquad (2.2)$$

$$(\rho c_k)_t + \nabla \cdot (\rho c_k \boldsymbol{u}) = \nabla \cdot (\rho D_k \nabla c_k) \qquad \text{Species} \qquad (2.3)$$

$$(\rho C_p T)_t + \nabla \cdot (\rho C_p T \boldsymbol{u}) = \nabla \cdot (\lambda \nabla T) \qquad \text{Energy/Heat} \qquad (2.4).$$

The variables and material properties of the multi-layer coating flow problem are shown in Table 2.1. Note that our equations of motion demonstrate a direct coupling between the motion of the solvents and the fluid dynamics within the bulk of the paint. The resin mass concentration is a function of the solvents, with $c_R = 1 - \sum_k c_k$, and the rheological parameters of viscosity and surface tension are scalar functions of the resin concentration. Our framework is capable of handling multiple solvents and an arbitrary number of paint layers; the numerical studies performed in this work consider up to three-solvent paints and up to three film-layers. The solvents may differ in both their mass diffusion coefficients as well as their evaporation rates. For simplicity, we assume that each solvent's mass diffusion coefficient is constant within each paint layer, but allow for different values in different layers. Both the variables and material properties may exhibit jumps at a material boundary, i.e., at the paint-gas or paint-paint surfaces. Therefore, to ensure conservation, we must examine the boundary and jump conditions present at these interfaces.

Symbol	Property	SI Units
ρ	Fluid density	kg m^{-3}
$\boldsymbol{u}$	Fluid velocity	m s^{-1}
p	Pressure	Pa
μ	Dynamic viscosity	Pa s
γ	Surface tension coefficient	N m^{-1}
c_R	Resin mass concentration	Unitless
c_k	Mass concentration of solvent k	Unitless
D_k	Mass diffusion coefficient for solvent k	m^2 s^{-1}
m	Total mass transfer rate	kg m^{-2} s^{-1}
m_k	Mass transfer rate for solvent k	kg m^{-2} s^{-1}
T	Temperature	K
C_p	Specific heat	J kg^{-1} K^{-1}
λ	Thermal conductivity	W m^{-1} K^{-1}
$\boldsymbol{g}$	Gravity	kg m s^{-2}

Table 2.1: Variables and coefficients of the multi-layer coating flow problem.

2.1.1 Interfacial jump conditions

In this section, interfacial jump conditions across a moving surface are first derived for a general conservation law and then applied to the conservation laws of the multi-layer coating flow problem. Similar derivations can be found in [56][57]. Consider a d-dimensional domain $\Omega \subset \mathbb{R}^d$ consisting of two separate phases i and j, with $\Omega = \Omega_i \bigcup \Omega_j$. Take $\Gamma = \Omega_i \bigcap \Omega_j$ to be the orientable codimension-one surface separating the two phases. Assume that Γ is sufficiently smooth. Consider the general conservation law for quantity a with flux $\boldsymbol{F}(a, \nabla a)$ and source f:

$$a_t + \nabla \cdot \boldsymbol{F} = f \quad \text{in } \Omega. \tag{2.5}$$

Now consider a subset $U \subset \Omega$ that contains a simple smooth section of the interface Γ. Denote this section as ζ and assume that ζ divides U into two simply connected regions $U_i \subset \Omega_i$ and $U_j \subset \Omega_j$. The total time derivative of quantity a over U is found using the Leibniz integral rule[1]

$$\frac{d}{dt}\int_{U(t)} a = \int_{U(t)} a_t + \int_{\partial U(t)} a\boldsymbol{\nu} \cdot \boldsymbol{n}, \tag{2.6}$$

where $\boldsymbol{\nu}$ is the velocity of surface ∂U and $\boldsymbol{n}$ is the outward-facing normal vector. Plugging (2.5) into (2.6) and applying the divergence theorem gives

$$\frac{d}{dt}\int_{U} a = \int_{\partial U} a\boldsymbol{\nu} \cdot \boldsymbol{n} - \int_{\partial U} \boldsymbol{F} \cdot \boldsymbol{n} + \int_{U} f. \tag{2.7}$$

[1]For brevity, we drop the measure of integration, which should be clear from the specified domain of integration. i.e., volume or surface.

Also, splitting U into U_i and U_j transforms (2.6) into

$$\frac{d}{dt}\left(\int_{U_i}+\int_{U_j}\right)a=\left(\int_{U_i}+\int_{U_j}\right)a_t+\int_{\partial U_i}a_i\boldsymbol{\nu}\cdot\boldsymbol{n}+\int_{\partial U_j}a_j\boldsymbol{\nu}\cdot\boldsymbol{n}, \tag{2.8}$$

where a_i and a_j are the limiting values of a restricted from phases i and j respectively. Again, plugging in the conservation law (2.5) and applying the divergence theorem gives

$$\frac{d}{dt}\left(\int_{U_i}+\int_{U_j}\right)a=\int_{\partial U_i}(a_i\boldsymbol{\nu}-\boldsymbol{F}_i)\cdot\boldsymbol{n}+\int_{\partial U_j}(a_j\boldsymbol{\nu}-\boldsymbol{F}_j)\cdot\boldsymbol{n}+\int_U f'+\int_\zeta f_S, \tag{2.9}$$

where f' is a modified volumetric source and f_S is a surface source with support on surface Γ. Expanding the right-hand side of (2.9) sans source terms gives

$$\left(\int_{\partial U_i\setminus\zeta}+\int_\zeta\right)(a_i\boldsymbol{\nu}-\boldsymbol{F}_i)\cdot\boldsymbol{n}+\left(\int_{\partial U_j\setminus\zeta}+\int_\zeta\right)(a_j\boldsymbol{\nu}-\boldsymbol{F}_j)\cdot\boldsymbol{n}.$$

Note that $\partial U=(\partial U_i\setminus\zeta)\bigcup(\partial U_j\setminus\zeta)$ so the above is equal to

$$\int_{\partial U}(a\boldsymbol{\nu}-\boldsymbol{F})\cdot\boldsymbol{n}+\int_\zeta(a_i\boldsymbol{\nu}-\boldsymbol{F}_i)\cdot\boldsymbol{n}-(a_j\boldsymbol{\nu}-\boldsymbol{F}_j)\cdot\boldsymbol{n}, \tag{2.10}$$

where we've modified the definition of the normal vector $\boldsymbol{n}$ on ζ to point from phase i into phase j. Plugging (2.10) into (2.9) gives

$$\frac{d}{dt}\int_U a=\int_{\partial U}(a\boldsymbol{\nu}-\boldsymbol{F})\cdot\boldsymbol{n}+\int_\zeta(a_i-a_j)\boldsymbol{\nu}\cdot\boldsymbol{n}-(\boldsymbol{F}_i-\boldsymbol{F}_j)\cdot\boldsymbol{n}+\int_U f'+\int_\zeta f_S. \tag{2.11}$$

Subtracting (2.7) from (2.11) results in

$$\int_\zeta[a\boldsymbol{\nu}\cdot\boldsymbol{n}-\boldsymbol{F}\cdot\boldsymbol{n}]=\int_U(f-f')-\int_\zeta f_S, \tag{2.12}$$

where $[a]=a_i-a_j$ denotes the jump in a across surface ζ. Take the limit of U smoothly collapsing into ζ. As a result, the volume terms vanish, giving the following interfacial jump condition on ζ

$$\int_\zeta[\boldsymbol{F}\cdot\boldsymbol{n}-a\boldsymbol{\nu}\cdot\boldsymbol{n}]=\int_\zeta f_S. \tag{2.13}$$

Since the choice of $\zeta\subset\Gamma$ is arbitrary, we may drop the integrals. Therefore (2.5) has the following jump condition of Rankine-Hugoniot type at interface Γ

$$[\boldsymbol{F}\cdot\boldsymbol{n}-aV]=f_S, \tag{2.14}$$

where $V=\boldsymbol{\nu}\cdot\boldsymbol{n}$ is the interface velocity in the normal direction, and again f_S is a surface force or source term. For problems where quantity a is transported by velocity field $\boldsymbol{u}$, $\boldsymbol{F}(a,\nabla a)=a\boldsymbol{u}+\boldsymbol{J}(a,\nabla a)$ and the following bulk conservation law and associated Rankine-Hugoniot jump condition hold:

$$a_t+\nabla\cdot(a\boldsymbol{u})+\nabla\cdot\boldsymbol{J}=f \quad \text{in } \Omega \tag{2.15}$$

$$[a(\boldsymbol{u}\cdot\boldsymbol{n}-V)+\boldsymbol{J}\cdot\boldsymbol{n}]=f_S \quad \text{on } \Gamma. \tag{2.16}$$

2.1.2 Mass conservation

Applying (2.16) to conservation of mass and the continuity equation of the Navier-Stokes equations (2.1) dictates that

$$[\rho(\boldsymbol{u}\cdot\boldsymbol{n}-V)]=0 \tag{2.17}$$

along a general moving surface and implies that the total mass flux m across the interface is given by

$$m=\rho(\boldsymbol{u}\cdot\boldsymbol{n}-V). \tag{2.18}$$

Here, the values of ρ and $\boldsymbol{u}$ may be determined by their restriction to either phase. Rearranging (2.18) provides an expression for the interface velocity that includes the fluid velocity and the motion induced by mass transfer

$$V=\boldsymbol{u}\cdot\boldsymbol{n}-m/\rho. \tag{2.19}$$

As solvents leave the domain at the free evaporative surface Γ_e, the effect of (2.19) is to produce a recession of the interface, thereby reducing the bulk liquid volume while also ensuring conservation of mass. Equation (2.19) is also applicable to the motion of the embedded paint-paint surfaces Γ_{ij}, where mass transfer between two paints causes one film-layer to shrink while the other swells. Paint cannot pass through the substrate, therefore (2.18), combined with a no-slip assumption sets $\boldsymbol{u}=0$ on Γ_{sub}.

In general, when there is mass transfer across an interface and the densities of the two phases differ, (2.17) and (2.18) produce a discontinuity in the fluid's normal velocity at the interface as a consequence of Stefan flow. The jump in the tangential velocity can be taken to be zero under a no-slip assumption; the jump in the fluid velocity field across a surface dictated by conservation of mass is then

$$[\boldsymbol{u}]=\left[\frac{1}{\rho}\right]m\boldsymbol{n}, \tag{2.20}$$

therefore when the densities of the two phases are equal, the fluids stick together and there is no jump in the velocity field. Our assumption of a constant paint density sets $[\boldsymbol{u}]=0$ on the embedded paint-paint surfaces Γ_{ij}. Lastly, since the fluids are incompressible and their density constant, the conservation of mass equation (2.1) within the incompressible Navier-Stokes equations reduces to the divergence-free constraint

$$\nabla\cdot\boldsymbol{u}=0. \tag{2.21}$$

2.1.3 Momentum conservation

The conservation condition (2.16) for the incompressible Navier-Stokes momentum equation (2.2) produces the following jump in stress

$$[\rho\boldsymbol{u}(\boldsymbol{u}\cdot\boldsymbol{n}-V)+p\boldsymbol{n}-\mu(\nabla\boldsymbol{u}+\nabla\boldsymbol{u}^T)\cdot\boldsymbol{n}]=\boldsymbol{f}_S. \tag{2.22}$$

The first term on the left-hand side of (2.22) is the stress caused by mass transfer and the latter two terms represent the stress-tensor (defined as: $\boldsymbol{\sigma} = -p\mathbb{I} + \mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)$) acting on the surface normal. Here, the right-hand side $\boldsymbol{f}_S$ captures the surface forces within the system. Using (2.18) and (2.20) gives an expression for the stress caused by mass transfer

$$[\rho \boldsymbol{u}(\boldsymbol{u} \cdot \boldsymbol{n} - V)] = m[\boldsymbol{u}] = \left[\frac{1}{\rho}\right] m^2 \boldsymbol{n}. \tag{2.23}$$

This term is negligible in the micro-flow regime due to the quadratic nature with respect to m and is included only for presentation. Plugging (2.23) into (2.22) gives a general expression for the jump in stress

$$[\boldsymbol{\sigma} \cdot \boldsymbol{n}] = \left[\frac{1}{\rho}\right] m^2 \boldsymbol{n} + \boldsymbol{f}_S. \tag{2.24}$$

The surface forces within the multi-layer coating flow problem are the forces of surface tension and Marangoni stresses, respectively setting $\boldsymbol{f}_S = -\gamma\kappa\boldsymbol{n} + \nabla_S\gamma$ along the embedded paint-paint surfaces Γ_{ij} and free evaporative surface Γ_e. These forces produce the following stress jump condition

$$[\boldsymbol{\sigma} \cdot \boldsymbol{n}] = \left[\frac{1}{\rho}\right] m^2 \boldsymbol{n} - \gamma\kappa\boldsymbol{n} + \nabla_S\gamma, \tag{2.25}$$

where γ is the coefficient of surface tension, κ is mean curvature of the interface, and $\nabla_S = (\mathbb{I} - \boldsymbol{n} \otimes \boldsymbol{n})\nabla$ is the surface gradient operator. As motivated in the introduction, surface tension gradients, or Marangoni forces, are powerful tangential forces that may produce short-wave and long-wave hydrodynamic instabilities within the system. In the multi-layer coating flow problem, Marangoni forces are caused by species concentration variations along the surface, with the surface tension coefficient being a function of resin mass concentration, i.e., $\gamma = \gamma(c_R)$. Therefore by the chain rule, Marangoni forces are of the form $\nabla_S\gamma = \frac{d\gamma}{dc_R}\nabla_S c_R$. A simple high-order accurate finite difference method for calculating the surface gradients within our problem is presented in section 4.3.2. Along free evaporative surface Γ_e, the stress from the gas phase is negligible and (2.25) reduces to the following stress boundary condition

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = -p_{\text{ext}}\boldsymbol{n} + \left[\frac{1}{\rho}\right] m^2 \boldsymbol{n} - \gamma\kappa\boldsymbol{n} + \nabla_S\gamma, \tag{2.26}$$

where p_{ext} is an external pressure, here set to atmospheric.

2.1.4 Species conservation

Applying (2.16) to the solvent convection-diffusion equations (2.3) and assuming that the system is free of chemical reactions gives the following expression for conservation of species across a general moving interface

$$m[c_k] - [\rho D_k \nabla c_k \cdot \boldsymbol{n}] = 0. \tag{2.27}$$

Expanding for each phase gives an equivalent Robin boundary condition for the solvent mass concentration c_k

$$m_k = mc_k - \rho D_k \nabla c_k \cdot \boldsymbol{n}, \tag{2.28}$$

where m_k is the mass flux for solvent k and $m = \sum_k m_k$. In the multi-layer coating flow problem, the solvent mass loss from evaporation is captured by (2.28) along the free evaporative surface Γ_e. The specific value for evaporative mass flux m_k—which is dependent on the solvent mass concentration at the interface—is explained in more detail in section 4.2.1. Section 3.3 presents newly developed LDG solvers for Poisson problems with Robin boundary conditions that calculate high-order accurate solutions on implicitly-defined domains in a dimension-independent fashion, for a wide range of variable diffusion and Robin coefficients. Meanwhile, the level set method captures the interface recession from evaporation by embedding the computed evaporation rate values into its speed law (2.19). Jump condition (2.27) also holds along the embedded paint-paint surfaces Γ_{ij}, however here we simplify[2] by enforcing continuity of the solvent mass concentration profile, setting $[c_k] = 0$ and reducing (2.27) to the standard jump condition $[D_k \nabla c_k \cdot \boldsymbol{n}] = 0$, noting again a constant paint density ρ. Summing (2.28) for each solvent gives the following expression for the total mass flux between paints at the embedded surfaces Γ_{ij}

$$m = -\frac{\rho}{c_R} \sum_k D_k \nabla c_k \cdot \boldsymbol{n}. \tag{2.29}$$

Using (2.29) in the interface velocity equation (2.19) ensures that the resins of the basecoat and clearcoat paints do not mix and that conservation of mass, volume, and species is respected between paint layers. The solvent convection-diffusion system (2.3) is closed by applying the no-penetration condition $\nabla c_k \cdot \boldsymbol{n} = 0$ along the substrate Γ_{sub}.

2.1.5 Energy conservation

In this work, the evaporative process is isothermal and the effects of temperature are not included, however for completeness, using (2.16) for energy conservation (2.4) gives

$$[\rho C_p T(\boldsymbol{u} \cdot \boldsymbol{n} - V) - \lambda \nabla T \cdot \boldsymbol{n}] = 0$$

$$Lm + [\lambda \nabla T \cdot \boldsymbol{n}] = 0, \tag{2.30}$$

where L is the latent heat of evaporation. Equation (2.30) is the classic Stefan condition. A reasonable assumption is that the temperature is continuous across the interface ($[T] = 0$) and equal to the saturation temperature T_{sat}, which may depend on species concentration. Along the substrate, a Neumann or Dirichlet boundary condition may be applied to close the energy system.

[2] A closure condition specifying the jump $[c_k]$ is required for the Robin-style jump condition (2.27). The specific value of this jump is, in general, unknown for multi-layer coatings and set to zero in this work. A new high-order accurate LDG method for solving Poisson problems with Robin jump conditions on implicitly-defined domains is presented in section 3.3.

2.1.6 Summary of equations

In summary, here are the equations of motion, boundary, and jump conditions for the multi-layer coating flow problem:

In Ω

$$
\begin{aligned}
&(\rho \boldsymbol{u})_t + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla p + \nabla \cdot (\mu(c_R)(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)) + \rho \boldsymbol{g} \\
&\nabla \cdot \boldsymbol{u} = 0 \\
&(c_k)_t + \nabla \cdot (c_k \boldsymbol{u}) = \nabla \cdot (D_k \nabla c_k) \\
&c_R = 1 - \sum_k c_k.
\end{aligned}
$$

On Γ_e

$$
\begin{aligned}
&V = \boldsymbol{u} \cdot \boldsymbol{n} - \frac{1}{\rho} m \\
&\boldsymbol{\sigma} \cdot \boldsymbol{n} = -p_{ext} \boldsymbol{n} - \gamma \kappa \boldsymbol{n} + \nabla_S \gamma \\
&m_k = m c_k - \rho D_k \nabla c_k \cdot \boldsymbol{n} \\
&m = \sum_k m_k.
\end{aligned}
$$

On Γ_{ij}

$$
\begin{aligned}
&V = \boldsymbol{u} \cdot \boldsymbol{n} - \frac{1}{\rho} m \\
&[\boldsymbol{u}] = 0 \\
&[\boldsymbol{\sigma} \cdot \boldsymbol{n}] = -\gamma \kappa \boldsymbol{n} + \nabla_S \gamma \\
&[c_k] = 0 \\
&[D_k \nabla c_k \cdot \boldsymbol{n}] = 0 \\
&m = -\frac{\rho}{c_R} \sum_k D_k \nabla c_k \cdot \boldsymbol{n}.
\end{aligned}
$$

On Γ_{sub}

$$
\begin{aligned}
&\boldsymbol{u} = 0 \\
&\nabla c_k \cdot \boldsymbol{n} = 0.
\end{aligned}
$$

2.2 The level set method

Key to the multi-layer coating flow problem is capturing the motion of the embedded paint-paint surfaces Γ_{ij} and the free evaporative surface Γ_e. The level set method of Osher and Sethian [36][37] provides a robust, powerful, and accurate technique for capturing the motion of evolving interfaces. In the level set framework, interfaces are defined implicitly as the zero isosurfaces of a higher dimensional level set function—typically a signed distance function—and the evolution of the level set function under a specified speed law defines the motion of the interfaces. The level set method has many advantages over traditional front-tracking methods including naturally handling topological changes, avoiding the bunching of interface-defining particles, and a simple dimension-independent implementation. The level set method has been used in numerous physical applications [37][58] and is an ideal framework for capturing surface evolution in the multi-layer coating flow problem. In this section, we highlight the basics of the level set method, including our specific implementation choices, and discuss the role of finite difference methods within our hybrid numerical framework for the multi-layer coating flow problem.

To illustrate the fundamentals of the level set method, consider the trajectory $x(t)$ of a

particle on an interface $\Gamma(t)$, which is moving with speed F in the normal direction $\boldsymbol{n}$. Assume Γ is a sufficiently smooth orientable codimension-one surface in $\mathbb{R}^d$. Define the *Level Set Function* $\phi : \mathbb{R}^d \times [0,T] \to \mathbb{R}$ such that $\phi(x(t),t) = 0$ for all time $t \in [0,T]$. Take the total temporal derivative of the level set function and apply the chain rule to get

$$\phi_t + \dot{x}(t) \cdot \nabla\phi = 0,$$

where the trajectory's temporal derivative $\dot{x}$ is equal to the velocity field $F\boldsymbol{n}$. Using $\boldsymbol{n} = \nabla\phi/|\nabla\phi|$ gives the *Level Set Evolution Equation*:

$$\phi_t + F|\nabla\phi| = 0. \tag{2.31}$$

With (2.31), we have recast the problem of interface motion into the evolution of a PDE of Hamilton-Jacobi type under *Speed Function* F, which may be solved in an Eulerian framework with simple standard finite difference methods [37]. The level set function implicitly defines the interface $\Gamma(t) = \{x \in \mathbb{R}^d \mid \phi(x,t) = 0\}$, the two regions $\Omega^-(t) = \{x \in \mathbb{R}^d \mid \phi(x,t) < 0\}$ and $\Omega^+(t) = \{x \in \mathbb{R}^d \mid \phi(x,t) > 0\}$, and their subsequent evolution under speed F. The speed function F may be coupled to the underlying physics driving the motion of the interface. For example, in the multi-layer coating flow problem, the motion of the embedded paint-paint surfaces Γ_{ij} and the free evaporative surface Γ_e is determined by (2.19), evolving under the paints' fluid velocity field and the motion induced by mass transfer.

An example of the level set method is shown in Figure 2.2; here a circular front in 2D (on the left) is expanding at a constant speed. The front is captured by the intersection of a higher dimensional surface (the cone on the right) and the xy plane. In this example, the cone is the level set function, the xy plane is the set of points with zero height, and their intersection is the zero level curve. The motion of the circular front is captured entirely through the evolution of the cone-shaped level set function and its interaction with the xy plane, e.g., the downward motion of the cone widens its intersection with the plane, corresponding to the expansion of the circular front. It is important to note here that the level set method is spatially dimension-independent and applicable to more intricate surface motion than this simple example. The level set method naturally handles geometric surface motion and motions determined by external physics, as well as topological changes—i.e., the mergers and separations of fronts.

The level set method is often referred to as an *Initial Value Formulation* for interface motion, i.e., the initial interface location $\Gamma(t=0)$ provides the data for an initial level set function ϕ_0; the ensuing motion of the front is then captured by the evolution equation (2.31). Note that while the initial level set function specifying $\Gamma(t=0)$ is not unique, the evolution of the surface under speed F is independent of ϕ_0. In practice, a well-behaved level set function is the *Signed Distance Function*. A signed distance function ensures that $|\nabla\phi| = 1$ for all $x \in \mathbb{R}^d$, i.e., the different level curves of ϕ are evenly spaced and not bunched together. The level set method requires all level sets—or at least all level sets in a given *Narrow Band*

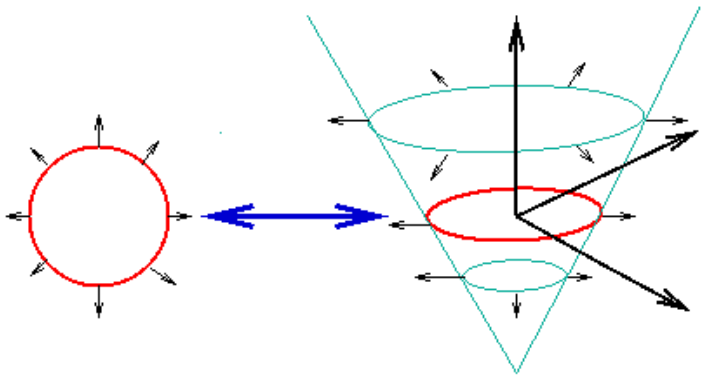


Figure 2.2: An example of the level set method. The red circle on the left is the front of interest. Its motion is determined by the evolution of the zero isosurface of the cone-shaped level set function on the right. Image reproduced from [59].

[60]—to evolve under the same speed law, not just the zero curve. Some choices of speed functions, such as ones taken directly from the velocity field of a fluid, may over time produce shearing of the level curves and degrade the accuracy of the method. Additionally, some speed laws are only physically defined on the interface itself. A careful choice of *Extension Velocity* F_{ext} can build a speed function away from the interface that ensures the level set function remains a signed distance function for all time [61]. At finite difference node x, take $F_{ext} = F(x_{cp})$ where x_{cp} is the closest point on the interface with respect to x. This extension velocity is constant along straight lines radiating in the direction of the interface normal and ensures that $\nabla F_{ext} \cdot \nabla \phi = 0$ when ϕ is a signed distance function. An efficient, high-order accurate algorithm for determining the closest point on an implicit interface defined by the level set method is presented in [62]. This algorithm generates a polynomial representation of the level set function and applies Newton's method, initialized at a point within a cloud of seed points, to determine the closest point on the interface. For interface dynamics with large deformations, it is advantageous to periodically *Reinitialize* the level set function back to a signed distance function. A simple technique of reinitialization is to use the above-mentioned closest point method to determine the distance of the finite difference nodes to the interface. For more information on the level set method, narrow banding, extension velocities, and reinitialization see the book [37], and for recent advancements in the field of level set methods see [58][63].

2.2.1 Height function and finite difference methods

In typical flow leveling problems, there is no self-folding of paint layers nor situations in which the paints develop significant profile steepness. For this reason, the surfaces within the multi-layer coating flow problem $(\Gamma_{\text{sub}}, \Gamma_{ij}, \Gamma_e)$ are each represented by a *Height Function* $h : \mathbb{R}^{d-1} \times [0, T] \to \mathbb{R}$, from which the level set function ϕ is constructed. Using a height function simplifies the mechanics of interface evolution. A height function does not need to

be reinitialized, nor does it need the construction of extension velocities—evolving only under the speed defined at the height function itself. The trade-off is that height functions cannot capture more intricate interfacial phenomena such as paint layers folding or intersecting with each other. In the multi-layer coating flow problem, the height function describing the substrate is held fixed and those capturing paint-paint and paint-gas surfaces evolve under the advection equation

$$h_t + \boldsymbol{v} \cdot \nabla h - v\hat{e}_d = 0, \tag{2.32}$$

where $\boldsymbol{v}$ and v are the horizontal and vertical components respectively of the interface velocity vector, incorporating the motion of the fluid and the motion induced by mass transfer (2.19). Here $\hat{e}_d$ represents the vertical dimension, while along the horizontal dimensions, ∇h is computed by high-order ENO schemes [64]. The temporal evolution of the height functions is captured simply via forward Euler. We note that if the surfaces $(\Gamma_{\text{sub}}, \Gamma_{ij}, \Gamma_e)$ intersect, the simulation is halted; further comments on this aspect are discussed later.

Within the hybrid numerical framework, a level set function ϕ is constructed from the height functions and used to generate numerical quadrature schemes for the implicit mesh DG methods [65][66]. The zero isosurfaces of ϕ align with the location of the height functions and interfaces, and the level set function together with a phase indicator defines the multi-phase liquid domain Ω. The level set function is defined at finite difference nodes located within the DG cells of the background quad/octree, as illustrated in Figure 2.3(right). In this work, the DG cells and finite difference nodes are uniformly spaced, with each cell containing two finite difference nodes per dimension[3]. The level set function is used within the hybrid numerical framework for the multi-layer coating flow problem for the following two purposes:

(i) The creation of level set polynomials for the generation of the numerical quadrature rules for the implicitly-defined elements and surfaces of the implicit mesh DG methods [65][66]. To couple the finite difference methods to the implicit mesh DG formulation, a piecewise polynomial representation of ϕ is constructed; specifically, the values of the level set function are bi-linearly or tri-linearly interpolated onto the Gauss-Lobatto nodes of the background DG cells, from which a bi-quadratic or tri-quadratic polynomial on each cell is naturally defined (see section 3.2). This process is highlighted in Figure 2.3(right). The level set polynomials are then used in the algorithms of [65] to generate the quadrature schemes. We note that this specification of the level set polynomials ensures interface continuity between the DG cells and that its cell-wise nature is amenable to parallelization.

(ii) The computation of interfacial curvature for surface tension calculations. *Mean Curvature* κ of the level set function ϕ is given by

$$\kappa = \nabla \cdot \boldsymbol{n} = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right). \tag{2.33}$$

[3]We found this to strike a good balance between the resolution of the finite difference grid and the piecewise polynomial bi-quadratic/tri-quadratic DG methods.

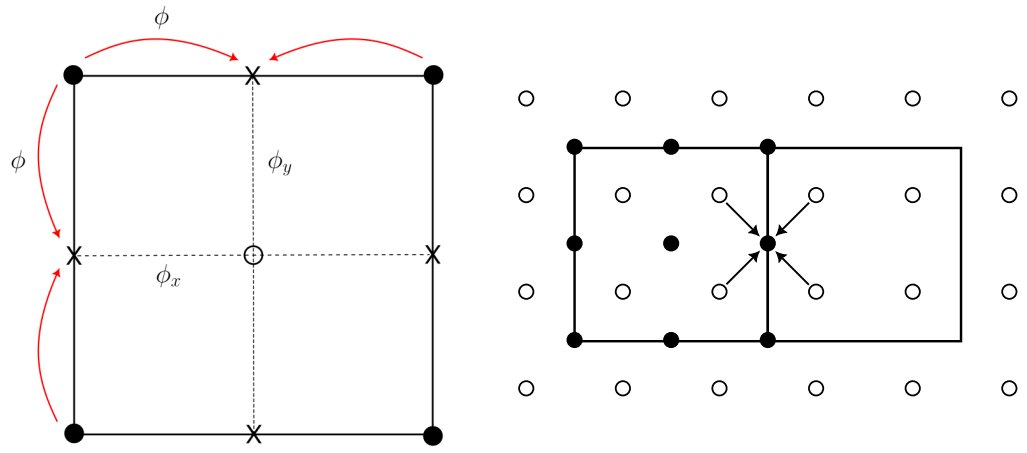


Figure 2.3: **Left:** The calculation of $\nabla\phi$ in 2D. Values of ϕ are averaged from the finite difference nodes ● onto fictitious cell edges $\times$ in each dimension. Then second-order finite differences compute the values of $\nabla\phi$ at cell center ○. **Right:** The finite difference nodes ○ located within the DG cells of the background quadtree. The level set function ϕ is interpolated from the finite difference nodes onto the cells' Gauss-Lobatto nodes ● and used in the generation of numerical quadrature schemes for implicitly-defined elements and surfaces [65].

To compute (2.33), consider a group of fictitious cells with the finite difference nodes as their vertices. Values of ϕ are averaged onto the center of cell edges and standard second-order centered finite differences are used to compute $\nabla\phi$ at cell centers, which is then normalized to compute the normal vector $\boldsymbol{n}$. This process is illustrated in Figure 2.3(left). The normal vector is then averaged back onto cell edge centers and second-order centered finite differences are used to determine the value of κ at the finite difference nodes. Values of curvature are then available via bi/tri-linear interpolation. This method for computing curvature is robust, second-order accurate, dimension-independent, and provides a suitable amount of averaging for smooth surface tension calculations.

Additionally, the finite difference methods discussed in section 4.3.2 are employed to compute surface gradients for Marangoni stresses. In summary, the finite difference methods within the hybrid numerical framework for the multi-layer coating flow problem provide a mechanism to robustly and accurately calculate the interface evolution as well as interfacial forces, whereas the implicit-mesh DG methods, discussed in the next chapter, provide a high-order accurate, sharp-interface method for computing fluid velocity fields, pressure, and solvent mass concentration profiles.

Chapter 3

Discontinuous Galerkin Methods

The discontinuous Galerkin finite element method (DG) is an increasingly popular numerical method for solving partial differential equations. Through the use of a piecewise polynomial solution space defined on a collection of discrete elements, DG methods produce high-order accurate PDE solutions while maintaining a compact stencil. Originally introduced by Reed and Hill for modeling neutron transport [67], DG methods have been applied to a range of problems, including hyperbolic conservation laws [68], parabolic convection-diffusion problems [69][70], and elliptic problems such as the Stokes equations [71][72]. A unified theory of discontinuous Galerkin methods that captures these various DG formulations is presented by Arnold et al. in [73]. DG is applicable to a variety of geometries and element shapes due to its weak formulation and has been applied on simplicial and rectangular meshes, as well as on meshes defined by polygons [74] and on those implicitly-defined by the level set method [75]. DG naturally extends to higher dimensions and its block sparse structures result in computations with high arithmetic intensity, which in combination with its natural locality makes DG highly amenable to parallelization [76]. For an introduction to discontinuous Galerkin methods see [77].

This chapter begins by introducing discontinuous Galerkin methods; then the implicit mesh DG methods used in our hybrid numerical framework for the multi-layer coating flow problem are discussed, including the notion of implicitly-defined meshes and their evolution, as well as the numerical quadrature schemes for implicitly-defined elements and surfaces; then new high-order accurate local discontinuous Galerkin methods (LDG) for Poisson problems with Robin boundary and jump conditions on implicitly-defined domains are presented; extensions of the recently developed operator coarsening multigrid methods [78] that solve the discrete linear systems are described; and lastly, convergence results and multigrid performance of the LDG methods for Poisson problems with Robin conditions are examined.

3.1 Introduction

Consider a d-dimensional domain $\Omega \subset \mathbb{R}^d$. The domain geometry is discretized into a collection of elements $\mathcal{E}$ on which the DG methods are applied. This work uses the implicit mesh DG framework of Saye [66][75], which defines meshes implicitly through a combination of the level set method with a structured background quadtree or octree. These implicitly-defined meshes contain both rectangular and interface-conforming curved elements. Owing to the use of a background quad/octree grid, it is natural to adopt a tensor-product piecewise polynomial space for the DG methods, including for the fluid velocity field $\boldsymbol{u}$, pressure p, and solvent mass concentration c_k within the multi-layer coating flow problem. Define $\mathcal{Q}_p(E)$ to be the space of d-dimensional tensor product polynomials of degree $p \geq 1$ on element $E \in \mathcal{E}$. Now define the discontinuous piecewise polynomial spaces V_h, V_h^d, and $V_h^{d\times d}$ for scalars, vectors, and rank-2 tensors respectively:

$$V_h = \{u : \Omega \to \mathbb{R} \mid u|_E \in \mathcal{Q}_p(E) \text{ for every } E \in \mathcal{E}\}$$
$$V_h^d = \{\boldsymbol{u} : \Omega \to \mathbb{R}^d \mid (\boldsymbol{u} \cdot \boldsymbol{e}_i)|_E \in \mathcal{Q}_p(E) \text{ for every } E \in \mathcal{E} \text{ and } 1 \leq i \leq d\}$$
$$V_h^{d\times d} = \{\boldsymbol{U} : \Omega \to \mathbb{R}^{d\times d} \mid (\boldsymbol{e}_i \cdot \boldsymbol{U} \cdot \boldsymbol{e_j})|_E \in \mathcal{Q}_p(E) \text{ for every } E \in \mathcal{E} \text{ and } 1 \leq i, j \leq d\},$$

where $\boldsymbol{e}_i$ denotes the standard basis vector in the direction of the i-th dimension. V_h, V_h^d, and $V_h^{d\times d}$ are square integrable L^2 spaces, with the natural inner product of V_h defined as $(u, v) = \int_\Omega uv$ for $u, v \in V_h$ and the L^2 norm defined as $||u||^2 = (u, u)$, with analogous definitions for V_h^d and $V_h^{d\times d}$. In our particular implementation of these methods, we are free to choose the polynomial degree; for the multi-layer coating flow problem, we found degree $p = 2$ (i.e., bi-quadratic and tri-quadratic polynomial spaces in 2D and 3D, respectively) provided a suitable balance between speed and high-order accuracy.

We now specify a basis for V_h, specifically a nodal basis, examining first the local element-wise definition of $\mathcal{Q}_p$, then expanding to all of Ω. For each element $E \in \mathcal{E}$, the space $\mathcal{Q}_p(E)$ has a dimension of $(p+1)^d$, meaning that $(p+1)^d$ nodal interpolation points/degrees of freedom are required to represent the solution on that element. Both rectangular and curved elements define their DG polynomials on the set of tensor product Gauss-Lobatto nodes located within the cells of the background quad/octree (see Figure 2.3(right)). Gauss-Lobatto nodes have the advantage of improved numerical conditioning when compared to evenly spaced nodes and have nodes located on the cell's boundary. For each element $E \in \mathcal{E}$ and for each element's local Gauss-Lobatto node i, $1 \leq i \leq (p+1)^d$, define the local basis function φ_i of space $\mathcal{Q}_p(E)$ as a p-th order d-dimensional Lagrange interpolating polynomial, such that $\varphi_i(x_j) = \delta_{ij}$ for local node x_j. Here δ_{ij} is the Kronecker delta function, meaning that the local basis function φ_i equals 1 on node i and 0 on all other nodes. For numerical conditioning reasons, the Lagrange interpolating polynomials are constructed from a linear combination of d-dimensional Legendre polynomials. For example, the basis functions in 1D are such that

$$\varphi_i(x) = \sum_{j=0}^{p} c_i^j P_j(x),$$

where P_j is the 1D Legendre polynomial of degree j. The coefficients c_i^j can be found by solving the following linear system:

$$\begin{pmatrix} P_0(x_0) & P_1(x_0) & \cdots & P_p(x_0) \\ P_0(x_1) & P_1(x_1) & \cdots & P_p(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ P_0(x_p) & P_1(x_p) & \cdots & P_p(x_p) \end{pmatrix} \begin{pmatrix} c_0^0 & c_1^0 & \cdots & c_p^0 \\ c_0^1 & c_1^1 & \cdots & c_p^1 \\ \vdots & \vdots & \ddots & \vdots \\ c_0^p & c_1^p & \cdots & c_p^p \end{pmatrix} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{pmatrix}.$$

So for $u \in V_h$ and element $E \in \mathcal{E}$, the restriction $u|_E \in \mathcal{Q}_p(E)$ can be expanded into its local basis functions

$$u|_E = \sum_{i=1}^{(p+1)^d} (u|_E)_i \varphi_i, \tag{3.1}$$

where here we abuse notation and view the polynomial function $u|_E$ as either (a) a polynomial in $\mathcal{Q}_p(E)$ and an L^2 function, or (b) a vector of coefficients relative to the local nodal basis. Now define the $(p+1)^d \times (p+1)^d$ *Elemental Mass Matrix* M_E relative to the nodal basis such that

$$M_{E,ij} = \int_E \varphi_i \varphi_j. \tag{3.2}$$

The elemental mass matrix is symmetric positive-definite and for rectangular elements is a scalar multiple of the tensor product of 1D mass matrices defined on a reference element, typically $[-1, 1]$ or $[0, 1]$. For curved elements, elemental mass matrices are computed via the numerical quadrature schemes of [65]. The L^2 *Projection* of function $f : E \to \mathbb{R}$ onto $\mathcal{Q}_p(E)$ is defined as

$$\mathbb{P}_E(f) = \underset{u \in \mathcal{Q}_p(E)}{\arg\min} \int_E (u - f)^2 = \underset{u \in \mathcal{Q}_p(E)}{\arg\min} ||u - f||^2,$$

i.e., as the best polynomial in $\mathcal{Q}_p(E)$ matching f in an L^2 sense. This minimization problem is convex and therefore a solution exists and is unique. The minimum occurs at the zero of the Fréchet derivative $D(||u - f||^2)v = (u - f, v) = 0$, for all test functions $v \in \mathcal{Q}_p(E)$. Therefore the L^2 projection onto element E is defined by $u \in \mathcal{Q}_p(E)$ such that

$$v^T M_E u = \int_E f v, \tag{3.3}$$

for all $v \in \mathcal{Q}_p(E)$. The right-hand side of (3.3) is computed by appropriate numerical quadrature rules and the elemental mass matrices are inverted via pre-computed Cholesky factorizations for curved elements and by an exact inverse for rectangular elements.

With the local element-wise definitions in place, the global bases, mass matrices, and L^2 projections are defined for V_h using the global piecewise polynomial basis across the entire domain Ω. Define the entire mesh *Mass Matrix* M as $M = \text{diag}(M_1, ..., M_{|\mathcal{E}|})$ for some

ordering of the elements in $\mathcal{E}$. The mass matrix is block-diagonal and symmetric positive-definite. Additionally, denote the global L^2 projection of $f : \Omega \to \mathbb{R}$ onto V_h as $\mathbb{P}_{V_h}(f)$. This projection is simply the local projection performed on each element of $\mathcal{E}$ and also extends to V_h^d and $V_h^{d\times d}$. Just as we abused notation in the local definition (3.1), we refer to the global DG piecewise polynomial function u as either (a) a function in V_h or (b) as a vector of coefficients relative to the nodal basis, the choice of which should be clear by the context.

3.2 Implicit mesh DG

The implicit mesh discontinuous Galerkin method was first introduced by Saye [75] in the context of high-order accurate fluid interface dynamics. The framework was developed further in the two-part paper [66][79]. In part one, Saye introduces the notion of implicitly-defined meshes, presents local discontinuous Galerkin methods for scalar and vector Poisson problems on multi-phase implicitly-defined domains, develops advection and projection operators for incompressible flow problems and geometric multigrid solvers for elliptic problems, and further introduces additional material essential for mesh evolution, with optimal high-order accuracy demonstrated in both 2D and 3D. In part two, the implicit mesh DG framework is applied to several problems, including unsteady flows in complex geometries, two-phase surface tension-driven flows, soap bubble dynamics, rigid body fluid-structure interaction problems, and free surface flows such as Plateau-Rayleigh instability. The accuracy of jump condition implementations is improved through the use of viscosity-upwinded weighted numerical fluxes [80] and, in [72], the framework is extended to solve multi-phase Stokes problems. The methods are made fast through the use of operator coarsening multigrid methods [78], which maintain PDE consistency throughout the multigrid hierarchy. The implicit mesh DG framework has recently been applied to the problems of wave propagation in elastic solids [81] and to rotary bell atomization [1]. The realization of fast high-order accurate solutions of sharply-captured evolving interface dynamics in a dimension-independent fashion makes the implicit mesh DG method an ideal framework for solving the multi-layer coating flow problem.

3.2.1 Implicitly-defined meshes

An implicitly-defined mesh is constructed from two objects: (i) a background quadtree or octree depending on whether the problem is in 2D or 3D respectively, and (ii) an implicit representation of the interfaces—the zero isocontour of a level set function in this work—which either defines the domain's boundary, the embedded surfaces within the domain, or both. The zero level set cuts through the cells of the background quad/octree, resulting in a collection of rectangular and cut cells for each phase of the domain, which are denoted as *Phase-cells*. Each phase-cell is then classified as empty, small, large, or entire by the volume fraction of that phase within the cell, with small and large classifications demarcated by a user-defined volume fraction threshold, taken to be 40% in this work. Empty phase-cells are discarded and small phase-cells are merged with neighboring large or entire phase-cells of the

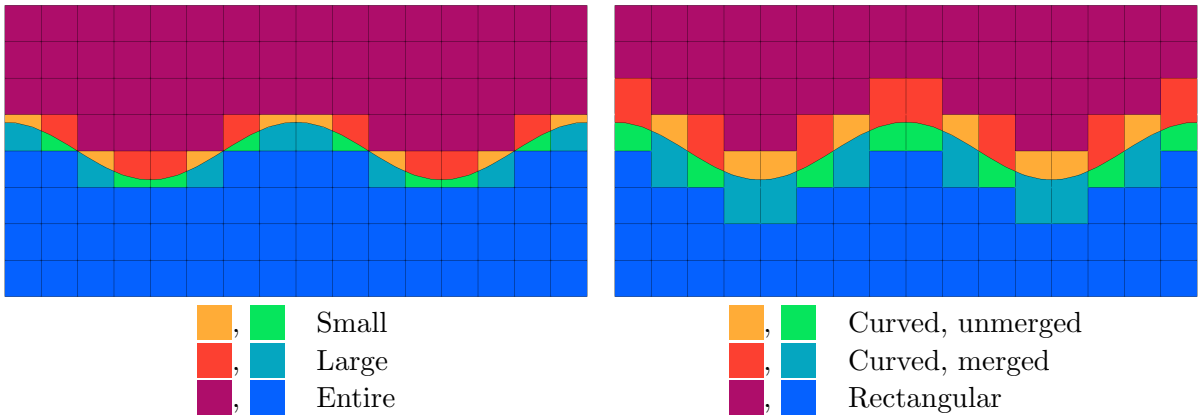


Figure 3.1: The construction of an implicit mesh for a two-phase (red and blue) domain in 2D with an embedded surface. **Left:** The zero isocontour of the level set function cuts through the cells of the background quadtree, resulting in phase-cells that are classified as empty, small, large, or entire. **Right:** The elements of the implicitly-defined mesh after cell merging. The mesh consists of mostly standard rectangular elements with a collection of curved interface-conforming elements, which may or may not be merged and extend outside of their cell.

same phase (known as the *Parent* phase-cell) to avoid the numerical conditioning issues and time step restrictions caused by arbitrarily-small cut cells. In [66], a general cell-merging algorithm is described, wherein a small phase-cell searches its neighboring phase-cells in the order of sharing a cell face, edge, and then vertex; the neighboring phase-cell with the largest volume fraction is then used for the cell merging procedure. In the multi-layer coating flow problem, we note the geometry allows for a simpler approach: small phase-cells are merged with the large or entire phase-cells directly above or below. Other cell-merging classifiers and procedures are also possible.

This process defines a collection of mesh elements that are mostly rectangular with a small number of interfacial curved elements that may or may not extend outside of their parent cell. In the context of DG, element identifiers, local basis functions, and elemental mass matrices of both element types are defined with respect to the parent cell and its tensor-product Gauss-Lobatto nodes. Curved interfacial elements are interface-conforming and the resulting mesh sharply captures the interfaces, allowing for the high-order accurate imposition of boundary and jump conditions and the capturing of thin solutal boundary layers present in evaporating Marangoni flows. Note that the interface is never explicitly constructed as a discretized mesh; instead, the geometry of interfacial elements and corresponding surfaces are determined solely by means of numerical quadrature in the weak formulation of the DG methods. The cell merging construction process is just a simple grouping of the quadrature rules for each phase-cell defining a curved element. An example of an implicit mesh and its cell merging procedure for a two-phase domain with an embedded surface is demonstrated in Figure 3.1.

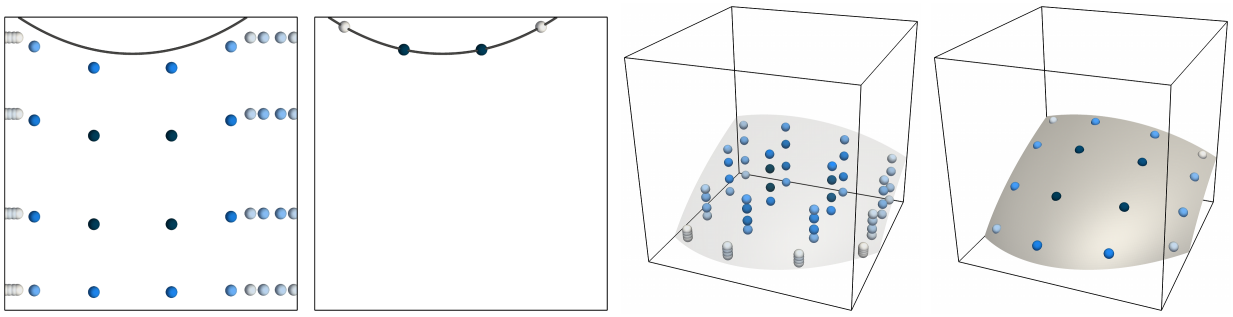


Figure 3.2: An example of the numerical quadrature rules for implicitly-defined volumes (far-left, center-right) and surfaces (center-left, far-right), for both 2D and 3D. Figures adapted from [65].

The elements then define a collection of faces on which the DG surface integrals are computed. The faces of the implicitly-defined meshes are classified as follows:

(i) *Intraphase* faces are shared between elements of the same phase, the collection of which is denoted as Γ_0. These faces are always flat and the normal vector is defined in the coordinate direction orthogonal to that face, taken to point from "left" to "right", i.e., $\boldsymbol{n} = \hat{x}$ for vertical faces in 2D.

(ii) *Interphase* faces are defined implicitly and lie between elements of differing phases. These faces are where interfacial jump conditions are applied. In the multi-layer coating flow problem, interphase faces are situated on Γ_{ij}, whose normal vector points in the direction defined by the level set function.

(iii) *Boundary* faces are situated on the boundary of the domain and are the faces on which boundary conditions are applied, e.g., the free evaporative surface Γ_e and the substrate Γ_{sub} in the multi-layer coating flow problem. The normal vectors for boundary faces are taken to be outwards pointing.

We note that in all numerical studies of the multi-layer coating flow problem, the interfaces remain smooth at all times and a unique normal vector is always well-defined. Now define the jump across a face as $[u] = u^- - u^+$, taken to align with the normal vector. Here u^- and u^+ represent either (a) the solution of a PDE restricted from the left and right phases respectively, or (b) the trace of the DG polynomials from the left and right elements, the choice of which should be clear by the context. Along boundary faces, define u^- to represent the PDE solution or trace of the DG polynomials from the interior element.

3.2.2 Numerical quadrature

The computation of volume and surface integrals arising from the variational forms of the DG methods requires appropriate numerical quadrature rules for the mesh elements and faces. Integrals along rectangular elements and faces use tensor-product Gauss-Legendre quadrature rules while the curved interfacial elements, implicit interfaces, and cell edges cut

by the zero level set use the high-order quadrature algorithms of [65]. An example of such schemes can be seen in Figure 3.2 and an open-source C++ implementation of this algorithm is available at [82].

3.2.3 Temporal evolution

We now briefly discuss the notion of temporal evolution within the implicit mesh DG framework—with the end goal of modeling time-dependent interface problems. At every time step $n = 0, 1, ...$, the level set function ϕ^n (defined in this work via the height functions) combined with the background quad/octree generates an implicitly-defined mesh as well as its elemental and surface quadrature rules. The DG polynomial space V_h^n (as well as the space of vectors and rank-2 tensors) and its corresponding nodal basis are defined specifically for the corresponding time step's implicit mesh. The polynomial space then generates the LDG operators specific to each mesh and interface configuration. As the level set function—the interfaces—evolves to the next time step ϕ^{n+1}, the previously defined implicit mesh no longer aligns with the interfaces, meaning that numerical quadrature rules, DG polynomial spaces, and LDG operators must be recreated for the new interface locations. State variables that were defined in the polynomial space of the previous mesh are then transferred onto the polynomial space of a new implicit mesh that captures the updated interface locations, i.e., a state variable in V_h^n is transferred to one appropriate in the new space V_h^{n+1}. In the general setting, this can be done via a general kind of L^2 projection. In fact, because the interface usually moves only a small fraction of the mesh spacing, it is possible to create time-evolving implicit meshes which, for the most part, use the same cell-merging decisions as prior time steps. Using this approach, it is often the case that a one-to-one correspondence exists between the elements of one mesh and the next, which simplifies the state transfer operation to a very simple injection procedure. The general state transference procedure is triggered when it is no longer possible to maintain the same cell-merging decisions and the one-to-one correspondence between meshes; for example, when a previously small phase-cell must now be classified as large (in this situation, we use a "fuzzy-threshold" on the small-large demarcation when determining the classification change; this is done to reduce the frequency of the general state transfer). For more details on this approach, and time-stepping implicit mesh DG methods in general, see [66].

3.3 Poisson problems with Robin boundary and jump conditions

A key driving force in the multi-layer coating flow problem is mass transfer at the free evaporative surface Γ_e and across the embedded paint-paint surfaces Γ_{ij}. As motivated in section 2.1, the motion of the dissolving solvents within the paints is described by the convection-diffusion equations (2.3); the species mass transfer through a surface is described, in general, by a Robin-style jump condition (2.27) and, in our free-surface evaporative model,

by a Robin boundary condition (2.28). Recall these equations of motion:

$$\begin{aligned}(c_k)_t + \nabla \cdot (c_k \boldsymbol{u}) &= \nabla \cdot (D_k \nabla c_k) \\ m[c_k] - [\rho D_k \nabla c_k \cdot \boldsymbol{n}] &= 0 \\ m c_k - \rho D_k \nabla c_k \cdot \boldsymbol{n} &= m_k.\end{aligned}$$

Note that the Robin conditions involve a linear combination of the solvent mass concentration and its diffusive flux. The mixed explicit-implicit time stepping method employed within the evaporative mass transfer system of the multi-layer coating flow problem leads to a heat operator[1] problem (4.1) with domain boundary conditions of Robin type. In turn, this requires the development of tailored local discontinuous Galerkin (LDG) methods [70][73] specifically targeting Robin conditions. In this section, we present the derivation of LDG methods for Poisson problems with Robin conditions, examining:

(i) Robin boundary conditions: Robin conditions along a domain boundary.

(ii) Robin jump conditions: Robin conditions across an embedded interface separating two volumetric regions.

The resulting LDG discretization has several favorable properties: for example, it is optimally high-order accurate, dimension independent, and applicable to a wide range of variable diffusion and Robin coefficients. In situations considering only Robin boundary conditions, the final linear system is symmetric positive-definite and, moreover, the LDG discretization is amenable to straightforward, fast, multigrid-preconditioned conjugate gradient solvers [78]. The inclusion of Robin jump conditions may break the symmetry of the Poisson system, however the choice of consistent numerical fluxes, derived in section 3.3.1, in the direction determined by viscosity-impedance upwinding (discussed in section 3.5.2), leads to LDG schemes that produce high-order accurate solutions to this problem.

Consider a two-phase, d-dimensional domain $\Omega = \Omega_i \bigcup \Omega_j$, consisting of phases i and j. Let Γ_R denote the section of $\partial\Omega$ on which Robin boundary conditions are applied, and let Γ_{ij}^R be the section of $\Omega_i \bigcap \Omega_j$ where Robin jump conditions are applied. Assume that both Γ_R and Γ_{ij}^R are sufficiently smooth and, for the sake of presentation, that neither is empty. We wish to find $u : \Omega \to \mathbb{R}$ such that

$$\begin{cases} -\nabla \cdot (\mu \nabla u) = f & \text{in } \Omega \\ [u] = g_{ij} & \text{on } \Gamma_{ij}^R \\ [\alpha u] + [\mu \nabla u \cdot \boldsymbol{n}] = r_{ij} & \text{on } \Gamma_{ij}^R \\ au + (b\mu \nabla u) \cdot \boldsymbol{n} = r & \text{on } \Gamma_R, \end{cases} \tag{3.4}$$

[1]The heat operator equation involves finding $u : \Omega \to \mathbb{R}$ such that $(\mathbb{I}/\delta - \nabla \cdot (D\nabla))u = f$, where $\mathbb{I}$ is the identity operator, δ is a constant (typically describing a discrete time step), D is the diffusion coefficient, and $f : \Omega \to \mathbb{R}$ is a source function, subject to suitable boundary conditions.

where $\mu, \alpha, a, b : \mathbb{R}^N \to \mathbb{R}^+$ are functions mapping onto the space of positive real numbers. Here μ takes the role of the diffusion coefficient and α, a, b are Robin coefficients. Assume f, g_{ij}, r_{ij}, and r are sufficiently smooth. Here $[u]$ denotes the jump in u across Γ_{ij}^R in the direction of the normal vector $\boldsymbol{n}$. To close the system, we specify a jump condition on u of the form $[u] = g_{ij}$. The LDG methods for (3.4) presented in section 3.3.2 consist of four steps [66]:

(i) Introduce the gradient $\boldsymbol{\eta} \in V_h^d$ such that $\boldsymbol{\eta} = \nabla u$ weakly via the strong-weak form.

(ii) Define $\boldsymbol{q} \in V_h^d$ as the L^2 projection of $\mu\boldsymbol{\eta}$.

(iii) Compute the divergence $w \in V_h$ such that $w = \nabla \cdot \boldsymbol{q}$ weakly via the weak-weak form.

(iv) Require that $-w$ equals the L^2 projection of f, while also incorporating penalty stabilization to enforce continuity, boundary, and jump conditions.

To approximate the solution of (3.4) on element $E \in \mathcal{E}$, we multiply the gradient ∇u and divergence $\nabla \cdot \boldsymbol{q}$ by a test function and integrate by parts to get the following weak equations:

$$\int_E \boldsymbol{\eta} \cdot \boldsymbol{\omega} = \int_E \nabla u \cdot \boldsymbol{\omega} + \int_{\partial E} (u^* - u)\boldsymbol{\omega} \cdot \boldsymbol{n}, \tag{3.5}$$

$$\int_E wv = -\int_E \boldsymbol{q} \cdot \nabla v + \int_{\partial E} v\boldsymbol{q}^* \cdot \boldsymbol{n}, \tag{3.6}$$

for all test functions $\boldsymbol{\omega}, v$ on element E. Here u^* and $\boldsymbol{q}^*$ are *Numerical Fluxes*, which arise since the values of u and $\boldsymbol{q}$ are discontinuous along element faces. The numerical fluxes may depend on the traces of the DG polynomials, and on boundary or jump data. Key to this work is the choice of PDE consistent numerical fluxes, which are described in the next section. This choice of numerical fluxes results in LDG schemes that solve (3.4) to high-order accuracy, on implicitly-defined domains, in both 2D and 3D, for a variety of diffusion and Robin coefficients.

3.3.1 Numerical flux motivation

We wish to motivate the choice of numerical fluxes used in equations (3.5) and (3.6), beginning with the Robin jump conditions. Without loss of generality, set $\mu = 1$ in (3.4); also let u^- and u^+ denote the restrictions of u from Ω_i and Ω_j respectively. Multiply the PDE by test function v and integrate by parts, then

$$\int_\Omega \Delta u v = -\int_\Omega \nabla u \cdot \nabla v + \int_{\Gamma_{ij}^R} (\nabla u^- \cdot \boldsymbol{n})v^- - \int_{\Gamma_{ij}^R} (\nabla u^+ \cdot \boldsymbol{n})v^+. \tag{3.7}$$

Add and subtract $\int_{\Gamma_{ij}^R} (\nabla u^+ \cdot \boldsymbol{n})v^-$ from (3.7) to get

$$\begin{aligned}\int_\Omega \Delta u v &= \ldots + \int_{\Gamma_{ij}^R} ((\nabla u^- \cdot \boldsymbol{n}) - (\nabla u^+ \cdot \boldsymbol{n}))v^- + \int_{\Gamma_{ij}^R} (\nabla u^+ \cdot \boldsymbol{n})v^- - (\nabla u^+ \cdot \boldsymbol{n})v^+ \\ &= \ldots + \int_{\Gamma_{ij}^R} [\nabla u \cdot \boldsymbol{n}]v^- + \int_{\Gamma_{ij}^R} (\nabla u^+ \cdot \boldsymbol{n})[v].\end{aligned} \tag{3.8}$$

Plugging the Robin jump condition $[\nabla u \cdot \boldsymbol{n}] = r_{ij} - [\alpha u]$ into (3.8) gives rise to a term of form:

$$\int_{\Gamma_{ij}^R} r_{ij} v^- - [\alpha u] v^- + \int_{\Gamma_{ij}^R} (\nabla u^+ \cdot \boldsymbol{n})[v]. \tag{3.9}$$

To remain consistent with (3.9), the numerical flux $\boldsymbol{q}^*$ along the Robin jump interface is taken to be

$$\begin{cases} \boldsymbol{q}_i^* & = \boldsymbol{q}^+ + r_{ij}\boldsymbol{n} - [\alpha u]\boldsymbol{n} \\ \boldsymbol{q}_j^* & = \boldsymbol{q}^+, \end{cases} \tag{3.10}$$

where $\boldsymbol{q}_i^*$ and $\boldsymbol{q}_j^*$ are the numerical fluxes for phases i and j respectively. Similarly, one could instead add and subtract $\int_{\Gamma_{ij}^R} (\nabla u^- \cdot \boldsymbol{n}) v^+$ from (3.7) to get a contribution of

$$\int_{\Gamma_{ij}^R} [\nabla u \cdot \boldsymbol{n}] v^+ + \int_{\Gamma_{ij}^R} (\nabla u^- \cdot \boldsymbol{n})[v]. \tag{3.11}$$

Plugging the Robin jump condition $[\nabla u \cdot \boldsymbol{n}] = r_{ij} - [\alpha u]$ into (3.11) gives

$$\int_{\Gamma_{ij}^R} r_{ij} v^+ - [\alpha u] v^+ + \int_{\Gamma_{ij}^R} (\nabla u^- \cdot \boldsymbol{n})[v]. \tag{3.12}$$

To remain consistent with (3.12), the numerical flux $\boldsymbol{q}^*$ is taken to be

$$\begin{cases} \boldsymbol{q}_i^* & = \boldsymbol{q}^- \\ \boldsymbol{q}_j^* & = \boldsymbol{q}^- - r_{ij}\boldsymbol{n} + [\alpha u]\boldsymbol{n}. \end{cases} \tag{3.13}$$

Note that (3.10) and (3.13) are multivalued, one-sided fluxes—in the opposite directions. Here, when an element has to "reach across" the interface, the Robin jump condition contributes to the numerical flux. It may be advantageous to vary the flux direction, so we choose a linear combination of (3.10) and (3.13) as our Robin jump numerical fluxes for $\boldsymbol{q}$

$$\begin{cases} \boldsymbol{q}_i^* & = \lambda(\boldsymbol{q}^+ + r_{ij}\boldsymbol{n} - [\alpha u]\boldsymbol{n}) + (1-\lambda)\boldsymbol{q}^- \\ \boldsymbol{q}_j^* & = \lambda\boldsymbol{q}^+ + (1-\lambda)(\boldsymbol{q}^- - r_{ij}\boldsymbol{n} + [\alpha u]\boldsymbol{n}), \end{cases} \tag{3.14}$$

where $\lambda \in [0,1]$ is a directional weighting parameter. In LDG, the numerical flux for u is taken in the opposite direction of that for $\boldsymbol{q}$. A similar calculation on the weak definition of ∇u gives the following numerical fluxes u^* along Γ_{ij}^R

$$\begin{cases} u_i^* & = \lambda u^- + (1-\lambda)(u^+ + g_{ij}) \\ u_j^* & = \lambda(u^- - g_{ij}) + (1-\lambda)u^+. \end{cases} \tag{3.15}$$

The choice of PDE consistent numerical fluxes for the Robin boundary condition is straightforward. Take the numerical fluxes to be a simple modification of the LDG fluxes for the finite element method's "natural" Neumann boundary condition, setting: $u^* = u^-$ and $\boldsymbol{q}^* = \frac{1}{b}(r - au^-)\boldsymbol{n}$. The numerical flux for u is taken from the trace of the interior element DG polynomials and the numerical flux of $\boldsymbol{q}$ incorporates the boundary data.

3.3.2 Local discontinuous Galerkin methods

With the choice of numerical fluxes determined, we now present the LDG method for (3.4) in the four steps previously stated:

(i) Define auxiliary variable $\boldsymbol{\eta} \in V_h^d$ such that $\boldsymbol{\eta} = \nabla u$ weakly for element $E \in \mathcal{E}$ via the strong-weak form (3.5)

$$\int_E \boldsymbol{\eta} \cdot \boldsymbol{\omega} = \int_E \nabla u \cdot \boldsymbol{\omega} + \int_{\partial E} (u^* - u)\boldsymbol{\omega} \cdot \boldsymbol{n},$$

for all test functions $\boldsymbol{\omega}$ on element E. Summing over all elements gives

$$\begin{aligned} \int_\Omega \boldsymbol{\eta} \cdot \boldsymbol{\omega} = \sum_E \int_E \nabla u \cdot \boldsymbol{\omega} + \int_{\Gamma_0} (u^* - u^-)\boldsymbol{\omega}^- \cdot \boldsymbol{n} - \int_{\Gamma_0} (u^* - u^+)\boldsymbol{\omega}^+ \cdot \boldsymbol{n} \\ + \int_{\Gamma_{ij}^R} (u_i^* - u^-)\boldsymbol{\omega}^- \cdot \boldsymbol{n} - \int_{\Gamma_{ij}^R} (u_j^* - u^+)\boldsymbol{\omega}^+ \cdot \boldsymbol{n} + \int_{\Gamma_R} (u^* - u^-)\boldsymbol{\omega}^- \cdot \boldsymbol{n}. \end{aligned} \tag{3.16}$$

Define the interfacial numerical flux u^* as motivated in section (3.3.1). In LDG methods, the numerical fluxes for the collection of intraphase faces Γ_0 are often one-sided [70][73]. For simplicity of presentation, the numerical flux for the collection of intraphase faces Γ_0 is set to u^- (from the left), therefore

$$u^* = \begin{cases} u^- & \text{on } \Gamma_0 \\ \lambda u^- + (1-\lambda)(u^+ + g_{ij}) & \text{for phase } i \text{ on } \Gamma_{ij}^R \\ \lambda(u^- - g_{ij}) + (1-\lambda)u^+ & \text{for phase } j \text{ on } \Gamma_{ij}^R \\ u^- & \text{on } \Gamma_R. \end{cases} \tag{3.17}$$

Plugging this numerical flux into (3.16) eliminates the Robin boundary contribution to the gradient and gives

$$\begin{aligned} \int_\Omega \boldsymbol{\eta} \cdot \boldsymbol{\omega} = \sum_E \int_E \nabla u \cdot \boldsymbol{\omega} + \int_{\Gamma_0} (u^+ - u^-)\boldsymbol{\omega}^+ \cdot \boldsymbol{n} \\ + \int_{\Gamma_{ij}^R} (1-\lambda)(u^+ - u^-)\boldsymbol{\omega}^- \cdot \boldsymbol{n} + \lambda(u^+ - u^-)\boldsymbol{\omega}^+ \cdot \boldsymbol{n} \\ + \int_{\Gamma_{ij}^R} (1-\lambda) g_{ij}\boldsymbol{\omega}^- \cdot \boldsymbol{n} + \lambda g_{ij}\boldsymbol{\omega}^+ \cdot \boldsymbol{n}. \end{aligned}$$

- Define the *Broken Gradient* operator $\nabla_h : V_h \to V_h^d$ and the *Lifting* operator $L : V_h \to V_h^d$ such that, respectively,

$$\int_\Omega \nabla_h u \cdot \boldsymbol{\omega} = \sum_E \int_E \nabla u \cdot \boldsymbol{\omega}, \tag{3.18}$$

$$\int_\Omega Lu\cdot\boldsymbol{\omega} = \int_{\Gamma_0}(u^+-u^-)\boldsymbol{\omega}^+\cdot\boldsymbol{n}+\int_{\Gamma_{ij}^R}(1-\lambda)(u^+-u^-)\boldsymbol{\omega}^-\cdot\boldsymbol{n}+\lambda(u^+-u^-)\boldsymbol{\omega}^+\cdot\boldsymbol{n}, \quad (3.19)$$

for all $\boldsymbol{\omega}\in V_h^d$. The broken gradient operator defines the gradient of polynomial $u\in V_h$ on an element-wise basis and the lifting operator incorporates the jumps in u across faces in each dimension, effectively taking the surface integrals and "lifting" them into the domain. Now, define the *Gradient* operator $G: V_h\to V_h^d$ as $G=\nabla_h+L$.

- Take $J_g\in V_h^d$ such that

$$\int_\Omega J_g(g_{ij})\cdot\boldsymbol{\omega} = \int_{\Gamma_{ij}^R}(1-\lambda)g_{ij}\boldsymbol{\omega}^-\cdot\boldsymbol{n}+\lambda g_{ij}\boldsymbol{\omega}^+\cdot\boldsymbol{n}, \quad (3.20)$$

for all $\boldsymbol{\omega}\in V_h^d$.

Together, the weak gradient is defined by $\boldsymbol{\eta}=Gu+J_g$.

(ii) Now define $\boldsymbol{q}\in V_h^d$ as the L^2 projection of $\mu\boldsymbol{\eta}$. In terms of DG operators, this is equivalent to $\boldsymbol{q}=M^{-1}M_\mu\boldsymbol{\eta}$, where M is the mass matrix and M_μ is the μ-weighted mass matrix such that $v^TM_\mu u=\int_\Omega v\mu u$ for all $u,v\in V_h$. So the k-th component of $\boldsymbol{q}$ is equal to

$$q_k = M^{-1}M_\mu(G_ku+J_{g,k}). \quad (3.21)$$

(iii) Next we seek to define the divergence of $\boldsymbol{q}$. For element $E\in\mathcal{E}$, let $w\in V_h$ be such that $w=\nabla\cdot\boldsymbol{q}$ weakly via the weak-weak form (3.6)

$$\int_E wv = -\int_E \boldsymbol{q}\cdot\nabla v+\int_{\partial E} v\boldsymbol{q}^*\cdot\boldsymbol{n},$$

for all test functions v on element E. Summing over all elements results in

$$\int_\Omega wv = -\sum_E\int_E\boldsymbol{q}\cdot\nabla v+\int_{\Gamma_0}(v^--v^+)\boldsymbol{q}^*\cdot\boldsymbol{n}+\int_{\Gamma_{ij}^R}v^-\boldsymbol{q}_i^*\cdot\boldsymbol{n}-v^+\boldsymbol{q}_j^*\cdot\boldsymbol{n}+\int_{\Gamma_R}v^-\boldsymbol{q}^*\cdot\boldsymbol{n}. \quad (3.22)$$

Define the numerical flux $\boldsymbol{q}^*$ as motivated in section 3.3.1, in the opposite direction as that for u, incorporating Robin jump and boundary data:

$$\boldsymbol{q}^* = \begin{cases} \boldsymbol{q}^+ & \text{on } \Gamma_0 \\ \lambda(\boldsymbol{q}^+ + r_{ij}\boldsymbol{n}-[\alpha u]\boldsymbol{n})+(1-\lambda)\boldsymbol{q}^- & \text{for phase } i \text{ on } \Gamma_{ij}^R \\ \lambda\boldsymbol{q}^+ + (1-\lambda)(\boldsymbol{q}^- - r_{ij}\boldsymbol{n}+[\alpha u]\boldsymbol{n}) & \text{for phase } j \text{ on } \Gamma_{ij}^R \\ \frac{1}{b}(r-au^-)\boldsymbol{n} & \text{on } \Gamma_R. \end{cases} \quad (3.23)$$

Plugging these numerical fluxes into (3.22) gives

$$\begin{aligned}\int_\Omega wv = &-\sum_E \int_E \boldsymbol{q}\cdot\nabla v - \int_{\Gamma_0}(v^+ - v^-)\boldsymbol{q}^+\cdot\boldsymbol{n} \\ &- \int_{\Gamma_{ij}^R}(1-\lambda)(v^+ - v^-)\boldsymbol{q}^-\cdot\boldsymbol{n} + \lambda(v^+ - v^-)\boldsymbol{q}^+\cdot\boldsymbol{n} \\ &+ \int_{\Gamma_{ij}^R}(1-\lambda)v^+ r_{ij} + \lambda v^- r_{ij} + \int_{\Gamma_R} v^-\frac{1}{b}r \\ &- \int_{\Gamma_{ij}^R}(1-\lambda)v^+[\alpha u] + \lambda v^-[\alpha u] - \int_{\Gamma_R} v^-\frac{1}{b}au^-.\end{aligned} \tag{3.24}$$

We now break (3.24) into three components. Notice that the first two lines are equivalent to $-(\nabla_h v, \boldsymbol{q}) - (Lv, \boldsymbol{q}) = -(Gv, \boldsymbol{q})$, where $(\cdot,\cdot)$ is the standard inner product. This component of the divergence operator is equivalent to the negative adjoint of the gradient operator. Secondly, define $J_R \in V_h$ such that

$$\int_\Omega J_R(r, r_{ij})v = \int_{\Gamma_{ij}^R}(1-\lambda)v^+ r_{ij} + \lambda v^- r_{ij} + \int_{\Gamma_R} v^-\frac{1}{b}r, \tag{3.25}$$

for all $v \in V_h$. Now define the block-sparse matrix $A_R(u)$ such that

$$v^T A_R(u)u = \int_{\Gamma_{ij}^R}(1-\lambda)v^+[\alpha u] + \lambda v^-[\alpha u] + \int_{\Gamma_R} v^-\frac{1}{b}au^-, \tag{3.26}$$

for all $v \in V_h$. Note that the Robin jump terms in A_R ($v^+[\alpha u]$ and $v^-[\alpha u]$) will break symmetry of the discrete Laplacian when the values of α differ across phases. This is consistent with the underlying PDE and jump conditions; in any case, the diagonal blocks of the discrete Laplacian will remain symmetric. Note the Robin boundary component of A_R is akin to penalization methods for weakly imposing Dirichlet boundary conditions in finite element methods [83][84]. Combining these terms gives the following weak definition of $\nabla\cdot\boldsymbol{q}$ incorporating the Robin boundary and jump data

$$w = -\sum_k M^{-1}G_k^T M q_k - M^{-1}A_R(u) + J_R. \tag{3.27}$$

Plugging in the definition of $\boldsymbol{q}$ (3.21) gives

$$w = -\sum_k M^{-1}G_k^T M_\mu(G_k u + J_{g,k}) - M^{-1}A_R(u) + J_R. \tag{3.28}$$

(iv) Additionally, LDG methods employ penalty stabilization to ensure the wellposedness of the discrete problem. For the Robin problem, penalization is added to weakly impose

continuity between intraphase elements and to impose the Robin jump condition in u. The penalty is of the form

$$\int_{\Gamma_0} \vartheta_0 [u][v] + \int_{\Gamma_{ij}^R} \vartheta_{ij}([u][v] - g[v]),$$

where ϑ_0 and ϑ_{ij} are positive penalty parameters. Define the block-sparse penalty matrix E and polynomial $a_R \in V_h$ such that

$$v^T E u = \int_{\Gamma_0} \vartheta_0 [u][v] + \int_{\Gamma_{ij}^R} \vartheta_{ij} [u][v], \tag{3.29}$$

$$\int_{\Omega} a_R v = \int_{\Gamma_{ij}^R} \vartheta_{ij} g[v], \tag{3.30}$$

for all $v \in V_h$. In general for Poisson problems, the penalty parameters should vary proportionally to the local diffusion coefficient μ and in inverse proportion to the mesh spacing h. Additionally, the penalty parameters are scaled with respect to polynomial degree p. For the tests in section 3.5, the penalty parameters are taken as $\vartheta_0 = 0.5\mu p/h$ and $\vartheta_{ij} = 8\min(\mu^-, \mu^+)p/h$, where μ^- and μ^+ are the local diffusion coefficients for the left and right elements respectively. In the simulations of multi-layer coating flows presented in Chapter 5, a higher degree of penalty stabilization is employed within the mass transfer system, with $\vartheta_0 = \beta\mu p/h$ and β ranging from 15-30.

Summary

The Poisson problem with Robin conditions (3.4) is solved via LDG by finding $u \in V_h$ such that the following block-sparse matrix equation holds:

$$\left(\sum_k G_k^T M_\mu G_k + A_R(u) + E \right) u = M \mathbb{P}_{V_h}(f) - \sum_k G_k^T M_\mu J_{g,k} + M J_R + M a_R, \tag{3.31}$$

where $\mathbb{P}_{V_h}(f)$ is the L^2 projection of f onto V_h. Note that both sides of the equation have been multiplied by the mass matrix M. For problems concerning only Robin boundary conditions, this results in a symmetric positive-definite linear system when the Robin coefficients are positive and the linear system (3.31) is solved by the fast operator coarsening multigrid-preconditioned conjugate gradient methods of [78], discussed next. The LDG methods developed in this section are demonstrated to be optimally high-order accurate in section 3.5. Rapid multigrid performance is also demonstrated for the Robin boundary problem in which the iteration count remains bounded as the background mesh spacing tends towards zero.

3.4 Operator coarsening multigrid

To solve the linear system (3.31), we extend the recently developed operator coarsening geometric multigrid methods of [78] to Poisson problems with Robin boundary conditions.

We note that, as discussed in section 3.3.2, the inclusion of Robin jump conditions may break the symmetry of the continuum and discrete Laplacian. Therefore, in section 3.5, the linear system (3.31) is solved via multigrid methods for symmetric Robin boundary problems and via a direct method from the parallel ScaLAPACKE library [85] for asymmetric problems involving Robin jump conditions.

Geometric multigrid methods are powerful solvers that act by reducing high-frequency eigenmodes in the error along a hierarchy of successively coarsened meshes. The operator coarsening multigrid paradigm maintains PDE consistency throughout each level of the hierarchy and sharply preserves the interfaces, while also avoiding the explicit construction of coarse meshes, making it an ideal solver for our problem. To illustrate a basic multigrid algorithm, consider a two-level mesh hierarchy consisting of a fine and coarse mesh. Given an initial fine-mesh approximate solution, geometric multigrid methods then perform the following:

1. First a relaxation method, or *Smoother*, is applied to the fine-mesh approximation. Smoothers quickly damp out high-frequency eigenmodes in the error but act slowly on low-frequency errors. Typically, the smoother is a stationary iterative method, such as the Jacobi or Gauss-Seidel methods.
2. The fine-mesh approximation is then transferred onto the coarse mesh. Here, the low-frequency errors appear higher-frequency due to the lower resolution of the coarse mesh. The smoother is then applied on the coarse mesh to damp out these modes.
3. The coarse-mesh correction is transferred back onto the fine mesh and the approximate solution updated.

This loose description of a multigrid *V-Cycle* illustrates that our geometric multigrid method requires the following three components: (a) a mesh hierarchy, (b) mesh-to-mesh transfer operators, and (c) a smoother. For a review of multigrid methods, see [86].

(a) In the implicit mesh DG framework, a mesh hierarchy is naturally defined by the structure of the background quad/octree. Fine-mesh elements are coarsened up the tree hierarchy onto their tree nodes' parents, creating a collection of coarse-mesh elements on which coarsened DG polynomials and LDG operators are defined. The interfaces are sharply preserved throughout the hierarchy and the coarse meshes are not explicitly constructed, with movement between meshes handled solely by the interpolation and restriction operators.

(b) To transfer quantities along the mesh hierarchy, we define the following *Interpolation* and *Restriction* operators. First, let V_f and V_c represent the fine- and coarse-mesh piecewise polynomial spaces respectively. The interpolation operator $I_c^f : V_c \to V_f$ defines a mapping from the coarse mesh onto the fine mesh via injection, i.e., a polynomial defined in V_c has its value preserved on V_f, with $I_c^f u_c = u_f$ for $u_c \in V_c$ and $u_f \in V_f$. We note that this choice of interpolation operator preserves constants. The restriction operator $R_f^c : V_f \to V_c$ is defined

as the adjoint of the interpolation operator, with

$$(R_f^c u_f, u_c)_{V_c} = (u_f, I_c^f u_c)_{V_f},$$
$$R_f^c = M_c^{-1}(I_c^f)^T M_f,$$

where M_f and M_c are the fine- and coarse-mesh mass matrices respectively, with $M_c = (I_c^f)^T M_f I_c^f$. The restriction operator may also be interpreted as the L^2 projection of the fine-mesh polynomials onto their coarse-mesh parents. Using the interpolation and restriction operators, we now compute the LDG operators on each level of the mesh hierarchy. For fine-mesh operator $A : V_f \to V_f$, define a *Coarsening Operator* $\mathcal{C}(A) : V_c \to V_c$ such that

$$(v_c, \mathcal{C}(A)u_c)_{V_c} = (I_c^f v_c, AI_c^f u_c)_{V_f},$$
$$\mathcal{C}(A) = M_c^{-1}(I_c^f)^T M_f A I_c^f = R_f^c A I_c^f, \tag{3.32}$$

for all $v_c, u_c \in V_c$. This is often referred to as "RAT" form and the coarsened operators are computed with simple block-sparse linear algebra. In [78], it was demonstrated that coarsening the discrete Laplacian in its primal form (its entirety) leads to poor multigrid performance. Instead, to maintain consistency with the underlying PDE at each level of the multigrid hierarchy, the operator coarsening paradigm states that LDG operators should be coarsened individually, then recombined to form the coarse-grid discrete Laplacian. The coarse-mesh LDG operators for the Poisson problem with Robin boundary conditions are therefore:

$$\begin{aligned}
M_c &= (I_c^f)^T M_f I_c^f \\
M_{\mu,c} &= (I_c^f)^T M_{\mu,f} I_c^f \\
G_c &= M_c^{-1}(I_c^f)^T M_f G_f I_c^f \\
E_c &= \frac{1}{2}(I_c^f)^T E_f I_c^f \\
A_{R,c} &= (I_c^f)^T A_{R,f} I_c^f \\
\Delta_c &= M_c^{-1}\left(\sum_k G_{c,k}^T M_{\mu,c} G_{c,k} + A_{R,c} + E_c\right).
\end{aligned}$$

Note that a factor of a half has been added to the coarsened penalty operator E_c in order to maintain the penalty's $1/h$ behavior throughout the multigrid hierarchy [80].

(c) The final key component of our multigrid method is the smoother, which is used to damp high-frequency errors. For our problem, we use a simple damped block Gauss-Seidel (block SOR) algorithm as our smoother. For all tests in section 3.5, $\nu = 3$ pre- and post-smoothing steps are applied for each V-cycle. The SOR damping parameter is set to $\omega = 0.9$.

The following is the algorithm for a single multigrid V-cycle: $V(A, \tilde{x}, b, l)$
Goal: Solve $Ax = b$, starting with an initial guess $\tilde{x}$

1. If l = bottom level, solve $Ax = b$ directly. Return x.
2. Smooth $A\tilde{x} = b$ via ν block Gauss-Seidel iterations.
3. Restrict the residual onto the next coarsest mesh: $r_c = (I_c^f)^T(b - A\tilde{x})$.
4. Recursively run the V-cycle on the residual at the next level, with initial guess zero: $x_c = V(A_c, 0, r_c, l + 1)$.
5. Interpolate the coarse-grid correction back onto the fine mesh and correct the solution: $\tilde{x} = \tilde{x} + I_c^f x_c$.
6. Smooth $A\tilde{x} = b$ via ν reverse-order block Gauss-Seidel iterations.
7. Return $\tilde{x}$.

To further accelerate performance, the multigrid algorithms are used as preconditioners for Krylov subspace methods. Krylov subspace methods work well to weakly impose the boundary and jump conditions, while multigrid solves the interior elliptic problem. This results in a fast, robust solver capable of solving Poisson problems with Robin boundary conditions in 6-12 iterations. The linear system (3.31) generated by the LDG methods for the Robin boundary problem is symmetric positive-definite and, for this problem, a single V-cycle is applied as a preconditioner for the conjugate gradient algorithm. Details on preconditioned conjugate gradient methods can be found in [87].

3.5 Convergence and multigrid tests

In this section, we test the order of accuracy of the LDG methods of section 3.3.2 for the Poisson problem with Robin boundary and jump conditions separately, in both 2D and 3D, as well as examine the performance of the multigrid algorithm developed in section 3.4 for the Robin boundary conditions. In particular, we demonstrate both high-order accuracy as well as rapid bounded multigrid performance for the challenging case of variable diffusion and Robin coefficients spanning several orders of magnitude on a curved implicitly-defined domain. Recall the Poisson problem with Robin conditions (3.4) is given by

$$\begin{cases} -\nabla \cdot (\mu \nabla u) = f & \text{in } \Omega \\ [u] = g_{ij} & \text{on } \Gamma_{ij}^R \\ [\alpha u] + [\mu \nabla u \cdot \boldsymbol{n}] = r_{ij} & \text{on } \Gamma_{ij}^R \\ au + (b\mu \nabla u) \cdot \boldsymbol{n} = r & \text{on } \Gamma_R. \end{cases}$$

We test convergence by comparing against an exact offset sinusoid solution, setting $u : \Omega \to \mathbb{R}$ to

$$u(x) = \prod_{i=1}^{d} \sin 2\pi(x_i - 0.05(\chi - 1)), \tag{3.33}$$

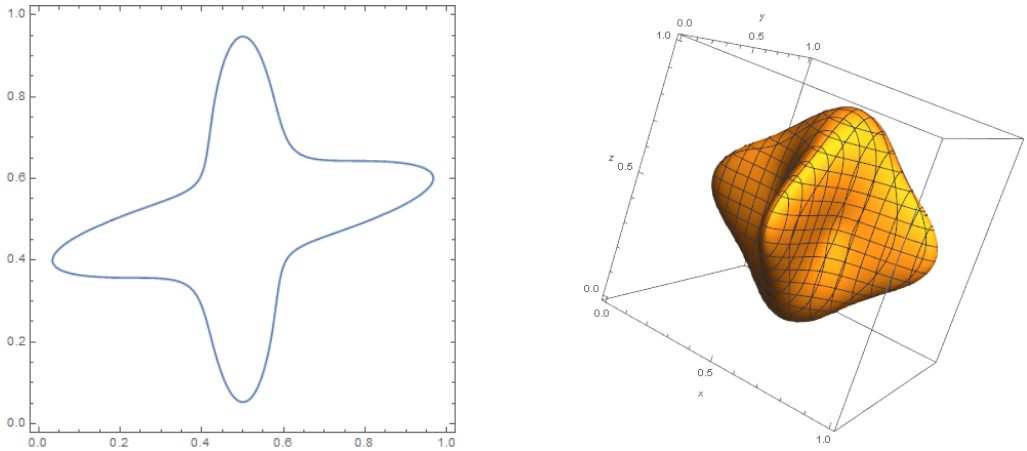


Figure 3.3: The implicitly-defined amoeba domain in 2D and 3D.

where χ represents the phase containing x. The exact solution is used to generate the source data f, Robin boundary data r, and interfacial Robin jump data g_{ij}, r_{ij}. Multigrid efficiency is assessed by using the average residual reduction factor per iteration of the multigrid-preconditioned conjugate gradient algorithm, which is defined by

$$\rho = \exp\left(\frac{1}{N}\log\left(\frac{||VAx_N - Vb||_2}{||VAx_0 - Vb||_2}\right)\right), \tag{3.34}$$

where N is the number of iterations needed to reduce the residual by a factor of 10^{10} from its initial value. Here V represents the multigrid V-cycle preconditioner while A represents the discrete Laplacian. All multigrid tests use a homogeneous $b = 0$ and a random initial guess x_0, thereby assessing performance across the full spectrum of eigenmodes. We test our formulation within an irregular "amoeba" domain embedded within a $[0,1]^d$ box, defined by the zero level set of

$$\phi(x,y,(z)) = \begin{cases} r^2 - 0.1 - (y_c^4 + 10x_c^3y_c - 20x_c^2y_c^2)/(2r^2) & \text{in 2D} \\ r^2 - 0.1 - (y_c^4 + 10x_c^3y_c - 20x_c^2y_c^2 + z_c^4)/(2r^2) & \text{in 3D} \end{cases},$$

where $(x_c, y_c, z_c) = (x, y, z) - 0.5$ and $r^2 = x_c^2 + y_c^2(+z_c^2)$. The amoeba is shown in Figure 3.3. For Poisson problems with Robin boundary conditions, the interior of the amoeba defines domain Ω and the interface/zero level set defines Γ_R. For Poisson problems with Robin jump conditions, take Ω_i to be the interior of the amoeba and Ω_j the exterior, with $\Omega = \Omega_i \bigcup \Omega_j$. The Robin jump conditions are applied on $\Gamma_{ij}^R = \Omega_i \bigcap \Omega_j$, and Poisson problems with Robin jump conditions are closed by applying Dirichlet conditions along the domain boundary.

3.5.1 Poisson problems with Robin boundary conditions

Test 1: Constant diffusion and Robin coefficients
We begin with perhaps the simplest nontrivial Robin boundary condition, setting unit diffusion coefficient $\mu = 1$, with equal unit weighting between the Robin coefficients, $a = b = 1$. Figure 3.4 presents the computed L^∞ errors against the exact solution along with the multigrid convergence rate ρ, in both 2D and 3D. For each polynomial order p, optimal $p+1$ order accuracy is achieved, as indicated by the fitted lines. Good multigrid performance is demonstrated in which the iteration count remains bounded as background mesh spacing h tends towards zero. The multigrid algorithm has a convergence rate of $\rho \leq 0.15$ in all cases, indicating 6-12 multigrid-preconditioned conjugate gradient iterations are needed to reduce the residual by a factor of 10^{10}.

Test 2: Variable diffusion and Robin coefficients spanning several orders of magnitude
Next, we examine a challenging case with variable coefficients, where the diffusion coefficient varies by four orders of magnitude and the Robin coefficients by eight orders of magnitude throughout the domain, setting

in 2D:	in 3D:
$\mu = 10^{2\sin(2\pi(x-0.1))\sin(2\pi(y+0.1))}$	$\mu = 10^{2\sin(2\pi(x-0.1))\sin(2\pi(y+0.1))\sin(2\pi(z-0.1))}$
$a = 10^{-4+8\sin(\pi x/2)\sin(\pi y/2)}$	$a = 10^{-4+8\sin(\pi x/2)\sin(\pi y/2)\sin(\pi(z+0.5)/2)}$
$b = 10^{4-8\sin(\pi x/2)\sin(\pi y/2)}$	$b = 10^{4-8\sin(\pi x/2)\sin(\pi y/2)\sin(\pi(z+0.5)/2)}$,

which gives a diffusion coefficient ranging from 10^{-2} to 10^2 and a spectrum of Robin coefficient ratios a/b ranging from 10^{-8} to 2×10^5 for both 2D and 3D. Figure 3.5 illustrates that the LDG method produces high-order accurate solutions and the multigrid algorithm performs well even in this challenging setting, achieving optimal $p+1$ order accuracy and good bounded multigrid performance with $\rho \leq 0.15$ for each polynomial order.

In the Limits
The Robin boundary condition, $au + (b\mu\nabla u)\cdot\boldsymbol{n} = r$ on Γ_R, can be viewed as a weighted combination of Dirichlet and Neumann boundary conditions. As $a \to 0$ with b nonzero, the Robin boundary condition approaches a pure Neumann condition, which for Poisson problems requires appropriate treatment of the kernel, being in that case the span of globally-constant functions. Note that this trivial kernel is not present within the heat operator (4.1) for the mass transfer system and the limit $a \to 0$, which represents the limit of zero evaporation in the multi-layer coating flow problem, poses no added difficulty to our hybrid numerical framework. In the limit $a \to \infty$ or $b \to 0$, the Robin boundary condition approaches a pure Dirichlet condition, whose LDG implementation requires additional penalty stabilization to ensure a well-conditioned discrete Laplacian. It is possible to incorporate this limit and its requirements into our LDG formulation—for example, one could adapt the Nitsche's finite element method of [88] for general Robin boundary conditions to our problem—but this limit is not relevant to the multi-layer coating flow problem and therefore not explored here.

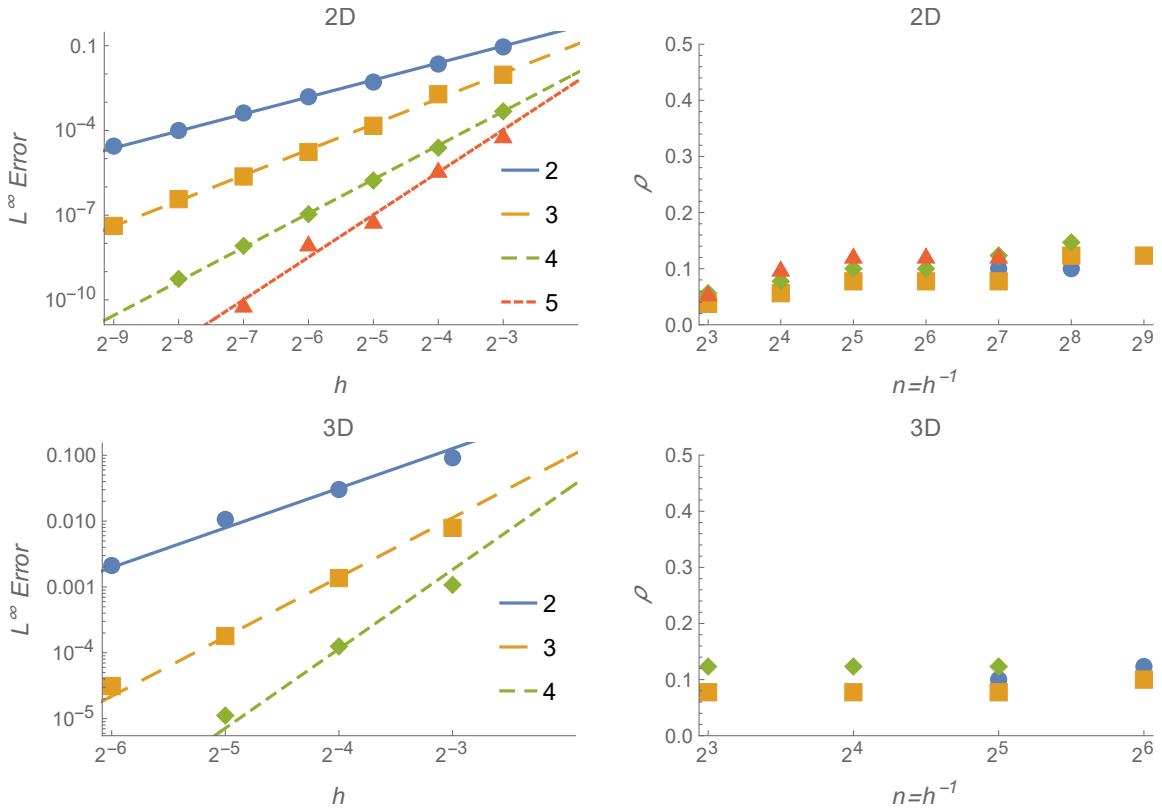


Figure 3.4: Convergence rates and multigrid performance for the Poisson problem with Robin boundary conditions on an implicitly-defined curved domain with $\mu, a, b = 1$. h denotes the background mesh spacing and polynomial degrees are represented by ●, ■, ♦, ▲ for $p = 1, 2, 3, 4$ respectively, with the slopes of the lines indicating asymptotic convergence rates.

3.5.2 Poisson problems with Robin jump conditions

We begin the analysis of our LDG methods for Poisson problems with Robin jump conditions by first recalling that our choice of interphase numerical fluxes along the Robin jump interface (3.14,3.15) has a directional weighting parameter $\lambda \in [0, 1]$ and that the numerical consistency of the discrete Laplacian with respect to the Robin jump conditions may depend on the flux direction. In [80], the direction of the interphase numerical fluxes was found to strongly affect both the multigrid performance and solution accuracy for the Poisson equation with standard jump conditions, particularly in cases exhibiting large jumps in the diffusion coefficient across the interface. For illustration, the Poisson equation with standard jump conditions is given by:

$$\begin{aligned} -\nabla \cdot (\mu \nabla u) &= f \\ [u] &= g_{ij} \\ [\mu \nabla u \cdot \boldsymbol{n}] &= h_{ij}, \end{aligned} \tag{3.35}$$

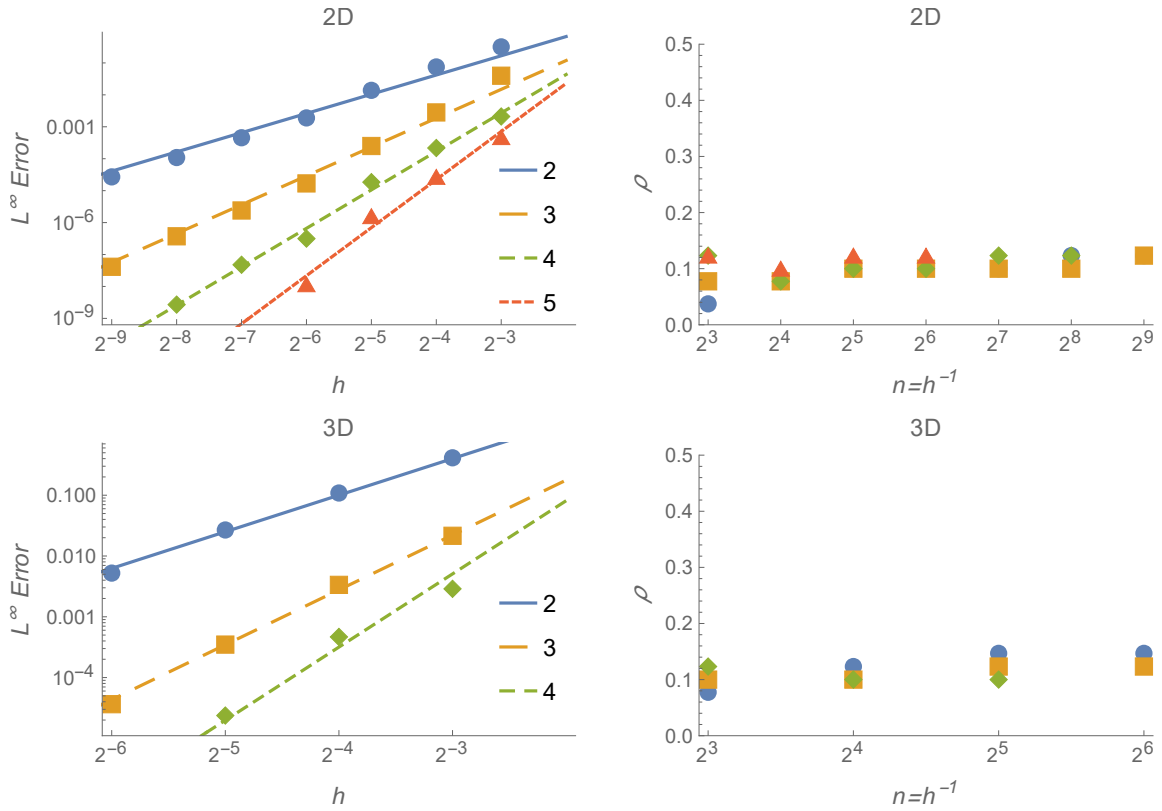


Figure 3.5: Convergence rates and multigrid performance for the Poisson problem with Robin boundary conditions on an implicitly-defined curved domain with variable μ, a, b spanning several orders of magnitude. h denotes the background mesh spacing and polynomial degrees are represented by ●, ■, ♦, ▲ for $p = 1, 2, 3, 4$ respectively, with the slopes of the lines indicating asymptotic convergence rates.

where f, g_{ij}, and h_{ij} are sufficiently smooth functions. In this setting, when there are large jumps in diffusivity across the surface, the phase with the smaller diffusion coefficient will, put simply, "see" the jump conditions as an almost Dirichlet boundary condition, while the phase with higher diffusion "sees" an almost Neumann condition. The *Viscosity-Upwinded Weighting* strategy of [80] biases the direction of the interphase numerical fluxes of the LDG discretization for (3.35)—equivalent to the numerical fluxes (3.14,3.15) without the Robin term $[\alpha u]\boldsymbol{n}$ within $\boldsymbol{q}^*$—to align with this effect. Specifically, viscosity-upwinded weighting sets λ such that

$$\lambda = \begin{cases} 0 & \text{if } \mu^- < \mu^+ \\ 1/2 & \text{if } \mu^- = \mu^+ \\ 1 & \text{if } \mu^- > \mu^+, \end{cases} \tag{3.36}$$

where μ^- and μ^+ are the local diffusion coefficients along the jump interface from the left and right elements' respectively. This choice of interphase numerical flux direction was

demonstrated to produce good multigrid performance and accurate solutions to (3.35) in situations with jumps in diffusion coefficient spanning several orders of magnitude. Note in particular that one-sided numerical fluxes (3.36) were found to produce optimal results [80].

We now examine the effect of biasing the directionality coefficient λ for the Robin jump conditions. Recall the Poisson equation with Robin jump conditions:

$$\begin{aligned} -\nabla \cdot (\mu \nabla u) &= f \\ [u] &= g_{ij} \\ [\alpha u] + [\mu \nabla u \cdot \boldsymbol{n}] &= r_{ij}. \end{aligned} \tag{3.37}$$

and our choice of numerical fluxes (3.14,3.15) for the Robin jump interface Γ_{ij}^R:

$$\begin{aligned} u_i^* &= \lambda u^- + (1-\lambda)(u^+ + g_{ij}), \quad \boldsymbol{q}_i^* = \lambda(\boldsymbol{q}^+ + r_{ij}\boldsymbol{n} - [\alpha u]\boldsymbol{n}) + (1-\lambda)\boldsymbol{q}^-, \\ u_j^* &= \lambda(u^- - g_{ij}) + (1-\lambda)u^+, \quad \boldsymbol{q}_j^* = \lambda \boldsymbol{q}^+ + (1-\lambda)(\boldsymbol{q}^- - r_{ij}\boldsymbol{n} + [\alpha u]\boldsymbol{n}). \end{aligned}$$

The term $[\alpha u]$ in the (3.37) introduces a source of *Impedance* to the flow of information across the interface, and we show that the direction of the interphase numerical fluxes for the Robin jump surface must take into account the values of the Robin coefficient α in addition to the values of diffusivity μ. To illustrate further, expanding the Robin jump condition gives the following value for u^-

$$u^- = \frac{1}{\alpha^-}\left(r_{ij} - (\mu^- \nabla u^- \cdot \boldsymbol{n} - \alpha^+ u^+ - \mu^+ \nabla u^+ \cdot \boldsymbol{n})\right).$$

Plugging this into the other jump condition ($[u] = g_{ij}$) gives

$$\left(1 - \frac{\alpha^+}{\alpha^-}\right) u^+ = \frac{1}{\alpha^-} r_{ij} - g_{ij} - \frac{1}{\alpha^-}\left(\mu^- \nabla u^- \cdot \boldsymbol{n} - \mu^+ \nabla u^+ \cdot \boldsymbol{n}\right). \tag{3.38}$$

Consider the limit $\alpha^- \to \infty$, then (3.38) approaches a Dirichlet boundary condition for the right phase[2] Ω^+ and, from a physical perspective, this limit represents a steep wall of impedance to the flow of information in Ω^+. Therefore, in situations where $\alpha^- \gg \alpha^+, \mu^-, \mu^+$, the interphase numerical fluxes (3.14,3.15) developed in section 3.3.1 for the Robin jump conditions should bias towards a Dirichlet condition for Ω^+. Specifically, the numerical flux $\boldsymbol{q}^*$ should bias towards $\boldsymbol{q}^+$ to maintain consistency with the specification of Dirichlet boundary conditions within the LDG framework and, to do so, one may set the directionality coefficient $\lambda = 1$. Similarly, if $\alpha^+ \gg \alpha^-, \mu^-, \mu^+$, then the interphase numerical fluxes for the Robin jump conditions should bias towards a Dirichlet condition for Ω^- and $\boldsymbol{q}^*$ should be biased towards $\boldsymbol{q}^-$, setting $\lambda = 0$. If either of the diffusion coefficients μ^-, μ^+ outweighs the Robin coefficients, then the flow of information is most strongly specified by diffusion and the viscosity-upwinded numerical fluxes should be used. Combining all cases gives the

[2] In terms of the previous notation: $\Omega^- = \Omega_i,\ \Omega^+ = \Omega_j$.

following *Viscosity-Impedance Upwinded* weighted numerical fluxes for the Poisson equation with Robin jump conditions, defined by

$$\lambda = \begin{cases} 0 & \text{if } \alpha^+ > \alpha^-, \mu^-, \mu^+ \\ 1 & \text{if } \alpha^- > \alpha^+, \mu^-, \mu^+ \\ 0 & \text{if } \mu^+ > \mu^-, \alpha^-, \alpha^+ \\ 1/2 & \text{if } \mu^+ = \mu^- > \alpha^-, \alpha^+ \\ 1 & \text{if } \mu^- > \mu^+, \alpha^-, \alpha^+. \end{cases} \tag{3.39}$$

To illustrate the role of the directionality coefficient λ and to further motivate the viscosity-impedance upwinding strategy, we test the computed LDG solutions for the Poisson problem with Robin jump conditions (3.37) as a function of λ, using a similar sweeping strategy as in [80]. In this sweep, the directionality coefficient is varied from $\lambda = 0$ through $\lambda = 0.5$ to $\lambda = 1$, representing varying the interphase numerical fluxes through a convex-combination of flux directions: starting from a one-sided formulation onto a central flux, and finishing with a one-sided flux in the opposite direction. Additionally, between these values, λ is set to $\lambda = 10^{-k}$ and $\lambda = 1 - 10^{-k}$, where $k = 10.5, 10, 9.5, ..., 1.5, 1$. These tests are performed in a unit square in 2D with Robin jump conditions applied on an embedded circle separating the two phases. A 16×16 DG background mesh is employed and Dirichlet conditions are applied on the domain boundary to close the Poisson system.

Figure 3.6(i) shows the L^∞ errors against the exact offset sinusoidal solution (3.33) for the λ sweep. Here the diffusion coefficients are set to unity $\mu^- = \mu^+ = 1$, and two different sets of Robin coefficients are considered: (1) $\alpha^- = 10^4$, $\alpha^+ = 10^8$ and (2) $\alpha^- = 10^8$, $\alpha^+ = 10^4$. The figure clearly illustrates the effect of biasing the direction of the interphase numerical fluxes on solution accuracy, with the direction given by viscosity-impedance upwinding resulting in accurately computed solutions. For further motivation, recall the definition of the block sparse Robin matrix $A_R(u)$ (3.26), which when restricted to the Robin jump conditions is given by

$$v^T A_R(u) u = \int_{\Gamma_{ij}^R} \lambda v^-[\alpha u] + (1-\lambda) v^+[\alpha u],$$

for all $v \in V_h$. The term A_R breaks the symmetry of the Laplacian and allows it to support complex eigenvalues. Figure 3.6(ii) shows the maximum imaginary eigenvalue of the discrete Laplacian for a given value of λ. The blank regions in the figure indicate that there are no complex eigenvalues, and we note that these regions align with the direction given by viscosity-impedance upwinding. Consider, for example, when $\alpha^- > \alpha^+, \mu^-, \mu^+$, then the viscosity-impedance upwinding strategy sets $\lambda = 1$ and the resulting term in A_R is $\int_{\Gamma_{ij}^R} v^-[\alpha u]$. The term $v^- \alpha^- u^-$ has a larger value than that of $v^- \alpha^+ u^+$, with the former term being along the matrix diagonal. Therefore, the choice of viscosity-impedance upwinding results in a discrete Laplacian that is more diagonally dominant and, in this test, contains all real eigenvalues. From a physical point of view, the complex eigenvalues arising from a

non-optimal choice of numerical flux direction may indicate reflection waves caused by the impedance of the Robin jump surface.

Figure 3.6(iii) highlights a case where the optimal flux direction given by viscosity-upwinding for the Poisson problem with standard jump conditions reverses in the presence of Robin jump conditions. This figure presents the L^∞ errors of the computed solution for two cases: (1) $\mu^- = 10^{-6}, \mu^+ = 10^2, \alpha^- = \alpha^+ = 0$ and (2) $\mu^- = 10^{-6}, \mu^+ = 10^2, \alpha^- = 10^8, \alpha^+ = 0$. The impedance forces a reversal of the optimal flow of information and, in this situation, the numerical flux direction given by viscosity-impedance upwinding is optimal.

Figure 3.6(iv) shows the results of a convergence study for the Poisson problem with Robin jump conditions (3.37) using the viscosity-impedance upwinding strategy. This test considers a challenging case where the diffusion coefficients vary by six orders of magnitude and the Robin coefficients eight orders of magnitude, setting

$$\begin{aligned}
\mu^- &= 10^{3\sin(2\pi(x-0.1))\sin(2\pi(y+0.1))} \\
\mu^+ &= 10^{3\sin(2\pi(x+0.2))\sin(2\pi(y-0.2))} \\
\alpha^- &= 10^{-4+8\sin(\pi x/2)\sin(\pi y/2)} \\
\alpha^+ &= 10^{4-8\sin(\pi x/2)\sin(\pi y/2)}.
\end{aligned}$$

The test is performed in 2D within the two-phase amoeba domain (see Figure 3.3) and the PDE is closed with Dirichlet boundary conditions. The figure illustrates slightly better convergence rates than the optimal $p+1$ order accuracy, for both the L^2 and L^∞ norms. This is likely due to the mesh resolutions considered here not fully reaching the asymptotic limit, noting that here the asymmetric discrete linear system (3.31) is inverted through a dense solver. Also, for this reason, tests of the LDG method for Poisson problems with Robin jump conditions are only presented in 2D. As with the Robin boundary problem, the LDG framework for Poisson problems with Robin jump conditions is constructed in a manner that naturally extends to 3D.

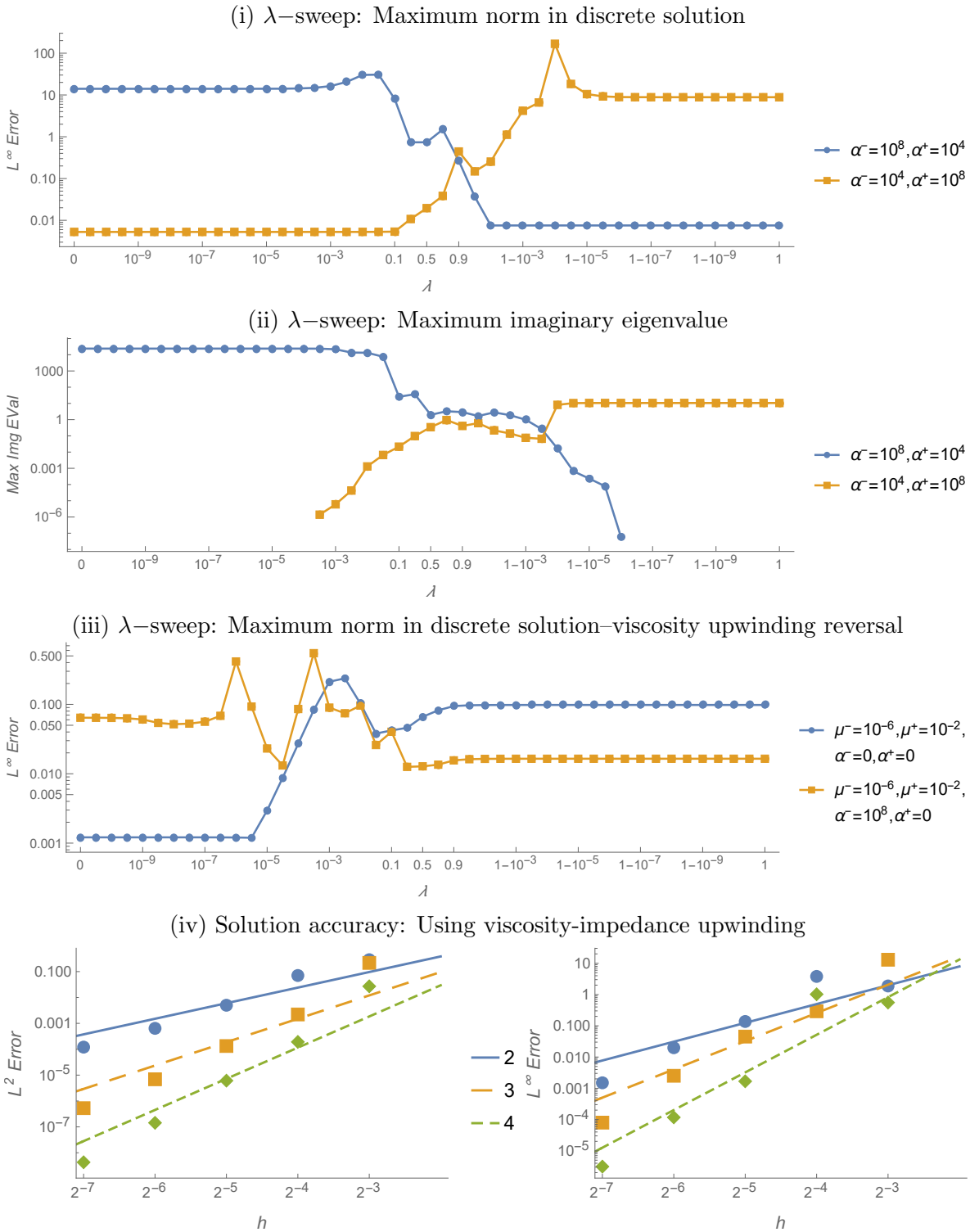


Figure 3.6: The test results for Poisson problems with Robin jump conditions. In (iv), h denotes the background mesh spacing and polynomial degrees are represented by $\bullet, \blacksquare, \blacklozenge$ for $p = 1, 2, 3$ respectively, with the slopes of the lines indicating asymptotic convergence rates.

Chapter 4

Numerical Methods for Coating Flow Dynamics

In Chapter 2, a multi-physics model for the multi-layer coating flow problem is presented. The model includes multi-phase interfacial quasi-Newtonian fluid dynamics, driven primarily by Marangoni forces, coupled to the transport, mixing, and evaporation of multiple dissolved species. To solve the set of coupled multi-physics equations outlined in section 2.1.6, in this chapter, we develop a hybrid numerical framework for the multi-layer coating flow problem consisting of finite difference level set methods [36][37] and high-order accurate multi-phase implicit mesh discontinuous Galerkin methods [66][75][79]. These methods use an implicit level set representation of the paint surfaces combined with a structured background quadtree or octree to create a collection of rectangular and interface-conforming curved elements on which high-order accurate DG methods are applied. The methods sharply capture evolving interface dynamics to high-order accuracy in a dimension-independent fashion. The numerical methods discussed in this chapter were developed in conjunction with the jointly-authored article

- L. P. Corcos, R. I. Saye, & J. A. Sethian, A hybrid finite difference level set–implicit mesh discontinuous Galerkin method for multi-layer coating flows, *Journal of Computational Physics* (under review).

The preliminaries of our hybrid numerical framework for the multi-layer coating flow problem are discussed in Chapters 2 and 3. In section 2.2, we describe the finite difference level set methods that capture surface motion; specifically, the height function representation for the surfaces $\Gamma_{\text{sub}}, \Gamma_{ij}, \Gamma_e$ that is used to create a level set function that generates the numerical quadrature rules for the implicit mesh DG methods. Additionally, in section 2.2.1, finite difference methods for calculating surface mean curvature within surface tension calculations are discussed. In section 3.2, the implicit mesh DG framework is discussed, including the notion of implicitly-defined meshes and their evolution, the definition of the DG polynomial spaces, and a discussion of numerical quadrature schemes for implicitly-defined elements

and surfaces. Tailored high-order accurate LDG methods for Poisson problems with Robin boundary conditions are developed in section 3.3.

In this chapter, we present the numerical methods for the multi-layer coating flow problem, capturing the couplings between the evaporative solvent mass transfer system and the multi-phase interfacial quasi-Newtonian fluid dynamics. First, we combine the new LDG formulation of section 3.3 for Poisson problems with Robin boundary conditions and the level set method to capture solvent evaporation. Next, the methods for solving quasi-Newtonian fluid flow are discussed, including the application of recently developed fast multigrid Stokes solvers [72] and the presentation of a finite difference projection algorithm for surface gradient calculations for Marangoni stresses. Lastly, the fully coupled algorithm for the multi-layer coating flow problem is presented and its convergence properties assessed.

4.1 Algorithm outline

We now outline the basic structure of our numerical algorithm for the multi-layer coating flow problem. In the present setting, it is especially important to capture spatial characteristics with sufficiently high-order accuracy, such as thin boundary layers. All simulations of multi-layer coatings use $p = 2$ order DG polynomials (i.e., bi-quadratic and tri-quadratic polynomial spaces in 2D and 3D, respectively), together with 2^{nd} order accurate finite difference methods. On the other hand, temporal accuracy is less important. Consequently, and for ease of implementation, we here describe a simple 1^{st} order mixed explicit-implicit time stepping method, where advective terms are treated explicitly and the diffusive and viscous terms are solved implicitly via backward Euler. Together, our particular implementation choices lead to a fully coupled numerical algorithm for the multi-layer coating flow problem that is 2^{nd} order accurate in space and 1^{st} order in time, demonstrated in section 4.5.1.

The developed numerical methods for approximating the multi-phase incompressible Navier-Stokes equations (2.2,2.21) and the convection-diffusion equations (2.3) treat advective terms explicitly-in-time, with numerical fluxes given by standard upwinding [66]. The viscous and diffusive terms are solved implicitly-in-time via backward Euler, requiring solutions to the Stokes and heat operator problems respectively. The algorithm for the multi-layer coating flow problem proceeds as follows:

1. At time step 0, initialize the height functions Γ_{sub}, Γ_{ij}, and Γ_e to create the initial interface configuration; construct the associated implicitly-defined mesh, DG polynomial spaces, and LDG operators; and set the initial DG state variables for the solvent mass concentrations c_k, paint velocity field $\boldsymbol{u}$, and pressure p.
2. Begin time stepping: For $n = 0, 1, 2, ...$

    (i) Compute the concentration and velocity field advection terms via an upwinding scheme.

(ii) Evolve the height functions under (2.32) to find ϕ^{n+1}. Create a new implicit mesh, transfer state variables onto the new mesh's DG polynomial spaces, and create LDG operators.

(iii) Update the solvent mass concentrations c_k^{n+1} by solving the mass transfer convection-diffusion equations (2.3), use this to determine the new concentration-dependent rheological parameters.

(iv) Update the velocity field $\boldsymbol{u}^{n+1}$ and pressure p^{n+1} by solving the incompressible Navier-Stokes equations (2.2,2.21) for quasi-Newtonian fluid dynamics.

(v) Repeat until the final time is reached.

The numerical methods for the evaporative mass transfer system (step (iii)) and the Marangoni-driven multi-phase quasi-Newtonian fluid dynamics (step (iv)) are presented in the upcoming sections. These methods are coupled together in the full numerical algorithm for the multi-layer coating flow problem, described in section 4.5.

4.2 Numerical methods for the mass transfer system

As motivated in section 2.1, the motion of solvent within the multi-layer coating flow problem is described by the convection-diffusion equations (2.3) coupled to evaporation and mixing at the paint-gas and paint-paint surfaces. Applying the mixed explicit-implicit time stepping scheme to (2.3) gives the following heat operator system for the mass transfer problem, solving for c_k^{n+1} such that

$$\frac{c_k^{n+1} - c_k^n}{\Delta t} + \nabla \cdot (c_k \boldsymbol{u})^n = \nabla \cdot \left(D_k \nabla c_k^{n+1} \right), \tag{4.1}$$

together with the Robin boundary condition $m_k = m c_k^{n+1} - \rho D_k \nabla c_k^{n+1} \cdot \boldsymbol{n}$ along Γ_e, as well as continuity and zero diffusive flux jump conditions along Γ_{ij}. The advective term $\nabla \cdot (c_k \boldsymbol{u})^n$ is treated explicitly via an upwinding scheme and the Robin boundary problem is solved by the LDG schemes discussed in section 3.3. As previously demonstrated, these LDG methods calculate high-order accurate solutions on implicitly-defined domains in a dimension-independent fashion, for a wide range of variable Robin coefficients. Solving the heat operator equation (4.1) with Robin boundary condition requires only a simple modification of the LDG schemes of section 3.3 for the Poisson equation (see equation (3.31)), setting the LDG heat operator to

$$\frac{M}{\Delta t} + \left(\sum_k G_k^T M_\mu G_k + A_R + E \right), \tag{4.2}$$

where M is the mass matrix. The heat operator linear system is symmetric positive-definite and equation (4.1) is solved via the fast multigrid-preconditioned conjugate gradient algorithms discussed in section 3.4.

4.2.1 Solvent mass flux

The solvent evaporative mass flux m_k through surface Γ_e accounts for two aspects: first, the evaporation rate should be proportional to the amount of the solvent at the interface and tend towards zero as the solvent mass concentration goes to zero; second, in multi-solvent cases, the solvent with the largest species concentration is preferential to evaporation. The following solvent evaporation rate m_k takes into account both considerations:

$$m_k = \frac{\varepsilon}{C} \cdot \frac{\left(c_k^n\right)^2}{\sum_j c_j^n}, \tag{4.3}$$

where ε is the coefficient of evaporation, a tunable, application-defined parameter that can incorporate additional physics if necessary. In all case studies in this work, this coefficient is taken to be a constant whose value is chosen to match experimental data. Note that (4.3) is normalized by the number of solvents C to ensure equivalent dynamics between the single solvent case and the case where multiple solvents all have equal mass diffusion and evaporation coefficients. Also note that the evaporative process introduces a fully non-linear constraint to the mass transfer system. This nonlinearity is treated in our model by using the solvent mass concentration values from the previous time step (i.e., from the traces of c_k^n) in the definition of the solvent mass flux m_k (4.3) and total evaporation rate m. The remaining description of the evaporation process is then the linear Robin boundary condition (2.28).

Additionally, for the inter-paint mixing rate (2.29), normal derivatives of the solvent mass concentration are computed directly from the DG polynomials c_k^n. The computed evaporation and mixing rates are embedded within the level set speed law (2.19) and used in the advection equation (2.32). After the solvent mass transfer system is advanced to the next time step, an updated value of resin mass concentration $c_R^{n+1} = 1 - \sum_k c_k^{n+1}$ is computed to define the concentration-dependent rheological parameters (viscosity and surface tension) for the quasi-Newtonian fluid dynamics.

4.3 Numerical methods for quasi-Newtonian fluid dynamics

To capture the flow and leveling of the quasi-Newtonian liquid paint films, we wish to solve the multi-phase incompressible Navier-Stokes equations (2.2,2.21) wherein the viscosity varies with respect to resin mass concentration, while also incorporating the various boundary and jump conditions that capture the effects of surface tension, Marangoni forces, and the couplings between film-layers. To do so, we use the LDG Stokes solver of [72] in conjunction with finite difference methods for computing interfacial curvature and surface gradients for the calculations of surface tension and Marangoni stresses respectively.

In more detail, our mixed first-order explicit-implicit time stepping scheme applied to (2.2, 2.21) results in the following Stokes system for updating the fluid velocity $\boldsymbol{u}^{n+1}$ and pressure p^{n+1}:

$$\rho\left(\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^{n}}{\Delta t}\right)+\nabla\cdot(\rho\boldsymbol{u}\boldsymbol{u})^{n}=-\nabla p^{n+1}+\nabla\cdot\left(\mu(c_R^{n+1})\left(\nabla\boldsymbol{u}^{n+1}+(\nabla\boldsymbol{u}^{n+1})^{T}\right)\right)+\rho\boldsymbol{g},$$
$$\nabla\cdot\boldsymbol{u}^{n+1}=0, \tag{4.4}$$

including the stress conditions (2.25,2.26) along the embedded paint-paint surfaces Γ_{ij} and free evaporative surface Γ_e, as well as the no-slip conditions on Γ_{ij} and substrate Γ_{sub}. These conditions are repeated below:

$$\begin{aligned}
&\text{On } \Gamma_e \quad && \boldsymbol{\sigma}^{n+1}\cdot\boldsymbol{n}=-p_{ext}\boldsymbol{n}-\gamma(c_R^{n+1})\kappa^{n+1}\boldsymbol{n}+\nabla_S\gamma(c_R^{n+1})\\
&\text{On } \Gamma_{ij} \quad && [\boldsymbol{u}^{n+1}]=0\\
&\qquad\qquad && [\boldsymbol{\sigma}^{n+1}\cdot\boldsymbol{n}]=-\gamma(c_R^{n+1})\kappa^{n+1}\boldsymbol{n}+\nabla_S\gamma(c_R^{n+1})\\
&\text{On } \Gamma_{\text{sub}} \quad && \boldsymbol{u}^{n+1}=0.
\end{aligned}$$

In (4.4), the advection term $\nabla\cdot(\rho\boldsymbol{u}\boldsymbol{u})^n$ is discretized by an upwinding scheme and the viscous components via backward Euler. The term κ^{n+1} represents the mean curvature of the interface at time step $n+1$, which is calculated using the second-order finite difference methods discussed in section 2.2.1. The concentration-dependent rheological parameters of viscosity μ and surface tension γ are scalar functions of the updated resin mass concentration c_R^{n+1}. The Marangoni stresses $\nabla_S\gamma(c_R^{n+1})$ are computed via a second-order accurate version of the finite difference projection algorithm for surface gradient calculations presented in section 4.3.2.

The time-dependent Stokes system (4.4) is solved by LDG schemes that provide high-order accurate solutions on multi-phase implicitly-defined domains while seamlessly incorporating the paint layer couplings, varying viscosity profile, and the boundary and jump conditions. This formulation enforces the divergence constraint without an intermediate projection step and, in [72], rapid multigrid performance is achieved when the proper pressure penalty parameter is chosen, with performance matching that of classical geometric multigrid methods applied to Poisson problems. In the remainder of this section, we summarize the LDG methods for solving the time-dependent Stokes system (4.4) and then present our finite difference surface gradient formulation for Marangoni stress calculations. For full details on the LDG Stokes solver, the reader is referred to [72]; also a detailed derivation of a new LDG method for Stokes problems with Navier-slip boundary conditions is presented in Appendix A.

4.3.1 LDG for the Stokes equations

The LDG methods of [72] for the Stokes equations are constructed similarly as that for the Poisson equation [66]; see section 3.3 for details on LDG methods for Poisson problems with

Robin boundary and jump conditions. For illustration, recall the time-dependent Stokes equations:

$$\begin{aligned}(\rho/\delta t)\boldsymbol{u} - \nabla\cdot(\mu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T)) + \nabla p &= \boldsymbol{f} \\ -\nabla\cdot\boldsymbol{u} &= f_{\text{div}},\end{aligned} \tag{4.5}$$

where δt is a constant often associated with a discrete time-step. The LDG methods for the Stokes equations (4.5), as well as the new LDG methods for Stokes problems with Navier-slip boundary conditions presented in Appendix A, follow five steps:

(i) Introduce the gradient $\boldsymbol{\eta} \in V_h^{d\times d}$ such that $\boldsymbol{\eta} = \nabla\boldsymbol{u}$ weakly via the strong-weak form.

(ii) Define the stress-tensor $\boldsymbol{\sigma} \in V_h^{d\times d}$ as the L^2 projection of $\mu(\boldsymbol{\eta} + \boldsymbol{\eta}^T) - p\mathbb{I}$.

(iii) Compute the divergence of the stress-tensor $\boldsymbol{w} \in V_h^d$ such that $\boldsymbol{w} = \nabla\cdot\boldsymbol{\sigma}$ weakly via the weak-weak form.

(iv) Enforce the divergence constraint $w \in V_h$ such that $w = \nabla\cdot\boldsymbol{u}$ via the strong-weak form.

(v) Require that $((\rho/\delta t)\boldsymbol{u} - \boldsymbol{w}, -w)$ equals the L^2 projection of $(\boldsymbol{f}, f_{\text{div}})$, while also incorporating penalty stabilization to enforce continuity, boundary, and jump conditions.

After performing these five steps, the LDG discretization for the time-dependent Stokes equations (4.5) describing the quasi-Newtonian fluid dynamics within the multi-layer coating flow problem results in the following symmetric block-form linear system

$$\begin{pmatrix} A & M\mathscr{G} \\ \mathscr{G}^T M & -E_p \end{pmatrix}\begin{pmatrix} \boldsymbol{u} \\ p \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ b_{\text{div}} \end{pmatrix}, \tag{4.6}$$

where M is the mass matrix, $\mathscr{G} = -M^{-1}G^TM$ is a discrete gradient operator[1], $\mathscr{G}^T$ is a discrete divergence operator that enforces the divergence constraint, E_p is the pressure penalty operator, and A is the $d\times d$ block operator containing the temporal and viscous components of the momentum equation, with

$$A_{ij} = \delta_{ij}\left(\frac{M_\rho}{\delta t} + \sum_{k=1}^{d} G_k^T M_\mu G_k + E_{\boldsymbol{u}}\right) + G_j^T M_\mu G_i, \tag{4.7}$$

where δ_{ij} is the Kronecker delta function, M_ρ and M_μ are the ρ and μ-weighted mass matrices respectively such that $v^T M_\rho u = \int_\Omega v\rho u$ and $v^T M_\mu u = \int_\Omega v\mu u$ for all $u, v \in V_h$, and $E_{\boldsymbol{u}}$ is the velocity penalty operator[2]. The right-hand side $(\boldsymbol{b}, b_{\text{div}})$ of linear system (4.6) combines the source, jump, and boundary data present within the Stokes system. The degrees of freedom for $\boldsymbol{u}$ and p are blocked together on an element-wise basis and the block-sparse linear system may be written as $\mathscr{A}x = b$, where $x = (\boldsymbol{u}, p)$. The resulting Stokes operator $\mathscr{A}$ is symmetric indefinite and the system (4.6) is solved via operator coarsening multigrid-preconditioned GMRES methods [72]. A few notes are in order:

[1]The lifting operator is modified from its form in Appendix A to impose Dirichlet boundary and interfacial jump conditions.

[2]The penalty operator $E_{\boldsymbol{u}}$ is similarly modified.

- Key to rapid multigrid performance is the choice of pressure penalty stabilization parameter τ_p within the discrete pressure penalty operator E_p, which is defined over all intraphase faces such that

$$v^T E_p p = \int_{\Gamma_0} \tau_p [v][p], \tag{4.8}$$

for all $v \in V_h$, where

$$\tau_p = \left(\frac{\mu}{\tau h} + \frac{h\rho}{\tau_0 \delta t} \right)^{-1} \tag{4.9}$$

harmonically averages the penalty weightings between the viscous and temporal components of the Stokes operator. Here τ and τ_0 are user-defined penalty parameters; optimal values of τ that lead to superior multigrid performance are given in [72]. This work uses the optimal pressure penalty parameter for the viscous-stress form Stokes operator and polynomial order $p = 2$: setting $\tau = 0.046$ in 2D and $\tau = 0.039$ in 3D, with the temporal penalty parameter set to $\tau_0 = 0.5p$.

- Additional speedups in multigrid performance can be achieved by a suitable rescaling of the discrete Stokes operator, effectively so that it has unit viscosity and unit length scale. Here we apply diagonal pre- and post-scalings of the Stokes operator, replacing (4.6) with

$$\begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix} \begin{pmatrix} A & M\mathscr{G} \\ \mathscr{G}^T M & -E_p \end{pmatrix} \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix},$$

where α and β are diagonal matrices with entries equal to $\sqrt{L/\mu}$ and $\sqrt{\mu/L}$ respectively, with L being the characteristic length scale of the problem ($L = 100\mu$m in the multi-layer coating flow problem). The scaled problem $\tilde{\mathscr{A}}\tilde{x} = \tilde{b}$ is then solved, where $\tilde{\mathscr{A}}$ is the scaled Stokes operator and $\tilde{b} = \text{diag}(\alpha, \beta)b$, after-which the original unscaled solution $x = \text{diag}(\alpha, \beta)\tilde{x}$ is computed. We note that this scaling is applied at each level of the multigrid hierarchy in the operator coarsening framework.

- Jumps in viscosity across phases are accurately handled by the application of the viscosity-weighted upwinding strategy of [80], which biases the direction of the interphase numerical fluxes based on the liquids' local viscosity coefficients.

4.3.2 Finite difference Marangoni formulation

To properly capture the Marangoni forces present within the multi-layer coating flow problem, which occur as a consequence of surface tension variations caused by solutal concentration gradients along the paint-gas and paint-paint surfaces, we have developed a finite difference algorithm for calculating surface gradients along an implicit interface defined via a height function. The algorithm can be stated in two equivalent manners, either as an extension (from interfacial values into the volumetric region) or a vertical projection (from

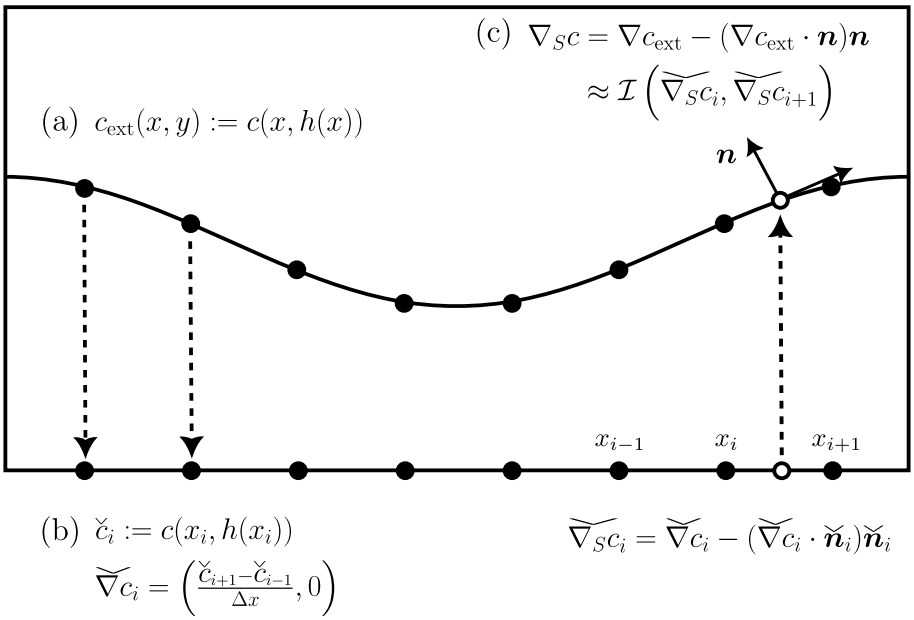


Figure 4.1: A 2nd-order example of the finite difference projection algorithm for surface gradient calculations in 2D. (a) Concentration values are projected downwards onto the finite difference nodes ● of the height function along the $d-1$ dimensional plane. (b) On this plane, the projected surface gradients are approximated through standard central finite differences. (c) The surface gradient approximation at point ○ along the height function is then the interpolated projected surface gradient lifted upwards from the horizontal plane.

interfacial values down onto a fictitious $d-1$ dimensional flat plane). We present both formulations, examining the surface gradient $\nabla_S c$ of a general species mass concentration in 2D, with 3D needing only a simple modification of the following presentation. The extension formulation extends the concentration values vertically from the height function, via the function $c_{\text{ext}}(x, y) := c(x, h(x))$. It is clear that

$$\nabla_S c = \nabla c_{\text{ext}} - (\nabla c_{\text{ext}} \cdot \boldsymbol{n})\boldsymbol{n}, \tag{4.10}$$

where $\boldsymbol{n}$ is the normal vector of the interface. The goal of the projection formulation is to approximate (4.10) via a finite difference method in the $d-1$ dimensional horizontal plane, specifically at the finite difference nodes on which the height function is defined. The projection formulation proceeds following the steps outlined in Figure 4.1.

(a) Project the values of c downwards from the height function onto the $d-1$ dimensional plane, setting $\breve{c}_i = c(x_i, h(x_i))$. Here $\breve{c}_i$ represents the projected value at finite difference node $x_i \in \mathbb{R}^{d-1}$ and the height function values $c(x_i, h(x_i))$ are computed from the traces of the DG polynomials. Additionally, the normal vector $\boldsymbol{n}$ is projected onto the $d-1$ dimensional plane, where

$$\text{2D:} \quad \boldsymbol{n} = \frac{1}{\sqrt{h_x^2 + 1}}(-h_x, 1), \qquad \text{3D:} \quad \boldsymbol{n} = \frac{1}{\sqrt{h_x^2 + h_y^2 + 1}}(-h_x, -h_y, 1),$$

setting $\breve{\boldsymbol{n}}_i = \boldsymbol{n}(x_i, h(x_i))$, with the derivatives of the height function calculated via central finite differences.

(b) Next the projected gradient $\widetilde{\nabla c_i}$ is computed via central finite differences. For example, a 2^{nd} order formulation in 2D is given by

$$\widetilde{\nabla c_i} = \left(\frac{\breve{c}_{i+1} - \breve{c}_{i-1}}{\Delta x}, 0 \right). \tag{4.11}$$

(c) The projected surface gradient is then calculated at each finite-difference node, with

$$\widetilde{\nabla_S c_i} = \widetilde{\nabla c_i} - (\widetilde{\nabla c_i} \cdot \widetilde{\boldsymbol{n}_i})\widetilde{\boldsymbol{n}_i}. \tag{4.12}$$

The values of the projected surface gradient are now available along the $d-1$ dimensional plane through interpolation. The surface gradient approximation $\nabla_S c \approx \mathcal{I}(\widetilde{\nabla_S c})$ is therefore the interpolated value projected vertically back onto the height function.

Figure 4.1 illustrates a 2^{nd} order accurate formulation of the surface gradient projection algorithm in 2D. The extension to 3D is straightforward, requiring only an additional dimension to the definitions of the normal vector and projected gradient. Figure 4.2(right) shows the results of a convergence study examining the finite difference projection algorithm for computing surface gradients within the Marangoni stress calculations. The L^∞ errors in the computed solutions against the surface gradient of an exact offset sinusoidal solution (3.33) are presented for both 2^{nd} and 4^{th} order formulations. 3D tests are performed along the surface of Figure 4.2(left) and 2D tests are performed along a central slice of this surface. The surface is embedded within a $[0, L]^d$ box, with $L = 100\mu$m to match the micro-fluidic domains of interest. The 2^{nd} order formulations use 2^{nd} order central differences with $p = 2$ DG polynomials and the 4^{th} order formulations use 4^{th} order central differences with $p = 3$ DG polynomials; the former are used in the simulations of multi-layer coatings. We have opted to use the 2nd order method in the remainder of the results presented in this work; this is simply for reasons of consistency, e.g., our finite difference curvature calculation (see section 2.2.1) is also second-order. This choice renders the full algorithm for the multi-layer coating flow problem spatially 2^{nd} order accurate.

4.4 A note on time step restrictions

In the micro-flow regime of coating flow dynamics, strong surface tension forces impose a severe constraint on the numerical time step size. Fully resolving small-scale capillary wave dynamics requires limiting the numerical time step within our hybrid framework to $\Delta t < Ch\mu/\gamma$, for some constant C, with h being the mesh spacing[3]. Typical simulations

[3]We note our time step restriction's linear $\mathcal{O}(h)$ dependence, as opposed to the $\mathcal{O}(h^{3/2})$ restriction commonly seen in capillary wave dynamics.

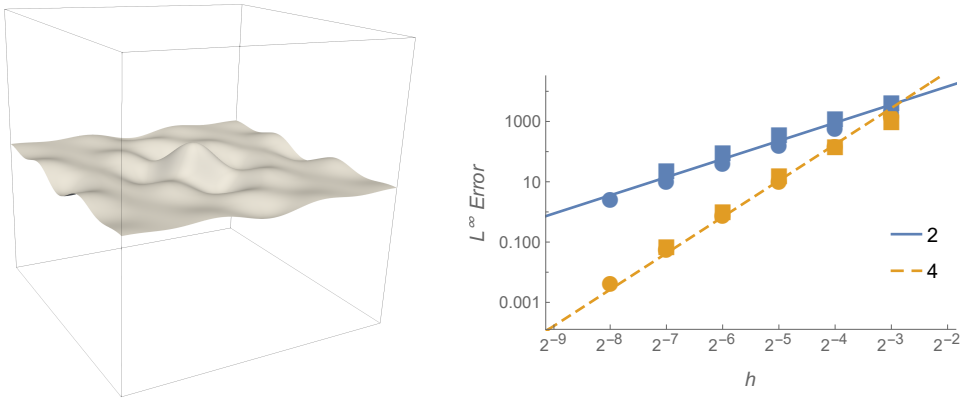


Figure 4.2: Convergence tests of the finite difference surface gradient calculation. **Left:** The 3D surface on which the tests are performed; 2D tests are performed along a central slice of this surface. **Right:** L^∞ errors and convergence rates of the finite difference surface gradient calculation for 2^{nd} and 4^{th} order formulations, with ●, ■ representing 2D and 3D calculations respectively, with the slopes of the lines indicating asymptotic convergence rates.

require a time step size around 10^{-4} seconds, which in conjunction with the long time scales relevant to the coating flow problem—on the order of hundreds of seconds—necessitates millions of time steps. Fortunately, the dynamics of paint drying itself provides a source of time step speedup. Typical paints used in industrial applications have a viscosity that, roughly speaking, increases exponentially as a function of resin concentration. Since the time step constraint is directly proportional to μ/γ, and μ tends to increase as solvent evaporates, it follows that we can increase the time step size as a simulation progresses. In our implementation, we update the time step size every 100 time steps, according to the formula

$$\Delta t = \min_{c_R} \left(\frac{\mu(c_R)}{\gamma(c_R)} / \frac{\mu_0}{\gamma_0} \right) \Delta t_0, \tag{4.13}$$

where Δt_0 is the initial stable time step size required for the initial ratio of viscosity to surface tension μ_0/γ_0. Here, the minimum value of resin concentration (i.e., c_R) is taken from the values of the DG polynomials at the Gauss-Lobotto nodes. This adaptive time stepping approach is a straightforward yet effective means to take advantage of the fact the capillary number is rapidly monotonically increasing: by the end of a typical simulation, the final time step can be more than 100 times larger than the initial time step.

Even with this large speedup, further mitigation of the capillary wave time step constraint was necessary in the present work: one of our objectives here is to explore several aspects of multi-layer coating flow, requiring a multitude of parametric studies. To help facilitate this, in addition to the above adaptive time stepping, we also damp the force of surface tension whereby the terms in (2.25,2.26) involving κ are multiplied by 0.1. This ad-hoc alteration is simply so that we can take time steps ten times larger, representing a modest but beneficial speedup. This alteration has little effect on the framework's ability to capture short-wave

Marangoni instabilities and plume structures; however the extent of the surface deformation patterns—caused primarily by the long-wave Marangoni instabilities—will be exaggerated. Nevertheless, we believe the insights drawn from our numerical simulations on multi-layer coatings, presented in Chapter 5, remain physically relevant.

The net effect of the procedures outlined in this section is to reduce the total number of time steps from millions to around 30,000–100,000 for the presented results.

4.5 Numerical algorithm for the multi-layer coating flow problem

Combining the numerical methods developed in the previous sections, the fully coupled numerical algorithm for the multi-layer coating flow problem is:

1. Set up the background quad/octree.
2. Define the initial height functions for surfaces $\Gamma_{\text{sub}}, \Gamma_{ij}, \Gamma_e$ and construct the initial level set function ϕ^0.
3. Use ϕ^0 to define the initial implicit mesh, DG polynomial spaces, and LDG operators.
4. Initialize the DG state variables $\boldsymbol{u}^0, p^0$, and c_k^0 at time $t = 0$.
5. For time step $n = 0$, 1, 2, ...

   (i) Every 100 time steps, update the time step size Δt.

   (ii) Compute advection terms: $\nabla \cdot (c_k \boldsymbol{u})^n, \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u})^n$.

   (iii) Calculate inter-paint mixing rates $m = -\frac{\rho}{c_R^n} \sum_k D_k \nabla c_k^n \cdot \boldsymbol{n}$ on Γ_{ij},
   solvent evaporation rates $m_k = \frac{\varepsilon}{C} \cdot \frac{\left(c_k^n\right)^2}{\sum_j c_j^n}$ on Γ_e,
   and total evaporation rate $m = \sum_k m_k$.

   (iv) Determine the interfacial speed functions (2.19): $V = \boldsymbol{u} \cdot \boldsymbol{n} - \frac{1}{\rho} m$.
   Advect the height functions (2.32) and update the level set function ϕ^{n+1}.

   (v) Use ϕ^{n+1} to create a new implicit mesh for time step $n + 1$.

   (vi) Transfer all necessary quantities onto the new mesh's DG polynomial spaces and define LDG operators.

   (vii) Solve the solvent mass transfer heat operator problem (4.1) for each c_k^{n+1} such that

$$\begin{cases} \left(\frac{\mathbb{I}}{\Delta t} - \nabla \cdot (D_k \nabla)\right) c_k^{n+1} = \frac{c_k^n}{\Delta t} - \nabla \cdot (c_k \boldsymbol{u})^n & \text{in } \Omega \\ \nabla c_k^{n+1} \cdot \boldsymbol{n} = 0 & \text{on } \Gamma_{\text{sub}} \\ [c_k^{n+1}] = 0 & \text{on } \Gamma_{ij} \\ [D_k \nabla c_k^{n+1} \cdot \boldsymbol{n}] = 0 & \text{on } \Gamma_{ij} \\ m_k = m c_k^{n+1} - \rho D_k \nabla c_k^{n+1} \cdot \boldsymbol{n} & \text{on } \Gamma_e. \end{cases}$$

(viii) Calculate the updated resin mass concentration $c_R^{n+1} = 1 - \sum_k c_k^{n+1}$.

(ix) Determine concentration-dependent rheological parameters: viscosity $\mu(c_R^{n+1})$, surface tension $\gamma(c_R^{n+1})$, and Marangoni forces $\nabla_S\gamma(c_R^{n+1}) = \frac{\partial\gamma}{\partial c_R^{n+1}}\nabla_S c_R^{n+1}$.

(x) Solve the time-dependent Stokes system (4.6) for $\boldsymbol{u}^{n+1}, p^{n+1}$ such that

$$\begin{cases} \frac{\rho}{\Delta t}\boldsymbol{u}^{n+1} - \nabla\cdot(\mu(\nabla\boldsymbol{u}^{n+1} + (\nabla\boldsymbol{u}^{n+1})^T)) + \nabla p^{n+1} = \frac{\rho}{\Delta t}\boldsymbol{u}^n - \nabla\cdot(\rho\boldsymbol{u}\boldsymbol{u})^n + \rho\boldsymbol{g} & \text{in } \Omega \\ \nabla\cdot\boldsymbol{u}^{n+1} = 0 & \text{in } \Omega \\ \boldsymbol{u}^{n+1} = 0 & \text{on } \Gamma_{\text{sub}} \\ [\boldsymbol{u}^{n+1}] = 0 & \text{on } \Gamma_{ij} \\ [\mu(\nabla\boldsymbol{u}^{n+1} + (\nabla\boldsymbol{u}^{n+1})^T)\cdot\boldsymbol{n} - p^{n+1}\boldsymbol{n}] = -\gamma\kappa\boldsymbol{n} + \nabla_S\gamma & \text{on } \Gamma_{ij} \\ \mu(\nabla\boldsymbol{u}^{n+1} + (\nabla\boldsymbol{u}^{n+1})^T)\cdot\boldsymbol{n} - p^{n+1}\boldsymbol{n} = -p_{\text{ext}}\boldsymbol{n} - \gamma\kappa\boldsymbol{n} + \nabla_S\gamma & \text{on } \Gamma_e. \end{cases}$$

4.5.1 Convergence study

We test convergence of the coupled multi-layer coating flow framework via grid-convergence, using a reference solution computed on the finest mesh and smallest time step. In designing a convergence test, complications arise from the Marangoni-driven dynamics. As discussed in the introduction, some surface tension profiles lead to hydrodynamic instabilities, which naturally complicate a grid convergence study. For example, Figure 5.1 in the results section illustrates the evolution of a single-layer coating in 2D with a surface tension value that increases with respect to resin concentration. The corresponding Marangoni-driven evaporative flow produces the short-wave hydrodynamic instabilities discussed in the introduction and perturbations in solutal concentration along the free evaporative surface grow to form Marangoni plumes, or roll-cells. Despite the physical significance of this Marangoni-driven flow regime, it is not suitable for a grid-convergence study as the instability naturally gives rise to grid-dependent dynamics, very similar in character to, e.g., the well-known Kelvin-Helmholtz instability.

It is instead appropriate to examine a regime in which the Marangoni forces play not only a pivotal role, but also one in which stable dynamics ensue. Such flows arise when the surface tension value decreases with respect to a resin concentration, placing the system into a short-wave-stable regime. One could view this as taking the unstable regime, but "flipping the sign" on the Marangoni forcing term within the simulation code. Doing so therefore stress tests each part of the implementation, in a physically-motivated set of parameters, but yields stable dynamics for the purposes of a grid convergence study.

Specifically, we examine convergence for both the single and two-layer coating flow problems in 2D within a $400\mu\text{m}\times100\mu\text{m}$ domain. The specified mesh sizes (e.g., 32×32) represent the number of DG cells of the background mesh within an $L \times L$ block, where $L = 100\mu\text{m}$ is the characteristic length scale in this and all remaining studies. The initial height function values and rheological parameters are as specified in Table B.1 in the appendix and the initial

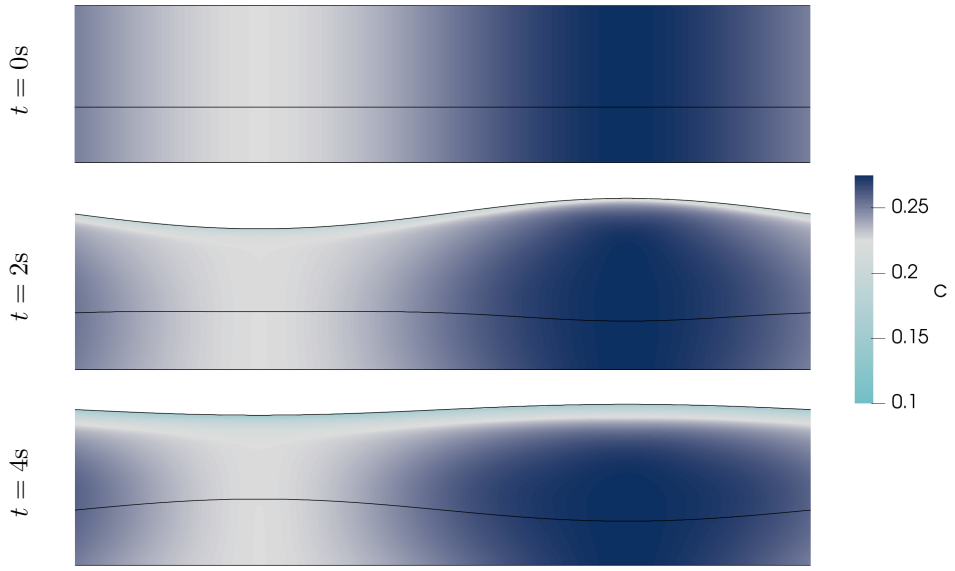


Figure 4.3: The evolution of the solvent mass concentration profile of a 128×128 mesh for the two-layer coating flow convergence test. The Marangoni effect drives flow along the top surface from regions of low surface tension (white) to regions of high tension (dark blue) while evaporation creates a low-solvent boundary layer. Marangoni plumes do not form as in Figure 5.1 since this test is in the short-wave-stable regime. The solvent mass concentration values are indicated by the color bar.

fluid velocity is set to zero. A single solvent is considered with its initial mass concentration profile set to $c = 0.25 + 0.025\cos(\pi x/2L)$ in order to produce smooth Marangoni forces along the free evaporative surface that are resolvable by the coarsest (16×16) mesh. Boundary and jump conditions that are not compatible with the initial conditions (i.e., specifically the Robin and stress conditions) are "slowly turned on" via a ramping method[4]. In this study, we set the time step size small enough so that the spatial errors dominate. Figure 4.3 shows the evolution of the solvent mass concentration profile of the two-layer problem for $T = 5$ seconds, during which fluid is pulled along the free evaporative surface by the Marangoni forces while the evaporation quickly forms a solutal boundary layer, noting a heightened evaporation coefficient ε.

The computed solutions are tested against a reference solution u_{ref} computed on a fine 256×256 ($h = L/2^8$) mesh, with comparison performed using a maximum norm metric. Since the interface locations will differ slightly for different background grid sizes, the error

[4]For example, an interfacial jump condition of the form $[u] = f$ is replaced by $[u] = R(t)f$, where $R(t)$ is a piecewise linear ramping function defined by $R(t) = t/T_R$ when $t \leq T_R$ and $R(t) = 1$ when $t > T_R$. Here T_R is the time at which ramping is complete, equal to 50% of the final time in our convergence tests.

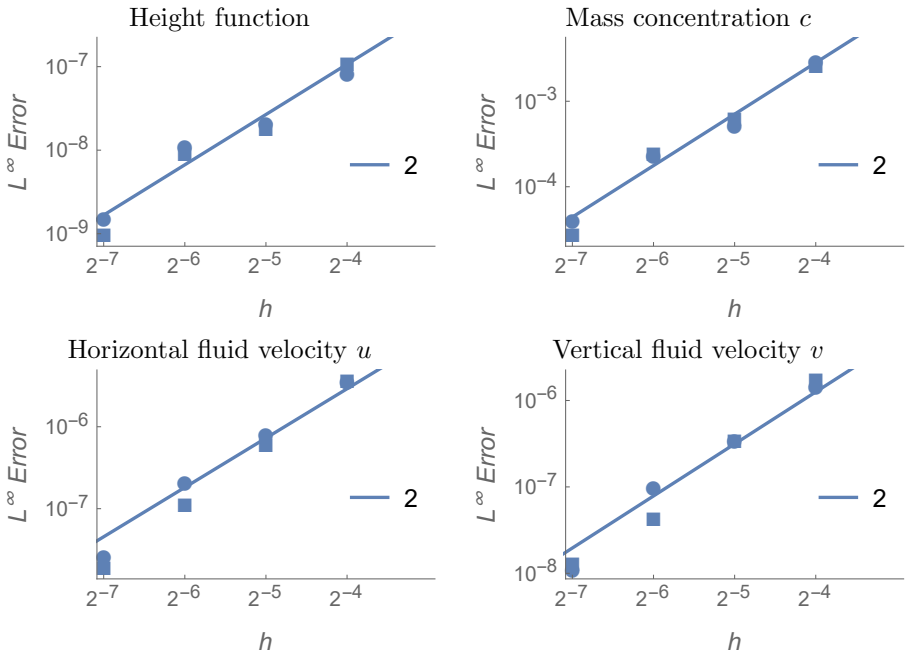


Figure 4.4: The L^∞ errors and convergence rates of the multi-physics coating flow problem with short-wave-stable parameters. Here ●, ■ represent single-layer and two-layer calculations respectively, with the slopes of the lines indicating 2^{nd} order asymptotic convergence rates.

metric for the computed DG polynomial values is calculated according to

$$\max_{t\in(0,T]} \sup_{x\in\bigcup_i(\Omega_i\bigcap\Omega_{\mathrm{ref},i})} ||u-u_{\mathrm{ref}}||_{L^\infty}, \tag{4.14}$$

where $\Omega_{\mathrm{ref},i}$ represents phase i of the reference mesh. This metric calculates the L^∞ errors between points that have the same phase identifier for both meshes and establishes convergence of the DG polynomial solutions when the interface location is convergent. Using this metric, Figure 4.4 demonstrates the results of this convergence test. We observe second-order spatial accuracy in the L^∞ norm for the height function locations, solvent mass concentration profile, and fluid velocity field, for all time $t \in (0, T]$.

Chapter 5

Results

In the previous sections, a mathematical model and several numerical methods are developed for the multi-layer coating flow problem. The framework solves the evolution equations, boundary, and jump conditions of this multi-physics problem and captures the coupling between the evaporative mass transfer system and the multi-phase interfacial quasi-Newtonian fluid dynamics. In this section, several results from the hybrid numerical framework are presented. These include numerical tests of well-known experimentally observed phenomena, specifically:

- Short-wave stationary Marangoni instabilities and the development of Marangoni plumes, which are described mathematically by Pearson [16] and examined experimentally in [19][55].
- 3D two-layer coating flows illustrating the formation of hexagonal-shaped Bénard cells [14][15].
- The long-wave oscillatory deformational modes of Scriven and Sternling [21] and their impact on immersed interfaces; these modes are highlighted in the experimental results of [10][22].

We examine these short- and long-wave Marangoni modes in a 2D parametric study on the flow and leveling of two-layer automobile paint coatings, presented in section 5.2. This study is performed at industrially-relevant conditions and identifies some key features impacting the final surface profile. Additionally, we apply our numerical framework to study several flow regimes not easily assessed through laboratory experiments, exploring:

- Multi-solvent evaporative paint dynamics, specifically single-layer coatings that are composed of multiple solvents each with different evaporation rates or diffusion coefficients.
- A Marangoni-driven drilling phenomena within a multi-layer coating, with links to experimentally observed "cratering" within automotive paint films [89].
- "Interfacial turbulence" within a multi-layer matter cascade.

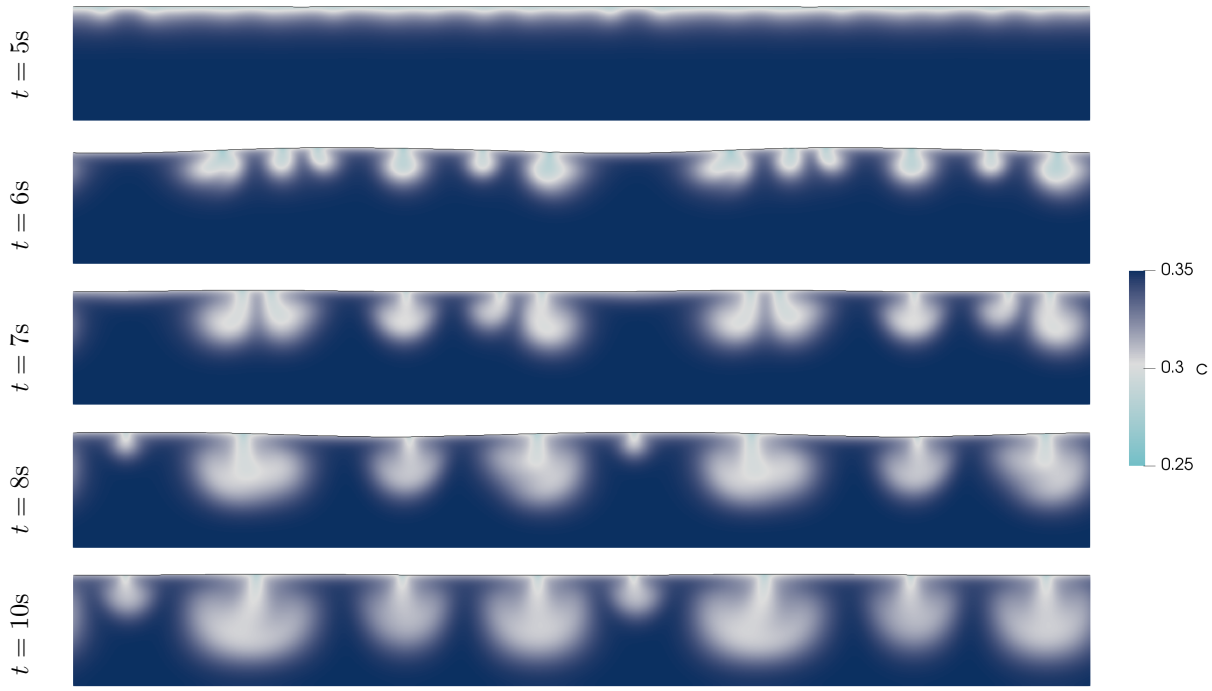


Figure 5.1: The evolution of the solvent mass concentration profile illustrating the formation and merger of Marangoni plumes/roll cells within a single paint film with short-wave-unstable parameters, at the indicated times. The DG background mesh has 32×32 cells per 100μm$\times 100\mu$m block and the 400μm$\times 100\mu$m computational domain is repeated along its periodic axis for presentation. The solvent mass concentration values are indicated by the color bar.

The physical parameters of the liquid paints are chosen to match industrially-relevant conditions and are motivated by experiments on automobile paint coatings. The specific rheological parameters for each study are given in Table B.1 in the appendix. All results assume a uniform initial coating and that the initial solvent mass concentrations are constant in each phase. All embedded paint-paint surfaces Γ_{ij} are initially flat while the initial free evaporative surface Γ_e includes a small perturbation to induce Marangoni flow.

5.1 Short-wave Marangoni instabilities

Figure 5.1 illustrates the evolution of a single-layer coating in 2D. The Marangoni-driven evaporative flow produces the short-wave hydrodynamic instabilities discussed in the introduction and perturbations in solutal concentration along the free evaporative surface grow to form Marangoni plumes/roll-cells. The initial small-scale plumes quickly merge together and coalesce into larger structures, with new plumes continuing to form and merge throughout the evaporative process. This result is consistent with that of [54][55], which classifies this phenomenon as the coarsening of low-order Marangoni roll cells into larger high-order

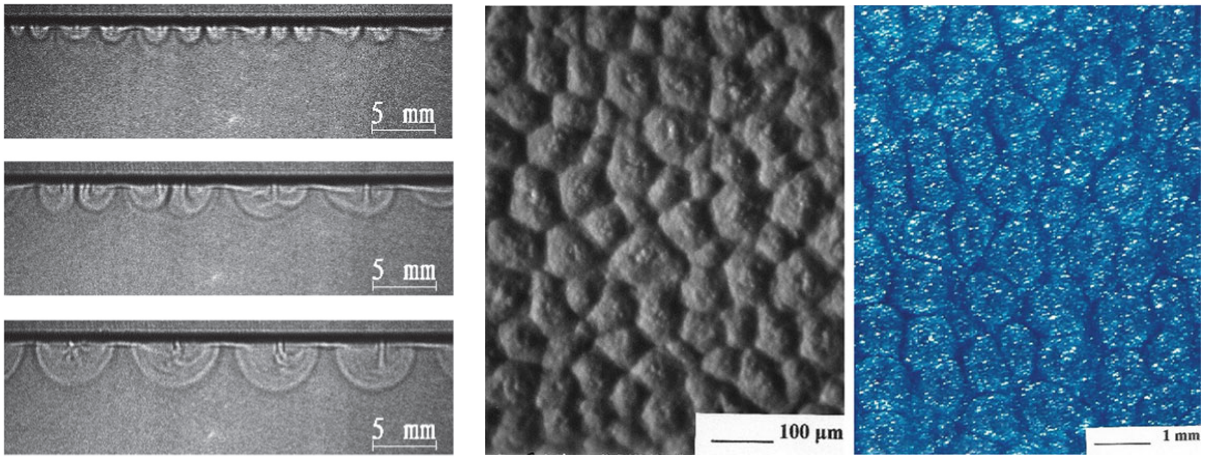


Figure 5.2: **Left**: Shadowgraph imagery illustrating the formation and merger of short-wave Marangoni plumes in an evaporating n-heptane/ether mixture, over 10s. **Center/Right**: Marangoni-induced hexagonal-shaped Bénard cells from an aluminum can coating and an automobile top coat. Images reproduced with permission from [90] and [89] respectively.

cells. Our numerical results are also in good qualitative agreement with the experimental results of Zhong et al. [90], which capture a side profile of the formation and merger of short-wave Marangoni plumes in an evaporating n-heptane/ether mixture through the use of shadowgraph imagery. These images are reproduced in Figure 5.2(left).

5.1.1 3D Results

Figures 5.3 and 5.4 show the solvent mass concentration profile from two 3D simulations on two-layer coatings within a 100μm cube; the clearcoat solvent mass diffusion coefficients are 5×10^{-12} m^2 s^{-1} and 4×10^{-12} m^2 s^{-1} respectively. A notable feature in this result is that the short-wave Marangoni plumes of the 2D simulations become sheets in 3D that are rooted to filaments on the free evaporative surface Γ_e. These filaments merge in a variety of patterns, including a hexagonal diamond pattern within the first few seconds of the simulation and later hexagonal reticulated patterns. Hexagons are a common shape found in Marangoni-driven flows and are often seen in thermally driven Rayleigh-Bénard convection cells [14]. Eventually, the system reaches a "steady-state" arrangement of circulating Bénard cells that impact and deform the basecoat. Here the cells form a square shape; this is, however, likely due to the enforcement of periodic boundary conditions on a small domain. Examples from [89] of hexagonal Bénard cells within paint coatings can be seen in Figure 5.2(center/right). The progression of the 3D Marangoni sheet structures is illustrated in Figs. 5.5 and 5.6, which capture the evolution of solvent mass concentration isocontours (for $c = 0.31$ and $c = 0.29$, resp.) for the 4×10^{-12} m^2 s^{-1} mass diffusion simulation (related to Figure 5.4). The isocontours highlight the Marangoni-induced flow patterns within the clearcoat over the first 20 seconds of the simulation.

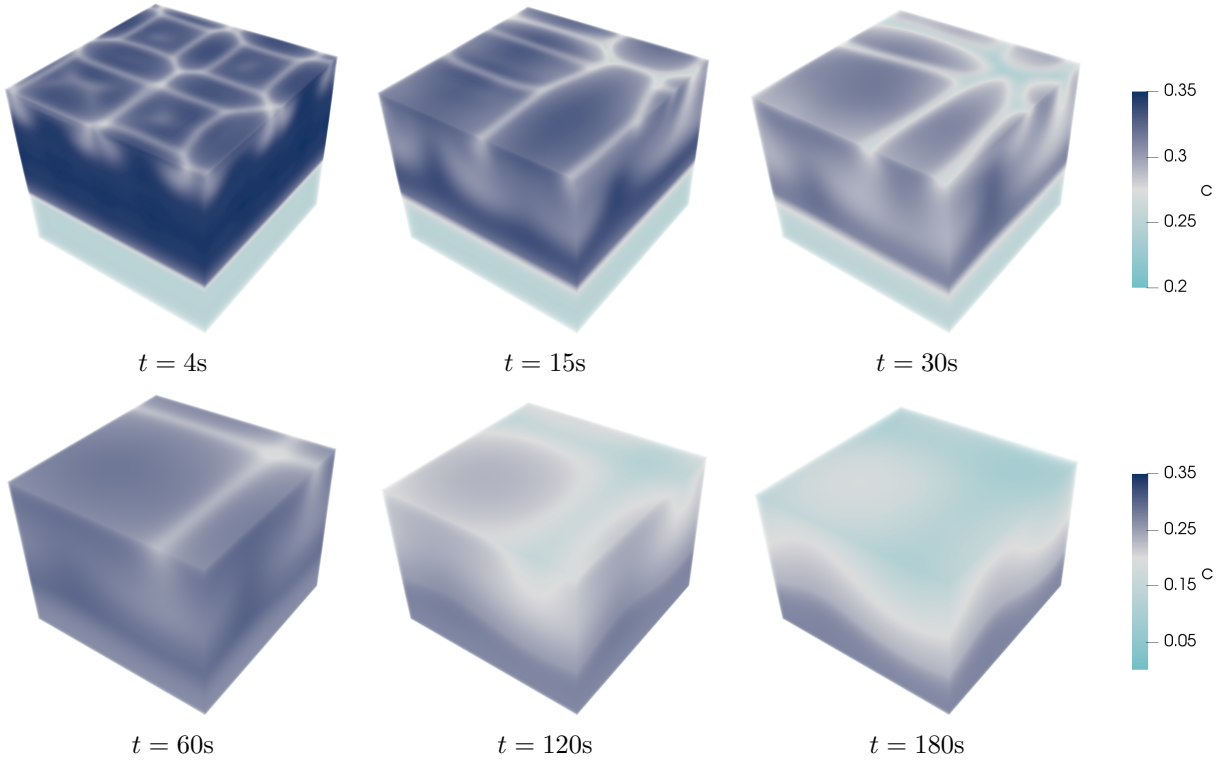


Figure 5.3: Snapshots of the solvent mass concentration profile in 3D with a clearcoat mass diffusion coefficient of 5×10^{-12} m^2 s^{-1}, at the indicated times. Hexagonal-shaped cells form on the top evaporative surface and Marangoni plumes form in a sheet-like structure. The system then develops steady-state Bénard cells. Note the separate color bars in each row specifying the solvent mass concentration.

5.1.2 Multi-solvent evaporation

Typical paints contain multiple solvents, each with their own material properties—density, diffusion coefficient, evaporation rate, etc. To demonstrate the motion and evaporation of coatings with multiple dissolved species, Figure 5.7 shows the mass concentration profile of a single-layer paint composed of three solvents that each evaporate at a different rate. The initial mass concentration is constant and equal for each solvent, the mass diffusion coefficients are set to 5×10^{-12} m^2 s^{-1}, and the ratio of the solvents' evaporation coefficients are 1, 1.5, and 2.3 times a base value of $\varepsilon = 3.33 \times 10^{-4}$ kg m^{-2} s^{-1}. The effect of the different evaporation rates is clear, e.g., with more material ejected in the bottom case, while each solvent has the same dynamical plume structure driven by the overall fluid flow and the short-wave Marangoni instabilities.

Figure 5.8 illustrates the evaporation of a single-layer paint composed of three solvents each with different mass diffusion coefficients. The mass diffusion coefficients are taken to be 1.25×10^{-12} m^2 s^{-1}, 2.5×10^{-12} m^2 s^{-1}, and 5×10^{-12} m^2 s^{-1}, and the evaporation

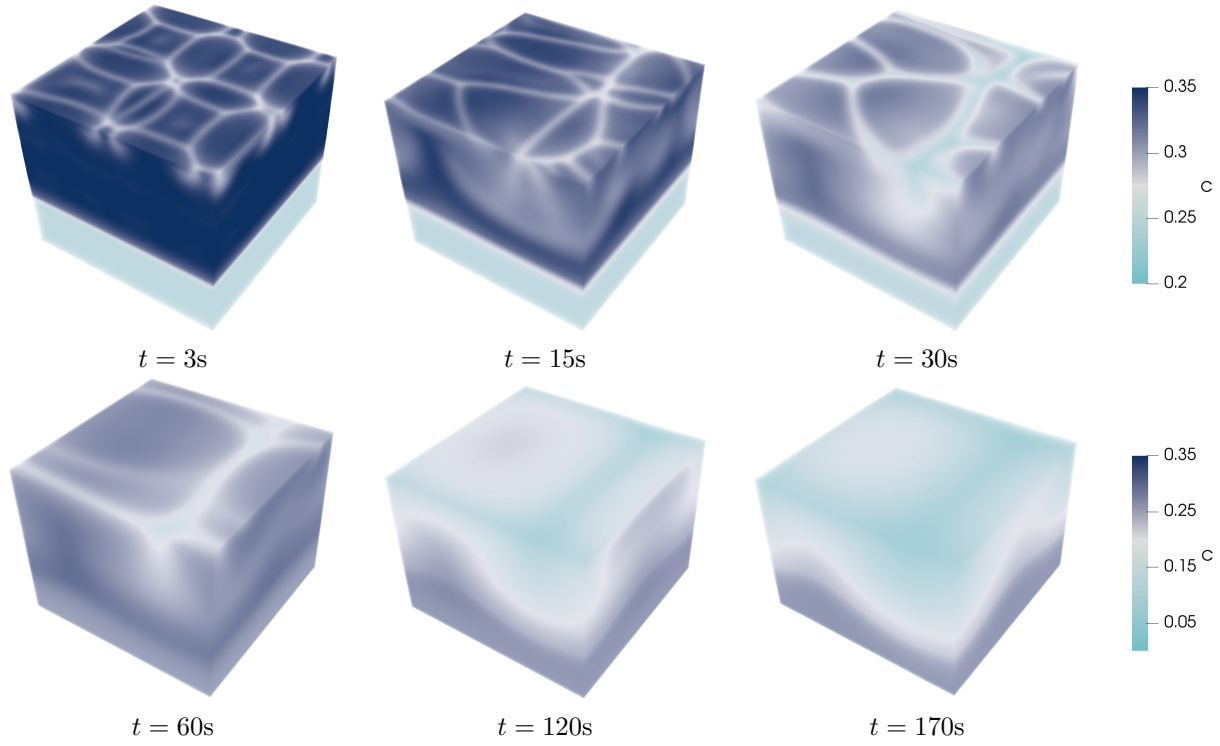


Figure 5.4: Snapshots of the solvent mass concentration profile in 3D with a clearcoat mass diffusion coefficient of 4×10^{-12} m^2 s^{-1}, at the indicated times. Hexagonal-shaped cells again form on the top evaporative surface and more filaments form in this lower diffusion case than in Figure 5.3. Note the separate color bars in each row specifying the solvent mass concentration.

coefficient is set to 3.33×10^{-4} kg m^{-2} s^{-1} for each solvent. This test illustrates the dramatic impact of the diffusion coefficient on the Marangoni plume structures within the coating flow problem. The low diffusion solvent exhibits tighter formations and thinner boundary layers; the vortices are almost "tree" like in nature, with the smaller forming plumes merging to form "branches" of the tree; whereas the higher diffusion solvents have smoothed-out features and thicker boundary layers. In general, larger Marangoni forces are present in low diffusion liquids, owing to the sharper/thinner solutal boundary layers forming at the free evaporative surface. Additional images of low diffusion Marangoni tree structures can be found in Appendix C.

5.2 2D parametric study: long-wave deformational modes in multi-layer automobile coatings

The mathematical model and numerical methods developed in this work are designed, in part, to predict the ultimate surface roughness of multi-layer automobile paint coatings. To this end, we present the results from a high-fidelity 2D parametric study on two-layer coatings

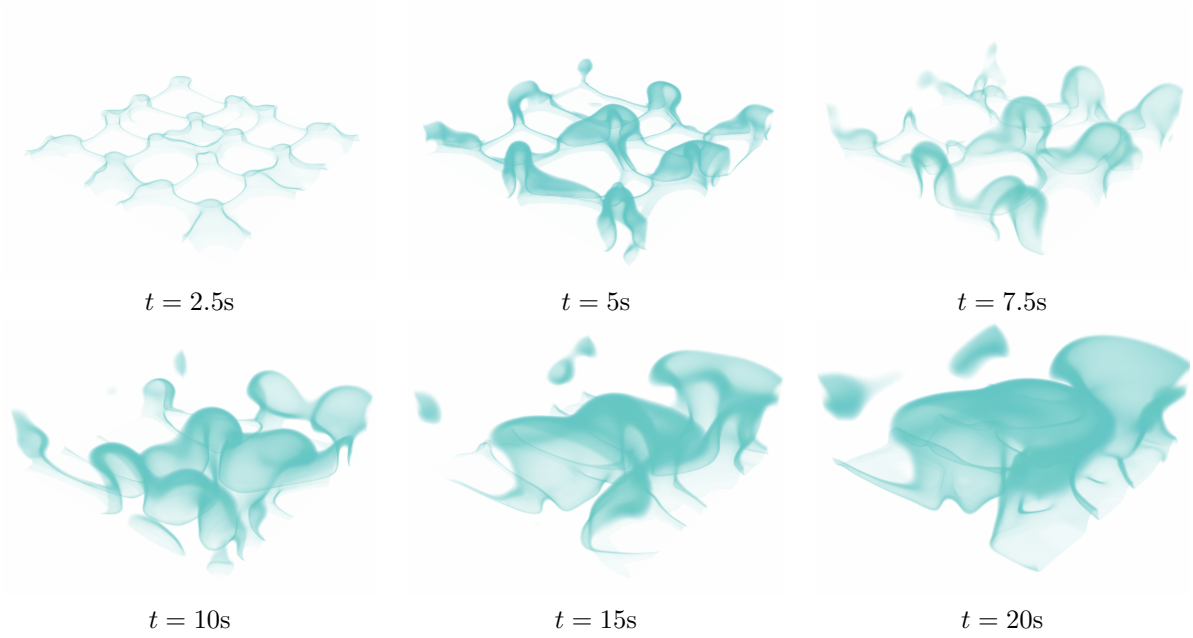


Figure 5.5: The evolution of solvent mass concentration isosurfaces for $c = 0.31$, illustrating the Marangoni-driven fluid flow within the bulk of the clearcoat, at the indicated times. For illustration, the z-axis is inverted so that the base of the figures is the free evaporative surface Γ_e.

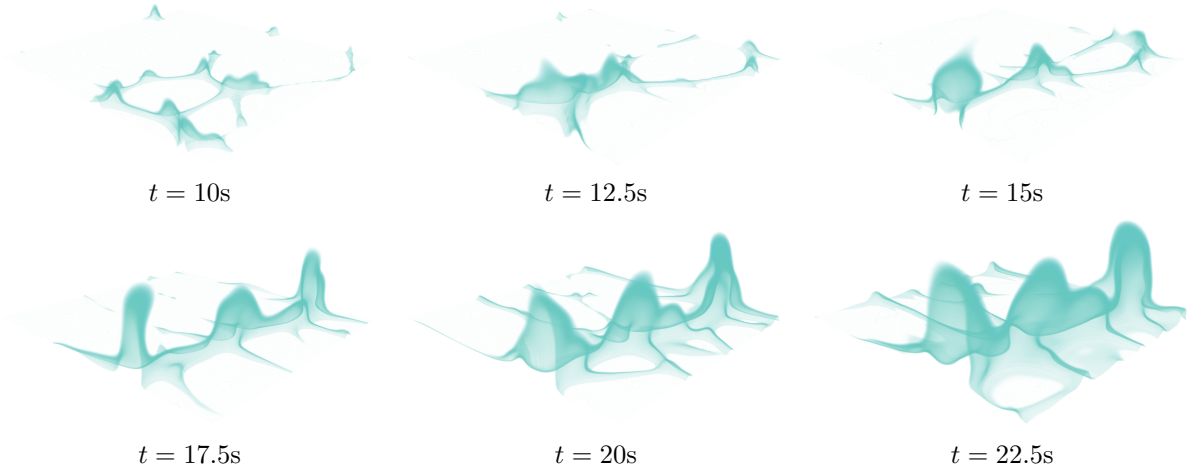


Figure 5.6: The evolution of solvent mass concentration isosurfaces for $c = 0.29$, illustrating the Marangoni-driven fluid flow within the bulk of the clearcoat, at the indicated times. For illustration, the z-axis is inverted so that the base of the figures is the free evaporative surface Γ_e.

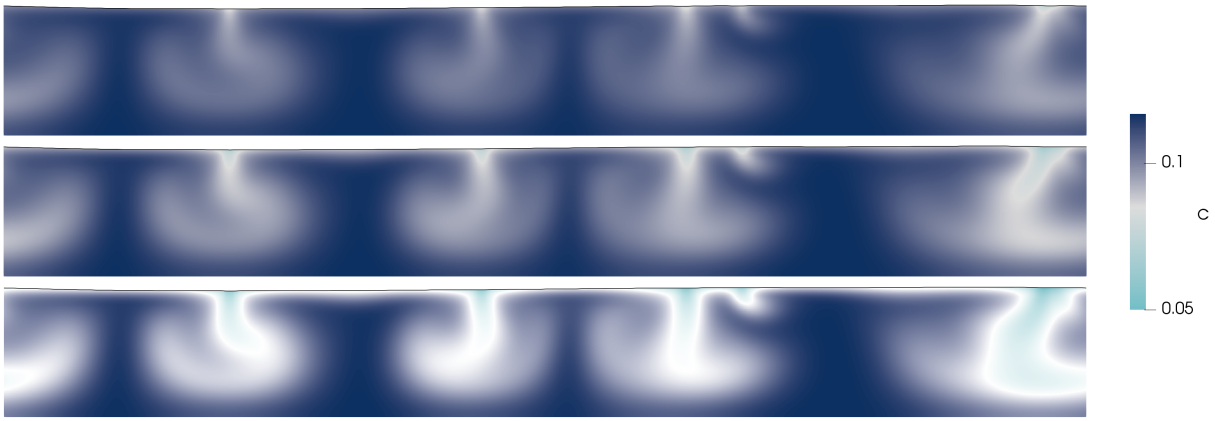


Figure 5.7: The solvent mass concentration profile of three solvents evaporating at different rates. The solvent evaporation rate (4.3) has **Top:** 1, **Middle:** 1.5, **Bottom:** 2.3 times a base evaporation coefficient ε of 3.3×10^{-4} kg m^{-2} s^{-1}. These results are taken 20 seconds into the simulation.

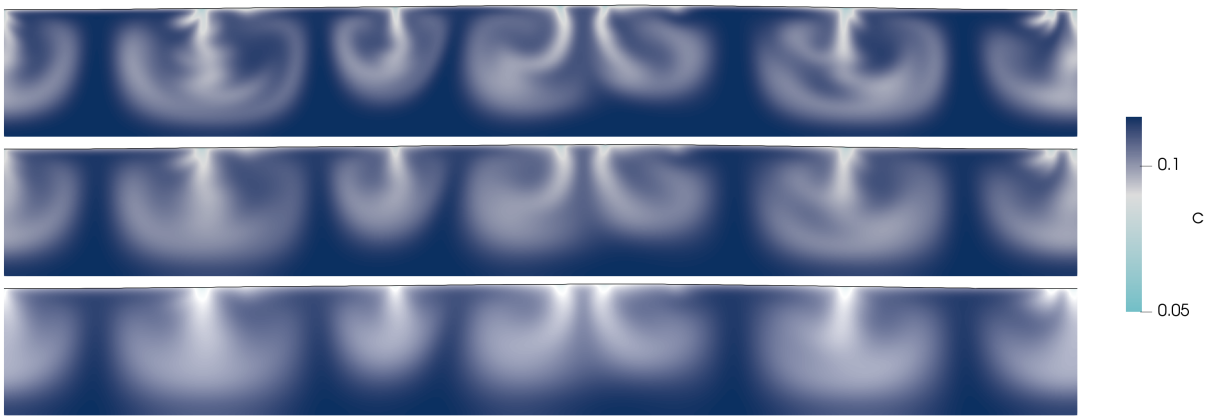


Figure 5.8: The solvent mass concentration profile of three solvents with different mass diffusion coefficients. The coefficients are **Top:** 1.25×10^{-12} m^2 s^{-1}, **Middle:** 2.5×10^{-12} m^2 s^{-1}, **Bottom:** 5×10^{-12} m^2 s^{-1}. These results are taken 20 seconds into the simulation.

that identifies some key features impacting the final surface profile. There is a range of physics and many possible parameters to study within the multi-layer coating flow problem. We identified a few key physical parameters and regimes to study, these include: the strength of embedded paint-paint surface tension, the values of clearcoat viscosity, the substrate profile as well as its direction against the pull of gravity, and the effect of different Marangoni forces. The results present a collage of the impact of these different physical effects; an in-depth study of individual parameters would also be possible. These studies introduce a notion of paint compatibility, demonstrate the telegraphing of substrate roughness, and illustrate the impact of the Marangoni effect and long-wave surface modes on the final paint surface smoothness.

To properly capture the long-wave surface modes found along dried paint films, the wavelengths of which range from $1 - 10$mm, the 2D simulations in this section are performed on long-skinny domains with a horizontal length of 25.6mm and an aspect ratio of 256×1. Each 100μm$\times 100\mu$m block of the domain is discretized by 32×32 background cells of the implicit mesh DG method—leading to computations with 250,000-400,000 elements and 9 to 14 million spatial degrees of freedom.

To eliminate variables and simplify our study, only single-solvent paints are considered and the following physical parameters are fixed throughout the parametric study unless otherwise specified:

(i) The fluid density and initial resin mass concentrations are as stated in Appendix B.

(ii) The basecoat mass diffusion coefficient is taken to be 1×10^{-12} m^2 s^{-1} and the clearcoat mass diffusion coefficient is 5×10^{-12} m^2 s^{-1}. We note that the mass diffusion coefficients for different materials can range over four orders of magnitude, anywhere from $10^{-14} - 10^{-10}$ m^2 s^{-1}; our choice of mass diffusion coefficients puts the simulations in the middle of this range. We conjectured this was of main physical relevance to the objectives of the parametric study; in addition, as shown in Figure 5.8, this choice of clearcoat mass diffusion provides smooth Marangoni dynamics.

(iii) The basecoat viscosity is taken from the high range of the PPG-provided viscosity values.

(iv) The evaporation coefficient is taken to be $\varepsilon = 3.33 \times 10^{-4}$ kg m^{-2} s^{-1}.

(v) Additionally, we note that since little is known about the values of surface tension of the embedded paint-paint surface, the Marangoni effect is not considered along Γ_{ij} in this section.

5.2.1 Embedded paint-paint surface tension studies

We begin our parametric study by examining the effect of different values of surface tension along the embedded paint-paint surface Γ_{ij}. Since this is one of the key couplings between the two paint layers, and since the exact values of embedded paint-paint surface tension coefficients for specific paints are in general unknown, it is a natural starting choice for our parametric study.

Study 1 — Very low embedded surface tension

We begin our study of embedded surface tension on the low end of the expected range of coefficient values, setting $\gamma_{ij} = 0.0015$ N m^{-1}. We take the clearcoat viscosity to be the PPG provided experimental data and employ a flat, horizontal substrate $\mathcal{S}_{\text{flat}}$. We note that the linear surface tension profile used in this test along the free evaporative surface Γ_e leads to relatively weak Marangoni forces. Figure 5.9 shows a close-up view of the solvent dynamics

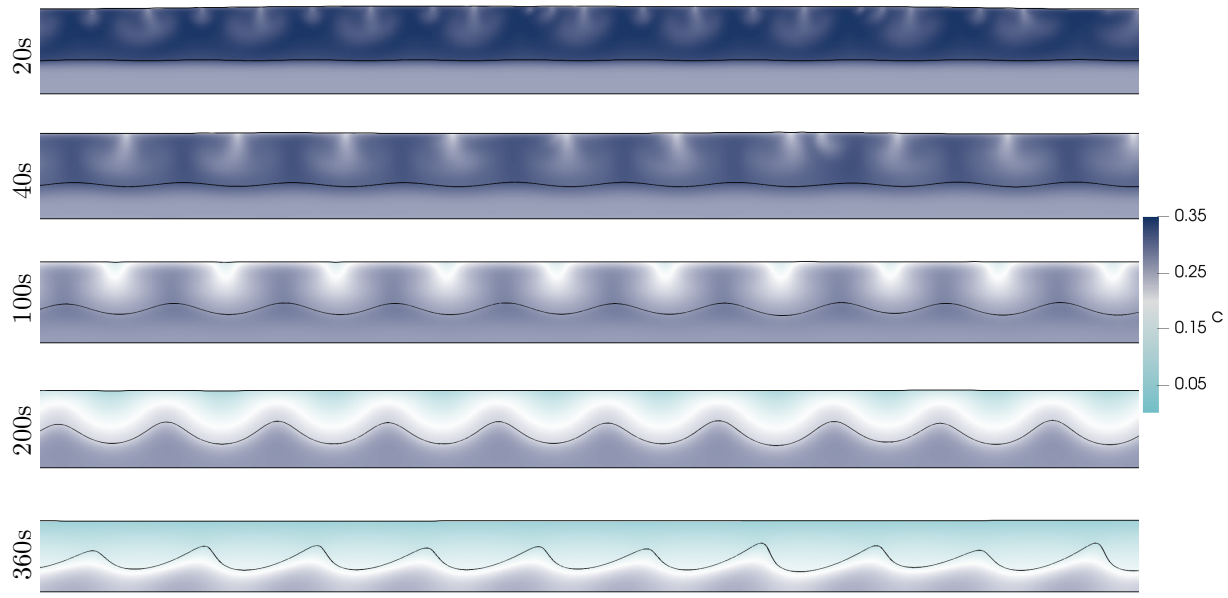


Figure 5.9: Study 1 - 1mm slices of the solvent mass concentration profile at the indicated times—providing a closer view of the Marangoni plumes and basecoat deformation.

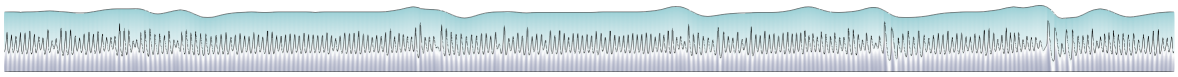


Figure 5.10: Study 1 - The solvent mass concentration profile at time 360s of the entire domain with the horizontal x-direction scaled by 1/20. This demonstrates the full extent of the basecoat deformation pattern.

within a 1mm slice, illustrating the progression of the short-wave Marangoni plumes towards steady-state Bénard cells and highlighting their uniformity throughout the clearcoat. During the initial stages of drying, the Marangoni effect drives the formation, merger, and coarsening of plumes of various sizes, as seen in Figure 5.1. Over time, the plumes reach the bottom of the clearcoat and larger individual circulation cells form. The geometry and position of these cells are approximately steady and the cells have a width on the order of the film height, in agreement with the experiments of [10]. Fluid flow persists within these cells and the circulation helps replenish the evaporative process. The circulating Bénard cells then act to deform the basecoat, both pushing downwards and pulling upwards on the embedded paint-paint surface Γ_{ij}, which quickly gives way due to its weak surface tension. The wavelengths imparted on the basecoat are about equal to the width of the Marangoni plumes—around 100μm. Figure 5.10 shows the entirety of the domain with the horizontal x-direction scaled by a factor of 1/20—where the sharpness of the deformed basecoat is evident.

Study 2 — Low embedded surface tension

The second study of embedded surface tension uses a potential formula found at [91]. The

formula utilizes each paint's surface tension coefficient and is

$$\gamma_{ij} = \gamma_{BC} + \gamma_{CC} - 2(\gamma_{BC}^d \gamma_{CC}^d)^{1/2},$$

where γ_{BC} is the coefficient of surface tension between the basecoat and air and γ_{CC} is similar for the clearcoat. Here, the superscript d indicates the component of surface tension caused by dispersion—the values of which are unknown for the paints. Instead the formula

$$\gamma_{ij} = \gamma_{BC} + \gamma_{CC} - 2a(\gamma_{BC}\gamma_{CC})^{1/2}$$

is used, where a is a multiplier that approximates the dispersion. In this study, we take the clearcoat surface tension to be the PPG standard value and the basecoat's to be the PPG low value. With $a = 0.9$, this gives an embedded surface tension value around $\gamma_{CC}/5$, which provides a range on γ_{ij} of 0.04 - 0.08 N m^{-1}. However, we note that the Marangoni stresses along Γ_{ij} are fixed to zero in this test.

Figure 5.11(top) shows the evolution of the solvent mass concentration within a $1.6mm$ window of a two-layer coating. This test highlights the coupling between the paint films within the multi-layer system and captures an irregularity not seen in the single-layer case. Around 45 seconds into the simulation, two large Marangoni plumes merge together and drill into the basecoat, causing a wide deformation that raises the basecoat and drags down the clearcoat along the edges of the plume. About 90 seconds in, the embedded paint-paint surface and the free evaporative surface are close to intersecting such that the clearcoat recedes and exposes the basecoat to air. In industrial operations and laboratory experiments, multi-layer paints sometimes develop holes while drying. A conjecture is that these holes form when the interfaces Γ_e and Γ_{ij} intersect, exposing the basecoats to air, and the recession progresses further such that substrate is exposed and the system undergoes dewetting. The drilling phenomena may be the first phase of this process. Additionally, this result is reminiscent of the "cratering" irregularity that sometimes occurs in automotive paint coatings [89], i.e., the rapid collapse of a coating caused by the Marangoni effect in the presence of a low surface tension impurity.

We now examine the fluid velocity field within the multi-layer coating flow problem. Figure 5.11(middle) shows the vertical component of the fluid velocity, where the presence of the Marangoni plumes corresponds with a downward (blue) motion of the fluid. The downward motion is matched by an upward (red) current adjacent to the plume. The eventual formation of steady-state Bénard cells can be seen through the progression of this figure. Note the basecoat has a higher viscosity than the clearcoat and subsequently its fluid velocity is much smaller. However, a subtle note is that at time 90s the red upward velocity extends more prominently into the basecoat at the deformed drilled component of the paint-paint surface than at the other steady-state cells, indicating an increased motion of the interface at that region. In the horizontal direction, the Marangoni forces pull fluid towards the plume, providing regions near the evaporative surface of leftwards (blue) and rightwards (red) motion,

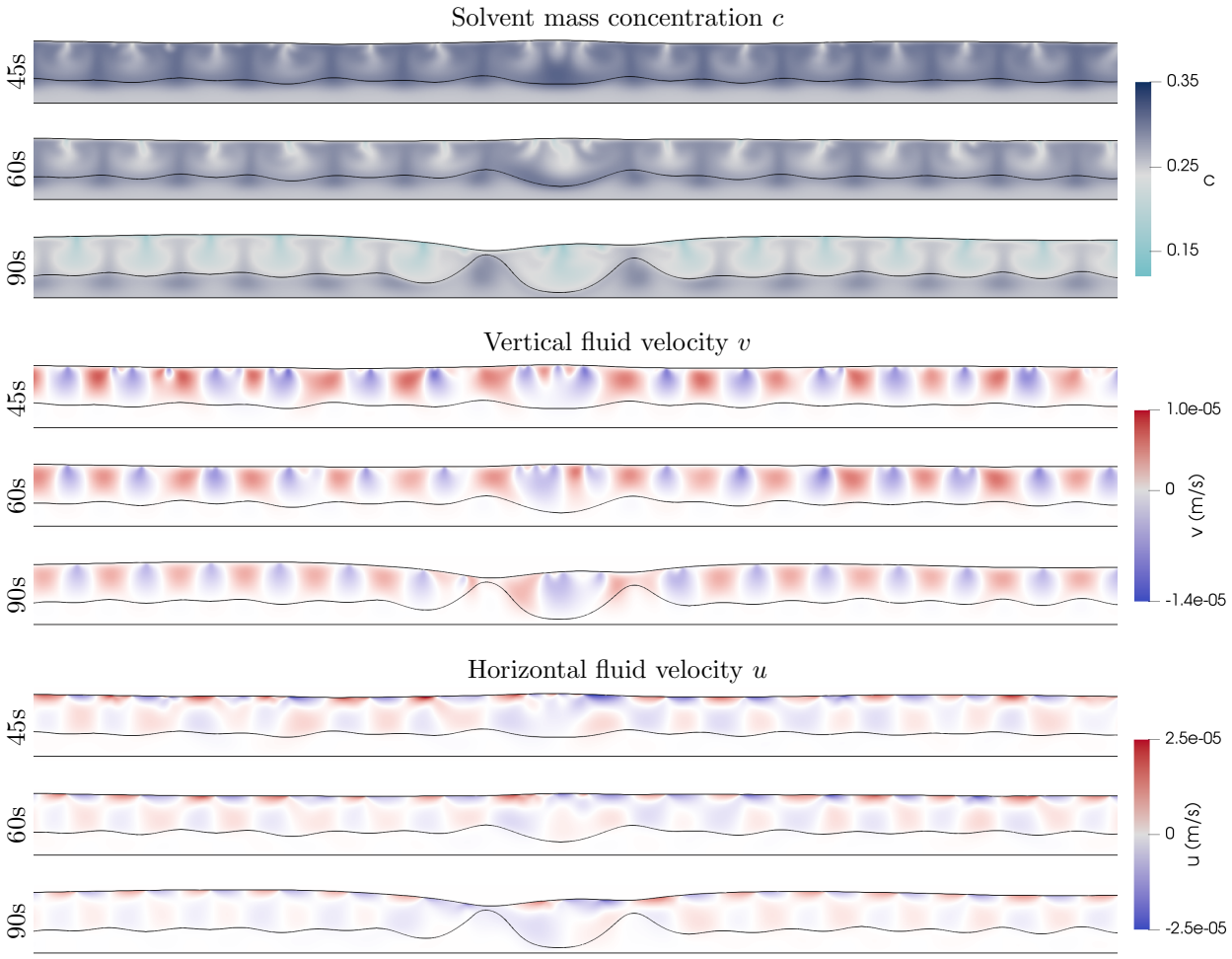


Figure 5.11: Study 2 - short-wave instability causing deformation of the basecoat - **Top:** A 1.6mm window of the solvent mass concentration profile at the indicated times. Here two large Marangoni plumes (near center of images) merge together and drill into the basecoat—setting the two surfaces on a collision course. **Middle:** The vertical component of the fluid velocity field within the same window at the same times. Blue represents a downward negative velocity and red an upwards positive velocity. **Bottom:** The horizontal component of the velocity field, which has an almost lattice structure within the clearcoat. The blue indicates a leftwards velocity and the red rightwards. The values of the solvent mass concentration and fluid velocity are indicated by the color bars.

the direct intersection of which results in the vortex. This, along with an almost lattice structure of the horizontal velocity component, is seen in Figure 5.11(bottom).

Study 3 — High embedded surface tension

For the final study examining the embedded paint-paint surface tension, we set the values of the surface tension for the free evaporative surface Γ_e and embedded paint-paint surface

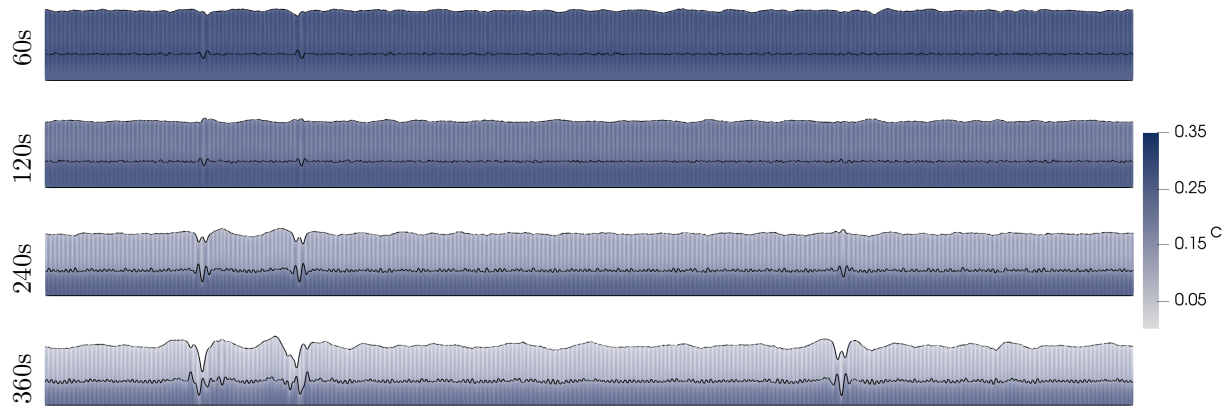


Figure 5.12: Study 3 - Long-wave instability causing deformation. Illustrated is the solvent mass concentration profile at the indicated times for a 25.6mm×0.1mm domain with the horizontal x-direction scaled by 1/20. The left side of the images illustrates the oscillatory nature of paint drying, with peaks becoming troughs, and vice versa. Here, the long-wave Marangoni instabilities grow and compound, and eventually the paint layers tear.

Γ_{ij} to equal the PPG standard value, i.e., $\gamma_{ij} = \gamma$. All other parameters are as in Study 2. The higher surface tension provides greater resistance to basecoat deformation caused by the short-wave Marangoni modes than in the previous studies, however as shown in Figure 5.12, long-wave Marangoni modes also imprint on the paint films. This figure shows the evolution of the solvent mass concentration profile along the entire 25.6mm×0.1mm domain, with the horizontal x-axis scaled by 1/20. Around 60 seconds into the simulation, long-wave surface modes become noticeable along the free evaporative surface. As described mathematically in [21][23] and captured experimentally in [10][22], these modes deform the evaporative surface and are oscillatory, e.g., the troughs on the left side of the figure at $t = 60$ seconds become peaks at around 120 seconds. Fourier frequency data (see Figure 5.15) shows that, in this regime of coating flow dynamics, the peak amplitude wavelength is roughly 1mm and that the oscillations grow in amplitude each cycle. The Fourier data also shows an impression of the long-wave frequencies of the paint-gas surface onto the paint-paint surface.

The embedded paint-paint surface tension studies have all shown Marangoni forces push and pull on the paint-paint surface Γ_{ij} and deform the basecoat, leading to misshaped clearcoats and a rough surface profile. One possibility is that low embedded surface tension and the resulting overall lack of force between the film-layers allows the paints to be malleable to any external force, including the intrinsic Marangoni forces. It is clear that increasing the coefficient of embedded surface tension will help to keep the paint-paint surface flat—indicating that paints that impart a higher force on each other and work to mitigate, or provide resistance to, the Marangoni forces are more likely to be "compatible" and perhaps paints which tear apart do so due to an incompatibility.

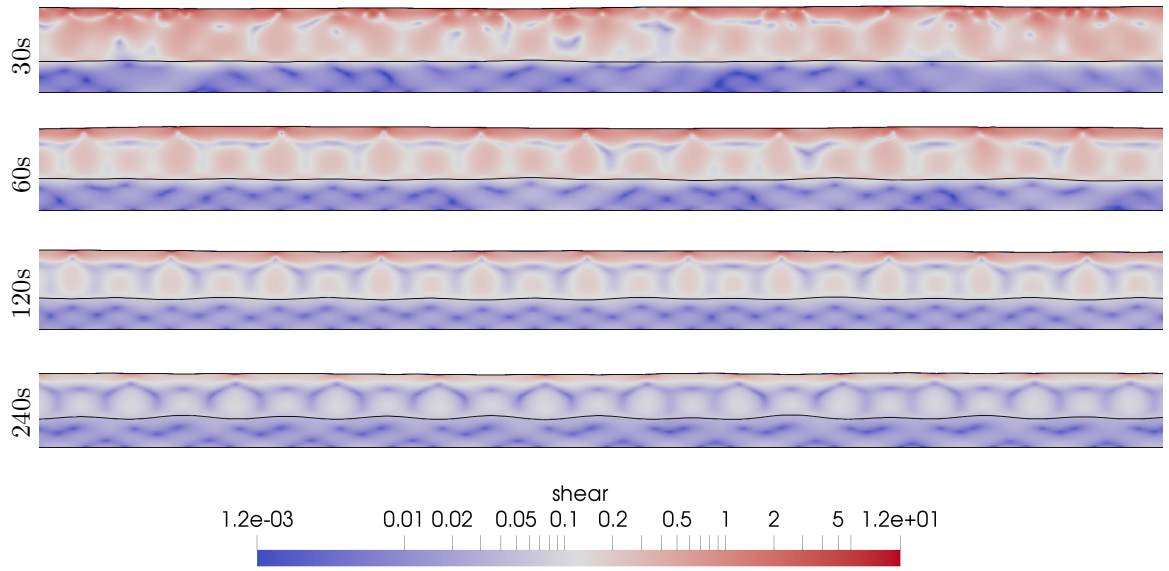


Figure 5.13: Study 3 - The shear rates (s^{-1}) within a 1mm slice. Note the logarithmic scale.

A note on shear rates

Typical liquid paints are non-Newtonian shear-thinning fluids, meaning the effective viscosity decreases under shear strain. Figure 5.13 examines the shear rates in the present study, corresponding to a 1mm slice at multiple time stamps; note the logarithmic scale. Rates up to 20 s^{-1} are observed during the chaotic initial phase of Marangoni plume formation. The majority of the shearing occurs within the solutal concentration boundary layer at the evaporative surface, with the highest values at the base of the Marangoni plumes. The shear rates within the remainder of the paint are several orders of magnitude lower and, overall, the rates decrease over time as the paints dry.

5.2.2 Substrate profile

Study 4 — Bumpy substrate

The previous studies all used a completely flat substrate $\mathcal{S}_{\text{flat}}$. In practice, the substrate will have a microscopic bumpy profile. In the numerical framework, this can be modeled easily by changing the stationary height function that defines Γ_{sub}. This study uses the same parameters as Study 3 and uses the PPG-provided smooth horizontal substrate $\mathcal{S}_{\text{smooth}}$(described in Appendix B). We note that this and all remaining studies use the high value of embedded surface tension. Figure 5.14 shows that this new slightly bumpy substrate profile quickly provides perturbations in the basecoat—and subsequently the clearcoat. As in the previous study, these perturbations build the Marangoni instabilities which lead to a rougher paint profile.

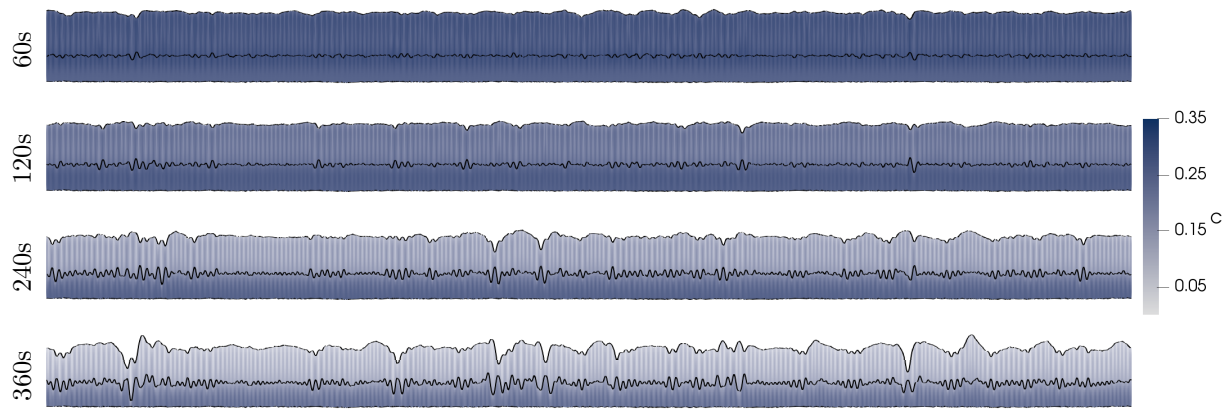


Figure 5.14: Study 4 - The solvent mass concentration profile on the PPG provided smooth substrate $\mathcal{S}_{\text{smooth}}$, at the indicated times. The entire domain is shown with the horizontal x-direction scaled by 1/20. The slightly bumpy nature of the smooth substrate quickly impacts the basecoat and clearcoat—resulting in higher wave amplitudes than in the flat substrate case of Study 3 and in Figure 5.12.

For a more detailed comparison, the exact interface roughness profile of both the free evaporative surface Γ_e and the embedded paint-paint surface Γ_{ij} is analyzed via Fourier transforms for both this and the previous study, the results of which are shown in Figure 5.15. At each time stamp, the modes of the evaporative surface have a larger amplitude in the case of the smooth substrate $\mathcal{S}_{\text{smooth}}$when compared to the flat substrate $\mathcal{S}_{\text{flat}}$. Both support roughly the same wavelengths, and the frequency profile is Gaussian-shaped in logarithmic frequency space. The peak-amplitude wavelength hovers around 1mm and the half-width half-max of the Gaussian is roughly at 0.5 and 1.5mm. The amplitudes steadily increase over time, consistent with the idea that the long-wave Marangoni modes grow in size each oscillation cycle. At the early stages of the simulation, the paint-paint surface supports a range of shorter wavelengths when compared to the evaporative surface. The frequencies are again in the shape of a Gaussian with the prominent mode shifting from 250μm to 150μm over time as the surface is further impacted by the Bénard cells. At later times, modes around 1mm appear as the modes of the evaporative surface and paint-paint surface couple together. As seen with the modes of the evaporative surface, the modes of the paint-paint surface have a larger amplitude in the case of the smooth substrate when compared to the flat substrate at all time stamps. Note the outlier at 100μm on the paint-paint surface of the completely flat run is likely due to an imprinting of the evaporative surface's initial condition, which contains this mode. This mode is damped out in the case of smooth substrate $\mathcal{S}_{\text{smooth}}$, perhaps further indicating the telegraphing of substrate profile onto the paints.

Study 5 — Vertical substrate

Under certain conditions, vertical substrates, for which the pull of gravity is parallel with

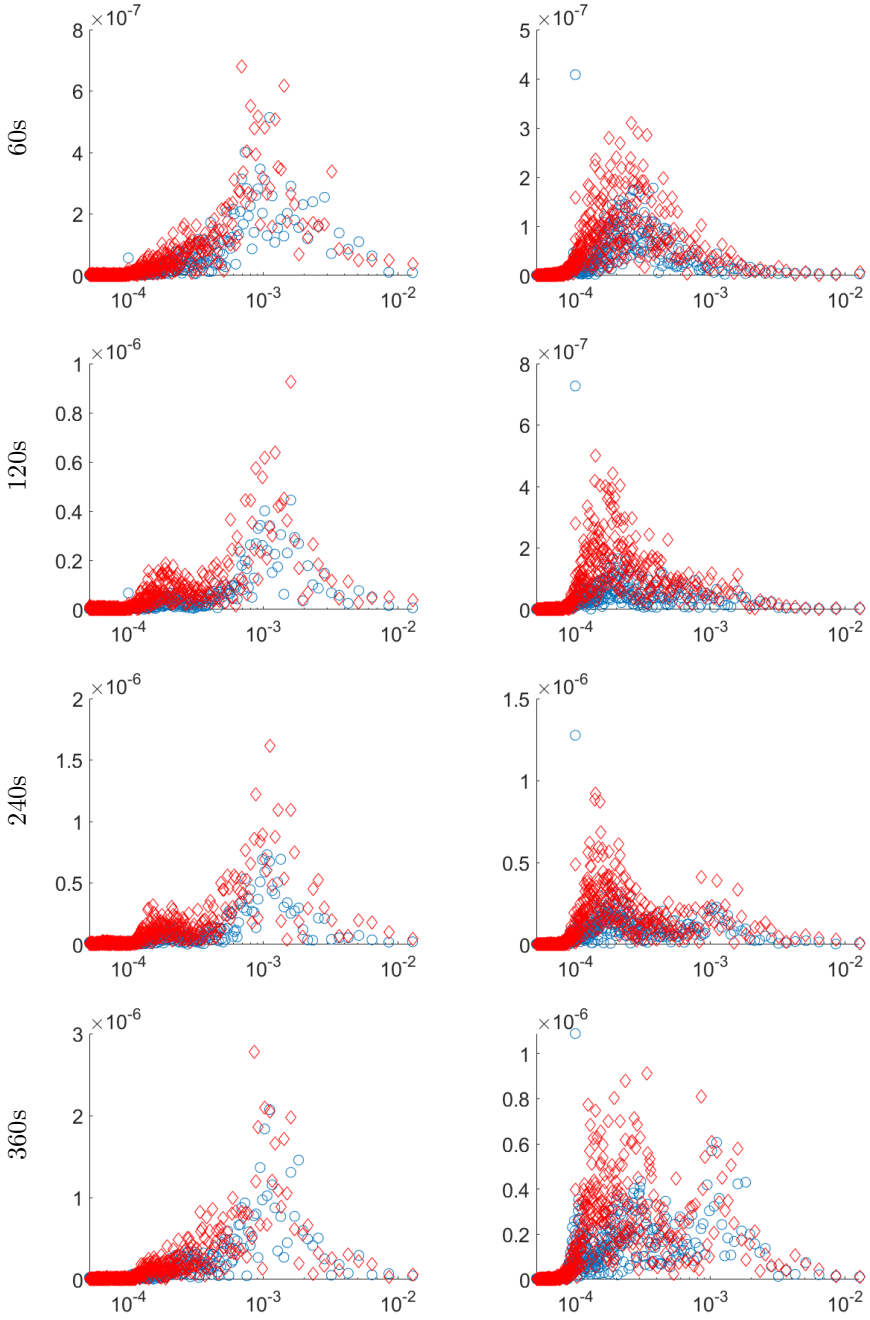


Figure 5.15: Fourier frequency data for the free evaporative surface Γ_e(left) and embedded paint-paint surface Γ_{ij}(right) for cases of the flat substrate $\mathcal{S}_{\text{flat}}$(blue circles) and the PPG smooth substrate $\mathcal{S}_{\text{smooth}}$(red diamonds), corresponding to Studies 3 and 4. The horizontal axis corresponds to the wavelength in meters, noting the logarithmic nature of the axis, and the vertical axis represents the wavelength's amplitude in meters.

the film-layers, produce longer wavelengths along the paint surfaces (compared to a horizontal substrate) as well as clearly-observable effects such as sagging. However, under the parameter and flow regimes of the present parametric study, we did not observe these effects. For numerically-simulated vertical substrates (including rough substrate $\mathcal{S}_{\text{rough}}$), the pull of gravity provided a slight drift of the paint profile but the dominant behavior was still the instabilities caused by the Marangoni forces, as witnessed in Studies 1–4.

5.2.3 Clearcoat viscosity

Study 6 — High viscosity

As a possible means to mitigate the Marangoni instabilities, we examine next the effect of clearcoat viscosity on the system. Viscosity acts to dampen momentum, providing friction within the bulk of the liquid paint. Therefore, a higher value of viscosity could potentially damp out the strength of the Marangoni modes and provide flatter interfaces. This study matches all other values of Study 3 except the clearcoat viscosity is taken to be from the PPG high range. Figure 5.16 shows that this change slightly dampens the effect of the strong Marangoni forces and this simulation is the first multi-layer coating to entirely dry, i.e., all solvents evaporate. The final simulation time in this study is 45 minutes. Here, the long-wave oscillatory Marangoni modes deform the paint-gas and paint-paint surfaces during the first 10 minutes of drying and these early deformations leave a lasting imprint on the final paint profile—with moderate smoothing and settling over time. This test demonstrates the utility of the adaptive time stepping technique of section 4.4, where the time step size is changed based on the ratio of viscosity to surface tension. In this particular problem, the time step size is nearly 400 times larger at the end of the simulation than at the beginning.

Study 7 — Low viscosity

On the opposite side of the spectrum, we tested a clearcoat with the PPG low viscosity. As expected, the low viscosity did little to stop the Marangoni instabilities and this run quickly terminated at 160s when the two paint surfaces collided with each other.

5.2.4 Additional studies

A few other ideas to weaken the Marangoni instabilities were also tested. The parameters of these studies include an experimental clearcoat viscosity, standard surface tension, and a flat vertical substrate $\mathcal{S}_{\text{flat}}$. In the first test, the evaporation rate is decreased by half—increasing the time frame for solvent diffusion to smooth out the gradients causing Marangoni forces. However, this did not produce the desired effect. Neither did the second idea of increasing the basecoat diffusion coefficient to match that of the clearcoat (5×10^{-12} m^2 s^{-1}), the idea being that this will make the two paints more compatible since solvents can be exchanged more easily between the two film-layers. Both of these tests are met by the same instability and deformation patterns seen in the previous studies.

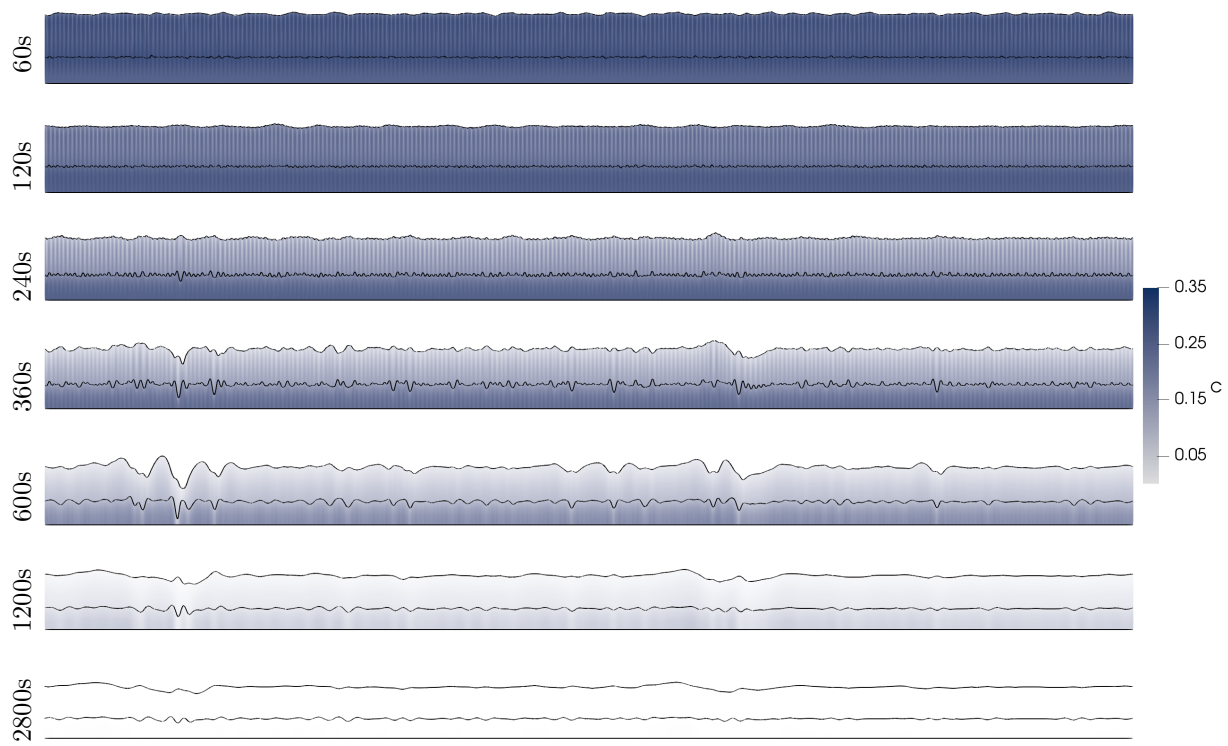


Figure 5.16: Study 6 - A two-layer paint drying in entirety. Illustrated is the solvent mass concentration profile at the indicated times for a 25.6mm×0.1mm domain with the horizontal x-direction scaled by 1/20. The deformations from the Marangoni effect leave an imprint on the final paint surface.

Study 8 — Lowered Marangoni forces

The simplest idea to mitigate the instabilities caused by the Marangoni effect is to decrease the strength of the Marangoni forces themselves—in this study, by a factor of a half. The results are shown in Figure 5.17. With the weakened Marangoni forces, the interfaces remain mostly flat throughout the entirety of the simulation, which is allowed to run for 45 minutes. There is little to no deformation of the basecoat or clearcoat. This study provides a clear insight that clearcoat paints used in multi-layer coatings should target a weak Marangoni profile.

5.3 Interfacial turbulence

In [18], Sternling and Scriven propose the Marangoni effect as a mechanism for producing interfacial turbulence, i.e., the spontaneous agitation of the interface between two unequilibrated liquids. Their work classifies the various regimes of stability and shows that mass

Figure 5.17: Study 8 - The solvent concentration profile with Marangoni forces halved, at the indicated times. The entire domain is shown with the horizontal x-direction scaled by a factor of 1/20. This result shows that decreasing the strength of the Marangoni forces helps maintain a flat paint profile.

flowing from a fluid of (i) higher viscosity and (ii) lower mass diffusivity across an interface whose surface tension (iii) decreases with respect to the mass concentration produces unstable fluid flow and may produce localized stirring and even droplets without any chemical reactions [19]. These parameters describe the short-wave-unstable evaporating Marangoni flows of the previous results and, in this section, the hybrid numerical framework is used to model interfacial turbulence within a three-layer matter cascade. Here, a species originating in the bottom layer flows upwards, passing through multiple interfaces in a Marangoni short-wave-unstable fashion.

The first example is shown in Figure 5.18. At $t = 0$, the bottom layer contains a species with a mass concentration of $c = 0.2$, while the other two layers do not contain the species. The physical parameters are as in Table B.1 and each subsequent layer has half the viscosity and double the mass diffusion coefficient and thickness of the previous layer. For these tests, evaporation is disabled at the top free surface. By 25 seconds into the simulation, the species has transferred from the bottom layer into the middle layer and the Marangoni effect produces the familiar short-wave plume structures in both phases. The species then reaches the next interface between the middle and top layers at around 50 seconds. Here,

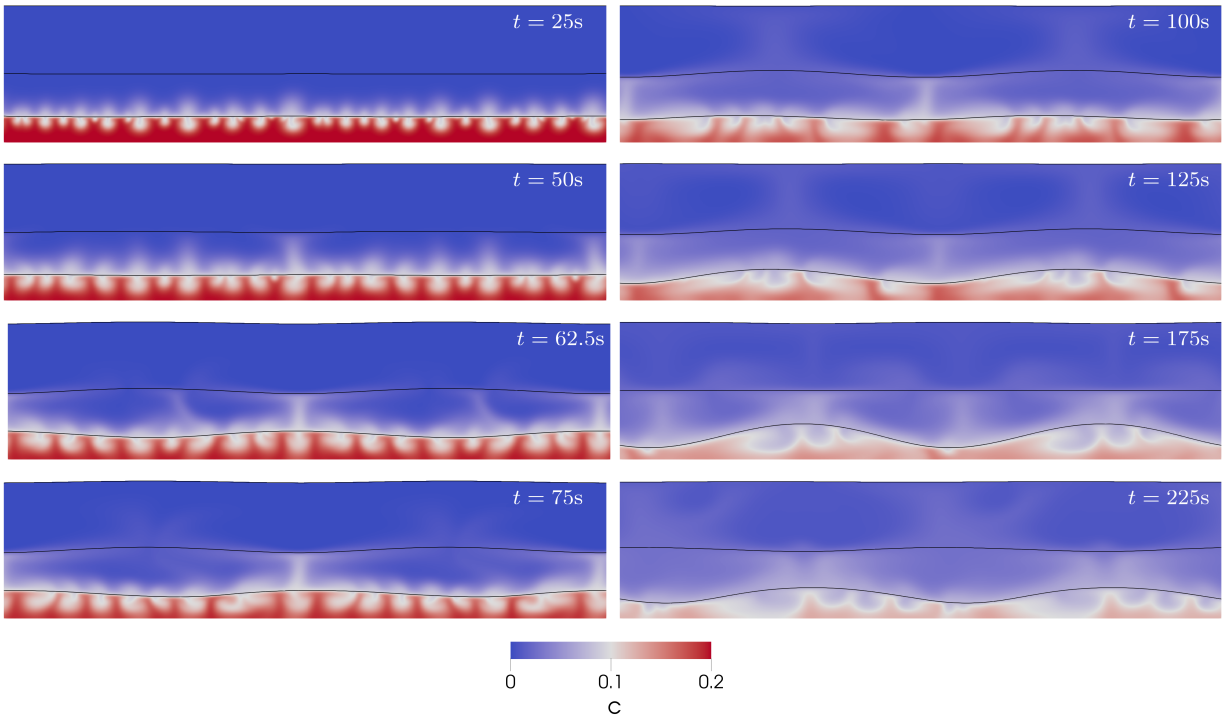


Figure 5.18: An interfacially turbulent matter cascade. Illustrated is the evolution and transfer of species between three fluid layers, starting from the bottom layer, at the indicated times. The Marangoni effect first generates the familiar plume structures in both the bottom and middle layers. Then, mass in the middle layer is captured and transported by the Marangoni stresses of the second interface, after which it enters the top layer. The process then repeats until the matter is exhausted. The 400μm$\times 200\mu$m computational domain is repeated across its periodic axis for presentation.

the Marangoni effect quickly captures and transports the matter tangentially along this interface, creating a "T" shape and a "matter conduit"—i.e., a concentrated region of mass flow—between the two surfaces. The species enters the top layer on the opposite side of the conduit; after which the flow reaches the top interface and Marangoni circulation cells form. After some time, the mass flow through the conduit dies down, and then at time 175 seconds, the process repeats.

In the last example, Figure 5.19 shows the results of a three-layer interfacially turbulent matter cascade with viscosity halved from the previous result. The color scheme is chosen to highlight the species mass concentration isocontours. In this example, similar matter junctions form between interfaces, with a higher degree of vorticity and more pronounced "matter turbulence" than in the previous result. This example highlights a cellular structure within the interfacially turbulent Marangoni matter cascade.

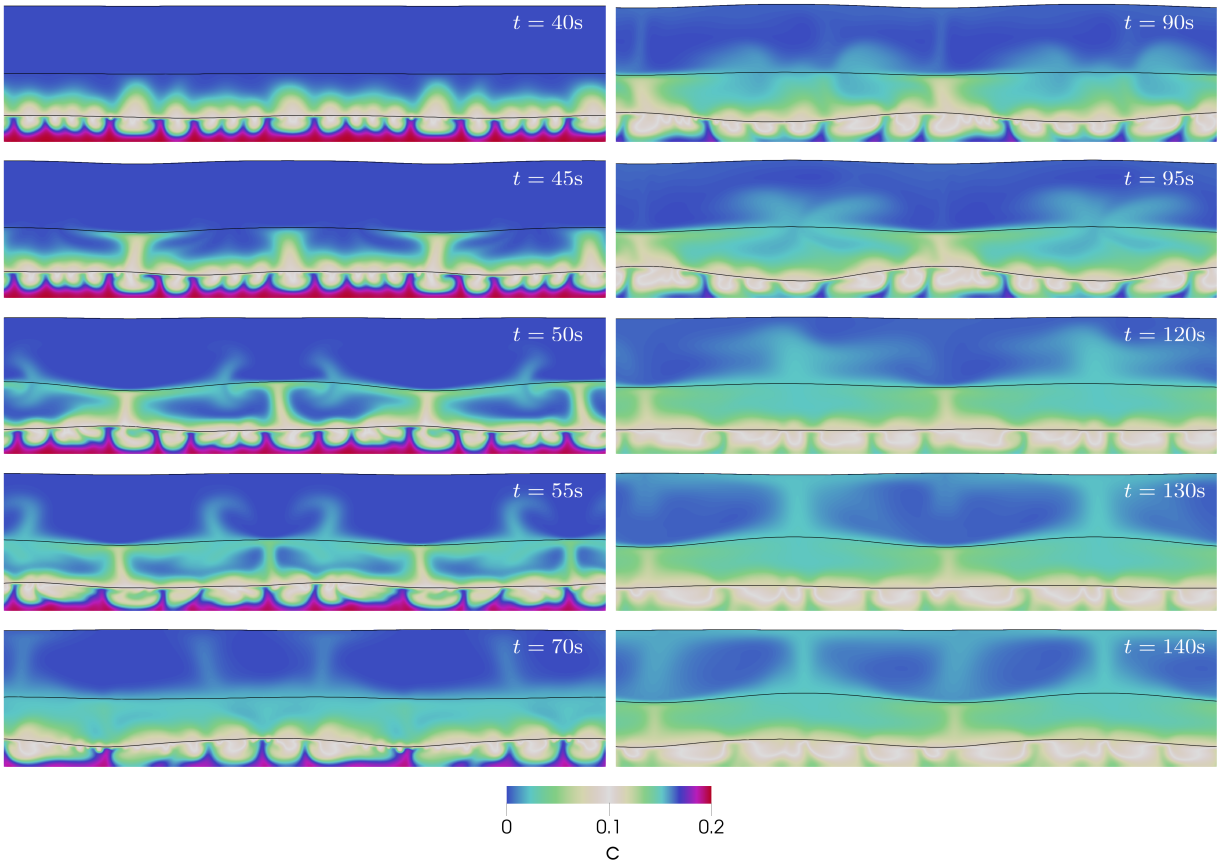


Figure 5.19: A three-layer interfacially turbulent matter cascade. The $400\mu\text{m}\times200\mu\text{m}$ computational domain is repeated across its periodic axis for presentation.

Chapter 6

Conclusions and Future Work

In this work, we developed a multi-physics mathematical model and accompanying high-order numerical framework to study multi-layer coating flow dynamics. These methods were used to study the fluid flow, leveling, and ultimate surface profile of multi-layer automobile paint coatings and to examine interfacial turbulence within a multi-layer matter cascade. Parametric studies were performed at industrially-relevant conditions, providing detailed insight into the physics of multi-layer coatings and examining the behavior and impact of various rheological parameters on the final paint smoothness. A few small-scale 3D simulations were also performed. Several numerical methods were developed, including: hybrid finite difference level set methods and implicit mesh discontinuous Galerkin methods for capturing sharp-interface multi-phase quasi-Newtonian fluid dynamics, making use of state-of-the-art fast multigrid Stokes solvers; local discontinuous Galerkin methods for Poisson problems with Robin boundary and jump conditions on implicitly-defined curved domains, to capture solvent evaporation; and a tailored finite difference projection algorithm for computing surface gradients within Marangoni stress calculations. Individually, the components range in orders of accuracy, from first-order mixed explicit-implicit time stepping methods, chosen for simplicity, to arbitrarily-high order accurate fluid dynamics via high-order DG methods; our particular choices led to an overall framework which is 2^{nd} order accurate in space and 1^{st} order in time. The numerical framework was implemented in C++ and parallelized using standard domain decomposition together with MPI; medium-scale 2D simulations over long-and-thin domains can be run on a modest multicore workstation; 3D simulations require larger supercomputing resources, chiefly due to the fact that many tens of thousands of time steps are needed to simulate over the long time scales associated with drying paint. The simulations of the 2D parametric study were performed on 8 nodes of the *Cori* supercomputer, using 256 total CPU cores. The typical duration of an individual time-step ranged from 5-7 seconds and the full simulations were run for 3-7 days.

Across this project, we have made significant progress toward understanding the key drivers of flow and leveling within the multi-layer system; the wide variety of insights include:

- The presence of short- and long-wave Marangoni instabilities, leading to the formation

of vortices/plumes within the clearcoat caused by solutal concentration gradients at the evaporative surface during the drying process. The plumes grow until reaching the basecoat, where steady-state circulating Bénard cells form. These cells, as well as long-wave oscillatory Marangoni modes, push and pull on the basecoat paint. This may greatly deform the film profile if the multi-layer paints are not "compatible"—meaning, the rheological parameters of two compatible paints act to mitigate the Marangoni forces. Additionally, clearcoats should aim to have a weak Marangoni profile.

- The typical Marangoni plume width was about twice its height, in agreement with experiments.
- The formation of a rough surface profile during the initial flash phase will leave imprints as the paints fully dry, with a slight degree of smoothing in the latter portion of drying time.
- The importance of the embedded paint-paint surface tension within the system. The force the film-layers impart on each other acts to keep the paint-paint surface flat. In low embedded surface tension simulations, the paint-paint surface quickly gave way to the Marangoni cells and significant deformation occurred. Paints that impart a higher degree of surface tension on each other are likely to be more compatible. Values of paint-paint surface tension coefficient are in general unknown; experimental studies into specific paint values would greatly aid the accuracy of future computational studies.
- A conjecture on the formation of holes within the multi-layer system is that incompatibility between paints allows for a high degree of interface deformation—leading to the basecoat or even the substrate to be exposed to air. The drilling phenomena observed in this work is one possible explanation of hole creation that is possibly linked to the experimentally observed cratering phenomena sometimes found in automobile paints.
- The telegraphing of substrate profile onto the paint-paint surface. Fourier frequency data found that the amplitude of wavelengths was higher for both the paint-paint surface and the evaporative surface in the case of the somewhat-bumpy smooth substrate $\mathcal{S}_{\text{smooth}}$, as compared to the completely flat substrate $\mathcal{S}_{\text{flat}}$.
- The dramatic difference in dynamics as a result of the mass diffusion coefficient. The mass diffusion coefficient dictates the width of the solvent boundary layer at the evaporative surface. The very-thin boundary layers caused by low mass diffusion coefficients result in large gradients that feed into the Marangoni forces. Consequently, for the purposes of carefully modeling coating flow dynamics, an accurate description and experimentally-derived values of mass diffusion coefficients for the various solvents constituting the paints are paramount.
- A "tree"-like structure of the Marangoni plumes for low values of solvent mass diffusion coefficient, with small plumes merging to form "branches" of the tree.
- The formation of Marangoni sheets in 3D—along with the merging of Marangoni filaments into physically consistent hexagonal patterns along the top evaporative surface.

Future Directions

Many avenues are available for future work on multi-layer coatings. Producing a smooth defect-free film of paint is of utmost importance to the automobile industry, therefore, a study on the formation of holes within multi-layer coatings may be considered. The drilling phenomena observed in this work is one possible explanation of hole creation, wherein the paint-paint and paint-gas surfaces intersect and expose the basecoats to air. Toward this end, one could generalize the numerical framework to allow the tracking of multiple intersecting paint-paint films, including contact lines and triple point motion. One possibility is to replace the use of height functions with a more general approach, such as the Voronoi implicit interface method (VIIM) [92]. Along the same lines, the finite difference algorithm for surface gradient and Marangoni stress calculations would need to be extended to more intricate geometries in this setting. The motion of contact lines between a paint surface and a solid wall may necessitate the use of slip models [93][94]. Towards this objective, a high-order accurate LDG method for Stokes problems with Navier-slip boundary conditions on implicitly-defined curved domains is presented in Appendix A. Additionally, further expansion of the domain size in 3D studies, which may require anisotropic meshes, will allow for capturing long-wave oscillatory Marangoni modes and provide greater insight into the nature of multi-layer coatings.

Numerous additional physical effects could be included in future work and some model assumptions reconsidered. For example, the model in this paper is purely isothermal; also, the interfaces do not carry solvent nor mass. At a first approximation, the effects of temperature could be incorporated by means of a quasi-thermodynamic model that slowly varies the rheological properties as a function of time. Meanwhile, dynamics constrained to the paint-paint and paint-gas interfaces, such as the motion and transfer of soluble or insoluble surfactants, could be incorporated via surface PDEs coupled to the volumetric flow dynamics. Doing so would allow for further fine-scaled studies of the intricate nature of the thin solutal boundary layer dynamics in evaporating Marangoni flows. The coating flow dynamics were found to vary dramatically with respect to the solvent mass diffusion coefficient—owing to the complex interplay of evaporation, diffusion, and Marangoni forces along the boundary layer. An experimentally-determined, variable diffusion coefficient could greatly alter the dynamics in unknown ways and provide a more complete model. One possible model for polymer-solvent diffusion using free-volume theory can be found in [95]. Incorporating polymer dynamics into the multi-layer coating flow problem may require replacing the idealized quasi-Newtonian fluid model with a fully non-Newtonian model.

Additional images of evaporative Marangoni flow can be found in Appendix C.
Thank you for reading my thesis. For making it to the end, here is a Marangoni flower.

— Luke P. Corcos

Figure 6.1: A Marangoni Flower

Bibliography

[1] R. I. Saye, J. A. Sethian, B. Petrouskie, A. Zatorsky, X. Lu, and R. Rock, "Insights from high-fidelity modeling of industrial rotary bell atomization," *The Proceedings of the National Academy of Sciences*, vol. 120, no. 4, 2023. DOI: 10.1073/pnas.2216709120.

[2] T. Pennington, "Waterborne painting process is a first at South Carolina BMW plant.," *Products Finishing, https://www.pfonline.com/articles/waterborne-painting-process-is-a-first-at-south-carolina-bmw- plant*,

[3] J. Thomson, "On certain curious motions observable at the surfaces of wine and other alcoholic liquors," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 10, no. 67, pp. 330–333, Nov. 1855, ISSN: 1941-5982. DOI: 10.1080/14786445508641982.

[4] C. Marangoni, "Ueber die Ausbreitung der Tropfen einer Flüssigkeit auf der Oberfläche einer anderen," *Annalen der Physik*, vol. 219, no. 7, pp. 337–354, 1871, ISSN: 1521-3889. DOI: 10.1002/andp.18712190702.

[5] D. Schwabe and A. Scharmann, "Some evidence for the existence and magnitude of a critical marangoni number for the onset of oscillatory flow in crystal growth melts," *Journal of Crystal Growth*, vol. 46, no. 1, pp. 125–131, 1979, ISSN: 0022-0248. DOI: 10.1016/0022-0248(79)90119-2.

[6] M. F. Schatz and G. P. Neitzel, "Experiments on Thermocapillary Instabilities," *Annual Review of Fluid Mechanics*, vol. 33, no. 1, pp. 93–127, 2001. DOI: 10.1146/annurev.fluid.33.1.93.

[7] D. Lohse, "Fundamental Fluid Dynamics Challenges in Inkjet Printing," *Annual Review of Fluid Mechanics*, vol. 54, no. 1, pp. 349–382, 2022. DOI: 10.1146/annurev-fluid-022321-114001.

[8] J. Ratulowski and H.-C. Chang, "Marangoni effects of trace impurities on the motion of long gas bubbles in capillaries," *Journal of Fluid Mechanics*, vol. 210, pp. 303–328, 1990, ISSN: 1469-7645, 0022-1120. DOI: 10.1017/S0022112090001306.

[9] A. Abbasian, S. R. Ghaffarian, N. Mohammadi, M. R. Khosroshahi, and M. Fathollahi, "Study on different planforms of paint's solvents and the effect of surfactants (on them)," *Progress in Organic Coatings*, vol. 49, no. 3, pp. 229–235, 2004, ISSN: 0300-9440. DOI: 10.1016/j.porgcoat.2003.09.020.

[10] N. Bassou and Y. Rharbi, "Role of Bénard-Marangoni Instabilities during Solvent Evaporation in Polymer Surface Corrugations," *Langmuir*, vol. 25, no. 1, pp. 624–632, 2009, ISSN: 0743-7463. DOI: 10.1021/la802979a.

[11] R. V. Craster and O. K. Matar, "Dynamics and stability of thin liquid films," *Reviews of Modern Physics*, vol. 81, no. 3, pp. 1131–1198, 2009. DOI: 10.1103/RevModPhys.81.1131.

[12] A. Oron, S. H. Davis, and S. G. Bankoff, "Long-scale evolution of thin liquid films," *Reviews of Modern Physics*, vol. 69, no. 3, pp. 931–980, 1997. DOI: 10.1103/RevModPhys.69.931.

[13] G. Durey, H. Kwon, Q. Magdelaine, *et al.*, "Marangoni bursting: Evaporation-induced emulsification of a two-component droplet," *Physical Review Fluids*, vol. 3, no. 10, p. 100 501, 2018.

[14] H. Bénard, "Les tourbillons cellulaires dans une nappe liquide. - Méthodes optiques d'observation et d'enregistrement," in *Journal de Physique Théorique et Appliquée*, vol. 10, 1901, pp. 254–266. DOI: 10.1051/jphystap:0190100100025400.

[15] M. J. Block, "Surface Tension as the Cause of Bénard Cells and Surface Deformation in a Liquid Film," *Nature*, vol. 178, no. 4534, pp. 650–651, Sep. 1956, ISSN: 1476-4687. DOI: 10.1038/178650a0.

[16] J. R. A. Pearson, "On convection cells induced by surface tension," *Journal of Fluid Mechanics*, vol. 4, no. 5, pp. 489–500, 1958, ISSN: 1469-7645, 0022-1120. DOI: 10.1017/S0022112058000616.

[17] L. E. Scriven and C. V. Sternling, "The Marangoni Effects," *Nature*, vol. 187, no. 4733, pp. 186–188, 1960, ISSN: 0028-0836, 1476-4687. DOI: 10.1038/187186a0.

[18] C. V. Sternling and L. E. Scriven, "Interfacial turbulence: Hydrodynamic instability and the marangoni effect," *AIChE Journal*, vol. 5, no. 4, pp. 514–523, 1959, ISSN: 1547-5905. DOI: 10.1002/aic.690050421.

[19] T. Sherwood and J. Wei, "Interfacial Phenomena in Liquid Extraction," *Industrial & Engineering Chemistry*, vol. 49, no. 6, pp. 1030–1034, 1957, ISSN: 0019-7866. DOI: 10.1021/ie50570a038.

[20] I. Langmuir and D. B. Langmuir, "The Effect of Monomolecular Films on the Evaporation of Ether Solutions.," *The Journal of Physical Chemistry*, vol. 31, no. 11, pp. 1719–1731, Nov. 1927, ISSN: 0092-7325. DOI: 10.1021/j150281a011.

[21] L. E. Scriven and C. V. Sternling, "On cellular convection driven by surface-tension gradients: Effects of mean surface tension and surface viscosity," *Journal of Fluid Mechanics*, vol. 19, no. 3, pp. 321–340, 1964, ISSN: 1469-7645, 0022-1120. DOI: 10.1017/S0022112064000751.

[22] S. J. Vanhook, M. F. Schatz, J. B. Swift, W. D. Mccormick, and H. L. Swinney, "Long-wavelength surface-tension-driven Bénard convection: Experiment and theory," *Journal of Fluid Mechanics*, vol. 345, pp. 45–78, 1997, ISSN: 1469-7645, 0022-1120. DOI: 10.1017/S0022112097006101.

[23] W. S. Overdiep, "The levelling of paints," *Progress in Organic Coatings*, vol. 14, no. 2, pp. 159–175, 1986, ISSN: 0300-9440. DOI: 10.1016/0033-0655(86)80010-3.

[24] S. D. Howison, J. A. Moriarty, J. R. Ockendon, E. L. Terrill, and S. K. Wilson, "A mathematical model for drying paint layers," *Journal of Engineering Mathematics*, vol. 32, no. 4, pp. 377–394, 1997, ISSN: 1573-2703. DOI: 10.1023/A:1004224014291.

[25] M. H. Eres, D. E. Weidner, and L. W. Schwartz, "Three-Dimensional Direct Numerical Simulation of Surface-Tension-Gradient Effects on the Leveling of an Evaporating Multicomponent Fluid," *Langmuir*, vol. 15, no. 5, pp. 1859–1871, 1999, ISSN: 0743-7463, 1520-5827. DOI: 10.1021/la980414u.

[26] P. de Gennes, "Instabilities during the evaporation of a film: Non-glassy polymer + volatile solvent," *The European Physical Journal E*, vol. 6, no. 1, pp. 421–424, 2001, ISSN: 1292-8941. DOI: 10.1007/s10189-001-8055-3.

[27] S. Kojima, T. Moriga, and K. Takenouchi, "The leveling of thermosetting waterborne coatings. Part I: Observation of leveling process," *Polymer Engineering & Science*, vol. 33, no. 20, pp. 1320–1328, 1993, ISSN: 1548-2634. DOI: 10.1002/pen.760332004.

[28] K. E. Strawhecker, S. K. Kumar, J. F. Douglas, and A. Karim, "The Critical Role of Solvent Evaporation on the Roughness of Spin-Cast Polymer Films," *Macromolecules*, vol. 34, no. 14, pp. 4669–4672, 2001, ISSN: 0024-9297. DOI: 10.1021/ma001440d.

[29] D. J. Taylor and D. P. Birnie, "A Case Study in Striation Prevention by Targeted Formulation Adjustment: Aluminum Titanate Sol-Gel Coatings," *Chemistry of Materials*, vol. 14, no. 4, pp. 1488–1492, 2002, ISSN: 0897-4756. DOI: 10.1021/cm010192c.

[30] E. Bodenschatz, W. Pesch, and G. Ahlers, "Recent Developments in Rayleigh-Bénard Convection," *Annual Review of Fluid Mechanics*, vol. 32, no. 1, pp. 709–778, 2000. DOI: 10.1146/annurev.fluid.32.1.709.

[31] S. H. Davis, "Thermocapillary Instabilities," *Annual Review of Fluid Mechanics*, vol. 19, no. 1, pp. 403–435, 1987. DOI: 10.1146/annurev.fl.19.010187.002155.

[32] A. F. Routh, "Drying of thin colloidal films," *Reports on Progress in Physics*, vol. 76, no. 4, p. 046 603, 2013, ISSN: 0034-4885. DOI: 10.1088/0034-4885/76/4/046603.

[33] E. Bänsch, S. Basting, and R. Krahl, "Numerical simulation of two-phase flows with heat and mass transfer," *Discrete and Continuous Dynamical Systems*, vol. 35, no. 6, p. 2325, 2015. DOI: 10.3934/dcds.2015.35.2325.

[34] J. Donea, S. Giuliani, and J. P. Halleux, "An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions," *Computer Methods in Applied Mechanics and Engineering*, vol. 33, no. 1, pp. 689–723, 1982, ISSN: 0045-7825. DOI: 10.1016/0045-7825(82)90128-1.

[35] S. G. Yiantsios, S. K. Serpetsi, F. Doumenc, and B. Guerrier, "Surface deformation and film corrugation during drying of polymer solutions induced by Marangoni phenomena," *International Journal of Heat and Mass Transfer*, vol. 89, pp. 1083–1094, 2015, ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2015.06.015.

[36] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations," *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988, ISSN: 0021-9991. DOI: 10.1016/0021-9991(88)90002-2.

[37] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences*. Cambridge University Press, 1999.

[38] R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher, "A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method)," *Journal of Computational Physics*, vol. 152, no. 2, pp. 457–492, 1999, ISSN: 0021-9991. DOI: 10.1006/jcph.1999.6236.

[39] S. Tanguy, T. Ménard, and A. Berlemont, "A Level Set Method for vaporizing two-phase flows," *Journal of Computational Physics*, vol. 221, no. 2, pp. 837–853, 2007, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2006.07.003.

[40] L. Rueda Villegas, R. Alis, M. Lepilliez, and S. Tanguy, "A Ghost Fluid/Level Set Method for boiling flows and liquid evaporation: Application to the Leidenfrost effect," *Journal of Computational Physics*, vol. 316, pp. 789–813, 2016, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2016.04.031.

[41] J. Papac, F. Gibou, and C. Ratsch, "Efficient symmetric discretization for the Poisson, heat and Stefan-type problems with Robin boundary conditions," *Journal of Computational Physics*, vol. 229, no. 3, pp. 875–889, 2010, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2009.10.017. (visited on 05/06/2022).

[42] K. Luo, C. Shao, M. Chai, and J. Fan, "Level set method for atomization and evaporation simulations," *Progress in Energy and Combustion Science*, vol. 73, pp. 65–94, 2019, ISSN: 0360-1285. DOI: 10.1016/j.pecs.2019.03.001.

[43] M. Chai, K. Luo, C. Shao, H. Wang, and J. Fan, "A finite difference discretization method for heat and mass transfer with Robin boundary conditions on irregular domains," *Journal of Computational Physics*, vol. 400, p. 108 890, 2020, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2019.108890.

[44] M. Chai, K. Luo, C. Shao, H. Wang, and J. Fan, "Interface-resolved detailed numerical simulation of evaporating two-phase flows with robin boundary conditions on irregular domains," *International Journal of Heat and Mass Transfer*, vol. 145, p. 118 774, 2019, ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2019.118774.

[45] R. Bhardwaj, X. Fang, and D. Attinger, "Pattern formation during the evaporation of a colloidal nanoliter drop: A numerical and experimental study," *New Journal of Physics*, vol. 11, no. 7, p. 075 020, Jul. 2009, ISSN: 1367-2630. DOI: 10.1088/1367-2630/11/7/075020.

[46] F. Girard, M. Antoni, S. Faure, and A. Steinchen, "Evaporation and Marangoni Driven Convection in Small Heated Water Droplets," *Langmuir*, vol. 22, no. 26, pp. 11 085–11 091, Dec. 2006, Publisher: American Chemical Society, ISSN: 0743-7463. DOI: 10.1021/la0615721.

[47] H. Hu and R. G. Larson, "Analysis of the Effects of Marangoni Stresses on the Microflow in an Evaporating Sessile Droplet," *Langmuir*, vol. 21, no. 9, pp. 3972–3980, Apr. 2005, ISSN: 0743-7463, 1520-5827. DOI: 10.1021/la0475270.

[48] O. E. Ruiz and W. Z. Black, "Evaporation of Water Droplets Placed on a Heated Horizontal Surface," *Journal of Heat Transfer*, vol. 124, no. 5, pp. 854–863, Sep. 2002, ISSN: 0022-1481. DOI: 10.1115/1.1494092.

[49] G. P. Sasmal and J. I. Hochstein, "Marangoni Convection With a Curved and Deforming Free Surface in a Cavity," *Journal of Fluids Engineering*, vol. 116, no. 3, pp. 577–582, Sep. 1994, ISSN: 0098-2202. DOI: 10.1115/1.2910316.

[50] M. Francois, J. Sicilian, and D. Kothe, "Modeling of Thermocapillary Forces within a Volume Tracking Algorithm," vol. 2, Jun. 2006.

[51] G. Son, "A sharp-interface level-set method for analysis of Marangoni effect on microdroplet evaporation," *International Communications in Heat and Mass Transfer*, vol. 58, pp. 156–165, 2014, ISSN: 0735-1933. DOI: 10.1016/j.icheatmasstransfer.2014.08.024.

[52] J.-J. Xu, Y. Yang, and J. Lowengrub, "A level-set continuum method for two-phase flows with insoluble surfactant," *Journal of Computational Physics*, vol. 231, no. 17, pp. 5897–5909, 2012, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2012.05.014.

[53] J.-J. Xu, W. Shi, and M.-C. Lai, "A level-set method for two-phase flows with soluble surfactant," *Journal of Computational Physics*, vol. 353, pp. 336–355, 2018, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2017.10.019.

[54] T. Köllner, K. Schwarzenberger, K. Eckert, and T. Boeck, "Multiscale structures in solutal Marangoni convection: Three-dimensional simulations and supporting experiments," *Physics of Fluids*, vol. 25, no. 9, p. 092 109, Sep. 2013, ISSN: 1070-6631. DOI: 10.1063/1.4821536.

[55] K. Schwarzenberger, T. Köllner, H. Linde, T. Boeck, S. Odenbach, and K. Eckert, "Pattern formation and mass transfer under stationary solutal Marangoni instability," *Advances in Colloid and Interface Science*, Manuel G. Velarde, vol. 206, pp. 344–371, Apr. 2014, ISSN: 0001-8686. DOI: 10.1016/j.cis.2013.10.003.

[56] R. Abeyaratne, "Continuum Mechanics," *Lecture Notes on The Mechanics of Elastic Solids*, 1998.

[57] L. C. Evans, *Partial Differential Equations*. American Mathematical Soc., 2010, vol. 19.

[58] R. I. Saye and J. A. Sethian, "Chapter 6 - A review of level set methods to model interfaces moving under complex physics: Recent challenges and advances," in *Handbook of Numerical Analysis*, ser. Geometric Partial Differential Equations - Part I, A. Bonito and R. H. Nochetto, Eds., vol. 21, Elsevier, 2020, pp. 509–554. DOI: 10.1016/bs.hna.2019.07.003.

[59] J. A. Sethian, "Level set methods: An initial value formulation.," *https://math.berkeley.edu/∼sethian/2006/Explanations/level_set_explain.html*,

[60] D. Adalsteinsson and J. A. Sethian, "A Fast Level Set Method for Propagating Interfaces," *Journal of Computational Physics*, vol. 118, no. 2, pp. 269–277, 1995, ISSN: 0021-9991. DOI: 10.1006/jcph.1995.1098.

[61] D. Adalsteinsson and J. A. Sethian, "The Fast Construction of Extension Velocities in Level Set Methods," *Journal of Computational Physics*, vol. 148, no. 1, pp. 2–22, 1999, ISSN: 0021-9991. DOI: 10.1006/jcph.1998.6090.

[62] R. Saye, "High-order methods for computing distances to implicitly defined surfaces," *Communications in Applied Mathematics and Computational Science*, vol. 9, no. 1, pp. 107–141, 2014, ISSN: 2157-5452. DOI: 10.2140/camcos.2014.9.107.

[63] F. Gibou, R. Fedkiw, and S. Osher, "A review of level-set methods and some recent applications," *Journal of Computational Physics*, vol. 353, pp. 82–109, 2018, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2017.10.006.

[64] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, "Uniformly High Order Accurate Essentially Non-oscillatory Schemes, III," in *Upwind and High-Resolution Schemes*, M. Y. Hussaini, B. van Leer, and J. Van Rosendale, Eds., Berlin, Heidelberg: Springer, 1997, pp. 218–290, ISBN: 978-3-642-60543-7. DOI: 10.1007/978-3-642-60543-7_12.

[65] R. Saye, "High-Order Quadrature Methods for Implicitly Defined Surfaces and Volumes in Hyperrectangles," *SIAM Journal on Scientific Computing*, vol. 37, no. 2, A993–A1019, 2015, ISSN: 1064-8275. DOI: 10.1137/140966290.

[66] R. Saye, "Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part I," *Journal of Computational Physics*, vol. 344, pp. 647–682, 2017, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2017.04.076.

[67] W. Reed and T. Hill, "Triangular mesh methods for the neutron transport equation," *Los Alamos Report LA-UR-73-479*, 1973.

[68] B. Cockburn and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework," *Mathematics of computation*, vol. 52, no. 186, pp. 411–435, 1989.

[69] D. N. Arnold, "An Interior Penalty Finite Element Method with Discontinuous Elements," *SIAM Journal on Numerical Analysis*, vol. 19, no. 4, pp. 742–760, 1982, ISSN: 0036-1429. DOI: 10.1137/0719052.

[70] B. Cockburn and C.-W. Shu, "The Local Discontinuous Galerkin Method for Time-Dependent Convection-Diffusion Systems," *SIAM Journal on Numerical Analysis*, vol. 35, no. 6, pp. 2440–2463, 1998, ISSN: 0036-1429. DOI: 10.1137/S0036142997316712.

[71] B. Cockburn, G. Kanschat, D. Schötzau, and C. Schwab, "Local Discontinuous Galerkin Methods for the Stokes System," *SIAM Journal on Numerical Analysis*, vol. 40, no. 1, pp. 319–343, 2002, ISSN: 0036-1429. DOI: 10.1137/S0036142900380121.

[72] R. Saye, "Fast multigrid solution of high-order accurate multiphase Stokes problems," *Communications in Applied Mathematics and Computational Science*, vol. 15, no. 2, pp. 147–196, 2020, ISSN: 2157-5452. DOI: 10.2140/camcos.2020.15.33.

[73] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, "Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems," *SIAM Journal on Numerical Analysis*, vol. 39, no. 5, pp. 1749–1779, 2002, ISSN: 0036-1429. DOI: 10.1137/S0036142901384162.

[74] W. Pazner and P.-O. Persson, "On the convergence of iterative solvers for polygonal discontinuous Galerkin discretizations," 2018. DOI: 10.2140/camcos.2018.13.27.

[75] R. Saye, "Interfacial gauge methods for incompressible fluid dynamics," *Science Advances*, vol. 2, no. 6, 2016. DOI: 10.1126/sciadv.1501869.

[76] A. Baggag, H. Atkins, and D. Keyes, "Parallel implementation of the discontinuous Galerkin method," *Tech. rep. Institute for Computer Applications in Science and Engineering*, 1999.

[77] J. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods*. Springer, ISBN: 978-0-387-72065-4.

[78] D. Fortunato, C. H. Rycroft, and R. Saye, "Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods," *SIAM Journal on Scientific Computing*, vol. 41, no. 6, A3913–A3937, 2019, ISSN: 1064-8275. DOI: 10.1137/18M1206357.

[79] R. Saye, "Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part II," *Journal of Computational Physics*, vol. 344, pp. 683–723, 2017, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2017.05.003.

[80] R. Saye, "Efficient multigrid solution of elliptic interface problems using viscosity-upwinded local discontinuous Galerkin methods," *Communications in Applied Mathematics and Computational Science*, vol. 14, no. 2, pp. 247–283, 2019, ISSN: 2157-5452. DOI: 10.2140/camcos.2019.14.247.

[81] V. Gulizzi and R. Saye, "Modeling wave propagation in elastic solids via high-order accurate implicit-mesh discontinuous galerkin methods," *Computer Methods in Applied Mechanics and Engineering*, vol. 395, p. 114 971, 2022.

[82] R. Saye, *Algoim – Algorithms for implicitly defined geometry, level set methods, and Voronoi implicit interface methods*, 2022. [Online]. Available: `https://algoim.github.io/`.

[83] J. Lions, "Problemes aux limites non homogenesa donées irrégulieres: Une méthode d'approximation," *Numerical Analysis of Partial Differential Equations (CIME 2 Ciclo, Ispra, 1967), Edizioni Cremonese, Rome*, pp. 283–292, 1968.

[84] J. Nitsche, "Über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind," in *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, Springer, vol. 36, 1971, pp. 9–15.

[85] L. S. Blackford, J. Choi, A. Cleary, *et al.*, *ScaLAPACK Users' Guide*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1997, ISBN: 0-89871-397-8 (paperback).

[86] W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial, 2nd Edition*. 2000, ISBN: 978-0-89871-462-3.

[87] Y. Saad, *Iterative Methods for Sparse Linear Systems, 2nd Edition*. 2003, ISBN: 978-0-89871-534-7.

[88] M. Juntunen and R. Stenberg, "Nitsche's method for general boundary conditions," *Mathematics of computation*, vol. 78, no. 267, pp. 1353–1374, 2009.

[89] C. K. Schoff, "Craters and Other Coatings Defects: Mechanisms and Analysis," in *Protective Coatings: Film Formation and Properties*, M. Wen and K. Dušek, Eds., Cham: Springer International Publishing, 2017, pp. 403–425, ISBN: 978-3-319-51627-1. DOI: 10.1007/978-3-319-51627-1_18.

[90] Y. Zhong, Y. Zhuo, Z. Wang, and Y. Sha, "Marangoni convection induced by simultaneous mass and heat transfer during evaporation of n-heptane/ether binary liquid mixture," *International Journal of Heat and Mass Transfer*, vol. 108, pp. 812–821, May 2017, ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2016.12.038.

[91] A. Bailey, "Surface and interfacial tension," *Thermopedia*, 2010. DOI: 10.1615/AtoZ.s.surface_and_interfacial_tension.

[92] R. I. Saye and J. A. Sethian, "The voronoi implicit interface method for computing multiphase physics," *Proceedings of the National Academy of Sciences*, vol. 108, no. 49, pp. 19 498–19 503, 2011.

[93] C. Huh and L. E. Scriven, "Hydrodynamic model of steady movement of a solid/liquid/fluid contact line," *Journal of Colloid and Interface Science*, vol. 35, no. 1, pp. 85–101, 1971, ISSN: 0021-9797. DOI: 10.1016/0021-9797(71)90188-3.

[94] V. Dussan, "The moving contact line: The slip boundary condition," *Journal of Fluid Mechanics*, vol. 77, no. 4, pp. 665–684, 1976, Publisher: Cambridge University Press, ISSN: 1469-7645, 0022-1120. DOI: 10.1017/S0022112076002838.

[95] J. S. Vrentas and J. L. Duda, "Molecular diffusion in polymer solutions," *AIChE Journal*, vol. 25, no. 1, pp. 1–24, 1979.

Appendix A

LDG for Stokes Problems with Navier-Slip Boundary Conditions

The motion of contact lines between a paint surface and a solid wall may necessitate the use of slip models [93][94]. As a preliminary step towards applying our hybrid numerical framework for the multi-layer coating flow problem to other flow settings, in this section, we develop high-order accurate local discontinuous Galerkin methods for Stokes problems with Navier-slip boundary conditions on implicitly-defined domains. The methods follow a similar structure as the LDG methods of [72]. The resulting discretization has several favorable properties: for example, it is optimally high-order accurate in the fluid velocity field and the final linear system is symmetric; moreover, the LDG discretization is amenable to straightforward, fast, multigrid-preconditioned GMRES solvers [72].

Let $\Omega \subset \mathbb{R}^d$ represent a single-phase d-dimensional domain with Navier-slip boundary conditions applied on the domain boundary, which is denoted Γ_{Nv}. Assume that Γ_{Nv} is sufficiently smooth. The time-independent Stokes problem with Navier-slip boundary conditions involves finding a velocity field $\boldsymbol{u} : \Omega \to \mathbb{R}^d$ and pressure $p : \Omega \to \mathbb{R}$ such that

$$\begin{cases} -\nabla \cdot (\mu(\nabla \boldsymbol{u} + \gamma \nabla \boldsymbol{u}^T)) + \nabla p = \boldsymbol{f} & \text{in } \Omega \\ -\nabla \cdot \boldsymbol{u} = f_{\text{div}} & \text{in } \Omega \\ \boldsymbol{u} \cdot \boldsymbol{n} = g & \text{on } \Gamma_{Nv} \\ (\mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)\boldsymbol{n} - p\boldsymbol{n} + \alpha \boldsymbol{u})_\tau = \boldsymbol{h} & \text{on } \Gamma_{Nv}, \end{cases} \tag{A.1}$$

where $\mu : \mathbb{R}^d \to \mathbb{R}^+$ and $\alpha : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ are functions mapping onto the space of positive real numbers and the space of non-negative real numbers respectively. Here μ represents the viscosity of the fluid and α is a slip parameter that goes as the inverse of the slip length, with $\alpha = 0$ reducing the problem to the "free"-slip condition and $\alpha \to \infty$ approaching the no-slip condition. We assume that the source data $\boldsymbol{f}, f_{\text{div}}$ and boundary data $g, \boldsymbol{h}$ are sufficiently smooth. Here $\boldsymbol{n}$ is the unit normal vector and τ is the tangential direction to the boundary. Let $\boldsymbol{u}_n = (\boldsymbol{u} \cdot \boldsymbol{n})\boldsymbol{n}$ represent the normal component of $\boldsymbol{u}$ and let $\boldsymbol{u}_\tau$ represent the projection

of $\boldsymbol{u}$ onto the tangential plane orthogonal to $\boldsymbol{n}$, defined as $\boldsymbol{u}_\tau = (\mathbb{I} - \boldsymbol{n} \otimes \boldsymbol{n})\boldsymbol{u}$, where $\mathbb{I}$ is the identity tensor and $\otimes$ is the Kronecker product. Also let $\boldsymbol{\sigma}_n = (\boldsymbol{n} \otimes \boldsymbol{n})\boldsymbol{\sigma}$. Note that the right-hand-side $\boldsymbol{h}$ of the Navier-slip condition is a vector along the tangential plane, with $\boldsymbol{h} \cdot \boldsymbol{n} = 0$. The value γ is either 0 or 1 depending on whether the Stokes problem is in regular form or viscous-stress form respectively, noting that viscous-stress form should be used for problems involving slip.

We present an LDG method for (A.1) similar to the LDG methods for Poisson problems with Robin boundary conditions developed in section 3.3.2, however, the LDG formulation for Stokes problems with Navier-slip boundary conditions is more mathematically intricate. The numerical fluxes are chosen to be consistent with the PDE and the boundary conditions, leading to an LDG discretization that provides a symmetric discrete linear system and high-order accurate solutions to (A.1) on implicitly defined domains, in a dimension-independent fashion, for variable viscosity and slip coefficients spanning several orders of magnitude. The LDG methods presented in section A.2 are constructed through the following five steps [72]:

(i) Introduce the gradient $\boldsymbol{\eta} \in V_h^{d\times d}$ such that $\boldsymbol{\eta} = \nabla \boldsymbol{u}$ weakly via the strong-weak form.

(ii) Define the stress-tensor $\boldsymbol{\sigma} \in V_h^{d\times d}$ as the L^2 projection of $\mu(\boldsymbol{\eta} + \gamma \boldsymbol{\eta}^T) - p\mathbb{I}$.

(iii) Compute the divergence of the stress-tensor $\boldsymbol{w} \in V_h^d$ such that $\boldsymbol{w} = \nabla \cdot \boldsymbol{\sigma}$ weakly via the weak-weak form.

(iv) Enforce the divergence constraint $w \in V_h$ such that $w = \nabla \cdot \boldsymbol{u}$ via the strong-weak form.

(v) Require that $-(\boldsymbol{w}, w)$ equals the L^2 projection of $(\boldsymbol{f}, f_{\text{div}})$, while also incorporating penalty stabilization to enforce continuity and boundary conditions.

A.1 Numerical fluxes

The choice of numerical fluxes (indicated by the $(*)$ superscript, see section 3.2 for details) for the LDG discretization for the Navier-slip boundary Γ_{Nv} takes from the boundary data in the direction it is available and takes from the interior element traces in the direction it is not (indicated by the $(-)$ superscript), setting

$$\begin{cases} \boldsymbol{u}^* & = g\boldsymbol{n} + \boldsymbol{u}_\tau^- \\ (\boldsymbol{\sigma} \cdot \boldsymbol{n})^* & = \boldsymbol{\sigma}_n^- \cdot \boldsymbol{n} + (\boldsymbol{h} - (\alpha \boldsymbol{u})_\tau^-). \end{cases} \tag{A.2}$$

Note how each numerical flux is split into normal and tangential components. The numerical flux for the stress tensor considers only the normal direction; this is sufficient for imposing the Navier-slip boundary conditions via the LDG weak forms. To prove the consistency of this numerical flux choice with respect to the boundary condition, consider their projection onto the surface normal vector $\boldsymbol{n}$, as well as onto a vector defined along the tangential plane, denoted $\boldsymbol{\tau}$. It is clear that $\boldsymbol{u}^* \cdot \boldsymbol{n} = g$ and $\boldsymbol{u}^* \cdot \boldsymbol{\tau} = \boldsymbol{u}_\tau^-$. For the normal stress numerical flux:

$$\boldsymbol{n} \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{n})^* = \boldsymbol{n} \cdot (\boldsymbol{n} \otimes \boldsymbol{n})\boldsymbol{\sigma}^- \cdot \boldsymbol{n} + \boldsymbol{n} \cdot (h - (\alpha \boldsymbol{u})_\tau^-) = \boldsymbol{n} \cdot \boldsymbol{\sigma}^- \cdot \boldsymbol{n}$$

and

$$\boldsymbol{\tau}\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{n})^* = \boldsymbol{\tau}\cdot(\boldsymbol{n}\otimes\boldsymbol{n})\boldsymbol{\sigma}^-\cdot\boldsymbol{n} + \boldsymbol{\tau}\cdot(h-(\alpha\boldsymbol{u})_\tau^-) = \boldsymbol{\tau}\cdot(h-(\alpha\boldsymbol{u})_\tau^-).$$

This demonstrates that our numerical flux choice for the Navier boundary conditions is consistent and uses the boundary data in the direction it is available while using values from the interior element traces in the directions it is not.

A.2 Local discontinuous Galerkin methods

We now present the LDG method for (A.1) in the five steps previously described.

(i) Define $\boldsymbol{\eta}\in V_h^{d\times d}$ such that $\boldsymbol{\eta}=\nabla\boldsymbol{u}$ weakly for element $E\in\mathcal{E}$ via the strong-weak form

$$\int_E \boldsymbol{\eta}:\boldsymbol{\omega} = \int_E \nabla\boldsymbol{u}:\boldsymbol{\omega} + \int_{\partial E}(\boldsymbol{u}^*-\boldsymbol{u})\cdot\boldsymbol{\omega}\cdot\boldsymbol{n},$$

for all test functions $\boldsymbol{\omega}$ on element E. Summing over all elements gives

$$\int_\Omega \boldsymbol{\eta}:\boldsymbol{\omega} = \sum_E\int_E \nabla\boldsymbol{u}:\boldsymbol{\omega} + \int_{\Gamma_0}(\boldsymbol{u}^*-\boldsymbol{u}^-)\cdot\boldsymbol{\omega}^-\cdot\boldsymbol{n} - \int_{\Gamma_0}(\boldsymbol{u}^*-\boldsymbol{u}^+)\cdot\boldsymbol{\omega}^+\cdot\boldsymbol{n} + \int_{\Gamma_{Nv}}(\boldsymbol{u}^*-\boldsymbol{u}^-)\cdot\boldsymbol{\omega}^-\cdot\boldsymbol{n},$$

where Γ_0 is the collection of intraphase faces. Take the intraphase numerical flux to be one-sided and from the left. Define the numerical flux $\boldsymbol{u}^*$ for the Navier boundary as in section A.1, setting

$$\boldsymbol{u}^* = \begin{cases}\boldsymbol{u}^- & \text{on } \Gamma_0\\ g\boldsymbol{n}+\boldsymbol{u}_\tau^- & \text{on } \Gamma_{Nv}.\end{cases}$$

Plugging in the numerical fluxes gives

$$\int_\Omega \boldsymbol{\eta}:\boldsymbol{\omega} = \sum_E\int_E \nabla\boldsymbol{u}:\boldsymbol{\omega} + \int_{\Gamma_0}(\boldsymbol{u}^+-\boldsymbol{u}^-)\cdot\boldsymbol{\omega}^+\cdot\boldsymbol{n} + \int_{\Gamma_{Nv}}(g-\boldsymbol{u}^-\cdot\boldsymbol{n})\boldsymbol{n}\cdot\boldsymbol{\omega}^-\cdot\boldsymbol{n}.$$

- Define the *Broken Gradient* operator $\nabla_h: V_h\to V_h^d$ and the *Lifting* operator $L: V_h\to V_h^d$ such that, respectively,

$$\int_\Omega \nabla_h u\cdot\boldsymbol{v} = \sum_E\int_E \nabla u\cdot\boldsymbol{v}, \quad \int_\Omega Lu\cdot\boldsymbol{v} = \int_{\Gamma_0}(u^+-u^-)\boldsymbol{v}^+\cdot\boldsymbol{n},$$

  for $u\in V_h$ and for all $\boldsymbol{v}\in V_h^d$. Now, define the *Gradient* operator $G: V_h\to V_h^d$ as $G=\nabla_h+L$.

- Define $J_g\in V_h^{d\times d}$ such that

$$\int_\Omega J_g(g):\boldsymbol{\omega} = \int_{\Gamma_{Nv}} g\boldsymbol{n}\cdot\boldsymbol{\omega}^-\cdot\boldsymbol{n},$$

  for all $\boldsymbol{\omega}\in V_h^{d\times d}$.

- Define the *Navier Boundary Lifting Operator* to be $B : V_h^d \to V_h^{d\times d}$

$$\int_\Omega B(\boldsymbol{u}) : \boldsymbol{\omega} = -\int_{\Gamma_{Nv}} (\boldsymbol{u}^- \cdot \boldsymbol{n})\boldsymbol{n} \cdot \boldsymbol{\omega}^- \cdot \boldsymbol{n},$$

for all $\boldsymbol{\omega} \in V_h^{d\times d}$. Breaking this down into components gives

$$\int_\Omega B_{ij}(\boldsymbol{u})v = -\int_{\Gamma_{Nv}} (\boldsymbol{u}^- \cdot \boldsymbol{n})n_i n_j v^- = -\sum_k \int_{\Gamma_{Nv}} u_k^- n_k n_i n_j v^-,$$

for all $v \in V_h$, and for $0 \le i, j, k \le d$. Here u_k and n_k denote the k-th component of $\boldsymbol{u}$ and $\boldsymbol{n}$ respectively. Now define sub-operator $b_{ij}^k : V_h \to V_h$ such that

$$\int_\Omega b_{ij}^k(u)v = -\int_{\Gamma_{Nv}} u^- n_k n_i n_j v^-,$$

for $u \in V_h$ and for all $v \in V_h$. So $B_{ij}(\boldsymbol{u}) = \sum_k b_{ij}^k(u_k)$.

Now piecing together $\boldsymbol{\eta} = \nabla \boldsymbol{u}$ gives

$$\boldsymbol{\eta}_{ij} = G_j u_i + J_{g,ij} + B_{ij}(\boldsymbol{u}) = G_j u_i + J_{g,ij} + \sum_k b_{ij}^k(u_k).$$

Take $\boldsymbol{\zeta} = \boldsymbol{\eta} + \gamma \boldsymbol{\eta}^T$, then

$$\boldsymbol{\zeta}_{ij} = G_j u_i + \gamma G_i u_j + J_{g,ij} + \gamma J_{g,ji} + \sum_k (b_{ij}^k + \gamma b_{ji}^k)(u_k).$$

(ii) Now define the stress tensor $\boldsymbol{\sigma} \in V_h^{d\times d}$ as the L^2 projection of $\mu\boldsymbol{\zeta} - p\mathbb{I}$. The components of which are given by

$$\boldsymbol{\sigma}_{ij} = M^{-1}M_\mu(G_j u_i + \gamma G_i u_j + J_{g,ij} + \gamma J_{g,ji} + \sum_k (b_{ij}^k + \gamma b_{ji}^k)(u_k)) - p\delta_{ij},$$

for $1 \le i, j \le d$. Here M is the mass matrix, M_μ is the μ-weighted mass matrix such that $v^T M_\mu u = \int_\Omega v\mu u$ for all $u, v \in V_h$, and δ_{ij} is the Kronecker delta.

(iii) Define the divergence of the stress tensor $\boldsymbol{w} \in V_h^d$ such that $\boldsymbol{w} = \nabla \cdot \boldsymbol{\sigma}$ on element $E \in \mathcal{E}$ weakly via the weak-weak form

$$\int_E \boldsymbol{w} \cdot \boldsymbol{v} = -\int_E \boldsymbol{\sigma} : \nabla \boldsymbol{v} + \int_{\partial E} \boldsymbol{v} \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{n})^*,$$

for all test functions $\boldsymbol{v}$ on element E. Summing over all elements gives

$$\int_\Omega \boldsymbol{w} \cdot \boldsymbol{v} = -\sum_E \int_E \boldsymbol{\sigma} : \nabla \boldsymbol{v} + \int_{\Gamma_0} (\boldsymbol{v}^- - \boldsymbol{v}^+) \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{n})^* + \int_{\Gamma_{Nv}} \boldsymbol{v}^- \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{n})^*.$$

Define the normal stress numerical flux $(\boldsymbol{\sigma}\cdot\boldsymbol{n})^*$ for the Navier boundary as in section A.1 and take the intraphase flux on Γ_0 from the right, i.e., in the opposite direction as that for $\boldsymbol{u}$, setting

$$(\boldsymbol{\sigma}\cdot\boldsymbol{n})^* = \begin{cases} \boldsymbol{\sigma}^+\cdot\boldsymbol{n} & \text{on } \Gamma_0 \\ \boldsymbol{\sigma}_n^-\cdot\boldsymbol{n} + (\boldsymbol{h} - (\alpha\boldsymbol{u})_\tau^-) & \text{on } \Gamma_{Nv}. \end{cases}$$

Note that $\boldsymbol{v}^-\cdot\boldsymbol{\sigma}_n^-\cdot\boldsymbol{n} = \boldsymbol{v}^-\cdot(\boldsymbol{n}\otimes\boldsymbol{n})\boldsymbol{\sigma}^-\cdot\boldsymbol{n} = (\boldsymbol{v}^-\cdot\boldsymbol{n})\boldsymbol{n}\cdot\boldsymbol{\sigma}^-\cdot\boldsymbol{n}$. Plugging in the numerical fluxes gives

$$\int_\Omega \boldsymbol{w}\cdot\boldsymbol{v} = -\sum_E\int_E \boldsymbol{\sigma}:\nabla\boldsymbol{v} - \int_{\Gamma_0}(\boldsymbol{v}^+-\boldsymbol{v}^-)\cdot\boldsymbol{\sigma}^+\cdot\boldsymbol{n} + \int_{\Gamma_{Nv}}(\boldsymbol{v}^-\cdot\boldsymbol{n})\boldsymbol{n}\cdot\boldsymbol{\sigma}^-\cdot\boldsymbol{n} + \int_{\Gamma_{Nv}}\boldsymbol{v}^-\cdot(\boldsymbol{h}-(\alpha\boldsymbol{u})_\tau^-).$$

- The first two terms in the equation are equivalent to $-\sum_i(\boldsymbol{\sigma}_i, Gv_i)$, where $\boldsymbol{\sigma}_i$ is the i-th row of $\boldsymbol{\sigma}$ and v_i the i-th component of $\boldsymbol{v}$, for $1\le i\le d$.
- Define $J_h\in V_h^d$ such that

  $$\int_\Omega J_h(\boldsymbol{h})\cdot\boldsymbol{v} = \int_{\Gamma_{Nv}}\boldsymbol{v}^-\cdot\boldsymbol{h},$$

  for all $\boldsymbol{v}\in V_h^d$.
- Define operator $A^{Nv}: V_h^d\to V_h^d$ to account for the slip term

  $$\int_\Omega A^{Nv}(\boldsymbol{u})\cdot\boldsymbol{v} = \int_{\Gamma_{Nv}}\boldsymbol{v}^-\cdot(\alpha\boldsymbol{u})_\tau^- = \int_{\Gamma_{Nv}}\boldsymbol{v}^-\cdot(\mathbb{I}-\boldsymbol{n}\otimes\boldsymbol{n})(\alpha\boldsymbol{u})^-,$$

  for all $\boldsymbol{v}\in V_h^d$, which component-wise gives

  $$\int_\Omega A_i^{Nv}(\boldsymbol{u})v_i = \int_{\Gamma_{Nv}} v_i^-(\alpha\boldsymbol{u})_i^- - \sum_j\int_{\Gamma_{Nv}} v_i^- n_i n_j(\alpha\boldsymbol{u})_j^-.$$

  The block operator A_{ij}^{Nv} is therefore given by

  $$\int_\Omega A_{ij}^{Nv}(\boldsymbol{u})v = \delta_{ij}\int_{\Gamma_{Nv}} v^-(\alpha\boldsymbol{u})_i^- - \int_{\Gamma_{Nv}} v^- n_i n_j(\alpha\boldsymbol{u})_j^-,$$

  for all $v\in V_h$ and for all $1\le i,j\le d$.
- The third term in the divergence equation is equivalent to

  $$\begin{aligned}\int_\Omega B(\boldsymbol{v}):\boldsymbol{\sigma} &= -\int_{\Gamma_{Nv}}(\boldsymbol{v}^-\cdot\boldsymbol{n})\boldsymbol{n}\cdot\boldsymbol{\sigma}^-\cdot\boldsymbol{n} = -\int_{\Gamma_{Nv}}\sum_k v_k^- n_k\boldsymbol{n}\cdot\boldsymbol{\sigma}^-\cdot\boldsymbol{n} \\ &= -\sum_k\int_{\Gamma_{Nv}} v_k^- n_k(\boldsymbol{n}\otimes\boldsymbol{n}):\boldsymbol{\sigma}^-.\end{aligned}$$

  So the value acting on component v_i is

  $$-\int_{\Gamma_{Nv}} v_i^- n_i(\boldsymbol{n}\otimes\boldsymbol{n}):\boldsymbol{\sigma}^- = -\sum_l\sum_k\int_{\Gamma_{Nv}} v_i^- n_i n_l n_k\sigma_{lk}^- = \sum_l\sum_k\int_\Omega b_{lk}^i(v_i)\sigma_{lk}.$$

Combining these terms gives the following definition for the i-th component of the discrete divergence $\nabla \cdot \boldsymbol{\sigma}$

$$w_i = -\sum_j M^{-1} G_j^T M \sigma_{ij} - \sum_l \sum_k M^{-1} {b_{lk}^i}^T M \sigma_{lk} - A_i^{Nv}(\boldsymbol{u}) + J_{h,i}.$$

Combining this with the definition of $\boldsymbol{\sigma}$ gives the following component-wise definition of the Stokes stress equation $\boldsymbol{w} = \nabla \cdot \boldsymbol{\sigma}$ taking into account the Naiver-slip boundary condition

$$\begin{aligned} w_i = &-\sum_j M^{-1} G_j^T M_\mu (G_j u_i + \gamma G_i u_j + J_{g,ij} + \gamma J_{g,ji} + \sum_k (b_{ij}^k + \gamma b_{ji}^k)(u_k)) + M^{-1} G_i^T M p \\ &- \sum_l \sum_k M^{-1} {b_{lk}^i}^T M_\mu (G_k u_l + \gamma G_l u_k + J_{g,lk} + \gamma J_{g,kl} + \sum_m (b_{lk}^m + \gamma b_{kl}^m)(u_m)) \\ &+ \sum_k M^{-1} {b_{kk}^i}^T M p - A_i^{Nv}(\boldsymbol{u}) + J_{h,i}, \end{aligned}$$

for each $1 \leq i \leq d$.

(iv) Define the divergence constraint $w \in V_h$, $w = \nabla \cdot \boldsymbol{u}$ in the strong-weak form such that for element $E \in \mathcal{E}$

$$\int_E wv = \int_E v \nabla \cdot \boldsymbol{u} + \int_{\partial E} v(\boldsymbol{u}^* - \boldsymbol{u}) \cdot \boldsymbol{n},$$

for all test functions v on element E. Summing over all elements and using the same numerical fluxes for $\boldsymbol{u}$ as before gives effectively the trace of $\boldsymbol{\eta}$, with

$$w = \sum_j G_j u_j + \sum_j \sum_k b_{jj}^k u_k + J_{g \cdot n},$$

where $J_{g \cdot n} \in V_h$ such that $\int_\Omega J_{g \cdot n} v = \int_{\Gamma_{Nv}} g v^-$, for all $v \in V_h$.

(v) To ensure wellposedness of the discrete problem, the LDG methods employ penalty stabilization to weakly impose continuity of the polynomials between elements and to weakly impose the normal component of the Navier boundary condition. The former terms are of the form $\int_{\Gamma_0} \vartheta_0 [\boldsymbol{u}] \cdot [\boldsymbol{v}]$ for the velocity field and $\int_{\Gamma_0} \vartheta_p [p][v]$ for pressure, with corresponding operators $E_{\boldsymbol{u}}$ and E_p respectively. Here $\vartheta_0, \vartheta_p, \vartheta_{Nv}$ are positive penalty parameters. The Navier boundary penalty term is given by

$$\int_{\Gamma_{Nv}} \vartheta_{Nv} (\boldsymbol{u}^- - (g\boldsymbol{n} + \boldsymbol{u}_\tau^-)) \cdot \boldsymbol{v}^- = \int_{\Gamma_{Nv}} \vartheta_{Nv} \boldsymbol{u}_n^- \cdot \boldsymbol{v}^- - \int_{\Gamma_{Nv}} \vartheta_{Nv}\, g\boldsymbol{n} \cdot \boldsymbol{v}^-,$$

for all $\boldsymbol{v} \in V_h^d$. Define $a_{Nv} \in V_h^d$ such that

$$\int_\Omega a_{Nv} \cdot \boldsymbol{v} = \int_{\Gamma_{Nv}} \vartheta_{Nv}\, g\boldsymbol{n} \cdot \boldsymbol{v}^-,$$

Stokes form	d	polynomial degree p			
		1	2	3	4
standard	2D	0.19	0.10	0.086	0.019
	3D	0.12	0.088	0.084	
viscous stress	2D	0.14	0.046	0.034	0.0095
	3D	0.12	0.039	0.040	

Table A.1: Optimal values of pressure penalty parameter ϑ, from [72].

and define the Navier boundary penalty operator $E^{Nv} : V_h^d \to V_h^d$ such that

$$\int_\Omega E^{Nv}(\boldsymbol{u}) \cdot \boldsymbol{v} = \int_{\Gamma_{Nv}} \vartheta_{Nv} \boldsymbol{u}_n^- \cdot \boldsymbol{v}^-,$$

where the block operator E_{ij}^{Nv} is given by

$$\int_\Omega E_{ij}^{Nv}(\boldsymbol{u}) v = \int_{\Gamma_{Nv}} \vartheta_{Nv} u_j^- n_j n_i v^-,$$

for all $v \in V_h$ and for all $1 \leq i, j \leq d$. In the convergence tests of section A.4, the penalty parameters are specified as follows: the velocity penalty parameters should scale proportionally with the local viscosity coefficient μ and polynomial degree p and in inverse proportion with mesh spacing h, setting $\vartheta_0, \vartheta^{Nv} = C\mu p/h$, with $C = 0.5$ and $C = 10$ respectively. A result of [72] is that the pressure penalty parameter ϑ_p plays a key role in solution accuracy and multigrid performance. In that work $\vartheta_p = \vartheta h/\mu$, where ϑ is a user-defined penalty parameter; the optimal values of ϑ that cluster the eigenvalues of the multigrid preconditioned system around one are specified in [72]. These values are used in this work and are repeated in Table A.1 for completeness.

Summary

The LDG formulation for the Stokes problem with Navier-slip boundary conditions (A.1) involves finding the velocity field $\boldsymbol{u} \in V_h^d$ and pressure $p \in V_h$ such that

$$\begin{aligned}
&\sum_j M^{-1} G_j^T M_\mu (G_j u_i + \gamma G_i u_j + \sum_k (b_{ij}^k + \gamma b_{ji}^k)(u_k)) + \sum_l \sum_k M^{-1} {b_{lk}^i}^T M_\mu (G_k u_l + \gamma G_l u_k \\
&+ \sum_m (b_{lk}^m + \gamma b_{kl}^m)(u_m)) + A_i^{Nv}(\boldsymbol{u}) + E_{\boldsymbol{u}} u_i + E^{Nv}(\boldsymbol{u})_i - (M^{-1} G_i^T M + \sum_k M^{-1} {b_{kk}^i}^T M) p \\
&= \mathbb{P}_{V_h}(f_i) - \sum_j M^{-1} G_j^T M_\mu (J_{g,ij} + \gamma J_{g,ji}) - \sum_l \sum_k M^{-1} {b_{lk}^i}^T M_\mu (J_{g,lk} + \gamma J_{g,kl}) \\
&+ J_{h,i} + a_{Nv,i},
\end{aligned} \tag{A.3}$$

for each $0 \leq i \leq d$, subject to divergence constraint

$$-\sum_j G_j u_j - \sum_j \sum_k b_{jj}^k u_k - E_p p = \mathbb{P}_{V_h}(f_{\text{div}}) + J_{g\cdot n}, \tag{A.4}$$

where $\mathbb{P}_{V_h}(f)$ is the L^2 projection of f onto the polynomial space V_h.

Block structure

Multiplying the above two equations (A.3,A.4) by the mass matrix M and putting them into block form gives

$$\begin{pmatrix} A & M\mathscr{G} \\ \mathscr{G}^T M & -ME_p \end{pmatrix} \begin{pmatrix} \boldsymbol{u} \\ p \end{pmatrix} = \begin{pmatrix} \boldsymbol{s} \\ s_{\text{div}} \end{pmatrix}. \tag{A.5}$$

The degrees of freedom for $\boldsymbol{u}$ and p are blocked together on an element-wise basis and the block-sparse linear system may be written as $\mathscr{A}x = s$, where $x = (\boldsymbol{u}, p)$ and $s = (\boldsymbol{s}, s_{\text{div}})$. The resulting Stokes operator $\mathscr{A}$ is symmetric indefinite and the system (A.5) is solved via operator coarsening multigrid-preconditioned GMRES methods [72], discussed in the next section. The components of linear system (A.5) are

- $\mathscr{G} = (\mathscr{G}_1, ..., \mathscr{G}_d)$ is a discrete gradient operator given by

  $$\mathscr{G}_i = -M^{-1} G_i^T M - \sum_k M^{-1} {b_{ii}^k}^T M.$$

  The divergence constraint is enforced in the lower left corner of (A.5) by the application of a discrete divergence operator that is the negative adjoint of this discrete gradient operator.

- A is a $d \times d$ block matrix, with block (i, j) given by

  $$A_{ij} = \delta_{ij} \left(\sum_k G_k^T M_\mu G_k \right) + \gamma G_j^T M_\mu G_i + \delta_{ij} M E_{\boldsymbol{u}} + \mathcal{BG}_{ij} + \mathcal{GB}_{ij} + \mathcal{BB}_{ij} + M A_{ij}^{Nv} + M E_{ij}^{Nv}.$$

  The first three terms in the definition of A_{ij} represent the base viscous component of the Stokes operator and the final five terms represent the contribution from the Navier boundary. Here operators $\mathcal{BG}, \mathcal{GB}, \mathcal{BB}$ are defined by

  $$\begin{aligned} \mathcal{BG}_{ij} &= \sum_k \left({b_{kj}^i}^T M_\mu G_k + \gamma {b_{jk}^i}^T M_\mu G_k \right) \\ \mathcal{GB}_{ij} &= \sum_k G_k^T M_\mu (b_{ik}^j + \gamma b_{ki}^j) \\ \mathcal{BB}_{ij} &= \sum_l \sum_k {b_{lk}^i}^T M_\mu (b_{lk}^j + \gamma b_{kl}^j). \end{aligned}$$

- The terms $(\boldsymbol{s}, s_{\text{div}})$ combine the right-hand sides of the Stokes momentum equation and divergence constraint respectively, including the source and boundary data.

A.3 Operator coarsening multigrid

To solve the linear system (A.5), we extend the operator coarsening geometric multigrid methods of [72][78] to the LDG methods developed in section A.2 for Stokes problems with Naiver-slip boundary conditions. Recall from section 3.4, that geometric multigrid methods act by reducing high-frequency eigenmodes in the error along a hierarchy of successively coarsened meshes. The operator coarsening paradigm coarsens LDG operators individually, maintaining PDE and boundary condition consistency throughout each level of the hierarchy while also sharply preserving the interfaces and avoiding the explicit construction of coarse meshes. The multigrid methods of this section use the same (a) quad/octree mesh hierarchy, (b) mesh-to-mesh transfer operators, and (c) damped block Gauss-Seidel smoother as in section 3.4.

Recall the definition (eq. (3.32)) of the coarsening operator $\mathcal{C}(A) : V_c \to V_c$ for the fine mesh operator $A : V_f \to V_f$:

$$\mathcal{C}(A) = M_c^{-1}(I_c^f)^T M_f A I_c^f,$$

where M_f and M_c are the fine and coarse mesh mass matrices respectively and $I_c^f : V_c \to V_f$ is the interpolation operator. Within the operator coarsening framework, the LDG operators of the Stokes system are coarsened individually and then recombined at each level of the hierarchy, with the coarse mesh LDG operators defined as

$$\begin{aligned}
M_c &= (I_c^f)^T M_f I_c^f \\
M_{\mu,c} &= (I_c^f)^T M_{\mu,f} I_c^f \\
G_c &= M_c^{-1}(I_c^f)^T M_f G_f I_c^f \\
b_c &= M_c^{-1}(I_c^f)^T M_f b_f I_c^f \\
E_{\boldsymbol{u},c} &= \frac{1}{2}(I_c^f)^T E_{\boldsymbol{u},f} I_c^f \\
E_c^{Nv} &= \frac{1}{2}(I_c^f)^T E_f^{Nv} I_c^f \\
E_{p,c} &= 2(I_c^f)^T E_{p,f} I_c^f \\
A_c^{Nv} &= (I_c^f)^T A_f^{Nv} I_c^f,
\end{aligned}$$

where the factor of half is applied to the coarsened penalty operators $E_{\boldsymbol{u}}, E^{Nv}$ and a factor of two is applied to E_p to maintain the penalties parameters' respective $1/h$ and h scaling throughout the multigrid hierarchy [80]. Recombining the coarse mesh LDG operators gives the following coarse mesh block-form Stokes operator

$$\mathscr{A}_c = \begin{pmatrix} A_c & M_c\mathscr{G}_c \\ \mathscr{G}_c^T M_c & -M_c E_{p,c} \end{pmatrix},$$

where

$$\mathscr{G}_{c,i} = -M_c^{-1} G_{c,i}^T M_c - \sum_k M_c^{-1} {b_{c,ii}^k}^T M_c,$$

and

$$A_{c,ij} = \delta_{ij}\left(\sum_k G_{c,k}^T M_{\mu,c} G_{c,k}\right) + \gamma G_{c,j}^T M_{\mu,c} G_{c,i} + \delta_{ij} M E_{\boldsymbol{u},c} + \mathcal{BG}_{c,ij} + \mathcal{GB}_{c,ij} + \mathcal{BB}_{c,ij} + M_c A_{c,ij}^{Nv} + M_c E_{c,ij}^{Nv},$$

with operators $\mathcal{BG}_c, \mathcal{GB}_c, \mathcal{BB}_c$ defined by

$$\mathcal{BG}_{c,ij} = \sum_k \left({b_{c,kj}^i}^T M_{\mu,c} G_{c,k} + \gamma {b_{c,jk}^i}^T M_{\mu,c} G_{c,k}\right)$$
$$\mathcal{GB}_{c,ij} = \sum_k G_{c,k}^T M_{\mu,c} (b_{c,ik}^j + \gamma b_{c,ki}^j)$$
$$\mathcal{BB}_{c,ij} = \sum_l \sum_k {b_{c,lk}^i}^T M_{\mu,c} (b_{c,lk}^j + \gamma b_{c,kl}^j).$$

The multigrid V-cycle algorithm for linear system $\mathscr{A}x = s$ is as in section 3.4. To further accelerate performance, the multigrid algorithms are applied as preconditioners for Krylov subspace methods. Krylov subspace methods work well to weakly impose the boundary conditions, while multigrid solves the interior elliptic problem. For the Stokes problem (A.5), a single V-cycle is applied as a left preconditioner to the GMRES method. For more information on preconditioned Krylov subspace methods, see [87].

A.4 Convergence tests

In this section, we test the order of accuracy of the developed LDG methods for the Stokes problem with Navier-slip boundary conditions (A.1). In particular, we demonstrate high-order accuracy for the challenging case of variable viscosity and slip coefficients spanning several orders of magnitude on an implicitly-defined curved domain, in both 2D and 3D. Convergence of the computed solution is tested against an exact offset sinusoid solution, setting $\boldsymbol{u} : \Omega \to \mathbb{R}^d$ and $p : \Omega \to \mathbb{R}$ to

$$u_i(x) = \prod_{j=1}^{d} \sin 2\pi(x_j - 0.2i), \tag{A.6}$$

$$p(x) = \mu_0 \prod_{j=1}^{d} \sin 2\pi(x_j + 0.2), \tag{A.7}$$

where $0.2i$ is a component-dependent shift and μ_0 is the typical viscosity coefficient for the problem. The exact solution is used to generate the source data $\boldsymbol{f}, f_{\text{div}}$ and the boundary data $g, \boldsymbol{h}$. We test our formulation within the irregular "amoeba" domain of section 3.5, shown in Figure 3.3, which is embedded within a unit $[0,1]^d$ box.

Note that the time-independent Stokes problem with Navier-slip boundary conditions has an associated kernel in the pressure field that is the span of globally constant functions. We nullify this trivial kernel by considering only mean-zero pressure fields, meaning that throughout the multigrid hierarchy, the following constant is removed from the computed pressure solution:

$$\frac{(1,p)}{(1,1)} = \frac{1^T M p}{1^T M 1},$$

where $(\cdot,\cdot)$ is the standard inner product, M is the mass matrix, and 1 represents the unit function. Additionally, we note that within certain geometries, a trivial kernel may also appear within the fluid velocity field. For example, consider the Stokes problem (A.1) on a simple 2D square domain, with periodic boundary conditions along the horizontal x-axis and Navier boundary conditions along the top and bottom walls. Let $\boldsymbol{u} = (u, v)$. Within this domain specification, the first Navier boundary condition $\boldsymbol{u}\cdot\boldsymbol{n} = g$ provides information only for the vertical fluid velocity v, and subsequently, the time-independent Stokes system (A.1) has an associated kernel that is the span of globally constant functions within the horizontal fluid velocity u. This kernel may be treated in a similar mean-zeroing manner, however, we note that this trivial velocity kernel in the fluid velocity field is not present within the test cases considered here.

Test 1: Constant diffusion and slip coefficients: We begin with perhaps the simplest non-trivial Navier-slip boundary condition, setting unit viscosity and slip coefficients: $\mu = \alpha = 1$. Figure A.1 shows the computed L^2 and L^∞ errors against the exact solution and demonstrates optimal $p+1$ order accuracy in the fluid velocity field, in both 2D and 3D. The pressure field loses a half order of accuracy in the L^2 norm and a full order of accuracy in the L^∞ norm. The loss of order is due to numerical boundary layers within the pressure field and is consistent with the findings of [72] for Stokes problems with other boundary condition types. This boundary layer does not affect the accuracy of the computed fluid velocity. We note the results of this test in Figure A.1 are similar for the free-slip condition $\alpha = 0$.

Test 2: Variable viscosity and slip coefficients spanning several orders of magnitude
Next, we examine a challenging case with variable coefficients, where the viscosity varies by four orders of magnitude and the slip coefficient by eight orders of magnitude throughout the domain, setting

in 2D:	in 3D:
$\mu = 10^{2\sin(2\pi(x-0.1))\sin(2\pi(y+0.1))}$	$\mu = 10^{2\sin(2\pi(x-0.1))\sin(2\pi(y+0.1))\sin(2\pi(z-0.1))}$
$\alpha = 10^{-4+8\sin(\pi x/2)\sin(\pi y/2)}$	$\alpha = 10^{-4+8\sin(\pi x/2)\sin(\pi y/2)\sin(\pi(z+0.5)/2)}$.

In this setting, additional speedups in multigrid performance can be achieved by a suitable rescaling of the discrete Stokes operator, effectively so that it has unit viscosity and unit length scale. Here we apply diagonal pre- and post-scalings to the Stokes operator, replacing (A.5) with

$$\begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix} \begin{pmatrix} A & M\mathscr{G} \\ \mathscr{G}^T M & -ME_p \end{pmatrix} \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix},$$

where α and β are diagonal matrices with entries equal to $\sqrt{L/\mu}$ and $\sqrt{\mu/L}$ respectively, with L being the characteristic length scale of the problem. The scaled problem $\tilde{\mathscr{A}}\tilde{x} = \tilde{s}$ is then solved, where $\tilde{\mathscr{A}}$ is the scaled Stokes operator and $\tilde{s} = \text{diag}(\alpha, \beta)s$, after-which the original unscaled solution $x = \text{diag}(\alpha, \beta)\tilde{x}$ is computed. We note that this scaling is applied at each level of the multigrid hierarchy in the operator coarsening framework.

The LDG method produces high-order accurate solutions even in this challenging setting, achieving optimal $p + 1$ order accuracy for the fluid velocity in both the L^2 and L^∞ norms. A similar loss of order in the pressure field occurs, as in the previous result. The convergence results are illustrated in Figure A.2.

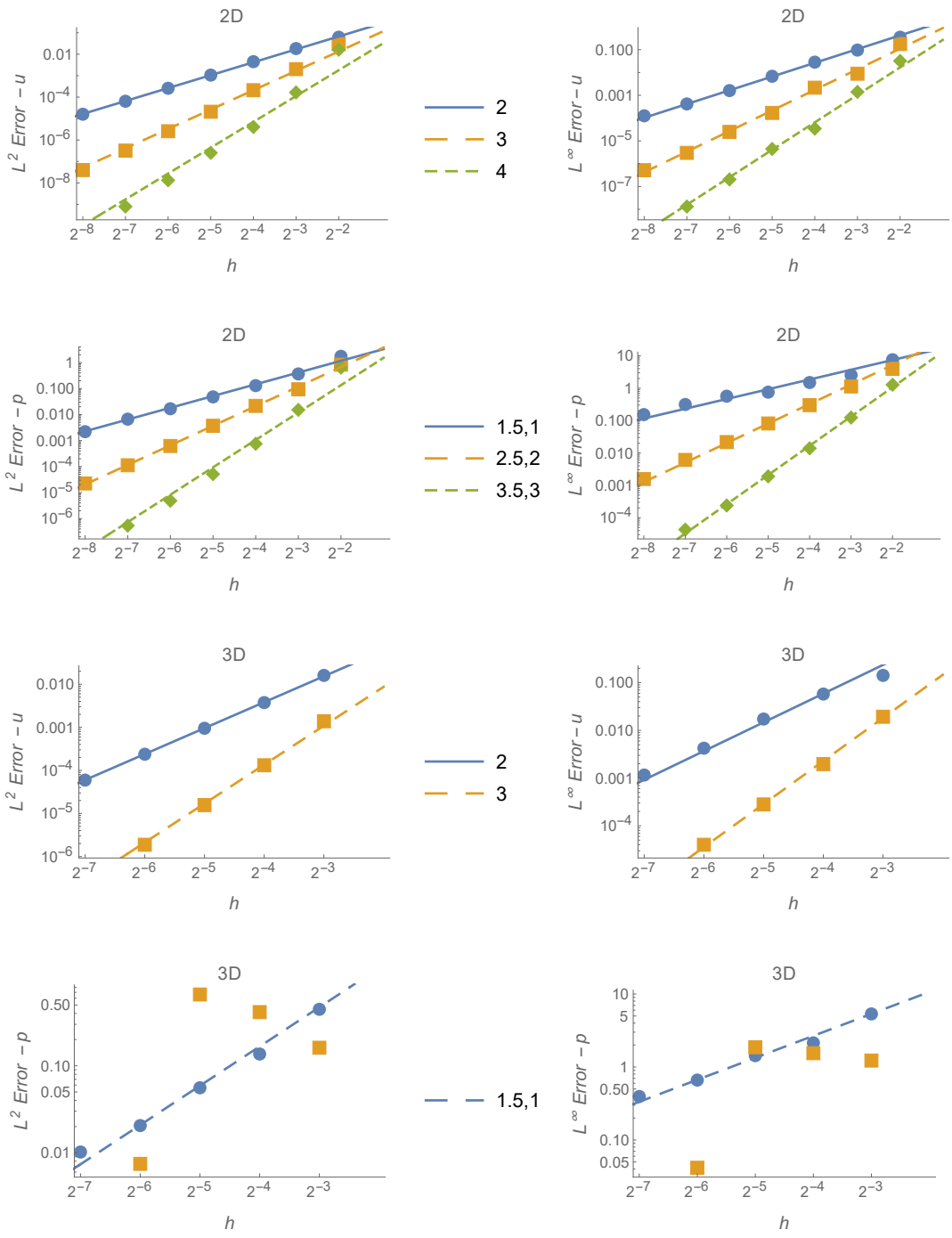


Figure A.1: Convergence rates for the Stokes problem with Navier-slip boundary conditions on an implicitly-defined curved domain with $\mu, \alpha = 1$. h denotes the background mesh spacing and polynomial degrees are represented by $\bullet, \blacksquare, \blacklozenge$ for $p = 1, 2, 3$ respectively, with the slopes of the lines indicating asymptotic convergence rates. Slope indicators with two comma-separated values correspond to the left and right columns respectively.

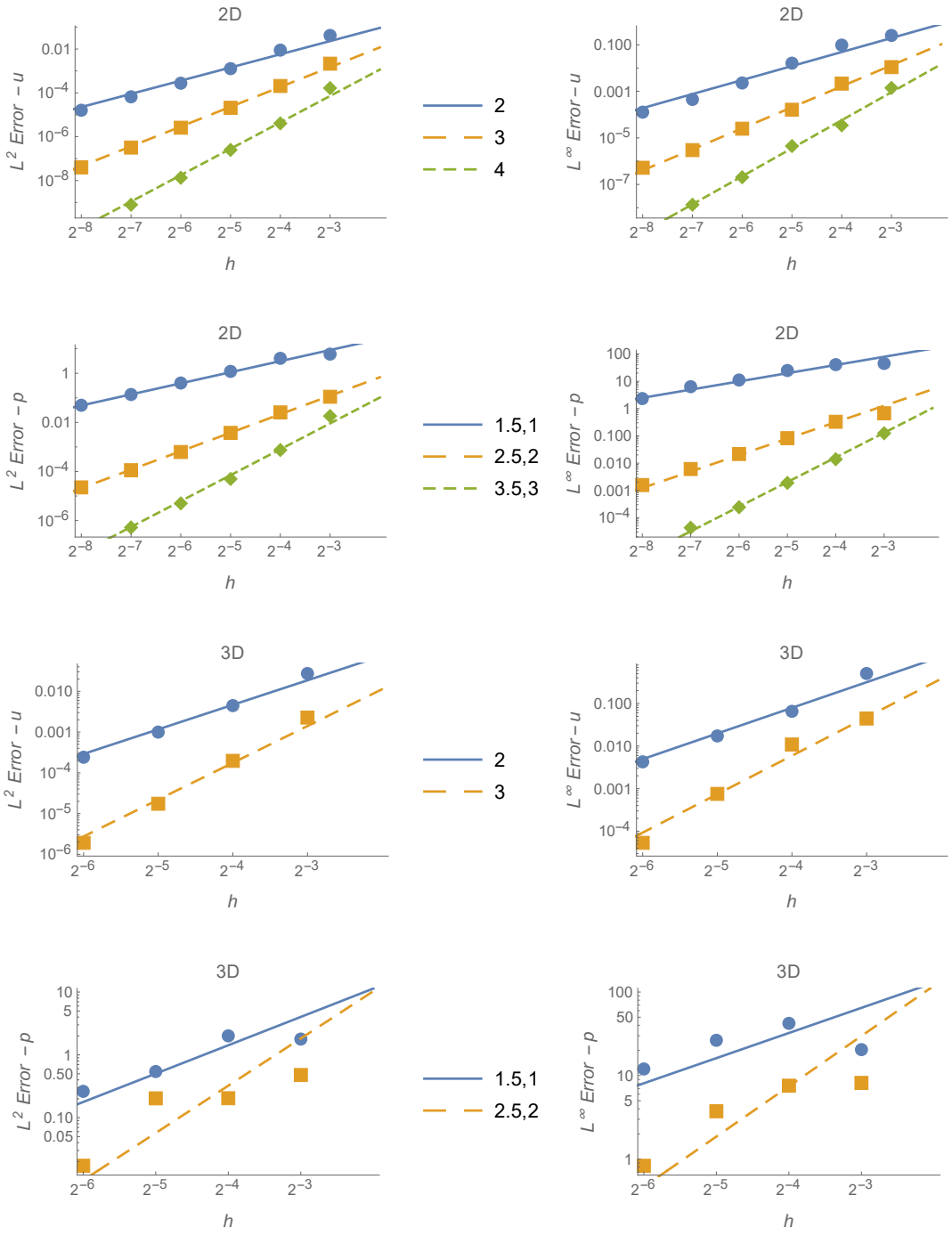


Figure A.2: Convergence rates for the Stokes problem with Navier-slip boundary conditions on an implicitly-defined curved domain with variable μ, α spanning several orders of magnitude. h denotes the background mesh spacing and polynomial degrees are represented by $\bullet, \blacksquare, \blacklozenge$ for $p = 1, 2, 3$ respectively, with the slopes of the lines indicating asymptotic convergence rates. Slope indicators with two comma-separated values correspond to the left and right columns respectively.

Appendix B

Physical Parameters

Laboratory experiments by PPG Industries Inc., along with standard reference values, specify many of the physical parameters involved in the multi-layer coating flow problem. Figure B.1 provides the range of PPG provided values for paint viscosity and surface tension with respect to the resin mass concentration, including regular/experimental, high, and low values. The figure indicates that the rheological parameters vary substantially over time as the paints dry; consequently, the numerical framework must handle a wide parameter range. Note the sharp exponential nature of the viscosity profile with respect to resin concentration acts to model the solidification of the paint.

Some parameters of the multi-layer coating flow problem are difficult to determine experimentally, such as the solvent mass diffusion coefficients, while for other parameters, little is known in general—e.g., the precise nature of embedded paint-paint surface tension forces. For these values, we rely on the physical intuition of prior work and are also guided by the results of our numerical simulations. For example, the mass diffusion coefficients for different materials can range over four orders of magnitude, anywhere from $10^{-14} - 10^{-10}$ $\mathrm{m}^2\ \mathrm{s}^{-1}$. In particular, as diffusion rates decrease, the thickness of solvent boundary layers decreases, meaning the numerical resolution must increase to correctly and accurately resolve the strong Marangoni forces at the interface. The smallest value of clearcoat diffusion resolvable by our current simulations was 10^{-12} $\mathrm{m}^2\ \mathrm{s}^{-1}$. Values of embedded paint-paint surface tension coefficients are taken to be within the range of the coefficients of paint-gas surface tension and, for some tests, an order of magnitude smaller.

The typical range of values for the paint parameters used in this work is stated below. The specific rheological values for each simulation are given in Table B.1.

- Fluid density: $\rho = 1000$ kg m^{-3}, approximately that of water.
- Basecoat viscosity: $\mu_{BC} \approx 20 - 500$ kg $\mathrm{m}^{-1}\ \mathrm{s}^{-1}$, often taken as the PPG-provided high-range viscosity.
- Clearcoat viscosity: $\mu_{CC} \approx 1 - 100$ kg $\mathrm{m}^{-1}\ \mathrm{s}^{-1}$, often from PPG provided experimental

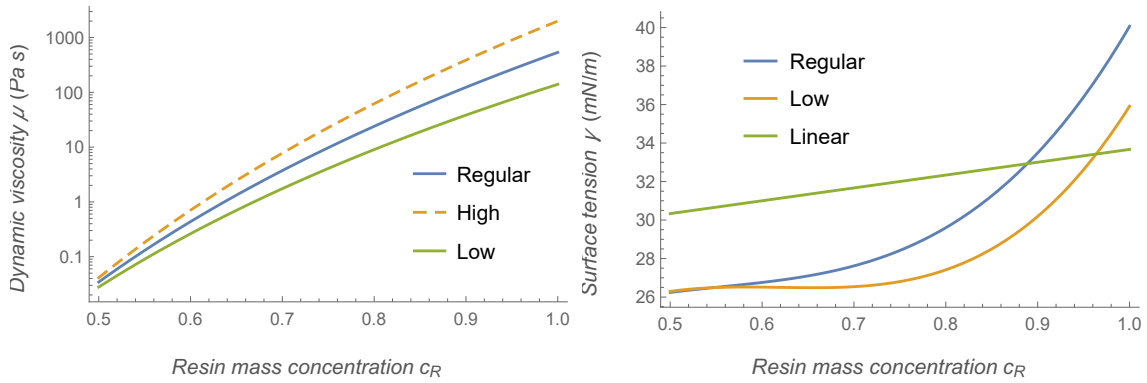


Figure B.1: PPG provided viscosity and surface tension values with respect to resin concentration.

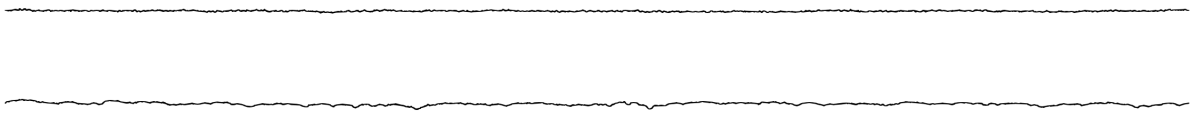


Figure B.2: The PPG provided smooth $\mathcal{S}_{\text{smooth}}$ (top) and rough $\mathcal{S}_{\text{rough}}$ (bottom) substrate profiles. The horizontal x-direction has been scaled by 1/40 to illustrate the features.

data.

- Paint-gas surface tension: $\gamma \approx 0.02 - 0.04$ N m^{-1}, see Figure B.1. Linear surface tension $= 27 + 6.67 c_R$ mN m^{-1}.
- Paint-paint surface tension: $\gamma_{ij} \approx 0.001 - 0.04$ N m^{-1}.
- Initial basecoat resin concentration: $c_{R,0}^{BC} = 0.75$.
- Initial clearcoat resin concentration: $c_{R,0}^{CC} = 0.65$.
- Solvent diffusion coefficient $D \approx 10^{-12} - 10^{-10}$ m^2 s^{-1}.
- Evaporation rate: $m \approx 10^{-5} - 10^{-4}$ kg m^{-2} s^{-1}.
- Gravity: $g \approx 9.8$ m s^{-2}.

The typical length and time scales are:

- Initial basecoat film thickness: $h_{BC} \approx 30 - 50$ μm.
- Initial clearcoat film thickness: $h_{CC} \approx 50 - 100$ μm.
- Horizontal length scale: tens of millimeters.
- Flash period: four to ten minutes.

Substrate profile

In addition to specifying some of the physical parameters, PPG also conducted laboratory experiments to determine the profile of two typical substrates, seen in Figure B.2. Correspondingly, we consider in this work three substrate profiles:

- $\mathcal{S}_{\text{flat}}$ – Perfectly flat substrate, used as a baseline.
- $\mathcal{S}_{\text{smooth}}$ – Mildly-rough substrate, with Fourier amplitudes up to 0.15μm.
- $\mathcal{S}_{\text{rough}}$ – Rough substrate, with Fourier amplitudes up to 0.45μm.

Parameter (units)	Symbol	Fig. 4.3	Fig. 5.1	Fig. 5.3	Fig. 5.4	Fig. 5.7
Number of solvents	C	1	1	1	1	3
Number of layers		2	1	2	2	1
Substrate/gravity		$\mathcal{S}_{\text{flat}}$	$\mathcal{S}_{\text{flat}}$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$
Mesh resolution		128	32	32	32	64
Domain width (μm)		400	400	100	100	400
Init. height Γ_e (μm)		90	90	90	90	50
Init. height Γ_{ij} (μm)		35	—	35	35	—
Fluid density (kg/m^3)	ρ	1000	1000	1000	1000	1000
Basecoat viscosity ($Pa \cdot s$)	μ_{BC}	4	—	Visc. High	Visc. High	—
Clearcoat viscosity ($Pa \cdot s$)	μ_{CC}	2	2	Visc. Reg	Visc. Reg	Visc. Reg
Basecoat diffusion (m^2/s)	D_{BC}	2×10^{-12}	—	1×10^{-12}	1×10^{-12}	—
Clearcoat diffusion (m^2/s)	D_{CC}	1×10^{-11}	1×10^{-11}	5×10^{-12}	4×10^{-12}	5×10^{-12}
Surface tension Γ_e (mN/m)	γ_e	$50 - 25c_R$	$10 + 25c_R$	St. Lin	St. Lin	St. Lin
Surface tension Γ_{ij} (mN/m)	γ_{ij}	30	—	3	3	—
Evaporation coeff ($kg/(m^2 \cdot s)$)	ε	1×10^{-3}	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}	$3.33, 5.0, 7.67 \times 10^{-4}$
Init. basecoat resin cons	c_R^{BC}	As specified	—	0.75	0.75	—
Init. clearcoat resin cons	c_R^{CC}	As specified	0.65	0.65	0.65	0.65

Parameter (units)	Symbol	Fig. 5.8/C.1	Study 1	Study 2	Study 3	Study 4
Number of solvents	C	3	1	1	1	1
Number of layers		1	2	2	2	2
Substrate/gravity		$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{smooth}}\downarrow$
Mesh resolution		64	32	32	32	32
Domain width (μm)		400	25600	25600	25600	25600
Init. height Γ_e (μm)		50	90	90	90	90
Init. height Γ_{ij} (μm)		—	35	35	35	35
Fluid density (kg/m^3)	ρ	1000	1000	1000	1000	1000
Basecoat viscosity ($Pa \cdot s$)	μ_{BC}	—	Visc. High	Visc. High	Visc. High	Visc. High
Clearcoat viscosity ($Pa \cdot s$)	μ_{CC}	Visc. Reg	Visc. Reg	Visc. Reg	Visc. Reg	Visc. Reg
Basecoat diffusion (m^2/s)	D_{BC}	—	1×10^{-12}	1×10^{-12}	1×10^{-12}	1×10^{-12}
Clearcoat diffusion (m^2/s)	D_{CC}	$1.25, 2.5, 5.0 \times 10^{-12}$	5×10^{-12}	5×10^{-12}	5×10^{-12}	5×10^{-12}
Surface tension Γ_e (mN/m)	γ_e	St. Lin	St. Reg	St. Reg	St. Reg	St. Reg
Surface tension Γ_{ij} (mN/m)	γ_{ij}	—	1.5	∼4-8	30	30
Evaporation coeff ($kg/(m^2 \cdot s)$)	ε	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}
Init. basecoat resin cons	c_R^{BC}	—	0.75	0.75	0.75	0.75
Init. clearcoat resin cons	c_R^{CC}	0.65	0.65	0.65	0.65	0.65

Parameter (units)	Symbol	Study 5	Study 6	Study 7	Study 8
Number of solvents	C	1	1	1	1
Number of layers		2	2	2	2
Substrate/gravity		$\mathcal{S}_{\text{flat}}$, $\mathcal{S}_{\text{rough}}\rightarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\downarrow$	$\mathcal{S}_{\text{flat}}\rightarrow$
Mesh resolution		32	32	32	32
Domain width (μm)		25600	25600	25600	25600
Init. height Γ_e (μm)		90	90	90	90
Init. height Γ_{ij} (μm)		35	35	35	35
Fluid density (kg/m^3)	ρ	1000	1000	1000	1000
Basecoat viscosity ($Pa \cdot s$)	μ_{BC}	Visc. High	Visc. High	Visc. High	Visc. High
Clearcoat viscosity ($Pa \cdot s$)	μ_{CC}	Visc. Reg	Visc. High	Visc. Low	Visc. Reg
Basecoat diffusion (m^2/s)	D_{BC}	1×10^{-12}	1×10^{-12}	1×10^{-12}	1×10^{-12}
Clearcoat diffusion (m^2/s)	D_{CC}	5×10^{-12}	5×10^{-12}	5×10^{-12}	5×10^{-12}
Surface tension Γ_e (mN/m)	γ_e	St. Reg	St. Reg	St. Reg	St. Reg, $\frac{\nabla_S \gamma}{2}$
Surface tension Γ_{ij} (mN/m)	γ_{ij}	30	30	30	30
Evaporation coeff ($kg/(m^2 \cdot s)$)	ε	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}	3.33×10^{-4}
Init. basecoat resin cons	c_R^{BC}	0.75	0.75	0.75	0.75
Init. clearcoat resin cons	c_R^{CC}	0.65	0.65	0.65	0.65

Parameter (units)	Symbol	Fig. 5.18	Fig. 5.19	Fig. C.2/ C.3
Number of solvents	C	1	1	1
Number of layers		3	3	1
Substrate/gravity		$\mathcal{S}_{\text{flat}}$	$\mathcal{S}_{\text{flat}}$	$\mathcal{S}_{\text{flat}}$
Mesh resolution		64	64	64
Domain width (μm)		400	400	400
Init. height Γ_e (μm)		180	180	90/180
Init. height Γ_{ij} (μm)		35,90	35,90	—
Fluid density (kg/m^3)	ρ	1000	1000	1000
Basecoat viscosity ($Pa \cdot s$)	μ_{BC}	8,4	4,2	
Clearcoat viscosity ($Pa \cdot s$)	μ_{CC}	2	1	Visc. Reg
Basecoat diffusion (m^2/s)	D_{BC}	$1.25, 2.5 \times 10^{-12}$	$1.25, 2.5 \times 10^{-12}$	—
Clearcoat diffusion (m^2/s)	D_{CC}	5×10^{-12}	5×10^{-12}	1.25×10^{-12}
Surface tension Γ_e (mN/m)	γ_e	$10 + 25c_R$	$10 + 25c_R$	St. Lin, $\frac{\nabla_S \gamma}{2}$
Surface tension Γ_{ij} (mN/m)	γ_{ij}	$10 + 25c_R$	$10 + 25c_R$	—
Evaporation coeff ($kg/(m^2 \cdot s)$)	ε	0	0	3.33×10^{-4}
Init. basecoat resin cons	c_R^{BC}	0.8,1	0.8,1	—
Init. clearcoat resin cons	c_R^{CC}	1	1	0.65

Table B.1: The rheological parameters used in the numerical simulations. Note the mesh resolution specifies the number of cells of the DG background mesh per 100μm block. The arrow by the substrate profiles specifies the direction of gravity, e.g., $\downarrow$ specifies a horizontally-oriented substrate with gravity pulling downwards. No arrow specifies that gravity is not considered in the simulation.

Appendix C

Additional Images

In the results presented in Chapter 5, the coating flow dynamics were found to vary dramatically with respect to the solvent mass diffusion coefficient—owing to the complex interplay of evaporation, diffusion, and Marangoni forces along the boundary layer. In particular, the multi-solvent evaporation study of Figure 5.8 illustrates the dramatic impact of the diffusion coefficient on the short-wave Marangoni plume structures. The low diffusion solvent exhibits tighter formations and thinner boundary layers when compared to the higher diffusion solvents; the vortices are almost "tree" like in nature, with the smaller forming plumes merging to form "branches" of the tree. Such low diffusion Marangoni flows lead to the Marangoni "flower" pattern of Figure 6.1.

In this section, we present additional images of evaporating Marangoni flow, exploring the formation of tree structures within low diffusion Marangoni plumes. The mass diffusion coefficient is set to 1.25×10^{-12} m^2 s^{-1}; this is about the smallest value of clearcoat diffusion resolvable by our current simulations. Figure C.1 highlights the evolution of the low diffusion solvent from the multi-solvent evaporation study of Figure 5.8. Figures C.2 and C.3 examine the low diffusion plume structures within a single solvent paint, considering different initial film thicknesses. The rheological parameters are as described in Appendix B. All simulations use 64×64 background DG cells per 100μm block to resolve the thin boundary layer dynamics associated with low diffusion Marangoni flows.

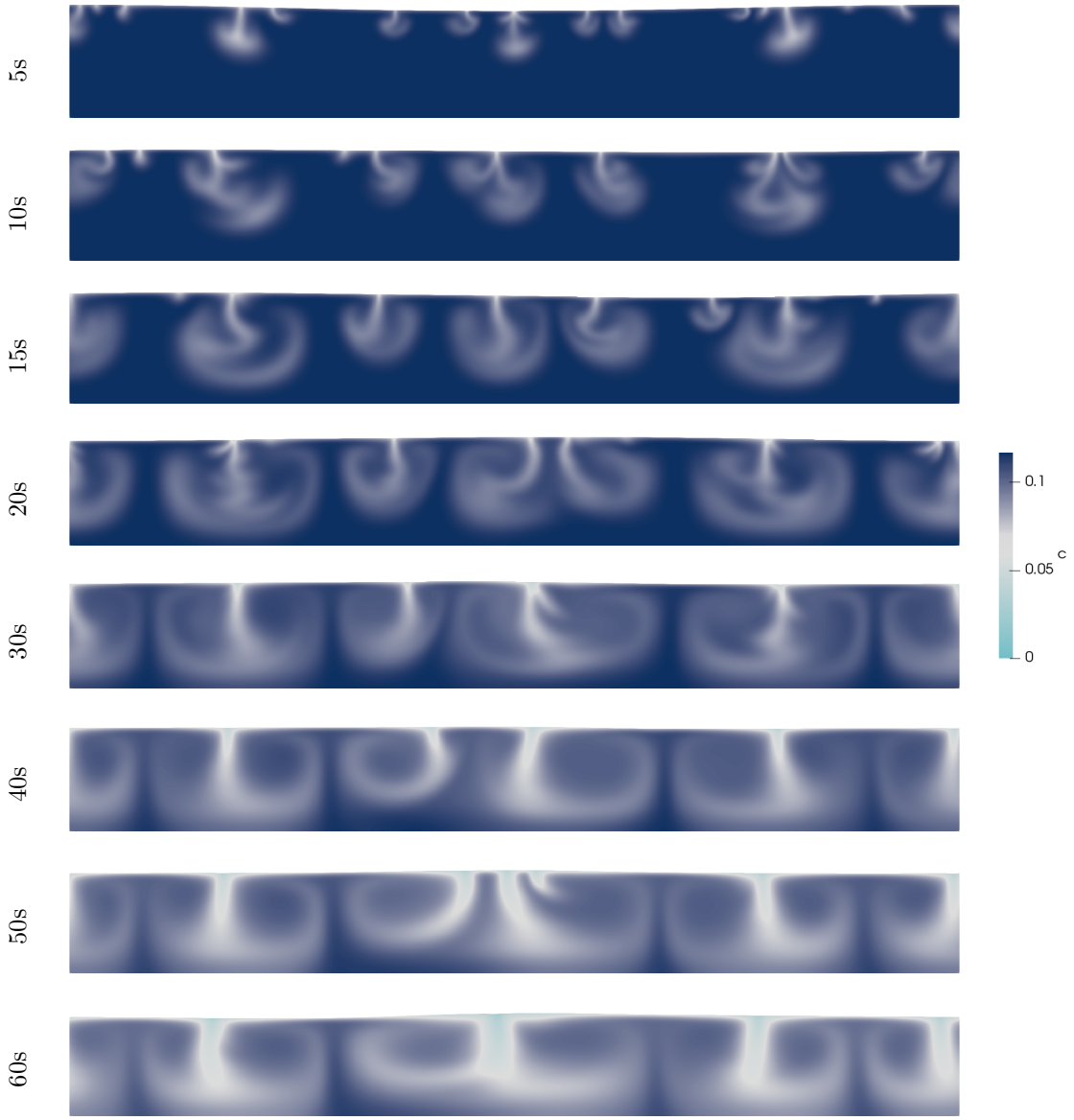


Figure C.1: The mass concentration profile of the low diffusion solvent from the multi-solvent evaporation study shown in Figure 5.8. These results consider an initial film thickness of 50μm.

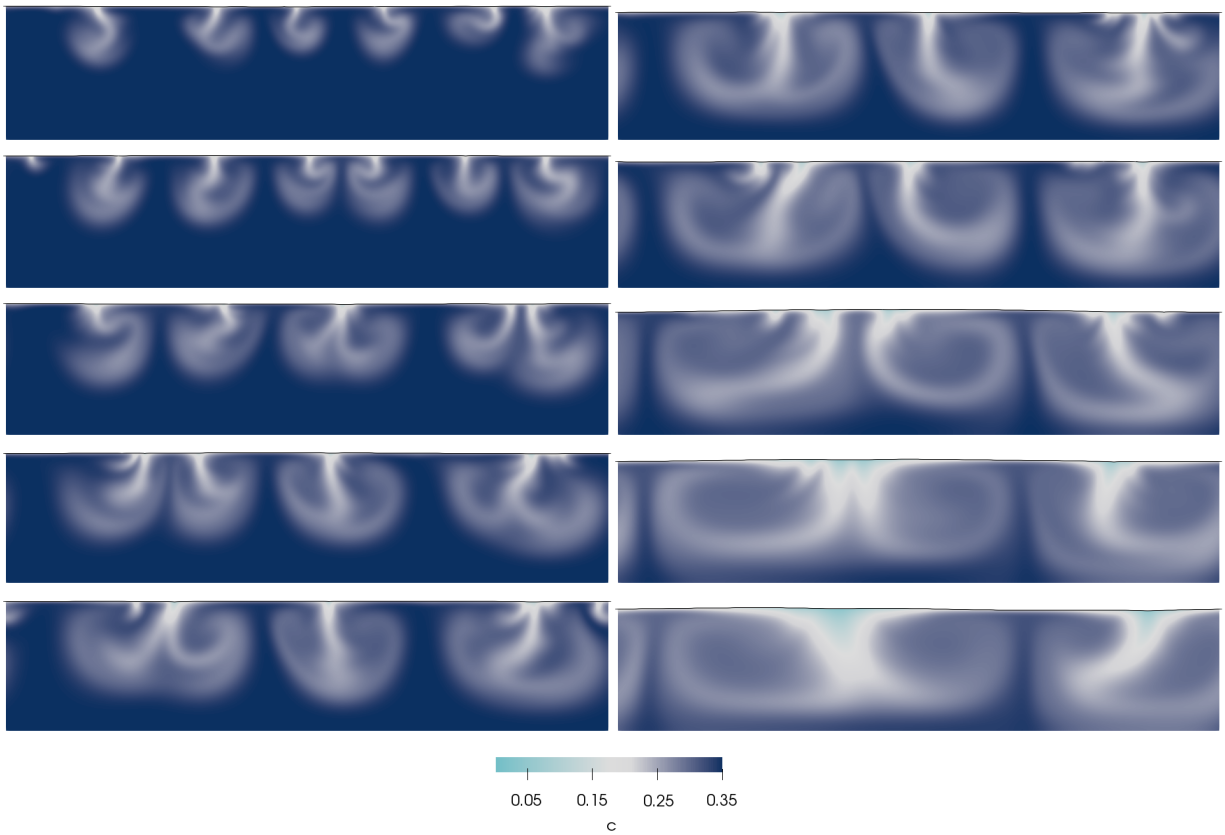
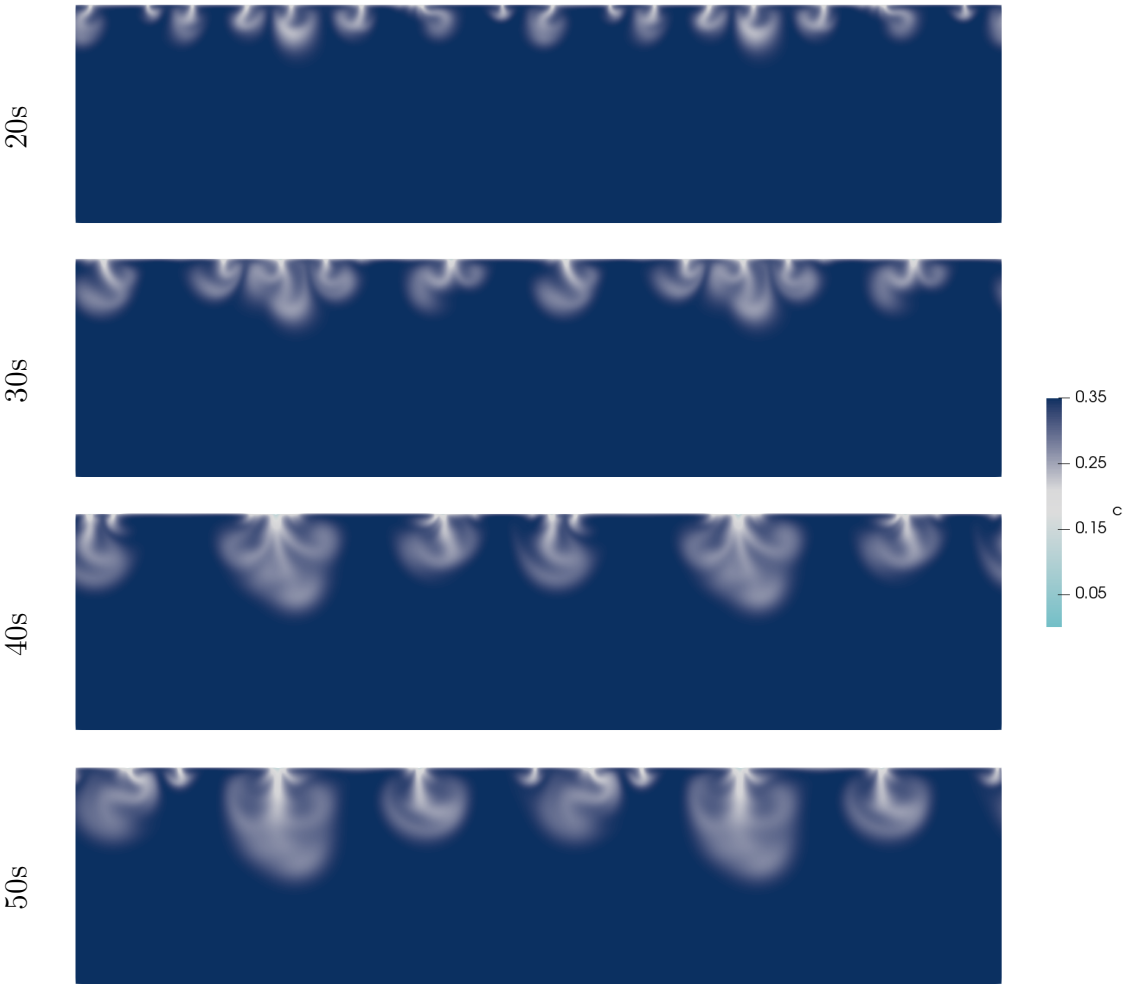


Figure C.2: The mass concentration profile of a low diffusion solvent. The results are shown at **Left:** $20, 30, 40, 50, 60$s; **Right:** $70, 80, 100, 120, 140$s. These results consider an initial film thickness of 90μm.


20s
30s
40s
50s
0.35
0.25
0.15
0.05
c


100s
110s
120s
130s
0.35
0.25
0.15
0.05
c

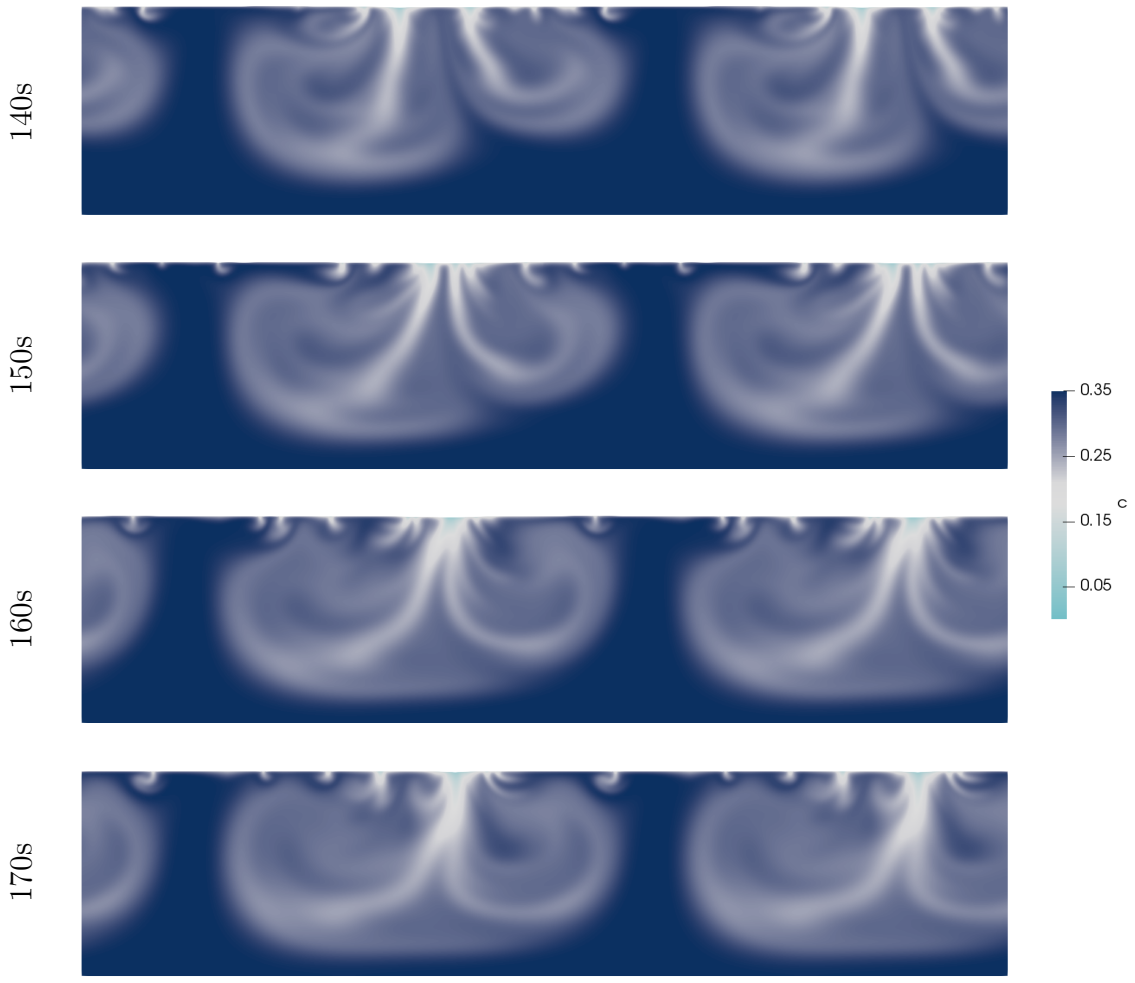


Figure C.3: The mass concentration profile of a low diffusion solvent. These results consider an initial film thickness of 180μm. For illustration, the domain is repeated across its periodic axis.