High Order Schemes for Gradient Flows

by

Alexander James Zaitzeff

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Applied and Interdisciplinary Mathematics)
in The University of Michigan
2020

Doctoral Committee:

Professor Selim Esedoḡlu, Co-Chair
Professor Krishna Garikipati, Co-Chair
Professor Smadar Karni
Arthur F Thurnau Professor Robert Krasny

Alexander J Zaitzeff

azaitzef@umich.edu

ORCID iD: 0000-0001-9731-1873

© Alexander J Zaitzeff 2020

ACKNOWLEDGEMENTS

Chapter II, chapter IV, and chapter V are papers that I wrote with Professor Selim Esedoḡlu and Professor Krishna Garikipati, both of whom helped write and edit the manuscripts. Additionally, my thesis abstract is based on the abstract of those papers. Professor Selim Esedoḡlu conceptualized many of the ideas that germinated into the theoretical results of this thesis. I gratefully acknowledge support from the NSF grant DMS-1719727. I also acknowledge the support of Benjamin Butler, who helped many of my IT issues for the computational aspects of my thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

ABSTRACT

First, two new classes of energy stable, high order accurate Runge-Kutta schemes for gradient flows in a very general setting are presented: a class of fully implicit methods that are unconditionally energy stable and a class of semi-implicit methods that are conditionally energy stable. The new schemes are developed as high order analogs of the minimizing movements approach for generating a time discrete approximation to a gradient flow by solving a sequence of optimization problems. In particular, each step entails minimizing the associated energy of the gradient flow plus a movement limiter term that is, in the classical context of steepest descent with respect to an inner product, simply quadratic. A variety of existing stable numerical methods can be recognized as (typically just first order accurate in time) minimizing movement schemes for their associated evolution equations, already requiring the optimization of the energy plus a quadratic term at every time step. Therefore, our methods give a painless way to extend the existing schemes to high order accurate in time schemes while maintaining their stability. Additionally, we extend the schemes to gradient flows with solution dependent inner product. Here, the stability and consistency conditions of the methods are given and proved, specific examples of the schemes are given for second and third order accuracy, and convergence tests are performed to demonstrate the accuracy of the methods.

Next, two algorithms for simulating mean curvature motion are considered. First is the threshold dynamics algorithm of Merriman, Bence, and Osher. The algorithm is only first order accurate in the two-phase setting and its accuracy degrades further to half order in the multi-phase setting, a shortcoming it has in common with other related, more recent algorithms. As a first, rigorous step in addressing this shortcoming, two different second order accurate versions of two-phase threshold dynamics are presented. Unlike in previous efforts in this direction, both algorithms come with careful consistency calculations. The first algorithm is consistent with its limit (motion by mean curvature) up to second order in any space dimension. The second achieves second order accuracy only in dimension two but comes with a rigorous stability guarantee (unconditional energy stability) in any dimension – a first for high order schemes of its type.

Finally, a level set method for multiphase curvature motion known as Voronoi implicit interface method is considered. Here, careful numerical convergence studies, using parameterized curves to reach very high resolutions in two dimensions are given. These tests demonstrate that in the unequal, additive surface tension case, the Voronoi implicit interface method does not converge to the desired limit. Then a variant that maintains the spirit of the original algorithm is presented. It appears to fix the non-convergence and as a bonus, the new variant extends the Voronoi implicit interface method to unequal mobilities.

CHAPTER I

Introduction

In this thesis, we develop higher order schemes for gradient flows with state of the art stability conditions. As a specific application, among others, we develop second order methods for two-phase threshold dynamics for simulating mean curvature motion. Both of these results are a first step in developing an efficient stable multi-phase mean curvature motion of first order and higher.

A gradient flow, or steepest descent, is an evolution equation of the form

$$u' = -\nabla_H E(u). \tag{1.1}$$

for $E : H \to \mathbb{R}$ and H is a Hilbert space with inner product $\langle \cdot, \cdot \rangle$. A fundamental property of equation (1.1) is that it dissipates the energy over time:

$$\frac{d}{dt}E(u) = \langle \nabla_H E(u), u' \rangle = -||\nabla_H E(u)||^2 \leq 0. \tag{1.2}$$

Our focus is on discretizations which are both high order in time and energy stable. By energy stable, we mean numerical schemes that have the discrete version of (1.2):

$$E(U^{N+1}) \leq E(U^N). \tag{1.3}$$

Our schemes can be seen as diagonally implicit Runge-Kutta (DIRK) multi-stage methods for generating unconditionally energy stable, high order in time numerical schemes. The advantage of the aforementioned method is accomplished by a black-box implementation of a standard backward Euler step. This allows one to increase the order of accuracy while preserving stability without developing new techniques. We briefly cover the vast literature of DIRK methods. For an extensive review of DIRK methods, see [27]. Diagonally implicit Runge-Kutta methods are linear M-stage schemes of the following form:

1. Set $U_0 = u_n$.
2. For $m = 1, \ldots, M$:

$$U_m = U_0 - k \sum_{i=1}^{m} \alpha_{m,i} \nabla_H E(U_i). \tag{1.4}$$

3. Set $u_{n+1} = U_M$.

The constants $\alpha_{m,i}$ are chosen to satisfy desired consistency and stability requirements.

When the energy E is convex, there are many DIRK methods that have some type of non-linear stability. For example, algebraic stability [4] and energy stability [45]. To our knowledge, there has been little, if any, work of non-linear stability of DIRK methods for general energies. We provide energy stable, high order DIRK methods for general energies in chapter II. We also exhibit an example, namely threshold dynamics, in which E is concave, yet the scheme is unconditionally stable with very low time step cost.

One downside to the DIRK method for non-linear $\nabla_H E$ is (1.4) involves solving an often costly system of non-linear equations. The implicit-explicit additive Runge-Kutta (ARK IMEX) [1] seeks to remedy this drawback by splitting the energy E into implicit and explicit parts. Specifically, let $E(u) = E_1(u) + E_2(u)$ where E_1 is treated implicitly and E_2 is treated explicitly. The ARK IMEX method is a linear M-stage scheme of the following form:

1. Set $U_0 = u_n$.
2. For $m = 1, \ldots, M$:

$$U_m = U_0 - k \sum_{i=1}^{m} \alpha_{m,i} \nabla_H E_1(U_i) - k \sum_{i=1}^{m-1} \tilde{\alpha}_{m,i} \nabla_H E_2(U_i). \tag{1.5}$$

3. Set $u_{n+1} = U_M$.

Where once again the constants $\alpha_{m,i}$ and $\tilde{\alpha}_{m,i}$ are chosen to satisfy desired consistency and stability requirements. There are second and third order unconditionally energy stable ARK IMEX methods [44]. However, the methods require that the part of the energy that is treated implicitly, E_1, is convex. In this thesis, we give conditionally energy stable higher order methods with no conditions on E_1 and E_2.

We extend both types of Runge-Kutta schemes to gradient flows with solution dependent inner product,

$$u' = -\mathcal{L}(u) \nabla_H E(u). \tag{1.6}$$

where $\mathcal{L}(u)$ is a positive definite operator. To our knowledge, this is the first time high order energy stable schemes for (1.6) have been proposed in a general setting.

We adapt our high order, energy stable schemes to threshold dynamics for simulating mean curvature motion. Unfortunately, threshold dynamics is not of form (1.1), so DIRK methods do not immediately generate high order schemes. However, threshold dynamics can be formally recognized as a gradient flow and, as such, some of our results apply. We briefly review mean curvature motion and algorithms for simulating its flow and indicate why we choose to focus on threshold dynamics.

1.1 Overview of Multiphase Mean Curvature Motion

Multiphase mean curvature motion arises as the gradient descent dynamics for energies of the form

$$E(\Sigma_1,\dots,\Sigma_n) = \sum_{i\neq j} \sigma_{ij}\text{Area}(\Gamma_{ij}). \tag{1.7}$$

Where $\Gamma_{ij} = (\partial\Sigma_i) \cap (\partial\Sigma_j)$ are the interfaces between the *phases* $\Sigma_1,\dots,\Sigma_n$ that partition a domain $D \subset \mathbb{R}^d$, $d \geq 2$:

$$\Sigma_i \cap \Sigma_j = (\partial\Sigma_i) \cap (\partial\Sigma_j) \text{ for any } i \neq j, \text{ and } \bigcup_{i=1}^{N} \Sigma_i = D.$$

The positive constants $\sigma_{ij} = \sigma_{ji}$ are known as *surface tensions* (or surface energy density). They need to satisfy the triangle inequality

$$\sigma_{ij} + \sigma_{ik} \geq \sigma_{jk} \text{ for any distinct } i, j \text{ and } k$$

for well-posedness of the model (1.7) (this inequality prevents “wetting” - where an interface between two phases is replaced by a thin third phase). Let a *triple junction* be formed by the meeting of three phases Σ_1, Σ_2, and Σ_3. They are points in two dimensions and occur along curves in three dimensions. Let θ_i be the angle between Γ_{ij} and Γ_{ik} at the junction. Then:

$$\frac{\sin\theta_1}{\sigma_{23}} = \frac{\sin\theta_2}{\sigma_{13}} = \frac{\sin\theta_3}{\sigma_{12}} \tag{1.8}$$

has to hold. This is known as the Herring angle condition [23]. We will now review algorithms for simulating mean curvature motion in this challenging multiphase setting.

1.2 Front Tracking methods

Front Tracking methods represent each interface as a parameterized curve $\gamma(s,t)$ for $s \in [0,1]$ [3]. Take the case of 3 parameterized curves γ^1,γ^2,γ^3 (representing, for example, Γ_{12}, Γ_{23}, and Γ_{13} respectively) that form a triple junction with angles θ_1, θ_2, and θ_3. Let $\gamma^1(1,t) = \gamma^2(1,t) = \gamma^3(1,t)$ and $\gamma^1(0,t), \gamma^2(0,t), \gamma^3(0,t) \in \delta D$. Each curve evolves according to $\gamma_t = \frac{\gamma_{ss}}{||\gamma_s||^2}$ constrained to be perpendicular at the boundary and satisfy $\frac{\gamma_s^1(1,t)}{||\gamma_s^1(1,t)||} \cdot \frac{\gamma_s^2(1,t)}{||\gamma_s^2(1,t)||} = \cos\theta_1$ and $\frac{\gamma_s^2(1,t)}{||\gamma_s^2(1,t)||} \cdot \frac{\gamma_s^3(1,t)}{||\gamma_s^3(1,t)||} = \cos\theta_2$ at the triple junction. The second condition ensures that (1.8) is satisfied during the evolution. The advantage of the method is that it is highly accurate. The major disadvantage is that it fails to model topological changes well. When a curve becomes smaller than a certain tolerance, it is deleted and the remaining curves are reorganized, termed as a 'surgery'. In order to perform a 'surgery' front tracking uses additional information about the physical system being modeled and each situation is implemented as a separate case. As a result, handling topological changes in front tracking is complicated and inelegant. Moreover, surgery in 3D is much harder and requires additional assumptions.

1.3 Level Set Methods

Introduced in [34], the level set formulation of only two phases is as follows: Denote one phase as Σ and the other as Σ^c and the interface as Γ. The interface is then embedded in a higher dimensional function $\phi(x)$ such that

$$\begin{cases} \phi(x) > 0 & x \in \Sigma \\ \phi(x) = 0 & x \in \Gamma \\ \phi(x) < 0 & x \in \Sigma^c. \end{cases}$$

Often the signed distance function, $d_\Gamma(x) = \min_{z\in\Gamma} ||x - z||_2$, is used for ϕ. For any level set function curvature at a point $x \in \Gamma$ is given by $\nabla \cdot (\frac{\nabla\phi(x)}{|\nabla\phi(x)|})$. To evolve Γ in time by curvature motion the function $\phi(x,t)$ is evolved by the initial value PDE

$$\phi_t - \nabla \cdot (\frac{\nabla \phi}{|\nabla \phi|})|\nabla \phi| = 0 \tag{1.9}$$

$$\phi(x, t = 0) = d_\Gamma(x), \tag{1.10}$$

The zero level set of $\phi(x,t)$ gives the interface after evolving by curvature motion for time t.

To extend this into the multiphase setting [30] suggested evolving each level set separately followed by a reconstruction of the phases. The steps are given in algorithm 1.

Algorithm 1 MBO Algorithm A

```
1: Let $nt = T/\Delta t$.
2: For $\Sigma_1, \dots, \Sigma_N$ let $\Gamma_i$ be the interface between $\Sigma_i$ and $\Sigma_i^c$.
3: Let $\phi_i^0(x,0) = d_{\Gamma_i}$
4: for $k \leftarrow 1$ to $nt$ do
5:     Evolve each $\phi_i^{k-1}$ using $\phi_t - \nabla \cdot (\frac{\nabla\phi}{|\nabla\phi|})|\nabla\phi| = 0$ for time $\Delta t$.
6:     Let $\phi_i^k(x,0) = \phi_i^{k-1}(x,\Delta t) - \max_{j\neq i} \phi_j^{k-1}(x,\Delta t)$
7: Then the new phases evolved under curvature motion for time $T$ are $\Sigma_i = \{x | \phi_i^{nt}(x) \geq 0\}$
```

The authors suggest replacing ϕ_i^k with the distance function of its zero level step at every step stating for numerical stability, but this turns out to be essential to the convergence of algorithm.

1.4 Variational Model

To extend this ad hoc method into the case of different surface tensions and give some theoretical backing, [48] proposed the following variation model based on the work of [36]:

$$\begin{gathered}\sum_{i=1}^{n} \sigma_i \int\int \delta(\phi_i(x,y,t))|\nabla \phi_i(x,y,t)| dxdy \\ \text{subject to} \sum_{i=1}^{n} H(\phi_i(x,y)) - 1 = 0\end{gathered} \tag{1.11}$$

where $\sigma_{ij} = \sigma_i + \sigma_j$. Curvature motion is given by gradient descent of (1.11). In order to apply the method they replace the above constraint with $\int_\Omega (\sum_i H(\phi_i(x,y)) - 1)^2 dxdy = \epsilon << 1$ and the $\delta(\phi)$'s with $|\nabla \phi|$. As pointed out in [17] the constraint may contribute to the stiffness of the problem and it is not clear that the dynamics of the system match that of mean curvature motion. Additionally, only a small subset of surface tensions that material scientists would like to model can be represented by this model. Additionally, the authors [48] give no numerical evidence of convergence in the unequal surface tension case. The authors of [17] give an unconstrained formulation but the method also suffers from stiffness and the convergence in the unequal surface tension case is unclear.

1.5 Voronoi Implicit Interface Method

Another way to extend level set methods into multiphase is the Voronoi Implicit Interface Method (VIIM) given by Saye and Sethian [40]. We will go into detail on this method in chapter V. The VIIM only works on a small subset of surface tensions and, as we show in chapter V, it does not converge in the unequal surface tension case.

1.6 Phase Field Methods

With two phases, Modica and Mortola [31] proposed the following energy:

$$E(u) = \int_\Omega \epsilon |\nabla u|^2 + \frac{1}{\epsilon} W(u) dx. \tag{1.12}$$

The variable u acts as a relaxation of a characteristic function. The function W, called a potential, is bistable with wells of equal depth (e.g. $W(u) = u^2(1-u)^2$). $W(u)$ penalizes u for violating the constraint $u(x) \in \{0,1\}$. Applying gradient descent to (1.12) results in the Allen-Cahn equation, $u_t = \epsilon \Delta u - \frac{1}{\epsilon} W(u)$, which has been shown to converge to mean curvature motion.

Building on this [19] looked at the following energy

$$E(u) = \int_\Omega \epsilon f(u, \nabla u) + \frac{1}{\epsilon} W(u) dx \tag{1.13}$$

where f is the generalized gradient energy. One part of the phase field is how to choose f and W in order to implement the curvature motion with the correct angle condition. One such example is $f(u, \nabla u) = \sum_{i<j} \frac{\tilde{\sigma}_{ij}}{\tilde{\mu}_{ij}} = |u_i \nabla u_j - u_j \nabla u_i|$ and $W(u) = \frac{16}{\pi^2} \sum_{i<j} \tilde{\mu}_{ij} \tilde{\sigma}_{ij} u_i u_j + \sum_{i<j<k} \sigma_{ijk} u_i u_j u_k$. For this they have $\tilde{\sigma}_{ij} = \sigma_{ij}$, $\tilde{\mu}_{ij} = \mu_{ij}$ and $\sigma_{ijk} = 5$. For these more general versions of phase field, the energy gradient terms in the energy are no longer quadratic resulting in gradient descent that is no longer a semilinear parabolic PDE. As a result the solution becomes more complicated. Additionally, the ϵ adds stiffness into the PDE.

1.7 Threshold Dynamics

The authors [30] looked at the splitting method applied to the Allen-Cahn equation which is

1. Evolve $\bar{u}_t = \epsilon \Delta \bar{u}$ with $\bar{u}(x, 0) = u^n$ for time T_d.
2. Evolve $u_t = -\frac{1}{\epsilon} W(u)$ with $u(x, 0) = \bar{u}(x, T_d)$ for some time T_r.
3. Set $u^{n+1} = u(x, T_r)$.

The question then becomes how to choose T_d and T_r to obtain the right solution. The correct solution can be found by taking $\lim T_r = \infty$. This results in a convolution step, where the $\bar{u}$ is evolved by the heat equation ($u_t = \Delta u$) to obtain $\bar{u}(x, T_d)$, then a thresholding step where $u^{n+1}(x) = 1$ where $\bar{u}(x, T_d) \geq \frac{1}{2}$ and zero elsewhere. The extension to multiple phases is given in algorithm 2.

To generalize the MBO method [15] considers the following approximation to

Algorithm 2 MBO algorithm

```
1: Let $G_{\delta t}(x) = \frac{1}{(4\pi(\delta t))^{d/2}} e^{-\frac{|x|^2}{4(\delta t)}}$
2: Given a initial partition $\Sigma_1^0, \dots, \Sigma_N^0$ and $nt = T/\delta t$
3: for $k \leftarrow 1$ to $nt$ do
4:     Let $\phi_i^k = G_{\delta t} * \mathbb{1}_{\Sigma_j^{k-1}}$.
5:     $\Sigma_i^k = \{x | \phi_i^k(x) \geq \max_{j \neq i} \phi_j^k(x)\}$
```

perimeter $P_{\delta t}(\Sigma) = \frac{1}{\sqrt{dt}} \int_{\Sigma^c} G_{\delta t} * \mathbb{1}_\Sigma dx$ where $G_{\delta t}(x) = \frac{1}{(4\pi(\delta t))^{d/2}} e^{-\frac{|x|^2}{4(\delta t)}}$. As a result

$$\text{Per}(\Gamma_{i,j}) \approx \frac{1}{\sqrt{\delta t}} \int \mathbb{1}_{\Sigma_i} G_{\delta t} * \mathbb{1}_{\Sigma_j} dx \tag{1.14}$$

and

$$E(\Sigma_1, \dots, \Sigma_N) \approx E_{\delta t}(\Sigma_1, \dots, \Sigma_N) = \frac{1}{\sqrt{\delta t}} \sum_{i,j=1}^{N} \sigma_{i,j} \int \mathbb{1}_{\Sigma_i} G_{\delta t} * \mathbb{1}_{\Sigma_j} dx. \tag{1.15}$$

Gradient descent on the energy approximation produces the algorithm 3. In future

Algorithm 3 EO algorithm

```
1: Given a initial partition $\Sigma_1^0, \dots, \Sigma_N^0$ and $nt = T/\delta t$
2: for $k \leftarrow 1$ to $nt$ do
3:     Let $\phi_i^k = G_{\delta t} * \sum_{j=1}^N \sigma_{i,j} \mathbb{1}_{\Sigma_j^{k-1}}$.
4:     $\Sigma_i^k = \{x | \phi_i^k(x) \leq \min_{j \neq i} \phi_j^k(x)\}$
```

work, the authors extend the method to anisotropic surface tensions [14]. One disadvantage is on a grid, taking too small a time step will cause the motion to get stuck, though there are various ways to avoid this. A variant of threshold dynamics that avoids this is distance function-based diffusion-generated motion [16]. Despite this drawback, threshold dynamics provably works for gradient flows of (1.7) for surface tension of interest to material scientists. Because of this, we focus on developing higher order methods for threshold dynamics.

This thesis is broken into four chapters.

- In chapter II, we develop new 2nd and 3rd order implicit schemes for general gradient flows that are unconditionally stable.

- In chapter III, we develop new 2nd and 3rd order semi-implicit schemes for general gradient flows that are conditionally stable and extend the results to gradient flows with solution dependent inner products.
- Chapter IV presents new 2nd order threshold dynamics algorithms, including an second order, energy stable algorithm in two dimensions.
- Chapter V presents highly accurate numerical convergence studies of the VIIM, showing that the VIIM does not converge in the unequal surface tension case. We also develop a new method in the spirit of the VIIM that fixes the non-convergence.

CHAPTER II

Variational Extrapolation of Implicit Schemes for General Gradient Flows

2.1 Introduction

We are concerned with numerical schemes for evolution equations that arise as gradient flow (steepest descent) for an energy $E : H \to \mathbb{R}$, where H is a Hilbert space with inner product $\langle \cdot, \cdot \rangle$:

$$u' = -\nabla_H E(u). \tag{2.1}$$

Equation (2.1) may represent a (scalar or vectorial) ordinary or partial differential equation. A fundamental property of equation (2.1) is that it dissipates the energy over time:

$$\frac{d}{dt}E(u) = \langle \nabla_H E(u), u' \rangle = -||\nabla_H E(u)||^2 \leq 0.$$

Our focus is on unconditionally energy stable, high order in time discretizations. To be precise, by energy stable we mean the following dissipative property:

$$E(u_{n+1}) \leq E(u_n) \tag{2.2}$$

where u_n denotes the approximation to the solution at the n-th time step. Thus, in the context of PDEs, where H is infinite dimensional, we are concerned with discrete in time, continuous in space schemes.

The backward Euler method for the abstract equation (2.1), with time step size $k > 0$, reads

$$\frac{u_{n+1} - u_n}{k} = -\nabla_H E(u_{n+1}). \tag{2.3}$$

As is well known and immediate to see, a solution u_{n+1} for the implicit scheme (2.3) can be found via the following optimization problem

$$u_{n+1} = \arg\min_u \left(E(u) + \frac{1}{2k}||u - u_n||^2 \right) \tag{2.4}$$

since (2.3) is the Euler-Lagrange equation for the optimization (2.4); here, $\|\cdot\|^2 = \langle \cdot, \cdot \rangle$. It follows that

$$E(u_{n+1}) \leq E(u_{n+1}) + \frac{1}{2k}||u_{n+1} - u_n||^2 \leq E(u_n) + \frac{1}{2k}||u_n - u_n||^2 = E(u_n) \tag{2.5}$$

so that scheme (2.3) is unconditionally stable, provided that optimization problem (2.4) can be solved.

Energetic formulation (2.4) of the backward Euler scheme (2.3) is often referred to as *minimizing movements*. It enables extending numerical schemes for the stationary optimization problem $\min_u E(u)$ to the dynamic, evolutionary problem (2.1) provided an additional, typically quadratic term in the cost function can be accommodated. The quadratic term $\frac{1}{2k}\|u - u_n\|^2$ in (2.4) is often referred to as the *movement limiter*, as it opposes deviation from the current configuration u_n. It encodes the inner product with respect to which the gradient flow is being generated. Beyond numerical analysis and computation, minimizing movements approximation of gradient flows have been instrumental in the analysis of evolution equations of the form (2.1), e.g. in defining and finding weak solutions beyond the formation of singularities when classical notions of solution cease to exist.

The following combination of desirable properties distinguish the new schemes introduced in this chapter:

1. Complete generality. There is no assumption (e.g. convexity) on the energy E in (2.1) beyond sufficient differentiability.
2. Unconditional energy stability.
3. High (at least up to third) order accuracy.
4. Each time step requires a few standard minimizing movements solves, equivalent to backward Euler substeps, or optimization of the associated energy plus a quadratic term.

Property 4 is perhaps the most unique and appealing aspect of the new framework: There are many existing schemes that can be recognized as some form of minimizing movements, sometimes relying on efficient optimization algorithms to solve (2.3) via (2.4). Our contribution shows how to painlessly jack up the order of accuracy of these schemes while preserving unconditional stability, relying only on a black-box implementation of the standard backward Euler scheme. In that sense, our new schemes can be understood as a variational analogue of Richardson extrapolation on (2.3), which in its standard form lacks the stability guarantees of our new schemes.

Many general purpose numerical schemes can certainly be used for solving (2.1), such as multistep or Runge-Kutta methods [5]. However, the energy stability of the standard examples of such schemes is either not immediate, or not true at all, at the level of generality we seek here, when applied to an equation of the form (2.1). Our focus is on high order schemes whose stability can be guaranteed over an entire class of evolution laws, namely gradient flows (2.1). Nevertheless, after some appropriate transformations, the new schemes we propose can be seen as a new, special class of diagonally implicit Runge-Kutta (DIRK) schemes tailored to these important dynamics. In the extensive literature on Runge-Kutta methods, one of the related contributions to the nonlinear notion of stability (2.2) we seek is B-stability for evolutions that satisfy a monotonicity (contractivity) condition [6]. In the context of gradient flows, this requires convexity of the energy E in (2.1), which is too restrictive for the applications we have in mind (see e.g. Examples (2.31) and (2.33) in section 2.5.2). Very recently, [44] & [45] propose high order Runge-Kutta schemes for gradient flows with stability guarantees. Among these, [45] concerns fully implicit schemes, as in the present work, but is again restricted to convex energies as in earlier works on B-convexity. The paper [44] studies implicit-explicit schemes that, in the spirit of convexity splitting [18], break up the energy into convex and concave parts, and treat the convex part implicitly and the concave part explicitly. The present work differs in placing no convexity assumptions on E, which is treated fully implicitly. An example where the energy is in fact *concave*, yet the optimization (2.4) is solvable at very low cost, is the threshold dynamics algorithm for motion by mean curvature [29, 30] that is known to be unconditionally energy stable [15]. We show in chapter IV how ideas developed in the present chapter can be used to jack up the order of accuracy of this intriguing algorithm while preserving its desirable stability properties, which appears to be beyond the scope of previous contributions. See also remark II.6 of section 2.5.2 in this context. Finally, we also mention recent work on the scalar auxiliary variable method [42] as another approach focusing on unconditional energy stability for gradient flows.

The rest of the chapter is organized as follows:

- section 2.2 presents the general framework for the new scheme, focusing on unconditional energy stability.
- section 2.3 focuses on consistency, showing how to attain 2nd and 3rd order accuracy.
- section 2.4 gives 2nd and 3rd order examples of the new schemes.

- section 2.5 presents numerical convergence studies on a number of well-known ordinary and partial differential equations that are gradient flows.

The code for section 2.5 is publicly available, and can be found at `https://github.com/AZaitzeff/gradientflow`.

2.2 The New Schemes: Stability

In this section, we formulate a wide class of numerical schemes that are energy stable by construction. We thus place stability front and center, leaving consistency to be dealt with subsequently. It is therefore important to allow many degrees of freedom in the scheme at this stage, in the form of a large number of coefficients, that will eventually be chosen, in the next section, to attain consistency at a high order of accuracy.

Our method is a linear M-stage scheme of the following form:

1. Set $U_0 = u_n$
2. For $m = 1, \ldots, M$:

$$U_m = \arg\min_u \left(E(u) + \sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} ||u - U_i||^2 \right). \tag{2.6}$$

3. Set $u_{n+1} = U_M$

Notice that the proposed scheme (2.6), as promised, merely requires the solution of exactly the same type of problem at every time step as the standard backward Euler scheme: minimization of the associated energy plus a quadratic term.

At this point, it is not clear why a scheme such as (2.6) should dissipate energy E at every iteration as in (2.2). However, in this section we establish quite broad conditions on the coefficients $\gamma_{m,i}$ that ensure energy dissipation (2.2); this is the essential observation at the heart of the present chapter. To demonstrate the idea, consider the following two-stage special case of scheme (2.6):

$$U_1 = \arg\min_u \left(E(u) + \frac{\gamma_{1,0}}{2k}||u - u_n||^2 \right) \tag{2.7}$$

$$u_{n+1} = \arg\min_u \left(E(u) + \frac{\gamma_{2,0}}{2k}||u - u_n||^2 + \frac{\gamma_{2,1}}{2k}||u - U_1||^2 \right) \tag{2.8}$$

and impose the conditions

$$\gamma_{1,0} - \frac{\gamma_{2,0}^2}{\gamma_{2,0} + \gamma_{2,1}} \geq 0 \text{ and } \gamma_{2,0} + \gamma_{2,1} > 0 \tag{2.9}$$

on the parameters. Set $\theta = \frac{\gamma_{2,0}}{\gamma_{2,0}+\gamma_{2,1}}$. Note that (2.8) is equivalent to

$$u_{n+1} = \arg\min_u \left(E(u) + \frac{\gamma_{2,0} + \gamma_{2,1}}{2k} \left\|u - \big(\theta u_n + (1-\theta)U_1\big)\right\|^2 \right). \tag{2.10}$$

This can be seen by expanding the norm squared and comparing the quadratic and linear terms in u. The constant terms are not equal but that does not matter for the minimization.

We have

$$\begin{aligned}
E(u_{n+1}) &\leq E(u_{n+1}) + \frac{\gamma_{2,0} + \gamma_{2,1}}{2k} \left\|u_{n+1} - \big(\theta u_n + (1-\theta)U_1\big)\right\|^2 && \text{(by (2.9))} \\
&\leq E(U_1) + \frac{\gamma_{2,0} + \gamma_{2,1}}{2k} \left\|U_1 - \big(\theta u_n + (1-\theta)U_1\big)\right\|^2 && \text{(by (2.10))} \\
&= E(U_1) + \frac{\gamma_{2,0}^2}{(\gamma_{2,1} + \gamma_{2,0})2k} \left\|U_1 - u_n\right\|^2 \\
&\leq E(U_1) + \frac{\gamma_{1,0}}{2k} \left\|U_1 - u_n\right\|^2 && \text{(by (2.9))} \\
&\leq E(u_n) && \text{(by (2.7))}
\end{aligned}$$

establishing unconditional energy stability of scheme (2.7) and (2.8) under the condition (2.9) on its parameters. We offer some insight to the conditions in (2.9). First, the condition $\gamma_{2,0} + \gamma_{2,1} > 0$ is reasonable as it requires that the function being minimized in (2.8) goes to $+\infty$ as $||u|| \to \infty$. What is more surprising is that one of the second stage coefficients can be negative while maintaining unconditionally stability. We can 'reward' the distance to one of the previous stages as long as the distance to the other stage is penalized sufficiently strongly. The condition $\gamma_{1,0} \geq \frac{\gamma_{2,0}^2}{\gamma_{2,0}+\gamma_{2,1}}$ requires that the penalization in the first stage has to be strong relative to the penalization in the second stage.

We will now extend this discussion to the general, M-stage case of scheme (2.6):

Theorem II.1. *Define the following auxiliary quantities in terms of the coefficients $\gamma_{m,i}$ of scheme* (2.6)*:*

$$\tilde{\gamma}_{m,i} = \gamma_{m,i} - \sum_{j=m+1}^{M} \tilde{\gamma}_{j,i} \frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}} \tag{2.11}$$

$$\tilde{S}_{j,m} = \sum_{i=0}^{m-1} \tilde{\gamma}_{j,i} \tag{2.12}$$

Where if $m = M$, the sum in (2.11) *is considered to be zero. If $\tilde{S}_{m,m} > 0$ for $m = 1, \ldots, M$, then scheme* (2.6) *satisfies the energy stability condition* (2.2)*: For every $n = 0, 1, 2, \ldots$ we have $E(u_{n+1}) \leq E(u_n)$.*

As we will see in section 2.3, the conditions on the parameters $\gamma_{i,j}$ of scheme (2.6) imposed in theorem II.1 are loose enough to enable meeting consistency conditions to high order. We will establish theorem II.1 with the help of the following two lemmas. The first lemma is the multi-step version of the equivalence of (2.8) and (2.10) in our two step example:

Lemma II.2. *Let the auxiliary quantities $\tilde{S}_{j,m}$, and $\tilde{\gamma}_{m,i}$ be defined as in theorem II.1.*

We have

$$\begin{aligned}&\arg\min_u \left(E(u) + \sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} ||u - U_i||^2 \right)\\ = &\arg\min_u \left(E(u) + \frac{1}{2k} \sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} ||u - \sum_{i=0}^{m-1} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i||^2 \right)\end{aligned}$$

Proof. As in the two step case the proof consists of expanding the norm squared terms and showing that all the quadratic and linear terms of u are equal. First the expansion of $\sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} ||u - U_i||^2$ is

$$\frac{||u||^2}{2k} \sum_{i=0}^{m-1} \gamma_{m,i} - \frac{1}{k} \langle u, \sum_{i=0}^{m-1} \gamma_{m,i} U_i \rangle + \text{terms that do not depend on } u. \tag{2.13}$$

Next, we will establish two identities to help us expand

$$\frac{1}{2k} \sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} ||u - \sum_{i=0}^{m-1} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i||^2.$$

First by rearranging (2.11),

$$\gamma_{m,i} = \sum_{j=m}^{M} \tilde{\gamma}_{j,i} \frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}. \tag{2.14}$$

Next, an identity of $\tilde{S}_{m,m}$:

$$\begin{aligned}\tilde{S}_{m,m} &= \sum_{i=0}^{m-1} \tilde{\gamma}_{m,i} = \sum_{i=0}^{m-1} \left[\gamma_{m,i} - \sum_{j=m+1}^{M} \tilde{\gamma}_{j,i} \frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}} \right]\\ &= \sum_{i=0}^{m-1} \gamma_{m,i} - \sum_{j=m+1}^{M} \left[\sum_{i=0}^{m-1} \tilde{\gamma}_{j,i} \right] \frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}} = \sum_{i=0}^{m-1} \gamma_{m,i} - \sum_{j=m+1}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}.\end{aligned}$$

We use this identity to establish the following:

(2.15)
$$\sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} = \tilde{S}_{m,m} + \sum_{j=m+1}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} = \sum_{i=0}^{m-1} \gamma_{m,i} - \sum_{j=m+1}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} + \sum_{j=m+1}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} = \sum_{i=0}^{m-1} \gamma_{m,i}.$$

Now we can calculate the expansion:

$$\begin{aligned}
&\frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||u-\sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2\\
=&\frac{||u||^2}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}-\frac{1}{k}\langle u,\sum_{i=0}^{m-1}\sum_{j=m}^{M}\tilde{\gamma}_{j,i}\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}U_i\rangle+\text{terms that do not depend on } u\\
=&\frac{||u||^2}{2k}\sum_{i=0}^{m-1}\gamma_{m,i}-\frac{1}{k}\langle u,\sum_{i=0}^{m-1}\gamma_{m,i}U_i\rangle+\text{terms that do not depend on } u.
\end{aligned}$$

Where the last equality follows from (2.14) and (2.15). Since this expansion matches (2.13) up to a constant in u the proof is complete. □

Now we will use lemma II.2 to relate the energy of sub-step m to sub-step $m-1$. This lemma is the crux of the proof of the theorem and where we use the condition that $\tilde{S}_{m,m} > 0$ for all m.

Lemma II.3. *Let the auxiliary quantities $\tilde{S}_{j,m}$, $\tilde{\gamma}_{m,i}$ be given in theorem II.1 and let $\tilde{S}_{m,m} > 0$ for $m = 1, \dots, M$. Then*

$$\begin{aligned}
&E(U_m)+\frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_m-\sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2\\
\leq&E(U_{m-1})+\frac{1}{2k}\sum_{j=m-1}^{M}\frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}||U_{m-1}-\sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i||^2
\end{aligned}$$

Proof. By (2.6) and lemma II.2,

$$U_m = \arg\min_u E(u)+\frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||u-\sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2.$$

Since U_m is the minimizer of the above optimization problem

$$E(U_m) + \frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_m - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2 \tag{2.16}$$

$$\leq E(U_{m-1}) + \frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_{m-1} - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2 \tag{2.17}$$

Next using the definition of auxiliary variables we can state an identity that will simplify (2.17). For $m > 1$ and $j \geq m$

$$\begin{aligned}
&\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_{m-1} - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2 = \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_{m-1}\left(1 - \frac{\tilde{\gamma}_{j,m-1}}{\tilde{S}_{j,m}}\right) - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2 \\
=&\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}||U_{m-1}\left(\frac{\tilde{S}_{j,m-1}}{\tilde{S}_{j,m}}\right) - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i||^2 = \frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}||U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i||^2.
\end{aligned}$$

Using this identity (2.17) is equal to

$$E(U_{m-1}) + \frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}||U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i||^2 \tag{2.18}$$

Now since $\tilde{S}_{m-1,m-1} > 0$,

$$\frac{\tilde{S}_{m-1,m-1}^2}{\tilde{S}_{m-1,m-1}}||U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{m-1,i}}{\tilde{S}_{m-1,m-1}}U_i||^2 > 0. \tag{2.19}$$

By adding (2.19) to (2.18), we have that (2.18) is less than or equal to

$$E(U_{m-1}) + \frac{1}{2k}\sum_{j=m-1}^{M}\frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}||U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i||^2$$

concluding the proof. □

Proof. (of theorem) The main idea of the proof is to use lemma II.3 repeatedly to relate the energy of $E(u_{n+1})$ to $E(u_n)$. First, since $\tilde{S}_{M,M} > 0$

$$E(u_{n+1}) = E(U_M) \leq E(U_M) + \frac{1}{2k}\frac{\tilde{S}_{M,M}^2}{\tilde{S}_{M,M}}||U_M - \sum_{i=0}^{M-1}\frac{\tilde{\gamma}_{M,i}}{\tilde{S}_{M,M}}U_i||^2$$

The right hand side of the equation is of the form required by lemma II.3. By using the lemma II.3 repeatedly we have

$$\begin{aligned}
&E(U_M) + \frac{1}{2k}\frac{\tilde{S}_{M,M}^2}{\tilde{S}_{M,M}}||U_M - \sum_{i=0}^{M-1}\frac{\tilde{\gamma}_{M,i}}{\tilde{S}_{M,M}}U_i||^2 \\
\leq &E(U_{M-1}) + \frac{1}{2k}\sum_{j=M-1}^{M}\frac{\tilde{S}_{j,M-1}^2}{\tilde{S}_{j,j}}||U_{M-1} - \sum_{i=0}^{M-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,M-1}}U_i||^2 \\
&\vdots \\
\leq &E(U_1) + \frac{1}{2k}\sum_{j=1}^{M}\frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}}||U_1 - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}}U_0||^2.
\end{aligned}$$

By (2.6) and lemma II.2

$$U_1 = \arg\min_u E(u) + \frac{1}{2k}\sum_{j=1}^{M}\frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}}||u - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}}U_0||^2$$

so

$$E(U_1) + \frac{1}{2k}\sum_{j=1}^{M}\frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}}||U_1 - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}}U_0||^2 \leq E(U_0) + \frac{1}{2k}\sum_{j=1}^{M}\frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}}||U_0 - U_0||^2 = E(u_n)$$

completing the proof of the theorem. □

Now the condition that $\tilde{S}_{m,m} > 0$ for $m = 1, \ldots, M$ is the multi-step equivalent of (2.9) in the two step case. Given the γ's you can calculate the auxiliary quantities (2.11) and (2.12) explicitly as follows:

for $m = M, M-1, \ldots, 1$:

1. Calculate $\tilde{\gamma}_{m,i}$ for $i = 1, 2, \ldots, m$
2. Calculate $\tilde{S}_{j,m}$ for $j = m, m+1, \ldots, M$.

Thus given γ's we can easily check if they satisfy the hypothesis of theorem II.1.

2.3 The New Schemes: Consistency

We now turn to the question of whether the coefficients $\gamma_{m,i}$ in scheme (2.6) can be chosen to ensure its high order consistency with the abstract evolution law (2.1). As mentioned before, the schemes are diagonal implicit Runge-Kutta, whose order conditions are well established (for example in [5]). For completeness, we derive the conditions here. From (2.6), each stage U_m satisfies the Euler-Lagrange equation:

$$\left[\sum_{i=0}^{m-1}\gamma_{m,i}\right]U_m + k\nabla_H E(U_m) = \sum_{i=0}^{m-1}\gamma_{m,i}U_i. \tag{2.20}$$

The consistency equations for the γs are found by carrying out a Taylor series expansion of U_m around $U_0 = u(t_0)$. We will calculate the one-step error. For $n \in \{1,2,3,\ldots\}$, let $D^nE(u) : H^n \to \mathbb{R}$ denote the multilinear form given by

$$D^nE(u)(v_1,\ldots,v_n) = \frac{\partial^n}{\partial s_1\cdots\partial s_n}E(u + s_1v_1 + s_2v_2 + \cdots + s_nv_n)\bigg|_{s_1=s_2=\cdots=s_n=0}$$

so that the linear functional $D^nE(u)(v_1,v_2,\ldots,v_{n-1},\cdot) : H \to \mathbb{R}$ may be identified with an element of H, which will be denoted simply as $D^nE(v_1,v_2,\ldots,v_{n-1})$ in what follows. We begin with the exact solution starting from $u(t_0)$:

$$\begin{cases} u_t = -\nabla E(u) & t > t_0 \\ u(t_0) = U_0 \end{cases}$$

The Taylor expansion of $u(k+t_0)$ around t_0 is

(2.21)

$$\begin{aligned} u(k+t_0) =& u(t_0) + ku_t(t_0) + \frac{1}{2}k^2u_{tt}(t_0) + \frac{1}{6}k^3u_{ttt}(t_0) + \text{h.o.t.} \\ =& U_0 - kDE(U_0) + \frac{1}{2}k^2D^2E(U_0)DE(U_0) \\ &- \frac{1}{6}k^3\big[D^2E(U_0)\left(D^2E(U_0)\left(DE(U_0)\right)\right) + D^3E(U_0)\big(DE(U_0), DE(U_0)\big)\big] + \text{h.o.t.} \end{aligned}$$

We now present the error at each stage of the multi-stage algorithm, (2.6), and the conditions required to achieve various orders of accuracy:

Claim II.4. *Let U_m be given in* (2.6) *for $m = 0, 1, \dots, M$. The Taylor expansion of U_m at each stage has the same form as* (2.21), *namely:*

$$U_m = U_0 - \beta_{1,m} k DE(U_0) + \beta_{2,m} k^2 D^2E(U_0)DE(U_0) \tag{2.22}$$

$$-k^3 \left[\beta_{3,m} D^2E(U_0)\left(D^2E(U_0)\left(DE(U_0)\right)\right) + \beta_{4,m} D^3E(U_0)\big(DE(U_0), DE(U_0)\big)\right] + \mathcal{O}(k^4)$$

where the coefficients obey the following recursive relation

$$\begin{aligned}
&\beta_{1,0} = \beta_{2,0} = \beta_{3,0} = \beta_{4,0} = 0 \\
&\beta_{1,m} = \frac{1}{S_m}\left[1 + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{1,i}\right] \\
&\beta_{2,m} = \frac{1}{S_m}\left[\beta_{1,m} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{2,i}\right] \\
&\beta_{3,m} = \frac{1}{S_m}\left[\beta_{2,m} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{3,i}\right] \\
&\beta_{4,m} = \frac{1}{S_m}\left[\frac{\beta_{1,m}^2}{2} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{4,i}\right]
\end{aligned} \tag{2.23}$$

with $S_m = \sum_{i=0}^{m-1} \gamma_{m,i}$. Furthermore, the following conditions for U_M in scheme (2.6) *are necessary and sufficient for various orders of accuracy:*

(2.24)

First Order:	*Second Order:*	*Third Order:*
$\beta_{1,M} = 1$	$\beta_{1,M} = 1$	$\beta_{1,M} = 1$
	$\beta_{2,M} = 1/2$	$\beta_{2,M} = 1/2$
		$\beta_{3,M} = 1/6$
		$\beta_{4,M} = 1/6$

Proof. We will now show by induction that the aforementioned consistency formulas, (2.22) and (2.23), hold.

Stage zero: U_0 trivially satisfies (2.22) and (2.23).

Stage m: Assume (2.22) and (2.23) up to stage $m-1$. First we are going to solve for $U_m - U_0$ in (2.20):

$$U_m - U_0 = -\frac{k}{S_m} DE(U_m) + \frac{1}{S_m}\sum_{i=0}^{m-1}\gamma_{m,i}U_i - U_0. \tag{2.25}$$

Now Taylor expand $DE(U_m)$ around U_0 in (2.25):

$$\begin{aligned} U_m - U_0 = &-\frac{k}{S_m}\Big[DE(U_0) + D^2E(U_0)(U_m - U_0) + \frac{1}{2}D^3E(U_0)\big(U_m - U_0, U_m - U_0\big)\Big] \\ &+ \frac{1}{S_m}\sum_{i=0}^{m-1}\gamma_{m,i}U_i - U_0 + \text{h.o.t.} \end{aligned}$$

Substituting the ansatz $U_0 + kA_1 + k^2A_2 + k^3A_3 + \mathcal{O}(k^4)$ for U_m and equation (2.22) for U_i, and retaining up to terms of third order, we have that

$$\begin{aligned} &kA_1 + k^2A_2 + k^3A_3 = \\ &\quad -\frac{k}{S_m}\Big[1 + \sum_{i=1}^{m-1}\gamma_{m,i}\beta_{1,i}\Big]DE(U_0) + k^2\Big[\frac{1}{S_m}D^2E(U_0)\big(-A_1 + \sum_{i=1}^{m-1}\gamma_{m,i}\beta_{2,i}DE(U_0)\big)\Big] \\ &\quad - k^3\Big[\frac{1}{S_m}D^2E(U_0)\big(A_2 + \sum_{i=1}^{m-1}\gamma_{m,i}\beta_{3,i}D^2E(U_0)DE(U_0)\big) \\ &\quad + \frac{1}{2}\frac{1}{S_m}D^3E(U_0)\big(A_1, A_1\big) + \frac{1}{S_m}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{4,i}D^3E(U_0)\big(DE(U_0), DE(U_0)\big)\Big] + \mathcal{O}(k^4) \end{aligned} \tag{2.26}$$

Solving for A_1, A_2, A_3 by matching terms of the same order in (2.26), we arrive at:

$$
\begin{aligned}
A_1 =& -\frac{1}{S_m}\left[1+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{1,i}\right]DE(U_0)\\
A_2 =& \frac{1}{S_m}D^2E(U_0)\big(-A_1+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{2,i}DE(U_0)\big)\\
=& \left(\frac{1}{S_m^2}\left[1+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{1,i}\right]+\frac{1}{S_m}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{2,i}\right)D^2E(U_0)\big(DE(U_0)\big)\\
A_3 =& -\frac{1}{S_m}D^2E(U_0)\big(A_2+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{3,i}D^2E(U_0)DE(U_0)\big)\\
& -\frac{1}{2}\frac{1}{S_m}D^3E(U_0)\big(A_1,A_1\big)-\frac{1}{S_m}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{4,i}D^3E(U_0)\big(DE(U_0),DE(U_0)\big)\\
=& -\left(\frac{1}{S_m^3}\left[1+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{1,i}\right]+\frac{1}{S_m^2}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{2,i}\right.\\
& \left.+\frac{1}{S_m}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{3,i}\right)D^2E(U_0)\left(D^2E(U_0)\left(DE(U_0)\right)\right)\\
& -\left(\frac{1}{2}\frac{1}{S_m^3}\left[1+\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{1,i}\right]^2+\frac{1}{S_m}\sum_{i=1}^{m-1}\gamma_{m,i}\beta_{4,i}\right)D^3E(U_0)\big(DE(U_0),DE(U_0)\big)
\end{aligned}
$$

completing the induction step.

Matching the consistency equations, (2.22) and (2.23), at U_M with the one step error (2.21) gives the conditions on U_M for various orders of accuracy (2.24), completing the proof. $\square$

In the next section, we give examples of γ's that satisfy the consistency equations (claim II.4) as well as the hypothesis of theorem II.1 concurrently.

2.4 The New Schemes: Examples

In this section, we exhibit second and third order examples of scheme (2.6) that satisfy concurrently the hypothesis guaranteeing unconditional energy stability (theorem II.1) and the consistency equations (claim II.4) up to second and third order.

We found the γ's by the following numerical procedure: we first found a set of γ's that satisfied the conditions of theorem II.1. Then we used the interior point method with the conditions of theorem II.1 as our constraint and an objective function that penalized the mismatch between the current $\beta_{1,M}$ and $\beta_{2,M}$ (and $\beta_{3,M}$ and $\beta_{4,M}$ for third order) and (2.24). After obtaining γ's that satisfied the consistency equations up to some small tolerance as well as our constraint, we sought a nearby algebraic solution to the consistency equations that still satisfied the conditions in theorem II.1. For some number of stages M, it is impossible to satisfy the consistency equation for a given order and the stability conditions. Therefore, we searched for γ's that encoded stable algorithms of various orders with different total number of stages and report a set of γ's with the lowest number of stages for a given order here. Using this method we were able to find second and third order stable schemes. Whether even higher order accuracy (together with stability) can be obtained with this class of schemes will require a more systematic approach to the solvability of the conditions on γ, and will be the subject of future work.

2.4.1 Second Order Examples

It can be shown that there is no unconditionally energy stable second order two-stage method. However, it turns out that three stages are sufficient for unconditional stability:

$$\gamma = \begin{pmatrix} \gamma_{1,0} & 0 & 0 \\ \gamma_{2,0} & \gamma_{2,1} & 0 \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} \end{pmatrix} = \begin{pmatrix} 5 & 0 & 0 \\ -2 & 6 & 0 \\ -2 & \frac{3}{14} & \frac{44}{7} \end{pmatrix} \approx \begin{pmatrix} 5.0 & 0 & 0 \\ -2.0 & 6.0 & 0 \\ -2.0 & 0.22 & 6.29 \end{pmatrix}. \tag{2.27}$$

This choice of γ's that endows the three-stage method (2.6) with unconditional stability and second order accuracy is by no means unique; indeed, here is another that has the additional benefit of having each one of its stages depend only on the previous one and u_n:

$$\gamma = \begin{pmatrix} \frac{9}{2} & 0 & 0 \\ -\frac{11}{6} & \frac{44}{7} & 0 \\ -\frac{287591}{148306} & 0 & \frac{944163}{148306} \end{pmatrix} \approx \begin{pmatrix} 4.5 & 0 & 0 \\ -1.83 & 6.29 & 0 \\ -1.94 & 0 & 6.37 \end{pmatrix}. \tag{2.28}$$

2.4.2 Third Order Example

We now exhibit a six stage version of scheme (2.6) that concurrently satisfies the conditions for unconditional energy stability (theorem II.1) as well the consistency equations (claim II.4) up to third order:

$$\gamma \approx \begin{pmatrix} 11.17 & 0 & 0 & 0 & 0 & 0 \\ -7.5 & 19.43 & 0 & 0 & 0 & 0 \\ -1.05 & -4.75 & 13.98 & 0 & 0 & 0 \\ 1.8 & 0.05 & -7.83 & 13.8 & 0 & 0 \\ 6.2 & -7.17 & -1.33 & 1.63 & 11.52 & 0 \\ -2.83 & 4.69 & 2.46 & -11.55 & 6.68 & 11.95 \end{pmatrix} \tag{2.29}$$

The exact values of the γ's above are given in the appendix (chapter A); they are all rational numbers but with long fractional representations. Again, we cannot rule out other solutions for γ, possibly with fewer stages.

2.5 The New Schemes: Numerical Tests

In this section, we will apply the second order (2.27) and third (2.29) order accurate unconditionally stable schemes to a variety of gradient flows. We found (2.27) before (2.28) and therefore ran all our numerical tests with the former. The gradient flows considered span linear and non-linear ordinary and partial differential equations. The corresponding energies include convex and non-convex forms. Careful numerical convergence studies are presented in each case to verify the anticipated convergence rates of previous sections.

Remark II.5. Note that equation (2.6) can be rewritten using only one quadratic movement limiter term, so a black box implementation for backward Euler (2.3), or equivalently (2.4), is all that is needed for our method, and is called once per stage.

2.5.1 Ordinary Differential Equations

First, we turn to the ODE $u' = -\sinh(u)$ with the corresponding energy $E(u) = \cosh(u)$. With initial condition $u(0) = -2$, the exact solution is $u_*(t) = -2\coth^{-1}(\exp(t)\coth(1))$. Table 2.1 and table 2.2 show the error in the solution at time $t = 2$ computed by the second order scheme (2.6) & (2.27) and the third order scheme (2.6) & (2.29), respectively, at various choices of the time step size. The anticipated order of convergence is clearly observed for both schemes. Figure 2.1 shows the energy at every time step for the third order method with 16 time steps. As expected, the energy decreases at every time step. There is little visual difference between fig. 2.1, the plot of the second order method with 16 time steps and the plot of the exact energy.

Number of time steps	2^4	2^5	2^6	2^7	2^8
Error at $t = 2$	5.25e-04	1.31e-04	3.27e-05	8.18e-06	2.05e-06
Order	-	2.00	2.00	2.00	2.00

Table 2.1: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.27) on the ODE $u' = -\sinh(u)$ with energy $E(u) = \cosh(u)$.

Number of time steps	2^4	2^5	2^6	2^7	2^8
Error at $t = 2$	1.19e-05	1.48e-06	1.85e-07	2.30e-08	2.88e-09
Order	-	3.00	3.00	3.00	3.00

Table 2.2: The new third order accurate, unconditionally stable, six-stage scheme (2.6) & (2.29) on the ODE $u' = -\sinh(u)$ with energy $E(u) = \cosh(u)$.

We next turn to an ODE with the non-smooth energy

$$E(u) = \begin{cases} \frac{1}{2}|u| & \text{if } |u| < 1 \\ |u-1| + \frac{1}{2} & \text{if } |u| \geq 1. \end{cases} \tag{2.30}$$

Since the energy is non-smooth we do not expect higher order convergence. As shown in fig. 2.2, the second (2.6) & (2.27) and third order scheme (2.6) & (2.29) obtain

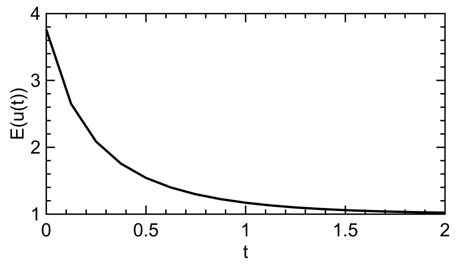

Figure 2.1: The new third six-stage scheme (2.6) & (2.29) with 16 time steps on the ODE on the ODE $u' = -\sinh(u)$ with energy $E(u) = \cosh(u)$

first order convergence on average. Notwithstanding, the energy decreases at every time step as shown in fig. 2.3 for the third order method with 16 time steps.

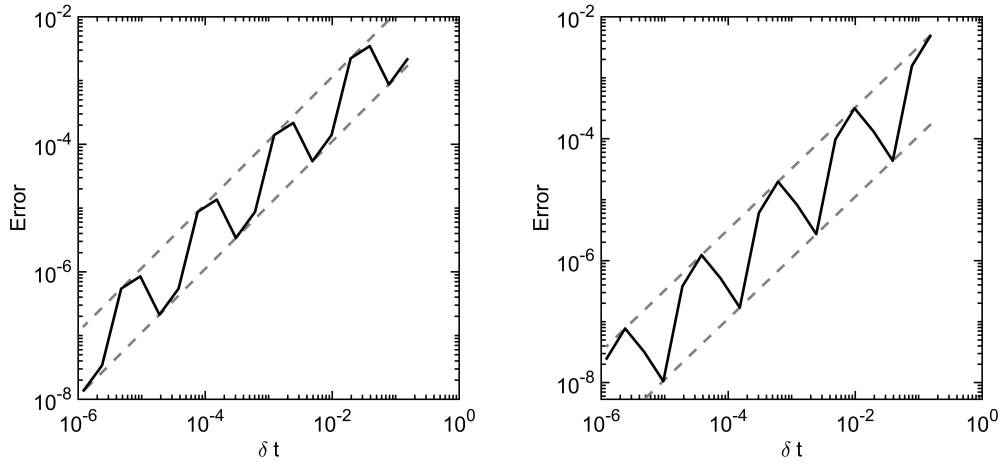

Figure 2.2: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.29) (right) and the new third six-stage scheme (2.6) & (2.29) (left) on the ODE induced by gradient flow on non smooth energy (2.30)

2.5.2 Partial Differential Equations

For PDEs, we start with a preliminary test on the one dimensional heat equation $u_t = u_{xx}$ on $x \in [-1, 1]$ subject to periodic boundary conditions with initial data $u(x, 0) = \sin(\pi x)$. This is gradient flow with respect to the L^2 inner product for the energy $E(u) = \frac{1}{2}\int u_x^2\, dx$. The exact solution is $u_*(x, t) = \sin(\pi x)\exp(-\pi^2 t)$. The spatial domain $[-1, 1]$. For this example as well as the other PDEs in this section, we

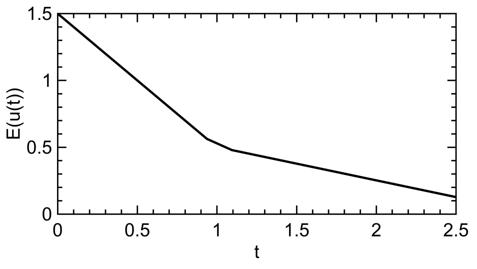

Figure 2.3: The new third order six-stage scheme (2.6) & (2.29) with 16 time steps on the ODE induced by gradient flow on non smooth energy (2.30)

choose the discretization of the Laplacian and number of spatial points so that the contribution to the error from the spatial discretization is negligible. Table 2.3 and table 2.4 show the L^2 error in the approximate solution at $t = \frac{1}{8}$, computed via the second order accurate scheme (2.6) & (2.27), and the third order accurate scheme (2.6) & (2.29), respectively. Figure 2.4 shows the energy at every time step for the third order method with 16 time steps. We see that the energy decreases at every time step.

Number of time steps	2^2	2^3	2^4	2^5	2^6	2^7
L2	1.09e-03	2.66e-04	6.59e-05	1.64e-05	4.09e-06	1.02e-06
Order	-	2.03	2.01	2.01	2.00	2.00

Table 2.3: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.27) on the one-dimensional heat equation $u_t = u_{xx}$.

Number of time steps	2^2	2^3	2^4	2^5	2^6	2^7
L2	2.30e-05	2.75e-06	3.36e-07	4.16e-06	5.17e-09	6.37e-10
Order	-	3.06	3.03	3.02	3.01	3.02

Table 2.4: The new third order accurate, unconditionally stable, six-stage scheme (2.6) & (2.29) on the one-dimensional heat equation $u_t = u_{xx}$.

We now turn to less trivial examples, starting with the Allen-Cahn equation

$$u_t = \Delta u - W'(u) \tag{2.31}$$

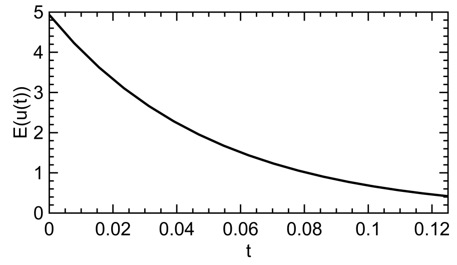

Figure 2.4: The new third six-stage scheme (2.6) & (2.29) with 16 time steps on the one-dimensional heat equation $u_t = u_{xx}$

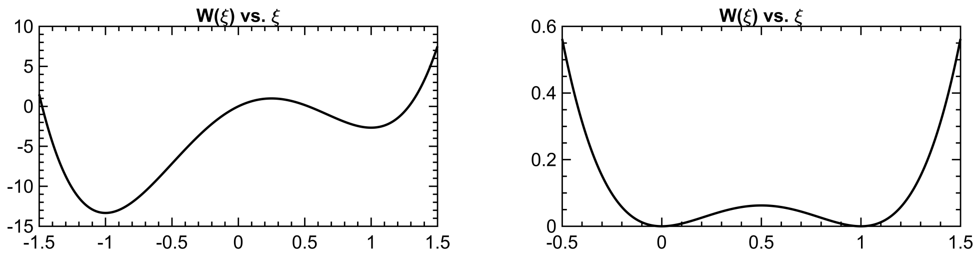

Figure 2.5: The double well potentials used in the Allen-Cahn (2.31) and Cahn-Hilliard (2.33) equations: One with unequal and the other with equal depth wells.

where $W : \mathbb{R} \to \mathbb{R}$ is a double-well potential. This is gradient flow for the energy

$$E(u) = \int \frac{1}{2}|\nabla u|^2 + W(u)\,dx \tag{2.32}$$

with respect to the L^2 inner product.

First, we consider equation (2.31) in one space dimension, with the potential $W(u) = 8u - 16u^2 - \frac{8}{3}u^3 + 8u^4$. This is a double well potential with unequal depth wells; see fig. 2.5. In this case, equation (2.31) is well-known to possess traveling wave solutions on $x \in \mathbb{R}$, see fig. 2.6. We choose the initial condition $u(x,0) = \tanh(4x + 20)$; the exact solution is then $u_*(x,t) = \tanh(4x + 20 - 8t)$. The computational domain is $x \in [-10, 10]$. We approximate the solution on $\mathbb{R}$ by using the Dirichlet boundary conditions $u(\pm 10, t) = \pm 1$: The domain size is large enough that the mismatch in boundary conditions do not substantially contribute to the error in the approximate solution over the time interval $t \in [0, 5]$. Table 2.5 and table 2.6

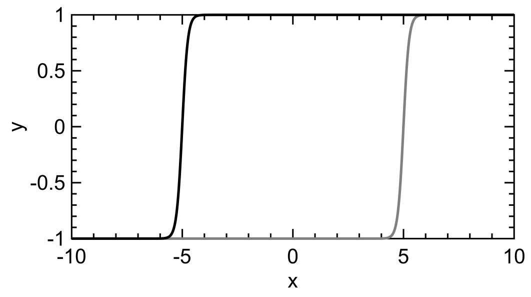

Figure 2.6: The initial condition (black) and the solution at final time (gray) in the numerical convergence study on the 1D Allen-Cahn equation (2.31) with a potential that has unequal depth wells.

Number of time steps	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
L2	5.14e-02	1.26e-02	3.13e-03	7.79e-04	1.94e-04	4.86e-05
Order	-	2.02	2.01	2.01	2.00	2.00

Table 2.5: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.27) on the one-dimensional Allen-Cahn equation (2.31) with a traveling wave solution.

tabulate the error in the computed solution at time $t = 5$ for our two new schemes.

Number of time steps	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
L2	9.06e-04	9.97e-05	1.20e-05	1.48e-06	1.85e-07	2.37e-08
Order	-	3.18	3.06	3.02	3.00	2.97

Table 2.6: The new third order accurate, unconditionally stable, six-stage scheme (2.6) & (2.29) on the one-dimensional Allen-Cahn equation (2.31) with a traveling wave solution.

Next, we consider the Allen-Cahn equation (2.31) in two space dimensions, with the potential $W(u) = u^2(1-u)^2$ that has equal depth wells; see fig. 2.5. We take the initial condition $u(x, y, 0) = \frac{1}{1+\exp[-(7.5-\sqrt{x^2+y^2})]}$ on the domain $x \in [-10, 10]^2$, and impose periodic boundary conditions. We run the system to find u at $t = 20$, (fig. 2.7 shows u at $t = 0$ and $t = 20$). As a proxy for the exact solution of the equation with this initial data, we compute a very highly accurate numerical approximation $u_*(x, y, t)$ via the following second order accurate in time, semi-implicit, multi-step

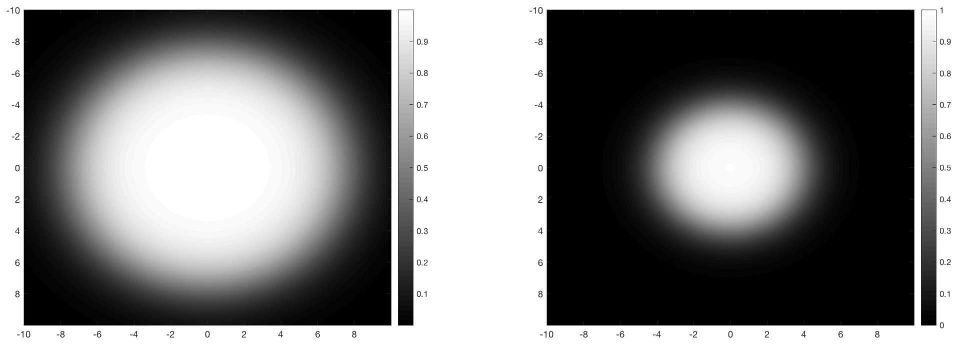

Figure 2.7: Initial condition and the solution at final time for the 2D Allen-Cahn equation with a potential that has equal depth wells.

Number of time steps	2^5	2^6	2^7	2^8	2^9	2^{10}
L2	3.43e-03	8.73e-04	2.21e-04	5.55e-05	1.39e-05	3.49e-06
Order	-	1.98	1.98	1.99	1.99	2.00

Table 2.7: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.27) on the two-dimensional Allen-Cahn equation (2.31) with a potential that has equal depth wells.

scheme [7] on an extremely fine spatial grid and take very small time steps:

$$\frac{3}{2}u^{n+1} - 2u^n + \frac{1}{2}u^{n-1} = k\Delta u^{n+1} - k(2W'(u^n) - W'(u^{n-1})).$$

Table 2.7 and table 2.8 show the errors and convergence rates for the approximate solutions computed by our new multi-stage schemes.

Number of time steps	2^3	2^4	2^5	2^6	2^7	2^8
L2	4.60e-03	5.41e-04	6.44e-05	7.98e-06	1.02e-06	1.33e-07
Order	-	3.09	3.07	3.01	2.97	2.94

Table 2.8: The new third order accurate, unconditionally stable, six-stage scheme (2.6) & (2.29) on the two-dimensional Allen-Cahn equation (2.31) with a potential that has equal depth wells.

As a final example, we consider the Cahn-Hilliard equation

$$u_t = -\Delta\big(\Delta u - W'(u)\big) \tag{2.33}$$

where we take W to be the double well potential $W(u) = u^2(1-u)^2$ with equal depth

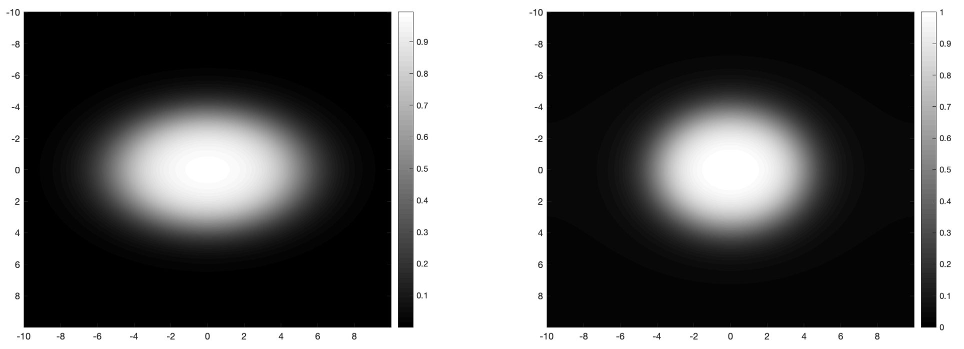

Figure 2.8: Initial condition and the solution at final time for the 2D Cahn-Hilliard equation with a potential that has equal depth wells.

Number of time steps	2^2	2^3	2^4	2^5	2^6
L2	1.16e-03	2.62e-04	6.41e-05	1.64e-05	4.22e-06
Order	-	2.15	2.03	1.97	1.95

Table 2.9: The new second order accurate, unconditionally stable, three-stage scheme (2.6) & (2.27) on the two-dimensional Cahn-Hilliard equation (2.33) with a potential that has equal depth wells.

wells and impose periodic boundary conditions. This flow is also gradient descent for energy (2.32), but with respect to the H^{-1} inner product:

$$\langle u\,,v\,\rangle = \int u\Delta^{-1}v\,dx.$$

Starting from the initial condition $u(x,y,0) = \frac{1}{1+\exp[-(5-\sqrt{x^2+2y^2})]}$ and running the system until $t = 20$ (see fig. 2.8). We computed a proxy for the "exact" solution once again using the second order accurate, semi-implicit multi-step scheme from [7] [43]:

$$\frac{3}{2}u^{n+1} - 2u^n + \frac{1}{2}u^{n-1} = -k\Delta[\Delta u^{n+1} - k(2W'(u^n) - W'(u^{n-1}))]$$

where the spatial and temporal resolution was taken to be high to ensure the errors are low. Table 2.9 and table 2.10 show the errors and convergence rates for the approximate solutions computed by our new multi-stage schemes.

Remark II.6. As further evidence of the generality and flexibility of the new schemes introduced in this chapter, we note that they can also be used to jack up the order of

Number of time steps	2^2	2^3	2^4	2^5
L2	2.20e-04	4.12e-05	6.73e-06	1.05e-06
Order	-	2.42	2.62	2.67

Table 2.10: The new third order accurate, unconditionally stable, six-stage scheme (2.6) & (2.27) on the two-dimensional Cahn-Hilliard equation (2.33) with a potential that has equal depth wells.

accuracy in time of less conventional numerical algorithms such as *threshold dynamics* [29, 30]. Also known as diffusion generated motion, threshold dynamics is an unconditionally stable algorithm for simulating the motion of interfaces by mean curvature, merely by alternating the two simple steps of convolution and thresholding. It was given a variational formulation in [15] that exhibits it as carrying out an approximate minimizing movements procedure at every time step. Although the stability calculation of section 2.2 applies verbatim, the consistency calculations of section 2.3 have to be redone. This is because (a) motion by mean curvature, although formally a gradient flow on perimeter, does not quite fit the classical formulation (2.1), and (b) the variational formulation in [15] shows that threshold dynamics carries out minimizing movements for *approximately* the right energy with respect to *approximately* the right metric: these additional errors have to be taken into account. Due to the substantial modifications to the consistency calculation required, extension of the new schemes to enhancing the order of accuracy of threshold dynamics will be taken up in chapter IV.

2.6 Conclusion

We presented a class of unconditionally stable, high order in time schemes for gradient flows. The new schemes can be thought of as a variational analogue of Richardson extrapolation: they enable jacking up the order of accuracy of standard backward Euler method, while maintaining its unconditional stability, at the expense of taking multiple backward Euler time substeps per full time step. What results is a universal method to jack up the accuracy to at least third order in time whenever a blackbox implementation of the standard backward Euler scheme is available, while increasing overall complexity by only a constant factor. We demonstrated the method and its advertised accuracy on a number of linear and nonlinear ODEs and PDEs.

Whether this class of schemes can be used to achieve arbitrarily high (i.e. ≥ 4) order in time accuracy will be the topic of a future investigation.

CHAPTER III

Variational Extrapolation of Semi-Implicit Schemes for General Gradient Flows for Fixed and Solution Dependent Inner Products

3.1 Introduction

This chapter is an extension of the ideas we developed in chapter II. Once again, we are concerned with numerical schemes for evolution equations that arise as gradient flow for an energy $f : H \to \mathbb{R}$, where H is a Hilbert space with inner product $\langle \cdot, \cdot \rangle$:

$$u' = -\nabla_H E(u). \tag{3.1}$$

Additionally, we will study gradient flows with a solution dependent inner product:

$$u' = -\mathcal{L}(u)\nabla_H E(u) \tag{3.2}$$

where $\mathcal{L}(u)$ is a positive definite operator that depends on u. One property of (3.1) and (3.2) is dissipation: $\frac{d}{dt}E(u) \leq 0$, to see this

$$\frac{d}{dt}E(u) = \langle \nabla_H E(u), u' \rangle = -\langle \nabla_H E(u), \mathcal{L}(u)\nabla_H E(u) \rangle \leq 0.$$

In chapter II, we focused on unconditionally stable numerical methods to solve (3.1). In this chapter, our first focus is on conditionally energy stable, semi-implicit methods. Specifically, let

$$E(u) = E_1(u) + E_2(u)$$

where in our numerical implementation we will handle E_1 implicitly and E_2 explicitly. Our numerical methods will guarantee that when the time step is less than a constant depending only on E_2 the following numerical dissipation property will hold:

$$E(u_{n+1}) \leq E(u_n) \tag{3.3}$$

where u_n denotes the approximation to the solution at the n-th time step. In the context of PDEs, where H is infinite dimensional, we are concerned with discrete in time, continuum in space schemes.

A basic semi-implicit scheme for the abstract equation (3.1), with time step size $k > 0$, reads

$$\frac{u_{n+1} - u_n}{k} = -\nabla_H E_1(u_{n+1}) - \nabla_H E_2(u_n). \tag{3.4}$$

Let $L_2(u, u_n)$ be the linearization of E_2 around u_n so

$$L_2(u, u_n) = E_2(u_n) + \langle \nabla_H E_2(u_n), u - u_n \rangle$$

Then (3.4) is the Euler-Lagrange equation for the optimization problem

$$u_{n+1} = \arg\min_u E_1(u) + L_2(u, u_n) + \frac{1}{2k} \|u - u_n\|^2 \tag{3.5}$$

where $\| \cdot \|^2 = \langle \cdot, \cdot \rangle$. For

$$\Lambda = \max\{0, \max_{u, \|v\|=1} D^2 E_2(u)\big(v, v\big)\} \tag{3.6}$$

where by $D^2 E_2(u)\big(v, w\big)$ we mean $\frac{d^2}{d\epsilon_2 d\epsilon_1} E_2(u + \epsilon_1 v + \epsilon_2 w)$ we have

$$E_2(u) \leq L_2(u, p) + \frac{\Lambda}{2} \|u - p\|^2 . \tag{3.7}$$

for any u and p. It follows that when $k \leq \frac{1}{\Lambda}$

$$\begin{aligned} E(u_{n+1}) = E_1(u_{n+1}) + E_2(u_{n+1}) &\leq E_1(u_{n+1}) + L_2(u_{n+1}, u_n) + \frac{1}{2k} \|u_{n+1} - u_n\|^2 \\ &\leq E_1(u_n) + L_2(u_n, u_n) + \frac{1}{2k} \|u_n - u_n\|^2 = E_1(u_n) + E_2(u_n) = E(u_n) \end{aligned}$$

so that scheme (3.4) is conditionally stable. These methods are equivalent to implicit-explicit additive Runge-Kutta (ARK IMEX) [1]. There are unconditionally energy stable ARK IMEX that rely on convexity splitting, the case where E_1 convex and E_2 concave [44]. Our class of methods have no assumption (e.g. convexity) on the implicit and explicit energies (E_1 and E_2), are conditional energy stability and high (at least up to third) order accuracy. Additionally, each time step requires a few standard minimizing movements solves, equivalent to semi-implicit substeps. This allows our schemes to effortlessly increase the order of existing stable semi-implicit methods.

Our second focus is on extending implicit and semi-implicit methods for general gradient flows to solve (3.2), the case when the inner product is solution dependent. There are stable methods for (3.2) on case by case basis, for example Cahn-Hillard with degenerate mobility [8, 22] and the porous medium equation [10, 11, 47]. To our knowledge, this is the first time energy stable methods for general gradient flows with solution dependent inner products have been considered.

The rest of the chapter is organized as follows:

- Section 3.2 presents and proves the conditions for conditional energy stability for our schemes.
- Section 3.3 states the consistency equations for the ARK IMEX schemes for solving gradient flows (3.1) and gives 2nd and 3rd order examples.
- Section 3.4 gives 2nd and 3rd order methods for solving gradient flows with solution dependent inner product (3.2) and provides consistency calculations.
- Section 3.5 presents numerical convergence studies several of well-known partial differential equations that are gradient flows.

The code for section 3.5 is publicly available, and can be found at `https://github.`

com/AZaitzeff/SIgradflow.

3.2 Stability of Our New Schemes

In this section, we formulate a wide class of numerical schemes that are energy stable by construction. The first of these schemes are Implicit-Explicit Additive Runge-Kutta (ARK IMEX) schemes, but we will write the schemes in what we will call minimizing movements form in order to prove energy stability more easily. The minimizing movement form of the M-stage of an ARK IMEX scheme is:

1. Set $U_0 = u_n$.
2. For $m = 1, \dots, M$:

$$U_m = \arg\min_u \left(E_1(u) + \sum_{i=0}^{m-1} \theta_{m,i} L_2(u, U_i) + \sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} \|u - U_i\|^2 \right). \tag{3.8}$$

   where

$$L_2(u,p) = E_2(p) + \langle \nabla_H E_2(p), u - p \rangle \tag{3.9}$$

3. Set $u_{n+1} = U_M$.

The schemes for solving a gradient flow with solution dependent inner product are a series of embedded ARK IMEX methods. The norm is fixed for each ARK IMEX step allowing the stability results of this section to apply schemes for solving (3.2). Now we establish quite broad conditions on the coefficients $\gamma_{m,i}$ $\theta_{m,i}$ that ensure conditional energy dissipation (3.3). Before we state and prove the conditions in generality, consider the following two-stage special case of scheme (3.8):

$$U_1 = \arg\min_u \left(E_1(u) + L_2(u, u_n) + \frac{\gamma_{1,0}}{2k} \|u - u_n\|^2 \right) \tag{3.10}$$

$$\begin{aligned} u_{n+1} = \arg\min_u \Bigg(& E_2(u) + \theta_{2,1} L_2(u, U_1) + \theta_{2,0} L_2(u, u_n) \\ & + \frac{\gamma_{2,0}}{2k} \|u - u_n\|^2 + \frac{\gamma_{2,1}}{2k} \|u - U_1\|^2 \Bigg) \end{aligned} \tag{3.11}$$

Let $\Lambda = \max\{0, \max_{x,\|v\|=1} D^2 E_2(x)\big(v, v\big)\}$. Note that this implies

$$E_2(u) \leq L_2(u, p) + \frac{\Lambda}{2} \|u - p\|^2 \tag{3.12}$$

for any u and p. Also note that $L_2(u, u) = E_2(u)$. Impose the conditions

$$\begin{aligned} & \gamma_{1,0} - k\Lambda - \frac{(\gamma_{2,0} - k\Lambda\theta_{2,0})^2}{(\gamma_{2,0} + \gamma_{2,1} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1})} \geq 0, \\ & \gamma_{2,1} + \gamma_{2,0} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1} > 0, \\ & \theta_{2,1} + \theta_{2,0} = 1 \text{ and} \\ & \theta_{2,1}, \theta_{2,0} \geq 0 \end{aligned} \tag{3.13}$$

on the parameters. Set $\mu = \frac{\gamma_{2,0} - k\Lambda\theta_{2,0}}{\gamma_{2,0} + \gamma_{2,1} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1}}$. First note that (3.11) is equivalent to

$$\begin{aligned} u_{n+1} = \arg\min_u & E_1(u) + \theta_{2,1} L_2(u, U_1) + \theta_{2,0} L_2(u, u_n) + \theta_{2,0}\Lambda \|u - u_n\|^2 + \\ & \theta_{2,1}\Lambda \|u - U_1\|^2 + \frac{\gamma_{2,0} + \gamma_{2,1} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1}}{2k} \left\|u - \big(\mu u_n + (1 - \mu) U_1\big)\right\|^2. \end{aligned} \tag{3.14}$$

This can be seen by expanding the norm squared and comparing the quadratic and linear terms in u. With these tools in hand we can prove energy dissipation:

$$
\begin{aligned}
&E(u_{n+1})\\
=&E_1(u_{n+1}) + E_2(u_{n+1})\\
\leq&E_1(u_{n+1}) + \theta_{2,1}[L_2(u_{n+1}, U_1) + \frac{\Lambda}{2}\left\|u_{n+1} - U_1\right\|^2]\\
&+ \theta_{2,0}[L_2(u_{n+1}, u_n) + \frac{\Lambda}{2}\left\|u_{n+1} - u_n\right\|^2] && \text{(by (3.12))}\\
\leq&E_1(u_{n+1}) + \theta_{2,1}[L_2(u_{n+1}, U_1) + \frac{\Lambda}{2}\left\|u_{n+1} - U_1\right\|^2]\\
&+ \theta_{2,0}[L_2(u_{n+1}, u_n) + \frac{\Lambda}{2}\|u_{n+1} - u_n\|^2]\\
&+ \frac{\gamma_{2,0} + \gamma_{2,1} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1}}{2k}\left\|u_{n+1} - \big(\mu u_n + (1-\mu)U_1\big)\right\|^2. && \text{(by (3.13))}\\
\leq&E_1(U_1) + \theta_{2,1}E_2(U_1) + \theta_{2,0}[L_2(U_1, u_n) + \frac{\Lambda}{2}\|U_1 - u_n\|^2]\\
&+ \frac{\gamma_{2,0} + \gamma_{2,1} - \Lambda\theta_{2,0} - \Lambda\theta_{2,1}}{2k}\left\|U_1 - \big(\mu u_n + (1-\mu)U_1\big)\right\|^2 && \text{(by (3.14))}\\
\leq&E_1(U_1) + \theta_{2,1}[L_2(U_1, u_n) + \frac{\Lambda}{2}\|u_{n+1} - u_n\|^2]\\
&+ \theta_{2,0}[L_2(U_1, u_n) + \frac{\Lambda}{2}\|U_1 - u_n\|^2]\\
&+ \frac{(\gamma_{2,0} - k\theta_{2,0})^2}{(\gamma_{2,0} + \gamma_{2,1} - k\Lambda\theta_{2,0} - k\Lambda\theta_{2,1})2k}\left\|U_1 - u_n\right\|^2\\
\leq&E_1(U_1) + [L_2(U_1, u_n) + \frac{\Lambda}{2}\|U_1 - u_n\|^2] + \frac{\gamma_{1,0} - k\Lambda}{2k}\left\|U_1 - u_n\right\|^2 && \text{(by (3.12))}\\
\leq&E(u_n). && \text{(by (3.10))}
\end{aligned}
$$

The first two conditions (3.13) require k to below a certain threshold. Hence the dissipation of (3.10) & (3.11) is conditional.

We will now extend this discussion to general, M-stage case of scheme (3.8):

Theorem III.1. *Fix a time step k. Define $\Lambda = \max\{0, \max_{x,\|v\|=1} D^2E_2(x)\big(v, v\big)\}$ and the following auxiliary quantities in terms of the coefficients $\gamma_{m,i}$ and $\theta_{m,i}$ of* *scheme* (3.8)*:*

$$\tilde{\gamma}_{m,i} = \gamma_{m,i} - k\Lambda\theta_{m,i} - \sum_{j=m+1}^{M} \tilde{\gamma}_{j,i}\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}} \tag{3.15}$$

$$\tilde{S}_{j,m} = \sum_{i=0}^{m-1} \tilde{\gamma}_{j,i} \tag{3.16}$$

If $\tilde{S}_{m,m} > 0$ *for* $m = 1, \ldots, M$*,* $\theta_{m-1,i} \geq \theta_{m,i} \geq 0$ *and* $\sum_{i=0}^{m-1} \theta_{m,i} = 1$*, then scheme* (3.8) *satisfies the energy stability condition* (3.3)*: For every* $n = 0, 1, 2, \ldots$ *we have* $E(u_{n+1}) \leq E(u_n)$*.*

As we will see in section 3.3, the conditions on the parameters $\gamma_{i,j}$ and $\theta_{m,i}$ of scheme (3.8) imposed in theorem III.1 are loose enough to enable meeting consistency conditions to high order. We will establish theorem III.1 with the help of a couple of lemmas:

Lemma III.2. *Let the auxiliary quantities* $\tilde{S}_{j,m}$*, and* $\tilde{\gamma}_{m,i}$ *be defined as in theorem III.1. We have*

$$\begin{aligned} &\arg\min E(u) + \sum_{i=0}^{m-1} \theta_{m,i} L_2(u, U_i) + \sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} \|u - U_i\|^2 \\ =&\arg\min E(u) + \sum_{i=0}^{m-1} \theta_{m,i}[L_2(u, U_i) + \frac{\Lambda}{2}\|u - U_i\|^2] + \frac{1}{2k}\sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} \left\| u - \sum_{i=0}^{m-1} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i \right\|^2 \end{aligned}$$

Proof. As in the two step case the proof consists of expanding the norm squared terms and showing that all the quadratic and linear terms of u are equal. First, the expansion of $\sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k}\|u - U_i\|^2$ is

$$\frac{\|u\|^2}{2k} \sum_{i=0}^{m-1} \gamma_{m,i} - \frac{1}{k}\langle u, \sum_{i=0}^{m-1} \gamma_{m,i} U_i\rangle + \text{terms that do not depend on } u. \tag{3.17}$$

Next, we will establish two identities to help us expand

$$\frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\|u-\sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\|^2.$$

First by rearranging (3.15),

$$\gamma_{m,i}-k\Lambda\theta_{m,i}=\sum_{j=m}^{M}\tilde{\gamma}_{j,i}\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}. \tag{3.18}$$

Next, an identity of $\tilde{S}_{m,m}$:

$$\begin{aligned}\tilde{S}_{m,m}&=\sum_{i=0}^{m-1}\tilde{\gamma}_{m,i}=\sum_{i=0}^{m-1}\left[\gamma_{m,i}-k\Lambda\theta_{m,i}-\sum_{j=m+1}^{M}\tilde{\gamma}_{j,i}\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}\right]\\&=\sum_{i=0}^{m-1}\left[\gamma_{m,i}-k\Lambda\theta_{m,i}\right]-\sum_{j=m+1}^{M}\left[\sum_{i=0}^{m-1}\tilde{\gamma}_{j,i}\right]\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}\\&=\sum_{i=0}^{m-1}\left[\gamma_{m,i}-k\Lambda\theta_{m,i}\right]-\sum_{j=m+1}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}.\end{aligned}$$

We use this identity to establish the following:

$$\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}=\tilde{S}_{m,m}+\sum_{j=m+1}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}=\sum_{i=0}^{m-1}\left[\gamma_{m,i}-k\Lambda\theta_{m,i}\right] \tag{3.19}$$

Now we can calculate the expansion:

$$\begin{aligned}&\frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\|u-\sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\|^2+\sum_{i=0}^{m-1}\theta_{m,i}\Lambda\left\|u-U_i\right\|^2\\=&\frac{\|u\|^2}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}-\frac{1}{k}\langle u,\sum_{i=0}^{m-1}\sum_{j=m}^{M}\tilde{\gamma}_{j,i}\frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}}U_i\rangle+\frac{\Lambda}{2}\|u\|^2\sum_{i=0}^{m-1}\theta_{m,i}+\Lambda\sum_{i=0}^{m-1}\langle u,\theta_{m,i}U_i\rangle\\&+\text{terms that do not depend on }u\\=&\frac{\|u\|^2}{2k}\sum_{i=0}^{m-1}\gamma_{m,i}-\frac{1}{k}\langle u,\sum_{i=0}^{m-1}\gamma_{m,i}U_i\rangle+\text{terms that do not depend on }u.\end{aligned}$$

Where the last equality follows from (3.18) and (3.19). Since this expansion matches (3.17) up to a constant in u the proof is complete. $\square$

Lemma III.3. *Let Λ and the auxiliary quantities $\tilde{S}_{j,m}$, $\tilde{\gamma}_{m,i}$ be given in theorem III.1. Additionally, let $\tilde{S}_{m,m} > 0$ for $m = 1, \ldots, M$. Then*

$$\begin{aligned}
&E_1(U_m) + \sum_{i=0}^{m-1} \theta_{m,i}[L_2(U_m, U_i) + \frac{\Lambda}{2}\left\|U_m - U_i\right\|^2] + \frac{1}{2k}\sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\left\|U_m - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\right\|^2 \\
\le& E_1(U_{m-1}) + \sum_{i=0}^{m-2} \theta_{m-1,i}[L_2(U_{m-1}, U_i) + \frac{\Lambda}{2}\left\|U_{m-1} - U_i\right\|^2] \\
&+ \frac{1}{2k}\sum_{j=m-1}^{M} \frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}\left\|U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i\right\|^2
\end{aligned}$$

Proof. By (3.8) and lemma III.2,

$$U_m = \arg\min_u E(u) + \sum_{i=0}^{m-1}\theta_{m,i}[L_2(u, U_i) + \Lambda\left\|u - U_i\right\|^2] + \frac{1}{2k}\sum_{j=m}^{M}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\|u - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\|^2.$$

Since U_m is the minimizer of the above optimization problem

$$\begin{aligned}
&E_1(U_m) + \sum_{i=0}^{m-1} \theta_{m,i}[L_2(U_m, U_i) + \frac{\Lambda}{2}\left\|U_m - U_i\right\|^2] \\
&+ \frac{1}{2k}\sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\left\|U_m - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\right\|^2 \\
\le& E_1(U_{m-1}) + \theta_{m,m-1}E_2(U_{m-1}) + \sum_{i=0}^{m-2} \theta_{m,i}[L_2(U_{m-1}, U_i) + \frac{\Lambda}{2}\left\|U_{m-1} - U_i\right\|^2] \\
&+ \frac{1}{2k}\sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}}\left\|U_{m-1} - \sum_{i=0}^{m-1}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}}U_i\right\|^2
\end{aligned} \tag{3.20}$$

We give two inequalities to aid us in the proof. First, using the definition of the auxiliary variables, we can state an identity that will simplify (3.20). For $m > 1$ and $j \geq m$

$$\begin{aligned}\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} \left\| U_{m-1} - \sum_{i=0}^{m-1} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i \right\|^2 &= \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} \left\| U_{m-1}\left(1 - \frac{\tilde{\gamma}_{j,m-1}}{\tilde{S}_{j,m}}\right) - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i \right\|^2 \\ =\frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} \left\| U_{m-1}\left(\frac{\tilde{S}_{j,m-1}}{\tilde{S}_{j,m}}\right) - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i \right\|^2 &= \frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}} \left\| U_{m-1} - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}} U_i \right\|^2 .\end{aligned} \tag{3.21}$$

Now since $\tilde{S}_{m-1,m-1} > 0$,

$$\frac{\tilde{S}_{m-1,m-1}^2}{\tilde{S}_{m-1,m-1}} \left\| U_{m-1} - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{m-1,i}}{\tilde{S}_{m-1,m-1}} U_i \right\|^2 > 0. \tag{3.22}$$

Using (3.21) and (3.22) we have

$$\begin{aligned}\frac{1}{2k} \sum_{j=m}^{M} \frac{\tilde{S}_{j,m}^2}{\tilde{S}_{j,j}} \left\| U_{m-1} - \sum_{i=0}^{m-1} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m}} U_i \right\|^2 &= \frac{1}{2k} \sum_{j=m}^{M} \frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}} \left\| U_{m-1} - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}} U_i \right\|^2 \\ &\leq \frac{1}{2k} \sum_{j=m-1}^{M} \frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}} \left\| U_{m-1} - \sum_{i=0}^{m-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}} U_i \right\|^2 .\end{aligned} \tag{3.23}$$

Next, since $\sum_{i=1}^{m-1} \theta_{m,i} = 1$ for all m we have the equality

$$\theta_{m,m-1} = 1 - \sum_{i=0}^{m-2} \theta_{m,i} = \sum_{i=0}^{m-2} \theta_{m-1,i} - \sum_{i=0}^{m-2} \theta_{m,i} \tag{3.24}$$

Using (3.12) and (3.24), we have our second inequality:

$$\begin{aligned}
&\theta_{m,m-1}E_2(U_{m-1}) + \sum_{i=0}^{m-2}\theta_{m,i}[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2] \\
&= \sum_{i=0}^{m-2}(\theta_{m-1,i}-\theta_{m,i})E_2(U_{m-1}) + \sum_{i=0}^{m-2}\theta_{m,i}[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2] \\
&\le \sum_{i=0}^{m-2}(\theta_{m-1,i}-\theta_{m,i})[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2] \qquad (3.25)\\
&\quad + \sum_{i=0}^{m-2}\theta_{m,i}[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2] \\
&= \sum_{i=0}^{m-2}\theta_{m-1,i}[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2]
\end{aligned}$$

Using inequalities (3.23) and (3.25), we have that (3.20) is less than or equal to

$$E_1(U_{m-1}) + \sum_{i=0}^{m-2}\theta_{m-1,i}[L_2(U_{m-1},U_i) + \frac{\Lambda}{2}\left\|U_{m-1}-U_i\right\|^2] \\ + \frac{1}{2k}\sum_{j=m-1}^{M}\frac{\tilde{S}_{j,m-1}^2}{\tilde{S}_{j,j}}\left\|U_{m-1} - \sum_{i=0}^{m-2}\frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,m-1}}U_i\right\|^2$$

concluding the proof. □

Proof. (of theorem) The main idea of the proof is to use lemma III.3 repeatedly to relate the energy of $E(u_{n+1})$ to $E(u_n)$. First, by (3.12) and our assumption that $\tilde{S}_{M,M} > 0$

$$\begin{aligned}
E(u_{n+1}) &= E_1(U_M) + E_2(U_M) \\
&\le E_1(U_M) + \sum_{i=0}^{M-1}\theta_{M,i}[L_2(U_M,U_i) + \frac{\Lambda}{2}\left\|U_M-U_i\right\|^2] \\
&\quad + \frac{1}{2k}\frac{\tilde{S}_{M,M}^2}{\tilde{S}_{M,M}}\|U_M - \sum_{i=0}^{M-1}\frac{\tilde{\gamma}_{M,i}}{\tilde{S}_{M,M}}U_i\|^2.
\end{aligned}$$

By using the lemma III.3 repeatedly we have

$$
\begin{aligned}
&E_1(U_M) + \sum_{i=0}^{M-1} \theta_{M,i}[L_2(U_M, U_i) + \frac{\Lambda}{2} \|U_M - U_i\|^2] + \frac{1}{2k} \frac{\tilde{S}_{M,M}^2}{\tilde{S}_{M,M}} \|U_M - \sum_{i=0}^{M-1} \frac{\tilde{\gamma}_{M,i}}{\tilde{S}_{M,M}} U_i\|^2 \\
\le & E_1(U_{M-1}) + \sum_{i=0}^{M-2} \theta_{M-1,i}[L_2(U_{M-1}, U_i) + \frac{\Lambda}{2} \|U_{M-1} - U_i\|^2] \\
& + \frac{1}{2k} \sum_{j=M-1}^{M} \frac{\tilde{S}_{j,M-1}^2}{\tilde{S}_{j,j}} \left\| U_{M-1} - \sum_{i=0}^{M-2} \frac{\tilde{\gamma}_{j,i}}{\tilde{S}_{j,M-1}} U_i \right\|^2 \\
& \vdots \\
\le & E_1(U_1) + L_2(U_1, U_0) + \frac{\Lambda}{2} \|U_1 - U_0\|^2 + \frac{1}{2k} \sum_{j=1}^{M} \frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}} \|U_1 - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}} U_0\|^2.
\end{aligned}
$$

By (3.8) and lemma III.2

$$
U_1 = \arg\min_u E_1(u) + L_2(u, U_0) + \frac{\Lambda}{2} \|u - U_0\|^2 + \frac{1}{2k} \sum_{j=1}^{M} \frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}} \|u - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}} U_0\|^2
$$

so

$$
\begin{aligned}
&E_1(U_1) + L_2(U_1, U_0) + \frac{\Lambda}{2} \|U_1 - U_0\|^2 + \frac{1}{2k} \sum_{j=1}^{M} \frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}} \|U_1 - \frac{\tilde{\gamma}_{j,0}}{\tilde{S}_{j,1}} U_0\|^2 \\
\le & E_1(U_0) + E_2(U_0) + \frac{\Lambda}{2} \|U_0 - U_0\|^2 + \frac{1}{2k} \sum_{j=1}^{M} \frac{\tilde{S}_{j,1}^2}{\tilde{S}_{j,j}} \|U_0 - U_0\|^2 \\
= & E(u_n)
\end{aligned}
$$

completing the proof of the theorem. □

Remark III.4. In the above proof, we assume that $E_2(u)$ is two times differentiable. You can drop this assumption if you replace $L_2(u,p)$ with another approximation $A_2(u,p)$ that has the properties $A_2(u,u) = E_2(u)$ and for some choice Λ, $E_2(u) \le A_2(u,p) + \frac{\Lambda}{2} \|u - p\|^2$ for all u and p.

3.3 Examples of the New Schemes for Gradient Flows

In this section, we give examples of higher order schemes for gradient flows that are conditionally stable. First, we give the conditions on $\gamma_{m,i}$ and $\theta_{m,i}$ in scheme (3.8) to ensure its high order consistency with the abstract evolution law (3.1). Recall that $U_0 = u_n$. From (3.8), each stage U_m satisfies the Euler-Lagrange equation:

$$\left[\sum_{i=0}^{m-1} \gamma_{m,i}\right] U_m + k\nabla_H E_1(U_m) = -\sum_{i=0}^{m-1} k\theta_{m,i}\nabla_H E_2(U_i) + \sum_{i=0}^{m-1} \gamma_{m,i} U_i. \tag{3.26}$$

(3.26) is equivalent to the form more often seen for ARK IMEX methods:

$$U_m = U_0 - k\sum_{i=1}^{m} \alpha_{m,i}\nabla_H E_1(U_i) - k\sum_{i=1}^{m-1} \tilde{\alpha}_{m,i}\nabla_H E_2(U_i) \tag{3.27}$$

where $\alpha_{m,i}$ and $\tilde{\alpha}_{m,i}$ depend on γ_{m_i} and $\theta_{m,i}$. The consistency equations for ARK IMEX methods have been previously worked out [26, 35, 44, 49]. As such, we will state without proof the conditions required to achieve various orders of accuracy in terms of γ and θ:

Claim III.5. *Let U_i be given in (3.8). The Taylor expansion of U_i at each stage has the form:*

(3.28)

$$\begin{aligned} U_i = U_0 &- \beta_{1,i}kDE(U_0) + k^2\left[\beta_{2,i}k^2D^2E_1(U_0)DE(U_0) + \beta_{3,i}D^2E_2(U_0)DE(U_0)\right] \\ &- k^3\big[\beta_{4,i}D^2E_1(U_0)\left(D^2E_1(U_0)\left(DE(U_0)\right)\right) + \beta_{5,i}D^2E_1(U_0)\left(D^2E_2(U_0)\left(DE(U_0)\right)\right) \\ &+ \beta_{6,i}D^2E_2(U_0)\left(D^2E_1(U_0)\left(DE(U_0)\right)\right) + \beta_{7,i}D^2E_2(U_0)\left(D^2E_2(U_0)\left(DE(U_0)\right)\right) \\ &+ \beta_{8,i}D^3E_1(U_0)\big(DE(U_0), DE(U_0)\big) + \beta_{9,i}D^3E_2(U_0)\big(DE(U_0), DE(U_0)\big)\big] + h.o.t. \end{aligned}$$

where for $l \in \{1, 2, 3, \ldots\}$, $D^lE(u) : H^l \to \mathbb{R}$ denotes the multilinear form given by

$$D^l E(u)\big(v_1, \ldots, v_l\big) = \frac{\partial^l}{\partial s_1 \cdots \partial s_l} E(u + s_1 v_1 + s_2 v_2 + \cdots + s_l v_l)\Big|_{s_1 = s_2 = \cdots = s_l = 0}$$

so that the linear functional $D^l E(u)\big(v_1, v_2, \ldots, v_{l-1}, \cdot\big) : H \to \mathbb{R}$ *may be identified with an element of* H*, and so on. The coefficients of* (3.28) *obey the following recursive relation:*

$$\begin{aligned}
&\beta_{1,0} = \beta_{2,0} = \ldots = \beta_{9,0} = 0 \\
&\beta_{1,m} = \frac{1}{S_m}\left[1 + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{1,i}\right] \\
&\beta_{2,m} = \frac{1}{S_m}\left[\beta_{1,m} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{2,i}\right] \\
&\beta_{3,m} = \frac{1}{S_m}\left[\sum_{i=0}^{m-1} \theta_{m,i}\beta_{1,i} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{3,i}\right] \\
&\beta_{4,m} = \frac{1}{S_m}\left[\beta_{2,m} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{4,i}\right] \\
&\beta_{5,m} = \frac{1}{S_m}\left[\beta_{3,m} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{5,i}\right] \\
&\beta_{6,m} = \frac{1}{S_m}\left[\sum_{i=0}^{m-1} \theta_{m,i}\beta_{2,i} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{6,i}\right] \\
&\beta_{7,m} = \frac{1}{S_m}\left[\sum_{i=0}^{m-1} \theta_{m,i}\beta_{3,i} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{7,i}\right] \\
&\beta_{8,m} = \frac{1}{S_m}\left[\frac{\beta_{1,m}^2}{2} + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{8,i}\right] \\
&\beta_{9,m} = \frac{1}{S_m}\left[\frac{1}{2}\sum_{i=0}^{m-1} \theta_{m,i}\beta_{1,i}^2 + \sum_{i=1}^{m-1} \gamma_{m,i}\beta_{9,i}\right]
\end{aligned} \tag{3.29}$$

with $S_m = \sum_{i=0}^{m-1} \gamma_{m,i}$*. Furthermore, the following conditions for* $u_{n+1} = U_M$ *in scheme* (3.8) *are necessary and sufficient for various orders of accuracy:*

$$\begin{array}{llll} & \underline{\textit{First Order:}} & \underline{\textit{Second Order:}} & \underline{\textit{Third Order:}} \\ & \beta_{1,M} = 1 & \beta_{1,M} = 1 & \beta_{1,M} = 1 \\ (3.30) & & \beta_{2,M} = 1/2 & \beta_{2,M} = 1/2 \\ & & \beta_{3,M} = 1/2 & \beta_{3,M} = 1/2 \\ & & & \beta_{4,M} = \beta_{5,M} = \ldots = \beta_{9,M} = 1/6 \end{array}$$

Now, we give second order and a third order example of method (3.8). However, The examples we give are not unique by any means. We begin with a five step method that is second order accurate:

$$\theta \approx \begin{pmatrix} 1. & 0 & 0 & 0 & 0 \\ 0.009 & 0.991 & 0 & 0 & 0 \\ 0.009 & 0.991 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1. & 0 \\ 0 & 0 & 0 & 1. & 0 \end{pmatrix}$$

$$\gamma \approx \begin{pmatrix} 8.841 & 0 & 0 & 0 & 0 \\ -0.925 & 5.360 & 0 & 0 & 0 \\ -4.443 & 6.041 & 0.950 & 0 & 0 \\ -3.288 & 5.895 & -0.351 & 0.172 & 0 \\ -3.895 & -0.335 & 4.964 & -1.722 & 7.684 \end{pmatrix} \tag{3.31}$$

which is stable for $k\Lambda \leq 3/872$.

Next we have a thirteen step method that is third order accurate:

$$\theta \approx \begin{pmatrix} 1. & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.049 & 0.951 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.024 & 0.075 & 0.901 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.017 & 0.042 & 0.113 & 0.829 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.012 & 0.029 & 0.071 & 0.386 & 0.501 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.01 & 0.023 & 0.06 & 0.366 & 0.457 & 0.085 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.007 & 0.018 & 0.05 & 0.351 & 0.437 & 0.06 & 0.076 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.003 & 0.005 & 0.006 & 0.008 & 0.009 & 0.011 & 0.028 & 0.929 & 0 & 0 & 0 & 0 & 0 \\ 0.002 & 0.002 & 0.002 & 0.002 & 0.003 & 0.004 & 0.009 & 0.029 & 0.948 & 0 & 0 & 0 & 0 \\ 0 & 0.001 & 0.001 & 0.001 & 0.001 & 0.002 & 0.004 & 0.007 & 0.011 & 0.971 & 0 & 0 & 0 \\ 0 & 0 & 0.001 & 0.001 & 0.001 & 0.001 & 0.003 & 0.005 & 0.008 & 0.912 & 0.069 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.001 & 0.002 & 0.003 & 0.005 & 0.107 & 0.025 & 0.857 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.001 & 0.001 & 0.002 & 0.013 & 0.007 & 0.018 & 0.958 \end{pmatrix}$$

$$\gamma \approx \begin{pmatrix} 11. & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2.1 & 15.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.4 & 1.6 & 17. & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 1.6 & -2.4 & 18.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3 & -8.5 & 3. & 9.6 & 7.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1.4 & -5.9 & -0.1 & 2. & 8. & 4.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -4. & -0.5 & -0.4 & -1.8 & 5.1 & 6.8 & 0.9 & 0 & 0 & 0 & 0 & 0 & 0 \\ -9.2 & 4.8 & 2.7 & -3.2 & 2.5 & 6.2 & 2.5 & 4.6 & 0 & 0 & 0 & 0 & 0 \\ -1.7 & -3.6 & -0.1 & 1.3 & 5.7 & 3.4 & -0.8 & -0.8 & 0.4 & 0 & 0 & 0 & 0 \\ -2.7 & -3.5 & 0.6 & 1.4 & 6.1 & 3.5 & -0.7 & -0.2 & -0.4 & 0.5 & 0 & 0 & 0 \\ 5.9 & -4.8 & -5.1 & -3.1 & 3.4 & 6.6 & -0.7 & -5.2 & 4.9 & -0.8 & 8.2 & 0 & 0 \\ 7.1 & 0.9 & -3.1 & -2.7 & -5.8 & -1.9 & 0.6 & -3.4 & 4.3 & -1.3 & 9.2 & 9.1 & 0 \\ 3.8 & 1.9 & 2.7 & 2.1 & -7.5 & -10.6 & -1.2 & 2. & 0.7 & -0.2 & -0.2 & 9.5 & 12.8 \end{pmatrix} \tag{3.32}$$

which is stable if $k\Lambda \leq 18/28567$. The coefficients to machine precision as well as code to verify theorem III.1 and claim III.5 can be found at `https://github.com/AZaitzeff/SIgradflow`. In the following section, we consider methods for (3.2), when the inner product changes with the solution.

3.4 Schemes for Solving Gradient Flows with Solution Dependent Inner Product

Now we move on to the problem of simulating flow (3.2),

$$u' = -\mathcal{L}(u)\nabla_H E(u).$$

We consider the case where $\mathcal{L}(u)$ is strictly positive definite. Our approach will be as follows:

1. Generate a u_* from u_n.
2. Construct $\mathcal{L}(u_*)$.
3. Use the algorithm (3.8) with norm $\|\cdot\|^2_{\mathcal{L}^{-1}(u_*)} = \langle \cdot, \mathcal{L}^{-1}(u_*)\cdot\rangle$ to generate u_{n+1}.

One advantage to constructing $\mathcal{L}(u_*)$ and then using it in (3.8) is that theorem III.1 immediately gives conditional energy stability for coefficients such as (3.31) or (3.32). Thus, we only need to consider what choice of u_* will give our algorithm the desired level of accuracy. Now at every step we are solving

(3.33)
$$\left[\sum_{i=0}^{m-1}\gamma_{m,i}\right]U_m + k\mathcal{L}(u_*)\nabla_H E_1(U_m) = -k\mathcal{L}(u_*)\sum_{i=0}^{m-1}\theta_{m,i}\nabla_H E_2(U_i) + \sum_{i=0}^{m-1}\gamma_{m,i}U_i.$$

We will set up the consistency equations for (3.2). Let $u_n = u(t_0)$. For convenience, denote $\mathcal{L}(u_n)$ as $\mathcal{L}_n$ and $E(u_n)$ as E_n. We begin with the exact solution starting from $u(t_0)$:

$$\begin{cases} u_t = -\nabla E(u) & t > t_0 \\ u(t_0) = U_0 \end{cases}$$

By Taylor expanding around t_0 we find

$$u(k+t_0) = u(t_0) + ku_t(t_0) + \frac{1}{2}k^2 u_{tt}(t_0) + \frac{1}{6}k^3 u_{ttt}(t_0) + O(k^4) \tag{3.34}$$

where the higher derivatives in time are found using (3.2):

$$\begin{aligned}
u_t(t_0) =& -\mathcal{L}_n DE_n \\
u_{tt}(t_0) =& D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n + \mathcal{L}_n D^2 E_n(\mathcal{L}_n DE_n) \\
u_{ttt}(t_0) =& -D\mathcal{L}_n(D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n)DE_n - D^2\mathcal{L}_n(\mathcal{L}_n DE_n, \mathcal{L}_n DE_n)DE_n \\
& - D\mathcal{L}_n(\mathcal{L}_n(D^2E_n(\mathcal{L}_n DE_n)))DE_n - 2D\mathcal{L}_n(\mathcal{L}_n DE_n)D^2E_n(\mathcal{L}_n DE_n) \\
& - \mathcal{L}_n D^2 E_n(D\mathcal{L}_n(\mathcal{L}_n DE_n))DE_n - \mathcal{L}_n D^2E_n(\mathcal{L}_n D^2 E_n(\mathcal{L}_n DE_n)) \\
& - \mathcal{L}_n D^3 E_n\big(\mathcal{L}_n DE_n, \mathcal{L}_n DE_n\big)
\end{aligned}$$

where for $l \in \{1, 2, 3, \ldots\}$, $D^l L(u) : H^l \to H$ denotes the multilinear form given by

$$D^l L(u)\big(v_1, \ldots, v_l\big) = \frac{\partial^l}{\partial s_1 \cdots \partial s_l} L(u + s_1 v_1 + s_2 v_2 + \cdots + s_l v_l)\bigg|_{s_1=s_2=\cdots=s_l=0}$$

so that $D^l L(u)\big(v_1, v_2, \ldots, v_l\big)$ is a linear operator from H to H.

In the next two subsections, we provide second and third order examples and accompanying consistency calculations. Both of these examples also have the additional property that $E(u_*) \le E(u_n)$.

3.4.1 Second Order Method

Our second order algorithm is laid out in algorithm 4. Now we will prove that it is indeed second order. First, the expansion of u_* is

Algorithm 4 A second order method for solving gradient flows with solution dependent inner product

```
Fix a time step size $k > 0$. Set $u_n = u_0$. Alternate the following steps:
    1. Find $u_*$:
        $$u_* + \frac{1}{2}k\mathcal{L}_n\nabla E_1(u_*) = u_n - \frac{1}{2}k\mathcal{L}_n\nabla E_2(u_n)$$
    2. Find $u_{n+1}$ using (3.33) with coefficients (3.31) and $\mathcal{L}(u_*)$ in the movement limiter.
```

$$u_* = u_n - \frac{1}{2}k\mathcal{L}_n DE_n + O(k^2) \tag{3.35}$$

We use (3.29) to get an expansion of u_{n+1}:

$$u_{n+1} = u_n - k\mathcal{L}(u_*)DE_n + \frac{1}{2}k^2\mathcal{L}(u_*)D^2E_n(\mathcal{L}(u_*)DE_n) + O(k^3) \tag{3.36}$$

Now, expand u_* around u_n in (3.36):

$$\begin{aligned} u_{n+1} =& u_n - k\mathcal{L}_n DE_n - kD\mathcal{L}_n(u_* - u_n)DE_n \\ &+ \frac{1}{2}k^2\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n) + O(k^3) \\ =& u_n - k\mathcal{L}_n DE_n + \frac{1}{2}k^2 D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n \\ &+ \frac{1}{2}k^2\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n) + O(k^3) \end{aligned}$$

The Taylor expansion of u_{n+1} matches (3.34) to second order.

3.4.2 Third Order Method

Now we present our third order algorithm for solving (3.2). It requires the use of two new sets of coefficients,

$$\begin{aligned} \theta &\approx \begin{pmatrix} 1. & 0 & 0. \\ -0.667 & 0.333 & 0 \\ 0 & 0 & 1.000 \end{pmatrix} \\ \gamma &\approx \begin{pmatrix} 1.833 & 0 & 0. \\ 0.556 & 0.667 & 0 \\ 1.030 & -0.026 & 0.159 \end{pmatrix} \end{aligned} \tag{3.37}$$

Algorithm 5 A third order method for solving gradient flows with solution dependent inner product

Fix a time step size $k > 0$ and set $u_n = u_0$. For convenience, we will denote $D^2\mathcal{L}(u_*)\big(\mathcal{L}(u_*)\nabla E(u_*), \mathcal{L}(u_*)\nabla E(u_*)\big)$ as $D^2\mathcal{L}(u_*)$.
Alternate the following steps:

1. Find u_{*_1} using (3.33) with the coefficients (3.37), $\mathcal{L}_n$ in the movement limiter and time step $\frac{1}{6}k$.
2. Find $\bar{u}$ using (3.33) with coefficients (3.32), $\mathcal{L}(u_{*_1}) - \frac{1}{72}k^2 D^2\mathcal{L}(u_{*_1})$ in the movement limiter and time step $\frac{1}{2}k$.
3. Find $u_{*_{2,1}}$:
$$u_{*_{2,1}} + \frac{2}{5}k\mathcal{L}_n\nabla E_1(u_{*_{2,1}}) = u_n - \frac{2}{5}k\mathcal{L}_n\nabla E_2(u_n).$$
4. Find $u_{*_{2,2}}$ using (3.33) with the coefficients (3.38), $\mathcal{L}(u_{*_{2,1}})$ in the movement limiter and time step $\frac{5}{6}k$.
5. Find u_{n+1} using (3.33) starting at $\bar{u}$ (instead of u_n) with coefficients (3.32), $\mathcal{L}(u_{*_{2,2}}) - \frac{1}{72}k^2 D^2\mathcal{L}(u_{*_{2,2}})$ in the movement limiter and time step $\frac{1}{2}k$.

and

$$\theta \approx \begin{pmatrix} 1. & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.708 & 0.292 & 0 & 0 & 0 & 0 & 0 \\ 0.013 & 0.018 & 0.969 & 0 & 0 & 0 & 0 \\ 0.008 & 0.012 & 0.867 & 0.113 & 0 & 0 & 0 \\ 0.006 & 0.009 & 0.206 & 0.056 & 0.724 & 0 & 0 \\ 0 & 0.005 & 0.05 & 0.025 & 0.053 & 0.867 & 0 \\ 0 & 0 & 0.015 & 0.009 & 0.015 & 0.04 & 0.920 \end{pmatrix}$$

$$(3.38) \qquad \gamma \approx \begin{pmatrix} 7.727 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.594 & 2.241 & 0 & 0 & 0 & 0 & 0 \\ 3.056 & -0.455 & 0.636 & 0 & 0 & 0 & 0 \\ -1.571 & 5.091 & -1.063 & 2.786 & 0 & 0 & 0 \\ -3.714 & 3.1 & -1.267 & 1.545 & 9.655 & 0 & 0 \\ -6.923 & 5.1 & -2.056 & 3.471 & 4.571 & 4.033 & 0 \\ -2.467 & -2.1 & 0.009 & -0.182 & 0.660 & 7.224 & 9.428 \end{pmatrix},$$

to achieve particular Taylor expansions. The values of (3.37) and (3.38) to machine precision can be found at `https://github.com/AZaitzeff/SIgradflow`.

Algorithm 5 details our third order version for solving gradient flows with solution dependent inner product. The method adds another condition for stability to hold, namely:

$$(3.39) \qquad \mathcal{L}(u) - \frac{1}{72}k^2 D^2\mathcal{L}(u)(w,w)$$

needs to be positive definite for all u and w. Now we will prove that algorithm 5 produces a third order approximation.

By applying (3.29), the coefficients (3.37) give the following expansion for u_{*_1}:

$$u_{*_1} = u_n - \frac{1}{6}k\mathcal{L}_n DE_n + \frac{1}{36}k^2\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n) + O(k^3) \tag{3.40}$$

Now we can expand $\bar{u}$ by using (3.29) and expanding u_{*_1} around u_n

$$\begin{aligned}
\bar{u} &= u_n - \frac{1}{2}k\mathcal{L}(u_{*_1})DE_n + \frac{1}{8}k^2\mathcal{L}(u_{*_1})D^2E_n(\mathcal{L}(u_{*_1})DE_n) \\
&- \frac{1}{48}k^3\mathcal{L}(u_{*_1})D^2E_n\left(\mathcal{L}(u_{*_1})D^2E_n\left(\mathcal{L}(u_{*_1})DE_n\right)\right) \\
&- \frac{1}{48}k^3\mathcal{L}(u_{*_1})D^3E_n\big(\mathcal{L}(u_{*_1})DE_n, \mathcal{L}(u_*)DE_n\big) \\
&+ \frac{1}{144}k^3D^2\mathcal{L}(u_{*_1})\big(\mathcal{L}(u_{*_1})DE(u_{*_1}), \mathcal{L}(u_{*_1})DE(u_{*_1})\big)DE_n + O(k^4) \\
&= u_n - \frac{1}{2}k\mathcal{L}_n DE_n + \frac{1}{12}k^2D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n + \frac{1}{8}k^2\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n) \\
&- \frac{1}{72}k^3D\mathcal{L}_n(\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n))DE_n \\
&- \frac{1}{48}k^3\mathcal{L}_n D^2E_n(D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n) \\
&- \frac{1}{48}k^3D\mathcal{L}_n(\mathcal{L}_n DE_n)D^2E_n(\mathcal{L}_n DE_n) \\
&- \frac{1}{48}k^3\mathcal{L}_n D^2E_n\left(\mathcal{L}_n D^2E_n\left(\mathcal{L}_n DE_n\right)\right) \\
&- \frac{1}{48}k^3\mathcal{L}_n D^3E_n\big(\mathcal{L}_n DE_n, \mathcal{L}_n DE_n\big) + O(k^4)
\end{aligned} \tag{3.41}$$

Now we will apply the same steps to derive the expansions of $u_{*_{2,1}}$

$$u_{*_{2,1}} = u_n - \frac{2}{5}k\mathcal{L}_n DE_n + O(k^2) \tag{3.42}$$

and $u_{*_{2,2}}$

$$\begin{aligned} u_{*_{2,2}} &= u_n - \frac{5}{6}k\mathcal{L}(u_{*_{2,1}})DE_n + \frac{11}{36}k^2\mathcal{L}(u_{*_{2,1}})D^2E_n(\mathcal{L}(u_{*_{2,1}})DE_n) + O(k^3) \\ &= u_n - \frac{5}{6}k\mathcal{L}_nDE_n \\ &+ \frac{1}{3}k^2D\mathcal{L}_n(\mathcal{L}_nDE_n)DE_n + \frac{11}{36}k^2\mathcal{L}_nD^2E_n(\mathcal{L}_nDE_n) + O(k^3) \end{aligned} \tag{3.43}$$

Finally, we can find the expansion of u_{n+1}. We will first apply (3.29) around $\bar{u}$

$$\begin{aligned} u_{n+1} = \bar{u} &- \frac{1}{2}k\mathcal{L}(u_{*_{2,2}})DE(\bar{u}) + \frac{1}{8}k^2\mathcal{L}(u_{*_{2,2}})D^2E(\bar{u})(\mathcal{L}(u_{*_{2,2}})DE(\bar{u})) \\ &- \frac{1}{48}k^3\mathcal{L}(u_{*_{2,2}})D^2E(\bar{u})\left(\mathcal{L}(u_{*_{2,2}})D^2E(\bar{u})\left(\mathcal{L}(u_{*_{2,2}})DE(\bar{u})\right)\right) \\ &- \frac{1}{48}k^3\mathcal{L}(u_{*_{2,2}})D^3E(\bar{u})\big(\mathcal{L}(u_{*_{2,2}})DE(\bar{u}), \mathcal{L}(u_{*_{2,2}})DE(\bar{u})\big) \\ &+ \frac{1}{144}k^3D^2\mathcal{L}(u_{*_{2,2}})\big(\mathcal{L}(u_{*_{2,2}})DE(u_{*_{2,2}}), \mathcal{L}(u_{*_{2,2}})DE(u_{*_{2,2}})\big)DE(\bar{u}) + O(k^4) \end{aligned}$$

expand $u_{*_{2,2}}$

$$\begin{aligned} u_{n+1} = \bar{u} &- \frac{1}{2}k\mathcal{L}_nDE(\bar{u}) + \frac{5}{12}k^2D\mathcal{L}_n(\mathcal{L}_nDE_n)DE(\bar{u}) + \frac{1}{8}k^2\mathcal{L}_nD^2E(\bar{u})(\mathcal{L}_nDE(\bar{u})) \\ &- \frac{1}{6}k^3D\mathcal{L}_n(D\mathcal{L}_n(\mathcal{L}_nDE_n)DE_n)DE(\bar{u}) \\ &- \frac{1}{6}k^3D^2\mathcal{L}_n(\mathcal{L}_nDE_n, \mathcal{L}_nDE_n)DE_n)DE(\bar{u}) \\ &- \frac{11}{72}k^3D\mathcal{L}_n(\mathcal{L}_nD^2E_n(\mathcal{L}_nDE_n))DE(\bar{u}) \\ &- \frac{5}{48}k^3D\mathcal{L}_n(\mathcal{L}_nDE(\bar{u}))D^2E(\bar{u})(\mathcal{L}_nDE(\bar{u})) \\ &- \frac{5}{48}k^3\mathcal{L}_nD^2E(\bar{u})(D\mathcal{L}_n(\mathcal{L}_nDE(\bar{u}))DE(\bar{u})) \\ &- \frac{1}{48}k^3\mathcal{L}_nD^2E(\bar{u})\left(\mathcal{L}_nD^2E(\bar{u})\left(\mathcal{L}_nDE(\bar{u})\right)\right) \\ &- \frac{1}{48}k^3\mathcal{L}_nD^3E(\bar{u})\big(\mathcal{L}_nDE(\bar{u}), \mathcal{L}_nDE(\bar{u})\big) + O(k^4) \end{aligned}$$

then expand $\bar{u}$ around u_n:

$$\begin{aligned}
u_{n+1} = u_n &- k\mathcal{L}_n DE_n + \frac{1}{2}k^2 D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n + \frac{1}{2}k^2\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n) \\
&- \frac{1}{6}k^3 D\mathcal{L}_n(D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n)DE_n \\
&- \frac{1}{6}k^3 D^2\mathcal{L}_n(\mathcal{L}_n DE_n, \mathcal{L}_n DE_n)DE_n)DE_n \\
&- \frac{1}{6}k^3 D\mathcal{L}_n(\mathcal{L}_n D^2E_n(\mathcal{L}_n DE_n))DE_n \\
&- \frac{1}{3}k^3 D\mathcal{L}_n(\mathcal{L}_n DE_n)D^2E_n(\mathcal{L}_n DE_n) \\
&- \frac{1}{6}k^3\mathcal{L}_n D^2E_n(D\mathcal{L}_n(\mathcal{L}_n DE_n)DE_n) \\
&- \frac{1}{6}k^3\mathcal{L}_n D^2E_n\left(\mathcal{L}_n D^2E_n\left(\mathcal{L}_n DE_n\right)\right) \\
&- \frac{1}{6}k^3\mathcal{L}_n D^3E_n\big(\mathcal{L}_n DE_n, \mathcal{L}_n DE_n\big) + O(k^4)
\end{aligned}$$

The Taylor expansion of u_{n+1} matches (3.34) to third order. As long as (3.39) holds,

$$E(u_{n+1}) \le E(\bar{u}) \le E(u_n)$$

by theorem III.1.

Remark III.6. In algorithm 4 and algorithm 5, we can instead handle $E(u)$ fully implicitly, as we do in chapter II. We need to substitute higher order implicit methods for the corresponding semi-implicit methods. In this case, the second order method is unconditionally stable and the third order is unconditionally stable if $D^2\mathcal{L}(u)\big(w,w\big)$ is negative semi-definite for all u and w. We detail the 3rd order, fully implicit algorithm for gradient flows with solution dependent inner product in the appendix chapter B.

3.5 Numerical Examples

In this section, we will apply the second and third order accurate conditionally stable schemes to a variety of gradient flows, some with fixed inner product and some with solution dependent inner product. Careful numerical convergence studies are presented in each case to verify the anticipated convergence rates of previous sections.

3.5.1 Gradient Flows with Fixed Inner Product

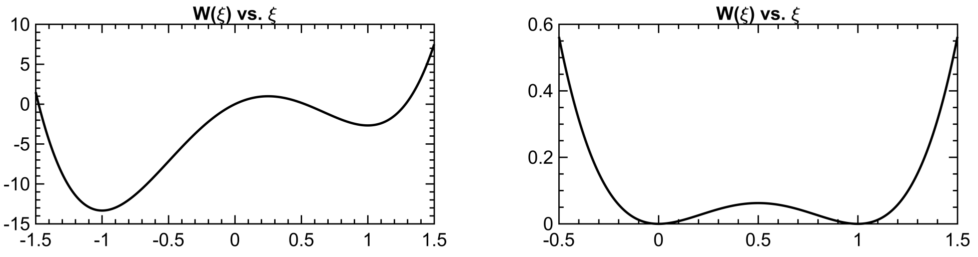

Figure 3.1: The double well potentials used in the Allen-Cahn (3.44) and Cahn-Hilliard (3.46) equations: One with unequal depth wells and the other with equal depth wells.

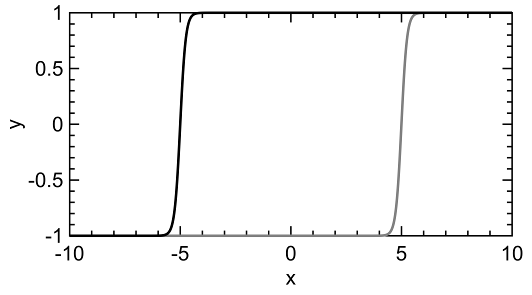

Figure 3.2: The initial condition (black) and the solution at final time (gray) in the numerical convergence study on the 1D Allen-Cahn equation (3.44) with a potential that has unequal depth wells.

We start with the Allen-Cahn equation

$$u_t = \Delta u - W'(u) \tag{3.44}$$

where $W : \mathbb{R} \to \mathbb{R}$ is a double-well potential. This corresponds to gradient flow for

Number of time steps	2^9	2^{10}	2^{11}	2^{12}	2^{13}
L^2 error (2nd order)	2.08e-01	5.96e-02	1.61e-02	4.22e-03	1.08e-03
Order	-	1.81	1.89	1.94	1.97
L^2 error (3rd order)	2.06e-03	3.26e-04	4.68e-05	6.32e-06	8.33e-07
Order	-	2.66	2.80	2.89	2.92

Table 3.1: The new second (3.31) and third (3.32) order accurate, conditionally stable schemes (3.8) on the one-dimensional Allen-Cahn equation (3.44) with a traveling wave solution.

the energy

$$E(u) = \int \frac{1}{2}\|\nabla u\|^2 + W(u)\,dx \tag{3.45}$$

with respect to the L^2 inner product.

First, we consider equation (3.44) in one space dimension, with the potential $W(u) = 8u - 16u^2 - \frac{8}{3}u^3 + 8u^4$. This is a double well potential with unequal depth wells; see fig. 3.1. In this case, equation (3.44) is well-known to possess traveling wave solutions on $x \in \mathbb{R}$, see fig. 3.2. We choose the initial condition $u(x,0) = \tanh(4x+20)$; the exact solution is then $u_*(x,t) = \tanh(4x+20-8t)$. The computational domain is $x \in [-10,10]$, discretized into a uniform grid of 8193 points. We approximate the solution on $\mathbb{R}$ by using the Dirichlet boundary conditions $u(\pm 10, t) = \pm 1$: The domain size is large enough that the mismatch in boundary conditions do not substantially contribute to the error in the approximate solution over the time interval $t \in [0,5]$. We use $E_1(u) = \int \frac{1}{2}|\nabla u|^2 dx$ and $E_2(u) = \int W(u) dx$. Table 3.1 tabulates the error in the computed solution at time $T = 5$ for our two new schemes.

Next, we consider the Allen-Cahn equation (3.44) in two space-dimensions, with the potential $W(u) = u^2(1-u)^2$ that has equal depth wells; see fig. 3.1. We take the initial condition $u(x,y,0) = \frac{1}{1+\exp[-(7.5-\sqrt{x^2+y^2})]}$ on the domain $x \in [-10,10]^2$, and impose periodic boundary conditions. Once again we use $E_1(u) = \int \frac{1}{2}\|\nabla u\|^2 dx$

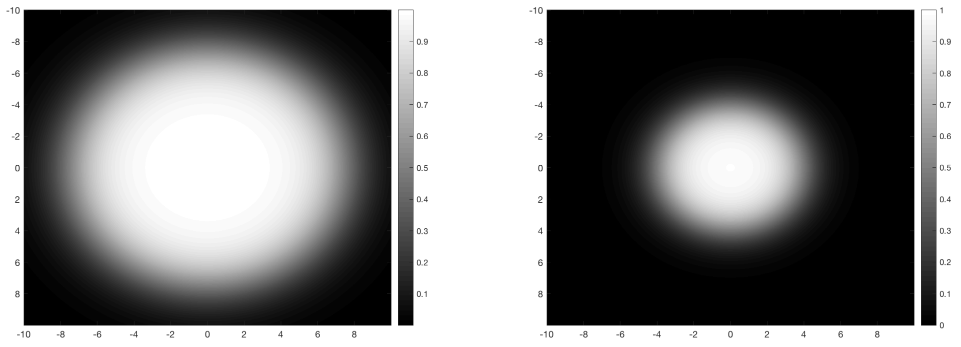

Figure 3.3: Initial condition and the solution at final time for the 2D Allen-Cahn equation with a potential that has equal depth wells.

and $E_2(u) = \int W(u)dx$. As a proxy for the exact solution of the equation with this initial data, we compute a very highly accurate numerical approximation $u_*(x, y, t)$ via the following second order accurate in time, semi-implicit, multi-step scheme [7] on an extremely fine spatial grid and take very small time steps:

$$\frac{3}{2}u^{n+1} - 2u^n + \frac{1}{2}u^{n-1} = k\Delta u^{n+1} - k(2W'(u^n) - W'(u^{n-1})).$$

This method has not been proven to be energy stable, but instead satisfies a less strict stability property. Table 3.2 show the errors and convergence rates for the approximate solutions computed by our new multi-stage schemes.

Number of time steps	2^8	2^9	2^{10}	2^{11}	2^{12}
L^2 error (2nd order)	3.62e-05	9.07e-06	2.27e-06	5.68e-07	1.41e-07
Order	-	2.00	2.00	2.00	2.00
L^2 error (3rd order)	2.35e-05	3.18e-06	4.15e-07	5.29e-08	6.24e-09
Order	-	2.88	2.94	2.97	3.08

Table 3.2: The new second (3.31) and third (3.32) order accurate, conditionally stable schemes (3.8) on the two-dimensional Allen-Cahn equation (3.44) with a potential that has equal depth wells.

As a final example, we consider the Cahn-Hilliard equation

$$u_t = -\Delta\big(\Delta u - W'(u)\big) \tag{3.46}$$

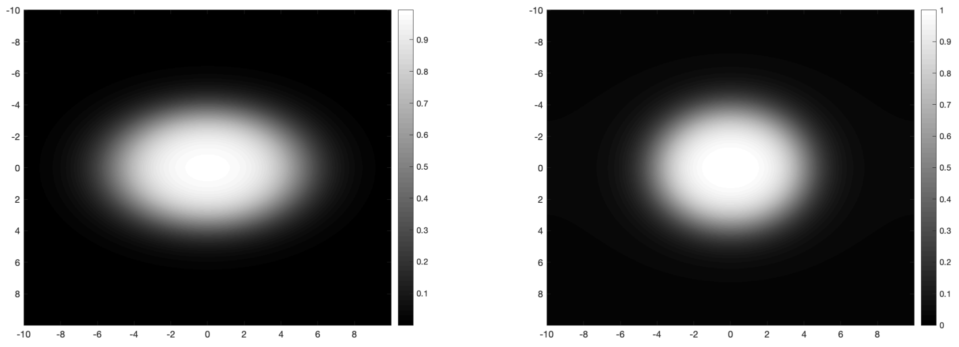

Figure 3.4: Initial condition and the solution at final time for the 2D Cahn-Hillard equation with a potential that has equal depth wells.

where we take W to be the double well potential $W(u) = u^2(1-u)^2$ with equal depth wells and impose periodic boundary conditions. This flow is also gradient descent for energy (3.45), but with respect to the H^{-1} inner product:

$$\langle u \, , v \rangle = \int u\Delta^{-1}v \, dx.$$

Starting from the initial condition $u(x,y,0) = \frac{1}{1+\exp[-(5-\sqrt{x^2+y^2})]}$, we computed a proxy for the "exact" solution once again using the second order accurate, semi-implicit multi-step scheme from [7]:

$$\frac{3}{2}u^{n+1} - 2u^n + \frac{1}{2}u^{n-1} = -k\Delta[\Delta u^{n+1} - (2W'(u^n) - W'(u^{n-1}))]$$

where the spatial and temporal resolution was taken to be high to ensure the errors are small. Table 3.3 show the errors and convergence rates for the approximate solutions computed by our new multi-stage schemes.

3.5.2 Gradient Flow For Solution Dependent Inner Product

Our first example we present in this section is the heat equation, $u_t = \Delta u$, but with a different energy. Under the Wasserstein metric (denoted as W_2), the heat

Number of time steps	2^7	2^8	2^9	2^{10}	2^{11}
L^2 error (2nd order)	6.20e-04	1.92e-04	5.59e-05	1.55e-05	4.09e-06
Order	-	1.69	1.78	1.85	1.92
L^2 error (3rd order)	6.45e-06	1.35e-06	2.51e-07	4.15e-08	7.20e-09
Order	-	2.25	2.43	2.60	2.53

Table 3.3: The new second (3.31) and third (3.32) order accurate, conditionally stable schemes (3.8) on the two-dimensional Cahn-Hilliard equation (3.46) with a potential that has equal depth wells.

equation is a gradient flow for the negative entropy [25]:

$$E(u) = \int u \log(u) dx. \tag{3.47}$$

However the minimization

$$\arg\min_u E(u) + \frac{1}{2k} W_2^2(u, u_n)$$

is a difficult optimization problem. On the other hand, we can approximate the the Wasserstein metric, $W_2(u, v)$, with

$$\langle u - v, \mathcal{L}(u)^{-1}(u - v)\rangle_{L^2} \text{ where } \mathcal{L}(u) = -\nabla \cdot u \nabla \tag{3.48}$$

when u and v are near each other. Indeed

$$-\mathcal{L}(u)\nabla_{L^2} E(u) = \nabla \cdot u \nabla(\log(u) + 1) = \Delta u.$$

Thus, we can alternatively think of the heat equation as minimizing movements on negative entropy with respect to the solution dependent inner product (3.48) and therefore use algorithm 4 and algorithm 5 to evolve the heat equation while decreasing the negative entropy (3.47) at every step.

We use the exact solution $u(x, t) = \cos(\pi x) \exp(-t\pi^2) + 2$ as our test with domain $x \in [0, 1]$ using derivative zero Neumann boundary conditions. Our initial data is $u(x, 0)$ and we run the simulation to final time $T = \frac{1}{10}$. We use $E_1(u) = \frac{1}{2} \int u^2 dx$

Number of time steps	2^3	2^4	2^5	2^6	2^7
L^2 error (2nd order)	1.06e-03	3.11e-04	8.58e-05	2.27e-05	5.85e-06
Order	-	1.77	1.86	1.92	1.96
L^2 error (3rd order)	1.00e-05	1.57e-06	2.20e-07	3.04e-08	4.29e-09
Order	-	2.69	2.82	2.87	2.83

Table 3.4: The new second (algorithm 4) and third (algorithm 5) order accurate, conditionally stable schemes for gradient flows with solution dependent inner product on the negative entropy (3.47) with $\mathcal{L}(u) = -\nabla \cdot u\nabla$ in the inner product to simulate the heat equation.

and $E_2(u) = \int u \log(u)dx - \frac{1}{2}\int u^2 dx$ in (3.8) so at every step we are solving a linear systems of equation. We run simulation for $T = \frac{1}{10}$. See table 3.4 for results.

The next example is the porous medium equation in one dimension,

$$u_t = \Delta u^2.$$

It is a gradient flow for the energy

$$E(u) = \int u^2 dx$$

under the Wasserstein metric. As with the heat equation, we can again replace the Wasserstein movement limiter with (3.48). Since ∇E is linear, we let $E_1(u) = E(u)$ and $E_2(u) = 0$. Our initial data is

$$u(x,0) = \begin{cases} \exp \frac{-1}{1-x^2} + \frac{1}{100} & \text{if } |x| < 1 \\ \frac{1}{100} & \text{otherwise} \end{cases}$$

in $x \in [-3, 3]$ with derivative zero Newman boundary conditions. We run the simulation for $T = 2$. See fig. 3.5 for our initial and final curve. We generate the 'true' solution using a explicit second order predictor corrector method with an high spatial and temporal resolution. See table 3.5 for results.

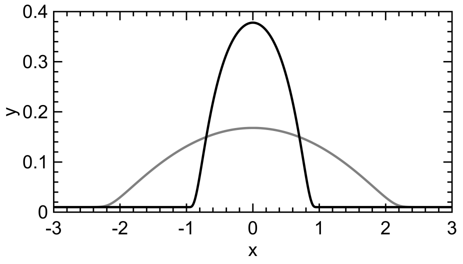

Figure 3.5: The initial (black) and final (grey) data for our porous medium example.

Number of time steps	2^5	2^6	2^7	2^8	2^9
L^2 error (2nd order)	5.28e-05	1.16e-05	2.71e-06	6.58e-07	1.62e-07
Order	-	2.19	2.09	2.04	2.02
L^2 error (3rd order)	1.11e-06	7.44e-07	1.62e-07	2.51e-08	3.36e-09
Order	-	0.57	2.21	2.68	2.90

Table 3.5: The new second and third order accurate, unconditionally stable schemes (see remark III.6) for gradient flows with solution dependent inner product on energy $\int u^2 dx$ with $\mathcal{L}(u) = -\nabla \cdot u\nabla$ in the inner product to simulate the porous medium equation.

3.6 Conclusion

We presented a new class of implicit-explicit additive Runge-Kutta schemes for gradient flows that are high order and conditionally stable. Additionally, we developed new high order stable schemes for gradient flows on solution dependent inner products. Both of these methods allow us to painlessly increase the order of accuracy of existing schemes for gradient flows without sacrificing stability. We provided many numerical examples of gradient flows, including those that have solution dependent inner product, and shown the methods achieve their advertised accuracy. Now whether these schemes can be used to achieve arbitrarily high (i.e. ≥ 4) order in time is left to future work.

CHAPTER IV

Second Order Threshold Dynamics Schemes for Two Phase Motion by Mean Curvature

4.1 Introduction

In this chapter, we will describe new, second order accurate in time versions of a popular algorithm for simulating the motion of interfaces by mean curvature known as threshold dynamics. The original version of the algorithm, which is only first order accurate in time in the two-phase setting, was proposed by Merriman, Bence, and Osher in [29, 30]. Since then, many extensions of the algorithm have been given, for instance to multiphase mean curvature motion, where it has proven particularly useful and flexible. There have also been high order accurate versions of the algorithm proposed in several previous studies, discussed in detail in section 4.2.

For a $(d-1)$-dimensional smooth interface $\Gamma \subset \mathbb{R}^d$ given as the boundary of a set $\Sigma \subset \mathbb{R}^d$, the original threshold dynamics algorithm generates a discrete in time approximation to its motion by mean curvature as follows: Here, $G_{\delta t}$ is the Gaussian

Algorithm 6 Original Threshold Dynamics of MBO'92

Fix a time step size $\delta t > 0$. Alternate the following steps:

1. Convolution:
$$\phi(x) = G_{\delta t} * \mathbf{1}_{\Sigma^k}.$$
2. Thresholding:
$$\Sigma^{k+1} = \Big\{ x : \phi(x) \geq \frac{1}{2} \Big\}.$$

kernel:

$$G_{\delta t}(\mathbf{x}) = \frac{1}{(4\pi\delta t)^{d/2}} e^{-\frac{|\mathbf{x}|}{4\delta t}}$$

Our goal in this chapter is to take a step towards providing more accurate versions of threshold dynamics. The accuracy issue is particularly acute in the multi-phase setting, where it decreases to half-order in time due to the presence of junctions. Here, we focus on the easier yet still challenging two-phase setting, to find a version of algorithm 6 that

- Maintains the simplicity and spirit of the original threshold dynamics algorithm (algorithm 6),
- Achieves second order accuracy in time,
- Maintains the variational interpretation, and the resulting stability properties, given in [15] for the original algorithm.

The chapter is organized as follows:

- In section 4.2, we recall previous efforts in designing second order versions of threshold dynamics.
- In section 4.3, we discuss necessary conditions for second order accuracy.
- In section 4.4, we present our first new algorithm: a natural two kernel extrapolation method, applied to the original threshold dynamics algorithm, to achieve second order accuracy in any space dimension.
- In section 4.5, we present our second new algorithm: a multi-step method that is second order accurate in two space dimensions, and unconditionally energy stable in any dimension.
- In section 4.6, we provide numerical verification of the advertised order of accuracy for both of our new algorithms.

The code for section 4.6 is publicly available, and can be found at `https://github.com/AZaitzeff/secondorderTD`.

4.2 Previous Work

In [38], Ruuth proposed the following method based on Richardson extrapolation to jack up the order of accuracy of algorithm 6 to second order in time:

Algorithm 7 Ruuth's Second Order Threshold Dynamics

Fix a time step size $\delta t > 0$. Set $\phi^k(x) = \mathbf{1}_{\Sigma^0}(x)$. Alternate the following steps:

1. First half time step:
$$\Sigma_1 = \Big\{x \,:\, G_{\delta t/2} * \phi^k \geq \frac{1}{2}\Big\}.$$
2. Second half time step:
$$\Sigma_2 = \Big\{x \,:\, G_{\delta t/2} * \mathbf{1}_{\Sigma_1} \geq \frac{1}{2}\Big\}.$$
3. Full time step:
$$\Sigma_3 = \Big\{x \,:\, G_{\delta t} * \phi^k \geq \frac{1}{2}\Big\}.$$
4. Linear combination:
$$\phi^{k+1} = \mathbf{2}\mathbf{1}_{\Sigma_2} - \mathbf{1}_{\Sigma_3}.$$

Although numerical experiments indicate this version indeed improves the accuracy in time to second order for smooth interfaces undergoing two-phase motion by mean curvature, the algorithm sacrifices an attractive simplicity of the original MBO scheme: it no longer generates binary functions exclusively that can be naturally identified with sets. Perhaps more importantly, there appears to be no clear extension of the variational interpretation given in [15] for the original algorithm 6 to this case. No comparison principle is expected to hold, as a non-positive weighted sum is involved. Hence, there is no rigorous result indicating the stability of the algorithm (or its convergence).

In [20], Grzhibovskis & Heinz propose another approach to improving the order of accuracy of algorithm 6 to second order. The idea is natural: To cancel out the leading order error in threshold dynamics by taking a linear combination of convolutions with two different radially symmetric kernels: The coefficients α and β are

Algorithm 8 Algorithm of Grzhibovskis & Heinz

1. Convolution step:
$$\phi(x) = \Big(\alpha K_1 - \beta K_2\Big) * \mathbf{1}_{\Sigma^k}.$$
2. Threhsolding step:
$$\Sigma^{k+1} = \Big\{x \,:\, \phi(x) \geq \frac{1}{2}\Big\}.$$

chosen so that the leading order correction to curvature in the standard consistency calculation for the original threshold dynamics algorithm 6 cancels out. Crucially, this necessitates that the resulting combined convolution kernel changes sign, even when the individual kernels K_1 and K_2 are positive. This means that the resulting algorithm can violate the comparison principle. But far more importantly, we show in section 4.3, that this algorithm *does not give second order accuracy* in time; it merely achieves a more accurate evaluation of the mean curvature term at every time step. In general, the dynamics generated is still only first order accurate, at least without being much more specific and deliberate about the choice of the kernels K_1 and K_2 – which the authors do not specify. (For example, in case both K_1 and K_2 are Gaussians – with potentially different mass and/or width – no choice of the coefficients α and β results in a second or higher order accurate in time scheme for motion by mean curvature.)

In this chapter, we will provide truly second order accurate in time versions of algorithm 6 that maintain its elegant and simple nature. Moreover, we will be able to provide rigorous stability results for our new algorithms.

4.3 Second Order Motion by Mean Curvature

First, we need to identify how far a surface travels under motion by mean curvature. In the vicinity of a point of interest on the surface, which we take to be the origin, let the surface be given as the graph of a smooth function $f(x, y, t)$: $\mathbb{R}^2 \times [0, \infty) \to \mathbb{R}$ with $f(0, 0, 0) = 0$, $f_x(0, 0, 0) = 0$ and $f_y(0, 0, 0) = 0$. Since the normal direction changes during the evolution, it is easier to insist that the numerically generated solution intersects a fixed line at nearly the same location as the true solution, at any given time. Thus, we will calculate how far the surface travels along the z-axis under mean curvature motion and under our algorithms. For a surface given as the graph of a function, motion by mean curvature takes the following form:

$$f_t = \frac{f_{xx}(1 + f_y^2) - 2f_x f_y f_{xy} + f_{yy}(1 + f_x^2)}{1 + f_x^2 + f_y^2}. \tag{4.1}$$

By a straightforward Taylor expansion we have for small t

$$\begin{aligned} f(0,0,t) = t\Big[f_{xx} + f_{yy}\Big] \\ + t^2\left[\frac{1}{2}(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) - (f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3)\right] + O(t^3) \end{aligned} \tag{4.2}$$

where the functions on the right hand side are evaluated at $(0, 0, 0)$. Over the course of this chapter, we will denote $f(0, 0, 0)$ as f, $f_x(0, 0, 0)$ as f_x, etc. for convenience.

It has been known and verified by Taylor expansion in previous publications (e.g. Ruuth [38]) that standard threshold dynamics is 1st order accurate. We will include the expansion of standard threshold dynamics (algorithm 6) here as a simple example of the method we use throughout this chapter. Let $\Sigma^0 = \{(x, y, z) : z \leq f(x, y, 0)\}$.

We work out the convolution of a Gaussian kernel with a characteristic function in chapter C of the appendix where we had to keep many more terms than in previous works to achieve our goals in this chapter. Applying calculation (C.11) to our problem we have:

$$\begin{aligned}
G_t * \mathbf{1}_{\Sigma^0}(0,0,z) - \frac{1}{2} &= -\frac{z}{2\sqrt{\pi t}} + \frac{z^3}{24\sqrt{\pi}t^{3/2}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(f_{xx}+f_{yy}) \\
&+ \frac{t^{3/2}}{4\sqrt{\pi}}(f_{xxxx}+2f_{xxyy}+f_{yyyy}) - \frac{z^2}{8\sqrt{\pi t}}(f_{xx}+f_{yy}) \\
&+ \frac{z\sqrt{t}}{2\sqrt{\pi}}(\frac{3}{4}f_{xx}^2 + \frac{3}{4}f_{yy}^2 + \frac{1}{2}f_{xx}f_{yy} + f_{xy}^2) \\
&- \frac{t^{3/2}}{2\pi}(\frac{5}{4}f_{xx}^3 + \frac{5}{4}f_{yy}^3 + \frac{3}{4}f_{xx}f_{yy}^2 + \frac{3}{4}f_{xx}^2 f_{yy} + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2) + \text{h.o.t.}
\end{aligned} \tag{4.3}$$

Next set (4.3) to zero and solve for z by using the ansatz $z = z_1 t + z_2 t^2 +$ remainder and matching terms of the same order in t. Up to second order:

$$\begin{aligned}
z &= t\Big[f_{xx}+f_{yy}\Big] \\
&+ t^2\Big[\frac{1}{2}(f_{xxxx}+2f_{xxyy}+f_{yyyy}) - \frac{2}{3}(f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3)\Big] + O(t^3).
\end{aligned} \tag{4.4}$$

Equation (4.4) gives the location of $\partial\Sigma^1$ along the z-axis. The equation (4.4) matches the two dimensional version calculated by Ruuth [37]. Additionally, for three dimensions some of the terms in (4.3) are calculated by Grzhibovskis & Heinz [21]. Their paper, focusing on Willmore flow, did not require all the terms calculated in (4.3). Comparing (4.4) to the location of the interface under mean curvature motion (4.2), we see that threshold dynamics is only first order accurate in time. At this point, we can also already see why algorithm 8 of Grzhibovskis & Heintz cannot be second order accurate: It would merely move the surface by $t(f_{xx}+f_{yy}) + O(t^3)$, which would still make it only a first order accurate approximation of the right hand side of (4.2). In the next two sections, we will present two second order methods. For each method, we will show that they do indeed match motion by mean curvature

(4.2) up to second order in the normal direction.

4.4 A More Natural Two Kernel Extrapolation

Our first method is a two stage algorithm using two different Gaussian kernels with differing amplitudes and widths. We detail the method in algorithm 9. Whether

Algorithm 9 Natural Two Kernel Extrapolation

Fix a time step size $\delta t > 0$. Alternate the following steps:

1. First stage:
$$\bar{\Sigma} = \left\{ x \, : \, G_{\delta t/2} * \mathbf{1}_{\Sigma^k} \geq \frac{1}{2} \right\}.$$
2. Second stage:
$$\Sigma^{k+1} = \left\{ x \, : \, \frac{\sqrt{2} G_{\delta t/2} * \mathbf{1}_{\bar{\Sigma}} - G_{\delta t} * \mathbf{1}_{\Sigma^k}}{\sqrt{2} - 1} \geq \frac{1}{2} \right\}.$$

algorithm 9 is unconditionally stable is currently unknown. We will devote the rest of the section to showing that algorithm 9 is indeed second order.

4.4.1 Consistency

Once again let $\Sigma^0 = \{(x, y, z) : z \leq f(x, y, 0)\}$ for f defined in section 4.3. First, we need to find the location and curvature of $\bar{\Sigma}$ along the z-axis. Let $h(x, y)$ be defined by the requirement that $G_{t/2} * \mathbf{1}_{\Sigma^0}(x, y, h(x, y)) = \frac{1}{2}$, so that $\bar{\Sigma} = \{(x, y, z) : z \leq h(x, y)\}$. From (4.4) we have that

$$\begin{aligned} (4.5) \quad h(0,0) &= \frac{t}{2} \Big[f_{xx} + f_{yy} \Big] \\ &+ \frac{t^2}{4} \left[\frac{1}{2}(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) - \frac{2}{3}(f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3) \right] + O(t^3) \end{aligned}$$

Now we would like to find $h_{xx}(0,0)$ and $h_{yy}(0,0)$. From (C.17) in the appendix, replacing h with f, z with h and t with $t/2$, we have that $h_{xx}(0,0)$ satisfies

$$\begin{aligned}
0 = &- \frac{\sqrt{2}h_{xx}}{2\sqrt{\pi t}} + \frac{\sqrt{2}h^2 h_{xx}}{4\sqrt{\pi}t^{3/2}} + \frac{\sqrt{2}f_{xx}}{2\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{2\pi}}(f_{xxxx} + f_{xxyy}) \\
&- \frac{\sqrt{2}h^2}{4\sqrt{\pi}t^{3/2}} f_{xx} + \frac{\sqrt{2}h}{2\sqrt{\pi t}}(\frac{3}{2}f_{xx}^2 + \frac{1}{2}f_{xx}f_{yy} + f_{xy}^2) \\
&- \frac{\sqrt{t}}{2\sqrt{2\pi}}\left(\frac{15}{4}f_{xx}^3 + \frac{3}{4}f_{xx}f_{yy}^2 + \frac{3}{2}f_{xx}^2 f_{yy} + 6f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2\right) \\
&+ \frac{\sqrt{t}h_{xx}}{2\sqrt{2\pi}}\left(\frac{3}{4}f_{xx}^2 + \frac{3}{4}f_{yy}^2 + \frac{1}{2}f_{xx}f_{yy} + f_{xy}^2\right) - \frac{\sqrt{2}hh_{xx}}{4\sqrt{\pi t}}(f_{xx} + f_{yy}) + \text{h.o.t.}
\end{aligned} \tag{4.6}$$

Plugging in h from (4.5) and solving for h_{xx} in (4.6), using the ansatz $h_{xx} = h_0 + th_1 + \text{remainder}$, we have

$$h_{xx} = f_{xx} + \frac{t}{2}(f_{xxxx} + f_{xxyy} - 2f_{xx}^3 - 4f_{xx}f_{xy}^2 - 2f_{yy}f_{xy}^2) + O(t^2)$$

We can use similar steps to find h_{yy}. Putting h_{xx} and h_{yy} together we arrive at

$$\begin{aligned}
h_{xx} + h_{yy} = &f_{xx} + f_{yy} \\
&+ \frac{t}{2}\left[(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) - 2(f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3)\right] + O(t^2).
\end{aligned} \tag{4.7}$$

Now we can solve for the location of Σ^1 along the z-axis. From the expansion of the Gaussian kernel convoluted with a characteristic function, (C.11), we have

(4.8)

$$
\begin{aligned}
&[\sqrt{2}G_{\delta t/2} * \mathbf{1}_{\bar{\Sigma}} - G_{\delta t} * \mathbf{1}_{\Sigma^0}](0,0,z) - \frac{\sqrt{2}-1}{2} \\
&= - \frac{z}{\sqrt{\pi t}} + \frac{z^3}{6\sqrt{\pi}t^{3/2}} + \frac{h}{\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(h_{xx} + h_{yy}) \\
&\quad + \frac{t^{3/2}}{8\sqrt{\pi}}(h_{xxxx} + 2h_{xxyy} + h_{yyyy}) - \frac{z^2}{2\sqrt{\pi}t^{3/2}}h - \frac{z^2}{4\sqrt{\pi t}}(h_{xx} + h_{yy}) + \frac{z}{2\sqrt{\pi}t^{3/2}}h^2 \\
&\quad + \frac{z}{2\sqrt{\pi t}}h(h_{xx} + h_{yy}) + \frac{z\sqrt{t}}{2\sqrt{\pi}}(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) - \frac{h^3}{6\sqrt{\pi}t^{3/2}} \\
&\quad - \frac{h^2}{4\sqrt{\pi t}}(h_{xx} + h_{yy}) - \frac{\sqrt{t}}{2\sqrt{\pi}}h(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) \\
&\quad - \frac{t^{3/2}}{4\sqrt{\pi}}(\frac{5}{4}h_{xx}^3 + \frac{5}{4}h_{yy}^3 + \frac{3}{4}h_{xx}h_{yy}^2 + \frac{3}{4}h_{xx}^2h_{yy} + 3h_{xx}h_{xy}^2 + 3h_{yy}h_{xy}^2) \\
&\quad + \frac{z}{2\sqrt{\pi t}} - \frac{z^3}{24\sqrt{\pi}t^{3/2}} - \frac{\sqrt{t}}{2\sqrt{\pi}}(f_{xx} + f_{yy}) \\
&\quad - \frac{t^{3/2}}{4\sqrt{\pi}}(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) + \frac{z^2}{8\sqrt{\pi t}}(f_{xx} + f_{yy}) \\
&\quad - \frac{z\sqrt{t}}{2\sqrt{\pi}}(\frac{3}{4}f_{xx}^2 + \frac{3}{4}f_{yy}^2 + \frac{1}{2}f_{xx}f_{yy} + f_{xy}^2) \\
&\quad + \frac{t^{3/2}}{2\sqrt{\pi}}(\frac{5}{4}f_{xx}^3 + \frac{5}{4}f_{yy}^3 + \frac{3}{4}f_{xx}f_{yy}^2 + \frac{3}{4}f_{xx}^2f_{yy} + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2) + \text{h.o.t}
\end{aligned}
$$

Note that the derivatives of h match the corresponding derivatives of f up to order t (stated precisely in (C.12)). Substituting (4.5) for h, (4.7) for $h_{xx} + h_{yy}$ and (C.12) for the other derivatives of h in (4.8), and simplifying, we have:

$$
\begin{aligned}
&[\sqrt{2}G_{\delta t/2} * \mathbf{1}_{\bar{\Sigma}} - G_{\delta t} * \mathbf{1}_{\Sigma^k}](0,0,z) - \frac{\sqrt{2}-1}{2} \\
&= -\frac{z}{2\sqrt{\pi t}} + \frac{z^3}{8\sqrt{\pi}t^{3/2}} + \frac{t^{3/2}}{4\sqrt{\pi}}(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) \\
&\quad - \frac{2t^{3/2}}{3\sqrt{\pi}}(f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3) + \frac{\sqrt{t}}{2\sqrt{\pi}}(f_{xx} + f_{yy}) \\
&\quad - \frac{3z^2}{8\sqrt{\pi t}}(f_{xx} + f_{yy}) + \frac{3z\sqrt{t}}{8\sqrt{\pi}}(f_{xx} + f_{yy})^2 \\
&\quad - \frac{\sqrt{t}}{12\sqrt{\pi}}(f_{xx} + f_{yy})^3 - \frac{\sqrt{t}}{4\sqrt{\pi}}(f_{xx} + f_{yy})(\frac{3}{4}f_{xx}^2 + \frac{3}{4}f_{yy}^2 + \frac{1}{2}f_{xx}f_{yy} + f_{xy}^2) \\
&\quad + \frac{t^{3/2}}{4\sqrt{\pi}}(\frac{5}{4}f_{xx}^3 + \frac{5}{4}f_{yy}^3 + \frac{3}{4}f_{xx}f_{yy}^2 + \frac{3}{4}f_{xx}^2 f_{yy} + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2) + \text{h.o.t.}
\end{aligned}
\tag{4.9}
$$

Finally, we set (4.9) to zero and solve for z up to order t^2 to obtain:

$$
\begin{aligned}
z &= t\Big[f_{xx} + f_{yy}\Big] \\
&\quad + t^2\Big[\frac{1}{2}(f_{xxxx} + 2f_{xxyy} + f_{yyyy}) - (f_{xx}^3 + 3f_{xx}f_{xy}^2 + 3f_{yy}f_{xy}^2 + f_{yy}^3)\Big] + O(t^3).
\end{aligned}
\tag{4.10}
$$

Thus, up to second order, algorithm 9 moves the interface in the normal direction by the same amount as mean curvature motion (4.2).

The drawback to algorithm 9 is that we do not know whether it is unconditionally stable. In the next section, we present an algorithm that is unconditionally stable but is only guaranteed to be second order in two dimensions.

4.5 Unconditionally Stable Multistage Methods

In this section, we provide a class of unconditionally stable threshold dynamics algorithms that are second order in two dimensions. The original threshold dynamics algorithm (algorithm 6) is unconditionally energy stable, specifically:

$$
E_t(\Sigma^{k+1}) \leq E_t(\Sigma^k) \tag{4.11}
$$

for energy

$$E_t(\Sigma) = \int_{\mathbb{R}^2} (1 - \mathbf{1}_\Sigma) G_t * \mathbf{1}_\Sigma dxdy. \tag{4.12}$$

Our class of methods preserves property (4.11) while at the same time achieving second order accuracy. We describe our M-stage method in algorithm 10.

Algorithm 10 M-Stage Unconditionally Stable Threshold Dynamics

Fix a time step size $\delta t > 0$ and a choice of γ's such that $\sum_{i=0}^{m-1} \gamma_{m,i} = 1$ for $m = 1, 2, \ldots, M$. Set $\tau = \delta t / \beta_{1,M}$ for $\beta_{1,M}$ defined in (4.17) and set $\bar{\Sigma}_0 = \Sigma^k$.

For $m = 1, 2, \ldots, M$:

$$\bar{\Sigma}_m = \Big\{ x : G_\tau * \sum_{i=0}^{m-1} \gamma_{m,i} \mathbf{1}_{\bar{\Sigma}_i} \geq \frac{1}{2} \Big\}. \tag{4.13}$$

Then set $\Sigma^{k+1} = \bar{\Sigma}_M$

Unlike the previous algorithm, algorithm 10 uses the same kernel at each stage. As will be shown later, this will allow us to prove unconditional stability (4.11). In the rest of the section, we will derive the consistency equations for γ, give the conditions on γ for unconditional stability to hold, and then give a particular choice of γ that makes algorithm 10 unconditionally stable and second order. We conclude with a discussion of one way to extend algorithm 10 to higher dimensions.

4.5.1 Consistency Equations

Similar to in three dimensions, let $f(x,t) : \mathbb{R} \times [0, \infty) \to \mathbb{R}$ be a graph that is the interface of a set in $\mathbb{R}^2$ with $f(0,0) = 0$, $f_x(0,0) = 0$. The distance the graph moves under mean curvature motion along the y-axis is:

$$f(0,t) = t f_{xx} + t^2 \left[\frac{1}{2} f_{xxxx} - f_{xx}^3 \right] + O(t^2). \tag{4.14}$$

We now present the consistency equations for algorithm 10:

Claim IV.1. *Let $\bar{\Sigma}_i$ be given in* (4.13) *and let*

$$\bar{\Sigma}_0 = \Sigma^k = \Big\{x : x \leq f(x,0)\Big\}$$

.

Define h_i as $\bar{\Sigma}_i = \Big\{x : x \leq h_i(x)\Big\}$, so $h_0(x) = f(x,0)$. The Taylor expansion of h_i at each stage has the same form as (4.14)*, namely:*

$$h_i(0) = t\beta_{1,i} f_{xx} + t^2(\beta_{2,i} f_{xxxx} - \beta_{3,i} f_{xx}^3) + O(t^3). \tag{4.15}$$

Additionally, the Taylor expansion of the second derivative of h_i has form

$$\frac{d^2}{dx^2} h_i(0) = f_{xx} + t(\beta_{4,i} f_{xxxx} - \beta_{5,i} f_{xx}^3) + O(t^2). \tag{4.16}$$

The coefficients in (4.15) *and* (4.16) *obey the following recursive relation:*

$$\begin{aligned}
&\beta_{1,0} = \beta_{2,0} = \beta_{3,0} = \beta_{4,0} = \beta_{5,0} = 0\\
\beta_{1,m} =& \Bigg[1 + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{1,i}\Bigg]\\
\beta_{2,m} =& \Bigg[\frac{1}{2} + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{2,i} + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{4,i}\Bigg]\\
\beta_{3,m} =& \Bigg[\frac{2}{3} + \frac{1}{6}\Bigg(\sum_{i=0}^{m-1} \gamma_{m,i}\beta_{1,i}\Bigg)^3 - \frac{1}{4}\Bigg(\sum_{i=0}^{m-1} \gamma_{m,i}\beta_{1,i}\Bigg)\Bigg(\sum_{i=0}^{m-1} \gamma_{m,i}\beta_{1,i}^2\Bigg)\\
&+ \frac{1}{12}\sum_{i=0}^{m-1} \gamma_{m,i}\beta_{1,i}^3 + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{3,i} + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{5,i}\Bigg]\\
\beta_{4,m} =& \Bigg[1 + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{4,i}\Bigg]\\
\beta_{5,m} =& \Bigg[2 + \sum_{i=0}^{m-1} \gamma_{m,i}\beta_{5,i}\Bigg]
\end{aligned} \tag{4.17}$$

Furthermore, the following conditions are necessary and sufficient for second order accuracy of algorithm 10:

(4.18)
$$\begin{aligned}\frac{\beta_{2,M}}{\beta_{1,M}^2} &= \frac{1}{2}\\ \frac{\beta_{3,M}}{\beta_{1,M}^2} &= 1\end{aligned}$$

Proof. We will prove (4.15) and (4.16) by induction. For h_1, these equations are the two dimensional version of equations (4.5) and (4.7) worked out in the previous section.

For the induction step, assume (4.15) and (4.16) up to $m-1$. We want to solve for y such that $\left[G_t * \sum_{i=0}^{m-1}\gamma_{m,i}\bar{u}_m\right](0,y) = \frac{1}{2}$ Using (C.11) for two dimensions and the linearity of the convolution we arrive at:

$$\begin{aligned}
&\left[G_t * \sum_{i=0}^{m-1}\gamma_{m,i}\bar{u}_i\right](0,y)\\
=&\frac{1}{2} - \frac{y}{2\sqrt{\pi t}} + \frac{y^3}{24\sqrt{\pi}t^{3/2}}\\
&+\frac{1}{2\sqrt{\pi t}}\sum_{i=0}^{m-1}\gamma_{m,i}\Bigg[h_i + t\frac{d^2}{dx^2}h_i + \frac{t^2}{2}\frac{d^4}{dx^4}h_i - \frac{y^2}{4t}h_i - \frac{y^2}{4}\frac{d^2}{dx^2}h_i + \frac{y}{4t}h_i^2\\
&+\frac{y}{2}h_i\frac{d^2}{dx^2}h_i + \frac{3ty}{4}(\frac{d^2}{dx^2}h_i)^2 - \frac{h_i^3}{12t} - \frac{h_i^2}{4}\frac{d^2}{dx^2}h_i - \frac{3t}{4}h_i(\frac{d^2}{dx^2}h_i)^2 - \frac{5t^2}{4}(\frac{d^2}{dx^2}h_i)^3\Bigg]\\
&+\text{h.o.t.}\\
=&\frac{1}{2} - \frac{y}{2\sqrt{\pi t}} + \frac{y^3}{24\sqrt{\pi}t^{3/2}}\\
&+\frac{1}{2\sqrt{\pi t}}\sum_{i=0}^{m-1}\gamma_{m,i}\Bigg[t\beta_{1,i}f_{xx} + t^2\beta_{2,i}f_{xxxx} - t^2\beta_{3,i}f_{xx}^3 + tf_{xx} + t^2\beta_{4,i}f_{xxxx} - t^2\beta_{5,i}f_{xx}^3\\
&+\frac{t^2}{2}f_{xxxx} - \frac{y^2}{4}\beta_{1,i}f_{xx} - \frac{y^2}{4}f_{xx} + \frac{yt}{4}\beta_{1,i}^2f_{xx}^2 + \frac{yt}{2}\beta_{1,i}f_{xx}^2\\
&+\frac{3ty}{4}f_{xx}^2 - \frac{\beta_{1,i}^3t^2}{12}f_{xx}^3 - \frac{\beta_{1,i}^2t^2}{4}f_{xx}^3 - \frac{3\beta_{1,i}t^2}{4}f_{xx}^3 - \frac{5}{4}t^2f_{xx}^3\Bigg] + \text{h.o.t.}
\end{aligned}$$

Setting the previous equation equal to a half and solving for y we have

$$
\begin{aligned}
y =& t f_{xx}\left[1+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right]+t^2 f_{xxxx}\left[\frac{1}{2}+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{2,i}+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{4,i}\right]\\
&-t^2 f_{xx}^3\left[\frac{2}{3}+\frac{1}{6}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)^3-\frac{1}{4}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}^2\right)\right.\\
&\left.+\frac{1}{12}\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}^3+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{3,i}+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{5,i}\right]+\text{h.o.t.}
\end{aligned}
\tag{4.19}
$$

Similarly, using (C.17) for two dimensions and the linearity of the convolution we derive:

(4.20)

$$
\begin{aligned}
&\frac{d^2}{dx^2}\left[G_t*\sum_{i=0}^{m-1}\gamma_{m,i}\bar{u}_i\right](x,y(x))|_{x=0}\\
&=-\frac{y_{xx}}{2\sqrt{\pi t}}+\frac{y^2 y_{xx}}{8\sqrt{\pi}t^{3/2}}\\
&\quad+\sum_{i=0}^{m-1}\gamma_{m,i}\left[\frac{1}{2\sqrt{\pi t}}\frac{d^2}{dx^2}h_i+\frac{\sqrt{t}}{2\sqrt{\pi}}\frac{d^4}{dx^4}h_i-\frac{y^2}{8\sqrt{\pi}t^{3/2}}(\frac{d^2}{dx^2}h_i)+\frac{y}{4\sqrt{\pi}t^{3/2}}h_i\left(\frac{d^2}{dx^2}h_i\right)\right.\\
&\quad+\frac{3y}{4\sqrt{\pi t}}\left(\frac{d^2}{dx^2}h_i\right)^2-\frac{h_i^2}{8\sqrt{\pi}t^{3/2}}\frac{d^2}{dx^2}h_i-\frac{3h_i}{4\sqrt{\pi t}}\left(\frac{d^2}{dx^2}h_i\right)^2\\
&\quad-\frac{15\sqrt{t}}{8\sqrt{\pi}}\left(\frac{d^2}{dx^2}h_{xx}\right)^3+\frac{y_{xx}h_i^2}{8\sqrt{\pi}t^{3/2}}+\frac{y_{xx}}{4\sqrt{\pi t}}h_i\left(\frac{d^2}{dx^2}h_i\right)+\frac{3\sqrt{t}}{8\sqrt{\pi}}t y_{xx}\left(\frac{d^2}{dx^2}h_i\right)^2\\
&\quad\left.-\frac{y_{xx}yh_i}{4\sqrt{\pi t}}-\frac{y_{xx}y}{4\sqrt{\pi t}}\frac{d^2}{dx^2}h_i\right]+\text{h.o.t.}
\end{aligned}
$$

Now substitute in (4.19), (4.15), (4.16) for y, h_i and $\frac{d^2}{dx^2}h_i$ respectively in (4.20):

$$\begin{aligned}
&\frac{d^2}{dx^2}\left[G_t * \sum_{i=0}^{m-1}\gamma_{m,i}\bar{u}_i\right](x,y(x))|_{x=0} \\
&= -\frac{y_{xx}}{2\sqrt{\pi t}} + \frac{1}{8\sqrt{\pi t}}\left(1+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)^2 y_{xx}f_{xx}^2 + \frac{1}{2\sqrt{\pi t}}f_{xx} + \frac{\sqrt{t}}{2\sqrt{\pi}}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{4,i}\right)f_{xx}^3 \\
&+ \frac{\sqrt{t}}{2\sqrt{\pi}}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{5,i}\right)f_{xxxx} + \frac{\sqrt{t}}{2\sqrt{\pi}}f_{xxxx} - \frac{\sqrt{t}}{8\sqrt{\pi}}\left(1+\sum\gamma_{m,i}\beta_{1,i}\right)^2 f_{xx}^3 \\
&+ \frac{\sqrt{t}}{4\sqrt{\pi}}\left(1+\sum_{i}^{m-1}\gamma_{m,i}\beta_{1,i}\right)\left(\sum\gamma_{m,i}\beta_{1,i}\right)f_{xx}^3 + \frac{3\sqrt{t}}{4\sqrt{\pi}}\left(1+\sum\gamma_{m,i}\beta_{1,i}\right)f_{xx}^3 \\
&- \frac{\sqrt{t}}{8\sqrt{\pi}}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)^2 f_{xx}^3 - \frac{3\sqrt{t}}{4\sqrt{\pi}}\left(\sum_{i}^{m-1}\gamma_{m,i}\beta_{1,i}\right)f_{xx}^3 - \frac{15\sqrt{t}}{8\sqrt{\pi}}f_{xx}^3 \\
&+ \frac{\sqrt{t}}{8\sqrt{\pi}}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)^2 y_{xx}f_{xx}^2 + \frac{\sqrt{t}}{4\sqrt{\pi}}\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)y_{xx}f_{xx}^2 + \frac{3\sqrt{t}}{8\sqrt{\pi}}y_{xx}f_{xx}^2 \\
&- \frac{\sqrt{t}}{4\sqrt{\pi}}\left(1+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)\left(\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)y_{xx}f_{xx}^2 - \frac{\sqrt{t}}{4\sqrt{\pi}}\left(1+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{1,i}\right)y_{xx}f_{xx}^2 \\
&+ \text{h.o.t.}
\end{aligned} \tag{4.21}$$

Setting (4.21) to zero and solving for y_{xx} we find

$$y_{xx} = f_{xx} + t\left[\left(1+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{4,i}\right)f_{xxxx} - \left(2+\sum_{i=0}^{m-1}\gamma_{m,i}\beta_{5,i}\right)f_{xx}^3\right] + O(t^2). \tag{4.22}$$

Equations (4.19) and (4.22) give the recursive relations (4.17).

The consistency equations (4.18) follow by the change of variable $\tau = t\beta_{1,M}$ for $h_M(0)$ and matching the Taylor expansion for motion by mean curvature (4.14). $\square$

4.5.2 Unconditional Stability

Next, we give conditions on the γ's that preserve unconditional stability in any dimension. Specifically, for energy

$$E_t(\Sigma) = \int_{\mathbb{R}^n}(1-\mathbf{1}_\Sigma)G_t * \mathbf{1}_\Sigma d\mathbf{x}. \tag{4.23}$$

our algorithm has the property $E_t(u_{n+1}) \le E_t(u_n)$. In chapter II, we proved conditions for unconditional stability of the following class of linear M-stage algorithms:

$$u_{n+1} = U_M = \arg\min_u E(u) + \sum_{i=0}^{M-1}\frac{\gamma_{M,i}}{2k}||u-U_i||^2 \tag{4.24}$$

where the intermediate stages U_m, for $m \geq 1$, are given by

$$U_m = \arg\min_u E(u) + \sum_{i=0}^{m-1} \frac{\gamma_{m,i}}{2k} ||u - U_i||^2. \tag{4.25}$$

for some energy E, fixed time step k and the stipulation $U_0 = u_n$. We state the stability conditions from that chapter below, and show that algorithm 10 falls into the desired class:

Theorem IV.2. *(From chapter II) Define the following auxiliary quantities in terms of the coefficients $\gamma_{m,i}$ of scheme* (4.24) *and* (4.25)*:*

$$\tilde{\gamma}_{m,i} = \gamma_{m,i} - \sum_{j=m+1}^{M} \tilde{\gamma}_{j,i} \frac{\tilde{S}_{j,m}}{\tilde{S}_{j,j}} \tag{4.26}$$

$$\tilde{S}_{j,m} = \sum_{i=0}^{m-1} \tilde{\gamma}_{j,i} \tag{4.27}$$

If $\tilde{S}_{m,m} > 0$ for $m = 1, \ldots, M$, then scheme (4.28) *ensures that for every $n = 0, 1, 2, \ldots$ we have $E(u_{n+1}) \leq E(u_n)$.*

It is shown in [15] that each step of the original threshold dynamics, algorithm 6, can be written as

$$\Sigma_{n+1} = \arg\min_\Sigma \int (1 - \mathbf{1}_\Sigma) G_t * \mathbf{1}_\Sigma + \int (\mathbf{1}_\Sigma - \mathbf{1}_{\Sigma_n}) G_t * (\mathbf{1}_\Sigma - \mathbf{1}_{\Sigma_n})\, d\mathbf{x}.$$

Similarly observe that (4.13) can be written as

$$\bar{\Sigma}_m = \arg\min_\Sigma \int (1 - \mathbf{1}_\Sigma) G_t * \mathbf{1}_\Sigma + \sum_{i=0}^{m-1} \gamma_{m,i} \int (\mathbf{1}_\Sigma - \mathbf{1}_{\bar{\Sigma}_i}) G_t * (\mathbf{1}_\Sigma - \mathbf{1}_{\bar{\Sigma}_i})\, d\mathbf{x}. \tag{4.28}$$

Moreover, as noted in [15], since $\widehat{G_t} > 0$,

$$\int u G_t * u \geq 0 \, d\mathbf{x}$$

with equality holding if and only if $u = 0$. Thus, algorithm 10 is of type (4.24) and (4.25) with energy (4.23) and inner product $\langle u, v \rangle = \int u G_t * v \, d\mathbf{x}$, so that conditions for unconditional stability identified in chapter II apply.

In the next section, we give examples of γ's that satisfy the consistency equations (claim IV.1) as well as the hypothesis of theorem IV.2 concurrently.

4.5.3 A Second Order Unconditionally Stable Example

In this section we present a set of γ's such that the second order consistency equations (4.18) and hypothesis of unconditional stability theorem IV.2 are satisfied simultaneously. We found the γ's numerically and then sought a nearby algebraic solution to the consistency equations that still satisfied the conditions in theorem IV.2. It can be shown that there is no unconditionally energy stable second order two- or three-stage method of type (4.13) for threshold dynamics. However, a four-step method exists with the following γ's:

$$\gamma = \begin{pmatrix} \gamma_{1,0} & 0 & 0 & 0 \\ \gamma_{2,0} & \gamma_{2,1} & 0 & 0 \\ \gamma_{3,0} & \gamma_{3,1} & \gamma_{3,2} & 0 \\ \gamma_{4,0} & \gamma_{4,1} & \gamma_{4,2} & \gamma_{4,3} \end{pmatrix} \approx \begin{pmatrix} 1 & 0 & 0 & 0 \\ -0.25 & 1.25 & 0 & 0 \\ 0.83 & -0.67 & 0.83 & 0 \\ -0.73 & 0.5 & -0.73 & 1.96 \end{pmatrix}. \tag{4.29}$$

The exact values are in the appendix (chapter D); they are all algebraic numbers but with long representations. This choice of γ's that endows algorithm 10 with unconditional stability and second order accuracy is not unique. In fact, one could find other choices of γ that preserve additional desirable properties. We summarize our results in the following theorem:

Theorem IV.3. *algorithm 10 with coefficients* (4.29) *is unconditionally energy stable and first order accurate in any dimension. Moreover, it is second order accurate in two dimensions.*

4.5.4 Consistency In Higher Dimensions

Unfortunately, there does not exist a choice of γ's such that algorithm 10 is second order in three dimensions. Using the Gaussian (or even a linear combination of Gaussian kernels) at every step in our multistage algorithm leads to consistency equations incompatible with second order mean curvature motion (4.2). Of course, one can choose a different kernel, denoted by K_t, at each stage. The consistency equations will be different. On the other hand, if the kernel has positive Fourier transform, $\widehat{K}_t \geq 0$, theorem IV.2 will hold for energy

$$E_t^*(\Sigma) = \int (1 - \mathbf{1}_\Sigma) K_t * \mathbf{1}_\Sigma d\mathbf{x}.$$

As an additional consideration, the energy induced by the kernel should have property

$$\lim_{t \to 0^+} E_t^*(\Sigma) \xrightarrow{\Gamma} c\mathrm{Per}(\Sigma) \tag{4.30}$$

for some constant c. (The Gaussian kernel has this property [31].) It is left to future work to find a scheme with kernel and γ's such that unconditional stability, positive Fourier transform, and property (4.30) hold, and that, furthermore, is second order in three (and higher) dimensions.

4.6 Numerical Tests

In this section, we present highly accurate convergence tests for the two new algorithms: algorithm 9 and algorithm 10. It is well known that naive implementations of threshold dynamics on uniform grids introduces large spatial discretization errors due to sampling characteristic functions. In fact, if the time step size is sufficiently small compared to the spatial discretization, interfaces can even get stuck. A very effective cure to this phenomenon is the adaptive implementation of Ruuth [38], which entails fast convolution on non-uniform grids. In practice, we recommend that the high order in time schemes introduced in this chapter be used in conjunction with such a spatial implementation.

Below, our focus is on numerically verifying the improvement in the convergence rate in time of the new threshold dynamics schemes on a few smooth interfaces. To ensure spatial errors are negligible in our experiments, for most of our experiments below, we simply represent the interfaces as graphs of functions. This is, of course, not meant as a practical implementation, but just as a very accurate and efficient way to minimize spatial errors – it allows us to reach very high spatial resolutions and accuracies – so that we can focus on time discretization errors. Section 4.6.1 explains the details of the implementation, and section 4.6.2 presents the results. The latter also contains an experiment with the practical implementation of the new schemes (albeit on uniform grids) to verify that no substantial qualitative difference is observed in the handling of topological changes vs. the original threshold dynamics algorithm.

4.6.1 Highly Accurate Threshold Dynamics For Graphs

This section explains how algorithm 9 and algorithm 10 have been implemented in the special case that the interface is given as the graph of a function for the purpose of numerical convergence tests given in section 4.6.2. We repeat that we are not advocating this implementation as a practical method – it is just for numerical convergence tests on smooth interfaces – but refer to Ruuth's adaptive spectral implementation [38] to be used in conjunction with our new algorithms.

Let $\Sigma = \{(\mathbf{x}, x_n) | x_n \leq f(\mathbf{x})\}$ for some function $f(\mathbf{x})$. Recall the definition of the Gaussian kernel $G_{\delta t}(\mathbf{x}) = \frac{1}{(4\pi\delta t)^{d/2}} e^{-\frac{|\mathbf{x}|^2}{4\delta t}}$. Then the convolution is

$$\begin{aligned} G_{\delta t} * \mathbb{1}_{\Sigma}(\mathbf{x}, x_n) &= \int_{\mathbb{R}^{d-1}} G_{\delta t}(\mathbf{x} - \bar{\mathbf{x}}) \int_{-\infty}^{f(\bar{\mathbf{x}})} \frac{1}{\sqrt{4\pi\delta t}} e^{-\frac{(x_n - \bar{x}_n)^2}{4\delta t}} d\bar{\mathbf{x}} d\bar{x}_n \\ &= \frac{1}{2} + \frac{1}{2} \int_{\mathbb{R}^{d-1}} G_{\delta t}(\mathbf{x} - \bar{\mathbf{x}}) \operatorname{erf}\left(\frac{f(\bar{\mathbf{x}}) - x_n}{2\sqrt{t}} \right) d\bar{\mathbf{x}} \end{aligned} \tag{4.31}$$

where $\operatorname{erf}(\cdot)$ is the error function. The latter integral is calculated numerically (e.g. Gaussian quadrature) inside a region $\Omega \subset \mathbb{R}^{n-1}$ such that $\int_{\Omega^c} G_{\delta t}(\mathbf{x} - \bar{\mathbf{x}}) d\bar{\mathbf{x}} < \epsilon$ for some preset tolerance ϵ.

With this tool in hand, we can implement the original threshold dynamics (algorithm 6) by tracking the interface at fixed points $\mathbf{x}$ throughout the evolution. We describe this in algorithm 11. The second order versions of threshold dynamics we proposed in this chapter, algorithm 9 and algorithm 10, are implemented similarly.

Algorithm 11 Highly Accurate Threshold Dynamics For Graphs

```
1: Fix total evolution time $T$, time step size $\delta t$, and points $\{\mathbf{x}^i\}_{i=1}^N \in D \subset \mathbb{R}^{d-1}$.
2: Set $\Sigma^0 = \{(\mathbf{x}, x_n) | x_n \leq f(\mathbf{x}, 0)\}$ and $K = T/\delta t$.
3: for $k \leftarrow 1$ to $K$ do
4:     For each $\mathbf{x}_i$ find $y^i$ such that
           $$\frac{1}{2} = G_{\delta t} * \mathbb{1}_{\Sigma^{k-1}}(\mathbf{x}^i, y^i)$$
       using e.g. the secant method, estimating the right hand side using (4.31).
5:     Set $f(\mathbf{x}^i, k\delta t) = y^i$ and $\Sigma^k = \{(\mathbf{x}, x_n) | x_n \leq f(\mathbf{x}, k\delta t)\}$.
```

4.6.2 Numerical Results

In this section, we will test algorithm 9 and algorithm 10 on a couple of two phase mean curvature motion evolution problems to demonstrate their accuracy.

In $\mathbb{R}^2$ we run our algorithms on the 'Grim Reaper Wave.' The interface is given by $f(x,t) = \operatorname{arcsinh}(\exp(-t)\cos(x))$. We run the evolution with initial data $f(x,0) = \operatorname{arcsinh}(\cos(x))$ to $T = 1$ on domain $x = [0, \pi]$ with homogeneous Neumann boundary conditions. We track 4000 points and report the L^2 error between final interface and $f(x,1)$. We include the errors for the original threshold dynamics in table 4.1 for comparison. The error for algorithm 9 is reported in table 4.2 and the error for algorithm 10, with γ's given in (4.29), is tabulated in table 4.3.

In three dimensions, we run mean curvature evolution tests with initial interface $f(x,y,0) = \cos(\pi y)\cos(\pi x) + \frac{1}{2}\cos(\pi y)$ to $T = 1/10$ on $(x,y) \in [-1,1] \times [-1,1]$ with periodic boundary conditions. We generate the 'true' solution by forward Euler

Number of time steps	8	16	32	64	128	256
L^2 error	5.4e-03	2.6e-03	1.3e-03	6.7e-04	3.3e-04	1.7e-04
Order	-	1.0	1.0	1.0	1.0	1.0

Table 4.1: Ordinary Threshold Dynamics, algorithm 6, on the 'Grim Reaper Wave.'

Number of time steps	8	16	32	64	128	256
L^2 error	1.9e-04	3.4e-05	8.4e-06	2.1e-06	5.1e-07	1.3e-07
Order	-	2.5	2.0	2.0	2.0	2.0

Table 4.2: Algorithm 9 on the 'Grim Reaper Wave.'

discretization of the PDE for mean curvature motion (4.1) using very small time steps and a very high spatial resolution. We tabulate the L^2 error between the 'true' interface and the output of algorithm 9 in table 4.4.

As an additional test in three dimensions, we use algorithm 9 to evolve a dumbbell by mean curvature motion using the practical implementation of algorithm 9, albeit on a uniform grid. The system goes through a topological change where the connected set breaks off into two components. In fig. 4.1, we show the original threshold dynamics algorithm 6 and algorithm 9 at time steps near the pinch off. There is a slight difference between the two algorithms at the temporal and spatial resolutions we have chosen. Presumably, this difference will shrink as we further refined our temporal and spatial resolution. Regardless, algorithm 9 behaves reasonably during the pinch off.

We expect algorithm 9 to revert to first order for any time interval containing a topological change. This is because the Taylor expansions in our consistency calculation are not valid at the moment of pinch off. To test this, we use algorithm 9 to evolve a vase-like shape in three dimensions that breaks into two components over time, see fig. 4.2. The surface is represented as a parametrized curve rotated around the z-axis. We use the highly accurate threshold dynamics implementation

Number of time steps	8	16	32	64	128	256
L^2 error	6.6e-05	1.6e-05	4.1e-06	1.0e-06	2.6e-07	6.3e-08
Order	-	2.0	2.0	2.0	2.0	2.0

Table 4.3: Algorithm 10 with γ's given in (4.29) on the 'Grim Reaper Wave.'

Number of time steps	4	8	16	32	64
L2	3.3e-03	6.5e-04	1.3e-04	2.7e-05	6.3e-06
Order	-	2.4	2.4	2.2	2.1

Table 4.4: Algorithm 9 on an interface in $\mathbb{R}^3$

for graphs from algorithm 11, treating the radius as a function of z, to ensure our spatial errors are negligible. The initial surface is the function $2 - \cos(z\frac{\pi}{5})$ rotated around the z-axis. We evolve the surface for $T = 3/4$ and use periodic boundary conditions at $z = -5$ and $z = 5$. We generate the 'true' solution by the original threshold dynamics (algorithm 6) version of algorithm 11, using small time steps and a high spatial resolution. We tabulate the L^2 error between the radius of the 'true' interface and the output of algorithm 9 in table 4.5. As expected, algorithm 9 reverts to first order.

Number of time steps	16	32	64	128
L2	3.9e-02	2.2e-02	1.2e-02	6.1e-03
Order	-	0.8	0.8	1.0

Table 4.5: Algorithm 9 on an shape that undergoes a topological change in $\mathbb{R}^3$.

4.7 Conclusion

We have presented second order threshold dynamic methods for simulating two phase mean curvature flow. Unlike the previous method, algorithm 7, our methods represent the phases via a binary function at every step. Additionally, we present an unconditionally stable method in two dimensions. We have demonstrated the

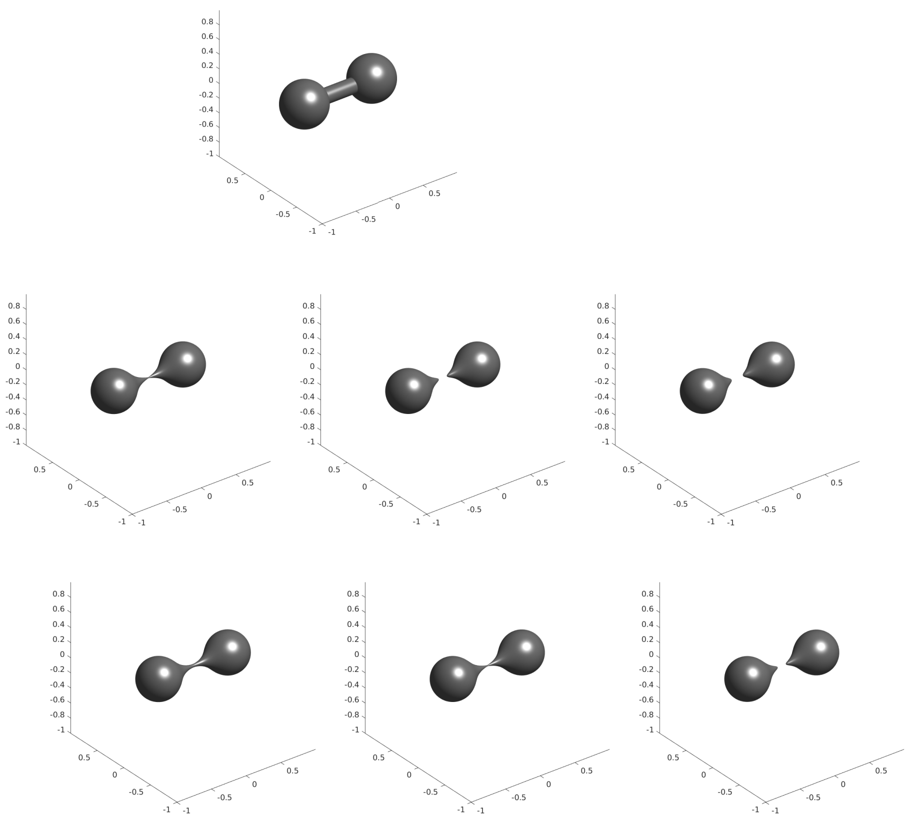

Figure 4.1: The original threshold dynamics algorithm 6 and algorithm 9 evolving a dumbbell by mean curvature motion. Top: the initial dumbbell. Center: algorithm 6 before and after the topological change. Bottom: algorithm 9 before and after the topological change at the same time values as the center row.

methods and their advertised accuracy on examples in two and three dimensions.

Finding a method that is second order in three and higher dimensions and unconditionally stable in the case of two phases is a topic of future investigation.

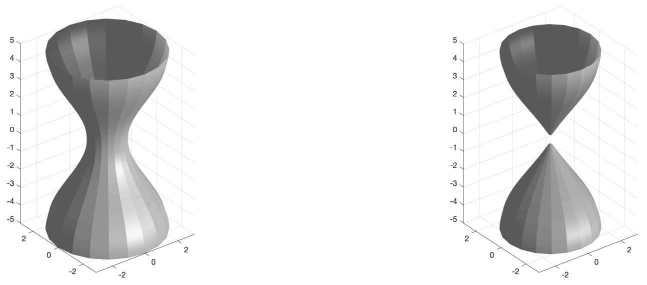

Figure 4.2: Initial (right) and final shape (left) for a test of how algorithm 9 behaves after a topological change.

CHAPTER V

On the Voronoi Implicit Interface Method

5.1 Introduction

The Voronoi implicit interface method (VIIM) [40] is a type of level set method [34] that is particularly suited to the treatment of multiphase flows. It has been demonstrated on problems that incorporate surface tension, such as dynamics of bubble clusters, and the motion of grain boundaries. Often, the hardest aspect of designing numerical schemes for such problems is ensuring that the correct angle conditions at triple junctions – in terms of the surface tensions of the interfaces meeting there – are satisfied. The purpose of this chapter is to investigate whether the VIIM in fact attains the correct angle conditions at junctions. To that end, and to study this essential difficulty in isolation, we focus on problems where surface tension is the sole driving force: multiphase motion by mean curvature. This evolution arises in many applications, including grain boundary motion in polycrystalline materials during annealing [32] and image segmentation algorithms in computer vision [33]. Once the correct behavior at junctions has been verified, additional driving forces, e.g. from bulk effects, fluid flow, etc., can be incorporated in the VIIM and similar algorithms with ease.

The first contribition of this chapter is to present convergence tests at very high resolutions for the VIIM. To our knowledge, this is the first time that such highly accurate convergence tests have been done for this method. One of our main results is that the VIIM does not, in general, converge to the correct solution when interfaces with unequal surface tensions meet at a triple junction. Our second contribution in this chapter is to present a modification of the Voronoi implicit interface method that fixes the non-convergence and ensures that the correct angle conditions are attained at all triple junctions.

The chapter is organized as follows:

- In section 5.2 we briefly review multiphase motion by mean curvature and recall its variational formulation.
- In section 5.3 we recall the Voronoi implicit interface method.
- In section 5.4 we present our implementation of the VIIM using parameterized curves. We then present high resolution convergence tests indicating that the VIIM does not converge to the correct solution when the surface tensions are unequal.
- In section 5.5 we present convergence tests using the parametrized curve representation of another recent algorithm, known as threshold dynamics, the convergence of which is supported by many recent results in the literature. This verifies the validity and accuracy of our parametrized curve implementation.
- In section 5.6 we present a modification to the VIIM, which may be called the dictionary mapping implicit interface method (DMIIM), that does converge in the unequal surface tension case. After discussing its relation to the VIIM, we subject this new variant to the same careful numerical convergence studies, again via an implementation on parametrized curves to reach very high resolutions, and thus demonstrate its accuracy and convergence. We then present further

examples of the new algorithm on implicitly defined interfaces on a grid.

The code for sections 4 through 6 is publicly available, and can be found at `https://github.com/AZaitzeff/DMIIM`.

5.2 Multiphase Motion by Mean Curvature

In this chapter, we will be concerned exclusively with gradient descent dynamics for energies of the form

$$E(\Sigma_1, \ldots, \Sigma_n) = \sum_{i \neq j} \sigma_{ij} \mathrm{Area}(\Gamma_{ij}). \tag{5.1}$$

where $\Gamma_{ij} = (\partial\Sigma_i) \cap (\partial\Sigma_j)$ are the interfaces between the *phases* $\Sigma_1, \ldots, \Sigma_n$ that partition a domain $D \subset \mathbb{R}^d$, $d \geq 2$:

$$\Sigma_i \cap \Sigma_j = (\partial\Sigma_i) \cap (\partial\Sigma_j) \text{ for any } i \neq j, \text{ and } \bigcup_{i=1}^{N} \Sigma_i = D.$$

Note that if two grains are not neighbors, the intersection is empty and the corresponding term in the sum of (5.1) drops out. The positive constants $\sigma_{ij} = \sigma_{ji}$ are known as *surface tensions* (or surface energy density). They need to satisfy the triangle inequality

$$\sigma_{ij} + \sigma_{ik} \geq \sigma_{jk} \text{ for any distinct } i, j \text{ and } k$$

for well-posedness of the model (5.1). Multiphase mean curvature motion arises as L^2 gradient descent on energies of this form, and can be described as follows:

1. At any point $p \in \Gamma_{ij}$ away from triple junctions where the interface is smooth, the normal speed, denoted $v_\perp(p)$, is given by

   $$v_\perp(p) = \mu_{ij}\sigma_{ij}\kappa_{ij}(p). \tag{5.2}$$

   Here, $\kappa_{ij}(p) = \kappa_{ji}(p)$ is the *mean curvature* at point p. The positive constants μ_{ij} are called *mobilities*. Unless otherwise stated, we will take $\mu_{ij} = 1$.

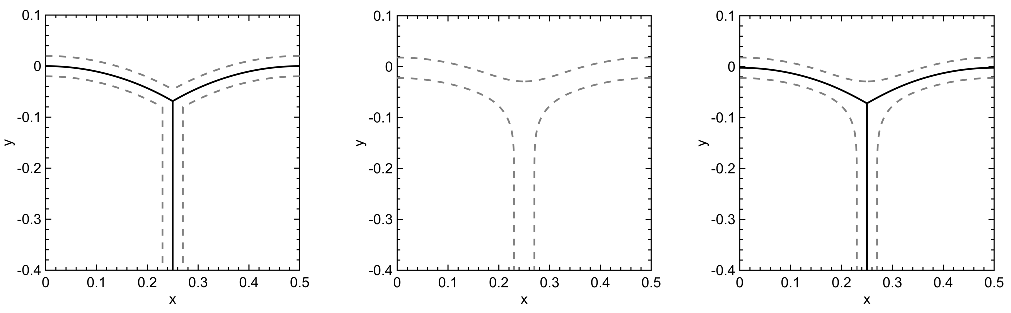

Figure 5.1: How the VIIM works: On the left we have the ϵ level sets (dotted lines), in the center we have the sets after being evolved by (5.5), the figure on the right shows the new interfaces after Voronoi reconstruction (solid line).

2. Let a *triple junction* be formed by the meeting of three phases Σ_1, Σ_2, and Σ_3. They are points in two dimensions and occur along curves in three dimensions. Let θ_i be the angle between Γ_{ij} and Γ_{ik} at the junction. Then:

$$\frac{\sin\theta_1}{\sigma_{23}} = \frac{\sin\theta_2}{\sigma_{13}} = \frac{\sin\theta_3}{\sigma_{12}} \tag{5.3}$$

has to hold. This is known as the Herring angle condition [23].

Until now, the VIIM has been described only for the very special class of surface tensions known as *additive* surface tensions: We call σ_{ij} additive if they can be written as

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j)$$

for some constants $\sigma_1, \sigma_2, \ldots, \sigma_n \geq 0$. Additive surface tensions thus have n degrees of freedom, constituting therefore a very small subclass of physically relevant surface tensions, which require $\binom{n}{2}$ degrees of freedom to fully specify.

5.3 The Voronoi Implicit Interface Method

We recall the level set formulation [34] of motion by mean curvature that gives the current configuration, $\partial\Sigma(t)$, of the boundary with initial configuration $\partial\Sigma(0)$ of a given initial set $\Sigma(0) \subset D$ in the two phase setting. Let $\phi : D \times \mathbb{R}^+ \to \mathbb{R}$ be a level set function for $\Sigma(0)$ so that $\phi(x,0) > 0$ for $x \in \Sigma^\circ(0)$ and $\phi(x,0) < 0$ for $x \in \Sigma^c(0)$. Often, a particularly convenient choice of level set function to represent the boundary $\partial\Sigma$ of a set Σ is the *signed distance function*,

$$d_\Sigma(\mathbf{x}) = \begin{cases} \min\limits_{\mathbf{z}\in\partial\Sigma} ||\mathbf{x}-\mathbf{z}||_2 & \mathbf{x} \in \Sigma \\ -\min\limits_{\mathbf{z}\in\partial\Sigma} ||\mathbf{x}-\mathbf{z}||_2 & \mathbf{x} \notin \Sigma. \end{cases} \tag{5.4}$$

The process of "reinitializing" a level set function ϕ by replacing it with the signed distance function to its 0-level set, $d_{\{x\,:\,\phi(x)\geq 0\}}(x)$, is known as "redistancing" in the level set literature, and is a common operation, typically applied only sporadically to prevent ϕ from becoming too steep or too flat (see, for example, [41]). In any case, if $\phi(x,t)$ solves the well-known PDE

$$\phi_t - \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|} \right) |\nabla\phi| = 0. \tag{5.5}$$

and we set $\Sigma(t) = \{\mathbf{x} \in D \,:\, \phi(\mathbf{x},t) \geq 0\}$, then $\partial\Sigma(t)$ evolves by motion by mean curvature.

There have been multiple algorithms proposed in the literature to extend the level set formulation of mean curvature motion to the multiphase setting. We note, in particular, the level set method of [46], the variational level set method of [48], and the distance function based diffusion generated motion of [13], [16], and [12]. The latter three contributions alternate diffusion by the linear heat equation applied individually to the level set functions of the phases (so that they are decoupled at this stage), a simple pointwise redistribution step that couples the phases, and redistancing on the new level set functions, to generate the desired multiphase evolution; in this sense, they are a cross between the convolution generated motion of [30] and the level set method.

In [39], Saye and Sethian introduced a variant of the algorithm in [13]. This new version also alternates redistancing and decoupled evolution of the level sets of individual phases with a pointwise redistribution step that imposes the requisite coupling. The key differences are: **1.** the decoupled motion of the level sets is by the nonlinear PDE (5.5) vs. the linear heat equation, and **2.** the redistribution step takes place after (vs. before) redistancing of the individually evolved level set functions. An additional novelty, mostly for convenience, is an innovative step to enable evolution of all the level set functions concurrently, by applying (5.5) to the unsigned distance function of the union of the $\epsilon > 0$ super-level sets of the phases,

$$\varphi_\epsilon(\mathbf{x}) = \min_{\mathbf{z} \in \cup_i \partial\Sigma_i} ||\mathbf{x} - \mathbf{z}||_2 - \epsilon.$$

Although only the equal surface tension case ($\sigma_{ij} = 1$ for all $i \neq j$) of multiphase motion by mean curvature was considered in the original paper [39], in a subsequent contribution [40], Saye and Sethian proposed an extension of their method to certain (additive) unequal surface tension networks. This is a very small subset of all surface tensions allowed by model (5.1). Moreover, the extension in [40] takes all mobilities to be equal, again a vast restriction over (5.2). One of the original motivations for the present study was to investigate if the algorithm could be extended to the full generality of model (5.1) & (5.2).

We will introduce some notation to represent various steps of the VIIM as described in [39], including the extension to additive surface tensions in [40]. To that end, first define the function $\mathcal{S}_{\Delta t}$ by

$$(d_{\Sigma_1(\sigma_1\Delta t,\epsilon)}, d_{\Sigma_2(\sigma_2\Delta t,\epsilon)}, \ldots, d_{\Sigma_n(\sigma_n\Delta t,\epsilon)}) = \mathcal{S}_{\Delta t}(\Sigma_1, \Sigma_2, \ldots, \Sigma_n)$$

where $\Sigma_i(t,\epsilon) = \{\mathbf{x} \,:\, \phi_i(\mathbf{x},t) \geq \epsilon\}$ denotes the ϵ-super level set of the solution $\phi_i(\mathbf{x},t)$ of mean curvature flow equation (5.5) at time t, starting from the initial condition $\phi_i(\mathbf{x},0) = d_{\Sigma_i}(\mathbf{x})$. Here σ_i is the surface tension associated with phase Σ_i whereas the surface tension corresponding to the interface Γ_{ij} is $\frac{1}{2}(\sigma_i + \sigma_j)$.

Next, the Voronoi reconstruction step of the VIIM (that reallocates points among the phases) will be represented by the function R_v, which maps an n-tuple of functions $(\phi_1, \dots, \phi_n)$ to an n-tuple of sets $\Omega_1, \Omega_2, \dots, \Omega_n$:

$$(\Omega_1, \Omega_2, \dots, \Omega_n) = R_v(\phi_1, \phi_2, \dots, \phi_n)$$

where

$$\Omega_i = \Big\{ \mathbf{x} \,:\, \phi_i(\mathbf{x}) = \max_j \phi_j(\mathbf{x}) \Big\}.$$

With this notation, the evolution of a multiphase system by the VIIM at the N-th time step with time step size Δt, $T = N\Delta t$, is given by

$$(R_v \circ \mathcal{S}_{\Delta t})^N (\Sigma_1, \Sigma_2, \dots, \Sigma_n). \tag{5.6}$$

In [39, 40], no extension of the algorithm is given to the far more general case of $\binom{n}{2}$ surface tensions; moreover, the mobilities of all the interfaces are assumed to be 1, with again no indication given for greater generality. The method is summarized by algorithm 12 and illustrated in fig. 5.1.

Algorithm 12 The Voronoi Implicit Interface Method

```
1: Given $\Sigma_1^0, \Sigma_2^0, \dots, \Sigma_n^0$.
2: Let $N = T/\Delta t$.
3: for $k \leftarrow 1$ to $N$ do
4:     Evolve each $\phi_i(\cdot, 0) = d_{\Sigma_i^{k-1}}$ by time $\sigma_i \Delta t$ by (5.5) to obtain $\phi_i(\cdot, \sigma_i \Delta t)$.
5:     Build the signed distance functions $d_{\Sigma_i^{k-1}(\sigma_i\Delta t, \epsilon)} = d_{\{\mathbf{x} : \phi_i(\mathbf{x}, \sigma_i \Delta t) \geq \epsilon\}}$.
6:     Construct the new phases $\Sigma_i^k = \{\mathbf{x} : d_{\Sigma_i^{k-1}(\sigma_i\Delta t,\epsilon)}(\mathbf{x}) = \max_j d_{\Sigma_j^{k-1}(\sigma_j\Delta t,\epsilon)}(\mathbf{x})\}$
```

Saye and Sethian state in [40] that convergence to the desired motion is obtained by taking the double limit, $\lim\limits_{\epsilon\to 0} \lim\limits_{\Delta t \to 0}$ in (5.6), with $N = \frac{T}{\Delta t}$. However, they also discuss two other limiting procedures: the "coupled" limit, $\lim\limits_{\epsilon = c\sqrt{\Delta t} \to 0^+}$, and the interchanged double limit, $\lim_{\Delta t \to 0} \lim_{\epsilon \to 0}$. In the case of equal surface tensions, the authors present numerical convergence studies for each of these three limits. In the case of unequal surface tension, they only cite qualitative evidence and only for the coupled limit. In the next section, we present highly accurate, exhaustive numerical convergence studies of the VIIM, in the equal and unequal surface tension cases, for *all* of these limits.

5.4 Testing the VIIM using Parameterized Curves

To carefully assess the convergence of the VIIM, we will implement it in $\mathbb{R}^2$ using *parametrized curves* to represent the interfaces $\Gamma_{ij} = (\partial\Sigma_i) \cap (\partial\Sigma_j)$, and test it on exact solutions away from topological changes. This will allow us to reach resolutions not easily attainable with the practical implementation of the algorithm for arbitrary initial data via implicit (level set) representation of the interfaces on a uniform grid. We stress that we are not advocating parametric representation in the context of the VIIM as a general numerical method, as it defeats the original purpose – painless handling of topological changes – of a level set based algorithm; we use it only to carry out reliable convergence studies.

Below, Section 4.1 recalls the well-known "Grim Reaper" exact solution of three phase motion by mean curvature, with equal and unequal surface tensions. Section 4.2 discusses in detail how each step of the VIIM is implemented via parametrized curves (front tracking). Finally, Section 4.3 presents the results of our numerical convergence study.

5.4.1 "Grim Reaper" Solution

"Grim Reapers" are a family of exact solutions to three-phase motion by mean curvature that include unequal surface tension and mobility cases. Our domain D

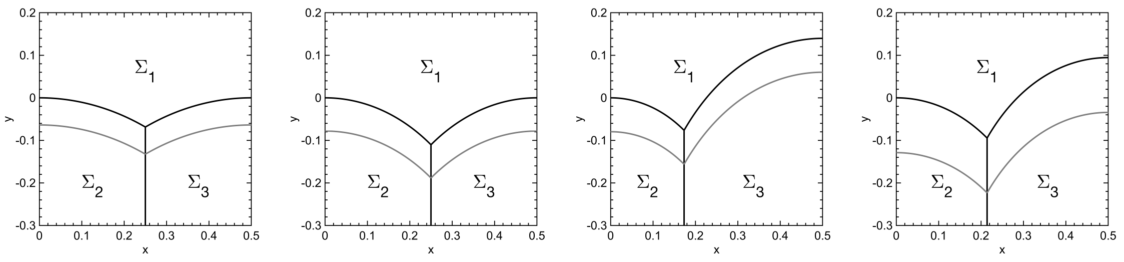

Figure 5.2: "Grim Reaper" exact solutions for angles (left to right) $(120°, 120°, 120°)$, $(90°, 135°, 135°)$, $(75°, 135°, 150°)$ with $\mu_i = 1$ and $(75°, 135°, 150°)$ with $\mu_i = \frac{1}{\sigma_i}$. The black line is $t = 0$ and the gray line is $t = \frac{18}{512}$.

will be $\left[0, \frac{1}{2}\right] \times [-\frac{1}{2}, 1] \subset \mathbb{R}^2$, and we will impose Neumann boundary conditions: interfaces intersect ∂D at right angles. In all our examples, the interface $\partial\Sigma_1(t)$ will be given as the graph of a function $f = f(x,t)$, at least on $t \in [0, \frac{18}{512}]$ where we choose 18/512 to allow the curves to travel an appreciable distance in our domain. The three phases $\Sigma_1(t)$, $\Sigma_2(t)$, and $\Sigma_3(t)$ will be described in terms of $f(x,t)$ as follows:

$$\begin{aligned}
\Sigma_1(t) =& \Big\{(x,y) : y \geq f(x,t)\Big\}\\
\Sigma_2(t) =& \Big\{(x,y) : x \leq \beta \text{ and } y \leq f(x,t)\Big\}\\
\Sigma_3(t) =& \Big\{(x,y) : x \geq \beta \text{ and } y \leq f(x,t)\Big\}
\end{aligned}$$

Below we list a number of specific grim reaper solutions that we use for convergence studies throughout the chapter:

- $(\theta_1, \theta_2, \theta_3) = (120°, 120°, 120°)$

  $(\sigma_{12}, \sigma_{13}, \sigma_{23}) = (1, 1, 1)$

  $\beta = \frac{1}{4}$

$$f(x,t) = \begin{cases} \frac{3}{2\pi}\log(\cos[\frac{2\pi}{3}x]) - \frac{2\pi}{3}t & \text{if } 0 \leq x \leq \frac{1}{4} \\ \frac{3}{2\pi}\log(\cos[\frac{2\pi}{3}(\frac{1}{2} - x)]) - \frac{2\pi}{3}t & \text{if } \frac{1}{4} < x \leq \frac{1}{2} \end{cases}$$

- $(\theta_1, \theta_2, \theta_3) = (90°, 135°, 135°)$

  $(\sigma_{12}, \sigma_{13}, \sigma_{23}) = (1, 1, \sqrt{2})$

  $\beta = \frac{1}{4}$

$$f(x,t) = \begin{cases} \frac{1}{\pi}\log(\cos[\pi x]) - \pi t & \text{if } 0 \le x \le \frac{1}{4} \\ \frac{1}{\pi}\log(\cos[\pi(\frac{1}{2} - x)]) - \pi t & \text{if } \frac{1}{4} < x \le \frac{1}{2} \end{cases}$$

- $(\theta_1, \theta_2, \theta_3) = (75°, 135°, 150°)$

  $(\sigma_{12}, \sigma_{13}, \sigma_{23}) = \left(\frac{\sqrt{2}}{4} + \frac{\sqrt{6}}{4}, \frac{\sqrt{2}}{2}, \frac{1}{2}\right)$

  $\beta = \frac{3}{46}(4\sqrt{2} - 3)$

$$f(x,t) = \begin{cases} \frac{1}{\frac{\pi}{6}(3+4\sqrt{2})}\log(\cos[\frac{\pi}{6}(3 + 4\sqrt{2})x]) - \frac{\pi}{12}(3+4\sqrt{2})t \\ \qquad\qquad \text{if } 0 \le x \le \frac{3}{46}(4\sqrt{2} - 3) \\ \frac{1}{\frac{\pi}{12}(3\sqrt{2}+8)}\log(\frac{2}{2^{\frac{\sqrt{2}}{4}}}\cos[\frac{\pi}{12}(3\sqrt{2} + 8)(\frac{1}{2} - x)]) - \frac{\pi}{12}(3 + 4\sqrt{2})t \\ \qquad\qquad \text{if } \frac{3}{46}(4\sqrt{2} - 3) < x \le \frac{1}{2} \end{cases}$$

- $(\theta_1, \theta_2, \theta_3) = (75°, 135°, 150°)$

  $(\sigma_{12}, \sigma_{13}, \sigma_{23}) = \left(\frac{\sqrt{2}}{4} + \frac{\sqrt{6}}{4}, \frac{\sqrt{2}}{2}, \frac{1}{2}\right)$

  $\mu_{ij} = \frac{1}{\sigma_{ij}}$

  $\beta = \frac{3}{14}$

$$f(x,t) = \begin{cases} \frac{6}{7\pi}\log(\cos[\frac{7\pi}{6}x]) - \frac{7\pi}{6}t & \text{if } 0 \le x \le \frac{3}{14} \\ \frac{6}{7\pi}\log(\sqrt{2}\cos[\frac{7\pi}{6}(\frac{1}{2} - x)]) - \frac{7\pi}{6}t & \text{if } \frac{3}{14} < x \le \frac{1}{2} \end{cases}$$

fig. 5.2 shows the exact solutions at time $t = 0$ and $t = \frac{18}{512}$. In the case where $\theta_2 = \theta_3$, $\Sigma_2(t)$ is the reflection of $\Sigma_3(t)$ around $x = \frac{1}{4}$, so we only need to track $\Sigma_1(t)$ and $\Sigma_2(t)$ on $\left[0, \frac{1}{4}\right] \times \left[-\frac{1}{2}, 1\right]$.

5.4.2 The VIIM via Parameterized Curves

We begin the section by describing two essential numerical procedures. The first finds the distance between a parameterized curve and a given point, and the second evolves a parameterized curve by curvature motion. We then discuss our implementation of the VIIM on the "Grim Reaper" test case using parameterized curves.

Distance Estimation for a Parameterized Curve

Finding the distance between a given point and a parameterized curve is important for constructing the ϵ level sets and for the Voronoi reconstruction step. To measure this distance with high accuracy we interpolate the parameterized curve with not-a-knot cubic splines [9]. Let $\{(x_i, y_i)\}_{i=1}^n$ be points on a curve that are given as the graph of a function of x and denote its cubic spline approximation as $\bar{f}(x)$. The cubic spline is a piecewise third order polynomial that is twice differentiable over $[x_1, x_n]$ with coordinates $y_i = \bar{f}(x_i)$. The signed distance between a given point $(\tilde{x}, \tilde{y})$ and the point which is closest to it on the aforementioned curve, is

$$\operatorname{sgn}(\tilde{y} - \bar{f}(\tilde{x})) \min_{x \in [x_1, x_N]} \sqrt{(x - \tilde{x})^2 + (\bar{f}(x) - \tilde{y})^2}.$$

We find the minimum using Newton's method (algorithm 13).

Algorithm 13 Netwon's method for finding distance between a parameterized curve and a point

```
1: Given a point $(\tilde{x}, \tilde{y})$, a cubic spline curve $y = \bar{f}(x)$, points $\{(x_i, y_i)\}_{i=1}^n$ with $y_i = \bar{f}(x_i)$ and
   tolerance $\delta$:
2: Let $I = \arg\min_i (x_i - \tilde{x})^2 + (y_i - \tilde{y})^2$
3: Set $x = x_I$ to be the starting point in Newton's method
4: while $|(x - \tilde{x}) + (\bar{f}(x) - \tilde{y})\bar{f}'(x)| \geq \delta$ do
5:     $x \leftarrow x - \frac{(x - \tilde{x}) + (\bar{f}(x) - \tilde{y})\bar{f}'(x)}{1 + (\bar{f}'(x))^2 + (\bar{f}(x) - \tilde{y})\bar{f}''(x)}$
6: Output $\operatorname{sgn}(\tilde{y} - \bar{f}(\tilde{x}))\sqrt{(x - \tilde{x})^2 + (\bar{f}(x) - \tilde{y})^2}$.
```

Newton's method uses the first two derivatives of the cubic spline. The interpolant is known to converge to the true curve in C^2 [2].

While the foregoing discussion applies to a curve that is a function of x, the above techniques can be used for any simple curve by applying an appropriate change of variables. In general, distance functions are not differentiable everywhere, however we only use the distance function at differentiable points in the “Grim Reaper” cases we consider.

Curvature Motion for a Parameterized Curve

Curvature motion for a parameterized curve, $\gamma(s,t) : [0,L] \times \mathbb{R}^+ \to \mathbb{R}^2$, is described by:

$$\gamma_t = \partial_s\left(\frac{\gamma_s}{|\gamma_s|}\right)\frac{1}{|\gamma_s|} \tag{5.7}$$

The differential equation (5.7) is implemented by a fully implicit Euler scheme where each iteration involves solving multiple tridiagonal linear systems. The scheme in time and space is

$$-\frac{\delta t}{h_i^{k+1}h_{i-\frac{1}{2}}^{k+1}}\gamma_{i-1}^{k+1} + \left[1 + \frac{\delta t}{h_i^{k+1}}\left(\frac{1}{h_{i-\frac{1}{2}}^{k+1}} + \frac{1}{h_{i+\frac{1}{2}}^{k+1}}\right)\right]\gamma_i^{k+1} - \frac{\delta t}{h_i^{k+1}h_{i+\frac{1}{2}}^{k+1}}\gamma_{i+1}^{k+1} = \gamma_i^k$$

where $h_i^k = \frac{1}{2}|\gamma_{i-1}^k - \gamma_{i+1}^k|$, $h_{i+\frac{1}{2}}^k = |\gamma_i^k - \gamma_{i+1}^k|$ and δt is the time step size for the finite difference scheme (different than $\sigma\Delta t$, the total time we evolve the curve between reconstructions). At each step we use the Newton iteration

$$-\frac{\delta t}{h_i^{k(l)}h_{i-\frac{1}{2}}^{k(l)}}\gamma_{i-1}^{k(l+1)} + \left[1 + \frac{\delta t}{h_i^{k(l)}}\left(\frac{1}{h_{i-\frac{1}{2}}^{k(l)}} + \frac{1}{h_{i+\frac{1}{2}}^{k(l)}}\right)\right]\gamma_i^{k(l+1)} - \frac{\delta t}{h_i^{k(l)}h_{i+\frac{1}{2}}^{k(l)}}\gamma_{i+1}^{k(l+1)} = \gamma_i^k$$

until $|\sum_{i=0}^N h_i^{k(l)} - \sum_{i=0}^N h_i^{k(l+1)}| < \delta$ for a small $\delta > 0$ and then set $\gamma^{k+1} = \gamma^{k(l)}$.

We detail how we handle all the boundary conditions in the $\theta_2 = \theta_3$ case in table 5.1. Then $\gamma(s, \sigma\Delta t)$ is given by γ^K for $K = \frac{\sigma\Delta t}{\delta t}$ with initial value $\gamma^0 = \gamma(s,0)$. Note that the choice of δt is independent from Δt. In our numerical studies we choose δt so small that the contribution to the overall error from the numerical solution $\gamma(s,\sigma\Delta t)$ is negligible.

Table 5.1: Boundary Conditions for PDE (5.7)

Case	x Boundary Condition	y Boundary Condition
$x(0) = 0$	$x(-s) = -x(s)$	$y(-s) = y(s)$
$x(L) = .25$	$x(L+s) = .5 - x(L-s)$	$y(L+s) = y(L-s)$
$y(L) = -.5$	$x(L+s) = x(L-s)$	$y(L+s) = -1 - y(L-s)$

The Implementation of the VIIM using Parameterized Curves

With these two tools in hand we can implement the VIIM using parameterized curves. At every iterate we track a series of (x, y) points that parameterize the interface of the sets. We will first consider the the case where $\theta_2 = \theta_3$ (see the first two images in fig. 5.2). Since Σ_3^k is the reflection of Σ_2^k around $x = .25$ we only need to track the interface Γ_{12}. The interface Γ_{12} remains a function of x at every time step. Thus to parameterize Γ_{12}, we fix x values in $[0, .25]$ and update corresponding y values for each step in the VIIM. The simulation of mean curvature motion for time T using parameterized curves is detailed below:

1. Choose Δt, the time between reconstructions, and n, the number of points. Set $N = T/\Delta t$.
2. Pick $\{x_i\}_{i=1}^n \in [0, .25]$ and set $y_i^0 = f(x_i, 0)$. In our implementation x_i's are chosen so that $(x_{i-1}-x_i)^2+(f(x_{i-1}, 0)-f(x_i, 0))^2$ are all equal for $i = 2, 3, \ldots, n$.
3. For $k = 1, \ldots, N$ do the following steps:
   (a) Build parameterization $\gamma^{\epsilon+} = \{(x, y) : d_{\Sigma_1^{k-1}}(x, y) = \epsilon\}$: For each x_i, we find $y_i^{\epsilon+}$ such that $d_{\Sigma_1^{k-1}}(x_i, y_i^{\epsilon+}) = \epsilon$, then $(x_i, y_i^{\epsilon+})$ is a parameterization of $\gamma^{\epsilon+}$. To find $d_{\Sigma_1^{k-1}}(x_i, y_i^{\epsilon+}) = \epsilon$, we use the secant method

(5.8)
$$\bar{y}^n \leftarrow \bar{y}^{n-1} + (d_{\Sigma_1^{k-1}}(x_i, \bar{y}^{n-1}) - \epsilon)\left(\frac{\bar{y}^{n-1} - \bar{y}^{n-2}}{d_{\Sigma_1^{k-1}}(x_i, \bar{y}^{n-1}) - d_{\Sigma_1^{k-1}}(x_i, \bar{y}^{n-2})}\right)$$

until

$$|d_{\Sigma_1^{k-1}}(x_i, \bar{y}^n) - \epsilon| < \delta \tag{5.9}$$

for $\delta > 0$. In our tests $|\frac{d}{dy} d_{\Sigma_1^{k-1}}(x_i, y)| > \frac{1}{4}$ near the true solution, so (5.9) can be used to bound the error in $\bar{y}^n$. Similar statements hold when we employ the secant method in subsequent steps. We choose δ small enough so that the error from estimating $\bar{y}^n$ is negligible compared to the overall error.

(b) Build parameterization $\gamma^{\epsilon-} = \{(x, y) : d_{\Sigma_2^{k-1}}(x, y) = \epsilon\}$: We let $x_i^{\epsilon-} = x_i \frac{.25-\epsilon}{.25}$ and find $y_i^{\epsilon-}$ such that $d_{\Sigma_2^{k-1}}(x_i^{\epsilon-}, y_i^{\epsilon-}) = \epsilon$ using the secant method. Then

$$\gamma_i^{\epsilon-} = \begin{cases} (x_i^{\epsilon-}, y_i^{\epsilon-}) & \text{if } i \le n \\ \left(x_n^{\epsilon-}, y_n^{\epsilon-} + (i-n)\frac{(-.5 - y_N^{\epsilon-})}{M} \right) & \text{if } n+1 \le i \le n+m \end{cases}$$

for some choice of m. We choose m such that

$$\frac{(-.5 - y_n^{\epsilon-})}{m} \approx \sqrt{(x_{n-1}^{\epsilon-} - x_n^{\epsilon-})^2 + (y_{n-1}^{\epsilon-} - y_n^{\epsilon-})^2}.$$

(c) Evolve $\gamma^{\epsilon+}$ and $\gamma^{\epsilon-}$ by (5.7) for time $\sigma_1 \Delta t$ and $\sigma_2 \Delta t$ respectively. Now $\gamma^{\epsilon+} = \partial \Sigma_1^{k-1}(\sigma_1 \Delta t, \epsilon)$ and $\gamma^{\epsilon-} = \partial \Sigma_2^{k-1}(\sigma_2 \Delta t, \epsilon)$.

(d) For each x_i find $\tilde{y}_i$ such that

$$d_{\Sigma_1^{k-1}(\sigma_1 \Delta t, \epsilon)}(x_i, \tilde{y}_i) = d_{\Sigma_2^{k-1}(\sigma_2 \Delta t, \epsilon)}(x_i, \tilde{y}_i).$$

We find $\tilde{y}_i$ using the secant method (the update is similar to (5.8)). To use $\gamma^{\epsilon-}$ to calculate $d_{\Sigma_2(\sigma_2 \Delta t, \epsilon)}(x, y)$, we apply a change of coordinates to make $\gamma^{\epsilon-}$ a function of x.

(e) Then assign each $\tilde{y}_i$ to y_i^k.

4. Then $\{(x_i, y_i^N)\}_{i=1}^n$ gives a parameterization of the interface Γ_{12} at time T.

5.4.3 Experimental Results

In experiments with the VIIM, it suffices to focus on the symmetric cases $\theta_2 = \theta_3$ to demonstrate non-convergence. Denote the output of the VIIM at time T by $\widehat{\Sigma}_j(T)$ and the true solution by $\Sigma_j(T)$. We track essentially the relative error (RE) of the area of symmetric difference in phase Σ_1:

$$\frac{\left|\widehat{\Sigma}_1(T) \triangle \Sigma_1(T)\right|}{\left|\Sigma_1(T) \triangle \Sigma_1(0)\right|} \tag{5.10}$$

where

$$A \triangle B = \{z : (z \in A \text{ and } z \notin B) \text{ or } (z \notin A \text{ and } z \in B)\}$$

but restricted to $\{(x, y) \ : 0 \leq x \leq 0.21\}$ to exclude a small neighborhood around the junction at $x = \frac{1}{4}$.

Each of the simulations uses the following parameters

- The total time the system is evolved: $T = 18/512$.
- Number of points tracked on the parameterized curve: $n = 2048$.
- Step size in (5.7): $\delta t = 2^{-12}\sigma\Delta t$.
- In the equal surface tension case ($\theta_1 = 120°$): $\sigma_1 = \sigma_2 = 1$
- We use $\theta_1 = 90°$ in our test for the unequal surface tension case, so that $\sigma_1 = 2 - \sqrt{2}$ and $\sigma_2 = \sqrt{2}$

Refining in δt or n did not significantly change the relative error. Additionally, we simulate the formal limit, $\lim\limits_{\epsilon \to 0^+} \lim\limits_{\Delta t \to 0}$, by setting $\epsilon \propto \Delta t^{1/4}$.

The results are collected in table 5.2 through table 5.7. In none of the limit cases does the unequal surface tension case converge to the correct curve. This is seen in the non-vanishing relative error for $\theta_1 \neq 120°$. Later we will give an alternative

Table 5.2: $\lim_{\epsilon\to 0^+}\lim_{\Delta t\to 0}$, $\theta_1 = 120°$

Δt	ϵ	RE	Order
2^{-13}	$2^{-28/4}$	0.0339	-
2^{-14}	$2^{-29/4}$	0.0269	0.33
2^{-15}	$2^{-30/4}$	0.0214	0.33
2^{-16}	$2^{-31/4}$	0.0172	0.32

Table 5.3: $\lim_{\epsilon\to 0^+}\lim_{\Delta t\to 0}$, $\theta_1 = 90°$

Δt	ϵ	RE	Order
2^{-13}	$2^{-28/4}$	0.0361	-
2^{-14}	$2^{-29/4}$	0.0517	-
2^{-15}	$2^{-30/4}$	0.0667	-
2^{-16}	$2^{-31/4}$	0.0813	-

Table 5.4: $\lim_{\epsilon=c\sqrt{\Delta t}\to 0^+}$, $\theta_1 = 120°$

Δt	RE: $c = 4$	Order	RE: $c = 2$	Order
2^{-13}	0.1675	-	0.0833	-
2^{-14}	0.1078	0.636	0.0561	0.572
2^{-15}	0.0714	0.595	0.0383	0.551
2^{-16}	0.0482	0.566	0.0264	0.537

to the Voronoi reconstruction step that, in our numerical tests, convergences in the unequal surface case.

5.5 Threshold Dynamics

In this section, we present convergence studies for the threshold dynamics algorithm of [15] using a parametrized curve implementation similar to the one in chapter IV. There is by now ample evidence, including a conditional proof [28], for the convergence of this algorithm to the correct limit, including very general unequal surface tensions cases. This section is thus meant as a verification of the parametrized curve implementation (rather than threshold dynamics, which is not in doubt), and give confidence to the non-convergence results it yielded on the VIIM, presented in the previous section.

Table 5.5: $\lim_{\epsilon=c\sqrt{\Delta t}\to 0^+}$, $\theta_1 = 90°$

Δt	RE: $c = 4$	Order	RE: $c = 2$	Order
2^{-13}	0.0256	-	0.0524	-
2^{-14}	0.0826	-	0.0791	-
2^{-15}	0.1166	-	0.0963	-
2^{-16}	0.1379	-	0.1077	-

Table 5.6: $\lim_{\Delta t \to 0}$ with $\epsilon = 0$, $\theta_1 = 120°$

Δt	RE	Order
2^{-13}	0.0071	-
2^{-14}	0.0050	0.51
2^{-15}	0.0035	0.51
2^{-16}	0.0025	0.50
2^{-17}	0.0017	0.50
2^{-18}	0.0012	0.50
2^{-19}	0.0009	0.50

Table 5.7: $\lim_{\Delta t \to 0}$ with $\epsilon = 0$, $\theta_1 = 90°$

Δt	RE	Order
2^{-13}	0.0056	-
2^{-14}	0.0065	-
2^{-15}	0.0071	-
2^{-16}	0.0075	-
2^{-17}	0.0077	-
2^{-18}	0.0078	-
2^{-19}	0.0079	-

Table 5.8: Threshold Dynamics $\theta_1 = 120°$

Δt	n	RE	Order
2^{-13}	512	0.0081	-
2^{-14}	1024	0.0056	0.523
2^{-15}	2048	0.0039	0.515
2^{-16}	4096	0.0028	0.510
2^{-17}	8192	0.0019	0.507
2^{-18}	16384	0.0014	0.504
2^{-19}	32768	0.0010	0.504

Table 5.9: Threshold Dynamics $\theta_1 = 90°$

Δt	n	RE	Order
2^{-13}	512	0.0095	-
2^{-14}	1024	0.0067	0.516
2^{-15}	2048	0.0047	0.511
2^{-16}	4096	0.0033	0.507
2^{-17}	8192	0.0023	0.505
2^{-18}	16384	0.0016	0.505
2^{-19}	32768	0.0012	0.501

We use the following parameters:

- Each of the following simulations is evolved for time $T = 18/512$.
- For the equal surface tension case $\theta_1 = 120°$, we use $\sigma_{12} = \sigma_{13} = \sigma_{23} = 1$.
- For the unequal surface tension case with $\theta_1 = 90°$, we use $\sigma_{12} = \sigma_{13} = 1$ and $\sigma_{23} = \sqrt{2}$.

The results are in table 5.8 and table 5.9. We see convergence to the correct solution, including in the unequal surface tension case, bolstering our confidence in the algorithm and the parametrized curve implementation developed and used in this chapter. It is thus highly unlikely that the non-convergence observed in the previous section with the VIIM is due to the parametrized curve representation; it is likely due to the VIIM itself.

5.6 Correcting the VIIM: Dictionary Mapping

Before the variational formulation of threshold dynamics given in [15] extended the original algorithm of [30] from equal to arbitrary surface tensions in a systematic manner, a more heuristic extension was proposed by Ruuth in [38]. In this approach, a projection step is employed to "force" the correct Herring angle conditions at any triple junction, while multiple (≥ 4) junctions are treated more heuristically. Ruuth's projection is designed so that the stationary configuration for the underlying curvature flow of three flat interfaces meeting at a triple junction with the correct Herring angles remains *fixed* under one iteration of the overall algorithm. Motivated by Ruuth's approach, in this section we propose a new algorithm: the dictionary mapping implicit interface method (DMIIM), which replaces the Voronoi reconstruction step of the VIIM with a *dictionary reconstruction step* that is designed to have as a fixed point three flat interfaces meeting with the correct Herring angles. The three phases in such a configuration consist of sectors and after evolution by curvature motion we want to restore these phases to their original form by our reconstruction. As such our dictionary reconstruction step is based on the curvature evolution of sectors. A heuristic extension to arbitrary number of phases, much as in [38], is also discussed. While an analogue of the more systematic approach of [15] would be more satisfactory, no such variational formulation for the VIIM (that is simple and efficient to implement) is currently available – a matter that remains under investigation.

We describe dictionary reconstruction in the case of three phases. The first step is to build a *template map* for a triple junction of interfaces with surface tensions σ_1, σ_2, and σ_3. To that end, let $(\theta_1, \theta_2, \theta_3)$ be the triple junction angles corresponding to these surface tensions as given by equation (5.3), and fix a time step size Δt. Let

$\Omega_i(0)$ for $i \in \{1,2,3\}$ denote the sector

$$\Omega_i(0) = \Big\{(\theta, r) \,:\, r \geq 0 \text{ and } \sum_{j=0}^{i-1} \theta_j \leq \theta \leq \sum_{j=0}^{i} \theta_j\Big\}$$

in polar coordinates, with the proviso $\theta_0 = 0$. Let $\Omega_i(\sigma_i \Delta t)$ be the evolution of $\Omega_i(0)$ via motion by mean curvature for time $\sigma_i \Delta t$. Recall that $d_{\Omega_i(\sigma_i\Delta t)}$ is the signed distance function to $\Omega_i(\sigma_i \Delta t)$, see (5.4). We will define the *template map*, $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$, as

$$\Phi(x,y) = (d_{\Omega_1(\sigma_1\Delta t)}(x,y), d_{\Omega_2(\sigma_2\Delta t)}(x,y), d_{\Omega_3(\sigma_3\Delta t)}(x,y)) \tag{5.11}$$

and define the *template surface* $S \subset \mathbb{R}^3$ as the image of $\mathbb{R}^2$ under Φ. The template map, Φ, maps points in $\mathbb{R}^2$ to distances to the evolved sectors. The map Φ is injective and we will use $\Phi^{-1} : S \to \mathbb{R}^2$ in our reconstruction algorithm.

Recall in the VIIM that Σ_i^k is phase i at time step k. We describe how the DMIIM reconstructs the new phases, Σ_1^{k+1}, Σ_2^{k+1}, and Σ_3^{k+1} from $\Sigma_1^k(\sigma_1\Delta t)$, $\Sigma_2^k(\sigma_2\Delta t)$, and $\Sigma_3^k(\sigma_3\Delta t)$. Define $\Pi_S : \mathbb{R}^3 \to \mathbb{R}^3$ as the closest point projection onto S (with respect to the standard Euclidean distance in $\mathbb{R}^3$). We define the reconstructed phases at the $(k+1)$-st time step as

(5.12)
$$\Sigma_i^{k+1} = \Big\{\mathbf{z} \in D \,:\, \Phi^{-1} \circ \Pi_S\big(d_{\Sigma_1^k(\sigma_1\Delta t)}(\mathbf{z}), d_{\Sigma_2^k(\sigma_2\Delta t)}(\mathbf{z}), d_{\Sigma_3^k(\sigma_3\Delta t)}(\mathbf{z})\big) \in \Omega_i(0)\Big\}.$$

We are thus assigning the point $\mathbf{z}$ to the phase (or phases) whose corresponding sector contains the preimage of

$$\Pi_S(d_{\Sigma_1(\sigma_1\Delta t)}(\mathbf{z}), d_{\Sigma_2(\sigma_2\Delta t)}(\mathbf{z}), d_{\Sigma_3(\sigma_3\Delta t)}(\mathbf{z})) \in S$$

under the one-to-one map Φ. The configuration $\big(\Omega_1(0), \Omega_2(0), \Omega_3(0)\big)$ is clearly fixed under the DMIIM algorithm, which thus treats a triple junction with the correct

Herring angles and straight interfaces exactly as it should. (Note that having these triple junctions fixed is a reassuring but not necessary condition for convergence of such algorithms; e.g. threshold dynamics [15] does not have this property in general). See fig. 5.3 for a schematic of dictionary reconstruction.

Given the surface tensions $(\sigma_1, \sigma_2, \sigma_3)$ and a time step size $\Delta t > 0$, the corresponding projection surface S is neither bounded nor smooth. Moreover, at its non-empty ridge, the closest point projection map $\Pi_S : \mathbb{R}^3 \to S$ contains multiple points. However, $\Pi_S(\mathbf{x})$ is non-empty at any $\mathbf{x} \in \mathbb{R}^3$.

The following two claims ensure that Π_S is well defined (i.e. $\Pi_S(\mathbf{x})$ contains a single point) and smooth in a neighborhood of the vacuum and overlap regions formed by evolving each phase by mean curvature motion, starting from an initial configuration of three smooth curves meeting at almost the correct Herring angles. The claims are straightforward but tedious to check, so the proofs are omitted.

The first concerns the surface S and follows from properties of the self-similar solution of curvature motion discussed in section 5.6.1:

Claim V.1. *Given* $\mathbf{r} = (r_1, r_2, r_3) \in (\mathbb{R}^+)^3$*, let* $T_\mathbf{r} = \cap_{j=1}^3 \{\mathbf{x} \in \mathbb{R}^2 \, : \, d_{\Omega_j(0)}(\mathbf{x}) < r_j\}$ *denote a neighborhood of the stationary triple junction with the exact Herring angles. There exists* $\mathbf{r} > \mathbf{0}$ *and* $\varepsilon > 0$ *such that the closest point projection map* $\Pi_S : \mathbb{R}^3 \to S$ *is well defined and smooth on* $N_\varepsilon = \{\mathbf{x} \in \mathbb{R}^3 \, : \, d(\mathbf{x}, \Phi(T_\mathbf{r})) < \varepsilon\}$.

The second can be checked e.g. using the comparison principle satisfied by motion by mean curvature:

Claim V.2. *Let* $\mathbf{r}, \varepsilon, T_\mathbf{r}$*, and* N_ε *be as in Claim 1. Let* $(\partial\Sigma_i(0)) \cap (\partial\Sigma_j(0))$*,* $i \neq j$*, be smooth curves meeting at a triple junction with angles* $(\theta_1', \theta_2', \theta_3')$*. There exist* $\delta > 0$

and $T > 0$ such that if $||(\theta_1, \theta_2, \theta_3) - (\theta_1', \theta_2', \theta_3')|| \leq \delta$ and $\Delta t \leq T$, then

$$\frac{1}{\sqrt{\Delta t}}(d_{\Sigma_1(\sigma_1 \Delta t)}(\mathbf{x}), d_{\Sigma_2(\sigma_2 \Delta t)}(\mathbf{x}), d_{\Sigma_3(\sigma_3 \Delta t)}(\mathbf{x})) \in N_\varepsilon \textit{ for any } \mathbf{x} \in \bigcup_{\substack{0 \leq t \leq \Delta t \\ j \in \{1,2,3\}}} \partial \Sigma_j(\sigma_j t).$$

We next explain one way to extend the DMIIM algorithm just described for three phases to the n-phase setting. Let $\sigma_1, \sigma_2, \ldots, \sigma_n$ be the surface tensions associated with the n phases. Let $\mathbf{I}(\mathbf{z}, \cdot) : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ be a bijection (permutation of the indices) so that

$$d_{\Sigma^k_{\mathbf{I}(\mathbf{z},1)}(\sigma_{\mathbf{I}(\mathbf{z},1)}\Delta t)}(\mathbf{z}) \geq d_{\Sigma^k_{\mathbf{I}(\mathbf{z},2)}(\sigma_{\mathbf{I}(\mathbf{z},2)}\Delta t)}(\mathbf{z}) \geq \cdots \geq d_{\Sigma^k_{\mathbf{I}(\mathbf{z},n)}(\sigma_{\mathbf{I}(\mathbf{z},n)}\Delta t)}(\mathbf{z})$$

so that $\mathbf{I}(\mathbf{z}, j)$ is the label of the phase with the j-th largest signed distance function at the point $\mathbf{z}$. Our simple extension, similar to the one in [38], is to allocate each $\mathbf{z}$ using the very same dictionary mapping discussed above, where the three phases used in the construction of the projection surface are the closest three to $\mathbf{z}$. Under this rule the new sets become

$$(5.13) \quad \Sigma_j^{k+1} = \Big\{ \mathbf{z} \in D \, : \, \mathbf{I}^{-1}(\mathbf{z}, j) \leq 3 \text{ and}$$
$$\Phi^{-1} \circ \Pi_S \big(d_{\Sigma^k_{\mathbf{I}(\mathbf{z},1)}(\sigma_{\mathbf{I}(\mathbf{z},1)}\Delta t)}(\mathbf{z}), d_{\Sigma^k_{\mathbf{I}(\mathbf{z},2)}(\sigma_{\mathbf{I}(\mathbf{z},2)}\Delta t)}(\mathbf{z}), d_{\Sigma^k_{\mathbf{I}(\mathbf{z},3)}(\sigma_{\mathbf{I}(\mathbf{z},3)}\Delta t)}(\mathbf{z}) \big) \in \Omega_{\mathbf{I}^{-1}(\mathbf{z},j)}(0) \Big\}$$

where Ω_j for $j \in \{1, 2, 3\}$, the map Φ and the surface S are constructed at each $\mathbf{z} \in D$ as in the three phase case, using the surface tensions $\sigma_{\mathbf{I}(\mathbf{z},1)}$, $\sigma_{\mathbf{I}(\mathbf{z},2)}$, and $\sigma_{\mathbf{I}(\mathbf{z},3)}$. High degree junctions are thus treated heuristically by this method. Indeed, it is not hard to come up with alternatives to the simple extension (5.13) explained above. Although we expect all such natural extensions to behave mostly the same, subtle differences between them cannot be ruled out at this point. We took the simplest example (5.13), and while points near high degree junctions can be far away from the template surface, we will show that it behaves reasonably in section 5.6.5.

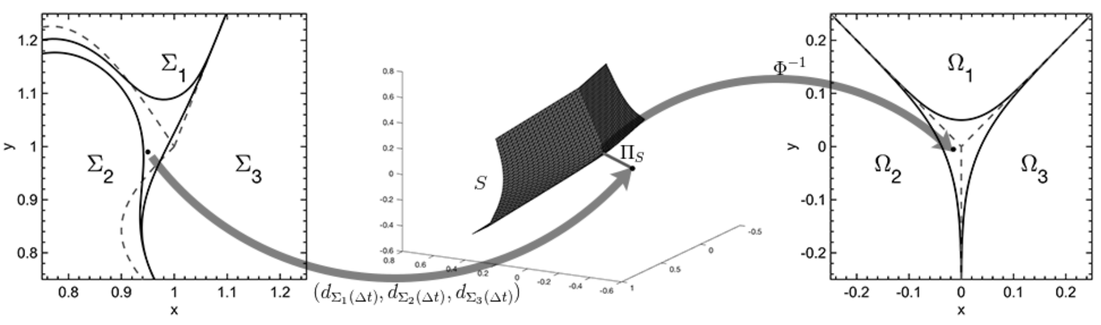

Figure 5.3: A schematic of the dictionary reconstruction step. The dashed lines are the interfaces at $T = 0$ and the solid lines are sets at time $T = \Delta t$. In this example the point would be allocated to Σ_2.

The DMIIM also allows some control over mobilities. Each phase Σ_i can be assigned a mobility μ_i. In the evolution step the level sets are evolved for time $\mu_i \sigma_i \Delta t$ by (5.5) and we use $\Omega_i(\mu_i \sigma_i \Delta t)$ in the construction of the template surface. The mobility at the interface Γ_{ij} becomes

$$\mu_{ij} = \frac{\mu_i \sigma_i + \mu_j \sigma_j}{2\sigma_{ij}}. \tag{5.14}$$

With this extra flexibility, the DMIIM allows the specification of the $\binom{n}{2}$ physically relevant surface tensions at the interfaces, Γ_{ij}, and the products, $\mu_i \sigma_i$, for each phase. The mobility at the interface Γ_{ij} is constrained by (5.14). We give an example in section 5.6.5.

Algorithm 14 details the steps in the DMIIM. In the next two subsections we describe how to find $\Omega_i(t)$ with high accuracy and how the projection is performed.

5.6.1 Self-Similar Solution to Curvature Motion

Building the template map in the DMIIM algorithm requires precomputing the solution of motion by mean curvature of a sector (denoted $\Omega(t)$ in the previous section) to high accuracy. The set $\Omega(t)$ is related to a self-similar solution of mean curvature motion; thus the computation of $\Omega(t)$ can be reduced to the following

Algorithm 14 Dictionary Mapping Implicit Interface Method

```
1: Given $\Sigma_1^0, \Sigma_2^0, \ldots, \Sigma_n^0$, $\sigma_1, \sigma_2, \ldots, \sigma_n$, $\mu_1, \mu_2, \ldots, \mu_n$, $\Delta t$, and $T$.
2: Let $N = T/\Delta t$. Define the *reduced mobilities* $\bar{\mu}_i = \mu_i \sigma_i$.
3: for $k \leftarrow 1$ to $N$ do
4:     Evolve each $\Sigma_i^k(0)$ by time $\bar{\mu}_i \Delta t$ to get $\Sigma_i^k(\bar{\mu}_i \Delta t)$.
5:     Construct the new phases
```

$$\Sigma_i^{k+1} = \Big\{ \mathbf{z} \in D \,:\, \mathbf{I}^{-1}(\mathbf{z}, i) \leq 3 \text{ and } \Phi^{-1} \circ \Pi_S \big(d_{\Sigma^k_{\mathbf{I}(\mathbf{z},1)}(\bar{\mu}_{\mathbf{I}(\mathbf{z},1)}\Delta t)}(\mathbf{z}), d_{\Sigma^k_{\mathbf{I}(\mathbf{z},2)}(\bar{\mu}_{\mathbf{I}(\mathbf{z},2)}\Delta t)}(\mathbf{z}), d_{\Sigma^k_{\mathbf{I}(\mathbf{z},3)}(\bar{\mu}_{\mathbf{I}(\mathbf{z},3)}\Delta t)}(\mathbf{z}) \big) \in \Omega_{\mathbf{I}^{-1}(\mathbf{z},i)}(0) \Big\}$$

ODE:

$$\begin{aligned} &\phi''(x) = \frac{1}{2}(\phi(x) - x\phi'(x))(1 + (\phi'(x))^2) \\ &\phi'(0) = 0 \\ &\lim_{x\to\infty} \phi(x) = \infty \\ &\phi(0) = \phi_0 > 0 \end{aligned} \tag{5.15}$$

on the domain $x \in (-\infty, \infty)$ and ϕ is even. Instead of the last condition of (5.15), we could specify a M such that $\lim_{x\to\infty} \phi'(x) = M > 0$. There is a bijective map between ϕ_0 and M [24]. We next explain how (5.15) arises.

For a curve given as the graph of a function $u(x, t)$, motion by curvature is described by the PDE

$$u_t = \frac{u_{xx}}{1 + u_x^2}.$$

When $u(x, t = 0) = M|x|$ for some M then

$$u(x, t) = \sqrt{t}\phi(x/\sqrt{t})$$

where ϕ satisfies (5.15). Let the positive y-axis bisect the sector $\Omega(0)$ with angle θ, then $\partial\Omega(0) = M|x|$ for $M = \cot(\theta/2)$ and

$$\Omega(t) = \{(x, y) : y \geq \sqrt{t}\phi(x/\sqrt{t})\}.$$

To find the numerical solution to the ODE (5.15), we use the Newton iteration

$$\left[\frac{2}{h^2} - \frac{x_i}{2h} - \frac{x_i}{2h}\left(\frac{\phi_{i+1}^k - \phi_{i-1}^k}{2h}\right)^2\right]\phi_{i-1}^{k+1} - \left[\frac{4}{h^2} + 1 + \left(\frac{\phi_{i+1}^k - \phi_{i-1}^k}{2h}\right)^2\right]\phi_i^{k+1} \tag{5.16}$$

$$+ \left[\frac{2}{h^2} + \frac{x_i}{2h} + \frac{x_i}{2h}\left(\frac{\phi_{i+1}^k - \phi_{i-1}^k}{2h}\right)^2\right]\phi_{i+1}^{k+1} = 0 \tag{5.17}$$

for $x_0 = 0$ and x_N being sufficiently large. The boundary conditions are $\phi_{-1} = \phi_1$ and $\phi_N = Mx_N$. Below we prove that ϕ_N is close to Mx_N as long as we choose x_N large enough, justifying the second boundary condition. We offer an improved bound over

$$\phi(x) = Mx + o\left(\frac{1}{x}\right), \text{ as } x \to \infty. \tag{5.18}$$

given in [24], where the ODE was previously studied. The improved bound implies we do not need to take x_N so large. In our simulations we choose x_N to be 10.

Claim V.3. *For a function $\phi(x)$ satisfying* (5.15) *the following bounds hold:*

$$|\phi(x) - xM| \le C_0 e^{-\frac{x^2(1+M^2)}{4}}, \, x \ge 0$$
$$|\phi'(x) - M| \le \frac{C_0}{x} e^{-\frac{x^2(1+M^2)}{4}}, \, x > 0$$
$$|\phi''(x)| \le C_1 e^{-\frac{x^2(1+M^2)}{4}}, \, x \ge 0$$

where C_0 and C_1 only depend on ϕ_0 (or M).

We first need the following lemmas

Lemma V.4. *The function ϕ satisfies the following properties:*

1. *$\phi''(x) > 0$.*
2. *$0 \le \phi'(x) < M$.*
3. *$\phi(x) > xM$.*

Proof. The proof of property 1 is given in [24]. As a result of $\phi''(x) > 0$, $\phi'(x)$ is a strictly increasing function with $\lim_{x\to\infty} \phi'(x) = M$, so property 2 follows. To prove property 3, let $h(x) = \phi(x) - xM$ on $x \geq 0$. By (5.18) $\lim_{x\to\infty} h(x) = 0$ and property 2 implies $h'(x) = \phi'(x) - M < 0$. Thus $h(x) > 0$ for all $x \geq 0$. □

Lemma V.5. *The function ϕ satisfies the first order differential equation*

$$\exp\left(\frac{\phi^2(0)}{2}\right)\phi^2(0) = (\phi(x) - x\phi'(x))^2 \exp\left(\frac{x^2}{2} + \frac{\phi^2(x)}{2}\right)(1 + (\phi'(x))^2)^{-1}.$$

Proof. Rearrange

$$\phi''(\tilde{x}) = \frac{1}{2}(\phi(\tilde{x}) - \tilde{x}\phi'(\tilde{x}))(1 + (\phi'(\tilde{x}))^2) \tag{5.19}$$

to

$$\frac{-2\tilde{x}\phi''(\tilde{x})}{\phi(\tilde{x}) - \tilde{x}\phi'(\tilde{x})} = -\tilde{x} + \tilde{x}(\phi'(\tilde{x}))^2.$$

By integrating both sides from 0 to x we obtain

$$2\log(\phi(x) - x\phi'(x)) - 2\log(\phi_0 = -\frac{1}{2}x^2 - \int_0^x \tilde{x}(\phi'(\tilde{x}))^2 d\tilde{x}. \tag{5.20}$$

Now rearrange (5.19) to

$$\frac{2\phi'(\tilde{x})\phi''(\tilde{x})}{1 + (\phi'(\tilde{x}))^2} = \phi(\tilde{x})\phi'(\tilde{x}) - \tilde{x}(\phi'(\tilde{x}))^2.$$

By integrating both sides from 0 to x we obtain

$$\log(1 + (\phi'(x))^2) = \frac{1}{2}\phi^2(x) - \frac{1}{2}\phi_0^2 - \int_0^x \tilde{x}(\phi'(\tilde{x}))^2 d\tilde{x}. \tag{5.21}$$

Both (5.20) and (5.21) have a $-\int_0^x \tilde{x}\phi'(\tilde{x})^2 d\tilde{x}$ term. Solving for that term in (5.20) and (5.21) and setting the equations equal to each other results in

$$\log(1 + (\phi'(x))^2) - \frac{1}{2}\phi^2(x) + \frac{1}{2}\phi_0^2 = 2\log(\phi(x) - x\phi'(x)) - 2\log(\phi_0) + \frac{1}{2}x^2.$$

The conclusion of the lemma follows from taking the exponential of both sides. □

Using the above two lemmas we can establish claim V.3:

Proof. (Of claim V.3) Applying the lemmas we have

$$\begin{aligned}&(\phi(x) - x\phi'(x))^2\\ =&\exp\left(-\frac{x^2}{2}+\frac{\phi_0^2}{2}-\frac{\phi^2(x)}{2}\right)(1+(\phi'(x))^2)\phi_0^2\\ \le&\exp\left(-\frac{x^2}{2}-\frac{(Mx)^2}{2}\right)(1+M^2)\phi_0^2\exp\frac{\phi_0^2}{2}\end{aligned}$$

Additionally, invoking inequalities 2 and 3 from lemma V.4.

$$|\phi(x) - x\phi'(x)| = |\phi(x) - xM + xM - x\phi'(x)| = |\phi(x) - xM| + |xM - x\phi'(x)|.$$

The first inequality follows from observing that $|\phi(x) - xM| < |\phi(x) - x\phi'(x)|$ and the second from $|M - \phi'(x)| < \frac{1}{x}|\phi(x) - x\phi'(x)|$ for $x > 0$. Then using (5.19) $|\phi''(x)| \le \frac{1}{2}|\phi(x) - x\phi'(x)|(1 + M^2)$ and the third inequality follows. □

5.6.2 Projecting onto the Template Surface

Another important step in the DMIIM algorithm introduced in section 5.6 is the closest point projection onto the template surface. Here, we discuss the details of a highly accurate method for projecting an arbitrary point $\mathbf{w} \in \mathbb{R}^3$ onto the template surface $S = \{\Phi(x, y) : (x, y) \in \mathbb{R}^2\}$ for the template map Φ defined in (5.11). Define the function

$$F(x, y) = \frac{1}{2}||\mathbf{w} - \Phi(x, y)||_2^2. \tag{5.22}$$

Denote (x^*, y^*) as the minimum of F or where $\Phi(x^*, y^*) = \Pi_S(\mathbf{w})$. To find (x^*, y^*) we will use Newton's method,

$$\begin{bmatrix} x^{n+1} \\ y^{n+1} \end{bmatrix} = \begin{bmatrix} x^n \\ y^n \end{bmatrix} - (D^2F)^{-1}\nabla F(x^n, y^n).$$

To choose (x^0, y^0), we make a coarse point cloud of the surface S and choose (x^0, y^0) so that $\Phi(x^0, y^0)$ is the nearest point to $\mathbf{w}$ in the point cloud.

The rest of this section details how to compute ∇F and $D^2 F$. First we find partial derivatives of F in terms of the distance functions, for example:

$$\begin{aligned}
\frac{\partial F}{\partial x} &= \sum_{i=1}^{3} \left(\frac{\partial}{\partial x} d_{\Omega_i(\sigma_i \mu_i \Delta t)} \right) (d_{\Omega_i(\sigma_i \mu_i \Delta t)}(x,y) - \mathbf{w}_i) \\
\frac{\partial^2 F}{\partial x^2} &= \sum_{i=1}^{3} \left(\frac{\partial^2}{\partial x^2} d_{\Omega_i(\sigma_i \mu_i \Delta t)} \right) (d_{\Omega_i(\sigma_i \mu_i \Delta t)} - \mathbf{w}_i) + \left(\frac{\partial}{\partial x} d_{\Omega_i(\sigma_i \mu_i \Delta t)} \right)^2 .
\end{aligned}$$

We will next demonstrate how to find explicit formulas for the partial derivatives of the distance functions. There exists a rotation of angle θ, denote as R_θ, such that

$$R_\theta(\Omega_i(\sigma_i \mu_i \Delta t)) = \{(x,y) : y \geq f(x)\}$$

for $f(x) = \sqrt{\sigma_i \mu_i \Delta t}\,\phi(|x|/\sqrt{\sigma_i \mu_i \Delta t})$. For the moment, consider the case where $\theta = 0$. Let

$$g(x,y,p) = \frac{-(x-p)f'(p) + (y - f(p))}{\sqrt{1 + (f'(p))^2}}.$$

Additionally, let $p^*(x,y) = \arg\min_p (x-p)^2 + (y - f(p))^2$ be the x-coordinate of the closest point on the curve $(q, f(q))$ to (x,y). Note that

$$\text{(5.23)} \qquad \frac{d}{dp}[(x-p)^2 + (y-f(p))^2]|_{p^*} = 2(x - p^*) + 2(y - f(p^*))f'(p^*) = 0.$$

We have $g(x,y,p^*(x,y)) = d_{\Omega_i(\sigma_i \mu_i \Delta t)}(x,y)$ (see proposition 1 in [16]). We can find the partial derivatives of the distance function by differentiating g, for example:

$$\begin{aligned}
\frac{d}{dx} d_{\Omega_i(\sigma_i \mu_i \Delta t)} &= g_x + g_p p^*_x \\
\frac{d^2}{dx^2} d_{\Omega_i(\sigma_i \mu_i \Delta t)} &= g_{xx} + 2 g_{px} p^*_x + g_{pp} (p^*_x)^2 + g_p p^*_{xx}.
\end{aligned}$$

The partial derivatives of p^* are obtained by implicitly differentiating $(x - p^*) + (y - f(p^*))f'(p^*) = 0$. We have that

$$g_p = \frac{[(x-p) + (y - f(p))f'(p)]f''(p)}{(1 + (f'(p))^2)^{3/2}} = 0$$

by (5.23), so we do not need to solve for p^*_{xx}. Applying the above for an arbitrary angle θ, the partial derivatives of the distance function are

$$
\begin{aligned}
\frac{\partial}{\partial x} d_{\Omega_i(\sigma_i\mu_i\Delta t)} &= \frac{-\cos(\theta) f'(p^*) + \sin(\theta)}{\sqrt{1+(f'(p^*))^2}} \\
\frac{\partial}{\partial y} d_{\Omega_i(\sigma_i\mu_i\Delta t)} &= \frac{\sin(\theta) f'(p^*) + \cos(\theta)}{\sqrt{1+(f'(p^*))^2}} \\
\frac{\partial^2}{\partial x^2} d_{\Omega_i(\sigma_i\mu_i\Delta t)} &= [-\cos^2(\theta) - 2\sin(\theta)\cos(\theta) f'(p^*) - \sin^2(\theta)(f'(p^*))^2] h(x,y,p^*) \\
\frac{\partial^2}{\partial x \partial y} d_{\Omega_i(\sigma_i\mu_i\Delta t)} &= [\sin(\theta)\cos(\theta) - \cos(2\theta) f'(p^*) - \sin(\theta)\cos(\theta)(f'(p^*))^2] h(x,y,p^*) \\
\frac{\partial^2}{\partial y^2} d_{\Omega_i(\sigma_i\mu_i\Delta t)} &= [-\sin^2(\theta) + 2\sin(\theta)\cos(\theta) f'(p^*) - \cos^2(\theta)(f'(p^*))^2] h(x,y,p^*) \\
h(x,y,p^*) &= \left(\frac{1}{1+(f'(p^*))^2 + [f(p^*) - (x\sin(\theta) + y\cos(\theta))] f''(p^*)}\right)\left(\frac{f''(p^*)}{[1+(f'(p^*))^2]^{3/2}}\right).
\end{aligned}
$$

We now have explicit formulas for every step of Newton's method allowing us to quickly and accurately minimize (5.22) to find the closest point projection.

5.6.3 The DMIIM's Relationship to the VIIM

In this section, we discuss the precise relationship between the DMIIM and the VIIM. We consider the very special case of equal surface tensions, $\sigma_i = 1$ (for all i), corresponding to the Herring angles of $(120°, 120°, 120°)$. This is the one case in which our numerical results from previous sections suggest the VIIM converges to the correct solution. The standard maximum principle for two-phase motion by mean curvature implies that overlaps cannot occur at the end of the curve evolution step of the VIIM or the DMIIM. The only interesting question is how the two algorithms allocate points in the "vacuum" region, $\{\mathbf{z} : d_{\Sigma_j(\Delta t)}(\mathbf{z}) < 0 \text{ for all } j\}$.

Due to the symmetry of this situation, if $\mathbf{p} \in S$, then any $\mathbf{q} \in \mathbb{R}^3$ obtained by a permutation of the components of $\mathbf{p}$ also satisfies $\mathbf{q} \in S$.

Let $\mathbf{x} \in \mathbb{R}^3$ be given, with $\mathbf{x}_i < 0$ for all i. Let $\mathbf{p} = \Pi_S(\mathbf{x})$ with $\mathbf{p} = \Phi_S(\mathbf{z})$ for some $\mathbf{z} \in \mathbb{R}^2$. We are thus assuming implicitly that $\Pi_S(\mathbf{x})$ consists of a single point

$\mathbf{p} \in S$.

Claim V.6. $\mathbf{x}_i = \max(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ *if and only if* $\mathbf{p}_i = \max(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$.

Proof. The proof will be broken up into three parts: For $i \neq j$ (1) if $\mathbf{x}_i \geq \mathbf{x}_j$ then $\mathbf{p}_i \geq \mathbf{p}_j$, (2) if $\mathbf{p}_i = \mathbf{p}_j$, then $\mathbf{x}_i = \mathbf{x}_j$ and (3) if $\mathbf{p}_i \geq \mathbf{p}_j$, then $\mathbf{x}_i \geq \mathbf{x}_j$. Statements (1) and (3) then imply the claim.

(1) If $\mathbf{x}_i \geq \mathbf{x}_j$ and $\mathbf{p}_i < \mathbf{p}_j$, then

$$||\mathbf{x} - \mathbf{q}|| \leq ||\mathbf{x} - \mathbf{p}||$$

where $\mathbf{q} \in S$ is obtained from $\mathbf{p}$ by interchanging its i-th and j-th components (since $\mathbf{p}_i \neq \mathbf{p}_j$, then $\mathbf{q} \neq \mathbf{p}$). Indeed,

$$||\mathbf{x} - \mathbf{p}||^2 = ||\mathbf{x} - \mathbf{q}||^2 + 2(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{p}_j - \mathbf{p}_i) \geq ||\mathbf{x} - \mathbf{q}||^2.$$

This contradicts $\mathbf{p} = \Pi_S(\mathbf{x})$; so the first statement is established.

(2) If $\mathbf{p}_i = \mathbf{p}_j$, let $\mathbf{n}$ denote a unit normal to S at $\mathbf{p}$. A short calculation shows that $\mathbf{n}_i = \mathbf{n}_j$. Indeed:

$$\mathbf{n}_i = \mathbf{n}_j = D_{\mathbf{u}} d_{\Omega_i(\Delta t)}|_{\Phi^{-1}(\mathbf{p})}$$

where $\mathbf{u}$ is the unit vector perpendicular to $\partial\Omega_i(0) \cap \partial\Omega_j(0)$ pointing into $\Omega_i(0)$. Since $\mathbf{p} = \Pi(\mathbf{x})$ implies that $\mathbf{x} - \mathbf{p} = \beta\mathbf{n}$ for some $\beta \in \mathbb{R}$, we get $\mathbf{x}_i = \mathbf{x}_j$.

(3) Assume $\mathbf{x}_i < \mathbf{x}_j$. Since $\mathbf{x}_i \leq \mathbf{x}_j$, by statement (1) $\mathbf{p}_i \leq \mathbf{p}_j$. Furthermore, since $\mathbf{x}_i \neq \mathbf{x}_j$ statement (2) implies $\mathbf{p}_i \neq \mathbf{p}_j$. Hence $\mathbf{p}_i < \mathbf{p}_j$. □

Claim V.7. *Let the phases* Σ_i^k *at time step* k *have smooth boundaries, with triple junctions in the same neighborhood of* $(120°, 120°, 120°)$ *as in claim V.2. Then, for every small enough time step size* $\Delta t > 0$*, the DMIIM and the VIIM yield the same* Σ_i^{k+1}.

Proof. Due to the symmetry $\mathbf{z} \in \Omega_i(0)$ if and only if $d_{\Omega_i(\Delta t)}(\mathbf{z}) = \max_j d_{\Omega_j(\Delta t)}(\mathbf{z})$. A consequence is that

$$\Phi(\Omega_i(0)) = \{\mathbf{p} \in S : \mathbf{p}_i = \max(\mathbf{p})\}. \tag{5.24}$$

The projection map Φ is well defined for points $(d_{\Sigma_1(\Delta t)}(\mathbf{z}), d_{\Sigma_2(\Delta t)}(\mathbf{z}), d_{\Sigma_3(\Delta t)}(\mathbf{z}))$, where $d_{\Sigma_j(\Delta t)}(\mathbf{z}) < 0$ for all j by claim V.2. Then claim V.6 along with (5.24) give us that $d_{\Sigma_i(\Delta t)}(\mathbf{z}) = \max_j d_{\Sigma_j(\Delta t)}(\mathbf{z})$ if and only if

$$\Pi_S(d_{\Sigma_1(\Delta t)}(\mathbf{z}), d_{\Sigma_2(\Delta t)}(\mathbf{z}), d_{\Sigma_3(\Delta t)}(\mathbf{z})) \in \Phi(\Omega_i(0))$$

completing the proof. □

The dictionary mapping implicit interface method is then an extension of the Voronoi implicit interface method to cases of unequal surface tension. As we show in the next section, the DMIIM numerically converges to the exact solution in cases of unequal surface tensions.

5.6.4 Numerical Results for the DMIIM

Algorithm 15 Parameterized DMIIM for "Grim Reaper" tests with $\theta_2 = \theta_3$.

```
1: Let $N = T/\Delta t$.
2: Choose points $\{x_i\}_{i=1}^n \in [0, .25]$ and set $y_i^0 = f(x_i, 0)$.
3: for $k \leftarrow 1$ to $N$ do
4:     Use $\{x_i, y_i^{k-1}\}_{i=1}^n$ to parameterize $\partial\Sigma_1$ and $\partial\Sigma_2$, denoted as $\gamma^+$ and $\gamma^-$ respectively.
5:     Evolve $\gamma^+$ and $\gamma^-$ by (5.7) for time $\mu_1\sigma_1\Delta t$ and $\mu_2\sigma_2\Delta t$ respectively.
6:     For each $x_i$ find $\tilde{y}_i$ such that
           $(x^*, y^*) = \Phi^{-1} \circ \Pi_S(d_{\Sigma_1(\sigma_1\mu_1\Delta t)}(x_i, \tilde{y}_i), d_{\Sigma_2(\sigma_2\mu_2\Delta t)}(x_i, \tilde{y}_i), d_{\Sigma_3(\sigma_3\mu_3\Delta t)}(x_i, \tilde{y}_i))$
       satisfies $(x^*, y^*) \in \partial\Omega_1(0) \cup \partial\Omega_2(0)$.
7:     $y_i^k \leftarrow \tilde{y}_i$
```

In this section, we perform on the DMIIM the same careful numerical convergence tests that we subjected the VIIM to. In addition we test on some examples where all the surface tensions are different. Algorithm 15 details the implementation of the

Table 5.10: DMIIM, $\theta_1 = 120°$

Δt	n	RE	Order
2^{-10}	1024	0.0202	-
2^{-11}	1449	0.0141	0.520
2^{-12}	2048	0.0099	0.513
2^{-13}	2897	0.0069	0.509
2^{-14}	4096	0.0049	0.507
2^{-15}	5793	0.0034	0.505
2^{-16}	8192	0.0024	0.503
2^{-17}	11586	0.0017	0.498

Table 5.11: DMIIM, $\theta_1 = 90°$

Δt	n	RE	Order
2^{-10}	1024	0.00207	-
2^{-11}	1449	0.00107	0.954
2^{-12}	2048	0.00056	0.922
2^{-13}	2897	0.00031	0.842
2^{-14}	4096	0.00018	0.772
2^{-15}	5793	0.00011	0.701
2^{-16}	8192	0.00007	0.614
2^{-17}	11586	0.00005	0.576

Table 5.12: DMIIM, $(75°, 135°, 150°)$, $\mu_i = 1$

Δt	n	RE	Order
2^{-10}	1024	0.0067	-
2^{-11}	1449	0.0053	0.338
2^{-12}	2048	0.0040	0.411
2^{-13}	2897	0.0029	0.450
2^{-14}	4096	0.0021	0.470
2^{-15}	5793	0.0015	0.480
2^{-16}	8192	0.0011	0.494

Table 5.13: DMIIM, $(75°, 135°, 150°)$, $\mu_i = \frac{1}{\sigma_i}$

Δt	n	RE	Order
2^{-10}	1024	0.0138	-
2^{-11}	1449	0.0094	0.548
2^{-12}	2048	0.0065	0.540
2^{-13}	2897	0.0045	0.533
2^{-14}	4096	0.0031	0.527
2^{-15}	5793	0.0022	0.522
2^{-16}	8192	0.0015	0.521

DMIIM with parameterized curves for "Grim Reaper" tests with $\theta_2 = \theta_3$. The implementation of the non-symmetric case is similar. We note that in our implementation of algorithm 15 that projecting onto the template surface is the computation bottleneck, taking about 25 times longer than the Voronoi reconstruction step it replaces in the VIIM. This is mainly due to the number of points we use to represent each Ω_i. We choose to err on the side of caution in our numerical studies, representing Ω_i with many more points than needed.

We run the following "Grim Reaper" tests:

1. Angles $(\theta_1, \theta_2, \theta_3) = (120°, 120°, 120°)$ with $\sigma_1 = \sigma_2 = \sigma_3 = 1$.
2. Angles $(\theta_1, \theta_2, \theta_3) = (90°, 135°, 135°)$ with $\sigma_1 = 2 - \sqrt{2}$ and $\sigma_2 = \sigma_3 = \sqrt{2}$
3. Angles $(\theta_1, \theta_2, \theta_3) = (75°, 135°, 150°)$ with $\sigma_1 = \frac{1}{4}(-2 + 3\sqrt{2} + \sqrt{6})$, $\sigma_2 = \frac{1}{4}(2 - \sqrt{2} + \sqrt{6})$, $\sigma_3 = \frac{1}{4}(2 + \sqrt{2} - \sqrt{6})$ and $\mu_i = 1$.
4. Angles $(\theta_1, \theta_2, \theta_3) = (75°, 135°, 150°)$ with the same σ_i's as the previous test

with $\mu_i = \frac{1}{\sigma_i}$.

Each of the simulations use the following parameters:

- The total time the system is evolved: $T = 18/512$.
- Number of points tracked on the parameterized curve: n as given in the table.
- Step size in (5.7): $\delta t = \sigma\Delta t/n$.
- We are measuring the relative error (RE) of the area of symmetric difference of phase Σ_1 in $\{(x, y) : 0 \le x \le \beta - .04 \text{ or } \beta + .04 \le x \le .5\}$, see (5.10).

The results are contained in table 5.10 through table 5.13.

5.6.5 Level Set Examples of the DMIIM

We demonstrate the level set formulation of the DMIIM in two and three dimensions. In 2d we evolve a system that goes through a well understood topological change. Initially, we have two "Grim Reapers" translating vertically towards each other until they collide. After the collision, two new junctions form that travel horizontally away from each other forming a new horizontal interface between the top and bottom phases.

The initial configuration is

$$\begin{aligned}
\Sigma_1^0 =& \Big\{(x,y) : y > \frac{3}{4} + f(\frac{1}{4} - |x - \frac{1}{4}|)\Big\} \\
\Sigma_2^0 =& \Big\{(x,y) : x < \frac{1}{4} \text{ and } \frac{1}{4} + f(x) < y < \frac{3}{4} + f(x)\Big\} \\
\Sigma_3^0 =& \Big\{(x,y) : x < \frac{1}{4} \text{ and } \frac{1}{4} + f(x) < y < \frac{3}{4} + f(x)\Big\} \\
\Sigma_4^0 =& \Big\{(x,y) : \frac{1}{4} + f(\frac{1}{4} - |x - \frac{1}{4}|)\Big\}
\end{aligned}$$

where $f(x) = \frac{1}{\pi}\log(\cos(\pi x))$. We use the surface tensions matrix

$$\sigma = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & \sqrt{2} & 1 \\ 1 & \sqrt{2} & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

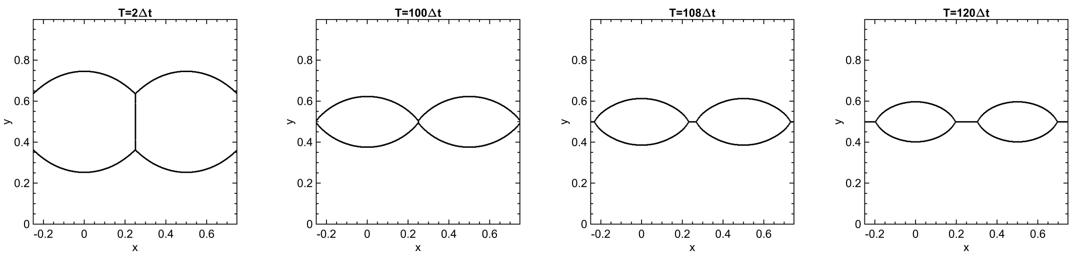

Figure 5.4: Two "Grim Reapers" colliding, computed using the DMIIM algorithm (practical implementation on uniform grid). The initial $(90°, 135°, 135°)$ angles change to $(120°, 120°, 120°)$ after the collision, as expected. The multiple junction that momentarily forms at collision appears to be handled appropriately.

and $(\mu_1\sigma_1, \mu_2\sigma_2, \mu_3\sigma_3, \mu_4\sigma_4) = (2 - \sqrt{2}, \sqrt{2}, \sqrt{2}, 2 - \sqrt{2})$. Before the topological change the angles are $(90°, 135°, 135°)$ at the triple junctions. After the change the angles are $(120°, 120°, 120°)$. For the simulation we use the domain $[0, 1] \times [0, 1]$ with periodic boundary conditions on a 512 by 512 grid. We set $\Delta t = \frac{1}{1600\pi \sin(3\pi/4)}$. Figure 5.4 shows this system at different times.

In three dimensions, we evolve a system starting from a Voronoi diagram of 8 points taken at random on the 3-torus. Six of the phases have surface tension equal to 1 while the other two phases have surface tension $\sqrt{2} - 1$. Thus three angle configurations are possible: $(120°, 120°, 120°)$, $(90°, 135°, 135°)$, and $\approx (146°, 107°, 107°)$ Quadruple points split and collide throughout the evolution causing the faces of the grains to undergo topological changes as seen in fig. 5.5. The two preceding examples show reasonable behavior and demonstrate the practical use of the DMIIM as a level set method.

5.7 Conclusion

In this work, we have presented careful numerical convergence studies showing that the Voronoi implicit interface method gets close but does not converge to the correct evolution in the unequal surface tension case of multiphase motion by mean

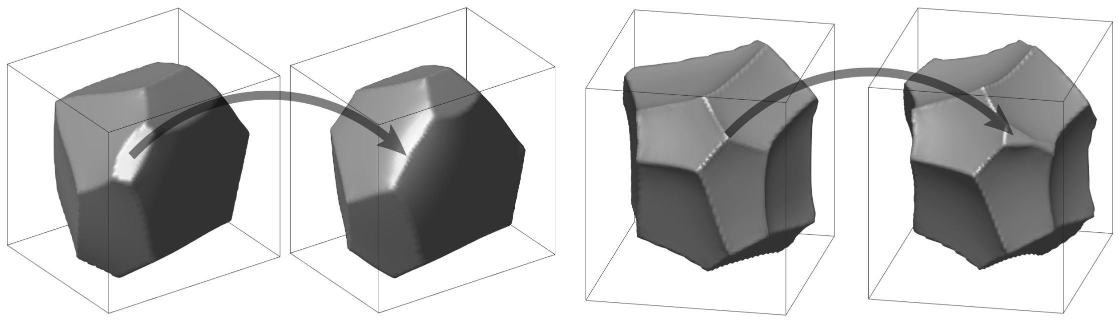

Figure 5.5: Two grains undergoing topological changes on the face of the grains. In the first grain, four quadruple points collide. In the second grain, a quadruple point splits.

curvature. In addition, we proposed a correction to the method that fixes the non-convergence while maintaining the simplicity and the spirit of the method; indeed, the new algorithm reduces to the original in the case of equal surface tensions. The correction is in the spirit of the projection method [38] of Ruuth proposed in the context of threshold dynamics. We subjected the new algorithm to the same rigorous numerical convergence studies as the original, verifying that the non-convergence of the latter is rectified. As in [38], the new algorithm is somewhat heuristic in the handling of higher order junctions (≥ 4) – but numerical evidence is presented that suggests it behaves reasonably even in their presence. Nevertheless, a more systematic approach, perhaps a variational interpretation of the VIIM in the spirit of [15], would be far preferable, not least as a more reliable extension to arbitrary junctions. Highlighting this need for further investigation of the VIIM is perhaps the most notable contribution of the present study.

CHAPTER VI

Conclusion

Over the course of this thesis, we have developed high order, stable multistage methods for gradient flows. Specifically, we have developed new second and third order, unconditionally energy stable DIRK methods, second and third order, conditionally energy stable ARK IMEX methods, and energy stable variants of the aforementioned methods in cases of solution dependent inner products. We demonstrated the methods on a variety of gradient flows, including the Allen-Cahn and Cahn-Hillard equations.

However, we only began to develop the mathematical theory of our methods for gradient flows. First, the coefficients of the multi-step schemes were found by a brute force search. For this reason, we were only able to present methods that were second and third order. Further work is needed to prove the existence of higher order versions of the multistage schemes and develop a systematic method of finding coefficients that satisfy consistency and stability. Additionally, the variant of gradient flow with solution dependent inner products introduced a new constraint on the time step for stability in order to achieve third order accuracy. Though the constraint is mild, a version of third order gradient flows for solution dependent inner products that introduces no additional constraint would be beneficial. Additionally, the examples that we used our methods on were relatively simple. The power of the methods we developed is their ability to use efficient black-box stable implementation of existing schemes to solve gradient flows and to painlessly jack up the order without sacrificing stability. Future work could focus on applications of our Runge-Kutta schemes and the solution dependent inner product variant.

In chapter IV, we showed that our stability results for Runge-Kutta schemes could be applied to an energy stable threshold dynamics scheme for simulating two phase mean curvature flow, albeit only in two dimensions. We also presented a second order threshold dynamic scheme in three dimensions with accompanying rigorous consistency calculations. An energy stable version of threshold dynamics that is second order in any dimension is lacking and could be a goal of future work. However, an even more important aim is to increase the order of accuracy of threshold dynamics in the multi-phase case. Threshold dynamics, as well as other level-set like algorithms for mean curvature motion, are only half order in the presence of more than two phases. Our work was a step in the development of a first (and higher) order threshold dynamic algorithm for multi-phase mean curvature flow.

Our focus on threshold dynamic methods was spurred by the fact that they are provably convergent in cases of unequal and anisotropic surface tension. On the other hand, we also considered the Voronoi Implicit Interface Method (VIIM). We provided rigorous numerical convergence studies for the VIIM and showed that the VIIM does not converge in the case of unequal surface tension. We then gave a variant, the DMIIM, that does converge to the physically correct solution and allows for some control over mobilities. However, the DMIIM sacrifices the efficiency of the VIIM. Even better would be a deeper mathematical understanding of the VIIM, including a variational formulation of the algorithm. This would allow the VIIM to be extended to more general cases of unequal surface tension and mobilities.

Overall, we have begun to tap into the rich mathematical theory of high order energy stable methods for general gradient flows, and mean curvature motion in particular.

APPENDICES

APPENDIX A

The Parameters that Render the Six-Step DIRK Scheme to be Third Order Accurate and Unconditionally Stable

We record here the exact values for the coefficients γ in the six-stage, third order accurate scheme introduced in section 2.4. They are rational numbers, but the irreducible fraction representation of some of them are quite long, and were therefore approximated above. With the universal, exact values given below, we can rigorously state that the new scheme introduced in this chapter can be used to jack up the order of accuracy in time of any backward Euler scheme (2.3) for gradient flows (2.1) to third order while maintaining unconditional energy stability.

The matrix of values is:

$$\gamma = \begin{pmatrix} \frac{67}{6} & 0 & 0 & 0 & 0 & 0 \\ -\frac{15}{2} & \frac{136}{7} & 0 & 0 & 0 & 0 \\ -\frac{21}{20} & -\frac{19}{4} & \frac{587}{42} & 0 & 0 & 0 \\ \frac{9}{5} & \frac{1}{21} & -\frac{47}{6} & \frac{69}{5} & 0 & 0 \\ \frac{31}{5} & -\frac{43}{6} & -\frac{4}{3} & \frac{13}{8} & \frac{242}{21} & 0 \\ -\frac{17}{6} & \frac{75}{16} & \gamma_{6,2} & \gamma_{6,3} & \gamma_{6,4} & \gamma_{6,5} \end{pmatrix}$$

where

$$\gamma_{6,2} = -\frac{968777683055918832164652607383223819953313438067 20345}{394175147873409241984526798239894762661497445562 95712}$$

$$\gamma_{6,3} = -\frac{9106775009032501797158777769188004800381259705116 73389}{788350295746818483969053596479789525322994891125 91424}$$

$$\gamma_{6,4} = \frac{298541672624278412218920487622549395057567998989 9779}{446910598495928845787445349478338733176300958688 160}$$

$$\gamma_{6,5} = \frac{523180952458721016795516949849623944572931703979 520653}{437972386526010268871696442488771958512774939514 39680}$$

It can be checked that these γ's satisfy the inequalities in the hypothesis of theorem II.1 for stability, and the consistency equations in claim II.4 for third order exactly. Code for doing so can be found at `https://github.com/AZaitzeff/gradientflow`.

APPENDIX B

A Third Order Fully Implicit Method for Gradient Flows with Solution Dependant Inner Product

Here we layout a fully implicit, third order algorithm for gradient flows with solution dependant inner product,

$$u_t = -\mathcal{L}(u)\nabla E(u).$$

In this case, each substep has form:

$$\left[\sum_{i=0}^{m-1} \gamma_{m,i}\right] U_m + k\mathcal{L}(u_*)\nabla_H E(U_m) = +\sum_{i=0}^{m-1} \gamma_{m,i} U_i. \tag{B.1}$$

The algorithm is energy stable as long as

$$\mathcal{L}(u) - \frac{1}{72}k^2 D^2\mathcal{L}(u)(w,w)$$

is positive definite for all u and w.

Fix a time step size $k > 0$. Set $u_n = u_0$. For convenience, we will denote $D^2\mathcal{L}(u_*)\big(\mathcal{L}(u_*)\nabla E(u_*), \mathcal{L}(u_*)\nabla E(u_*)\big)$ as $D^2\mathcal{L}(u_*)$.

Alternate the following steps:

1. Find u_{*_1}:

$$u_{*_1} + \frac{1}{6}k\mathcal{L}_n\nabla E(u_{*_1}) = u_n.$$

2. Find $\bar{u}$ using (B.1) with coefficients (2.29), $\mathcal{L}(u_{*_1}) - \frac{1}{72}D^2\mathcal{L}(u_{*_1})$ and time step $\frac{1}{2}k$.

3. Find $u_{*_{2,1}}$:

$$u_{*_{2,1}} + \frac{2}{5}k\mathcal{L}_n \nabla E(u_{*_{2,1}}) = u_n.$$

4. Find $u_{*_{2,2}}$ using (B.1) with the following coefficients

$$\begin{pmatrix} 6.17 & 0 & 0 & 0 \\ -0.5 & 6 & 0 & 0 \\ -3 & 2 & 7 & 0 \\ -3.1 & 0 & 2.23 & 7.40 \end{pmatrix}, \tag{B.2}$$

$\mathcal{L}(u_{*_{2,1}})$ and time step $\frac{5}{6}k$.

5. Find u_{n+1} using (B.1) starting at $\bar{u}$ (instead of u_n) with coefficients (2.29), $\mathcal{L}(u_{*_{2,2}}) - \frac{1}{72}D^2\mathcal{L}(u_{*_{2,2}})$ and time step $\frac{1}{2}k$.

The exact values for (B.2) can be found at `https://github.com/AZaitzeff/SIgradflow`.

APPENDIX C

Taylor Expansion of Characteristic Function with a Gaussian Kernel

In this appendix, we work out the Taylor expansion of the convolution of a Gaussian kernel with a characteristic function. A simpler version of the following calculation is worked out in two dimensions by Ruuth [37] and up to first order in arbitrary dimensions by Grzhibovskis & Heinz [20]. First, let us introduce the following notation for the 1D Gaussian for convenience,

$$g_t(x) = \frac{1}{2\sqrt{\pi t}} \exp\left[-\frac{x^2}{4t}\right] \text{ and let } G_t(x,y,z) = g_t(x)g_t(y)g_t(z).$$

Now take a function $h(x,y)$ with the following properties:

$$\begin{aligned} h(0,0) &= \mathcal{O}(t) \\ h_x(0,0) &= \mathcal{O}(t) \\ h_y(0,0) &= \mathcal{O}(t) \end{aligned} \tag{C.1}$$

The papers mentioned above ([20, 37]) use the assumption that

$$h(0,0) = h_x(0,0) = h_y(0,0) = 0. \tag{C.2}$$

These simpler assumptions are sufficient for their calculation of the Taylor expansion for a single step of threshold dynamics. However, after the first stage, the interface may no longer satisfy (C.2). So we required the more general conditions given by (C.1) for Taylor expansions of the interface after the first stage.

Now let $\Sigma = \{(x,y,z) : z \leq h(x,y)\}$. The goal is to see how $\left[G_t * \mathbb{1}_\Sigma\right](x,y,z)$ behaves along the z-axis near the origin. First, we simplify $\left[G_t * \mathbb{1}_\Sigma\right](x,y,z)$:

$$\begin{aligned}
&\left[G_t * \mathbb{1}_\Sigma\right](x,y,z)\\
&= \int_{\mathbb{R}^2} g_t(y-\tilde{y})g_t(x-\tilde{x}) \int_{-\infty}^{\infty} g_t(z-\tilde{z})\mathbb{1}_\Sigma(\tilde{x},\tilde{y},\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&= \int_{\mathbb{R}^2} g_t(y-\tilde{y})g_t(x-\tilde{x}) \int_{-\infty}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&= \int_{\mathbb{R}^2} g_t(y-\tilde{y})g_t(x-\tilde{x}) \int_{-\infty}^{0} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&\quad + \int_{\mathbb{R}^2} g_t(y-\tilde{y})g_t(x-\tilde{x}) \int_{0}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&= \frac{1}{2} - \frac{z}{2\sqrt{\pi t}} + \frac{z^3}{24\sqrt{\pi}t^{3/2}} + \int_{\mathbb{R}^2} g_t(x-\tilde{x})g_t(y-\tilde{y}) \int_{0}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&\quad + \text{h.o.t.}
\end{aligned} \tag{C.3}$$

Setting $x = y = 0$, we will now simplify the last term of (C.3),

$$\int_{\mathbb{R}^2} g_t(x-\tilde{x})g_t(y-\tilde{y}) \int_{0}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}. \tag{C.4}$$

First note that, near $z = 0$,

$$\int_{0}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z} = \frac{1}{2\sqrt{\pi t}} \int_{0}^{h(\tilde{x},\tilde{y})} 1 - \frac{(z-\tilde{z})^2}{4t} d\tilde{z} + \text{h.o.t.} \tag{C.5}$$

Substituting approximation (C.5) into (C.4) and integrating,

$$\begin{aligned}
&\frac{1}{2\sqrt{\pi t}} \int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y}) \int_{0}^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y}\\
&= \frac{1}{2\sqrt{\pi t}} \int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y}) \int_{0}^{h(\tilde{x},\tilde{y})} 1 - \frac{(z-\tilde{z})^2}{4t} d\tilde{z}d\tilde{x}d\tilde{y} + \text{h.o.t.}\\
&= \frac{1}{2\sqrt{\pi t}} \int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y}) \left[h(\tilde{x},\tilde{y}) + \frac{-3h(\tilde{x},\tilde{y})z^2 + 3(h(\tilde{x},\tilde{y}))^2 z - (h(\tilde{x},\tilde{y}))^3}{12t}\right] d\tilde{z}d\tilde{x}d\tilde{y} + \text{h.o.t.}
\end{aligned} \tag{C.6}$$

Denote the Taylor expansion of $h(\tilde{x},\tilde{y})$ around $(0,0)$ as $P[ht](\tilde{x},\tilde{y})$:

$$\begin{aligned}
P[ht](\tilde{x},\tilde{y}) &= h + \tilde{x}h_x + \tilde{y}h_y + \frac{\tilde{x}^2}{2}h_{xx} + \tilde{x}\tilde{y}h_{xy} + \frac{\tilde{y}^2}{2}h_{yy} + \frac{\tilde{x}^3}{6}h_{xxx} + \frac{\tilde{x}^2\tilde{y}}{2}h_{xxy} + \frac{\tilde{x}\tilde{y}^2}{2}h_{xyy}\\
&\quad + \frac{\tilde{y}^3}{2}h_{yyy} + \frac{\tilde{x}^4}{24}h_{xxxx} + \frac{\tilde{x}^3\tilde{y}}{6}h_{xxxy} + \frac{\tilde{x}^2\tilde{y}^2}{4}h_{xxyy} + \frac{\tilde{x}\tilde{y}^3}{6}h_{xyyy} + \frac{\tilde{y}^4}{24}h_{yyyy} + \text{h.o.t.}
\end{aligned} \tag{C.7}$$

Where $h = h(0,0)$, $h_x = h_x(0,0)$, etc. in order to simplify notation. Now substitute the Taylor expansion of $h(\tilde{x},\tilde{y})$ about $(0,0)$ into (C.6):

$$\text{(C.8)}\quad \frac{1}{2\sqrt{\pi t}}\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y} = \\ \frac{1}{2\sqrt{\pi t}}\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\Bigg[P[ht](\tilde{x},\tilde{y}) - \frac{1}{4t}P[ht](\tilde{x},\tilde{y})z^2 + \frac{1}{4t}\big(P[ht](\tilde{x},\tilde{y})\big)^2 z \\ - \frac{1}{12t}\big(P[ht](\tilde{x},\tilde{y})\big)^3\Bigg]d\tilde{x}d\tilde{y} + \text{h.o.t.}$$

Now we can integrate (C.8). For a non-negative integer n we have:

$$\text{(C.9)}\quad \int_{-\infty}^{\infty} x^n g_t(x)dx = \begin{cases} \frac{(2n)!}{n!}t^{n/2} & x \text{ for } n \text{ even} \\ 0 & \text{for } n \text{ odd} \end{cases}$$

Using (C.9), we simplify (C.8)

$$\text{(C.10)}\quad \frac{1}{2\sqrt{\pi t}}\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})d\tilde{z}d\tilde{x}d\tilde{y} = \\ \frac{h}{2\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(h_{xx}+h_{yy}) \\ + \frac{t^{3/2}}{4\sqrt{\pi}}(h_{xxxx}+2h_{xxyy}+h_{yyyy}) - \frac{z^2}{8\sqrt{\pi}t^{3/2}}h - \frac{z^2}{8\sqrt{\pi t}}(h_{xx}+h_{yy}) + \frac{z}{8\sqrt{\pi}t^{3/2}}h^2 \\ + \frac{z}{4\sqrt{\pi t}}h(h_{xx}+h_{yy}) + \frac{z\sqrt{t}}{2\sqrt{\pi}}(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) - \frac{h^3}{24\sqrt{\pi}t^{3/2}} \\ - \frac{h^2}{8\sqrt{\pi t}}(h_{xx}+h_{yy}) - \frac{\sqrt{t}}{2\sqrt{\pi}}h(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) \\ - \frac{t^{3/2}}{2\sqrt{\pi}}(\frac{5}{4}h_{xx}^3 + \frac{5}{4}h_{yy}^3 + \frac{3}{4}h_{xx}h_{yy}^2 + \frac{3}{4}h_{xx}^2h_{yy} + 3h_{xx}h_{xy}^2 + 3h_{yy}h_{xy}^2) + \text{h.o.t.}$$

Now substituting (C.10) for the last term of (C.3) we arrive at the expansion of $\Big[G_t * \mathbb{1}_\Sigma\Big](0,0,z)$ near $z=0$:

$$\text{(C.11)}\quad \Big[G_t * \mathbb{1}_\Sigma\Big](0,0,z) = \\ \frac{1}{2} - \frac{z}{2\sqrt{\pi t}} + \frac{z^3}{24\sqrt{\pi}t^{3/2}} + \frac{h}{2\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(h_{xx}+h_{yy}) \\ + \frac{t^{3/2}}{4\sqrt{\pi}}(h_{xxxx}+2h_{xxyy}+h_{yyyy}) - \frac{z^2}{8\sqrt{\pi}t^{3/2}}h - \frac{z^2}{8\sqrt{\pi t}}(h_{xx}+h_{yy}) + \frac{z}{8\sqrt{\pi}t^{3/2}}h^2 \\ + \frac{z}{4\sqrt{\pi t}}h(h_{xx}+h_{yy}) + \frac{z\sqrt{t}}{2\sqrt{\pi}}(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) - \frac{h^3}{24\sqrt{\pi}t^{3/2}} \\ - \frac{h^2}{8\sqrt{\pi t}}(h_{xx}+h_{yy}) - \frac{\sqrt{t}}{2\sqrt{\pi}}h(\frac{3}{4}h_{xx}^2 + \frac{3}{4}h_{yy}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2) \\ - \frac{t^{3/2}}{2\sqrt{\pi}}(\frac{5}{4}h_{xx}^3 + \frac{5}{4}h_{yy}^3 + \frac{3}{4}h_{xx}h_{yy}^2 + \frac{3}{4}h_{xx}^2h_{yy} + 3h_{xx}h_{xy}^2 + 3h_{yy}h_{xy}^2) + \text{h.o.t}$$

In chapter IV, we use the previous calculation to find the location of an interface along the z-axis after thresholding, i.e. finding z such that $\Big[G_t * \mathbb{1}_\Sigma\Big](0,0,z) = \frac{1}{2}$.

We also need to find how the derivatives of our interface, given by $z_x(0,0)$, $z_y(0,0)$, $z_{xx}(0,0)$ etc., relate to the derivatives of the original interface given by $h(x,y)$. The derivatives of z match the derivatives of h to order t:

$$\frac{\partial^{m+n}}{\partial x^n y^m} z(x,y)|_{(x,y)=(0,0)} = \frac{\partial^{m+n}}{\partial x^n y^m} h(x,y)|_{(x,y)=(0,0)} + O(t). \tag{C.12}$$

Note that $z_x(0,0)$ and $z_y(0,0)$ are of $\mathcal{O}(t)$ for h having properties (C.1). For our calculations, we also need to find $z_{xx}(0,0)$ and $z_{yy}(0,0)$ to $\mathcal{O}(t^2)$. So we also include the calculation of $\frac{\partial^2}{\partial x^2}\left[G_t * \mathbb{1}_\Sigma\right](x,y,z(x,y))|_{(x,y)=(0,0)}$:

$$\begin{aligned}
&\frac{\partial^2}{\partial x^2}\left[G_t * \mathbb{1}_\Sigma\right](x,y,z(x,y))|_{(x,y)=(0,0)}\\
&= -\frac{z_{xx}}{2\sqrt{\pi t}} + \frac{3z^2 z_{xx} + 6z(z_x)^2}{24\sqrt{\pi}t^{3/2}} + \int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})\left[\frac{\tilde{x}^2-2t}{4t^2} + \frac{\tilde{x}(\tilde{z}-z)z_x}{2t^2}\right.\\
&\quad\left. + \frac{(\tilde{z}-z)^2 z_x^2}{4t^2} + \frac{(\tilde{z}-z)z_{xx}}{2t} - \frac{z_x^2}{2t}\right] d\tilde{z}d\tilde{x}d\tilde{y} + \text{h.o.t.}
\end{aligned} \tag{C.13}$$

The terms $\frac{z(z_x)^2}{4\sqrt{\pi}t^{3/2}}$ and

$$\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})\left[\frac{\tilde{x}(\tilde{z}-z)z_x}{2t^2} + \frac{(\tilde{z}-z)^2 z_x^2}{4t^2} - \frac{z_x^2}{2t}\right] d\tilde{z}d\tilde{x}d\tilde{y}$$

turn out to be $\mathcal{O}(t^{3/2})$, which is higher than the order needed for the calculations in this chapter. We will simplify the two remaining terms in the integrand, starting with the term $\frac{\tilde{x}^2-2t}{4t^2}$. As in the previous calculation substitute in the approximation (C.5) and integrate:

$$\begin{aligned}
&\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y})\int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})\frac{\tilde{x}^2-2t}{4t^2} d\tilde{z}d\tilde{x}d\tilde{y}\\
&= \frac{1}{2\sqrt{\pi t}}\int_{\mathbb{R}^2} g_t(\tilde{y})g_t(\tilde{x})\left[P[ht](\tilde{x},\tilde{y}) - \frac{1}{4t}P[ht](\tilde{x},\tilde{y})z^2\right.\\
&\quad\left. + \frac{1}{4t}\left(P[ht](\tilde{x},\tilde{y})\right)^2 z - \frac{1}{12t}\left(P[ht](\tilde{x},\tilde{y})\right)^3\right]\frac{\tilde{x}^2-2t}{4t^2} d\tilde{x}d\tilde{y} + \text{h.o.t.}
\end{aligned} \tag{C.14}$$

Then use (C.9) to further simplify (C.14):

$$
\begin{aligned}
&= \frac{h_{xx}}{2\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(h_{xxxx} + h_{xxyy}) \\
&- \frac{z^2}{8\sqrt{\pi}t^{3/2}}h_{xx} + \frac{z}{4\sqrt{\pi}t^{3/2}}hh_{xx} + \frac{z}{2\sqrt{\pi t}}\left(\frac{3}{2}h_{xx}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2\right) \\
&- \frac{h^2}{8\sqrt{\pi}t^{3/2}}h_{xx} - \frac{h}{2\sqrt{\pi t}}\left(\frac{3}{2}h_{xx}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2\right) \\
&- \frac{\sqrt{t}}{2\sqrt{\pi}}\left(\frac{15}{4}h_{xx}^3 + \frac{3}{4}h_{xx}h_{yy}^2 + \frac{3}{2}h_{xx}^2h_{yy} + 6h_{xx}h_{xy}^2 + 3h_{yy}h_{xy}^2\right) + \text{h.o.t}
\end{aligned}
\tag{C.15}
$$

We now turn to the $\frac{(\tilde{z}-z)z_{xx}}{2t}$ term, following the same steps:

$$
\begin{aligned}
&\int_{\mathbb{R}^2} g_t(\tilde{x})g_t(\tilde{y}) \int_0^{h(\tilde{x},\tilde{y})} g_t(z-\tilde{z})\frac{(\tilde{z}-z)z_{xx}}{2t}d\tilde{z}d\tilde{x}d\tilde{y} \\
=&\frac{1}{2\sqrt{\pi t}}\int_{\mathbb{R}^2} g_t(\tilde{y})g_t(\tilde{x})\Bigg[P[ht](\tilde{x},\tilde{y}) - \frac{1}{4t}P[ht](\tilde{x},\tilde{y})z^2 \\
&+ \frac{1}{4t}\big(P[ht](\tilde{x},\tilde{y})\big)^2 z - \frac{1}{12t}\big(P[ht](\tilde{x},\tilde{y})\big)^3\Bigg]\frac{(\tilde{z}-z)z_{xx}}{2t}d\tilde{x}d\tilde{y} + \text{h.o.t.} \\
=&\frac{z_{xx}}{2\sqrt{\pi t}}\left[\frac{h^2}{4t} + \frac{1}{2}h(h_{xx}+h_{yy}) + \frac{3t}{4}(h_{xx}^2 + h_{yy}^2) + \frac{t}{2}h_{xx}h_{yy} + th_{xy}^2 - \frac{zh}{2t} - \frac{z}{2}(h_{xx}+h_{yy})\right] \\
&+ \text{h.o.t.}
\end{aligned}
\tag{C.16}
$$

Substituting (C.15) and (C.16) into (C.13), we arrive at the simplification

$$
\begin{aligned}
&\frac{\partial^2}{\partial x^2}\Big[G_t * \mathbb{1}_\Sigma\Big](x,y,z(x,y))|_{(x,y)=(0,0)} \\
&= -\frac{z_{xx}}{2\sqrt{\pi t}} + \frac{z^2 z_{xx}}{8\sqrt{\pi}t^{3/2}} + \frac{h_{xx}}{2\sqrt{\pi t}} + \frac{\sqrt{t}}{2\sqrt{\pi}}(h_{xxxx} + h_{xxyy}) \\
&- \frac{z^2}{8\sqrt{\pi}t^{3/2}}h_{xx} + \frac{z}{4\sqrt{\pi}t^{3/2}}hh_{xx} + \frac{z}{2\sqrt{\pi t}}\left(\frac{3}{2}h_{xx}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2\right) \\
&- \frac{h^2}{8\sqrt{\pi}t^{3/2}}h_{xx} - \frac{h}{2\sqrt{\pi}\sqrt{t}}\left(\frac{3}{2}h_{xx}^2 + \frac{1}{2}h_{xx}h_{yy} + h_{xy}^2\right) \\
&- \frac{\sqrt{t}}{2\sqrt{\pi}}\left(\frac{15}{4}h_{xx}^3 + \frac{3}{4}h_{xx}h_{yy}^2 + \frac{3}{2}h_{xx}^2h_{yy} + 6h_{xx}h_{xy}^2 + 3h_{yy}h_{xy}^2\right) \\
&+ \frac{z_{xx}}{2\sqrt{\pi t}}\Bigg[\frac{h^2}{4t} + \frac{1}{2}h(h_{xx}+h_{yy}) + \frac{3t}{4}(h_{xx}^2 + h_{yy}^2) + \frac{t}{2}h_{xx}h_{yy} + th_{xy}^2 \\
&- \frac{zh}{2t} - \frac{z}{2}(h_{xx}+h_{yy})\Bigg] + \text{h.o.t.}
\end{aligned}
\tag{C.17}
$$

Finding $\frac{\partial^2}{\partial y^2}\Big[G_t * \mathbb{1}_\Sigma\Big](x,y,z(x,y))|_{(x,y)=(0,0)}$ is similar. We use both (C.11) and (C.17) in several calculations throughout the course of chapter IV.

APPENDIX D

The Parameters that Render the Four-Step Threshold Dynamic Scheme to be Second Order Accurate

We record here the exact values for the coefficients γ in the four-stage, second order accurate scheme introduced in section 4.5.3. They are algebraic numbers, but the representations of some of them are quite long and therefore we have approximated them above. With the exact values given below, we can rigorously state that the algorithm 10 is second order while maintaining unconditional energy stability. The matrix of values is:

$$\gamma = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{1}{4} & \frac{5}{4} & 0 & 0 \\ \frac{5}{6} & -\frac{2}{3} & \frac{5}{6} & 0 \\ \gamma_{4,0} & \frac{1}{2} & \gamma_{4,2} & \gamma_{4,3} \end{pmatrix}.$$

$$\begin{aligned}\gamma_{4,2} =& \Big(\sqrt[3]{735478578874058654996000064\sqrt{1334953188776447143443 77} - 2347474537124305956620735764885584867 1} \\ & - \frac{5551049511730043591353151}{\sqrt[3]{735478578874058654996000064\sqrt{133495318877644714344377} - 23474745371243059566207357648855848671}} \\ & - 456109196575\Big)/3627134098848\end{aligned}$$

$$\begin{aligned}\gamma_{4,3} =& \Big(- 5586815667458 \\ & \times \sqrt[3]{735478578874058654996000064\sqrt{133495318877644714344377} - 23474745371243059566207357648855848671} \\ & + \left(735478578874058654996000064\sqrt{133495318877644714344377} - 23474745371243059566207357648855848671\right)^{2/3} \\ & + \frac{3101269038296848848713708970145646015 8}{\sqrt[3]{735478578874058654996000064\sqrt{133495318877644714344377} - 23474745371243059566207357648855848671}} \\ & + \frac{30814150681678355363112149018994128529535197628801}{\left(735478578874058654996000064\sqrt{133495318877644714344377} - 23474745371243059566207357648855848671\right)^{2/3}} \\ & + 7974522440634228925392639\Big)/18386964471851466374900016\end{aligned}$$

$$\gamma_{4,0} = 1 - \frac{1}{2} - \gamma_{4,2} - \gamma_{4,3}$$

It can be checked that these γs satisfy the inequalities in the hypothesis of theorem IV.2 for stability, and the consistency equations in claim IV.1 for second order exactly.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Uri M Ascher, Steven J Ruuth, and Raymond J Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25(2-3):151–167, 1997.

[2] Rick K Beatson. On the convergence of some cubic spline interpolation schemes. *SIAM journal on numerical analysis*, 23(4):903–912, 1986.

[3] Lia Bronsard and Brian TR Wetton. A numerical method for tracking curve networks moving with curvature motion. *Journal of Computational Physics*, 120(1):66–87, 1995.

[4] Kevin Burrage. Efficiently implementable algebraically stable runge–kutta methods. *SIAM Journal on Numerical Analysis*, 19(2):245–258, 1982.

[5] John C Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, 2016.

[6] John Charles Butcher. A stability property of implicit Runge-Kutta methods. *BIT Numerical Mathematics*, 15(4):358–361, 1975.

[7] Long Qing Chen and Jie Shen. Applications of semi-implicit Fourier-spectral method to phase field equations. *Computer Physics Communications*, 108(2-3):147–158, 1998.

[8] Wenbin Chen, Cheng Wang, Xiaoming Wang, and Steven M Wise. Positivity-preserving, energy stable numerical schemes for the Cahn-Hilliard equation with logarithmic potential. *Journal of Computational Physics: X*, 3:100031, 2019.

[9] Carl De Boor. *A Practical Guide to Splines*. Applied mathematical sciences. Springer, Berlin, 2001.

[10] Felix Del Teso, Jørgen Endal, and Espen R Jakobsen. Robust numerical methods for nonlocal (and local) equations of porous medium type. part ii: Schemes and experiments. *SIAM Journal on Numerical Analysis*, 56(6):3611–3647, 2018.

[11] Chenghua Duan, Chun Liu, Cheng Wang, and Xingye Yue. Numerical methods for porous medium equation by an energetic variational approach. *Journal of Computational Physics*, 385:13–32, 2019.

[12] M. Elsey, S. Esedoḡlu, and P. Smereka. Simulations of anisotropic grain growth: Efficient algorithms and misorientation distributions. *Acta Materialia*, 61(6):2033–2043, 2013.

[13] Matt Elsey, Selim Esedoḡlu, Peter Smereka, et al. Diffusion generated motion for grain growth in two and three dimensions. *Journal of Computational Physics*, 228(21):8015–8033, 2009.

[14] Selim Esedoḡlu, Matt Jacobs, and Pengbo Zhang. Kernels with prescribed surface tension & mobility for threshold dynamics schemes. *Journal of Computational Physics*, 337:62–83, 2017.

[15] Selim Esedoḡlu and Felix Otto. Threshold dynamics for networks with arbitrary surface tensions. *Communications on Pure and Applied Mathematics*, 68(5):808–864, 5 2015.

[16] Selim Esedoḡlu, Steven Ruuth, Richard Tsai, et al. Diffusion generated motion using signed distance functions. *Journal of Computational Physics*, 229(4):1017–1042, 2010.

[17] Selim Esodoḡlu, Peter Smereka, et al. A variational formulation for a level set representation of multiphase flow and area preserving curvature flow. *Communications in Mathematical Sciences*, 6(1):125–148, 2008.

[18] David J Eyre. Unconditionally gradient stable time marching the Cahn-Hilliard equation. *MRS Online Proceedings Library Archive*, 529, 1998.

[19] Harald Garcke, Britta Nestler, and Barbara Stoth. A multiphase field concept: numerical simulations of moving phase boundaries and multiple junctions. *SIAM Journal on Applied Mathematics*, 60(1):295–315, 1999.

[20] Richards Grzhibovskis and Alexei Heintz. A convolution-thresholding approximation of generalized curvature flows. *SIAM journal on numerical analysis*, 42(6):2652–2670, 2005.

[21] Richards Grzhibovskis and Alexei Heintz. A convolution thresholding scheme for the Willmore flow. *Interfaces and Free Boundaries*, 10(2):139–153, 2008.

[22] Daozhi Han and Xiaoming Wang. A second order in time, uniquely solvable, unconditionally stable numerical scheme for Cahn–Hilliard–Navier–Stokes equation. *Journal of Computational Physics*, 290:139–156, 2015.

[23] Conyers Herring. Surface tension as a motivation for sintering. In *Fundamental Contributions to the Continuum Theory of Evolving Phase Interfaces in Solids*, pages 33–69. Springer, 1999.

[24] Naoyuki Ishimura. Curvature evolution of plane curves with prescribed opening angle. *Bulletin of the Australian Mathematical Society*, 52(2):287–296, 1995.

[25] Richard Jordan, David Kinderlehrer, and Felix Otto. The variational formulation of the Fokker–Planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

[26] Christopher A Kennedy and Mark H Carpenter. Additive Runge-Kutta schemes for convection-diffusion-reaction equations. *Applied Numerical Mathematics*, 44(1-2):139–181, 2003.

[27] Christopher A Kennedy and Mark H Carpenter. Diagonally implicit Runge-Kutta methods for ordinary differential equations. *A Review. NASA Report. Langley research center. Hampton VA*, 23681:162, 2016.

[28] Tim Laux and Felix Otto. Convergence of the thresholding scheme for multi-phase mean-curvature flow. *Calc. Var. Partial Differential Equations*, 55(5):Art. 129, 74, 2016.

[29] Barry Merriman, James K. Bence, and Stanley J. Osher. Diffusion generated motion by mean curvature. *The Computational Crystal Growers. AMS Selection in Math.*, pages 73–83, 1992.

[30] Barry Merriman, James K. Bence, and Stanley J. Osher. Motion of multiple junctions: a level set approach. *J. Comput. Phys.*, 112(2):334–363, 1994.

[31] Luciano Modica and Stefano Mortola. Un esempio di γ-convergenza. *Boll. Un. Mat. Ital. B*, 14:285–299, 1977.

[32] William W Mullins. Two-dimensional motion of idealized grain boundaries. *Journal of Applied Physics*, 27(8):900–904, 1956.

[33] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

[34] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988.

[35] Lorenzo Pareschi and Giovanni Russo. Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxation. *Journal of Scientific Computing*, 25(1):129–155, Oct 2005.

[36] Fernando Reitich and H Mete Soner. Three-phase boundary motions under constant velocities. i: The vanishing surface tension limit. *Proceedings of the Royal Society of Edinburgh Section A: Mathematics*, 126(4):837–865, 1996.

[37] Steven J. Ruuth. *Efficient algorithms for diffusion-generated motion by mean curvature*. PhD thesis, University of British Columbia, 1996.

[38] Steven J. Ruuth. Efficient algorithms for diffusion-generated motion by mean curvature. *J. Comput. Phys.*, 144(2):603–625, 1998.

[39] Robert I Saye and James A Sethian. The Voronoi implicit interface method for computing multiphase physics. *Proceedings of the National Academy of Sciences*, 108(49):19498–19503, 2011.

[40] Robert I Saye and James A Sethian. Analysis and applications of the Voronoi implicit interface method. *Journal of Computational Physics*, 231(18):6051–6085, 2012.

[41] James A Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 1999.

[42] Jie Shen, Jie Xu, and Jiang Yang. The scalar auxiliary variable (sav) approach for gradient flows. *Journal of Computational Physics*, 353:407–416, 2018.

[43] Jie Shen and Xiaofeng Yang. Numerical approximations of Allen-Cahn and Cahn-Hilliard equations. *Discrete Contin. Dyn. Syst*, 28(4):1669–1691, 2010.

[44] Jaemin Shin, Hyun Geun Lee, and June-Yub Lee. Unconditionally stable methods for gradient flow using convex splitting Runge–Kutta scheme. *Journal of Computational Physics*, 347:367–381, 2017.

[45] Jaemin Shin and June-Yub Lee. An energy stable Runge–Kutta method for convex gradient problems. *Journal of Computational and Applied Mathematics*, 367:112455, 2020.

[46] Kurt A Smith, Francisco J Solis, and David Chopp. A projection method for motion of triple junctions by level sets. *Interfaces and free boundaries*, 4(3):263–276, 2002.

[47] Michael Westdickenberg and Jon Wilkening. Variational particle schemes for the porous medium equation and for the system of isentropic euler equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 44(1):133–166, 2010.

[48] Hong-Kai Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *J. Comput. Phys.*, 127(1):179–195, 1996.

[49] Evgeniy Zharovsky, Adrian Sandu, and Hong Zhang. A class of implicit-explicit two-step Runge–Kutta methods. *SIAM Journal on Numerical Analysis*, 53(1):321–341, 2015.