AN ABSTRACT OF THE THESIS OF

Lisa Bigler for the degree of Master of Science in Mathematics presented on January 3, 2019.

Title: Numerical Study of Convexity Splitting Scheme for Coupled Phase Field and Stefan Free Boundary Problems

Abstract approved: _____

Malgorzata Peszyńska

In this work, we consider a convexity splitting scheme for a coupled phase field and energy equation, a modification of Stefan problem. The Stefan problem is a free boundary value problem that models the temperature in a homogeneous multiphase medium. Each phase is modeled using a heat diffusion parabolic partial differential equation and incorporates latent heat at the phase interface. The existence of the jump in the heat flux due to the latent heat leads to a nonlinear free boundary problem. The Stefan problem has a sharp interface, thus coupling the Stefan problem with a phase field model allows for a more realistic diffuse region of phase transition. Solving this phase field and temperature system implicitly can by computationally expensive and using time lagging methods can be unstable, thus we look into the viability of using an efficient and stable convexity splitting scheme for the coupled system, in several variants.

Numerical Study of Convexity Splitting Scheme for Coupled Phase Field and Stefan Free Boundary Problems

by

Lisa Bigler

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented January 3, 2019
Commencement June 2019

Master of Science thesis of Lisa Bigler presented on January 3, 2019

APPROVED:

Major Professor, representing Mathematics

Head of the Department of Mathematics

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Lisa Bigler, Author

ACKNOWLEDGEMENTS

Academic

I am indebted to Dr. Malgorzata Peszynska for her guidance, support and advice. Her drive and enthusiasm inspire me every day.

I would also like to thank the National Science Foundation; research in this work was partially supported by grant number NSF DMS-1522734 Phase transitions in porous media across multiple scales, (Principal Investigator Malgorzata Peszynska) under the direction of Malgorzata Peszynska.

Personal

I wish to thank my family for there never ending support, and my cohort for creating a collaborative, enthusiastic community. Finally, I would like to thank my friends, who have remained supportive throughout my move and adjustment to graduate school.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

NUMERICAL STUDY OF CONVEXITY SPLITTING SCHEME FOR COUPLED PHASE FIELD AND STEFAN FREE BOUNDARY PROBLEMS

1. INTRODUCTION

Modeling phase transitions has important applications in geosciences, engineering, and medicine. In particular, modeling ice–water phase transitions is relevant for modeling sea–ice, permafrost in the Arctic, food processing, as well as for numerous biomedical applications such as frostbite and processes occurring during cryosurgery.

The relevant models are nonlinear differential equations which account for heat conduction and phase transition when ice is melting or water is freezing due to the external environmental conditions. These processes can occur very fast or slowly, thus the models come in several variants. Due to their complexity almost no closed form analytical solutions are available, thus we resort to numerical schemes which approximate the solutions. A key component of every scheme is a solver for the nonlinear algebraic system of equations. A fully implicit nonlinear scheme requires, e.g., Newton's method for its solution. The implementation of such a solver might be cumbersome, thus many researchers resort to explicit or semi-implicit approaches.

In this thesis we consider a particular numerical technique for phase transitions called convexity splitting which does not require a fully implicit nonlinear solver. We consider this technique first for a so-called phase field model itself, and next for that model coupled to the Stefan problem.

The Stefan problem is a free boundary value problem used to model temperature in a homogeneous media in which phase transitions occur, such as water and ice. The Stefan problem is a nonlinear partial differential equation (PDE) of parabolic type, solved for the temperature $u(x,t)$. This PDE is solved subject to some boundary and initial conditions.

The location of the free boundary needs to be kept track of because the phase interface is where the nonlinearity comes into effect as the jump in the heat flux due to latent heat. Solutions to the Stefan problem are difficult to compute numerically due

to the fact that the free boundary at the interface of phases moves, and its position is unknown. The free boundary is defined implicitly by the jump condition of the heat flux due to latent heat. This condition with the latent heat term appears under the time derivative in the original Stefan problem; we explain this connection in Chapter 4.

Alternatively to this jump condition in the Stefan problem, the PDE can be rewritten with the order parameter φ, which takes on value "-1" for solid phase material and "1" for liquid phase material. In other words, φ is the sign of the temperature u.

A phase field model for this free boundary value problem relaxes the Stefan condition, $\varphi \in \text{sign}(u)$, and gives its own dynamics, i.e. $\varphi(x,t)$ is the solution to another PDE for which the temperature $u(x,t)$ is a source term.

The phase field equation is a nonlinear equation which has been studied on its own, even without the source term. In particular a class of convexity splitting schemes have been introduced as a stable alternative to fully implicit schemes.

However, there is relatively little work devoted to the coupled system. In this work, we explore the effect of those splitting schemes on the phase field model, coupled to the original Stefan problem. In particular, we find that for the cases explored, the splitting method is as stable as fully implicit methods. with less computational effort than the fully implicit scheme, but can come at the cost of higher local truncation error.

The outline of this thesis are as follows. Chapter 2 includes background information on the finite difference approximation for ordinary differential equations (ODEs) and parabolic PDEs, on the Newton's method, and the splitting method. This leads into Chapter 3 which covers some numerical methods for nonlinear parabolic PDEs including the convexity splitting method. Chapter 4 discusses the weak formulation of the Stefan problem, phase fields, and the coupling the Stefan problem with a phase field model. We also compare the stability and the computational effort for numerical methods for this coupled system. Chapter 5 includes our conclusions.

2. BACKGROUND

In this Chapter we introduce the splitting method as well as some preliminary material known from standard textbooks on finite difference approximation to the solutions of ODEs and PDEs. In Section 2.5. we also recall the Newton's method for solving nonlinear equations, as well as introduce the technique called "convexity splitting".

2.1. Splitting for a model ODE

Consider an initial value problem for the nonlinear ODE

$$\frac{du}{dt} = u - u^3, \quad u(0) = u_0. \tag{2.1}$$

There are many possible ways to approximate the solutions to (2.1). In particular, one can consider a fully implicit or an explicit scheme, respectively

$$U^{n+1} - U^n = \tau[U^{n+1} - (U^{n+1})^3]. \tag{2.2}$$

$$U^{n+1} - U^n = \tau[U^n - (U^n)^3]. \tag{2.3}$$

Each of these methods is first order accurate in τ. Fully implicit method requires that we solve the nonlinear algebraic equation. Fully explicit scheme provides the solution without the need for the nonlinear solver, but may require small time steps.

The main goal of this section is to look at a method proposed in [4] which is somewhat "between" the fully implicit and explicit approaches. The method proposed in Eyre's paper [4] considers the convexity of a solution at each time step and splits the functional into so called contractive and expansive terms; we attempt to explain these notions below. This section will discuss this convexity splitting method and compare the results to other

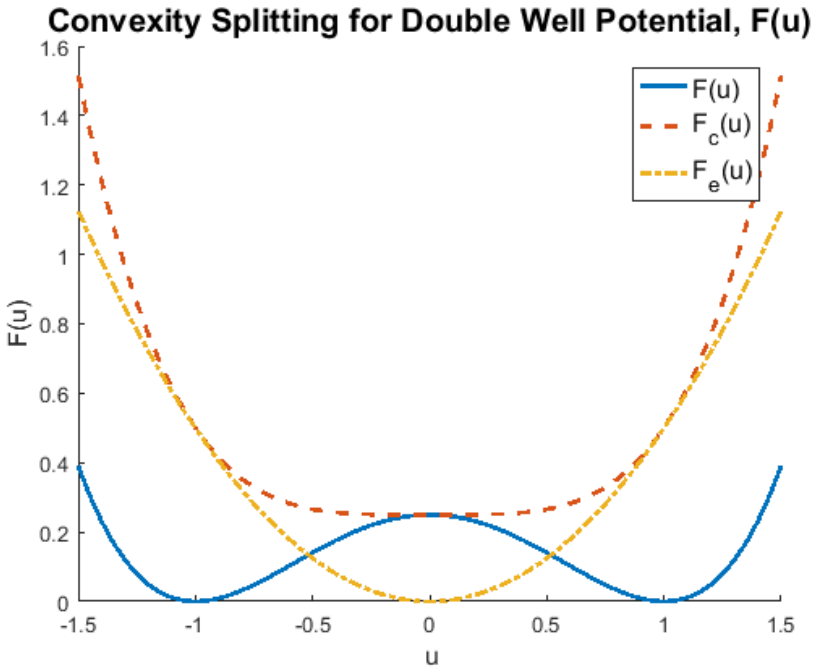

FIGURE 2.1: Splitting for $F(u)$ in section 2.1., where $F_c(u)$ is the contractive part and $F_e(u)$ is the expansive part and there difference is $F(u)$.

methods.

Consider the equation (2.1), which can be written as

$$\frac{du}{dt} = -\nabla F(u), \quad u(0) = u_0, \tag{2.4}$$

where

$$F(u) = \frac{(u^2-1)^2}{4}. \tag{2.5}$$

This function $F(\cdot)$ is the well-known double-well potential plotted in Figure 2.1. We note that the use of ∇ in (2.4) is consistent with the literature [4]; for the present case we could have used $F'(u)$ instead.

The main idea described in [4] is to break $F(u)$ up into an expansive term, $F_e(u)$,

and a contractive term, $F_c(u)$ such that

$$F(u) = F_c(u) - F_e(u). \tag{2.6}$$

More precisely, we split $F(u)$ into the difference of two convex terms, F_c and F_e, such that $u' = -\nabla F_c(u)$ is contractive and that $u' = \nabla F_e(u)$ is expansive, where u' is the time derivative of u.

In order to define what is meant by expansive and contractive, some assumptions need to be stated. Here we follow [4]. Consider a nonlinear function $F(\cdot) : R^p \to \mathbb{R}$ which satisfies

$$\begin{cases} F(u) \geq 0 & \forall u \in \mathbb{R}^p \\ F(u) \to \infty & \text{as } ||u|| \to \infty \\ \langle J(\nabla F)(u)u, u\rangle \geq \lambda & \forall u \in \mathbb{R}^p \end{cases} \tag{2.7}$$

where J is the Jacobian (matrix of partial derivatives), $\lambda \in \mathbb{R}$ and $\langle\cdot\rangle$ is the euclidean inner product with $||\cdot||$ representing the norm induced by the euclidean inner product.

If $\lambda \geq 0$ then $F(\cdot)$ is convex and the flow is considered contractive. Here we recall that for convex functions only one minimizer, or equilibrium, point exists. If $\lambda < 0$, then $F(\cdot)$ is locally concave and flows can expand. In this case, local equilibrium might be the local maximizer.

For example, when $F(\cdot)$ is given as in (2.5), we have that $\nabla F(u) = u^3 - u$, and $J(\nabla F)(u) = 3u^2 - 1$, so that $\lambda = -1$. The function $F(\cdot)$ has three equilibria (critical points) $-1, 0, 1$. The function F is convex outside $|u| \leq \sqrt{\frac{1}{3}}$, and concave inside that set. The equilibrium $u = 0$ is the local maximizer, and the equilibria ± 1 are the local minimizers.

2.2. Splitting, and notion of gradient stability

Here we illustrate how splitting works for the double-well potential function. We also define the particular notion of gradient stability which extends the usual notion of numerical stability for numerical schemes for ODEs that $||U^{n+1}|| \leq ||U^n||$, which is expected for a numerical scheme for an ODE for which the true solution exhibits the behavior that the distance $|u(t)| \to 0$ as $t \to \infty$. In fact, we consider the more general situation in which the true solution $|u(t) - u_\infty| \to 0$ and u_∞ is the limit equilibrium of the problem.

We note that not all ODEs have a meaningful equilibrium u_∞. For example, $u' = u$ or $u' = u^3$ do not, but $u' = -u$ or $u' = -u^3$ do, both with $u_\infty = 0$.

In turn, $u' = u - u^3$ has two possible equilibria $u_\infty = 1, u_\infty = -1$, each attainable from an initial condition which is positive, or negative, respectively. For these two equilibria we do not expect that $||U^{n+1}|| \leq ||U^n||$, but rather than for some function $\mathcal{F}(U^{n+1}) \leq \mathcal{F}(U^n)$. The function $\mathcal{F}$ depends on the problem and on the scheme. In particular, we expect that $\mathcal{F}(u) = \frac{1}{4}(u^2-1)^2$, same as the double well potential, describes the behavior of the time steps U^n as the solution converges towards $u_\infty = \pm 1$.

2.2.1 Splitting

In Figure 2.1 we show one example of a concave-convex or expansive-contractive splitting,

$$F_c(u) = \frac{u^4}{4} + \frac{1}{4} \quad \text{and} \quad F_e(u) = \frac{u^2}{2}. \tag{2.8}$$

For this splitting,

$$-\nabla F_c(u) = -u^3 \quad \text{and} \quad \nabla F_e(u) = u. \tag{2.9}$$

Consider

$$\frac{du}{dt} = -u^3, \tag{2.10}$$

then when u is negative its derivative is positive and thus u increasing and when u is positive its derivative is decreasing, thus its behavior is contractive. Another way to see this is, the solution for equation (2.10) is

$$u = \text{sign}(u_0)\frac{1}{\sqrt{2t+c}}, \quad \text{where } c = \frac{1}{u_0^2}. \tag{2.11}$$

Note that as $t \to \infty$, $u \to 0$.

Now consider the function F_e, and consider

$$\frac{du}{dt} = u \tag{2.12}$$

then when u is positive, its derivative is positive thus u is increasing and when u is negative its derivative is negative, hence u is decreasing. The solution to equation (2.12) is

$$u = u_0 e^t. \tag{2.13}$$

As $t \to \infty$, $|u| \to \infty$ where the sign depends on c, thus u is expanding with time.

The reason for breaking up F in such a way it to provide a gradient stable method. Gradient stability is defined in [4] for unconditionally stable and [9] for conditionally stable schemes. A one-step numerical method for equation (2.4) is unconditionally gradient stable if there exists some function $\mathcal{F} : \mathbb{R}^p \to \mathbb{R}$ such that for all $\tau > 0$ and for all initial values

(i) $\mathcal{F}(U) \geq 0$ for all $U \in \mathbb{R}$.

(ii) $\mathcal{F}(U) \to \infty$ as $||U|| \to \infty$.

(iii) $\mathcal{F}(U^{n+1}) \leq \mathcal{F}(U^n)$ for all $U^n \in \mathbb{R}^p$.

(iv) If $\mathcal{F}(U^n) = \mathcal{F}(U_0)$ for all $n \geq 0$ then U_0 is in the set of zeros of $\nabla\mathcal{F}$.

The function $\mathcal{F}$ may or may not be the same as F in the original equation (2.4). For many numerical schemes, $\mathcal{F}$ can be taken to be F, including the splitting method proposed in equation (2.16) from Section 2.2.4. The $F(\cdot)$ in equation (2.5) can be thought of as an energy and $F(U^{n+1}) \leq F(U^n)$ as the energy dissipation property.

2.2.2 Example: gradient stability for the fully implicit scheme for $u' = -u^3$

The magnitude of solutions to $u' = -u^3$ exhibit decay towards $u_\infty = 0$. The fully implicit scheme

$$U^{n+1} - U^n + \tau(U^{n+1})^3 = 0 \tag{2.14}$$

and we can rewrite $U^n = U^{n+1} + \tau(U^{n+1})^3$. If $U^n = 0$ then $U^{n+1} = 0$, thus consider the case where $||U^n|| \neq 0$. This leads to the following

$$\begin{aligned} ||U^n|| &= ||U^{n+1} + \tau(U^{n+1})^3|| \\ &= ||U^{n+1}||\,||1 + \tau(U^{n+1})^2|| \end{aligned}$$

thus

$$\frac{||U^n||}{||U^{n+1}||} = ||1 + \tau(U^{n+1})^2|| \geq 1.$$

Therefore, the traditional stability holds $||U^{n+1}|| \leq ||U^n||$. In addition, we can see that $u^3 = \nabla(u^4)/4 = \nabla F(u)$ and thus, equivalently, $F(U^{n+1}) \leq F(U^n)$.

2.2.3 Example: gradient stability for the fully implicit scheme for $u' = u - u^3$

For this case, we have

$$U^{n+1} - U^n + \tau(U^{n+1})^3 - \tau U^{n+1} = 0 \tag{2.15}$$

and we can calculate $U^n = U^{n+1} + \tau((U^{n+1})^3 - U^{n+1}) = U^{n+1}(1 - \tau(1 - (U^{n+1})^2))$. If τ is small enough one can see that the sign of U^{n+1} matches the sign of U^n. However, we will not have in general that $||U^{n+1}|| \leq ||U^n|| = ||U^{n+1}||(1 - \tau(1 - (U^{n+1})^2))$, unless $(U^{n+1})^2 > 1$.

On the other hand, the "energy" of U^{n+1}, a nonlinear function of the distance from the equilibria ± 1, is likely to decay. To see this, consider the double well potential energy function $F(u) = \frac{1}{4}(u^2 - 1)^2$. When comparing $F(A) \leq F(B)$ we only need to check if $(A^2 - 1)^2 \leq (B^2 - 1)^2$.

Thus to show that $F(U^{n+1}) \leq F(U^n)$, we need to check if $(A^2 - 1)^2 \leq (B^2 - 1)^2$, with $A = U^{n+1}$, and $B = U^n = A(1 - \tau(1 - A^2))$.

In particular, we can consider two cases.

Let $|A| > 1$. Then we show that $|B| > |A| > 1$ and thus $(A^2 - 1)^2 \leq (B^2 - 1)^2$. To show $|B| > |A|$ start with $|A| > 1$ then $1 - A^2 < 0$, thus $1 - \tau(1 - A^2) > 1$ concluding that $|B| = |A|\,|(1 - \tau(1 - A^2))| > |A|$.

Let $|A| < 1$. Then we show that $1 > |A| > |B|$ and thus $(A^2 - 1)^2 \leq (B^2 - 1)^2$. To show $|A| > |B|$, start with $|B| = |A|\,|1 - \tau(1 - A^2)|$ and notice that $|1 - \tau(1 - A^2)| < 1$. In order for $|A| > |B|$ we need that $0 < \tau < \frac{2}{1 - A^2}$, thus the finite difference implicit method is conditionally stable.

2.2.4 Example: gradient stability for the convexity splitting for $u' = u - u^3$

The convexity splitting method discussed in [4] evaluates the contractive term implicitly, while evaluating expansive term explicitly as follows

$$U^{n+1} - U^n = \tau[\nabla F_e(U^n) - \nabla F_c(U^{n+1})]. \tag{2.16}$$

A formal proof that this method is unconditionally gradient stable is provided by Eyre in [4].

We will show here that the splitting as in equation (2.8), $F_c(u) = \frac{u^4}{4} + \frac{1}{4}$ and $F_e(u) = \frac{u^2}{2}$, satisfies the energy dissipation property and discuss why the method is unconditionally gradient stable using $\mathcal{F} = F$.

To show $F(U^{n+1}) \leq F(U^n)$ first note that from equation (2.16), the numerical scheme is

$$U^{n+1} - U^n = \tau[U^n - (U^{n+1})^3], \tag{2.17}$$

and thus

$$U^n = \frac{U^{n+1} + \tau(U^{n+1})^3}{1+\tau}. \tag{2.18}$$

We will show that the property (iii) holds with the choice of $\mathcal{F} = F$.

As above, we want to show $F(A) \leq F(B)$, and we only need to check if $(A^2 - 1)^2 \leq (B^2 - 1)^2$. In the convexity splitting scheme we have that with $A = U^{n+1}$, and $B = U^n = A\left(\frac{1+\tau A^2}{1+\tau}\right)$.

As before, we consider two cases.

Let $|A| \geq 1$. Then $\frac{1+\tau A^2}{1+\tau} \geq 1$ and thus $A^2\left(\frac{1+\tau A^2}{1+\tau}\right)^2 - 1 \geq A^2 - 1 \geq 0$, hence $(A^2-1)^2 \leq (B^2-1)^2$.

Let $|A| < 1$, then $0 < \frac{1+\tau A^2}{1+\tau} < 1$ and thus $|B| = |A\left(\frac{1+\tau A^2}{1+\tau}\right)| < |A| < 1$ thus $(A^2-1)^2 < (B^2-1)^2$. Thus, in this case F satisfies the energy dissipation property, property (iii) in the definition of unconditionally gradient stable.

As we see, calculations even for this particularly simple splitting are involved. The theorem in [4] provides an elegant way to prove the general case as long as the splitting satisfies the expansive-contractive criteria.

We see that (i) and (ii) are easily satisfied since $F(U)$ due to the form of $F9\cdot)$. We see that $F(U) \geq 0$ and $F(U) \to \infty$ as $||U|| \to \infty$. Thus both (i) and (ii) are met. Now for the fourth requirement, (iv) in the definition. Since F satisfies equations (2.7), since F is a gradient flow, given by the equation (2.4), the zeros of ∇F are isolated and since F and ∇F are smooth functions, (iv) is met. Theorem 2 in [4] provides more details on why.

This calculation shows that we have unconditional gradient stability, i.e., without assuming τ is small enough.

2.2.5 Different splitting

The equation for the splitting used in Figure 2.1 and in Equation (2.16) is

$$U^{n+1} + \tau(U^{n+1})^3 = (1+\tau)U^n. \tag{2.19}$$

This scheme is nonlinear and requires a method such as Newton's method which might be computationally expensive.

In the interest in finding a stable method that is less computationally heavy than a fully implicit method, equations 2.1, 2.4, and 2.5, can be split up differently into

$$F_c(u) = \frac{5u^2}{2} + \frac{1}{4} \quad \text{and} \quad F_e(u) = \frac{6u^2}{2} - \frac{u^4}{4}. \tag{2.20}$$

This splitting is depicted in Figure 2.2. The scheme for this splitting becomes

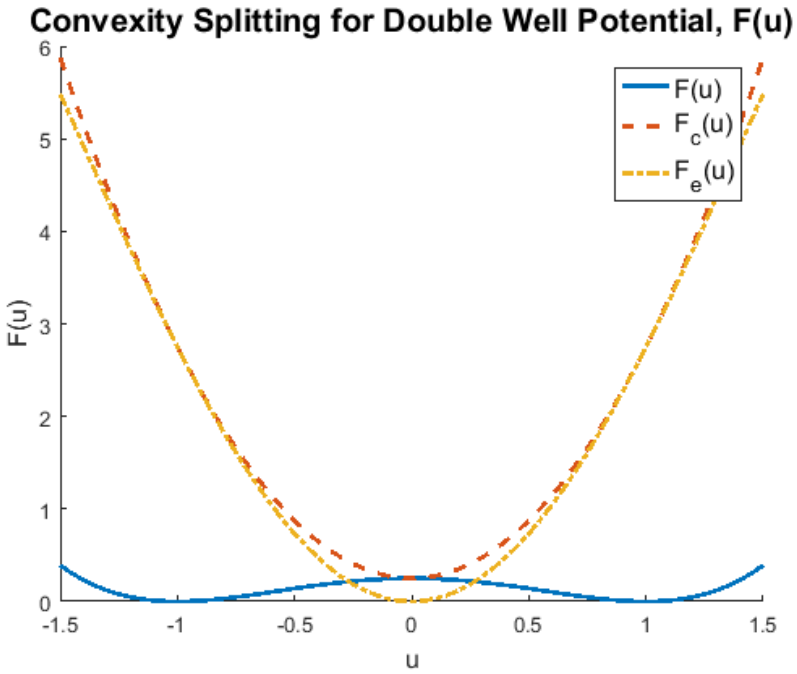

FIGURE 2.2: Plot for convexity splitting based on equations 2.20 in section 2.1.

$$(1+5\tau)U^{n+1} = (1+6\tau)U^n - \tau(U^n)^3. \tag{2.21}$$

This scheme, in contrast to (2.19), can be solved using a linear solver.

2.3. Example of application of splitting method

We illustrate the splitting and compare the solutions to the scheme (2.21) to those for both MATLAB ode45 solver and backward Euler given in equation (2.2). The latter is solved using Newton's method (introduced later) at each time step. We use ode45 solver as a proxy for exact solution. While the exact solution can be computed easily for the present ODE case, it may not be possible in general. We see easily the exact solution is

$$u(t) = \text{sign}(u_0)\frac{e^t}{\sqrt{c+e^{2t}}}, \text{ where } c = \frac{1}{u_0^2} - 1,$$

u_0	τ	backward Euler error	α_{BE}	splitting scheme error	α_{SS}	ode45 error
-0.75	0.5	0.0245122	-	0.1127655	-	0.0000390
-0.75	0.05	0.0029494	0.920	0.0207292	0.736	
-0.75	0.005	0.0003011	0.991	0.0022693	0.961	
-0.75	0.0005	0.0000302	0.999	0.0002291	0.996	
-0.75	0.00005	0.0000030	1.000	0.0000229	1.000	

TABLE 2.1: Error, $||U - u||_\infty$ as in equation (2.22), with $t_{final} = 5$ and with initial condition $u_0 = -0.75$.

for $u_0 \neq 0$; for $u_0 = 0$, $u(t) = 0$ for all t.

To get a sense of the computational effort required, we keep track of the number of iterations depending on the time step: the maximum number of iterations was 5 when $\tau = 0.5$, 3 when $\tau = 0.05$ and 2 for the rest of the values of τ; see Tables 2.1 and 2.2. We also show the error, calculated, as in (2.22) using the infinity norm of the difference between the respective method and the exact solution, i.e.

$$||U - u||_\infty, \tag{2.22}$$

where U is a vector of the values calculated using the numerical method at each time step between $t = 0$ and $t = t_{final} = 5$ and u is similarly a vector of exact solutions calculated at each time step between $t = 0$ and $t = t_{final} = 5$. The built-in MATLAB ODE solver, ode45, determines a variable time step, thus the solution to ode45 was only calculated once for each initial condition. The relative tolerance and absolute tolerance for ode45 was set to the default setting of RelTol=1e-3 and AbsTol=1e-6. These tolerances are used to help determine the time step size by decreasing the step size when the error is too large. The error, e, is calculated by the difference between a 4th order Runge-Kutta scheme and 5th order Runge-Kutta scheme. A couple other initial conditions were tried with very similar results.

We comment now on the results. To understand why the error in the splitting

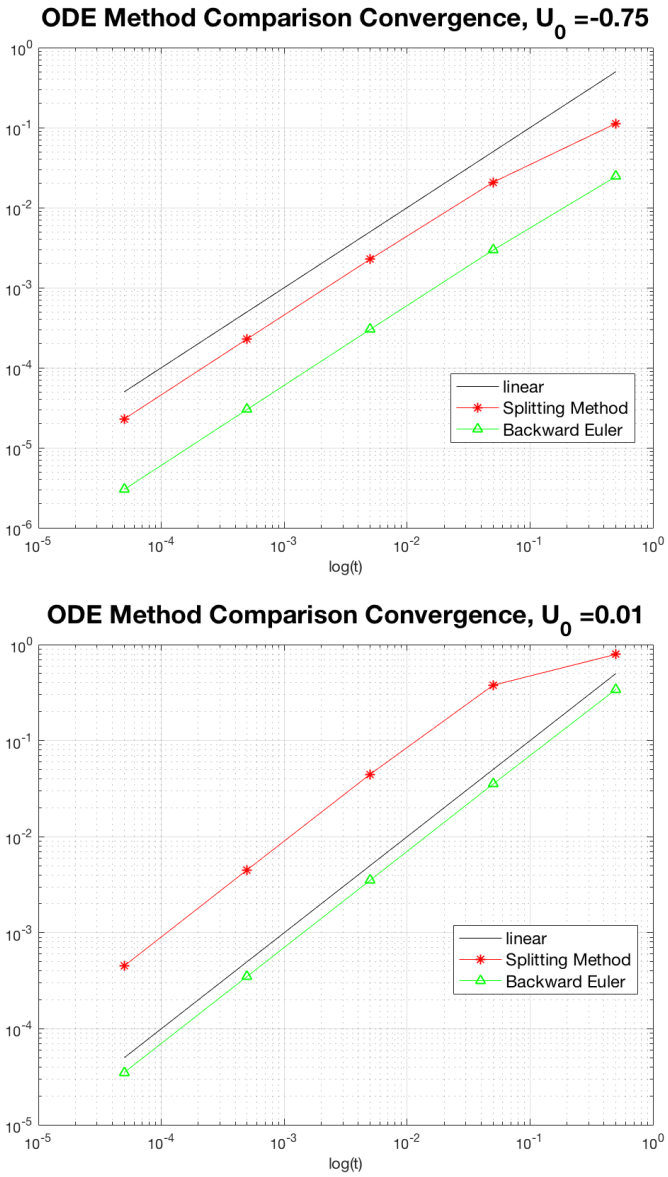

FIGURE 2.3: Error, calculated using equation (2.22), plots for backward Euler and the splitting scheme on loglog graphs, where the error was calculated using the infinity norm of the difference between the respective method and the exact solution for two different initial conditions.

u_0	τ	backward Euler error	α_{BE}	splitting scheme error	α_{SS}	ode45 error
0.01	0.5	0.3418509	-	0.7913338	-	
0.01	0.05	0.0352905	0.986	0.3770012	0.322	
0.01	0.005	0.0035146	1.002	0.0446279	0.927	0.0000158
0.01	0.0005	0.0003513	1.000	0.0045159	0.995	
0.01	0.00005	0.0000351	1.000	0.0004521	1.000	

TABLE 2.2: Error, $||U - u||_\infty$ as in equation (2.22), with $t_{final} = 5$ and with initial condition $u_0 = 0.01$.

method is higher than for backward Euler, we consider the local truncation error (LTE) analysis.

2.3.1 ODE local truncation error analysis

As described in [6] the local truncation error is calculated by evaluating the finite difference scheme at the exact solution, as in (2.26).

Using the backward Euler method for $u' = f(u) = u - u^3$ and following the notation in [6],

$$\frac{U^{n+1} - U}{\tau} = f(U^{n+1}) \tag{2.23}$$

$$= U^{n+1} - (U^{n+1})^3. \tag{2.24}$$

Using the Taylor expansion

$$u(t) = u(t + \tau - \tau) = u(t + \tau) - \tau u'(t + \tau) + \frac{\tau^2}{2} u''(t + \tau) - \dots \tag{2.25}$$

and using it in the LTE analysis for the backward Euler method, equation (2.23), the LTE

is then calculated

$$\begin{aligned} LTE &= \frac{u(t+\tau)-u(t)}{\tau} - f(u(t+\tau)) \\ &= \frac{u(t+\tau)}{\tau} - \frac{u(t+\tau)}{\tau} + u'(t+\tau) - \frac{\tau}{2}u''(t+\tau) + \cdots - f(u(t+\tau)) \\ &= -\frac{\tau}{2}u''(t+\tau) + \dots \\ &= \mathcal{O}(\tau) \end{aligned} \tag{2.26}$$

since $u'(t) = f\left(u(t)\right) = u - u^3$. Now consider for the splitting method, written as

$$\frac{U^{n+1}-U^n}{\tau} = 6U^n - (U^n)^3 - 5U^{n+1}. \tag{2.27}$$

The LTE, multiplied by τ to simplify the right hand side, is then calculated as

$$\begin{aligned} \tau \cdot LTE =& (1+5\tau)u(t+\tau) - (1+6\tau)u(t) + \tau(u(t))^3 \\ =& (1+5\tau)\big(u(t+\tau) - \tau u'(t+\tau) + \frac{\tau^2}{2}u''(t+\tau) - \dots\big) \\ & - (1+6\tau)u(t) + \tau(u(t))^3 \\ =& -\tau u'(t) + \tau u'(t) + 5\tau^2 u'(t) + \frac{\tau^2}{2}u''(t) + 5\frac{\tau^3}{2}u''(t) + \dots. \end{aligned}$$

This implies that the LTE is

$$\begin{aligned} LTE =& 5\tau u'(t) + \frac{\tau}{2}u''(t) + \tau^2 u'''(t) + \dots \\ =& \mathcal{O}(\tau). \end{aligned}$$

Both the splitting method and backward Euler are $\mathcal{O}(\tau)$ however the LTE for the splitting method has an extra term $5\tau u'(t)$ compared to the backward Euler, thus the error in the splitting method is higher than that of backward Euler when the signs of u' and u'' align.

The results are also plotted in Figure 2.4 for the initial condition $U_0 = 0.01$ as well

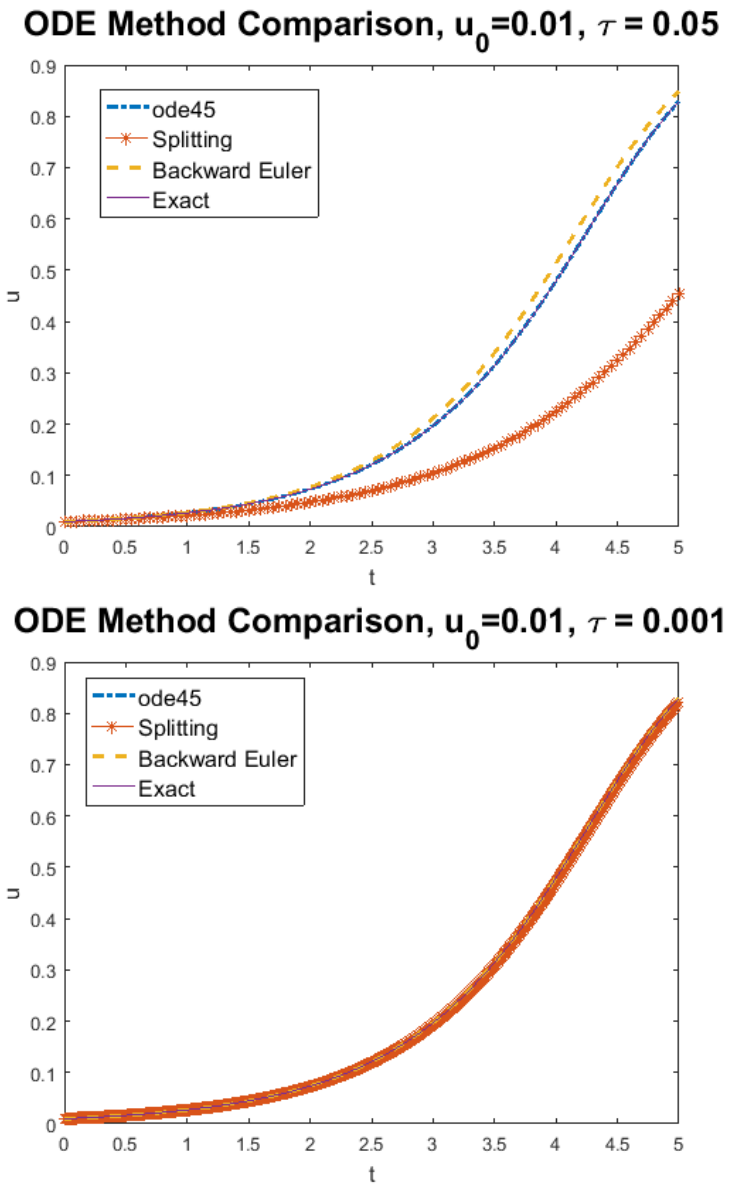

FIGURE 2.4: Solutions to the ODE (2.4) with F as in (2.5) using three methods and the exact solution. The three methods are backward Euler (2.2), MATLAB ode45, and the splitting method as in (2.21), the exact solution

as $\tau = 0.05$ and $\tau = 0.01$. The exact solution

$$u = \frac{e^t}{\sqrt{9999 + e^{2t}}} \tag{2.28}$$

is also plotted in Figure 2.4. The relatively large local truncation error of the splitting method is visible in Figure 2.4.

2.4. Parabolic PDEs

The goal of this sections is to look at finite difference methods for parabolic PDEs of the form

$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\Big(k\frac{\partial u}{\partial x}\Big) = f(x,t),\ x \in \Omega, t > 0, \tag{2.29}$$

with a variety of initial and boundary conditions. We assume that the heat conductivity coefficient $k > 0$, and that $\Omega \subset \mathbb{R}$ is open and bounded. Equation (2.29) is the heat or diffusion equation.

We follow the standard finite difference approximation as defined by LEVEQUE [6]. In particular, we consider two different ways to set up nodes and implement boundary conditions.

2.4.1 Explicit, implicit and Crank-Nicolson method for diffusion equation

The first algorithm is set up to calculate a numerical solution for

$$u_t - ku_{xx} = f(x,t) \qquad x \in (a,b),\ t > 0 \tag{2.30a}$$

$$B.C. \qquad u(a,t) = h_l(t), \qquad u(b,t) = h_r(t) \quad t > 0 \tag{2.30b}$$

$$I.C. \qquad u(x,0) = u_0(x) \qquad x \in (a,b) \tag{2.30c}$$

where $k > 0$ is a constant. Equation (2.30) is known as the diffusion equation with Dirichlet boundary conditions. Dirichlet boundary conditions specify the value at the boundary for all time. For example, if equation (2.30) models a thin insulated tube's temperature, the Dirichlet boundary conditions specified in equation (2.30b) means the temperature is set and known at the end points.

We first consider the general scheme which is Crank-Nicolson scheme when $\mu = 1/2$ but can be changed to explicit or implicit using $\mu = 1$ is used for an implicit method, $\mu = 0$ for explicit. The main equation for the interior nodes of this scheme is

$$\begin{aligned} U_j^n - rk\mu\big(U_{j+1}^n - 2U_j^n + U_{j-1}^n\big) = \\ U_j^{n-1} + rk(1-\mu)\big(U_{j+1}^{n-1} - 2U_j^{n-1} + U_{j-1}^{n-1}\big) + \Delta t\big(\mu f_j^n + (1-\mu) f_j^{n-1}\big), \end{aligned} \tag{2.31}$$

where $r = \frac{\tau}{h^2}$ where τ is the size of the uniform time step and h is the size of the uniform spatial step. The value U_j^n stands for the numerical solution at time step n and spatial node j. Here $h = 1, 2, \ldots, M-1$ where M is the number of cells the spatial domain is evenly split into and the time index $n = 1, 2, \ldots, N$, where N is the number of uniform time steps to get form t_0 to t_{final}. The values of h and τ written in terms of M and N are

$$h = \frac{b-a}{M} \quad \text{and} \quad \tau = \frac{t_{final} - t_0}{N}. \tag{2.32}$$

The equation (2.31) models the interior nodes. Boundary nodes can be calculated using the Dirichlet boundary conditions.

Letting B be the matrix representation of the left hand side of (2.31) such that BU^n

gives the corresponding jth component, we get the the following tridiagonal matrix,

$$B = \begin{bmatrix} 1+\mu 2rk & -\mu rk & & & & \\ -\mu rk & 1+\mu 2rk & -\mu rk & & & \\ & \ddots & \ddots & \ddots & & \\ & & -\mu rk & 1+\mu 2rk & -\mu rk & \\ & & & -\mu rk & 1+\mu 2rk & \end{bmatrix}, \tag{2.33}$$

we can then write the scheme as

$$BU^n = R, \tag{2.34}$$

where both the vector R and the matrix$B$ are dependent on μ and R stands for the right hand side and not to be confused with the residual. Let

$$U^n = \begin{bmatrix} U_1^n \\ \vdots \\ U_{M-1}^n \end{bmatrix} \qquad \text{and} \qquad R = \begin{bmatrix} R_1 \\ \vdots \\ R_{M-1}. \end{bmatrix} \tag{2.35}$$

In the algorithm, a matrix Q was used to help calculate R_j in equation (2.36) for $j = 1, \ldots, N$,

$$Q = \begin{bmatrix} 1-2rk(1-\mu) & rk(1-\mu) & & & & \\ rk(1-\mu) & 1-2rk(1-\mu) & rk(1-\mu) & & & \\ & \ddots & \ddots & \ddots & & \\ & & rk(1-\mu) & 1-2rk(1-\mu) & rk(1-\mu) & \\ & & & rk(1-\mu) & 1-2rk(1-\mu) & \end{bmatrix} \tag{2.36}$$

Both A and Q are $N \times N$ matrices. A vector, P, was used to incorporate the

boundary conditions

$$P = \begin{bmatrix} \mu rkh_l(t_n) + rk(1-\mu)h_l(t_{n-1}) \\ 0 \\ \vdots \\ 0 \\ \mu rkh_r(t_n) + rk(1-\mu)h_r(t_{n-1}) \end{bmatrix}. \tag{2.37}$$

The vector R is then calculated using Q, P, the right hand side equation, f, as seen in equation (2.38) below. In equation (2.38), x is a vector of the interior points, length $M-1$. Then R and matrix A are used to calculate U^n in equation (2.39)

$$R = QU^{n-1} + P + \tau(\mu f(x, t_n) + (1-\mu)f(x, t_{n-1})) \tag{2.38}$$

$$U^n = B^{-1}R. \tag{2.39}$$

The algorithm steps through this process until the specified end time is reached.

Using the same technique as in section 2.1. by using a Taylor expansion and putting in the true solution to the PDE into the numerical approximation, the LTE for this method can be calculated. The LTE of the implicit and explicit methods is $\mathcal{O}(h^2 + \tau)$ and for the Crank-Nicolson is $\mathcal{O}(h^2 + \tau^2)$. For a stable method, the global error behaves like the LTE.

2.4.2 Fully implicit algorithm for diffusion equation with Dirichlet boundary conditions

The governing equation for this section is equation (2.30). We write out the fully implicit scheme in a different way which is more convenient later. The main differences between this and the previous method are the way the boundary conditions are implemented and that this method is only implemented fully implicitly.

The problem has inputs M, a, b, T, dt, k correspondingly number of x steps, left end

point, right end point, final time, time step size, τ, and the diffusion coefficient. The functions f, h_l and h_r can be manually changed inside the algorithm. This algorithm calculates the error, and plots the calculated solution. This method models the PDE in equation (2.30), with the finite difference method

$$U_j^n + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) = \tau f_j^n + U_j^{n-1} \tag{2.40}$$

which is equation (2.31) with $\mu = 1$. Note that $f_j^n = f(x_j, t_n)$ and all other notation is the same as that in section 2.4.1, with h and τ as given in equation (2.32).

Writing this out in matrix form we have.

$$\left(\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \\ & & & & 1 \end{bmatrix} - k\frac{\tau}{h^2}\begin{bmatrix} 0 & & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & & 0 \end{bmatrix}\right)\begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_M^n \\ U_{M+1}^n \end{bmatrix} = \begin{bmatrix} 0 \\ \tau f \\ \vdots \\ \tau f \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ U_2^{n-1} \\ \vdots \\ U_M^{n-1} \\ 0 \end{bmatrix} \tag{2.41}$$

and let

$$B = \begin{bmatrix} 1 & & & & \\ -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2} & -k\frac{\tau}{h^2} & & \\ & \ddots & \ddots & \ddots & \\ & & -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2} & -k\frac{\tau}{h^2} \\ & & & & 1 \end{bmatrix}.$$

This method includes the boundary nodes, so the vectors are length $M+1$ as opposed to the length $M-1$ vectors us in the method in section 2.4.1 which does not include the

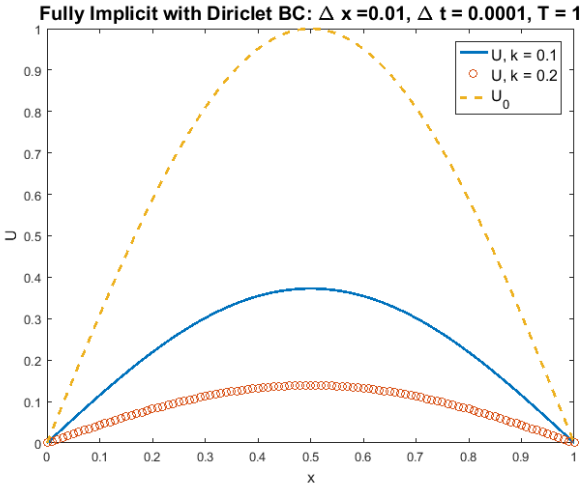

FIGURE 2.5: Numerical solution using fully implicit finite difference algorithm for the linear diffusion equation from section 2.4.2 where $f = 0$, $a = 0$, $b = 1$, $T = t_{final} = 1$, $u_0 = \sin(\pi x)$. Plotted are numerical solutions for two diffusion coefficients, $k = 0.1$ and $k = 0.2$.

boundary node. Then equation (2.41) can be rewritten as

$$BU^n = \tau f(\mathbf{x}, t_n) + U^{n-1}. \tag{2.42}$$

Where $\mathbf{x}$ is a vector containing all x nodes. Equation (2.42) can be solved using a linear solver since the right hand side of (2.42) is a know vector and the matrix B is a known invertible matrix.

To get a feel for how the solution behaves, Figure 2.5 was created using values $a = 0$, $b = 1$, $t_{final} = 1$ and $\tau = h^2$. Figure 2.5 was created in MATLAB using the build-in linear solver mldivide, i.e. \, to solve (2.42) at each time step. The calculated solution for two values of the diffusion coefficient k were plotted as well as the initial condition.

2.4.3 Convergence and comparison

We check now that the two schemes presented above are convergent. To compare the two methods and calculate the order of convergence of the methods, consider the

example,

$$u_t - u_{xx} = 0 \qquad x \in (0,1),\, t > 0 \tag{2.43a}$$

$$B.C. \qquad u(0,t) = 0, \qquad u(1,t) = 0 \quad t > 0 \tag{2.43b}$$

$$I.C. \qquad u(x,0) = u_0(x) = \sin(\pi x) \quad x \in (a,b). \tag{2.43c}$$

Note this is equation (2.30) with $f = 0$, $k = 1$ and u_0 specified. The analytic solution to (2.43) is $u = \sin(\pi x)e^{-\pi^2 t}$.

Table 2.3 shows the error and calculated order of accuracy for h, α for given h and τ using the fully implicit diffusion equation algorithm.

The error value, e_∞, was calculated using the known solution, u. In words, $e_\infty(h)$ was calculated by taking the maximum absolute difference of the calculated solution and the exact solution at each time step then taking the maximum of all the time steps, for a given spatial step h and $\tau \sim h^2$. In symbols,

$$e_\infty(h) = \max_n ||U^n - u^n||_\infty = \max_n \max_j |U_j^n - u_j^n|. \tag{2.44}$$

The order of convergence was calculated using

$$\alpha_i = \frac{\log e_\infty(h_{i-1}) - \log e_\infty(h_i)}{\log h_{i-1} - \log h_i} \tag{2.45}$$

where the lower index is to indicate different values of h used to calculate the order of convergence α. The expected value for α is 2, since both of these methods have an error of $\mathcal{O}(h^2 + \tau)$.

Both methods, (2.31) with $\mu = 1$ and (2.40), gave identical error. This is to be expected since their only difference when $\mu = 1$ is how the boundary nodes are implemented. Tables 2.3 and 2.4, were created using the identical values calculated using both methods

h	τ	e_∞	α
0.1	0.01	0.0203	-
0.01	0.0001	2.1171e-04	1.9832
0.002	4.00e-06	8.4718e-06	1.9982

TABLE 2.3: Error, e_∞ as in equation (2.44), and order of convergence, α as in (2.45) for finite difference methods (2.31) and(2.40).

h	τ	$e_{2,\Delta}$	α
0.1	0.01	0.014369	-
0.01	0.0001	1.4970e-4	1.9822
0.002	4.00e-06	5.9905e-06	1.9997

TABLE 2.4: Error, $e_{2,\Delta}$ as in equation (2.46), and order of convergence, α as in (2.45) for finite difference methods (2.31) and(2.40).

with $\tau \sim h^2$.

Table 2.4 shows the error results of the e_{ℓ_2}, where

$$e_{2,\Delta}(h) = \max_n ||U^n - u^n||_{2,\Delta} = \max_n \{\sqrt{h}\Big(\sum_{j=1}^{M^*} |U_j^n - u_j^n|^2\Big)^{\frac{1}{2}}\} \tag{2.46}$$

where M^* is $M-1$ for method (2.31) and $M+1$ for (2.40). These numbers were calculated using the built in norm function in MATLAB.

2.5. Newton's method

In this Section we provide details on Newton's method, a nonlinear solver used for fully implicit schemes applied to nonlinear ODEs or PDEs.

Newton's method is an iterative method used to find roots of $C_1(\mathbb{R}^n)$ functions, where $C_1(\mathbb{R}^n)$ stands for continuous functions with continuous partial derivatives.

To use Newton's method, the system of equations needs to be written in residual

form as in equation (2.47),

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) & = f_1(\mathbf{x}) = 0 \\ f_2(x_1, x_2, \ldots, x_n) & = f_2(\mathbf{x}) = 0 \\ & \vdots \\ f_m(x_1, x_2, \ldots, x_n) & = f_n(\mathbf{x}) = 0, \end{cases} \tag{2.47}$$

where $\mathbf{x} \in \mathbb{R}^n$. Newton's method involves moving along linear approximations to try and find solutions to (2.47). To begin the iterations an initial guess, $x^{(0)}$ is needed; in a time dependent problem, this initial guess is often taken to be the previous time step value. The system of equations, (2.47), in then evaluated at this initial guess, to obtain the residual vector $\mathbf{R}$,

$$\begin{bmatrix} f_1(\mathbf{x^{(0)}}) \\ f_2(\mathbf{x^{(0)}}) \\ \vdots \\ f_m(\mathbf{x^{(0)}}) \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix} = \mathbf{R}. \tag{2.48}$$

Next, check if

$$||\mathbf{R}||_\infty < \text{tol} \tag{2.49}$$

where tol is a predetermined tolerance, dependent on how precise the solution to equation (2.47) needs to be. If equation (2.49) holds, then the initial guess is accepted as the solution. If not, then a new guess is calculated. The way a new guess is calculated if described in the section 2.5.1 for the case $n = m = 1$, and in Section 2.5.2 for the case $n = m > 1$.

2.5.1 Newton's method in one dimension

First consider the case where $n = m = 1$. Then the linear approximation from $f(x^{(0)}$ for $f(x)$ is

$$f(x) \approx f(x^{(0)}) + f'(x^{(0)})(x - x^{(0)}). \tag{2.50}$$

Now since for Newton's method we are solving for a root of f as written in (2.47), we want to know the x such that $f(x) = 0$. Call this x that we are solving for $x^{(1)}$. The next step is to set $f(x^{(0)}) = 0$ and try and solve for $x^{(1)}$, giving

$$f(x^{(0)}) \approx f'(x^{(0)})(x^{(0)} - x^{(1)}). \tag{2.51}$$

We can know $f(x^{(0)})$ and $f'(x^{(0)})$ so we can solve for $x^{(1)}$, let

$$x^{(1)} = x^{(0)} - \frac{f(x^{(0)}}{f'(x^{(0)})}. \tag{2.52}$$

With this linear approximation $f(x^{(1)}) \approx 0$. We can continue in a similar fashion using $x^{(1)}$ as the guess and solving for $x^{(2)}$ to get a better guess of the root. In other words $x^{(1)}$ becomes our new guess so we begin again by calculating the residual, then check if $||\mathbf{R}||_\infty <$ tol and so on. This process continues until either, (i) the residual's norm is less than the specified accuracy value or, (ii) the specified maximum number of iterations is met,. In both of these cases the Newton iterations cease. In case one, the current iteration's guess, $x^{(i)}$, becomes the numerical estimate from the solution. In the other case an error message occurs.

2.5.2 Newton's method in multiple dimensions

In multiple dimension, the process is the same, however $f'(x)$ is the Jacobian of f

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}. \tag{2.53}$$

Thus, equation (2.51) can be written as

$$f(\mathbf{x^{(0)}}) \approx J_f(\mathbf{x^{(0)}})s \tag{2.54}$$

where $\mathbf{s} = \mathbf{x^{(0)}} - \mathbf{x^{(1)}}$. A linear solver is then used to solve for $\mathbf{s}$ and then $\mathbf{x^{(1)}} = \mathbf{x^{(0)}} - \mathbf{s}$. Then, just as in the one dimensional case in section 2.5.1, the process continues and $\mathbf{x^{(k)}}$ becomes the new guess until a solution is found or the maximum number of iterations is met.

3. NUMERICAL METHODS FOR NONLINEAR PARABOLIC PDES

The goal of this Chapter is to look at finite difference methods for nonlinear parabolic PDEs of the form

$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\Big(k\frac{\partial u}{\partial x}\Big) + g(u) = f(x,t),\ x \in \Omega, t > 0, \tag{3.1}$$

with a variety of initial and boundary conditions and where $\Omega \subset \mathbb{R}$ is open and bounded. Equation (3.1) is the heat or diffusion equation, with a nonlinear "reaction term" $g(u)$.

The Chapter will then go on to explore the impacts of the diffusion coefficient k, the forcing term f, different boundary conditions, $g(u)$ and different way to implement $g(u)$ in the algorithm.

3.1. Nonlinear diffusion equation, fully implicit

When $g(u)$ is nonlinear, equation (3.1) becomes a nonlinear PDE. Methods, similar to those discussed in Section 2.4., may be used if $g(u)$ is evaluated at a previous time step, for which a numerical solution U is already known. Methods which use a previous time step are called time lagging or explicit methods. This section will discuss implicit methods which involve both $g(u)$ and the spatial derivative approximation being evaluated at the current time step being calculated. Implicit methods require algorithms more involved than those in Section 2.4., in this work an iterative method using Newton's method will be used to implicitly calculate numerical solutions to (3.1).

3.1.1 Nonlinear diffusion equation with Dirichlet boundary conditions

This section will discuss nonlinear diffusion equations of the form

$$u_t - ku_{xx} + g(u) = f \qquad x \in (a,b),\ t > 0 \tag{3.2a}$$

$$B.C. \quad u(a,t) = h_l(t) = 0, \qquad u(b,t) = h_r(t) = 0 \quad t > 0 \tag{3.2b}$$

$$I.C. \qquad u(x,0) = u_0(x) \qquad x \in (a,b) \tag{3.2c}$$

were f is a function of both x and t and $k > 0$ is a scalar constant. The discretization used for this problem was a fully implicit method,

$$U_j^n - U_j^{n-1} + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) + \tau g(U_j^n) = f(x_j, t_n), \text{ for } 2 \le j \le M,\ n > 0, \tag{3.3a}$$

$$U_1^n = h_l(t_n) = 0, \quad \text{for } n > 0, \tag{3.3b}$$

$$U_{M+1}^n = h_r(t_n) = 0, \quad \text{for } n > 0, \tag{3.3c}$$

where h and τ are as before in equation (2.32) spatial index is $j = 1, 2, \ldots, M+1$ and the time domain indexing $n = 1, 2, \ldots, N$, just as in section 2.4.2. The notation $t_n = n\tau + t_0$ where t_0 is typically 0 and $x_j = a + jh$ is the x value at the j^{th} node.

Because $g(u)$ is not linear in u, using a fully implicit scheme requires an iterative method. Consider the method (3.3) written in residual form

$$R_j^{n_k} = U_j^n - U_j^{n-1} + k\frac{\tau}{h^2}(2U_j^{n_k} - U_{j-1}^{n_k} - U_{j+1}^{n_k}) + \tau g(U_j^{n_k}) - \tau f(x_j, t_n) \tag{3.4a}$$

and for the boundary

$$R_1^{n_k} = U_1^{n_k} - u(x_1, t_n) \tag{3.4b}$$

$$R_{M+1}^{n_k} = U_{M+1}^{n_k} - u(x_{M+1}, t_n) \tag{3.4c}$$

where n_k represents the time step and the Newton iteration, i.e. n is the time step and the subscript k is the Newton iteration. For detail on Newton's Method see section 2.5..

In this case $u(x_1, t_n) = u(x_{M+1}, t_n) = 0$ for all $t_n > 0$. The Jacobian is as follows

$$J = \begin{bmatrix} 1 & & & & & \\ -k\frac{\tau}{h^2} & 1 + 2k\frac{\tau}{h^2} + g'(U_2^{n_k}) & -k\frac{\tau}{h^2} & & & \\ & \ddots & \ddots & \ddots & & \\ & & -k\frac{\tau}{h^2} & 1 + 2k\frac{\tau}{h^2} + g'(U_M^{n_k}) & -k\frac{\tau}{h^2} & \\ & & & & 1 & \end{bmatrix}.$$

3.1.2 Example

Next, consider equations (3.2) with

$$g(u) = u^3 - u \tag{3.5}$$

$$f(x,t) = \big(1 + (k\pi^2 + 1)(t+1)\big)\sin(\pi x) - \big((t+1)\sin(\pi x)\big)^3. \tag{3.6}$$

The exact solution for this system of equations is

$$u = (t+1)\sin(\pi x). \tag{3.7}$$

Plots for the PDE (3.2) are included in figures 3.1, 3.2, and 3.3. Figure 3.1 was created using (3.3) with $g(u)$ and f as in (3.5) and (3.6). Plots for $f \equiv 0$ and $g(u)$ as in (3.5) and with $g \equiv 0$ and $f \equiv 1$ are included as figures , 3.2, and 3.3 respectively. The figure for when $f \equiv 0$ and $g \equiv 0$ is discussed in section 2.4.2 as figure 2.5.

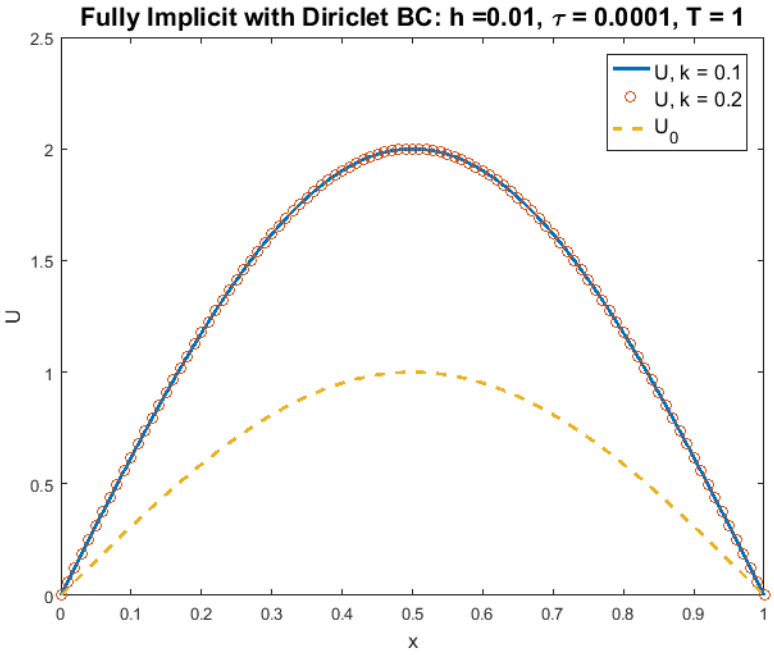

FIGURE 3.1: Plot related to section 3.1.1 using the method in equations (3.3) with $g(u)$ and f as in (3.5) and (3.6).

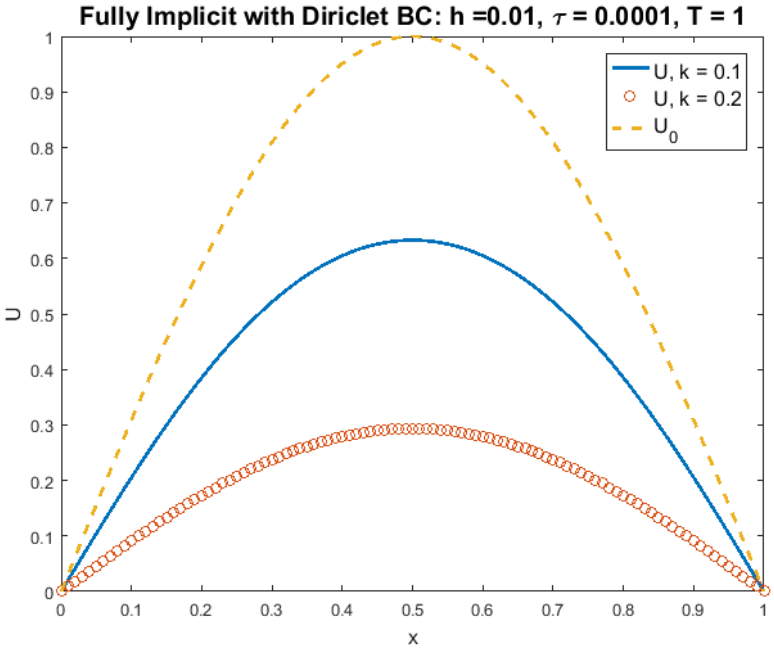

FIGURE 3.2: Plot related to section 3.1.1 using the method in equations (3.3) with $g(u)$ as in (3.5) and $f \equiv 0$.

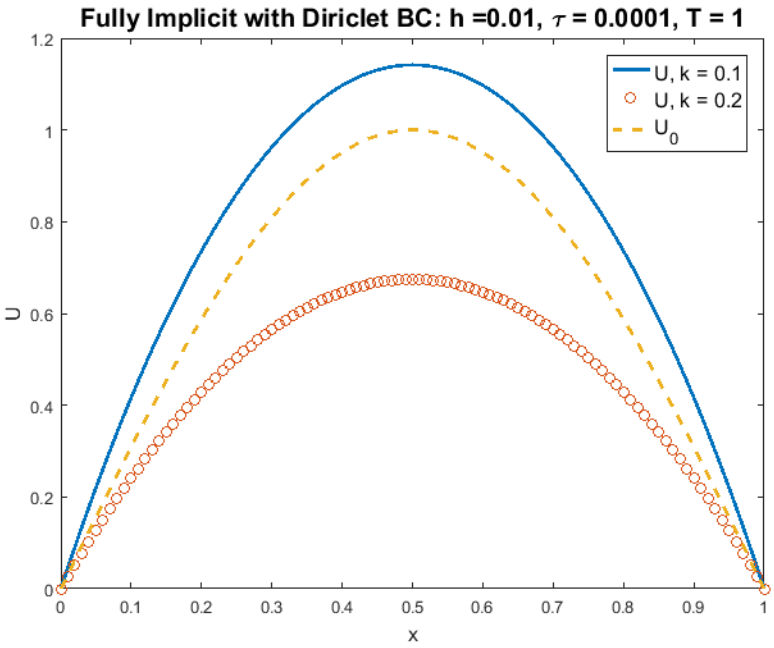

FIGURE 3.3: Plot related to section 3.1.1 using the method in equations (3.3) with $g(u) \equiv 0$ and $f \equiv 1$.

3.1.3 Nonlinear diffusion equation with Neumann boundary conditions

Neumann boundary conditions specify the derivative at the boundary. For a physical example, if Dirichlet boundary conditions specify a temperature a boundary then Neumann boundary conditions can be used to specify the heat flux. If the end of the one dimensional temperature model are perfectly insulated the heat flux, ku', would be set to zero. The boundary condition is the only thing that differs between equations (2.30) and equation (3.8). The initial boundary value problem with Neumann boundary conditions is

$$u_t - ku_{xx} + g(u) = f \qquad x \in (a,b),\ t > 0, \tag{3.8a}$$

$$B.C. \qquad u'(a,t) = 0, \quad u'(b,t) = 0 \quad t > 0, \tag{3.8b}$$

$$I.C. \qquad u(x,0) = u_0(x) \qquad x \in (a,b). \tag{3.8c}$$

There are only a couple of changes between the algorithm for Neumann and Dirichlet boundary condition. These changes are the boundary conditions and the Jacobian. Below is an implicit numerical scheme for equation (3.8). First is the interior nodes, followed by the boundary conditions. For $j = 2, 3, \ldots, M$

$$R_j^{n_k} = U_j^{n_k} - U_j^{n-1} + k\frac{\tau}{h^2}(2U_j^{n_k} - U_{j-1}^{n_k} - U_{j+1}^{n_k}) + \tau g(U_j^{n_k}) - \tau f(x_j, t_n) \tag{3.9a}$$

and for the boundary, $j = 1$ and $j = M + 1$

$$R_1^{n_k} = U_1^{n_k} - U_1^{n-1} + k\frac{\tau}{h^2}(2U_1^{n_k} - 2U_2^{n_k}) + \tau g(U_1^{n_k}) - \tau f(x_1, t_n) \tag{3.9b}$$

$$R_{M+1}^{n_k} = U_{M+1}^{n_k} - U_{M+1}^{n-1} + k\frac{\tau}{h^2}(2U_{M+1}^{n_k} - 2U_M^{n_k}) + \tau g(U_{M+1}^{n_k}) - \tau f(x_{M+1}, t_n) \tag{3.9c}$$

As before, in equation (3.4b), the notation n_k is used to denote an iterative method. Equations (3.9b) and (3.9c) are a second order method for Neumann boundary conditions. The method is derived by approximating the derivative at $a = x_1$ using a ghost node at x_0 by taking the average of the spatial derivative approximation on either side of the first node. The average of the derivative approximations is

$$\frac{1}{2}\left(\frac{U_2^{n_k} - U_1^{n_k}}{h} + \frac{U_1^{n_k} - U_0^{n_k}}{h}\right) = \frac{U_2^{n_k} - U_0^{n_k}}{2h}. \tag{3.10}$$

Setting equation (3.10) equal to zero as specified in the boundary conditions stated in equation (3.8) and doing some algebraic manipulations give that $U_2^{n_k} = U_0^{n_k}$. Since x_0 is a ghost node, $U_0^{n_k}$ does not exist; thus, it is replaced with $U_2^{n_k}$ in equation (3.9) to obtain equation (3.9b). Similarly, for the right end point at x_{M+1}, using a ghost node approximation gives equation (3.9c). The Jacobian for this method is

$J =$

$$\begin{bmatrix} 1+2k\frac{\tau}{h^2}+\tau g'(U_1^{n_k}) & -2k\frac{\tau}{h^2} & & & & \\ -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2}+g'(U_2^{n_k}) & -k\frac{\tau}{h^2} & & & \\ & \ddots & \ddots & \ddots & & \\ & & -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2}+g'(U_M^{n_k}) & -k\frac{\tau}{h^2} & \\ & & & -2k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2}+\tau g'(U_{M+1}^{n_k}) \end{bmatrix}.$$

A simplified, first order, method can be used to approximate the Neumann boundary conditions. Instead of equation (3.10), to approximate the derivative,

$$\frac{U_2^{n_k}-U_1^{n_k}}{h}. \tag{3.11}$$

Just as in the second order method, if equation (3.11) is set to zero, then simplifying give $U_2^{n_k}=U_1^{n_k}$. Thus,

$$R_1^{n_k} = U_2^{n_k} - U_1^{n_k} \tag{3.12}$$

$$R_{M+1}^{n_k} = U_{M+1}^{n_k} - U_M^{n_k} \tag{3.13}$$

is the residual form of a first method for Neumann boundary conditions. The Jacobian for this method is

$$J = \begin{bmatrix} -1 & 1 & & & & \\ -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2}+g'(U_2^{n_k}) & -k\frac{\tau}{h^2} & & & \\ & \ddots & \ddots & \ddots & & \\ & & -k\frac{\tau}{h^2} & 1+2k\frac{\tau}{h^2}+g'(U_M^{n_k}) & -k\frac{\tau}{h^2} & \\ & & & -1 & 1 \end{bmatrix}.$$

3.1.4 Nonlinear diffusion equation, time lagging

For the time lagging method, the function $g(u)$ is evaluated at U^{n-1}, thus the time lagging numerical method for equation (3.8) is

$$U_j^n - U_j^{n-1} + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) + \tau g(U_j^{n-1}) = \tau f(x_j, t_n). \tag{3.14a}$$

for the interior nodes. For the boundary nodes

$$U_1^n - U_1^{n-1} + k\frac{\tau}{h^2}(2U_1^n - 2U_2^n) + \tau g(U_j^{n-1}) = \tau f(x_1, t_n), \tag{3.14b}$$

$$U_{M+1}^n - U_{M+1}^{n-1} + k\frac{\tau}{h^2}(2U_{M+1}^n - 2U_M^n) + \tau g(U_{M+1}^{n-1}) = \tau f(x_{M+1}, t_n). \tag{3.14c}$$

This method does not require an iterative method and can be solved similarly to how it is in equation (2.42), however can be less stable than a fully implicit method.

3.1.5 Nonlinear diffusion equation, splitting method with Neumann boundary conditions

The main idea for the splitting method is to break up the nonlinearity, in our case the $g(u) = u^3 - u$, into an expansive and an contractive term and evaluate the contractive term implicitly and the expansive term explicitly. This is more explicitly discussed in the section 2.1.. The first splitting method used is

$$U_j^n - U_j^{n-1} + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) + \tau 5U_j^n = \tau 6U_j^{n-1} - \tau(U_j^{n-1})^3 + \tau f_j^n. \tag{3.15a}$$

and for the boundaries

$$U_1^n - U_1^{n-1} + k\frac{\tau}{h^2}(2U_1^n - 2U_2^n) + \tau 5U_1^n = \tau 6U_1^{n-1} - \tau(U_1^{n-1})^3 + \tau f_1^n, \tag{3.15b}$$

$$U_{M+1}^n - U_{M+1}^{n-1} + k\frac{\tau}{h^2}(2U_{M+1}^n - 2U_M^n) + \tau 5U_{M+1}^n = \tau 6U_{M+1}^{n-1} - \tau(U_{M+1}^{n-1})^3 + \tau f_{M+1}^n. \tag{3.15c}$$

3.2. Comparing fully implicit, time lagging, and the splitting methods for the nonlinear diffusion equation

This section compares the fully implicit (Section 3.1.3), time lagging (Section 3.1.4), and the splitting method (Section 3.1.5) for the nonlinear diffusion equation (3.8). The

first example explored is

$$u_t - ku_{xx} + u^3 - u = 0 \qquad x \in (a,b),\, t > 0 \tag{3.16a}$$

$$B.C. \quad u'(0,t) = 0, \quad u'(1,t) = 0 \qquad t > 0 \tag{3.16b}$$

$$I.C. \quad u(x,0) = \begin{cases} -1 & \text{for } x = (0, \frac{1}{2}), \\ 0 & \text{for } x = \frac{1}{2}, \\ 1 & \text{for } x = (\frac{1}{2}, 1). \end{cases} \tag{3.16c}$$

Figure 3.4 shows some plot for this example. Figure 3.5 shows plots for equation (3.16) with the initial value changed to

$$u_0 = \frac{1}{3}\cos(7\pi x) + \frac{1}{3}\cos(29\pi x) + \frac{1}{3}\cos(\pi x). \tag{3.17}$$

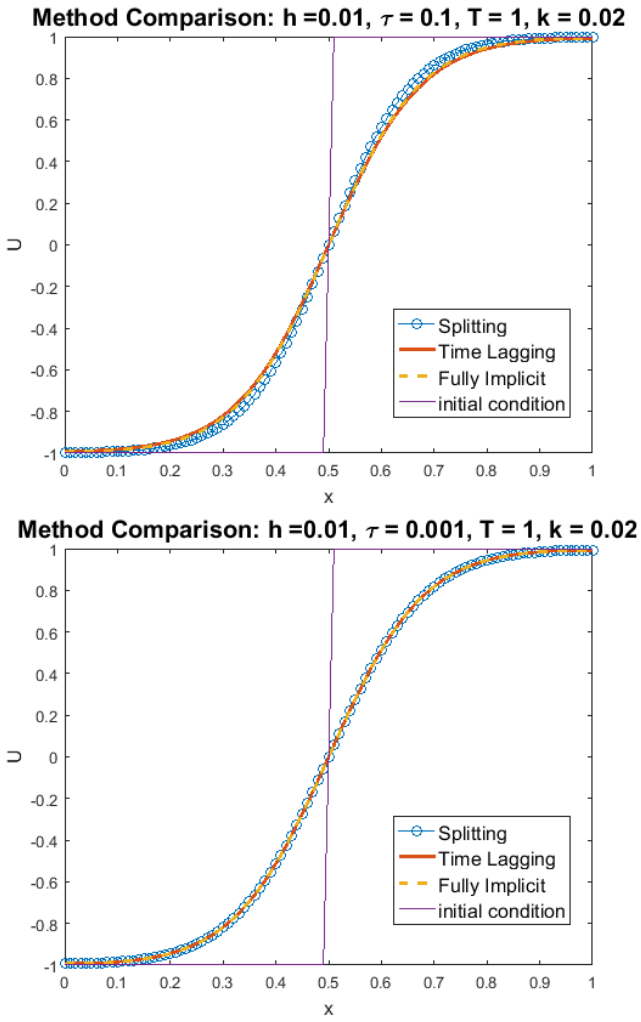

FIGURE 3.4: Plots for all three methods, fully implicit, time lagging and splitting for two different values of τ for equation (3.16).

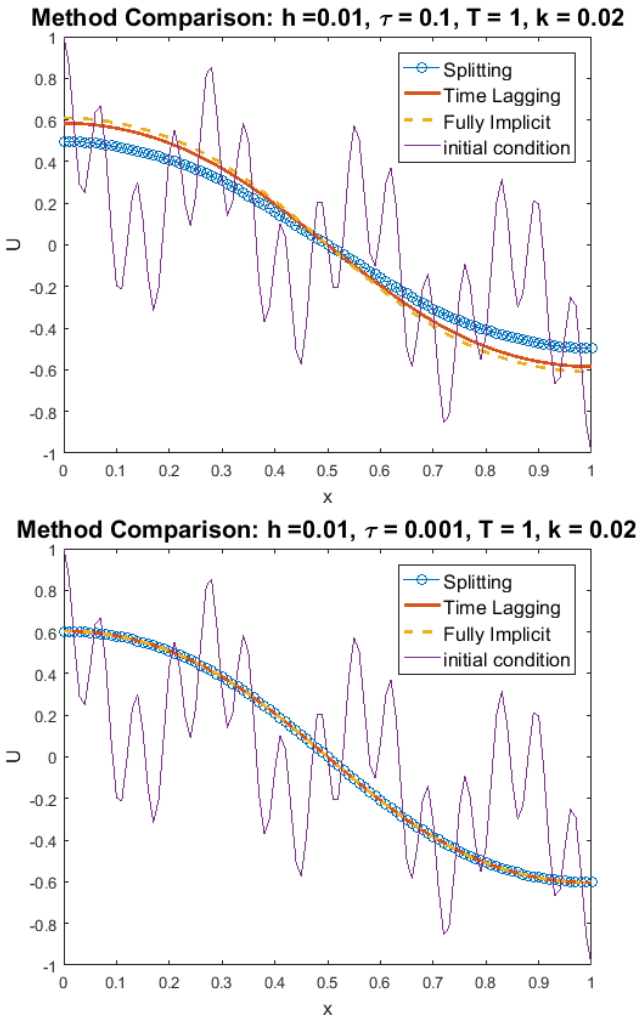

FIGURE 3.5: Plots for all three methods, fully implicit, time lagging and splitting for two different values of τ for equation (3.16) with u_0 as in equation (3.17).

4. PHASE FIELD MODELS

The goal of this section is to give an overview the of Stefan problem and write it both in the equilibrium form and in the phase field form. The equilibrium form is when the order parameter $\varphi \in \text{sign}(u)$, where $u(x,t)$ is the temperature. The phase field form is when the order parameter has its own dynamics described by an ODE or PDE.

First in section 4.1. we explain how the strong form of the equilibrium form of the energy equation with the Stefan condition on the interface is "translated" to the weak form which includes φ under the time derivative. We next introduce the phase field model coupled to the energy equation, and show how to solve it numerically with different numerical schemes.

4.1. Stefan problem

Here we give an overview the Stefan problem using notation mostly from Guenther and Lee's book, [5] and connecting it to the analysis used in [8].

The Stefan problem describes change-of-phase problems. The problem addressed in this section will be a phase change between water and ice. At the initial time the problem discussed here consists of an insulated pipe of length two with the left half filled with water and the right half filled with ice with heat applied to just the left end.

Let the specific heat, density, thermal conductivity, and temperature of water be denoted respectively by c_w, ρ_w, k_w, and u_w. Similarly for ice with i instead of w. Let $s(t)$ be the location of the ice-water interface at any given time. Our objective is to find the temperature distribution over time, $t > 0$.

Both the ice section and the water section satisfy the heat equation. Combining the constants, let $a_w = k_w/c_w\rho_w$ and similarly for ice. Then we have a heat equation written

separately in each material

$$\begin{cases} \frac{\partial u_w}{\partial t} = a_w \frac{\partial^2 u_w}{\partial x^2} & 0 < x < s(t), \quad t > 0 \\ \frac{\partial u_i}{\partial t} = a_i \frac{\partial^2 u_i}{\partial x^2} & s(t) < x < 2, \quad t > 0. \end{cases} \tag{4.1}$$

In what follows the w and i subscripts of u will be dropped. Let the temperature at the ice-water interface be zero,

$$u\big|_{s(t)^-} = u\big|_{s(t)^+} = 0 \tag{4.2}$$

and let

$$a u_x\big|_{s(t)^-} = a u_x\big|_{s(t)^+} - \mathcal{L}\dot{s} \quad t \geq 0 \tag{4.3}$$

where $\dot{s}$ is the time derivative of the position of the phase transition. Equation (4.3) is the well known Stefan condition which requires that additional input of heat proportional to the latent heat $\mathcal{L}$ is provided in order to change the phase at $u = 0$ from ice to water. This additional amount of heat is linked to the speed of the melting front $\dot{s}$.

To solve this problem numerically, one would have to account explicitly for the position of the interface. Another, weak formulation is more convenient. We derive it now.

Approaching this system in a weak sense, let $\varphi \in C_0^\infty(Q)$ where $Q = (0,2) \times (0,T)$ and $u \in H^1(Q)$. Changing notations slightly let

$$u_t - \nabla(k\nabla u) = 0 \qquad \text{on } \Omega_1 \cup \Omega_2 \tag{4.4a}$$

$$-[k\nabla u \cdot \nu] = \mathcal{L} v \cdot \nu \qquad \text{on } S \tag{4.4b}$$

$$u = 0 \qquad \text{on } S \tag{4.4c}$$

Here $Q = \Omega \times (0,T)$. The spatial domain is denoted as Ω, where Ω_i, $i = 1,2$, are the domains of each phase, letting Ω_1 be water and Ω_2 be ice. The location of the phase change S is equal to $S := \partial Q_1 \cap \partial Q_2$. The latent heat is denoted as $\mathcal{L}$ and v is the velocity of the interface, given as $\mathcal{L}$ and $\dot{s}$ respectively. The value k is also set in each Ω_i and is denoted a above. The vector ν is the unit vector field normal to $S \cap \Omega \times \{t\}$. Let n_x be the unit normal vector in the spatial domain and n_t be the unit normal vector time.

Now to formulate the weak form of the Stefan problem we consider multiplying (4.4a) by φ. Let $\varphi \in C_0^\infty(Q)$ and first consider

$$\begin{aligned}
\iint_{Q_1} u_t \varphi dx dt &= -\iint_{Q_1} u\varphi_t dx dt + \int_{\partial Q_1} u\varphi dS \\
&= -\iint_{Q_1} u\varphi_t dx dt + \int_{\partial Q_1 \setminus S} u\varphi dS + \int_S u\varphi dS \\
&= -\iint_{Q_1} u\varphi_t dx dt
\end{aligned}$$

since $u = 0$ on S and $\varphi = 0$ on $\partial Q_1 \setminus S$. This holds for Q_2 as well.

Next we sum over both domains. Denoting $Q_{1,2} := Q_1 \cup Q_2$ we begin with

$$\begin{aligned}
0 = &\iint_{Q_1} u_t \varphi dx dt + \iint_{Q_1} -\nabla(k\nabla u)\varphi dx dt + \iint_{Q_2} u_t \varphi dx dt + \iint_{Q_2} -\nabla(k\nabla u)\varphi dx dt \\
= &\iint_{Q_{1,2}} u_t \varphi dx dt + \iint_{Q_1} (k\nabla u)\nabla\varphi dx dt - \int_{\partial Q_1} \varphi(k\nabla u)\cdot n_x^1 dS + \iint_{Q_2} (k\nabla u)\nabla\varphi dx dt \\
&- \int_{\partial Q_2} \varphi(k\nabla u)\cdot n_x^2 dS
\end{aligned}$$

calculated using Greens theorem in the spatial domain and n_x^i, $i = 1,2$, are the normal vectors out of the respective Ω_i and let $n_x = n_x^1 = -n_x^2$. Note as above $\varphi = 0$ on the boundaries that are not S. Also note that equation (4.4b) can be rewritten noting with

n_x instead of ν, thus

$$\begin{aligned}
0 &= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_1} (k\nabla u)\nabla\varphi dxdt - \int_{\partial Q_1} \varphi(k\nabla u)\cdot n_x dS + \iint_{Q_2} (k\nabla u)\nabla\varphi dxdt \\
&\qquad + \int_{\partial Q_2} \varphi(k\nabla u)\cdot n_x dS \\
&= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_1} (k\nabla u)\nabla\varphi dxdt + \iint_{Q_2} (k\nabla u)\nabla\varphi dxdt + \int_S -\varphi[(k\nabla u)\cdot n_x]dS \\
&= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt + \int_S \varphi\mathcal{L}v\cdot n_x dS.
\end{aligned}$$

Now consider how $v\cdot n_x$ can be rewritten, buy starting with derivative of u on S which is constant. Thus,

$$\begin{aligned}
0 &= \frac{d}{dt}u(x(t),t) && (4.5)\\
&= \nabla_x u\cdot\frac{dx}{dt} + u_t && (4.6)
\end{aligned}$$

where $\frac{dx}{dt} = v$. Normalizing this vector by $||(\nabla_x u, u_t)||$, where $||\cdot||$ is the Euclidean norm, or the magnitude of the vector, gives a unit vector normal to the spatial and time cylinder, thus

$$0 = v\cdot n_x + n_t. \tag{4.7}$$

Using this relation and the fact the the jump of the Heaviside graph, $[H(u)]$ has a jump at S we get

$$\begin{aligned}
0 &= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt + \int_S \varphi\mathcal{L}v\cdot n_x dS \\
&= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_S \varphi\mathcal{L}n_t dS \\
&= \iint_{Q_{1,2}} u_t\varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_S \varphi\mathcal{L}[H(u)]n_t dS.
\end{aligned}$$

Rewriting the normal in time into $n_t = n_t^1 = -n_t^2$ we get

$$\begin{aligned}
0 &= \iint_{Q_{1,2}} u_t \varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_S \varphi \mathcal{L}[H(u)] n_t^1 dS \\
&= \iint_{Q_{1,2}} u_t \varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_{\partial Q_1} \varphi \mathcal{L}H(u) n_t^1 dS + \int_{\partial Q_2} \varphi \mathcal{L}H(u) n_t^1 dS \\
&= \iint_{Q_{1,2}} u_t \varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_{\partial Q_1} \varphi \mathcal{L}H(u) n_t^1 dS - \int_{\partial Q_2} \varphi \mathcal{L}H(u) n_t^2 dS.
\end{aligned}$$

Since $H(u)$ is constant on Q_1 and Q_2, we have $(H(u))_t = 0$ on Q_1 and Q_2. Adding two integrals that equate to zero gives

$$\begin{aligned}
0 &= \iint_{Q_{1,2}} u_t \varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \int_{\partial Q_1} \varphi \mathcal{L}H(u) n_t^1 dS - \int_{\partial Q_2} \varphi \mathcal{L}H(u) n_t^2 dS \\
&= \iint_{Q_{1,2}} u_t \varphi dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt + \iint_{Q_1} \mathcal{L}(H(u))_t \varphi dxdt - \int_{\partial Q_1} \varphi \mathcal{L}H(u) n_t^1 dS \\
&\qquad + \iint_{Q_2} \mathcal{L}(H(u))_t \varphi dxdt - \int_{\partial Q_2} \varphi \mathcal{L}H(u) n_t^2 dS \\
&= -\iint_{Q_{1,2}} u \varphi_t dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt - \iint_{Q_1,Q_2} \mathcal{L}(H(u))\varphi_t dxdt \\
&= -\iint_{Q_{1,2}} (u + \mathcal{L}H(u))\varphi_t dxdt + \iint_{Q_{1,2}} (k\nabla u)\nabla\varphi dxdt \\
&= \iint_{Q_{1,2}} (u + \mathcal{L}H(u))_t \varphi dxdt - \iint_{Q_{1,2}} \nabla(k\nabla u)\varphi dxdt.
\end{aligned}$$

This implies that in the weak sense we obtain

$$\big(u + \mathcal{L}H(u)\big)_t - \nabla(k\nabla u) = 0. \tag{4.8}$$

Finally, we remark that another way to rewrite the problem (4.8) is to replace $H(u)$

by sign(u).

$$\frac{\partial}{\partial t}\left(u + \frac{\mathcal{L}}{2}\varphi\right) - \nabla \cdot (k\nabla u) = 0 \tag{4.9a}$$

$$\varphi \in \text{sign}(u) \tag{4.9b}$$

This is derived from the fact that $\text{sign}(u) = 2H(u) - 1$ and thus $H(u) = \frac{\text{sign}(u)+1}{2}$ and therefore

$$\begin{aligned} \frac{\partial}{\partial t}H(u) &= \tfrac{\partial}{\partial t}(\mathcal{L}\tfrac{\text{sign}(u)+1}{2}) \\ &= \tfrac{\partial}{\partial t}(\mathcal{L}\tfrac{\varphi+1}{2}) \\ &= \tfrac{\partial}{\partial t}(\mathcal{L}\tfrac{\varphi}{2}). \end{aligned}$$

The formulation (4.9) of Stefan problem is the starting point for the phase field model.

4.2. Phase field model

The phase field model models the behavior of φ in the region close to the interface between two phases. For example, a phase field model could be used to model solid and liquid, or ice and water. The Stefan problem (4.9) uses the sign graph sign(u) at the location of the interface; this allows to introduce the notion that there can be a slush mixture of ice and water close to the interface.

Incorporating a phase field model, can create a more realistic model, and give φ its own dynamics instead of (4.9b).

The phase field model used in this work is the Allen-Cahn equations as used in [10],

$$\varphi_t + \frac{1}{\varepsilon}g(\varphi) - \varepsilon\Delta\varphi = \mathcal{L}u. \tag{4.10}$$

Here $g(\varphi) = \frac{\partial G}{\partial \varphi}$ and G is the double well potential, $G(\varphi) = (\varphi^2 - 1)^2/4$. Thus $g(\varphi) = \varphi^3 - \varphi$. This double well potential has stable equilibria at $\varphi = -1$ and $\varphi = 1$ and an unstable equilibrium at $\varphi = 0$. Recall that we discussed the properties of G and g and of relevant numerical schemes in more details in section 2.1..

Typically Neumann boundary conditions are used to model the fact that order parameter does not leave the domain. In addition, we specify some initial conditions.

4.3. Literature review

The Allen-Cahn equation has been studied since it was introduced in 1979 by the publication of [1]. The format the the Allen-Cahn equation presented in [1] and used in [2, 7] is

$$u_t - \nabla u + \frac{1}{\varepsilon^2} f(u) = 0 \tag{4.11}$$

typically with Dirichlet boundary conditions, suitable initial conditions and with $f(u) = F'(u)$ where $F(u)$ is a given energy potential. In [2, 7], the energy potential is the double well potential as used in this work; that is f is equal to the function g in equation (4.10). In this section, we will discuss the findings of [2, 7, 3], which discuss different aspects of numerical methods for the Allen-Cahn equation.

In [2], a convergence requirement for explicit, or time lagging, finite difference numerical method is discussed. As $\varepsilon \to 0$ equation (4.11) converges to the solution given my mean curvature flow. The question is, do numerical methods do the same and if so are there spatial and time steps size restrictions for this convergence? Article [2] discusses this requirement for a finite difference method to converge to the mean curvature flow as $\varepsilon \to 0$. The finite difference method used in [2] is the same as the one discussed in Section 3.1.4, though the paper includes the factor of ε^2.

Let Ω be the spatial domain, broken up into $\Omega^+(t)$, $\Omega^-(t)$, and $\Gamma(t)$ be where $u \approx 1$, $u \approx -1$ and $-1 < u < 1$ respectively. Consider equation (4.11), with the $u = -1$ on the boundary, $\partial\Omega$, and the initial condition of $\Omega = \Omega^+ \cup \Omega^-$. Then the theorem stated and proven in [2] states that for $p > 1$ if

$$h \leq \varepsilon^p$$
$$\frac{1}{\tau} \geq \frac{2N}{h^2} + \frac{1}{\varepsilon^2}||f'||_{c^0(-2,2)}$$

where N is the dimension of space being modeled, then as $\varepsilon \to 0$ the numerical solution approaches the solution of the motion by mean curvature flow. That is

$$\lim_{\varepsilon \to 0} u^{\varepsilon,h,\tau}(x,t) = \begin{cases} 1, & \text{if } x \in \Omega^+ \\ -1, & \text{if } x \in \Omega^- \end{cases}$$

where $u^{\varepsilon,h,\tau}$ is notation used in [2] to emphasis that u is dependent on ε, h, and τ. The paper then goes into constructing an auxiliary function in order to construct sub and super solutions of which are not considered for this work. This convergence analysis could be used help set stability requirements for sizes of h and τ when testing for convergence to the mean gradient flow for the numerical model of the phase field using a time lagging finite difference model. Further analysis could be used to help determine stable values of ε and τ for coupled system discussed in the following section, Section 4.4. Overall [2] provided thorough proof, however there are many typos included.

In [7], stability of numerical methods for both the Allen-Cahn and Cahn-Hilliard equations are discussed. Instead of taking the finite difference approach as done in this work as well in previously discussed [2], [7] approaches the problem in a weaker sense by working in Sobolev spaces, in particular they consider schemes that are spectral-Galerkin method in space. As shown in Section 4.1., the Stefan problem is well suited for methods

that use the weak formulation. In [7], the error for the weak formulation of the phase filed model is discussed. Galerkin finite element methods could be a way to approach the coupled phase field and temperature system discussed in Section 4.4.. Convexity splitting is considered in [7], with the splitting dependent on a value S. In [7], it is shown that both the stability and the error depend on S. This allows for and adjustable splitting method, where S can be adjusted to give a stable method for any value of the time step but also minimize error, depending on the other parameters. This article, [7], is clearly written, easy to follow and discusses many methods for both the Allen-Cahn and Cahn-Hilliard, including a second order splitting method.

In contrast to the finite difference method discussed in this work, [3] discusses Fourier spectral discretization in space and an error based adjustable implicit time stepping method for the Allen-Cahn is used and compared to a convexity splitting method. The Allen-Cahn equation used in [3] is

$$u_t = \varepsilon^2 \nabla u - f(u)$$

where $f(u)$ is as in (4.11). The adjustable time step method used in [3] was shown to be stable, accurate, and less expensive alternative to fully implicit methods when high accuracy is required, since the splitting method was shown to require small time steps to maintain accuracy.

4.4. Phase field and temperature coupled system

The phase field and energy equation coupled system is

$$\begin{cases} \varphi_t + \frac{1}{\varepsilon} g(\varphi) - \varepsilon \Delta \varphi = \mathcal{L} u \\ \frac{\partial}{\partial t} \left(u + \frac{\mathcal{L}}{2} \varphi \right) - \nabla \cdot (k \nabla u) = 0 \end{cases} \tag{4.12}$$

where the phase field model is equation (4.10) with $f = \mathcal{L}u$ and the energy equation is (4.8) with $H(u)$ replaced with $\frac{\mathcal{L}}{2}\phi$.

The numerical methods for this coupled system is

$$\begin{cases} \phi_j^n - \phi_j^{n-1} + \frac{\tau}{\varepsilon} g(\phi_j^{n_*}) + \varepsilon \frac{\tau}{h^2}(2\phi_j^n - \phi_{j-1}^n - \phi_{j+1}^n) = \tau \mathcal{L} U_j^{n_{**}} \\ U_j^n + \frac{\mathcal{L}}{2}\phi_j^n - U_j^{n-1} - \frac{\mathcal{L}}{2}\phi_j^{n-1} + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) = 0, \end{cases} \tag{4.13a}$$

for the interior nodes and for the boundary nodes,

$$\begin{cases} \phi_1^n - \phi_1^{n-1} + \frac{\tau}{\varepsilon} g(\phi_1^{n_*}) + \varepsilon \frac{\tau}{h^2}(2\phi_1^n - 2\phi_2^n) = \tau \mathcal{L} U_1^{n_{**}} \\ U_1^n + \frac{\mathcal{L}}{2}\phi_1^n - U_1^{n-1} - \frac{\mathcal{L}}{2}\phi_1^{n-1} + k\frac{\tau}{h^2}(2U_1^n - 2U_2^n) = 0, \end{cases} \tag{4.13b}$$

$$\begin{cases} \phi_{M+1}^n - \phi_{M+1}^{n-1} + \frac{\tau}{\varepsilon} g(\phi_{M+1}^{n_*}) + \varepsilon \frac{\tau}{h^2}(2\phi_{M+1}^n - 2\phi_M^n) = \tau \mathcal{L} U_{M+1}^{n_{**}} \\ U_{M+1}^n + \frac{\mathcal{L}}{2}\phi_{M+1}^n - U_{M+1}^{n-1} - \frac{\mathcal{L}}{2}\phi_{M+1}^{n-1} + k\frac{\tau}{h^2}(2U_{M+1}^n - 2U_M^n) = 0. \end{cases} \tag{4.13c}$$

If $n_* = n$, then the numerical system of is considered implicit in ϕ, and if $n_* = n-1$ then the system is considered time lagging in ϕ. Similarly for the temperature U, if $U^{n_{**}} = U^n$ then the system is implicit in U, the temperature, and if $n_{**} = n-1$ then the system is considered time lagging in U. Time lagging is denoted by L and implicit is denoted by I. In the examples to follow, the subscript denotes the variable it is referring to, i.e. $\phi\ L_\phi\, I_U$ is the plot of ϕ using the method that is time lagging in ϕ and implicit in U.

4.4.1 Examples

We now present a few examples. These demonstrate the difference between the fully implicit, time lagging, and convexity splitting approaches. These differences depend on the parameters of the problem.

In Figures 4.1 and 4.2, S stand for the splitting method below for $j = 2, 3, \ldots, M$

$$\begin{cases} \phi_j^n - \phi_j^{n-1} + \varepsilon \frac{\tau}{h^2}(2\phi_j^n - \phi_{j-1}^n - \phi_{j+1}^n) + \frac{\tau}{\varepsilon} 5\phi_j^n = \frac{\tau}{\varepsilon} 6\phi_j^{n-1} - \frac{\tau}{\varepsilon}(\phi_j^{n-1})^3 + \tau \mathcal{L} U_j^{n_{**}} \\ U_j^n + \frac{\mathcal{L}}{2}\phi_j^n - U_j^{n-1} - \frac{\mathcal{L}}{2}\phi_j^{n-1} + k\frac{\tau}{h^2}(2U_j^n - U_{j-1}^n - U_{j+1}^n) = 0, \end{cases} \tag{4.14a}$$

and for the boundary,

$$\begin{cases} \phi_1^n - \phi_1^{n-1} + \varepsilon \frac{\tau}{h^2}(2\phi_1^n - 2\phi_2^n) + \frac{\tau}{\varepsilon} 5\phi_1^n = \frac{\tau}{\varepsilon} 6\phi_1^{n-1} - \frac{\tau}{\varepsilon}(\phi_1^{n-1})^3 + \tau \mathcal{L} U_1^{n_{**}} \\ U_1^n + \frac{\mathcal{L}}{2}\phi_1^n - U_1^{n-1} - \frac{\mathcal{L}}{2}\phi_1^{n-1} + k\frac{\tau}{h^2}(2U_1^n - 2U_2^n) = 0, \end{cases} \tag{4.14b}$$

$$\begin{cases} \phi_{M+1}^n - \phi_{M+1}^{n-1} + \varepsilon \frac{\tau}{h^2}(2\phi_{M+1}^n - 2\phi_M^n) + \frac{\tau}{\varepsilon} 5\phi_{M+1}^n = \frac{\tau}{\varepsilon} 6\phi_{M+1}^{n-1} - \frac{\tau}{\varepsilon}(\phi_{M+1}^{n-1})^3 + \tau \mathcal{L} U_{M+1}^{n_{**}} \\ U_{M+1}^n + \frac{\mathcal{L}}{2}\phi_{M+1}^n - U_{M+1}^{n-1} - \frac{\mathcal{L}}{2}\phi_{M+1}^{n-1} + k\frac{\tau}{h^2}(2U_{M+1}^n - 2U_M^n) = 0. \end{cases} \tag{4.14c}$$

The solutions plotted in both Figures 4.1 and 4.2 correspond to the initial conditions

$$u_0 = 2x - 1, \tag{4.15}$$

$$\varphi_0 = \operatorname{sign}(u_0), \tag{4.16}$$

where $\operatorname{sign}(u)$ is the graph such that

$$\operatorname{sign}(u) = \begin{cases} -1 & \text{for } u < 0, \\ [-1, 1] & \text{for } u = 0, \\ 1 & \text{for } u > 0. \end{cases} \tag{4.17}$$

These initial conditions are included in the Figures for reference, and to demonstrate

the evolution. The boundary conditions for Figures 4.1 and 4.2 are

$$\begin{aligned} u(0,t) &= -3 \qquad \text{and} \qquad \varphi'(0,t) = 0 \\ u(1,t) &= 5 \qquad\qquad\quad \varphi'(1,t) = 0. \end{aligned} \tag{4.18}$$

The difference between Figures 4.1 and 4.2 is the time step value, τ. For 4.1, $\tau = 0.01$ and the numerical solutions for every method are comparable and reasonable. Comparable, in that the solutions are indistinguishable as plotted in 4.1. Reasonable in that with the given boundary conditions the temperature of the system, U, should diffuse until there is a linear temperature distribution. Also, the stable long term behavior for the phase field, ϕ, is that it should cross the plot for U wherever $U = 0$, so that the "mushy" region is centered around the temperature that the phase shift occurs. However, in Figure 4.2, where $\tau = 0.1$ the solution is not stable for the methods that are time lagging in ϕ, labeled L_ϕ in the figures. Also, in Figure 4.1 and other figures for this example, T is the final time, and L is the latent heat. The fully implicit method required 6 Newton iterations while the splitting method can be solved without an iterative method or with one iteration of an iterative method.

In the coupled system of equations (4.12) with specified boundary conditions and initial conditions, there are three variable, $\mathcal{L}$, ε, and k, that impact the behavior of the system. These impacts, along with the impacts of the time step and spatial step values, τ and h, will be explored in the next few paragraphs and in Figures 4.3, 4.4, 4.5, 4.6, 4.7, and 4.8. The previous example, with initial condition as in equations (4.15) and (4.16) and the boundary conditions in (4.18), will be used to explore the impacts of these variables on the coupled system.

The variable values in Figure 4.3 were chosen as a stable middle ground to compare variations in $\mathcal{L}$, ε, k, τ and h against. Figures 4.4, 4.5, 4.6, 4.7, and 4.8 were created by increasing and decreasing $\mathcal{L}$, ε, k, τ or h from those in Figure 4.3.

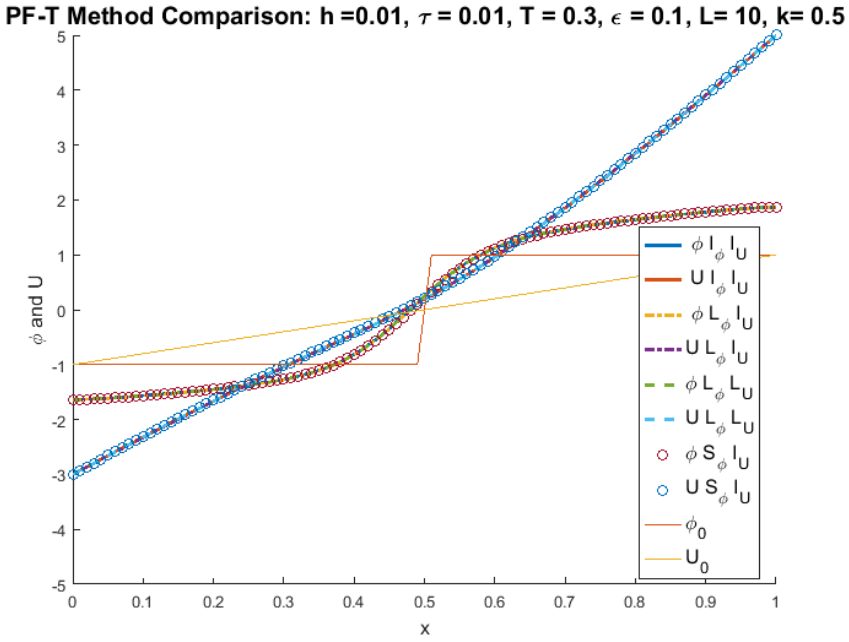

FIGURE 4.1: Image of stable conditions for numerical solutions and initial conditions for (4.12) using fully implicit, time lagging in ϕ, time lagging in both ϕ and U, and the splitting methods.

Figure 4.4 show the impacts of changing the latent heat $\mathcal{L}$. The latent heat is part of the forcing term for the phase field, larger values of $\mathcal{L}$ can force the phase field beyond the stable equilibria of -1, and 1. The latent heat couples the system in equation (4.12), i.e. if $\mathcal{L} = 0$ then

$$\begin{cases} \varphi_t + \frac{1}{\varepsilon} g(\varphi) - \varepsilon \Delta \varphi = 0 \\ \frac{\partial}{\partial t}(u) - \nabla \cdot (k \nabla u) = 0. \end{cases} \tag{4.19}$$

The effect of the strength of this coupling can be seen in Figures 4.4 and 4.3. The implicit method took 4 Newton iterations for the coupled system to create the images in Figures 4.4.

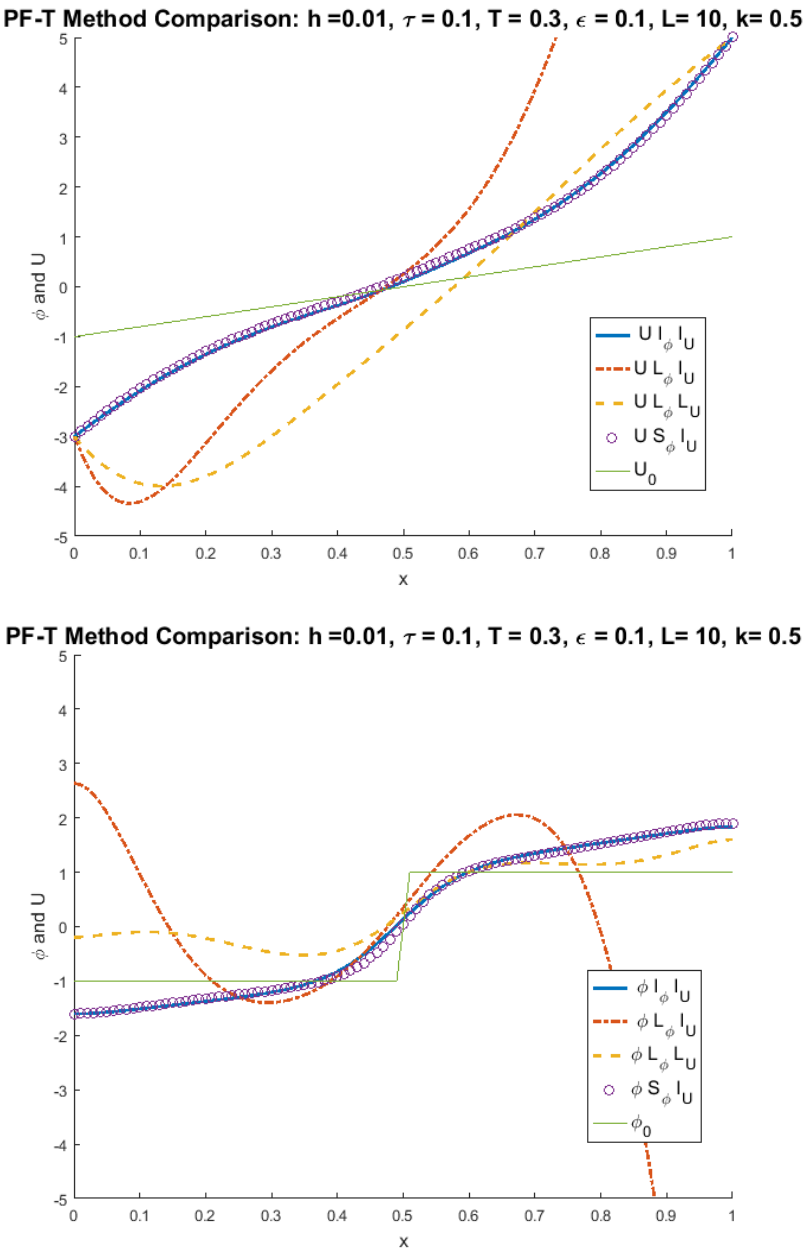

FIGURE 4.2: Image of numerical solutions for (4.12) and initial conditions, (4.18), using fully implicit, time lagging in ϕ, time lagging in both ϕ and U, and the splitting methods. Here the fully implicit and splitting methods are stable, however both time lagging methods are not. The difference between these images and the on in 4.1 is the time step τ. Values of the phase field ϕ and the temperature u are plotted separately for clarity.

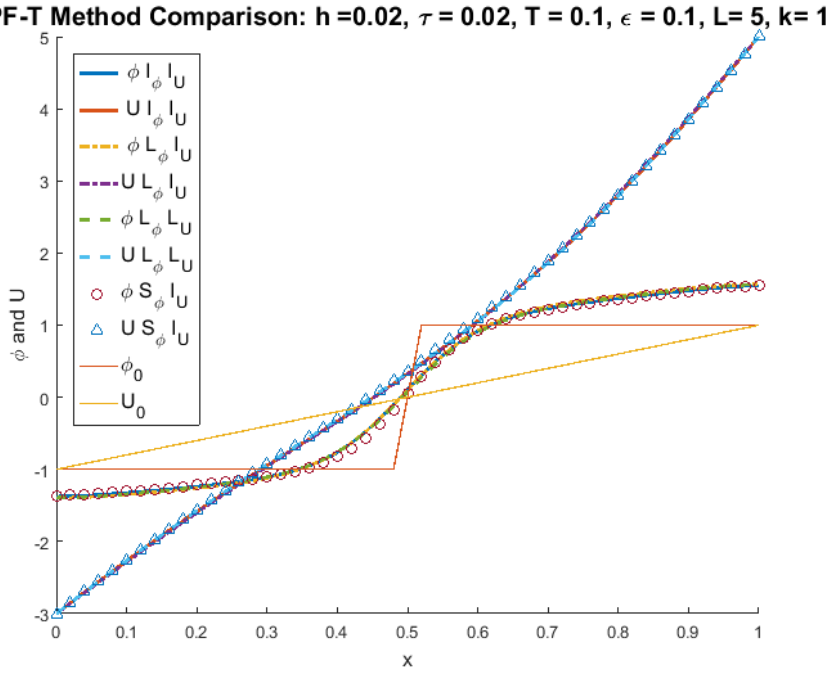

FIGURE 4.3: Image of numerical solutions and initial conditions for (4.12) using fully implicit, time lagging in ϕ, time lagging in both ϕ and U, and the splitting methods. The boundary conditions and initial conditions are specified in equations (4.15) and (4.16). This image is used as a base to compare impacts of values $\mathcal{L}$, ε, k, τ and h on the numerical solution for each method. Here all methods align except for the splitting method deviating slightly from the others for the phase field model, ϕ, for $3.5 < x < 6.5$.

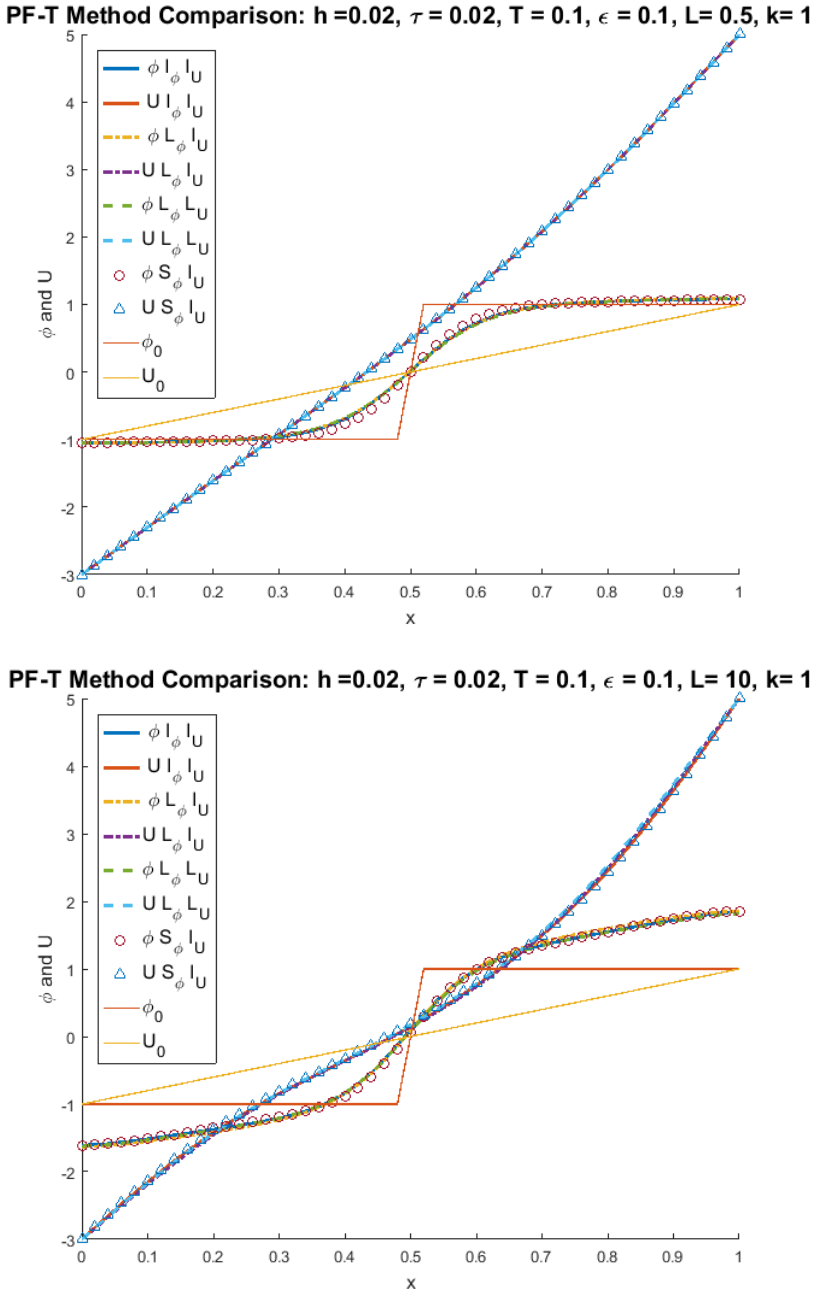

FIGURE 4.4: Image of numerical solutions for (4.12) with initial condition as in equations (4.15) and (4.16) and with the boundary conditions in (4.18) for two different values of $\mathcal{L}$. The value of the latent heat $\mathcal{L}$ impacts the slopes of both the phase field, ϕ, and the temperature, U. In this example, bigger values of latent heat drive the phase field below its stability value of -1 and above its stability value of 1 and the time lagging methods became unstable at $\mathcal{L} = 12$. The figure also shows that system, (4.12), is less coupled with lower values of $\mathcal{L}$.

Next, the impacts of the diffusion coefficient k will be discussed. Here k is the same in both phases, future models will implement different values of k for different phases. Figure 4.5 shows the impacts of changing the the diffusion coefficient k. The diffusion coefficient quantifies how fast energy propagates through a material. In this case, the energy propagating is measured in temperature. The higher value of k diffused faster as expected, which can be seen in this figure by the fact that the temperature in the plot where $k = 10$ is at the steady state linear distribution expected for the given boundary conditions while for $k = 0.1$, the temperature is far from steady state and the latent heat plays a bigger roll. The state $k = 1$ in-between $k = 0.5$ and $k = 10$ can be seen in Figure 4.3. The images in Figure 4.5 required 4 Newton iterations for the implicit method to converge.

The value ε in equation (4.12) regulates the width of the interface. These effects can be seen in Figure 4.6. Note that both large and small values of ε can create instabilities. To avoid these instabilities,a method that does not smooth the interface is likely a better choice for small values of ε, small being relative to the problem, for the example used in Figure 4.6, it is best to use values of ε above 0.05. The none of the method used to form Figure 4.6 converge if $\varepsilon > 10$, though this puts the width of the phase transition wider than the whole domain and thus is likely not to capture the dynamics that the model is built for.

The stability of the system, (4.12), is sensitive to the time step τ. Especially the time lagging methods in Figure 4.7, become unstable with large values of τ. By "unstable" we mean that their solutions oscillate between different values from one time step to another. We also see that the values of φ do not seem to conform well to the sign of u.

The exact values of τ that allow these time lagging methods become unstable depend on all the other parameters, especially $\mathcal{L}$ and ε. The smaller τ is, the more the solutions

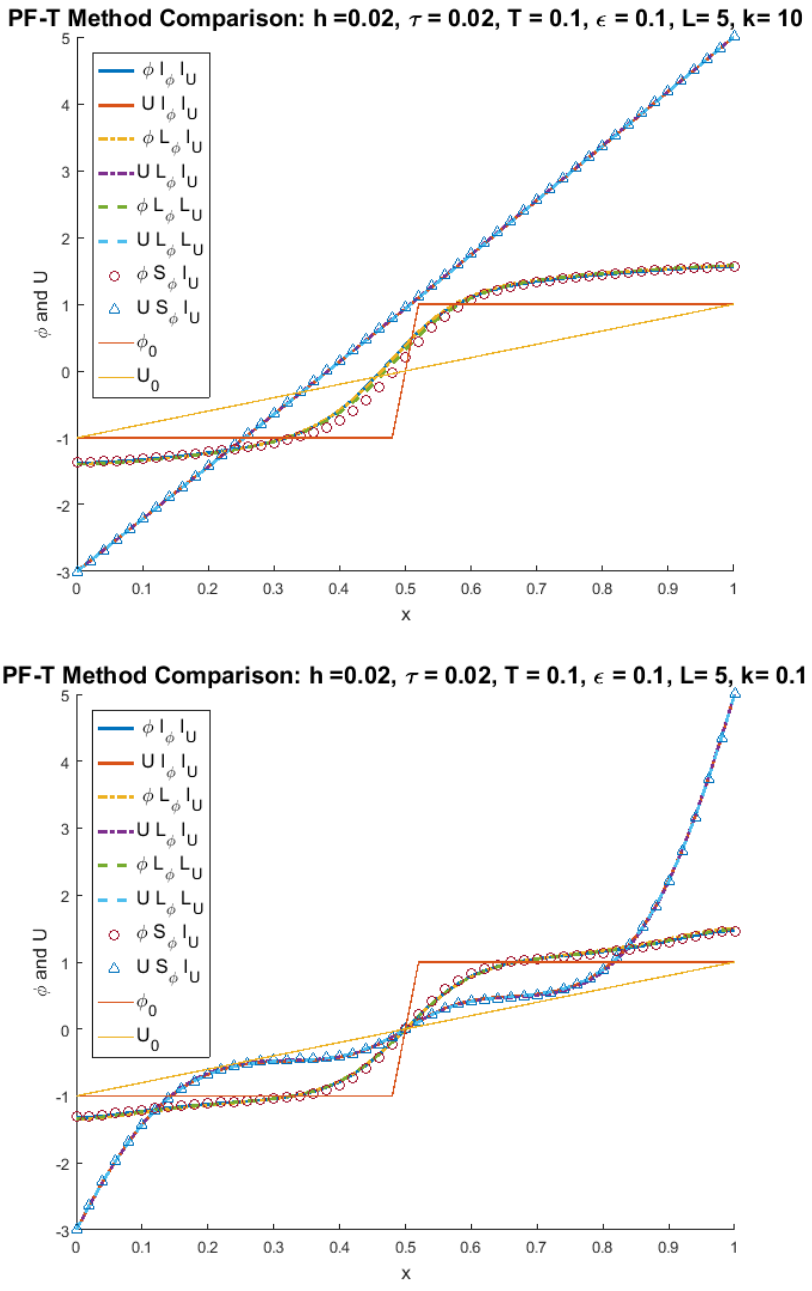

FIGURE 4.5: Image of numerical solutions for (4.12) with initial condition as in equations (4.15) and (4.16) and with the boundary conditions in (4.18) for two different values of the diffusion coefficient, k, for the temperature. The large the diffusion coefficient is, the fast the temperature can change, meaning a system with all else kept the same will reach steady state faster with a higher value of k. These effects are seen here in these images.

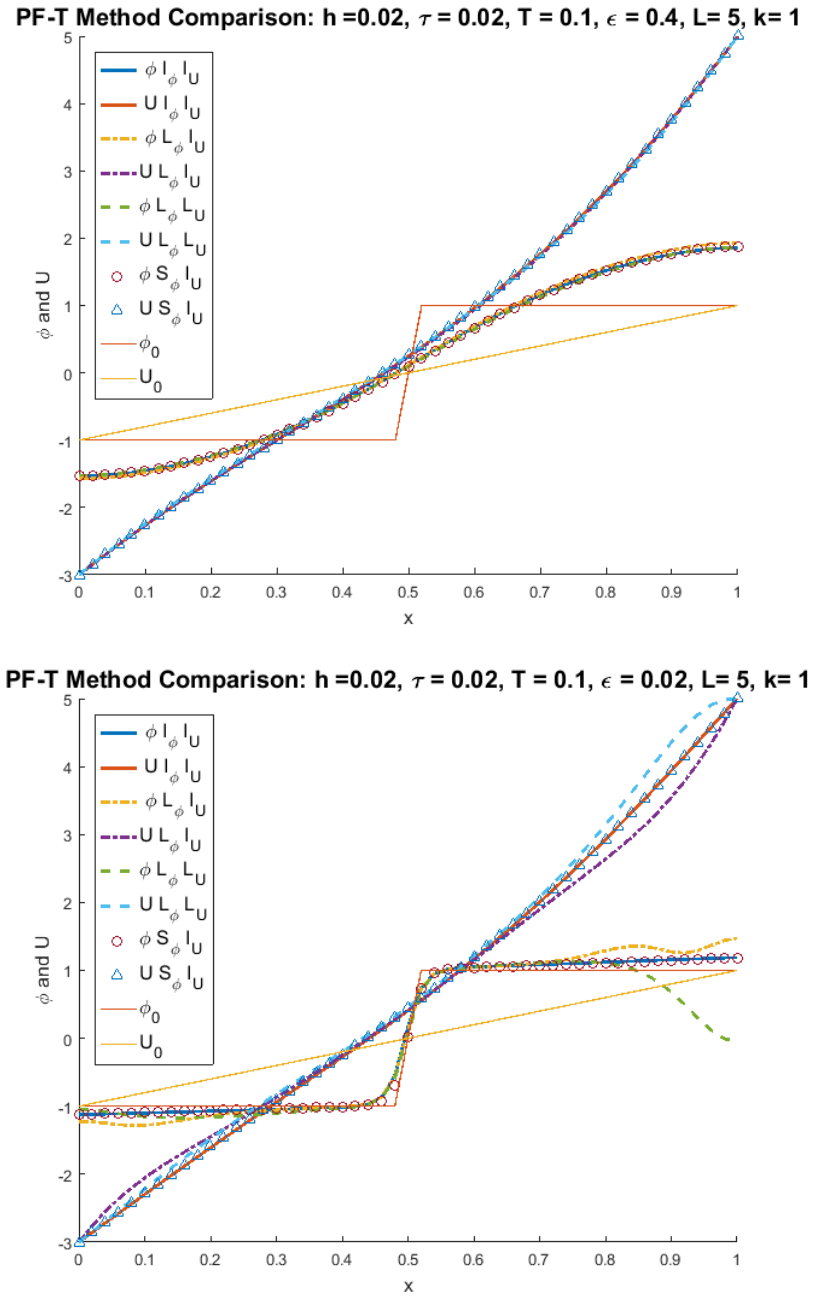

FIGURE 4.6: Image of numerical solutions for (4.12) with initial condition as in equations (4.15) and (4.16) and with the boundary conditions in (4.18) for two different values of the ε. The value ε is in the phase field model as a width regulator. As seen in these images and in Figure 4.3 the width of the phase transition appears to be about 2ε. The variable ε also impacts the stability, the instability for the time lagging methods can be seen for low values of ε as shown here when $\varepsilon = 0.02$. Note that none of the methods here converge for $\varepsilon = 0.01$.

created by the methods used in Figure 4.7 align, though the computational time goes up. For Figure 4.7, the number of Newton iterations the implicit method took was 5 for $\tau = 0.1$ and 3 for $\tau = 0.002$.

The spatial discretization parameter h mostly impacts the smoothness of the numerical solution and does not have much effect on the stability for the example used in Figure 4.8. The number of Newton iterations needed for the implicit method in Figure 4.8 was 4 for both images.

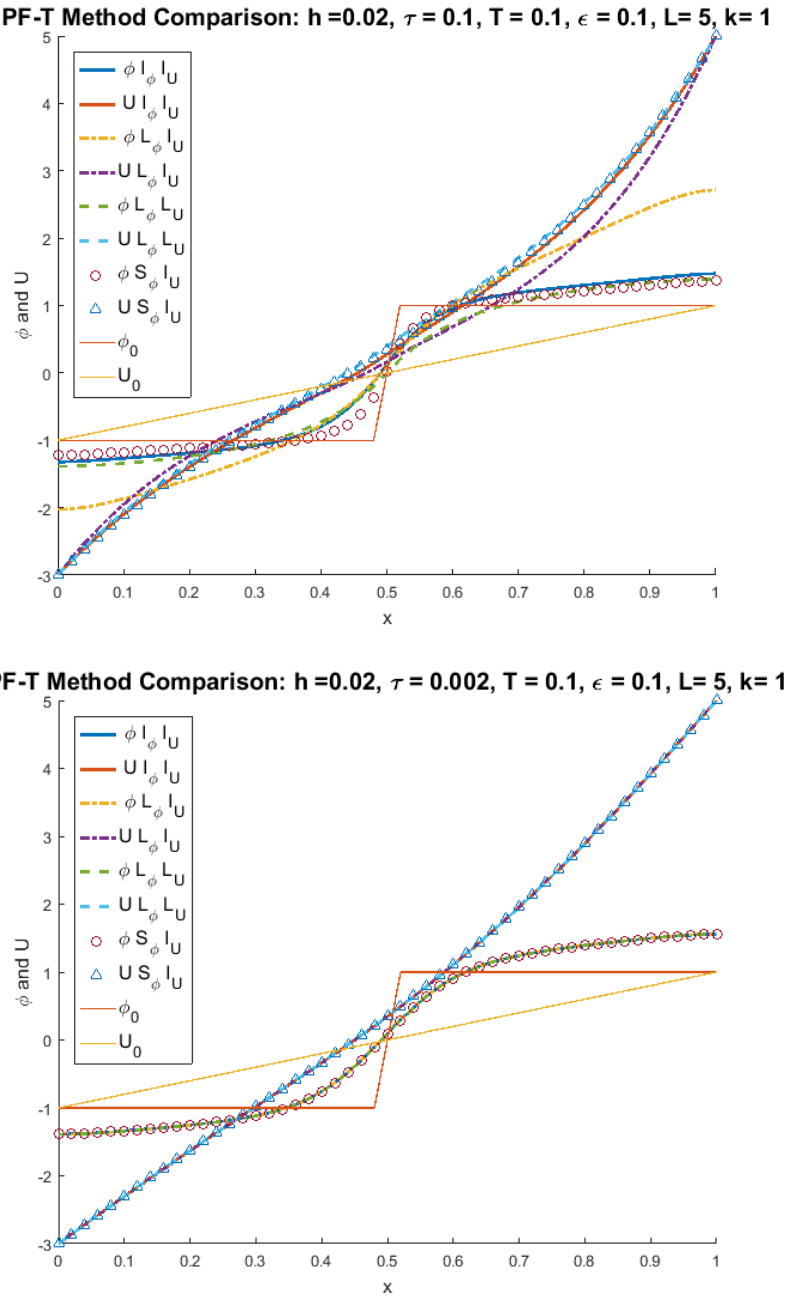

FIGURE 4.7: Image of numerical solutions for (4.12) with initial condition as in equations (4.15) and (4.16) and with the boundary conditions in (4.18) for two different values of τ. The stability of the system is heavily dependent on the time step τ. As seen and in 4.2 and here when $\tau = 0.1$, the time lagging methods are not very stable. Also note, the slight lag in the phase field for the splitting method, which diminishes as τ tends towards zero. Similar behavior is seen for the ODE in Section 2.1..

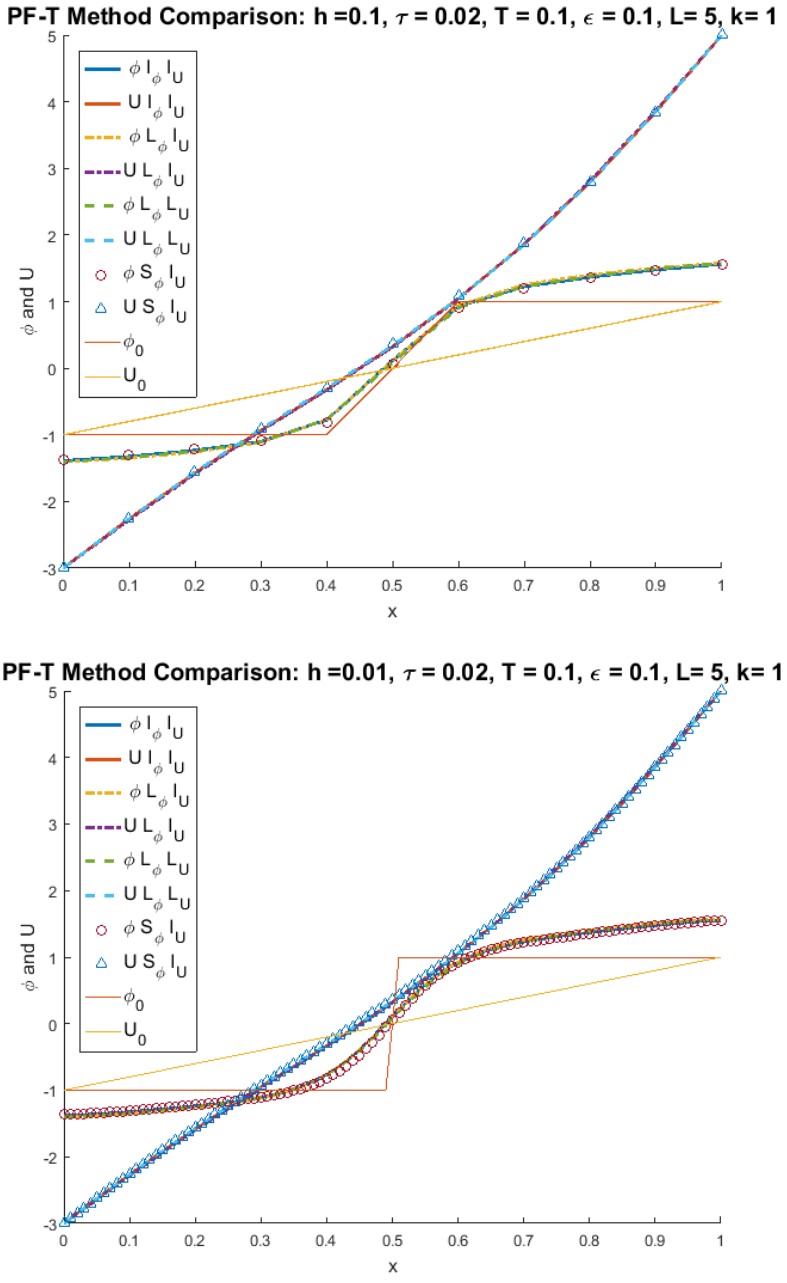

FIGURE 4.8: Image of numerical solutions for (4.12) with initial condition as in equations (4.15) and (4.16) and with the boundary conditions in (4.18) for two different values of the spatial step h. The value of h does not appear to impact the stability much but heavily impacts the quality of the approximations.

5. CONCLUSION

In this work we explored the use of time-splitting for a coupled energy equation and phase field model.

After developing a background in the splitting method for ordinary differential equations, numerical methods for parabolic partial differential equations, and Newton's method, numerical methods for nonlinear partial differential equations were discussed. The techniques used for the nonlinear PDEs were later used for the phase field model. After numerical methods for nonlinear PDEs were discussed, the Stefan problem was introduced and the weak form was formulated. Next, the phase field model was introduced to spread the interface of the phases over a region instead of the sharp interface used in the Stefan problem. The phase field model and the Stefan problem were then coupled to form a temperature model for a multiphase homogeneous medium in one dimension. This coupled system has been studied before, but of great interest was the application of the convexity splitting method to this system.

Based on the examples considered, the convexity splitting method for the coupled phase field and temperature model appears to be as stable as the implicit method with less computational effort. However, the splitting method lagged behind the implicit and time lagging methods, for some values of $\mathcal{L}$, ε, k, τ or h but reached the same steady state. These effects were seen when using the splitting method for the ODE as well.

Analysis for stability based on the time step would be beneficial and would give a better idea of when to use a convexity splitting method over a time lagging method, for a computationally fast method.

BIBLIOGRAPHY

[1] Samuel M Allen and John W Cahn. "A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening". In: *Acta metallurgica.* 27.6 (1979), pp. 1085–1095. ISSN: 0001-6160.

[2] Xinfu Chen. "Convergence of numerical solutions to the Allen-Cahn equation". In: *Applicable analysis* 69.1-2 (1998), pp. 47–56.

[3] Andrew Christlieb et al. "High accuracy solutions to energy gradient flows from material science models". In: *Journal of Computational Physics* 257 (2014), pp. 193 –215. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2013.09.049. URL: http://www.sciencedirect.com/science/article/pii/S0021999113006633.

[4] David J. Eyre. "An Unconditionally Stable One-Step Scheme for Gradient Systems". Mar. 1997.

[5] Ronald B. Guenther and John W. Lee. *Partial Differential Equations of Mathematical Physics and Integral Equations.* New York: Dover Publications, Inc., 1988. ISBN: 973-0-486-68889-3.

[6] Randall LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems (Classics in Applied Mathematics Classics in Applied Mathemat).* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007. ISBN: 0898716292, 9780898716290.

[7] Jie Shen and Xiaofeng Yang. "Numerical approximations of allen-cahn and cahn-hilliard equations". In: *Discrete Contin. Dyn. Syst* 28.4 (2010), pp. 1669–1691.

[8] R. E. Showalter. *Monotone Operations in Banach Space and Nonlinear Partial Differential Equations.* Rhode Island: American Mathematical Society, 1997. ISBN: 0-8218-0500-2.

[9] A. M. Stuart and A. R. Humphries. "Model Problems in Numerical Stability Theory for Initial Value Problems". In: *SIAM Review* 36.2 (1994), pp. 226–257. ISSN: 00361445. URL: http://www.jstor.org/stable/2132462.

[10] A. Visintin. *Models of Phase Transitions.* Birkhäuser Boston: Birkhäuser, 1996. ISBN: 978-1-4612-8641-7.