# Linearly Stabilized Schemes for the Time Integration of Stiff Nonlinear PDEs

by

**Kevin Chow**

B.Math., University of Waterloo, 2014

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Mathematics
Faculty of Science


**SIMON FRASER UNIVERSITY**
**Fall 2016**

# Approval

| | |
|---|---|
| **Name:** | **Kevin Chow** |
| **Degree:** | **Master of Science (Mathematics)** |
| **Title:** | ***Linearly Stabilized Schemes for the Time Integration of Stiff Nonlinear PDEs*** |
| **Examining Committee:** | **Chair:** Dr. Weiran Sun<br>Assistant Professor |

**Dr. Steven Ruuth**
Senior Supervisor
Professor _____

**Dr. Robert Russell**
Supervisor
Professor Emeritus _____

**Dr. Ben Adcock**
Internal/External Examiner
Assistant Professor _____

**Date Defended:** 9 December 2016

# Abstract

In many applications, the governing PDE to be solved numerically will contain a stiff component. When this component is linear, an implicit time stepping method, unencumbered by time step-size restrictions, is preferred. On the other hand, if the stiff component is nonlinear, the complexity and cost per step of using an implicit method is heightened, and explicit methods may be preferred for their simplicity and ease of implementation. In this thesis, we analyze new and existing linearly stabilized schemes for the purpose of integrating stiff nonlinear PDEs in time. These schemes compute the nonlinear term explicitly and, at the cost of solving a linear system with a matrix that is fixed throughout, are unconditionally stable, thus combining the advantages of explicit and implicit methods. Applications are presented to illustrate the use of these methods.

**Keywords:** Stiff PDEs; time stepping; stability; IMEX methods; exponential time differencing

# Dedication

# Acknowledgements

# Table of Contents

**Approval** ii

**Abstract** iii

**Dedication** iv

**Acknowledgements** v

**Table of Contents** vi

**List of Tables** viii

**List of Figures** ix

**1 Introduction** 1

1.1 Linear Stability Analysis . . . 2

1.1.1 Stability and the scalar test equation . . . 2

1.1.2 Stability contours . . . 3

1.1.3 Relation to stiff PDEs . . . 3

1.2 Order of Accuracy . . . 4

1.2.1 Richardson extrapolation . . . 5

1.3 Overview . . . 5

**2 Adding Zero, Unconditional Stability and a Modified Test Equation** 7

2.1 Prototype 1D Problem . . . 7

2.1.1 A first order linearly stabilized scheme . . . 8

2.1.2 A von Neumann stability analysis . . . 9

2.1.3 Second order by Richardson extrapolation . . . 11

2.2 A Modified Test Equation . . . 11

2.2.1 Forward Euler . . . 12

2.2.2 Linearly stabilized semi-implicit Euler . . . 12

2.2.3 Explicit-implicit-null . . . 12

2.3 Numerical Results . . . 13

2.3.1 Stability contours . . . 14
2.3.2 Numerical convergence test . . . 14

**3 IMEX Linear Multistep Methods** **16**
3.1 IMEX Formulas . . . 16
3.2 Analysis of the Amplification Polynomials . . . 17
3.2.1 Schur and von Neumann polynomials . . . 18
3.2.2 Amplification polynomials of second order IMEX methods . . . 18
3.2.3 Amplification polynomials of third and fourth order IMEX methods 19
3.3 Convergence of Linearly Stabilized IMEX Methods . . . 20
3.3.1 Stability contours . . . 20
3.3.2 Numerical convergence test . . . 20
3.3.3 Failure with high order IMEX methods . . . 23
3.4 Comparison of the Second Order Methods . . . 25
3.4.1 A 2D test problem . . . 25
3.4.2 Loss of accuracy with EIN . . . 26
3.4.3 Amplification factors at infinity . . . 28

**4 Higher Order with Exponential Integrators** **31**
4.1 Exponential Runge-Kutta . . . 31
4.2 Linearly stabilized ETDRK2 and ETDRK4 . . . 33
4.2.1 Stable evaluation of the matrix exponential and related functions . . 36

**5 Numerical Experiments** **37**
5.1 Image Inpainting . . . 37
5.1.1 TV inpainting . . . 39
5.1.2 TV-$H^{-1}$ inpainting . . . 40
5.2 Motion by Mean Curvature . . . 43
5.2.1 Shrinking dumbbell in 2D and 3D . . . 43
5.2.2 Anisotropic mean curvature motion . . . 46
5.3 Phase Separation . . . 49
5.3.1 Cahn-Hilliard on surfaces using the closest point method . . . 49
5.3.2 A modified Cahn-Hilliard with nonlocal interactions . . . 50

**6 Conclusion** **53**

**Bibliography** **55**

**Appendix A Code** **58**
A.1 Time integration of the Cahn-Hilliard equation using linearly stabilized time stepping schemes . . . 58

# List of Tables

Table 3.1 Amplification polynomials of second order IMEX methods . . . . . 19
Table 3.2 Amplification polynomials of third and fourth order IMEX methods . 20
Table 3.3 (Non)convergence of SBDF3 at various $p$ . . . . . 24

Table 5.1 Iteration counts for TV image restoration . . . . . 40
Table 5.2 Iteration counts for TV-$H^{-1}$ image restoration . . . . . 42

# List of Figures


Figure 1.1 Examples of stability contour plots . . . . . 3

Figure 2.1 Numerical solution to a 1D axisymmetric mean curvature motion problem . . . . . 8
Figure 2.2 Stability contours for SBDF1 at various $p$ . . . . . 14
Figure 2.3 Stability contours for EIN at various $p$ . . . . . 15
Figure 2.4 Numerical convergence study with SBDF1 and EIN . . . . . 15

Figure 3.1 Stability contours for CNAB at various $p$ . . . . . 21
Figure 3.2 Stability contours for mCNAB at various $p$ . . . . . 21
Figure 3.3 Stability contours for CNLF at various $p$ . . . . . 22
Figure 3.4 Stability contours for SBDF2 at various $p$ . . . . . 22
Figure 3.5 Stability contours for SBDF3 at various $p$ . . . . . 23
Figure 3.6 Stability contours for SBDF4 at various $p$ . . . . . 23
Figure 3.7 Numerical convergence study with IMEX methods . . . . . 24
Figure 3.8 Instabilities using linearly stabilized SBDF3 . . . . . 25
Figure 3.9 Numerical convergence study comparing second order methods . . . 27
Figure 3.10 Amplification factors as $z \to -\infty$ . . . . . 29

Figure 4.1 Stability contours for ETDRK2 at various $p$ . . . . . 33
Figure 4.2 Stability contours for ETDRK4 at various $p$ . . . . . 34
Figure 4.3 Amplification factors as $z \to -\infty$ . . . . . 35
Figure 4.4 Error constant and its dependence on $p$ . . . . . 35

Figure 5.1 Photograph of a sea turtle overwritten with text . . . . . 39
Figure 5.2 Photograph of a bullfinch vandalized by a cartoonish fox . . . . . 39
Figure 5.3 Image restoration by TV inpainting . . . . . 41
Figure 5.4 Image restoration by TV-$\mathrm{H}^{-1}$ inpainting . . . . . 42
Figure 5.5 Mean curvature flow of a dumbbell-shaped curve in 2D . . . . . 44
Figure 5.6 Convergence of linearly stabilized schemes to a dumbbell-shaped curve evolved under mean curvature flow . . . . . 44
Figure 5.7 Zoom-in over $[0.0, 0.20] \times [0.475, 0.54]$ to inspect convergence . . . . 45
Figure 5.8 A zoom-in over $[1.30, 1.41] \times [0.0, 0.30]$ to inspect convergence . . . 46

Figure 5.9 Mean curvature flow of a dumbbell-shaped curve in 3D . . . . . 47
Figure 5.10 Anisotropic mean curvature flow . . . . . 48
Figure 5.11 Cahn-Hilliard on a torus . . . . . 50
Figure 5.12 Cahn-Hilliard on a cow-shaped surface . . . . . 51
Figure 5.13 Energy minimizing patterns of the nonlocal Cahn-Hilliard . . . . . 52

# Chapter 1

# Introduction

In this thesis, we propose and analyze some new linearly stabilized schemes for the time integration of stiff nonlinear PDEs. A linearly stabilized scheme of first order has been used in a number of areas, with the first known example being from a paper by Douglas and Dupont [8] where this technique was used for the solution to a variable coefficient heat equation on rectangular domains. In subsequent years, the idea has been rediscovered by Smereka [30], and Eyre [10], who first used the name "linearly stabilized". Others have gone on to apply these schemes to Hele-Shaw flows, interface motion, image processing, and solving PDEs on surfaces [10, 24, 11, 25, 18].

In each of the references mentioned in the previous paragraph, the authors have succeeded in implementing only a first order time stepping method. Recently in [9], Duchemin and Eggers consolidated the approach and produced a second order linearly stabilized scheme they refer to as the explicit-implicit-null (EIN) method. Their method attains second order accuracy by extrapolating the first order results. Moreover, they identified that the key principle for the success of any linearly stabilized scheme is unconditional stability. Indeed, a significant section of their paper is devoted to showing that their method is unconditionally stable under only a mild condition on a parameter that is introduced.

Our derivations for new linearly stabilized schemes also begins by ensuring that the newly derived schemes are in fact unconditionally stable. The techniques we employ in our stability analysis are those of a standard linear stability analysis and are reviewed first. A brief discussion of order of accuracy and Richardson extrapolation is also included before an overview of the thesis is given.

## 1.1 Linear Stability Analysis

### 1.1.1 Stability and the scalar test equation

Linear stability analysis is a method of analysis predicated on requiring that the numerical solution replicates properties inherent in the exact solution to the test equation,

$$u' = \lambda u, \quad \lambda < 0. \tag{1.1}$$

Over one time step, the exact solution is

$$u(t^n + \Delta t) = e^{\lambda \Delta t} u(t^n). \tag{1.2}$$

Observe that, in general, the exact solution satisfies

$$\frac{\left|u(t^n + \Delta t)\right|}{\left|u(t^n)\right|} = \left|e^{\lambda \Delta t}\right| < 1, \quad \text{for all } \lambda \Delta t < 0. \tag{1.3}$$

The analogous property for numerical methods is what we will refer to as linear stability.

For example, applying the forward Euler method to the test equation (1.1), yields

$$\frac{u^{n+1} - u^n}{\Delta t} = \lambda u^n \iff u^{n+1} = \underbrace{(1 + \lambda \Delta t)}_{=\xi_{FE}} u^n, \tag{1.4}$$

where $u^n$ is an approximation to $u(t^n)$. Then imposing $|\xi_{FE}| < 1$, we get

$$|1 + \lambda \Delta t| < 1 \iff -2 < \lambda \Delta t < 0, \tag{1.5}$$

which implies that $\Delta t < 2/|\lambda|$ must be satisfied for stability. As the time step-size, $\Delta t$, is constrained, we say forward Euler is conditionally stable.

As another example, we may apply the backward Euler method to the test equation. This yields

$$\frac{u^{n+1} - u^n}{\Delta t} = \lambda u^{n+1} \iff u^{n+1} = \underbrace{\frac{1}{1 - \lambda \Delta t}}_{=\xi_{BE}} u^n. \tag{1.6}$$

This time, imposing $|\xi_{BE}| < 1$ adds no new constraint to the time step-size. When no additional constraints are imposed on the time step-size, we say the numerical method is unconditionally stable.

More generally, to determine the stability constraint of any one-step method, we apply the method to the test equation, rearrange as $u^{n+1} = \xi(\lambda \Delta t) u^n$, and impose $\left|\xi(\lambda \Delta t)\right| < 1$. The quantity $\xi(\lambda \Delta t)$ is commonly referred to as the amplification factor, and the region,

$\{\lambda\Delta t \in \mathbb{C} \mid |\xi(\lambda\Delta t)| < 1\}$, the stability region. A numerical method is unconditionally stable if its stability region contains the entire negative real axis. It is conditionally stable otherwise.

### 1.1.2 Stability contours

A stability contour plot is a graphical device for understanding the stability constraint of a method. It offers a way for us to verify calculations done analytically, or to visualize the stability region of a numerical method where an analytic solution is infeasible. Stability contour plots in this thesis all show contours of the amplification factor $\xi(\lambda\Delta t)$ plotted over a subset of the region $\{\lambda\Delta t \in \mathbb{C} \mid \mathrm{Im}(\lambda\Delta t) \geq 0\}$, with a focus on the left half plane and the negative real line. Fig. 1.1 shows stability contours of the forward Euler and the backward Euler method. Note that the regions are symmetric with respect to the real axis and thus only $\mathrm{Im}(\lambda\Delta t) \geq 0$ will be plotted.

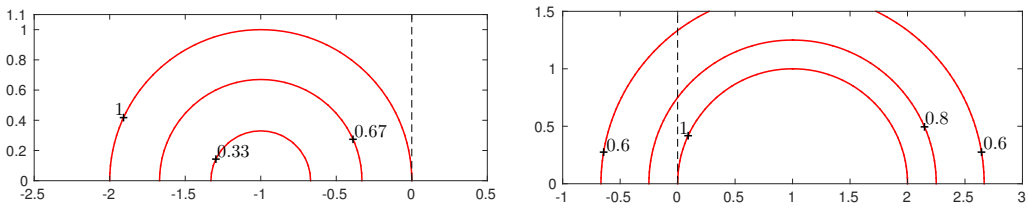


Figure 1.1: On the left is the stability contour plot for forward Euler. The stability region is the interior of the 1-contour. On the right is the stability contour plot for backward Euler. The stability region is the region outside the 1-contour.

### 1.1.3 Relation to stiff PDEs

Recall that our motivation is to develop methods suited to the time integration of stiff nonlinear PDEs. So how does the time step restriction of a numerical method derived from application to the test equation relate to time step selection for a stiff nonlinear PDE? The relation is as follows. Suppose the PDE has been discretized in space and we are to advance the solution of the resulting large system of ODEs, $u' = F(u)$, by one time step, i.e. advance the numerical solution $u^n$ to $u^{n+1}$. Over just one time step, it may be reasonable to consider the linearization,

$$u' = u^n + \left.\frac{\partial F}{\partial u}\right|_{u=u^n} (u - u^n) = \bar{u} + A(u - u^n),$$

or setting $v = u - u^n$, $v' = Av$. Further assuming that $A$ is diagonalizable, $A = T^{-1}DT$, where $D = \text{diag}(\lambda_1, \ldots, \lambda_N)$, we get

$$v' = T^{-1}DTv \iff (Tv)' = D(Tv) \tag{1.7}$$

$$\iff (Tv)'_k = \lambda(Tv)_k, \quad k = 1, \ldots, N. \tag{1.8}$$

In other words, under appropriate conditions, it may be fair to analyze the dynamics of the nonlinear system by inspecting the eigenvalues of the Jacobian from its linearized state. By requiring that the computation is stable for each eigenmode, the time step constraint will then be dictated by the largest absolute eigenvalue.

For instance, suppose we found, from a linearized system of ODEs, the eigenvalues to be $2(\cos(k\Delta x) - 1)/\Delta x^2$, $k = 1, \ldots, N$. Then the largest absolute eigenvalue can be bounded as

$$\left| \frac{2}{\Delta x^2}(\cos(k\Delta x) - 1) \right| \leq \frac{4}{\Delta x^2}, \tag{1.9}$$

and stable time step-sizes for forward Euler must then satisfy $\Delta t < \Delta x^2/4$. On the other hand, unconditionally stable methods such as backward Euler, maintain stability irrespective of the grid size $\Delta x$.

That unconditionally stable methods maintain stability irrespective of the grid size is crucial for the solution to stiff problems. In Chapter 5, we will solve problems in 2D and 3D where the largest absolute eigenvalues scale like $\mathcal{O}(h^2)$ and $\mathcal{O}(h^4)$, where $h$ is the spatial grid size. In those cases, a conditionally stable method such as forward Euler requires $\Delta t = \mathcal{O}(h^k)$, $k = 2$ or 4, and would give unnecessarily fine temporal resolution. More critically, this would greatly increase the cost of computation, or else, to combat this deficiency, have low spatial resolution.

## 1.2 Order of Accuracy

If two competing numerical methods, consume similar levels of resources (e.g., CPU time, or memory) but one gives more accurate approximations, then likely it would be deemed superior to the other.

For a time stepping method applied to the initial value problem

$$\begin{cases} u' = F(u), \quad 0 \leq t \leq T, \\ u(0) = u_0, \end{cases} \tag{1.10}$$

with step-size $\Delta t$, we will say that the method is convergent of order $k$ (or $k$th order accurate) if the global error behaves as

$$\|u^n - u(t^n)\| = \mathcal{O}(\Delta t^k), \quad \text{as} \quad \Delta t \to 0. \tag{1.11}$$

This points to a preference for high order accurate methods. Intuitively, in order to achieve some desired level of accuracy, a low order method will require a finer time step-size than a method of higher order accuracy. Then if each time step had comparable costs, the higher order method will require less resources overall to compute the solution.

Finally, we note that familiar methods such as forward Euler and backward Euler are first order accurate.

### 1.2.1 Richardson extrapolation

One of the methods under consideration relies on Richardson extrapolation, so we include here a brief note on this technique. A comprehensive text discussing its validity can be found in [27].

Suppose we are approximating some quantity, $q$, via a rule $\bar{q}$ that is dependent on the step-size $h$ and that the error is of the form

$$q - \bar{q}(h) = C_1 h^k + C_2 h^{k+1} + \cdots . \tag{1.12}$$

Observe that if we evaluate both $\bar{q}(h)$ and $\bar{q}(h/2)$, then we can eliminate the leading order error term,

$$q - q^*(h) = q - \frac{2^k \bar{q}(h/2) - \bar{q}(h)}{2^k - 1} = C h^{k+1} + \cdots , \tag{1.13}$$

to achieve higher order accuracy.

## 1.3 Overview

In Chapter 2, we formally introduce the notion of linear stabilization. Motivation for this technique is supplied by the need to handle a stiff nonlinear PDE describing axisymmetric mean curvature flow and leads us to the well-known first order linearly stabilized scheme and the EIN method of Duchemin and Eggers. Following that, the framework in which we analyze the stability of linearly stabilized schemes is set.

In Chapter 3 we investigate implicit-explicit (IMEX) linear multistep methods within the linear stabilization framework. A detailed comparison of the schemes based on IMEX methods and the EIN method is conducted. Our experiments suggest criteria in addition to unconditional stability are necessary for practical linearly stabilized schemes. This in turn eliminates third and higher order multistep-based linearly stabilized schemes from use.

In Chapter 4, we explore the use of exponential Runge-Kutta methods to mend this deficiency. A second order and a fourth order exponential Runge-Kutta method are verified to exhibit unconditional stability over an unbounded parameter range. Necessary considerations are given to the error constants of these schemes and our analysis narrows their range of applicability.

In Chapter 5, application of our linearly stabilized schemes to a number of 2D and 3D problems are presented. Not surprisingly, our second order schemes offer improvements over the commonly used first order linearly stabilized scheme. The experiments show that our schemes provide substantial efficiency improvement yet the complexity of its implementation is no greater than solving a heat equation with standard implicit methods.

Finally, some concluding remarks are presented in Chapter 6.

# Chapter 2

# Adding Zero, Unconditional Stability and a Modified Test Equation

To construct time integration schemes for stiff, nonlinear PDEs, we set out two key design principles. Firstly, we want to handle the nonlinearity simply and inexpensively. Secondly, we must be free to select time step-sizes reflecting the accuracy requirement, rather than step-sizes that are primarily constrained by stability. Linearly stabilized schemes, as we will see, adhere to both principles and are remarkably easy to implement.

## 2.1 Prototype 1D Problem

As a prototype, let us consider the following 1D axisymmetric mean curvature motion problem [9]:

$$u_t = \frac{u_{xx}}{1+u_x^2} - \frac{1}{u}, \qquad 0 < x < 10, \quad t > 0, \tag{2.1a}$$

with initial and boundary conditions

$$u(x,0) = 1 + 0.10 \sin\left(\frac{\pi}{5}x\right), \tag{2.1b}$$

$$u(0,t) = u(10,t) = 1. \tag{2.1c}$$

A time evolution of this problem is plotted in Fig. 2.1.

The presence of the $u_{xx}$ guarantees that (2.1) is stiff, suggesting that an implicit time stepping scheme would prove more efficient. However, having to additionally handle the factor of $(1+u_x^2)^{-1}$ complicates the linear algebra. Rather than a static linear system for which we could preprocess and solve efficiently, at each time step it is necessary to solve a

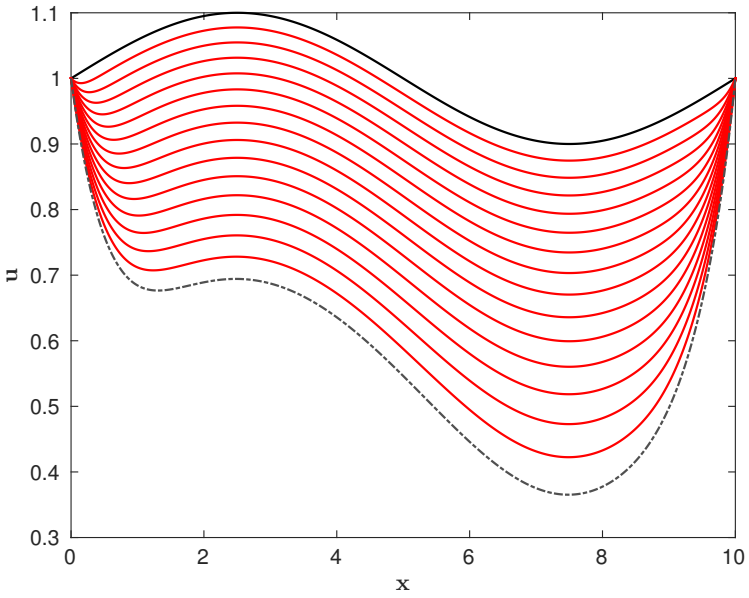


Figure 2.1: Time evolution according to (2.1). The black curve is the initial state. The gray dashed curve at the bottom is the final state at $T = 0.35$.

nonlinear system. Thus we are presented with a scenario where neither an implicit nor an explicit approach proves particularly palatable.

### 2.1.1 A first order linearly stabilized scheme

As demonstrated in Duchemin and Eggers [9] as well as in an earlier paper by Smereka [30], an efficient method to handling (2.1) is to add and subtract a linear Laplacian term to the right-hand side,

$$u_t = \underbrace{\frac{u_{xx}}{1+u_x^2} - \frac{1}{u} - u_{xx}}_{\mathcal{N}(u)} + \underbrace{u_{xx}}_{\mathcal{L}u}, \tag{2.2}$$

and then time step as

$$\frac{u^{n+1} - u^n}{\Delta t} = \mathcal{N}(u^n) + \mathcal{L}u^{n+1}. \tag{2.3}$$

Since this is our first instance of a linearly stabilized scheme, we remark on some of the key properties. We first note that in the continuous case, the modified equation (2.2) is unchanged from (2.1a). Next, note in the discrete case (2.3), the nonlinear term is evaluated explicitly, and ignoring the $\mathcal{L}u^{n+1}$ term, it is a forward Euler step. Over on the linear term, the implicit solve in this time stepping procedure is a backward Euler step. This method of time stepping is known as implicit-explicit (IMEX) or semi-implicit Euler [2, 30]. As it is a combination of explicit and implicit Euler steps, the accuracy is first order. We also note

that the simplicity of the implicit term means that the related linear algebra is efficient and easy to implement. Lastly, as a result of the implicit solution to the $\mathcal{L}u$ term, we may expect this scheme to have improved stability compared to a purely explicit scheme, and indeed this is the case. Discretizing with second order centred differences in space, we will show next that this scheme is unconditionally stable.

### 2.1.2 A von Neumann stability analysis

With the prescribed spatial-temporal discretization, we have at the interior nodes,

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = 4\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{4\Delta x^2 + (u_{j+1}^n - u_{j-1}^n)^2} - \frac{1}{u_j^n} - \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} + \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{\Delta x^2}. \tag{2.4}$$

The von Neumann stability analysis then proceeds by writing the numerical solution as the exact solution to the difference equation (2.4) perturbed by a single Fourier mode,

$$u_j^n = \bar{u}(j\Delta x, n\Delta t) + \xi^n e^{ikj\Delta x} = \bar{u}_j^n + \xi^n e^{ikj\Delta x}. \tag{2.5}$$

Recording the result term-by-term, we have

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = \frac{\bar{u}_j^{n+1}-\bar{u}_j^n}{\Delta t} + \frac{(\xi-1)\xi^n e^{ikj\Delta t}}{\Delta t}, \tag{2.6a}$$

$$\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{\Delta x^2} = \frac{\bar{u}_{j+1}^{n+1} - 2\bar{u}_j^{n+1} + \bar{u}_{j-1}^{n+1}}{\Delta x^2} + \frac{2}{\Delta x^2}\left(\cos(k\Delta x) - 1\right)\xi^{n+1}e^{ikj\Delta x}, \tag{2.6b}$$

$$\frac{u_{j+1}^{n} - 2u_j^{n} + u_{j-1}^{n}}{\Delta x^2} = \frac{\bar{u}_{j+1}^{n} - 2\bar{u}_j^{n} + \bar{u}_{j-1}^{n}}{\Delta x^2} + \frac{2}{\Delta x^2}\left(\cos(k\Delta x) - 1\right)\xi^{n}e^{ikj\Delta x}, \tag{2.6c}$$

$$\frac{1}{u_j^n} = \frac{1}{\bar{u}_j^n + \xi^n e^{ikj\Delta x}} \approx \frac{1}{\bar{u}_j^n} - \xi^n e^{ikj\Delta x}\frac{1}{(\bar{u}_j^n)^2}, \tag{2.6d}$$

and

$$
\begin{aligned}
4\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{4\Delta x^2 + (u_{j+1}^n - u_{j-1}^n)^2} &= 4\frac{\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n + 2(\cos(k\Delta x) - 1)\xi^n e^{ikj\Delta x}}{4\Delta x^2 + [\bar{u}_{j+1}^n - \bar{u}_{j-1}^n - 2i\sin(k\Delta x)\xi^n e^{ikj\Delta x}]^2} \\
&= 4\frac{2(\cos(k\Delta x) - 1)[\frac{\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n}{2(\cos(k\Delta x)-1)} + \xi^n e^{ikj\Delta x}]}{4\Delta x^2 + (2i\sin(k\Delta x))^2[\frac{\bar{u}_{j+1}^n - \bar{u}_{j-1}^n}{2i\sin(k\Delta x)} - \xi^n e^{ikj\Delta x}]^2} \\
&\approx 4\frac{\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n}{4\Delta x^2 + (\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)^2} \\
&+ 8(\cos(k\Delta x) - 1)\xi^n e^{ikj\Delta x}\frac{1}{4\Delta x^2 + (\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)^2} \\
&- 8i\sin(k\Delta x)\xi^n e^{ikj\Delta x}(\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)(\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n) \\
&\approx 4\frac{\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n}{4\Delta x^2 + (\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)^2} \\
&+ \frac{2}{\Delta x^2}(\cos(k\Delta x) - 1)\xi^n e^{ikj\Delta x}\frac{1}{1 + (D_1\bar{u}_j^n)^2},
\end{aligned}
\tag{2.6e}
$$

where $D_1\bar{u}_j^n = (\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)/(2\Delta x)$. Combining, (2.6) simplifies to

$$
\frac{\xi - 1}{\Delta t} = \frac{2}{\Delta x^2}\frac{\cos(k\Delta x) - 1}{1 + (D_1\bar{u}_j^n)^2} + \frac{1}{(\bar{u}_j^n)^2} + \frac{2}{\Delta x^2}(\cos(k\Delta x) - 1)(\xi - 1), \tag{2.7}
$$

from which we can then isolate the amplification factor,

$$
\xi = 1 + \underbrace{\Delta t\frac{\frac{2}{\Delta x^2}\frac{\cos(k\Delta x)-1}{1+(D_1\bar{u}_j^n)^2} + \frac{1}{(\bar{u}_j^n)^2}}{1 - \frac{2\Delta t}{\Delta x^2}(\cos(k\Delta x) - 1)}}_{=w}. \tag{2.8}
$$

In the next steps, we will show $|\xi| < 1$ for all $\Delta t > 0$, i.e. stability is unconditional. We show the equivalent statement $-2 < w < 0$. First, $w < 0$. As the denominator is positive, $w > 0$ will hold true so long as the numerator is negative,

$$
\frac{2}{\Delta x^2}\frac{\cos(k\Delta x) - 1}{1 + (D_1\bar{u}_j^n)^2} + \frac{1}{(\bar{u}_j^n)^2} < 0 \iff \left(\frac{\Delta x}{\bar{u}_j^n}\right)^2 < \frac{2(1 - \cos(k\Delta x))}{1 + (D_1\bar{u}_j^n)^2}. \tag{2.9}
$$

This last relation is satisfied on the assumption that $\Delta x \ll \bar{u}_j^n$. Next, we examine the numerator of $w + 2$,

$$
\begin{aligned}
&\frac{2\Delta t}{\Delta x^2}\frac{\cos(k\Delta x) - 1}{1 + (D_1\bar{u}_j^n)^2} + \frac{\Delta t}{(\bar{u}_j^n)^2} + 2 - 4\frac{\Delta t}{\Delta x^2}(\cos(k\Delta x) - 1) \\
&\quad = 2\frac{\Delta t}{\Delta x^2}(\cos(k\Delta x) - 1)\left(\frac{1}{1 + (D_1\bar{u}_j^n)^2} - 2\right) + 2 + \frac{\Delta t}{(\bar{u}_j^n)^2}.
\end{aligned}
\tag{2.10}
$$

Since each term is positive without restriction, we have that $w+2>0$, and thus $|\xi|<1$ for all $\Delta t>0$.

### 2.1.3 Second order by Richardson extrapolation

As stated at the outset, the time stepping procedure in (2.3) is only first order. The work of Duchemin and Eggers [9] (and as was suggested but not implemented in [30]) extends the method to second order by Richardson extrapolation. Moreover, they generalized the approach with a free parameter, $p$, i.e.,

$$u_t = \frac{u_{xx}}{1+u_x^2} - \frac{1}{u} - pu_{xx} + pu_{xx}, \tag{2.11}$$

and derived restrictions to $p$ on the condition that the resulting scheme be unconditionally stable. With the semi-implicit Euler approach, they found $p \geq 0.5/(1+(D_1\bar{u}_j^n)^2)$ to be sufficient. With the additional Richardson extrapolation, the restriction is $p \geq (2/3)/(1+(D_1\bar{u}_j^n)^2)$.

## 2.2 A Modified Test Equation

This method of linear stabilization would be extremely cumbersome if in each case we had to perform a von Neumann analysis to determine the stability. Thankfully, there is an alternative. To begin, let us now consider the more general problem, $u' = F(u)$, which may be from a spatial discretization of some nonlinear PDE, and we assume $F(u)$ is a stiff, nonlinear term. We can modify, in a way analogous to (2.2), by subtracting and adding a linear term that "resembles" $F(u)$,

$$u_t = \underbrace{(F(u) - pLu)}_{\mathcal{N}(u)} + \underbrace{pLu}_{\mathcal{L}u}, \tag{2.12}$$

and demand unconditionally stable time stepping. This last request we now address.

To progress, we abandon (2.12) and instead examine a related, but simplified scenario that we will refer to as the modified test equation:

$$u' = (1-p)\lambda u + p\lambda u, \quad \text{where} \quad \lambda < 0,\, p > 0. \tag{2.13}$$

In (2.13), the quantity $\lambda$ captures the character of $F$ (e.g. the eigenvalues of the Jacobian of the linearized $F$), and the quantity $p\lambda u$ represents the linear component that closely resembles $F(u)$. Note that when $p=0$, the modified test equation reduces to the standard test equation, $u' = \lambda u$.

We will discuss next the stability properties of three time stepping methods as applied to the modified test equation (2.13).

### 2.2.1 Forward Euler

Forward Euler is a first order time stepping method that treats the right-hand side explicitly. Application to (2.13) is therefore no different than to the standard test equation. Thus there is no hope of unconditional stability.

### 2.2.2 Linearly stabilized semi-implicit Euler

Semi-implicit Euler time stepping was demonstrated on the 1D curvature motion problem (2.1) via (2.2) and (2.3), and its stability further analyzed. In the case of the modified test equation, we identify $\mathcal{N}(u^n) = (1-p)\lambda u^n$ and $\mathcal{L}u^{n+1} = p\lambda u^{n+1}$, to get

$$\frac{u^{n+1}-u^n}{\Delta t} = (1-p)\lambda u^n + p\lambda u^{n+1} \iff u^{n+1} = \underbrace{\left(1+\frac{\lambda\Delta t}{1-p\lambda\Delta t}\right)}_{=\xi_E} u^n. \tag{2.14}$$

Enforcing stability, i.e. $|\xi_E| < 1$, for all $\lambda\Delta t < 0$, we find

$$|\xi_E| < 1 \iff -2 < \frac{\lambda\Delta t}{1-p\lambda\Delta t} < 0 \iff p > 0 \quad \text{and} \quad (2p-1)\lambda\Delta t < 2. \tag{2.15}$$

Thus unconditional stability is guaranteed if $p \geq 1/2$.

Going forward, we shall refer to this scheme as SBDF1.

### 2.2.3 Explicit-implicit-null

In [9], Duchemin and Eggers extended the SBDF1 approach to second order by using Richardson extrapolation, and they referred to their methodology as explicit-implicit-null (EIN). For their method, the amplification factor, $\xi_{EIN}$, can be expressed in terms of $\xi_E$,

$$\xi_{EIN} = 2\xi_E^2(\Delta t/2) - \xi_E(\Delta t) = 1 + \underbrace{\frac{z\left(p(3p-2)z^2 + 2(1-4p)z + 4\right)}{(1-pz)(2-pz)^2}}_{=w}, \tag{2.16}$$

where $z = \lambda\Delta t$. Similar to before, we enforce $|\xi_{EIN}| < 1$, or equivalently $-2 < w < 0$, for all $z < 0$ to derive a restriction on $p$. We first observe that since $p > 0$, the denominator of $w$ is positive for all $z < 0$. Thus a necessary condition is $3p - 2 > 0 \iff p > 2/3$, as we require the quadratic in the numerator to be positive for all $z < 0$. Further, the roots of that quadratic are positive whenever they are real,

$$\frac{1}{2p(3p-2)}\left(2(4p-1) \pm 2\sqrt{(4p-1)^2 - 4p(3p-2)}\right) > 0. \tag{2.17}$$

Therefore, $p > 2/3$ is necessary and sufficient for $w < 0$. Next, we show $w + 2 > 0$. The numerator of $w + 2$ can be simplified to

$$-p(2 - 3p + 2p^2)z^3 + (2 - 8p + 10p^2)z^2 + 4(1 - 4p)z + 8. \tag{2.18}$$

The coefficients of the powers of $z$ have the property

$$\begin{aligned}
&[z] = 4(1 - 4p) > 0 \quad \text{whenever} \quad p > 1/4, \\
&[z^2] = 2 - 8p + 10p^2 > 0 \quad \text{since its discriminant } 8^2 - 4(2)(10) < 0, \\
&[z^3] = -p(2 - 3p + 2p^2) > 0 \quad \text{since its discriminant } (-3)^2 - 4(2)(2) < 0,
\end{aligned}$$

for all $z < 0$, thus guaranteeing the numerator is positive. And since the denominator, as stated previously, is positive, we are guaranteed $w + 2 > 0$, and thus unconditional stability is guaranteed if $p > 2/3$.

*Remark* 1. It is perhaps more faithful to write the restrictions as $p\lambda/\lambda > 2/3$ rather than simply $p > 2/3$, as it is necessarily the ratio of the two that must satisfy the restriction, and not the parameter $p$. This distinction is vital for any problem beyond a simple test equation, where the largest absolute eigenvalue of the nonlinear operator, $\lambda_F$, and the largest absolute eigenvalue of the linear operator, $\lambda_{\mathcal{L}}$, may follow the same scaling, e.g. $\lambda_F, \lambda_{\mathcal{L}} = \mathcal{O}(\Delta x^{-2})$, but the actual values may be far apart, e.g. $\lambda_F \approx 100\lambda_{\mathcal{L}}$.

*Remark* 2. What we now understand is that in order to linearly stabilize effectively, we need the ratio of the eigenvalues to meet a specific bound. This bound, if we assume a fixed and reasonable choice of a linear operator $\mathcal{L}$, however, is specific to the time stepping procedure, and is met by choosing a sufficiently large value of $p$.

*Remark* 3. Finally, we must mention that this provides us with a simple avenue to selecting $p$ without a von Neumann analysis, as the latter in many cases may be infeasible. For example, the analysis in (2.6) reveals that

$$\lambda_F = \frac{2}{\Delta x^2}\frac{\cos(k\Delta x) - 1}{1 + (D_1\bar{u}_j^n)^2} \quad \text{and} \quad \lambda_{\mathcal{L}} = \frac{2}{\Delta x^2}(\cos(k\Delta x) - 1). \tag{2.19}$$

So then to apply, e.g. EIN, to (2.11) with centred differences, we would select $p$ to satisfy

$$\frac{2}{3} < \frac{p\lambda_{\mathcal{L}}}{\lambda_F} = p(1 + (D_1\bar{u}_j^n)^2) \iff p > \frac{2}{3}(1 + (D_1\bar{u}_j^n)^2)^{-1}. \tag{2.20}$$

## 2.3 Numerical Results

We present in this section some numerical tests to support our claims.

### 2.3.1 Stability contours

Stability contour plots are shown in Figs. 2.2 and 2.3 as a verification of the earlier analysis of SBDF1 and the EIN method. In the first case, we found the parameter restriction to be $p \geq 0.5$, and this is in agreement with Fig. 2.2. When we set $p = 0.45$, the stable range along the negative real axis is clearly bounded. But at $p = 0.50$ and greater, its stability region includes the entirety of the negative real axis.

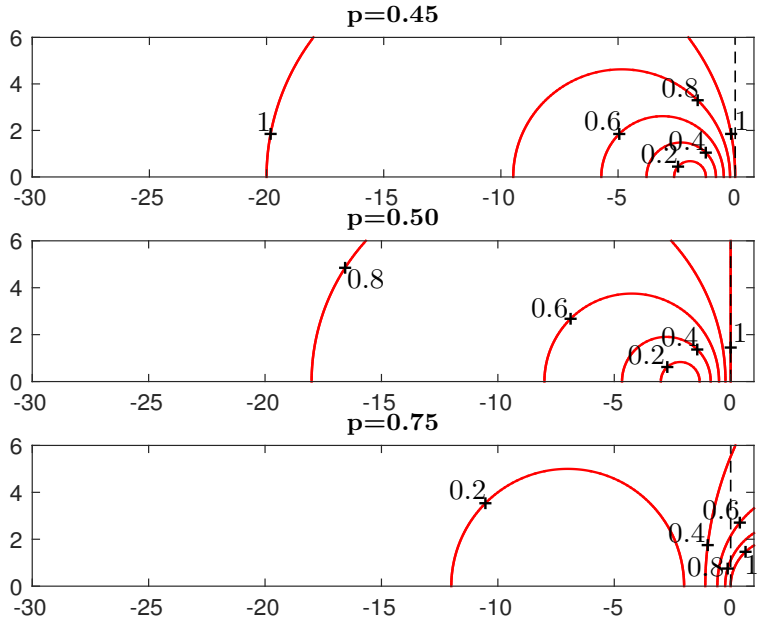


Figure 2.2: Stability contours for SBDF1 at various $p$

The same is demonstrated for the EIN method in Fig. 2.3. Again we see that below the derived threshold, the stability region is bounded. But beyond that, the stability region contains the entire negative real axis.

### 2.3.2 Numerical convergence test

The convergence of SBDF1 and EIN are demonstrated next on (2.1) with (2.2) in place of (2.1a). We solve to time $T = 0.35$ with $N = 2048$ spatial grid nodes. As a reference solution, we use an explicit third order Runge-Kutta method with time step-size $\Delta t = 1.46 \times 10^{-5}$. (Typical explicit methods demand step-sizes comparable for stability.) With the linearly stabilized schemes, we will solve (2.1) with step-sizes as large as $\Delta t = 2.09 \times 10^{-2}$. We show the max norm relative error.

Results of the numerical convergence study are shown in Fig. 2.4. Both schemes converge with the expected order of accuracy. No issues with stability are observed.

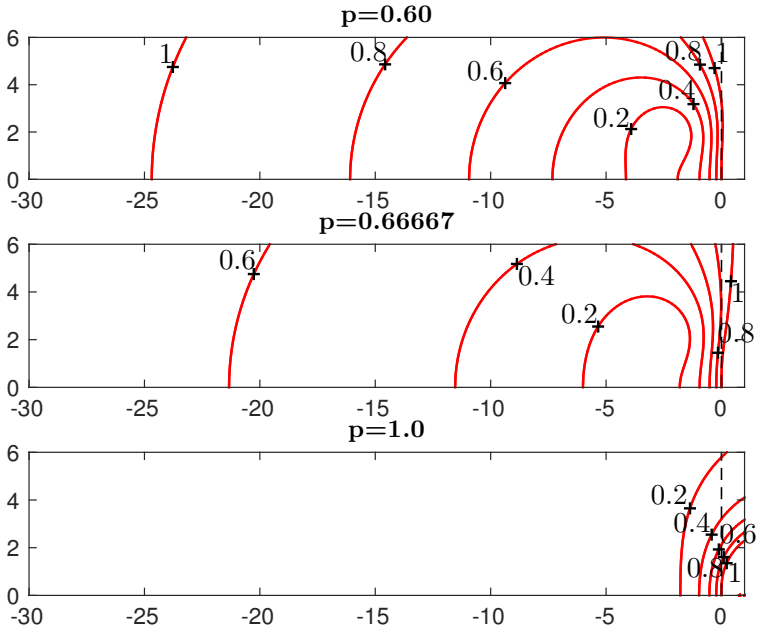


Figure 2.3: Stability contours for EIN at various $p$.

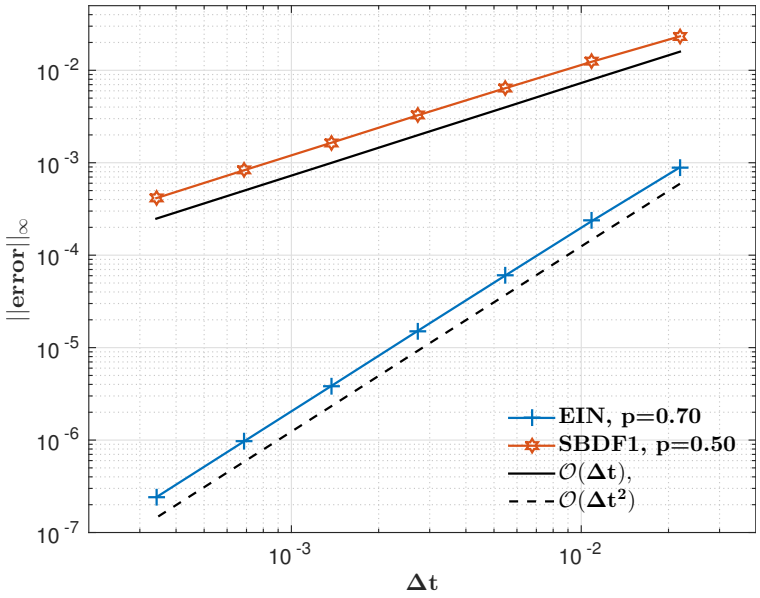


Figure 2.4: Result of numerical convergence study to (2.1) with SBDF1 and EIN. Values $p = 0.50$ and $p = 0.70$ were chosen for SBDF1 and EIN respectively.

# Chapter 3

# IMEX Linear Multistep Methods

For equations whose right-hand side comprise of a stiff linear component and a nonstiff nonlinear part, a popular class of methods to apply are the implicit-explicit linear multistep methods[1]. The simplest of these is the semi-implicit Euler – forward Euler to the nonlinear component and backward Euler to the linear, stiff component – a scheme that we reviewed in Chapter 2.

In this chapter, we investigate the use of IMEX methods within the linear stabilization framework. Implicit solution of the added linear term provides the needed stability, and the remaining terms, including the stiff nonlinear term, are solved explicitly.

## 3.1 IMEX Formulas

In [2], IMEX schemes up to order four are investigated and a select number are singled out for their extensive use in the literature or for desired properties such as strong high frequency damping. As we are familiar with the first order variant, we begin by listing the second order methods of interest. These, and the higher order variants, are presented as applied to the ODE

$$u' = f + g,$$

where $f$ we identify as the nonlinear/nonstiff component and $g$ the stiff linear component.

**Second order methods**

CNAB:

$$\frac{u^{n+1} - u^n}{\Delta t} = \frac{3}{2} f^n - \frac{1}{2} f^{n-1} + \frac{1}{2}(g^{n+1} + g^n), \tag{3.1}$$

[1] We will refer to these simply as IMEX methods.

mCNAB:

$$\frac{u^{n+1}-u^n}{\Delta t} = \frac{3}{2}f^n - \frac{1}{2}f^{n-1} + \frac{9}{16}g^{n+1} + \frac{3}{8}g^n + \frac{1}{16}g^{n-1}, \tag{3.2}$$

CNLF:

$$\frac{u^{n+1}-u^{n-1}}{2\Delta t} = f^n + \frac{1}{2}(g^{n+1}+g^{n-1}), \tag{3.3}$$

SBDF2:

$$\frac{3u^{n+1}-4u^n+u^{n-1}}{2\Delta t} = 2f^n - f^{n-1} + g^{n+1}. \tag{3.4}$$

**Third order methods**
SBDF3:

$$\frac{1}{\Delta t}\left(\frac{11}{6}u^{n+1} - 3u^n + \frac{3}{2}u^{n-1} - \frac{1}{3}u^{n-2}\right) = 3f^n - 3f^{n-1} + f^{n-2} + g^{n+1}. \tag{3.5}$$

**Fourth order methods**
SBDF4:

$$\frac{1}{\Delta t}\left(\frac{25}{12}u^{n+1} - 4u^n + 3u^{n-1} - \frac{4}{3}u^{n-2} + \frac{1}{4}u^{n-3}\right) = 4f^n - 6f^{n-1} + 4f^{n-2} - f^{n-3} + g^{n+1}. \tag{3.6}$$

In the next section, we will apply these IMEX schemes to the modified test equation to determine for each scheme the range of $p$ suitable for linear stabilization.

## 3.2 Analysis of the Amplification Polynomials

Let us restate here the modified test equation and the basic assumptions we make. The modified test equation is

$$u' = (1-p)\lambda u + p\lambda u, \quad \text{where} \quad \lambda < 0,\, p > 0.$$

The goal is, for each IMEX scheme, to identify the restriction on the parameter $p$ such that when satisfied, we may freely choose the time step-size without being subject to a stability constraint.

### 3.2.1 Schur and von Neumann polynomials

The resulting polynomial from applying an $n$th order IMEX method to the modified test equation will be a degree $n$ polynomial in the amplification factor, $\xi$. The tool of choice for analyzing these amplification polynomials is the theory of von Neumann polynomials [31, Chapter 4]. Below we give the relevant definitions and theorems.

**Definition 3.1.** The polynomial $\phi$ is a Schur polynomial if all its roots, $r_q$, satisfy $|r_q| < 1$.

**Definition 3.2.** The polynomial $\phi$ is a von Neumann polynomial if all its roots, $r_q$, satisfy $|r_q| \leq 1$.

**Definition 3.3.** The polynomial $\phi$ is a simple von Neumann polynomial if $\phi$ is a von Neumann polynomial and its roots on the unit circle are simple roots.

**Definition 3.4.** For any polynomial $\phi(\xi) = \sum_{j=0}^{n} a_j \xi^j$, we define the polynomial $\phi^*$ by $\phi^*(\xi) = \sum_{j=0}^{n} a_{n-j}^* \xi^j$, where $^*$ on the coefficient denotes the complex conjugate.

**Definition 3.5.** For any polynomial $\phi_n(\xi) = \sum_{j=0}^{n} a_j \xi^j$, we define recursively the polynomial $\phi_{n-1}$ by

$$\phi_{n-1}(\xi) = \frac{\phi_n^*(0)\phi_n(\xi) - \phi_n(0)\phi_n^*(\xi)}{\xi}. \tag{3.7}$$

**Theorem 3.6.** *$\phi_n$ is a simple von Neumann polynomial if and only if either*

*(a) $|\phi_n(0)| < |\phi_n^*(0)|$ and $\phi_{n-1}$ is a simple von Neumann polynomial or*

*(b) $\phi_{n-1}$ is identically zero and $\phi_n'$ is a Schur polynomial.*

### 3.2.2 Amplification polynomials of second order IMEX methods

We start by applying CNAB (3.1) to the modified test equation (2.13). Combined with the ansatz $u^n = \xi^n$ and setting $z = \lambda \Delta t$, we get the amplification polynomial

$$\Phi_2(\xi) = \left(1 - \frac{1}{2}zp\right)\xi^2 - \left(1 + z\left(\frac{3}{2} - p\right)\right)\xi + \frac{1}{2}z(1-p). \tag{3.8}$$

The next series of steps will show that for all $z < 0$ (i.e. $\lambda \Delta t < 0$,) (3.8) is a simple von Neumann polynomial if and only if $p \geq 1$. We do so by showing that Theorem 3.6 holds for $\Phi_2$. This will imply that the amplification factor satisfies $|\xi| < 1$ for all $\lambda \Delta t < 0$.

We first give $\Phi_2^*$ and $\Phi_1$ as defined by the processes in Definitions 3.4 and 3.5,

$$\Phi_2^*(\xi) = \frac{1}{2}z(1-p)\xi^2 - \left(1 + z\left(\frac{3}{2} - p\right)\right)\xi + 1 - \frac{1}{2}zp, \tag{3.9}$$

and

$$\Phi_1(\xi) = \left(1 - zp + z^2\left(\frac{1}{2}p - \frac{1}{4}\right)\right)\xi - 1 - z(1-p) + z^2\left(\frac{3}{4} - \frac{1}{2}p\right). \tag{3.10}$$

Next, we verify that if $p \geq 1$, then $|\Phi_2(0)| < |\Phi_2^*(0)|$. Reformulating the expression as

$$|\Phi_2(0)| < |\Phi_2^*(0)| \iff 0 < \left(\Phi_2^*(0)\right)^2 - \Phi_2^2(0) = 1 - zp - \frac{1}{2}z^2\left(\frac{1}{2} - p\right),$$

(and keeping in mind that we ask this inequality to hold only for $z < 0$,) we find that the contribution from each term in the rightmost quadratic is positive, thus verifying the claim. Finally, we show that $\Phi_1$ is simple von Neumann directly. Denoting the root of $\Phi_1$ as $\xi_1$,

$$|\xi_1| < 1 \iff \left|\frac{(2p-3)z - 2}{(2p-1)z - 2}\right| < 1 \iff 0 < 8z((p-1)z - 1),$$

which holds for all $z < 0$ if and only if $p \geq 1$.

Other IMEX schemes are analyzed in the same way. The amplification polynomials of the second order schemes are recorded for reference in Table 3.1, along with the parameter restriction for which we observe unconditional stability.

Table 3.1: Amplification polynomial of second order IMEX methods. The rightmost column is the guide for choosing $p$.

| **Method** | **Amplification Polynomial** | $p\lambda/\lambda \in$ |
|---|---|---|
| CNAB | $\left(1 - \frac{1}{2}zp\right)\xi^2 - \left(1 + z\left(\frac{3}{2} - p\right)\right)\xi + \frac{1}{2}z(1-p)$ | $[1, \infty)$ |
| mCNAB | $\left(1 - \frac{9}{16}zp\right)\xi^2 - \left(1 + z\left(\frac{3}{2} - \frac{9}{8}p\right)\right)\xi + \frac{1}{2}z\left(1 - \frac{9}{8}p\right)$ | $[8/9, \infty)$ |
| CNLF | $(1 - pz)\,\xi^2 - 2z(1-p)\xi - (1 + pz)$ | $[1/2, \infty)$ |
| SBDF2 | $\left(\frac{3}{2} - zp\right)\xi^2 - 2\left(1 + z(1-p)\right)\xi + \frac{1}{2} + z(1-p)$ | $[3/4, \infty)$ |

### 3.2.3 Amplification polynomials of third and fourth order IMEX methods

We continue with the analysis for higher order IMEX schemes. Again, amplification polynomials and parameter restrictions are derived. Because the expressions and manipulations quickly become cumbersome and tedious for higher order methods, the computer algebra system, MAPLE™, was used for the majority of the calculations.

Listed in Table 3.2 are respectively the amplification factor and the parameter restriction for SBDF3 and SBDF4. We must point out a crucial difference. In contrast to the second order methods, the derived parameter restriction leaves only a finite interval. This will

Table 3.2: Amplification polynomial and choice of parameter when applying high order IMEX methods to the modified test equation for unconditional stability.

| **Method** | **Amplification Polynomial** | $p\lambda/\lambda \in$ |
|---|---|---|
| SBDF3 | $\left(\frac{11}{6}-zp\right)\xi^3 - 3\left(1+z(1-p)\right)\xi^2 + \frac{3}{2}\left(1+2z(1-p)\right)\xi$ $-\frac{1}{3}\left(1+3z(1-p)\right)$ | $[7/8, 2]$ |
| SBDF4 | $\left(\frac{25}{12}-zp\right)\xi^4 - 4\left(1+z(1-p)\right)\xi^3 + 3\left(1+2z(1-p)\right)\xi^2$ $-\frac{4}{3}\left(1+3z(1-p)\right)\xi + \frac{1}{4}\left(1+4z(1-p)\right)$ | $[15/16, 5/4]$ |

be addressed further in Section 3.3.3 where it is demonstrated that this detail renders the linearly stabilized SBDF3 and SBDF4 ineffective.

## 3.3 Convergence of Linearly Stabilized IMEX Methods

We present in this section numerical tests to support our claims. Stability contours are shown and a numerical convergence test to (2.1) is conducted. We then provide an answer as to why linearly stabilized SBDF3 and SBDF4 fail.

### 3.3.1 Stability contours

Presented in Figs. 3.1 to 3.4 are stability contours for the second order IMEX schemes, (3.1) to (3.4), applied to the modified test equation, (2.13). For each, we provide stability contours with $p$ set at $0.95p_0$, $p_0$, $1.5p_0$, where $p_0$ is the minimum required for unconditional stability (Table 3.1, rightmost column).

In Fig. 3.5 are stability contours for SBDF3 (3.5), and in Fig. 3.6 are stability contours for SBDF4 (3.6). In each, we plot four instances to capture the transitions from conditional to unconditional stability and from unconditional stability back to conditional stability.

### 3.3.2 Numerical convergence test

Convergence of the proposed schemes will be tested on the 1D curvature motion problem, which we restate below:

$$u_t = \frac{u_{xx}}{1+u_x^2} - \frac{1}{u} - pu_{xx} + pu_{xx}, \quad 0 < x < 10, \quad t > 0, \tag{3.11a}$$

with initial and boundary conditions

$$u(x,0) = 1 + 0.10\sin\left(\frac{\pi}{5}x\right) \tag{3.11b}$$

$$u(0,t) = u(10,t) = 1. \tag{3.11c}$$

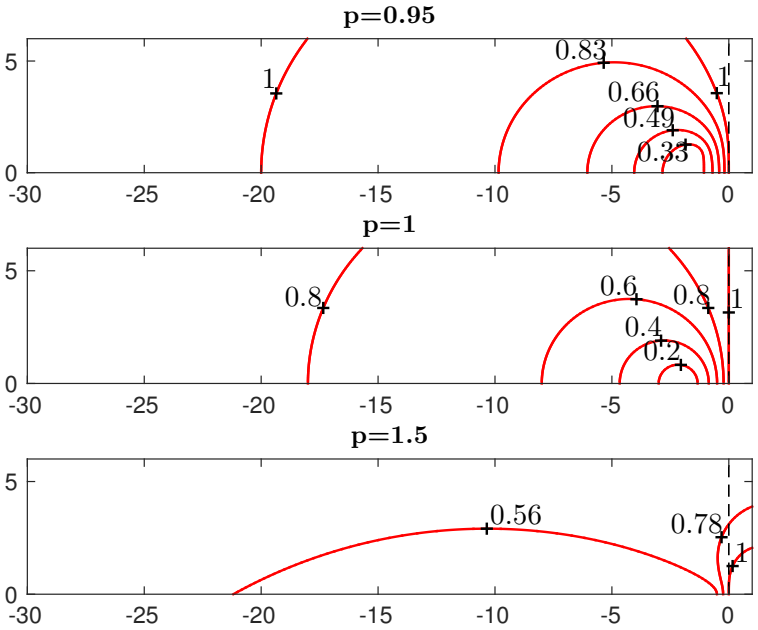


Figure 3.1: Stability contours for CNAB at various $p$.

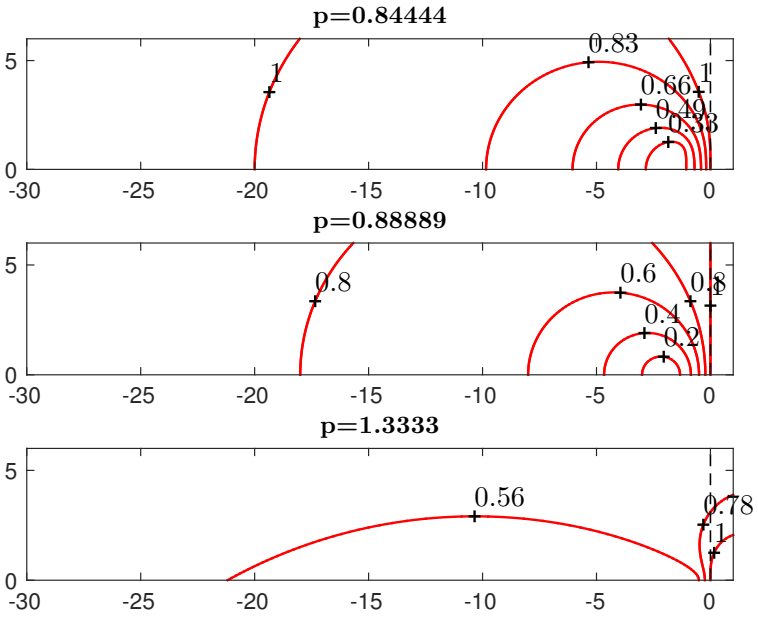


Figure 3.2: Stability contours for mCNAB at various $p$.

As in Section 2.3.2, we solve to time $T = 0.35$ with $N = 2048$ spatial grid nodes. A reference solution is generated using Heun's third order Runge-Kutta method [16] with time step-size $\Delta t = 1.46 \times 10^{-5}$, and a max norm relative error is calculated. Starting values necessary for the multistep methods are found using the same third order Runge-Kutta method. The values of $p$ used for each scheme are as indicated in Fig. 3.7.

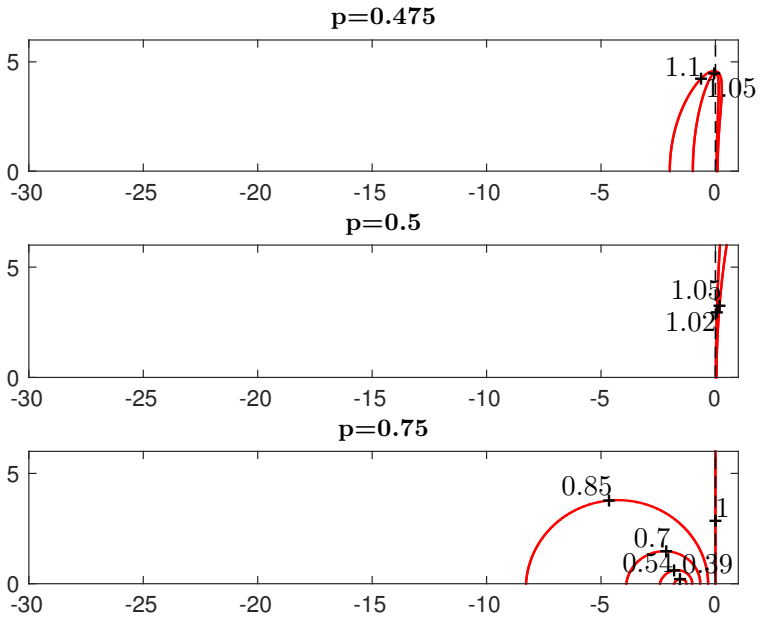


Figure 3.3: Stability contours for CNLF at various $p$.

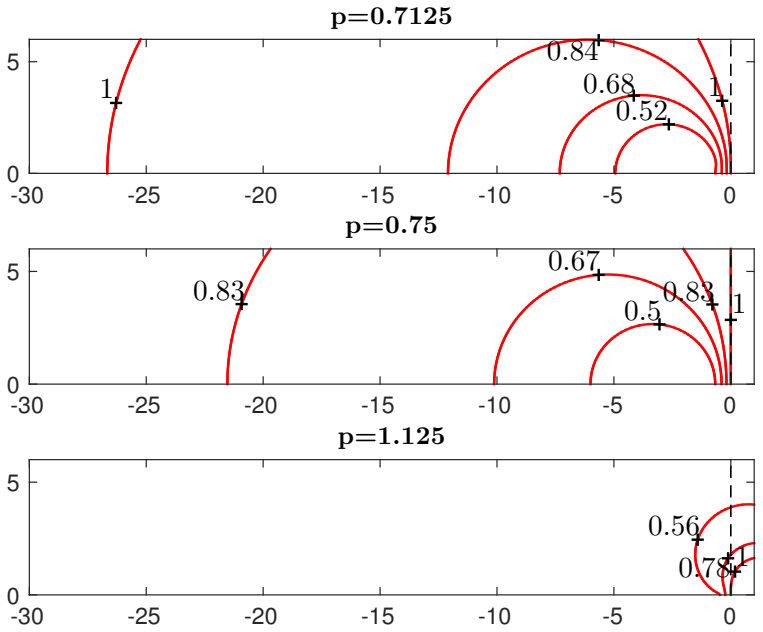


Figure 3.4: Stability contours for SBDF2 at various $p$.

Results of the numerical convergence study are shown in Fig. 3.7. Each of the second order methods converge with the order of accuracy expected, with SBDF2 having the largest errors, followed by CNAB/mCNAB (identical performance), CNLF, and finally EIN. SBDF3 converges nicely with $p = 0.875$. SBDF4 does not exhibit fourth order convergence and in fact fails at both $\Delta t = 6.84 \times 10^{-4}$ and $\Delta t = 3.42 \times 10^{-4}$. We discuss next why SBDF4 fails, and show also that SBDF3 suffers from the same defect.

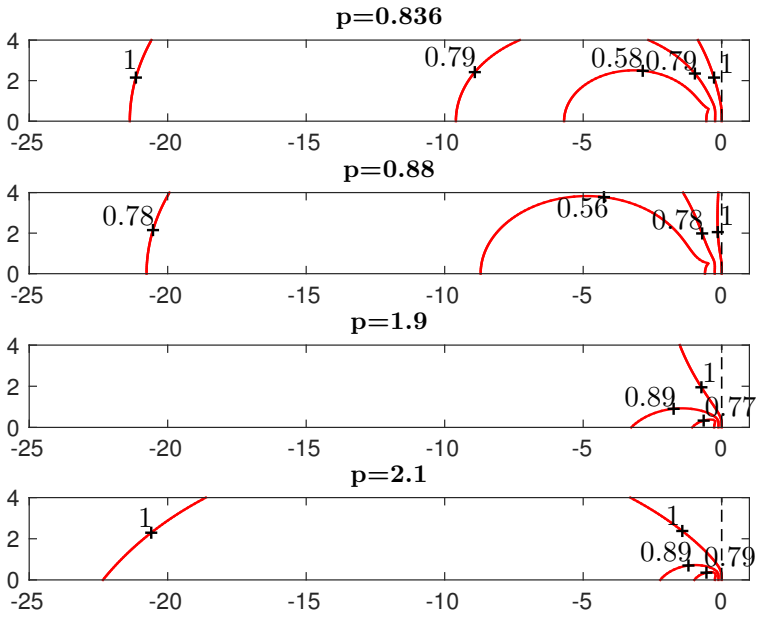


Figure 3.5: Stability contours for SBDF3 at various $p$.

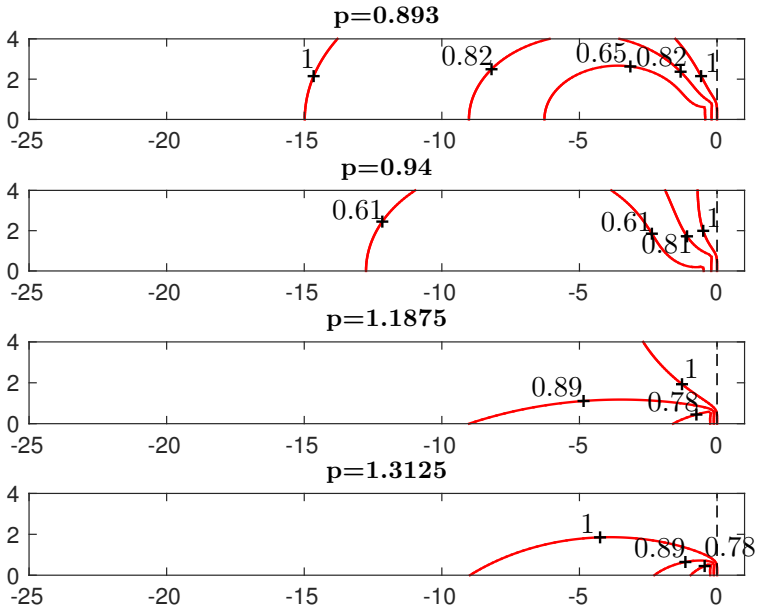


Figure 3.6: Stability contours for SBDF4 at various $p$.

### 3.3.3 Failure with high order IMEX methods

First, let us tabulate the observed (non)convergence of SBDF3 at various values of $p$. In Table 3.3, we document three cases. The first case is the one shown in Fig. 3.7. The second case exhibits a drastic drop in the observed convergence rate and in the third case the method diverges as the time step-size is reduced. We attribute the divergence of SBDF3 and SBDF4 to the fact that their parameter restriction is a finite interval.

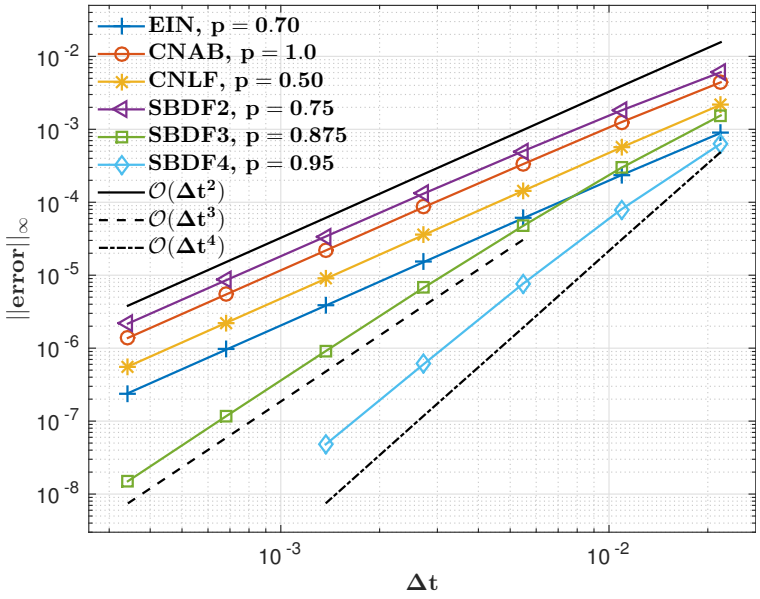


Figure 3.7: Numerical convergence study to (3.11) with IMEX methods. Convergence of EIN is also included for comparison. The test with mCNAB is omitted, but would otherwise overlap with CNAB.

Table 3.3: (Non)convergence of SBDF3 at various $p$.

| | Observed convergence rate | | |
|---|---|---|---|
| $\Delta t$ | $p = 0.875$ | $p = 1.475$ | $p = 1.675$ |
| $2.19 \times 10^{-2}$ | – | – | – |
| $1.09 \times 10^{-2}$ | 2.39 | 2.39 | 2.39 |
| $5.47 \times 10^{-3}$ | 2.64 | 2.64 | 1.23 |
| $2.73 \times 10^{-3}$ | 2.80 | 2.80 | $-1.51$ |
| $1.37 \times 10^{-3}$ | 2.90 | 2.90 | $-3.95$ |
| $6.84 \times 10^{-4}$ | 2.95 | 2.95 | diverge |
| $3.42 \times 10^{-4}$ | 2.97 | 2.08 | diverge |

To see this, recall the parameter restrictions listed in Table 3.2 and the relation (2.19). Combining, we have for SBDF3 the parameter restriction

$$\frac{7}{8(1+(D_1\bar{u}_j^n)^2)} \leq p \leq \frac{2}{1+(D_1\bar{u}_j^n)^2}, \quad j = 1, \ldots, N, \tag{3.12}$$

or equivalently

$$\max_{1\leq j\leq N} \frac{7}{8(1+(D_1\bar{u}_j^n)^2)} \leq p \leq \min_{1\leq j\leq N} \frac{2}{1+(D_1\bar{u}_j^n)^2}. \tag{3.13}$$

Failure of the method is due to being unable to satisfy the parameter constraint at every grid node simultaneously. From Fig. 2.1, we see that $\max_j (D_1\bar{u}_j^n)^2$ is increasing as the

solution evolves and is most extreme near the boundaries. Thus we expect instabilities to develop, and to develop in those regions first. This analysis is corroborated by Fig. 3.8, where we see instabilities developing near the right-hand boundary.

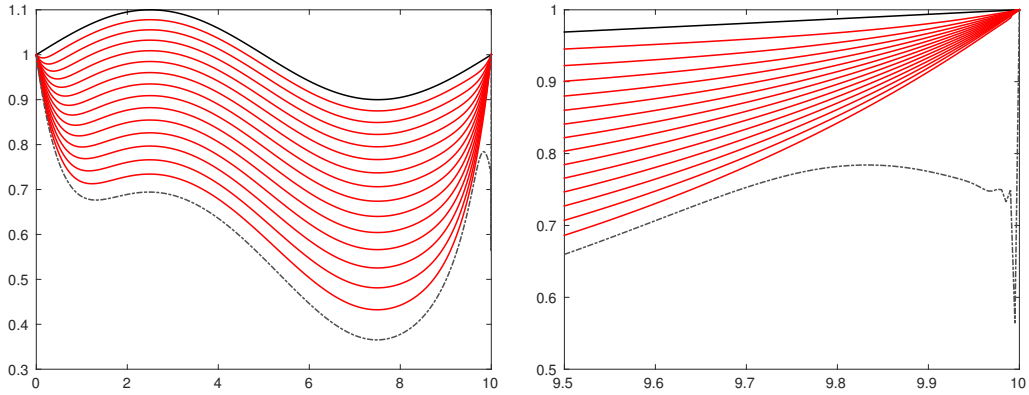


Figure 3.8: Instabilities using linearly stabilized SBDF3. With $p = 1.675$ and $\Delta t = 9.2 \times 10^{-4}$, we observe instabilities developing near the right-hand boundary of the gray dashed curve. The figure on the right is a zoom-in to the right-hand boundary.

With SBDF4, the issue is worse as the restriction is tighter. The results in Fig. 3.7 only appeared acceptable at coarse step-sizes because the low number of time steps did not allow for the instabilities to amplify to the extent that they dominate the solution. We conclude that linear stabilization with SBDF3 and SBDF4 is not recommended.

A natural follow-up to ask is if all third and fourth order IMEX schemes are unsuited for combination with linear stabilization. To this we provide a partial answer. Third order, three step schemes form a three parameter family, and fourth order, four step schemes form a four parameter family [2]. An extensive search through the parameter space so far has yielded no evidence of schemes with an unbounded $p$-parameter restriction.

This leaves us a number of competing second order methods to consider. Our next task is to compare the performance of our IMEX based schemes vs. the EIN method of [9].

## 3.4 Comparison of the Second Order Methods

Of the methods that we have proposed, only the second order variants allow for unconditional stability. Along with the EIN method, we have a total of five second order linearly stabilized schemes to consider. We now proceed with a comparison of the methods.

### 3.4.1 A 2D test problem

Let us consider as a test problem, the following nonlinear PDE from [33]:

$$u_t = \Delta(u^5), \quad 0 \leq x, y \leq 1, \quad t > 0, \tag{3.14a}$$

with initial and boundary conditions set so that the exact solution is

$$u(x,y,t) = \left(\frac{4}{5}(2t+x+y)\right)^{1/4}. \tag{3.14b}$$

Further assuming a uniform grid and centred differences in space, the eigenvalues of the Jacobian of the linearization are estimated to be in the interval

$$\left(-\frac{64}{h^2}(1+t), -16\pi^2(t+h)\right). \tag{3.15}$$

To (3.14), we propose to stabilize with a $p\Delta u$ term, i.e., replace (3.14a) with

$$u_t = \Delta(u^5) - p\Delta u + p\Delta u, \quad 0 \le x, y \le 1, \quad t > 0. \tag{3.16}$$

The parameter $p$ will then be chosen according to the ratio

$$\frac{p\lambda_{\mathcal{L}}}{\lambda_{\mathcal{N}}} \approx \frac{-8p/h^2}{-64(1+t)/h^2} = \frac{p}{8(1+t)}. \tag{3.17}$$

Let us remark on the importance of this test problem. In the analysis of (2.11), we performed a von Neumann analysis to obtain precise eigenvalue estimates that are independent of $t$. On the contrary, the interval (3.15) grows with $t$. Nevertheless, $p$ is never updated; it is chosen once and fixed at that value as we time step. Consequently at early times, $p$ may be substantially greater than necessary. Moreover, to compensate for the fact that the operator we introduced to stabilize is less stiff than the nonlinear term, we are forced to select a relatively large value for $p$. Such situations are common in applications making this an interesting problem to investigate performance and the behaviour of discretization errors.

### 3.4.2 Loss of accuracy with EIN

We test our second order methods on (3.16) with initial and boundary conditions set by (3.14b). We solve to time $T = 0.40$ with spatial grid size $\Delta x = \Delta y = h = 0.015$. As a reference solution, we use an explicit third order Runge-Kutta method with time step-size $\Delta t = 6.25 \times 10^{-6}$. (Typical explicit methods demand comparable step-sizes for stability.) With the linearly stabilized schemes, we will solve (3.16) with a variety of time step-sizes up to $\Delta t = 1.25 \times 10^{-2}$ and compute the max norm relative error.

Results of the numerical convergence test are plotted in Fig. 3.9. These paint a grim picture for the EIN method and for CNLF. We will first discuss the horrific performance of the EIN method, after which we comment on the relative performance of the IMEX based schemes.

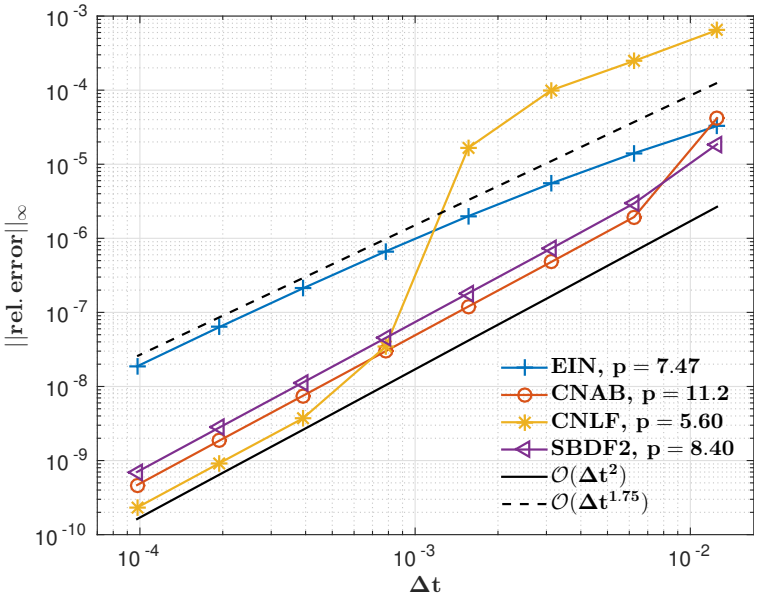


Figure 3.9: Numerical convergence study comparing second order methods. Test with mCNAB ($p = 9.97$) is omitted, but would otherwise overlap with CNAB.

Comparing the performance of EIN in Fig. 3.7 and Fig. 3.9, we observe a drastic reduction in the order of accuracy. In the former, EIN converged with second order accuracy and recorded, amongst the second order methods, the lowest error at any fixed $\Delta t$ (although an analysis of error vs. computing time would treat it less favorably). In the plot for the latter problem, we do not (yet) observe second order convergence. Indeed, further testing shows that the EIN method only begins to demonstrate full second order rate of convergence for time steps $\Delta t$ below $1 \times 10^{-5}$.

In fact, we argue that this behaviour can already be observed in the original paper by Duchemin and Eggers [9]. In their experiments with Hele-Shaw interface flow and with the Kuramoto-Sivashinsky equation, they fail to accurately reproduce the reference figures taken from prior publications [15, 17]. In both cases, a large value of $p$ was necessary to obtain the desired stability.

We offer an explanation. In the simple case of the modified test equation (2.13), we can apply the EIN method and derive the amplification factor (2.16). A series expansion at $z = 0$ yields

$$\xi_{EIN} = 1 + z + \frac{z^2}{2} + \frac{1}{2}(p - p^2)z^3 + \mathcal{O}(p^3 z^4). \tag{3.18}$$

This expression deviates from the exact solution, $\exp(z)$, in the $z^3$ term and exhibits a quadratic dependence on $p$. Thus, the leading error term will be large if $p$ is large and $\Delta t$ is not sufficiently small to control the error. That is exactly what is observed in Fig. 3.9.

On the other hand, the new multistep methods we have proposed all show only a linear dependence on $p$ under the same analysis, and do not suffer catastrophically when $p$ is large.

### 3.4.3 Amplification factors at infinity

In the last section, we discovered a deficiency of the EIN method. The leading error term, which is $\mathcal{O}(\Delta t^3)$, has a coefficient that is quadratic in $p$, which degrades the observed order of accuracy. Thus an effective linearly stabilized time stepping scheme should have an error coefficient that is linear with respect to $p$.

The form of the error constant does not explain the equally horrific performance of CNLF or the sharp dip in the observed convergence of CNAB near $\Delta t = 1 \times 10^{-2}$ (see Fig. 3.9). To posit an explanation, we think back to our discussion on stability and amplification factors. In that discussion, explicit schemes were said to be ill-suited to stiff problems because they are conditionally stable which imposes a severe step-size restriction. Although we have now guaranteed that our schemes are stable, the accumulation of slow decaying high frequency error modes can drive up the error and force us to use smaller time steps to adequately damp and get the expected convergence order.

To explore this aspect, we consider the method's amplification factor as $\lambda\Delta t \to -\infty$. For example, with the EIN method, consider the amplification factor in (2.16). As $z \to -\infty$, we find

$$\lim_{z\to-\infty} |\xi_{EIN}| = \left|\frac{p^2 - 3p + 2}{p^2}\right|. \tag{3.19}$$

For the multistep schemes, we first find the limiting expression of the amplification polynomial, and then take the larger of the absolute value of the two roots. For instance, with CNAB, we start from (3.8) and compute

$$\lim_{z\to-\infty} \frac{\Phi_2}{z} = -\frac{1}{2}p\xi^2 - \left(\frac{3}{2} - p\right)\xi - \frac{1}{2}p + \frac{1}{2}. \tag{3.20}$$

The amplification factor for CNAB is thus

$$\max\left\{\frac{\left|p - 1 + \sqrt{-2p+1}\right|}{p}, \frac{\left|1 - p + \sqrt{-2p+1}\right|}{p}\right\} \tag{3.21}$$

Fig. 3.10 shows the amplification factor (as $z \to -\infty$) for all of our second order schemes as well as SBDF1, and explains the behaviour of CNLF. It is known that CNLF is weakly damping at high frequencies and should not be used for diffusive problems [2]. This is equally true in the linear stabilization framework. Unless very small time steps are taken with this scheme, it gives very poor damping of high frequency error modes.

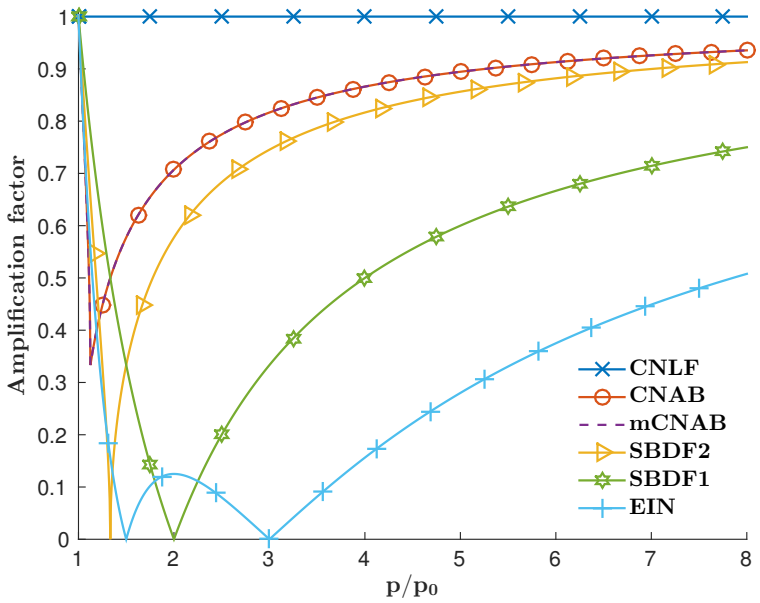


Figure 3.10: Amplification factors as $z \to -\infty$. The normalization along the horizontal axis is with respect to the lower limit of the parameter restriction of each scheme.

Of the schemes we consider, SBDF2 provides the most damping. CNAB (and mCNAB) may suffer from large errors at the coarsest time steps, but remains a useful alternative to SBDF2 as it has a smaller error constant as $\Delta t \to 0$. The scheme mCNAB was derived in [2] as an alternative to CNAB with stronger high frequency damping, however, in our framework, both schemes perform identically. Thus there is little reason to consider mCNAB over CNAB.

Finally, let us remark on the plotting range along the horizontal axis and how it pertains to the implementation of linearly stabilized schemes. First note that we have chosen to plot in a normalized parameter space. There are several reasons for this, but the most compelling arises from how we use these schemes. Implementation of linearly stabilized schemes start by choosing $p$ to satisfy the constraint

$$\frac{p\lambda_{\mathcal{L}}}{\lambda_{\mathcal{N}}} \geq p_0 \iff \frac{p}{p_0} \geq \frac{\lambda_{\mathcal{N}}}{\lambda_{\mathcal{L}}}, \tag{3.22}$$

where $p_0$ is the lower limit of the parameter restriction that guarantees unconditional stability, and where $\lambda_{\mathcal{L}}, \lambda_{\mathcal{N}}$ are the largest absolute eigenvalues of linear and nonlinear operators. The quantity $\lambda_{\mathcal{N}}/\lambda_{\mathcal{L}}$ is independent of the choice of time stepping method and if this quantity were overestimated in our solution procedure, the effect on $p/p_0$ is equal irrespective of the scheme. Note also that we have plotted over a large range of parameter values. The reason is again rooted in the implementation. In our derivations, we operate as if we can access the exact eigenvalues at every time step. This is impractical and impossible. In practice, the ratio of eigenvalues will be an overestimate, and the value of $p$ is fixed throughout

the time evolution (or at least for a great number of time steps.) Therefore it is necessary to chart the behaviour of the methods over a wide range of $p$.

# Chapter 4

# Higher Order with Exponential Integrators

The investigation with multistep schemes left us with a major question. Since the linearly stabilized schemes derived from SBDF3, SBDF4 were shown to be unsuitable for practical use, is it possible to construct practical high order linearly stabilized time stepping methods? In this chapter, we consider two methods coming from the class of exponential integrators. We will demonstrate that the second and fourth order exponential Runge-Kutta from Cox and Matthews [7] work well within our linear stabilization framework.

## 4.1 Exponential Runge-Kutta

Consider the ODE

$$\frac{du}{dt} = \mathcal{N}(u) + \mathcal{L}u. \tag{4.1}$$

Exponential time differencing, or exponential integrators, is a family of time stepping methods that treats the linear part exactly, and approximates the nonlinear part by some suitable quadrature formula. As an example, the exponential Euler method has the formula

$$u^{n+1} = e^{\Delta t \mathcal{L}} u^n + \mathcal{L}^{-1}(e^{\Delta t \mathcal{L}} - 1)\mathcal{N}(u^n). \tag{4.2}$$

This is a first order accurate exponential integrator.

Our investigation covers explicit exponential Runge-Kutta methods only. This family of one-step methods have the form

$$u^{n+1} = e^{\Delta t\mathcal{L}} u_n + \Delta t \sum_{i=1}^{s} b_i(\Delta t\mathcal{L})\mathcal{N}(U^{n,i}) \tag{4.3a}$$

$$U^{n,i} = e^{c_i \Delta t\mathcal{L}} u^n + \Delta t \sum_{j=1}^{i-1} a_{ij}(\Delta t\mathcal{L})\mathcal{N}(U^{n,j}), \tag{4.3b}$$

and can be presented in the familiar Butcher tableau:

$$\begin{array}{c|cccc} c_1 & & & & \\ c_2 & a_{21} & & & \\ \vdots & \vdots & \ddots & & \\ c_s & a_{s1} & \cdots & a_{s,s-1} & \\ \hline & b_1 & \cdots & b_{s-1} & b_s \end{array}. \tag{4.4}$$

Note that we have suppressed the argument, but these are indeed functions of $\Delta t\mathcal{L}$. In particular, we focus on the second and fourth order exponential Runge-Kutta formulas of Cox and Matthews [7];

$$\begin{array}{c|cc} 0 & & \\ 1 & \varphi_{1,2} & \\ \hline & \varphi_1 - \varphi_2 & \varphi_2 \end{array}, \tag{4.5}$$

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & \frac{1}{2}\varphi_{1,2} & & & \\ 1/2 & 0 & \frac{1}{2}\varphi_{1,3} & & \\ 1 & \frac{1}{2}\varphi_{1,3}(\varphi_{0,3} - 1) & 0 & \varphi_{1,3} & \\ \hline & \varphi_1 - 3\varphi_2 + 4\varphi_3 & 2\varphi_2 - 4\varphi_3 & 2\varphi_2 - 4\varphi_3 & 4\varphi_3 - \varphi_2 \end{array}, \tag{4.6}$$

where

$$\varphi_{k+1}(z) = \frac{\varphi_k(z) - 1/k!}{z}, \quad \varphi_0(z) = \exp(z), \quad \text{and} \quad \varphi_{i,j}(z) = \varphi_i(c_j z). \tag{4.7}$$

We refer to this pair of exponential Runge-Kutta methods as ETDRK2 and ETDRK4, respectively.

## 4.2 Linearly stabilized ETDRK2 and ETDRK4

In the last chapter, we identified some useful criteria for quickly assessing the practicality of newly constructed linearly stabilized methods.

First, we apply the schemes (4.5) and (4.6) to the modified test equation (2.13) and imposed unconditional stability. We are only interested in schemes with an unbounded parameter restriction. For ETDRK2 and ETDRK4, with the help of the computer algebra system, MAPLE™, we determined the parameter restriction to be $[1/2, \infty)$ in both cases. In Figs. 4.1 and 4.2 are stability contour plots that support this claim.

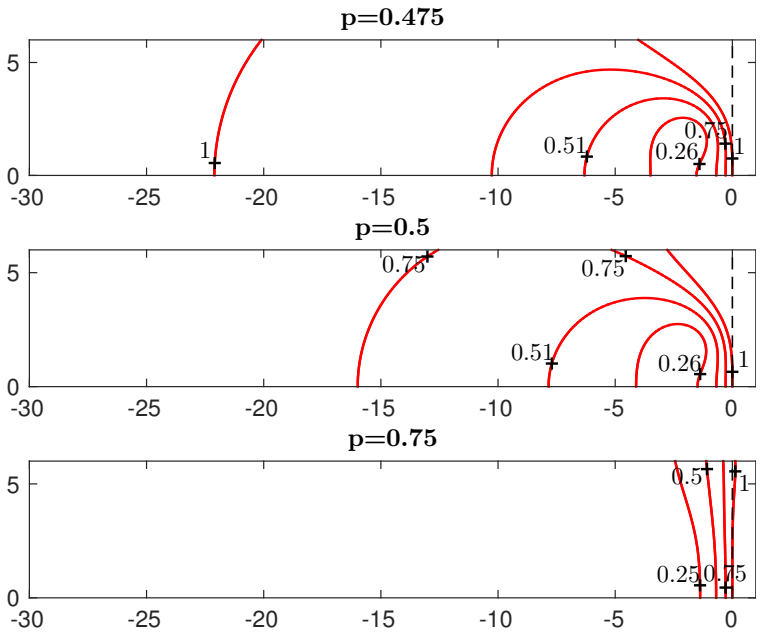


Figure 4.1: Stability contours for ETDRK2 at various $p$.

Two other factors were considered that affect the performance. The first is the amplification factor as $z \to -\infty$, and the second is a proxy to the error constant of the numerical scheme. The former is plotted in Fig. 4.3 for both ETDRK2 and ETDRK4. It shows that the ETDRK schemes provide strong damping as $z \to -\infty$ for a wide range of $p$ and may be a good candidate for taking large time steps. For the proxy to the error constant, we have previously considered series expansion at $z = 0$ of the amplification factors:

$$\xi_{ETDRK2} = 1 + z + \frac{z^2}{2} + \left(-\frac{1}{4}p^2 + \frac{5}{12}p\right) z^3 + \cdots, \tag{4.8}$$

$$\xi_{ETDRK4} = \exp(z) + \left(\frac{1}{576}p^4 - \frac{11}{576}p^3 + \frac{29}{720}p^2 - \frac{1}{32}p + \frac{1}{120}\right) z^5 + \cdots. \tag{4.9}$$

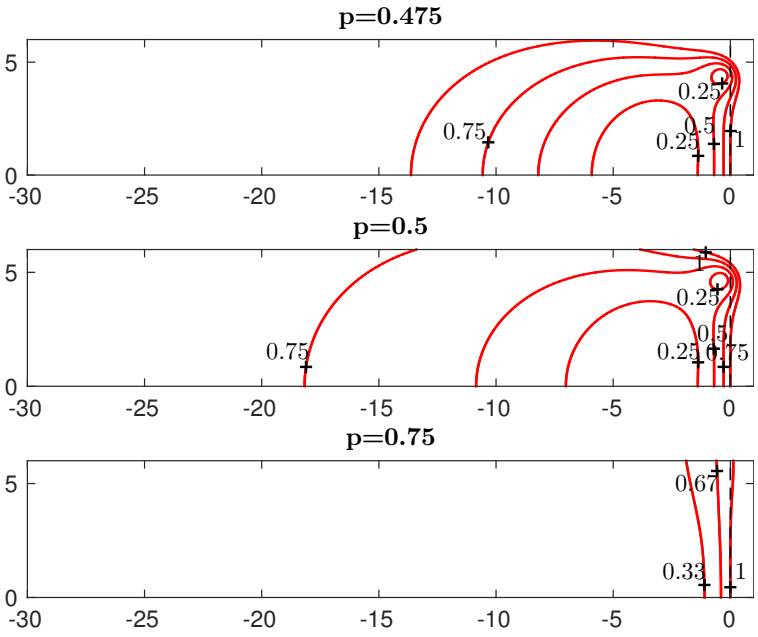


Figure 4.2: Stability contours for ETDRK4 at various $p$.

Let us take this one step further by taking the difference with the exact solution, $\exp(z)$. We find

$$\delta_1(p) = |\exp(z) - \xi_{ETDRK2}| = \left|\frac{1}{4}p^2 - \frac{5}{12}p + \frac{1}{6}\right|, \tag{4.10}$$

$$\delta_2(p) = |\exp(z) - \xi_{ETDRK4}| = \left|\frac{1}{576}p^4 - \frac{11}{576}p^3 + \frac{29}{720}p^2 - \frac{1}{32}p + \frac{1}{120}\right|, \tag{4.11}$$

and for the EIN method we have

$$\delta_3(p) = |\exp(z) - \xi_{EIN}| = \left|\frac{1}{2}p^2 - \frac{1}{2}p + \frac{1}{6}\right|. \tag{4.12}$$

Recall from before that the observed convergence rate of EIN was less than we expected from the truncation error. We reasoned that the source of this discrepancy was that the error constant is large whenever $p$ is large. Thus, the step-size had to be chosen to be very small before observing second order convergence.

The situation is similar with the ETDRK schemes. The error term has quadratic and quartic polynomials in $p$ for ETDRK2 and ETDRK4 respectively, and thus we expect these schemes to fare poorly when $p$ is large. Nonetheless, Chapter 5 demonstrates that these schemes can outperform SBDF2 and CNAB when $p$ is small and the step-size is coarse. To this end, Fig. 4.4 provides a plot that shows $\delta_1$, $\delta_2$, and $\delta_3$ against $p$. Also plotted are the analogous expressions for SBDF2 and CNAB. As was suggested, there may be a range of $p$

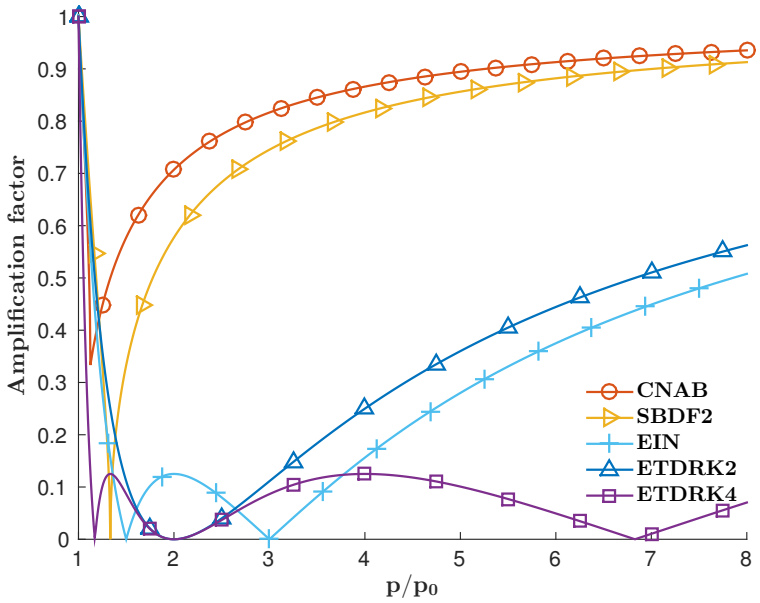


Figure 4.3: Amplification factors as $z \to -\infty$ with ETDRK schemes. Included also for the purpose of comparison are some second order schemes from the last chapter. The normalization along the horizontal axis is with respect to the lower limit of the parameter restriction of each scheme.

for which these ETDRK schemes are workable. A careful error analysis on fully nonlinear problems would be of some interest and is left as future work.

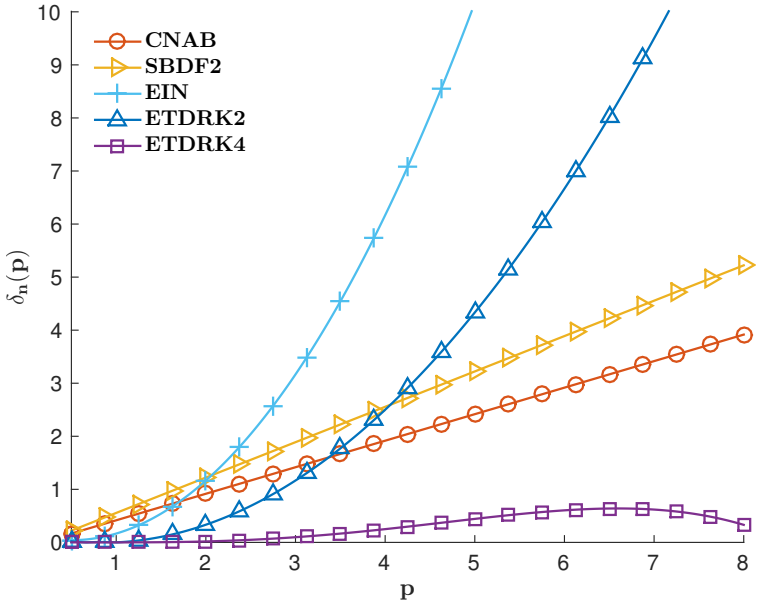


Figure 4.4: Error constant and its dependence on $p$.

### 4.2.1 Stable evaluation of the matrix exponential and related functions

Finally, we wish to discuss briefly the matter of implementing ETDRK schemes.

Section 4.1 presents two exponential Runge-Kutta methods in tableau form. The entries of these tableaus are functions of the operator, $\Delta t\mathcal{L}$. For example, to implement ETDRK2, we must compute the functions

$$\varphi_1(c\Delta\mathcal{L}) = (c\Delta t\mathcal{L})^{-1}(\exp(c\Delta t\mathcal{L}) - 1), \tag{4.13}$$

$$\varphi_2(c\Delta\mathcal{L}) = (c\Delta t\mathcal{L})^{-2}(\exp(c\Delta t\mathcal{L}) - 1 - c\Delta t\mathcal{L}), \tag{4.14}$$

for $c \in [0, 1]$, or be able to efficiently evaluate the related matrix-vector multiplications without explicit construction.

Difficulties with the evaluation of the matrix exponential and related functions of the form (4.7) are well-known and well-studied, eg. [19, 12, 13, 14]. Thus widespread adoption of exponential time differencing methods requires the development of stable and efficient algorithms for computing the matrix exponential.

In our examples, we follow the direction of Kassam and Trefethen [17]. They take a contour integral approach to the evaluation of functions in the form of (4.7) coupled with the trapezoidal rule for fast, accurate, and stable computations. To avoid the unpleasantness of boundary conditions (and the subsequent complications), we choose our domain to be periodic when using ETDRK2 or ETDRK4.

Lastly, let us mention that there are other popular approaches. Amongst these are methods based on scaling and squaring, Padé approximants [19, 13], and Krylov subspace methods [14, 28, 29].

# Chapter 5

# Numerical Experiments

This chapter is entirely devoted to solving stiff PDEs prevalent in a number of fields. The experiments we demonstrate will fall under three categories: image inpainting, interface motion, and phase separation. For each problem, we will give the PDE model and then discuss how we stabilize and select the parameters. Our experiments will show the feasibility of these schemes in 2D and 3D.

Before proceeding further, we would like to make a few notes on our implementation of these schemes. As stated from the outset, our goal is to provide simple, accurate, and efficient time stepping methods for nonlinear PDEs. Too often, new efficient methods are sufficiently complicated that users resort to simple explicit schemes that require lengthy computing time rather than invest an indeterminate amount of time understanding, implementing, and debugging.

In our implementation, the choice of $p$ is fixed throughout the time evolution. Alternately, one may want to adapt $p$ as the solution evolves to avoid overestimates of $p$ that could lead to larger errors. However, we do not pursue this here. So while our theory speaks of approximating the eigenvalues of the linearized system, we do not incur this cost in our computations.

Note that a static value of $p$ offers the advantage that the linear system to be solved is the same at each time step, i.e. the matrix to be inverted is static. Any expensive preprocessing/factorizing of this matrix needs only be done once.

## 5.1 Image Inpainting

Image inpainting is the task of repairing corrupted images and damaged artwork [3]. In the inpainting examples to follow, the user identifies in the image the region to be inpainted, and from there, a PDE model is evolved to fill-in the inpainting region using the neighbouring information.

Two PDE models are selected. The first is a second order model from Shen and Chan [26], and the second is a recent fourth order model from Schönlieb and Bertozzi [25],

$$u_t = \nabla \cdot \left( \frac{\nabla u}{\sqrt{|\nabla u|^2 + \epsilon^2}} \right) + \lambda_D (u_0 - u), \tag{5.1}$$

$$u_t = -\Delta \nabla \cdot \left( \frac{\nabla u}{\sqrt{|\nabla u|^2 + \epsilon^2}} \right) + \lambda_D (u_0 - u). \tag{5.2}$$

We refer to these as TV inpainting and TV-$H^{-1}$ inpainting.

The solution, $u$, is the inpainted image we seek. The quantity, $u_0$, is the initially corrupted image, and $\epsilon > 0$ is a regularization parameter. Denoting the image domain $\Omega$, and the inpainting region, $D$, we then define $\lambda_D$ as

$$\lambda_D(x) = \begin{cases} \lambda_0, & x \in \Omega \setminus D \\ 0, & \text{otherwise,} \end{cases} \tag{5.3}$$

for some $\lambda_0 > 0$.

As for initial conditions, we have two vandalized images to be restored. The first is Fig. 5.1 where we have overwritten with text a photograph of a sea turtle. We would like to restore the photograph by removing the text. The second is Fig. 5.2, where the fox figure requires removal. Although this may look simpler, it is in fact a more challenging scenario because the thickness of the inpainting region requires correctly extending level lines over long distances [25].

The image is restored in a channel-by-channel manner. As a stopping criteria, we look at the difference in successive images,

$$\frac{\left\| u^{n+1} - u^n \right\|_2}{\left\| u^{n+1} \right\|_2} < \text{tol}, \tag{5.4}$$

where the tolerance is set at $\text{tol} = 9 \times 10^{-5}$. In addition, we also set a maximum of 500 iterations per channel. We report the time step-size and the total number of iterations used for each scheme.

Finally, we note that the spatial discretization is by second order centred differences with uniform spacing, $h$, in both $x$ and $y$.


Figure 5.1: Photograph of a sea turtle overwritten with text.

Figure 5.2: Photograph of a bullfinch vandalized with a cartoon fox.

### 5.1.1 TV inpainting

We first show that the TV inpainting model can easily be handled by our methods. In this model, there are two terms on the right-hand side, both potentially stiff. The second term is stabilized by adding and subtracting $-p_0\lambda_0 u$, where $p_0$ is the minimum value required for unconditional stability from applying the time stepping method to the modified test equation. For the first term, we stabilize by adding and subtracting $p_1\Delta u$. To determined

$p_1$, we bound the first term as

$$\begin{aligned}\nabla \cdot \left( \frac{\nabla u}{\sqrt{|\nabla u|^2 + \epsilon^2}} \right) &= \frac{u_{xx}(u_y^2 + \epsilon^2) + u_{yy}(u_x^2 + \epsilon^2)}{(u_x^2 + u_y^2 + \epsilon^2)^{3/2}} - \frac{2u_x u_y u_{xy}}{(u_x^2 + u_y^2 + \epsilon^2)^{3/2}} \\ &\leq \frac{(u_{xx} + u_{yy})(u_x^2 + u_y^2 + \epsilon^2)}{(u_x^2 + u_y^2 + \epsilon^2)^{3/2}} + \frac{(u_x^2 + u_y^2 + \epsilon^2)u_{xy}}{(u_x^2 + u_y^2 + \epsilon^2)^{3/2}} \\ &= \frac{u_{xx} + u_{yy} + u_{xy}}{\sqrt{u_x^2 + u_y^2 + \epsilon^2}}.\end{aligned} \tag{5.5}$$

We then consider the auxiliary equation $u_t = u_{xx} + u_{yy} + u_{xy}$ discretized by centred differences in space and forward Euler in time and apply a von Neumann analysis with $u_{jk}^n = \xi^n \exp(i\omega_1 jh) \exp(i\omega_2 kh)$ to get

$$\frac{\xi - 1}{\Delta t} = \frac{1}{h^2}(-4 + 2\cos(\omega_1 h) + 2\cos(\omega_2 h) - \sin(\omega_1 h)\sin(\omega_2 h)) \geq -\frac{8}{h^2}. \tag{5.6}$$

Combined with the assumption of the extreme case, $\sqrt{u_x^2 + u_y^2 + \epsilon^2} \geq \epsilon$, we set $p_1$ according to

$$p_1 \frac{8/h^2}{8/(\epsilon h^2)} \geq p_0 \iff p_1 \geq \frac{1}{\epsilon} p_0. \tag{5.7}$$

The images restored by TV inpainting and SBDF1, SBDF2, and CNAB are shown in Fig. 5.3. Parameters were set as $\epsilon = 0.10$ and $\lambda_0 = 20$. Table 5.1 provides the iteration count required with each method. We observe lower iteration counts and better efficiency with second order methods.

Table 5.1: Iteration counts for TV image restoration.

| | Sea Turtle | | Bullfinch | |
|---|---|---|---|---|
| | $\Delta t$ | Iterations | $\Delta t$ | Iterations |
| SBDF1 | 0.50 | 141 | 0.33 | 393 |
| SBDF2 | 0.18 | 115 | 0.22 | 168 |
| CNAB | 0.16 | 104 | 0.18 | 195 |

### 5.1.2 TV-H$^{-1}$ inpainting

For TV-H$^{-1}$ inpainting, we stabilize (5.2) using

$$u_t = -\Delta \nabla \cdot \left( \frac{\nabla u}{\sqrt{|\nabla u|^2 + \epsilon^2}} \right) + \lambda_D(u_0 - u) + p_1 \Delta^2 u + p_0 \lambda_0 u - p_1 \Delta^2 u - p_0 \lambda_0 u. \tag{5.8}$$

Figure 5.3: Image restoration by TV inpainting using a second order linearly stabilized time stepping method.

We note $p_0$ is chosen exactly as in TV inpainting. Determination of $p_1$ applies the bound set in (5.5) and (5.6) to get

$$p_1 \frac{(8/h^2)^2}{(8/h^2)(8/(\epsilon h^2))} \geq p_0 \iff p_1 \geq \frac{1}{\epsilon} p_0, \tag{5.9}$$

which is also the same as TV inpainting.

Interestingly, in the same paper where they propose (5.2) for image inpainting, the authors offer exactly (5.8) and time stepping with SBDF1 as the solution algorithm. In Table 5.2, we chart again the iteration counts required for each of SBDF1, SBDF2, CNAB, with parameters $\epsilon = 0.10$ and $\lambda_0 = 30$. Once again, the second order methods are an improvement over the first order method, with the improvement especially notable in the more difficult example with the bullfinch.

As a final note, we mention other interesting developments [4, 21, 22]. In these papers, a number of image restoration models have been proposed involving high order derivatives interacting nonlinearly. It would be of interest to compare the effectiveness of our schemes with the methods proposed in these papers.

Figure 5.4: Image restoration by TV-$H^{-1}$ inpainting using a second order linearly stabilized time stepping method.

Table 5.2: Iteration counts for TV-$H^{-1}$ image restoration.

| | Sea Turtle | | Bullfinch | |
|---|---|---|---|---|
| | $\Delta t$ | Iterations | $\Delta t$ | Iterations |
| SBDF1 | 0.65 | 143 | 0.30 | 1002 |
| SBDF2 | 0.12 | 119 | 0.54 | 401 |
| CNAB | 0.10 | 108 | 0.64 | 347 |

## 5.2 Motion by Mean Curvature

In this section, we study the problem of interface evolution under mean curvature flow. The level set equation for motion by mean curvature is

$$u_t = \kappa \left|\nabla u\right| = \left|\nabla u\right| \nabla \cdot \left(\frac{\nabla u}{\left|\nabla u\right|}\right). \tag{5.10}$$

Our interest is in the time evolution of the interface, $\Gamma = \Gamma(t)$, described by the zero level set of the function $u$,

$$\Gamma(t) = \{x \in \mathbb{R}^d \mid u(x,t) = 0\}. \tag{5.11}$$

We will demonstrate the effectiveness of our schemes on examples similar to that of Smereka [30]. In [30], he uses what we call linearly stabilized SBDF1 to take large, stable time steps. Further, in the same paper, it was also suggested that Richardson extrapolation may be used to attain second order convergence, but was not implemented. We follow the example set in [30] and stabilize (5.10) with a Laplacian term, $p\Delta u$, to get

$$u_t = \kappa \left|\nabla u\right| - p\Delta u + p\Delta u. \tag{5.12}$$

An analysis similar to (5.5) and (5.6) yields $p \geq p_0$ to be sufficient for unconditional stability.

Let us point out a key difference between this problem and the inpainting problem of the previous section. In the inpainting problem, the system was to be driven to steady state. As such, we were afforded a range of time step-sizes where the solution method was computationally efficient. Indeed, the step-size did not affect the visual quality. For mean curvature flow, computing time and accuracy are directly related to the choice of step-size. Thus we are interested in large step-sizes only if an acceptable level of accuracy is maintained.

### 5.2.1 Shrinking dumbbell in 2D and 3D

Our first example is the evolution of a dumbbell-shaped curve in 2D. Fig. 5.5 plots the motion of this curve under mean curvature flow. From the initial dumbbell shape, the corners smooth out and then the curve shrinks as shown, until it eventually collapses down to a point. These reference solutions were generated to time $T = 1.25$ with an explicit Runge-Kutta method and a small time step-size. On a periodic grid of size $256 \times 512$ and approximating the spatial derivatives using second order centred differences, the number of time steps needed for stability is $\mathcal{O}(10^4)$.

In Fig. 5.6, we show the convergence of SBDF1, SBDF2, CNAB, EIN, ETDRK2, ETDRK4 for the same problem, but plotting in the positive quadrant only. As shown, each scheme is stable and convergent using much fewer than $10^4$ time steps. However, it is im-

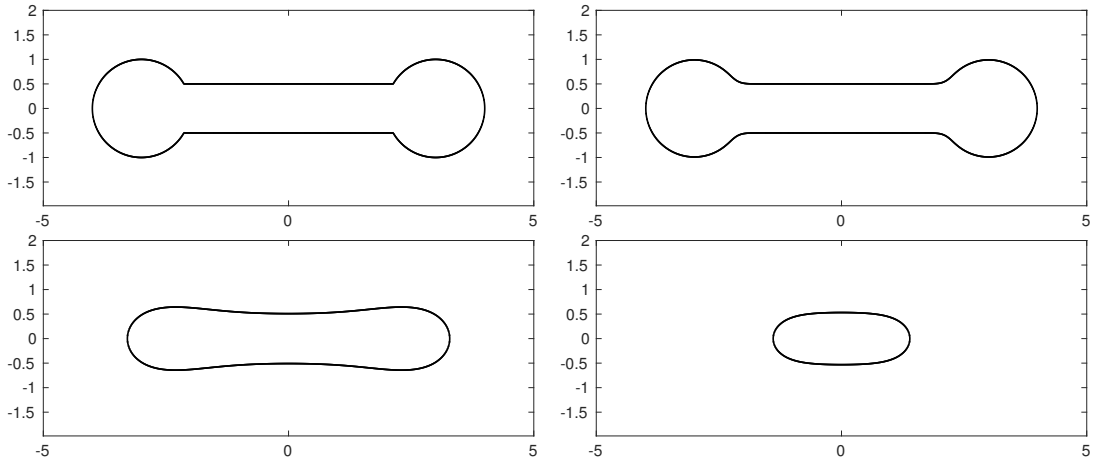

Figure 5.5: Mean curvature flow of a dumbbell-shaped curve in 2D. From the top left to the bottom right, the plots show the evolution at times $t = 0$, 0.01, 0.5, 1.25.

portant to note that amongst the schemes, the number of time steps needed to achieve an acceptable level of accuracy ranges from 50 to 800.

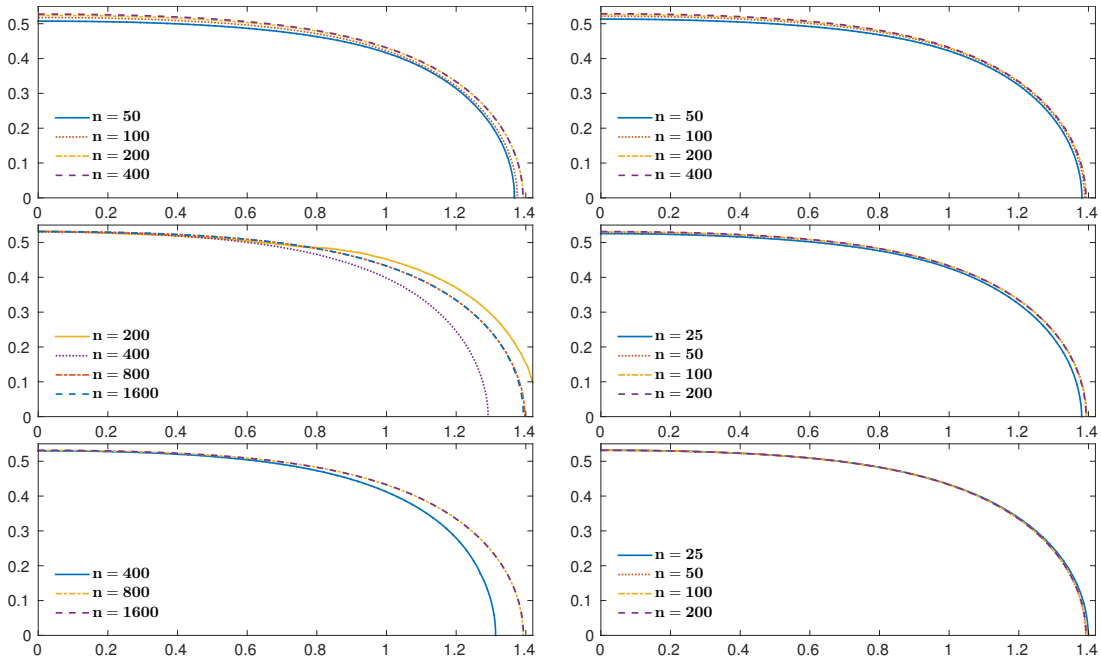


Figure 5.6: Convergence of various linearly stabilized time stepping methods to a dumbbell-shaped curve in 2D evolved under mean curvature flow. In the column to the left from top to bottom, we have SBDF1, SBDF2, and CNAB. In the column to the right from top to bottom, we have EIN, ETDRK2, and ETDRK4. Within each plot, each curve is the solution using the number of time steps, $n$, as indicated.

The most disappointing of these is CNAB, for which a minimum of 400 time steps was needed to get a solution profile that fits within the frame we have chosen. Furthermore, it

needs close to 800 time steps to present a solution competitive with the other methods in accuracy. SBDF2 likewise fared poorly in that regards, needing close to 800 time steps for a solution competitive with the other methods in accuracy.

We now take a more careful look at the ETDRK schemes and the EIN method. Recall that these are the schemes that have strong damping as $z \to -\infty$ (see Fig. 4.3). Zoom-ins to the solution by these methods are presented in Figs. 5.7 and 5.8. These show that the ETDRK schemes offer good accuracy and convergence using just 50 time steps. The EIN method, however, is again hampered by slow convergence. This is especially clear in Fig. 5.7. If we further factor in computing time, then both CNAB and SBDF2 outperform the EIN method.

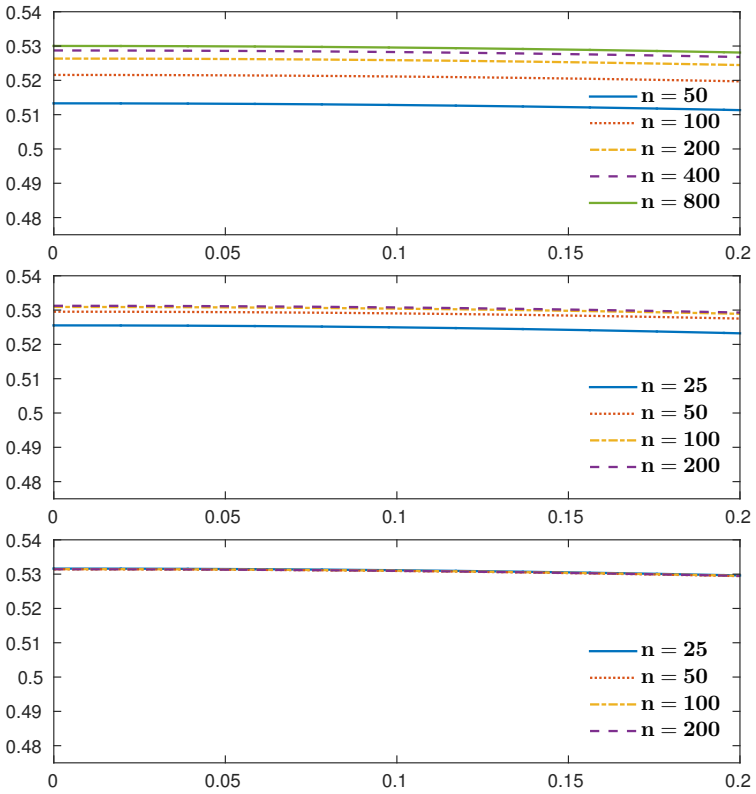


Figure 5.7: Zoom-in over $[0.0, 0.20] \times [0.475, 0.54]$. From top to bottom we have the EIN method, ETDRK2, and ETDRK4. We see slow convergence of the EIN method, and good convergence of the ETDRK schemes.

Next, we take this example into 3D. Here, the advantages of our schemes are further magnified. In 2D, one could argue that the computations can be completed using standard explicit schemes within reasonable computing times. In 3D, doing so may require trade offs in the grid size, or computing only over short time periods.

Setting the initial condition to be the dumbbell-shaped curve of the top left image in Fig. 5.9, the curve is then evolved by mean curvature flow. We use a periodic grid of size $256 \times 128 \times 128$ and solve to time $T = 0.75$. With forward Euler, we needed 3000 time steps

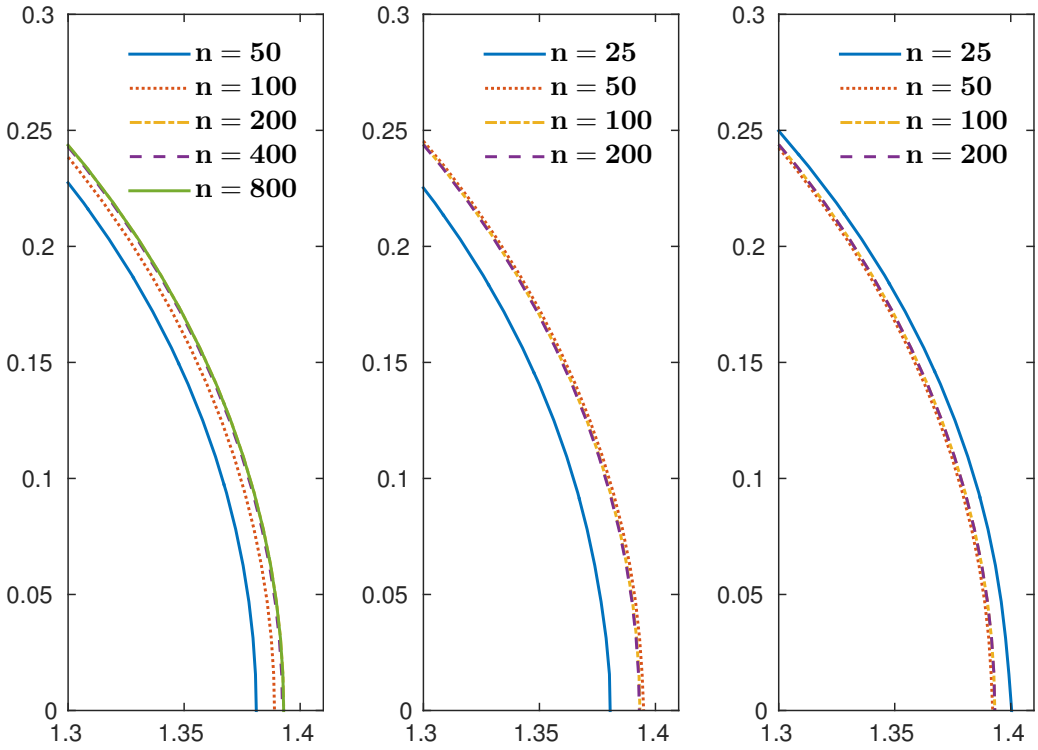


Figure 5.8: Zoom-in over $[1.30, 1.41] \times [0.0, 0.30]$. From left to right we have the EIN method, ETDRK2, and ETDRK4. We see slow convergence of the EIN method, and good convergence of the ETDRK schemes.

for stability, which corresponds to over 28 minutes in MATLAB 2014b on an Intel®Core™i5-4570 CPU@3.20GHz workstation running Linux. With the linearly stabilized ETDRK2, we solved the same problem using 75 time steps in 103 seconds.

### 5.2.2 Anisotropic mean curvature motion

So far, the motion law considered may be more appropriately stated as isotropic mean curvature motion. In [20], Oberman et at., present a method for anisotropic mean curvature flow:

$$u_t = (\gamma(\omega) + \gamma''(\omega)) \, |\nabla u| \, \nabla \cdot \left( \frac{\nabla u}{|\nabla u|} \right), \tag{5.13}$$

where $\omega = \arctan(u_y/u_x)$, and

$$\gamma(\omega) = \gamma_n(\omega) = \frac{1}{n^2+1}(n^2 + 1 - \sin(n\omega)), \quad \text{for } n = 0, 2, 4, 8. \tag{5.14}$$

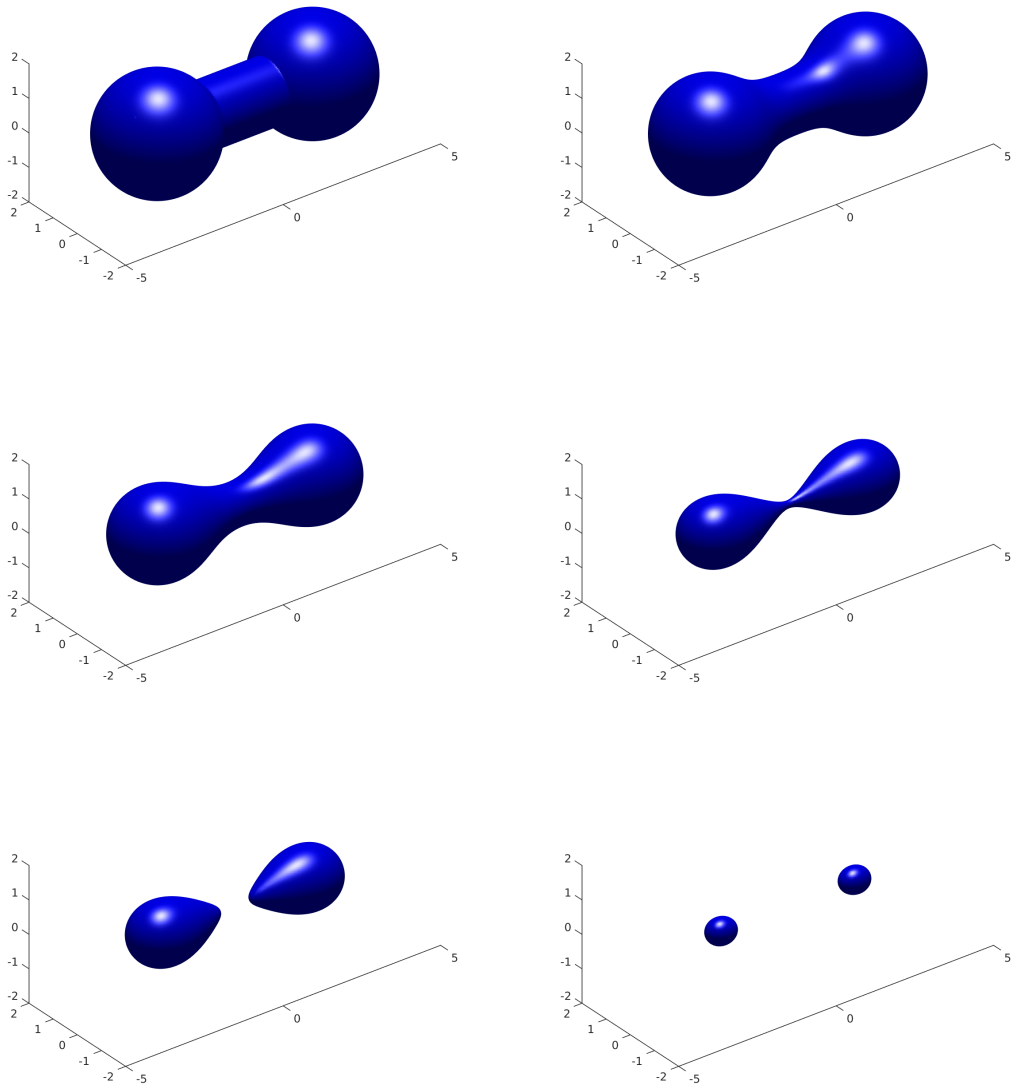

Figure 5.9: Mean curvature flow of a dumbbell-shaped curve in 3D. From the left to right, top to bottom, the plots show the evolution at times $t = 0$, 0.10, 0.30, 0.525, 0.55, 0.75.

Under isotropic mean curvature motion, a simple, smooth closed contour in 2D has a circular limiting shape as it reduces to a point. Under (5.13) and (5.14), the limiting shape will have $n$-fold rotational symmetry.

For our methods, the added factor of $\gamma(\omega) + \gamma''(\omega)$ presents no additional difficulty at all. Again, we can stabilize with $p\Delta u$, with $p = (1 + (n^2 - 1)/(n^2 + 1))p_0$. Setting $n = 4$, and on a periodic grid of size $256 \times 256$, we take 500 time steps with a linearly stabilized ETDRK2 to produce the solution presented in Fig. 5.10.

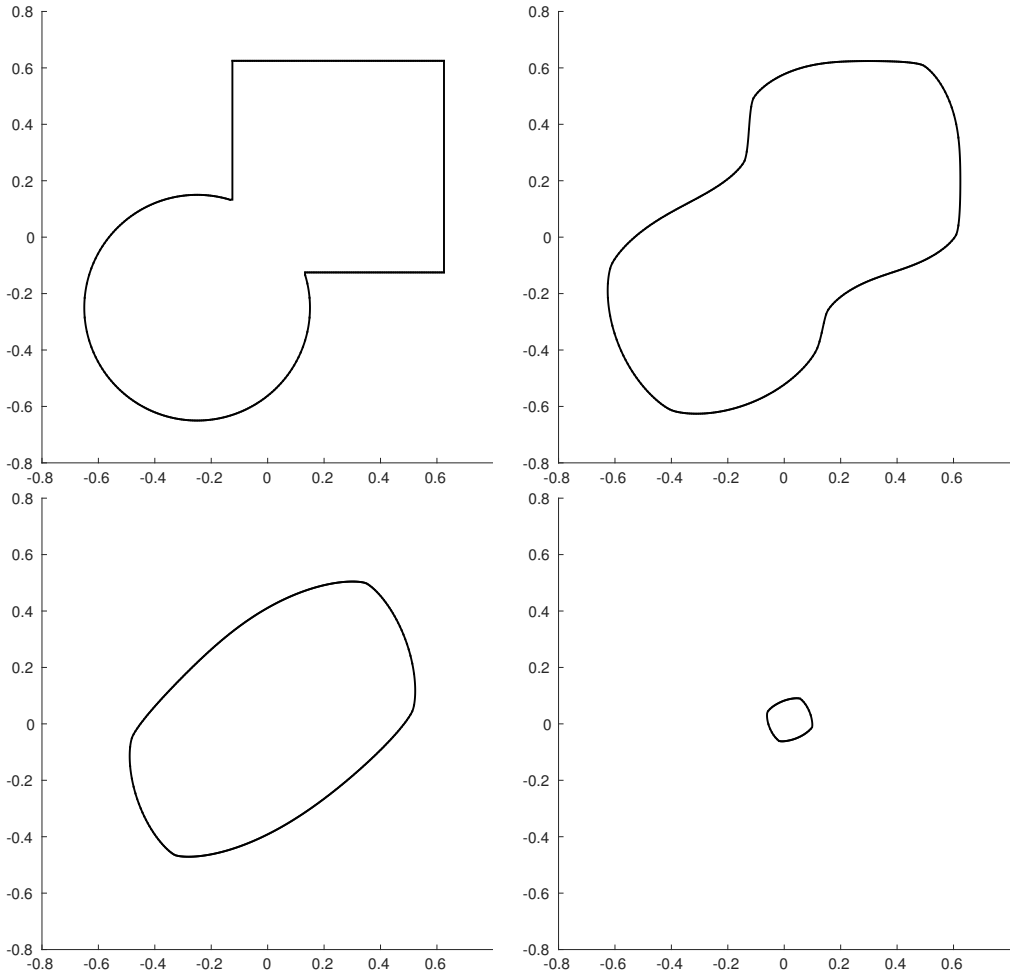


Figure 5.10: Anisotropic mean curvature flow in 2D. The plots show the evolution of the curve at times $t = 0$, 0.01, 0.06, 0.16. The initial curve smooths and shrinks to a curve exhibiting four-fold symmetry as it collapses to a point.

## 5.3 Phase Separation

For our last batch of experiments, we will define and evolve phase separation models on rectangular grids in 2D, and on surfaces in 3D using the closest point method [23, 18].

### 5.3.1 Cahn-Hilliard on surfaces using the closest point method

In this section, the model equation is

$$u_t = -\epsilon^2 \Delta_{\mathcal{S}}^2 u + \Delta_{\mathcal{S}}(u^3 - u), \tag{5.15}$$

where $\epsilon > 0$ is a small parameter that determines interfacial thickness, $\mathcal{S}$ is a surface in $\mathbb{R}^3$, and the solution $u$ indicates the domains of the separating binary fluid.

We assume initially that the solution is well-mixed and thus the initial conditions are chosen uniformly at random from $[-\epsilon, \epsilon]$. As time progress, the initially well-mixed solution aggregates into homogeneous zones separated by transition layers of thickness $\mathcal{O}(\epsilon)$. Within the homogeneous zones, $u$ takes on the value 1 or $-1$.

To discretize the surface operators, we use the closest point method of Ruuth and Merriman [23]. Their approach is to extend the solution off the surface into the embedding space at each time step in a way that allows them to replace the surface operators with their Cartesian analogs. It then becomes a matter of selecting from standard finite difference methods for the discretization of the spatial operators. We also mention the work of Macdonald and Ruuth [18] that provides for implicit time stepping with the closest point method. See [1] for open access closest point method software.

To use a linearly stabilized method for time stepping, we must determine $p$. To determine $p$, we linearize (5.15). Substituting with $u = u^n + \delta v$, we have

$$\begin{aligned}(u^n + \delta v)_t &= -\epsilon^2 \Delta_{\mathcal{S}}^2 (u^n + \delta v) + \Delta_{\mathcal{S}}((u^n + \delta v)^3 - u^n - \delta v) \\ &= -\epsilon^2 \delta \Delta_{\mathcal{S}}^2 v + \delta^3 \Delta_{\mathcal{S}} v^3 + 3\delta^2 u^n \Delta_{\mathcal{S}} v^2 + 3\delta (u^n)^2 \Delta_{\mathcal{S}} v - \delta \Delta_{\mathcal{S}} v.\end{aligned} \tag{5.16}$$

Differentiating with respect to $\delta$ and then setting $\delta = 0$, we arrive at the linearized equation

$$v_t = -\epsilon^2 \Delta_{\mathcal{S}}^2 v + (3(u^n)^2 - 1)\Delta_{\mathcal{S}} v. \tag{5.17}$$

Thus if we are to stabilize with $p\Delta_{\mathcal{S}} u$, then we may choose $p$ as

$$p \geq (3\bar{u}^2 - 1)p_0, \tag{5.18}$$

where $\bar{u} = \sup_{\mathcal{S}} |u^n|$. For simplicity, we use $p = 2p_0$ as $u$ takes values in $[-1, 1]$.

Figs. 5.11 and 5.12 show the solution to (5.15) on two different surfaces at time $T = 100$. In both examples, the spatial grid is uniform, with $h = 0.14$ for the torus and $h = 0.075$ for

the cow. We then set $\epsilon = 2h$. Time stepping is done with SBDF2 and for the time step-size, we have chosen $\Delta t = 0.5h$ for the torus and $\Delta t = 1$ for the cow.

The latter $\Delta t = 1$ condition is chosen to emphasize that we may choose the step-size independent of stability concerns. For the Cahn-Hilliard equation, the dynamics may be broadly split into two stages [32]. The first is a fast coarsening stage in which a pattern of internal layers is formed from the initial state over an $\mathcal{O}(1)$ time interval. This is followed by an exponentially slow coarsening during which the internal layers may collapse together to reach a stable/energy minimizing state. It is during this slow phase that our schemes provide the greatest benefit.

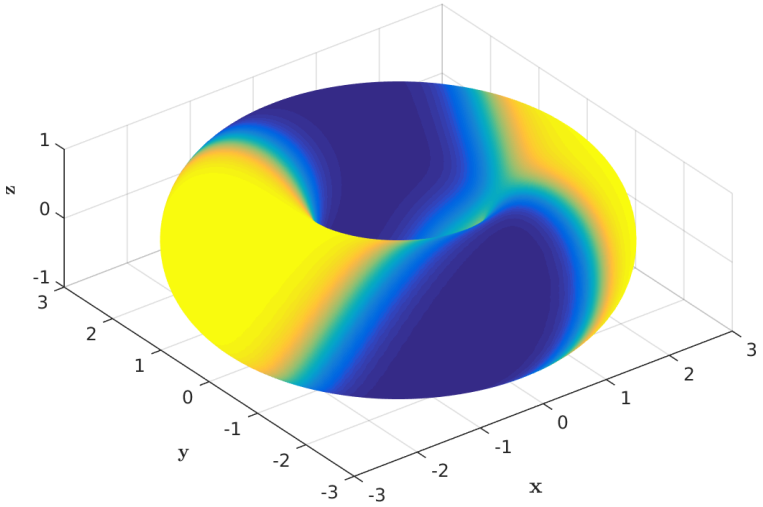


Figure 5.11: Cahn-Hilliard on a torus.

### 5.3.2 A modified Cahn-Hilliard with nonlocal interactions

For our last example, we solve

$$u_t = -\epsilon^2 \Delta^2 u + \Delta(u^3 + 3mu^2 - (1 - 3m^2)u) - u, \quad (x, y) \in [-2\pi, 2\pi]^2, \quad t > 0, \qquad (5.19)$$

with periodic boundary conditions and uniform random perturbations in $[-\epsilon, \epsilon]$ for the initial condition. In this model, $\epsilon > 0, m \geq 0$ are parameters that characterize the steady state behaviour of the solution [6, 5]. As described in [6], (5.19) is of interest for its connections to the self-assembly of diblock copolymers. In model (5.19), different regions in the parameter space $(\epsilon, m)$ dictate the formation of distinct energy-minimizing patterns from an initial well-mixed state. Numerically, one expects difficulties generating these energy-minimizing states. As in the previous example, we consider our methods for handling the exponentially slowing dynamics that take place as the solution inches toward steady state. We will show that our methods are stable and can quickly advance the solution to determine the minimizing states.

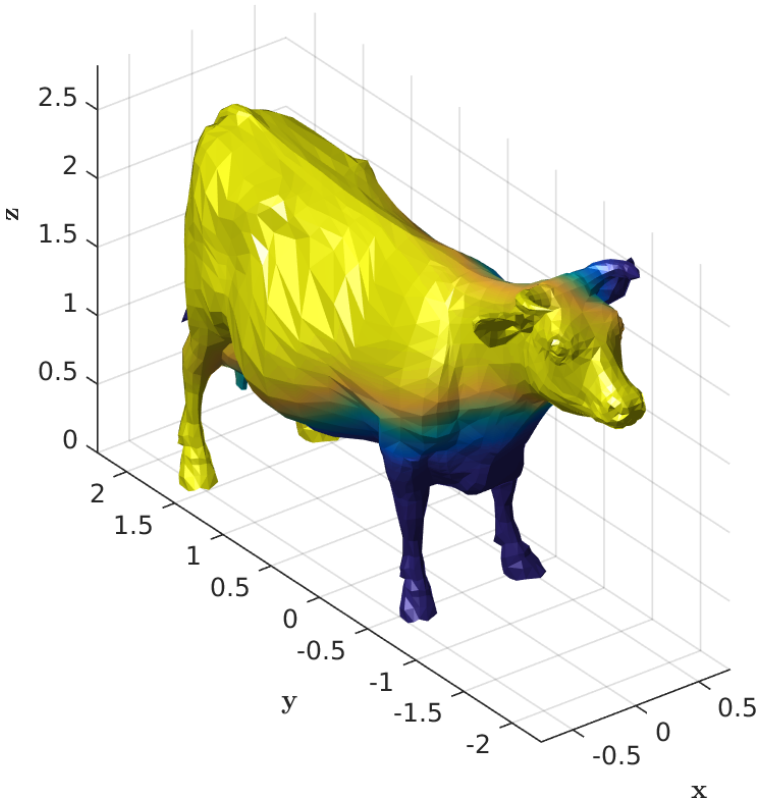


Figure 5.12: Cahn-Hilliard on a cow-shaped surface.

To stabilize (5.19), we first need to estimate the term $\Delta(u^3 + 3mu^2 - (1 - 3m^2)u)$. We proceed as in (5.16) to find the linearized form

$$v_t = -\epsilon^2 \Delta^2 v - v + (3(u^n)^2 + 6mu^n - (1 - 3m^2)\Delta v. \tag{5.20}$$

Thus if we stabilize with $p\Delta u$, choosing $p \geq (2 + 6m + 3m^2)p_0$ will be sufficient to guaranteeing unconditional stability.

Fig. 5.13 gives experiments showing that our methods can be used to access the energy-minimizing states of (5.19). As a final note, we mention that we adopt the strategy of spectral filtering that was proposed in [5].

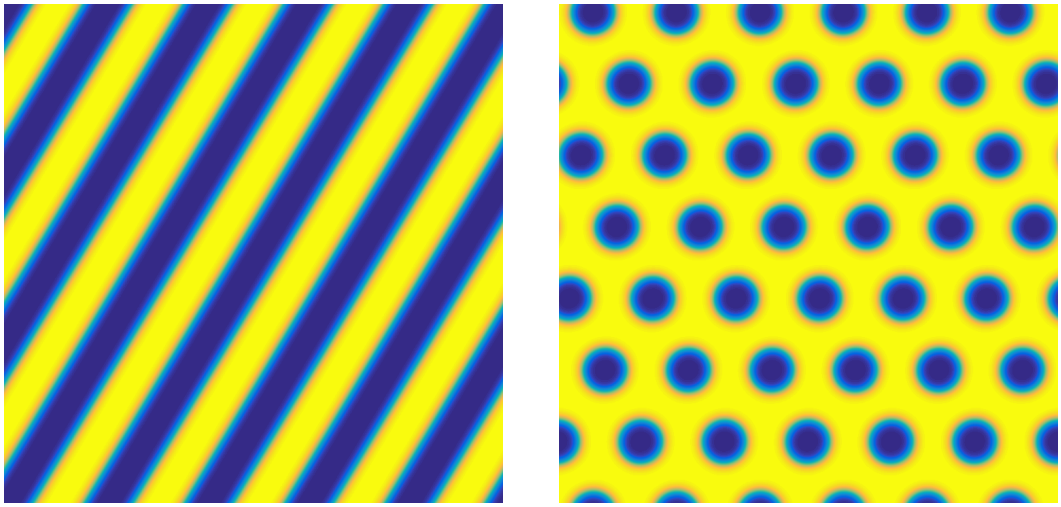

Figure 5.13: Energy minimizing patterns of the nonlocal Cahn-Hilliard (5.19). Left: Lamellae, with $(\epsilon, m) = (0.10, 0)$. Right: Hexagonally packed spots, with $(\epsilon, m) = (0.10, 0.40)$.

# Chapter 6

# Conclusion

In this thesis, we gave details of a framework for developing effective linearly stabilized time stepping methods. As was already known, unconditional stability is required. In our work, we further deduced that the parameter restriction over which we enjoy unconditional stability must be unbounded.

Alongside stability, there is the matter of accuracy. As the former dictated the range from which we can select the parameter $p$, it was then natural to explore how the discretization error behaves as a function of $p$. We recommended the simple procedure of applying the numerical method to the modified test equation and examining the series expansion at $\Delta t = 0$. When the coefficient of the leading order error term is a degree two or higher polynomial in $p$, the schemes fared poorly for $p$ large.

We also addressed the feasibility of taking large time step-sizes. The schemes that perform well possess strong damping as $z \to -\infty$. We have shown that this quality, as was with the previous two, is easy to check for any time stepping method.

On the matter of developing linearly stabilized schemes with the aforementioned properties, we have proposed a number of new methods based on IMEX multistep methods and exponential Runge-Kutta methods that perform exceptionally on at least two of the three, but none that perform strongly in all three.

We recommend SBDF2 for its ease of use and its superior damping to CNAB, although CNAB remains a useful alternative as it has a smaller error constant as $\Delta t \to 0$. Implementing these schemes require no more expertise than applying an implicit method to solve a constant coefficient heat equation. When the domain is periodic and $p$ can be chosen quite small, ETDRK2 and ETDRK4 offer strong performance at large step-sizes. Of the existing linearly stabilized methods, SBDF1 and the EIN method, neither are competitive with our schemes due to some combination of slower convergence rate and comparatively high computing times.

A number of questions have been raised throughout this thesis and are worthy of further consideration. We have identified three properties critical for effective linearly stabilized

schemes, and the derivation of high order methods satisfying all three remains open. While the metrics we supplied are easy to compute sufficient as a guide, a study of the discretization error on a fully nonlinear problem would elevate our current level of understanding. Adaptivity could also be investigated for both the step-size and for selecting $p$. Finally, comparing our methods to popular algorithms in applications such as image processing would be of interest.

## notes

emphasize that the original equation is solve. there is no linearization, ...

compare to other nonlinear PDE solvers

# Bibliography


[1] Closest point method software. `https://github.com/cbm755/cp_matrices`. Accessed: 2016-06-24.

[2] Uri M. Ascher, Steven J. Ruuth, and Brian T.R. Wetton. Implicit-explicit methods for time-dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 32(3):797–823, 1995.

[3] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

[4] Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010.

[5] Rustum Choksi, Mirjana Maras, and JF Williams. 2D phase diagram for minimizers of a Cahn-Hilliard functional with long-range interactions. *SIAM Journal on Applied Dynamical Systems*, 10(4):1344–1362, 2011.

[6] Rustum Choksi, Mark A. Peletier, and JF Williams. On the phase diagram for microphase separation of diblock copolymers: An approach via a nonlocal Cahn-Hilliard functional. *SIAM Journal on Applied Mathematics*, 69(6):1712–1738, 2009.

[7] Steven M. Cox and Paul C. Matthews. Exponential time differencing for stiff systems. *Journal of Computational Physics*, 176(2):430–455, 2002.

[8] Jim Douglas Jr and Todd Dupont. Alternating-direction Galerkin methods on rectangles. In Bert Hubbard, editor, *Numerical Solution of Partial Differential Equations II*, pages 133–214. Academic Press, 1971.

[9] Laurent Duchemin and Jens Eggers. The explicit–implicit–null method: Removing the numerical instability of PDEs. *Journal of Computational Physics*, 263:37–52, 2014.

[10] David J. Eyre. An unconditionally stable one-step scheme for gradient systems. *Unpublished article*, 1998.

[11] Karl Glasner. A diffuse interface approach to Hele–Shaw flow. *Nonlinearity*, 16(1):49, 2002.

[12] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[13] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, 2008.

[14] Marlis Hochbruck and Christian Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.

[15] Thomas Y. Hou, John S. Lowengrub, and Michael J. Shelley. Removing the stiffness from interfacial flows with surface tension. *Journal of Computational Physics*, 114(2):312–338, 1994.

[16] Willem Hundsdorfer and Jan G. Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, volume 33. Springer Science & Business Media, 2013.

[17] Aly-Khan Kassam and Lloyd N. Trefethen. Fourth-order time-stepping for stiff PDEs. *SIAM Journal on Scientific Computing*, 26(4):1214–1233, 2005.

[18] Colin B. Macdonald and Steven J. Ruuth. The implicit closest point method for the numerical solution of partial differential equations on surfaces. *SIAM Journal on Scientific Computing*, 31(6):4330–4350, 2009.

[19] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[20] Adam Oberman, Stanley Osher, Ryo Takei, and Richard Tsai. Numerical methods for smooth and crystalline mean curvature flow. *Communications in Mathematical Sciences*, 9:637–662, 2011.

[21] Konstantinos Papafitsoros and Carola-Bibiane Schönlieb. A combined first and second order variational approach for image reconstruction. *Journal of mathematical imaging and vision*, 48(2):308–338, 2014.

[22] Konstantinos Papafitsoros, Carola-Bibiane Schönlieb, and Bati Sengul. Combined first and second order total variation inpainting using split Bregman. *Image Processing On Line*, 3:112–136, 2013.

[23] Steven J. Ruuth and Barry Merriman. A simple embedding method for solving partial differential equations on surfaces. *Journal of Computational Physics*, 227(3):1943–1961, 2008.

[24] David Salac and Wei Lu. A local semi-implicit level-set method for interface motion. *Journal of Scientific Computing*, 35(2-3):330–349, 2008.

[25] Carola-Bibiane Schönlieb and Andrea Bertozzi. Unconditionally stable schemes for higher order inpainting. *Communications in Mathematical Sciences*, pages 413–457, 2011.

[26] Jianhong Shen and Tony F. Chan. Mathematical models for local nontexture inpaintings. *SIAM Journal on Applied Mathematics*, 62(3):1019–1043, 2002.

[27] Avram Sidi. *Practical Extrapolation Methods: Theory and Applications*. Cambridge University Press, New York, NY, USA, 1st edition, 2003.

[28] Roger B. Sidje. Expokit: a software package for computing matrix exponentials. *ACM Transactions on Mathematical Software (TOMS)*, 24(1):130–156, 1998.

[29] Valeria Simoncini and Daniel B. Szyld. Recent computational developments in Krylov subspace methods for linear systems. *Numerical Linear Algebra with Applications*, 14(1):1–59, 2007.

[30] Peter Smereka. Semi-implicit level set methods for curvature and surface diffusion motion. *Journal of Scientific Computing*, 19(1):439–456, 2003.

[31] John C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations, Second Edition*. Society for Industrial and Applied Mathematics, 2004.

[32] Xiaodi Sun and Michael J. Ward. Dynamics and coarsening of interfaces for the viscous Cahn–Hilliard equation in one spatial dimension. *Studies in Applied Mathematics*, 105(3):203–234, 2000.

[33] P. J. van der Houwen. On the time integration of parabolic differential equations. In G. Alistair Watson, editor, *Numerical Analysis: Proceedings of the 9th Biennial Conference Held at Dundee, Scotland, June 23–26, 1981*, pages 157–168. Springer Berlin Heidelberg, Berlin, Heidelberg, 1982.

# Appendix A

# Code

## A.1 Time integration of the Cahn-Hilliard equation using linearly stabilized time stepping schemes

```
function[]=CHp_2D()
% Solve the Cahn-Hilliard in 2d,
%   u_t = -ep^2*L^2(u) + L(u^3-u), (x,y)\in [0,2*pi]^2, t>0,
% with spectral method in space and a linearly stabilized method in time.
% For 3d, change wave numbers to the commented line. Adjust LR and the
% means in the ETDRK schemes. Adjust visualization commands.

%% Parameters: spatial, temporal, model, visualization
  N=256;  L=2*pi;
  T0=0; Tf=10; nt=400; dt=(Tf-T0)/nt; meth='cnab'; meth=upper(meth);
  epsilon=0.1; ep2=epsilon^2;
  vis_update=40;

%% Initial condition and storage
  rng('default'); u=2*epsilon*(rand(N,N)-0.50); v=fftn(u);

%% System for inversion
% Wave numbers
  k=[0:N/2-1 0 -N/2+1:-1]'*(2*pi/L); k=-k.^2;
  k=bsxfun(@plus,k,k');
%   k=bsxfun(@plus,bsxfun(@plus,k,k'),reshape(k,1,1,N));
  kdt=dt*k;
% Choose method
  switch meth
    case 'SBDF1'
      fignum=1; p=1.0; A=1-p*kdt+(dt*ep2)*k.^2;

    case 'SBDF2'
      fignum=2; p=1.5; A=1+(2/3)*(-p*kdt+(dt*ep2)*k.^2);
```

```
    case 'CNAB'
      fignum=3; p=2; A=1+(0.5)*(-p*kdt+(dt*ep2)*k.^2);
      B=(1-0.5*(-p*kdt+(dt*ep2)*k.^2))./A;

    case 'ETDRK4'
      fignum=4; p=1.0;
      Ln=p*kdt-(dt*ep2)*k.^2; E=exp(Ln); E2=exp(0.5*Ln);
      M=16; r=exp(1i*pi*((1:M)-0.5)/M);
      LR=bsxfun(@plus,Ln(:,:,ones(M,1)),reshape(r,1,1,M));
      A=kdt.*real(mean((exp(0.5*LR)-1)./LR,3));
      b1=kdt.*real(mean((-4-LR+exp(LR).*(4-3*LR+LR.^2))./LR.^3,3));
      b2=2*kdt.*real(mean((2+LR+exp(LR).*(-2+LR))./LR.^3,3));
      b4=kdt.*real(mean( (-4-3*LR-LR.^2+exp(LR).*(4-LR))./LR.^3,3));

    case 'ETDRK2'
      fignum=5; p=1.0;
      Ln=p*kdt-(dt*ep2)*k.^2; E=exp(Ln);
      M=16; r=exp(1i*pi*((1:M)-0.5)/M);
      LR=bsxfun(@plus,Ln(:,:,ones(M,1)),reshape(r,1,1,M));
      A1=kdt.*real(mean((exp(LR)-1)./LR,3));
      A2=kdt.*real(mean((exp(LR)-LR-1)./LR.^2,3));
  end
  fg=figure(100+fignum);

%% Begin time stepping
  for kt=1:nt,
    switch meth
      case 'SBDF1'
        rhs=v+kdt.*fftn(u.^3-(1+p)*u); v=rhs./A;

      case 'SBDF2'
        if kt==1,
          v1=v; f1=kdt.*fftn(u.^3-(1+p)*u);
          v=(v1+f1)./(1-p*kdt+(dt*ep2)*k.^2);
        else
          f=kdt.*fftn(u.^3-(1+p)*u);
          u=((4/3)*v-(1/3)*v1+(2/3)*(2*f-f1))./A;
          v1=v; f1=f; v=u;
        end

      case 'CNAB'
        if kt==1,
          f1=kdt.*fftn(u.^3-(1+p)*u);
          v=(v+f1)./(1-p*kdt+(dt*ep2)*k.^2);
        else
          f=kdt.*fftn(u.^3-(1+p)*u);
```

```
        v=B.*v+0.5*(3*f-f1)./A;
        f1=f;
      end

    case 'ETDRK4'
      f=fftn(u.^3-(1+p)*u); a=E2.*v+A.*f; ar=real(ifftn(a));
      fa=fftn(ar.^3-(1+p)*ar); b=E2.*v+A.*fa; br=real(ifftn(b));
      fb=fftn(br.^3-(1+p)*br); c=E2.*a+A.*(2*fb-f); cr=real(ifftn(c));
      fc=fftn(cr.^3-(1+p)*cr);
      v=E.*v + b1.*f + b2.*(fa+fb) + b4.*fc;

    case 'ETDRK2'
      f=fftn(u.^3-(1+p)*u); a=E.*v+A1.*f; ar=real(ifftn(a));
      fa=fftn(ar.^3-(1+p)*ar);
      v=a+A2.*(fa-f);
  end
  u=real(ifftn(v));

% Plots
  if kt==nt || mod(kt,vis_update)==0,
    set(0,'currentfigure',fg); drawnow;
    pcolor(u); shading('interp'); axis('equal','off');
    title(['CH by ',meth,' with dt=',num2str(dt,'%10.2g'), ...
      ' at T=',num2str(dt*kt,'%10.2g')]);
  end
 end
end
```