2008

Development of a Higher-Order Ice Sheet Model Using a Rescaled Coordinate System

James Fishbaugh
The University of Montana

DEVELOPMENT OF A HIGHER-ORDER ICE SHEET MODEL USING A RESCALED COORDINATE SYSTEM

By

James Michael Fishbaugh

Bachelor of Science, University of Montana, Missoula, MT, 2006

Thesis

presented in partial fulfillment of the requirements
for the degree of

Master of Science
in Computer Science

The University of Montana
Missoula, MT

Spring 2008

Approved by:

Dr. David A. Strobel, Dean
Graduate School

Dr. Jesse Johnson, Chair
Computer Science

Dr. Joel Henry
Computer Science

Dr. James Jacobs
Physics and Astronomy

Fishbaugh, James, M.S., May 2008 Computer Science

Development of a Higher-Order Ice Sheet Model Using a Rescaled Coordinate System

Chairperson: Dr. Jesse Johnson

The Intergovernmental Panel on Climate Change (IPCC) has estimated between 9 and 88 cm of sea level rise over the next hundred years. Of this, only negative 19 to 11 cm is attributed to the largest ice masses on the planet, the Antarctic and Greenland ice sheets. Over the last decade, dramatic activity in the outlet glaciers of Greenland and the Antarctic Peninsula raise the possibility that these large ice sheets will have a much greater contribution to sea level rise over the next century than was predicted by the IPCC. Recent studies have shown these areas are exhibiting decadal scale changes in response to climate forcings, whereas IPCC models show that ice is not responsive to climate change over such short periods of time. Many believe the IPCC type models fail to show short term climate responses due to the simplifications they make to ice sheet mechanics. Here, we develop a higher-order model – a new ice sheet model which contains all relevant flow physics. In order to gauge our progress, we perform a verification of our model around a structured set of experiments. The analysis reveals our model is performing well over a range of different scenarios.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank all of the people who have been influential in my development as a scientist and a person during my time at the University of Montana. First, I'd like to thank Dr. Jesse Johnson for his unwavering support and for his countless explanations. Furthermore, he always treated me as a peer even when it was clear I was not. To Dr. Joel Henry, who has served as my mentor for the last several years. All my major life decisions have been influenced by his recommendations. Finally, to my peers, for all the collaboration and laughs we have shared.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Introduction

In recent years, attention has been drawn to the Earth's changing climate and the potential impact such a change would have on its inhabitants. Of particular concern is the effect that changing climate will have on sea level, as 11 of the world's 15 largest cities are located along the coast [Gornitz, 2000]. As such, a rise in sea level would displace millions of people worldwide. In the western world, this has the greatest impact on California, the gulf of Mexico, and Florida. A catastrophic rise in sea level would displace several million people in these areas. However, the impact worldwide is much more grim, as the population density is very high in areas such as Southeast Asia. The same catastrophic sea level rise would displace hundreds of millions in Asia. This type of flooding would topple established economies and overwhelm the population of other metropolitan areas as people migrate inland.

The scenario presented above projects the global impact of sea level rise under the assumption we take no steps to prevent against flooding. However, it is more likely that these areas will attempt to defend against the rising sea. This can be accomplished a number of ways, but the most common method of protection is the construction of dikes or levees. However, construction of such structures is extremely expensive. According to the Environmental Protection Agency (EPA), a 1 m rise

Table 1.1 Summary of Results from the EPA's Report on Sea Level Rise. Costs in billions of dollars [Titus et al., 1991].

	Baseline	**50cm**	**100cm**	**200cm**
No Shores Protected				
Dryland Lost (sq mi)	N.C.	3,315-7,311	5,123-10,330	8,191-15,39
Wetlands Lost (%)	N.C.	17-43	26-66	29-76
Developed Areas Protected				
Dryland Lost (sq mi)	1,470-4,686	2,200-6,100	4,100-9,200	6,400-13,500
Wetlands Lost (%)	9-25	20-45	29-69	33-80
Value of Lost Land	16-47	52-130	86-212	112-297
Wetlands	5-43	11-82	17-128	19-144
Undeveloped Land	6-19	13-34	21-71	29-121
Land for Dikes	0	9-33	14-48	22-74
Cost of Coastal Defense	**4**	**55-123**	**143-305**	**402-645**
Open Coast: Sand	4	15-81	27-146	59-284
Open Coast: Structures	0	29-36	62-170	257-316
Sheltered Shores: Dikes	0	5-13	11-33	30-101
Total Cost	20-51	128-232	270-475	576-880
If All Shores Protected				
Wetlands Lost (%)	N.C.	38-61	50-82	66-90

would cost between 270 and 475 billion dollars to defend worldwide. Table 1.1 shows the summary of the EPA's report on sea level rise, summarizing both the impact and cost of sea level rise.

There are two major contributors to sea level, both directly influenced by climate. First, thermal expansion takes place as water expands in volume as temperature increases. The other influence is melt water from warming glaciers and ice sheets. Presently, thermal expansion is responsible for the majority of sea level rise, though glaciers play a significant role. However, our current understanding of the impact of glaciers is limited. The contributors to sea level change are shown in figure 1.1.

Of primary concern are the two largest ice masses on the planet, Greenland and Antarctica. The combined volume of ice from the two is staggering, equivalent to approximately 75 m of sea level equivalent[Van Der Veen, 1999]. While it is highly

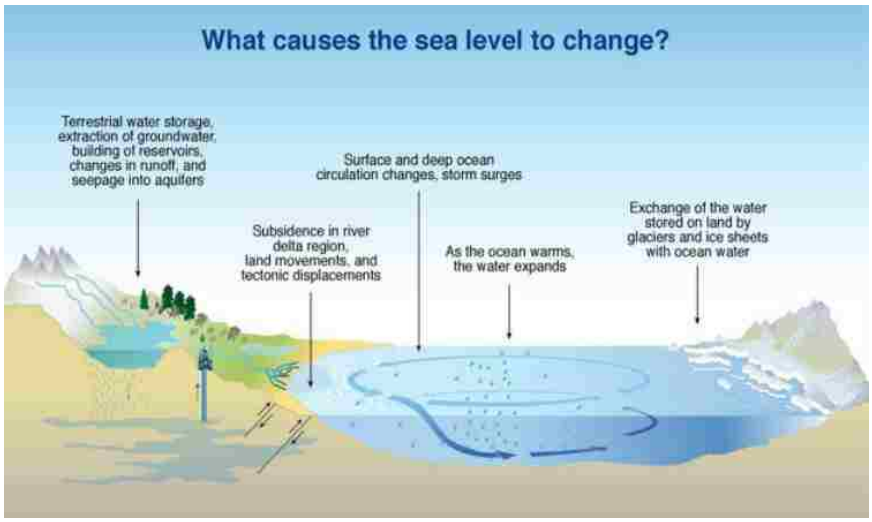


Figure 1.1 The various contributors to sea level change.

unlikely the entire ice sheets will disappear in the short term, the stability of Greenland and Antarctica is an open and fiercely argued issue. There is no agreed upon forecast, as current ice sheet models are unable to resolve fast responses to climate change.

Making matters more uncertain, we have little physical data from glaciers. They are often located in remote and difficult areas to access, making long studies particularly challenging. Making the task even more demanding is hauling the instrumentation required for drilling and taking measurements. In addition to the logistics, gathering the data is also a challenge. Ice sheets move at such slow speeds, and behave in vastly different ways depending on where the measurements are taken, that it is hard to obtain meaningful data.

The most critical missing data are time series on appropriately long scales. Glaciation/deglaciation cycles (due to the Earth's orbital obliquity) are roughly 100,000 years. Data gathering efforts have been limited to the last hundred years; we do not have data over a sufficient length of time. In light of all these shortcomings, we have

very little data and are forced to rely heavily on simulations.

1.2 Motivation

Reacting to global concern, the Intergovernmental Panel on Climate Change (IPCC) released a climate assessment which included an investigation into sea level change. They estimate between 9 and 88 cm of sea level rise over the next century, mostly attributed to thermal expansion. The contribution from Greenland and Antarctica was estimated at -19 to +11 cm, implying that these areas may either grow or shrink in size over the next one hundred years. This sort of uncertainty does not provide readers of the report with confidence in one outcome or the other. Figure 1.2 summarizes the sea level predictions released by the IPCC.

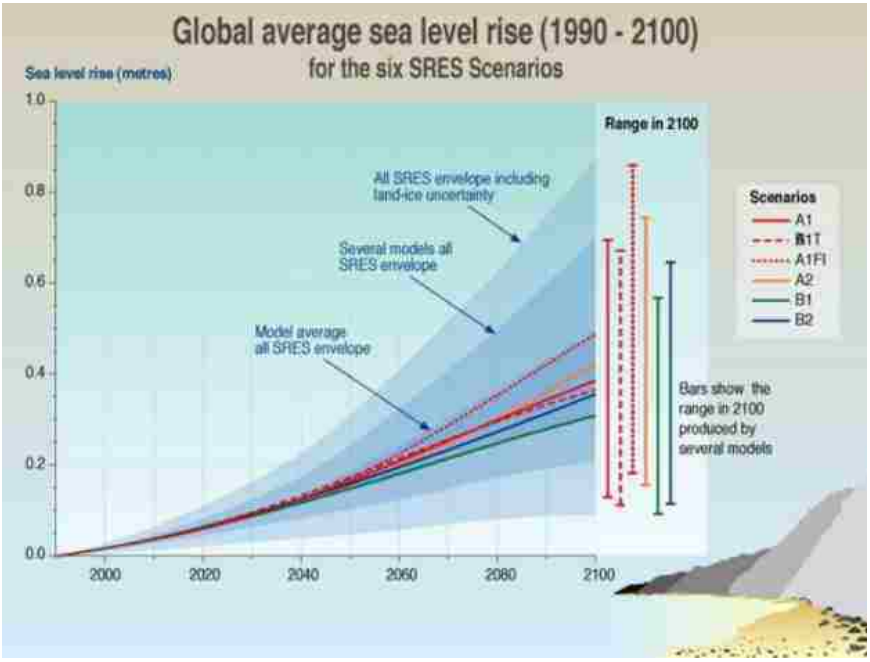


Figure 1.2 The IPCC's prediction of sea level rise over the next century.

However, recent dramatic activity in the outlet glaciers of Greenland and the

Antarctic Peninsula raise the possibility that these large ice sheets will have a much greater contribution to sea level rise over the next century than was predicted by the IPCC. Studies have shown that these areas exhibit changes over much smaller time scales than previously assumed, responding to climate forcings on a decadal scale [Rignot et al., 2008, Rignot and Kanagaratnam, 2006]. Areas on the Antarctic peninsula, for example, have sped up dramatically in the last 15 years, as shown in figure 1.3. Previously, we believed that glaciers were not influenced by climate changes on scales of less than a century.

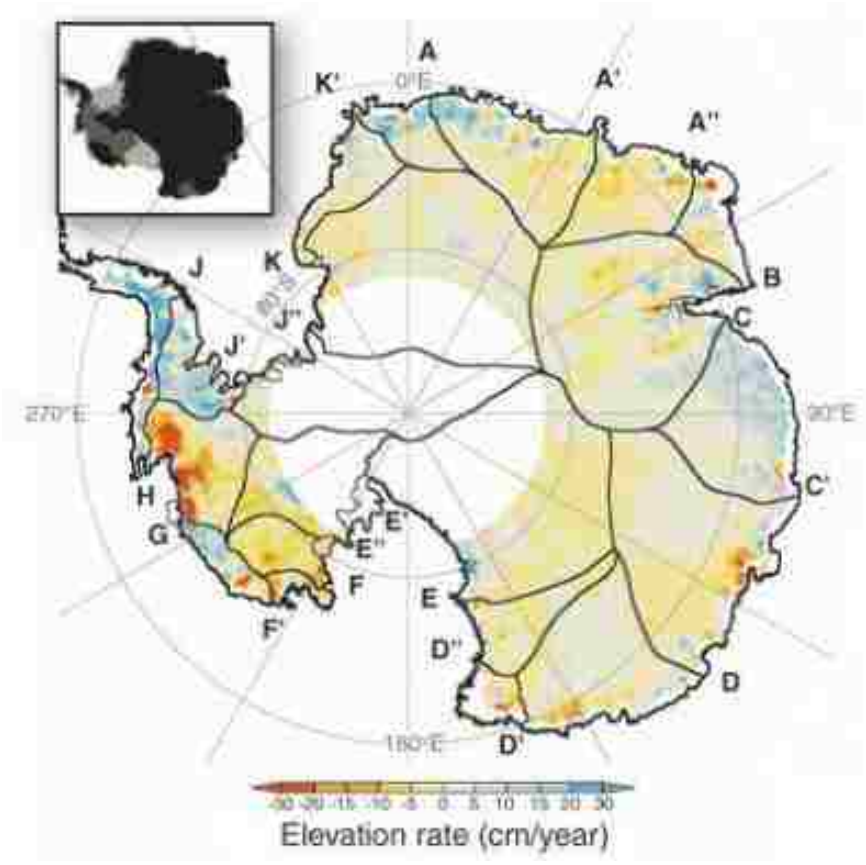


Figure 1.3 Rate of elevation change of the Antarctic Ice Sheet from 1992 to 2003. Inset shows bedrock geometry, highlighting floating (light gray), marine-base (mid-gray), and continental-based (black) sectors [Shepherd and Wingham, 2007].

Armed with this new knowledge of fast ice sheet response, we are skeptical of the results published by the IPCC. The models used by the IPCC are generally believed to not account for this recently discovered phenomenon due their simplification of ice sheet mechanics. As a result, we propose that the estimations made by the IPCC are possibly inaccurate. In response to the shortcomings of existing strategies, we develop a higher-order ice sheet model that contains all the relevant flow physics. A higher-order model is one that contains additional mechanical effects and does not apply the shallow ice approximation, discussed in chapter 2.

1.3 Goals

There are several goals for this work, both in the short and long term. The immediate focus is developing a higher-order ice sheet model that is highly flexible, capable of handling many different types of experiments. In order to achieve this goal we must be establish that our model is working as desired by comparing our results to those obtained from a structured set of experiments. The primary goal here is developing a model that can complete the experiments proposed by the Ice Sheet Model Intercomparison Project for Higher-Order Models (ISMIP-HOM).

In addition to the short term goal, there are over-arching long term goals that derive directly from the motivation for this work. We hope to establish a new model with more realistic physics to improve the accuracy of ice sheet simulations. This would allow us, or future researchers, to revisit the IPCC's study and reevaluate the impact glaciers will have on sea level rise.

Even more generally, this model will serve as an demonstration of the benefit, or lack thereof, of incorporating the additional physics. It is possible that the additional numerics have little impact on the simulations. We need to obtain measurements

to quantify the impact, and weigh that against the cost, in terms of complexity, efficiency, and robustness. This is currently an open question.

1.4 Benefits

Ice sheet models are the basis for predicting the evolution of the world's glaciers. These predictions can have a great impact on humanity, as policy makers use them in drafting important legislation. The less uncertainty present in simulations, the better the world can adapt to a changing environment. Our reliance on models in crafting policy makes it necessary to have highly accurate solutions. Higher-order ice sheet models, such as the one developed here, will pave the way for more correct and realistic predictions. In turn, this will lead to quicker enactment of appropriate policy.

This work will also benefit other researchers in the field, who can gain knowledge from this effort. As mentioned in section 1.3, the ice sheet modeling community needs to reach a conclusion on the benefit of higher-order models. Our work can be used in combination with other higher-order models to determine the effect the additional numerics have on simulations. The lessons learned by what worked well and what fell short could be integrated into the next generation of ice sheet models.

1.5 Thesis Organization

The rest of this thesis is organized as follows:

- **Chapter 2** provides an overview of the physics of ice, ice sheet modeling, and an introduction to the software package being used.
- **Chapter 3** details the numerical methods that define the model as well as the software specific implementation.
- **Chapter 4** describes the results of the research.
- **Chapter 5** contains concluding observations, as well as direction for future work.

CHAPTER 2 OVERVIEW

2.1 Properties of Ice

Ice is defined as water frozen in a solid state. However, ice has properties of both a solid and a liquid. While commonly viewed as an unmoving solid, it is more accurately thought of as a highly viscous fluid. Due to its physical composition, ice behaves in interesting ways, most notably in terms of flow.

2.1.1 Flow

Ice is a very slow moving liquid, with flow governed by the well established laws of fluid dynamics. However, whereas most fluid dynamic problems deal with Newtonian fluids where the stress and strain rates are linearly related, this is not the case with ice. Ice is considered a non-Newtonian fluid; it has a non-linear viscosity. This means that ice offers different resistance to deformation based on the amount and direction of stress applied. These types of fluids create a highly non-linear situation, where viscosity is needed to solve for velocity, however, viscosity is described in terms of velocities. This circular dependency makes fluids with non-linear viscosity more difficult to model than Newtonian fluids.

Due to the viscosity of ice, its flow is described by the Stokes equations, which governs the flow of a fluid where viscous forces are very large compared to inertial forces; acceleration can be ignored. Stokes flow is a simplification of the Navier-Stokes

equations. The Stokes equations will be presented and discussed in greater detail in chapter 3.

2.2 Ice Sheet Modeling

2.2.1 Shallow-Ice Approximation

Most current ice sheet models involve simplifications with respect to the mechanics of ice flow. These simplifications are based on the shallow-ice approximation, which operates under the assumption that the horizontal extent of an ice mass is much greater than the thickness and that the slopes of the surface and bedrock are relatively low [Le Meur et al., 2004].

These situations are represented well by so called low-order models. These models consider the system to be only influenced by vertical stresses, where the force is perpendicular to flow; horizontal shear stresses acting parallel to flow are ignored. In addition, ice flow is driven solely by gravity, neglecting any other sources responsible for flow, such as topographic features.

However, the shallow-ice approximation does not capture some key areas in ice sheets where the most interesting and dynamic events occur, such as a region near the grounding line or a divide [Pattyn, website]. Furthermore, according to Le Meur et al., many mountain glaciers exhibit thicknesses equal to their width, rendering the shallow-ice approximation inaccurate. Ultimately, shallow-ice approximation models are inadequate for handling these types of situations.

2.2.2 Higher-Order Models

In order to accurately model areas of an ice sheet where the aspect ratio is large, additional numerics beyond those included in shallow-ice approximation models must

be included. A higher-order model is one that incorporates further mechanics effects, principally longitudinal shear stresses [Pattyn and Payne, 2006]. Further effects can be introduced, such as allowing for basal sliding and implementing downstream features that retard flow.

While higher-order models contain addition mechanical effects, they still make some simplifications to ice flow equations [Pattyn and Payne, 2006]. For instance, the higher-order model proposed by Pattyn 2003 he applies the hydrostatic approximation in the vertical, implying that the variational stress is neglected. This type of assumption can have a large impact on certain simulations, particularly in "regions where the flow regime changes, such as near the ice divide or near the margin" [Pattyn, 2003].

In contrast to simplified higher-order models, full Stokes models make no such assumptions. They include all relevant stresses as well as additional forces driving acceleration. These types of models are valid for all types of simulations, but introduce considerably more complexity in implementation and are often time consuming to run. Due to the difficulty in implementation, very few full Stokes models exist.

2.3 COMSOL Multiphysics

2.3.1 Overview

Comsol Multiphysics is a software package used for modeling systems in physics and engineering. It is a general finite element solver, capable of modeling many common systems out of the box, as well as allowing for others through the addition of modules. Most notably, Comsol allows for user to specify their own partial differential equations (PDEs) and couple them with external data and geometries. Figure 2.1 shows the many different types of simulations available to a user.

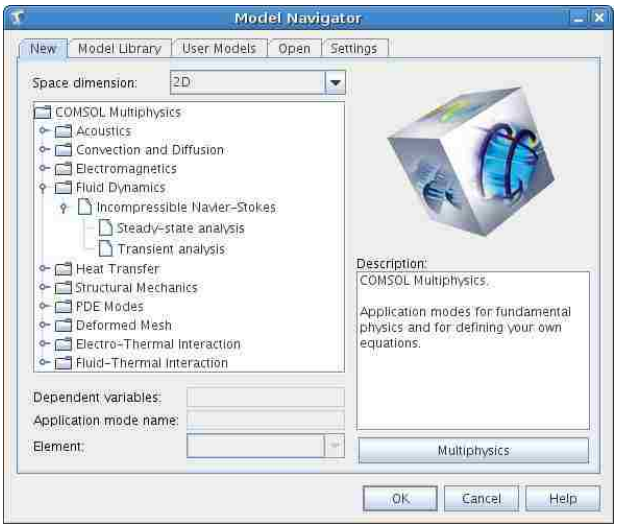


Figure 2.1 The COMSOL Model Navigator lets the user choose from many types of multi-physics simulations.

The software provides power through both predefined modeling interfaces and with its flexibility and avenues for customization. There are existing interfaces for various applications including fluid flow, where simulations can be set up and run in short periods of time. Figure 2.2 shows the main COMSOL application set up in Navier-Stokes mode. Conversely, the user also has full access to the inner workings, both in terms of the physics and the solver. Furthermore, material properties, source terms, and boundary conditions can be all uniquely specified by the user [com, website].

Other notable features include full control of meshing, the ability to view post-processing results, and a comprehensive interface to MATLAB. A final prominent feature of COMSOL worth mentioning is the scripting language built-in to the application. This allows users to write scripts to specify model parameters, build user interfaces, and to visualize and analyze their data. For all of the reasons discussed, COMSOL is an excellent resource for modelers.

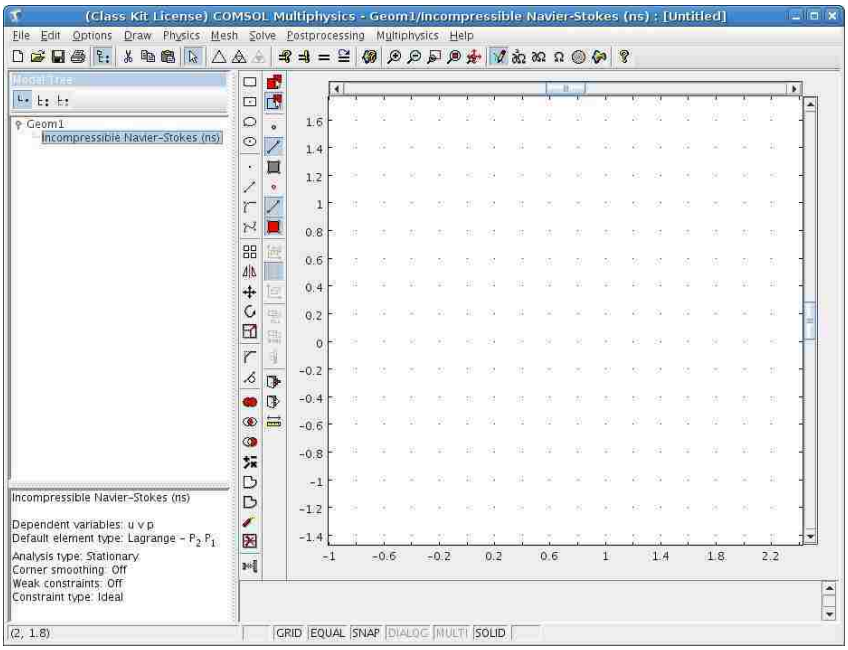


Figure 2.2 The main screen of the COMSOL application.

2.3.2 Limitations

Though COMSOL has a great deal of built in facilities for modeling, it is not without its limitations. In developing this model, we are confined to work inside these limitations and tasked with overcoming them. The intent of this work is to implement a full Stokes ice sheet model using COMSOL, however, full Stokes flow is not inherent to the software. Instead, we must achieve this through combining our understanding of ice mechanics with the inner workings of COMSOL's Navier-Stokes mode. The details of the implementation are discussed in chapter 3.

Additionally, the unstructured mesh which the finite element solver uses can be troublesome. For certain domains, the length of the glacier can be much greater than its thickness, causing the mesh to be stretched considerably in the horizontal dimension. As such, refining the mesh in the vertical adds a non-trivial number of

vertices, resulting in more computation nodes and extending run time. For example, the mesh presented in figure 2.3 has adequate resolution horizontally but poor vertical resolution. Further refinement would introduce unwanted complexity.

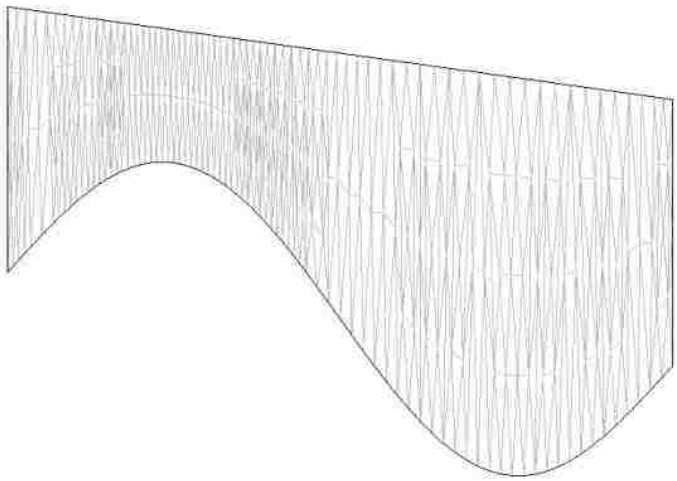


Figure 2.3 An example mesh where the ice is considerably wider than it is thick.

The cost of mesh complexity can be measured in terms of model execution. While complexity in a mesh often equates to a longer simulation, that is not the greatest concern. Instead, a highly complex mesh can often result in a run that never converges to a solution. This is the case when domains such as the one shown in figure 2.3 are evolved over time. While performing time evolution, the grid points that constitute the mesh tend to overlap, effectively folding over each other. This produces inaccurate results or leads to model runs that never finish.

In order to overcome this limitation, we introduce a coordinate transformation to take all variability out of defining a domain. By doing so, we greatly simplify the underlying mesh. Figure 2.4 demonstrates the effects of the coordinate transformation. The transformed mesh has greater clarity and simplicity, leading to more efficient execution, and most importantly, convergence. This scheme will ultimately allow for

performing time dependent experiments, making it a pivotal step in obtaining the goals of the research. The coordinate transformation is detailed in chapter 3.

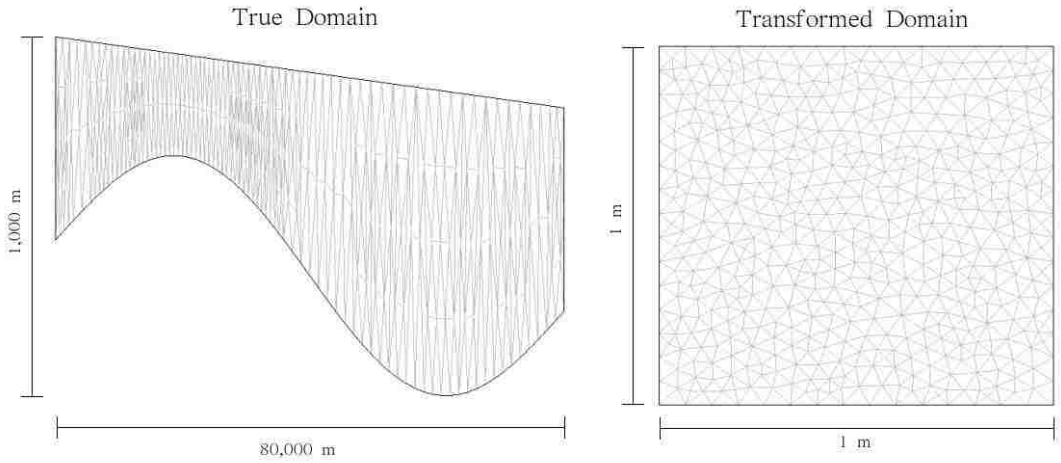


Figure 2.4 A comparison of the standard and transformed mesh. Notice the simplicity and clarity present in the mesh from the rescaled coordinate system.

CHAPTER 3 METHODS

3.1 Numerics

The majority of current large scale ice sheet models are based on the SIA, where ice flow is driven by gravity and almost no shearing exists near the surface. A higher-order ice sheet model is one that considers longitudinal stresses in addition to horizontal plane shear stress [Pattyn and Payne, 2006]. Furthermore, lower-order models do not account for interactions at the margins of sliding ice or downstream obstacles. As a result, these models do a poor job of modeling these situations.

Though current higher-order models do consider additional stresses, they also make reductions in the numerics to reduce complexity. For example, the model proposed by Pattyn in 2003 applies the hydrostatic approximation in the vertical, implying that variational stress is neglected [Pattyn, 2003]. Our model makes no such assumptions, including all relevant stresses. A discussion of the numerics of our model follows.

3.1.1 Field Equations

As discussed in chapter 2, the flow of a fluid is described by the Navier-Stokes equations. The flow of ice, considered a non-Newtonian fluid with a non-linear viscosity, is described by a simplified version of Navier-Stokes, called simply the Stokes equations. The equations are based on Newton's second law, which states that the net force on an object is proportionate to the rate of change of its linear momentum,

most commonly written as

$$\mathbf{F} = \frac{d}{dt}(m\mathbf{v}), \tag{3.1}$$

where $\mathbf{F}$ is the net force, m is the mass of the object, and $\mathbf{v}$ is the velocity vector.

Following directly from Newton's second law, the Stokes equations represent conservations of mass and momentum, written as

$$\nabla \cdot \mathbf{v} = 0, \tag{3.2}$$

$$\nabla \cdot \sigma = \mathbf{F}, \tag{3.3}$$

where ∇ is a vector differential operator written as

$$\nabla = \frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k}, \tag{3.4}$$

$\mathbf{v}$ is the velocity vector (u,v,w), σ is the stress tensor, and $\mathbf{F}$ is the external force. For the domain being modeling, we are concerned with stress, which is force per unit area, written as

$$\mathbf{F} = \rho\mathbf{g}. \tag{3.5}$$

where g is gravitational acceleration and ρ is the density of ice.

The stress tensor, σ is a three dimensional array which describes the various stresses acting on the ice, written as

$$\sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{pmatrix} \tag{3.6}$$

where the subscripts indicate the axis perpendicular to the plane the force is acting

Table 3.1 Constants Used in the Model

Symbol	Constant	Value	Units
ρ	ice density	910	kg m^{-3}
g	gravitation constant	9.81	m s^{-2}
A	ice 'hardness'	$1e^{-16}$	$Pa^{-3}a^{-1}$
L	length of domain	experiment specific	m
n	exponent in Glen's flow law	3	
ϵ	prevents floating-point overflow	$1e^{-18}$	

upon as well as the direction of the force. For example, σ_{xy} represents a force acting perpendicular to the x axis in the y direction. It is important to note that σ_{ii} are normal stresses and the remaining terms represent shear stress. Furthermore, there is redundancy due to symmetry, where $\sigma_{ij} = \sigma_{ji}$.

Treating ice as an incompressible fluid with constant density, the equation for conservation of momentum can be rewritten as

$$\nabla \cdot \sigma = \rho \mathbf{g}, \tag{3.7}$$

by substituting known quantities into equation 3.3.

By considering a Cartesian coordinate system (x, y, z) with the z-axis denoting the vertical dimension, we can also rewrite conservation of mass (3.2) as

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0. \tag{3.8}$$

Furthermore, by applying the same coordinate system and considering gravitational acceleration only important in the vertical direction, momentum conservation (3.3)

becomes

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} = 0, \tag{3.9}$$

$$\frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} = 0, \tag{3.10}$$

$$\frac{\partial \sigma_{zx}}{\partial x} + \frac{\partial \sigma_{zy}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} = \rho g. \tag{3.11}$$

Next, we introduce Glen's flow law, also referred to as the constitutive law of ice. It relates stress to strain rate as follows,

$$\dot{\varepsilon}_{ij} = A \tau_e^{n-1} \tau_{ij} \tag{3.12}$$

where $\dot{\varepsilon}_{ij}$ is the strain rate, A is the flow parameter, τ_e is the second invariant of the stress tensor, n is the flow law constant, and τ_{ij} is the deviatoric stress tensor (σ with influence from pressure removed). In this model, we only consider isothermal experiments; the value for A is constant. The constants are used in this model are summarized in table 3.1.

In modeling ice sheets, we are primarily concerned with shearing forces acting on the ice mass. As such, it is convenient to separate the stress tensor into two components. This is accomplished by subtracted the mean normal pressure component, resulting in τ_{ij}, the deviatoric stress tensor [Lund, 2000]. The mean normal pressure component is written as

$$P_{mn} = -\frac{\sigma_{xx} + \sigma_{yy} + \sigma_{zz}}{3}, \tag{3.13}$$

where the normal stresses are summed and averaged. The effect of mean normal

stresses are then removed, leaving

$$\tau = \sigma + \mathbf{I}P_{mn}, \tag{3.14}$$

where $\mathbf{I}$ is the 3x3 identity matrix. This scheme emphasizes the forces of shearing rather than pressure.

The dynamic viscosity of ice is directly related to the constitutive law, written as

$$\eta = \frac{1}{2}A^{-1/n}(\epsilon + \dot{\varepsilon}_e)^{(1-n)/n}, \tag{3.15}$$

where $\dot{\varepsilon}_e^{(1-n)/n}$ is the second invariant of the strain rate tensor and ϵ is a small constant $(1e^{-18})$ which ensures finite viscosity. Finally, we can relate the deviatoric stress to viscosity and strain rate as follows

$$\tau_{ij} = 2\tau\dot{\varepsilon}_{ij}, \tag{3.16}$$

which concludes our treatment of field equations.

3.1.2 Boundary Conditions

For experiments conducted here, periodic boundary conditions are used at the edges of the domain, simulating an infinite system. In the 2D case, what flows out the right side flows into the left. The surface of the ice is considered stress-free, described as

$$\sigma \cdot \mathbf{n}_s = 0, \tag{3.17}$$

where $\mathbf{n}_s$ is the normal vector to the surface. The bed is a no-slip boundary, written as

$$\mathbf{v} = 0, \tag{3.18}$$

meaning there is no velocity over the bedrock in any direction.

However, for some experiments, ice is allowed to slide over the bed. Basal sliding is achieved through a friction coefficient and the following friction law,

$$\beta^2 \mathbf{t} \cdot \mathbf{v} = \mathbf{t} \cdot (\sigma \mathbf{n}_b) \tag{3.19}$$

where β^2 is a positive scalar quantity, $\mathbf{t}$ are the unit tangent vectors, and $\mathbf{n}_b$ are the unit normal vectors pointing into the bedrock [Pattyn et al., 2008].

3.1.3 Coordinate Transformation

For numerical convenience and to improve element quality in the finite element solver, this model works on a transformed domain. Rather than explicitly defining the domain, a 1x1 square (in the 2D case) or a 1x1x1 cube (in the 3D case, see figure 3.1) is used and a transformed coordinate system captures the true geometry. The transformation, described here in two dimensions, but easily extended to three, maps $(x,\, z) \rightarrow (x',\, z')$ where

$$x' = \frac{x}{L}, \tag{3.20}$$

$$z' = \frac{z - b}{s - b}, \tag{3.21}$$

where L is the horizontal extend and s and b represent the surface and bed. The vertical coordinate z' is defined such that $z' = 1$ at the surface and $z' = 0$ at the bed.

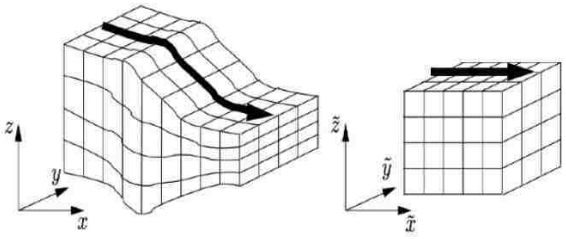


Figure 3.1 A conceptual model of the coordinate transformation in three dimensions.

In addition to coordinate scaling, all function derivatives must also be transformed to reflect the change in coordinates. Using the chain rule, the derivatives for x and z transform to

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial x'}\frac{\partial x'}{\partial x} + \frac{\partial f}{\partial z'}\frac{\partial z'}{\partial x}, \tag{3.22}$$

$$\frac{\partial f}{\partial z} = \frac{\partial f}{\partial x'}\frac{\partial x'}{\partial z} + \frac{\partial f}{\partial z'}\frac{\partial z'}{\partial z}, \tag{3.23}$$

where the simple derivatives, $\partial z'/\partial z$, $\partial x'/\partial x$, and $\partial x'/\partial z$ are written as

$$\frac{\partial z'}{\partial z} = \frac{\partial}{\partial z}\left(\frac{z-s}{H}\right) = \frac{1}{H}, \tag{3.24}$$

$$\frac{\partial x'}{\partial x} = \frac{\partial}{\partial x}\left(\frac{x}{L}\right) = \frac{1}{L}, \tag{3.25}$$

$$\frac{\partial x'}{\partial z} = \frac{\partial}{\partial z}\left(\frac{x}{L}\right) = 0. \tag{3.26}$$

The derivation of $\partial z'/\partial x$ is less straightforward and shown below for clarity.

$$\begin{aligned}
\frac{\partial z'}{\partial z} &= (z-b)\frac{\partial}{\partial x} + \frac{1}{(s-b)}\frac{\partial}{\partial x}(z-b) \\
&= -\frac{(z-b)}{(s-b)^2}\left(\frac{\partial s}{\partial x} - \frac{\partial b}{\partial x}\right) - \frac{1}{(s-b)}\frac{\partial b}{\partial x} \\
&= -\frac{(z-b)}{(s-b)}\frac{1}{(s-b)}\left(\frac{\partial s}{\partial x} - \frac{\partial b}{\partial x}\right) - \frac{1}{H}\frac{\partial b}{\partial x} \\
&= -\frac{z'}{H}\left(\frac{\partial s}{\partial x} - \frac{\partial b}{\partial x}\right) - \frac{1}{H}\frac{\partial b}{\partial x} \\
&= -\frac{1}{HL}\left((1-z')\frac{\partial b}{\partial x} + (z')\frac{\partial s}{\partial z}\right)
\end{aligned} \tag{3.27}$$

Finally, we introduce a convenience variable $a_x = \partial z'/\partial x$ [Pattyn, 2003].

The complete form of the first order derivatives can be find using substitution into equations 3.22 and 3.23, written as

$$\frac{\partial f}{\partial x} = \frac{1}{L}\frac{\partial f}{\partial x'} + a_x\frac{\partial f}{\partial z}, \tag{3.28}$$

$$\frac{\partial f}{\partial z} = \frac{1}{H}\frac{\partial f}{\partial z}. \tag{3.29}$$

Finally, the second order derivatives are shown below. The derivation of b_x, $\partial^2 f/\partial x^2$, $\partial^2 f/\partial z^2$, and $\partial^2 f/\partial z \partial x$ can be obtained using an approach similar to that

employed in finding the first order derivatives [Pattyn, 2003].

$$b_x = \frac{1}{L}\frac{\partial a_x}{\partial x'} + a_x\frac{\partial a_x}{\partial z'}, \tag{3.30}$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{1}{L}\frac{\partial^2 f}{\partial x'^2} + \frac{2a_x}{L}\frac{\partial^2 f}{\partial z'\partial x'} + a_x^2\frac{\partial^2 f}{\partial z'^2} + b_x\frac{\partial f}{\partial z'}, \tag{3.31}$$

$$\frac{\partial^2 f}{\partial z^2} = \frac{1}{H^2}\frac{\partial^2 f}{\partial z'^2}, \tag{3.32}$$

$$\frac{\partial^2 f}{\partial z \partial x} = \frac{1}{HL}\left(\frac{\partial f}{\partial z'\partial x'} + La_x\frac{\partial^2 f}{\partial z'^2} - \frac{1}{H}\frac{\partial f}{\partial z'}\frac{\partial H}{\partial x'}\right). \tag{3.33}$$

3.2 COMSOL Multiphysics

Now that the numerical methods have been specified, they need to be implemented and integrated into COMSOL. One can begin from the fluid mechanics mode, however there are several key changes and additions required. The first step in setting up the model is defining the geometry of the subdomain and establishing periodic boundary conditions. Next, a user must add required constants, scalar expressions, subdomain and boundary settings, as well as modifying the equation system for both the subdomain and boundaries.

3.2.1 Constants

Our model has several parameters that do not change during a simulation. These are stored in COMSOL as constants, which are accessible everywhere in the application. Constants are defined in one location for easy reference and modification. Figure 3.2 shows a screenshot of the constants used for a specific model run, and includes the physical constants specified in table 3.1 as well as several convenience variables. Any additional constants required for an experiment are to be included here as well.

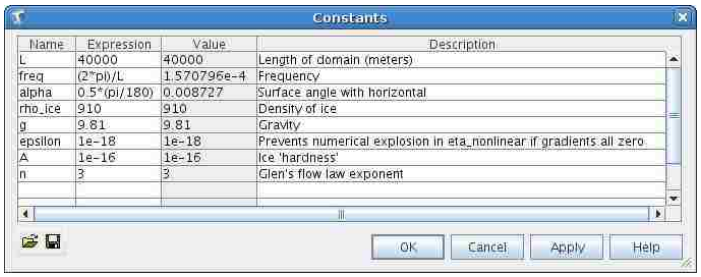


Figure 3.2 The dialog where model constants are added.

3.2.2 Scalar Expressions

Equations that support the model are entered as scalar expressions. These define expressions such as the surface, bed, ice thickness, as well as the numerous derivative transformations. As with constants, scalar expressions are convenience variables which provide the facility to easily make changes and add new expressions. Furthermore, scalar expressions persist outside of the sub domain (defined by the boundaries of the geometry), insulating them from domain changes.

The scalar expressions also include COMSOL specific variables. For example, in COMSOL `ux` stands for $\partial u/\partial x$, `px` stands for $\partial p/\partial x$, and so forth. Also present in the scalar expressions are built-in functions such as `sin` and `atan` which are the sine and arctangent functions, as well as and `diff` which computes derivative values. Figure 3.3 shows a screen shot of the scalar expressions used for a particular experiment. Note how the equations defined as scalar expressions reference model constants.

For consistency, similar naming conventions are used for introducing new variables which mimic the structure of COMSOL expressions. For example, `uxp` is short for ux' and represents $\partial u/\partial x'$. Likewise, `uxxp` represents $\partial u/\partial x'\partial x'$, with the rest of the expressions easily identified by applying this convention. Adopting this naming scheme allows for simplicity and readability when replacing the terms in the under-

lying physics of the system.

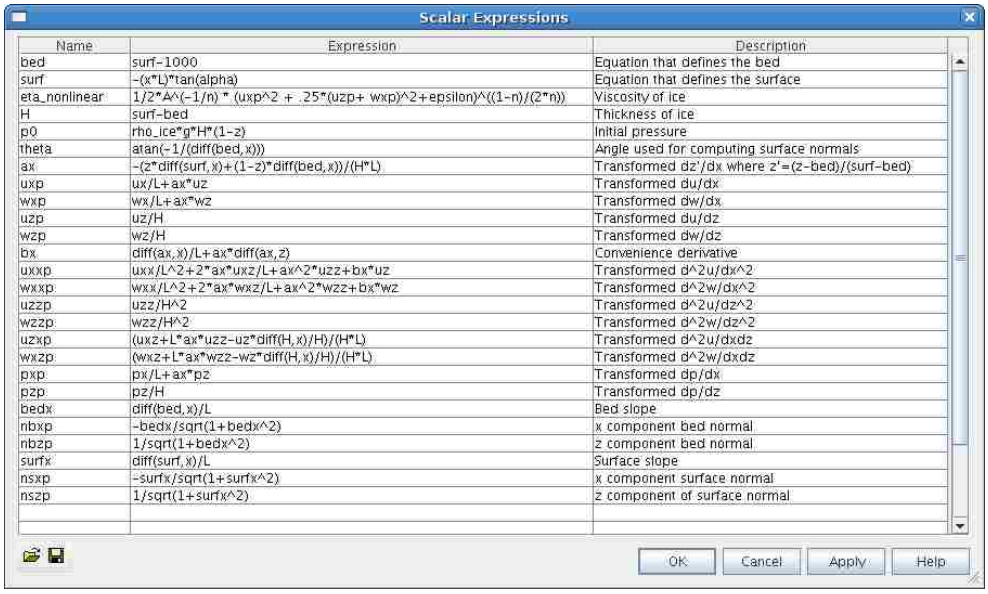


Figure 3.3 The dialog where scalar expressions are added.

A final important note is the implementation of the dynamic viscosity of ice, η, present in the scalar expressions. It is labeled `eta_nonlinear`, and is described in terms of velocity derivatives and constants A, n, and ϵ, defined as COMSOL constants. This term is responsible for the highly non-linear situation, where velocity values are needed to define the viscosity, while simultaneously the viscosity must be known in order to solve for velocity.

3.2.3 Subdomain Settings

Subdomain settings define several important parameters for the simulation. It is here where the density, viscosity, and external forces are entered. Furthermore, initial conditions are defined here, shown in figure 3.5. The parameters used for this model are shown in figure 3.4. Again, notice the use of constants and scalar expression in

defining the subdomain settings. As discussed in section 3.1.1, the driving force in the simulation is gravitational acceleration in the z direction.

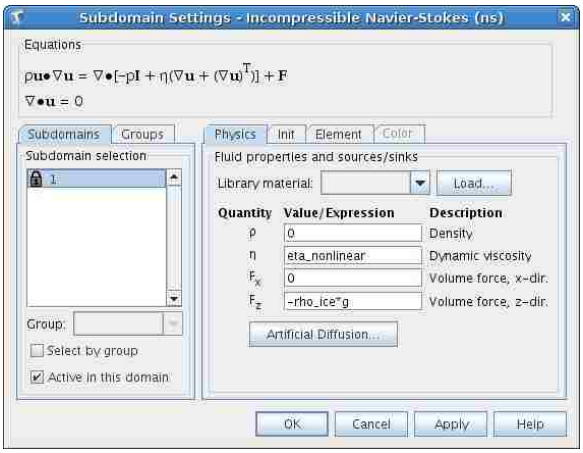


Figure 3.4 The dialog where subdomain changes are made.

Additionally, the subdomain settings window is the location where Stokes flow is activated. By default, COMSOL is set to Navier-Stokes mode when solving a fluid flow problem. However, as discussed in chapter 2, ice is best represented by a reduced version of Navier-Stokes, referred to as the Stokes equations. By setting the value of ρ equal to zero in the subdomain settings, the equation for conservation of momentum is effectively reduced to that described by Stokes flow. The density of ice is captured by the force term F_z.

3.2.4 Boundary Settings

In order to proceed with an simulation, the conditions at the boundaries must be specified. The boundary settings dialog is where boundary conditions are defined. For example, if the ice is frozen to the bedrock, a no-slip boundary is applied to the bed, as shown in figure 3.6. Notice the equation displayed in the upper left, which defines the behavior for the bed. This is in agreement with equation 3.18. For simulations

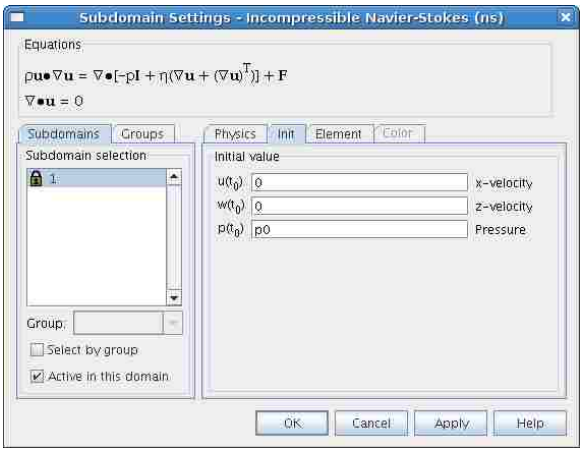


Figure 3.5 The dialog where initialization parameters are set.

where sliding is allowed, the bed is defined as a stress boundary and modifications are made to the boundary equation system, described in section 3.2.5.2.

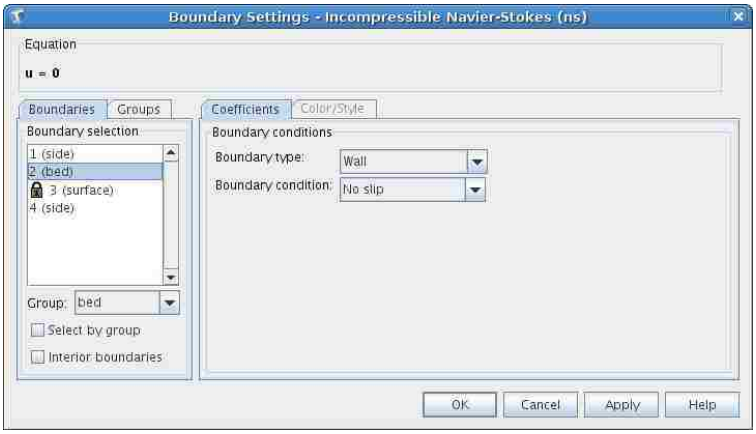


Figure 3.6 Setting a no slip surface boundary.

The boundaries at the surface and sides are most commonly set as open, stress free boundaries, shown in figure 3.7. Notice the equation in the dialog appears different than equation 3.17. By setting the quantity $\mathtt{f}_0 = 0$, we reduce the right hand side to zero, while changes previously made to the subdomain settings are in effect, modifying

the left hand side to the desired expression. This type of condition implies no stress at the boundary.

3.2.5 Equation System

After specifying model constants/scalar expressions and defining subdomain/boundary settings, changes to the equation system must be made. COMSOL performs simulations on the geometry defined, so changing the equation system is required in order to achieve the coordinate transformation. Without modifications, the simulation would be carried out for the explicit geometry rather than the transformed domain.

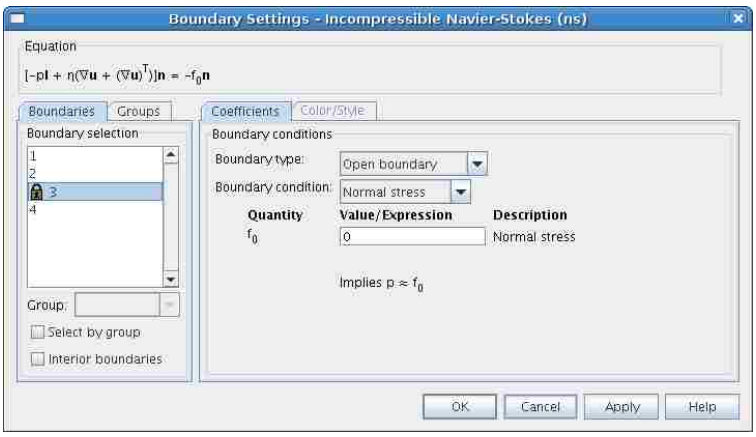


Figure 3.7 Setting an open boundary.

3.2.5.1 Subdomain Equations

The physics behind the model are implemented as subdomain equations. More specifically, Stokes flow is implemented by the subdomain equations. In order to accommodate the coordinate transformation, the function derivatives in the subdomain equation system must be updated. Our transformed derivatives, which are defined as scalar expressions, will need to replace the default numerics supplied by COMSOL.

However, not every term in the equation system must be modified. The important terms are $\mathbf{F}$, $\mathbf{\Gamma}$, `V_ns`, and `divU_ns`, the last two being vorticity and divergence of velocity, located under the *Variable* tab. After these have been specified, clicking the *Differentiate* button will propagate all the necessary changes to the other terms. The other terms are of lesser importance, as they are derived from $\mathbf{F}$ and $\mathbf{\Gamma}$. Figure 3.8 shows the dialog where changes are made to the subdomain equation system.

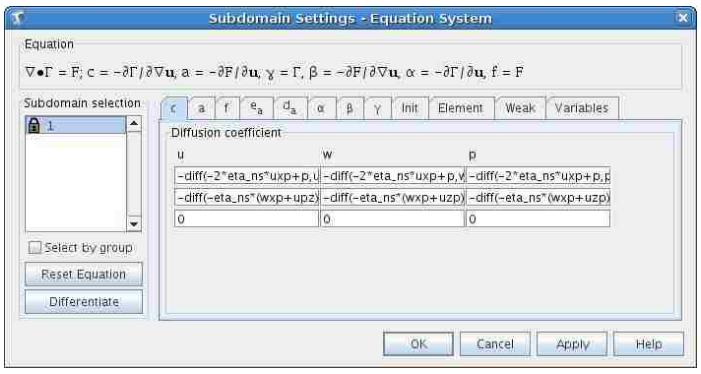


Figure 3.8 The dialog where subdomain equation changes are made.

Outside of simply changing every occurrence of a function derivative to its transformed variety, other changes must be made. This is due to the following equation in COMSOL,

$$\nabla \cdot \mathbf{\Gamma} = \mathbf{F}. \tag{3.34}$$

The problem lies in COMSOL using $\nabla = \partial/\partial x\hat{i} + \partial/\partial z\hat{j}$ for the explicitly defined geometry, rather than our $\nabla' = \partial/\partial x'\hat{i} + \partial/\partial z'\hat{j}$ which reflects the coordinate transformation. Modifications to $\mathbf{F}$ and $\mathbf{\Gamma}$ are required to reconcile this difference.

The first step is to modify $\mathbf{F}$ so the external forces are acting on the transformed

geometry. The modification to $\mathbf{F}$ is as follows

$$F_1 = F_x - a_x \frac{-2\eta(\partial u/\partial x') + p}{dz}, \tag{3.35}$$

$$F_2 = F_z - a_x \frac{-\eta(\partial w/\partial x')(\partial u/\partial z')}{dz}, \tag{3.36}$$

where F_3 remains unchanged. These modifications reflect the difference between the true force as calculated by COMSOL and a small amount of additional force present in the true geometry.

Next, $\mathbf{\Gamma}$ is updated to reflect the change of coordinates. Primarily, the changes involved using the transformed derivatives, but we also introduce scaling terms, required by the substitution of ∇'. Changes to $\mathbf{\Gamma}$ are

$$\Gamma_{11} = (-2\eta(\partial x/\partial x') + p)/L, \tag{3.37}$$

$$\Gamma_{12} = (\eta(\partial x/\partial z') + (\partial w/\partial x'))/H, \tag{3.38}$$

$$\Gamma_{21} = (\eta(\partial w/\partial x') + (\partial u/\partial z'))/L, \tag{3.39}$$

$$\Gamma_{22} = (-2\eta(\partial w/\partial z') + p)/H, \tag{3.40}$$

$$\Gamma_{31} = 0, \tag{3.41}$$

$$\Gamma_{32} = 0. \tag{3.42}$$

Finally, the equations for vorticity and divergence of velocity must be updated with the transformed derivatives. This step simply changes the default function derivatives to the transformed derivatives we defined as scalar expressions. With these changes in place, the underlying physics will be in agreement with the coordinate transformation.

3.2.5.2 Boundary Equations

In addition to the subdoimain equations, changes to the boundary equations for the surface need to be made to account for the coordinate transformation. Since COMSOL calculates surface normal vectors based on the geometry defined, our transformed domain would lead to incorrect normals. In order to overcome this circumstance, we must compute normal vectors for the true surface, calculated as follows,

$$s_s = \left(\frac{ds}{dx}\right)/L, \tag{3.43}$$

$$ns'_x = -\frac{s_s}{\sqrt{1+s_s^2}}, \tag{3.44}$$

$$ns'_z = \frac{1}{\sqrt{1+s_s^2}}. \tag{3.45}$$

where s_s is the slope of the surface, and ns'_x and ns'_z are the x and z component of the vector normal to the surface.

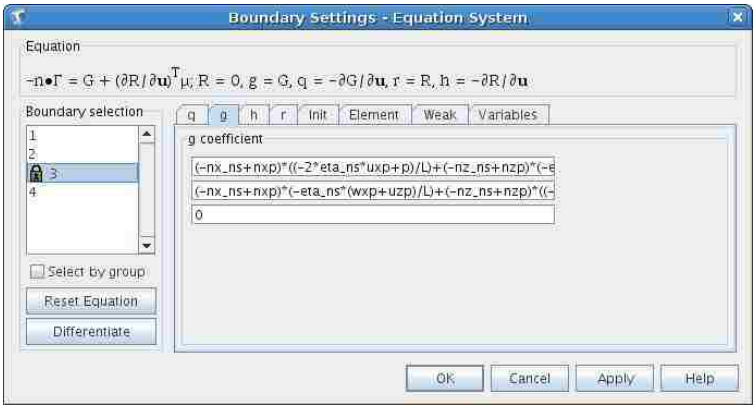


Figure 3.9 The dialog where boundary equation changes are made.

Once the normals vectors are calculated, they must be added to the boundary equations dialog box, shown in figure 3.9. The term of interest in the equation system

is **G**, which must be modified if the model is to reflect the proper surface normals. The modifications needed for correct surface normals are

$$G_1 = (-ns_x + ns'_x)((-2\eta\frac{\partial u}{\partial x'} + p)/L) + (-ns_z + ns'_z)(\eta(\frac{\partial u}{\partial z'} + \frac{\partial w}{\partial x'})/H), \quad (3.46)$$

$$G_2 = (-ns_x + ns'_x)(\eta(\frac{\partial w}{\partial x'} + \frac{\partial u}{\partial z'})/L) + (-ns_z + ns'_z)((-2\eta\frac{\partial w}{\partial z'} + p)/H), \quad (3.47)$$

$$G_3 = 0. \quad (3.48)$$

For simulations that require basal sliding, changes must also be made to the boundary at the bed. The same problem of miscalculated normals present in the surface exist at the bed. A similar approach must be employed in finding the bed normals, calculated as

$$b_s = \left(\frac{db}{dx}\right)/L, \quad (3.49)$$

$$nb'_x = -\frac{b_s}{\sqrt{1+b_s^2}}, \quad (3.50)$$

$$nb'_z = \frac{1}{\sqrt{1+b_s^2}}. \quad (3.51)$$

where b_s is the slope of the bed, and nb'_x and nb'_z are the x and z component of the

vector normal to the bed. The changes to **G** are as follows

$$G_1 = \eta n b_z (\frac{\partial u}{\partial z'} + \frac{\partial w}{\partial x'})/H + \eta(nb_x'^2 - nb_z'^2)(\frac{\partial u}{\partial z'} + \frac{\partial w}{\partial x'}) - \ldots \tag{3.52}$$

$$2nb_x' nb_z' (2\eta \frac{\partial u}{\partial x'}) - (-unb_z' + wnb_x')\beta^2,$$

$$G_2 = -nb_z(-2\eta \frac{\partial w}{\partial z'} + p)/H + \eta(nb_x'^2 - nb_z'^2)(\frac{\partial u}{\partial z'} + \frac{\partial w}{\partial x'}) - \ldots \tag{3.53}$$

$$2nb_x' nb_z' (2\eta \frac{\partial u}{\partial x'}) - (-unb_z' + wnb_x')\beta^2,$$

$$G_3 = 0. \tag{3.54}$$

These changes complete the COMSOL implementation.

CHAPTER 4 RESULTS

4.1 Verification

It is critical that we take measures to ensure our model is working correctly. Traditionally, this is done by comparing model results with observed data. However, that would require high resolution historical data spanning 100,000 years. In the absence of physical data for comparison, we must rely on established solutions. Current criterion exist in the form of a set of structured experiments. The research presented here focuses on verification using the Ice Sheet Model Intercomparison Project - Higher-Order Models.

4.1.1 ISMIP-HOM

In order to establish benchmarks and detect weaknesses in the numerical methods of higher-order ice sheet models, the Ice Sheet Model Intercomparison Project - Higher-Order Models (ISMIP-HOM) was released in early 2006. The project is a set of experiments released to the public - any modeler who wanted to participate was welcome. Though each experiment was specified in detail, each participant was free to use techniques of their choice in reaching a solution – as long is it was considered a higher-order model. The experiments are accessible for many types of models, valid for both finite element and finite difference methods, and are independent of grid type (regular or irregular) [Pattyn and Payne, 2006].

Twenty contributors submitted results, which were published in The Cryosphere Discussions in 2008. Of the twenty participants, twelve used full Stokes models. These models showed a particularly strong agreement with each other as well as with analytical solutions[1] [Pattyn et al., 2008]. This makes these experiments an excellent resource for verifying the correctness of this work. Due to the success of the ISMIP-HOM, the experiments have become a valuable resource for verification. This research will primarily be focused on verification with one of the submittors, Dr. Jesse Johnson, due to the availability of both his model and data.

4.1.1.1 Experiment B

Experiment B considers a slab of ice with mean thickness $H = 1000$ with a linearly sloping surface. The basal topography consists of a series of sinusoidal bumps. The geometry is defined as

$$z_s(x, y) = -x \tan(\alpha), \tag{4.1}$$

$$z_b(x, y) = z_s(x, y) - 1000 + 500 \sin(\omega x), \tag{4.2}$$

where $\alpha = 0.5°$ and $\omega = 2\pi/L$. The experiment is performed for domains of increasing length, with $L = 5$, 10, 20, 40, 80, and 160 km.

Periodic boundary conditions are implemented at the edges, with the sides and the surface being open, stress free boundaries. The ice is frozen to the bedrock in this experiment, corresponding to no velocity ($\mathbf{v}_b = 0$) at the bed. An example domain where $L = 40{,}000$ m is shown in figure 4.1.

The surface velocity for all length scales is shown in figure 4.2. There is a high

[1] An analytical solution exists for only one of the experiments, which was not considered in this work.

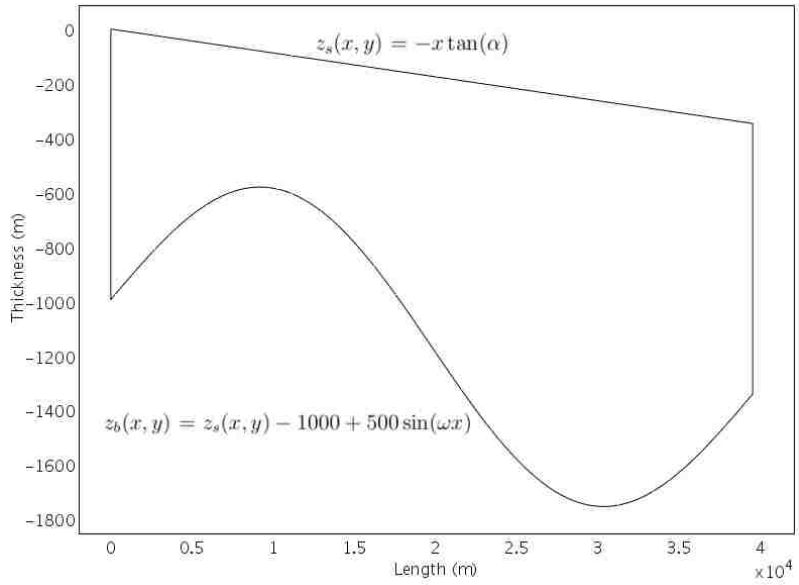


Figure 4.1 An example domain for ISMIP-HOM experiment B with $L =$ 40,000 m.

level of variability in the solutions for shorter domains of $L = 5$ and $L = 10$ km. However, the differences present are influenced by the type of model used. The full Stokes models, denoted as `FS` in the legend, agree strongly with each other. As the horizontal extent of the topography increases, the variability in solutions reached by different models decreases.

In addition to the variability in solutions, another interesting outcome is present in the domain of length $L = 5$ km. While experiments conducted on domains of greater length show a strong correlation in the qualitative shape of the curves, that is not the case with the 5 km domain. While the results from full Stokes models are tightly clustered, there is a much higher level of variability in the non-full Stokes solutions. The extrema shown by the group of models denoted as `NFS` are considerably different than that of the majority of full Stokes and other non-full Stokes models alike. This sort of discrepancy between models appears unique to the dimensions of this particular

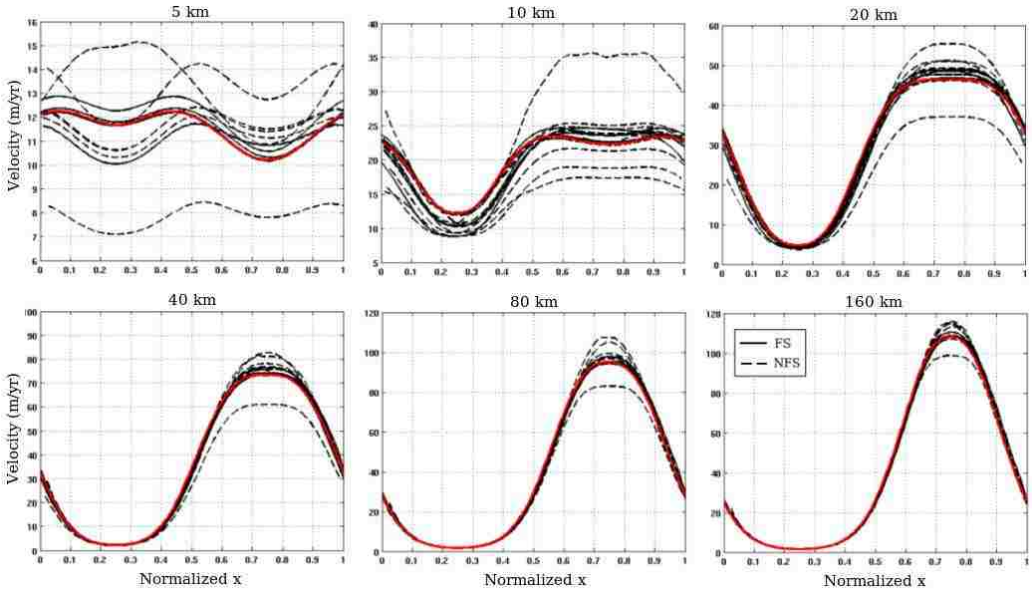


Figure 4.2 Results for ISMIP-HOM experiment B. Surface velocity for different length scales L [Pattyn et al., 2008]. Our results shown in red.

domain.

Surface velocity for all domain lengths produced by our model is shown in red in figure 4.2. A cursory glance reveals that our results are equivalent to those produced by full Stokes models. However, as with the published data, intricacies in the results are found in the smaller domains. For domains greater than $L = 10$ km, our results are highly similar. The velocities on the 10 km domain are also quite similar, with only a small dip near $x = 0.75$ that appears slightly more exaggerated on the transformed geometry. The same pronounced dip in velocity is present on the 5 km domain. However, our results are tightly clustered with other full Stokes solutions.

An emerging trend suggests that domains with a large aspect ratio are more difficult to model. It is promising that full Stokes models are reaching nearly the same solutions. However, the other groups of models are also computing results in agree-

ment with each other, though at times contrasting with full Stokes solutions. Without an analytical solution, we are left to speculate as to which solution is more believable or realistic. Answering this question remains a priority in the field.

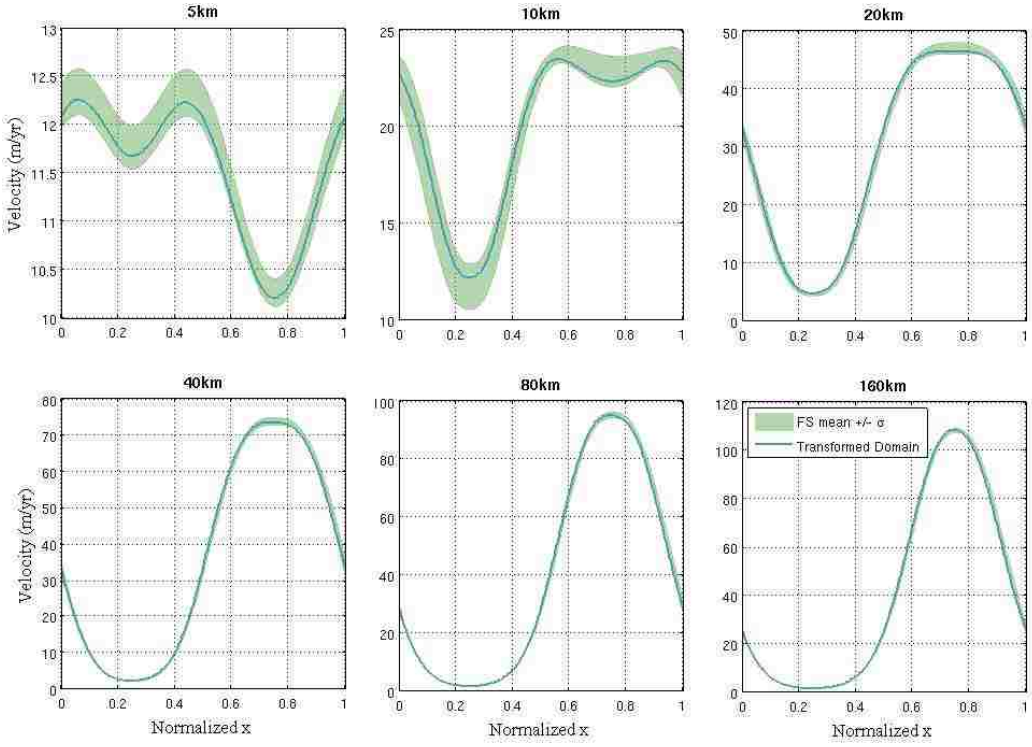


Figure 4.3 ISMIP-HOM results for experiment B. Surface velocity for different length scales L [Pattyn et al., 2008]. Shaded area represents one standard deviation from the mean from full Stokes models. Our results shown in blue.

Figure 4.3 represents a more quantitative look at the results. The shaded area depicts one standard deviation from the mean of all full Stokes models. As expected, the smaller domains of $L = 5$ and $L = 10$ km show the greatest variability. As the domain length increases, the shaded region becomes smaller, becoming nearly non-visible for the largest domains. Our results are shown in blue, located inside the shaded region for all domain lengths. This demonstrates a strong agreement between

our solution and the mean of full Stokes contributors to the ISMIP-HOM.

In addition to velocity, pressure is computed everywhere in the ice mass. However, it turns out that pressure is not a very interesting result. This doesn't come as a surprise, as our model is concerned with deviatoric stress, where influence from pressure is removed. Figure 4.4 shows pressure at the bed for $L = 40$ km for both the true and transformed domains. Pressure is identical in both instances. In fact, pressure is always the same regardless of domain length. Researchers in the field are primarily interested in flow, so the focus for the remainder of the work will be on velocity.

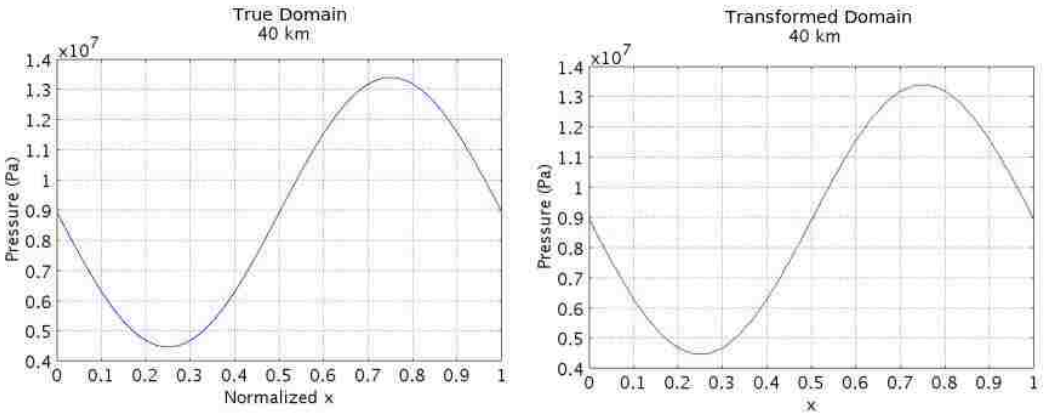


Figure 4.4 Pressure at the bed for ISMIP-HOM experiment B from both the true and transformed domain with $L = 40$ km.

With our focus narrowed, we investigate the full velocity field rather than a cross section. Figure 4.5 depicts the results from Johnson's model. A trend is clearly visible; velocity is fastest where the ice is thickest and slows in thin areas. We also see a velocity gradient present that increases from the bed up towards the surface, due to ice frozen to the bedrock. Velocity increases along with horizontal extent, however not in a linear fashion, summarized in table 4.1.

Table 4.1 Maximum Velocity for ISMIP-HOM Experiment B (m/yr)

	True Domain	Transformed Domain
$L = 5$ km	12.261	12.252
$L = 10$ km	24.553	24.286
$L = 20$ km	47.564	47.481
$L = 40$ km	73.828	73.801
$L = 80$ km	94.857	95.052
$L = 160$ km	108.021	108.468

The same trend present in the solution from Johnson's model appears in our results. This is positive confirmation that our transformed domain is "feeling" the effects of the topography. Our results are almost identical, both qualitatively and quantitatively, disagreeing by only tenths of a meter per year. Interestingly, the differences found in the surface velocity for the 5 km domain are not as obvious in the full velocity field. Across all lengths of domains, the full velocity fields tend to agree slightly better than comparing only surface velocity. This implies we might need to employ more careful treatment with respect the boundary conditions at the surface. Figure 4.6 shows the velocity fields for the transformed domain.

Our results for ISMIP-HOM experiment B are encouraging; our model produces output that is in agreement with established solutions. It appears a limitation of our model exists for domains of high aspect ratio. It is difficult to reach that conclusion though, as it is unclear how to measure the accuracy of our outcome when no analytical solution exists. It is entirely possible that our solution is more realistic than the current benchmarks, but also possible our results are less plausible. Furthermore, the differences in surface velocity between our model and others also give us cause for concern. However, it is clear that these perceived shortcomings are minor and our model is performing well.

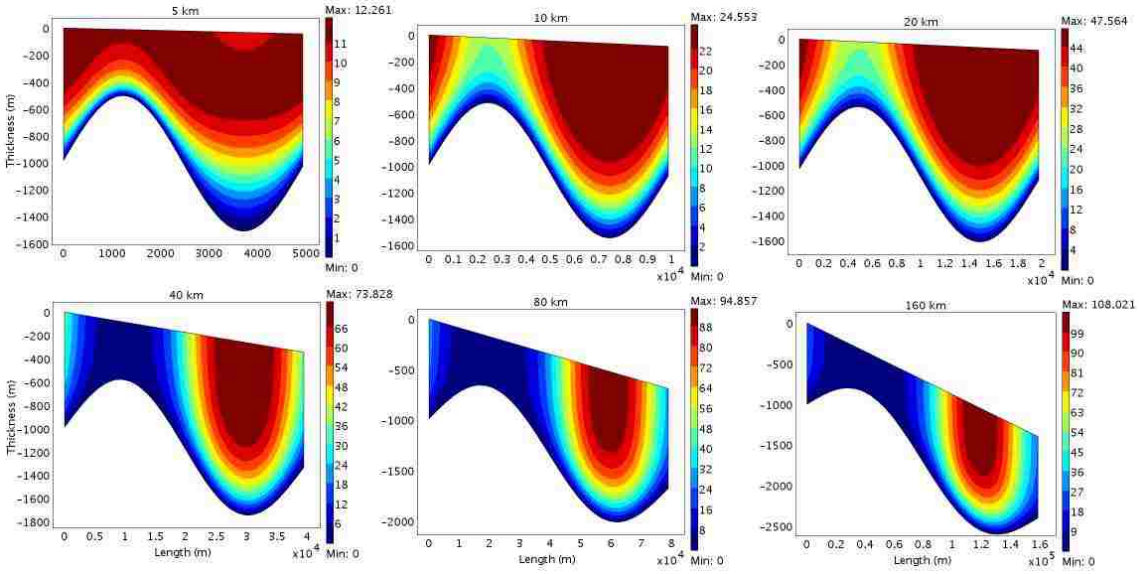


Figure 4.5 Velocity field (m/yr) results from ISMIP-HOM experiment B with Johnson's model for different length scales L.

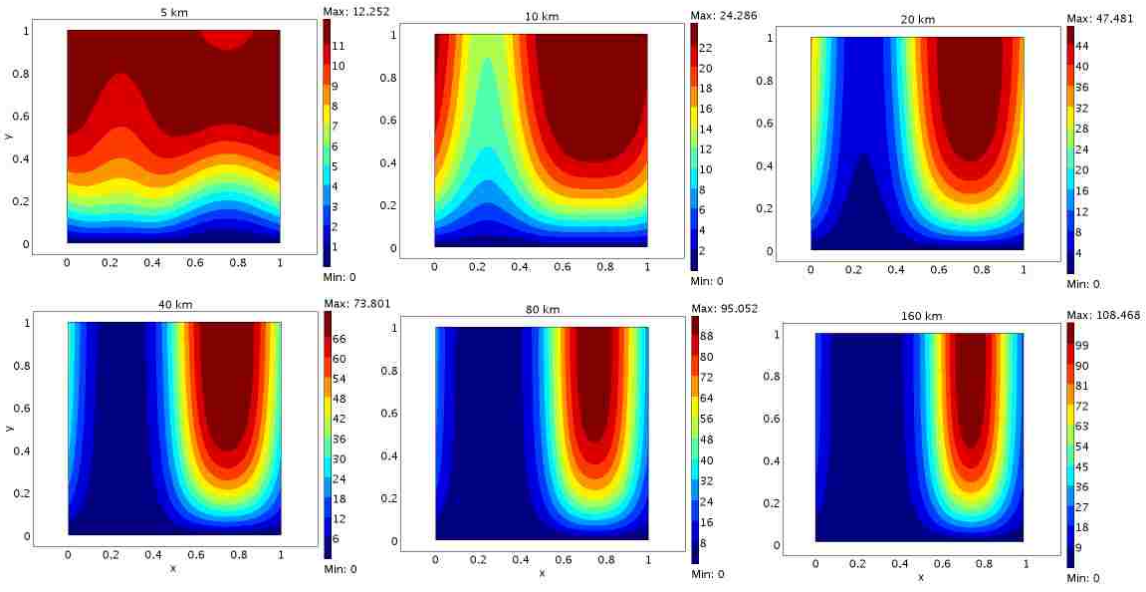


Figure 4.6 Velocity field (m/yr) results from ISMIP-HOM experiment B with the transformed domain for different length scales L.

4.1.1.2 Experiment D

Experiment D involves ice flow over simple topography where the surface and bed are linear and parallel. The surface and bed are described by

$$z_s(x, y) = -x \tan(\alpha), \tag{4.3}$$

$$z_b(x, y) = z_s(x, y) - 1000. \tag{4.4}$$

Again, periodic boundary conditions exist at the edges and the surface is an open boundary. What is unique about this experiment is the ice is allowed to slide over the bed. The basal friction field is defined as

$$\beta^2 = 1000 + 1000 \sin(\omega x). \tag{4.5}$$

where $\omega = 2\pi/L$. A high value for β^2 corresponds to a large amount of friction between the ice and bedrock.

Whereas the previous exercise was concerned with the effect of topography on the flow of ice, this experiment is interested in the influence of basal sliding. The domain, shown in figure 4.7, is a simplistic one. By eliminating surface and bed features, the effects of sliding are isolated. Where the friction is very high, the ice is nearly frozen to the bed. However, when the friction is low we expect the ice to speed up dramatically in that region.

The maximum velocity found by both Johnson's and our model is shown in figure 4.8 and summarized in table 4.2. Unlike the previous experiment, there is large amount of discrepancy in the outcomes. Other than the maximum velocity increasing with length of domain, there is no common trend in the data produced by the two models. Excluding the 40 and 160 km domains, our model predicted higher velocities

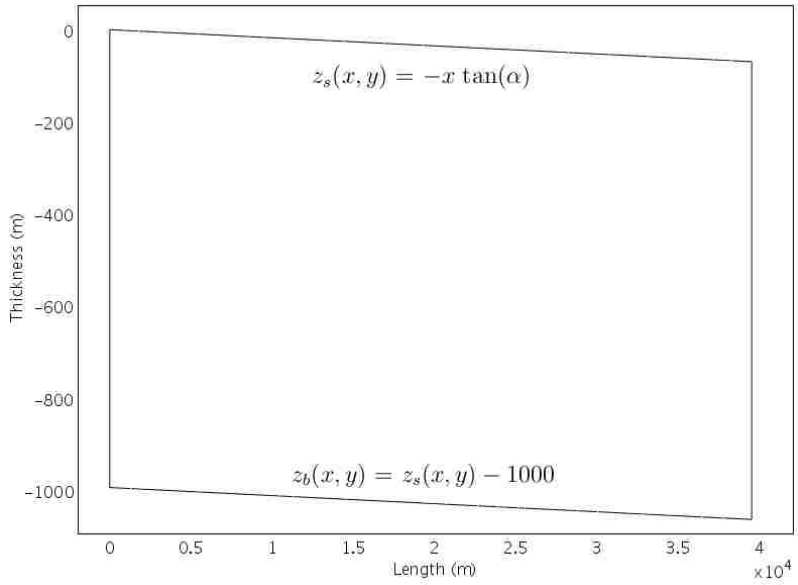


Figure 4.7 An example domain for ISMIP-HOM experiment D with $L =$ 40,000 m.

than Johnson's. However, the results from our model implies there is an asymptotic limit to the velocity whereas Johnson's solution implies no such bound. Our data seem more consistent with the results from experiment B, which also showed velocities approaching an upper limit.

Curiously, the two models reach almost identical solutions for the 40 km domain. Johnson's model reaches this outcome with a considerable speed increase from the

Table 4.2 Maximum Velocity for ISMIP-HOM Experiment D (m/yr)

	True Domain	Transformed Domain
$L = 5$ km	8.559	16.438
$L = 10$ km	9.04	20.672
$L = 20$ km	12.284	28.373
$L = 40$ km	40.969	40.259
$L = 80$ km	61.202	96.075
$L = 160$ km	150.272	105.94

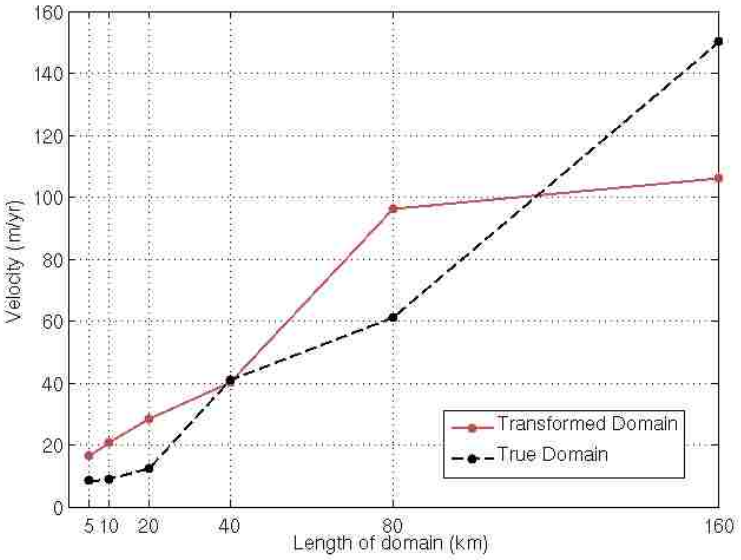


Figure 4.8 Maximum velocity for ISMIP-HOM experiment D for both the true and transformed domain for all lengths L.

20 km domain, equivalent to a 234% increase. Our model, however, depicts only a 42% increase for the same domain change. Due to the large variability in the other data points, the near perfect correlation for the 40 km domain is likely attributed to coincidence.

Figure 4.9 shows the surface velocities for all domain lengths. There is a much higher level of variability present in this experiment than found in experiment B, both between the different types of models and in results obtained from the same model type. This implies that different schemes exist for handling sliding, which produce contrasting outcomes. Our results, shown in red, appear to blend in well with the ISMIP-HOM contributors in all domains except $L = 160$ km. From analyzing the 160 km domain, it is clear that our solution is wrong for that domain.

Taking a closer look at how our results compare to the modelers of the ISMIP-HOM, figure 4.10 shows our results compared with the mean of all full Stokes models. In

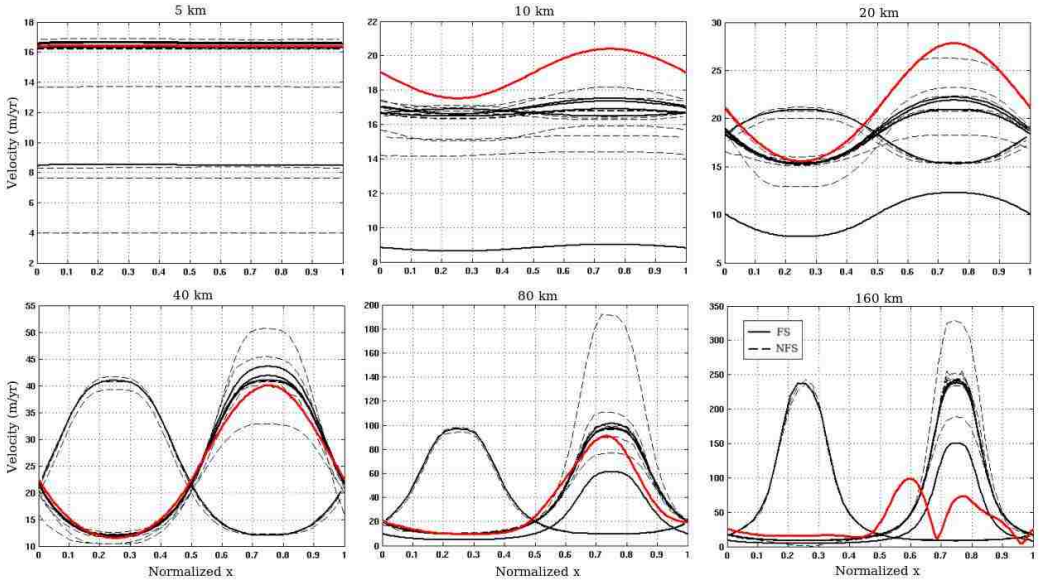


Figure 4.9 Results for ISMIP-HOM experiment D. Surface velocity for different length scales L [Pattyn et al., 2008]. Our results shown in red.

contrast to experiment B, the shaded area representing one standard deviation from the mean is quite large for all domains. For the domains $L = 5$, $L = 10$, and $L = 80km$, our results (shown in blue) lie inside the shaded area, demonstrating strong agreement with other full Stokes modelers. However, the other three domains show significant portions of our results lying outside of the shaded region. It is clear that the magnitude of the velocities found using our model are inconsistent with the mean of the ISMIP-HOM contributors.

Continuing the comparison between models, next we investigate the similarities and differences in the full velocity fields. Rather than considering all six domains as in experiment B, here we focus on one short and one long domain, the 5 an 80 km specifically. Figure 4.11 shows the velocity field for the true and transformed 5 km domain. Qualitatively, the results are very similar. Both depict a region of slower

velocity due to high friction at the border of the right side and continuing into the left. Furthermore, the streamlines look quite similar for both. Clearly, the spatial element of sliding is being captured by our model while it appears to struggle with the magnitude.

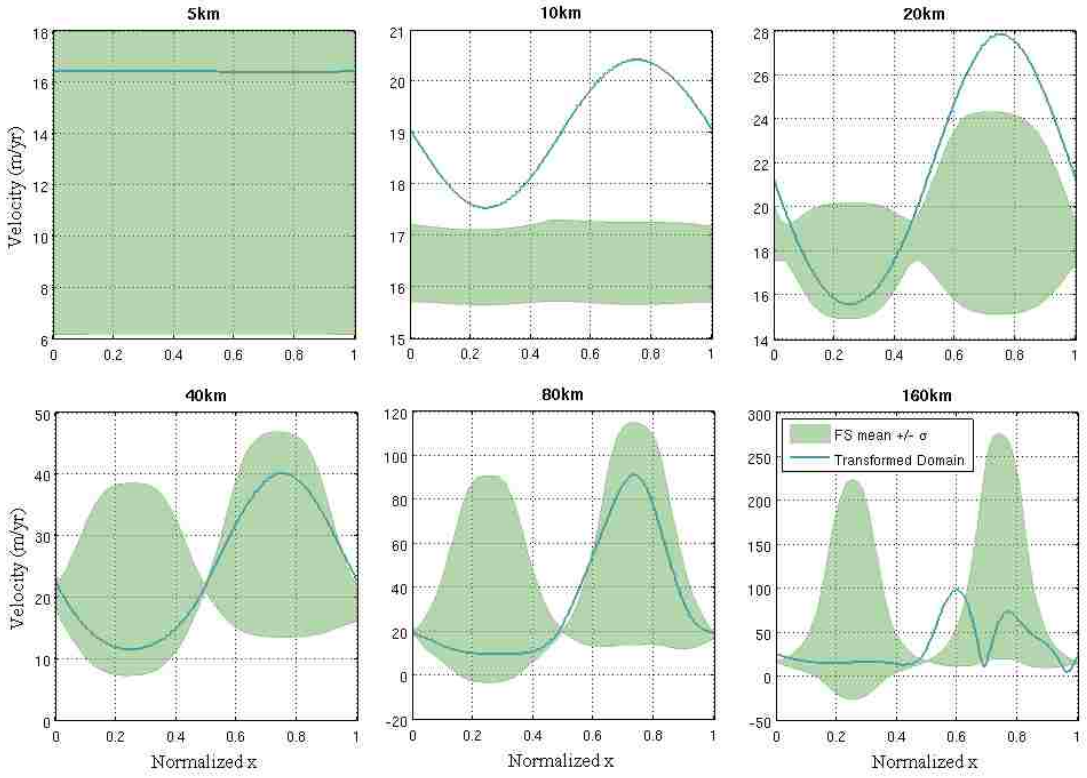


Figure 4.10 ISMIP-HOM results for experiment B. Surface velocity for different length scales L [Pattyn et al., 2008]. Shaded area represents one standard deviation from the mean from full Stokes models. Our results shown in blue.

The results from the 80 km experiment are consistent with those from the 40 km domain. Visually, our results look very similar to Johnson's, with an area of increased velocity near the right side. The streamlines are also fairly agreeable between the two, becoming more and more sinusoidal as the distance from the bed increases. This is further indication that our model is accommodating basal sliding.

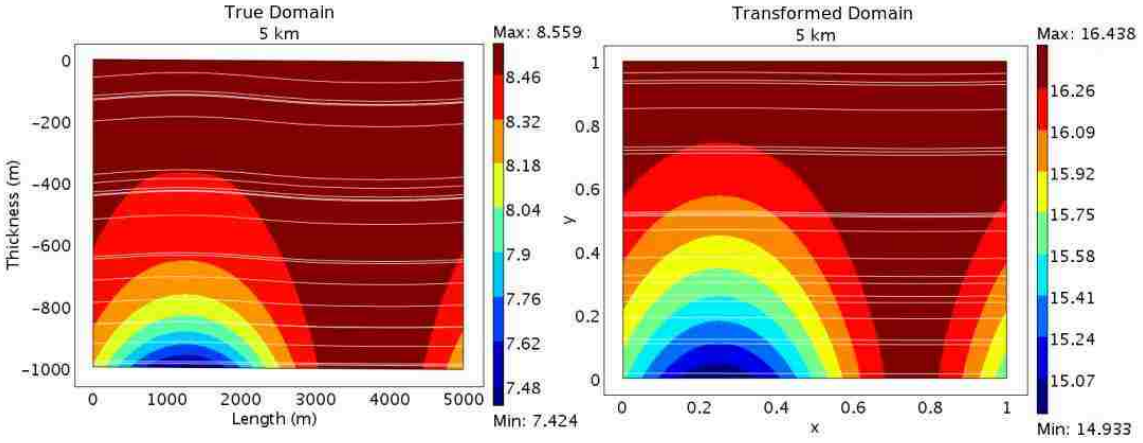


Figure 4.11 Velocity field for ISMIP-HOM experiment D for both the true and transformed domain with $L = 5$ km. Streamlines are indicated in white.

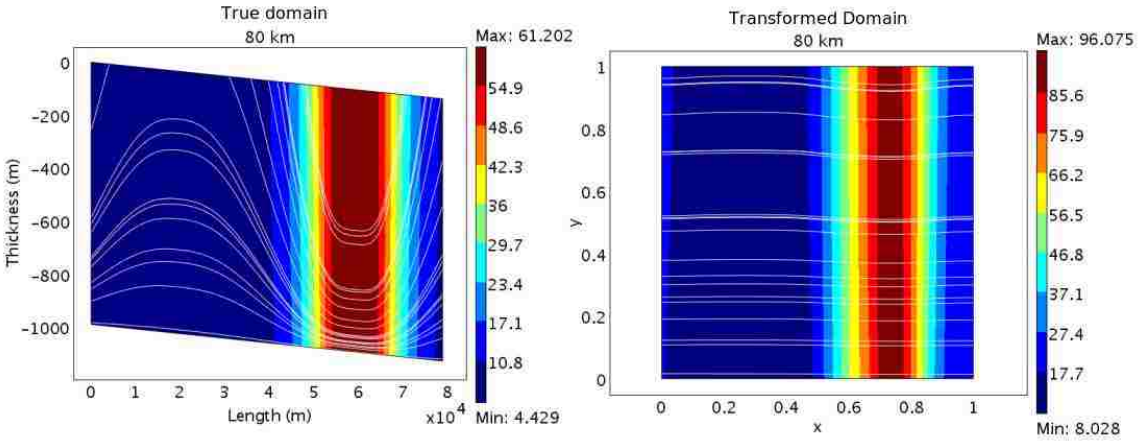


Figure 4.12 Velocity field for ISMIP-HOM experiment D for both the true and transformed domain with $L = 80$ km. Streamlines are indicated in white.

Defining and implementing sliding is still relatively unknown to ice sheet modelers. It was the most difficult aspect of implementation for us, as we were never able to reach convergence with the non-linear solver. However, we were able to reach a solution by setting the dynamic viscosity to a realistic constant. There are two possible reasons for this. First, COMSOL is unable to reconcile the numerics on the transformed domain. Second, and more likely, there is an error present in our scheme.

As a result of our simplification of viscosity, our results are to be viewed in a speculative context. By using a constant viscosity, the exact values of the velocity become less meaningful. However, the distribution of velocity across the ice mass predicted by our model is unaffected by the constant viscosity, due to very little deformation occurring when ice is allowed to slide. Our model appears to excel at predicting the spatial and qualitative effects of basal sliding. Ironing out the issues in sliding remains a focus of our work.

4.1.1.3 Experiment E

Experiment E simulates a temperate glacier in the European Alps named Haut Glacier d'Arolla. In this experiment, the surface and bed are governed by external data physically taken from the glacier. The bed has zero basal velocity ($\beta^2 = \infty, \mathbf{v}_b = 0$) and there are no boundary conditions at the edges, where the thickness and velocity are zero. The domain is depicted in figure 4.13.

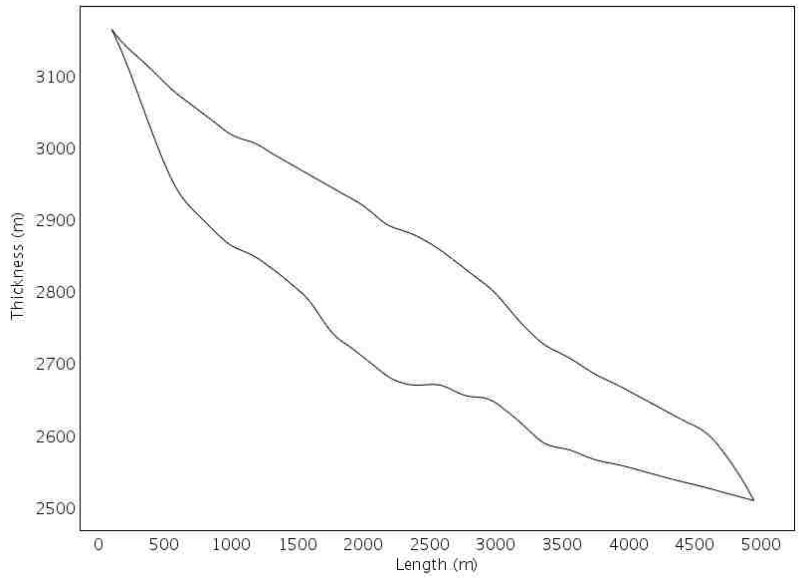


Figure 4.13 The domain for ISMIP-HOM experiment E, Haut Glacier d'Arolla in the European Alps.

The boundary conditions at the edges present a problem for our model. Due to the coordinate transformation, there are scaling terms of $1/H$ occurring numerous times in the equation system. This creates a conflict when there are areas of ice with zero thickness, as dividing by zero often causes problems with computer simulations. Rather than modifying our model by adding a small epsilon value to the numerous occurrences of $1/H$, we modified the data slightly so there were no locations in the

ice mass with zero thickness.

Figure 4.14 shows the surface velocity from the contributors to the ISMIP-HOM with our results in red and blue. There is a strong agreement between the submittors to the ISMIP-HOM, evidenced by the similar surface velocity as well as the narrow shaded area in the graph on the right, denoting one standard deviation from the mean. Our results, while similar, fall outside of one standard deviation for most values of x. This slight difference further suggests an error present in our scheme. In addition, there is an interesting occurrence near the bottom right corner where our model appears to overlook a sharp decrease in velocity, instead showing a smooth reduction. This is minor, however, and most likely attributed to our model using finite thickness at the edges.

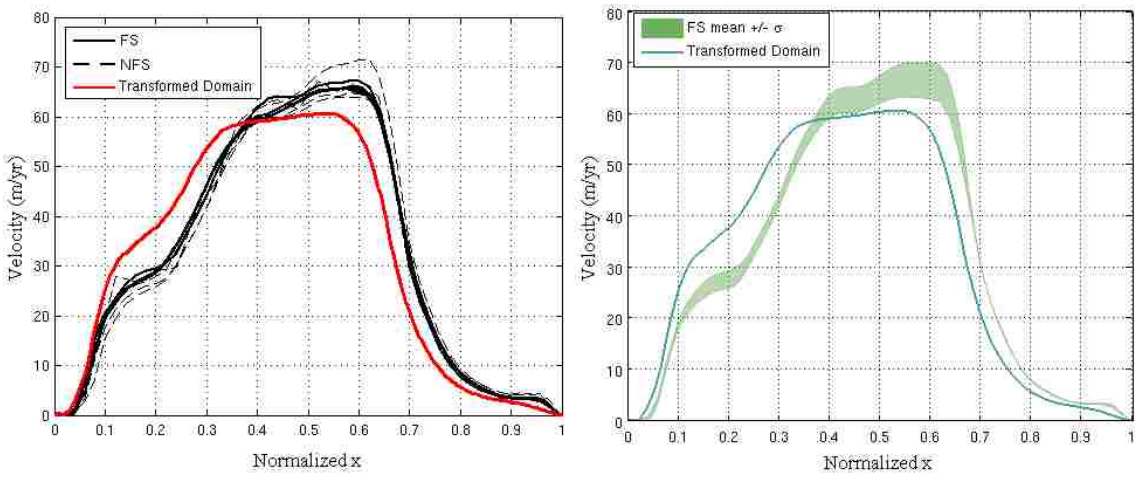


Figure 4.14 Left: Arolla surface velocity (m/yr). Right: Shaded area depicts one standard deviation from the mean [Pattyn et al., 2008]. In both cases, our results are shown in red.

The full velocity fields are displayed in figure 4.15. For this simulation, a reader must really use their imagination to see the similarities in the results. This is a artifact

of the coordinate transformation; one must imagine the solution for the transformed domain stretched considerably in the horizontal. The same velocity distribution is present in both, with the fastest moving ice near the center where it is thickest, then slowing down to nearly zero over the last 20%. Our model is predicting slightly slower values than Johnson's model, particularly in higher velocity areas.

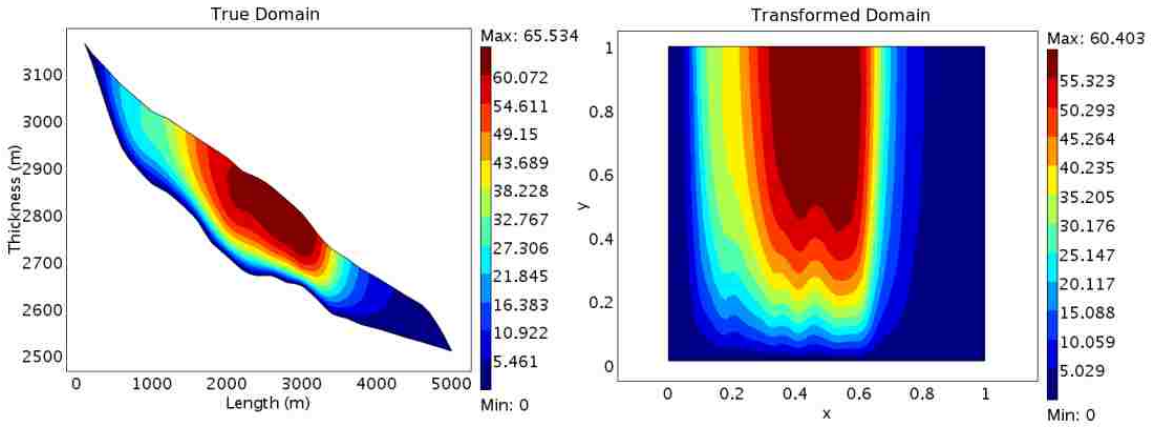


Figure 4.15 Comparison of Arolla velocity field (m/yr) for both the true and transformed domains.

Successfully modeling this scenario is a promising result. If our model was unable to handle geometry measured from real glaciers, one would be entirely justified in questioning its value. However, we have demonstrated the capability of modeling domains existing in nature, a necessary step if we are to model areas in Greenland and Antarctica in the future. As with the previous two experiments, there are slight inconsistencies in our solutions which may be attributed to an error in our implementation – such as our treatment of boundary conditions – or may be of no consequence.

CHAPTER 5 CONCLUSIONS AND FUTURE DIRECTIONS

5.1 Conclusions

The verification of our model around a structured set of experiments has demonstrated its ability to handle a range of different scenarios. It excelled at modeling the simple topography of experiment B, producing results that match our conceptual understanding of ice flow as well as the efforts of the ice sheet modeling community. Basal sliding proved a more difficult dynamic to capture, but our results are promising. It appears that minor changes are all that is necessary to remedy the shortcomings. Lastly, implementing the Arolla experiment highlighted the ability of our model to handle realistic topography. It is clear from the analysis that our model is performing well.

In addition to the strengths of our model, we have learned several weaknesses. This knowledge is almost more important, as it provides focus for improvement. The major flaw in our model occurs when ice is allowed to slide over the bed. While our results show realistic velocity distribution, we were unable to reach a solution using a nonlinear viscosity. The model would come tantalizingly close to convergence, suggesting that only a small error persists.

The results from verification also indicate other errors might exist. We would expect our results to match closely with those achieved by other full Stokes models, though

our solutions were slightly different, specifically in the Arolla experiment. This may be attributed to COMSOL being unable to reconcile the numerics on the transformed domain, leading to slight deviations in outcome. However, our results suggest that the source is our treatment of the boundary conditions. Specifically, in the translation between the true and transformed normal vectors and the required modification to the equation system.

In light of the shortcomings, the most important conclusion is a positive one. Our model has demonstrated the ability to handle various scenarios using a transformed coordinate system. This type of scheme will ultimately allow for greater power and flexibility, as the uniform dimensions of the geometry reduces variability and produces a superior triangular mesh. This is especially pivotal for performing time dependent experiments, where the domain is free to evolve over time. Our efforts lay the groundwork for these types of simulations in the future.

5.2 Future Directions

Undoubtedly, there are many avenues for future work. The work described here represents our first efforts, the initial step in an iterative process of development. However, the logical next step would be a careful review and analysis of the numerics of our model and the resulting implementation. This would be most beneficial with an emphasis on improving the boundary conditions and rectifying the mechanics of basal sliding.

In addition, it would be helpful and beneficial to "un-transform" the results for visualization purposes. In other words, the results would be viewed on the true geometry, rather than the unit square. This would make results much clearer, as it would be easier to draw conclusions in the proper geometric context. Furthermore,

one could perform a quantitative comparison between our results and others if they were on a similar grid. Our current comparison methods of full velocity fields are strictly observational.

Beyond developing solutions to existing shortcomings, future work should focus on extending the applicability of the model. Though our work focused on two-dimensional experiments, it is critical to have a model capable of three-dimensional simulations. The transition to three-dimensional domains appears to be straightforward, as the numerics of our model are specified generically and are easily extended.

Finally and most importantly, our model must be extended to work with experiments of free surface evolution. Running time dependent simulations is critical, especially in terms of predicting sea level change. In order to achieve this goal, our flow model must coupled with thermodynamic elements. Fortunately, adding the effects of heat is considerably easier than solving for flow. This step is currently being developed, and promising progress is being made.

BIBLIOGRAPHY

Comsol multiphysics. World Wide Web electronic publication. URL http://www.comsol.com/products/multiphysics/. Accessed on 04/20/08.

Vivien Gornitz. Coastal populations, topography, and sea level rise. World Wide Web electronic publication, 2000. URL http://www.giss.nasa.gov/research/briefs/gornitz_04/.

Emmanuel Le Meur, Olivier Gagliardini, Thomas Zwinger, and Juha Ruokolainen. Glacier flow modelling: a comparison of the shallow ice approximation and the full-stokes solution. *Comptes Rendus Physique*, 5(7):709–722, 2004.

Bjorn Lund. Properties of the stress tensor. World Wide Web electronic publication, 2000. URL http://www.geofys.uu.se/bl/Avh/node6.html.

F. Pattyn, L. Perichon, A. Aschwanden, B. Breuer, B. de Smedt, O. Gagliardini, G. H. Gudmundsson, R. Hindmarsh, A. Hubbard, J. V. Johnson, T. Kleiner, Y. Konovalov, C. Martin, A. J. Payne, D. Pollard, S. Price, M. Rckamp, F. Saito, O. Soucek, S. Sugiyama, and T. Zwinger. Benchmark experiments for higher-order and full stokes ice sheet models (ismip-hom). *The Cryosphere Discussions*, 2(1):111–151, 2008. ISSN 1994-0432. URL http://www.the-cryosphere-discuss.net/2/111/2008/.

Frank Pattyn. A new 3D higher-order thermomechanical ice-sheet model: Basic sensitivity, ice-stream development and ice flow across subglacial lakes. *J. Geophys. Res.*, 108(B8, 2382):doi:10.1029/2002JB002329, 2003.

Frank Pattyn. 2d higher-order flowline model (ice2d). World Wide Web electronic publication. URL `http://homepages.ulb.ac.be/~fpattyn/2dhomodel.html`.

Frank Pattyn and Tony Payne. Benchmark experiments for numerical higher-order ice-sheet models. Located at http://homepages.ulb.ac.be/ fpattyn/ismip/, 2006.

Eric Rignot and Pannir Kanagaratnam. Changes in the velocity structure of the greenland ice sheet. *Science*, 311(5763):986–990, February 2006. doi: 10.1126/science.1121381. URL `http://dx.doi.org/10.1126%2Fscience.1121381`.

Eric Rignot, Jonathan L. Bamber, Michiel R. van den Broeke, Curt Davis, Yonghong Li, Willem J. van de Berg, and Erik van Meijgaard. Recent antarctic ice mass loss from radar interferometry and regional climate modelling. *Nature Geosci*, 1 (2):106–110, February 2008. doi: 10.1038/ngeo102. URL `http://dx.doi.org/10.1038%2Fngeo102`.

Andrew Shepherd and Duncan Wingham. Recent sea-level contributions of the antarctic and greenland ice sheets. *Science*, 315(5818):1529–1532, March 2007. doi: 10.1126/science.1136776. URL `http://dx.doi.org/10.1126%2Fscience.1136776`.

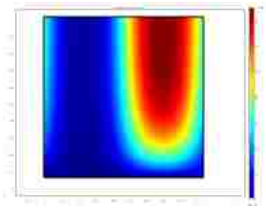
James G. Titus, Richard A. Park, Stephen P. Leatherman, Weggell J. Richard, Michael S. Greene, Paul W. Mausel, Scott Brown, Gary Gaunt, Manjit Trehan, and Gary Yohe. Greenhouse effect and sea level rise: The cost of holding back the sea. *Coastal Management*, 19:171–204, 1991.

C.J. Van Der Veen. *Fundamentals of Glacier Dynamics*. A.A. Balkema, 1999.

APPENDIX A Model Reports

The following pages contain model reports generated by COMSOL for each of the three ISMIP-HOM experiments.

Domain transformation



1. Table of Contents

2. Model Properties

Property	Value
Model name	Domain transformation
Author	Jesse Johnson and James Fishbaugh
Company	University of Montana
Department	Computer Science
Reference	
URL	
Saved date	Mar 18, 2008 12:11:20 PM
Creation date	Jan 25, 2008 11:40:16 AM
COMSOL version	COMSOL 3.4.0.248

File name: /home/jfishbaugh/documents/research/ISMIP-HOM/expBtransform/2D/expBtransform_2D_40.mph

Application modes and modules used in this model:

- Geom1 (2D)
  - Incompressible Navier-Stokes

2.1. Model description

This model uses a coordinate transformation to allow a 1x1 to represent more complicated geometry. This model incorporates a no-slip bed as well as continuous boundaries on the sides. Equations for the surface and bed, along with the length of domain, allow the square domain to mimic the true domain. Transformed velocity and pressure derivatives are used in place of the original derivatives at Physics->Equation System->Subdomain Settings. In addition, transformed surface normals are inserted at Physics->Equation Settings->Boundary Settings.

2.2. Model Result

In the simplest case, change the bed and surface equations as well as the length variable to represent the domain you are interested in modeling. For more complicated simulations, you may have to change boundary condtions as well.

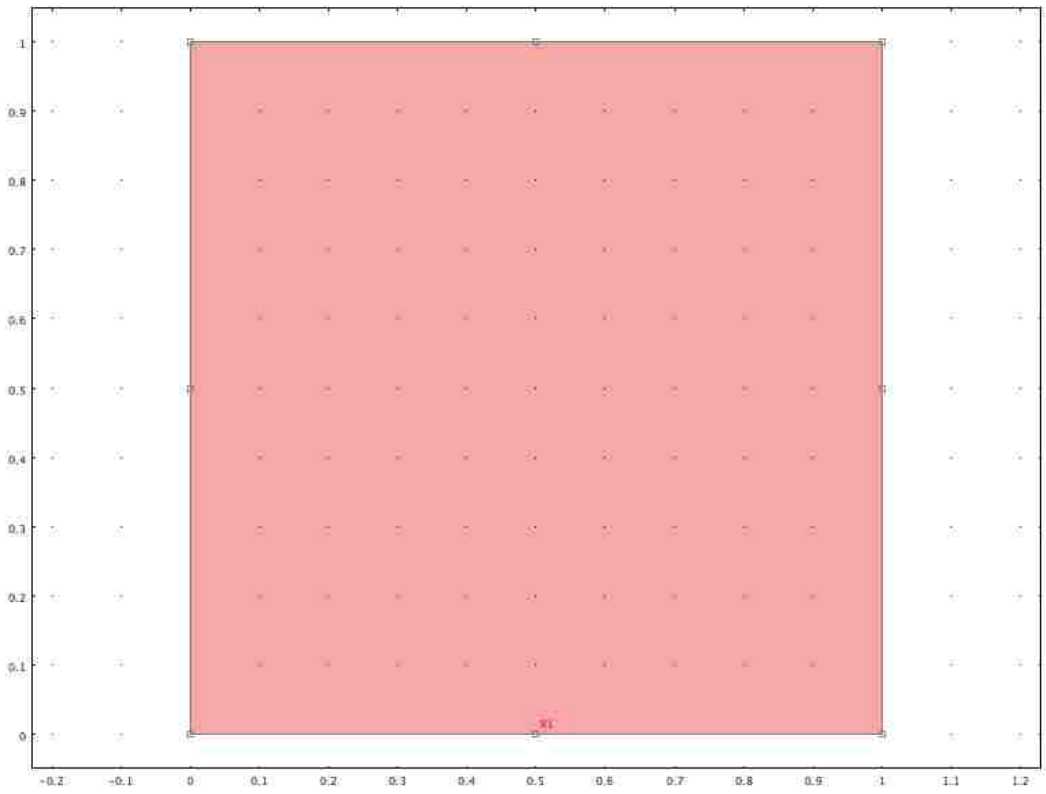
3. Constants

Name	Expression	Value	Description
L	40000		Length of domain (meters)
freq	(2*pi)/L		Frequency
alpha	0.5*(pi/180)		Surface angle with horizontal
rho_ice	910		Density of ice
g	9.81		Gravity
epsilon	1e-18		Prevents numerical explosion in eta_nonlinear if gradients all zero
A	1e-16		Ice 'hardness'
n	3		Glen's flow law exponent

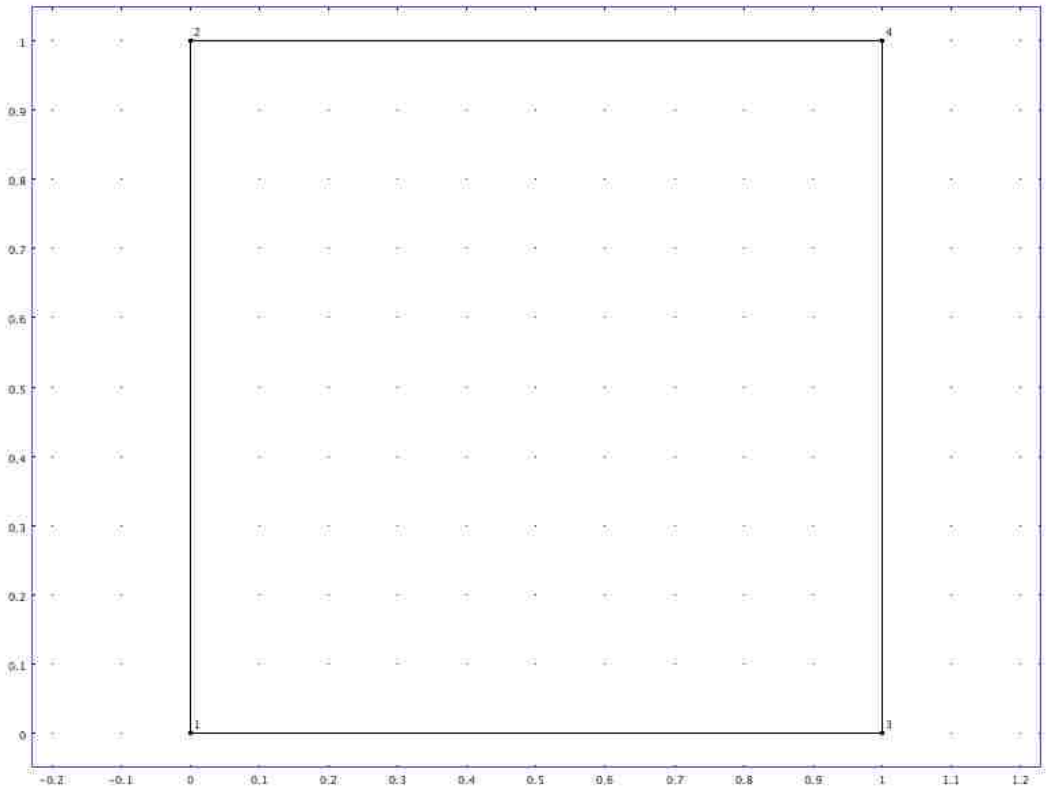
4. Geometry

Number of geometries: 1

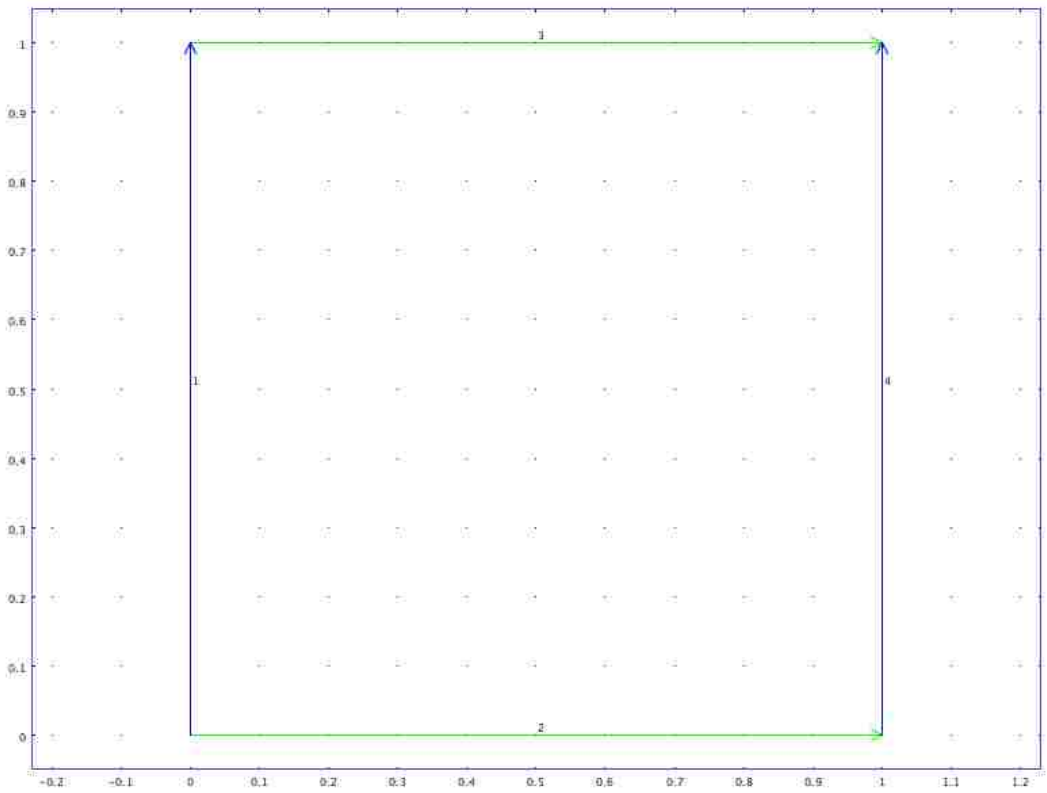
4.1. Geom1



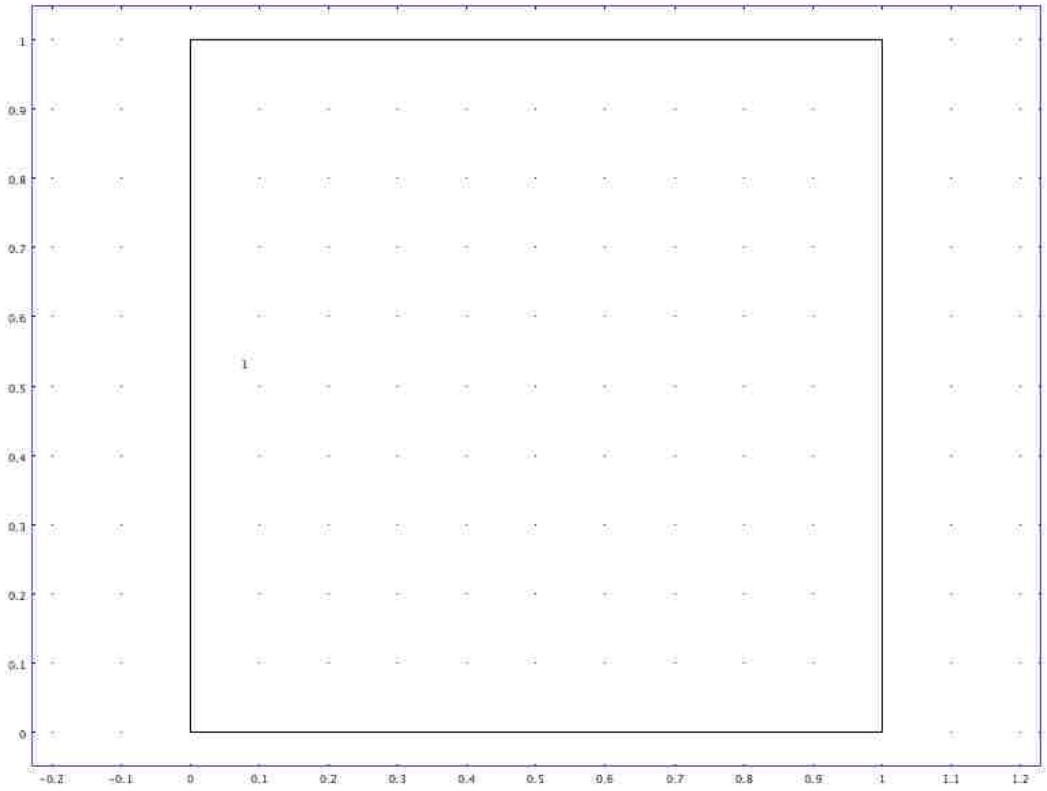
4.1.1. Point mode



4.1.2. Boundary mode



4.1.3. Subdomain mode



5. Geom1

Space dimensions: 2D

Independent variables: x, z, y

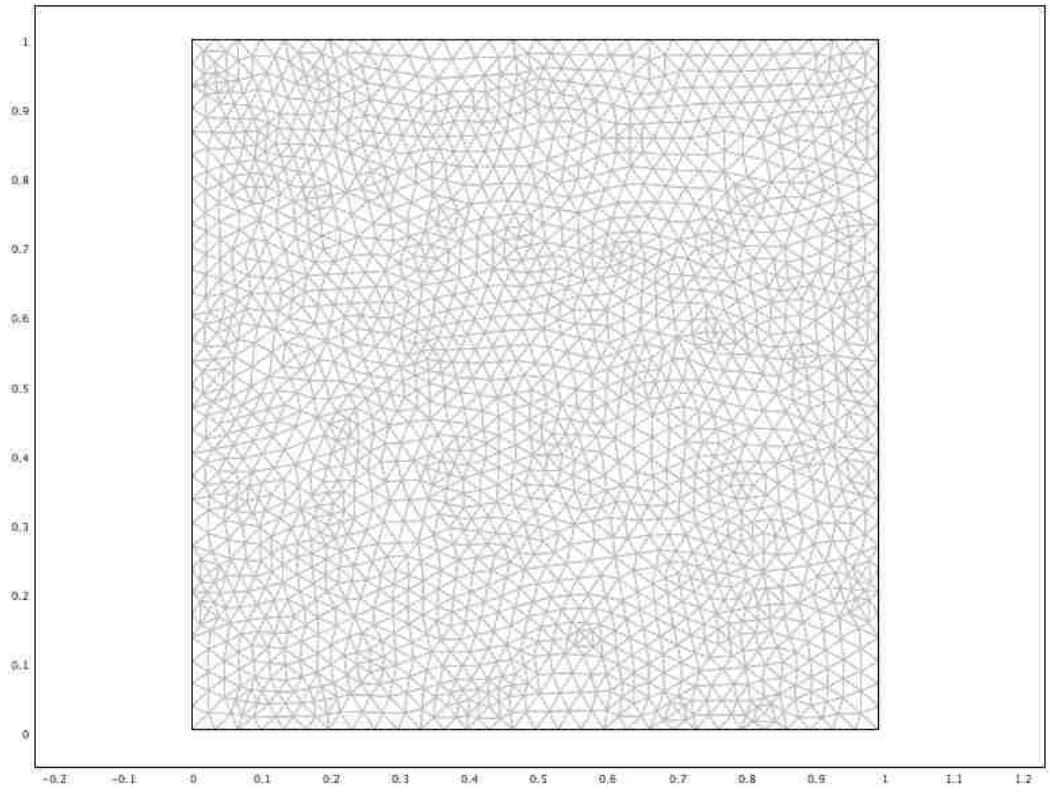
5.1. Scalar Expressions

Name	Expression	Description
bed	surf-1000+500*sin(freq*x*L)	Equation that defines the bed
surf	-(x*L)*tan(alpha)	Equation that defines the surface
eta_nonlinear	1/2*A^(-1/n) * (uxp^2 + .25*(uzp+ wxp)^2+epsilon)^((1-n)/(2*n))	Viscosity of ice
H	surf-bed	Thickness of ice
p0	rho_ice*g*H*(1-z)	Initial pressure
theta	atan(-1/(diff(surf,x)))	Angle used for computing surface normals
ax	-(z*diff(surf,x)+(1-z)*diff(bed,x))/(H*L)	Transformed dz'/dx where z'=(z-bed)/(surf-bed)
uxp	ux/L+ax*uz	Transformed du/dx
wxp	wx/L+ax*wz	Transformed dw/dx
uzp	uz/H	Transformed du/dz
wzp	wz/H	Transformed dw/dz
bx	diff(ax,x)/L+ax*diff(ax,z)	Convienence derivative with respect to x
uxxp	uxx/L^2+2*ax*uxz/L+ax^2*uzz+bx*uz	Transformed d^2u/dx^2
wxxp	wxx/L^2+2*ax*wxz/L+ax^2*wzz+bx*wz	Transformed d^2w/dx^2
uzzp	uzz/H^2	Transformed d^2u/dz^2
wzzp	wzz/H^2	Transformed d^2w/dz^2
uzxp	(uxz+L*ax*uzz-uz*diff(H,x)/H)/(H*L)	Transformed d^2u/dxdz
wxzp	(wxz+L*ax*wzz-wz*diff(H,x)/H)/(H*L)	Transformed d^2w/dxdz
pxp	px/L+ax*pz	Transformed dp/dx
pzp	pz/H	Transformed dp/dz
nxp	sin(theta)	Calculates the x surface normals
nzp	cos(theta)	Calculates the z surface normals

5.2. Mesh

5.2.1. Mesh Statistics

Number of degrees of freedom	17547
Number of mesh points	1977
Number of elements	3832
Triangular	3832
Quadrilateral	0
Number of boundary elements	120
Number of vertex elements	4
Minimum element quality	0.692
Element area ratio	0.204



5.3. Application Mode: Incompressible Navier-Stokes (ns)

Application mode type: Incompressible Navier-Stokes

Application mode name: ns

5.3.1. Application Mode Properties

Property	Value
Default element type	Lagrange - P_2 P_1
Analysis type	Stationary
Corner smoothing	Off
Frame	Frame (ref)
Weak constraints	Off
Constraint type	Ideal

5.3.2. Variables

Dependent variables: u, w, p, nxw, nyw

Shape functions: shlag(2,'u'), shlag(2,'w'), shlag(1,'p')

Interior boundaries not active

Locked Boundaries: 3

5.3.3. Boundary Settings

Boundary	1, 4	2	3
Type	Open boundary	Wall	Open boundary
opentype	ntotstress	**novisc**	ntotstress
stresstype	totstress	totstress	**ntotstress**

5.3.4. Subdomain Settings

Locked Subdomains: 1

Subdomain	1
Integration order (gporder)	**4 4 2**
Constraint order (cporder)	**2 2 1**
Density (rho)	**0**
Dynamic viscosity (eta)	**eta_nonlinear**
Volume force, z-dir. (F_y)	**-rho_ice*g**

6. Periodic Conditions

6.1. Geom1

6.1.1. Source Boundary: 4

Name	Value
Expression	u
Transformation type	Linear
Destination, Boundary 1 (Geom1)	u
Source vertices	4, 3
Destination vertices	2, 1
Name	ucon

6.1.2. Source Boundary: 4

Name	Value
Expression	w
Transformation type	Linear
Destination, Boundary 1 (Geom1)	w
Source vertices	4, 3
Destination vertices	2, 1
Name	wcon

6.1.3. Source Boundary: 4

Name	Value
Expression	p

Transformation type	Linear
Destination, Boundary 1 (Geom1)	p
Source vertices	4, 3
Destination vertices	2, 1
Name	pcon

7. Solver Settings

Solve using a script: off

Analysis type	Stationary
Auto select solver	On
Solver	Stationary
Solution form	Automatic
Symmetric	auto
Adaption	Off

7.1. Direct (UMFPACK)

Solver type: Linear system solver

Parameter	Value
Pivot threshold	0.1
Memory allocation factor	0.7

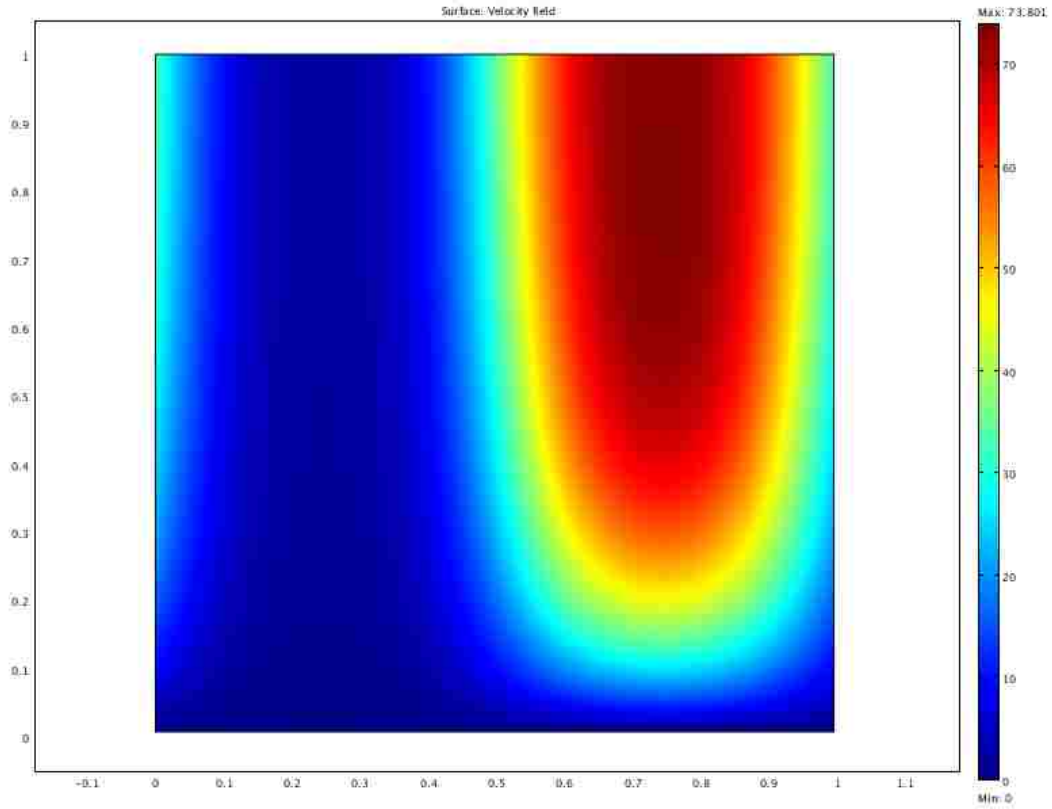
7.2. Stationary

Parameter	Value
Linearity	Automatic
Relative tolerance	1.0E-6
Maximum number of iterations	75
Manual tuning of damping parameters	Off
Highly nonlinear problem	On
Initial damping factor	1.0E-4
Minimum damping factor	1.0E-8
Restriction for step size update	10.0

7.3. Advanced

Parameter	Value
Constraint handling method	Elimination
Null-space function	Automatic
Assembly block size	5000
Use Hermitian transpose of constraint matrix and in symmetry detection	Off
Use complex functions with real input	Off
Stop if error due to undefined operation	On
Store solution on file	Off
Type of scaling	Automatic
Manual scaling	
Row equilibration	On
Manual control of reassembly	Off
Load constant	On
Constraint constant	On
Mass constant	On
Damping (mass) constant	On
Jacobian constant	On
Constraint Jacobian constant	On

8. Postprocessing



9. Equations

9.1. Boundary

Dependent variables: u, w, p

9.1.1. Boundary: 3

g coefficient

(-nx_ns+nxp)*((-2*eta_ns*uxp+p)/L)+(-nz_ns+nzp)*(-eta_ns*(uzp+wxp)/H)-diff(-2*eta_ns*uxp+p,z)*ax
(-nx_ns+nxp)*(-eta_ns*(wxp+uzp)/L)+(-nz_ns+nzp)*((-2*eta_ns*wzp+p)/H)-diff(-eta_ns*(wxp+uzp),z)*ax
0

9.2. Subdomain

Dependent variables: u, w, p

9.2.1. Subdomain: 1

Diffusion coefficient (c)

u	w	p
-diff(-2*eta_ns*uxp+p,uxp), -diff(-eta_ns*(uzp+wxp),uxp), -diff(-2*eta_ns*uxp+p,uzp), -diff(-eta_ns*(uzp+wxp),uzp)	-diff(-2*eta_ns*uxp+p,wxp), -diff(-eta_ns*(uzp+wxp),wxp), -diff(-2*eta_ns*uxp+p,wzp), -diff(-eta_ns*(uzp+wxp),wzp)	-diff(-2*eta_ns*uxp+p,pxp), -diff(-eta_ns*(uzp+wxp),pxp), -diff(-2*eta_ns*uxp+p,pzp), -diff(-eta_ns*(uzp+wxp),pzp)
-diff(-eta_ns*(wxp+upz),uxp), -diff(-2*eta_ns*wzp+p,uxp), -diff(-eta_ns*(wxp+uzp),uzp), -diff(-2*eta_ns*wzp+p,uzp)	-diff(-eta_ns*(wxp+uzp),wxp), -diff(-2*eta_ns*wzp+p,wxp), -diff(-eta_ns*(wxp+uzp),wzp), -diff(-2*eta_ns*wzp+p,wzp)	-diff(-eta_ns*(wxp+uzp),pxp), -diff(-2*eta_ns*wzp+p,pxp), -diff(-eta_ns*(wxp+uzp),pzp), -diff(-2*eta_ns*wzp+p,pzp)
0	0	0

Absorption coefficient (a)

u	w	p
-diff(-rho_ns*(u*uxp+w*uzp),u)	-diff(-rho_ns*(u*uxp+w*uzp),w)	-diff(-rho_ns*(u*uxp+w*uzp),p)
-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),u)	-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),w)	-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),p)
-diff(-divU_ns,u)	-diff(-divU_ns,w)	-diff(-divU_ns,p)

Source term (f)

F_x_ns-diff(-2*eta_ns*uxp+p,z)*ax

F_y_ns-diff(-eta_ns*(wxp+uzp),z)*ax
-divU_ns

Conservative flux convection coeff. (al)

u	w	p
-diff(-2*eta_ns*uxp+p,u), -diff(-eta_ns*(uzp+wxp),u)	-diff(-2*eta_ns*uxp+p,w), -diff(-eta_ns*(uzp+wxp),w)	-diff(-2*eta_ns*uxp+p,p), -diff(-eta_ns*(uzp+wxp),p)
-diff(-eta_ns*(wxp+uzp),u), -diff(-2*eta_ns*wzp+p,u)	-diff(-eta_ns*(wxp+uzp),w), -diff(-2*eta_ns*wzp+p,w)	-diff(-eta_ns*(wx+uz),p), -diff(-2*eta_ns*wz+p,p)
0, 0	0, 0	0, 0

Convection coefficient (be)

u	w	p
-diff(-rho_ns*(u*uxp+w*uzp),uxp), -diff(-rho_ns*(u*uxp+w*uzp),uzp)	-diff(-rho_ns*(u*uxp+w*uzp),wxp), -diff(-rho_ns*(u*uxp+w*uzp),wzp)	-diff(-rho_ns*(u*uxp+w*uzp),pxp), -diff(-rho_ns*(u*uxp+w*uzp),pzp)
-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),uxp), -diff(F_y_ns-rho_ns*(u*wxp+w*wzp),uzp)	-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),wxp), -diff(F_y_ns-rho_ns*(u*wxp+w*wzp),wzp)	-diff(F_y_ns-rho_ns*(u*wxp+w*wzp),pxp), -diff(F_y_ns-rho_ns*(u*wxp+w*wzp),pzp)
-diff(-divU_ns,uxp), -diff(-divU_ns,uzp)	-diff(-divU_ns,wxp), -diff(-divU_ns,wzp)	-diff(-divU_ns,pxp), -diff(-divU_ns,pzp)

Conservative flux source term (ga)

(-2*eta_ns*uxp+p)/L, -eta_ns*(uzp+wxp)/H
-eta_ns*(wxp+uzp)/L, (-2*eta_ns*wzp+p)/H
0, 0

10. Variables

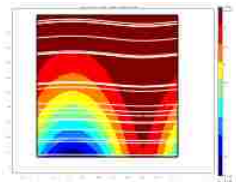
10.1. Boundary

Name	Description	Expression
K_x_ns	Viscous force per area, x component	eta_ns * (2 * nx_ns * ux+nz_ns * (uz+wx))
T_x_ns	Total force per area, x component	-nx_ns * p+2 * nx_ns * eta_ns * ux+nz_ns * eta_ns * (uz+wx)
K_z_ns	Viscous force per area, z component	eta_ns * (nx_ns * (wx+uz)+2 * nz_ns * wz)
T_z_ns	Total force per area, z component	-nz_ns * p+nx_ns * eta_ns * (wx+uz)+2 * nz_ns * eta_ns * wz

10.2. Subdomain

Name	Description	Expression
U_ns	Velocity field	sqrt(u^2+w^2)
V_ns	Vorticity	wxp-uzp
divU_ns	Divergence of velocity field	uxp+wzp
cellRe_ns	Cell Reynolds number	rho_ns * U_ns * h/eta_ns
res_u_ns	Equation residual for u	rho_ns * (u * uxp+w * uzp)+pxp-F_x_ns-eta_ns * (2 * uxx+uzz+wxz)
res_sc_u_ns	Shock capturing residual for u	rho_ns * (u * uxp+w * uzp)+pxp-F_x_ns
res_w_ns	Equation residual for w	rho_ns * (u * wxp+w * wzp)+pzp-F_y_ns-eta_ns * (wxx+uzx+2 * wzz)
res_sc_w_ns	Shock capturing residual for w	rho_ns * (u * wxp+w * wzp)+pzp-F_y_ns
beta_x_ns	Convective field, x component	rho_ns * u
beta_z_ns	Convective field, z component	rho_ns * w
Dm_ns	Mean diffusion coefficient	eta_ns
da_ns	Total time scale factor	rho_ns
taum_ns	GLS time-scale	nojac(0.5 * h/max(rho_ns * U_ns,6 * eta_ns/h))
tauc_ns	GLS time-scale	nojac(0.5 * U_ns * h * min(1,rho_ns * U_ns * h/eta_ns))

Domain transformation



1. Table of Contents

2. Model Properties

Property	Value
Model name	Domain transformation
Author	Jesse Johnson and James Fishbaugh
Company	University of Montana
Department	Computer Science
Reference	
URL	
Saved date	Apr 24, 2008 10:51:46 PM
Creation date	Jan 25, 2008 11:40:16 AM
COMSOL version	COMSOL 3.4.0.248

File name: /home/jfishbaugh/documents/research/ISMIP-HOM/expDmostrecent.mph

Application modes and modules used in this model:

- Geom1 (2D)
  - Incompressible Navier-Stokes

2.1. Model description

This model uses a coordinate transformation to allow a 1x1 to represent more complicated geometry. This model incorporates a no-slip bed as well as continuous boundaries on the sides. Equations for the surface and bed, along with the length of domain, allow the square domain to mimic the true domain. Transformed velocity and pressure derivatives are used in place of the original derivatives at Physics->Equation System->Subdomain Settings. In addition, transformed surface normals are inserted at Physics->Equation Settings->Boundary Settings.

2.2. Model Result

In the simplest case, change the bed and surface equations as well as the length variable to represent the domain you are interested in modeling. For more complicated simulations, you may have to change boundary condtions as well.

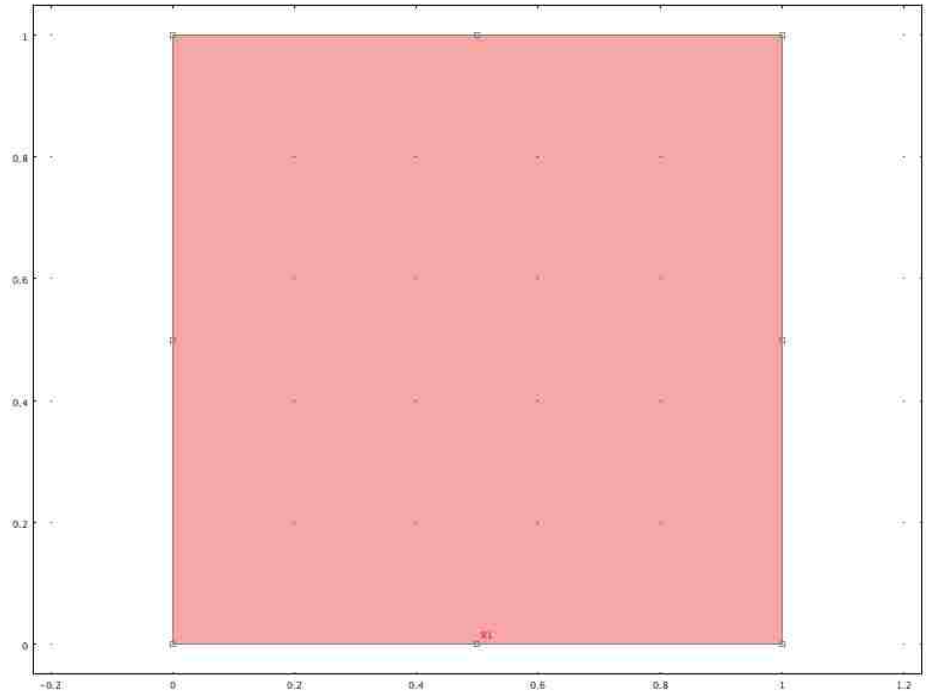
3. Constants

Name	Expression	Value	Description
L	5000		Length of domain (meters)
freq	(2*pi)/L		Frequency
alpha	0.1*(pi/180)		Surface angle with horizontal
rho_ice	910		Density of ice
g	9.81		Gravity
epsilon	1e-18		Prevents numerical explosion in eta_nonlinear if gradients all zero
A	1e-16		Ice 'hardness'
n	3		Glen's flow law exponent

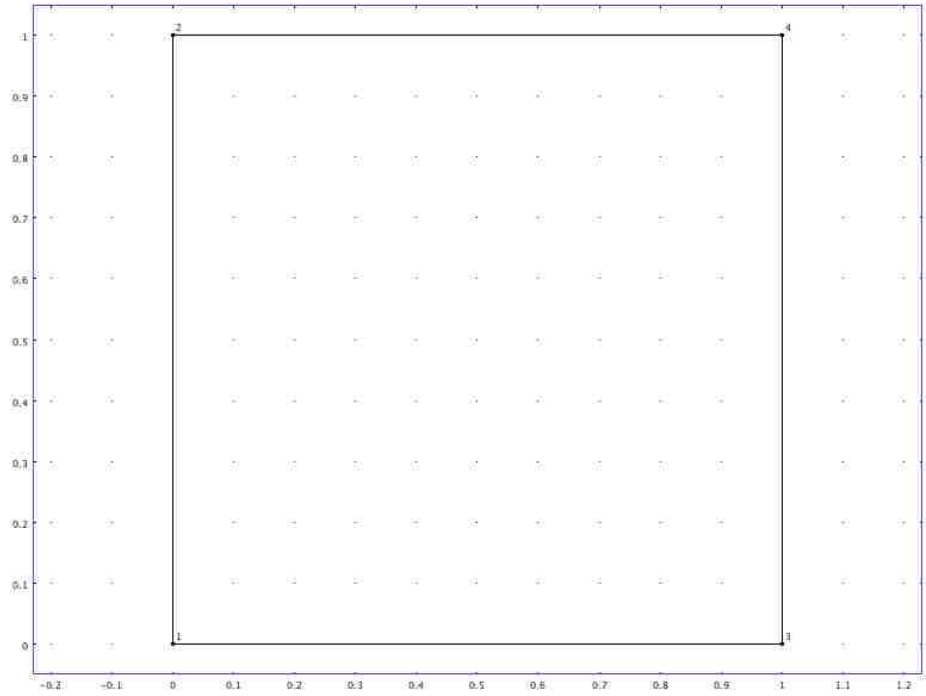
4. Geometry

Number of geometries: 1

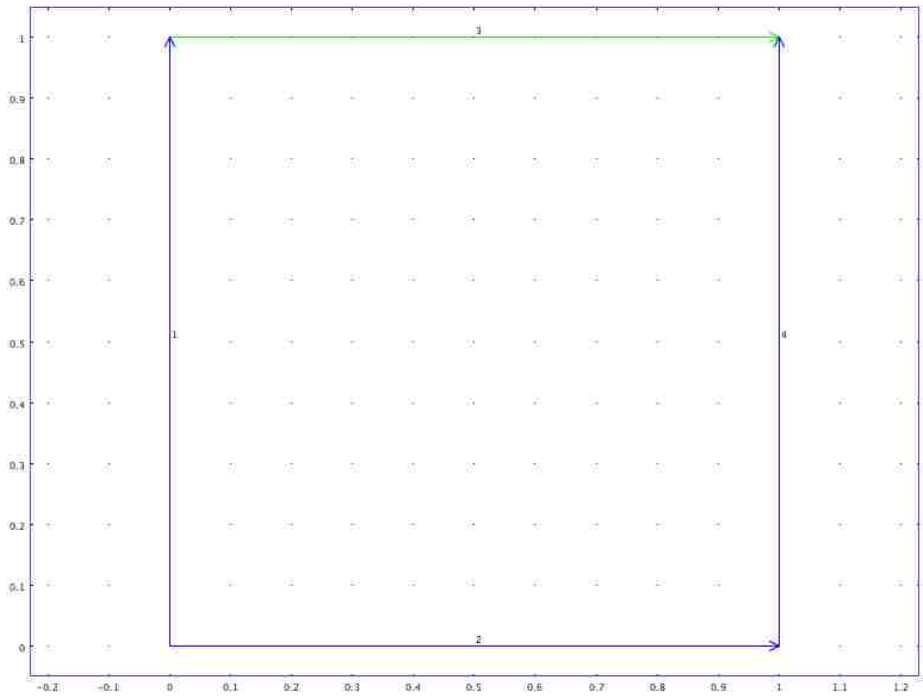
4.1. Geom1



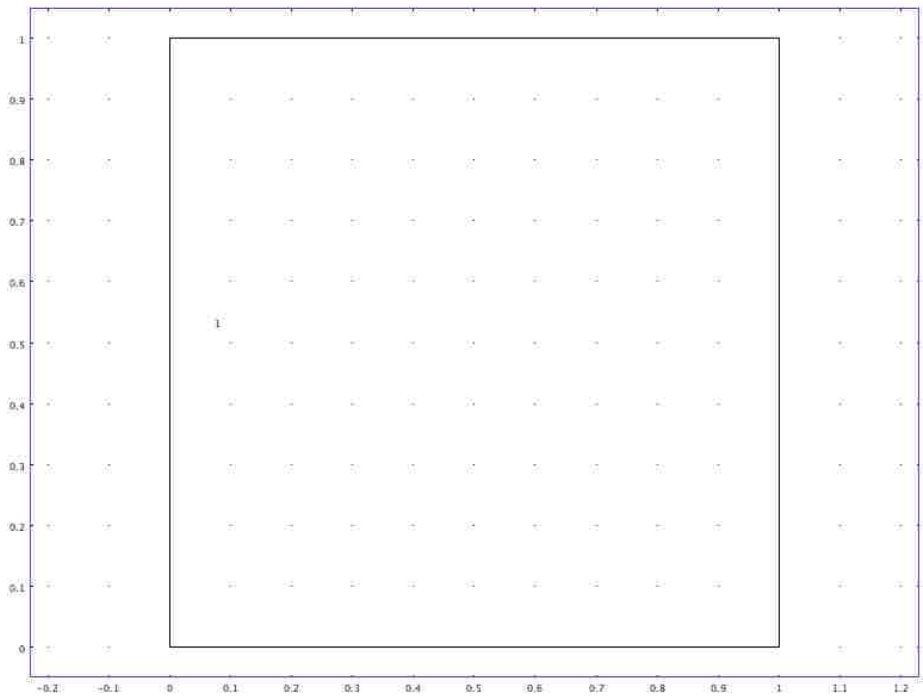
4.1.1. Point mode



4.1.2. Boundary mode



4.1.3. Subdomain mode



5. Geom1

Space dimensions: 2D

Independent variables: x, z, y

5.1. Scalar Expressions

Name	Expression	Description
bed	surf-1000	Equation that defines the bed
surf	-(x*L)*tan(alpha)	Equation that defines the surface
eta_nonlinear	1/2*A^(-1/n) * (uxp^2 + .25*(uzp+ wxp)^2+epsilon)^((1-n)/(2*n))	Viscosity of ice
H	surf-bed	Thickness of ice
p0	rho_ice*g*H*(1-z)	Initial pressure
theta	atan(-1/(diff(bed,x)))	Angle used for computing surface normals
ax	-(z*diff(surf,x)+(1-z)*diff(bed,x))/(H*L)	Transformed dz'/dx where z'=(z-bed)/(surf-bed)
uxp	ux/L+ax*uz	Transformed du/dx

wxp	wx/L+ax*wz	Transformed dw/dx
uzp	uz/H	Transformed du/dz
wzp	wz/H	Transformed dw/dz
bx	diff(bed,x)/L	bed slope
uxxp	uxx/L^2+2*ax*uxz/L+ax^2*uzz+bx*uz	Transformed d^2u/dx^2
wxxp	wxx/L^2+2*ax*wxz/L+ax^2*wzz+bx*wz	Transformed d^2w/dx^2
uzzp	uzz/H^2	Transformed d^2u/dz^2
wzzp	wzz/H^2	Transformed d^2w/dz^2
uzxp	(uxz+L*ax*uzz-uz*diff(H,x)/H)/(H*L)	Transformed d^2u/dxdz
wxzp	(wxz+L*ax*wzz-wz*diff(H,x)/H)/(H*L)	Transformed d^2w/dxdz
pxp	px/L+ax*pz	Transformed dp/dx
pzp	pz/H	Transformed dp/dz
nxp	-bx/sqrt(1+bx^2)	x component bed normal
nzp	1/sqrt(1+bx^2)	z component bed normal

5.2. Expressions

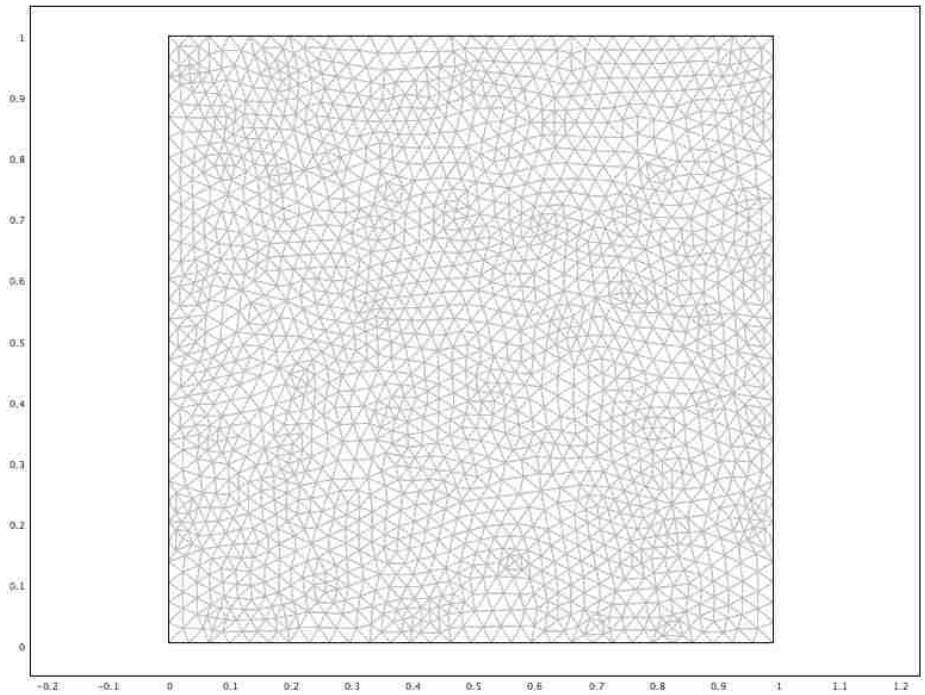
5.2.1. Boundary Expressions

Boundary	2
beta2	1000*(1+sin(freq*x*L))
taub	T_x_ns*nxp+T_z_ns*nzp
uTangent	-u*nzp+w*nxp

5.3. Mesh

5.3.1. Mesh Statistics

Number of degrees of freedom	4464
Number of mesh points	510
Number of elements	958
Triangular	958
Quadrilateral	0
Number of boundary elements	60
Number of vertex elements	4
Minimum element quality	0.692
Element area ratio	0.204



5.4. Application Mode: Incompressible Navier-Stokes (ns)

Application mode type: Incompressible Navier-Stokes

Application mode name: ns

5.4.1. Application Mode Properties

Property	Value
Default element type	Lagrange - P_2 P_1
Analysis type	Stationary

Corner smoothing	Off
Frame	Frame (ref)
Weak constraints	Off
Constraint type	Ideal

5.4.2. Variables

Dependent variables: u, w, p, nxw, nyw

Shape functions: shlag(2,'u'), shlag(2,'w'), shlag(1,'p')

Interior boundaries not active

Locked Boundaries: 2-3

5.4.3. Boundary Settings

Boundary	1, 4	2	3
Type	Open boundary	Stress	Open boundary
walltype	noslip	**mvwall**	noslip
opentype	ntotstress	**novisc**	ntotstress
stresstype	totstress	**ntotstress**	**ntotstress**

5.4.4. Subdomain Settings

Locked Subdomains: 1

Subdomain	1
Integration order (gporder)	**4 4 2**
Constraint order (cporder)	**2 2 1**
Density (rho)	**0**
Dynamic viscosity (eta)	**1e6**
Volume force, z-dir. (F_y)	**-rho_ice*g**

6. Periodic Conditions

6.1. Geom1

6.1.1. Source Boundary: 4

Name	Value
Expression	u
Transformation type	Linear
Destination, Boundary 1 (Geom1)	u
Source vertices	4, 3
Destination vertices	2, 1
Name	ucon

6.1.2. Source Boundary: 4

Name	Value
Expression	w
Transformation type	Linear
Destination, Boundary 1 (Geom1)	w
Source vertices	4, 3
Destination vertices	2, 1
Name	wcon

6.1.3. Source Boundary: 4

Name	Value
Expression	p
Transformation type	Linear
Destination, Boundary 1 (Geom1)	p
Source vertices	4, 3
Destination vertices	2, 1
Name	pcon

7. Solver Settings

Solve using a script: off

Analysis type	Stationary
Auto select solver	On
Solver	Stationary
Solution form	Automatic
Symmetric	auto
Adaption	Off

7.1. Direct (UMFPACK)

Solver type: Linear system solver

Parameter	Value
Pivot threshold	0.1
Memory allocation factor	0.7

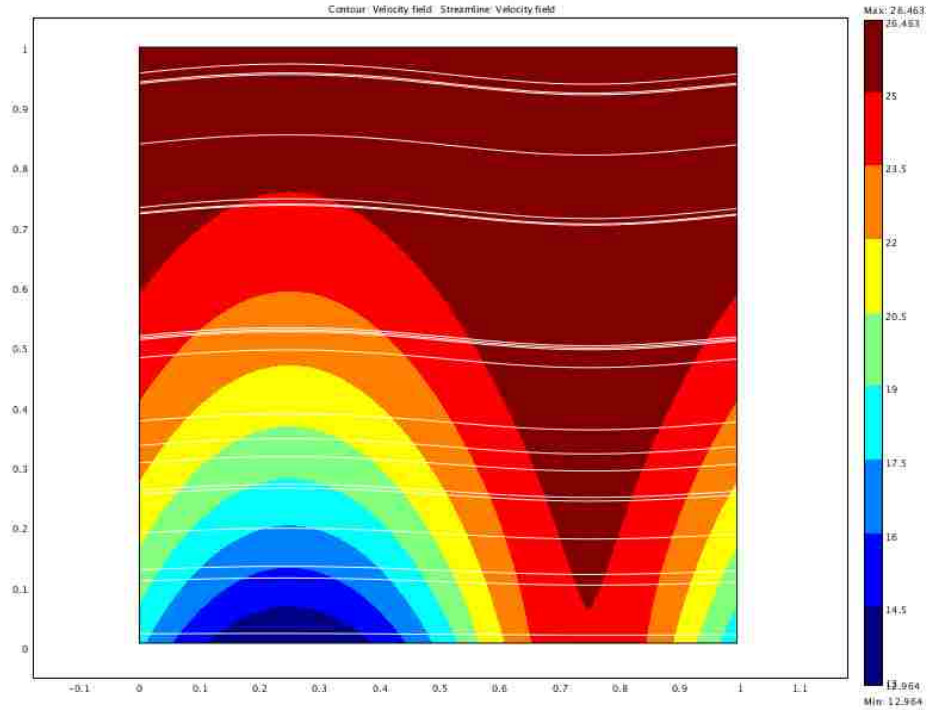
7.2. Stationary

Parameter	Value
Linearity	Automatic
Relative tolerance	1.0E-6
Maximum number of iterations	75
Manual tuning of damping parameters	Off
Highly nonlinear problem	Off
Initial damping factor	1.0
Minimum damping factor	1.0E-4
Restriction for step size update	10.0

7.3. Advanced

Parameter	Value
Constraint handling method	Elimination
Null-space function	Automatic
Assembly block size	5000
Use Hermitian transpose of constraint matrix and in symmetry detection	Off
Use complex functions with real input	Off
Stop if error due to undefined operation	On
Store solution on file	Off
Type of scaling	Automatic
Manual scaling	
Row equilibration	On
Manual control of reassembly	Off
Load constant	On
Constraint constant	On
Mass constant	On
Damping (mass) constant	On
Jacobian constant	On
Constraint Jacobian constant	On

8. Postprocessing



9. Equations

9.1. Boundary

Dependent variables: u, w, p

9.1.1. Boundary: 2

q coefficient

u	w
-diff(eta_ns*(uzp+wxp)*nz_ns/H+(nxp^2-nzp^2)*eta_ns*(uzp+wxp)-2*nx*nz*(2*eta_ns*uxp+p)-(-u*nzp+w*nxp)*beta2,u)	-diff(eta_ns*(uzp+wxp)*nz_
0	0
0	0

g coefficient

eta_ns*(uzp+wxp)/H*nz_ns+(nxp^2-nzp^2)*eta_ns*(uzp+wxp)-2*nxp*nzp*(2*eta_ns*uxp)-(-u*nzp+w*nxp)*beta2
-nz_ns*(-2*eta_ns*wzp+p)/H+(nxp^2-nzp^2)*eta_ns*(uzp+wxp)-2*nxp*nzp*(2*eta_ns*uxp)-(-u*nzp+w*nxp)*beta2
0

Constraint (constr = 0) (constr)

0
u*nxp+w*nzp
0

9.1.2. Boundary: 3

q coefficient

u	w
-diff((nxp-nx_ns)*(-2*eta_ns*uxp+p)/L-(nzp-nz_ns)*eta_ns*(uzp+wxp)/H,u)	-diff((nxp-nx_ns)*(-2*eta_ns*uxp+p)/L-(nzp-nz_ns)*eta_ns*(uzp+wxp)/H,w
-diff(-(nxp-nx_ns)*eta_ns*(wxp+uzp)/L+(nzp-nz_ns)*(-2*eta_ns*wzp+p)/H,u)	-diff(-(nxp-nx_ns)*eta_ns*(wxp+uzp)/L+(nzp-nz_ns)*(-2*eta_ns*wzp+p)/H
0	0

g coefficient

(nxp-nx_ns)*(-2*eta_ns*uxp+p)/L+(nzp-nz_ns)*(-eta_ns*(uzp+wxp)/H)
(nxp-nx_ns)*(-eta_ns*(wxp+uzp)/L)+(nzp-nz_ns)*(-2*eta_ns*wzp+p)/H
0

9.2. Subdomain

Dependent variables: u, w, p

9.2.1. Subdomain: 1

Diffusion coefficient (c)

u	w	p
-diff(-2*eta_ns*ux+p,ux), -diff(-eta_ns*(uz+wx),ux), -diff(-2*eta_ns*ux+p,uz), -diff(-eta_ns*(uz+wx),uz)	-diff(-2*eta_ns*ux+p,wx), -diff(-eta_ns*(uz+wx),wx), -diff(-2*eta_ns*ux+p,wz), -diff(-eta_ns*(uz+wx),wz)	-diff(-2*eta_ns*ux+p,px), -diff(-eta_ns*(uz+wx),px), -diff(-2*eta_ns*ux+p,pz), -diff(-eta_ns*(uz+wx),pz)
-diff(-eta_ns*(wx+uz),ux), -diff(-2*eta_ns*wz+p,ux), -diff(-eta_ns*(wx+uz),uz), -diff(-2*eta_ns*wz+p,uz)	-diff(-eta_ns*(wx+uz),wx), -diff(-2*eta_ns*wz+p,wx), -diff(-eta_ns*(wx+uz),wz), -diff(-2*eta_ns*wz+p,wz)	-diff(-eta_ns*(wx+uz),px), -diff(-2*eta_ns*wz+p,px), -diff(-eta_ns*(wx+uz),pz), -diff(-2*eta_ns*wz+p,pz)
0	0	0

Absorption coefficient (a)

u	w	p
0	0	0
-diff(F_y_ns,u)	-diff(F_y_ns,w)	-diff(F_y_ns,p)
-diff(-divU_ns,u)	-diff(-divU_ns,w)	-diff(-divU_ns,p)

Source term (f)

F_x_ns-diff(-2*eta_ns*uxp+p,z)*ax
F_y_ns-diff(-eta_ns*(wxp+uzp),z)*ax
-divU_ns

Conservative flux convection coeff. (al)

u	w	p
-diff(-2*eta_ns*ux+p,u), -diff(-eta_ns*(uz+wx),u)	-diff(-2*eta_ns*ux+p,w), -diff(-eta_ns*(uz+wx),w)	-diff(-2*eta_ns*ux+p,p), -diff(-eta_ns*(uz+wx),p)
-diff(-eta_ns*(wx+uz),u), -diff(-2*eta_ns*wz+p,u)	-diff(-eta_ns*(wx+uz),w), -diff(-2*eta_ns*wz+p,w)	-diff(-eta_ns*(wx+uz),p), -diff(-2*eta_ns*wz+p,p)
0, 0	0, 0	0, 0

Convection coefficient (be)

u	w	p
0, 0	0, 0	0, 0
-diff(F_y_ns,ux), -diff(F_y_ns,uz)	-diff(F_y_ns,wx), -diff(F_y_ns,wz)	-diff(F_y_ns,px), -diff(F_y_ns,pz)
-diff(-divU_ns,ux), -diff(-divU_ns,uz)	-diff(-divU_ns,wx), -diff(-divU_ns,wz)	-diff(-divU_ns,px), -diff(-divU_ns,pz)

Conservative flux source term (ga)

(-2*eta_ns*uxp+p)/L, -eta_ns*(uzp+wxp)/H

-eta_ns*(wxp+uzp)/L, (-2*eta_ns*wzp+p)/H
0, 0

10. Variables

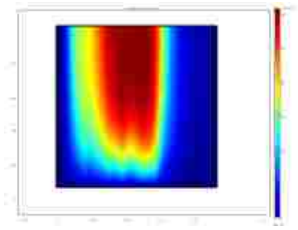
10.1. Boundary

Name	Description	Expression
K_x_ns	Viscous force per area, x component	eta_ns * (2 * nx_ns * ux+nz_ns * (uz+wx))
T_x_ns	Total force per area, x component	-nx_ns * p+2 * nx_ns * eta_ns * ux+nz_ns * eta_ns * (uz+wx)
K_z_ns	Viscous force per area, z component	eta_ns * (nx_ns * (wx+uz)+2 * nz_ns * wz)
T_z_ns	Total force per area, z component	-nz_ns * p+nx_ns * eta_ns * (wx+uz)+2 * nz_ns * eta_ns * wz

10.2. Subdomain

Name	Description	Expression
U_ns	Velocity field	sqrt(u^2+w^2)
V_ns	Vorticity	wxp-uzp
divU_ns	Divergence of velocity field	uxp+wzp
cellRe_ns	Cell Reynolds number	rho_ns * U_ns * h/eta_ns
res_u_ns	Equation residual for u	rho_ns * (u * ux+w * uz)+px-F_x_ns-eta_ns * (2 * uxx+uzz+wxz)
res_sc_u_ns	Shock capturing residual for u	rho_ns * (u * ux+w * uz)+px-F_x_ns
res_w_ns	Equation residual for w	rho_ns * (u * wx+w * wz)+pz-F_y_ns-eta_ns * (wxx+uzx+2 * wzz)
res_sc_w_ns	Shock capturing residual for w	rho_ns * (u * wx+w * wz)+pz-F_y_ns
beta_x_ns	Convective field, x component	rho_ns * u
beta_z_ns	Convective field, z component	rho_ns * w
Dm_ns	Mean diffusion coefficient	eta_ns
da_ns	Total time scale factor	rho_ns
taum_ns	GLS time-scale	nojac(0.5 * h/max(rho_ns * U_ns,6 * eta_ns/h))
tauc_ns	GLS time-scale	nojac(0.5 * U_ns * h * min(1,rho_ns * U_ns * h/eta_ns))

Domain transformation



1. Table of Contents

2. Model Properties

Property	Value
Model name	Domain transformation
Author	Jesse Johnson and James Fishbaugh
Company	University of Montana
Department	Computer Science
Reference	
URL	
Saved date	Mar 17, 2008 1:49:45 PM
Creation date	Jan 25, 2008 11:40:16 AM
COMSOL version	COMSOL 3.4.0.248

File name: /home/jfishbaugh/documents/research/ISMIP-HOM/arrollaMax.mph

Application modes and modules used in this model:

- Geom1 (2D)
  - Incompressible Navier-Stokes

2.1. Model description

This model uses a coordinate transformation to allow a 1x1 to represent more complicated geometry. This model incorporates a no-slip bed as well as continuous boundaries on the sides. Equations for the surface and bed, along with the length of domain, allow the square domain to mimic the true domain. Transformed velocity and pressure derivatives are used in place of the original derivatives at Physics->Equation System->Subdomain Settings. In addition, transformed surface normals are inserted at Physics->Equation Settings->Boundary Settings.

2.2. Model Result

In the simplest case, change the bed and surface equations as well as the length variable to represent the domain you are interested in modeling. For more complicated simulations, you may have to change boundary condtions as well.

3. Constants

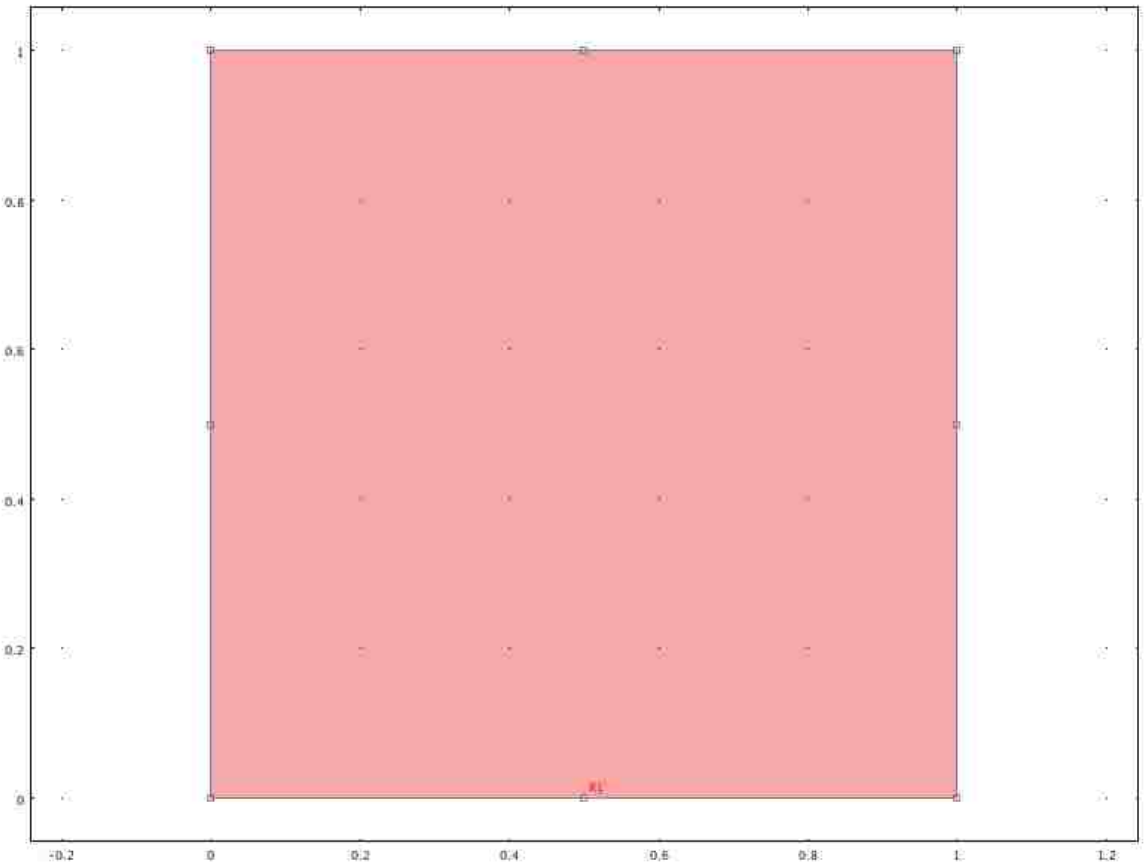
Name	Expression	Value	Description
L	5000		Length of domain (meters)
freq	(2*pi)/L		Frequency
alpha	0.5*(pi/180)		Surface angle with horizontal

rho_ice	911		Density of ice
g	9.81		Gravity
epsilon	1e-18		Prevents numerical explosion in eta_nonlinear if gradients all zero
A	1e-16		Ice 'hardness'
n	3		Glen's flow law exponent

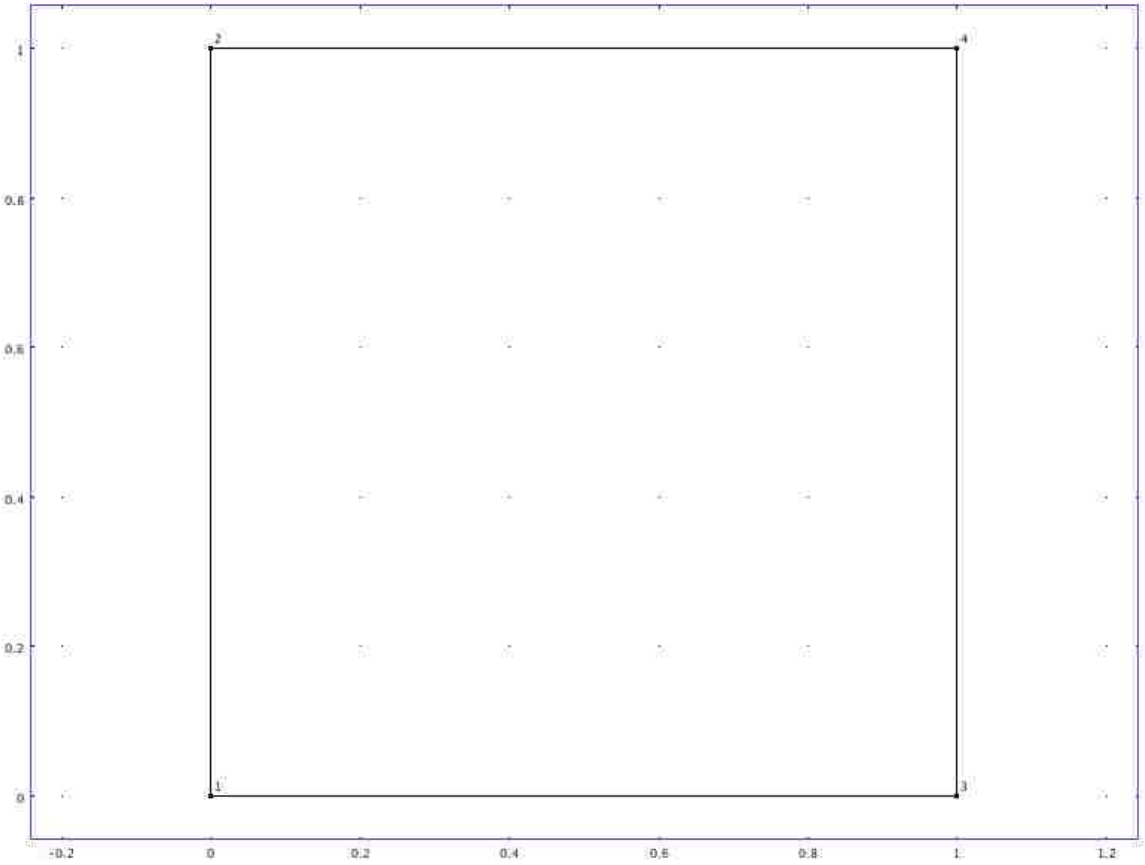
4. Geometry

Number of geometries: 1

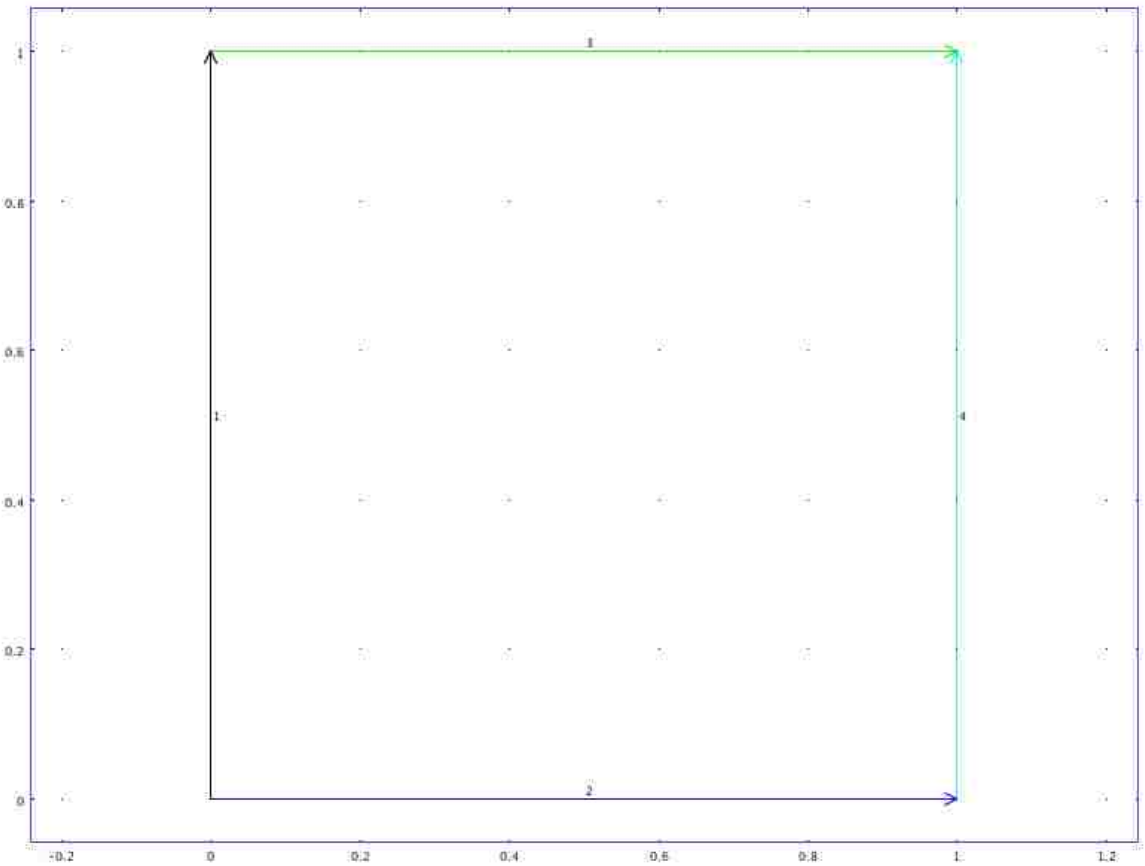
4.1. Geom1



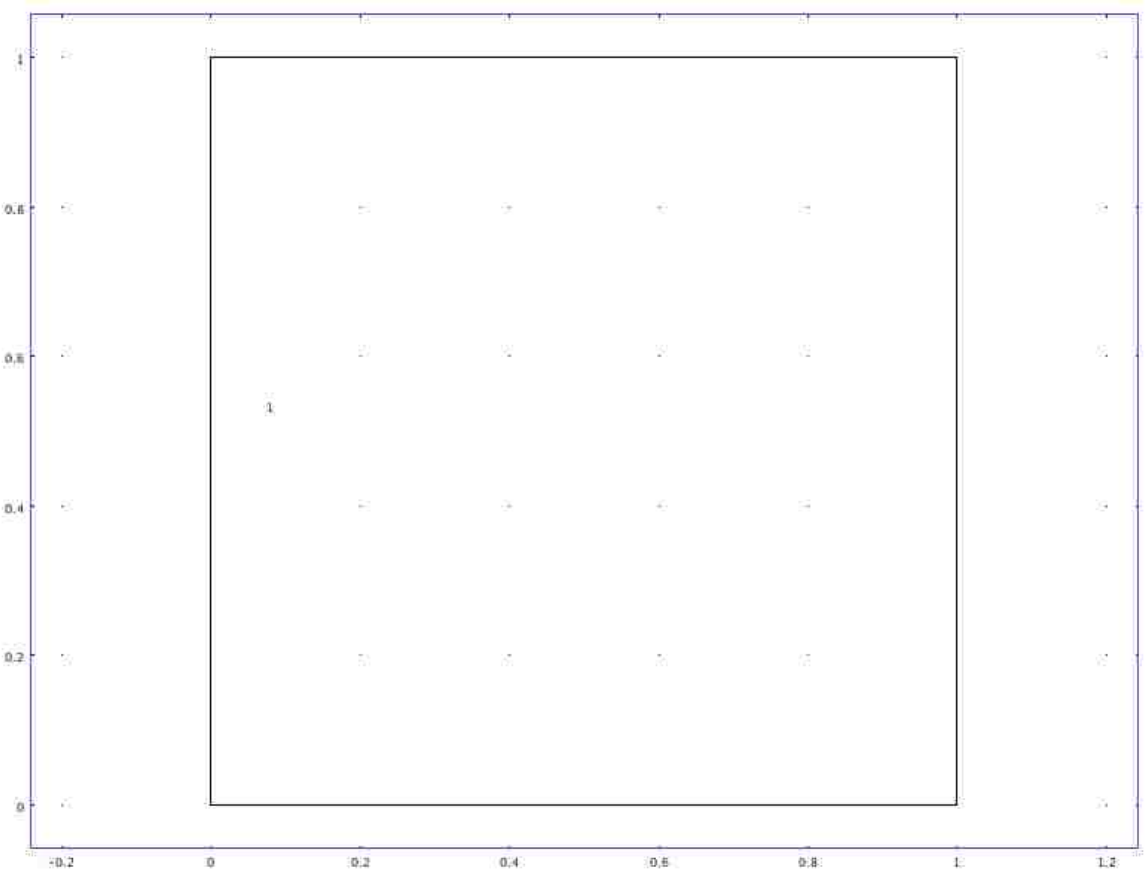
4.1.1. Point mode



4.1.2. Boundary mode



4.1.3. Subdomain mode



5. Geom1

Space dimensions: 2D

Independent variables: x, z, y

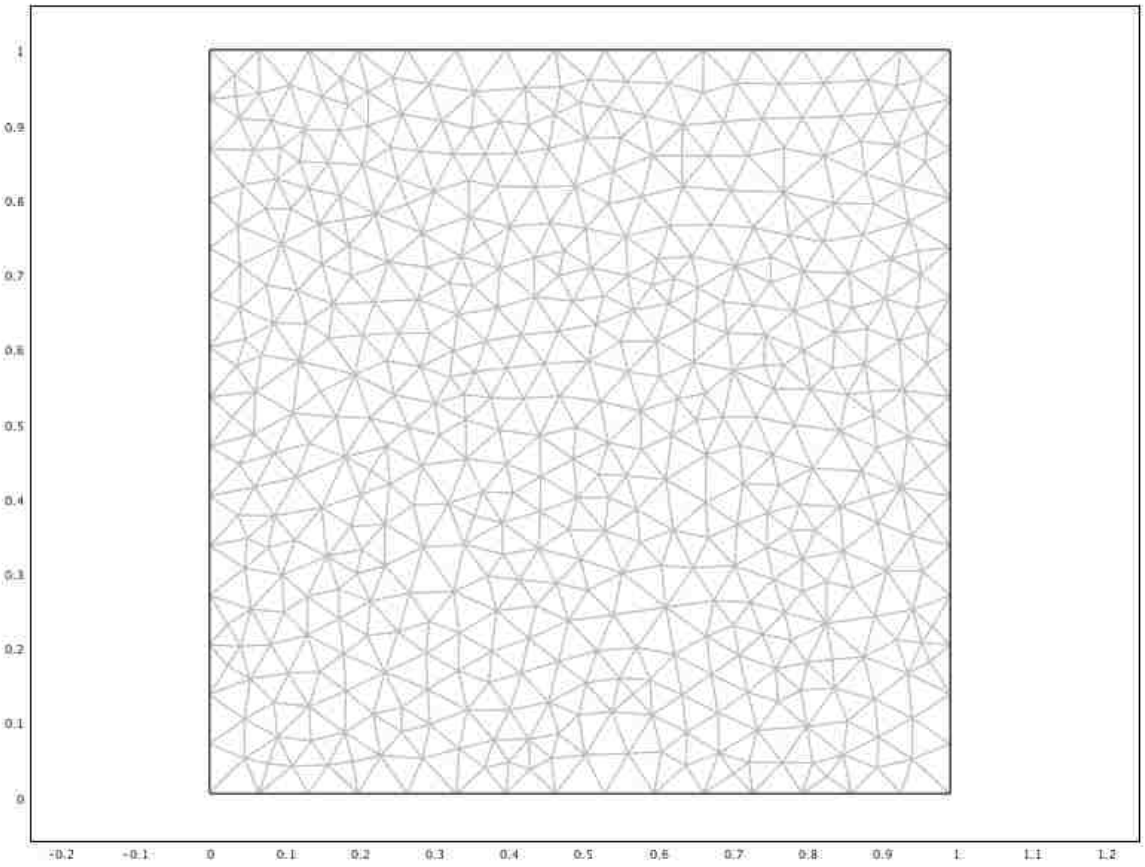
5.1. Scalar Expressions

Name	Expression	Description
bed	bedf(x*L)	Equation that defines the bed
surf	surff(x*L)	Equation that defines the surface
eta_nonlinear	1/2*A^(-1/n) * (uxp^2 + .25*(uzp+ wxp)^2+epsilon)^((1-n)/(2*n))	Viscosity of ice
H	max(surf-bed,3)	Thickness of ice
p0	rho_ice*g*H*(1-z)	Initial pressure
theta	atan(-1/(diff(surf,x)))	Angle used for computing surface normals
ax	-(z*diff(surf,x)+(1-z)*diff(bed,x))/(H*L)	Transformed dz'/dx where z'=(z-bed)/(surf-bed)
uxp	ux/L+ax*uz	Transformed du/dx
wxp	wx/L+ax*wz	Transformed dw/dx
uzp	uz/H	Transformed du/dz
wzp	wz/H	Transformed dw/dz
bx	diff(ax,x)/L+ax*diff(ax,z)	Convienence derivative with respect to x
uxxp	uxx/L^2+2*ax*uxz/L+ax^2*uzz+bx*uz	Transformed d^2u/dx^2
wxxp	wxx/L^2+2*ax*wxz/L+ax^2*wzz+bx*wz	Transformed d^2w/dx^2
uzzp	uzz/H^2	Transformed d^2u/dz^2
wzzp	wzz/H^2	Transformed d^2w/dz^2
uzxp	(uxz+L*ax*uzz-uz*diff(H,x)/H)/(H*L)	Transformed d^2u/dxdz
wxzp	(wxz+L*ax*wzz-wz*diff(H,x)/H)/(H*L)	Transformed d^2w/dxdz
pxp	px/L+ax*pz	Transformed dp/dx
pzp	pz/H	Transformed dp/dz
nxp	sin(theta)	Calculates the x surface normals
nzp	cos(theta)	Calculates the z surface normals

5.2. Mesh

5.2.1. Mesh Statistics

Number of degrees of freedom	4464
Number of mesh points	510
Number of elements	958
Triangular	958
Quadrilateral	0
Number of boundary elements	60
Number of vertex elements	4
Minimum element quality	0.692
Element area ratio	0.204



5.3. Application Mode: Incompressible Navier-Stokes (ns)

Application mode type: Incompressible Navier-Stokes

Application mode name: ns

5.3.1. Application Mode Properties

Property	Value
Default element type	Lagrange - P_2 P_1
Analysis type	Stationary
Corner smoothing	Off
Frame	Frame (ref)
Weak constraints	Off
Constraint type	Ideal

5.3.2. Variables

Dependent variables: u, w, p, nxw, nyw

Shape functions: shlag(2,'u'), shlag(2,'w'), shlag(1,'p')

Interior boundaries not active

Locked Boundaries: 1, 3-4

5.3.3. Boundary Settings

Boundary	1	2	3
Type	Wall	Wall	Open boundary
intype	uv	**p**	uv
outtype	p	p	p
opentype	ntotstress	**novisc**	ntotstress
stresstype	**ntotnflow**	totstress	**ntotstress**
Normal inflow velocity (U0in)	**0**	1	1

Boundary	4
Type	Wall
intype	uv
outtype	**ntotstress**
opentype	ntotstress
stresstype	**ntotnflow**
Normal inflow velocity (U0in)	1

5.3.4. Subdomain Settings

Locked Subdomains: 1

Subdomain	1
Integration order (gporder)	**4 4 2**
Constraint order (cporder)	**2 2 1**
Density (rho)	**rho_ice**
Dynamic viscosity (eta)	**eta_nonlinear**
Volume force, z-dir. (F_y)	**-rho_ice*g**

6. Interpolation Functions

6.1. Interpolation Function: bedf

Interpolation method: Piecewise Cubic

Data source type: File

x	f(x)

6.2. Interpolation Function: surff

Interpolation method: Piecewise Cubic

Data source type: File

x	f(x)

7. Solver Settings

Solve using a script: off

Analysis type	Stationary
Auto select solver	On
Solver	Stationary
Solution form	Automatic
Symmetric	auto
Adaption	Off

7.1. Direct (UMFPACK)

Solver type: Linear system solver

Parameter	Value
Pivot threshold	0.1
Memory allocation factor	0.7

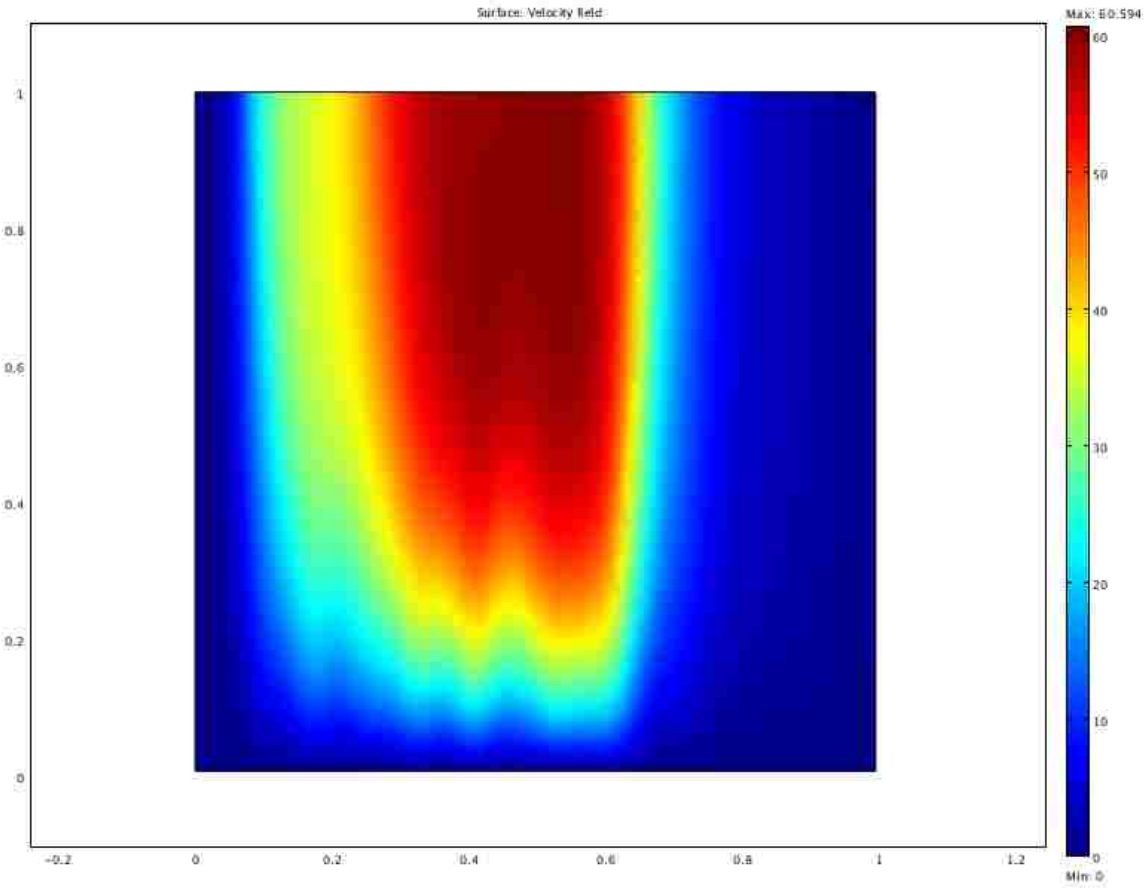
7.2. Stationary

Parameter	Value
Linearity	Automatic
Relative tolerance	1.0E-6
Maximum number of iterations	75
Manual tuning of damping parameters	Off
Highly nonlinear problem	On
Initial damping factor	1.0E-4
Minimum damping factor	1.0E-8
Restriction for step size update	10.0

7.3. Advanced

Parameter	Value
Constraint handling method	Elimination
Null-space function	Automatic
Assembly block size	5000
Use Hermitian transpose of constraint matrix and in symmetry detection	Off
Use complex functions with real input	Off
Stop if error due to undefined operation	On
Store solution on file	Off
Type of scaling	Automatic
Manual scaling	
Row equilibration	On
Manual control of reassembly	Off
Load constant	On
Constraint constant	On
Mass constant	On
Damping (mass) constant	On
Jacobian constant	On
Constraint Jacobian constant	On

8. Postprocessing



9. Equations

9.1. Boundary

Dependent variables: u, w, p

9.1.1. Boundary: 1

9.1.2. Boundary: 3

g coefficient

(nx_ns-nxp)*((-2*eta_ns*uxp+p)/L)+(nz_ns-nzp)*(-eta_ns*(uzp+wxp)/H)
(nx_ns-nxp)*(-eta_ns*(wxp+uzp)/L)+(nz_ns-nzp)*((-2*eta_ns*wzp+p)/H)
0

9.1.3. Boundary: 4

9.2. Subdomain

Dependent variables: u, w, p

9.2.1. Subdomain: 1

Diffusion coefficient (c)

u	w	p
-diff(-2*eta_ns*ux+p,ux), -diff(-eta_ns*(uz+wx),ux), -diff(-2*eta_ns*ux+p,uz), -diff(-eta_ns*(uz+wx),uz)	-diff(-2*eta_ns*ux+p,wx), -diff(-eta_ns*(uz+wx),wx), -diff(-2*eta_ns*ux+p,wz), -diff(-eta_ns*(uz+wx),wz)	-diff(-2*eta_ns*ux+p,px), -diff(-eta_ns*(uz+wx),px), -diff(-2*eta_ns*ux+p,pz), -diff(-eta_ns*(uz+wx),pz)
-diff(-eta_ns*(wx+uz),ux), -diff(-2*eta_ns*wz+p,ux), -diff(-eta_ns*(wx+uz),uz), -diff(-2*eta_ns*wz+p,uz)	-diff(-eta_ns*(wx+uz),wx), -diff(-2*eta_ns*wz+p,wx), -diff(-eta_ns*(wx+uz),wz), -diff(-2*eta_ns*wz+p,wz)	-diff(-eta_ns*(wx+uz),px), -diff(-2*eta_ns*wz+p,px), -diff(-eta_ns*(wx+uz),pz), -diff(-2*eta_ns*wz+p,pz)
0	0	0

Absorption coefficient (a)

u	w	p
-diff(-rho_ns*(u*ux+w*uz),u)	-diff(-rho_ns*(u*ux+w*uz),w)	-diff(-rho_ns*(u*ux+w*uz),p)
-diff(F_y_ns-rho_ns*(u*wx+w*wz),u)	-diff(F_y_ns-rho_ns*(u*wx+w*wz),w)	-diff(F_y_ns-rho_ns*(u*wx+w*wz),p)
-diff(-divU_ns,u)	-diff(-divU_ns,w)	-diff(-divU_ns,p)

Source term (f)

F_x_ns-ax*diff((-2*eta_ns*uxp+p),z)
F_y_ns-ax*diff((-eta_ns*(wxp+uzp)),z)
-divU_ns

Conservative flux convection coeff. (al)

u	w	p
-diff(-2*eta_ns*ux+p,u), -diff(-eta_ns*(uz+wx),u)	-diff(-2*eta_ns*ux+p,w), -diff(-eta_ns*(uz+wx),w)	-diff(-2*eta_ns*ux+p,p), -diff(-eta_ns*(uz+wx),p)
-diff(-eta_ns*(wx+uz),u), -diff(-2*eta_ns*wz+p,u)	-diff(-eta_ns*(wx+uz),w), -diff(-2*eta_ns*wz+p,w)	-diff(-eta_ns*(wx+uz),p), -diff(-2*eta_ns*wz+p,p)
0, 0	0, 0	0, 0

Convection coefficient (be)

u	w	p
-diff(-rho_ns*(u*ux+w*uz),ux), -diff(-rho_ns*(u*ux+w*uz),uz)	-diff(-rho_ns*(u*ux+w*uz),wx), -diff(-rho_ns*(u*ux+w*uz),wz)	-diff(-rho_ns*(u*ux+w*uz),px), -diff(-rho_ns*(u*ux+w*uz),pz)
-diff(F_y_ns-rho_ns*(u*wx+w*wz),ux), -diff(F_y_ns-rho_ns*(u*wx+w*wz),uz)	-diff(F_y_ns-rho_ns*(u*wx+w*wz),wx), -diff(F_y_ns-rho_ns*(u*wx+w*wz),wz)	-diff(F_y_ns-rho_ns*(u*wx+w*wz),px), -diff(F_y_ns-rho_ns*(u*wx+w*wz),pz)
-diff(-divU_ns,ux), -diff(-divU_ns,uz)	-diff(-divU_ns,wx), -diff(-divU_ns,wz)	-diff(-divU_ns,px), -diff(-divU_ns,pz)

Conservative flux source term (ga)

(-2*eta_ns*uxp+p)/L, (-eta_ns*(uzp+wxp))/H
(-eta_ns*(wxp+uzp))/L, (-2*eta_ns*wzp+p)/H
0, 0

10. Variables

10.1. Boundary

Name	Description	Expression
K_x_ns	Viscous force per area, x component	eta_ns * (2 * nx_ns * ux+nz_ns * (uz+wx))
T_x_ns	Total force per area, x component	-nx_ns * p+2 * nx_ns * eta_ns * ux+nz_ns * eta_ns * (uz+wx)
K_z_ns	Viscous force per area, z component	eta_ns * (nx_ns * (wx+uz)+2 * nz_ns * wz)
T_z_ns	Total force per area, z component	-nz_ns * p+nx_ns * eta_ns * (wx+uz)+2 * nz_ns * eta_ns * wz

10.2. Subdomain

Name	Description	Expression
U_ns	Velocity field	sqrt(u^2+w^2)
V_ns	Vorticity	wxp-uzp
divU_ns	Divergence of velocity field	uxp+wzp
cellRe_ns	Cell Reynolds number	rho_ns * U_ns * h/eta_ns
res_u_ns	Equation residual for u	rho_ns * (u * ux+w * uz)+px-F_x_ns-eta_ns * (2 * uxx+uzz+wxz)
res_sc_u_ns	Shock capturing residual for u	rho_ns * (u * ux+w * uz)+px-F_x_ns
res_w_ns	Equation residual for w	rho_ns * (u * wx+w * wz)+pz-F_y_ns-eta_ns * (wxx+uzx+2 * wzz)
res_sc_w_ns	Shock capturing residual for w	rho_ns * (u * wx+w * wz)+pz-F_y_ns
beta_x_ns	Convective field, x component	rho_ns * u
beta_z_ns	Convective field, z component	rho_ns * w
Dm_ns	Mean diffusion coefficient	eta_ns
da_ns	Total time scale factor	rho_ns
taum_ns	GLS time-scale	nojac(0.5 * h/max(rho_ns * U_ns,6 * eta_ns/h))
tauc_ns	GLS time-scale	nojac(0.5 * U_ns * h * min(1,rho_ns * U_ns * h/eta_ns))