A Fast and Efficient Solver for Viscous-Plastic Sea Ice Dynamics

by

Clint Seinen
B.A.Sc., University of British Columbia, 2013

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

in the Department of Mathematics and Statistics

© Clint Seinen, 2017
University of Victoria

All rights reserved. This thesis may not be reproduced in whole or in part, by photocopying or other means, without the permission of the author.

A Fast and Efficient Solver for Viscous-Plastic Sea Ice Dynamics

by

Clint Seinen
B.A.Sc., University of British Columbia, 2013

Supervisory Committee

Dr. Boualem Khouider, Supervisor
(Department of Mathematics and Statistics)

Dr. Junling Ma, Departmental Member
(Department of Mathematics and Statistics)

ABSTRACT

Sea ice plays a key role in the global climate system. Indeed, through the albedo effect it reflects significant solar radiation away from the oceans, while it also plays a key role in the momentum and heat transfer between the atmosphere and ocean by acting as an insulating layer between the two. Furthermore, as more sea ice melts due to climate change, additional fresh water is released into the upper oceans, affecting the global circulation of the ocean as a whole. While there has been significant effort in recent decades, the ability to simulate sea ice has lagged behind other components of the climate system and most Earth System Models fail to capture the observed losses of Arctic sea ice, which is largely attributed to our inability to resolve sea ice dynamics. The most widely accepted model for sea ice dynamics is the Viscous-Plastic (VP) rheology, which leads to a very non-linear set of partial differential equations that are known to be intrinsically difficult to solve numerically. This work builds on recent advances in solving these equations with a Jacobian-Free Newton-Krylov (JFNK) solver. We present an improved JFNK solver, where a fully second order discretization is achieved via the *Crank Nicolson* scheme and consistency is improved via a novel approach to the rheology term. More importantly, we present a significant improvement to the Jacobian approximation used in the Newton iterations, and partially form the action of the matrix by expressing the linear and nearly linear terms in closed form and approximating the remaining highly non-linear term with a second order approximation of its Gateaux derivative. This is in contrast with the previous approach which used a first order approximation for the Gateaux derivative of the whole functional. Numerical tests on synthetic equations confirm the theoretical convergence rate and demonstrate the drastic improvements seen by using a second order approximation in the Gateaux derivative. To produce a fast and efficient solver for VP sea ice dynamics, the improved JFNK solver is then coupled with a non-oscillatory, central differencing scheme for transporting sea ice as well as a novel method for tracking the ice domain using a level set method. Two idealized test cases are then presented and simulation results discussed, demonstrating the solver's ability to efficiently produce Viscous-Plastic, physically motivated solutions.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

Dr. Boualem Khouider
for introducing me to this exciting topic and patiently providing valuable advice and guidance through this endeavor, all while aiding me financially through personal research grants and giving me the opportunity to pursue my interests.

Dr. Junling Ma
for being on my examination committee, taking the time to review this work and provide appreciated comments.

Dr. Ben Nadler
for being the external examiner, providing insight from a fresh point of view.

The National Science and Engineering Research Council of Canada
for providing financial assistance through Dr. Khouider's research grant.

The University of Victoria and the Mathematics and Statistics Department
for giving me this opportunity, providing me with an outstanding learning environment, and aiding me financially through this ordeal.

My wonderful friends and family
for your outstanding support and patience through this challenging yet fruitful time of my life.

If I have seen further than others, it is by standing upon the shoulders of giants.
Sir Isaac Newton

Chapter 1

Introduction

1.1 The Problem

Since the advent of satellite observations, from 1979 onwards, we have seen a drastic reduction in the Arctic sea ice extent at the end of the melt season (September), with an observed rate of decline of $> 11\%$. To make matters worse, there is evidence to believe that this rate is increasing [15]. We are also seeing considerably less multi-year ice and additional observations indicate regionally varying reductions in ice thickness [15, 29]. While there has been significant effort to understand sea ice evolution in recent years, this observed loss is only partially captured by Earth System Models (ESMs) [14]. Most simulations point to a continued reduction in ice mass and extent throughout the 21st century, but there is significant inter-model scatter [15]. It is expected that we may see a seasonally ice-free Arctic within this century, but the timing remains a point of contention [32]. Unfortunately, this uncertainty in sea ice projections directly translates to added uncertainty in our ability to predict the global climate [15]. Indeed, sea ice has a major impact on the heat and momentum exchange between the atmosphere and the ocean. Through the albedo affect, it influences how much of the sun's heat is absorbed by the ocean, and it acts as an insulating layer, partially decoupling the thermodynamic exchange between the ocean and atmosphere. Additionally, as more sea ice melts, it affects the freshwater content of the upper ocean which in turn influences the global overturning circulation of the ocean [14]. Improving our understanding of the evolution of sea ice cover has thus been touted as a "grand challenge of climate science" [15].

To accurately simulate sea ice thickness and the accompanying concentration

fields, it is crucial that we capture sea ice drift, which hinges on sea ice dynamics and the associated rheology formulation, linking the applied stresses to resulting deformations [14, 18]. At the typical grid scales of ESMs (10 to 100 km), sea ice has historically been treated as a continuum. One of the earliest approaches for the constitutive laws adapted to this continuum was a viscous law, which was employed with some success by both US and Russian researchers [8]. Despite this success, it was noted that this approach was inconsistent with our physical understanding of the local behaviour of ice [8]. At smaller scales, sea ice cover is formed by individual ice floes that move around and collide with each other under external forcing from the surrounding environment. At critical tensile stresses, the ice fails and separates, resulting in swaths of open water, or leads, with little resistance to further divergence [8, 5]. For critical compressive stresses, the ice breaks and piles up, forming sinuous ridges above and below the ice, known as pressure ridges. For both pressure ridges and leads, the deformation is clearly irreversible [5], which, combined with the apparent need for critical stresses, motivated other groups to consider plastic laws [3]. Equipped with these ideas and motivated by early successes of the viscous approach, Hibler [8] derived the widely used Viscous-Plastic (VP) constitutive law, arguing that at large enough temporal and spatial scales (i.e. those used in ESMs), collections of perfectly plastic floes lead to an averaged viscous behaviour [5]. Using this rheology formulation, Hibler went on to create the VP sea ice model [9], using an elliptical yield curve and a normal flow rule, which became widely accepted as the classical sea ice dynamics model [7].

Despite its popularity and wide spread use in sea ice studies [9, 35, 7, 11, 26, 21, 19, 18, 17], Hibler's constitutive law leads to a highly non-linear and highly stiff set of partial differential equations that are known to be very difficult to solve numerically [35, 11, 19, 18]. The difficulties arise because, by design, the elliptical yield curve allows sea ice to resist large stresses in compression and shear, yet have very limited tensile strength [18]. To apply an explicit time stepping scheme to the VP equations, stability analyses have shown that the required time step for typical ESM grid resolutions of 100 or 10 km, are on the order of a second and 100th of a second, respectively [18]. As a result of these strict requirements, early modelers typically adopted an implicit approach. The original VP model by Hibler [9] was based on a modified Euler time step, where the solution was first advanced to the middle of the time step by solving the linearized system implicitly via a successive overrelaxation (SOR) solver. The non-linear coefficients were then updated and the linearized sys-

tem solved again to advance the solution to the next time step [19]. This approach can be thought of as a multi time-level approach, where each intermediate time level is a pseudo-time step. In later work by Zhang and Hibler [35], it was demonstrated that a single pseudo-time step failed to result in a fully VP solution, i.e. all stress states lie either inside or on the yield curve. Additionally, work by Hunke and Zhang [12] showed that, despite the observed near instantaneous reaction, this approach resulted in a very slow transient response to forcing unless small time steps, relative to the time-scale of the changing forcing, were used. More recently, Lemieux and Tremblay [19] have shown that these problems could be attributed to poor non-linear convergence and that they could be remedied through many pseudo-time steps, or Outer Loop (OL) iterations. Nevertheless it was shown that even with many OL iterations, these solvers converge very slowly to the non-linear solution. These issues, in effect, greatly reduced the effectiveness of the standard implicit solver.

As one method to deal with these numerical difficulties, Hunke and Dukowicz [11] added an artificial elastic term to the VP constitutive law, creating the Elastic-Viscous-Plastic (EVP) model, which was further improved upon in [10]. This addition of elasticity greatly reduced the stability requirements for the VP equations, allowing for a fully explicit numerical scheme, naturally suited for parallel architectures [11, 10]. In addition to its computational benefits, this method was significantly better at capturing the transient response to forcing [10]. The basic idea of this method is to approximate the VP solution by damping the resulting elastic waves via subcycling [18], which can thought of as intermediate time steps. Provided enough subcycles were employed, the EVP solution should reduce to the VP solution, however Hunke [10] pointed out that in areas of rigid ice, the resulting waves don't damp as quickly as expected, particularly at fine resolutions, and these residual waves can have some effect on the resulting deformation rates. Later, Losch and Danilov [26] showed that, despite being a numerically motivated modification, the EVP scheme can also result in notably different solutions from the fully VP approach, caused by the smaller EVP viscosities.

To maintain the original VP equations, Lemieux et al [20] decided to tackle the non-linearities head on, and proposed a new, fully implicit scheme for the VP equations using a Jacobian-Free Newton-Krylov (JFNK) method. This method is a Newton scheme based on an approximated Jacobian matrix built from a first order, finite difference approach to a Gateaux derivative. To solve for the Newton correction at each non-linear iteration, a Krylov subspace method is employed to solve the re-

sulting linear system. It should also be noted that due to the formulation of the original VP equations [9], the resulting viscosities could become arbitrarily high and so they were typically capped by a large, but fixed, upper limit. However, Lemieux and Tremblay [19] recently pointed out that this approach hindered convergence and that it could be improved with a smooth formulation. Combined with the JFNK scheme and continuously differentiable viscosities, Lemieux et al [18] compared the convergence properties of the new solver to those of the EVP method and showed that it consistently produced more accurate solutions with comparable computational efficiency, opening up new opportunities for the, once inferior, implicit solver for the VP equations.

1.2 Proposed Research

Despite the clear benefits of the JFNK scheme, there is still room for improvement. For instance, to the best of the author's knowledge, three out of four of the VP studies utilizing the JFNK solver [20, 18, 27] achieve only first order accuracy in time, via a backward Euler approach. Additionally, all four implementations [20, 18, 27, 17] utilize a simple first order approximation to the Gateaux derivative, which is less than optimal. It should also be noted that in these implementations and others, the method of discretization for the necessary equations requires boundary conditions be imposed on the viscosities, which are diagnostic variables, determined purely from the prognostic variables – specifically, [21, 19, 20, 18, 17] utilize Taylor expansions. As will be shown, the formulation of the viscosities is a very convoluted expression, and thus this ad-hoc approach can easily result in incompatibilities with the physically motivated boundary conditions of the prognostic variables.

In this thesis, we build on this work and improve the JFNK method using 1) a second order *Crank-Nicolson* scheme instead of the first order backward Euler approach, 2) an improved approximation to the Jacobian matrix, namely by formulating all the linear and nearly linear parts in closed form and using a second order approximation for the remaining highly non-linear part, and 3) a novel approach to the discretization of the rheology term, which circumvents the need to apply boundary conditions to any diagnostic variables.

In addition to producing sea ice velocities, to simulate sea ice dynamics, solvers also require a) a method for transporting the ice according to conservation laws and b) a method for dynamically setting the computational domain and setting boundary

conditions, depending on where ice is present. To accomplish this, we propose coupling our improved JFNK solver with the non-oscillatory, central differencing, finite volume method developed by Nessyahu and Tadmor [28], and implement a method commonly used to track flow interfaces in engineering applications and track the ice terminus via the level set method described in Sussman et al [33].

It is worth stating that this thesis is a *mathematical* study of the widely used VP sea ice model, focusing on the difficulties associated with the resulting equations. While there is debate over the validity of the VP model, a detailed study of the physics is beyond the scope of this work and thus we limit ourselves to using the commonly used equations and parameterizations associated with this model. This work aims to leverage known, advanced techniques in numerical methods to improve the representation of VP dynamics in today's ESMs.

1.3 Outline

This thesis is structured as follows. In Chapter 2, we introduce the Viscous-Plastic sea ice dynamics model, highlighting the resulting non-linearities. In Chapter 3 we present our improved Jacobian-Free Newton-Krylov method and the novel rheology treatment. To conclude Chapter 3, we perform extensive numerical tests on a synthetic set of equations, validating the solver and confirming fully second order convergence. Additionally, we assess the solver's sensitivity to ad-hoc parameters and demonstrate the drastic improvement caused by our second order approach to the Jacobian approximation. Next, in Chapter 4 we introduce our method for transporting the ice according to its conservation laws and our novel approach for setting the computational domain via a level set method. In this chapter we also justify the need for and describe our methods for numerical smoothing and Newton Damping. Finally, in Chapter 5, we combine all the necessary components of our dynamics solver and present two idealized sea ice simulations, assessing the performance of our solver and discussing convergence to a Viscous-Plastic solution. We conclude with closing remarks and highlight the key take aways from this study in Chapter 6.

Chapter 2

The Viscous-Plastic Sea Ice Model

The evolution of large scale sea ice dynamics is primarily governed by the two-dimensional Sea Ice Momentum Equation (SIME) [3, 34],

$$\rho h \frac{D\boldsymbol{u}}{Dt} = -\rho h f(\boldsymbol{k} \times \boldsymbol{u}) + \tau_a - \tau_w + \nabla \cdot \boldsymbol{\sigma} - \rho h g \nabla H_d, \tag{2.1}$$

where, $\boldsymbol{u} = u\boldsymbol{i} + v\boldsymbol{j}$ is the horizontal sea ice velocity and $\boldsymbol{i}, \boldsymbol{j}$, and $\boldsymbol{k}$ are the Cartesian unit vectors. We denote by x and y, the coordinates in the east-west or zonal direction, $\mathbf{i}$, and in the north-south or meridional direction, $\mathbf{j}$, respectively, while t is time. ρ and h represent the sea ice density and thickness, respectively, while f is the Coriolis parameter. τ_a and τ_w are the wind stress and water drag forcing terms. $\boldsymbol{\sigma}$ is the internal sea ice stress tensor and $\nabla \cdot \boldsymbol{\sigma}$ is known as the rheology term. H_d is the sea surface height and g is the acceleration due to gravity. In (2.1), $\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y}$ is the material derivative with respect to the ice motion. Because the pack ice velocities are relatively small, the contribution of the non-linear advection in the SIME is typically neglected [35, 21, 20, 17, 26, 10, 8].

As done in [34], the sea surface tilt, ∇H_d, is set by the geostrophic balance

$$-f\boldsymbol{k} \times \boldsymbol{u}_w^g = g\nabla H_d, \tag{2.2}$$

where $\boldsymbol{u}_w^g$ is the geostrophic ocean current. Also following [18], the air and water drag terms are expressed via an empirical quadratic law with constant turning angles and constant drag coefficients,

$$\tau_a = \rho_a C_{da} |\boldsymbol{u}_a^g| (\boldsymbol{u}_a^g \cos\theta_a + (\boldsymbol{k} \times \boldsymbol{u}_a^g) \sin\theta_a), \tag{2.3}$$

$$\tau_w = C_w((\boldsymbol{u} - \boldsymbol{u}_w^g)\cos\theta_w + (\boldsymbol{k} \times (\boldsymbol{u} - \boldsymbol{u}_w^g))\sin\theta_w), \tag{2.4}$$

$$C_w = \rho_w C_{dw} |\boldsymbol{u} - \boldsymbol{u}_w^g|. \tag{2.5}$$

where $\boldsymbol{u}_a^g$ is the geostrophic wind and ρ_a and ρ_w represent the air and water densities, respectively, while C_{da} and C_{dw} are the air and water drag coefficients. Note that the sea ice velocity is neglected in the formulation of τ_a as $|\boldsymbol{u}_a^g| >> |\boldsymbol{u}|$ [18].

Now, although the expression in (2.4)-(2.5) is non-linear in $\boldsymbol{u}$, the key non-linearity of the SIME comes from the formulation of the rheology term, $\nabla \cdot \boldsymbol{\sigma}$. We follow Hibler [9], and write the VP constitutive law, relating internal stresses and strain rates, as

$$\sigma_{ij} = 2\eta\dot{\epsilon}_{ij} + [\zeta - \eta]\dot{\epsilon}_{kk}\delta_{ij} - P\delta_{ij}/2, \quad i, j = 1, 2, \tag{2.6}$$

where σ_{ij} is the i, j component of the internal stress tensor, ζ and η are the bulk and shear viscosities, respectively, and δ_{ij} is the Kronecker delta. The strain rates, $\dot{\epsilon}_{ij}$, are defined as

$$\dot{\epsilon}_{11} = \frac{\partial u}{\partial x}, \quad \dot{\epsilon}_{22} = \frac{\partial v}{\partial y}, \quad \dot{\epsilon}_{12} = \dot{\epsilon}_{21} = \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \quad \dot{\epsilon}_{kk} = \dot{\epsilon}_{11} + \dot{\epsilon}_{22}. \tag{2.7}$$

The parameterization of the ice strength or pressure, P, is formulated according to [9]

$$P = P^* h \, \exp\left[-C(1 - A)\right], \tag{2.8}$$

where $P^* = 27.5 \times 10^3$ N m^{-2} is the ice strength parameter and $C = 20$, the ice concentration parameter, is an empirical constant that characterizes the dependence of the compressive strength of ice on the area fraction or coverage, A [18].

In Hibler's original formulation [9], the viscosities are formulated using an elliptical yield curve with a normal flow rule, i.e.

$$\zeta = \frac{P}{2\Delta}, \quad \eta = \zeta e^{-2}, \tag{2.9}$$

where $e = 2$ is the principal axis ratio of the elliptical yield curve [9] and

$$\Delta = \left[(\dot{\epsilon}_{11}^2 + \dot{\epsilon}_{22}^2)(1 + e^{-2}) + 4e^{-2}\dot{\epsilon}_{12}^2 + 2\dot{\epsilon}_{11}\dot{\epsilon}_{22}(1 - e^{-2})\right]^{1/2}. \tag{2.10}$$

It is the combination of (2.6), (2.9), and (2.10) that leads to the very non-linear nature of the SIME.

It is easy to see that when the strain rates go to zero, ζ and η can become

arbitrarily large. To remedy this, during numerical simulations, Hibler [9] proposed the capping of the viscosities via

$$\zeta = \min\left(\frac{P}{2\Delta}, \zeta_{max}\right), \tag{2.11}$$

where $\zeta_{max} = kP$ and $k = 2.5 \times 10^8 s$. Unfortunately, this method leads to a rheology term that is not continuously differentiable. A smooth formulation is beneficial for numerical purposes and Lemieux and Tremblay [19] have demonstrated faster convergence can be achieved when one is used. Therefore, we follow [19] and redefine the bulk viscosity as

$$\zeta \equiv \zeta_{max}\tanh\left(\frac{P}{2\Delta\zeta_{max}}\right) = kP\tanh\left(\frac{1}{2\Delta k}\right). \tag{2.12}$$

In addition to the momentum equations in (2.1), sea ice dynamics models also involve the continuity equations,

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\boldsymbol{u}) = S_h, \tag{2.13}$$

and

$$\frac{\partial A}{\partial t} + \nabla \cdot (A\boldsymbol{u}) = S_A, \tag{2.14}$$

which describe how the sea ice thickness, h, and area fraction of thick ice, A, are advected by the sea-ice velocity field, $\boldsymbol{u}$, and altered according to the thermodynamic source/sink terms, S_h and S_A, due to melting and freezing, and the convergence and divergence of mass, which lead to the formation of leads and ridges. For this study, we focus on dynamics and thus set $S_h = S_A = 0$.

Chapter 3

The Improved JFNK Method for the Viscous Plastic Sea Ice Momentum Equations

Due to the non-linearities imposed by the rheology term in (2.1), the standard VP model for sea ice dynamics is intrinsically very difficult to solve. This has led the sea ice community to implement expensive implicit solvers [21] or modify the constitutive law, numerically motivated, to reduce stability requirements and utilize explicit solvers [11]. For our solver we build on the implicit, Newton solver introduced in [20] and improved upon in [18] and use a fully second order discretization to produce our JFNK method. Furthermore, we improve upon the first order approximation to the Jacobian matrix, forming the linear and nearly linear parts in closed form and adopt a second order approximation to the remaining highly non-linear term. In addition to the improved JFNK method, we implement a novel treatment of the rheology term, negating the need to enforce ad-hoc boundary conditions on the viscosity terms, improving consistency.

In this chapter we present our modification to the treatment of the rheology term and then the numerical algorithm is presented and discussed. The algorithm is then validated on a set of synthetic equations, confirming second order convergence. Finally we present the solver's sensitivity to various ad-hoc parameters.

3.1 The Rheology Treatment

To present our treatment of the rheology term, we begin by expanding it and writing the u and v components of (2.1) as,

$$-\rho h\frac{\partial u}{\partial t}+\rho h f v+\tau_{au}-\tau_{wu}+\frac{\partial \sigma_{11}}{\partial x}+\frac{\partial \sigma_{12}}{\partial y}-\rho h g\frac{\partial H_d}{\partial x}=0, \tag{3.1}$$

and

$$-\rho h\frac{\partial v}{\partial t}-\rho h f u+\tau_{av}-\tau_{wv}+\frac{\partial \sigma_{21}}{\partial x}+\frac{\partial \sigma_{22}}{\partial y}-\rho h g\frac{\partial H_d}{\partial y}=0, \tag{3.2}$$

where τ_{au} and τ_{wu} represent the u-components of the air and water drag, and likewise for the v-components. As already mentioned in the previous chapter, consistent with existing sea-ice models, the advection non-linearities are neglected in (3.1) and (3.2), as these terms are typically small.

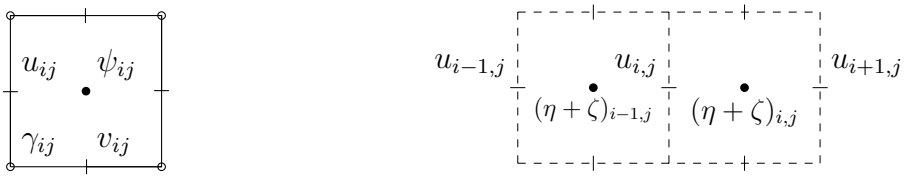
The rheology terms in (3.1) and (3.2) are given by

$$\begin{aligned}\frac{\partial \sigma_{11}}{\partial x}+\frac{\partial \sigma_{12}}{\partial y}=&\frac{\partial}{\partial x}\left[(\eta+\zeta)\frac{\partial u}{\partial x}\right]+\frac{\partial}{\partial x}\left[(\zeta-\eta)\frac{\partial v}{\partial y}\right]\\&-\frac{1}{2}\frac{\partial P}{\partial x}+\frac{\partial}{\partial y}\left[\eta\frac{\partial u}{\partial y}\right]+\frac{\partial}{\partial y}\left[\eta\frac{\partial v}{\partial x}\right],\end{aligned} \tag{3.3}$$

and

$$\begin{aligned}\frac{\partial \sigma_{21}}{\partial x}+\frac{\partial \sigma_{22}}{\partial y}=&\frac{\partial}{\partial x}\left[\eta\frac{\partial u}{\partial y}\right]+\frac{\partial}{\partial x}\left[\eta\frac{\partial v}{\partial x}\right]\\&+\frac{\partial}{\partial y}\left[(\eta+\zeta)\frac{\partial v}{\partial y}\right]+\frac{\partial}{\partial y}\left[(\zeta-\eta)\frac{\partial u}{\partial x}\right]-\frac{1}{2}\frac{\partial P}{\partial y}.\end{aligned} \tag{3.4}$$

Following [18], we discretize the SIME on the Arakawa C-grid, where the ice velocities, u and v, are anchored respectively at the centres of the vertical (along the y coordinate) and the horizontal (along the x coordinate) edges, as illustrated in Figure 3.1a. For simplicity in exposition, the collection of u and v points are referred to as the u-grid and v-grid, respectively. On the C-grid the centred points are referred to as tracer points, while the corner points are node points.



(a) The Arakawa C-grid

(b) Typical stencil of $\frac{\partial(\eta+\zeta)}{\partial x}$ at $u_{i,j}$

Figure 3.1: Grid Details

Typically, scalar values like the ice strength, P, and viscosities, η and ζ, are contained at the tracer points and interpolated to node points or the velocity grids as needed. This configuration allows for easy implementation of centred differences at the velocity grids. Using this approach, derivatives like $\frac{\partial}{\partial x}\left[(\eta+\zeta)\frac{\partial u}{\partial x}\right]$ in (3.3) - (3.4) are discretized according to the stencil in Figure 3.1b, amounting to.

$$\frac{\partial}{\partial x}\left[(\eta+\zeta)\frac{\partial u}{\partial x}\right] \approx \frac{(\eta+\zeta)_{i,j}[u_{i+1,j}-u_{i,j}]-(\eta+\zeta)_{i-1,j}[u_{i,j}-u_{i-1,j}]}{dx^2}, \tag{3.5}$$

at u point (i,j) (hereafter denoted by $u_{i,j}$), where dx represents the spatial resolution, which is assumed to be the same in both x and y-directions. The numerical derivative (3.5) requires boundary conditions for the ice viscosities, ζ and η, which are diagnostic variables, completely determined by the ice velocities, u, v and the ice strength, P. However, due to the convoluted expression of Δ, deriving boundary conditions that are compatible with those eventually imposed on $\boldsymbol{u}$ is not a trivial task; Lemieux et al. [18], for instance, use first order Taylor approximation to overcome this difficulty. To avoid this entirely, we propose expanding (3.3) and (3.4) before their discretization, amounting to

$$\frac{\partial}{\partial x}\left[(\eta+\zeta)\frac{\partial u}{\partial x}\right] = (\partial_x\eta+\partial_x\zeta)\frac{\partial u}{\partial x}+(\eta+\zeta)\frac{\partial^2 u}{\partial x^2} \tag{3.6}$$

$$\frac{\partial}{\partial x}\left[(\zeta-\eta)\frac{\partial v}{\partial y}\right] = (\partial_x\zeta-\partial_x\eta)\frac{\partial v}{\partial y}+(\zeta-\eta)\frac{\partial^2 v}{\partial x\partial y} \tag{3.7}$$

$$\frac{\partial}{\partial y}\left[\eta\frac{\partial u}{\partial y}\right] = \partial_y\eta\frac{\partial u}{\partial y}+\eta\frac{\partial^2 u}{\partial y^2} \tag{3.8}$$

$$\frac{\partial}{\partial y}\left[\eta\frac{\partial v}{\partial x}\right] = \partial_y\eta\frac{\partial v}{\partial x}+\eta\frac{\partial^2 v}{\partial x\partial y} \tag{3.9}$$

$$\frac{\partial}{\partial x}\left[\eta\frac{\partial u}{\partial y}\right] = \partial_x\eta\frac{\partial u}{\partial y}+\eta\frac{\partial^2 u}{\partial x\partial y} \tag{3.10}$$

$$\frac{\partial}{\partial x}\left[\eta\frac{\partial v}{\partial x}\right] = \partial_x\eta\frac{\partial v}{\partial x}+\eta\frac{\partial^2 v}{\partial x^2} \tag{3.11}$$

$$\frac{\partial}{\partial y}\left[(\eta+\zeta)\frac{\partial v}{\partial y}\right] = (\partial_y\eta+\partial_y\zeta)\frac{\partial v}{\partial y}+(\eta+\zeta)\frac{\partial^2 v}{\partial y^2} \tag{3.12}$$

$$\frac{\partial}{\partial y}\left[(\zeta-\eta)\frac{\partial u}{\partial x}\right] = (\partial_y\zeta-\partial_y\eta)\frac{\partial u}{\partial x}+(\zeta-\eta)\frac{\partial^2 u}{\partial x\partial y}. \tag{3.13}$$

Moreover, to get $\partial_*\zeta$ and $\partial_*\eta$, where ∂_* represents a generic spatial derivative, we differentiate (2.12) to produce

$$\partial_*\zeta = k\,\tanh\left(\frac{1}{2\Delta k}\right)\partial_* P - \frac{P}{2\Delta^2}\left[1-\tanh^2\left(\frac{1}{2\Delta k}\right)\right]\partial_*\Delta, \tag{3.14}$$

and for mathematical convenience, we use $\partial_*\Delta = \frac{\partial_*(\Delta^2)}{2\Delta}$ to get our final form

$$\partial_*\zeta = k\,\tanh\left(\frac{1}{2\Delta k}\right)\partial_* P - \frac{P}{4\Delta^3}\left[1-\tanh^2\left(\frac{1}{2\Delta k}\right)\right]\partial_*(\Delta^2), \tag{3.15}$$

where we calculate $\partial_*(\Delta^2)$ via

$$\begin{aligned}\partial_*(\Delta^2) = (1+e^{-2})\left[2\dot\epsilon_{11}\partial_*\dot\epsilon_{11}+2\dot\epsilon_{22}\partial_*\dot\epsilon_{22}\right]+8e^{-2}\epsilon_{12}\partial_*\epsilon_{12}\\ +2(1-e^{-2})\left[\dot\epsilon_{22}\partial_*\dot\epsilon_{11}+\dot\epsilon_{11}\partial_*\dot\epsilon_{22}\right].\end{aligned} \tag{3.16}$$

Now, using (3.15) and (3.16), we no longer require ad-hoc boundary conditions for the viscosities, such as Taylor expansions. Note that finite differences are still used to calculate the spatial derivatives of P, which is a diagnostic variable as well, but this doesn't cause a problem due to its rather simple expression, (2.8), in terms of h

and A.

3.2 The Numerical Approach: a Fully Second Order JFNK Solver

3.2.1 The Discretization

Temporally, letting dt be our temporal resolution, we solve (3.1) and (3.2) at times dt, $2dt$, $3dt$, ..., ndt, ..., T, identifying variables at time level n via superscripts. When progressing to time level $n+1$, we follow [18] and utilize the time splitting approach used in many sea ice models, [9, 13, 25, 21, 20], and hold h and A at time level n, thus for clarity we avoid using superscripts on these variables. Early implementations of this solver, [20, 18], use a backward Euler approach and achieve first order accuracy in time; we instead utilize the *Crank Nicholson* discretization and centre (3.1) and (3.2) at time level $n + 1/2$. The Crank Nicholson discretization achieves second order accuracy in time via centred differences on the temporal derivatives and the trapezoidal rule on the remaining terms, resulting in

$$\begin{aligned} -\frac{\rho h}{dt}u^{n+1} + \frac{\rho h f}{2}v^{n+1} + \frac{1}{2}\tau_{au}^{n+1} - \frac{1}{2}\tau_{wu}^{n+1} + \frac{1}{2}\frac{\partial \sigma_{11}}{\partial x}^{n+1} + \frac{1}{2}\frac{\partial \sigma_{12}}{\partial y}^{n+1} - \frac{\rho h g}{2}\frac{\partial H_d}{\partial x}^{n+1} \\ = -\frac{\rho h}{dt}u^{n} - \frac{\rho h f}{2}v^{n} - \frac{1}{2}\tau_{au}^{n} + \frac{1}{2}\tau_{wu}^{n} - \frac{1}{2}\frac{\partial \sigma_{11}}{\partial x}^{n} - \frac{1}{2}\frac{\partial \sigma_{12}}{\partial y}^{n} + \frac{\rho h g}{2}\frac{\partial H_d}{\partial x}^{n} \end{aligned} \quad (3.17)$$

and

$$\begin{aligned} -\frac{\rho h}{dt}v^{n+1} - \frac{\rho h f}{2}u^{n+1} + \frac{1}{2}\tau_{av}^{n+1} - \frac{1}{2}\tau_{wv}^{n+1} + \frac{1}{2}\frac{\partial \sigma_{21}}{\partial x}^{n+1} + \frac{1}{2}\frac{\partial \sigma_{22}}{\partial y}^{n+1} - \frac{\rho h g}{2}\frac{\partial H_d}{\partial y}^{n+1} \\ = -\frac{\rho h}{dt}v^{n} + \frac{\rho h f}{2}u^{n} - \frac{1}{2}\tau_{av}^{n} + \frac{1}{2}\tau_{wv}^{n} - \frac{1}{2}\frac{\partial \sigma_{21}}{\partial x}^{n} - \frac{1}{2}\frac{\partial \sigma_{22}}{\partial y}^{n} + \frac{\rho h g}{2}\frac{\partial H_d}{\partial y}^{n}. \end{aligned} \quad (3.18)$$

We then expand the rheology and water drag terms and group the right hand sides of the above equations into $\mathcal{CN}_u$ and $\mathcal{CN}_v$ and group the terms on the left hand sides

independent of $\boldsymbol{u}$ into r_u and r_v to get

$$-\frac{\rho h}{dt}u^{n+1}+\frac{\rho h f}{2}v^{n+1}-\frac{1}{2}C_w^{n+1}(u^{n+1}\cos\theta_w-v^{n+1}\sin\theta_w)+\frac{1}{2}\frac{\partial}{\partial x}\left[(\eta+\zeta)\frac{\partial u}{\partial x}\right]^{n+1}$$
$$+\frac{1}{2}\frac{\partial}{\partial x}\left[(\zeta-\eta)\frac{\partial v}{\partial y}\right]^{n+1}+\frac{1}{2}\frac{\partial}{\partial y}\left[\eta\frac{\partial u}{\partial y}\right]^{n+1}+\frac{1}{2}\frac{\partial}{\partial y}\left[\eta\frac{\partial v}{\partial x}\right]^{n+1}=\mathcal{CN}_u+r_u \quad (3.19)$$

and

$$-\frac{\rho h}{dt}v^{n+1}-\frac{\rho h f}{2}u^{n+1}-\frac{1}{2}C_w^{n+1}(v^{n+1}\cos\theta_w+u^{n+1}\sin\theta_w)+\frac{1}{2}\frac{\partial}{\partial x}\left[\eta\frac{\partial u}{\partial y}\right]^{n+1}+\frac{1}{2}\frac{\partial}{\partial x}\left[\eta\frac{\partial v}{\partial x}\right]^{n+1}$$
$$+\frac{1}{2}\frac{\partial}{\partial y}\left[(\eta+\zeta)\frac{\partial v}{\partial y}\right]^{n+1}+\frac{1}{2}\frac{\partial}{\partial y}\left[(\zeta-\eta)\frac{\partial u}{\partial x}\right]^{n+1}=\mathcal{CN}_v+r_v. \quad (3.20)$$

Keeping in mind our treatment of the rheology terms described in Section 3.1, it is at this point where we discretize the above equations spatially. For boundary conditions, we enforce Dirichlet conditions at land boundaries, $\boldsymbol{u}=0$, and Neumann conditions at open-water boundaries, $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}}=0$ ($\boldsymbol{n}$ is the outward normal to the water-ice lateral boundary). As previously mentioned, we discretize (3.19)-(3.20) on the Arakawa C-grid (Figure 3.1a), achieving second order accuracy through centred differences. For our specific discretization, only the area fraction (A), ice thickness (h) and strength (P) are held at the tracer points; due to our treatment of the rheology term, in addition to the non-linear drag coefficient (C_w), the viscosities and their necessary derivatives are now calculated directly at u and v points. Note that on the C-grid, u and v points are not collocated, and thus to populate v in the u equation of the SIME, and u for the v equation, we use a second order bi-linear interpolation from the four surrounding points.

3.2.2 The Solver

Implementing the discretization described above leads to a system of non-linear equations of the form

$$A(\boldsymbol{u}^{n+1})\boldsymbol{u}^{n+1}=\boldsymbol{b}(\boldsymbol{u}^{n+1}), \quad (3.21)$$

where $\boldsymbol{u}^{n+1}$ is the N dimensional solution vector, created by first stacking all u points in our computational domain and then all the v points, i.e.

$$\boldsymbol{u}^{n+1} = \begin{bmatrix} \boldsymbol{u}_u^{n+1} \\ \boldsymbol{u}_v^{n+1}. \end{bmatrix}$$

where $\boldsymbol{u}_u$ and $\boldsymbol{u}_v$ are two vectors containing the u-grid and v-grid velocities, respectively, at the associated Arakawa C-grid points that are within the ice field. Consequently, $A(\boldsymbol{u}^{n+1})$ is an $N \times N$ matrix containing all the necessary coefficients (both linear and non-linear) from the left hand side of (3.19) and (3.20), and $\boldsymbol{b}(\boldsymbol{u}^{n+1})$ is an N dimensional vector containing the remaining terms, i.e. $\mathcal{CN}_u$ and $\mathcal{CN}_v$ as well as r_u and r_v. We note that $\boldsymbol{b}$ is a function of $\boldsymbol{u}^{n+1}$ due to the presence of the non-linear coefficient C_w^{n+1}.

To solve (3.21), things are more involved than simply inverting the matrix, as both A and $\boldsymbol{b}$ are nonlinear functions of $\boldsymbol{u}^{n+1}$. We must incorporate a non-linear solver. For this, we follow Lemieux et al. ([20, 18]) and utilize a JFNK approach. JFNK methods amount to applying Newton's method to a multivariate system, where the Newton updates are obtained by solving the resulting linear system of equations via a Krylov-type subspace approximation method. A good survey of the JFNK methods is found in [16], but we provide a brief description here for the sake of completeness.

Keeping in mind that we are solving for the solution at time level $n+1$, we replace $\boldsymbol{u}^{n+1}$ with a non-linear iterate, $\boldsymbol{u}^k$, where the superscript now identifies the iteration number, and write the associated residual as

$$\boldsymbol{F}(\boldsymbol{u}^k) = A(\boldsymbol{u}^k)\boldsymbol{u}^k - \boldsymbol{b}(\boldsymbol{u}^k). \tag{3.22}$$

Defining $\delta\boldsymbol{u}^k = \boldsymbol{u}^{k+1} - \boldsymbol{u}^k$, we take a first order, multivariate Taylor Expansion about $\boldsymbol{F}(\boldsymbol{u}^k)$,

$$\boldsymbol{F}(\boldsymbol{u}^{k+1}) \approx \boldsymbol{F}(\boldsymbol{u}^k) + J(\boldsymbol{u}^k)\delta\boldsymbol{u}^k, \tag{3.23}$$

where $J(\boldsymbol{u}^k)$ is the Jacobian Matrix of $\boldsymbol{F}(\boldsymbol{u}^k)$, with respect to $\boldsymbol{u}^k$. Analogous to single variable Newton methods, the left hand side is set to zero and we obtain the following linear system

$$J(\boldsymbol{u}^k)\delta\boldsymbol{u}^k = -\boldsymbol{F}(\boldsymbol{u}^k), \tag{3.24}$$

which is solved via a Krylov Subspace method to produce the update vector. For our JFNK method we use Intel's MKL GMRES routine to solve the system in (3.24).

We note that this linear solver is *Matrix-Free*, meaning that the matrix $J(\boldsymbol{u}^k)$ isn't explicitly required, only its *action* on a vector is needed which translates to significant storage benefits. Unfortunately, due to the complexity of the SIME, even formulating the action of the Jacobian will be both computationally intensive and prone to errors. Thus, previous implementations [18] of this method approximate the Jacobian's action via a first order Gateaux derivative approximation making this method fully *Jacobian-Free*, i.e. they use [18]

$$J(\boldsymbol{u}^k)\boldsymbol{v} \approx \frac{\boldsymbol{F}(\boldsymbol{u}^k + \epsilon\boldsymbol{v}) - \boldsymbol{F}(\boldsymbol{u}^k)}{\epsilon}, \tag{3.25}$$

where ϵ is a small perturbation and the error is $O(\epsilon)$. Equation (3.25) is analogous to a forward difference approach. Consequently, using a centred difference approach,

$$J(\boldsymbol{u}^k)\boldsymbol{v} \approx \frac{\boldsymbol{F}(\boldsymbol{u}^k + \epsilon\boldsymbol{v}) - \boldsymbol{F}(\boldsymbol{u}^k - \epsilon\boldsymbol{v})}{2\epsilon}, \tag{3.26}$$

one can achieve a second order approximation, with $O(\epsilon^2)$ error, for the Gateaux derivative. Now although (3.26) is already a better approximation than (3.25), we take it a step further by expanding (3.26) according to (3.22), which after some re-arranging results in

$$\begin{aligned}\frac{\boldsymbol{F}(\boldsymbol{u}^k + \epsilon\boldsymbol{v}) - \boldsymbol{F}(\boldsymbol{u}^k - \epsilon\boldsymbol{v})}{2\epsilon} = \frac{A(\boldsymbol{u}^k + \epsilon\boldsymbol{v})\boldsymbol{u}^k - A(\boldsymbol{u}^k - \epsilon\boldsymbol{v})\boldsymbol{u}^k}{2\epsilon} + \\ \frac{A(\boldsymbol{u}^k + \epsilon\boldsymbol{v})\epsilon\boldsymbol{v} + A(\boldsymbol{u}^k - \epsilon\boldsymbol{v})\epsilon\boldsymbol{v}}{2\epsilon} - \frac{\boldsymbol{b}(\boldsymbol{u}^k + \epsilon\boldsymbol{v}) - \boldsymbol{b}(\boldsymbol{u}^k - \epsilon\boldsymbol{v})}{2\epsilon}.\end{aligned} \tag{3.27}$$

Then taking the limit in the last two terms as $\epsilon \to 0$, we get the final form of our Jacobian approximation,

$$J(\boldsymbol{u}^k)\boldsymbol{v} \approx \frac{A(\boldsymbol{u}^k + \epsilon\boldsymbol{v})\boldsymbol{u}^k - A(\boldsymbol{u}^k - \epsilon\boldsymbol{v})\boldsymbol{u}^k}{2\epsilon} + A(\boldsymbol{u}^k)\boldsymbol{v} - D\boldsymbol{b}(\boldsymbol{u}^k)\boldsymbol{v}, \tag{3.28}$$

where the third term represents the contribution from the Jacobian matrix of $\boldsymbol{b}(\boldsymbol{u}^k)$, which is easily calculated in closed form, the second term is the contribution from the linear portion of the Jacobian Matrix of $A(\boldsymbol{u}^k)\boldsymbol{u}^k$ and the first term is a second order approximation of the contribution from the non-linear part. Thus, this method amounts to computing the "simple" parts of the Jacobian in closed form and using a second order Gateaux derivative approximation for the remaining terms.

With the above methods in mind, the final step is to provide stopping criteria for both the non-linear and linear solvers. For the linear solver, we follow [18] and set

$$||J(\boldsymbol{u}^k)\delta\tilde{\boldsymbol{u}}^k + \boldsymbol{F}(\boldsymbol{u}^k)|| < \gamma(k)||\boldsymbol{F}(\boldsymbol{u}^k)||, \tag{3.29}$$

as our convergence criterion for the GMRES solver. Here $\delta\tilde{\boldsymbol{u}}^k$ is the approximate solution to (3.24), and $\gamma(k)$ is given by [18]

$$\gamma(k) = \begin{cases} \gamma_{ini} & \text{if} \quad ||\boldsymbol{F}(\boldsymbol{u}^k)|| \geq res_t \\ \min\left(\gamma_{ini}, ||\boldsymbol{F}(\boldsymbol{u}^k)||/||\boldsymbol{F}(\boldsymbol{u}^{k-1})||\right) & \text{if} \quad ||\boldsymbol{F}(\boldsymbol{u}^k)|| < res_t, \end{cases} \tag{3.30}$$

where $\gamma_{ini} = 0.99$. Note that unless otherwise noted, $||\cdot||$ represents the L_2 functional norm. This approach is referred to as an "inexact" Newton Method; in [20] it is noted that over solving of (3.24) in early iterations can not only be detrimental to computational costs, but it can also prevent the method to converge at all! In [20] and [18], they vary the value of res_t with spatial resolution, but for this study we keep it constant at 0.625; studying the effects of the value is left for future work.

To create a stopping criterion for the Newton iterations, when it is possible to bring $||\boldsymbol{F}(\boldsymbol{u})|| \to 0$, we propose using a tolerance proportional to the discretization error. Unfortunately, as we are working in a dimensionalized system, careful consideration must be taken to match the units of the discretized equation. Using the Coriolis parameter f to set a reference time scale, we set our nonlinear stopping criterion to

$$||\boldsymbol{F}(\boldsymbol{u}^k)|| < \rho h_0 f u_0 \epsilon_{tol}, \tag{3.31}$$

where u_0 is a typical sea ice velocity, chosen to be 0.1 m/s, $h_0 = 1$ m is a reference ice thickness, the ice density is set to $\rho = 900$ kg m^{-3}, and

$$\epsilon_{tol} = \gamma_{nl}\left(\frac{dx}{L_x}\right)^2, \tag{3.32}$$

where L_x is the length scale of our domain, chosen to be the domain extent in the x-direction, and γ_{nl} is a tolerance coefficient parameter whose default values is $\gamma_{nl} = 10$. As it will be demonstrated in Section 3.3.5, the numerical solution can be sensitive to the choice of the value of γ_{nl}, and its optimal value remains resolution dependent. A universal expression for ϵ_{tol} remains to be discovered.

It is worth noting that depending on the simulation, it is not always possible to

bring $||\boldsymbol{F}(\boldsymbol{u})|| \to 0$, so different stopping criteria are required; this will be discussed in more detail in Chapter 5. Nevertheless, for the simulations discussed in this chapter, bringing the residual to 0 is possible and thus (3.31) is used.

3.3 Validation with Synthetic Equations

To validate and assess our solver's performance, we elect to test its ability to reproduce a known solution. Unfortunately, the SIME doesn't have any known analytical solutions. To get around this, we follow [1] and construct our own. It is always possible to do so if appropriate forcing is added to the governing equations, the only requirement is that the chosen solution has enough structure to exercise higher-derivative calculations [30]. The solution need not be physical in any way. If we define

$$\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

to be our chosen solution, this procedure amounts to solving

$$-\rho h \frac{\partial \boldsymbol{u}}{\partial t} - \rho h f(\boldsymbol{k} \times \boldsymbol{u}) + \tau_a - \tau_w + \nabla \cdot \boldsymbol{\sigma} - \rho h g \nabla H_d = \mathcal{L}(\boldsymbol{w}), \tag{3.33}$$

where $\mathcal{L}(\boldsymbol{w})$ is a known forcing term at each grid point, defined by

$$\mathcal{L}(\boldsymbol{w}) = -\rho h \frac{\partial \boldsymbol{w}}{\partial t} - \rho h f(\boldsymbol{k} \times \boldsymbol{w}) + \tau_a - \tau_w(\boldsymbol{w}) + \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{w}) - \rho h g \nabla H_d, \tag{3.34}$$

which can be calculated exactly for any known $\boldsymbol{w}$.

For our known solution, we choose the two dimensional, moving sine wave,

$$\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{10}\sin\left[(\frac{4x}{L_x} - 2)^2 + (\frac{4y}{L_y} - 2)^2 + ct\right] \\ \frac{1}{10}\cos\left[(\frac{4x}{L_x} - 2)^2 + (\frac{4y}{L_y} - 2)^2 + ct\right] \end{bmatrix}, \tag{3.35}$$

where L_x and L_y are our domain's extent in the x and y-directions and $c = 5 \times 10^{-6}$ s^{-1} is the phase. In addition to this solution having significant structure it also has values of similar order to that expected for sea ice velocities. Figure 3.2 shows this solution with $t = 0$ s, over the entire domain with $L_x = L_y = 2000$ km. Note that in being consistent with the Arakawa C-grid, we elect to produce w_1 on the u-grid and w_2 on the v-grid.

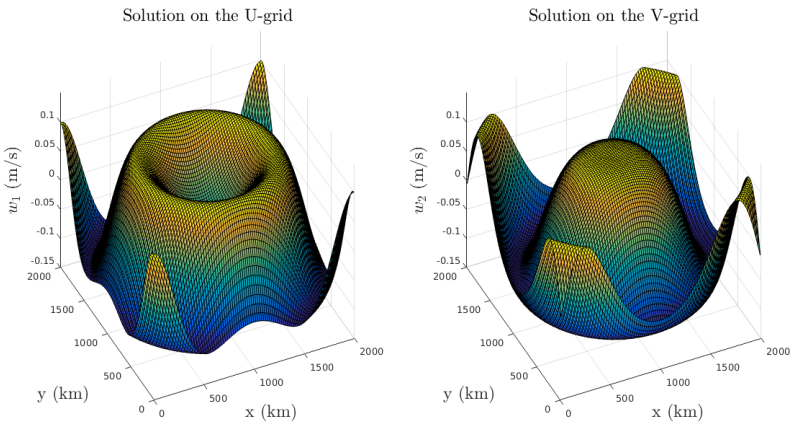


Figure 3.2: Validation solution structure: w_1 on the u-grid (left), w_2 on the v-grid (right)

For the actual experiment, we limit the computational domain to two separate ice patches in the southwest and northeast corners of the square ocean basin, surrounded by land, as shown in a toy example in Figure 3.3. It has to be noted that special care must be taken when building the non-linear system in (3.21) so computations are only limited to areas with ice; the justification for this and the method is discussed in Chapter 4.

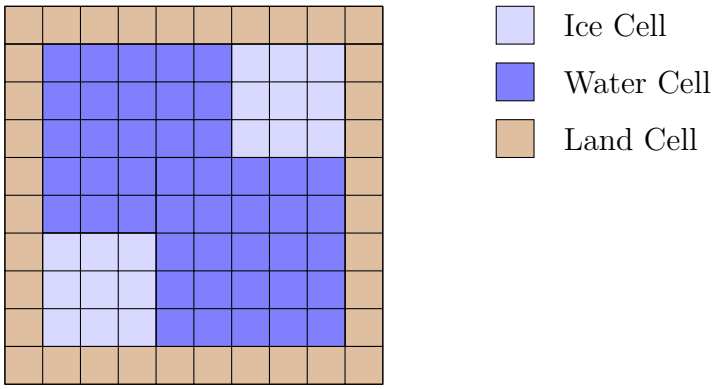


Figure 3.3: Toy example of validation simulation domain

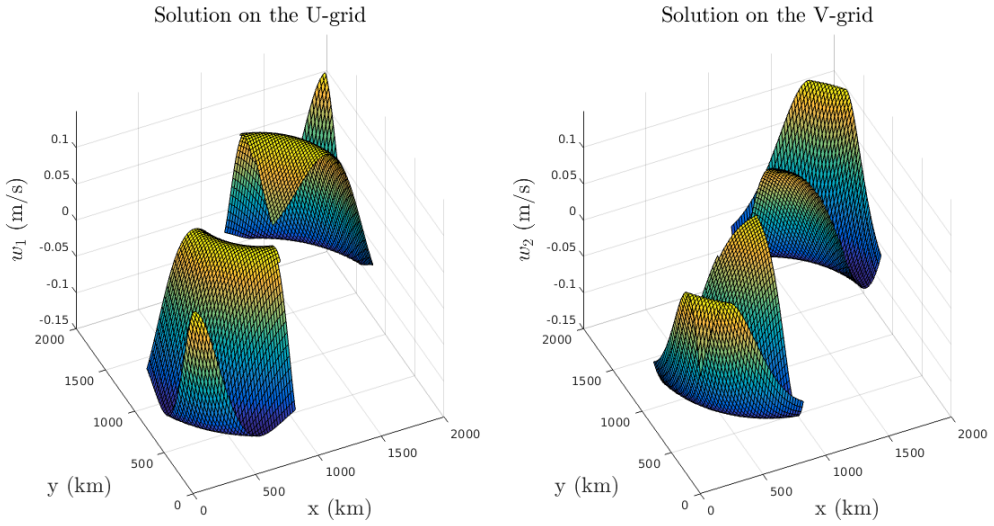


Figure 3.4: Validation solution ($t = 0$) limited to the ice-only computational domain

The reasoning behind this domain choice is two fold. First, it allows us to test all boundary types, i.e. open water boundaries to the north, south, east, and west, and likewise with land boundaries. Second, note the local minima and maxima of the test solution near the centre of the domain in Figure 3.2; the presence of these extrema will lead to a capping of the viscosities which may cause large errors and plague our convergence estimates. Thus, for the sake of convergence tests, we avoid this area.

It should be noted that in order to properly produce (3.35), slight modifications must be made to the boundary conditions. We still use Dirichlet conditions at land boundaries and Neumann conditions at open water boundaries, but they are no longer homogeneous, i.e. we use

$$\boldsymbol{u} = \beta(\boldsymbol{x}, t) \quad \text{at land boundaries,} \tag{3.36}$$

and

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} = \alpha(\boldsymbol{x}, t) \quad \text{at open water boundaries,} \tag{3.37}$$

where β and α are the known solution and its derivatives, determined from (3.35). Once a solver is given the ability to use non-homogeneous conditions, homogeneous conditions can be easily applied as they are only a special case of (3.36) and (3.37) with $\beta(\boldsymbol{x}, t) = \alpha(\boldsymbol{x}, t) = 0$.

3.3.1 The Validation Simulation

In current ESMs, solvers are typically ran on time scales on the order of months to years. These scales are not practical for validation tests, thus we elect to test the validation problem over a week long simulation. This temporal scale allows for effective numerical testing yet provides preliminary insight into longer time scales. For all simulations, we set our physical parameters as laid out in [18]. Table 3.1 contains the chosen numerical parameters for this validation simulation.

Table 3.1: Numerical Parameters for Validation Simulation

Symbol	Definition	Value
dx	Spatial Resolution	20 km
dt	Temporal Resolution	10 min
L_x	X-Extent	2000 km
L_y	Y-Extent	2000 km
T	Final Time	7 days
ϵ	Jacobian Perturbation	10^{-6}
res_t	Residual Transition	0.625
γ_{nl}	Non-Linear Tolerance Coef.	10
k_{max}	Maximum Number of Non-Linear Iterations	200

Although it should have little to no affect on the validation simulation, a forcing field must be specified; we use the forcing field described in Chapter 5. As the forcing is more related to the physical simulations, we leave its discussion for Section 5.1. Provided the solver is working correctly, the effects of the forcing should be entirely canceled in these simulations.

Once ran, the solver behaved quite well. At this temporal resolution progressing 7 days amounts to 1,008 time steps and not once did the solver fail to converge. In fact, as shown in Figure 3.5 the solver never got close to the maximum allowable iterations of 200; the mean iteration count to reach convergence was 8.40, with a standard deviation of 0.93.

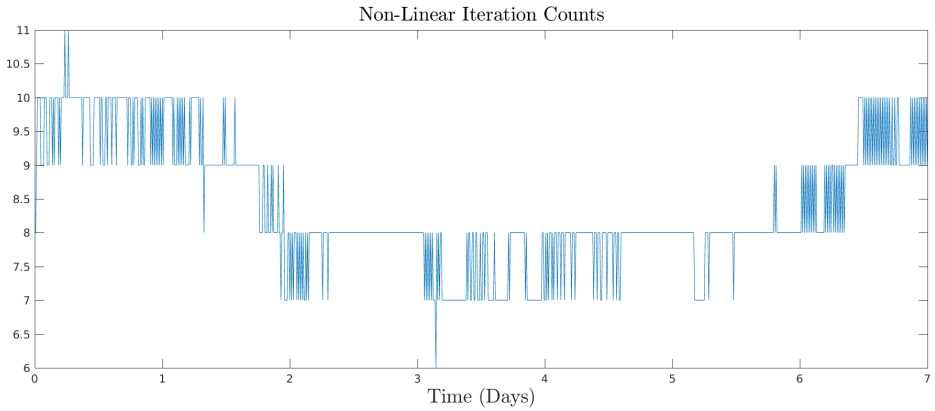
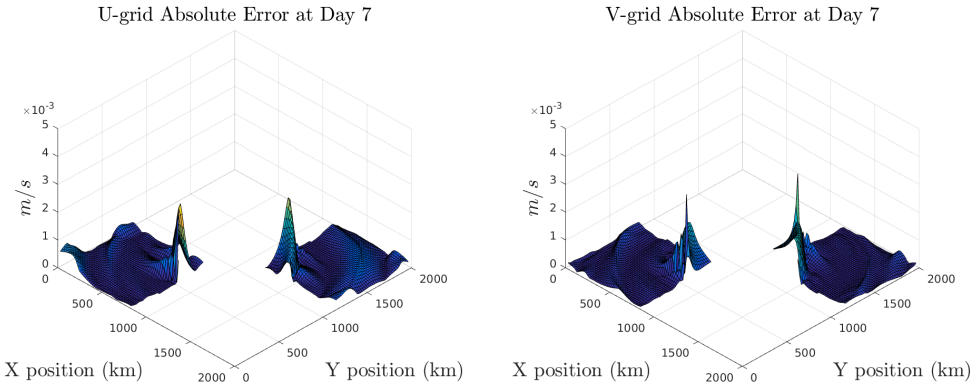


Figure 3.5: Validation Simulation: number of non-linear iterations per time step

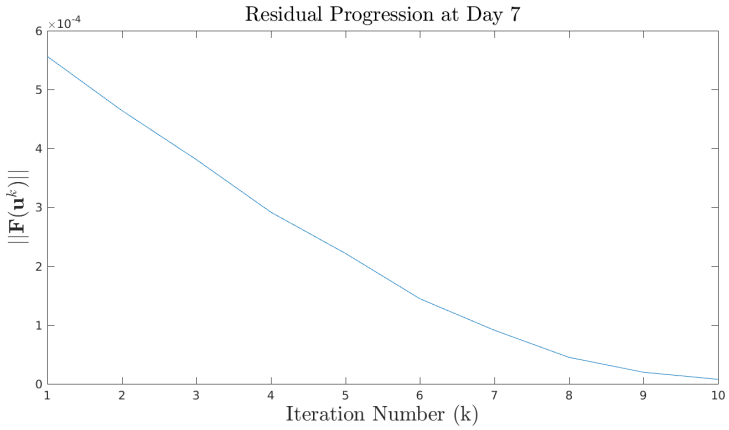
The solver was set to save the resulting solution and residual progress at the end of every day. Table 3.2 contains the resulting error norms at these benchmark time levels and as an example, Figure 3.6 shows the absolute error at the end of the simulation and the residual progression for the final solve, which represents the typical behaviour seen throughout the simulation. Although there are some larger errors near the centre of the domain in Figure 3.6a, they are relatively unnoticeable in the solution. This is easily confirmed by the series of plots of both the numerical solutions and the associated errors provided in Appendix A. Additionally, as seen in Table 3.2 the error measures for day 7 are the highest throughout the entire simulation, as one would expect because of error accumulation. However, according to Table 3.2, a major increase is seen only at day 7; interestingly the errors were stable and even slightly decreasing with time during the first 6 days.

Table 3.2: Validation Simulation Errors, $\boldsymbol{E} = \texttt{abs}(\boldsymbol{u}_{exact} - \boldsymbol{u})$

t	U-grid		V-grid	
(Days)	$\|\|\boldsymbol{E}\|\|_{L_2}$	$\|\|\boldsymbol{E}\|\|_{L_\infty}$	$\|\|\boldsymbol{E}\|\|_{L_2}$	$\|\|\boldsymbol{E}\|\|_{L_\infty}$
1	1.0366e-04	8.5029e-04	8.6420e-05	9.3697e-04
2	1.0238e-04	6.8575e-04	6.4094e-05	3.6776e-04
3	1.0003e-04	6.4721e-04	6.5920e-05	3.7804e-04
4	9.4786e-05	6.7398e-04	6.1935e-05	3.5718e-04
5	1.1281e-04	2.0831e-03	6.1343e-05	9.5595e-04
6	1.0384e-04	7.2041e-04	5.9857e-05	3.6952e-04
7	1.8859e-04	2.7447e-03	1.8719e-04	3.1491e-03



(a) Final absolute error ($t = 7$ days) for the validation simulation



(b) Typical progression of non-linear residual over each solve (shown $t = 7$ days)

Figure 3.6: Validation Simulation performance examples

3.3.2 Grid-Refinement Tests

With the successful simulation described above, we move on to convergence tests. Due to our fully second order discretization, it is expected that in the limiting behaviour as $dx, dt \to 0$, we will see second order convergence. To assess this, we perform grid-refinement tests and run the validation simulation with three different resolutions:

- *Coarse Resolution:* $dx|dt = 40$ km$|20$ min
- *Moderate Resolution:* $dx|dt = 20$ km$|10$ min
- *Fine Resolution:* $dx|dt = 10$ km$|5$ min.

With these simulations, the convergence rates were estimated between the coarse and

moderate resolutions and the moderate and fine resolutions; Table 3.3 below shows the resulting estimates.

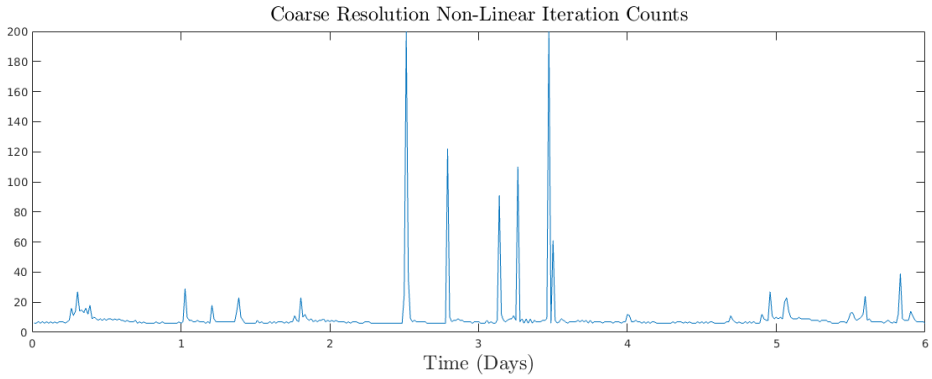
Table 3.3: Estimated Convergence Rates

Day	U-grid Convergence Rates				V-grid Convergence Rates			
	Coarse → Mod.		Mod. → Fine		Coarse → Mod.		Mod. → Fine	
	L_2	L_∞	L_2	L_∞	L_2	L_∞	L_2	L_∞
1	2.0	2.0	0.5	0.3	1.7	2.2	-1.1	-1.6
2	4.1	5.4	2.0	2.1	4.7	6.3	2.0	2.0
3	3.5	5.0	2.0	2.0	3.7	5.5	2.0	1.8
4	4.7	6.1	2.0	2.2	4.6	6.1	2.0	2.0
5	4.1	3.5	0.3	-1.2	4.8	4.9	0.2	-1.3
6	3.5	4.5	2.0	1.8	4.2	6.3	2.1	2.2

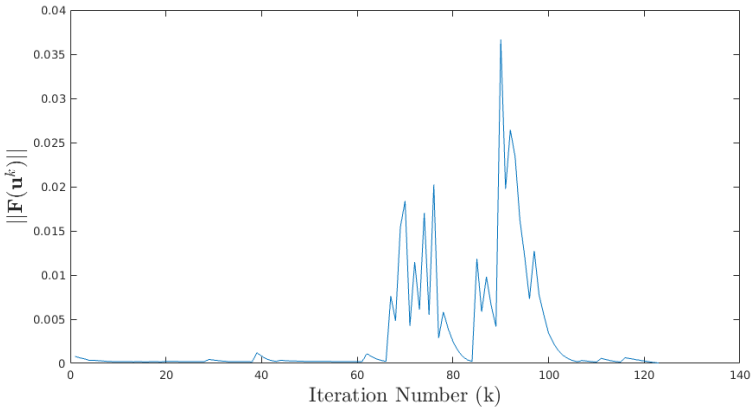
Note the second order convergence seen in both the L_2 and L_∞ norms for four of the six days measured when going from the moderate to fine resolution. As the theoretical convergence rate is only expected in the limiting behaviour as $dx, dt \to 0$, it is not surprising that it isn't observed every where, and thus these results are accepted as validating the second order convergence.

Now, although the solver is exhibiting the theoretical convergence rate, there are still some problems. For instance, beyond the first day, the convergence rates between the coarse and moderate resolutions are significantly higher than the theoretical expectation. This has been attributed to convergence problems noted for the coarse simulation. As shown in Figure 3.7a, the coarse resolution simulation had trouble converging at various times throughout the simulation, where many more than the standard sub 10 iterations were required; Figure 3.7b shows a specific example of one of these solves. When this occurred, it appears that the solver was close to the actual solution in the first few iterations but for unknown reasons, was unable to get below the specified tolerance. The solver then appears to search around for a solution, at some points making large steps away from it, until coming back to the solution neighbourhood and converging. Note that although for this specific case, convergence was reached, this is not always so. It is believed that this "searching" behaviour is what introduces the added error seen in the coarse simulation. This added error inflates

the difference between the coarse and moderate simulations, plaguing the convergence estimates.



(a) Non-linear iteration counts per time step



(b) Residual progression at $t = 2.8$ days

Figure 3.7: Coarse Simulation convergence problems

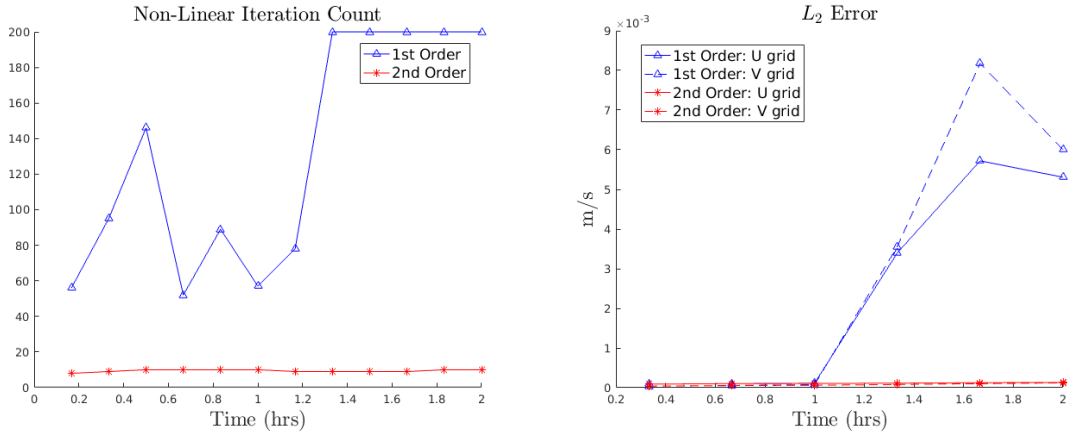
It should be noted that convergence problems were not limited to the coarse simulation. Although the fine simulation behaved very well over the first six days, taking ~ 10 iterations for every solve and producing acceptable error, after the sixth day the solver broke down and stopped converging altogether, which is why day 7 is not included in Table 3.3. The specific reason for this is unknown, but this problem and the issues with the coarse simulation point towards a resolution sensitivity that should be explored in future work. The occasional convergence failure of the JFNK method was also reported in [20, 18].

3.3.3 The Jacobian Approximation: Failure of the First Order Scheme

In addition to assessing the convergence rate of this solver, we also assess the effects of using the second order approximation for the Jacobian multiplication in (3.28). To determine if this second order approximation isn't overkill, we re-ran the moderate resolution simulation with the following first order approximation,

$$J(\boldsymbol{u}^k)\boldsymbol{v} \approx \frac{A(\boldsymbol{u}^k + \epsilon\boldsymbol{v})\boldsymbol{u}^k - A(\boldsymbol{u}^k)\boldsymbol{u}^k}{\epsilon} + A(\boldsymbol{u}^k)\boldsymbol{v} - D\boldsymbol{b}(\boldsymbol{u}^k)\boldsymbol{v}. \tag{3.38}$$

This simulation utilized the same parameters as those defined in Table 3.1 but due to convergence problems, the first order simulation only progressed two hours, or twelve time steps; solutions were saved every 20 minutes. Figure 3.8a shows a comparison of the resulting iteration counts between the second and first order simulation, while Figure 3.8b shows comparisons of the resulting error norms.

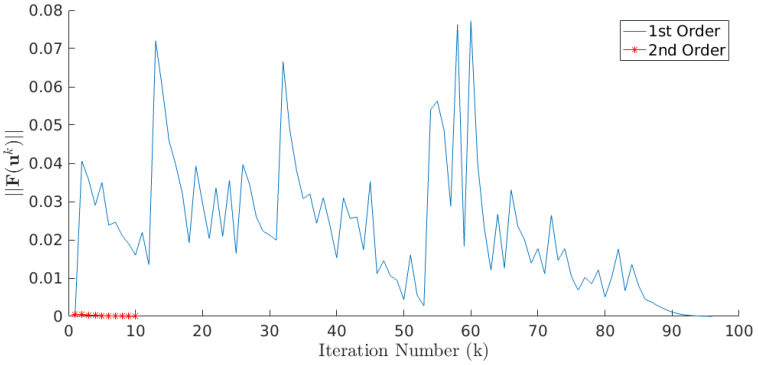


(a) Iteration Count (b) L_2 error

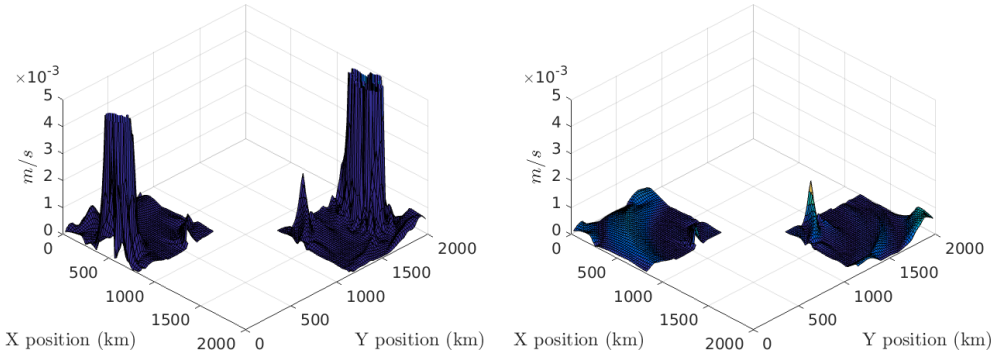
Figure 3.8: Comparison of solver performance with first and second order Jacobian approximations

Prior to the first order simulation failing at the one hour and twenty minute mark, it produced comparable errors to those from its second order counterpart. Regardless, as shown in Figure 3.8a, to produce these solutions the first order solver used significantly more computational resources and reached the solution through a rather spurious path (Figure 3.9a). Tests on the linear solver alone (not shown here) suggest that, using the first order approximation, there are some issues when solving the linear system at the first non-linear iteration; these troubles are likely to blame for

the large spike in the residual seen at the first iteration in Figure 3.9a. After the one hour and twenty minute mark, the first order simulation breaks down completely and is no longer capable of converging in 200 iterations, introducing the large errors seen in Figures 3.8b and 3.9b. Due to the significant reduction in computational resources and improvement in convergence behaviour, the second order approximation constitutes a drastic improvement for the JFNK approach for the SIME.



(a) Typical convergence behaviour (shown $t = 20$ min)



(b) Comparison of U-grid absolute error with first (left) and second (right) order approximation after solver breaks down (shown $t = 80$ min)

Figure 3.9: First order Jacobian solver convergence problems

3.3.4 Conditional Termination

Motivated by the searching behaviour seen in the coarse simulation, we additionally propose the use of "Conditional Termination" to limit the computational effort. For this solver to work, between time steps, the solution cannot undergo drastic changes. Therefore, as the previous time step's solution is used as the initial iterate for the non-linear solver, it should be "close" to the solution in the early iterates and large

steps away are likely detrimental. With this in mind, "Conditional Termination" adopts a "good enough" approach and, given a tolerance $r > 1$, when going from iteration k to $k+1$, if

$$||\boldsymbol{F}(\boldsymbol{u}^{k+1})|| > r||\boldsymbol{F}(\boldsymbol{u}^{k})||$$

it rejects $\boldsymbol{u}^{k+1}$ and accepts $\boldsymbol{u}^{k}$ as the solution, terminating the solver.

To test this method, we set $r = 2$ and re-ran the coarse simulation, as this was the only resolution that experienced this searching behaviour. We compared the solutions produced via Conditional Termination and those produced via the original simulation, referred to as the control; Figure 3.10 shows the resulting comparison. It can be seen that for all days considered, the Conditional Termination run produces comparable errors to those from the control simulation and in most cases there is actually a reduction.

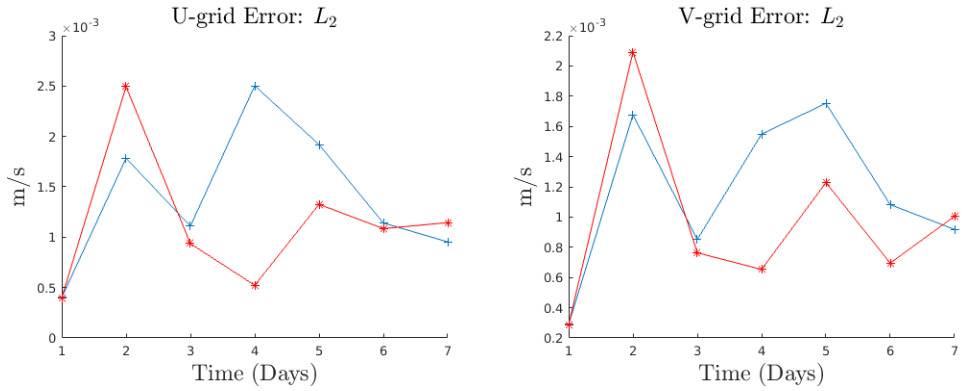


Figure 3.10: Conditional Termination (red) vs. Control (blue): Absolute error comparison

As this method doesn't significantly alter the error, the main benefit comes from saving computational effort. As discussed in Section 3.3.2, the solver sometimes steps away from the solution neighbourhood, searching sporadically until it eventually returns. This sporadic search uses valuable computational resources without the pay off of reduced error. The Conditional Termination saves this wasted effort by not allowing the solver to leave the solution neighbourhood. As can be seen in Figure 3.11, this leads to a significant reduction in the number of non-linear iterations when the solver has problems converging. In the context of ESMs, saving computational effort is a considerable benefit and thus this method should be considered and tested in the context of actual geophysical simulations.

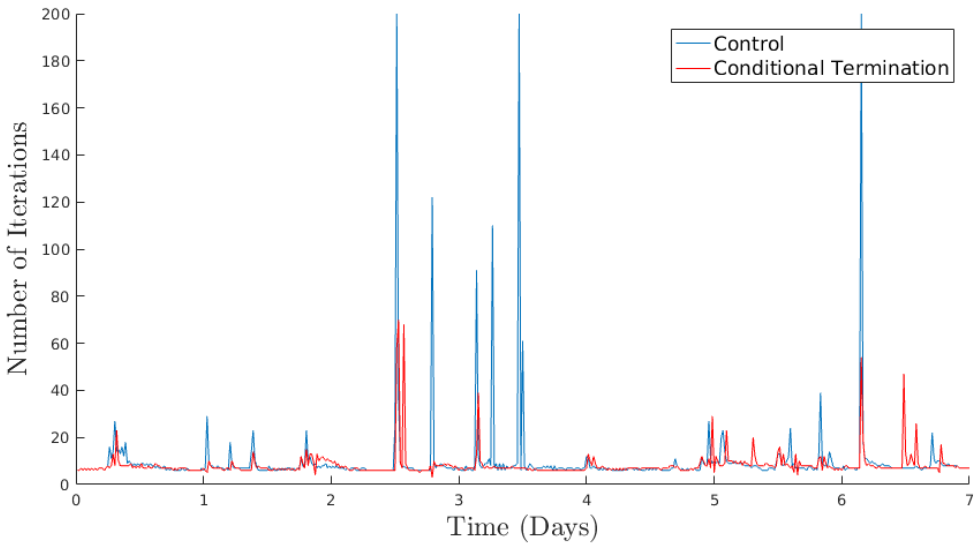


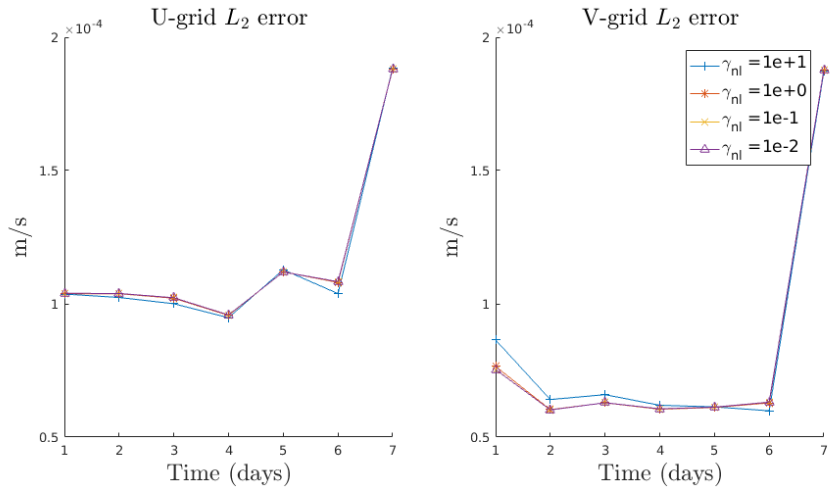
Figure 3.11: Conditional Termination vs Control: Non-linear iteration counts

3.3.5 Sensitivity to Ad-hoc Parameters

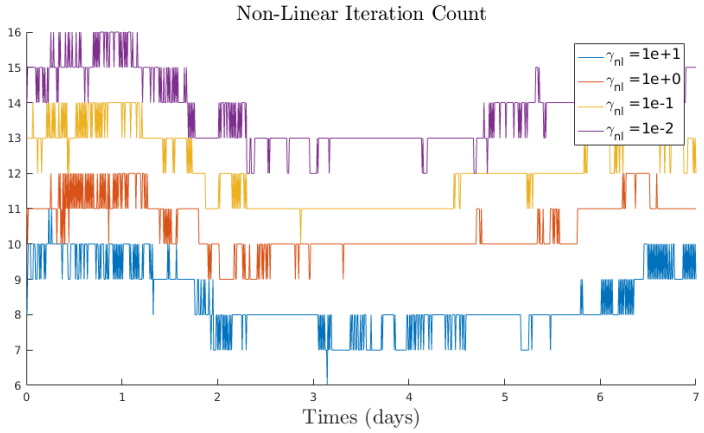
For our final numerical tests, we assess the impact of ad-hoc parameters, specifically the non-linear tolerance coefficient, γ_{nl}, and the small perturbation, ϵ, used in the approximation to the Jacobian multiplication.

Non-Linear Tolerance Coefficient: γ_{nl}

For the primary validation simulation, γ_{nl} was set to 10. To see if added accuracy could be gained by decreasing this value, we re-ran the simulation (at the moderate resolution) with $\gamma_{nl} = 1, 0.1$, and 0.01; Figure 3.12a shows the resulting L_2 errors at the end of each day and Figure 3.12b shows iteration counts over the entire simulation.



(a) L_2 error at the end of each day



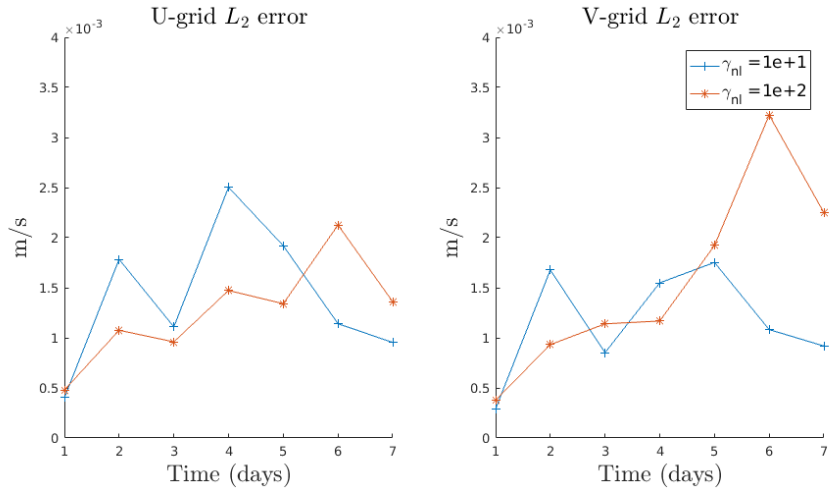
(b) Non-linear iteration counts per time step

Figure 3.12: Tighter non-linear tolerance tests

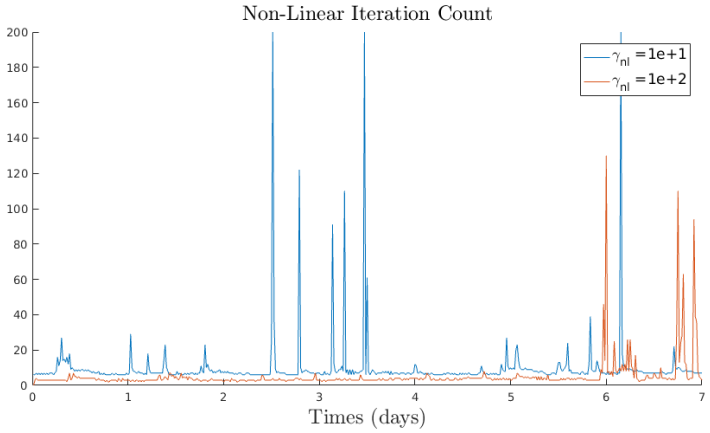
As can be seen from Figure 3.12, the solver still performed well with these tighter tolerances, consistently converging within 20 iterations. Nevertheless, to produce the lower residuals, additional iterations were required and no considerable change in error was noted. Thus, these tighter tolerances only result in wasted computational effort.

To see if added computational effort could be saved with $\gamma_{nl} > 10$, an additional run was attempted with $\gamma_{nl} = 100$ but the solver broke down at the 3 hour mark (18 time steps), consistently failing to converge in the allowable iteration count. This suggests that, for the validation simulation, at the moderate resolution, a choice of $\gamma_{nl} \sim 10$ is near the optimal value; setting it smaller only results in wasted computational resources while setting it larger causes the solver to break down. However,

it should be noted that the looser tolerance affects each resolution differently. For instance, with $\gamma_{nl} = 100$, the solver is capable of completing the coarse resolution simulation and preliminary tests show little affect on the fine simulation; Figure 3.13a shows the resulting error comparison for the coarse simulation with $\gamma_{nl} = 10$ and 100, while Figure 3.13b shows the iteration counts. While the looser tolerance results in similar errors, it significantly reduces the searching behaviour in the coarse simulation prior to day 6. This difference in behaviour between resolutions suggests that additional consideration is required in the derivation of the convergence formula (3.31); further study is required but perhaps a linear formula is more suitable.



(a) L_2 error at the end of each day

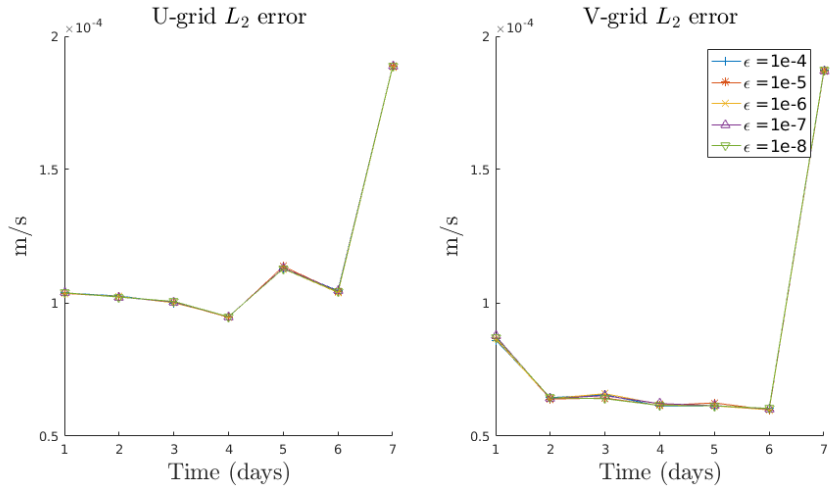


(b) Non-linear iteration counts per time step

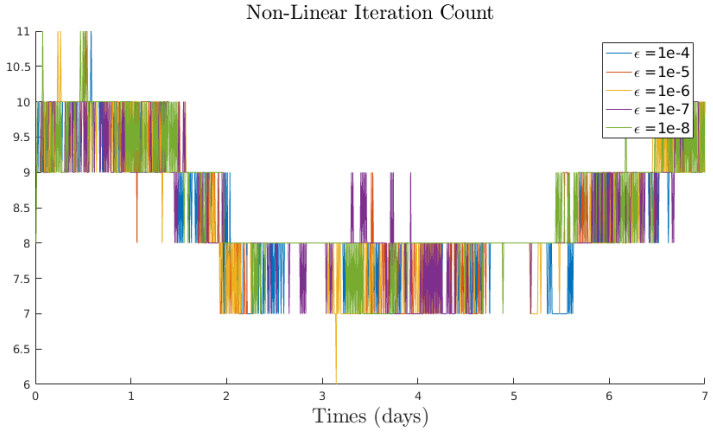
Figure 3.13: Looser non-linear tolerance tests with the coarse resolution

Jacobian Approximation Perturbation: ϵ

For the bulk of this work, we followed [18] and set $\epsilon = 10^{-6}$ as our default value. To assess the effect of this parameter, we re-ran the moderate resolution simulation with $\epsilon = 10^{-4}, 10^{-5}, 10^{-7}$, and 10^{-8}. Figure 3.14a shows the resulting error norms and Figure 3.14b shows the iteration counts. As can be seen, changing the value of ϵ in the Jacobian approximation results in negligible changes in both the errors and required non-linear iterations; there is little sensitivity to its value and thus $\epsilon = 10^{-6}$ is acceptable. In contrast, similar tests conducted with the first order solver (not shown) showed that the choice of ϵ can improve or reduce its ability to convergence, further highlighting the benefit of the second order approximation.



(a) L_2 error at the end of each day



(b) Non-linear iteration counts per time step

Figure 3.14: ϵ sensitivity tests

3.4 Conclusion

In this chapter we build on earlier work by Lemieux et al. [20, 18], and present JFNK solver for the highly non-linear SIME, with a fully second order discretization, using the *Crank Nicolson* scheme. Earlier implementations of this solver attained first order accuracy in time through a backward Euler approach. This improved accuracy is achieved without additional computational complexity as the inherent non-linear system remains the same. We also present a novel treatment of the rheology term in (2.1), circumventing the need to set ad-hoc boundary conditions for the viscosities. This is accomplished by using the smooth viscosity formulation, presented in [19], to create an analytical, closed form expression for the viscosity derivatives. This allows us to only impose boundary conditions on the prognostic variables $\boldsymbol{u}, h$, and A. Finally and most importantly, we improve on the simple first order approximation of the Jacobian matrix used in [18, 20] for the Newton iterations of the JFNK solver. Instead of the forward difference approach to the Gateaux derivative, we propose a more accurate formulation by expressing the linear terms and nearly linear terms in closed form and using a second order centred difference approach for the remaining highly non-linear term.

Numerical tests on a system of synthetic equations were conducted in Section 3.3. These tests allowed us to validate our solver and verify that our code, written in FORTRAN 90 for added portability and compatibility with existing ESMs, is functioning as desired. In Table 3.2, we see that for the moderate resolution simulation, the numerical error is well behaved and doesn't amplify significantly in time, likely attributed to the stability properties of the Crank Nicolson method. In Table 3.3, we also see that the resulting numerical solution appears to be converging at the expected second order rate. Although we don't see the theoretical rate everywhere, we still accept these tests as validation of the second order convergence. In an analytical sense, second order convergence is only expected in the limiting behaviour as $dx, dt \to 0$, and thus it is not surprising that for the highly non-linear SIME, we fail to see it across the board. It is expected that for points where the error doesn't decrease with the expected rate when moving from one resolution to the next, will be compensated by a rapid decrease when passing to the next grid. For instance, the mediocre convergence rate seen at day 5, in the L_2 norm (or the negative rate seen in the L_∞ norm) when going from the moderate to the fine grid is compensated by the large decrease seen when progressing from the coarse to moderate grid.

This validation simulation also showed us the efficiency of our JFNK solver, which is a very desirable feature in the context of ESMs. At the moderate resolution, over the 1,008 time steps, the solver never got close to the maximum iteration count of 200 – the maximum observed iteration count was 11! That being said, occasionally the solver does fail to converge in a reasonable number of iterations, especially for the coarse resolution simulation. Nevertheless, tests performed with the Conditional Termination approach (Section 3.3.4) show that formal convergence is not always required. Whenever an adequately large increase in residual is noted, halting the solver results in an equally accurate solution with significantly less non-linear iterations. This is due to the previous time step's solution being used for the initial iterate of the JFNK solver and the fact that the solution shouldn't change too much between consecutive time levels.

In Section 3.3.3, we compared our improved second order Jacobian approximation against its first order counterpart. As shown in Figure 3.8, we see that the JFNK solver performs significantly worse with the first order approximation – the solver isn't capable of converging within the imposed 200 iteration maximum after seven time steps! Prior to this breakdown, the first order solver produced comparable errors to those produced by the second order solver but, as seen in Figure 3.8a, required significantly more iteration counts, ranging between 60 and 160 in the first seven time steps. As an additional difference between these two approaches, we show in Section 3.3.5 that for values of ϵ ranging between 10^{-8} and 10^{-4}, the performance of the second order solver remains relatively unchanged, while for similar tests (not shown) on the first order solver have shown that altering ϵ can significantly alter its behaviour. This severe difference in behaviour differs significantly from the results in [20], which state that the added accuracy gained through a centred approach to the Gateaux derivative is negligible. This discrepancy is worth investigating further, but this has been left for future work. Nevertheless, the use of the second order approach to the Gateaux derivative in (3.28) proved crucial to our JFNK solver and is considered a drastic improvement to its first order counter part.

We additionally test our solver's sensitivity to the tolerance coefficient γ_{nl} in (3.31). As seen in Figure 3.12, at the moderate resolution and for $\gamma_{nl} \leq 10$ the solver's accuracy is fairly insensitive to its value and that tightening up the non-linear tolerance only leads to wasted computational effort. Considering that with $\gamma_{nl} = 100$, the moderate resolution simulation breaks down, one could suspect that the optimal value of γ_{nl} is near 10. However, when similar tests were completed for the coarse resolution,

the results in Figure 3.13 suggest that $\gamma_{nl} = 100$ is better suited. This steers us to believe that our proposed tolerance formula (3.31) is far from universal and that additional consideration must be given to this topic.

To conclude this discussion, it is worth noting that although early implementations of the JFNK solver [20, 18] achieved first order accuracy in time, more recently, Lemieux et al. presented another version of the solver [17] that is also fully second order, gaining the added accuracy through second order backward differencing. In this work, they note in passing that a Crank Nicolson approach was tried but not successful due to a resulting undamped oscillation caused by the water stress term. In these experiments, we failed to see these problems, but due to the idealized nature of these experiments and the fact that the forcing effects should be entirely canceled out, it is difficult to compare results. Additionally, the solver in [17] doesn't incorporate the common time-splitting approach for the SIME and the continuity equations, adding additional differences between solvers. Nevertheless, in [17] the same approximation for the Jacobian matrix from [20, 18] is used, and it is stated that altering the value of ϵ in their Gateaux derivative does affect the convergence behaviour of their solver, and the value is adjusted to improve robustness. This further highlights the improvement of our Jacobian approximation, as our implementation is insensitive to this value.

With our validated and improved JFNK solver for the SIME, we now move to presenting other crucial components required to produce a working sea ice dynamics solver.

Chapter 4

An Efficient Sea Ice Dynamics Solver with a Moving Boundary

Although the solving of (2.1) accounts for the majority of the complexity associated with sea ice dynamics, additional considerations are also required. To produce an actual sea ice simulation, our solver also incorporates Newton Damping of the JFNK solver; smoothing of the resulting solutions via Hyper-Viscosity; a second order scheme for advecting the sea ice according to (2.13) and (2.14); and a level set method for tracking the ice terminus. This chapter provides justification for and describes these considerations. Note that in this chapter we show the results of numerical tests performed with ice placed in the southwest and northeast corners of our domain; this domain and other details pertaining to these simulations will be discussed in detail in Chapter 5. This chapter is limited to the discussion of individual components of the solver, less the JFNK solver, required to produce working simulations.

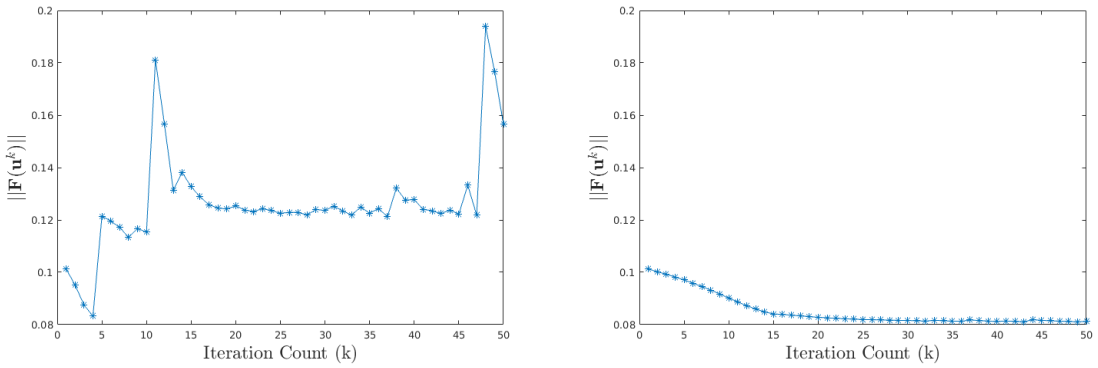
4.1 Newton Damping

Although, as shown in the previous chapter, for the synthetic equations (3.33) the JFNK solver generally behaves quite well when searching for a solution, this behaviour is not always mimicked for the true SIME. When attempting to solve equation (2.1) for the first time step, the residual quickly decreases and then jumps back up again (Figure 4.1a); this, and the searching behaviour discussed in Section 3.3, has been attributed to the solver overshooting the actual solution. In [4], it is shown for a

different set of equations, that this overshooting of Newton Algorithms can prevent convergence altogether. To remedy this, the author utilizes a Damped Newton Algorithm, where the step size is reduced by a factor greater than one, and shows that with the proper choice of a damping factor, convergence can be guaranteed. Although the SIME represents a completely different system, [4]'s results suggest that the same approach can be applied to improve the convergence behaviour of our algorithm. Thus, at each update of the non-linear iterate, we adopt the following damped approach,

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \frac{1}{\tau}\delta\boldsymbol{u}^k, \tag{4.1}$$

where τ is known as the damping factor. Figure 4.1 shows the resulting change in behaviour with damping activated with $\tau = 10$; it almost completely eliminates the spurious behaviour seen in 4.1a! Nevertheless, this improvement does come at the cost of slower convergence, but improving the solver's ability to reach convergence is of greater importance.



(a) No Damping

(b) Damping activated, $\tau = 10$.

Figure 4.1: Effect of Newton Damping on residual progression

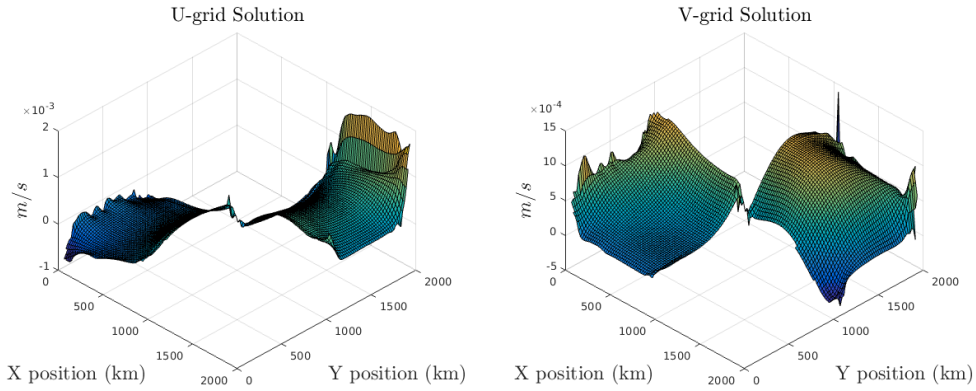
4.2 Smoothing with Hyper-Viscosity

Even with the damping described above, once the solver reaches it's minimum residual, small oscillations around this minima can cause small oscillations in the velocity solutions. When progressing in time, these small oscillations can grow and eventually lead to large jumps or spikes in the solutions that are completely non-physical. For instance, Figure 4.2a shows the final velocities after a week long simulation and clearly there are some non-physical jumps in the solutions. To deal with this, we

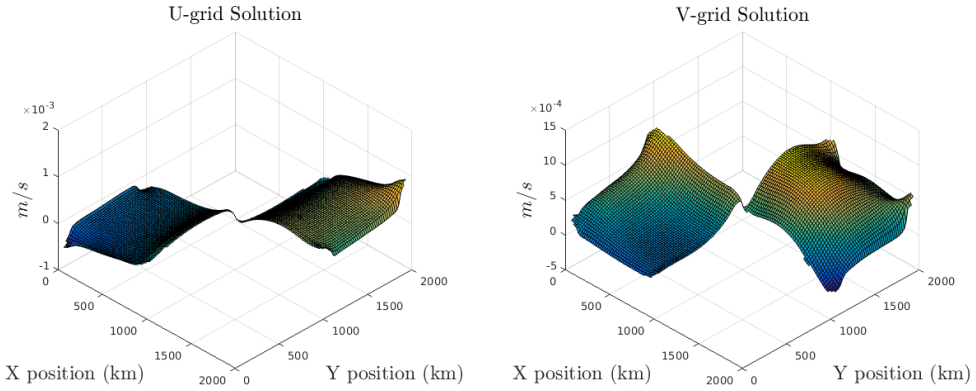
implement numerical smoothing via Hyper-Viscosity after each JFNK solve, which amounts to solving

$$\frac{\partial \boldsymbol{u}}{\partial t} = -\nu \nabla^4 \boldsymbol{u}, \tag{4.2}$$

for one artificial time-step, where ν is the hyper-viscosity parameter and $\nabla^4 = (\partial_{xxxx} + \partial_{yyyy})$. Equation (4.2) behaves much like an aggressive diffusion equation, smoothing out spikes and jumps while progressing in time, and with the proper choice of ν and time-step, it will prevent the large spikes present in Figure 4.2a from forming by eliminating smaller oscillations when they occur. For instance, with an artificial time-step of 1 hour, and $\nu = 1 \times 10^{12}$, Figure 4.2b shows the final velocities for the same test simulation that produced the solutions in Figure 4.2a – all the jumps and spikes are gone!



(a) Final solution without numerical smoothing



(b) Final solution with numerical smoothing

Figure 4.2: Effect of numerical smoothing

4.2.1 Choosing the Hyper-Viscosity Parameter

In choosing $\nu = 1 \times 10^{12}$, multiple considerations were required; a value too large would result in excessive smoothing, altering the solution too much, while a value too small would have negligible effect. To estimate a suitable value, we consider the two dimensional Fourier Transform of the u-component solution, i.e.

$$\hat{u}^{k,l}(t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} u(x,y,t)e^{-ikx}e^{-ily}dxdy, \tag{4.3}$$

where k and l are angular wave numbers, referred to simply as wave numbers from now on, and $i = \sqrt{-1}$. Using (4.3), we then transform the u-component of (4.2), i.e.

$$\widehat{\frac{\partial u}{\partial t}}^{k,l} = -\nu\left(\widehat{\frac{\partial^4 u}{\partial x^4}}^{k,l} + \widehat{\frac{\partial^4 u}{\partial y^4}}^{k,l}\right), \tag{4.4}$$

to produce the simple ODE,

$$\frac{\partial \hat{u}}{\partial t}^{k,l} = -\nu(k^4 + l^4)\hat{u}^{k,l}, \tag{4.5}$$

which affords the analytical solution,

$$\hat{u}^{k,l}(t) = \hat{u}^{k,l}(0)e^{-\nu(k^4+l^4)t}, \tag{4.6}$$

where $\hat{u}^{k,l}(0)$ is the initial wave form associated with the wave number pair (k,l). As can be seen from (4.6), depending on the wave numbers, the choice of ν determines how quickly waves in Fourier Space will be damped. Noting that numerical errors excite waves with large wave numbers, resulting in spikes that occur over a small number of grid points, we want to choose ν so we damp these wave numbers significantly, but have only a minor affect on the smaller wave numbers. Specifically, on our grid, we consider the following wave numbers

$$k_{tar} = l_{tar} = \frac{2\pi}{2dx} \tag{4.7}$$

$$k_{mod} = l_{mod} = \frac{2\pi}{4dx} \tag{4.8}$$

$$k_{min} = l_{min} = \frac{2\pi}{L_x}. \tag{4.9}$$

k_{tar} and l_{tar} are the largest wave numbers that will be seen on our grid so we wish to damp these significantly, while only damping k_{mod} and l_{mod} moderately, and k_{min} and l_{min} a negligible amount. To accomplish this, a damping factor of e^{-1} is deemed satisfactory for (4.8) and thus we wish to choose ν such that

$$|\nu(k_{mod}^4 + l_{mod}^4)|\hat{dt} \approx 1, \tag{4.10}$$

where $\hat{dt}$ is the artificial time-step. With $dx = 20\ km$ and choosing $\hat{dt} = 1$ hour, satisfying (4.10) amounts to setting $\nu \approx 3.6 \times 10^{12}\ m^4/s$.

Theoretically, choosing the above value of ν satisfies our goals, resulting in a damping factor of approximately e^{-16} for our target wave numbers, and $e^{-5\times10^{-9}}$ for k_{min} and l_{min}. However, we must also consider how we solve (4.2) numerically. Spatially, we utilize the same boundary conditions as those used in the SIME and approximate the 4th order derivatives using a second order centred difference scheme, i.e. at $u_{i,j}$

$$\frac{\partial^4 u}{\partial x^4} \approx \frac{u_{i+2,j} - 4u_{i+1,j} + 6u_{i,j} - 4u_{i-1,j} + u_{i,j-2}}{dx^4}, \tag{4.11}$$

and

$$\frac{\partial^4 u}{\partial y^4} \approx \frac{u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j-2}}{dx^4}. \tag{4.12}$$

Temporally, we use a first order forward difference formula, amounting to

$$\frac{\partial u}{\partial t} \approx \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\hat{dt}}, \tag{4.13}$$

at $u_{i,j}$. To limit computational resources, we use a fully explicit scheme, which in combining equations (4.11) - (4.13) with the u-component of (4.2) gives us the following numerical scheme

$$u_{i,j}^{n+1} = u_{i,j}^n - \frac{\nu\hat{dt}}{dx^4}\Big(u_{i+2,j}^n - 4u_{i+1,j}^n + 6u_{i,j}^n - 4u_{i-1,j}^n + u_{i,j-2}^n + u_{i,j+2}^n - 4u_{i,j+1}^n + 6u_{i,j}^n - 4u_{i,j-1}^n + u_{i,j-2}^n\Big). \tag{4.14}$$

As is typical with explicit schemes, we must be wary of numerical stability. To assess this, we use Von-Neumann analysis and utilizing the inverse transform of (4.3), consider our gridded solution in terms of the contribution from one wave number pair

(k, l) and the associated growth factor, ρ, at time level n,

$$u(x_i, y_j, t^n) = u^n_{i,j} = \rho^n e^{ikx} e^{ikl}, \tag{4.15}$$

Note that the i used in sub-scripts represent the index location of our solution, while those in the exponents are the complex solution to $\sqrt{-1}$. Now, by expressing our solution as (4.15) and utilizing properties of the L_2 norm and Fourier Transform, stability can be shown if $|\rho| \leq 1$ for all wave number pairs.

Plugging (4.15) into (4.14) and canceling the relevant terms results in

$$\rho = 1 - \frac{\nu \hat{dt}}{dx^4}\Big(12 + e^{i2kdx} - 4e^{ikdx} - 4e^{-ikdx} + e^{-i2kdx} + e^{i2ldx} - 4e^{ildx} - 4e^{-ildx} + e^{-i2ldx}\Big). \tag{4.16}$$

Then using Euler's Identity, $e^{ix} = \cos(x) + i\sin(x)$, and simplifying leads to

$$\rho = 1 - \frac{\nu \hat{dt}}{dx^4}(12 + 2\cos(2kdx) - 8\cos(kdx) + 2\cos(2ldx) - 8\cos(ldx)), \tag{4.17}$$

which we write more succinctly as

$$\rho = 1 - \frac{\nu \hat{dt}}{dx^4}\mathcal{H}(k, l). \tag{4.18}$$

Noting that $f(\theta) = 2\cos(2\theta) - 8\cos(\theta)$ is bounded as $-6 \leq f(\theta) \leq 10$, it can be seen that $0 \leq \mathcal{H}(k, l) \leq 32$, which gives the bounds for the growth factor,

$$1 - \frac{\nu \hat{dt}}{dx^4}32 \leq \rho \leq 1. \tag{4.19}$$

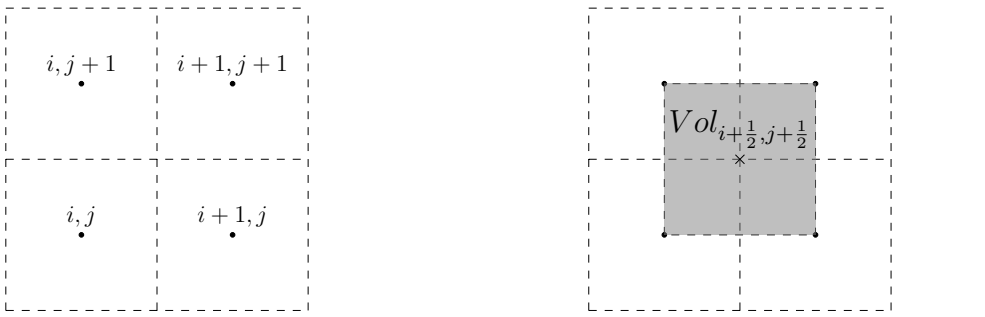
Therefore, for stability, we require $1 - \frac{\nu \hat{dt}}{dx^4}32 \geq -1$, which affords the limit of $\nu \leq 2.78 \times 10^{12}\ m^4/s$, and therefore our original estimate would result in an unstable scheme! This agrees precisely with numerical tests (not shown) as setting $\nu = 3.6 \times 10^{12}\ m^4/s$ resulted in additional oscillations attributed to numerical instabilities. This highlights worthwhile observation. For the theoretical model, (4.2), the value of $\nu = 3.6 \times 10^{12}$ m^4/s would be a suitable value to smooth out velocity solutions, but as (4.14) is simply a numerical approximation to (4.2), we must be wary of the errors it introduces as well; if we choose $\nu > 2.78 \times 10^{12}\ m^4/s$, not only we will add additional errors, but

these errors will grow in time. It should also be noted that if we take ν too close to its stability limit, we will also limit (4.14)'s ability to damp it's own errors as $|\rho| \to 1$ for certain wave numbers. Thus to compromise between damping errors introduced via (4.14) and smoothing those introduced via our JFNK solver, we set $\nu = 1 \times 10^{12}$ m^4/s.

4.3 Conservation of Thickness and Area Fraction

Once the JFNK solver produces acceptable results, the next step is to advect the thickness (h) and area fraction (A) according to the continuity equations (2.13) and (2.14). For this, we adopt the non-oscillatory central differencing, finite volume scheme from Nessayhu and Tadmor [28]. This method is based on a second order extension of the staggered Lax-Fredrichs (LxF) scheme, overcoming the heavy diffusion of the LxF scheme through the use of high resolution interpolants.

Prior to presenting the scheme, note that for physically motivated reasons, both h and A are forced to remain greater or equal to zero, and A is also capped at 1. Now, to describe this scheme, we consider the continuity equation for ice thickness on a simplified, non-staggered grid. Unless otherwise noted, the exact same derivation is applicable to (2.14). The same ideas of this derivation are extended to the Arakawa C-grid, but additional book keeping headaches are required due to the multiple staggered grids. It should also be noted, that due to the time-splitting nature of the dynamics solver, the velocity fields are assumed to be constant for each advection step.



(a) The non-staggered grid (b) Staggered volume at $(i + \frac{1}{2}, j + \frac{1}{2})$

Figure 4.3: The simplified grid

Consider the simplified grid in Figure 4.3a and the staggered volume in 4.3b. Note that we let Δx be the grid resolution instead of dx and Δt be the temporal resolution

instead of dt to avoid confusion with the differentials in the integrals below. With this in mind, we assume a uniform grid resolution of Δx and begin by integrating (2.13) over the staggered volume at $(i+\frac{1}{2}, j+\frac{1}{2})$ to get

$$\int_{y_j}^{y_{j+1}} \int_{x_i}^{x_{i+1}} \frac{\partial h}{\partial t}\, dxdy = -\int_{y_j}^{y_{j+1}} \int_{x_i}^{x_{i+1}} \nabla \cdot (\boldsymbol{u}h) dxdy, \tag{4.20}$$

which by assuming that h varies smoothly in time, can be re-written as

$$\frac{d}{dt}\int_{y_j}^{y_{j+1}} \int_{x_i}^{x_{i+1}} h\, dxdy = -\int_{y_j}^{y_{j+1}} \int_{x_i}^{x_{i+1}} \nabla \cdot (\boldsymbol{u}h) dxdy. \tag{4.21}$$

Then, through the Divergence Theorem, we write the right hand side as

$$\begin{aligned}\frac{d}{dt}\int_{y_j}^{y_{j+1}} \int_{x_i}^{x_{i+1}} h\, dxdy &= -\int_{\partial\Omega} \boldsymbol{n} \cdot (\boldsymbol{u}h) ds \\ &= -\int_{y_j}^{y_{j+1}} \left[(uh)_{i+1} - (uh)_i\right] dy - \int_{x_i}^{x_{i+1}} \left[(vh)_{j+1} - (vh)_j\right] dx\end{aligned} \tag{4.22}$$

where $\partial\Omega$ is the boundary of the staggered volume. We then integrate with respect to time, from time level $n \to n+1$ and divide by Δx^2 to consider the average thickness on the left hand side, i.e.

$$\begin{aligned}\bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \bar{h}^{n}_{i+\frac{1}{2},j+\frac{1}{2}} = &-\frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}} \int_{y_j}^{y_{j+1}} \left[(uh)_{i+1} - (uh)_i\right] dydt \\ &- \frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}} \int_{x_i}^{x_{i+1}} \left[(vh)_{j+1} - (vh)_j\right] dxdt,\end{aligned} \tag{4.23}$$

and it is now at this stage where approximations come into play.

On the right hand side of (4.23), we approximate the spatial integrals using the second order trapezoidal rule, i.e. for an arbitrary function, $f(x)$, the integral from a to b can be approximated via

$$\int_a^b f(x)\, dx \approx \frac{(b-a)}{2}\left[f(a) + f(b)\right], \tag{4.24}$$

and thus applying (4.24) to (4.23), we arrive at

$$\bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \bar{h}^{n}_{i+\frac{1}{2},j+\frac{1}{2}} \approx -\frac{1}{2\Delta x}\int_{t^n}^{t^{n+1}} \left[(uh)_{i+1,j+1} + (uh)_{i+1,j} - (uh)_{i,j+1} - (uh)_{i,j}\right] dt \\ - \frac{1}{2\Delta x}\int_{t^n}^{t^{n+1}} \left[(vh)_{i+1,j+1} + (vh)_{i,j+1} - (vh)_{i+1,j} - (vh)_{i,j}\right] dt. \quad (4.25)$$

To approximate the temporal integrals we use the Mid-Point Euler Method, amounting to the second order, predictor-corrector method,

$$\bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \bar{h}^{n}_{i+\frac{1}{2},j+\frac{1}{2}} \approx -\frac{\Delta t}{2\Delta x}\left[(uh)^{n+\frac{1}{2}}_{i+1,j+1} + (uh)^{n+\frac{1}{2}}_{i+1,j} - (uh)^{n+\frac{1}{2}}_{i,j+1} - (uh)^{n+\frac{1}{2}}_{i,j}\right] \\ - \frac{\Delta t}{2\Delta x}\left[(vh)^{n+\frac{1}{2}}_{i+1,j+1} + (vh)^{n+\frac{1}{2}}_{i,j+1} - (vh)^{n+\frac{1}{2}}_{i+1,j} - (vh)^{n+\frac{1}{2}}_{i,j}\right], \quad (4.26)$$

where

$$\begin{aligned} h^{n+\frac{1}{2}}_{i,j} &= h^n_{i,j} - \frac{\Delta t}{2}\left[\tilde{\partial}_x(uh) + \tilde{\partial}_y(vh)\right]^n_{i,j} \\ &= h^n_{i,j} - \frac{\Delta}{2}\left[h\tilde{\partial}_x u + u\tilde{\partial}_x h + h\tilde{\partial}_y v + v\tilde{\partial}_y h\right]^n_{i,j}, \end{aligned} \quad (4.27)$$

and $\tilde{\partial}_x$ and $\tilde{\partial}_y$ are numerical derivatives which will be discussed below.

As discussed in [28], to combat the heavy diffusion typically seen with the LxF scheme, the staggered average

$$\bar{h}_{i+\frac{1}{2},j+\frac{1}{2}} = \frac{1}{\Delta x^2}\int_{y_j}^{y_{j+1}}\int_{x_i}^{x_{i+1}} h\, dxdy, \quad (4.28)$$

must be calculated beyond the zeroth order, piecewise constant approximation. Thus, we use the first order piecewise linear approximation

$$\bar{h}_{i+\frac{1}{2},j+\frac{1}{2}} \approx \frac{1}{4}(h_{i,j}+h_{i,j+1}+h_{i+1,j+1}+h_{i+1,j}) + \frac{\Delta x}{16}(\tilde{\partial}_x h_{i,j} + \tilde{\partial}_x h_{i,j+1} - \tilde{\partial}_x h_{i+1,j+1} - \tilde{\partial}_x h_{i+1,j}) \\ + \frac{\Delta x}{16}(\tilde{\partial}_y h_{i,j} - \tilde{\partial}_y h_{i,j+1} - \tilde{\partial}_y h_{i+1,j+1} + \tilde{\partial}_y h_{i+1,j}). \quad (4.29)$$

To maintain the desired second order accuracy of this scheme, the numerical derivatives in (4.27) and (4.29) must be evaluated with at least first order accuracy [28]. Therefore, for this implementation, we use second order centred differences. Unfortunately, while the velocity fields should vary smoothly in space, discontinuities

in h and A are not only possible, they are physically expected and as a result, a naive implementation of centred differences could result in spurious oscillations in these regions. To combat this and create a non-oscillatory scheme, we utilize van Leer's monotonized central-difference (MC) limiter [22] for the thickness and area fractions, i.e. for the x-derivative of h,

$$\tilde{\partial}_x h_{i,j} = \text{minmod}\left(\frac{2(h_{i+1,j} - h_{i,j})}{\Delta x}, \frac{h_{i+1,j} - h_{i-1,j}}{2\Delta x}, \frac{2(h_{i,j} - h_{i-1,j})}{\Delta x}\right). \tag{4.30}$$

In smooth regions, this reduces to centred differences and achieves our accuracy goals. Near discontinuities, this limiter gives up second order accuracy to avoid introducing oscillations. It should also be noted that (4.30) maintains relatively sharp jumps in comparison to the more basic minmod limiter, yet avoids artificially steepening smooth regions like the superbee limiter [22].

Noting that once the staggered volumes are updated to time level $n+1$, the updated values must be mapped back to the non-staggered grid via

$$\begin{aligned} h_{i,j}^{n+1} = \frac{1}{4}(&\bar{h}^{n+1}_{i-\frac{1}{2},j-\frac{1}{2}} + \bar{h}^{n+1}_{i-\frac{1}{2},j+\frac{1}{2}} + \bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} + \bar{h}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2}}) \\ &+ \frac{\Delta x}{16}(\tilde{\partial}_x \bar{h}^{n+1}_{i-\frac{1}{2},j-\frac{1}{2}} + \tilde{\partial}_x \bar{h}^{n+1}_{i-\frac{1}{2},j+\frac{1}{2}} - \tilde{\partial}_x \bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \tilde{\partial}_x \bar{h}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2}}) \\ &+ \frac{\Delta x}{16}(\tilde{\partial}_y \bar{h}^{n+1}_{i-\frac{1}{2},j-\frac{1}{2}} - \tilde{\partial}_y \bar{h}^{n+1}_{i-\frac{1}{2},j+\frac{1}{2}} - \tilde{\partial}_y \bar{h}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} + \tilde{\partial}_y \bar{h}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2}}), \end{aligned} \tag{4.31}$$

the above scheme can be succinctly summarized in the following steps:

1. At t^n, calculate the staggered averages according to (4.29) and (4.30).
2. Update the non-staggered values to $t^{n+\frac{1}{2}}$ according to (4.27) and (4.30).
3. Update the staggered averages to t^{n+1} via (4.26).
4. Update the non-staggered values to t^{n+1} via (4.31).

When considering boundary conditions, Dirichlet conditions are automatically enforced for both A and h at open and land boundaries, as they equal zero when no ice is present. Nevertheless, special consideration is required at land boundaries in order to limit the numerical diffusion into land cells. Due to the nature of this problem, staggered averages are required at the corner of land cells when ice is attached. As no ice is present in land cells, the averages would be artificially low at these points.

To account for this, we don't include the area contribution from land cells when the staggered averages are being calculated, and thus the averages agree more with the actual values in the neighbouring ice cells.

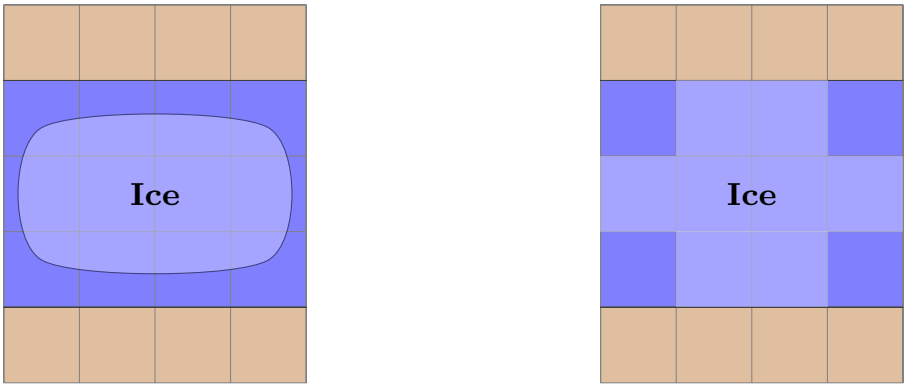
With this scheme presented, it is worth stating that one of the primary benefits of these methods are that due to their staggered nature, we integrate over the discontinuities and for hyperbolic systems, negate the need to solve the Riemann Problem. This benefit is lost for our specific application, as due to the Arakawa C-grid when we solve the SIME we produce velocities at the edges of each cell and no Riemann Solver is required. Regardless, this method was chosen as it provided an accurate and efficient method to approximate (2.13) and (2.14) that was relatively easy to implement.

4.4 Masking the Computational Domain with a Level Set Method

As mentioned at the beginning of this chapter, we perform numerical tests with two separate ice packs, one in the southwest corner and the other in the northeast. Additionally, in Chapter 3, our validation simulation was performed on a similarly limited domain. These domains should bring two questions to mind: (1) why do we limit calculations to these domains and (2) how do we limit them? In terms of (1), we limit our domains as ice doesn't cover the entire globe! Sea ice solvers must incorporate a method for constricting domains to where ice is actually present. Additionally, as an added complication, this domain must have the ability to move as, temporally, sea ice extent varies significantly, i.e. there is much more sea ice in the Arctic during the winter months than the summer, a fact that is becoming even more pronounced with global warming. Therefore, it is imperative that sea ice solvers have the ability to dynamically set domains.

In addressing how the domain is limited, a relatively straight forward method is a volume cut-off procedure. The volume is calculated in each cell throughout the domain according to the ice thickness (h) and area fraction (A). If the volume is above a certain threshold the cell is considered ice, else it is considered open water or land. This method ultimately leads to a discrete treatment of the boundary - it always fall on edges of grid cells. Considering that standard grid cells for these solvers have resolutions on the order of 10 to 100 km [18], this is an obvious limitation, particularly

near land boundaries. Consider the situation in Figure 4.4, where the true domain is seen in Figure 4.4a. If a volume cut-off method is used to set what we call the computational domain, the resulting configuration is shown in 4.4b. The solver will treat the ice as being attached to land, enforcing a Dirichlet condition of $\boldsymbol{u} = 0$, when in reality, Neumann conditions should be applied. Another issue with this method (and those like it) is that it relies on advecting the strictly positive scalars, h and A. Due to the strictly positive nature of these values, cusps will always be present at the ice terminus, making schemes that advect these values particularly susceptible to numerical diffusion, increasing uncertainty in the boundary location. Nevertheless, despite these limitations, methods like these are easy and cheap to implement and to the best of the author's knowledge, are standard practice. For example, in the current release of the widely used Los Alamos Sea Ice Model (CICE), a combined mass/area cutoff method is used to set the ice extent masks [2].



(a) True Domain (b) Computational Domain

Figure 4.4: Setting the computational domain with a volume cutoff method

For our solver, we avoid these discrete methods and borrow a method commonly used to track interfaces in two-phase flows [33], the signed distance function, $\phi(\boldsymbol{x})$, which satisfies

$$\begin{aligned} |\nabla\phi(\boldsymbol{x})| &= 1 \quad \text{for} \quad \boldsymbol{x} \in (\Omega \cup \Omega^C) \\ \phi(\boldsymbol{x}) &> 0 \quad \text{for} \quad \boldsymbol{x} \in (\Omega - \partial\Omega) \\ \phi(\boldsymbol{x}) &= 0 \quad \text{for} \quad \boldsymbol{x} \in \partial\Omega \\ \phi(\boldsymbol{x}) &< 0 \quad \text{for} \quad \boldsymbol{x} \in \Omega^C, \end{aligned} \tag{4.32}$$

where Ω is the ice pack domain and $\partial\Omega$ is its boundary. To use this method, once the

solution to (4.32) is produced, we simply limit our calculations to Ω, where $\phi \geq 0$. As we use the signed distance function, there is no longer a cusp at the ice terminus. Additionally, and more importantly, this method also affords sub-grid resolution near boundaries. Although we will still only have values for ϕ at a discrete number of points, we can use those values and its gradients near the terminus to estimate the location of boundaries with-in cells! Therefore this method would result in a better treatment of the configuration in Figure 4.4a as it could tell where the ice is actually connected to land or not.

4.4.1 Solving the Distance Function

Existence and uniqueness proofs for (4.32) are provided in [31], but to produce the solution to (4.32), we implement the level set approach described in Sussman et al [33]. According to our desired domain, we create the function $\phi_0(\boldsymbol{x})$ such that $\phi_0(\boldsymbol{x}) = 0$ is the ice boundary; this function need not be the distance function. We then define $\phi(\boldsymbol{x})$ to be the signed distance function sharing the same zero level set as $\phi_0(\boldsymbol{x})$, with $\phi(\boldsymbol{x}) > 0$ inside the ice pack, and $\phi(\boldsymbol{x}) < 0$ outside. We produce $\phi(\boldsymbol{x})$ by solving the following differential equation to steady state,

$$\begin{aligned} \frac{\partial \phi}{\partial t} &= S(\phi_0)(1 - \sqrt{\phi_x^2 + \phi_y^2}) \\ \phi(\boldsymbol{x}, 0) &= \phi_0(\boldsymbol{x}), \end{aligned} \tag{4.33}$$

where S is a smooth sign function, defined as

$$S(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2 + \epsilon_d^2}}, \tag{4.34}$$

and ϵ_d is a "small" constant implemented for numerical purposes; we follow [33] and set ϵ_d to be the grid resolution dx. Due to this formulation of (4.34), (4.33) has the property that its solution remains unchanged at the interface and thus finding its steady state equates to finding the solution of (4.32), as $\phi_t = 0$ is equivalent to $|\nabla \phi(\boldsymbol{x})| = 1$. Unfortunately, (4.33) is a non-linear differential equation and thus care must be taken to produce its solution. To accomplish this, we utilize the non-linear scheme presented in Sussman et al [33]. First, noting that we place $\phi(\boldsymbol{x})$ on the tracer

point grid, we set the one sided derivatives at point (i, j) via

$$\begin{aligned} a &= (\phi_{i,j} - \phi_{i-1,j})/dx \\ b &= (\phi_{i+1,j} - \phi_{i,j})/dx \\ c &= (\phi_{i,j} - \phi_{i,j-1})/dx \\ d &= (\phi_{i,j+1} - \phi_{i,j})/dx, \end{aligned}$$

and define the function

$$G(\phi_{i,j}) = \begin{cases} \sqrt{\max((a^+)^2, (b^-)^2) + \max((c^+)^2, (d^-)^2)} - 1 & \text{if} \quad \phi^0_{i,j} > 0 \\ \sqrt{\max((a^-)^2, (b^+)^2) + \max((c^-)^2, (d^+)^2)} - 1 & \text{if} \quad \phi^0_{i,j} < 0 \\ 0 & \text{if} \quad \phi^0_{i,j} = 0, \end{cases} \tag{4.35}$$

where $\phi^0_{i,j}$ is ϕ_0 at point (i, j), and the positive and negative exponents denote the positive or negative part, i.e. $a^+ = \max(0, a)$ and $a^- = \min(a, 0)$. Then, we use (4.35) to update ϕ according to

$$\phi^{n+1}_{i,j} = \phi^n_{i,j} - \Delta t_\phi S(\phi^0_{i,j}) G(\phi^n_{i,j}), \tag{4.36}$$

where Δt_ϕ is the artificial time-step set to $dx/10$ for our application. As discussed in [33], the above method is a consistent and monotone scheme for (4.32) and is known to converge to the unique solution.

Initialization and Convergence

During the initialization phase of the solver, to create our idealized domain, we begin by setting $\phi^0_{i,j} = dx/2$ for all points that are to be considered ice, else $\phi^0_{i,j} = -dx/2$. With ϕ_0 initialized, we then use (4.35) and (4.36) until the following stopping criterion is met

$$U^n = \frac{\sum_{|\nabla\phi| \neq 0} |\phi^n_{i,j} - \phi^{n-1}_{i,j}|}{M} < \frac{\beta \Delta t_\phi dx^2}{L_x^2}, \tag{4.37}$$

where M is the number of points included in the sum, and β is a numerical parameter set to 1.5. Note that the requirement that $|\nabla\phi| \neq 0$ is enforced to stop contribution from points that haven't been updated yet (i.e. they are too far from the ice boundary). As a result of this, relatively few points would be considered in the early iterations and thus we only check this stopping criteria provided a minimum

of 50 iterations have been completed.

To validate the initialization of (4.32), we assessed the solver's ability to produce the correct solution for the domain shown below in Figure 4.5, with $L_x = L_y = 2000$ km. Four different resolutions were tested to allow for estimations of the convergence rate of this solver. Additionally, for the same reason, the ratio between dx and Δt_ϕ was fixed at $dx/\Delta t_\phi = 10$. The spatial resolutions tested were $dx = 5, 10, 20$, and 40 km, Figure 4.6 shows the resulting contours and Figure 4.7 shows the progression of (4.37) for each resolution.

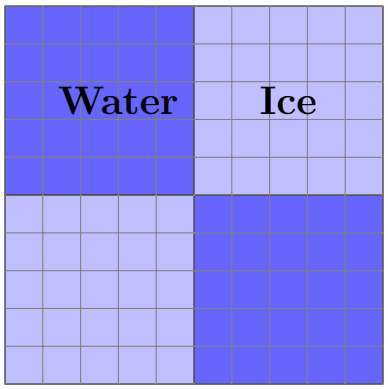
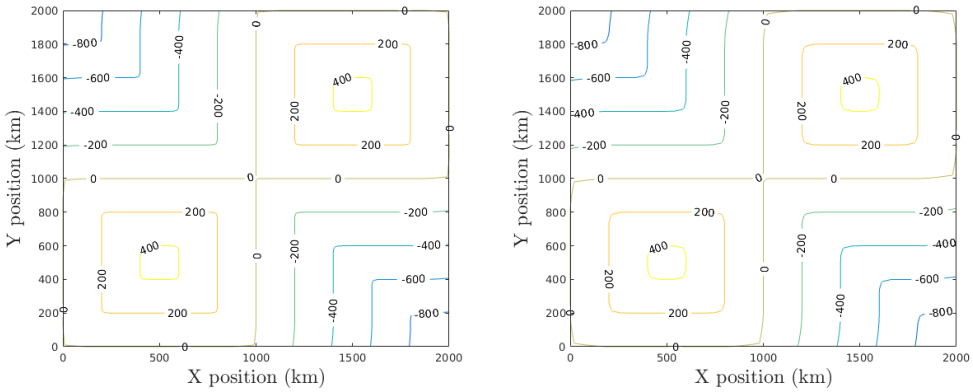
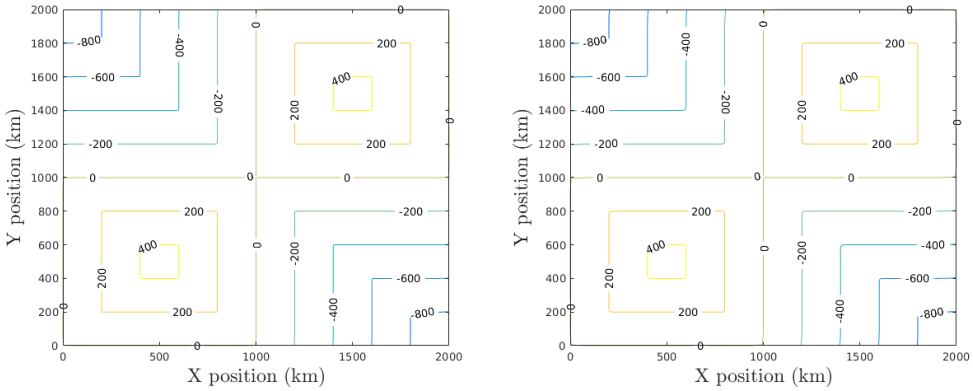


Figure 4.5: Initial computational domain



(a) Left: $dx = 20$ km, Right: $dx = 40$ km



(b) Left: $dx = 5$ km, Right: $dx = 10$ km

Figure 4.6: Distance function intialization results

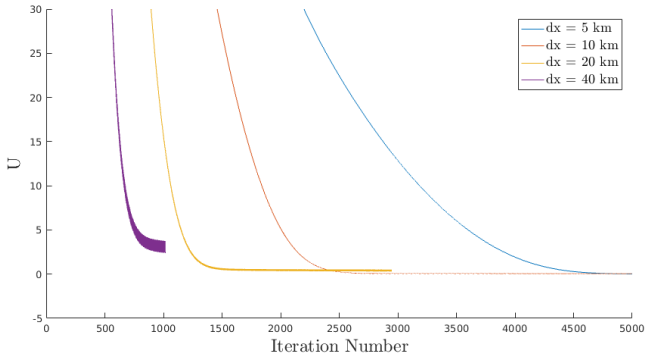


Figure 4.7: Distance function convergence behaviour

As can be seen in Figure 4.6, the solver appears to be producing the expected solution. Additionally, Figure 4.7 shows behaviour that definitely suggests that it is

converging to something, but exactly what is unknown as we do not have a closed form solution to (4.32). For a more robust validation we calculate the convergence rate for these tests, which should be 1 as the scheme is first order in both time and space. Much like the convergence tests done in Chapter 3, we could assess the convergence rate for a modified form of (4.32) that does afford exact solutions, but we instead compare our produced solutions at three different resolutions, which allows us to avoid considering the exact solution. To see how, consider the pointwise truncation error with an arbitrary resolution, dx,

$$\epsilon_{i,j}(dx) = \hat{\phi}_{i,j}(dx) - \phi(x_{i,j}), \tag{4.38}$$

where $\hat{\phi}$ and ϕ represent the approximated and exact solutions, respectively. Now, although we don't have the exact solution, we can get around this and recover the truncation error by considering the solution difference for two different resolutions, i.e. for dx and $dx/2$

$$\begin{aligned} \hat{\phi}_{i,j}(dx) - \hat{\phi}_{i,j}(dx/2) &= \epsilon_{i,j}(dx) + \phi(x_{i,j}) - \epsilon_{i,j}(dx/2) - \phi(x_{i,j}) \\ &= \epsilon_{i,j}(dx) - \epsilon_{i,j}(dx/2). \end{aligned} \tag{4.39}$$

With the truncation error in hand, we can estimate the convergence rate for a pth order scheme, by noting that

$$\epsilon_{i,j}(dx) \approx C dx^p, \tag{4.40}$$

so through (4.39), we can write

$$\hat{\phi}_{i,j}(dx) - \hat{\phi}_{i,j}(dx/2) \approx C dx^p - C(dx/2)^p. \tag{4.41}$$

Then, if we check the difference between $\hat{\phi}(dx)$ and $\hat{\phi}(dx/4)$, and consider the ratio

$$R = \frac{\hat{\phi}_{i,j}(dx) - \hat{\phi}_{i,j}(dx/4)}{\hat{\phi}_{i,j}(dx/2) - \hat{\phi}_{i,j}(dx/4)} \approx \frac{dx^p - dx^p/4^p}{dx^p/2^p - dx^p/4^p}, \tag{4.42}$$

through some re-arranging we arrive at

$$R \approx 2^p + 1, \tag{4.43}$$

thus we can estimate the convergence rate via

$$p \approx \frac{\log(R-1)}{\log(2)}. \tag{4.44}$$

Note that the above equations consider the *pointwise* truncation error, but a similar behaviour is expected in the *global* truncation error. Thus, to assess global convergence we define

$$R = \frac{||\hat{\phi}(dx) - \hat{\phi}(dx/4)||}{||\hat{\phi}(dx/2) - \hat{\phi}(dx/4)||}, \tag{4.45}$$

where the norms considered are the L_1, L_2 and L_∞. Letting $T1$ represent the test considering $dx = 40, 20$, and 10 km, and $T2$ represent the test using $dx = 20, 10$, and 5 km, Table 4.1 below shows the estimated convergence rates. As can be seen, we are seeing our expected convergence rate of $p \approx 1$, and we can thus be confident that our distance function solver is working correctly.

Table 4.1: Distance Function Convergence Rates

Test	L_1	L_2	L_∞
$T1$	1.1	1.3	1.1
$T2$	1.1	1.3	1.1

It should be noted that to perform this test we required $\hat{\phi}$ at the same number of grid points for all four resolutions. To produce values from the coarser resolutions at the same points as the finer resolutions, linear interpolation was used. Although this does add error, these interpolations are second order in space and thus the scheme's truncation error dominates, so the convergence rate estimates in Table 4.1 are unaffected.

4.4.2 Advecting and Correcting the Distance Function

After the original initialization of ϕ, in order to track the moving ice pack, we must also advect it according to

$$\frac{\partial \phi}{\partial t} + (\boldsymbol{u} \cdot \nabla)\phi = 0, \tag{4.46}$$

after the JFNK solver produces $\boldsymbol{u}$. Note that (4.46) is different than those used to advect the ice thickness and area fraction as it is not a conservation law. Nevertheless, we adopt a very similar scheme as that described in Section 4.3 by writing (4.46) in

balance law form, or as a hyperbolic conservation law with a source term, i.e.

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\boldsymbol{u}\phi) = \phi(\nabla \cdot \boldsymbol{u}). \tag{4.47}$$

After going through the same derivation as that laid out in equations (4.20) - (4.23), the above equation becomes

$$\begin{aligned} \bar{\phi}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \bar{\phi}^{n}_{i+\frac{1}{2},j+\frac{1}{2}} = &- \frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}}\int_{y_j}^{y_{j+1}} \left[(u\phi)_{i+1} - (u\phi)_i\right] dydt \\ &- \frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}}\int_{x_i}^{x_{i+1}} \left[(v\phi)_{j+1} - (v\phi)_j\right] dxdt \\ &+ \frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}}\int_{x_i}^{x_{i+1}}\int_{y_j}^{y_{j+1}} \phi(\nabla \cdot \boldsymbol{u}) dxdt, \end{aligned} \tag{4.48}$$

where again, Δx is the uniform grid resolution. The only added complication is approximating the third integral. As discussed in [6], Nessyahu and Tadmor's [28] scheme can easily be applied to balance laws like (4.47) by approximating the source term integral using the second order mid-point rule combined with the Mid-Point Euler Method. Thus we approximate the third integral via

$$\begin{aligned} \frac{1}{\Delta x^2}\int_{t^n}^{t^{n+1}}\int_{x_i}^{x_{i+1}}\int_{y_j}^{y_{j+1}} \phi(\nabla\cdot\boldsymbol{u}) dxdt \approx \frac{\Delta t}{4}\Big(&\phi^{n+\frac{1}{2}}_{i,j}\left[\tilde{\partial}_x u_{i,j} + \tilde{\partial}_y v_{i,j}\right] + \phi^{n+\frac{1}{2}}_{i,j+1}\left[\tilde{\partial}_x u_{i,j+1} + \tilde{\partial}_y v_{i,j+1}\right] + \\ &\phi^{n+\frac{1}{2}}_{i+1,j+1}\left[\tilde{\partial}_x u_{i+1,j+1} + \tilde{\partial}_y v_{i+1,j+1}\right] + \phi^{n+\frac{1}{2}}_{i+1,j}\left[\tilde{\partial}_x u_{i+1,j} + \tilde{\partial}_y v_{i+1,j}\right]\Big), \end{aligned} \tag{4.49}$$

where

$$\phi^{n+\frac{1}{2}}_{i,j} = \phi^n_{i,j} - \frac{\Delta t}{2}(u_{i,j}\tilde{\partial}_x\phi^n_{i,j} + v_{i,j}\tilde{\partial}_y\phi^n_{i,j}), \tag{4.50}$$

and Δt is the temporal resolution, and $\tilde{\partial}_*$ is a general numerical derivative. Again, the numerical derivatives of the velocities are calculated via centred differences, while the numerical derivatives of ϕ are calculated using the MC limiter. Thus, using equations (4.49) and (4.50) combined with the scheme described in Section 4.3 we have a non-oscillatory, nearly second order scheme for advecting ϕ according to (4.46).

Correcting $\phi(\boldsymbol{x})$

It must be noted that, as discussed in [33], after the distance function is advected, it will undoubtedly become warped. Therefore after each advection step it should be

corrected. Regardless, the zero level set will remain correct over each advection step so no re-initialization is required and the correction can be easily accomplished by simply calling the same solver, less the initialization step. Nevertheless, numerical experiments with our domains have shown significant numerical diffusion, or smoothing, of the zero level set if the correction is called too often and allowed to iterate to convergence. This results in unwanted volume loss, as the zero level set is used to define where we reconstruct area fraction and ice thickness after each advection step. Considering that sea ice velocities are on the order of 0.1 m/s, it is expected that the distance function will remain relatively unchanged between time-steps and therefore to limit numerical diffusion, we only call the distance function correction every 10 time-steps and limit the correction to 10 iterations.

Additionally, as ϕ is not conserved like the ice thickness and area fraction, it is possible that h and A may go to zero inside the zero level set of ϕ. To account for this, at twice the frequency of the normal distance function correction, we look through the computational domain and check for cells with h or $A = 0$. If any are noted, we reinitialize the value of ϕ at these points to be $-dx/2$ and call the distance function correction, again limited to 10 iterations to limit excessive numerical diffusion. This limit has been deemed acceptable as more often than not, this will occur at points near the boundary. The affects of this general correction limit will be discussed further in Chapter 5.

4.5 Conclusion

It is the combination of these components and the JFNK solver that create an efficient sea ice dynamics solver. Without these components, the JFNK solver would be wasted as we would be unable to produce meaningful simulations. As future work, it would be beneficial to implement and test other methods against those introduced in this chapter. For instance, implementing a higher order ENO and WENO scheme for the advection step would be a worthwhile endeavor, additionally an implicit method for numerical smoothing would be beneficial to circumvent the stability requirements and a higher order distance function solver could help negate the numerical diffusion seen by the first order solver. A discussion on the effects of the implemented methods will be discussed further in Chapter 5.

Chapter 5

Sea Ice Simulation Test Cases

Equipped with the level set method for setting the solver's domain; controlling oscillations through numerical smoothing; the Newton Damped, improved JFNK method for solving (2.1); and the (nearly) second order, non-oscillatory method for (2.13) and (2.14), we have a working solver for viscous-plastic sea ice dynamics. In this chapter, we present two different test cases and discuss the results and convergence to the viscous-plastic solution and the effects of different stopping criteria.

Table 5.1 below contains parameters that, unless otherwise noted, are used for all simulations discussed in this chapter.

Table 5.1: Constant Numerical Parameters for Sea Ice Simulation

Symbol	Definition	Value
dx	Spatial Resolution	20 (km)
dt	Temporal Resolution	1 (hour)
L_x	X-Extent	2000 (km)
L_y	Y-Extent	2000 (km)
T	Final Time	52 (weeks)
ϵ	Jacobian Perturbation	10^{-6}
res_t	Residual Transition	0.625
γ_{ini}	Initial Linear Tolerance	0.99

5.1 The Idealized Domain and Forcing

Like the tests done in Sections 4.4 and 3.3, we set our domain according to the checkerboard pattern shown below in Figure 5.1. Again, this domain was chosen as it allows us to assess the solver's performance at all boundary types, in every direction. Additionally, as will be discussed in Section 5.3.1, this domain affords significant flexibility in terms of domain movement as only half the initial area is covered in ice.

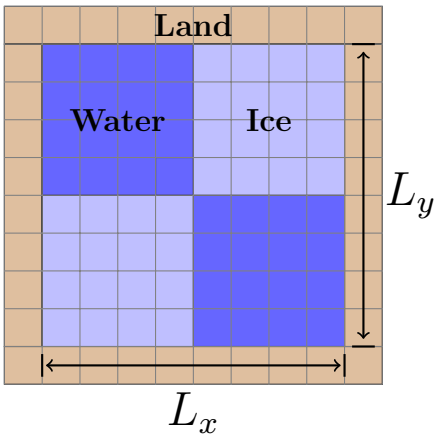


Figure 5.1: The idealized domain

To provide ocean and wind forcing on this domain, we adopt the following idealized forcing fields from [10], devised for similar idealized tests,

$$\begin{aligned} u_{wind} &= 5 + \left[\sin\left(\frac{2\pi t}{\Theta}\right) - 3\right] \sin\left(\frac{2\pi x}{L_x}\right) \sin\left(\frac{\pi y}{L_y}\right) \\ v_{wind} &= 5 + \left[\sin\left(\frac{2\pi t}{\Theta}\right) - 3\right] \sin\left(\frac{2\pi y}{L_y}\right) \sin\left(\frac{\pi x}{L_x}\right) \end{aligned} \tag{5.1}$$

$$\begin{aligned} u_{ocean} &= 0.1(2y - L_y)/L_y \\ v_{ocean} &= -0.1(2x - L_x)/L_x \end{aligned} \tag{5.2}$$

where $\Theta = 4$ days, defines the period of wind forcing. These forcing fields amount to a constant ocean gyre in the middle of the domain, with speeds up to 0.1 m/s, and an oscillating wind field with values ranging between 1 and 9 m/s; Figure 5.2 shows the resulting vector fields at $t = 3$ days.

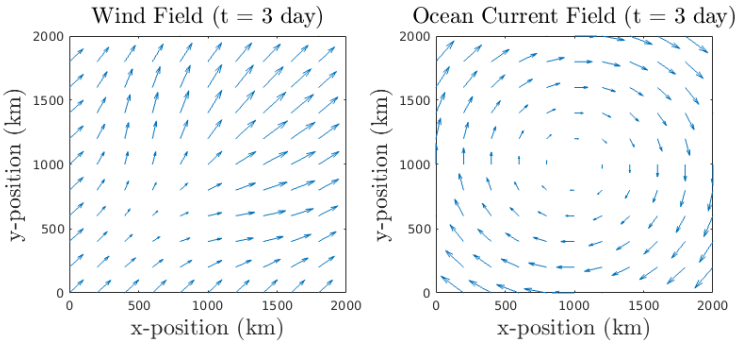
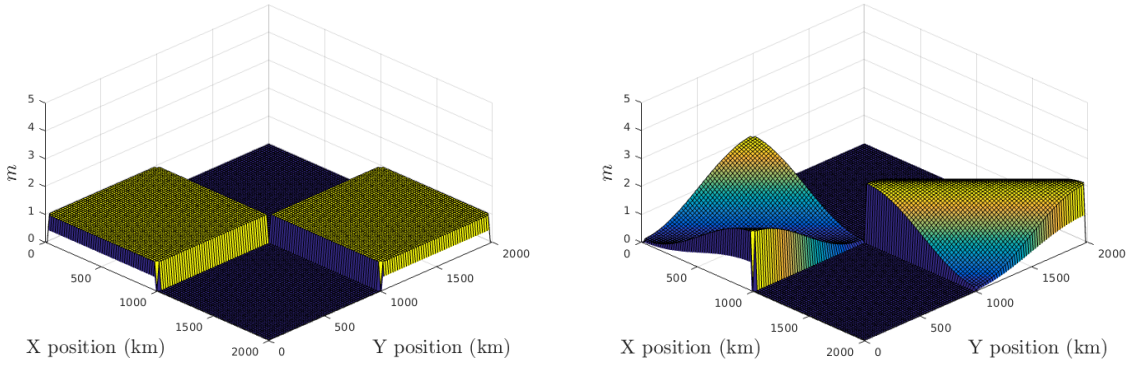


Figure 5.2: The idealized forcing

5.1.1 Initial Conditions

For initial conditions we consider two different cases, referred to as Test Case 1 (TC1) and Test Case 2 (TC2). For both cases, we begin with the ice at rest, i.e. $\boldsymbol{u} = 0$, and assume full area coverage ($A = 1$) where ice is present. For TC1, we use a uniform thickness field of 1 m, as shown in Figure 5.3a, while for TC2 we use the smoothly varying thickness field shown in Figure 5.3b, which ranges between 0.1 and 2.1 m, defined by

$$h(x,y) = \begin{cases} 1.1 + \sin\left(\frac{2\pi}{xL_x}|x^2 - xy + x\frac{L_x}{2}| + \frac{\pi}{2}\right) & \text{if} \quad x \leq \frac{L_x}{2} \\ 1.1 - \sin\left(\frac{2\pi}{xL_x}|x^2 - xy + x\frac{L_x}{2}| + \frac{\pi}{2}\right) & \text{if} \quad x > \frac{L_x}{2}. \end{cases} \tag{5.3}$$



(a) TC1 (b) TC2

Figure 5.3: Initial thickness fields

It should be noted that due to these initial conditions (and those expected in

realistic configurations), there can be a significant discontinuities in the ice strength (P) at the ice terminus. Considering that finite differencing is used to calculate the strength gradients, $\partial_* P$, this adds numerical difficulties. To remedy this and control the strength gradients, we implement a minor smoothing algorithm for the ice strength near the terminus. To illustrate, we provide a one dimensional example; at x_i, if $\phi_i < ndx$ we smooth the strength according to

$$P_i = \frac{\sum_{k=-n}^{n} \tilde{P}_{i+k} |\tanh(\phi_{i+k}/dx)|}{\sum_{k=-n}^{n} |\tanh(\phi_{i+k}/dx)|}, \tag{5.4}$$

where $\tilde{P}$ is the non-smoothed strength value and n is a positive integer, which we set to three. For this one dimensional example, with $h = 1$ m and $A = 1$, (5.4) results in the smoothing shown in Figure 5.4.

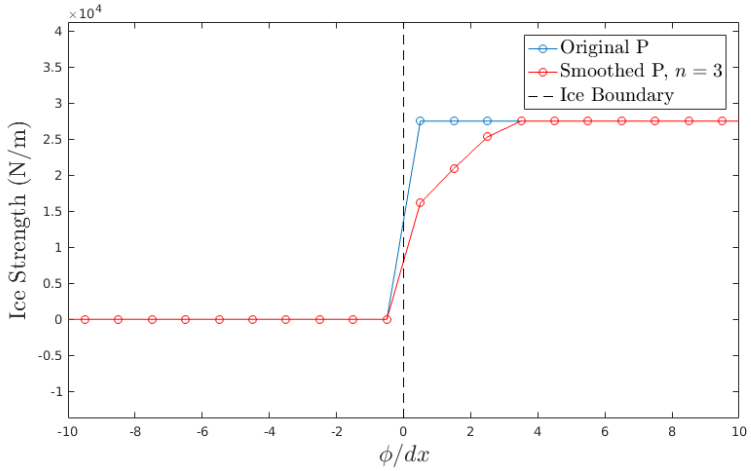


Figure 5.4: One dimensional smoothing example for the ice strength (P)

5.2 Termination Criterion

For termination criterion, we are unable use the discretization error method of Section 3.3, as for these idealized conditions, the solver is unable to reach the same level of convergence, i.e. it is unable to make $||\boldsymbol{F}(\boldsymbol{u})|| \to 0$, which will be referred to as non-linear convergence. The precise reason for this is currently unknown, but it could potentially be attributed to the idealized nature of these experiments. In previous work by others, with similar conditions, the fully explicit EVP method or semi-implicit/explicit VP methods, solving the linearized equations, are used and non-

linear convergence isn't discussed [10, 26]. In work where the non-linear convergence is truly reached and discussed, the solvers were ran on a realistic domain and the equations were given a $5 - 10$ year "spin-up" period [19, 20, 18] that was used to produce realistic initial conditions. As (2.1), (2.13), and (2.14) represent a system of coupled PDE's, it is not surprising the idealized initial conditions described in the previous section don't afford an exact solution. Nevertheless, experiments like this are still valuable to assess the solver's performance and "convergence" is said to have been reached when the residual stagnates. The stopping criteria must reflect this. As is done in [10, 26, 19], the physical nature of the solutions can be used to judge the solver and different convergence properties can be assessed; Section 5.3.3 contains a detailed discussion on this topic.

In all these simulations, we utilize the Newton Damping described in Section 4.1, but in two different forms, Constant Damping and Adaptive Damping. For both these forms, slightly different stopping tests are used.

5.2.1 Constant Damping

For Constant Damping, in (4.1), we use $\tau = 10$, without modification. We set $k_{max} = 200$, and after a minimum number of iterations, k_{min}, we halt the solver at iteration k, if

$$\gamma_{pt} < \frac{||\boldsymbol{F}(\boldsymbol{u}^k)||}{||\boldsymbol{F}(\boldsymbol{u}^{k-1})||} \leq 1, \tag{5.5}$$

where γ_{pt} is referred to as the plateau tolerance, typically set so $0.95 \leq \gamma_{pt} < 1$. This method is referred to as the Plateau Test. In addition to (5.5), we also utilize the Conditional Termination method described in Section 3.3 with a significantly smaller r due to the decreased step size.

5.2.2 Adaptive Damping

For Adaptive Damping, we adopt a similar approach to [17] to increase the robustness of the solver, and allow for increases and decrease in τ. Note, this implementation differs from the method in [17] as their approach allowed only for increases. Using this method, at each non-linear solve, we begin with $\tau = 1$ and at iteration k, if $||\boldsymbol{F}(\boldsymbol{u}^k)|| \geq ||\boldsymbol{F}(\boldsymbol{u}^{k-1})||$, we increase τ, i.e.

$$\tau = \alpha_{inc}\tau, \tag{5.6}$$

where $\alpha_{inc} > 1$, and recalculate $\boldsymbol{u}^k$ and $||\boldsymbol{F}(\boldsymbol{u}^k)||$; if a decrease in the residual is noted, proceed to iteration $k+1$, else increase τ again until a decrease is achieved.

In some cases, it is beneficial to decrease τ and make the step size larger. If a decrease is noted without having to increase τ, the magnitude of the decrease is checked and if a small decrease is observed, i.e.

$$||\boldsymbol{F}(\boldsymbol{u}^k)|| \geq \gamma_\tau ||\boldsymbol{F}(\boldsymbol{u}^{k-1})||,$$

where $\gamma_\tau < 1$, then proceed to iteration $k+1$, but decrease τ according to

$$\tau = \max\Big(1, \frac{\tau}{\alpha_{dec}}\Big), \tag{5.7}$$

where $\alpha_{dec} > 1$, for the next Newton update.

For this method of damping, a maximum iteration count is also imposed, but it is expected that this method will reach it's minimum residual faster and its limit is an order of magnitude less than that imposed for Constant Damping. This is also justified by noting that this method doesn't allow for residual increases, so the solver is unable to jump away from (i.e. overshoot) the solution. This method incorporates Conditional Termination as well, but with a slight modification; if at iteration k, the solver is unable to produce a decrease in residual without $\tau > \tau_{lim}$, the $k-1$ solution is accepted and the solver is terminated.

For a sufficiently small k_{max}, these two stopping conditions are capable of producing acceptable solver performance, but nevertheless, the Plateau Test can also be applied to this method, provided γ_{pt} is close enough to 1.

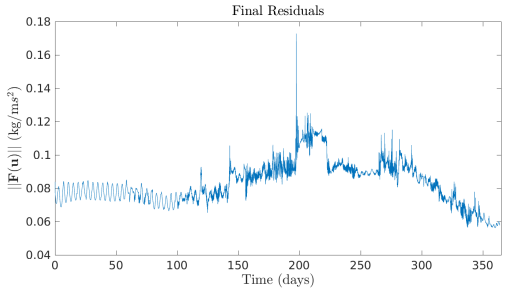
5.3 Results

For our primary simulations, we utilize Constant Damping and the parameters described in Table 5.2; these simulations will be referred to as TC1P and TC2P. Considering that a 52 week simulation amounts to 8736 time steps, it is unrealistic to inspect the solutions at every time level. Thus, for gridded parameters, we limit our inspections to those produced every four days, amounting to 91 inspection points.

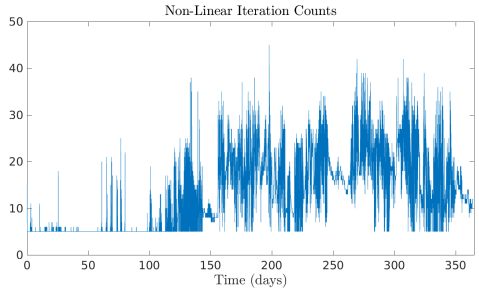
Table 5.2: Numerical Parameters for TC1P and TC2P

Symbol	Definition	Value
γ_{pt}	Plateau Tolerance	0.997
r	Conditional Termination Increase Tolerance	1.1
k_{min}	Minimum Number of Iterations	5

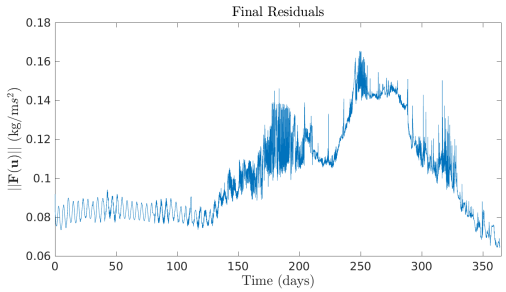
Although we don't see the same level of performance as we saw for the validation simulations, considering the added complications pertaining to the true system, the solver behaved quite well for both TC1P and TC2P. As shown in Figure 5.5, both simulations satisfied the plateau criteria, or were halted via conditional termination, in well under 50 iterations. The average iteration counts were 12.8 with a standard deviation of 8, and 13.7 with a standard deviation of 8.5, for TC1P and TC2P, respectively. Additionally, for the 91 inspected time-steps, the solver consistently produced smooth velocity solutions without any discernible numerical oscillations; as a typical example, Figure 5.6 shows the velocity solutions, in surface format, after the first four days for TC1P.



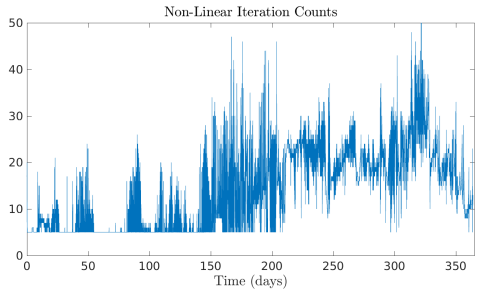
(a) TC1P: Final Non-Linear Residuals



(b) TC1P: Non-Linear Iterations



(c) TC2P: Final Non-Linear Residuals



(d) TC2P Non-Linear Iterations

Figure 5.5: Performance of TC1P and TC2P simulations

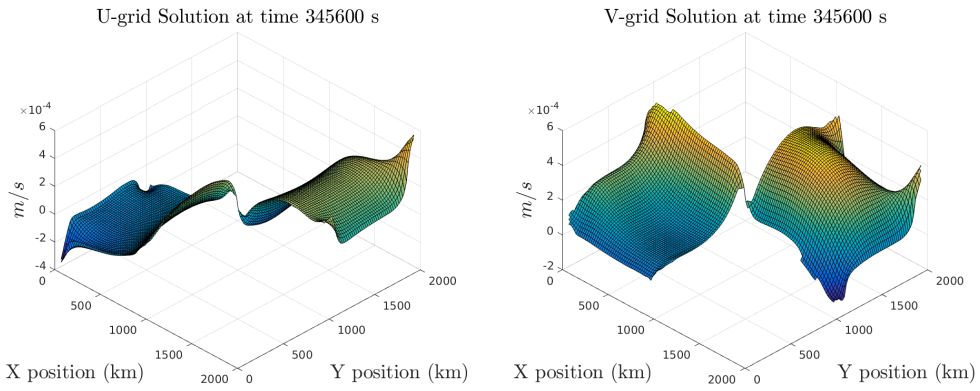
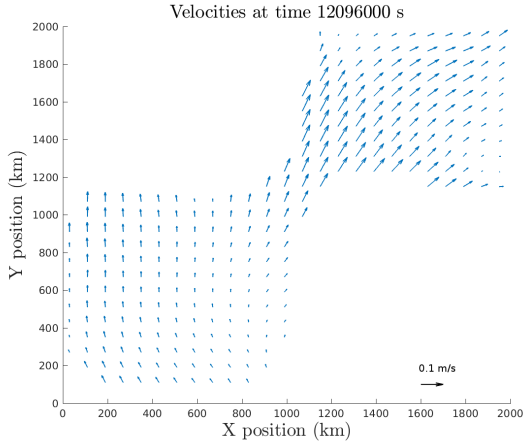


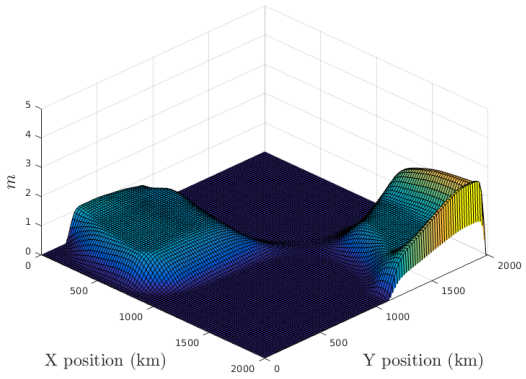
Figure 5.6: Velocity solution example (shown, TC1P, $t = 4$ days)

In addition to performing well numerically, the solver also performed well in producing physically justified solutions. In the first 100 days, the velocities remain quite low, as it takes a lot of work to get massive parcels of ice moving. This initial acceleration period also explains why we see periodic oscillations in the final residuals with a period matching that of the wind forcing. The sea ice velocities only change slightly, causing negligible changes in the residual so forcing oscillations dominate, causing the sinusoidal behaviour seen in Figures 5.5a and 5.5c.

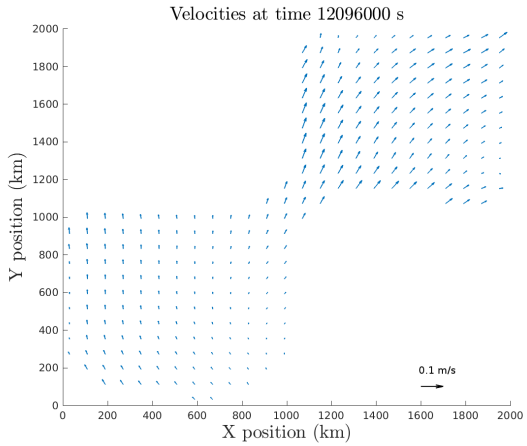
After the acceleration period, when significant movement is observed, we see a strong tendency towards the northeast corner of the domain (Figures 5.7a and 5.7c), consistent with the wind forcing. This trend leads to the build up of ice seen in Figures 5.7b and 5.7d. Additionally, though the ocean field is obviously less important than the wind forcing, we still see its effects in the southwest portion of the domain via a tendency for the ice to flow in a clockwise direction. Note that in Figures 5.7a and 5.7c, we also see the differences between TC1P and TC2P; due to the large mass of ice along the line $y = x$ in the northeast corner for TC2P, it resists northeast motion more than TC1P. Additionally the southwest parcel in TC2P resists the counter clockwise motion to a greater degree due to the large mass on its northern side.



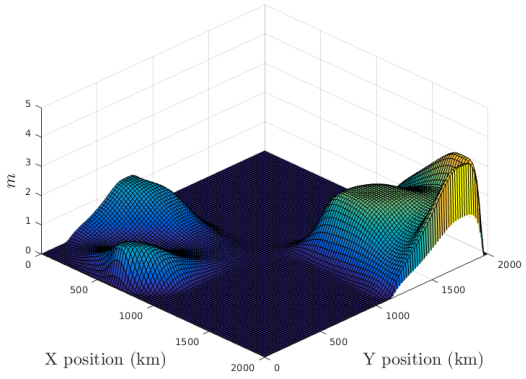
(a) TC1P Velocities



(b) TC1P Thickness



(c) TC2P Velocities



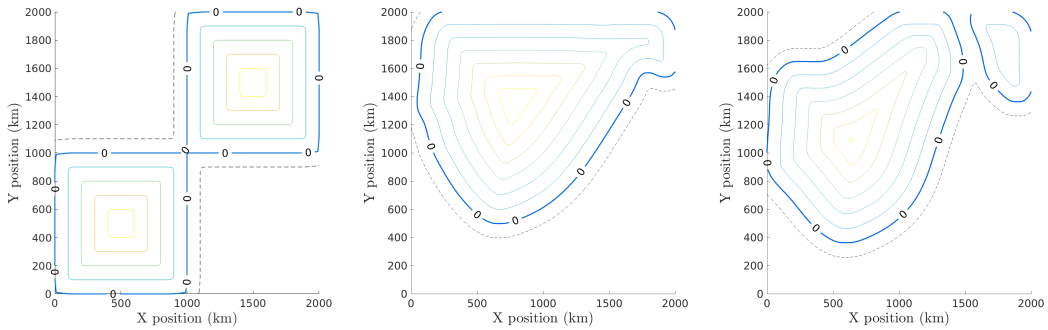
(d) TC2P Thickness

Figure 5.7: TC1P and TC2P at day 140

5.3.1 Domain Movement

As time progresses beyond the initial acceleration phase, the computational domain is significantly modified so we rely on the distance function, $\phi(\boldsymbol{x})$, introduced in Section 4.4, to set the domain and apply boundary conditions. For this to be done properly, assuming that the zero level set is advected correctly, the main requirement is that the gradients near the zero level set don't become significantly warped. To ensure this, we could correct the distance function to the same level of convergence used in the initialization but, as mentioned in Section 4.4, this leads to too much numerical diffusion. Instead, we assume the zero level set and the surrounding gradients only

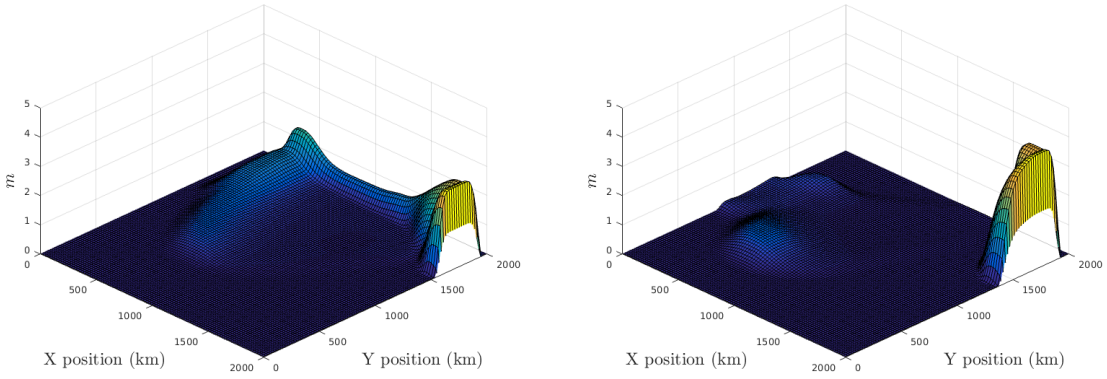
undergo minor changes between time steps and limit how often we call the correction and how many iterations of the algorithm are used. As can be seen in Figure 5.8, this is an acceptable approach as although the ice configuration has changed significantly by the end of the simulations, we still see similar gradients to the initialized $\phi(\boldsymbol{x})$, which is the same for both test cases, after 8000+ time steps.



(a) Initial (b) TC1P Final (c) TC2P Final

Figure 5.8: Evolution of the distance function, $\phi(\boldsymbol{x})$, for TC1P and TC2P

5.3.2 Loss of Volume



(a) TC1P (b) TC2P

Figure 5.9: TC1P and TC2P final thickness fields

Figure 5.9 shows the final thickness fields at the end of the 52 week simulations. In comparison to the initial fields in Figure 5.3, it is obvious that there has been significant loss of volume. This loss has been attributed to the combination of the numerical diffusion caused by our advection schemes for A, h and $\phi(\boldsymbol{x})$, and our

method for correcting $\phi(\boldsymbol{x})$. To assess the magnitude of the loss, we calculated the total volume at each time step for TC1P – Figure 5.10 shows its progression. While the total loss is substantial, it is worth noting that this occurred over 8736 time steps, and that the average loss between time steps is 0.02%, i.e.

$$\frac{V^{n+1}}{V^n} \approx 0.9998,$$

where V^n is the total volume at time level n. As diffusion is expected when solving conservation laws and advection equations, this is not surprising. Nevertheless, over so many time steps the loss becomes substantial ($0.9998^{8736} \approx 0.17$). It is difficult to compare this noted diffusion to others' work as previous idealized tests focus on the treatment of (2.1) [11, 26], and to the author's knowledge, long term simulations tend to have thermodynamics turned on, counteracting the numerical diffusion (i.e. the spin up period in [18]). It is possible that a higher order scheme for correcting $\phi(\boldsymbol{x})$, or different transport schemes, like the incremental remapping scheme in [24], would reduce this volume degradation but exploring this possibility has been left for future work.

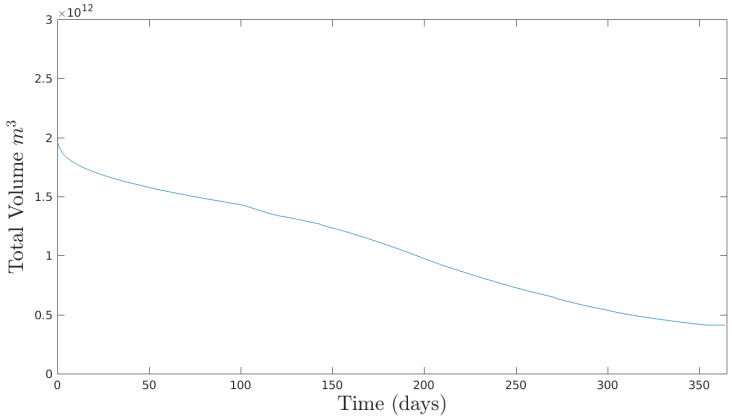


Figure 5.10: TC1P volume progression

5.3.3 Convergence

Although non-linear convergence is the best metric for assessing model accuracy, as discussed in Section 5.2, we are unable to employ this method. Additionally, this is not typically done as the most common methods for solving (2.1) are fully explicit or focus on implicitly solving the linearized VP equations [19]. Therefore, we utilize a method

commonly used in the sea ice community ([19, 10]) and assess model performance on its ability to produce stress states that lie on or inside the VP yield curve (Figure 5.11), expressed as [8]

$$Y(\sigma_1, \sigma_2) = \left(\frac{\sigma_1 + \sigma_2 + P}{P}\right)^2 + \left(\frac{\sigma_2 - \sigma_1}{P}e\right)^2 - 1 = 0, \tag{5.8}$$

where σ_1 and σ_2 are the principal components of the stress tensor, $\boldsymbol{\sigma}$, or its eigenvalues, calculated via

$$\begin{aligned} \sigma_1 &= \frac{\sigma_{11} + \sigma_{22}}{2} + \sqrt{\frac{(\sigma_{11} + \sigma_{22})^2}{4} - (\sigma_{11}\sigma_{22} - \sigma_{12}^2)} \\ \sigma_2 &= \frac{\sigma_{11} + \sigma_{22}}{2} - \sqrt{\frac{(\sigma_{11} + \sigma_{22})^2}{4} - (\sigma_{11}\sigma_{22} - \sigma_{12}^2)}. \end{aligned} \tag{5.9}$$

Note that $\frac{\sigma_{11}+\sigma_{22}}{2}$ represents the average normal stresses, while $\sqrt{\frac{(\sigma_{11}+\sigma_{22})^2}{4} - (\sigma_{11}\sigma_{22} - \sigma_{12}^2)}$ is the maximum shear stress. In plotting stress states on the yield curve, we consider the normalized principal stresses, $\tilde{\sigma}_1 = \frac{\sigma_1}{P}$ and $\tilde{\sigma}_2 = \frac{\sigma_2}{P}$, so the yield curve shown below is normalized for all grid points.

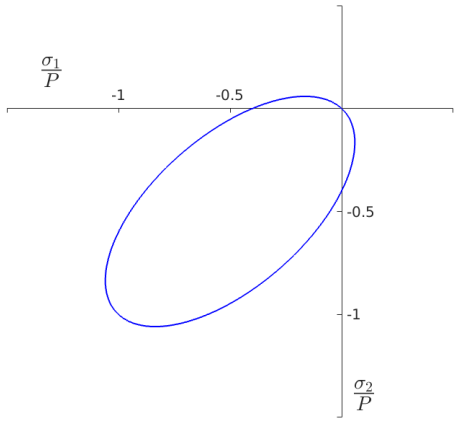


Figure 5.11: Viscous-Plastic elliptical yield curve

For a truly VP solution, referred to as VP convergence, stress states will either lie inside the curve or on it. States inside the yield curve are considered viscous, while those on the curve are plastic. States outside the curve are unrealistic and represent model inaccuracies [19]. Therefore for VP convergence, we hope for σ_1 and σ_2 such that $Y(\sigma_1, \sigma_2) \leq 0$, but in a practical sense, it is expected that there will be some

noise around curve and thus we say a point is a VP stress state if

$$Y(\sigma_1, \sigma_2) \leq \delta, \tag{5.10}$$

where δ is a small number; we follow [19] and set $\delta = 0.005$.

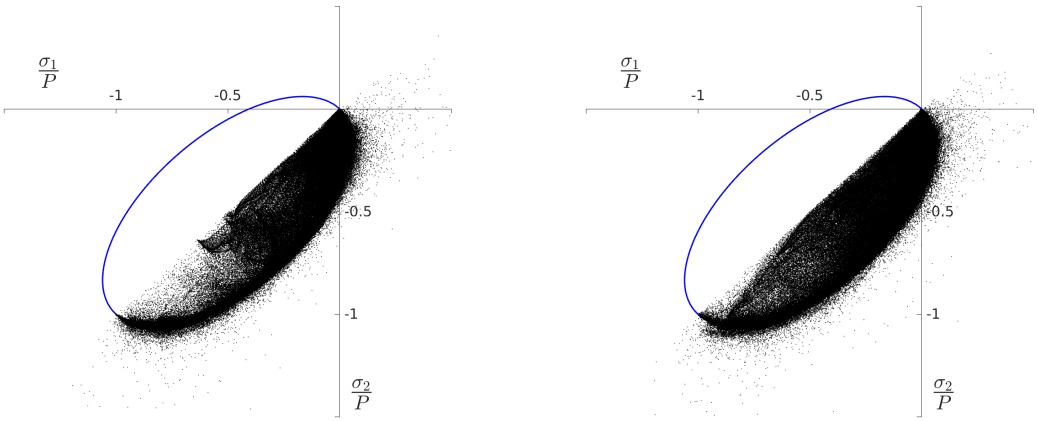
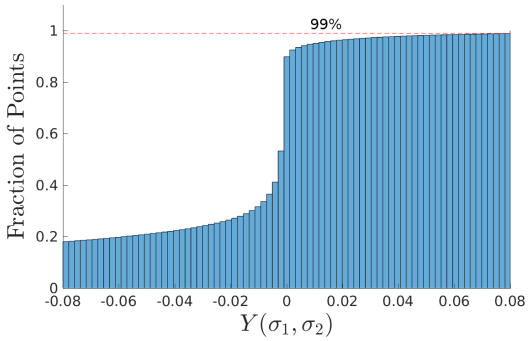
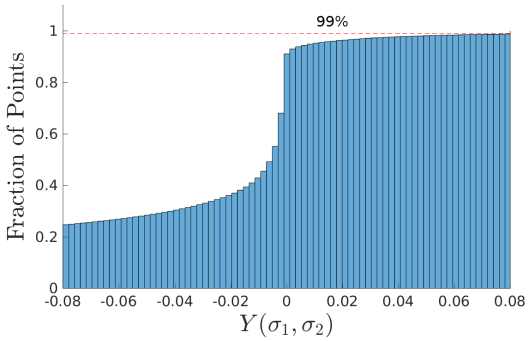


Figure 5.12: TC1P (left) and TC2P (right) principal stresses

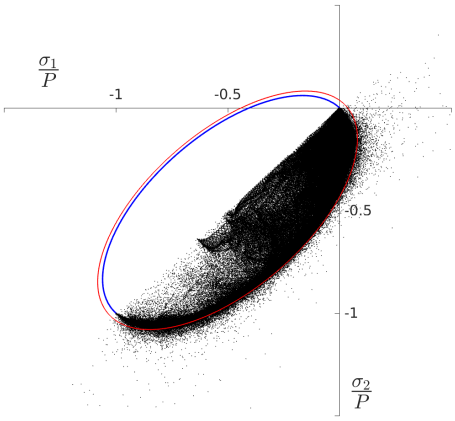
Figure 5.12 shows a scatter plot of the stress states for the 91 measured time steps amounting to 859599 and 828925 points for TC1P and TC2P, respectively. It is worth mentioning that all points falling below the diagonal is a result of the maximum shear stress being a strictly positive value. Of the points considered in this analysis, it was found that 6.6% of points for TC1P, and 6.3% for TC2P, failed to satisfied (5.10). In [19], VP convergence is said to be achieved if 99% of points meet the VP criteria. Although our simulations don't quite make this requirement, they are close. This is made more apparent if we consider the CDF for $Y(\sigma_1, \sigma_2)$ in Figures 5.13a and 5.13b. Although we would have to increase δ to approximately 0.08 to capture 99% of the points, for both TC1P and TC2P, as shown in Figures 5.13c and 5.13d, this represents a small modification to the yield curve. Considering that the average iteration count for both of these simulations was below 15, this is considered an acceptable level of error given the efficiency of the solver. As a comparison, using a GMRES solver to repeatedly solve the linearized equations and updating the non-linear coefficients after each solve, [19] states the need for 40 iterations or Outer Loops (OLs) to produce a fully VP solution.



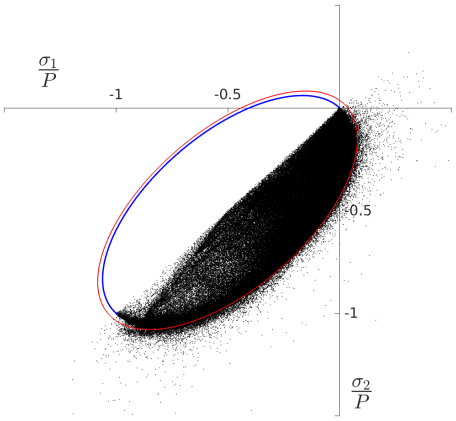
(a) TC1P: CDF of $Y(\sigma_1, \sigma_2)$



(b) TC2P: CDF of $Y(\sigma_1, \sigma_2)$



(c) TC1P: 99% Level Curve (Red)

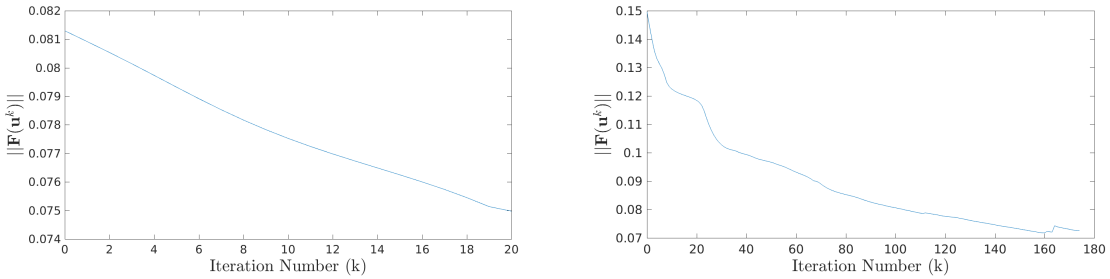


(d) TC2P: 99% Level Curve (Red)

Figure 5.13: Principal stress analysis for TC1P and TC2P

To improve VP convergence, we focus on TC1P, and note that for both test cases, the typical residual progression, shown in Figure 5.14a, suggests that the plateau tolerance is not tight enough as the residual appears to capable of decreasing more. Thus we repeated the same simulation on TC1 with $\gamma_{pt} = 0.999$, referred to as TC1P*. While this definitely made the solver reach more extreme plateaus, it tended to reduce the smoothness with which these plateaus were reached, see Figure 5.14b. Additionally, this simulation took significantly more iterations, with a mean count of 66.5 and a standard deviation of 38.5. This added computational cost came with no additional accuracy as spurious oscillations were added to the velocity solutions and 9.9% of the grid points now failed to satisfy (5.10), suggesting that allowing the Newton Solver to iterate too far is detrimental to solver performance. Unnecessary iterations don't

simply lead to wasted resources, they also lead to decreased accuracies.



(a) $\gamma_{pt} = 0.997$ (b) $\gamma_{pt} = 0.999$

Figure 5.14: TC1 residual progression with Constant Damping

Next, we utilized the Adaptive Damping described in Section 5.2.2, with $k_{max} = 50$, $\alpha_{inc} = \alpha_{dec} = 2$, $\gamma_\tau = 0.9$, $\tau_{lim} = 32$, and the plateau termination turned off; the solver was allowed iterate to k_{max} if able to. This simulation is referred to TC1Ad50. Unfortunately, this simulation performed even worse than TC1P*, resulting in more spurious oscillations in the velocity solutions and 10.2% of points failing to satisfy (5.10). As shown in Figure 5.15, when able to reach k_{max}, the solver reached a plateau well before $k = 50$, therefore we reduced k_{max} to 30 and re-ran the simulation (TC1Ad30). Again, spurious jumps and spikes were noted in some velocity solutions and 10.5% of points didn't satisfy the VP requirement. Although still worse than TC1P, this is considered an improvement on TC1Ad50 as there was no considerable increase in error, but computational resources were saved as the average iteration count dropped from 29.2 (std = 17.7) to 22.0 (std = 10.2). k_{max} was not reduced further to avoid forcing the solver to stop pre-maturely.

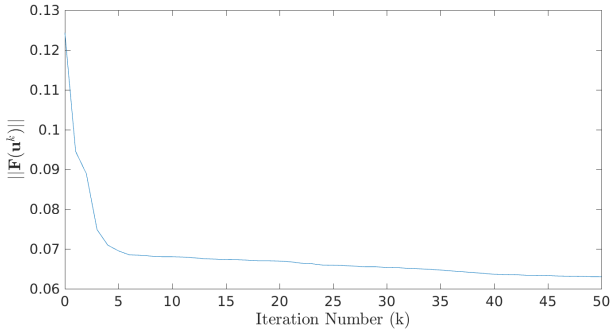


Figure 5.15: TC1Ad50 residual progression with k_{max} iterations

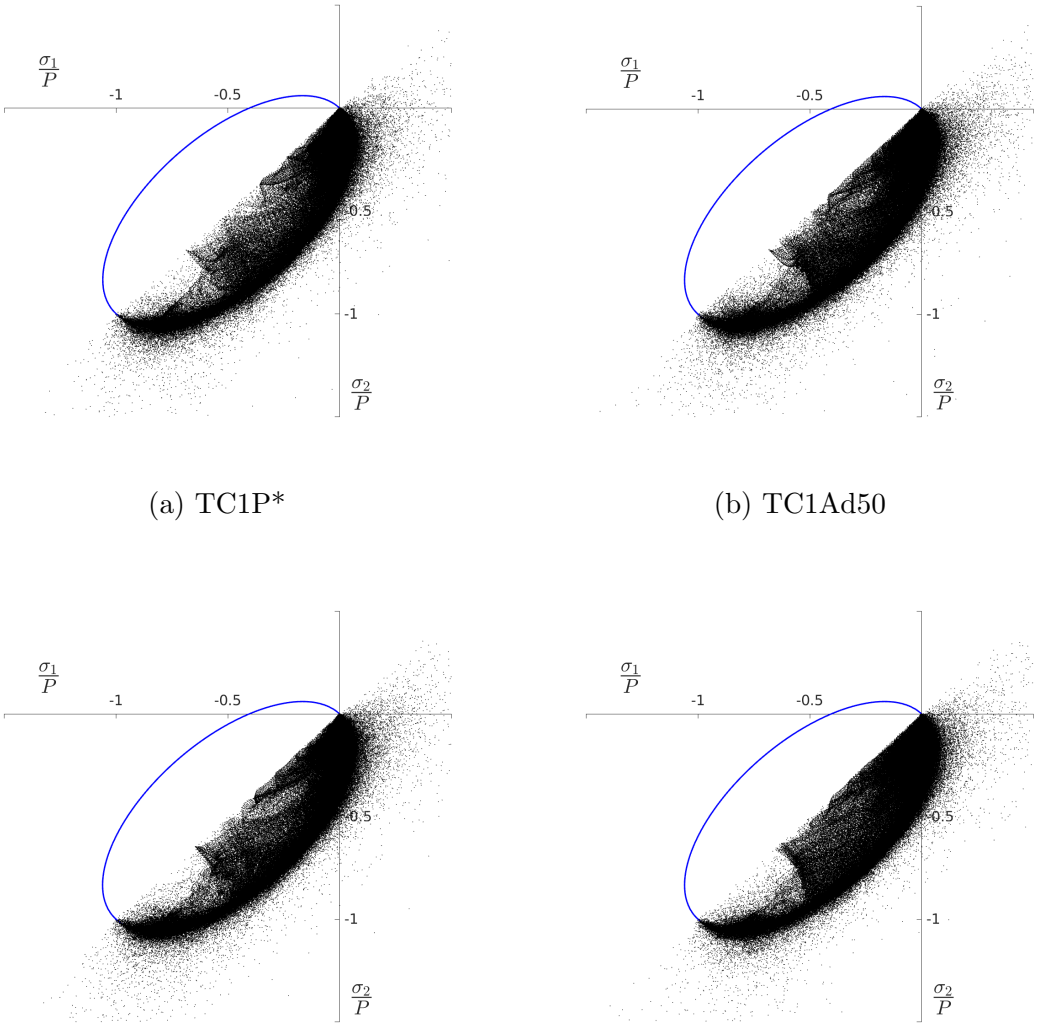
As a final effort to improve convergence, based on the behviour of TC1Ad50 (and

TC1Ad30), we note that a plateau termination criterion may be beneficial to the Adaptive Damping approach. Therefore we ran an additional simulation with $k_{max} = 30$ and plateau termination activated with $\gamma_{pt} = 0.997$, set to mimic TC1P. This simulation is identified as TC1AdPt. While this simulation resulted in a significant reduction in computational effort, with an average iteration count of 9.3 (std = 4.8), no significant improvement in VP convergence was noted as 9.9% failed to satisfy the VP criteria. Nevertheless, this does suggest that the Adaptive Damping approach should be coupled with a termination criteria beyond an iteration maximum and conditional termination. Note that again, spurious oscillations were noted in some velocity solutions.

While none of these additional simulations resulted in improved VP convergence, they still provide insight into the behaviour of the solver; Table 5.3 summarizes the results and Figure 5.16 shows the resulting stress states for each simulation. For Constant Damping, it is clear that over iteration can be detrimental to the solution. This appears to be true for Adaptive Damping as well, but to a lesser extent. By allowing the TC1AdPt to terminate with a more flexible criteria, significant computational effort was saved, compared to TC1Ad30 and TC1Ad50, with little change in accuracy. Additionally, although TC1P produced the best results using Constant Damping, the fact that TC1AdPt resulted in a similar level of convergence to TC1P* using significantly less computational effort suggests the Adaptive Damping approach is a more efficient method but further testing is required to confirm this. Nevertheless, it appears that for both methods, additional thought must be given to the stopping criteria in order to improve performance beyond TC1P.

Table 5.3: Convergence Tests Results

Simulation ID	% Non-VP Stress States	Avg. Iteration Count (std)
TC1P	6.6%	12.8 (8.0)
TC1P*	9.9%	66.5 (38.5)
TC1Ad50	10.2%	29.2 (17.7)
TC1Ad30	10.5%	22.0 (10.2)
TC1AdPt	9.9%	9.3 (4.8)



(a) TC1P*

(b) TC1Ad50

(c) TC1Ad30

(d) TC1AdPt

Figure 5.16: Convergence tests stress states

5.4 Discussion

Overall, for both TC1P and TC2P, the solver behaved well; producing consistently acceptable solutions in an acceptable amount of iterations. For both test cases, the dynamics solver produced physically justified solutions, consistent with the idealized forcing fields, (5.1) and (5.2). Over the course of these simulations, avoiding correcting $\phi(\boldsymbol{x})$ to full convergence at each iteration was shown to be a valid approach, as after 8736 time steps, we still see acceptable gradients in the final version of $\phi(\boldsymbol{x})$. With

these successes aside, significant volume loss was noted for both TC1P and TC2P, but as the average volume loss between time steps was approximately 0.02%, this is not surprising; the shear number of time steps considered make this loss seem more drastic than it really is. That being said, additional work should be done to reduce this degradation, which may be accomplished with a higher order method for correcting $\phi(\boldsymbol{x})$ or through different schemes for transporting A, h, and $\phi(\boldsymbol{x})$.

In assessing the convergence of this solver, while non-linear convergence is the best metric [19], we were forced to consider a different method and utilized VP convergence, a method commonly used in the sea ice community [19, 10, 35]. For both TC1P and TC2P, the solver produced acceptable results while maintaining numerical efficiency, nevertheless there is still room for improvement. Our additional tests were unable to produce better VP convergence than TC1P, but valuable insight into the solver behaviour was still gained. For both Constant Damping and Adaptive Damping, it was shown that increased iteration counts don't automatically result in improved convergence; in fact, for Constant Damping, over iteration was shown to be quite detrimental to the solution. Although we achieved the best results with Constant Damping, the potential computational gains via Adaptive Damping can't be ignored, but our tests suggest that careful thought must be used in devising a proper termination criterion. This appears to be the case for both damping methods as there is significant sensitivity to the termination method in both computational efficiency and VP convergence.

It should be noted that in all simulations besides TC1P and TC2P, spurious oscillations were sometimes noted in the velocity solutions. Preliminary tests were conducted with larger values of ν in (4.2), for more aggressive smoothing to control these oscillations, but this was ineffective and in some cases additional errors were noted. This has been attributed to the fact that at $\nu = 1\times10^{12}\ m^4/s$ it is already quite close to its stability limit for the numerical scheme. As discussed in Section 4.2.1, as ν approaches the stability limit for (4.14), the scheme becomes less effective at damping it's own errors. Additionally, it should be noted that due to the formulation of the elliptical yield curve, Gray and Killworth have shown that the VP equations, as used here, can be linearly unstable [7]. This instability is not a numerical construct, but a quality of the equations themselves. Gray and Killworth attribute it to the fact that the yield curve (Figure 5.11), by crossing into the second and fourth quadrants, allows for tensile stresses in divergence. However, they also argue that the equations are non-linearly stable due to the vanishing of the ice strength P in divergent flow, which is

the key driver of the instability. Nevertheless, it is warned that when using numerical schemes to solve the aforementioned equations, these instabilities may excite grid scale waves and pollute the numerical solution, particularly at fine grid scales. As a method to counter these inherent problems, Gray and Killworth suggest the use of numerical diffusion as a method to stabilize the equations [7]; further motivating us to implement more aggressive smoothing through an implicit method. That being said, it is hard to say if the noted oscillations in our convergence tests are the result of these instabilities, but it is a possibility. It is also feasible that they are the result of the undamped oscillation described in [17], caused by the Crank Nicolson treatment of the ocean forcing in regions of low concentration (A), where the external forcing dominates the rheology term. Nevertheless, this is not expected as our observed oscillations were not specific to regions with low A. Investigating the true cause of these oscillations has been left as future work, but it is the author's expectation that they are linked to the need for a better convergence criterion of the Newton solver as the solutions produced in TC1P and TC2P were consistently smooth.

Chapter 6

Conclusion

Sea ice dynamics play a vital role in the evolution of ice cover and thickness in Earth's polar regions, which in turn influences the exchange of heat, moisture, and momentum between the atmosphere and the underlying ocean [18]. For this reason and other climatic feedback effects, the ability for climate models to accurately project the future of sea ice is crucial for accurate projections of the global climate [15]. To simulate the necessary dynamics and produce the resulting thickness and concentration fields, it is crucial that that we capture ice drift, which hinges on the formulation of the rheology [14], relating the applied stresses to resulting deformations. Over recent years, the most widely used models for sea ice rheology has been the Viscous-Plastic (VP) model introduced by Hibler [9], or the Elastic-Viscous-Plastic (EVP) model introduced by Hunke et al [11], which is a numerically motivated modification to Hibler's original model. Taking the VP equations as they stand leads to considerable numerical difficulties and a Sea Ice Momentum Equation (SIME) that is inherently difficult to solve. These difficulties stem from the highly non-linear nature of the resulting equations, which are related to the enormous range of effective viscosities in the model [10]. These non-linearities result in massive changes in internal stresses when going from a slightly convergent flow to a slightly diverging one [17]. To apply an explicit numerical scheme to these equations, stability analyses (by others) have shown that a time step on the order of second would be required on a 100 km grid, or 100th of a second for a 10 km grid [18]. These requirements have motivated modelers to typically use implicit, iterative methods. In the past, many of these VP solvers use a Picard like method, where the linearized equations are implicitly solved and then the non-linear coefficients are updated via an outer-loop (OL) iteration; this process is typically repeated a fixed number of times before moving on to the

next time step, without consideration for true non-linear convergence [18]. Lemieux and Tremblay [19] have shown that this approach can result in significant errors and that these "Picard Solvers" converge very slowly to the non-linear solution. To improve convergence, Lemieux et al. recently introduced a Jacobian-Free Newton-Krylov (JFNK) solver for the SIME, converging 3 to 7 times faster than the typical Picard Solver [20, 18], which opened up new opportunities to apply advanced non-linear solvers to the SIME.

In this thesis we build on this work. Early implementations of the JFNK solver attained first order accuracy in time using a backward Euler approach, but in Section 3.2.2 we present a *Crank Nicolson* approach to the solver, achieving fully second order accuracy. As another proposed improvement, we go beyond the simple first order approximation for the Jacobian matrix in the Newton iterations, and formulate its linear and nearly linear components in closed form and use a second order approximation for the remaining terms. In addition to this, in Section 3.1 we present a novel treatment of the rheology term that avoids the need to set boundary conditions on the viscosities. To finish Chapter 3, we perform extensive numerical tests on our improved JFNK solver, validating it and confirming second order convergence on a set of synthetic equations. In Chapter 4, we present necessary additional components for our dynamics solver: Newton Damping; numerical smoothing with hyper-viscosity; a non-oscillatory, (nearly) second order scheme for transporting the ice; and finally our method for masking the domain via the distance function. In Chapter 5, we combine the aforementioned components and present two idealized test simulations, both of which performed well. We conclude the chapter with a discussion on the solvers ability to converge to a truly VP solution.

In Section 3.3, we validate second order convergence through grid refinement tests which also verified that our JFNK solver, written in FORTRAN 90, was functioning as desired. In addition to this surface level validation, these experiments provided valuable insight into the solver. For the moderate grid, we saw that the solver is very efficient, as over 1,008 time steps, the maximum non-linear iteration count for convergence was 11! With that said, we also saw that the solver does suffer from some convergence problems, particularly for the coarse grid, where the solver was occasionally unable to meet our formal convergence requirement in a reasonable number of iterations. We additionally saw a complete breakdown of the solver with the fine grid after the six day mark. Nevertheless, as shown in Section 3.3.4, formal convergence is not always required and we can use Conditional Termination to halt the solver early,

i.e. at iteration k, if we see a significant increase in residual when progressing to $k+1$, we can halt the solver, accepting $\boldsymbol{u}^k$ as the solution. As shown through our tests, this approach doesn't cause a significant change in accuracy, yet saves considerable computational resources when the solver has troubles converging. It should be noted, that while we see that formal convergence, according to our criterion (3.31), isn't required, these noted problems with the coarse and fine grids suggest that our proposed formula for convergence is far from universal and that additional consideration on this topic is required.

In Section 3.2.2, we show that, utilizing the funtional form of the non-linear residual, $\boldsymbol{F}(\boldsymbol{u})$, i.e.

$$\boldsymbol{F}(\boldsymbol{u}) = A(\boldsymbol{u})\boldsymbol{u} - \boldsymbol{b}(\boldsymbol{u}),$$

we can create a better approximation to the Gateaux derivative $J(\boldsymbol{u})\boldsymbol{v}$, than the approach used by Lemieux et al [20, 18, 17, 27], where

$$J(\boldsymbol{u})\boldsymbol{v} \approx \frac{\boldsymbol{F}(\boldsymbol{u}+\epsilon\boldsymbol{v}) - \boldsymbol{F}(\boldsymbol{u})}{\epsilon}$$

is used. Our approach can be thought of as a *Partially Jacobian-Free* (PJF) method, as we formulate the action of the Jacobian in closed form for the linear and nearly linear terms, and utilize a second order approximation for the remaining terms, producing (3.28). Additional work must be done to assess the full affects of the PJF method, but in Section 3.3.3 we see that the second order approach to (3.28) is drastically superior to its first order counter-part. This is in sharp contrast to what is reported in [20] as it is stated that for the fully Jacobian-Free method, a second order approximation results in a negligible change in accuracy, while for our tests we see that the second order approach shows significantly better convergence behaviour. The reason for this discrepancy is currently unknown, but it could be the result of our PJF method; additional investigation is warranted. We also see in Section 3.3.5 that the second order PJF method is not sensitive to the chosen value of ϵ, while additional tests on the first order approach suggests significant sensitivity; this agrees precisely with later work by Lemieux et al [27, 17], where a distinct sensitivity to ϵ for their first order approach is noted.

After presenting the additional components of our dynamics solver in Chapter 4, we perform idealized sea ice simulations in Chapter 5, using the domain, idealized forcing, and initial conditions laid out in Section 5.1. We refer to these simulations as

TC1P and TC2P, depending which initial condition was used. For both TC1P and TC2P, we utilized Constant Damping combined with Conditional Termination and the Plateau Test, as laid out in Section 5.2, with the parameters presented in Table 5.2. Both of these simulations performed quite well over the year long simulation with a time step of 1 hour (or 8736 time steps), producing consistently acceptable, physically justifiable solutions in an acceptable amount of iterations. The iteration counts ranged quite significantly between 5 and 50, with averages of 12.8 and 13.7 and standard deviations of 8 and 8.5, for TC1P and TC2P, respectively. While this is a notably larger spread than that seen in the validation simulations, given the added complications associated with the true system, this is not surprising. These simulations also confirmed that using the distance function is a suitable method for limiting the computational domain and that our assumptions that neither a correction after every advection step nor full convergence to (4.37) is required, were valid. Nevertheless, as seen in Section 5.3.2, due to the large number of time steps, a significant loss of volume was noted for both TC1P and TC2P. As the average loss between time steps was 0.02%, this is not surprising as numerical diffusion is expected when solving (2.13) and (2.14), but it is possible that the use of $\phi(\boldsymbol{x})$ to limit the domain added to this; it is worth testing if higher order schemes for solving the distance function would result in less volume degradation. Undoubtedly, this volume loss can also be (at least) partially attributed to our transport scheme, laid out in Section 4.3. It is feasible that a different method, like a higher order ENO/WENO method or the incremental mapping scheme in [24] would reduce this observed degradation, but this has been left as future work.

In Section 5.3.3, we conclude Chapter 5 with a detailed discussion on the convergence of our dynamics solver. Lemieux and Tremblay [19] have shown that non-linear convergence is the best metric for assessing solver accuracy, as using other methods common to the sea ice community, like convergence to VP stress states (VP convergence) or the average kinetic energy of the ice pack (KE convergence) can leave notable errors. Nevertheless, as discussed in Section 5.2, we are unable to achieve full non-linear convergence and another method is required. In [19] it is shown that assessing VP convergence is a better approach than KE convergence and thus we adopt this method. Using this approach, we find that for $\sim 850,000$ measured stress states for both TC1P and TC2P, 93.4% and 93.7%, respectively, fall within or on the VP yield curve. Lemieux and Tremblay [19] accept a solution as a VP solution if 99% of the stress states satisfy the yield curve, and state that using a Picard type solver,

as laid out in [18], 40 iterations are typically required to achieve VP convergence. While we don't see this level of VP convergence, considering that the average iteration counts were ~ 13 for both simulations, and that 99% of the states are relatively close to the yield curve (as seen in Figure 5.13), we consider this as acceptable VP convergence achieved in an efficient number of iterations. To improve VP convergence for Test Case 1, we first decreased the tolerance of the Constant Damping approach by setting $\gamma_{pt} = 0.999$, and then implemented the Adaptive Damping described in Section 5.2.2, with and without the Plateau Test (5.5). As seen in Table 5.3, none of these attempts led to improved convergence, but these tests did make it apparent that added iterations aren't automatically translated into added convergence. This is particularly true for the Constant Damping approach, where over iteration proves quite detrimental to the solution. Additionally, while Constant Damping produced the best results, the potential computational gains from Adaptive Damping can't be ignored; TC1AdPt produced the same level of convergence as TC1P* with nearly an order of magnitude reduction in average iteration counts. Additional work must be done in this regard, but it appears that a more robust stopping criterion is required for the Adaptive Damping approach in order to reap its computational benefits. This is also true for the Constant Damping approach, as its convergence is quite sensitive to the stopping method used.

As noted at the end of Section 5.4, all attempts at improving convergence resulted in simulations with (occasionally) observed discontinuities (spikes, oscillations, and jumps) in the velocity solutions. We attempted to control these oscillations by increasing the hyper-viscosity parameter ν, but due to the stability requirements on (4.14), this was unsuccessful; to achieve this, an implicit smoothing scheme is likely required. However, we currently see three possible options for these oscillations. First, in work by Gray and Killworth [7], it is shown that due to the formulation of the VP model, the resulting equations are actually linearly unstable, as tensile stresses can be felt in divergence, pulling the ice apart further. However it is also stated that the equations are non-linearly stable as the ice strength P is the driving factor behind this instability and when it occurs, the ice thickness and area coverage decrease, so $P \to 0$, and thus the instability dies off. Nevertheless, Gray and Killworth warn that the linear instabilities may excite grid scale waves, polluting the numerical solution. Second, in later work by Lemieux et al [17], it is stated that a Crank Nicolson approach was attempted for the JFNK solver, but undamped oscillations were noted in regions of low concentration where external forcing dominates the rheology term.

The oscillations were credited to the Crank Nicolson treatment of the forcing terms. Third, it is possible that these oscillations can be credited to our seemingly poor stopping condition on the Newton iterations. It is the author's expectation that this is the likely cause as no obvious oscillations were noted in the solutions for TC1P or TC2P, while altering the stopping criterion for TC1P* led to notable fluctuations in the solutions. It is possible that they can be attributed to the instabilities noted by Gray and Killworth, or the undamped oscillations noted by Lemieux et al, but additional work is required on this. That being said, it is expected that our noted oscillations aren't attributed to those mentioned in [17] as they aren't limited to regions with low A.

It is worth stating that in Lemieux et al's implementations of the JFNK solver using a backward Euler approach [20, 18], there is no mention of a need for numerical smoothing and the solver appears to be closer to reaching the theoretical, quadratic convergence rate of Newton solvers. It is possible that this apparent better behaviour could be attributed to the linear instability problem discussed above and careful investigation is warranted to clarify this. While the Crank Nicolson scheme should result in a more accurate representation of the true equations, it is known to exhibit a marginal A-stability region [23], which may be unable to damp the unstable modes inherent to the VP model. In contrast, the backward Euler method is known to have an A-stability region that extends far into the right portion of the complex plane, making it L-stable. In a practical sense, this difference in A-stability regions implies that the Crank Nicolson scheme is incapable of damping growing modes, while the backward Euler can. At coarser grid scales, the added damping of the backward Euler method may suffice to hide these instabilities, but as stated by Gray and Killworth [7], the instability will become more apparent as the grid scale is refined and thus it is expected that the backward Euler method will become incapable of controlling the instability at finer grid scales. A detailed and robust comparison between these two methods would shed light on these differences and provide valuable insight into this instability associated with the VP model. It is the author's opinion that instead of relying a less accurate, more diffusive scheme to control the inherent small scale instability of this model, additional work should be conducted to create a fully second order discretization with better damping properties, e.g. a backward differentiation formula (BDF) scheme [23], or as suggested in [7], use a modified yield curve that doesn't exhibit this linear instability. Nevertheless, in this work we show that numerical smoothing can be implemented as a working solution to

these problems, and allowed our solver to produce acceptable solutions in appropriate computational time.

So in conclusion, overall our improved, second order JFNK method proved to be a successful method for solving the SIME, performing quite efficiently and accurately for the tests laid out in Chapters 3 and 5. Combined with the additional components in Chapter 4, we produced a fast and efficient solver for viscous-plastic sea ice dynamics. While using a fully second order discretization of the SIME is beneficial, it is expected that the most significant improvement of our JFNK implementation is the second order, *Partially Jacobian-Free* approach to the Gateaux derivative in the Newton iterations, as the second order approach was shown to be a drastic improvement to the first order counter-part. Additional work must be done to assess the full effect of partially forming the Jacobian. It is also worth noting that, as our treatment of the rheology term fully negates the need to set boundary conditions on diagnostic variables, it should be considered in future versions of SIME solvers. It is expected that its use leads to a better behaved matrix, or at least an easier to analyze matrix, as viscosities from multiple surrounding grid points are no longer present in each row of the matrix $A(\boldsymbol{u})$, but again, future work is needed. We must also assess the effects of the use of the distance function to set the computational domain. As it no longer involves the tracking of a strictly positive scalar, it should be a better method those that rely on h and A, but as simply solving the distance function can lead to diffusion on the zero level set, i.e. the boundary, higher order methods should be tested. Additionally, the apparent added sub-grid resolution near boundaries should be assessed. The last quality of our implementation that we'd like to highlight is the Adaptive Damping approach described in Section 5.2.2; a similar implementation was recently used in Lemieux et al's later work on the JFNK solver [17], but it didn't allow the step size to increase again after it was decreased. It is felt that with the derivation of a robust stopping criterion, that this method can result in accurate solutions with significant computational gains.

To highlight some additional future endeavors to those mentioned above, a more robust comparison must be done with the work conducted by Lemieux et al, particularly with the recent second order JFNK solver in [17], which achieved second order accuracy in time through second order backward differences. This study would be quite extensive as in this implementation Lemieux et al also modifies their approach to avoid the standard time splitting method used in most sea ice models. Nevertheless, comparisons to [20, 18], in addition to probing the differences between the

backward Euler and Crank Nicolson methods, could also prove valuable in assessing the differences caused by our Jacobian approximation, particularly as to why we see such different effects from the second order approach. It is also worth noting, that in all the implementations of the JFNK solver for the SIME [20, 18, 27, 17], a preconditioning operation is utilized to improve convergence of the linear system, but the full affects of this aren't documented. In our application we completely avoid a preconditioning step, which may suggest that it isn't required; this could explain the noted differences in the solvers' reactions to the second order approach to the Gateaux derivative. That being said, it must be stated that a detailed analysis of the number of linear iterations required for each Newton step was not conducted, and additional testing must be done on our model to assess what affects a preconditioning step would have. Nevertheless, this is considered a very valuable undertaking as in the one parallel application [27], it is stated that the preconditioning operation is the most expensive routine for the JFNK code.

Appendix A

Validation Simulation Plots

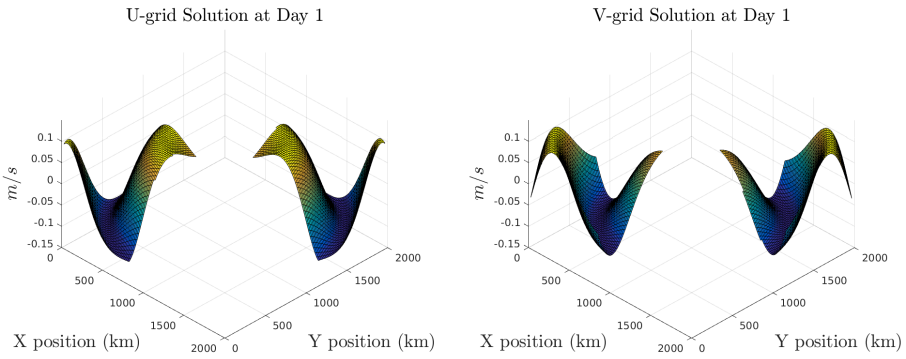


Figure A.1: Validation Simulation: Solution at Day 1

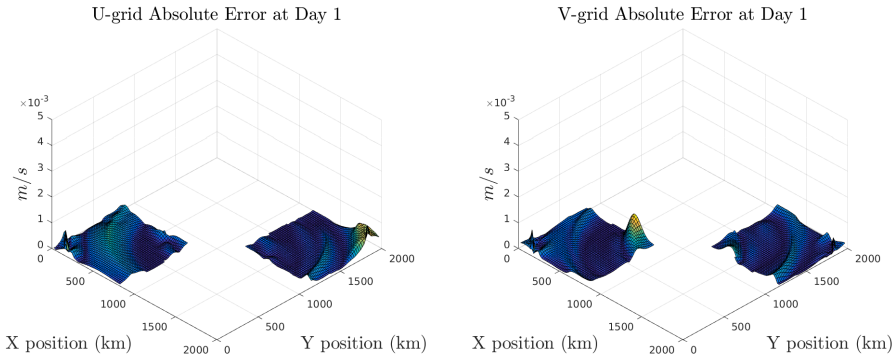


Figure A.2: Validation Simulation: Absolute Error at Day 1

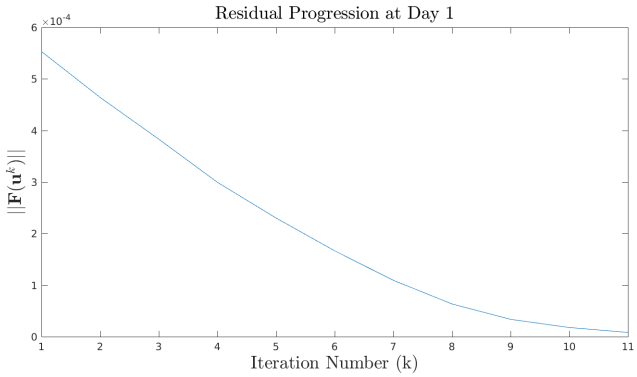


Figure A.3: Validation Simulation: Residual Progression at Day 1 Solve

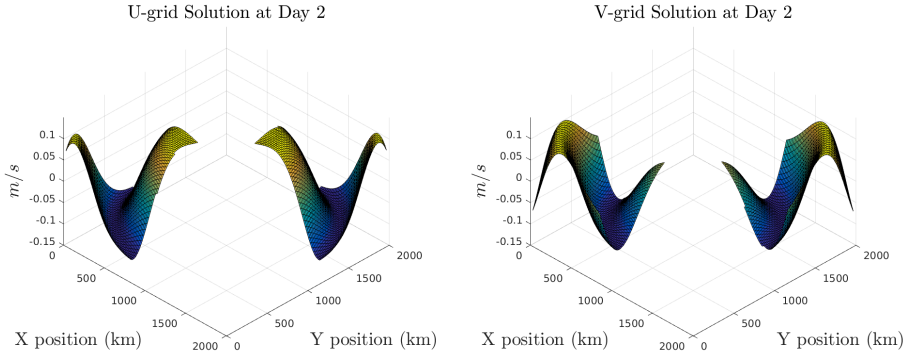


Figure A.4: Validation Simulation: Solution at Day 2

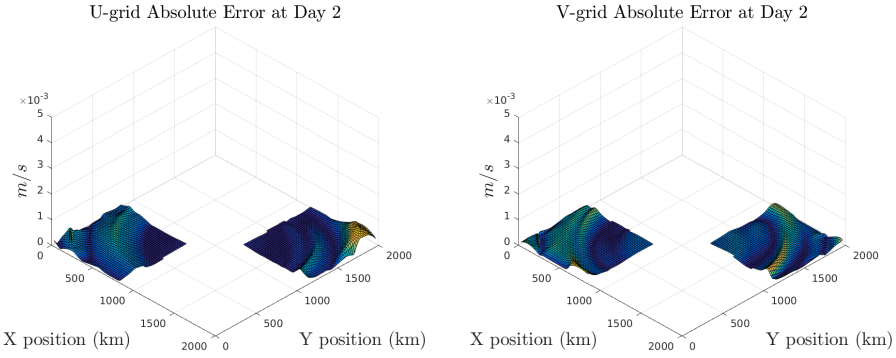


Figure A.5: Validation Simulation: Absolute Error at Day 2

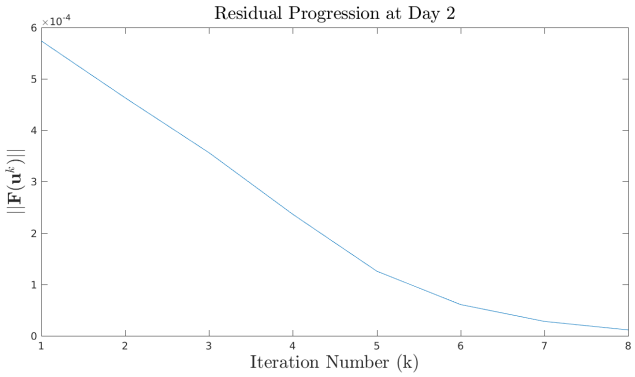


Figure A.6: Validation Simulation: Residual Progression at Day 2 Solve

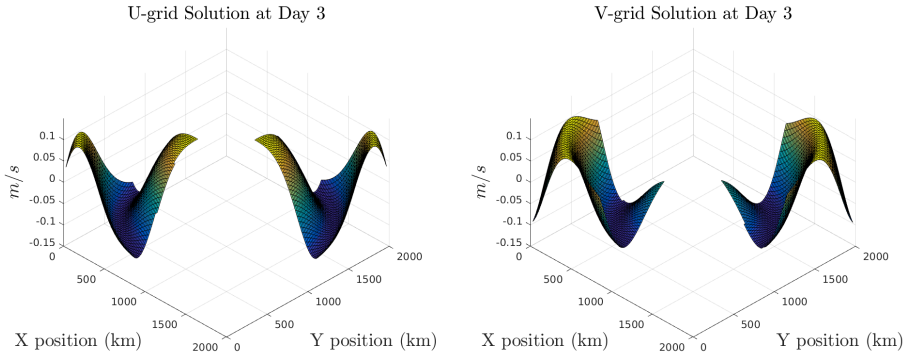


Figure A.7: Validation Simulation: Solution at Day 3

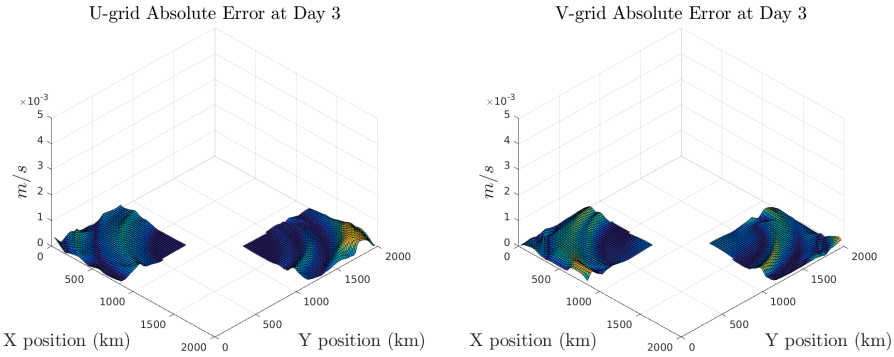


Figure A.8: Validation Simulation: Absolute Error at Day 3

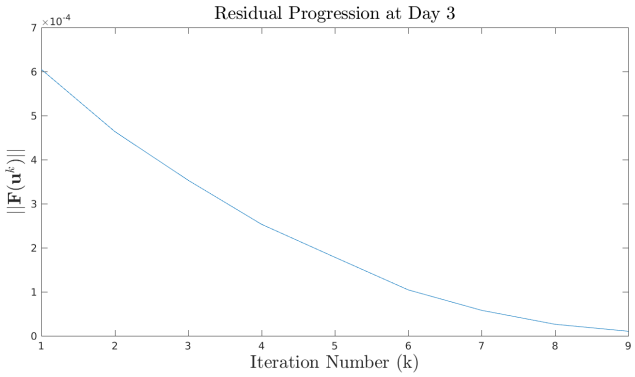


Figure A.9: Validation Simulation: Residual Progression at Day 3 Solve

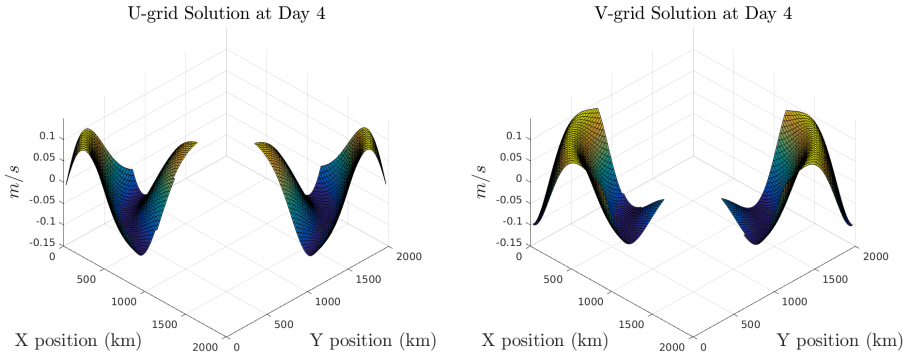


Figure A.10: Validation Simulation: Solution at Day 4

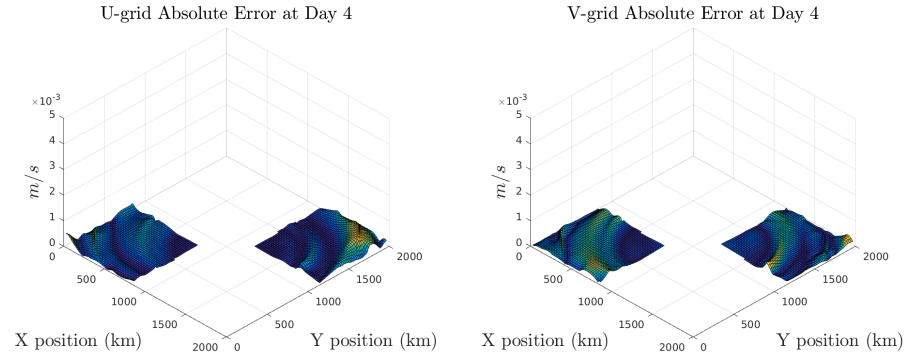


Figure A.11: Validation Simulation: Absolute Error at Day 4

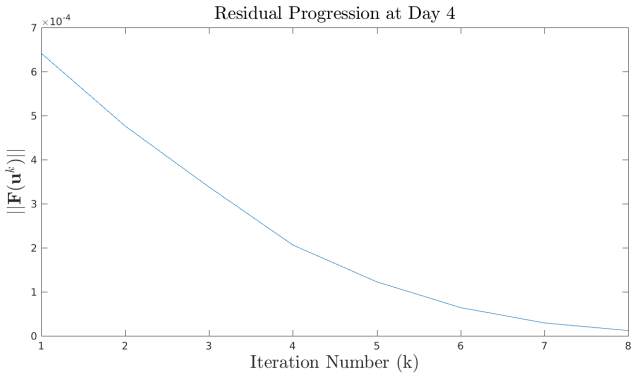


Figure A.12: Validation Simulation: Residual Progression at Day 4 Solve

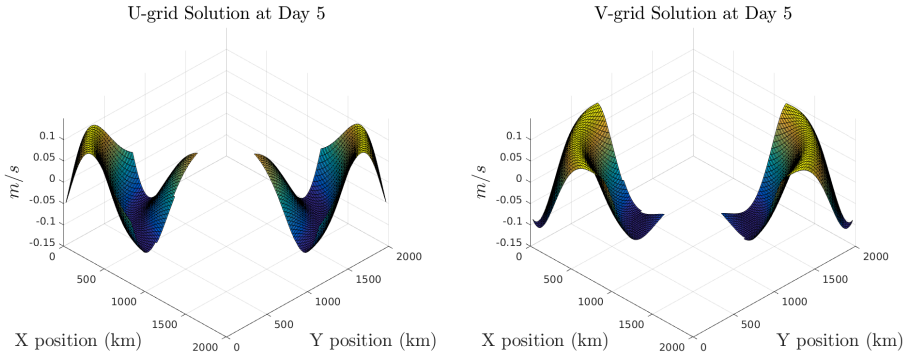


Figure A.13: Validation Simulation: Solution at Day 5

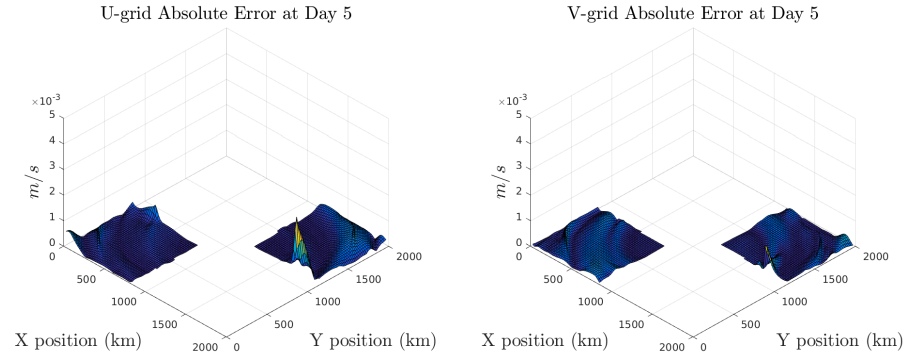


Figure A.14: Validation Simulation: Absolute Error at Day 5

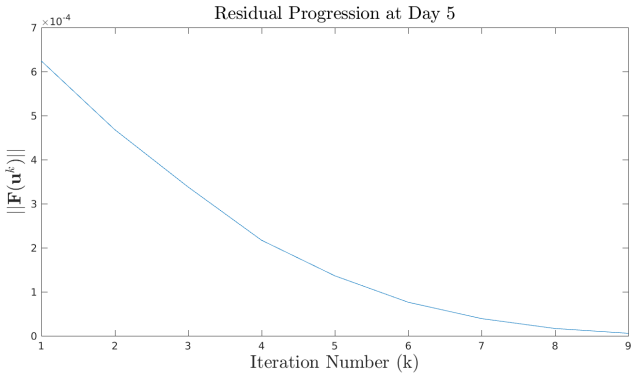


Figure A.15: Validation Simulation: Residual Progression at Day 5 Solve

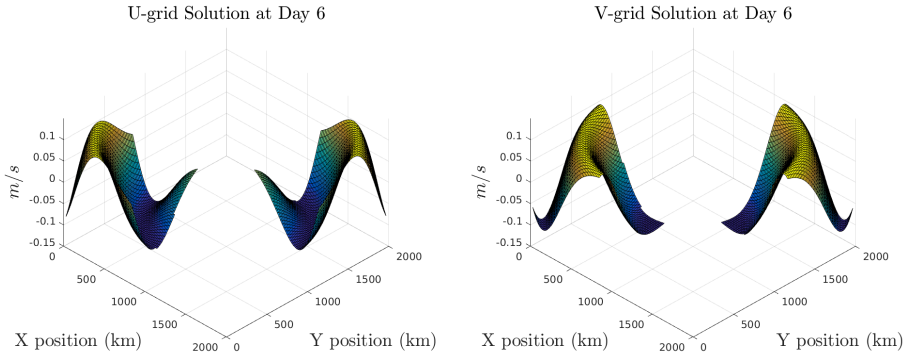


Figure A.16: Validation Simulation: Solution at Day 6

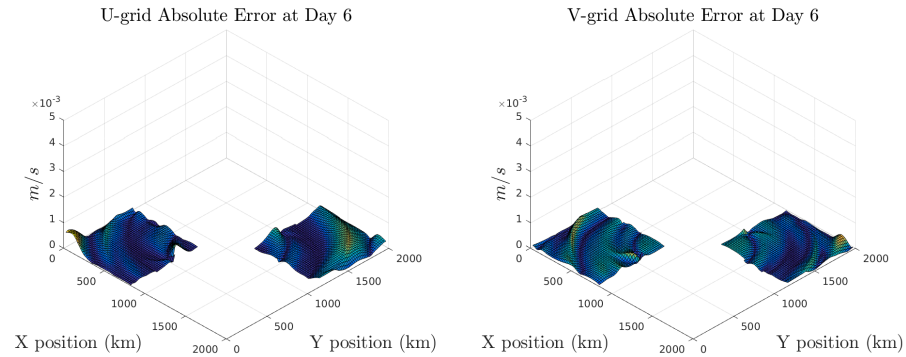


Figure A.17: Validation Simulation: Absolute Error at Day 6

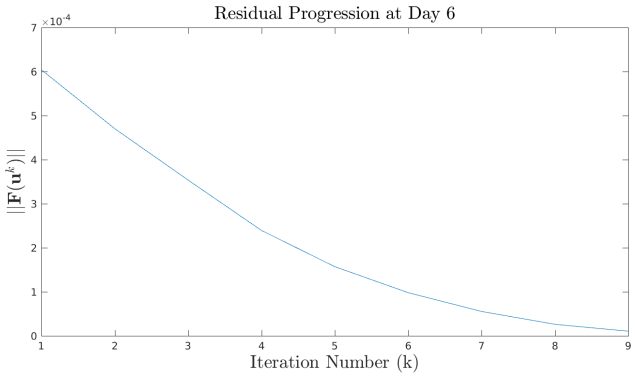


Figure A.18: Validation Simulation: Residual Progression at Day 6 Solve

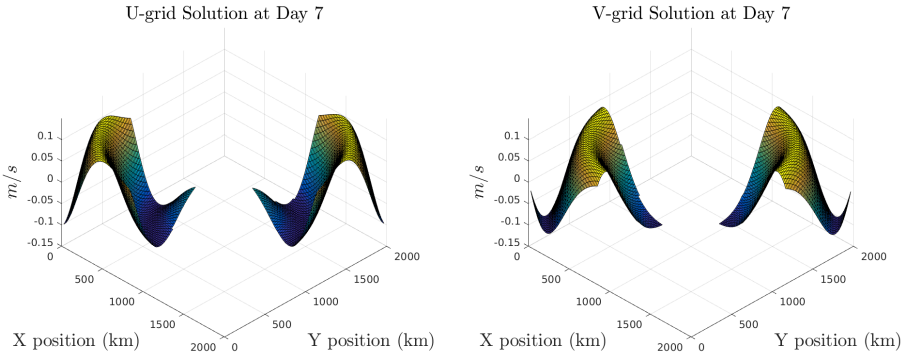


Figure A.19: Validation Simulation: Solution at Day 7

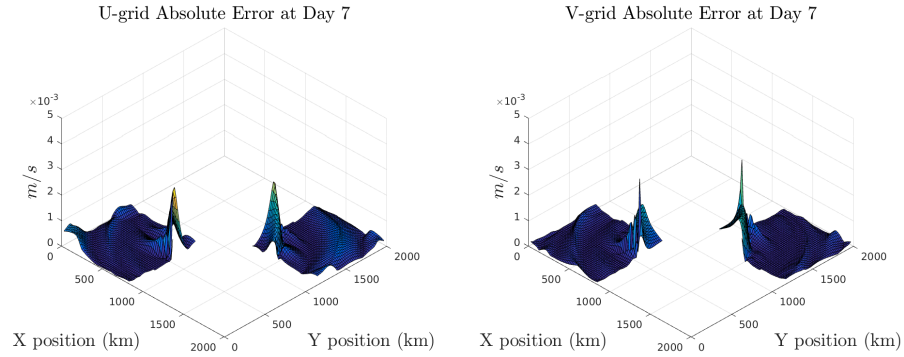


Figure A.20: Validation Simulation: Absolute Error at Day 7

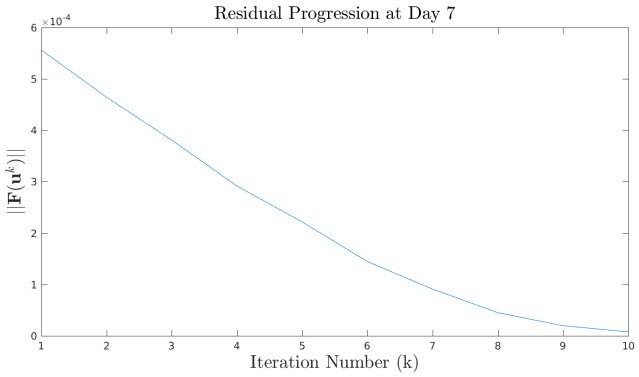


Figure A.21: Validation Simulation: Residual Progression at Day 7 Solve

Bibliography

[1] M. De La Chevrotiere. *Stochastic and Numerical Models for Tropical Convection and Hadley-Monsoon Dynamics*. PhD thesis, The University of Victoria, 2015.

[2] CICE Consortium. Cice: The los alamos sea ice model - version 5.1. Los Alamos National Laboratory, 2015. accessed from http://oceans11.lanl.gov/trac/CICE.

[3] M. Coon, G. Maykut, R. Pritchard, D. Rothrock, and A. Thorndlike. Modeling the pack ice as an elastic-plastic material. *AIDJEX Bulletin*, 24, 1974.

[4] L-P. Saumier Demers. An efficient numerical algorithm for the l2 optimal transport problem with applications to image processing. Master's thesis, The University of Victoria, 2008.

[5] D. Feltham. Sea ice rheology. *Annual Review of Fluid Mechanics*, 40, 2008.

[6] R. Giovanni. *Central Schemes for Balance Laws*, pages 821–829. Birkhäuser Basel, Basel, 2001.

[7] J. Gray and P. Killworth. Stability of the viscous-plastic sea ice rheology. *Journal of Physical Oceanography*, 25, 1994.

[8] W. Hibler. A viscous sea ice law as a stochastic average of plasticity. *Journal of Geophysical Research*, 82(27), 1977.

[9] W. Hibler. A dynamic thermodynamic sea ice model. *Journal of Physical Oceanography*, 9:815–846, 1979.

[10] E. Hunke. Viscous-plastic sea ice dynamics with the evp model: Linearization issues. *Journal of Computational Physics*, 170, 2001.

[11] E. Hunke and J. Dukowicz. An elastic-viscous-plastic model for sea ice dynamics. *Journal of Physical Oceanography*, 27, 1997.

[12] E. Hunke and Y. Zhang. A comparison of sea ice dynamics models at high resolutions. *Monthly Weather Review*, 127, 1998.

[13] J. Hutchings, H. Jasak, and S. Laxon. A strength implicit correction scheme for the viscous-plastic sea ice model. *Ocean Modelling*, 7, 2004.

[14] S. Juricke and T. Jung. Influence of stochastic sea ice parametrization on climate and the role of the atmosphere-sea ice-ocean interaction. *Phil. Trans. R. Soc. A*, 372, 2014.

[15] V. Kattsov, V. Ryabinn, J. Overland, M. Serreze, M. Visbeck, J. Walsh, W. Meier, and X. Zhang. Arctic sea-ice change: a grand challenge of climate science. *Journal of Glaciology*, 2010.

[16] D. Knoll and D. Keyes. Jacobian-free newton-krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193, 2004.

[17] J-F. Lemieux, D. Knoll, M. Losch, and C. Girard. A second-order accurate in time implicit-explicit (imex) integration scheme for sea ice dynamics. *Journal of Computational Physics*, 2014.

[18] J-F. Lemieux, D. Knoll, B. Tremblay, D. Holland, and M. Losch. A comparison of the jacobian-free newton-krylov method and the evp model for solving the sea ice momentum equation with a viscous-plastic formulation: A serial algorithm study. *Journal of Computational Physics*, 231:5926–5944, 2012.

[19] J-F. Lemieux and B. Tremblay. Numerical convergence of viscous-plastic sea ice models. *Journal of Geophysical Research*, 114, 2009.

[20] J-F. Lemieux, B. Tremblay, J. Sedlacek, P. Tupper, S. Thomas, D. Huard, and J-P. Auclair. Improving the numerical convergence of viscous-plastic sea ice models with the jacobian-free newton-krylov method. *Journal of Computational Physics*, 229:2840–2852, 2010.

[21] J-F. Lemieux, B. Tremblay, S. Thomas, J. Sedlacek, and L. Mysak. Using the preconditioned generalized minimum residual (gmres) method to solve the sea-ice momentum equation. *Journal of Geophysical Research*, 113, 2008.

[22] R. LeVeque. *Finite volume methods for hyperbolic problems*. Cambridge University Press, 2002.

[23] R. LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems.* SIAM, 2007.

[24] W. Lipscomb and E. Hunke. Modeling sea ice transport using incremental remapping. *Monthly Weather Review*, 136(6), 2004.

[25] W. Lipscomb, E. Hunke, W. Maslowski, and J. Jakacki. Riding, strength, and stability in high-resolution sea ice models. *Journal of Geophysical Research*, 112, 2007.

[26] M. Losch and S. Danilov. On solving the momentum equations of dynamic sea ice models with implicit solvers and the elastic-viscous-plastic technique. *Ocean Modelling*, 41, 2012.

[27] M. Losch, A. Fuchs, J-F. Lemieux, and A. Vanselow. A parallel jacobian-free newton-krylov solver for a coupled sea ice-ocean model. *Journal of Computational Physics*, 257, 2014.

[28] H. Nessyahu and E. Tadmor. Non-oscillatory central differencing for hyperbolic conservation laws. *Journal of Computational Physics*, 87(2):408–463, 1990.

[29] D. Perovich and J. Richter-Menge. Regional variability in sea ice melt in a changing arctic. *Philosophical Transactions A*, 373, 2015.

[30] P.J. Roache. Quantification of uncertainty in computational fluid dynamics. *Annu. Rev. Fluid. Mech.*, 29, 1997.

[31] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*, 29:867–884, 1992.

[32] J. Stroeve, V. Kattsov, A. Barrett, M. Serreze, T. Pavlova, M. Holland, and W. Meier. Trends in arctic sea ice extent for cmip5, cmip3 and observations. *Geophysical Research Letters*, 29, 2012.

[33] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

[34] L. Tremblay and L. Mysak. Modeling sea ice as a granular material, including the dilatancy effect. *Journal of Physical Oceanography*, 27, 1997.

[35] J. Zhang and W. Hibler. On an efficient numerical method for modeling sea ice dynamics. *Journal of Geophysical Research*, 102, 1997.